

GardenCare

Applicazioni e Servizi Web

Asia Lucchi - 0000980152 (asia.lucchi@studio.unibo.it)
Andrea Negri - 0000990221 (andrea.negri4@studio.unibo.it)

20 Giugno 2021

Indice

1	Introduzione	2
2	Requisiti e funzionalità	3
3	Design	4
3.1	Target Users Analysis	4
3.1.1	Personas	4
3.1.2	Scenari d'uso	4
3.2	Interfacce utente	7
3.2.1	Mappa	7
3.2.2	Informazioni del giardino	9
3.2.3	Area manutentore	12
3.3	Architettura del sistema	16
3.3.1	Componenti Base	16
3.3.2	Componenti Composti	17
3.3.3	Schema componenti	19
3.4	Database	20
4	Tecnologie	22
5	Test	25
6	Deployment	26
7	Conclusioni	27

Capitolo 1

Introduzione

L'idea da cui parte il progetto è quella di realizzare una piattaforma per la gestione smart dei giardini comunali, così da poterli controllare in maniera centralizzata (e quindi comoda) e poter agire tempestivamente dove necessario.

La piattaforma realizzata è stata chiamata **GardenCare**, ed è una SPA¹ che consente agli utenti manutentori (d'ora in avanti utente e manutentore verrano utilizzati come sinonimi) di controllare lo stato dei vari giardini tramiti degli appositi sensori (di temperatura e umidità ad esempio), e di gestire le manutenzioni necessarie (programmarle, eliminarle e segnarle come eseguite).

L'applicativo si compone di una mappa dalla quale è possibile avere una visione d'insieme dello stato giardini e dalle pagine di amministrazione delle manutenzioni.

¹Single Page Application

Capitolo 2

Requisiti e funzionalità

Le funzionalità offerte dalla piattaforma sono:

- Visualizzazione di una mappa con i giardini gestiti;
- Visualizzazione delle informazioni relative ad ogni giardino tra cui:
 - Informazioni del giardino;
 - Sensori installati nel giardino e il loro stato;
 - Le previsioni del tempo nella località del giardino;
 - Le manutenzioni previste nel giardino.
- Area riservata agli utenti manutentori in cui poter gestire le manutenzioni per i vari giardini. La gestione prevede la possibilità di creare una manutenzione, eliminarla o segnarla come eseguita
- Possibilità per un utente di registrarsi alla piattaforma

Capitolo 3

Design

In questo capitolo verrà spiegato come si è arrivati alla realizzazione dell'applicativo e cosa si è realizzato, partendo dallo studio degli utenti con i relativi casi d'uso, confrontando poi i mockup con le effettive interfacce utente, elencando i componenti realizzati e utilizzati e infine mostrando lo schema della base di dati implementata.

3.1 Target Users Analysis

3.1.1 Personas

Gli utenti della piattaforma saranno tutti giardinieri di professione e non amatoriali; il sistema infatti non è pensato per essere utilizzato da chiunque ma da persone che hanno esperienza nel campo del giardinaggio (i sensori utilizzati misurano caratteristiche che hanno valore per chi sa di cosa si sta parlando, non per tutti).

Ivan

Ivan è un uomo di 53 anni e lavora come giardiniere da più di 30 anni; nessuno conosce i giardini comunali come lui e a loro dedica ogni più piccola attenzione. Ha iniziato il suo percorso lavorativo da ragazzino nel settore alberghiero, e l'ha proseguito per qualche anno anche dopo la maturità, ma vista la pesantezza del lavoro ha deciso di lasciare tutto a 21 anni e provare qualcosa di nuovo. È così che ha scoperto il giardinaggio, e ad esso ha dedicato il resto della sua vita.

Menelao

Menelao è un ragazzo di 28 anni, e lavora per un'azienda di giardinaggio che si occupa anche della manutenzione dei giardini comunali. È nato e cresciuto a stretto contatto con la natura, in un paese di collina, e da piccolo, insieme al nonno, si prendeva cura dell'orto, del giardino e dei fiori di casa; è così che è nata in lui la passione per il giardinaggio. È inoltre un appassionato di NFL che però riesce a seguire solo in streaming di notte visto il diverso fuso orario.

3.1.2 Scenari d'uso

Vediamo ora quali sono gli scenari d'uso dell'applicativo, utilizzando le personas viste precedentemente e un breve racconto che ha loro come protagonisti.

Scenario d'uso di Ivan

Racconto su Ivan:

Ivan arriva al lavoro lunedì mattina e si siede alla sua postazione per vedere cosa lo aspetta oggi; non vede l'ora di recarsi al primo giardino e iniziare a lavorare veramente, non sopporta dover stare seduto davanti a un monitor a non far nulla. Apre la pagina di GardenCare e vede sulla mappa che ci sono varie icone rosse in diversi giardini. "Il caldo inizia a farsi sentire" pensa tra sè e sè. Seleziona il primo giardino e vede che la temperatura e l'umidità sono molto alte; nota anche che tra le cose da fare è già segnato un intervento previsto per il pomeriggio. "Bene, a questo ci pensa Menelao" bofonchia debolmente. Seleziona un altro giardino e vede che gli stessi parametri sono fuori limite e non è segnata alcuna manutenzione; inoltre sono molto numerosi per cui non ci vorrà poco a sistemare il tutto. Felice di aver trovato la sua metà, almeno per la mattina, si alza e segna sulla lavagna il luogo in cui si troverà durante la mattinata così da farlo sapere agli altri; poi si dirige verso il suo camioncino e incontra Menelao all'uscita...

A Ivan non piace molto la tecnologia e infatti, come si legge nel racconto, utilizza GardenCare solo per vedere cosa è mostrato sulla mappa ma quando decide cosa fare non programma alcuna manutenzione tramite il sito ma la scrive a mano.

I goal che porta a compimento quindi sono:

- Visualizzare i giardini con il loro stato sulla mappa;
- Visualizzare le informazioni di un singolo giardino;

Nel primo caso, **l'unico task** che compie è quello di aprire la pagina principale di GardenCare dove viene **visualizzata la mappa**. A colpo d'occhio Ivan vede quali sono i giardini che sono messi peggio degli altri e ad essi dà la priorità.

Nel secondo caso, invece, Ivan controlla lo stato dei sensori nei giardini, quindi **i task eseguiti sono visualizzare la mappa e visualizzare le informazioni di un giardino**. In base a quello che vede prende le sue decisioni smettendo però di interagire con GardenCare.

Scenario d'uso di Menelao

Racconto su Menelao:

Menelao sta entrando nella sede dell'azienda quando ecco che vede uscire Ivan. I due si salutano e scambiano quattro chiacchere; alla fine Menelao gli chiede:

"Dove vai stamattina?"

