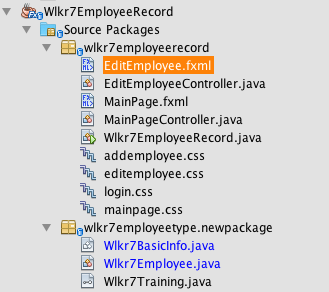
Documentation for Wlkr7EmployeeRecord

This JavaFX Application is made by Wei Xian Low (PawPrint: WLKR7) for the CS3330 Final Project.

This application has utilized multiple classes, which include, an abstract class, a subclass, and an interface.



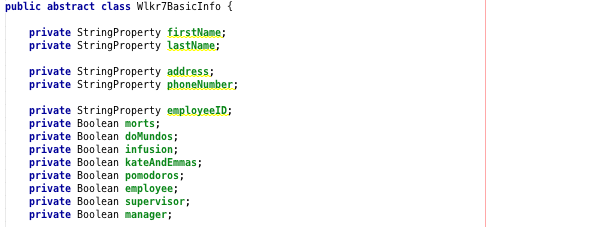
Based on the snapshot on the left, the abstract class, subclass, and the interface is located inside the package called: “wlkr7employeetype.newpackage”

The java file named “Wlkr7BasicInfo.java” is the abstract class.

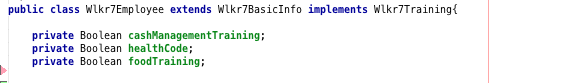
The java file named “Wlkr7Employee.java” is the subclass implementing the abstract class “Wlkr7BasicInfo.java”

The java file named “Wlkr7Training.java” is the interface class where it’s implemented by the file “Wlkr7Employee.java”

**-Object Oriented Elements**



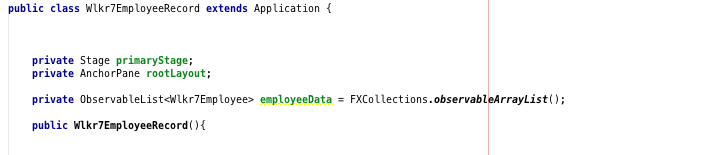
The above picture shows a snapshot of codes contained in the file “Wlkr7BasicInfo.java”. This shows that the class is an abstract class.



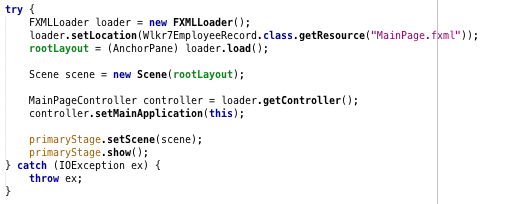
The above picture shows a snapshot of codes contained in the file “Wlkr7Employee.java”. This shows that this class is a subclass of “Wlkr7BasicInfo” and it implements the interface file “Wlkr7Training”

**-Utilizing Code Elements:**

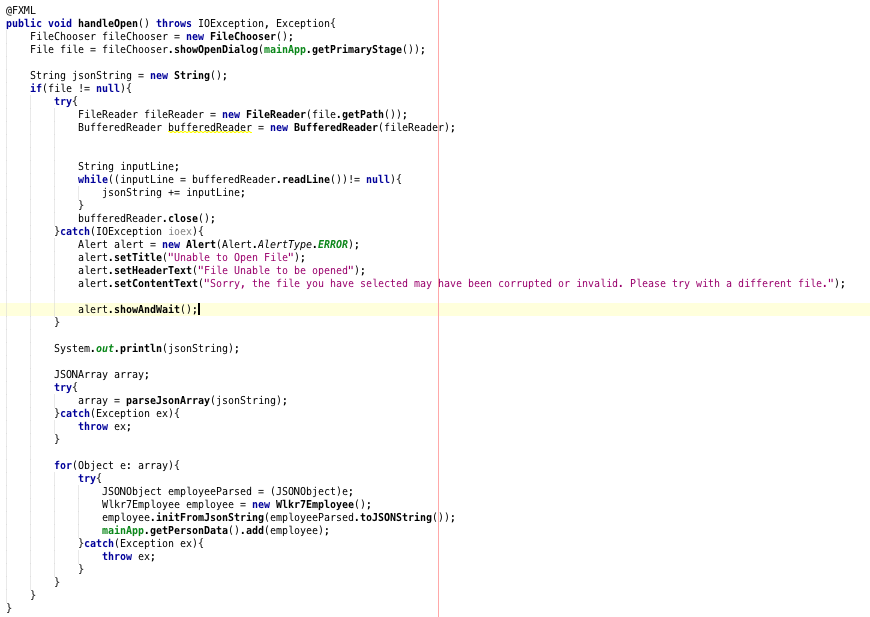
Collection Classes:



