POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Informatica

Tesi di Laurea Magistrale

Modellizzazione e Prototipo di un Applicativo di Security Assessment per l'Analisi e la Validazione della Postura di Sicurezza Informatica

Relatore
Prof. Alessandro Savino
Dr. Nicolò Maunero

Candidato
Francesco Galasso

Tutor Aziendale GPI Cyberdefence
Dr.ssa Manuela Sforza

Anno Accademico 2023-2024

Sommario

La crescente complessità dei sistemi aziendali e l'evoluzione delle minacce informatiche impongono nuove e significative sfide, con la necessità di superare le limitazioni dei processi tradizionali, i quali spesso si basano su strumenti manuali come i fogli di calcolo. Questi strumenti, sebbene utili in passato, si rivelano poco funzionali di fronte a volumi crescenti di dati e a contesti operativi sempre più complessi e dinamici. Una risposta efficace a tali esigenze risiede nell'automazione di questi processi, consentendone una gestione più rapida, accurata e sicura. Questo lavoro di tesi si pone l'obiettivo di progettare un applicativo di Security Assessment, capace di ingegnerizzare e automatizzare il processo di analisi e validazione della postura di sicurezza di un perimetro.
L'applicativo è stato concepito a partire da una struttura dati robusta, utilizzando sistemi di gestione di database relazionali. In generale, sono due i parametri fondamentali su cui il progetto si basa: i requisiti funzionali e i requisiti di sicurezza. I requisiti funzionali includono caratteristiche essenziali, come ad esempio la velocità operativa e la gestione di sessioni real-time, che sono cruciali per un'applicazione destinata a operare in ambienti dinamici. D'altra parte, i requisiti di sicurezza sono indispensabili per garantire la protezione di dati sensibili e il monitoraggio efficace delle attività. Sulla base di queste considerazioni, è stata condotta un'analisi costi-benefici relativa alla scelta delle tecnologie front-end e back-end. In questa fase, sono stati valutati i pro e i contro delle principali opzioni disponibili per entrambe le categorie, assicurandosi che le scelte tecnologiche fossero in linea con gli obiettivi progettuali. La robustezza delle soluzioni adottate e, di conseguenza, del sistema è stata infine dimostrata attraverso la simulazione di una minaccia informatica in ambienti diversi, evidenziandone l'efficacia e l'affidabilità.
È importante sottolineare che il progetto, oltre al valore funzionale, persegue una missione culturale: sensibilizzare l'utenza e contribuire alla disseminazione della cultura di sicurezza informatica, promuovendo una maggiore consapevolezza.

Indice

Elenco delle figure

Capitolo 1

Introduzione

Il lavoro di tesi è stato svolto in un'azienda che si occupa di sicurezza informatica in cui ho avuto la possibilità di fare un tirocinio curriculare.
Il progetto in questione ha come obiettivo la trasformazione di un servizio di Cybersecurity, attualmente gestito in maniera manuale, in un sistema completamente digitalizzato tramite lo sviluppo di una piattaforma web.
Ad oggi, l'erogazione del servizio necessita di una forte componente legata a processi umani, in quanto ci sono fogli Excel da compilare a mano, funzioni di calcolo e grafici da impostare, template predefiniti parzialmente compilati per la produzione di report. Tale modalità operativa, pur essendo funzionale, presenta diverse criticità in termini di efficienza, tracciabilità e standardizzazione. In generale, questo modo di procedere, con l'aumento della complessità, risulta essere molto poco scalabile e prono ad errori.
L'obiettivo del progetto è quindi quello di sviluppare un applicativo web che digitalizzi completamente l'intero flusso di lavoro. In pratica, tutte le operazioni gestite finora manualmente saranno integrate all'interno di un sistema automatizzato. La nuova applicazione consentirà di centralizzare e organizzare meglio le informazioni e il flusso di lavoro, riducendo il rischio di errori umani. Sarà possibile inserire e analizzare i dati in modo strutturato, automatizzare i calcoli necessari, dare una visione real-time al cliente, generare automaticamente report personalizzati basandosi su modelli predefiniti che saranno compilati dinamicamente, scaricare gli stessi in formato pdf.
Questo processo di digitalizzazione del servizio non solo migliorerà l'efficienza operativa, riducendo il tempo e le risorse impiegate nelle operazioni manuali,

ma permetterà anche di garantire maggiore precisione, velocità e sicurezza nella gestione delle informazioni sensibili, offrendo un servizio di Cybersecurity scalabile. Inoltre, l'applicazione web offrirà una user experience ottimizzata per gli operatori interni e per i clienti, facilitando il monitoraggio delle attività e la tracciabilità delle azioni.
Il progetto non può limitarsi alla semplice digitalizzazione del servizio, ma deve incorporare fin dall'inizio una forte attenzione alla security by design. Questo principio prevede che la sicurezza venga integrata in ogni fase del ciclo di sviluppo del software, piuttosto che essere considerata un'aggiunta successiva. In altre parole, ogni decisione progettuale, come la scelta delle tecnologie utilizzate per lo sviluppo, deve tenere conto delle possibili vulnerabilità e implementare soluzioni che garantiscano un elevato livello di protezione.
Infatti, una parte rilevante della tesi riguarda lo studio delle tecnologie di sviluppo adottate. Sarà svolta un'analisi dettagliata per giustificare le scelte fatte, con un focus specifico sulle loro capacità di supportare la sicurezza. Per dimostrare l'efficacia delle scelte prese, la tesi si concentrerà anche sulla simulazione di un attacco informatico. In particolare, verranno simulate due situazioni: una in cui l'applicazione è sviluppata con le tecnologie scelte e un'altra in cui vengono utilizzate tecnologie diverse, meno focalizzate sulla sicurezza. Attraverso questa simulazione, sarà possibile confrontare come le due applicazioni reagiscono a un attacco e dimostrare la maggiore robustezza della prima soluzione. Questo esperimento offrirà una prova concreta dell'importanza di integrare la sicurezza nelle fasi iniziali del ciclo di sviluppo e di scegliere tecnologie che permettano di mitigare le vulnerabilità.
L'elaborato è strutturato come segue:

- **Capitolo 2**: Questo capitolo fornisce un quadro dettagliato del background, concentrandosi quindi sulla teoria che c'è dietro al servizio offerto dall'azienda;

- **Capitolo 3**: Vengono descritti ad alto livello il servizio erogato dall'azienda e l'architettura dell'applicativo da sviluppare;

- **Capitolo 4**: In questo capitolo, vengono esaminate le fasi opertive dello sviluppo dell'applicativo svolte fino a questo punto. Inoltre, vengono

gettate le basi per la scelta delle tecnologie, argomento che sarà poi approfondito nel capitolo successivo;

- **Capitolo 5**: Si confrontano le principali tecnologie di sviluppo front-end e back-end, analizzandone pro e contro. Da questo confronto, viene spiegato il motivo delle scelte tecnologiche adottate per l'applicativo in questione e si accenna alla fase di sviluppo del modulo di Login. Inoltre, si anticipa il tema del capitolo 6;

- **Capitolo 6**: Si presenta la simulazione di un attacco informatico effettuato in ambienti differenti, attraverso cui si dimostra la solidità delle tecnologie scelte per la piattaforma, illustrando i risultati ottenuti e il processo seguito per raggiungerli;

- **Capitolo 7**: Questo capitolo conclude il lavoro di tesi, evidenziando gli sviluppi futuri e la mission del progetto.

Capitolo 2

Background

Questo capitolo ha l'obiettivo di fornire una panoramica sul Framework Nazionale per la Cybersecurity e la Data Protection del 2019, che costituisce la base dell'applicativo che si intende sviluppare per ingegnerizzare un servizio di security assessment attualmente offerto dall'azienda. Nel corso del capitolo, verrà illustrata la linea evolutiva che il Framework ha subito nel tempo, modellata dall'evoluzione delle minacce informatiche e dalle conseguenti esigenze di protezione sempre più stringenti. Si forniranno informazioni sul suo utilizzo pratico, accennando le versioni precedenti e concentrando l'attenzione sull'ultima, che rappresenta lo scenario più attuale.

2.1 Cybersecurity: una priorità globale

Negli ultimi anni, la cybersecurity ha assunto un ruolo sempre più centrale nel mondo moderno, diventando una delle priorità principali per aziende, governi e cittadini.
La digitalizzazione è aumentata esponenzialmente in ogni settore, sia perché ci si è resi conto che spostare online le attività di business implica possibilità di sviluppo maggiori, sia per fattori più indiretti come l'emergenza sanitaria causata dal COVID. Di conseguenza, è emersa con forza la consapevolezza che le minacce informatiche non solo sono più diffuse, ma anche più sofisticate e difficili da contrastare. Il numero crescente di attacchi informatici ha messo in luce la vulnerabilità dei sistemi digitali su cui si basa gran parte delle nostre attività. Le conseguenze di un attacco non riguardano solo la perdita

di dati, ma possono mettere in pericolo la sicurezza nazionale, la stabilità economica e la fiducia dei consumatori. In questo scenario, la cybersecurity è passata dall'essere una preoccupazione tecnica confinata ai dipartimenti IT a diventare una vera e propria questione strategica a livello globale.
Un altro aspetto molto importante è l'evoluzione della consapevolezza dei rischi legati alla sicurezza informatica. Se in passato molte organizzazioni tendevano a sottovalutare la portata delle minacce, oggi c'è una comprensione molto più chiara del fatto che nessun sistema, per quanto ben progettato, si possa considerare immune agli attacchi. Questo ha portato a una maggiore enfasi sulla prevenzione, piuttosto che sulla sola reazione, e alla diffusione di buone pratiche come la crittografia dei dati, l'autenticazione multifattoriale e i programmi di educazione alla sicurezza per dipendenti e utenti.

2.2 Storia ed evoluzione dei Framework per la gestione della Cybersecurity

In questo paragrafo verrà tracciata l'evoluzione del Framework, analizzando le versioni antecedenti. Si farà un breve riferimento al NIST Cybersecurity Framework e al Framework Nazionale per la Cybersecurity e la Data Protection del 2015, offrendo una panoramica sulle loro principali caratteristiche e sull'impatto che hanno avuto nell'ambito della sicurezza informatica.

2.2.1 NIST Cybersecurity Framework

Andando in ordine cronologico, il Cybersecurity Framework ideato dal NIST si può considerare il nostro punto di partenza.
"Il NIST Cybersecurity Framework (CSF) 2.0 fornisce indicazioni all'industria, alle agenzie governative e ad altre organizzazioni per la gestione dei rischi di cybersecurity. Offre una tassonomia di risultati di alto livello in materia di cybersecurity che può essere utilizzata da qualsiasi organizzazione - indipendentemente dalle dimensioni, dal settore o dalla maturità - per comprendere, valutare, dare priorità e comunicare meglio i propri sforzi in materia di cybersecurity. Il CSF non prescrive come raggiungere i risultati. Piuttosto, rimanda a risorse online che forniscono ulteriori indicazioni sulle pratiche e sui controlli che potrebbero essere utilizzati per raggiungere tali

risultati" [1].
Attraverso questa definizione, possiamo dire di aver inquadrato uno dei temi fondamentali di questo elaborato. Infatti, nonostante col tempo ci siano state delle evoluzioni in termini di Framework su cui ci concentremo maggiormente, la mission di fondo resta la stessa.
In sostanza, il Framework è stato sviluppato per aiutare le organizzazioni a gestire e ridurre i rischi legati alla Cybersecurity in modo strutturato. Si tratta di uno strumento flessibile e adattabile, pensato per supportare aziende di qualsiasi settore e dimensione nella protezione delle proprie infrastrutture informatiche. Esso offre una serie di linee guida e pratiche che consentono di identificare, proteggere, rilevare, rispondere e recuperare da incidenti di sicurezza. Tuttavia, non impone soluzioni specifiche: lascia la libertà di adottare le tecnologie e i processi più adatti al contesto specifico dell'organizzazione. Il suo scopo principale è quello di migliorare la gestione della sicurezza informatica, aiutando le aziende a sviluppare strategie coerenti per affrontare le minacce digitali, con un focus sul miglioramento continuo.

2.2.2 Framework Nazionale per la Cybersecurity - 2015

Nel 2015 è stato introdotto il Framework Nazionale per la Cybersecurity, sviluppato grazie alla collaborazione tra il mondo accademico, gli enti governativi e le imprese commerciali. Questo Framework, progettato per essere utilizzato sia da organizzazioni pubbliche che private e indipendentemente dalla loro dimensione, rappresenta una risorsa pratica pensata per strutturare e gestire i processi di sicurezza informatica in modo efficace. Dunque, lo scopo di questo strumento *"è quello di offrire alle organizzazioni un approccio volontario e omogeneo per affrontare la cyber security al fine di ridurre il rischio legato alla minaccia cyber. L'approccio di questo Framework è intimamente legato a una analisi del rischio e non a standard tecnologici"* [2].
Il Report del CIS Sapienza (Cyber Intelligence and Information Security), a cura di Roberto Baldoni e Luca Montanari [2], sottolinea che al fine di mantenere un'armonia a livello internazionale il Framework si ispira al Cybersecurity Framework del NIST, di cui abbiamo dato una visione ad alto livello nel paragrafo precedente, che era stato concepito per le infrastrutture critiche. Nonostante ciò, questo strumento è stato adattato alla realtà italiana, caratterizzata soprattutto dalla presenza di piccole e medie imprese.

È importante evidenziare che il Framework non è uno standard di sicurezza. Piuttosto, può essere visto come una struttura di riferimento in cui possono essere inquadrati standard e normative. La definizione degli standard è compito di organismi di standardizzazione nazionali e internazionali, nonché degli enti regolatori del settore, e l'adozione del Framework è volontaria. *"L'adozione di questo Framework da parte delle organizzazioni residenti nel nostro paese può portare ad un irrobustimento dell'intero sistema paese rispetto ad attacchi di tipo cibernetico"* [2].

2.3 Framework Nazionale per la Cybersecurity e la Data Protection - 2019

Negli ultimi anni, le minacce informatiche sono diventate sempre più sofisticate e numerose.
Tra i rischi più rilevanti emergono i data breach, ovvero violazioni che permettono di accedere illegalmente a informazioni riservate contenute nelle banche dati di aziende, enti pubblici e organizzazioni di ogni tipo. Questi attacchi comportano il furto di dati sensibili, come informazioni personali, finanziarie o industriali, mettendo a rischio la privacy degli individui e la sicurezza delle operazioni aziendali. Le conseguenze di tali violazioni possono essere devastanti, sia in termini economici che reputazionali, poiché le aziende non solo devono affrontare i costi del ripristino della sicurezza e del recupero dei dati, ma rischiano anche di perdere la fiducia dei clienti. Inoltre, con l'introduzione del Regolamento Generale sulla Protezione dei Dati (GDPR) nell'Unione Europea, i data breach comportano anche pesanti sanzioni economiche. Le aziende che non rispettano i requisiti di protezione dei dati, come l'adozione di misure adeguate per prevenire tali violazioni, possono essere soggette a multe che arrivano fino al 4% del fatturato globale annuo o a 20 milioni di euro, a seconda di quale sia l'importo maggiore. Questa normativa ha contribuito a spingere le organizzazioni a rafforzare i propri sistemi di sicurezza e ad adottare protocolli più rigidi per la protezione dei dati, rendendo la cybersecurity una questione cruciale non solo per evitare violazioni, ma anche per garantire conformità legale.
Alla luce di quanto appena detto, viene presentata una versione aggiornata del Framework Nazionale per la Cybersecurity e la Data Protection, pensata

come un supporto per le organizzazioni che devono implementare strategie e processi mirati alla sicurezza informatica e alla tutela dei dati personali.
Come la versione rilasciata nel 2015, accennata nel paragrafo precedente, anche questa nuova edizione si ispira al Cybersecurity Framework del NIST, includendo componenti pensate per affrontare le sfide legate alla sicurezza nelle supply chain e approfondendo le misure necessarie per garantire la protezione dei processi di autenticazione e la gestione dell'identità.
In ogni caso, la nuova variante include nuovi elementi focalizzati sugli aspetti chiave della protezione dei dati, in linea con le disposizioni del Regolamento Generale sulla Protezione dei Dati (GDPR). A tal fine, sono stati presi in considerazione i seguenti aspetti [4] relativi alla data protection:

- I processi di data management, con particolare attenzione a quelli relativi alle informazioni personali;
- Le procedure per il trattamento dei dati personali;
- La definizione di ruoli e responsabilità nella gestione dei dati personali;
- La valutazione dell'impatto sulla protezione dei dati personali;
- Le modalità di registrazione e comunicazione in caso di incidenti che comportino una violazione dei dati personali.

È importante sottolineare che il Framework non deve essere considerato come uno strumento per garantire compliance alle normative vigenti, bensì come un supporto utile alle organizzazioni nel percorso di gestione della sicurezza informatica. Infatti, sebbene non assicuri automaticamente l'adempimento dei requisiti di legge, la sua applicazione può facilitare lo sviluppo di strategie e processi che risultano allineati con le normative in materia di cybersecurity e protezione dei dati, come il GDPR. In questo modo, l'organizzazione può ottimizzare le risorse e ridurre i costi necessari per implementare le misure di sicurezza, aumentando al contempo l'efficacia delle soluzioni adottate. Inoltre, per quelle aziende che già hanno avviato iniziative in linea con il Regolamento Generale sulla Protezione dei Dati, il Framework può diventare un riferimento prezioso per guidare le indispensabili attività di continuous monitoring, assicurando che le misure di protezione rimangano aggiornate e adeguate alle minacce in continua evoluzione.

In definitiva, il Framework offre una base strutturata per costruire e mantenere un sistema di sicurezza informatica robusto, che possa rispondere in modo efficace alle sfide contemporanee [3]. Esso permette di avere un sistema di gestione in grado di valutare i controlli di sicurezza presenti e, ove mancanti, progettarne l'implementazione in ordine di priorità. È un modello che permette l'intersoggettività interpretativa così come la documentabilità delle scelte. L'organizzazione, infatti, è tenuta non solo ad implementare le misure di sicurezza, ma anche a dimostrare, ove richiesto, la sua compliance normativa.
A questo punto, procederemo con un'analisi dettagliata del Framework Nazionale per la Cybersecurity e la Data Protection, esaminandone le varie componenti e il modo in cui può essere applicato all'interno delle organizzazioni per migliorare la gestione della sicurezza informatica e la protezione dei dati. Faremo riferimento al documento ufficiale intitolato "Framework Nazionale per la Cybersecurity e la Data Protection" del CIS-Sapienza (Research Center of Cyber Intelligence and Information Security), elaborato in collaborazione con il CINI Cybersecurity National Lab (Consorzio Interuniversitario Nazionale per l'Informatica) [4].

2.3.1 Nozioni fondamentali del Framework

Come già accennato, il Framework riprende alcuni degli aspetti fondamentali del NIST Cybersecurity Framework: **Framework Core**, **Profile** e **Implementation Tier (Livelli di Maturità)**. Esaminiamoli singolarmente.

Framework Core

"Il core rappresenta la struttura del ciclo di vita del processo di gestione della cybersecurity, sia dal punto di vista tecnico sia organizzativo" [4]. In sostanza, rappresenta l'insieme dei requisiti che garantiscono la sicurezza del ciclo di vita del dato.
Il Core ha una struttura gerarchica, composta da **function**, **category** e **subcategory**:

- **Function**: Le funzioni rapprentano le tematiche principali da tenere in considerazione quando si inizia un percorso che mira alla gestione della postura di sicurezza di un perimetro. Esse sono:

- **IDENTIFY** (ID): La funzione IDENTIFY si riferisce alla comprensione del contesto aziendale, degli asset che supportano i processi critici e dei rischi associati. Questa analisi consente all'organizzazione di allineare risorse e investimenti con la strategia di gestione del rischio e gli obiettivi aziendali. In particolare, implica l'identificazione del perimetro di analisi e dei flussi operativi, nonché il monitoraggio della supply chain e delle interdipendenze funzionali. Inoltre, considera i requisiti normativi relativi alla protezione dei dati e i processi preesistenti di valutazione dei rischi. Le category incluse in questa funzione sono: Asset Management, Business Environment, Governance, Risk Assessment, Risk Management Strategy, Supply Chain Risk Management e Data Management;
- **PROTECT** (PR): La funzione PROTECT è correlata all'implementazione di misure destinate a salvaguardare i processi di business e gli asset aziendali, a prescindere dalla loro natura informatica. Questo comporta la protezione del perimetro aziendale attraverso misure tecniche e organizzative che siano pertinenti al contesto e adeguate al profilo di rischio intrinseco. Tali misure hanno lo scopo di garantire la riservatezza, l'integrità e la disponibilità dei dati e delle informazioni (CIA). Le category incluse in questa funzione sono: Identity Management, Authentication and Access Control, Awareness and Training, Data Security, Information Protection Processes and Procedures, Maintenance, Protective Technology;
- **DETECT** (DE): La funzione DETECT si riferisce alla definizione e attuazione di attività idonee per identificare prontamente incidenti di sicurezza informatica. Questo implica l'implementazione di controlli da parte dell'organizzazione per rilevare anomalie e, di conseguenza, attivare una risposta proattiva. Le category incluse in questa funzione sono: Anomalies and Events, Security Continuous Monitoring, Detection Processes;
- **RESPOND** (RE): La funzione RESPOND è correlata alla definizione e attuazione delle attività necessarie per intervenire in caso di rilevamento di un incidente di sicurezza informatica. L'obiettivo principale è limitare l'impatto di un potenziale incidente, garantendo una risposta efficace che dimostri resilienza sia dal punto di vista

tecnico, attraverso la mitigazione, sia da quello organizzativo, mediante una comunicazione appropriata. Le category incluse in questa funzione sono: Response Planning, Communications, Analysis, Mitigation, Improvements;

- **RECOVER** (RC): La funzione RECOVER è legata alla definizione e all'implementazione delle attività necessarie per gestire i piani e le operazioni di ripristino dei processi e dei servizi colpiti da un incidente. L'obiettivo è garantire la resilienza dei sistemi e delle infrastrutture, sostenendo un recupero tempestivo delle operazioni aziendali dopo un evento critico. Ciò include anche la verifica delle attività necessarie per ripristinare la situazione preesistente alla crisi. Le category incluse in questa funzione sono: Recovery Planning, Improvements, Communications;

- **Category**: Le categorie formulano i requisiti necessari sul piano strategico;

- **Subcategory**: Le sottocategorie specificano sul piano tattico le attività da implementare.

Il Framework, pertanto, delinea per ciascun Core, dato dall'insieme di funzione, categoria e sottocategoria, le attività abilitanti, vale a dire i processi e le tecnologie necessarie per gestire ogni singola funzione. Inoltre, il Framework Core presenta anche delle **informative reference**, cioè un insieme di riferimenti che legano la singola subcategory alle procedure di sicurezza indicate negli standard di settore o da regolamentazioni generali.

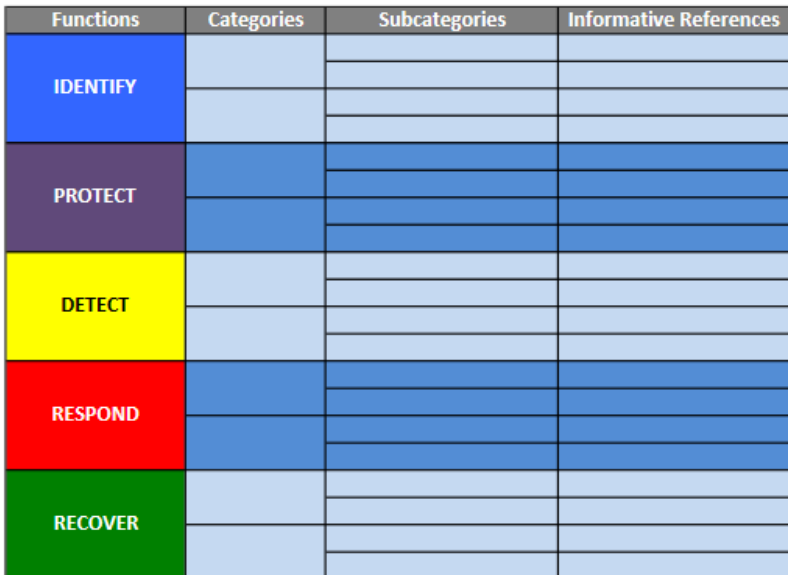


Figura 2.1. Struttura del Framework Core

Profile

I Profili rappresentano il risultato della selezione, da parte di un'organizzazione, di specifiche category e subcategory del Framework applicabili al contesto in base a vari fattori, come la valutazione del rischio, il contesto di business, l'applicabilità delle varie subcategory all'organizzazione.
Il profilo si sviluppa in due fasi: il profilo target e il profilo corrente. Il **profilo target** rappresenta il modello di riferimento applicabile, nel mondo ideale, al rischio intrinseco precedentemente calcolato. Il **profilo corrente**, invece, descrive le misure applicate nel nostro perimetro oggetto di analisi. Confrontando i due profili, un'organizzazione può identificare le aree di miglioramento e stabilire priorità per colmare le eventuali lacune.
La selezione delle subcategory avviene considerando gli obiettivi dell'organizzazione e la valutazione dei propri rischi, e può includere anche pratiche aggiuntive non previste dal Framework, al fine di gestire in modo completo il rischio. Il profilo corrente può quindi essere utilizzato per definire priorità e per misurare i progressi che ci sono, o comunque che ci dovrebbero essere, verso il profilo target. I profili possono essere utilizzati anche per fare un'autovalutazione o per comunicare il livello di gestione del rischio cyber all'interno o all'esterno dell'organizzazione.

