

Università degli Studi di Salerno

Corso di Ingegneria del Software

AstroMark
System Design
Versione 1.0



Data: 24/11/2024

Progetto: AstroMark	Versione: 1.0
Documento: System Design	Data: 24/11/2024

Partecipanti:

Nome	Matricola
Giuseppe Cavallaro	0512116926
Mario Cosenza	0512116320
Mario Fasolino	0512116965
Giulio Sacrestano	0512116812

Scritto da:	Giuseppe Cavallaro
	Mario Cosenza
	Mario Fasolino
	Giulio Sacrestano

Revision History

Data	Versione	Descrizione	Autore
24/11/2024	1.0	Aggiunta introduzione, architettura corrente e proposta del software e servizi dei sottosistemi	Giuseppe Cavallaro, Mario Cosenza, Mario Fasolino, Giulio Sacrestano

Indice

1.	Introduzione	4
1.1.	Scopo del Sistema.....	4
1.2.	Obbiettivi e criteri di successo del progetto.....	5
1.3.	Definizioni, acronimi e abbreviazioni.....	9
1.4.	Riferimenti	10
1.5.	Sintesi.....	10
2.	Architettura corrente del software	11
3.	Architettura software proposta.....	12
3.1	Panoramica.....	12
3.2	Decomposizioni sottosistemi	13
3.3	Mappatura Hardware/Software	16
3.4	Gestione dati persistenti	19
3.5	Controllo accessi e sicurezza.....	20
3.6	Controllo globale del software.....	28
3.7	Condizioni di boundary.....	29
4.	Servizi dei sottosistemi	30
	Glossario	35

1. Introduzione

1.1. Scopo del Sistema

Il sistema AstroMark è concepito per offrire una piattaforma open-source dedicata alla gestione della didattica nelle scuole secondarie di secondo grado. L'obiettivo principale è quello di fornire un'alternativa economicamente sostenibile e tecnologicamente avanzata alle piattaforme commerciali attualmente dominanti nel mercato, come Argo e ClasseViva. Queste piattaforme esistenti sono spesso caratterizzate da costi elevati e da una notevole difficoltà nel passare a soluzioni alternative, il che limita le opzioni per le scuole.

AstroMark intende affrontare questi problemi offrendo una piattaforma flessibile e personalizzabile che permette una gestione semplificata delle funzioni scolastiche. In particolare, il sistema dovrà integrare funzionalità per la registrazione delle presenze, la gestione dei voti, le comunicazioni tra scuola e famiglia e, soprattutto, una sezione dedicata all'orientamento degli studenti, che rappresenta una lacuna significativa nelle offerte attuali.

Sarà progettato per essere accessibile anche a utenti con limitate competenze informatiche, come genitori e docenti, garantendo un'interfaccia intuitiva e user-friendly.

1.2. Obiettivi e criteri di successo del progetto

Gli obiettivi principali del progetto **AstroMark** includono la creazione di una piattaforma educativa economica e altamente flessibile, in grado di soddisfare le esigenze specifiche delle scuole secondarie di secondo grado. La piattaforma non solo deve sostituire le soluzioni commerciali esistenti, ma deve anche innovare e migliorare l'esperienza utente, offrendo funzionalità avanzate e una gestione intuitiva. L'obiettivo finale è garantire che **AstroMark** diventi un punto di riferimento nel panorama educativo italiano, supportando le scuole nel fornire un'istruzione di qualità e preparare gli studenti al loro futuro.

Di seguito vengono presentati i principali obiettivi di design per il sistema, che delineano le linee guida fondamentali per la sua architettura, funzionalità e prestazioni. Questi obiettivi mirano a garantire una soluzione robusta, scalabile e facile da usare, rispondendo alle esigenze degli utenti finali e delle parti coinvolte, in un contesto di efficienza operativa e sostenibilità a lungo termine.

Portability

Garantire la piena compatibilità della piattaforma con i principali browser e dispositivi hardware in uso. Questo obiettivo mira a rendere la piattaforma accessibile a un'ampia gamma di utenti, indipendentemente dal dispositivo o browser utilizzato, riducendo al minimo le barriere di accesso tecnologiche. Il supporto a versioni stabili e recenti dei browser assicura una migliore esperienza utente, sfruttando le ultime funzionalità disponibili.

Robustness

Assicurare la massima sicurezza delle comunicazioni e dei dati degli utenti attraverso l'implementazione di protocolli avanzati e tecniche di protezione. La crittografia TLS 1.3 rappresenta uno standard moderno che previene intercettazioni, garantendo la riservatezza delle informazioni sensibili. Inoltre, tecniche come l'hashing sicuro delle credenziali e la protezione contro vulnerabilità comuni, come SQL injection e XSS, mirano a proteggere l'integrità della piattaforma contro eventuali attacchi informatici.

User-friendliness

Rendere la piattaforma utilizzabile anche da utenti con disabilità, assicurando un'esperienza inclusiva e accessibile. Questo obiettivo si traduce nell'adozione di strumenti e test avanzati, come Lighthouse e JAWS, per garantire un elevato punteggio di accessibilità e un'interfaccia che rispetti i requisiti per ipovedenti. La piattaforma si impegna a ridurre al minimo le difficoltà di navigazione e a supportare le esigenze di un pubblico diversificato.

High-performance

Ottimizzare la velocità e le prestazioni del sito web per garantire un'esperienza fluida e rapida, indipendentemente dal carico. L'adozione di tecniche come il lazy loading, la minimizzazione di codice, l'uso di CDN e il caching dei contenuti punta a ridurre i tempi di caricamento e migliorare l'efficienza complessiva del sistema. Questo è fondamentale per mantenere alta la soddisfazione degli utenti e gestire efficacemente i picchi di utilizzo.

Adaptability

Garantire che la piattaforma sia pienamente funzionale su dispositivi con diverse risoluzioni, dai monitor desktop ai dispositivi mobili. La responsività dell'interfaccia è fondamentale per soddisfare le aspettative degli utenti moderni, che si aspettano un'esperienza ottimale indipendentemente dal dispositivo utilizzato. I test su emulatori e dispositivi reali contribuiscono a prevenire problemi di visualizzazione e interazione.

Reliability

Assicurare la stabilità operativa del sistema anche in contesti con carichi specifici, garantendo un utilizzo senza interruzioni per almeno 1500 utenti contemporanei.

Fault tolerance

- Proteggere la piattaforma da potenziali vulnerabilità e attacchi informatici comuni, garantendo la continuità delle operazioni. L'adozione di tecniche di prevenzione avanzate, come la protezione contro SQL injection e XSS, migliora la resilienza del sistema, prevenendo exploit che potrebbero compromettere la sicurezza o l'affidabilità del servizio.
- Facilitare la gestione della piattaforma e il ripristino dei dati in caso di guasti, attraverso un sistema di log dettagliato e backup regolari. La registrazione precisa delle anomalie consente al gruppo di sviluppo di individuare rapidamente le problematiche e intervenire in modo efficace, mentre le procedure di backup assicurano che i dati degli utenti siano sempre protetti e recuperabili.

Well-defined interfaces

Implementare una gestione rigorosa e chiara dei permessi di accesso, consentendo agli utenti di accedere esclusivamente alle sezioni della piattaforma per le quali dispongono delle autorizzazioni necessarie. Questo approccio garantisce una separazione netta dei ruoli e delle responsabilità, migliorando la sicurezza dei dati sensibili e la trasparenza delle operazioni all'interno del sistema.

