**“Software Engineering”**

**Course**

**a.a. 2019-2020**

**Lecturer: Prof. Henry Muccini (henry.muccini@univaq.it)**

**<5, Portami a Destinazione>**

|  |  |
| --- | --- |
| **Date** | <05/12/2019> |
| **Deliverable** | Documento Iniziale – D1 |
| **Team (Name)** | NewBie Team |

|  |  |  |
| --- | --- | --- |
| **Team Members** | | |
| **Name & Surname** | **Matriculation Number** | **E-mail address** |
| Luca Di Laudo | *248444* | *luca.dilaudo@student.univaq.it* |
| Mattia Masci | *244146* | *mattia.masci@student.univaq.it* |
| Giovanni Gemini | *241972* | *giovanni.gemini@student.univaq.it* |

Table of Contents of this deliverable

Sommario:

List of Challenging/Risky Requirements or tasks………………………………3

A Stato dell’Arte…………….…………………………………….……………...….................5

B Raffinamento dei Requisiti………………………………….……………...................8

B.1 Requisiti Funzionali.................................................................................9

B.2 Requisiti non Funzionali........................................................................11

B.3 Scenari d’uso dettagliati.........................................................................12

B.4 Excluded Requirements.........................................................................15

B.5 Assunzioni..................................................................................................15

B.6 Use Case Diagram....................................................................................16

C Architettura Software..........................................................................................17

C.1 Component Diagram...............................................................................17

C.2 Sequence Diagram...................................................................................19

D Dati e la loro Modellazione................................................................................22

D.1 Modello E-R...............................................................................................22

D.2 Tabelle.........................................................................................................24

E Design Decisions...................................................................................................26

F Design di Basso Livello.......................................................................................29

F.1 Class Diagram...........................................................................................29

G FR’s & NFR’s...........................................................................................................31

H Effort Recording...................................................................................................33

H.1 Pert Diagram..................................................................................................33

H.2 Logging……………………………………………………………………………….….33

## List of Challenging/Risky Requirements or Tasks

|  |  |  |  |
| --- | --- | --- | --- |
| **Challenging Task** | **Date the task is identified** | **Date the challenge is resolved** | **Explanation on how the challenge has been managed** |
| Come visualizzare una mappa all'interno del software | 20/11/19 | 20/11/19 | Dopo esserci documentati sulle tecniche più usate abbiamo deciso che importeremo una mappa dall'esterno. Controllando in giro abbiamo notato che la maggior parte delle aziende utilizza Google Maps per incorporare mappe nel proprio sito. Documentandoci su Google Maps Platform abbiamo trovato la libreria “Maps JavaScript API” che si integra perfettamente con il nostro progetto e ci permette di collegare i markers con il nostro database e di manipolarli attraverso il PHP. |
| Usare MySQL o NoSQL | 27/11/19 | 27/11/19 | Abbiamo deciso di utilizzare il database MySQL visto che dopo un'analisi veloce abbiamo notato che I dati da manipolare hanno una struttura definita per cui MySQL dovrebbe permetterci di lavorare con efficienza. Inoltre a partire dall’aggiornamento ‘MySQL8’ sono state introdotte diverse funzionalità sulla geolocalizzazione, per esempio il tipo POINT ci permette di definire un punto con delle coordinate precise, e aggiungendo anche uno SRID (un ID numerico che si collega a un sistema di riferimento spaziale) avremo la certezza di calcolare delle misure corrette. Per finire nella documentazione di Google Maps viene spiegato che la libreria già menzionata supporta solo database MySQL. |
| Come progettare il prototipo | 27/11/19 | 02/12/19 | Visto che il sistema è pensato per essere un servizio aggiunto al sito di un'università abbiamo deciso di implementare il progetto come un’applicazione web, in particolare svilupperemo un web service che rispecchierà i vincoli architetturali REST. |
| Rischio di creare confusione in caso ci siano molti POI in uno spazio ristretto | 13/12/19 | 17/12/19 | Come prima cosa abbiamo evitato questo ostacolo nel caso dell’utilizzo da parte di un utente non registrato impostando inizialmente tutti i filtri a zero. Il software viene creato per offrire una piattaforma che accompagni in modo semplice e veloce l’utente nella ricerca di ciò di cui ha bisogno, quindi l’utilizzatore medio a cui noi facciamo riferimento generalmente farà ricerche mirate a seconda delle proprie necessità momentanee (ovvero selezionerà un servizio per volta o comunque un numero esiguo di questi contemporaneamente). Così facendo non limitiamo l’utente con restrizioni di nessuna natura, ma dovrebbe risultare automatico per chi entra non loggandosi nel sito lavorare senza creare confusione.  Per quanto riguarda gli utenti registrati abbiamo agito cercando di tendere all’essenziale, spiegheremo tutto nei dettagli nella parte ‘design decisions’. |
| Grandezza delle icone che rappresentano I POI | 02/01/20 | 02/01/20 | La grandezza delle icone non è un dettaglio da sottovalutare, infatti sceglierle troppo grandi potrebbe nascondere ciò che si trova intorno a queste, mentre optare per delle icone troppo piccole implicherebbe un grado di zoom esagerato per distinguerle. Inoltre, se possibile, anche la soglia di zoom oltre il quale non mostrare le icone va scelta accuratamente per evitare il rischio di creare confusione e scarsa usabilità. |

1. Stato dell’Arte

Consultando vari siti universitari italiani abbiamo notato che molti di questi sono sprovvisti di una mappa virtuale, per quanto riguarda i pochi che ne sono dotati viene offerto solo un servizio base. Confrontando tutte le pagine visitate abbiamo cercato di estrapolare il minor numero di esempi per sottolineare i dettagli che pensiamo siano più interessanti.

Un esempio è l'Università di Bologna che mette a disposizione una mappa abbastanza rudimentale creata attraverso Leaflet, la principale libreria JavaScript open source per mappe interattive ottimizzate per dispositivi mobili. Secondo la documentazione il peso di questa libreria è veramente minimo ma comunque permette tutte le funzionalità di mappatura di cui la maggior parte degli sviluppatori ha bisogno.

Per la verità pensiamo che l’interfaccia sia poco ‘attraente’ e interattiva, inoltre non viene mostrata la posizione dell’utilizzatore né quindi possono essere calcolati percorsi per arrivare ai POI. Per quest’ultimi vengono usate delle icone abbastanza piccole che vengono mostrate solo da una certa soglia di zoom, questa è una soluzione interessante anche se c’è il rischio di dover ricercare i punti con una visione troppo ravvicinata che potrebbe indurre a ‘perdersi nella mappa’. Trovando il giusto limite di zoomata potrebbe essere un buon espediente.