Implementation Tier (Livelli di Maturità)

Gli Implementation Tier ci danno delle informazioni relative a quanto e come l'organizzazione in questione integri processi di Cybersecurity. I livelli di valutazione sono quattro e sono (dal più debole al più forte):

- **Parziale**: Il modello di gestione del rischio di cybersecurity di un'organizzazione risulta parziale se non considera in modo strutturato il rischio cyber o le minacce esterne. Spesso, il rischio cyber viene affrontato con approcci occasionali e in modo reattivo, con una consapevolezza limitata a livello aziendale. Inoltre, non esistono processi formali per condividere informazioni legate alla cybersecurity con soggetti esterni;

- **Informato**: Il modello di gestione del rischio cyber di un'organizzazione è informato se esistono processi interni che considerano il rischio cyber, ma questi non coinvolgono l'intera azienda. Sebbene la consapevolezza del rischio sia diffusa, manca un approccio gestionale completo che coinvolga tutti i livelli aziendali. L'organizzazione è consapevole del proprio ruolo nell'ecosistema di riferimento, ma lo scambio di informazioni sugli eventi di cybersecurity rimane limitato e per lo più passivo;

- **Ripetibile**: Il modello di gestione del rischio cyber di un'organizzazione è ripetibile quando è formalmente strutturato, approvato e soggetto a continui aggiornamenti, basati sui risultati del processo di gestione del rischio. Questo approccio coinvolge tutti i livelli aziendali, con personale adeguatamente formato per ricoprire i ruoli assegnati. L'organizzazione condivide regolarmente informazioni di cybersecurity con altri attori del proprio ecosistema;

- **Adattivo**: Il modello di gestione del rischio cyber di un'organizzazione è considerato adattivo quando le procedure di cybersecurity vengono continuamente aggiornate in base alle esperienze passate e agli indicatori di rischio. In questo modo, l'organizzazione è in grado di adattarsi costantemente alle minacce emergenti e di rispondere in modo efficace agli attacchi più sofisticati. La condivisione di informazioni con altri attori del settore avviene in modo continuo e in tempo reale.

Come abbiamo notato dal titolo, ho indicato l'aspetto dell'Implementation Tier anche con il nome di **Livelli di Maturità**. Questo è un punto importante sia del Framework in generale, sia nello specifico della versione che è stata adattata dall'azienda in cui è stato svolto il tirocinio. Infatti, nonostante a livello generale sia vero il fatto che la parte relativa all'Implementation Tier rappresti una delle tre nozioni fondamentali del Framework Nazionale per la Cybersecurity e la Data Protection, è anche vero che l'azienda considera i Livelli di Maturità, che sono comunque discussi nel documento a cui ci stiamo riferendo, come l'effettivo aspetto chiave del Framework assieme al Core e al Profilo. Andiamo dunque ad approfondire questo concetto.
I Livelli di Maturità rappresentano il grado di integrazione che il sistema di gestione della sicurezza dei dati e delle informazioni ha all'interno del quadro gloable dei processi aziendali. Detta in questo modo, sembra una definizione piuttosto simile a quella di Implementation Tier data precedentemente. In realtà, i livelli di maturità *"permettono di fornire una misura della maturità di un processo di sicurezza, della maturità di attuazione di una tecnologia specifica o una misura della quantità di risorse adeguate impiegate per l'implementazione di una data subcategory"* [4]. In altre parole, per Livello di Maturità si intende il grado di attuazione di una determinata procedura di sicurezza. Sono importanti poiché consentono di avere un landmark per valutare i passi da compiere in termini di sicurezza, permettendo di definire anche una scala di priorità. I livelli di Maturità adattati al Framework utilizzato dall'azienda sono quattro e saranno discussi nel capitolo successivo.
I Livelli di Maturità, dunque, consentono di delineare le procedure di sicurezza che sono necessarie per raggiungere il livello desiderato.

2.3.2 Altri elementi del Framework

Livelli di priorità

I livelli di priorità rappresentano l'importanza strategica di un controllo rispetto ad un altro nel raggiungimento del profilo target. In sostanza, si dà importanza prioritaria a quegli interventi che possano ridurre con un impatto maggiore i livelli di rischio.
La determinazione dei livelli di priorità per le subcategory deve basarsi su due criteri fondamentali:

- Capacità di ridurre il rischio cyber, intervenendo su uno o più fattori chiave per la determinazione dello stesso, ovvero:
  - Esposizione alle minacce, ossia i fattori che influenzano la probabilità di un attacco;
  - Probabilità del loro verificarsi, cioè la frequenza con cui possono manifestarsi;
  - Impatto che queste minacce avrebbero sulle operazioni aziendali o sugli asset, ovvero l'entità del danno potenziale;
- Facilità di implementazione, considerando il livello di maturità tecnologica e organizzativa necessario per eseguire l'azione specifica.

I livelli di priorità previsti dal Framework Nazionale per la Cybersecurity e la Data Protection sono tre:

- **Alta**: Si riferisce a interventi che consentono di diminuire in modo significativo uno dei tre fattori chiave del rischio cyber. Questi interventi sono considerati prioritari e devono essere implementati a prescindere dalla loro complessità di realizzazione;
- **Media**: Riguarda interventi che contribuiscono a ridurre uno dei tre fattori chiave del rischio cyber e che, generalmente, risultano facili da implementare;
- **Bassa**: Si riferisce a interventi che offrono una riduzione di uno dei tre fattori chiave del rischio cyber, ma che sono generalmente complessi da realizzare, spesso richiedendo modifiche organizzative significative o importanti cambiamenti infrastrutturali.

Prototipi di contestualizzazione

I prototipi rappresentano i template che innestati sul Core intercettano la conformità a specifiche normative inerenti alla cybersecurity. Essi possono servire, ad esempio, per cogliere attraverso il Framework:

- Requisiti normativi che impongono l'adozione di specifiche misure per la cybersecurity o la protezione dei dati;

- Regolamenti tecnici e di esecuzione che delineano controlli precisi in ambito cybersecurity o data protection;
- Best practice di settore relative alla sicurezza informatica o alla protezione dei dati.

Un prototipo di contestualizzazione, applicato ad una specifica subcategory del Core, va a specificare la relativa **classe di implementazione** con tre possibilità:

- **Obbligatoria**: Ogni volta che quel determinato prototipo è implementato, la subcategory in questione deve essere inclusa;
- **Consigliata**: Ogni volta che quel determinato prototipo è implementato, si consiglia di includere la subcategory in questione;
- **Libera**: Ogni volta che quel determinato prototipo è implementato, l'inclusione della subcategory in questione è lasciata alla libera scelta degli operatori.

Inoltre, un prototipo di contestualizzazione può definire, per ogni subcategory, un certo livello di priorità per la relativa implementazione.
A questo punto, c'è un altro aspetto molto importante su cui porre l'attenzione. Infatti, è importante sottolineare che per ogni prototipo c'è una sorta di guida all'implementazione. Si tratta di un documento che ci dà alcune informazioni in merito al prototipo stesso, quali:

- Il contesto in cui ha senso applicare quel certo prototipo;
- Vincoli di vario tipo sulla selezione delle subcategory e sul modo in cui possono essere definiti i livelli di priorità;
- I **controlli**. Ad ogni prototipo, infatti, corrisponde un elenco di controlli di sicurezza che saranno innestati sulle subcategory considerate. Questi controlli saranno organizzati in modo opportuno nei diversi livelli di maturità eventualmente previsti.

Arrivati a questo punto, concentriamoci su come utilizzare il Framework. A questo proposito, ci sono due passaggi importanti da fare: la **contestualizzazione** del Framework ad un certo ambito applicativo e l'**applicazione** del Framework ad un'organizzazione.

2.3.3 Contestualizzazione del Framework

La contestualizzazione è quella fase in cui l'analista seleziona, tra tutte le misure desiderabili contenute nel modello, quelle ritenute applicabili al perimetro su cui si sta analizzando e validando la postura di sicurezza e che dunque costituiscono le misure adeguate rispetto al profilo di rischio considerato.
Quest'attività parte prendendo in considerazione gli elementi fondamentali del Framework che abbiamo precedentemente descritto. Questi elementi sono di carattere generale e, quindi, quando si contestualizza il Framework, tutti o alcuni di essi possono essere considerati.
Una contestualizzazione del Framework avviene attraverso una serie di passi che vengono eseguiti:

1. Scelta delle funzioni, categorie e sottocategorie che risultano essere pertinenti all'organizzazione in questione sulla base di alcuni criteri che possono essere il settore produttivo, la dimensione, l'espansione sul terriorio. A ciò si aggiunge la scelta dei Prototipi da innestare al Core. A questo proposito, implementare un prototipo in una contestualizzazione prevede i seguenti step:

   (a) Tutte le sottocategorie che sono obbligatorie per quello specifico prototipo vengono inserite nella contestualizzazione;

   (b) Le sottocategorie che sono consigliate per quello specifico prototipo devono essere valutate con attenzione;

   (c) Rispetto dei vincoli sulla selezione delle sottocategorie, specificati nella guida relativa all'applicazione di quel determinato prototipo;

   (d) Indicare, per ogni sottocategoria selezionata nei passi precedenti, un livello di priorità, facendo sempre riferimento a ciò che è spiegato nella guida;

   (e) Gli eventuali controlli di sicurezza specificati nella guida di applicazione del prototipo possono essere inclusi nelle linee guida all'applicazione della contestualizzazione.

2. Unione dei due moduli logici (Core e Prototipi);

3. Selezione delle misure applicabili, considerando anche i livelli di priorità, i livelli di maturità (almeno per le subcategory a priorità alta) e la classe

di attività. In generale, è opportuno definire delle linee guida almeno per le subcategory a priorità alta.

Il processo di contestualizzazione è in genere messo a punto dall'organizzazione stessa, anche se può essere delegato a terzi, prendendo talvolta spunto da caratteristiche che derivano da altre organizzazioni.
A questo punto, ripetendo il processo per tutti i protipi che sono da includere nel caso specifico, la contestualizzazione che viene fuori può prevedere, se necessario, l'aggiunta di caratteristiche ulteriori.

2.3.4 Applicazione del Framework

L'applicazione del Framework si articola in una serie di passaggi chiave:

1. **Identificare una contestualizzazione del Framework**: Se l'organizzazione opera in un settore regolamentato, dovrebbe adottare una delle contestualizzazioni previste dal regolatore del settore o svilupparne una propria, eventualmente utilizzando prototipi che interpretino le normative applicabili. Se l'organizzazione non appartiene a un settore regolamentato, può scegliere una delle contestualizzazioni esistenti o crearne una specifica per implementare il Framework;

2. **Definire priorità e ambito**: È fondamentale identificare periodicamente gli obiettivi strategici e le priorità aziendali, per individuare le aree e le funzioni critiche su cui concentrare l'attenzione;

3. **Identificare sistemi e asset**: Definire quali informazioni e sistemi, sia IT che industriali, siano essenziali per il funzionamento dell'organizzazione, così da valutare correttamente i rischi e comprendere le reali esigenze di protezione;

4. **Determinare il profilo corrente**: Valutare lo stato di implementazione e il livello di maturità per ogni subcategory del Framework, al fine di elaborare uno o più profili correnti, riferiti alle aree/funzioni coinvolte nell'implementazione. In sostanza, il profilo corrente del perimetro altro non è che una fotografia dei controlli implementati nel perimetro rispetto a tutti i controlli considerati adeguati;

5. **Analizzare il rischio**: Analizzare i rischi utilizzando una metodologia appropriata, adattata al contesto operativo e al settore in cui l'organizzazione opera;

6. **Determinare il profilo target**: L'organizzazione deve stabilire un profilo target, che rappresenti il livello di implementazione e maturità desiderato per ciascuna subcategory del Framework. In sostanza, il profilo target del perimetro è il profilo ideale. Se possibile, è preferibile integrare la gestione del rischio cyber nel programma generale di gestione del rischio aziendale, così che il top management possa prendere decisioni in modo coordinato e strategico;

7. **Determinare il gap rispetto al profilo target**: Confrontare il profilo corrente con quello target per individuare i gap che ci sono nella gestione della cybersecurity.
   Il grado di scostamento tra i due profili (target e corrente) è ciò che determinerà la vulnerabilità, che definisce in sostanza il profilo di rischio che rimane. La variabile **Vulnerabilità** è in genere misurata su una scala qualitativa che può variare in base alle scelte fatte da chi eroga il servizio. Nel capitolo successivo, riprenderemo questo concetto e daremo delle indicazioni più specifiche, contestualizzate alla realtà aziendale;

8. **Definire e attuare una roadmap per raggiungere il profilo target**: Definire le azioni necessarie per raggiungere il profilo target, stabilendo un piano operativo che consideri i rischi emersi e le specificità dell'organizzazione. Questo prevede l'elaborazione di un piano temporale per implementare i controlli del Framework.
   A questo proposito, l'azienda ha adottato una strategia che prevede di pianificare tre "To be", rispettivamente "To be - I fase", "To be - II fase" e "To be - III fase", che accenneremo nel terzo capitolo;

9. **Misurare le performance**: Stabilire metriche per monitorare e revisionare periodicamente l'efficacia del profilo target, tenendo conto anche dei costi operativi. Le valutazioni sull'efficienza del profilo corrente devono guidare la definizione del nuovo profilo target per favorire il miglioramento continuo.

Capitolo 3

Cyberframe: architettura

Il terzo capitolo si concentra sulla spiegazione di come l'azienda ha riadattato il Framework Nazionale per la Cybersecurity e la Data Protection, trattato nei capitoli precedenti, per fornire un servizio di Cybersecurity su misura. Verrà quindi illustrato a grandi linee l'operato dell'azienda, evidenziando i principali adattamenti introdotti rispetto allo standard al fine di rispondere meglio alle esigenze specifiche di mercato.

3.1 Punto di partenza

Come abbiamo già accennato nei capitoli precedenti, l'azienda in cui ho svolto il tirocinio eroga attualmente il servizio di Cybersecurity in questione in maniera manuale, utilizzando un template ben strutturato e complesso in formato Excel. Questo file Excel, frutto di una meticolosa progettazione, è suddiviso in diverse schede (o tab), ognuna delle quali racchiude informazioni specifiche che guidano il processo di assessment. Di seguito un esempio di tale template al fine di avere una visione generale più chiara:

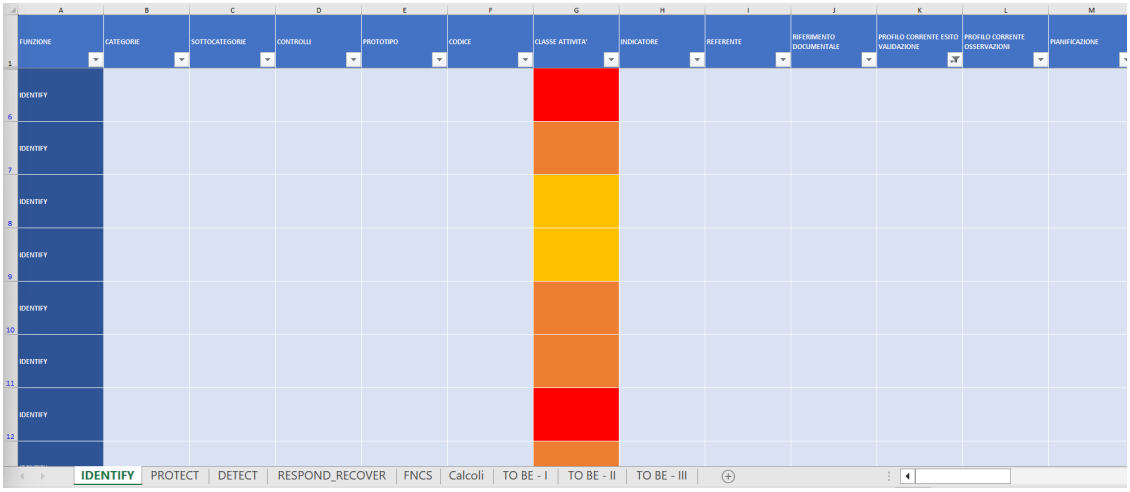


Figura 3.1. Template Excel

3.1.1 Sezione Function

Il cuore di questo template ruota attorno a uno degli elementi fondamentali del Framework Nazionale per la Cybersecurity e la Data Protection: il **Core**. Come abbiamo visto, il Core ha una struttura gerarchica ed è suddiviso in tre livelli di dettaglio crescente: **function**, **category** e **subcategory**, dove la function rappresenta il livello di astrazione più alto. Le funzioni sono cinque e coprono l'intero ciclo di gestione della Cybersecurity: dall'identificazione dei rischi alla protezione degli asset, dalla rilevazione degli incidenti alla risposta e al recupero successivo. All'interno del template Excel, ci sono quattro schede principali dedicate a queste funzioni. Nello specifico, un tab è riservato alla funzione **IDENTIFY** (ID), un altro alla funzione **PROTECT** (PR), il terzo alla funzione **DETECT** (DE) e, infine, un quarto tab combina insieme le funzioni **RESPOND** (RE) e **RECOVER** (RC). Queste schede non solo strutturano il lavoro di assessment, ma permettono anche una gestione organizzata delle varie attività, semplificando il processo di analisi e monitoraggio dei rischi. Ciascuna tab rappresenta un'area operativa specifica del Core, offrendo un quadro chiaro e ben definito delle attività da svolgere in ciascuna fase del processo di Cybersecurity.
In aggiunta a queste quattro schede principali, esistono ulteriori tab che

servono ad approfondire altri aspetti del processo di Cybersecurity e che esamineremo successivamente. Intanto, possiamo approfondire la struttura delle quattro schede principali, partendo dal presupposto che queste, nonostante rappresentino funzioni diverse, condividono la stessa configurazione e le stesse colonne, le quali sono progettate per guidare e standardizzare il processo di assessment e, quindi, ciò che le differenzia è solo il tema del ciclo di gestione della Cybersecurity trattato. Questa struttura che ora andremo ad esaminare permette una visione chiara e organizzata delle attività e dei controlli da effetturare, creando un flusso ordinato di informazioni che agevola l'intero processo di assessment.
Ognuno dei diversi campi specifici che caratterizzano uno dei quattro tab dedicati alle funzioni ha un ruolo chiave nel raccogliere e organizzare i dati essenziali. Concentrandosi su di essi, l'azienda riesce a mantenere un alto livello di coerenza e accuratezza nella gestione dei dati critici per ogni funzione. Tra le colonne principali troviamo:

- **Funzione**: Come abbiamo spiegato nel capitolo precedente, le funzioni rappresentano le tematiche principali da tenere in considerazione quando si inizia un percorso che mira alla gestione della postura di sicurezza di un perimetro;

- **Categorie**: Le categorie formulano i requisiti necessari sul piano strategico;

- **Sottocategorie**: Le sottocategorie specificano sul piano tattico le attività da implementare;

- **Prototipo**: I prototipi rappresentano i template che innestati sul Core intercettano la conformità a specifiche normative inerenti alla Cybersecurity. Ad oggi, l'azienda ha ingegnerizzato i seguenti prototipi:

  - **Circolare n. 2/2017**: misure minime AgID (**AGID**);
  - **Regolamento (UE) 2016/679**: protezione dei dati (**GDPR**);
  - **NIST SP 800-218 1.1**: sviluppo sicuro (**SSDF**);
  - **NIST AI 100-1 - RMF 1.1**: intelligenza artificiale (**AI**);
  - **DPCM 81/2021**: normative operatori di servizi essenziali (**OSE**);

  - **Determina AgID 628/2021 e s.m.i**: requisiti per il cloud (**Allegati A/A2 – B/B2**);
  - **Regolamento (UE) 2022/2554**: enti finanziari (**DORA**).

- **Controlli**: Ad ogni prototipo, corrisponde un elenco di controlli di sicurezza che saranno innestati sulle subcategory considerate. Questi controlli saranno organizzati in modo opportuno nei diversi livelli di maturità eventualmente previsti;

- **Classe attività**: Nel template aziendale, il campo "Classe attività" assume un ruolo simile a quello dei livelli di priorità definiti nel Framework e descritti nel secondo capitolo, i quali rappresentano l'importanza strategica di un controllo rispetto ad un altro nel raggiungimento del profilo target. Tuttavia, se nel Framework sono previsti tre livelli di priorità (alta, media e bassa), nel modello adottato dall'azienda il concetto è stato riadattato con quattro possibili classi di attività:

  - **Obbligatoria**;
  - **Consigliata Alta**;
  - **Consigliata Media**;
  - **Consigliata Bassa**.

  Anche se il princpio è simile, in quanto entrambe le classificazioni stabiliscono l'importanza relativa dei controlli, la struttura aziendale permette una valutazione più articolata delle attività e introduce una maggiore granularità, distinguendo le attività obbligatorie da quelle consigliate e stabilendo ulteriori livelli di specificità per questa seconda casistica. Il valore che può assumere questo campo dipende, appunto, dal livello di rilevanza del controllo corrente e il tutto è finalizzato ad un'ottimizzazione della gestione e della distribuzione delle risorse nel processo di assessment;

- **Indicatore**: Il campo Indicatore, non sempre presente, permette di fornire maggiori informazioni in merito al controllo corrente, consentendo di capire meglio cosa andrebbe fatto o cosa dovrebbe esserci affinchè quel controllo possa considerasi implementato.
  Nella figura sottostante, troviamo un esempio di quanto appena detto:

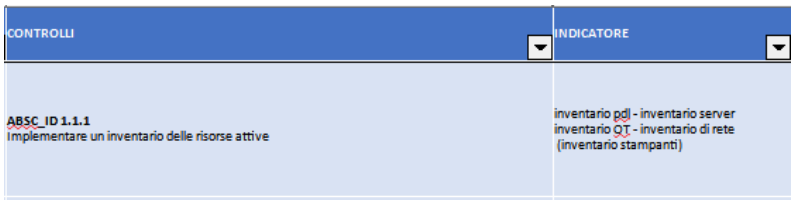


Figura 3.2. Esempio Indicatore

- **Riferimento documentale**: Questo campo contiene i documenti che attestano il livello in cui l'azienda si posiziona rispetto all'implementazione del controllo corrente;
- **Profilo corrente esito validazione**: Nel capitolo 2, abbiamo accennato i **Livelli di Maturità** e abbiamo detto che essi sono una sorta di reingegnerizzazione dell'**Implementation Tier**. Questo campo si riferisce proprio a questo aspetto del Framework.
  Escludendo le procedure di sicurezza considerate non applicabili per l'organizzazione, il Livello di Maturità può essere (dal minore al maggiore):
  - **Non implementato**: La procedura di sicurezza in questione non è considerata dall'organizzazione in nessuna forma;
  - **Parzialmente implementato**: Sebbene la procedura di sicurezza in questione sia considerata dall'organizzazione su tutte le aree in cui ciò andrebbe fatto, l'applicazione della stessa è parziale;
  - **Implementato su perimetro verticale**: L'organizzazione implementa pienamente una certa procedura di sicurezza su una determinata area, ma la stessa non è considerata o è considerata parzialmente in altri contesti relativi al perimetro stesso. Ad esempio, immaginando che sia necessario un inventario degli asset, ciò è presente in modo completo per i client e in modo incompleto per i server;
  - **Implementato**: La procedura di sicurezza in questione è considerata appieno dall'organizzazione.
- **To be - I Fase, To be - II Fase, To be - III Fase**: Questi campi permettono di definire le azioni necessarie per avvicinarsi al profilo target, stabilendo un piano operativo che consideri i rischi emersi e le

specificità dell'organizzazione. Questo prevede l'elaborazione di un piano temporale per implementare i controlli del Framework. Ecco l'obiettivo di questi tre "To be": ognuno di questi rappresenta un vero e proprio piano di azione, in cui si fanno delle proiezioni nel tempo volte a migliorare gradualmente il grado di maturità relativo ai controlli che sono stati selezionati per l'organizzazione in questione.

