**Beecounter**

**Server di sviluppo**

Per poter utilizzare il sito/app in locale è necessario installare [NodeJS](https://nodejs.org/it/download/).  
Fatto ciò per installare le dipendenze è necessario spostarsi della cartella root del progetto e digitare *npm install*.  
Installate le dipendenze digitate *npm start* per far partire il server e *ng serve* per l'applicazione Angular4.  
Adesso è possibile recarsi a http://localhost/4200 per testare l'applicazione in locale.

**Sito dell'applicazione**

Il sito dove poter trovare l'applicazione è [Bee Counter](https://beecounter.herokuapp.com/).

**Installazione applicazione**

Per poter installare l'applicazione sul proprio dispositivo è necessario utilizzare [Chrome](https://www.google.it/chrome/browser/desktop/) e:

* Dispositivo desktop Aprire gli strumenti per sviluppatori con F12 o premere i tasti Ctrl + Maius + I o dal menù delle impostazioni -> altri strumenti -> strumenti per sviluppatori.

Dagli strumenti aprire la tab Application e poi l'opzione Manifest e cliccare su Add to homescreen

Troverete l'applicazione sul desktop e nel drawer delle app di Chrome.

* Dispositivo mobile  
  Aprire il sito e aspettare che ci venga chiesto di aggiungere l'app alla schermata Home, se non ci viene chiesto subito è necessario ricaricare il sito

**Funzionamento applicazione**

L’applicazione Bee Counter è una Progressive Web App, quindi non è nient’altro che un sito che viene salvato nella memoria del cellulare/tablet/computer visto poi come un’applicazione vera e propria.

Application

Server

mysqljs

HTTP

Back-end

server

NodeJS

+

ExpressJS

Database

MySQL

HTML

+

CSS

+

JavaScript

Angular4

query

Array JSON

Richieste HTTP

Array JSON

File HTML, CSS e JavaScript

HTTP

richieste

HTTP

Client

Come si può vedere dalla figura il funzionamento è molto semplice, esiste l’applicazione (Angular4) che comunica con il back-end (NodeJS/Express) che a sua volta invia una query al database per ricevere un array di oggetti JSON con il risultato della query, che viene poi aggiunto al file HTML che vedrà il Client alla fine del processo.

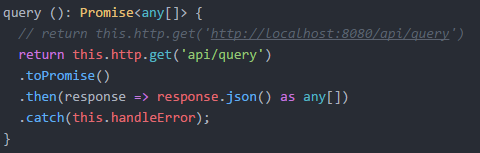


Figura : Esempio di un metodo con richiesta GET al server

Nella figura viene illustrato un metodo, della classe *Database Service*, che invia una richiesta GET e aspetta, con il metodo *toPromise()*, di ricevere un array di oggetti. (Se si vuole lavorare in locale è opportuno commentare riga che non richiede al localhost).

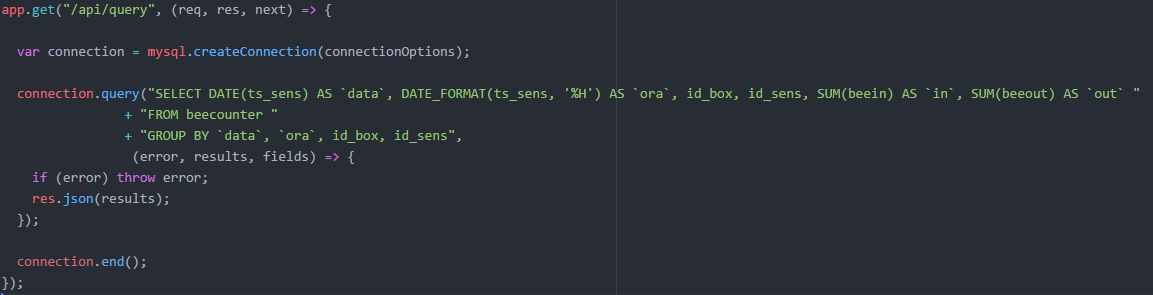


Figura : End point del server della richiesta in figura 1

Qui si vede come per ogni richiesta viene prima creata una connessione con il metodo *mysql.createConnection()* per inviare la query al database la tramite il metodo *connection.query()* e poi viene subito chiusa con il metodo *connection.end()* per non lasciare connessioni aperte al database.