"Ho visto che il parco in centro ha dei problemi con le alte temperature e l'umidità, mi sono segnato lì tutta la mattina, mi sa che avrò da fare per un bel po'"

"Capito, buon lavoro allora, a più tardi!"

E con un gesto i 2 si salutano. Menelao entra in ufficio, si siede a apre GardenCare. Vede che ci sono numerosi giardini che segnalano problemi, per cui decide di prendersi 5-10 minuti per organizzare bene la sua giornata. Effettua il login sulla piattaforma ed entra nella sua area privata dove gestisce il proprio calendario. Per prima cosa segnala come eseguita una manutenzione del venerdì prima, cosa che si era dimenticato di fare, e poi guarda quali sono gli impegni della settimana. Il pomeriggio deve andare a mettere a posto un parco in periferia che tra l'altro ha dei valori segnalati fuori range; la mattina invece è libero. Inizia a guardare i giardini e senza rendersene conto seleziona quello in centro: i sensori segnalano dei valori molto alti e di manutenzioni programmate non ce ne sono, per cui inizia a creare una nuova manutenzione ma poi si ricorda della chiaccherata avuta con Ivan e che andava lui in centro. Guarda allora sulla lavagna dell'ufficio e vede che lì Ivan si è segnato. "Ecco, come sempre si segna sulla lavagna, ma di farlo nella piattaforma non sia mai... Ahh lui e la fobia per la tecnologia" pensa un po' divertito, un po' scocciato dal comportamento del collega. Per evitare altri frantendimenti, segna lui una manutenzione per la mattina nel parco in centro, cosicché gli altri la vedano; poi prosegue con la ricerca di un giardino da andare a salvare!

Menelao, a differenza di Ivan, trova utile e semplice gestire il proprio lavoro tramite GardenCare e quindi utilizza appieno le sue funzionalità. I suoi goal sono identici a quelli di Ivan (e quindi anche i task) ma a questi aggiunge anche:

- Utilizzo del calendario nell'area riservata
- Possibilità di segnalare come eseguita una manutenzione
- Possibilità di creare una nuova manutenzione

Nel primo caso, **il task** che deve compiere è il **login**. Per far ciò però, Menelao si sarà prima **registrato sulla piattaforma**.

Nel secondo caso, **i task** da compiere sono stati **effettuare il login, selezionare una manutenzione e segnarla come completata**.

Infine, nell'ultimo caso, **i task** da compiere sono stati **effettuare il login e creare una nuova manutenzione**.

3.2 Interfacce utente

Viste le personas e gli scenari d'uso, i goal e i task previsti per realizzare le funzionalità volute vengono ora utilizzati per creare le interfacce utente. Ogni funzionalità prevista verrà ora analizzata, mostrando come è stata pensata e come è stata realizzata.

N.B. Per ogni interfaccia vengono specificati, alla fine, i componenti utilizzati. Ancora non si è parlato di cosa siano i componenti, verrà fatto nella sezione successiva, ma si è preferito illustrare prima dove vengono utilizzati e poi spiegare cosa sono e cosa fanno. In questo modo il lettore ha già un'idea di quale sia la funzione dei componenti, avendo viste le interfacce che sono state il punto di partenza per la realizzazione dell'applicativo.

3.2.1 Mappa

La mappa rappresenta la pagina principale dell'applicativo; l'idea da cui si è partiti è rappresentata dal mockup mostrato in Figura 3.1.

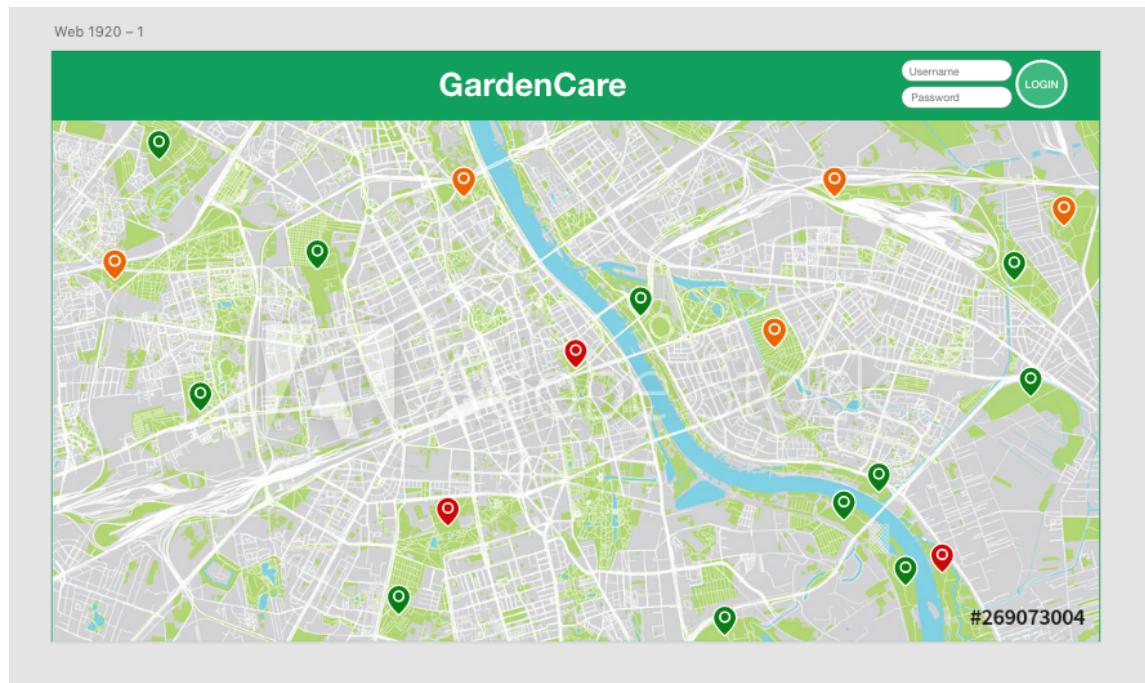


Figura 3.1: Mockup home page

Lasciando momentaneamente da parte la barra superiore con titolo e campi per il login, si può vedere come la mappa sia formata essenzialmente da 2 elementi:

- Mappa geografica: una normale mappa come quelle mostrate, ad esempio, nei navigatori, con cui quindi l'utente dovrebbe avere già dimestichezza;
- Icône: le icône mostrano all'utente quali sono i giardini gestiti dall'applicativo.

La diversa colorazione delle icône indica qual è lo stato del giardino in base allo stato dei sensori presenti:

- Verde: nessun sensore rileva valori pericolosi;

- Giallo: 1 sensore rileva valori pericolosi;
- Rosso: 2 o più sensori rilevano valori pericolosi

Partendo dal mockup è stata realizzata quindi la pagina principale di GardenCare, che viene mostrata in Figura 3.2.

