Food Ordering System for the GMIT Catering Company

Ronan Connolly

Vladislav Marisevs

B.Sc.(Hons) in Software Development

April 14, 2016

Final Year Project

Advised by: Dr John Healy

Department of Computer Science and Applied Physics
Galway-Mayo Institute of Technology (GMIT)

Contents

About this project

Abstract This project sets out to create a food ordering system for a local company. The systems primary components are a mobile application that the user interacts with and a web application that the staff interact with.

The need for such a system stems from two problems, firstly the issue of rush hour times during business hours where there is a vast number of customers to service, and secondly to bring more presence and promotion to the business, as they are finding it hard to reach out to their current customer base and would be customers.

We aim to solve the first problem by having a system in which customers can pre-order sandwiches and other products via a mobile application. Users will be able to top up their account, order products, pick a collection time, and view their balance, products and past order history.

The second problem will be solved by implementing push notifications into the application so that the company can let customers know about menus, events and various other updates. Another way to increase promotion and presence is by having various information about the company on the application; for instance: opening times, contact details and general information.

All of the information on the web application can be updated; this includes the menus, opening times, user and staff details, and much more. This information is reflected in the web application. Interactions from within the mobile application including: topping up, logging in, registration and ordering go through the web application.

We plan to create a cohesive, thoughtfully designed, robust system that solves these two problems.

Authors This project was created by two fourth year software development students: Ronan Connolly & Vladislav Marisevs, as part of our Bachelors of Science honours degree in Software Development.

Ronan was in charge of creating all aspects of the user facing mobile app. Vladislav was in charge of creating all aspects of the staff facing web app.

We spent most of our shared time coming up with the overall architecture we would implement, and an interface to be used between mobile and web app for transfer of data.

Acknowledgements We would like to acknowledge and thank our supervisor Dr John Healy for all the time and effort he has put into helping us throughout this project, he gave us a good structure and set milestones for us in order to keep on top of things.

We'd also like to thank the GMIT Catering Company staff for the time spent meeting with us in order to continuously improve and adapt the project.

Chapter 1

Introduction

We set out to create a food ordering system for the GMIT Catering Company (known henceforth as the company). The basic structure is a mobile application (henceforth known as mobile app) for the Android and iOS systems that the user interacts with, and a server (henceforth known as web app) that the staff can log into in order to view transactions, user details and to update the mobile app.

The reason such a system is needed is that queues during peak times tend to be enormous and currently it's hard to service all the customers.

Another reason is to encourage customers to get into a habit of repeat ordering, if it is an easy process then it should increase purchases.

Lastly, the company wants to increase presence and promotion in the college, in order to achieve this end we have implemented push notifications where staff can send a notification to all users. On top of this the mobile application itself serves as a promotional device, containing details of various aspects of the company.

The components contained within the mobile app include pages for login, registration, about the company and user details There is also a way to top up and order sandwiches. A huge emphasis is put on design for this project, using the companies colour theme and creating a nice icon. This mobile app was created using the Ionic Framework which is programmed primarily using the AngularJS framework.

The components contained within the web app include many pages such as the login system, orders, stock, user details, vouchers(for adding credit to your account), settings(collection and opening times) and accounts(staff) pages. This web app was created in PHP using Zend Framework 2. Most of the information on the web app is reflected on the mobile app.

The two applications talk to each other via JSON over HTTP Get and

Post requests.

We set out to create a well thought out, carefully designed, robust food ordering system using modern technologies. This project could be extended in the future to be used

We used an agile structure where we had certain components we needed complete by specific dates. We had various meetings each month with our supervisor and several members of the company.

In order to develop this system we required to connect different platforms together and let them communicate. We were using Ionic Framework for creating cross mobile application that would talk to Zend Framework 2 which will act as administration website and API for controlling data transfer between MySQL database and mobile program.

Cross mobile application will authenticate users using their credentials. This option will let us to create account wallet, identify person and their order history. Mobile app design uses native phone features and user interface components to let user operate without any special training.

Administration website allows user to configure and manage whole system. User can change opening, closing and food collection times. In order to access these settings user should authenticate him self and then will be able to nominate new administration members. This website also allows to view list of orders, customers and track their history.

Chapter 2

Context

- Our project consists of creating a food ordering system for the GMIT Catering Company.
- Our basic objectices were to create a mobile and web application to deal with the above item.

Below we will:

- list what each chapter contains below.
- list out the various elements in our Github repositories below.
- explain the various components of our project.

2.1 Web App

2.2 Mobile App

The mobile application is created using bleeding edge technologies such as the Ionic framework, which utilizes the AngularMVW framework, which in turn is programmed using JavaScript. HTML and CSS were also heavily utilized.

Initially I had no idea about JavaScript, web development or cross platform development. I spent much time studying JavaScript, Angular, Ionic, and the MEAN Stack (MongoDB, ExpressJS, AngularJS, and NodeJS). I then spent a lot fo time trying out various cross platform development frameworks.