Ease of use

Offrire un'esperienza utente familiare e intuitiva, ispirandosi a piattaforme già conosciute dal target di riferimento, come Argo. Le sessioni di usability testing con utenti reali mirano a identificare eventuali difficoltà nell'utilizzo e a migliorare l'interfaccia, rendendola più facile da apprendere e utilizzare.

Efficiency

Ridurre il carico sul database e ottimizzare i tempi di risposta della piattaforma attraverso l'uso di tecniche di caching avanzate e una corretta configurazione delle politiche di memorizzazione dei contenuti. Questo obiettivo contribuisce a migliorare le prestazioni complessive del sistema e a garantire un'esperienza utente fluida, anche in presenza di elevati volumi di traffico.

Scalability

Progettare il sistema in modo che sia facilmente scalabile, sia verticalmente che orizzontalmente. Questo implica che l'autenticazione degli utenti, elemento cruciale in un sistema distribuito, dovrà in futuro essere basata su tecnologie come JWT o OAuth, evitando la conservazione dello stato della sessione sui server. L'eliminazione della necessità di sincronizzare lo stato tra i server riduce significativamente la complessità operativa e facilita l'integrazione di nuovi nodi per gestire carichi di lavoro crescenti.

Rapid Development

Accelerare lo sviluppo delle prime versioni del sistema adottando inizialmente un'architettura più semplice per l'autenticazione, basata su form e cookie. Pur riconoscendo che questa scelta può limitare la scalabilità iniziale, si minimizzerà la quantità di dati memorizzati nella sessione per garantire una transizione agevole a soluzioni più avanzate in futuro. Questa strategia consente di velocizzare il rilascio del software, riservando l'introduzione di autenticazioni stateless, come JWT, a una fase successiva, quando le esigenze di scalabilità diventeranno prioritarie.

Cost-effectiveness

Realizzare un sistema che massimizzi il rapporto tra qualità e costi, garantendo che le risorse impiegate per l'infrastruttura, lo sviluppo e la manutenzione siano ottimizzate per soddisfare le esigenze operative senza sprechi o investimenti eccessivi.

In seguito, vengono analizzati alcuni trade-off necessari per il sistema, che riflettono le scelte e le priorità che hanno dovuto essere bilanciate durante il processo di progettazione. Tali trade-off riguardano aspetti critici come la gestione delle risorse, la complessità tecnica e la sostenibilità economica, con l'obiettivo di ottimizzare le prestazioni e la funzionalità senza compromettere la qualità complessiva del sistema.

Reliability vs Cost-effectiveness

Per garantire l'affidabilità del sistema, in particolare la stabilità operativa e la continuità del servizio con un carico massimo di 1500 utenti contemporanei, è necessario adottare soluzioni tecnologiche robuste, come infrastrutture cloud scalabili, bilanciamento del carico, e monitoraggio in tempo reale delle prestazioni. Tuttavia, queste scelte comportano un incremento significativo dei costi operativi, sia in termini di infrastruttura che di manutenzione.

Per mantenere la cost-effectiveness, è possibile optare per configurazioni iniziali meno onerose, utilizzando server condivisi o piani di hosting più economici, ma ciò potrebbe ridurre l'affidabilità complessiva, portando a rallentamenti o interruzioni del servizio in caso di picchi di utilizzo imprevisti.

vs Rapid Development

Scalability

Per ottenere un sistema facilmente scalabile sia verticalmente che orizzontalmente bisogna porre particolare attenzione anche a come viene effettuata l'autenticazione degli utenti. In un sistema distribuito con architettura REST i server non dovrebbero preservare alcuno stato di sessione prediligendo sempre soluzioni stateless come JWT o OAuth. L'utilizzo di cookie per l'autenticazione rende più complessa la scalabilità del sistema. Realizzare un sistema di autenticazione JWT è più complesso rispetto ad una autenticazione tramite form e cookie.

Quindi per una semplicità di sviluppo le prime versioni del software verranno prodotte con questa autenticazione riducendo comunque al minimo i dati memorizzati nella sessione permettendo una futura evoluzione del sistema rapida.

1.3. Definizioni, acronimi e abbreviazioni

Di seguito è fornito un elenco degli acronimi, abbreviazioni con le relative definizioni utilizzati in questo documento:

Termine	Definizione
AES-256	Advanced Encryption Standard a 256 bit
AWS	Amazon Web Service
DDoS	Distributed Denial of Service
DNS	Domain Name System
FR	Requisito funzionale
HTTP	Hypertext Transfer Protocol
JDBC	Java Database Connectivity
JSF	JavaServer Faces
JWT	JSON Web Token
NA	Not Accessible
OAuth	Open Authorization
PHP	Hypertext Preprocessor
SSL	Secure Sockets Layer
TLS	Transport Layer Security
XSS	Cross-Site-Scripting

1.4. Riferimenti

Il presente progetto si basa sull'analisi e il confronto con piattaforme di gestione della didattica già consolidate e affermate nel settore, le quali hanno dimostrato notevole efficacia. Tra queste, un punto di riferimento significativo è rappresentato dalle soluzioni sviluppate da Argo per la gestione della didattica.

Di seguito si presenta un elenco dei documenti chiave del progetto a cui si fa esplicito riferimento:

- **Problem Statement:** documento che definisce i problemi principali che il progetto intende affrontare e risolvere.
- **Requirements Analysis Document (RAD):** documento che definisce e analizza i requisiti funzionali e non funzionali di un sistema, descrivendo le necessità degli utenti e i vincoli progettuali.

Oltre ai documenti del progetto, si fa riferimento a opere di letteratura tecnica e risorse online che hanno contribuito allo sviluppo metodologico e concettuale di questo lavoro:

- [*Object-Oriented Software Engineering Using UML, Patterns, and Java™ Third Edition*](#) di Bernd Bruegge & Allen H. Dutoit.
- [The Architecture of the Spring Boot REST Applications](#), un approfondimento sull'architettura delle applicazioni REST basate su Spring Boot.
- [Representational State Transfer \(REST\)](#), una descrizione concettuale e tecnica delle architetture RESTful.
- [Argo - Le nostre infrastrutture](#), una descrizione dettagliata delle infrastrutture e dei sistemi offerti da Argo per la gestione della didattica.

1.5. Sintesi

In questo paragrafo sono stati affrontati temi legati alla creazione di una piattaforma scolastica innovativa, con particolare attenzione alla sostenibilità economica, alla semplificazione della gestione scolastica e al miglioramento dell'accessibilità per gli utenti.

Viene discusso il bilanciamento tra affidabilità del sistema e controllo dei costi operativi, evidenziando le sfide e le opportunità di sviluppo. Si analizzano inoltre le strategie per garantire scalabilità e continuità del servizio, considerando la necessità di un'evoluzione graduale del progetto. Infine, si sottolinea l'importanza di progettare una soluzione adatta a esigenze diverse e facilmente adattabile nel tempo.

La restante parte del documento presenta una descrizione accurata dei sottosistemi individuati, della architettura del sistema e di come questo viene gestito.

2. Architettura corrente del software

Il sistema **AstroMark** è ancora in fase di sviluppo non è presente alcun sistema funzionante allo stato corrente. Le principali piattaforme concorrenti sono quelle offerte da Argo Software Srl e quelle del Gruppo Spaggiari Parma (ClasseViva). I software prodotti da Argo Software Srl come **Argo Famiglia** utilizzano un'infrastruttura moderna con database server, repository dei documenti e le relative copie di backup ospitati presso la sicura e affidabile piattaforma cloud AWS, strutturata in "regioni", costituite da più "zone di disponibilità", fisicamente separate e ridondanti fra loro.

