_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE UPV INGENIEROS DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

CLASS ATTENDANCE BY USING IBEACON™ TECHNOLOGY IN A HYBRID APPLICATION

Ruben Alliet

Bachelor informatics

Tutor: Rafael Llobet Azpitarte

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València.

Curso 2015-16

Valencia, 6 de julio de 2016

Acknowledgments

First of all I want to thank my family, my friends and my mentor at Universitat Politècnica de València Rafael Llobet Azpitarte for all the support during this project. They kept me motivated and focused until the end of the project. Without them it would have been much harder to make this all possible.

Abstract

This bachelor thesis was carried out in the department of ETSIT and ETSINF at the Universitat Politècnica de València(UPV). The aim of the project is to enhance the current process of taking attendances of students in lessons or exams. This will be done by the writing of an application, which is hybrid.

The project consists of three parts. First of all the iBeacon™ that transmits his UUID in a range of 15 meters, which identifies the classroom. This identifier will be received by the hybrid application. Which is the second part of the project. This multiplatform application written in HTML,CSS and JS will communicate through API requests with the third part, the API. This will manipulate and read out the student attendances in the database. Both API and database are hosted on the same Windows server.

Keywords: Hybrid application, API, NodeJS, ExpressJS, MongoDB, BLE, iBeacon™, Windows server 2012

Resumen

Esta tesis de licenciatura se llevó a cabo en el departamento de la ETSIT y ETSINF en la Universidad Politécnica de Valencia (UPV). El objetivo del proyecto es mejorar el actual proceso de registrar asistencias de los estudiantes en las clases o exámenes. Esto se hará mediante la escritura de una aplicación híbrida.

El proyecto consta de tres partes. En primer lugar, la iBeacon™ que transmite su UUID en un rango de 15 metros, que identifica el aula. Este identificador es recibido por la aplicación híbrida, que es la segunda parte del proyecto. Esta aplicación multiplataforma escrita en HTML, CSS y JS se comunicará a través de solicitudes de la API con la tercera parte, la API. Esta manipulará y leerá las asistencias de los estudiantes en la base de datos. Tanto la API como la base de datos están alojadas en el mismo servidor de Windows.

Palabras clave: Hybrid application, API, NodeJS, ExpressJS, MongoDB, BLE, iBeacon™, Windows server 2012

Index

1.Introduction

This final project will cover the full development of an application and a backend database that allows a student and/or a teacher to monitor attendances of students in lessons and exams. The development of the idea and the operation of the project itself will be explained comprehensively in this document.

First of all we need to get an image of the current process. How do teachers take attendance of students? How do they monitor it? Thereafter all the problems associated with that current process will be discussed.

After this all the problems of the current process will be explained. Then the solutions are provided for these problems.

All this information results in a project description. This will give a clear image of how the result of the project will look like. After defining the project description we need to draw up a project schedule to distribute optimally the available time.

During the implementation of the project, all the used technologies and tools of the project will be discussed comprehensively.

Finally a conclusion will be written in which we will summarize the objectives that have been accomplished in the project and which not. In this conclusion I will also reflect on all the things I have learned and the things I should avoid in the future.

1.1.Current situation

At the end of a lesson or exam, a student goes to the desk of the teacher and signs the paper to prove his or her attendance in an exam or lesson. The teacher needs to keep all the attendance sheets and can't lose any of them. Then teachers have to take the sheets to the desk of the concierge. The school staff must scan and process these sheets in order to store the attendance of each student in a database.

1.2.Problems

In the current situation we face the following problems:

- Loss of time when waiting in the queue to sign the paper
- Possible loss of attendance sheets
- Possible human errors during the scanning and storing process of the attendances in the database

1.3.Motivation

As a programmer you have a lot of freedom in the implementation of this project idea. Also lots of different interesting technologies needs to be combined to make the entity work. For example iBeacons™, it's a relative new promising technology with lots of capabilities. Different technologies need to be compared and the most suitable needs to be chosen. Bit by bit you get closer to your end goal. During the development of the project many obstacles pop up that needs to be solved, which is very challenging. As a student you can understand the context of the project. I wanted to create something to make life easier. And I think this project could be very interesting for students and teachers.

1.4.Objectives

General objective

To build an application that allows students to prove their attendance in lessons or exams. When a student enters the classroom the application will communicate with the iBeacon™ that is attached on the wall. The received UUID of the iBeacon™, that identifies the classroom and the id of the user will be used by the application to send an API request. This request will set the attendance of the student for a lesson or exam in the database. Students will be able to consult their own attendance history. And teachers will be able to consult the attendance history of all students.

Specific objectives

- Configure and implement the communication process of the iBeacon™
- Implement an application for students and teachers that is available for all platforms
- Implement an API that communicates with the database
- Design and populate a database to store the attendance history

Result

Implementing this project idea will solve the queue problem of signing the attendance sheets. No more attendance sheets will be necessary because the application manages all that data. And as last, possible human errors during the scanning and storing process of the attendances in the database are avoided because the process is now automated. Now the staff of the school doesn't need to spend time anymore for processing the attendance sheets.

2.Project description

Create an application that allows students to prove their attendance in lessons or exams by using iBeacon™ technology. When a student enters a classroom the application will communicate with the iBeacon™ that is attached on the wall. The received UUID of the iBeacon™ and the id of the user will be used by the application (IOS, Android or Windows) to send an API request to the Windows Server. This API request will set the attendance of the student at a lesson or exam in the database. The application lets students consult their own attendance history. Teachers can consult the attendance history of all students. If the iBeacon™ is broken or the battery is dead, the teacher can always set the attendance of a student manually.

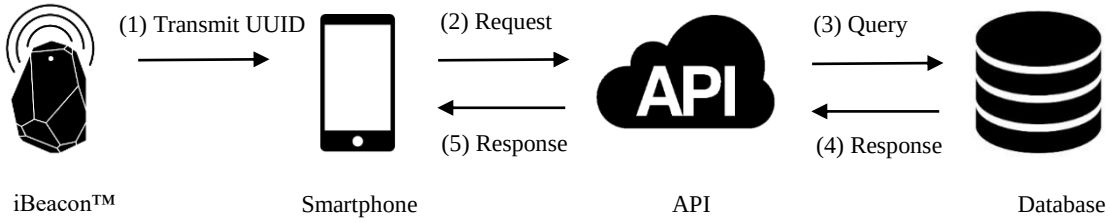


Image 1. Interaction of the architecture.

3.Project schedule

The project schedule is a very important aspect to obtain a good result. This makes the difference between a successful and an unsuccessful project and whether the project has been completed efficiently and within the specified time span.

I have split up the project into multiple parts and subparts if necessary. This creates a better overview of what has been done and what remains to be done. Each part of the project has an expected time slot. During the development, I will try to follow this schedule as closely as possible.

Project parts	Expected time required
Part 1: The purchase and configuration of the IBeacon™	2 weeks including the delivery time of the iBeacon™.
Part 2: Create Database and API - Find hosting for database and API - Create Database design and populate it - Write the API - Test API and Database	3 weeks
Part 3: Create application with Ionic - Prototyping with Ionic Creator - Fine tune layout and flow of the application by coding - Integrate API requests into the hybrid application - Test application	4 weeks
Part 4: Final testing phase	2 days
Part 5: Write memory - Research - Create interaction schemes - Write text	6 weeks

Table 1. Project schedule.

4.Used technologies

In this part of the essay I will explain the used technologies in this project. First I will explain the architecture diagram of the project to get the necessary understanding in the general process. Secondly, all the chosen technologies used for the application will be discussed and motivated. And as last, the technologies of the server will be described.

4.1.Architecture diagram

On the image below we can see the general structure of the project implementation. The database and API are hosted on a Windows server located in Ghent. The server is situated in a private network hosted by Odisee University college Ghent. It is only accessible when connection is made to the Odisee VPN. Because of the VPN there is a small delay that is noticeable when doing API requests.

The Application that I have written is available for IOS , Android and Windows Phone. Because of that, every student should have the ability to use the application. The applications will be downloadable in its associated app store when the project is officially launched.

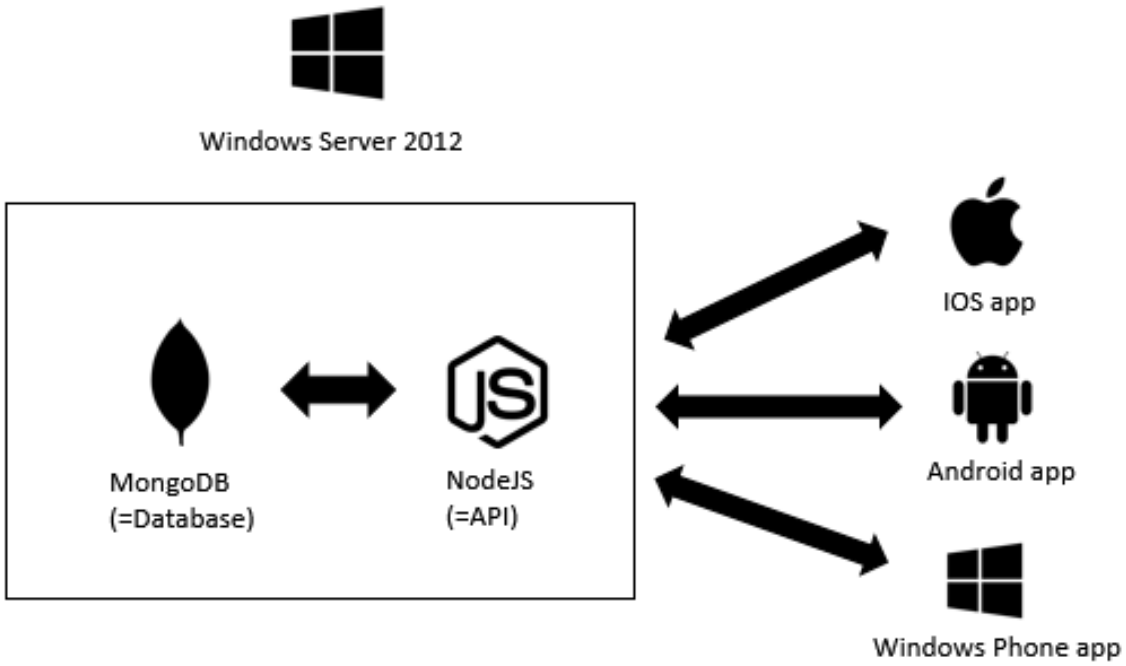


Image 2. Architecture diagram.

When the application performs a request to the API then the data will be manipulated or read out of the database depending on the type of request and its parameters. After the request a response will be sent back in JSON format. The API is written in Node.js. This technique allows you to build fast, scalable network applications capable of handling a huge number of simultaneous connections with a high throughput.

4.2.Application

In this part I will explain all the used technologies for the application. First I will explain and motivate my choice for each technology and if necessary sub technologies. Then I will explain what the technology is used for in the application. As last I will explain the configuration or implementation that is associated with that particular technology.

4.2.1.Bluetooth beacons

4.2.1.1.What is it?

Beacons are small devices that enable more accurate location and are more energy efficient than a cell tower triangulation, GPS and Wi-Fi. [1] Beacons transmit small amounts of data by using Bluetooth Low Energy (BLE) up to 100 meters. For this reason they are often used for indoor location position determination, but beacons can also be used outside.

Beacons are most of the time powered by small batteries, but they can also be plugged into a USB port instead to ensure consistent power. In addition to standalone beacon devices, mobile phones, tablets and PCs with BLE support can all function as beacons, with the ability to emit and receive beacon signals.

4.2.1.2.Project purpose

In the project the Bluetooth beacon will be used for scanning the attendance of students. Every classroom gets an iBeacon™ with a UUID[2]. The purpose of that ID is to distinguish beacons in your network, from all other iBeacons™. Each iBeacon™ will transmit their UUID with a broadcasting interval of 900ms and a transmission power of -12dBm. Which is equals to 15m. The transmission power and broadcasting interval can be adjusted by the configuration app.

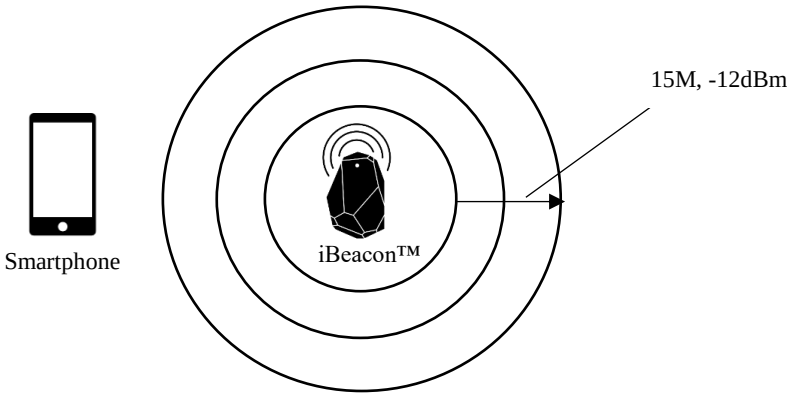


Image 3. IBeacon™ proximity.

A detailed explanation about that can be found in 7.2.1.3. When the smartphone of the student is in the range of a beacon it will receive a signal and extract the UUID out of it. This UUID will be used to pass it to the API. Then the API will check, depending on the database results, whether the student is in the right classroom and on the right time. If he is, the attendance of the student will be set in the database.

4.2.1.3. Configuration

To configure the iBeacon™, download 'BeaconCFG' from the Google play store. This application is recommended by the vendor of the iBeacon™. After downloading and installing the application follow these steps:

(1) Turn on Bluetooth.

(2) Open the application called 'BeaconCFG'.

(3) Start page appears with all the detected beacons. Push on the upper right button to change the settings.

(4) All possible settings appear. Now select 'Automatic Configuration'.

(5) Assign following settings to the iBeacon™:

Property	Value
Major	0
Minor	0
Serial Number	1
IBeacon™ Name	1E 1.0 (name of the classroom)
UUID	Choose one of the available values in the list
Transmission power	-12dBm,15m (classroom range)
Broadcasting interval	900ms (save energy)
Change password	12345678
Reboot password	minew123 (standard password)

Table 2. IBeacon™ configuration settings.

