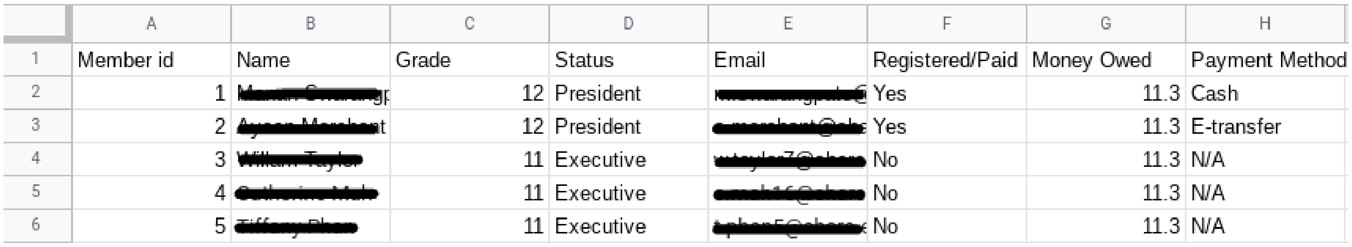
Criterion B: Design

File supplied by the client

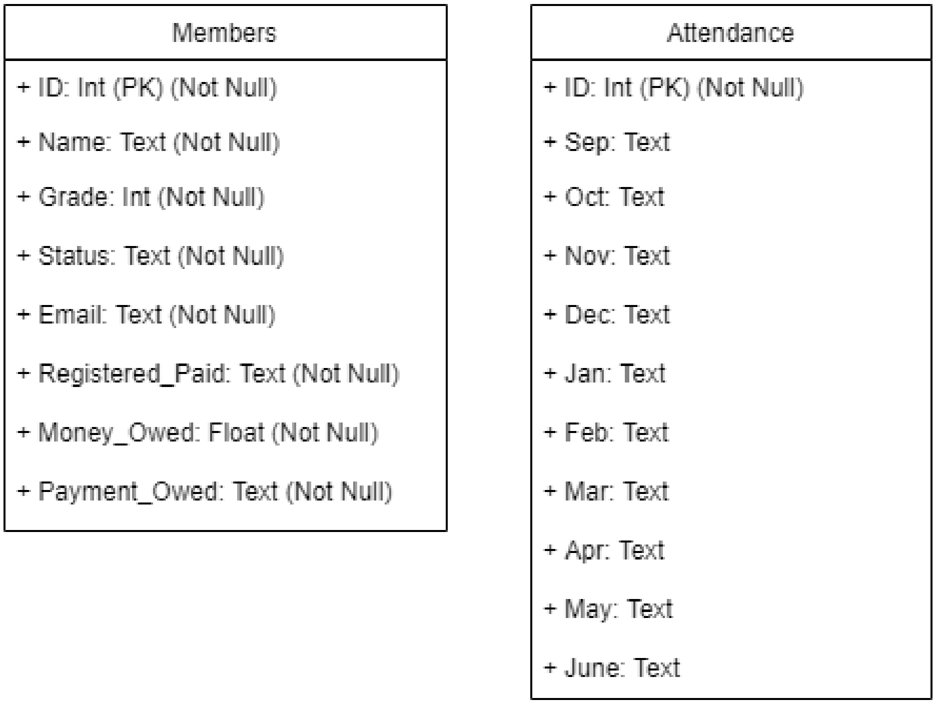
The client provided a Google Spreadsheet with information for 5 members. The spreadsheet was later converted into a CSV file and imported into the database(s).



	A	B	C	D	E	F	G	H
1	Member id	Name	Grade	Status	Email	Registered/Paid	Money Owed	Payment Method
2	1	[redacted]	12	President	[redacted]	Yes	11.3	Cash
3	2	[redacted]	12	President	[redacted]	Yes	11.3	E-transfer
4	3	[redacted]	11	Executive	[redacted]	No	11.3	N/A
5	4	[redacted]	11	Executive	[redacted]	No	11.3	N/A
6	5	[redacted]	11	Executive	[redacted]	No	11.3	N/A

Figure 1: Spreadsheet as supplied by the client (name/email has been cut out to secure member identity)

UML sheet displaying tables



Members

+ ID: Int (PK) (Not Null)
+ Name: Text (Not Null)
+ Grade: Int (Not Null)
+ Status: Text (Not Null)
+ Email: Text (Not Null)
+ Registered_Paid: Text (Not Null)
+ Money_Owed: Float (Not Null)
+ Payment_Owed: Text (Not Null)

Attendance

+ ID: Int (PK) (Not Null)
+ Sep: Text
+ Oct: Text
+ Nov: Text
+ Dec: Text
+ Jan: Text
+ Feb: Text
+ Mar: Text
+ Apr: Text
+ May: Text
+ June: Text

Figure 2: UML tables displaying the 2 different databases; members and attendance