Nel complesso crediamo che tutto questo non sia abbastanza per invogliare le persone ad un impiego frequente del software, quindi visto che lo scopo finale è rendere popolare e affidabile il sistema scartiamo gran parte delle caratteristiche di questa possibilità come fonte d’ispirazione.

Link: <https://www.unibo.it/uniboWeb/unibomappe/default.aspx>

A nostro avviso il sito più fornito tra quelli che abbiamo visionato è quello della "Sapienza Università di Roma". La mappa dell'università è creata utilizzando le API di Google Maps Platform, una raccolta di API che consente di sovrapporre dei dati a una mappa personalizzata di Google. In particolare, tutto questo è stato fatto attraverso GeoCMS che è un Content Management System basato appunto sulla tecnologia Google Maps Platform per la gestione di informazioni geolocalizzate destinate alla fruizione su web, dispositivi mobili e altre piattaforme.

Questo strumento fornisce un pacchetto di servizi web attraverso cui gestire, pubblicare e ricercare dati geo, offrendo un supporto sistemistico e funzionale che agevola queste attività. Naturalmente tutto questo è a pagamento.

La pagina della Sapienza a prima vista è animata e molto intuitiva, mette a disposizione una mappa con molte funzionalità di cui tra l'altro tante delle quali richieste nella specifica del nostro progetto.

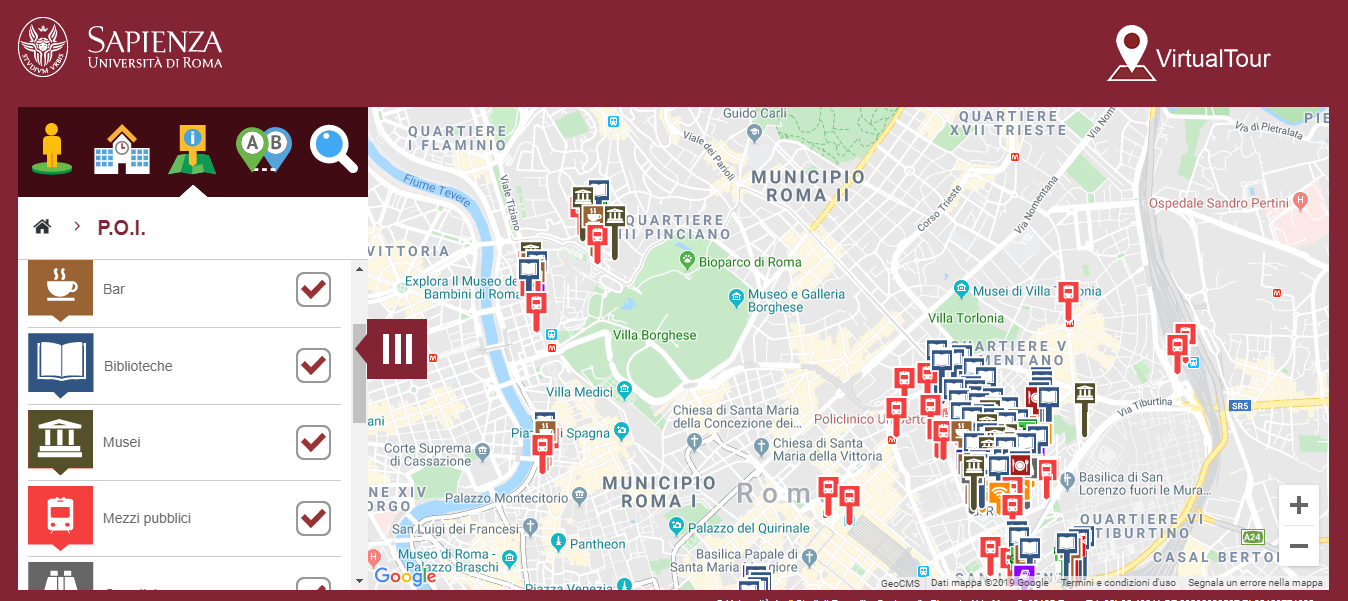
Uno dei problemi del sito è che scegliendo la categoria POI viene mostrata direttamente una mappa con tutti i servizi selezionati e questo rende la vista molto confusionaria. Inoltre, le icone dei POI selezionati, che sono abbastanza grandi, rimangono in vista qualunque sia il grado di zoom impostato rendendo anche difficoltosa la selezione se la visuale è lontana, questa potrebbe essere sicuramente una funzionalità da migliorare. L’unico metodo per avere una percezione precisa delle collocazioni dei POI quando ce ne sono diversi adiacenti è zoomare e avere una visuale mirata.

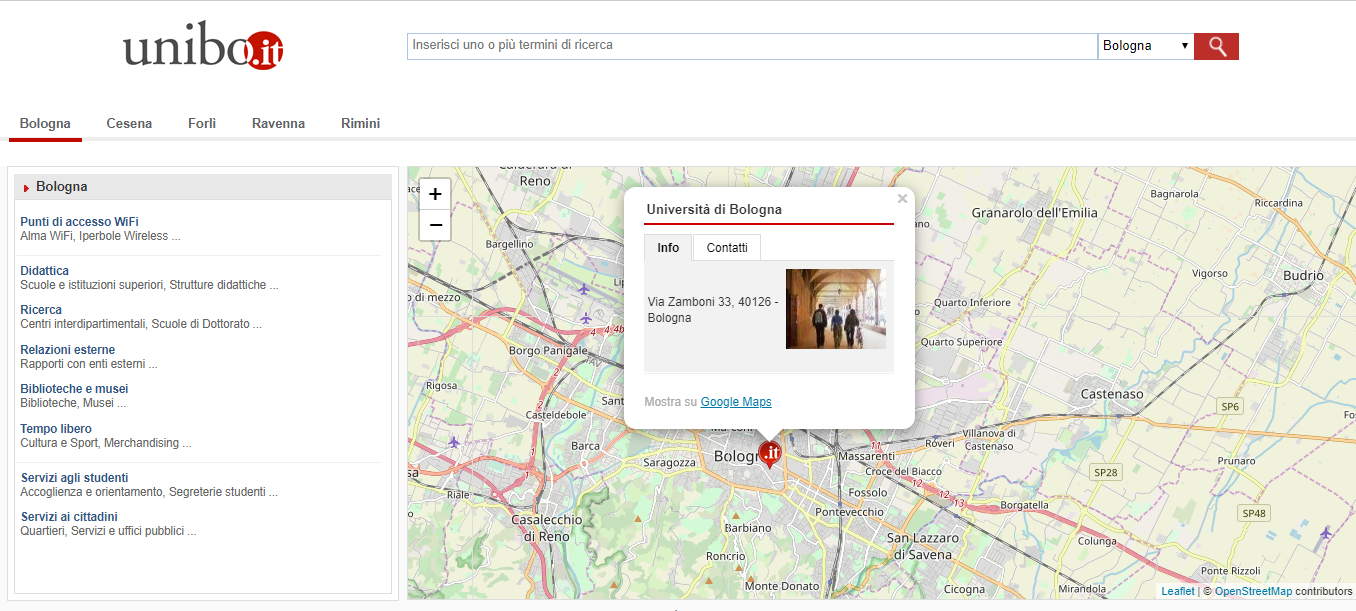
Un servizio molto utile che mette a disposizione il tour virtuale della Sapienza è quello che permette all’utente di calcolare il percorso in base alla propria posizione verso i vari edifici dell’Ateneo. Purtroppo, è possibile utilizzare questa comodità solo in una sezione apposita e unicamente per gli edifici, quindi non per tutto l’insieme dei POI.

