Politecnico di Torino

Politecnico di Torino

Corso di Laurea Magistrale in

Ingegneria del Cinema e dei Mezzi di Comunicazione

A.a. 2022/2023

Sessione di Laurea Dicembre 2023

CoReDex: progetto ed implementazione di una applicazione mobile basata sulla gamification per la mappatura delle Soft Skills

Relatore:

Prof. Giovanni Malnati

Candidata:

Margherita Del Balio

Ai miei nonni, Luciana e Enzo.

Ringraziamenti

Desidero esprimere la mia gratitudine al mio relatore, il prof. Giovanni Malnati, e all'azienda Concept Reply, in particolare a Paolo Carletto, per aver reso possibile questo progetto di tesi, un vero e proprio gioiello per me.

Un ringraziamento speciale va al mio manager Stefano Ricci, che da più di tre anni mi insegna non solo tutto ciò che c'è da sapere nel mondo tecnologico, ricordandomi l'importanza di seguire la letteratura, ma anche che non esisterà mai nessuno in grado di battere il suo tiramisù. Mi hai costantemente spronata a dare il meglio di me, anche nei momenti di smarrimento, non solo ti stimo ma ti voglio bene.

Un enorme grazie ai miei compagni di progetto: Luca, Maurizio, Amedeo e Fabio, che hanno collaborato attivamente con me nella realizzazione di CoReDex, dando vita alle nostre idee.

Ame, la tua sensibilità ha aiutato a realizzare l'interfaccia utente più bella del mondo, e la tua generosità riempie le mie giornate lavorative, sei un vero amico.

Fabio, grazie per le sessioni di karaoke e per i giorni trascorsi a creare script Python e grafici colorati.

Mauri, sei come un fratello maggiore per me, la nostra telepatia è affascinante e spero che non si interrompa mai.

Luca, senza di te non esisterebbero le challenges su CoReDex e nemmeno la felicità in ufficio, riempi di energia la 2.7. Grazie per tutte le cose belle che fai, dall'amare i cani al sopportare la mia ansia.

Grazie ai miei due colleghi e amici Giulia e Strizzi: non mi stancherò mai di dirvi quanto io vi voglia bene. Giuli, sei semplicemente fantastica. Ste, grazie di nuovo per tutti i quiz che mi hai aiutata a creare.

Ringrazio tutti i colleghi che hanno reso grande CoReDex in ufficio, in particolare Amerigo e Tone.

Grazie, inoltre, a tutti gli amici che nei mesi passati mi hanno incoraggiata in ogni momento, dandomi consigli e festeggiando con me ogni piccolo traguardo, dal far funzionare il login alla creazione delle medaglie: Vittorio, Michi (grazie per avermi riportato il PC a casa in una giornata di maggio), Dario, Lucio, Liberato, Stefano, Vale, Ale, Ele (sempre voi tre), Rebecca, Rob, Michele, Nico, Giomba, Vincenzo, Rocco e chiunque abbia condiviso con me questo percorso.

Un grazie speciale va a Giulia e Andrea, amici del cuore che hanno sempre creduto in me nonostante la distanza.

Grazie ad Ale, che nell'ultimo mese mi ha ripetuto giorno e notte di non mollare, e ha festeggiato con me i miei ultimi traguardi.

Infine, grazie alla mia fantastica famiglia: mamma, babbo, Agnese, nonna Luciana, nonno Enzo, i miei cugini Sara e Lorenzo.

Mamma e babbo, avete sempre sostenuto e incoraggiato le mie scelte con un amore incondizionato, e continuate tutt'ora a farlo. Ve ne sarò sempre grata.

Agni, la tua intelligenza, bellezza e reciproca ammirazione mi rendono fiera di essere tua sorella.

Nonnini, dedico a voi tutti i miei anni di studio fuori casa, ogni esame superato, ogni parola di questa tesi e ogni riga di codice scritto, ogni traguardo raggiunto e quelli che verranno. Vi voglio bene.

Abstract

CoReDex – Concept Reply Index è un'innovativa applicazione mobile ideata all'interno di Concept Reply, un'azienda Azienda che fa parte della multinazionale Reply con sedi sparse in tutta Italia. Nata dalla necessità di rafforzare la connessione e la conoscenza reciproca tra i dipendenti, specialmente in un'era caratterizzata dal lavoro da remoto, CoReDex si ispira all'idea di un "Pokédex aziendale", trasformando ogni dipendente in un "CoReMon" da scoprire e "catturare".

L'applicazione integra elementi ludici per rendere più coinvolgente l'esperienza dei dipendenti. Tra queste, spicca la sezione "CoReGym", dove i dipendenti partecipano a sfide e quiz settimanali, accumulando punti esperienza e guadagnando medaglie. Questo approccio si arricchisce di un obiettivo strategico: attraverso meccanismi di gamification, CoReDex punta a far emergere le competenze nascoste dei dipendenti, con un focus specifico sulle "soft skills". Ogni utente può, infatti, identificare e riconoscere le competenze dei propri colleghi, contribuendo a delineare un quadro più chiaro delle capacità presenti all'interno dell'azienda.

Sviluppata con un approccio cross-platform e rilasciata esclusivamente per i dipendenti di Concept Reply, CoReDex sfrutta il potenziale del framework React Native sviluppato da Meta, adottando JavaScript come principale linguaggio di programmazione.

Questa tesi esplora ogni aspetto del processo creativo e di sviluppo di CoReDex. Dalla fase iniziale di ricerca e applicazione delle meccaniche di gamification, fino allo sviluppo dell'Esperienza Utente (UX) tramite le tecniche di User Centered Design e dell'Interfaccia Utente (UI), passando per le scelte architetturali, come

l'adozione della Clean Architecture. I capitoli tecnici si addentrano nelle specificità dello sviluppo frontend e backend, mentre altri segmenti si concentrano sull'interazione e il coinvolgimento degli utenti, sia all'interno che all'esterno dell'app. Viene poi presentata dettagliatamente ogni sezione dell'applicazione, con un focus particolare sulla sezione "CoReGym", il cuore dell'app, che attraverso sfide settimanali mappa le competenze nascoste (soft skills) dei dipendenti. Il capitolo conclusivo analizza i dati raccolti, offrendo una panoramica grafica e concettuale delle soft skills rilevate, nonché delle percezioni reciproche delle competenze tra i dipendenti.

Sommario

Indice delle Figure

Introduzione

CoReDex – Concept Reply Index è un'applicazione mobile sviluppata all'interno dell'ambiente aziendale di Concept Reply, un'azienda di medie dimensioni. Trasportando i colleghi all'interno di un mondo immaginario, quello dei "CoReMons", l'app offre loro l'opportunità di conoscersi più a fondo e mantenersi in contatto.

L'ispirazione per questa app è scaturita da un incontro informale tra alcuni colleghi e uno dei partner di Concept, durante il quale si discuteva dell'azienda e dei suoi prossimi eventi. Durante questa conversazione, è emerso un problema significativo: i dipendenti di Concept non si conoscevano tra loro. Questo problema è in parte dovuto al fatto che l'azienda ha sedi distaccate in Italia, in parte all'accentuarsi del lavoro da remoto, che ha limitato le interazioni tra colleghi a schermi virtuali su piattaforme come Microsoft Teams.

Inizialmente, si era pensato a un album di figurine come soluzione al problema. Questa idea aveva un certo fascino nostalgico, ma non si adattava perfettamente all'ambito dell'azienda, specializzata in informatica e IoT.

Alla luce della cultura aziendale, ricca di passioni "nerd", Maurizio, un collega di Concept, ha avuto un'idea brillante: perché non creare un Pokédex aziendale? È così che nasce CoReDex: un'app il cui obiettivo principale è "catturare" i colleghi presenti in ufficio, in modo simile a come si catturano i Pokémon nei giochi.

L'idea si è evoluta con l'obiettivo aggiuntivo di mappare le soft skills dei dipendenti attraverso un'analisi "dal basso". Questo processo si avvale delle dinamiche di gamification all'interno dell'applicazione per ottenere una comprensione più profonda delle capacità individuali.

Con CoReDex, Concept Reply mira a migliorare la conoscenza reciproca tra colleghi e a valorizzare le abilità individuali, in un contesto social e di gioco.

Questa tesi si dedica all'analisi approfondita del percorso di sviluppo di CoReDex, concepita e realizzata come un progetto collaborativo all'interno di Concept Reply. Il capitolo seguente offre una panoramica dettagliata sulle strategie di gamification analizzate e implementate nell'app, gettando le basi per la comprensione delle sue dinamiche e funzionalità.

1 Gamification

La gamification rappresenta un approccio innovativo che utilizza elementi e meccaniche tipiche dei giochi in contesti non ludici, come l'ambito aziendale o l'istruzione, creando una sinergia tra meccaniche, dinamiche ed emozioni.

Le meccaniche, elementi tangibili e visibili, costituiscono l'interfaccia di gioco con cui gli utenti interagiscono, e comprendono punti, badge, livelli e obiettivi (Werbach & Hunter, 2015). Queste meccaniche forniscono un feedback immediato, offrono riconoscimenti visivi per le conquiste e guidano gli utenti attraverso sfide e compiti.

Le dinamiche riguardano gli effetti psicologici, come la competizione, la collaborazione e la scoperta, che motivano gli utenti a interagire con le meccaniche (Deterding, 2012). Ad esempio, la competizione incoraggia il miglioramento continuo, mentre la collaborazione potenzia il senso di comunità e appartenenza.

Infine, le emozioni evocate durante il gioco, come il senso di conquista, la curiosità e la sorpresa, sono fondamentali per creare esperienze memorabili (Ryan, Rigby, & Przybylski, 2006). Queste emozioni, in risposta alle dinamiche e alle meccaniche, rendono l'esperienza di gamification profondamente coinvolgente.

Il principale obiettivo della gamification è trasformare attività quotidiane, talvolta meccaniche e tediose, in esperienze creative e divertenti, incentivando la partecipazione attiva e il raggiungimento di obiettivi specifici. La gamification, pertanto, non è un semplice inserimento di elementi ludici in contesti diversi, ma un'arte sofisticata che richiede un bilanciamento sapiente di vari elementi per coinvolgere profondamente gli utenti e motivarli intrinsecamente.

1.1 Numeri e cenni storici

La gamification ha vissuto una crescita significativa nel corso del tempo. Nel 1971, con la diffusione dei videogiochi, le persone hanno iniziato a sviluppare una naturale inclinazione verso il mondo del gioco, aprendo la strada a un futuro in cui il gioco sarebbe stato parte integrante delle attività lavorative e formative. [1]

All'inizio del millennio, il 70% delle aziende aveva pianificato di utilizzare la gamification per migliorare i processi aziendali entro il 2015. Tuttavia, il 2012 ha rivelato che l'80% di queste implementazioni rischiava il fallimento a causa di approcci errati. È emerso che la gamification richiedeva una comprensione approfondita del mondo dei videogiochi, altrimenti i progetti erano destinati a fallire. Pertanto, è fondamentale coinvolgere opportunamente gli utenti, offrendo opzioni e scelte durante l'esperienza e permettendo loro di progredire, visualizzando in modo tangibile i propri successi. Inoltre, il mancato coinvolgimento dei dipendenti rappresentava una sfida, con il 50% dei processi di cambiamento aziendali che falliva a causa di questa mancanza. Tuttavia, questa consapevolezza ha alimentato una crescita notevole del mercato dei videogiochi e della gamification, con una previsione di crescita del 30% tra il 2020 e il 2015 e un mercato globale di gamification che ha raggiunto i 7,717 miliardi di dollari nel 2019, con ulteriori prospettive di crescita del 30% entro il 2025.

Un esempio di progetto di gamification di successo è "Zombie's Run" [2] nel settore del fitness. In questo gioco, l'obiettivo è scappare da un'orda di zombie attraverso la corsa reale. La chiave del successo di Zombie's Run è stata l'attenzione posta sulla

[1] "History of Gamification" - https://www.growthengineering.co.uk/history-of-gamification/
[2] "Zombies, Run!", 2012

persona, mettendo il giocatore al centro dell'esperienza di gioco, evidenziando così l'importanza di coinvolgere attivamente il giocatore per ottenere risultati positivi.

1.2 Benefici della Gamification in Ambito Aziendale

Tra i vantaggi dell'usare la gamification in contesti aziendali ci sono:

1. **Aumento dell'Engagement**: la gamification, attraverso meccaniche come punti, badge e leaderboards, stimola l'engagement dei dipendenti, incoraggiando la partecipazione attiva e regolare alle attività aziendali. Questo risultato è stato confermato da diverse ricerche che mostrano un aumento significativo dell'interazione e dell'impegno in ambienti gamificati (Hamari, Koivisto, & Sarsa, 2014).
2. **Promozione dell'Apprendimento e della Formazione**: le tecniche di gamification possono facilitare e rendere più efficaci i programmi di formazione e lo sviluppo delle competenze, creando un ambiente di apprendimento coinvolgente e interattivo. La gamification rende il processo di apprendimento meno monotono e più stimolante (Caponetto, Earp, & Ott, 2014).
3. **Incremento della Produttività**: integrando elementi ludici nelle attività quotidiane, i dipendenti sono spinti a migliorare la loro produttività ed efficienza. La competizione amichevole e il riconoscimento delle prestazioni possono servire come incentivi potenti per migliorare (Sailer, Ulrich Hense, Mayr, & Mandl, 2017).
4. **Miglioramento della Collaborazione**: giochi che premiano la collaborazione e il lavoro di squadra possono migliorare le relazioni tra i dipendenti e

promuovere un ambiente di lavoro più coeso e supportivo (Zichermann & Cunningham, 2011).

5. **Identificazione e Valorizzazione delle Competenze**: la gamification permette di identificare e valorizzare le competenze dei dipendenti in modo innovativo e coinvolgente, facilitando il riconoscimento dei talenti nascosti e delle abilità di ciascun membro del team (Farzan, DiMicco, Millen, & Dugan, 2008).
6. **Rafforzamento della Cultura Aziendale**: creando un senso di appartenenza e condivisione degli obiettivi, la gamification può contribuire a definire e rafforzare la cultura aziendale, migliorando il senso di identità e di appartenenza tra i dipendenti (Mollick & Rothbard, 2014).

1.3 Il ruolo del Gamification Designer

Il ruolo del gamification designer è una sfida complessa, poiché richiede di integrare elementi di divertimento in contesti non ludici. Questo professionista deve formare le persone a guardare le dinamiche lavorative e dei processi da una prospettiva diversa che includa elementi ludici, sfidando le concezioni tradizionali.

Il gamification designer deve adattarsi a un ruolo impegnativo, ovvero quello di considerare il processo che deve essere gamificato e posizionare le tecniche di gioco allo stesso livello di importanza del processo stesso. Tuttavia, questa è una sfida particolarmente complessa, soprattutto in un contesto aziendale in cui è difficile conciliare il divertimento con le responsabilità lavorative. Spesso, le proposte di elementi divertenti vengono viste come futile perdita di tempo.

In questo contesto, il gamification designer deve considerare i partecipanti come veri e propri giocatori, tenendo sempre al centro dell'attenzione e dell'esperienza di

gioco la sua audience. Gli utenti devono comprendere l'importanza degli obiettivi stabiliti e devono avere la libertà di scegliere le azioni da intraprendere all'interno del sistema di gioco. In questo coinvolgimento, l'obiettivo è far vivere ai partecipanti l'esperienza ottimale, raggiungendo lo stato di "flow", in cui sono bilanciati stress e noia, fornendo un'esperienza con una difficoltà via via crescente in relazione alle abilità acquisite dall'utente.

Il motivo principale per cui si utilizza la gamification in determinati processi è che un gioco consente di concentrare le proprie energie in modo positivo su qualcosa in cui si è bravi e che piace. In altre parole, il gioco contrasta emotivamente la depressione.

Nel contesto di CoReDex, questa strategia si manifesta in modo brillante per migliorare il coinvolgimento e la conoscenza tra i colleghi all'interno dell'azienda, trasformando l'ambiente aziendale nell'affascinante universo dei "CoReMons".

1.4 Meccaniche, Dinamiche ed Emozioni

Approfondiamo i concetti di meccaniche, dinamiche ed emozioni in ambito di gamification (Mazzaglia, 2021).

1.4.1 Meccaniche

Le meccaniche sono gli elementi tangibili e visibili che definiscono l'interfaccia del gioco. Le meccaniche sono ciò che gli utenti vedono e con cui interagiscono direttamente. Alcune delle meccaniche più comuni includono:

- **Punti**: sono una rappresentazione numerica di un successo o di una conquista. Offrono un riscontro immediato all'utente e possono indicare progresso, successo o potere.
- **Badge**: questi riconoscimenti visivi fungono da testimonianza delle competenze acquisite e delle sfide superate. Sono simili ai trofei nei giochi o alle medaglie nei contesti sportivi.
- **Missioni e Obiettivi**: compiti o sfide che gli utenti devono completare e che forniscono una direzione chiara su ciò che devono fare, introducendo spesso una struttura narrativa. Gli obiettivi giornalieri o settimanali creano engagement a breve termine, offrendo agli utenti sfide da completare in un tempo limitato. Questo genera un senso di urgenza e attira l'utente a interagire con l'applicazione o il servizio regolarmente (Danny, 2013).
- **Classifiche**: mostrano una comparazione delle prestazioni tra i giocatori.
- **Livelli**: indicano progressi o avanzamenti attraverso l'esperienza di gioco. Man mano che l'utente accumula esperienza (o punti), può avanzare di livello, ottenendo riconoscimenti, poteri o responsabilità aggiuntive. Questa meccanica incentiva gli utenti a perseguire obiettivi a lungo termine e a rimanere impegnati nell'esperienza (Hamari, Koivisto, & Sarsa, 2014).
- **Narrativa**: incorporare una storia o una narrativa nel sistema di gamification può aumentare l'immersione e l'identificazione con l'esperienza. Una trama avvincente può motivare gli utenti a proseguire per scoprire come si sviluppa la storia (Nicholson, 2014).

1.4.2 Dinamiche

Mentre le meccaniche sono ciò che vediamo, le dinamiche sono ciò che sentiamo. Queste sono le forze motivazionali intrinseche che spingono gli utenti a interagire

con le meccaniche. Le dinamiche sono gli effetti psicologici sottostanti che la gamification cerca di stimolare. Alcune delle dinamiche più frequenti sono:

- **Competizione**: il desiderio di essere il migliore o di superare gli altri. La rivalità tra i giocatori può spingere alla crescita e al miglioramento continuo.
- **Collaborazione**: lavorare insieme ad altri e condividere risorse o informazioni per raggiungere un obiettivo comune. Può aumentare il senso di comunità.
- **Scoperta**: l'esplorazione può guidare l'utente attraverso la comprensione di nuove aree e concetti, stimolando la curiosità e il desiderio di apprendimento.
- **Raccolta**: l'impulso di collezionare oggetti o completare insiemi.

1.4.3 Emozioni

Le emozioni sono ciò che un giocatore sperimenta durante il gioco: possono essere sia positive sia negative e sono fondamentali per rendere un'esperienza memorabile.

Quando le meccaniche e le dinamiche sono ben bilanciate, gli utenti sperimentano emozioni positive, come:

- **Conquista**: la soddisfazione di raggiungere un obiettivo. Permette di sentirsi competenti, riuscendo a superare una o più sfide. È una delle emozioni più gratificanti nell'esperienza di gamification.
- **Curiosità**: l'interesse a sapere di più e ad esplorare. Si tratta di un elemento essenziale per mantenere l'utente impegnato ed esplorare nuovi ambiti.
- **Sorpresa**: emozione che nasce da reazioni inattese che rendono l'esperienza più avvincente.

- **Appartenenza**: il senso di far parte di una comunità o di un gruppo. È un'emozione che nasce quando la dinamica della "collaborazione" è ben sviluppata nell'esperienza.

1.5 Meccaniche di Gamification in CoReDex

CoReDex utilizza diverse meccaniche di gamification per coinvolgere e motivare i suoi utenti. Ecco un'analisi delle principali meccaniche applicate:

1. **Punti Esperienza (XP points) e livelli**: un elemento centrale dell'app sono i punti esperienza, che gli utenti guadagnano catturando colleghi CoReMons e completando sfide nella CoReGym (la sezione palestra dell'app). Questi XPs rappresentano l'attività e l'interazione all'interno dell'app e motivano gli utenti a partecipare attivamente. Guadagnando punti esperienza, il livello degli utenti aumenta.
2. **Obiettivi giornalieri/missioni**: per aumentare l'interattività e l'elemento di divertimento, CoReDex presenta quiz settimanali che sfidano gli utenti e mettono alla prova le loro conoscenze. Queste challenges offrono un'opportunità per accumulare XP aggiuntivi, crescere di livello e guadagnare medaglie.
3. **Badge e Medaglie**: come nei giochi Pokémon, gli utenti di CoReDex possono guadagnare badge e medaglie per compiere determinate azioni o raggiungere traguardi specifici. Questi riconoscimenti visivi fungono da segni di prestigio e incoraggiano il completamento di obiettivi.
4. **Leaderboard**: la classifica degli utenti con il punteggio XP più alto crea una competizione amichevole tra i colleghi. Questo elemento di competizione

spinge gli utenti a partecipare attivamente all'app e a cercare di guadagnare la posizione di vertice nella classifica.

5. **Narrativa**: CoReDex immerge gli utenti all'interno di un universo specifico, che prende spunto da altri universi noti alla maggior parte dei giocatori (Pokémon, Digimon), permettendo di sfruttare con vantaggio alcuni bias mentali. L'esperienza di engagement prevede un video in cui la rappresentazione illustrata di uno dei Partner di Concept dà il benvenuto al futuro giocatore all'interno di questo mondo, e lo sfida ad avventurarcisi all'interno. Un altro aspetto fornito dall'app è la possibilità per gli utenti di personalizzare il proprio profilo, specificando ad esempio il tipo di allenatore CoReMon e il colore preferito. Questa personalizzazione è ispirata all'universo Pokémon e consente agli utenti di esprimere la propria individualità, riflettendosi nella UI generale dell'app in modo coerente con la narrativa che si vuole offrire.

In definitiva, CoReDex trasforma l'esperienza lavorativa quotidiana in un'avventura coinvolgente, incoraggiando il coinvolgimento attivo dei colleghi, la valorizzazione delle loro abilità e la creazione di connessioni significative tra di loro.

Oltre all'obiettivo più ovvio di catturare tutti i colleghi e completare il CoReDex, l'app promuove l'obiettivo più profondo di scoprire e valorizzare le competenze uniche dei colleghi all'interno dell'azienda, spingendo ogni utente ad attribuire competenze specifiche ai propri colleghi, rivelando e valorizzando le abilità nascoste di ciascuno all'interno dell'azienda.

1.6 Gamification e Serious Games

La gamification e i serious games costituiscono due lati della stessa medaglia nell'incorporazione di elementi ludici in contesti non prettamente giocosi. Questi due concetti vengono adoperati per generare esperienze accattivanti, incentivare l'apprendimento e indurre comportamenti determinati.

Clark Abt, negli anni '70, fu il primo a introdurre il termine "serious game". Nel suo libro omonimo, Abt descrive i serious games come "*attività con uno scopo educativo esplicito e attentamente studiato, non pensate per essere giocate principalmente per divertimento.*"

In queste attività, *"la libertà sperimentale ed emotiva del gioco attivo"* è abbinata con *"la serietà del pensiero e i problemi per cui costruirlo*" (Abt, 1987).

Esistono differenze sostanziali tra gamification e serious games: la prima integra degli elementi tipicamente associati ai giochi, quali punti, medaglie e classifiche, all'interno di attività tipicamente non ludiche, con lo scopo di intensificarne l'engagement e motivare gli utenti a svolgere tali attività. In contrasto, i serious games sono vere e proprie applicazioni ludiche spesso adottate in contesti educativi, aziendali o formativi, mirate a facilitare l'apprendimento di nozioni e lo sviluppo di competenze specifiche mediante la simulazione di scenari reali in un ambiente sicuro e monitorato.

CoReDex implementa una forma di gamification che può essere interpretata come un adattamento "soft" dei serious games. Se i serious games sono esperienze ludiche complesse e immersive con finalità educative o professionali ben definite, la gamification operata da CoReDex sfrutta singoli elementi di gioco per perseguire finalità analoghe in termini di mappatura delle competenze nascoste dei dipendenti.

Inoltre, l'intersezione di gamification e serious games nel contesto di applicazioni come CoReDex rivela notevoli opportunità di creare sinergie efficaci. Potenzialmente, future versioni dell'app potrebbero integrare serious games finalizzati allo sviluppo di competenze sia tecniche che trasversali (hard e soft skills), o alla formazione generale dei dipendenti in ambiti tecnologici. Un'integrazione di questo tipo potrebbe arricchire ulteriormente l'esperienza di apprendimento e sviluppo dei dipendenti di Concept Reply, rendendola più profonda e stratificata.

1.7 Considerazioni Etiche della Gamification

La gamification in ambito aziendale, mentre presenta numerosi benefici, solleva anche importanti considerazioni etiche che devono essere affrontate con attenzione.

La partecipazione a sistemi gamificati in azienda dovrebbe essere volontaria. Inoltre, è fondamentale rispettare la privacy dei dipendenti, proteggendo le informazioni sensibili e assicurandosi che i dati raccolti durante l'esperienza gamificata non siano utilizzati in modo inappropriato.

Per quanto riguarda le attività di gamification, queste devono essere progettate in modo equo per evitare discriminazioni e garantire che tutti i dipendenti, indipendentemente dalle loro abilità o conoscenze pregresse, abbiano le stesse opportunità di successo (Kim, 2015).

In CoReDex tutti hanno le stesse opportunità di successo, proponendo quiz e challenges all'altezza di ogni dipendente. I dati personali, quali nome, cognome, e-mail e foto, vengono protetti da sistemi di autenticazione e di crittograf

2 UI, UX e Interaction Design

2.1 Principi di UI e UX

Nel contesto di CoReDex, sia l'Interfaccia Utente (UI) che l'Esperienza Utente (UX) rivestono un ruolo cruciale. La UI non è soltanto l'interfaccia visiva attraverso la quale gli utenti interagiscono con l'applicazione; essa rappresenta anche il meccanismo principale che facilita le interazioni dell'utente con il sistema. D'altra parte, la UX è una mappa complessiva del percorso dell'utente, garantendo un'esperienza che sia fluida e gratificante dall'inizio alla fine.

L'UX comprende l'intero arco dell'esperienza di un utente con un sistema, prodotto o servizio. Questo inizia dalla fase pre-utilizzo, quando gli utenti si predispongono mentalmente per le aspettative sul prodotto; passa attraverso la fase di utilizzo, in cui gli utenti elaborano le sensazioni provate durante l'uso in cui l'obiettivo è raggiungere i risultati desiderati senza difficoltà; conclude con la fase post-utilizzo, caratterizzata dalle reazioni e risposte emotive degli utenti nell'utilizzo del prodotto, le quali influenzano il livello di soddisfazione ottenuto nel raggiungimento di specifici obiettivi.

Secondo Nielsen e Norman, l'Esperienza Utente (UX) comprende l'intero spettro dell'interazione da parte dell'utente con il prodotto, dall'anticipazione dell'uso fino al ricordo delle sensazioni provate durante l'utilizzo. Un utente potrebbe raggiungere gli obiettivi prefissati con un prodotto o servizio, ma se l'esperienza non è positiva, emergono frustrazione e insoddisfazione, che potrebbero condurre all'abbandono del prodotto stesso.

L'usabilità, intesa come facilità d'uso del prodotto, rappresenta un obiettivo primario della UX, manifestandosi in modo evidente durante l'interazione con il

prodotto. La sua definizione formale è la seguente: "*il grado in cui un prodotto può essere usato da particolari utenti per raggiungere certi obiettivi con efficacia, efficienza, soddisfazione in uno specifico contesto d'uso*"[3]; in sostanza, essa stabilisce il livello di semplicità e gratificazione associato all'interazione umana con uno strumento.

