Sofware Engineering: Fast-Food management system project report

Due on Monday, October 2 2017

Daniel Holmes COMS3002

Group 4

Contents

Introduction

Project Overview

Fast Food Management System brings the people to food and bye-bye to queues in Fast food restaurants. Registered fast food restaurants will be alleviated of queues by receiving orders online and providing users with estimate preparation time. Customers will then collect the food at the establishment or eat-in.

Intended Audience and Reading Suggestions

Software development teams, our peers and superiors alike are who this document is addressed to. Now, this document is written in an effort to facilitate cross-departmental contribution to this project. We open with an overall description to state our intended end product followed by external interface requirements to expand upon our development strategies. System features are included to further modularize the necessary tasks our project must undertake. Nonfunctional requirements explain our intended performance security goals.

Product Scope

Current Situation:
Similar services are being provided by Square Order, Grub-Hub and eHungry. These applications provide users the means to place orders online through a marketspace site implementation that is a platform on which restaurants can host their mobile ordering functionality. We intend on creating an application that maintains the individualism of each restaurant. Our application is customer oriented in that instead of pooling all our registered restaurants we will present the most convenient establishments based on the customers geo-location.

Work Partitioning:

- **Sign Up**:
  - INPUT:
    Customer enters their email address and password which will be stored in the user database.
  - OUTPUT:
    Once successfully stored in our database the customer will now have an active account through which they may access the application functionality.
- **Sign In**:
  - INPUT:
    Customer enters their email address and password which will be checked in the user database as to whether the user exists.
  - OUTPUT:
    Once successfully validated in our database the customer will now be taken to their Home page where they may choose to log-out,enter administration services (Restaurant only) or initiate the restaurant locator and proceed to order food.
- **Restaurant Locator**:
  - INPUT:
    Geo-location of the customer's device location is gathered and transmitted to our server.

  - OUTPUT:

    Based on proximity information the application will provide the user with a list of near-by restaurants which, once one is selected, will lead the customer to the ordering menu of their selected establishment.

- **Order Booking**:

  - INPUT:

    Customer enters their menu selections and proceed to place their order. The order is received by our back-end and payment must be made before finalizing the order.

  - OUTPUT:

    Once an order is successfully placed the customer is issued a unique reference number which will be required by the establishment to collect their order and complete the transaction.

Definitions

Three-tier client-server architecture : System Architecture that will define three logically independent tiers, namely, Presentation, Data management and Domain logic.
Geo-location : An estimation of real world geographical location.
Service providers : Refers to restaurants and onther registered fast food outlets
PostgreSQL : An open source Database management system.
iOS(X) : Operating system used by mobile Apple devices.
Android : Google's Most common open-source operating system used by mobile devices.
Windows 10 mobile : Windows operating system ported for mobile device functionality.
AngularJS : A JavaScript framework that allows a dynamic page refresh.
CSS3 : A framework used to format webpage layout.
Ionic : A modern HTML5, CSS framework that supports mobile application development.
JSON : (JavaScript Object Notation) Is used to transmit data between server and web application.
JQuery : JavaScript framework that allows manipulation of a web page to enble user interaction.

References

- van Vliet, H, (2007), Software Engineering Principles and Practice,Wiley.

Overall Description

Product perspective

The application will take the form of a three-tier client-server architecture. On accessing the application, customers are able to place orders at their selected establishment. Geo-location information is gathered and fast food restaurants will appear in order of proximity from our customers recorded geo-location. Menu presentation of the selected restaurant is followed by a place order interface the customer will use to select their desired items. Payment credentials will be secured and stored prior to issuing an order number to our customer which must be presented on collection of their order. Administrator functionality will include the option to alter/delete menu presentations with store and review functionality for customer orders, this is necessary for monthly reports.

Stakeholders

- **Developers**:
  - **Front-end**: Front-end developers will be using Android Studio and Java to create the aesthetics of the application
  - **Back-end**: Back-end developers will be using the online Firebase database Console to maintain data integrity throughout the system for our Customers and Clients.
- **The Client**: Our clientele will be Fast-Fast restaurant industry. We provide a restaurant with the means to optimize their service delivery by providing a means to expand their customer base. Not only will residents of a particular location be potential customers but now anyone with the application could be a customer.
- **Hands-on users of the product**: Our customers are the fast-food consumers who need efficient preparation and delivery of placed orders.

  Admin :
  - Admin can view orders made by customers
  - Can change menu of their restaurant
  - They can also give feed back

  Customers :
  - Customers can create an account
  - Customers can make an order, they can cancel an order (before the delay time have not elapsed)
  - They can give feedback
  - They can also specify a restaurant they want to place on order to

  The only training necessary is for admin users who will need basic computer literacy.
- **Priorities assigned to users**:

  Customers : Have the least privileged permissions. Customers need only interact with the database passively when signing up and Logging in to the application, their priority is to order food and receive a reference number for order collection.

  Admin :Have the most privileged permissions. Admin reserves the right to cancel an order on non-collection and may change their menu selections. Admin priorities are limited to updating the menu selections, acknowledging incoming orders and preparing the customers order.