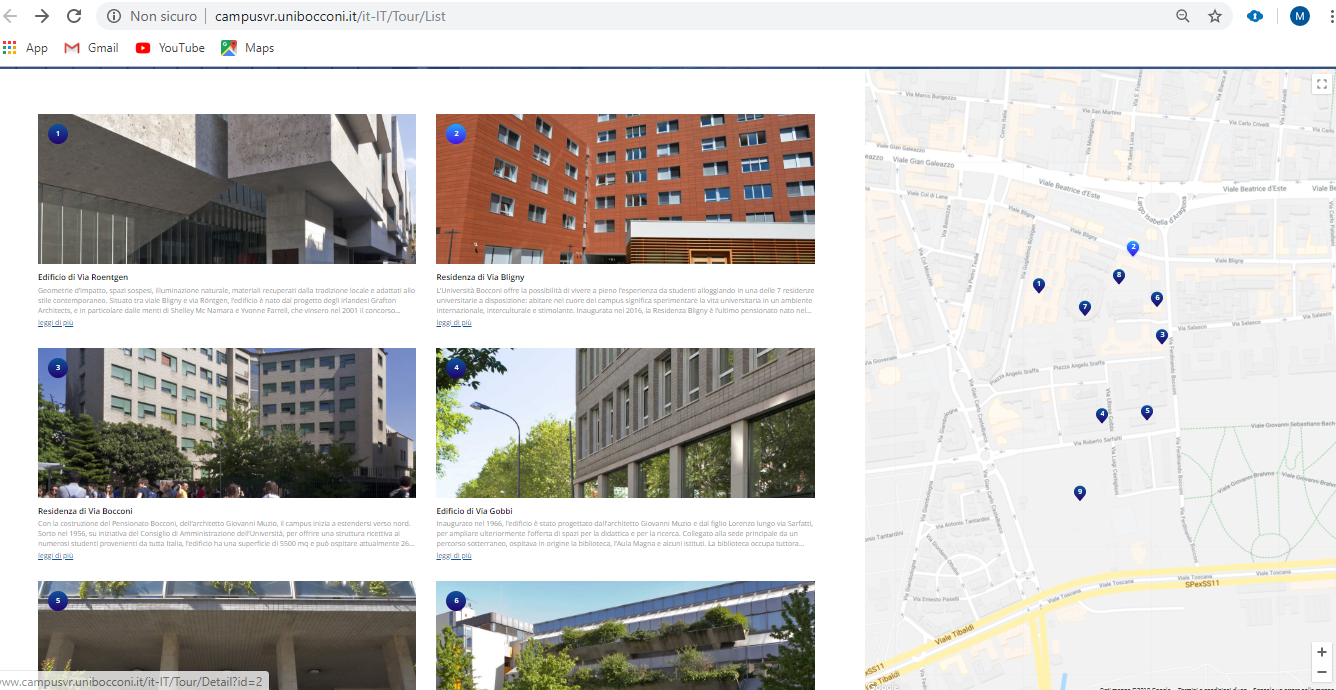
Una funzionalità che non viene fornita da nessuno di questi siti è la visualizzazione della propria posizione in tempo reale sulla mappa, questo implica che l’utente debba fare uno sforzo magari evitabile per orientarsi. L’aggiunta della posizione istantanea potrebbe far veramente la differenza tra un servizio utile e uno completamente soddisfacente. Comunque questo è un servizio che può causare difficoltà nell’implementazione poiché determinare la posizione all’interno degli edifici potrebbe non essere così semplice considerando scenari particolari dove non c’è copertura della rete cellulare o della rete WI-FI.

Tornando alla Sapienza ci è piaciuto molto il menù attraverso cui si scelgono le varie categorie di POI, la homepage è progettata in modo da rendere chiaro lo scopo e le funzionalità del sistema per cui prenderemo spunto da questa GUI.

Link: <https://virtualtour.uniroma1.it/>

**





B. Raffinamento dei Requisiti

Nel contesto dell’innovazione tecnologica che sta travolgendo tutto il mondo ormai da anni è stata riscontrata la necessità di realizzare un software, in ambito universitario, in grado di agevolare gli studenti, i docenti e tutto il personale di raggiungere i vari punti di interesse all'interno degli atenei. Si potrebbe voler sapere la posizione di edifici specifici per i propri studi, per svolgere attività amministrative, o ancora degli uffici dei professori, delle mense...

Il fine ultimo di questo software quindi è quello di condurre una persona in un determinato luogo con lo scopo di farlo nel modo più semplice e affidabile possibile.

Poiché l'applicativo dovrà essere utilizzato in un contesto prettamente universitario, e quindi da una vasta tipologia di persone che rivestono diversi ruoli, da uno studente immatricolato ad uno non immatricolato, a un nuovo professore o ancora persone invitate ad una discussione di laurea, l'interfaccia utente dovrà essere semplice ed intuitiva così da ottimizzare al meglio la User-Experience cercando di evitare il più possibile eventuali bug o errori nel sistema.

