



POLITECNICO DI BARI

DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELL'INFORMAZIONE
Corso di Laurea Triennale in Ingegneria Informatica e dell'Automazione

Tesi di Laurea in Ingegneria del Software e Fondamenti Web

Corridors per l'Indoor Positioning: Progettazione e Pianificazione dello Sviluppo del CMS dedicato

Relatore
Ing. Antonio Ferrara

Laureando
Dario Marchitelli

Anno Accademico 2021 - 2022

Abstract

Negli ultimi anni si è assistito ad una diffusione sempre maggiore di software per la localizzazione e la navigazione all'interno di aree estese. Questi sistemi, che sfruttano il GPS, non sono adatti agli ambienti più piccoli in cui si necessita una precisione maggiore. Corridors è un sistema che permette la localizzazione e la navigazione indoor degli spazi del Politecnico di Bari. L'applicazione mobile mostra una mappa interattiva contenente dei Point Of Interest cliccabili e raggiungibili attraverso il sistema di navigazione integrato. La necessità di affiancare a tale app una dashboard per la gestione degli elementi visibili sulla mappa e la presentazione dei dati sul posizionamento degli utenti, ha portato alla progettazione e allo sviluppo di un CMS dedicato. All'interno di questa tesi sono presentati gli aspetti di progettazione grafica, funzionale e architetturale del CMS, ponendo particolare attenzione al modello utilizzato per il processo di sviluppo.

Indice

1 Introduzione	1
1.1 Progettazione	1
1.2 Pianificazione	2
1.3 Struttura della tesi	2
2 Corridors	5
2.1 Indoor Positioning	5
2.1.1 iBeacon BLE	5
2.1.2 Multilaterazione	6
2.2 L'applicazione mobile	7
2.3 L'SDK Nextome	8
2.3.1 Localizzazione	8
2.3.2 Navigazione	9
2.4 L'architettura dell'app Corridors	10
2.5 Corridors CMS	11
3 Background	13
3.1 Software Development Life Cycle	13
3.2 Software Developement Life Cycle Models	15
3.2.1 Modello a cascata	15
3.2.2 B-Model	16
3.2.3 Modello iterativo	16
3.2.4 RAD e Agile	17
4 Requisiti, casi d'uso e flussi d'interazione	21
4.1 Definizione dei requisiti	21
4.1.1 Requisiti funzionali	21
4.2 Tassonomia dei casi d'uso	22
4.2.1 Casi d'uso per lo studente	23
4.2.2 Casi d'uso per l'utente amministratore	23
4.3 Flussi di interazione	27
4.3.1 Registrazione	27
4.3.2 Login	29
4.3.3 Richiesta API sicura	30
4.3.4 Richiesta di approvazione POI	32

5 Progettazione di Corridors CMS	37
5.1 Progettazione del logo	37
5.2 Progettazione dell'interfaccia utente	37
5.2.1 Topbar	39
5.2.2 Sidebar	39
5.2.3 Struttura delle pagine	40
5.2.4 POI	40
5.2.5 Eventi	42
5.2.6 Heatmap	43
5.2.7 Utenti	43
5.2.8 Richiesta POI	44
5.3 Progettazione di sistema	45
5.3.1 Architettura	45
5.3.2 Database	48
6 Pianificazione dello sviluppo	51
6.1 Joint Application Development	51
6.1.1 Iterazioni di Corridors CMS	52
6.1.2 Suddivisione del lavoro in task	52
6.1.3 Delivery continuo	53
6.1.4 Gold plating	53
6.2 Git	53
6.3 ClickUp	53
6.3.1 Funzionalità generali	54
6.3.2 Il workspace Corridors	54
6.3.3 Vantaggi dell'utilizzo di ClickUp	55
6.3.4 ClickUp per il JAD	56
7 Conclusioni	57

Elenco delle figure

2.1 Portale di gestione del venue Nextome	6
2.2 Rappresentazione grafica dell'algoritmo di trilaterazione	7
2.3 Screenshot dell'app Corridors	8
2.4 Rappresentazione dell'algoritmo di Dijkstra	9
2.5 Architettura Corridors App	10
3.1 Modello a cascata	16
3.2 Modello iterativo	17
3.3 Rapid Application Development	18
4.1 Rappresentazione dei casi d'uso dello studente	23
4.2 Rappresentazione dei casi d'uso dell'amministratore relativi alle CRUD dei POI	24
4.3 Rappresentazione dei casi d'uso dell'amministratore relativi alla visualizzazione dei POI	25
4.4 Rappresentazione dei casi d'uso dell'amministratore relativi alle CRUD degli eventi	26
4.5 Rappresentazione dei casi d'uso dell'amministratore relativi alla visualizzazione degli eventi	26
4.6 Rappresentazione del caso d'uso dell'amministratore relativo agli utenti	27
4.7 Rappresentazione del caso d'uso dell'amministratore relativo al- l'affollamento	27
4.8 Flusso per la registrazione	29
4.9 Flusso per il login	31
4.10 Flusso per la richiesta di un'api sicura	32
4.11 Flusso per la richiesta di approvazione di un POI	34
5.1 Versioni del logo	38
5.2 Logo finale	39
5.3 Topbar	39
5.4 Sidebar	40
5.5 Schermate del CMS relative ai POI	42
5.6 Schermate del CMS relative agli eventi	44
5.7 Visualizzazione posizioni su Heatmap	45
5.8 Visualizzazione utenti su tabella	45
5.9 Schermate del CMS relative alla visualizzazione lato <i>Studente</i> . .	46
5.10 Architettura multi-tier del sistema Corridors	47
5.11 Schema del Database	48

6.1	Whiteboard per il brainstorming	54
6.2	Liste dei task	55

Capitolo 1

Introduzione

Grazie all’evoluzione della tecnologia Bluetooth [1], negli ultimi anni è stato possibile creare tecnologie di posizionamento la cui precisione ha superato di gran lunga quella offerta da un normale sistema di geolocalizzazione [2]. Questa metodologia garantisce un’accuratezza talmente elevata da poter rilevare, in maniera pressoché puntuale, la posizione dell’utente all’interno di una stanza e in uno specifico piano. Una soluzione di questo tipo si adatta perfettamente a spazi fitti e labirintici come quelli del Politecnico di Bari, per il quale è stato progettato e sviluppato un sistema ad hoc, basato sulla tecnologia Bluetooth, che permette la localizzazione e la navigazione indoor.

Corridors è il sistema creato appositamente per il Politecnico di Bari, si tratta di un’applicazione mobile sviluppata per Android basata sulla tecnologia dell’Indoor Positioning. Essa illustra una mappa con diversi punti di interesse, mostra la posizione del dispositivo che la sta eseguendo e consente l’avvio della navigazione verso uno dei punti di interesse visualizzati.

Tutte le informazioni mostrate all’interno di Corridors sono memorizzate e modificabili all’interno di Firebase [3], una piattaforma per la creazione di applicazioni per dispositivi mobile e web sviluppata da Google. La dashboard offerta da Firebase consente una consultazione e una modifica dei dati all’interno del database NoSQL integrato di basso livello. Una gestione di questo tipo è impensabile per tutte le tipologie di utenti, perciò è necessario stabilire una modalità di interazione più semplice e immediata come un CMS¹ ad hoc: Corridors CMS.

Questo elaborato descrive, analizza e giustifica le scelte di pianificazione e progettazione attuate per lo sviluppo di Corridors CMS, un sistema informativo articolato che utilizza le più innovative tecnologie per lo sviluppo web che si interfacciano fra di loro mediante l’utilizzo di REST API. Questo progetto costituisce solo una parte dell’intero sistema, comportandosi come un gestore delle informazioni che si interfaccia direttamente con le componenti che si occupano della localizzazione.

1.1 Progettazione

Uno dei temi principali attorno ai quali verte questa tesi è la progettazione. Essa è definita come un processo logico volto a realizzare un prodotto che soddisfi

¹Content Management System

in modo ottimale requisiti esplicativi (relativi a obiettivi e a vincoli tecnologici, di costo, tempo, qualità) vigenti nell'intero ciclo di vita del prodotto, mediante una sequenza di scelte, qualitative e quantitative, basate sulla tecnologia disponibile. La progettazione si svolge in settori legati alla realizzazione e gestione di strutture organizzative (anche in ambiti non tecnologici), alla pianificazione economica o delle attività, alla gestione di risorse economiche, umane, fisiche, naturali, informative, di tempo.[\[4\]](#)

Questo elaborato tratterà il tema della progettazione, che rappresenta una parte fondamentale all'interno del ciclo di vita di un progetto software. In particolare, verranno descritti e trattati i requisiti funzionali, i casi d'uso e i flussi di interazione con l'utente del CMS. In seguito saranno evidenziati anche gli aspetti relativi alla progettazione del design e dell'architettura del sistema.

1.2 Pianificazione

Un altro tema fondamentale, approfondito in questo documento, è la pianificazione. Essa è definita come la capacità di identificare obiettivi e priorità, di organizzare le risorse per il futuro e di distribuire le responsabilità in modo appropriato ed efficiente. Significa essere in grado di programmare ed implementare attività di diverso genere, di ottimizzare costi e ricavi, di gestire in maniera efficace le risorse assegnate, di mantenere gli impegni e le scadenze e di valutare l'efficacia del lavoro svolto. Più precisamente, pianificare significa formulare previsioni, stabilire obiettivi e strategie appropriate per raggiungerli. Chi si occupa di pianificazione riesce facilmente a formulare budget, a programmare e ad implementare procedure. Così la capacità organizzativa può essere vista come l'attitudine a identificare le funzioni lavorative chiave per raggiungere i risultati richiesti e a utilizzare specifici metodi organizzativi (delega, risorse in outsourcing, ecc...) per conseguire gli obiettivi indicati. La competenza di pianificazione e organizzazione indica anche il grado di attenzione del soggetto verso la qualità/quantità del lavoro da svolgere.[\[5\]](#)

Nel ambito della pianificazione dello sviluppo, si definisce un modello da seguire durante tutte le fasi del progetto. In questo elaborato verranno motivate e documentate le scelte progettuali relative al modello utilizzato e le piattaforme che hanno permesso di gestirne i meccanismi e i processi correlati.

1.3 Struttura della tesi

I capitoli che seguono sono così strutturati:

Capitolo 2

Nel secondo capitolo si introduce Corridors, l'applicazione mobile su cui è stato costruito il CMS oggetto di questa tesi. In particolare si descrive la tecnologia dell'Indoor Positioning su cui essa si basa, si approfondiscono tutte le sue funzionalità e se ne presenta l'architettura, che comprende anche l'SDK fornito da Nextome per l'implementazione dell'Indoor Positioning.

Capitolo 3

Nel terzo capitolo si descrive il ciclo di vita dello sviluppo di un software (SDLC²) e le fasi che lo compongono, si fa riferimento ai modelli di SDLC più utilizzati, con una particolare attenzione al *Rapid Application Development*.

Capitolo 4

Nel quarto capitolo si illustra la prima fase della progettazione con i requisiti funzionali del CMS che permettono di delinearne preliminarmente le funzionalità. In seguito, si sono definiti i casi d'uso del sistema per ogni tipologia di utente e i flussi di interazione mediante dei workflow, così da descrivere dettagliatamente il comportamento atteso del sistema.

Capitolo 5

Il quinto capitolo si focalizza sulla progettazione grafica e dell'architettura del CMS. In particolare viene descritto il design del logo, la struttura delle pagine e le scelte progettuali relative all'architettura del sistema e ai componenti che la compongono, dettagliando anche la struttura del database.

Capitolo 6

Il sesto capitolo descrive in dettaglio il modello di SDLC utilizzato per lo sviluppo di Corridors CMS mettendone in luce i vantaggi e le peculiarità. Si illustrano anche i sistemi che hanno supportato il team durante lo sviluppo e come essi si sono rivelati fondamentali per la buona riuscita del progetto.

Capitolo 7

All'interno del settimo capitolo si giunge alle conclusioni di questo elaborato di tesi e si fa riferimento ai possibili sviluppi futuri.

²SDLC: Software Development Life Cycle

Capitolo 2

Corridors

Corridors CMS è stato concepito per affiancare un applicazione mobile già esistente sviluppata per Android: Corridors.

Corridors è un sistema di Indoor Positioning utilizzato all'interno del Politecnico di Bari che sfrutta la tecnologia messa a disposizione dall'azienda Nextome per offrire un servizio di localizzazione e navigazione all'utente che lo utilizza.

2.1 Indoor Positioning

L'applicazione Corridors può prendere vita solo grazie a un'importantissima tecnologia abilitante: l'Indoor Positioning. In particolar modo, la soluzione adottata è quella offerta da Nextome [6], azienda di Conversano (BA) specializzata nel delivery e setup di ISP¹.

Un Indoor Positioning System rappresenta un insieme di apparati che consentono la localizzazione di oggetti e/o persone in un ambiente dove sistemi come il GPS perdono di efficacia, come avviene in luoghi chiusi o sotterranei. Esistono numerosi approcci che rendono possibile l'indoor positioning e ognuno di essi adotta algoritmi e/o tecnologie differenti, come: Wi-Fi, RFID, Bluetooth, Zigbee, Ultra Wideband (UWB).

La tecnologia adottata da Nextome è quella del Bluetooth Low Energy (BLE). Grazie alla presenza di un'infrastruttura, formata da diversi dispositivi, opportunamente installata nell'ambiente indoor e di un'app che incorpora le componenti software che sono in grado di interagire con tale infrastruttura, l'utente può localizzarsi all'interno di un qualsiasi ambiente con il proprio smartphone. Per fare ciò, bisogna realizzare un'infrastruttura che preveda l'installazione di una serie di dispositivi BLE, chiamati *iBeacon*, posizionati strategicamente all'interno dell'ambiente indoor.

2.1.1 iBeacon BLE

Si tratta di dispositivi molto compatti alimentati a batteria che emettono periodicamente un segnale Bluetooth che contiene informazioni precedentemente memorizzate [7] e che identificano il dispositivo stesso:

¹ISP: Indoor Positioning System

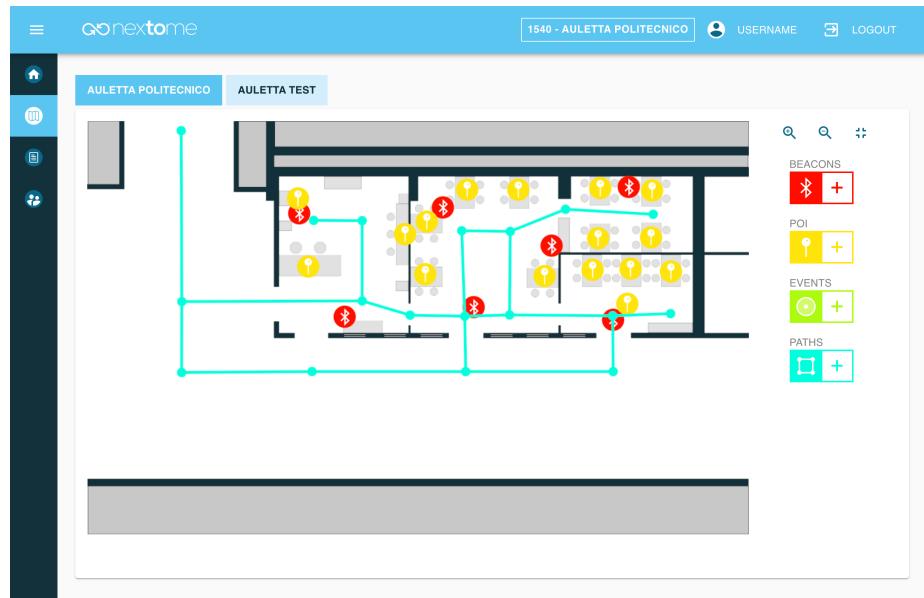


Figura 2.1: Portale di gestione del venue Nextome

- Major
- Minor
- UUID

Ogni iBeacon installato va registrato opportunamente sul portale messo a disposizione da Nextome: la pagina di configurazione presenta una mappa in cui è rappresentato in 2D l'ambiente che si vuole prendere in considerazione, come mostrato in Figura 2.1. Non bisogna far altro che aggiungere il beacon sulla mappa, posizionandolo quanto più precisamente possibile, per poi compilare correttamente il campo UUID con quello del corrispettivo beacon installato nell'ambiente reale. Conoscere la posizione di tutti i beacon è di fondamentale importanza per la componente software presente nello smartphone che si occuperà di localizzare l'utente sulla mappa 2D in base ai beacon presenti nelle vicinanze.

