**ASSIGNMENT 1 FRONT SHEET**

|  |  |  |  |
| --- | --- | --- | --- |
| **Qualification** | **BTEC Level 5 HND Diploma in Computing** | | |
| **Unit number and title** |  | | |
| **Submission date** |  | **Date Received 1st submission** |  |
| **Re-submission Date** |  | **Date Received 2nd submission** |  |
| **Student Name** |  | **Student ID** |  |
| **Class** |  | **Assessor name** |  |
| **Student declaration**  I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice. | | | |
|  |  | **Student’s signature** |  |

**Grading grid**

|  |
| --- |
| Grade (0-10) |
|  |

|  |  |  |
| --- | --- | --- |
| **❒ Summative Feedback: ❒ Resubmission Feedback:** | | |
| **Grade:** | **Assessor Signature:** | **Date:** |
| **IV Signature:** | | |

Table of Contents

[I. Introduction 4](#_Toc106928223)

[II. Requirements 4](#_Toc106928224)

[III. UI design 5](#_Toc106928225)

[1. Login 5](#_Toc106928226)

[2. Registering 6](#_Toc106928227)

[3. Student list 7](#_Toc106928228)

[IV. Implementation 9](#_Toc106928229)

[V. Test 9](#_Toc106928230)

[VI. Result 9](#_Toc106928231)

[VII. Conclusion 9](#_Toc106928232)