B.1 Servizi (con prioritizzazione)

Abbiamo prioritizzato i requisiti funzionali trovati in priorità bassa, media, alta, secondo lo schema a pag.35 delle slides riguardanti la “Requirement Engineering”.

Comandi Vocali (MEDIA)

L’utente avrà la possibilità di utilizzare i comandi vocali per interagire con il sistema, come ad esempio la ricerca di uno specifico POI o Edificio;

Registrazione (BASSA)

Un utente avrà la possibilità di registrarsi nel portale specificando Nome, E-mail, Password, e facoltativamente il proprio CdL.

Login (BASSA)

L’utente avrà due possibilità di effettuare il Login:

* in una sezione dedicata utilizzando E-mail e Password (in base a questo il sistema poi specializzerà la visualizzazione della mappa secondo il CdL frequentato);
* senza immettere E-mail e Password (con vista generale riguardo tutto il complesso Universitario).

Per quanto riguarda l'admin gli sarà possibile un accesso separato in cui immettere il proprio codice univoco.

Modifica dati personali (BASSA)

Sarà permesso all’Utente di poter cambiare le proprie informazioni personali in caso di particolari situazioni, come ad esempio il passaggio da una facoltà ad un’altra.

Visualizzazione mappa (ALTA)

Il sistema sarà in grado di visualizzare una mappa che segnalerà tutti i POI sul territorio in cui sono distribuiti. I POI saranno rappresentati con icone diverse a seconda della tipologia di questi.

Sarà sempre presente un’icona nella mappa che mostrerà la propria posizione in tempo reale, ammesso che l’utente acconsenta la condivisione dei propri dati di geolocalizzazione.

Sarà possibile usufruire di una funzionalità che, immettendo la posizione di partenza e quella di arrivo, calcolerà il percorso adeguato per portare il richiedente a destinazione.

Filtraggio POI (ALTA)

Il sistema sarà in grado di filtrare e rappresentare sulla mappa in base alle esigenze dell'utente, selezionando/deselezionando servizi, edifici, uffici o utilizzando una barra di ricerca, i vari POI desiderati.

Inoltre, non appena un utente registrato entrerà nella propria pagina visualizzerà una mappa all'interno di cui saranno segnalati automaticamente i POI più importanti collegati al proprio CdL, sarà poi premura dell'utilizzatore lasciare inalterate o cambiare le impostazioni divisualizzazione. A questo scopo verrà fornita un’opzione che permetterà di passare da una visualizzazione incentrata sul proprio CdL ad una generica di tutto l’Ateneo.

Gestione back-end del sistema(MEDIA)

L'admin, dopo aver effettuato il login tramite una UI specifica, potrà aggiungere/rimuovere POI, che siano edifici, uffici o fornitori di servizi. L’admin sarà in grado quindi di gestire in back-end tutto il sistema apportando le relative modifiche per aggiungere/rimuovere gli oggetti sopra descritti.

B.2 Requisiti non Funzionali

Usability

Il Sistema dovrà essere strutturato in modo basico e di facile comprensione per garantire una user-experience funzionale ma allo stesso tempo semplice. La GUI dovrà essere il più funzionale e user-friendly possibile.

Scalability

Il sistema dovrà essere in grado di gestire un gran numero di utenti in contemporanea, ancor di più nei periodi di iscrizione in cui l'affluenza nelle sedi si intensifica. Il DBMS, che sarà una delle parti del software messa più a dura prova in questi periodi, dovrà essere messo in condizione di eseguire un gran numero di transazioni all'occorrenza.

Capacity

Il software dovrà essere in grado di poter funzionare in una grande Università potenzialmente distribuita su più sedi, e con una molteplicità importante di POI*.*

Supportability

Gli utenti useranno il software attraverso i sistemi più disparati (dagli smartphone ai pc, con la conseguente varietà di piattaforme), per questo dovrà essere progettato tenendo conto di questa diversità.

Maintainability

Il software dovrà rispettare la Manutenibilità, ovvero la capacità di essere Modulare. Così sarà più facile individuare una parte di codice interessata ad eventuali modifiche o correzioni. Per fare ciò è necessario che il codice sia ben commentato e documentato;

B.3 Scenari d’uso dettagliati

*Raggiungimento di un POI (Utente con Account)*

1. L’utente apre la pagina web;
2. Una volta aperta la pagina, l’utente effettua il login con password;
3. Il software mostra una vista della mappa specializzando la vista in base al CdL inserito al momento della registrazione, ovvero mostrando tutti i POI vicini all’edificio di appartenenza del corso di laurea;
4. L’utente potrà scegliere di quale Servizio necessita nel menù apposito;
5. Non appena che l’utente avrà scelto, il software elaborerà la destinazione e mostrerà tutti i POI collegati al determinato Servizio.

Raggiungimento di un POI (Utente senza account)

1. L’utente apre la pagina;
2. Una volta aperta la pagina, l’utente effettua il login come ospite;
3. Il software mostra una mappa priva di markers, quindi non prioritizzando per mezzo del CdL come nel caso dell’utente registrato;
4. L’utente potrà scegliere di quale Servizio necessita nel menù apposito;
5. Non appena che l’utente avrà scelto, il software elaborerà la destinazione e mostrerà tutti i POI collegati al determinato Servizio.

Registrazione Utente

1. L’utente apre la pagina;
2. Una volta aperta la pagina l’utente si reca nella sezione della registrazione;
3. Aperta la pagina di registrazione inserisce i dati richiesti dal software come E-mail, scelta della password, CdL...;
4. Appena finito di immettere tutti i dati richiesti il sistema invia una mail di conferma all’indirizzo che l’utente ha inserito al momento della registrazione;
5. L’utente dovrà recarsi nella propria casella di posta per cliccare sul link di avvenuta conferma e verrà rimandato alla pagina iniziale del software come utente registrato a tutti gli effetti.

Modifica dati personali

1. L’utente apre la pagina;
2. Una volta aperta la pagina, l’utente dovrà necessariamente effettuare il login;
3. Appena effettuato il login, l’utente si deve recare nella sezione del proprio profilo e cliccare sul tasto di modifica del profilo;
4. Una volta modificati i propri dati personali, per mezzo di un form, apparirà una finestra con la richiesta di conferma delle modifiche apportate;
5. Confermato il tutto il software registrerà nel database i dati modificati dall’utente, quindi le modifiche verranno apportate ufficialmente.