Pertanto, è in uso un'architettura distribuita con una forte separazione tra logica applicativa e di presentazione, infatti il software è accessibile sia da web browser che da applicazione Android e iOS con tecnologie differenti ad esempio JSF, Vue.js, PHP. Separando quindi il front-end dal back-end la piattaforma è facilmente adattabile alle esigenze grafiche del momento.

La piattaforma fa uso anche di **Amazon CloudFront** il servizio di CDN di AWS questo permette di gestire eventuali picchi di traffico sulla rete e migliorando i tempi di accesso. Una priorità assoluta è la gestione dei dati anche in questo caso ci sono nodi multipli e ridondati essendo i dati centrali in ogni applicazione specialmente per un registro elettronico. La piattaforma è anche scalabile dovendo supportare in media 250 000 utenti distinti al giorno. L'infrastruttura di Argo garantisce alte prestazioni anche in presenza di carichi di lavoro elevati, come un traffico di rete superiore a 50 Tb al mese, oltre 3 miliardi di richieste HTTP(S) mensili e più di 550 Tb di file e documenti gestiti in modo ridondato e sottoposti a backup regolari.

Tutti i dati sono protetti con crittografia AES-256, e la gestione avanzata dei sistemi consente ridondanza e replicazione per minimizzare i rischi di interruzioni o perdite di dati.

La sicurezza è ulteriormente garantita da frequenti test antintrusione eseguiti da esperti e dalla scelta strategica di gestire internamente le infrastrutture, grazie a una squadra specializzata operativa 7 giorni su 7.

ClasseViva invece utilizza come provider di **CDN Akamai** e anche in questo caso è utilizzata una architettura di distribuita almeno su tre livelli.

Anche AstroMark dovrà utilizzare infrastrutture scalabili e l'applicazione dovrà utilizzare un'architettura distribuita, permettendo l'evoluzione indipendente di client e server.

3. Architettura software proposta

3.1 Panoramica

Il sistema **AstroMark** sarà progettato come una piattaforma open-source dedicata alla gestione della didattica nelle scuole secondarie di secondo grado.

L'architettura software proposta adotta un modello **three-tier**, che organizza la piattaforma in tre principali strati: **interfaccia utente**, **logica applicativa** e **gestione dei dati**. Questo modello consente una chiara separazione delle funzionalità, facilitando la scalabilità, la manutenzione e l'evoluzione del sistema, rispondendo efficacemente alle esigenze di flessibilità, sicurezza e prestazioni richieste dalle scuole.

La **natura distribuita** dell'architettura permette la gestione indipendente di ciascun layer, con i vari sottosistemi che operano in sinergia per garantire una user experience fluida e sicura.

I principali **sottosistemi** che compongono l'architettura saranno progettati per gestire funzionalità specifiche, come la gestione dell'interfaccia utente, l'autenticazione degli utenti, la gestione dei profili e dei permessi, la comunicazione in tempo reale (chat), la gestione dell'orario e delle prenotazioni, nonché l'orientamento scolastico e la valutazione.

Il **sottosistema di interfaccia utente** sarà progettato per essere intuitivo e accessibile anche agli utenti con competenze informatiche limitate, come genitori e docenti.

La **gestione dell'autenticazione** e la protezione contro accessi non autorizzati sono affrontate tramite il sottosistema dedicato, che include funzionalità come login, gestione delle sessioni e protezione delle informazioni sensibili.

La piattaforma include inoltre sottosistemi per la gestione di **attività scolastiche**, come l'assegnazione dei compiti, la registrazione delle presenze e dei voti, e la gestione delle giustificazioni. Il **sottosistema di orientamento** sfrutta algoritmi per suggerire percorsi di studio personalizzati, mentre il **sottosistema di gestione delle classi** supporta la gestione delle attività didattiche quotidiane, tra cui la pianificazione degli orari e la gestione delle lezioni.

Il **database** funge da cuore del sistema, archiviando tutte le informazioni relative agli utenti, alle attività scolastiche e amministrative, e garantendo l'integrità e la sicurezza dei dati. L'adozione di un'architettura distribuita permette di gestire in modo efficiente le operazioni concorrenti e di bilanciare il carico tra i vari nodi, assicurando una buona scalabilità e continuità del servizio anche con un alto numero di utenti simultanei.

3.2 Decomposizioni sottosistemi

Sono stati individuati e definiti i seguenti sottosistemi, che rappresentano le principali componenti funzionali e strutturali, all'interno delle quali verrà decomposto il sistema AstroMark, al fine di facilitarne lo sviluppo, la gestione e la manutenzione:

Sottosistema User Interface

Responsabile della gestione dell'interfaccia utente, fornisce una visualizzazione intuitiva e user-friendly delle funzionalità della piattaforma. Include elementi grafici, layout responsivi e interazioni con gli utenti, assicurando accessibilità e facilità d'uso per tutte le categorie di utilizzatori.

Sottosistema Authentication

Gestisce i meccanismi di autenticazione degli utenti, garantendo accesso sicuro alla piattaforma. Comprende funzionalità come login/logout, gestione delle sessioni e protezione contro tentativi di accesso non autorizzati. Il sottosistema gestisce anche la funzionalità di primo login dell'utente.

Sottosistema User Management

Permette la gestione dei profili utente, includendo la modifica e cancellazione degli account, nonché l'assegnazione di ruoli e permessi. Il sottosistema gestisce anche il recupero delle credenziali personali.

Sottosistema Chat

Offre una piattaforma di comunicazione in tempo reale tra utenti. Include funzionalità come gestione di una conversazione studente/docente relativa ad un assegno e la gestione dei ticket per la segreteria.

Sottosistema Agenda

Gestisce la pianificazione e l'organizzazione dell'orario delle classi e di ricevimento del docente. Permette ad un docente di firmare le ore di lezione ed inoltre, include anche le funzionalità per la prenotazione del ricevimento.

Sottosistema Orientation

Fornisce un'interfaccia basata su algoritmi per la gestione di funzionalità avanzate legate all'orientamento degli studenti, come suggerimenti personalizzati sui percorsi di studio o raccomandazioni sulla base di analisi dei dati scolastici.

Sottosistema Class Management

Supporta docenti e segreteria nella gestione delle attività didattiche. Include funzionalità come la ricerca dell'elenco degli studenti di una classe, assegnazione di docenti e studenti alla classe, modifica delle informazioni sul piano di studio della classe.

Sottosistema School Management

Dedito alla gestione amministrativa dell'istituto scolastico, include strumenti per la creazione di account segreteria, docente, genitore, studente. Permette la creazione di un nuovo istituto e la modifica di uno esistente.

Sottosistema Justification and Behavior

Gestione delle giustificazioni per assenze, ritardi o uscite anticipate, con funzionalità per l'inserimento di una motivazione da parte dei genitori. Il sottosistema consente anche la gestione delle note disciplinari.

Sottosistema Rating

Gestione della valutazione degli studenti, compresa la registrazione dei voti e la preparazione degli scrutini.

Sottosistema Classwork

Permette ai docenti di specificare le attività svolte in classe, di assegnare i compiti per casa e monitorare l'avanzamento delle consegne degli studenti.

