Task Number	Action	Outcome	Time	Target Completion Date	Criterion
1	Product Brainstorm	Propose 3 potential problems and solutions to my advisor	1 week	10/13/2023	A
2	Discussion of brainstorm	Discuss brainstormed ideas with my advisor and finalize a problem	3 days	10/16/2023	A
3	Initial meeting with the client	Discuss potential solutions and outline the functionality of the product with the client	1 day	10/17/2023	A
4	Scenario description	Write a scenario debrief based on the interview with the client including problems and solutions for the product	3 days	10/20/2023	A
5	Proposed product draft	Write an overview of the proposed solution	1 week	10/27/2023	A
6	Success Criteria	Outline success criteria for the product	2 days	10/29/2023	A
7	Submit Criterion A	Submit a draft of Criterion A to	1 day	10/30/2023	A

		my advisor for feedback			
8	Screen Prototypes	Create a hand-drawn prototype for each major screen of the program	3 weeks	11/21/2023	B
9	Flowchart	Flowchart the major methods and screens of the program	3 weeks	12/14/2023	B
10	Unit Test Plan	Outline the test plans based off the pre-defined success criteria	1 week	12/21/2023	B
11	UML Diagram	Complete the UML data diagram for the program	2 weeks	1/4/2024	B
12	Pseudocode	Create pseudocode to plan the product's algorithms and processes better	2 weeks	1/18/2024	B
13	Draft Criterion B	Submit a draft version of Criterion B to my advisor and discuss for feedback	1 week	1/25/2024	B
14	Revise Criterion A	Revise Criterion A based on the feedback of my advisor and submit a final version	1 week	2/4/2024	A

15	Revise Criterion B	Revise Criterion B based on the feedback of my advisor and submit a final version	1 week	2/11/2024	B
16	Criterion C - Checkpoint 1	Complete the authenticatio n screen, main screen, and tutor management screen	4 weeks	3/11/2024	C
17	Criterion C - Checkpoint 2	Finish program	3.5 weeks	4/4/2024	C
18	Criterion C - Extended writing	Complete the extended writing detailing the code structure and programming techniques	3 days	4/7/2024	C
19	Criterion D	Record the demonstratio n video for Criterion D	1 day	4/11/2024	D
20	Criterion E	Write down revisions and future improvement s for the program	1 day	4/12/2024	E
21	Finalize and complete IA	Last minute revisions and changes, submit for grading	1 day	4/13/2024	All

Figure #9

Tutor / tutee Management - Authenticated user ✓

* Both screens are separate but identical in design and functionality except for "Add tutor" or "Add tutee" Tutor management screen shown below *

1	John	Doe	view notes	Edit	Remove
2	Mary	Jane	view notes	Edit	Remove
3	George	Washington	view notes	Edit	Remove
4	Steve	Jones	view notes	Edit	Remove
5	Bob	Phillips	view notes	Edit	Remove
+	Add tutor				

Back | Save → | Filter →

Figure #10

Search / filter function ✓

* This function will be identical in design and functionality for tutor/tutee management, creating an appointment, and search tutor/tutee functions for both authenticated and unauthenticated users. Screens will only differ in filtration characteristics listed in the success criteria. An example for tutor search (unauthenticated) is shown below *