2.2.1 Cross Platform Frameworks

I first tried out Cordova, which I found had the capabilities to do pretty much anything, but the community and framework is very sparse, it's hard to get anything up and running, and the stylings are horrid. [**?**]

Next I tried out JQeuryMobile which had better styling but again I came across many issues. [**?**]

Finally I came across the Ionic Framework which uses Cordova underneath. This framework has a huge community of developers, it has the support of Google, Microsoft and many other large companies. They closely work with the Angular team at Google and the TypeScript team at Microsoft. They have a 24/7 group chat system set up with various sub rooms. They have amazing documentation, regular blog posts, quick response to questions on the blog, forums and chat. With Ionic you get:

- All the capabilities of Cordova, which allows you to access the Mobiles native APIs easily
- A slick native UI experience. The app changes design depending on the platform it's running on
- Extensive tooling. Starting an app, templates, app store image creation, logo creator, etc
- Rapid development cycle, there are constant updates (which can cause issues, but is usually great)

[**?**]

2.2.2 Tooling

Once I decided on using the Ionic framework I spent my time completing various JavaScript, Angular and Ionic tutorials. This was a steep learning curve as there is so much tooling for all the JavaScript frameworks. I installed NodeJS in order to use NPM (Node Package Manager) in order install Ionic.

Then Ionic came with it's own tools for various tasks, including SASS for programmatic CSS, Bower (Like NPM or Maven) for adding in new components, Grunt for running tasks (like Ant) and some others like Gulp (similar again to NPM).

Each of these tools takes time to learn. I read documentation and completed at least one tutorial for each.

2.3 Push Server

In order to facilitate push notifications I needed to get their device token and save it in a database. I created a controller in the mobile application, once the app starts it gets the device token along with some other information and sends it to the push server.

The push server is always listening for incoming requests, once it receives one it evaluates it, adds a timestamp and saves the user object (JSON) to a CouchDB server (Using IBM's Cloudant Web Server).

The push server is a MEAN Stack, Yeoman scafolded project that uses NodeJS as the environment, Express as the Web Framework and Angular as the MVC (Model View Controller) framework in order to build the web application.

The reasons we chose to have the push notification server separate is that:

- We did not realise we needed to save the device tokens initially and to implement this new table into the SQL schema is a big effort
- We wanted to try out a MEAN Stack application
- We felt there was no big dissadvantage on having the push server seperate
- The mobile app is so tightly integrated with the push server (everything is written in JavaScript), that it made creating the web application extremely simple
- The GMIT Catering company may assign the task of push notifications to somebody that they do not wish to have access to the food ordering system, such as a social media expert.

Users can:

- Login/Logout via username and password
- View how many devices are registered for push notifications
- View previously sent push notifications
- Send new push notifications, and see if they were sent successfully.

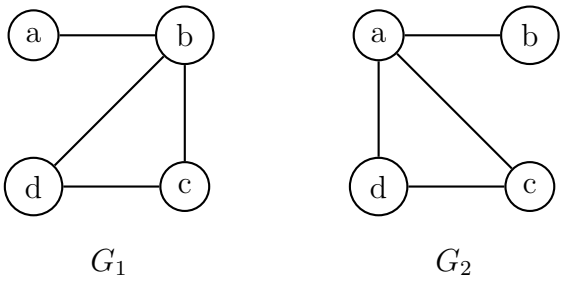
Chapter 3

Methodology

About one to two Pages
Describe the way you went about your project:

- We used an Agile / incremental and iterative approach to development.
- There were plently of meeting and planning with our supervisor and the companies manager.
- We did user testing on the app, and constantly made sure all the routes were working.
- We designed a RESTful style route interface. This let us work on our own components without the web and mobile app waiting for each other.
- We used Github during the development process, tracking tasks via the Github Issue tracker.
- We tested out various technologies at each step of the way and discussed why we were choosing each one.

Check out the nice graphs in Figure 3.2, and the nice diagram in Figure **??**.



G_1 G_2

Figure 3.1: Nice pictures

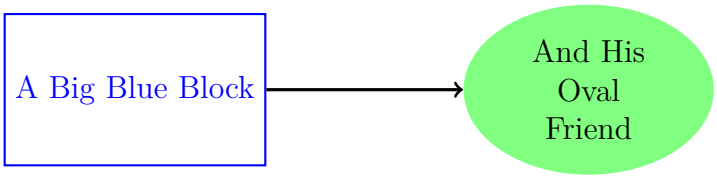


Figure 3.2: Nice pictures

Chapter 4

Technology Review

4.1 Web App

I decided to use PHP since I was already proficient at it and it works really well as a web app because....

