Bilkent University

Senior Design Project

Project: HygieneScore

Low Level Design Report

Alper Eroğlu, Ege Karaarslan, Onatkut Dağtekin, Özge Karaaslan, Defne Demirtürk

Supervisor: Özgür Ulusoy
Jury Members: Mustafa Özdal and R. Gökberk Cinbiş

hijyenskoru.com

Low Level Design Report

February 20, 2017

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS492.

Contents

1. Introduction

The increased accessibility, range and vastness of the internet, motivate people to include it to their daily lives as an essential component. One of the main reasons for this is the availability of seemingly infinite amount of information on the internet. However, most of the information on the web is not refined, not verified and due to these facts is misleading. This can be seen abundantly in the dining industry.[1] Coarse, deficient and vague advertisements of restaurants populate the internet and diminish the joy people get from dining outside. This is mainly due to the unseen and unpublished facts about the infrastructure and hygiene of the restaurants.

The purpose of the Hygiene Score system is to assist people, so that they can reach refined, verified and explicit data when searching for a place to eat. This will be done by the rating system and the educational components provided by the system. The rating system will be liable due to the fact that it uses the rankings made by professional food inspectors that have access to the infrastructure of the restaurants, as well as the rankings of the users that visit the restaurant. The system will combine these rankings accordingly and generate a final rating that is easy to evaluate by the people that are searching for a place to eat. Moreover, the users will be also trained in hygiene and food quality through education modules in the system. These will be available for both the consumers and owners. Through this training, the consumers will be available to detect the flaws that are not easy to see and the owners will be more careful and considerate about their establishment.

In this Low-Level Design Report, the system design presented in the High-Level Design Report will be refined and expanded upon. Initially the design goals that were stated in the High-Level Design will be evaluated further and against each other, and the trade-offs between these design goals will be established. Next, the abstract subsystem decomposition that was presented in the High-Level Design will be displayed in terms of packages that resemble more realistic real-life systems. Finally, the classes in these packages will be presented through their interfaces, which will contain the description, the package it belongs into, its attributes and the operations that the class provides.

1.1 Object Design Trade-Offs

Functionality vs Usability

Hygiene Score is a system that requires user interaction and aims to address a wide range of users. Thus, our system must be usable and not complicated for users to understand. Having a lot of functionality in the system decreases the usability of it. Therefore, we will favor usability more than functionality by keeping our functions enough to create a decent system which will serve its purpose and by creating a GUI that will be understandable by the users.

Compatibility vs Extensibility

The compatibility is very important for Hygiene Score since our aim is to reach to wide range of users. This is the reason why our system must be compatible with various versions of Android OS. At this point compatibility and extensibility conflicts with each other. In the aspect of extensibility, old versions of Android do not give the flexibility to the programmers as the new versions do. Because of this, we will choose compatibility over extensibility.

Memory Space vs Cost

Hygiene Score will store a big amount of data consisting of many restaurant and user information. We want to provide fast service to our users. To be able to achieve this, our services should have high capacity and our data must be fast accessible. Thus, we will focus on memory space rather than cost.

1.2 Interface Documentation Guidelines

Our interfaces are documented in the following format:

Class Name	Name of the class/interface
Class Description	Short description of class/interface
Package	Package name that includes the class
Attribute	Attribute name and brief description of the attribute
Operations	Operation definitions provided by the class

Figure 1: Class Interface Template

1.3 Engineering Standards

We have used Unified Modelling Language (UML) to develop diagrams and demonstrate scenarios. We have used IEEE Citation Style Guide[2] for giving references while writing our reports. Interaction between software components is specified by Java API and Android SDK.

1.4 Definitions, acronyms, and abbreviations

UML: Unified Modeling Language

IEEE: The Institute of Electrical and Electronics Engineers

SDK: Software Development Kit

APK: Android Application Package

API: Application Programming Interface

2. Packages

This section shows the detailed description of the packages of our system, Hygiene Score. In addition the basic functionalities are described as well.