Availability
- ☐ Wed Jan 1, 5pm
- ☑ Thu Jan 2, 3pm

Prior classes
- ☐ Algebra 2
- ☐ Geometry
- ☑ Statistics

Skills
- ☑ Microsoft Excel

Apply Filter | Back

Figure #11

Add tutor/tutee - Authenticated user. ✓

Name: New User

Availability
- ☐ Wed Jan 1, 5pm
- ☐ Thu Jan 2, 3pm

prior classes
- ☐ Algebra 2

Skills
- ☐ Microsoft Excel

proficiency
- ☐ Spanish

Create | Add Note | Cancel

Figure #12

Edit tutor / tutee - Authenticated user. ✓

Name: John Doe

Availability
- ☑ Wed Jan 1, 5pm
- ☐ Thu Jan 2, 3pm

prior classes
- ☑ Algebra 2

Skills
- ☐ Microsoft Excel

proficiency
- ☑ Spanish

Save | Add Note | Back

Show 2 panels: filters and list of users

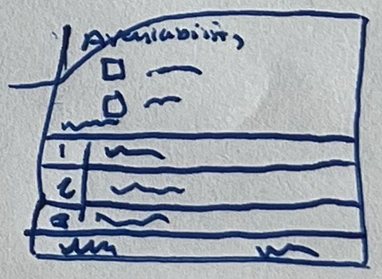


Figure # 5

General Management – Authenticated user ✓

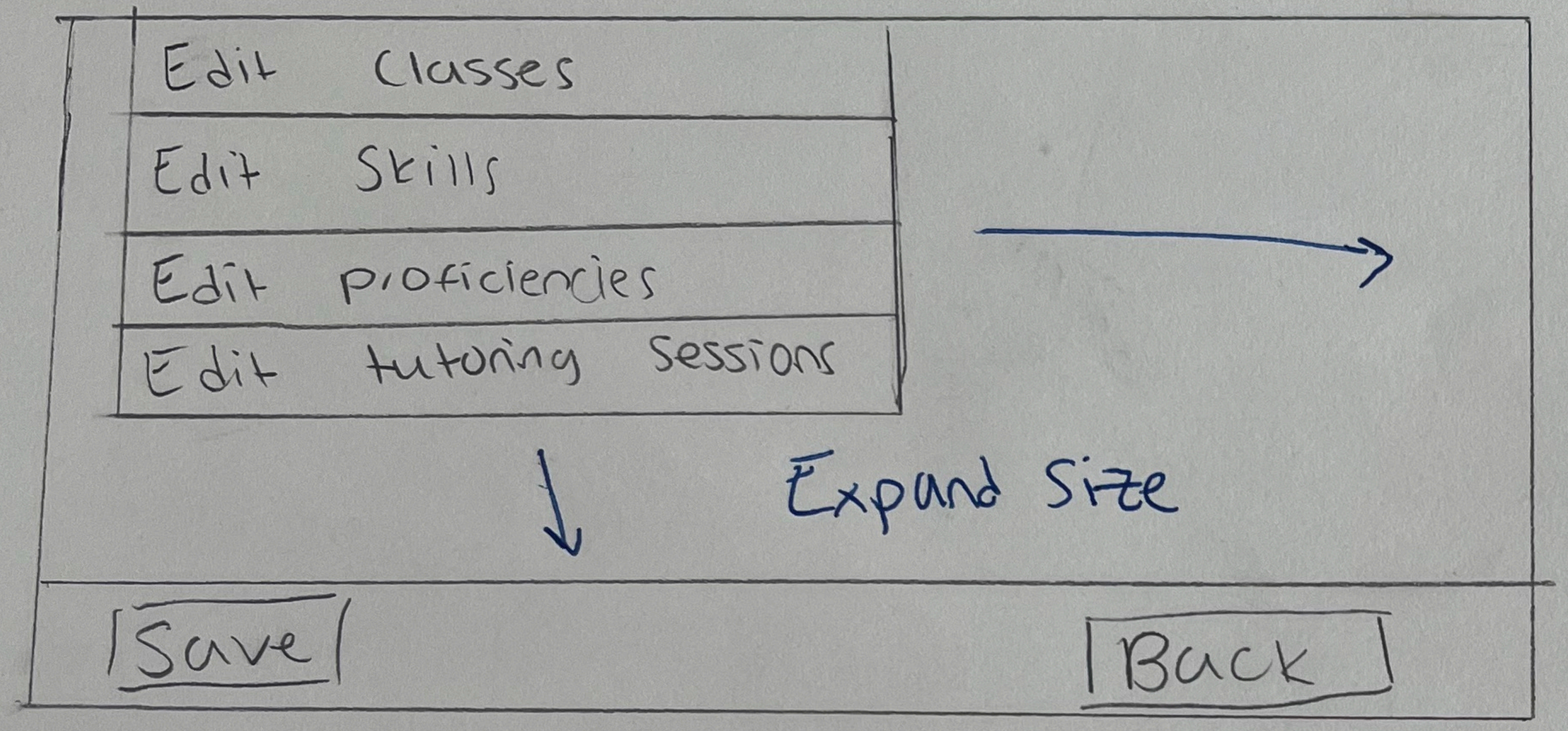


Figure # 6

Edit Classes / Skills / proficiencies / tutoring Sessions ✓

* screens will be similar for each function, example shown below *

Classes			
1	Algebra 1	Edit	Remove
2	Geometry	Edit	Remove
3	Statistics	Edit	Remove
4	English 10	Edit	Remove
5	English 11	Edit	Remove
+	Enter Name		

Save Back

Figure #7

Create Appointment - Authenticated user ✓ Online

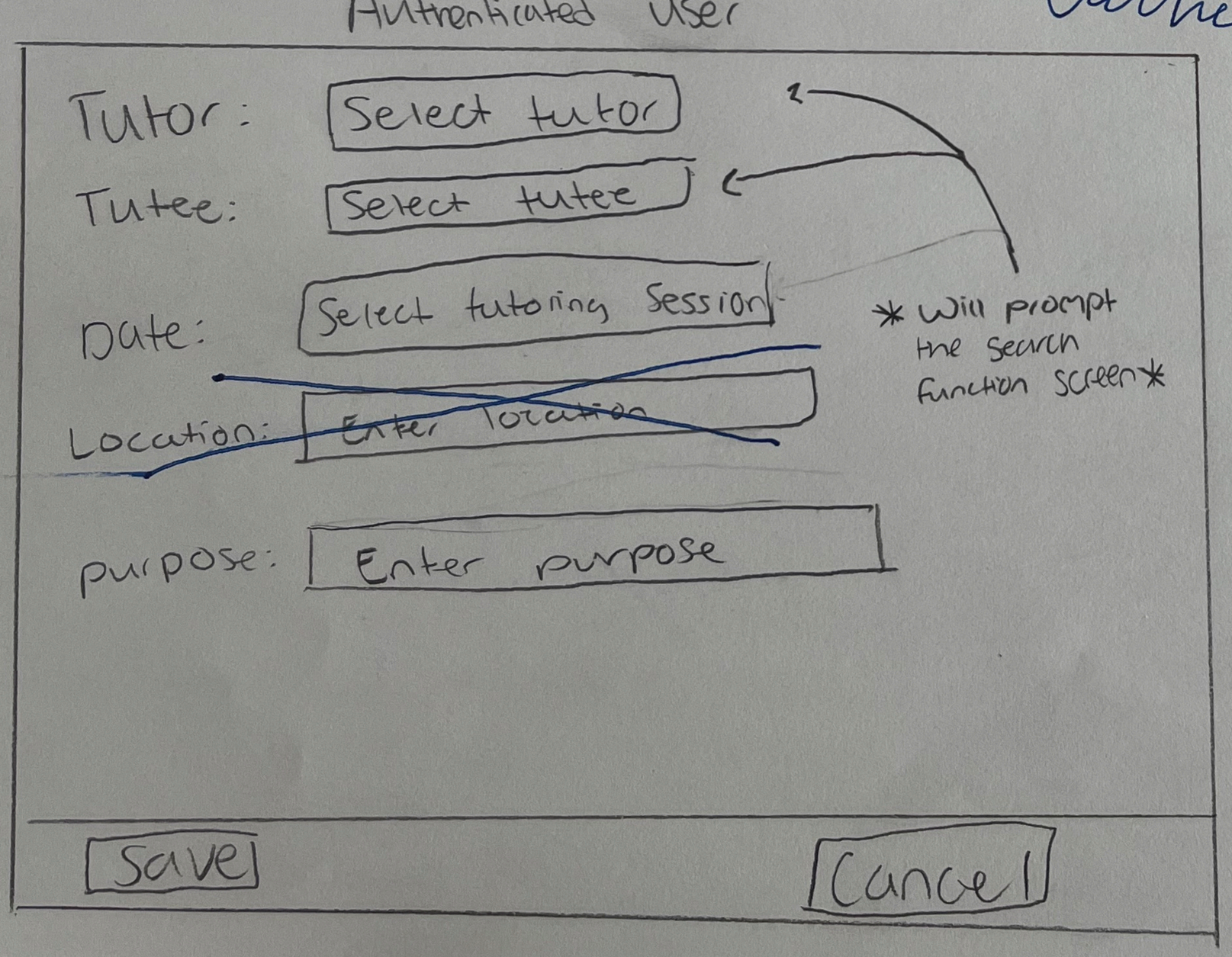


Figure #8

View completed tutor hours - Authenticated user ✓

1	John Doe	7 hours	edit
2	Mary Jane	3 hours	edit
3	George Washington	27 hours	edit
4	Steve Jones	0 hours	edit
5	Bob phillips	2 hours	edit