La progettazione centrata sull'utente (User Centered Design o UCD) pone l'usabilità come meta principale, integrando le necessità, le aspettative e i limiti degli utenti in ogni fase del ciclo di sviluppo del software. L'UCD è un processo iterativo che include fasi di pianificazione, analisi del contesto, definizione dei requisiti, sviluppo (con la creazione di modelli, simulazioni e prototipi) e valutazione continua (monitorando il prodotto a lungo termine, per esempio attraverso l'analisi dei dati d'uso).

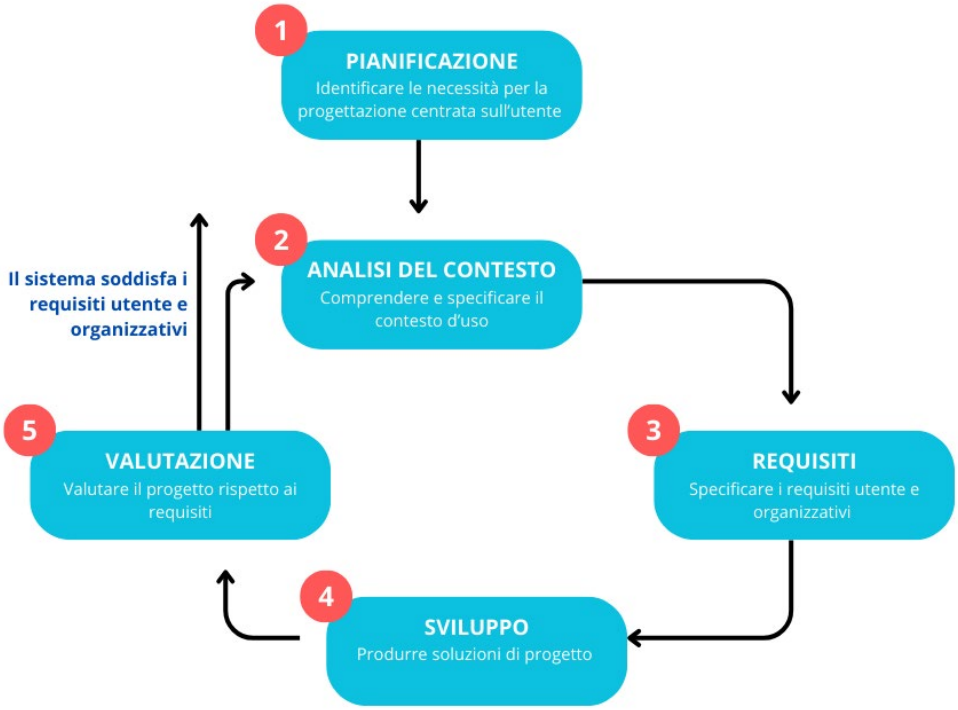

Figura 2.1 - Ciclo UCD

[3] Usabilità, Def. - ISO 9241-210:2010

Nel ciclo UCD, l'utente diventa protagonista nelle fasi di **analisi**, **sviluppo** e **valutazione**, svolgendo un ruolo chiave nella definizione dei requisiti, nella sperimentazione dei prototipi e contribuendo attivamente con strumenti appropriati.

Nel contesto specifico di CoReDex, la progettazione della UX ha adottato l'approccio UCD. Gli utenti finali, familiarizzati con la cultura nerd e appassionati di videogiochi, sono stati coinvolti sin dalla prima versione dell'app, lanciata a giugno 2023. Essi hanno avuto l'opportunità di testare il sistema, fornire feedback costruttivi, segnalare bug e problemi e, alcuni hanno anche contribuito attivamente al miglioramento e sviluppo dell'app, proponendo nuovi quiz, sfide o suggerendo future sezioni e funzionalità.

L'usabilità, insieme ad altri obiettivi come accessibilità, performance ed estetica, è facilitata da un design dell'interfaccia utente (UI) ben eseguito.

La UI, composta da schermate, pagine e vari punti di interazione visiva, guida gli utenti attraverso l'interfaccia del prodotto in modo intuitivo. L'User Interface Design punta a creare un'interfaccia che non solo sia esteticamente piacevole ma anche funzionale ed efficiente. Ciò viene realizzato attraverso un uso accurato degli elementi visivi, interattivi e spaziali, compresi colori, immagini, tipi di carattere, icone, spaziature e layout, contribuendo anche a formare un'interfaccia coesa e intuitiva.

In sintesi, mentre l'UX Design è responsabile del funzionamento complessivo del prodotto e dell'esperienza complessiva dell'utente, l'UI Design si concentra sull'aspetto visuale e sull'estetica dell'interfaccia del prodotto.

In termini semplici, l'UX si occupa di "come si sente" l'utente nell'usare il prodotto, mentre la UI si occupa di "come esso appare". Entrambi gli aspetti sono indispensabili per il successo del prodotto e devono lavorare in maniera sinergica.

L'integrazione armoniosa tra UI e UX in CoReDex è stata guidata dai principi chiave di usabilità e dai punti chiave dell'User Centered Design (UCD), assicurando un prodotto che sia non solo visivamente accattivante ma anche intuitivo e soddisfacente da usare.

2.2 UI in CoReDex

In CoReDex, la cura dedicata all'interfaccia utente (UI) è evidente. L'interfaccia si distingue per la sua intuitività, con un layout essenziale e chiaramente organizzato che facilita la navigazione. Le schede dei CoReMon, ad esempio, sono disposte in modo logico, rendendo l'accesso e l'interpretazione da parte dell'utente fluidi e intuitivi. La possibilità di personalizzare il profilo utente aggiunge valore, permettendo a ciascuno di esprimere la propria identità all'interno dell'applicazione.

I colori selezionati per l'interfaccia includono varie sfumature di rosso, grigio e bianco. Le schede CoReMon, in particolare, utilizzano colori ispirati ai tipi di Pokémon presenti nel celebre franchise, contribuendo a creare un'atmosfera familiare per gli appassionati. Il carattere tipografico sans-serif è stato scelto per garantire leggibilità e un'estetica contemporanea e pulita.

In termini di **Architettura dell'Informazione** (IA), l'app è stata strutturata in modo da guidare intuitivamente l'utente attraverso le diverse funzionalità. L'IA categorizza e organizza le informazioni in modo chiaro e gerarchico, delineando gruppi e sottogruppi di dati (L. Rosenfeld, Morville, & Arango, 2015).

L'Architettura dell'Informazione rappresenta una disciplina focalizzata sulla disposizione e l'ordinamento dei contenuti in ambienti digitali, quali siti web e applicazioni. L'intento principale è facilitare l'orientamento dell'utente, permettendogli di sfruttare le funzionalità del sistema in modo ottimale. Questo si traduce in una riduzione della complessità percepita e un incremento dell'usabilità del prodotto digitale.

In CoReDex, l'approccio adottato per l'IA è di tipo gerarchico, il quale stabilisce una chiara relazione tra elementi principali e secondari all'interno dell'applicazione.

L'architettura dell'informazione si distingue nettamente dalla strategia di navigazione impiegata. La navigazione, infatti, agevola lo spostamento dell'utente da una sezione all'altra dell'applicazione, offrendo percorsi intuitivi senza aderire rigidamente a un modello gerarchico predeterminato. Questa dinamica consente una fruizione flessibile e user-friendly del sistema, guidando l'utente attraverso le sezioni con facilità e coerenza.

CoReDex è articolata in cinque sezioni principali, rappresentate da altrettante icone nel menu di navigazione situato nella parte inferiore dell'app (bottom bar).

Figura 2.2 - Bottom Bar di CoReDex

All'interno di queste sezioni, si possono trovare sottosezioni aggiuntive. Per quanto riguarda la navigazione, ogni sezione e sottosezione offre bottoni e collegamenti ad altre aree dell'app, anche se non direttamente correlate alla sezione principale in cui l'utente si trova. Questa struttura consente di accedere a punti focali specifici dell'app da diversi "access points", tutti logicamente interconnessi.

Lo spazio bianco, o "spazio negativo", è stato utilizzato strategicamente per evidenziare bottoni, call-to-action e altri elementi chiave, come le schede dei CoReMons e il codice QR nella sezione "Meet a Colleague". Questo utilizzo consapevole dello spazio contribuisce a creare un senso di ordine e coerenza visiva, facilitando l'identificazione e il raggruppamento logico degli elementi.

Infine, implementando la legge di Pareto (80/20)[4], secondo la quale l'80% degli utenti utilizzerà solo il 20% dell'app e delle sue funzionalità, abbiamo identificato e reso facilmente accessibili le funzioni e i contenuti principali dell'app, che la maggior parte degli utenti utilizzerà. Dopo la schermata iniziale (splash screen), l'utente viene immediatamente indirizzato verso la sezione CoReMons, che evidenzia l'obiettivo primario dell'app: "catturare" il maggior numero possibile di colleghi.

2.3 Interaction Design

L'Interaction Design (IxD) in CoReDex svolge un ruolo vitale nel garantire un'esperienza utente fluida e intuitiva. L'Interaction Design, come definito da Cooper, è "*la pratica di progettare prodotti digitali interattivi, ambienti, sistemi e servizi*" (Cooper, 2014).

L'Interaction Design, in altre parole, si configura come una disciplina che studia e valuta l'ottimizzazione delle interazioni tra utenti e sistemi digitali con l'obiettivo di rendere quest'ultimo più intuitivo e user-friendly.

[4] Pareto Principle - https://www.simplypsychology.org/pareto-principle.html

Nel contesto di CoReDex, l'IxD è ben curato per assicurare che ogni interazione, dalla semplice cattura di un CoReMon alla più complessa partecipazione nelle attività di CoReGym, non solo sia intuitiva ma anche ricompensante per l'utente. L'approccio adottato ha l'obiettivo di creare un ciclo positivo di feedback, in cui ogni azione compiuta dall'utente genera una risposta immediata e soddisfacente dal sistema, promuovendo così un'esperienza di utilizzo gradevole e motivante.

Importante in questo quadro è la considerazione dell'accessibilità. CoReDex è progettata per essere accessibile a un ampio spettro di utenti, con diversi livelli di esperienza tecnologica e abilità. Questo non si traduce soltanto nell'implementazione di funzionalità che rispettano i principi dell'accessibilità, ma anche nel design di interazioni che siano comprensibili e facilmente eseguibili da tutti.

Ogni elemento interattivo presente in CoReDex è accuratamente progettato per essere intuitivo, il che contribuisce a minimizzare la curva di apprendimento e a facilitare la rapida familiarizzazione degli utenti con le funzionalità dell'app. L'aspetto visivo è di fondamentale importanza in questo processo: animazioni ed effetti visivi non solo accompagnano, ma anche rafforzano le interazioni degli utenti, fornendo un feedback visivo immediato e comprensibile che li guida e rassicura durante la navigazione.

La sezione "Meet" è particolarmente significativa in questo contesto. Attraverso questa sezione, gli utenti possono catturare un CoReMon o condividere il proprio codice QR per essere catturati. Al momento dello scanning del QR code di un collega, si avvia un'animazione descrittiva del processo di cattura. Questa animazione può illustrare il successo o il fallimento della cattura, oppure visualizzare uno stato intermedio, indicando, ad esempio, che il collega in questione è già stato catturato in passato.

I feedback visivi presenti sono arricchiti da risposte tattili durante l'interazione dell'utente con elementi specifici dell'app, come bottoni, link, icone, o durante azioni quali lo scroll.

Inoltre, nelle sezioni che richiedono una maggiore interazione con il server, come quelle in cui vengono scaricati dati e immagini significativi, viene visualizzato uno spinner per indicare lo stato di caricamento della pagina.

Figura 2.3 - Spinner di caricamento

Questo elemento, rappresentato da una versione stilizzata e minimale del logo dell'app, la CoReBall, è integrato armoniosamente nell'interfaccia, fornendo un segnale visivo del caricamento in corso senza risultare invadente o fastidioso per l'utente.

In sintesi, l'Interaction Design in CoReDex costituisce un equilibrio raffinato tra funzionalità ed estetica, tra intuitività e profondità delle interazioni proposte. Attraverso uno studio accurato delle dinamiche di utilizzo e un design orientato all'utente, CoReDex offre un'interfaccia reattiva e piacevole, che facilita e arricchisce l'esperienza di ogni utente.

2.4 UX in CoReDex

L'esperienza utente (UX) è al fulcro della strategia di CoReDex, con un focus puntato su un'esperienza globale sia coinvolgente che gratificante per l'utente. Il percorso dell'utente inizia con un accattivante Splash Screen all'apertura

dell'applicazione. Successivamente, l'utente è guidato attraverso la schermata di login o registrazione o indirizzato direttamente alla sezione principale. In quest'ultima, è possibile esplorare e filtrare l'elenco dei CoReMon, distinti tra catturati e non, visualizzando per ciascuno il profilo dettagliato, achievement e punti XP accumulati.

La progettazione UX di CoReDex facilita un flusso di navigazione intuitivo e piacevole attraverso le diverse sezioni. La sezione "Gym" riveste un ruolo magnetico, attraendo l'attenzione degli utenti grazie alle challenges settimanali a cui è possibile partecipare. Successivamente, l'utente può navigare verso la sezione "Meet", o esplorare e personalizzare il proprio profilo, accedendo anche alla sezione bacheca, arricchita con funzionalità social e di condivisione.

Questo percorso utente è stato attentamente progettato per essere fluido e coerente, assicurando che ogni transizione e ogni interazione risponda in maniera intuitiva alle aspettative dell'utente. L'obiettivo è offrire un'esperienza utente omogenea e senza intoppi, dalla prima interazione alla navigazione avanzata, consentendo agli utenti di immergersi completamente nell'ecosistema di CoReDex con facilità e piacere.

2.5 Elementi di Engagement

CoReDex integra vari elementi chiave per offrire un'esperienza utente avvincente e coinvolgente. Le sfide settimanali sono un esempio lampante: concepite per incitare la partecipazione attiva, queste non solo forniscono un contesto ludico e educativo, ma sono anche un veicolo per un apprendimento piacevole e stimolante su tematiche sia di cultura ed interesse generale, sia legate

maggiormente al mondo tech. Questo dualismo è fondamentale per mantenere l'utente interessato e motivato nell'utilizzo continuativo dell'applicazione.

Il sistema di punteggi, con i suoi Experience Points (XP), è un altro pilastro dell'engagement in CoReDex. Gli XP sono un efficace strumento di gratificazione intrinseca, un meccanismo che premia l'engagement e la fedeltà dell'utente, incentivando la partecipazione attiva e continuativa. Ogni azione compiuta, ogni sfida vinta, ogni nuovo livello raggiunto viene riconosciuto e premiato con XP ed eventualmente con medaglie, creando un ciclo di feedback positivo che rinforza il coinvolgimento dell'utente.

Un ulteriore elemento che stimola l'engagement è la sezione "**Leaderboard**". Questa schermata visivamente accattivante presenta una classifica generale degli utenti, evidenziando la lista con i primi 10 migliori utenti e un podio con i primi tre CoReMon, basandosi sul numero di punti accumulati e sui livelli raggiunti.

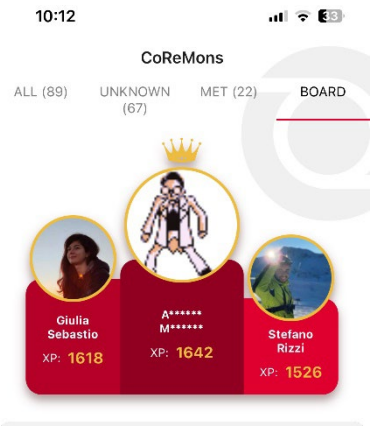

Figura 2.4 – Podio nella schermata Leaderboard

La "Leaderboard" non solo introduce un elemento di competizione amichevole, ma offre anche riconoscimento e visibilità ai membri più attivi e impegnati della comunità, incoraggiando gli altri utenti a emularne il successo. La classifica genera

così un ambiente dinamico e competitivo, in cui gli utenti sono motivati a migliorare e progredire continuamente, mantenendo alta l'attenzione e l'interesse verso l'app.

Questi elementi, integrati e bilanciati, lavorano in sinergia per creare un'esperienza utente coinvolgente e gratificante, in linea con i principi fondamentali dell'User Experience Design e delle teorie del coinvolgimento e della motivazione utente.

Nielsen, in tal senso, ha proposto una serie di principi e linee guida, fondamentali per la progettazione di un'esperienza utente efficace e coinvolgente. Tra i concetti chiave introdotti, spiccano l'importanza dell'usabilità e la necessità di ridurre al minimo la curva di apprendimento per l'utente. Nielsen sostiene che un prodotto deve essere semplice e intuitivo da utilizzare, riducendo così le barriere all'ingresso e facilitando l'impegno immediato dell'utente (Nielsen, 1993).

Le teorie del coinvolgimento e della motivazione utente di Nielsen si concentrano anche sull'importanza del feedback positivo e immediato. Gli utenti, ricevendo riscontri immediati e positivi alle loro azioni, sperimentano un senso di gratificazione e realizzazione che ne incrementa la motivazione. Il ciclo di feedback positivo diviene pertanto un fattore cruciale per mantenere l'utente impegnato e soddisfatto durante l'interazione con il prodotto.

Nielsen ha inoltre sottolineato l'importanza della flessibilità e dell'efficienza d'uso. Offrire diversi modi per realizzare ogni azione, permette agli utenti esperti di accelerare le loro interazioni, incrementando così la loro produttività e soddisfazione. Simultaneamente è fondamentale garantire un ambiente sicuro in cui gli utenti possano esplorare e interagire liberamente, senza timore di commettere errori irreparabili.

3 Sviluppo Frontend

Il presente capitolo ha lo scopo di descrivere la realizzazione del frontend di CoReDex, delineando e analizzando in dettaglio le scelte tecnologiche e metodologiche adottate nel processo di sviluppo.

Il framework scelto è **React Native**, sull'analisi delle specifiche esigenze del progetto e sul confronto con altre tecnologie (linguaggi e framework) disponibili nel panorama del mobile development. React Native si è imposto come soluzione ideale grazie alle sue peculiarità, che saranno esaminate e discusse in seguito.

Successivamente, il capitolo si addentrerà nella descrizione dettagliata di React Native, offrendo un'overview dei suoi componenti fondamentali, delle sue logiche implementative e dei principi guida alla base del suo funzionamento e utilizzo.

Saranno infine presentate anche le principali sfide e questioni che emergono nell'utilizzo di React Native e come queste sono state affrontate e risolte durante la fase di sviluppo.

3.1 Selezione delle Tecnologie

La base tecnologica di CoReDex è essenzialmente costituita da React Native per il frontend, con l'utilizzo congiunto di **JavaScript** e **TypeScript** come linguaggi di programmazione.

React Native è un framework open source creato da Facebook che permette di sviluppare applicazioni mobile per iOS e Android utilizzando JavaScript e React (React Native, s.d.). In tale quadro, JavaScript svolge un ruolo centrale in questo contesto, essendo il linguaggio di programmazione utilizzato per scrivere i componenti React. Si tratta di un linguaggio interpretato, dinamico e di alto livello,

che facilita la stesura di codice dichiarativo. Quest'ultimo, nel contesto specifico di React e React Native, è indispensabile per generare interfacce utente dinamiche e reattive.

Per il progetto CoReDex, la scelta iniziale è ricaduta su JavaScript, per poi evolvere verso l'adozione di TypeScript nella scrittura dei componenti più complessi. JavaScript, è un linguaggio dinamico e non tipizzato, e se da un lato risulta flessibile e di facile utilizzo, dall'altro può causare errori difficilmente identificabili durante le fasi di sviluppo e manutenzione del codice. In contrasto TypeScript, sviluppato e mantenuto da Microsoft, agisce come un'estensione di JavaScript introducendo un sistema di tipi statico. Questo significa che, pur essendo compatibile con tutto il codice JavaScript esistente, TypeScript offre la possibilità di definire e utilizzare tipi di dati, il che si traduce in un codice più leggibile, affidabile e manutenibile. La tipizzazione statica consente, inoltre, l'identificazione precoce di errori durante la compilazione, facilitando le attività di debugging e migliorando la qualità del software risultante. In sintesi, mentre JavaScript costituisce la base per uno sviluppo agilmente creativo e dinamico, TypeScript incrementa la sicurezza e la precisione del codice, aspetti fondamentali in progetti complessi e di lavoro collaborativo. Con l'aumento della complessità del progetto, la transizione da JavaScript a TypeScript è apparsa una scelta logica e vantaggiosa.

3.2 React Native

React Native rappresenta uno dei framework più innovativi ed efficienti nell'ambito dello sviluppo di applicazioni mobile. Lanciato da Meta nel 2015, il framework consente di realizzare app per iOS e Android attraverso un codice quasi interamente basato su JavaScript e React. La sua struttura consente agli sviluppatori di riutilizzare il codice, accelerando il processo di sviluppo e mantenendo

un'esperienza utente elevata, quasi indistinguibile da quella fornita da applicazioni native.

Il framework opera attraverso componenti che permettono di creare codice modularizzato e riutilizzabile. Questo approccio consente di avere un codice base chiaro e manutenibile, fattore cruciale per garantire la sostenibilità del progetto nel lungo termine.

Un altro vantaggio è il fatto che React Native utilizza il Virtual DOM, cioè una versione semplificata e ottimizzata del Document Object Model (DOM) conservata in memoria. Questa rappresentazione permette di effettuare aggiornamenti al layout dell'app in modo rapido ed efficiente, garantendo così prestazioni ottimali. Il Virtual DOM agisce come un intermediario, minimizzando il lavoro computazionale necessario per implementare le modifiche visive, garantendo un risultato fondamentale per il fluido funzionamento delle applicazioni.

3.3 Perché React Native?

La scelta del framework è dovuta a varie motivazioni:

- **Versatilità Cross-Platform**: React Native facilita lo sviluppo di applicazioni cross-platform, permettendo agli sviluppatori di creare un codice sorgente unificato funzionante sia su Android che su iOS. Questa caratteristica non solo assicura un'esperienza utente consistente e uniforme attraverso dispositivi diversi, ma rende anche il processo di sviluppo più efficiente. Utilizzando questo framework, gli sviluppatori possono economizzare tempo e risorse: le fasi di codifica, testing e debugging vengono eseguite una singola volta per entrambe le piattaforme. L'utilizzo di un linguaggio di programmazione unico per Android e iOS abbrevia i cicli di sviluppo, facilitando il rilascio di applicazioni solide e ad alte prestazioni.

- **Community e supporto**: il framework gode di una vasta e attiva community di sviluppatori e un ampio supporto da parte di Meta. Questa solida base di utenti ed esperti contribuisce alla risoluzione di problemi, condivisione di best practices e sviluppo di plugin ed estensioni per estendere le funzionalità di React Native.
- **Codice Riutilizzabile e Hot Reloading**: con React Native, è possibile riutilizzare il codice scritto, riducendo drasticamente il tempo di sviluppo. Inoltre, la funzione di Hot Reloading consente di visualizzare in tempo reale le modifiche apportate al codice senza dover ricompilare l'intera applicazione, rendendo il processo di sviluppo più rapido ed efficiente.
- **Facilità di apprendimento**: per gli sviluppatori già versatili in JavaScript e React, l'apprendimento di React Native si presenta come un processo semplice ed intuitivo, agevolando così la transizione da uno sviluppo orientato al web a quello mobile.
- **Ecosistema e librerie**: React Native è supportato da un ricco ecosistema di librerie e moduli di terze parti che facilitano l'integrazione di funzionalità avanzate e personalizzate, rispondendo così a un'ampia varietà di requisiti di sviluppo.

Riassumendo, la scelta di React Native è stata guidata dalla necessità di un framework che offrisse versatilità, supporto, prestazioni e una curva di apprendimento agevole, e che permettesse lo sviluppo simultaneo dell'app sia per dispositivi Android che iOS utilizzando un unico linguaggio di programmazione.

3.4 Comprendere il Legame: da React Native a React

Un'attenzione particolare va posta sul rapporto stretto e indispensabile tra React Native e React, dal momento che quest'ultimo costituisce il fondamento su cui React

Native dispiega la sua efficienza e potenza. **React** è una libreria JavaScript open-source, rilasciata da Facebook (ora Meta) nel 2013, essenziale per sviluppare interfacce utente (UI) dinamiche e interattive per applicazioni web.

Il concetto chiave introdotto da React è quello di "componenti", elementi componibili e riutilizzabili che consentono di sviluppare UI complesse, mantenendo il codice snello e facilmente manutenibile. Ogni componente in React rappresenta un segmento indipendente e riutilizzabile dell'UI, facilmente integrabile in applicazioni più ampie.

Fondamentale è anche l'approccio dichiarativo adottato da React, che, come precedentemente illustrato, semplifica tanto la scrittura quanto la lettura del codice, focalizzandosi su "cosa" si desidera ottenere piuttosto che sul "come" raggiungere l'obiettivo. Questa metodologia permette di automatizzare numerosi aspetti legati al rendering e all'aggiornamento della UI, lasciando agli sviluppatori la libertà di concentrarsi sulla logica applicativa.

React Native, ereditando questi principi, li estende al contesto mobile, consentendo lo sviluppo di applicazioni utilizzando la stessa filosofia e tecniche proprie di React. Questa importante caratteristica facilita la transizione degli sviluppatori da web a mobile, assicurando che le competenze e le best practices acquisite con React siano facilmente trasferibili e applicabili in React Native; tutto ciò rende questo framework non una mera trasposizione di React nel mobile, ma una sua efficace e potente estensione.

3.5 React Native: le basi

La seguente sezione è dedicata alla descrizione dettagliata delle logiche cardine alla base del framework scelto. Si focalizzerà in particolar modo sulla delineazione dei concetti quali "componente", "stylesheet", "props", "hooks" (focalizzandosi

specificatamente sugli hooks “useState” e “useEffect”) e “context”. Questi elementi sono tra i più frequentemente utilizzati nella scrittura del codice di CoReDex. Illustrarne le caratteristiche ed il funzionamento permette di fornire una visione chiara ed approfondita delle metodologie pratiche e programmatiche adottate nella realizzazione dell’applicazione.

3.5.1 Componenti

Un componente in React Native rappresenta un elemento fondamentale dell’interfaccia utente, con le caratteristiche di essere autonomo e riutilizzabile.

Ogni componente è strutturato per essere un modulo che incapsula un pezzo di interfaccia utente, includendo sia la logica che la presentazione visiva.

I componenti possono essere distinti in due categorie principali: quelli standard, forniti direttamente dalla libreria React Native e i componenti custom, creati dagli sviluppatori per soddisfare le esigenze specifiche del progetto.

Esempi di componenti standard sono “View”, “Text”, “Image”. I componenti standard coprono le funzionalità base per la creazione dell’interfaccia utente e gestiscono in modo nativo la visualizzazione e l’interazione con gli elementi dell’UI sui dispositivi mobile. Sono ottimizzati da un punto di vista di prestazioni e sono pronti all’uso, senza la necessità di installazioni aggiuntive o configurazioni complesse.

I componenti custom, d’altra parte, sono definiti dagli sviluppatori per incorporare funzionalità specifiche, raggruppando più componenti standard ed eventualmente altri componenti custom in un’unica unità funzionale e riutilizzabile. Creare componenti custom è un processo fondamentale per organizzare e strutturare il codice in modo efficiente e permette di creare in modo modulare UI complesse e personalizzate. Tali componenti possono essere molto

semplici oppure contenere al loro interno delle logiche più complesse, come la gestione di stati interni e chiamate a funzioni.

In CoReDex i componenti custom sono stati definiti tramite componenti funzionali, che sono una forma più concisa e moderna di componenti rispetto ai classici componenti di classe.

Un componente funzionale in React e React Native è essenzialmente una funzione JavaScript indipendente che accetta in input degli attributi dentro un oggetto di proprietà (chiamato "props") e restituisce in output degli elementi React, che descrivono ciò che dovrebbe apparire sullo schermo del dispositivo mobile. Questi componenti sono leggeri, semplici da scrivere e comprendere e favoriscono l'adozione di best practices come la composizione e la riusabilità del codice.

