

# Task Management System

Introduction to Programming  
CMPT 120L



Marist College  
School of Computer Science and Mathematics

Submitted To:  
Dr. Reza Sadeghi  
[reza.sadeghi@marist.edu](mailto:reza.sadeghi@marist.edu)

Fall 2023

## Final Project Progress Report Phase #4

### Team Name

MC Squared

### Team Members

- |                         |  |
|-------------------------|--|
| 1. Andrew Lombardo      | andrew.lombardo1@marist.edu (Team Head)        |
| 2. Benjamin Brandt      | benjamin.brandt1@marist.edu (Team Member)      |
| 3. Nicholas Muratore    | nicholas.muratore1@marist.edu (Team Member)    |
| 4. Christopher Castillo | christopher.castillo1@marist.edu (Team Member) |

### Description of Team Members

#### 1. Andrew Lombardo

My name is Andrew Lombardo, and I am a freshman here at Marist. I have been programming for around nine years, mainly using Java. I've been with this team the whole semester, so I am confident in their abilities. The reason I am a team leader is because I took the initiative by creating the documentation, GitHub repository, and organizing our objectives.

#### 2. Benjamin Brandt

I'm a freshman at Marist studying Computer Science with a concentration in Software Development. This is the group I've worked with for the last few in-class projects, and I thought we worked well together. We all have attributes that complement each other, and I think we produced good work.

#### 3. Nicholas Muratore

I'm a senior at Marist studying Cybersecurity with minors in CS, IS, and IT. I have broad exposure to Python using it for Quantum Computing, BLE device identification using Pyshark, Administration, Web Development, Automation, and now GUI Programming. This is the same group I've been with all year and I'm excited to work with them further on this project.

#### 4. Christopher Castillo

I am a freshman at Marist studying Computer Science with a concentration in Game Design and Programming. My reason for being part of this group is because I have been working with the members for all group projects and I do not see any reason to be in another group.

## Table of Contents

1. GitHub Repository.....	6
2. Project Objective.....	7
3. Graphical User Experience Design.....	9-15
4. Graphical User Interface Design.....	16-56
5. Virtual Environment.....	57
6. Data Storage.....	58
7. Libraries.....	59
8. References.....	60

## Table of Figures

1. Figure 1: Login Page Flowchart.....	8
2. Figure 2: Main Page Flowchart.....	9
3. Figure 3: Add Task Flowchart.....	10
4. Figure 4: Edit Task Flowchart.....	11
5. Figure 5: Remove Task Flowchart.....	12
6. Figure 6: Search Task Flowchart.....	13
7. Figure 7: Calendar Page Flowchart.....	14
8. Figure 8: Navigation Bar Flowchart.....	15
9. Figure 9: Login Layout Drawing.....	16
10. Figure 10: Login Layout .....	16
11. Figure 11: User Layout Drawing.....	18 & 24
12. Figure 12: User Page Layout.....	19
13. Figure 13: Admin User Navigation Bar Layout Drawing.....	22
14. Figure 14: Admin User Navigation Bar Layout.....	22
15. Figure 15: Admin User's Task Management Page Layout.....	25
16. Figure 16: Admin User's User Management Page Layout Drawing.....	28

17. Figure 17: Admin User's User Management Page Layout.....	28
18. Figure 18: Admin User's Admin Management Page Layout Drawing.....	30
19. Figure 19: Admin User's Admin Management Page Layout.....	30
20. Figure 20: Calendar Page Layout Drawing.....	32
21. Figure 21: Calendar Page Layout.....	33
22. Figure 22: Management System Layout.....	37
23. Figure 23: Virtual Environment Video.....	54
24. Figure 24: Data Storage.....	55
25. Figure 25: Data Storage.....	55
26. Figure 26: Data Storage.....	55

## **[1] GitHub Repository Address**

[https://github.com/Levalvus/Final\\_Project--Task\\_Manager--MCSquared/tree/main](https://github.com/Levalvus/Final_Project--Task_Manager--MCSquared/tree/main)

## **[2] Project Objective**

Project Title: Task Management System

Summary: A Task management system (TMS) displays a calendar for the desired week, month, or year. Also, TMS organizes the personal tasks of different users on a specific day. The users should be able to see their calendar data & update them. Your TMS will store the data of different user types in distinct comma-separated value (CSV) files. This system should at least support the following items.

1. The admin user is capable of:
  - a. Having admin user and password for login (a string of at least 8 characters)
  - b. Changing the admin user and admin password
  - c. Adding a normal user to TMS by creating a new username and password. A normal user is not able to define or remove other users.
  - d. Remove users from TMS by removing their username, password, and corresponding recorded data.
2. Each user should be able to:
  - a. Add a task to TMS. The task contains: a title, time, duration, and description
  - b. Remove a task
  - c. Edit a task's details
  - d. Search through TMS based on time, title, or duration and list the results on the screen. For instance, it should be able to list all scheduled work for one day.
3. TMS should be a user-friendly software, such that:
  - a. It shows a welcome page and provides a menu of all functions to the user on all pages
  - b. It illustrates the reports in a tabular form. For instance, it displays a well-organized calendar of every month or year.
  - c. It shows a warning if a user tries to input contact information with a name that exists in the history.
  - d. TMS should provide an exit function and thank the user for using this software.
4. Optional: TMS should protect the user information, such that:
  - a. TMS passwords and the recorded information should be ciphered. In the simplest case, you can use the Caesar cipher methodology. The easiest way to understand the Caesar cipher is to think of cycling the position of the letters. In a Caesar cipher with a shift of 3, A becomes D, B becomes E, C becomes F, etc. When reaching the end of the alphabet it cycles around, so X becomes A, Y becomes B, and Z becomes C.

### [3] Graphical User Experience Design

## Login Page

**Abstract:** The program is opened and the login page pops up. On this page, the user or admin will log on to our Task Management System. When the credentials are entered, the program will check to see if they are a match. If not, then the user will have to re-enter credentials. Otherwise, they will be granted access. If admin credentials are entered it will bring them to the admin page. If user credentials are entered it will bring them to the normal page. Lastly, the exit option will close the program.

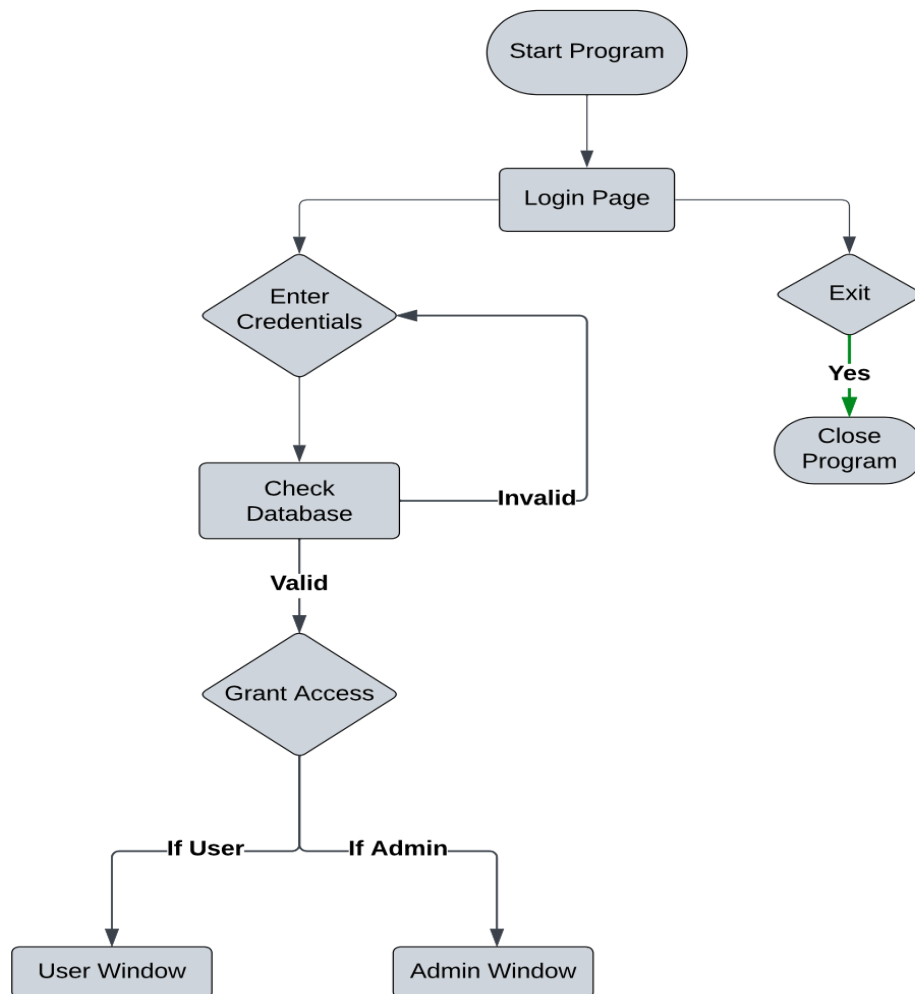


Figure 1: Login Page Flowchart



# Graphical User Experience Design

## *Main User Page*

**Abstract:** After logging in, the main window pops up. In this window, there is an add task in which the user can enter attributes for the task and save it to our system; an edit task, where the user edits an existing task and saves it to our system; a remove task, where the user can select a task created and remove it; and a search task, where the user can search a task by name. Then, our admin page has the option to go to the task page, user management, or admin management. The privileges from that consist of adding a user and by creating its credentials, removing the user by username, and deleting its data. The admin user can also edit their username and password. Lastly, the logout button to go back to the login page.

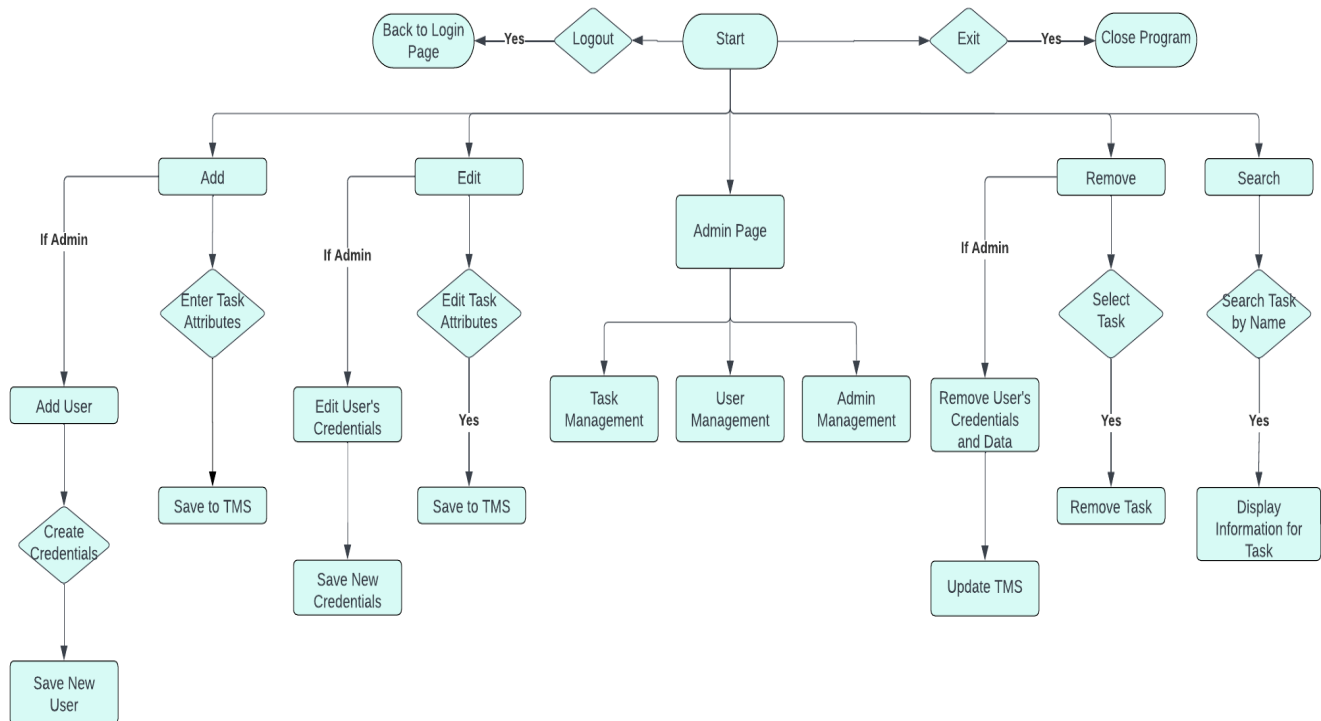


Figure 2: Main Page Flowchart

# Graphical User Experience Design

## *Add Task*

**Abstract:** The add feature allows the user/admin to add a task. Users will be asked to include its time, date, title, duration, and a general description for it to qualify as a valid task. If any of the features are not valid, the user will need to re-enter the item. If it is a valid entry, the task will be saved to the Task Management System. The user can exit the page, which brings them back to the login page.