Ricerca di un POI riguardante il proprio CdL (Utente con Account)

1. L’utente apre la pagina ed effettua il login;
2. Appena effettuato il login l’utente vedrà nella mappa i POI più importanti in modo da mantenere il più possibile pulita la schermata (Mense, Segreterie, Punti informativi, Mezzi pubblici) riguardanti il proprio CdL;
3. Appena individuata l’icona adeguata per il servizio cercato l’utente può raggiungerla o cliccarci sù per reperire le informazioni necessarie.

Ricerca di un POI non riguardante il proprio CdL (Utente con Account)

1. L’utente apre la pagina ed effettua il Login;
2. Appena effettuato il login l’utente vedrà i POI riguardanti il proprio CdL;
3. L’utente seleziona POI nel menù laterale, nello spazio sottostante compariranno tutti i servizi di cui gli unici già selezionati saranno ‘Mense’, ‘Segreterie’, ‘Punti informativi’ e ‘Mezzi pubblici’;
4. A questo punto verrà selezionata la casella di fianco lo specifico servizio ricercato (ammesso che non sia uno di quelli già spuntati) e compariranno all’interno della mappa tutte le icone rappresentanti quel servizio (attinenti al proprio CdL);
5. Si avrà la possibilità di visualizzare i POI riguardanti l’intero Ateneo spuntando l’apposita casella ‘Visualizza in base al CdL di appartenenza’;
6. Allora l’utente vedrà tutti i POI (collegati ai servizi già selezionati) sparsi per l’Ateneo (quindi una vista generalizzata su tutte le sedi) e potrà filtrare ad esempio per il servizio ‘Biblioteche’, di conseguenza visualizzerà anche le biblioteche di altre sedi che non riguardano il proprio CdL.

Ricerca di un Ufficio di un Professore

1. L’utente apre la pagina effettuando il Login oppure entrando come ospite;
2. L’utente seleziona Dipartimenti nel menù laterale, nello spazio sottostante compariranno tutti i dipartimenti dell’Ateneo;
3. A questo punto verrà selezionato il dipartimento nel quale il professore ricercato insegna, una volta cliccato su questo l’utente dovrà scegliere se elencare gli Edifici che compongono il dipartimento oppure gli Uffici dei docenti che vi insegnano;
4. Dopo che si è scelto Uffici verranno elencati i vari Professori, selezionando l’insegnante cercato verrà mostrato l’Edificio all’interno di cui è posizionato il corrispondente Ufficio con un commento riguardante il piano in cui trovarlo e il numero della stanza.

B.4 Excluded Requirements

-Non è previsto che l'utente possa mandare segnalazioni (come mandare messaggi, dare feedback o segnalare nuovi POI) al sistema, lo scopo del software è fornire all’utilizzatore un servizio basico per avere un orientamento all’interno del complesso universitario in caso di bisogno. Si potrà pensare di aggiungere questa funzionalità durante raffinazioni successive.

B.5 Assunzioni

-I POI saranno categorizzati e mostrati nel sistema divisi in edifici, fornitori di servizi (servizio inteso nell’accezione più generale possibile, quindi includeranno anche segreterie studenti, musei, punti informativi…) e uffici.

-Ogni ufficio sarà collocato all'interno di un edificio.

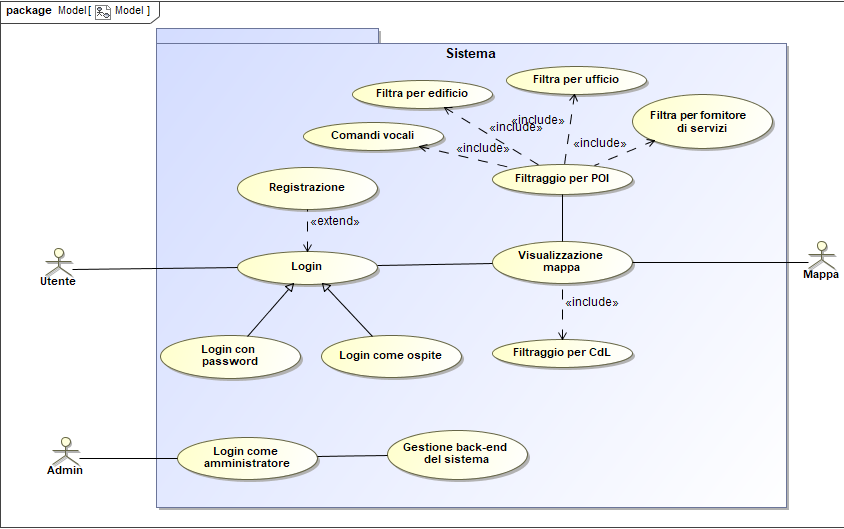
-Quando si parlerà di uffici ci si riferirà solo a uffici di professori.

-I fornitori di servizi saranno in qualche modo attinenti a quelli che potrebbero essere i bisogni e gli interessi degli studenti, dei docenti e del personale all'interno dell'università.

-Ogni utente potrà visualizzare tutti i POI in relazione con l'università.

-Il software utilizzerà un sistema di navigazione esterno che verrà importato.

B.6 Use Case Diagrams



C. Architettura Software

Per quanto riguarda la progettazione abbiamo utilizzato lo stile architetturale Model / View / Controller (MVC).

Le componenti sono classificate in:

* **modello**, per mantenere la conoscenza del dominio;
* **visualizzazione**, per presentare i dati all’utente;
* **controller**, per gestire l'elaborazione dei dati immessi dall'utente e rendere utilizzabili quelli di cui ha bisogno.

Questo paradigma trova la sua applicazione principale nei programmi con interfaccia grafica (GUI) e nei sistemi con interfaccia web.

Grazie a questo modello siamo riusciti a fornire un quadro chiaro del design del progetto da realizzare e a dettare le distinzioni di responsabilità tra le tre categorie e il flusso di informazioni tra di esse.

C.1The static view of the system: Component Diagram

*Immagine che contiene screenshot

Descrizione generata automaticamente*

Il Component Diagram ha lo scopo di rappresentare la struttura interna del sistema software, modellando i componenti principali e le relazioni fra di essi.

Nel dettaglio, i componenti presenti nel nostro sistema sono:

* **View**: contiene tutte le pagine mostrate all’utente e le componenti con cui esso interagirà, in particolare:

- UserInterface: è la parte di sistema che interagisce con l’utente, comunica con la componente “Map” in modo tale che possa visualizzare l'intera mappa dell'ateneo.

