WI-LO Web Application: a simple webapp for WIreless LOcator based on MEVN stack

Fulvio Marcelli*, Natale Vadala*,
* DISI, University of Bologna, Italy

Emails: fulvio.marcelli@studio.unibo.it, natale.vadala@studio.unibo.it

***Abstract*—This project aims to allow users to quickly and easily insert floor plans, in 2 dimensions and of various formats (planimetries), of interesting buildings and to indicate the relative Reference points over them.**
This is very useful in the development of Indoor localization applications, that are increasingly of common interest in today's world. In fact this is one of the future developments proposed by the authors of the WiLo article. [1].
To solve this problem we decided to develop a web application exploiting the MEVN stack (MongoDB, ExpressJS, VueJS, Node.js) for its characteristics that allow a high modularity, modifiability, reusability of the code and easy integration of many APIs and (JavaScript) libraries for map manipulation, to build easily a simple GIS-like application.
The web application created in this way allows the user:

- **to register his own building by locating it on the terrestrial map,**
- **record the plan of interest of which he can import the 2d planimetry in any image format,**
- **place the plan on the terrestrial map,**
- **insert reference points (anchors) on the plan,**
- **save positions and coordinates on the database.**

The software developed is an open-source re-implementation inspirated by the Indoor Atlas application. [5]

I. Introduction

The presented software is in fact a web application (exploitable by browsers) to offer, generally, the possibility to a user to upload the image of a plan on a map (satellite), to be able to add anchors (reference points) and save positions and coordinates on a database.
This data can be used to measure the performance of network devices (routers, switches, beacons) or to practice, for example, measurements and obtain a test environment for indoor location techniques.
This project was born from the idea offered after reading a scientific article of the University of Bologna concerning the study of indoor localization techniques using the smartphone sensors, just the WI-LO [1] framework.
Among the future developments presented by the authors of WI-LO there was the possibility for users to import 2D plans in multiple formats, and it is this idea to be the ultimate goal of this web-app.

II. Related works

Our project target is that to implement a web application for allow at users to import plans in 2 dimensions and in various formats, to face one of the future developments proposed by the authors of WILO.

A. WILO paper

The system introduced by the authors of article WILO [1] is a hybrid system for indoor localization which combines the following techniques:

- Pedestrian Dead Reckoning,
- Radio Fingerprinting,
- Human Activity Recognition (HAR),
- source Wi-Fi, LTE, BLE,
- magnetometer (MAG),

to obtain an accuracy indoor localization and providing a practical and energy-efficient solution for Android devices.

B. Indoor Atlas

An other system that addresses the problem of indoor location is Indoor Atlas [5]. This system has been implemented from a Finnish startup and is based on a cloud of geomagnetic maps, generated by users during the training phase performed using the Indoor Atlas Map Creator 2 [6] app and exploiting the magnetometer of the mobile device.

Also this system uses a hybrid solution combining:

- geomagnetic map,
- dead reckoning through gyroscope and accelerometer,
- Wi-Fi signal,
- barometric height information,

to get the most accurate indoor location possible.

This system also offers to the user the possibility to insert the plan of a building in 2 dimensions and place it on the terrestrial map exploiting a web application from which we have taken inspiration for the realization of our project.
It also provides an SDK to be integrated in your application for indoor location that communicates with Atlas cloud which calculates the current location of the device.

Fig. 1. MEVN stack. Source: https://medium.com/@anaida07/mevn-stack-application-part-1-3a27b61dcae0