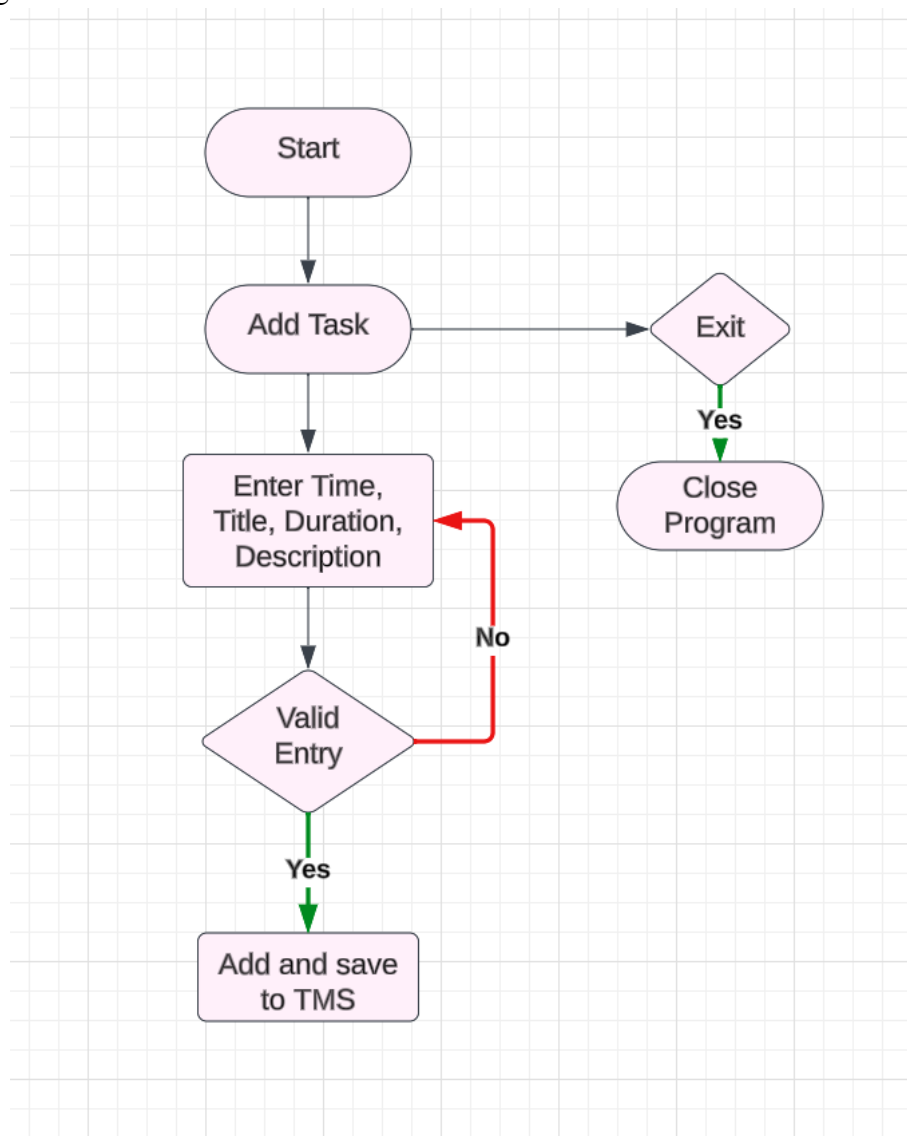


Figure 3: Add Task Flowchart

# Graphical User Experience Design

## *Edit Task*

**Abstract:** The edit feature allows the user to search for any task that is a part of their schedule and change specific information about it. Using the exact title for the task, they can then edit the task's date, start time, description, and duration. The user can leave this page by logging out.

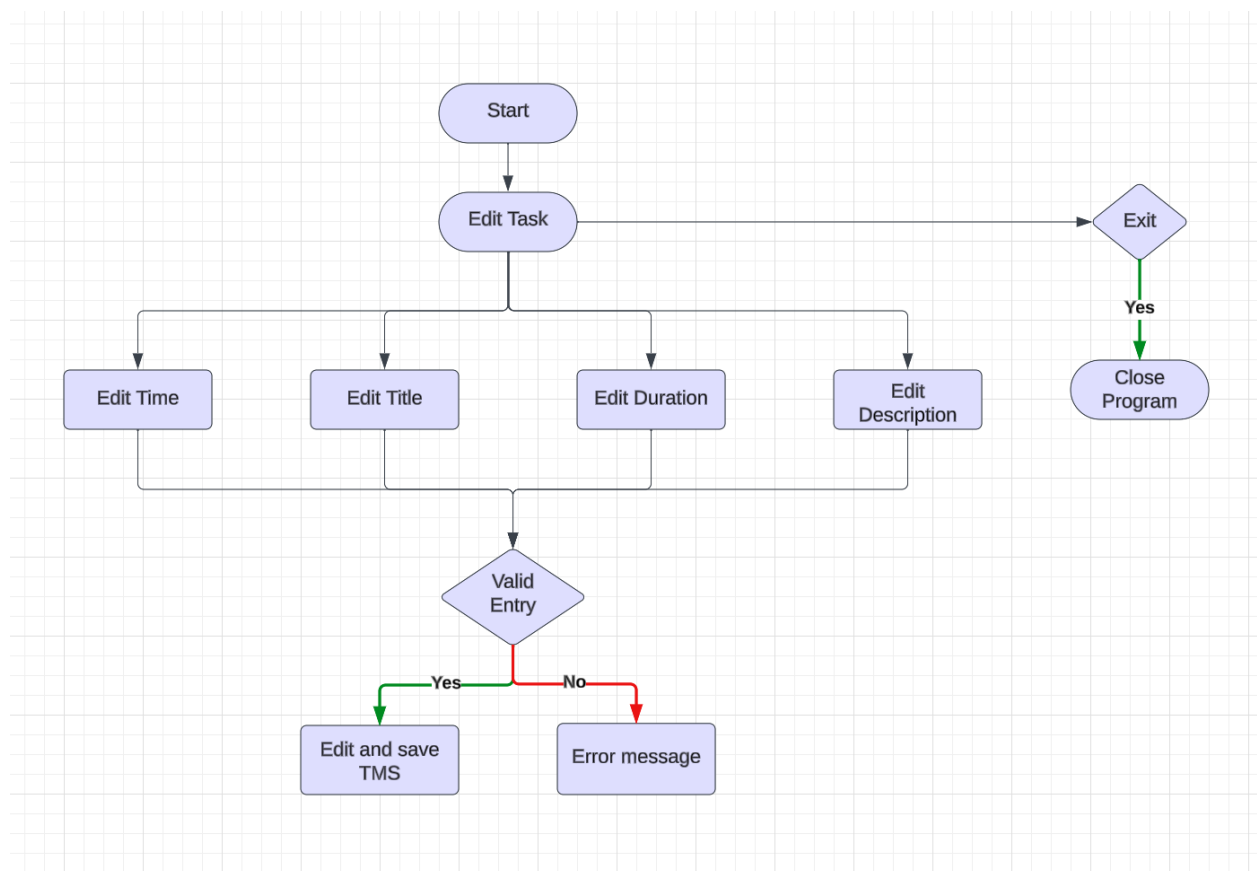


Figure 4: Edit Task Flowchart

# Graphical User Experience Design

## *Remove Task*

**Abstract:** The remove task allows the user to discard a task they do not need or no longer want. They input the information about the given task, then once they hit the remove button, the data will be deleted from the task management system. The user can leave this page by logging out.

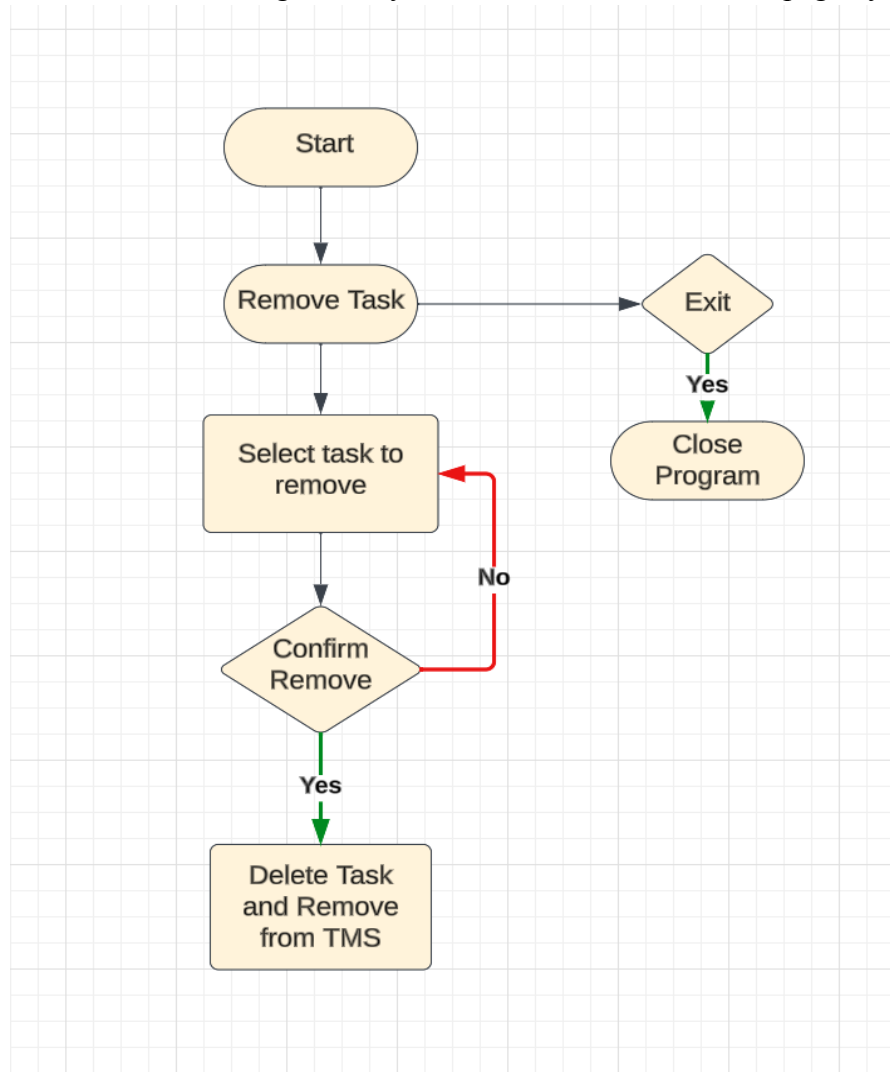


Figure 5: Remove Task Flowchart

# Graphical User Experience Design

## *Search Task*

**Abstract:** The search feature allows the user to search for any task that is a part of their schedule. The user can search through all tasks with a given title, duration, or start time and will be offered a list of all tasks that match those requirements. The user can leave this page by logging out.

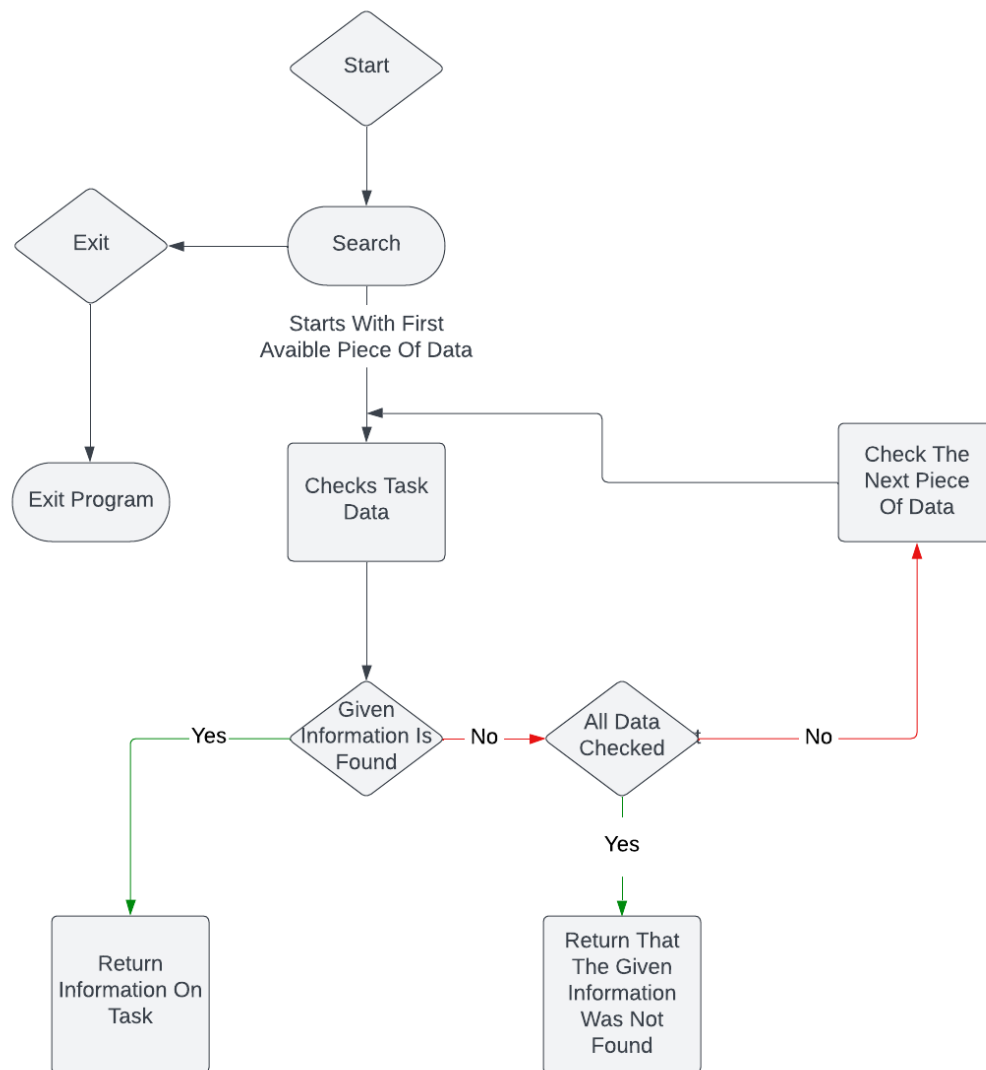


Figure 6: Search Task Flowchart

# Graphical User Experience Design

## *Calendar*

**Abstract:** The calendar is a unique window that takes all tasks from each user's schedule and displays them in an organized window. The user can click on the interactive display, which will then display any tasks for the given day. The user can leave this page either by closing the program or exiting back to the user's respective page.

