High-Level Architectural Design
21 Questions

Gabriel Lopez de Leon - lopezdg - 1310514
Curtis Milo - milocj - 1305877
Maxwell Moore - moorem8 - 1320009
Alexandra Rahman - rahmaa25 - 1305735
Connor Sheehan - sheehacg - 1330964

March 7th, 2016

Contents

List of Figures

List of Tables

1 Introduction

This document is called the High-Level Architectural Design document and it will be giving the stakeholders, in this case the professor and the teaching assistants, a broad overview of how the project *21 Questions* is designed and will be organized.

1.1 Purpose

The High-Level Architectural Design document outlines the functionality of the system through Use Case diagrams, and details the key classes of the system and how they relate via the use of an Analysis Class Diagram. The main purpose of this document is to explain in detail the software system to be developed, which in this case is the application, *21 Questions*. Through the use of the various diagrams mentioned before, the main components of the product and their relationships with each other are shown. The main target audience for this document is the software developers as they need to see, in greater detail, how the system and its modules interact and how they are designed to work in relation to each other.

1.2 System Description

21 Questions is an android application that can be used as a location identifier whose intended use is for any user above the age of ten. The application requires minimal training, experience or technical expertise to use, and can be easily picked up and used by anyone. 21 Questions is a simple game that asks the user a series of twenty-one polar or binary questions to try to identify their area of interest. In this game the area of interest is limited to an establishment, building, place, or effigy with a focus on locations only with an end goal of displaying the result through Google Maps.

1.3 Overview

This document will outline the design of the *21 Questions* application from an architectural perspective. The document will begin from a use case outlook, outlining application functionality from a practical point of view and taking different actors and stakeholders into consideration. Next, an analysis class diagram and associated interpretation details is outlined, to specify application behaviours and resources in a modularized form. Following this section is a detailed architectural design as well as a set of class responsibility collaboration cards. These sections specify modules in greater detail, including interfaces to be implemented in the future. The order of these sections reflects a systematic progression from requirements to a more easily constructed application.

2 Use Case Diagram

a) User wants to enter new search (Figure 1).

BE1.1 The user presses the start button on the start screen.

BE1.2 The system will respond by bringing them to the question screen and asking them a question.

BE1.3 The user will answer yes, no or undecided to the question.

BE1.4 The system will ask the user another question.

BE1.5 After 21 questions, the system will respond by displaying a map to the user.

BE1.6 The user will hit done.

BE1.7 The system will ask if this was the location the user had in mind.

BE1.8 The user will answer yes it was or no it was not.