III. Architecture

The web-application presented is built exploiting the MEVN stack (MongoDB, ExpressJS, VueJS, Node.js), because it's high a great solution for develop PWA[1] (Progressive Web Application), providing high modularity, modifiability and re-usability of code (from the "front-end developer" perspective mostly).
Another key point of this stack is the easily to integrate JavaScript libraries from the Node environment, that is a very mature and large community.
Lastly, Express for the routing logic and APIs creating and MongoDB as the NO-SQL solution are very simple to use and very useful to build simple REST web-application.
However, who is writing based his work also on preceding studies about this architecture: in the paper Using the MEAN Stack to Implement a RESTful Service for an Internet of Things Application [2], there is the description for an implementation of the full stack (using Angular[2] instead of VueJS) to exploit a fast, scalable, secure and easy to build RESTful service. The authors wanted to develop a little server for domotic scopes. We can confirm that this architecture is a good approach to implement other kinds of IoT applications (crowdsensing, industrial monitoring, wireless networks, smart cities, agricolture).
The scheme is simple: there are 2 ways to interact with our server. The first one is a Web Application and its role is to provide to the user a manipulated view of the data (in a readable form, not JSON) and the possibility to add data (in our case, images over maps). The Web Application contains an Authentication Mechanism because there is the possibility to have more users, with different views. The second way to interact is with the API built on ExpressJS, that are simple GET/POST HTTP requests (encoded in JSON) to the server. These methods are used by the device, however using the MEAN stack we have a high amount of reusable code (all the code is written in JavaScript), indeed the Web Application exploits the API too.

[1]https://developers.google.com/web/progressive-web-apps/
[2]https://angular.io/

A. Stack MEVN

After the birth of the Internet of Things concept, and the exponential growth of interest in big data during the last few years, some of the most sought requirements are now scalability, flexibility and adaptability.
These three concepts are well covered by the use of the MEVN development stack. The MEVN stack (MongoDB, ExpressJS, VueJS, Node.js) is born to fix the drawbacks of the existing and more common LAMP stack (Linux, Apache, MySQL, PHP). This new set of tools is suitable for switching from a Pages-driven web development to a Data-driven approach. These tools are based around the JavaScript language, allowing the homogeneity of the programming language (the code) for the whole system.
Traditionally we call this set of tools a stack because of their coverage cover a vertical idea of a web application, but in reality they arent layered (as for the s LAMP stack), they are communicating tiers instead (3-tiers architecture). This is one of the reasons why it allows you to have a MVC architecture. The stack is composed of 3 tiers, all these components are all open-source tools as well.

B. MongoDB

The name derives from humongous, that means enormous, as one of its major advantage is that it offers a dynamic schema, allowing you to store data with different structure within the database This explains its name: MongoDB is highly scalable. MongoDB is the most famous NoSQL Database. It is a document-oriented database, in which documents replace the rows concept of MySQL, and collections replaces the old idea of tables. A collection is a set of documents, which are key-value pairs. A value can be a key-value pair itself, so that document can represent complex nested structures. Documents are JSON (JavaScript Object Notation) objects, which MongoDB converts, under the hood, into BSON (Binary JSON) before storing them.

C. Express

The routing logic, the API management and, more in general, the controller functions are provided by ExpressJS. Express is a free and open-source web-application framework with templating, routing, and session management built in. It is the standard server framework for Node.js. It offers an elegant solution to handling routing and HTTP operations. It is a minimal web server built on Node.js and it provides the main features needed to deliver web applications to browser and

mobile devices. Its developers took inspiration from Sinatra, a popular Ruby framework.

D. Vue.js

Vue.js is an open-source JavaScript framework for building user interfaces and SPA (single-page applications).
It's a progressive framework for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects. On the other hand, Vue is also perfectly capable of powering sophisticated Single-Page Applications when used in combination with modern tooling and supporting libraries.
At the core of Vue.js is a reactive data-binding system that makes it extremely simple to keep the data and the DOM in sync. When using jQuery to manually manipulate the DOM, the code we write is often imperative, repetitive and error-prone. Vue.js embraces the concept of data-driven view. Many comparison between Vue.js and other frameworks are findable here[3].
Another key-point of Vue.js is the pssibility of creation of Progressive Web Applications(PWA): a PWA is a web application that offers an app-like user experience. PWAs benefit from the modern Web technological innovations (Service Workers, Native APIs, JS frameworks) and raise web application quality standard. [4] [5]
Obviously, one of the software at the base of the development of our project is Webpack. Webpack is defined by its own developers as a static module bundler for JavaScript applications. Its purpose is therefore to create a package of assets that can be used directly in the browser starting from a set of source files structured on different files and with complex dependency schemes[6].