(6) Now click save to save all settings and reboot the iBeacon™. All settings will be set when the iBeacon™ is back on.

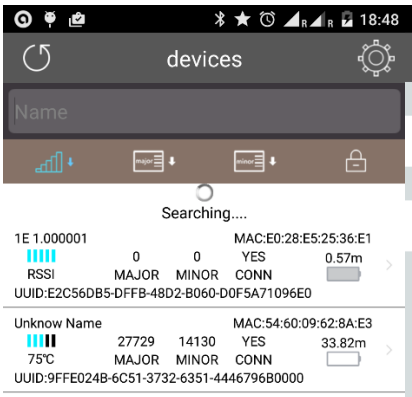


Image 4. start page of BeaconCFG.

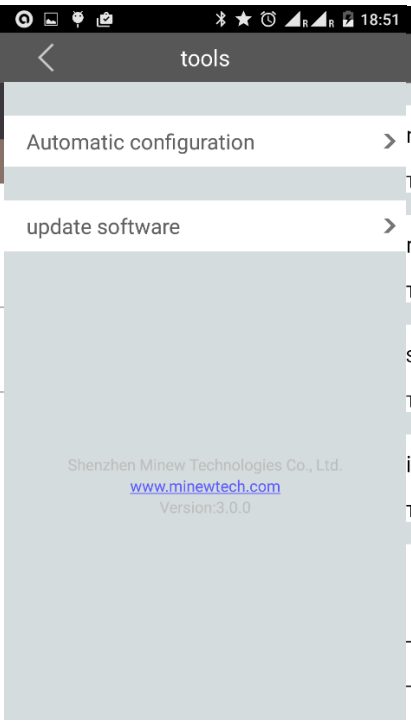


Image 5. settings page of BeaconCFG.

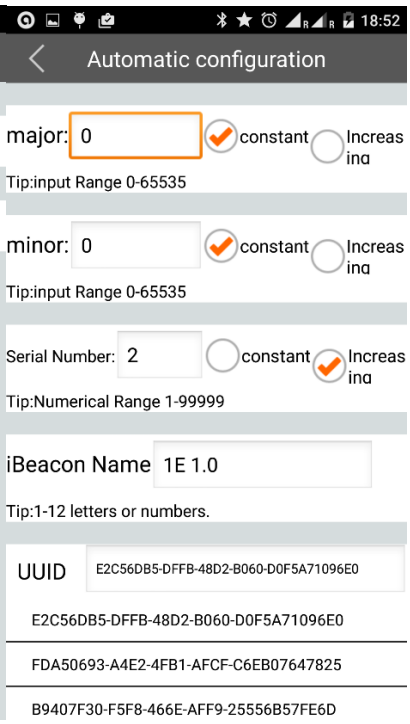


Image 6. automatic configuration page of BeaconCFG.

4.2.2.Hybrid application: Cordova

4.2.2.1.What is it?

Cordova, formerly called Phone Gap is a platform to build Native Mobile Applications using HTML5, CSS and JavaScript. Cordova acts as a container for running a web application written in HTML, CSS and JS. Normally Web applications cannot use the native device functionalities like Bluetooth, GPS, Accelerometer , NFC etc. By using Cordova we can access those and package the web application in the devices installer format. [3]

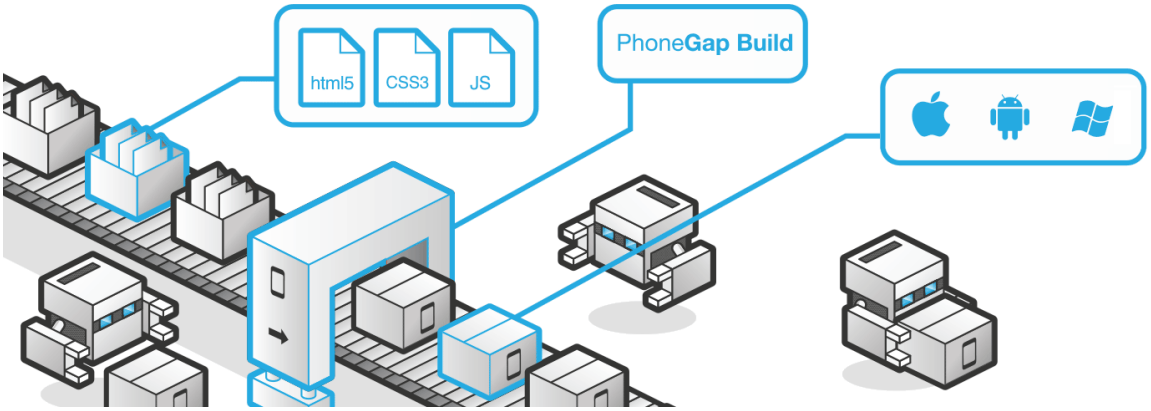


Image 7. Build diagram PhoneGap.

4.2.2.2.Subtechnologies

Ionic

Ionic is an HTML5 mobile app development framework targeted at building hybrid mobile apps. Ionic serves as the front-end UI framework that handles the looks and feel of the UI interactions your app needs to look like a real native application.[4] We can compare it to bootstrap but for applications. Ionic has a wide support of common native mobile components, animations and design. In short Ionic serves as the missing front-end framework for Cordova.

Angular JS

AngularJS is a structural framework for dynamic web apps and hybrid applications. It is used in Ionic and makes it possible to extend the functionality of your HTML syntax. Angular also provides data binding and dependency injection. We can describe HTML as what it would have been, if it had been designed for applications. [5]

Angular attempts to minimize the impedance mismatch between document centric HTML and what an application needs by creating new HTML constructs. Angular teaches the browser new syntax through a construct we call directives.

Here some examples [5]:

- Data binding, as in {{ }}
- DOM control structures for repeating, showing and hiding DOM fragments.
- Forms and form validation support.
- Attaching new behaviour to DOM elements like DOM event handling
- Grouping of HTML into reusable components.

4.2.2.3.How does it work?

The user interface of a Cordova application is a WebView that occupies the complete screen and runs in the native container. It is the same web view that is used by the native operating systems. The native containers change depending on the OS, internally the web pages remain the same. The browser rendering of webpages acts different for each operating system: [3]

- IOS uses the UIWebView class
- Android uses android.webkit.webview
- Windows uses WebViewClass and the similar goes to other OS .

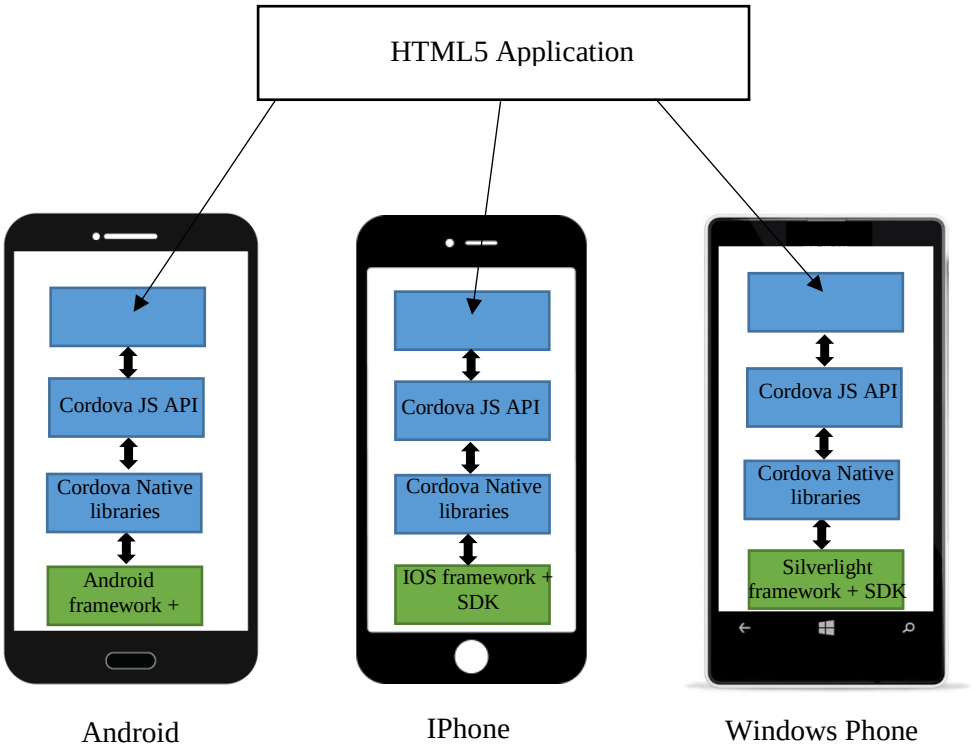


Image 8. Structure of a hybrid application.

The Cordova libraries communicate with the Native Framework of the selected OS . For example the JS library of the IBeacon™ will communicate with the native library of the IBeacon™.

4.2.2.4.Project purpose

In the project, the hybrid application will serve as the client application used by students and teachers. Teachers will be able to check the attendance of students for each lesson or exam. Students will be able to prove their attendance for each lesson or exam. And also get their personal history of attendances. The application will perform all the API requests. When the application is officially launched it will be available on Google Play, App Store and Windows Phone Store.

4.2.2.5.Installation

For the prototyping we use Ionic creator, therefore we only need to open our browser and login, no installation is necessary because it is a web application. For coding, some tools need to be installed. First install Node.js. After that you need to install the latest Cordova and Ionic command-line tools. Then it is necessary to follow the Android and iOS platform guides to install the required platform dependencies. In the case of Android it is necessary to install the appropriate SDK tools.[6] After downloading and installing the SDK manager.

4.2.2.6.Implementation

Prototyping & design

I started prototyping the application using the web application named Ionic Creator. For more details about Ionic Creator go to 9.6. This prototyping tool was very useful for creating the flow of the application and the general structure. After the prototyping phase all generated code needs to be extracted. Now the design will be fine-tuned by coding. All the coding happens in Sublime text, a cross-platform source code editor with a Python application programming interface. More about this code editor can be found in the part 9.2.

Functionality

A very important functionality of the project was the iBeacon™ proximity detection. Therefore a library was necessary for the communication between the smartphone and the iBeacon™. I have chosen for the library called ‘cordova-plugin-iBeacon™’ written by Peter Metz. This library can be found on https://github.com/petermetz/cordova-plugin-iBeacon™.

4.2.2.7.Folder structure

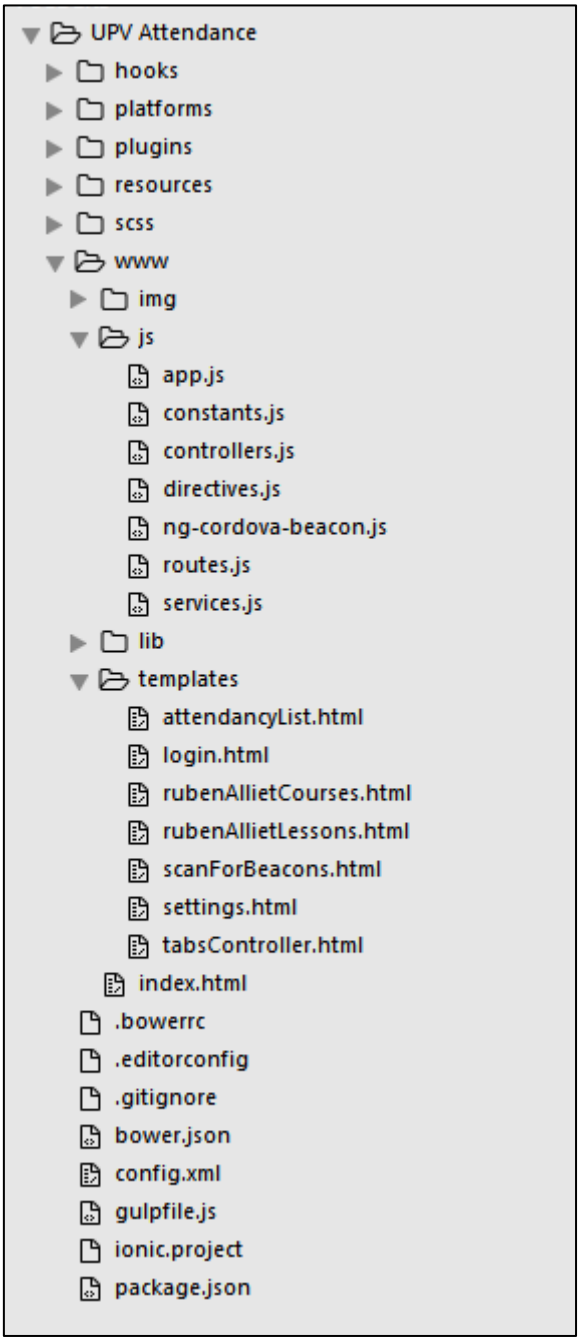


Image 9. Folder structure of the application

In this part I will discuss the directory structure of the application to gain the necessary understanding of the operation of the application.

The main directory is called 'UPV Attendance' and contains several folders like hooks, it contains scripts used to customize Cordova commands. Then we have the folder 'platforms', which contains the build of the hybrid application for each platform that is added. In addition, we have the folder "plugins" that contains all the Cordova plugins to extend the current functionality. 'Resources' is the folder that contains the icons and splashscreens for each platform.

Another folder is called ‘scss’ this one contains all the layout that is used for the application. The folder that is called ‘www’ contains all the main data for the hybrid application. It contains all the images, JavaScript files, HTML files and AngularJS libraries. Beside the folders we also have other files, they are used for general configuration purposes.[7]

4.2.2.8.Application flow

On the following pages you can see all the pages of the application. On image 10 we can see the login page, both student and teacher will reach this page when opening the application. Depending on the user role, the application will redirect the user to another page:

If you are a student the app will redirect you to the personal attendance list(image 16). From there you can navigate to the attendances of a specific course (image 17). If you click on the tab ‘scan Beacon’ (image 11) you will be able to prove your attendance for a specific lesson or exam. If your attendance is set, the Bluetooth symbol will turn green. If a user doesn’t know how the attendance application works he can always check the info tab. This page contains a slider that explains step-by-step how the application works(image 12 to 15).