Quando per esempio un utente richiede informazioni riguardanti un determinato Servizio, oppure un determinato POI, tale componente invia una richiesta HTTP alla componente “Server”. Quest'ultima, dopo una serie di interazioni con le altre parti del sistema, caricherà una pagina con le informazioni richieste sul client;

- AdminInterface: è l’interfaccia accessibile solo dagli amministratori, essi potranno gestire in back-end tutto il sistema aggiungendo e rimuovendo per esempio dati di uno specifico Utente, Servizio, POI, ecc...

- Map: tale componente ha lo scopo di comunicare con “Google Maps” per visualizzare una mappa fornita di tutte le informazioni necessarie all’utente.

* **Server**: elabora le richieste del client per dotarlo dei servizi di cui ha bisogno.

- Routes: componente logica del sistema che mappa le varie richieste con gli elementi del software atti a soddisfarle;

- Middleware: agisce da filtro per le richieste HTTP, nel nostro caso si occuperà di proteggere quelle rotte che possono essere utilizzate solo dagli utenti loggati o dall’admin. Nel caso in cui un URL sia protetto e l’utente non sia autenticato verrà subito reindirizzato nella pagina di login.

* **Controller**: hanno lo scopo di raggruppare la logica di gestione delle richieste, ogni controller mette a disposizione dei metodi di base (index, show, update, destroy) per andare a operare sugli elementi che gli competono. Vengono richiamati dalle routes o dal middleware e interagendo con il Model mettono a disposizione i dati o i servizi necessari.
* **Model**: gestisce lo scambio di informazioni tra la sorgente dati e l’interfaccia utente, attraverso il Controller.

-Migrations e Seeds: procedure che hanno lo scopo di creare tabelle e relazioni all’interno del database e di riempirlo con i dati necessari.

- Database: si occupa di memorizzare e rendere disponibile il reperimento dei dati che il sistema gestisce. Attraverso i vari Model comunicheremo con il Database in modo semplice per mettere a disposizione dei Controller i dati. Questo sistema ci garantisce sicurezza e consistenza dei dati.

C.2 The dynamic view of the software architecture: Sequence Diagram

Il Sequence Diagram è un diagramma utilizzato per descrivere i vari scenari attinenti a un progetto, ovvero una determinata sequenza di azioni in cui tutte le scelte sono state già effettuate.

Nel primo esempio mostriamo come la richiesta dell'utente venga deviata verso il middleware che, fatti i dovuti controlli, reindirizzerà l’elaborazione al **LoginController**. Quest’ultimo mostrerà una pagina di login dove immettere i dati, una volta ricevuti li invierà al database per controllarne l’esistenza. Avvenuta la validazione sarà restituito il controllo all’**HomeController** che visualizzerà all’utente la pagina richiesta.

Nel secondo esempio invece è rappresentato un possibile scenario in cui l'**amministratore** implementa un comando che scatena delle esecuzioni, abbiamo riportato delle possibili operazioni esemplificative richiamate a seconda del comando eseguito. In particolare gli esempi riportano la richiesta e relativa risposta per la lista di tutti, o anche un singolo, markers, e per la creazione, modifica o eliminazione di una riga nel database della tabella analoga.

Immagine che contiene testo, mappa

Descrizione generata automaticamenteImmagine che contiene interni, tavolo

Descrizione generata automaticamente

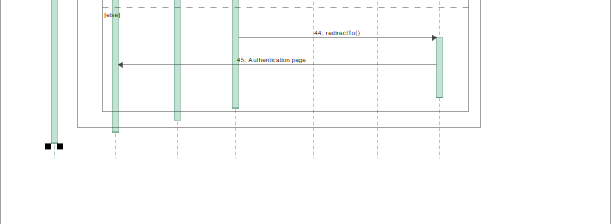
Immagine che contiene testo, mappa

Descrizione generata automaticamenteImmagine che contiene testo, mappa

Descrizione generata automaticamente

Immagine che contiene testo, mappa

Descrizione generata automaticamente



D. Dati e loro modellazione

D.1 MODELLO E-R

Immagine che contiene mappa, testo, interni

Descrizione generata automaticamente

D.2 TABELLE

-Nella tabella ‘Users’ saranno memorizzati tutti gli utenti registrati immettendo Nome, Cognome, Codice fiscale, Password, E-mail e CdL. Inoltre ci sarà un campo specifico che sarà “not null” solo in caso l’utente sia un Admin, qui sarà registrato un codice che utilizzerà per effettuare il login.

-La tabella ‘Degreecourses’ conterrà tutti i corsi di laurea che si svolgono nei 'Dipartimenti' di tutte le 'Università'.

-La tabella ‘Marker’ sarà quella con più istanze e starà a elencare tutti i possibili punti di interesse, la colleghiamo ai servizi così da poter dividere i POI per categorie.

-Nella tabella ‘Service’ sarà presente un campo Boolean per distinguerli in importanti e non.

-La tabella 'Markerhasdegreecourse' ci aiuterà a rintracciare quali POI sono appunto collegati, per vicinanza o attinenza, ai vari CdL.

-Grazie alle relazioni tra le tabelle ‘Offices’ e ‘Marker’, ‘’Office’ e ‘’Teacher’ potremo risalire alla collocazione dei vari uffici all'interno degli edifici universitari.

-La tabella ‘’Photo’ servirà per mettere a disposizione delle immagini rappresentanti gli edifici.

-Nella tabella ‘’Classroom’ saranno registrate tutte le aule più importanti di ogni edificio.

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

Immagine che contiene screenshot

Descrizione generata automaticamente

E. Design Decisions

LOGIN

Nella pagina iniziale sarà possibile scegliere tra le varie opzioni di accesso: login come ospite, login con autenticazione. Per gli admin sarà fornito un URL apposito per l’accesso in cui inserire i propri dati personali. In particolare gli amministratori avranno tutti gli stessi privilegi e permessi.

DATI E PROTEZIONE

Il DBMS ci fornirà un livello di sicurezza base, non adotteremo tecniche specifiche visto che non ci saranno dati sensibili all'interno del sistema ma solo quei dati indispensabili di cui il il software ha bisogno per offrire le varie funzionalità.

FRAMEWORK E LINGUAGGI UTILIZZATI

Per creare il web service che implementerà il servizio della mappa utilizzeremo primariamente Laravel.

