**MESS MANAGEMENT SYSTEM**



**Session 2023 - 2027**

**Submitted by:**

Sumaira Hafeez 2023-CS-01

**Supervised by:**

Sir Dr. Awais Hassan

**Course:**

CSC-102 Programming Fundamentals

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan.**

Table of Contents

1.

[ **Short Description of your project** 3](#_Toc153821373)

[ **Users of Application (minimum 2 users for your project)** 3](#_Toc153821374)

[ **Functional Requirements** 3](#_Toc153821375)

[ **Wireframes** 5](#_Toc153821376)

[ **Data Structures (Parallel Arrays)** 17](#_Toc153821377)

[ **Function Prototypes** 18](#_Toc153821378)

[ **Functions Working Flow** 21](#_Toc153821379)

[ **Complete Code of the Business Application** 22](#_Toc153821380)

[ **Weakness in the Business Application** 59](#_Toc153821381)

[ **Future Directions** 60](#_Toc153821382)

**MESS MANAGEMENT SYSTEM**

# **Short Description of your project**

The business application is concerned with the problems faced by hostellite student and their messing problems. It will improve the dynamics and use of computer science in our daily life by limiting the use of paper work and man power it will improve the digital systems for hostel systems. At the end of project, I would like to deliver the best results of this application that will assist both the mess manager and the students.

# **Users of Application (2 users for my project)**

* + This application is specifically for the university students who stay away from their homes for the sake of study. It includes two users:
  + Student: The hostel students.
  + Mess Manager: The person who is responsible for providing the students with messing service.

# **Functional Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| User Story ID | *As a* | *I want to perform* | *So that I can* |
| **1** | Student | Add Mess Attendence | Students will be able to off days. |
|  |  | Update Mess attendence | Students can update attendence in case of emergency |
|  |  | View Attendence | Students can view their attendene |
|  |  | Check Bill | Students will be able to check their mess bill |
|  |  | Vote for Menu | They can decide what they want to eat |
|  |  | Complaint | They can complaint if they have any problem |
|  |  | Edit Account | Students can edit the account |
|  |  | Log Out | Students can log out of their accounts |
| **2** | Manager | Menu | Manager will be able to create and view menu |
|  |  | .Add stock | Manager will be able to add stock he need for meals. |
|  |  | View Stock | Manager can view the stock |
|  |  | View Votes | Manager can view votes made by students |
|  |  | Billing | Manager can make bills |
|  |  | Profit | Manager can calculate its profit |
|  |  | View complaints | Manager can view complaints made by students |
|  |  | Log Out | Manager can log out of their accounts |