Mandated Constraints

This section describes constraints on the eventual design of the product. They are the same as other requirements except that constraints were mandated at the beginning of the project.

- Pick-up location must be selected by the client.
- There must be a time period allowed for changes to a customers order.
- The service is only usable for customers that issue orders within 20km of a pick-up or eat-in restaurant.

Solution Constraints

- High accuracy GPS is very crucial for the system to work at it's best. Also the app should not use more than 128MB of the RAM and take minimum CPU time as that may hamper smooth operation of the entire smartphone.
- HTTP will be our communication protocol. We expect encrypted data to shared as JSON arrays. We want to minimize response time so persistent open threads will need to be available on the server-side.

Implementation Environment of the current system

This software is designed to run on android smartphones. It shall be developed using Android Studio IDE in java, in conjunction with the Google API to provide for geographic-location services. Firebase will be the open source database management system that will be in place to implement all relationships that exist between datasets.

User Documentation

- Consumers will be given a 1-page comprehensive graphical user-manual
- Clients will also be given their own user-manual which will also be used as a staff training manual

Proposed Software Architecture

System Architecture that will define three logically independent tiers, namely, Presentation, Data management and Domain logic.

- **Firebase Database** The database communicates with the application using JSON arrays sent using HTTPS. We use it to validate user login and store user information after sign up. It returns restaurant location and menu details to the Customer-side app and delivers customer order details to the Client-side app
  - **Tables**:
    - Users:
      -Two columns, User email and Password.
      -User email address will be used as a Primary key.
      -A user Profile object will be created using these values
    - Restaurants:
      -Two columns, Restaurant location and Restaurant menu
      -Restaurant location will be used as a primary key
      -A restaurant ordering object will be created on the customer app using these values.

External Interface Requirements

Customer Interfaces

Currently we are only focusing on the mobile versions, Android mobile in particular. That being said, the main feel of the app will be a modern design so tiles and buttons will be constructed using XML HTML5. The first page is a Home screen. The footer of the app will have 3 fixed buttons:

- Order - To show pending orders and well as approved orders
- Logout - To go to the Home screen
- (Idle) - This button on the footer doesn't have a function as of yet

Hardware Interfaces

The app should run on any Android device running API 26 or later.

Software Interfaces

A noSQL database management system,Firebase shall be used for this three-tier system. The Google Maps API should be used for users to visualize their locations as well as locations of the local restaurants.

Functional Requirements: System Features

Pages and permissions

- Restaurant locator

- Identify restuarant name
- Calculate distance to restaurant
- Connect to restaurant server
- Load admin/customer interface

- Admin

- Can view and change the restaurant menu
- Order coordination to ensure effiecient meal preparation for shortest waiting times
- Can view user accounts and see accumulated reciepts and log

- Customer

- Places an order online
- Selects preferred restaurant and places order
- Payment credentials are stored and protected
- Upon successful placement of order, a unique reference number is given to the customer. The number shall be used to access the customers payment credentials
- Customer selects preparation method, eat-in/collect. Their order is then verified and at this point the customer may change their order
- Order is verified and stored in archive. Approximate preperation time and map route(address) is displayed

- Customer locator

- Identify customers location
- Retrieve proxity information from restaurant server and display restaurant location to customer and route.

Logistic management

- User accounts

- Users can register an account that will need their email and password in this way customers can store their receipts with us

- Menu display

- Restaurant menus need to be quite presantable
- The order placing interface will comprise of drop box item selections and view of accumulated cost

- Unique reference number

Non-functional Requirements

Performance Requirements

Search feature :

- The search feature should be easy for the user to find
- Results from any search should be easy to use

GPS :

- The results displayed in the map view should be user friendly and easy to understand
- Selecting a pin on the map should only take one click
- Retreving customer's location should'nt take too long
- Getting the nearest restuarant also shound'nt take too long
- Preferable timmings

others :

- Dropdown menus should be identified and used easily
- Submitting an order should take as lil time as possible
- Admin (restaurant) should recieve an order as soon as the delay time had elapsed
- If the system loses the connection to the Internet or to the GPS device or the system gets some strange input, the user should be informed

Safety and Security Requirements

- Communication between the system and server should be secured. Messages for log-in communication should be encrypted so that other would not get any infomation from them.
- Admin Account should be secured,the Resturant owner should not be allowed to log in (for a certain period of time) after three times of failed log-in attempts. And this must be reported to the developers to check if the system is being attacked or it was just a mistake by the restaurant owner.

Non-functional Requirements [Safety and Security Requirements] continued on next page...

- Since the system require users to pay before an order is made, Customers' sensative information should be heavily protected.
- Orders should'nt be mixed up,make sure that the order made goes to the correct restaurant, so to avoid this orders may be stored with respect to restuarants. If a user chooses to sign up, their email adress/cellphone no should be protected.

Software Quality Attributes

Maintainability : The app shall be written in a way that allows scalability of database resources and maintainability of User information.
Availability : The server shall be running persistent connections to restaurants that are open and are using their Client -side app.
Reliability : Users will always be shown all restaurants in 20km radius that are registered with this system.