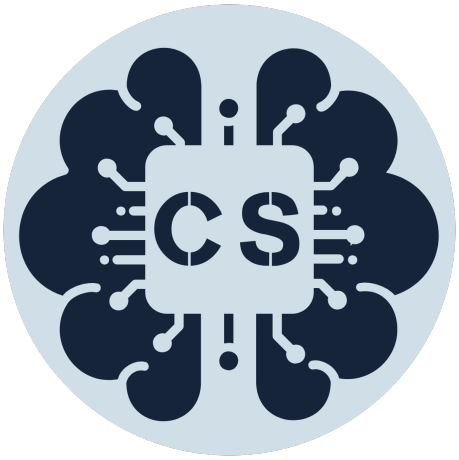
Specifica Tecnica

CyberSorcerers Team



Membri del team:
Sabrina Caniato
Giulia Dentone
Nicola Lazzarin
Giovanni Moretti
Andrea Rezzi
Samuele Vignotto

Informazioni sul documento		
Destinatari:	Prof Tullio Vardanega	Prof Riccardo Cardin
G al pedice:	Consultare il Glossario	

Registro dei Cambiamenti - Changelog$_G$

0.2.0	23/05/2024	Samuele Vignotto	Giovanni Moretti	Stesura della sezione 'Requisiti soddisfatti'.
0.1.1	23/05/2024	Giulia Dentone	Sabrina Caniato	Stesura della sezione 'Tecnologie'.
0.0.1	03/05/2024	Giulia Dentone	Samuele Vignotto	Definizione struttura del documento e scheletro delle sezioni. Scrittura introduzione ed obiettivi delle diverse sezioni.

Contents

1 Introduzione

1.1 Scopo del documento

Questo documento ha lo scopo di delineare e giustificare le decisioni architetturali prese durante le fasi di progettazione e sviluppo del prodotto. Sono presentati i diagrammi dei componenti React e dei pacchetti per illustrare le scelte dei pattern architetturali adottati per realizzare la struttura finale del prodotto. Inoltre, viene fornita una sezione dedicata ai requisiti soddisfatti dal team, offrendo così una panoramica completa dello stato di avanzamento del lavoro.

1.2 Scopo del prodotto

L'azienda proponente ha richiesto la creazione di una web app$_G$ che, tramite l'uso di IA$_G$ (in questo caso ChatGPT4 e Bedrock) è in grado di creare epic user stories$_G$ a partire dalle richieste del cliente e confrontarle con il codice sviluppato in modo da informare il cliente dello stato di avanzamento dello sviluppo del prodotto. Inoltre deve essere possibile, sia per il Project Manager$_G$, sia per il cliente rilasciare dei feedback (nel primo caso riguardanti l'adeguatezza delle stories, nel secondo caso riguardanti il prodotto finale) al fine di migliorare l'IA$_G$. È inoltre richiesta un' analisi comparativa tra le due IA$_G$ utilizzate e lo sviluppo di un plug-in$_G$ utile agli sviluppatori e al Project Manager$_G$.

1.3 Glossario

Alcuni termini presenti nel documento potrebbero essere ambigui, pertanto verranno inseriti nel Glossario v.1.0.0. La loro presenza all'interno di esso sarà indicata tramite una G maiuscola a pedice.

2 Riferimenti

2.1 Riferimenti normativi

- Capitolato **C7 - ChatGPT vs BedRock developer Analysis**

  https://github.com/CyberSorceres/CyberSorceresRepository
- Norme del way of working v 1.0.0
- Regolamento del progetto didattico

  https://www.math.unipd.it/ tullio/IS-1/2023/Dispense/PD2.pdf

2.2 Riferimenti informativi

- Slide del corso di Ingegneria del Software - Analisi dei requisiti

https://www.math.unipd.it/ tullio/IS-1/2023/Dispense/T5.pdf

- Slide del corso di Ingegneria del Software - Progettazione e programmazione: Diagrammi delle classi

  https://www.math.unipd.it/ rcardin/swea/2023/Diagrammi%20delle%20Classi.pdf

- Slide del corso di Ingegneria del Software - Solid Programming

  `https://www.math.unipd.it/ rcardin/swea/2021/SOLID%20Principles%20of%20Object-Orient`