The above picture is a snapshot of codes contained in the file “Wlkr7EmployeeRecord.java”. This shows that there is an ArrayList utilized here to serve as a collection class handling an ArrayList of java object which is Wlkr7Employee.

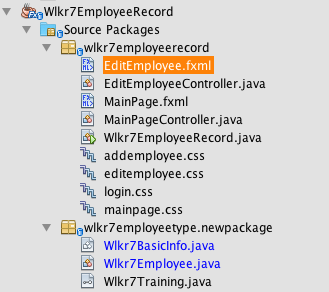


The above picture is a snapshot of codes contained in the file “Wlkr7EmployeeRecord.java”. This is an example of one of many Exception handling in this program. A try-catch is used to catch an exception and a different set of code is used to handle the captured exception.



The above picture is a snapshot of codes contained in the file “MainPageController.java”. This is another example of Exception handling in this program. Multiple try-catch is used to ensure an exception is properly handled based on the codes.

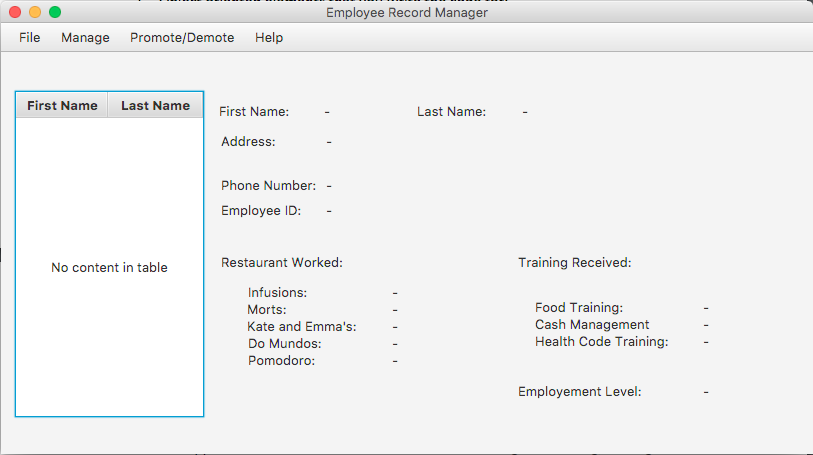
**-Model-View-Controller (MVC)**

Based on the picture on the left, there’s a clear look on how the MVC is implemented here in this program. This program uses JavaFX to develop the MVC.

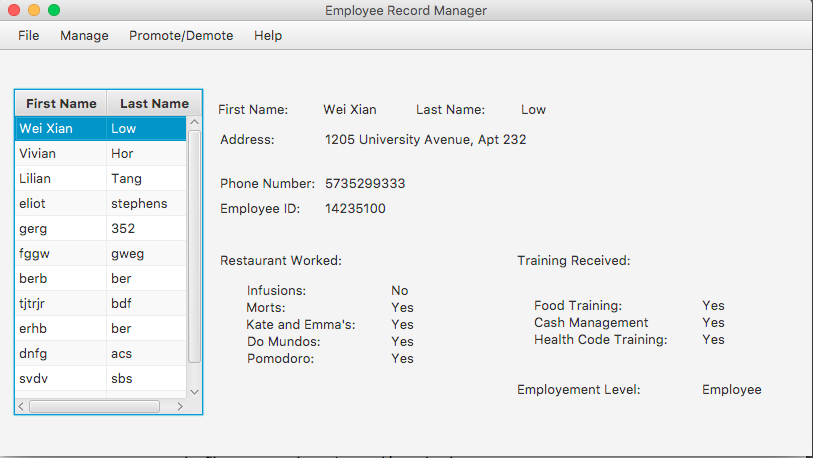
There are three scenes where user could interact with as shown by the .fxml files.