2.1.2 Multilaterazione

La soluzione di Indoor Positioning adottata prevede che l'app ricavi la posizione dell'utente in tempo reale attraverso un algoritmo RSSI-based, detto algoritmo di multilaterazione (generalizzazione del più semplice algoritmo di trilaterazione). Affinché si possa parlare di multilaterazione, bisogna supporre che l'utente sia situato in prossimità di almeno tre beacon. In realtà, il sistema cerca di restituire un risultato anche se l'utente si trova in prossimità di uno o due beacon, ma si tratta di risultati generalmente non attendibili.

²RSS: Received Signal Strength

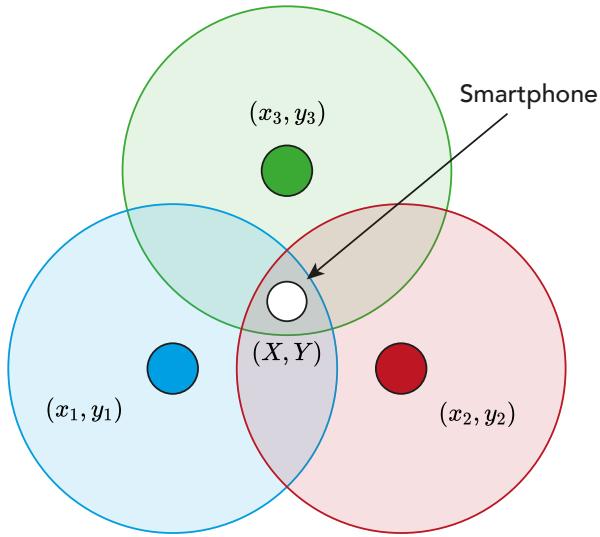


Figura 2.2: Rappresentazione grafica dell’algoritmo di trilaterazione

Per poter applicare l’algoritmo, l’utente deve trovarsi nell’intersezione di almeno tre cerchi, rappresentanti il raggio di azione dei singoli beacon. L’app conosce le posizioni dei beacon, indicate in Figura 2.2 con $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$, ma non conosce ancora la posizione relativa dello smartphone, indicata con (X, Y) . Quest’ultima posizione viene stimata sulla base del calcolo dall’attenuazione dei segnali ricevuti dai beacon limitrofi. Una volta stimata la distanza dei beacon rispetto allo smartphone, l’app può stimare la posizione di quest’ultimo (e quindi dell’utente che lo utilizza) sulla mappa 2D. In generale, all’aumentare del numero dei beacon coinvolti nel calcolo, l’accuratezza del posizionamento aumenta. È possibile osservare un’applicazione dell’algoritmo in [8].

2.2 L’applicazione mobile

L’applicazione mobile di Corridors rappresenta l’accesso più immediato che l’utente ha per usufruire dei servizi del sistema di Indoor Positioning del Politecnico di Bari. Allo stadio corrente, l’app fornisce all’utente quelle che sono le funzionalità più basilari previste in fase di progettazione:

- Visualizzare una mappa interattiva; (Figura 2.3a) in cui l’utente possa localizzarsi;
- Cercare un POI tra quelli sulla mappa mediante un input box di ricerca come mostrato in Figura 2.3c;
- Filtrare i POI visualizzati per categoria;
- Visualizzare i dettagli di un POI dalla modale apposita posizionata in basso (Figura 2.3b);

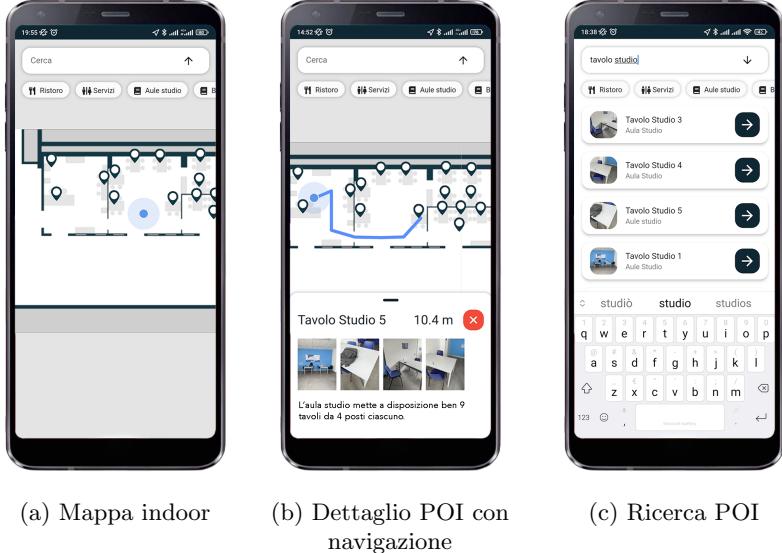


Figura 2.3: Screenshot dell'app Corridors

- Avviare la navigazione presso il POI selezionato (Figura 2.3b).

2.3 L'SDK Nextome

Fondamentale per lo sviluppo vero e proprio dell'app Corridors è stato l'SDK³ Nextome [9]: esso permette lo sviluppo di app Android o iOS che prevedono casi d'uso relativi alla tecnologia di Indoor Positioning. L'integrazione dell'SDK nell'app Corridors è stata effettuata adottando i parametri suggeriti dalla documentazione ufficiale [9]. L'SDK si occupa di diverse mansioni tra cui:

- Interagire con i server di Nextome per scaricare le ultime versioni delle risorse che verranno poi mostrate su mappa;
- Calcolare e aggiornare la posizione corrente dell'utente sulla mappa;
- Mettersi in ascolto di specifici eventi (l'avvenuto download delle risorse, la rilevazione di una nuova posizione dell'utente, il passaggio da un ambiente indoor ad outdoor, ecc.).

2.3.1 Localizzazione

Tra le funzioni appena citate, l'SDK è la componente dell'app che applica l'algoritmo di localizzazione. L'algoritmo viene eseguito periodicamente dall'app per determinare la posizione dell'utente in real-time (circa 200ms di refresh), senza interagire con i server Nextome. Una volta scaricate tutte le risorse (mappa, posizione dei beacon, posizione dei POI, ecc.), l'SDK dispone di tutti i dati necessari per l'applicazione dell'algoritmo e riesce a calcolare la posizione dell'utente anche in assenza di connessione ad Internet. Inoltre, supponendo che

³SDK: Software Development Kit

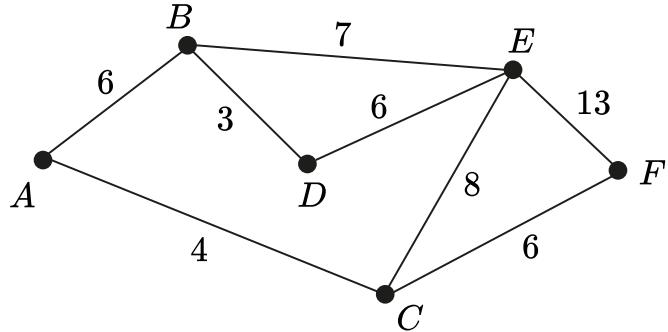


Figura 2.4: Rappresentazione dell'algoritmo di Dijkstra

il sistema sia già stato popolato con i dati relativi all'ambiente in questione e sapendo che le risorse appena citate sono soggette ad aggiornamenti in genere poco frequenti, l'utente sarebbe in grado di potersi localizzare anche in modalità offline, sfruttando la versione delle risorse presente in cache.

2.3.2 Navigazione

L'SDK, inoltre, ha permesso di implementare all'interno dell'app Corridors la funzionalità di navigazione indoor: l'utente può avviare la navigazione verso un POI target selezionato o una posizione custom selezionabile attraverso una pressione prolungata sulla mappa.

È importante che l'applicazione determini un percorso veritiero, ossia un percorso che non attraversi muri e/o altre barriere e che quindi sia realmente percorribile. Per fare ciò devono essere definiti tutti i possibili percorsi sulla mappa presente sul portale Nextome (Figura 2.1). Una volta definiti i vari percorsi (come quelli in azzurro in Figura 2.1), l'SDK è in grado di calcolare il percorso migliore per raggiungere la posizione desiderata attraverso l'applicazione dell'algoritmo di Dijkstra.

Dijkstra

L'algoritmo di Dijkstra [10] rappresentato in Figura 2.4 è un algoritmo che permette di trovare il percorso più breve tra due nodi di un grafo pesato. Un grafo pesato è definito da un insieme di nodi interconnessi tra loro mediante degli archi pesati. Gli archi sono detti pesati in quanto viene associato ad ognuno di essi un certo valore non negativo che, in generale, può rappresentare qualsiasi grandezza che distingua un arco da un altro. Il peso può indicare una qualsiasi grandezza misurabile come ad esempio una lunghezza, un tempo o altro ancora. Nel caso in esame:

- un nodo rappresenta un POI e/o punti generici di congiunzione di due o più archi;

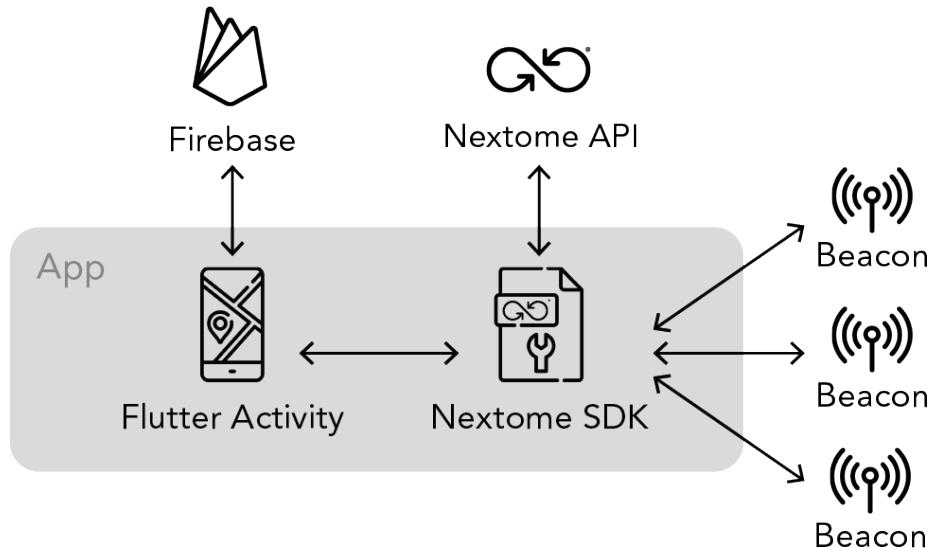


Figura 2.5: Architettura Corridors App

- un arco rappresenta il percorso fisico che collega due nodi: il peso degli archi è dato dalla lunghezza effettiva del percorso fisico rappresentato dal singolo arco.

2.4 L'architettura dell'app Corridors

L'app Corridors è stata strutturata in tre componenti principali, rappresentate in Figura 2.5

- **Componente nativa:** contiene l'integrazione dell'SDK, ovvero tutta la logica necessaria ad ascoltare gli eventi lanciati dall'SDK e le varie implementazioni degli event handler che inoltrano le informazioni ricevute al componente Flutter;
- **Componente Flutter:** contiene il componente mappa, la grafica che la circonda (barra di ricerca, bottom sheet con dettagli POI, ecc.) e la logica necessaria a gestire gli input utente e gli input provenienti dalla parte nativa;
- **Firebase:** è utilizzato per l'autenticazione degli utenti e la memorizzazione di alcuni dati relativi ai POI che non possono essere memorizzati sui server Nextome.

Al momento l'app si interfaccia direttamente con i server Nextome e con Firebase per ricavare le informazioni relative ai POI. Tuttavia, visto l'elevato grado di personalizzazione che si è reso necessario affinché si potessero realizzare certi requisiti funzionali, si è deciso di realizzare un intero portale di gestione dedicato al sistema Corridors.

2.5 Corridors CMS

La necessità di sviluppare un CMS apposito scaturisce dal fatto che la gestione delle informazioni mediante Firebase risulta scomoda, non intuitiva e difficilmente personalizzabile.

In precedenza tutti i dati mostrati all'interno dell'applicazione mobile erano modificabili solo mediante la dashboard di Firebase, che offre un'interazione di basso livello con il database NoSQL integrato. Per questo motivo, chiunque avesse voluto creare un punto d'interesse nella mappa avrebbe dovuto operare direttamente nel database aggiungendo un nuovo elemento alla collezione, specificandone volta per volta gli attributi chiave-valore.

Una gestione di questo tipo è impensabile per utenti convenzionali, perciò è necessario stabilire una modalità di interazione con il sistema informativo che sia più semplice e accessibile a tutti. A tale scopo, in questo lavoro di tesi si è sviluppato un CMS, accessibile dal browser, che possa consentire a chiunque di interfacciarsi con il database di Corridors in modo semplice e intuitivo.

Corridors CMS ha l'obiettivo di gestire l'insieme delle risorse coinvolte nell'intero sistema, al momento riassumibili in: POI, eventi, posizioni e utenti. La presente Tesi di Laurea si focalizza sulle modalità di progettazione del CMS e ne descrive la pianificazione dello sviluppo.

Per ulteriori approfondimenti sugli altri aspetti del CMS si rimanda alle Tesi di Laurea di Masciullo Paolo, Laera Sante e Di Bari Vito che trattano e approfondiscono rispettivamente:

- Progettazione dell'architettura e sviluppo del backend in SpringBoot;
- Progettazione e sviluppo dell'interfaccia utente in React;
- Progettazione e implementazione dell'infrastruttura di sicurezza.

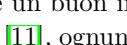
Capitolo 3

Background

L'obiettivo di questa tesi è quello di trattare e approfondire gli aspetti relativi a progettazione e pianificazione di Corridors CMS. Di seguito saranno presentati i concetti teorici principali riguardanti il ciclo di vita dello sviluppo di un prodotto software e sarà fornita una panoramica dei modelli di sviluppi più utilizzati.

3.1 Software Development Life Cycle

Il Software Development Life Cycle (SDLC) è uno dei punti chiave nello sviluppo di un progetto software ed è l'applicazione di pratiche business standard allo sviluppo di un software. Esso delinea i task richiesti per costruire l'intera applicazione, aiutando a ridurre gli sprechi e ad incrementare l'efficienza dello sviluppo e la qualità del software. Prevede anche il monitoraggio dell'avanzamento dei lavori permettendo così al progetto di progredire seguendo la giusta direzione e continuando ad essere un buon investimento per l'azienda.

Il SDLC è strutturato in 6 fasi , ognuna delle quali si occupa di un aspetto diverso dello sviluppo.

Pianificazione

La fase di pianificazione inizia con il riconoscimento dei requisiti che il software deve soddisfare. In questo frangente verranno generati molteplici concept che soddisfano i requisiti individuati, fra i quali verrà scelto il più opportuno. In questa fase vengono definite anche delle soluzioni di design (sotto forma di disegni, modelli e prototipi), una stima delle risorse umane richieste e una schedulazione preliminare del progetto.

Sviluppo

La fase di sviluppo inizia con il raffinamento, con sufficienti dettagli tecnici, dei requisiti di sistema e di design così da trasformarli in un prodotto usufruibile durante la fase di utilizzazione. Il sistema sarà semplicemente un prototipo che può essere sottoposto agli stakeholders e a coloro che useranno il sistema in modo che possano fornire dei feedback mediante delle recensioni tecniche. In questa fase vengono anche specificati e stabiliti alcuni degli aspetti delle successive fasi e

i sistemi che li ”abilitano”. L’output sarà un sistema o un prototipo del sistema finale assieme alla documentazione e alle stime dei costi delle fasi future.

Produzione

Questa fase inizia con l’approvazione definitiva del sistema e si sviluppa con la distribuzione e la produzione, che possono avvenire individualmente o mediante una produzione di massa. In questo step il sistema può subire miglioramenti o re-design in seguito alla modifica di alcuni dei requisiti specificati nelle fasi precedenti.

Utilizzazione

La fase di utilizzazione inizia subito dopo l’installazione e la messa a punto del sistema e prevede che il prodotto venga usato nel posto per il quale è stato pensato e per gli usi prestabiliti, offrendo i servizi richiesti. Questo step include i processi relativi a:

- Utilizzo del sistema;
- Monitoraggio delle performance;
- Identificazione, classificazione e segnalazione di anomalie, mancanze, e fallimenti.

La risposta ai problemi individuati può essere di 3 tipi:

- Non adottare alcuna contromisura;
- Effettuare migliorie minori che non vanno a stravolgere l’intero sistema saltando alla fase di supporto;
- Effettuare modifiche di ordine maggiore tornando alle fasi di sviluppo e produzione;
- Dismettere il sistema.

Questa fase termina quando il sistema viene spento o messo fuori servizio.

