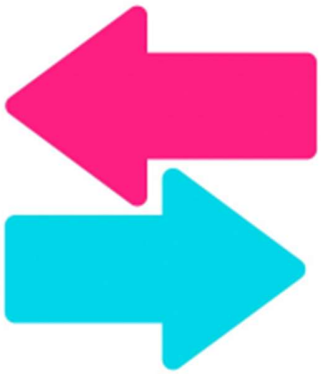


Swap Rooms

PROFESSIONAL PRACTICE IN IT PROJECT

Queli Serrano | G00361469
Simon O'Sullivan | G00359267

Table of Contents

Introduction

THE IDEA

As college student's many of us don't have a lot of money left to travel to new places beside our college accommodations and our family's homes. Most of us are young and haven't experienced life outside these places. Having this In mind we started to think about ways for us too travel to new city's maybe during an holiday, or even just a weekend, but the price of accommodation in Ireland is one of the biggest expenses for any person. That's when we came up with the idea of creating a platform where students from all over the country could connect with other students to swap locations for a few days and experience how life is like in another city.

The idea of the project is that students will be able to put up their current room's on an offer in the platform, for a specific date or time period and other students will be able to put their own rooms up on the platform. They will then be able to search through the available rooms at specific times and connect with a student that has a room that interest's the user and is available at a suitable time for both of the students, once a suitable room is found the student sends a request to the other student and if booth students agree they will have each other's contacts to arrange the swap. The communication between the users would be made through email outside off the application itself. The application simply allows students to connect with other students that wish to swap rooms for a period.

The project can be split up into multiple tasks that are divided between the members of our team which are the following.

- Designing and creating the database – Simon O'Sullivan and Queli Serrano
- Php scripts with database queries – Simon O'Sullivan and Queli Serrano

-Designing the look and structure of the pages was also a task completed by both members of the team, according to the units that we were developing. For example, Simon developing the login and registration system, will also handle the styling of that unit. While Queli developing the offers system will handle the styling of that unit. But both members keeping a standard theme for the website and styling their unities according to the theme.

-Login and Registration system / Account Information – Simon O'Sullivan

- Sign up – Simon O'Sullivan
- Login – Simon O'Sullivan

- Displaying and editing account information – Simon O'Sullivan

- Offers system – Queli Serrano

- Creating offers – Queli Serrano
- Viewing all offers and individual offers – Queli Serrano
- Accepting, rejecting, and notifying users about offers – Queli Serrano

Technology Used and Why

-Angular

The main technology used in our project is Angular, when it came to deciding which technology we would use for our front end we research for a bit and came up with a few options each with their own qualities and defects. For instances REACT was a tempting option but besides us already having experience with react we preferred to choose angular as our frontend framework because of its architecture which keeps every part of the application separate as its mainly based on three layers (Mode, Controller, View). It seemed the most suitable solution for our problem which would allow us both to work on different components independently of the rest of the applications. It also provides us with multiple data binding options, and it is simple to set up and scale using dependencies. Angular is also a pretty good framework when it comes to testing the application and the use of typescript allows us to make the application more compact than using JavaScript and makes it easier to navigate.

Angular also fully supports dependency injections allowing us to use services and delegate functions to those services to provide instead of the component doing all the work. This also allows to a lot of reusability of code in the project.

-Bootstrap

For styling our application we decided to use bootstrap as it provides us with a good framework to make our application visually appealing to the users and responsive to different devices that it may be accessed on. As we are using angular, dedicated angular libraries used which contain components of the Bootstrap library but are powered by Angular.

-Php

For our backend, we decided to use PHP over Node as our scripting language as it is simple to configure, most internet hosting services support PHP leaving our options opened when it comes to hosting. But the main reason for us to choose PHP over Node is that we plan on using a MySQL database and PHP is known for working quite well with MySQL databases without the need of external libraries that would be required with node.

-MySQL

For our database we decided to use MySQL over MongoDB because we require a database that would allow us to store information in a relational manner, therefore MongoDB would not suit our needs for this application.

-Google Cloud

For hosting our application, we decided in a later stage to host it on a virtual machine that we own on the google cloud platform simply out of convenience since we

had a virtual machine set up already and liked the idea of hosting our own application in our own virtual machine.

