Coupon Now

Final Report
Fall 2018/Winter 2019

By:

Zain Mustafa Rafae Saeed Dethanone Oudomsouk Robert Selman

Contents

Overview	2
Requirements and Specifications	3
Functional requirements	3
Business Owners	3
Consumers	3
Non-Functional Requirements	3
Technology Stack	4
Work Breakdown Structure	5
Project Work Details	5
Gantt Chart	6
Implementation	6
Work Distribution	6
Development and Testing	8
Challenges	10
Setup and Deployment	10
Web Application	10
Mobile Application	11
AWS Setup	11
Summary	12
Appendix	13
Project Repositories	13
Relevant documentation	13

Overview

CouponNow is a way for local businesses to advertise themselves through digital couponing. Using the website, business owners can register as many businesses or locations as they want and create advertising campaigns with coupons that remain valid from a defined start to end date. Each campaign also has an "interest tag" attached to it so customers can filter what they would like to see.

Customers just have to sign up and download the app, then to find coupons they can choose their interests or check for all coupons within a preferred range. If the user sees something they like, they can save the coupon to be used before it expires. And as the customer begins to use more coupons, the system will learn their preferences and make sure they see the most relevant coupons first. CouponNow makes it easy for people to discover local restaurants, bars, stores, etc. and also find deals.

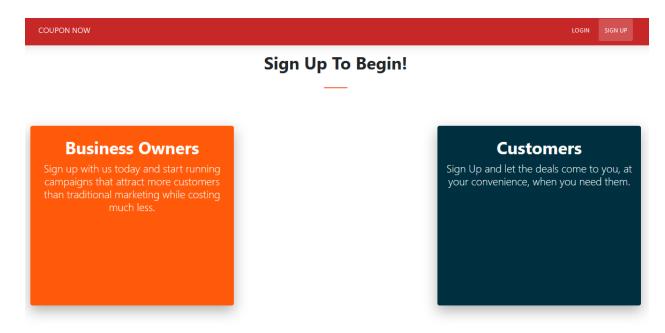


Figure 1: Sign up page for CouponNow

Requirements and Specifications

Functional requirements

Business Owners

- System allows business owners to register and sign in to their dashboards where they will be able to add their own businesses.
- Businesses can be added based on location(s) of the store, the type of the business as well as business licenses ID.
- System validates these new entries based on the given business license. Once verified the business owner will be allowed to start a campaign.
- Each campaign will allow the business owner to set start and end dates for each coupon while keeping track of how many coupons sold in each campaign.
- Business owners will have the option to view analytics regarding each campaign and/or business as well as generate reports.
- Based on the coupon count used for each campaign, our system will set a
 percentage fees that the owner must pay.
- A payment tab will be set with the amount the owners owes as payable amount.

Consumers

- Register and sign in to the application
- Create a profile which contains information on user interests
- Can set their range for receiving coupons
- Receive notifications for available coupons in the set range
- Manually browse coupons
- Accept/Reject coupons to save them to a list or ignore them
- View accepted coupons list
- Select and use coupon from accepted coupons list
- View (transaction) history for information, e.g. current savings
- Change notification settings

Non-Functional Requirements

- Usability interface should be simple and easy to navigate
- Security payments must be secure
- Scalability the system must be robust so that it can handle large amounts of data

• Availability - availability of coupons should coincide with business hours

Technology Stack

The web application was developed using the MEAN stack:

- M: MongoDB, a NoSQL database that uses JSON-like documents.
- E: Express.js, a backend web framework running on Node.js.
- A: Angular, a frontend JavaScript MVC framework that runs on the browser JavaScript engine.
- N: Node.js, an execution environment for backend JavaScript.

The mobile application was developed using NativeScript:

• NativeScript was used because it works well with Angular Framework and allowed us to integrate our already developed code into the mobile application.

MongoDB was a good choice for us because it allows for flexible development. For a project in its early stage with a short development timeline and schemas that are subject to change, NoSQL gives the developers a lot of power. The lack of traditional relational constraints allowed many changes to be made to schemas as different requirements were discovered. Additionally for future development, NoSQL is considered to be relatively simple to scale due to being able to distribute the data across multiple servers.