I componenti custom sono scritti in JSX (JavaScript XML), una sintassi estensione di JavaScript utilizzata per scrivere l'UI in modo che assomigli a XML o HTML [5]. Il JSX facilita la lettura e la scrittura del codice React Native, rappresentando i componenti come elementi, similmente ai tag HTML.

L'esempio sottostante mostra il codice relativo al componente funzionale **InfoComponent**: il componente è scritto in TypeScript e JSX; restituisce una View al cui interno sono presenti del testo e un'icona oppure un'immagine, a seconda degli attributi passati come props al componente stesso.

[5] JSX: Putting markup into JavaScript - https://react.dev/learn/writing-markup-with-jsx

```jsx
export default function InfoComponent({
  text,
  iconName = null,
  isTextAbove = false,
  iconSize = 40,
  style = {},
}: InfoComponentProps) {
  return (
    <View style={[styles.root, { ...style }]}>
      {isTextAbove && <Text style={styles.text}>{text}</Text>}
      {iconName && (
        <CustomIcon
          name={iconName}
          size={iconSize}
          color={APP_COLORS.midGrey}
          isCircled={false}
          style={{ marginVertical: 20 }}
        />
      )}
      {!iconName && (
        <StandardImage
          style={{ margin: 50 }}
          staticImage={require("../../../assets/images/errorPokeball.png")}
          imgDimension="large"
        />
      )}
      {!isTextAbove && <Text style={styles.text}>{text}</Text>}
    </View>
  );
}
```

Figura 3.1 - Componente JSX "InfoComponent"

L'immagine sottostante descrive la UI costruita con il componente InfoComponent quando la prop "isTextAbove" è "false" e quando non viene fornito nessun iconName.

Figura 3.2 - UI del componente InfoComponent

L'architettura di React e React Native enfatizza la modularità e la riusabilità del codice, consentendo una gestione efficace dei componenti sviluppati, e dando la

possibilità di impiegare il componente InfoComponent (e tutti gli altri definiti durante la fase di sviluppo) in diverse sezioni del codice. Tale pratica risponde dinamicamente alle necessità di UI e alle logiche applicative intrinseche all'app.

Tale strategia di sviluppo non solo minimizza la verbosità del codice, ma contribuisce anche significativamente alla riduzione degli errori e delle necessità di modifica. In particolare, ogni volta che è necessario modificare la UI associata a InfoComponent, le modifiche possono essere implementate direttamente all'interno del codice sorgente di InfoComponent. Questo approccio propaga automaticamente le variazioni a tutti i segmenti dell'UI dell'app integranti InfoComponent, garantendo coerenza e integrità nell'esperienza utente finale e ottimizzando il processo di sviluppo e manutenzione del software.

3.5.2 Props

Ogni componente padre può passare alcune informazioni ai suoi componenti figli assegnando loro delle "**props**".

Le props in React Native, simili a quelle in React, rappresentano un meccanismo attraverso cui i dati vengono passati ai componenti, consentendo di seguire un modello dichiarativo per la creazione dell'interfaccia utente. I componenti utilizzano le props per comunicare tra di loro.

La caratteristica fondamentale alla base delle props è che queste sono valori immutabili passati come input dai componenti padri ai componenti figli. Tali valori possono essere di qualsiasi tipo, inclusi oggetti, funzioni, array e tipi primitivi.

Nel contesto di un componente funzionale React Native, le props vengono ricevute come un argomento della funzione, e possono essere utilizzate per determinare l'output del componente. Tali props possono essere trasferite ai componenti figli mediante l'espansione dell'oggetto props all'interno del

componente discendente. Questo processo avviene assegnando al figlio, da parte del componente genitore, ciascuna prop mediante la dichiarazione del nome della prop e l'attribuzione del relativo valore.

```
<View style={styles.detail}>
  <InfoComponent
    text="Nothing to see here..."
    isTextAbove={false}
    iconSize={50}
    style
  />
</View>
```

Figura 3.3 – Assegnazione delle props al componente InfoComponent

Oltre alle props personalizzate, i componenti React Native possono ricevere una serie di props predefinite che influenzano il comportamento o lo stile del componente. Ad esempio, il componente View accetta props come “style”, “accessible”, etc. che ne modulano l’aspetto e l’interattività.

Anche le funzioni possono essere trasmesse come props dai componenti padre ai componenti figlio. Tali props, denominate "**callback**", permettono ai componenti figlio di comunicare con i componenti padre. Ciò è particolarmente utile per segnalare modifiche di stato o eventi scaturiti dall'interazione dell'utente con l'app.

```
<ImagePicker
  style={{ marginTop: 20 }}
  onSave={onSave}
  apiMethod={apiMethod}
  handleImageModal={onClose}
  imageXAspectRatio={imageXAspectRatio}
  imageYAspectRatio={imageYAspectRatio}
/>
```

Figura 3.4 - Callback come props nel componente ImagePicker

Nell’esempio precedente, il componente ImagePicker accetta come props tre callback: onSave, apiMethod e onClose.

```
const handleSubmit = async () => {
  // ImagePicker saves the taken photo to disk and returns a local URI to it
  handleImageModal();
  try {
    const blob = await fetchImageFromUri(image.uri);
    const res = await apiMethod(blob);
    onSave(image.uri);
  } catch (err) {
    bLogger.error("Error while uploading image: ", err);
  }
};
```

Figura 3.5 – Funzione definita dentro al componente ImagePicker che richiama una callback passata come prop

Quando l'utente preme il bottone "Save Image", presente nel codice sorgente di ImagePicker, viene chiamata la funzione "handleSubmit", anch'essa definita all'interno di ImagePicker. In un determinato punto di tale funzione, viene invocata la callback onSave(), la quale notifica al componente padre che l'immagine è stata salvata con successo.

3.5.3 Stato e Hooks

La gestione dello stato in React Native assicura il dinamismo e la reattività delle applicazioni. Lo stato di un componente rappresenta una struttura dati che può cambiare nel tempo, in base alle interazioni tra utente e app, influenzando il rendering del componente stesso.

Quando lo stato di un componente subisce una modifica, il framework provvede a ri-renderizzare tale componente, presentando all'utente le informazioni aggiornate. Se il componente modificato agisce come componente padre, il suo cambiamento di stato innescherà il re-rendering non soltanto del componente stesso ma anche di tutti i suoi componenti figli che dipendono da tale stato attraverso le props assegnate. In altre parole, ogni componente figlio che ha ricevuto lo stato

aggiornato come prop verrà automaticamente ri-renderizzato con le informazioni più recenti.

È importante sottolineare che React Native esegue questo processo in maniera ottimizzata. Infatti, il framework assicura che il re-rendering venga propagato esclusivamente ai componenti il cui aggiornamento è necessario, ovvero quelli che dipendono dallo stato o dalle props modificate. Questa ottimizzazione mira a minimizzare il carico computazionale, rendendo il processo di aggiornamento più efficiente.

Un modo diretto e comprensibile per gestire lo stato e gli effetti collaterali nei componenti è dato dagli "**hooks**". Gli hooks, introdotti con la versione 16.8 di React, sono funzioni speciali che permettono di utilizzare lo stato e altri concetti di React in componenti funzionali (React Dev, s.d.). All'interno di CoReDex, sono stati utilizzati principalmente due hooks: useState e useEffect.

3.5.4 Hooks: useState

L'hook "useState" è una funzione che permette ai componenti di mantenere e manipolare il proprio stato locale. Quando invocato, useState accetta un valore iniziale dello stato e restituisce un array di due elementi:

- **Valore dello stato corrente**: è il primo elemento dell'array. Questo valore si aggiorna ogni volta che lo stato cambia, e la sua modifica causerà il re-render del componente.
- **Funzione di aggiornamento**: è il secondo elemento dell'array. Si tratta di una funzione che, quando viene chiamata, consente di aggiornare lo stato con un nuovo valore.

Nell'esempio che segue, dentro al componente Pokeball, è definito uno stato: "isModalVisible". L'hook "useState(false)" inizializza isModalVisible con il valore "false".

```
const Pokeball = ({ pokeballCount }) => {
  const [isModalVisible, setIsModalVisible] = useState(false);

  const onPressWrapper = () => {
    // Run haptic feedback if needed and run the onPress callback
    Haptics.selectionAsync();
    setIsModalVisible(true);
  };

  return (
    <>
      <View style={styles.container}>
        <TouchableOpacity
          onPress={() => {
            onPressWrapper();
          }}
        >
          <Image source={require("../../../../assets/PokeballProfile.png")} style={styles.pokeballImage} />
        </TouchableOpacity>
        <Text style={styles.pokeballCount}>{pokeballCount}</Text>
      </View>
      <Modal
        visible={isModalVisible}
        onRequestClose={() => {
          setIsModalVisible(false);
        }}
      >
        <View style={styles.centeredView}>···
        </View>
      </Modal>
    </>
  );
};
```

Figura 3.6 - Set di uno stato dentro al componente Pokeball

Quando l'utente clicca sull'immagine della Pokeball (il componente Image), che funge da figlio di TouchableOpacity, si attiva la funzione onPressWrapper(). Questa funzione modifica lo stato isModalVisible, assegnandogli il valore "true" attraverso la funzione setIsModalVisible(). Di conseguenza, il componente Pokeball subisce un re-rendering, rendendo visibile il componente Modal al suo interno, la cui visibilità è controllata dallo stato isModalVisible.

Successivamente, alla chiusura del modale da parte dell'utente, lo stato isModalVisible viene reimpostato a "false", provocando un ulteriore re-rendering

del componente Pokeball e, per estensione, del componente Modal annidato, che risulta nuovamente invisibile.

Gli stati e le props sono intrinsecamente connessi in React Native. Mentre gli stati rappresentano dati mutevoli, le props non possono essere modificate all'interno del componente. Gli stati controllano i dati all'interno dei componenti, mentre le props facilitano il trasferimento di dati tra componenti, instaurando una dinamica fluida di scambio informazioni.

3.5.5 Hooks: useEffect

L'hook "useEffect" permette di eseguire effetti collaterali all'interno dei componenti. Gli effetti collaterali sono tutte quelle operazioni che non sono legate al rendering del componente, come le chiamate API, le sottoscrizioni o la manipolazione manuale del DOM. Queste operazioni non riguardano il rendering immediato dell'interfaccia utente, ma sono cruciali per la logica funzionale dell'applicazione.

La useEffect accetta due parametri:

- **Funzione che contiene il codice dell'effetto collaterale**: è il primo parametro dell'hook, ed è sempre obbligatorio.
- **Array di dipendenze**: è il secondo argomento. La funzione dell'effetto collaterale viene invocata ogniqualvolta si verifica una modifica in una delle dipendenze elencate nell'array. Se l'array è vuoto, la funzione dell'effetto collaterale si attiva esclusivamente al primo rendering (o "mount") del componente. Omettendo completamente l'array, il side effect verrà eseguito a seguito di ogni re-render del componente.

Nell'esempio seguente, è definita una useEffect con array di dipendenze vuoto.

```
export default function SplashScreen({ route, navigation }) {
  const [isVersionValid, setIsVersionValid] = useState(null);
  const bLogger = new BetterLogger(BetterLogger.VerbosityLevel.ERROR, SplashScreen.name);

  useEffect(() => {
    isCurrentNewerOrEqualThanMin() //utility fn
      .then((isValid: boolean) => {
        bLogger.debug("isCurrentNewerOrEqualThanMin() -> isValid: ", isValid);
        setIsVersionValid(isValid);
      })
      .catch((error) => {
        bLogger.error("isCurrentNewerOrEqualThanMin() -> error", error);
        setIsVersionValid(false);
      });
  }, []);
```

Figura 3.7 - UseEffect dentro al componente SplashScreen

Dopo il primo rendering del componente SplashScreen di CoReDex, viene eseguita la funzione associata al codice dell'effetto collaterale definita dentro alla useEffect: al suo interno viene invocata la funzione asincrona isCurrentNewerOrEqualThanMin(). Questa funzione consulta il server verificando se la versione attuale dell'app corrisponde all'ultima disponibile o se esistono versioni più aggiornate, restituendo una promise. In base al risultato di tale promise, il valore dello stato “isVersionValid” viene opportunamente impostato a “true” o “false”.

3.5.6 Context

Il Context API in React Native (e React) fornisce un modo per facilitare la condivisione di dati tra componenti senza la necessità di passare esplicitamente le props attraverso ogni livello dell’albero dei componenti. È particolarmente utile quando si hanno dati “globali” condivisi ed utilizzati da molti componenti, come temi, autenticazione, o lingua.

In CoReDex i context sono stati utilizzati per i seguenti scopi: autenticazione, gestione dei colleghi dell'utente attraverso componenti diversi, gestione delle varie challenges e del loro aggiornamento e completamento.

Un esempio è il context "authContext", utilizzato nella gestione dell'autenticazione e delle informazioni relative all'utente loggato.

```
export interface IAuthContext {
  user: any;
  userInfo: any;
  loading: any;
  login: any;
  logout: any;
  updateUser: any;
  getUserMedal: any;
  expEvents: IExpEvent[];
}

const authContext = createContext<IAuthContext | null>(null);
```

Figura 3.8 - authContext

La creazione di authContext avviene tramite il metodo createContext(); inizialmente il context è inizializzato a "null". Questo oggetto è l'entità principale utilizzata per condividere i dati di autenticazione tra i vari componenti.

```
export function ProvideAuth({ children }) {
  const auth = useProvideAuthContext();
  return <authContext.Provider value={auth}>{children}</authContext.Provider>;
}
```

Figura 3.9 - Context Provider

Il ProvideAuth funge da Provider. Questo componente utilizza un hook personalizzato useProvideAuthContext() per gestire lo stato e le logiche relative all'autenticazione. Il valore restituito da questo hook viene passato ai componenti figli attraverso il componente authContext.Provider.

```
function useProvideAuthContext() {
  const [user, setUser] = useState(null);
  const [userInfo, setUserInfo] = useState(null);
  const [loading, setLoading] = useState<boolean>(true);

  const login = (email, password) => {
    return Account.logIn(email, password);
  };

  const logout = async () => {
    const res = await Account.logOut();
    setUser(null);
    return res;
  };

  const updateUser = (newUserInfo) => {
    setUserInfo(newUserInfo);
  };
```

Figura 3.10 - Hook personalizzato useProvideAuthContext

L'hook personalizzato "useProvideAuthContext" gestisce diversi stati:

- "**user**": informazioni sull'utente correntemente loggato.
- "**userInfo**": dettagli ulteriori sull'utente.
- "**loading**": stato booleano che indica se il fetching delle informazioni dell'utente è in fase di caricamento.
- "**appState**": tiene traccia dello stato dell'applicazione (attiva o in background).

L'hook implementa diverse funzioni, come "login" e "logout", utilizzate per gestire le operazioni di autenticazione, e "updateUser" per aggiornare localmente le informazioni dell'utente. Sono usate inoltre delle useEffect per effettuare operazioni collaterali, come il caricamento dei dati dell'utente.

```
export const useAuth = () => {
  return useContext(authContext);
};
```

Figura 3.11 - Hook useAuth

Il Context può essere consumato nei componenti figli attraverso l'hook useAuth, facilitando l'accesso ai dati di autenticazione e alle funzioni correlate in qualsiasi componente figlio, come mostrato nell'esempio che segue.

```
export default function EditProfileScreen({ navigation, route }) {
  const auth = useAuth();
  const userInfo = auth && auth.userInfo;

  const editUser = (obj) => {
    const newUser = { ...userInfo, ...obj };
    auth.updateUser(newUser);
  };
```

Figura 3.12 - Utilizzo del context dentro al componente EditProfileScreen

Il componente EditProfileScreen permette all'utente di modificare alcune informazioni relative al proprio profilo. Quando viene invocata la funzione "editUser(obj)", viene prima costruito un oggetto "newUser" per poi essere chiamato il metodo auth.updateUser(newUser) fornito dal context "auth" restituito da useAuth(): questo metodo serve per aggiornare le informazioni relative all'utente nel context stesso. Come risultato, qualsiasi componente dell'applicazione che consuma il context useAuth() sarà notificato di questo aggiornamento e avrà accesso alle informazioni aggiornate dell'utente, facilitando così la gestione dello stato dell'utente in maniera centralizzata e coerente attraverso l'intera applicazione.

3.6 React Native VS. Web View

Dopo aver esplorato i fondamenti di React Native, è importante comprendere come questa tecnologia si posizioni rispetto ad altre soluzioni di sviluppo mobile, in particolare rispetto alla tecnologia delle WebView.

Il campo dello sviluppo di applicazioni mobile è caratterizzato da diverse tecnologie e approcci. In questo contesto, per degli sviluppatori pratici con

JavaScript e sviluppo web, React Native e WebView rappresentano due possibili soluzioni, ciascuna con peculiarità, vantaggi e sfide specifiche.

Come esplorato nelle sezioni precedenti, React Native è un framework che consente di sviluppare applicazioni mobile utilizzando JavaScript e React. Grazie al suo approccio che facilita una fluida interazione tra il codice JavaScript e le piattaforme native, React Native offre performance che si avvicinano a quelle delle applicazioni native, permettendo allo stesso tempo lo sviluppo attraverso una base di codice condivisa su diversi sistemi operativi.

Le WebView, d'altro canto, sono componenti che permettono di visualizzare contenuti web all'interno di un'applicazione nativa. Per "nativo" ci si riferisce a piattaforme software specifiche per dispositivi mobili, come Android e iOS. Le WebView permettono dunque di incorporare pagine web e contenuti scritti con HTML, CSS e JavaScript direttamente all'interno dell'app, consentendo agli sviluppatori di creare interfacce utente utilizzando tecnologie web (Rosser, s.d.). Le WebView offrono inoltre delle logiche per consentire una certa interazione tra i contenuti web e le funzionalità native del dispositivo, nonché strumenti per navigare attraverso pagine web in modo simile a quanto accade nei browser.

In generale, questi componenti sono spesso utilizzati per sviluppare app ibride, che sono in parte native e in parte basate su contenuti web. Uno dei vantaggi è il poter combinare la potenza delle funzionalità native con la flessibilità e la portabilità delle tecnologie web. Inoltre, le WebView permettono di visualizzare contenuti web dinamici e aggiornati senza la necessità di aggiornare l'intera applicazione.

Se si mettono a confronto le due tecnologie, si nota che React Native generalmente garantisce prestazioni superiori rispetto alle WebView. Ciò è attribuibile al meccanismo di funzionamento di ReactNative, che converte il codice JavaScript in codice nativo. Questo processo permette un'interazione più efficiente

con il sistema operativo del dispositivo mobile, riducendo i tempi di latenza e ottimizzando l'utilizzo delle risorse hardware.

Dall'altro lato, le WebView eseguono il codice JavaScript all'interno di un browser incorporato nell'applicazione nativa. Nonostante questa tecnologia permetta di visualizzare contenuti web dinamici e complessi, viene introdotto un overhead significativo. Tale overhead è dovuto sia al bisogno di interpretare il codice JavaScript in real time, sia alla necessità di gestire un'interfaccia web nel contesto di un'app mobile. Come conseguenza, si ha l'inevitabile rallentamento dell'applicazione, specialmente in situazioni in cui le risorse del dispositivo sono limitate o il carico di lavoro risulta essere particolarmente intenso.

Per quanto riguarda il rapporto con le funzionalità native, React Native fornisce un accesso più ampio e diretto alle API native del dispositivo, permettendo agli sviluppatori di integrare in modo fluido servizi e componenti hardware specifici, come il GPS, la fotocamera o le notifiche push. In contrasto, le WebView presentano restrizioni nell'accesso alle funzionalità native: la loro architettura pone delle limitazioni nell'interazione diretta con il sistema operativo. Tali limitazioni possono essere superate grazie all'utilizzo di plugin o servizi di terze parti, introducendo tuttavia una certa complessità nel processo di sviluppo, nonché una minore stabilità nel funzionamento dell'app, problemi di sicurezza e di manutenibilità.

Un'ultima differenza riguarda l'esperienza utente. La gestione avanzata della memoria e l'esecuzione ottimizzata del codice in React Native contribuiscono a un'esperienza utente fluida e reattiva, avvicinandosi alla qualità delle applicazioni sviluppate con codice nativo. Le app che utilizzano delle WebView possono invece mostrare limitazioni in termini di reattività e consistenza dell'esperienza utente. Non sempre, infatti, è possibile replicare la fluidità e la velocità di risposta delle app interamente native o di quelle sviluppate con React Native. Spesso gli utenti sperimentano ritardi nelle transizioni e poca reattività ai tocchi, nonché una

mancanza di omogeneità visiva con il sistema operativo e con le altre applicazioni. Tutto ciò si traduce in un'esperienza utente frammentata e meno intuitiva, che potrebbe diminuire l'usabilità dell'app.

3.7 React Native VS. Applicazioni Native

Risulta ora essenziale affrontare un'ulteriore comparazione di rilievo per fornire un quadro completo delle scelte tecnologiche fatte durante lo sviluppo di CoReDex: la differenza tra React Native e codice puramente nativo.

Optare per uno dei due approcci rappresenta uno degli snodi fondamentali del processo di ideazione di un'applicazione mobile. Diversamente dalla scelta tra WebView e React Native, dove il linguaggio di programmazione predominante è JavaScript, in questo caso si verifica un cambiamento significativo: si assiste infatti non solo a modifiche nel processo e nelle logiche di scrittura del codice, ma anche nella selezione del linguaggio di programmazione utilizzato.

3.7.1 Codice Nativo: pro e contro

Lo sviluppo mediante codice nativo conduce a vantaggi notevoli. Uno dei principali è rappresentato dalla garanzia di prestazioni ineguagliabili. Essendo scritto in un linguaggio specifico per ciascun sistema operativo (solitamente Swift per iOS e Kotlin per Android), il codice nativo è in grado di comunicare più efficacemente con l'hardware del dispositivo, offrendo un'esperienza utente altamente fluida e reattiva (LEDU Education Ecosystem, s.d.).

Un ulteriore vantaggio è l'accesso completo e diretto alle API fornite dai vari sistemi operativi. Questo non solo consente una maggiore flessibilità nello sviluppo,

ma permette anche di integrare rapidamente e in modo nativo le nuove funzionalità rilasciate dai sistemi operativi stessi (Apple Developer, s.d.).

Infine, le interfacce utente delle applicazioni native sono generalmente più aderenti alle linee guida di design stabilite da iOS e Android. L'esperienza utente offerta è non solo più fluida e intuitiva, ma anche più in linea con le aspettative degli utenti che sono abituati al look-and-feel tipico delle piattaforme per cui l'app è stata sviluppata. Ciò aiuta a creare applicazioni che sono percepite come più affidabili e professionali dagli utenti finali (Savonin, s.d.).

Pur offrendo indubbi benefici, lo sviluppo con codice nativo comporta anche delle sfide. Una delle principali è la necessità di gestire e mantenere set di codici distinti per ciascuna piattaforma (iOS, Android), richiedendo un impegno notevole in termini di tempo e risorse umane. Questo scenario impone agli sviluppatori di gestire due codebase separatamente, ognuna con le sue specificità e requisiti, aumentando così la complessità del processo di sviluppo e mantenimento del software (Savonin, s.d.).

Questo aspetto influenza anche la curva di apprendimento: gli sviluppatori devono infatti acquisire padronanza di linguaggi di programmazioni diversi (ognuno con la propria sintassi e i propri paradigmi), in funzione della piattaforma per cui intendono sviluppare.

3.8 React Native: pro e contro

Tramite React Native è possibile colmare alcune sfide associate allo sviluppo con codice nativo.

Come già sottolineato, una delle sue caratteristiche principali è la promessa di "*Learn once, write anywhere*", ovvero la possibilità di utilizzare un codice base unico

che possa essere distribuito su molteplici piattaforme, riducendo significativamente i tempi di sviluppo e i costi associati.

Potendo utilizzare un unico linguaggio di programmazione, la riusabilità del codice è massimizzata, mentre il tempo di sviluppo diminuisce notevolmente. Inoltre, esiste una grande community attorno allo sviluppo in React Native, in grado di fornire supporto e accesso a librerie e framework.

Le prestazioni di React Native sono, in molti casi, comparabili a quelle delle applicazioni scritte in codice nativo. Il framework è ottimizzato per offrire un'esperienza utente fluida e reattiva ed è in grado di eseguire il codice JavaScript in maniera efficiente, minimizzando il divario prestazionale rispetto al codice nativo.

Infine, React Native possiede un'altra caratteristica vantaggiosa che merita di essere evidenziata: l'Hot Reload. Questa funzionalità permette agli sviluppatori di vedere immediatamente l'effetto delle ultime modifiche apportate al codice senza dover ricompilare completamente l'applicazione, riducendo il tempo di attesa per visualizzare le modifiche e facilitando soprattutto il processo di debug e testing.

Così come lo sviluppo in codice nativo, anche React Native presenta alcuni svantaggi. Sebbene il framework offra ottime performance, in alcuni casi (soprattutto per app più complesse) non è possibile eguagliare la velocità e l'efficienza delle app native.

Un altro limite risiede nel fatto che alcune API native più recenti o avanzate potrebbero non essere supportate da React Native. Inoltre, le app React Native tendono ad avere dimensioni maggiori rispetto a quelle native.

Infine, nonostante la community attorno al framework sia attiva e in crescita, gli aggiornamenti frequenti del framework possono a volte indurre problemi di

compatibilità tra librerie e richiedere un certo tempo necessario alla manutenzione del codice.

3.9 Perché React Native per CoReDex?

Valutando attentamente i pro e i contro di ciascuna tecnologia, il team di sviluppo di CoReDex ha optato per React Native come soluzione ottimale di sviluppo. Questa scelta è motivata dal desiderio di creare un'applicazione performante, reattiva e cross-platform, con un processo di sviluppo snello ed efficiente. In questo contesto, React Native emerge come la tecnologia che meglio soddisfa le esigenze e gli obiettivi del progetto, rappresentando un compromesso equilibrato tra performance, produttività e portabilità del codice su diverse piattaforme mobile.

4 Clean Architecture

La Clean Architecture è un concetto introdotto da Robert C. Martin (noto anche come Uncle Bob) nel suo libro "Clean Architecture: A Craftsman's Guide to Software Structure and Design".

Robert C. Martin dà una definizione di "architettura":

"L'architettura di un sistema software è la forma data a tale sistema da coloro che l'hanno realizzato. L'aspetto di tale forma è dato dalla suddivisione di tale sistema in componenti, dalla disposizione di tali componenti e dai modi in cui tali componenti comunicano fra loro. Lo scopo di tale forma è di facilitare lo sviluppo, il deployment, il funzionamento e la manutenzione del sistema software che essa contiene. La strategia che governa questa idea di "facilitare" consiste nel lasciare aperte quante più opzioni possibili, per il tempo più lungo possibile."

Dunque, l'obiettivo centrale dell'architettura è garantire che il sistema sia intuitivo e semplice da sviluppare, mantenere ed espandere. Ciò mira a ridurre al minimo i costi associati a queste attività, massimizzando al contempo l'efficienza e la produttività degli sviluppatori. Per raggiungere tali obiettivi, CoReDex adotta i principi della Clean Architecture.

La Clean Architecture è incentrata su un principio fondamentale: la separazione delle responsabilità. L'obiettivo principale è quello di creare un sistema software dove ogni componente ha una responsabilità ben definita e i diversi livelli dell'applicazione sono disaccoppiati tra loro, promuovendo così la manutenibilità, la scalabilità e la facilità di testing (Martin, Clean Architecture: A Craftsman's Guide to Software Structure and Design, 2017).

Questo si traduce in un'architettura composta da diversi strati concentrici, dove ogni strato interno è isolato da quelli esterni.