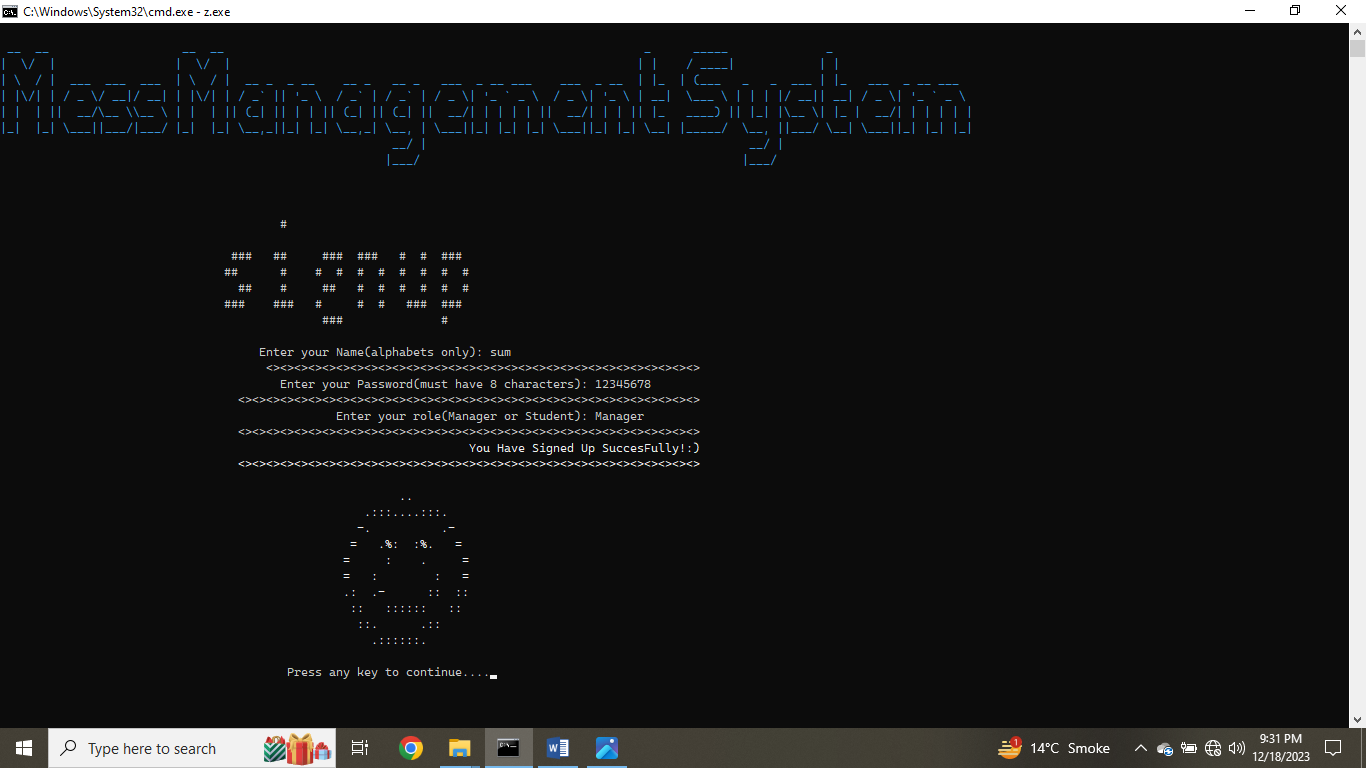
## **Wireframes**



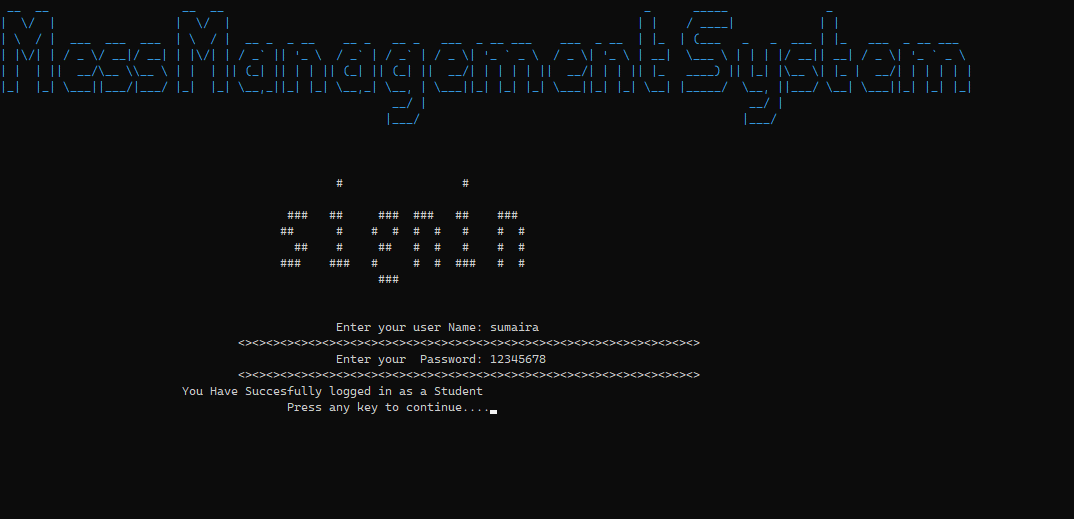
**Figure 1: LogIn Page**



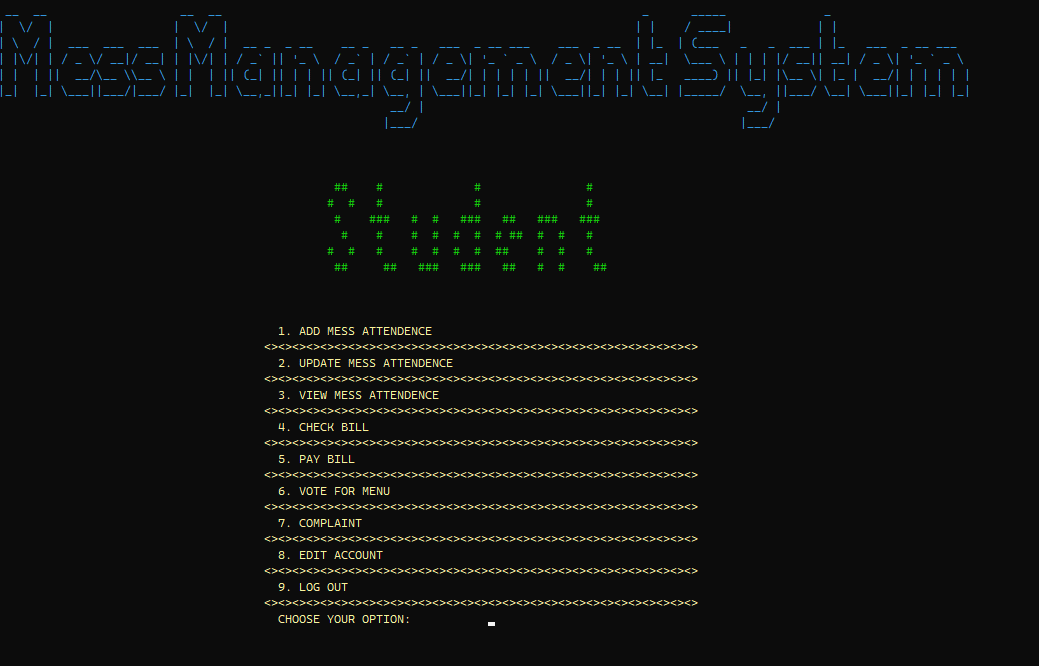
**Figure 2: SignUp Page(as Student)**



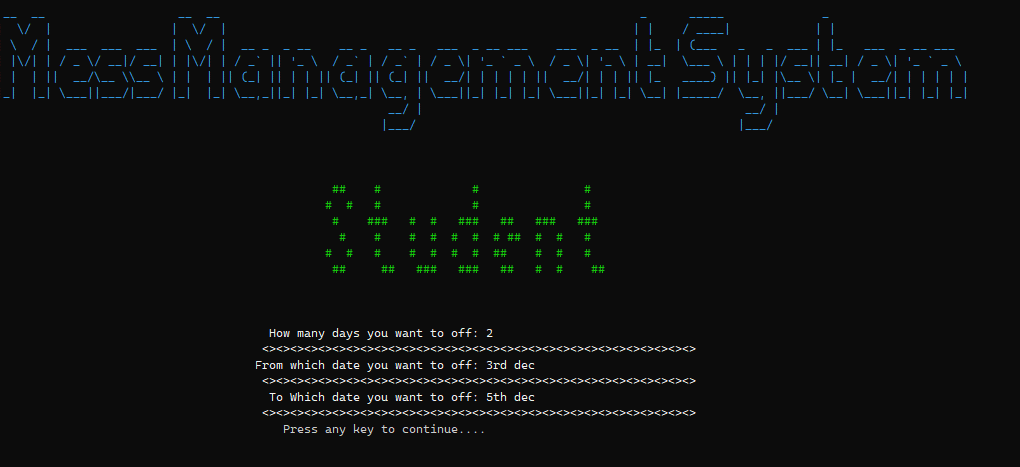
**Figure 3: SignUp Page(as Manager)**



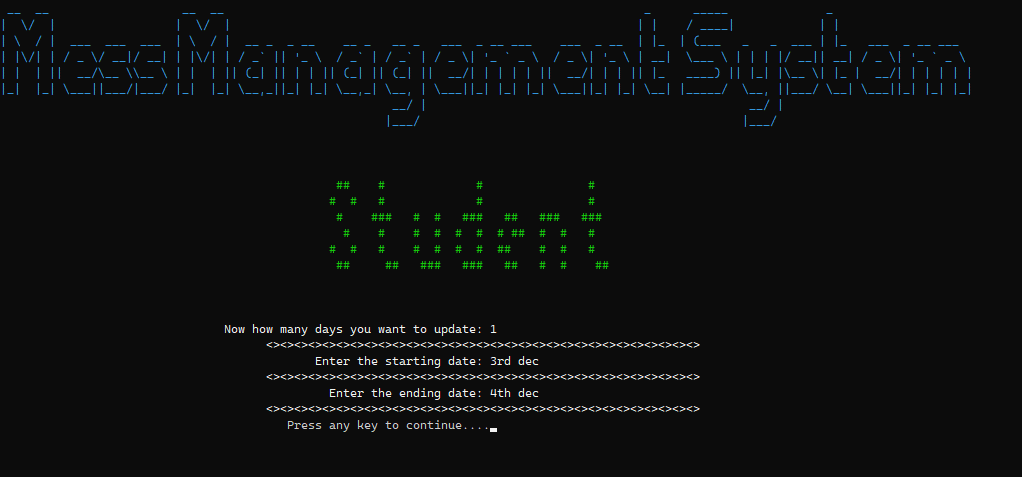
**Figure 4: SignIn Page**



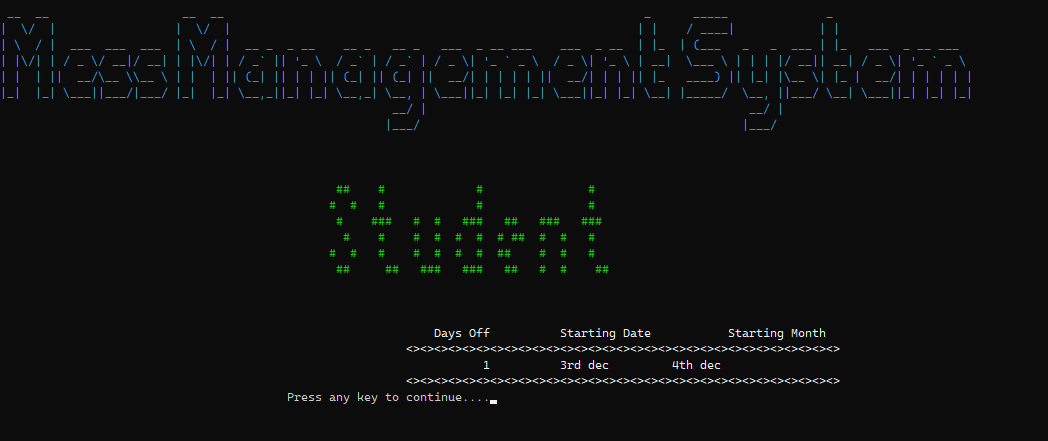
**Figure 5: Student Page**



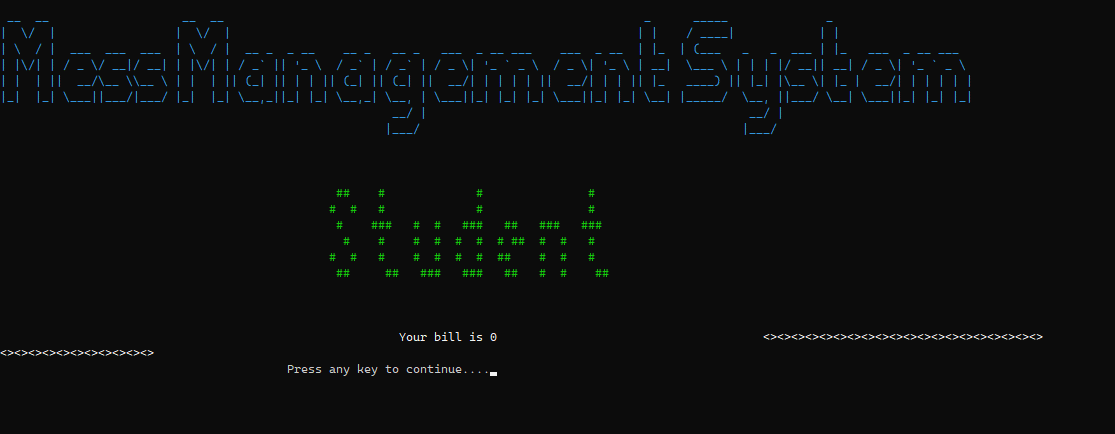