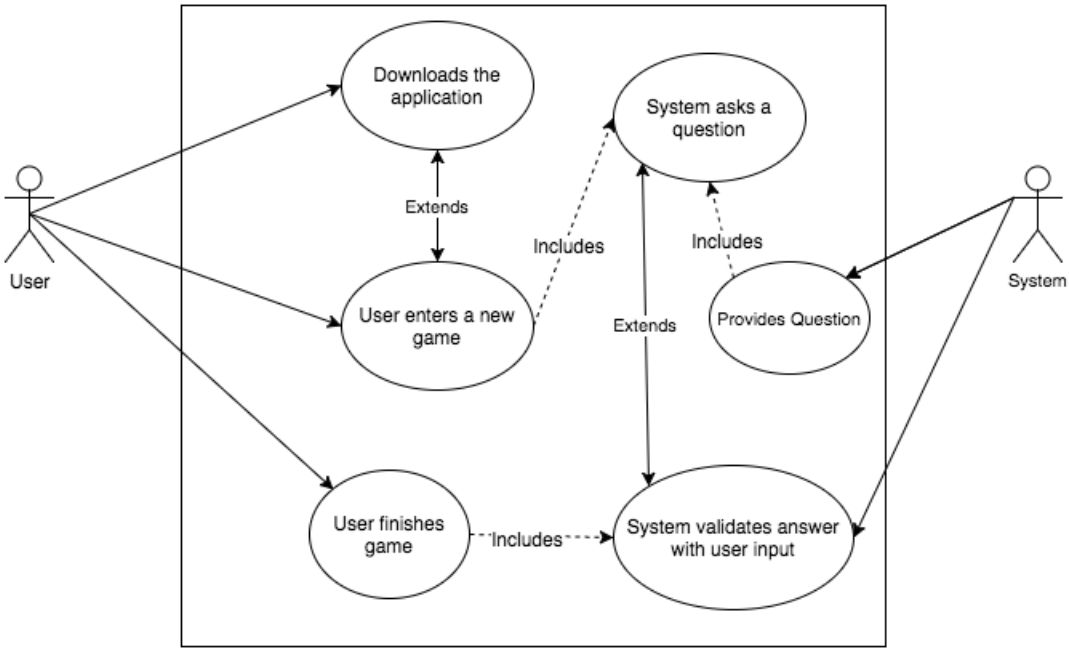

Figure 1: Use Case for BE1

b) An unlisted establishment requests to be included in the application (Figure 2).

BE2.1 A new business opens or a business opens a new location.

BE2.2 The business or establishment contacts the development team to inform them they wish to be added to the system.

BE2.3 The IT specialists will release a form for the business to fill out.

BE2.4 The business will return the form to the developers.

BE2.5 The IT specialists will verify that the information is valid and add it to the system.

BE2.6 The business is added to the system database for use in the application.

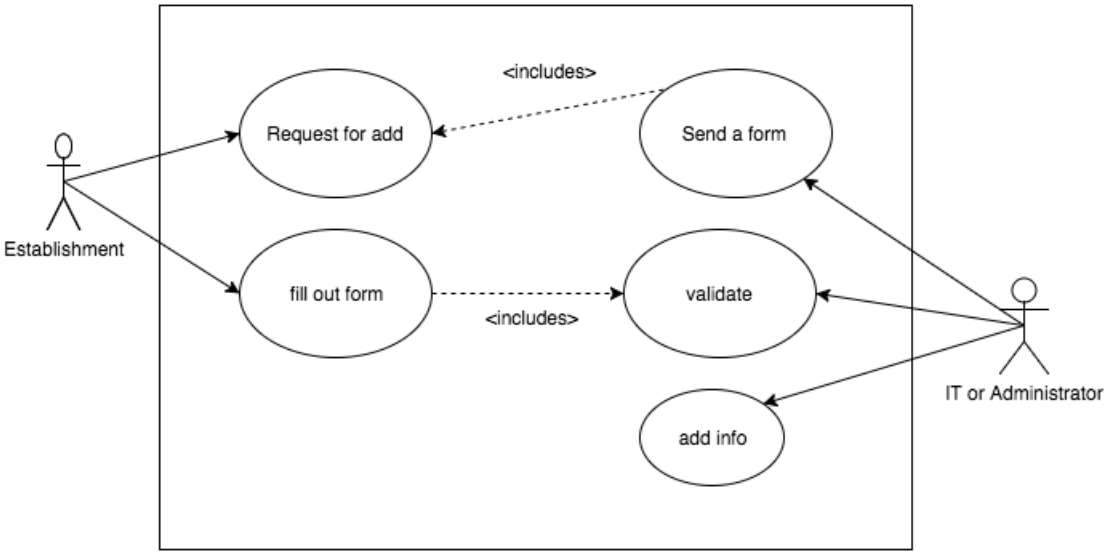

Figure 2: Use Case for BE2

c) Updates or maintenance of the application is required (Figure 3).

BE3.1 Internal management states an issue and requests an update/maintenance.

BE3.2 Update/maintenance is given a priority.

BE3.3 The IT specialist notifies the users that the system will update and be shut down for a certain period of time, if necessary.

BE3.4 The system will disconnect.

BE3.5 The necessary changes shall be made by the IT specialists.