Back

Figure #1

Start Screen ✓

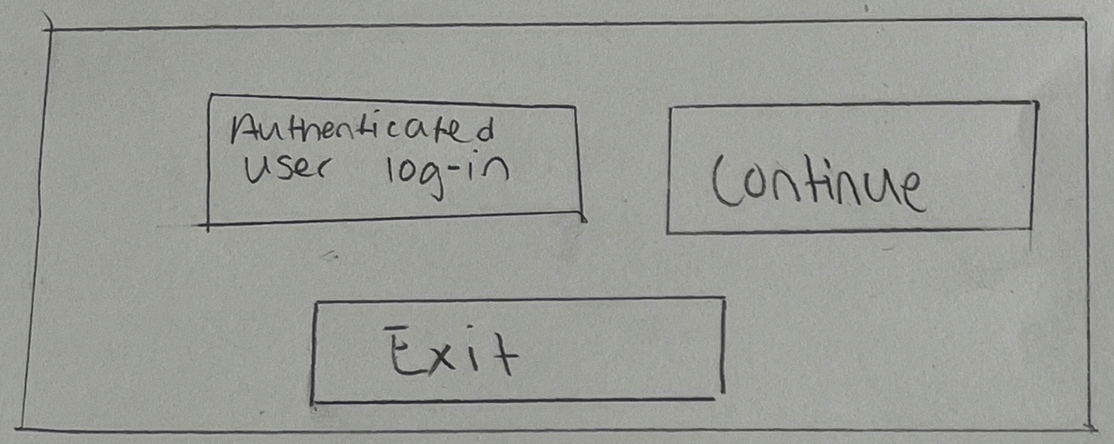


Figure #2

Authentication ✓

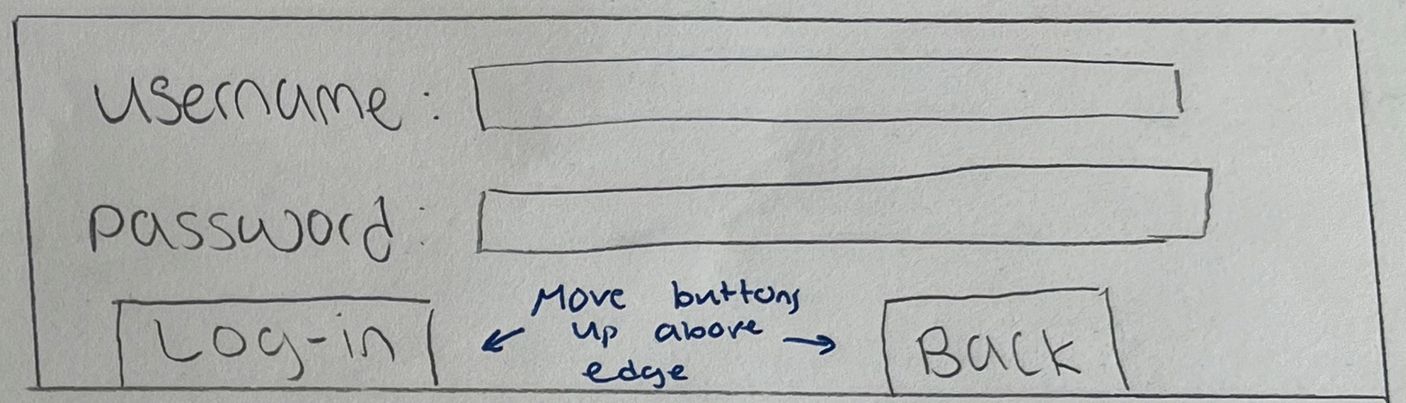


Figure #3

Primary Screen - Authenticated User ✓

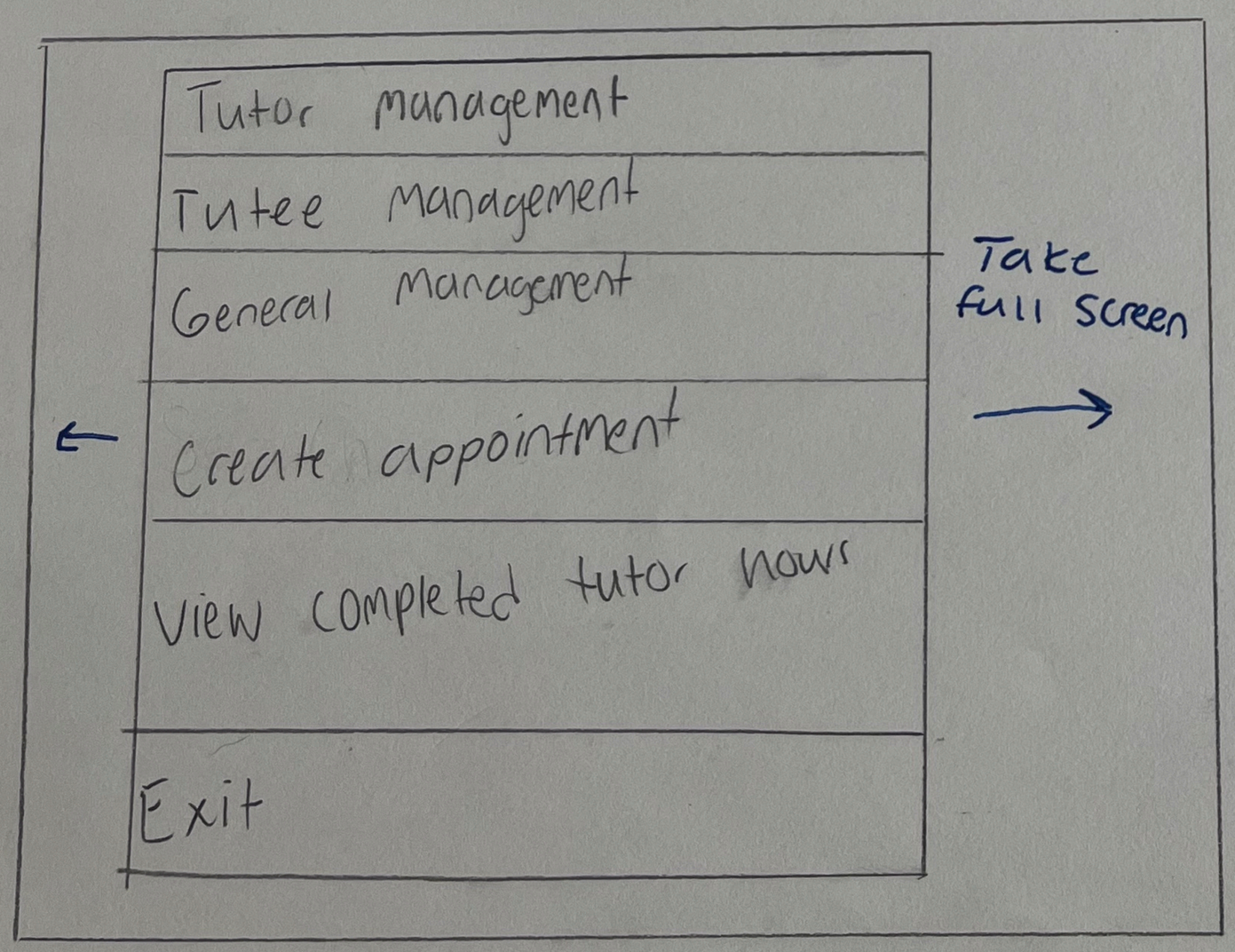
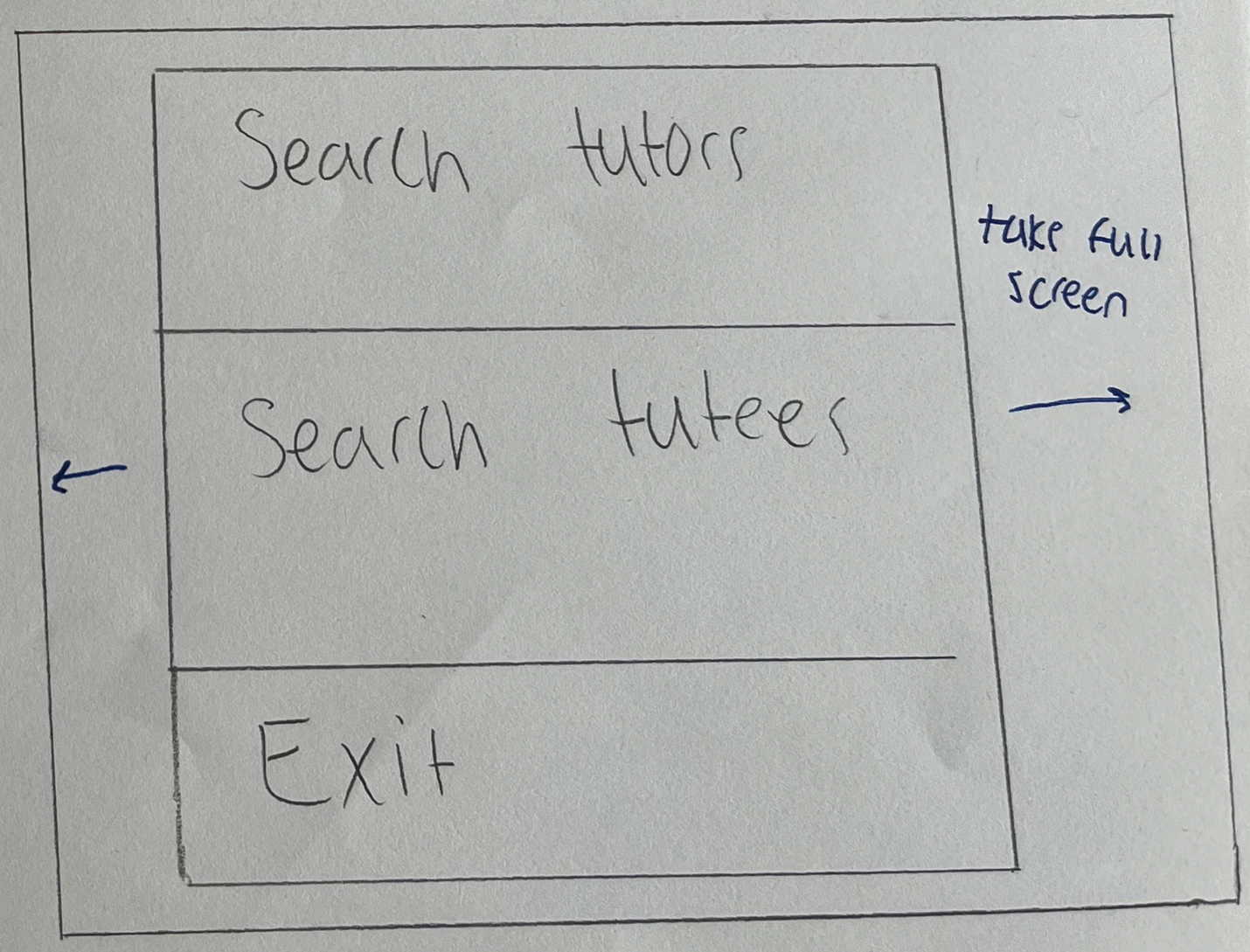