There are also controller files that controls and manipulate data based on the user’s interaction with the scene.

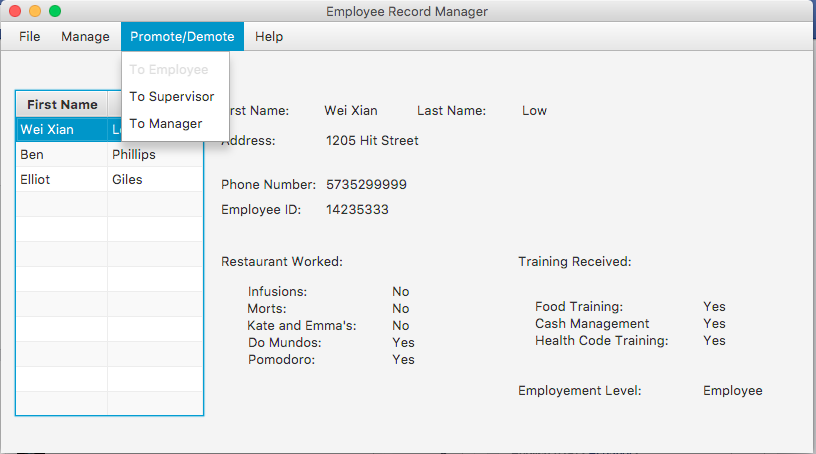
**-User Interfaces/Scenes**

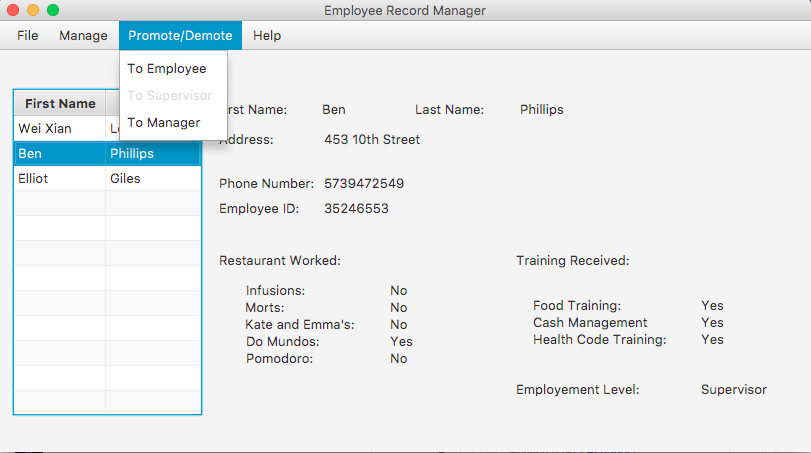


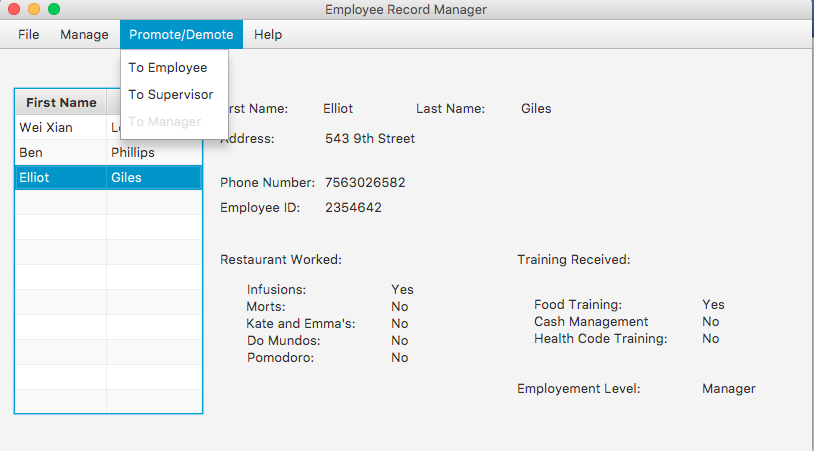
The above picture is a snapshot taken from the program. This scene serves as a main screen where user will first see when the program has complete its initialization.