4.1.1 php

PHP is one of the widely used server scripting language. It is quite powerful tool for making dynamic web pages and it is free. There are many popular websites that are written on ***php*** such as Facebook, Wikipedia, Flickr, Mailchimp, Wordpress.com. PHP code can be embedded into html page. This type of structure allow to use it in various web template systems, content management systems or web frameworks.

From beginning ***php*** wasn't object oriented language, but in the later versions they designed it.

4.1.2 Zend Framework 2

Prototype was developed using ***Zend Framework 2***. This is an open source framework for developing Web applications and services. It was written on ***php*** and it is loosely coupled architecture, which allow developer to code each component independently and designed with Model View Controller structure. ***Zend Framework 2*** uses 100% object-oriented code and utilises most of the new features of ***PHP 5.3***, namely namespaces, late static binding, lambda functions and closures. [**?**]

Barry and Elhakeem [**?**] argues that Zend Framework enables simple, rapid and agile web application development process, and it also offers AJAX support to convert XML

data into JSON format and integrates the most widely used APIs and Web Services of third-party companies such as Google, Microsoft, Amazon, Flickr and Yahoo. ZF provides many options for validation such as Dojo tools it also allow to filter all inputs.

Based on Hyun Jung La and Soo Dong Kim publication we can say that this project has a Balanced Model View Controller Architecture [?]. The ***Zend Framework 2*** contains applications business logic or Model. And it is also responsible for validating user inputs or Server Side Controller and Web application Views. While ***Ionic*** is responsible for Client Side View and Controller (on the smartphone). In this case Mobile application uses ***zf2*** Controllers to connect to ***MySQL*** database.

4.1.3 MySQL

To store information we were using ***MySQL*** database management system. It's reliability is proven through the years. ***MySQL*** offers complete ACID (atomic, consistent, isolated, durable) transaction support, unlimited row-level locking, distributed capability, and multi-version transaction support where readers never block writers and vice-versa. Easy manageable platform that allow a DBA to manage, troubleshoot, and control the operation of many ***MySQL*** servers from a single workstation. [?]

4.1.4 Composer

For such large distributed projects like this more often used some sort of dependency managers, that allow to force the development cycle and reuse any of already written packages. One of the most popular dependency manager for ***php*** is ***composer***. After installing it we can start using it, simply by creating *composer.json* file which describes the dependencies for your project and may contain other metadata as well. [?]

composer.json example:

```
{
    "require": {
        "php": ">=5.5",
        "zendframework/zendframework": "~2.5",
        "zendframework/zftool": "dev-master",
```

```
    }
}
```

As you can see, require takes a key value pairs where key maps to package names and value is version of it. We also can configure the minimal supported release or nominal version. When *composer.json* is completed, we can run command that will download chosen packages and configure them in our project. To do that run this command in console:

```
$ php composer.phar update
```

4.1.5 WAMP Server

4.1.6 Server Grove Hosting

There is not so many hosting companies that supports Model View Controller structured ***php*** websites, but the good old days when we were querying **.php* files are gone. This type of structure hides the extension and it will confuse the attacker on what sort of platform it was written.

One of most popular hosting companies in Ireland is Blacknight Solutions. For one of my portfolio websites I was using their services. At that time I was not bothered with developing my own website from the scratch and I have used *Drupal* Content Management System. It is quite easy to install and setup CMS for lite blog type of website, but when it comes to connecting 2 different platforms it is no longer useful. When I developed my porfolio website on ***zf2*** and tried to host it on their server I had a lot of issues with their policy. Customers don't have access to main *php.ini* file which can change write properties that has to be on to use MVC structured websites.

To solve this problems I have found *Server Grove* hosting company that fully supports ***Zend Framework 2***, ***Git Client*** and ***composer***. It took me a while to move my domain name from BlackNight Solutions, but once I have it done all technical support on Server Grove is brilliant.

4.1.7 GitHub

text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text

text text text text text text text text text text text text text
text text text text text text text text text text text text text
text text text text text text text text text text text text text
text text text text text text text text text text text text text text text text text
text text text text text text text text text text text text text text text text text
text text text text text text text text text text

4.1.8 JSON REST interface

[?]

4.2 Mobile App

4.2.1 Ionic Framework

HTML/CSS

JavaScript

AngularJS

4.2.2 Tools

Yeoman

Grunt

Jasmine

Jade

Bower

NPM

BASH

4.2.3 Alternatives

Cordova

PhoneGap

JQueryMobile

Xamarin

4.3 Push Server

4.3.1 CouchDB/Cloudant

4.3.2 NodeJS

4.3.3 ExpressJS

4.4 REST Architecture

4.4.1 HTTP Requests

4.4.2 Custom API

4.5 JSON

Chapter 5

System Design

As many pages as needed.

- Architecture, UML etc. An overview of the different components of the system. Diagrams etc... Screen shots etc.

Column 1	Column 2
Rows 2.1	Row 2.2

Table 5.1: A table.

Chapter 6

System Evaluation

Chapter 7

Conclusion