Figure #4

Primary Screen - Unauthenticated User ✓



Unit Test Plan

Step Description	Input	Expected Outcome
The user runs the program	None	A start screen displaying options for authenticated login, continue, and exit are provided
The user clicks “continue”	Button	The primary screen for unauthenticated users is shown
The user clicks “search tutors” within the unauthenticated primary screen	Button click	A search menu appears allowing the user to toggle filters, apply filters, and go back. The menu also shows a list of tutors that are updated as filters are applied
The user clicks “apply filter”	Button click	The displayed tutors should be filtered by the selected filters and removed from the displayed list if they do not apply
The user toggles a filter	Button click	The filter’s checkbox should toggle between active and inactive
The user clicks “back”	Button click	The user should be returned to the primary screen for unauthenticated users
The user clicks “search tutees” within the unauthenticated primary screen	Button click	A search menu appears allowing the user to toggle filters, apply filters, and go back. The menu also shows a list of tutees that are updated as filters are applied
The user clicks “apply filter”	Button click	The displayed tutees should be filtered by the selected filters and removed from the displayed list if they do not apply

The user clicks “exit” within the unauthenticated primary screen	Button click	The user should be returned to the initial start menu
The user clicks “Authenticated user login” within the initial start menu	Button click	The user is brought to the authentication screen
The user types a username	Textbox	The user’s keyboard input should appear in the textbox
The user types a password	Textbok	The user’s keyboard input should appear in the textbox, masked from displaying the user’s password however
The user clicks log-in	Button click	The program should read credentials from an external data file
The user clicks log-in	Button click	The program should determine if the user’s credentials are correct
The user’s credentials are invalid	None	The program alerts the user that their credentials are invalid
The user’s credentials are valid	None	The program displays the primary screen for authenticated users
The primary screen for authenticated users is displayed	None	The user should see options for “tutor management” “tutee management” “general management” “create appointment” “view completed tutor hours” and “exit”
The user selects “exit”	Button click	The user should be returned to the initial start screen
The user selects tutor management	Button click	The list of current tutors is loaded from an external data file
The user selects tutor	Button click	The user is shown a list of