E. Node.js

Node.js is a server side JavaScript execution environment. Its written in C++, distributed since 2009 and is built on top of Google Chromes V8 JavaScript Runtime engine. Node.js uses event-driven, asynchronous programming, and designed for net- work services. Developer have to register the corresponding callback function according to its business logic. This method, together with asynchronous callback functions, allows applications to make the most of the available re- sources. The biggest differences between a platform based on LAMP stack and its equivalent built on top of Node.js are that Node.js takes charge of most of the hassle of managing competing requests and blocking functions, thanks to its event-driven logic and the asynchronous function. Furthermore Node.js has a well established and fast growing community which builds lots of useful open source modules, and makes them available to all developers through a package manager (Npm). The missing ones can be implemented and integrated by the developer of the platform, for all his needs.

[3]https://vuejs.org/v2/guide/comparison.html

[4]https://blog.sicara.com/a-progressive-web-application-with-vue-js-webpack-material-design-part-1-c243e2e6e402

[5]https://medium.com/bam-tech/a-progressive-web-application-with-vue-js-webpack-material-design-part-2-a5f19e70e08b

[6]https://www.html.it/articoli/webpack-il-module-bundler-per-javascript/

IV. Implementation

A. OpenLayers

OpenLayers is an open-source (provided under the 2-clause BSD License) JavaScript library for displaying map data in web browsers as slippy maps.
It provides an API for building rich web-based geographic applications similar to Google Maps and Bing Maps. [7]

B. OpenStreetMap

OpenStreetMap (OSM) is a collaborative project to create a free editable map of the world. Rather than the map itself, the data generated by the project is considered its primary output. The creation and growth of OSM has been motivated by restrictions on use or availability of map information across much of the world, and the advent of inexpensive portable satellite navigation devices. OSM is considered a prominent example of volunteered geographic information. [8][9] OpenLayers (and VueLayers too) permits to use different providers for the maps, and actually the WILO webapp project uses OSM, but it can be changed to use Google, Bing or Mapbox Maps.

C. Mongoose

The Mongoose library is located in the large package environment for Node.js. It's a MongoDB object modeling tool designed to work in an asynchronous environment.
It can be installed in an Node.js environment through the packet manager npm[10] (Node Package Manager).
This package is very useful to define schema of the *documents* that must to be save on the database. It permits to define reference between documents too (e.g. each user has 0-n buildings, referenced by the building id; each building has 0-n floors, referenced by the floor id, each floor has 0-n anchors, referenced by the anchor id).
Mongoose is useful to initialize the database istance: in fact it allows to create, at the first execution, the necessary documents / tables (declared in the models) inside the database, doing in fact what should have been done by the developer before launching his own web application.

D. APIs

The APIs, created with ExpressJS, represents the CRUD implementation of the documents obtained following the models described.

