*Checkpoint Management System*

1: Rafael M. Cabrera   
2: *Computer Engineering*  
3: *De La Salle University Manila*4: Dumaguete City, Philippines  
5: raffy\_cabrera@dlsu.edu.ph d 1: Kalvin Kenjo H. Yap  
2: *Computer Engineering*  
3: *De La Salle University Manila*4: Cebu City, Philippines  
5: kalvin\_yap@dlsu.edu.ph  
1: Daine Janssen R. Jadman  
2: *Computer Engineering*  
3:*De La Salle University Manila*4: Cebu City, Philippines  
5: daine\_jadman@dlsu.edu.ph

*Abstract*— The importance of maintaining information has been very paramount today due to the current pandemic the world is facing. The increasing number of patients varies for countries, however, there is a high increase of patients in the Philippines due to COVID-19. Furthermore, this calls for the need to be able to have a suitable computer programmed system implemented using JavaFX in order to aid in combating the spread of the virus. This program aims to achieve a structured approach to aid the need of information management especially with regards to assist a community. Managing information is the optimal way to make a healthy working environment. The main features of the program were a success which contained all data structures needed to implement the program. As a result, the program enables users to have an easier time to access important information rather than manually storing the information. Consequently, digital information storing provides more flexibility and efficiency compared to using traditional means of recording data.

Keywords—digital, storing, COVID-19, database, JavaFX

# Introduction

## Significance of the Study

The ongoing pandemic, with its confirmed cases rapidly changing daily[1], has brought upon measures that require sustainability and effectiveness in order to combat this dilemma. Therefore, various innovations were created to assist in the easing of numerous issues that have been brought upon by the pandemic. A relevant problem that had been focused and considered is the absence of a digitalized, systematic, and proper application that records the activity and movement of visitors and residents passing the gates or checkpoints of a subdivision[2]. Hence, this study provides a solution to ease several underlying complications of checkpoint monitoring which are looking for previous entries of a certain individual, the recording of individuals quickly, the contactless recording of information, and the monitoring of households going over the allowed limit of individuals passing through[3].

## Definition of Terms

1. Class – It is used in creating an object in an object-oriented programming language.
2. Function – A set of instructions used to create output from input of the programmer.
3. Data Structure - A data structure is typically designed to store data with different types and various algorithms.
4. JavaFX - A software platform for creating and delivering desktop applications, as well as rich Internet applications that can run across a wide variety of devices[4].
5. FXML - A user interface markup language created by Oracle Corporation for defining the user interface of a JavaFX application.

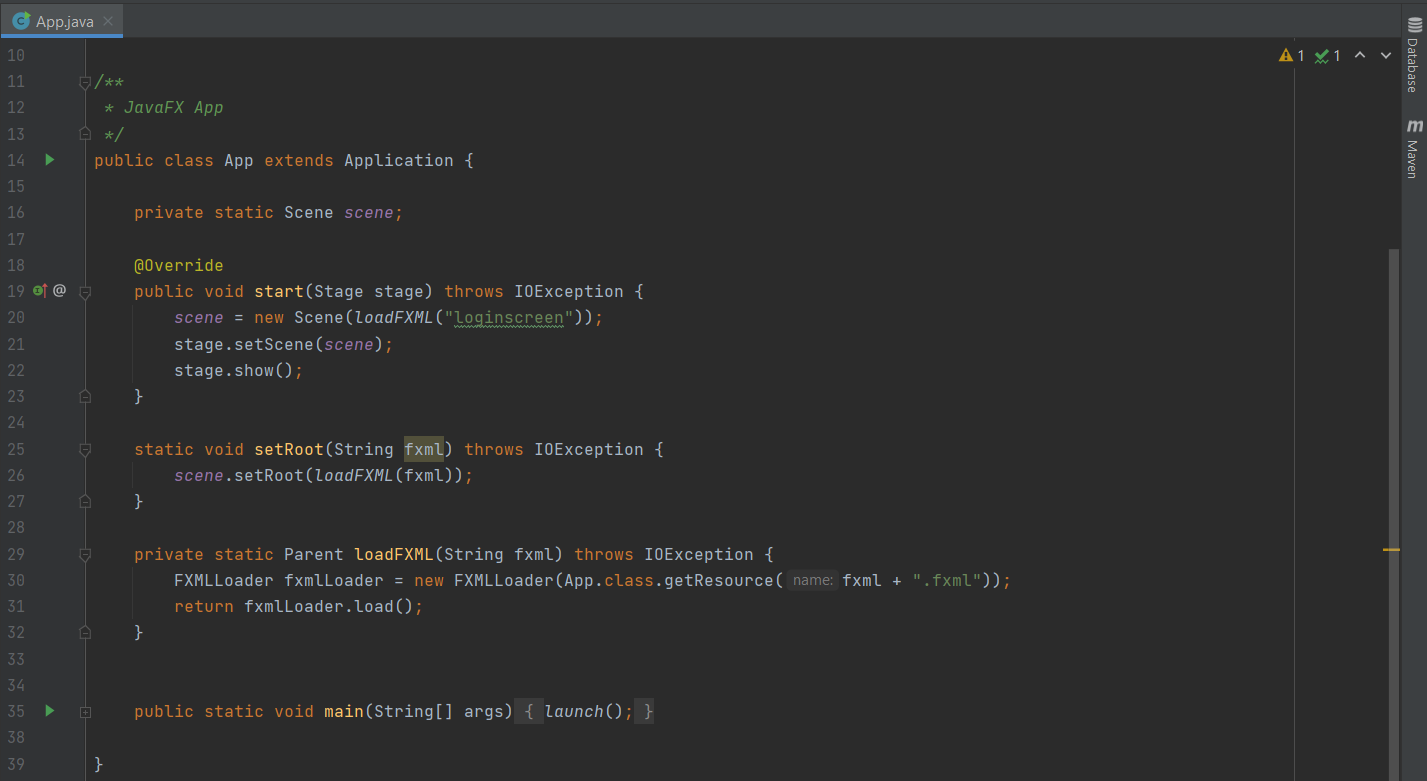
# Related Work

Object-oriented methods represent some of the most promising ways of meeting the demands of the most advanced applications. That is why, in recent years, a growing number of demographic and urban microsimulation systems are being developed within an object-oriented framework. This paper presents a microsimulation model defined in an extension of the Unified Modeling Language (UML) that facilitates the modeling of households[5]. In order to continuously record information, a log storing apparatus for investigating operations, the types of data handled, the contents to be processed, and for saving acquired information through the investigation are handled by the user[6]. The systems for recording, searching, and outputting display information are provided. Furthermore, the invention relates to a method and apparatus for recording and maintaining stored information system object relationship information. Information contained within a stored information system are searched to identify dependency relationships between objects. This object relationship information is stored and maintained in an information base[7]. In addition, the user may create, edit and manipulate the database (dynamically translated into a set of relational database structures), to create, edit and manipulate objects for that object, and to create, edit and manipulate queries to be applied to that object database[8]. The development of applications in the present day uses Java Programming or specifically JavaFX for its implementation. Additionally, the reasoning behind the usage of JavaFX is because of its wide variety, applicability, and availability of libraries and engines. Furthermore, applications using the Java library can be run in multiple platforms such as desktops, mobile devices, televisions, tablets etc.[9]

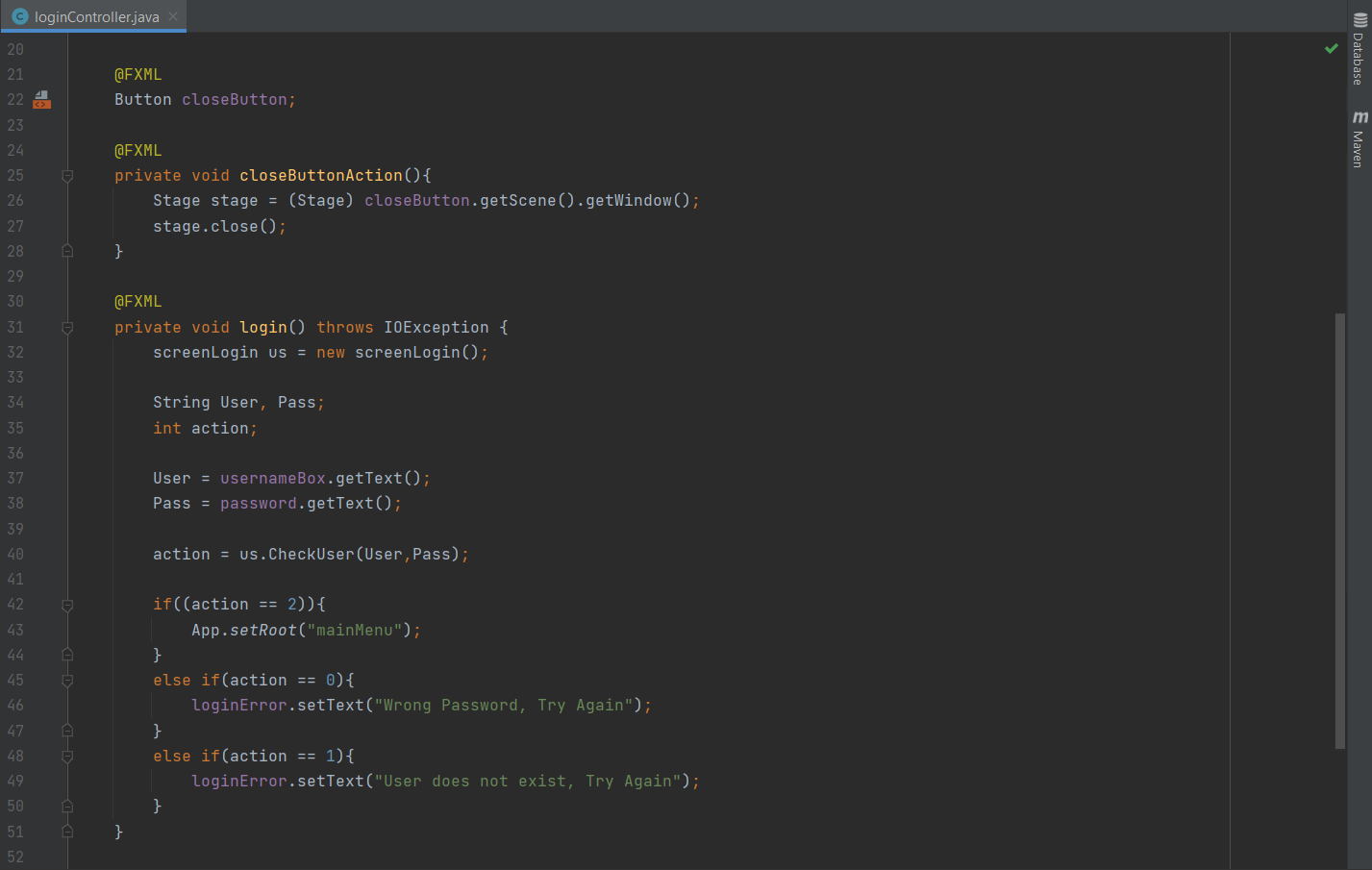
# Proposed Application and Methodology

The proposed application involved creating a program that allows the user to input information of the visitors and residents passing through the checkpoint. In this research, the checkpoints were targeted for this project since the number of individuals passing through are numerous.

