Politecnico di Torino

Politecnico di Torino

Laurea Magistrale in Ingegneria Informatica
A.a 2020/2021
Sessione di laurea Luglio 2021

Ideazione e prototipazione di un sistema di gamification per il testing di applicazioni web

Relatori:
Prof. Maurizio Morisio
Prof. Luca Ardito
Dott. Riccardo Coppola

Candidato:
Davide Gallotti

Sommario

Nel pieno adempiersi della rivoluzione digitale si delinea in maniera sempre piú evidente un modo di concepire la realtá che sta cambiando, senza possibilitá di freni, il concetto stesso che abbiamo sempre avuto del mondo. Le tecnologie, i sistemi aziendali, il modo stesso di insegnare e impare le cose stanno mutando vertiginosamente. Con questa tesi si vuole andare esattamente in questa direzione, cercando di adattare l'approccio lavorativo a quello che lo sviluppo sociale e tecnologico sta da anni provando a dire: abbiamo bisogno di rendere fruibili e godibili i luoghi che viviamo quotidianamente: le scuole, gli uffici, le palestre, le universitá, le aziende in cui lavoriamo. L'obiettivo del prototipo di cui parleremo é quello di migliorare le performance, la predisposizione emotiva e la socievolezza del tester di applicazioni grafiche attraverso un sistema di gamification, indirizzando l'implementazione tecnica sugli strumenti utilizzati nel mondo dello sviluppo delle applicazioni web. Il testing, infatti, essendo una delle attivitá che piú impattano sullo ciclo di sviluppo del software, ha bisogno di ricevere un'attenzione particolare in tutti i processi produttivi. Migliorare questo aspetto dello sviluppo potrebbe avere un diretto beneficio in termini economici, andando a impattare su tutte quelle metriche aziendali che, all'essenza del discorso, fanno capire se un prodotto é valido o meno.

Indice

Parte I

Introduzione

Capitolo 1

La Fase di Testing

1.1 Ciclo di sviluppo di un software

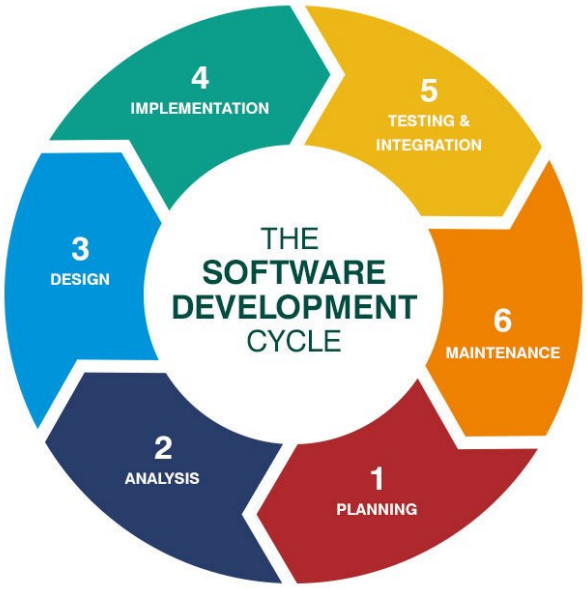


Figura 1.1. Ciclo di sviluppo di un software

Per ciclo di sviluppo di un software si intende la metodologia che scompone in varie attivitá tutto il processo di sviluppo di applicativi informatici. A differenza di molti processi, lo sviluppo di un software ha

una ricorrenza ciclica: questo implica che arrivati al punto in cui il software viene rilasciato sul mercato inizia una nuova fase chiamata **mantenimento** che comporta la reiterazione di tutte le fasi precedentemente svolte, proprio a causa dell'impossibilitá di ottenere un prodotto che sia costantemente allineato con i requisiti e le specifiche necessarie nel ciclo di vita del software stesso. In questo capitolo si approfondirá principalmente la fase di testing che sarà quella su cui si concentrerà lo studio di ricerca sull'inserimento di aspetti di **gamificazione** di cui parleremo nei capitoli successivi.

1.2 Introduzione al testing

La fase di testing é l'insieme delle procedure finalizzate a garantire il corretto funzionamento di un software per permetterne l' utilizzo da parte dell'utente finale. L'obiettivo é quello di individuare mancanze da un punto di vista di correttezza e affidabilità in modo da portare il prodotto sviluppato a rispettare le specifiche tecniche e i requisiti dettati dal cliente o dall'azienda stessa.
In sostanza il compito di un tester é quello di risolvere i malfunzionamenti dell'applicativo risultanti da errori durante la fase di sviluppo. Per avere maggior chiarezza delle terminologie utilizzate si prenderá in considerazione le definizioni data dall'IEEE:

1. **Malfunzionamento**: un funzionamento di un programma diverso da quanto previsto dalla sua specifica

2. **Difetto**: la causa di un malfunzionamento

3. **Errore**: l'origine del difetto

E' importante stabilire i vari casi d'uso del software per cercare di coprire la totalità delle possibilità di interazione. E' possibile utilizzare vari tools che aiutino il tester nel suo compito. In ambito professionale, nell'ultimo periodo, si sta andando nella direzione dei test automatizzati per facilitare il lavoro e per renderlo più stabile anche nelle fasi intermedie di sviluppo in cui il codice viene modificato in continuazione.
Oltre l'utilizzo dei tools è necessario tenere in considerazione l'utilizzo di un sistema di feedback da parte dell'utente che, eventualmente, può

segnalare la presenza di eventuali bug o problematiche, favorendo così la fase di manutenzione e di miglioramento.

1.3 Benefici della fase di testing

Testare nel migliore dei modi un software consente di ottenere numerosi vantaggi:

1. *Guadagno economico*:

   Riuscire a trovare i difetti e i malfuzionamenti di un prodotto prima della sua uscita sul mercato permette di risparmiare tempo, ma soprattutto denaro. E' molto meno costoso farlo in una fase iniziale di realizzazione che dover riprendere in mano il codice e sistemare tutto in fase di manutenzione.

2. *Sicurezza*:

   La fase di testing è necessaria anche per riuscire a indentificare falle di sicurezza nel prodotto prima che queste possano portare dei danni all'utilizzatore finale. Per guadagnarsi la fiducia del consumatore è fondamentale dare nelle sue mani un prodotto funzionante, ma soprattutto sicuro.

3. *Qualità del prodotto*:

   Il testing permette anche di aumentare la qualità di un prodotto con tutte le conseguenze che comporta. Un prodotto migliore è un prodotto più vendibile e più utile.

4. *Soddisfazione dell'utente*:

   Un prodotto migliore rende anche più soddisfatto l'utilizzatore finale permettedogli di avere un'esperienza meno frustrante e più confortevole.

1.3.1 Costi esponenziali di un bug fix

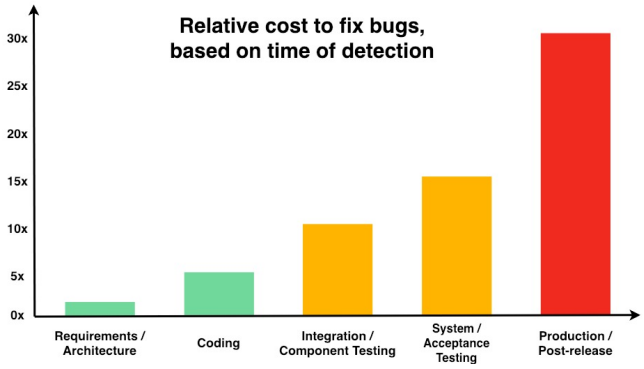


Figura 1.2. Costi esponenziali di un bug fix

Da un punto di vista economico diventa di rilevanza considerare i costi di risoluzione dei vari problemi nelle singole fasi di sviluppo. Risulta evidente come un errore risolto in una fase primordiale del ciclo di vita di un software vada ad impattare molto meno rispetto a un bug-fix in fase di manutezione. Questo sia perché la quantitá di righe di codice é minore, sia perché in una fase successiva potrebbe essere molto piú complicato modificare la struttura del software, rivelatasi solo successivamente insufficiente a garantirne il corretto funzionamento.

1.4 Tipologie di test

Di seguito vengono elencate le principali tipologie di test svolti solitamente nel corso dello sviluppo di un software.

1.4.1 Unit Test

Gli unit test sono il primo livello del testing software. L'obiettivo é quello di isolare una singola funzionalitá del programma e verificare che funzioni come da requisiti. Separare logicamente le varie porzioni del codice aiuta a identificare piú facilmente i problemi, restringendo il campo d'azione. Spesso vengono utilizzate funzionalitá di mock per affiancare gli unit test in modo da simulare il piú possibile il comportamento reale.

1.4.2 Integration Test

Gli integration test sono la fase successiva: essi si occupano di ragruppare varie funzionalitá del software per fare delle tipologie di test piú complete.

1.4.3 Graphic User Interface Test

Il test GUI è un tipo di test del software che controlla l'interfaccia utente grafica del software. Lo scopo del test dell'interfaccia utente grafica (GUI) è garantire che le funzionalità dell'applicazione software funzionino secondo le specifiche controllando schermate e controlli come menu, pulsanti, icone, ecc. Sará proprio questa tipologia di test quella interassata dallo studio fatto in questa tesi.

1.4.4 End-to-end Test

Per End-to-End si intende la verifica del flusso completo di esecuzione di un software in modo da verificare che nell'utilizzo completo di ogni funzionalitá non si riscontrino problematiche. Nel caso di software che interagiscono all'interno di una rete si parla anche di **chain test**, ovvero test della catena, in modo da verificare che ogni componente comunichi nel modo corretto con le altre. Questo test viene principalmente fatto per capire se il flusso tra il back-end dell'applicazione e il front-end venga eseguito nel modo corretto.

1.4.5 Test automatizzati

Nello stato attuale si stanno utilizzando sempre maggiormente dei tools che permettono l'esecuzione automatica di test, in modo da poter verificare piú velocemente e con costi minori le funzionalitá del software. L'automazione permette in questo modo di affrontare anche casistiche molto complesse e difficili da gestire nel modo tradizionale, rendendo di fatto l'utilizzo di questa pratica sempre piú necessaria.

1.5 Tools

I tools presi in esame sono quelli relativi al testing per applicazioni che presentano una GUI e principalmente si occupano di andare a testare che tutte le parti grafiche dell'applicativo funzionino a dovere senza incorrere in vari errori d'esecuzione e valutare che il risultato sia conforme con quello atteso. Solitamente i tools utilizzati hanno l'obiettivo

di essere il più universali possibili cercando di adattarsi alle varie piattaforme di esecuzione, tuttavia questa universalità non è sempre garantita ed è necessario andare ad agire nel dettaglio modifcando alcuni parametri

1.5.1 Automation APIs / Frameworks

Questi tools dispongono di una interfaccia costituita da tutti gli elementi della GUI disposti gerarchicamente nella view e con cui l'utente può interagire. Cercano di essere il più universali possibile, nonostante hanno alcuni casi limite in cui gli attribuiti o gli stati dell'app se non compatibili con la convenzione utilizzata possono comportare uno stato d'errore durante l'esecuzione dello script. Nella maggior parte dei casi sono cross platform, ma hanno il problema della frammentazione. Inoltre sono i costi di mantenimento sono molto alti e spesso è difficile mantenersi al passo con lo sviluppo dell'applicazione.

1.5.2 Record and Replay Tools (R&R)

Attraverso l'utilizzo di tools di questo tipo è possibile creare facilmente dei script di testing per valutare il software.Tutte le interazioni, anche le più complesse, che l'utente può fare attraverso il software gvengono registrate ed eventualmente riprodurle a schermo per verificare che tutto stia funzionando come previsto. Tuttavia il tempo necessario per verificare ogni singola casistica è abbastanza elevato e potrebbe esserne una limitazione.

1.5.3 Automated Test Input Generation Techniques

Un'altra tipologia di approccio per le app mobile è quella basata sulla generazione automatica di input di varia natura. Negli ultimi hanno si sta puntando molto su questa tecnica che però presenta ancora delle sfide irrisolte che comprendono la generazione di eventi di sistema, il costo di riorganizzare l'app e quello di inserire input manuali per specifiche operazioni più complesse.

1.5.4 Bug and Error Reporting/Monitoring Tools

Alcuni tools si occupano di creare dei report automatici sui bug ed errori dell'app. Solitamente sono integrati con dei sistemi di feedback dati dall'utente stessi che può inviare un report di una sua problematica e aiutare più facilmente i tester nel risovlerla. Il limite è basato sul tipo

di report possibili attraverso sistemi automatizzati che si limitano per lo più ad essere crash o eccezioni restringendo l'utilità degli stessi.

1.6 Continous, Evolutionary e Large scale

Durante lo sviluppo di applicazioni mobile sta prendendo piede il paradigma CEL (Continous, Evolutionary e Large scale). Il panorama dei dispositivi mobile è molto vasto e questo comporta un'attenzione maggiore durante la fase di test:

1. Continous (Continuoità): é importante che la fase di sviluppo e di test procedano di pari passo. Il software deve essere continuamente testato seguendo l'evoluzione degli obiettivi da raggiungere che possono variare durante tutto il ciclo di sviluppo.

2. Evolutionary (Evoluzione): Il codice sorgente e gli "artefatti" dei test non devono evolversi indipendemente l'uno dall'altro. Ogni modifica del codice sorgente e dell'eventuale obiettivo da raggiungere deve essere seguita accuratamente dai test in modo che si possa valutare più accuratamente che il prodotto sia conforme a quello desiderato.

3. Large Scale: la grandissima quantità di dispositivi mobile obbliga gli sviluppatori ad assicurasi che il prodotto sia stabile e funzionante per tutti i dispositivi che si intende supportare. I test devono valutare anche la diversità di dispositivi su cui l'applicazione girerà, in modo da evitare malfunzionamenti. E' possibile utilizzare un engine che permetta di testare il codice scritto per tutti i dispositivi in modo da testare le reali condizioni dei vari dispositivi e di conseguenza indentificare i problemi eventuali.

1.6.1 Continous Integration / Continous Development (CI/CD)

Il CI/CD é una pratica che permette di distribuire in maniera continua il software in fase di sviluppo automatizzando alcune procedure relative alla build, al testing e al deployment dell'applicativo. Principalmente viene utilizzato in contesti in cui é presente un versioning del codice scritto, con l'appoggio di repository online che contengono tutto l'albero delle modifiche fatte nel corso delle varie versioni. Prima di rilasciare

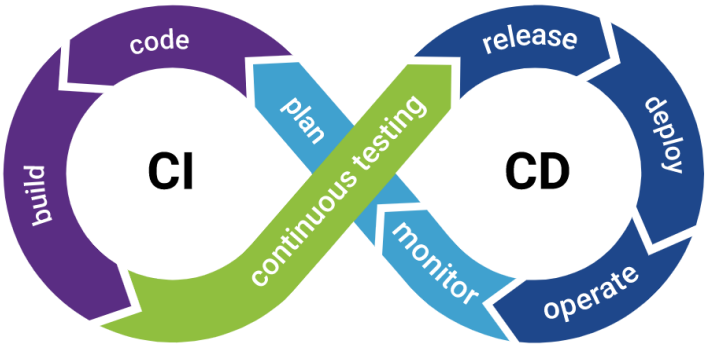


Figura 1.3. CI/CD

una nuova versione, vengono eseguite in automatico una serie di operazioni automatizzate, solitamente lato server in modo da distribuire il carico di operazioni e favorire la collaborazione tra vari team di sviluppo. E' una pratica che viene sempre piú usata e favorisce la riuscita di un buon prodotto dopo ogni versione di rilascio.
I software di versioning piú utilizzati sono **GIT** e **SVN**. Una delle piattaforme piú avanzate nell'ambito del CI/CD é sicuramente **GitLab** che permette la creazione di pipeline di operazioni all'inserimento di una nuova versione del software in un determinato branch della repository.