BE3.6 The user will be notified if they need to update the application version.

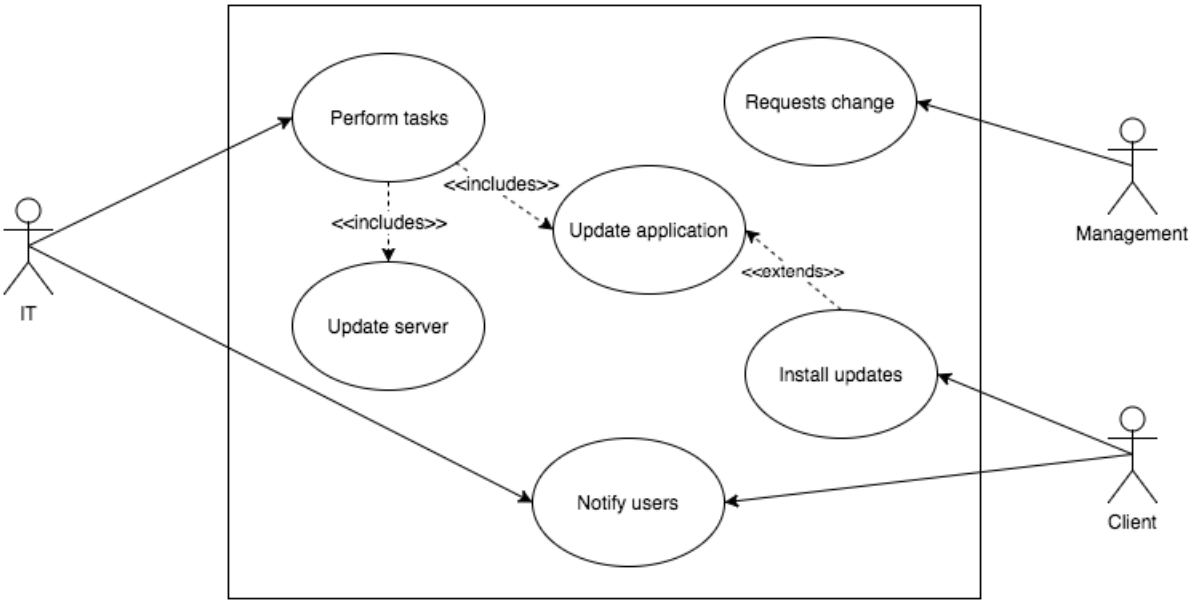

Figure 3: Use Case for BE3

d) Management requests implementation or change of experts (Figure 4).

BE4.1 Internal management states a functionality needs to be changed (added or removed).

BE4.2 The change is given a priority.

BE4.3 The function will be added or removed.

BE4.4 A small focus group is selected.

BE4.5 A survey is created.

BE4.6 Update is released to focus group.

BE4.7 Survey is sent to focus group.

BE4.8 The update will be released depending on the results of the survey.

BE4.9 If the survey results are not favourable, the function will be under review and released again to the focus group (repeat steps 5-9). Otherwise the update is released to the general public and the user is notified that they need to update app version.