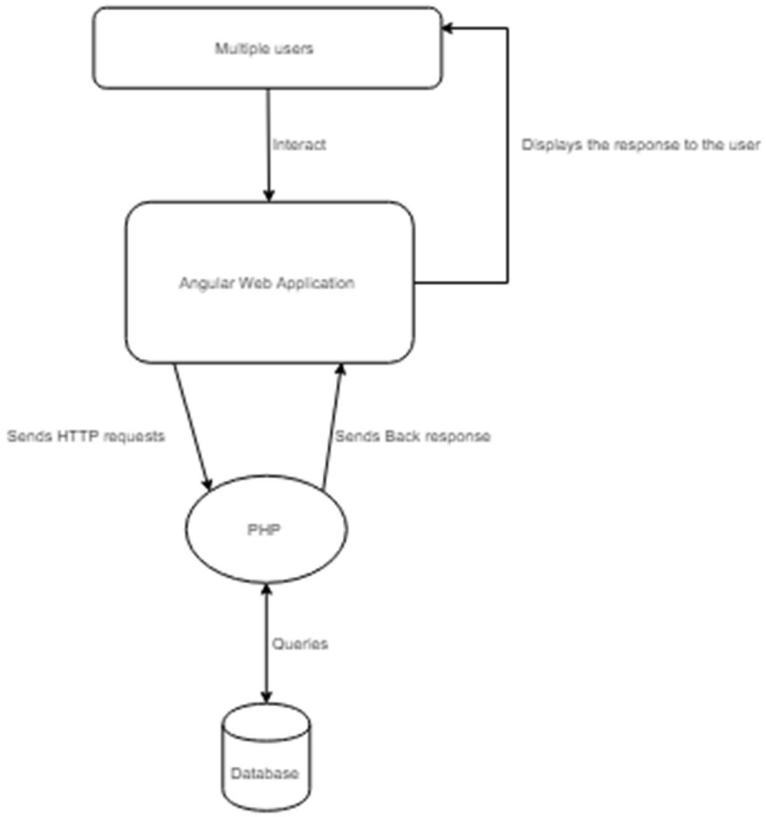
-Wamp Server

For our server on our virtual machine we used Wamp, as it is a technology that we are familiar with and supports our MySQL database and our php scripts.

Architecture of the Solution

Overview

In a very simple view, the architecture of this project is having multiple users interact with the angular application which will send http requests to our php pages that query our database and return the response to the angular application which will then process the response and return it to the user or users.



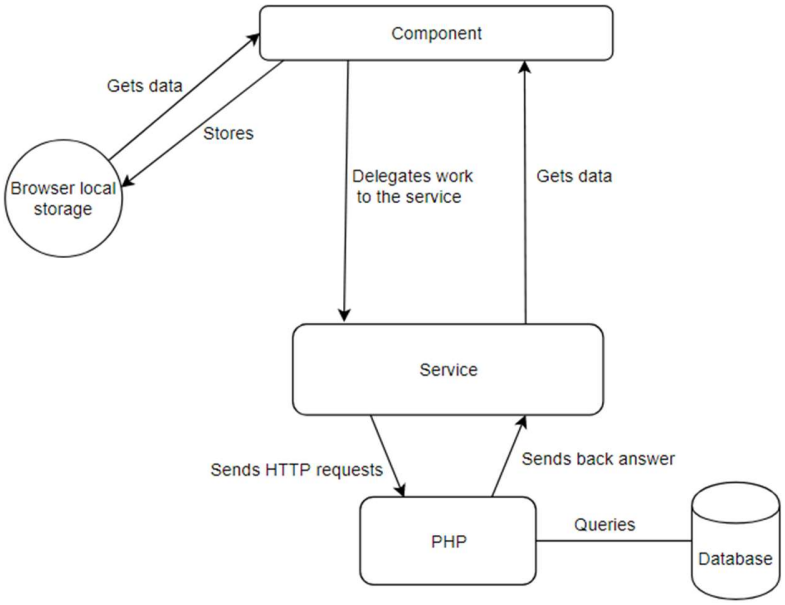
The solution

To solve our problem, we were required to create a web application that would allow multiple users to interact with it, register, login, create offers, search for available offers and accept or decline offers. For this as mentioned above we decided to use Angular.