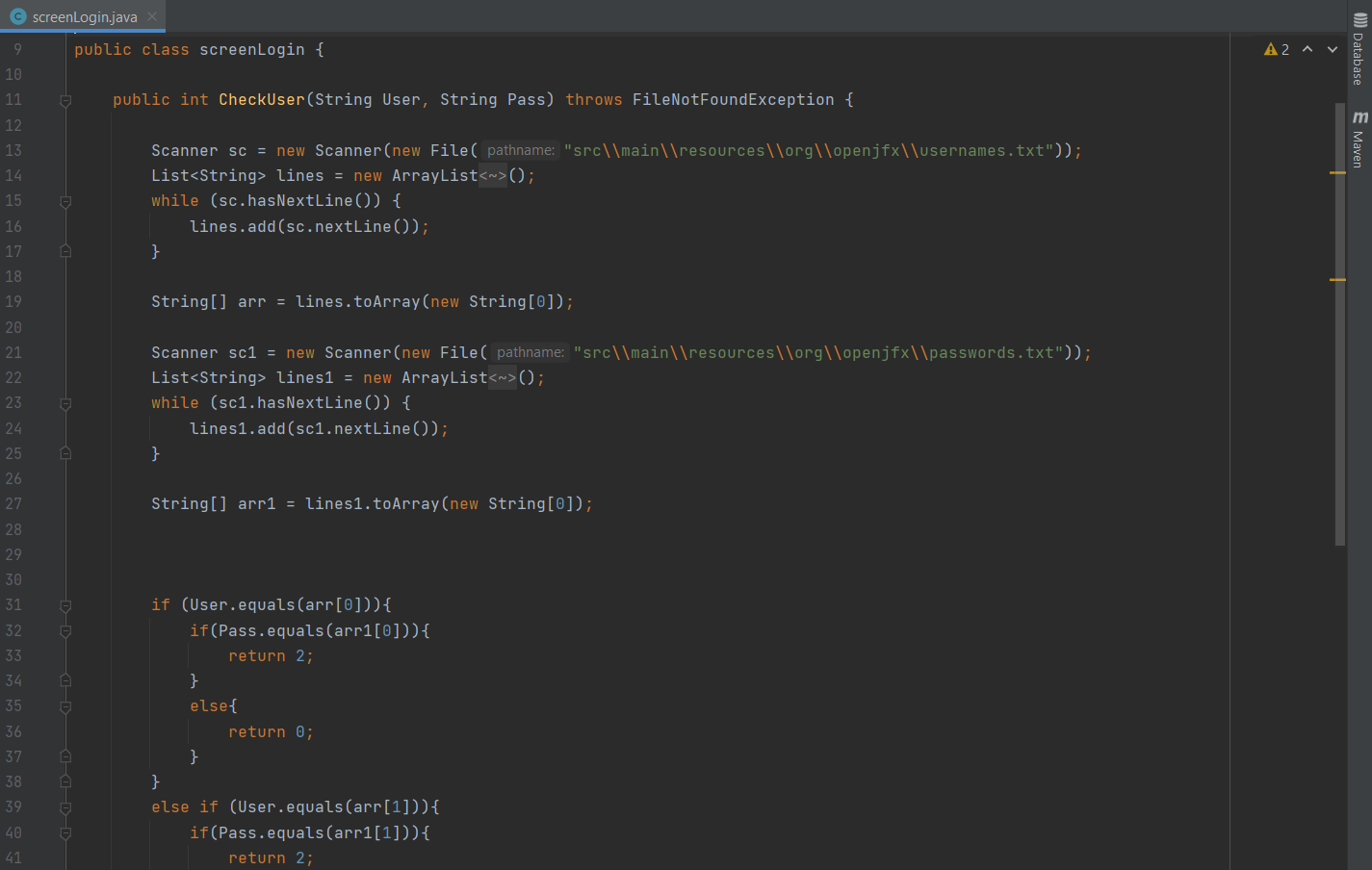
In addition, the methodology of this research paper starts with the front end of the program, the FXML file. Constructing the layout and GUI using Gluon’s Scene Builder application. Moreover, each implementation of the classes and data structures will be discussed in Part IV.