Express.js and Node.js were chosen because using JavaScript on the front- and backend simplified development. Also, CouponNow is primarily API-driven with little to no CPU-heavy tasks in its current state which allowed us to take advantage of the asynchronous event-driven style of Node.js for quick development of features.

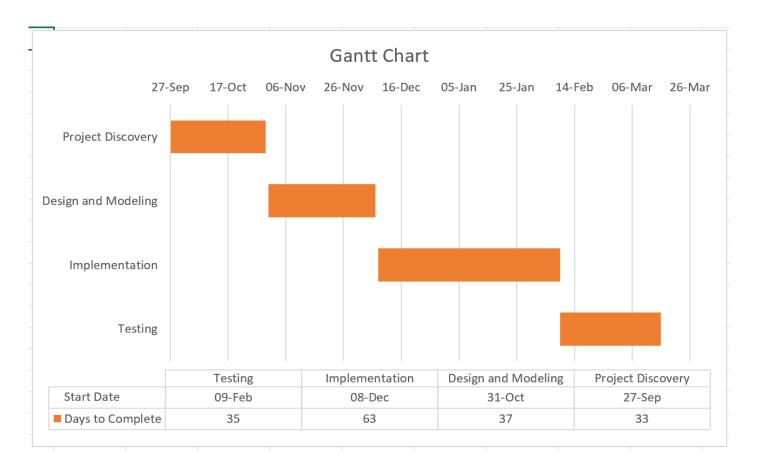
Angular is a widely-used frontend framework with a lot of support and it was a good learning opportunity for our team which did not have much prior experience with frontend frameworks. Its modularity and opinionated nature helped create a well-structured codebase.