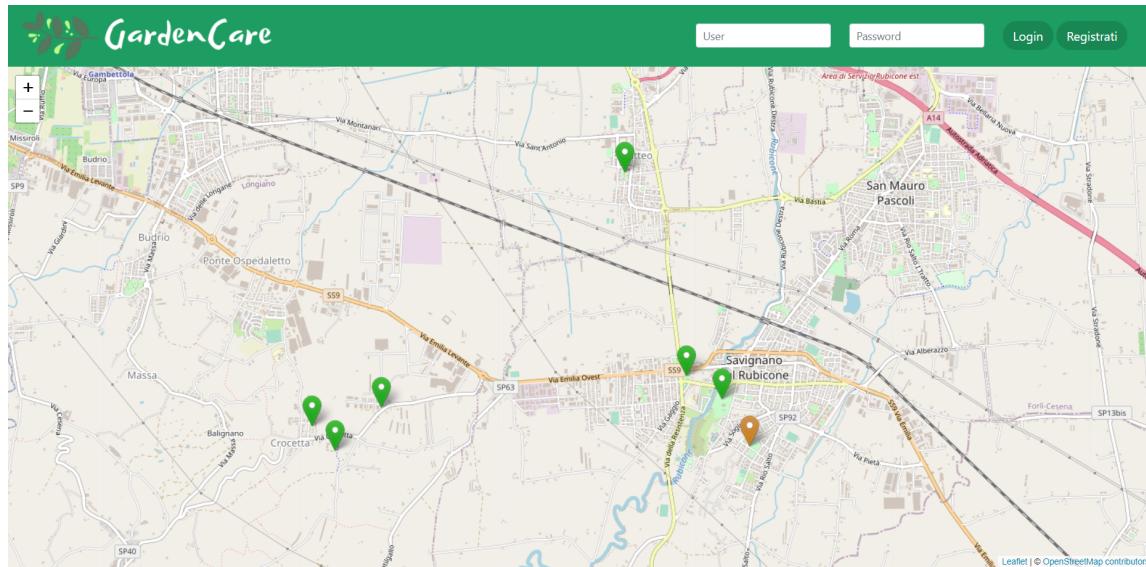


Figura 3.2: Home page

Da questa interfaccia l'utente può svolgere qualsiasi azione:

- Cliccando su un icona visualizza le informazioni di un giardino;
- Compilando i campi per il login (con delle credenziali corrette) può effettuare il login per pianificare nuove manutenzioni;
- Può registrarsi alla piattaforma tramite l'apposito pulsante.

I componenti utilizzati sono: Navbar e Home.

Passiamo quindi ora all'analisi di queste altre funzionalità.

3.2.2 Informazioni del giardino

Per ogni giardino mostrato sulla mappa è possibile visualizzare un insieme di informazioni che aiutano gli utenti a prendere decisioni sulle eventuali manutenzioni da effettuare. Prima di vedere quali sono queste informazioni viene mostrato il mockup di partenza (Figura 3.3).

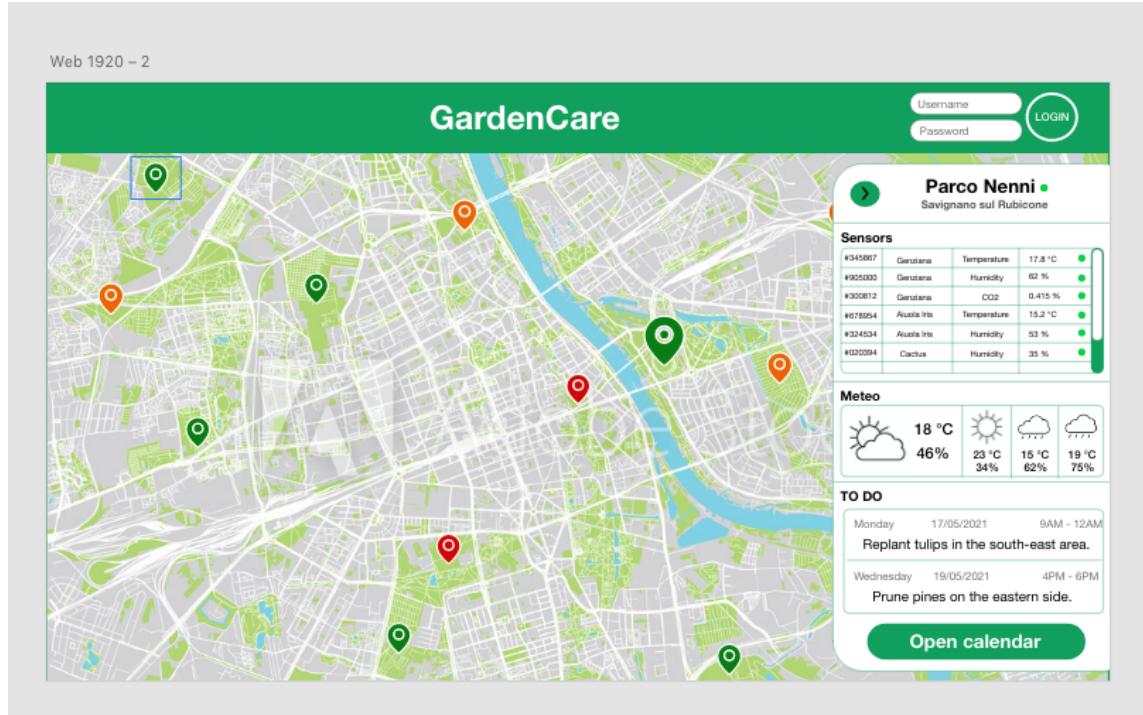


Figura 3.3: Mockup dettaglio giardino

Le informazioni mostrate sono (dall'alto verso il basso):

- Nome del giardino e località in cui si trova;
- Lista dei sensori presenti nel parco. Per ogni pianta possono essere presenti più sensori, ognuno dei quali misura caratteristiche diverse. In base al valore misurato viene segnalato lo stato del sensore: verde se i valori vanno bene, rossi altrimenti.
- Il meteo previsto nella località del giardino, che può aiutare i manutentori a decidere quando effettuare determinate manutenzioni;
- La lista delle cose da fare (to do) che di fatto sono le manutenzioni programmate nel giardino;
- Infine viene data la possibilità di aprire il calendario relativo al giardino che mostra le manutenzioni programmate e consente di crearne di nuove (mockup in Figura 3.4)



Figura 3.4: Mockup calendario manutenzioni giardino

La realizzazione di quanto descritto sopra si può vedere in Figura 3.5 (Dettaglio di un giardino) e in Figura 3.6 (Calendario delle manutenzioni di un giardino).