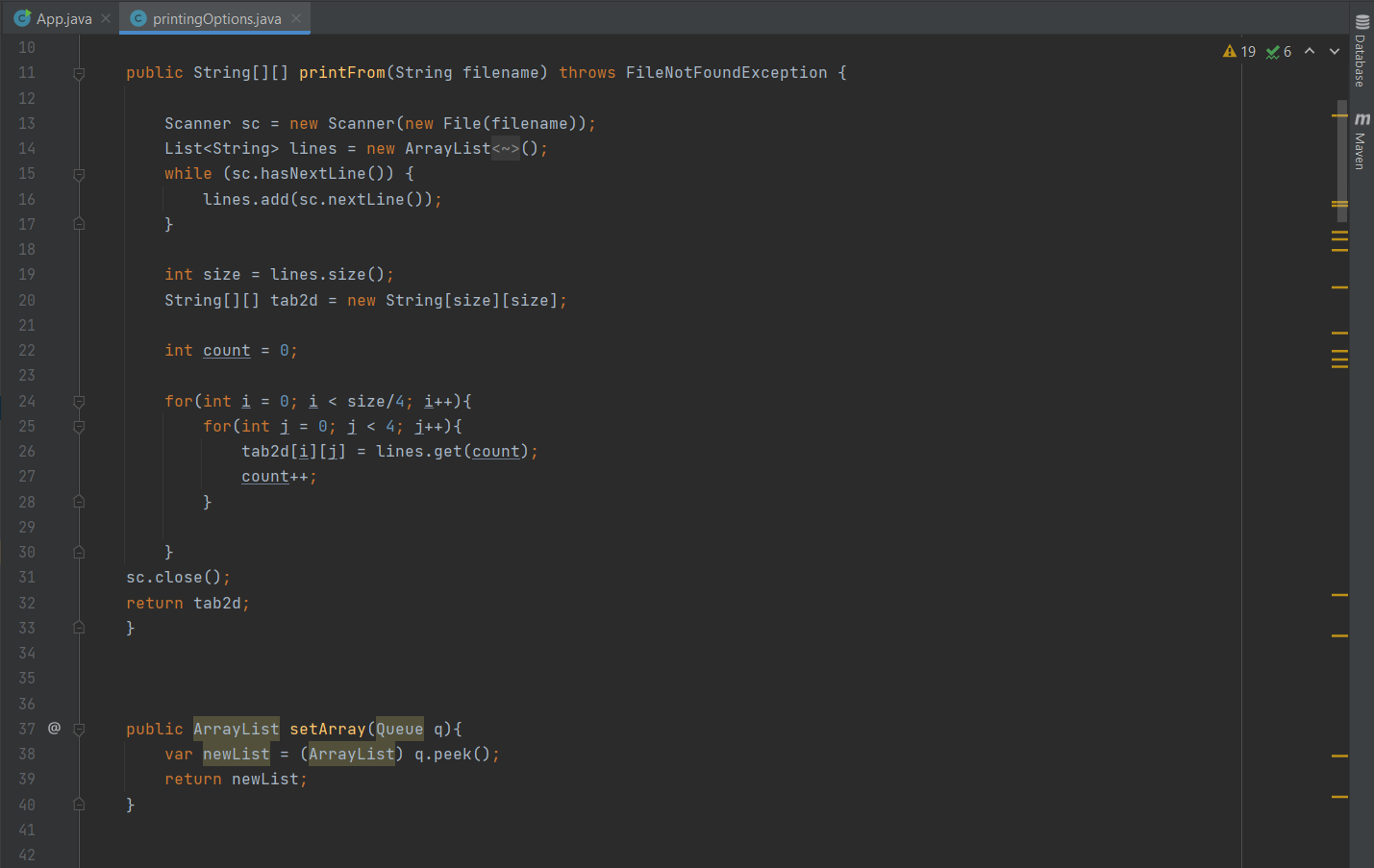
*Figure 1.1 Main Run Class*



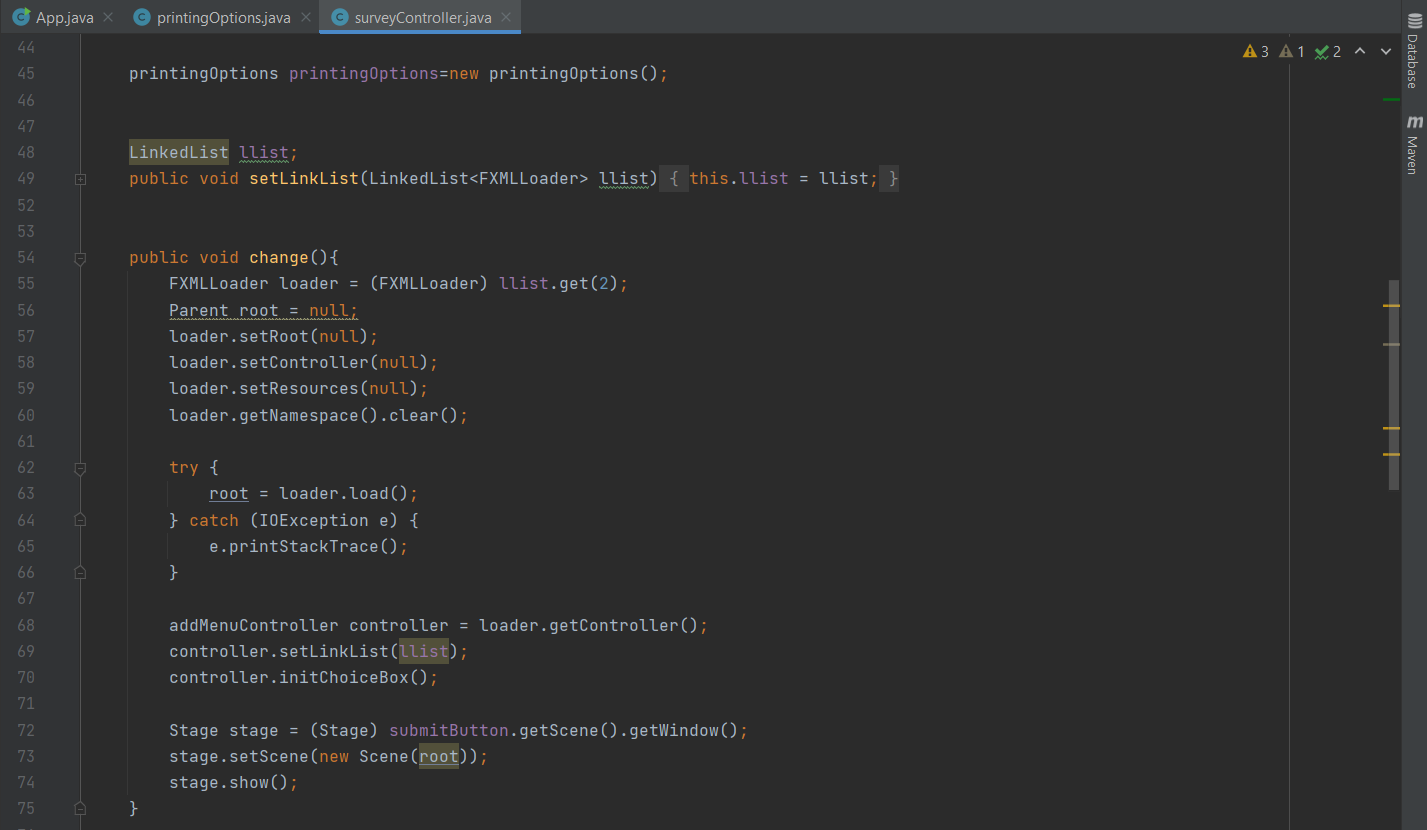
*Figure 1.2 Login Controller Class*



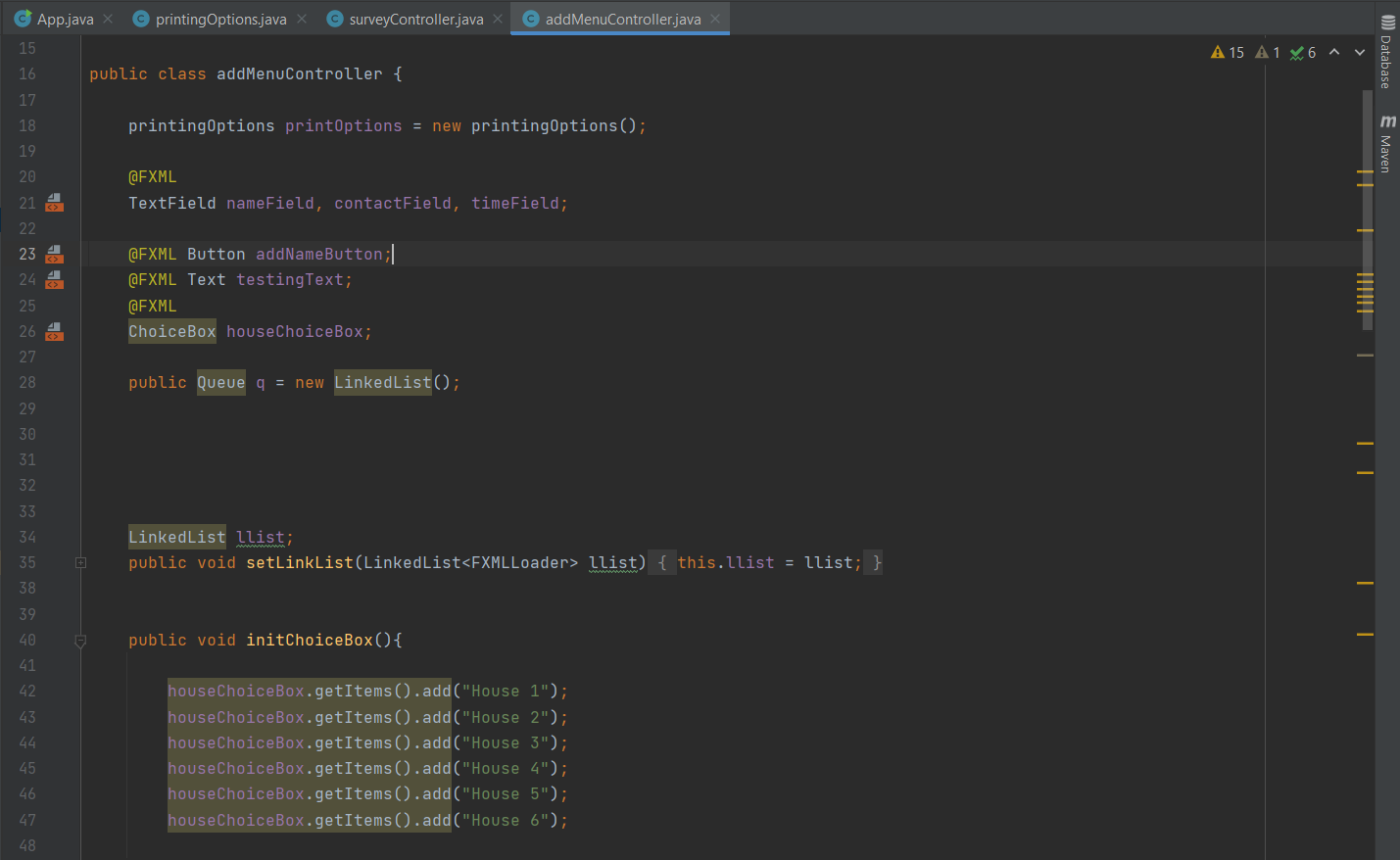
*Figure 1.3 Screen Login Class*



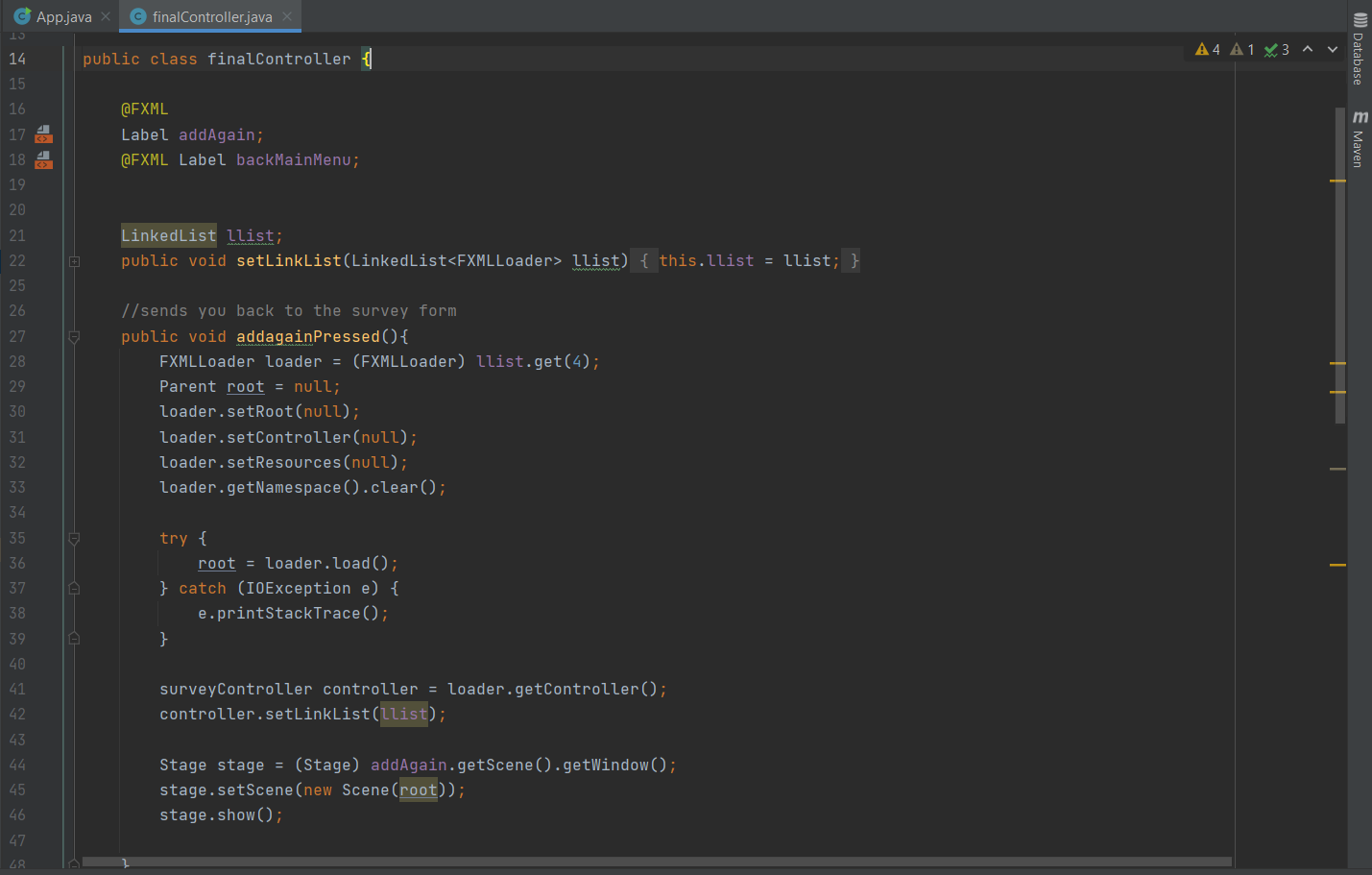
*Figure 1.4 Printing Options Class*



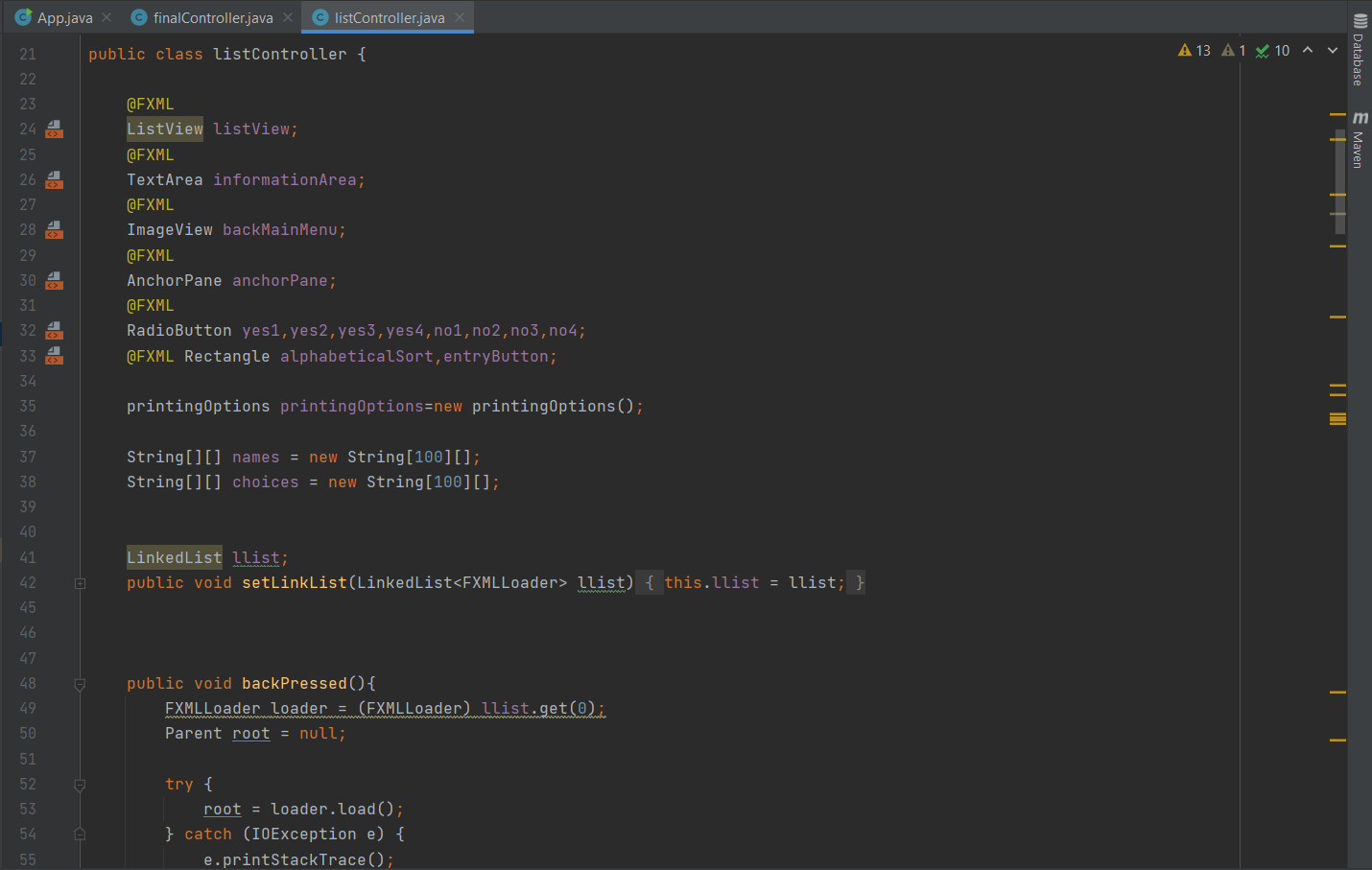
*Figure 1.5 Survey Controller*



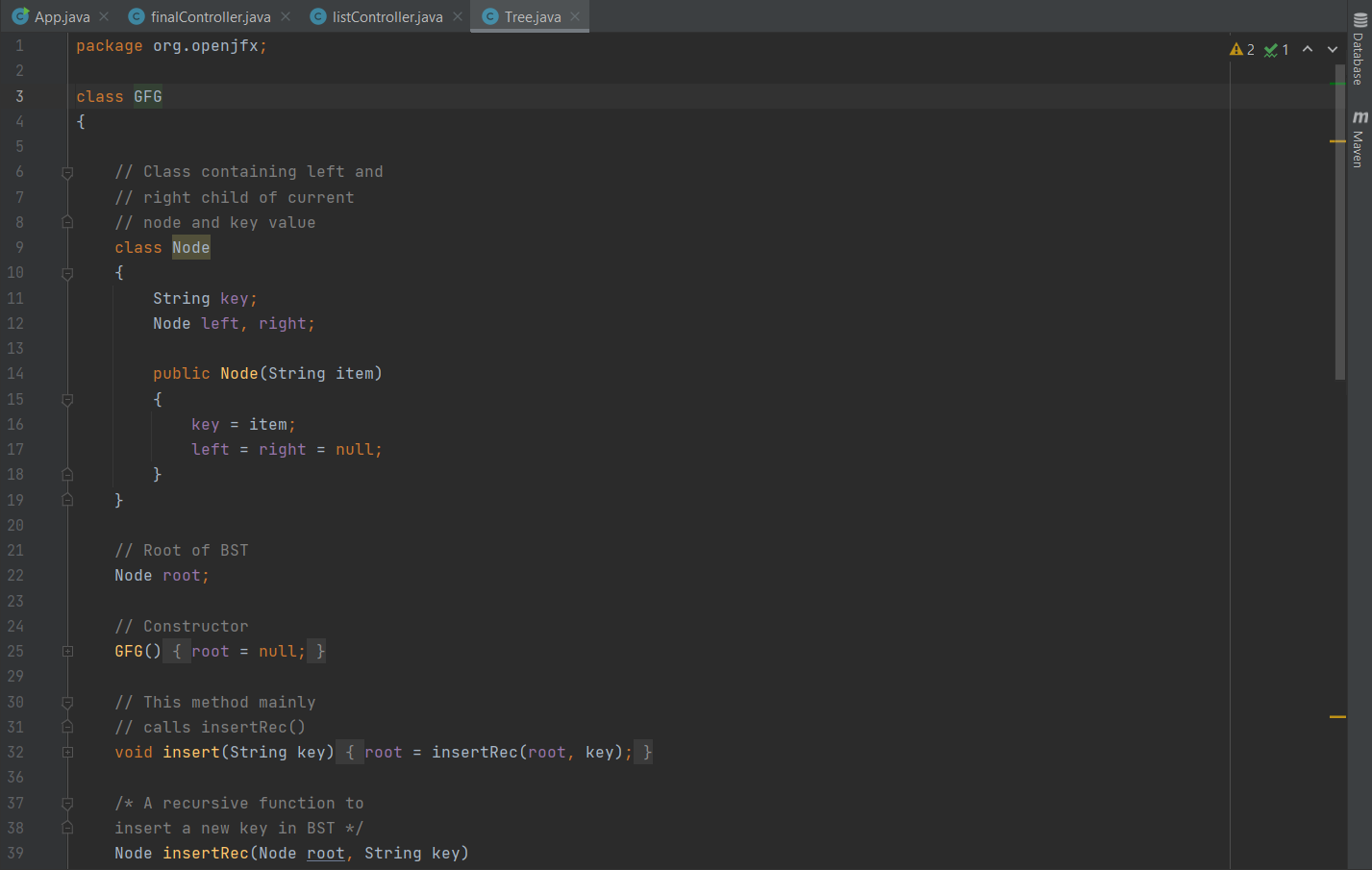
*Figure 1.6 Add Menu Controller*



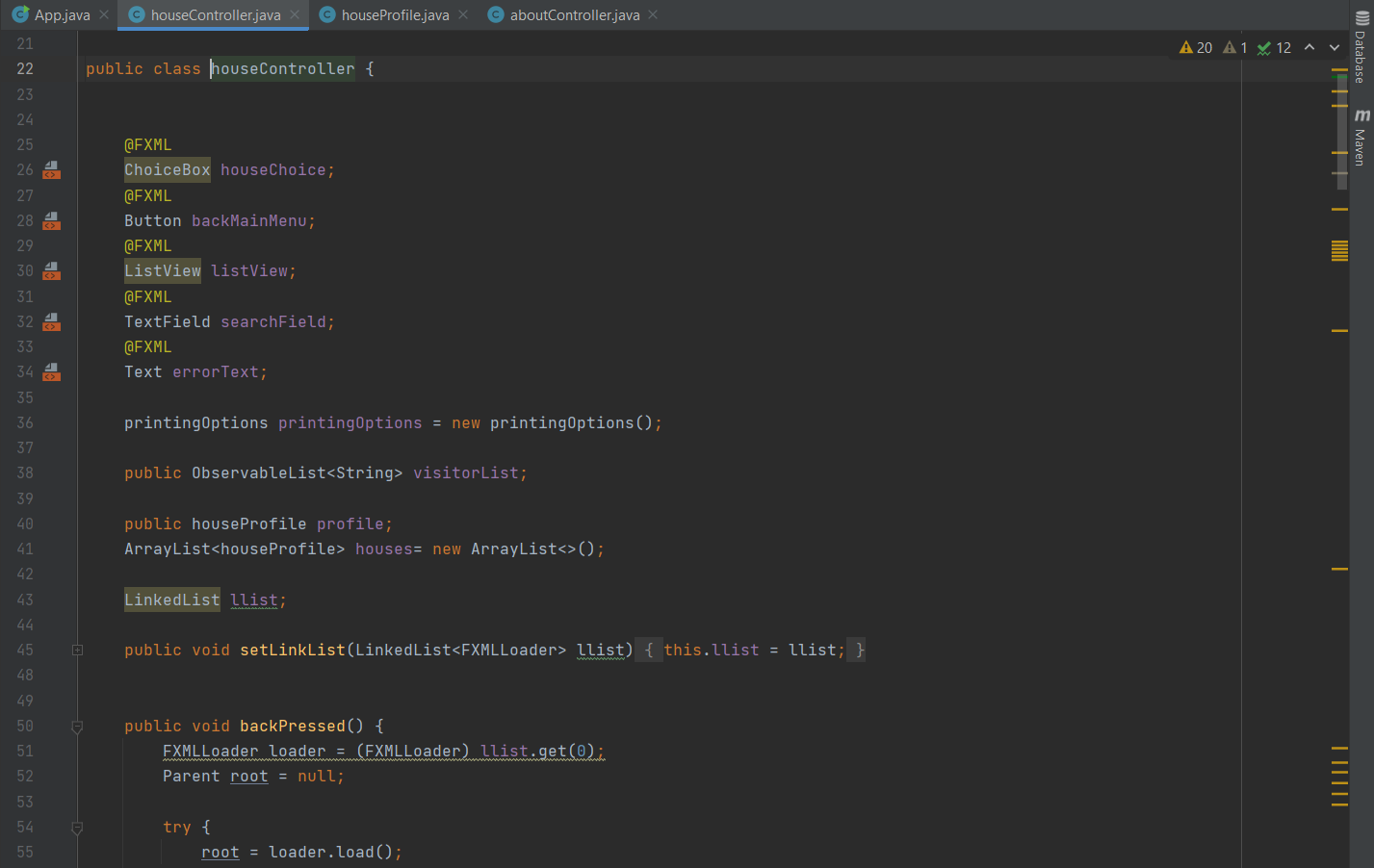
*Figure 1.7 Final Controller*



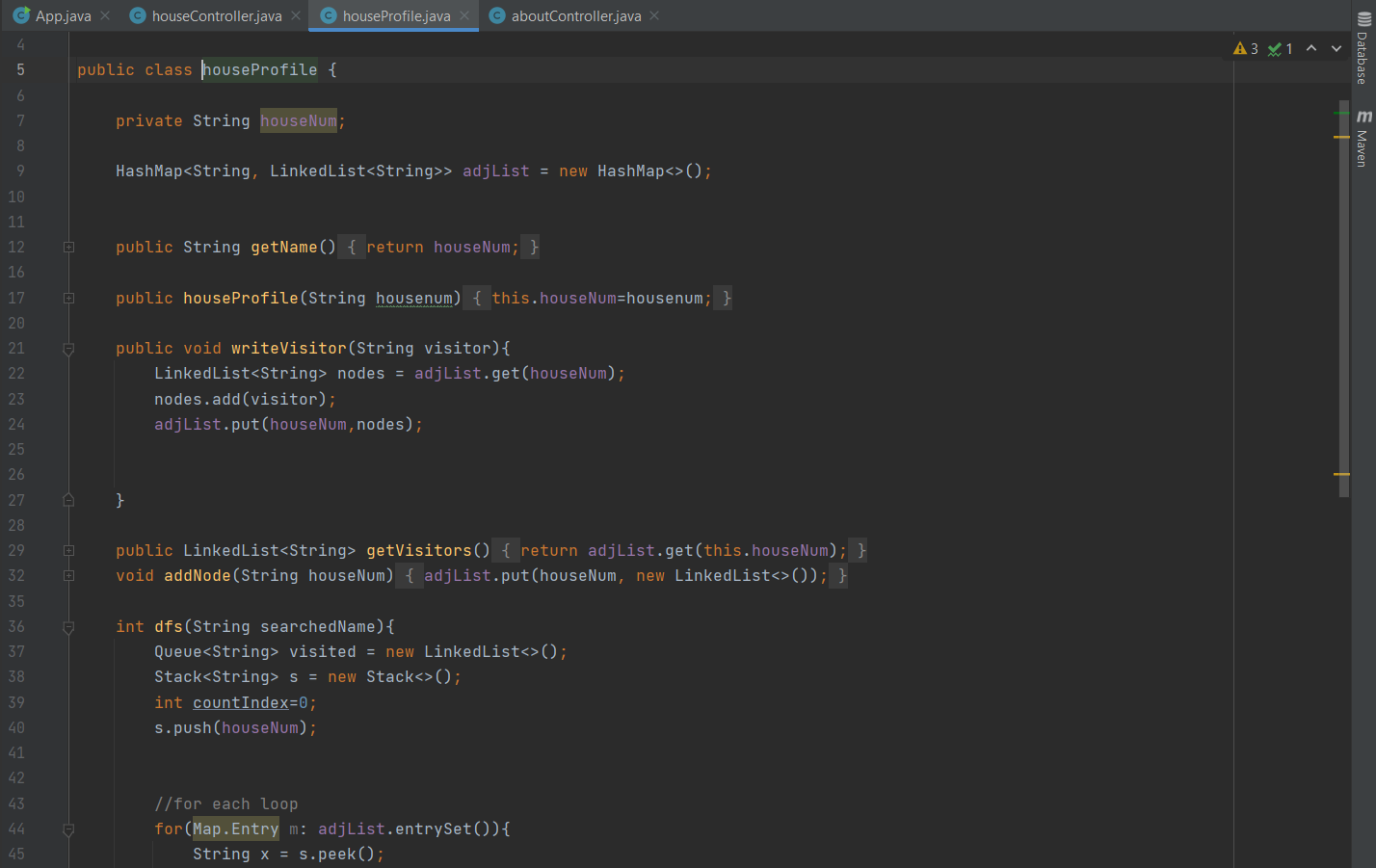
*Figure 1.8 List Controller*



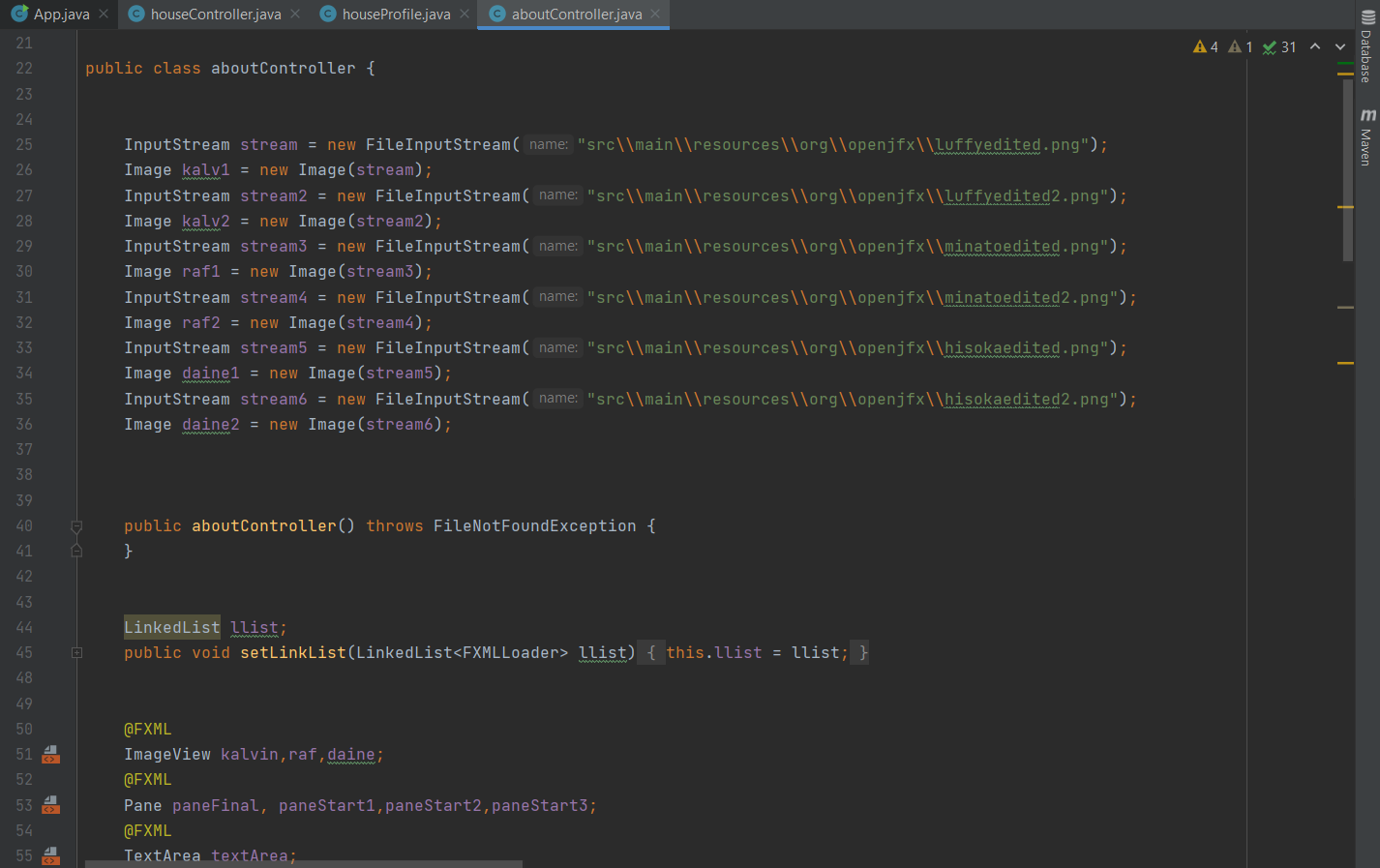
*Figure 1.9 Tree Class*