Work Breakdown Structure

Project Work Details

Brainstorm Brainstorm Brainstorm Ideas that develop interest Decide on what the team hopes the er Identify what type of user will the project Identify what type of user will the project Identify what type of user will the project Identify what tacach type of user will accord Identify what each type of user will accord Identify what each type of user will accord Identify the assumptions that will be made white down user stories for the consum Identify the assumptions that will be made white down user stories for the business Identify the functional requirements. Functional Requirements Functional Requirements Prioritize the functional requirements of Design Website Identify Interface Implement Login Interface Implement Login Interface Implement Login Interface Implement Company Interface Implement Compan	et al.		
Project Discovery Requirement Discovery Requirement Discovery Requirement Analysis (Consumer) Project Discovery Requirement Analysis (Consumer) Requirement Analysis (Donsumer) Requirement Analysis (Business Owner) Requirement Analysis (Business	Accumulate interest from each student in the group		
Project Discovery Project Discovery Requirement Analysis (Consumer) Project Discovery Requirement Analysis (Consumer) Requirement Analysis (Business Owner) Requirement Analysis (Business Owner) Project Discovery Requirement Analysis (Business Owner) Requirement Analysis (Business Owner) Provide down user stories for the bousiness Owner) Provide own user stories for the bousiness Owner Own			
Project Discovery Requirement Analysis (Consumer) Requirement Analysis (Business Owner) Design Analysis (Business Owner) Requirement Analysis (Business Owner) Requiremen			
Project Discovery Requirement Analysis (Consumer) Derity the assumptions that will be may be a subjected to the consumer of the will be may be a sumption that will be may be the following the assumptions that will be may be a sumption to the business be dentity the functional requirements. I be a sumption to the functional requirements of dependent of the business of the consumers for dependent of the functional requirements of the profit of the business of			
Project Discovery Requirement Analysis (Consumer) Requirement Analysis (Business Owner) Requirement Analysis (Business Owner) Requirement Analysis (Business Owner) Requirement Analysis (Business Owner) Identify the assumptions that will be may write down user stories for the business described the first own and requirements. Functional Requirements Functional Requirements Prioritize the functional requirements. Design Login Interface (Design Design Login Interface) Design Analysis (Business Owner Deshaboard Implement Login Interface) Design and Implement displaying own Tagooners to be added to website Finalize website (Business Owner Deshaboard Implement Owner) Program Application Design Program Application Design Design and Implement Man application Implement Owner Deshaboard Implement Owner Design and Implement Owner Design Owner Design and Implement Owner Design Owner			
Program Program Program Design and Implementation Design and Implementation to I			
Program Program Design and Implementation Des			
Program Program Design and Implementation Program Application Design Application Design Design and Implementation Design and Implementation Design and Implementation Design and Implementation Program Application Design Design and Implementation Design			
Functional Requirements Functional Requirements Functional Requirements Functional Requirements Prioritize the functional requirements. Check the requirements for dependent List how the functional requirements since the functional requirements. Check the requirements of dependent List how the functional requirements of Design Design Mebsite Layout using Wirefram Design Design Exciness Owner Dashobard Implement Logn Interface Implement Logn Interface Implement Company Information Implement Company Information Implement Company Interface Design and Implement Logn Interface Design and Implement Logn Interface Design and Implement Logn Interface Design and Implement Company Interface Design and Implement Information Incorporate heitarchy for most used on Create as soften for the District Incorporate Heitarchy for most used on Create as soften for the District Incorporate Heitarchy for most used on Create as soften for the District Incorporate Heitarchy for most used on Create as soften for the District Incorporate Heitarchy for most used on Create as soften for the District Incorporate Heitarchy for most used on Create as soften for the District Incorporate Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for most used on Create as soften for Heitarchy for	nade for the business owners using the product.		
Prioritize the functional requirements Check the requirements of Deck the requirements for depote ments in the functional requirements of depote ments of the functional requirements of Design Login threshold requirements of Design Website Layout using Wirefram Design Login Interface Implement Up Interface Implement Open Interface Implement Open Check of Design and Implement Design Check of Design and Implement Design and Implement of Design and Implement Design of Design and Implement Design of Design and Implement Design of Design and Implement Main application Implement Off Design and Implement Main application Implement Off Design and Implement Notice of Design and Im	ess owners using the product.		
Program Program Design and Implementation Des			
Design Website Layout using Wirefram	i.		
Program Design Login Interface Implement Login Interface Implement Login Interface Design Business Owner Dashboard Int Implement Usiness Owner Dashboard Int Implement Campaign information Interface Implement Campaign of Implement Campaign Camp	encies upon one another.		
Program Design Login Interface Implement Login Interface Implement Login Interface Design Business Owner Dashboard Int Implement Usiness Owner Dashboard Int Implement Campaign information Interface Implement Campaign of Implement Campaign Camp			
Program Design and implement action Design and implement displaying camp Tag content to be added to website Plays and implement of the business of the program of the pro			
Program Design and Implementation Design and Implementation Design and Implementation Design and Implement despise of implement des			
Program Design and Implementation Design and Implement displaying camp Tag content to be added to website Pinalize website Jayou (Color Schemer Design GUI Jayouts for application Implement GUI Jayouts Design and implement Login Interface Design and implement Main application Implement Design and implement Main application Design and implement Positications to Incorporate PS tracking of the Dusnin Incorporate Personal Positions to Incorporate Positions to Incorporate Personal Positions to Inco			
Program Program Program Application Design Application Design Business Owner Database Business Owner Database Database Design and Implementation Create a schema for the business own Identify Primary keys needed for difference in Create a schema for the business own Identify Primary keys needed for difference in Create a schema for the business own Identify Intervention Supplementation Implementation Implementati	nterface		
Program Program Design and Implementation Design and Implement displaying camp Tag content to be added to website Finalize website layout (Color Scheme: Design GUI layouts for application Implement GUI layouts or application Implement GUI layouts Design and implement Login Interface Design and implement Login Interface Design and implement Login Interface Design and implement notifications to to Incorporate Petratority for most used or Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Consumer Database Consumer Database Database Design and Implementation Database Design and Implementation Database Design and Implementation Consumer Database Unit Testing Database Design campaign information interface Implement displaying and implement displaying camp and implement displaying camp and implement Database Design Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Identify individual units in the source or Test section with to identify if they can be Identify the components in the web app Testific components can communicate Identify the components in the web app Testific components can communicate Identify the components in the web app Testific components that have been Test all the user stories that have been Database Design Identify the components that have been Test all the user stories that have been Test all the user stories that have been Test all the user stories that have been Database Design Identify the Components can communicate Identify the components that have been Testific Identify the Components can communicate Identify the Components that Insure the Apple Identified Identify the Components that Insure the Identify the Components Can Communicate Iden			