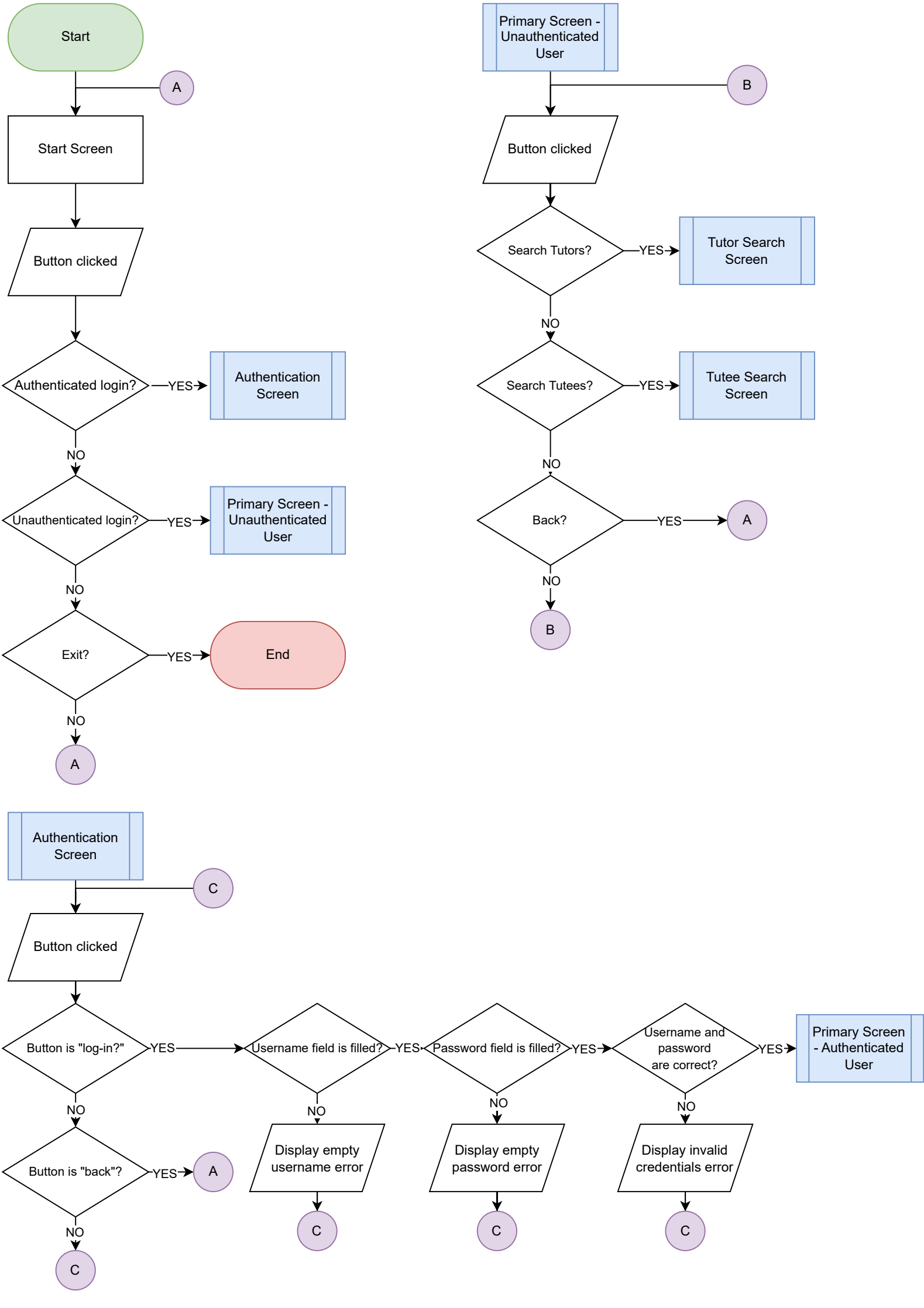
management		current tutors with options to view notes, edit, remove each tutor and options to go back, save changes, and filter tutors
The user clicks view notes	Button click	The user's note data is loaded from an external data file
The user clicks views notes	Button click	The user is displayed the text notes of a tutor if available
The user clicks edit (tutor)	Button click	The properties of a tutor are loaded from an external data file
The user clicks edit (tutor)	Button click	The user is prompted a screen to edit the properties of a tutor
The user clicks remove	Button click	The tutor is removed from the list of tutors
The user clicks back	Button click	The user is returned to the primary authenticated user page without saving any changes
The user clicks save	Button click	The user's changes to tutors are saved to an external file
The user clicks filter	Button click	Filters are loaded from the external general management data file
The user clicks filter	Button click	A filter function panel is shown to the user allowing them to narrow tutor results
The user clicks a filter	Button click	The status of the filter is toggled
The user clicks apply filter	Button click	The user is returned to their previous screen with their search results narrowed according to their selected filters
The user clicks back	Button click	The user is returned to their

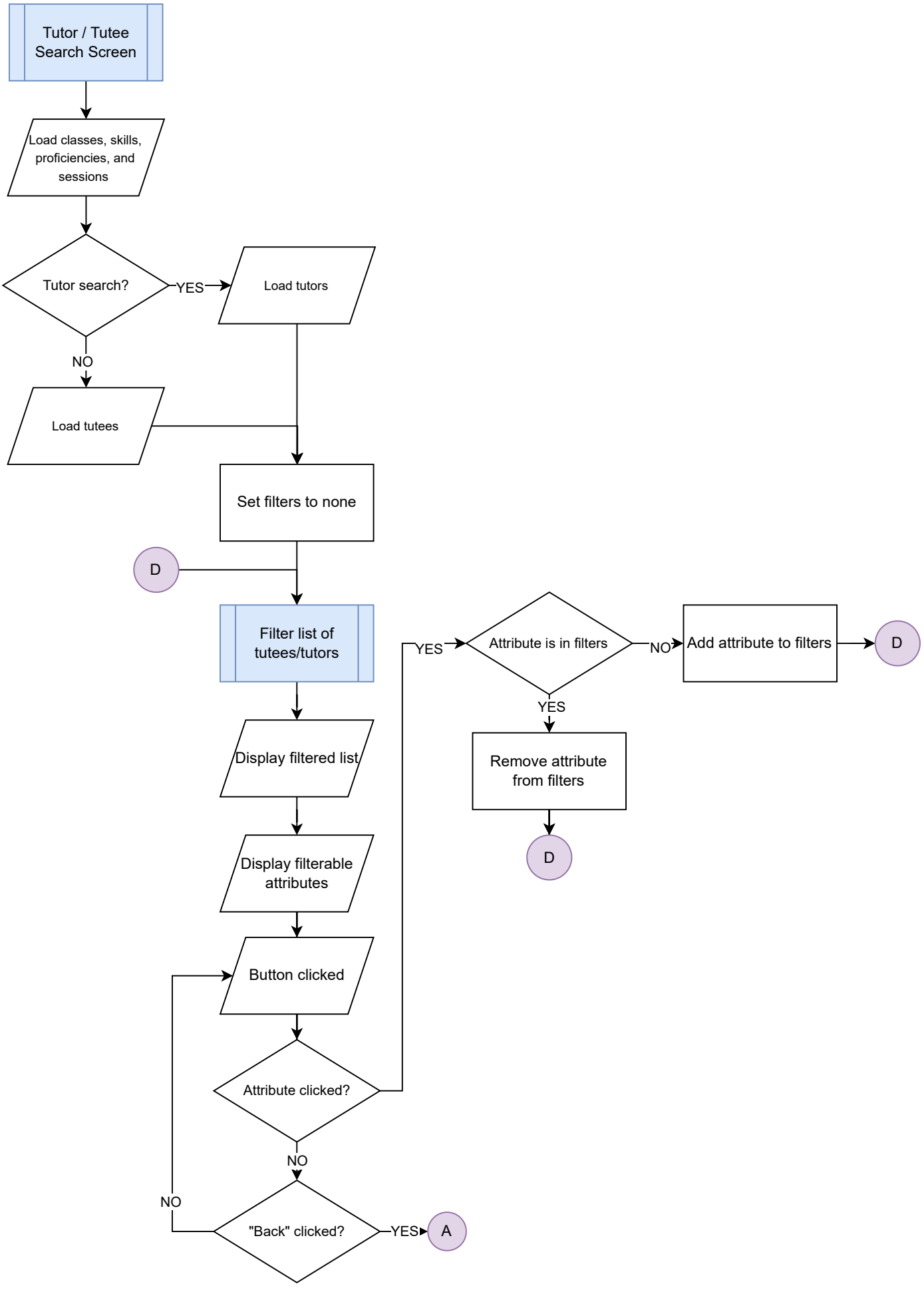
		previous screen without any filters
The user selects tutee management	Button click	The list of tutees is loaded from an external data file
The user selects tutee management	Button click	The user is shown a list of current tutees with options to view notes, edit, remove each tutee and options to go back, save changes, and filter tutees
The user clicks view notes	Button click	The tutee's note data is loaded from an external data file
The user clicks views notes	Button click	The user is displayed the text notes of a tutee if available
The user clicks edit	Button click	The properties of a tutee are loaded from an external data file
The user clicks edit	Button click	The user is prompted a screen to edit the properties of a tutee
The user clicks remove	Button click	The tutee is removed from the list of tutees
The user clicks back	Button click	The user is returned to the primary authenticated user page without saving any changes
The user clicks save	Button click	The user's changes to tutees are saved to an external file
The user clicks filter	Button click	Filters are loaded from the external general management data file
The user clicks filter	Button click	A filter function panel is shown to the user allowing them to narrow tutee results
The user clicks general management	Button click	The user is brought to a screen displaying options to

