Web Project

Studio Saber - Photography Studio Web Application

By Mohamed Iheb Faiza

Introduction

Presentation

Web Project is the project undertaken as mandatory requirement for the subject “Web Architecture” that is being conducted by Polytech Montpellier France in the 3rd year of Computer Science and Management Engineering course.
The aim of the course is to learn about the different aspects of Web Architecture like Protocols, Data Representation, Transfer Methods, Security, etc.

Instructions

For this Project, we, students, have the freedom to choose our own subject for the web application.
Accordingly, i chose to make a web application for a Photography Studio that i have worked at in Tunisia.
We also have the freedom to choose the Web architecture that we see fits for the subject. Therefore, the choice of each of the technologies used in the project must be justified.
The Database used for the project must be available for reading and writing data.
Authentication counts as a bonus in the project. However, one must never ever use **Sessions**.
A trophy called ICAR will be offered to the student who has the best web application.

Studio Saber

Studio Saber is a Photography Studio owned by Moez Ishak and located in Bizerte in the north of Tunisia. The Studio is well known in the region and has its own client base. Studio Saber is specialized in Wedding Photography but also does other events like Parties, Concerts, PhotoShoots, Seminars, School Photos, etc.
The Studio offers several other services for the clients like Equipment Renting, Photo Editing, Video Montage and even Advertisement Posters Design.
During busy seasons, bookings and rents increase very much soo it gets so difficult to manage them.
That is why i decided to solve this problem by creating a web application that allows Clients to book a Photographer or rent Equipment online.

Specification

Here's how Studio Saber works :
The owner hires several **Staff** members. A Staff Member can be a Photographer or a Cameraman or both and has an *email address, telephone number, photo, description* and his own *wage*.
Clients are identified with their *email address* and have their *first-name, last-name*, *telephone number* and a *password* to login and book staff members or rent equipment**.**
Clients can **book** a Photographer/Cameraman for a certain **Event** which has a *name, date, address* and a *duration* (in hours)*.* The Client's bookings are to be confirmed by the owner.
Every **Equipment** has a *name, type* (Camera, Camcorder or Accessory), *description, image, rental* (per day)*,* and it can be available or not.
When Renting an **Equipment** the *date* and *duration* are saved and the Equipment is reserved for 48h for the client to retrieve it from the Studio. **Rents** are also to be confirmed by the Studio's owner.

Functionalities

These are the Functionalities that the web application should have :

Normal visitors can :

- See information about the Studio and its services.
- Find where the Studio is located.
- See a portfolio of the best photos made by the Studio.
- See Staff members.
- Create a Client account.

Clients can :

- Book a Photographer for a certain Event.
- Rent Equipment
- See own bookings
- See own rents

Staff Members can :

