README

For

Pizza Ordering System, Version 2.0

Version 1.0

*Prepared by*

GROUP G

Varsha Chandrashekar

Harshil Shah

Sandeep Kasavaraju

November 23, 2014

**Table of Contents**

1. **External JAR dependencies**
2. **Database elements**
3. **Running from Command Line**
4. **Patterns and Refactoring**
5. **Strong and Weak Points of Assignment**

**1.EXTERNAL JAR DEPENDENCIES**

Filename **: http-20070405.jar**

There is one external jar being used in reference for HTTP

Filename: **android.jar**

This jar is not necessarily external and is automatically generated but in exceptional cases may need to be added explicitly.

**2.DATABASE ELEMENTS**

We have made use of files as our storage mechanism and they will be used to store and retrieve data. The following are the files being used:

**Menu**

Text file used to store the menu. All changes which the Store Manager makes will be reflected in this file and it will be shown on the Android application and also on in store.

**CustomerLogins**

Text file used to store the customers details who are registered for the Frequent Buyers program to track their username, password and their reward points.

**coupon.txt**

Text file that has a list of coupons that the Pizza Store offers and which the customers can use to get discounts.

**order.txt**

Text file used to store the orders placed by the customers and which the Chef can see and mark as completed.

**config.txt**

Text file that contains initialization parameters for important data elements.

**employees.txt**

Text file that contains the usernames and passwords for the employees of the Pizza Store i.e the Cashiers,Chefs and Store Managers.This file is used to validate the login for the employees on the Java in store application.

**redeemCount**

File that contains the limit for the points redemption to receive a free pizza certificate from the store. This file is accessed by the Store Manager and can by modified by him/her only.

**3.RUNNING FROM COMMAND LINE**

For starting the Server from the Java use the command:

~$ : Java –cp PizzaOrderingSystem.jar Server

To install Android Application:

~$ : adb install –r path\PizzaOrderingSystem.apk

To run the Application:

~$: adb shell am start –n path/cs.cs414.a5.g.pizzaorderingsystemclient.MainActivity

\*For running the Android application from the command line or the terminal Android Software Development Kit must be prior deployed in the system.

**4.PATTERNS AND REFACTORING**

**Patterns used:**

* Façade- We have used Façade pattern in implementing the Server.
* Singleton – We have used this pattern for two utility classes MenuUtil.java and DataUtil.java.
* Model-View-Controller : We have used the MVC pattern in creating and developing the client server android application.

**Refactoring:**

* Re-factored code from A4 to come up with a singleton class which contains similar looking methods hence eliminating the necessity of duplicated code.
* Removed switch case statements in Menu class which was not necessary and became an anti-pattern.
* Removing the commented code and text that is not useful. In the course of programming the project we had to comment out a lot of code and this has been refactored and removed.
* Refactored and removed Lazy classes from A4 which were not doing any work and were not required and converted to inline classes where necessary.
* Message chaining was initially implemented in the Java application calling parent classes to retrieve their parent classes and so on while implementing suggestions and it turned out to be anti-productive. Changed logic to prevent this.
* Made use of new classes to prevent Data Clumps since client server programming required many data items to be passed to and fro and which have to pass through the Activities in Android.
* Removed temporary variables in many classes which were not necessary for the flow of the application.

**5.STRONG AND WEAK POINTS OF ASSIGNMENT**

Strong Points:

* Good Topic to work on
* Aids in learning Android development.
* Aids in learning client-server development using Java and Android.

Weak Points:

* Problem statement lacks detailed information.