Supporto

La fase di supporto inizia con la fornitura di servizi di manutenzione, logistica e altri meccanismi di supporto all’utilizzo del sistema. Questo step comprende le procedure da mettere in atto nel caso in cui certe anomalie, mancanze o fallimenti si verifichino e definisce i sistemi di supporto utilizzati.

Ritiro

Durante la fase di ritiro è prevista la rimozione del sistema e di tutti i relativi servizi operativi e di supporto. Questo step può essere causato da:

- Perdita di interesse degli stakeholders;
- Sostituzione del sistema con uno nuovo avente migliori performance;

- Danni irreparabili;
- Fallimenti catastrofici;
- Aumento spropositato dei costi.

3.2 Software Developement Life Cycle Models

Esistono vari modelli che descrivono i diversi approcci al Software Developement Life Cycle. Un modello di SDLC è usato generalmente per descrivere gli step che si susseguono durante il ciclo di vita del sistema. Questi modelli possono essere di tre tipi:

- **Lineare:** rappresenta un modello di tipo sequenziale in cui la fine di una fase è definitiva e porta alla fase successiva;
- **Iterativo:** questo tipo di modello consente e standardizza il ritorno ad una fase precedente in maniera ciclica; è basato sull'idea che un sistema, durante il suo ciclo di vita, avrà necessità di evolversi e migliorare continuamente;
- **Ibrido:** questi modelli combinati linear-iterativi stabiliscono che il processo di iterazione delle fasi, ad un certo punto, si fermerà.

Di seguito verranno descritti i modelli più importanti e popolari che hanno rivoluzionato il mondo dello sviluppo software:

3.2.1 Modello a cascata

Il modello a cascata (*waterfall model*) rappresentato in [Figura 3.1](#) è stato documentato per la prima volta da Bennington [\[12\]](#) nel 1956 e migliorato nel 1970 da Winston Royce [\[13\]](#). Ha rappresentato la base per tutti i successivi modelli in quanto sostiene che i requisiti di sistema vadano definiti e analizzati prima di qualsiasi progettazione e sviluppo.

Il modello a cascata è una metodologia di sviluppo lineare in cui ogni fase finisce prima che la successiva cominci [\[14\]](#). È composto da sette fasi:

- **Studio di fattibilità:** consiste nell'analisi e nella valutazione sistemistica delle caratteristiche, dei costi e dei possibili risultati del progetto sulla base di un'idea preliminare ed è finalizzato a mettere in luce i punti di forza e quelli di debolezza del progetto definendone la probabilità di successo. Se quest'ultimo parametro viene ritenuto adeguato allora si passa alla fase successiva;
- **Definizione dei requisiti:** questo modello richiede che i requisiti siano ben documentati prima dell'inizio di qualsiasi altra fase;
- **Design e progettazione:** comprende la progettazione sia fisica che logica del sistema;
- **Implementazione:** consiste nell'effettiva fase di sviluppo e di scrittura del codice;

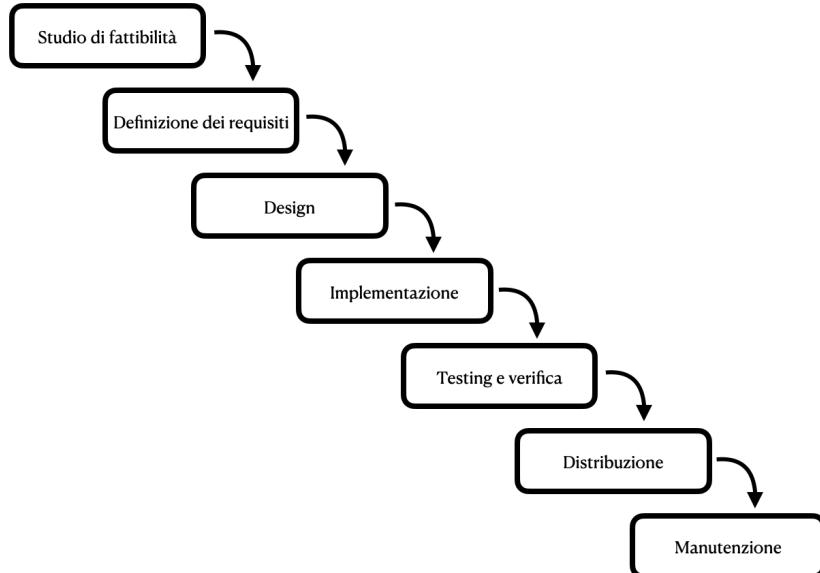


Figura 3.1: Modello a cascata

- **Testing e verifica:** consiste nel testare il software e verificare che i requisiti documentati nella seconda fase siano stati soddisfatti;
- **Distribuzione:** il codice viene trasferito nell'ambiente di produzione;
- **Manutenzione:** il team corregge i difetti del software e alleggerisce il codice caricando in produzione il codice aggiornato.

Royce riconoscendo che possono verificarsi situazioni non previste nelle fasi successive a quella di progettazione ha migliorato questo modello prevedendo una retroazione ad ogni fase, così che si possa tornare alle fasi precedenti sotto certe condizioni.

3.2.2 B-Model

Elaborato nel 1988 da Birrel e Ould [15], rappresenta una estensione del modello a cascata. Il B-Model aggiunge al modello precedente un *Maintenance Cycle* che consiste nella ripetizione delle fasi precedenti in maniera ciclica. L'idea alla base di questo modello è quella di assicurare un miglioramento costante del sistema che sia direttamente parte della fase di sviluppo.

3.2.3 Modello iterativo

Questo modello rappresentato in Figura 3.2 consente lo sviluppo del sistema progettando, sviluppando e testando un componente per volta. Esso infatti non richiede la definizione di tutti i requisiti necessari prima dell'inizio delle fasi di design e sviluppo. Estende il modello a cascata prevedendo la possibilità di effettuare più iterazioni fra le fasi del SDLC. Presenta diversi punti di forza:

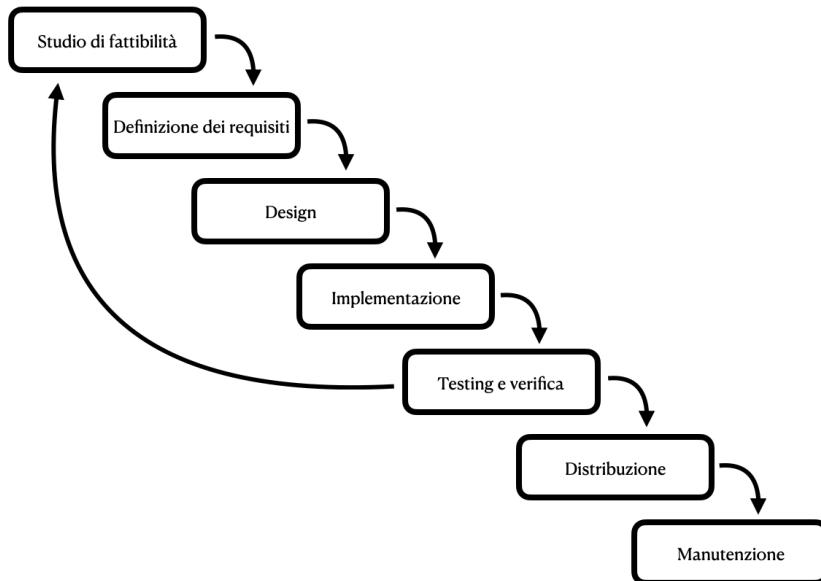


Figura 3.2: Modello iterativo

- Le risoluzioni a criticità riscontrate nelle iterazioni precedenti possono essere incorporate nell'iterazione corrente.
- Gli stakeholder possono essere coinvolti durante le iterazioni per individuare in anticipo problemi e rischi architetturali.
- È flessibile rispetto al cambio o all'aggiunta di requisiti.

3.2.4 RAD e Agile

Il *Rapid Application Development* rappresentato in Figura 3.3 è stato sviluppato da James Martin nel 1991 [16]. È una metodologia che usa la prototipazione come un meccanismo per lo sviluppo iterativo. Il RAD promuove un'atmosfera collaborativa in cui gli stakeholders partecipano attivamente alla fase di prototipazione e di testing.

Recentemente, RAD è stato utilizzato in un senso più ampio e generico che comprende una varietà di modelli volti ad accelerare lo sviluppo del software:

Agile

Secondo l'approccio Agile le modifiche ai requisiti e alle funzionalità vengono evitate suddividendo un progetto in sottoprogetti più piccoli. Lo sviluppo avviene a brevi intervalli e le versioni del software vengono rilasciate di volta in volta guadagnando piccole modifiche e aggiunte incrementali.

Applicare questo approccio ai progetti di grandi dimensioni può essere problematico perché richiede una comunicazione in tempo reale, preferibilmente su base personale, faccia a faccia. Inoltre, i metodi Agile producono poca

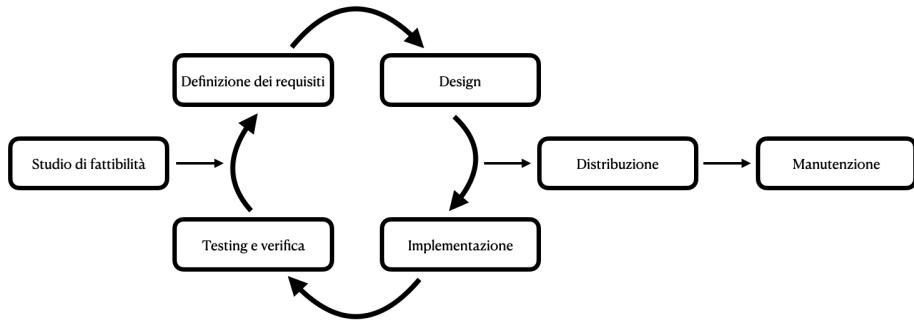


Figura 3.3: Rapid Application Development

documentazione durante lo sviluppo (richiedendo una quantità significativa di documentazione post-progetto).

Extreme Programming (XP)

XP [17] è una metodologia di sviluppo software orientato primariamente alle esigenze del cliente. Si basa sul TDD¹ che consiste nello scrivere i test prima ancora di iniziare a sviluppare la parte funzionale. Secondo questo metodo si tende a scrivere soltanto il codice strettamente necessario e tutto ciò che non è indispensabile può essere aggiunto in un secondo momento.

Una volta sviluppata una determinata funzionalità, il codice prodotto viene trasferito nell'ambiente comune di sviluppo e vengono fatti i partire i test automatici, ciò consente di volta in volta l'integrità del codice sorgente e di dare al team la possibilità di accedere in qualsiasi momento ad una versione funzionante del software, questa pratica è chiamata *Continuous Integration*. Una delle tecniche più conosciute fra quelle introdotte dall'Extreme Programming è il *Pair Programming*. Questa pratica è basata sull'idea che due sviluppatori lavorino insieme alla scrittura del codice: uno assume il ruolo di "guidatore" e l'altro di "osservatore".

Joint Application Development (JAD)

JAD è improntato allo sviluppo collaborativo con l'utente finale o il cliente, coinvolgendolo durante le fasi di progettazione e sviluppo attraverso workshop (noti come sessioni JAD).

Questa tecnica ha uno svantaggio dovuto alla possibilità di creare *scope creep*: può succedere che il cliente, durante le fasi di design e sviluppo, richieda sempre e continuamente l'aggiunta o la modifica di requisiti, cambiando di volta lo scope e portando a costi di sviluppo o di manutenzione non previsti.

Lean Development (LD)

Il paradigma alla base del Lean Development è “80% today is better than 100% tomorrow”, per questo motivo con LD si cerca sempre di consegnare un progetto in anticipo anche in assenza di alcune delle funzionalità richieste.

¹TDD: Test Driven Development

Scrum

Il termine *Scrum* viene utilizzato per la prima volta in un articolo di Nonaka e Takeuchi [18] e rappresenta un framework agile, incrementale e iterativo per lo sviluppo di prodotti, applicazioni e servizi. Scrum [19] è un vero e proprio framework e si basa sull'empirismo: la conoscenza deriva dall'esperienza e le decisioni vanno prese alla luce di ciò che si conosce.

I tre pilastri che sostengono l'empirismo sono:

- **Trasparenza:** tutti coloro che partecipano ad un progetto sanno qual è lo scopo (trasparenza verticale) e sanno che cosa fanno gli altri (trasparenza orizzontale). La trasparenza indica che il lavoro e le misure delle loro performance sono visibili a tutti;
- **Ispezione:** basato sul concetto di controllo empirico del processo, con ciò si intende che ogni iterazione ed incremento viene verificato in base alle metriche di misurazione decise per apportare modifiche alle iterazioni successive rendendo estremamente adattabile l'andamento del processo;
- **Adattamento:** è la conseguenza dell'ispezione in quanto il team di sviluppo, piuttosto che seguire un piano preordinato, ripianifica il lavoro in base ai risultati dell'ispezione per apportare il maggior valore al cliente finale orientato al miglioramento continuo delle proprie performance.

La struttura del framework è composta da:

Responsabilità e ruoli All'interno di *Scrum* esistono tre ruoli distinti:

- Product Owner;
- Scrum Master;
- Developer.

Tutti insieme formano lo Scrum Team, che ha la caratteristica di essere autogestito e cross-funzionale, vale a dire che ha, al suo interno, tutte le competenze per rilasciare un incremento di software senza dipendere da team esterni o da persone esterne al team.

Artefatti e documentazione Per artefatti si intende le modalità con cui Scrum visualizza il lavoro e il “valore”. Questi artefatti sono: Product Backlog, Sprint Backlog e Increment. Ogni artefatto include un “commitment”, con il fine di migliorare la trasparenza e attraverso il quale è possibile misurare i progressi del team. Questi commitment sono:

- **Product Goal:** fa parte del Product Backlog e descrive uno stato futuro del prodotto, e viene posto come obiettivo a lungo termine;
- **Sprint Goal:** è parte dello Sprint Backlog e descrive il motivo per cui l'iterazione aggiunge valore al prodotto ed è considerato un obiettivo più a breve termine;
- **Definition of Done:** descrive quanto un incremento software raggiunge uno standard qualitativo adeguato a consentire il rilascio all'utente finale.

Eventi e meeting I meeting in *Scrum* sono i seguenti:

- **Sprint Planning:** evento di pianificazione che avviene all'inizio dello Sprint (iterazione);
- **Daily Scrum:** evento giornaliero di 15 minuti in cui gli sviluppatori modificano e correggono i loro piani per il raggiungimento dello Sprint Goal;
- **Sprint Review:** evento che chiude uno Sprint e dà l'occasione di ispezionare il lavoro svolto;
- **Sprint Retrospective:** ultimo meeting legato allo Sprint che ha la funzione di ispezionare i processi, le pratiche e altri aspetti legati alla collaborazione.

I modelli appena descritti rappresentano le metodologie più popolari per lo sviluppo di un software; ognuno di essi presenta diversi vantaggi e svantaggi che lo possono rendere più o meno adatto ad un determinato progetto. In fase di progettazione è importante individuare il modello di SDLC che permetta al software di essere sviluppato in modo da sfruttare al massimo le risorse a disposizione e fornire al cliente un risultato che soddisfi le sue aspettative.

Capitolo 4

Requisiti, casi d'uso e flussi d'interazione

In questo capitolo verrà illustrata la prima parte della fase di progettazione di Corridors CMS nella quale verranno descritte le metodologie utilizzate per la definizione delle funzionalità offerte dal portale. In particolare saranno specificati i requisiti, i diagrammi dei casi d'uso e i flussi di interazione con gli utenti.

4.1 Definizione dei requisiti

A seguito della fase di brainstorming e di analisi iniziale, il team ha prodotto come risultato una lista dei requisiti che il software avrebbe dovuto soddisfare. Un requisito secondo IEEE¹ può essere [20]:

- Una condizione o una capacità di cui un utente ha bisogno per risolvere un problema o raggiungere un determinato obiettivo;
- Una condizione o una capacità che deve essere rispettata o posseduta da un sistema o da un suo componente per soddisfare uno contratto, uno standard, specifiche, o altri documenti formalmente imposti;
- Una rappresentazione documentata di una condizione o di una capacità specificate nei punti precedenti.

4.1.1 Requisiti funzionali

Un requisito funzionale è un requisito che specifica una funzione che un sistema o un suo componente deve poter eseguire [20]. Di seguito saranno illustrati i requisiti funzionali di Corridors CMS, suddivisi per ruolo, individuati nella fase di progettazione.