[7]https://en.wikipedia.org/wiki/OpenLayers

[8]https://en.wikipedia.org/wiki/OpenStreetMap

[9]https://www.openstreetmap.org

[10]https://www.npmjs.com/

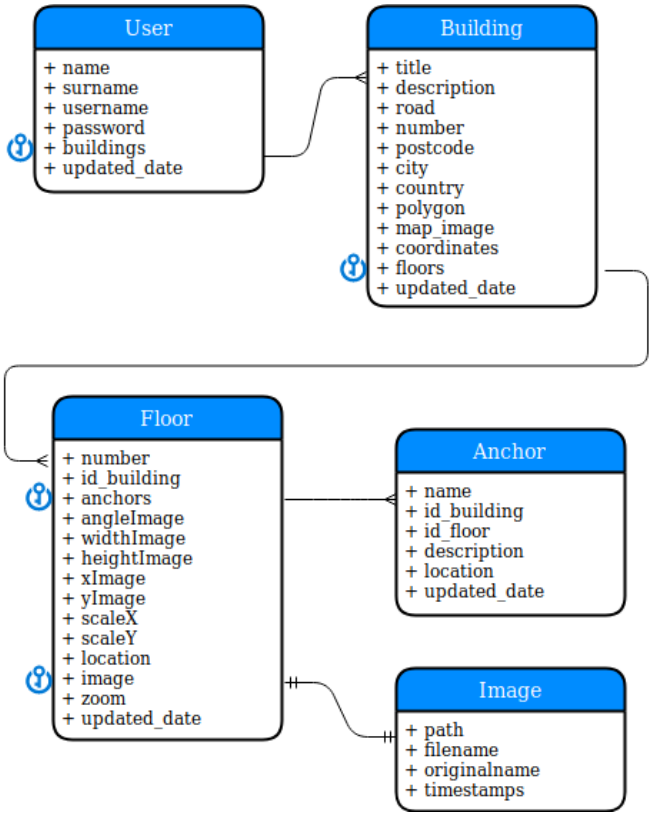

Fig. 2. Database models

We have the *api/* route to obtain the elements like buildings, floors, anchors, and the *auth/* route to obtain user related data.

```
-> http://localhost:3000/
  '/auth/user'
  '/api/building'
  '/api/floor'
  '/api/anchor'
  '/api/image'

for each of these elements:
    /* GET ALL elements*/
    method: 'get'
    path: 'element/'

    /* GET SINGLE element BY ID */
    method: 'get'
    path: 'element/:id'

    /* SAVE element */
    method: 'post'
    path: 'element/'

    /* UPDATE element */
    method: 'put'
    path: 'element/:id'

    /* DELETE element */
    method: 'delete'
    path: 'element/:id'
```

At the backend level, we can enumerate the APIs, that are filtered to follow the relative references between documents on the frontside.

On the web-application, the browsable paths are the following:

```
-> http://localhost:3000/#/ (Note the dash)
// Authentication
path: '/auth'
children: [
  '/login/',
  '/logout/',
  '/registration/'
]

// Home - Building level
path: '/'
children: [
  '/about',
  '/add-building',
  '/show-building/:id_building',
  '/edit-building/:id_building',
  '/buildings',
  '/myfloors'
]

// Floor level
path: '/building/:id_building',
children: [
  '/add-floor',
  '/show-floor/:id_floor',
  '/edit-floor/:id_floor'
  '/floors'
]

// Anchor level
path: '/building/:id_building/floor/:id_floor',
children: [
  '/add-anchor',
  '/show-anchor/:id_anchor',
  '/edit-anchor/:id_anchor'
  '/anchors'
]

// Admin
path: '/admin',
children: [
  '/users',
  '/user/:id_user',
  '/edit-user/:id_user',
   //and the show-edit views
   for each of these elements
```

```
  '/buildings',
  '/floors',
  '/anchors'
]
```

V. Usability of the app

The developed web-application presents a classic layout with a sidebar, to navigate between tables or between actions (CRUD operations) among the data, and a topbar, useful to navigate between the home, the information page (Abous Us) and the authentication dropdown list.
On the topbar there is also a switch button to apply the default or the dark theme to the app (nightly vision).
All the components used in the app are part of the BootstrapVue[11] library, a mature and well-done implementation of Bootstrap V4 components and grid system available for Vue.js 2.5+, complete with extensive and automated WAI-ARIA accessibility markup.

As described in the APIs section, we have planned and implemented an "Admin section", in which the user that has the flag *is_admin* setted to *true* in the DB can query and read and modify all the users and relative buildings, floors and anchors.