Capitolo 2

Applicazioni mobile e web

2.1 Il Web e i dispositivi mobile

Il Word Wide Web, in gergo chiamato Internet, rappresenta allo stato attuale uno degli strumenti tecnologici più utilizzati di sempre. Il tasso di diffusione nella maggior parte dei paesi sviluppati ha raggiunto valori tra il 90 e il 100 per cento. Nei paesi in via di sviluppo, sebbene il tasso di crescita è di gran lunga inferiore, riesce a trovare sempre più rapida diffusione, anche grazie ai lavori di ampliamento delle infrastrutture di rete, concedendo l'accesso alla "Grande Ragnatela" ad un numero sempre maggiore di utenti. La grande connessione di dispositivi ha favorito la nascita di applicazioni informatiche legate direttamente alle tecnologie web, grazie all'utilizzo di server in grado di fornire servizi agli utenti. La motivazione principale per cui le applicazioni web stanno iniziando a diventare sempre più preponderanti nel mercato software è da ritrovarsi principalmente nella necessità di fornire all'utente un servizio sempre aggiornato, eliminando l'onere dell'utente di aggiornare continuamente il software. La versione piú affidabile viene distribuita direttamente dallo sviluppatore, rendendo di fatto molto più semplice tenere i propri utenti al passo con i tempi e con le falle di sicurezza che potrebbero comparire negli anni.
Dall'altro lato uno degli strumenti più utilizzati quotidianamente è lo

smartphone, oggetto in grado di rendere ormai possibile qualsiasi operazione informatica partendo dalla sua funzione principale, la comunicazione telematica, arrivando ai sempre più diffusi pagamenti elettronici. Le applicazioni mobile diventano perciò uno strumento fondamentale per qualsiasi settore e azienda, rendendo di fatto necessaria un'accurata fase di testing, in grado di fornire sicurezza e solidità. Molti settori critici quali finanza, istruzione e salute sono attualmente integrati completamente con applicazioni di tipo informatico basate su tecnologie web o mobile. Per questo motivo non si può ignorare l'importanza strategica che le aziende inforamtiche operano all'interno della nostra società. Il ruolo del programmatore è essenziale e il corretto funzionamento di un'applicazione può influenzare in modo sostanziale la vita delle persone.
In poche parole le applicazioni mobile e web sono ormai diventate fondamentali nei contesti quotidiani in cui l'uomo del ventunesimo secolo si trova a vivere e occorre porsi nuove domande su come migliorare la qualità dei software prodotti e la salute psicologica delle persone che si trovano a lavorare in questo ambito. Il ruolo del tester infatti in molti contesti può diventare frustrante, soprattutto data la monotona routine di esecuzione. Bisogna iniziare a percorrere strade alternative, alla ricerca di quello che potrebbe essere oggi,una rivoluzione nel modo di concepire il lavoro e generalmente le attivitá produttive dell'ingegno umano.

2.2 Analogie tra applicazioni mobile e web

La velocità di diffusione delle applicazioni web ha favorito la nascita di nuovi approcci di sviluppo basati sull'utilizzo di librerie o framework in grado di facilitare gli sviluppatori nella progammazione. Il linguaggio più utilizzato nel mondo web è Javascript, ma sempre piú aziende stanno investendo in nuove tecnologie per rendere lo sviluppo di un software indipendente dalla piattaforma su cui sta girando. Le grandi aziende informatiche come Google e Facebook si sono adoperate per creare i propri framework di sviluppo.

2.3 Flutter

Flutter é un framework sviluppato da Google principalmente per applicazioni mobile. Nella nuova versione (2.2 uscita a Maggio 2021) é

prevista una versione stabile per lo sviluppo di web app. Il linguaggio di programmazione é Dart. Fa parte di tutta quella categoria di framework cross-platform pensati per velocizzare lo sviluppo per piattaforme diverse. É il piu recente dei framework e proprio per questo motivo é considerato ancora una tecnologia acerba, sebbene stia prendendo velocmente piede.

2.4 React e React Native

React e React Native sono dei framework rispettivamente per applicazioni web e applicazioni mobile sviluppate da Facebook. Sono la principale alternativa a Flutter , ma sono attualmente considerati dei framework molto piú collaudati, dato che esistono da circa 7/8 anni. Il linguaggio di programmazione é Javascript il che li rende molto versatili e facili da imparare per gli sviluppatori web tradizionali. Condividono molte caratterstiche sebbene abbiamo la parte di composizione dei widget adattata al dispositivo su cui viene applicata.

2.5 Angular, React e Vue

Nel ambito web Angular é il framework piú anziano. Il piú popolare rimane peró React. Vue.js nasce come alternativa piú leggera di Angular.

1. Angular: piú completo, all in one
2. React: improntato all'UI, necessita di Redux per lo state-managment
3. Vue: alternativa rapida e con curva di apprendimento bassa.

2.6 Ionic e Cordova

Ionic e Cordova consentono a un'applicazione Web di essere raggruppata in un'app nativa che può essere installata su un dispositivo mobile. Incorporano una visualizzazione Web, che è un'istanza del browser isolata che esegue le applicazioni web su qualsasi dispositivi su cui viene installato. Inoltre entrambi forniscono API per interagire a livello nativo con il dispositivo, rendendo utilizzabili anche il GPS o la fotocamera. Peccano in prestazioni, ma essendo basati in HTML, CSS e Javascript sono solitamente considerati framework piú accessibili e meno costosi.

2.7 Software per il testing

2.7.1 Selenium

Il tool di testing piú utilizzato commercialmente é Selenium. Permette di svolgere numerosi tipologie di testing:

1. Acceptance testing
2. Functional testing
3. Performance testing
4. Load testing
5. Stress testing
6. Test driven development
7. Behavior-driven development

Attraverso un WebDriver permette di interagire con l'applicazione web e sono presenti numerose API disponibili nella maggior parte dei linguaggi di programmazione piú usati. Attraverso il Selenium IDE é possibile scrivere i test cases. Inoltre é possibile registrare una serie di azioni da svolgere sull'applicazione e farle eseguire in sequenza, verificano che tutto stia procedendo come previsto. Esiste anche un ulteriore modulo chiamato Selenium Grid che permette di eseguire in parallelo diverse tipologie di test su piú server.

2.7.2 Cypress

Cypress é framework basato su Javascript per test end-to-end. É sviluppato sopra Mocha e gira su direttamente su Browser, abilitando test asincroni. Cypress puó essere utilizzato anche per unit test e integration test. A differenza di Selenium, non é possibile lanciare test su due browser differenti e supporta solo API basate su Javascript per interagire con le pagine web.
Cypress include diverse caratteristiche tra cui:

1. Screenshots e video automatici al fallimento di test o video interi durante tutta la test suite.

2. Log degli errori e stack traces utilizzabili per fare debug in modo piú semplice

3. Disponibile su diversi Browser tra cui i piú comuni (Chrome, Firefox, ecc..)

4. Utilizza attese automatiche senza necessitá di aggiungere sleep nel codice

2.7.3 Screenster

Screnster é un software con licenza a pagamento che permette di riconoscere cambiamenti nell'UI dall'esecuzione di un test ad un altro. Integra permettamente Selenium, con il quale é possibile scrivere i proprio test customizzati. Risulta essere un tool piú rapido ed efficiente, evitando per alcune tipologie di test di scrivere codice. Ha una serie di funzionalitá che permettono di schedulare i test da fare in cloud,vederne la revisione e consentendo di fare anche piú test in parallelo. Inoltre non é necessario testare la pagina web nel suo complesso, ma é possibile selezionare una zona specifica e testare solo quella.

Capitolo 3

Gamification

3.1 Introduzione alla Gamification

All'interno di un team di sviluppo, il ruolo del PM (Project Manager) oltre quello di gestione delle prioritá e delle attivitá, ha iniziato a essere anche quello di creare l'ambiente più profittevole e confortevole possibile. Per progetti di sviluppo software si cerca sempre di raggiungere risultati nel modo più veloce e proficuo possibile. Uno dei metodi piú studiati e utilizzati in questi anni per migliorare il rendimento è un processo chiamato "gamification". L'obiettivo delle aziende che stanno iniziando a utilizzare questo metodo è quello di ricevere risultati migliori e di migliorare l'integrazione dei vari team di lavoro.

La gamification è un metodo di lavoro che utilizza meccaniche tipiche del gioco in contesti non ludici. E' possibile utilizzarla in una moltitudine di situazioni quali apprendimento, allenamento fisico, salute, politica e tecnologia. Può essere banalmente utilizzato anche per cercare di creare un'interazione più fidelizzata tra l'utente di un sistema e il sistema stesso, ad esempio in sistemi di marketing. Nel caso dello sviluppo di software può essere utilizzato per rendere meno tedioso il lavoro del tester e può aggiungere una forma di competitività sana all'interno dell'azienda.

La gamification utilizza dinamiche di gioco (storia, emozioni, relazioni), meccaniche di gioco (sfide, condivisione, vittorie) e componenti di giochi (classifiche, trofei, livelli, obiettivi). Questi elementi possono essere integrati in qualsiasi contesto.

3.2 Dinamiche, meccaniche e componenti

Prima di parlare di esempi di gamification è necessario evidenziare nel dettaglio quali sono gli elementi che compongono un gioco e quali sono le motivazioni che portano questa metolodogia a essere interessante nei più svariati ambiti.
Il Gioco è un'attività che accompagna l'uomo in ogni fase della sua vita. L'approccio al gioco crea delle aspettative, dei desideri e dei bisogni. Giá nel 1938, Johan Huizinga, nel suo libro "Homo Ludens", lascia intendere proprio tramite l'assonanza a "Homo Sapiens", che il gioco in realtá é una delle caratteristiche principali dell'essere umano. La sua complessitiá, nonostante anche molti animali giochino, rende questa pratica indetificativa della nostra specie. Nel suo libro cerca di evidenziare come tutte le caratterstiche del gioco andrebbero rivalutate e studiate nei contesti sociali a cui solitamente vengono utilizzate. Queste caratteristiche si indentificano in quelle che vengono chiamate **dinamiche** di gioco. Nell'esempio specifico di un ambiente lavorativo, le dinamiche di gioco possono essere inquadrate attraverso il desiderio e il bisogno di rendere l'attività professionale meno frustrante e più gratificante, producendo risultati positivi non solo a livello psicologico ma anche produttivo.
Affinché il gioco possa produrre questi risultati si creano alcune **meccaniche** che attraverso regole ben determinate permettono al giocatore di muoversi nel mondo di gioco e interagire con esso. Nel caso specifico di un tester le meccaniche di gioco potrebbero essere indetificate in tutte quelle regole che stabiliscono gli obiettivi da raggiugnere e i punteggi assocciati.
Le meccaniche di gioco devono essere il più chiare possibili in modo da permettere al giocatore di interagire con le **componenti** di gioco nel modo più naturale. Le componenti sono la realizzazione concreta delle meccaniche: l'idea astratta di punteggio può tradursi in una classifica, mentre il concetto di casualitá puó essere rappresentato attraverso l'utilizzo di un mazzo di carte. L'insieme di dinamiche, meccaniche e componenti indentificano un gioco e sono la base di partenza per crearne uno.

3.3 La tassonomia di Bartle

Richard Bartle nel suo articolo "Hearts, clubs , diamonds, spades: players who suit MUDs" tenta di fare una classificazione rigorosa delle tipologie di approccio possibili al gioco. Nell'articolo vengono analizzate le attivitá tipicamente piú apprezzate:

1. Realizzazione nel contesto del gioco
2. Esplorare il gioco
3. Socializzare con gli altri
4. Imporsi sugli altri

Da questa iniziale tassonomia viene poi estrapolato un grafico che analizza principlamente due dinamiche tipiche del gioco (Azione e Interazione) e due componenti principali (Mondo e Giocatori). Ogni individuo puó identificarsi in uno spazio bidimensionale rappresentabile da questo grafico in grado di evidenziare una determinata "personalitá" nel contesto ludico.

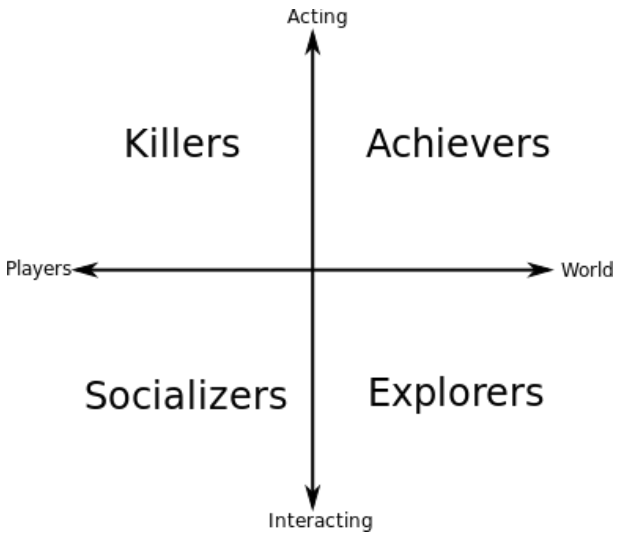


Figura 3.1. Tassonomia di Bartle

Attraverso questa categorizzazione é possibile progettare un sistema di gamificazione piú efficace, concentrandosi principalmente sulle caratteristiche tipiche di chi effettivamente userá il servizio.

3.4 Octalysis

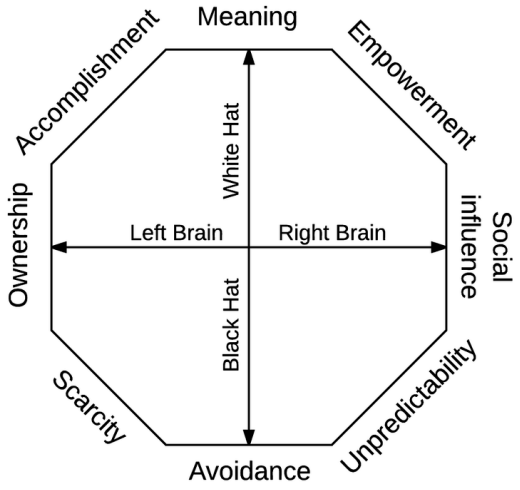


Figura 3.2. Octalysis

L'Octalysis é un framework invenato da Yu-Kai Chou, esperto di design comportamentale. Nel suo libro "Actionable Gamification" sostiene che un buon design di gamification debba essere valutato secondo otto punti cardine, che rappresentano in maniera astratta i vertici di un ottagono: l'octalysis. Questo ottagono puó essere utilizzato, oltre che per creare, anche per analizzare servizi esistente che, anche se non in modo preponderante, contengono al loro interno tutti i punti (analogamente al loro obiettivo) che Yu-Kai Chou tratta nel suo libro principale.

3.4.1 Chiamata

Il primo punto analizzato é la "Chiamata", ovvero il desidero instrinseco dell'uomo di partecipare a progetti piú grandi di sé e soprattutto con un intento risolutivo rispetto a varie problematiche. Tipicamente utilizzato come espiediente letterario, la chiamata dell'eroe, é in realtá un desiderio che almeno una volta nella vita ognuno di noi ha provato. Esempi pratici di servizi che utilizzano questo espediente sono:

1. Wikipedia: un servizio di enciclopedia libera e aperta a tutti, in cui

ognuno puó dare il suo contributo condividendo le proprie conoscenze. Attualmente rientra tra i 10 siti piú visitati del mondo.

2. Treedom: un progetto nato per finanziare e supportare tutte quelle le piccole e medie realtá rurali, con lo scopo di creare ecosistemi sostenibili garantendo sovranitá alimentare ai paesi in via di sviluppo e possibilitá di sostentamento economico. Attraverso il loro sito é possibile regalare un albero, piantando la specie selezionate in un territorio a scelta dell'utente.

3. "Think Different": é il nome della piú grande campagna pubblicitaria della Apple. Attraverso questo slogan, la grandissima multinazionale americana é riuscita a ritagliarsi una grossa fetta di mercato, instillando nelle persone un forte "senso epico". Utilizzare un prodotto Apple non doveva riguardare la tecnologia in senso tecnico, ma trasportava il suo utilizzatore nel futuro, rendendolo di fatto partecipe del cambiamento che Apple stava portando nell'ecosistema informatico ed elettronico attraverso i suoi prodotti.

3.4.2 Potenziamento

Il secondo principio fondamentale dell'Octalysis si basa su quella sensazione scaturita da una profonda crescita personale e dal raggiungimento dei propri obiettivi. É la spinta che ci fa concentrare sul nostro percorso di vita, che ci rende entusiasta di apprendere nuove conoscenze e in un perfetto ciclo che si auto alimenta genera una serie di sensazioni positive che ci spingono a fare sempre di meglio. Un esempio di servizio che sfrutta questa tendenza umana al "Potenziamneto" é uno dei primi siti di E-Commerce nati sul web: Ebay. Ebay infatti rivoluzionó il modo di concepire le interazioni degli utenti nel proprio sito, creando dei sistemi di feedback realitivi al numero di vendite di un determinato negozianate, affiancandolo ad un sistema di votazione che fu poi copiato da tutti: un numero di stelle da uno a cinque che identificano il livello di gradimento dell'acquirente rispetto al prodotto ricevuto. Ebay fu uno dei primi siti a istituire un sistema di "Obiettivi" basato sui risultati ottenuto attraverso i lori servizi. Nei suoi primi anni di attivitá infatti, l'azienda americana di e-commerce, utilizzava un sistema di

certificazione chiamato *"The Yellow Star Award"* che premiava il negozioante attraverso un riconoscimento simbololico dopo suoi primi dieci prodotti venduti.

