Criterion B: Design

Flowcharts

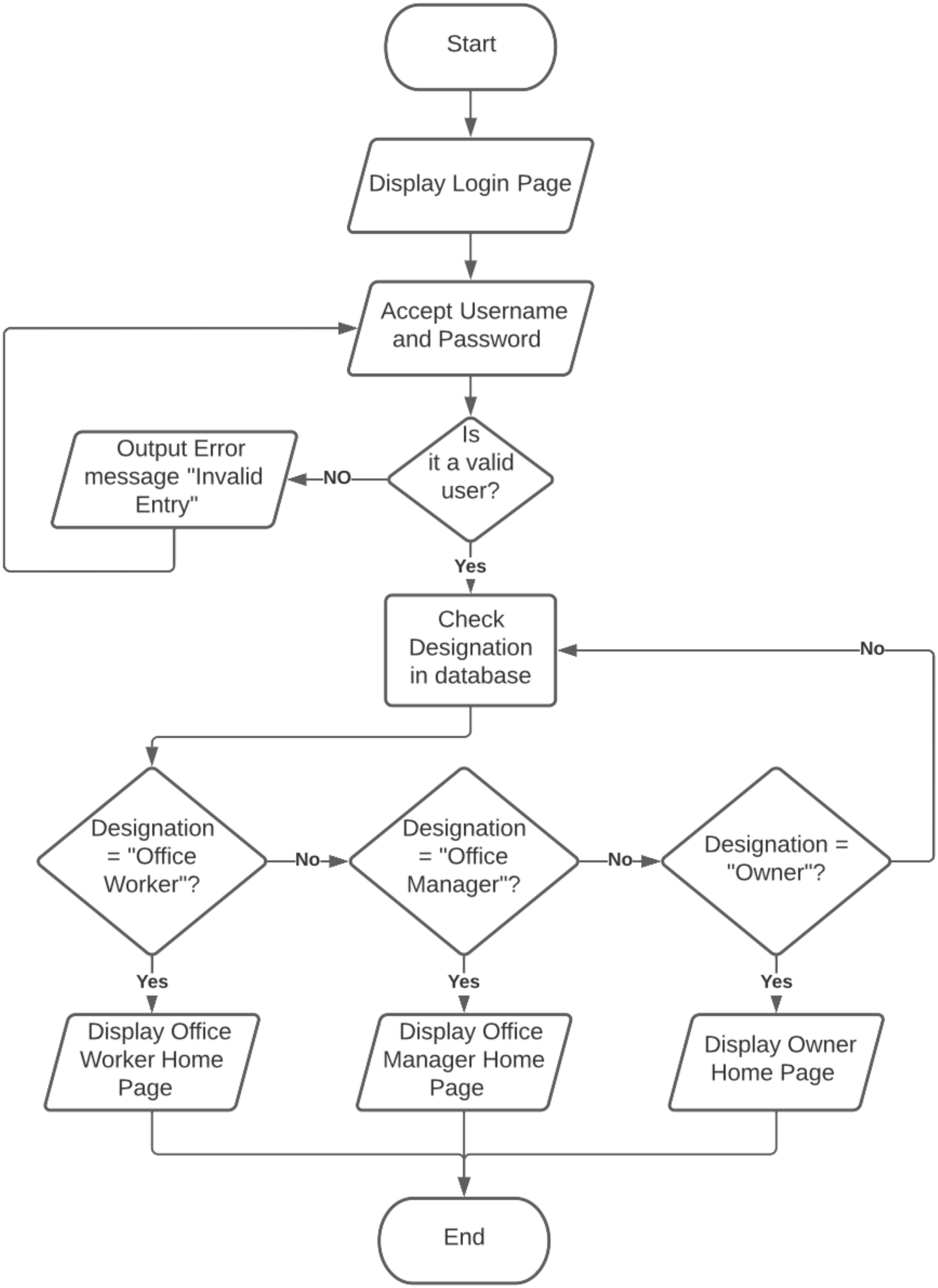

Start
Display Login Page
Accept Username and Password
Is it a valid user?
NO
Output Error message "Invalid Entry"
Yes
Check Designation in database
No
Designation = "Office Worker"?
No
Designation = "Office Manager"?
No
Designation = "Owner"?
Yes
Yes
Yes
Display Office Worker Home Page
Display Office Manager Home Page
Display Owner Home Page
End

Figure 1 - Login Page from a flowchart perspective

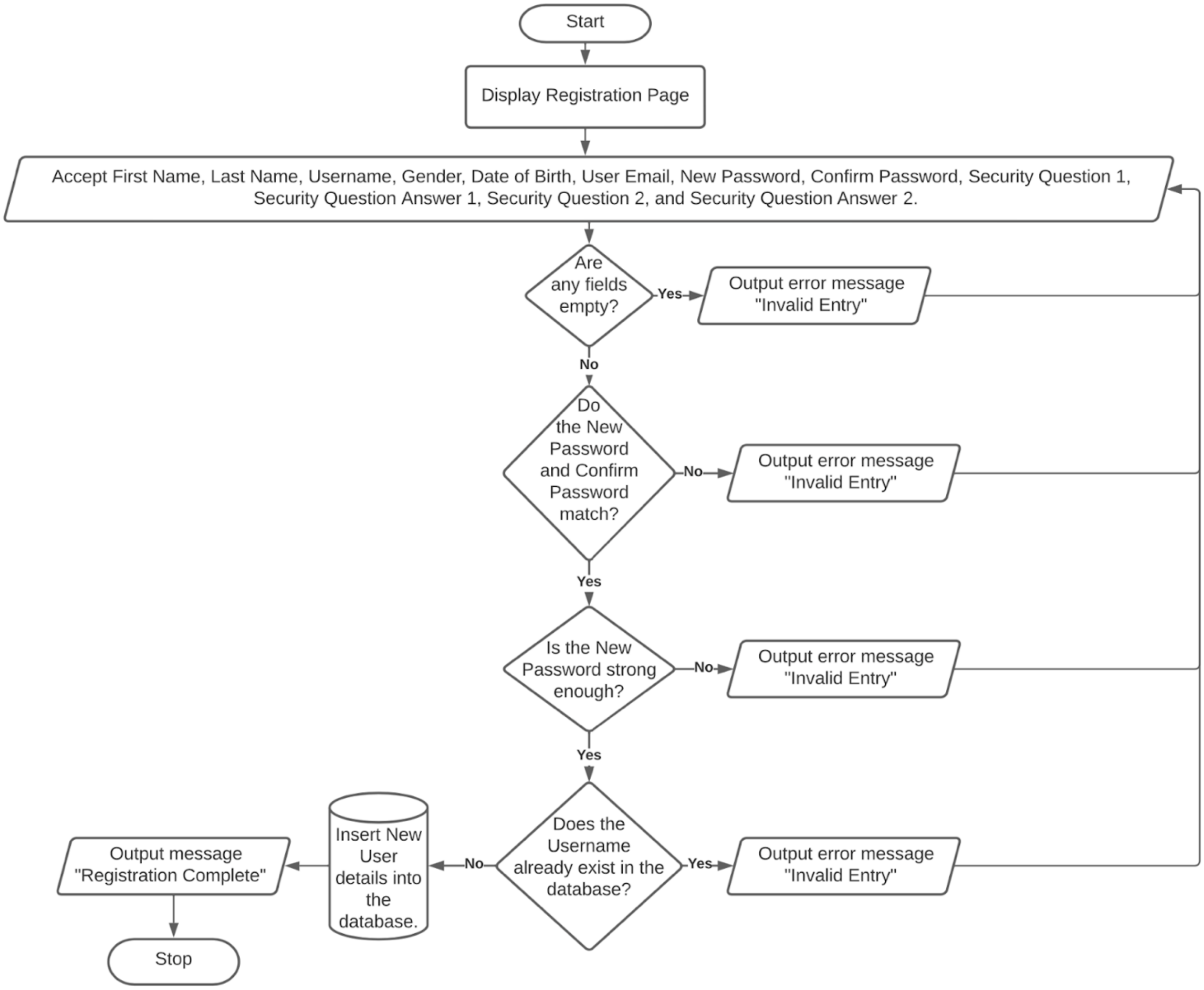

Start
Display Registration Page
Accept First Name, Last Name, Username, Gender, Date of Birth, User Email, New Password, Confirm Password, Security Question 1, Security Question Answer 1, Security Question 2, and Security Question Answer 2.
Are any fields empty?
Yes
Output error message "Invalid Entry"
No
Do the New Password and Confirm Password match?
No
Output error message "Invalid Entry"
Yes
Is the New Password strong enough?
No
Output error message "Invalid Entry"
Yes
Does the Username already exist in the database?
Yes
Output error message "Invalid Entry"
No
Insert New User details into the database.
Output message "Registration Complete"
Stop