If you are a teacher the app will redirect you to the page where all students are listed(image 18). When the user types a name in the search field, the application starts searching for corresponding results.[8] When the teacher clicks on a name the courses of this specific student are shown(image 19). From there you can navigate to the attendances of a specific course (image 20). As a teacher you have the appropriate user role to change the status of a lesson or exam.[9]

When all required activities are executed, the user can log out by clicking on the ‘log out’ tab.

Student

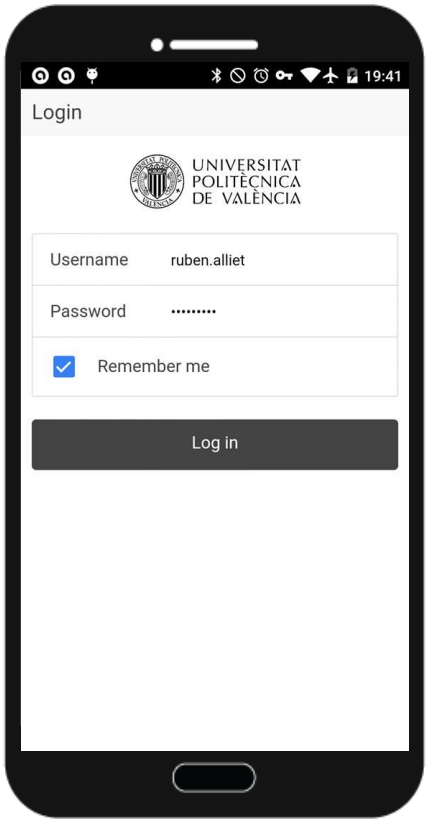


Image 10. Login page.

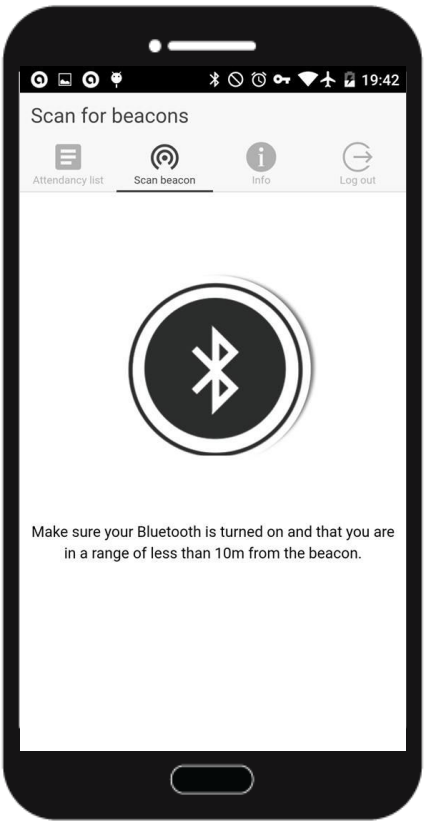


Image 11. scan IBeacon™ page.

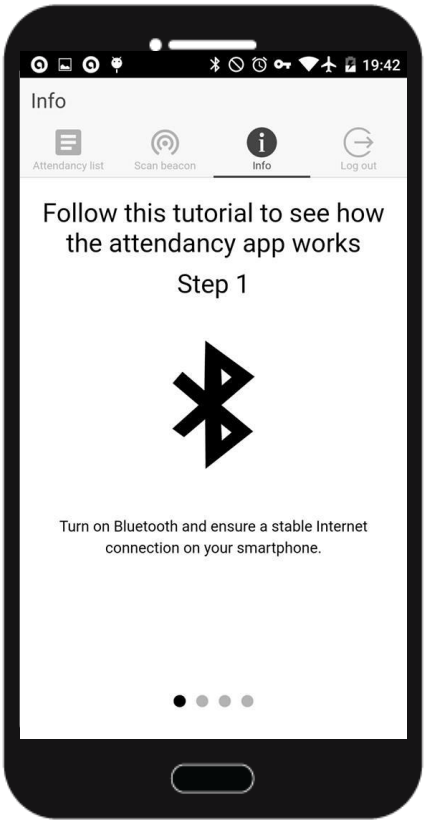


Image 12. Info page step 1.

Image 13. Info page step 2.

Image 14. Info page step 3.

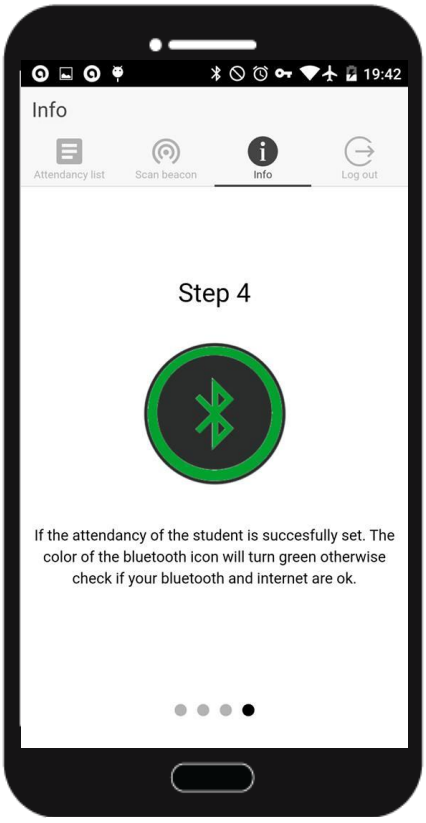


Image 15. Info page step 4.

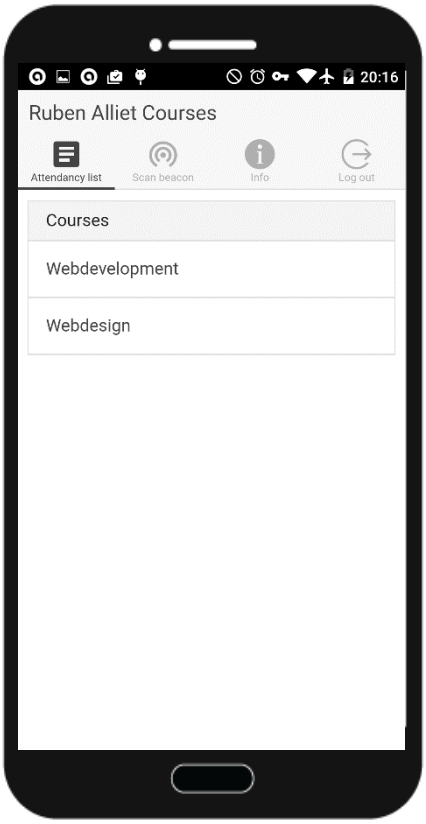


Image 16. Courses of student.

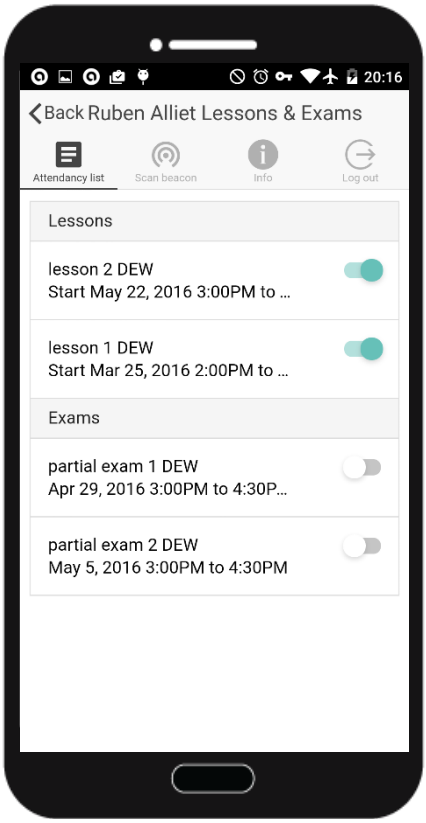


Image 17. Lessons and exams of student .

Teacher

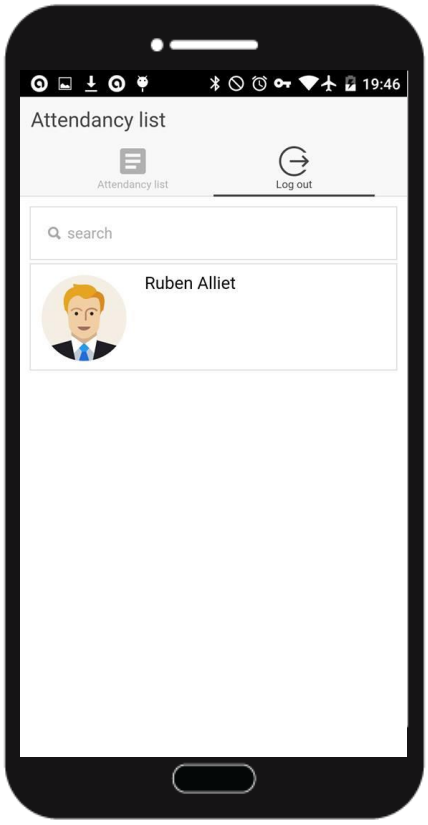


Image 18. List of students.

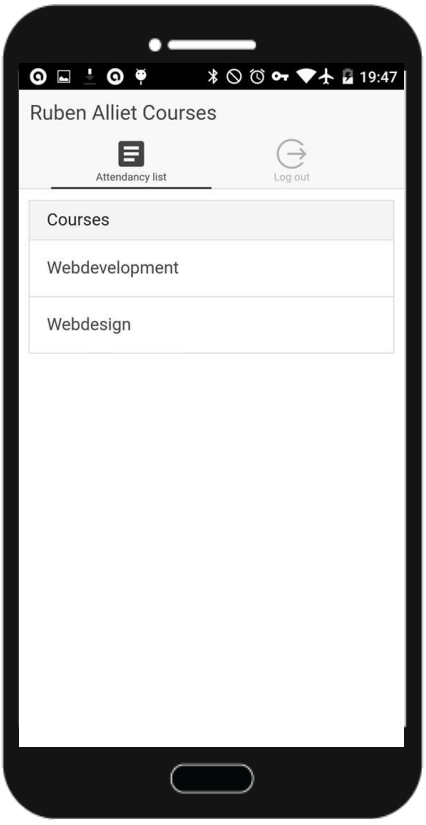


Image 19. Courses of student

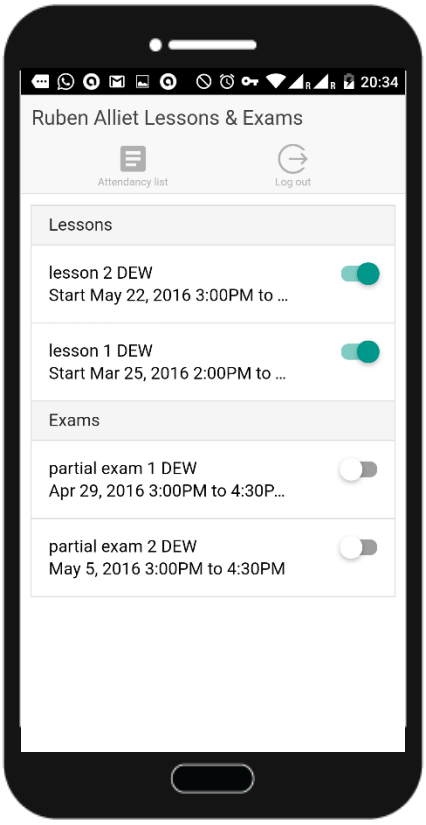


Image 20. Lessons and exams of students

4.3.Server

In this part I will explain all the used technologies for the server. First I will explain and motivate my choice for each technology and if necessary sub technologies. Then, I will explain what the technology is used for in the server. As last, I will explain the configuration or implementation that is associated with that particular technology.

4.3.1.API

4.3.1.1.What is it?

Application programming interface (API) is a list of routine definitions, protocols, and tools for building software and applications. [10] In our case the API serves as a connection between the smartphone application and the database. As you can see on the image below:

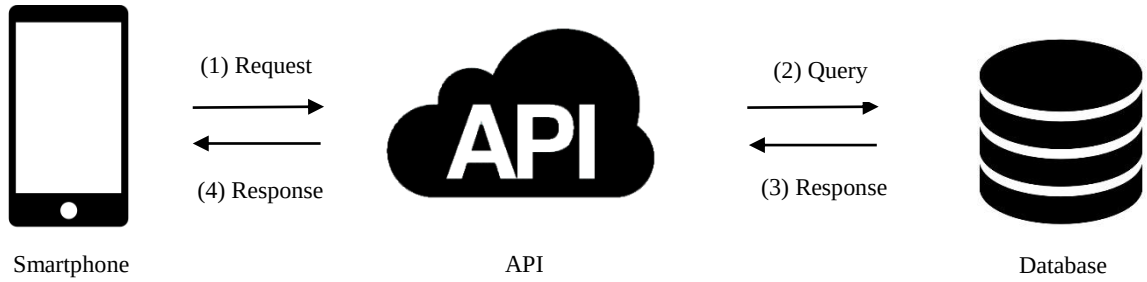


Image 21. communication of smartphone with database by using an API.

The client does a request to the API. Then the corresponding code is executed on the API. This performs a query to the database. The database replies with a response to the API. Thereafter the API will send a response in JSON format to the application. This process repeats itself constantly.