3.4.3 Realizzazione

Per descrivere a pieno il concetto espresso dal terzo principio dell'Octalysis non si puó ignorare la grandissima multinazionale di produzione di giocattoli LEGO. L'idea principale espressa dalla conquista e la consapevolezza di utilizzare la propria creativitá per raggiungere un obiettivo é proprio quella che si nasconde nella semplcitá dei mattoncini di plastica colorata nati nel XX secolo: da delle semplici linee guida l'utilizzatore del servizio, in questo caso il giocatore, puó esprimere a pieno la sua creativitá realizzando qualcosa di non direttamente previsto dal creatore, andando a influenzare tutti quei meccanismi di auto compiacimento derivati dalla soddisfazione di raggiungere solo con i propri sforzi e le proprie idee un determinato obiettivo. Un esempio perfetto di questo principio dell'Octalysis é il Crowdsourcing, ovvero quel modello di sviluppo colletivo di un progetto basato sulla copartecipazione di un gruppo di persone, solitamente esterne ad un'azienda. In genere viene proposto a studenti, che si adoperano in una proficua collaborazione per la generazione di un'idea o di un progetto che possa risolvere un determinato problema (collegandosi anche con il principio della "Chiamata dell'Eroe"). Un altro esempio di ottimo utilizzo di questo principio é quello utilizzato da molte aziende per incentivare i propri dipendenti ad utilizzare le scale invece che l'ascensore. L'obiettivo solitamente é di rendere divertente o stimolante salire le scale al fine di migliorare e rafforzare le sane abitudini del dipendente. In Svezia ad esempio grazie a The Fun Theory é nato il progetto "The Piano Staircase", in cui le scale di una metropolitana sono state trasformate attraverso degli adesivi in tasti di un pianoforte. Molte aziende, solitamente nel Nord Europa, invece scrivono il numero di calorie consumate raggiungendo quella specifica porzione di scala, in modo che il dipendente abbia un feedback chiaro e instantaneo del benificio ricevuto dalla scelta di utilizzare le scale per raggiungere il proprio ufficio.

3.4.4 Possesso

Questo principio nasce dalla motivazione che ci porta a desiderare di possedere qualcosa che spesso tende a far crescere in noi il desiderio di proteggere e migliorare quello che otteniamo e a volta ci stimola a ottenere qualcosa di ancora migliore. É interessante notare che questo principio non si applica solamente a oggetti materiali, ma si lega a tutti quei beni virtuali che ottieniamo principalmente dalle nostre attivitá sul Web: monete virtuali, oggetti virtuali ecc.. L'esempio perfetto é il Tamagochi, ovvero un piccolo dispositivo elettronico con un display che faceva interagire con un animale virtuale al suo interno. Questo pattern in realtá puó essere sfruttato in numere altri servizi, tenendo conto che il senso di esclusivitá di un determinato prodotto possa portare l'acquirente o l'utilizzare ad essere invogliato ad utilizzare o semplicemente possedere il servizio proposto. Il collezionismo nasce proprio da questo principio psicologico che spinge l'uomo a possedere gli oggetti. Un utilizzo in ambito informatico é quello creato da Google Analytics in cui é presente un dashboard contenente tutti i dati realtivi al proprio servizio offerto, in forma grafica. Questo contribuisce a dare all'utente Google un senso di possesso e realizzazione rispetto a determinati obiettivi che si pone, evidenziando in maniera chiara e evidente il numero di interazioni e di conseguenza l'impatto che si sta avendo sul mondo. Questo aiuta dare un senso di possesso e desiderio di miglioramento sui dati visualizzati, influezando a tutti gli effetti le statistiche raggiunte ad esempio dal proprio sito o blog.

3.4.5 Influenza Sociale

Questo principio rappresenta l'influenza che un gruppo sociale applica sull'individuo. Ci sono numerosi esempi nella vita quotidiana che portano l'uomo a modificare il suo comportamento in funzione di una determinato gruppo sociale. Ad esempio puó risultare difficile parlare in pubblico, per paura del giudizio altrui. Questo lato dell'ottagono é quello che va a spiegare la motivazione che proviamo nel momento in cui ci mettiamo alla prova e ci sfidiamo con qualcuno. Confrontarsi con gli altri infatti é uno degli stimoli piú grandi che ci spinge a fare o non fare qualcosa. É fondamentale per questo motivo riuscire a progettare in termini di servizi basati su elementi di gamification una forma di competizione sana, che non sia frustrante per l'utilizzatore ma che

possa piuttosto aiutarlo a mettere tutto il suo impegno nell'attivitá che si vuole potenziare. In ambito lavorativo utilizzare meccaniche di sfida tra i vari dipendenti non é sempre una scelta sensata: come prima cosa non puó essere utilizzata per lavori creativi, in cui é difficile stabilire dei criteri oggettivi di valutazione dei risultati ottenuti. Inoltre la sfida deve essere limitata ad un tempo ristretto, in modo da non diventare frustrante con il passare del tempo. Per questo un'ottima strategia di design per sistemi gamificati che sfruttano questo principio e di proporre delle sfide in un arco di tempo limitato (ad esempio un paio di settimane l'anno).Altro parametro fondamentale é quello di assicurasi che ogni partecipante abbia la sensazione di poter vincere la sfida proposta, assicurandosi quindi di non mescolare persone con background troppo diversi sia di competenze che di esperienza.

3.4.6 Scarsità

Questo principio si basa su quella motivazione derivata dal desiderare qualcosa che non si puó avere nell'immediato o é molto difficile da ottenere. É il principio su cui si basano le offerte a tempo limitato, le edizioni limitate di vari prodotti, ma anche quella sensazione istintiva che ci fa credere che ció che costa di piú é migliore. Una tecnica molto utilizzata é il "Torture break" ovvero limitare le interazioni a un servizio in base al tempo: ad esempio puoi fare un numero di partite fisse al giorno, oppure devi aspettare un determinato lasso di tempo per sbloccare la successiva. É una tecnica relativa alla Black Hat Gamification, tuttavia utilizzata in modo corretto puó portare una concreta motivazione.

3.4.7 Imprevedibilità

Il settimo principio dell'Octalysis si basa sull'innata curiositá umana e sulle sensazioni che si provano nel momento in cui l'esito di un determinato evento é imprevedibile. Il primo esempio che si puó fare fa parte sicuramente del Black Hat Side, ovvero il gioco d'azzardo. Tuttavia esistono versioni del gioco d'azzardo che si basano sullo stesso principio che peró non sono cosí negative come ad esempio gli Easter Eggs. Utilizzare questo principio in maniera sana é complicato, ma un esempio virtuoso puó essere il pulsante "Mi sento fortunato" del motore di ricerca Google: digitando la parola chiave desiderata e cliccando il pulsante

si viene rindirizzato in un sito web casuale associato a quella specifica parola chiave, stimolando sia la curiositá dell'utente che l'utilizzo del servizio.

3.4.8 Perdita

La motivazione dettata dalla paura di perdere qualcosa é forse uno dei piú terribili principi su cui basare un sistema di Gamification per qualsiasi tipo di servizio. Nonostante questo dosare questo elemento nella giusta quantitá puó essere davvero un grande stimolo per migliorare la produttivitá di alcune attivitá. Bisogna stare attenti soprattutto ad evitare il senso di frustrazione e demotivazione derivato dal perdere continuamente i propri progressi, ma rendere qualcosa meno istantaneo e piú in un certo piú sofferto aiuta ad aumentare la capacitá di soddisfazione che scaturisce dal sentire di meritare e di essersi guadagnati quello che si ha. Inevitabilmente questo principio va a completare molti di quelli precendenti come il numero 6 (Scarcitá) soprattutto nelle situazioni in cui non cogliere una determinata occasione, ad esempio un'offerta limitata del prodotto desiderato, ci porta inevitabilmente a perderla. Un altro collegamento puó essere fatto con il principio dell'influenza sociale che spesso governa le nostre azioni per paura di perdere la stima delle persone che ci sono vicine.

3.4.9 La tassonomia di Bartle nell'Octalysis

Riprendendo le 4 tipologie di giocatore descritte da Bartle (Explorer, Achivier, Socializer, Killer) é evidente che non tutte si relazionano agli 8 principi dell'Octalysis allo stesso modo. Potrebbe per questo motivo risultare utile ai fini di un buon design considerare ulteriori livelli di progettazione rendendo l'Octalysis un sistema a due dimensione dove per ogni tipologia di giocatore si affianca un determinato grafico di proprietá del sistema che si vuole costruire. Un ulteriore livello puó riguardare anche le varie fasi di utilizzo del sistema di Gamification (Scoperta, Inserimento, Impalcatura,Fine Gioco). Ognuna di queste fasi di esplorazione stimola diverse dinamiche a seconda delle quali vanno scelte le meccaniche e quindi le componenti corrette per il sistema progettato.

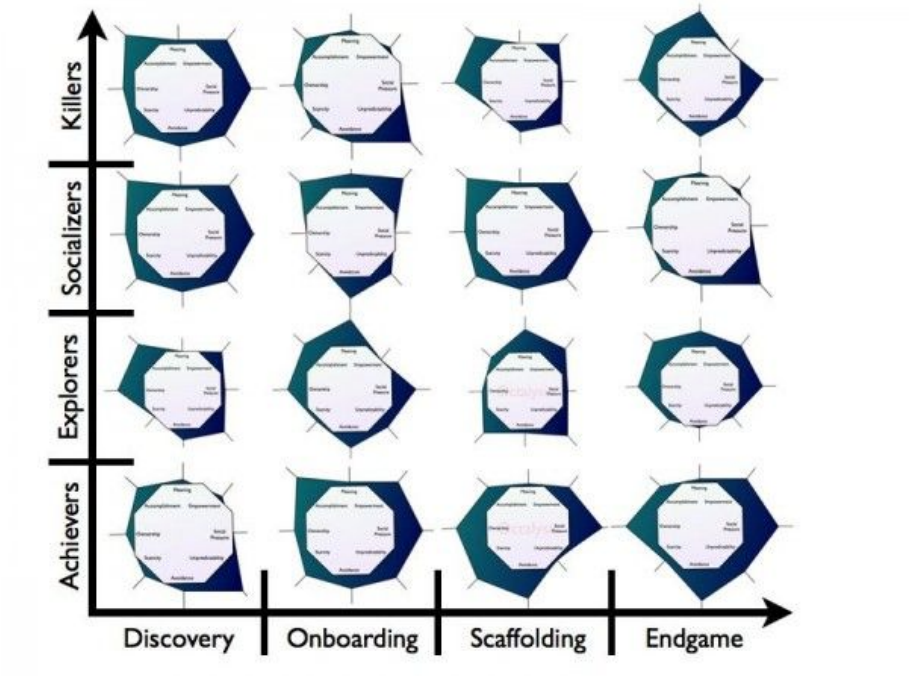


Figura 3.3. Octalysis a 2 dimensioni

3.5 Black Hat and White Hat Gamification

Ognuno dei principi di Gamification trattati dall'Octalysis puó essere categorizzato in due grandi spazi "etici". Nel momento in cui ci si pone come obiettivo quello di utilizzare delle dinamiche psicologiche legate al gioco per coinvolgere il giocatore, diventa fondamentale utilizzare tali conoscenze e tali emozioni nel modo piú corretto possibile per evitare di sfruttare in modo poco sano la sfera emotiva dell'utilizzatore del servizio. Per questo motivo, se si vuole fare Gamification in modo etico, sarebbero da preferire tutte quelle dinamiche che stimolano il giocatore in un percorso personale di crescita piuttosto che di "distruzione". Prendendo ad esempio il principio "Perdita" potrebbe risultate frustante e poco produttivo per la psiche della persona che si approccia ad un sistema focalizzato sulla paura. Black Hat e White Hat Gamification indicano proprio questo: l'utlizzo delle dinamiche positive e negative legate all'essere umano.

3.6 Gamification nel Software Development

All'interno dello sviluppo software la gamification è studiata da poco. I primi studi risalgono circa al 2011. Ma già negli anni successivi è stato preso maggiormente di mira per approfondire meglio le possibilità che poteva offrire questo metodo di lavoro. La gamification attualmente non viene utilizzata in tutte le fasi di sviluppo. La percentuale di utilizzo della gamification nelle varie fasi è:

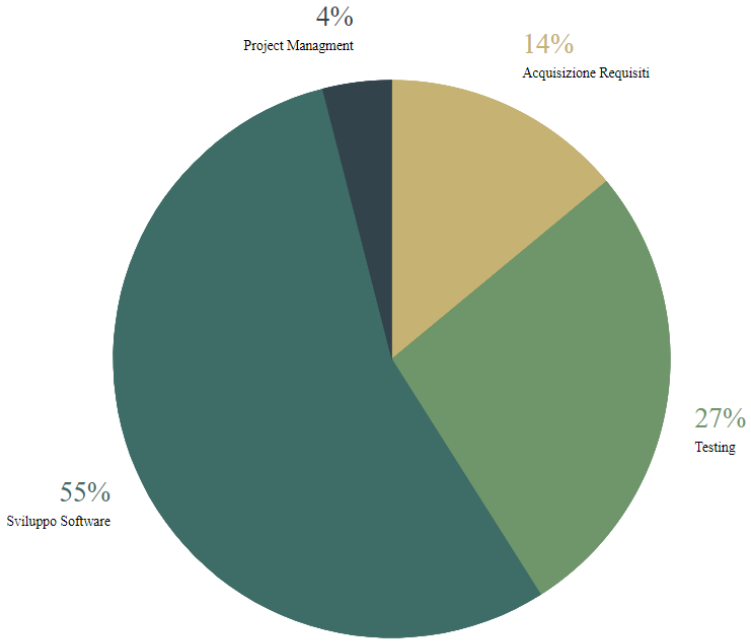


Figura 3.4. Gamification nel Software Development

Utilizzandolo in questi contesti si cerca sempre di motivare il programmatore o comunque il dipendente che deve occuparsi di un determinato task. Si possono tracciare i progressi del lavoro e poi confrontarli con altri team. Nella fase di testing principalmente si cercano di privilegiare i test automatizzati e la tracciabilità degli errori.

3.7 Benefici della Gamification

In molti casi l'utilizzo di elementi di gamificazione ha favorito la produttività dei dipendenti, riducendo gli errori, ottenendo risultati migliori e

mitigando la tediosità dei vari lavori. Nella fase di raccolta dei requisiti è aumentato il numero di idee e proposte, evidenziando un notevole aumento della creatività e della partecipazione. Nella fase di sviluppo la qualità del codice è risultata essere migliore: erano presenti meno errori, ed è aumentata la percentuale di errori trovati durante la stesura del codice. Nella fase di testing a risentirne positivamente è stata la qualità del prodotto finale, la qualità del codice (test automatizzati) e l'apprendimento reciproco. I tester possono avere un riscontro pratico del loro lavoro che può essere valutato direttamente attraverso un voto o un risultato pratico. Gli elementi "competitivi" hanno favorito una collaborazione dei team in cui fosse possibile tracciare anche un progresso del lavoro visibile a tutti attraverso classifiche, badge o punteggi.

3.8 Prolemi della Gamification

Solitamente non sono stati evidenziate problematiche da parte degli utenti in meccaniche di sviluppo gamificate. In alcuni casi però è stato osservato un aumento dello stress dopo l'utilizzo a lungo termine che potrebbe compromettere la produttività dell'intero team. La parte più complicata però rimane l'inserimento della gamification nei progetti. Oltre tutte le scelte progettuali vanno aggiunte anche quelle relative alla gamification tra cui la scelta del software, la sua implementazione, le sue meccaniche e il suo effettivo utilizzo (team o azienda esterna che si occupa solo di questo? Valutare convenienza effettiva). Per questo motivo la maggior parte dei risultati sono stati valutati fino ad adesso su ristretti gruppi di test per un periodo di massimo 5 mesi.

3.8.1 Integrare un processo di sviluppo con la gamification

Potrebbe risultare complicato modificare le abitudini e gli standard delle aziende riguardo le procedure con cui sviluppano i loro prodotti. L'inserimento di un sistema produttivo diverso all'interno di un'azienda é un processo complesso che va seriamente preso in considerazione dal PM per valutare se all'interno del contesto lavorativo abbia senso o meno utilizzare metodologie alternative a quelle giá attualmente in uso. Per queso motivo risulterebbe molto utile l'intervento di un'azienda esterna specializzata in realizzazione di sistemi gamificati costitutita

da team di esperti con competenze trasversali tra cui psicologi, ingegneri, economisti in modo da poter quantificare e valutare un sistema ad hoc per l'azienda in questione e presentarlo all'interno di quello specifico contesto con il minor disagio possibile. Attualmente una famosa azienda che si occupa di questo é Laborplay, situata in provincia di Firenze e che collabora con le piu grandi multinazionali. Tuttavia questa metodologia é ancora in fase sperimentale, nonostante sempre maggiori evidenze stiano dimostrandone l'efficacia. Rimangono perció aperte numerose possibilitá in questo settore, anche in contesti diversi come quello educativo.