- See bookings
- Confirm availability

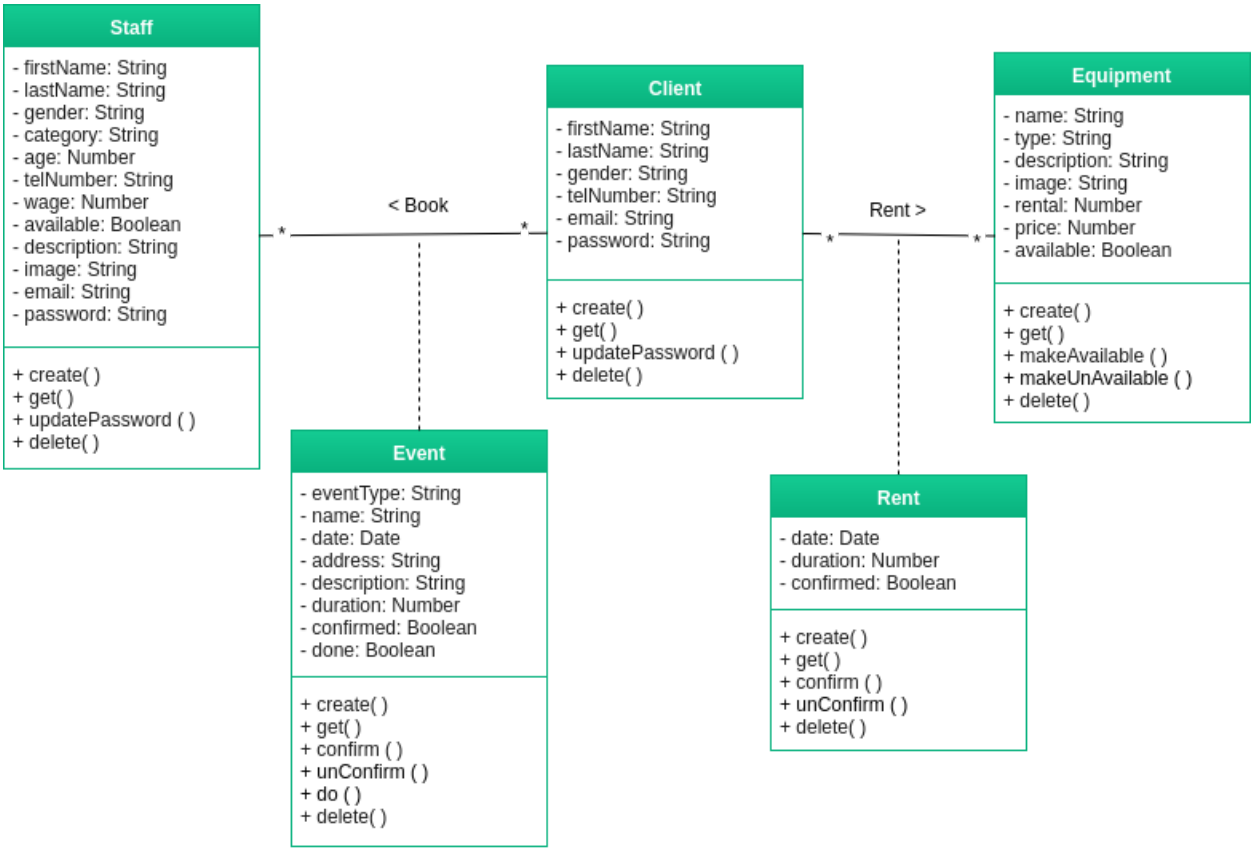
Administrator can :

- See, remove Clients
- Add, see, remove Staff Members
- See, confirm, remove bookings
- Add, remove Equipment
- See, confirm, remove rents

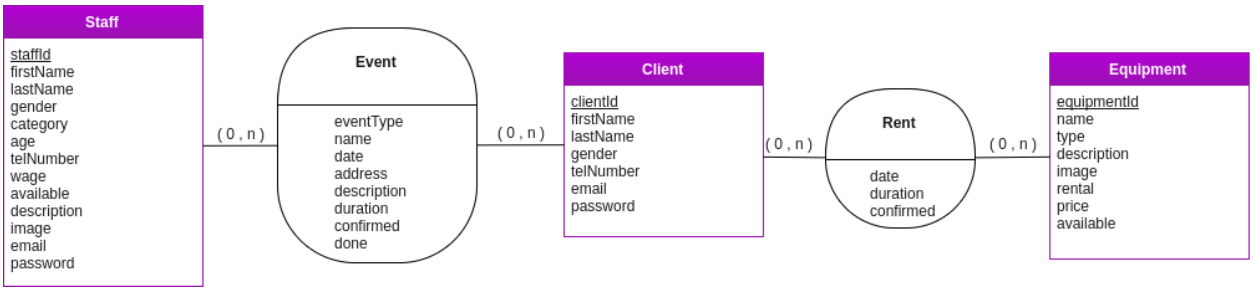
Conception

Traditional Schema Design

UML Class Diagram



Entity Relationship Diagram



Document Schema design

Because i will be using a NoSQL database, designing the Data Model will be different from a normal Relational Database Data Model.
Document Schema design is more flexible, rich of shapes (no rectangles) and offers a Higher level business representation. MongoDB has no declared schema.

MongoDB

The vocabulary used for MongoDB Data Modeling is the following :

RDBMS	MongoDB
Table	Collection
Row	Document
Join	Embedding and Linking

As an example, here is how the Clients collection is modeled in MongoDB :

```
clients :
{
        _id : 592758af5e41385f027e3735
        firstName: “John”,
        lastName: “Jones”,
        gender: “Male”,
        telNumber: “0610948788”,
        email: “john@mail.com”,
        password: “$2a$10$j0F8GE8kCbQG.X9cjCVWXuqTM1nRqJzsQJkOYymCAnZt4Bt9afeZi”
}
```

Architecture

My Data Model is very simple and minimalistic, for this reason i need also a simple, fast and modern Web Application architecture. That's why i am using the **MEAN Stack** with a **RESTful API**.
MEAN stands for :

MongoDB

The leading NoSQL database, empowering businesses to be more agile, faster and scalable.

express

Express

A minimal and flexible node.js web application framework, providing a robust set of features for building single and multi-page, and hybrid web applications.

