Student Learning Management System

Team FireDrake

Introduction to Programming - CMPT 120L

MARIST COLLEGE
ORARE ET LABORARE
1929
POUGHKEEPSIE, N.Y.

Professor Reza Sadeghi

November 20th, 2022

Table of Contents

Table of Figures

Project Progress Report 1 of the Student Learning Management System

Team Name

Name of the Team ---Team FireDrake

Github Link

https://github.com/cfsarmiento/CMPT120L_StudentLearningManagementSystem_TeamFireDrake

Team Members

1. Christian Sarmiento---------------------------------christian.sarmiento1@marist.edu (Team Head)

2. Gabrielle Knapp--------------------------------------gabrielle.knapp1@marist.edu (Team Member)

3. Michael McMahon-------------------------------michael.mcmahon2@marist.edu (Team Member)

4. Ethan Morton -- ethan.morton1@marist.edu (Team Member)

5. Cody Carruthers--------------------------------------cody.carruthers1@marist.edu (Team Member)

6. Paul Bergeron---paul.bergeron1@marist.edu (Team Member)

Description of Team Members

1. ***Christian Sarmiento***

   I am a transfer sophomore student from White Plains, NY, transferring from Westchester Community College. I am majoring in Data Science & Analytics with a minor in Business, and I know Python and R proficiently. I have done research projects with multivariable regression machine learning prediction models, as well as basic experience in backend development for a website utilizing API data. I have proficient knowledge in statistical analysis and data cleaning with R. Currently at Marist, I am a part of the Computer Science Society, Marist S.T.A.T, and the Marist Hispanic Society. In the future, I wish to potentially add an additional minor in Computer Science, join the Marist

Analytics Club if there is a board again, as well as furthering my resume through side projects, undergraduate research, and relevant coursework.

2. ***Gabrielle Knapp***

   Gabrielle Knapp is a freshman at Marist College double majoring in Computer Science and Economics with minors in Spanish and Information Science. She comes from Harrisburg, Pennsylvania and has been enjoying her time adjusting to college life. Activities she currently is participating in include intramural badminton, student government, model united nations, and campus ministry. In her free time, she loves reading, going on long walks, and exploring the outdoors. One thing she misses most about home is going on weekly hikes around the Appalachian Mountain Range that surrounds Harrisburg. She is excited to see how she learns and grows her programming skills throughout this class and her entire time at Marist. Gabrielle is excited to bring her hardworking, can-do attitude to her group in this project for her Introduction to Programming class.

3. ***Michael McMahon***

   I am a freshman from Long Island, New York and I am majoring in Computer Science with a minor in music and in the future, possibly a major or minor in cyber security. I love to hang out with friends, watch sports particularly Football and Baseball, listen to music, and play video games. Besides Python, I also have some experience working with Javascript. I am currently in the games society, computer society and I am a member of the Symphonic band and the brass ensemble. I am looking forward to developing my skills in programming throughout my time here and possibly branch out to other things using my knowledge.

4. ***Ethan Morton***

   I am a freshman and I'm majoring in computer science with a concentration in software development. I plan on minoring in mathematics, data science, information technology, information systems, and cybersecurity. Besides python, which I'm currently learning, I know Java, C#, HTML, CSS, javascript, and a little bit of SQLite, XML, and Lua. I am from Granby, Connecticut and I enjoy playing tennis, hanging out with my friends, and playing games. The clubs I'm in include: campus ministry, computer society, games society, and I hope to join club tennis next semester.

5. ***Cody Carruthers***

   I am a freshman majoring in Computer science and I have a concentration in game design/development. I know Python, C#, Lua, Javascript, Java, and C++ and I used to teach coding before college. I love playing video games and I'm in the games society and love to play soccer. I'm from Southern Connecticut on Long Island Sound.

6. ***Paul Bergeron***

   I am a freshman here at Marist college majoring in cyber security with minors in Information Systems, information sciences, and computer science. I am from Ellicott City, Maryland, and I enjoy playing basketball, working out, playing video games, and hanging out with friends. I am on two esports teams here at Marist, the collegiate League of Legends A team, and collegiate CS:GO team. I know java and am currently learning python. I am currently applying to internships and am excited to see what the future might hold.

Project Descriptions

Summary

This project focuses on creating an app that allows students to manage their classes, accompanying assignments, and grades. This application will make it very easy for the student to view how well they are doing within a given class, semester (past and present), and as well as cumulatively throughout their years here at Marist. The student is able to provide their courses and respective assignment requirements at the beginning of the semester, and as the semester progresses the student is able to provide their grades. This application will give them a rundown of their grades in each of their classes at any given point in the semester (letter grade and number grade), as well as their GPA for that semester at that given point. Additionally, this application will have a Potential GPA Calculator feature where a student is able to put in hypothetical grades for each class and get a potential GPA in return.

Requirements

- Username & password, ability to change both
- The user should be able to:
    - Input list of current and future course rosters
    - Input past GPAs by semester
    - Input entry and graduating month/year (to calculate how many past semesters have been taken)
    - Ability to input each assignment by type for every class. The assignments can have grades put into them at any point
    - Input custom weights for grades for each assignment type in every class

- The Program is Capable of Taking Input and:
  - Calculate grades for each class (letter and number), with weights for each assignment
  - Calculate semester GPA, as well as accumulative GPA
  - Keeps track of past semester GPAs
  - Detect how many past semesters have been taken
  - Keep track of class assignments, by type, in tables
  - Calculate potential GPA given hypothetical class overall grades
    - User will have to provide a grade for each class, calculator defaults each value for class to current grade in the class

Interface Description

- Login Screen
  - For non-first time: username and password
  - First Time Set Up:
    - Username, password (can be changed later in settings)
- Main Home Page
  - Main Pane
    - Table of assignments, by class (selected on the sidebar)
      - Assignment type
      - Assignment grade
    - On top of the table: course name, course grade, semester GPA, & accumulative GPA, and status will be displayed

            - Status refers to if the class is being completed or if the student has dropped/withdrawn from the class
        - Add/Edit Assignments button where you can add, remove and edit assignments
    - Sidebar
        - Classes (each class is a button that will display the grade and assignment info in the main pane
        - Cumulative Rundown (Displayed in Main Pane)
            - Table of all semesters and their GPAs
                - Semester Year, Session (Fall/Winter/Sprint/Summer), and Final GPA will be displayed in tabular form
        - Potential GPA calculator
        - Settings
            - Add New Class
                - Course Abbreviation/Level (Ex. CMPT 120L)
                - Course Name (Ex. Intro to Programming)
                - Assignment Types (Select all that apply)
                    - Homework
                    - Tests
                    - Projects
                    - Quizzes
                    - Papers
                - Assignment Weights

- Drop/Withdrawal From Current Class
    - Able to select from current classes
    - Drop or withdrawal buttons
- Semester Settings (Pop Up)
    - Add Past Final Semester GPA (If applicable - cannot be undone)
        - Year
        - Session
        - Final GPA
    - New Semester (Pop Up)
        - Year
        - Session

