Documentation

PasoEats Group 2025: Brian, Connor, Reese, Armando

Git Book Documentation

This document has been implemented in gitbook by Reese. This alternative has more comprehensive class documentation, including examples of code. The link to the gitbook:

https://pasoeats.gitbook.io/pasoeats

Work Division

Reese: Main Code / Git Book Documentation

Brian: Group Manager / UI Code / Documentation

Connor: Documentation / Design / GUI Code

Armando: Edits to Main Class

Future Development

Future development would center around the need for the following:

- **Refinement of Component Responsibilities** - The UserInterface and Manager classes should have better division of behavior. The Manager should focus on data manipulation but have no display functions. This would not include any debug code that might be used for development to display values along the way – but this code should also be removed before release. This would leave the UserInferface class to display information based on a request sent to the manager.
- **Graphical User Interface Development** – Connor has started our next iteration of the user interface. The next iteration includes the development hopes to improve accessibility for a wider range of users. The initial design aims to replicate the functionality of the existing command-line interface. Subsequent releases will likely prioritize further enhancements and updates to the graphical user interface. The GUI is still in beta and unfinished, and has components not integrated yet. In future development this would be remedied.

- **Persistent Data Storage** – Implementing persistent data storage is a fundamental requirement for the application to effectively manage academic information. We are considering the following approaches:
    - **JSON Database File** – One potential solution involves updating the application's shutdown process to export all current data into a JSON file. Upon startup, the application would then read and load this data. Or this could also be access by each mutator and accessor method to edit the file as it happens which would limit the amount of exporting at shutdown time.
    - **Database Integration** – An alternative approach involves integrating a database system (e.g., mySQL) to manage persistent data. This would require modifying existing functions, to interact directly with the database for data retrieval and modification. This service would also need to be run with or before our application has started.
- **Application Configuration** – As the application evolves, a dedicated configuration file will be essential for managing various settings related to data import/export and internal application behavior.
- **Software Testing** - A critical aspect of future development will be the implementation of comprehensive software testing procedures to ensure the reliability and stability of the application. This will encompass Unit, Integration, and System level testing.

OOP Explanation

Polymorphism: Student and Instructor both have inherited methods of the same name, but are overridden and behave differently.

Encapsulation: Frequent use of setters and getters that interact with private variables and public methods to hide the variables from outside sources.

Inheritance: Student and Instructor classes both inherit the Person class and it’s members.

Abstraction: There is complexity abstraction from the user with the console user interface and the graphic user interface. The class Person is also abstract.

Data Structures Justification

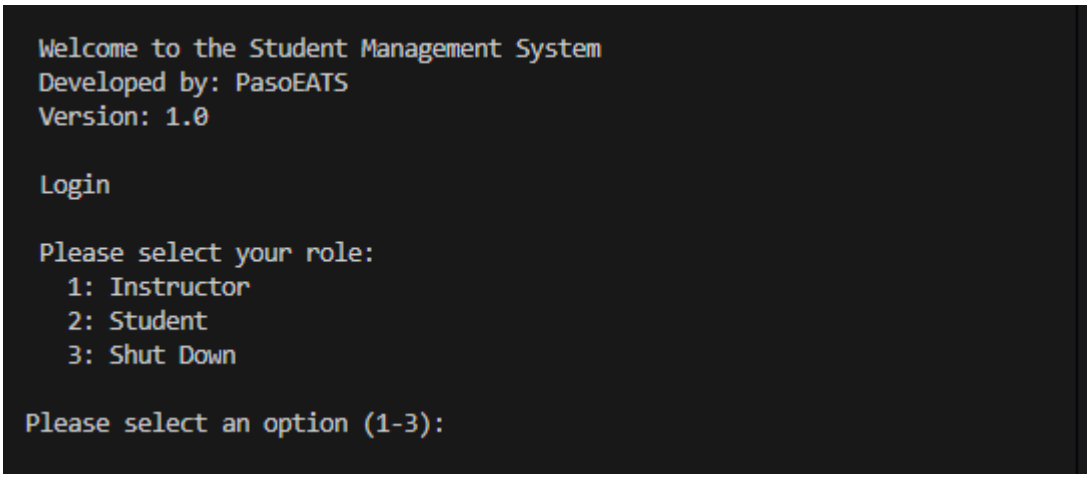
ArrayLists are used to handle all of the objects in the project. This is because ArrayLists are easy to edit and can change size dynamically, allowing for as many or few objects as needed. This