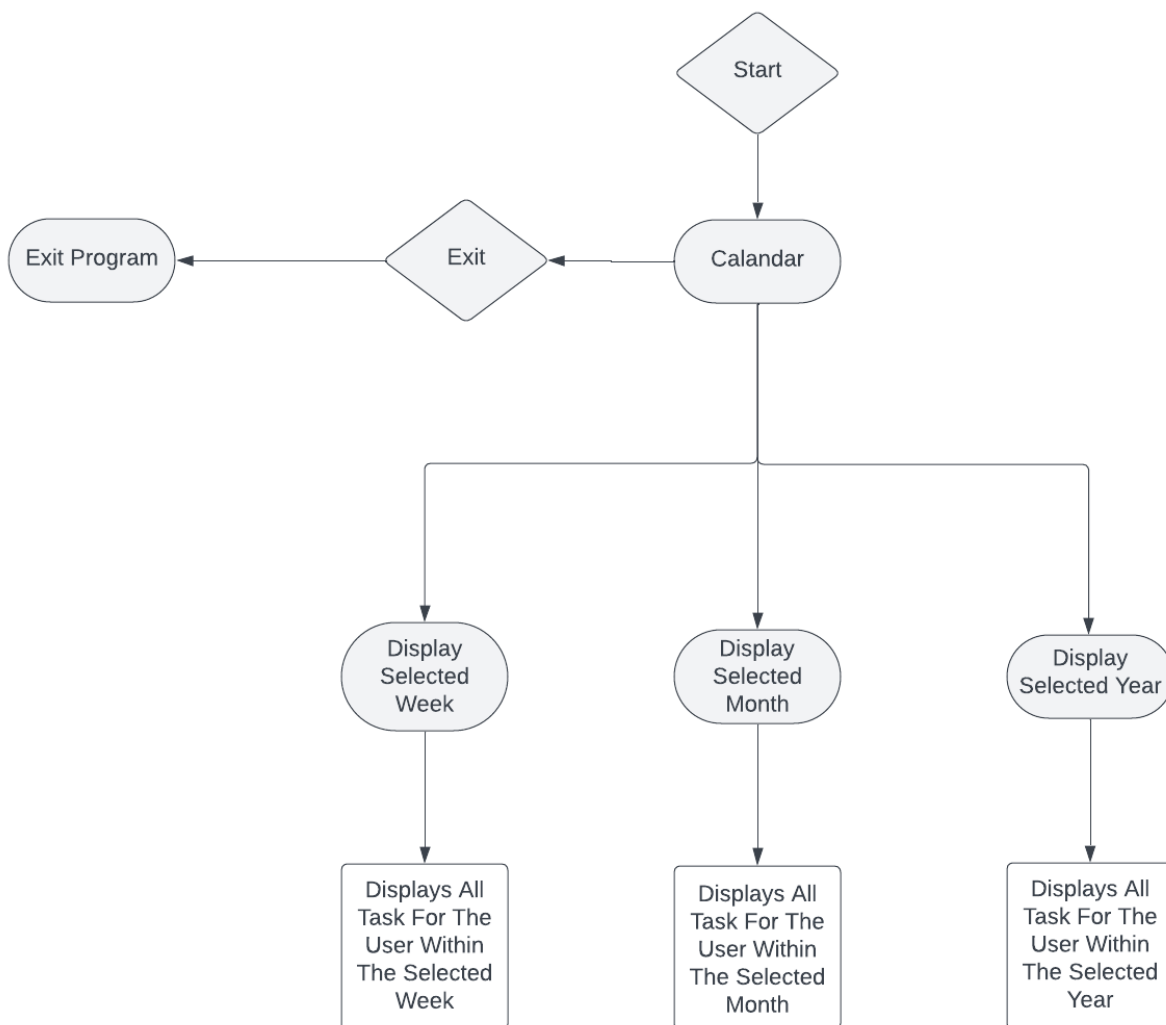


Figure 7: Calendar Flowchart

# Graphical User Experience Design

## *Navigation Bar*

**Abstract:** The navigation bar gives the admin user access to the three actions they can complete: user management, admin management, and task management. Clicking on each of the buttons will allow the user to enter a screen to complete one of the tasks. All of the tasks have been previously defined, hence the yellow note that they all connect to. Finally, the user is allowed to log out of the system at the bottom right of the interface.

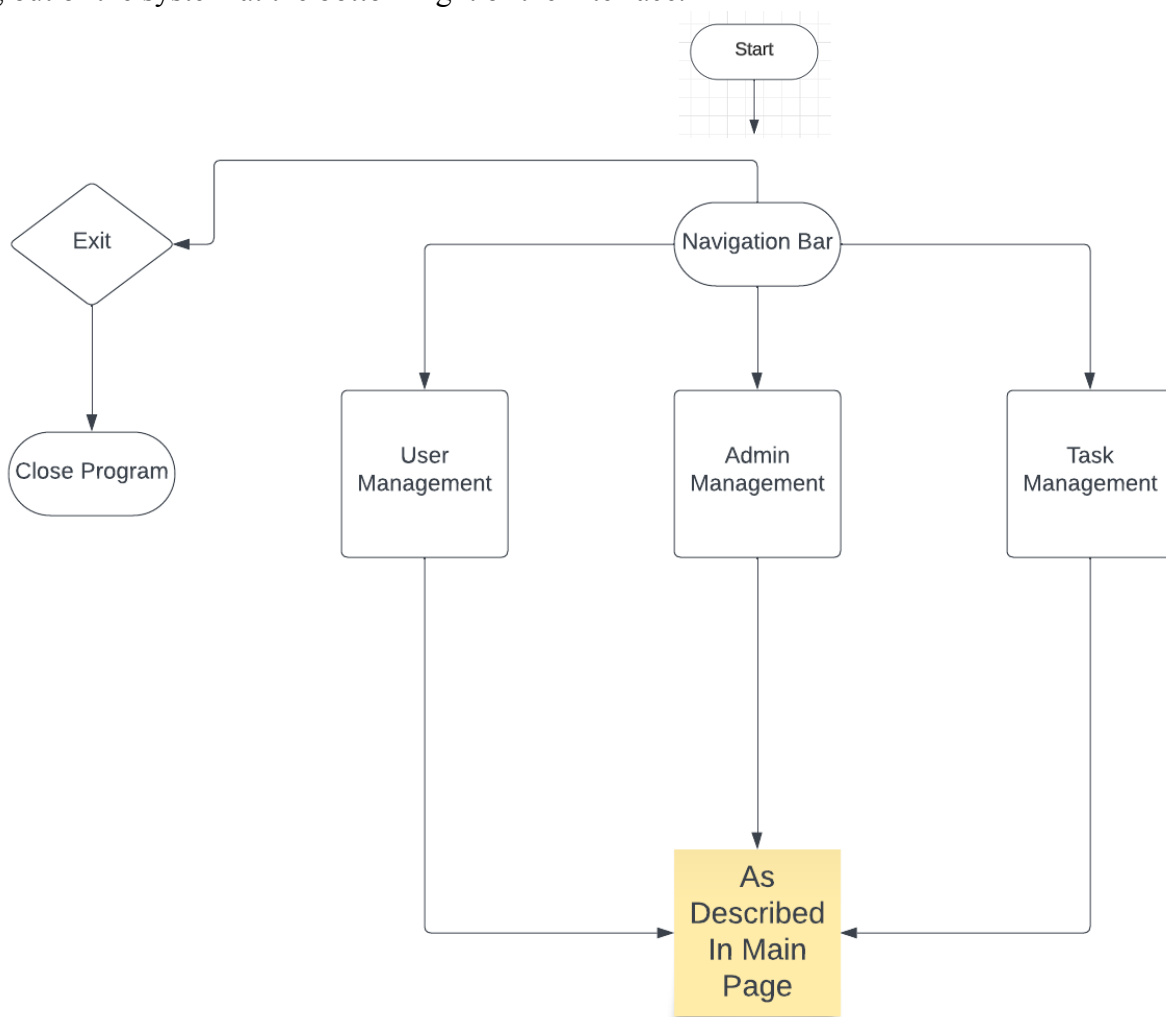


Figure 8: Navigation Bar Flowchart

# Graphical User Experience Design

## *Internal Connection Flow Chart*

**Abstract:** Here, we provide an overview of the entire layout and its pages. Starting the program we have the login page which you can log in or look at the about us section/welcome page. From there you can go to the user page to handle tasks, or the admin page to handle admin operations. Admin may work on user management, admin management, or task management. Both admins and regular users have task capabilities as well as the calendar page to look at tasks on a weekly, monthly, yearly basis and display tasks for those dates.

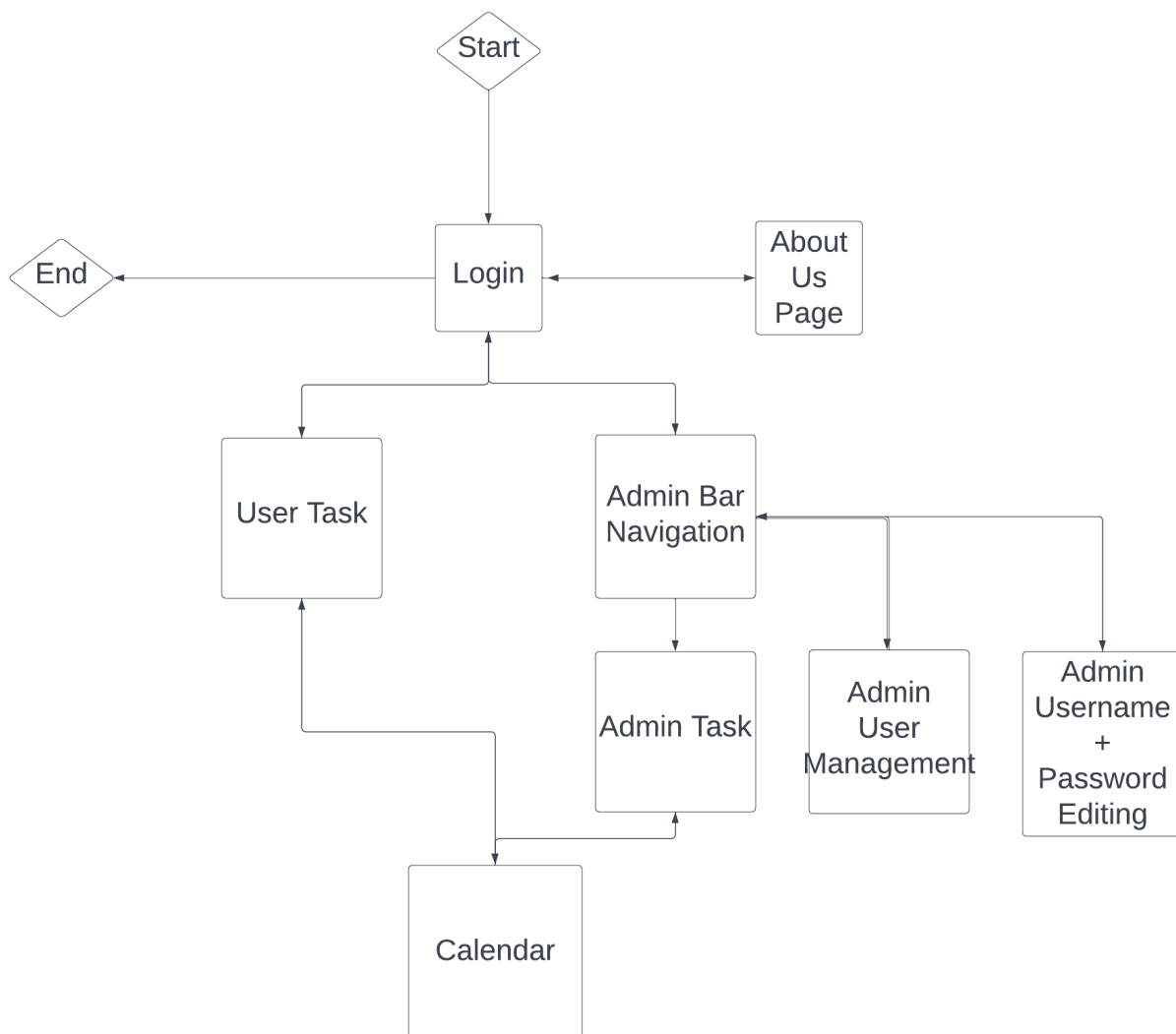


Figure 22: Management System Layout



## [4] Graphical User Interface Design

### *Login Page*

**Abstract:** The login page contains a banner with the name of our project Task Management System. It has two images on each side resembling tasks and management. The word Login is displayed in large letters at the top. Following it, the username label for the user to enter the specified data and its entry bar to physically enter and the same thing for the password label and its entry bar are displayed. Below these, the submit button for entering the credentials is placed which will check the database for authorization. The exit button is displayed in the corner to leave the program.

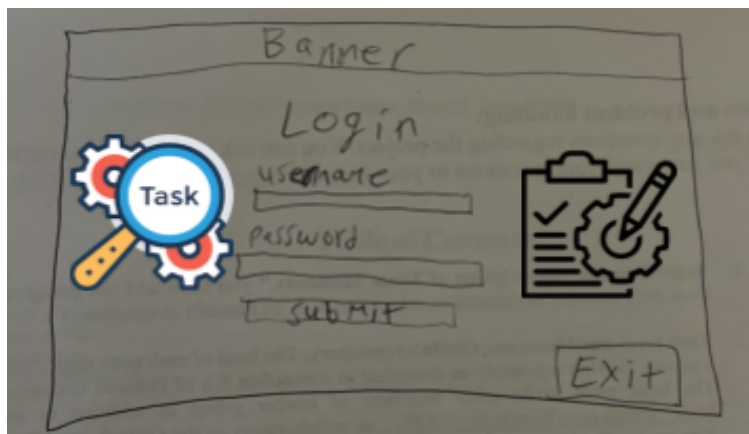


Figure 9: Login Page Layout Drawing

**Implementation:** Below we created the page in Python. The username label and entry bar and the password label with its entry bar are displayed in the center of the screen. Using tkinter's image attribute, we added images through its relative path for all users to display the image on their system. Finally, using the tkinter button attribute, we added an exit button with the command destroy which closes the window.

