**Criterion B: Design**

Table of Contents:

**Design Overview2**

*Language*2

*Data Inputs from the User*2

*User Interfaces Design*3

*Classes Design*7

**System Flowchart10**

**Test Plan11**

**Design Overview**

**Language:**

The language chosen to develop this software is Java programming language with JavaFX library, and FXML language is used to create GUI.

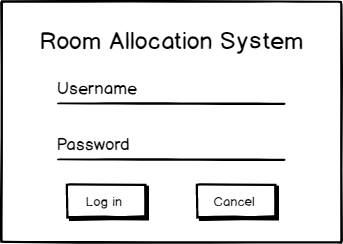
**Data Inputs from the User:**

The following data are required to be inputted from the user before making room allocations.

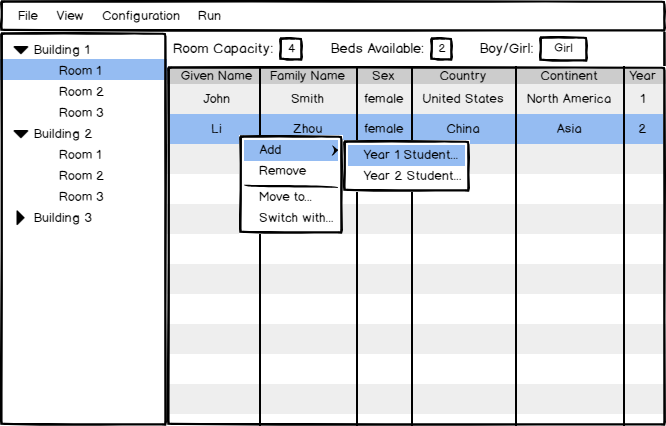
|  |  |  |
| --- | --- | --- |
| **Data Field** | **Description** | **Data Type** |
| Username | The name set by the client for login | String |
| Password | Password set by the client for login | String |
| Room No./Names | The numbers or names of all rooms and the buildings number or names they belong to | String |
| Maximum Capacity of all rooms | The maximum number of people who can live in each room | int |
| Type of all rooms | Whether it’s a boy’s room or girl’s room | String |
| Students’ names | Family names and given names of all students | String |
| Students’ sex | Whether a student is male or female | String |
| Students’ nationalities | Countries which all students are from | String |

**User Interfaces Design:**

The following screen layouts are designed with **Balsamiq Cloud[[1]](#footnote-1)**.

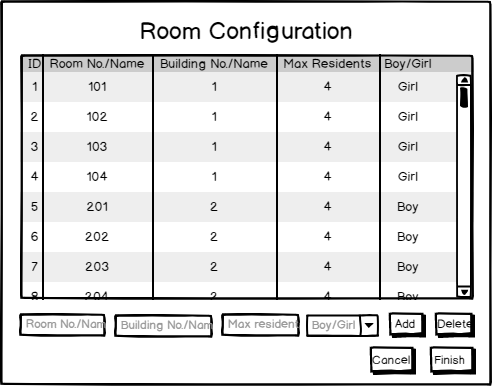
Figure 1. Login Window

If the username and password are entered correctly, main window will show after pressing log in.

Figure 2. Main Window

By changing the selected room, all contents will be automatically updated. The user can manually manipulate students in this window

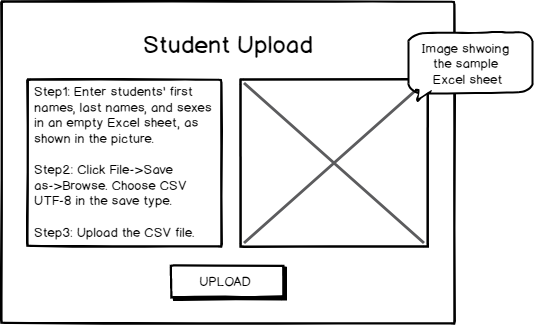
Figure 3. Room Configuration Window

****

Here the client can configurate the rooms as her wishes.