Design and Implementation Design and Implement displaying owary Tag content to be added to website Finalize website Jayout (Cloor Scheme: Design GUI Jayouts for application Implement GUI Jayouts Design and Implement Login Interface Design and Implement Dayin Interface Design and Implement Notifications to Incorporate PS tracking of the Busin Incorporate PS tracking of the Busin Incorporate Heirarchy for most used or Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Dusiness Owner Database Consumer Database Consumer Database Consumer Database Unit Testing Unit Testing Test and Incorporate Service that have been Identify the components in the web app Test if components can communicate Identify the components in the web app Test if components that have been Test all the user stories that have been Canned to the string Interface Identify the user stories that have been Test all the user stories that have been Canned Testing Select a potential user to test the appli			
Program Program Application Design and implement Login Interface Design and implement Login Interface Design and implement notifications to incorporate CPS tracking of the Busines Incorporate Personal Incorpor			
Program Application Design and implement Main applications to Incorporate GPS tracking of the Busines Normalize Tables Finalize Business Owner Database Design Identify Primary keys needed for differe List attributes of each table Identify Primary keys needed for differe List attributes of each table Identify Primary Between Busines Normalize Tables Finalize Consumer Database Design Unit Testing Application Testing Identify individual units in the source or Test according to the Components in the web application Testing Applicat			
Program Application Design Design and implement Login Interface Design and implement Nain application Design and implement notifications to to Incorporate Petratorly for most used or Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Business Owner Database Design and Implementation Consumer Database Consumer Database Unit Testing Unit Testing Identify individual units in the source or Test each unit to identify if they can be Identify the components in the web ap Test if components can communicate Identify the components in the web ap Test if components can communicate Identify the user stories that have been Test all the user stories that have been Canner and Canner Identify the components that have been Test all the user stories that have been Canner Identify the components that have been Canner Identify the Canner Identify the components that have been Test all the user stories that have been Canner Identify the Canner Identify the Canner Identify the components that have been Canner Identify the Canner Identified Identif	mpaign statistics		
Program Application Design Application Design and implement Main application Design and implement Notifications to Incorporate GPS tracking of the Business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Business Normalize Tables Business Owner Database Business Owner Database Business Owner Database Enablize Business Owner Database Identify relationships between Business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Unit Testing Unit Testing Test ach unit to identify if they can be Identify the components in the web-app Test if components can communicate Identify the components in the web-app Test if components can communicate Identify the components in the web-app Test if components that have been Test all the user stories that have been Education Test and Test all the user stories that have been Danes and Test all the user stories that have been Education Test and Test all the user stories that have been Danes and Test all the user stories that have been Danes and Test all the user stories that have been Danes and Test all the user stories that have been Danes and Test and Test all the user stories that have been Danes and Test all the user stories that have been Danes and Test all the user stories that have been Danes and Test all the user stories that have been Danes and Test all the user stories that have been Danes and Test all the user stories that have been Danes and	F		
Program Application Design Application Design Application Design Application Design Application Design Application Design Design and implement Login Interface Design and implement Main application Design and implement notifications to Incorporate GPS tracking of the Busin Incorporate GPS tracking of the Busin Incorporate Application Incorporate Applications Incorporate Application Incorporate Applications Incorporate Application Incorporate App	ies, Font style, etc)		
Program Application Design Design and implement Login Interface Design and implement Main applications to Incorporate GPS tracking of the Busine Incorporate GPS tracking of the Business Owner Database Business Owner Database Business Owner Database Business Owner Database Database Design and Implementation Database Design and Implementation Consumer Database Consumer Database Consumer Database Design and Implementation Consumer Database Unit Testing Unit Testing Test if components can communicate Identify the ownponents in the web applements that have been Test allow user stories that have been Database Design United Stories and Stories and Stories that have been Test allow user stories that have been Database Design United Stories and Stories			
Application Design Design and implement Main application Design and implement notifications to to Incorporate GPS tracking of the Busin Incorporate Petratorby for most used or Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Primalize Business Owner Database Enables Finalize Business Owner Database Design and Implementation Consumer Database Consumer Database Consumer Database Unit Testing Unit Testing Test ach unit to identify if they can be Identify the components in the web application Testing Identify the components in the web application Testing Identify the components on the web application Testing Identify the components in the web application Testing Identify the components to the Answebern Install the user stories that have been Identify the user stories that have been Identify the user stories that have been Identify the components that nave been Identify the components that have been Identify the components that have been Identify the components that have been Identify the user stories that have been Identify the components that have been Identify the Identify the components that have been Identify the Identificati			
Design and implement notifications to incorporate GPS tracking of the Busin Incorporate CPS tracking of the Busin Incorporate PS tracking of the Busin Incorporate heirarchy for most used or Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify Primary keys needed for differe List attributes of each table Identify included in the Business Normalize Tables Finalize Business Dwner Database Design and Implementation Consumer Database Design and Implementation Unit Testing Unit Testing Test ach unit to identify if they can be Identify the components in the web appropriate in the set of the propriate in the web appropriate in the set of the propriate in the propr			
Incorporate GPS tracking of the Busin Incorporate heirarchy for most used or Create a schema for the business own Identify primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Business Owner Database Create a schema for the business own Identify primary keys needed for differe List attributes of each table Identify primary keys needed for differe List attributes of each table Identify primary keys needed for differe List attributes of each table Identify primary keys needed for differe List attributes of each table Identify primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Identify individual units in the source or Test each unit to identify if they can be Identify the components in the web ap Test if components can communicate Identify the user stories that have been Test all the user stories that have been Test all the user stories that have been Test all the user stories that have been			