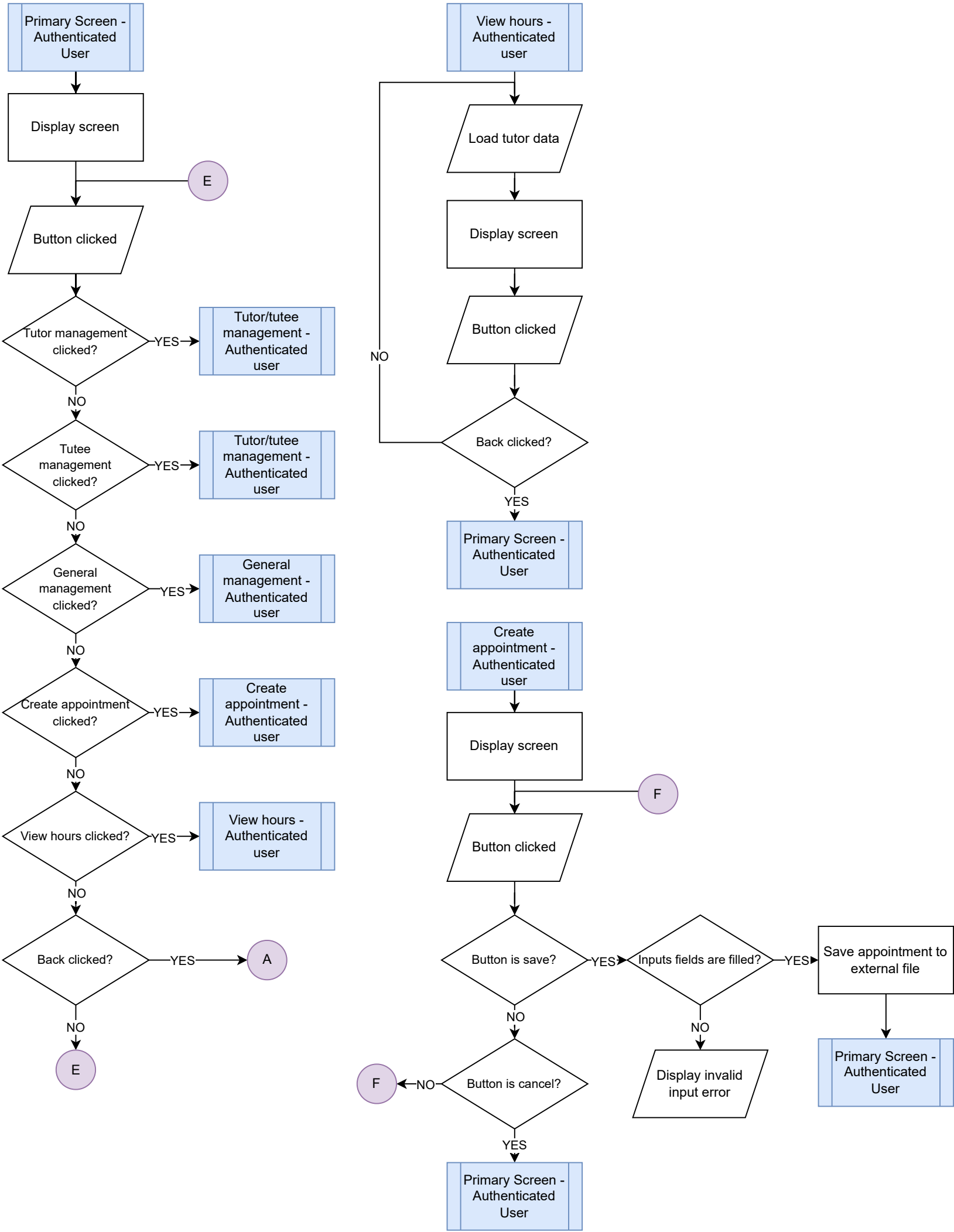
		edit classes, edit skills, edit proficiencies, and edit tutoring sessions
The user clicks save	Button click	Any of the user's changes to the general management panel are saved to the general management file and the user is returned to the primary authenticated user screen
The user clicks back	Button click	The user is returned to the primary authenticated user screen without any changes saved
The user clicks edit classes/skills/proficiencies/tutoring sessions	Button click	The user is presented a screen listing the current classes/skills/proficiencies/tutoring sessions with options to edit and remove each class/skill/proficiency/tutoring session. The user is presented a textbox to add classes/skills/proficiencies/tutoring sessions. The user is presented a save and back button
The user clicks the edit button on a class/skill/proficiency/tutoring session	Button click + textbox	The user's changes to the textbox are applied to the classes/skills/proficiencies/tutoring sessions on button click
The user clicks the remove button on a class/skill/proficiency/tutoring session	Button click	The class/skill/proficiency/tutoring session is removed from the list of classes/skills/proficiencies/tutoring sessions
The user clicks the add button	Textbox + button click	The user's inputted value is added to the class/skill/proficiency/tutoring session list
The program displays an error	None	The user is notified that their

