Bilkent University

Computer Science Department



**Senior Design Project**

*Project Name: Hygiene Score*

High-Level Design Report

Project Members: Ege Karaarslan, Alper Eroğlu, Onatkut Dağtekin, Özge Karaaslan, Defne Demirtürk

Supervisor: Özgür Ulusoy

Jury Members: Mustafa Özdal and R. Gökberk Cinbiş

Dec 30, 2016

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 CS491/2.

Contents

[1. Introduction 3](#_Toc470864788)

[1.1 Purpose of the system 3](#_Toc470864789)

[1.2 Design goals 3](#_Toc470864790)

[1.3 Definitions, acronyms, and abbreviations 4](#_Toc470864791)

[1.4 Overview 5](#_Toc470864792)

[2. Current Software Architecture 5](#_Toc470864793)

[3.Proposed Software Architecture 6](#_Toc470864794)

[3.1 Overview 6](#_Toc470864795)

[3.2 Subsystem decomposition 6](#_Toc470864797)

[3.3 Hardware/software mapping 8](#_Toc470864798)

[3.4 Persistent Data Management 8](#_Toc470864799)

[3.5 Access Control and Security 10](#_Toc470864800)

[3.6 Global software control 11](#_Toc470864801)

[3.7 Boundary conditions 11](#_Toc470864802)

[4. Subsystem Services 12](#_Toc470864803)

[5. Glossary 14](#_Toc470864804)

[6. References 15](#_Toc470864805)

# 1. Introduction

This report will explain the high-level design of our project, Hygiene Score. First section of this report will explain the  purpose and design goals of the system. The second section will explain the current software architecture. After that, proposed software architecture will be explained, covering topics such as subsystem decomposition, hardware/software mapping, persistent data management, access control and security. The last section will contain the subsystem services such as database manager, UI, inspection grading manager and connection manager.

## 1.1 Purpose of the system

Hygiene Score is an application that wishes to meet people’s demand for more information and transparency by providing enlightenment about unseen feature of restaurants supervised by professional inspectors. From an owner’s point of view, they are investing grave amount of money to advertisements to attract customers to their establishments. With the recent increase of demand for transparency, being undisguised is a form of great advertisement. This application is a way for them to open their doors and show their infrastructure in a single environment and promote their restaurant in the most noble of ways. The customer will also have a part in the evaluation process by commenting on the restaurant and rating the features visible to them. Both the customer and the owners of the restaurants will be able to learn more about the evaluation of the hygienic process by watching/reading about the tutorials included in the application/website which will differ by the type of account they have.

## 1.2 Design goals

**User-friendly Interface**

Hygiene Score’s target audience is generally the new millennial generation, although the user’s age could increase in daily life. As younger generation is our target audience, our application should have a user friendly user friendly interface otherwise no matter how great our application is, people will not get used to using neither our application nor our web site.

**Maintainability**

For our application, Hygiene Score, the maintainability would have  both the back end and the application client on the market works as intended. For the mobile application of Hygiene Score maintaining the latter one will be through the updates through Android Play Store. The former one is related to the server side of the system. Maintaining server side operations will be our responsibility as long as the application is being used.

**Platform Compatibility**

Hygiene Score will be developed as an application that is compatible with multiple platforms. The users will be able to use it as a web-application on Windows and also as a mobile application on Android phones.

**Reliability**

The system is divided into certain subsystems which will depend on each other and give error message depending on the exception. When the subsystems are integrated, they will work without any failures and they should be working correctly when they are working as single components.

**Perfomance**

This application aims to bring the necessary information to users as fast as possible with minimal number of errors and it should support a high number of users without any failures.

**Security**

Since system stores critical personal information such as passwords, they will be protected by the use of encryption. Properties in MySQL will be modified to satisfy this goal.