Incorporate heirarchy for most used or Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Database Design and Implementation Consumer Database Consumer Database Consumer Database Consumer Database Consumer Database Unit Testing Unit Testing List attributes of each table Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Identify individual units in the source or Test each unit to identify if they can be Identify the components in the web application Testing Test if components can communicate Identify the components in the web application Testing System Testing Consumer Database Design Identify the user stories that have been Test all the user stories that have been Test all the user stories that have been Test all the user stories that have been Database Design Integration Testing Consumer Database Design Identify the components in the web application of the properties of t			
Business Owner Database Business Owner Database Business Owner Database Business Owner Database Database Design and Implementation Consumer Database Consumer Database Consumer Database Consumer Database Consumer Database Consumer Database Unit Testing Unit Testing Test account to identify individual units in the source or Test account to identify the components on the web application Testing Testing System Testing Create a schema for the business own Identify Primary keys needed for different Identify Primary keys neede			
Database Design and Implementation Database Design and Implementation Database Design and Implementation Consumer Database Consumer Database Consumer Database Consumer Database Distributes of each table and include the properties of the business own Identify Primary keys needed for difference and Identify Primary Prim			
Business Owner Database Database Design and Implementation Database Design and Implementation Consumer Database Consumer Database Consumer Database Consumer Database Didentify Primary keys needed for differe List attributes of each table Identify relationships between Business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Consumer Database Unit Testing Unit Testing Identify individual units in the source or Test each unit to identify if they can be Identify the uponents in the web ap Test if components can communicate Identify the user stories that have been Test all the user stories that have been Select a potential user to test the appli			
Database Design and Implementation Database Design and Implementation Consumer Database Consumer Database	erent table		
Database Design and Implementation Database Design and Implementation Consumer Database Didentify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Finalize Consumer Database Design Unit Testing Unit Testing Identify individual units in the source or Test each unit to identify if they can be Integration Testing Test if components can communicate Test if components can communicate Test if the user stories that have been Test all the user stories that have been Test all the user stories that have been Select a potential user to test the appli			
Database Design and Implementation Create a schema for the business Dwner Database Decreate a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Unit Testing Unit Testing Unit Testing Unit Testing Identify individual units in the source or Test each unit to identify if they can be Identify the components in the web appropriate and Identify the user stories that have been Test if components sent one with the user stories that have been Test all the user stories that have been Select a potential user to test the appli	ess Owner tables and Consumer tables		
Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Unit Testing Unit Testing Integration Testing Testing System Testing Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Identify individual units in the source or Test each unit to identify if they can be Identify the opponents in the web applied to the primary in the pri			
Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Unit Testing Unit Testing Integration Testing Testing System Testing Create a schema for the business own Identify Primary keys needed for differe List attributes of each table Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Identify individual units in the source or Test each unit to identify if they can be Identify the opponents in the web applied to the primary in the pri	Design Design		
Consumer Database List attributes of each table Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Unit Testing Unit Testing Unit Testing Identify individual units in the source or Test each unit to identify if they can be Integration Testing Identify the components on communicate Testing System Testing Identify the user stories that have been Test all the user stories that have been Test all the user stories that have been Select a potential user to test the appli			
Lonsumer Database Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Unit Testing Unit Testing Unit Testing Identify individual units in the source or Test each unit to identify if they can be Integration Testing Identify the components in the web app Test if components can communicate Testing System Testing Identify the user stories that have been Test all the user stories that have been Select a potential user to test the appli	rent table		
Lonsumer Database Identify relationships between Busines Normalize Tables Finalize Consumer Database Design Unit Testing Unit Testing Unit Testing Identify individual units in the source or Test each unit to identify if they can be Integration Testing Identify the components in the web app Test if components can communicate Testing System Testing Identify the user stories that have been Test all the user stories that have been Select a potential user to test the appli			
Normalize Tables Finalize Consumer Database Design Unit Testing Unit Testing Identify individual units in the source or Test each unit to identify if they can be Integration Testing Identify the components in the web ap Test if components can communicate Testing System Testing Identify the user stories that have been Test all the user stories that have been Select a potential user to test the appli	ess Owner tables and Consumer tables		
Unit Testing Integration Testing Integration Testing Test if components in the web applicate in the user stories that have been integration. Testing System Testing Appendance Testing Select a potential user to test the appli			
Unit Testing Identify individual units in the source of Test each unit to identify if they can be Integration Testing Identify the components in the web applied Testing System Testing Identify the user stories that have been Testing System Testing Select a potential user to test the applied Select a potential user to test the applied Select appointment of the source of the sour			
Integration Testing Integration Testing Identify the components in the web application Testing Test if components can communicate Identify the user stories that have been Test all the user stories that have been Test all the user stories that have been Section Select a potential user to test the applications.			
Integration Testing Identify the components in the web app Test if components can communicate Testing System Testing Identify the user stories that have been Test all the user stories that have been Appendance Testing Select a potential user to test the appli			
Testif components can communicate Testing System Testing System Testing Appendance Testing Select a potential user to test the appli			
Testing System Testing Identify the user stories that have been Test all the user stories that have been Appentance Testing Select a potential user to test the appli			
Test all the user stories that have been Appentance Testing Select a potential user to test the appli			
Acceptance Testing Select a potential user to test the appli			
Hodentance Lesting			
Get feedback from the potential user.			
lteratively fix the bugs found.			
List out the buse found in the custom			