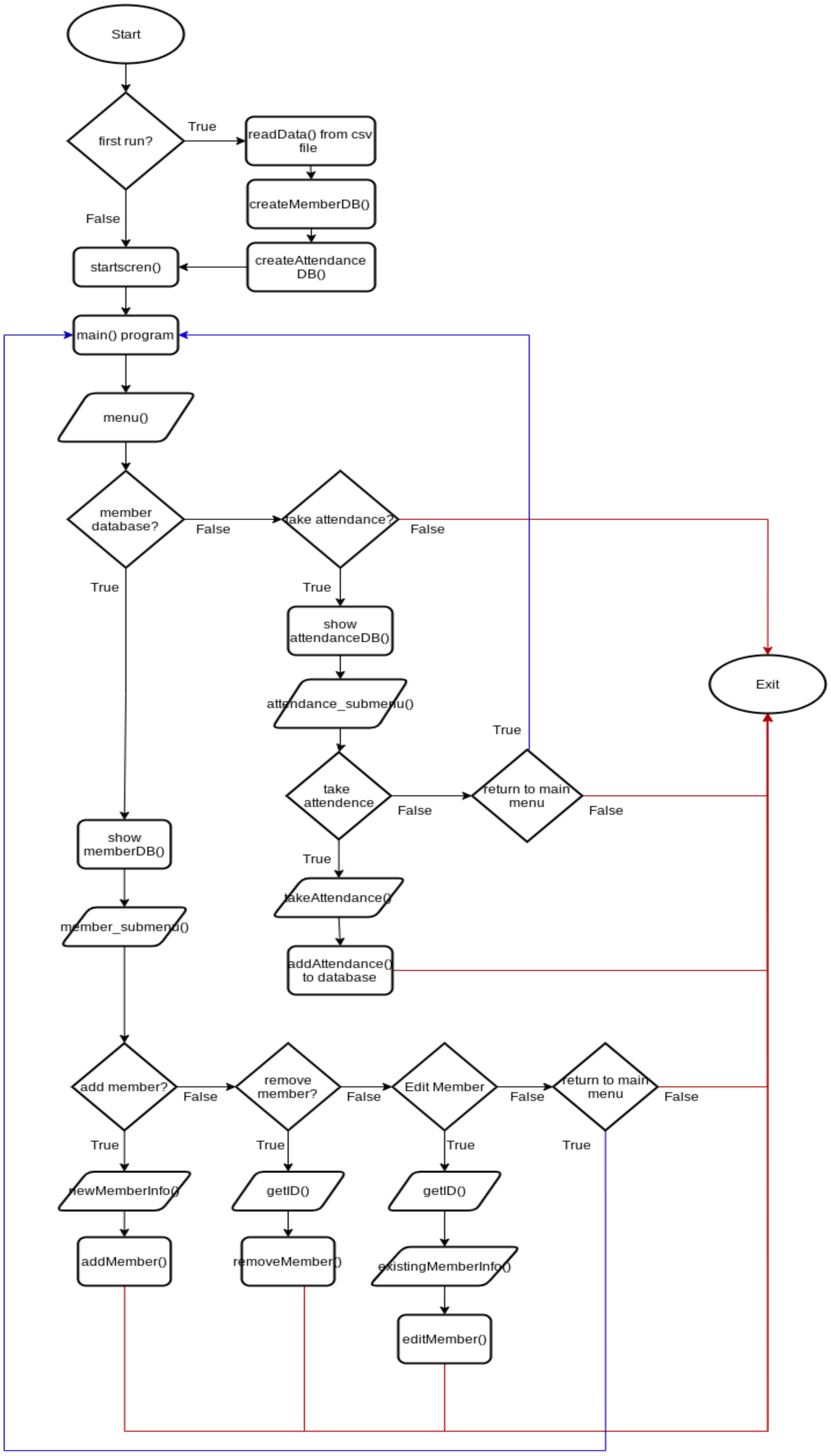

Start
first run?
True
readData() from csv file
createMemberDB()
createAttendance DB()
False
startscren()
main() program
menu()
member database?
False
take attendance?
False
True
True
show attendanceDB()
attendance_submenu()
Exit
True
take attendence
False
return to main menu
False
True
show memberDB()
takeAttendance()
addAttendance() to database
member_submenu()
add member?
False
remove member?
False
Edit Member
False
return to main menu
False
True
True
True
True
newMemberInfo()
getID()
getID()
addMember()
removeMember()
existingMemberInfo()
editMember()