Graphical User Experience Design

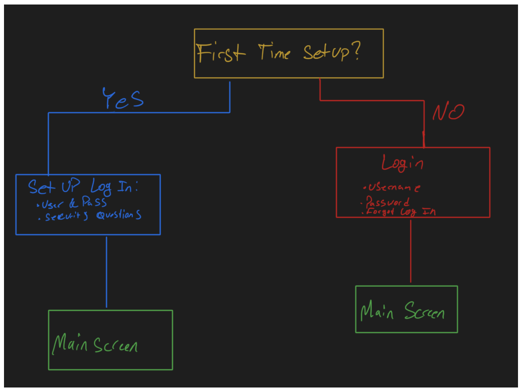
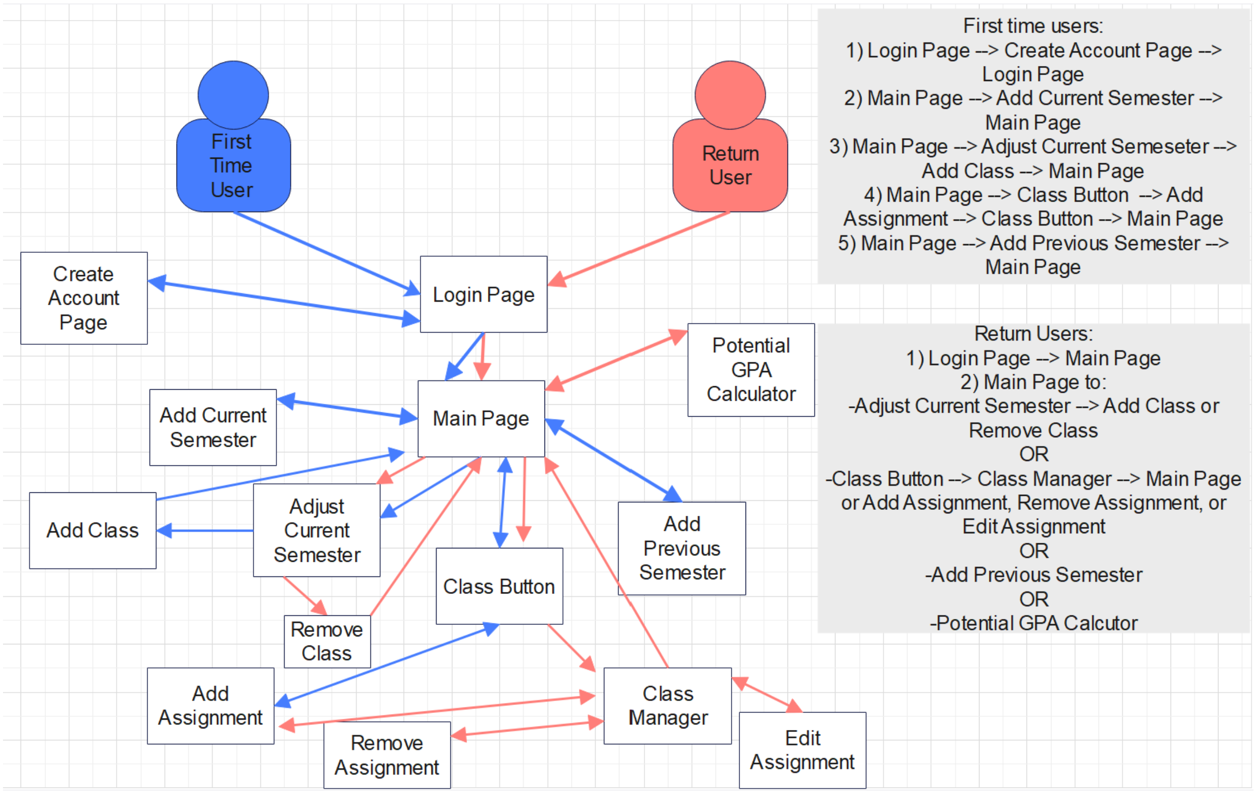


figure 1 (login GUE design)



First time users:
1) Login Page --> Create Account Page --> Login Page
2) Main Page --> Add Current Semester --> Main Page
3) Main Page --> Adjust Current Semeseter --> Add Class --> Main Page
4) Main Page --> Class Button --> Add Assignment --> Class Button --> Main Page
5) Main Page --> Add Previous Semester --> Main Page

Return Users:
1) Login Page --> Main Page
2) Main Page to:
-Adjust Current Semester --> Add Class or Remove Class
OR
-Class Button --> Class Manager --> Main Page or Add Assignment, Remove Assignment, or Edit Assignment
OR
-Add Previous Semester
OR
-Potential GPA Calcutor

figure 2 (page connection web)

Graphical User Interface Design

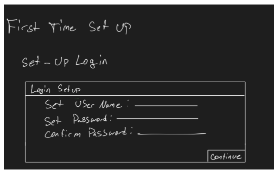


figure 3 (first time set up design)

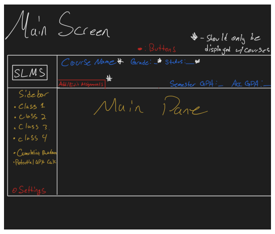


figure 4 (main screen design)

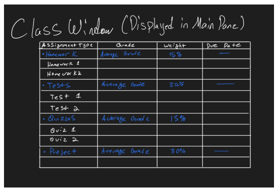


figure 5 (class window design)

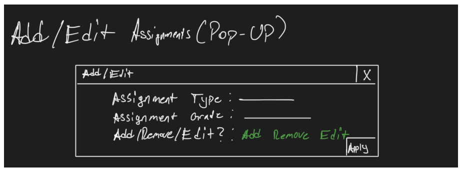

Add/Edit Assignments (Pop-Up)
Add/Edit
Assignment Type: ______
Assignment Grade: ______
Add/Remove/Edit?: Add Remove Edit
Apply

figure 6 (add/edit assignment design)

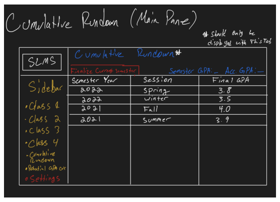


figure 7 (cumulative page design)

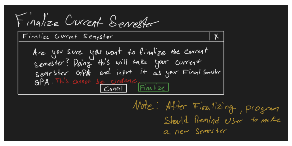

Finalize Current Semester
Cancel
Finalize
Note: After Finalizing, program should Remind User to make a new Semester

figure 8 (finalize semester design)

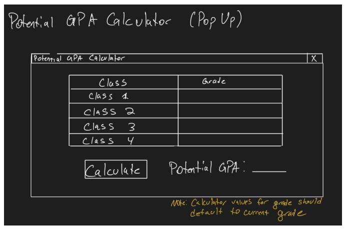


figure 9 (potential gpa calc design)

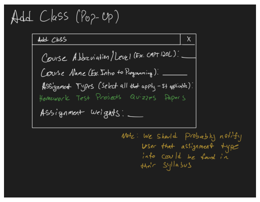

Add Class (Pop-Up)
Add Class
Course Abbreviation/Level: ______
Course Name: ______
Assignment Types
Homework Tests Projects Quizzes Papers
Assignment Weights: ______

figure 10 (add class design)

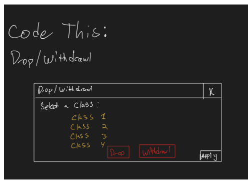


figure 11 (drop class design)

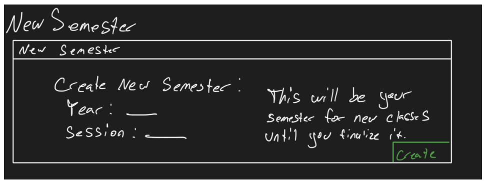


figure 12 (add new semester design)

Implementation

Login

<u>Interface Description:</u> Two labels to the left of two entry boxes, and then two buttons

<u>Image of Output:</u>

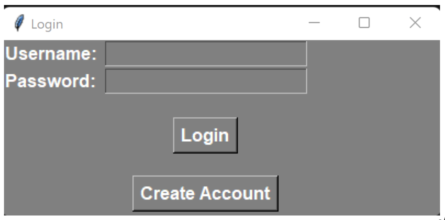


figure 13 (login screen)

<u>Code Description:</u> This is the window that pops up to take the user's input of their unique username and passcode. The user has the option to create a new account through this window or to enter credentials if they already have an account and this will send them to the main page.

<u>Code:</u>

```
import tkinter as tk
from tkinter import *
window=tk.Tk()
window.configure(bg='grey')
window.title('Login')
window.geometry('400x160')
usernameLabel=tk.Label(window, text = 'Username: ', bg='grey', fg = 'white', font='Helvetica 12 bold')
usernameLabel.grid(column = 0, row = 0)
passwordLabel=tk.Label(window, text = 'Password: ', bg='grey', fg = 'white', font='Helvetica 12 bold')
passwordLabel.grid(column = 0, row = 1)
usernameEntry=tk.Entry(window, bg = 'grey', fg='white', font="Helvetica 12 bold" )
usernameEntry.grid(row = 0, column = 1)
passwordEntry=tk.Entry(window, bg = 'grey', fg='white', font="Helvetica 12 bold")
passwordEntry.grid(row = 1, column = 1)
loginButton=tk.Button(window, bg = 'grey', fg='white', font= 'Helvetica 12 bold',text = 'Login', command = Login)
loginButton.grid(column = 1, row = 3, pady = 20)
createAccountButton=tk.Button(window, bg = 'grey', fg='white', font='Helvetica 12 bold', text = 'Create Account', command = CreateAccount)
createAccountButton.grid(column = 1, row = 4)
window.mainloop()
```

Create Account

<u>Interface Description:</u> A set of three labels with entry boxes next to each one to get user input, then a finalization button on the bottom.