La figura seguente proviene dal template utilizzato dall'azienda e, nello specifico, è un estratto del tab relativo alla funzione Identify. Lo scopo è quello di dare un'idea "visiva" di come esso si presenta.

FUNZIONE	CATEGORIE	SOTTOCATEGORIE	CONTROLLI	PROTOTIPO
IDENTIFY	**ASSET MANAGEMENT (ID.AM):** I dati, il personale, i dispositivi e i sistemi e le facilities necessari all'organizzazione sono identificati e gestiti in coerenza con gli obiettivi e con la strategia di rischio dell'organizzazione	**ID.AM-1:** Sono censiti i sistemi e gli apparati fisici in uso nell'organizzazione ISO/IEC 27001:2013 A.8.1.1, A.8.1.2 Allegato A Determina AgID 628/21 Allegato B Determina AgID 628/21	**Allegato A2-B2-O-1** Tutti i sistemi e gli apparati fisici sono censiti ed esiste un elenco di quelli approvati da attori interni al soggetto.	ALLEGATO A2-B2
IDENTIFY	**ASSET MANAGEMENT (ID.AM):** I dati, il personale, i dispositivi e i sistemi e le facilities necessari all'organizzazione sono identificati e gestiti in coerenza con gli obiettivi e con la strategia di rischio dell'organizzazione	**ID.AM-1:** Sono censiti i sistemi e gli apparati fisici in uso nell'organizzazione ISO/IEC 27001:2013 A.8.1.1, A.8.1.2 Allegato A Determina AgID 628/21 Allegato B Determina AgID 628/21 DPCM 81/2021	**OSE_ID 2.1.1.1** Tutti i sistemi e gli apparati fisici sono censiti ed esiste un elenco di quelli approvati da attori interni al soggetto. L'elenco contiene, ove possibile, i riferimenti agli identificativi della componentistica del bene ICT così come indicati nel modello dei beni ICT.	OSE
IDENTIFY	**ASSET MANAGEMENT (ID.AM):** I dati, il personale, i dispositivi e i sistemi e le facilities necessari all'organizzazione sono identificati e gestiti in coerenza con gli obiettivi e con la strategia di rischio dell'organizzazione	**ID.AM-1:** Sono censiti i sistemi e gli apparati fisici in uso nell'organizzazione ISO/IEC 27001:2013 A.8.1.1, A.8.1.2 Allegato A Determina AgID 628/21 Allegato B Determina AgID 628/21	**Allegato A2-B2-O-2** Tutti i sistemi e gli apparati fisici presenti sulle reti sono censiti e l'accesso alla rete è consentito esclusivamente a quelli approvati.	ALLEGATO A2-B2
IDENTIFY	**ASSET MANAGEMENT (ID.AM):** I dati, il personale, i dispositivi e i sistemi e le facilities necessari all'organizzazione sono identificati e gestiti in coerenza con gli obiettivi e con la strategia di rischio dell'organizzazione	**ID.AM-1:** Sono censiti i sistemi e gli apparati fisici in uso nell'organizzazione ISO/IEC 27001:2013 A.8.1.1, A.8.1.2 Allegato A Determina AgID 628/21 Allegato B Determina AgID 628/21 DPCM 81/2021	**OSE_ID 2.1.1.2** Tutti sistemi e gli apparati fisici presenti sulle reti sono censiti e l'accesso alla rete e' consentito esclusivamente a quelli approvati.	OSE
IDENTIFY	**ASSET MANAGEMENT (ID.AM):** I dati, il personale, i dispositivi e i sistemi e le facilities necessari all'organizzazione sono identificati e gestiti in coerenza con gli obiettivi e con la strategia di rischio dell'organizzazione	**ID.AM-1:** Sono censiti i sistemi e gli apparati fisici in uso nell'organizzazione ISO/IEC 27001:2013 A.8.1.1, A.8.1.2 Allegato A Determina AgID 628/21 Allegato B Determina AgID 628/21	**ABSC_ID 1.1.1** Implementare un inventario delle risorse attive	AGID
IDENTIFY	**ASSET MANAGEMENT (ID.AM):** I dati, il personale, i dispositivi e i sistemi e le facilities necessari all'organizzazione sono identificati e gestiti in coerenza con gli obiettivi e con la strategia di rischio dell'organizzazione	**ID.AM-1:** Sono censiti i sistemi e gli apparati fisici in uso nell'organizzazione ISO/IEC 27001:2013 A.8.1.1, A.8.1.2 Allegato A Determina AgID 628/21 Allegato B Determina AgID 628/21	**ABSC_ID 1.1.2[1]** Implementare l'inventario attraverso uno strumento automatico	AGID
IDENTIFY	**ASSET MANAGEMENT (ID.AM):** I dati, il personale, i dispositivi e i sistemi e le facilities necessari all'organizzazione sono identificati e gestiti in coerenza	**ID.AM-1:** Sono censiti i sistemi e gli apparati fisici in uso nell'organizzazione ISO/IEC 27001:2013 A.8.1.1, A.8.1.2	**ABSC_ID 1.1.3**	AGID

Figura 3.3. Esempio struttura tab function

A questo punto, dopo aver chiarito i campi principali delle schede relative alle funzioni, facciamo un passo avanti che permette di proiettarci alla descrizione di altri due aspetti fondamentali del template. Per poter fare ciò, poniamo l'attenzione su alcuni concetti:

- Attraverso la prima parte delle colonne dei tab che abbiamo esaminato, viene tracciato il cosiddetto **profilo target**, ossia il livello ideale di implementazione dei controlli di sicurezza per l'azienda. Questo profilo rappresenta l'obiettivo a cui tendere, ovvero uno scenario in cui tutti i controlli sono perfettamente implementati e gestiti, consentendo di

raggiungere il massimo livello di sicurezza previsto per il perimetro considerato. L'adozione completa del profilo target corrisponde, quindi, allo stato ottimale verso cui orientare le strategie aziendali;

- Un altro aspetto fondamentale è la colonna denominata "Profilo corrente esito validazione", che offre una valutazione della maturità effettiva dei controlli che andrebbero implementati per il perimetro in esame. In altre parole, questa colonna ci dà una misura dello stato attuale di ciascun controllo e descrive il **profilo corrente** dell'azienda;
- Infine, poniamo di nuovo l'attenzione sui "**To Be**". Come già accennato, essi rappresentano delle proiezioni per lo sviluppo futuro e delineano le tappe da seguire per avvicinarsi progressivamente al profilo target. La logica dei To Be è quindi quella di offrire una visione intermedia che supporta l'azienda nel processo di crescita e di maturazione della sicurezza, in un percorso pianificato che gradualmente riduce il divario tra profilo corrente e profilo target.

Questi concetti appena descritti sono elementi essenziali per le successive analisi e per i calcoli di gap e maturità. Questi calcoli sono riportati in ulteriori tab del template che analizzeremo brevemente nel sottoparagrafo successivo, trattandosi di una componente centrale per la misurazione e l'ottimizzazione continua della sicurezza aziendale.

3.1.2 Calcoli e To be

Il passo successivo, una volta definito il profilo target e il profilo corrente in base ai parametri discussi, consiste nell'eseguire una serie di calcoli volti a misurare la **postura di sicurezza** del perimetro oggetto di analisi. Posto che i calcoli utilizzati rappresentano un know-how aziendale, in questa sede ne descriveremo soltanto i principi generali. A questo proposito, nel template è presente un'apposita tab denominata "**Calcoli**".
Alla base del processo, vi è innanzitutto la distinzione tra le varie **Categorie** di perimetro sottoposto a risk assessment, ovvero "Ente" e "Applicazione", che identificano i diversi tipi di ambito da analizzare e proteggere. Per la Categoria "Ente", le possibili **Tipologie** includono "Piccolo Comune", "Pubblica Amministrazione" e "Infrastruttura Critica"; per la Categoria "Applicazione", invece, si possono selezionare le opzioni "Applicazione Software" o

"Medical Device". La scelta della Categoria e della rispettiva Tipologia rappresenta un passo chiave, in quanto influisce direttamente su alcuni **valori di ponderazione** che verranno poi moltiplicati nei calcoli specifici, uno per ogni diversa Classe di Attività possibile.
Oltre a questi valori di ponderazione che abbiamo accennato, altri parametri influenzano i calcoli. Infatti, il campo "Profilo corrente esito validazione", che stabilisce il livello di maturità dei controlli implementati, contribuisce a determinare ulteriori valori di ponderazione che, a loro volta, verranno applicati nelle operazioni di calcolo.
Alla fine del processo, si arriva alla definizione di due valori: uno per il profilo target e uno per il profilo corrente. L'ottenimento di questi due valori rappresenta un punto cruciale, poiché il confronto tra essi permette di calcolare la **postura di sicurezza** del perimetro, ovvero l'indice di sicurezza del perimetro in questione rispetto al complessivo totale delle misure di sicurezza considerate adeguate. La postura di sicurezza viene espressa come percentuale, calcolata sommando il punteggio ponderato ottenuto dai singoli controlli (profilo corrente) diviso il punteggio totale (profilo target). Questo valore percentuale risulta particolarmente utile, poiché consente di derivare un'ulteriore variabile: la **Vulnerabilità**.
Il grado di scostamento tra i due profili (target e corrente) è ciò che determinerà la Vulnerabilità, che definisce in sostanza il profilo di rischio che rimane. La variabile Vulnerabilità è misurata su una scala qualitativa che varia da 1 a 3, interpretabile attraverso i seguenti criteri:

- **V=1**: La variabile Vulnerabilità ha valore 1 se PC >= 70% PT, cioè se il Profilo Corrente implementa almeno il 70% dei controlli del Profilo Target;

- **V=3**: La variabile Vulnerabilità ha valore 3 se PC <= 30% PT, cioè se il Profilo Corrente implementa meno del 30% dei controlli del Profilo Target;

- **V=2**: In tutti gli altri casi.

Possiamo concludere questo discorso dicendo che la Vulnerabilità non è altro che l'inverso logico della postura di sicurezza, in quanto ad una postura di sicurezza alta corrisponde una Vulnerabilità bassa e viceversa.

Il template prevede anche tre ulteriori tab, denominate "**To Be - I**", "**To Be - II**" e "**To Be - III**". Queste tre sezioni permettono di ripetere l'analisi già descritta per la tab "Calcoli", ma adottando come profilo corrente le diverse proiezioni future (I, II, o III) anziché lo stato attuale. In questo modo, è possibile calcolare ipotetiche posture di sicurezza in base agli obiettivi prefissati per ogni fase di sviluppo, confrontandole costantemente con il profilo target (che non cambia) e monitorando l'evoluzione nel processo di risk assessment. In sostanza, si capisce il livello che si otterrebbe qualora si rispettassero le stime fatte per il futuro, portando il livello di maturità dei controlli a quello definito in fase di assessment.
Infine, possiamo dire che, a corredo di queste analisi, il template integra una serie di grafici che permettono di visualizzare in maniera più chiara e schematica i risultati.

Capitolo 4

Cyberframe: lavoro svolto

Nel quarto capitolo della tesi, si procede con una panoramica dell'applicativo vero e proprio. Sarà data una visione generale sull'applicazione web che si intende sviluppare, con un focus su una descrizione ad alto livello della sua architettura e delle funzionalità principali. In questo modo, sarà possibile iniziare a comprendere il valore aggiunto che si vuole raggiungere attraverso lo sviluppo dell'applicativo, fornendo una visione complessiva del tool per il risk assessment in questione e permettendo di prepararsi ad entrare nei dettagli tecnici e specifici.
A questo punto, si inizia ad esplorare la parte tecnica e operativa del lavoro. Questo capitolo si focalizzerà su alcuni degli aspetti pratici delle attività che ho svolto, offrendo una descrizione delle varie fasi tecniche e degli strumenti impiegati. Infatti, saranno descritti sia il lato più pratico, sia gli strumenti software e tecnologici adottati, offrendo dunque una base teorica per comprendere come ogni elemento contribuisca a realizzare un sistema automatizzato e sicuro.
In chiusura di capitolo, verrà introdotta la questione della scelta delle tecnologie front-end e back-end, aspetto su cui mi sono concentrato in modo specifico. Sebbene in questo capitolo non tratterò in dettaglio questa tematica, sottolineerò l'importanza di selezionare in modo accurato le tecnologie per entrambi gli aspetti. Infatti, il ruolo di queste scelte tecnologiche si rivela cruciale per garantire l'efficienza, la sicurezza e la scalabilità dell'applicativo, parametri chiave per un sistema di questo tipo. Questo accenno introduttivo servirà a porre le basi per i successivi due capitoli di quest'elaborato.

4.1 Cyberframe: architettura nuovo applicativo

Nella seconda sezione di questo capitolo, introduciamo la visione generale dell'applicativo di security assessment che si intende sviluppare. Questa sezione mira a fornire un quadro d'insieme sul funzionamento dell'applicativo, senza entrare nei dettagli tecnici specifici che saranno poi affrontati nei capitoli successivi. Qui vogliamo concentrarci sull'user experience prevista per i destinatari, vale a dire gli operatori dell'azienda e i clienti, con una descrizione delle schermate principali che riflettono il flusso di interazione.
La progettazione di queste schermate è stata supportata dall'utilizzo di **Miro**, uno strumento collaborativo online versatile, molto utile nella fase di prototipazione e pianificazione di progetti complessi. Miro consente di creare schemi, mappe concettuali, wireframe e mockup in modo intuitivo, facilitando la condivisione di idee e la collaborazione tra team. Grazie a Miro, è stato possibile simulare con precisione il flusso delle schermate e l'user experience, permettendo di visualizzare i processi chiave e le funzionalità centrali dell'applicativo ancor prima di procedere alla sua effettiva realizzazione.

4.1.1 Benefici lato azienda

Lato azienda, l'applicativo rappresenta un significativo passo avanti nella gestione delle attività di security assessment, semplificando e ottimizzando molte delle operazioni che prima richiedevano l'utilizzo di file Excel complessi e articolati o comunque, in generale, di attività manuali. Lavorare con un sistema digitale dedicato, infatti, riduce sensibilmente il rischio di errori. L'applicativo rende questo processo più sicuro e immediato, consentendo agli operatori di concentrarsi maggiormente sull'analisi e meno sulla gestione manuale dei dati.
Una delle caratteristiche principali del sistema è l'automazione di molte funzioni chiave, come la scelta dei prototipi e la conseguente classificazione dei controlli, il calcolo della vulnerabilità e della postura di sicurezza. Ogni azione avviene all'interno di un'interfaccia utente intuitiva, in cui è possibile effettuare scelte e impostazioni con pochi clic, mentre l'applicativo, collegato

a un database, si occupa di salvare e incrociare i dati in tempo reale. Grazie a questa automazione, il sistema calcola e visualizza istantaneamente i risultati, garantendo un monitoraggio real-time aggiornato e coerente con i parametri impostati, evitando che l'operatore debba effettuare manulamente le varie operazioni.
Inoltre, il database integrato consente di avere una base di dati condivisa e facilmente accessibile, dove ogni modifica o aggiornamento ai dati è immediatamente disponibile a tutti gli utenti autorizzati. Ciò facilita anche la creazione di report e la consultazione di dati storici, permettendo all'operatore di accedere velocemente ai risultati di analisi precedenti e di osservare come le misure adottate abbiano influito sulla sicurezza del perimetro nel tempo. Questo sistema centralizzato non solo semplifica le operazioni quotidiane, ma supporta una gestione della sicurezza più solida e basata su dati concreti, migliorando l'affidabilità complessiva del processo di assessment e ottimizzando i tempi operativi.
In sostanza, l'applicativo permette un'esperienza di lavoro più fluida e precisa, in cui l'operatore può focalizzarsi sulle azioni da intraprendere e sui risultati strategici dell'analisi anziché sui dettagli tecnici della compilazione, con un flusso di lavoro più ordinato e una gestione della sicurezza decisamente più controllata ed efficiente.

4.1.2 Benefici lato cliente

Lato cliente, l'user experience è strutturata per offrire un accesso real-time chiaro e immediato ai risultati dell'assessment, senza richiedere particolari competenze tecniche per interpretare i dati. L'applicativo è pensato per presentare in modo visivo e intuitivo tutte le informazioni fondamentali sullo stato di sicurezza del perimetro analizzato. Attraverso l'uso di grafici real-time, tabelle riepilogative e report sintetici, l'utente finale può ottenere una panoramica complessiva del livello di sicurezza raggiunto, comprendendo rapidamente il rischio attuale e le informazioni principali.

4.1.3 Funzionalità principali e schermate chiave dell'applicativo

In questa sezione, verranno illustrate le principali funzionalità previste per l'applicativo di security assessment, accompagnate da una descrizione delle schermate chiave e della relativa interfaccia. L'obiettivo è fornire una panoramica delle modalità di utilizzo pensate per semplificare il lavoro degli operatori aziendali e offrire ai clienti un'esperienza intuitiva e accessibile. Le funzionalità dell'applicativo sono state progettate per minimizzare errori e automatizzare il più possibile i processi di valutazione della sicurezza, superando le limitazioni legate all'uso di modelli statici come Excel e permettendo, tramite un'interfaccia chiara e guidata, di accedere direttamente ai dati aggiornati e strutturati in un database centralizzato.

Modulo di Login

La schermata iniziale dell'applicativo si presenta come una pagina di login semplice ma completa, progettata per garantire accesso rapido e sicuro agli utenti. Gli elementi che troviamo sono il logo aziendale e il logo dell'applicativo, che identificano in modo chiaro l'ambiente di lavoro e rafforzano il brand dell'azienda, e lo spazio per la pubblicità. Nella parte destra della schermata, sono collocati i campi per l'inserimento delle credenziali, che rappresentano il punto d'accesso principale per gli operatori e i clienti dell'azienda. Oltre a questi, sono presenti alcune opzioni utili per gestire le credenziali in modo autonomo, in quanto l'utente può scegliere di ricordare i dati inseriti per evitare di ripetere il login ogni volta e, inoltre, in caso di dimenticanza della password, è disponibile un link che guida al recupero delle credenziali. Il pulsante principale per l'accesso, collocato in una posizione ben visibile, consente di procedere con il login una volta inseriti i dati corretti. Per i nuovi utenti che non hanno ancora un account, è presente il pulsante di registrazione, che reindirizza direttamente alla pagina dedicata alla creazione di un nuovo profilo.
La figura seguente, con le opportuni censure dovute a informazioni sensibili, rappresenta quanto appena descritto:

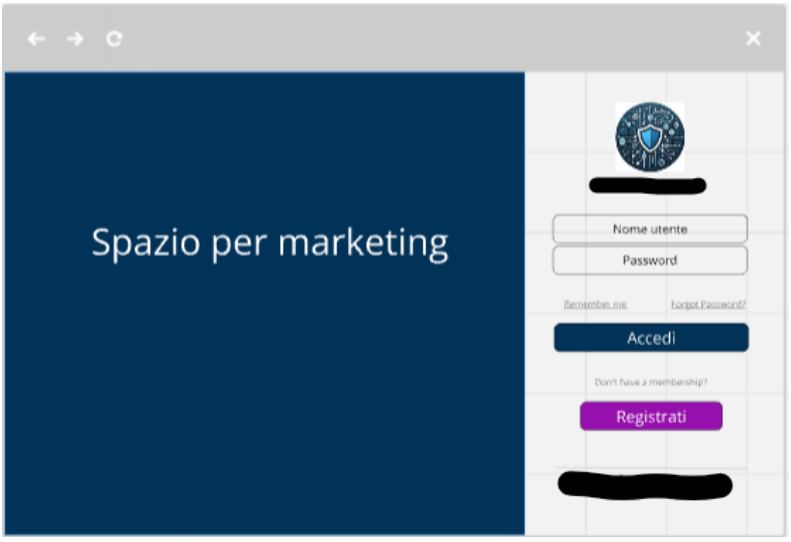


Figura 4.1. Modulo di Login

Dashboard

La schermata di Dashboard rappresenta il fulcro dell'esperienza utente nell'applicativo, offrendo una panoramica completa delle principali funzioni disponibili e facilitando l'accesso rapido a diverse sezioni tramite una serie di scorciatoie. In primo piano, infatti, troviamo una serie di shortcut che permettono agli utenti di accedere direttamente alle sezioni più utilizzate dell'applicativo, semplificando e velocizzando il flusso di lavoro quotidiano. È quindi possibile personalizzare la propria esperienza di utilizzo definendo nuove scorciatoie, rendendo così l'ambiente più adattabile alle necessità specifiche di ciascun utente.
L'elemento centrale e più caratteristico di questa schermata è il menù laterale, posizionato sulla sinistra. Questo menù si compone di una serie di voci disposte verticalmente, ognuna affiancata da un'icona rappresentativa e dal nome della sezione stessa. Il design è stato pensato per offrire un'esperienza flessibile: l'utente può scegliere se mantenere il menù espanso per visualizzare sia le icone che le voci o comprimerlo, riducendolo a una colonna di sole icone, così da focalizzare l'attenzione sulle informazioni centrali della dashboard. Il menù laterale è un elemento chiave poiché consente di avere, a colpo d'occhio, un riepilogo completo di tutte le funzionalità disponibili nell'applicativo. Selezionando ciascuna voce, si accede rapidamente alla relativa

sezione, consentendo così di svolgere una serie di operazioni senza la necessità di navigare più volte all'interno di vari livelli dell'applicativo. Infatti, è importante sottolineare che questo menù di cui parliamo è comune a tutto l'applicativo, ad esclusione della scheramta di Login/Registrazione.
Un aspetto fondamentale dell'applicativo in generale è la possibilità per l'utente di accedere a tutte le funzionalità e sezioni dello stesso in modo intuitivo e trasversale. L'applicativo è infatti strutturato in modo tale che ogni area consente di raggiungere altre parti dell'applicativo stesso, sia tramite il menù laterale sia attraverso collegamenti presenti direttamente nelle schermate operative. Questo design è stato pensato per garantire un'esperienza dinamica e senza interruzioni, dove l'utente non è vincolato a una navigazione rigida e può spostarsi fluidamente tra le sezioni in base alle proprie necessità.
Di seguito viene riportata la schermata Dashboard:

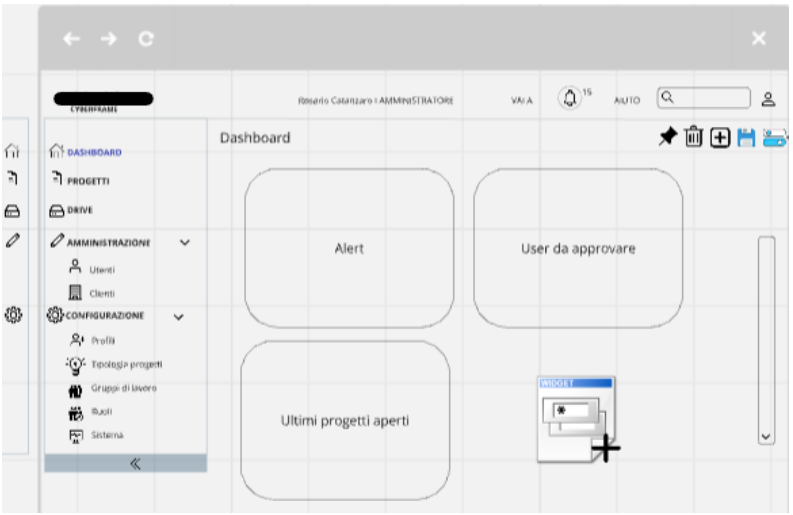


Figura 4.2. Dashboard

Modulo Progetti/Cruscotto FNCS

La sezione operativa rappresenta il vero cuore pulsante dell'applicativo. In sostanza, è la parte in cui si manifesta concretamente il processo di automazione a cui l'ingegnerizzazione dell'applicativo aspira. È qui, infatti, che si traduce l'obiettivo di spostare tutte le operazioni attualmente svolte manualmente in azienda verso una gestione più automatizzata e standardizzata.