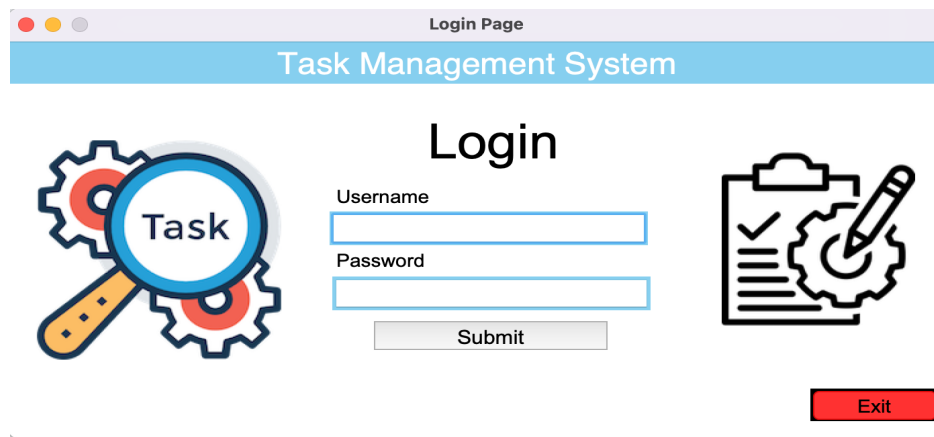


Figure 10: Login Page Layout

# Graphical User Interface Design

## *Login Page Code*

```
#Login Page - TMS
import tkinter as tk
import tkmacosx as tkm
from PIL import ImageTk, Image
window=tk.Tk()
window.title("Login Page")
window.geometry('700x330')
window.resizable(False,False)
window.configure(bg='white')

task_image = ImageTk.PhotoImage(file="/Users/nickmuratore/Downloads/TASK.png")
clipboard_image = ImageTk.PhotoImage(file="/Users/nickmuratore/Downloads/clipboard.png")

task_label = tk.Label(window, image=task_image, bg='white')
task_label.place(x=20, y=80)
clip_label = tk.Label(window, image=clipboard_image, bg='white')
clip_label.place(x=530, y=90)

banner = tk.Label(window, text="Task Management System", bg='#89CFF0', fg='White', font=("Helvetica", 25))
banner.pack(side=tk.TOP, fill=tk.X)

label = tk.Label(text='Login',bg='white',font=('Arial',40))
label.place(x=310, y=55)

userEnt = tk.Entry(window,width = 25,bg='white')
userEnt.configure(highlightbackground='#89CFF0')
userEnt.place(x=240, y=141)

userLabel = tk.Label(text='Username',bg='white',font=('Arial',15))
userLabel.place(x=242,y=116)

passEnt = tk.Entry(window,width = 25,bg='white')
passEnt.configure(highlightbackground='#89CFF0')
passEnt.place(x=242, y=195)

passLabel = tk.Label(text='Password',bg='white',font=('Arial',15))
passLabel.place(x=242,y=169)

submitButton = tk.Button(text='Submit',
                        font=('Arial',16),
                        bg='white',
                        width=16)
submitButton.place(x=270, y=229)
submitButton.configure(highlightbackground='white')

ExitButton = tkm.Button(text='Exit',font=('Arial',15),bg='#FF3131',command=window.destroy)
ExitButton.place(x=598,y=288)
ExitButton.configure(highlightbackground='black')

window.mainloop()
```

# Graphical User Interface Design

## *User Page*

**Abstract:** The user page of the task management system consists of the title, description, date (month, date, year format), duration, and start time for the given task, which are all shown on the left side of the interface. On the right side, the user has access to the add, edit, search, and remove commands, and is also able to access a calendar to view their tasks. Finally, the user is allowed to log out of the system at the bottom right of the interface.

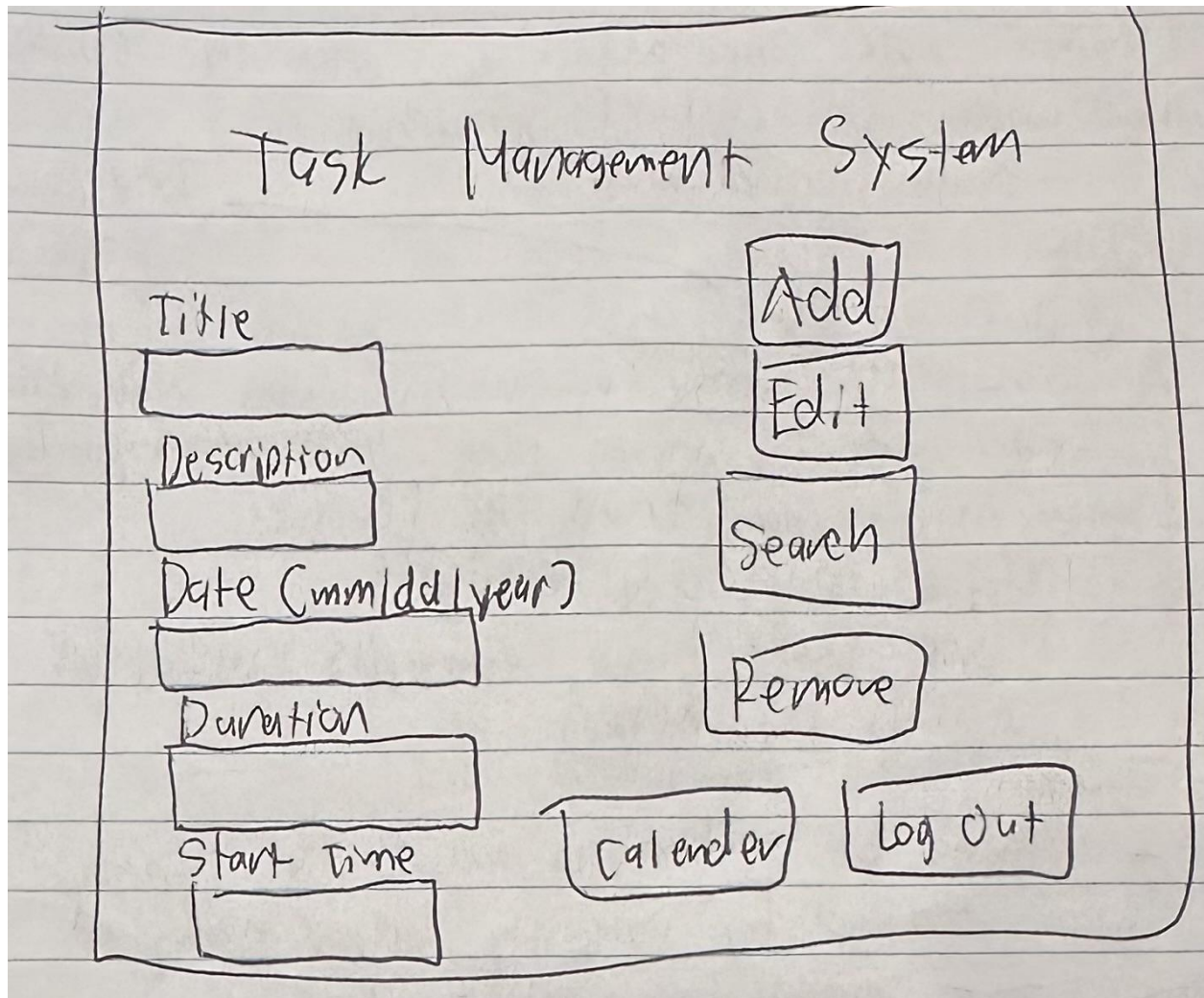


Figure 11: User Page Layout Drawing

**Implementation:** For our User Page, we created entry bars and entry labels for each attribute of the tasks like title, description, date, duration, and start time. On the right side, we have our four functions placed in buttons to call them upon being clicked. For example, the add button to add the task, the edit button to edit the task, the search button to search the task, and then the remove button to remove the task. At the bottom left, we have our calendar button which upon being clicked will bring the user to the calendar page. Finally, a logout button to log the user out destroys the window and displays the login page.



The image shows a user interface for task management. It features a light pink background. At the top, the title "Task Management" is displayed in a large, white, sans-serif font. Below the title, on the left side, there are five input fields, each with a label above it: "Title", "Description", "Date (mm/dd/year)", "Duration", and "Start Time (Military)". The labels are in a bold, black, sans-serif font. To the right of these input fields, there are four buttons stacked vertically: "Add", "Edit", "Search", and "Remove". These buttons are light gray with a thin black border. At the bottom of the interface, there are two more buttons: "Calender" (note the spelling) on the left and "Log Out" on the right. The "Calender" button is blue with white text, and the "Log Out" button is red with white text.

Figure 12: User Page Layout

# Graphical User Interface Design

## *User Page Code*

```

import tkinter as tk
import tkmacosx as tkm
window=tk.Tk()
window.title("User Task Management")
window.geometry('400x450')
window.resizable(False,False)
window.configure(bg="#D58A94")

banner = tk.Label(window, text="Task Management", bg='#D58A94', fg='white', font=("Helvetica", 25))
banner.pack(side=tk.TOP, fill=tk.X)

titleEnt = tk.Entry(window,width = 25)
titleEnt.place(x=10, y=100)

titleLabel = tk.Label(text='Title',font=('Arial',15))
titleLabel.place(x=10,y=70)

desEnt = tk.Entry(window,width = 25)
desEnt.place(x = 10, y=160)

desLabel = tk.Label(text='Description',font=('Arial',15))
desLabel.place(x=10,y=130)

dateEnt = tk.Entry(window,width = 25)
dateEnt.place(x=10, y=220)

dateLabel = tk.Label(text='Date (mm/dd/year)',font=('Arial',15))
dateLabel.place(x=10,y=190)

durEnt = tk.Entry(window,width = 25)
durEnt.place(x = 10, y=280)

durLabel = tk.Label(text='Duration',font=('Arial',15))
durLabel.place(x=10,y=250)

startLabel = tk.Label(text='Start Time (Military)',font=('Arial',15))
startLabel.place(x=10,y=310)

startEnt = tk.Entry(window,width = 25)
startEnt.place(x = 10, y=340)

addButton = tk.Button(text='Add',
                      font=('Arial',12),
                      width=16)
addButton.place(x=230, y=120)

```

```
editButton = tk.Button(text='Edit',
                        font=('Arial',12),
                        width=16)
editButton.place(x=230, y=170)

searchButton = tk.Button(text='Search',
                          font=('Arial',12),
                          width=16)
searchButton.place(x=230, y=220)

removeButton = tk.Button(text='Remove',
                          font=('Arial',12),
                          width=16)
removeButton.place(x=230, y=270)

ExitButton = tk.Button(text='Log Out',font=('Arial',15),bg='red',command=window.destroy)
ExitButton.place(x=225,y=390)

CalenderButton = tk.Button(text='Calender',font=('Arial',15),bg='blue',command=window.destroy)
CalenderButton.place(x=50,y=390)

window.mainloop()
```

## Graphical User Interface Design

### *Admin User Navigation Bar Page*

**Abstract:** The admin user navigation bar consists of everything from the normal user page interface but with a few more options. The admin has access to the management page, where they can add a task's title, description, date (month, day, year format), duration, and start time, as well as all the user commands of add, edit, search, and remove. The admin user can also access the admin management section, where they can change their user and password (must be at least 8 characters), remove users from the task management system, and add new users.

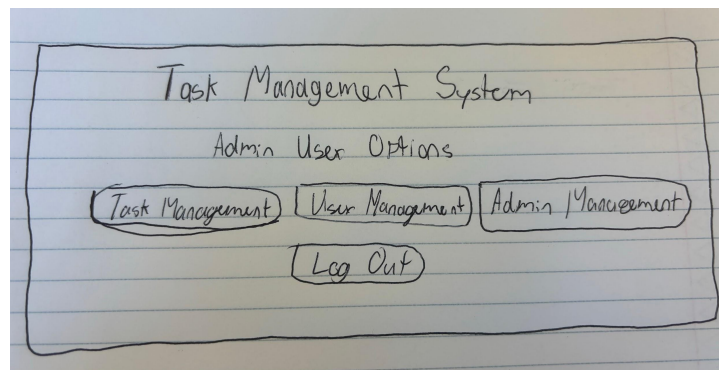


Figure 13: Admin User Navigation Bar Layout Drawing

**Implementation:** This is our administrator page in which we have three buttons. A task management button which upon clicking will bring the user to the task management page for adding, editing, searching, and removing tasks. A user management button will bring the user to the user management page to create and remove users. An admin management button that allows the admin to edit their credentials, and lastly a logout button which upon clicking brings you to the login page and logs the user out.

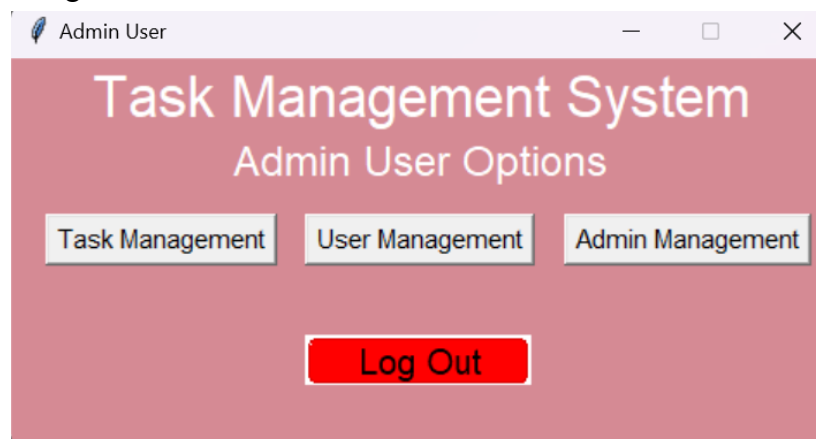


Figure 14: Admin User Navigation Bar Page



## Graphical User Interface Design

### *Admin User Navigation Bar Code*

```
#Admin User Nav Bar - TMS
import tkinter as tk
import tkmacosx as tkm
window=tk.Tk()
window.title("Admin User")
window.geometry('475x225')
window.resizable(False,False)
window.configure(bg="#D58A94")

banner = tk.Label(window, text="Task Management System", bg='#D58A94', fg='white',
font=("Helvetica", 25))
banner.pack(side=tk.TOP, fill=tk.X)

sub_banner = tk.Label(window, text="Admin User Options", bg='#D58A94', fg='white',
font=("Helvetica", 18))
sub_banner.pack(side=tk.TOP, fill=tk.X)

Task_Btn = tk.Button(text='Task Management',font=('Arial',11))
Task_Btn.place(x=20, y=90)

User_Btn = tk.Button(text='User Management',font=('Arial',11))
User_Btn.place(x=170, y=90)

Admin_Btn = tk.Button(text='Admin Management',font=('Arial',11))
Admin_Btn.place(x=320, y=90)

ExitButton = tkm.Button(text='Log Out',font=('Arial',15),bg='red',command=window.destroy)
ExitButton.place(x=170,y=160)

window.mainloop()
```



## Graphical User Interface Design

### *Admin User's Task Management Page*

**Abstract:** The user page of the Task Management System consists of the title, description, date (month, date year format), duration, and the start time for the given task, which are all shown on the left side of the interface. On the right side, the user has access to the add, edit, search, and remove commands, and is also able to access a calendar to view their tasks. Finally, the user is allowed to log out of the system at the bottom right of the interface.