## 1.3 Definitions, acronyms, and abbreviations

GUI:Graphical User Interface

UI: User Interface

SQL: Structured Query Language

HTTP: Hyper-Text Transfer Protocol

## 1.4 Overview

This project aims to show the information gathered by the professionals during an inspection of a food establishment to the general public. The customer will also take part in evaluation process as well, although they will have less impact than the professionals. To use the system having an account is not necessary however the functions of the system will be limited. By creating an account a user can have multiple functions such as rating a restaurant, watching tutorials. The functionalities will differ by the account type of the user and the impact of the evalution will differ as well with inspectors having more impact than regular customers. Also, the customer will only have limited aspects to evaluate on.

# 2. Current Software Architecture

There are similar apps to Hygiene Score such as Zomato, Yelp and Foursquare. However, none of these apps use the data gathered from professionals only from the customer’s of the restaurant. Hygiene Score enables both the user and the professional to take effect in certain aspects of the food establishments, with professional evaluation having more impact, resulting in a better evaluation of the mentioned food establishment. The application will also contain tutorials to provide information to the customers and owners of the restaurants about the process of the evaluation of a food establishment in terms of hygiene. Commenting on the restaurants is a common feature of the products mentioned in the start of this section and will also be included in the functionalities of Hygiene Score.

# 3.Proposed Software Architecture

## 3.1 Overview

## Hygiene Score will be a web application and an Android application. For Android, we will be using Java as implementation language. For the web application, PHP, HTML and Bootstrap will be used. We will be using the MVC design pattern and client-server architecture. MySQL will be used to implement the database.

## 3.2 Subsystem decomposition

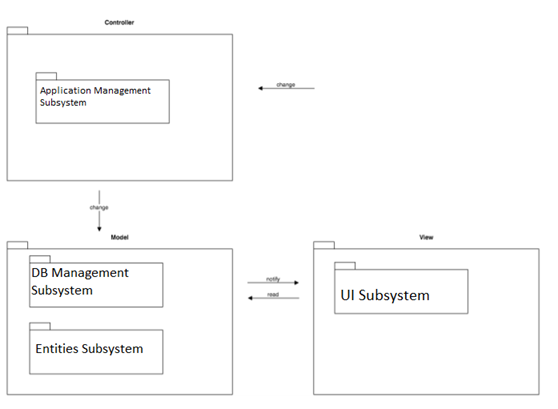


Figure 1: Subsystem Decomposition of Hygiene Score

We will be managing our subsystem decomposition by acquiring two different architectures together. Namely these architectures are, the client-server architecture and the MVC architecture. We will be using the MVC architecture to compose the smaller subsystem components of client-server architecture.

We will be connecting with our users through the client side of our system. Our Android application and website will be representing the client side of the client-server architecture. Mainly this side represents the user interface of Hygiene Score. Our Android application will be getting data of restaurants and it will use these data to generate the UI content accordingly. Furthermore, the user input, that is the grading of restaurants or the profile setup, is given through the android application interface and send to the server over mobile internet.

In the server side our system will be composed of a database controller and the corresponding manager that is for communicating with the database. In the server side there will be a database, a database controller and a communication manager. In this way, with dividing the server side our system will have modularity. Database will be where we will store everything including user profiles, comments, scores, restaurant details, and photos if there are any. The users will be actually using the database through the database connection manager of our system when they are using Hygiene Score’s application and website. We are already familiar with PHP and MySQL so we will be going with these as our DB of choice. Database manager is responsible for database operations such as deleting data, updating data etc.

## 3.3 Hardware/software mapping

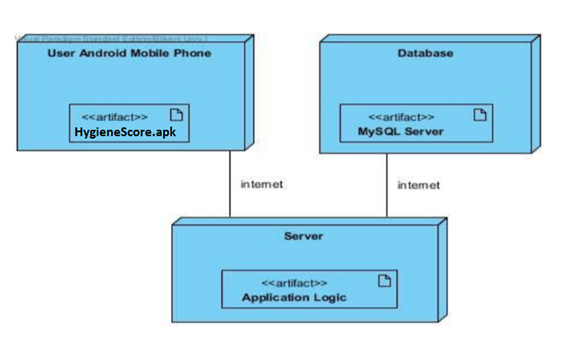
****

Figure 2: Hardware/Software Mapping

In Hygiene Score, the connection between hardware and software parts of the system will be handles with a client-server approach. Our system shall have a server which will be working continuously and also it shall have a database that will gather and track all the relevant data coming from web sites. These data will be used instantly in our Android application and website. In order to use our application, our users should install the HygieneScore.apk priorly. They should also have continuous internet connection for the system to be able to gather content from the server. The server will display the relevant data for the specific user through the internet.