Attraverso questo passaggio, l'applicativo mira a ridurre i margini di errore, velocizzare i processi e permettere agli operatori di concentrarsi maggiormente sull'analisi e sul miglioramento dei risultati. Lato cliente, invece, dà la possibilità di avere una piattaforma a cui fare accesso e di seguire real-time l'avanzamento dei lavori.
Cliccando sulla voce "Progetti" nel menù principale, si accede a una schermata dedicata in cui compare una lista di tutti i progetti creati, con una distinzione visiva tra progetti aperti e progetti chiusi, rispettivamente contrassegnati da un'icona con un lucchetto aperto e chiuso. Questa schermata include anche un'icona "+" che consente di avviare un progetto nuovo, facilitando così l'avvio rapido di un assessment completamente personalizzato e allineato alle esigenze del cliente.
Selezionando un progetto, sia esso nuovo, aperto o chiuso, l'utente viene reindirizzato a una schermata operativa in cui è possibile scegliere tra due sezioni principali, "Dati progetto" e "Cruscotto FNCS", tramite un menù presente in alto a destra. La sezione Cruscotto FNCS è particolarmente importante, poiché offre un insieme di strumenti chiave per configurare e monitorare tutti i parametri del perimetro in questione, andando a definire progressivamente il profilo target e il profilo corrente con una notevole precisione e velocità.
Nell'ipotesi di avviare un progetto da zero, una volta selezionato "Cruscotto FNCS" si accede a un menù orizzontale in cui sono presenti una serie di voci di configurazione, attraverso cui è possibile determinare e visualizzare ogni aspetto del perimetro oggetto di analisi. Le prime scelte importanti da definire riguardano la Categoria e la Tipologia del perimetro. Come abbiamo precedentemente detto, l'utente può decidere se il perimetro è un "Ente" o un'"Applicazione". Nel caso di un "Ente", le opzioni disponibili includono "Piccolo Comune", "Pubblica Amministrazione" e "Infrastruttura Critica"; in caso di "Applicazione", l'utente può scegliere tra "Applicazione Software" o "Medical Device". Queste scelte influiscono, ad esempio, su una serie di valori di ponderazione utilizzati nei calcoli successivi, che contribuiscono a determinare sia il profilo target che il profilo corrente.
A questo punto, nella voce del menù "Esecuzione Operativa" l'utente può iniziare a configurare e visualizzare il profilo target direttamente tramite query al database aziendale. Qui, l'interfaccia è pensata per rendere intuitivo il

processo di definizione del profilo target e del profilo corrente: diverse colonne appaiono già popolate e definitive, altre sono popolate e modificabili, altre ancora richiedono un lavoro manuale da parte degli operatori. Ogni modifica permette di aggiornare i parametri del progetto e di definire pian piano il profilo corrente e le proiezioni dei tre To Be.
La sezione "Statistiche" consente di monitorare il progetto attraverso grafici e tabelle interattive che si aggiornano in tempo reale, in base alle modifiche apportate nella sezione "Esecuzione Operativa". In questo modo, il cliente può osservare real-time l'evoluzione del profilo di sicurezza graficamente in modo chiaro e immediato.
Infine, una funzione essenziale è la generazione della relazione tecnica. Tramite una voce del menù chiamata "Relazione Tecnica", c'è infatti la possibilità, una volta che sono presenti a sistema tutti i dati necessari, di generare e scaricare un report completo e dettagliato, che riassume i risultati dell'assessment in una forma comprensibile e professionale, perfetta per essere condivisa con il cliente finale. Questo report include tutte le informazioni fondamentali emerse nel progetto. Il template della relazione è standard e non modificabile. Sarà possibile modificare solo alcune parti e ciò dipende dalle specificità del cliente e del lavoro in questione.
Per avere un'idea più chiara di quanto detto, si riporta la schermata che è possibile vedere in "Progetti/Cruscotto FNCS/Prototipi":

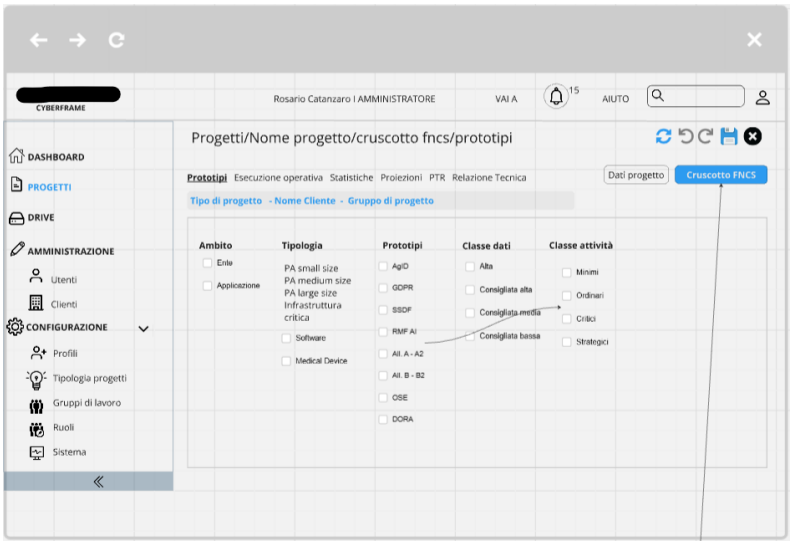


Figura 4.3. Progetti/Cruscotto FNCS/Prototipi

Questa visione generale dell'applicativo vuole evidenziare l'architettura user-centered che intende coniugare efficienza e usabilità. L'obiettivo finale è fornire agli operatori aziendali uno strumento rapido e preciso per il processo di risk assessment e offrire ai clienti un quadro chiaro delle condizioni di sicurezza dei propri asset e delle possibili azioni da intraprendere. Inoltre, c'è un importante concetto di mission legato alla Cybersecurity di cui faremo accenno successivamente.

4.2 Cyberframe: implementazione nuovo applicativo

La prima fase del mio lavoro si è concentrata sullo studio del background, ponendo le basi necessarie per comprendere a fondo non solo gli standard che regolano il Framework Nazionale per la Cybersecurity e la Data Protection, ma anche il contesto specifico aziendale in cui esso si inserisce.
Per quanto riguarda la prima parte, ho dunque approfondito la conoscenza del Framework, un insieme di linee guida e pratiche che offre un approccio standardizzato alla gestione della sicurezza informatica e della protezione dei dati sensibili. Esso rappresenta uno strumento di fondamentale importanza per molte organizzazioni, fornendo un modello strutturato per identificare, proteggere, rilevare, rispondere e ripristinare la sicurezza dei dati e dei sistemi informatici. Grazie alla sua impostazione modulare e flessibile, il Framework consente alle aziende di adattare le misure di sicurezza in base alle loro necessità operative e ai rischi specifici che devono affrontare, promuovendo così una visione della sicurezza che si adatta alle evoluzioni e alle sfide del settore.
Oltre allo standard, la fase di studio ha incluso un'analisi approfondita di come l'azienda abbia reinterpretato i regolamenti generali per rispondere alle proprie esigenze specifiche. Sono stati selezionati gli elementi centrali del Framework, mantenendo inalterati i principi e le pratiche che rispondono meglio ai requisiti della propria infrastruttura e alla natura del servizio offerto. Altri elementi, invece, sono stati modificati o riadattati, personalizzandoli per rispondere meglio alla struttura e ai flussi interni. Questo processo di adattamento ha reso possibile una conformità "su misura" agli standard nazionali, integrando così le esigenze di sicurezza con le esigenze di efficienza operativa specifiche della realtà di business. Durante questa fase, dunque,

mi è stato fornito accesso al know-how aziendale, permettendomi di comprendere a fondo il servizio di Cybersecurity che l'azienda eroga attualmente in modalità manuale. Questo approfondimento mi ha consentito di entrare nei meccanismi del servizio, studiando i processi, le procedure operative e le tecniche con cui viene gestita la sicurezza per i clienti. Allineandomi al contesto operativo esistente, sono riuscito a comprendere meglio le criticità e i punti di forza dell'approccio attuale, elementi fondamentali su cui basare il percorso di ingegnerizzazione e automatizzazione che si mira a raggiungere con l'applicativo.

4.2.1 DBMS: un approccio più pratico e diretto

Dopo aver acquisito una comprensione chiara della teoria e del contesto aziendale, ho iniziato a sviluppare un'idea per il database dell'applicativo. In questo processo, ho adottato un approccio particolarmente orientato alla praticità, poiché mi sono concentrato subito sull'aspetto della progettazione logica e della conseguente progettazione fisica. Quello che ho fatto è stato, quindi, adottare una modalità più pragmatica, saltando la fase di modellazione concettuale e concentrandomi subito sulla progettazione logica, avendo già in mente una chiara idea delle strutture dati necessarie. In altre parole, ho iniziato direttamente a delineare le tabelle e i campi principali, ipotizzando la forma e la struttura che il database avrebbe potuto avere per rispondere alle esigenze dell'applicativo. Questa scelta è stata motivata sia dall'urgenza di creare un prototipo rapido, sia dalla collaborazione di un team con esperienza nel design di database, che ha potuto integrare e completare le fasi precedenti con un approccio di reverse engineering rispetto alla metodologia standard.
Questa fase di progettazione logica è stata guidata dagli elementi chiave identificati durante l'analisi del Framework. Chiaramente, per l'applicativo definitivo saranno necessarie ulteriori tabelle per supportare altre funzionalità. In base a quanto appreso, ho sviluppato uno schema di tabelle iniziali per gestire i dati più importanti del processo di security assessment. Nello specifico, le tabelle in questione sono quattro:

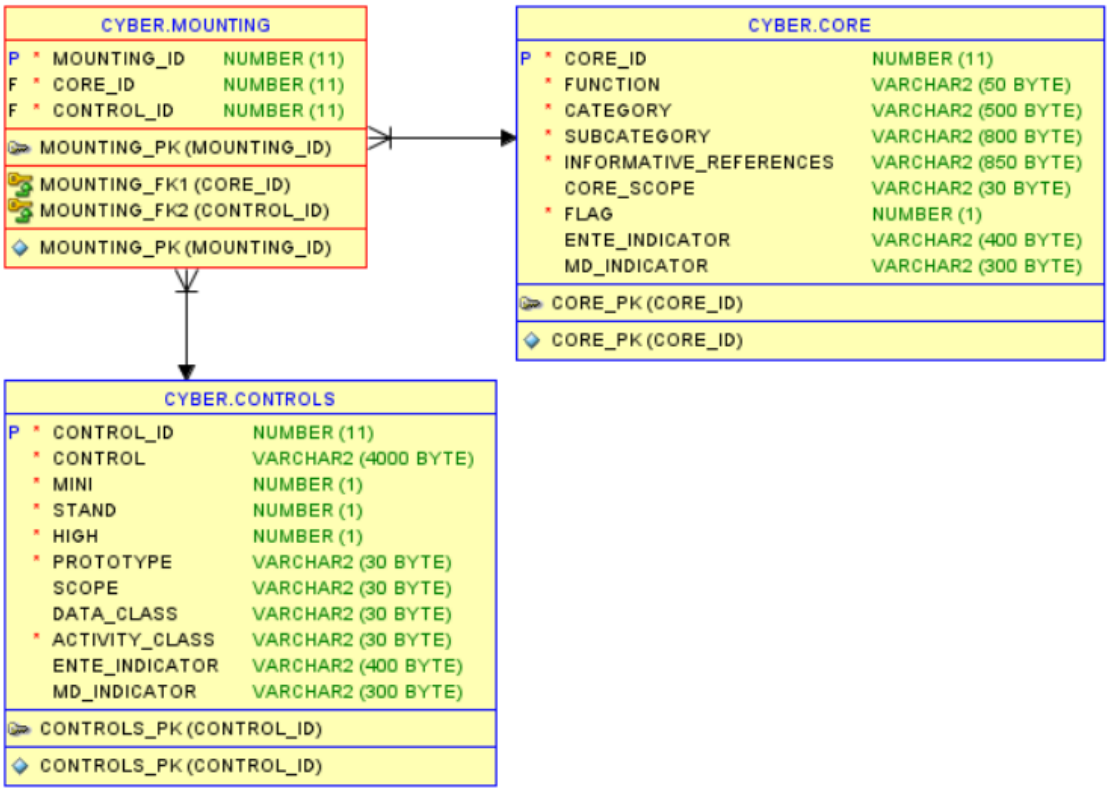
- **Core**: Nella tabella Core, inserisco tutti i core del framework, differenziati da idCore che è la primary key univoca;

- **Controlli**: Nella tabella Controlli, inserisco tutti i controlli possibili, anche in questo caso discriminati dalla primary key idControllo. Un'intuizione importante in tal senso è stata quella di vedere il prototipo, discusso nei capitoli precedenti, come un attributo e non come un'entità, semplificando la gestione del DBMS;

- **Montaggio**: La tabella Montaggio è, invece, la relazione tra le due precedenti table (tabella relazionale) e contiene quindi tutte le combinazioni possibili (core, controllo). La primary key è idMontaggio, mentre idCore e idControllo sono le foreign keys che servono appunto proprio per montare i controlli sui core;

- **StoricoLavori**: Questa tabella ha il compito di conservare lo storico di tutti i lavori fatti su tutte le categorie e le tipologie di perimetri, a prescindere dal fatto che siano lavori conclusi o in corso d'opera. Infatti, dato un determinato lavoro relativo a un certo perimetro, in questa tabella io vedrò tante righe quante sono i core e i controlli montati sul core relativi a questo perimetro e per discriminarle dalle altre utilizzerò la colonna "NomeLavoro", che conterrà il nome del lavoro considerato.

Queste tabelle rappresentano la struttura base dell'applicativo, in quanto rispondono alle esigenze strettamente operative legate al servizio offerto. Infatti, attraverso opportune query di cui poi daremo un accenno, è possibile, dopo aver selezionato i parametri opportuni, ricavere come output il profilo target. In seguito, con il lavoro successivo degli operatori, è poi possibile comporre il profilo corrente e tutte le informazioni mancanti, con la possibilità di salvare di volta in volta le modifiche.
Di seguito il modello relazionale del database:


CYBER.MOUNTING
P * MOUNTING_ID NUMBER (11)
F * CORE_ID NUMBER (11)
F * CONTROL_ID NUMBER (11)
MOUNTING_PK (MOUNTING_ID)
MOUNTING_FK1 (CORE_ID)
MOUNTING_FK2 (CONTROL_ID)
MOUNTING_PK (MOUNTING_ID)
CYBER.CORE
P * CORE_ID NUMBER (11)
* FUNCTION VARCHAR2 (50 BYTE)
* CATEGORY VARCHAR2 (500 BYTE)
* SUBCATEGORY VARCHAR2 (800 BYTE)
* INFORMATIVE_REFERENCES VARCHAR2 (850 BYTE)
CORE_SCOPE VARCHAR2 (30 BYTE)
* FLAG NUMBER (1)
ENTE_INDICATOR VARCHAR2 (400 BYTE)
MD_INDICATOR VARCHAR2 (300 BYTE)
CORE_PK (CORE_ID)
CORE_PK (CORE_ID)
CYBER.CONTROLS
P * CONTROL_ID NUMBER (11)
* CONTROL VARCHAR2 (4000 BYTE)
* MINI NUMBER (1)
* STAND NUMBER (1)
* HIGH NUMBER (1)
* PROTOTYPE VARCHAR2 (30 BYTE)
SCOPE VARCHAR2 (30 BYTE)
DATA_CLASS VARCHAR2 (30 BYTE)
* ACTIVITY_CLASS VARCHAR2 (30 BYTE)
ENTE_INDICATOR VARCHAR2 (400 BYTE)
MD_INDICATOR VARCHAR2 (300 BYTE)
CONTROLS_PK (CONTROL_ID)
CONTROLS_PK (CONTROL_ID)

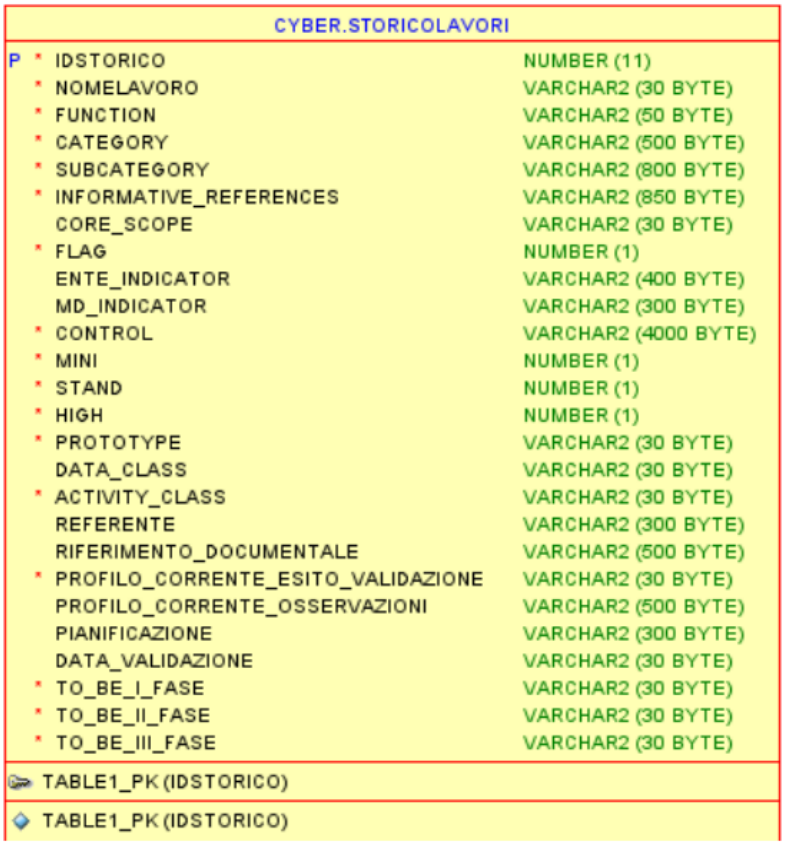


Figura 4.4. Modello relazionale database

Una volta impostata la progettazione logica, ho creato un prototipo di database utilizzando **XAMPP** come piattaforma locale di sviluppo. *"XAMPP è una distribuzione di Apache completamente gratuita e semplice da installare, contenente MySQL, PHP e Perl. Il pacchetto open source XAMPP è stato creato per essere estremamente facile da installare e utilizzare."* [5]. In sostanza, si tratta di un ambiente di sviluppo che fornisce un pacchetto integrato, comprendente Apache (per il server web), MariaDB (come sistema di gestione di database relazionali - RDBMS - che consente di organizzare e gestire dati in modo strutturato, precedentemente MySQL), PHP e Perl. Questo strumento è molto utile per testare il funzionamento del database in ambiente locale, in quanto consente di eseguire un server MySQL e di utilizzare **phpMyAdmin** per l'interfaccia di gestione del database. PhpMyAdmin è un'interfaccia web particolarmente pratica, che permette di gestire i database tramite un'interfaccia grafica, facilitando operazioni come la creazione di tabelle, l'inserimento e la manipolazione di dati e l'esecuzione di query

SQL senza necessità di scrivere codice complesso direttamente su un terminale SQL.
Questa impostazione iniziale ha permesso di testare il database e, quindi, di verificare il corretto funzionamento delle strutture e le relazioni tra le tabelle stesse. Per fare ciò, è stata effettuata una simulazione delle attività operative attraverso l'inserimento di dati di prova all'interno delle tabelle del db, le quali sono state poi interrogate per mezzo di query SQL. Le query provate sono state quattro, ognuna legata a una specifica fase del risk assessment:

- **Creazione** del profilo target;
- **Inserimento/modifica**, nella tabella StoricoLavori, delle informazioni relativo ad un perimetro di analisi;
- **Calcolo** del punteggio totale (target ideale);
- **Calcolo** della somma del punteggio ponderato ottenuto dai singoli controlli.

Fatto ciò, per poter verificare gli output delle query in maniera più strutturata e scalabile, è stata implementata un'interfaccia web in HTML integrato in PHP connessa con il db appena creato, permettendo dunque un'interazione con esso. Si tratta ovviamente di un'interfaccia molto basilare, finalizzata alla prova delle query sulla base dell'interazione dell'utente. Come IDE ho utilizzato Visual Studio Code. La prima pagina si presenta come segue:

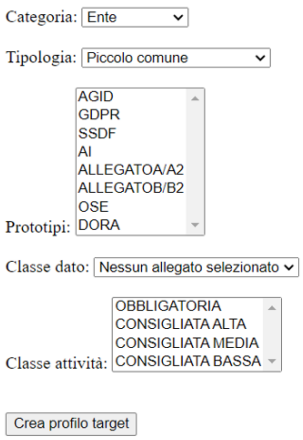


Figura 4.5. Interfaccia di prova

L'utente ha la possibilità di scegliere la Categoria, avendo due possibili alternative e cioè "Ente" o "Applicazione". Sulla base della Categoria scelta, il successivo menù a tendina, relativo alle tipologie, cambia le opzioni. Infatti, alla scelta della categoria "Ente" si ha la possibilità di scegliere tra le Tipologie "Piccolo Comune", "Pubblica Amministrazione" e "Infrastruttura Critica", mentre alla scelta della Categoria "Applicazione" le opzioni riguardanti le relative Tipologie cambiano e diventano "Applicazione Software" o "Medical Device". L'utente può ovviamente scegliere una sola categoria e una sola tipologia. Successivamente, si possono scegliere i prototipi ed è possibile effettuare una scelta multipla. Se saranno scelti anche i prototipi relativi agli allegati, il successivo menù a tendina "Classe dato" permetterà la scelta tra le opzioni "Ordinari", "Critici", "Strategici", altrimenti resterà non selezionabile. Infine, si sceglie la classe di attività. Alla pressione del button "Crea profilo target", partirà l'opportuna query al db.