AngularJS

A MVW (Model-View-Whatever) open-source JavaScript web framework that facilitates the creation of single-page applications (SPA) and data-driven apps.

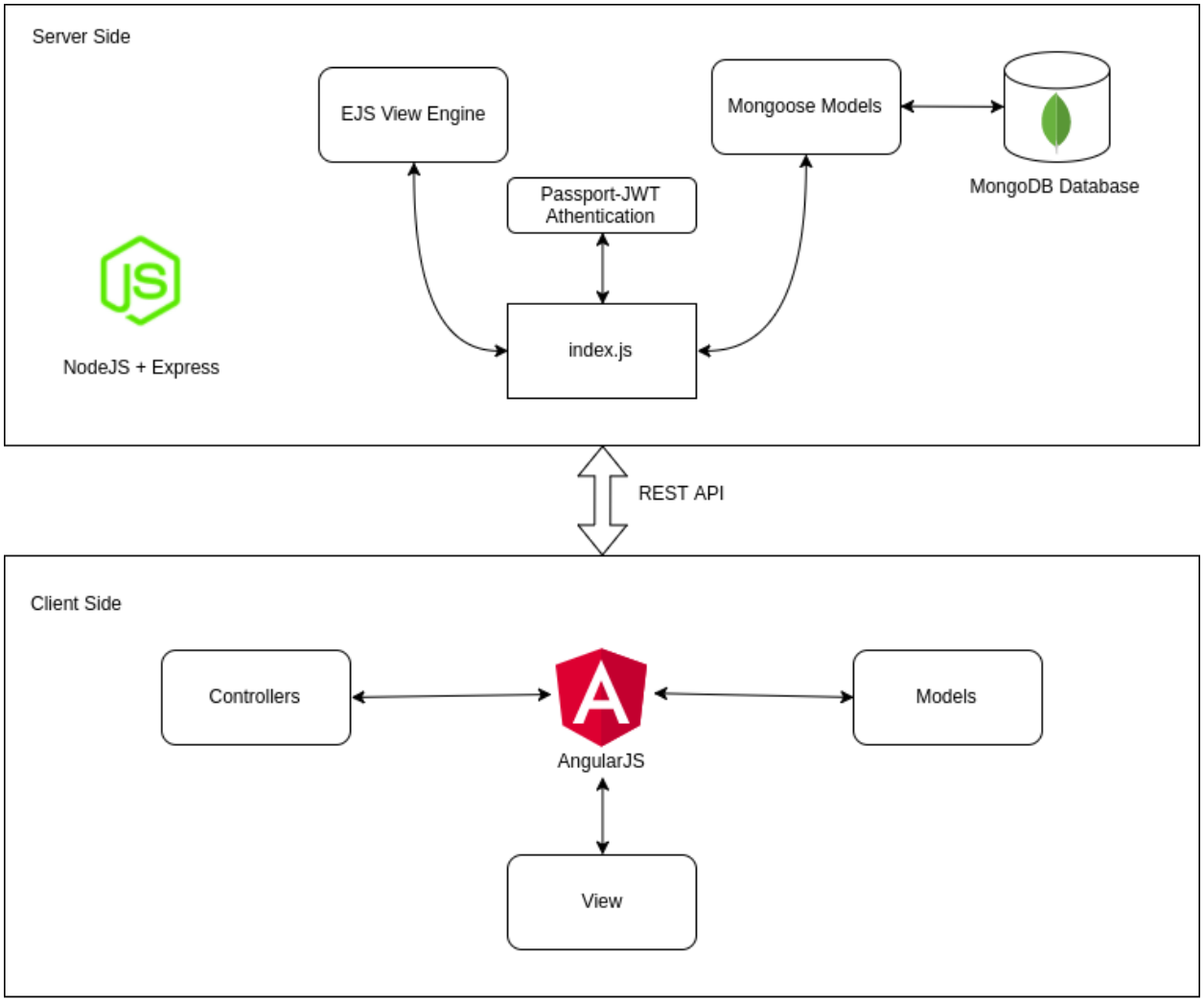
Node.js

A platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications.