Figure 2 - Registration Page from a flowchart perspective

Relationship between the classes

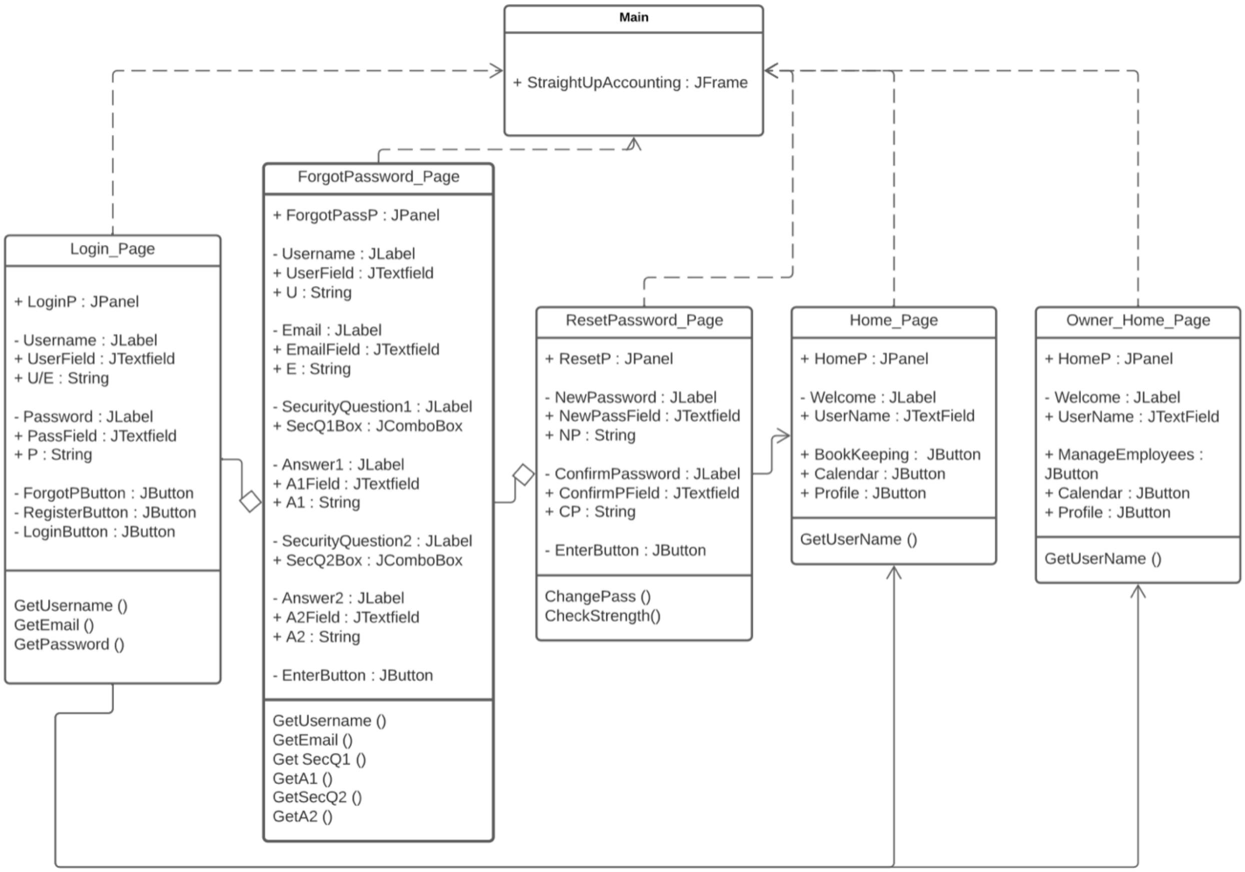


Figure 3 - Success Criteria 1, 3, and 4

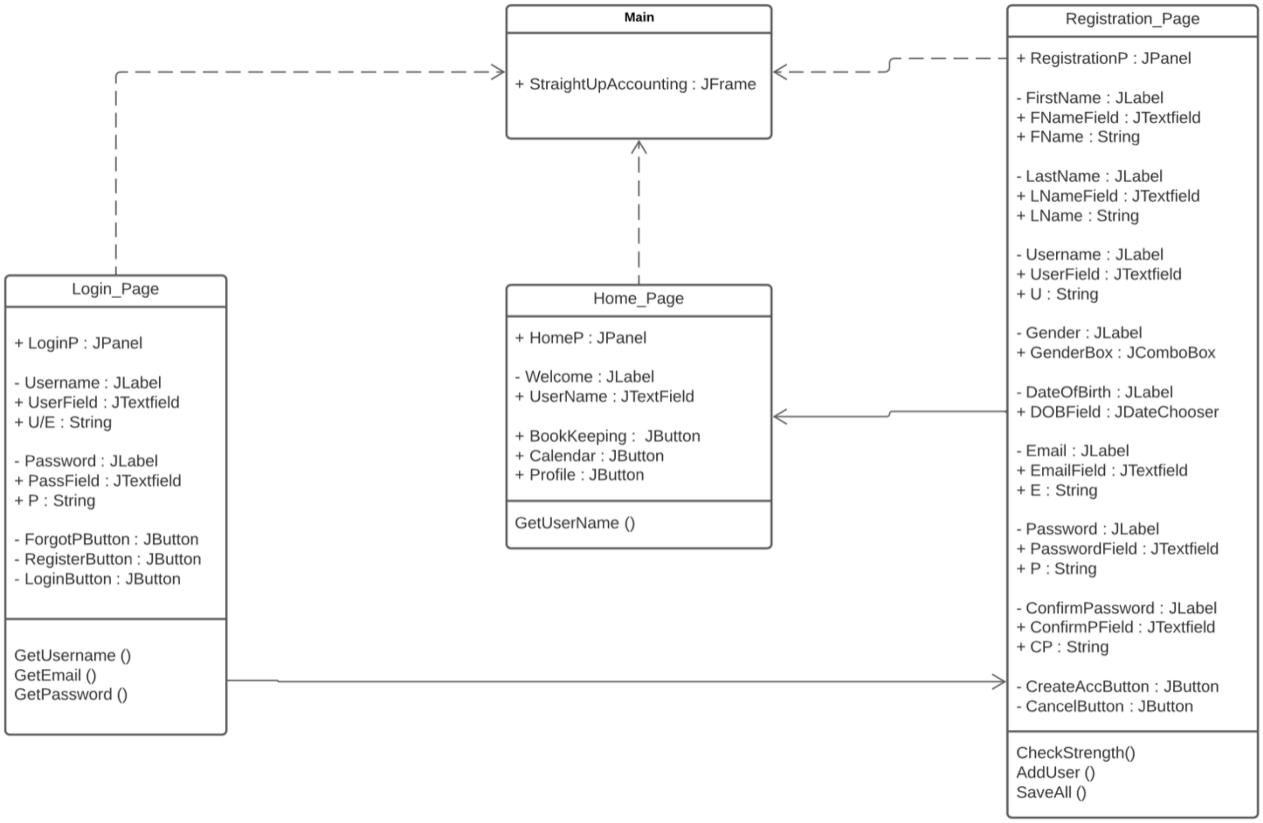


Figure 4 - Success Criteria 1 and 4

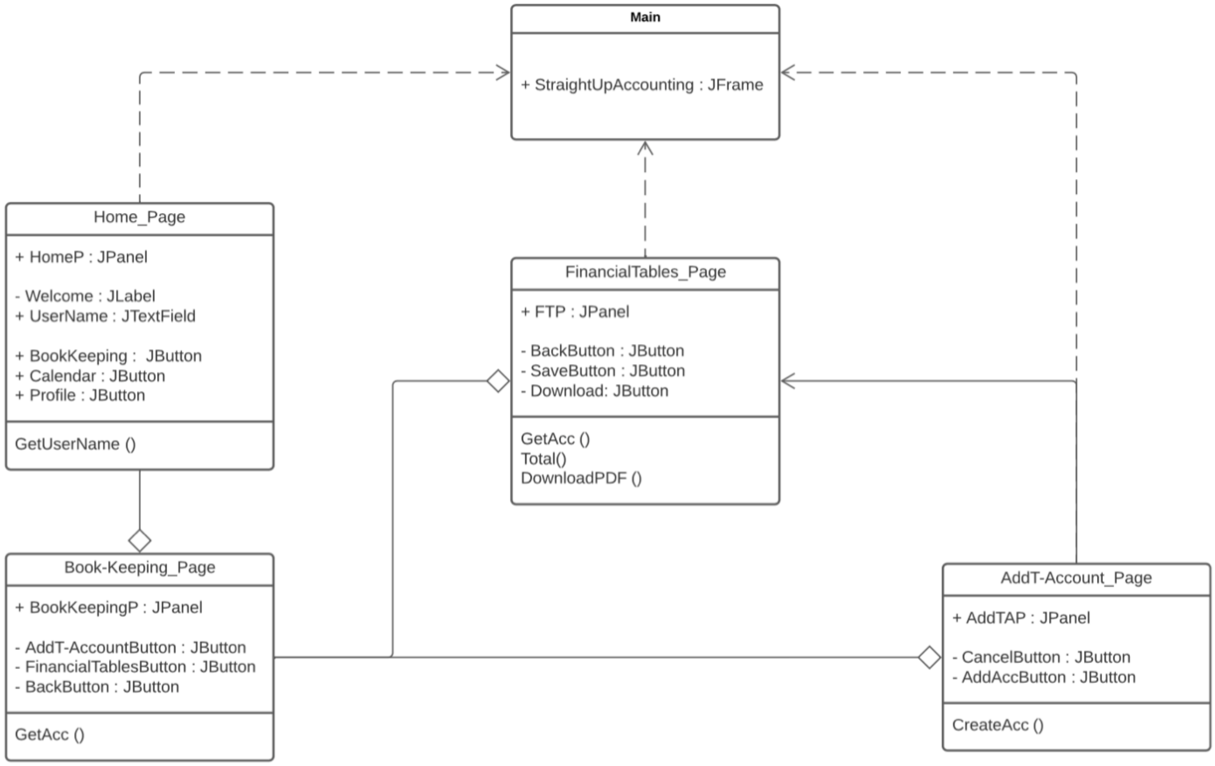


Figure 5 - Success Criteria 7, 8, and 9

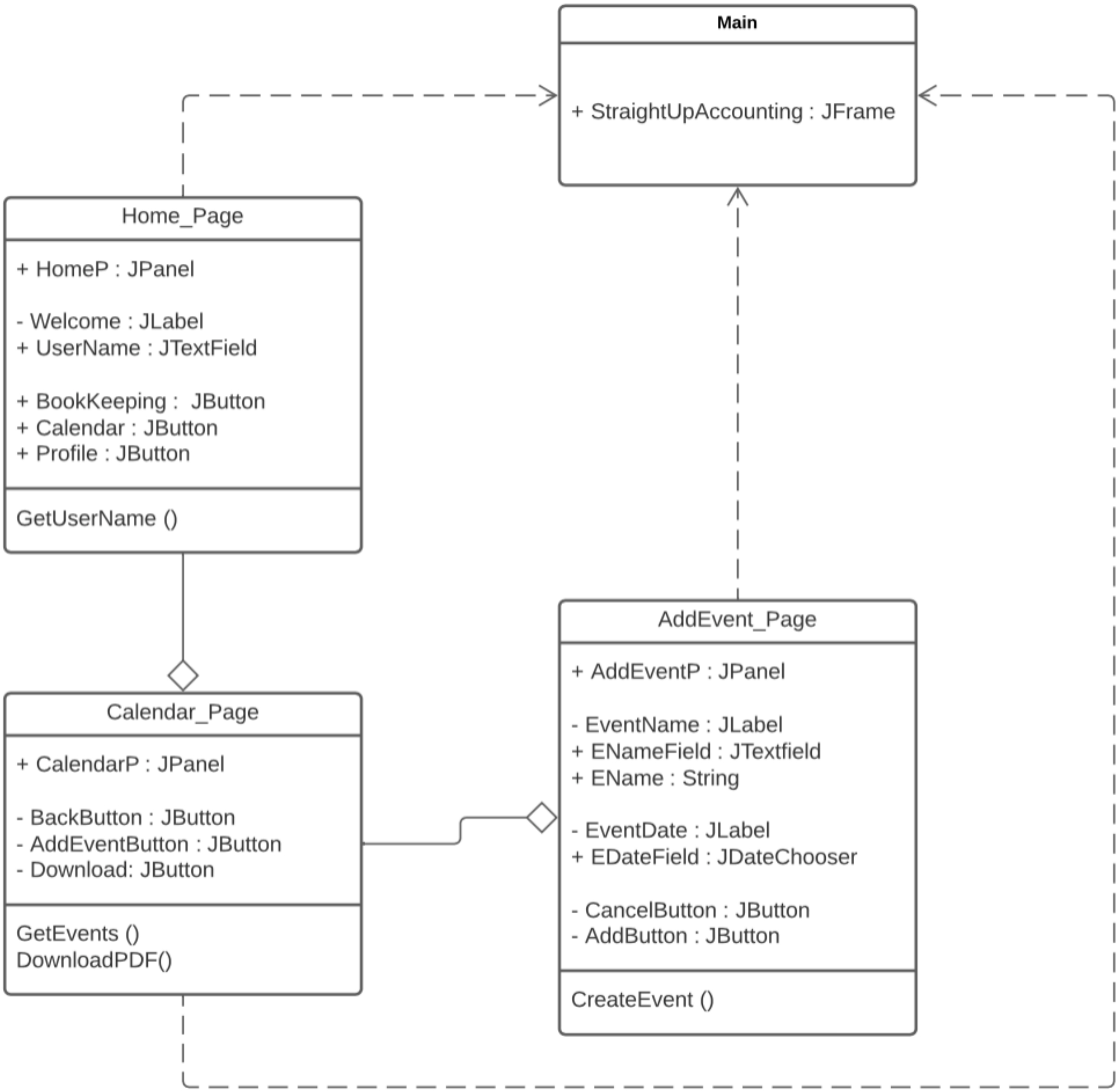


Figure 6 - Success Criteria 6 and 9

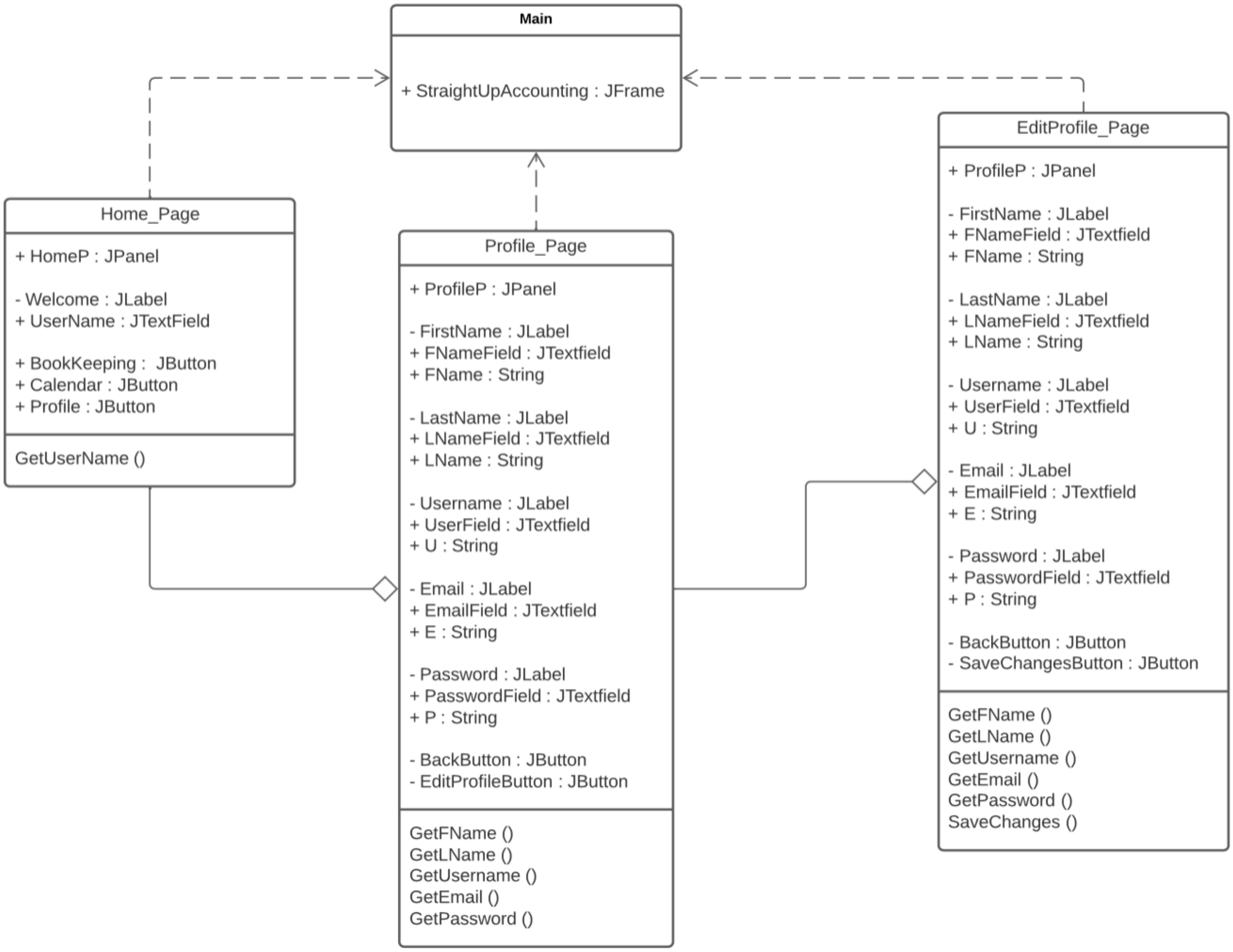


Figure 7 - Success Criteria 4, 5, and 6

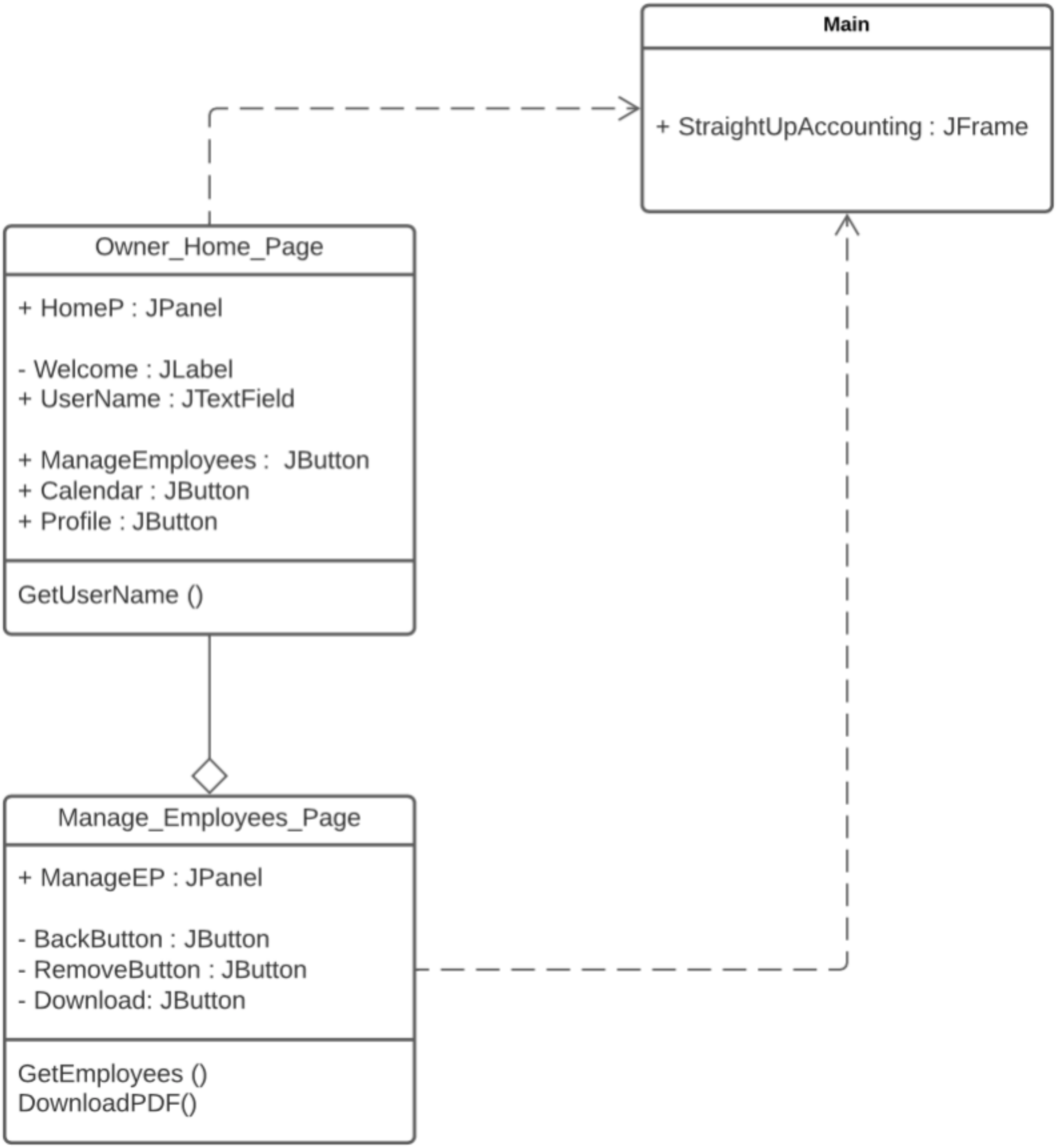


Figure 8 - Success Criteria 10

Functionality of each class

1. Main - consists of the JFrame and contains the main class
2. Login_Page - a login page which requires a unique username and password to access the program.
3. ForgotPassword_Page - a page which prompts the user to enter their unique username and answer their security questions in order to change their password.
4. ResetPassword_Page - a page which allows the user to change their password.
5. Home_Page - a page which consists of buttons which direct the user to the book-keeping, calendar and profile aspects of the program.
6. Owner_Home_Page - a page which consists of buttons which direct the user to the managing employees, calendar and profile aspects of the program.
7. Registration_Page - a registration page which requires the user to enter their personal information and a unique username in order to be a registered user of the program.
8. Book-Keeping_Page - a page which allows the user to view their entered T-Accounts.
9. FinancialTables_Page - a page which organizes all of the data from the T-Accounts into the appropriate financial table, including calculating totals.
10. AddT-Account_Page - a page which allows the user to enter T-Accounts into the program.
11. Calendar_Page - a page which allows the user to view and edit all of their saved events.
12. AddEvent_Page - a page which allows the user to add new events into their calendar.
13. Profile_Page - a page which allows the user to view all of their personal information which they entered when registering.
14. EditProfile_Page - a page which allows the user to edit their personal information and profile picture.
15. Manage_Employees_Page - a page which allows the user to view and edit all of their employees in a business.

JPanel Designs (Graphical User Interface)