3.9 IBM Open Badges

Un esempio di utilizzo di sistemi di Gamification in aziende famose é riscontrabile nella famosa IBM che grazie ai suoi *Open Badges* distribuisce dei riconoscimenti ai suoi dipendenti tramite delle medaglie virtuali che possono attestare i risultati ottenuto dal professionista in questione. L'ottenimento di questi badge é raggiungibile solamente attraverso un esame che possa attestare una determinata competenza, rilasciando cosí delle certificazioni riutilizzabili anche in altri contesti lavorativi.

Parte II

Progetto

Capitolo 4

Gamification Engine

La progettazione di un servizio di gamificazione relativa al testing grafico di applicazioni nasce con la tesi di Tommaso Fulcini: "Gamification applicata al GUI testing di applicazioni web". In questo capitolo si vogliono analizzare le precedenti componenti del sistema sviluppato e proporre un progetto che vada a migliore e implementare tutte le principali problematiche precedentemente riscontrate.

4.1 Scout

4.1.1 Augmented Testing

Il sistema di Gamification da cui si parte per questa tesi si basa su dei plugin in Java integrati in un software prototipale che implementa l' **Augmented Testing** chiamato **Scout** (Nass et al.,2020). La possibilitá di integrare questo software in maniera libera rende l'utilizzo di Scout un'ottima scelta (nonostante il codice sorgente non sia disponibile) in contesti, come quello attuale, in cui si vuole modificare radicalmente l'ambiente di lavoro del tester. Il concetto di Augmented Testing si basa sull'integrazione nell'interfaccia grafica di elementi aggiuntivi che servono a rendere piú semplice e intuitiva l'interazione con l'applicazione web (o mobile) oltre ad aggiungere eventuali caratteristiche come quelle che di seguito tratteremo.

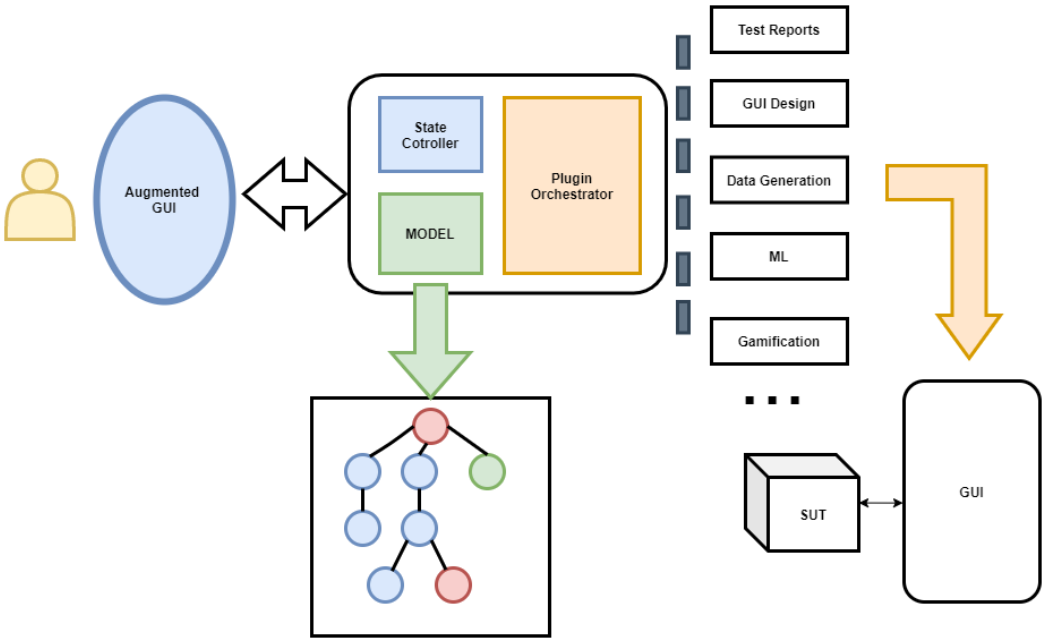


Figura 4.1. Scout schema

4.1.2 Struttura

Attraverso un sistema State-Model Scout permette di conservare e gestire tutti gli stati e le interazioni che l'utente compie all'interno dell'albero rappresentate il modello dell'applicazione web (o mobile) che sta testando. Il suo sistema permette di includere tutte le dipendenze e le funzionalita che servono all'applicazione per funzionare. In questo modo partendo dall'applicazione web, viene creata un interfaccia grafica che implementa tutti i plugin selezionati per quel contesto, viene creato il modello e attivato lo state controller in grado di registrare le interazioni dell'utente. Il tutto viene combinato per creare l'interfaccia aumentata con cui l'utente si interfaccerá in ultima istanza. Attraverso il mouse é possibile navigare tra i vari rami dell'albero, aggiungere eventuali errori e confermare che il widget selezionato sia conforme agli standard produttivi del software. Attraverso i vari plugin si possono analizzare tutte le varie informazioni raccolte durante il testing.

4.2 Componenti precedenti

4.2.1 Punteggio

Il punteggio é una valutazione che ha lo scopo di creare una metrica oggettiva per valutare la qualitá della sessione di testing appena svolta. Alla fine di ogni sessione viene visualizzata una finestra di riepilogo in cui l'utente puó visionare i punteggi totali e parziali relative alle metriche introdotte dal plugin di Gamification. Il punteggio viene calcolato secondo la seguente formula:

$$\mathbf{P} = \mathbf{a} \cdot \mathbf{C} + \mathbf{b} \cdot \mathbf{EX} + \mathbf{c} \cdot \mathbf{EF} + [\mathbf{d} \cdot \mathbf{T}] + [\mathbf{e} \cdot \mathbf{PR}]$$

Dalla tesi estrapoliamo la descrizione dei singoli valori della formula presa in esame:

le prime tre componenti costituiscono il **punteggio base** e ognuna di esse è pesata con il proprio coefficiente. In particolare:

- **C**: rappresenta la componente di **coverage** e descrive il grado di copertura medio raggiunto dal tester nelle varie pagine visitate durante la sessione, calcolata in termini di widget interagiti. In particolare, C viene calcolata come:

  $$C = \frac{\sum_{\forall i \in P} pc_i}{|P|}$$

  dove al numeratore troviamo la sommatoria delle coverage (pc_i) raggiunte nelle singole pagine dell'insieme *P* e al denominatore la cardinalità dell'insieme stesso. La coverage relativa ad una singola pagina viene a sua volta calcolata con il seguente rapporto:

  $$pc_i = \frac{w_{hl,i}}{w_{tot,i}}$$

  dove $w_{hl,i}$ è il numero di widget diversi con cui il tester ha interagito all'interno dell'i-esima pagina del set *P*, mentre $w_{tot,i}$ è il numero totale di widget cliccabili presenti nella medesima pagina *i*.

- **EX**: rappresenta la componente **esplorativa** della sessione e si basa sul numero di pagine che il tester considerato ha visitato e

che non erano ancora state visitate da nessun altro fino a quel momento. La componente, inoltre, prende anche in considerazione i widget con cui il tester ha interagito per primo, anche all'interno di pagine già visitate in precedenza da almeno un collega. EX_{comp}, nel dettaglio, viene calcolata come:

$$EX = \frac{k}{b} \cdot \frac{p_{new}}{p_{tot}} + \frac{h}{b} \cdot \frac{hw_{new}}{hw_{tot}}, \qquad k + h = b;$$

dove il primo rapporto rappresenta la sottocomponente relativa alle nuove pagine scoperte (percentuale delle pagine nuove rispetto al totale delle pagine visitate durante la sessione), mentre il secondo quella relativa ai widget (percentuale degli *highlighted widget* nuovi rispetto al totale dei widget con cui il tester ha interagito nel corso della sessione). h e k rappresentano le percentuali di ogni sottocomponente rispetto al punteggio base totale.

- **EF**: rappresenta la componente di **efficienza** e serve sostanzialmente ad evitare che un tester cerchi di aumentare eccessivamente il proprio punteggio cliccando ripetutamente su widget già evidenziati in precedenza. Viene dunque calcolata come:

$$EF = \frac{w_{hl}}{w_{int}}$$

dove w_{hl} è il numero totale di *highlighted widget* della sessione, mentre w_{int} è il numero di interazioni effettuate sui widget (comprensivo quindi dei click effettuati su widget già evidenziati). Per la natura stessa di questa componente, se il test viene effettuato coscienziosamente, il suo valore sarà tendenzialmente molto vicino a 1.

4.2.2 Barra di progressione

Nella versione precedente é prevista una barra di progressione in alto alla finestra di testing che rendere graficamente accessibile il valore della coverage nella pagina corrente. Questo elemento, molto utile per il tester, subirá solamente un restyling grafico, ma svolge giá alla perfezione il suo compito in accordo con il principo di "Potenziamento". La barra di progressione é un espediente molto usato. Un esempio virtuoso é quello di Linkedn, famosissimo social network utilizzato per

creare la propria rete professionale. In questo contesto durante la creazione del proprio profilo viene visualizzata una barra di progressione che rendere consapevole l'utente della percentuale di elementi mancanti e di quelli invece presenti. Nel contesto del Gamification Service, svolge esattamente lo stesso compito per quello che riguarda i widget analizzati.

4.2.3 Stella delle nuove pagine

Anche in questo caso la componente subisce un cambio estetico, per una questione prettamente funzionale: essere piú visibile. La stessa aiuta a rendere consapevole l'utente delle proprie azioni e va valorizzata in questo contensto.

4.2.4 Easter Egg

Gli Easter Egg sono una componente precedentemente inserita per simulare la condizione, molto frequente, per cui un collegamento ipertestuale punti a una risorsa non piú esistente. All'inizio di ogni sessione di test vengono estratte a sorte alcuni rami del modello ad albero dell'interfaccia che simulano la presenza di questi link corrotti attraverso la visualizzazione di figure geometriche colorate all'interno della pagina.

4.2.5 Classifica

Utilizzando il punteggio precedentemente ottenuto é possibile ottenere una classifica dei migliori tester basata sui risultati ottenuti. Questa é accessibile sempre dalla finestra di riepilogo e ha lo scopo di inserire all'interno del plugin sviluppato una componente competitiva che potrebbe stimulare l'utente a dare sempre il meglio in ogni sessione.

4.3 Progetto

4.3.1 Login

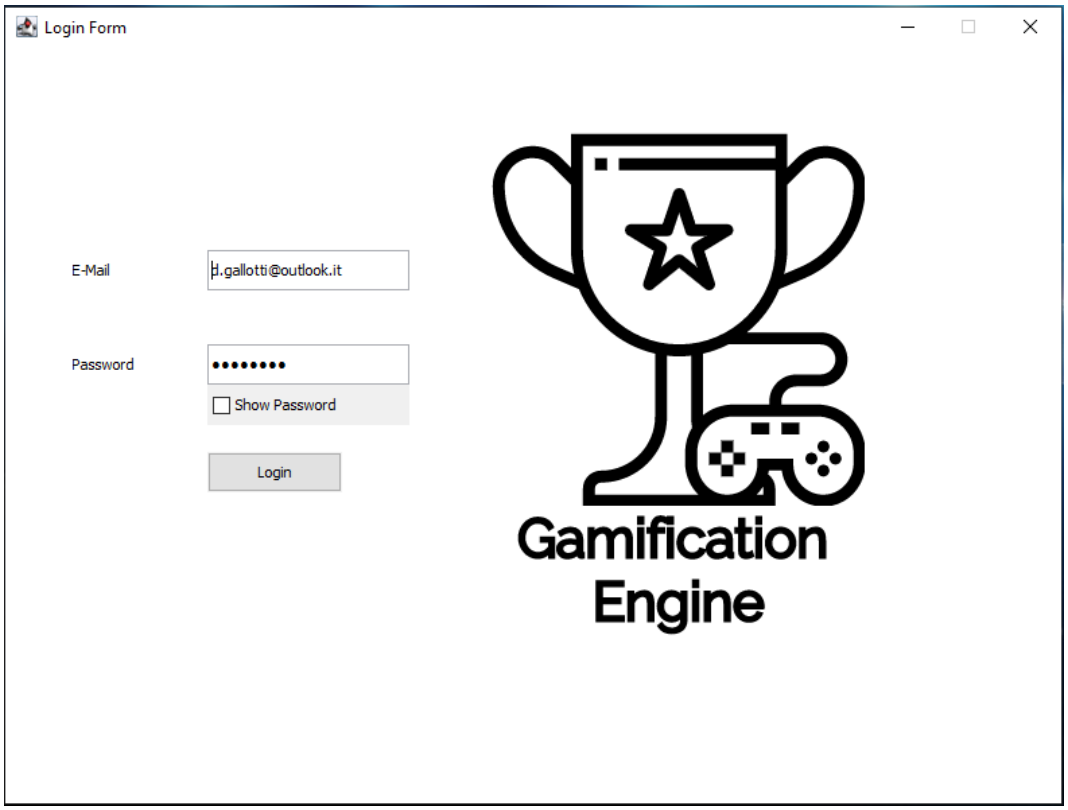


Figura 4.2. Schermata di login

La prima componente implementata é l'interfaccia che permette di fare il login al servizio. Oltre a permettere nel modo piú semplice possibile l'autenticazione dell'utente, utilizza essa stessa una dinamica tipica in contesti di Gamificazione: il naturale senso di **possesso** e il desiderio di **appartenenza**. L'autenticazione di un utente infatti, oltre a rendere sicura l'estrazione di informazioni personali, é il modo piú semplice per dare la sensazione all'utente di appartenere a qualcosa o di possederla. Non bisogna trascurare infatti il concetto che le stesse interfacce a volte possono presentare elemnti di gamificazione intrinsechi che vengono spesso pensati e concettualizzati nei processi di UI/UX design per migliorare l'interazione con il software

4.3.2 Profilo

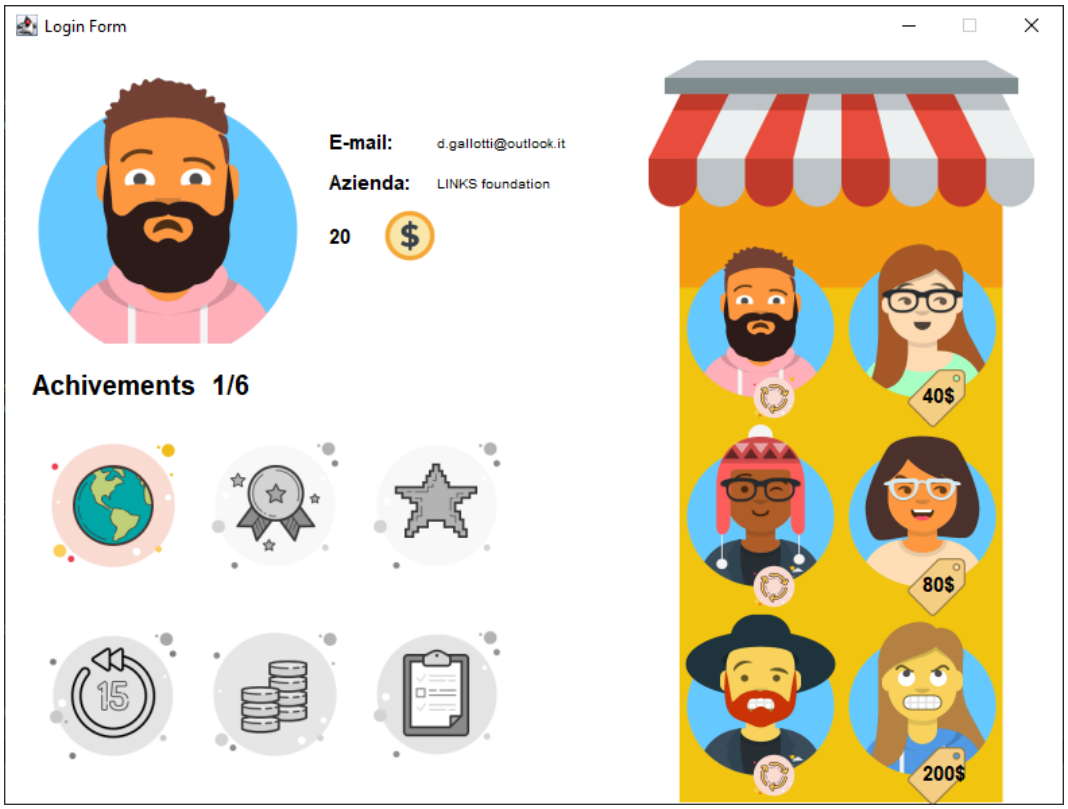


Figura 4.3. Scout schema

A seguire troviamo il profilo dell'utente, il vero cuore del sistema di gamificazione progettato. Contiene il riepilogo delle informazioni piú importanti:

1. Username
2. E-mail
3. Azienda di appartenenza
4. Moneta virtuale
5. Record personale (di cui parleremo in una nelle sezioni seguenti)
6. Avatar personalizzabile

7. Obiettivi sbloccati e da sbloccare
8. Negozio per nuovi avatar

Le dinamiche utilizzate in questo contesto sono quelle di socializzazione che peró approfondiremo meglio nel dettaglio nella sezione relativa alla classifica. Tuttavia come precedentemente mostrato nella sezione realtiva all'Octalysis, il profilo utente vuole andare a coprire tutte quelle emozioni relative all'osservazione di cosa si sta facendo e di cosa si é fatto. Per questo motivo, nonostante sia presente una versione abbastanza rudimentale del profilo utente, é importante, nel caso di sviluppi futuri andare a aggiornare e perfezionare il numero e la qualitá delle informazioni presenti anche in accordo a quanto detto nel capitolo di analisi degli otto principi di gamificazione utilizzati in questo progetto.