# Introduction

# Requirements

# UI design

## Login

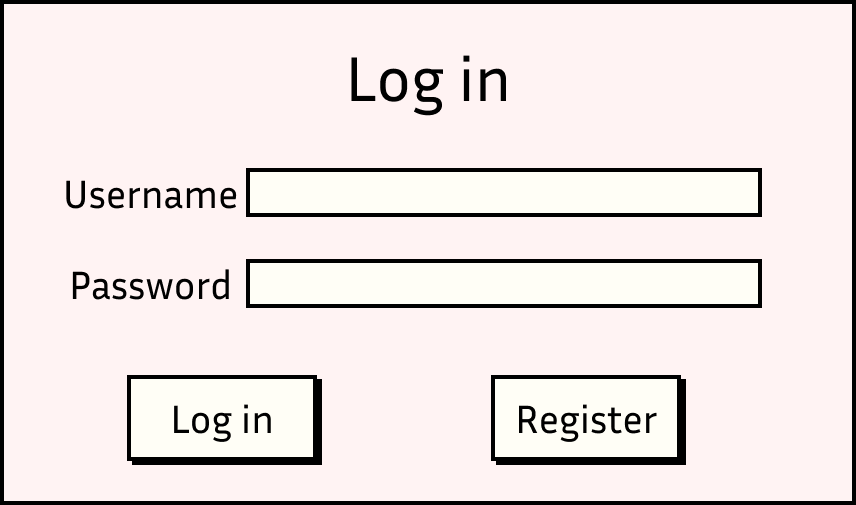


Figure 1 Login screen

## Registering

Graphical user interface, application

Description automatically generated with medium confidence

Figure 2 Register window

## Student list

Graphical user interface

Description automatically generated

Figure 3 Index window

## Add student

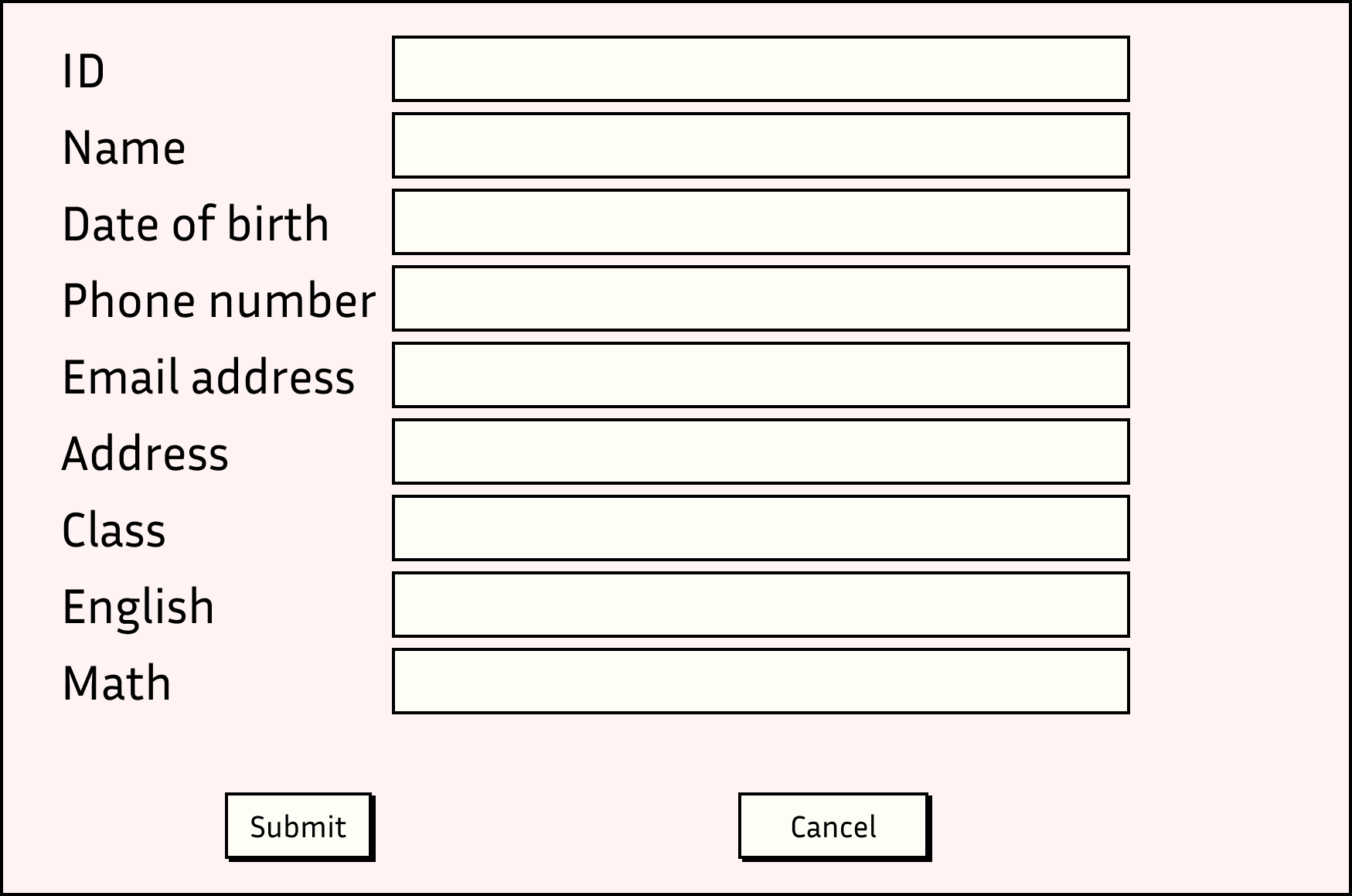


Figure 4 Window to add new student

## Edit student

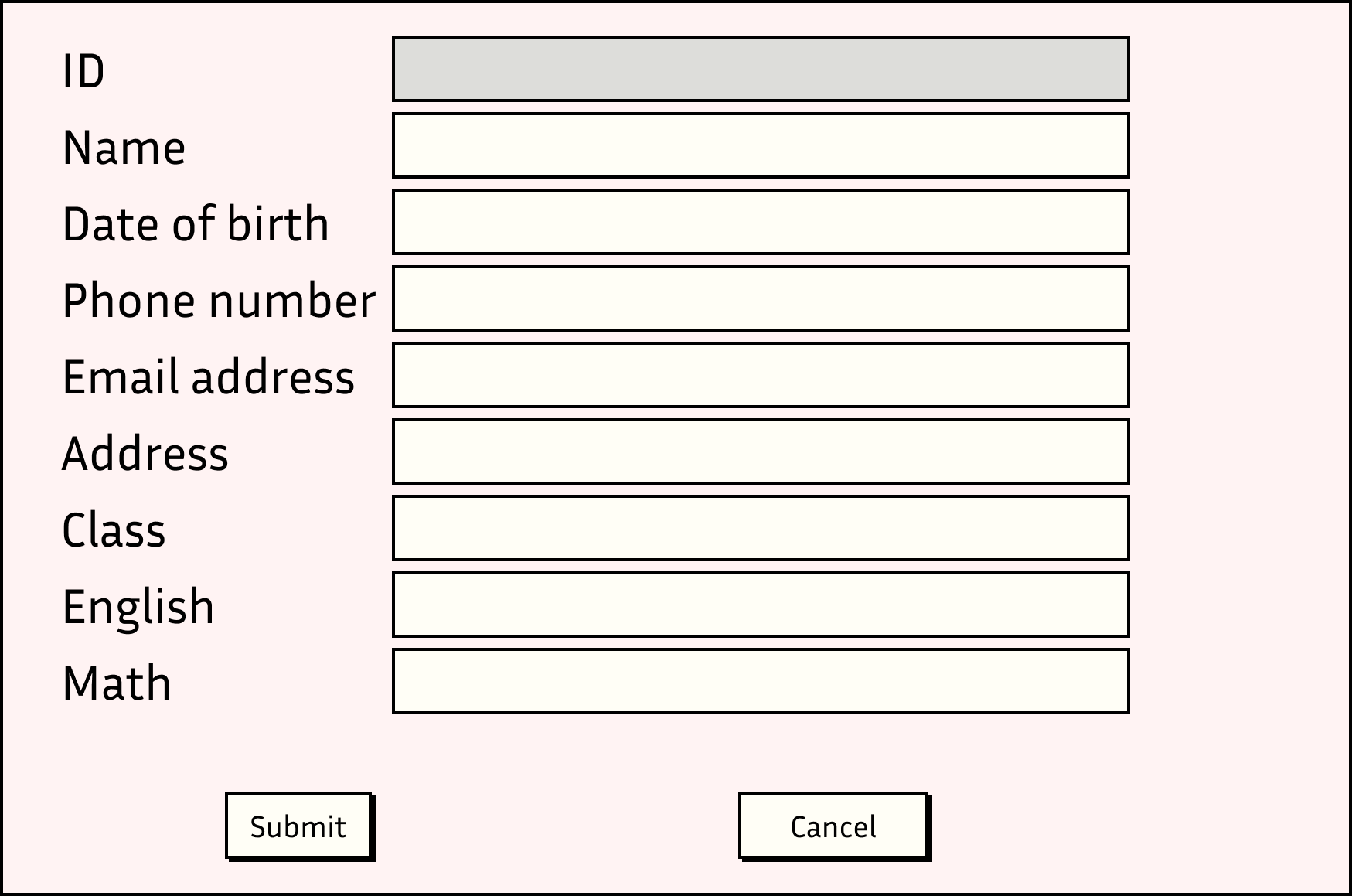


Figure 5 Edit a student's information

## View student information

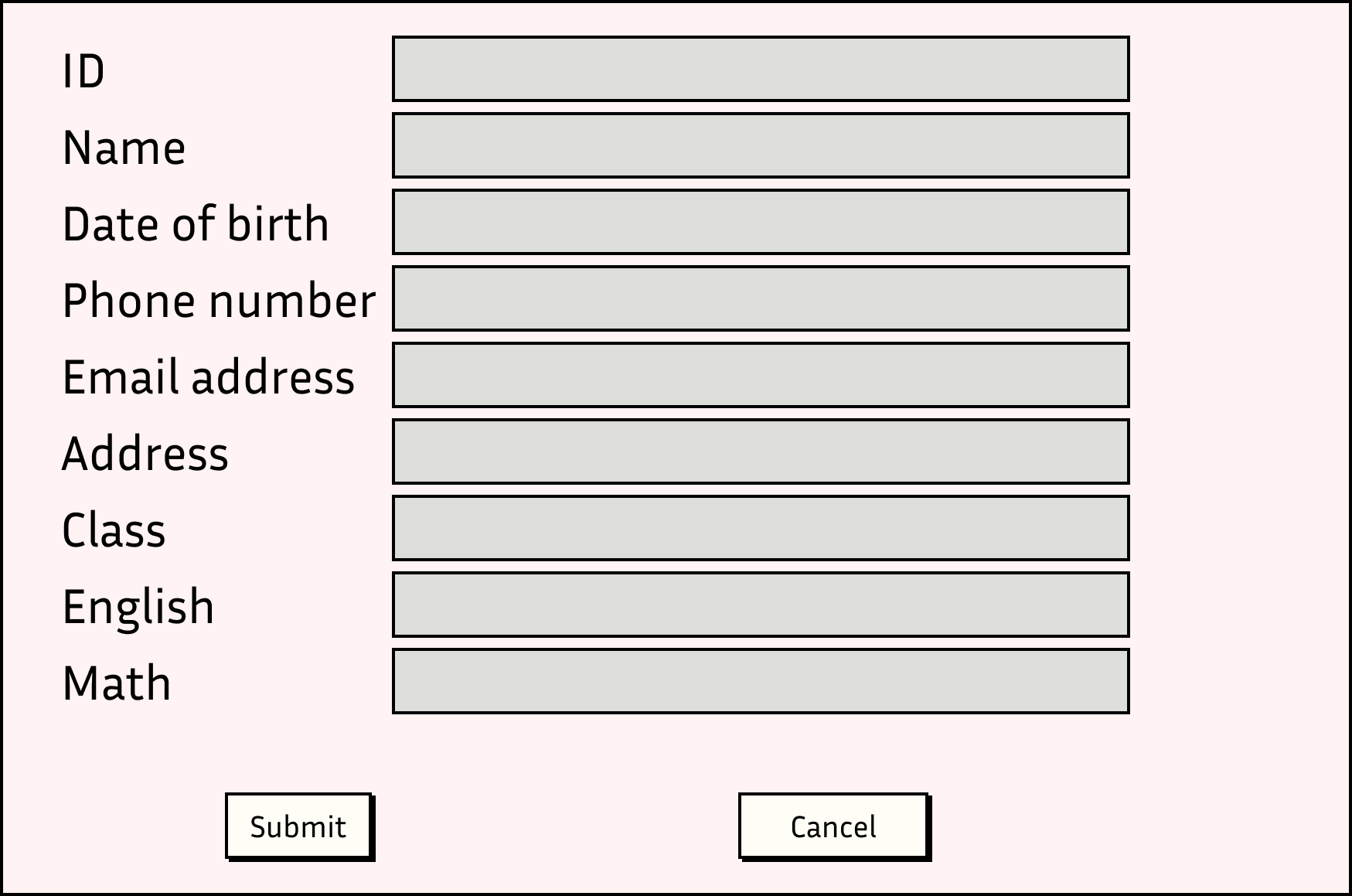


Figure 6 View a student's information

# Implementation

## Program structure

This program uses the MVC model which divides the code into three different components called Model, View and Controller. Each components handles a specific side of the program. (Tutorialspoint, 2022)

In this program, the Model component is used to store the structure of every object data type. The Controller the uses it to create instances of the objects and then store them in a file.

The View component is responsible for the application's UI logic. It presents the user with forms, fields, and table for them to interact with the program intuitively.

## Classes

### PersonModel

StudentModel holds the components of necessary to create a person object and modify it. Having a Person model is optimal because should the program expand and implement the feature to manage teachers’ information, the development would take much less time and efforts

|  |  |
| --- | --- |
| Variable | Type |
| ID | string |
| Name | string |
| Email | string |
| Phone number | string |
| Address | string |
| Date of birth | string |

Table 1 Variables of PersonModel

|  |  |
| --- | --- |
| Method | Description |