**Figure 6: Add Attendence**



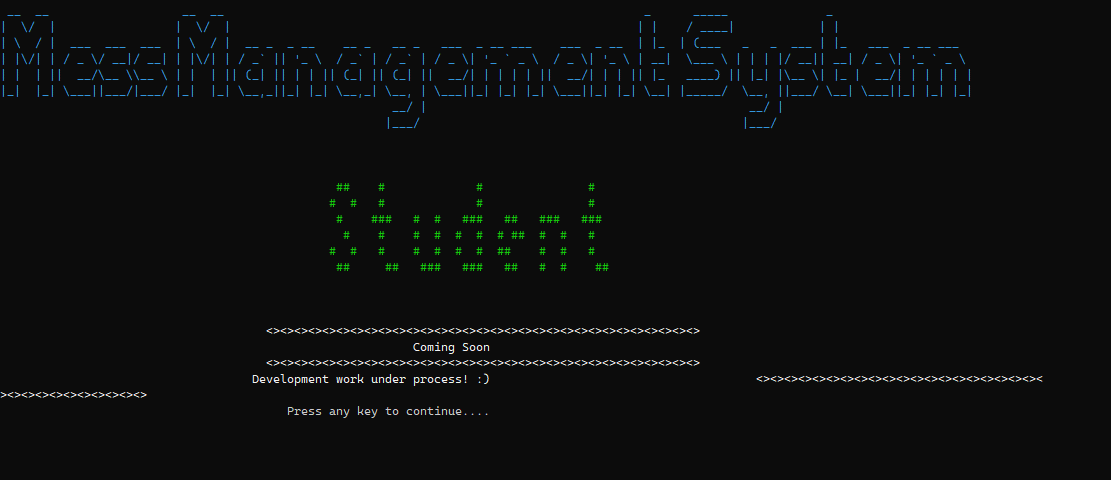
**Figure 7: Update Attendence**



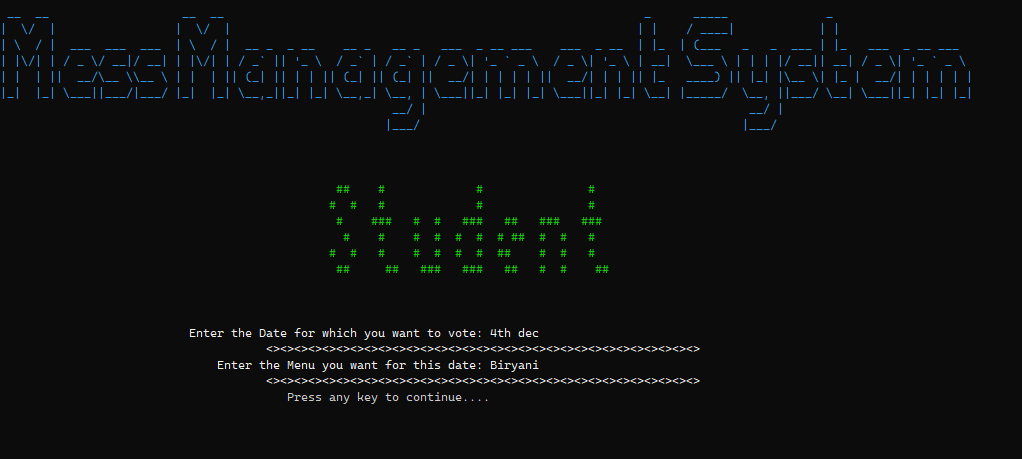
**Figure 8: View Attendence**



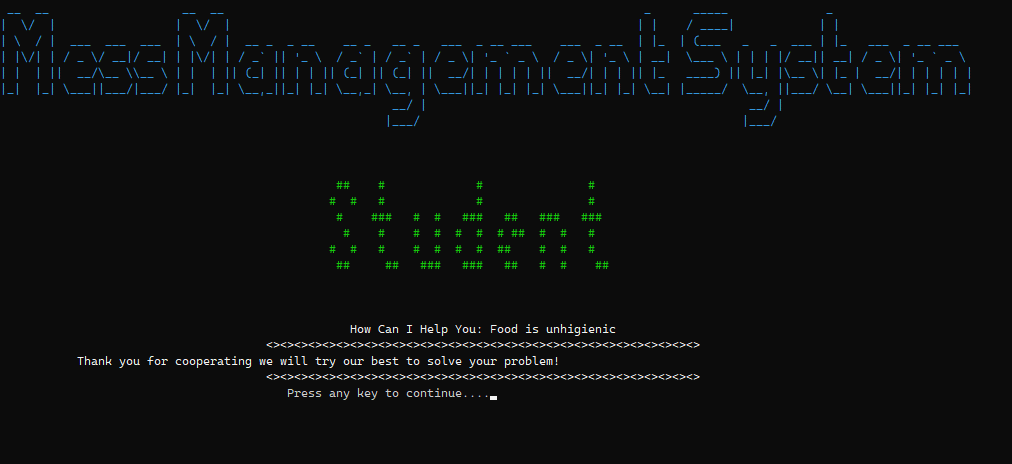
**Figure 9: Check Bill**



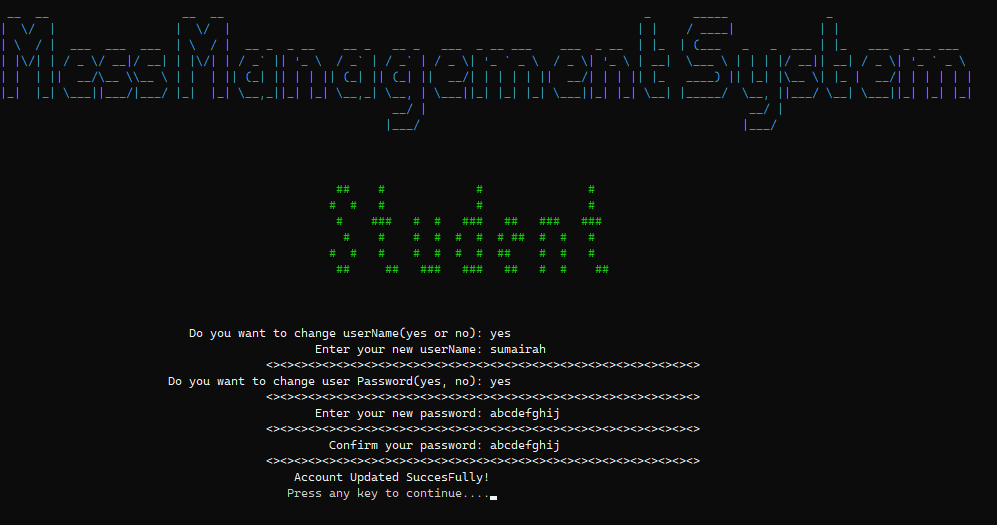
**Figure 10: Pay Bill**



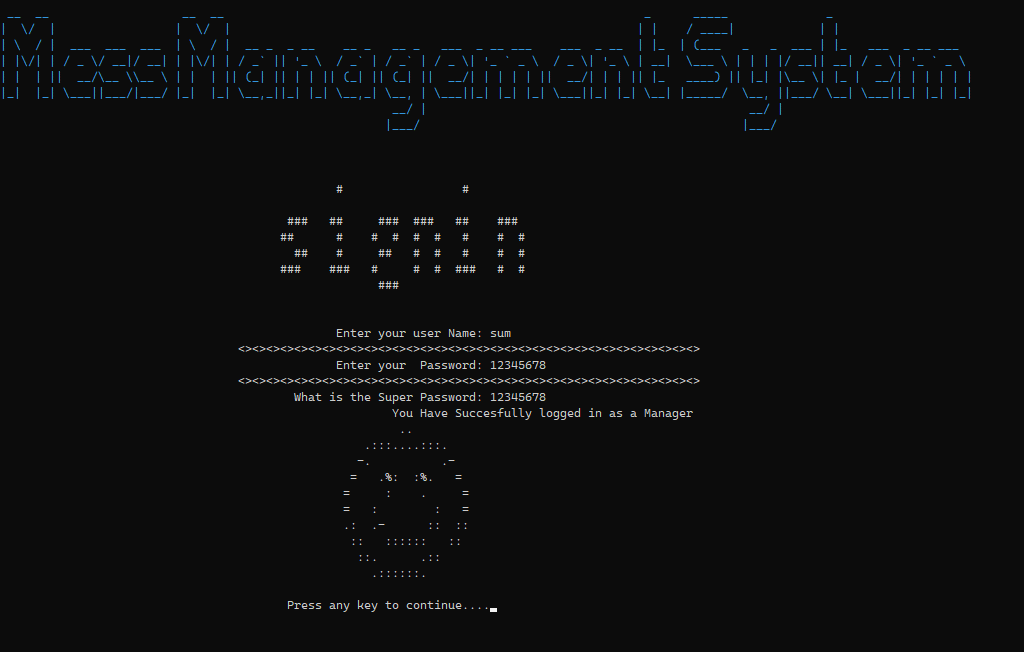
**Figure 11: Vote For Menu**



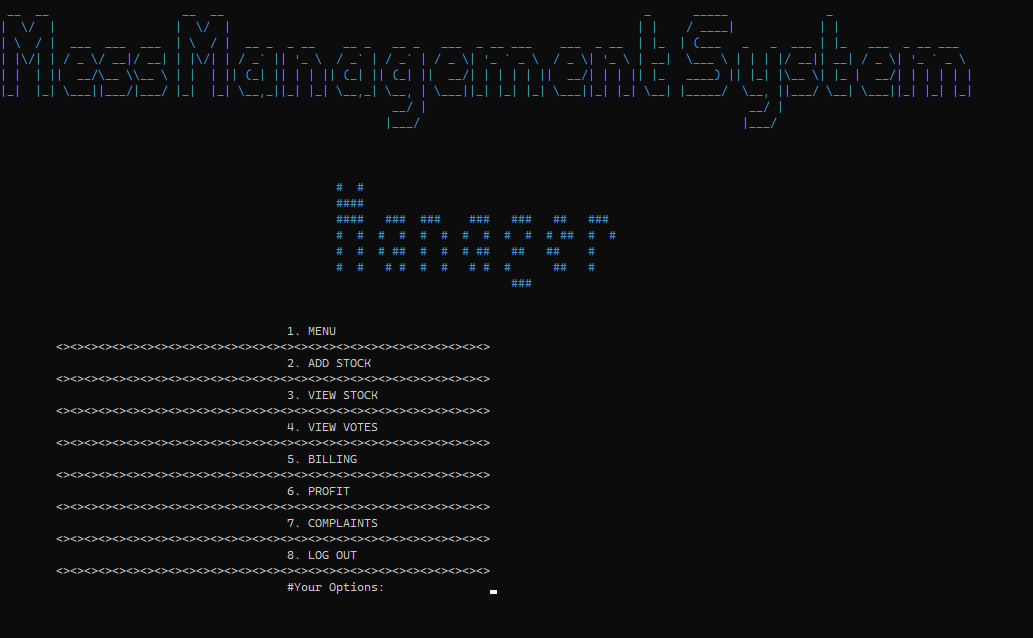