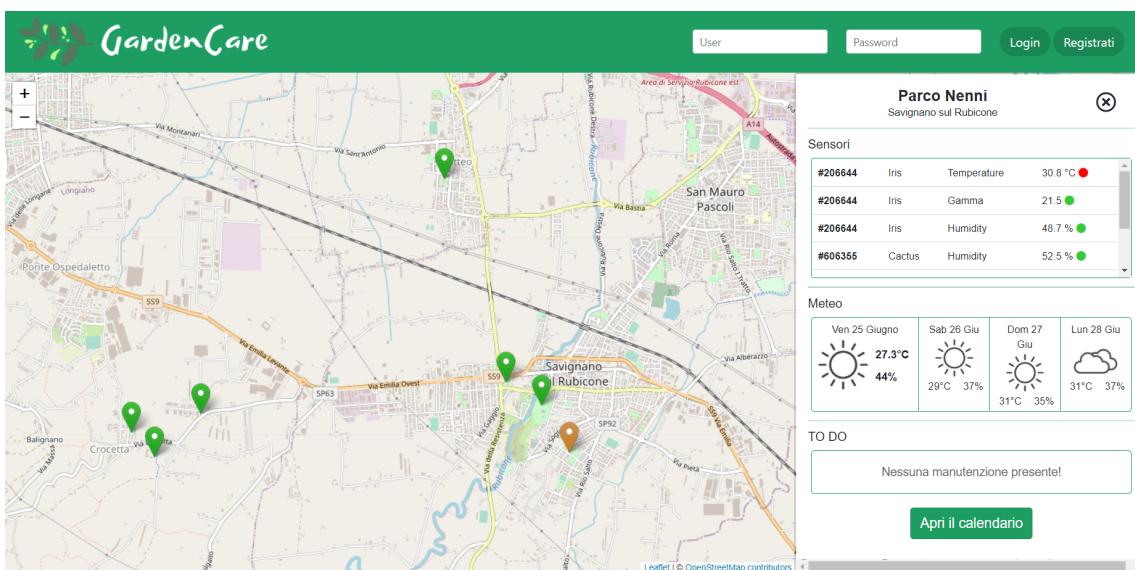


Figura 3.5: Dettaglio giardino

The screenshot shows two main sections. On the left, a calendar grid for June 21 to June 27, 2023, with specific times (07, 08, 09, 10, 11, 12, 13, 14, 15, 16, 17, 18) listed vertically. The days are labeled: Lunedì 21 Giu, Martedì 22 Giu, Mercoledì 23 Giu, Giovedì 24 Giu, Venerdì 25 Giu, Sabato 26 Giu, Domenica 27 Giu. A small green icon with a plus sign is visible in the cell for Friday at 09:00. On the right, a detailed view for 'Parco Nenni' in Savignano sul Rubicone. It includes a table for 'Sensori' showing data for Iris and Cactus across various parameters like Temperature and Humidity, with status indicators (red or green). Below this is a 'Meteo' section showing weather forecasts for the next four days (Ven 25 Giugno to Lun 28 Giu) with icons for sun, clouds, and rain, along with temperature and humidity percentages.

Figura 3.6: Calendario manutenzioni giardino

A differenza di quanto visto nel mockup, il calendario è stato realizzato in una pagina a parte, accessibile solo dai manutentori (lo si può vedere dalla barra superiore, in cui non sono più presenti i campi per il login) e dalla quale si possono programmare le manutenzioni sul giardino selezionato.

I componenti utilizzati sono: Informazioni del giardino per il Dettaglio giardino, e Pagina del giardino per il calendario delle manutenzioni del giardino. In entrambe le interfacce viene utilizzata anche la Navbar.

3.2.3 Area manutentore

Quest'area serve agli utenti per gestire le proprie manutenzioni. Essendo un'area riservata è accessibile solo agli utenti registrati che devono quindi fornire user e password per accedervi.

In quest'area ogni utente visualizzerà il proprio calendario, dove sono riportate le proprie manutenzione programmate e dove potrà aggiungerne di nuove o eliminare quelle vecchie. Verranno inoltre elencati i giardini i cui sensori segnalano dei valori pericolosi. In Figura 3.7 è mostrato il mockup di riferimento.

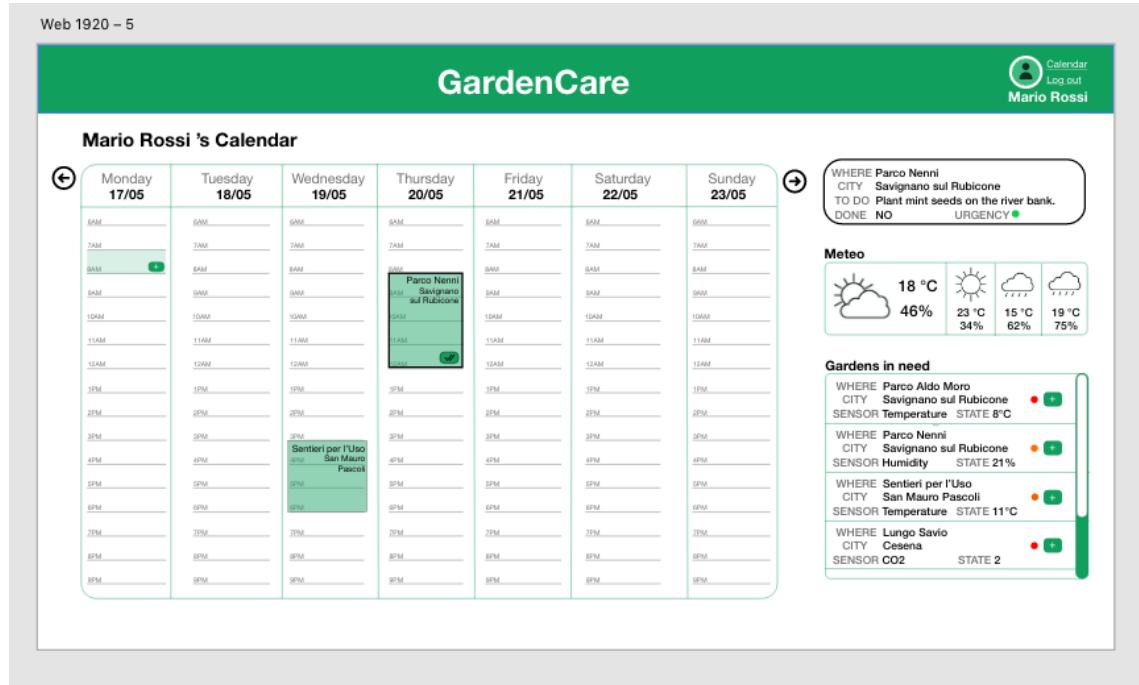


Figura 3.7: Mockup area manutentore

Cliccando su una manutenzione, l'utente potrà vederne i dettagli (in alto a destra), il meteo, e potrà segnarla come eseguita tramite l'apposito pulsante presente sul calendario (il tick). In basso a destra vengono invece mostrato i giardini che hanno bisogno di una manutenzione; per aggiungere una manutenzione l'utente può cliccare sul pulsante '+' presente di fianco ai giardini oppure selezionare una data e un orario desiderati sul calendario e cliccare sul '+' che verrà mostrato (in figura, è mostrato in alto a sinistra nel calendario).