Popolamento delle tabelle

Dopo aver verificato, attraverso l'inserimento di dati di prova, che la struttura del database soddisfacesse i requisiti progettuali e che le tabelle fossero funzionali, ho proceduto con la fase successiva del lavoro.
A questo punto, l'obiettivo era quello di riempire le tabelle con i dati reali che avrebbero costituito il cuore del sistema. Il processo di popolamento si è rivelato particolarmente articolato, poiché riguardava il riempimento delle tre tabelle principali del database, ovvero Core, Controlli e Montaggio, e l'obiettivo era quello di evitare di svolgere l'attività manualmente. Infatti, la grande mole di dati avrebbe reso quest'alternativa quasi impossibile a causa dell'ampia finestra di esposizione all'errore.
Data la dimensione dei dati e la necessità di assicurare un riempimento accurato e consistente delle tabelle, il popolamento è stato suddiviso in due fasi principali. PhpMyAdmin, l'interfaccia di gestione di database inclusa in XAMPP, consente, tra le tante cose, di caricare direttamente i dati nelle tabelle tramite file CSV. Questa funzionalità è stata cruciale per il processo, poiché consentiva di impostare i dati in un file CSV e importarli nel database, permettendo quindi un inserimento di dati efficiente e strutturato.
Nella prima fase del popolamento, mi sono concentrato sulle tabelle Core e Controlli. Questo passaggio ha richiesto un lavoro preliminare di raccolta

e organizzazione dei dati aziendali in un formato compatibile con le tabelle del database. Partendo dal know-how aziendale, mi è stata fornita una serie di documentazioni che ho studiato a fondo, analizzando le informazioni necessarie per la corretta configurazione delle tabelle. Ho quindi creato dei file Excel personalizzati per adattarli ai campi specifici delle tabelle Core e Controlli nel database, garantendo una perfetta corrispondenza tra i dati nel file e la struttura richiesta dalle tabelle. Una volta completata la personalizzazione dei file Excel, ho convertito ciascun file in formato CSV, che è uno dei formati compatibili con phpMyAdmin per il caricamento dei dati. Grazie a questa configurazione, ho importato i file CSV in phpMyAdmin e popolato rapidamente le tabelle Core e Controlli con dati reali, senza errori di corrispondenza o di struttura. Questo processo ha permesso di stabilire una base solida di dati, configurando gli elementi principali che sarebbero stati utilizzati nelle fasi successive.
Il popolamento della tabella Montaggio ha richiesto un approccio ancora più elaborato. Diversamente dalle tabelle Core e Controlli, la tabella Montaggio ha una funzione relazionale e contiene non solo un identificativo univoco (idMontaggio), ma anche due chiavi esterne (idCore e idControllo) che creano una relazione tra le tabelle Core e Controlli. Dato il ruolo fondamentale di questa tabella per stabilire i legami tra le altre entità, si è reso necessario un processo di popolamento più articolato.
Per affrontare questa sfida, ho sviluppato uno script Python con l'obiettivo di automatizzare la generazione di un file CSV adeguato per la tabella Montaggio. Lo script è stato progettato per leggere e processare un file Excel contenente i dati specifici relativi al profilo di montaggio e ai controlli associati ai rispettivi core. Il file Excel di input comprende il set completo di controlli per ogni funzione di sicurezza, montati sui core corrispondenti. Questo processo è stato ripetuto quattro volte, una per ciascuna delle funzioni principali, con l'eccezione delle funzioni di Respond e Recover che, essendo collegate, sono state elaborate congiuntamente. Lo script Python lavora in modo sequenziale: per ogni riga del file Excel di input, verifica i valori presenti e utilizza il contenuto delle colonne e certi criteri di disposizione dei dati per ottenere le chiavi primarie corrispondenti nel database già popolato. Per ogni entry, lo script recupera l'idCore in base al core specificato e, allo stesso modo, trova l'idControllo per il controllo associato. In questo modo, il

file CSV prodotto include direttamente i valori numerici che rappresentano le chiavi esterne necessarie per popolare correttamente la tabella Montaggio. Una volta generati i file CSV, sono stati importati in phpMyAdmin nella tabella Montaggio, completando così il popolamento di questa tabella relazionale.
Grazie a questo processo, le tre tabelle principali del database, Core, Controlli e Montaggio, sono state riempite con dati reali, costituendo una base dati robusta e pronta per la fase operativa dell'applicativo. Il popolamento automatizzato delle tabelle non solo ha ridotto il rischio di errore umano, ma ha anche accelerato significativamente il processo, consentendo di completare un'attività che altrimenti sarebbe stata lunga e prona ad errori se eseguita manualmente. Le tabelle ora contengono dati reali e verificati, che permettono al database di rispondere efficacemente alle esigenze operative dell'applicativo di security assessment, garantendo una struttura dati capace di supportare l'operatività dell'applicativo in sviluppo, gettando le basi per una gestione dati che sia scalabile, sicura e che permetta l'aggiornamento in tempo reale.
Dopo aver progettato e popolato le tabelle iniziali su XAMPP con l'ausilio di phpMyAdmin, si è reso necessario procedere con una migrazione dell'intero database verso l'ambiente **Oracle**, utilizzando **SQL Developer** come interfaccia principale per la gestione e il controllo delle query. Questa migrazione si è resa necessaria principalmente per ragioni aziendali: Oracle è una soluzione di database molto diffusa e consolidata all'interno dell'organizzazione, il che garantisce un maggiore supporto interno e facilità di integrazione con i sistemi esistenti. Grazie all'esperienza consolidata dell'azienda con Oracle, si poteva contare su risorse già disponibili e su competenze specifiche che sarebbero state utili per lo sviluppo e la manutenzione a lungo termine dell'applicativo. Oltre al supporto interno, Oracle offre funzionalità di gestione avanzate e una sicurezza migliorata rispetto alle soluzioni come XAMPP, che sono più indicate per ambienti di test e sviluppo preliminare.
*"**Oracle Database** è uno tra i più famosi software di database management system sviluppato da Oracle Corporation. Scritto in linguaggio C, fa parte dei cosiddetti RDBMS (Relational DataBase Management System), ovvero di sistemi di database basati sul modello relazionale affermatosi come standard di riferimento dei database a partire dagli anni 80 del XX secolo."* [6].

In sostanza, Oracle è una delle piattaforme più utilizzate in ambito enterprise per la gestione di database relazionali e offre numerose funzionalità che lo rendono adatto per applicazioni di dimensioni e criticità elevate. Tra i principali vantaggi, troviamo:

- **Scalabilità e performance**: Oracle è progettato per gestire grandi volumi di dati in modo efficiente, offrendo una capacità di scalabilità sia verticale che orizzontale. Ciò consente di adeguare il database a un crescente numero di dati e di utenti senza compromettere la performance, aspetto fondamentale per un applicativo di security assessment che deve poter crescere con l'aumentare dei requisiti operativi;

- **Sicurezza avanzata**: Oracle mette a disposizione strumenti di sicurezza avanzati, come la crittografia nativa, il controllo degli accessi basato sui ruoli e la gestione dei privilegi in modo granulare, che sono essenziali per garantire l'integrità e la riservatezza dei dati sensibili all'interno del database;

- **Integrazione con gli strumenti aziendali**: La migrazione a Oracle ha permesso una maggiore integrazione con gli strumenti e le infrastrutture già in uso in azienda, consentendo una gestione dei dati più centralizzata e una standardizzazione che semplifica la manutenzione e l'interoperabilità con altri sistemi;

- **Backup e ripristino avanzati**: Oracle offre anche opzioni di backup e ripristino avanzate che permettono di minimizzare i tempi di inattività in caso di incidenti o malfunzionamenti, garantendo che il sistema sia sempre disponibile e operativo.

Per quanto riguarda **SQL Developer**, è l'interfaccia grafica ufficiale di Oracle per la gestione dei database, utilizzata per eseguire query SQL, modellare dati e visualizzare le tabelle e le relazioni. Grazie a SQL Developer, è stato possibile eseguire un controllo accurato della migrazione dei dati, verificando che la struttura delle tabelle e i dati fossero correttamente trasferiti dal database MySQL su XAMPP a quello su Oracle. SQL Developer, inoltre, semplifica l'automazione di alcune attività ricorrenti, come il backup e la gestione dei dati, migliorando l'efficienza operativa.

L'attività di migrazione verso Oracle è stata notevolmente facilitata dal lavoro preparatorio svolto inizialmente su XAMPP. Il processo di progettazione delle tabelle, i test effettuati sui dati e l'organizzazione preliminare dell'architettura hanno reso la struttura del database già completa e pronta per essere migrata. La disponibilità di tabelle definite e testate ha permesso di affrontare la migrazione come un'attività di conversione, anziché dover riprogettare il database da zero, riducendo notevolmente il tempo e lo sforzo richiesti.
Come abbiamo già accennato, **XAMPP** è una soluzione di sviluppo locale che combina Apache (per il web server), MariaDB (o MySQL per il database) e PHP/Perl (per lo scripting lato server), il che lo rende uno strumento ideale per ambienti di test e prototipazione. **PhpMyAdmin**, in particolare, è un'interfaccia web per la gestione di database MySQL, utile per visualizzare le tabelle, eseguire query e monitorare l'andamento del database. Tuttavia, phpMyAdmin ha dei **limiti** in termini di sicurezza e funzionalità avanzate rispetto a Oracle, rendendolo meno adatto per ambienti di produzione che richiedono performance elevate e una gestione rigorosa della sicurezza.
La migrazione a Oracle è stata vantaggiosa non solo per i miglioramenti in termini di sicurezza e scalabilità, ma anche perché la compatibilità con SQL Developer ha permesso di facilitare il processo e di sfruttare pienamente tutte le funzionalità avanzate del database, garantendo un ambiente più robusto e performante per l'applicativo.

4.3 Un primo passo verso la scelta delle tecnologie

A questo punto del lavoro di tesi, è essenziale aprire una discussione su un tema centrale, che riveste un'importanza strategica nello sviluppo dell'applicativo: la scelta delle tecnologie di sviluppo, sia per il front-end che per il back-end. Nei prossimi due capitoli questa tematica verrà trattata con un'analisi dettagliata, in cui esaminerò le diverse opzioni disponibili, le ragioni alla base delle decisioni prese e i criteri specifici che hanno guidato la selezione delle tecnologie più adatte per raggiungere gli obiettivi di progetto.

Importanza di una scelta adeguata nelle tecnologie di sviluppo

Scegliere le tecnologie più appropriate per lo sviluppo di un applicativo di security assessment non è un processo banale. Al contrario, rappresenta una fase cruciale che può determinare il successo o il fallimento di un progetto, soprattutto in ambiti delicati come quello della Cybersecurity. La scelta tecnologica, infatti, influenza non solo l'esperienza dell'utente finale, ma anche l'efficienza, la scalabilità, la sicurezza, la manutenibilità e la capacità di rispondere ad eventuali sviluppi futuri. La questione si fa ancora più rilevante se si considera che ogni applicazione è strettamente legata al contesto operativo in cui si inserisce e alle esigenze specifiche del settore. In sostanza, adottare una tecnologia senza una valutazione adeguata può comportare conseguenze pesanti sul lungo periodo, rendendo l'applicativo obsoleto, vulnerabile o inadatto a soddisfare i requisiti attesi.
Una scelta adeguata delle tecnologie di sviluppo è indispensabile per raggiungere un equilibrio ottimale tra performance e affidabilità. In questo contesto, è stato adottato un approccio che ha preso in considerazione due parametri principali: i **requisiti funzionali** e i **requisiti di sicurezza**. Questi due parametri hanno costituito la bussola attraverso cui l'analisi costi-benefici ha guidato la scelta delle tecnologie finali, per garantire che il sistema possa rispondere efficacemente alle necessità del progetto.
Partendo dai **requisiti funzionali**, questi riguardano quelle caratteristiche che l'applicativo deve soddisfare per essere efficiente e utilizzabile. In particolare, i requisiti funzionali considerati in questa analisi includono aspetti come la velocità operativa, che è determinante in un contesto in cui la rapidità di accesso e di elaborazione dei dati deve supportare decisioni puntuali. Inoltre, un'applicazione come questa deve poter garantire la gestione di sessioni real-time, in modo che l'utente possa lavorare direttamente e senza ritardi sulle operazioni richieste. Questa funzionalità non solo migliora l'esperienza dell'utente, ma si traduce in un aumento della produttività operativa. Un altro requisito funzionale che è stato considerato riguarda la gestione locale del repository, ovvero la capacità di gestire dati e risorse in modo efficiente all'interno della stessa infrastruttura aziendale, senza doversi appoggiare a soluzioni esterne. Questa gestione locale rappresenta un aspetto importante non solo in termini di performance, ma anche di riservatezza, poiché consente all'azienda di mantenere il pieno controllo sui propri dati senza dipendere da

soluzioni esterne.
Accanto ai requisiti funzionali, un secondo elemento cardine per la scelta delle tecnologie è stato rappresentato dai **requisiti di sicurezza**. È stato necessario dare priorità a tecnologie che potessero garantire standard di sicurezza elevati, proteggendo non solo i dati aziendali ma anche le operazioni stesse da eventuali minacce. La sicurezza include molteplici aspetti, dalla protezione dei dati al controllo degli accessi, e ogni scelta tecnologica deve supportare misure di prevenzione contro attacchi informatici, accessi non autorizzati e vulnerabilità di sistema. Una scelta accurata delle tecnologie può fare la differenza tra un sistema sicuro e uno che espone l'azienda a rischi operativi. Inoltre, sono stati considerati anche i protocolli di cifratura e le pratiche di autenticazione. A tal proposito, è stato importante identificare tecnologie che offrissero librerie integrate per la sicurezza e che supportino la cifratura dei dati, evitando vulnerabilità di terze parti.

Considerazioni per i capitoli successivi

Come vedremo, questa fase si è conclusa con la selezione di tecnologie sia front-end che back-end che rispondono a criteri di efficienza, sicurezza e facilità di mantenimento nel tempo, riducendo i rischi legati a obsolescenza e aggiornamenti continui. Queste scelte permettono all'applicativo di essere scalabile, facilmente aggiornabile e soprattutto sicuro, ottimizzando le risorse aziendali e garantendo una piattaforma tecnologica solida. I prossimi capitoli entreranno nel dettaglio delle scelte effettuate per le tecnologie di sviluppo. Verranno esaminati i diversi framework e strumenti front-end e back-end presi in considerazione, discutendone e dimostrando i pro e contro in modo da fornire una visione completa delle ragioni che hanno motivato la scelta finale.

Capitolo 5

Scelta delle tecnologie

Questo capitolo rappresenta un punto cruciale del lavoro di tesi, in quanto affronta la selezione delle tecnologie di sviluppo front-end e back-end che meglio si adattano al contesto operativo del progetto. Questa scelta non è stata casuale, ma frutto di un'analisi approfondita delle principali opzioni disponibili, considerando vantaggi e svantaggi di ciascuna. Svolgere questo processo in modo dettagliato era indispensabile per garantire che le tecnologie adottate rispondessero pienamente ai requisiti identificati in precedenza, sia dal punto di vista funzionale, sia sotto il profilo della sicurezza. In questa parte del lavoro, verranno quindi esaminate le principali tecnologie di sviluppo disponibili sul mercato, mettendo in evidenza le loro caratteristiche distintive, i rispettivi punti di forza e le potenziali limitazioni, contestualizzando l'analisi al caso specifico dell'applicativo di Security Assessment. Questo confronto porterà a identificare le soluzioni tecnologiche più idonee sia per il front-end che per il back-end.
Una volta definite e giustificate le scelte tecnologiche, il capitolo accenna anche alla fase di sviluppo del modulo di Login, il primo componente realizzato per l'applicativo. È importante sottolineare questo aspetto, in quanto il risultato di questa fase ha rappresentato una sorta di banco di prova relativamente alle tecnologie scelte.

5.1 Front-end

In questa sezione, verranno analizzate le principali tecnologie di sviluppo front-end disponibili al giorno d'oggi. L'obiettivo è quello di fornire una panoramica delle opzioni più diffuse, valutandone i punti di forza e di debolezza per identificare quelle più adatte al caso specifico del progetto trattato in questa tesi.

5.1.1 Framework front-end: definizione e vantaggi

"Un framework front end è un insieme di strumenti, librerie e convenzioni che forniscono una struttura predefinita per lo sviluppo dell'interfaccia utente di un'applicazione web. È un'infrastruttura software che offre degli strumenti di sviluppo organizzati e semplificati per gli sviluppatori front-end, consentendo loro di creare interfacce utente efficienti, scalabili e coerenti." [7]. In sostanza, si tratta di un "kit" che semplifica e standardizza il processo di sviluppo, offrendo elementi già pronti come modelli, librerie di codice e soluzioni per risolvere problemi comuni. Questo permette anche di velocizzare lo sviluppo, evitando di dover partire da zero ogni volta.
Ecco alcuni vantaggi principali nell'utilizzo di framework front-end:

- **Velocità e produttività**: I framework offrono librerie di componenti già pronti, come pulsanti, menu e griglie, riducendo il tempo necessario per sviluppare l'interfaccia utente. Forniscono una struttura chiara e standardizzata, che facilita il lavoro degli sviluppatori e accelera il processo di sviluppo;

- **Manutenibilità e scalabilità**: Permettono di scrivere codice modulare e riutilizzabile, rendendo più semplice la gestione e l'espansione del progetto. Inoltre, i framework sono progettati per adattarsi facilmente a nuove esigenze e permettono l'implementazione di nuove funzionalità senza riscrivere il codice esistente;

- **Standardizzazione e coerenza**: Offrono convenzioni e best practices che aiutano a mantenere un design coerente e un codice uniforme, indipendentemente dal numero di sviluppatori coinvolti. La standardizzazione riduce il rischio di errori o incoerenze nel codice;

- **Efficienza nella risoluzione di problemi comuni**: Molti framework includono soluzioni integrate per problemi frequenti, come gestione di eventi, aggiornamenti del DOM e comunicazione con API, evitando di dover mettere il programmatore nelle condizioni di risolvere problemi che sono in realtà riconducibili a pattern comuni. Inoltre, i framework popolari sono supportati da comunità ampie e attive, che offrono risorse, documentazione e soluzioni a problemi comuni;

- **Prestazioni ottimizzate**: Molti framework utilizzano tecniche avanzate per ottimizzare il rendering del contenuto, migliorando le prestazioni delle applicazioni. Facilitano la creazione di interfacce responsive, adattabili a diverse dimensioni di schermo e dispositivi;

- **Compatibilità cross-browser**: I framework includono soluzioni che garantiscono che l'interfaccia utente funzioni correttamente su tutti i browser moderni, evitando problemi di compatibilità;

- **Integrazione con altri strumenti**: I framework front-end si integrano facilmente con altri strumenti e librerie, come gestori di stato, API di backend o framework di testing;

- **Riduzione del carico cognitivo**: Con molte complessità già gestite dal framework, gli sviluppatori possono concentrarsi di più sulla progettazione dell'esperienza utente piuttosto che su dettagli tecnici di basso livello.

5.1.2 Principali tecnologie di sviluppo front-end

Per introdurre l'analisi delle principali tecnologie di sviluppo front-end, è stata utile una risorsa estremamente importante e riconosciuta nell'ambito dello sviluppo JavaScript: il sito **State of JS**. Questo portale è una delle piattaforme di riferimento per chiunque voglia comprendere le tendenze, le preferenze e le statistiche relative all'ecosistema JavaScript, il linguaggio più diffuso per lo sviluppo front-end. In sostanza, State of JS è una sorta di *"sondaggio annuale degli sviluppatori sull'ecosistema JavaScript"* [7]. Ogni anno, il sito raccoglie e analizza dati forniti da migliaia di sviluppatori provenienti da tutto il mondo, offrendo una panoramica dettagliata dello stato

attuale delle tecnologie, librerie e framework legati a questo linguaggio. Il sito fornisce statistiche aggiornate su:

- **Framework e librerie front-end più popolari**, confrontando metriche come l'adozione, la soddisfazione e l'interesse da parte degli sviluppatori;
- **Tecnologie back-end collegate a JavaScript**, come Node.js;
- **Strumenti per lo sviluppo**, inclusi package manager, build tools e altre utility dell'ecosistema JavaScript;
- **Tendenze nel linguaggio JavaScript**, come l'adozione di nuove funzionalità del linguaggio o cambiamenti nelle preferenze.

Il sito State of JS è diventato una risorsa fondamentale per:

- **Monitorare l'evoluzione dell'ecosistema JavaScript**: Fornisce un'istantanea aggiornata delle tecnologie più rilevanti;
- **Orientare le scelte degli sviluppatori**: Aiuta i professionisti a selezionare strumenti e framework che abbiano un ampio supporto della comunità e un futuro promettente;
- **Capire le tendenze di mercato**: Le aziende possono usarlo per decidere quali tecnologie adottare nei loro progetti, in base a metriche di popolarità e soddisfazione.

Secondo i dati di State of JS, nell'ambito dello sviluppo front-end **React.js**, **Angular** e **Vue.js** si posizionano stabilmente tra i framework più utilizzati e apprezzati:

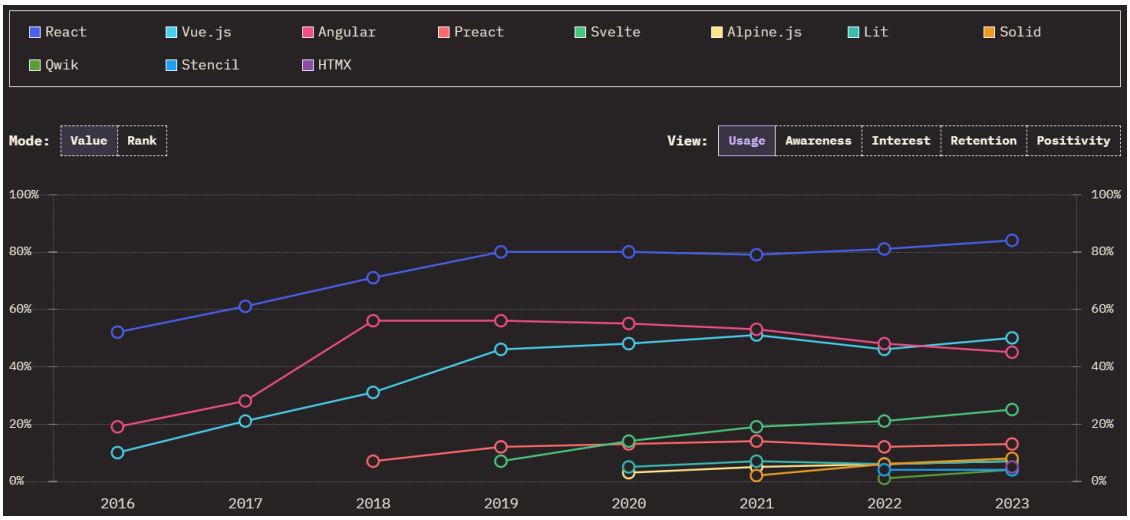


Figura 5.1. Statistiche di State of JS

Questi strumenti, dunque, rappresentano il podio dell'ecosistema front-end e sono le tecnologie che meritano un'analisi approfondita, sia per la loro diffusione, sia per l'impatto che hanno avuto nello sviluppo moderno.
In questa prima fase, concentriamoci sui loro aspetti **funzionali**.

React.js: definizione e aspetti funzionali

"React.js – comunemente noto come React – è una libreria JavaScript open-source, sviluppata da Facebook, anch'essa utilizzata, ovviamente, per la creazione di interfacce utente (UI) dinamiche e reattive. React è stato sviluppato da Facebook e viene ampiamente utilizzato per lo sviluppo di applicazioni web moderne" [7]. Esso si focalizza sulla creazione di componenti riutilizzabili e sulla gestione efficiente dello stato delle applicazioni, rendendo lo sviluppo front-end più modulare e scalabile.
Le **caratteristiche principali** di React.js sono le seguenti:

- **Architettura a componenti**: React segue un approccio basato sui componenti. Ogni parte dell'interfaccia utente, come un pulsante, una barra di navigazione o una sezione, è costruita come un componente autonomo. I componenti possono essere riutilizzati e combinati per creare strutture complesse;

- **Virtual DOM (Document Object Model)**: React utilizza un DOM virtuale (Document Object Model) per aggiornare l'interfaccia utente in modo più efficiente. Invece di manipolare direttamente il DOM reale, React crea una copia virtuale. Quando ci sono cambiamenti nello stato o nei dati, il Virtual DOM calcola le differenze con il precedente stato e aggiorna solo le parti del DOM reale che hanno subito modifiche. Questo approccio migliora le prestazioni e rende React particolarmente adatto per applicazioni complesse e interattive;
- **Data flow unidirezionale**: React utilizza un flusso di dati unidirezionale, dove le informazioni fluiscono dal genitore al figlio attraverso le props. Questo approccio semplifica la gestione dello stato e rende il comportamento dei componenti più prevedibile;
- **JSX (JavaScript XML)**: React utilizza JSX, un'estensione della sintassi JavaScript che consente di scrivere codice che sembra HTML direttamente all'interno di file JavaScript. JSX migliora la leggibilità e l'organizzazione del codice;
- **Gestione dello stato**: React offre due modi principali per gestire lo stato interno. Un modo prende il nome di **Hook useState** e permette di aggiungere stato a un componente funzionale. L'altro modo, invece, si riferisce a **Hook useReducer** ed è utile per gestire stati complessi;
- **React Hooks**: Li abbiamo nominati nel punto precedente. Consentono di utilizzare funzionalità avanzate, come lo stato e il ciclo di vita dei componenti funzionali, semplificando lo sviluppo;
- **React Router**: React non include funzionalità di routing di base. Tuttavia, la libreria React Router permette di gestire facilmente la navigazione e la costruzione di applicazioni a pagina singola (SPA);
- **Ecosistema e compatibilità**: React può essere utilizzato con diverse librerie e strumenti per ampliare le sue funzionalità.

React.js presenta i seguenti **vantaggi**:

- **Prestazioni elevate grazie al Virtual DOM**: L'uso del Virtual DOM riduce il costo delle operazioni di aggiornamento del DOM reale, migliorando la velocità delle applicazioni;

- **Componenti riutilizzabili**: La creazione di componenti modulari e indipendenti consente di riutilizzare il codice, migliorando la produttività e la manutenibilità;
- **Ampio ecosistema e supporto della community**: Essendo uno dei framework più utilizzati, React ha una community vasta e attiva, oltre a un'enorme quantità di risorse, plugin e librerie esterne disponibili;
- **Compatibilità cross-platform**: Con strumenti come React Native, il codice può essere riutilizzato per creare applicazioni mobili per iOS e Android, mantenendo la stessa base di sviluppo;
- **Facilità di apprendimento**: Per chi ha una conoscenza di base di JavaScript e HTML, React è relativamente semplice da imparare, grazie anche a una documentazione eccellente;
- **Supporto a lungo termine**: Essendo sviluppato e mantenuto da Meta (Facebook), React ha un supporto garantito e aggiornamenti regolari.