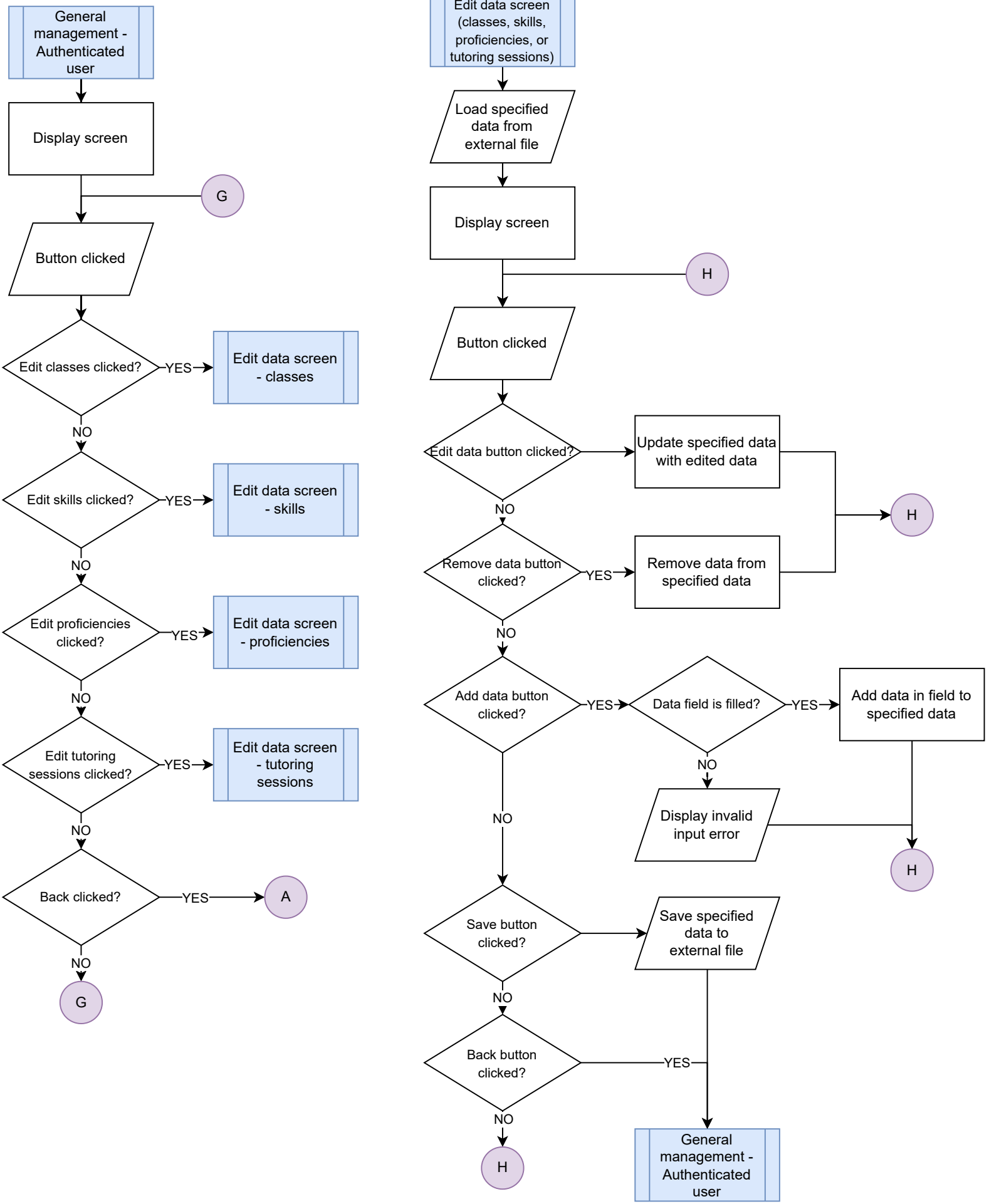
for an empty text box		textbox is empty
The user clicks the save button	Button	The user's changes are saved to the general management file
The user clicks the back button	Button	The user's changes are not saved to the general management file and the user is returned to the general management screen
The user clicks create appointment	Button	The user is provided a menu to schedule an appointment with inputs for a tutor, tutee, date, and purpose
The user clicks save appointment	Button	The user's appointment is saved to an external data file
The user's inputs are invalid or missing	None	The program alerts the user that their appointment inputs are incorrect
The user clicks cancel	Button	The user's changes to the appointment are not saved and the user is returned to the primary screen for authenticated users
The user clicks view completed tutor hours	Button	The user is provided a list of each tutor and the number of hours they have tutored so far
The user clicks edit on a tutor's hours	Button	The user is provided a textbox to edit the number of hours that a tutor has completed
The user clicks back within the total hour screen	Button	The user is returned to the primary screen for authenticated users
The user clicks exit within the primary authenticated user screen	Button	The user is returned to the initial start screen