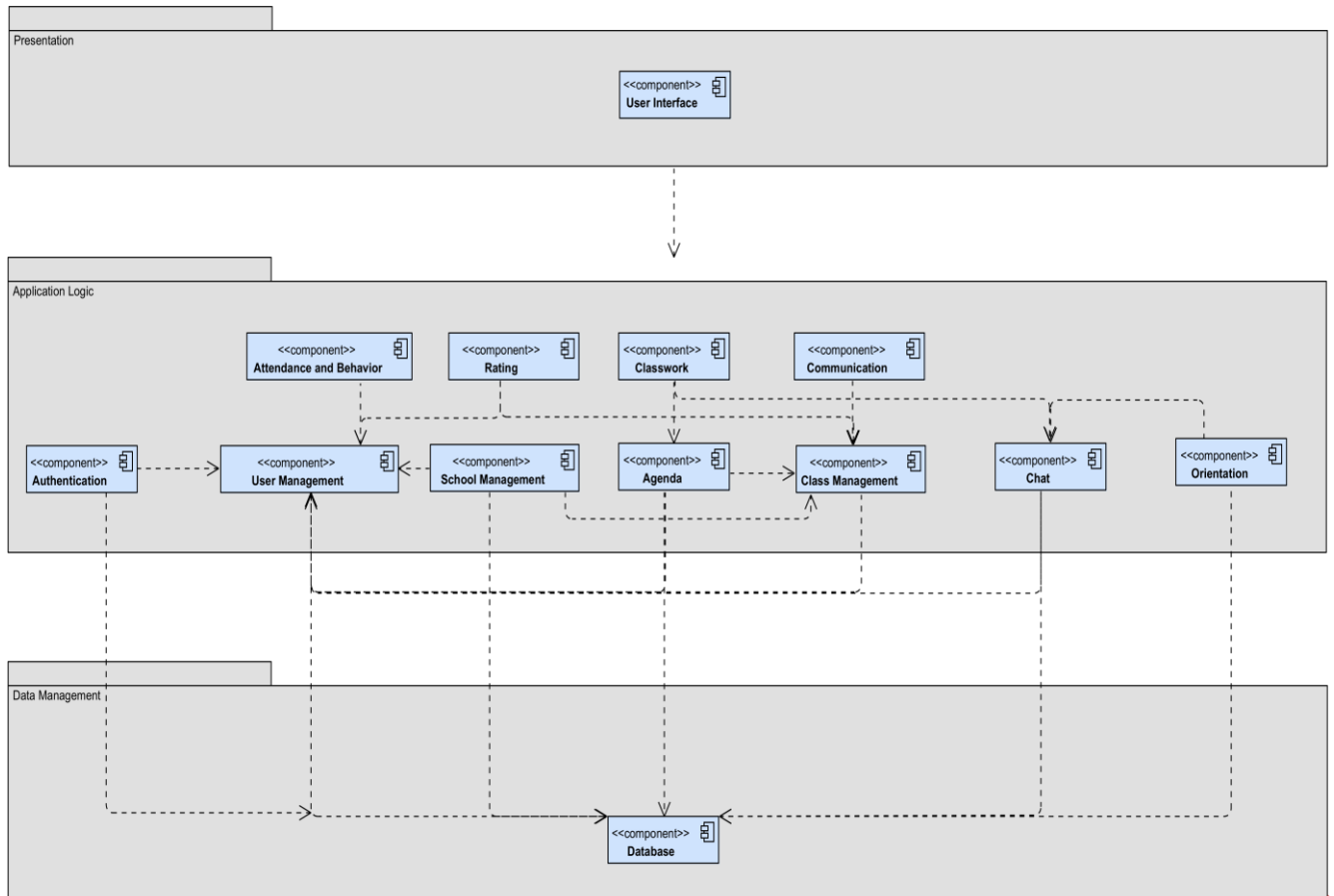
Sottosistema Communication

Facilita lo scambio di informazioni tra docenti, studenti e genitori, integrando una bacheca di comunicazioni per la classe e consentendo la creazione di nuovi avvisi da parte del docente.

Sottosistema Database

Componente fondamentale per l'archiviazione e la gestione dei dati della piattaforma. Contiene informazioni relative agli utenti, alla didattica, alle attività scolastiche e amministrative, garantendo sicurezza, integrità e performance nelle operazioni di lettura e scrittura.

Di seguito il diagramma dei componenti suddiviso nei layer Presentation, Application Logic, Data Management, per una migliore leggibilità sono omesse le dipendenze tra la user interface e i componenti della application logic.



3.3 Mappatura Hardware/Software

Data la necessità di realizzare una piattaforma distribuita e facilmente scalabile, è stato scelto uno stile architetturale **three-tier**. L'architettura adotta un modello a **ciclo chiuso**, che consente ai vari layer di accedere esclusivamente al layer immediatamente inferiore, senza possibilità di interazioni dirette con livelli più profondi. Questo approccio promuove una chiara separazione delle responsabilità, semplifica la manutenzione e aumenta la sicurezza del sistema, riducendo il rischio di errori o accessi non autorizzati ai componenti più sensibili.

La mappatura del software sui nodi hardware verrà effettuata utilizzando soluzioni offerte da provider cloud, come **AWS** tramite il servizio **Amazon EC2**, che consente una rapida scalabilità sia orizzontale che verticale. Per semplificare l'installazione, l'applicazione sarà distribuita in formato **JAR** con un web container integrato (Tomcat o Jetty).

L'applicazione comunicherà con un modulo di Machine Learning, che fornirà un'interfaccia API per ottenere informazioni utili sull'orientamento scolastico. Le informazioni saranno recuperate tramite richieste HTTP e il modulo di orientamento, grazie alla sua flessibilità, potrà essere eseguito sia sullo stesso web server che su un nodo separato.

Per ridurre il carico derivante dalle richieste di contenuti statici, verranno utilizzate **Content Delivery Network**, come quelle offerte da **CloudFlare**, che garantiscono anche protezione contro attacchi **DDoS**.

Il **DBMS** per la gestione del database relazionale della piattaforma sarà eseguito idealmente su un nodo separato, per minimizzare il rischio che eventuali malfunzionamenti del web server possano impattare sui dati persistenti. Un ulteriore nodo sarà dedicato alla gestione degli allegati ai messaggi delle chat, utilizzando un database basato su oggetti e sfruttando la rete di CloudFlare per garantire performance elevate e sicurezza.

Le richieste del client saranno effettuate tramite **HTTPS** verso un **server Edge** di CloudFlare localizzato geograficamente più vicino all'utente, individuato tramite DNS, oppure direttamente verso uno dei web server, specificando l'indirizzo IP di quest'ultimo. Le richieste saranno elaborate dal server, che, se necessario, contatterà il server HTTP dedicato alle API di Machine Learning e interrogherà il database tramite **JDBC**.

Come tutte le architetture anche la three-tier presenta vantaggi e svantaggi:

Vantaggi

Modularità e Manutenibilità

- Ogni livello è separato e ha una responsabilità ben definita, rendendo il codice più organizzato e semplice da mantenere.
- Gli aggiornamenti a un livello non richiedono necessariamente modifiche agli altri livelli.

Scalabilità

- È possibile scalare orizzontalmente (aggiungendo server) uno o più livelli indipendentemente dagli altri, in base alle esigenze di carico.

Flessibilità

- Ogni livello può essere implementato usando tecnologie diverse. Ad esempio, il front-end potrebbe essere sviluppato in React, il middleware con Spring Boot e il database con PostgreSQL.
- Si può sostituire una tecnologia a un livello senza compromettere l'intera applicazione.