React.js presenta i seguenti **svantaggi**:

- **Richiede una curva di apprendimento iniziale**: Sebbene React sia semplice nei concetti di base, per padroneggiare completamente la libreria è necessario imparare concetti avanzati come il Virtual DOM, Hooks, Context API e JSX;
- **Necessità di configurazione**: React si concentra sull'interfaccia utente e non include altre funzionalità come il routing o la gestione globale dello stato. Per progetti complessi, sono richiesti strumenti aggiuntivi (es. Redux, React Router), che aumentano la complessità;
- **Aggiornamenti frequenti**: Essendo in continua evoluzione, gli sviluppatori devono aggiornare costantemente le loro conoscenze per stare al passo con le nuove versioni e funzionalità;
- **Forte flessibilità**: A differenza di framework come Angular, React non impone una struttura rigida. Questo dà molta flessibilità, il che può diventare uno svantaggio in quanto può portare a incoerenze nei team di sviluppo.

Angular: definizione e aspetti funzionali

"Angular è un framework front end open source sviluppato da Google. È basato sul linguaggio JavaScript e offre un'ampia gamma di funzionalità per semplificare lo sviluppo di applicazioni web complesse. Angular si è affermato come uno dei più popolari e completi framework front end" [7]. Esso è un framework di tipo full-stack. Offre un insieme di strumenti e una struttura ben definita per lo sviluppo di applicazioni web dinamiche, scalabili e robuste.
La prima versione di Angular era basata su JavaScript e seguiva un approccio Model-View-Controller (MVC). In seguito, Angular ha abbandonato JavaScript per **TypeScript** e ha adottato un'architettura basata su componenti.
Le **caratteristiche principali** di Angular sono le seguenti:

- **Basato su TypeScript**: Angular utilizza TypeScript, un super-set di JavaScript, che aggiunge funzionalità di tipizzazione statica e sviluppo orientato agli oggetti. TypeScript permette un miglior refactoring del codice, rilevamento di errori durante la fase di sviluppo e maggiore scalabilità per progetti complessi;

- **Architettura a componenti**: Angular divide un'applicazione in **moduli** (raggruppano funzionalità correlate) e **componenti** riutilizzabili (ogni componente è responsabile di una parte dell'interfaccia utente e del comportamento associato);

- **Two-Way Data Binding**: La sincronizzazione bidirezionale tra il modello (dati) e la vista (interfaccia utente) è uno dei punti di forza di Angular, garantendo che le modifiche ai dati siano riflesse automaticamente nell'interfaccia e viceversa;

- **Dependency Injection (DI)**: Angular include un sistema di iniezione delle dipendenze per semplificare la gestione dei servizi e ridurre le dipendenze dirette tra i componenti;

- **Router integrato**: Angular ha un sistema di routing potente e flessibile per la gestione della navigazione tra pagine e la configurazione di percorsi dinamici;

- **Reactive Forms e Template-Driven Forms**: Offre due approcci per la gestione dei moduli, dati dai Reactive forms, adatti per applicazioni complesse con un controllo completo, e dai Template-Driven Forms, per scenari semplici e dichiarativi;

- **Piattaforma multipiattaforma**: Angular consente lo sviluppo di **web app tradizionali**, **Progessive Web Apps (PWA)**, vale a dire applicazioni che funzionano offline e somigliano ad app native, e **app mobile native**.

Angular presenta i seguenti **vantaggi**:

- **Framework completo**: Angular offre tutto il necessario per sviluppare un'applicazione web complessa, senza dover dipendere da librerie esterne per funzionalità fondamentali come routing, gestione dello stato o moduli;

- **Community solida**: Essendo supportato da Google, Angular ha una vasta base di utenti e una community che garantisce aggiornamenti regolari e una buona documentazione;

- **Scalabilità**: L'approccio modulare e TypeScript rendono Angular ideale per progetti grandi e complessi;

- **Potente CLI (Command Line Interface)**: La CLI di Angular semplifica lo sviluppo e l'automazione di compiti ripetitivi, come la generazione di componenti, servizi e moduli;

- **Ottimizzazione per le prestazioni**: Angular offre funzionalità come il **lazy loading** (caricamento ritardato) e la compilazione **Ahead-of-Time** (AoT) per migliorare le prestazioni delle applicazioni.

Angular presenta i seguenti **svantaggi**:

- **Curva di apprendimento ripida**: Angular è un framework complesso con una quantità significativa di concetti e strumenti da apprendere, come TypeScript, RxJS (per la programmazione reattiva) e Dependency Injection;

- **Eccessiva complessità per progetti piccoli**: Per applicazioni semplici, Angular potrebbe risultare eccessivo rispetto a soluzioni più leggere come React o Vue.js;

- **Aggiornamenti frequenti**: Sebbene utili, gli aggiornamenti regolari di Angular possono richiedere lavoro aggiuntivo per mantenere il codice compatibile con le versioni più recenti.

Vue.js: definizione e aspetti funzionali

"Vue.js è un altro framework front end basato sul linguaggio JavaScript, open-source, e utilizzato per la creazione di interfacce utente (UI) reattive e scalabili. È, come i precedenti, progettato per semplificare lo sviluppo di applicazioni web complesse e fornisce un approccio incrementale e focalizzato sui componenti" [7]. Esso è progressivo e flessibile, noto per la sua semplicità, facilità di integrazione e capacità di crescere da un semplice strumento per la gestione della vista a un framework completo per applicazioni complesse. Le **caratteristiche principali** di Vue.js sono le seguenti:

- **Framework progressivo**: Vue può essere adottato gradualmente. Infatti, come già accennato, può essere usato come libreria per gestire solo la vista oppure può essere esteso con librerie ufficiali o di terze parti per coprire routing, gestione dello stato e altro;

- **Architettura a componenti**: Vue segue un'architettura basata su componenti, dove ogni parte dell'interfaccia utente è definita come un componente riutilizzabile;

- **Two-Way Data Binding**: Simile ad Angular, Vue offre un meccanismo di data binding bidirezionale, che sincronizza automaticamente i dati tra il modello e la vista;

- **Virtual DOM**: Vue utilizza un DOM virtuale per migliorare le prestazioni, rendendo il rendering più rapido ed efficiente;

- **Facilità d'uso e di integrazione**: Grazie alla sua leggerezza e semplicità, Vue può essere facilmente integrato in progetti esistenti senza doverli riscrivere da zero;

- **Struttura flessibile**: Vue non impone un'architettura rigida, rendendolo adatto a progetti di ogni dimensione. È particolarmente apprezzato per la sua capacità di adattarsi ai requisiti specifici di un'applicazione;
- **Supporto per TypeScript**: Sebbene scritto in JavaScript, Vue offre un ottimo supporto per TypeScript, rendendolo ideale per progetti moderni e scalabili;
- **Reattività**: Vue utilizza un sistema reattivo basato su osservatori, che permette di monitorare automaticamente i cambiamenti nei dati e aggiornare la vista di conseguenza.

Vue.js presenta i seguenti **vantaggi**:

- **Curva di apprendimento dolce**: Vue è noto per essere facile da imparare, anche per sviluppatori alle prime armi. Richiede solo una conoscenza base di HTML, CSS e JavaScript per iniziare;
- **Ecosistema snello ma completo**: Offre strumenti ufficiali per la gestione del routing, per la gestione dello stato globale, per creare e configurare rapidamente progetti Vue;
- **Comunità attiva e documentazione eccellente**: La documentazione di Vue è considerata una delle migliori nel panorama dei framework front-end e la community è ampia e fornisce risorse e librerie utili;
- **Flessibilità**: Può essere utilizzato sia come libreria leggera che come framework completo, a seconda delle esigenze del progetto;
- **Prestazioni elevate**: Grazie al Virtual DOM e al sistema reattivo, Vue è altamente performante, anche in applicazioni complesse;
- **Leggerezza**: Vue è più leggero rispetto ad Angular e React, rendendolo una scelta ideale per applicazioni dove il peso del framework è un fattore critico.

Vue.js presenta i seguenti **svantaggi**:

- **Minor adozione aziendale**: Rispetto ad Angular e React, Vue ha una penetrazione minore nel mondo enterprise, sebbene stia guadagnando terreno;

- **Comunità più piccola rispetto a React**: Anche se attiva, la community di Vue non è vasta quanto quella di React, il che significa meno risorse e plugin rispetto a React;
- **Forte flessibilità**: La mancanza di una struttura rigidamente definita può portare a soluzioni diverse nello stesso progetto e quindi a incoerenze, aumentando la complessità in team più grandi;
- **Limitata capacità per applicazioni molto grandi**: Anche se Vue è scalabile, alcune aziende preferiscono Angular o React per applicazioni su larga scala a causa della loro ampia adozione e supporto aziendale.

A questo punto, analizziamo queste tre tecnologie di sviluppo front-end dal punto di vista della **sicurezza**.

React.js: caratteristiche di sicurezza

React.js offre le seguenti **protezioni**:

- **Virtual DOM**: React, come abbiamo detto, utilizza un Virtual DOM, che riduce l'esposizione diretta al DOM reale. Ciò permette di mitigare alcuni vettori d'attacco;
- **Sicurezza da XSS (Cross-Site Scripting)**: Di default, React esegue l'escaping delle stringhe inserite nei componenti, proteggendo contro attacchi XSS. Un esempio può essere l'utilizzo dell'API *dangerouslySetInnerHTML*, anche se in questo caso lo sviluppatore deve comunque assicurarsi che i dati siano sanitizzati, poiché React non lo fa automaticamente;
- **Gestione sicura delle componenti dinamiche**: React non consente l'inclusione diretta di codice JavaScript eseguibile nelle proprietà degli elementi (ad esempio *onclick*), riducendo il rischio di attacchi.

Potenziali **vulnerabilità**:

- **Uso improprio di *dangerouslySetInnerHTML***: Se i dati non sono sanitizzati prima di essere passati all'API, l'applicazione può essere vulnerabile ad attacchi XSS. Si possono utilizzare librerie di sanitizzazione come DOMPurify o implementare una validazione rigorosa;

- **Dipendenze esterne**: Poiché React è un ecosistema flessibile, molte funzionalità di sicurezza dipendono dalle librerie scelte per routing, gestione dello stato, e così via. L'uso di librerie non mantenute o insicure può introdurre vulnerabilità;
- **CSRF (Cross-Site Request Forgery)**: React non offre protezioni integrate contro attacchi CSRF, quindi è necessario configurare token CSRF lato server.

Angular: caratteristiche di sicurezza

Angular offre le seguenti **protezioni**:

- **Protezione automatica da XSS**: Angular include una protezione di default contro XSS grazie al meccanismo di data binding e alla sanitizzazione automatica di qualsiasi contenuto dinamico. Le espressioni Angular sono automaticamente sanificate prima di essere inserite nel DOM;
- **Strict Template Security**: Angular applica restrizioni severe per i contenuti template. Ad esempio, non consente l'inclusione di codice JavaScript arbitrario o URL non fidati nei collegamenti;
- **Content Security Policy (CSP)**: Angular supporta l'implementazione di CSP, che limita l'esecuzione di codice non fidato e riduce i rischi associati a XSS;
- **Protezione contro Dependency Injection (DI) Attacks**: Angular utilizza un sistema sicuro di Dependency Injection, che riduce i rischi di attacchi derivanti da oggetti malevoli o configurazioni errate;
- **Protezione da CSRF e altre vulnerabilità comuni**: Angular può essere facilmente configurato per gestire token CSRF tramite l'interazione con API lato server;
- **HTTP Interceptors**: Gli HTTP Interceptors semplificano l'aggiunta di header di sicurezza (ad esempio, token di autenticazione o protezioni contro CSRF), rendendo più sicuro il flusso di comunicazione tra il front-end e il back-end;

- **Component-Based Authorization**: Angular consente un controllo granulare degli accessi basato sui componenti, permettendo di limitare la visibilità o l'accesso a specifiche sezioni dell'applicazione in base ai ruoli degli utenti.

Potenziali **vulnerabilità**:

- **Esecuzione arbitraria tramite *bypassSecurityTrust... API***: Angular consente agli sviluppatori di "fidarsi manualmente" di contenuti tramite funzioni come *bypassSecurityTrustHtml*. L'uso improprio di queste API può annullare le protezioni integrate;
- **Errori nella configurazione di CSP**: Sebbene supporti CSP, un'implementazione errata può introdurre falle di sicurezza.

Vue.js: caratteristiche di sicurezza

Vue.js offre le seguenti **protezioni**:

- **Sanitizzazione automatica dei dati**: Vue, come React, esegue il data binding e l'escaping automatico delle stringhe inserite nei template, riducendo il rischio di XSS;
- **Separazione della logica dalla vista**: L'approccio di Vue a componenti e data binding limita la possibilità di inclusione di codice JavaScript arbitrario;
- **Protezione da attacchi XSS e Injection**: Vue non consente l'esecuzione di script nei template, a meno che non venga esplicitamente autorizzato (ad esempio, tramite *v-html*).

Potenziali **vulnerabilità**:

- **Uso improprio di *v-html***: Simile a *dangerouslySetInnerHTML* in React, l'uso di *v-html* per inserire HTML dinamico può introdurre vulnerabilità XSS se i dati non sono correttamente sanitizzati;
- **Mancanza di protezioni avanzate integrate**: Rispetto ad Angular, Vue non offre soluzioni integrate per CSP o CSRF. Gli sviluppatori devono configurare manualmente queste protezioni tramite librerie o soluzioni lato server;

- **Dipendenze non mantenute**: L'ecosistema di Vue si basa su molte librerie di terze parti, che possono introdurre vulnerabilità se non adeguatamente mantenute.

5.1.3 Angular: la scelta front-end per l'applicativo

Sulla base di quanto discusso, va da sé che la tecnologia scelta lato front-end per l'applicativo da sviluppare è **Angular**. Di seguito, vengono contestualizzati al progetto in questione i punti esaminati:

- La struttura fortemente tipizzata e modulare garantisce una gestione chiara e organizzata del codice, fondamentale per un'applicazione come quella che si vuole sviluppare, che deve mantenere la scalabilità e consentire la suddivisione in diverse funzionalità;

- L'aggiornamento in tempo reale dei dati visualizzati e la gestione dinamica delle sessioni, funzionalità richieste dall'applicativo, sono supportate dal sistema di binding bidirezionale di Angular;

- Le Reactive Forms di Angular, con il loro sistema di validazione integrata, sono particolarmente utili per implementare funzionalità come il modulo di Login. Consentono di definire controlli robusti, con validazioni dinamiche che si adattano ai requisiti del progetto;

- La potenza e la rigidità strutturale di Angular, rispetto a framework più leggeri come Vue.js, consentono lo sviluppo di funzionalità avanzate, come la gestione di repository locali.

- Gli aspetti di sicurezza precedentemente elencati sono cruciali nel contesto dell'applicativo di Risk Assessment, che include dati sensibili e funzionalità di monitoraggio;

- Angular è una soluzione all-in-one, ideale per un progetto che richiede un framework robusto e completo. Rispetto a React, che si concentra solo sulla vista, Angular fornisce un ecosistema che include Routing Integrato, Dependency Injection (DI), CLI (Command Line Interface). In sostanza, non sono necessarie dipendenze esterne;

- Angular è sviluppato e mantenuto da Google, il che garantisce aggiornamenti regolari, sicurezza e un lungo ciclo di vita. Inoltre, gode di una comunità attiva che offre ampio supporto tecnico e risorse, fattore che riduce il rischio di incontrare blocchi tecnici durante lo sviluppo.

Angular è stato scelto **rispetto a React.js** in quanto è vero che quest'ultimo offre maggiore flessibilità, ma questa caratteristica può essere un'arma a doppio taglio in un contesto del genere, dove rigidità e coerenza sono necessarie. Angular, basandosi su pratiche consolidate, riduce la complessità.
Angular è stato scelto **rispetto a Vue.js** in quanto quest'ultimo è più leggero, ma meno adatto a progetti complessi e strutturati. L'applicativo in questione richiede funzionalità avanzate e un ecosistema completo, che Vue non può offrire al livello di Angular.

5.2 Back-end

Analogamente al lavoro svolto per il front-end, in questa sezione verranno analizzate le principali tecnologie di sviluppo back-end disponibili al giorno d'oggi, al fine di identificare quelle più adatte al caso specifico del progetto trattato in questa tesi.

5.2.1 Framework back-end: definizione e vantaggi

I framework back-end sono fondamentali per costruire il lato server di un'applicazione web. Offrono un'infrastruttura preconfigurata che semplifica attività comuni come la connessione ai database o la gestione delle autenticazioni. I principali vantaggi includono [8]:

- **Velocità nello sviluppo**: Forniscono componenti già pronti per compiti standard, riducendo i tempi, e promuovono l'uso di pattern collaudati, evitando errori e lavoro ridondante;

- **Scalabilità e prestazioni**: Integrano funzioni per gestire alti volumi di utenti senza cali di performance e facilitano la distribuzione su server multipli;

- **Struttura standardizzata**: La possibilità di seguire regole e layout comuni rende il lavoro in team più efficace, permettendo di formare con più facilità nuovi sviluppatori;
- **Focalizzazione sulla logica aziendale**: Automatizzano i compiti di routine, lasciando più tempo per lavorare sugli aspetti distintivi dell'app, offrendo soluzioni integrate per la sicurezza e la stabilità.

Nella scelta di un framework back-end, i principali aspetti da considerare includono [8]:

- **Prestazioni e velocità**: Capacità di gestire rapidamente molte richieste, ottimizzazione delle operazioni sui database, supporto per eseguire più operazioni in parallelo senza rallentamenti;
- **Facilità d'uso**: Codice intuitivo e ben documentato, presenza di strumenti e guide per velocizzare lo sviluppo;
- **Supporto della community**: Una community attiva garantisce accesso a risorse, soluzioni e aggiornamenti, disponibilità di plugin e librerie aggiuntive;
- **Scalabilità**: Capacità di crescere insieme al progetto, gestendo più utenti e dati, supporto per strategie come caching e distribuzione;
- **Sicurezza**: Difese integrate contro attacchi comuni, come SQL injection e XSS, meccanismi per validare dati e proteggere informazioni sensibili, crittografia sicura per il trasferimento di dati e gestione delle credenziali.

5.2.2 Principali tecnologie di sviluppo back-end

Per studiare le tecnologie e le tendenze back-end, una risorsa molto utile è stata **Daily.dev**.
Daily.dev è una piattaforma riconosciuta nel mondo dello sviluppo software che aggrega notizie e articoli tecnici per sviluppatori. È particolarmente utile per rimanere aggiornati su strumenti, tendenze, pratiche e novità riguardanti vari aspetti dello sviluppo software, tra cui front-end, back-end, DevOps e molto altro. Essa analizza l'ecosistema di sviluppo in base a metriche come popolarità, adozione da parte delle aziende e nuove funzionalità introdotte

nelle tecnologie.
Nello specifico, l'articolo *Top 10 Backend Frameworks [2024]* fornisce una panoramica aggiornata e dettagliata delle tecnologie e tendenze emergenti nel panorama dello sviluppo back-end. Questo articolo si basa su una combinazione di dati provenienti da esperti del settore, sondaggi e analisi delle preferenze della comunità degli sviluppatori. Nel contesto relativo allo sviluppo dell'applicativo, esso risulta essere un'ottima risorsa, in quanto offre dati utili per scegliere tecnologie che soddisfino requisiti specifici di progetto, come scalabilità, prestazioni e sicurezza, evidenziando amche l'importanza delle preferenze della community nello sviluppo di tecnologie back-end [8].
In questo articolo, viene offerta una panoramica dei 10 migliori framework back-end per il 2024, ponendo l'attenzione su performance, facilità d'uso, supporto della comunità, scalabilità e caratteristiche di sicurezza. Nel 2024, i tre framework più popolari sono **Django**, framework Python apprezzato per velocità ed efficienza, **Express.js**, leggero e basato su Node.js, eccellente per gestire operazioni simultanee, e **Spring Boot**, ideale per lo sviluppo rapido di applicazioni Java.

Django: definizione e aspetti funzionali

Django è un framework web open source scritto in Python, noto per la sua filosofia "batteries included", che fornisce una vasta gamma di strumenti pronti all'uso. È particolarmente adatto per lo sviluppo di applicazioni scalabili e sicure.
Le **caratteristiche principali** di Django sono le seguenti:

- **Linguaggio**: Basato su Python, noto per semplicità e leggibilità;
- **Architettura**: MVC (Model-View-Controller), per separare logica e presentazione;
- **Strumenti inclusi**: Gestione dell'autenticazione, ORM integrato, sistema di template;
- **Supporto alla scalabilità**: Funzioni di caching avanzate, compatibilità con cloud e database distribuiti;
- **Sicurezza**: Protezione integrata contro XSS, CSRF, SQL Injection e session fixation.

Django presenta i seguenti **vantaggi**:

- **Facilità d'uso**: Adatto ai principianti grazie a Python e a documentazione eccellente;
- **Ecosistema completo**: Include strumenti preconfigurati per autenticazione, database e routing;
- **Scalabilità**: Ideale per applicazioni di grandi dimensioni e scalabili;
- **Sicurezza robusta**: Molte misure di sicurezza integrate.

Django presenta i seguenti **svantaggi**:

- **Overhead iniziale**: Può essere pesante per progetti molto piccoli;
- **Curva di apprendimento avanzata**: Mentre è semplice iniziare, alcune funzionalità avanzate possono essere complesse;
- **Rigidità**: L'approccio "opinioni forti" di Django può limitare la flessibilità in alcuni casi. Quando si dice che Django adotta un approccio con "opinioni forti" ("strongly opinionated"), si intende che il framework ha delle convenzioni ben definite su come un'applicazione dovrebbe essere strutturata e sviluppata. Questo significa che Django fornisce un'architettura preconfigurata e impone una serie di decisioni di progettazione e sviluppo, come il modo in cui organizzare il codice, quali strumenti utilizzare e come implementarli.

Express.js: definizione e aspetti funzionali

Express.js è un framework minimalista per Node.js che offre una base leggera per creare applicazioni web server-side. È noto per la sua semplicità e flessibilità.
Le **caratteristiche principali** di Express.js sono le seguenti:

- **Linguaggio**: Basato su JavaScript;
- **Architettura**: Non impone una struttura rigida, consentendo personalizzazione completa;
- **Performance**: Ottimizzato per gestire molte richieste simultanee;

- **Modularità**: Supporta pacchetti esterni per aggiungere funzionalità;
- **Sicurezza**: Non ha funzionalità di sicurezza avanzate predefinite, ma è compatibile con moduli come Helmet.js per protezioni extra.

Express.js presenta i seguenti **vantaggi**:

- **Flessibilità**: Permette agli sviluppatori di configurare il framework a piacimento;
- **Prestazioni**: Ideale per applicazioni in tempo reale e microservizi;
- **Community**: Numerosi plugin e librerie disponibili grazie alla vasta comunità;
- **Compatibilità**: Può essere facilmente integrato con front-end moderni come React o Angular.

Express.js presenta i seguenti **svantaggi**:

- **Meno "batteries included"**: Mancanza di funzionalità integrate, come ORM o autenticazione;
- **Maggiore responsabilità per lo sviluppatore**: La flessibilità richiede una gestione manuale delle configurazioni;
- **Sicurezza base**: Richiede l'aggiunta di moduli esterni per protezioni solide.

Spring Boot: definizione e aspetti funzionali

Spring Boot è un framework basato su Java che semplifica lo sviluppo di applicazioni robuste e scalabili, grazie alla sua configurazione automatica e all'integrazione con l'ecosistema Spring.
Le **caratteristiche principali** di Spring Boot sono le seguenti:

- **Linguaggio**: Basato su Java, il linguaggio enterprise per eccellenza;
- **Architettura**: Supporta microservizi e applicazioni monolitiche;
- **Performance**: Configurazione automatizzata e integrazione nativa con strumenti di monitoraggio;

- **Scalabilità**: Ottimizzato per lavorare su ambienti cloud e server distribuiti;
- **Sicurezza**: Usa Spring Security per protezioni avanzate.

Spring Boot presenta i seguenti **vantaggi**:

- **Velocità di sviluppo**: Configurazione automatica per iniziare rapidamente;
- **Ecosistema ricco**: Ampia gamma di moduli per gestione di database, sicurezza e monitoraggio;
- **Sicurezza avanzata**: Include strumenti di crittografia e protezione da attacchi comuni;
- **Affidabilità enterprise**: Ideale per progetti aziendali di grandi dimensioni.