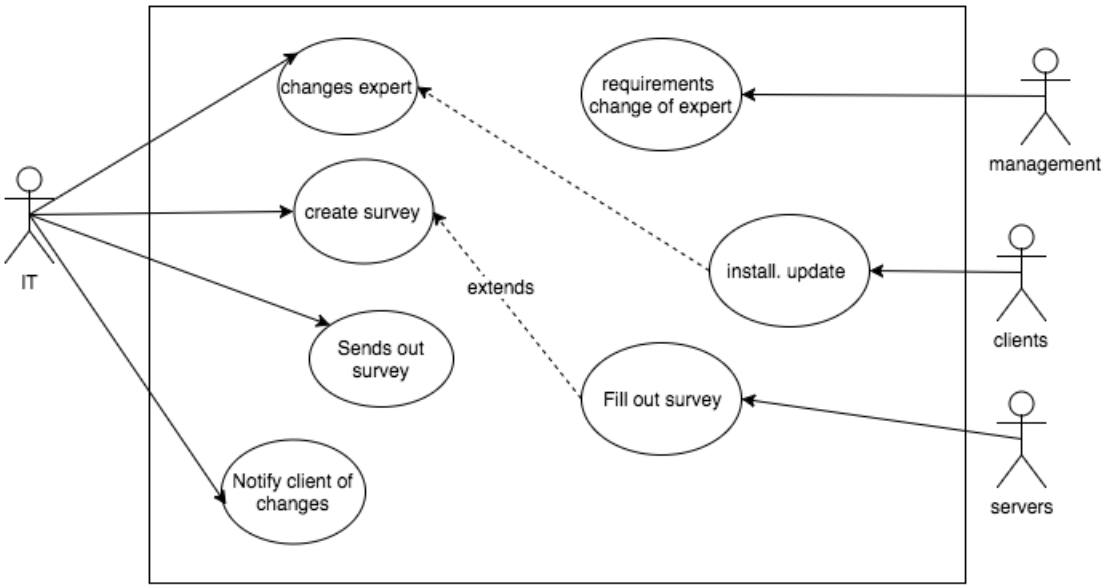

Figure 4: Use Case for BE4

e) User flags an incorrect or inappropriate search or result (Figure 5).

BE5.1 A business, user or internal management recognizes that the content is inappropriate.

BE5.2 The content shall automatically be hidden from other users.

BE5.3 The content is given a priority.

BE5.4 The content is put into a priority queue.

BE5.5 An IT will review the content and make appropriate changes (remove if necessary).