| Constructor | To create a new person with all the variables included within the model |
| Empty constructor | To create a new person with no information and then use setters to set each information |
| Getters | Gets the information of the instance and returns it |
| Setters | Sets the information of the instance |

Table 2 Methods of PersonModel

### StudentModel

StudentModel holds the components of necessary to create a student object and modify it.

|  |  |
| --- | --- |
| Variable | Type |
| mathGrade | int |
| englishGrade | int |
| gpa | double |
| classID | String |

Table 3 Variables of StudentModel

|  |  |
| --- | --- |
| Method | Description |
| Constructor | To create a new person with all the variables included within the model |
| Empty constructor | To create a new person with no information and then use setters to set each information |
| Getters | Gets the information of the instance and returns it |
| Setters | Sets the information of the instance |
| calculateGPA | This function is to calculate the GPA of the student by dividing the sum of Math and English grade of the student |
| getStudentInfo | Print the student's information out in JSON format |
| getStudentInfo2 | Print the student's information out in CSV format |

Table 4 Methods of StudentModel

### UserModel

UserModel is a model used to hold the information of every user’s login information including username and password.

|  |  |
| --- | --- |
| Variable | Type |
| username | string |
| password | string |

Table 5 Variables of UserModel

|  |  |
| --- | --- |
| Method | Description |
| Constructor | To create a new person with all the variables included within the model |
| Empty constructor | To create a new person with no information and then use setters to set each information |