4.3.3 Avatar

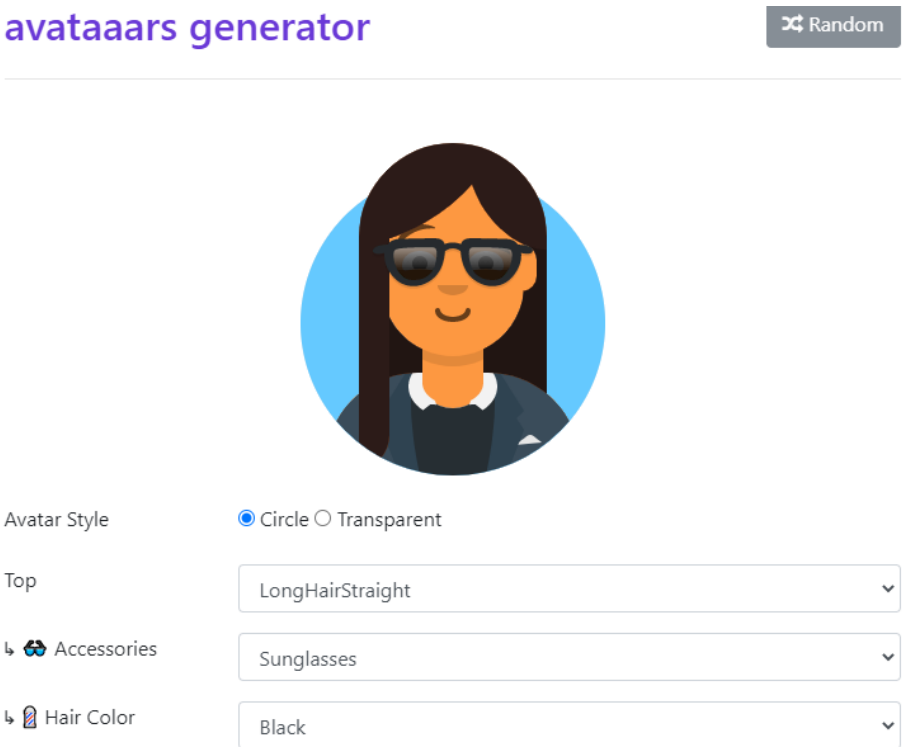


Figura 4.4. Avataaars Generator

L'avatar é la rappresentazione grafica dell'identitá di una persona all'interno di ambienti virtuali. Questo escamotage é da sempre utilizzato

per favorire la personalizzazione del proprio profilo nel mondo di internet. I benefici risultanti da questo espediente grafico hanno orgine nel complicato rapporto tra la realtá e il virutale che negli ultimi 20 anni ha completamente ridimensionato ogni aspetto della nostra esistenza. Anche nell'esplorazione di ambienti virutali ognuno di noi sente il bisogno di appartenere, in qualche modo, anche fisicamente all'ecosistema tecnologico che sta utilizzando. Gli strumenti informatici, soprattutto quelli online, non sono esclusivamente un potenziamento esterno delle capacitá umane, ma vengono sempre con piú enfasi vissute come ambienti simili a quelli reali, in cui un utente si trova maggiormente a proprio agio solo utilizzando le stesse "regole". Nella sua versione definitiva l'avatar dovrebbe aver un editor in cui personanlizzare ogni singolo aspetto del personaggio virtuale (colore dei occhi, capelli, accessori ecc..). In questa versione, dato che si tratta di un prototipo sono stati preselezionati sei avatar creati dal progetto, open source e open license "Avataaars Generator" su **GitHub** dell'utente *fangpenlin*. L'idea finale sarebbe di integrare completamente l' editor all'interno del contesto Scout e aggiungere un prezzo, in base alla desiderabilitá attesa dei singoli accessori per ogni elemento che si puó utilizzare per personalzzare l'avatar. In questo modo il principio della *Realizzazione* si realizza nella sua completezza, che oltre a dare una migliore rappresentazione virtuale del tester, concede un ulteriore stimolo a procedere con le sue attivitá di testing.

4.3.4 Obiettivi

Prima di parlare degli obiettivi conviene analizzare alcune metriche oggettive che vengono utilizzate all'interno del plugin sviluppato. Le metriche in questione sono:

1. **Coverage**: il rapporto tra il numero di elementi evidenziati nella pagine e il numero totale di elementi presenti. Nella valutazione della coverage all'interno della sessione di testing si valuterá la coverage media di tutte le pagine.

2. **Issues**: il numero di elementi evidenziati che presentato dei problemi grafici e che vanno mandati a revisione.

3. **Tempo di sessione**: il numero di secondi, minuti e ore in cui si é svolta la sessione.

La componente degli obiettivi é pensata per adiempiere a pieno al principio di *Potenziamento*. Definisce delle linee guida da seguire per svolgere un proprio lavoro al meglio. Per questo motivo, oltre ad avere una valenza funzionale in quanto stabilisce delle metriche nel modo piú oggettivo possibile, ha lo scopo di gratificare anche il lavoro svolto dal tester, premiandolo con dei riconoscimenti evidenti sui risultati raggiunti. Nel profilo é possibile visualizzare questa informazione, sia per dare un senso di progressione al lavoro svolto, sia per chiarire quali sono gli obiettivi su cui si é piú in deficit durante la sessione di testing. Per la versione prototipale sono stati pensati sei obiettivi (eventualmente da integrare con altri in una versione definitiva):

1. **Prime scoperte**: si ottiene raggiungendo una coverage media del 50%

2. **Esploratore esperto**: si ottiene raggiungendo una coverage media del 70%. L'idea dietro questo achivement é quella di spronare il tester a ottenere una coverage più alta possibile in modo da individuare un numero di errori ragionevolmente significativi per evitare malfunzionamenti.

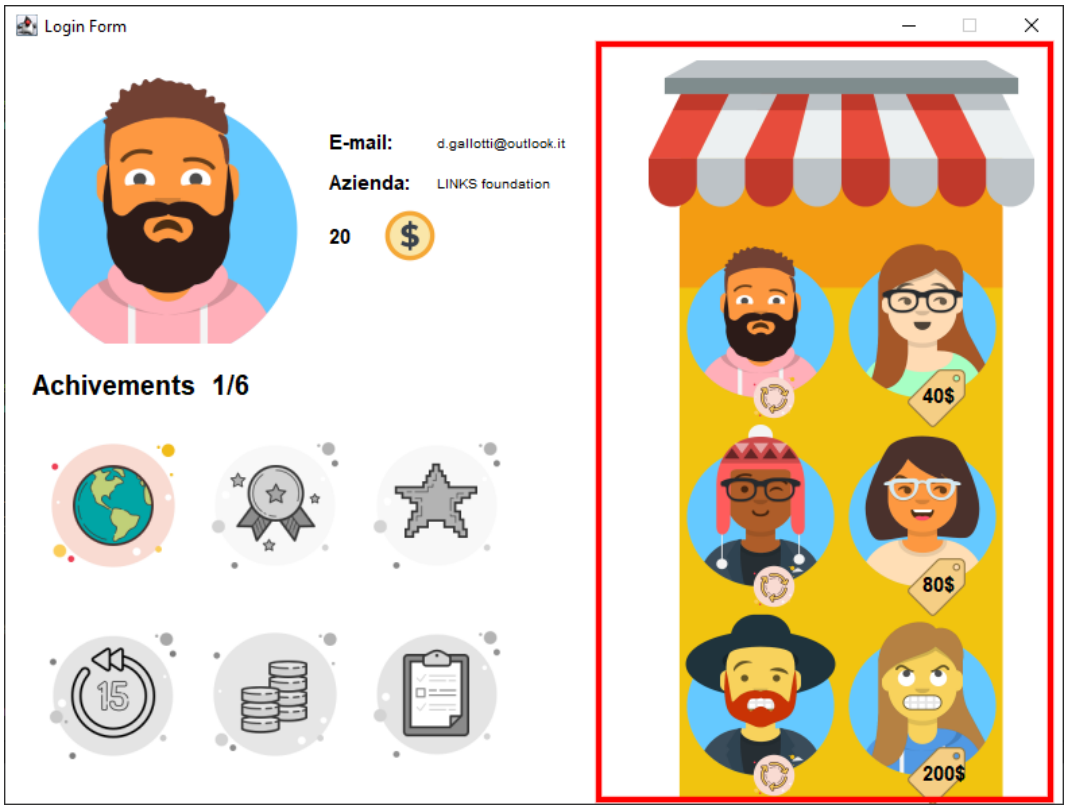
3. **Nessun segreto**: si ottiene scoprendo il 100% degli easter eggs presenti all'interno di una sessione di testing. Server per motivare l'utente a controllare nel dettaglio le singole sezioni dell'interfaccia, senza trascurare nulla. Se vogliamo può essere visto come l'obiettivo più completo riguardante la metrica della coverage.

4. **Pazienza:** si ottiene svolgendo una sessione di testing della durata uguale o maggiore di 15 minuti. Ha lo scopo di mostrare che la velocità non è l'unico fattore importante quando si parla di testing, ma piuttosto è necessario spesso prendersi il tempo necessario e agire con calma, senza aver fretta di concludere un lavoro.

5. **Zio Paperone:** si ottiene raggiungendo una quantità di monete uguale o maggiore di 2000 in un'unica sessione. Sebbene la valuta virtuale sia una metrica aggiunta e non derivata, ovvero viene innestata nel sistema e non viene calcolata dal sistema, potrebbe apprentemente riusltare un fattore superfluo per quello che riguarda una sessione di testing, tuttavia vuole rappresentare

6. **Repellente:** si ottiene segnalando 10 o più issues in un'unica sessione. L'idea dietro questo obiettivo è di quantificare e gratificare in maniera numerica il vero obiettivo di una sessione di testing che è quello di trovare dei problemi nell'interfaccia grafica.

4.3.5 Negozio



Un'altra componente fondamentale è il negozio di avatar: in questa sezione l'utente può spendere il frutto del suo lavoro, per personalizzare come più preferisce il proprio avatar. Questa personalizzazione andrà a influenzare principalmente il modo in cui gli altri utenti vedono il profilo del tester nella classifica aziendale. Nella sua versione prototipale è possibile acquistare solo 6 avatar, il cui prezzo è stato scelto in base alla quantità di dettagli e di caratteristiche peculiari dell'avatar: più l'avatar ha delle caratteristiche ben precise maggiore sarà il suo costo (e presumibilmente anche il desiderio dell'utente di possederlo), nella versione finale andrebbero aggiunti invece degli avatar interi le singole componenti e i singoli accessori, in modo che l'utente possa

facilmente scegliere e costruire l'avatar virtuale che più lo rappresenti. All'interno di questo negozio sarebbe possibile in accordo con il principio di *Scarcity and Impatience* aggiungere in alcuni periodo anche degli accessori **LIMITED EDITION** per invogliare maggiormente l'acquirente durante le sue sessioni di testing.

4.3.6 Punteggio

Il punteggio è l'unica componente non rappresentata da una finestra dell'interfaccia grafica. Viene visualizzato alla fine di una sessione di testing e rappresenta in un valore compreso tra 0 e 100 l'efficacia di quella specifica sessione di testing basandosi sulle varie metriche precedentemente descritte. Il suo valore viene convertito in valuta virtuale ottenuta dall'utente. Maggiore sarà il punteggio, prima l'utente potrà personalizzare e migliorare il suo avatar come meglio preferisce.

4.3.7 Classifica

La classifica non subisce grosse modifiche, ma viene implementata a livello globale e attivata solo in specifici intervalli di tempo scelti da Project Manager, in modo da non rendere frustrante il lavoro e in modo che ognuno periodicamente sia messo nelle stesse opportunità di arrivare primo. In accordo con gli obiettivi aziendali e gli incentivi personali sul lavoro del dipendente è possibile utilizzare questa classifica anche per premi aziendali se la direzione o il PM lo ritengano opportuno. L'utilizzo di un sistema distribuito in modo da poter gestire le classifiche in ambiente sia aziendale sia domestico, soprattutto in questo periodo storico in cui lo smart working sta ottenendo sempre maggiore consenso, lo rendendo uno strumento facile, veloce ed efficace da usare secondo le varie necessità.

4.3.8 Feedback grafici

Un buon feedback grafico aiuta l'utente a muoversi con dimestichezza all'interno dell'applicazione, ecco alcuni esempi di feedback relative alle possibili azioni che l'utente puó compiere.

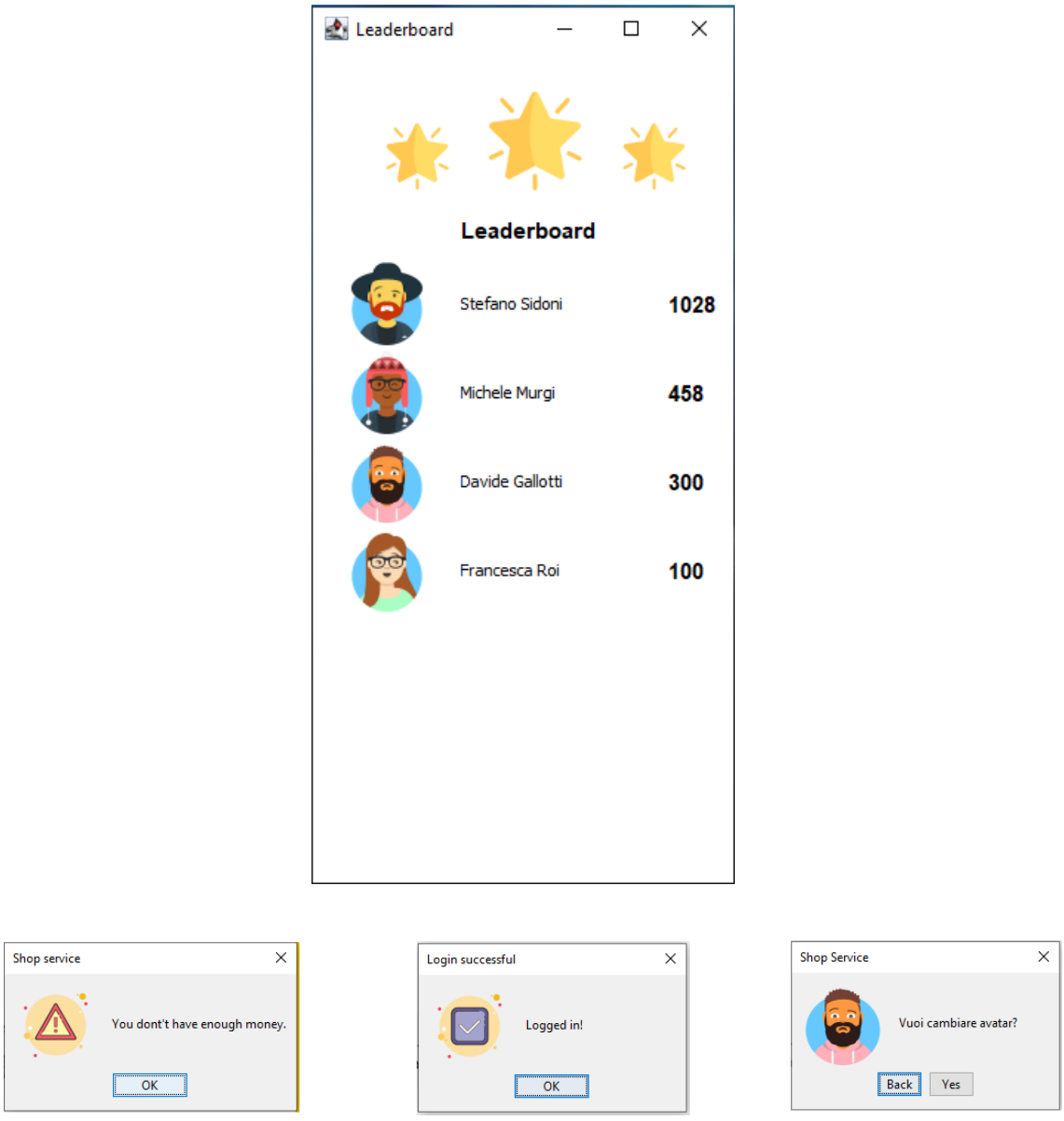


Figura 4.5. Feedback grafici

4.4 Octalysis del Gamification Engine

Utilizzando come metro di riferimento l'Octalysis si é voluto rappresentare con un grafico le caratteristiche principali di questo sistema di gamification, in modo da evidenziare in maniera visiva tutte le sue caratteristiche nel complesso.