also has the benefit of always being able to add new objects, such as new students, classes, or instructors.

Output Examples

Console User Interface

Start Screen:



Student Path Example:

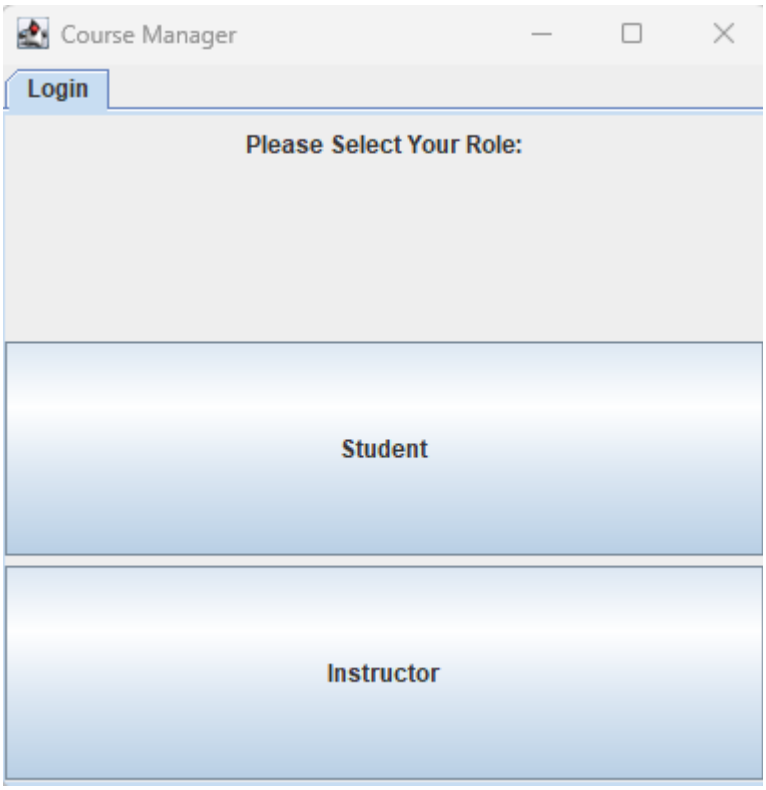
```
 Login

 Please select your role:
   1: Instructor
   2: Student
   3: Shut Down

Please select an option (1-3): 2

 Student Login

Please enter your Student ID: 4

Welcome, Alice!

 Student Menu

   1: View Grade Summary
   2: Modify My Name
   3: Modify My Email
   4: Export Data
   5: Log Out

Please select an option (1-5): 1

Grade Summary for Alice

Assignments:
Assignment 1: 70/100
Assignment 2: 100/100
Assignment 3: 31/100
Assignment 4: 26/100
Assignment 5: 93/100

Overall:
Total Score: 320 / 500
Percentage: 64.00%
Letter Grade: D

Press Enter to continue...
```

Instructor Path Example (inc. invalid input):