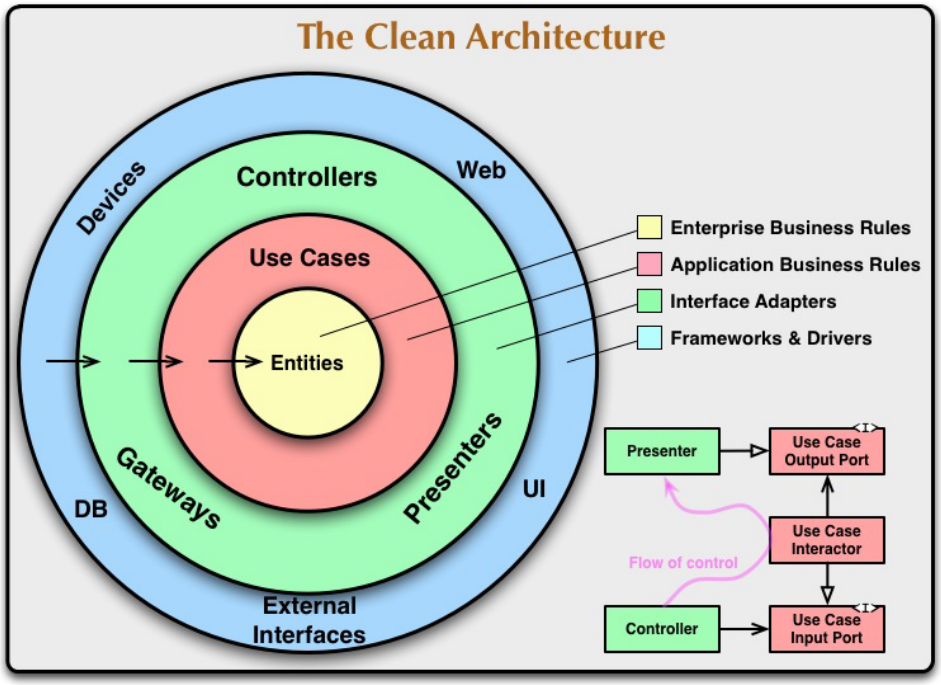

Figura 4.1 - The Clean Architecture (Martin, The Clean Architecture, s.d.)

In termini pratici, la Clean Architecture suggerisce di dividere un'applicazione in almeno quattro livelli:

1. **Entities**: contengono le regole di business e i modelli di dati.
2. **Use Cases**: contengono la logica applicativa specifica.
3. **Interface Adapters**: convertitori tra dati in un formato utilizzabile dagli use case e dagli entities in formati più adatti per i database o altri agenti esterni.
4. **Frameworks and Drivers**: è il livello più esterno e comprende UI, database, framework, ecc.

Nell'ambito di CoReDex, l'adozione della Clean Architecture mira a:

- **Migliorare la manutenibilità e la scalabilità**: separando le preoccupazioni, il codice diventa più facile da gestire e aggiornare. La struttura modulare consente di aggiungere nuove funzionalità o modificare quelle esistenti senza influenzare in modo significativo altri componenti del sistema. Ad esempio, i cambiamenti nell'UI o in un database non influenzano la logica di business situata nei livelli interni.
- **Facilitare le operazioni di testing**: con una chiara separazione dei livelli e delle responsabilità, la Clean Architecture rende più semplice scrivere test unitari e di integrazione. Ogni componente può essere testato indipendentemente, aumentando l'affidabilità dei test e del software stesso.
- **Non dipendere dai framework**: la Clean Architecture promuove un design che non dipende da librerie o framework specifici. Questo rende l'applicazione meno vulnerabile ai cambiamenti nei framework esterni e facilita eventuali aggiornamenti o migrazioni.
- **Aumentare la flessibilità e l'adattabilità**: il codice indipendente da framework specifici permette di adattarsi facilmente a nuove tecnologie o framework. Gli sviluppatori possono sostituire facilmente un database o una libreria esterna senza impattare l'intero sistema.
- **Ottenere una chiarezza architetturale**: la Clean Architecture impone una struttura chiara e coerente che aiuta sia gli sviluppatori che i nuovi membri del team a comprendere rapidamente il funzionamento dell'applicazione.
- **Aumentare la riusabilità del codice**: grazie alla sua struttura modulare e alla separazione delle preoccupazioni, parti di codice possono essere facilmente riutilizzate in diversi progetti o parti dell'applicazione.

Nel caso specifico di CoReDex, è stata specificamente adottata una struttura con cartelle come "data", "domain" e "presentation", ognuna corrispondente a un livello della Clean Architecture. In particolare:

1. **Data**: Questo livello gestisce i dati dell'applicazione e comunica con il backend, rientrando nell'ambito degli Interface Adapters.
2. **Domain**: Include "**entities**" e "**usecase**", rispecchiando i livelli centrali della Clean Architecture. Gli "entities" rappresentano i modelli di dati e le regole di business, mentre gli "usecase" gestiscono la logica applicativa.
3. **Presentation**: Corrisponde alla UI dell'app, situata nel livello esterno della struttura a livelli concentrici.

L'adozione di questa architettura in CoReDex ha permesso di costruire un'applicazione mobile robusta, facilmente estensibile e manutenibile nel tempo. La struttura di ogni livello verrà descritta nel dettaglio all'interno di questo capitolo.

4.1 Cartella "Data"

La cartella "Data" in CoReDex funge da ponte tra la logica di business dell'applicazione (situata nel dominio) e il mondo esterno, come i servizi di backend o le fonti di dati esterne.

Tale cartella è suddivisa principalmente in due sottocartelle: "sources" e "repositories".

4.1.1 Sources

Qua risiedono i moduli responsabili dell'interazione diretta con il backend, effettuando varie operazioni sui dati. Ogni classe definita all'interno dei file

contenuti in questa cartella estende una classe base, "BaseDBDataSource", che fornisce un'implementazione semplificata dei metodi http standard, nascondendo la complessità delle chiamate API e gestendo in modo uniforme i risultati e gli errori.

```
export class BaseDBDataSource {
  constructor(baseUrl, baseOptions, authenticated) {
    this.baseUrl = baseUrl || BaseDBDataSource.defaultBaseUrl;
    this.baseOptions = baseOptions;

    if (authenticated) {
      AuthDataSource.subscribe((authJwt) => {
        this.baseOptions = {
          ...this.baseOptions,
          headers: { Authorization: authJwt },
        };
      });
    }
  }

  static defaultBaseUrl = API_BASE_URL;

  async get(url, options) {
    console.log("HTTP -> GET", url);
    const res = await axios.get(url, { ...this.baseOptions, ...options });
    if (res.status === 200) return res.data.data;
    return null;
  }
```

Figura 4.2 – BaseDBDataSource

Un esempio di file contenuto in questa sottocartella, che estende BaseDBDataSource, è **UsersDataSource.js**. All'interno di questo file, la classe **UsersDataSourceClass** definisce varie funzioni per eseguire richieste HTTP al server, servendosi dei metodi di BaseDBDataSource: ad esempio **getAll()** è utilizzata per recuperare tutti gli utenti utilizzando il metodo "get" che ha ereditato per fare una richiesta http GET all'endpoint "/users/".

```
class UsersDataSourceClass extends BaseDBDataSource {
    constructor () {
        super(null, {}, true);
    }

    async getAll () {
        logger.log('Getting all users');
        return await this.get(`${this.baseUrl}/users/`);
    }
```

Figura 4.3 - UsersDataSourceClass

Nel contesto di CoReDex, questi moduli forniscono un'astrazione delle operazioni sui dati, permettendo al resto dell'applicazione di rimanere indipendente dalle specifiche tecniche del servizio di backend.

4.1.2 Repositories

I repositories agiscono come intermediari tra la logica di business (rappresentata dagli usecase) e le fonti di dati (come database e API esterne), implementando la logica per la gestione dei dati e l'interazione con i "sources". Forniscono un punto centralizzato per gestire le operazioni di dati, semplificando l'accesso e la manipolazione dei dati per i livelli superiori dell'architettura.

Un esempio di repository è dato da **UsersRepository**: esso funge da punto centrale per la gestione e l'accesso ai dati degli utenti. È stato progettato per interagire con la sorgente dati ("UsersDataSource" descritta precedentemente) e fornire un'interfaccia semplificata per il recupero e la manipolazione delle informazioni relative agli utenti.

Al suo interno, oltre a UsersDataSource, vengono utilizzati altri due componenti: RepoItem e RepoDictionary.

- **RepoItem**: gestisce il recupero di singoli elementi o collezioni di dati (es. lista di tutti gli utenti). Se i dati sono già stati recuperati e non sono scaduti (in base a **expiration**), viene restituito il dato salvato; altrimenti, avviene un nuovo recupero.
- **RepoDictionary**: utilizzato per gestire una collezione di **RepoItem** indicizzata da chiavi (es. ID utente). Ogni volta che si richiede un utente specifico, verifica se è già presente nel dizionario e, in caso contrario, crea un nuovo **RepoItem** per esso.

```
class UsersRepositoryClass {
  constructor() {
    this.repo = {
      usersList: new RepoItem(
        async () => await UsersDataSource.getAll(),
        1000
      ) /* Ex: RepoItem.data = UsersDataSource.getAll() = [{user info data}, {user info data}, ...] */,
      users: new RepoDictionary(
        async (id) => await UsersDataSource.getById(id),
        1000
      ),
      /* Ex: RepoDictionary.data =
      { userId01: UsersDataSource.getById(userId01), userId02: UsersDataSource.getById(userId02) } = {
        userId01: {user info data},
        userId02: {user info data},
        userId12: {user info data},
        userId777: {user info data},
      } */
    };
    logger.log("Repository created");
  }

  async getAll() {
    const res = await this.repo.usersList.get();
    return res;
  }
```

Figura 4.4 - UsersRepositoryClass

In questo esempio, la funzione getAll() utilizza “usersList” (un “RepoItem”) per ottenere un elenco di tutti gli utenti. La funzione è ottimizzata per ridurre le chiamate API ridondanti grazie al caching implementato in “RepoItem”.

Dal punto di vista di interazione con la Clean Architecture, UsersRepository rappresenta l'implementazione concreta del modello di repository nel livello di "Data". Astrae la logica di interazione con le API server, consentendo ai livelli superiori (domain e presentation) di rimanere disaccoppiati dalle specificità tecniche delle fonti di dati.

Per quanto riguarda il livello "Domain", UsersRepository è utilizzato negli "use cases" per eseguire operazioni legate ai dati, facilitando l'accesso ai dati necessari per la logica di business senza esporre dettagli implementativi.

Inoltre, il caching implementato riduce il carico sul server, migliorando di conseguenza la reattività dell'app. Centralizzare la logica di accesso ai dati rende il codice più pulito e facile da mantenere.

In conclusione, la cartella "Data" gioca un ruolo essenziale nell'architettura di CoReDex, supportando l'integrazione e la gestione efficiente dei dati all'interno dell'applicazione.

4.2 Cartella "Domain"

La cartella "Domain" contiene la logica di business, separandola dalle altre parti dell'applicazione, come l'interfaccia utente o l'accesso ai dati. Analizziamo le sottocartelle "entities" e "usecase" per comprendere meglio il loro ruolo e funzionamento.

4.2.1 Entities

Le "entities" sono classi o strutture che rappresentano concetti centrali nel dominio dell'applicazione. Un'entità può essere sia un oggetto dotato di metodi, sia

una collezione di strutture e funzioni dati. Questi oggetti sono il cuore della logica di business e definiscono le regole essenziali e le funzionalità dell'app. In altre parole, sono le "cose" con cui l'app lavora e su cui opera.

Tali strutture sono fondamentali in quanto rappresentano e incapsulano dati e logiche di business, separandoli dall'accesso ai dati. Inoltre, tendono ad essere stabili nel tempo, poiché rappresentano concetti di alto livello del dominio dell'app e possono essere riutilizzate in diversi scenari all'interno dell'applicazione, assicurando coerenza e riducendo la duplicazione del codice.

Un esempio di entity è "**Account.js**". Essa rappresenta un utente all'interno dell'applicazione e incapsula tutti i dettagli relativi ad esso, come l'identità (id, e-mail, nome, cognome), lo stato di login (gestito con jwt token) e le operazioni che possono essere eseguite sull'account (login, logout, sign up).

```
class AccountEntity {
  constructor() {
    this.user = null;
    this.jwt = "";
    this.subscribers = [];

    AuthDataSource.subscribe((jwt, user) => {
      this.user =
        user && new User(user.email.split("@")[0].replace(".", "-"), user.email, user.given_name, user.family_name);
      this.jwt = jwt; //auth

      if (this.user) {
        (async () => {
          const userData = await UsersRepository.getById(this.user.id);
          this.user.populate(userData);
          this.subscribers.map((cb) => cb(this.user));
        })();
      } else {
        this.subscribers.map((cb) => cb(this.user));
      }
    });
  }

  subscribe(cb) {
    this.subscribers.push(cb);
    cb(this.user);
  }

  logIn(email, password) {
    return AuthDataSource.logIn(email, password);
  }
```

Figura 4.5 - Account Entity

Account.js utilizza AuthDataSource e UsersRepository per le operazioni di autenticazione e recupero dei dati. Questo mostra l'integrazione tra i livelli di "Domain" e "Data", ma mantiene la logica di business separata dall'effettivo accesso ai dati.

All'interno della classe AccountEntity viene gestito lo stato dell'utente corrente e sono forniti i metodi per la sottoscrizione a cambiamenti dello stato. Questo è cruciale per mantenere un flusso di dati coerente all'interno dell'applicazione.

4.2.2 Usecase

La sottocartella "usecase" contiene classi e funzioni che implementano la logica di business dell'applicazione. Gli usecase sono responsabili di eseguire operazioni specifiche, manipolando le entities e interagendo con il livello "Data". Ogni usecase è responsabile di una specifica operazione.

Questa suddivisione in funzioni specifiche permette sia di concentrarsi su un singolo aspetto della logica di business, rendendo il codice più chiaro, sia di modificare facilmente tale logica, poiché le modifiche sono contenute all'interno di usecase specifici.

Un esempio di "usecase" è **ListUserColleaguesUC.js**. Il suo compito è elencare i colleghi di un utente specifico.

```
export const ListUserColleagues = async (id) => {
  logger.log(`Getting colleagues of ${id}`);
  const colleagues = await UsersRepository.getColleagues();
  return colleagues.filter((u) => u.id !== id);
};
```

Figura 4.6 - ListUserColleagues usecase

Esso dimostra l'astrazione della logica di business (elencare colleghi) dalla logica di accesso ai dati (interrogare un repository).

Al suo interno sono nascosti i dettagli su come i dati sono effettivamente recuperati e gestiti, fornendo un'interfaccia semplice per il resto dell'applicazione. Per accedere ai dati necessari, lo usecase in esempio usa UsersRepository, evidenziando l'interazione tra gli usecase e i repositories definiti nel livello "Data".

4.3 Cartella "Presentation"

La cartella "Presentation" ospita i componenti dell'interfaccia utente (UI), comprese le schermate (screens) e i componenti più piccoli utilizzati per costruirle. Questa cartella si concentra esclusivamente sulla logica di presentazione, ben separata dalla logica di business, gestendo come i dati vengono visualizzati e come l'utente può interagire con essi. Tale separazione tra le due logiche consente agli sviluppatori di modificare l'interfaccia utente senza influenzare la logica sottostante.

La cartella è suddivisa in due sottocartelle "Screens" e "Components".

4.3.1 Screens

Gli screens sono i componenti di più alto livello nella gerarchia dell'UI, rappresentano intere schermate o pagine all'interno dell'applicazione. Ogni schermata è responsabile di una specifica funzione o vista, come la visualizzazione del profilo dell'utente o la gestione di un form di login.

Un esempio di screen è **ColleaguesListScreen**. Al suo interno è mostrato l'elenco di colleghi dell'utente, permettendogli di interagire con loro.

```
function ColleaguesListScreen() {
  const [replyersList, setReplyersList] = useState([]);
  const [unknownReplyersList, setUnknownReplyersList] = useState([]);
  const [metReplyersList, setMetReplyersList] = useState([]);

  let auth = useAuth();
  let colleaguesCtx = useColleaguesContext();
  const colleagues = colleaguesCtx && colleaguesCtx.userColleagues;

  useEffect(() => {
    if (colleaguesCtx.userColleagues.length === 0) {
      ListUserColleagues(auth.user.id)
        .then((res) => {
          colleaguesCtx.updateUserColleagues(res);
        })
        .catch((err) => console.error(err));
    }
  }, []);

  {/* .... */}
```

Figura 4.7 - ColleaguesListScreen

Tale componente utilizza direttamente lo usecase ListUserColleagues per ottenere i dati e li presenta attraverso componenti come ColleaguesList.

```
//fn called on return method of ColleaguesListScreen
const renderScene = ({ route }) => {
  const name = route.key.split("(")[0].trim();
  switch (name) {
    case "unknown": ···
    case "met": ···
    case "board": ···
    default:
      return (
        <ColleaguesList
          refreshing={refreshing}
          onRefresh={onRefresh}
          colleagues={colleagues}
        />
      );
  }
};
```

Figura 4.8 - Funzione che renderizza la lista di colleghi

4.3.2 Components

I components sono unità più piccole e riutilizzabili che compongono le schermate. Possono variare da pulsanti e input a elementi più complessi. Sono stati descritti nel dettaglio all'interno del capitolo relativo alla realizzazione del front-end di CoReDex.

4.4 Flusso di dati

Questa sezione esplora come i diversi componenti dell'architettura di CoReDex interagiscono, con un focus particolare sul trasferimento di dati dal backend al frontend.

Come già evidenziato nell'introduzione del capitolo, alla base della Clean Architecture c'è la separazione dei ruoli: ogni parte dell'architettura ha un ruolo specifico e ben definito, assicurando una buona organizzazione e manutenibilità del sistema. Inoltre, la comunicazione tra i diversi livelli dell'architettura (Presentation, Domain, Data) avviene in modo strutturato.

4.5 Esempi di flusso di dati

Quello che segue è un esempio di come le varie parti del sistema descritte precedentemente lavorino armonicamente tra di loro, permettendo il corretto ed efficace funzionamento dell'app.

Concentriamoci sull'esempio pratico della visualizzazione dell'elenco dei colleghi in CoReDex.

Dalla Richiesta Iniziale all'Elaborazione dei Dati

- L'utente interagisce con l'interfaccia utente (livello Presentation), ad esempio, cliccando sull'icona della bottom-bar per visualizzare l'elenco dei suoi colleghi. Si apre la schermata **ColleaguesListScreen** e la **useEffect** eseguita al mount del componente chiama lo usecase **ListUserColleagues**.
- La richiesta viene inoltrata agli usecase appropriati nel livello Domain, dove viene processata.
- Gli usecase interagiscono con il livello Data, specificamente con i repositories, per recuperare i dati richiesti dal backend. Nel nostro caso specifico, **ListUserColleagues** interagisce con **UsersRepository** per ottenere l'elenco dei colleghi.

Recupero e Presentazione dei Dati:

- Il repository, utilizzando i datasource, effettua le richieste al backend e riceve i dati. In questa fase, **UsersRepository** richiama **UsersDataSource** per effettuare una chiamata API al backend e recuperare i dati necessari.
- I dati recuperati sono poi restituiti agli usecase che li elaborano secondo le regole di business.

- Una volta ottenuti i dati, vengono passati indietro al livello Presentation attraverso il livello Domain e infine visualizzati nel componente di UI **ColleaguesListScreen.**

In conclusione, il flusso di dati in CoReDex dimostra come una struttura basata sulla Clean Architecture possa effettivamente facilitare lo sviluppo e la manutenzione di un'applicazione complessa, garantendo al contempo che ogni parte del sistema svolga il suo ruolo in modo efficiente.

5 Sviluppo Backend

Il backend di CoReDex si articola in diversi componenti chiave, ognuno dei quali svolge funzioni vitali per garantire un'esperienza utente fluida, sicura e coinvolgente.

Questo capitolo si propone di esplorare in dettaglio le tre "anime" principali del backend, delineando come ciascuna contribuisce al funzionamento generale dell'app e all'interazione con gli utenti.

1. **Gestione e Manipolazione dei Dati:** la prima e fondamentale funzione del backend di CoReDex è quella di raccogliere, immagazzinare, gestire e restituire dati. Questo include la gestione di un database dinamico con DynamoDB, l'elaborazione di richieste attraverso API REST, e l'impiego di funzioni Lambda, le quali possono essere attivate da eventi HTTP o da timer programmati. Questo sistema è la spina dorsale dell'app, assicurando che tutte le interazioni degli utenti con l'app vengano registrate, elaborate e utilizzate per migliorare l'esperienza di gioco e la personalizzazione.
2. **Autenticazione:** La gestione dell'autenticazione è un altro aspetto cruciale del backend. Implementata attraverso AWS Cognito, questo componente assicura che ogni accesso all'app sia sicuro e che i dati degli utenti siano protetti. Le user pool di Cognito sono fondamentali per mantenere un ambiente sicuro e per facilitare la gestione degli accessi e delle identità degli utenti.
3. **Portale Amministrativo:** Il backend ospita anche un portale amministrativo che utilizza le stesse API dell'app mobile. Realizzato con React e seguendo i principi della Clean Architecture, questo portale è accessibile tramite il dominio "admin.coredex.link" e sfrutta CloudFront di AWS per la

distribuzione di contenuti. Il portale offre un'interfaccia intuitiva per la gestione dell'app, come ad esempio l'approvazione della registrazione di ogni utente.

Oltre a questi componenti principali, il backend di CoReDex gestisce anche gli asset multimediali attraverso l'uso di bucket S3, separando i file multimediali (come le immagini degli utenti e delle medaglie) dalle build dell'app. Questa struttura non solo ottimizza le prestazioni dell'app, ma ne facilita anche la manutenzione e l'aggiornamento.

Infine, un aspetto fondamentale dello sviluppo del backend, è l'uso del Cloud Development Kit (CDK) di AWS. Il CDK consente una configurazione efficiente e flessibile dell'ambiente di backend utilizzando JavaScript, permettendo una gestione più agile e dinamica delle risorse cloud.

Nel corso di questo capitolo, esploreremo in dettaglio ciascuno di questi aspetti.

5.1 Gestione dei dati

Il backend di CoReDex è progettato per gestire in modo efficiente una vasta gamma di dati, che vanno dalle informazioni degli utenti alle loro interazioni all'interno dell'app.

Questa sezione si focalizza su tre componenti chiave che costituiscono l'anima del backend: il database, le API REST, e le funzioni Lambda.

5.1.1 Database con DynamoDB

CoReDex utilizza come database DynamoDB di AWS, un database NoSQL noto per la sua scalabilità e performance.

Alcune delle sue caratteristiche principali (AWS, Funzionalità di Amazon DynamoDB, s.d.) includono:

1. **Scalabilità automatica**: DynamoDB si adatta automaticamente al carico di lavoro, garantendo tempi di risposta costanti anche sotto carichi elevati.
2. **Modello di dati flessibile**: supporta sia il modello di dati chiave-valore sia quello dei documenti, offrendo flessibilità nella gestione dei dati.
3. **Performance elevate**: offre latenze nell'ordine dei millisecondi.
4. **Affidabilità**: immagazzina i dati su dischi SSD e li replica automaticamente su più Availability Zones.
5. **Costi**: per i volumi che ci sono su CoReDex, DynamoDB è gratuito.

Nel contesto di CoReDex, DynamoDB è utilizzato per memorizzare e gestire tutti i dati relativi agli utenti e alle loro interazioni. Per esemplificare, analizziamo i file '**user-schema.js**' e '**user.js**'.

Il file "**user-schema.js**" rappresenta una delle possibili entity del database: definisce lo schema per i dati degli utenti in CoReDex. Ogni attributo, come e-mail, ruolo, livello di esperienza, è definito con precisione, garantendo che i dati immessi nel database rispettino una struttura coerente e prevedibile.

```
exports.UserSchema = {
    id: {
        _type: 'string',
        _required: true
    },
    email: {
        _type: 'string',
        _required: true
    },
    status: {
        _type: 'enum',
        _enum: ['registered', 'active', 'inactive'],
        _default: 'registered'
    },
    code: {
        _type: 'number'
    },
    // Bio
    firstName: {
        _type: 'string',
        _required: true
    },
```

Figura 5.1 - Schema DB degli utenti

Questo schema facilita la validazione dei dati e la loro manipolazione nelle operazioni del database.

Il file "**user.js**" illustra come le operazioni CRUD (Create, Read, Update, Delete) relative ai vari utenti vengano implementate su DynamoDB. Questo include la creazione di nuovi utenti, la lettura delle informazioni degli utenti, l'aggiornamento dei dati e la gestione di specifiche funzionalità come il consumo delle CoReBall.

Ad esempio, la funzione **getAll()** recupera tutti gli utenti, mentre **getById()** recupera un utente specifico in base al suo ID. Queste funzioni utilizzano l'AWS SDK per interagire con DynamoDB, eseguendo operazioni efficienti e sicure sui dati.

```
static async getById (id) {
    let user = new User({id});
    console.log(user);
    console.log(user.primaryKeys);
    try {
        user = await db
            .get({
                TableName: DB_TABLE,
                Key: user.primaryKeys
            })
            .promise();
        return user && new User(user.Item).toJSON();
    } catch (error) {
        console.log(error);
        throw error;
    }
}
```

Figura 5.2 - getById implementata dentro user.js

Oltre a "user.js" esistono altri file di questo genere, ognuno per ogni schema definito e rappresentante una delle entity, come "game.js" che gestisce le operazioni CRUD relative alle challenges settimanali. Ciascuno di questi file estende la classe "DynamoEntity", che funge da base comune per le entità che interagiscono con il database DynamoDB, fornendo metodi base per gestire le operazioni sui dati prima del loro salvataggio nel database.

5.1.2 Lambda Functions

Le funzioni Lambda in AWS rappresentano un servizio di computing serverless che consente l'esecuzione di codice in risposta a eventi e l'automazione di varie attività senza la necessità di gestire il server (AWS, Caratteristiche di AWS Lambda, s.d.).

Queste funzioni sono ideali per operazioni scalabili e orientate agli eventi, che sono fondamentali nel contesto di CoReDex. Tali funzioni, infatti, sono progettate per eseguire codice in risposta a specifici trigger o eventi, come richieste HTTP, modifiche ai dati in un database, o un evento programmato. Inoltre, le funzioni Lambda possono essere facilmente integrate con altri servizi AWS, come Amazon S3, DynamoDB e API Gateway, per creare architetture complesse e reattive.

Nel contesto di CoReDex, ogni funzione Lambda è salvata in un diverso file javascript. Ad esempio, la Lambda function "**users.js"** gestisce le richieste relative agli utenti. Questa funzione agisce come un controller che risponde alle richieste HTTP per operazioni come la creazione, l'aggiornamento, il recupero e l'eliminazione di utenti. Al suo interno utilizza un router per mappare le richieste HTTP ai relativi handler.

A titolo di esempio analizziamo una richiesta GET a **/users**, innescata da un'azione dell'utente come l'apertura della schermata con l'elenco degli utenti. Questa richiesta raggiungerà il backend di CoReDex per poi essere indirizzata a una specifica funzione Lambda tramite il servizio API Gateway di AWS. L'API Gateway agisce come un intermediario che riceve le richieste HTTP e le inoltra alle appropriate funzioni Lambda.

L'endpoint **/users** è gestito dalla funzione Lambda definita nel file **users.js**. Questa funzione è programmata per eseguire una serie di operazioni quando viene attivata da una richiesta GET a tale endpoint, come il recupero dell'elenco degli utenti dal database. Successivamente, crea una risposta che include tale elenco e la invia al client.

```
router.get('/users', async (event, context) => {
  try {
    const users = await User.getAll();
    return APIResponse({
      count: users.length,
      data: users,
      event: {path: event.path, params: event.pathParameters}
    });
  } catch (err) {
    console.log(err);
    return APIError(err);  }
});
```

Figura 5.3 - GET /users

Infine, il client che ha inviato la richiesta (ad esempio, l'app CoReDex sul dispositivo dell'utente), riceve l'elenco degli utenti e li visualizza di conseguenza nella relativa schermata dell'app.