Admin

- L'amministratore deve visualizzare tutti gli utenti registrati;

¹IEEE: Institute of Electrical and Electronics Engineers

- L'amministratore può creare un POI;
- L'amministratore deve poter modificare i dati relativi ad un POI;
- L'amministratore deve poter validare un POI;
- L'amministratore può eliminare un POI;
- L'amministratore deve avere a disposizione una visualizzazione tabellare dei POI;
- L'amministratore deve poter vedere una mappa che rappresenti tutti i POI;
- L'amministratore deve poter applicare un filtro che mostri i POI non validati nella visualizzazione tabellare;
- L'amministratore può creare un evento collegandolo ad un specifico POI;
- L'amministratore deve avere a disposizione una visualizzazione tabellare degli eventi;
- L'amministratore deve poter modificare i dati relativi ad un evento;
- L'amministratore deve poter applicare un filtro che mostri gli non validati nella visualizzazione tabellare;
- L'amministratore deve poter validare un evento;
- L'amministratore può eliminare un evento;
- L'amministratore può vedere il livello di affollamento degli spazi.

Studente

- Lo studente deve avere una visualizzazione a mappa di tutti i POI validati;
- Lo studente, cliccando su un punto della mappa può richiedere l'inserimento di un POI.

4.2 Tassonomia dei casi d'uso

Come possiamo immaginare, nessun sistema software è isolato. Qualsiasi applicativo deve necessariamente interagire con un endpoint (umano o automatico) che lo utilizza per effettuare determinate azioni o perseguire scopi prefissati; l'utente quindi si aspetta che il sistema si comporti in maniera prevedibile.

Un caso d'uso [21] descrive il comportamento di un sistema o di una sua parte e rappresenta una descrizione di una serie di sequenze di azioni che esso esegue per fornire un risultato osservabile ad un attore. I casi d'uso sono utilizzati per schematizzare il corretto funzionamento del software che si sta sviluppando senza specificare come i comportamenti del sistema siano stati effettivamente implementati. Questo consente agli sviluppatori, agli utenti finali e ai committenti di avere una visione condivisa delle features, slegandole dal background culturale personale.



Figura 4.1: Rappresentazione dei casi d'uso dello studente

I casi d'uso sono anche utili a verificare l'avanzamento dei lavori e stabilire se si sta andando nella direzione giusta durante la fase di sviluppo.

Gli attori sono rappresentati da un insieme coerente di ruoli che gli utenti possono impersonare durante l'utilizzo del software. Negli schemi che verranno descritti di seguito saranno rappresentati i casi d'uso per i ruoli di *Studente* e *Admin*.

4.2.1 Casi d'uso per lo studente

I casi d'uso relativi all'utente di tipo *Studente* rappresentati in Figura 4.1 sono legati a delle semplici azioni che chiunque si registri al portale può effettuare.

Visualizzazione a mappa di tutti i POI All'interno di Corridors CMS lo studente deve avere la possibilità di consultare i POI presenti nella mappa, vederne le informazioni principali e, ovviamente, la loro posizione. Ciò consentirebbe all'utente di individuare la mancanza di un particolare POI di suo interesse e di richiederne l'aggiunta.

Richiesta di inserimento di un POI Lo studente, all'interno del portale, deve avere la possibilità di richiedere l'aggiunta di un nuovo POI che poi sarà soggetto alla validazione di un utente amministratore. In questa fase l'attore deve poter specificare:

- Titolo;
- Descrizione;
- Posizione del POI all'interno della mappa.

Come si nota, l'utente di tipo *Studente* può effettuare solamente un numero limitato di operazioni e ciò è dovuto al fatto che il ruolo è accessibile a tutti i fruitori della piattaforma.

4.2.2 Casi d'uso per l'utente amministratore

I casi d'uso relativi all'utente di tipo *Admin* rappresentano le funzionalità principali per cui Corridors CMS è stato concepito e realizzato. Queste azioni consentono di gestire POI, eventi, utenti e insights sulla cronologia delle posizioni degli utenti.

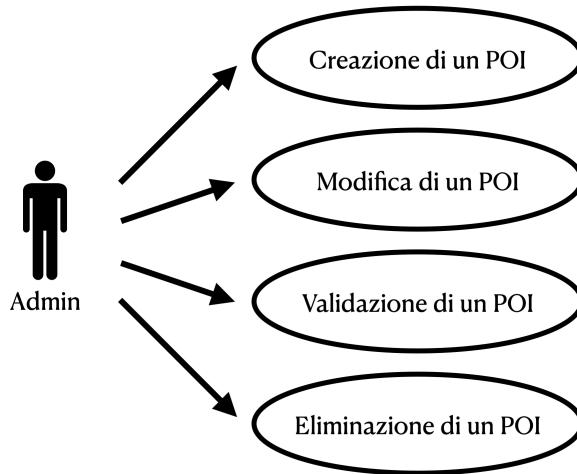


Figura 4.2: Rappresentazione dei casi d'uso dell'amministratore relativi alle CRUD dei POI

POI

I casi d'uso relativi ai POI rappresentati in [Figura 4.2](#) e [Figura 4.3](#) sono i seguenti:

Creazione di un POI L'admin deve poter creare un POI inserendo titolo, descrizione e posizione; il POI creato non richiede ulteriori validazioni.

Modifica di un POI L'utente amministratore deve poter modificare titolo, descrizione e posizione di un POI.

Validazione di un POI L'admin deve poter gestire la validazione di un POI richiesto dagli utenti di tipo *Studente*. Un POI validato dall'amministratore diventa visibile a tutti gli utenti.

Eliminazione di un POI L'utente amministratore deve poter eliminare un POI e tutte le informazioni relative ad esso. Tuttavia, se il POI è collegato a degli eventi, essi devono mantenere le informazioni sul POI anche se è stato eliminato.

Visualizzazione tabellare di tutti i POI L'admin deve poter vedere la lista di tutti i POI all'interno di una tabella, così che le informazioni mostrate risultino più schematiche anche se ben approfondate

Visualizzazione a mappa di tutti i POI L'utente amministratore avrà una visione d'insieme di tutti i POI memorizzati mediante una mappa.

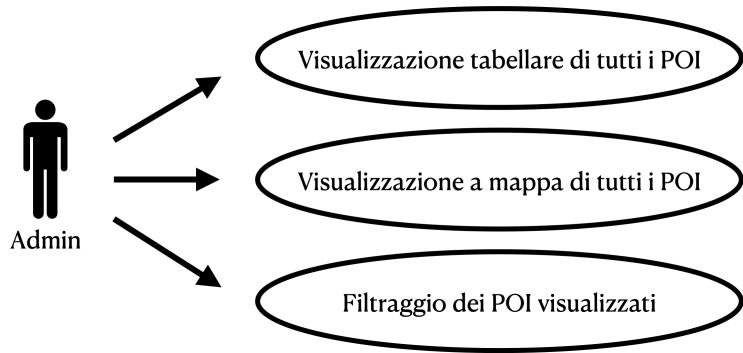


Figura 4.3: Rappresentazione dei casi d'uso dell'amministratore relativi alla visualizzazione dei POI

Filtraggio dei POI visualizzati L'admin deve poter filtrare i POI mediante:

- **Query string:** la stringa inserita viene cercata all'interno di tutti i campi del POI;
- **Stato di validazione:** l'utente sceglie di mostrare solo i POI corrispondenti ad un certo stato di validazione.

Eventi

I casi d'uso relativi agli eventi rappresentati in [Figura 4.4](#) e [Figura 4.5](#) sono i seguenti:

Creazione di un evento L'admin deve poter creare un evento inserendo titolo, descrizione, periodo di svolgimento, e il POI collegato. L'evento creato non richiede ulteriori validazioni.

Modifica di un evento L'utente amministratore deve poter modificare titolo, descrizione, periodo di svolgimento, e il POI collegato.

Validazione di un evento L'admin deve poter gestire la validazione di un evento richiesta dagli utenti di tipo *Studente*. Un evento validato dall'amministratore diventa visibile a tutti gli utenti.

Eliminazione di un evento L'utente amministratore deve poter eliminare un evento e tutte le informazioni relative ad esso.

Visualizzazione tabellare degli eventi L'admin deve poter vedere la lista di tutti gli eventi all'interno di una tabella, così che le informazioni mostrate risultino più schematiche anche se ben approfondite.

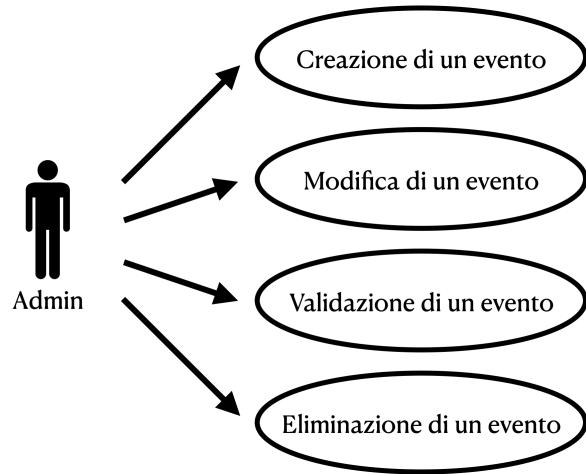


Figura 4.4: Rappresentazione dei casi d'uso dell'amministratore relativi alle CRUD degli eventi



Figura 4.5: Rappresentazione dei casi d'uso dell'amministratore relativi alla visualizzazione degli eventi

Filtraggio degli eventi visualizzati L'admin deve poter filtrare gli eventi mediante:

- **Query string:** la stringa inserita viene cercata all'interno di tutti i campi dell'evento e del POI collegato;
- **Stato di validazione:** l'utente sceglie di mostrare solo gli eventi corrispondenti ad un certo stato di validazione.

Utenti

Visualizzazione di tutti gli utenti registrati L'admin deve avere la possibilità di visualizzare tutti gli utenti registrati e di consultarne le informazioni principali, come mostrato in [Figura 4.6](#).

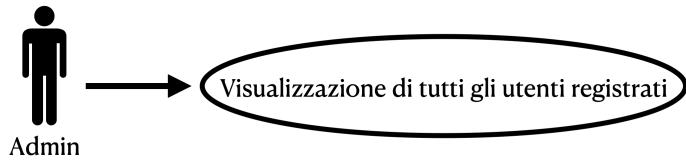


Figura 4.6: Rappresentazione del caso d'uso dell'amministratore relativo agli utenti

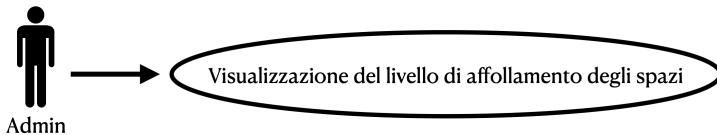


Figura 4.7: Rappresentazione del caso d'uso dell'amministratore relativo all'affollamento

Affollamento

Visualizzazione del livello di affollamento degli spazi L'admin deve poter visualizzare in modo semplice e intuitivo lo stato di affollamento nelle varie zone della mappa, come mostrato in Figura 4.7.

4.3 Flussi di interazione

Un flusso di interazione [22] è definito come una serie di passi che devono essere eseguiti in maniera sequenziale per il completamento di una determinata attività e ha un inizio e una fine ben definiti. I workflow definiti di seguito sono stati rappresentati mediante lo standard open source *Business Process Modeling Notation* (BPMN) [23] che consente di modellare le fasi di un processo aziendale dall'inizio alla fine illustrando visivamente una sequenza dettagliata di attività necessarie per il completamento del processo.

Le sezioni orizzontali rappresentano i diversi contesti in cui l'attività può essere eseguita, mentre i rettangoli rappresentano i singoli step. Ogni step può essere automatico o richiedere l'intervento di un operatore ed è collegato al successivo mediante una freccia. Il collegamento fra due step non è sempre lineare ma può contenere un blocco romboidale utile a gestire situazioni in cui è necessario l'inserimento di una condizione per stabilire quale debba essere il passo successivo.

4.3.1 Registrazione

Quando un nuovo utente vuole iniziare ad utilizzare le funzioni della dashboard di Corridors ha bisogno innanzitutto di registrarsi. In questa fase l'utente dovrà specificare nome, cognome, email e password che verranno poi memorizzati nel database. All'interno di questo flusso, rappresentato in Figura 4.8, oltre

all’utente e al server, è opportuno coinvolgere un servizio che gestisca l’autenticazione. Esso, infatti, permetterebbe di semplificare in maniera sostanziale la gestione dell’accesso degli utenti mettendo a disposizione una serie di API che gestiscono tutte le possibilità che l’implementazione di un’autenticazione dovrebbe controllare (register, login, logout, cambio password, api sicure, ecc.).

Inserimento credenziali

Questa è la prima fase e consiste nell’inserire tutti i dati relativi all’utente all’interno del form di registrazione. Al click del pulsante “Registrati” la richiesta viene presa in carico dal servizio di autenticazione che si occupa di fare una serie di verifiche prima di effettuare il salvataggio preliminare.

Salvataggio preliminare dell’utente

Quando la richiesta di registrazione arriva all’endpoint del servizio di autenticazione, esso effettua innanzitutto due controlli preliminari sui dati passati:

- Controlla se la mail è stata già registrata in una richiesta di registrazione precedente. Se la mail è già memorizzata il flusso torna nella fase di inserimento delle credenziali inviando un avviso all’utente e comunicando che la mail specificata è già in uso;
- Controlla se la password inserita contiene almeno 6 caratteri. Se la password inserita dall’utente non ha abbastanza caratteri, il flusso ritorna nella fase di inserimento delle credenziali mandando un avviso all’utente con la comunicazione che è necessario specificare una password di almeno sei caratteri.

Se i due controlli vengono superati i dati inseriti dall’utente vengono preliminarmente salvati dal servizio di autenticazione. Dal momento che è necessaria l’accettazione della mail di conferma da parte dell’utente, non si è ancora sicuri che la richiesta di registrazione sia andata a buon fine. Dopo la memorizzazione delle informazioni relative all’utente, il servizio di autenticazione si occuperà di inviare una mail di conferma all’utente, che passerà alla fase successiva. In questo momento il server backend non avrà ancora ricevuto alcuna informazione riguardo la richiesta di registrazione avvenuta. Oltre all’invio della mail, verrà richiamata anche un’API backend per il salvataggio delle informazioni dell’utente all’interno del database. In questo step sia il servizio di autenticazione che il database su server avranno il campo relativo alla validazione dell’email a `false`

Convalida registrazione

In questa fase l’utente, visionata la mail ricevuta, dovrà confermare l’avvenuta registrazione. L’accettazione comporta una chiamata al servizio di autenticazione che memorizzerà l’avvenuta verifica della mail.

Bisogna osservare che, al contrario del servizio di autenticazione, il server non ha avuto alcun aggiornamento rispetto alla buona riuscita della registrazione. La gestione della mail di conferma è, infatti, gestita totalmente dal servizio di autenticazione, che non ha modo di chiamare dei servizi lato server. La soluzione a questo problema è legata al fatto che comunque, alla prima operazione di login

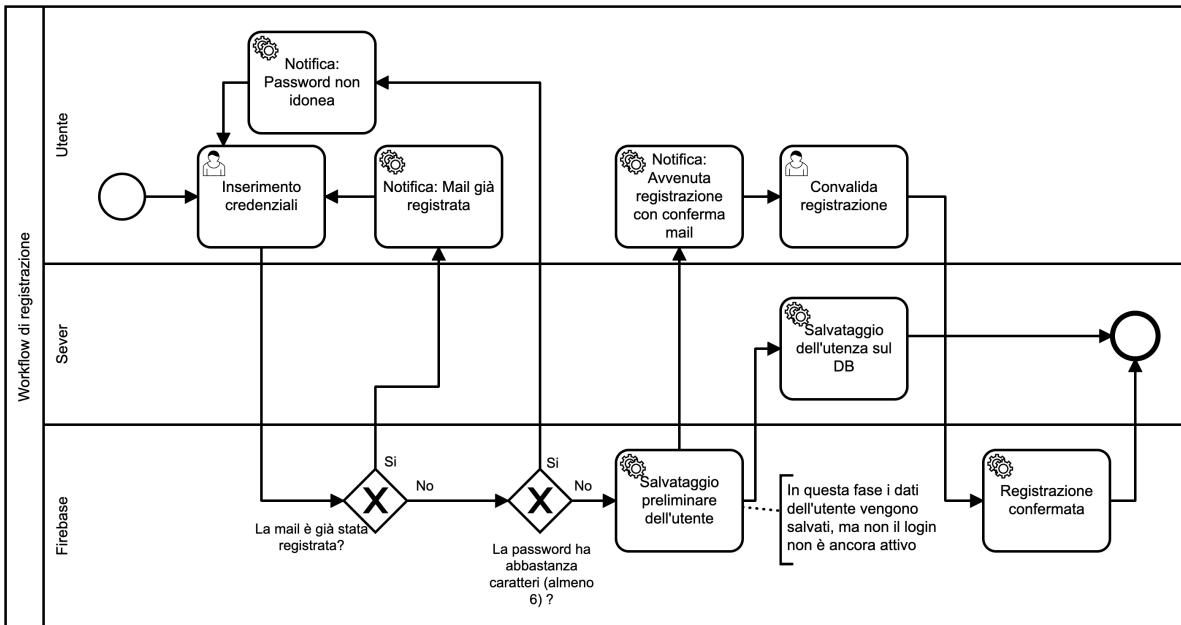


Figura 4.8: Flusso per la registrazione

andata a buon fine, viene comunicato implicitamente che l'indirizzo mail è stato confermato. Il client quindi, al primo login, potrà comunicare al server che anche la validazione dell'email è andata a buon fine..