**Figure 12: Complaints**



**Figure 13: Edit Account**



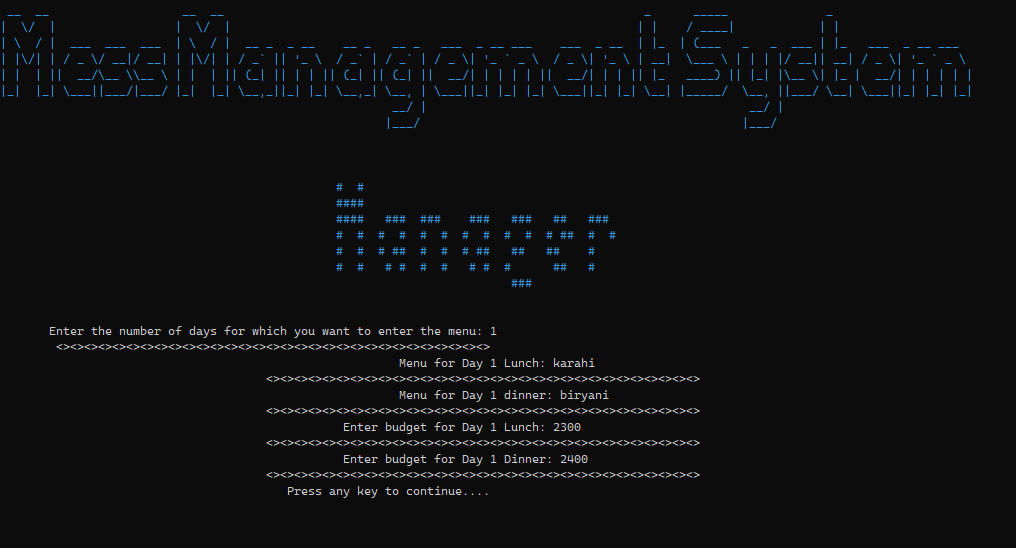
**Figure 14: Sign In as a Manager**



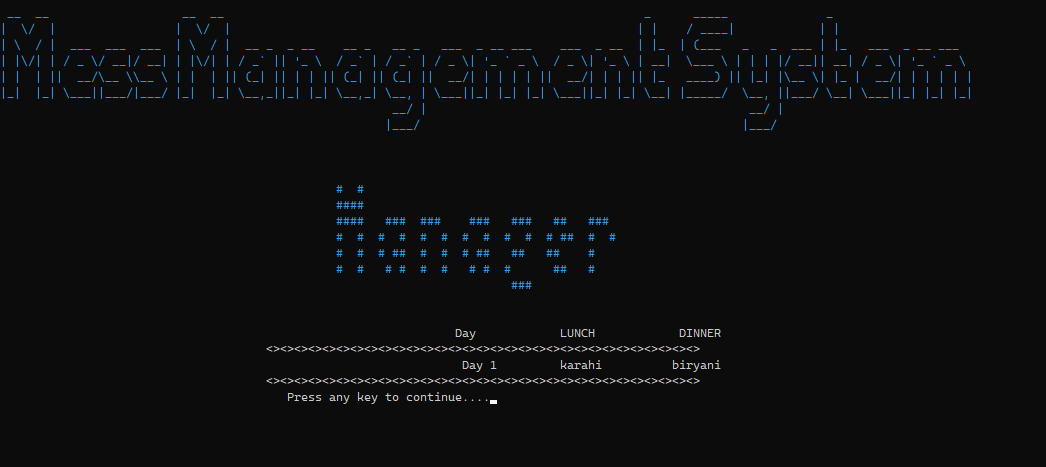
**Figure 15: Manager Page**



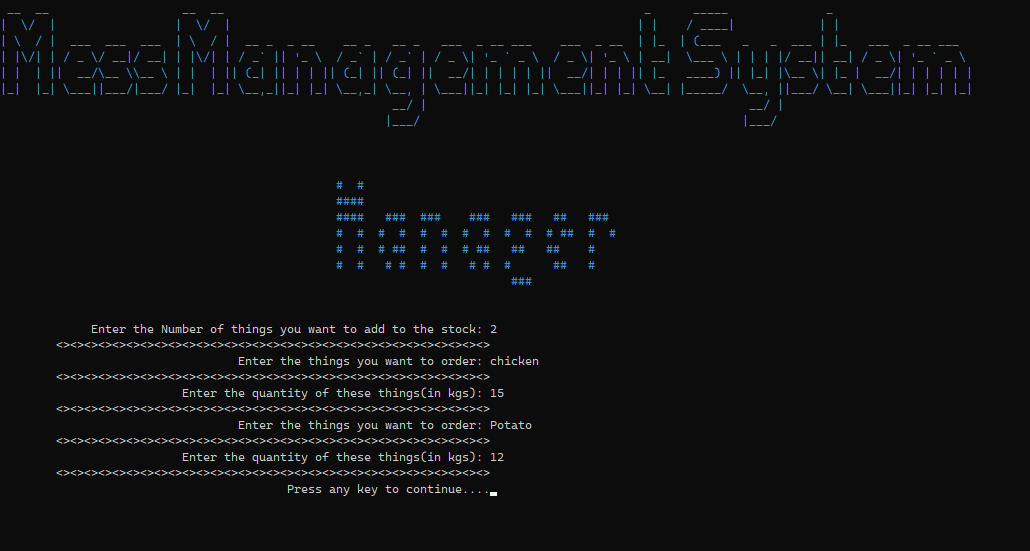
**Figure 16: Menu Page**



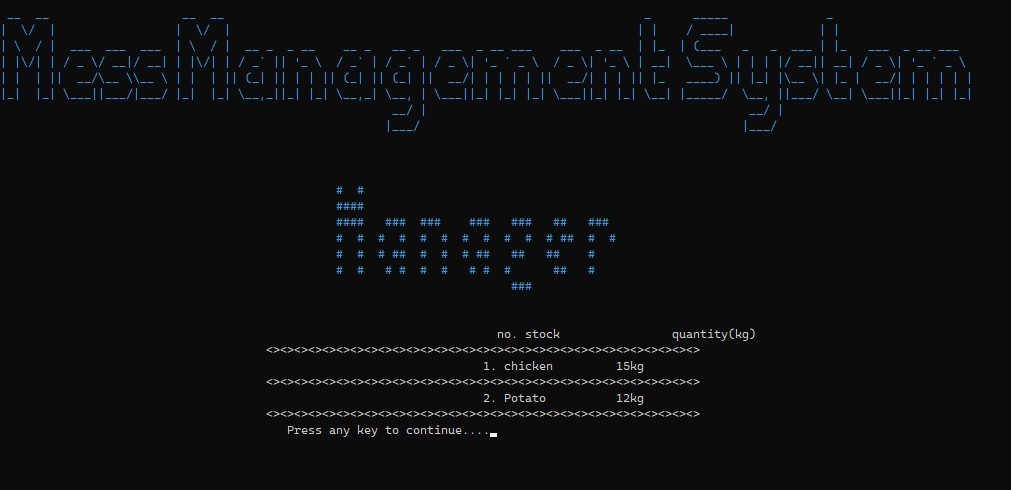
**Figure 17: Create Menu**



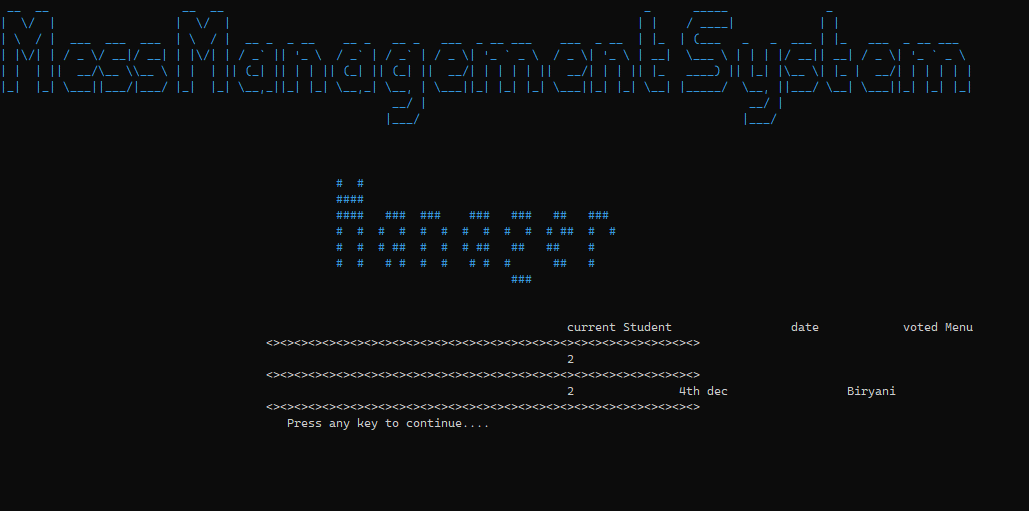