Gantt Chart



Implementation

The team decided that everyone should work full stack to prevent blocking during the short development time. Over the ~3 months, we divided the work as follows:

Work Distribution

Dethanone Oudomsouk

- Customer Setup
 - Created web interface for Customers to create an account, add and view their preferences
- API's for Customers
 - Implemented all API's facilitating the storage and retrieval of Customer data from MongoDB database
- Load and Bug testing

- Conducted usability tests to ensure the application met the project requirements
- Conducted load and stress testing using Locust

Robert Selman

- Business Owner Setup
 - Created web interface for Business Owners to create an account, add businesses and add locations for their businesses
- API's for Business Owners
 - Implemented all API's facilitating the storage and retrieval of Business
 Owner data from MongoDB database
- Load and bug testing
 - Conducted usability tests to ensure the application met the project requirements

Rafae Saeed

- Team Lead
 - Responsible for code integration and maintenance of master branch in repository
- Campaign Setup
 - Created web interface for Business Owners to create campaigns
- API's for Campaigns
 - Implemented all API's facilitating the storage and retrieval of Campaign data from MongoDB database
- Resolving Bugs
 - Responsible for debugging and resolving bugs found during testing

Zain Mustafa

- Initial Setup for Web Application
 - Designed project skeleton and structure of web application
- API's for Signup/Login
 - Implemented all API's facilitating the storage and retrieval of Login and Signup of web
 - Implemented all mobile API calls.
- Developed the Mobile Application

Development and Testing

We prioritized the development of the web application first. This was the largest chunk of the development time because everyone had at least one new technology to learn, whether it was the language itself (e.g. JavaScript) or the stack.