Alcune funzioni Lambda in CoReDex sono programmate per eseguire compiti automaticamente in momenti specifici. Il file "**week-starter.js**" viene eseguito in base a una pianificazione temporale ogni domenica: si connette a DynamoDB, recupera i dati degli utenti e per ognuno verifica se è necessario ricaricare le CoReBalls. In caso positivo aggiorna il loro conteggio nel database. Lo script gestisce condizioni specifiche, come il non superamento del limite massimo di CoReBalls (20 per ogni utente) e l'aggiornamento del conteggio solo se necessario.

5.1.3 REST API

Una REST API (Representational State Transfer Application Programming Interface) è un'interfaccia utilizzata per lo scambio di informazioni in modo sicuro tra due sistemi informatici su Internet (AWS, Cos'è un'API RESTful?, s.d.).

Le API RESTful facilitano questi scambi di informazioni seguendo standard di comunicazione software che sono sicuri, affidabili ed efficienti.

Operano su protocolli HTTP standard e utilizzano metodi HTTP come GET, POST, PUT e DELETE per la manipolazione delle risorse. Le API REST di CoReDex sono state progettate per garantire una comunicazione efficiente e flessibile tra il backend e l'applicazione frontend. Alcune delle loro caratteristiche sono le seguenti:

1. **Endpoint Chiari e Ben Definiti**: Ogni endpoint delle API REST è stato creato per gestire specifiche risorse, come utenti, relazioni tra utenti o eventi. Ad esempio, l'endpoint **/users** è utilizzato per recuperare informazioni sugli utenti.
2. **Utilizzo di Metodi HTTP Standard**: Le API sfruttano i metodi HTTP come GET per recuperare dati, POST per creare nuove risorse, PUT per aggiornare risorse esistenti e DELETE per rimuovere risorse.
3. **Formato dei Dati**: Le API REST di CoReDex scambiano dati prevalentemente in formato JSON, un formato leggero e facilmente manipolabile sia in JavaScript lato client che server.

Per quanto riguarda l'implementazione delle REST API in CoReDex, si utilizza un sistema di routing e di gestione delle richieste: il sistema di routing indirizza le richieste HTTP ai corretti handler.

Le API gestiscono inoltre gli errori restituendo codici di stato http appropriati e messaggi di errore descrittivi, aiutando a capire facilmente quando qualcosa va storto e perché.

Dal punto di vista della sicurezza, CoReDex utilizza Cognito per l'autenticazione degli utenti. Ogni richiesta delle API è accompagnata da un token JWT (JSON Web

Token) per garantire che l'utente sia autenticato e autorizzato ad accedere alla risorsa richiesta.

Le API REST di CoReDex, dunque, forniscono un modo standardizzato, sicuro ed efficiente per gestire le richieste e trasferire dati.

5.2 Autenticazione

CoReDex utilizza AWS Cognito per la gestione dell'autenticazione.

AWS Cognito è un servizio di AWS che fornisce autenticazione, autorizzazione e gestione degli utenti per applicazioni web e mobile. Con Cognito, gli sviluppatori possono facilmente aggiungere funzionalità di accesso e controllo degli accessi stessi alle loro applicazioni. Questo servizio offre anche la gestione delle sessioni utente, l'integrazione con altre AWS services e la sicurezza tramite token JWT (AWS, What is Amazon Cognito?, s.d.).

AWS Cognito è stato scelto per la sua robustezza e flessibilità, poiché integra funzionalità di autenticazione, autorizzazione e gestione degli utenti.

Per gestire gli utenti si utilizza AWS Cognito User Pool, un servizio che permette agli sviluppatori di aggiungere funzioni di registrazione, accesso e controllo degli accessi alle applicazioni web e mobile. Offre anche funzionalità come la verifica dell'identità e il recupero della password degli utenti.

Il setup della User Pool è stato realizzato con AWS CDK definendo un "UserPool" e un "UserPoolClient" all'interno di uno stack CDK. Questo processo è illustrato nei file "**user-pool.js**" e "**cdk-stack.js**".

- **user-pool.js**: Questo file contiene la definizione dello User Pool, compresi i dettagli come la policy delle password, gli attributi richiesti per gli utenti

(come nome e cognome), e le configurazioni relative al recupero dell'account. Include anche la creazione di un dominio per la User Pool e un client per la stessa.

- **cdk-stack.js**: Questo file mostra come lo stack CDK viene configurato per includere la User Pool. Viene inoltre mostrato come vengono gestite le dipendenze e le connessioni tra diverse risorse AWS, come le Lambda Functions e la User Pool.

Viene creato uno UserPool e configurato un UserPoolClient per determinare come l'applicazione client interagisce con lo UserPool. Vengono impostate le opzioni di dominio per lo UserPool, definendo come gli utenti accederanno al servizio di autenticazione. Infine, una funzione Lambda (**postUserRegistrationLambda** definita in cdk-stack.js) è utilizzata per eseguire operazioni post-conferma, come l'aggiornamento di tabelle nel database.

Per quanto riguarda l'integrazione con l'applicazione, l'ID e il ClientID dello UserPool vengono esposti come output dallo stack CDK, permettendo all'applicazione client di utilizzarli per interagire con Cognito. L'app è così in grado di implementare funzionalità come la registrazione, il login e la gestione dei profili utente.

5.3 Portale Web Amministrativo

Il portale amministrativo di CoReDex è un'applicazione web progettata per facilitare la gestione e l'amministrazione dell'app mobile. Questo portale consente agli amministratori di eseguire principalmente operazioni relative alla gestione degli utenti, come l'approvazione della loro registrazione e dell'upload di nuove immagini del profilo.

Il portale è stato realizzato con React. Dal punto di vista dell'interfaccia utente, la web app si mostra come intuitiva e reattiva, consentendo agli admin di visualizzare intuitivamente l'elenco degli utenti e le loro caratteristiche principali.

Figura 5.4 - Sezione Utenti del portale di amministrazione

Così come la mobile app, anch'essa incorpora i principi della Clean Architecture per assicurare una struttura del codice chiara e manutenibile.

Dal punto di vista delle API REST, il web portal utilizza le stessi API usate dall'applicazione mobile.

5.3.1 Gestione degli utenti

Tramite il portale, è possibile visualizzare l'elenco di tutti gli utenti iscritti a CoReDex. Vengono visualizzati automaticamente per primi tutti gli utenti che richiedono un'approvazione (nuovi utenti appena registrati oppure utenti già

approvati in passato che richiedono una modifica della propria immagine del profilo).

Ogni nuovo utente, al momento della registrazione, è tenuto a caricare una foto in cui è visibile il proprio volto. Prima che l'utente possa accedere pienamente alle funzionalità dell'app, la sua foto deve essere approvata da un amministratore. Questo passaggio è necessario per far passare lo status dell'utente da "registrato" ad "attivo".

Questo sistema di approvazione non riguarda soltanto i nuovi utenti, ma è esteso anche a quelli già attivi. Infatti, ogni qual volta un utente esistente decide di cambiare la propria immagine del profilo, la nuova foto deve ricevere l'approvazione degli amministratori prima di diventare visibile agli altri utenti di CoReDex. Fino all'approvazione, sia l'utente che ha effettuato la modifica sia gli altri membri della piattaforma continueranno a vedere l'immagine precedente.

5.3.2 Tecnologia e infrastruttura

Il portale di amministrazione è ospitato su Amazon S3 e distribuito globalmente tramite Amazon CloudFront.

AWS CloudFront è un servizio di rete di distribuzione dei contenuti (CDN) offerto da Amazon Web Services. CloudFront distribuisce in modo efficiente i contenuti agli utenti finali con bassa latenza e alte velocità di trasferimento. Utilizza una rete globale di data center, chiamati edge locations, per memorizzare temporaneamente (cache) i contenuti più vicino agli utenti finali, riducendo i tempi di caricamento e migliorando la reattività dell'applicazione (AWS, Amazon CloudFront, s.d.).

Utilizzare Amazon CloudFront ha portato i seguenti benefici:

- **Prestazioni Ottimizzate**: Grazie alla cache dei contenuti presso gli edge locations, il portale di amministrazione carica più rapidamente, indipendentemente dalla posizione geografica dell'utente.
- **Scalabilità**: CloudFront gestisce automaticamente gli aumenti del traffico, garantendo che il portale rimanga operativo e reattivo anche durante i picchi di accessi.

L'integrazione del dominio del portale di amministrazione con CloudFront assicura che le richieste al dominio siano gestite efficacemente tramite il CDN. Inoltre, grazie all'utilizzo del servizio DNS di AWS, il dominio è stato configurato per risolvere in modo ottimale gli indirizzi IP degli edge locations di CloudFront, semplificando la gestione del traffico e migliorando la resilienza del sistema.

Oltre ad Amazon CloudFront, è stato utilizzato **Amazon Simple Storage Service (S3)**, un servizio di storage offerto da AWS. S3 permette di memorizzare e recuperare qualsiasi quantità di dati, in qualsiasi momento, da qualsiasi parte del web. Fornisce soluzioni di storage affidabili, scalabili e sicure (AWS, Amazon S3, s.d.).

CoReDex utilizza Amazon S3 per gestire gli asset e i contenuti multimediali tramite due bucket S3:

- **Bucket per gli asset multimediali**: Questo bucket ospita le immagini associate a ciascun utente, le immagini delle medaglie e quelle caricate dagli utenti nella sezione "eventi" dell'app. La sua capacità di gestire grandi volumi di dati assicura una scalabilità ottimale e garantisce la costante disponibilità e protezione degli asset da eventuali perdite o danni. In aggiunta, l'uso di S3 alleggerisce il carico sul server dell'applicazione,

affidando la gestione del traffico relativo agli asset direttamente al servizio S3 (AWS, Amazon Simple Storage Service Documentation, s.d.).

- **Bucket per le build dell'app**: Questo bucket contiene le build dell'applicazione, facilitando la distribuzione e l'aggiornamento della stessa senza incrementarne eccessivamente le dimensioni sul dispositivo dell'utente. Grazie a questo sistema, immagini e altri asset vengono scaricati solo quando necessario. La collocazione delle build dell'app su un bucket S3 semplifica notevolmente il processo di aggiornamento e distribuzione delle nuove versioni dell'applicazione.

5.4 Considerazioni finali

La realizzazione del backend di CoReDex rappresenta un'esemplificazione dell'efficacia e della flessibilità offerte dalle soluzioni cloud moderne. Grazie all'utilizzo di servizi AWS come DynamoDB, Lambda, Cognito e S3, si è ottenuto un sistema robusto e scalabile, capace di gestire efficacemente i dati degli utenti, le interazioni con l'applicazione e l'autenticazione.

Il portale amministrativo, sviluppato con React e implementato su AWS CloudFront, sfrutta le stesse API dell'app mobile, garantendo coerenza e facilità di gestione. L'adozione di S3 per l'hosting delle immagini e delle build dell'applicazione riduce il carico sul server, ottimizzando la distribuzione e l'aggiornamento dell'app.

Il backend è stato realizzato usando il Cloud Development Kit (CDK) di AWS. Il CDK è un framework open source che consente di definire l'infrastruttura cloud del backend utilizzando linguaggi di programmazione familiari, come JavaScript, TypeScript, Python, Java e C#. Questo strumento semplifica la creazione e il

deployment di risorse AWS complesse, permettendo agli sviluppatori di utilizzare costrutti di alto livello e astrazioni, riducendo così la complessità e accelerando il processo di sviluppo (AWS, What is the AWS CDK?, s.d.).

L'uso del CDK ha permesso una gestione più efficiente e una configurazione più rapida del backend, enfatizzando la sinergia tra sviluppo applicativo e infrastrutturale.

In sintesi, la combinazione di tecnologie all'avanguardia e l'approccio moderno alla gestione dell'infrastruttura cloud hanno reso il backend un sistema performante, in linea con le esigenze di un'applicazione mobile innovativa come CoReDex.

6 Meccanismi di coinvolgimento utente

Il presente capitolo si dedica all'analisi dei meccanismi implementati per incrementare l'engagement degli utenti dall'istante del lancio dell'app, estendendosi fino a tre mesi di utilizzo. Durante questo periodo, CoReDex ha adottato strategie in-app e offline per garantire un costante coinvolgimento degli utenti, pur trovandosi ancora in una fase iniziale. Un obiettivo rilevante in vista dello sviluppo futuro è l'intensificazione, sia in qualità che in quantità, di queste tattiche di coinvolgimento, allo scopo di conservare e alimentare la passione e la sfida che gli utenti trovano in CoReDex.

L'app è emersa senza una strategia di marketing convenzionale, essendo concepita come applicazione interna non ufficiale e lanciata come una "sorpresa" alla conclusione dell'evento estivo di Concept Reply nel giugno 2023. Il coinvolgimento degli utenti è stato amplificato sfruttando tecniche di engagement sia in-app — attraverso l'implementazione di principi di gamification — sia off-app, collegando eventi aziendali fisici e virtuali e integrandoli nella sezione "Social" di CoReDex. La visione a lungo termine è di sviluppare un ecosistema transmediale, suscitando nei fruitori un ancora più profondo interesse per l'universo di CoReDex.

6.1 In-App Engagement

Gli strumenti più efficaci a disposizione per catalizzare l'engagement in-app sono indubbiamente i meccanismi di gamification. Questi non sono introdotti meramente per trasferire in app elementi ludici, ma sono strategie calibrate per

innescare la motivazione intrinseca degli utenti, incentivandoli a navigare e interagire con le diverse funzioni proposte.

La prima versione di CoReDex ha introdotto sfide settimanali, livelli, e sistemi di punteggio, meccanismi che non solo iniettano elementi ludici, rendendo l'esperienza più piacevole e stimolante, ma instillano anche un senso palpabile di progresso e realizzazione. Il meccanismo dei "livelli" si è rivelato particolarmente efficace nell'incoraggiare la partecipazione attiva degli utenti, suscitando una competizione amichevole per la conquista dei primi posti in classifica e l'accumulo di medaglie. Prossime iterazioni dell'app prevedono di capitalizzare ulteriormente su queste dinamiche, rendendo accessibili certi contenuti esclusivamente a utenti con un numero sufficiente di punti esperienza, premiando così la loro fedeltà e impegno non solo con riconoscimenti virtuali, ma anche con premi tangibili.

Il sistema di Leaderboard incorporato instilla un senso di competizione positiva e costruttiva all'interno della comunità, fungendo da catalizzatore motivazionale. Visualizzare il proprio nome ascendere nella classifica può infondere un senso di realizzazione; simultaneamente, osservare i traguardi raggiunti da altri può servire da ispirazione e sprone a una maggiore partecipazione.

Il sistema di ricompense implementato in CoReDex è studiato con attenzione per bilanciare gratificazioni immediate e premi a lungo termine, dall'accumulo di punti esperienza per azioni quotidiane, a medaglie speciali assegnate per obiettivi significativi.

Il processo di personalizzazione in app rappresenta un altro pilastro dell'engagement, offrendo agli utenti la possibilità di modellare i propri profili in maniera distintiva. Questo livello di personalizzazione, che va dalla scelta di avatar che evocano la nostalgia dei vecchi giochi Pokémon, alla descrizione di progetti

lavorativi personali, non solo amplifica il senso di individualità, ma fortifica anche il senso di appartenenza verso l'applicazione.

Un sistema di notifiche intelligente è in fase di sviluppo, progettato per informare gli utenti su aggiornamenti, sfide disponibili e successi raggiunti senza inondarli di messaggi.

Infine, il sistema di feedback integrato in CoReDex rappresenta un canale cruciale per il miglioramento continuo dell'esperienza in-app. Gli utenti sono invitati a lasciare commenti, suggerimenti e recensioni, contribuendo così a un ciclo di feedback positivo che informa e guida lo sviluppo futuro dell'app.

6.2 Off-App Engagement

Il coinvolgimento degli utenti trascende l'ambiente digitale, e le strategie di engagement implementate da CoReDex si estendono anche al mondo offline, articolandosi attraverso eventi, premiazioni e avvalendosi della progettazione di un universo transmediale.

Uno degli approcci salienti nell'engagement off-app è l'organizzazione di eventi aziendali speciali. Questi eventi funzionano non solo come luoghi di socializzazione e interazione tra i membri del team, ma anche come contesti in cui viene promosso l'uso attivo di CoReDex. Durante il loro svolgimento, entra in gioco la Leaderboard di CoReDex, permettendo di assegnare premi fisici e distintivi ai migliori utenti in classifica, sotto forma di t-shirt personalizzate, gadget esclusivi ed esperienze uniche, favorendo inoltre l'organizzazione di sessioni di team-building off-app correlate all'universo CoReMon. Questi premi tangibili funzionano come incentivi aggiuntivi, stimolando una partecipazione attiva e continua da parte degli utenti, sia in app- che off-app.

L'universo di CoReDex è stato introdotto attraverso un'esperienza transmediale. Il termine "transmedia" si riferisce a una narrativa che si sviluppa attraverso diversi formati e piattaforme media, con ciascun medium che fornisce un contributo unico e prezioso alla narrazione complessiva, collaborando sinergicamente per creare un'esperienza narrativa olistica e coinvolgente (Jenkins, 2007). I vari media non si sovrappongono, ma ognuno di essi propone esperienze diverse che, concatenate, formano l'esperienza finale del prodotto mediale. In questo contesto, dunque, gli utenti sono invitati a cercare nuovi contenuti attraverso vari canali, esplorando il mondo della narrazione in modi unici e interessanti.

Durante l'evento di lancio, è stato presentato un video di engagement in cui il partner di Concept, trasformato nel carismatico Professor CoReMon, ha invitato gli utenti a unirsi all'avventura di CoReDex, incoraggiandoli ad iscriversi e diventare parte integrante del mondo CoReMon. Questo approccio ha fornito agli utenti un punto di ingresso, o "Rabbit Hole", nel mondo dei CoReMon, contestualizzando l'app e l'esperienza che si vuole fornire: il video ha agito come catalizzatore, incoraggiando la curiosità degli utenti.

Infine, un'altra caratteristica fondamentale per l'engagement off-app è data dall'incoraggiamento all'interazione fisica tra gli utenti. Per "catturare" i colleghi, gli utenti devono intraprendere una ricerca fisica, muovendosi attraverso gli spazi dell'ufficio e interagendo direttamente. Questa dinamica di "caccia al tesoro" non solo aggiunge un livello di gioco e divertimento all'interazione, ma favorisce anche la collaborazione e la comunicazione all'interno del team, mimando quella che è una delle meccaniche di gioco alla base dell'universo Pokémon. La necessità di scansionare i QR code reciproci per procedere nel gioco implica un coinvolgimento diretto e personale, rafforzando i legami tra colleghi.

7 CoReDex: l'app e le sue sezioni chiave

Dopo aver analizzato in dettaglio i concetti e le tecnologie fondamentali che sostengono CoReDex, ci concentreremo ora sulla descrizione delle specifiche sezioni dell'applicazione.

Abbiamo precedentemente esaminato sia il processo di design dell'esperienza utente sia i meccanismi di gamification che ne sono alla base. Ora, l'obiettivo è quello di delineare le funzionalità, la logica operativa e l'interfaccia di ciascuna delle schermate dell'app.

In particolare, esploreremo:

- **CoReMons**: è il cuore pulsante dell'applicazione, qua gli utenti possono navigare tra i diversi CoReMons e consultare la leaderboard.
- **Profile**: la sezione dedicata al profilo dell'utente, offre un riepilogo delle informazioni del profilo e dei progressi raggiunti.
- **Meet**: sezione dedicata all'interazione tra CoReMon, permette all'utente di catturare i colleghi e di farsi catturare da loro.
- **Concept**: bacheca informativa legata al mondo dell'azienda e agli eventi live, arricchita durante occasioni speciali da una sezione social interattiva.
- **CoReGym**: sezione dedicata alle challenges settimanali.

Attraverso questo capitolo, il lettore avrà la possibilità di immergersi più profondamente nell'esperienza CoReDex, comprendendo come ogni componente dell'app sia stato progettato e implementato.

7.1 CoReMons

La sezione CoReMons agisce come home dell'applicazione, costituendo il nucleo centrale dell'esperienza utente. In questo spazio, gli utenti navigano tra vari elementi interattivi e ottengono una visione complessiva del proprio CoReDex, visualizzando quali CoReMon sono stati catturati e quali restano ancora da scoprire in termini di dettagli.

All'apertura dell'app, successivamente allo splash screen, l'utente viene accolto all'interno della home. Questa sezione si caratterizza per un elenco di cards disposte su due colonne, ciascuna delle quali rappresenta un diverso CoReMon. Queste schede funzionano come portali interattivi attraverso i quali gli utenti possono accedere a informazioni dettagliate su ogni CoReMon.

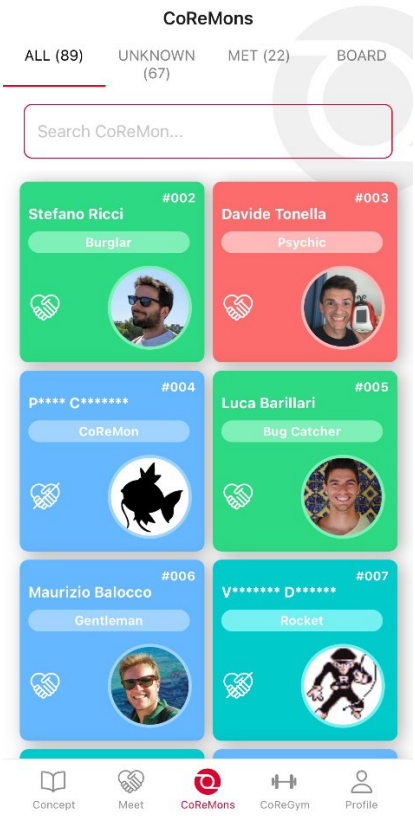

Figura 7.1 - Home, sezione CoReMons

Una barra dei filtri e una barra di ricerca sono posizionate nella parte superiore dell'interfaccia. La barra dei filtri permette di organizzare e visualizzare i CoReMon in tre categorie: "all", che elenca tutti gli utenti attivi; "unknown", che elenca i colleghi non ancora acquisiti e di cui si ignorano i dettagli; e "met", che mostra l'elenco dei CoReMon acquisiti dall'utente. La barra di ricerca facilita il filtraggio dei CoReMon: inserendo il nome di un collega, gli utenti possono verificare la sua presenza nel sistema e determinare se è già stato acquisito.

Ogni scheda fornisce informazioni preliminari sul CoReMon: un codice identificativo numerico (che rimanda ai codici numerici associati univocamente ad ogni Pokémon), il nome del collega, e la tipologia di allenatore selezionata, accompagnate da un'icona che indica se il CoReMon è stato o meno catturato (due mani che mimano il gesto di incontro tra colleghi) e la foto scelta dal collega.

Se la scheda riguarda un collega non acquisito, alcune informazioni vengono nascoste, lasciando visibili solo il codice identificativo e la tipologia di allenatore. Il nome viene parzialmente occultato, mostrando solo le iniziali, e la foto del profilo è sostituita dall'avatar corrispondente al tipo di allenatore. In assenza di una tipologia di allenatore definita, viene visualizzata l'immagine silhouette di un Magikarp, in omaggio al franchise Pokémon.

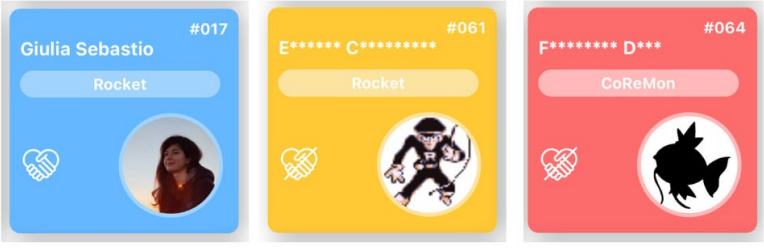

Figura 7.2 - Le tre tipologie di schede

Se l'utente seleziona la scheda di un collega già catturato, mediante una transizione di tipo “slide”, si troverà all'interno del profilo del collega in questione. Questa sezione sfoggia colori dominanti che rispecchiano la scelta cromatica effettuata dall'utente durante la personalizzazione del proprio profilo. Nel profilo del collega, la foto dell'utente è visualizzata in primo piano, accompagnata da dettagli articolati in due categorie: “Info” e “Medals”.

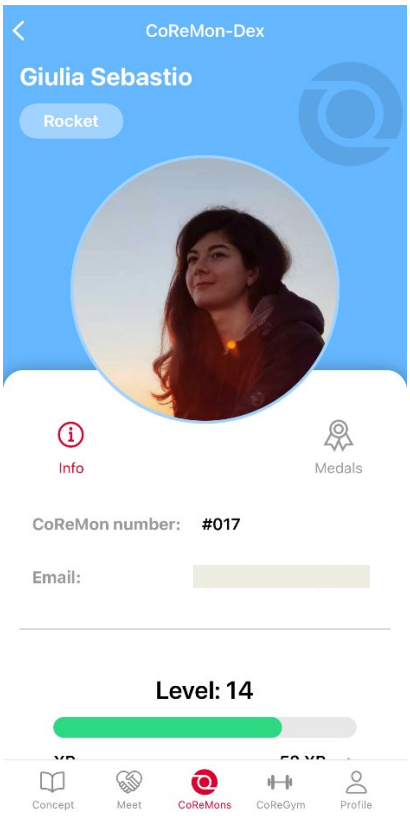

Figura 7.3 - Profilo collega

La scheda “Info”, visibile di default all'accesso del profilo, fornisce una serie di dati. Essa riporta, oltre al codice identificativo nel CoReDex e all’indirizzo e-mail aziendale del collega, elementi legati alla gamification e alla competizione tra utenti, quali il livello dell'utente e una barra progresso degli XP points. Questa barra,

consente di comprendere quanti punti XP il collega necessita per avanzare al livello successivo. Sotto alla barra degli XP sono indicate ulteriori informazioni, quali il numero di CoReMons catturati e il numero di challenges settimanali completate.

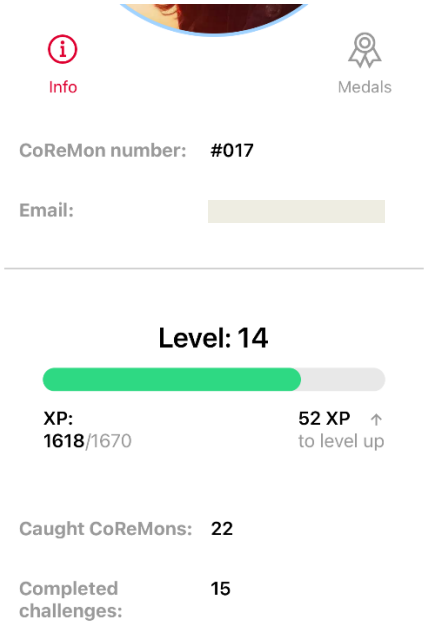

Figura 7.4 - Scheda Info

La scheda "Medals" elenca le medaglie che il collega ha accumulato nel corso del gioco. Alcune medaglie possono essere ottenute una sola volta (per esempio durante eventi aziendali) o assegnate manualmente dagli amministratori (come la medaglia "Firestarter"). Altre medaglie presentano "livelli", distinti visivamente dal numero di stelle posizionate di esse. Un numero maggiore di stelle rende la medaglia più prestigiosa, simbolizzando un avanzamento significativo in un determinato ambito dell'applicazione.

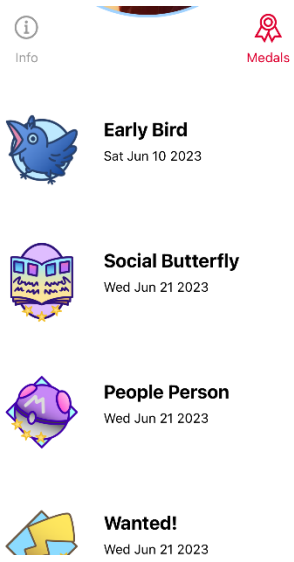

Figura 7.5 - Scheda Medals

Interagendo con le medaglie, selezionando l'icona corrispondente, l'utente attiverà un modal contenente descrizioni dettagliate relative al distintivo ottenuto dal collega.