Spring Boot presenta i seguenti **svantaggi**:

- **Steep learning curve**: Richiede familiarità con Java e con l'ecosistema Spring;
- **Complessità**: Può risultare sovradimensionato per progetti semplici;
- **Consumo di risorse**: Ha requisiti di memoria e CPU più elevati rispetto a framework più leggeri.

Nell'esaminare i tre framework back-end, abbiamo accennato qualcosa anche in merito alla sicurezza. Riassumendo:

- **Django** offre protezioni predefinite contro vulnerabilità comuni come XSS, CSRF e SQL Injection. Le funzionalità di autenticazione e gestione delle sessioni sono robuste e facili da implementare;
- **Express.js** è meno sicuro "out of the box", ma può essere potenziato con moduli come Helmet.js e validator.js per gestire minacce comuni. La sicurezza dipende fortemente dall'implementazione dello sviluppatore;
- **Spring Boot** offre un livello di sicurezza elevato grazie all'integrazione con Spring Security. Fornisce strumenti avanzati per autenticazione, autorizzazione e protezione dei dati sensibili.

5.2.3 Spring Boot: la scelta back-end per l'applicativo

Arrivati a questo punto, abbiamo anche in questo caso la conferma che scegliere il framework da utilizzare lato back-end per lo sviluppo di un'applicazione web non è semplice ed è un punto estremamente importante.
Django è ideale per progetti che richiedono scalabilità, sicurezza integrata e uno sviluppo rapido, come piattaforme di e-commerce o siti complessi.
Express.js è perfetto per applicazioni leggere, API RESTful o progetti che richiedono massima personalizzazione e rapidità.
Spring Boot è adatto per applicazioni enterprise, sistemi finanziari e progetti che necessitano di alta affidabilità e scalabilità.
Alla luce delle considerazioni fatte, va da sé anche in questo caso che la tecnologia scelta lato back-end per l'applicativo da sviluppare è **Spring Boot**.
Innanzitutto, l'applicativo richiede una robusta infrastruttura back-end per gestire funzionalità critiche come autenticazione, gestione dei dati e scalabilità. Spring Boot si distingue per i seguenti aspetti:

Prestazioni e scalabilità

- **Gestione di carichi elevati**: Grazie all'architettura basata su **Spring Framework** e all'integrazione con tecnologie come Hibernate per l'accesso ai dati, Spring Boot è capace di supportare applicazioni ad alto traffico;

- **Scalabilità orizzontale e verticale**: Può essere facilmente distribuito su più server o adattato per gestire crescenti carichi di lavoro, grazie al supporto nativo per configurazioni di cloud computing come AWS, GCP o Azure;

- **Caching e prestazioni ottimizzate**: Spring Boot consente l'implementazione di caching avanzato, che riduce i tempi di risposta nelle richieste ripetute.

Modularità e flessibilità

- La struttura modulare di Spring Boot facilita l'implementazione di nuovi moduli senza compromettere la stabilità dell'intero sistema;

- È ideale per lo sviluppo di microservizi, un requisito sempre più comune per applicazioni moderne.

Sicurezza

Spring Boot include **Spring Security**, un modulo estremamente avanzato per la gestione della sicurezza:

- **Autenticazione e autorizzazione**: Supporta protocolli standard come OAuth2, JWT e LDAP, garantendo una gestione flessibile e sicura delle identità degli utenti;

- **Protezione contro attacchi comuni**: Prevenzione da CSRF (Cross-Site Request Forgery) attraverso token CSRF integrati e mitigazione delle vulnerabilità di XSS (Cross-Site Scripting) e SQL Injection grazie a validazioni preconfigurate e alla gestione avanzata delle query tramite ORM come Hibernate;

- **Crittografia e gestione sicura dei dati sensibili**: Supporta facilmente la crittografia dei dati sensibili in transito e a riposo, utilizzando standard di crittografia avanzati.

Cercando di contestualizzare ancora di più la scelta fatta con il progetto specifico in questione, Spring Boot risulta la scelta più adatta per i seguenti motivi:

- **Integrazione con il Modulo di Login**: Il modulo di Login beneficia delle capacità di gestione sicura delle sessioni offerte da Spring Boot. La libreria integrata consente la gestione dei token di accesso in modo sicuro e scalabile. Inoltre, per applicativi che devono rispettare normative come il GDPR, Spring Boot offre strumenti per la gestione trasparente e sicura dei dati personali;

- **Interoperabilità**: Spring Boot supporta l'integrazione con tecnologie front-end moderne come Angular, utilizzata nel progetto. Questa combinazione consente uno sviluppo coerente e fluido tra front-end e back-end;

- **Complessità dell'applicativo**: Il progetto richiede la gestione di un grande volume di dati e operazioni complesse, che possono essere gestite in modo ottimale grazie alla robustezza di Spring Boot. Inoltre, le funzionalità di monitoraggio e logging avanzate offerte da Spring Boot (es. Spring Actuator) sono essenziali per tracciare l'esecuzione e diagnosticare problemi in ambienti di produzione.

Sebbene **Django** e **Express.js** siano framework validi, Spring Boot offre vantaggi significativi in termini di sicurezza e scalabilità, particolarmente rilevanti per un'applicazione enterprise. Ad esempio, Django è molto potente, ma legato al linguaggio Python, che potrebbe non essere ottimale. Express.js è estremamente leggero e flessibile, ma la mancanza di una struttura predefinita e di funzionalità avanzate di sicurezza lo rende meno adatto per un'applicazione con requisiti stringenti come quella in questione.

5.3 Modulo di Login e prossimi passi

Dopo aver effettuato un'analisi approfondita e selezionato le tecnologie Angular per il front-end e Spring Boot per il back-end, lo sviluppo dell'applicativo è iniziato con la realizzazione del **modulo di Login**.
Questo modulo è stato sviluppato utilizzando le tecnologie citate nei capitoli precedenti e facendo uso di Visual Studio Code (VS Code) come ambiente di sviluppo integrato (IDE).
La creazione del modulo di Login ha rappresentato un passaggio fondamentale per il progetto, non solo per la sua funzione intrinseca nell'autenticazione degli utenti, ma anche per una verifica ulteriore relativa alle tecnologie selezionate.
Partiamo con il dire che la realizzazione di questo modulo ha permesso di:

- Testare le capacità di Angular nel gestire un'interfaccia utente interattiva e sicura, con particolare attenzione alla gestione di input utente e alla comunicazione con il back-end;

- Valutare la robustezza di Spring Boot nella gestione delle richieste di autenticazione, nell'implementazione della logica di sicurezza e nell'interazione con i database per la memorizzazione e la verifica delle credenziali;

- Valutare le caratteristiche di sicurezza offerte. Infatti, poiché il modulo di Login gestisce dati sensibili, come le credenziali degli utenti, è stato necessario implementare misure di sicurezza avanzate, come l'applicazione delle funzionalità di Spring Security per la prevenzione di attacchi comuni, quali ad esempio CSRF e XSS. Inoltre, è stato necessario anche validare i dati in ingresso per garantire l'integrità delle informazioni;
- Verificare l'interoperabilità tra front-end e back-end, nel nostro caso specifico vale a dire la sinergia tra Angular e Spring Boot, mettendo alla prova la comunicazione attraverso API REST, testando la velocità e l'affidabilità delle chiamate HTTP, e l'integrazione delle due tecnologie per gestire sessioni utente e flussi di autenticazione.

Oltre a ciò che abbiamo detto, però, la parte importante su cui porre l'attenzione, come accennato anche nella fase introduttiva di questo capitolo, è il ruolo strategico del modulo di Login come **banco di prova** in relazione alle tecnologie scelte. Infatti, come vedremo nel capitolo 6, l'attività è proseguita con la simulazione di una minaccia informatica per dimostrare da un punto di vista pratico la robustezza e l'efficacia delle tecnologie selezionate. Questa simulazione, effettuata in ambienti differenti, ha incluso, appunto, anche il modulo di Login sviluppato in questa fase, permettendo di validare le scelte effettuate per la costruzione dell'applicativo.

Capitolo 6

Simulazione della minaccia, POC (Proof of Concept) sperimentale e risultati ottenuti

In questo capitolo si intende dimostrare, attraverso un'analisi pratica e concreta, la solidità delle tecnologie scelte per la realizzazione dell'applicativo, con particolare attenzione alle loro caratteristiche di sicurezza. Come dimostrato nel capitolo 5, la scelta di **Angular** per il front-end e **Spring Boot** per il back-end è stata guidata da criteri di funzionalità, scalabilità e sicurezza. Questo capitolo, dunque, rappresenta un'estensione pratica di queste analisi e la validità di tali scelte sarà analizzata utilizzando un vettore d'attacco specifico: **Cross-Site Scripting (XSS)**.
La struttura del capitolo, dopo una parentesi relativa all'XSS, sarà suddivisa in due fasi operative:

- **Verifica della protezione contro XSS offerta da Angular**: Sarà esplorato il modo in cui il framework front-end gestisce l'input utente e protegge automaticamente contro l'iniezione di codice malevolo;
- **Valutazione della sicurezza di Spring Boot nella gestione lato server dei dati e delle richieste**: Si dimostrerà come il framework

protegge contro eventuali attacchi di XSS lato back-end.

6.1 Cross-Site Scripting: una minaccia diffusa per il web moderno

In questo paragrafo verrà analizzato il concetto di Cross-Site Scripting (XSS), una delle vulnerabilità più diffuse e insidiose nel panorama della sicurezza delle applicazioni web. Dopo aver esplorato questo aspetto, si discuterà il motivo per cui l'XSS è stato scelto come vettore d'attacco per la simulazione nel contesto del progetto in questione.

6.1.1 Panoramica

Il sito dell'OWASP ci dà molte informazioni relative all'XSS. *"Gli attacchi Cross-Site Scripting (XSS) sono un tipo di injection, in cui script dannosi vengono iniettati in siti web altrimenti benigni e affidabili. Gli attacchi XSS si verificano quando un attaccante usa un'applicazione web per inviare codice dannoso, generalmente sotto forma di script lato browser, ad un altro utente finale. Le falle che consentono il successo di questi attacchi sono piuttosto diffuse e si verificano ovunque un'applicazione web utilizzi l'input di un utente all'interno dell'output che genera senza convalidarlo o codificarlo."* [9]. In un attacco XSS, dunque, lo script dannoso viene eseguito nel browser della vittima, che non ha modo di distinguere il codice malevolo da quello legittimo. Poiché il browser considera il codice come proveniente da una fonte fidata, lo esegue, consentendo al malintenzionato di accedere a informazioni sensibili come cookie, token di sessione o altre credenziali memorizzate dal browser per il sito in questione. Inoltre, questi script possono essere utilizzati per modificare il contenuto delle pagine HTML, rendendo l'attacco ancora più invasivo e pericoloso [9].
Riassumendo, possiamo dire che gli attacchi di Cross-Site Scripting (XSS) si verificano quando:

1. Dati provenienti da una fonte non sicura, come una richiesta web, vengono introdotti in un'applicazione web;

2. Dati vengono inclusi in contenuti dinamici inviati agli utenti senza essere sottoposti a controlli per rilevare eventuali elementi pericolosi.

Il codice malevolo che raggiunge il browser dell'utente può essere scritto in JavaScript, ma anche in HTML, Flash o altri linguaggi eseguibili dal browser. La gamma di attacchi basati su XSS è estremamente ampia. Tuttavia, i più comuni includono:

- Il furto di dati sensibili, come cookie o informazioni di sessione, inviandoli all'attaccante;
- Il reindirizzamento della vittima verso siti web controllati dall'aggressore;
- L'esecuzione di azioni dannose sul computer dell'utente, mascherate come legittime dal sito web compromesso.

Questo rende gli attacchi XSS particolarmente insidiosi, poiché essi sfruttano la fiducia tra l'utente e il sito vulnerabile [9].

6.1.2 Tipi di XSS

Le principali tipologie di XSS, discusse ampiamente in un articolo dell'OWASP [10], sono tre e includono:

- **Reflected XSS**: La Reflected XSS, nota anche come Non Persistente o di Tipo I, si verifica quando i dati forniti dall'utente vengono immediatamente restituiti dall'applicazione web come parte di una risposta (ad esempio, in un messaggio di errore, nei risultati di una ricerca o in qualsiasi altro contenuto restituito). Questi dati vengono inclusi nella risposta senza essere adeguatamente sanificati per evitare problemi di sicurezza nel browser. Inoltre, i dati non vengono memorizzati permanentemente dal server, ma restano transitori, spesso non uscendo nemmeno dal contesto del browser dell'utente (DOM-based XSS);
- **Stored XSS**: La Stored XSS, nota anche come Persistente o di Tipo II, si verifica quando i dati forniti da un utente vengono salvati sul server di destinazione, ad esempio in un database, in un forum di discussione, nei registri dei visitatori o nei campi dei commenti. Successivamente, una

vittima accede a questi dati attraverso l'applicazione web, senza che tali informazioni vengano adeguatamente sanificate per garantire la sicurezza durante la visualizzazione nel browser. Con l'introduzione di tecnologie moderne come HTML5, è possibile immaginare uno scenario in cui il payload dell'attacco venga memorizzato direttamente nel browser della vittima (ad esempio, in un database HTML5), senza mai essere inviato al server;

- **DOM Based XSS**: La DOM Based XSS, nota anche come di tipo 0, è un tipo di attacco in cui il payload malevolo viene eseguito a seguito della modifica dell'ambiente DOM (Document Object Model) nel browser della vittima, che viene utilizzato dallo script client side originale, in modo che il codice lato client (Javascript o HTML) venga eseguito in modo "inaspettato", nel senso che viene indotto a comportarsi in modo inaspettato a causa delle manipolazioni maligne del DOM. A differenza di altre forme di XSS, in questo caso la pagina stessa (la risposta HTTP della pagina) non viene alterata, ma il codice lato client presente nella pagina agisce diversamente per via delle modifiche apportate al DOM durante l'esecuzione.

Per molto tempo, si è pensato che i tre tipi di XSS (Stored, Reflected e DOM Based) fossero categorie distinte. In realtà, questi tipi possono sovrapporsi: è possibile avere attacchi XSS basati su DOM sia memorizzati che riflessi. Allo stesso modo, possono verificarsi attacchi XSS non basati su DOM, ma memorizzati e riflessi. Tuttavia, questa classificazione può risultare confusa. Per semplificare e chiarire meglio i tipi di XSS esistenti, a partire dalla metà del 2012 la comunità di ricerca ha introdotto due nuove categorie per organizzare questi attacchi in modo più intuitivo:

- **Server XSS**: Gli attacchi di tipo Server XSS si verificano quando dati forniti da un utente non fidato vengono inclusi in una risposta HTTP generata dal server. Questi dati possono provenire direttamente dalla richiesta inviata al server o da una posizione memorizzata, come un database. Per questo motivo, è possibile avere sia attacchi di tipo Reflected Server XSS, sia di tipo Stored Server XSS.
  In questa tipologia di attacco, la vulnerabilità risiede interamente nel codice eseguito lato server. Il browser, invece, si limita a visualizzare

la risposta generata e ad eseguire eventuali script validi presenti nella risposta stessa;

- **Client XSS**: Gli attacchi di tipo Client XSS si verificano quando dati forniti da un utente non fidato vengono utilizzati per aggiornare il DOM con una chiamata JavaScript non sicura. Una chiamata JavaScript viene considerata non sicura se può essere sfruttata per introdurre codice JavaScript valido nel DOM. Questi dati possono provenire dal DOM stesso o essere inviati dal server (ad esempio tramite una chiamata AJAX o un caricamento di pagina). La fonte originale dei dati potrebbe essere una richiesta dell'utente o una posizione memorizzata sul client o sul server. Pertanto, è possibile avere sia Reflected Client XSS che Stored Client XSS.

Con queste nuove definizioni, il concetto di DOM Based XSS non cambia. Il DOM Based XSS è semplicemente una sottocategoria del Client XSS, dove la fonte dei dati si trova nel DOM, invece che provenire dal server.
Questa nuova distinzione aiuta a comprendere meglio dove si verifica l'attacco e come agisce.
La **misura di protezione** più efficace e semplice contro gli attacchi di tipo **Server XSS** è la codifica dell'output lato server, sensibile al contesto. Sebbene sia possibile utilizzare tecniche di validazione dell'input o sanitizzazione dei dati per ridurre il rischio di XSS lato server, queste strategie sono generalmente più complesse da implementare correttamente rispetto alla codifica dell'output sensibile al contesto.
La **migliore difesa**, invece, contro gli attacchi di tipo **Client XSS** consiste nell'utilizzo di API JavaScript sicure. Tuttavia, spesso gli sviluppatori non hanno una chiara consapevolezza di quali API JavaScript o metodi delle librerie siano effettivamente sicuri. Se si identifica un metodo JavaScript come non sicuro, la soluzione consigliata è adottare un metodo alternativo che garantisca maggiore sicurezza. Qualora non fosse possibile, un'altra strategia è applicare la codifica dell'output sensibile al contesto direttamente nel browser, prima di passare i dati al metodo JavaScript non sicuro. Questo approccio aiuta a mitigare i rischi derivanti da eventuali vulnerabilità.

6.1.3 Sintassi di XSS

Possiamo individuare differenti sintassi alternative per quanto riguarda l'XSS [9]:

- **Script negli attributi**: Gli attacchi XSS non sono necessariamente condotti tramite l'utilizzo dei tag *<script>...</script>*. Infatti, ci sono altri tag che fanno esattamente la stessa cosa, per esempio *<body onload=alert('test1')>* oppure altri attributi come *onmouseover*, *onerror*;
- **Script tramite schemi URI codificati**: Per nasconderci dai filtri delle applicazioni web, possiamo provare a codificare i caratteri delle stringhe, ad esempio *a=&\#X41* (UTF-8) e usarli nei tag *IMG*: *<IMG SRC=j&\#X41vascript:alert('test2')>*. Da notare che ci sono differenti notazioni di codifica UTF-8 che ci danno ancora più possibilità;
- **Codifica del codice**: Possiamo codificare lo script in base64 e metterlo nel tag *META*. Così facendo, non utilizziamo più *alert()*.

6.1.4 Perché la scelta di XSS?

La scelta di utilizzare il Cross-Site Scripting (XSS) come vettore d'attacco per le simulazioni effettuate è valida e giustificabile per diversi motivi, sia generali, sia specifici e relativi al progetto in questione.

Motivi generali

Di seguito, è presente un elenco delle motivazioni generali che consentono di giustificare la scelta del vettore d'attacco [11] [12]:

- **Diffusione dell'XSS**: L'XSS è uno dei problemi di sicurezza più comuni nel panorama delle applicazioni web. Secondo OWASP, si stima che circa due terzi delle applicazioni web siano vulnerabili a qualche forma di XSS. Questa vulnerabilità è presente in numerosi framework e tecnologie utilizzate per sviluppare applicazioni, rendendola un problema estremamente rilevante per sviluppatori e aziende;
- **Pericolosità e impatti**: L'XSS consente agli attaccanti di rubare sessioni, dirottare account, aggirare l'autenticazione a più fattori (MFA) o diffondere malware. La pericolosità aumenta con lo Stored XSS, che può

avere un impatto grave o critico. Riflette scenari di vita reale e fornisce una visione pratica di rischi significativi per la sicurezza;

- **Facilità di sfruttamento**: È un attacco relativamente semplice da eseguire, anche da parte di attori non particolarmente esperti. Questa semplicità, combinata con il fatto che strumenti automatici possono spesso rilevare e sfruttare vulnerabilità XSS, lo rende una minaccia pratica e concreta;
- **Evoluzione delle tecnologie e persistenza del problema**: Anche con l'introduzione di tecnologie moderne come React o Angular, il problema XSS non è completamente risolto. Questi framework possono avere lacune che richiedono ulteriori misure, come la codifica e la sanitizzazione dell'output.

Motivi specifici al progetto

Di seguito, è presente un elenco delle motivazioni relative al progetto che consentono di giustificare la scelta del vettore d'attacco:

- **Adattabilità al contesto simulativo**: L'XSS offre un'ampia gamma di scenari di attacco, da semplici riflessioni di input non sanitizzati a complesse modifiche basate sul DOM. Questo lo rende particolarmente adatto per simulazioni che vogliono dimostrare vari tipi di vulnerabilità;
- **Focus sulla prevenzione**: Scegliere l'XSS consente di illustrare strategie difensive facilmente applicabili, come la codifica contestuale o l'uso di API JavaScript sicure. Questo aspetto è particolarmente rilevante per un progetto che mira a educare o sensibilizzare sulle best practice di sicurezza [11] [12];
- **Rilevanza didattica**: Dato il suo impatto e la sua diffusione, simulare attacchi XSS permette di fornire un apprendimento pratico e contestualizzato su come gestire un rischio di sicurezza reale e significativo.

6.2 Robustezza di Angular: dimostrazione pratica

In questo paragrafo, si analizzerà la robustezza intrinseca di Angular come framework front-end, dimostrando la sua capacità di proteggere le applicazioni dagli attacchi XSS (Cross-Site Scripting) anche in assenza di protezioni lato back-end. Per fare ciò, è stata condotta una simulazione pratica che si è concentrata sul modulo di Login sviluppato e introdotto nel capitolo 5.

6.2.1 Parte operativa

Il modulo di Login, come previsto dalle scelte tecnologiche effettuate, è stato sviluppato utilizzando i framework Angular (lato front-end) e Spring Boot (lato back-end). In particolare, per dimostrare da un punto di vista pratico la robustezza di Angular si è deciso di rimuovere temporaneamente i controlli di sicurezza implementati lato back-end. Questa scelta ha consentito di isolare e testare esclusivamente le capacità di Angular nell'affrontare attacchi XSS, senza l'influenza delle ulteriori protezioni garantite dal back-end. È importante sottolineare che i framework back-end come Spring Boot, il linguaggio Java e alcune librerie specifiche che si possono utilizzare server-side offrono potenti meccanismi di difesa, quali l'escaping automatico dei dati e la validazione degli input lato server. Tuttavia, lo scopo di questa prima simulazione era quello di dimostrare la capacità intrinseca e automatica di Angular di fornire, da solo, una protezione significativa contro questi tipi di attacchi, grazie alla sua progettazione orientata alla sicurezza.
Come accennato nel capitolo 4, il modulo di Login è stato progettato per essere intuitivo e funzionale, includendo sia la possibilità di autenticarsi, sia quella di registrarsi come nuovi utenti. Tra gli elementi distintivi, è presente un pulsante dedicato alla registrazione, che attiva il modulo per la creazione di un nuovo account. Quando l'utente preme questo pulsante, il modulo presenta un form di registrazione strutturato in modo semplice e chiaro. Questo form è composto dai campi per l'inserimento delle informazioni richieste, quali nome utente, nome, cognome, e-mail, password e conferma di quest'ultima, corredati da controlli di validazione lato front-end per garantire l'aderenza dei dati a requisiti predefiniti (ad esempio, un indirizzo e-mail valido o una

password sufficientemente robusta). Il design del modulo è pensato per guidare l'utente attraverso un'esperienza fluida e sicura, evitando confusione o errori. L'immagine allegata illustra chiaramente l'interfaccia del form di registrazione e la disposizione dei suoi elementi:

Figura 6.1. Form di registrazione

L'attacco **XSS** è stato simulato utilizzando il form di registrazione appena descritto.
In particolare, l'attacco è stato progettato inserendo payload dannosi direttamente nei campi di input del modulo, con l'obiettivo di evidenziare il comportamento del framework di fronte a dati potenzialmente pericolosi. Il test si è concentrato sul campo relativo al nome utente, un campo che rappresenta un esempio tipico di input ricevuto direttamente dall'utente e successivamente processato dall'applicazione. Sono stati utilizzati payload di attacco costruiti con frammenti di codice JavaScript, come ad esempio *<script>alert("XSS");</script>*. L'idea alla base di questa simulazione è che, in assenza di meccanismi di protezione, il codice inserito nel campo del nome utente potrebbe essere interpretato e eseguito dal browser, causando potenzialmente effetti indesiderati, come l'esecuzione di script arbitrari. Il payload è stato testato sia durante la compilazione del form, sia nel momento della sua successiva elaborazione da parte del front-end, al fine di verificare eventuali falle.