In Figura 3.8 viene mostrata la realizzazione di quanto descritto sopra.

The screenshot shows the GardenCare application interface. At the top, there is a navigation bar with a logo, the text "GardenCare", and buttons for "Manutenzioni" and "Logout". Below the navigation bar is a "Calendario di Andrea Negri" (Andrea Negri's Calendar) for June 26 to July 4. The calendar grid shows various scheduled tasks. To the right of the calendar are sections for "Manutenzione selezionata" (Selected Maintenance), "Meteo" (Weather), and "Giardini" (Gardens). The "Manutenzione selezionata" section displays a task for "Parco Villa Conte Marchesi di Bagno" with details: CITY Savignano sul Rubicone, TO DO Prova di visualizzazione, DONE NO, and GARDEN STATE (green dot). The "Meteo" section shows a 5-day forecast from June 25 to July 1, with temperatures and humidity levels. The "Giardini" section lists four gardens with their locations, sensors, and current status (e.g., IS 81%, IS 30.5 °C).

Figura 3.8: Area manutentore

È inoltre presente il tasto elimina in alto a destra, tramite cui l'utente può eliminare la manutenzione selezionata dopo un'ulteriore conferma, come si vede in Figura 3.9.

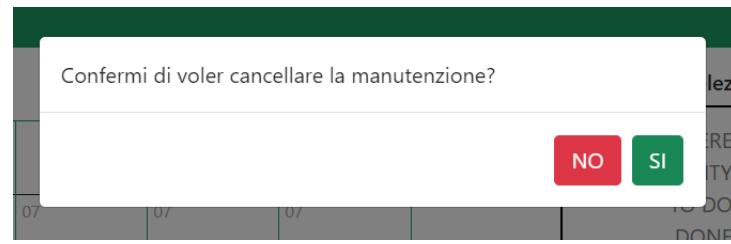


Figura 3.9: Conferma per la cancellazione di una manutenzione

In Figura 3.10 è invece mostrata la finestra che consente di aggiungere una nuova manutenzione.

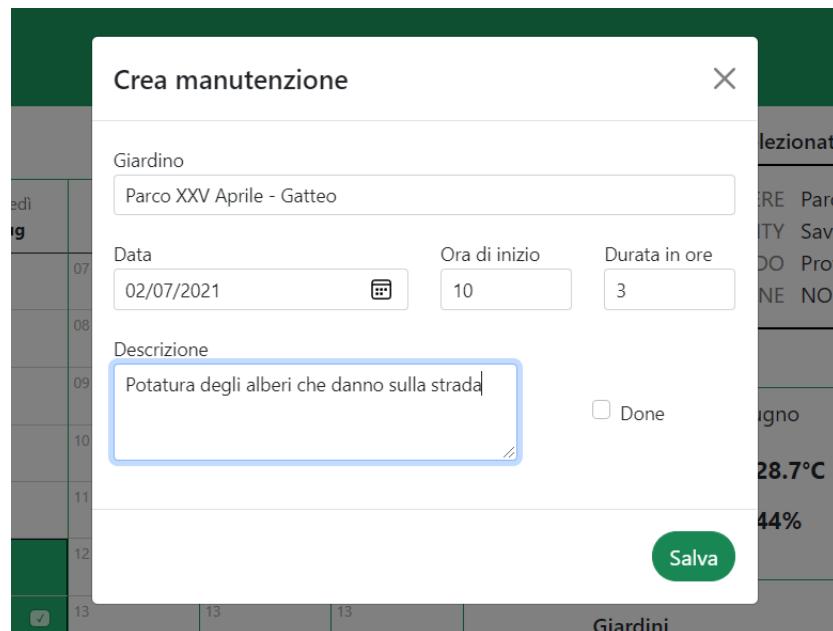


Figura 3.10: Creazione manutenzione

Una volta salvata la manutenzione apparirà nel calendario del manutentore. I componenti utilizzati sono: Pagina del manutentore e Navbar.

Registrazione nuovo manutentore

Per poter accedere all'area manutentori, un utente può registrarsi tramite l'apposita pagina, mostrata in Figura 3.11 .

The screenshot shows a registration form titled "Iscriviti" (Sign Up) in a light gray box. At the top right of the box is a small white button labeled "User". The form is divided into two main sections: "Informazioni personali" (Personal Information) and "Credenziali" (Credentials).
Informazioni personali:
- Nome: Input field labeled "Nome".
- Cognome: Input field labeled "Cognome".
- Codice Fiscale: Input field labeled "Codice Fiscale".
- Indirizzo: Input field labeled "Indirizzo".
- Telefono: Input field labeled "Telefono".
Credenziali:
- Nome Utente: Input field labeled "Nome Utente".
- Password: Input field labeled "Password".
- Conferma Password: Input field labeled "Conferma password".
At the bottom right of the form is a green rounded rectangular button labeled "Conferma" (Confirm).

Figura 3.11: Form di registrazione

I componenti utilizzati sono: Form di registrazione e Navbar.

3.3 Architettura del sistema

I componenti (si parla di Vue component ma per brevità verranno chiamati semplicemente componenti d'ora in avanti) che verranno ora mostrati vengono utilizzati da 2 processi, uno principale e uno di supporto:

- Processo principale: è il processo che di fatto gestisce tutte le funzionalità previste dall'applicativo; al suo interno vengono dichiarati e utilizzati tutti i componenti che verranno in seguito analizzati (nello specifico, all'interno di questo progetto la pagina che gestisce tutto è index.html);
- Processo di supporto: è un processo secondario che ha il compito di aggiornare i valori dei sensori periodicamente.

I componenti si dividono in:

- **componenti base:** non utilizzano altri componenti per fornire le proprie funzionalità;
- **componenti composti:** utilizzano altri componenti, base e non, più del proprio codice per fornire le proprie funzionalità.

Per ogni componente verrà ora spiegata la sua utilità.

3.3.1 Componenti Base

Meteo

Al suo interno vengono mostrate le previsioni del tempo in base alla latitudine e alla longitudine del giardino selezionato. Le previsioni sono relative ai successivi 4 giorni (incluso il giorno attuale). Per recuperare le previsioni viene utilizzata un API esterna che verrà illustrata nel capitolo 4.