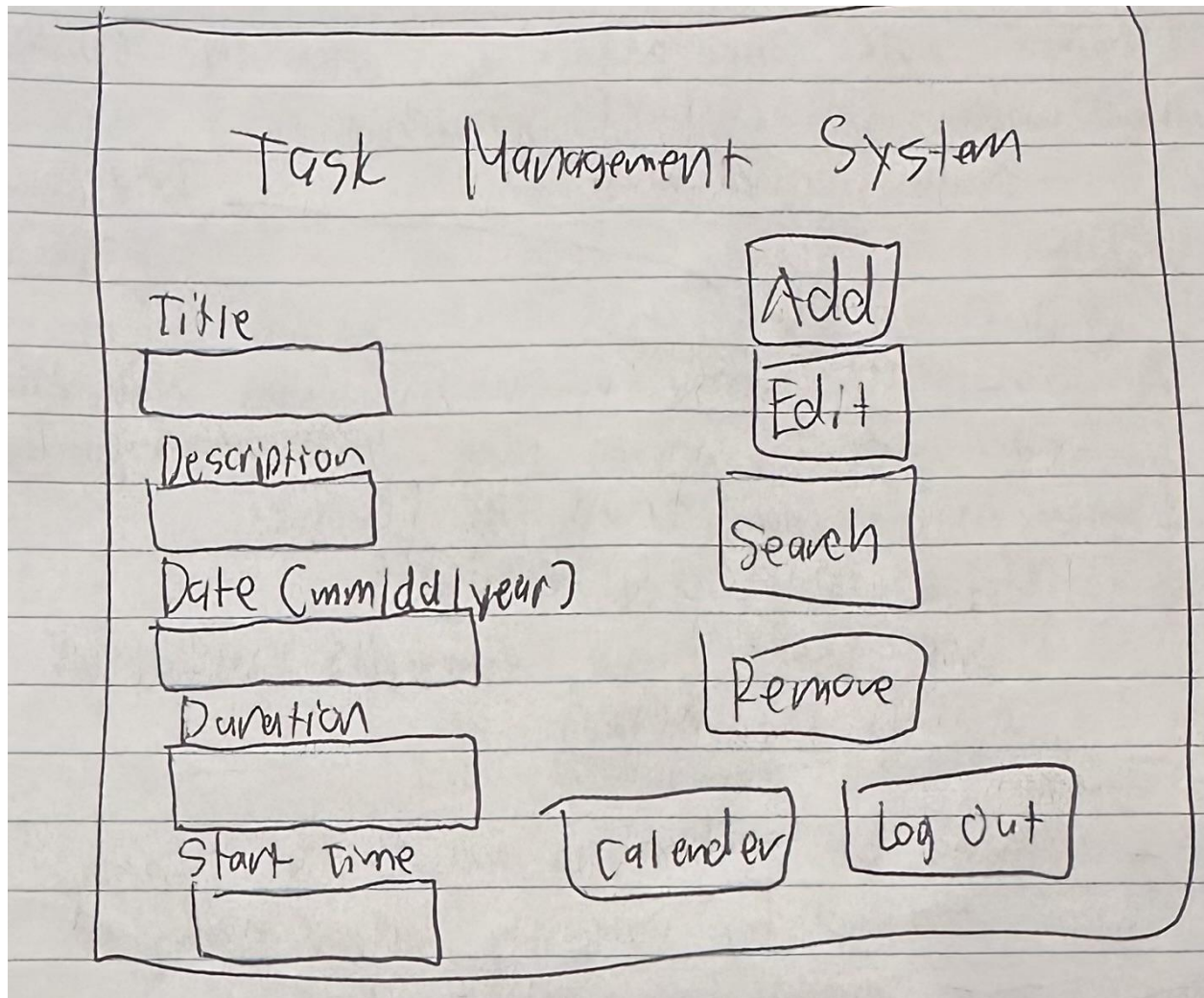


Figure 11: Admin User's Task Management Layout Drawing

**Implementation:** For our Admin Page, similarly, we created entry bars and entry labels for each attribute of the tasks like title, description, date, duration, and start time. On the right side, we have our four functions placed in buttons to call the function upon being clicked. For example, the Add button to add the task, the Edit button to edit the task, the Search button to search the task, and the remove button to remove the task. At the bottom left, we have our calendar button which upon being clicked will bring the admin to the calendar page which displays a calendar. Finally, a logout button to log the admin out simply destroys the window and displays the login page.



The image shows a web form titled "Task Management" with a light red background. On the left side, there are five input fields with labels: "Title", "Description", "Date (mm/dd/year)", "Duration", and "Start Time (Military)". Each label is in a white box above its corresponding input field. On the right side, there are four buttons: "Add", "Edit", "Search", and "Remove", each in a light gray box. At the bottom left, there is a blue button labeled "Calender" (note the spelling). At the bottom right, there is a red button labeled "Log Out".

*Figure 15: Admin User's Task Management Page Layout*

## Graphical User Interface Design

### *Admin User's Task Management Page Code*

```
import tkinter as tk
import tkmacosx as tkm
window=tk.Tk()
window.title("Admin User Task Management")
window.geometry('400x450')
window.resizable(False,False)
window.configure(bg="#D58A94")

banner = tk.Label(window, text="Task Management", bg='#D58A94', fg='white',
font=("Helvetica", 25))
banner.pack(side=tk.TOP, fill=tk.X)

titleEnt = tk.Entry(window,width = 25)
titleEnt.place(x=10, y=100)

titleLabel = tk.Label(text='Title',font=('Arial',15))
titleLabel.place(x=10,y=70)

desEnt = tk.Entry(window,width = 25)
desEnt.place(x = 10, y=160)

desLabel = tk.Label(text='Description',font=('Arial',15))
desLabel.place(x=10,y=130)

dateEnt = tk.Entry(window,width = 25)
dateEnt.place(x=10, y=220)

dateLabel = tk.Label(text='Date (mm/dd/year)',font=('Arial',15))
dateLabel.place(x=10,y=190)

durEnt = tk.Entry(window,width = 25)
durEnt.place(x = 10, y=280)

durLabel = tk.Label(text='Duration',font=('Arial',15))
durLabel.place(x=10,y=250)

startLabel = tk.Label(text='Start Time (Military)',font=('Arial',15))
startLabel.place(x=10,y=310)
```

```
startEnt = tk.Entry(window,width = 25)
startEnt.place(x = 10, y=340)

addButton = tk.Button(text='Add',
                      font=('Arial',12),
                      width=16)
addButton.place(x=230, y=120)

editButton = tk.Button(text='Edit',
                      font=('Arial',12),
                      width=16)
editButton.place(x=230, y=170)

searchButton = tk.Button(text='Search',
                        font=('Arial',12),
                        width=16)
searchButton.place(x=230, y=220)

removeButton = tk.Button(text='Remove',
                        font=('Arial',12),
                        width=16)
removeButton.place(x=230, y=270)

ExitButton = tk.Button(text='Log Out',font=('Arial',15),bg='red',command=window.destroy)
ExitButton.place(x=225,y=390)

CalenderButton =
tk.Button(text='Calender',font=('Arial',15),bg='blue',command=window.destroy)
CalenderButton.place(x=50,y=390)

window.mainloop()
```

## Graphical User Interface Design

### *Admin User's User Management Page*

**Abstract:** The admin's user management page is where the admin can access their specific abilities that a normal user cannot. On this screen, an admin can add or remove a user from the system. As well as, have the option to log out.

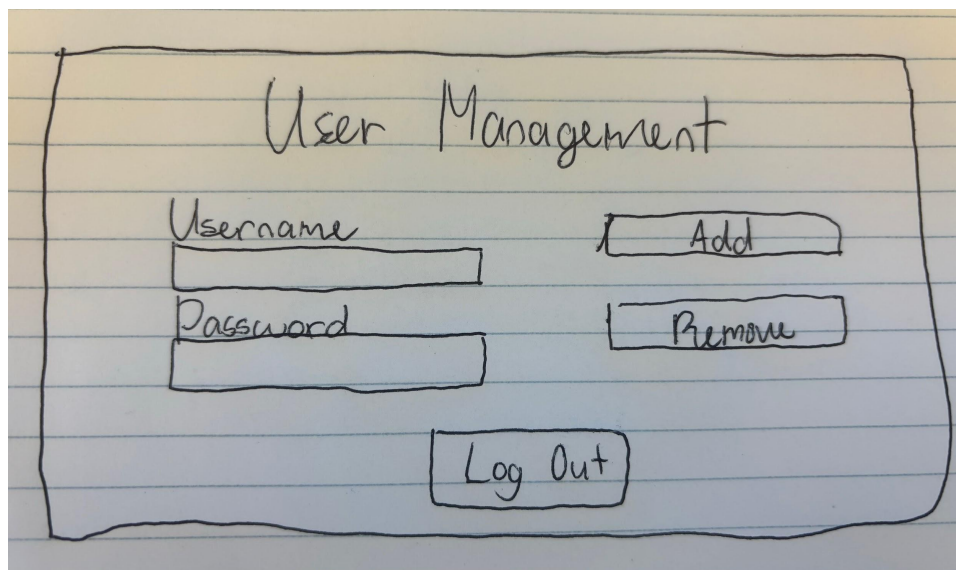
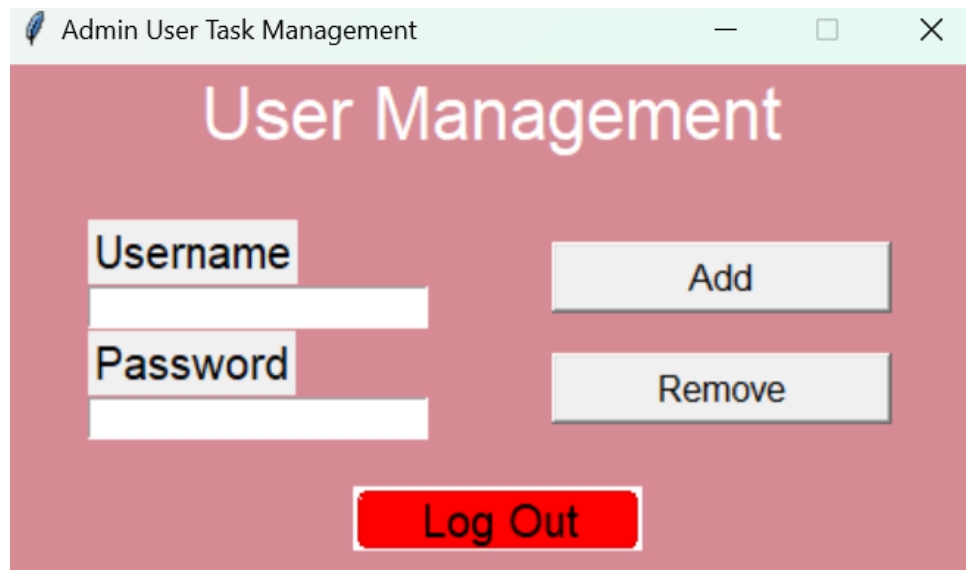


Figure 16: Admin User's User Management Page Layout Drawing

**Implementation:** For our user management page, on the left, we have a username label and entry bar, as well as a password label and entry bar. Then, on the right, we have an add button which upon clicking adds the user to our system, and a remove button which removes the user from our system. Finally, there is a logout button to log the admin out which destroys the window and displays the login page.



*Figure 17: Admin User's User Management Page Layout*

## Graphical User Interface Design

### *Admin User's User Management Page Code*

```
#Admin User Page - TMS
import tkinter as tk
import tkmacosx as tkm
window=tk.Tk()
window.title("Admin User Task Management")
window.geometry('445x250')
window.resizable(False,False)
window.configure(bg="#D58A94")

banner = tk.Label(window, text="User Management", bg='#D58A94', fg='white', font=("Helvetica", 25))
banner.pack(side=tk.TOP, fill=tk.X)

addUserEnt = tk.Entry(window,width = 25)
addUserEnt.place(x=40, y=100)

addUserLabel = tk.Label(text='Username',font=('Arial',15))
addUserLabel.place(x=40,y=70)

removeUserEnt = tk.Entry(window,width = 25)
removeUserEnt.place(x = 40, y=150)

removeUserLabel = tk.Label(text='Password',font=('Arial',15))
removeUserLabel.place(x=40,y=120)

addUserButton = tk.Button(text='Add',
                           font=('Arial',12),
                           width=16)
addUserButton.place(x=250, y=80)

removeUserButton = tk.Button(text='Remove',
                              font=('Arial',12),
                              width=16)
removeUserButton.place(x=250, y=130)

ExitButton = tkm.Button(text='Log Out',font=('Arial',15),bg='red',command=window.destroy)
ExitButton.place(x=160,y=190)

window.mainloop()
```

## Graphical User Interface Design

### *Admin User's Admin Management Page*

**Abstract:** This is the page specific to the admin and the admin only. This is where they can manage the Task Management System. They can use this page to change their name and password. The admin user can leave this page using the log-out option.