Integreremo il framework con MySQL così da interagire attraverso il linguaggio PHP.

Per testare gli ENDPOINT con i vari esempi di risposta, di errore e le corrispondenti descrizioni testuali usufruiremo di Postman.

MAPPA E FILTRAGGIO

Abbiamo deciso che la scelta migliore è incorporare la mappa di Google nella nostra pagina visto il poco tempo a disposizione. La mappa sarà sempre visibile nella homepage e verrà inizializzata con icone diverse a seconda dell'utente che esegue il login.

Andremo a costruire il database in modo da poter fornire questa visualizzazione diversificata dei POI.

Faremo in modo di mostrare all’utente un menù per la selezione dei POI in modo da permettere una ricerca più naturale di questi. Nel menù principale si potrà scegliere tra ‘Dipartimenti’ e ‘Servizi’ (POI), cliccando su ciascuna di queste gli verranno visualizzate tutte le opzioni disponibili.

Cliccando su Dipartimenti verrà mostrato un elenco dei vari dipartimenti riguardanti l’Ateneo, ognuno di questi sarà collegato con uno o più edifici che verranno mostrati e successivamente anche rappresentati sulla mappa se richiesto dall’utente. Una volta scelto un dipartimento saranno fornite due opzioni:

1. ‘Edifici’ per elencare tutti gli edifici che compongono il dipartimento. Selezionandone uno poi verrà selezionato sulla mappa lo stabile analogo con la possibilità di avere informazioni o foto di questo.
2. ‘Uffici’ per elencare tutti gli uffici dei docenti attinenti allo specifico dipartimento. Cliccando poi su uno specifico Professore verrà selezionato sulla mappa l’edificio che ne ospita l’ufficio e con un commento verranno indicati il piano e il numero della stanza corrispondenti.

Uno dei nostri scopi è rendere la visualizzazione il più possibile ordinata e pulita per cui discutendo sui vari casi abbiamo preso queste decisioni:

* Per quanto riguarda gli utenti non registrati, che quindi non avranno un settaggio dei POI già impostato, abbiamo deciso che la mappa non mostrerà inizialmente alcuna icona per non rendere confusionaria la rappresentazione. Per questo originariamente saranno deselezionati tutti i filtri.
* All’opposto troviamo il caso degli utenti registrati che dopo aver effettuato il login potranno visualizzare nella mappa solo i POI attinenti al proprio CdL, in particolare verranno automaticamente impostati solo i più importanti (mense, segreterie, punti informativi e mezzi pubblici). Per fare una ricerca su altri punti l’utente dovrà normalmente selezionare/deselezionare i vari servizi, sempre tenendo conto che verranno indicati solo quelli collegati al proprio CdL. L’utente può richiedere una visione generalizzata dei servizi spuntando la casella ‘Visualizza in base al CdL di appartenenza’ (selezionata di default), che annullerà le impostazioni automatiche del sistema e gli permetterà di comportarsi come se fosse un ospite non loggato.

Per finire si potrà usufruire di una barra di ricerca che permetterà di inserire una stringa e ricercare tutti gli Edifici e/o Aule inerenti a questa (comunque filtrati in base a CdL se l’utente è loggato). In particolare, per le Aule verrà mostrato l’Edificio di appartenenza e un commento specificherà il piano in cui sono situate ed il numero di stanza.

Tutte le disposizioni prese sono precauzioni per aiutare l’utente ad evitare situazioni confusionarie, sarà comunque sempre compito di questo utilizzare il software nel modo adeguato.

COMANDI VOCALI

Il nostro obiettivo è quello di fornire qualcosa che ci eviti di impiegare troppo tempo per implementare una funzione che comunque non è di primaria importanza per il progetto. Una buona soluzione per rispettare l’aggiunta di questa funzionalità è integrare nel software un sistema esterno di supporto vocale che dia la possibilità di tradurre i comandi vocali in operazioni svolte dal sistema.

Un esempio potrebbe essere la libreria Voix.js, questo set di funzioni è realizzato in JavaScript e mette a disposizione il supporto per integrare comandi vocali nelle pagine del proprio sito Web o della propria Web application. Voix.js è gestibile tramite un’API denominata ‘voix()’, richiamandola si potranno definire delle istruzioni passando ad essa come parametri il comando vocale da interpretare e la funzionalità ad esso associata.

Un ulteriore esempio potrebbe essere l’integrazione con l’assistente vocale “Alexa” di proprietà di Amazon che nell’ultimo periodo sta spopolando e cambiando completamente il concetto di assistente virtuale, specialmente perché sviluppare una skill Alexa è semplice e ci sono svariati corsi sia online che non dove si spiega in modo intuitivo i vari processi per la realizzazione.

F. Design di Basso Livello

F.1 CLASS DIAGRAM

Immagine che contiene screenshot

Descrizione generata automaticamente

Abbiamo suddiviso il nostro class diagram secondo le funzionalità che offrono le varie classi.

Nella sezione **view** possiamo trovare tutte le classi che interagiscono per andare a formare l’interfaccia del sistema.

Nella pagina di autenticazione l’utente dovrà immettere i propri dati e inviarli attraverso un form, a questo punto verrà richiamato il **LoginController** che si occuperà di validare o no la richiesta.

Simile procedimento per la modifica dei dati personali o la registrazione, in cui il sistema richiederà attraverso un form di inserire informazioni personali all’utente e successivamente di validare la richiesta per mezzo di una store() o un’update().

Nella propria homepage l’utente potrà selezionare o deselezionare i POI da visualizzare in un menù apposito, di tutte queste operazioni si occuperà la classe **POI**.

Il metodo ‘viewlist()’, chiamato successivamente a un click di selezione nella barra del menù, manderà in esecuzione a sua volta uno tra viewlistofservices() o viewlistofdepartments(), il metodo invocato poi chiamerà direttamente la classe **Controller** che attraverso il metodo index() mostrerà il giusto elenco di dati richiesti. A questo punto l’utente potrà muoversi nei vari elenchi per selezionare i POI desiderati o i dati su di questi attraverso metodi come infopoi(), infobuilding(), infophoto()… **Menù** è una classe logica formata dall’aggregazione di tutte le classi che mettono a disposizione operazioni di visualizzazione. Ogni metodo all’interno della classe **Menù**, e delle altre che la formano, invocherà nel Controller una funzione per listare qualcosa (index()) oppure una per avere informazioni dettagliate su qualcosa (info()). Per quanto riguarda i metodi di manipolazione degli oggetti (come delete(), create(), update()) saranno accedibili solo dagli amministratori.