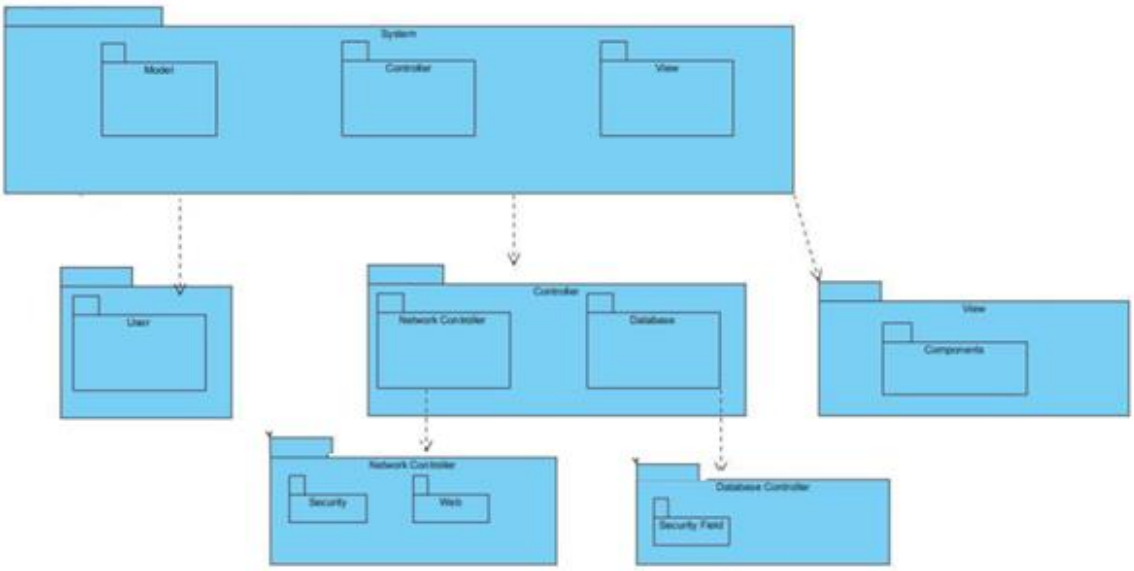

Figure 2: Package Diagram

· The Model package manages the data part, the logic and also the systems rules.

· Controller part manages how the Model is going to be presented to the user.

· View part can be designed according to the changes made by the user.

· Controller part can communicate with Model and View parts if necessary.

· Model and View parts cannot communicate with each other by any means.

· View sends the action when user interacts with UI.

3.Class Interfaces

DatabaseManager:

Class Name	DatabaseManager
Class Description	Responsible of communicating with the database which keeps the restaurant and account information.
Package	Controller
Attributes	restaurantList<Restaurant>: keeps the restaurant list accountList<Account>: keeps the account list
Operations	searchRestaurant(name): finds a specific restaurant from the list authenticate(email, password): used for login updateDatabase(tableName,parameters[ ],operation):updates database values based on operation connect(databaseName):connects to a database facebookLogin(id,email): connects to the app via Facebook profile

TutorialManager:

Class Name	TutorialManager
Class Description	Manages the tutorials based on users account type and shows it to the user
Package	Controller
Attributes	-authType:Account:keeps the users account type for future use
Operations	-loadTutorial(authType,video):load the video based on account type -continueTutorial(timelapsed,tutorialNum): continues tutorial from a given time -checkPayment(account,module):checks if the user has paid for the specific module

GUIEngine:

Class Name	GUIEngine
Class Description	Class that is responsible of communicating with the user
Package	View
Attributes	-
Operations	showContent(view):Shows contents of GUI update(view): updates GUI based on the current page submitComment(restaurant, content): adds comment to a specified restaurant

CustomerTutorialManager:

Class Name	CustomerTutorialManager
Class Description	This class is used for keeping track of videos that customers can see
Package	Controller
Attributes	-customerVideoList<VideoList>:keeps the list of videos for Customer type users
Operations	-listVideos():lists video for customers

OwnerTutorialManager:

Class Name	OwnerTutorialManager
Class Description	This class is used for keeping track of videos that owners can see
Package	Controller
Attributes	-ownerVideoList<VideoList>:keeps the list of videos for Owner type users
Operations	-listVideos():lists video for owners

EducationModule:

Class Name	EducationModule
Class Description	This class contains videos and permissions to the videos
Package	Model
Attributes	-videos<VideoList>: keeps the list of videos -permissions<PermssionList>: keeps permissions for each video
Operations	-authorize(Account):loads the tutorial class based on account type

Account:

Class Name	Account
Class Description	This class is where the accounts and profiles of users are arranged and organized.
Package	Model
Attributes	-email:String -password:String -favorites<RestaurantList>: keeps favorite restaurant list for specific customer
Operations	-makeComment(content): creates a comment based on content -createProfile(): creates a profile with given information -manageProfile(): changes information about the account -getHelp(): provides information about the application -viewRestaurant(restaurant):view information about specific restaurant -searchRestaurant(name): -addFavorite(restaurant): adds restaurant to favorite list

Restaurant:

Class Name	Restaurant
Class Description	The class where the information about restaurant and the operations
Package	Model
Attributes	-name:String -owner:Owner -address:String -rating:Rate -website:String -comments:<CommentList>:stores comments about the restaurant
Operations	-showComments():shows comments for that restaurant -showRating():shows rating for that restaurant -openWebsite():opens website for that restaurant

Rate:

Class Name	Rate
Class Description	Keeps the score for a restaurant
Package	Model
Attributes	-hygieneScore:double:keeps the rating
Operations	-calculateScore(ScoreToBeAdded, Account,oldScore): calculates new Score based on account type

Comment:

Class Name	Comment
Class Description	The customer comments are managed here
Package	Model
Attributes	-user -date:Date -content:String -numberOfLikes:int
Operations	-like() -dislike() -writeComment(newContent):changes the content string

CustomerRate:

Class Name	CustomerRate
Class Description	Contains information about what customers can rate
Package	Model
Attributes	-wcCleanliness:double -service:double
Operations	-evaluateWcCleanliness(value): adds new value to the wc cleanliness criteria -evaluateService(value): adds new value to the service criteria

InspectorRate:

Class Name	InspectorRate
Class Description	Contains information about what inspectors can rate
Package	Model
Attributes	-kitchenQuality:double -cleanliness:double -foodQuality:double
Operations	-evaluateKitchenQuality(value):changes the value of kitchenQuality variable -evaluateCleanliness(value):changes the value of cleanlinessvariable -evaluateFoodQuality(value):changes the value of foodQualityvariable

Admin:

Class Name	Admin
Class Description	The capabilities associated with the Admin type user is managed.
Package	Model
Attributes	Inherited from Account Class
Operations	-editComment(comment): enables editing comments created by other users

Customer:

Class Name	Customer
Class Description	The capabilities associated with the Customer type user is managed.
Package	Model
Attributes	Inherited from Account Class
Operations	-rateRestaurant(restaurant,value): add a rating value to a specific restaurant -payForModule(): pay for a specific module -viewModule(): view a module paid before

Inspector:

Class Name	Inspector
Class Description	The capabilities associated with the Inspector type user is managed.
Package	Model
Attributes	Inherited from Account Class
Operations	-rateRestaurant(restaurant,value): add a rating value to a specific restaurant -approveAdditionRequest(restaurant): add a new restaurant to the database -createEducationalModule(name,duration,address): add a new module to the education module list

Owner:

Class Name	Owner
Class Description	The capabilities associated with the Owner type user is managed.
Package	Model
Attributes	Inherited from Account Class -restaurantList<RestaurantList>:keeps the list of restaurant for the specific owner
Operations	-requestRestaurant(name,address,website): -createRestaurantProfile(): create a new restaurant profile -manageRestaurantProfile(restaurant): change the attributes of a restaurant -payForModule(): pay for a specific module -viewModule(): view a module paid before

4.Glossary

PHP is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language.

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible.

APK(Android application package) is the package file format used by the Android operating system for distribution and installation of mobile apps and middleware.

UML (The Unified Modeling Language) is a general-purpose, developmental, modeling language in the field of software engineering, that is intended to provide a standard way to visualize the design of a system

5. References

[1]Special Report By Tom Rawstorne, "Disturbing proof that persuasive online reviews may be FAKE," in *Daily Mail*, Daily Mail, 2015. [Online]. Available: http://www.dailymail.co.uk/news/article-3276617/Disturbing-proof-online-review-book-holiday-FAKE-Investigation-reveals-entire-industry-dedicated-generating-bogus-appraisals-cash.html. Accessed: Feb. 20, 2017.

[2]"IEEE Citation Reference," in *IEEE*, 2009. [Online]. Available: https://www.ieee.org/documents/ieeecitationref.pdf. Accessed: Feb. 20, 2017.

[3]S. A. Gordon, "What is an APK file and how do you install one?," in *Androidpit*, 2016. [Online]. Available: https://www.androidpit.com/android-for-beginners-what-is-an-apk-file. Accessed: Feb. 20, 2017.

[4]"PHP," in *Wikipedia*, Wikimedia Foundation, 2017. [Online]. Available: https://en.wikipedia.org/wiki/PHP. Accessed: Feb. 20, 2017.

[5]"Java (programming language)," in *Wikipedia*, Wikimedia Foundation, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Java_(programming_language). Accessed: Feb. 20, 2017.

[6]"Android application package," in *Wikipedia*, Wikimedia Foundation, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Android_application_package. Accessed: Feb. 20, 2017.

[7]"Unified modeling language," in *Wikipedia*, Wikimedia Foundation, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Unified_Modeling_Language. Accessed: Feb. 20, 2017.