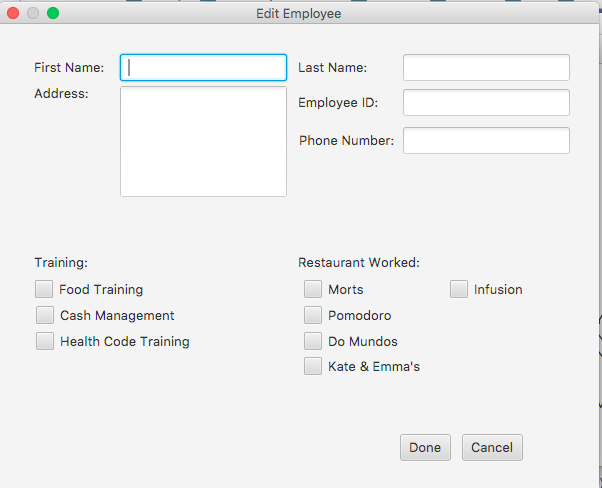
The above picture is a snapshot taken from the program. This scene shows how the program will look like when it’s populated with data and information provided by the user.



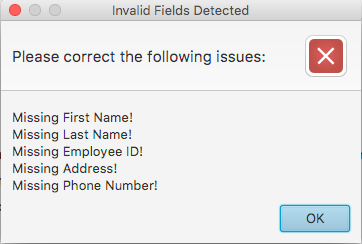




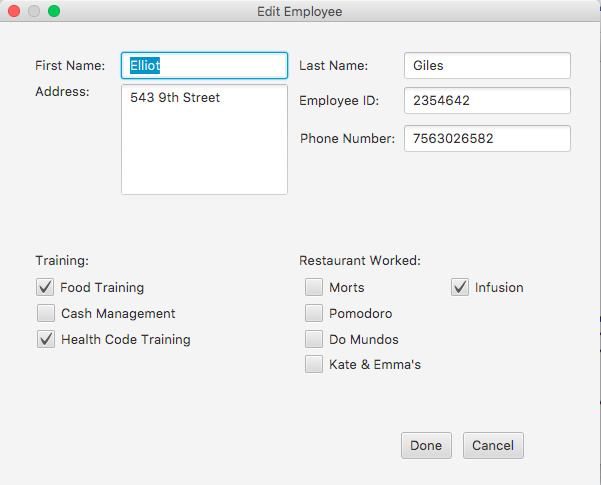
The above snapshots are taken from the application showing the availability of a menu item depending on the state of the data selected from the TableView on the left of the program. If the employee has an employee status, the employee menu item will be disable to remind the user that the selected employee is an employee. This works too when the employee is a supervisor, and a manager.



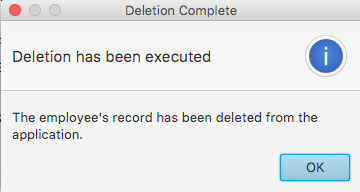
The above snapshot is taken from the application, showing the ability to have a pop-up window when the user clicks on the add employee menu item on the menubar. An empty form will be produced for the user to interact with.



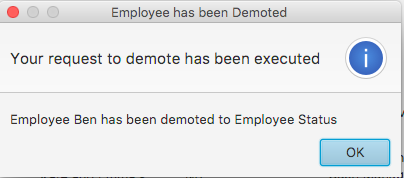
The above snapshot is taken from the application, if the user were to click done on the previous picture without filling out any information required, an alert will be prompted to the user to warn them of their actions.