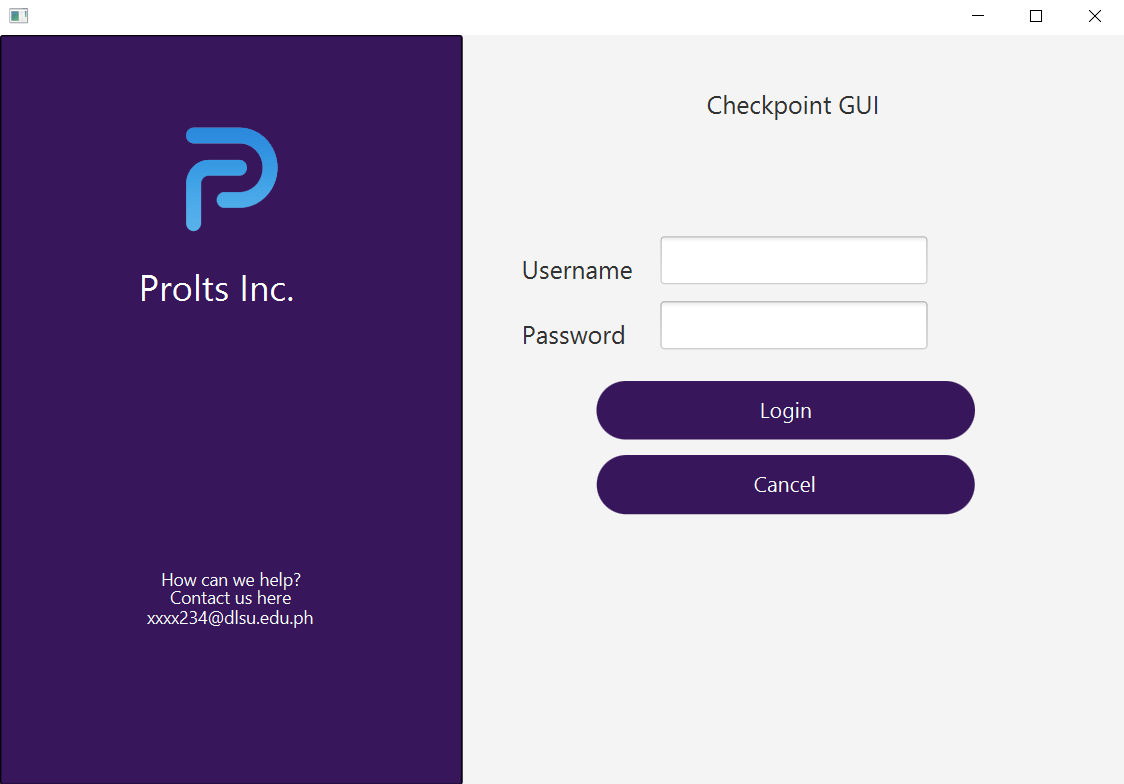
*Figure 1.10 House Controller*



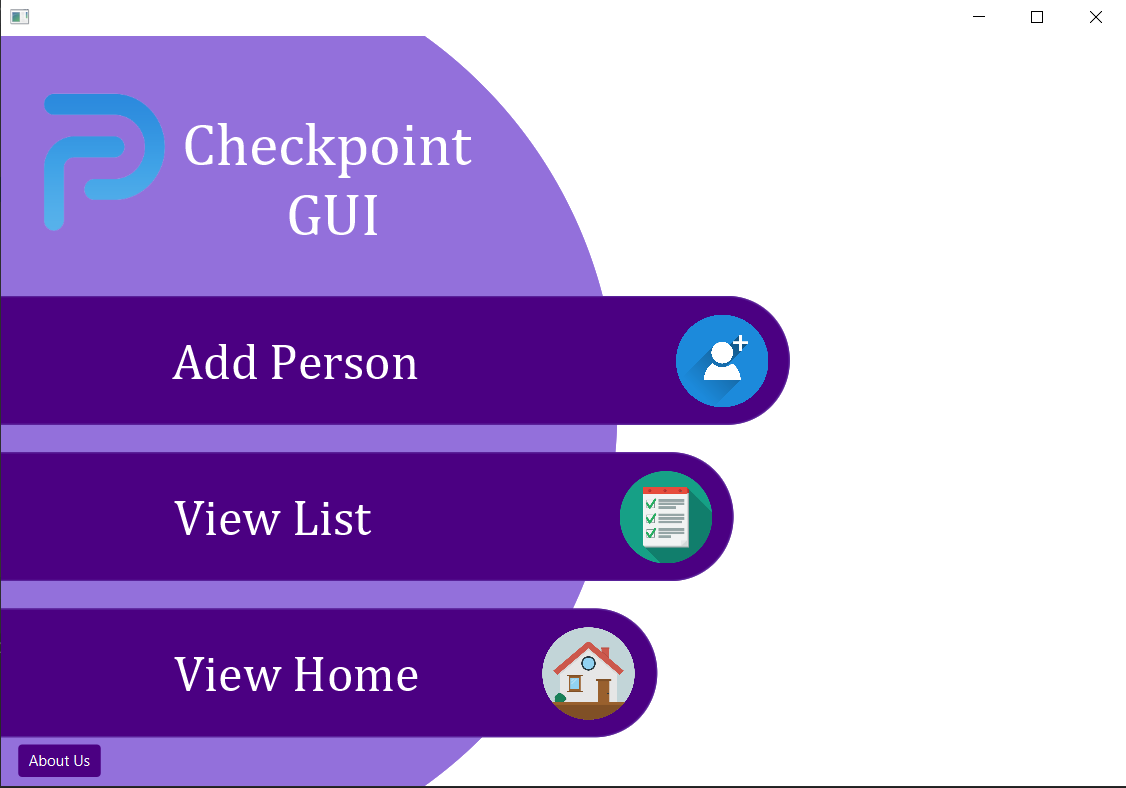
*Figure 1.11 House Profile*



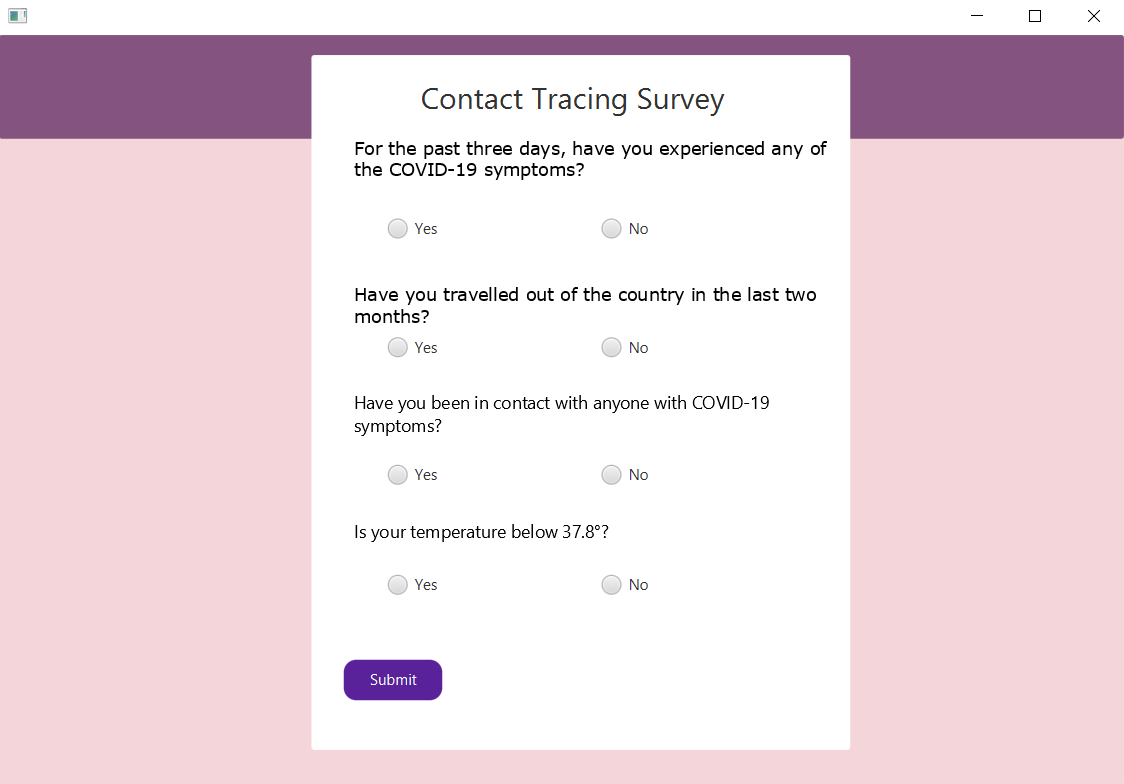
*Figure 1.12 About Controller*



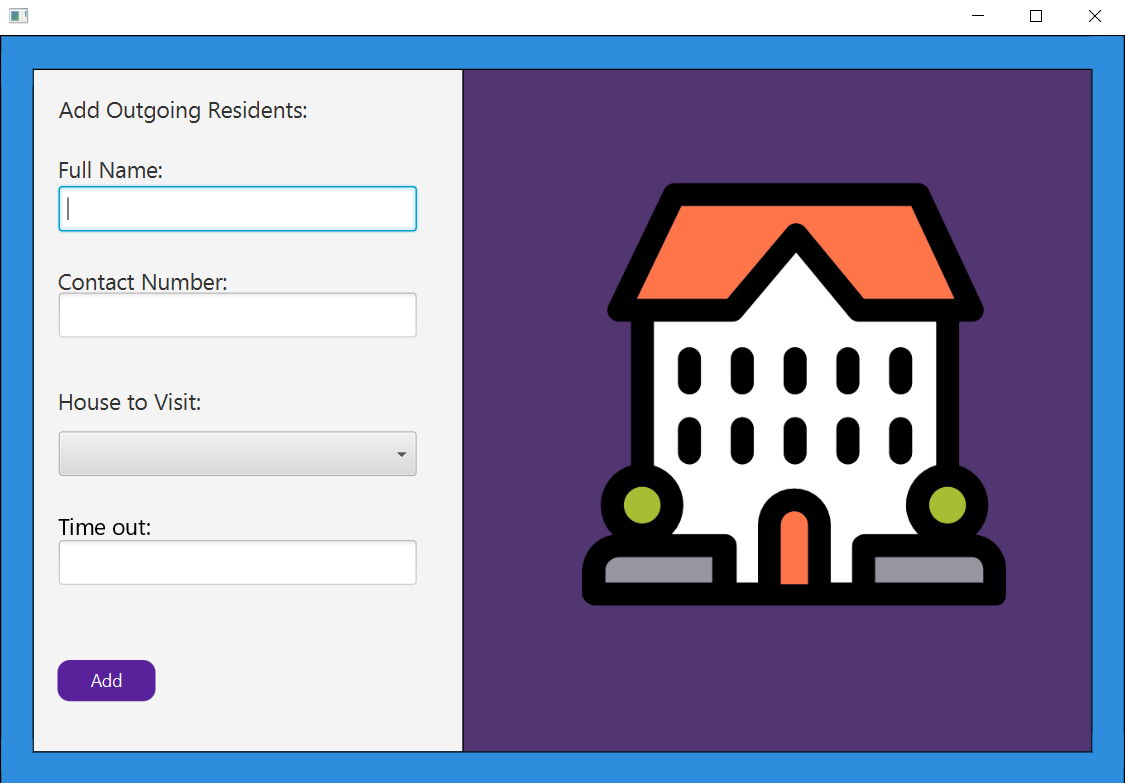
*Figure 1.13 Output Login System*



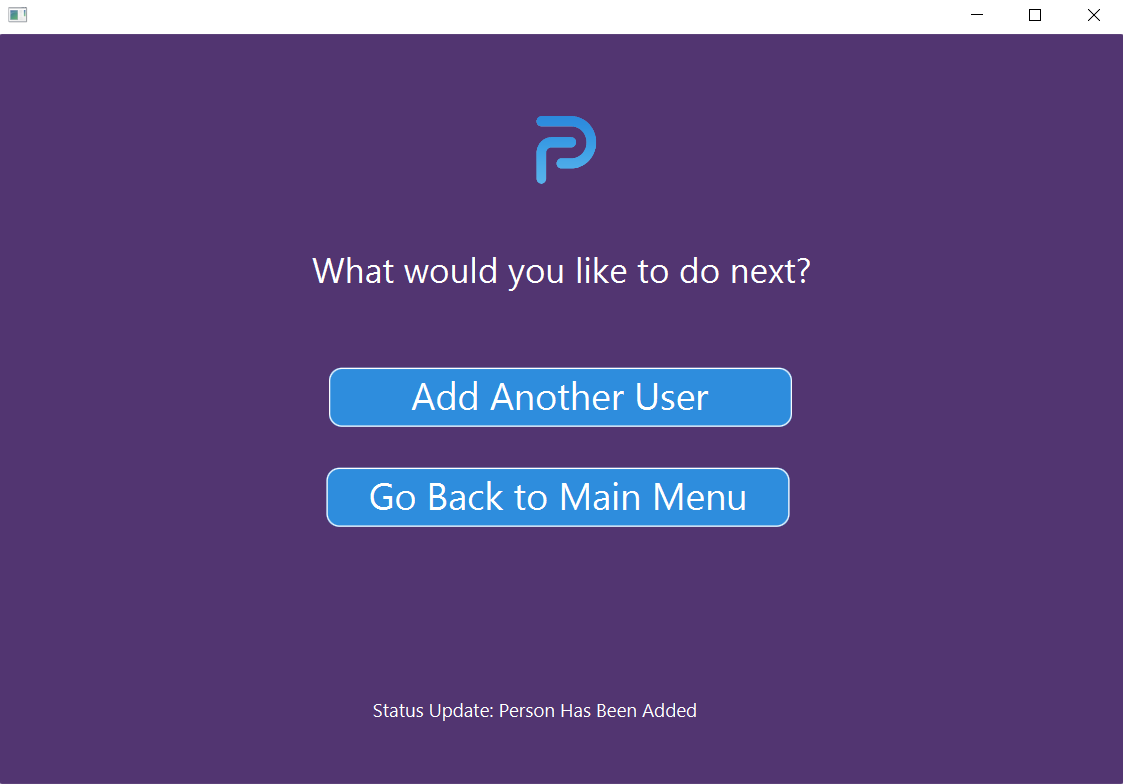
*Figure 1.14 Output Main Menu*



*Figure 1.15 Survey/Add Person Menu*



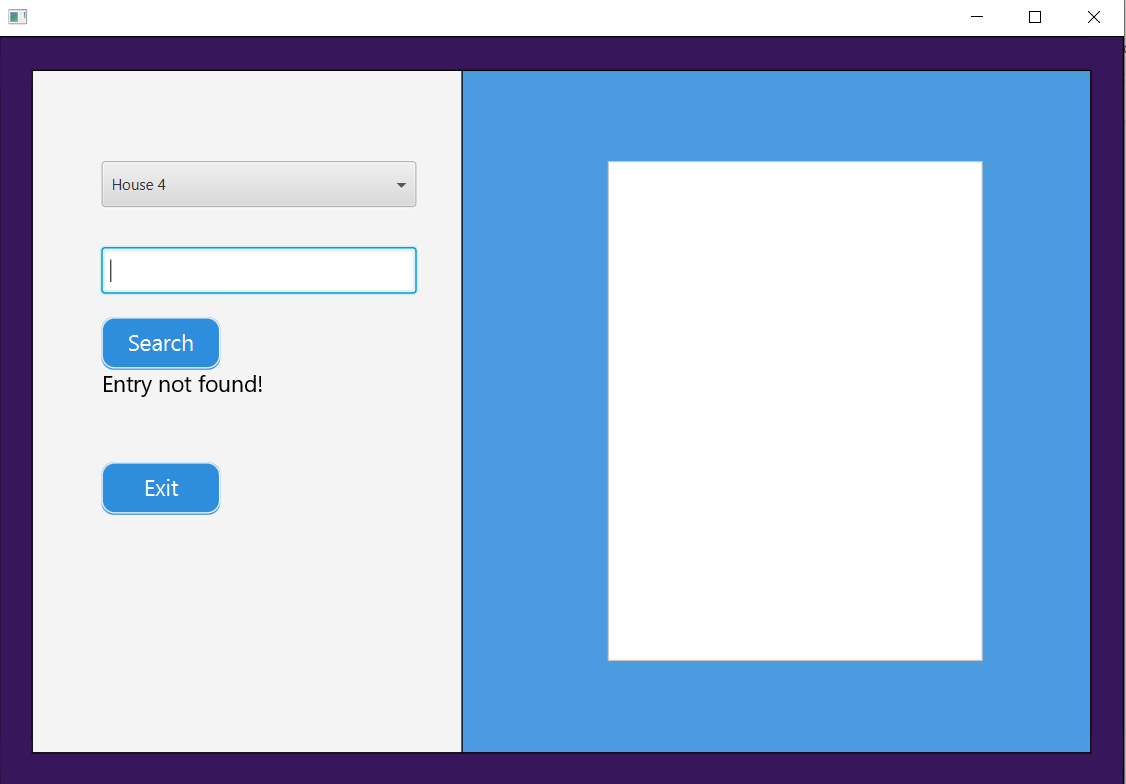
*Figure 1.16 Add Screen Menu*



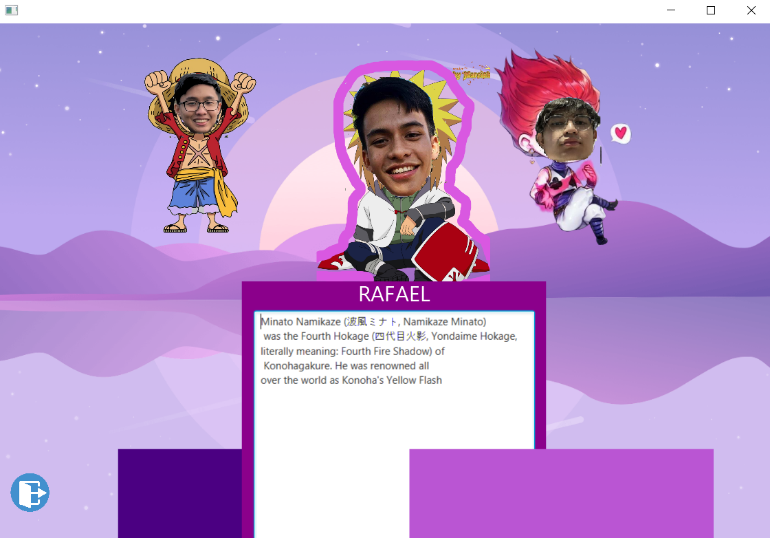
*Figure 1.17 Final Controller Screen*



*Figure 1.18 List Screen*



*Figure 1.19 House View Screen*



*Figure 1.20 About Us Screen*