Figura : metodo che viene eseguito all'apertura della pagina /table

Nella figura 3 viene mostrato il metodo *ngAfterViewInit()*, proprio di Angular, che viene eseguito ogni volta viene aperta la pagina /table e consiste nell’inviare una richiesta, tramite il *Database Service (db)*, e aspettare che arrivi un array come risposta. Il resto degli attributi che si vedono servono per determinare i minimi e i massimi delle date/arnie/sensori per avere dei form meglio funzionanti nella parte di HTML.

La parte HTML è formata da un navbar che permette di navigare tra le due parti del sito (tabella e grafico, ndr), un footer con il copyright e la parte centrale che mostra la tabella o il grafico e le opzioni per una visualizzazione dei dati come si vuole. Inoltre i dati della tabella vengono inseriti in gruppi da 200 per permettere un caricamento più veloce, specialmente sui dispositivi mobile.

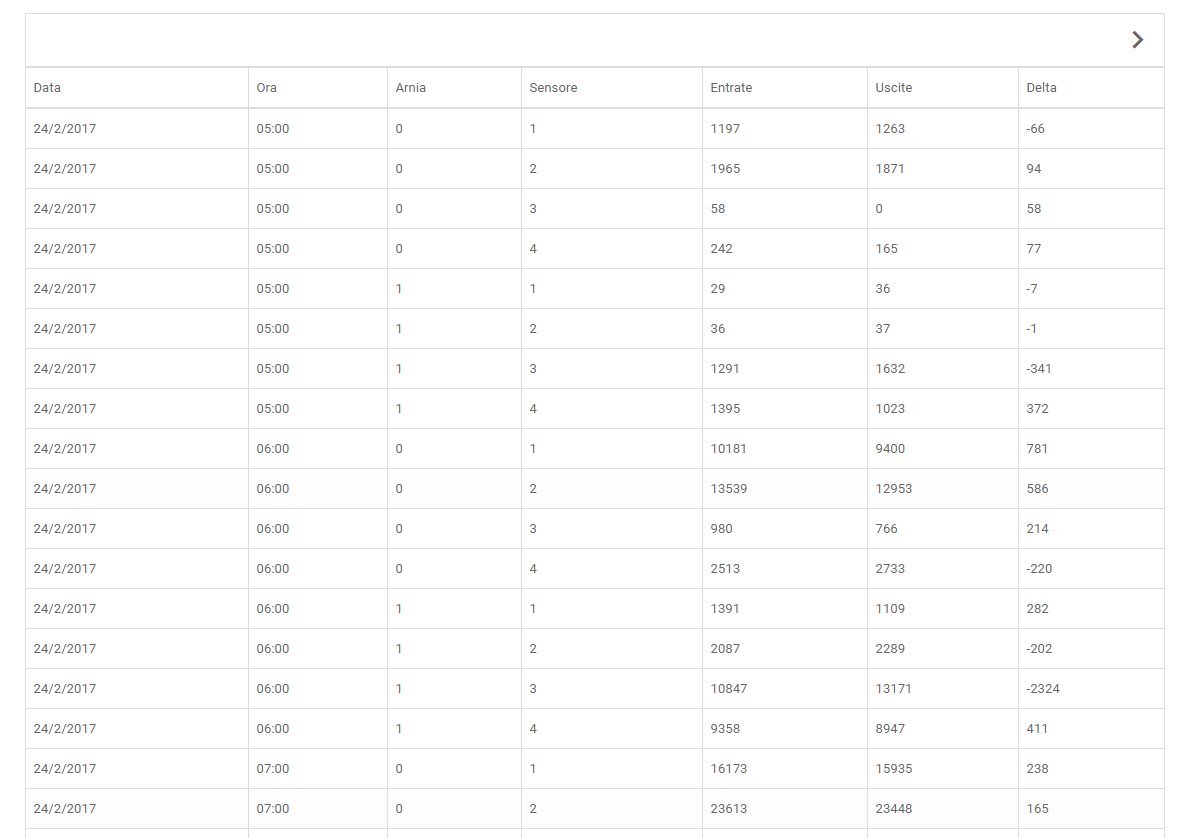


Figura : tabella del sito

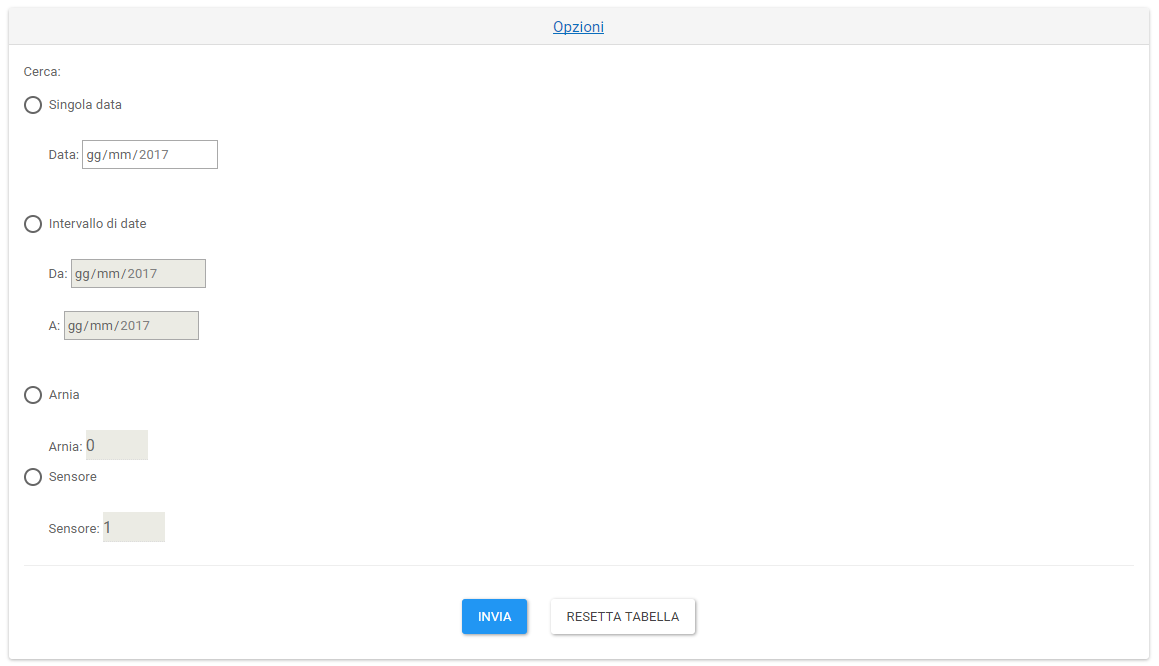


Figura : Opzioni tabella

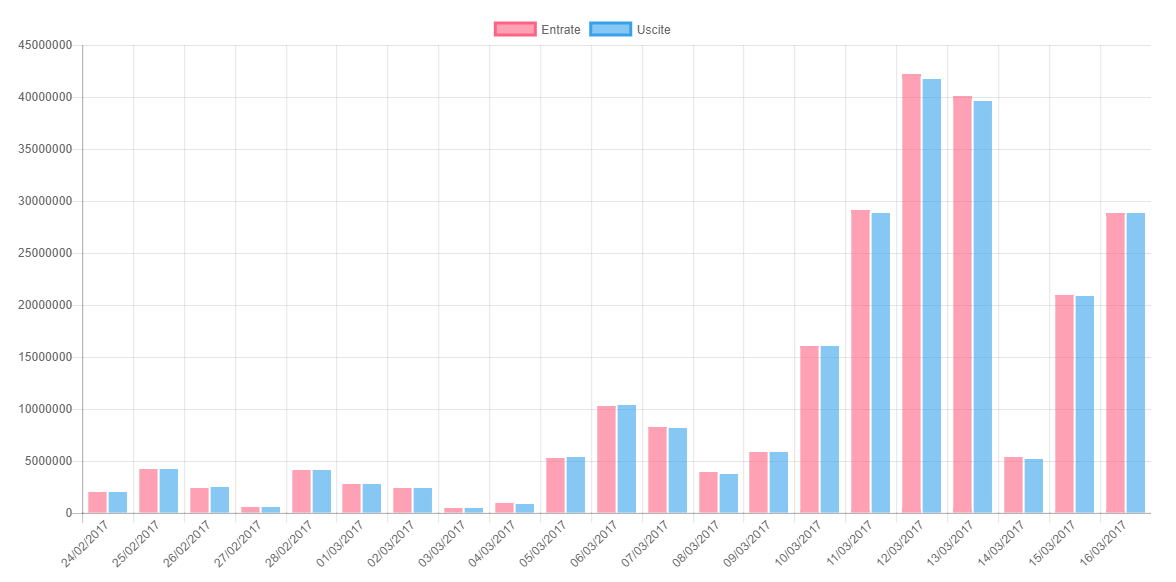


Figura : Grafico