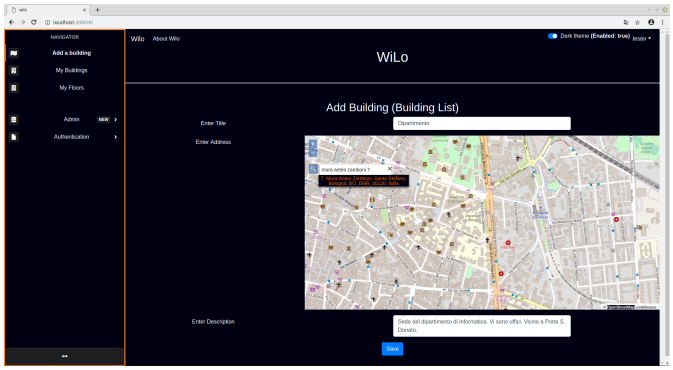

Fig. 3. Creation of a building at the Computer Science department of Bologna

A. Authentication

The authentication mechanism of the app is token based, using JWT (JSON Web Token)[12], a JSON-based open standard (RFC 7519) for creating access tokens that assert some number of claims, very usable especially in a web-browser single-sign-on (SSO) context.
JWT relies on other JSON-based standards: JWS (JSON Web Signature) RFC 7515 and JWE (JSON Web Encryption) RFC 7516.[13] [14] [15] [16]

[11] https://bootstrap-vue.js.org/
[12] https://jwt.io/
[13] https://tools.ietf.org/html/draft-ietf-oauth-json-web-token-32
[14] https://tools.ietf.org/html/draft-ietf-jose-json-web-signature-41
[15] https://tools.ietf.org/html/draft-ietf-jose-json-web-encryption-40
[16] https://tools.ietf.org/html/draft-ietf-jose-json-web-algorithms-40

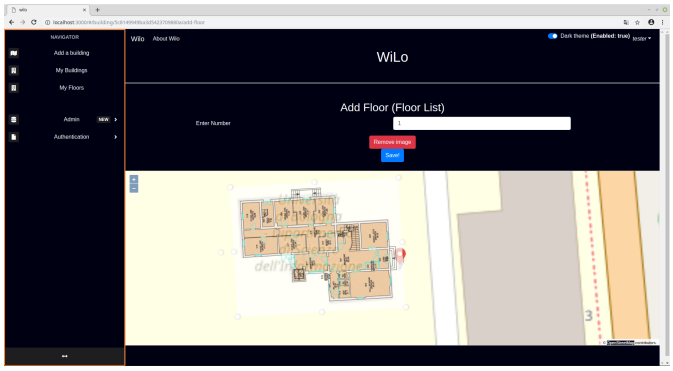

Fig. 4. Creation of the 1st floor over the Computer Science department of Bologna

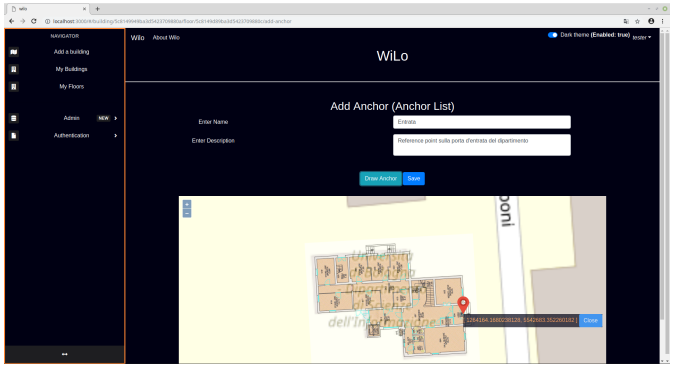

Fig. 5. Creation of an anchor over the 1st floor of Computer Science department of Bologna

This type of authentication permits to allow only the authorized users (or devices) to query to the APIs end-point by the use of a JWT as an authorization header for each query, in addiction to the security controls made in the front-side of the webapp.

VI. Conclusions

After the description of the WILO Webapp, we can make some considerations about the limitations and the problems encountered during the development phase of the project.
We have to admit that this application is a Proof of Concept,