<u>Image of Output:</u>

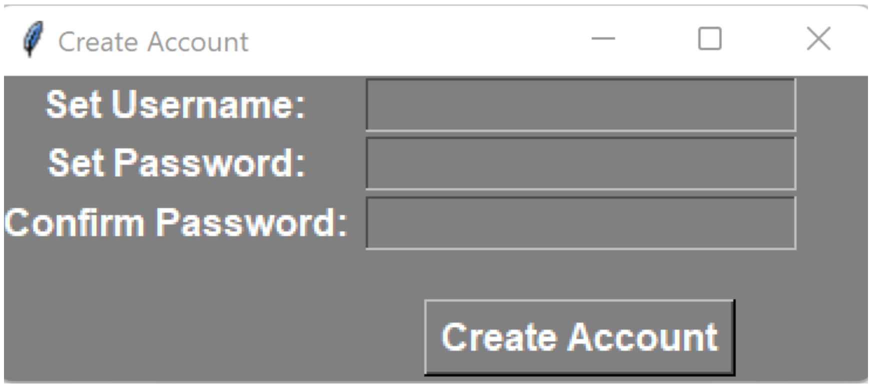


figure 14 (create account screen)

<u>Code Description:</u> This allows the user to start their process in the SLMS by creating an account via a username/passcode that is saved into a csv file.

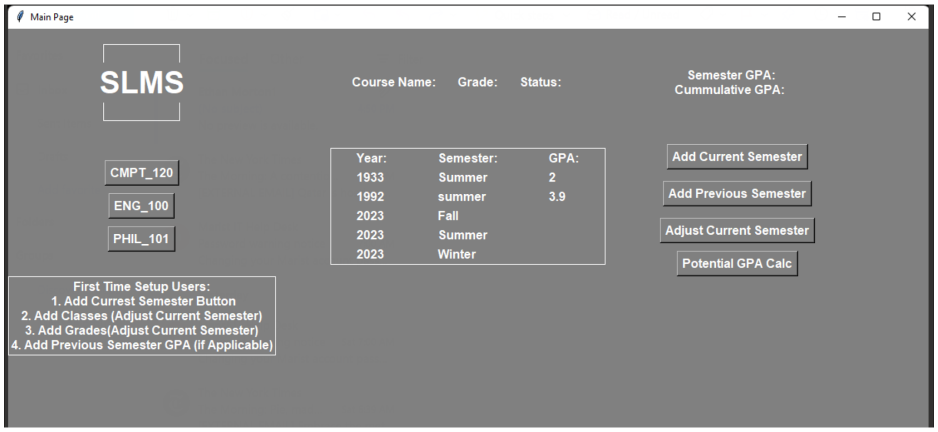
<u>Code:</u>

```
import tkinter as tk
from tkinter import *
window=tk.Tk()
window.configure(bg='grey')
window.title('Create Account')
window.geometry('375x130')
usernameLabel=tk.Label(window, text = 'Set Username: ', fg = 'white', bg='grey', font='Helvetica 12 bold')
usernameLabel.grid(column = 0, row = 0)
passwordLabel=tk.Label(window, text = 'Set Password: ', fg = 'white', bg='grey', font='Helvetica 12 bold')
passwordLabel.grid(column = 0, row = 1)
confirmLabel=tk.Label(window, text = 'Confirm Password: ', fg = 'white', bg='grey', font='Helvetica 12 bold')
confirmLabel.grid(column = 0, row = 2)
usernameEntry=tk.Entry(window, bg = 'grey', fg='white', font='Helvetica 12 bold')
usernameEntry.grid(row = 0, column = 1)
passwordEntry=tk.Entry(window, bg = 'grey', fg='white', font='Helvetica 12 bold')
passwordEntry.grid(row = 1, column = 1)
confirmEntry=tk.Entry(window, bg = 'grey', fg='white', font='Helvetica 12 bold')
confirmEntry.grid(row = 2, column = 1)
createAccountButton=tk.Button(window, bg = 'grey', fg='white', font='Helvetica 12 bold', text = 'Create Account', command = CreateAccount)
createAccountButton.grid(column = 1, row = 3, pady = 20)
window.mainloop()
```

Main Screen

Interface Description: Five frames forming a larger rectangle, one with the project title in the upper left, one displaying the buttons of the classes on the left hand side and an instruction guide for first time users, one with a label on the top which will display information when a class button is clicked, one displaying the cumulative rundown of semesters and gpas in the center of the page, and one on the right hand side of the page with several button options.

Image of Output:



Year:	Semester:	GPA:
1933	Summer	2
1992	summer	3.9
2023	Fall	
2023	Summer	
2023	Winter	

First Time Setup Users:
1. Add Currest Semester Button
2. Add Classes (Adjust Current Semester)
3. Add Grades(Adjust Current Semester)
4. Add Previous Semester GPA (if Applicable)

figure 15 (main screen)

Code Description: This is the main page of the project and holds all commands and windows that the user can access through the SLMS. It holds buttons for classes the user has inputted, which when clicked will display key class information such as grade, assignments, and status. It holds an instruction checklist for first-time users, it holds a textbox including a cumulative rundown of gpas from the user's previous and current semesters. It also holds buttons to access actions such as "add current semester", "add previous semester", "adjust current semester", and "potential gpa calculator". This is the hub of this app.

Code:

```
import tkinter as tk

from tkinter import *
```

```
window=tk.Tk()

window.title("Main Page")

window.geometry("1200x550")

window.config(bg="Gray")

frame1=tk.Frame(window,bg="Gray",highlightbackground="White",highlightthickness=1,width=100,height=100)

frame1.grid(row=0,column=0)

tk.Label(window,bg="Gray",fg="White",text="SLMS",font='Helvetica 30 bold').grid(row=0,column=0)

frame2=tk.Frame(window,bg="Gray",width=500,height=140)

frame2.grid(row=0,column=1)

label= tk.Label(bg="Gray", fg="White", text="Course Name:       Grade:       Status:       ", font='Helvetica 12 bold').grid(row=0,column=1)

label= tk.Label(bg="Gray", fg="White", text="Semester GPA:    \nCummulative GPA:     ", font='Helvetica 12 bold').grid(row=0,column=2)

frame3=tk.Frame(window,bg="Gray",width=249,height=249)

frame3.grid(row=1,column=0)

frame4=tk.Frame(window,bg="Gray",highlightbackground="White",highlightthickness=1,width=500,height=100)

frame4.grid(row=1,column=1)

#textBox1=Text(frame4, bg="Gray", fg="White").grid(row=1, column=0)

yearFrame = tk.Frame(frame4, bg = "Gray", width = 166, height = 300, padx = 30)

yearFrame.grid(row=0,column=0)

semesterFrame = tk.Frame(frame4, bg = "Gray", width = 166, height = 300, padx = 30)

semesterFrame.grid(row=0,column=1)

gpaFrame = tk.Frame(frame4, bg = "Gray", width = 166, height = 300, padx = 30)

gpaFrame.grid(row=0,column=2)

yearLabel = tk.Label(yearFrame, bg = "Gray", fg = "White", text = "Year:", font = 'Helvetica 12 bold')

yearLabel.grid(row=0,column=0, sticky = W)

semesterLabel = tk.Label(semesterFrame, bg = "Gray", fg = "White", text = "Semester:", font = 'Helvetica 12 bold')

semesterLabel.grid(row=0,column=0, sticky = W)

gpaLabel = tk.Label(gpaFrame, bg = "Gray", fg = "White", text = "GPA:", font = 'Helvetica 12 bold')

gpaLabel.grid(row=0,column=0, sticky = W)

frame6=tk.Frame(window,bg="Gray",highlightbackground="White",highlightthickness=1,width=500,height=150)

frame6.grid(row=2,column=0)

ftsLabel = tk.Label(frame6, bg = 'grey', fg = 'White', text = 'First Time Setup Users:\n 1. Add Currest Semester Button\n2. Add Classes (Adjust Current Semester)\n3. Add Grades(Adjust Current Semester)\n4. Add
Previous Semester GPA (if Applicable)',font = 'Helvetica 12 bold')

ftsLabel.grid(row=2,column=0,sticky = W)

frame5=tk.Frame(window,bg="Gray", width=500,height=300)

frame5.grid(row=1, column=2)

tk.Button(frame5,bg="Gray",fg="White",text="Add Current Semester",font='Helvetica 12 bold', command = AddCurrentSemester).grid(row=0,column=2, pady=10)

tk.Button(frame5,bg="Gray",fg="White",text="Add Previous Semester",font='Helvetica 12 bold', command = AddPreviousSemester).grid(row=1, column=2, pady=5)

tk.Button(frame5,bg="Gray",fg="White",text="Adjust Current Semester",font='Helvetica 12 bold', command = AdjustSemester).grid(row=2,column=2, pady=10)

tk.Button(frame5,bg="Gray",fg="White",text="Potential GPA Calc",font='Helvetica 12 bold', command = PotentialGPA_Calculator).grid(row=3,column=2)

window.mainloop()
```