## 3.4 Persistent Data Management

Since Hygiene Score can be classified as a social media application that includes education modules, persistent data of Hygiene Score can be classified as user data and education module data. User data can be classified into two further categories, which are customer’s and restaurant owners.

        Restaurant customer data will consist of personal information, messages, comments, likes/dislikes for comments, score given to a restaurant.

        Restaurant owner data will consist of restaurant information, comments, payments made for the education module designed for restaurant owners, last visited education module information, education quiz results, restaurant’s score and past restaurant scores.

        Education module data will consist of different videos and it will hold last visited education module information and education quiz results of a user, along with payments made for the module by the user.

        To keep all the data described above, we will use a database (MySQL). Since Hygiene Score will be a multiuser application, there will be a lot of data exchange going on and in order not to frustrate the user, Hygiene Score’s interaction with the database needs to be rapid and efficient. Additionally, the transactions of each user will be logged in order to increase fault tolerance and recoverability of data in case of a failure.

## 3.5 Access Control and Security

Hygiene Score is a multi-user system and each user will have different kinds of access levels. The table 1 below is an access matrix showing some of the actions each user type can do. Guest is any user without a login and customer is the restaurant customer who logged in and owner is the restaurant owner who logged in.

|  |  |  |  |
| --- | --- | --- | --- |
| **Action\User Type** | **Guest** | **Customer** | **Owner** |
| See Hygiene Score | + | + | + |
| See Comments | + | + | + |
| Write Comments | - | + | + |
| Pay for Education Module | - | + | + |
| See Hygiene Score Progress Graph | - | + | + |
| Like Comment | - | + | + |
| See Inspection Details | - | + | + |
| Alter Restaurant Information | - | - | + |
| View Restaurant Information | + | + | + |
| Add/Delete Restaurant | - | - | + |
| Delete Account | - | + | + |
| Give Hygiene Score To a Restaurant | - | + | - |

Table 1: Granted Permissions to certain actions of users matrix

To provide security, the HTTP authentication [1] will be used for securing login and for database protection, database auditing [2] will be used and precautions for SQL injection [3] will be taken. To provide security in payments, PCI protocol [4] will be used.

## 3.6 Global software control

For Hygiene Score we plan to use event-driven control as the type of the control flow. Another option for control flow is procedure-driven control. However, we do not prefer this approach because it needs objects with concurrency inheritance to be mapped to the flow of control. Therefore, it makes flexibility lower than event-driven control and it is harder to maintain the sequence for the large number of objects. The reason we prefer the event-driven control is that user actions such as mouse clicks and key presses determine the flow of the program.

## 3.7 Boundary conditions

Boundary conditions has three cases: Initialization,Termination and Failure.

Initialization: It indicates how the system will start.

* The user should enter to the web page of the Hygiene Score if he/she prefers to web application of the program. If they want to use the android application, they first should go to the Google Play Store and download the application to their phone.
* If the user has an account for Hygiene Score, they can login to the system by using their username and password.
* After this process, the user will be directed to the main page if they have used the correct username and password.
* Then, user can use the all the functionalities of Hygiene Score as they want.

Termination: It indicates how the system will shut down.

* Users of the Hygiene Score can log out from the system anytime they want regardless to what page they are on.
* When user presses the log out button, system will let user exit from the web and mobile application safely.
* System also enables the user to close the application (web-mobile) without logging out the system.

Failure: It indicates how the system react when faced with unusual activities.

* When there is a disconnection from the internet, Hygiene Score fails to process the user operations.
* For the mobile application, if the user does not update the application for a long time, system will not be able to keep up with the differences and show an error message regarding to the problem.