Figura 7.6 - Dettaglio medaglia

Ritornando alla sezione CoReMons, l'ultima opzione disponibile nell'header è il tab "Board". Selezionandolo, gli utenti saranno indirizzati alla leaderboard. Questa schermata presenta una classifica che elenca i dieci utenti più attivi e con il maggior

numero di punti accumulati nell'app, oltre a illustrare un podio dove sono messi in risalto i primi tre partecipanti.

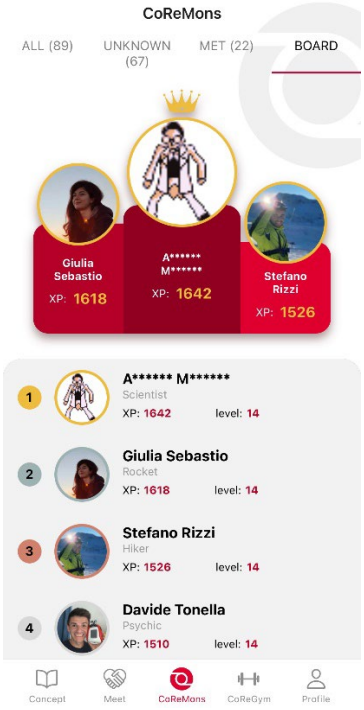

Figura 7.7 - Sezione Leaderboard

7.2 Profile

Accedendo alla sezione "Profile" attraverso l'icona corrispondente posizionata sulla navbar principale dell'app, gli utenti si ritroveranno all'interno del proprio profilo. La veste grafica di questa sezione ricorda strettamente quelle dedicate ai profili dei colleghi, con l'aggiunta di due elementi distintivi: un hamburger menu collocato in alto a destra e un'icona raffigurante una CoReBall, accanto alla quale è indicato il numero di CoReBall disponibili.

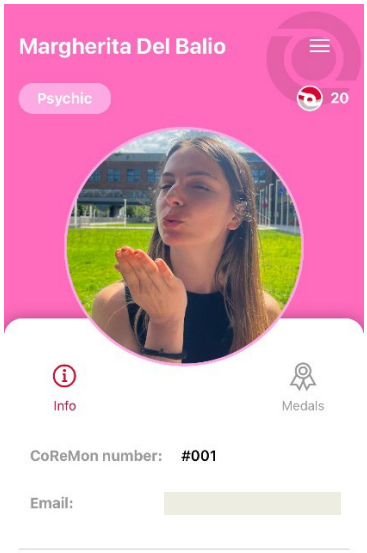

Figura 7.8 - Profilo Utente

Toccando l'icona della CoReBall, un modale spiega all'utente la funzione di queste sfere: servono per "catturare" i colleghi nella sezione "Meet" dell'app. Gli utenti non godono di tentativi di cattura illimitati; infatti, possono detenere un massimo di 20 CoReBall.

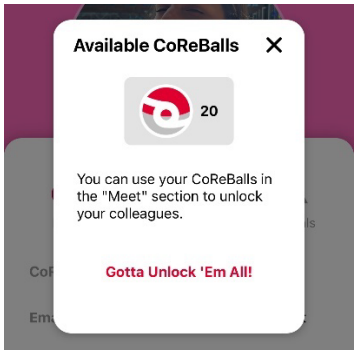

Figura 7.9 - Dettaglio CoReBalls

Ogni settimana, il loro arsenale viene automaticamente rifornito di 5 nuove CoReBall. Questo meccanismo si inserisce nell'ambito della gamification, generando un senso di attesa e stimolando la riflessione su una strategia di gioco

ottimale per completare il proprio CoReDex in tempi brevi. Ad esempio, durante un evento aziendale con partecipazione di colleghi da diverse regioni italiane, un utente potrebbe decidere di utilizzare le preziose CoReBall per catturare colleghi che lavorano in uffici lontani, con i quali avrebbe rare occasioni di interazione diretta.

Selezionando l'hamburger menu, mediante una transizione dal basso, si presenterà un elenco di opzioni: "Edit Your Profile", che indirizza alla sezione dell'app destinata all'aggiornamento delle informazioni personali; "Share Your QR Code", che fornisce un accesso diretto alla sezione "Meet", consentendo di condividere il proprio codice QR per facilitare la "cattura" da parte degli altri; "App Info", che visualizza dettagli sulla realizzazione dell'applicazione; e "Logout", che permette di uscire dall'account nell'app.

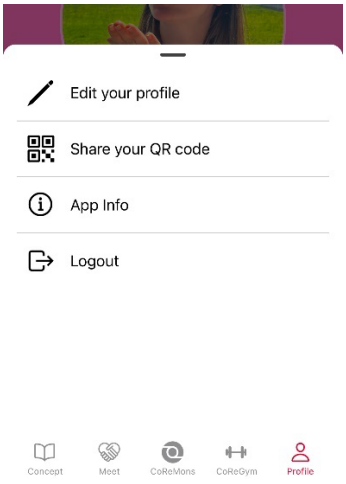

Figura 7.10 - Dettaglio Menu

Nella sottosezione dedicata alla modifica del profilo, l'utente ha la possibilità di aggiornare: la propria foto del profilo, che deve necessariamente ritrarre il volto dell'utente per garantire il riconoscimento da parte degli altri partecipanti; il colore tema del profilo, che influenzerà anche la colorazione della scheda personale

visualizzata nella sezione CoReMons degli altri utenti; e la propria Trainer Class, associata a un avatar specifico. Quest'ultimo sarà mostrato agli altri CoReMons, nel caso in cui l'utente non sia ancora stato "catturato" dai colleghi.

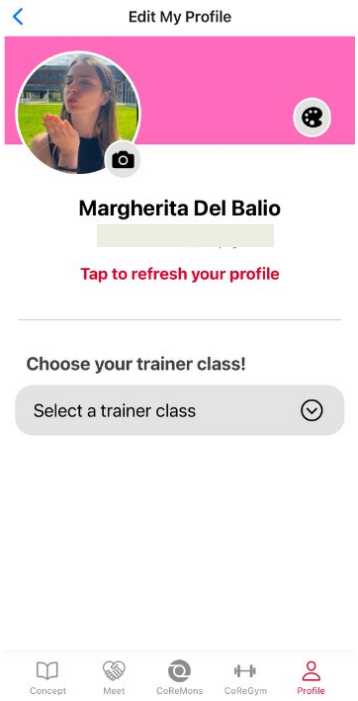

Figura 7.11 - Schermata Edit Profile

Per quanto concerne la selezione della foto profilo, l'utente, che può decidere di scattare una nuova foto o selezionarne una dalla galleria del dispositivo, non visualizzerà immediatamente la nuova immagine. Invece, vedrà una preview della foto, che risulterà essere in attesa di approvazione. Questo processo è necessario per prevenire che gli utenti carichino immagini non rappresentative di sé stessi. Pertanto, ogni immagine caricata necessita di un'approvazione da parte degli amministratori attraverso il portale web dedicato.

Se l'utente seleziona "App Info" dal menu, avrà accesso alla sezione "About This App". In questa area sono forniti dettagli circa l'implementazione dell'applicazione,

il suo obiettivo principale e le tecnologie impiegate durante il suo sviluppo. In aggiunta, è presente un pulsante che indirizza verso un modulo da compilare per segnalare problemi riscontrati nell'utilizzo dell'app. Questa sezione include anche un elenco degli sviluppatori che hanno contribuito al progetto, con l'indicazione del loro ruolo predominante all'interno del team di sviluppo.

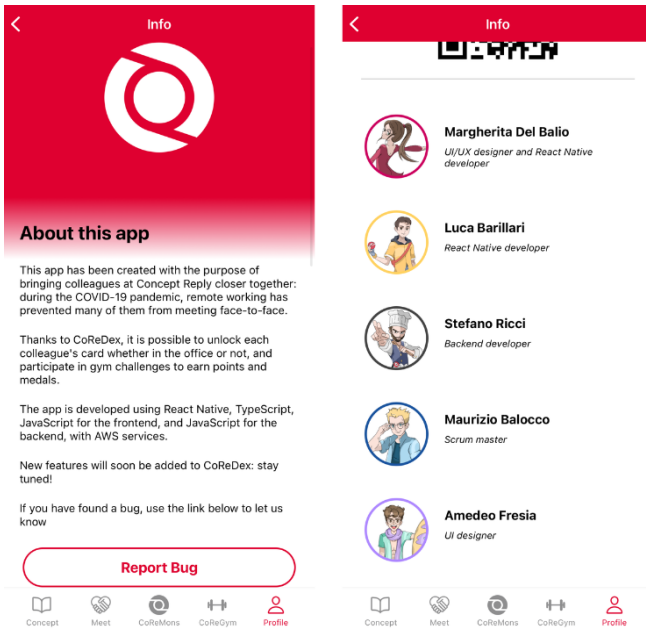

Figura 7.12 - Schermata App Info

7.3 Meet

Questa sezione, accessibile toccando la relativa icona nella navbar dell'app, è divisa in due schede: "Show QR" e "Meet a Coremon".

Il tab "show QR" mostra il codice QR relativo al proprio profilo. Quando un altro utente intende "catturare" l'utente loggato in app, deve semplicemente scannerizzare il codice QR mostrato, utilizzando lo scanner presente nella scheda "Meet a CoreMon".

Figura 7.13 - Sezione Show QR

Il QR code, criptato e privo di informazioni leggibili, viene rigenerato ogni 6 secondi. Questo codice è il risultato della criptazione (operata dalla libreria "sjcl"[6]) di una stringa contenente l'ID dell'utente e un timestamp (inclusi secondi e millisecondi) riferito al momento di generazione del codice. La funzione di criptazione e generazione del codice viene invocata ogni 6 secondi, grazie all'utilizzo combinato di un hook useEffect e di un timer (generato tramite la funzione JavaScript setTimeout()).

```
useEffect(() => {
  if (!qrValue) {
    generateNewQRCodeValue();
  }
  const intervalId = setInterval(() => {
    generateNewQRCodeValue(); //encrypt of QR value
  }, DELAY);
  return () => { //clean-up function
    clearInterval(intervalId);
  };
}, []);
```

Figura 7.14 - UseEffect che genera la stringa criptata

[6] Stanford Javascript Crypto Library - https://www.npmjs.com/package/sjcl

La seconda scheda, "Meet a Colleague", permette all'utente di scannerizzare il codice QR criptato di un collega, avviando il processo di "cattura".

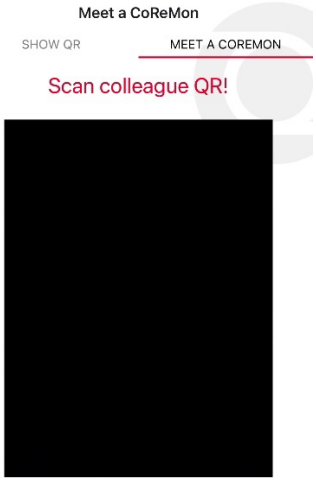

Figura 7.15 - Sezione Meet a Coremon

Per poter catturare un collega, l'utente deve possedere almeno una CoReBall. Una volta rilevato il codice QR, si avvia un'animazione che rappresenta visivamente il processo di cattura, il quale può concludersi in tre modi diversi: con il "Successo" (se la cattura del nuovo collega avviene senza problemi), con un "Errore" (in caso di problemi nel processo di cattura, con una descrizione dell'errore presentato all'utente), oppure mostrando il messaggio "Collega già catturato" (se l'utente tenta di catturare un collega già acquisito in precedenza). Nei casi di errore o di tentativo di cattura di un collega già acquisito, l'utente non consumerà nessuna CoReBall.

7.4 Concept

La sezione "Concept", raggiungibile anch'essa dalla barra di navigazione principale dell'app, funge da bacheca aziendale. In essa è presente un link denominato "Concept Reply". Questo link si presenta come un box rettangolare

cliccabile che indirizza direttamente al sito web di Concept. Il sito in questione è sviluppato mediante tecnologie web come JavaScript, HTML e CSS.

Figura 7.16 - Sezione sito Concept Reply

Per integrare e visualizzare efficacemente il sito all'interno dell'app, è stato utilizzato il componente <WebView>, precedentemente discusso nel Capitolo 4.

È interessante notare che "Concept" è l'unica sezione dell'app dove viene impiegata questa soluzione ibrida per la presentazione delle informazioni. Questa scelta è stata dettata dalla natura del sito web di Concept, il quale è privo di elementi graficamente pesanti o complessi da renderizzare. Optare per l'utilizzo di una WebView, invece di sviluppare una sezione specifica in React Native all'interno dell'app per visualizzare i contenuti relativi a Concept, si è rivelata una strategia efficace.

```
export default function CompanyWebViewScreen() {
  const webviewRef = useRef(null);

  return (
    <View style={styles.container}>
      <WebView
        ref={webviewRef}
        source={{ uri }}
        style={styles.webview}
        onShouldStartLoadWithRequest = {(event) => {
          if (event.url.startsWith(ASSETS_BASE_URL)) {
            return true;
          }
          Linking.openURL(event.url);
          return false;
        }}
      />
    </View>
  );
}
```

Figura 7.17 - Implementazione della WebView

Un ulteriore vantaggio di questa decisione è la praticità negli aggiornamenti: qualora il sito web di Concept subisca modifiche o aggiornamenti, non sarà necessario implementare aggiornamenti correlati all'interno dell'app mobile CoReDex.

La sezione "Concept" dell'app, oltre al collegamento diretto al sito web di Concept Reply, si arricchisce di un'area social dedicata agli eventi aziendali. Durante eventi speciali offline, si attiva una funzione "Events", dove gli utenti possono caricare foto scattate in tempo reale all'evento, esplorare e apprezzare quelle degli altri mettendo un "cuore", e consultare la classifica delle 10 foto più popolari basata sui cuori accumulati. Questa funzione crea un efficace ponte transmediale tra l'evento reale e l'app, potenziando l'engagement e l'intrattenimento tra i partecipanti.

7.5 CoReGym

La sezione in esame, insieme a “Meet” rappresenta il fulcro per l’accumulo di punti esperienza e per l’ottenimento di medaglie da parte dell’utente.

Settimanalmente, in questo spazio vengono lanciate delle challenges, ciascuna delle quali si concretizza in un quiz tematico. Ogni settimana è contrassegnata come uno "sprint", e i quiz proposti possono rifarsi a particolari giornate celebrative, nazionali o internazionali, o eventi storici che cadono nel range temporale della settimana attuale.

All'interno di questa sezione, l’utente si imbatterà in diversi tab posti in alto, specificatamente: “Current Sprint”, “Sprints History” e “Propose Challenge”.

La scheda “Current Sprint” mostra l’elenco delle sfide settimanali in corso. Tipicamente, ogni settimana viene proposta almeno una challenge.

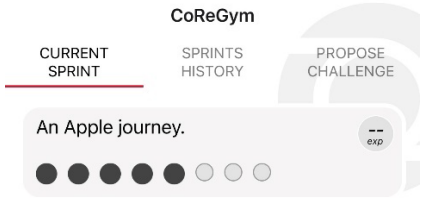

Figura 7.18 - Sprint attivo nella settimana corrente

Facendo clic sul box di una challenge, gli utenti hanno l'opportunità di svolgere il quiz relativo, accumulando punti esperienza in base al numero di risposte corrette fornite e al completamento del quiz stesso.

Per una visione retrospettiva, la scheda “Sprints History” offre una panoramica degli sprint passati. Ogni sprint appare come un box contenente il nome dello

sprint, i punti esperienza ottenuti dall'utente attraverso la partecipazione alle challenges, il numero di challenges completate, e le date di inizio e fine dello sprint.

Figura 7.19 - Elenco degli Sprint passati

Premendo su uno degli sprint elencati, gli utenti possono accedere a dettagli più specifici e visualizzare l'elenco delle challenges proposte durante quello sprint. Ogni challenge è visualizzata dentro un riquadro che mostra il titolo della challenge, il numero di domande formulate (rappresentato da pallini, dove ogni pallino contrassegna una domanda), i punti XP guadagnati e un'icona di stato che segnala il completamento della challenge.

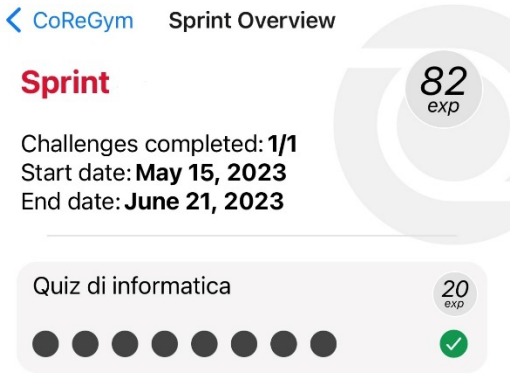

Figura 7.20 - Overview di uno Sprint passato

Toccando una challenge, gli utenti possono accedere a un riepilogo delle domande con le risposte fornite, presentato sotto forma di carosello.

L'ultimo tab, "Propose Challenge", presenta un messaggio di call-to-action rivolto agli utenti, invitandoli a inviare una mail agli sviluppatori per collaborare allo sviluppo di nuove sezioni e funzionalità, o per proporre idee per quiz e challenges, o suggerire miglioramenti generali. Il messaggio è seguito dall'elenco delle e-mail di contatto degli sviluppatori.

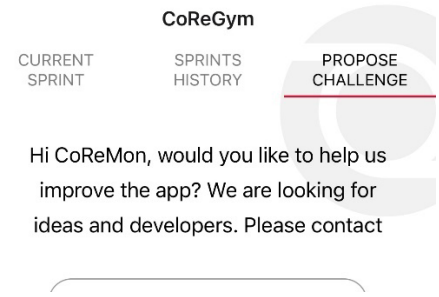

Figura 7.21 - Sezione "Propose Challenge"

Il design di ogni sprint, insieme alle correlate challenge, saranno oggetto di un'analisi approfondita nel capitolo seguente. Questo permetterà di esplorare in maniera più precisa e dettagliata le dinamiche, le regole e le logiche che stanno alla base della creazione e della gestione degli sprint e delle sfide settimanali proposte agli utenti nell'ambito dell'app.

8 CoReGym: Soft Skills Mapping

Uno degli obiettivi di CoReDex, che evidenzia il valore significativo e le potenzialità insite nell'applicazione in un contesto aziendale, è la creazione di una mappatura delle competenze nascoste di ciascun dipendente di Concept Reply. Tale mappatura è focalizzata in particolare sul rilevamento e l'identificazione delle cosiddette soft skills, competenze trasversali fondamentali nel mondo del lavoro.

CoReDex non rende obbligatorio, per gli utenti registrati, partecipare alle attività ludiche quali quiz e challenges. Pertanto, è complesso al momento attuale ottenere un quadro completo e dettagliato delle competenze di ciascun dipendente, dal momento che la partecipazione alle attività proposte è facoltativa e non uniformemente distribuita tra gli utenti.

Tuttavia, il sistema implementato in-app consente di raccogliere efficacemente dati preziosi in merito alle soft skills degli utenti partecipanti. Questo processo di raccolta e analisi dati si concretizza principalmente attraverso il meccanismo delle challenge, che vengono proposte agli utenti in ogni Sprint.

Attraverso l'analisi dei risultati di ogni challenge proposta agli utenti, CoReDex è in grado di tracciare un profilo delle competenze trasversali di ciascun partecipante, contribuendo a delineare un quadro sempre più preciso e affidabile delle risorse umane dell'azienda.

8.1 Creazione delle Challenges

Le challenge proposte su CoReDex sono programmate con cadenza settimanale. Nella versione attuale dell'app, le challenge disponibili si articolano esclusivamente in quiz a risposta multipla. Ogni fine settimana, il sistema propone uno o più quiz

incentrati su specifici argomenti. Gli utenti dispongono di una settimana per completare i quiz proposti; in caso contrario, se non riusciranno a completare il quiz entro il termine prestabilito, non accumuleranno punti esperienza (XP) e perderanno l'opportunità di completare quello specifico quiz.

La selezione dei temi per ciascun quiz è influenzata da diversi fattori. Si tende a privilegiare temi di rilevanza sociale (ad esempio, "Giornata dell'Uguaglianza di Genere"), argomenti di cultura generale (come "Il Lancio di Apollo 11") o temi legati alla cultura nerd e informatica (quali la nascita del "World Wide Web"). Ogni settimana, in funzione del calendario, viene effettuata una ricerca di eventi e celebrazioni internazionali legati a queste categorie, verificatesi negli anni precedenti.

I quiz vengono strutturati in formato JSON mediante l'utilizzo di uno script Python. Lo script, prendendo in ingresso dei file di testo con le domande, le possibili risposte, il tipo di domanda e la risposta corretta (se applicabile), genera un file JSON. Questo file rispecchia fedelmente la struttura dati accettata dal database per il salvataggio. È previsto che in futuro la procedura di creazione dei quiz sarà ulteriormente semplificata attraverso l'implementazione di un editor interno alla piattaforma web di amministrazione. Questo strumento permetterà di formattare i quiz in modo intuitivo, riducendo il carico di lavoro per gli amministratori.

8.1.1 Struttura dei quiz

Ogni quiz è strutturato come segue: contiene una descrizione e una call to action per l'utente, specifica una data di inizio e di fine, e include otto domande. Di queste, cinque sono domande standard, mentre le rimanenti tre sono domande di

“networking” finalizzate alla mappatura delle competenze. Ogni challenge settimanale è racchiusa all’interno di uno sprint.

L'app, sfruttando le logiche dei context e registrando le risposte degli utenti in tempo reale sul server, consente di avviare un quiz e completarlo successivamente. Gli utenti possono quindi interrompere un quiz e riprenderlo in seguito, ripartendo dal punto in cui si erano fermati, senza però avere la possibilità di ricominciarlo da capo o di modificare le risposte già fornite.

Il design dei quiz è stato intenzionalmente sviluppato senza voler apporre un limite di tempo per il completamento di ogni quiz per non costringere gli utenti a completarli in una sessione unica. Questa scelta, sebbene offra flessibilità, presenta anche degli svantaggi, come la possibilità che un utente possa cercare le risposte online per massimizzare i punti esperienza ottenuti. Abbiamo scelto di accettare questo rischio, privilegiando la flessibilità e la comodità per l'utente, permettendogli di completare le challenge nel momento e nel modo che ritiene più opportuno, con la possibilità di interrompere e riprendere in caso di necessità.

8.1.2 Domande “Standard”

Ogni domanda “standard” è formulata come una domanda a scelta multipla, presentando 4 o 5 possibili risposte, tra cui solo una è corretta. Ciascuna domanda, a seconda del tema generale del quiz, può essere pertinente o meno al settore in cui opera Concept. Ad esempio, in un quiz dedicato a "PokemonGO", è stata inclusa una domanda sul linguaggio di programmazione utilizzato per sviluppare il gioco. Nel quiz introduttivo su Concept Reply, era presente una domanda riguardante l'acronimo di un protocollo di comunicazione in ambito IoT. Altre domande sono più generiche e indagano la conoscenza dell’utente sul tema specifico del quiz. Il

tono adottato nei quiz è informale e stimolante, progettato per incitare gli utenti a dare il massimo.

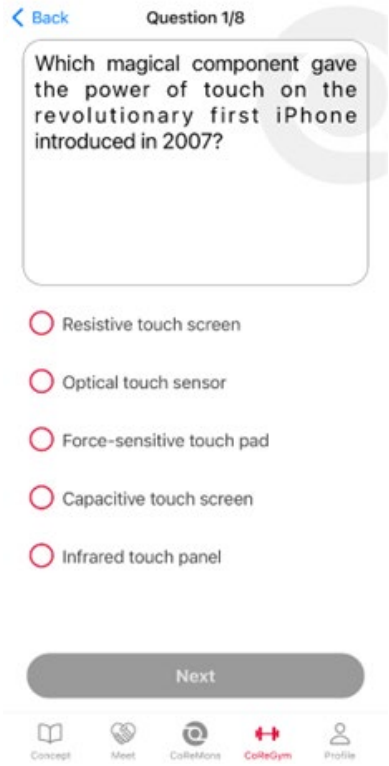

Figura 8.1 - Esempio di domanda standard

L'obiettivo nella formulazione delle domande dei quiz è offrire un'esperienza coinvolgente, equilibrando la difficoltà delle domande per mantenere gli utenti in uno stato di "flow". Questo termine descrive un senso di immersività e coinvolgimento durante un'attività ludica, senza cadere nella monotonia o frustrazione. Pertanto, le domande sono progettate per variare in difficoltà, alternando domande semplici a quelle più sfidanti, in modo da stimolare l'attenzione e la motivazione degli utenti, guidandoli attraverso un percorso di apprendimento e scoperta progressivo.

8.1.3 Domande di “Networking”

Le domande categorizzate come "networking" costituiscono la seconda varietà di quesiti presenti nei quiz. Si articolano presentando una descrizione che è strettamente collegata al tema centrale del quiz. In questa descrizione, viene proposta una situazione problematica o un quesito specifico, il cui obiettivo è di far emergere una competenza specifica che si ritiene essere fondamentale in un collega all'interno dell'ambiente aziendale. L'utente è quindi chiamato a selezionare un collega dall'elenco di iscritti su CoReDex, attribuendogli un livello di "confidenza" relativa alla skill in questione.

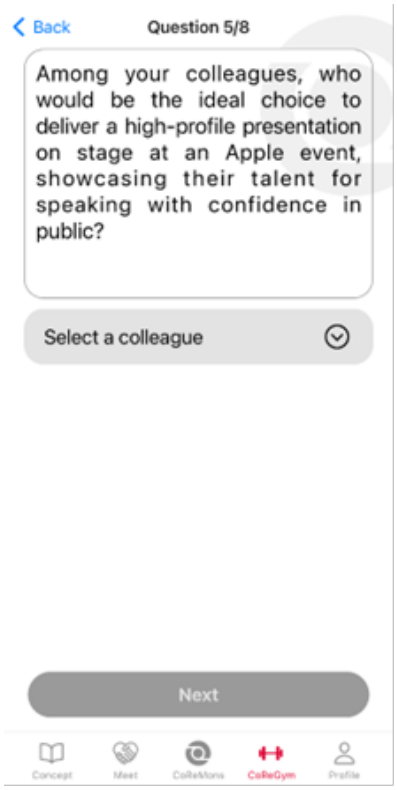

Figura 8.2 - Esempio di domanda di networking

La selezione della confidenza ha uno scopo preciso. Gli utenti hanno la possibilità di identificare un collega (o se stessi) come totalmente competente nella

skill evidenziata dalla domanda (focalizzata principalmente su soft skills, ma anche su hard skills in alcuni casi) o come parzialmente competente, assegnando un punteggio che varia da 1 a 5 stelle. Questa meccanica offre anche agli utenti neoassunti o a coloro che non conoscono molti colleghi la chance di selezionare un individuo di cui si fidano, anche se non sono completamente certi della sua competenza nella skill indicata. In questo scenario, l'utente può assegnare un livello di confidenza non massimo, permettendo una valutazione comunque informata e ponderata.

Assegnare una soft skill a un collega è un'azione che va oltre la semplice progressione nel quiz, offrendo un momento di riflessione sulle abilità dei propri colleghi. Tale riconoscimento contribuisce a valorizzare le qualità individuali, promuovendo un clima lavorativo positivo e costruttivo.

8.2 Quiz esempio: "It's-a me, Mario!"

Il quiz realizzato durante lo sprint #13 di CoReDex, che andava dal 10 al 17 settembre 2023, ha riguardato il videogioco "Super Mario Bros".

La descrizione del quiz è la seguente: "Il 13 settembre 1985, Nintendo presentò al mondo il suo iconico idraulico con il primo lancio di Super Mario Bros in Giappone! Quanto sei esperto riguardo l'universo Nintendo e le avventure di Mario?".