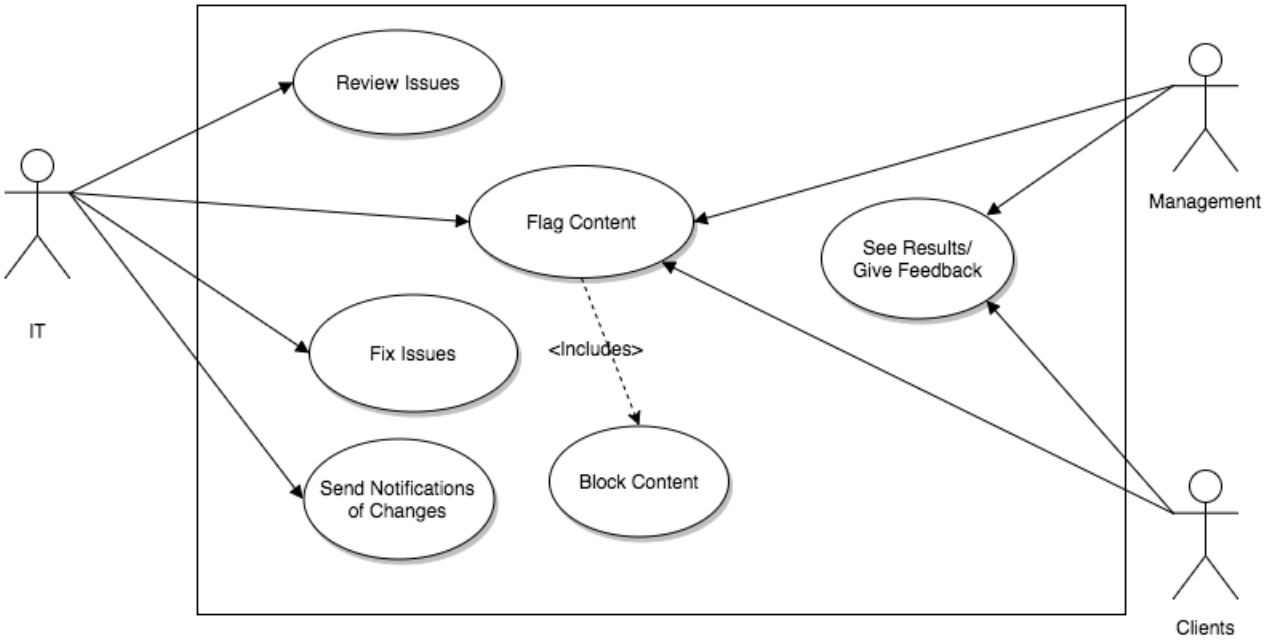

Figure 5: Use Case for BE5

3 Analysis Class Diagram

Figure 6 is an *analysis class diagram* for the *21 Questions* application.