2.3 Riferimenti tecnici

- Documentazione di React
  https://react.dev/
- Documentazione di Typescript
  https://www.typescriptlang.org/docs/
- Documentazione di MongoDB
  https://www.mongodb.com/docs/
- Documentazione di Amazon AWS
  https://docs.aws.amazon.com/it_it/
- Serverless Microservice Patterns
  https://medium.com/@jeremydaly/serverless-microservice-patterns-for-aws-6dadcd2
- Aws Reference Architecture Diagrams
  https://aws.amazon.com/it/architecture/reference-architecture-diagrams
- React design patterns
  https://refine.dev/blog/react-design-patterns/

3 Tecnologie

In questa sezione è presente una panoramica generale delle tecnologie necessarie per la realizzazione del prodotto (in particolare del front-end, back-end,databse e plugIn), gli strumenti e le librerie utilizzate per lo sviluppo, il testing e la distribuzione.

3.1 Tecnologie per la codifica

Tecnologia	Descrizione	Versione
Linguaggi		
HTML	Linguaggio di markup per delineare la struttura delle pagine e definire i componenti dell'interfaccia.	5
CSS	Linguaggio per la gestione dello stile degli HTML	3
Javascript	Superset di JavaScript per utilizzare tipizzazione	5.0.x
Framework		
React	Libreria grafica per lo sviluppo front-end che permette di gestire le unità grafiche in maniera modulare	18.0.x
Servizi e strumenti		
Node.js	Ambiente di runtime open-source per l'esecuzione di codice JavaScript lato server tramite appositi script.	19.0.x
NPM	Gestore dell'installazione della gestione dei pacchetti utilizzati in TypeScript e nell'ambiente di esecuzione Node.js.	3
AWS Cognito	Servizio di gestione dell'autenticazione.	2023-16-02
AWS MongoDB	Servizio di database non relazionale gestito in modo scalabile.	2019-11-21
AWS Lambda	Servizio che consente di eseguire codice in maniera serverless, garantendo la scalabilità automatica durante l'esecuzione.	2023-03-16
AWS API Gateway	Servizio di gestione (creazione, pubblicazione e protezione) delle API.	2023-04-06
Git	Sistema di controllo del versionamento e della gestione del codice.	2.4.x

Table 1: Tabella delle tecnologie per la codifica

3.2 Tecnologie per l'analisi del codice

Tecnologia	Descrizione	Versione
ViTest	Framework di test per TypeScript che permette la creazione di mock e il testing del codice in modo asincrono.	1.6.0

4 Architettura

Nella fase di progettazione è stata scelta un'architettura a microservizi come la più conforme alle carateristiche di funzionamento e strutturali delle tecnologie AWS, soprattutto API Gateway. Inoltre è quella che consente nel nostro caso la comunicazione per consentire la comunicazione tra le diverse componenti, considerando anche la presenza di un plugIn. Dati i requisiti del nostro progetto, abbiamo deciso che non fosse adeguato adottare un'unica struttura architetturale per l'intera l'infrastruttura. Abbiamo suddiviso il sistema in tre parti principali:

- Front-end: la parte client dell'applicazione eseguibile localmente su qualsiasi browser.
- Back-end: utilizza le tecnologie AWS (listate nella sezione 3.1), con l'interazione tramite NodeJS lato client e la comunicazione con il plugIn.
- PlugIn: comunica con il back-end grazie ad una libreria sviluppata dal team.

La comunicazione tra le diverse parti avviene attraverso l'uso di API Gateway.