## 4. Subsystem Services

**Database Manager**

Database manager will be connecting the database to the application. The requests from the application/website will be delivered to the subsystem and to the database afterwards. It will also be responsible for the modification of the database such as removal, addition and alteration of data.

**UI Manager**

This subsystem will be responsible for the modifications to the GUI. This subsystem will have connections to the other subsystems. However, the changes done to the GUI will depend on the actions of this subsystem. It will handle the updates to the GUI and presentation of data to the GUI. It will gather data from wherever necessary, format it and present it to the user. After an update or request for an update, it communicates with the necessary subsystem to make necessary changes to GUI.

**Authentication Manager**

This subsystem will be responsible of the login procedure of the application and accessibility of the functions to the user. It will enable actions based on the type of the account. If no log in action is done the authentication manager will set the account type to default which is “Guest”. The tutorials shown to the user will depend on the account type from this subsystem.

**Tutorials Manager**

This subsystem will be responsible for the type of tutorials shown to the user, so it will work closely with the authentication manager. Based on the results from the authentication manager, the tutorials manager will show the allowed tutorials to the user. This subsystem will also work with the UI manager to handle the video requests from the user. This subsystem will also hold the information about the videos and textual information about the tutorials.

**Notification Manager**

This system is responsible for sending messages or notifications to the users of the systems. [5] These can be messages sent to them through the system’s messaging functionality or the occasional updates to them concerning their preferences or essential information about the application. These can be sent through the email or SMS texts services of the user. Also the notification tools provided by the Android system[6] can be used to inform the mobile application users through their mobile devices directly.

**Payments Manager**

This system is responsible for handling the payments made by the users to access functionalities such as tutorials. The system will authenticate the payment information given by the user and handle the payment transactions accordingly. It will inform the user through the notification manager about the status of the payment. A third party service or embedded service can be implemented for this purpose[7].

**Inspection Grading Manager**

This system handles the inspection and grading made by the customers and inspector type users of the application. The algorithm that decides upon the derived final grade of the system is used here. The input grades given by the inspector and other customers will be processed here based on the decision algorithm to output a final grade for the restaurant that will be published in the application.

## 5. Glossary

**Auditing** is the monitoring and recording of selected user database actions. It can be based on individual actions, such as the type of SQL statement executed, or on combinations of factors that can include user name, application, time. Security policies can trigger auditing when specified elements in an Oracle database are accessed or altered, including the contents within a specified object.

The **Payment Card Industry Data Security Standard** (PCI DSS) is a proprietary information security standard for organizations that handle branded credit cards from the major card schemes including [Visa](https://en.wikipedia.org/wiki/Visa_Inc.), MasterCard, American Express, Discover, and JCB. The PCI Standard is mandated by the card brands and administered by the Payment Card Industry Security Standards Council.

**E-commerce** (electronic commerce or EC) is the buying and selling of goods and services, or the transmitting of funds or data, over an electronic network, primarily the internet. These business transactions occur either as business-to-business, business-to-consumer, consumer-to-consumer or consumer-to-business.

6. References

1. "Basic Access Authentication". *En.wikipedia.org*. N.p., 2016. Web. 25 Dec. 2016.

2. "Database Auditing: Security Considerations". *Docs.oracle.com*. N.p., 2016. Web. 25 Dec.   2016.

 3. "SQL Injection Example And Tutorial". *Programmer and Software Interview Questions and      Answers*. N.p., 2016. Web. 25 Dec. 2016.

4. “Council". *Tr.pcisecuritystandards.org*. N.p., 2016. Web. 28 Dec. 2016.

5. "Notification system". *En.wikipedia.org*. N.p., 2016. Web. 25 Dec. 2016.

6. “Notification Manager”. Developer.android.com. N.p., 2016. Web. 28 Dec. 2016.

7. ”How to process and accept credit cards on your website”. Itworld.com. N.p., 2013. Web. 28 Dec. 2016.