4.3.2 Login

Quando l'utente ha necessità di accedere alle funzionalità di Corridors CMS deve effettuare l'accesso. Un utente non loggato non può utilizzare alcuna delle funzionalità offerte dal sistema e viene sempre rimandato alla pagina per l'inserimento delle credenziali. Questo flusso, rappresentato in Figura 4.9, serve a proteggere il sistema e a offrire all'utente, in base al suo ruolo, solo le funzionalità a cui può avere accesso.

Inserimento credenziali

Nella pagina di login l'utente dovrà valorizzare i due campi di email e password e in seguito cliccare il pulsante per effettuare l'accesso. Questa sezione ingloba anche un pulsante per il redirect alla pagina di registrazione, nel caso in cui l'utente non abbia mai utilizzato il CMS prima, e un collegamento alla pagina per il recupero delle credenziali.

Verifica delle credenziali

Dopo l'inserimento delle credenziali, esse vengono inviate al servizio di autenticazione affinché controlli se l'email ricevuta sia effettivamente registrata e se la

password inserita coincide.

Credenziali non trovate Se le credenziali non corrispondono ad alcun record memorizzato nel servizio di autenticazione, il workflow torna in fase di inserimento delle credenziali e l’utente viene notificato dell’errore. A questo punto egli deciderà se inserire delle credenziali diverse, fare il recupero delle credenziali oppure registrarsi.

Credenziali trovate Se le credenziali inserite dall’utente sono corrette, il servizio di autenticazione risponde alla richiesta effettuata dal client con il token di accesso (che verrà utilizzato in seguito per la richiesta di API sicure) e con le informazioni sull’utente loggato, fra cui anche il ruolo che permetterà al client di capire quali sezioni effettivamente mostrare.

Aggiornamento utente

L’interazione avvenuta fino ad adesso, come si nota, interessa esclusivamente il client e il servizio di autenticazione: il server non è stato ancora notificato del tentativo di login e dell’avvenuta autenticazione dell’utente. Il client quindi, non appena viene notificato dell’avvenuta autenticazione, chiama un servizio backend passando le informazioni sull’utente. Il server utilizzerà queste informazioni per aggiornare i seguenti campi:

- `last_logged_in`: data e ora dell’ultimo accesso effettuato dall’utente;
- `email_verified`: valore booleano che indica se la verifica della mail è stata effettuata. Al primo login dell’utente il server potrà settare a true questo campo, che non verrà più modificato per tutti i login successivi.

Alla fine del flusso l’utente sarà considerato correttamente loggato e potrà utilizzare tutti i servizi offerti da Corridors CMS. Ciò potrà avvenire solo se l’utente effettuerà le chiamate alle API, specificando anche il token ottenuto in questo frangente. Sarà proprio il token a garantire l’affidabilità dell’utente fornendo sicurezza e robustezza all’intero sistema.

4.3.3 Richiesta API sicura

Quando l’utente accede ad una nuova sezione di Corridors o vuole compiere una determinata operazione che coinvolge anche il backend, egli effettua una richiesta ad una API. Questa richiesta è detta “sicura” perché il server, prima di fornire una risorsa, controlla l’identità dell’host che l’ha effettuata e verifica che l’utente sia autenticato mediante l’utilizzo di un token.

Anche all’interno di questo flusso, rappresentato in Figura 4.10, oltre all’utente e al server viene coinvolto anche un servizio di autenticazione. Esso, infatti, permette di semplificare in maniera sostanziale la gestione dell’autenticazione mettendo a disposizione una serie di API che gestiscono tutte le possibilità che l’implementazione di un’autenticazione dovrebbe controllare (register, login, logout, cambio password, api sicure ecc.).

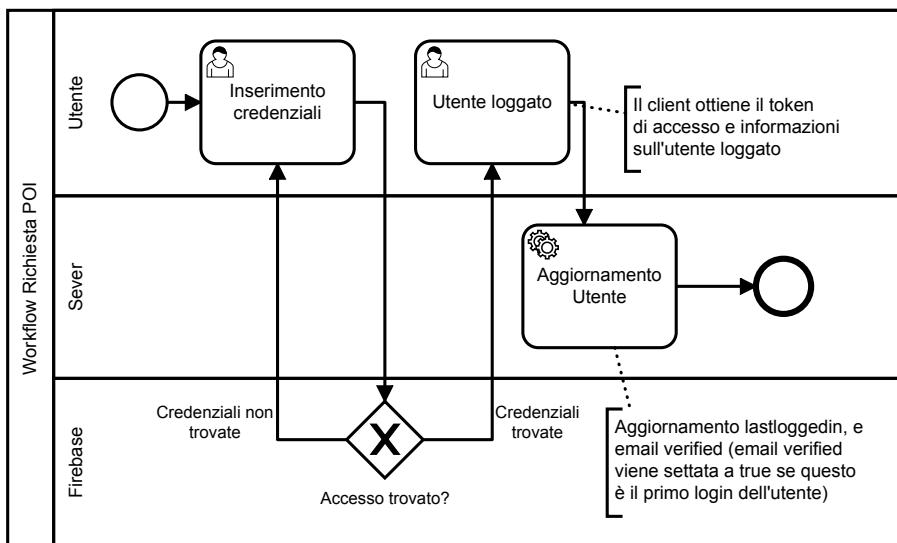


Figura 4.9: Flusso per il login

Richiesta

Questa è la prima fase e consiste nell'invio da parte del portale frontend di una richiesta alle API che vengono esposte lato server. La richiesta in questione può essere di qualsiasi natura: l'aggiunta o la modifica di un POI, la visualizzazione della pagina relativa agli eventi oppure la visualizzazione della heatmap. Nella richiesta deve essere presente l'header che gestisce l'autenticazione: **Authentication**, che deve essere settato al valore del token dell'utente.

Richiesta verifica token

Durante questa fase il server ha ricevuto la richiesta dal client e si prepara a servire la risorsa. Prima di rispondere, il server, deve assicurarsi che l'endpoint che ha generato la richiesta sia autenticato; per fare ciò effettua una richiesta al servizio di autenticazione che verifica se il token inviato dall'utente è valido.

Il token non è valido La risorsa non viene servita e il client viene notificato con una risposta con status code 401, quindi non autorizzato.

Il token è valido Si passa nella fase successiva in cui il server genera la risposta.

Generazione della risposta

In questo momento il server è sicuro che l'utente che ha effettuato la richiesta è autenticato, può quindi elaborarla facendo le dovute operazioni anche sul database e formulare la risposta da inviare al client.

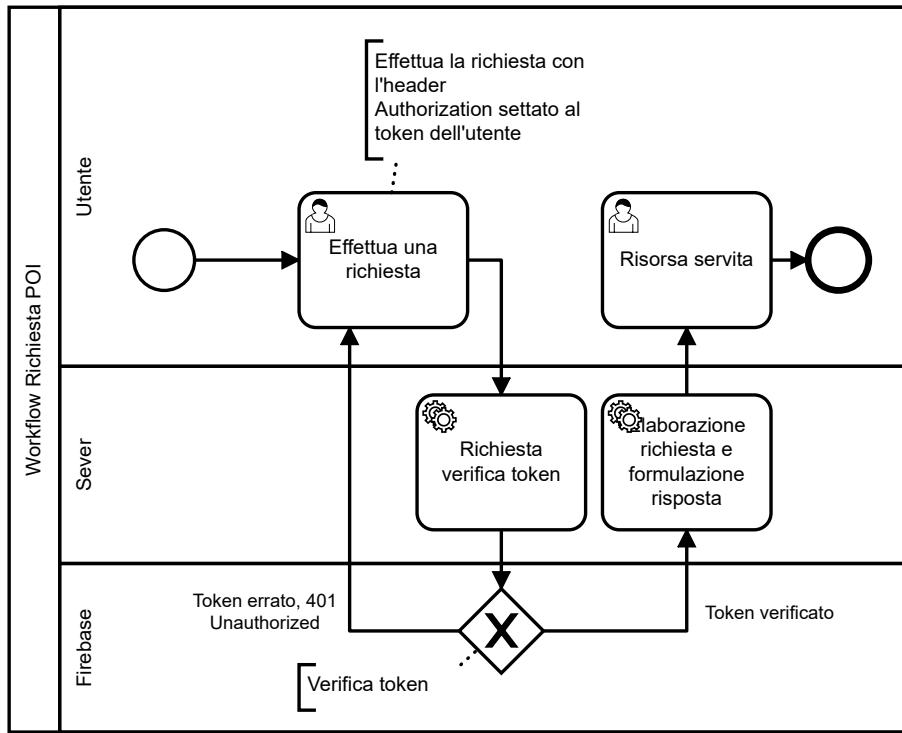


Figura 4.10: Flusso per la richiesta di un'api sicura

Risorsa servita

In questa fase il client riceve la risorsa che aveva precedentemente richiesto e la serve all'utente.

Il flusso appena descritto aiuta a garantire la privacy e la sicurezza dell'applicazione, rendendola inaccessibile a utenti esterni malintenzionati. L'affidabilità è garantita dal meccanismo dei token che viene utilizzato per l'autenticazione degli utenti.

4.3.4 Richiesta di approvazione POI

La richiesta di approvazione di un POI, rappresentata in Figura 4.11, procedura con cui uno studente può fare richiesta di aggiunta di un POI, è strutturata in diverse fasi e coinvolge sia delle operazioni sul web server, sia quelle di un amministratore di sistema.

Richiesta POI

La prima fase è quella di compilazione della richiesta: lo studente richiede l'aggiunta di un POI mediante un click sulla posizione desiderata nella mappa che viene mostrata nella schermata principale. Ciò comporta la comparsa di una modale che permette di specificare un nome e una descrizione. Al click del pul-

sante “Invia” la richiesta passa al server. Il POI in questione sarà visualizzato nella mappa dell’admin e in quella dell’utente richiedente.

Controllo preliminare e salvataggio sul Database

Non appena la richiesta arriva al server si attiva un meccanismo che stabilisce se è idonea o meno. Si definisce idonea una richiesta che rispetta i requisiti minimi di validità in riferimento ai dati inseriti all’interno della modale e la posizione del POI rispetto a quelli già presenti. Il controllo sui dati inseriti nella modale viene fatto dal server assicurandosi che il POI abbia un titolo non vuoto. La verifica relativa al posizionamento viene fatta, sempre dal backend, accertandosi che il POI abbia una distanza da tutti gli altri che sia maggiore di una soglia specificata in base alle condizioni progettuali. Attualmente il parametro stabilito è di due metri, ma non è escluso che in futuro possa essere modificato per soddisfare eventuali necessità.

Richiesta non idonea Se la richiesta non risulta idonea, il flusso torna nella prima fase e l’utente viene notificato dell’accaduto.

Richiesta idonea Se invece la richiesta è ritenuta idonea (in quanto tutti i requisiti di base sono stati rispettati), viene salvata all’interno del database. Lo studente viene notificato dell’avvenuta presa in carico da parte del sistema. L’amministratore, invece, viene informato dell’arrivo di una nuova richiesta di approvazione.

Validazione POI

In questa fase l’utente amministratore controlla i dati inseriti dallo studente e ne valuta l’approvazione ufficiale. Nello specifico, i vincoli formali che la richiesta deve soddisfare sono:

- La posizione del POI è opportuna e coerente con il titolo e la descrizione inseriti;
- Il nome e la descrizione devono essere evidentemente non ambigue e diverse da quelle di altri POI, in modo da favorirne l’indicizzazione e la ricerca;
- Il nome e la descrizione devono essere ben formati e grammaticalmente corretti;
- Il POI deve essere posizionato in corrispondenza dell’ingresso del luogo che si vuole contrassegnare e non al suo interno. Questo vincolo fa in modo che l’individuazione del percorso verso il POI sia il più corretta e intuitiva possibile.

A seguito di queste verifiche l’amministratore può decidere di validare direttamente la richiesta, di apporre delle modifiche o di rigettarla.

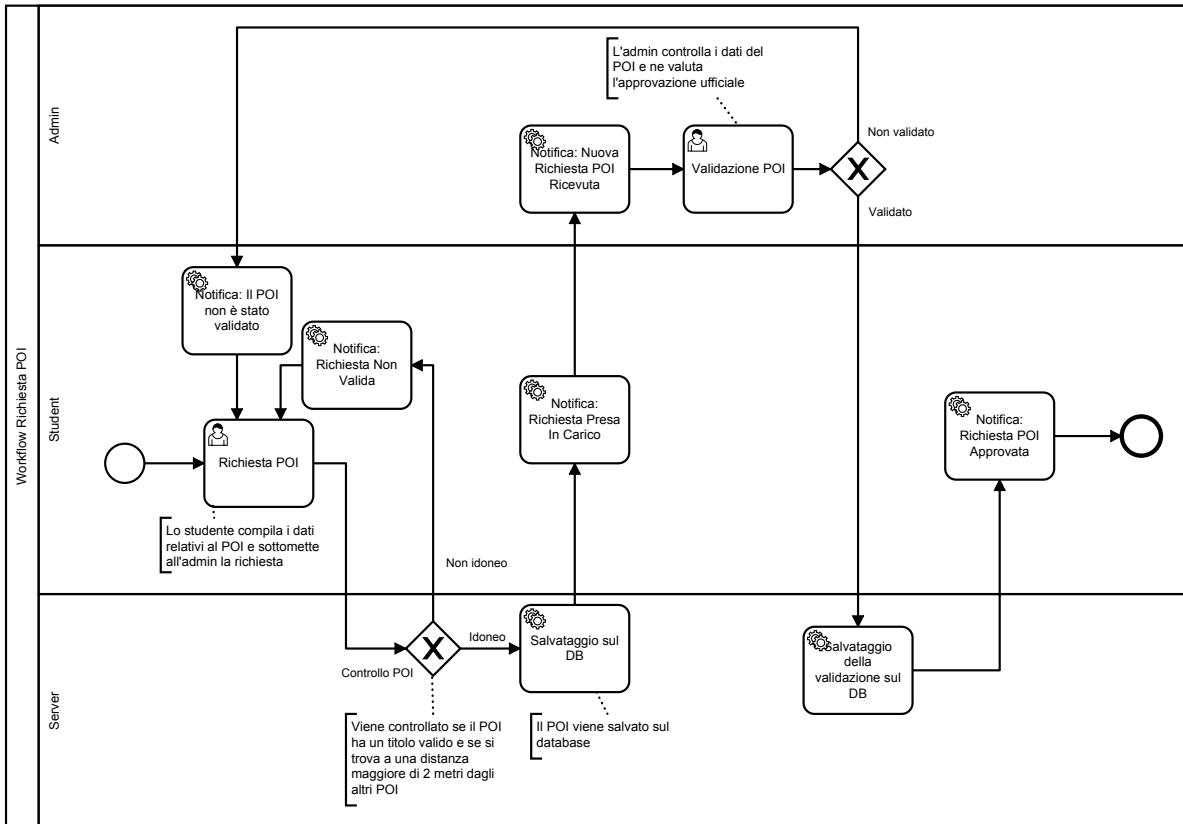


Figura 4.11: Flusso per la richiesta di approvazione di un POI

Approvazione diretta In questo caso l'amministratore ritiene che tutti i vincoli sono rispettati e decide di validare la richiesta senza fare alcuna modifica selezionando l'apposita riga nella dashboard e premendo il tasto “Valida”. Il server inserirà nel database la data di validazione corrispondente alla richiesta e lo studente richiedente riceverà una notifica informativa. Da questo momento il POI sarà visibile da tutti gli studenti.

Approvazione con modifica In questo caso l'amministratore può decidere di apportare delle correzioni alla richiesta dell'utente. Nello specifico, attraverso Corridors CMS, l'admin può visualizzare le informazioni della richiesta cliccando sull'icona corrispondente. Il click sul bottone mostra una modale con la possibilità di modificare i campi del nome e della descrizione e di modificare il posizionamento del POI all'interno della mappa. Dopo l'aggiornamento dei campi, l'utente può procedere alla validazione del POI.