4.3.1.2.Subtechnologies

4.3.1.2.1.Node.js

Node.js is a runtime environment for developing server-side Web applications. It uses the V8 engine manufactured by Google for use in Chrome. The V8 engine compiles and executes JavaScript at a very high speed. The reason for that is the compilation into native machine code. Another characteristic of this runtime environment is that it uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. It operates on a single thread, this allows support for thousands of concurrent connections without incurring the cost of thread.[11]

I had the possibility to choose between several server-side languages for writing the API. So I did the necessary research in order to come to the right decision. On image 22 we can see that the amount of requests handled in one second is the highest for Node.js. Also in contrast to other web-languages, Node.js is written in JavaScript. It's an omnipresent language, so in terms of maintenance that is a very big plus.[11]

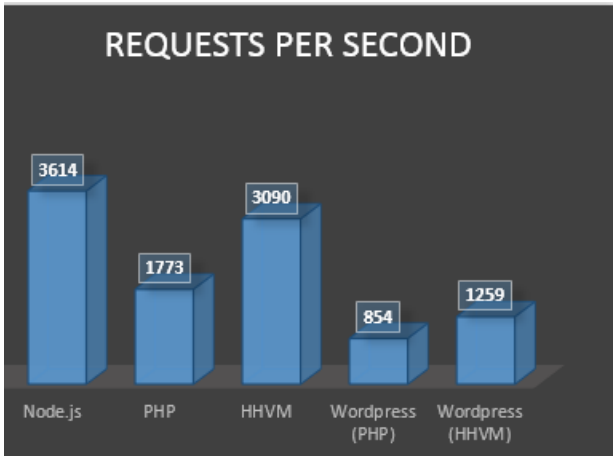


Image 22. Amount of Basic HTTP requests per second to an API.

	CPU time	System time	RAM
PHP 5.6.4	102.69s	104.20s	2497508 KB
HHVM 3.5.0	12.56s	14.83s	362488 KB
Node.js v0.10.35	2.64s	2.64s	92240 KB

Table 3. CombSort Strict CPU Test.

As we can see on table 3, node.js has returned the best results. We are therefore inclined to opt for Node.js.

Node.js also has a package manager npm (node.js package manager). Two important modules I have used are Express and Async. The first module is a minimalist web framework and the second is a utility module which provides straight-forward, powerful functions for working with asynchronous JavaScript.

4.3.1.2.2.Jason Web Token

JSON Web Token (JWT) is a JSON-based open standard (RFC 7519) for passing claims between parties in a web application environment. The tokens are compact, URL-safe and usable especially in web browser single sign-on (SSO) context. JWT claims can be typically used to pass identity of authenticated users between an identity provider and a service provider, or any other type of claims as required by business processes. The tokens can also be authenticated and encrypted.[12]

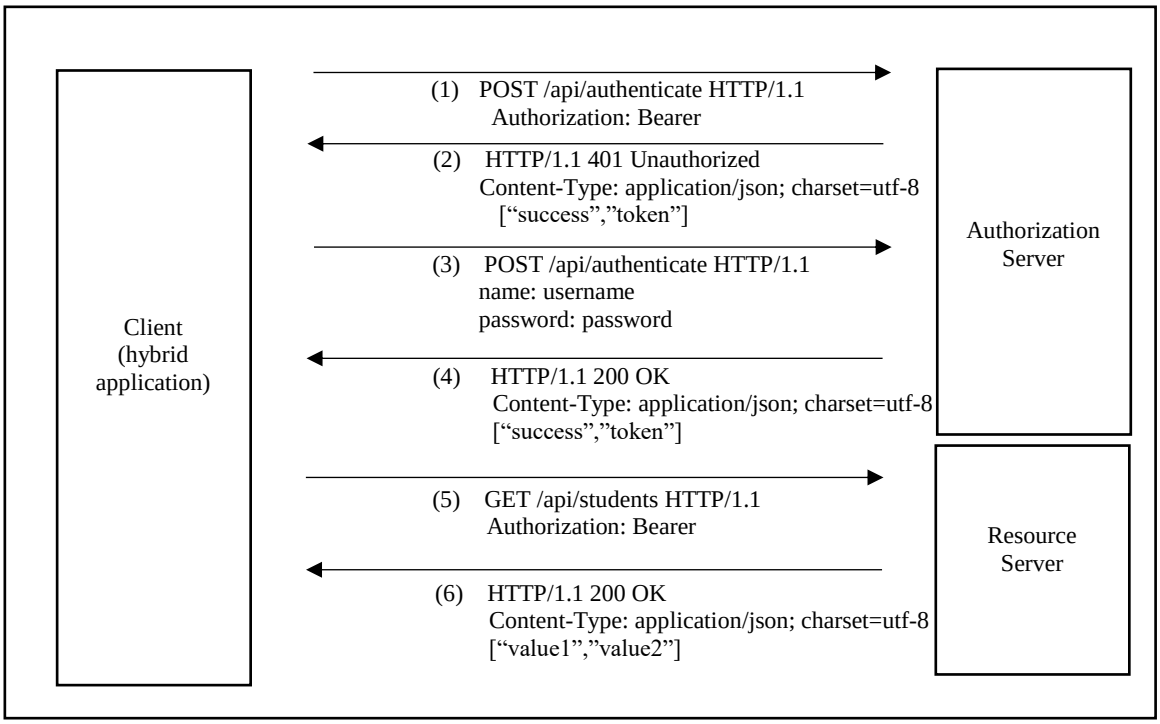


Image 23. The process of obtaining and using a JWT token.

(1) Client does a request to obtain a JWT token from the authorization server with invalid credentials

(2) Authorization server checks the credentials and sees that they are invalid. The server responses that the Authentication failed.

(3) Client does a request to obtain a JWT token from the authorization server with valid credentials.

(4) Authorization Server checks the credentials and sees that they are valid. The server responses with a token.

(5) Client does a request to get data from the resource server with a valid token in the header.

(6) Resource server replies with the requested data in JSON format.

Here is an example of a JWT token:

eyJhbGciOiJIUzI1NiIsdnR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIisibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.TJVA95OrM7E2cBab30RMHreDcEfxjoYZgeFONFh7HgQ

The token consists of three parts:

header.claims.signature

All the parts of the token are base64 URL-safe encoded to make them safe to add in urls.

Header

The header tells us that the token is of the type JWT and that the used algorithm to generate the signature is AES256. [13]

```
{
   "alg" : "AES256",
   "typ" : "JWT"
}
```

Claims

This is the core of the token. It contains all information about the user that we want to transfer between parties. We could authenticate with one authentication server and access protected resources on the resource server. Or perhaps the API issues the token and we store the token on our application. [13]

```
{
  "sub": "1234567890",
  "name": "ruben.alliet",
  "admin": true
}
```

Signature

The signature is generated with the private key to hash the content of the header and claims.
Only the original token will match the signature. So only applications with the appropriate token will match the signature.[13]

4.3.1.3.Project purpose

In the project, the API will be used for reading and manipulating the attendance data of students. When the student walks by an iBeacon™ an API request will be sent. Then the API will reply with a JSON format response to show what the result of the request is. If the student searches for their attendance history the same process repeats itself. The API will also ensure that unauthorized users don't get access to the data on the DB.

The API has the following features:

- Authentication (api/authentication)
- Get data of all students (api/students)
- Get data of all courses of a student (api/students/:userId/courses)
- Get data of all lessons of a student (api/students/:userId/courses/:courseId/lessons)
- Get data of all exams of a student (api/students/:userId/courses/:courseId/exams)
- Set student on present for exam (api/students/:userId/courses/:courseId/exams/:examId)
- Set student on present for lesson (api/students/:userId/courses/:courseId/lessons/:lessonId)
- Set student on present with the UUID of the beacon (api/students/:userId/beacon/:beaconId)

4.3.1.4.folder structure

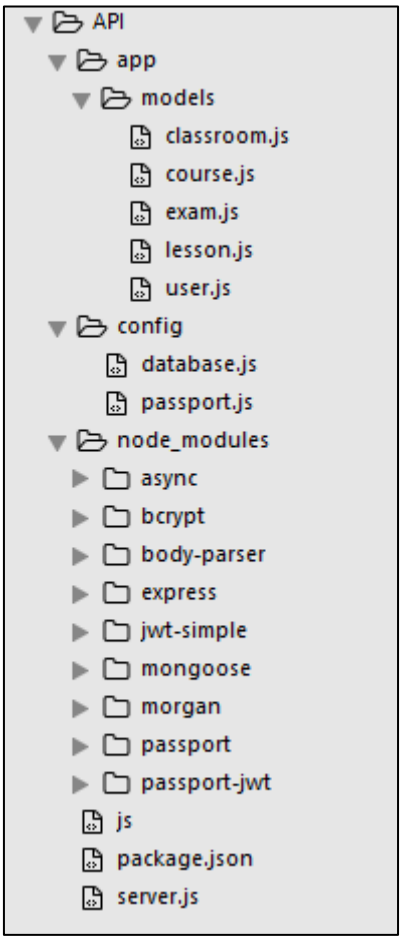


Image 24. folder structure of the API.

In this part I will discuss the directory structure of the API to gain the necessary understanding of its operation.

The ‘app’ directory contains a folder called ‘models’ in this directory each collection of the database has a model. All functions and all properties with their data types are written in this file.

In the ‘config’ directory we can find database.js, this file contains the necessary data to login onto the MongoDB server. Passport.js serves as the authentication for users by linking the right user to the right token.

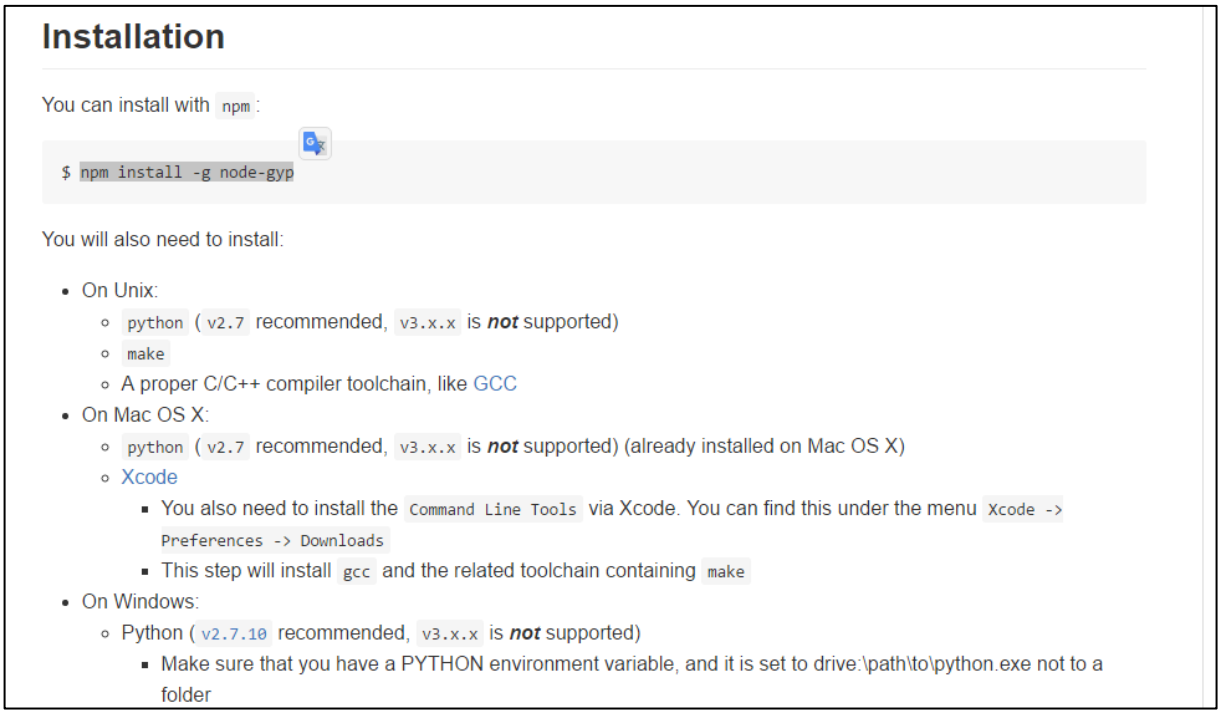
The folder ‘node_modules’ contains all necessary libraries for building the API.

Passport library is used for authentication, mongoose for the connection with the database, bcrypt for the hashing of the password, async for making use of powerful functions within asynchronous JavaScript…

The message that is written down in the file called ‘js’ will be shown when the server is started. Package.json is a JSON formatted file that contains all the names of the libraries and their specific version that needs to be installed. And as last, Server.js contains all the main code of the API, it includes all API routes and their corresponding code.

4.3.1.5.Implementation

I started writing my API by using the example from https://devdactic.com/restful-api-user-authentication-1/ it was very useful in order to understand the operation of the API.
I used the existing API as a source and customized it according to the needs of the project.
In the beginning of the development process of the API there were some package dependency conflicts. The problem was that the bcrypt package couldn't be installed because of the missing node-gyp package. Python 2.7 needed to be installed on the Windows server to avoid errors when trying to install the node-gyp package.



Installation

You can install with `npm`:

```
$ npm install -g node-gyp
```

You will also need to install:

- On Unix:
  - `python` (`v2.7` recommended, `v3.x.x` is ***not*** supported)
  - `make`
  - A proper C/C++ compiler toolchain, like GCC
- On Mac OS X:
  - `python` (`v2.7` recommended, `v3.x.x` is ***not*** supported) (already installed on Mac OS X)
  - Xcode
    - You also need to install the `Command Line Tools` via Xcode. You can find this under the menu `Xcode -> Preferences -> Downloads`
    - This step will install `gcc` and the related toolchain containing `make`
- On Windows:
  - Python (`v2.7.10` recommended, `v3.x.x` is ***not*** supported)
    - Make sure that you have a PYTHON environment variable, and it is set to drive:\path\to\python.exe not to a folder

Image 25. Installation requirements npm node-gyp package.

In the current state of the development of the API, all necessary packages are installed. Now the coding can start.

A model needs to be created for each collection in the database. This model contains properties of each collection with their datatypes and the associated functions with this model.

```
var mongoose = require('mongoose');
var Schema = mongoose.Schema;
var ObjectId = Schema.ObjectId;


// set up a mongoose model
var ClassroomSchema = new Schema({
    _id : Schema.ObjectId,
    building : String,
    name : String,
    minor : Number,
    major : Number,
    beacon_uuid : String
});

module.exports = mongoose.model('Classroom', ClassroomSchema);
```

Image 26. Example of a model in node.js.