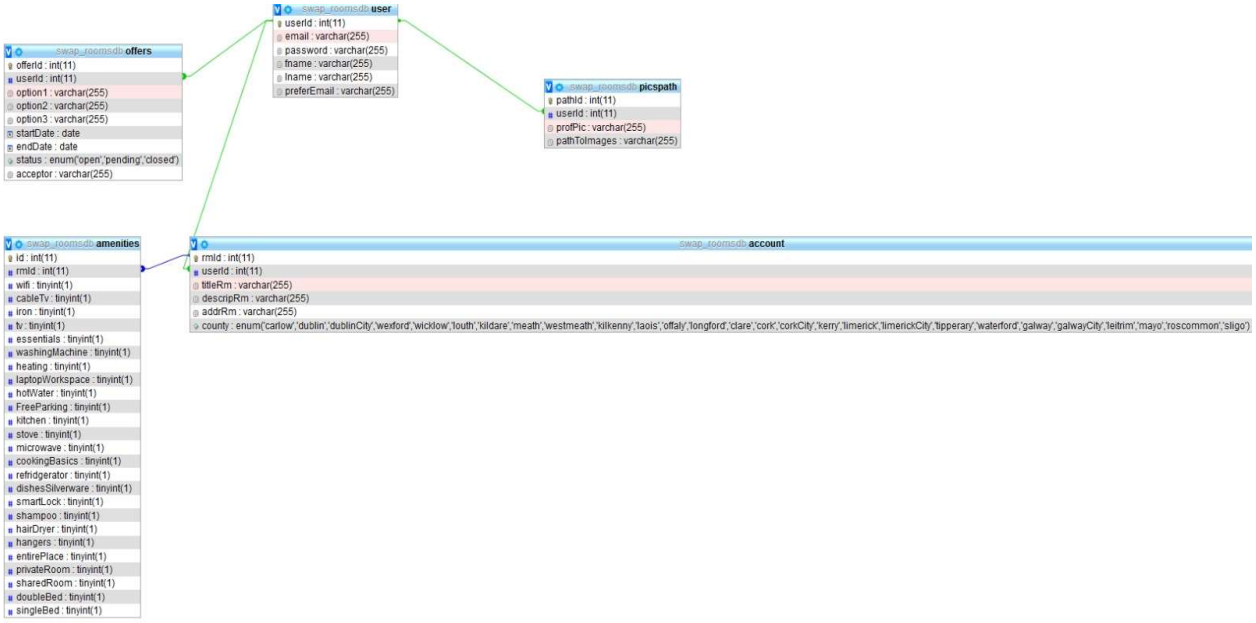
We generated components for each functionality that we required. We used typescript from angular to program the logic of each component where appropriate fetching data from the browser's local storage and through services. We used html with bootstrap to display the contents off the components in a visually pleasing manner.

The components delegate work to the service provider that will handle the communication with the php scripts, the services send http requests to the php which will

then receive and process those requests and query the database accordingly. Once the queries return, the php script will return the answer to the service provider which will then return the data received to the component. The component will process the service response and act according that response either by storing data in the local storage in case of a login for example, or by displaying a success or error message.



For our database we decided to use MySQL because it is a relational database as we mentioned above. The schema of the database it's the following:

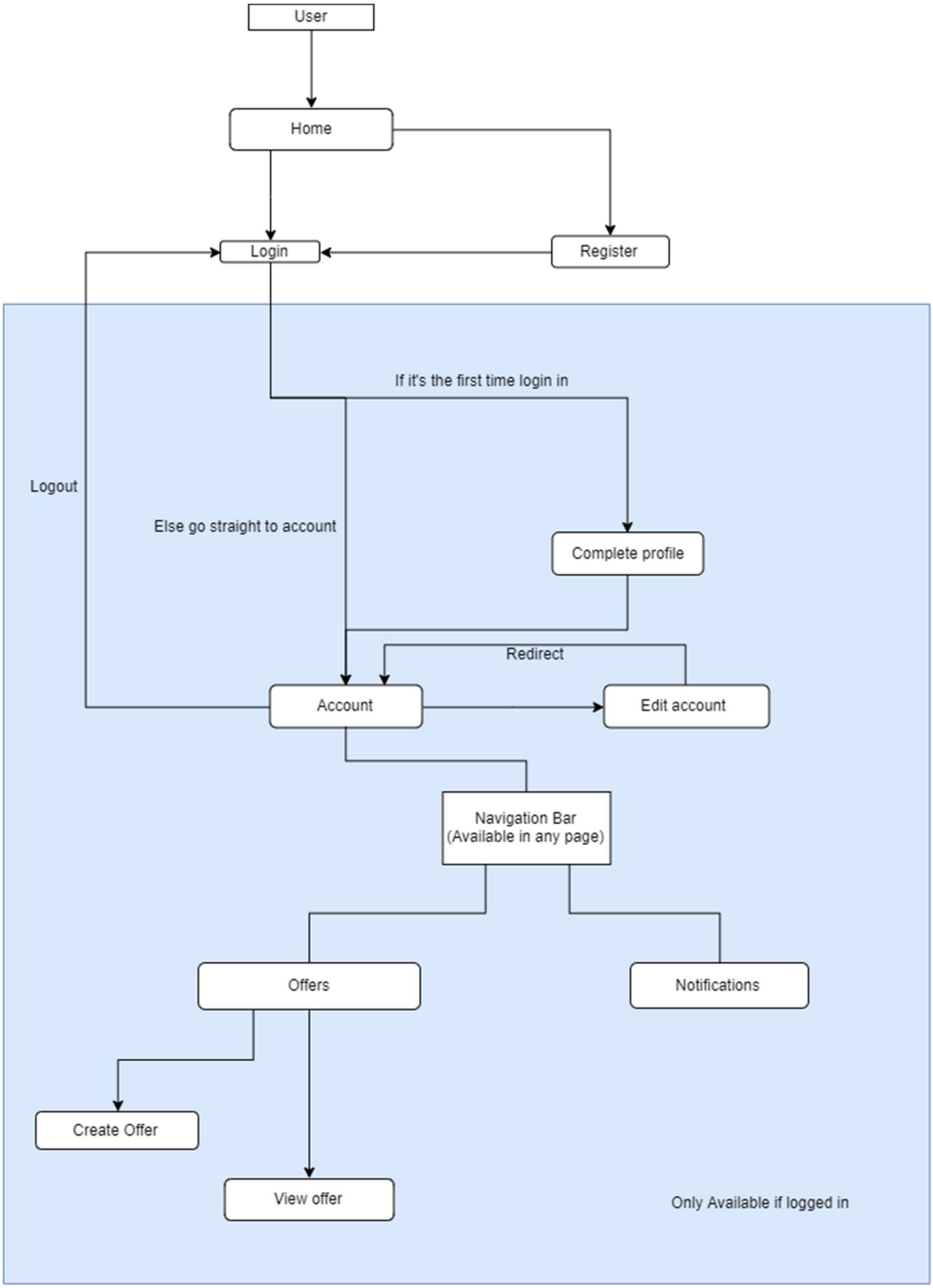


A short description of the tables used:

- User table
    - Stores the main user details like their email and password for logins as well as very basic details like the user's name. It also stores a user ID that is a unique, auto incremented field used to identify the user.
- Account Table
    - Stores the main details about the room and contains the user ID of the owner of the room as a foreign key. It also contains a room ID that is unique and auto incremented and is used to identify the room.
- Amenities Table
    - Stores all the other optional details about the room, it uses the room ID as a foreign key to relate to the account table.
- Offers Table
    - Stores all the offers and their details, including the preferred cities to swap with as option 1, option 2 and option 3. It also stores the user ID as a foreign key.
- Picspath Table
    - Stores the paths to the images and profile pictures of the user, using the user ID as a foreign key.

The application is built using the Angular command and it's hosted in our virtual machine in google cloud with the aid of Wamp. The database is also hosted in the virtual machine and accessed using Wamp.

Design Methodology



Features of the Implementation

The user can:

- Register into the System
    - Using his email address and a password of his choice.
- Login
    - Using the credentials of a previous registered account, to access any functionality of the application the user is required to be logged into the system.
- Add details to his profile about him and the room that he has available
    - The user is required to add details about himself and about the room that he intends to be swapping with other users.
    - The user can also upload images of his room and a profile picture for himself.
- Edit his details
    - The user can edit his personal details, room details as well as his password at any moment through his account page.
- Create offers with his room
    - The user can create an offer with his room details and the dates that he has the room available for swapping, as well as preferred cities to swap with.
- Accept or reject offers on his own room
    - The user can accept or reject offers that are accepted by other user on his room offer.
    - If the user accepts the offer, they will be displayed with contact details to contact each other to arrange the swap.
    - Else if the offer is rejected the user's room will still be listed for swap.
- Request to swap room with any available offer
    - The user is also able to request a swap in any listed offer that will be later accepted or rejected as mentioned above.
- Filter the available offers by location
    - When looking through the available offers the user is able to filter the available offers by location, by typing the desired location into the search offer bar.