Figure 3: Flowchart outlining the main algorithms of the program

Subprograms and Important Algorithms

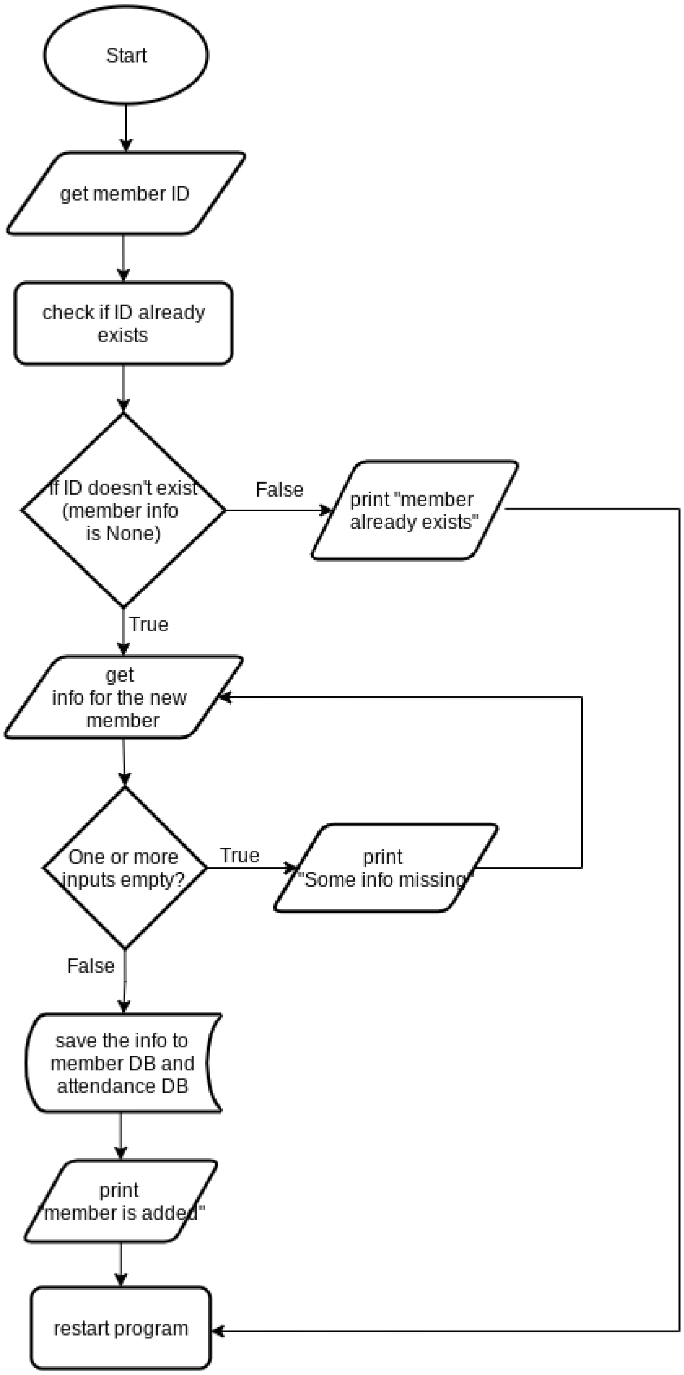

Start
get member ID
check if ID already exists
if ID doesn't exist (member info is None)
False
print "member already exists"
True
get info for the new member
One or more inputs empty?
True
print "Some info missing"
False
save the info to member DB and attendance DB
print "member is added"
restart program

Figure 4: Adding new members to the database(s)

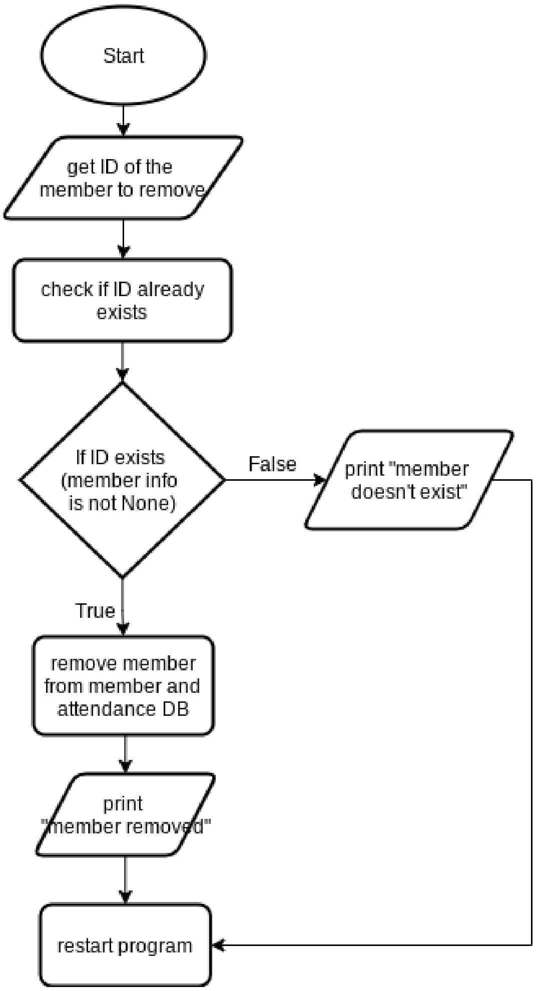

Start
get ID of the member to remove
check if ID already exists
If ID exists (member info is not None)
False
print "member doesn't exist"
True
remove member from member and attendance DB
print "member removed"
restart program

Figure 5: Removing member from both database(s)