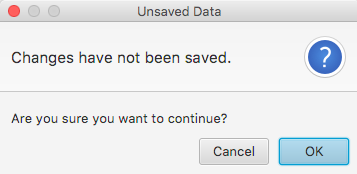
The above snapshot is taken from the application. This snapshot shows that when an employee is selected from the TableView, where the user chooses to edit the employee information, a pop-up window will appear showing the form where it has already been populated with the selected employee’s information.



The above snapshot is taken from the application. This snapshot shows an alert prompt that is given to the user if the user chooses to delete the selected employee’s record.

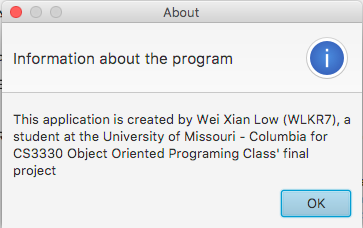


The above snapshot is taken from the application. This snapshot shows an alert prompt that is given to the user if the user chooses to promote or demote an employee’s status. A different message appears depending on the state of the promotion or demotion. In the snapshot above, the employee is demoted to a basic employee status.



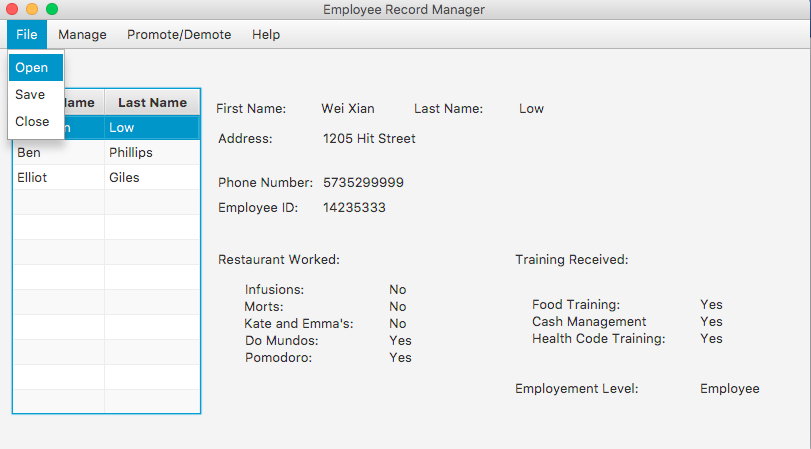
The above snapshot is taken from the application. This snapshot shows an alert prompt to the user if the user clicks the closes menu item from the menu bar or directly closes the window. An alert will be given warning user that the data is not saved after modification of employee’s record.

**-About information**

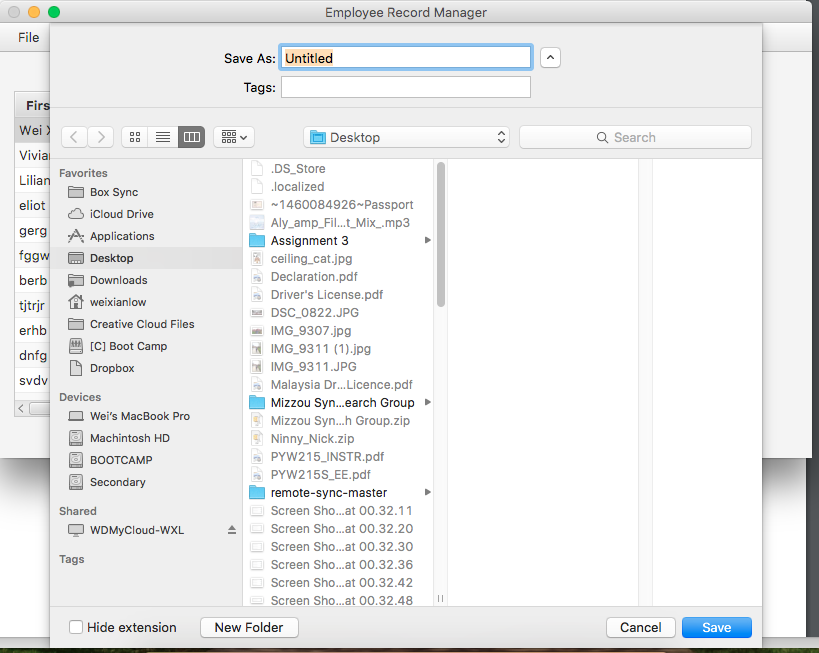


The above snapshot is taken from the application. This snapshot shows that the about information could be reached by selecting the about menu item in the menu bar.