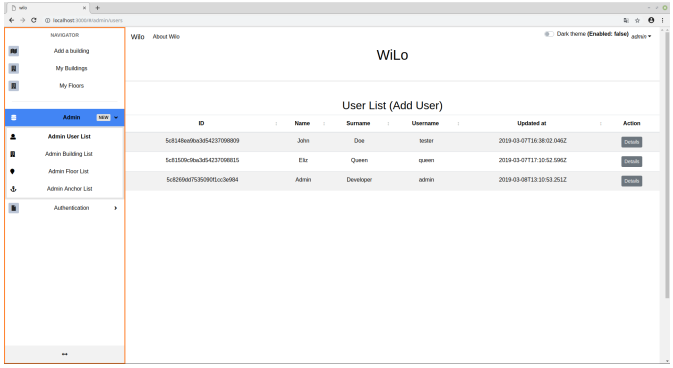

Fig. 6. White theme

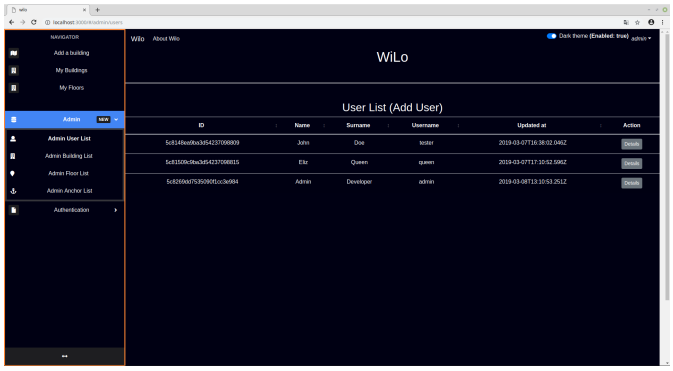

Fig. 7. Dark theme

because of some concepts that we didn't consider to relax the features and to concentrate the work on the development of the main features. In fact, the application can be improved, on many points: - from the security / authentication side, a complete auditing/penetration test may be helpful, - the application of anti-dos / ddos / xss filters and client side attacks in general , - the improvement of the authentication system with recovery / reset password for users, - the application of rules more rigid on user input and database contents, - the implementation of a solid check on uploaded images so that they can not contain unpleasant or defective or malicious content, - the *dockerization* of the application (the writers have foreseen the insertion of the application inside more docker container, so you can work on any server with docker installed. The *dockerization* of the app is proceeding on another branch of the repository) moreover, - also graphically the application can be significantly improved, with the change of the libraries used in the frontend (e.g. migrating from Bootstrap to Material Design). However, we think it is a good proof of concept of what the initial idea has led us to develop this application: the possibility to add 2d floor plans and reference points on an online map.

Some possible future works, starting by this idea, can be the implementation of the possibility to export solid (3d) figures from a building with more floors, to have a 3d idea of the building, or the creation of "blind elements" on map, useful to define reference points in which the radio frequencies/ signal of the communications are obfuscated: in an indoor localization environment may be helpful to have the possibility to draw anchors/reference points, in which a source of signal is present, or blind elements in which any communication is lost.
We sincerely hope that this work can help someone, that someone will improve it and that, perhaps, can serve as a basis for some project, given the commitment and the beautiful experience of study obtained by the writer during development of this idea.

REFERENCES

[1] Stefano Traini, Luca Sciullo, Angelo Trotta, Marco Di Felice , Practical Indoor Localization via Smartphone Sensor Data Fusion Techniques: A Performance Study .
[2] Andrew John Poulter , Steven J. Johnston, Simon J. Cox, F., Using the MEAN Stack to Implement a RESTful Service for an Internet of Things Application, *IEEE 2nd World Forum on Internet of Things (WF-IoT)*, 2015.
[3] S. L. Bangare, S. Gupta, M. Dalal, A. Inamdar, Using Node.Js to Build High Speed and Scalable Backend Database Server, *International Journal of Research in Advent Technology (E-ISSN: 2321-9637) Special Issue National Conference NCPCI-2016*, 2016.
[4] Aishwarya A. Kadam, Harshada R. Chaudhari, Chandani M. Patil, S. P.Chavhan, Web Application Development by Using MEAN Stack, *International Journal of Science Technology, Management and Research*, Volume 2, Issue 4, April 2017.
[5] IndoorAtlas, https://www.indooratlas.com/
[6] Mapping Quick Start Guide with MapCreator 2 - IndoorAtlas, https://docs.indooratlas.com/app/