Sensori

Mostra i sensori relativi ad un giardino. Per ogni sensore vengono mostrati il numero identificativo, la pianta/albero su cui si trova, la caratteristica misurata (per esempio temperatura o umidità), il valore rilevato per la caratteristica, e un cerchietto colorato che indica lo stato del sensore in base al valore misurato (verde se il valore rientra in un range consentito, rosso altrimenti). Un sensore rosso indica che per quella determinata pianta/albero, quella determinata caratteristica è a dei livelli che potrebbero causare danni.

Todo

I todo (cose da fare) servono a elencare le prossime manutenzioni previste per un giardino in maniera più succinta rispetto a visualizzarle in un calendario. In questa maniera possono essere visualizzati per ogni giardino anche sulla mappa (il risultato verrà mostrato in seguito nell'apposita sezione del paragrafo 3.2).

Testata giardino

Mostra semplicemente il nome del giardino e la località in cui si trova.

Manutenzione

Mostra i dettagli relativi ad una manutenzione selezionata: giardino in cui dovrà essere eseguita, la località, una descrizione che indica le cose da fare, lo stato attuale del giardino (in base ai sensori), i sensori presenti e se è già stata completata oppure no.

Giardini bisognosi

È un elenco che viene mostrato al manutentore nella sua pagina privata, in cui sono elencati alcuni dei giardini gestiti ordinati in base al loro stato (per primi vengono mostrati quelli rossi, poi gialli ed eventualmente verdi) in maniera tale da poter tenere sotto controllo in quali sia più urgente una manutenzione.

Calendario

È un calendario settimanale in cui sono evidenziate le manutenzioni programmate e dal quale si possono creare nuove manutenzioni, segnare come eseguite le manutenzioni e selezionarle per visualizzarne i dettagli.

Form di registrazione

La form attraverso la quale gli utenti possono registrarsi alla piattaforma.

Navbar

La barra di navigazione attraverso cui l'utente può muoversi nel sito; il suo aspetto cambia a seconda che l'utente abbia effettuato il login oppure no. Se non è loggato, la barra consente di loggarsi o registrarsi al sito. Se l'utente è loggato, consente di accedere alla pagina del manutentore o di effettuare il logout. In entrambe le situazioni viene mostrato il logo di GardenCare attraverso il quale si può tornare alla pagina principale. È l'unico componente che viene sempre mostrato perché indipendente dagli altri.

3.3.2 Componenti Composti

Informazioni giardino

Le informazioni di un giardino vengono mostrate quando se ne seleziona uno. Di fatto vengono visualizzati i seguenti componenti base: Testata giardino, Sensori, Meteo, Todo. È inoltre possibile navigare fino alla Pagina del giardino, ma solo se si ha effettuato il login.

Pagina del giardino

La pagina del giardino mostra in maniera più completa le sue informazioni; oltre ad utilizzare i componenti visti sopra, utilizza anche il calendario tramite il quale è possibile programmare una nuova manutenzione.

Pagina del manutentore

La pagina del manutentore consente, ad un utente che ha eseguito il login, di visualizzare il proprio calendario con le manutenzioni programmate nei vari giardini e di programmare nuove manutenzioni. I componenti utilizzati sono: calendario (in questo caso vengono caricate tutte le manutenzioni dell'utente relative quindi a più giardini), manutenzione, meteo (relativo alla manutenzione selezionata) e infine i giardini bisognosi.

Home

La pagina principale mostra la mappa (che non è un componente; la tecnologia utilizzata verrà spiegata nel capitolo 4) e utilizza la Pagina del giardino che viene mostrata quando viene selezionato un giardino.

Router

Insieme ai vari componenti di visualizzazione elencati sopra, viene utilizzato anche un vue-router che consente, tramite una router-view, di visualizzare i componenti desiderati (che insieme compongono le varie interfacce utente).

3.3.3 Schema componenti

In Figura 3.12 viene mostrato uno schema riassuntivo dell'architettura dei componenti.

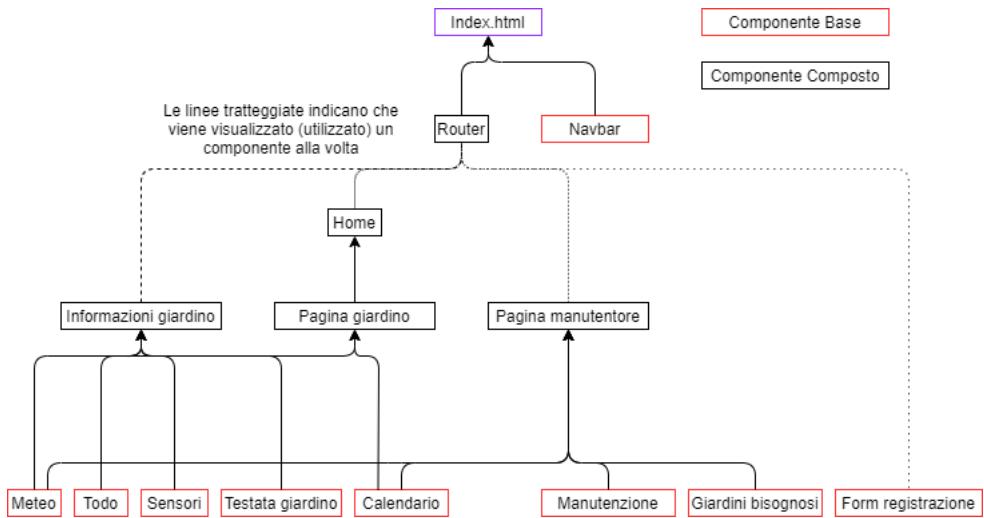


Figura 3.12: Architettura dei componenti

3.4 Database

Viene ora illustrato il database utilizzato dall'applicazione; in Figura 3.13 è mostrato il suo schema.