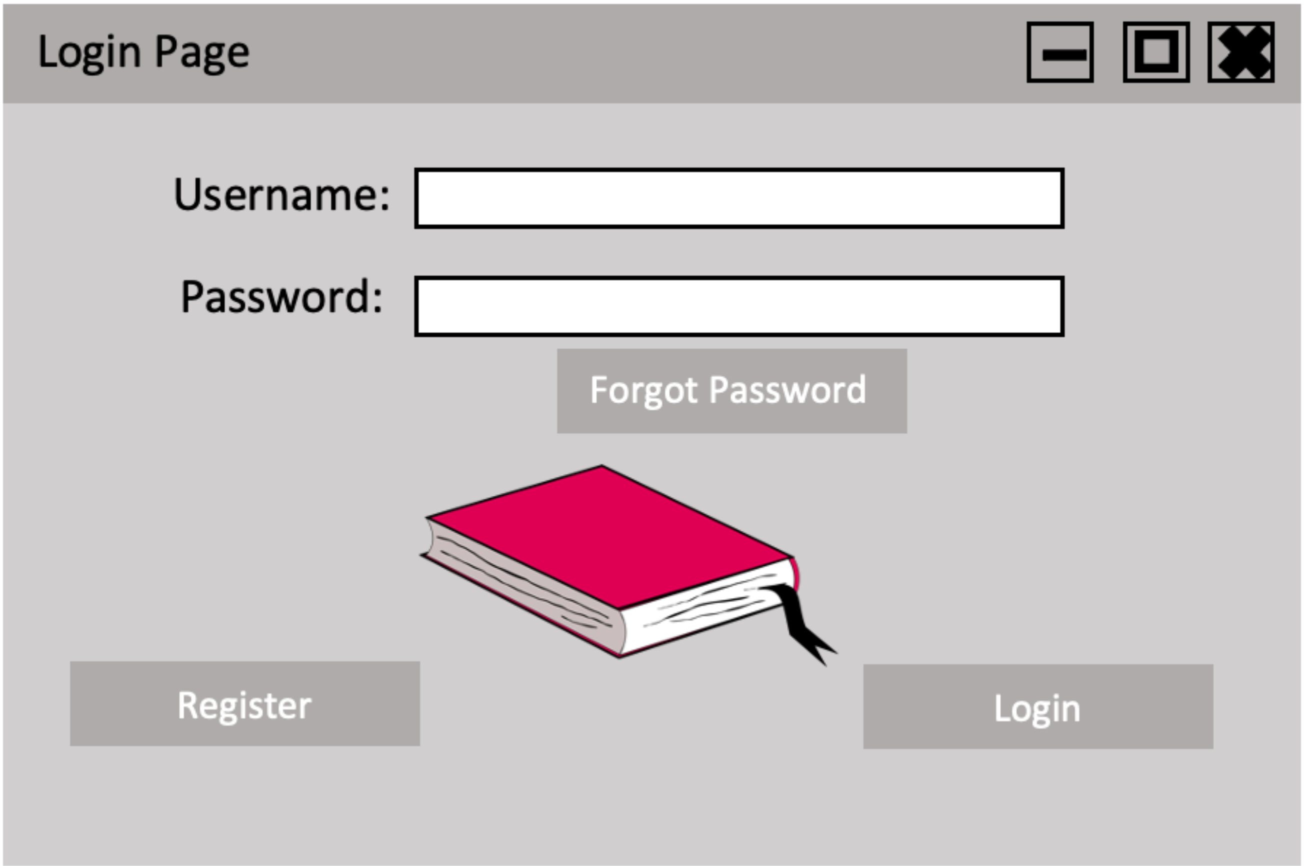


Figure 9 - Login Page Design

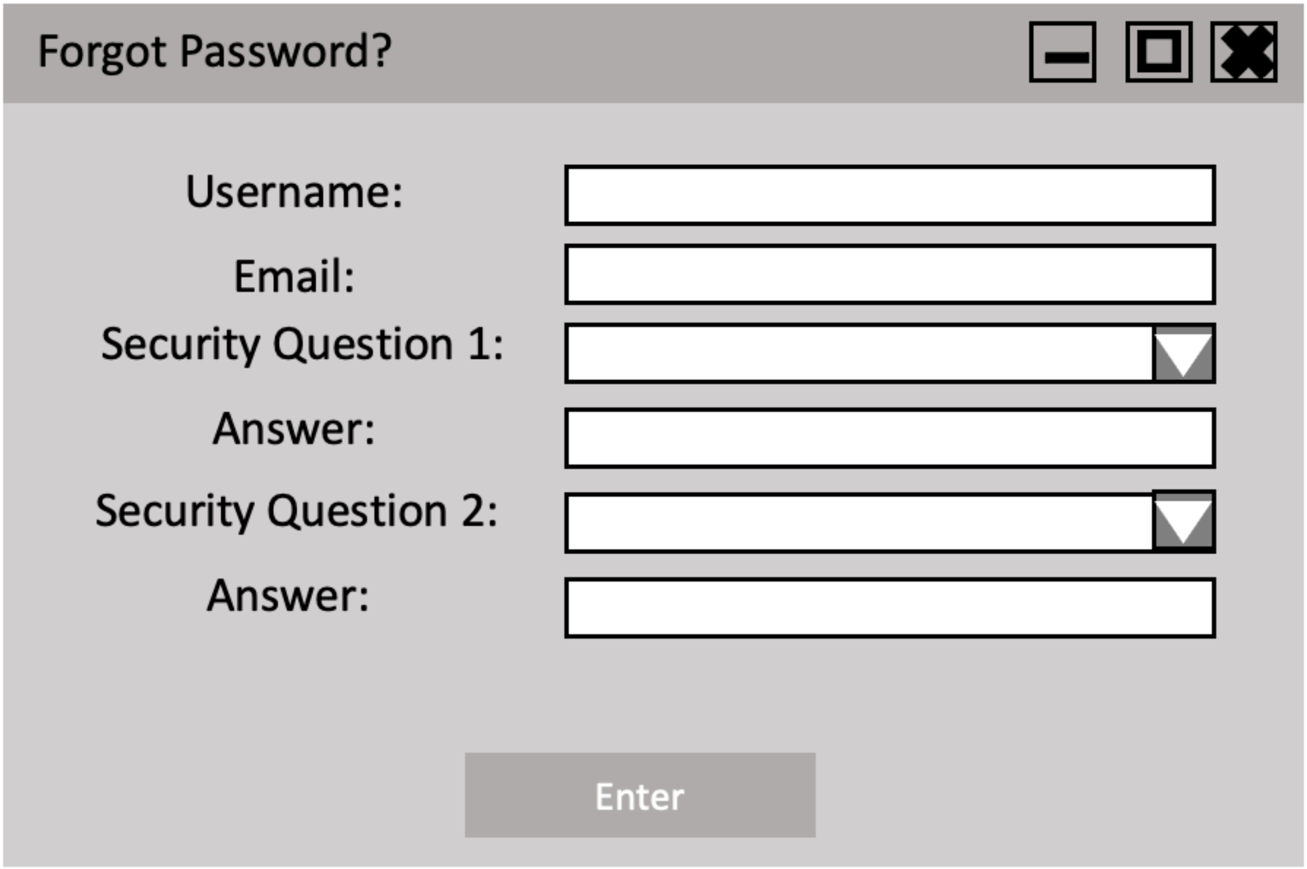


Figure 10 - Forgot Password Page Design

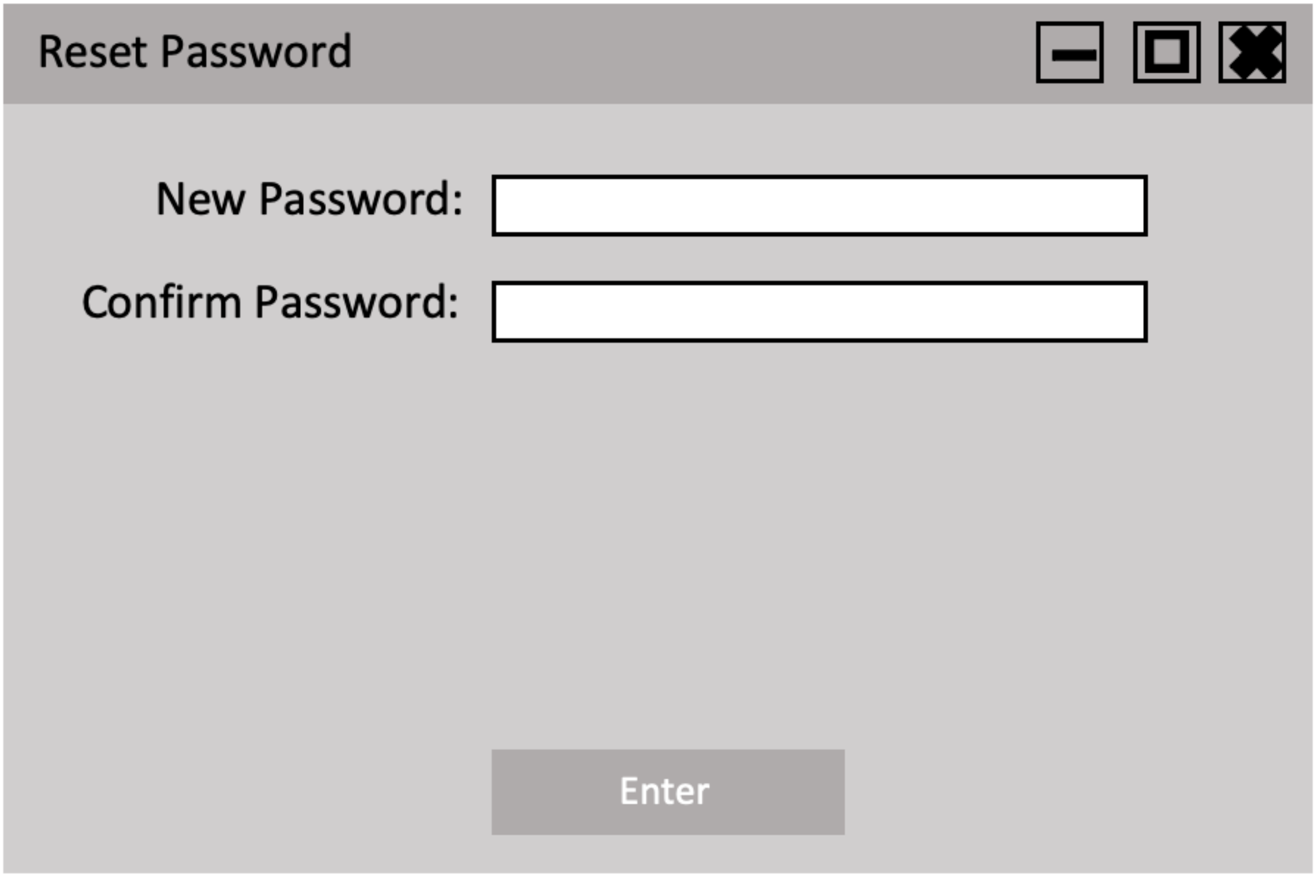


Figure 11 - Reset Password Page Design

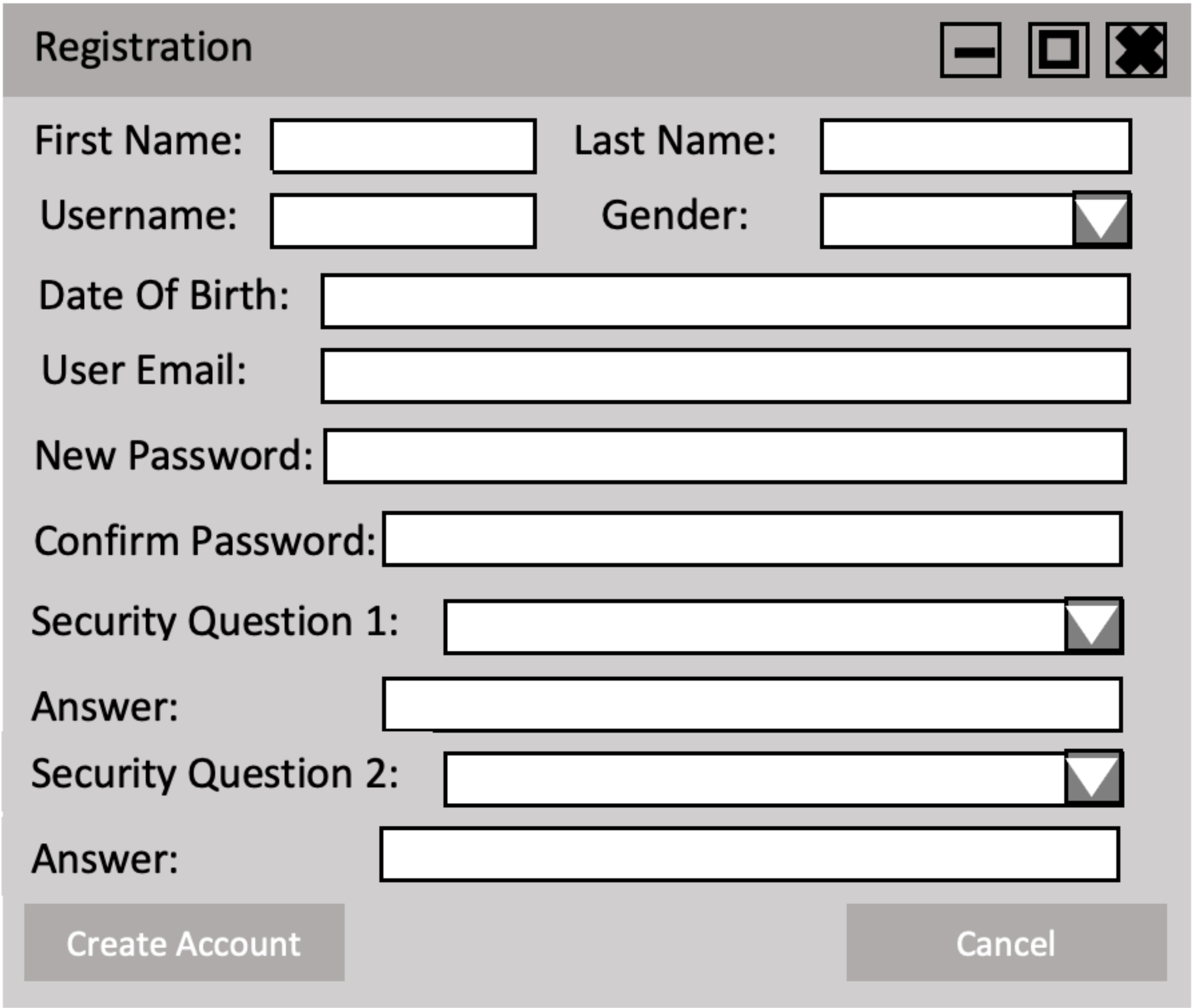