| Getters | Gets the information of the instance and returns it |
| Setters | Sets the information of the instance |
| getUserInfo | Print the user information out in JSON format |
| getUserInfo2 | Print the user information out in CSV format |

Table 6 Methods of UserModel

### LoginController

This Controller is used to control events regarding logins. When a login is performed, the program reads the file containing a list of users, parses them into an ArrayList of User objects before trying to find a match with the login input by the user. After the login is performed, a message displaying its status will pop up and lead the user to the appropriate window. When the user click on the “Register” button, they are taken to the register window.

### RegisterController

This Controller is used to control events regarding registration. When a registration is performed, the program reads the file containing a list of users, parses them into an ArrayList of User objects before trying to find a match with the registration input by the user. If there is a match with the username, an error message is displayed and asks the user to enter a different username. If there is no matching username, a success message is displayed, and the user is taken to the login window.

### StudentController

This Controller is used to control events regarding student information. It is responsible for writing student information onto a file. It is also responsible for reading student information from a file and put them into an ArrayList of Student objects.

## important algorithms

## error handling

# Test

# Result

# Conclusion

# References

Tutorialspoint, 2022. *MVC Framework - Introduction.* [Online]   
Available at: https://www.tutorialspoint.com/mvc\_framework/mvc\_framework\_introduction.htm  
[Accessed 24 June 2022].