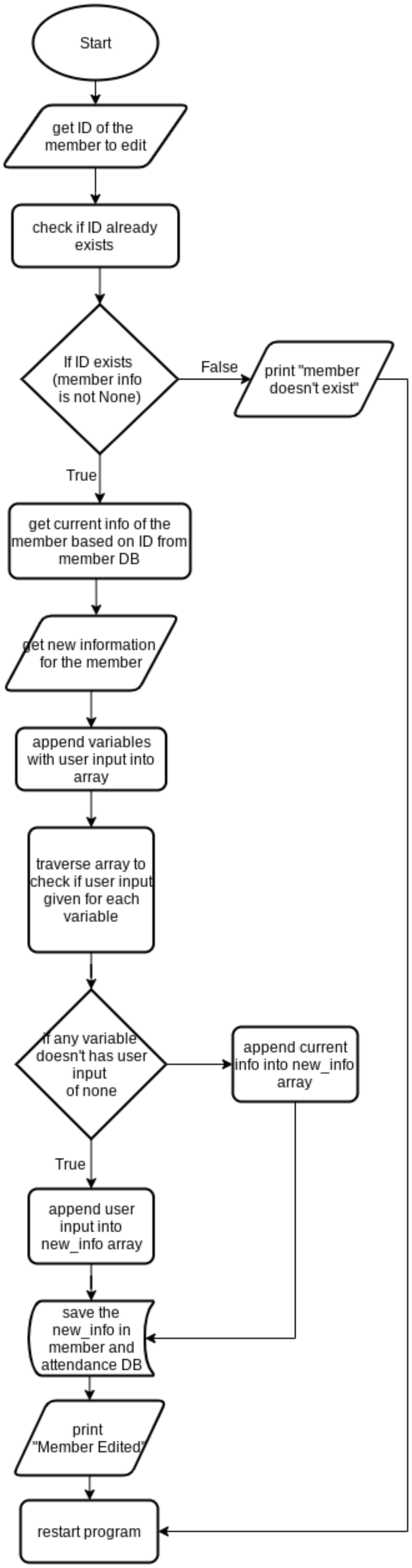

Start
get ID of the member to edit
check if ID already exists
If ID exists (member info is not None)
False
print "member doesn't exist"
True
get current info of the member based on ID from member DB
get new information for the member
append variables with user input into array
traverse array to check if user input given for each variable
if any variable doesn't has user input of none
append current info into new_info array
True
append user input into new_info array
save the new_info in member and attendance DB
print "Member Edited"
restart program

Figure 6: Editing member in the member DB

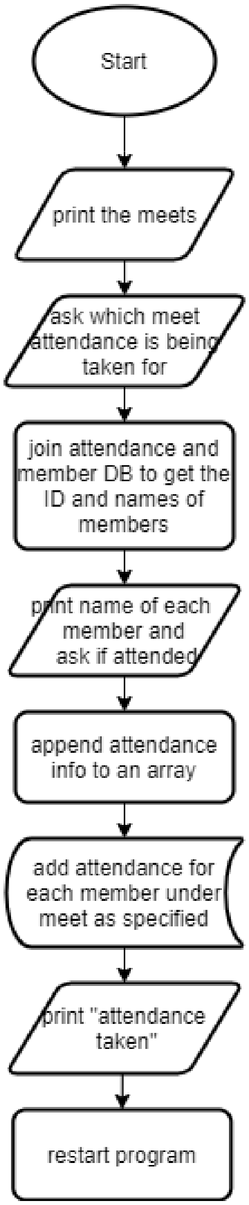
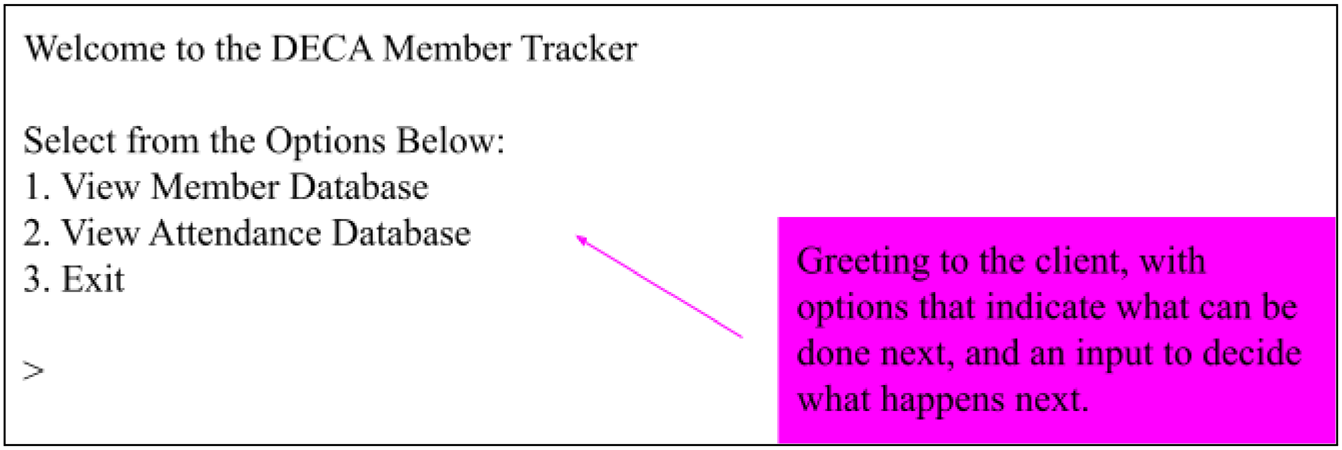

Start
print the meets
ask which meet attendance is being taken for
join attendance and member DB to get the ID and names of members
print name of each member and ask if attended
append attendance info to an array
add attendance for each member under meet as specified
print "attendance taken"
restart program