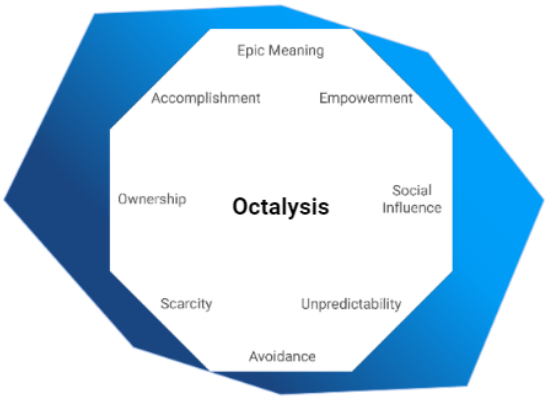


Figura 4.6. Gamification Engine - Octalysis

1. **Chiamata** [1/10]: non é presente nessun meccanismo che punti su questa emozione
2. **Potenziamento e Creativitá** [5/10]: dato dalla possibilitá di acquistare avatar personalizzati nel negozio
3. **Influenza sociale** [7/10]: presenza della classifica aziendale basata sul punteggio ottenuto dalle varie sessioni di testing
4. **Imprevedibilitá**[6/10]: presenza degli easter eggs che compaino casualmente all'interno della pagina
5. **Fuga** [1/10]: nessun elemente presente
6. **Scarsitá** [2/10]: possibilitá di inserire **limited edition** nel negozio virtuale
7. **Scarsitá** [8/10]:profilo utente, statistiche, collezioni
8. **Realizzazione** [7/10]: Obiettivi

Di seguito un confronto con gli Octalysis rispettivi di Facebook e Linkedin

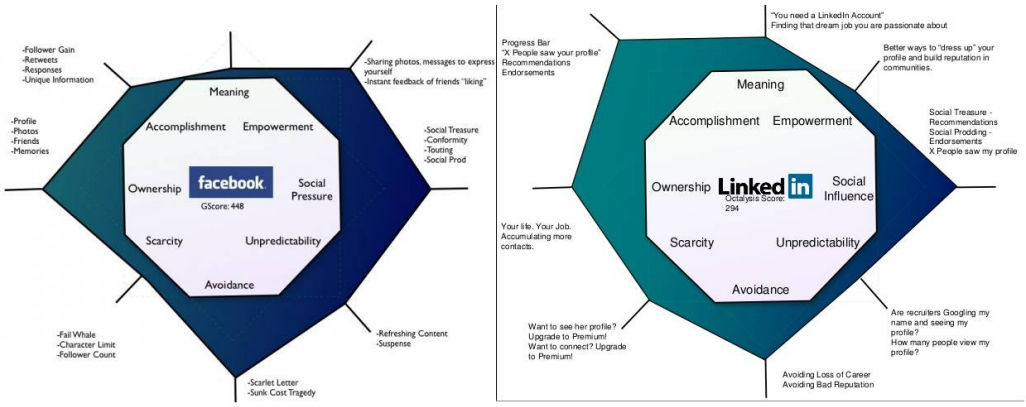


Figura 4.7. Octalysis di Facebook e Linked in

4.5 Simulazione di una sessione di testing

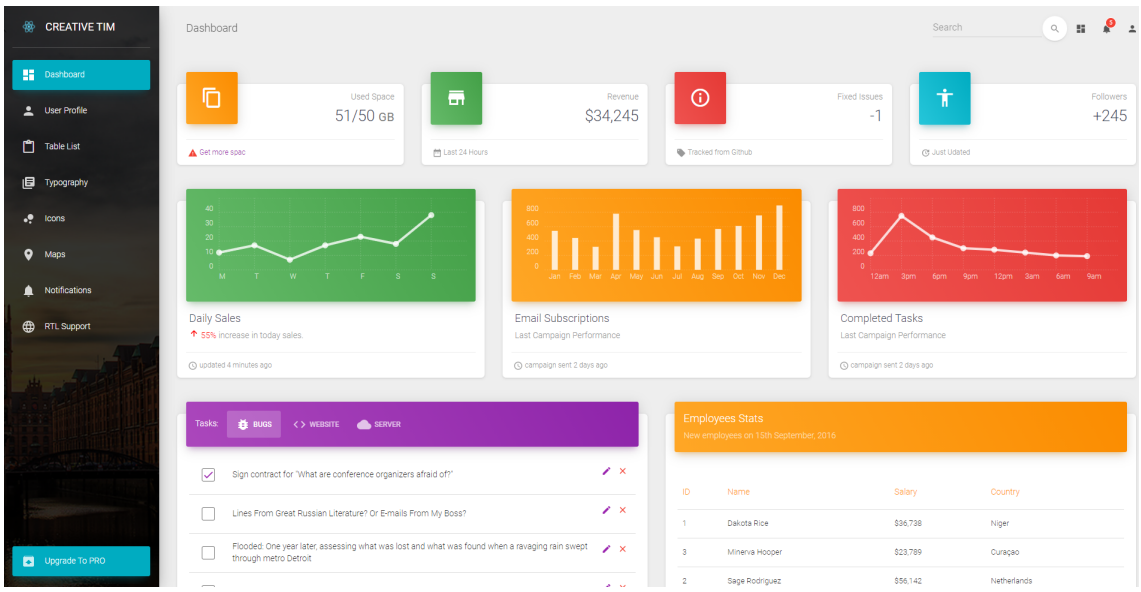


Figura 4.8. UML component diagram

Per comprendere a pieno l'utilizzo del sistema sviluppato di seguito

viene mostrata un'esempio di sessione di testing. Per la sessione in corso viene utilizzato un protitipo di un applicativo web partendo da un template in React. Il sito viene hostato su un server in locale.

1. All'avvio viene fatto il login inserendo le credenziali.
2. Parte la sessione di test visualizzando la pagine Home del progetto.
3. La barra di progresso é completamente rossa a indicare che non si é ancora evidenziato nessun widget.
4. La schermata composta da vari elementi é divisa in sezioni, ognuna rappresentante un aglomerato di dati.
5. Si evidenza un errore nel primo riguardo il cui é presente un numero che supera il limite massimo consentito di spazio utilizzabile: 51/50 GB. Probabilmente un errore di calcolo lato backend. Si evidenza il widget in rosso e si va avanti con la verifica.
6. Il secondo riguadro non presenta nessuna criticitá, si evidenzia di verde.
7. Il terzo riguardo presenta un -1 dove il minimo valore accettato é 0. Sará una variabile inizializzata e mai usata? Viene aggiunta la segnalazione e si procede con il prossimo widget.

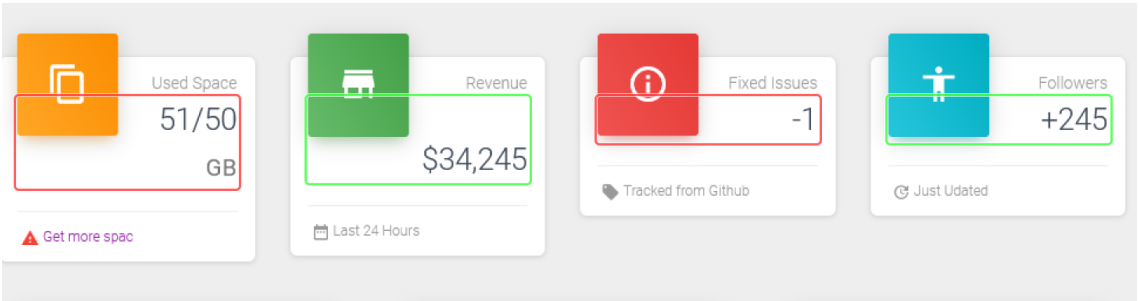


Figura 4.9. Controllo dei widget

8. La percentuale positiva di 55% é colorata di rosso: da specifica deve essere verde. Altro errore aggiunti.
9. Non vengono individauti uteriori errori all'intenro della pagina.

10. Il tester a questo punto passa alla sezione profilo.

11. La stella della nuova pagina compare in altro a sinitra.

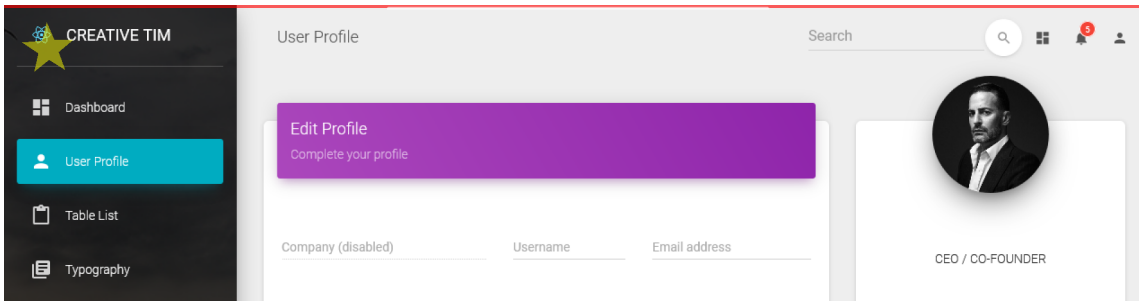


Figura 4.10. Stella nuova pagina

12. E’ presente anche un easter eggs.

13. Si evidenziano i widget principali di verde: non sembrano esserci problemi.

14. La sezione tablelist é ancora in fase di sviluppo.

15. Il tester decide di chiudere la sessione ottenendo un punteggio di 50 che vengono convertite in monete.

16. Avendo ottenuno una coverage media del 50% ottiene l’achivement corrispondente.

17. L’utente aveva 100 monete precedenti e con le sue 150 moente ottenute decide di acquistare un nuovo avatar.

Capitolo 5

Implementazione tecnica

5.1 Interazione tra gli elementi del Gamification Engine

Il sistema preeedentemente descritto viene implementato attraverso tre pricinpali componenti a loro volta suddivise in sotto componenti come illustrato nel grafico seguente:

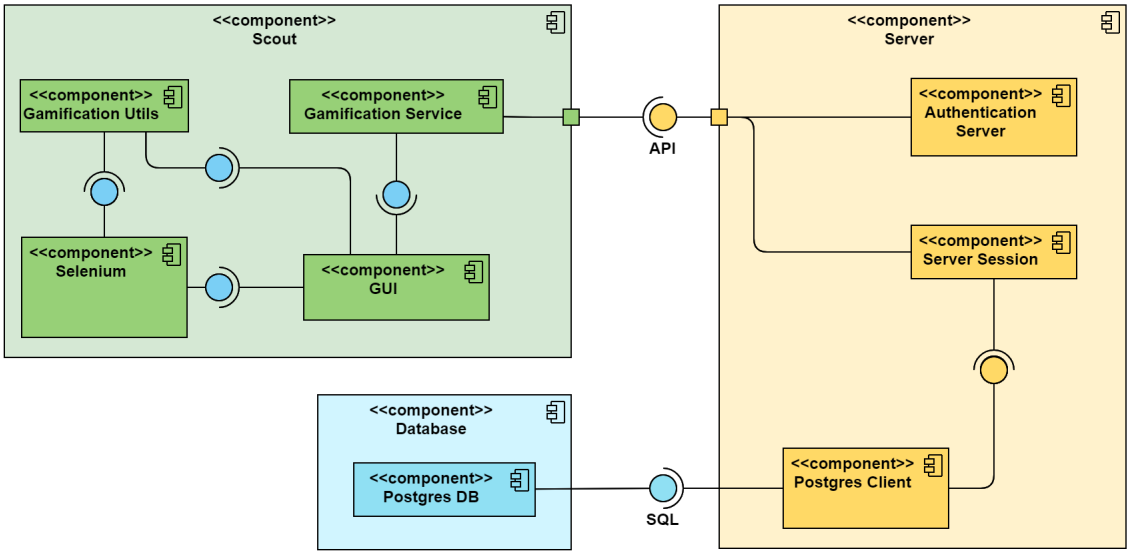


Figura 5.1. UML component diagram

Le tre componenti principali sono:

1. **Scout**: software di partenza per il testing delle applicazioni. A sua volta é composto da 4 sotto componenti. Il **Gamification Service** si occupa di comunicare con il servizio API implementando tutte le funzioni che consentono di trasformare tutte le informazioni contenute nel database ottenute dal server in dati aggregati servibili in Java. (trattato nel capitolo precedente). La **GUI** contiene tutte le interfacce per interagire con il servizio. Il plugin **Selenium** utilizzato per interagire e testare la web app. Le **Gamification Utils** sono una serie di funzioni che analizzano i dati e implementano la logica delle scelte progettuali descritte.

2. **Server API**: un servizio d'interfaccia con API RESTful programmato in javascript e esposto tramite node.js. Una componente si occupa di autenticare l'utente, attualmente non rispetta tutti i criteri di sicurezza informatica che andrebbero implementati per una versione definitiva. La componente principale é quella relativa alla sessione che interagento con il **Postgres Client** chiede al database tutte le informazioni desiderate dall'utente.

3. **Database Postgres**: database relazionale in cui sono contenute tutte le informazioni degli utenti e del Gamification Engine.

In questo capitolo tratteremo tutti i dettagli tenici relativi alle singole componenti, in modo da rendere il sistema sviluppato (in fase prototipale) utilizzabile per il futuro per ulteriori ricerche o applicazioni. L'obiettivo, infatti, come fatto in questa tesi é quello di andare a costruire i vari mattoncini che possono migliorare sempre di piú il prodotto sia a livello tecnico che concettuale. Le sezioni che seguiranno, per questo motivo, vanno viste come un modo non solo per descrivere quanto fatto in questo lavoro, ma anche come fare per interagire e modificare ogni singola componente del sistema. C'é da aggiungere inoltre che la maggior parte delle scelte attuali non rendono il sistema utilizzabile in una fase produttiva, ma sono da intendersi come prototipi utili per testare le singole dinamiche e meccaniche per il servizio di gamificazione precedentemente concettualizzato e descritto.

5.2 Database PostgreSQL

Ogni informazione precedentemente contenuta in un file .txt é stata trasposta in un database relazionale implementato con PostgreSQL. Di seguito un diagramma ER che descrive la struttura progettuale del database.