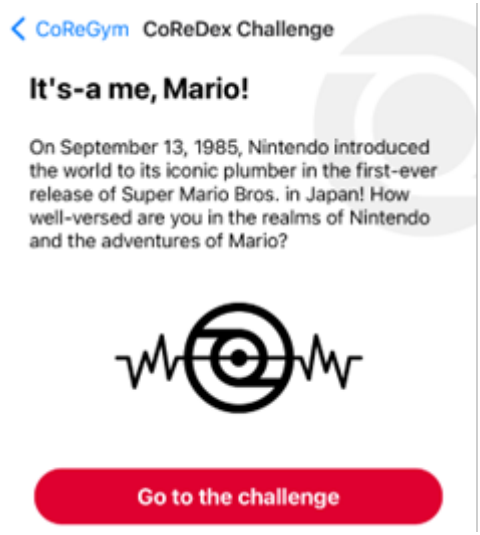

Figura 8.3 - Descrizione del quiz

Come si evince dalla descrizione, il quiz riguarda uno specifico evento accaduto nella cultura nerd negli anni passati durante la settimana dello sprint in questione.

Analizziamo le domande proposte: il quiz si apre con una domanda di tipo "standard" in cui viene chiesto a quale categoria di gioco appartiene il primo titolo sull'idraulico Mario, "Super Mario Bros.".

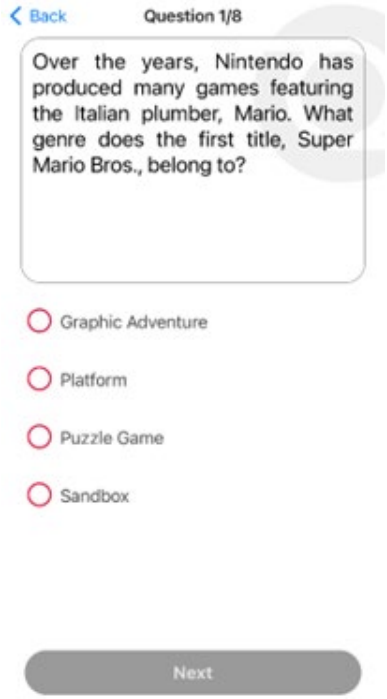

Figura 8.4 - Prima domanda standard

Seguono altre domande di tipologia standard, che si alternano a tre domande di "networking": due mirano a mappare delle soft skills, una riguarda una hard skill.

La prima domanda di networking proposta chiede all'utente di indicare quale collega è il più adatto, secondo lui, ad organizzare un torneo di videogiochi in ufficio dopo lavoro: la soft skill in questione è "organizzazione".

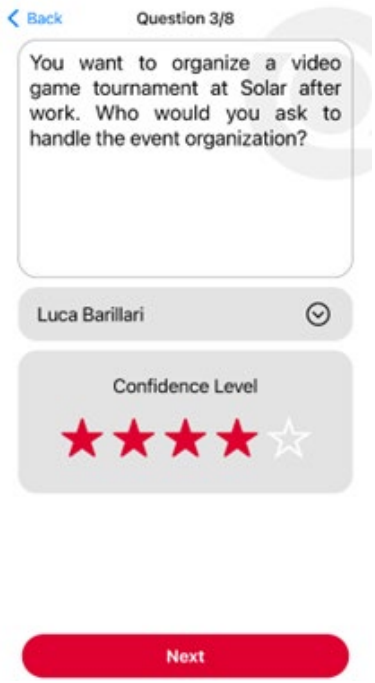

Figura 8.5 - Prima domanda di networking

La seconda domanda mette in luce la soft skill "creatività": viene chiesto all'utente di indicare il collega che sarebbe maggiormente in grado di trovare l'idea migliore per un nuovo videogioco.

Figura 8.6 - Seconda domanda di networking

Infine, l'ultima domanda di networking mappa una hard skill: "programmazione". Viene infatti chiesto di indicare quale collega possiede maggiori skill di programmazione per poter essere proclamato "master programmer" ed entrare a far parte della squadra di sviluppo di un nuovo videogioco.

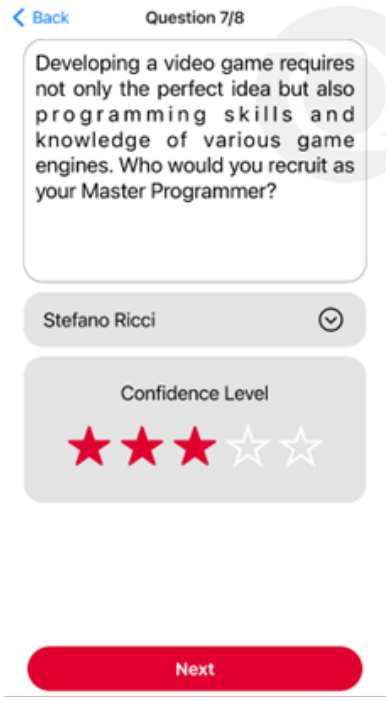

Figura 8.7 - Terza domanda di networking

Le altre domande "standard", con livelli di difficoltà diversi, hanno a che fare con la cultura nerd dietro al gioco di Super Mario Bros: una è ad esempio sul linguaggio di programmazione usato per programmare il gioco, un'altra è sui precedenti titoli Nintendo, un'altra ancora sul nome della console in cui è stato originariamente rilasciato il gioco, l'ultima riguarda un aspetto specifico di uno dei mondi del videogioco.

Al termine del quiz, l'utente può visualizzare il punteggio ottenuto e il riepilogo delle risposte date. Può accedere a questa sezione ogni volta che preme sul titolo del quiz dalla schermata CoReGym dopo averlo completato.

Figura 8.8 - Risultati del quiz

In conclusione, tramite questi quiz si mira a stimolare la curiosità e la conoscenza degli utenti sulle tematiche proposte (nel caso del quiz in esempio, sul videogioco di Super Mario e sul mondo Nintendo), spingendoli a creare una mappa delle soft skills e hard skills dell'intera azienda grazie alla contestualizzazione nel tema scelto di domande specifiche di networking.

Questo approccio può essere esteso oltre la mappatura delle competenze, applicandolo a qualsiasi obiettivo che richieda la creazione di una rete di relazioni e conoscenze tra i dipendenti aziendali.

9 Analisi dei risultati delle challenges

In questo capitolo, esploreremo in modo approfondito i risultati ottenuti attraverso l'utilizzo di CoReDex, focalizzandoci in particolare sulla mappatura delle competenze dei dipendenti di Concept Reply.

CoReDex, nata con l'obiettivo di rafforzare le connessioni interpersonali e la consapevolezza delle competenze all'interno dell'azienda, si è avvalsa come già specificato nei capitoli precedenti di dinamiche di gamification per stimolare l'interazione tra i colleghi e la riflessione sulle abilità individuali e di gruppo.

Il periodo considerato per questa analisi copre i primi tre mesi di utilizzo dell'applicazione, un arco temporale significativo per osservare le prime tendenze e reazioni degli utenti. Durante questo periodo, i dipendenti hanno partecipato attivamente alle sfide proposte dall'app, fornendo un insieme di dati prezioso e variegato sulle loro percezioni reciproche in termini di soft skills.

I dati sono stati raccolti attraverso le risposte fornite dagli utenti alle domande di networking presenti nei quiz settimanali. Queste domande, incentrate su specifiche competenze, hanno permesso agli utenti di identificare e attribuire particolari abilità a sé stessi o ai loro colleghi, accompagnando ogni scelta con un livello di "confidence". L'analisi dei dati raccolti mira a delineare una mappa delle competenze percepite all'interno dell'organizzazione, evidenziando le abilità più riconosciute e valorizzate dai colleghi.

In questo capitolo, verranno presentati e discussi i risultati di tale analisi, fornendo un quadro dettagliato della distribuzione e della percezione delle competenze soft all'interno dell'azienda. Attraverso questa esplorazione, si potranno trarre conclusioni importanti sulle dinamiche interne dell'azienda e sul potenziale impatto di CoReDex nell'analisi delle risorse umane.

9.1 Raccolta e Analisi dei dati: Data Mining

Il data mining è il processo di esplorazione e analisi di grandi insiemi di dati per scoprire modelli significativi, correlazioni e insight. Utilizzando metodi statistici, algoritmi e tecniche analitiche, il data mining trasforma grandi quantità di dati grezzi in informazioni utili per la presa di decisioni, la strategia aziendale e la comprensione dei comportamenti e delle tendenze (AWS, Che cos'è il data mining?, s.d.).

CoReDex raccoglie dati riguardanti le interazioni degli utenti, le loro risposte ai quiz e le valutazioni reciproche sulle competenze. Attraverso il data mining, è possibile:

1. **Identificare tendenze**: analizzare come le competenze vengono percepite e valorizzate all'interno dell'organizzazione.
2. **Rilevare modelli di interazione**: capire come i dipendenti interagiscono tra loro e quali sono le dinamiche di gruppo emergenti.
3. **Valutare l'impatto della gamification**: misurare l'efficacia delle meccaniche di gioco nel promuovere l'engagement e nel mappare competenze.
4. **Evidenziare competenze chiave**: identificare le competenze soft e hard più riconosciute e apprezzate, contribuendo a una migliore gestione delle risorse umane.

Il processo di raccolta dati per l'analisi delle competenze in CoReDex si è avvalso di uno script Python, progettato per elaborare due file JSON. Il primo file include l'elenco completo degli utenti con dettagli come id, e-mail, nome, cognome e status. Il secondo, invece, contiene i quiz archiviati nel sistema, dai quali sono state estratte

esclusivamente le domande focalizzate sul networking. Durante la creazione dei quiz, ogni domanda di networking viene associata a un array di "tags", che indicano le soft skills correlate, agevolando così una corrispondenza diretta e organizzata tra le domande poste e le competenze associate.

Mediante questi dati, lo script interagisce con il database attraverso un'API specifica, al fine di raccogliere le risposte fornite dagli utenti a tutte le domande di networking. Il risultato è un file CSV, strutturato con colonne come “GameId”, “UserId”, “QuestionId”, “Skill”, “ColleagueId” e “Confidence”. In ogni riga, il documento specifica quale utente (UserId) ha selezionato quale collega (ColleagueId) come rappresentante di una determinata competenza (Skill) in una domanda di networking (QuestionId) di un particolare quiz (GameId), nonché il livello di fiducia (Confidence) assegnato.

Per organizzare ulteriormente i dati raccolti nel file CSV, è stato impiegato un secondo script Python. Questo passaggio ha l'obiettivo di estrarre l'elenco delle competenze e, per ciascuna di esse, creare una cartella contenente due file CSV, “nodes” e “edges”. Il file “Nodes” racchiude l'elenco degli utenti coinvolti nelle domande di networking per ogni competenza (sia come rispondenti sia come destinatari delle risposte), mentre il file “Edges” delinea le connessioni tra gli utenti e le loro scelte, mostrando la frequenza di selezione di ciascun collega e il relativo punteggio di fiducia assegnato.

Questi file sono stati utilizzati per generare grafici relazionali attraverso il servizio web Graph Commons [7], offrendo una rappresentazione visiva e intuitiva dell'analisi delle competenze nascoste di ogni utente di CoReDex.

[7] Graph Commons - https://graphcommons.com/

9.2 Recap delle soft skills evidenziate

In CoReDex, le domande di networking nei quiz sono progettate per mettere in luce una vasta gamma di soft skills. Di seguito, è indicato un recap delle principali soft skills che sono state evidenziate attraverso i quiz di CoReDex:

1. **Artistic Sense**: la capacità di apprezzare, valutare e creare bellezza artistica, importante per i ruoli che richiedono creatività e sensibilità estetica.
2. **Public Speaking**: l'abilità di parlare efficacemente in pubblico, fondamentale per chi assume ruoli di leadership o deve presentare idee e progetti.
3. **Collaboration**: la predisposizione a condividere con uno o più colleghi idee e risorse per il raggiungimento di un obiettivo comune.
4. **Support**: la capacità di fornire assistenza e supporto ai colleghi, importante per creare un ambiente lavorativo solidale e produttivo.
5. **Critical Thinking**: l'abilità di analizzare fatti e situazioni per formare un giudizio, essenziale per la risoluzione di problemi e la presa di decisioni.
6. **Organization**: la competenza nell'organizzare risorse e il proprio lavoro, indispensabile per il successo di progetti e iniziative.
7. **Creativity**: la capacità di generare idee nuove e originali, fondamentale in settori innovativi come quello tecnologico.
8. **Trust**: la qualità di essere affidabili e onesti, che permette di costruire fiducia tra colleghi e clienti.
9. **Sociable Approach**: la facilità nel relazionarsi con gli altri, fondamentale per ruoli che richiedono interazione con clienti o collaboratori.

10. **Photography**: la capacità di catturare e creare immagini di impatto, utile in settori come il marketing e la comunicazione.
11. **Charisma**: la qualità di essere persuasivi e capaci di ispirare gli altri, importante per i leader e figure di riferimento.
12. **Problem Solving**: l'abilità di identificare problemi e trovare soluzioni efficaci, essenziale in quasi ogni ruolo lavorativo.
13. **Project Management**: la competenza nel gestire progetti, assicurando che gli obiettivi siano raggiunti in tempo e nel rispetto del budget.
14. **Empathy**: la capacità di comprendere e condividere i sentimenti degli altri, cruciale per costruire relazioni di lavoro positive.
15. **Time Handling**: l'abilità di gestire il proprio tempo in modo efficace, massimizzando la produttività.
16. **Strategic Thinking**: la capacità di pensare in modo strategico e pianificare a lungo termine, importante per ruoli di leadership e gestione.
17. **Social Awareness**: la sensibilità alle dinamiche sociali e alle esigenze degli altri, utile in contesti collaborativi.
18. **Autonomy**: l'abilità di lavorare in modo indipendente, fondamentale in ruoli che richiedono iniziativa e autogestione.
19. **Competitiveness**: la tendenza a perseguire il successo e superare gli altri, importante in contesti competitivi.
20. **Teamwork**: la capacità di lavorare efficacemente all'interno di un team, fondamentale in qualsiasi ambiente lavorativo.
21. **Curiosity**: l'interesse e la volontà di apprendere ed esplorare nuovi orizzonti, importante per l'innovazione e la crescita personale.

22. **Leadership**: la capacità di guidare e motivare un team, essenziale per chi ricopre ruoli direttivi.

9.3 Analisi delle competenze

Tra le soft skills proposte nei quiz di CoReDex sono state scelte alcune di esse per creare grafici chiari ed efficaci, che illustrano visivamente le scelte dei singoli utenti. Questo approccio offre una panoramica comprensibile della rappresentanza di ogni competenza, valutata "dal basso". Le competenze esaminate comprendono: fiducia (trust), lavoro di squadra (teamwork), capacità oratoria (public speaking), leadership, organizzazione, empatia, creatività e approccio sociale (sociable approach).

I grafici visualizzano una rete di nodi e archi, dove ogni nodo rappresenta un utente e ogni arco indica una relazione tra gli utenti. Gli archi sono rappresentati da frecce che possono collegare i nodi in entrata o in uscita. Lo spessore di ciascun arco varia a seconda del livello di fiducia (confidence) attribuito: le frecce più sottili indicano valori di fiducia più bassi.

La grandezza dei nodi, rappresentati come cerchi, è determinata sia dalla quantità di archi entranti, che riflette il numero di colleghi che hanno selezionato quell'utente-nodo per rappresentare la competenza specifica, sia dal peso complessivo degli archi, ovvero dal livello aggregato di fiducia conferito.

9.3.1 Trust

La fiducia è un elemento fondamentale nelle relazioni professionali e nei ruoli che richiedono discrezione e integrità. Indica inoltre una predisposizione per ruoli

che richiedono un alto livello di affidabilità e responsabilità. Un alto livello di fiducia nei colleghi corrisponde a un ambiente di lavoro sicuro e supportivo.

Questa soft skill è stata proposta in alcune domande, ad esempio: "*Ne Le follie dell'Imperatore, l'iconico duo composta da Kronk e Yzma rappresenta una perfetta combinazione di cervello e muscoli. Se fossi Yzma, quale collega sceglieresti come tuo Kronk per la sua affidabilità?* oppure "*Immagina (non sarà così difficile) che sia stata una giornata lunga e stressante al lavoro, ma che finalmente sia ora di uscire e bere qualcosa con i colleghi (al Coguaro o al tuo bar preferito). A chi chiederesti di ordinarti uno Spritz?*".

Il grafico seguente è stato realizzato analizzando i dati raccolti.

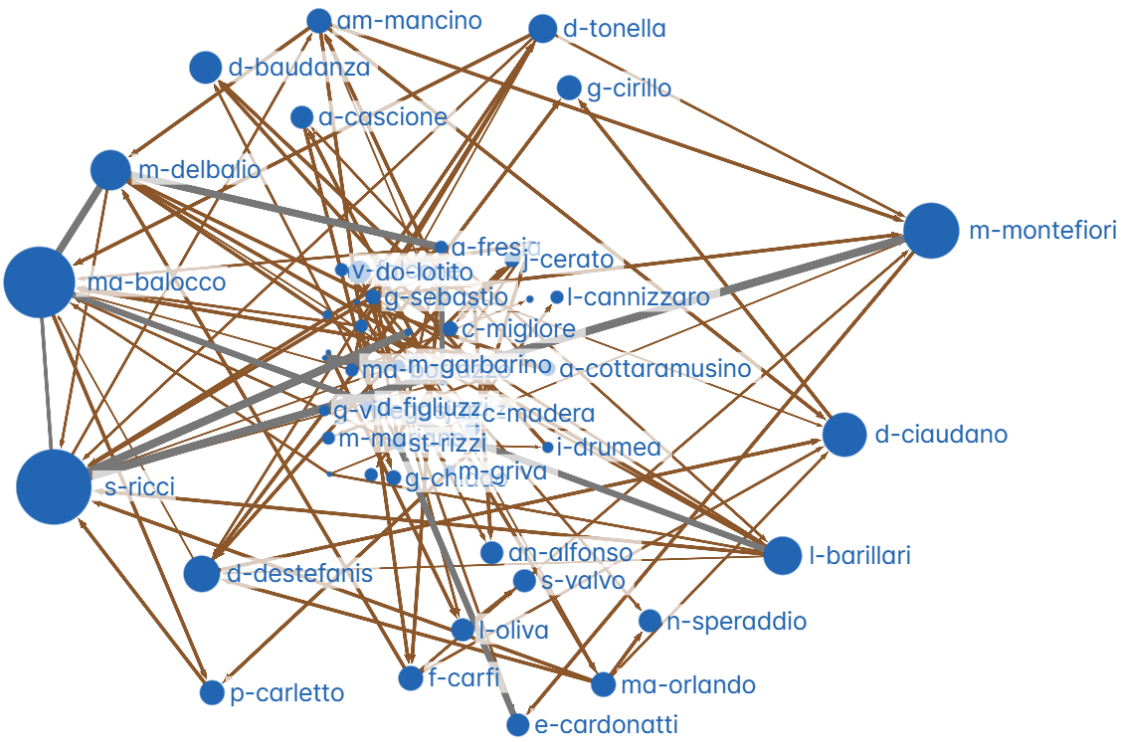

Figura 9.1 - Grafico Fiducia

Ciò che emerge dal grafico è che, tra i tanti utenti che hanno risposto alle domande relative a questa soft skill, "s-ricci" e "ma-balocco" risultano essere i due colleghi di cui ci si può fidare di più in azienda. Oltre a questi due nodi, ce ne sono

altri con una media importanza (“m-montefiori”, “m-delbalio”, “d-ciaudano”, “l-barillari”).

9.3.2 Teamwork

L'analisi rivela come il lavoro di squadra sia percepito e praticato tra i vari colleghi. Saper lavorare bene in gruppo rapportandosi con gli altri colleghi è una competenza essenziale in quasi tutti i contesti lavorativi.

Questa competenza è apparsa nella seguente domanda: “*Che brutta giornata! Hai fatto un casino a lavoro, hai accidentalmente eliminato una tabella nel database di production. Da solo non ce la farai mai, devi fare squadra con qualcuno che ti aiuti a venirne a capo il prima possibile! Chi può aiutarti a risolvere il tuo problema (o a comprare un passaporto falso per trasferirti per sempre in Messico)?*”

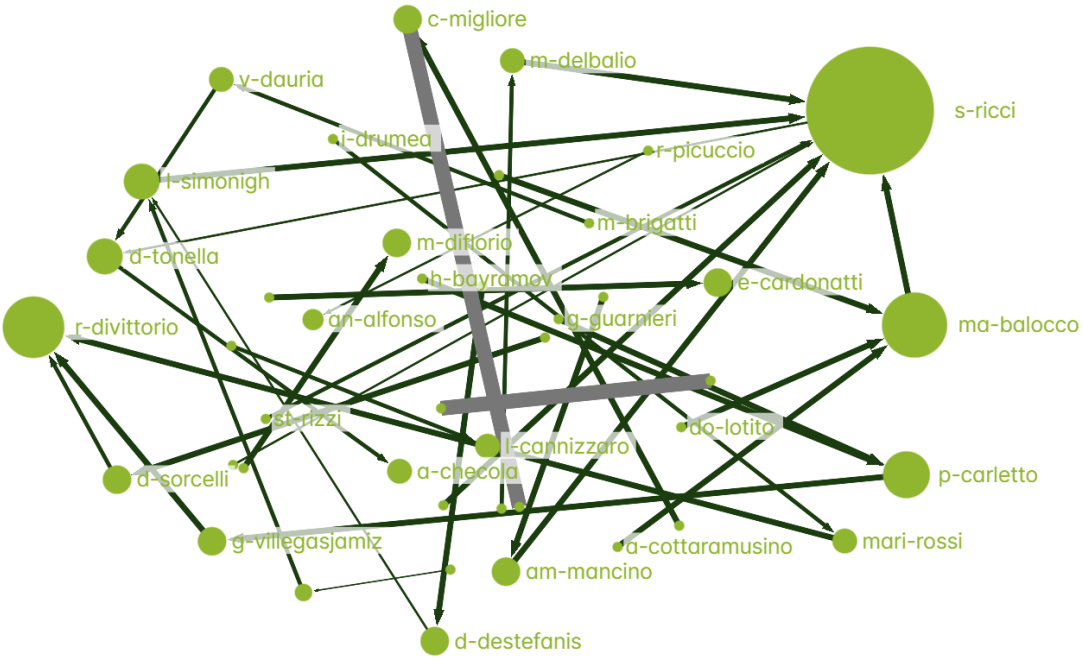

Figura 9.2 - Grafico Teamwork

l grafico indica “s-ricci” come il miglior collega con cui fare lavoro di squadra, ma in sua assenza anche “ma-balocco”, “r-divittorio” e “p-carletto” possono essere compagni vincenti.

9.3.3 Public Speaking

Questa abilità è fondamentale per ruoli che richiedono presentazioni frequenti, come il marketing, le vendite o le posizioni esecutive. L'analisi di come i dipendenti percepiscono i colleghi in questa abilità fornisce un’indicazione su chi può rappresentare efficacemente l'azienda in pubblico o su chi può motivare i team interni all’azienda.

Esempi di domande associate a questa skill sono: "*Immagina di vivere nel 1888 e di dover pubblicizzare la prima macchina fotografica con pellicola a rullino. Tra i tuoi colleghi, chi sarebbe la persona ideale per presentare il prodotto a quante più persone possibile con eccellenti capacità di parlare in pubblico?”* oppure "*Tra i tuoi colleghi, chi sarebbe la scelta ideale per tenere una presentazione di alto profilo sul palco di un evento Apple, mettendo in mostra il proprio talento nel parlare con sicurezza in pubblico?*".

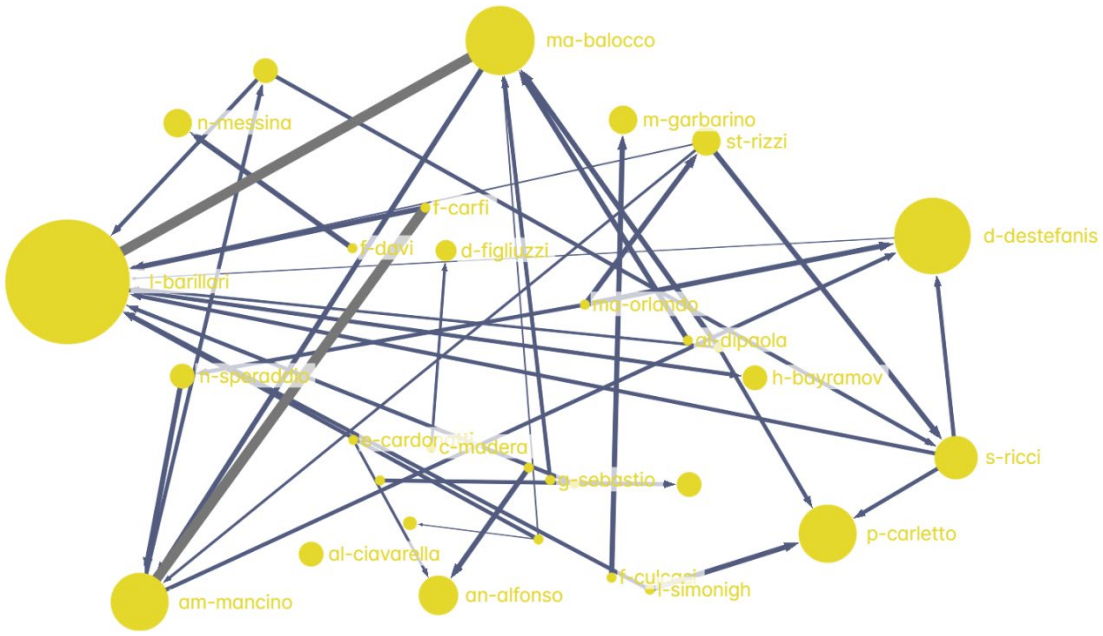

Figura 9.3 - Grafico Public Speaking

Dal grafico emerge “l-barillari” come il migliore a parlare in pubblico secondo i suoi colleghi, tra i quali spiccano anche “ma-balocco”, “d-destefanis”, “p-carletto” e “a-mancino”.

9.3.4 Leadership

Chi dimostra forti competenze in leadership è adatto a ruoli direttivi o manageriali. Queste persone sono capaci di guidare i team, ispirare il personale e prendere decisioni strategiche, nonché gestire progetti e iniziative aziendali di vasta portata. La mappatura di questa competenza mostra chi sono i potenziali leader informali all'interno dell'azienda, indipendentemente dalle loro posizioni ufficiali.

Sono state proposte le seguenti domande: "*Immagina di affrontare una sfida magica come Harry Potter. Tra i tuoi colleghi, chi incarna lo stesso spirito di leadership, guidando la squadra attraverso le sfide come se fossero avventure a Hogwarts?”* oppure "*Immagina*

di essere coinvolto in un progetto di esplorazione aerospaziale. Chi sceglieresti come collega con esperienza nella gestione di progetti e nel coordinamento del team?".

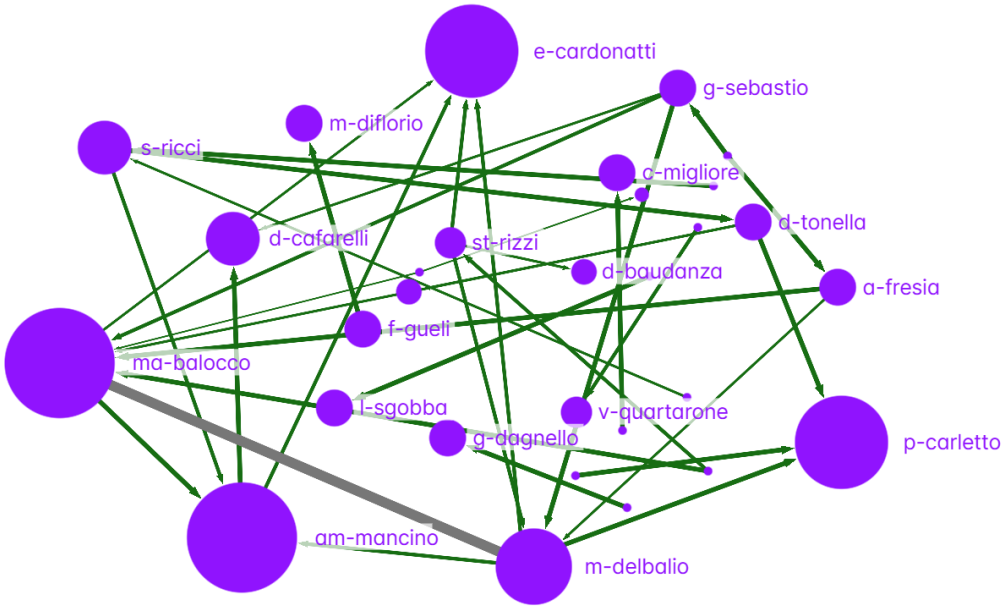

Figura 9.4 - Grafico Leadership

Dal grafico spiccano “ma-balocco”, “am-mancino”, “p-carletto” e “e-cardonatti”, ma anche “m-delbalio”, “s-ricci” e “d-cafarelli”.