Figure 7: Taking attendance and adding it into the attendance DB

User-Interface Overview



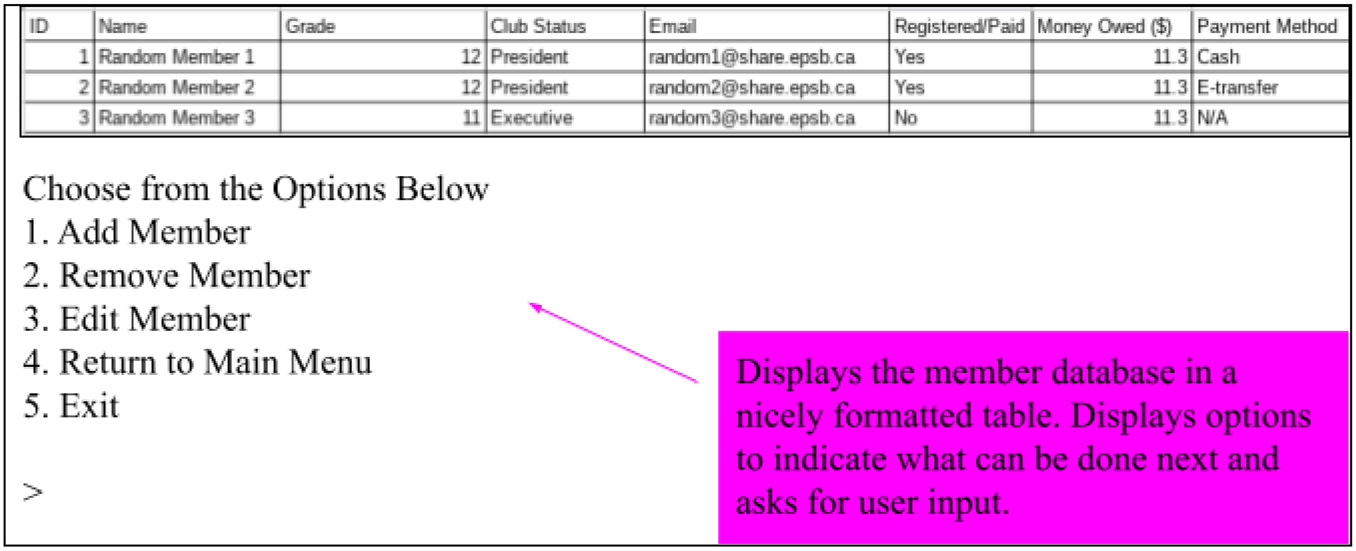
```
Welcome to the DECA Member Tracker

Select from the Options Below:
1. View Member Database
2. View Attendance Database
3. Exit

>
```

Greeting to the client, with options that indicate what can be done next, and an input to decide what happens next.

Figure 8: Main menu displayed upon entering program



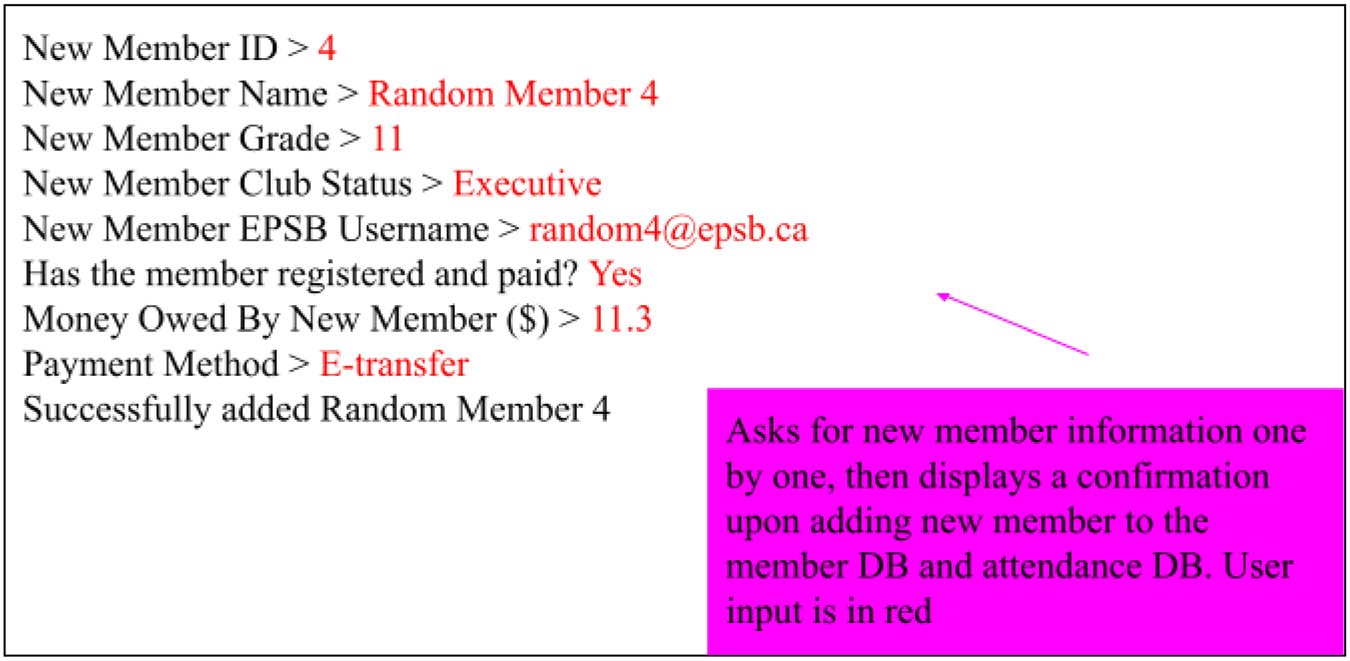
ID	Name	Grade	Club Status	Email	Registered/Paid	Money Owed ($)	Payment Method
1	Random Member 1	12	President	random1@share.epsb.ca	Yes	11.3	Cash
2	Random Member 2	12	President	random2@share.epsb.ca	Yes	11.3	E-transfer
3	Random Member 3	11	Executive	random3@share.epsb.ca	No	11.3	N/A

```
Choose from the Options Below
1. Add Member
2. Remove Member
3. Edit Member
4. Return to Main Menu
5. Exit

>
```

Displays the member database in a nicely formatted table. Displays options to indicate what can be done next and asks for user input.