Figure 12 - Registration Page Design

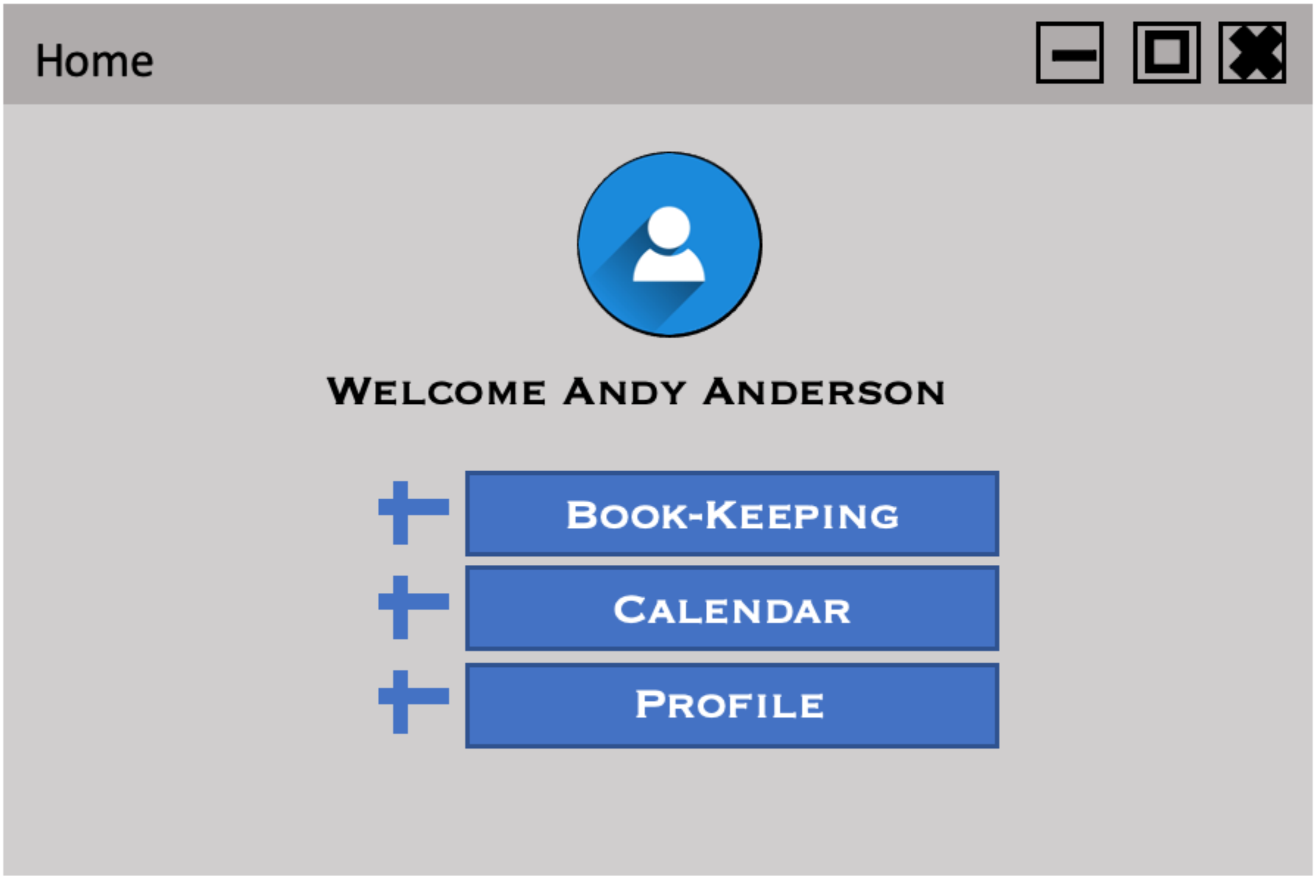


Figure 13 - Office Worker and Office Manager Page Design

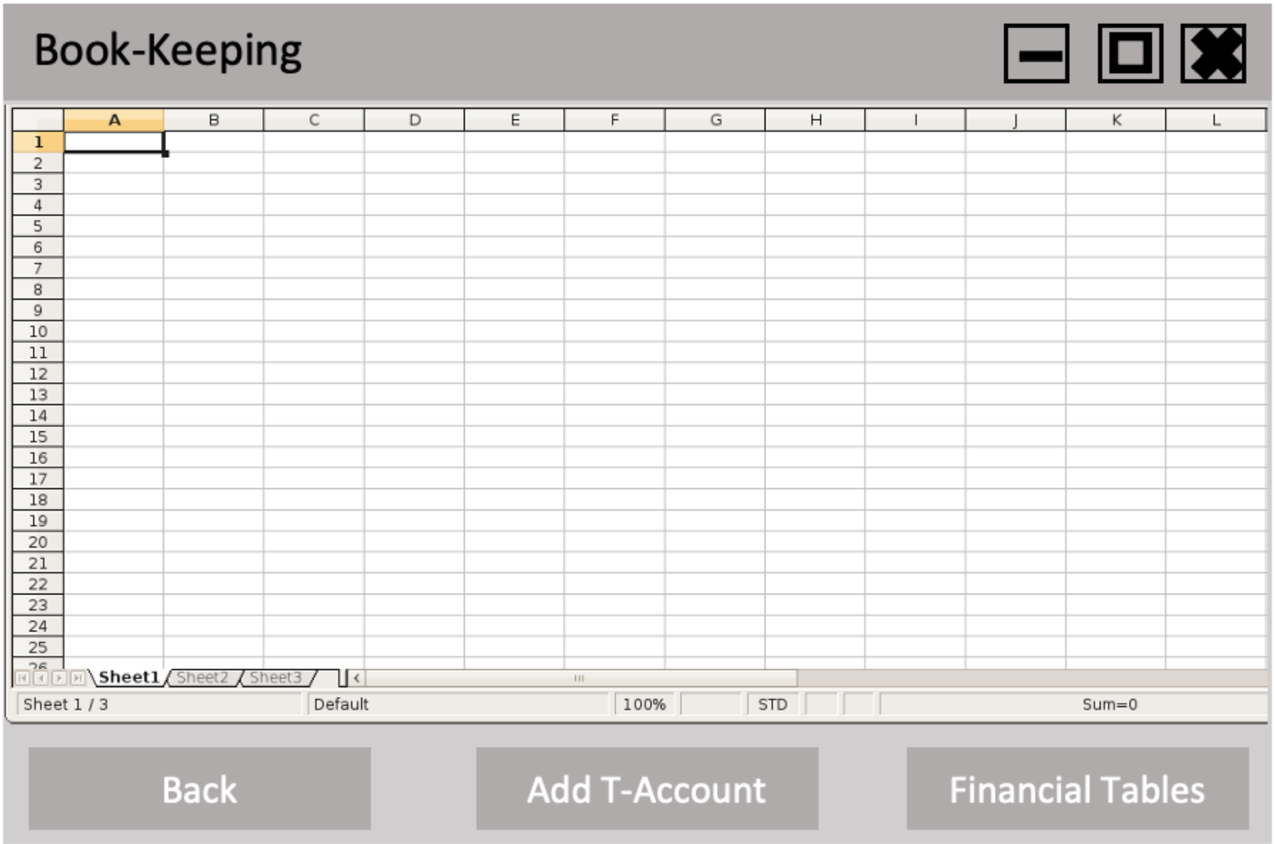
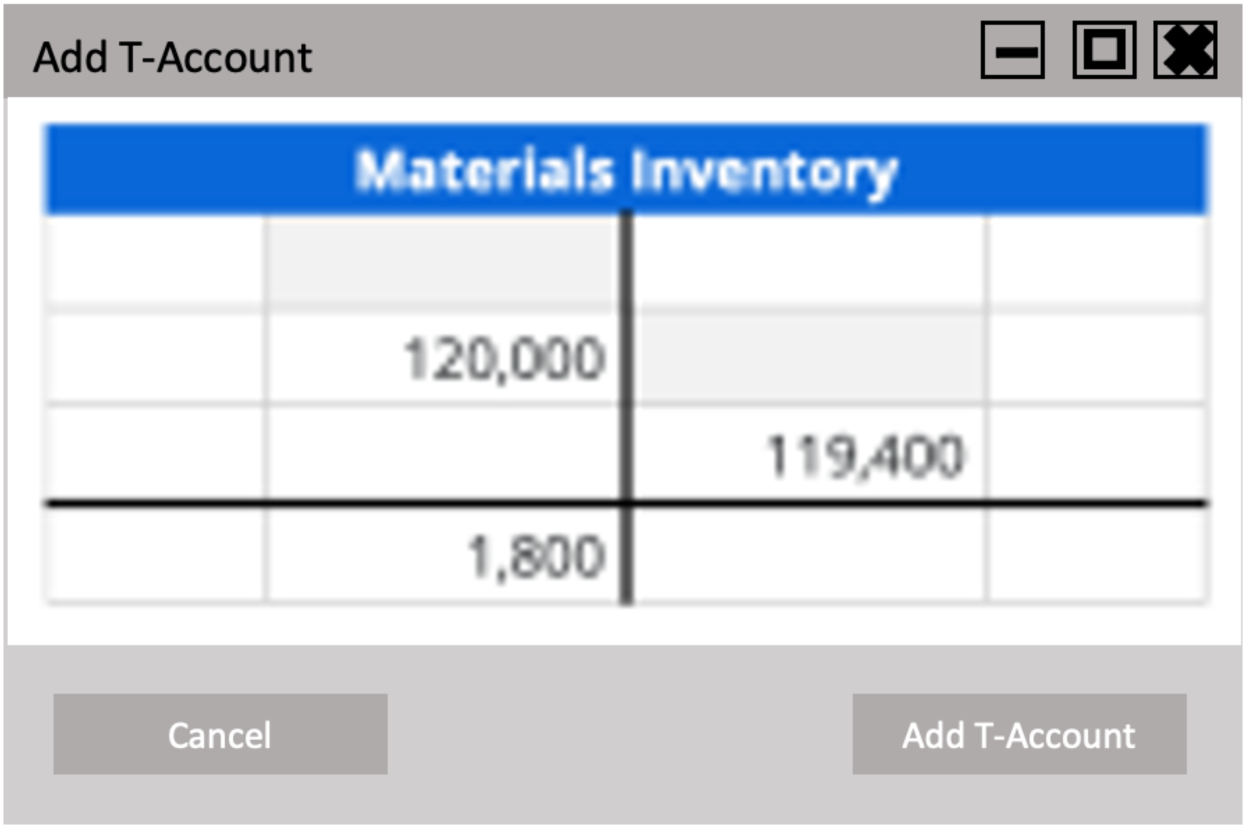
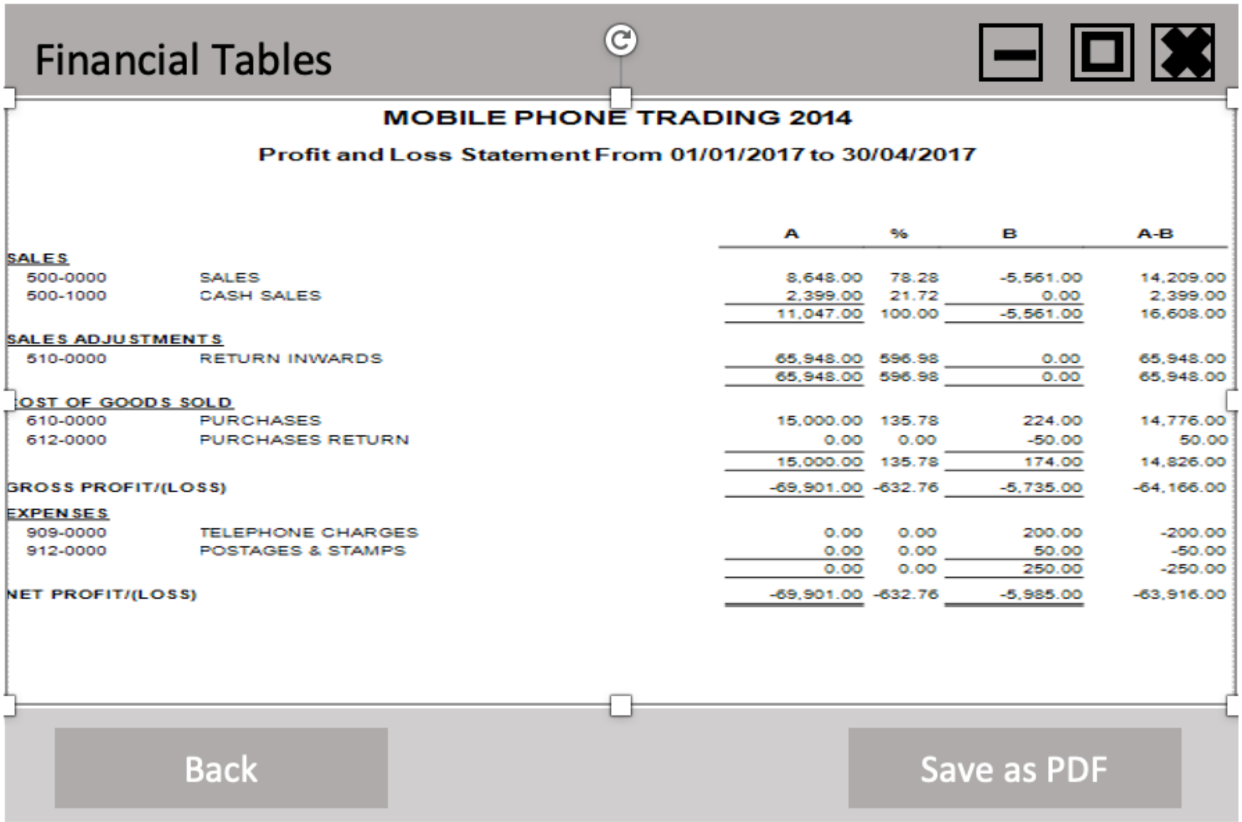