Negazione Quando l'amministratore ritiene che la richiesta non rispetti i vincoli, può decidere di rigettarla. Ciò comporta il ritorno alla prima fase, quella di richiesta POI, e lo studente riceve una notifica che lo informa del rifiuto.

Il flusso appena descritto ha una duplice funzione: alleggerire gli amministratori del carico di lavoro, che altrimenti comprenderebbe l'inserimento di un POI per volta, e consentire allo studente di percepirci parte integrante della realtà del Politecnico in quanto può partecipare attivamente al miglioramento delle funzionalità di Corridors.

Capitolo 5

Progettazione di Corridors CMS

5.1 Progettazione del logo

Il logo di Corridors è stato progettato e realizzato con lo scopo di mostrare sinteticamente ed efficacemente la funzionalità principale dell'applicazione e, a contempo, si è cercato di renderlo originale e distinguibile. Durante la fase di design sono state create più versioni del logo, rappresentate in [Figura 5.1](#), l'una distinta dall'altra per gli elementi presenti all'interno, le dimensioni e i colori. Alla fine la scelta è ricaduta su quella che è stata ritenuta più rappresentativa, che è stata successivamente perfezionata fino a raggiungere l'aspetto attuale.

Nel logo definitivo, mostrato in [Figura 5.2](#), è presente una strada, che rappresenta il sistema di corridoi, strade e snodi del Politecnico di Bari, su cui è posizionato un indicatore tondo con una scia conica che rimanda alla funzionalità di positioning offerta dall'applicazione. In alto a destra è presente il logo del Politecnico di Bari e in basso il logo Nextome, entrambi componenti fondamentali della vision che sta alla base dell'idea e dello sviluppo di Corridors.

In un primo momento la scelta era ricaduta sull'utilizzo di colori chiari, con l'idea di riprendere quelli del logo del Politecnico. Successivamente, per avere un contrasto maggiore e un aspetto più accattivante, si è deciso di utilizzare un colore più scuro per lo sfondo. Il logo rappresenta la base e il punto di partenza per la definizione dei colori e dell'aspetto grafico di Corridors CMS.

5.2 Progettazione dell'interfaccia utente

L'interfaccia utente è una delle parti fondamentali di una web application e consente l'interazione del server con l'esterno. Una progettazione grafica efficace permette di presentare, in modo completo e intuitivo, il contenuto informativo all'utente, consentendogli una buona fruizione di tutte le funzionalità che la web app offre. Un'interfaccia mal progettata limita l'utente perché non gli permette di sfruttare al massimo le potenzialità che il sistema offre. Lo scopo della progettazione della UI è quello di presentare le informazioni in modo chiaro e senza ambiguità, così da non far confondere l'operatore durante l'utilizzo. La

Corridors

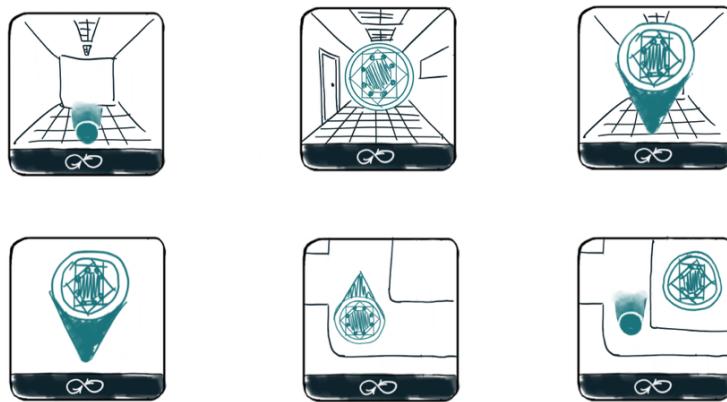


Figura 5.1: Versioni del logo

UI deve essere progettata in modo tale da mantenere una certa consistenza grafica fra le pagine e fra gli elementi della pagina utilizzando colori e stili simili.

Corridors CMS è stato realizzato in modo da garantire un facile utilizzo e una grafica che segua i canoni di Material Design [24] per la realizzazione di grafiche consistenti, conformi anche alle applicazioni mobile. Per garantire l'accessibilità da qualsiasi dispositivo la struttura della pagina e gli elementi che la compongono cambiano responsivamente in base al terminale da cui viene visualizzata. Nello specifico, la visuale da smartphone si differenzia da quella desktop per alcune caratteristiche:

- La sidebar è nascosta e visualizzabile cliccando un apposito pulsante;
- La topbar presenta delle icone piuttosto che degli elementi testuali;
- Vengono nascosti i filtri per le visualizzazioni tabellari;
- Le modali coprono l'intera pagina verticalmente;
- Sono stati inseriti dei bottom sheet in alternativa ai popup sulla mappa.

Di seguito sono descritti gli elementi e le sezioni principali del CMS e come essi sono connessi fra di loro.



Figura 5.2: Logo finale



Figura 5.3: Topbar

5.2.1 Topbar

La topbar, di cui ne è rappresentata l'implementazione in Figura 5.3, è un elemento del layout posizionato nella parte superiore della pagina, essa rimane consistente nelle varie pagine del portale e non cambia al variare della schermata visualizzata. Questo componente contiene diversi elementi:

- Informazioni sull'utente loggato;
- Possibilità di effettuare il logout dell'utenza attualmente autenticata;
- Il logo di Corridors che al click riporta alla pagina principale del portale, quella che permette la visualizzazione della mappa con i POI;
- Un burger button per l'espansione della sidebar.

5.2.2 Sidebar

La sidebar, implementata come in Figura 5.4, è un elemento del layout posizionato nella parte sinistra della pagina e può essere nascosta e mostrata all'occorrenza.

Presenta un menu laterale composto da una lista di collegamenti a tutte le pagine che sono cliccabili e visitabili dall'utente correttamente loggato. Ogni elemento della lista contiene un'icona rappresentativa e un titolo. Facendo click sul burger button presente nella topbar è possibile espandere e contrarre il menu mostrando o nascondendo il titolo dei vari elementi. Le funzionalità principali, come la visualizzazione dei POI e quella degli eventi, sono in cima alla lista per rendere più accessibili le sezioni che saranno visitate più frequentemente dagli utenti.

Mentre nella visualizzazione desktop la sidebar viene mostrata di default, nella visualizzazione mobile è normalmente nascosta, ma è possibile farla comparire a tutta pagina sempre mediante il burger button a sinistra della barra in alto.

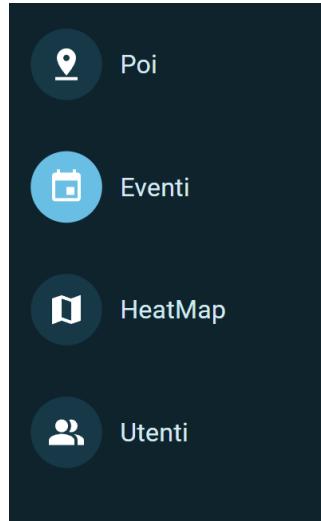


Figura 5.4: Sidebar

5.2.3 Struttura delle pagine

Dopo la progettazione degli elementi statici del layout, sono state identificate le pagine necessarie e i componenti che ne devono formare il contenuto principale. Si è scelto di dedicare una pagina per ogni servizio offerto dalla web app:

Lato admin:

- Gestire POI;
- Gestire eventi;
- Visualizzare le aree più affollate;
- Gestire gli utenti.

Lato utente:

- Richiesta di creazione di un POI;
- Visualizzazione dei POI già creati e approvati.

5.2.4 POI

Questa pagina, accessibile solamente agli admin, deve consentire la gestione dei POI e la validazione delle richieste di creazione di POI da parte dell'utente. È possibile visualizzare le informazioni e mettere in atto le azioni sui POI mediante due modalità diverse, ci sono quindi due tipi di visualizzazioni diverse:

Visualizzazione tabellare

Questa visualizzazione è utile se non si conosce precisamente la posizione del POI, ma ne è noto solo il nome, in questo caso la ricerca nella tabella sarebbe facilitata. Può anche tornare utile in fase di validazione delle richieste di

creazione fatte dagli utenti, perché permette di visualizzare tutte le richiesta in maniera più intuitiva e con una maggiore visione d'insieme. Con questa modalità di visualizzazione la pagina cambia completamente la struttura in base al dispositivo da cui la si visualizza.

Dispositivi desktop Nel layout su desktop, implementato come in [Figura 5.5a](#), l'elemento principale è la tabella che è posizionata al centro della pagina che si espande per tutta la sua lunghezza. La tabella mostrata contiene le informazioni relative ai POI memorizzati nel database e in particolare abbiamo:

- Checkbox di selezione;
- Titolo;
- Descrizione;
- Data di validazione;
- Pulsanti per la modifica o la rimozione dell'elemento.

In alto a destra, sopra la tabella, sono stati posizionati dei filtri utili a differenziare i POI mostrati in base allo stato di validazione. In basso, sono stati posizionati i pulsanti per la validazione o invalidazione multipla dei POI. In alto alla pagina, è stato posizionato il pulsante per la creazione di un nuovo elemento.

Al click del pulsante per la creazione o la modifica di un POI viene aperta una modale, realizzata in fase di implementazione come in [Figura 5.5c](#), che è sovrapposta al resto del contenuto della pagina. Essa offre la possibilità di valorizzare i dati e la posizione di un POI per la prima volta o di modificarli in seguito. La modale contiene:

- Un titolo che descrive se si sta effettuando una modifica o una creazione;
- Un corpo con il form per l'inserimento dei dati, eventualmente già compilato;
- La mappa in cui è possibile selezionare la posizione del POI;
- Un pulsante per il salvataggio.

Dispositivi mobile Nel layout su mobile implementato come in [\(Figura 5.5d\)](#), si è reso necessario collassare la tabella in una serie di elementi posizionati verticalmente uno sotto l'altro. Questi elementi contengono le informazioni necessarie del POI e inglobano anche i pulsanti per la modifica e l'eliminazione. In questa visualizzazione i filtri della lista posizionati in alto a sinistra non contengono più tre pulsanti ma solamente uno indicante quale delle tre casistiche è correntemente applicata; se viene cliccato, mostra un menù in cui è possibile selezionare un filtro da applicare. Su dispositivi piccoli con schermo verticale la modale tende a riempire l'intera altezza della pagina.

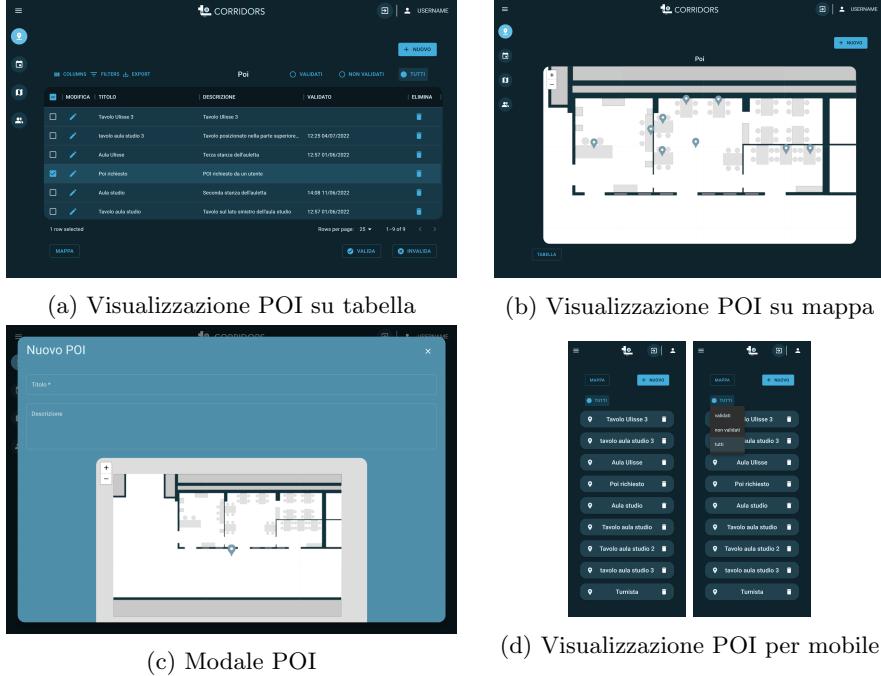


Figura 5.5: Schermate del CMS relative ai POI

Visualizzazione mediante mappa

Questa visualizzazione, rappresentata in Figura 5.5b, fornisce le informazioni relative alla posizione dei POI in maniera più intuitiva, rendendo più facile l'individuazione di aree non mappate o POI posizionati troppo vicini fra di loro. Utilizzando questo layout, il comportamento degli elementi sui dispositivi desktop è molto simile a quello sui dispositivi mobile: la mappa è centrale e occupa gran parte della pagina, in essa sono rappresentati i POI sotto forma di marker. Posizionando il mouse in corrispondenza di un marker, viene mostrato un piccolo tooltip contenente il titolo del POI; cliccandolo o toccandolo si apre la modale di modifica.

5.2.5 Eventi

Questa pagina, accessibile solo agli admin, deve permettere la gestione degli eventi e la validazione delle richieste di creazione degli stessi da parte degli studenti.

Gli eventi forniscono informazioni sulle attività correntemente svolte nei pressi del POI collegato, così da poterne mostrare le informazioni quando un utente viene localizzato nelle vicinanze mediante l'app Corridors. In particolare, oltre alle informazioni su titolo e descrizione dell'evento, c'è bisogno che vengano specificate le date di inizio e fine e il periodo in cui l'evento avrà luogo.

Questa sezione si serve di un sistema di validazione molto simile a quello utilizzato per i POI che, anche in questo caso, serve a distinguere gli eventi creati da admin e le richieste di creazione di eventi provenienti dagli studenti.

All'interno di questa pagina, implementata come in [Figura 5.6a](#), è presente solo la visualizzazione tabellare: questa scelta è scaturita dal fatto che un evento è intrinsecamente collegato ad un POI e, al contempo, è possibile collegare ad esso più di un evento. Ciò comporterebbe la sovrapposizione di più marker sulla mappa che non permetterebbe di distinguere eventi diversi correlati allo stesso POI. La gestione della responsività degli elementi sui diversi dispositivi è identica a quella della pagina precedentemente trattata e ne è rappresentata l'implementazione in [Figura 5.6c](#). In particolare nella tabella sono presenti i dati relativi agli eventi:

- Titolo;
- Descrizione;
- Date di inizio e fine;
- POI associato.

I filtri per la visualizzazione degli eventi validati sono posizionati in alto a destra in layout desktop e in alto a sinistra, sotto forma di menu, in layout mobile. Come accade nei POI è possibile anche effettuare una selezione multipla degli eventi con conseguente validazione.

Anche in questo caso il click dei pulsanti di aggiunta o modifica dell'evento comportano la comparsa di una modale che permette l'inserimento e la modifica delle informazioni relative all'evento. In particolare, la selezione del POI è realizzata con una barra di ricerca, sotto la quale, di volta in volta, compaiono i POI che soddisfano il criterio di ricerca inserito; sarà infine possibile selezionare quello corretto.

5.2.6 Heatmap

Lo scopo della pagina dedicata alla heatmap, di cui ne è stata rappresentata l'implementazione in [Figura 5.7](#), è quello di segnalare graficamente le aree più affollate, mostrando in modo generico e anonimo le posizioni registrate da tutti gli utenti che utilizzano Corridors. Mentre si utilizza l'app mobile, essa memorizza periodicamente le posizioni indoor degli utenti calcolate dall'SDK e fornisce successivamente all'admin un report più o meno chiaro dell'affollamento nelle diverse zone del campus in diverse finestre temporali.

Nello specifico, la pagina è composta interamente da una mappa indoor che colora le aree in modo differente in base al livello di affollamento: colori freddi come blu e celeste sono usati per segnalare aree meno affollate, mentre colori caldi come giallo, arancione e rosso ne indicano una maggiore affluenza.

5.2.7 Utenti

Questa sezione, visualizzabile solo lato admin, permette la consultazione degli utenti registrati. Il layout della pagina, implementato come in [Figura 5.8](#), è costituito da una tabella che raccoglie diverse informazioni sugli utenti:

- UID: che rappresenta l'identificativo fornito dal servizio di autenticazione;
- Cognome;

(a) Visualizzazione eventi su tabella

(b) Modale degli eventi



(c) Visualizzazione eventi per dispositivi mobile

Figura 5.6: Schermate del CMS relative agli eventi

- Nome;
- Email;
- Telefono.

In questo caso la tabella non necessita di particolari filtri e interazioni con l'utente ma rimane coerente al resto del CMS mantendo le stile delle tabelle presenti nelle altre sezioni.