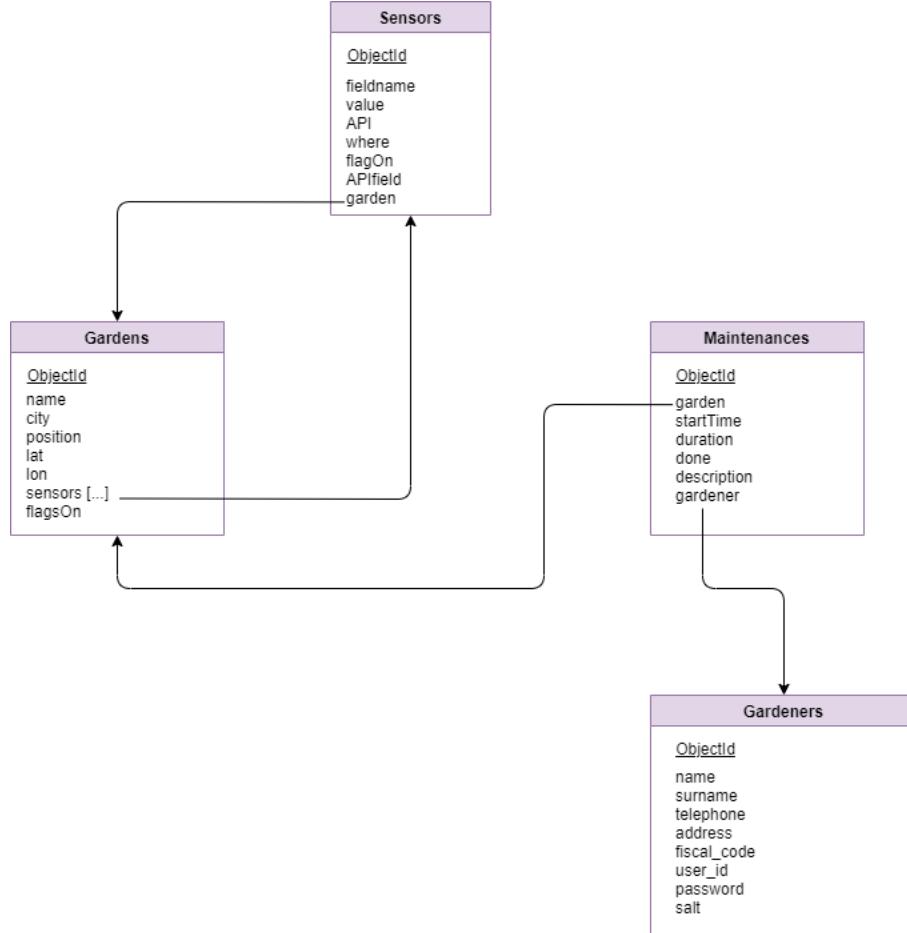


Figura 3.13: Schema del DB

Le 4 collezioni utilizzate sono:

- **Giardini**(Gardens): ogni documento è uno dei giardini gestiti dal sistema. I suoi campi sono:
 - name: nome del giardino
 - city: località in cui si trova il giardino
 - lat: latitudine del giardino
 - lon: longitudine del giardino
 - sensors[]: elenco dei sensori presenti nel giardino
 - flagsOn: numero dei sensori che registrano valori fuori range
- **Manutenzioni**(Maintenances): ogni documento è una delle manutenzioni programmate. I suoi campi sono:
 - garden: riferimento al giardino in cui dovrà essere svolta

- startTime: ora d'inizio
- duration: durata (in ore) prevista
- done: flag che indica se è già stata eseguita oppure no
- description: descrizione che indica cosa dovrà essere svolto
- gardener: riferimento al giardiniere che la deve svolgere

- **Giardinieri**(Gardeners): corrisponde all'utente manutentore. I suoi campi sono:

- name: nome
- surname: cognome
- telephone: numero di telefono
- address: indirizzo
- fiscal_code: codice fiscale
- user_id: username che verrà utilizzato per accedere all'applicativo. Deve essere univoco all'interno della collezione
- password: hash della password che l'utente sceglie al momento della registrazione. Viene utilizzato durante il login
- salt: valore di sale che viene utilizzato per creare l'hash della password al momento della registrazione

- **Sensori**(Sensors): ogni documento rappresenta uno dei sensori utilizzati in uno dei giardini. I suoi campi sono:

- fieldname: caratteristica misurata (temperatura, umidità, ...)
- value: valore registrato
- API: id del sensore nell'API thingspeak
- where: pianta/albero in cui è inserito il sensore
- flagOn: flag che indica se l'ultima registrazione del sensore era fuori range oppure no
- APIfield: nome del campo riguardante questo tipo di misura nell'API thingspeak per questo specifico sensore
- garden: riferimento al giardino in cui si trova il sensore

Si è scelto di registrare i valori dei sensori nel database e non reperirli su richiesta dell'utente perchè sono necessari anche per valutare i colori delle icone dei vari giardini presenti sulla mappa. A differenza del meteo alla cui API si fa saltuariamente una sola richiesta quando si apre la pagina del giardino, nel caso dei sensori si sarebbero dovute fare innumerevoli richieste all'API semplicemente per visualizzare correttamente la schermata home. Ci è sembrata una soluzione più performante e robusta fare una sola richiesta ad un database sul sever e non affidarsi ciecamente alla disponibilità dell'API thingspeak dato che la visualizzazione dalla mappa dei giardini è una delle funzionalità core dell'applicativo realizzato.

Capitolo 4

Tecnologie

Utilizzando uno stack MEAN, sono stati utilizzati:

- *mongoDB*: database non relazionale (modello documentale) in cui sono state salvate le collezioni viste precedentemente;
- *express*: framework js lato server, tramite il quale è stato possibile usare Mongoose, bcrypt, jsonwebtoken e childprocess;
- *Vue.js*: framework js lato client che ha consentito la realizzazione di una SPA formata, come visto precedentemente, da varie interfacce ottenute combinando tra loro i componenti. Queste interfacce sono tutte gestite lato client fino a che non effettuano richieste esplicite al server; in questa maniera il cambio di pagina avviene sul client e il server viene interpellato solo quando una di queste pagine ha bisogno di interagire in maniera diretta (dialogando con le API o col DB per esempio) col server;
- *Node.js*: ambiente per l'esecuzione delle applicazioni lato server; express gira all'interno di Node.

Uno dei motivi per cui è stato scelto Vue e non Angular né React è che era in parte già conosciuto all'interno del team di sviluppo; inoltre ha la migliore curva di apprendimento e questo ha consentito di iniziare a lavorare abbastanza speditamente fin da subito, grazie alla sua sintassi diretta e semplice da usare e ricordare fin dai primi utilizzi.