Affidabilità

- La separazione dei livelli consente una maggiore tolleranza ai guasti: un problema in un livello può essere isolato senza compromettere gli altri.
- È possibile implementare meccanismi di failover e backup specifici per ogni livello.

Sicurezza

- L'accesso ai dati è mediato dal livello di logica di business, riducendo i rischi di attacchi diretti al database.
- È possibile implementare meccanismi di autenticazione e autorizzazione nei vari livelli.

Riutilizzabilità

- Il middleware può essere condiviso tra più applicazioni, evitando duplicazioni di codice.

Svantaggi

Complessità Implementativa

- Richiede un maggiore sforzo progettuale per garantire l'integrazione fluida tra i livelli.
- Il debugging e il testing possono essere più complessi, poiché il comportamento dipende dall'interazione tra i vari livelli.

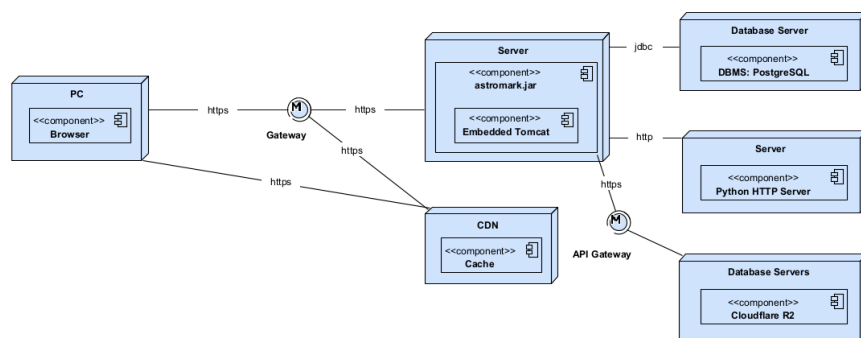
Overhead di Comunicazione

- I livelli distribuiti possono introdurre ritardi di rete e aumento del traffico, specialmente in applicazioni con alti volumi di richieste.
- L'uso di protocolli di comunicazione (es. REST) può aggiungere latenza.

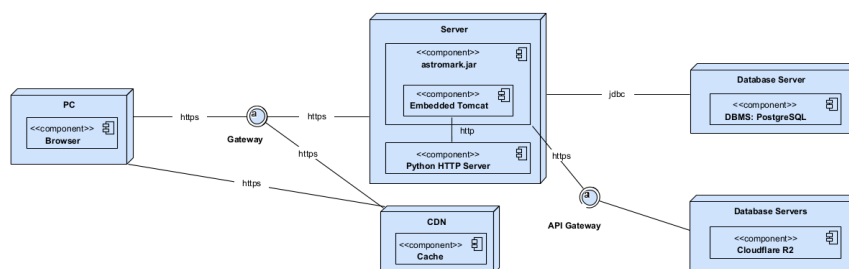
Costi Iniziali Elevati

- I costi di infrastruttura, configurazione e manutenzione sono maggiori rispetto a un'architettura monolitica.
- Sono necessari strumenti avanzati per il monitoraggio e il bilanciamento del carico tra i livelli.

Deploy diagram con sottosistema di orientamento realizzato in Python distribuito su un nodo separato dal web server.



Deploy diagram con sottosistema di orientamento in esecuzione nel nodo del web server.



3.4 Gestione dati persistenti

La gestione della persistenza nel progetto sarà realizzata utilizzando il database relazionale **PostgreSQL**, insieme a **CloudFlare R2**, una soluzione di **object storage** compatibile con **Amazon S3**, per la gestione di specifici file non relazionali.

Tutte le entità del sistema, ad eccezione di *User*, *Chat*, *Timetable*, *Timeslot* e *TeacherClass*, verranno memorizzate nel database PostgreSQL. La scelta di non includere queste entità nella persistenza relazionale è motivata dal conflitto di impedenza tra un linguaggio orientato ad oggetti come Java e uno schema relazionale.

PostgreSQL è stato selezionato come DBMS per le sue caratteristiche avanzate che garantiscono affidabilità, prestazioni e flessibilità. Tra i suoi principali vantaggi vi sono il supporto completo alle transazioni **ACID**, la gestione efficiente di relazioni tra entità grazie a vincoli di chiave primaria ed esterna, indici avanzati e la possibilità di eseguire query SQL avanzate.

L'architettura transazionale del database garantirà consistenza e sicurezza dei dati in un ambiente multiutente, con l'adozione di meccanismi di ottimizzazione delle query e connessioni sicure per gestire carichi di lavoro significativi in modo efficace.

Per quanto riguarda i file associati alle entità, come gli allegati ai messaggi nelle chat, essi non verranno salvati direttamente nel database relazionale, ma gestiti attraverso CloudFlare R2 compatibile con l'**API S3** di Amazon, consente una facile integrazione tramite SDK standard e offre flessibilità nel caricamento, nella gestione e nel recupero di file. Questo approccio permette di alleggerire il carico sul database relazionale, evitando di compromettere le prestazioni con la memorizzazione di file di grandi dimensioni, e sfruttare i vantaggi dello storage a oggetti, come la scalabilità elastica e i costi ridotti per lo storage a lungo termine.

L'integrazione tra questi sistemi sarà gestita attraverso configurazioni centralizzate in **Spring Boot**, garantendo una comunicazione sicura e ottimizzata tra il livello applicativo, il database e il sistema di storage. La combinazione di PostgreSQL per i dati strutturati e R2 per i file non relazionali rappresenta una soluzione robusta e scalabile, capace di soddisfare i requisiti operativi del progetto, mantenendo un'elevata efficienza e sostenibilità nel lungo termine.

3.5 Controllo accessi e sicurezza

La sicurezza dell'applicazione sarà garantita da una serie di misure avanzate che proteggono sia il front-end che il back-end, assicurando una protezione completa dei dati e delle comunicazioni. Il front-end utilizzerà **React** insieme a **React Router**, che gestisce la navigazione e il routing tra le varie pagine, permettendo un'esperienza utente fluida e sicura. React Router è una libreria potente che consente di definire le rotte in modo dichiarativo, migliorando la gestione della navigazione. La comunicazione tra client e server avverrà tramite HTTPS, garantendo che tutti i dati trasmessi siano criptati e protetti da intercettazioni, utilizzando il protocollo di sicurezza **SSL/TLS** per prevenire attacchi di tipo man-in-the-middle e garantire la riservatezza e integrità delle informazioni. La sicurezza sarà ulteriormente rafforzata dall'utilizzo di una **Security Filter Chain** che gestirà le richieste in entrata, assicurando che solo quelle autorizzate possano accedere alle risorse del server.

La **Security Filter Chain** è una componente di **Spring Security** che permette di applicare politiche di autenticazione e autorizzazione, garantendo che solo gli utenti con i giusti permessi possano accedere alle risorse protette. Inoltre, l'applicazione gestirà l'autenticazione tramite cookie per memorizzare in modo sicuro le informazioni di sessione e proteggere gli accessi.

Le password degli utenti saranno protette tramite l'hashing irreversibile con **bcrypt**, un algoritmo di hashing progettato per resistere agli attacchi di forza bruta. Questo rende improbabile il recupero delle password originali anche in caso di compromissione dei dati. Ogni utente avrà un nome utente unico, basato sul proprio nome e cognome, con l'aggiunta di un numero, se necessario, per evitare conflitti. La password di primo accesso verrà inviata tramite e-mail e deve essere cambiata al primo login. L'autenticazione avverrà tramite un form che richiede il codice della scuola, il nome utente e la password, garantendo che ogni utente sia correttamente legato alla scuola di appartenenza, che sia genitore, studente, docente o personale di segreteria.