Durante il test, tutti i campi del modulo sono stati compilati correttamente, con l'eccezione del campo relativo al nome utente, dove è stato intenzionalmente inserito un payload dannoso progettato per sfruttare una vulnerabilità di tipo XSS (Cross-Site Scripting). Come evidenziato nell'immagine seguente, il payload è stato iniettato specificamente nel campo dedicato al nome utente, con l'intenzione di verificare se il browser interpretasse il contenuto come script eseguibile o se invece il framework applicasse meccanismi di protezione per neutralizzare la minaccia:

Figura 6.2. Iniezione payload

A questo punto, è stato premuto il pulsante "Iscriviti".

6.2.2 Risultati e considerazioni

Premendo il pulsante "Iscriviti", l'utente viene correttamente registrato nel database, nonostante il nome utente contenga un payload malevolo. Questo succede in quanto, come abbiamo già anticipato, per questa simulazione sono stati disabilitati i controlli lato back-end. Questo è verificabile attraverso strumenti come SQL Developer, dove si può osservare che nel campo "USERNAME" è stato salvato l'intero payload, come mostrato nella seguente figura:

Figura 6.3. Memorizzazione payload database

Successivamente, effettuando il login con le credenziali appena registrate, si accede al componente successivo dell'applicativo, chiamato *Dashboard*. Nella navbar della Dashboard, viene mostrato il nome utente dell'utente autenticato. Tuttavia, invece di eseguire il payload (che potrebbe far apparire un alert con il testo "XSS"), Angular lo rende innocuo, visualizzandolo come semplice testo.
Il comportamento che si osserva, infatti, è il seguente.
Nel browser, visualizzando la pagina, si vede esattamente il nome utente di chi ha effettuato il login, nel nostro caso specifico *<script>alert("XSS")</script>*. Esso, però, si presenta come testo, non come un tag *<script>* che viene eseguito. Questo è il comportamento di **escaping** che Angular applica automaticamente. In altre parole, il testo HTML viene trattato come contenuto di testo, non come codice HTML o Javascript:

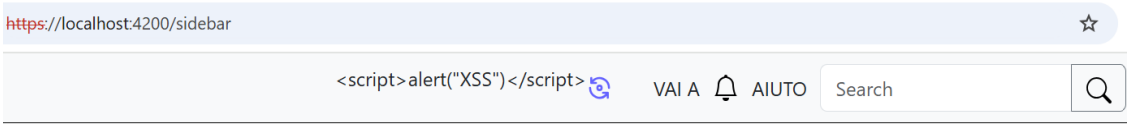


Figura 6.4. Utente loggato

A dimostrazione di quanto appena detto, aprendo gli strumenti di sviluppo e andando alla tab **"Ispeziona -> Elements"**, il browser mostra il DOM reindirizzato, che rappresenta il contenuto effettivo della pagina web come il browser la interpreta. Come si vede dalla seguente figura, il tag *<p>* contiene il testo con i caratteri di escape:

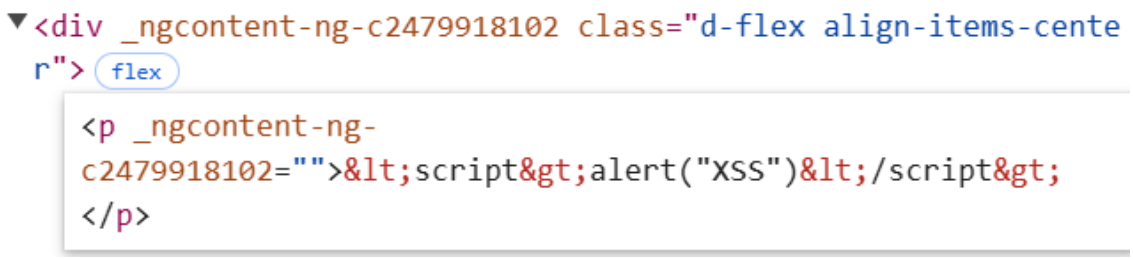


Figura 6.5. Escaping automatico di Angular

A questo punto, rendiamo il tutto un po' più formale. Le **protezioni intrinseche che offre Angular contro l'XSS** sono:

- **Escape contestuale automatico**: Angular applica automaticamente l'**escape** dei caratteri speciali quando visualizza dati nel DOM, trasformandoli in entità HTML sicure. Ad esempio, *< diventa &lt*, *> diventa &gt*, *' diventa &#39*. In questo modo, un input come *<script>alert("XSS");</script>* viene interpretato come stringa letterale e non come codice eseguibile. Questa protezione, come abbiamo visto, è verificabile ispezionando l'elemento DOM (via strumenti del browser), dove il nome utente risulta rappresentato con caratteri di escape [13] [14];
- **Data Binding sicuro**: Angular utilizza un sistema di **data binding** che protegge il DOM dai dati non attendibili. Gli sviluppatori interagiscono con le variabili nell'applicativo attraverso meccanismi controllati che impediscono manipolazioni dirette del DOM. Questo sistema protegge automaticamente le applicazioni da potenziali vulnerabilità, riducendo i rischi di errore umano [14] [15];
- **Sanitizzazione dei dati con whitelist**: Quando necessario, Angular applica un filtro sui contenuti per garantire che solo elementi e attributi HTML sicuri vengano resi nel DOM. Questo processo utilizza una "whitelist" di tag e attributi consentiti, eliminando tutto ciò che potrebbe essere pericoloso, come script o eventi (ad esempio, *onclick* o *onload*) [9].

È importante sottolineare il fatto che **non è una buona norma omettere i controlli di sicurezza lato back-end**. In questa simulazione, essi sono stati disabilitati per permetterci di concentrarci sulla potenza di Angular, ma

in generale la registrazione di un payload dannoso nel database rappresenta un rischio significativo in contesti diversi:

- Se i dati vengono recuperati e visualizzati in un front-end che non utilizza Angular (o un framework con protezioni simili), lo script potrebbe essere eseguito;
- In caso di accesso diretto ai dati attraverso API o altre interfacce, un payload XSS memorizzato potrebbe compromettere la sicurezza del sistema.

Le validazioni lato back-end, come quelle offerte da Spring Security, sono essenziali per impedire che dati pericolosi vengano salvati nel database. Senza queste protezioni, il payload XSS diventa uno "script stored", nello specifico **Stored XSS**, potenzialmente sfruttabile in altre parti del sistema. Inoltre, così facendo la sicurezza del sistema dipenderebbe unicamente dal front-end, aumentando il rischio in caso di integrazioni future con altri client. Infine, sebbene Angular protegga automaticamente il rendering del DOM, il payload XSS rimarrebbe comunque memorizzato nel database, rendendolo una vulnerabilità latente e, di conseguenza, rendendo quella di Angular una mitigazione temporanea. Questa vulnerabilità potrebbe essere sfruttata in altri contesti, come l'accesso diretto ai dati o l'uso di un framework diverso [16] [9].
In conclusione, possiamo dire che la simulazione effettuata dimostra come Angular fornisca un livello di protezione robusto contro gli attacchi XSS, trasformando potenziali minacce in semplice testo grazie a meccanismi intrinseci come l'escape contestuale, il data binding sicuro e la sanitizzazione dei dati. Tuttavia, la protezione completa dell'applicativo richiede l'integrazione di controlli lato back-end per impedire che payload dannosi vengano memorizzati o utilizzati in contesti non protetti.

6.3 Robustezza di Spring Boot: dimostrazione pratica

Per completare l'analisi della robustezza dei framework scelti per il nostro applicativo, è essenziale valutare il contributo offerto dal framework di back-end, Spring Boot, nella protezione contro vettori di attacco comuni e, anche

in questo caso, è stato utilizzato l'XSS (Cross-Site Scripting), nello specifico **Reflected XSS**.
Seguendo un approccio simile a quello adottato per dimostrare la sicurezza di Angular, ho deciso di creare un progetto parallelo a scopo dimostrativo, con l'obiettivo di concentrarmi esclusivamente sulle capacità di protezione del back-end. Il motivo di questa scelta è legato al fatto che nel modulo di Login sviluppato per l'applicativo principale, il front-end utilizza, appunto, Angular e, come dimostrato nel paragrafo precedente, questo include già di default un sistema di protezione contro l'XSS. Per questa ragione, non era possibile isolare e analizzare esclusivamente l'azione di Spring Boot contro questo vettore di attacco. Di conseguenza, è stato sviluppato un progetto parallelo estremamente semplificato, che ha permesso di mettere in evidenza il ruolo del back-end in un contesto privo di altre protezioni.
Il progetto di prova è stato concepito come segue:

1. **Front-end**: Un semplice **form di Login** sviluppato in puro **HTML**, senza l'utilizzo di framework front-end o librerie che possano introdurre protezioni aggiuntive contro l'XSS. Questo approccio garantisce un'interfaccia priva di sanitizzazioni o escape automatici dei dati;

2. **Back-end**: Il back-end è stato sviluppato utilizzando **Java** e il framework **Spring Boot**, con due configurazioni distinte:

    - Una prima versione **senza controlli** o protezioni esplicitamente configurate, per simulare gli effetti di un attacco XSS in un back-end privo di difese;
    - Una seconda versione con l'**integrazione dei controlli** di sicurezza, per dimostrare la protezione automatica fornita dal framework.

Questa simulazione si propone di:

- Dimostrare come un back-end senza protezioni sia vulnerabile ad attacchi XSS quando i dati pericolosi vengono elaborati o restituiti al client senza adeguati controlli;

- Mostrare come l'abilitazione delle protezioni possa mitigare o eliminare completamente tali vulnerabilità, garantendo un ulteriore livello di sicurezza indipendente dal front-end.

Questo progetto parallelo consente, dunque, di isolare e valutare il ruolo fondamentale del back-end nella sicurezza complessiva dell'applicativo, offrendo una visione più chiara delle difese intrinseche fornite server side.

6.3.1 Parte operativa, risultati e considerazioni

L'applicazione di prova è stata progettata per integrare front-end e back-end utilizzando Thymeleaf, semplificando l'architettura.
Andando più nello specifico, il **front-end** è composto da due pagine HTML:

- *index.html*: Un modulo di Login che invia i dati al back-end tramite POST;
- *welcome.html*: Una pagina di benvenuto che visualizza dinamicamente il nome dell'utente autenticato.

L'integrazione di front-end e back-end è stata realizzata tramite **Thymeleaf**, un motore di template per applicazioni Java che genera contenuti HTML dinamici. Grazie a Thymeleaf, il back-end gestisce direttamente la logica di rendering, evitando la necessità di un server separato per il front-end [17] [18].
Per quanto rigaurda il **back-end**, esso include:

- *DemoApplication.java*: Punto di avvio dell'applicazione;
- *User.java*: Una classe modello semplice con campi *username* e *password*;
- *LoginController.java*: Gestisce le richieste del modulo di login e passa i dati al template *welcome.html*.

A questo punto, concentriamoci sulle due simulazioni effettuate.
In entrambi i casi, ho utilizzato il payload malevolo *<script>alert("XSS");</script>*, iniettandolo nel campo "Username" e osservando il comportamento del sistema.
Inoltre, ho scelto di utilizzare *th:utext* invece di *th:text* per renderizzare il contenuto dinamico nel template HTML. La differenza principale tra questi due attributi di Thymeleaf è che *th:text* applica un escaping automatico del contenuto, garantendo che i caratteri speciali (come <, >, &, ecc.) vengano trattati come testo normale e non come codice HTML o JavaScript eseguibile. Al contrario, *th:utext* non esegue alcun escaping e permette l'inserimento

di HTML "raw" (grezzo) nel DOM, il che significa che eventuali tag HTML o script potrebbero essere eseguiti nel browser se l'input non viene opportunamente sanificato. La scelta di utilizzare *th:utext* è stata dettata dall'esigenza di concentrarsi esclusivamente sulla sanificazione dell'input lato back-end, piuttosto che fare affidamento su Thymeleaf per l'escaping automatico.
Nella prima simulazione, vale a dire la **vulnerabilità XSS senza sanificazione**, il back-end accetta i dati degli utenti senza, appunto, alcuna sanificazione. Questi dati vengono inviati al back-end e successivamente visualizzati nella pagina di benvenuto [19]. La seguente figura mostra il risultato di tali passaggi:

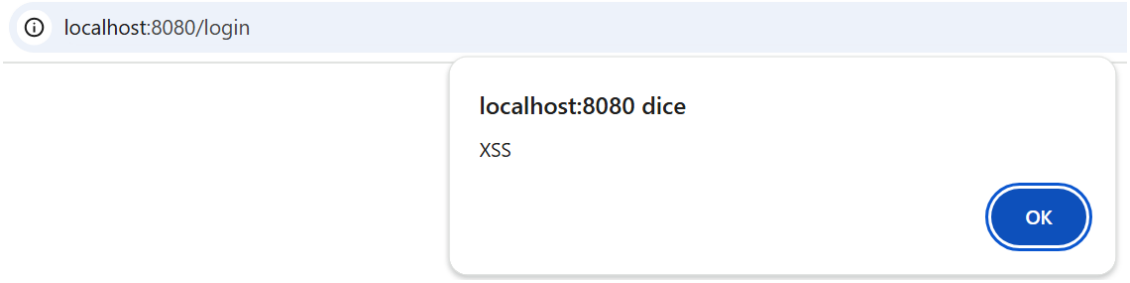


Figura 6.6. XSS riuscito

Nella seconda simulazione, vale a dire la **vulnerabilità XSS con sanificazione**, è stata introdotta una sanificazione lato back-end utilizzando *HtmlUtils.htmlEscape*, una funzione offerta nativamente da Spring Framework. Questa funzione codifica i caratteri HTML speciali (come <, >, &), trasformandoli nelle corrispondenti entità sicure. La modifica chiave nel controller è stata *String sanitizedUsername = HtmlUtils.htmlEscape(username);* . Dopo la sanificazione, il valore sanificato viene passato alla vista, eliminando qualsiasi rischio di XSS anche al di fuori del contesto di Thymeleaf.
HtmlUtils.htmlEscape è una funzione della classe *HtmlUtils* di Spring Framework. Questa classe è parte del pacchetto *org.springframework.web.util*, incluso automaticamente in ogni progetto Spring Boot, senza la necessità di dipendenze esterne. È quindi una soluzione nativa e affidabile per garantire la sicurezza contro XSS. [20] [21]. La seguente figura mostra il risultato di tali passaggi:

Figura 6.7. XSS bloccato

Oltre a *HtmlUtils.htmlEscape*, Spring Framework e Spring Security offrono altre soluzioni per proteggere contro vulnerabilità [22] [23]:

- **Custom SecurityConfig**: Essa offre protezione generale dell'applicazione. Infatti, la classe *SecurityConfig* protegge l'applicazione a livello globale, non solo per XSS (aggiungendo una *Content-Security-Policy* (CSP) agli header HTTP per prevenire l'esecuzione di script non autorizzati), ma anche contro altri tipi di vulnerabilità, come il clickjacking (con *X-Frame-Options*), lo sniffing dei contenuti (con *X-Content-Type-Options*) e altre vulnerabilità HTTP;
- **Sanificazione avanzata con altre librerie di Spring Security**: Oltre all'escaping HTML, Spring Security offre filtri preconfigurati per bloccare richieste pericolose o manipolare l'input dell'utente in modo sicuro.

Capitolo 7

Conclusioni e sviluppi futuri

Il lavoro di tesi ha permesso di affrontare e risolvere diverse sfide nel campo dell'ingegnerizzazione di sistemi e della relativa attenzione al concetto di sicurezza informatica. Nello specifico, grazie a questo progetto sono state gettate le basi per la realizzazione di un applicativo che automatizza un servizio di Risk Assessment, integrando funzionalità avanzate per offrire una visione real-time dell'impatto delle modifiche sulla postura di sicurezza di un'organizzazione. Questo strumento rappresenta un importante passo avanti. Infatti, permette di velocizzare le attività aziendali da un punto di vista pratico, rendendole meno prone ad errori, garantisce una gestione interna migliore delle informazioni di sicurezza e offre un valore aggiunto ai clienti finali. Grazie al lavoro svolto, è stato messo a punto il modulo di Login. Questo componente è stato sviluppato con un approccio ingegneristico, che pone le fondamenta per futuri ampliamenti funzionali e integrazioni.
I **risultati** raggiunti sono tangibili nell'ingegnerizzazione di un modulo di Login sicuro, che rappresenta un primo passo verso la creazione di un ecosistema integrato che possa permettere l'automazione del servizio di Risk Assessment che l'azienda offre, ad oggi, manualmente. La scelta delle tecnologie si è basata su criteri di affidabilità, scalabilità e conformità agli standard di sicurezza. L'implementazione è stata supportata anche da dimostrazioni pratiche che ne validano la robustezza. Infatti, le tecnologie adottate hanno mostrato di poter soddisfare le esigenze di un sistema orientato alla sicurezza. Ad esempio,

è stata dimostrata l'efficacia delle configurazioni di protezione predefinite e personalizzate nella prevenzione di attacchi comuni, come l'XSS. Le dimostrazioni pratiche hanno evidenziato come il modulo reagisca in scenari di test realistici, sottolineando l'importanza di scelte tecnologiche mirate.
Nonostante i risultati ottenuti, il lavoro presenta alcune **limitazioni** che aprono la strada a futuri sviluppi:

- **Funzionalità limitate al Login**: Al momento, il progetto si concentra esclusivamente sull'autenticazione, lasciando agli sviluppi futuri aspetti quali sviluppo degli altri moduli applicativi, autorizzazione granulare, gestione delle sessioni;

- **Integrazione con sistemi di monitoraggio**: Non sono ancora presenti connessioni con strumenti di monitoraggio e risposta agli incidenti, che rappresentano un aspetto cruciale in un ecosistema completo di sicurezza;

- **Scalabilità limitata**: Essendo un prototipo, il modulo non è stato ottimizzato per ambienti con un numero elevato di utenti simultanei o con richieste distribuite su più server.

Gli **sviluppi futuri** possono espandere il lavoro attuale:

- **Estensione delle funzionalità**: Sviluppare i moduli successivi, integrando la gestione delle autorizzazioni per controllare l'accesso a risorse diverse sulla base dei ruoli utente e aggiungendo funzionalità per l'audit delle attività degli utenti, garantendo tracciabilità e conformità normativa;

- **Automazione avanzata**: Implementare algoritmi per rilevare automaticamente tentativi di accesso sospetti o non autorizzati e automatizzare la generazione di report sulla sicurezza per fornire analisi periodiche agli amministratori;

- **Integrazione con altri sistemi**: Collegare il modulo a strumenti SIEM (Security Information and Event Management) per analisi approfondite degli eventi di sicurezza e sviluppare API che permettano di integrare il modulo con piattaforme di gestione identità e accessi (IAM);

- **Interfaccia utente migliorata**: Creare dashboard per amministratori e utenti finali, che offrano una visione chiara degli accessi e delle eventuali anomalie;
- **Promozione della consapevolezza**: Realizzare guide e materiali educativi basati sulle esperienze del progetto, evidenziando l'importanza di un approccio strutturato alla sicurezza.

Come abbiamo già accennato, oltre agli obiettivi tecnici il progetto mira a lasciare un impatto positivo a lungo termine, sia a livello funzionale che culturale:

- **Funzionale**: Ingegnerizzare un sistema che sarà patrimonio dell'azienda e che darà agli operatori e al cliente finale, tra le tante cose, una visione real-time relativa al cambiamento della postura di sicurezza dell'organizzazione in esame. Da questo punto di vista, dunque, lo scopo è quello di migliorare le capacità operative aziendali e garantire la protezione degli asset digitali;
- **Culturale**: Questo progetto sottolinea che la sicurezza non è un'opzione, ma una necessità strategica. Esso, dunque, contribuisce alla disseminazione della cultura di sicurezza informatica, sia per lo scopo che ha l'applicativo stesso, sia per l'attenzione avuta nei confronti della Cybersecurity, promuovendo la necessità di affrontare la sicurezza in ogni fase dello sviluppo software. Sensibilizzare le organizzazioni e, in generale, le persone su questo tema significa contribuire a un futuro più consapevole e resiliente, dove la sicurezza informatica è parte integrante della strategia aziendale.

In sintesi, il lavoro di tesi non si limita a risolvere un problema tecnico, ma mira a creare un impatto più ampio e duraturo. Le sfide affrontate e i risultati ottenuti costituiscono una base solida per lo sviluppo futuro, non solo dal punto di vista funzionale, ma anche come contributo alla cultura della sicurezza. Questo progetto dimostra come la tecnologia possa essere uno strumento potente non solo per migliorare i processi, ma anche per diffondere consapevolezza e responsabilità, contribuendo a creare un futuro più sicuro e resiliente.

Bibliografia

[1] National Institute of Standards and Technology (2024) "The NIST Cybersecurity Framework (CSF) 2.0." (National Institute of Standards and Technology, Gaithersburg, MD), NIST Cybersecurity White Paper (CSWP) NIST CSWP 29. DOI: 10.6028/NIST.CSWP.29

[2] Baldoni, R. and Montanari, L. (2016) "2015 Italian Cyber Security Report - Un Framework Nazionale per la Cyber Security" Available at: https://www.cybersecurityframework.it/sites/default/files/CSR2015_web.pdf

[3] "Framework Nazionale per la Cybersecurity e la Data Protection" Available at: https://www.cybersecurityframework.it/framework2

[4] Angelini, M. et al. (2019) "Framework Nazionale per la Cybersecurity e la Data Protection." Available at: https://www.cybersecurityframework.it/sites/default/files/framework2/Framework_nazionale_cybersecurity_data_protection.pdf Versione 2.0

[5] "XAMPP installers and downloads for Apache Friends." (n.d.). Available at: https://www.apachefriends.org/it/index.html

[6] contributori di Wikipedia. (2024, September 24). Oracle Database. Wikipedia. Available at: https://it.wikipedia.org/wiki/Oracle_Database

[7] Aulab. (2023, September 26). Come imparare a programmare: la guida ufficiale. Available at: https://aulab.it/i-migliori-framework-front-end

[8] Top 10 Backend Frameworks [2024]. (2024, July 2). Available at: https://daily.dev/blog/top-10-backend-frameworks-2024

[9] KirstenS (no date) Cross site scripting (XSS), Cross Site Scripting (XSS) | OWASP Foundation. Available at: https://owasp.org/www-community/attacks/xss/

[10] Types of XSS | OWASP Foundation. (n.d.). Available at: https://owasp.org/www-community/Types_of_Cross-Site_Scripting

[11] Cross Site Scripting Prevention - OWASP Cheat Sheet series. (n.d.). Available at: https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

[12] OWASP Top Ten 2017 | A7:2017-Cross-Site Scripting (XSS) | OWASP Foundation. (n.d.). Available at: https://owasp.org/www-project-top-ten/2017/A7_2017-Cross-Site_Scripting_(XSS)

[13] XSS in Angular and AngularJS - payloads all the things. (n.d.). Available at: https://swisskyrepo.github.io/PayloadsAllTheThings/XSS%20Injection/5%20-%20XSS%20in%20Angular/

[14] Chintanonweb. (2023, September 6). Angular security: Best practices for securing your Angular apps. DEV Community. Available at: https://dev.to/chintanonweb/angular-security-best-practices-for-securing-your-angular-apps-1pea

[15] Preventing XSS in angular. (n.d.). Available at: https://pragmaticwebsecurity.com/articles/spasecurity/angular-xss.html

[16] Spring Security. (n.d.). Spring Security. Available at: https://spring.io/projects/spring-security

[17] Tutorial: Thymeleaf + Spring. (n.d.). Available at: https://www.thymeleaf.org/doc/tutorials/3.0/thymeleafspring.html

[18] Getting started with the Standard dialects in 5 minutes - Thymeleaf. (n.d.). Available at: https://www.thymeleaf.org/doc/articles/standarddialect5minutes.html

[19] Getting started | Securing a web application. (n.d.). Getting Started | Securing a Web Application. Available at: https://spring.io/guides/gs/securing-web

[20] HTMlUtils (Spring Framework 6.2.0 API). (n.d.). Available at: https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/web/util/HtmlUtils.html

[21] Cross Site Scripting Prevention - OWASP Cheat Sheet series. (n.d.-b). Available at: `https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html`

[22] Security HTTP response headers:: Spring Security. (n.d.). Available at: `https://docs.spring.io/spring-security/reference/features/exploits/headers.html#headers-csp`

[23] Navuluri, B. (no date) Content security policy with spring security, Baeldung. Available at: `https://www.baeldung.com/spring-security-csp`