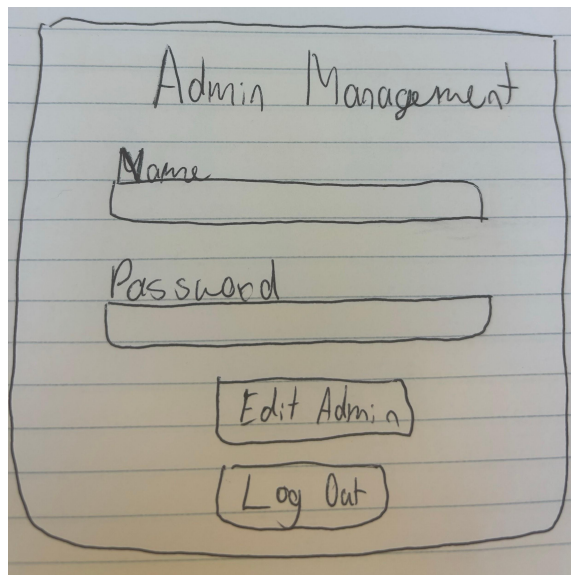
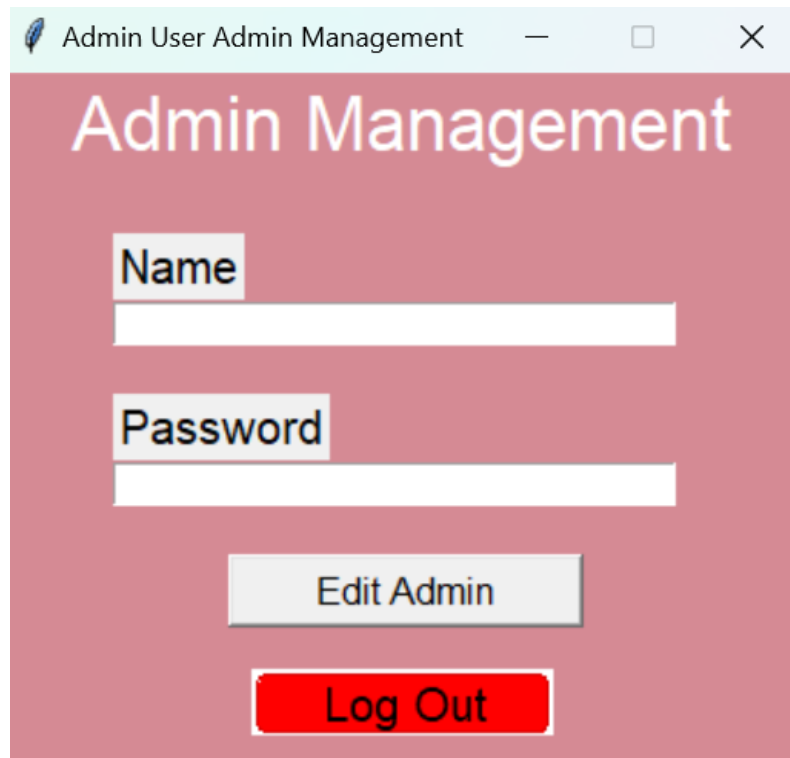


Figure 18: Admin User's Admin Management Page Layout Drawing



**Implementation:** For our admin management page, similarly, on the left we have a username label and entry bar, a password label, and an entry bar. Then, on the right, we have an add button which upon click adds the user to our system, and a remove button which upon click removes the user from our system. Finally, a logout button to log the admin out by destroys the window and displays the login page.

A screenshot of a web application window titled "Admin User Admin Management". The window has a light blue header bar with a feather icon, a minus sign, a square icon, and a close button. The main content area has a pink background. At the top, the text "Admin Management" is displayed in a large, white, serif font. Below this, there are two input fields: "Name" and "Password", each with a label above it. The "Name" label is in a white box, and the "Password" label is in a white box. Below the input fields, there are two buttons: "Edit Admin" and "Log Out". The "Edit Admin" button is light gray, and the "Log Out" button is red with white text.

*Figure 19: Admin User's Admin Management Page*

## Graphical User Interface Design

### *Admin User's Admin Management Page Code*

```
import tkinter as tk
import tkmacosx as tkm
window=tk.Tk()
window.title("Admin User Admin Management")
window.geometry('350x330')
window.resizable(False,False)
window.configure(bg="#D58A94")

banner = tk.Label(window, text="Admin Management", bg='#D58A94', fg='white',
font=("Helvetica", 25))
banner.pack(side=tk.TOP, fill=tk.X)

nameEnt = tk.Entry(window,width = 40)
nameEnt.place(x=50, y= 100)

nameLabel = tk.Label(text='Name',font=('Arial',15))
nameLabel.place(x=50,y=70)

passwordEnt = tk.Entry(window,width = 40)
passwordEnt.place(x=50, y= 170)

passwordLabel = tk.Label(text='Password',font=('Arial',15))
passwordLabel.place(x=50,y=140)

editAdminButton = tk.Button(text='Edit Admin',
                             font=('Arial',12),
                             width=16)
editAdminButton.place(x=100, y=210)

ExitButton = tkm.Button(text='Log Out',font=('Arial',15),bg='red',command=window.destroy)
ExitButton.place(x=110,y=260)

window.mainloop()
```

# Graphical User Interface Design

## *Calendar Page*

**Abstract:** The calendar page showcases the various tasks through an interactive calendar. The user can click on every day of the calendar which was made through the tkcalendar module. After clicking on a specific day, the user can then hit a button that will display the tasks set for that day. The user can also leave the page, which will load the correct task management page depending on if the current user is the admin or just a generic user.

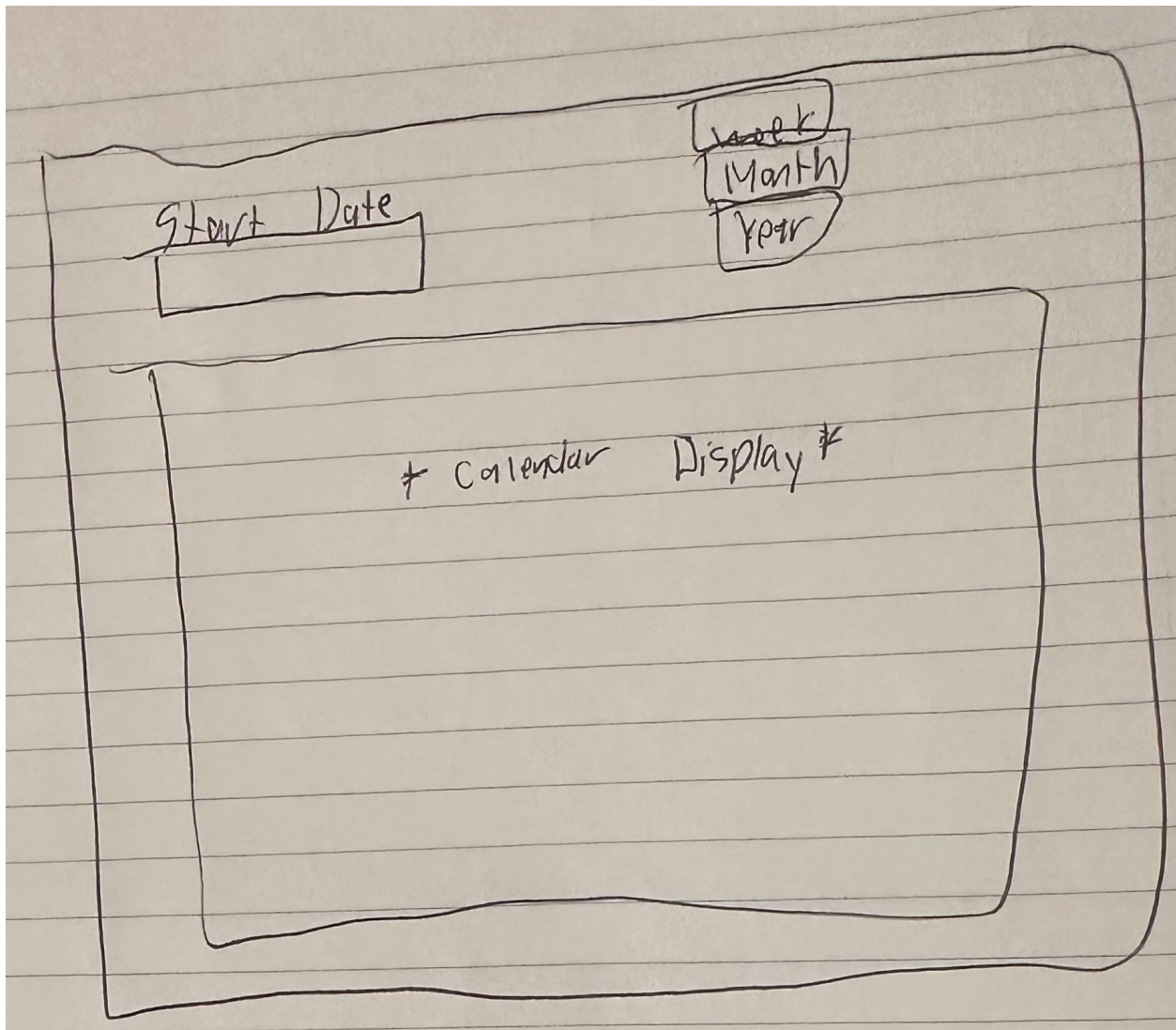


Figure 20: Calendar Page Layout Drawing

**Implementation:** For our calendar page, we have a date label with the proper format for date entry. Using tkinter's calendar function, we implemented a visual calendar for people to see days, weeks, and months in advance. The user may click on the any of the dates, followed by the tasks button in order to show any task they have on that date.

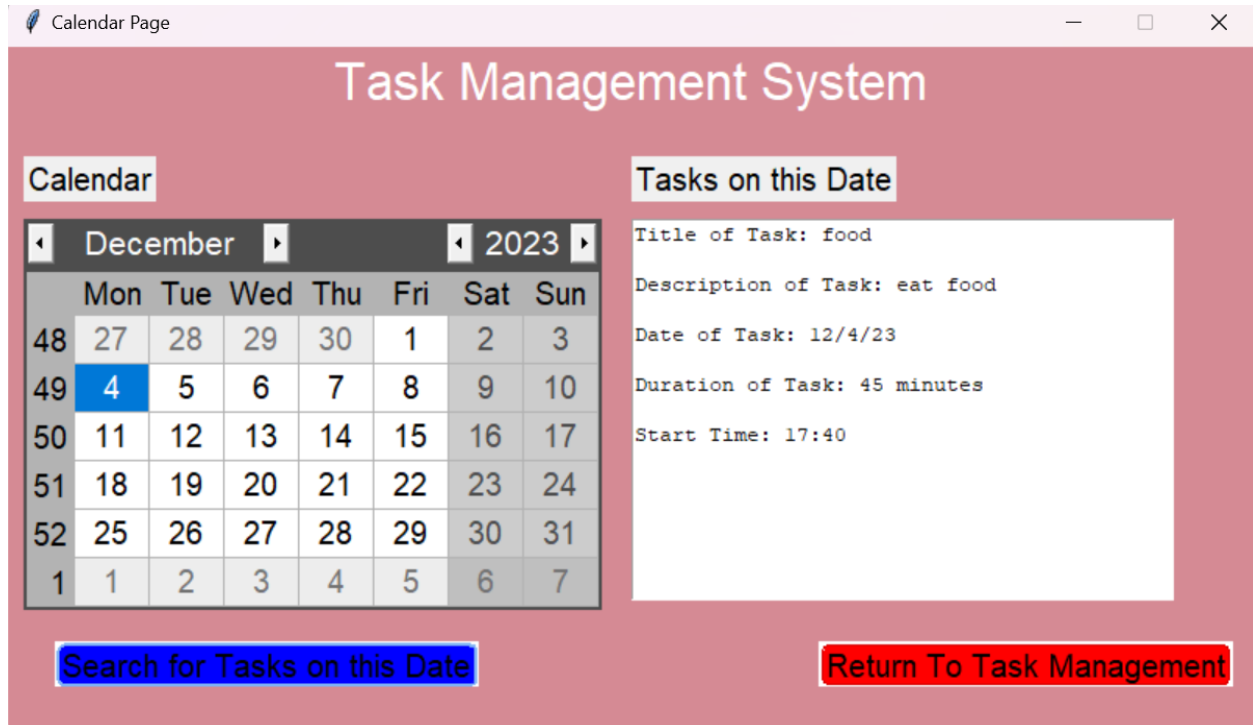


Figure 21: Calendar Page Layout

# Graphical User Interface Design

## *Calendar Page Code*

```
#Calendar Page - TMS
import tkinter as tk
import tkmacosx as tkm
window=tk.Tk()
window.title("Calendar Page")
window.geometry('800x600')
window.resizable(False,False)
window.configure(bg="#D58A94")

banner = tk.Label(window, text="Task Management System", bg='#D58A94', fg='white', font=("Helvetica", 25))
banner.pack(side=tk.TOP, fill=tk.X)

dateEnt = tk.Entry(window,width = 25)
dateEnt.place(x = 10, y=80)

dateLabel = tk.Label(text='Date (mm/dd/year)',font=('Arial',15))
dateLabel.place(x=10,y=50)

addButton = tk.Button(text='Week',
                      font=('Arial',12),
                      width=16)
addButton.place(x= 200, y=50)

editButton = tk.Button(text='Month',
                      font=('Arial',12),
                      width=16)
editButton.place(x=400, y=50)

searchButton = tk.Button(text='Year',
                        font=('Arial',12),
                        width=16)
searchButton.place(x=600, y=50)

displayBox = tk.Label(window, text = 'Calendar Display Will Go Here')
displayBox.place(x = 0, y = 110)

ExitButton = tkm.Button(text='Close Window',font=('Arial',15),bg='red',command=window.destroy)
ExitButton.place(x=650,y=550)

window.mainloop()
```

# Graphical User Interface Design

## *Task Management System Code*

```
#Libraries
import tkinter as tk
import tkmacosx as tkm
from PIL import ImageTk, Image
import tkcalendar as tkc
import os
import csv

#Creating the CSV files
os.chdir('C:\\Users\\chris\\OneDrive - Marist College\\Desktop\\Intro to Programming\\Final
Project Phase 03\\TMS Phase 03 CSV')

#Creating the variables of admin username and admin password to compare for later in the
program

passwordFile = 'Passwords.csv'
with open(passwordFile, 'a') as file:
    field = ['Username', 'Password']
    csvwriter = csv.writer(file)
    csvwriter.writerow(field)
file.close()

#User Passwords
admin_passwordFile = 'Admin_Passwords.csv'
with open(admin_passwordFile, 'a') as file:
    field = ['Username', 'Password']
    csvwriter = csv.writer(file)
    csvwriter.writerow(field)
file.close()

#Task
taskFile = 'Task.csv'
with open(taskFile, 'a') as file:
    field = ['User', 'Title', 'Description', 'Date', 'Start Time', 'Duration']
    csvwriter = csv.writer(file)
    csvwriter.writerow(field)
file.close()
```

```

global track_username
track_username = ""

global track_password
track_password = ""

#Page Functions
def loadLogIn():
    #Login Page - TMS
    window1=tk.Tk()
    window1.title("Login Page")
    window1.geometry('800x340')
    window1.resizable(False,False)
    window1.configure(bg='#D58A94')

    #task_image = ImageTk.PhotoImage(file="/Users/nickmuratore/Downloads/TASK.png")
    #clipboard_image =
ImageTk.PhotoImage(file="/Users/nickmuratore/Downloads/clipboard.png")

    #task_label = tk.Label(window1, image=task_image, bg='white')
    #task_label.place(x=20, y=80)

    #clip_label = tk.Label(window1, image=clipboard_image, bg='white')
    #clip_label.place(x=530, y=90)

    nick_image = tk.PhotoImage(file="CLIPBOARD.png")
    nick_picture = tk.Label(window1, image=nick_image)
    nick_picture.place(x=65,y=110)
    andrew_image = tk.PhotoImage(file="TASK.png")
    andrew_picture = tk.Label(window1, image=andrew_image)
    andrew_picture.place(x=580,y=85)

    banner = tk.Label(window1, text="Task Management System", bg='#D58A94', fg='white',
font=("Helvetica", 25))
    banner.pack(side=tk.TOP, fill=tk.X)

    label = tk.Label(text='Login Page', bg='#D58A94', fg= 'white', font=('Arial',30))
    label.pack(side=tk.TOP, fill=tk.X)

    userEnt = tk.Entry(window1,width = 35,bg='white')
    userEnt.configure(highlightbackground='#89CFF0')
    userEnt.place(x=300, y=148)