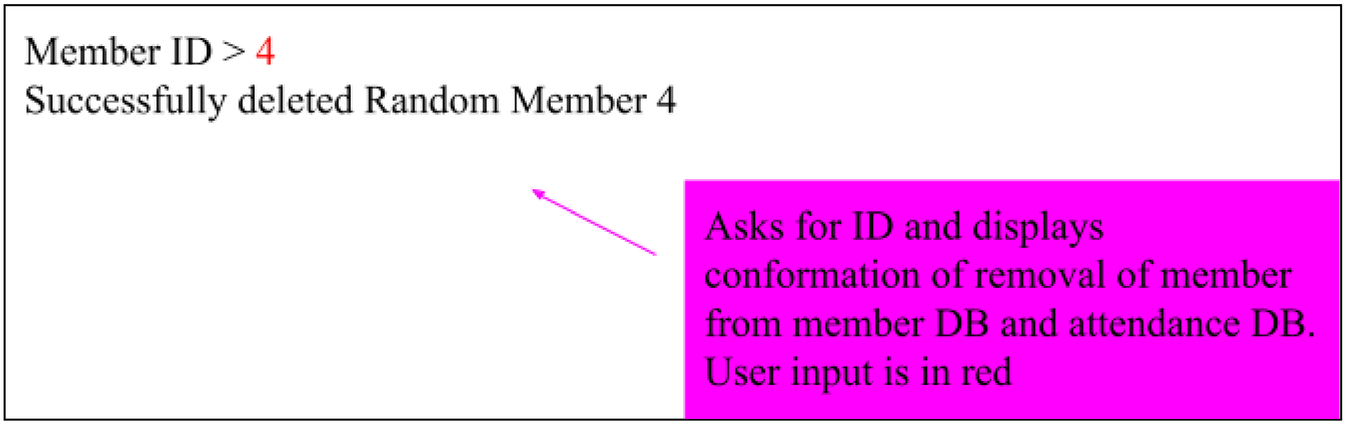
Figure 9: Member submenu and DB displayed if option 1 selected in main menu



```
New Member ID > 4
New Member Name > Random Member 4
New Member Grade > 11
New Member Club Status > Executive
New Member EPSB Username > random4@epsb.ca
Has the member registered and paid? Yes
Money Owed By New Member ($) > 11.3
Payment Method > E-transfer
Successfully added Random Member 4
```

Asks for new member information one by one, then displays a confirmation upon adding new member to the member DB and attendance DB. User input is in red

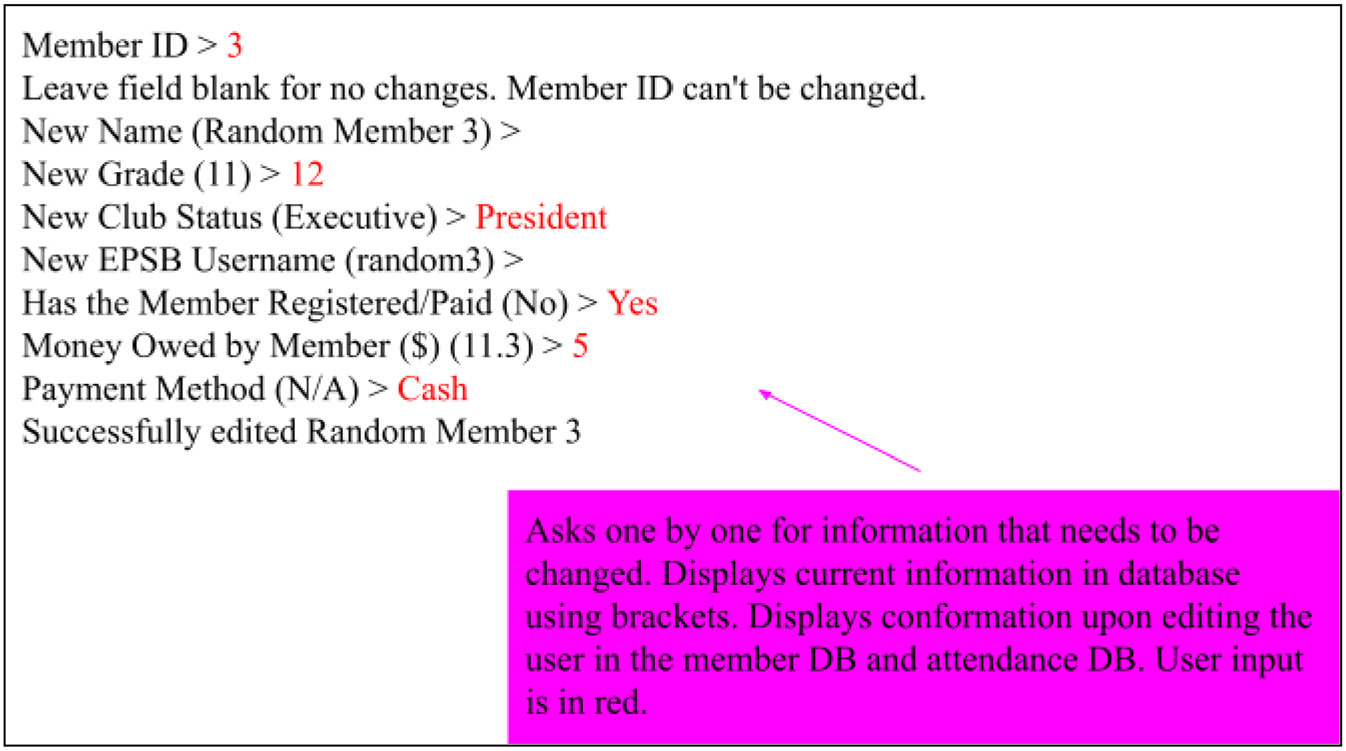
Figure 10: Get info and add new member if option 1 selected in member submenu



```
Member ID > 4
Successfully deleted Random Member 4
```