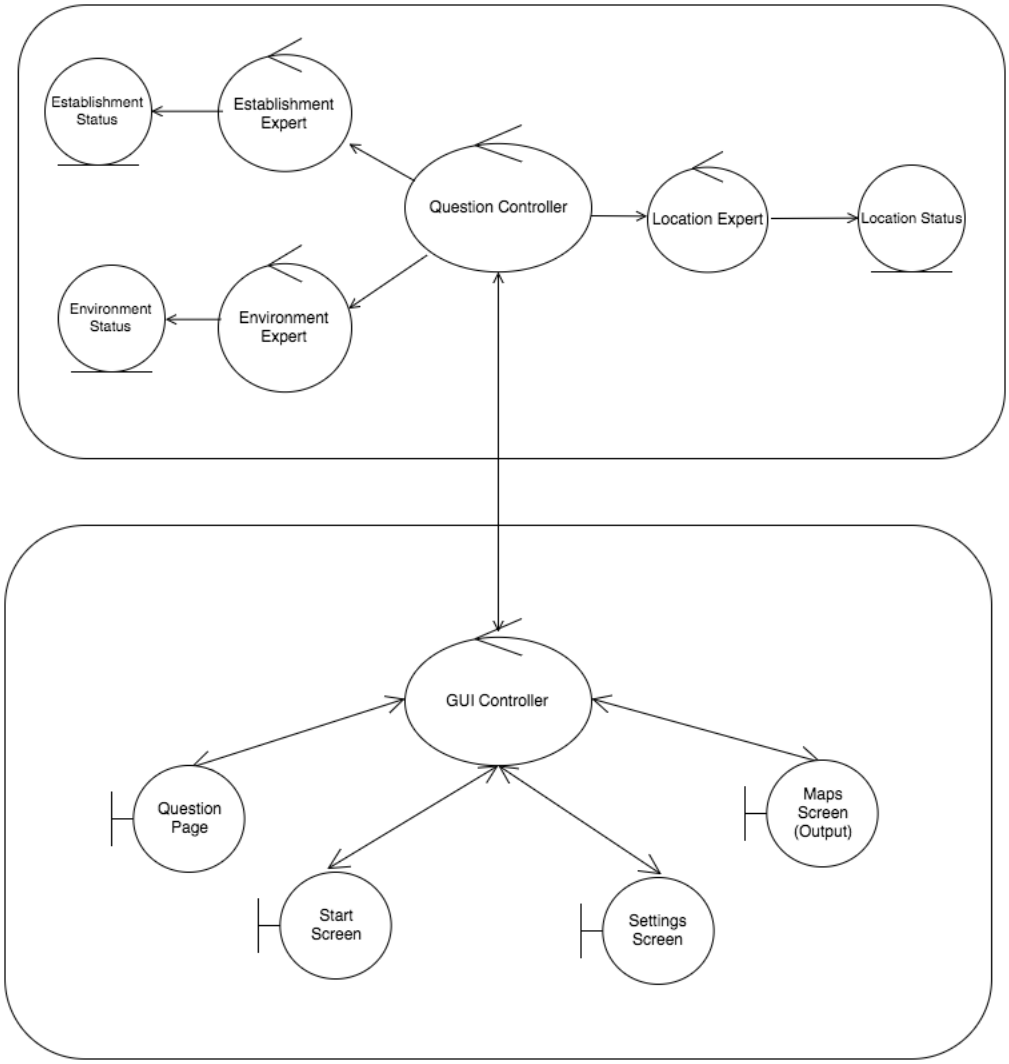

Figure 6: Analysis Class Diagram

4 Architectural Design

The desired architecture for this system is the blackboard architecture pattern. The blackboard pattern involves a *blackboard* global memory space, a set of specialized knowledge modules and a set of controllers to synchronize interaction between the modules. The system will use three experts as the knowledge modules. Each expert will expose its own interface to be interacted with by controllers. Each expert will be able to communicate with a globally shared database. The database is essential to ensure that multiple clients will have access to the same information without having to perform updates.

4.1 System Architecture

Figure 6 contains the analysis class diagram. The diagram is designed so that each entity or boundary class will communicate with their corresponding subsystem controller, and the subsystem controllers will handle communication between each subsystem. The design is separated into three logical units: interface, logic and data. The system design is a black board model with a data store, with the experts all using the same interface so that they can properly communicate to the question controller class. The system will have several controllers, one for handling the pages of the system and another for controlling which expert can ask questions. Each expert will create a guess of what they think is the item in which the user is thinking of. The controller will ask one expert at a time to provide a question to ask the user, the users response will bring the system closer to the correct result. The experts will then provide their best guess to the controller, the controller will then compile these answers into a result which will be displayed on a map for the user. In this way our system will not rely on a single expert, making it easy to add and remove experts in order to provide a better result.

4.2 Subsystems

The database controller will deal with different clients asking for unique information. This entitles that it will process any information, run the query that is needed, and return the information that is related to the problem.

The next tier to the system is the logical layer. This layer is responsible for the control flow of the game, ensuring that each expert entity will properly get to ask their respective questions. The controller will deal with receiving the information that is passed in from the GUI controller and dealing with incoming information from the data layer. Using the guesses form each of the experts, the controller will finally create a best guess solution to the problem.

Finally the GUI controller, which is responsible for ensuring the proper screen is shown. This controller receives, process and sends information from the view screens to the question controller. The GUI controller uses information from the logic controller in the questions or map boundary class. The start screen, which will show the initial menu a user will be viewing, takes user input to start a game or change the setting of the game. The question controller will display the questions for the user to answer and will pass off this information to the GUI controller. The setting screen will provide a sub-menu which would allow a user to change the functionality of the system in regards to accessibility, font colors/sizes and preferences. Finally the map screen will be where the user is provided with a map of what the final result is and will take feedback from the user on the correctness of the result.

5 Class Responsibility Collaboration (CRC) Cards

Class Name: Expert Controller	
Responsibility:	**Collaborators:**
• Deals with messages passed from the Question Controller • Passes desired questions to the Question Controller • Creates and deletes tables that relate to experts • Accesses the information by performing a query on the tables • Adds information to tables • Modifies information in the tables	• Establishment Table • Environment Table • Location Table

Table 1: CRC for Expert Controller

Class Name: Question Controller	
Responsibility:	**Collaborators:**
• Requests questions from the Expert Controller based on the expert's needs • Accesses questions that have already been asked as well as the answers to said questions • Keeps track of the overall state of the game. This includes the number of questions total that have been asked • Provides a solution based on the experts' best guesses • Process answers from the GUI Controller • Provides the next question for the GUI Controller to display	• Expert Controller • Facts Data • GUI Controller

Table 2: CRC for Question Controller

Class Name: Graphics User Interface Controller	
Responsibility:	**Collaborators:**
• Controls which boundary class the user can interact with • Processes and verifies information to send to the Question Controller • Receives information from the Question Controller to display information or direct to answer • Tracks the current and past state	• Question Controller • Map Screen • Start Screen • Question Screen • Setting Screen