Add Current Semester

Interface Description: A label on top of two labels alongside two entry boxes to get input from users. The bottom consists of two buttons, one to finalize the creation and one to go back.

Image of Output:

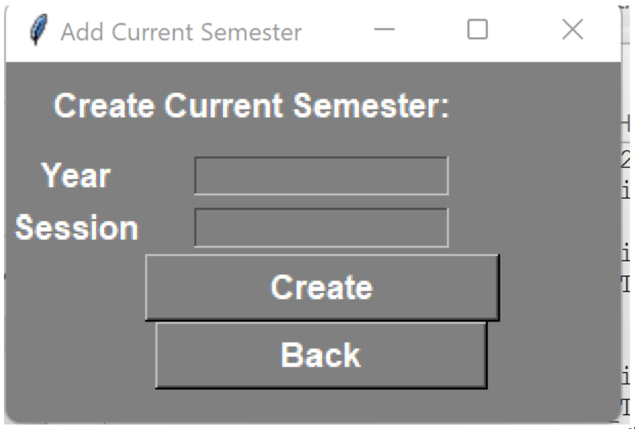


figure 16 (add current semester screen)

Code Description: This allows the user to start their process in the SLMS by adding the current semester, to which they can add classes and grades.

Code:

```
import tkinter as tk
from tkinter import *
new_semester = tk.Tk()
new_semester.title('Add Current Semester') # title
new_semester.geometry('300x175')  # window dimensions
new_semester.configure(bg = 'grey')  # color
newSemesterFrame = tk.Frame(new_semester,bg = 'grey',width = 100,height = 50)
newSemesterFrame.grid(row = 1, column = 0)
createSemesterLabel = tk.Label(new_semester,text = 'Create Current Semester:',bg = 'grey', fg = 'white',font = 'Helvetica 12 bold', padx = 20,pady = 10)
createSemesterLabel.grid(row = 0, column = 0)
newYearLabel = tk.Label(newSemesterFrame,text = 'Year',bg = 'grey',  fg = 'white',font='Helvetica 12 bold')
newYearLabel.grid(row = 0, column = 0)
entryNewYear = tk.Entry(newSemesterFrame, bg = 'grey')
entryNewYear.grid(row = 0, column = 1)
newSessionLabel = tk.Label(newSemesterFrame,text = 'Session', bg = 'grey',fg = 'white', font ='Helvetica 12 bold')
newSessionLabel.grid(row = 1, column = 0)
entrySessionYear = tk.Entry(newSemesterFrame,bg = 'grey')
entrySessionYear.grid(row = 1, column = 1)
btnNewSemester = tk.Button(newSemesterFrame,text = 'Create',bg = 'grey', fg='white', font='Helvetica 12 bold',padx = 55,command = CreateSemester)
btnNewSemester.grid(row = 2, column = 1)
backButton = tk.Button(newSemesterFrame,text = 'Back',bg = 'grey', fg='white', font='Helvetica 12 bold',padx = 55,command = Back)
backButton.grid(row = 3, column = 1)
new_semester.mainloop()
```

Adjust Current Semester

Interface Description: Presents the user with a label and then four buttons, which will direct the user to different windows.

Image of Output:

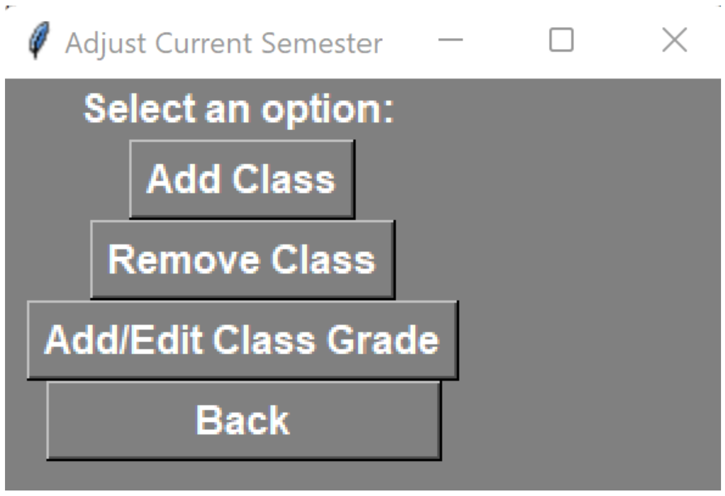


figure 17 (adjust current semester screen)

Code Description: This allows the users to edit data already entered by adding/removing a class or adding/editing a class grade.

Code:

```
import tkinter as tk
window = tk.Tk()
window.title('Adjust Current Semester')  # title for window
window.geometry('300x300')  # length x width
window.configure(bg = 'grey')  # background color
frame1 = tk.Frame(window, bg = 'grey', width = 200).grid(row=1, column=1)
lbl1=tk.Label(frame1, text="Select an option: ", bg = 'grey',fg = 'white',font = 'Helvetica 12 bold').grid(row = 0, column = 1)
btn1=tk.Button(frame1,text = "Add Class",bg = 'grey',fg = 'white',font = 'Helvetica 12 bold', command=AddClass).grid(row = 1, column = 1)
btn2=tk.Button(frame1,text = "Remove Class",bg = 'grey',fg = 'white',font = 'Helvetica 12 bold', command=RemoveClass).grid(row = 2, column = 1)
btn3=tk.Button(frame1,text = "Add/Edit Class Grade",bg = 'grey',fg = 'white',font = 'Helvetica 12 bold', command=EditClass).grid(row = 3, column = 1)
backButton = tk.Button(frame1,text = 'Back',bg = 'grey', fg='white', font='Helvetica 12 bold',padx = 55,command = Back)
backButton.grid(row = 4, column = 1)
window.mainloop()
```

Add Class

Interface Description: A label in front of an entry box to get input from user, then another label and five buttons that turn color when pressed above five entry boxes to get numerical input from users. Then a label in front of three more buttons that change color when pressed, and finally a finalization button on the bottom of the window.