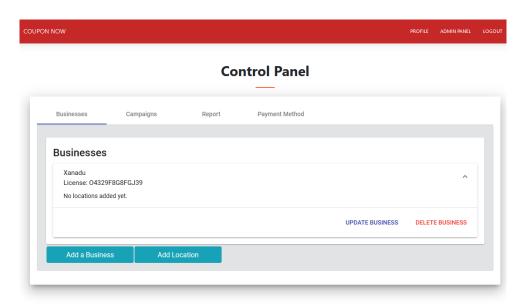


Figure 2: Business owner control panel showing businesses list

Following the website, the mobile application was developed. Many of the API calls developed for the web app were also reusable on the mobile app, so this greatly shortened this phase of development.

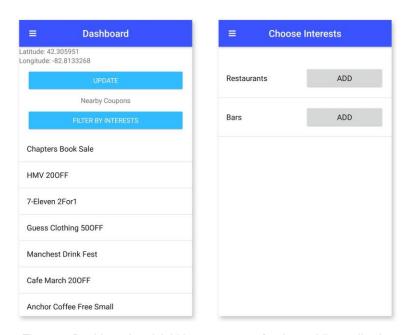


Figure 3: Dashboard and Add Interests page for the mobile application

We also decided to do load testing after development, to find potential areas for future improvement. The following are the results of approximately 5 minutes of load testing using Locust. There were 1000 users in this simulation, spawned at a rate of 20 per second and attempting a login upon spawning. *List Businesses* was called every 5-20 seconds and *Get Coupons* was called every minute.

Туре	Name	# Requests	# Fails	Median (ms)	Average (ms)	Min (ms)	Max (ms)
POST	List Businesses	1533	50	54	1559	31.586170196533203	52523.8835811615
POST	Business Owner Login	91	13	44000	42646	358.80112648010254	72639.80674743652
POST	Customer Login	909	119	48000	42946	169.51513290405273	72814.59259986877
GET	Get Coupons	3831	364	230	3426	51.969051361083984	44688.4183883667
	Total	6364	546	220	9182	31.586170196533203	72814.59259986877

Figure 4: Load testing chart

Failure Rates

- List Businesses ≅ 3.3%
- Business Owner Login ≅ 14.3%
- Customer Login ≅ 13.1%
- Get Coupons (Customer) ≅ 9.5%
- Overall ≅ 8.6%

Response Time:

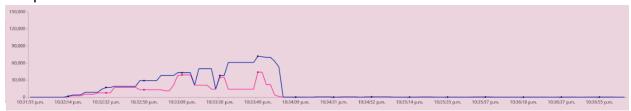


Figure 5: Response time graph

Number of Users:

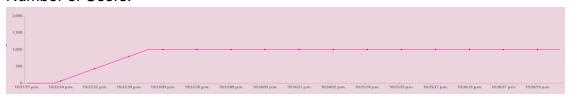


Figure 6: Number of users graph

We can see that the logins had the highest failure rates and once the logins are complete, the response times of the other requests stabilize significantly. The AWS machine we used for hosting is relatively low-powered, but the load testing suggests

that there may be room for optimization in the login code if this app were to be scaled up in any way.

Challenges

During the implementation of the project our team faced several challenges. The biggest challenge the team faced was time restrictions. CouponNow is a functional prototype that implements most of our initial requirements but there are still features that we did not have time to fully implement. We would have also liked for more time to more thoroughly test our application and use those results to optimize performance. Due to our time constraints we focused on usability, load and stress testing to ensure we would have a usable application at the end of our project.

Another challenge the team faced was working with unfamiliar technology. This was intentional because we decided at the beginning of the project to learn a new technology stack. The difficulties of this challenge were mitigated by planning time in advance for team members to learn new technologies and strong communication between team members meant that obstacles could be solved collaboratively as they appeared.

An example of this was finding a way to efficiently query business locations given a mobile users location. An unoptimized query would need to check every business location in the database and this would have a large impact on the performance of our application, especially as the number of users and businesses in our database increased. Our solution was to store business locations as GeoJSON objects and to implement a Geospatial Index on the location field. This allows for much more efficient location queries and will improve the scalability of our application.