Figure 14 - Book-Keeping Page Design



Materials Inventory			
	120,000		
		119,400	
	1,800		

Figure 15 - Add T-Account Page Design

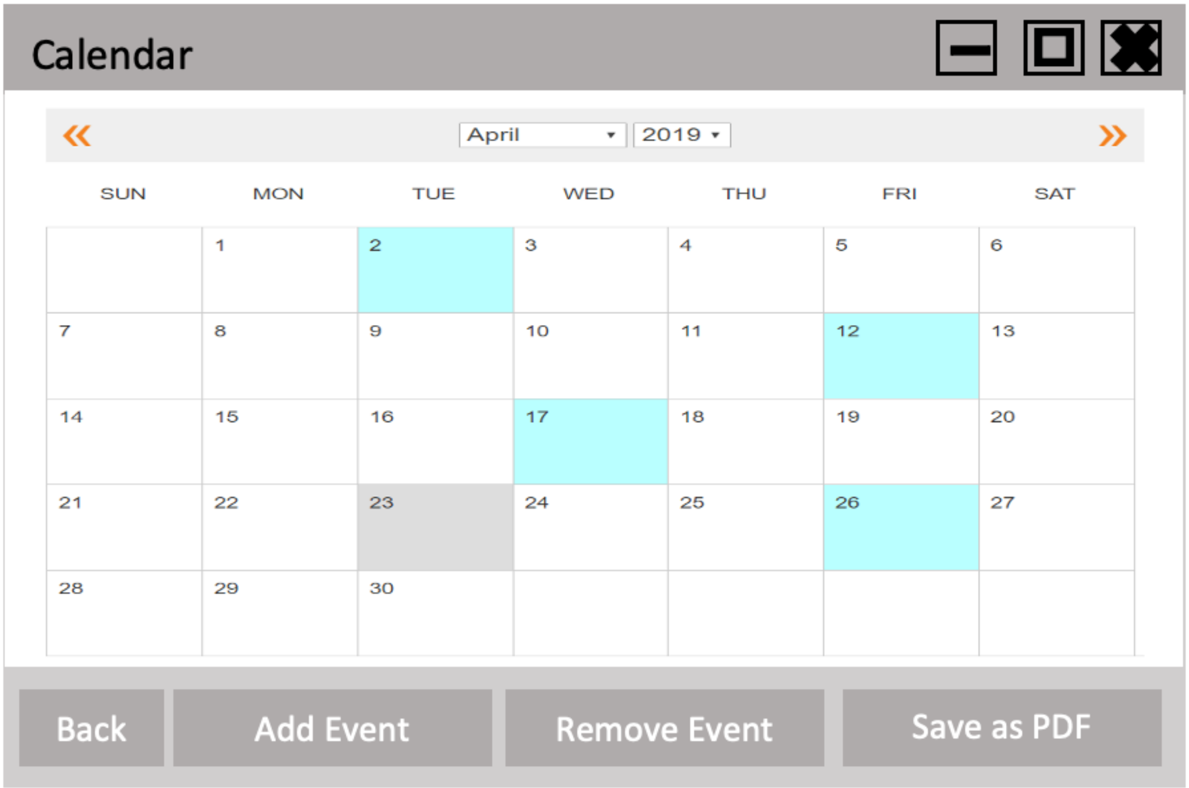


MOBILE PHONE TRADING 2014

Profit and Loss Statement From 01/01/2017 to 30/04/2017

		A	%	B	A-B
SALES					
500-0000	SALES	8,648.00	78.28	-5,561.00	14,209.00
500-1000	CASH SALES	2,399.00	21.72	0.00	2,399.00
		11,047.00	100.00	-5,561.00	16,608.00
SALES ADJUSTMENTS					
510-0000	RETURN INWARDS	65,948.00	596.98	0.00	65,948.00
		65,948.00	596.98	0.00	65,948.00
OST OF GOODS SOLD					
610-0000	PURCHASES	15,000.00	135.78	224.00	14,776.00
612-0000	PURCHASES RETURN	0.00	0.00	-50.00	50.00
		15,000.00	135.78	174.00	14,826.00
GROSS PROFIT/(LOSS)		-69,901.00	-632.76	-5,735.00	-64,166.00
EXPENSES					
909-0000	TELEPHONE CHARGES	0.00	0.00	200.00	-200.00
912-0000	POSTAGES & STAMPS	0.00	0.00	50.00	-50.00
		0.00	0.00	250.00	-250.00
NET PROFIT/(LOSS)		-69,901.00	-632.76	-5,985.00	-63,916.00

Figure 16 - Financial Tables Page Design



April ▾ 2019 ▾

SUN	MON	TUE	WED	THU	FRI	SAT
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

Figure 17 - Calendar Page Design

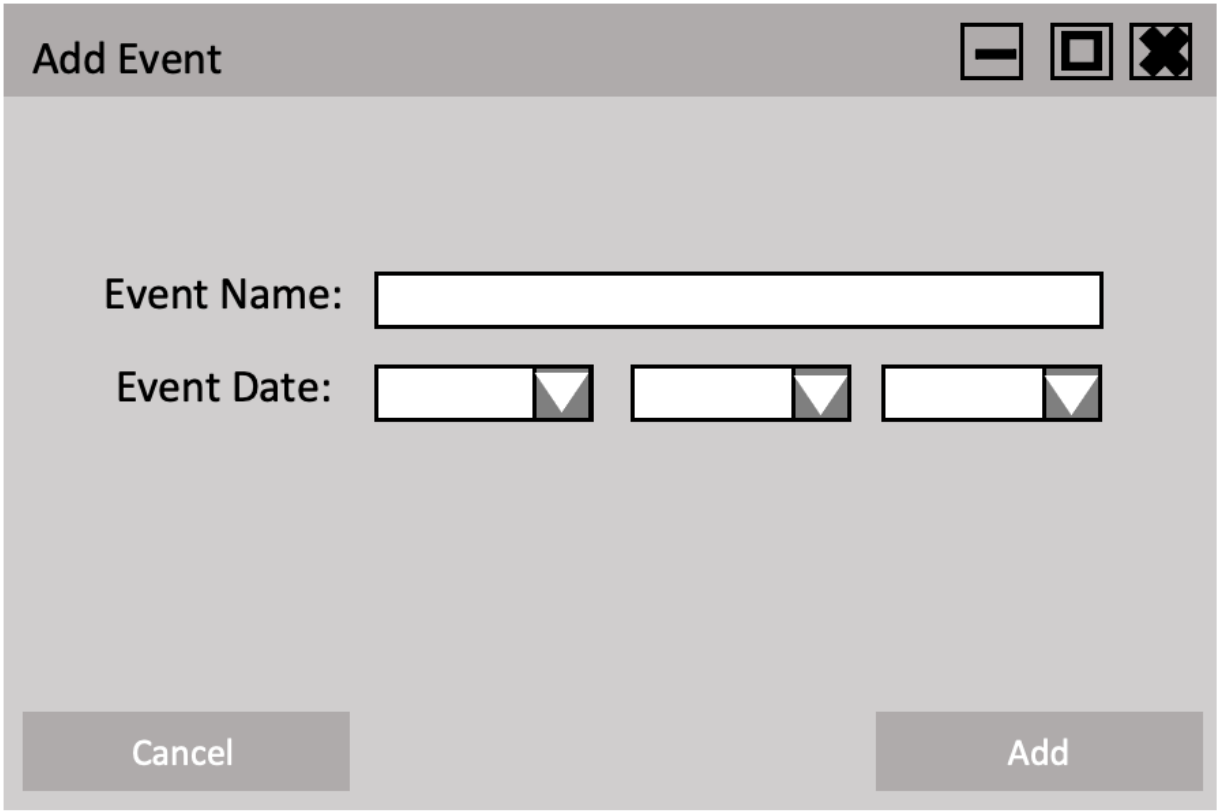
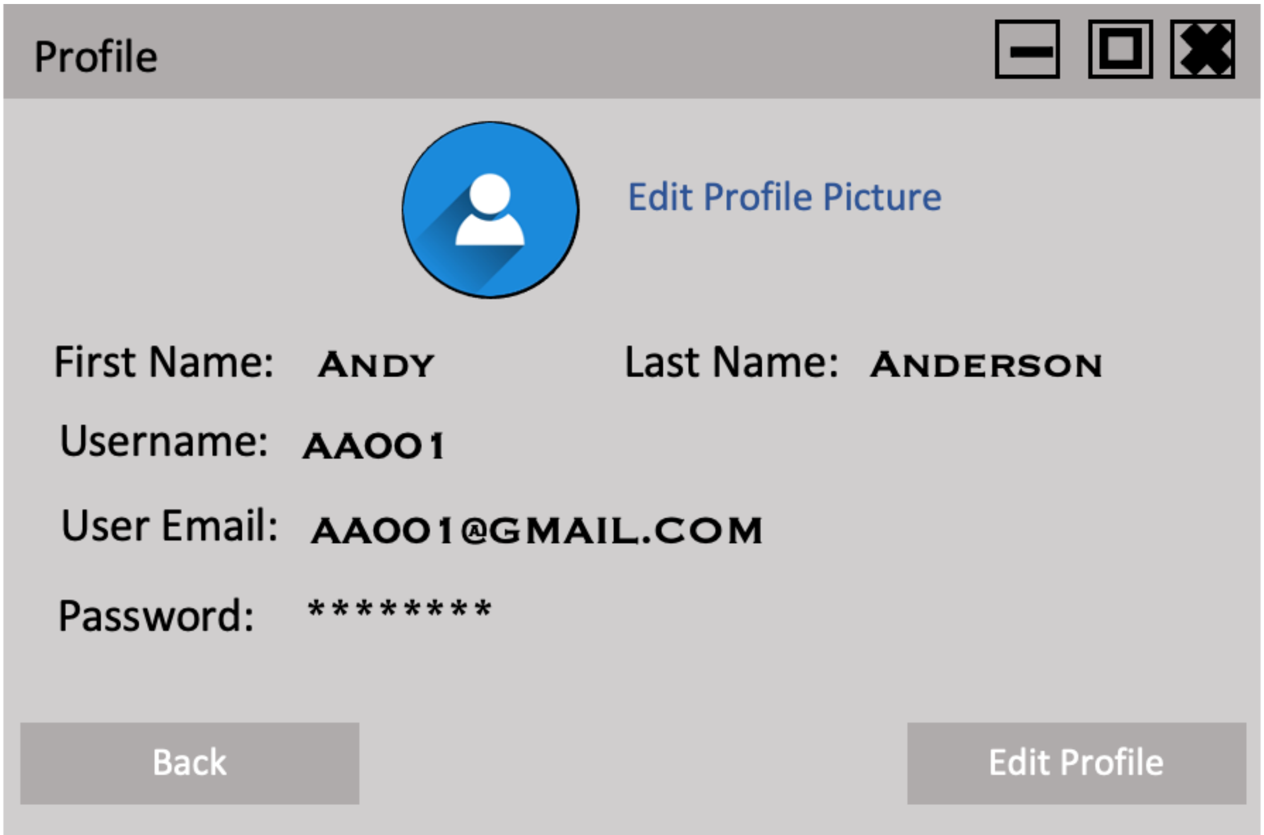