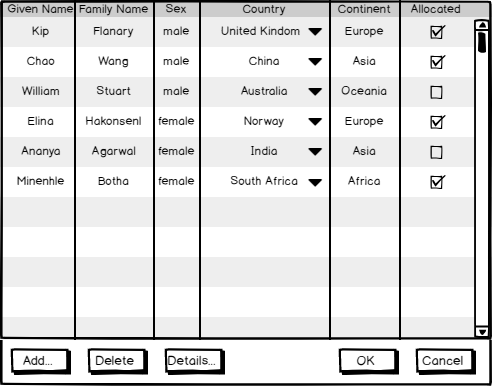
(IDs are autoincrement)

Figure 4. Student Upload Window

****

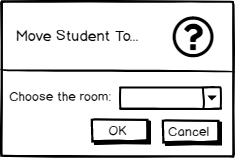
This window is for client to upload the information of all students

Figure 5. Student Configuration

****

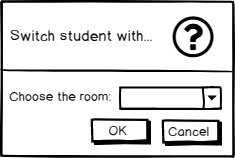
Here the client can modify the student info, and check whether students are allocated, and where is the student allocated.

Figure 6. Move Student Window

****

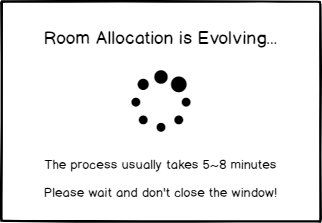
By clicking the “Move To…” in the Main Window, this window will pop up and all room that correspond with the sex of the chosen student will be added into the ComboBox.

Figure 7. Switch Student Window

****

By clicking the “Switch with…” in the Main Window, this window will pop up and all room that correspond with the sex of the chosen student will be added into the ComboBox.

Figure 8. Running Genetic Algorithm Animation

****

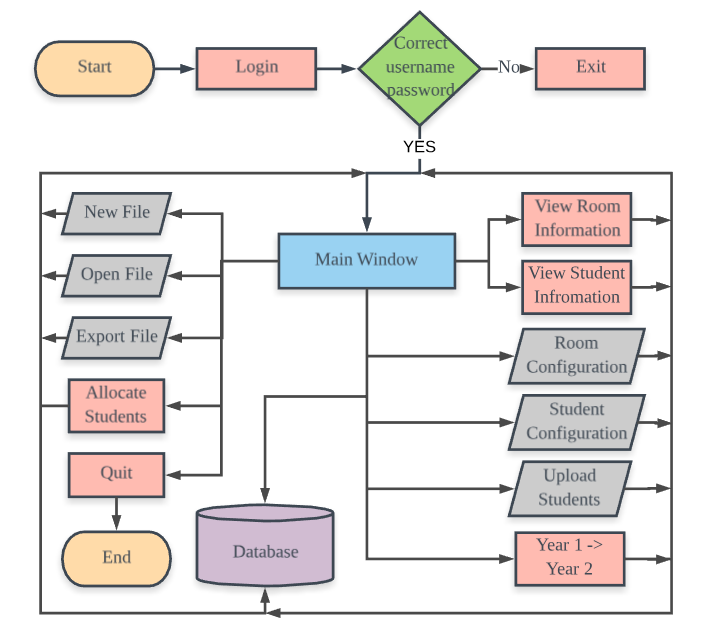
This window will automatically pop up while genetic algorithm is running in the background thread to allocate room. Once the allocation is done, the spinner will change to a tick, telling the client that the allocation is done.

**Classes Design:**

The software is designed with JavaFX. Therefore, there are a lot of different controller classes, each handling the functionality within one Stage (Window), and there are also some functional classes to provide other functionalities.