Oltre alle tecnologie previste dallo stack MEAN ne sono state utilizzate altre che ora verranno illustrate:

- Gestione formattazione:
 - *Bootstrap*: framework per la gestione del CSS, utilizzato in tutti i componenti
 - *Fontawesome*: sito che fornisce set di icone predefinite da poter utilizzare nel proprio progetto; è stato utilizzato nella creazione dei template html dei vari componenti
- Gestione database e richieste al server:
 - *Mongoose*: framework per la modellazione di oggetti mongodb su Node.js; fa da tramite tra express e mongoDB fornendo una sintassi più comoda per realizzare le query sulle collezioni

- *Axios*: client HTTP basato su promise per Node.js; consente di effettuare le richieste al server (get e post in questo progetto, ma consente anche di fare altri tipi di richieste, come put e delete ad esempio), di passare i vari parametri e gestire le risposte ricevute;
- Generazione e gestione della mappa:
 - *OpenStreetMap*: fornisce la mappa che viene visualizzata nella pagina principale;
 - *Leaflet*: framework js per la creazione di mappe interattive; viene utilizzato insieme a OpenStreetMap e consente di interagire con la mappa, dando la possibilità di selezionare i giardini presenti;
 - *Leaflet-sidebar*: framework js che viene utilizzato per la creazione della sidebar che viene visualizzata quando si seleziona un giardino sulla mappa;
- Gestione login e registrazione:
 - *bcrypt*: è una funzione di hashing per salvare le password in maniera sicura nel database; viene utilizzato al momento della registrazione di un utente quando si deve salvare la password nel db (viene salvato il suo hash con il relativo valore di sale), e al login quando bisogna confrontare la password ricevuta dall'utente con quella salvata;
 - *jsonwebtoken*: framework che implementa i JSON Web Tokens. Al momento del login il server rilascia un token al client contenente dei dati utili all'autenticazione; in questa maniera, durante le successive richieste, il client invierà anche il token ricevuto, e il server lo controllerà per verificare se è valido e quindi autorizzare il client. In questa maniera il server diventa stateless, non avendo bisogno di salvare le informazioni relative alle sessioni dei vari client collegati.
- Dati real-time per meteo e sensori:
 - *openweathermap*: API che permette di ottenere le informazioni riguardanti il meteo in una determinata posizione specificando latitudine e longitudine, è stata utilizzata per mostrare temperatura, percentuale di umidità e previsioni del tempo per ciascun giardino. Le richieste all'API sono effettuate direttamente quando l'utente apre la scheda dello specifico giardino, le informazioni non sono mantenute nel database.
 - *thingspeak*: API per raccogliere le misurazioni di sensori di ogni tipo e da ogni parte del mondo. Nel sistema è stata usata per ottenere nuovi dati per aggiornare le informazioni dei sensori nel database, i quali a fini di demo sono stati semplicemente accoppiati con sensori resi pubblici da utenti di thingspeak che fornissero tipi di misure coerenti con la ragione d'essere dell'applicativo (temperatura, umidità, TVOC, etc.). In uno scenario d'uso reale chiaramente ci sarebbero dei veri sensori nei vari giardini da registrare sulla piattaforma thingspeak e legare all'entità del giardino corrispondente sul database.
- Utilizzo di più processi:

- *childprocess*: Estensione di Node.js che permette di creare asincronamente dei processi figli. Nel sistema è stato usato per creare, tramite una **fork**, un processo sul server per l’aggiornamento dei sensori separato rispetto a quello che resta in ascolto di richieste da parte dei client. Il processo in questione esegue periodicamente (ogni 5 minuti) una funzione che richiede le informazioni più recenti per ogni sensore presente nel database.

Capitolo 5

Test

Finita la realizzazione, la piattaforma è stata fatta provare a degli utenti che non hanno seguito lo sviluppo (e che quindi non sono soggetti ad un bias cognitivo come quello sviluppato da noi programmati) e che non hanno alcuna esperienza di giardinaggio (per cui non possono attingere a conoscenze pregresse nel campo in questione per facilitare l'utilizzo della piattaforma).

Dai loro feedback si è potuto capire come alcuni aspetti dell'interfaccia non fossero chiari e non aiutassero l'utente a comprendere appieno come utilizzarla: ad esempio, per accedere alla pagina del manutentore, inizialmente il pulsante era chiamato "Board personale" e questo non risultava chiaro all'utente. È stato quindi sostituito con "Manutenzioni" e questo ha risolto il problema.

Inoltre, seguendo l'utente durante l'utilizzo della piattaforma, è stato possibile individuare alcuni errori di programmazione sfuggiti precedentemente: la registrazione andava a buon fine anche se erano presenti degli errori nell'inserimento dei dati per una mal gestione degli errori, il cambio manuale dell'URL della pagina portava a comportamenti indesiderati.

Capitolo 6

Deployment

In questo breve capitolo viene spiegato come scaricare e utilizzare la piattaforma Gardencare. I passi da compiere sono:

1. Scaricare il repo con l'applicazione dal seguente link Github: <https://github.com/aslu98/asw-smart-gardening>
2. Importare le collections presenti all'interno della cartella `./mongodb-collections` su mongodb attraverso il comando `mongoimport --collection gardens --db smart-gardening nomeCollezione.json`
3. Entrare all'interno della cartella `./code` con una console in cui è possibile utilizzare i comandi NPM e lanciare il comando `npm install`; aspettare che venga completata l'installazione dei package richiesti dall'applicazione
4. Rimanendo all'interno della cartella `./code`, sempre da console, lanciare il comando `node index.js`; aspettare il messaggio da parte del server che comunica di essere in ascolto sulla porta 3000
5. Aprire il proprio browser e digitare l'indirizzo `http://localhost:3000/`. Ora potete utilizzare GardenCare!

Capitolo 7

Conclusioni

Asia Lucchi

Grazie a questo progetto ho fatto una minima esperienza di utilizzo dello stack MEAN, decisamente più moderno e comodo da usare rispetto a LAMP, che avevo visto durante il progetto di Tecnologie Web. Ho anche compreso come combinando Vue.js e lo sfruttamento di API pubbliche si riescono a realizzare velocemente e in modo non troppo complesso degli applicativi molto interessanti ed user-friendly. Il progetto potrebbe avere varie vie per continuare a svilupparsi:

- innanzitutto si potrebbero installare realmente dei sensori da collegare ai giardini
- si potrebbe inserire un utente amministratore di più alto livello che ha la possibilità di aggiungere giardini e sensori
- si potrebbe rendere il sito web più accessibile da mobile, dato che questa sua versione iniziale è stata pensata principalmente per il web con solo qualche accorgimento sulle dimensioni per quanto riguarda la visualizzazione mobile
- si potrebbe differenziare il livello di warning dei sensori anche in base alla tipologia di pianta che stanno controllando (ad esempio una temperatura di 35 gradi non è drammatica per un cactus come lo è per un bonsai)

,

Andrea Negri

Questo progetto mi ha dato la possibilità di approfondire le mie conoscenze riguardanti l'utilizzo dello stack MEAN, di cui, fino ad ora, avevo utilizzato solo raramente mongoDB. È stato molto interessante, e anche soddisfacente, creare una SPA, cosa che non avevo mai fatto prima anche se già conoscevo superficialmente Vue.

Un aspetto importante, non tanto riguardante l'ambito dello Sviluppo Web, ma direi più che altro dello sviluppo e del lavoro in generale, che ho potuto imparare è l'importanza dell'organizzazione: per la prima volta, infatti, ho deciso di organizzare il lavoro quotidiano settimana per settimana, e questo ha portato a dei notevoli vantaggi e a un modo di lavorare migliore rispetto a quelli che avevo sperimentato fino a prima.