The images above show the output of the program (*1.13 – 1.20*), with the first image being the login screen. Following the login screen is the main menu screen which transitions after logging in. The main menu screen shows the following user options respectively: Adding a person; Viewing the list of people; and viewing the houses that contains the people inputted. The add person menu leads the user to answer a survey which contains questions regarding their medical status relating to the current pandemic (COVID-19). After this menu proceeds another input screen which allows the user to input their information such as their name, contact number, address, and the time of leave. Furthermore, this leads to the last screen of the feature which gives the user to choose to add another person or go back to the main menu. The second option is called the view list option which lets the user view people and the information they have inputted. The house view screen shows the houses (address of the people) they belong to and shows the people who live in them. Moreover, a search feature is implemented in this menu to search for the exact names of the people that were inputted in the previous menu. Lastly, the last image (*1.20*) is an additional or bonus feature that shows the information of the people who worked and created the program.

# Implementation of Data Structures

## Arrays



*Figure 2.1 screenLogin.java*

Aside from multiple other arrays throughout the program to handle the inputs, the login system uses String arrays to hold the usernames and passwords that are taken from the text files.

## Linked List

*Figure 2.2 mainMenuController.java*

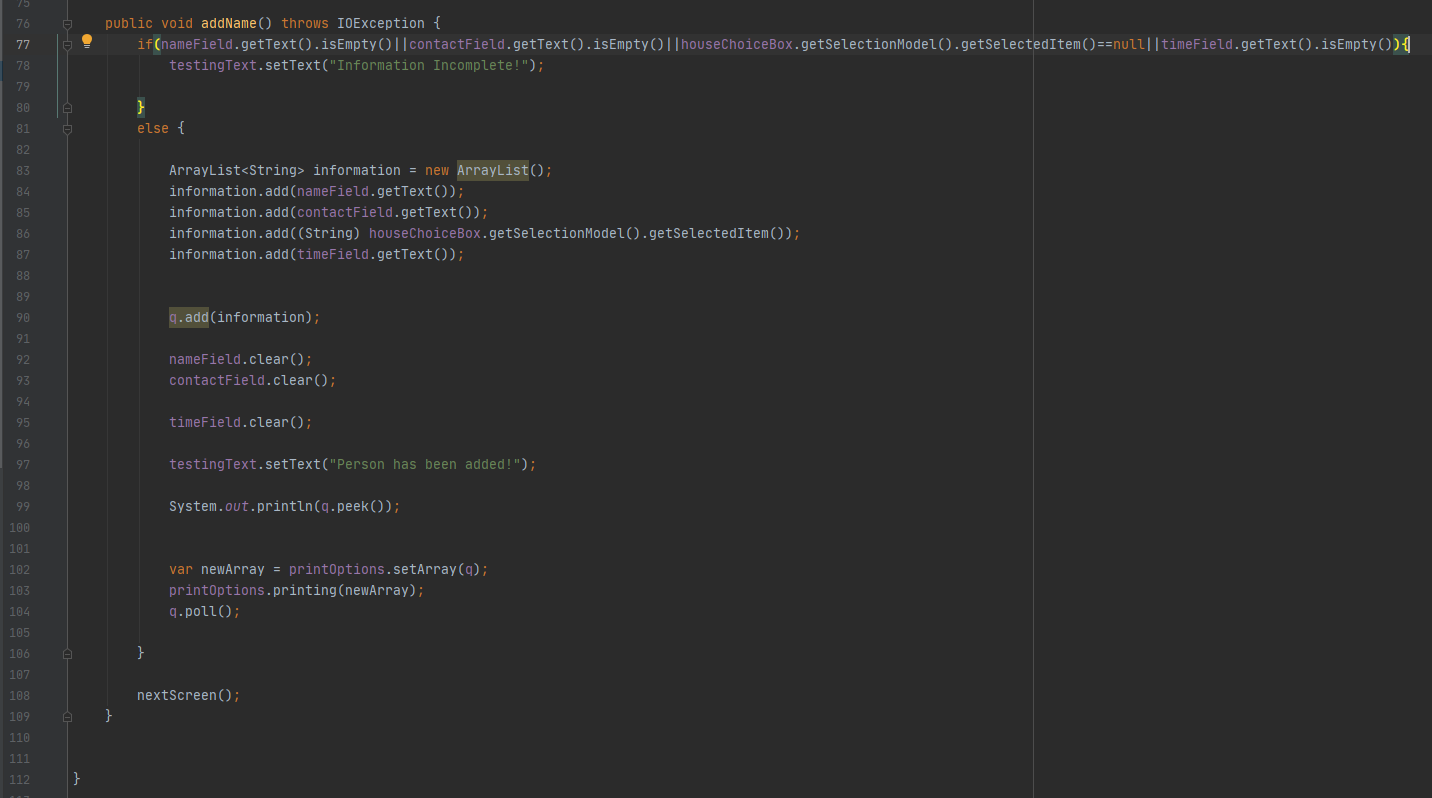
In the mainMenuController, the initialize function fills a LinkedList with FXML loaders. This is used to handle changing scenes and fxmls. The other FXMLs have a setLinkList function that takes the linkedlist initialized in the mainMenuController and transfers it to the called FXML so it may be used again to change screens and FXMLs.