Setup and Deployment

Web Application

- 1. In order to run the web application on http://localhost:4200, these steps can be followed:
- 2. Install NodeJS (v11.*): https://nodejs.org/en/
- 3. Install Angular CLI (v7.*) by running the command: **npm install -g @angular/cli**: https://cli.angular.io/
- 4. Install MongoDB (v4.*): https://www.mongodb.com/

- 5. Install Git: https://git-scm.com/
- 6. Clone the repository: https://bitbucket.org/saeed11b/couponnow/src/master/
- 7. Install nodemon from the command line: npm install -g nodemon
- 8. Install local dependencies in couponnow/WebApp/: npm install
- 9. Run MongoDB if it is not already automatically running
- 10. Run the frontend server in couponnow/WebApp/: ng serve
- 11. Run the backend server in couponnow/WebApp/backend/: nodemon server.js
- 12. To create a production instance of the project run the command : **ng build -- prod**

Mobile Application

- 1. To run the mobile application on your android device, these steps can be followed:
- Install NodeJS (v11.*): https://nodejs.org/en/
- 3. Install Angular CLI (v7.*): by running the command: **npm install -g @angular/cli** : https://cli.angular.io/
- 4. Install MongoDB (v4.*): https://www.mongodb.com/
- Install NativeScript: by running the command: npm install -g nativescript: https://docs.nativescript.org/angular/start/quick-setup
- 6. Install Git: https://git-scm.com/
- 7. Clone the repository: https://bitbucket.org/zainmustafac/couponnow-mobile/src/master/
- 8. Install local dependencies in *couponnow/WebApp/*: **npm install**
- 9. Make sure the server is running from the web application repository (follow the instructions above for the web application).
- 10. Initiate debug mode from your android phones settings menu.
- 11. Connect your phone to your computer via a USB cable.
- 12. To start the application on your phone run the command: **tns run android**, from the terminal while in your root directory of where the code is.
- 13. To build a .apk file of the mobile application run the command: tns build android

AWS Setup

- Create an Amazon Web Services account: https://aws.amazon.com/.
- 2. Through your console start a EC2 instance running ubuntu 18.04.
- 3. Connect to your EC2 instance using your local terminal.
- 4. Install NodeJS on your EC2 instance: sudo apt install nodejs npm
- 5. Install Angular on your EC2 instance: **npm install -g @angular/cli** (optional)
- 6. Install MongoDB on your EC2 instance: sudo apt-get install -y mongodb-org
- 7. Install http-server package from npm: npm install http-server -g

- 8. Install pm2 package from npm: sudo npm install -g pm2
- 9. Clone the backend folder to any folder in your EC2 instance from the git repository: https://bitbucket.org/saeed11b/couponnow/src/master/
- 10. Copy the build folder of your web application to the **/root/public/** directory in your EC2 instance.
- 11. Start the MongoDB in your EC2 instance: sudo systemctl start mongodb
- 12. Start the backend server instance by going into the directory you clones the backend folder in and run the command: **pm2 start "node server"**
- 13. Start your http-server by running the command: pm2 start http-server -- -p 4200

Summary

Overall, the team was able to complete the majority of the initial scope that was planned and the general concept of digital coupon delivery works. Some features were not able to be completed during the given time for this project, however. There is no payment system implemented yet nor is there a way for a profit to be made from the application. Also from the customer side, there is a primitive system in place to help users build up their preferences so that they see their most relevant interests first, but there is not a strong enough model to be able to collect data to generate reports and analyse consumer data/trends. This could be important in the future to make the application more profitable.

Additionally, at a lower and more development-focused level, the project could benefit from more rigorous testing and in general some more DevOps training and application. There are no end-to-end tests or unit tests, which could greatly reduce headaches for the developers. It can take very long to manually test features using the website and it can take even longer to discover bugs that are introduced into the product as a result of new features being improperly implemented. Then continuous integration system could be used to automate these tests and perform code analysis (e.g. additional linting for coding standards). Now, any code can be merged into the master with errors and other issues like unresolved merge conflicts.

Appendix

Project Repositories

Web Application: https://bitbucket.org/saeed11b/couponnow/src/master/
Mobile Application: https://bitbucket.org/zainmustafac/couponnow-mobile/src/master/

Relevant documentation

- MongoDB https://docs.mongodb.com/
- AngularJS https://angular.io/docs
- Express.js https://expressjs.com/
- Node.js https://nodejs.org/en/docs/
- Mongoose https://mongoosejs.com/docs/guide.html
- NativeScript https://docs.nativescript.org/
- Google Maps API https://developers.google.com/maps/documentation/
- LocationIQ https://locationiq.com/docs