What is good about the MEAN stack is that everything uses JavaScript, even the database. This allowed me to learn it easily and to work efficiently on both front end and back end. MEAN stack is the latest technology to the old-fashioned LAMP/WAMP stack, it is rapidly improving and easy to collaborate around.

API stands for **A**pplication **P**rogramming **I**nterface. It is the part of the server that receives requests and sends responses.
REST stands for **RE**presentational State Transfer. It is an architecture that allows client-server communication through a uniform interface. REST is stateless and cacheable. A RESTful API allows me to easily communicate with the server side using JSON (JavaScript Object Notation) which is used in the MEAN Stack for Data Representation.

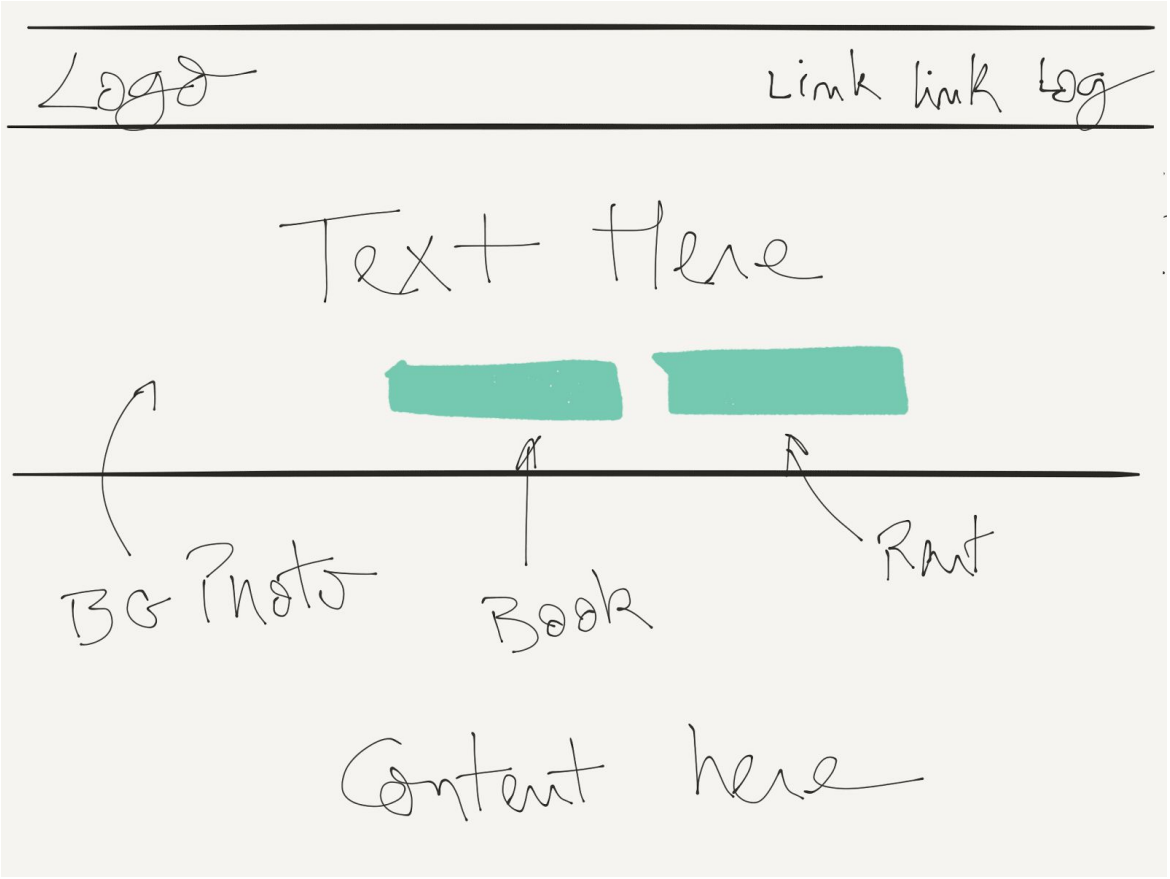
Same Language, same objects.