Image of Output:

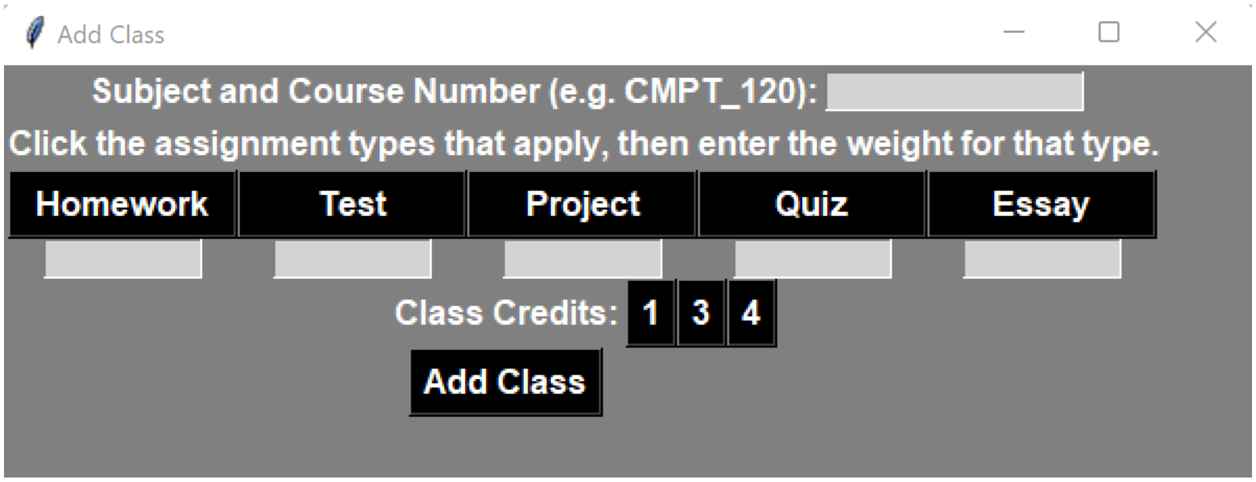

Add Class
Subject and Course Number (e.g. CMPT_120):
Click the assignment types that apply, then enter the weight for that type.
Homework
Test
Project
Quiz
Essay
Class Credits:
1
3
4
Add Class

figure 18 (add class screen)

Code Description: This allows us to get data inputted from the user on the classes they are taking. We get the class name, assignment types and weights, and the number of credits.

Code:

```
import tkinter as tk
from tkinter import *
window=tk.Tk()
window.configure(bg='grey')
window.title('Add Class')
window.geometry('600x200')
labelFrame = tk.Frame(window, bg='grey')
labelFrame.grid(row = 0, column = 0)
secondFrame = tk.Frame(window,bg='grey')
secondFrame.grid(row = 1, column = 0)
buttonFrame = tk.Frame(window,bg='grey')
buttonFrame.grid(row = 2, column = 0)
bottomFrame = tk.Frame(window,bg='grey')
bottomFrame.grid(row = 3, column = 0)
```

```
creditsFrame = tk.Frame(bottomFrame,bg='grey')

creditsFrame.grid(row = 1, column = 1)

courseAbbrLabel = tk.Label(labelFrame, text = 'Subject and Course Number (e.g. CMPT_120):', fg = 'white', bg='grey', font='Helvetica 12 bold' )

courseAbbrLabel.grid(column = 0, row = 0)

courseAbbrEntry = tk.Entry(labelFrame, bg = 'lightgrey')

courseAbbrEntry.grid(column = 1, row = 0)

assignmentTypeLabel = tk.Label(secondFrame, text = 'Click the assignment types that apply, then enter the weight for that type.', bg='grey', font='Helvetica 12 bold',fg = 'white')

assignmentTypeLabel.grid(column = 0, row = 0)


homeworkButton=tk.Button(buttonFrame, bg = 'black', text = 'Homework', fg='white',font='Helvetica 12 bold', width = 10, command = Homework)

homeworkButton.grid(column = 0, row = 0)

homeworkEntry = tk.Entry(buttonFrame, bg = 'lightgrey', width = 12)

homeworkEntry.grid(column = 0, row = 1)

testButton=tk.Button(buttonFrame, bg = 'black', text = 'Test', fg='white',width = 10, font='Helvetica 12 bold',command = Test)

testButton.grid(column = 1, row = 0)

testEntry = tk.Entry(buttonFrame, bg = 'lightgrey', width = 12)

testEntry.grid(column = 1, row = 1)

projectButton=tk.Button(buttonFrame, bg = 'black', text = 'Project', fg='white',width = 10, font='Helvetica 12 bold',command = Project)

projectButton.grid(column = 2, row = 0)

projectEntry = tk.Entry(buttonFrame, bg = 'lightgrey', width = 12)

projectEntry.grid(column = 2, row = 1)

quizButton=tk.Button(buttonFrame, bg = 'black', text = 'Quiz', font='Helvetica 12 bold', fg='white',width = 10, command = Quiz)

quizButton.grid(column = 3, row = 0)

quizEntry = tk.Entry(buttonFrame, bg = 'lightgrey', width = 12)

quizEntry.grid(column = 3, row = 1)

essayButton=tk.Button(buttonFrame, bg = 'black', text = 'Essay', font='Helvetica 12 bold',fg='white',width = 10, command = Essay)

essayButton.grid(column = 4, row = 0)

essayEntry = tk.Entry(buttonFrame, bg = 'lightgrey', width = 12)

essayEntry.grid(column = 4, row = 1)

creditsLabel = tk.Label(bottomFrame, text = 'Class Credits:', font='Helvetica 12 bold', fg='white',bg = 'grey')

creditsLabel.grid(column = 0, row = 1)

credit1Button = tk.Button(creditsFrame, bg = 'black', fg='white',font='Helvetica 12 bold',text = '1', command = Credit1)

credit1Button.grid(column = 0, row = 0)

credit3Button = tk.Button(creditsFrame, bg = 'black', fg='white',font='Helvetica 12 bold',text = '3', command = Credit3)

credit3Button.grid(column = 1, row = 0)

credit4Button = tk.Button(creditsFrame, bg = 'black', fg='white',font='Helvetica 12 bold',text = '4', command = Credit4)

credit4Button.grid(column = 2, row = 0)

addButton = tk.Button(bottomFrame, bg = 'black', fg='white',font='Helvetica 12 bold',text = 'Add Class', command = AddClass)

addButton.grid(column = 0, row = 2)

window.mainloop()
```

Add/Edit Grades

Interface Description: A main label, and then three labels to the left of three entry boxes, with a button at the bottom right hand corner.

Image of Output:

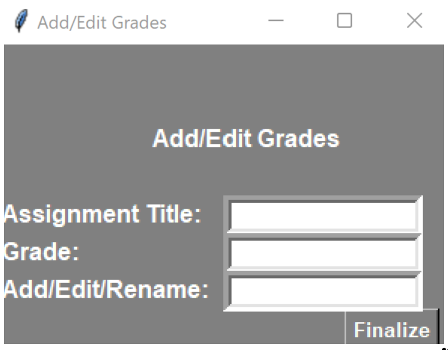


figure 19 (add/edit grades screen)

Code Description: This is the window that allows the user to add or edit a grade into the csv file.

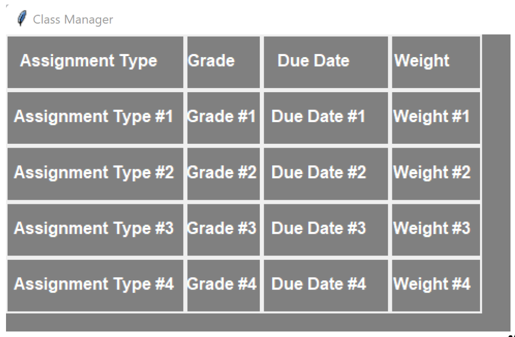
Code:

```
from tkinter import *
window=Tk()
window.configure(bg='grey')
btn1=Button(window, text="Finalize", bg='grey', fg='white',font=("Helvetica 10 bold"))
btn1.place(x=230, y=175)
lbl1=Label(window, text="Assignment Title: ", bg='grey', fg='white', font=("Helvetica 12 bold"))
lbl1.place(x=0, y=100)
lbl1=Label(window, text="Grade: ", bg='grey', fg='white', font=("Helvetica 12 bold"))
lbl1.place(x=0, y=125)
lbl1=Label(window, text="Add/Edit/Rename: ", bg='grey', fg='white', font=("Helvetica 12 bold"))
lbl1.place(x=0, y=150)
lbl=Label(window, text="Add/Edit Grades", bg= 'grey', fg='white', font=("Helvetica 12 bold"))
lbl.place(x=100, y=50)
txtfld1=Entry(window, text="Assignment Title: ", bd=5)
txtfld1.place(x=150, y=100)
txtfld2=Entry(window, text="Grade: ", bd=5)
txtfld2.place(x=150, y=125)
txtfld3=Entry(window, text="Add/Edit/Rename: ", bd=5)
txtfld3.place(x=150, y=150)
window.title('Add/Edit Grades')
window.geometry("300x200+20+20")
window.mainloop()
```

Class Manager

Interface Description: Uses canvases to create a grid of outputs.