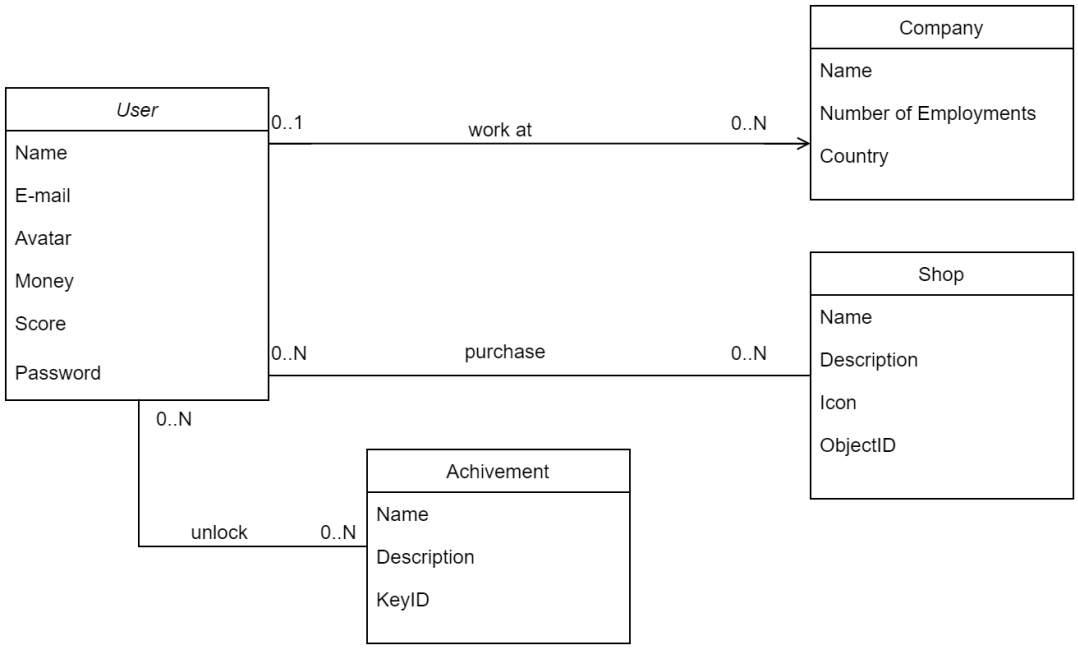


Figura 5.2. Modello E-R

5.2.1 Tabelle derivate

Dal diagramma si estrapolano le seguenti tabelle:

1. Contiene la lista degli utenti registrati al servizio di gamificazione
   User (**email**, name, avatar, money, score, password, company)

2. Lista degli obiettivi sbloccabili
   Achivement (**keyID**, name, description)

3. Insieme degli oggetti virtuali acquistabili
   Shop (**objectID**, price, avatar)

4. Lista delle Aziende che utilizzano il servizio
   Company (**name**, **country**,numberOfEmployeements)

5. Relazioni tra gli utenti e i corrispettivi obiettivi sbloccati
   Unlock: (**email**, **keyID**)

6. Contiene la lista di tutti gli acquisti virtuali
   Purchase (**email**,**objectID**)

e la seguente vista:

1. **Classifica**: essendo un'informazione intrinseca del database non necessitá di una tabella ad hoc, ma puó essere generata a partire da una semplice query sui dati:

```
SELECT email,avatar,points
FROM User
ORDER BY points DESC
LIMIT 10;
```

5.3 Documentazione del servizio API per il Gamification Engine

Per comunicare con il database é stato implementato un servizio API RESTful in grado di fornire delle funzioni d'interfaccia al database. In questo modo i dati oltre ad essere scollegati dal plugin sviluppato per Scout, sono anche utilizzabili logicamente per compiere delle azioni all'interno del contesto di gamificazione creato per il tester. Di seguito un elenco delle varie chiamate e del loro utilizzo.

5.3.1 LOGIN

POST
End point: /login
Descrizione: Questa API permette di verificare che le credenziali utente siano corrette e ritorna le informazioni relative al profilo

Data:

```
{
    "email": "name@domain.com",
    "password" : "sTrongPassword"
}
```

Risultato:

```
{
    "code": 200,
    "avatar":"avatars4.png",
    "money" : 20,
    "email" : "d.gallotti@outlook.it",
    "username":"Davide Gallotti"
}


\textbf{Errori:} \begin{lstlisting}[language=json,
   firstnumber=1]
{
    "code": 400,
    "desription": "Account not found."
}
```

```
{
    "code": 401,
    "desription": "Authentication Error"
}
```

5.3.2 SHOP

GET

End point: /shop

Descrizione: Questa API permette di ottenere tutti gli oggetti acquistabili nello shop

Risultato:

```
{
    "code": 200,
    "shop": [
        {
            "object_id": "0",
            "name": "avatars1",
            "price": "20"
        },
        {
            "object_id": "1",
            "name": "avatars2",
            "price": "40"
        },
        ...
    ]
}
```

5.3.3 RANK

GET

End point: /get-leaderboard

Descrizione: Questa API permette di ottenere la classifica degli utenti con il punteggio piú alto

Risultato:

```
{
    "code": 200,
    "rank": [
        {
            "username": "Stefano Sidoni",
            "avatar": "avatars4.png",
            "points": "1028"
        },
        {
            "username": "Michele Murgi",
            "avatar": "avatars2.png",
            "points": "458"
        },
        ...
    ]
}
```

5.3.4 PURCHASE

GET

End point: /purchase

Descrizione: Questa API permette di aggiornare le informazioni riguardo l'acquisto di oggetti nel negozio all'interno dell'applicativo.

Parametri:

```
- email : utilizzata dall'utente in fase di registrazione
- objectID (preso dall'objectID dell'API shop)
```

Risultato:

```
{
    "code": 200,
    "description": "Object correctly purchased."
}
```

Errore:

```
{
    "code": 400,
    "description": "Purchase error"
}
```

5.3.5 PURCHASED

GET

End point: /purchased

Descrizione: Questa API permette di ottenere tutti gli acquisti di un determinato utente.

Parametri:

```
- email : utilizzata dall'utente in fase di registrazione
```

Risultato:

```
{
    "code": 200,
    "shopped": [
        {
            "email": "d.gallotti@outlook.it",
            "objectID": "0"
        },
        ...
    ]
}
```

Errore:

```
{
    "code": 400,
    "description": "Email not found"
}
```

5.3.6 ACHIVEMENTS

GET

End point: /unlock

Descrizione: Questa API permette di recuperare tutte le informazioni sugli achivements ottenibili.

Risultato:

```
{
  "code": 200,
  "achivements": [
    {
      "name": "Prime scoperte",
      "description": "Si sblocca raggiungendo una coverage
   del 20%",
      "keyID": "1"
    },
    ...
  ]
}
```

5.3.7 UNLOCK ACHIVEMENT

GET

End point: /achivements

Descrizione: Questa API permette di registrare l'ottenimento di un'achivement per l'utente desiderato

Parametri:

```
- email : utilizzata dall'utente in fase di registrazione
- keyID : identificativo numerico dell'achivement
```

Risultato:

```
{
  "code": 200,
  "achivements": [
    {
      "name": "Explorer",
      "description": "Unlock with a coverage almost of 40%",
      "keyID": "1"
    },
    ...
  ]
}
```

Errore:

```
{
  "code": 400,
  "description": "No email found"
}
```

5.3.8 USER ACHIVEMENTS

GET

End point: /obtained-user

Descrizione: Questa API ritorna un array di achivements ottenuto dall'utente desiderato.

Parametri:

```
- email : utilizzata dall'utente in fase di registrazione
```

Risultato:

```
{
    "code": 200,
    "achivements": [
        {
            "email": "d.gallotti@outlook.it",
            "keyID": "1"
        }
    ]
}
```

Errore:

```
{
  "code": 400,
  "description": "No email found"
}
```

5.3.9 UPDATE AVATAR

GET

End point: /update-avatar

Descrizione: Questa API aggiorna l'avatar selezionato dall'utente.

Parametri:

```
- email : utilizzata dall'utente in fase di registrazione
```

Risultato:

```
{
    "code": 200,
    "description": "Avatar changed."
}
```

Errore:

```
{
  "code": 400,
  "description": "No email found"
}
```

Capitolo 6

Sperimentazione del plugin

6.1 Validazione

6.1.1 Preparazione

Dopo aver concluso lo sviluppo dell'applicativo si vuole procedere con una sessione di validazione preliminare in modo da valutare e quantificare la qualitá del lavoro svolto. Dato il periodo in cui i test si sono svolti risulta particolarmente complesso riuscire a riportare delle metriche statisticamente valide dato il piccolo campione di persone a cui é stato sottoposto l'esperimento. Sono state scelte 10 persone con differente livello di background rispetto alle tematiche del testing grafico di applicazioni web e con diverso livello di esperienza nel mondo lavorativo. Sono stati preparati due progetti fittizi con degli errori preconfezionati in modo da essere facilmente riscontrabili nelle valutazioni del punteggio ottenuto dai tester. Ognuno di essi ha ricevuto un profilo presistente con delle credenziali fittizie (rappresentanti quelle aziendali). Ai tester é stata inoltre fornita una postazione con le stesse condizioni iniziali per svolgere l'esperimento e oguno ha potuto chiedere qualsiasi cosa per risolvere i dubbi che aveva in ogni momento dell'esperimento. I test si sono svolti i giorni differenti, ma

6.1.2 Esperimento

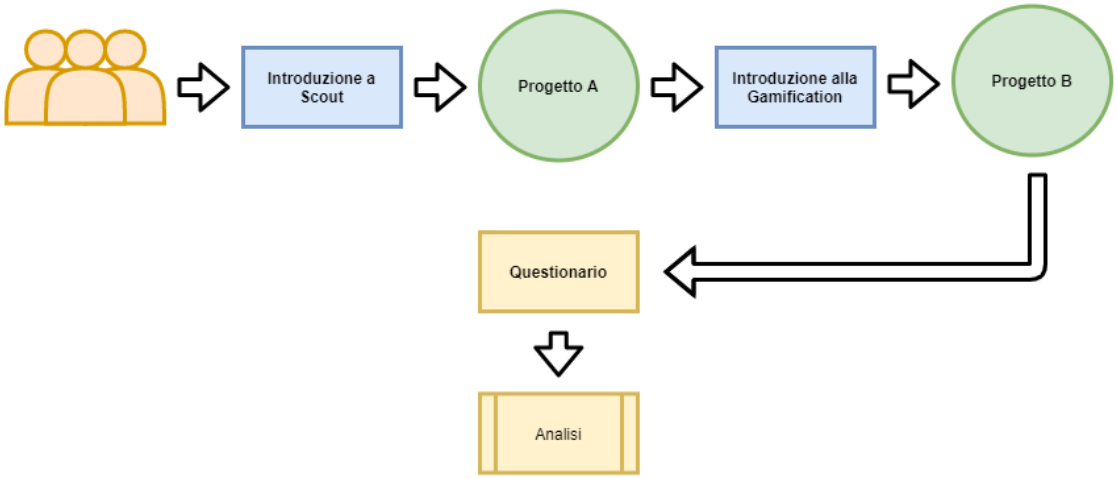


Figura 6.1. Procedura esperimento di validazione

I volontari hanno ricevuto una breve introduzione al concetto di testing (a prescindere dalle condizioni di partenza) e all'ambiente di lavoro basato su Scout. Subito dopo ad ognuno di essi é stato presentato il primo progetto con il plugin di Gamification disabilitato, a seguire c'é stata una breve introduzione sulla Gamification e sugli elementi presenti all'interno del software che questa volta sono stati attivati. A questo punto si sono occupati di testare il secondo progetto. L'obiettivo é stato quello di valutare se ci fossero differenze e nel caso quali, utilizzando il sistema descritto nei capitoli precedenti o la versione base di Scout.

6.1.3 Questionario

Dopo l'esperimento tutti i volontari sono stati sottoposti ad un test con il quale é stato possibile analizzare i risultati ottenuti. Di seugito la tabella con tutte le domande presenti:

Tabella 6.1: Domande del Questionario

ID	Domanda (Tipo)
1.1	Età (Scelta Multipla)
1.2	Hai mai avuto esperienze professionali con Java, come studente o lavoratore? (Scelta Multipla)
1.3	Quanti anni di esperienza hai con la programmazione Web? (Scelta Multipla)
1.4	Quali tra questi linguaggi hai mai usato per la programmazione Web? (Checkbox)
1.5	Hai mai avuto esperienze di testing di applicazioni o siti Web? (Scelta Multipla)
1.6	Quali tool hai utilizzato per fare testing di applicazioni Web? (Aperta)
2.1	Hai compreso il modo di utilizzare Scout e il suo contesto di utilizzo? (Likert)
2.2	Hai trovato la Barra di Progresso utile a mostrare i tuoi progressi nel testing di una pagina? (Likert)
2.3	La Classifica finale ha stimolato il tuo senso di sfida verso i tester precedenti? (Likert)
2.4	Conoscere l'esistenza di un Punteggio finale ti ha spinto a migliorare la tua performance? (Likert)
2.5	La Stella che contrassegna le pagine scoperte ti ha incoraggiato ad esplorare più approfonditamente l'applicazione da testare? (Likert)
2.6	Gli Easter Egg indotti ti hanno stimolato ad interagire con il maggior numero possibile di widget all'interno dell'applicazione? (Likert)

Tabella 6.1: Domande del Questionario

ID	Domanda (Tipo)
2.7	Quali tra i seguenti elementi erano a te familiari prima dello svolgimento dell'esperimento? (Checkbox)
2.8	Quali, tra gli elementi selezionati nella risposta alla domanda precedente, sono stati utilizzati in maniera a te familiare? (Checkbox)
2.9	Quali tra i seguenti elementi ritieni essenziali in una sessione di testing? (Checkbox)
2.10	Hai trovato comprensibile l'interfaccia del Profilo?
2.11	Hai trovato semplice l'utilizzo del negozio?
2.12	Gli obiettivi ti hanno stimolato a essere piú attento durante la sessione di testing?
2.13	Il desiderio di acquistare un'avatar ti ha stimolato a ottenere un miglior punteggio?
2.14	Hai trovato comprensibile la Schermata di Riepilogo e facilmente decodificabili le informazioni presenti al suo interno? (Likert)
2.15	Trovi che il tool utilizzato sia facilmente integrabile in un ambiente di testing esistente (lavorativo o di studio)? (Likert)
2.16	La versione di Scout che utilizza la Gamification ha aumentato la tua consapevolezza riguardo alla tua performance rispetto a quella senza? (Likert)
2.17	Nel complesso, trovi migliore la versione di Scout che utilizza la Gamification rispetto a quella senza? (Likert)
2.18	Hai riscontrato dei problemi durante lo svolgimento dell'esperimento? Se sì, descrivili brevemente (Aperta)
2.19	Hai dei suggerimenti per migliorare il tool proposto? (Aperta)

6.2 Performance

6.2.1 Punteggio

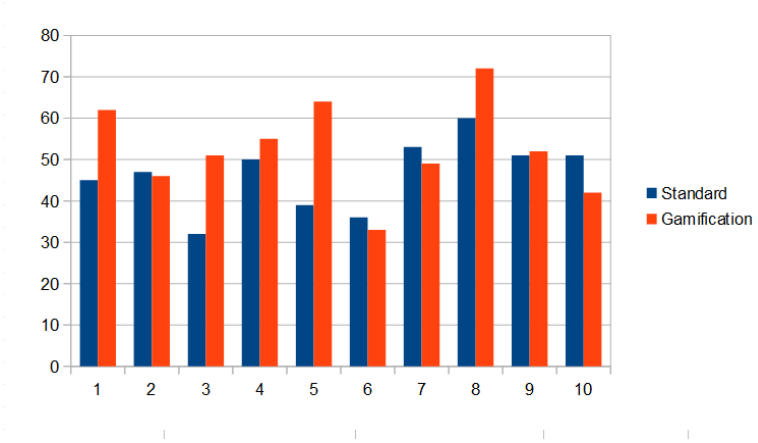


Figura 6.2. Punteggio ottenuto nelle due sessioni di Testing

7/10 partecipanti hanno migliorato il loro punteggio dopo la prima sessione di testing per un miglioramento medio di circa il 16%.

6.2.2 Issues trovati

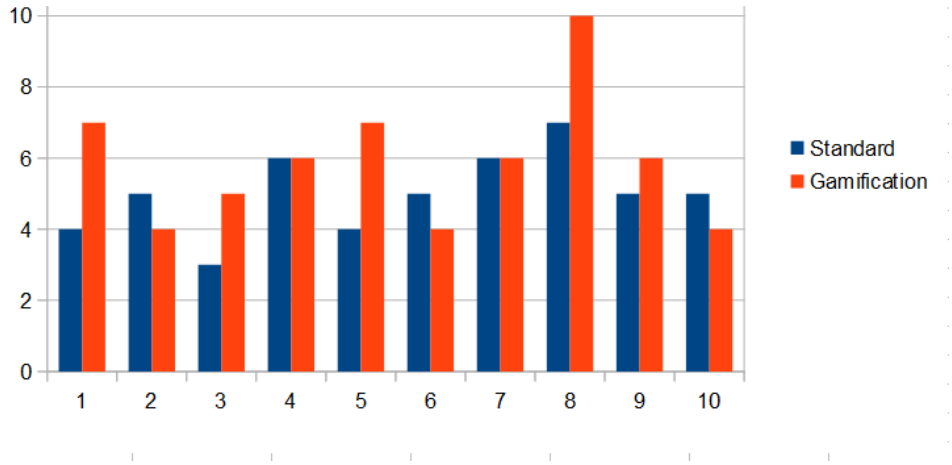


Figura 6.3. Issues trovati nelle due sessioni di testing

Altro risultato significativo é sul numero di bug riscontrati nella sessione di testing con gli elementi di gamification inseriti che é maggiore in media di circa il 21% rispetta alla sessione senza gamification. Nei casi in cui non si sono visionati miglioramenti significativi, la differenza tra i due approcci é molto simile, ma tutti

6.3 Componenti

6.3.1 Classifica

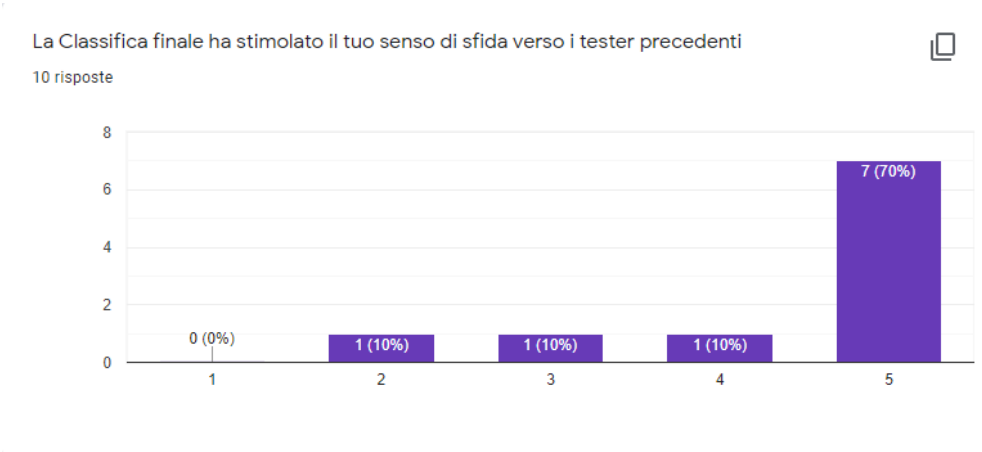


Figura 6.4. Valutazione della classifica

A parte in casi rari la classifica ha ricevuto il gradimento aspettato. Questo potrebbe dipendere principalmente dall'analisi fatta nel capitolo 3 della tassonomia di Bartle. Non tutti i giocatori infatti hanno le stesse predisposizioni e nessun sistema puó avere una validitá assoluta.