Asks for ID and displays conformation of removal of member from member DB and attendance DB. User input is in red

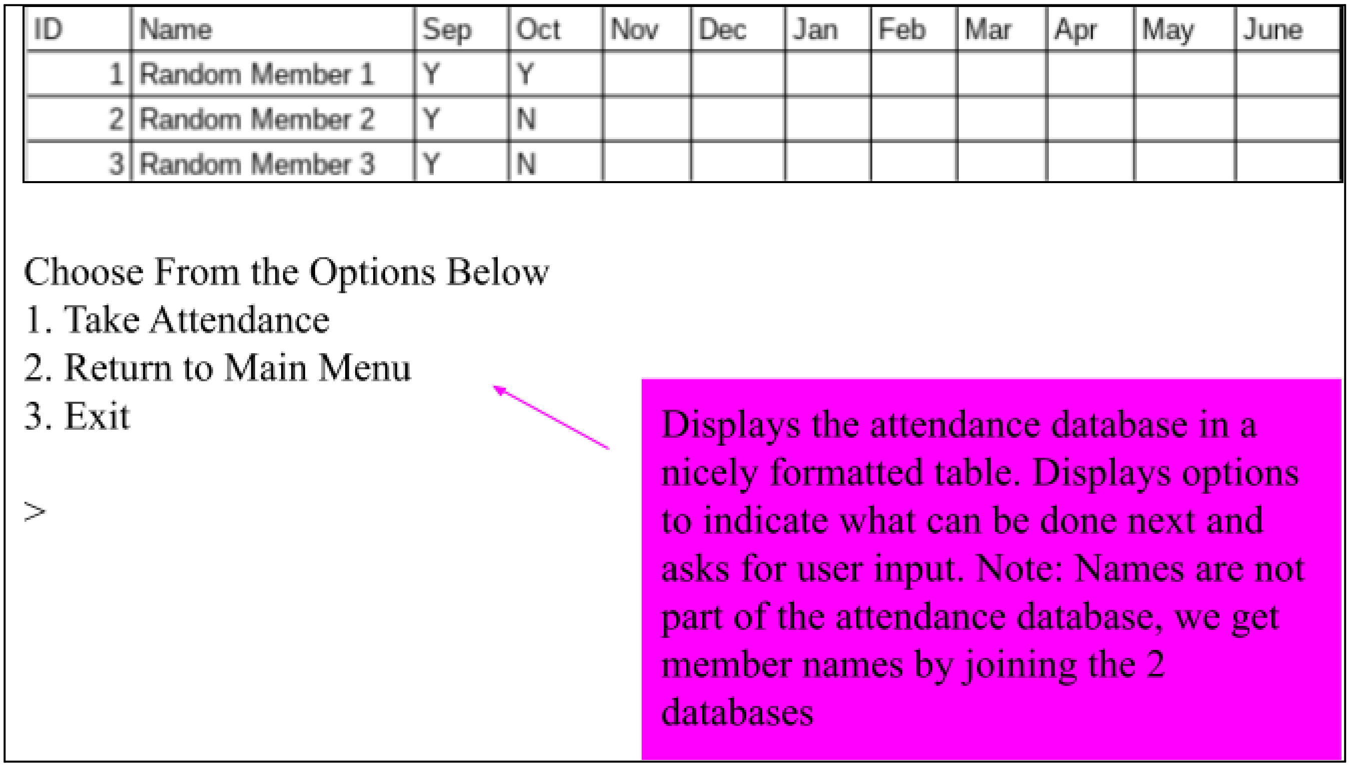
Figure 11: Get ID and remove member if option 2 selected in member submenu



```
Member ID > 3
Leave field blank for no changes. Member ID can't be changed.
New Name (Random Member 3) >
New Grade (11) > 12
New Club Status (Executive) > President
New EPSB Username (random3) >
Has the Member Registered/Paid (No) > Yes
Money Owed by Member ($) (11.3) > 5
Payment Method (N/A) > Cash
Successfully edited Random Member 3
```

Asks one by one for information that needs to be changed. Displays current information in database using brackets. Displays conformation upon editing the user in the member DB and attendance DB. User input is in red.