```

```
userLabel = tk.Label(text='Username',bg='white',font=('Arial',15))
userLabel.place(x=300,y=116)
```

```
passEnt = tk.Entry(window1,width = 35,bg='white', show='*')
passEnt.configure(highlightbackground='#89CFF0')
passEnt.place(x=300, y=210)
```

```
passLabel = tk.Label(text='Password',bg='white',font=('Arial',15))
passLabel.place(x=300,y=179)
```

```
def user_search():
    u = str(userEnt.get())
    p = str(passEnt.get())
    with open('Admin_Passwords.csv', 'r') as file:
        AdminContent = csv.reader(file)
        for lines in AdminContent:
            try:
                if lines[0] == u and lines[1] == p:
                    return 1
            except IndexError:
                None
    with open('Passwords.csv', 'r') as file:
        content = csv.reader(file)
        for lines in content:
            try:
                if lines[0] == u and lines[1] == p:
                    return 2
            except IndexError:
                None
```

```
def logInDirections():
    if(user_search() == 1):
        global track_username
        global track_password
        track_username = str(userEnt.get())
        window1.destroy()
        loadNavBar()
    elif(user_search() == 2):
        track_username = str(userEnt.get())
        window1.destroy()
        loadUserPage()
    else:
        print('No')
```



```

submitButton = tk.Button(text='Submit',font=('Arial',16),bg='white',width=16, command =
logInDirections)
submitButton.place(x=310, y=239)
submitButton.configure(highlightbackground='white')

about_us_pageButton = tk.Button(text='About Us',font=('Arial',9),bg='white',width=8,
command = lambda : (window1.destroy(), about_us_page()))
about_us_pageButton.place(x=720, y=300)
about_us_pageButton.configure(highlightbackground='white')

ExitButton = tk.Button(text='Exit',font=('Arial',15),bg='#FF3131', command =
window1.destroy)
ExitButton.place(x=345,y=300)
ExitButton.configure(highlightbackground='black')

window1.mainloop()

#User Page
def loadUserPage():
    window2=tk.Tk()
    window2.title("User Task Management")
    window2.geometry('750x450')
    window2.resizable(False,False)
    window2.configure(bg="#D58A94")

    andrew_image = tk.PhotoImage(file="TASK.png")
    andrew_picture = tk.Label(window2, image=andrew_image)
    andrew_picture.place(x=420,y=120)

    banner = tk.Label(window2, text="Task Management", bg='#D58A94', fg='white',
font=("Helvetica", 25))
    banner.pack(side=tk.TOP, fill=tk.X)

    titleEnt = tk.Entry(window2,width = 25)
    titleEnt.place(x=10, y=100)

    titleLabel = tk.Label(text='Title',font=('Arial',15))
    titleLabel.place(x=10,y=70)

    desEnt = tk.Entry(window2,width = 25)
    desEnt.place(x = 10, y=160)

```

```

desLabel = tk.Label(text='Description',font=('Arial',15))
desLabel.place(x=10,y=130)

dateEnt = tk.Entry(window2,width = 25)
dateEnt.place(x=10, y=220)

dateLabel = tk.Label(text='Date (mm/dd/yy)',font=('Arial',15))
dateLabel.place(x=10,y=190)

durEnt = tk.Entry(window2,width = 25)
durEnt.place(x = 10, y=280)

durLabel = tk.Label(text='Duration',font=('Arial',15))
durLabel.place(x=10,y=250)

startLabel = tk.Label(text='Start Time (Military)',font=('Arial',15))
startLabel.place(x=10,y=310)

startEnt = tk.Entry(window2,width = 25)
startEnt.place(x = 10, y=340)

def add():
    u = str(track_username)
    t = str(titleEnt.get())
    d = str(desEnt.get())
    da = str(dateEnt.get())
    du = str(durEnt.get())
    s = str(startEnt.get())
    taskFile = 'Task.csv'
    with open(taskFile, 'a') as file:
        field = [u, t, d, da, du, s]
        csvwriter = csv.writer(file)
        csvwriter.writerow(field)
    file.close()

addButton = tk.Button(text='Add',font=('Arial',12),width=16, command=add)
addButton.place(x=230, y=120)

def edit():
    u = str(track_username)
    t = str(titleEnt.get())
    d = str(desEnt.get())
    da = str(dateEnt.get())
    du = str(durEnt.get())

```

```

s = str(startEnt.get())
data = []
with open('Task.csv', 'r') as file:
    content = csv.reader(file)
    for lines in content:
        try:
            if lines[0] == u and lines[1] == t:
                newLine = [u, t, d, da, du, s]
                data.append(newLine)
            else:
                data.append(lines)

        except IndexError:
            None
with open('Task.csv', 'w') as file:
    writer = csv.writer(file)
    writer.writerows(data)
file.close()
file.close()

editButton = tk.Button(text='Edit',font=('Arial',12),width=16, command = edit)
editButton.place(x=230, y=170)

def search():
    u = str(track_username)
    t = str(titleEnt.get())
    d = str(desEnt.get())
    da = str(dateEnt.get())
    du = str(durEnt.get())
    s = str(startEnt.get())
    with open('Task.csv', 'r') as file:
        content = csv.reader(file)
        for lines in content:
            try:
                if lines[0] == u and lines[1] == t and lines[2] == d and lines[3] == da and lines[4]
                == du and lines[5] == s:
                    window = (f'Title of Task: {lines[1]}\n\nDescription of Task: {lines[2]}\n\nDate
of Task: {lines[3]}\n\nDuration of Task: {lines[4]}\n\nStart Time: {lines[5]}')
                    lblTitleofDisplay=tk.Label(text='Searched Task',font=('Arial',15))
                    lblTitleofDisplay.place(x=400,y=70)
                    lblDisplay=tk.Text(window2, width = 40, height = 20)
                    lblDisplay.place(x=400,y=100)
                    display_text = (window)
                    lblDisplay.delete(1.0, 'end')

```

```
lblDisplay.insert(tk.END, display_text)
```

```
except IndexError:
    None
file.close()
```

```
searchButton = tk.Button(text='Search',font=('Arial',12),width=16,command=search)
searchButton.place(x=230, y=220)
```

```
def remove():
    u = str(track_username)
    t = str(titleEnt.get())
    d = str(desEnt.get())
    da = str(dateEnt.get())
    du = str(durEnt.get())
    s = str(startEnt.get())
    data = []
    with open('Task.csv', 'r') as file:
        content = csv.reader(file)
        for lines in content:
            try:
                if lines[0] != u and lines[1] != t and lines[2] != d and lines[3] != da and lines[4] != s
and lines[5] != du:
                    data.append(lines)
            except IndexError:
                None
    with open('Task.csv', 'w') as file:
        writer = csv.writer(file)
        writer.writerows(data)
    file.close()
```

```
removeButton = tk.Button(text='Remove',font=('Arial',12),width=16,command=remove)
removeButton.place(x=230, y=270)
```

```
ExitButton = tk.Button(text='Log Out',font=('Arial',15),bg='red',command= lambda:
(window2.destroy(), loadLogIn()))
ExitButton.place(x=225,y=390)
```

```
CalenderButton = tk.Button(text='Calender',font=('Arial',15),bg='blue', command = lambda:
(window2.destroy(), loadCalendarPage1()))
CalenderButton.place(x=50,y=390)
```

```

window2.mainloop()

#Admin Nav Bar
#Admin User Nav Bar - TMS
def loadNavBar():
    window3=tk.Tk()
    window3.title("Admin User")
    window3.geometry('475x225')
    window3.resizable(False,False)
    window3.configure(bg="#D58A94")

    banner = tk.Label(window3, text="Task Management System", bg='#D58A94', fg='white',
font=("Helvetica", 25))
    banner.pack(side=tk.TOP, fill=tk.X)

    sub_banner = tk.Label(window3, text="Admin User Options", bg='#D58A94', fg='white',
font=("Helvetica", 18))
    sub_banner.pack(side=tk.TOP, fill=tk.X)

    Task_Btn = tk.Button(text='Task Management',font=('Arial',11), command = lambda:
(window3.destroy(), loadAdminTaskMan()))
    Task_Btn.place(x=20, y=90)

    User_Btn = tk.Button(text='User Management',font=('Arial',11), command = lambda:
(window3.destroy(), loadAdminUserMan()))
    User_Btn.place(x=170, y=90)

    Admin_Btn = tk.Button(text='Admin Management',font=('Arial',11), command = lambda:
(window3.destroy(), loadAdminAdminMan()))
    Admin_Btn.place(x=320, y=90)

    ExitButton = tk.Button(text='Log Out',font=('Arial',15),bg='red',command= lambda:
(window3.destroy(), loadLogIn()))
    ExitButton.place(x=170,y=160)

    window3.mainloop()

#Admin User - Task Management
#Admin User Page - TMS
def loadAdminTaskMan():
    window4=tk.Tk()
    window4.title("Admin Task Management")
    window4.geometry('750x450')
    window4.resizable(False,False)

```

```

window4.configure(bg="#D58A94")

andrew_image = tk.PhotoImage(file="TASK.png")
andrew_picture = tk.Label(window4, image=andrew_image)
andrew_picture.place(x=420,y=120)

banner = tk.Label(window4, text="Admin Task Management", bg='#D58A94', fg='white',
font=("Helvetica", 25))
banner.pack(side=tk.TOP, fill=tk.X)

titleEnt = tk.Entry(window4,width = 25)
titleEnt.place(x=10, y=100)

titleLabel = tk.Label(text='Title',font=('Arial',15))
titleLabel.place(x=10,y=70)

desEnt = tk.Entry(window4,width = 25)
desEnt.place(x = 10, y=160)

desLabel = tk.Label(text='Description',font=('Arial',15))
desLabel.place(x=10,y=130)

dateEnt = tk.Entry(window4,width = 25)
dateEnt.place(x=10, y=220)

dateLabel = tk.Label(text='Date (mm/dd/yy)',font=('Arial',15))
dateLabel.place(x=10,y=190)

durEnt = tk.Entry(window4,width = 25)
durEnt.place(x = 10, y=280)

durLabel = tk.Label(text='Duration',font=('Arial',15))
durLabel.place(x=10,y=250)

startLabel = tk.Label(text='Start Time (Military)',font=('Arial',15))
startLabel.place(x=10,y=310)

startEnt = tk.Entry(window4,width = 25)
startEnt.place(x = 10, y=340)

def add():
    u = str(track_username)
    t = str(titleEnt.get())
    d = str(desEnt.get())

```

```

da = str(dateEnt.get())
du = str(durEnt.get())
s = str(startEnt.get())
taskFile = 'Task.csv'
with open(taskFile, 'a') as file:
    field = [u, t, d, da, du, s]
    csvwriter = csv.writer(file)
    csvwriter.writerow(field)
file.close()

```

```

addButton = tk.Button(text='Add',font=('Arial',12),width=16, command=add)
addButton.place(x=230, y=120)

```

```

def edit():
    u = str(track_username)
    t = str(titleEnt.get())
    d = str(desEnt.get())
    da = str(dateEnt.get())
    du = str(durEnt.get())
    s = str(startEnt.get())
    data = []
    with open('Task.csv', 'r') as file:
        content = csv.reader(file)
        for lines in content:
            try:
                if lines[0] == u and lines[1] == t:
                    newLine = [u, t, d, da, du, s]
                    data.append(newLine)
            else:
                data.append(lines)
        except IndexError:
            None
    with open('Task.csv', 'w') as file:
        writer = csv.writer(file)
        writer.writerows(data)
    file.close()
    file.close()

```

```

editButton = tk.Button(text='Edit',font=('Arial',12),width=16, command = edit)
editButton.place(x=230, y=170)

```

```

def search():
    u = str(track_username)
    t = str(titleEnt.get())

```

```

d = str(desEnt.get())
da = str(dateEnt.get())
du = str(durEnt.get())
s = str(startEnt.get())
with open('Task.csv', 'r') as file:
    content = csv.reader(file)
    for lines in content:
        try:
            if lines[0] == u and lines[1] == t and lines[2] == d and lines[3] == da and lines[4]
== du and lines[5] == s:
                window = (f'Title of Task: {lines[1]}\n\nDescription of Task: {lines[2]}\n\nDate
of Task: {lines[3]}\n\nDuration of Task: {lines[4]}\n\nStart Time: {lines[5]}')
                lblTitleofDisplay=tk.Label(text='Searched Task',font=('Arial',15))
                lblTitleofDisplay.place(x=400,y=70)
                lblDisplay=tk.Text(window4, width = 40, height = 20)
                lblDisplay.place(x=400,y=100)
                display_text = (window)
                lblDisplay.delete(1.0, 'end')
                lblDisplay.insert(tk.END, display_text)

        except IndexError:
            None
file.close()

searchButton = tk.Button(text='Search',font=('Arial',12),width=16,command=search)
searchButton.place(x=230, y=220)

def remove():
    u = str(track_username)
    t = str(titleEnt.get())
    d = str(desEnt.get())
    da = str(dateEnt.get())
    du = str(durEnt.get())
    s = str(startEnt.get())
    data = []
    with open('Task.csv', 'r') as file:
        content = csv.reader(file)
        for lines in content:
            try:
                if lines[0] != u and lines[1] != t and lines[2] != d and lines[3] != da and lines[4] != s
and lines[5] != du:
                    data.append(lines)
            except IndexError:
                None