5.2.8 Richiesta POI

Questa pagina, rappresentata in Figura 5.9a, è visualizzabile solamente dagli studenti e offre la possibilità di inviare una richiesta di creazione di POI visualizzando, al contempo, quelli già validati. In questo caso la scelta è ricaduta su un layout a mappa perché è più intuitivo e meno macchinoso per un utente che non utilizza il portale assiduamente.

L'elemento principale è quindi la mappa che, anche in questo caso, occupa la dimensione dell'intera pagina in quanto lo studente non necessita di funzionalità o pulsanti aggiuntivi. Quando l'utente clicca su una zona della mappa, crea un

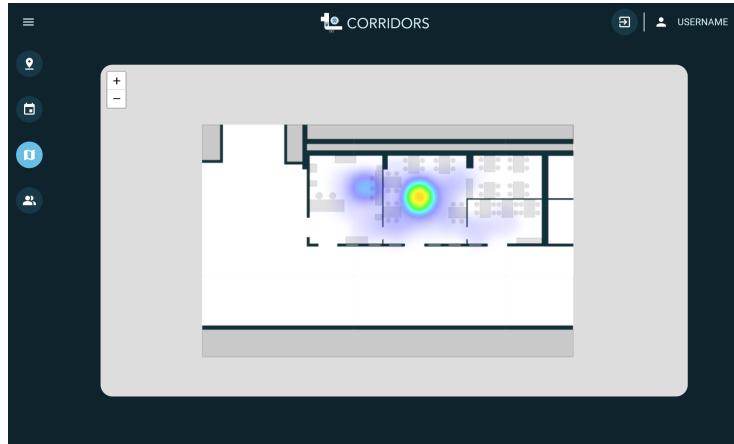


Figura 5.7: Visualizzazione posizioni su Heatmap

UID	NOME	COGNOME	USERNAME	EMAIL	NUMERO DI TELEFO...
ns4YZ9SOFZquWF3yLbwRwfJ1z72			p.masciullo@studenti...		
WaFkEnr20EwWmcmImxvXCY8ic...			admin@admin.com		
sflJcNn2PXf1xebsQyljjsCVOC3			d.marchitelli1@etude...		
j5NeNQ7rlH7dsjajqjdNNW7kTjy2			derio002@hotmail.it		
mifDn36XCGBtUJ4T4YtkJwFF7v1			vitodbc@gmail.com		
Y3C7Wiev2QevA47BEdnIM0EPsY73			laera.sante@gmail.co...		

Figura 5.8: Visualizzazione utenti su tabella

marker con una piccolo popup collegato. Questo elemento contiene un piccolo form, implementato come in Figura 5.9b, in cui è possibile inserire il titolo e la descrizione del POI che si vuole richiedere. Nella visualizzazione mobile in Figura 5.9c il form diventa un bottomsheet.

5.3 Progettazione di sistema

5.3.1 Architettura

L'architettura del sistema, rappresentata in Figura 5.10, è stata progettata sulla base di due principi:

- **Single Responsibility:** ogni componente del sistema incapsula una sola responsabilità, in questo modo i cambiamenti su un componente del sistema, non impattano sulle funzionalità di altri componenti;



(a) Mappa dei POI per studenti

(b) Popup per richiesta nuovo POI



(c) Bottomsheet di richiesta nuovo POI per dispositivi mobile

Figura 5.9: Schermate del CMS relative alla visualizzazione lato *Studente*

- **Separation Of Concerns:** ogni componente deve encapsulare una particolare preoccupazione dell'intera applicazione in maniera modulare, ossia esponendo all'esterno delle interfacce che nascondano i dettagli implementativi.

Il CMS Corridors è un sistema complesso che segue il modello di architettura multi-tier. In questo tipo di architettura è possibile riconoscere una serie di componenti logico-fisiche, chiamate “tier”, che rappresentano i vari livelli dell'applicazione, come ad esempio: browser, web server, application server, database server e così via. Inoltre, una caratteristica dei tier è quella che ognuno di essi può comportarsi da client e/o server nei confronti dei tier adiacenti [25] per richiedere/fornire una o più risorse.

Di seguito verranno descritti i vari tier che compongono l'architettura del sistema.

Corridors App

È il componente attraverso il quale l'utente può accedere ai servizi e le funzionalità offerte da Corridors, offre il servizio di localizzazione e di navigazione.

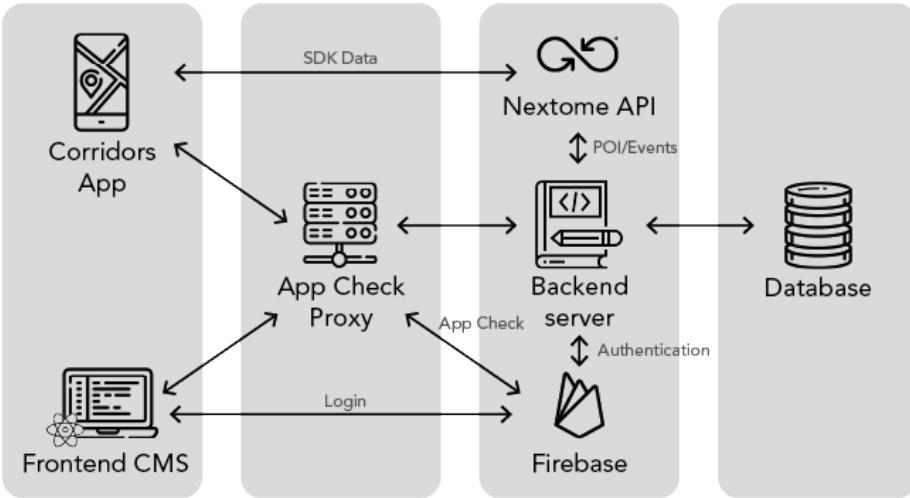


Figura 5.10: Architettura multi-tier del sistema Corridors

Frontend CMS

È il componente realizzato mediante React, un framework Javascript che si occupa di mostrare l'interfaccia utente del CMS consentendo la visualizzazione dei contenuti e l'interazione con l'utente. Questo componente deve interfacciarsi con i servizi offerti dal backend mediante delle REST API protette da autenticazione.

Backend

È il componente che realizza la logica applicativa di Corridors esponendo le API che permettono ai componenti client di accedere alle risorse del sistema. Il backend si occupa di interagire con tutti i componenti del sistema, descritti in seguito, che gestiscono le risorse da servire ai client. Il componente è stato realizzato mediante SpringBoot, framework Java, molto robusto e ideale in contesti strutturati e complessi come Corridors CMS.

Database

Ospita l'RDBMS¹ PostgreSQL che contiene il database relazionale di Corridors su cui sono immagazzinate la maggior parte delle risorse del sistema (utenti, POI, eventi e log delle posizioni). Lo schema del database verrà schematizzato e spiegato in seguito.

Firebase

È una piattaforma Google che offre servizi utili allo sviluppo di un'applicazione web. Corridors ne utilizza 4 in particolare:

- **Firebase Authentication** [3]: gestisce l'autenticazione esponendo delle API utilizzabili per i form di registrazione, login e password dimenticata;

¹RDBMS: Relational DataBase Management System

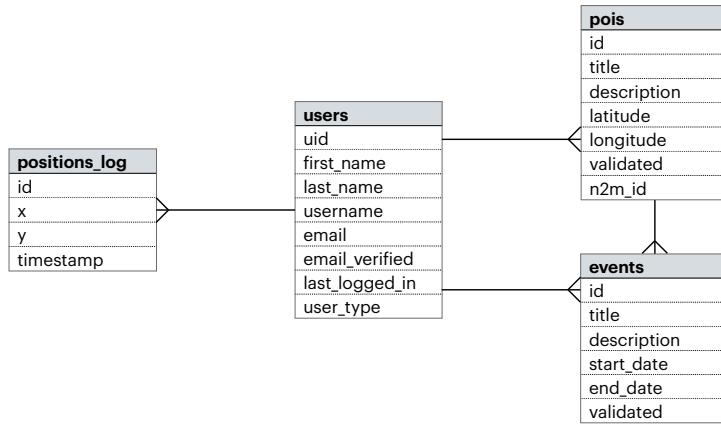


Figura 5.11: Schema del Database

- **Firebase App Check** [26]: aggiunge un ulteriore livello di protezione delle API rispetto a quello offerto dalla Spring Boot Security, soprattutto per quanto riguarda le API richiamate dall'app Corridors, sui client mobile;
- **Firestore Database**: usato per archiviare le immagini relative ai POI;
- **Firebase Hosting**: utilizzato per l'hosting e il deployment del frontend.

App Check Proxy

Il servizio Firebase App Check citato precedentemente è disponibile solo per applicazioni web Node.js, ma Corridors è stata sviluppata con Spring Boot, motivo per cui è stato sviluppato un ulteriore tier con Node.js. Questo tier si interpone tra i client e il server backend fungendo da proxy. Si occupa di verificare che le app Corridors che effettuano le richieste al backend siano state installate in maniera genuina dagli store ufficiali e quindi non manomesse sfruttando proprio il servizio Firebase App Check.

Nextome API

Per permettere la navigazione verso i POI creati e gestiti dal CMS Corridors, bisogna consentire all'SDK Nextome di poterli riconoscere all'interno del proprio database; è necessario quindi allineare i due database.

Questo componente rappresenta le RESTful API third-party che Nextome ha messo a disposizione per effettuare le azioni CRUD su POI ed Eventi direttamente sul proprio database. Essendo tali API protette, il sistema Corridors può accedervi solamente mediante un token fornito da Nextome stessa.

5.3.2 Database

Come mostrato in Figura 5.11 il database di Corridors è formato da 4 tabelle:

Tabella users

Questa tabella descrive gli utenti registrati a Corridors CMS:

- **uid**: identificativo unico per gli utenti;
- **first_name - last_name**: nome e cognome dell'utente;
- **username**;
- **email**;
- **email_verified**: è un booleano che specifica se è stata effettuata la verifica via email dell'account;
- **last_logged_in**: è un timestamp che specifica quando è avvenuto l'ultimo login;
- **user_type**: rappresenta il ruolo dell'utente che può essere di tipo *Studente* o *Admin*.

Tabella pois

Questa è la tabella relativa ai POI e ne contiene le informazioni principali:

- **id**: contiene un identificativo univoco relativo al POI;
- **title**: titolo;
- **description**: descrizione;
- **latitude - longitude**: utili per la memorizzazione della posizione del POI all'interno della mappa;
- **validated**: contiene il timestamp di validazione, se è nullo il POI non è stato ancora validato;
- **n2m_id**: rappresenta l'identificativo del POI all'interno del database interno Nextome.

Questa tabella presenta anche un collegamento con **users** relativamente all'utente che ha creato o richiesto il POI. Questo collegamento è realizzato mediante una uno-a-molti considerando che un utente può creare/richiedere più POI.

Tabella events

Questa tabella contiene le informazioni relative agli eventi:

- **id**: contiene un identificativo univoco relativo all'evento;
- **title**: titolo;
- **description**: descrizione;
- **start_date - end_date**: rappresentano il periodo temporale di svolgimento dell'evento;

- **validated**: contiene il timestamp di validazione, se è nullo il POI non è stato ancora validato.

Questa tabella presenta un collegamento con **users** relativo all'utente che ha creato o richiesto l'evento, questo collegamento è realizzato mediante una uno-a-molti considerando che un utente può creare/richiedere più eventi. Gli eventi necessitano anche di un collegamento uno-a-molti con la tabella dei POI in quanto ad un POI possono essere collegati più eventi.

Tabella positions

Questa tabella contiene la cronologia delle posizioni che verranno utilizzate per la creazione della heatmap:

- **id**: contiene un identificativo univoco della riga;
- **x - y**: rappresentano la posizione dell'utente;
- **timestamp**: contiene un riferimento temporale relativo a quando la posizione è stata memorizzata.

La tabella presenta un collegamento uno-a-molti con **users** relativo all'utente che ha registrato una determinata posizione.

La progettazione di Corridors CMS ha permesso di stabilire, in maniera particolareggiata, come le funzionalità ideate e approfondite nel capitolo precedente mediante la definizione dei requisiti, la rappresentazione dei diagrammi dei casi d'uso e la descrizione dei diversi flussi di interazione con l'utente debbano essere implementate descrivendo dettagliatamente sia gli aspetti grafici che quelli architetturali.

Capitolo 6

Pianificazione dello sviluppo

Durante la pianificazione dello sviluppo si è reso necessario individuare quale sarebbe stato il modello più corretto da adottare per il raggiungimento degli obiettivi attesi.

Essendo Corridors CMS un progetto relativamente piccolo che richiede delle funzionalità abbastanza limitate, la soluzione più semplice sarebbe stata quella di implementare un modello tradizionale come quello a cascata. Il team di sviluppo, vista la natura del progetto, si è subito reso conto che non sarebbe stato possibile avere l'intera lista dei requisiti prima dell'inizio delle fasi di design e sviluppo: di settimana in settimana si sarebbero aggiunti nuovi requisiti e ciò, con un modello a cascata, avrebbe portato ad una rischedulazione continua dei task che compongono il progetto con una conseguente notevole perdita di tempo e risorse.

Si è quindi optato per la metodologia RAD¹, che ha consentito lo sviluppo di Corridors CMS seguendo diverse iterazioni basate sulle differenti funzionalità che sono state man mano pensate e implementate.

6.1 Joint Application Development

Per lo sviluppo di Corridors CMS è stato utilizzato *Joint Application Development* che rappresenta una variante di RAD. JAD estende la metodologia appena descritta includendo il cliente, rappresentato dal relatore, nelle fasi di design e di sviluppo del sistema. Questo metodo standardizza l'inclusione del cliente all'interno delle fasi cruciali del progetto mediante le *JAD sessions* che rappresentano degli incontri/workshop periodici.

Le sessioni JAD sono state organizzate solitamente su base settimanale e hanno rappresentato la fine di un'iterazione e l'inizio di una nuova. Le singole iterazioni hanno avuto una durata di massimo 3 settimane.

¹Rapid Application Development

6.1.1 Iterazioni di Corridors CMS

Di seguito verranno descritte le iterazioni che si sono susseguite durante lo sviluppo di Corridors CMS:

1. Durante la prima iterazione ci si è posti l'obiettivo di ottenere un prototipo molto basilare che comprendesse un layout per le pagine che si sarebbero sviluppate in futuro. Nello specifico è stato fatto il setup del progetto e sono stati creati i componenti relativi a topbar, sidebar e altri che definiscono i diversi layout degli elementi base delle pagine web, come pulsanti, tavole e select.
2. La seconda iterazione vede la creazione delle pagine (lato admin) relative a POI ed eventi. Sono stati anche implementati i relativi casi d'uso di creazione, modifica, visualizzazione e eliminazione. Alla fine di questa iterazione quindi era possibile consultare la lista di POI ed eventi in modalità tabellare e effettuarne le operazioni CRUD.
3. Durante la terza iterazione è stato implementato il meccanismo di validazione di POI ed eventi lato admin ed è avvenuta l'implementazione del componente mappa generico che sarebbe poi stato utilizzato nelle successive iterazioni per la visualizzazione dei POI e la creazione della heatmap.
4. Nella quarta iterazione è avvenuta l'implementazione della mappa nella sezione relativa ai POI per la visualizzazione a mappa e la selezione della posizione del POI all'interno della modale.
È stata inoltre implementata la gestione dell'autenticazione che comprende:
 - Visualizzazione tabellare degli utenti registrati lato admin;
 - Pagine di login e registrazione;
 - Implementazione dei meccanismi di sicurezza delle API mediante token;
 - Gestione dei differenti ruoli dell'utente: *Studente* e *Admin*.
5. Durante la quinta e ultima iterazione sono state implementate diverse funzionalità:
 - Sezione per la visualizzazione a mappa dei POI lato utente;
 - Richiesta di inserimento di un nuovo POI mediante click sulla mappa;
 - Query spaziale sui POI richiesti che verifica che il POI sia sufficientemente distante da tutti gli altri già validati;
 - Heatmap che rappresenta il livello di affollamento sulla mappa attraverso diverse colorazioni.

6.1.2 Suddivisione del lavoro in task

Per portare avanti celermente lo sviluppo era necessario che nessun membro del team si trovasse nella situazione di essere fermo in attesa della fine dello sviluppo di un altro. A tal proposito si è scelto di suddividere il lavoro in task

completamente parallelizzabili e non sequenziali dando a ognuno la possibilità di dedicarsi al progetto in maniera svincolata rispetto agli altri componenti del gruppo.

Come è possibile vedere dalle iterazioni messe in atto, ognuna di esse non si è concentrata su una singola feature ben strutturata, ma su diverse funzionalità piccole e slegate tra di loro, ciò ha consentito la parallelizzazione del lavoro all'interno della singola iterazione.