## Queue



*Figure 2.3 printingOptions.java*

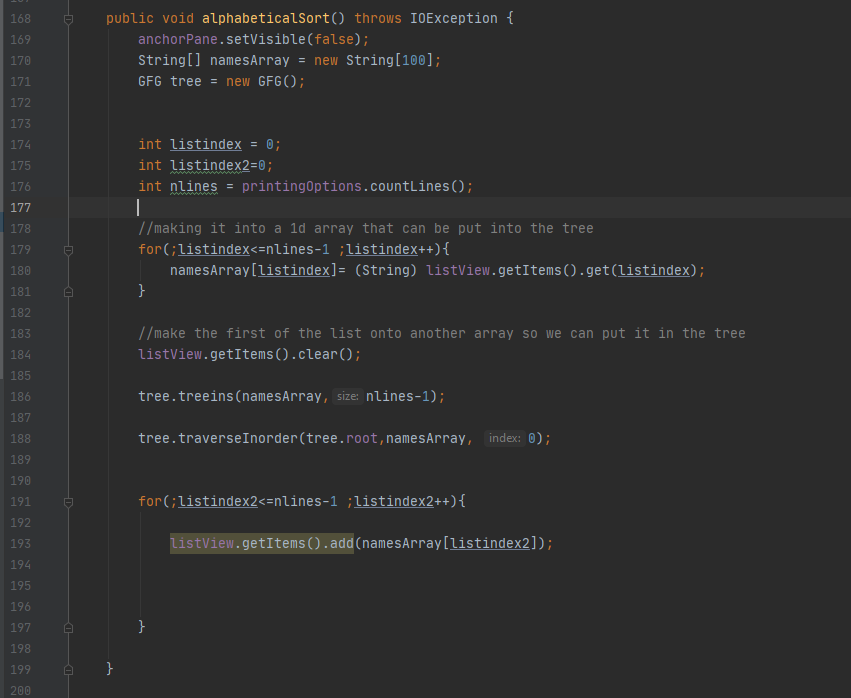
In printingOptions, the functions printing and printingChoices use queues. It takes the ArrayList given when the function is called and transfers it to a queue then for each item in the queue gets printed onto their respective text file.



*Figure 2.4 addMenuController.java*

In the addMenuController, the addname function takes the inputs from the textfields in the FXML and put it into a queue so it may be properly arranged.

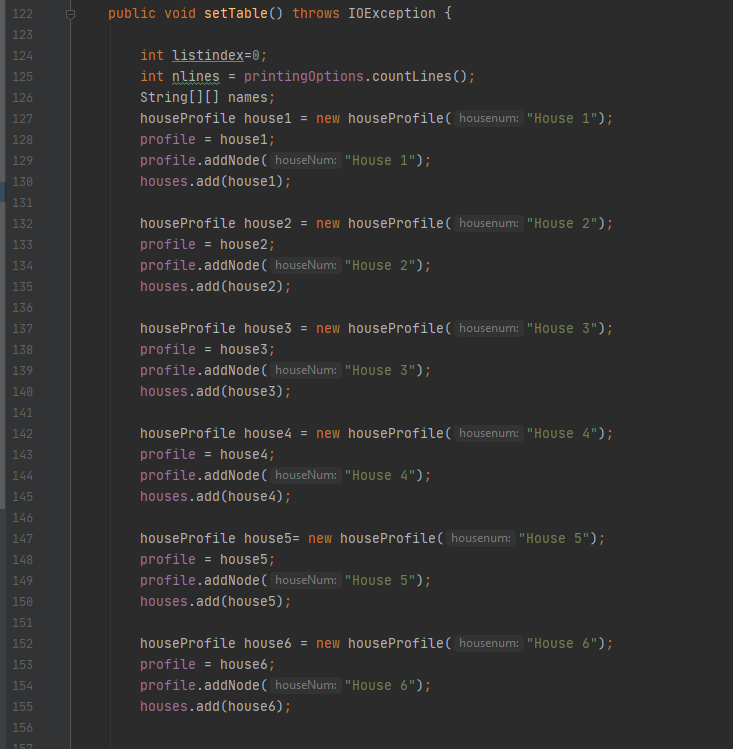
## Tree



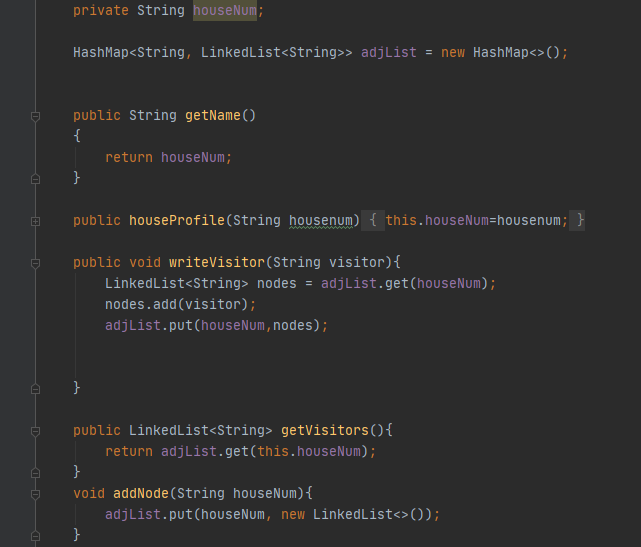
*Figure 2.5 Listcontroller.java*

In listcontroller, the alphabeticalSort function is used to sort the list alphabetically, the way this is done is by inputting the list onto a tree which arranges it. The tree structure that was used for this class is in Tree.java which is a basic tree structure that has the functionalities as well as a traverseInorder function.

## Graph (Hashmap and Linked List)



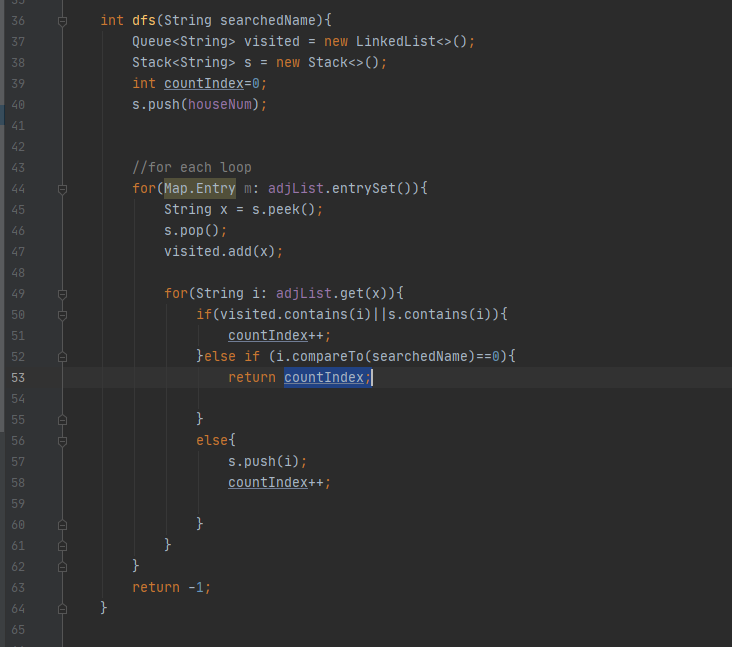
*Figure 2.6 houseController.java*



*Figure 2.7 Houseprofile.java*

The houseprofile uses a graph that uses a hashmap and linkedlist to handle the relationships between the visitors and the houses they visit. The houseprofiles get created for each house and for each visitor so that they can be added onto each other’s linked list for the adjacency.

## Stack



*Figure 2.8 DFS search in houseprofile.java*

In houseprofile there is a function to perform Depth first search which utilizes a queue and a string to visit each node. This depth first search is used in the housecontroller.java in order to search for a give name in the search tab and find which house they are in.

# Discussion of data and results

The process of creating the backbone to front end of the application was successful. The program was able to fully implement all the data structures required, and these would be arrays, stack, queue, linked list, tree, and graph (hashmap). The implementation of the data structures made the handling and storing of data efficient and flexible. The inputting, deletion, sorting, and organization of data was made possible as well.

In conclusion, it may be lacking in some features, which can be the facilitating and simplification of the collection of data from external devices. Lastly, as a standalone project, with the purpose to assist individuals that need an application that records data for current states of individuals in a community, it has met to its desired objective. With this, it has enabled the programmers to apply everything needed to be able to create a successful program that benefits the community.

# Conclusion and Future Work

The implementation of the Checkpoint Management system (CMS) serves a great purpose for the community especially in this time period which is currently experiencing the pandemic. The program provides a wide reach in terms of its audience since it can cater to not just people infected by COVID-19 but also those people who would like to maintain the consistent cataloging of information about the current states of various individuals inside of a small community. As a result, multiple establishments are able to use this program to full effect since they also need to exercise proper safety and management of barangays, hotels, subdivisions, and compounds.

The researchers recommend that the program can be implemented through a mobile application or even a database integration. Furthermore, additional features can be added such as the changing of dates, the addition of dates, the editing of history, a registration button, levels of account priority, or even basic accounts for the household members. Another recommendation for future researchers would be to implement a separate application form for visitors and residents answering the survey which would in turn be sent to the main program that records all the data. This concludes the section of conclusion and future work recommendations.

# Contributions

Rafael M. Cabrera:

* Worked on the list, tree, house, house profile, and about controller.
* Created the design for the about screen.
* Debugged multiple revisions of the program.
* Recorded the project presentation.
* Worked on the implementation of the FXML files.
* Compilation and implementation of the program.
* Worked on the paper.

Daine Janssen R. Jadman:

* Created the FXML designs for the house, add, login, final, and survey screen.
* Worked on the login controller, screen login, and the printing options.
* Finalized the topic and features.
* Worked on the paper.

Kalvin Kenjo H. Yap:

* Created the designs for main menu and the list screen.
* Worked on the main menu, add menu, survey, and final controller.
* Worked on the making of the FXML files.
* Performed multiple revisions of the paper.

# References

|  |  |
| --- | --- |
| [1] | Who coronavirus disease (covid-19) dashboard. (n.d.). Retrieved February, 2021, from https://covid19.who.int/?gclid=EAIaIQobChMI5bmBmsrS7gIVwxx9Ch0sOQHdEAAYASAAEgJV3PD\_BwE |
| [2] | Semilla, N. (2020, June 25). COVID-19 lockdown: A checkpoint for every village in Cebu City. Retrieved February 2021, from https://newsinfo.inquirer.net/1296903/covid-19-lockdown-a-checkpoint-for-every-village-in-cebu-city |
| [3] | Border checkpoints, police patrols to continue even under mgcq. (2020, July 10). Retrieved February, 2021, from https://cebudailynews.inquirer.net/325724/border-checkpoints-police-patrols-to-continue-even-under-mgcq |
| [4] | Javafx. (n.d.). Retrieved February, 2021, from https://openjfx.io/ |
| [5] | Svinterikou, M.; Kanaroglou, P. (2006, September 3). A Microsimulation Approach to the Modelling of Urban Population and Housing Markets Within an Object-Oriented Framework [PDF]. Ontario: Econstor. |
| [6] | Us7206835b2 - user operation log recording and storing apparatus using object-oriented technology. (n.d.). Retrieved February, 2021, from https://patents.google.com/patent/US7206835B2/en |
| [7] | Us7290007b2 - method and apparatus for recording and managing data object relationship data. (n.d.). Retrieved February, 2021, from https://patents.google.com/patent/US7290007B2/en |
| [8] | Covid-19 effects on the workload and mental health of Iranian healthcare workers. (2020, May 19). Retrieved February, 2021, from https://www.researchsquare.com/article/rs-27866/v1 |
| [9] | Sharma, D. (2017, April). Developing applications using JavaFX. Retrieved February 2021, from https://www.researchgate.net/publication/316960258\_Developing\_applications\_using\_JavaFX. |