In the file server.js all API routes are defined as you can see on the image below:

```
// route to all the students (GET http://localhost:8080/api/students)
apiRoutes.get('/students', passport.authenticate('jwt', { session: false }), function(req, res) {
    var token = getToken(req.headers);
    if (token) {
        var decoded = jwt.decode(token, config.secret);

        // find each person with a last name matching 'Ghost'
        var query = User.find({ 'user_role': 'student' });

        // selecting the `name` and `occupation` fields
        query.select('_id name  ');

        // execute the query at a later time
        query.exec(function(err, students) {
            if (err) throw err;

            if (!students) {
                return res.status(403).send({ success: false, msg: 'Authentication failed. User not found.
            } else {
                res.json(students);
            }
        });
    } else {
        return res.status(403).send({ success: false, msg: 'No token provided.' });
    }
});
```

Image 27. Example of an API route that collects all the names of the students.

When a request is done to a route of the API then the passport will check the JWT token. If the header of the request contains a valid token then the corresponding code will be executed. In this case all the names of the students would be queried out of the database. And it would be sent in JSON format to the client.

4.3.1.6.Authentication

When the user tries to login on to the application, following method is called and executed:

```
UserSchema.methods.comparePassword = function (passw, cb) {
    bcrypt.compare(passw, this.password, function (err, isMatch) {
        if (err) {
            return cb(err);
        }
        cb(null, isMatch);
    });
};

module.exports = mongoose.model('User', UserSchema);
```

Image 28. ComparePassword method in the user model.

It checks whether the entered password is correct, I have used bcrypt, a password hashing function to hash all the passwords. The prefix (1) "$2a$", "$2b$" or "$2y$" in the hash string from the database indicates that the hash string is a bcrypt hash.
The rest of the hash string includes the cost parameter (2), a 128-bit salt which is base-64 encoded and consists out of 22 characters. The remaining 184 bits present the resulting hash value (3) which is also base-64 encoded and consists out of 31 characters. The cost parameter represents a key expansion iteration count as a power of two, which serves as an input for the crypt algorithm. [14]
For example, the shadow password record:

$2a$10$N9qo8uLOickgx2ZMRZoMyeIjZAgcfl7p92ldGxad68LJZdL17lhWy
(1) prefix (2) Cost parameter (128 bit) (3) Hash value(184 bits)

When the comparePassword method finds a match it calls the callback function, with as parameter that there is a match, otherwise the callback function will be called the error as parameter.

```
// check if password matches
user.comparePassword(req.body.password, function(err, isMatch) {
    if (isMatch && !err) {
        var token = jwt.encode(user, config.secret);
        // if user is found and user role is teacher generate student token
        if (user.user_role == 'student') {
            res.json({ success: true, token: 'JWT ' + token + '*student' });
        // if user is found and user role is teacher generate teacher token
        } else if (user.user_role == 'teacher') {
            res.json({ success: true, token: 'JWT ' + token + '*teacher' });
        } else {
            res.json({ success: false, msg: 'Authentication failed. User does not have a role' });
        }
    } else {
        res.send({ success: false, msg: 'Authentication failed. Wrong password.' });
    }
});
```

Image 29. Corresponding code of the API route /authenticate.

When the authentication process ended successfully, it will give back a token. The type of token depends on the user role. Each user role has its own privileges:

Teacher

- Put each student on present for a lesson or exam manually
- Check the attendance history of each student in lessons or exams

Student

- Set own attendance for a lesson or exam by using IBeacon™ proximity
- Check own attendance history in lessons or exams

The obtained token will be included in the header of each API request. Confidentiality is now ensured.

4.3.2.Database: Mongo DB

4.3.2.1.What is it?

MongoDB is a NoSQL database, it has no traditional table-based relational database structure. In fact it has JSON-like documents(BSON) with dynamic schemas. This format makes the integration of data in certain types of applications more efficient. MongoDB is also free and open-source. Relationships are made in the code. On the chart on the right we can see that the Mongo handles the basic queries much quicker than SQL. Another benefit of Mongo is that you can keep your data in its native JSON format, no translation is necessary. [15] This is why our API is written in Node.JS.

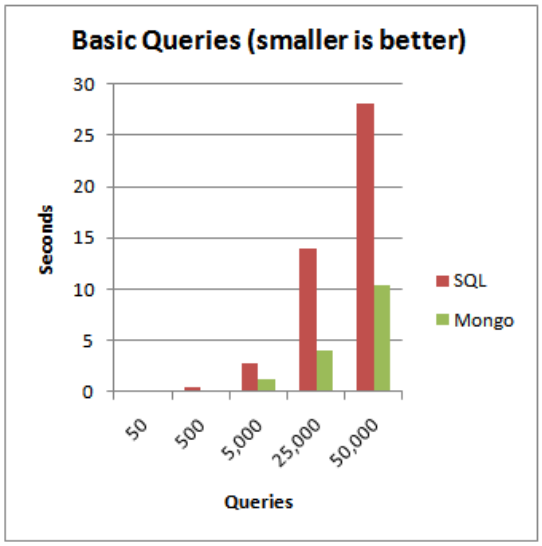


Image 30. Basic queries on SQL and Mongo ,x-axis: amount of seconds, y-axis: amount of requests.

4.3.2.2.Installation

Log in on the Windows server. Download the msi installation package of MongoDB and follow the wizard. Open the command prompt with administrator privileges and execute the following command line instructions[16]:

```
cd /
cd mongodb/bin
mongod.exe
```

Image 31. Startup MongoDB server.

Now MongoDB is running and connections can be established by the API. MongoDB can be shut down by closing the command or with ctrl + c.

4.3.2.3.Implementation

First we start up our MongoDB server. To know how to start up the server, check the previous paragraph called ‘Installation’. Now we can open the program RoboMongo and create the following five collections with their associated properties:

- Classrooms
  - _id — id of the classroom
  - building — name of the building where the classroom is located
  - name — name of the classroom
  - minor — default value 0 because not used. Intended to identify and distinguish an individual iBeacon™
  - major — default value 0 because not used. Intended to identify and distinguish a group of iBeacon™
  - beacon_uuid — unique id of the beacon in the classroom

- Courses
  - _id — id of the course
  - field_of_study — field of study of that particular course
  - year — which year does the course take place
  - name — name of the course
  - lessons[ ] — an array that contains all the lesson id’s of a course
  - exams[ ] — an array that contains all the exam id’s of a course

- Exams
  - _id — id of the exam
  - name — name of the exam
  - start — start date and time of the exam
  - end — end date and time of the exam
  - classroom — contains the classroom id
  - users_present[ ] — an array that contains all the users

- Lessons
  - _id — id of the lesson
  - name — name of the lesson
  - start — start date and time of the lesson
  - end — end date and time of the exam
  - classroom — contains the classroom id
  - users_present[ ] — an array that contains all the users

- Users
  - _id — id of the user
  - email — email of the user
  - password — encrypted password of the user
  - name — name of the user
  - user_role — user can have the role of a student or a teacher
  - courses[ ] — all the courses that a student is participating in
  - lessons_present[ ] — all the lessons where the student was present
  - exams_present[ ] — all the exams where the student was present

All the data is added in BSON format here an example:

```
{
   "_id" : ObjectId("57376465bf2537f1e69f941a"),
   "building" : "1B",
   "name" : "Laboratorio DSIC8",
   "minor" : 0,
   "major" : 0,
   "beacon_uuid" : "e2c56db5-dffb-48d2-b060-d0f5a71096e0"
}
```

Image 32. BSON format in MongoDB.

All the created collections are linked to each other with their unique object id. As you can see on the example below:

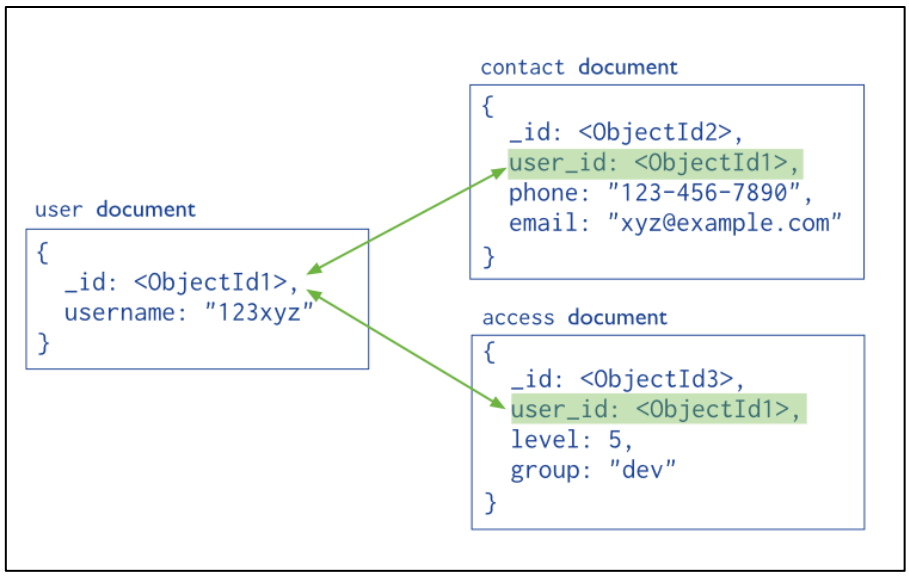


Image 33. Many to Many relationships in MongoDB.

Relationships are obtained by coding in the API. Although MongoDB is no relational database, there are still more or less relationships possible by coding.

5.Testing and Maintenance

In this part I will talk about tools and strategies that I have used to make maintenance and future implementations more easy.

5.1.Documentation

Both API and the application are provided with the required commentary to facilitate maintenance and possible expansion of the application. After some time you may have forgotten the operation of a particular piece of code. In this case the commentary could be very useful.

5.2.Testing and debugging

Different tools were used for testing and debugging. I will discuss all of them chronological.
First I will talk about Ionic View. This tool was used to share the application layout with others.
Second I will discuss desktop browser testing, most of the time I made use of that tool because it was really quick. As last I also made use of USB debugging which was very useful for testing the plugins and the fine tuning of the application.

5.2.1.Ionic View

Ionic view is an application that gives a preview of your application. All without ever going through the App Store. Ionic View is available in the application store of each platform.

It is a very useful tool to share your app with friends and interact with your application. Two big disadvantages are that debugging is not possible and that plugins don't work. Because the application you see in the preview is just the HTML5 application and not the native one.

5.2.2.Desktop browser testing

Desktop browser testing, another way to test your application. It works as follows:
Open the path of your application in the command prompt and execute the command line instruction 'ionic serve'. Now the browser will start a live-reload server for your project. When changes are made to any HTML, CSS, or JavaScript files, the browser will automatically reload your project.
Unfortunately with desktop browser testing it is also not possible to test the plugins. But it has the advantage that the testing is much faster than with Ionic View and debugging is possible.

5.2.3.USB debugging

The last way to test our application makes it possible to test our plugins in contrast to the other tools. My phone is a OnePlus x, the standard USB driver of OnePlus doesn't support USB debugging so I needed to follow next steps[17]:

- Install the latest Samsung drivers on your computer (SAMSUNG USB Driver v1.5.33.0)
- Restart the computer
- Go to Device Manager, find the Android device, and select Update Driver Software.
- Select Browse my computer for driver software
- Select Let me pick from a list of device drivers on my computer
- Select ADB Interface from the list
- Select SAMSUNG Android ADB Interface

After installing the right USB driver, we will need to enable USB debugging on our device. This can be done as follows[18]:

- Settings > About phone and tap Build number seven times.
- On your Android device, select Settings > Developer options.
- In Developer options, select the USB debugging checkbox
- Connect the Android device to your development machine using a USB cable.
- navigate to chrome://inspect. Confirm that Discover USB devices is checked
  To start debugging, click inspect

Now we can set checkpoints in our code and start debugging our application.

5.3.Publishing

In this part of the essay I will first discuss the hosting of the database and the API. And then the availability of the application.

5.3.1.API & database

I used a virtual windows server hosting both API & database that can be managed from the URL https://vmcloud.ikdoeict.be/cloud/org/kwaliteitskunde/. To visit this URL, first you need to connect with the VPN of Odisee university college and then you need to login with the student credentials. If the application wants to connect to the server to perform API requests, it also needs to make a connection with that same VPN.

5.3.2.Application

If the project idea would be approved, the application would be available on Google Play, App Store and Windows Phone Store.

In the meantime my application is installed via USB debugging. I open the path of my application in the command prompt and then I execute the command line instruction ‘ionic run android’. The HTML application will be compiled to a native Android application. And then it will be installed. An important side note, we always need to add the platforms to the application that we want to target. For example if I want to target Android ,IOS and Windows then I will need to execute first the following three command line instructions:

- Ionic platform add ios
- Ionic platform add android
- Ionic platform add windows

6.Used Tools and programs

In this part I will discuss the most important tools and programs that I have used for this project. For each tool I will explain what it is. Then the purpose where I have used the tool for and as last I will discuss the advantages and disadvantages of it.

6.1.Vysor

Vysor is a Chrome app that allows screen casting from your android device to your computer. If you want to use it you need to turn on USB debugging on your phone or tablet, and connect to the computer. Once ADB(Android Debug Bridge) connects to your machine and Vysor is installed, your mobile screen will show up on your desktop. Your mouse and keyboard can now control your Android device. Including keyboard shortcuts for back, home and multitasking. The application is compatible with Mac and Windows.[19]

In my case I used it to enhance development speed. Because I didn't need to switch between the smartphone and computer, everything was centralized. It is also very useful tool to give a demo for a great amount of people.