Image of Output:


Class Manager

Assignment Type	Grade	Due Date	Weight
Assignment Type #1	Grade #1	Due Date #1	Weight #1
Assignment Type #2	Grade #2	Due Date #2	Weight #2
Assignment Type #3	Grade #3	Due Date #3	Weight #3
Assignment Type #4	Grade #4	Due Date #4	Weight #4

figure 20 (class manager screen)

Code Description: Allows user to see the assignments, grades, due dates, and weight amounts for all of the assignments in a particular class.

Code:

```
import tkinter as tk
from tkinter import *
window=Tk()
window.title('Class Manager')
window.geometry('1000x300')
window.configure(bg='grey')
canvas=Canvas(window, width= 170, height= 50, bg="grey")
canvas2=Canvas(window, width= 70, height= 50, bg="grey")
canvas3=Canvas(window, width= 120, height= 50, bg="grey")
canvas4=Canvas(window, width= 85, height= 50, bg="grey")
canvas5=Canvas(window, width= 170, height= 50, bg="grey")
canvas6=Canvas(window, width= 170, height= 50, bg="grey")
canvas7=Canvas(window, width= 170, height= 50, bg="grey")
canvas8=Canvas(window, width= 170, height= 50, bg="grey")
canvas9=Canvas(window, width= 70, height= 50, bg="grey")
canvas10=Canvas(window, width= 70, height= 50, bg="grey")
canvas11=Canvas(window, width= 70, height= 50, bg="grey")
canvas12=Canvas(window, width= 70, height= 50, bg="grey")
canvas13=Canvas(window, width= 120, height= 50, bg="grey")
canvas14=Canvas(window, width= 120, height= 50, bg="grey")
canvas15=Canvas(window, width= 120, height= 50, bg="grey")
canvas16=Canvas(window, width= 120, height= 50, bg="grey")
canvas17=Canvas(window, width= 85, height= 50, bg="grey")
canvas18=Canvas(window, width= 85, height= 50, bg="grey")
canvas19=Canvas(window, width= 85, height= 50, bg="grey")
canvas20=Canvas(window, width= 85, height= 50, bg="grey")
canvas.create_text(80,25, text="Assignment Type", fill="white", font=('Helvetica 12 bold'))
canvas.grid(row=0,column=0)
canvas2.create_text(25,25, text="Grade", fill="white", font=('Helvetica 12 bold'))
```

```
canvas2.grid(row=0, column=1)
canvas3.create_text(50,25, text="Due Date", fill="white", font=('Helvetica 12 bold'))
canvas3.grid(row=0, column=2)
canvas4.create_text(30,25, text="Weight", fill="white", font=('Helvetica 12 bold'))
canvas4.grid(row=0, column=3)
canvas5.create_text(85,25, text="Assignment Type #1", fill="white", font=('Helvetica 12 bold'))
canvas5.grid(row=1, column=0)
canvas6.create_text(85,25, text="Assignment Type #2", fill="white", font=('Helvetica 12 bold'))
canvas6.grid(row=2, column=0)
canvas7.create_text(85,25, text="Assignment Type #3", fill="white", font=('Helvetica 12 bold'))
canvas7.grid(row=3, column=0)
canvas8.create_text(85,25, text="Assignment Type #4", fill="white", font=('Helvetica 12 bold'))
canvas8.grid(row=4, column=0)
canvas9.create_text(35,25, text="Grade #1", fill="white", font=('Helvetica 12 bold'))
canvas9.grid(row=1, column=1)
canvas10.create_text(35,25, text="Grade #2", fill="white", font=('Helvetica 12 bold'))
canvas10.grid(row=2, column=1)
canvas11.create_text(35,25, text="Grade #3", fill="white", font=('Helvetica 12 bold'))
canvas11.grid(row=3, column=1)
canvas12.create_text(35,25, text="Grade #4", fill="white", font=('Helvetica 12 bold'))
canvas12.grid(row=4, column=1)
canvas13.create_text(55,25, text="Due Date #1", fill="white", font=('Helvetica 12 bold'))
canvas13.grid(row=1, column=2)
canvas14.create_text(55,25, text="Due Date #2", fill="white", font=('Helvetica 12 bold'))
canvas14.grid(row=2, column=2)
canvas15.create_text(55,25, text="Due Date #3", fill="white", font=('Helvetica 12 bold'))
canvas15.grid(row=3, column=2)
canvas16.create_text(55,25, text="Due Date #4", fill="white", font=('Helvetica 12 bold'))
canvas16.grid(row=4, column=2)
canvas17.create_text(40,25, text="Weight #1", fill="white", font=('Helvetica 12 bold'))
canvas17.grid(row=1, column=3)
canvas18.create_text(40,25, text="Weight #2", fill="white", font=('Helvetica 12 bold'))
canvas18.grid(row=2, column=3)
canvas19.create_text(40,25, text="Weight #3", fill="white", font=('Helvetica 12 bold'))
canvas19.grid(row=3, column=3)
canvas20.create_text(40,25, text="Weight #4", fill="white", font=('Helvetica 12 bold'))
canvas20.grid(row=4, column=3)
window.mainloop()
```

Add Previous Semester

Interface Description: A label on top, then three labels in front of three entry boxes. Then two buttons: one to finalize the past semester data, the other to return the user to the main page.

Image of Output:

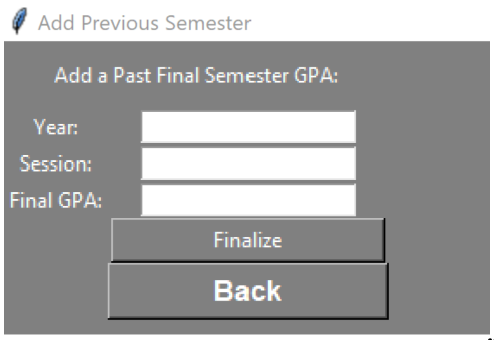


figure 21 (add previous semester screen)

Code Description: This allows the user to add a gpa from a previous semester.

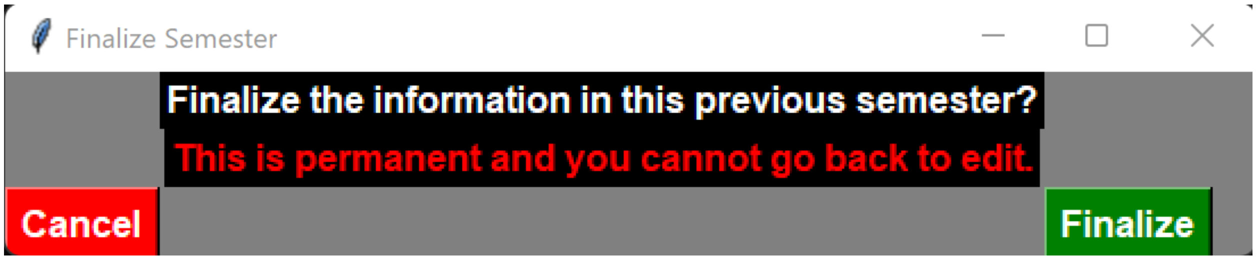
Code:

```
import tkinter as tk
from tkinter import *
semester_settings = tk.Tk()
semester_settings.title('Add Previous Semester')  # title for window
semester_settings.geometry('300x150')  # window size
semester_settings.configure(bg ='grey')  # color
inputFrame = tk.Frame(semester_settings,   bg = 'grey',width = 300,height= 100)
newSemesterFrame = tk.Frame(semester_settings,bg = 'grey',width = 200,height = 50)
inputFrame.grid(row = 1, column = 0)
newSemesterFrame.grid(row = 4, column = 0)
addGPALabel = tk.Label(semester_settings,text = 'Add a Past Final Semester GPA:',bg = 'grey',fg = 'white',padx = 10,pady = 10)
addGPALabel.grid(row = 0, column = 0)
yearLabel = tk.Label(inputFrame, text = 'Year:',bg = 'grey',  fg = 'white')
yearLabel.grid(row = 0, column = 0)
entryYear = tk.Entry(inputFrame, bg = 'white')
entryYear.grid(row = 0, column = 1)
sessionLabel = tk.Label(inputFrame,text = 'Session:', bg = 'grey', fg = 'white')
sessionLabel.grid(row = 1, column = 0)
entrySession = tk.Entry(inputFrame,bg = 'white')
entrySession.grid(row = 1, column = 1)
finalGPALabel = tk.Label(inputFrame,text = 'Final GPA:', bg = 'grey',fg = 'white')
finalGPALabel.grid(row = 2, column = 0)
entryFinalGPA = tk.Entry(inputFrame, bg = 'white')
entryFinalGPA.grid(row = 2, column = 1)
btnFinalize = tk.Button(inputFrame, text = 'Finalize', bg = 'grey',fg = 'white',padx = 55,command = Finalize)
btnFinalize.grid(row = 3, column = 1)
backButton = tk.Button(inputFrame,text = 'Back',bg = 'grey', fg='white', font='Helvetica 12 bold',padx = 55,command = Back)
backButton.grid(row = 4, column = 1)
semester_settings.mainloop()
```