6.1.3 Delivery continuo

La metodologia utilizzata si è rivelata utile anche rispetto al contesto nel quale questo progetto è stato portato avanti, quello universitario. Questa estrema divisione delle mansioni, infatti, ha la potenzialità di offrire continuamente delivery di nuove funzionalità e ciò ha consentito, settimana per settimana, di testare e far testare le nuove versioni percependo e implementando le valutazioni dei tester quasi in real-time. Ciò ha portato ad una veloce progressione delle funzionalità implementate all'interno del CMS senza bisogno di ulteriori test, perché già eseguiti alla fine della singola iterazione.

6.1.4 Gold plating

Il gold plating rappresenta l'aggiunta, da parte degli sviluppatori, di nuove funzionalità non richieste dal cliente e non specificate in fase di progettazione. Questo può accadere quando i membri del team vogliono offrire al cliente qualcosa di aggiuntivo sperando che ciò lo renda più soddisfatto e felice del progetto nel complesso. Tuttavia esso è fortemente sconsigliato perché porta ad un insensato aumento dei costi oppure ad una riallocazione delle risorse.

Anche il team di sviluppo di Corridors CMS si è ritrovato in questa situazione, ma la definizione dei requisiti fatta precedentemente e una comunicazione continua ha permesso di evitare il problema.

6.2 Git

Per lo sviluppo del progetto è stato di fondamentale importanza l'utilizzo di un Versioning Control System (VCS) come Git [27] che permettesse al gruppo di collaborare sul codice contemporaneamente e senza particolari sovrapposizioni. Git si basa sul concetto che ad ogni gruppo di cambiamenti svolti su una serie di file, viene fatto corrispondere un ID univoco, un nome e una descrizione che insieme vanno a delineare un commit. È inoltre possibile lavorare su diverse versioni (branch) dello stesso progetto, ognuna delle quali avrà le proprie commit. Il branch principale è il master e tutti gli altri sono sue copie. Durante lo sviluppo di Corridors CMS ad ogni task è corrisposto uno specifico branch. Quando un task viene completato, il codice correlato deve essere aggiunto su master, questa operazione è chiamata *merge*.

6.3 ClickUp

Il software che abbiamo utilizzato per l'organizzazione del lavoro è ClickUp.

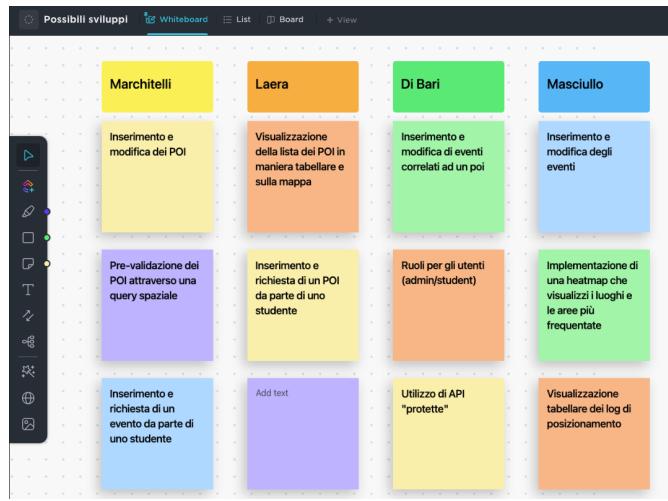


Figura 6.1: Whiteboard per il brainstorming

6.3.1 Funzionalità generali

Una volta formato un team di lavoro è possibile creare un workspace che rappresenta genericamente il progetto che si vuole realizzare. All'interno del workspace è possibile creare entità di natura diversa:

Liste Che contengono i task e sono visualizzabili in diverse forme

Docs Documenti di qualsiasi natura in condivisione

Whiteboard Che attraverso l'utilizzo di vari template permette di creare schemi e cards di supporto durante una seduta di brainstorming

6.3.2 Il workspace Corridors

Il workspace che abbiamo messo a punto per lo sviluppo della dashboard di Corridors è composto da una whiteboard e due liste.

Whiteboard - Possibili sviluppi

Questa whiteboard, rappresentata in Figura 6.1, è stata creata durante la fase di brainstorming e ha permesso al team di inserire di volta in volta tutte le nuove funzionalità da implementare all'interno del progetto. È usata anche per elencare tutti i possibili sviluppi futuri per l'intero Corridors.

In seguito, queste card, sono state analizzate dall'intero team e sono servite per la creazione della lista dei requisiti e dei casi d'uso.

Liste - FRONTEND e BACKEND

Queste liste, rappresentate in Figura 6.2, contengono tutti i vari task assegnati contenenti le features sia frontend che backend di Corridors. Ogni task è carat-

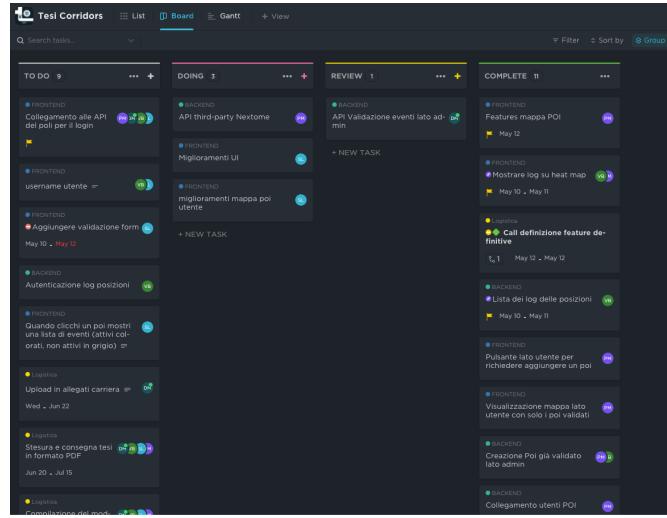


Figura 6.2: Liste dei task

terizzato da un titolo riassuntivo, una descrizione dettagliata della funzionalità che si andrà ad implementare e un indicatore dello stato di avanzamento.

All'interno della visualizzazione *board* sono state inserite 4 colonne per l'allocatione dei task:

TO DO All'interno di questa colonna vengono inseriti tutti i task appena creati e assegnati.

DOING Quando viene avviato lo sviluppo di uno specifico task, esso viene inserito nella colonna *doing* fino al suo completamento. Contestualmente deve essere creato anche un branch collegato al task in cui verranno inseriti i commit, man mano che lo sviluppo procede.

REVIEW Quando il task è ritenuto completo viene creata una *pull request* e viene spostato in questa colonna, in attesa di un test interno da parte di un altro membro del team che verifichi che la funzionalità sia stata implementata in maniera corretta.

COMPLETE Se il tester interno reputa il l'implementazione valida, viene approvata la pull request e contestualmente fatto il merge del branch su master. Se il merge è andato a buon fine il task viene spostato in questa colonna.

6.3.3 Vantaggi dell'utilizzo di ClickUp

L'approccio mediante ClickUp ha portato i seguenti vantaggi al team di sviluppo:

- Assegnazione dei task direttamente all'utente interessato;

- Inserimento di riferimenti temporali all'interno di un task, con lo scopo di specificare quando esso deve essere completato e la sua durata;
- Possibilità di visualizzare in tempo reale lo stato di avanzamento del progetto e del singolo task;
- Utilizzo da qualsiasi dispositivo;
- Integrazione con Git;
- Inserimento di eventi e riunioni all'interno del calendario;
- Visualizzazione dello sviluppo mediante componenti altamente personalizzabili.

6.3.4 ClickUp per il JAD

Questo strumento ha consentito al team di tenere traccia di tutti gli aspetti dello sviluppo di Corridors CMS in maniera semplice e intuitiva. Nello specifico ha permesso di realizzare il modello JAD descritto precedentemente, inglobando un'interfaccia utile all'esecuzione delle operazioni richieste ad ogni iterazione:

- **Definizione dei requisiti:** mediante la whiteboard *Possibili sviluppi*;
- **Progettazione e individuazione dei task:** inserendoli nella colonna *TO DO*;
- **Implementazione:** mediante il collegamento a Git e la possibilità di spostare i task nella colonna *REVIEW* quando ne viene ultimata l'implementazione;
- **Testing:** da parte degli altri membri del team quando notano che il task è arrivato in *REVIEW*.

Capitolo 7

Conclusioni

Corridors CMS è stato sviluppato con lo scopo di facilitare l'accesso al sistema informativo dell'app Corridors. Il risultato è stato la creazione di un portale semplice e intuitivo che permettesse agli utenti di operare autonomamente sui dati memorizzati nel database di Corridors.

Gli altri aspetti principali del CMS sono stati sviluppati nelle Tesi di Laurea di Masciullo Paolo, Laera Sante e Di Bari Vito che trattano e approfondiscono rispettivamente:

- Progettazione dell'architettura e sviluppo del backend in SpringBoot;
- Progettazione e sviluppo dell'intefaccia utente in React;
- Progettazione e implementazione dell'infrastruttura di sicurezza.

La dashboard ingloba tutte le funzionalità definite in fase di progettazione in quanto rispetta i requisiti funzionali e i diagrammi dei casi d'uso e implementa anche i diversi flussi di interazione con cui il sistema si interfaccia con gli utenti.

Il progetto, ad oggi, gestisce i flussi di interazione manualmente mediante il codice. L'evoluzione del portale prevede l'implementazione di nuovi workflow che andranno ad estendere l'esperienza utente. Per fare ciò si rende necessario l'utilizzo di un sistema che permetta di automatizzare e orchestrare i flussi fra gli utenti, i componenti del sistema e i diversi dispositivi. Questo tipo di servizio potrebbe essere offerto da Camunda [28], utile alla modellazione e automatizzazione dei diversi flussi che vengono rappresentati mediante lo standard open source *Business Process Modeling Notation* (BPMN) [23]. Il BPMN modella le fasi di un processo aziendale dall'inizio alla fine, illustrandone visivamente la sequenza dettagliata delle attività.

Il *Joint Application Development* ha guidato il team durante lo sviluppo del CMS, consentendo un delivery continuo di nuove funzionalità affinché il progetto si completasse nei tempi previsti. Il modello utilizzato si è rivelato ottimale in quanto ha saputo adattare lo sviluppo al contesto di realizzazione del progetto, parallelizzare il lavoro e favorire, mediante le *JAD sessions*, l'intervento del relatore nelle diverse fasi di progettazione e implementazione di ogni iterazione.

L'inserimento di nuove funzionalità permetterebbe un'evoluzione di Corridors che andrebbe di pari passo con la crescita del team di sviluppo. Ciò richiederebbe l'utilizzo di un modello di sviluppo più organico che potrebbe essere

rappresentato dallo *Scrum*, un framework basato sui principi di trasparenza, ispezione e adattamento che standardizza lo sviluppo dividendolo in Sprint (iterazioni), definisce i ruoli all'interno del team, standardizza i documenti che devono essere prodotti e stabilisce i meeting che devono essere effettuati dal team.

Lo Scrum porterebbe con sé numerosi vantaggi come l'accelerazione del *time-to-market* applicativo, una migliore qualità del codice, una riduzione del rischio di fallimento, una maggiore collaborazione e soddisfazione dell'utente e una massima compatibilità con le tecniche di sviluppo del futuro.

L'implementazione di questa nuova metodologia consentirebbe una crescita incrementale e costante delle funzionalità e dei servizi offerti dall'intero sistema Corridors grazie alla possibilità di coordinare un team di sviluppo sempre più grande ed eterogeneo.

Bibliografia

- [1] “IEEE Standard for Telecommunications and Information Exchange Between Systems - LAN/MAN - Specific Requirements - Part 15: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs),” *IEEE Std 802.15.1-2002*, pp. 1–473, Jun. 2002, conference Name: IEEE Std 802.15.1-2002.
- [2] L. Pei, R. Chen, J. Liu, T. Tenhunen, H. Kuusniemi, and Y. Chen, “Inquiry-Based Bluetooth Indoor Positioning via RSSI Probability Distributions,” in *2010 Second International Conference on Advances in Satellite and Space Communications*, Jun. 2010, pp. 151–156.
- [3] “Firebase App Check | Firebase Documentation.” [Online]. Available: <https://firebase.google.com/docs/app-check>
- [4] “progettazione nell’Enciclopedia Treccani.” [Online]. Available: <https://www.treccani.it/enciclopedia/progettazione>
- [5] OriginalSkills, “Pianificazione e organizzazione del lavoro,” Jun. 2018. [Online]. Available: <https://originalskills.com/it/blog/pianificazione-e-organizzazione-del-lavoro/>
- [6] “Nextome.com - Indoor Navigation, Indoor Positioning and Tracking.” [Online]. Available: <https://www.nextome.com>
- [7] A. Inc, “iBeacon.” [Online]. Available: <https://developer.apple.com/ibeacon/>
- [8] L. S. De Oliveira, O. K. Rayel, and P. Leitao, “Low-Cost Indoor Localization System Combining Multilateration and Kalman Filter,” in *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, Jun. 2021, pp. 1–6, iSSN: 2163-5145.
- [9] “Nextome SDK Documentation.” [Online]. Available: <http://docs.nextome.dev/>
- [10] D. R. Lanning, G. K. Harrell, and J. Wang, “Dijkstra’s algorithm and Google maps,” in *Proceedings of the 2014 ACM Southeast Regional Conference*, ser. ACM SE ’14. New York, NY, USA: Association for Computing Machinery, Mar. 2014, pp. 1–3. [Online]. Available: <https://doi.org/10.1145/2638404.2638494>

- [11] “IEEE Guide—Adoption of ISO/IEC TR 24748-1:2010 Systems and Software Engineering—Life Cycle Management—Part 1: Guide for Life Cycle Management,” *IEEE Std 24748-1-2011*, pp. 1–96, Jun. 2011, conference Name: IEEE Std 24748-1-2011.
- [12] H. D. Benington, “Production of Large Computer Programs,” *IEEE Annals of the History of Computing*, vol. 5, no. 4, pp. 350–361, Oct. 1983. [Online]. Available: <http://ieeexplore.ieee.org/document/4640770/>
- [13] D. W. W. Rovce, “MANAGING THE DEVELOPMENT OF LARGE SOFTWARE SYSTEMS,” p. 11.
- [14] A. Sinha and P. Das, “Agile Methodology Vs. Traditional Waterfall SDLC: A case study on Quality Assurance process in Software Industry,” in *2021 5th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)*, Sep. 2021, pp. 1–4, iSSN: 2767-9934.
- [15] N. D. Birrell and M. A. Ould, *A practical handbook for software development*, 1st ed. Cambridge: Cambridge Univ. Press, 1988.
- [16] J. Martin, *Rapid application development*. New York : Toronto : New York: Macmillan Pub. Co. ; Collier Macmillan Canada ; Maxwell Macmillan International, 1991.
- [17] “Extreme Programming - Metodologia di sviluppo agile,” Feb. 2016. [Online]. Available: <https://www.agileway.it/extreme-programming-metodologia-sviluppo-agile/>
- [18] H. Takeuchi and I. Nonaka, “The New New Product Development Game,” *Harvard Business Review*, Jan. 1986, section: Product development. [Online]. Available: <https://hbr.org/1986/01/the-new-new-product-development-game>
- [19] “Scrum: cos’è, come funziona e com’è strutturato,” Nov. 2015. [Online]. Available: <https://www.agileway.it/scrum-metodologia-agile/>
- [20] “IEEE Standard Glossary of Software Engineering Terminology,” *IEEE Std 610.12-1990*, pp. 1–84, Dec. 1990, conference Name: IEEE Std 610.12-1990.
- [21] G. Booch, J. Rumbaugh, and I. Jacobson, *The unified modeling language user guide*, ser. The Addison-Wesley object technology series. Reading Mass: Addison-Wesley, 1999.
- [22] “What is a Workflow? A Simple Guide to Getting Started | Process Street | Checklist, Workflow and SOP Software,” Oct. 2021. [Online]. Available: <https://www.process.st/what-is-a-workflow/>
- [23] “BPMN Specification - Business Process Model and Notation.” [Online]. Available: <https://www.bpmn.org/>
- [24] “Material Design.” [Online]. Available: <https://material.io/design/introduction/#principles>
- [25] L. Shklar and R. Rosen, *Web application architecture: principles, protocols, and practices*. Chichester, England ; Hoboken, NJ: John Wiley, 2003.

- [26] “Firebase Authentication | Firebase Documentation.” [Online]. Available: <https://firebase.google.com/docs/auth>
- [27] “Git.” [Online]. Available: <https://git-scm.com/>
- [28] J. Claria, “The Universal Process Orchestrator.” [Online]. Available: <https://camunda.com/>