Advantages	Disadvantages
User-friendly	Limited image quality
Very quick installation	Wireless screen casting not possible

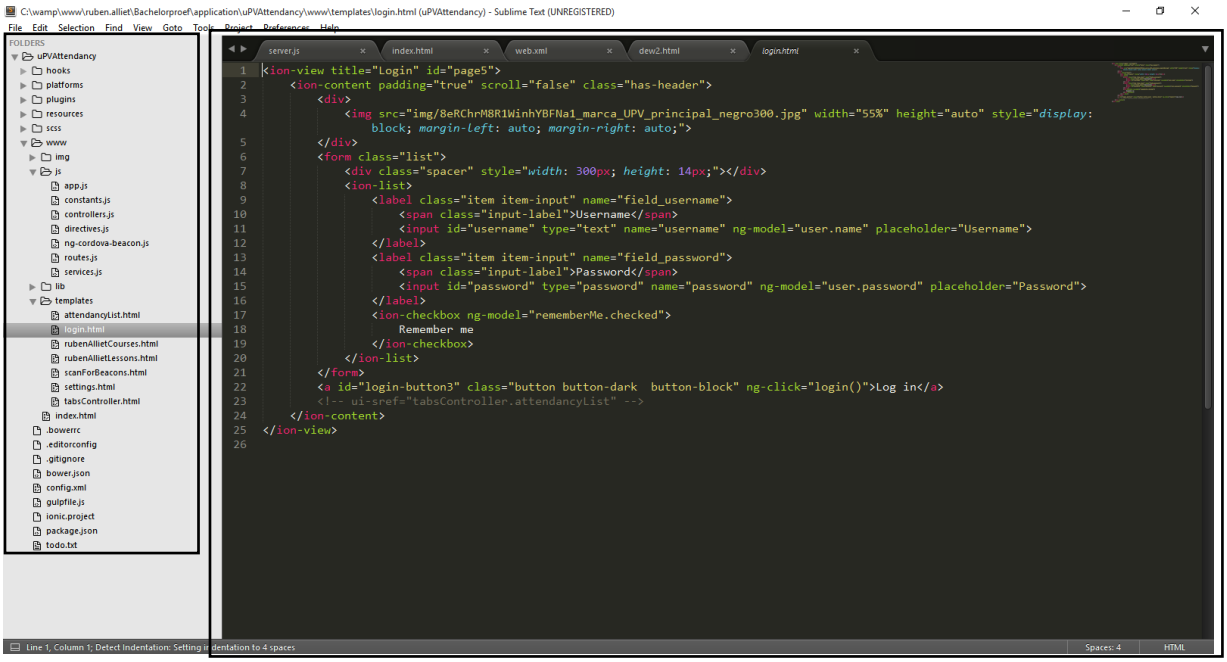
Table 4. Advantages & disadvantages of Vysor.

6.2.Sublime Text

Sublime Text is a cross-platform source code editor that natively supports many programming languages and mark-up languages. Its functionality can be extended by users with plugins, typically community-built and maintained under MIT licenses.

First I have installed package control to be able to install other packages. I have used a package for syntax highlighting and another one that checks the indentation of your code and fixes it if necessary(HTML-CSS-JS prettify). The API and application are created using this code editor.

1



2

Image 34. GUI of Sublime text.

1. Project folder structure
2. Code editor view, each file presents a tab

Advantages	Disadvantages
User-friendly	No syntax correction
Syntax highlighting	No debugging
Extensibility	Limited syntax autocomplete
Performance	

Table 5. Advantages and disadvantages of Sublime text

6.3.RoboMongo

Robomongo is a shell-centric cross-platform(which means support for Windows,Linux and Mac) MongoDB management tool. In contrast to other MongoDB tools, Robomongo embeds the actual mongo shell in a tabbed interface with access to a shell command line as well as GUI interaction. As a user you always have the ability to choose.

In my case I've used this tool for creating and populating my database design and also for monitoring the manipulated objects.

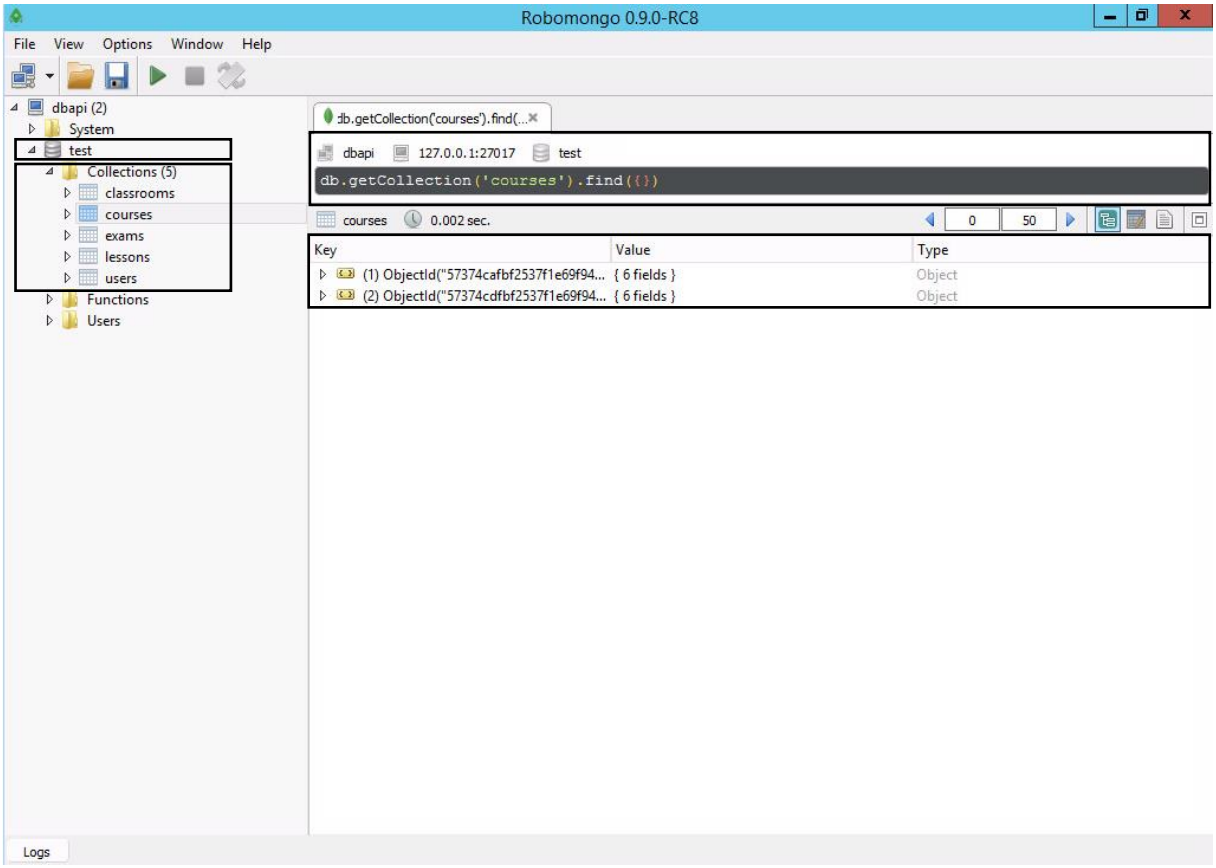


Image 35. GUI of RoboMongo.

1. Database
2. Collections in Database
3. Mongo shell (shell command line)
4. Mongo shell(GUI)

Advantages	Disadvantages
Mongo shell command line and GUI	Error handling when wrong data is added
Quick interaction	Adding data should be easier

Table 6. Advantages & disadvantages of RoboMongo

6.4.Postman

Postman is a REST Client that runs as an application inside the Chrome browser. It is very useful for interfacing with REST API's. The most important features are:

- History of sent requests
- Create requests quickly
- Replay and organize
- Switch context quickly
- Customize with scripts
- Robust testing framework
- Automate collections

In my case I have used this tool to test my API and possibly find some errors or shortcomings.
I could save all my API requests during the testing phase of the project which was very useful for repetitive tasks.

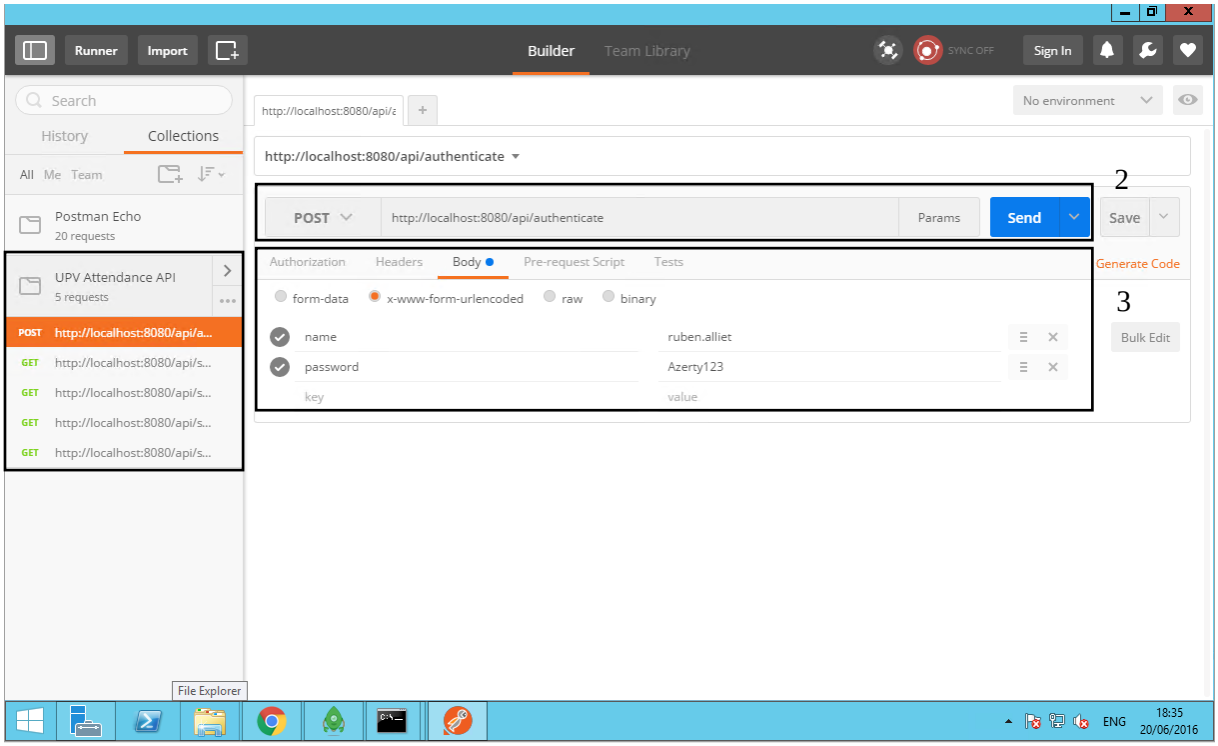


Image 36. GUI of Postman.

1. Collection of stored API requests
2. API Request with URL 'http://10.129.32.5:8080/api/authenticate' and method POST
3. Possible parameters that can be passed with the API request

Advantages	Disadvantages
Structure of the GUI	Only available in google chrome
Very quick installation	

Table 7. Advantages and disadvantages of Postman

6.5.mremoteNG

mremoteNG is a Remote Desktop manager that centralizes all your remote connections.
It is a fork of mRemote, an open source, tabbed, multi-protocol, remote connections manager.
The program supports the following protocols:

- RDP (Remote Desktop/Terminal Server)
- VNC (Virtual Network Computing)
- ICA (Citrix Independent Computing Architecture)
- SSH (Secure Shell version 1 & 2)
- Telnet (TELecommunication NETwork)
- HTTP/HTTPS (Hypertext Transfer Protocol)
- rlogin
- Raw Socket Connectionse connections in a simple yet powerful tabbed interface.

I have used this tool for the development of the API on the windows server. To connect with the windows server I needed to use the RDP(Remote Desktop Protocol) protocol. It is a proprietary protocol developed by Microsoft, which provides a user with a graphical interface to connect to another computer over a network connection. The server is located in the network of my university 'Odisee' so it was necessary to establish a VPN connection with my university first.

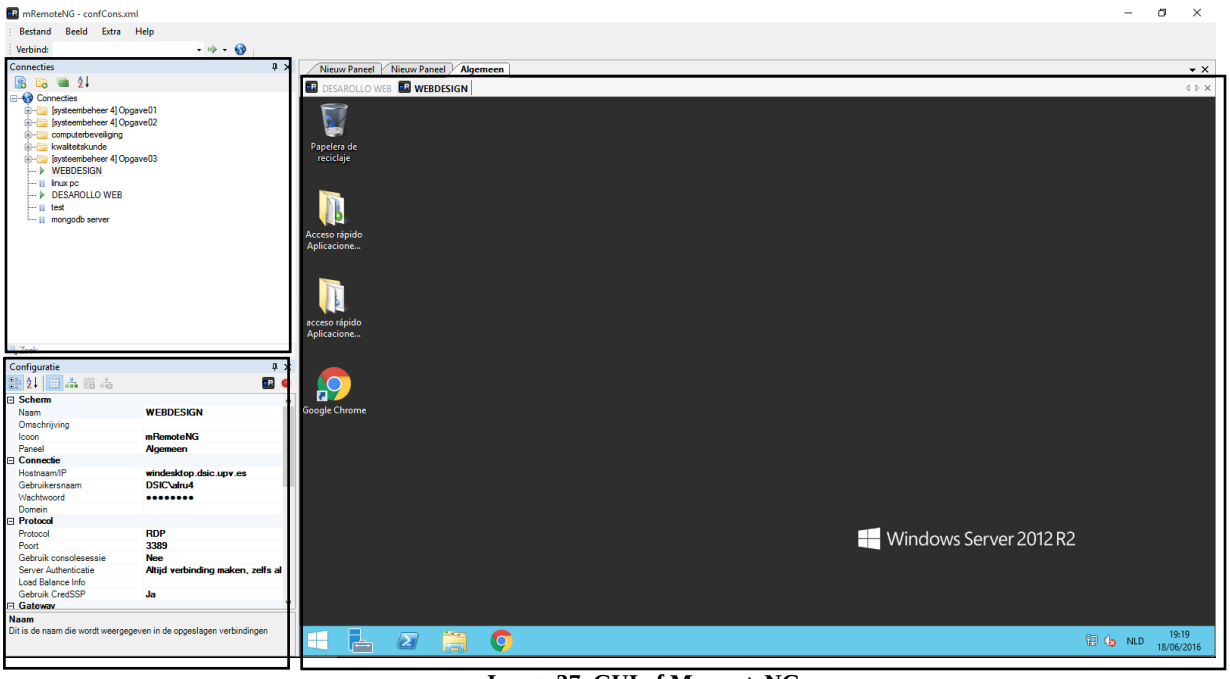


Image 37. GUI of MremoteNG.