Finalize Current Semester

Interface Description: A label warning users of the impacts of finalizing the data from a past semester and then two buttons on either bottom corner.

Image of Output:



Finalize the information in this previous semester?
This is permanent and you cannot go back to edit.

figure 22 (finalize semester screen)

Code Description: This gives users the opportunity to cancel or finalize changes they made to previous semester data.

Code:

```
import tkinter as tk
window = tk.Tk()
window.title('Finalize Semester')
window.geometry('550x80')
window.configure(bg = "grey")
descriptionText0 = tk.Label(window, text = "Finalize the information in this previous semester?", bg = "black", fg = 'white', font='Helvetica 12 bold')
descriptionText0.grid(column = 1, row = 0)
descriptionText1 = tk.Label(window, text = "This is permanent and you cannot go back to edit.", bg = "black", fg = 'red', font='Helvetica 12 bold')
descriptionText1.grid(column = 1, row = 1)
cancelButton = tk.Button(window, bg = "red", fg = "white", text = "Cancel", font='Helvetica 12 bold')
cancelButton.grid(column = 0, row = 2)
saveButton = tk.Button(window, bg = "green", fg = "white", text = "Finalize", font='Helvetica 12 bold')
saveButton.grid(column = 2, row = 2)
window.mainloop()
```

Potential GPA Calculator

Interface Description: A button to return to the main page in the left corner. Then a series of labes creating three columns of nine rows, the top row describing each column as "class", "grade", and "credits", and the first row labeling two entry boxes next to each label. A label at the bottom instructs users to leave unused entries blank. There is a button in the bottom left corner to calculate the potential gpa, and this is displayed in the bottom right corner via a label.

Image of Output:

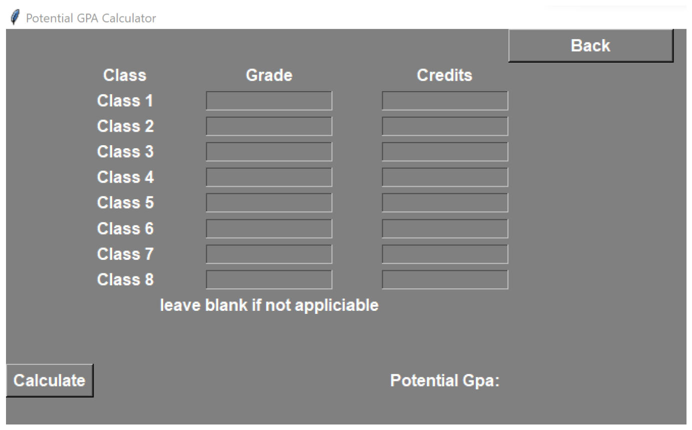


figure 23 (potential gpa calculator screen)

Code Description: This gives users the opportunity to cancel or finalize changes they made to previous semester data.

Code:

```
import tkinter as tk
window=tk.Tk()
window.title('Potential GPA Calculator')
window.geometry('570x350')
window.configure(bg='grey')
param=tk.Label(window,text='leave blank if not appliciable',fg='white',font='Helvetica 12 bold',bg='grey')
param.grid(column=2,row=10)
title1=tk.Label(window,text='Class',fg='white',font='Helvetica 12 bold',bg='grey')
title1.grid(column=1,row=1)
cl1=tk.Label(window,text='Class 1',fg='white',font='Helvetica 12 bold',bg='grey')
cl1.grid(column=1,row=2)
cl2=tk.Label(window,text='Class 2',fg='white',font='Helvetica 12 bold',bg='grey')
cl2.grid(column=1,row=3)
cl3=tk.Label(window,text='Class 3',fg='white',font='Helvetica 12 bold',bg='grey')
```

```
cl3.grid(column=1,row=4)
cl4=tk.Label(window,text='Class 4',fg='white',font='Helvetica 12 bold',bg='grey')
cl4.grid(column=1,row=5)
cl5=tk.Label(window,text='Class 5',fg='white',font='Helvetica 12 bold',bg='grey')
cl5.grid(column=1,row=6)
cl6=tk.Label(window,text='Class 6',fg='white',font='Helvetica 12 bold',bg='grey')
cl6.grid(column=1,row=7)
cl7=tk.Label(window,text='Class 7',fg='white',font='Helvetica 12 bold',bg='grey')
cl7.grid(column=1,row=8)
cl8=tk.Label(window,text='Class 8',fg='white',font='Helvetica 12 bold',bg='grey')
cl8.grid(column=1,row=9)
grade=tk.Label(window,text='Grade',fg='white',font='Helvetica 12 bold',bg='grey')
grade.grid(column=2,row=1)
credit=tk.Label(window,text='Credits',fg='white',font='Helvetica 12 bold',bg='grey')
credit.grid(column=3,row=1)
gpalab=tk.Label(window,text="Potential Gpa:",fg='white',font='Helvetica 12 bold',bg='grey')
gpalab.grid(column=3,row=11,pady=45)
entbox1=tk.Entry(window,bg='grey')
entbox1.grid(row=2,column=2)
entbox2=tk.Entry(window,bg='grey')
entbox2.grid(row=3,column=2)
entbox3=tk.Entry(window,bg='grey')
entbox3.grid(row=4,column=2)
entbox4=tk.Entry(window,bg='grey')
entbox4.grid(row=5,column=2)
entbox5=tk.Entry(window,bg='grey')
entbox5.grid(row=6,column=2)
entbox6=tk.Entry(window,bg='grey')
entbox6.grid(row=7,column=2)
entbox7=tk.Entry(window,bg='grey')
entbox7.grid(row=8,column=2)
entbox8=tk.Entry(window,bg='grey')
entbox8.grid(row=9,column=2)
entbox1c=tk.Entry(window,bg='grey')
entbox1c.grid(row=2,column=3)
entbox2c=tk.Entry(window,bg='grey')
entbox2c.grid(row=3,column=3)
entbox3c=tk.Entry(window,bg='grey')
entbox3c.grid(row=4,column=3)
entbox4c=tk.Entry(window,bg='grey')
entbox4c.grid(row=5,column=3)
entbox5c=tk.Entry(window,bg='grey')
entbox5c.grid(row=6,column=3)
entbox6c=tk.Entry(window,bg='grey')
entbox6c.grid(row=7,column=3)
entbox7c=tk.Entry(window,bg='grey')
entbox7c.grid(row=8,column=3)
entbox8c=tk.Entry(window,bg='grey')
entbox8c.grid(row=9,column=3)
continue1=tk.Button(window,bg='grey',fg='white',text='Calculate',font='Helvetica 12 bold',command=calculate)
continue1.grid(column=0,row=11,pady=45)
backButton = tk.Button(window,text = 'Back',bg = 'grey', fg='white', font='Helvetica 12 bold',padx = 55,command = Back)
backButton.grid(row = 0, column = 12)
window.mainloop()
```

Drop/Withdrawal

<u>Interface Description:</u> A label on top of a button for each class that the user inputted. Then a button to drop or withdraw from a certain class. Then a button to finalize the decision.