Table 3: CRC for Graphics User Interface Controller

Class Name: Start Screen	
Responsibility:	**Collaborators:**
• Provides an interface that the user can interact with • Sends information to the GUI Controller	• GUI Controller

Table 4: CRC for Start Screen

Class Name: Settings Screen	
Responsibility:	**Collaborators:**
• Displays a variety of settings to the user • Allows the user to verify the changes to the current settings • Allows the user to go back to previous screens	• GUI Controller

Table 5: CRC for Settings Screen

Class Name: Question Screen	
Responsibility:	**Collaborators:**
• Receives information from the GUI Controller • Takes user input from the users response for the question • Sends the answer of the question to the GUI Controller • Allows user to quit the current game	• GUI Controller

Table 6: CRC for Question Screen

Class Name: Map Screen	
Responsibility:	**Collaborators:**
• Informs the user of the guesses on a map • Informs the user of the guesses as an address • Receives and passes information from the GUI Controller • Requests the user feedback on the correctness of the guess	• GUI Controller

Table 7: CRC for Map Screen

Class Name: Location Data	
Responsibility:	**Collaborators:**
• Holds the question that have been asked and their associated answers • Holds the number of questions the expert himself has asked • Holds most viable current guesses • Provides a probability in which the guess is correct	• Question Controller

Table 8: CRC for Location Data

Class Name: Environment Data	
Responsibility:	**Collaborators:**
• Holds the question that have been asked and their associated answers • Holds the number of questions the expert himself has asked • Holds most viable current guesses • Provides a probability in which the guess is correct	• Question Controller

Table 9: CRC for Environment Data

Class Name: Establishment Data	
Responsibility:	**Collaborators:**
• Holds the question that have been asked and their associated answers • Holds the number of questions the expert himself has asked • Holds most viable current guesses • Provides a probability in which the guess is correct	• Question Controller

Table 10: CRC for Establishment Data

Class Name: Location Database	
Responsibility:	**Collaborators:**
• Allows for the addition of a location • Allows for the removal of a location • Holds a set of questions that correspond to a certain location • Searches for location based on question results	• Expert Controller

Table 11: CRC for Location Database

Class Name: Environment Database	
Responsibility:	**Collaborators:**
• Allows for the addition of an environment • Allows for the removal of an environment • Holds a set of questions that correspond to certain environment • Searches for an environment based on question results	• Expert Controller

Table 12: CRC for Environment Database

Class Name: Establishment Database	
Responsibility:	**Collaborators:**
• Allows for the addition of an establishment • Allows for the removal of an establishment • Holds a set of questions that correspond to certain establishments • Searches for an establishment based on question results	• Expert Controller

Table 13: CRC for Establishment Database

A Division of Labour

Team Member	Contributions
Gabriel Lopez de Leon	Added Purpose (Section 1.1), created use case diagram for BE5 and assisted in writing the CRC.
Maxwell Moore	Wrote introduction(1.0), helped design BE's entered 1,3 into draw.io for submission, wrote original section 4 in its entirety, helped refine the class analysis diagram, as well as the CRC cards..
Curtis Milo	Generated the use case diagrams for every business event and the scenarios that correspond to them. Created and refined the class analysis diagram. Created and refined the CRC cards, helped input the class responsibility collaboration cards into the document. Finally edited the grammar of the document and ensured that everything was correct according to the design ideas proposed in the team meetings.
Alexandra Rahman	Wrote the system description and helped come up with the scenarios and use cases. Created the use case diagram for BE4. Collaborated on the CRC cards and the analysis class diagram. Refined the CRC cards and the analysis class diagram. Helped input the class responsibility collaboration cards into the document. Finally, helped edit and format the document.
Connor Sheehan	Created use case diagram for BE3. Added overview section. Added styling. Proofread final submission and clarified language.

Table 14: Division of Labour

Gabriel Lopez de Leon Date

Curtis Milo Date

Maxwell Moore Date

Alexandra Rahman Date

Connor Sheehan Date