**Figure 18: View Menu**



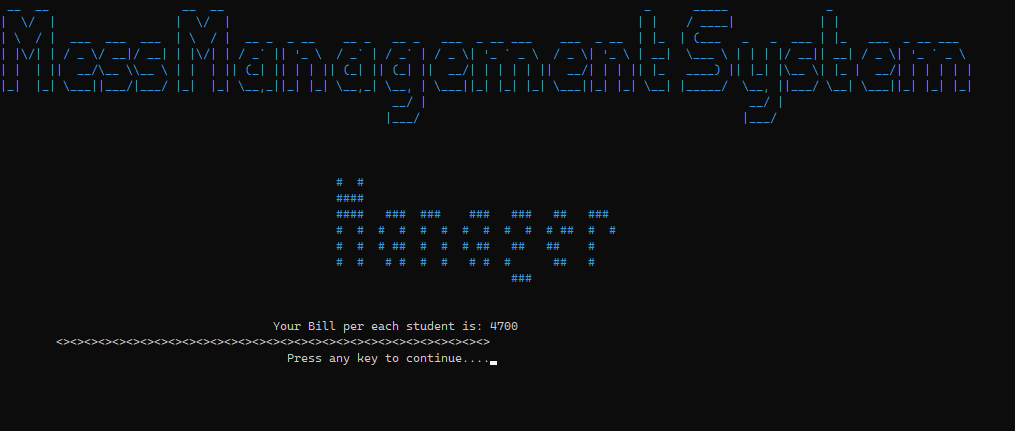
**Figure 19: Add Stock**



**Figure 20: View Stock**



**Figure 21: View Voted Menu**



**Figure 22: Billing**



**Figure 23: Profit**



**Figure 24: Check Complaints**



**Figure 25: Exit Program(with File Handling)**

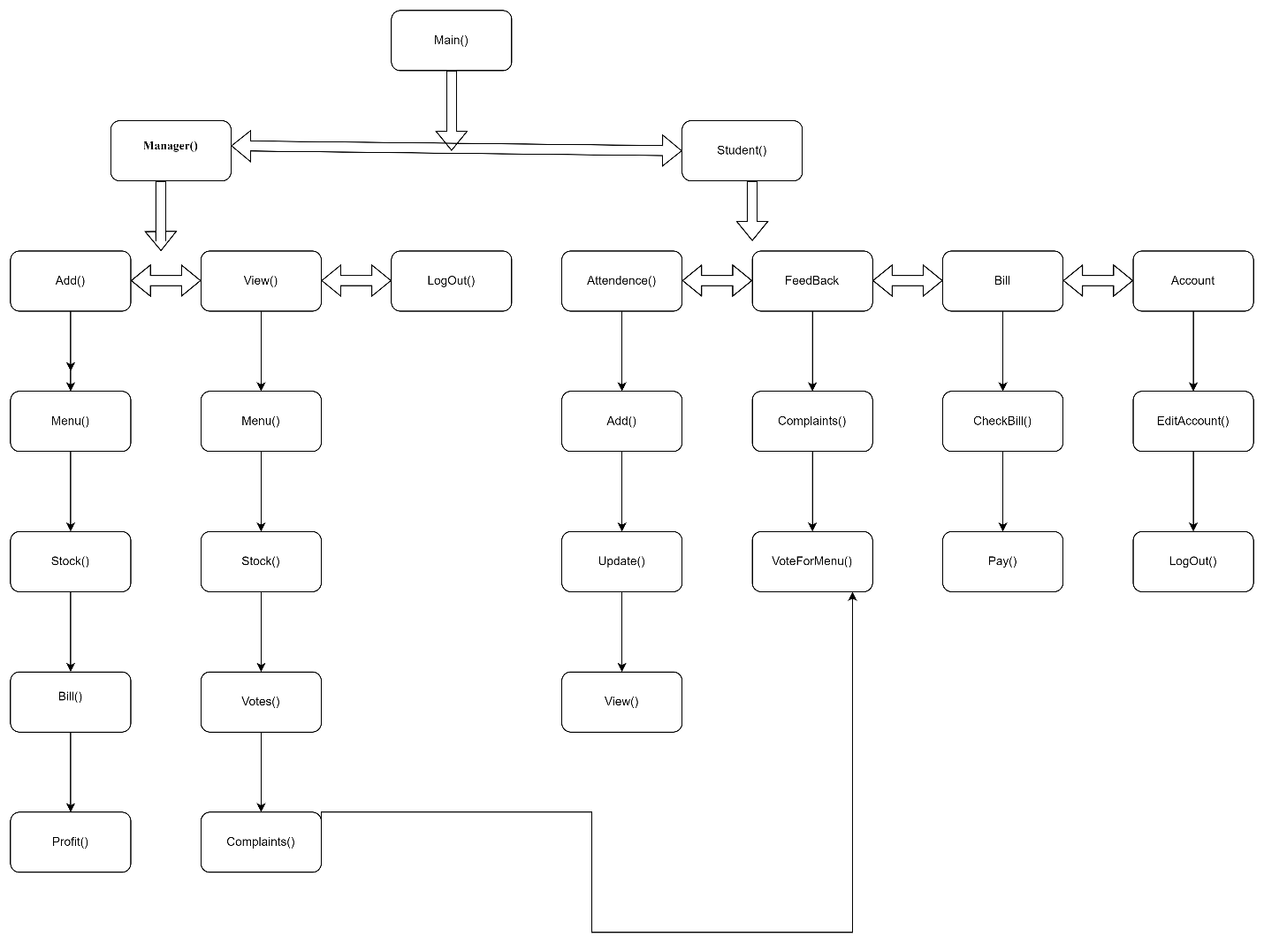
# **Data Structures (Parallel Arrays)**

* int studentIndex = 0;
* int userCount = 0;
* int userNum=500;
* int currentStudent;
* string userName[userNum];
* string userPassword[userNum];
* string userRole[userNum];
* string roomNumber[userNum];
* string offDate[300];
* string offMonth[300];
* string updateDays[100];
* string updateMonths[100];
* string toUpdateDays[100];
* string toUpdateMonths[100];
* int bill, profit;
* string fileName = "signIn.txt";
* string date[100],
* votedMenu[100];
* string complaintBox[100];
* string room;
* string name, password, role;
* int days;
* string offDates;
* int option;
* string inputName, inputPassword;
* int  budget, things;
* string stock[things];
* int quantity[things];
* string menulunch[days];
* string menuDinner[days];
* int budgetlunch[days];
* int budgetDinner[days];
* string lunchbudget, dinnerbudget;
* string thing;
* string quant;
* string day;
* string startDate;
* string endDate;
* string choice;
* string password;
* string confirmPassword;
* string newUserName;