User Interface

For a Photography Studio the design of the web application is so important for branding. For this purpose, the web application has to be responsive and the theme must have a certain color palette. Images has to have high quality but this shouldn't affect the performance.
In addition, using angularJs, i can make it a Single-page application.

First Maquette :



Logo :

Color Palette :

Project Management

Time Management

The time factor was so important in this project. Project had to be done in 14 Days. On the first day i made this table to estimate the time i have to spend on each task :

Code	Task Name	Previous Task	Duration (hours)
A	Conception	-	4
B	Learning the technologies	A	8
C	Database implementation	B	2
D	Coding	B	12
E	Testing	D	2
F	Documentation	C	5
G	Presentation Preparing	F	3
		Total	36
		Days	14
		Average	2

Learning

There are a ton of ways to learn the technologies i used in the project. Here's my main learning resources :

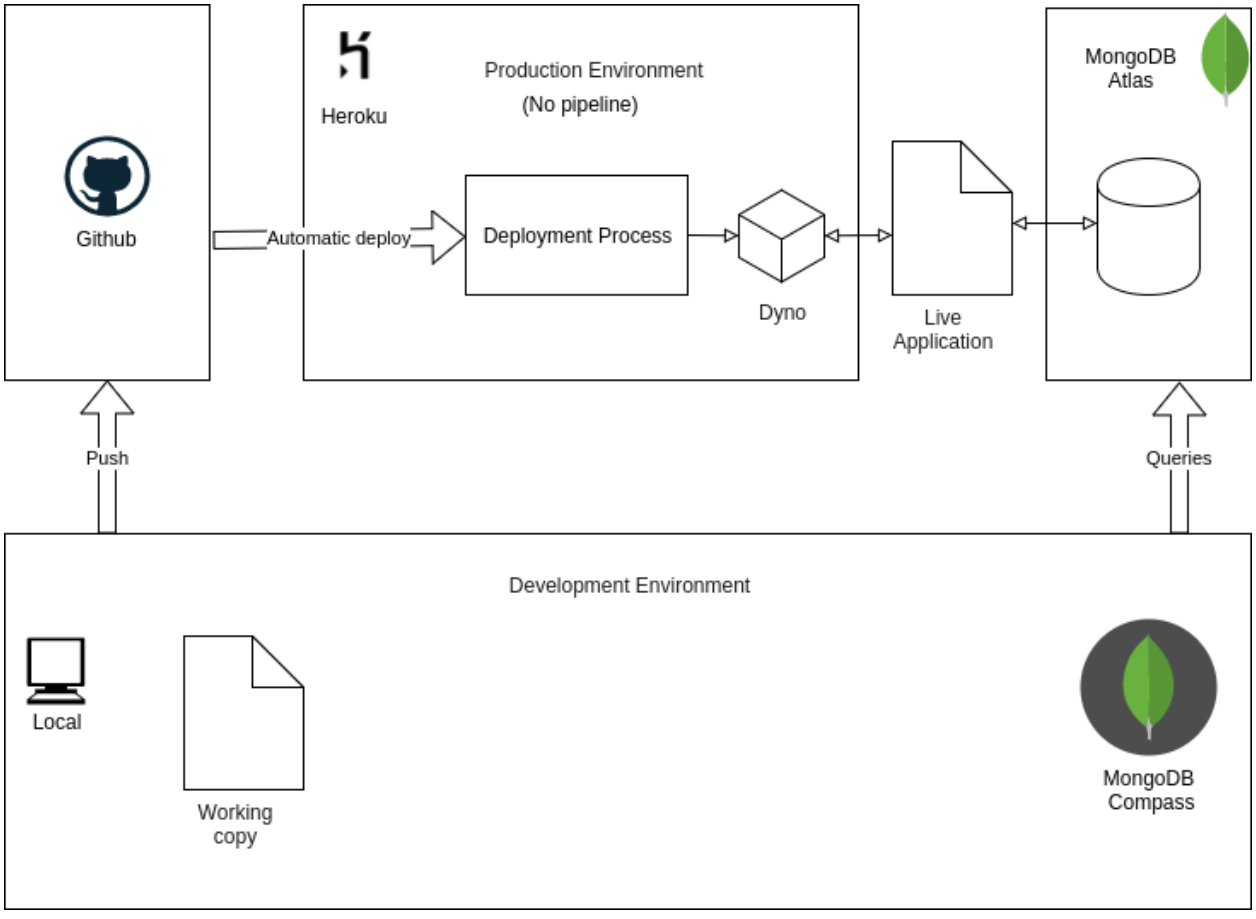
- Manuals and Documentation
- Codeschool.com
- W3school.com
- Tutorialspoint.com
- Youtube Tutorials