9.3.5 Organization

La capacità di gestire in modo efficiente tempo e risorse è cruciale per rispettare le scadenze e gestire progetti complessi, garantendo al contempo un equilibrio tra lavoro e vita privata. L’analisi di questa competenza aiuta a identificare i colleghi che meglio gestiscono il proprio lavoro.

Tra le domande proposte per questa competenza, troviamo: "*Concept Reply ha organizzato una festa in spiaggia come attività di team building ma per sbaglio ha messo uno zero di troppo sull’ordine della birra. Ora i Replyer devono finire 700 litri di birra ghiacciata;*

quindi, si decide di dare un premio alla squadra che beve più birra. Quale dei tuoi colleghi pensi che possa aiutarti a non sprecare la birra che ti è stata ordinata?" oppure "*Devi organizzare per Concept un concorso fotografico dove gli scatti migliori verranno premiati. Chi chiameresti per aiutarti?*".

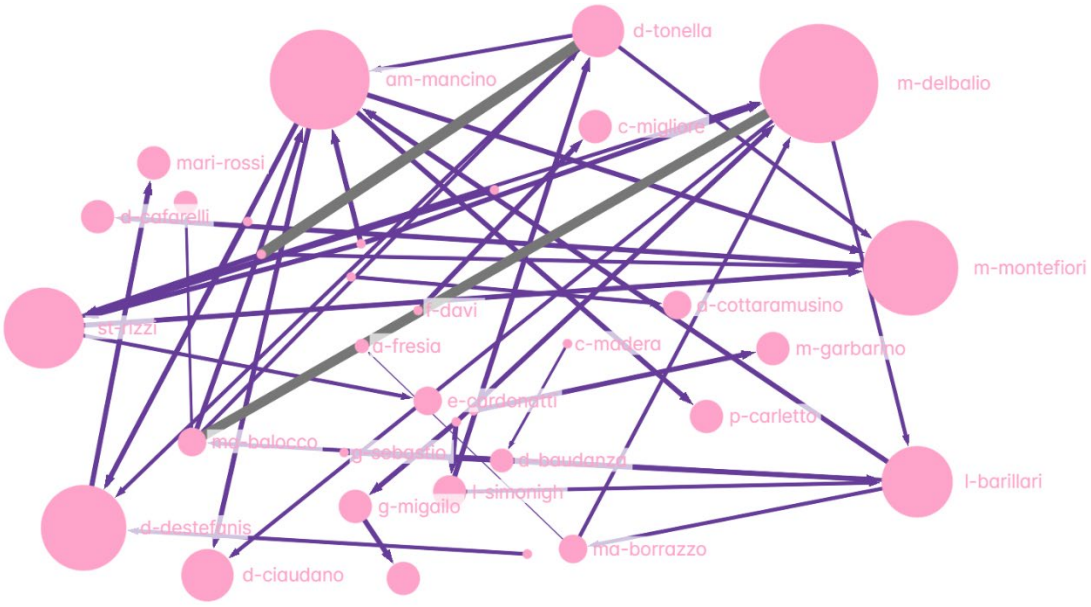

Figura 9.5 - Grafico Organization

I dipendenti di Concept ritengono che "m-delbalio" sia la persona più organizzata, seguita da "am-mancino", "m-montefiori", "d-destefanis", "st-rizzi" e "l-barillari".

9.3.6 Empathy

In un ambiente lavorativo, la comprensione e la sensibilità alle esigenze e ai sentimenti altrui non solo migliorano il clima lavorativo, ma anche la qualità del lavoro di squadra e la gestione dei clienti, contribuendo a creare un ambiente di lavoro inclusivo e solidale. L'analisi mostra come i dipendenti percepiscono la capacità dei loro colleghi di comprendere e condividere i sentimenti degli altri.

Domande tipiche su questa competenza sono: "*All'interno del tuo ambiente di lavoro, chi rappresenta lo spirito empatico e la saggezza di Silente, illuminando il percorso degli altri con gentilezza e intuizione?*" oppure *"Questa settimana celebriamo anche la Giornata Mondiale del Cane. Pensando ai tuoi colleghi, chi tra loro credi avrebbe apprezzato particolarmente un quiz su questo argomento, dimostrando non solo un amore per gli animali ma anche una notevole empatia nei confronti del benessere animale?"*.

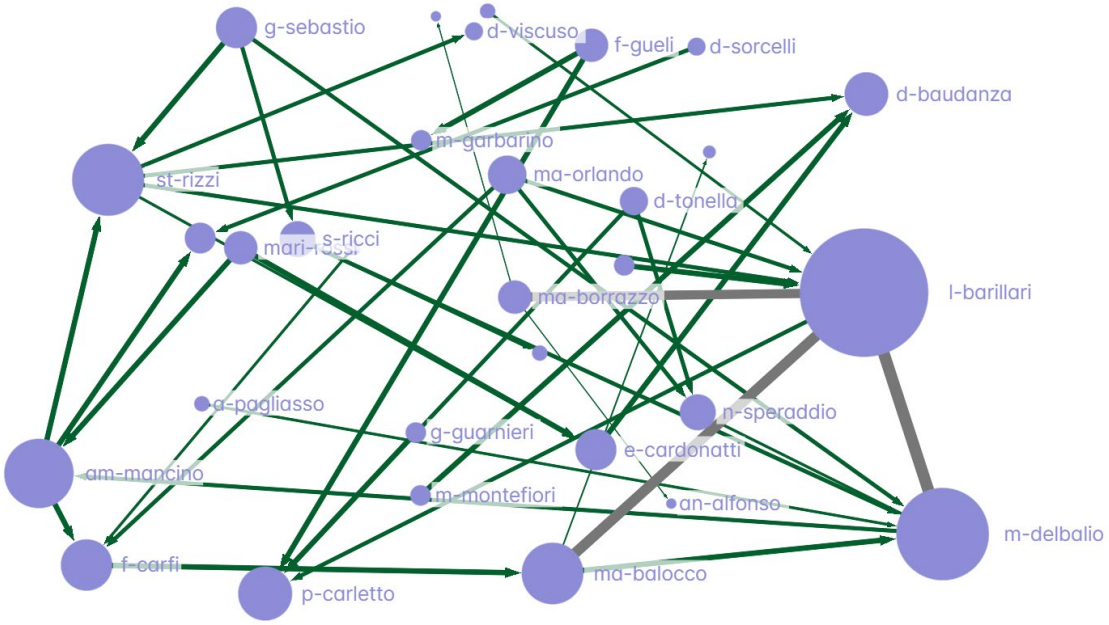

Figura 9.6 - Grafico Empathy

"l-barillari" risulta essere il collega più empatico di Concept, seguito da "m-delbalio", "st-rizzi", "am-mancino" e "ma-balocco".

9.3.7 Creativity

In un settore in rapida evoluzione come la tecnologia, la creatività e il senso artistico sono fondamentali. Queste competenze permettono di pensare fuori dagli schemi, innovare e creare soluzioni uniche a problemi complessi. L'analisi di questa

competenza evidenzia i dipendenti che sono visti come motori di idee e soluzioni innovative.

Esempi di domande sulla creatività sono: "*Stai lavorando a una missione di ricerca sulla luna e hai bisogno di un membro del team che sia creativo e capace di pensare in modo innovativo. Chi sceglieresti per questo ruolo?*" oppure "*Se le auto del futuro avessero tutte un bagagliaio delle dimensioni di una FIAT Panda, quale dei tuoi colleghi pensi avrebbe le competenze per progettare la bici pieghevole del futuro?*".

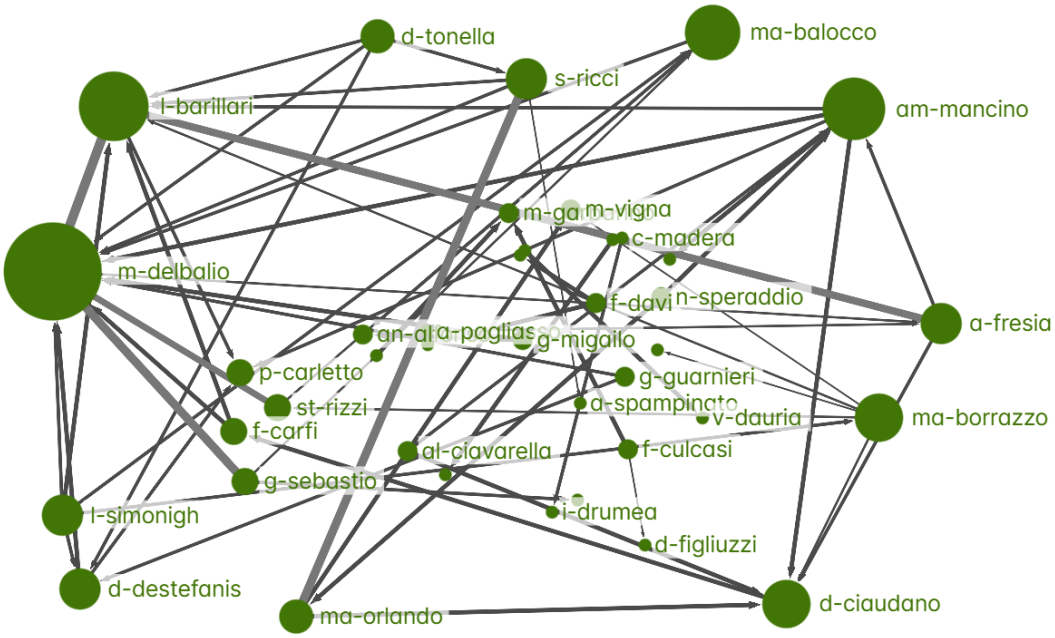

Figura 9.7 - Grafico Creativity

Dal grafico "m-delbalio" spicca come creativa del gruppo.

9.3.8 Sociable approach

Questa competenza riflette la capacità di stabilire relazioni positive con colleghi e clienti. L'analisi fornisce una panoramica di come i dipendenti si relazionano tra loro e con l'esterno.

Sono state proposte le seguenti domande: "*Se dovessi andare ad un concerto, chi sarebbe il partner perfetto?*" oppure "*Domande come questa non hanno una risposta giusta, ma esistono per vedere come le persone in azienda interagiscono tra loro... Quindi, se dovessi indovinare, chi è stato l'ultimo ad andare a letto dopo il primo giorno dell'evento aziendale estivo?*".

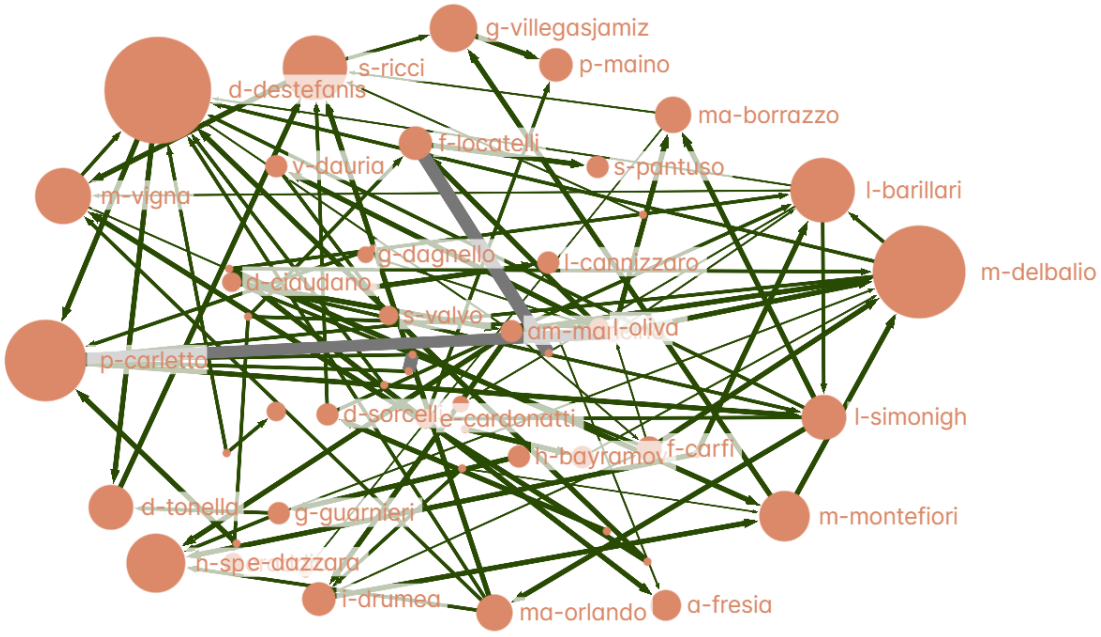

Figura 9.8 - Grafico Sociable Approach

Senza ombra di dubbio "d-destefanis" è la persona più socievole di Concept. Lo seguono "m-delbalio", "p-carletto", "s-ricci" e "l-barillari". Altri colleghi degni di nota sono "m-vigna", "n-speraddio" e "m-montefiori".

9.4 Panoramica riassuntiva delle soft skills analizzate

Dall'analisi dei risultati per ogni competenza emergono alcuni colleghi in particolare.

Il grafico a colonne impilate sottostante illustra i 20 colleghi che si sono distinti maggiormente nelle risposte degli utenti, evidenziando le specifiche abilità per le quali sono stati più frequentemente riconosciuti.

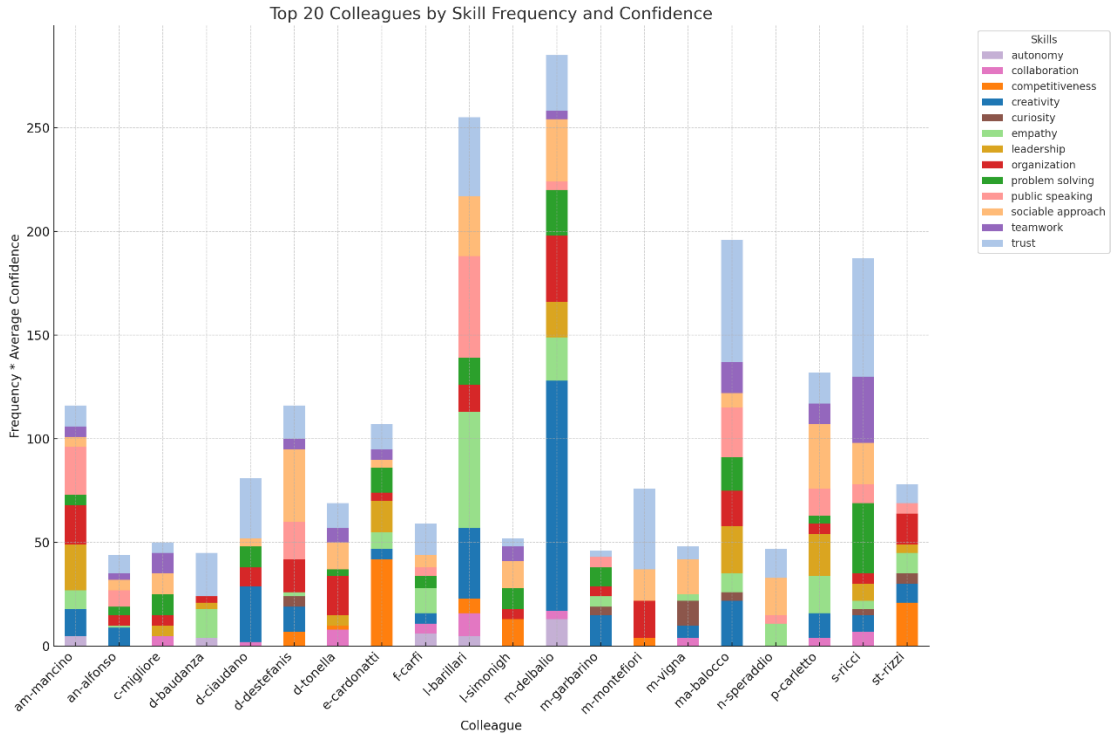

Figura 9.9 - I 20 colleghi più frequenti nelle risposte di networking

Ogni colonna nel grafico rappresenta un collega specifico, con i colleghi ordinati alfabeticamente sull'asse X per facilitare il riconoscimento e la comparazione.

Le colonne sono suddivise in segmenti colorati, dove ogni colore corrisponde a una competenza specifica, come "creativity", "trust", "leadership", e altre.

L'altezza di ogni segmento colorato all'interno di una colonna riflette un punteggio ponderato, derivato dalla frequenza con cui un determinato collega è stato valutato per una specifica competenza, moltiplicato per la media delle

valutazioni di "confidence" per quella competenza. Questo metodo di calcolo fornisce un indicatore combinato dell'importanza e della fiducia attribuita a ciascuna competenza per ogni individuo.

Dall'analisi di questo grafico emergono interessanti osservazioni. Ad esempio, alcune competenze come "creativity" e "empathy" presentano punteggi particolarmente elevati per certi individui, suggerendo un forte riconoscimento di queste abilità in specifici membri del team. Allo stesso tempo, la distribuzione variabile dei colori e delle altezze delle colonne sottolinea come le competenze siano diversamente valorizzate e riconosciute tra i diversi membri del gruppo, riflettendo un'eterogeneità di punti di forza e di focalizzazione delle competenze all'interno dell'azienda.

9.5 Considerazioni sui risultati della mappatura delle competenze

Nel valutare i risultati ottenuti attraverso le challenge di CoReDex, è fondamentale considerare alcuni aspetti chiave. Innanzitutto, è importante ribadire che la partecipazione ai quiz non è obbligatoria. Questo implica che i dati raccolti, pur essendo rappresentativi, non coprono l'intero spettro dei dipendenti dell'azienda. Di conseguenza, le informazioni ottenute riflettono le percezioni e le valutazioni di una porzione degli utenti piuttosto che dell'intero collettivo aziendale.

Nonostante l'aspetto volontario della partecipazione alle challenge, l'analisi dei dati raccolti ha rivelato tendenze significative riguardo alle predisposizioni dei colleghi verso specifiche competenze. È interessante osservare come i risultati statistici si allineino in larga misura con la percezione generale che i dipendenti hanno delle

abilità dei loro colleghi. Questa corrispondenza si è verificata anche per quegli utenti che hanno partecipato in misura minore alle challenge, suggerendo una certa affidabilità dei risultati ottenuti.

La varietà e la quantità di nodi identificati per ogni competenza sono state influenzate dal numero di domande specifiche per quella skill all'interno dei quiz. Alcune competenze potrebbero essere state più enfatizzate rispetto ad altre a causa di un maggior numero di domande a loro dedicate. Questo aspetto deve essere tenuto in considerazione nell'interpretare i risultati.

Un altro fattore da considerare è la fluttuazione della frequenza di risposta e partecipazione alle challenge. Ci sono state settimane in cui l'engagement degli utenti è stato più alto, influenzando così la quantità e la qualità dei dati raccolti. Queste variazioni possono essere attribuite a diversi fattori, come l'interesse generato dai temi dei quiz o la disponibilità temporale degli utenti in specifici periodi.

Nonostante queste limitazioni, è innegabile che l'approccio di gamification adottato abbia offerto una visione innovativa e coinvolgente sulle competenze nascoste dei dipendenti. I quiz hanno permesso di scoprire e valorizzare abilità che altrimenti potrebbero non essere state riconosciute in un contesto lavorativo tradizionale. Questo ha stimolato la curiosità e l'interazione tra i colleghi, contribuendo a creare un ambiente lavorativo più consapevole delle diverse abilità presenti all'interno dell'azienda.

In conclusione, pur con le sue limitazioni, questo metodo ha offerto una prospettiva unica e preziosa sulle competenze e sulle dinamiche interpersonali in azienda, rivelando l'efficacia della gamification come strumento di mappatura delle competenze. In futuro potrà essere ulteriormente esteso e migliorato, con l'obiettivo di aumentare ancora di più l'engagement e proporre sfide ancora più accattivanti.

10 Conclusioni

Quest'ultimo capitolo è dedicato alle riflessioni e prospettive future su CoReDex.

L'obiettivo principale dell'app, quello di rafforzare le relazioni tra i dipendenti e scoprire le soft skills nascoste, è stato ampiamente raggiunto. La partecipazione attiva dei colleghi nel "catturarsi" a vicenda e la loro eccitazione per le sfide settimanali sono la prova tangibile dell'entusiasmo e del coinvolgimento che l'app ha suscitato. La prima mappatura delle competenze soft ha fornito un quadro prezioso delle abilità interpersonali e professionali all'interno dell'azienda.

Tuttavia, il percorso non è stato privo di ostacoli. Nonostante l'ampia adesione, in alcuni momenti si è registrato un calo di engagement, sollevando la necessità di rivedere e potenziare alcune meccaniche di gioco. Inoltre, ci siamo imbattuti in alcuni bug tecnici che hanno richiesto interventi tempestivi per garantire un'esperienza utente fluida e piacevole.

Guardando al futuro, vedo un orizzonte ricco di possibilità e miglioramenti per CoReDex. Un'evoluzione naturale potrebbe essere l'introduzione di medaglie che riconoscano e valorizzino sia le competenze soft che hard identificate attraverso le challenge, aggiungendo un ulteriore strato di riconoscimento e apprezzamento delle abilità individuali. Potenziare il sistema di livellamento, rendendo accessibili contenuti e giochi esclusivi in base al livello raggiunto, potrebbe aumentare ulteriormente l'engagement e la motivazione.

Una funzionalità attualmente in fase di implementazione è il concetto dei "Pacchetti Settimanali di Colleghi". Questa novità prevede che ogni settimana gli utenti ricevano un pacchetto di "figurine digitali", ognuna rappresentante un collega scelto randomicamente dal database. Attraverso questo sistema, gli utenti potranno

accedere ad anteprime dei profili dei colleghi ancora da catturare in CoReDex, aumentando l'interattività e il coinvolgimento nell'app.

Immagino anche l'introduzione di nuove sfide, come giochi di memoria o attività che riflettano la cultura nerd e tecnologica di Concept Reply, per mantenere alto l'interesse e la partecipazione. Un altro sviluppo interessante potrebbe essere l'utilizzo di CoReDex come strumento di formazione, implementando quiz e attività educative che si intrecciano con gli obiettivi di apprendimento aziendali.

Infine, sviluppare un sistema di notifiche efficace potrebbe giocare un ruolo chiave nel mantenere gli utenti coinvolti e informati sulle nuove sfide, gli aggiornamenti dei colleghi e le opportunità di interazione all'interno dell'app.

In conclusione, CoReDex si è rivelato un progetto di successo e di grande valore, sia per l'azienda che per i suoi dipendenti. Le lezioni apprese, le difficoltà incontrate e le idee per il futuro delineano un percorso eccitante per l'evoluzione continua di questa applicazione, che si prefigge di rimanere un punto di riferimento nella gamification aziendale e nella mappatura delle competenze.

Bibliografia

(s.d.). Tratto da React Native: https://reactnative.dev/

Abt, C. C. (1987). *Serious Games.*

Apple Developer. (s.d.). *Apple Developer*. Tratto da Apple Developer: https://developer.apple.com/search/?q=api

AWS, A. (s.d.). *Amazon CloudFront*. Tratto da AWS: https://aws.amazon.com/it/cloudfront/

AWS, A. (s.d.). *Amazon S3*. Tratto da AWS: https://aws.amazon.com/it/s3/

AWS, A. (s.d.). *Amazon Simple Storage Service Documentation*. Tratto da AWS: https://docs.aws.amazon.com/s3/

AWS, A. (s.d.). *Caratteristiche di AWS Lambda*. Tratto da AWS: https://aws.amazon.com/it/lambda/features/

AWS, A. (s.d.). *Che cos'è il data mining?* Tratto da AWS: https://aws.amazon.com/it/what-is/data-mining/

AWS, A. (s.d.). *Cos'è un'API RESTful?* Tratto da AWS: https://aws.amazon.com/it/what-is/restful-api/

AWS, A. (s.d.). *Funzionalità di Amazon DynamoDB*. Tratto da AWS: https://aws.amazon.com/it/dynamodb/features/

AWS, A. (s.d.). *What is Amazon Cognito?* Tratto da AWS: https://docs.aws.amazon.com/cognito/latest/developerguide/what-is-amazon-cognito.html

AWS, A. (s.d.). *What is the AWS CDK?* Tratto da AWS: https://docs.aws.amazon.com/cdk/v2/guide/home.html

Caponetto, I., Earp, J., & Ott, M. (2014). *Gamification and Education: A Literature Review.*

Cooper, A. (2014). *About Face: The Essentials of Interaction Design.*

Danny, P. (2013). The effect of virtual achievements on student engagement. *ACM Digital Library*.

Deterding, S. (2012). *Gamification: Designing for Motivation.* Tratto da Social Mediator.

Farzan, R., DiMicco, J. M., Millen, D., & Dugan, C. (2008). *Results from deploying a participation incentive mechanism within the enterprise.* Retrieved from ACM Digital Library.

Hamari, J., Koivisto, J., & Sarsa, H. (2014). Does Gamification Work? – A Literature Review of Empirical Studies on Gamification. *Annual Hawaii International Conference on System Sciences (HICSS).*

Jenkins, H. (2007). *Transmedia Storytelling 101*. Tratto da HENRY JENKINS: https://henryjenkins.org/blog/2007/03/transmedia_storytelling_101.html

Kim, B. (2015). *Understanding Gamification.*

L. Rosenfeld, Morville, P., & Arango, J. (2015). *Information Architecture: For the Web and Beyond.*

LEDU Education Ecosystem. (s.d.). *Native vs. cross-platform app development: pros and cons*. Tratto da Medium: https://codeburst.io/native-vs-cross-platform-app-development-pros-and-cons-49f397bb38ac

Martin, R. C. (2017). *Clean Architecture: A Craftsman's Guide to Software Structure and Design.*

Martin, R. C. (s.d.). *The Clean Architecture*. Tratto da The Clean Code Blog: https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html

Mazzaglia, M. (2021). Gamification - slides corso di Ingegneria del Cinema.

Mollick, E., & Rothbard, N. (2014). Mandatory Fun: Consent, Gamification and the Impact of Games at Work. *SSRN Electronic Journal*.

Nicholson, S. (2014). A RECIPE for Meaningful Gamification. In L. C. T. Reiners, *Gamification in Education and Business.*

Nielsen, J. (1993). *Usability Engineering.*

React Dev. (s.d.). *Built-in React Hooks*. Tratto da React Dev: https://react.dev/reference/react/hooks

Rosser, J. (s.d.). *What is a Webview? And how can it drive your mobility strategy?* Tratto da https://medium.com/@jeffrey.rosser/what-is-a-webview-and-how-can-it-drive-your-mobility-strategy-cf323a4b3bcf

Ryan, R., Rigby, C., & Przybylski, A. (2006). The Motivational Pull of Video Games: A Self-Determination. *Motivation and Emotion*.

Sailer, M., Ulrich Hense, L., Mayr, S., & Mandl, H. (2017). How gamification motivates: An experimental study of the effects of specific game design elements on psychological need satisfaction. *Computers in Human Behavior*.

Savonin, M. (s.d.). *Native vs. Cross-Platform Apps: Analysis 2023*. Tratto da Keenethics: https://keenethics.com/blog/cross-platform-vs-native

Werbach, K., & Hunter, D. (2015). *The Gamification Toolkit: Dynamics, Mechanics, and Components for the Win.*

Zichermann, G., & Cunningham, C. (2011). *Gamification by Design: Implementing Game Mechanics in Web and Mobile Apps.*