Ogni categoria ha permessi e accessi specifici, assicurando che gli utenti possano accedere solo alle informazioni e alle funzionalità pertinenti al loro ruolo, migliorando la sicurezza e l'organizzazione complessiva dell'applicazione.

Ma

	Ingegneria del Software	Pagina 21 di 35
--	-------------------------	-----------------

		getSchool addReceptionBooking		addStudent getStudent getTicket getSchool	
Teacher	getUsername getName getSurname getGender getSchoolClasses getTeaching getTeacherClasses getSchool	getUsername getName getSurname getGender getSchoolClasses getTeaching getTeacherClass getSchool	getUsername getName getSurname getGender getEmail getTaxID getBirthDate getResidentialAddresses setEmail setUsername remove setResidentialAddresses getSchoolClasses getTeaching getTeacherClass getSignedHour getReceptionTimeTable getTicket getSchool	createTeacher getUsername getName getSurname getGender getEmail getTaxID getBirthDate getResidentialAddress setEmail setUsername remove setResidentialAddress getSchoolClasses getTeaching getTeacherClass getSignedHour getReceptionTimeTable getTicket getSchool addTeaching addSchoolClass getTicket getSchool	N.A.
Secretary	N.A.	getUsername getName getSurname getSchool	getUsername getName getSurname getSchool	createSecretary getUsername getName getSurname getGender getEmail getTaxID getBirthDate getResidentialAddress setEmail setUsername remove setResidentialAddress getEnableSecretaryAccountCreation setEnableSecretaryAccountCreation getSchool	createSecretary getUsername getName getSurname getGender getEmail getTaxID getBirthDate getResidentialAddress setEmail setUsername remove setResidentialAddress getEnableSecretaryAccountCreation setEnableSecretaryAccountCreation
School-Manager	N.A.	N.A.	N.A.	N.A.	createSchoolManager getUsername getName getSurname

					getGender getEmail getTaxID getBirthDate getResidentialAddress setEmail setUsername remove getOneFactorRemoveSchool setOneFactorRemoveSchool
School	getCode getPhoneNumber getAddress getName getEmail getSchoolPrincipalFullName	getCode getPhoneNumber getAddress getName getEmail getSchoolPrincipalFullName getSecretary	getCode getPhoneNumber getAddress getName getEmail getSchoolPrincipalFullName getSchoolClass getSecretary	getCode getPhoneNumber getAddress getName getEmail getSchoolPrincipalFullName addClass addStudent addSecretary addTeacher addParent getStudent getSchoolClass getTeacher getParent getSecretary	createSchool getCode getPhoneNumber getAddress getName getEmail getSchoolPrincipalFullName setPhoneNumber setAddress setName setEmail setSchoolPrincipalFullName addSecretary getSecretary
School-Class	getNumber getLetter getYear getTeacher getSchool getCommunication getStudyPlan getStudent getClassTimetable	getNumber getLetter getYear getTeacher getSchool getCommunication getStudyPlan getStudent getClassTimetable	getNumber getLetter getYear getStudentMarkByDate getSchool getCommunication getStudyPlan getStudent getClassTimetable addCommunication	createClass getNumber getLetter getYear addStudent getStudentMarkByDate getTeacher getSchool getCommunication getStudyPlan getStudent getClassTimetable addStudent setStudyPlan addStudent	N.A.
Comunication	getTitle getDescription getDate	getTitle getDescription getDate	createComunication getTitle getDescription getDate setTitle setDescription	N.A.	N.A.
Teacher-	isCoordinator	isCoordinator	isCoordinator	createTeacherClass isCoordinator	N.A.

Class					
Class-Timetable	getTitle getStartValidity getEndValidity getExpectedHours getTimeSlot isRedDate isValid getTotalHours getTeachingTimeslot getSchoolClass	getTitle getStartValidity getEndValidity getExpectedHours getTimeSlot isRedDate isValid getTotalHours getTeachingTimeslot getSchoolClass	getTitle getStartValidity getEndValidity getExpectedHours getTimeSlot isRedDate isValid getTotalHours signHour getTeachingTimeslot getSchoolClass	createClassTimetable getTitle getStartValidity getEndValidity getExpectedHours setTitle setStartValidity setEndValidity getTimeSlot addTimeSlot isRedDate isValid getTotalHours getTeachingTimeslot getSchoolClass addTeachingTimeslot	N.A.
Teaching-Timeslot	getHour getDay getTeaching getClassTimetable	getHour getDay getTeaching getClassTimetable	getHour getDay getClassroom isSigned getSignedHour getTeaching getClassTimetable addSignedHour	createTeacherTimeslot getHour getDay getClassroom getTeaching getClassTimetable setClassTimetable	N.A.
Signed-Hour	isSubstitution getTeacher getClassActivity getHomework	isSubstitution getTeacher getClassActivity getHomework	createSignedHour getDateTimeSign isSubstitution getTeacher getClassActivity getHomework setClassActivity setHomework	N.A.	N.A.
Class-Activity	getDescription getTitle getSignedHour	getDescription getTitle getSignedHour	createClassActivity getDescription getTitle setDescription setTitle getSignedHour	N.A.	N.A.
Homework	getDueDate getDescription getTitle isNeedsChat getSignedHour getHomeworkChat	getDueDate getDescription getTitle isNeedsChat getSignedHour getHomeworkChat	createHomework getDueDate getDescription getTitle isNeedsChat setDueDate setDescription setTitle setNeedsChat getSignedHour getHomeworkChat	N.A.	N.A.
Message	createMessage getText	createMessage getText	createMessage getText	createMessage getText	N.A.

	getAttachment getDateTime downloadAttachment getSender	getAttachment getDateTime downloadAttachment getSender	getAttachment getDateTime downloadAttachment getSender	getAttachment getDateTime downloadAttachment getSender	
Home-workChat	createHomeworkChat getTitle getDate isCompleted sendMessage getMessageNumber getHomework	getTitle getDate isCompleted getMessageNumber getHomework	getTitle getDate isCompleted sendMessage getMessageNumber closeWithFeedback getStudent getHomework	N.A.	N.A.
Ticket	N.A.	createTicket getTitle getDate sendMessage getMessageNumber getStatus reopen	createTicket getTitle getDate sendMessage getMessageNumber getStatus reopen	getTitle getDate getStatus sendMessage getMessageNumber reopen close getParent getTeacher	N.A.
Teaching	getTimeOfActivity getTeacher getSubject	getTimeOfActivity getTeacher getSubject	getTimeOfActivity getTeacher getSubject	createTeaching setTimeOfActivity getTimeOfActivity getTeacher getSubject addTeachingTimeslot	N.A.
Subject	getTitle	getTitle	getTitle	createSubject getTitle	N.A.
StudyPlan	getTitle getSubject	getTitle getSubject	getTitle getSubject	createSubject getTitle getSubject addSubject	N.A.
Semester-Report	getSemester isPublic getYear isPassed isViewed setViewed getSubject getSudent	getSemester isPublic getYear isPassed isViewed setViewed getSubject getSudent	createSemesterReport getSemester isPublic getYear isPassed isViewed addMark getSubject getSudent addSubject	N.A.	N.A.
Mark	getDate getType getRating getDescription isSufficient getTeaching getStudent	getDate getType getRating getDescription isSufficient getTeaching getStudent	createMark getDate getType getRating getDescription isSufficient getTeaching	N.A.	N.A.