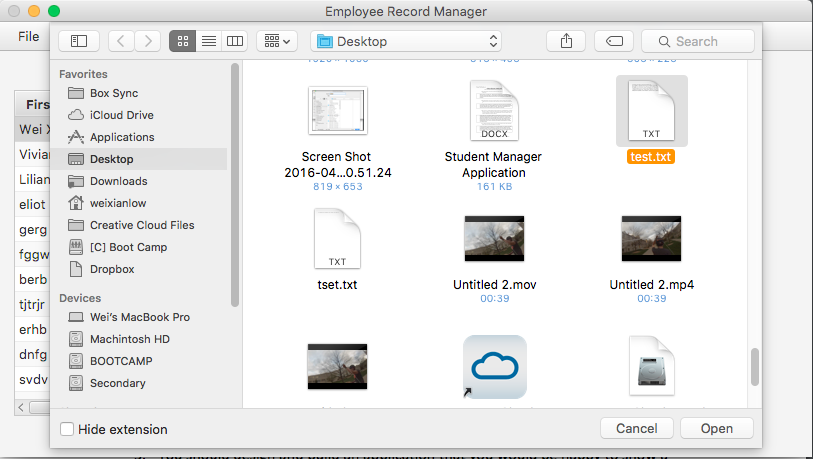
**-Save and Open data**



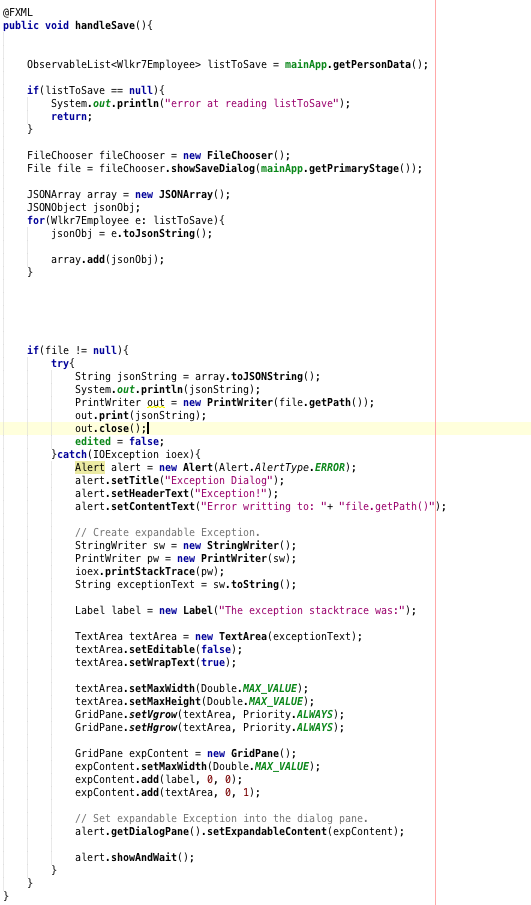
The above snapshot is taken from the application. This snapshot shows that users are able to save and open their data that is created in the application. The Open and Save menu item is available at the menubar.



The above snapshot is taken from the application. This snapshot shows that a save dialog is shown to the user in determining where to save the data.



The above snapshot is taken from the application. This snapshot shows that an open dialog is showed to the user in determining which file to open.



The above snapshot of code is taken from the file “MainPageController.java”. This portion of the code shows the code responsible in handling saving user’s data. The code saves the employee’s record in an JSONObject format, in which later is handled and saved in an JSONArray object. The final save file will include the JSON string where it’s readable by user and allows open source of save file to be open in other application.



The snapshot of code above is taken from the file “MainPageController.java”. This portion of the code handles the open operation when asked by user. This code will open a file previously saved by the user. The code will then parse the JSON related code and execute them inside the program.