1. List of connections
2. Configuration for each connection(protocol,port number,hostname/ip,credentials…)
3. GUI of connected computer

Advantages	Disadvantages
Copy-paste data	Unclear error messages
Quick interaction	
Clear UI structure	

Table 8. Advantages & Disadvantages of Postman.

6.6.Ionic Creator

Ionic Creator is a drag-&-drop prototyping tool for creating hybrid apps using Ionic.

You just start with visually building your app by dragging mobile components (1) into the device emulator (2) or add HTML. We can change the properties of our page or mobile components depending on what you selected, in the pane on the right(3). It is also possible to preview and interact with your projects on the device emulator. If we want to design another page we just click on a different item in the list view on the panel in the upper left corner(4). We can also change the selected platform by opening the dropdown in the top bar of the screen(5). For sharing your app with colleagues and clients we can generate a URL.

When the prototyping is done we can export a fully functional Ionic project, or even a native package format to install directly on your devices. Be aware this is only a prototype, the appropriate code should be adapted as to be able to work properly. But despite of that, it is a very good basis to make an application

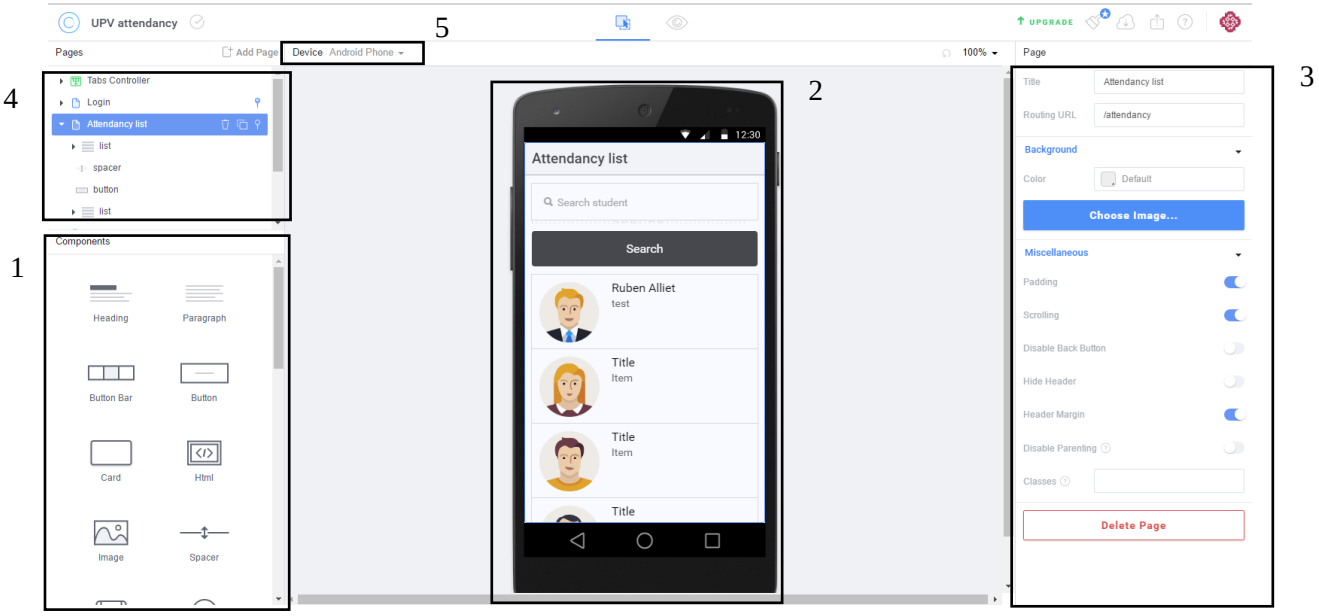


Image 38. GUI of Ionic creator.

1. Available components
2. Design preview
3. Page/Component properties
4. Page overview
5. Change the platform preview

Advantages	Disadvantages
Really quick	Available components
No installation	Only 1 project is free
Application flow overview	

Table 9. Advantages & disadvantages of Ionic creator.

6.7.GIT

Git is a free and open source distributed VCS(Version Control System) designed to handle small to very large projects. But what is a version control system? A version control system is a software tool that helps a software team or an individual manage changes to source code over time. It keeps track of every modification to the code straightforward. in a special kind of database. If a developer makes a mistake, it is always possible to turn back to the previous version. The command line instructions are straightforward and easy to learn. Git also has a tiny footprint and a great performance.[20]

Here is a scheme of how the storage of Git works:

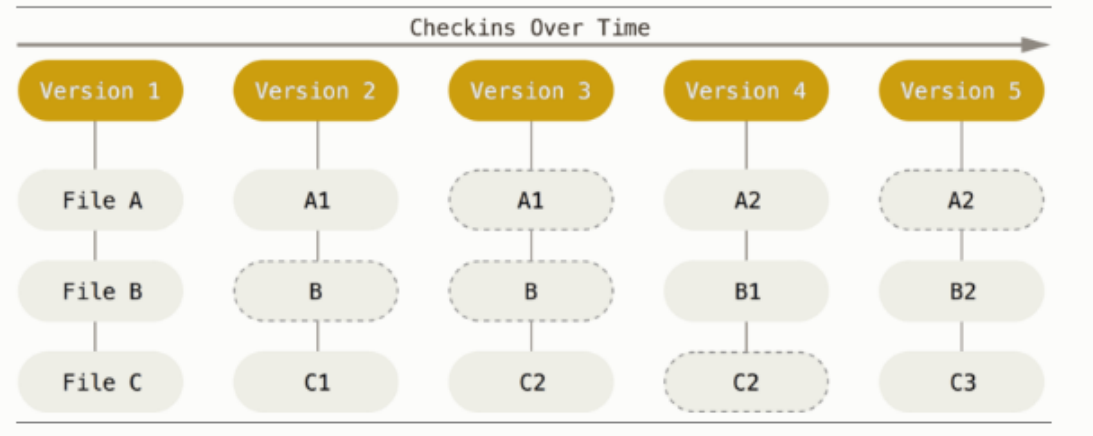


Image 39. Git storage strategy.

Storing of data happens when changes are made to a base version of a file.

Each time a commit happens, Git basically takes a snapshot of what all your files look like at that moment and stores a reference to it. When files have not changed, anything is stored, just a link to the previous identical file that already has been stored.

I have chosen for git because it is very well integrated in companies. So it thinks it's very useful to understand this tool and be able to work with it. Following command line instructions were very useful during the project:

Git global setup
git config --global user.name "Ruben Alliet"
git config --global user.email "ruben.alliet@student.odisee.be"
Create a new repository
git clone git@git.ikdoeict.be:ruben.alliet/Bachelorproef.git
cd Bachelorproef
touch README.md
git add README.md
git commit -m "add README"
git push -u origin master
Existing folder or Git repository
cd existing_folder
git init
git remote add origin git@git.ikdoeict.be:ruben.alliet/Bachelorproef.git
git add .
git commit

Check appendix C to see the meaning of those command line

Image 40. Basic git commands.

In my case I have used git to store all the code of my project, the versioning of the code was very useful because at a certain point in time I was stuck with a certain piece of code, so I returned to a previous version of the code which was very handy for solving the problem I had to deal with.

Advantages	Disadvantages
Quick data transfer	Merge conflicts when working in group
Easy to learn	
Extensibility and statistics of your project	

Table 10. Advantages & disadvantages of GIT

7.Future implementations

The current implementation also has some shortcomings like a high implementation cost, lack of data centralization and user confidentiality. Therefore several solutions will be discussed in this part of the essay. As first NFC could solve the high implementation cost. As second data exchange with Google Calendar could solve the lack of data centralization. And as last the fingerprint login could solve the user confidentiality issue.

7.1.NFC

NFC stands for near field communication. This technology enables two electronic devices, one of which is usually a portable device such as a smartphone, to establish communication by bringing them within a range of 4 cm of each other.[21]

In the current implementation we have used iBeacons™ to determine whether or not the student is present in class. But buying iBeacons™ for each classroom is not very cheap. That's why the following idea could be interesting:

Each student card contains an NFC chip, this chip would communicate with the smartphone of the teacher. After each lesson all the students would go to the teacher and put their card within a range of 4 cm of the teacher's smartphone to prove their attendance in class. If the student forgot his or her student card, the teacher can always set the attendance manually with the application.

7.2.Data exchange with Google Calendar

Google Calendar is a widely used tool for planning your activities during the day. In the current situation a student can see the attendance status of all his or her lessons and exams on this application. The idea was to change the colour of a lesson or exam in the app depending on the status of it. Green would mean successfully scanned in. Yellow would mean not scanned in yet and red would say that the student was too late. For the colour blind there would be a small note with the status for each exam and lesson. To implement this idea the existing attendance API should be adjusted to be able to communicate with Google Calendar. If this is implemented then it would be no longer necessary to open the UPV attendance application and view the attendance history of your lessons and exams.

7.3.Fingerprint login

The current implementation of the project has a significant shortcoming. Student X can check-in student Y if student X knows the credentials of student Y. This problem could be solved by implementing a login that is not based on credentials but on biometrics. However a fingerprint sensor is necessary on your smartphone. There is a library that verifies fingerprints that can be find on the link 'https://github.com/leecrossley/cordova-plugin-touchid.git'. The system would work as follows. If the user wants to login on the application he needs to put his thumb on the fingerprint sensor of the smartphone. The application will give feedback if the login succeeded or failed. User confidentiality is now ensured.

8.Conclusion

A cross platform application that allows students to prove their attendance in lessons and exams implemented. An iBeacon™ has been successfully configured to allow identification of the classroom. Also, a database has been created to store all the student attendances. Finally an API that allows manipulating and reading attendance data out of the database has been created.

This prototype has been tested in a small simulated environment. I came to the conclusion that it is a very effective way to take attendances of students. It could save lots of time for students, teachers and the staff responsible for scanning and processing the attendance sheets every day. For this reason, it is also a more productive solution in terms of human resources. In addition, it is also a more ecological solution as it saves paper.

Used technologies

- Hybrid application (Cordova)
- API (Node.js)
- Database (MongoDB)

Working of the process

- Each classroom has an iBeacon™, when the student enters the classroom and comes within the radius(15M) of the iBeacon™, the smartphone will receive the UUID of the iBeacon™

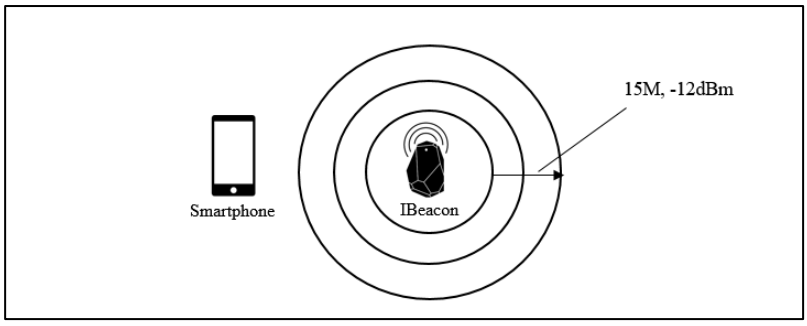


Image 3. IBeacon™ proximity

- The UUID of the iBeacon™ and the id of the user will be used by the application(IOS, Android or Windows) to send an API request to the Windows Server

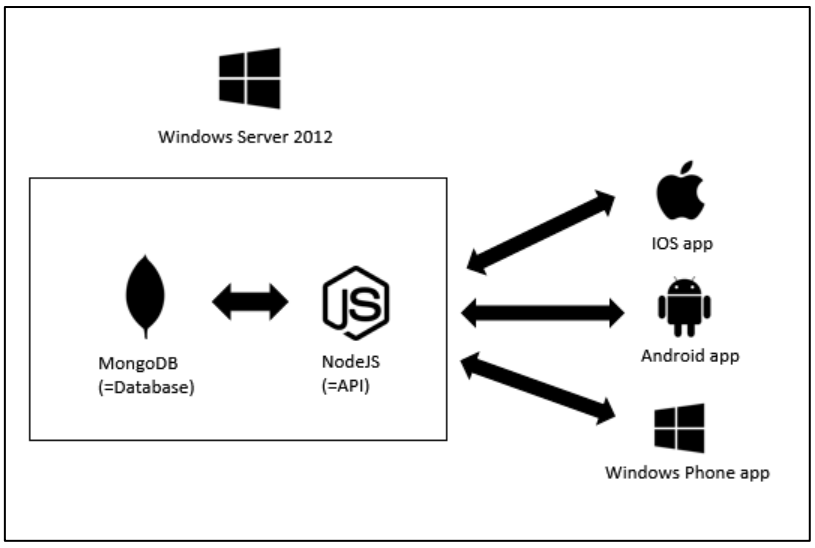


Image 2. Architecture diagram.

- Then the API will use these two parameters to set the attendance for the student's exam or lesson in the database.

- If the IBeacon™ is broken or the battery is dead, the teacher can always set the attendance of a student manually
- In addition, it is also possible to view your own attendance history as a student or the one of other students, the latter only if you are a teacher.

Epilogue

This technology could be used in a real school context to improve the current process for checking class attendance. The technology could also be applied for other purposes like conferences, office…

It was a very interesting project, where many different technologies needed to be combined. Also many problems needed to be solved which was very defiant. I have discovered many new technologies and their associated advantages and disadvantages. Making this project helped me in becoming more independent. I myself am very satisfied with the result.

9.Acronyms and initialisms

API = Application Programming Interface

DB = Database

BLE = Bluetooth Low Energy

PHP = Hypertext Preprocessor

GIT = source control management

IDE = integrated development environment

NFC = near field communication

JSON = JavaScript Object Notation

RDP = Remote Desktop Protocol

VNC = Virtual Network Computing

ICA = Citrix Independent Computing Architecture

SSH = Secure Shell

SDK = Software Development Kit

Telnet = TELecommunication NETwork

HTTP = HyperText Transfer Protocol

HTTPS = HyperText Transfer Protocol Secured

HTML = HyperText Markup Language

CSS = Cascading Style Sheets

JS = JavaScript

JWT=Jason Web Token

npm = node.js package manager

NoSQL = Non Structured Query Language

10.Bibliography

[1] Anonymous, “iBeacon™”, https://en.wikipedia.org/wiki/IBeacon™ [Online]. 21 June 2016, at 12:35

[2] Simon Toulson, “iBeacon™ Parameters: UUID, Major and Minor”,https://kontakt.io/blog/iBeacon™-configuration-guide-tranmission-power/ [Online]. 04 June 2016, at 03:06

[3] Virinchy P , “What is Cordova and how does it work ?”,http://scn.sap.com/community/developer-center/mobility-platform/blog/2014/07/27/what-is-cordova-and-how-does-it-work [Online]. 27 July 2014, at 15:27

[4] Anonymous, “Chapter 1: All About Ionic” ,http://ionicframework.com/docs/guide/preface.html [Online].