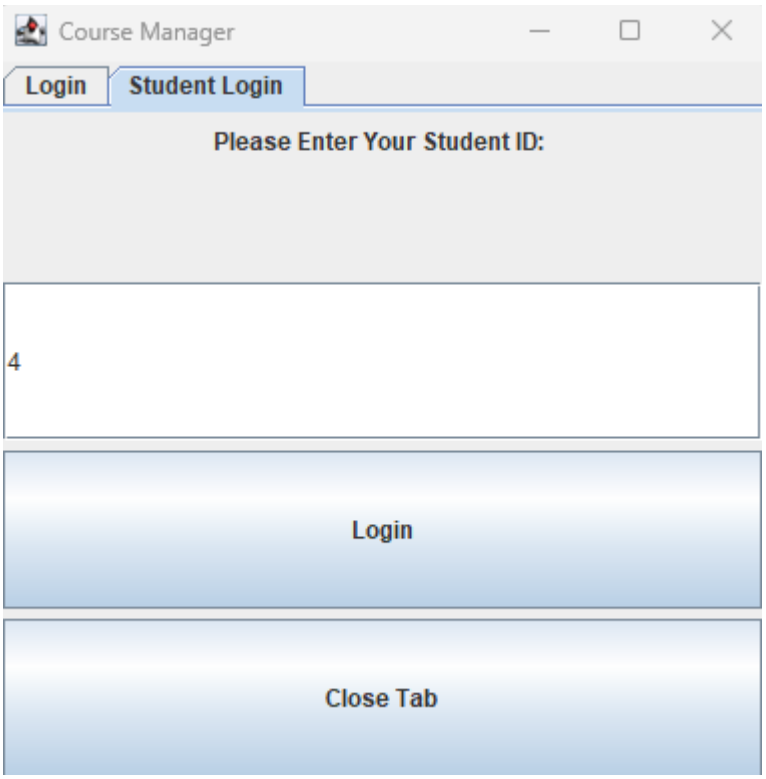
```
 Login

 Please select your role:
   1: Instructor
   2: Student
   3: Shut Down

Please select an option (1-3): 1


 Instructor Menu

   1: Manage Students
   2: Manage Assignments
   3: Manage Instructors
   4: Manage Courses
   5: Export Data
   6: Log Out

Please select an option (1-6): 1

 Manage Students Menu

   1: Add New Student
   2: Display All Students
   3: Back to Main Menu

Please select an option (1-3): 1
Enter Student ID: 4
Student ID 4 already exists.

Press Enter to continue...

 Manage Students Menu

   1: Add New Student
   2: Display All Students
   3: Back to Main Menu

Please select an option (1-3): 1
Enter Student ID: 76
Enter Student Name: Cody
Enter Student Email: cody@example.com

Cody has enrolled!

Press Enter to continue...
```

BETA: Graphic User Interface (GUI)

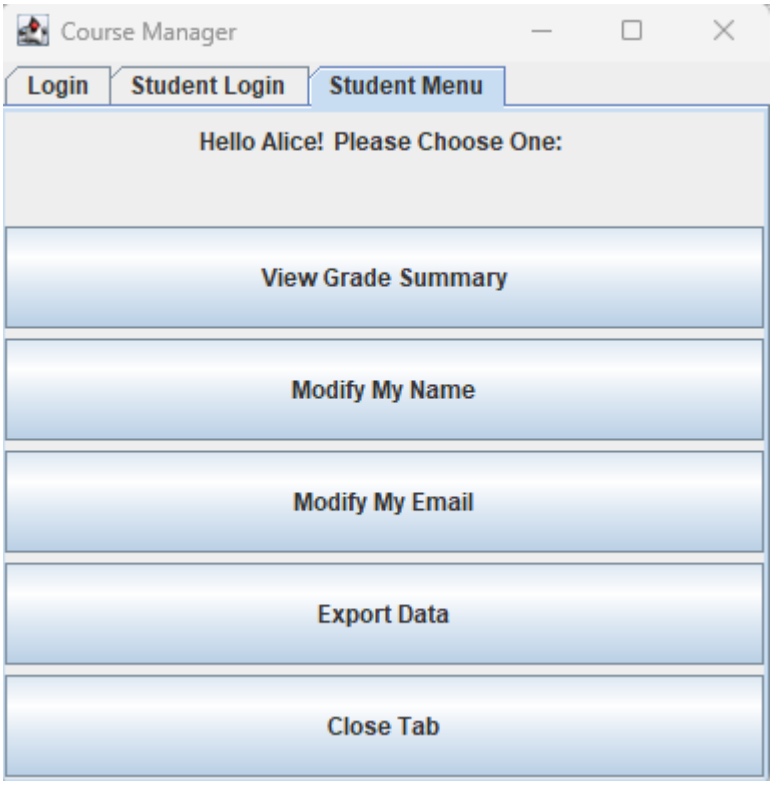
Login Menu:



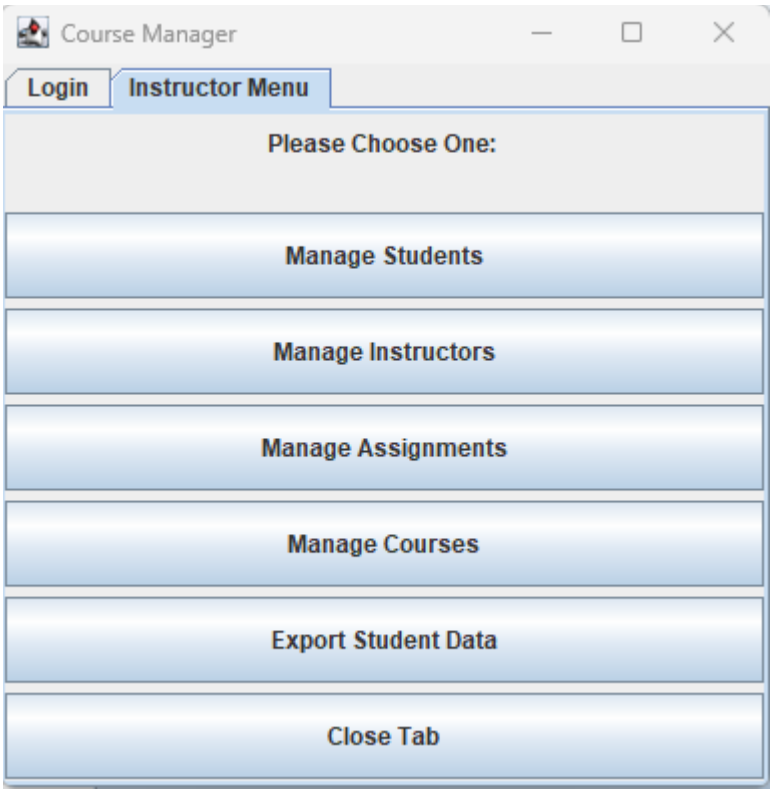
Student Login:



Student Menu:



Instructor Menu:



Class Designs

Person

Basic person class to be utilized by Instructor and Student.

Variable	Purpose
private String name private String email	Private variables

Method or Constructor	Purpose
public Person(String name, String email)	Constructor
public void setName(String name) public void setEmail(String email)	Setters
public String getName() public String getEmail()	Getters
public printDetails()	Print to Console

Instructor extends Person

Inherits basic Person class and handles/stores instructors.

Variable	Purpose
private String courseName	Private variable

Method or Constructor	Purpose
public Instructor(String name, String email, String courseName)	Constructor
public String setCourseName(String courseName)	Setter
public String getCourseName()	Getter
public void printDetails()	Print to Console

Student extends Person

Inherits basic Person class and handles/stores students.

Variable	Purpose
private String studentID	Private variable
private ArrayList assignments	Hold student's assignments

Method or Constructor	Purpose
public Student(String name, String email, String studentID)	Constructor
public void setID(String studentID) public void addAssignment(String assignmentName, int score, int maxScore)	Setter / Mutator
public String getID() public ArrayList getAssignmnets() public Assignment getAssignment(String assignmentName)	Getters
public void printDetails()	Print to Console

Grade

Calculates and stores students' grades.