A seconda delle procedure chiamate il **Controller** si occuperà di mostrare i riferimenti nel menù principale o direttamente localizzati nella mappa.

Attraverso l’interazione tra la libreria **Voix**.**js** e il Controller potranno essere impartiti comandi vocali pre-settati al sistema e all’utente per utilizzarli basterà premere un bottone che chiamerà il metodo register().

La classe controller farà da intermediario tra GUI e **Database** e verrà interrogata ogniqualvolta ci sarà bisogno di un’esecuzione.

Per finire il **Database** interagirà con le altre classi lavorando allo scopo di memorizzare i dati e di renderli disponibili nel momento del bisogno.

G. Explain how the FRs and the NFRs are satisfied by design

**NFRs**

Supportability e Maintanability

Per garantire la supportabilità del codice utilizzeremo I linguaggi già specificati che assicureranno la corretta eseguibilità su una grande varietà di sistemi.

Inoltre la specificità di ogni linguaggio e i vincoli architetturali impartiti da Laravel ci faciliteranno il compito di dividere il codice in moduli aventi ognuno una propria logica e di creare interfacce per determinati compiti che andremo a richiamare al momento opportuno*.*

Usabilità

Garantiremo la navigabilità del sito organizzando una homepage minimalista e il più intuitiva possibile, ci impegneremo allo scopo di rendere I comandi e l'interfaccia grafica semplici per quanto ci sarà possibile. A questo scopo per la GUI ci ispireremo ad altri siti web affermati in modo da renderla familiare.

Capacity e Scalability

Gestiremo il database in modo da rendere il più efficienti possibili le transizioni e le queries, inoltre per i periodi di grande attività terremo conto di poter aggiungere un altro database di supporto in modo da incrementare le performance.

**FRs**

Nella parte logica del sistema dovrà esserci un componente che si occuperà di eseguire queries sul database per poter filtrare correttamente I POI e visualizzarli nella mappa, realizzeremo questa funzione attraverso i vari controller implementati nella sezione apposita.

Il sistema dovrà fornire un metodo per controllare l’accesso da parte degli utenti effettuando un controllo e successivamente riportandoli nella homepage del sito. Realizzeremo del codice lato client per verificare che verranno immesse tutte le informazioni importanti e sarà LoginController a controllare e validare i dati inseriti nel form per il login. Verrà chiamata una query che controllerà se i dati inseriti esistono all'interno del database, visualizzando poi la giusta vista all'interno del sistema.

Per quanto riguarda la visualizzazione di una mappa forniremo il servizio importandola attraverso Google Maps che ci permetterà di incorporarla e utilizzarla in modo attinente ai nostri scopi. Per rispettare Il copyright inseriremo uno script che ci fornirà Google nel nostro codice. Nel complesso attraverso l'interazione tra codice PHP e MySQL andremo a organizzare le icone da visualizzare nella mappa secondo i vari elementi da filtrare.

H. Effort Recording

H.1 PERT

Immagine che contiene screenshot

Descrizione generata automaticamenteImmagine che contiene screenshot

Descrizione generata automaticamente

H.2 LOGGING

Immagine che contiene screenshot

Descrizione generata automaticamente

**Categorization**  
When logging the time spent on the project, please create different sub- categories.Specifically, it is important to clearly distinguish between two main categories: the time spent for “**learning**” (the modeling languages, the tools, etc.) from the time needed for “**doing**” (creating the models, taking the decisions, …). Learning tasks are in fact costs to be paid only once, while doing costs are those that will be repeated through the project.

*For each category, please define sub-categories. Examples follow. You may add other sub-categories you find useful.*

|  |  |
| --- | --- |
| ***Learning***   * ***Requirements Engineering*** * ***Non functional Requirements*** * ***Use Case Diagrams*** * ***Tool study*** | ***Doing:***   * ***Requirements discovery*** * ***Requirements Modeling (UC diagrams)*** |

***Summary Statistics****Based on the attributes defined above, calculate the summary statistics of the time spent for “learning”, the time spent for “doing”, and the total time.*

***Note: this Deliverable report shall document only the Summary Statistics for the different deliverables (D1, D2, and Final). Detailed information shall be reported in the Excel file.***

***COPY HERE (computed from the spreadsheet): i) the total number of hours spent by the group (that is, hours per task X number of people working on that task), ii) the time spent for LEARNING and for DOING***

Appendix. Prototype   
*<Provide a brief report on your prototype, and especially: information on what you have implemented, how the implementation covers the FR and NFR, how the prototypes demonstrates your project correctness with respect to the FR and NFR. You may add some screenshots to describe what required above. Be ready to show your prototype during the oral examination>*

Di seguito riportiamo uno screenshot riguardante la pagina home visualizzata dopo il login di un generico utente:

Immagine che contiene screenshot

Descrizione generata automaticamente

Tutto è stato implementato secondo le decisioni prese precedentemente, in particolare la struttura del menù è la stessa descritta nelle “design decisions”.

Come si può vedere in foto la mappa è stata importata ma non ci è permesso interagirci perchè la piattaforma di Google Maps richiede di registrare i nostri dati di fatturazione essendo il servizio a pagamento.

Da parte nostra abbiamo integrato la libreria “Map JavaScript API” e inserito la chiave fornitaci, ma nella documentazione viene subito specificato l’avviso per i nuovi utenti:

Immagine che contiene screenshot

Descrizione generata automaticamente

Inoltre nel documento (<https://developers.google.com/maps/documentation/javascript/mysql-to-maps?hl=it>) è scritto che nel database a cui collegare le API debba esserci una tabella ‘Markers’ con determinati attributi, andando a vedere nel nostro progetto Laravel nella cartella Database -> Migrations -> create\_markers\_table si può notare l’esatta implementazione di tutto ciò. Il passo successivo sarebbe stato esportare i dati della tabella dal database SQL in un formato XML, usando istruzioni PHP. Una volta fatto questo la mappa avrebbe potuto utilizzare il file XML per recuperare i dati del marker attraverso chiamate JavaScript asincrone e visualizzarli correttamente.

Per le restrizioni già citate non è stato possibile collegare la mappa con il nostro database ma è comunque tutto implementato per mettere a disposizione questa funzionalità.