|  |  |  |
| --- | --- | --- |
| **Package Name** | **Class Name** | **Functionalities** |
| / | Main | * The entry of the program * Connect the software to Database * Read the last opened file |
| GA | DNA | * Represent an allocation (the list of all Rooms that contain students) * Calculate the fitness of this allocation * Mutation (make changes in the current allocation) |
| GA | Population | * Contains 1000 DNAs * Select next generation of 1000 DNAs based on their fitness * Apply mutation with specific mutation rate to each selected DNA * Evaluate whether the evolution should stop and output the result |
| functional | AutoComplete  ComboBox | * An imported class, to make selcteCountry ComboBox autocomplete |
| functional | HandleButton | * Handle all the “next”, “cancel” buttons |
| functional | Room | * Container of properties of a room |
| functional | Student | * Container of properties of a student (Country is of ComboBox type for the purpose of TableView) |
| functional | StudentString | * Container of properties of a student (country property is of String type) |
| controllers.login | LoginController | * Show the login screen and check if the username and password are correct. * If correct, show the main window. |
| controllers.main | MainController | * Show the main window * Controls all the actions that happen through it |
| controllers.main | AddYear1Student  Controller | * Show the list of unallocated first-year students that matches with the sex of the chosen room. * Add the chosen student into the chosen room |
| controllers.main | AddYear2Student  Controller | * Same with above except the students are second-year students. |
| controllers.main | SwitchStudents  Controller | * Switch the chosen student in another room with the chosen student in this room. |
| controllers.main | RunningGA  Controller | * Show the process indicator while the genetic algorithm is allocating rooms in the background thread. |
| controllers.main | ShowUnallocated  StudentsController | * Show the list of unallocated students before starting allocation. * Ask the user for confirmation to start allocation |
| controllers.newFile | Directory  Controller | * Show the window for entering new file name and choosing its directory |
| controllers.newFile | RoomConfig  Controller | * Show the window for configuring student rooms |
| controllers.newFile | StudentConfig  Controller | * Show the window for uploading student CSV files including their name and sex |
| controllers.newFile | StudentConfig2  Controller | * Show the window for choosing students nationalities in ComboBoxes. * Map the students’ nationalities to the continents they are from |
| controllers.view | RoomController | * Show the number of rooms and beds for boys and girls. |
| controllers.view | StudentController | * Show the number of boys and girls and totoal number of students |
| controllers.  configuration | AddOrDelete  RoomController | * Show the window for adding or deleting student rooms |
| controllers.  configuration | UpdateRoomInfo  Controller | * Modify room No./Names, building No./Names, Max Residents, and Boy/Girl. |
| controllers.  configuration | Year1Student  Controller | * Add/Delete Year 1 Students * Modify names, sexes, countries, and continents of Year 1 students. |
| controllers.  configuration | Year1StudentAdd  ClickedController | * Create a pop-up window for adding new Year 1 student |
| controllers.  configuration | Year2Student  Controller | * Add/Delete Year 2 Students * Modify names, sexes, countries, and continents of Year 2   students |
| controllers.  configuration | Year2StudentAdd  ClickedController | * Create a pop-up window for adding new Year 2 student |
| controllers.  configuration | UploadYear1  StudentController | * Override the existing Year 1 student by uploading new students file. |
| controllers.  configuration | UploadYear2  StudentController | * Override the existing Year 2 student by uploading new students file. |

**System Flowchart**

****

**Test Plan**

Before releasing the software to the client, the software should go through alpha testing first to detect and identify all possible bugs. Therefore, I have made the following test plan to ensure the functionality of the software.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test No.** | **Test Name** | **Test Purpose** | **Testing Method** | **Expect Outcome** |
| 1 | Login Test | To ensure the login system is working correctly | 1. Enter the wrong username and password. 2. Enter the correct username and password. | 1. Does not log in 2. Logs in |
| 2 | Create New File Test | To ensure the client can create new file | Go through the process of creating a new file. | No error message is displayed; all data are updated in the database |
| 3 | Open File Test | To ensure that the client can open another file | Go through the process of opening a file | No error message is displayed; new data are being updated from the new database file |
| 4 | Export Test | To ensure that the client can export data to Excel sheet | Click “File” -> “Export to Excel File” in the main window | The created Excel sheet should have the exact same allocation as displayed in the software |
| 5 | Room Config  Test | To ensure that rooms can be configured at any time | Try complex configurations combination at the same time. | All changes should be updated properly without throwing error messages |
| 6 | Student Config Test | To ensure that students can be configured at any time | Try complex configurations combination at the same time. | All changes should be updated properly without throwing error messages |
| 7 | Context Menu Test | To ensure that “Add”, “Delete”, “Move to”, and “Switch with” functionalities in the context menu are working properly | Try each function multiple times with students from different rooms. | All changes should be updated properly without throwing error messages |
| 8 | Allocation Test | To ensure that the Genetic Algorithm is working to optimize for the diversity in all rooms | Run the allocation for 10 times. After each time, check all the rooms for their diversity. | In most of the rooms all students are from different countries and no more than 2 students are from Asia.  In each room no more than 2 students are from the same country. |

1. https://balsamiq.com/ [↑](#footnote-ref-1)