Tools

I used several tools in the project, here are the most important ones :

- **Heroku** to deploy the web application on the cloud.
- **MongoDB Atlas** to host the database with **Compass** as a client.
- **Github** for version control and source code management.
- **Google docs** for writing this paper.

Deployment Architecture



Development

Back-end

Building the API :

I have successfully built an API that can:

- Handle CRUD for all items
- Have a standard URL (http://studio-saber.com/client/all and http://studio-saber.com/client/:clientId)
- Use the proper HTTP verbs to make it RESTful (GET, POST, PATCH, and DELETE)
- Return JSON data
- Log some requests to the console

Because i was making cross-origin requests to my API, i used Browsersync to solve this problem locally and a CORS module for the deployed version.
Express allowed me to quickly and easily set up routes.

Models :

I'm using **mongoose** (mongoDB nodeJs module) to define data models and to easily **C**reate, **R**ead, **U**pdate and **R**emove data from the database.

Views :

I'm using **EJS** as a view engine. EJS is a simple templating language that allows me to generate HTML markup with plain JavaScript and organize my views.

Password Encryption :

To ensure security, i'm using **bcrypt** node.Js module to encrypt Client's passwords.

Authentication :

The authentication engine i'm using is called **passport** which has a ton of authentication strategies including facebook and twitter login but the only one i'm using is JWT.

JWT (JSON Web Tokens) is a very simple and secure authentication strategy for REST APIs. It is an open standard for web authentications and is totally based in JSON token requests between client and server.
Client makes a request once by sending their login credentials and password.
Server validates the credentials and, if everything is right, it returns to the client a JSON with a token that encodes data from a user logged into the system.

Client, after receiving this token, can store it the way he wants, whether via Localstorage, Cookie or other client-side storage mechanisms. I'm not using Cookies so the token is saved in Localstorage.
Every time the client accesses a route that requires authentication, it will only send this token to the API to authenticate and release consumption data.
Unfortunately, this last step is not yet implemented in my Web Application.

Front-end

The whole front end side is taken care of by **angularJs**. It handles all the data binding, form validation, http requests and even showing error messages.
It allowed me to dynamically update the single page as the user interacts with the app.
The **bootstrap** framework is used to make the application responsive.

Results

Achievements

Normal visitors can :
☑ See information about the Studio and its services.
☑ Find where the Studio is located.
☑ See a portfolio of the best photos made by the Studio.
☑ See Staff members.
☑ Create a Client account.
Clients can :
Book a Photographer for a certain Event.
☐ Rent Equipment
☐ See own bookings
☐ See own rents
Staff Members can :
☐ See bookings
☐ Confirm availability
Administrator can :
☑ See, remove Clients
☑ Add, see, remove Staff Members
☑ See, confirm, remove bookings
☑ Add Equipment
☐ remove Equipment
☐ See, confirm, remove rents

On the server side, the API is ready to make the actions of all the unachieved functionalities. Therefore, the user interface must be completed to finish the project.

Effort

This is the actual time that i have spent on each task :

Code	Task Name	Previous Task	Duration (hours)
A	Conception	-	5
B	Learning the languages	A	18
C	Database Implementation	B	4
D	Coding	B	12
E	Testing	D	2
F	Documentation	C	5
G	Preparing Presentation	F	0
		Total	46
		Days	14
		Average	3

Experience

It was a very positive experience for me. I learned a lot about both front end and back end Web development and that's because i spent more time on learning the technologies and how web applications work.
I have experience that if i had more time i can make the project more successful.

Future of the project

My aim in the future is to bring this project to life by finishing it and then adding more features making it usable by the Studio's owner.
Therefore, i would like to develop the mobile application for the Studio which will be easy knowing that we can use the API to communicate with the server .
Another important idea is to **uberize** the whole concept of booking Photographers and that is by making the Studio's application open for every Photographer who wants to work independently and make money online.