Variable	Purpose
private int grade	Private variables

Method or Constructor	Purpose
public void setGrade(int grade)	Setters

public int getGrade() public String getLetterGrade(Arraylist assignments) public void getClassGrades()	Getters

Course

For handling and storing course information.

Variable	Purpose
private String courseName private String roomNumber private String meetTime private String instructor priate String schedule private ArrayList roster	Private variables

Method or Constructor	Purpose
public Course(String courseName, String roomNumber, String meetTime, String instructor, String schedule)	Constructor
public void setCourseName(String courseName) public void setRoomNumber(String roomNumber) public void setMeetTime(String meetTime) public void setInstructor(String instructor) public void setSchedule(String schedule) public void setRoster(ArrayList roster) public void addStudent(Student student)	Setters
public String getCourseName() public String getRoomNumber() public String getMeetTime() public String getInstructor() public String getSchedule() public ArrayList getRoster()	Getters
public void printStudentRoster() public void printDetails()	Print to console

Assignment

For handling and storing assignment information.

Variable	Purpose

private String assignmentName private int score private int maxScore	Private variables

Method or Constructor	Purpose
public Assignment(String assignmentName, int score, int maxScore)	Constructor
public void setAssignmentName(String assignmentName) public void setScore(int score)	Setters
public String getAssignmentName() public int getMaxScore() public int getScore()	Getters
public void printDetails()	Print to console

Manager

For handling interaction between classes and storing objects.

Variable	Purpose
private ArrayList students private ArrayList instructors private ArrayList courses private Random random	Private variables

Method or Constructor	Purpose
public void addStudent(String email, String name, String studentID) public void getStudentGrade(String studentID) public ArrayList getStudentIDs() public void enrollStudent(String studentID, String courseName) public Student getStudent(String studentID)	Manage Students
public void addInstructor(String email, String name, String courseName) public Instructor getInstructor(String name)	Manage Instructors
public void addCourse(String courseName, String roomNumber, String meetTime, String instructor, String schedule)	Manage Courses

public Course getCourse(String name)	
public void addAssignment(String courseName, String assignmentName, int maxScore)	Manage Assignments
public String getExportFolder() public String getExportFilename() public String getInstructorExportFilePath() public String getStuExportFilePath(Student student) public void exportInstructorData(Instructor instructor) public void exportStudentData(Student student)	Manage Exporting
public void printStudents() public void printStudentIDs() public void printInstructors() public void printInstructorNames() public void printCourses() public void printCourseNames()	Print to Console

User Interface (Console)

Interact with the user via the console and the program via manager.

Variable	Purpose
private Manager manager private Scanner scanner	Private variables

Method or Constructor	Purpose
public UserInterface(Manager manager)	Constructor
public void startApp() private void printWelcomeMessage() private void printLoginScreen() private void printGoodbyeMessage()	Startup / Exit
private void printInstructorMenu() private void instructorMainMenuLoop() private void manageInstructorsMenu() private void manageInstructors()	Instructor
private void printStudentMenu() private void studentMainMenuLoop(Student student) private void manageStudentsMenu() private void manageStudents() private void gradeSummary(Student student)	Student

private void changeStudentName(Student student) private void changeStudentEmail(Student student)	
private void manageAssignmentsMenu() private void manageAssignments()	Assignment
private void manageCoursesMenu() private void manageCourses()	Courses
private void exportInstructorData() private void exportStudentData(Student student)	Exporting
private int readIntInput(String prompt) private String readStringInput(String prompt) private void waitForEnter()	Utility

BETA: GUI (Graphic User Interface)

Interact with the user via a GUI and the program via manager.

Plan for further development: Each menu has its own set of variables for visuals, as well as its own class acting as an ActionListener. This can be seen in the current code, but given future development a full documentation would be completed.