			getStudent		
Absence	isNeedsJustification isJustified getJustificationText getDate	isNeedsJustification isJustified getJustificationText getDate Justifie	createAbsence isNeedsJustification isJustified getJustificationText getDate setJustificationText	N.A.	N.A.
Delay	isNeedsJustification isJustified getJustificationText getDate getTime	isNeedsJustification isJustified getJustificationText getTime Justifie	createDelay isNeedsJustification isJustified getJustificationText getTime setJustificationText	N.A.	N.A.
Note	getDate getDescription	getDate getDescription setViewed	createNote getDate getDescription setDescription isViewed	N.A.	N.A.
Reception Timetable	N.A.	getTitle getStartValidity getEndValidity getTextInfoReception getBookedSlots getTimeSlot isRedDate isValid book getNotConfirmed getRefused getTeacher getReceptionTimeslot	createReceptionTimetable getTitle getStartValidity getEndValidity getTextInfoReception getBookedSlots setTextInfoReception getTimeSlot isRedDate isValid refuse getNotConfirmed confirm getRefused getTeacher getReceptionTimeslot addReceptionTimeslot	N.A.	N.A.
Reception Timeslot	N.A.	getHour getDay getMode getCapacity getReceptionTimetable getReceptionBooking addReceptionBooking	createReceptionTimeslot getHour getDay getMode getCapacity getReceptionTimetable getReceptionBooking	N.A.	N.A.
Reception-Booking	N.A.	createReceptionBooking getDate isConfirmed isRefused getReceptionTimeslot	getDate isConfirmed isRefused setRefused setConfirmed getParent getReceptionTimeslot	N.A.	N.A.

Nel design delle classi *SemesterReport* e *Secretary*, è stato adottato un approccio specifico per garantire che determinati metodi siano accessibili esclusivamente quando vengono soddisfatte condizioni ben definite, legate al ruolo e agli attributi dell'attore che invoca i metodi.

- **Classe SemesterReport:** Per la gestione degli scrutini e delle valutazioni, i metodi *createSemesterReport*, *addMark* e *addSubject* possono essere utilizzati solo dall'attore **Professore** a condizione che quest'ultimo sia il **coordinatore** della classe a cui lo studente dello scrutinio appartiene. Questo controllo garantisce che solo i professori autorizzati abbiano accesso alle operazioni sensibili legate alla creazione e modifica dei report semestrali.
- **Classe Secretary:** Per la gestione degli account e delle operazioni amministrative, i metodi *createSecretary* e *setEnabledSecretaryAccountCreation* sono accessibili dall'attore **Segreteria** solo se il suo attributo *enableSecretaryAccountCreation* è impostato a *true*. Questa restrizione assicura che l'attore segreteria possa effettuare la creazione di account di segreteria unicamente quando dispone dei permessi necessari.

Questa soluzione garantisce un controllo rigoroso degli accessi e preserva l'integrità e la sicurezza delle operazioni all'interno del sistema, assicurando che ogni attore possa eseguire solo le azioni per cui è esplicitamente autorizzato.

3.6 Controllo globale del software

Sarà adottato il paradigma di controllo del flusso **event-driven**, che consente un'elaborazione basata su eventi per garantire flessibilità e scalabilità. Ogni esecuzione dell'applicazione su un nodo opererà in completa trasparenza rispetto agli altri nodi, favorendo la modularità e riducendo la complessità gestionale. In base alla posizione geografica dell'utente, la richiesta sarà indirizzata al web server più adeguato, sfruttando meccanismi di bilanciamento del carico per ottimizzare le prestazioni del sistema distribuito.

Ogni web server sarà configurato per rimanere in attesa di richieste provenienti dal web browser, che potranno riguardare sia un endpoint del back-end sia una pagina web statica o dinamica. L'applicazione utilizzerà container web come **Tomcat** o **Jetty** che permettono di accedere a contenuti statici, come le pagine HTML, oltre a gestire richieste più complesse legate alla business logic. Le richieste HTTP verranno elaborate da una **Dispatcher Servlet**, che le instraderà verso i controller appropriati, responsabili della gestione delle richieste specifiche in base al pattern dell'URL. Per supportare richieste concorrenti, il container web utilizzerà un sistema di **thread pool**, che consente di gestire più richieste in parallelo senza compromettere le prestazioni. Questa soluzione garantisce un'elaborazione efficiente delle richieste anche in condizioni di carico elevato, riducendo il tempo di latenza percepito dagli utenti.

Affinché il sistema distribuito operi in maniera affidabile e coerente, sarà necessario limitare le situazioni di concorrenza tra i servizi offerti. Tuttavia, vi saranno scenari in cui una sincronizzazione tra componenti sarà inevitabile. Per le prime versioni di **AstroMark**, lo sforzo principale sarà orientato all'implementazione di meccanismi di **table lock** nel database, in modo da semplificare la gestione della concorrenza iniziale. Questa soluzione, sebbene semplice, garantirà la consistenza dei dati in operazioni critiche.

Successivamente, si prevede di adottare un meccanismo più sofisticato basato su **messaggistica asincrona tramite code**, che sarà particolarmente utile per gestire scenari specifici, come le **prenotazioni per i ricevimenti**: dove è necessario garantire che più richieste concorrenti non interferiscano. Inoltre, sarà fondamentale garantire che modifiche a stati critici, come lo stato di un utente, vengano effettuate in maniera **sincrona**, evitando situazioni di incoerenza.

Un'ulteriore problematica di concorrenza potrebbe derivare dalla possibilità che uno stesso utente sia connesso contemporaneamente da dispositivi diversi o attraverso nodi differenti del sistema distribuito. In tali casi, le richieste verranno elaborate senza una sincronizzazione esplicita, ma dovranno rispettare requisiti ACID. Questo significa che, in presenza di vincoli specifici definiti a livello di database, solo una delle operazioni concorrenti potrà avere successo.

3.7 Condizioni di boundary

Vengono ora riportate le condizioni di boundary del sistema AstroMark. La condizione di terminazione è omessa, poiché non sono previsti software specifici per la gestione di questa fase, che verrà trattata tramite processi standard di spegnimento e riavvio del sistema.

Configurazione

L'amministratore del sistema dovrà configurare il certificato SSL per web server e opzionalmente delegare la gestione del dominio a CloudFlare configurando opportunamente i record DNS, andranno sostituite le credenziali di accesso al database fornite per il test con quelle utilizzate per il database di produzione. Per accedere alla singola VM è possibile utilizzare SSH e procedere all'apertura delle porte 80 e 443 definendo una specifica regola per il Firewall.

Inizializzazione

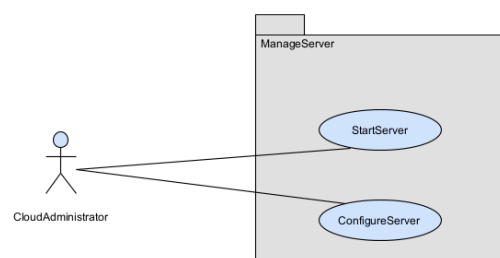
L'amministratore del sistema potrà utilizzare il servizio Amazon EC2 dove sono possibili diverse modalità di inizializzazione del sistema in base anche al numero di nodi scelto. In caso di un solo nodo per il web server e uno per il database questi possono essere avviati individualmente anche da interfaccia grafica. Verrà inoltre fornito un file bash con uno script per permettere di eseguire all'avvio del sistema l'applicazione.