[5] Anonymous, “What Is Angular?”, https://docs.angularjs.org/guide/introduction [Online].

[6] Anonymous, “Start building with Ionic!”, http://ionicframework.com/getting-started/ [Online].

[7] Anonymous, “Structure”, http://ionicframework.com/docs/concepts/structure.html [Online].

[8] Abdullah Buhadod, "Ionic Framework Tutorial : Implement List With Search Feature", https://www.youtube.com/watch?v=JSAdFuTYYe4 [Online]. 3 July 2015

[9] Simon R, “User Authentication With AngularJS For Your Ionic App”, https://www.youtube.com/watch?v=PE81fyfEJUA [Online]. 22 April 2015

[10] Anonymous, “Application programming interface” , https://nl.wikipedia.org/wiki/Application_programming_interface [Online]. 29 May 2016, at 21:57

[11] Gerard Sychay, “TOP 10 REASONS TO USE NODE.JS”, http://blog.modulus.io/top-10-reasons-to-use-node [Online]. 10 september 2014

[12] Anonymous, “JSON Web Token” , https://en.wikipedia.org/wiki/JSON_Web_Token [Online]. 29 May 2016

[13] Simon, “OAuth 2 VS JSON Web Tokens: How to secure an API” , http://www.seedbox.com/en/blog/2015/06/05/oauth-2-vs-json-web-tokens-comment-securiser-un-api/ [Online]. 5 march 2015

[14] Anonymous, “bcrypt”, https://en.wikipedia.org/wiki/Bcrypt [Online]. 24 June 2016, at 22:28

[15] Anonymous, “MongoDB” ,https://en.wikipedia.org/wiki/MongoDB [Online]. 20 June 2016, at 19:20

[16] Anonymous, “Install mongodb on windows”,https://docs.mongodb.com/v3.0/tutorial/install-mongodb-on-windows/[Online].

[17] Shubham Upadhyay, “What should I do if my OnePlus One is not in 'active devices' in Android studio?”, https://www.quora.com/What-should-I-do-if-my-OnePlus-One-is-not-in-active-devices-in-Android-studio [Online]. 8 February 2015

[18] Anonymous, “Remote Debugging on Android with Chrome”, https://developer.chrome.com/devtools/docs/remote-debugging [Online].

[19] Anonymous, “Vysor”, http://www.vysor.io/ [Online].

[20] Davide Salvador, “Git Intermediate Workshop slides v1.3”, http://www.slideshare.net/DavideSalvador/corso-git-materiale-di-presentazione [Online].

[21] Cameron Faulkner, “What is NFC? Everything you need to know”,http://www.techradar.com/news/phone-and-communications/what-is-nfc-and-why-is-it-in-your-phone-948410[Online]. 17 November 2015

Images

Image 1: iBeacon™ icon,
https://d30y9cdsu7xlg0.cloudfront.net/png/55198-200.png;
Smartphone icon,
https://image.freepik.com/free-icon/smart-phone-with-white-screen_318-35639.jpg;
API icon,
http://saffwood.co.uk/images/APICloud.png;
Database icon,
https://cdn1.iconfinder.com/data/icons/universal-mobile-solid-icons-vol-4/48/185-512.png

Image 2: Windows icon,
https://cdn3.iconfinder.com/data/icons/picons-social/57/32-windows8-128.png;
MongoDB icon,
http://plainicon.com/dboard/userprod/2800_a1826/prod_thumb/plainicon.com-50274-512px-31d.png;
Node.JS,
http://findicons.com/files/icons/2773/pictonic_free/512/prog_nodejs02.png;
Apple,
https://cdn1.iconfinder.com/data/icons/simple-icons/4096/apple-4096-black.png;
Android,
http://simpleicon.com/wp-content/uploads/android.svg

Image 3: iBeacon™,
https://d30y9cdsu7xlg0.cloudfront.net/png/55198-200.png;
Smartphone,
https://image.freepik.com/free-icon/smart-phone-with-white-screen_318-35639.jpg

Image 7: https://build.phonegap.com/images/marketing/build-diagram.png

Image 8: Original image,
http://scn.sap.com/servlet/JiveServlet/downloadImage/38-111196- 506707/620-348/Cordova_framework.png;
Iphone,
https://d3nj7353mvgauu.cloudfront.net/media/categories/iphone-6-6s-6-plus-6s-plus-1.png;
Android,
https://cdn4.iconfinder.com/data/icons/smart-phones-technologies/512/android-phone-color.png;
Windows,
http://img03.deviantart.net/85d2/i/2013/134/4/1/lumia_521_psd_template_by_saphfan32-d65af5y.png

Image 10-20: Android,
https://cdn4.iconfinder.com/data/icons/smart-phonestechnologies/512/android-phone-color.png

Image 21: Smartphone, https://image.freepik.com/free-icon/smart-phone-with-white-screen_318-35639.jpg; API, http://saffwood.co.uk/images/APICloud.png; Database, https://cdn1.iconfinder.com/data/icons/universal-mobile-solid-icons-vol-4/48/185-512.png

Image 22: http://hostingadvice.digitalbrandsinc.netdna-cdn.com/wp-content/uploads/2015/03/nodejs-vs-php-performance-requests-per-second.png

Image 23: https://lbadri.files.wordpress.com/2012/09/untitled4.png?w=700

Image 25: https://github.com/nodejs/node-gyp

Image 30: https://michaelckennedy.files.wordpress.com/2014/03/basic-indexed-query-speed-graph.png?w=352&h=359

Image 33: https://docs.mongodb.com/manual/core/data-model-design/

Image 39: http://www.slideshare.net/DavideSalvador/corso-git-materiale-di-presentazione

Image 40: https://confluence.atlassian.com/bitbucketserver/basic-git-commands-776639767.html

Tables

Table 3: http://blog.modulus.io/top-10-reasons-to-use-node

Table 11: http://www.warski.org/blog/2014/01/how-iBeacon™s-work/

Table 13: https://confluence.atlassian.com/bitbucketserver/basic-git-commands-776639767.html

11.Appendices

11.1.Appendix A: iBeacon™ abbreviations

Property	Value	Meaning
UUID	UUID stands for Universally Unique Identifier. It contains 32 hexadecimal digits, split into 5 groups, separated by dashes. For example: **f7826da6-4fa2-4e98-8024-bc5b71e0893e**	The purpose of the ID is to distinguish iBeacons™ in your network, from all other beacons
Major	unsigned integer values between 1 and 65535.	Major and Minor values are numbers assigned to your iBeacons™, in order to identify them with greater accuracy than using UUID alone. Major values are intended to identify and distinguish a group – for example all beacons in on a certain floor or room in your venue could be assigned a unique major value
Minor	unsigned integer values between 1 and 65535.	Major and Minor values are numbers assigned to your iBeacons™, in order to identify them with greater accuracy than using UUID alone. Minor values are intended to identify and distinguish an individual – for example distinguishing individual beacons within a group of beacons assigned a major value.
Transmission power	Value measured in dBM (Decibel-milliwatts) and corresponds to a number rating (from 0 to 7) that you change via the Admin App of your beacon.	Your Transmission Power setting determines how powerful the signal will be transmitted by your beacon.
Broadcasting interval	100ms – 1000ms	The frequency of broadcast signals
Connection mode	Yes/No	The ability of an external device to connect with a Bluetooth beacon.

Table 11. iBeacon™ abbreviations.

11.2.Appendix B: iBeacon™ properties

Item	Value	Remarks
Case Color	White, pink, green	Other colors can be customized
Battery Model	1 x CR2477	1pc CR2477 coin battery, 1000mAH, 3.0V
Operation Voltage	1.8-3.6V	DC
Transmission Circuit	10.5mA(Max.)	Tested at 0dBm transmission power
Transmission Range	100 meters	Maximum
Antenna	50ohm	On board / PCB Antenna
Accessories	Double adhesive	1pc, high-strength, 3M brand
Net Weight	21.0g	With battery but without package
Size	Ø36 x 16 mm	Null

Table 12. IBeacon™ properties.

11.3. Appendix C: Basic git commands

Git task	Notes	Git commands
Tell Git who you are	Configure the author name and email address to be used with your commits. Note that Git strips some characters(for example trailing periods) from user.name.	git config --global user.name "Ruben Alliet" git config --global ruben.alliet@student.odisee.be
Create a new local repository		git init
Check out a repository	Create a working copy of a local repository:	git clone /path/to/repository
	For a remote server, use:	git clone username@host:/path/to/repository
Add files	Add one or more files to staging (index):	git add <filename> git add *
Commit	Commit changes to head (but not yet to the remote repository):	git commit -m "Commit message"
	Commit any files you've added withgit add, and also commit any files	git commit -a

	you've changed since then:	
Push	Send changes to the master branch of your remote repository:	git push origin master
Status	List the files you've changed and those you still need to add or commit:	git status
Connect to a remote repository	If you haven't connected your local repository to a remote server, add the server to be able to push to it:	git remote add origin <server>
	List all currently configured remote repositories:	git remote -v
Branches	Create a new branch and switch to it:	git checkout -b <branchname>
	Switch from one branch to another:	git checkout <branchname>
	List all the branches in your repo, and also tell you what branch you're currently in:	git branch
	Delete the feature branch:	git branch -d <branchname>
	Push the branch to your remote repository, so others can use it:	git push origin <branchname>
	Push all branches to your remote repository:	git push --all origin
	Delete a branch on your remote repository:	git push origin :<branchname>
Update from the remote repository	Fetch and merge changes on the remote server to your working directory:	git pull
	To merge a different branch into your active branch:	git merge <branchname>
	View all the merge conflicts: View the conflicts against the base file:	git diff git diff --base <filename> git diff <sourcebranch> <targetbranch>

	Preview changes, before merging:	
	After you have manually resolved any conflicts, you mark the changed file:	git add <filename>
Tags	You can use tagging to mark a significant changeset, such as a release:	git tag 1.0.0 <commitID>
	CommitId is the leading characters of the changeset ID, up to 10, but must be unique. Get the ID using:	git log
	Push all tags to remote repository:	git push --tags origin
Undo local changes	If you mess up, you can replace the changes in your working tree with the last content in head: Changes already added to the index, as well as new files, will be kept.	git checkout -- <filename>
	Instead, to drop all your local changes and commits, fetch the latest history from the server and point your local master branch at it, do this:	git fetch origin git reset --hard origin/master
Search	Search the working directory for foo():	git grep "foo()"

Table 13. Git commands and their meaning.

11.4.Time diagram

Date	Type	Description
16/02/2016	Meeting with tutor	First meeting with Rafael Llobet Azpitarte. Discussing several subjects and inform about the different requirements for the thesis
23/02/2016	Meeting with tutor	Brainstorm about possible project ideas. It resulted in the following project proposal: • IBeacon™ indoor navigation
07/03/2016	Meeting with tutor	Adjustment of the current project idea to class attendance by using IBeacon™ technology in a hybrid application.
08/03/2016	Work	Brainstorm about the idea and possible implementations of the project idea
09/03/2016	Work	Order and purchase of an IBeacon™
16/03/2016	Work	Configuration of IBeacon™s with recommended vendor application, Reset password is necessary. The password is not provided by the vendor so research was necessary.
17/03/2016	Work	Find hosting for the Database and API
18/03/2016	Work	Installation of MongoDB and node.js on server. Determine communication between these two technologies.
19/03/2016	Work	Design of NoSQL database
20/03/2016	Work	Populate database
21/03/2016	Work	Started writing basic API routes in node.js and testing them locally in the program Postman
23/03/2016	Work	Authentication with Jason web token in the API
25/03/2016	Work	Test API remotely and adjust firewall rules to get access to the API remotely.
27/03/2016	Work	Import ‘async’ library to include powerful functions for working with asynchronous JavaScript. And write the other remaining API routes.
07/04/2016	Meeting with tutor	Presenting API,database and IBeacon™ configuration

28/04/2016	Work	Start prototyping application with Ionic Creator and add HTML
04/05/2016	Work	Fine tune lay-out and flow of the application
08/05/2016	Work	Import and implementation of IBeacon™ library to get communication between the smartphone and IBeacon™
10/05/2016	Work	Implementation of the API requests in the application.
15/05/2016	Work	Authentication in the application
23/05/2016	Work	Beacon scan implemented Whole project is complete
24/05/2016	Meeting with tutor	Evaluation of the application and API
28/05/2016	Work	Start writing essay applying template layout
30/05/2016	Work	Determine structure of the essay and create architecture diagram and IBeacon™ image
02/06/2016	Work	Write introduction,current situation,objectives and solutions
03/05/2016	Work	Write project description
05/05/2016	Work	Write text about architecture diagram and IBeacon™
07/05/2016	Work	Write text about hybrid application and the used subtechnologies like Ionic and AngularJS.
10/05/2016	Work	Write about API
15/05/2016	Work	Write about MongoDB
20/05/2016	Work	Write about used subtechnologies
22/05/2016	Work	Write about API part 2
24/05/2016	Work	Write about MongoDB part 2
26/05/2016	Work	Write about used subtechnologies part 2
29/05/2016	Work	Write about future implementations
05/06/2016	Work	Write about used tools and programs
20/06/2016	Work	Collect all the abbreviations in a legend
22/06/2016	Meeting with tutor	Evaluation of the memory, discussing possible improvements and layout of the thesis

23/06/2016	Work	Execute the proposed improvements
24/06/2016	Work	Improve the layout
25/06/2016	Work	Write bibliography
29/06/2016	Finished	Submit the project

Table 14. Time diagram.