<u>Image of Output:</u>

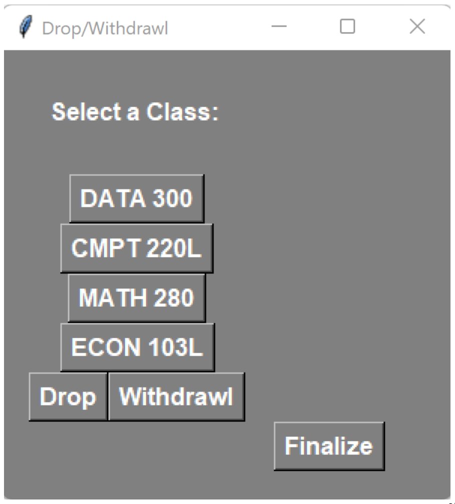


figure 24 (remove class screen)

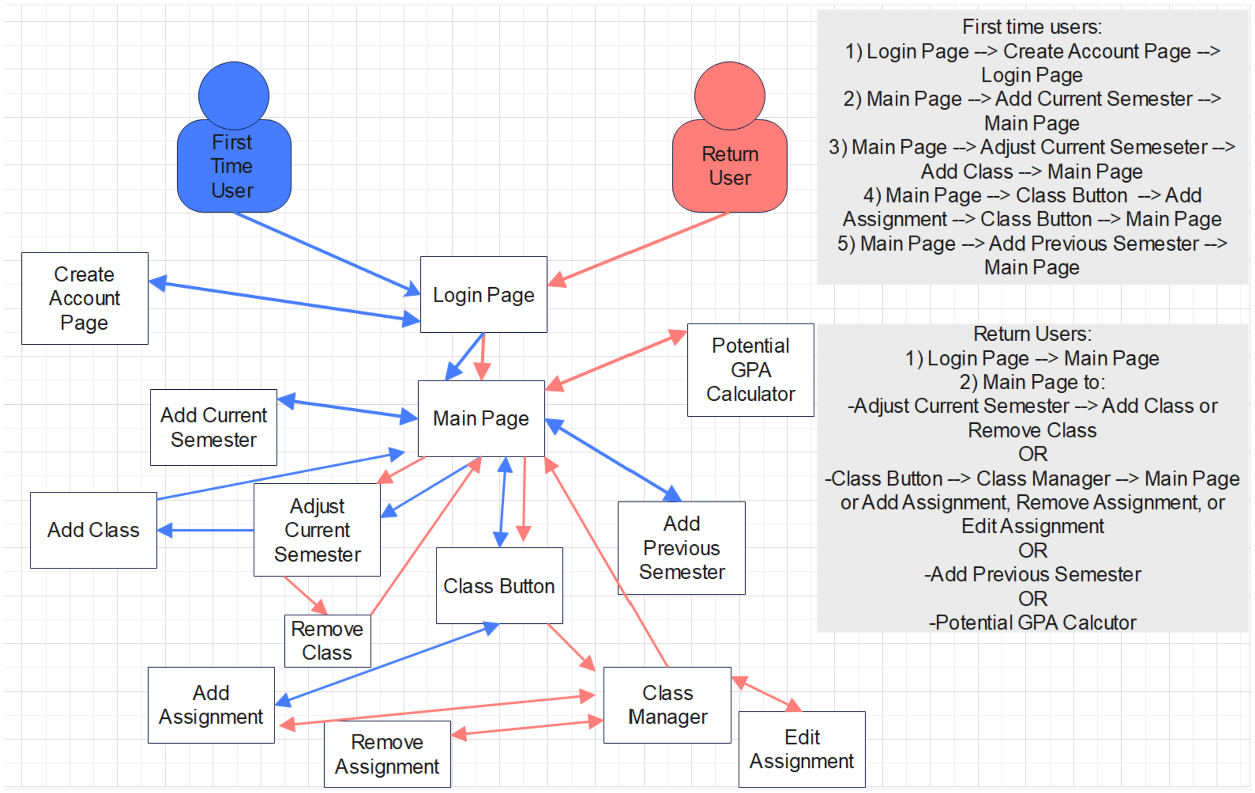
<u>Code Description:</u> This allows users to delete a certain class from the system/csv file.

<u>Code:</u>

```
import tkinter as tk
drop_withdrawl = tk.Tk()
drop_withdrawl.title('Drop/Withdrawl')  # title for window
drop_withdrawl.geometry('300x300')  # length x width
drop_withdrawl.configure(bg = 'grey')  # background color
classSelectionFrame = tk.Frame(drop_withdrawl, bg = 'grey', width = 200, height = 100)
dropWithdrawlFrame = tk.Frame(drop_withdrawl, bg = 'grey' , height = 50, width = 300)
classSelectionFrame.grid(row = 1, column = 5)
dropWithdrawlFrame.grid(row = 2, column = 5)
selectClassLabel = tk.Label(drop_withdrawl,text = 'Select a Class: ',bg = 'grey',  fg = 'white',  font = 'Helvetica 12 bold', padx = 30, pady = 30)
selectClassLabel.grid(row = 0, column = 5)
test_classes = ['DATA 300', 'CMPT 220L', 'MATH 280', 'ECON 103L'] # actual data from backend
count = 0
for c in test_classes:
   tk.Button(classSelectionFrame,text = c,bg = 'grey',fg = 'white',font = 'Helvetica 12 bold').grid(row = count, column = 1)
   count = count+1
btnDrop = tk.Button(dropWithdrawlFrame, text = 'Drop', fg='white', bg='grey', font='Helvetica 12 bold')
btnDrop.grid(row = 0, column = 0)
btnWithdrawl = tk.Button(dropWithdrawlFrame, text = 'Withdrawl', fg='white', bg='grey', font='Helvetica 12 bold')
btnWithdrawl.grid(row = 0, column = 1)
btnApplyChanges = tk.Button(drop_withdrawl, text = 'Finalize', fg='white', bg='grey', font='Helvetica 12 bold')
btnApplyChanges.grid(row = 4, column = 9)
drop_withdrawl.mainloop()
```

Page Connections

Web of User Experience:



First time users:
1) Login Page --> Create Account Page --> Login Page
2) Main Page --> Add Current Semester --> Main Page
3) Main Page --> Adjust Current Semeseter --> Add Class --> Main Page
4) Main Page --> Class Button --> Add Assignment --> Class Button --> Main Page
5) Main Page --> Add Previous Semester --> Main Page

Return Users:
1) Login Page --> Main Page
2) Main Page to:
-Adjust Current Semester --> Add Class or Remove Class
OR
-Class Button --> Class Manager --> Main Page or Add Assignment, Remove Assignment, or Edit Assignment
OR
-Add Previous Semester
OR
-Potential GPA Calcutor

figure 25 (web of user experience screen)

Data Storage

CSV Files and Usage:

CSV File: loginInfo.csv

File Description: A csv file labeled “loginInfo.csv” will be created for every unique account created. It is created in the ‘createAccount’ page and includes the username and password for the account.

CSV File: currentLogin.csv

File Description: A csv file labeled “currentLogin.csv” when the program is first logged into. This stores the current account that is currently logged into the SLMS program.

CSV File: semesterInfo.csv

File Description: A csv file labeled “semesterInfo.csv” will be created for every unique account. It will store the semester info, previous and current. For the previous semester info it will store the year, session, and gpa. For the current semester info it will store the year and session.

CSV File: course.csv

File Description: A csv file labeled “course.csv” will be created for every unique class in every unique account created. It is created in the ‘addClass’ page and stores the course name, credits, and assignment weights. It also stores the assignments inputted through the ‘addAssignment’ page and holds the assignment title, grade and weight.