6.3.2 Profilo

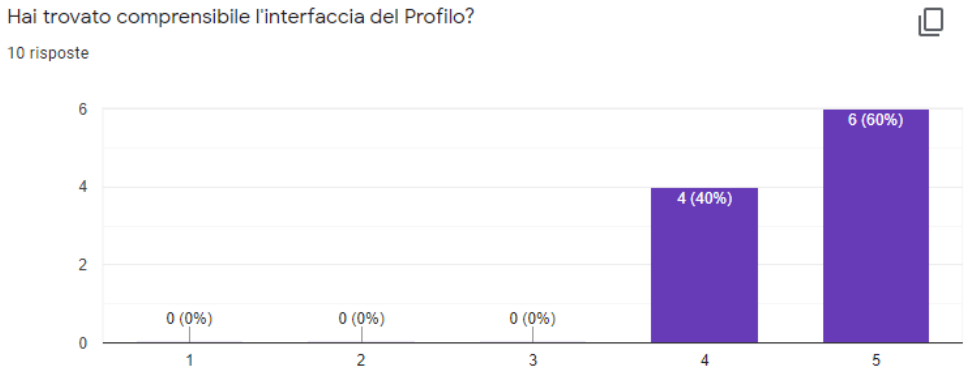


Figura 6.5. Valutazione del profilo

Le statistiche relative al gradimento della sezione "Profilio" evidenzia un ottimo riscontro su questa componente. Nonostante le migliorie proposte nella sezione di feedback, la totalitá dei tester si é mostrata soddisfatta dell'usabilitá dell'applicativo.

6.3.3 Obiettivi

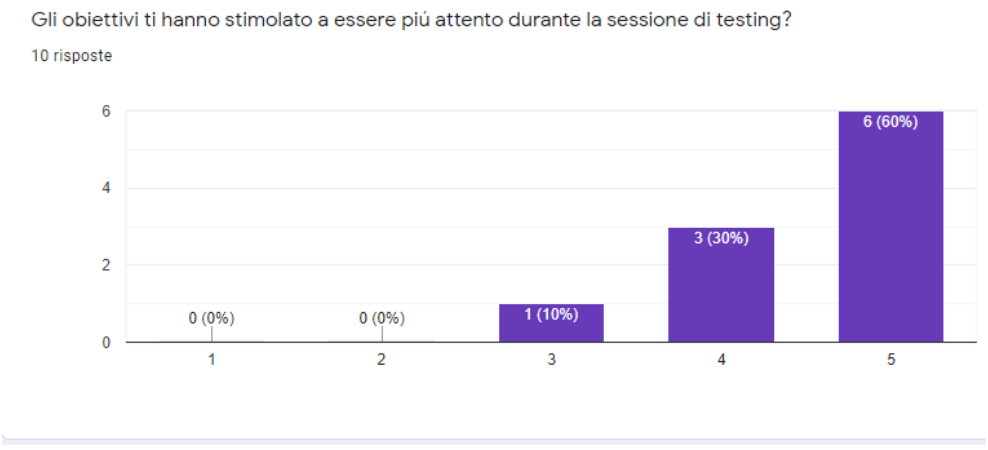


Figura 6.6. Gradimento obiettivi

6.3.4 Negozio

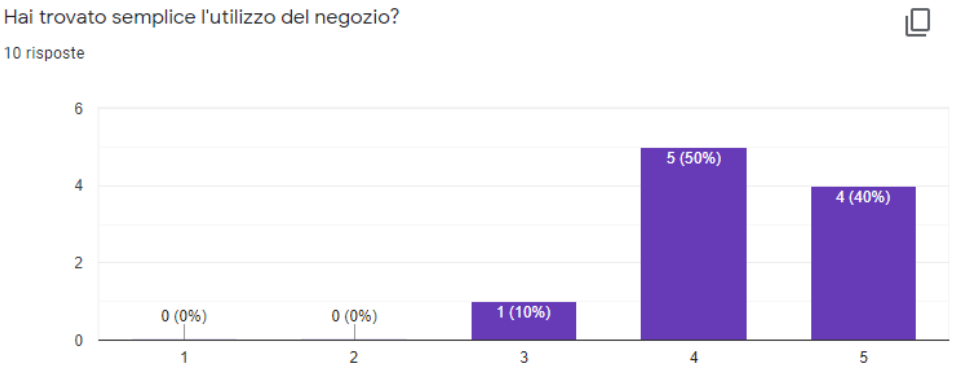


Figura 6.7. Utilizzo del negozio

Sia gli achivements che il negozio sono stati anche molto aprezzati, nella sezione aperta alcuni partecipanti hanno mostrato un interesse nella quantitá di personalizzazione degli avatar che in un futuro aggiornamento potrebbe essere estesa come descritto nella sezione 4 di questa tesi.

6.4 Usabilitá

6.4.1 Utilizzo in contesti lavorativi

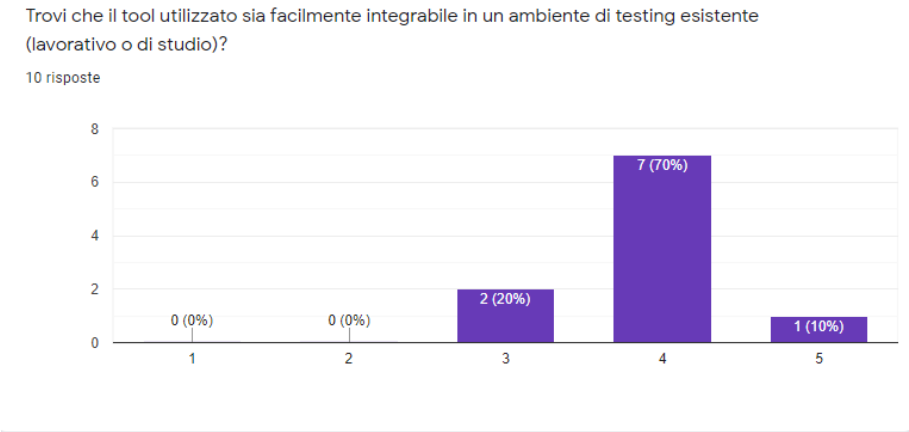


Figura 6.8. Applicabilitá in contesti lavorativi

6.4.2 Consapevolezza

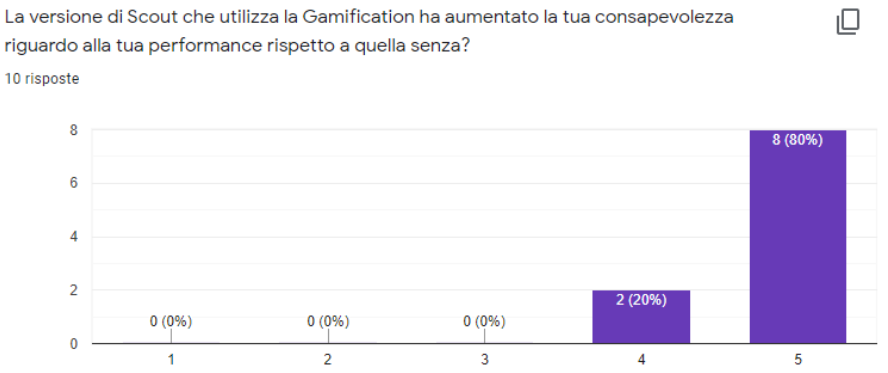


Figura 6.9. Consapevolezza delle proprie performance

6.4.3 Valutazione complessiva

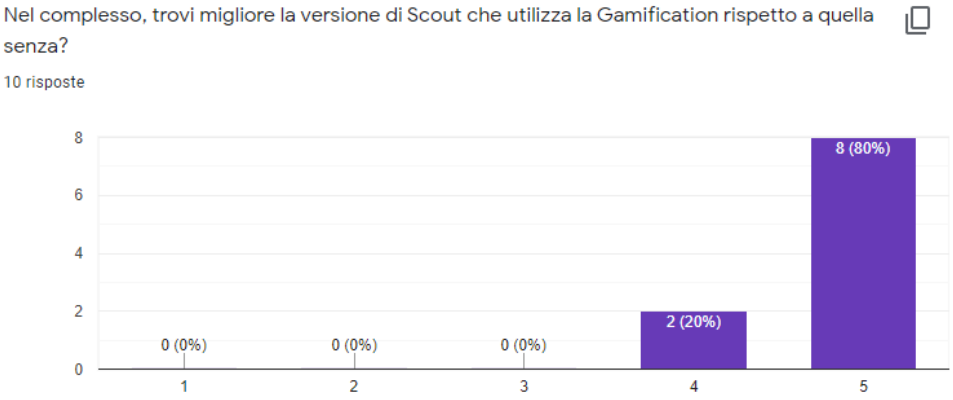


Figura 6.10. Valutazione complessiva

Come si evince dalle risposte precedenti, tutti i pratecipanti all'esperimento valutano il protitipo testato una buona alternativa alle strumentazioni attualmente in uso in contesti lavorativi.

6.5 Feedback

Nel questionario é stata aggiunta una domanda generale relativa ai futuri miglioramenti del tool proposto. Ecco elencati di seguito alcune delle osservazioni fatte:

1. "Sarebbe bello avere un negozio virtuale piú ampio e un maggior numero di obiettivi presenti."

2. "Ho trovato le informazioni all'interno del profilo un po' scarne, potrebbe essere interessante avere maggiori statische visualizzabili al suo interno."

3. "Ho notato che non é possibile visualizzare lo stato di segnalazione dei problemi trovati all'interno dell'applicativo. Penso che una caratteristica del genere possa essere molto importante anche per far comprendere al tester quanto il suo lavoro abbia impattato sullo sviluppo del software."

Capitolo 7

Conclusioni

A seguito dell'analisi svolta nel capitolo precedente si puó considerare l'esperimento un'ottima base di partenza. Nonostante ci sono grandi margini di miglioramento, il sistema di Gamification proposto ha portato a dei risultati in media maggiori rispetto alla versione standard di Scout. Il paramentro piú importante di valutazione peró é stato l'entusiasmo dimostrato dai partecipanti rispetto al sistema proposto, che va a modificare radicalmente tutti gli aspetti piú pesanti e ripetivi dell'attivitá di testing, riducendo, in potenza, lo stress e la frustrazione emotiva che puó ruotare attorno a questo lavoro. L'inserimento di componenti esterne con l'obiettivo di andare a coinvolgere emotivamente il tester, ha riscosso un notevole successo e al netto dei risultati ottenuti, con tutte le limitazioni del caso dovute al contesto storico in cui si sono svolti i vari esperimenti, ha evidenziato un miglioramento significativo anche dell'impostazione mentale con cui si va ad affrontare il lavoro in questione. Alessandro Baricco nel libro "The Game" fa un'analisi lucida, chiara ed estremamente realistica del cambiamento sociale che sta portando con sé l'evoluzione tecnologica digitale. Ogni cosa tende ad assomigliare sempre di piú ad un gioco, a partire dalla presentazione nel 2007 del primo iPhone. La tendenza ad affrontare la superficialitá degli strumenti che utilizzamo quotidianamente si pone in contraddizione con la ricerca della profonditá filosofica dell'inizio del XX secolo. Tuttavia a differenza dei pregiudizi o delle semplificazioni, questa superficalitá si mostra come un modo diverso di vivere il mondo, associando alla realtá virtuale un contesto molto piú reale di quello che attribuiamo solitamente al mondo digitale. Gli strumenti, come questo

sistema proposto, diventano un'estensione dell'attivitá creativa umana e non solamente una distrazione, come spesso si tende a credere. A partire dalle regole del Game Design si potrebbero in linea del tutto teorica ottenere dei risultati molto buoni per quelle che sono le metriche produttive attuali, capovolgendo significativamente l'approcio classico a tutte le professione e le attivitá umane. Risulta fondamentale, per questo motivo, continuare a porsi nuove domande per poter andare incontro alle esigenze emotive delle persone e non solo economiche delle aziende. La scomessa, in questo approccio, sta nel comprendere che il benessere emotivo delle persone porta a una maggiore produttivitá delle stesse, con conseguenze positive per tutte le aspettative del caso. Un approccio standard legato esclusivamente a metriche produttive, potrebbe essere in alcuni contesti, soprattutto a lungo andare, controproducente. La ricerca, le osservazioni, i pensieri raccolti in questa tesi vogliono essere un passo verso la creazione di strumenti che siano pensati attorno alle persone e non attorno al profitto, non dimenticando che come ogni idea, va accuramente verificata, validata e testata soprattutto in virtú di quello che saranno poi nel concreto le ricerche empiriche sul campo.

Bibliografia

Baricco, A., *The Game*, Einaudi, 2020

Bertolo, M. e Marian I., *Game Design - Gioco e giocare tra teoria e progetto*, Milano, Pearson, 2020.

Borjesson, E. and Feldt, R. , *Automated System Testing Using Visual GUI Testing Tools: A Comparative Study in Industry,*2012

Buckley, I. and Clarke, P., *An approach to Teaching Software Testing Supported by Two Different Online Content Delivery Methods*, 2018

Fraser, G., *Gamification of Software Testing*, 2017

Hamari, J. Koivisto,H. Sarsa, T. *"Does Gamification Work? — A Literature Review of Empirical Studies on Gamification"*, Conference: the 47th Hawaii International Conference on System Sciences, 2014

Herranz, E. and Colomo-Palacios, R.o and de Amescua, A. and Yilmaz, M., *Gamification as a Disruptive Factor in Software Process Improvement Initiatives*, Milano, Pearson, 2020.

Huzina, J., *Homo Ludens, 1938*

Herranz, E. and Colomo-Palacios, R. and Amescua-Sec, A., Towards a New Approach to Supporting Top Managers in SPI Organizational Change Management,*2012*

Nass,M. and Alégroth,E. and Feldt, R., Augmented Testing: Industry Feedback To Shape a New Testing Technology*, 2019*

Pedreira, O., García,F., Brisaboa, N., Piattini, M., Gamification in software engineering – A systematic mapping*, 2014*

José Rojas, M. and Fraser, G., Code Defenders: A Mutation Testing Game, Proc. of The 11th International Workshop on Mutation Analysis, 2016

Prasetya, W. and Leek, C. and Melkonian, O. and ten Tusscher J. and van Bergen., J. and Everink, J. and van der Klis, T. and Meijerink, R. and Oosenbrug, R. and Oostveen, J. and van den Pol, T. and van Zon, W., Having Fun in Learning Formal Specifications, 2019

Pranoto, H., Cuk Tho, H. L. Warnars, E. Abdurachman, F. Gaol, B., *Usability testing method in augmented reality application, 2017*

Smiderle, R., Rigo, S. J., Marques, Leonardo B., Peçanha de Miranda Coelho, J. A. Jaques, Patricia A. , "The impact of gamification on students' learning, engagement and behavior based on their personality traits"*, 2020*

Strmečki, D., Bernik A., e Radošević, D., Gamification in E-Learning: Introducing Gamified Design Elements into E-Learning Systems *, 2015*

Yu-Kai Chou, Actionable Gamification - Beyon Points, Badges and Leaderboards*, 2014*