Limitations and Known Bugs

When deciding to use Angular as our main framework for the project we did not researched in depth the version of Angular that we decided to use, this caused us a lot of problems with libraries that we tried to use to achieve specific requirements of our project. Libraries that were not compatible with the versions of the technologies that we were using, or we could not get them to work in our solution. This limited a lot of the visual components of our application, causing some pages to not look as we initially intended them to look like.

Uploading profile pictures, the picture only updates after the page is refreshed

In some cases, the pictures do not display correctly.

Testing Plans

We performed Unit tests in each component the developed as we went along the project, verifying that each unit is working on its own as they are required to be.

After unit tests are completed for multiple components, we decided to use an incremental approach to the integration testing. Therefore, as the components were developed individually and tested for individual functionality, they would then be integrated one by one and tested to see if they still work as expected when put together. This allowed us to find functionality mistakes early which could then be corrected more easily than if we decided to leave it all to the end.

When it comes to System Testing, we got member of our families to test the functionality and requirements of the project, as a user. This provided us with feedback about how intuitive the application was for those users, how well it performed and if it successfully completed the functionality that it is supposed to provide. We also got to experience how the system performed with multiple users interacting with it at the same time.

Recommendations for Future Development

With this project we learned a lot about creating a full project by ourselves, from deciding the topic of the project, to deciding the technologies used, planning, testing, and implementing the project. For a future reference what we regret the most about our project was not putting in as much effort in the beginning of the project, for densely researching our project and taking a lot more time to plan out the entire project before starting to implement it.

During the project we found a lot of problems and limitations with the technologies that we were using and we had to go out of our way to try and find a ways around those problems where possible, when in reality they could have been fully avoided by putting in more effort in planning and researching the project.

Another big thing for us to keep in mind for future projects it's the security involved in our application. In later stages of the project we noticed that some of the decision that we made during the project caused a lot of vulnerabilities in the systems security, even though security was not the main focus of our project it could have been a lot better if we once again had been more careful with the planning phase of the project.

Conclusions

With this project we aimed to plan and develop a complete project that would consist of a web application that the users to list their rooms up for swapping with other users of the application.

The goals of the project were for us to experience in firsthand what is like to develop an entire application from scratch, make decisions about the design of the application, the planning, and the implementations of the application. While working as part of a team and on our own.

With this project we have learned a great deal about the full process involved in designing software from scratch, the main things that we learn with this project can be listed as:

- How critical the early stages of designing and planning a project are before even thinking about implementing a project.
- How each decision during the project can have a huge impact on the project, good or bad.
- How difficult we found coming up with an idea due to the freedom provided in the project, we felt a bit overwhelmed with the amount of possibilities and technologies available for us to use.
- We learned how to use PHP to interact with a server.
- The importance of software security when developing an application.
- Dealing with the pressure of having multiple projects to complete simultaneously while having to keep up with the classes and our personal life's.
- Working as part of a team and effectively communicating with your colleague, dividing workload between the members of the team, and sharing the responsibly of the project with everyone involved.
- The importance of helping members of your team whenever they need it and asking for help yourself whenever you require it.

Looking back on the project it was not one of our best projects, we struggled with the decision of the project and the planning as well as starting our project. Once we finally got going with the project, we were able to carry through with most of the implementation that we had planned for the project. We did not get the project to be as polished and finished as we would like it to be, we still learned a great deal from all our mistakes during this project. Most of the technology that we had selected for the project delivered as we expected while some caused us a little bit off trouble.

Overall, the experience of completing this project teach us a lot about the world of software development and the importance of every stage involved in the development of software. Even with all our mistakes and failures throughout the project we were able to provide a working project and better ourselves as professionals by overcoming the challenges presented to us during this project. At the end of the day we can be sure that

this project help us realize that we always have something else to learn or something to improve on, and that only by trying and making tons of mistakes so that we can learn from them and grow, we'll be able to become better software developers and better co-workers in the future.