Guasto e gestione eccezione

Nella improbabile ipotesi di un'interruzione del sistema dovuta al fornitore l'applicazione non supporta meccanismi di ripristino completo allo stato precedente. Analogamente, nel caso di un arresto anomalo del software dovuto a errori introdotti durante la fase di implementazione, non sono previste soluzioni automatiche per correggere il problema; l'unica operazione possibile sarà chiudere il sistema e procedere con il riavvio.

Ogni eccezione che si verifica verrà registrata in un file di log per agevolare le attività di diagnosi e debugging. Se l'errore riguarda funzionalità del front-end, sarà comunque consentito tentare una nuova richiesta senza dover riavviare l'intero sistema, offrendo un minimo di continuità operativa per l'utente.

Diagramma dei casi d'uso dell'amministratore



4. Servizi dei sottosistemi

Di seguito sono descritti i servizi offerti dai sottosistemi del sistema AstroMark, evidenziando le funzionalità principali che ciascun sottosistema mette a disposizione per garantire il corretto funzionamento e l'interazione tra le diverse componenti del sistema.

User Management	Servizio	Descrizione
	Notifica errori	Comunica gli errori e/o i problemi agli utenti.
	Interazione con il sistema	Permette agli utenti di inserire e modificare dati attraverso degli elementi di interazione
	Navigazione	Gestisce la struttura e la logica di navigazione dell'applicazione.
	Visualizzazione delle informazioni	Consente la visualizzazione delle informazioni fornite dal sistema

User Management

Servizio	Descrizione
Modifica dati	Permette ad un utente di modificare le proprie informazioni
Preparazione dati basati sull'utente	Prepara i dati da fornire all'interfaccia grafica in base alla tipologia di utente della scuola
Recupero password	Fornisce le operazioni per la modifica della propria password dopo averla dimenticata
Rimozione account	Serie di operazioni per rimuovere un utente dalla scuola

School Management

Servizio	Descrizione
Aggiunta insegnante	Inserimento di un insegnante alla piattaforma
Aggiunta scuola	Creare e aggiungere una nuova scuola
Aggiunta/Rimozione classi	Gestione delle classi che sono presenti nella piattaforma
Aggiunta/Rimozione studenti in una scuola	Gestione degli studenti che sono presenti nella piattaforma
Inserimento genitori o tutori legali	Gestione dei genitori o responsabili degli studenti che sono presenti nella piattaforma
Inserimento informazioni scuola	Permette di inserire informazioni sulla scuola
Inserimento personale segreteria	Gestione degli utenti segreteria che sono presenti nella piattaforma
Recupero contatti scuola	Restituisce le informazioni di contatto della scuola
Rimozione una scuola	Rimuovere una scuola esistente nel sistema

Class Management

Servizio	Descrizione
Acquisizione informazioni classe	Permette di raggruppare e restituire tutte le informazioni associate ad una classe
Estrazione comunicazioni classe	Fornisce tutte le comunicazioni inviate ad una classe
Informazioni piano di studi	Raccoglie i macro-argomenti che verranno affrontati nell'anno scolastico nelle varie discipline
Inserimento di uno studente di una scuola	Inserisce uno studente di una scuola in una specifica classe
Recupero orario della classe	Restituisce l'orario settimanale di una classe
Recupero studenti classe	Restituisce tutte le informazioni che sono associate agli studenti di una classe
Rimozione studente	Rimuove uno studente dalla classe

Agenda

Servizio	Descrizione
Assegnazione di un ora ad un professore	Mappa un professore ad un ora specifica della settimana
Creazione orario scolastico	Formulazione e inserimento dell'orario settimanale per una classe
Firma dell'ora	Permette ad un professore di registrare la propria lezione in un orario a lui assegnato
Modifica l'orario scolastico	Operazioni di modifica ad un orario creato
Recupero informazioni ora	Restituisce le informazioni legate ad una ora specifica di una settimana
Prenotazione ora ricevimento	Crea un appuntamento tra un professore e un genitore/tutore in un'ora della settimana
Rimozione prenotazione	Rimuove un appuntamento tra un professore e un genitore/tutore in un'ora della settimana

Chat

Servizio	Descrizione
Invio messaggio	Consente di inviare del testo in una chat
Invio allegato	Consente di inviare un file in una chat
Ottenimento numero messaggio	Permette di tenere traccia della sequenza dei messaggi

Orientation

Servizio	Descrizione
Consiglio orientamento	Formula una serie di dati utili per l'orientamento in uscita di uno studente

Classwork

Servizio	Descrizione
Aggiunta chat per compito	Fornisce una chat per discussioni al compito assegnato
Assegnazione compito	Inserisce un compito per una classe
Controllo informazioni per compito	Restituisce le informazioni associate al compito
Gestione data scadenza per compito	Imposta una data limite per cui consegnare il compito
Inserimento attività	Permette di inserire informazioni sulle attività svolte in una classe in una specifica ora

Communication

Servizio	Descrizione
Informazioni comunicazione	Fornisce gli strumenti per ottenere i dettagli di una comunicazione
Inserimento comunicazione	Operazioni che permettano di creare una comunicazione e renderla visibile
Modifica comunicazione	Azioni per modificare una comunicazione creata
Rimozione comunicazione	Fornisce la possibilità di eliminare una comunicazione creata

Justification and Behavior

Servizio	Descrizione
Creazione assenza o ritardo	Permette la creazione di una assenza o di un ritardo
Panoramica studente	Fornisce le statistiche dettagliate sui ritardi e le assenze.
Registrazione giustificazione assenza/ritardo	Consente l'inserimento di una nuova giustificazione associata a uno studente per una assenza/ritardo, specificando i dettagli

Rat	Servizio	Descrizione
	Formulazione delle medie per materie e/o periodo temporale	Calcola la media matematica specifica per un periodo di tempo o specifica materia
	Inserimento voto	Consente di inserire una valutazione ad uno studente
	Inserimento voto scrutinio	Consente agli utenti predisposti di inserire un voto in uno scrutinio
	Recupero informazioni scrutinio	Fornisce l'insieme di tutte le informazioni associate ad uno scrutinio
	Recupero voti giornalieri	Fornisce l'insieme delle valutazioni ottenute
	Rimozione voto	Permette di rimuovere una valutazione ad uno studente

Database

Servizio	Descrizione
Mantenimento dati	Archivia e gestisce le informazioni relative agli utenti e alla scuola e le rispettive relazioni
Modifica dati	Permette la modifica dei dati caricati garantendo la consistenza e la coerenza dei dati stessi
Recupero informazioni	Recupera all'interno della base di dati le informazioni richieste riguardo alle entità

Glossario

Termine	Definizione
Amazon EC2	Elastic Compute Cloud
Argo	Soluzione consolidata di gestione didattica utilizzata nelle scuole italiane
Caching	Tecnica per migliorare le prestazioni riducendo l'accesso diretto al database
CDN	Rete per distribuire contenuti statici riducendo la latenza
ClasseViva	Altra piattaforma diffusa per la gestione didattica nelle scuole italiane
CloudFlare R2	Servizio di archiviazione cloud senza costi di uscita.
File Bash	Script utilizzato per automatizzare comandi e operazioni nel terminale Unix/Linux.
JAWS	Screen reader utilizzato per verificare l'accessibilità da tastiera
PostgreSQL	Database relazionale open-source avanzato.
SQL injection	Tecnica di attacco che permette di manipolare il database inviando comandi malevoli.
Vue.JS	Framework JavaScript per la creazione di interfacce utente.