Figure 12: Get new information and edit member if option 3 selected in the member submenu



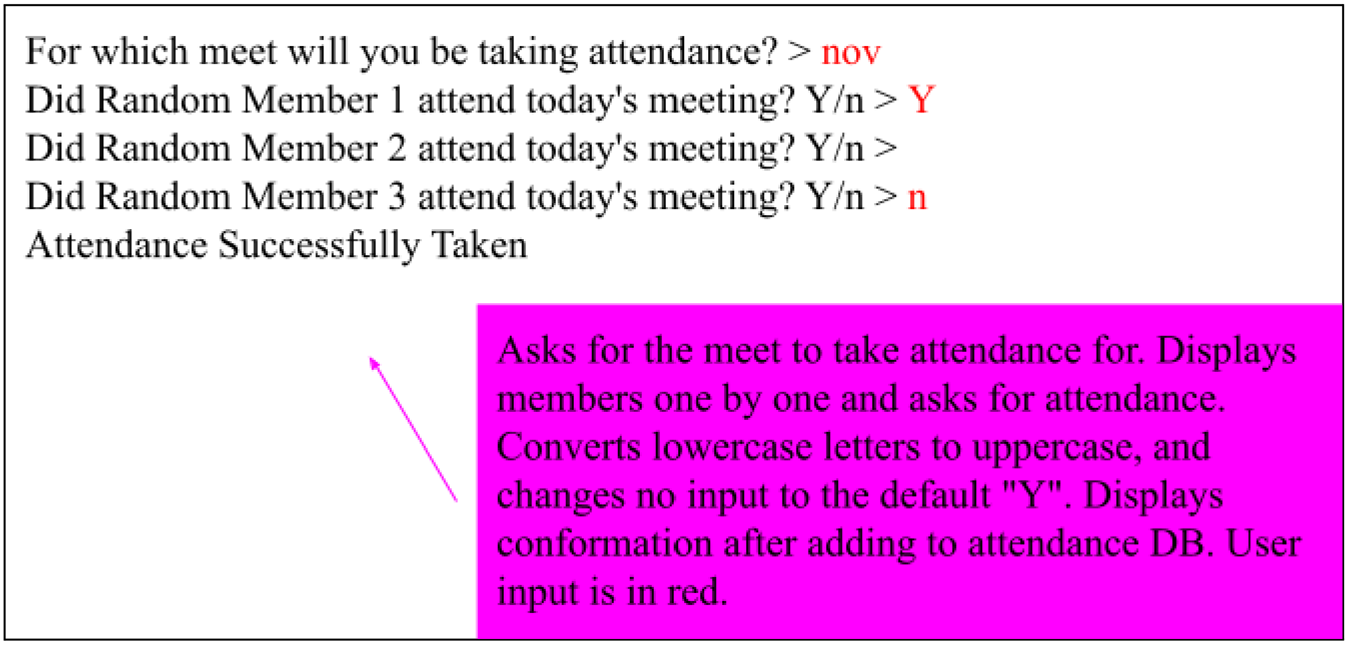
ID	Name	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	June
1	Random Member 1	Y	Y								
2	Random Member 2	Y	N								
3	Random Member 3	Y	N								

```
Choose From the Options Below
1. Take Attendance
2. Return to Main Menu
3. Exit


>
```

Displays the attendance database in a nicely formatted table. Displays options to indicate what can be done next and asks for user input. Note: Names are not part of the attendance database, we get member names by joining the 2 databases

Figure 13: Attendance submenu and DB displayed if option 1 selected in main menu



```
For which meet will you be taking attendance? > nov
Did Random Member 1 attend today's meeting? Y/n > Y
Did Random Member 2 attend today's meeting? Y/n >
Did Random Member 3 attend today's meeting? Y/n > n
Attendance Successfully Taken
```

Asks for the meet to take attendance for. Displays members one by one and asks for attendance. Converts lowercase letters to uppercase, and changes no input to the default "Y". Displays conformation after adding to attendance DB. User input is in red.

Figure 14: Take attendance and add to attendance DB if option 1 selected in attendance submenu

Test Plan

Action to be Tested	Test Method
Program will display a greeting and a working menu upon start	- Run the program and see if the menu is displayed. - Put an option not in the menu to see if it redirects back to the menu - input the menu option and check if it is redirected to the option.
Program displays data-entries in a nicely formatted table	- Select view attendance database check if values are in a nice table - Return to main menu and select view member database and check if values are in a nice table
Program asks for member info and adds members if add member is selected	- Select add member from the member submenu - Input ID, if ID exists there will be a warning given - If ID doesn't exist, enter the member info but don't add enter information for all inputs - Check if the program displays a warning and redirects to add member - Enter information for all inputs and check if the member and attendance DB has the member

Program asks for info to edit, then makes edits to member if edit member is selected	- Select edit member from the member submenu - Input ID, if ID doesn't exist there will be a warning given - If ID does exist, enter info to edit and see if member DB is edited
Program asks for ID and removes member, if remove member is selected	- Select edit member from the member submenu - Input ID, if ID doesn't exist there will be a warning given - If ID does exist, check attendance and member DB to see if member is removed
Program displays a submenu allowing to take attendance if attendance DB is selected	- Select view attendance DB from main menu - Put an option not in the menu to see if it redirects back to the submenu - Input submenu option and check if redirected to option
Program displays members individually and asks for attendance then updates attendance DB if take attendance selected	- Select take attendance from attendance submenu - Enter the meeting number - Check if the program displays members one by one asking for attendance - Input attendance for each member and check if attendance DB is updated
Program displays error message if any data-entry errors occur	- When entering information, input unexpected or null values - Check if any warning is displayed
Program restarts without user manually having to restart the program	- After performing each function check if the user is redirected to the main menu

Word count: 43