4.1 Architettura front-end

4.1.1 Pattern utilizzati

4.2 Architettura back-end

4.2.1 Pattern utilizzati

5 Requisiti soddisfatti

Codice	Descrizione	Stato
ROF1	Accesso a web app tramite login composto da email e password.	Soddisfatto
ROF2	Scrittura di richieste di business tramite box testuale da web app.	Soddisfatto
ROF3	Invio delle richieste di businessda web app.	Soddisfatto
ROF4	Visualizzazione andamento sviluppo richieste tramite barra di completamento basata sulla percentuale di user stories completate.	Soddisfatto
ROF5	Approvazione o rifiuto del risultato relativo all'implementazione di una user story.	Soddisfatto
RDF6	Ricezione notifiche quando user story completata.	Non soddisfatto
ROF7	Funzionalità di tag nel plug-in.	Soddisfatto
ROF8	Lista di user stories assegnate da Project Manager sia su web app che su plug-in.	Soddisfatto
RDF9	Ricezione notifica su web app quando nuova user story è assegnata dal Project Manager.	Soddisfatto
ROF10	Invio del codice sviluppato a IA per richiesta verifica.	Soddisfatto
ROF11	Visualizzazione user stories generate da IA.	Soddisfatto
ROF12	Invio di feedback sulle user stories generate all'IA.	Soddisfatto
ROF13	Suddivisione delle user stories troppo grandi.	Soddisfatto
ROF14	Assegnazione user storieS agli sviluppatori.	Soddisfatto
RDF15	Ricezione notifiche quando user story$_G$ viene generata in seguito a richiesta del cliente.	Non soddisfatto
ROF16	Invio richiesta di modifiche relative a user stories a IA prima di approvazione.	Soddisfatto
ROF17	Visualizzazione andamento epic/user stories assegnate.	Soddisfatto
ROF18	Creazione di un plug-in per VSCode.	Soddisfatto
RDF19	Creazione di un plug-in per XCode.	Soddisfatto
ROF20	I linguaggi supportati dal plug-in sono Typescript e Javascript.	Soddisfatto

RDF21	Altri linguaggi che potrebbero essere supportati in futuro sono Kotlin$_G$ e Swift.	Non soddisfatto
ROF22	Gestione degli input (prevenzione da Injection Cross Site Scripting e sanificazione dell'input.)	Soddisfatto
ROQ1	Il progetto deve essere accessibile pubblicamente su GitHub o su un'altra repository pubblica.	Soddisfatto
ROQ2	Il prodotto deve essere sviluppato conformemente a quanto stabilito nelle *Norme Way of Working*$_G$.	Soddisfatto
ROQ3	Deve essere effettuato il testing delle unità e dell'integrazione con una copertura minima dell'80%.	Soddisfatto
ROQ4	Deve essere fornita una documentazione completa sulle scelte implementative e progettuali effettuate.	Soddisfatto
ROQ5	Deve essere fornito un manuale per l'utilizzo del prodotto.	Soddisfatto
ROQ6	Deve essere fornita una documentazione che compara la capacità di ChatGPT e quella di AWS Bedrock nell'interpretare del codice sorgente ed associare le user stories generate.	Soddisfatto
ROQ7	Deve essere fornita una documentazione che prova un'interpretazione corretta da parte dell'IA che si basa: sulle epic/user stories generate dall'IA, i test generati dall'IA, i criteri di accettazione delle epic/user stories forniti dal proponente.	Soddisfatto
ROV1	L'applicazione per l'interazione con la piattaforma dev'essere sviluppata attraverso l'uso di tecnologie web.	Soddisfatto
ROV2	Le due IA utilizzate per l'analisi sono AWS Bedrock e ChatGPT.	Soddisfatto
RDV3	L'applicazione deve essere utilizzabile tramite browser (Chrome 123.0, Firefox 124.0, Safari 17.0) di dispositivi mobili(Android 14.0, iOS 17.0).	Non soddisfatto
ROV4	Il front-end dell'applicazione verrà sviluppato in React.	Soddisfatto
ROV5	Ogni AWS Lambda-function deve essere sviluppata in Node.js.	Soddisfatto
ROV6	Tutte le API devono essere integrate in AWS API-gateway.	Soddisfatto

ROV7	L'applicativo deve essere compatibile con il browser Google Chrome dalla versione 121.	Soddisfatto
ROV8	L'applicativo deve essere compatibile con il browser Firefox dalla versione 122.	Soddisfatto
ROV9	L'applicativo deve essere compatibile con il browser Microsoft Edge dalla versione 121.	Soddisfatto
ROV10	Il plug-in deve essere compatibile con VSCode dalla versione 1.84.1 .	Soddisfatto
ROV11	L'accesso deve essere controllato da AWS Cognito, con autenticazione univoca.	Soddisfatto
ROV12	I ruoli devono essere definiti all'interno della piattaforma per evitare accessi non autorizzati.	Soddisfatto
ROV13	Protezione delle informazioni trasmesse tra browser e server tramite protocollo SSL/TLS.	Soddisfatto

Table 2: Tabella dei requisiti soddisfatti

5.1 Resoconto dei requisiti soddisfatti

6 Requisiti soddisfatti

Tipologia requisito	Istanze	Totale soddisfatto
Requisito funzionale	22	90.91%
Requisito obbligatorio	36	100%