Figure 18 - Add Event Page Design



First Name: ANDY

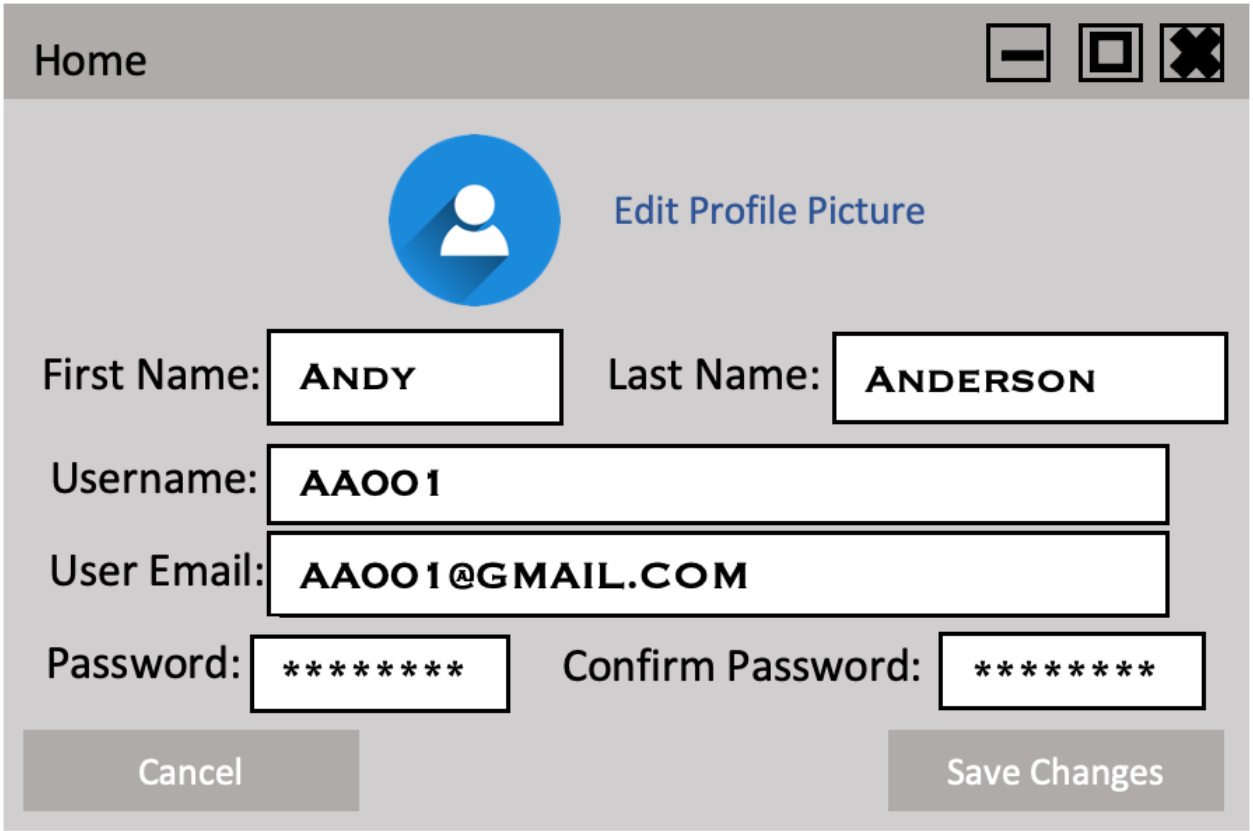
Last Name: ANDERSON

Username: AA001

User Email: AA001@GMAIL.COM

Password: ********

Figure 19 - Profile Page Design



First Name: Andy

Last Name: Anderson

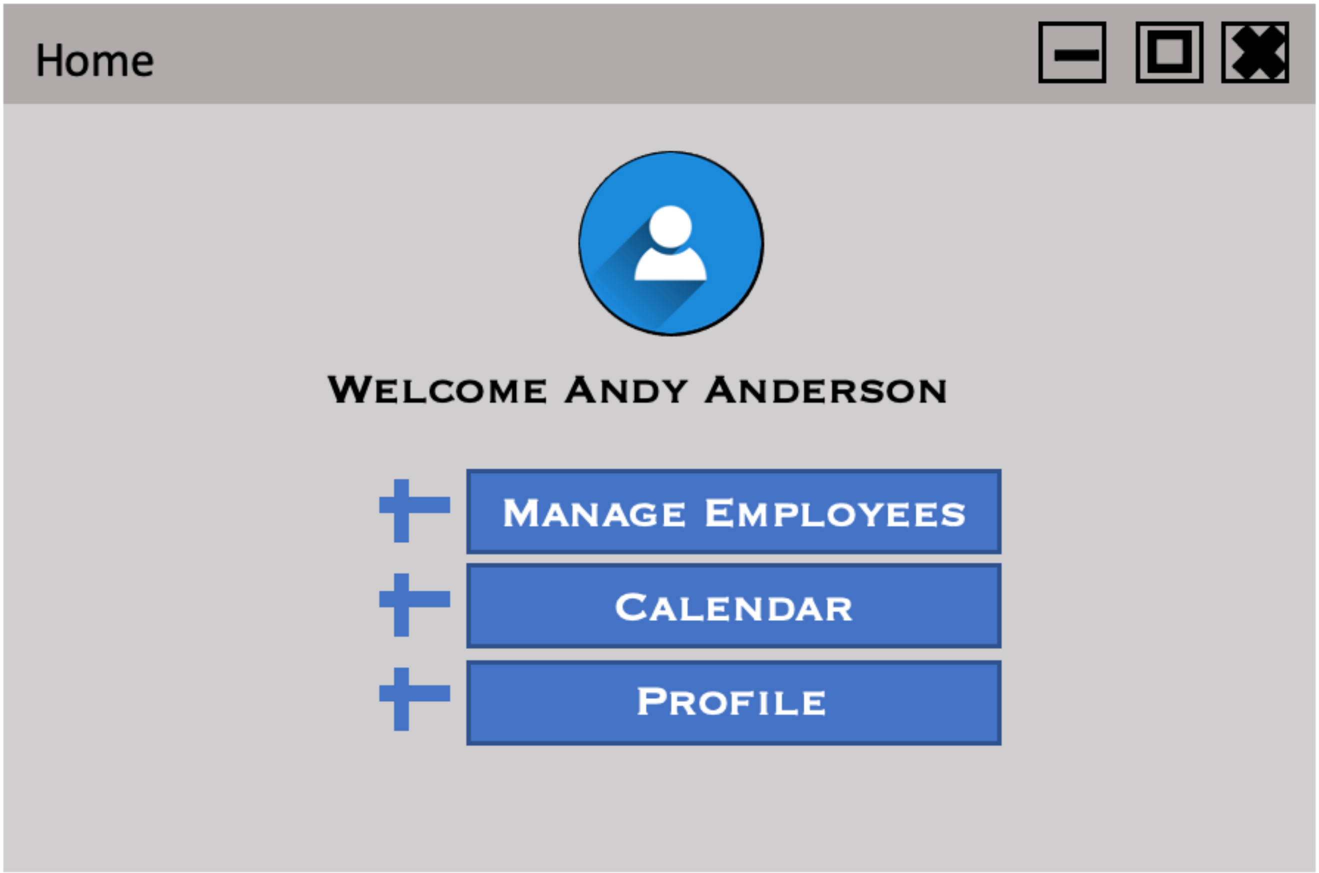
Username: AA001

User Email: AA001@GMAIL.COM

Password: ********

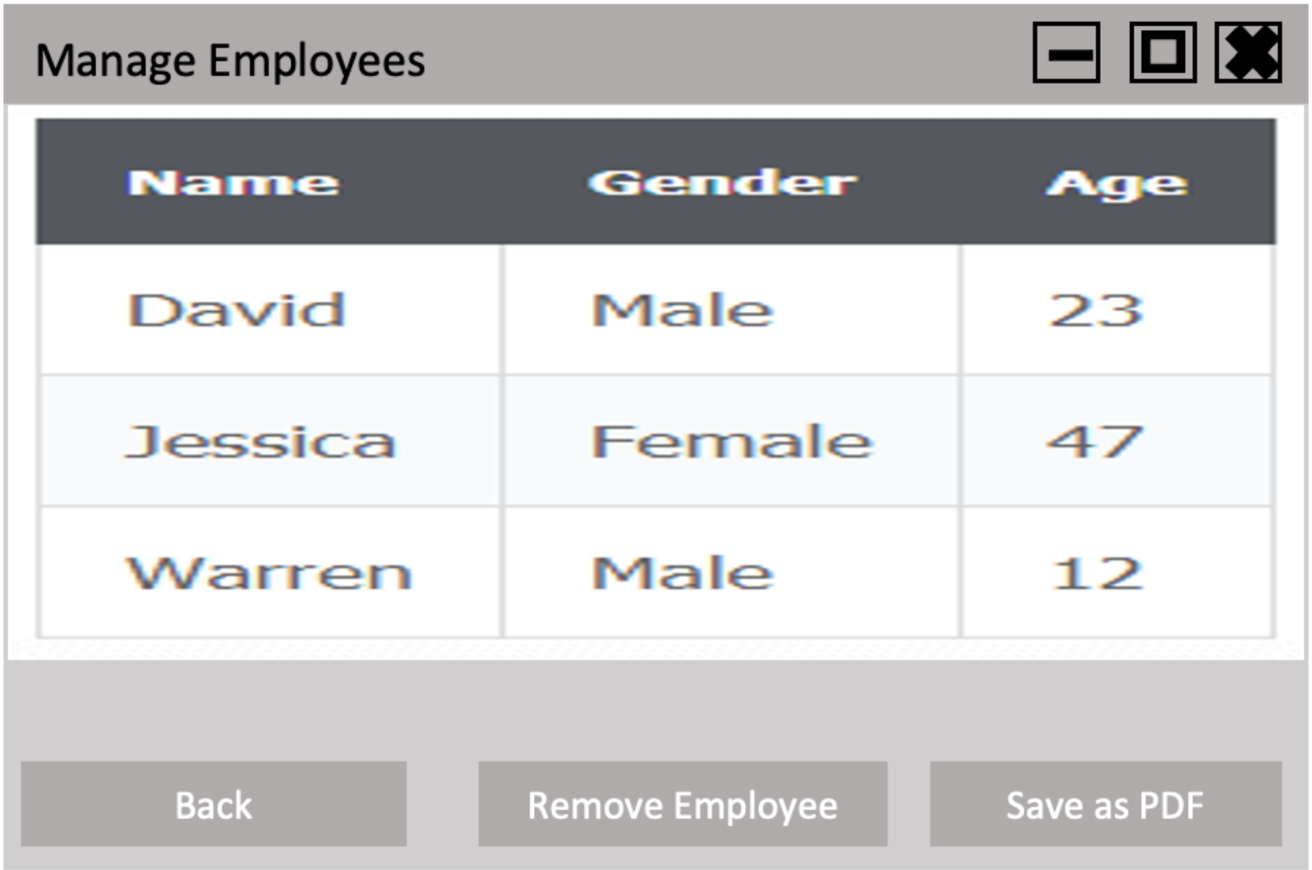
Confirm Password: ********

Figure 20 - Edit Profile Page Design



Welcome Andy Anderson

Figure 21 - Owner Home Page Design



Name	Gender	Age
David	Male	23
Jessica	Female	47
Warren	Male	12

Figure 22 - Manage Employees Page Design

SQL Table Design: Databases and tables to be created before the program
A database schema called easyaccounts_db will be created in MySQL with the following tables.

Field Name	Data Type	Description
Username	VARCHAR(10)	Unique string used to identify each user.
First_Name	VARCHAR(10)	Used to store the first name of the user.
Last_Name	VARCHAR(20)	Used to store the last name of the user.
Designation	VARCHAR(15)	Used to store the role/designation of the user.
Password	VARCHAR(20)	String used to login to the program.
Gender	VARCHAR(6)	Used to store the gender of the user.
Date_Of_Birth	DATE	Used to store the date of birth of the user.
Email_Address	VARCHAR(30)	Used to store the email address of the user.

Security_Question_1	VARCHAR(100)	Used to store the first security question of the user.
Security_Answer_1	VARCHAR(45)	Used to store the first security question’s answer of the user.
Security_Question_2	VARCHAR(100)	Used to store the second security question of the user.
Security_Answer_2	VARCHAR(45)	Used to store the second security question’s answer of the user.
Profile_Pic	BLOB	Used to store the profile picture uploaded by the user in binary format.

Table 1: ea_users

Field Name	**Data Type**	**Description**
id	INT	Unique identifier for each event.
Username	VARCHAR(10)	Unique string used to separate the events for each user.
EventName	VARCHAR(30)	String used to store the name of any event input by a user.
EventDate	DATE	Used to store the date of an event entered by a user.
EventDescription	VARCHAR(50)	Used to store the description of an event entered by a user.
EventType	VARCHAR(20)	Used to store the type of event entered by a user.

Table 2: userevents

Field Name	**Data Type**	**Description**
id	INT	Unique identifier for each event.
Username	VARCHAR(10)	Unique string used to separate the elements for each user.
ElementName	VARCHAR(45)	String used to store the name of any element input by a user.
ElementValue	DOUBLE	Used to store the number entered by the user for the respective element.
ElementType	VARCHAR(30)	String used to separate the elements for the different tables.
ElementTable	VARCHAR(26)	String used to separate the elements for the different financial tables.

Table 3: user_tables

Test Plan

Action to be tested	Test Method
Unique usernames so that there is no confusion and overwriting of another user's data.	Creating an account with a username and then creating another account with the same username to see if an error message pops up.
Ensuring that the password is case sensitive to provide more security.	Filling in a password with random letters in uppercase and lowercase to see if an error message is given.
When registering, check the strength of the entered password to ensure it meets the criteria.	Add 25 to a total based on whether the entered password has capital letters, common letters, digits, and symbols. If the total is equal to 100 then it will accept the password. If it is less than 100 then an error message will be given.
When resetting passwords, it should ask security questions which the user filled in when registering before allowing the user to reset the password to avoid tampering with another user's account.	Using the JComboBox and JTextField features on IntelliJ, the answers in the JTextFields will be matched with the security questions in the JComboBoxes and then compared to the data stored in the MySQL to check whether or not the data entered is correct.
Separate classes for each designation in the business and for each aspect of the program, such as the calendar or profile page.	Using the JFrame and JPanel features of IntelliJ, a separate JPanel will be created for each designation when they log in. A separate JPanel will be created for each aspect of the program such as the Calendar or Profile Page.
Log the user into the appropriate homepage based on the designation with which they are saved in MySQL.	Logging in with different accounts which have different designations to check whether it actually logs the user into the appropriate home page.
Program provides a user-friendly interface.	Testing the program with some of my peers to get feedback on the usability and functionality. Based on their feedback, the program's interface will be improved.
Program allows users to edit their profile information.	Using features of IntelliJ such as JTextFields, the program will set the text in the JTextFields as the user's data from the different columns in the database and the user will be able to change whatever they want. The program will then save the changes in the database.
Program allows users to	Using the JFileChooser feature in IntelliJ, a window will open when

change their profile picture.	the “Edit Profile” button is clicked which will allow the user to pick a picture they have saved in their pictures folder on their laptop.
Program automatically sets an empty profile picture icon for a newly registered user.	When registering a user, The program will automatically add the path of the icon which is stored in the program files as the default value for the profile picture in the database.
Program automatically resizes the window to fit the respective interfaces.	Using the .setSize feature in IntelliJ, The window will be resized to the appropriate size depending on which button is clicked.
Program allows users to view events.	Using the JTable and DefaultTableModel features in IntelliJ, a DefaultTableModel will be created using a While loop to retrieve all of the events for the user from the database.