```



```

with open('Task.csv', 'w') as file:
    writer = csv.writer(file)
    writer.writerows(data)
file.close()

```

```

removeButton = tk.Button(text='Remove',font=('Arial',12),width=16,command=remove)
removeButton.place(x=230, y=270)

```

```

ExitButton = tkm.Button(text='Return To Nav Bar',font=('Arial',15),bg='red',command=
lambda: (window4.destroy(), loadNavBar()))
ExitButton.place(x=225,y=390)

```

```

CalenderButton = tkm.Button(text='Calender',font=('Arial',15),bg='blue', command = lambda:
(window4.destroy(), loadCalendarPage2()))
CalenderButton.place(x=50,y=390)

```

```

window4.mainloop()

```

```

#Admin User - User Management

```

```

#Admin User Page - TMS

```

```

def loadAdminUserMan():

```

```

    window5=tk.Tk()
    window5.title("Admin User Task Management")
    window5.geometry('445x250')
    window5.resizable(False,False)
    window5.configure(bg="#D58A94")

```

```

    banner = tk.Label(window5, text="User Management", bg='#D58A94', fg='white',
font=("Helvetica", 25))
    banner.pack(side=tk.TOP, fill=tk.X)

```

```

    addUserEnt = tk.Entry(window5,width = 25)
    addUserEnt.place(x=40, y=100)

```

```

    addUserLabel = tk.Label(text='Username',font=('Arial',15))
    addUserLabel.place(x=40,y=70)

```

```

    passwordEnt = tk.Entry(window5,width = 25)
    passwordEnt.place(x = 40, y=150)

```

```

    passwordLabel = tk.Label(text='Password',font=('Arial',15))
    passwordLabel.place(x=40,y=120)

```

```
def add():
    u = str(addUserEnt.get())
    p = str(passwordEnt.get())
    passwordFile = 'Passwords.csv'
    with open(passwordFile, 'a') as file:
        field = [u, p]
        csvwriter = csv.writer(file)
        csvwriter.writerow(field)
    file.close()
```

```
addUserButton = tk.Button(text='Add', font=('Arial',12), width=16, command = add)
addUserButton.place(x=250, y=80)
```

```
def remove():
    u = str(addUserEnt.get())
    p = str(passwordEnt.get())
    data = []
    with open('Passwords.csv', 'r') as file:
        content = csv.reader(file)
        for lines in content:
            try:
                if lines[0] != u and lines[1] != p:
                    data.append(lines)
            except IndexError:
                None
    with open('Passwords.csv', 'w') as file:
        writer = csv.writer(file)
        writer.writerows(data)
    file.close()
```

```
removeUserButton = tk.Button(text='Remove', font=('Arial',12), width=16, command =
remove)
removeUserButton.place(x=250, y=130)
```

```
ExitButton = tkm.Button(text='Return To Nav Bar',font=('Arial',15),bg='red', command=
lambda: (window5.destroy(), loadNavBar()))
ExitButton.place(x=140,y=190)
```

```
window5.mainloop()
```

```
#Admin user - Admin Management
#Admin User Page - TMS
def loadAdminAdminMan():
    window6=tk.Tk()
```

```

window6.title("Admin User Admin Management")
window6.geometry('350x330')
window6.resizable(False,False)
window6.configure(bg="#D58A94")

banner = tk.Label(window6, text="Admin Management", bg='#D58A94', fg='white',
font=("Helvetica", 25))
banner.pack(side=tk.TOP, fill=tk.X)

nameEnt = tk.Entry(window6,width = 40)
nameEnt.place(x=50, y= 100)

nameLabel = tk.Label(text='Name',font=('Arial',15))
nameLabel.place(x=50,y=70)

passwordEnt = tk.Entry(window6,width = 40)
passwordEnt.place(x=50, y= 170)

passwordLabel = tk.Label(text='Password',font=('Arial',15))
passwordLabel.place(x=50,y=140)

def edit_admin():
    u = str(track_username)
    new_username = str(nameEnt.get())
    new_password = str(passwordEnt.get())
    data = []
    newLines = [new_username, new_password]
    with open('Admin_Passwords.csv', 'r') as file:
        content = csv.reader(file)
        for lines in content:
            try:
                if lines[0] == u:
                    data.append(newLines)
            else:
                data.append(lines)
        except IndexError:
            None
    file.close()
    with open('Admin_Passwords.csv', 'w') as file:
        writer = csv.writer(file)
        writer.writerows(data)
    file.close()

```

```
editAdminButton = tk.Button(text='Edit Admin', font=('Arial',12), width=16, command =
edit_admin)
editAdminButton.place(x=100, y=210)
```

```
ExitButton = tk.Button(text='Return To Nav Bar',font=('Arial',15),bg='red',command=
lambda: (window6.destroy(), loadNavBar()))
ExitButton.place(x=90,y=260)
```

```
window6.mainloop()
```

```
#Calendar Page
```

```
#Calendar Page - TMS
```

```
def loadCalendarPage1():
```

```
    window7=tk.Tk()
```

```
    window7.title("Calendar Page")
```

```
    window7.geometry('800x440')
```

```
    window7.resizable(False,False)
```

```
    window7.configure(bg="#D58A94")
```

```
    banner = tk.Label(window7, text="Task Management System", bg='#D58A94', fg='white',
font=("Helvetica", 25))
```

```
    banner.pack(side=tk.TOP, fill=tk.X)
```

```
    calendar = tkc.Calendar(window7,font=('Arial',15), selectmode='day', year=2023, month=12,
day=13)
```

```
    calendar.place(x = 10, y = 110)
```

```
    lblTitleofDisplay=tk.Label(text='Tasks on this Date',font=('Arial',15))
```

```
    lblTitleofDisplay.place(x=400,y=70)
```

```
    lblTitleofCalendar=tk.Label(text='Calendar',font=('Arial',15))
```

```
    lblTitleofCalendar.place(x=10,y=70)
```

```
    print(calendar.get_date())
```

```
def week():
```

```
    u = str(track_username)
```

```
    with open('Task.csv', 'r') as file:
```

```
        content =csv.reader(file)
```

```
        for lines in content:
```

```
            try:
```

```
                if lines[0] == u and lines[3] == calendar.get_date():
```

```
                    p = (f'Title of Task: {lines[1]}\n\nDescription of Task: {lines[2]}\n\nDate of Task:
{lines[3]}\n\nDuration of Task: {lines[4]}\n\nStart Time: {lines[5]}')
```

```

        window = str(p)
        lblDisplay=tk.Text(window7, width = 43, height = 15)
        lblDisplay.place(x=400,y=110)
        display_text = (window)
        lblDisplay.delete(1.0, 'end')
        lblDisplay.insert(tk.END, display_text)
    except IndexError:
        None
    file.close()

    weekButton = tkm.Button(text='Search for Tasks on this Date',font=('Arial',15),bg='blue',
command = week)
    weekButton.place(x= 30, y=380)

    ExitButton = tkm.Button(text='Return To Task Management',font=('Arial',15),bg='red',
command = lambda: (window7.destroy(), loadUserPage()))
    ExitButton.place(x=520,y=380)

    window7.mainloop()

def loadCalendarPage2():
    window8=tk.Tk()
    window8.title("Calendar Page")
    window8.geometry('800x440')
    window8.resizable(False,False)
    window8.configure(bg="#D58A94")

    banner = tk.Label(window8, text="Task Management System", bg='#D58A94', fg='white',
font=("Helvetica", 25))
    banner.pack(side=tk.TOP, fill=tk.X)

    calendar = tkc.Calendar(window8,font=('Arial',15), selectmode='day', year=2023, month=12,
day=13)
    calendar.place(x = 10, y = 110)

    lblTitleofDisplay=tk.Label(text='Tasks on this Date',font=('Arial',15))
    lblTitleofDisplay.place(x=400,y=70)

    lblTitleofCalendar=tk.Label(text='Calendar',font=('Arial',15))
    lblTitleofCalendar.place(x=10,y=70)

    print(calendar.get_date())

```

```

def week():
    u = str(track_username)
    with open('Task.csv', 'r') as file:
        content = csv.reader(file)
        for lines in content:
            try:
                if lines[0] == u and lines[3] == calendar.get_date():
                    p = (f'Title of Task: {lines[1]}\n\nDescription of Task: {lines[2]}\n\nDate of Task: {lines[3]}\n\nDuration of Task: {lines[4]}\n\nStart Time: {lines[5]}')
                    window = str(p)
                    lblDisplay=tk.Text(window8, width = 43, height = 15)
                    lblDisplay.place(x=400,y=110)
                    display_text = (window)
                    lblDisplay.delete(1.0, 'end')
                    lblDisplay.insert(tk.END, display_text)
            except IndexError:
                None
    file.close()

weekButton = tkm.Button(text='Search for Tasks on this Date',font=('Arial',15),bg='blue',
command = week)
weekButton.place(x= 30, y=380)

ExitButton = tkm.Button(text='Return To Task Management',font=('Arial',15),bg='red',
command = lambda: (window8.destroy(), loadAdminTaskMan()))
ExitButton.place(x=520,y=380)

window8.mainloop()

def about_us_page():
    window9=tk.Tk()
    window9.title("About Us Page")
    window9.geometry('820x600')
    window9.resizable(False,False)
    window9.configure(bg='pink')

    nick_image=tk.PhotoImage(file="AboutNick.png")
    nick_picture=tk.Label(window9, image=nick_image)
    nick_picture.place(x=20,y=250)
    andrew_image=tk.PhotoImage(file="AboutAndrew.png")
    andrew_picture=tk.Label(window9, image=andrew_image)
    andrew_picture.place(x=200,y=250)
    chris_image=tk.PhotoImage(file="AboutChris.png")
    chris_picture=tk.Label(window9, image=chris_image)

```

```

chris_picture.place(x=600,y=250)
ben_image=tk.PhotoImage(file="AboutBen.png")
ben_picture=tk.Label(window9, image=ben_image)
ben_picture.place(x=410,y=250)
# Main project description
main_description_label=tk.Label(window9, text="About Our Project",bg='pink',
font=('Helvetica', 22, 'bold'))
main_description_label.pack(pady=10)

main_description_text="    Our project is a Task Management System (TMS) that allows
users to manage their tasks
by adding, removing, editing, and searching their tasks. For each task, they enter a title,
description,
date, and duration to which their tasks will be stored in a csv database. Hope you enjoy our
project
and find it useful. Let's meet the creators of TMS, otherwise known as MC Squared!"

main_description=tk.Label(window9, text=main_description_text, bg='pink',
font=('Helvetica', 12), justify='left')
main_description.place(x=45, y=75)

# Nick
nick_label=tk.Label(window9, text="Nick",bg='pink', font=('Helvetica', 16, 'underline'))
nick_label.place(x=50,y=200)

nick_placeholder_label=tk.Label(window9,
text="Senior\nCybersecurity",bg='pink',font=('Helvetica', 11))
nick_placeholder_label.place(x=28,y=500)

# Andrew
andrew_label=tk.Label(window9, text="Andrew", bg='pink', font=('Helvetica', 16,
'underline'))
andrew_label.place(x=250,y=200)

andrew_placeholder_label=tk.Label(window9, text="Freshmen\nSoftware
Development",bg='pink',font=('Helvetica', 11))
andrew_placeholder_label.place(x=200,y=500)

# Ben
ben_label=tk.Label(window9, text="Ben",bg='pink', font=('Helvetica', 16, 'underline'))
ben_label.place(x=485,y=200)

```

```
ben_placeholder_label=tk.Label(window9, text="Freshmen\nSoftware
Development",bg='pink',font=('Helvetica', 11))
ben_placeholder_label.place(x=415,y=500)

# Chris
Chris_label=tk.Label(window9, text="Chris",bg='pink', font=('Helvetica', 16, 'underline'))
Chris_label.place(x=675,y=200)

Chris_placeholder_label=tk.Label(window9, text="Freshmen\nGame Design &
Programming",bg='pink',font=('Helvetica', 11))
Chris_placeholder_label.place(x=600,y=500)

ExitButton=tkm.Button(text='Return To Log In',font=('Arial',15),bg='red', command =
lambda: (window9.destroy(), loadLogIn()))
ExitButton.place(x=20,y=20)

window9.mainloop()

#On Opening

#Way To Check If It's Been Previously Called

#How to create and check location on users laptop?
#Some condition in the csv file?
#Check For Admin user

loadLogIn()
```



## [5] Virtual Environment

**Abstract:** In this virtual environment, we created a local environment for the project we worked on. This environment enabled us to import explicit libraries only directly pertaining to the project. We utilized commands like “python3 - m venv my\_env”, “source my\_env/bin/activate”, and “(my\_env)” in order to create the environment, display the packages used, and then run our task management system from the virtual environment in idle.

*Link To Video Of Code Running In Virtual Environment*

<https://drive.google.com/drive/folders/1qI3AcKppAlemQWo13VqlGnrQaR795cU7?usp=sharing>

## [6] Data Storage

**Abstract:** The below data storage you may see admin tasks displayed in csv format which includes the user, title, description, date, start time, duration and their entries. Following that, we have the same thing yet this time being a normal user like “MaristNick” for example. Lastly, we have different password storages. We wanted to show a hashed password version and a normal one. So, we have the admin user with their hashed password, then the normal user with their plain text password.

```
User,Title,Descirption,Date,Start Time,Duration
AdminUser,Basketball Practice,Practice with team in school gym,12/10/23,16:00,2 hours
AdminUser,Walk Dogs,Take the dogs around the lake ,12/12/23,12:00,1 hour
```

Figure 24: Task Storage

```
User,Title,Descirption,Date,Start Time,Duration
MaristNick,Presentation,Hash Library,12/5/23,17:00,7 minutes
```

Figure 25: Username/Password Storage

```
AdminUser,91e7428b8c52fce4c7e896df716777352064582e58f0a0c9787404e7455d6bd5
Username,Password
MaristNick,Cyber2121!!
```

Figure 26: Password Storage with hashed example and non-hashed password

## **[7] Libraries**

Tkinter

Tkmacosx

Tkcalendar

PIL

## **[8] References**

<https://www.lucidchart.com>

<https://www.geeksforgeeks.org/>

<https://pypi.org/>

<https://stackoverflow.com/>

[\(156\) Create A Date Picker Calendar - Python Tkinter GUI Tutorial #72 - YouTube](#)