# **Function Prototypes**

* *//Headers//*
* void **header**();
* void **logInPageHeader**();
* void **happyFace**();
* void **sadFace**();
* void **signUpHeader**();
* void **signInHeader**();
* void **managerPage**();
* void **studentHeader**();
* *//logInPage//*
* int **loginPage**();
* *//signUp//*
* bool **signUpCredentials**(**string** &name, **string** &password, **string** &role, **string** userName[], **string** userPasword[], **string** userRole[], int &userCount, **string** roomNumber[], **string** &room, int &studentIndex);
* bool **signUp**(**string** &name, **string** &password, **string** &role, **string** userName[], **string** userPassword[],**string** userRole[], int &userCount);
* *//signIn//*
* bool **signInValidity**(**string** &InputName, **string** &InputPassword, **string** userName[], **string** userPassword[], int &userCount, int &currentStudent);
* **string** **signIn**(**string** &InputName, **string** &InputPassword, **string** userName[], **string** userPassword[], **string** userRole[], int &userCount);
* *//validations//*
* bool **nameValidity**(**string** &name);
* bool **checkValidity**(**string** &name, **string** userName[], int &userCount);
* bool **passwordValidity**(**string** &password);
* bool **checkRoomNumberValidity**(**string** &room, **string** roomNumber[], int &studentIndex);
* void **studentRegistration**(**string** name, **string** password, **string** role, int &studentIndex, **string** roomNumber[],**string** &room);
* void **whatIfRoomIsValid**(bool check, int &studentIndex);
* bool **checkManager**();
* bool **checkStudentIndex**(int &studentIndex);
* bool **checkNewUserName**(**string** &newUserName, **string** userName[], int &currentStudent);
* bool **intValidations**(**string** code);
* *//setColor//*
* **string** **setcolor**(unsigned short color);
* *//press Me To Move On//*
* void **breakTime**();
* *//managerEnd//*
* int **ManagerfirstPage**();
* int **callingManagerMenu**(int days, int budget, int things, int studentIndex, int bill, int profit, **string** complaintBox[], int &currentStudent, **string** date[], **string** votedMenu[]);
* int **Managermenu**();
* void **functionalityOfManager**(int &days, int &budget, int &things, int &studentIndex, int &bill, int &profit, **string** complaintBox[], int &currentStudent, **string** date[], **string** votedMenu[]);
* void **create**(**string** menulunch[], **string** menuDinner[], int budgetlunch[], int budgetDinner[], int days, int &budget, int things, int studentIndex, int &bill, int &profit, **string** complaintBox[], int &currentStudent, **string** date[], **string** VotedMenu[]);
* void **viewMenu**(**string** menulunch[], **string** menuDinner[], int days, int budget, int things, int studentIndex, int bill, int profit, **string** complaintBox[], int &currentStudent, **string** date[], **string** votedMenu[]);
* void **callAddStock**(int &things);
* void **addStock**(**string** stock[], int quantity[], int &things, int bill, int profit);
* void **viewStock**(**string** stock[],int quantity[], int things);
* int **makeBill**(int &budget, int studentIndex, int &bill, int profit);
* void **viewStudentVotedMenu**(**string** date[], **string** votedMenu[], int currentStudent);
* void **viewComplaints**(**string** complaintBox[], int studentIndex);
* int **calculateProfit**(int bill, int profit);
* *//student End//*
* int **functionalityOfStudent**(**string** offDate[], **string** offMonth[],int &days, int currentStudent,int &bill, **string** date[], **string** votedMenu[], **string** complaintBox[], **string** userName[], **string** userPassword[]);
* int **studentFirstPage**();
* void **addAttendence**(**string** offDate[], **string** offMonth[],int days, int currentStudent, int bill);
* void **updateAttendence**(**string** offDates[], **string** offMonths[], int &days, int currentStudent, int bill);
* void **viewAttendence**(**string** offDates[], **string** offMonths[], int days, int currentStudent, int bill);
* int **checkBill**(int &bill, **string** offDate[], **string** offMonth[], int days, int currentStudent);
* void **studentVotedMenu**(**string** date[], **string** votedMenu[], int curentStudent);
* void **editStudentAccount**(int currentStudent, **string** userName[], **string** userPassword[]);
* void **studentsComplaint**(**string** complaintBox[], int currentStudent);
* void **editUserPassword**(**string** &newUserName, **string** userName[], int currentStudent, **string** userPassword[], **string** choice, **string** &confirmPassword, **string** &password);
* void **editUserName**(**string** &newUserName, **string** userName[], int currentStudent, **string** userPassword[], **string** choice);
* *//File Handlings//*
* *//STORE FUNCTIONS//*
* void **storeSignIn**(**string** userName[], **string** userPassword[], **string** userRole[], **string** fileName, int &userCount);
* **string** **getField**(**string** record, int order);
* *//LOAD FUNCTIONS//*
* void **loadSignIn**(**string** userName[], **string** userPassword[], **string** userRole[], **string** fileName, int &userCount);

# **Functions Working Flow**



%20id%3D%2254%22%20value%3D%22%22%20style%3D%22edgeStyle%3DorthogonalEdgeStyle%3Brounded%3D0%3BorthogonalLoop%3D1%3BjettySize%3Dauto%3Bhtml%3D1%3B%22%20edge%3D%221%22%20source%3D%2255%22%20target%3D%2256%22%20parent%3D%221%22%3E%3CmxGeometry%20relative%3D%221%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%2255%22%20value%3D%22%22%20style%3D%22rounded%3D1%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%221090%22%20y%3D%22420%22%20width%3D%22120%22%20height%3D%2260%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3CmxCell%20id%3D%2256%22%20value%3D22%20style%3D%22rounded%3D1%3BwhiteSpace%3Dwrap%3Bhtml%3D1%3B%22%20vertex%3D%221%22%20parent%3D%221%22%3E%3CmxGeometry%20x%3D%221090%22%20y%3D%22560%22%20width%3D%22120%22%20height%3D%2260%22%20as%3D%22geometry%22%2F%3E%3C%2FmxCell%3E%3C%2Froot%3E%3C%2FmxGraphModel%3E

# **Weakness in the Business Application**

My application has following weaknesses.

1. Some Validations are not working.
2. Some arrays are declared within functions.
3. File Handling is only done on sign in information.
4. GUI is not well.
5. A function of Pay Bill has not made yet.
6. Single responsibility is missing at some functions.
7. Some Functionalities of Manager will not work unless a student is registered first.
8. Even though the file handling include student registration it will not consider those because student index will return to zero after every exit.
9. Because of File Handling there is an empty box for vote Menu and Complaints and student Index is already set at total signins as in File Handling.

# **Future Directions**

I have following plans for the future of my application:

1. I will include arrow button instead of asking options.
2. I will include more functionalities
3. I will improve its User InterFace
4. I will make it more independent and easily amendable
5. I will do File Handling with all data.
6. I will counter all the weakness.