Program allows users to add events.	Using the UPDATE statement syntax in MySQL, JTable feature, and DefaultTableModel feature, the program will create a connection to the database which will be used to add new events into the database and the appropriate table.
Program allows users to remove events.	Using the REMOVE statement syntax in MySQL, JTable feature, and DefaultTableModel feature, the program will create a connection with the database which will be used alongside the REMOVE statement to delete the record which was selected from the JTable in the program.
Program allows users to save events listed as a PDF.	Using the .print() method in IntelliJ along with error handling, the program will prompt the user and ask if they wish to save as a PDF and where they would like to save the PDF to.
Program will save events in a table.	Using the JTable and DefaultTableModel features and the .setModel() and .next() methods in IntelliJ, the program will connect to the database and use the .next() method to retrieve all of the events from the database and arrange them into a DefaultTableModel and then the .setModel() method will be use to set the model of the JTable as the DefaultTableModel which was created.
Program will update events in the table when the “remove event” button is clicked.	Using the JTable and DefaultTableModel features and the .setModel() and .next() methods in IntelliJ, the program will connect to the database and use the .next() method to retrieve all of the events from the database after an event has been removed and arrange them into a DefaultTableModel and then the .setModel() method will be use to set the model of the JTable as the DefaultTableModel which was created.
Program will change the financial table when the respective button is clicked.	Using methods such as .isVisible() and .setVisible() in IntelliJ along with an if statement and features in IntelliJ such as JPanels, JButtons and Action Listeners, the JPanels will be set invisible and

	visible depending on which button has been clicked.Using features of IntelliJ such as JButtons, JPanels, JTables, DefaultTableModels and Action Listeners along with methods such as .setVisible(), the program will set the respective JPanel visible and the other JPanel invisible when the respective button has been clicked. Once the JPanel has been set visible, the program will create a connection with the database and retrieve all of the data for the respective tables.
Program will add a new row to the financial table when the “Enter” key is pressed on the keyboard.	Using features in IntelliJ such as Focus Listeners, Key Listeners, DefaultTableModels and JTables along with the .setModel() method, a new row will be added to the DefaultTableModel of the JTable which has focus and then the DefaultTableModel will be applied to the JTable using the .setModel() method.Using Features such as JTables, DefaultTableModels, Focus Listeners and Key Listeners along with methods such as .setModel(), .addRow() and .getModel(), the program will get the DefaultTableModel of the JTable and add a row and then return the TableModel and set as the model of the JTable when the “Enter” key has been pressed. Once the new data has been entered, the data will be updated in the database.
Program will remove the selected row from the financial table when the “Delete” key is pressed on the keyboard.	Using features in IntelliJ such as Focus Listeners, Key Listeners, DefaultTableModels and JTables along with the .setModel() method, the selected row will be removed from the DefaultTableModel of the JTable which has focus and then the DefaultTableModel will be applied to the JTable using the .setModel() method. Once that has been done, the program will create a connection to the database and update the database.Using Features such as JTables, DefaultTableModels, Focus Listeners and Key Listeners along with methods such as .setModel(), .getModel(), .removeRow() and .getSelectedRow(), the program will get the DefaultTableModel of the JTable and get the selected row and the data in it when the“Delete” key has been pressed. The program will then delete the record with that information in the database. Once the data in the database has been updated, the data will be retrieved and then set in the DefaultTableModel. The TableModel will then be set as the model of the JTable.
Program will allow the owner to manage employees.	Using the JTable and DefaultTableModel features and the .setModel() and .next() methods in IntelliJ, the program will connect to the database and use the .next() method to retrieve all of the employees other than owners from the database and arrange them into a DefaultTableModel and then the .setModel() method will be use to set the model of the JTable as the DefaultTableModel which was created. Using features in IntelliJ such as Focus Listeners, DefaultTableModels and JTables along with

	the .setModel() method, the selected row will be removed from the DefaultTableModel of the JTable which has focus and then the DefaultTableModel will be applied to the JTable using the .setModel() method. Once that has been done, the program will create a connection to the database and update the database.Using the JTable and DefaultTableModel features in IntelliJ, a DefaultTableModel will be created using a While loop to retrieve all of the employees other than owners from the database. Using the REMOVE statement syntax in MySQL, JTable feature, and DefaultTableModel feature, the program will create a connection with the database which will be used alongside the REMOVE statement to delete the record which was selected from the JTable in the program.
Program will allow users to save the events table, the employees table, and the separate financial tables as PDFs.	Using features such as JButtons and JTables along with methods in IntelliJ such as .print() and error handling such as try and catch methods, the program will print the respective JTable as a PDF depending on the button which has been clicked.