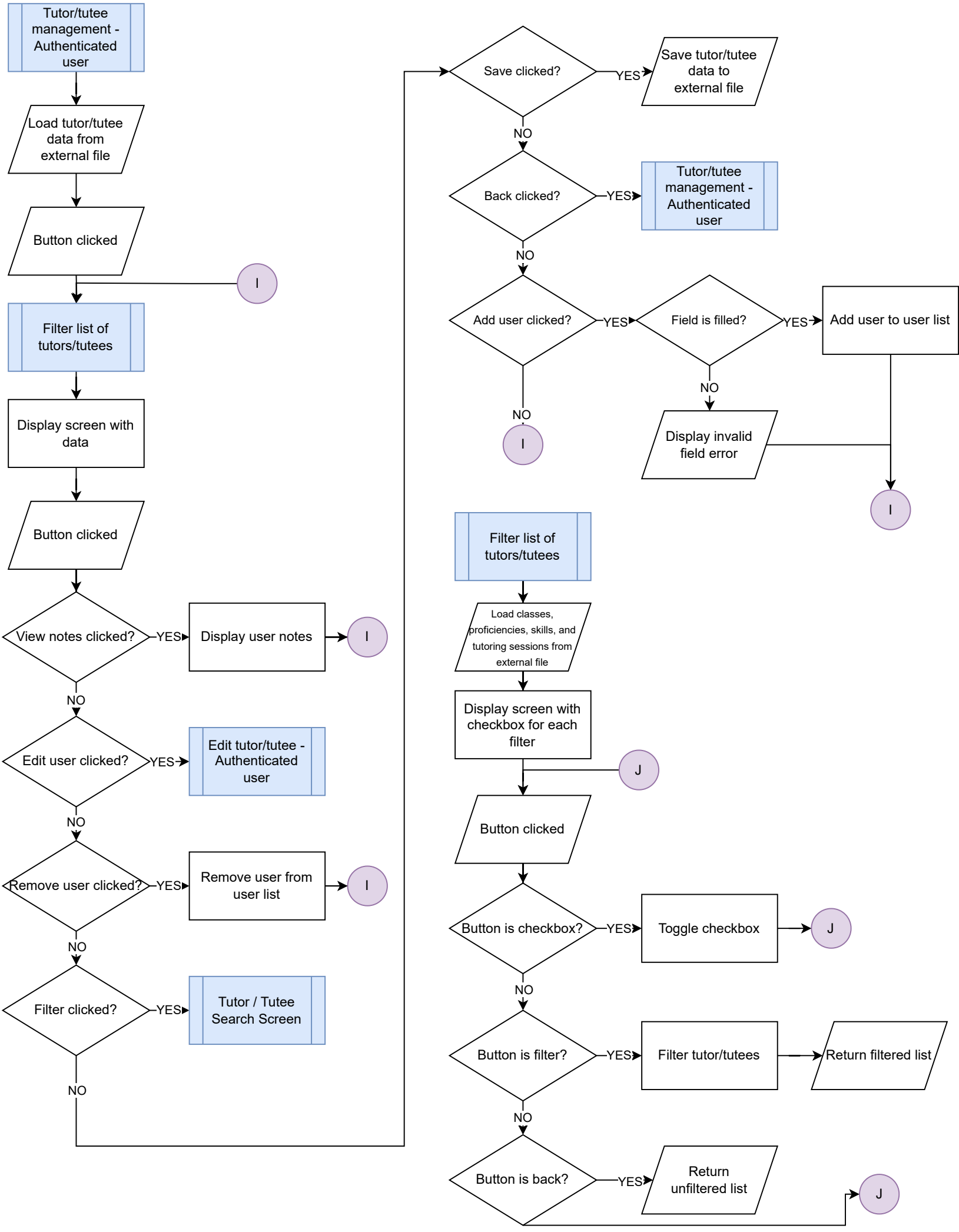
The user clicks exit within the initial start screen	Button	The program is closed and exits cleanly
An external data file throws an exception while reading	Internal exception	The program handles the exception and notifies the user of the error
An external data file throws an exception while writing	Internal exception	The program handles the exception and notifies the user of the error

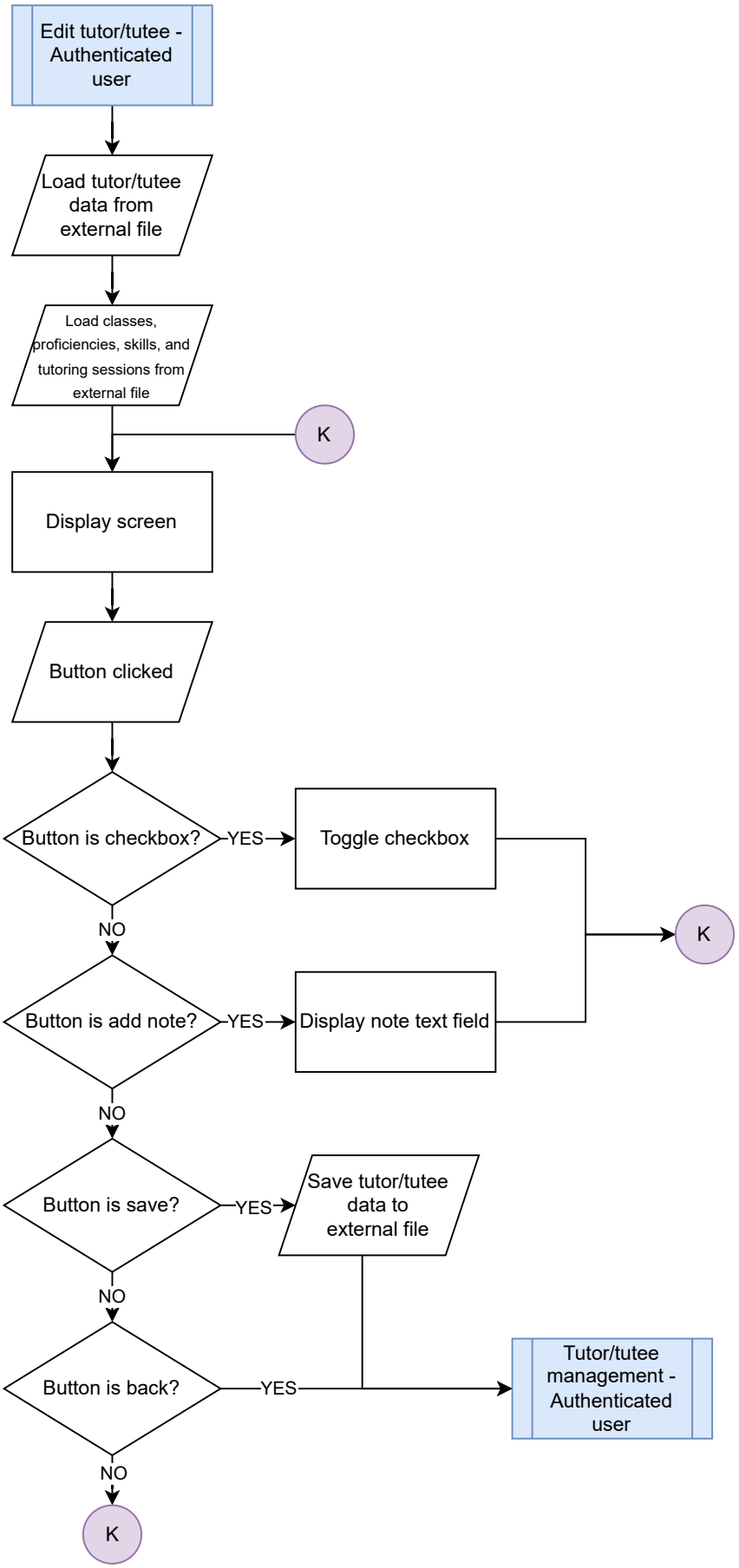

Start
A
Start Screen
Button clicked
Authenticated login?
YES
Authentication Screen
NO
Unauthenticated login?
YES
Primary Screen - Unauthenticated User
NO
Exit?
YES
End
NO
A
Primary Screen - Unauthenticated User
B
Button clicked
Search Tutors?
YES
Tutor Search Screen
NO
Search Tutees?
YES
Tutee Search Screen
NO
Back?
YES
A
NO
B
Authentication Screen
C
Button clicked
Button is "log-in?"
YES
Username field is filled?
YES
Password field is filled?
YES
Username and password are correct?
YES
Primary Screen - Authenticated User
NO
Button is "back"?
YES
A
NO
C
NO
Display empty username error
C
NO
Display empty password error
C
NO
Display invalid credentials error
C

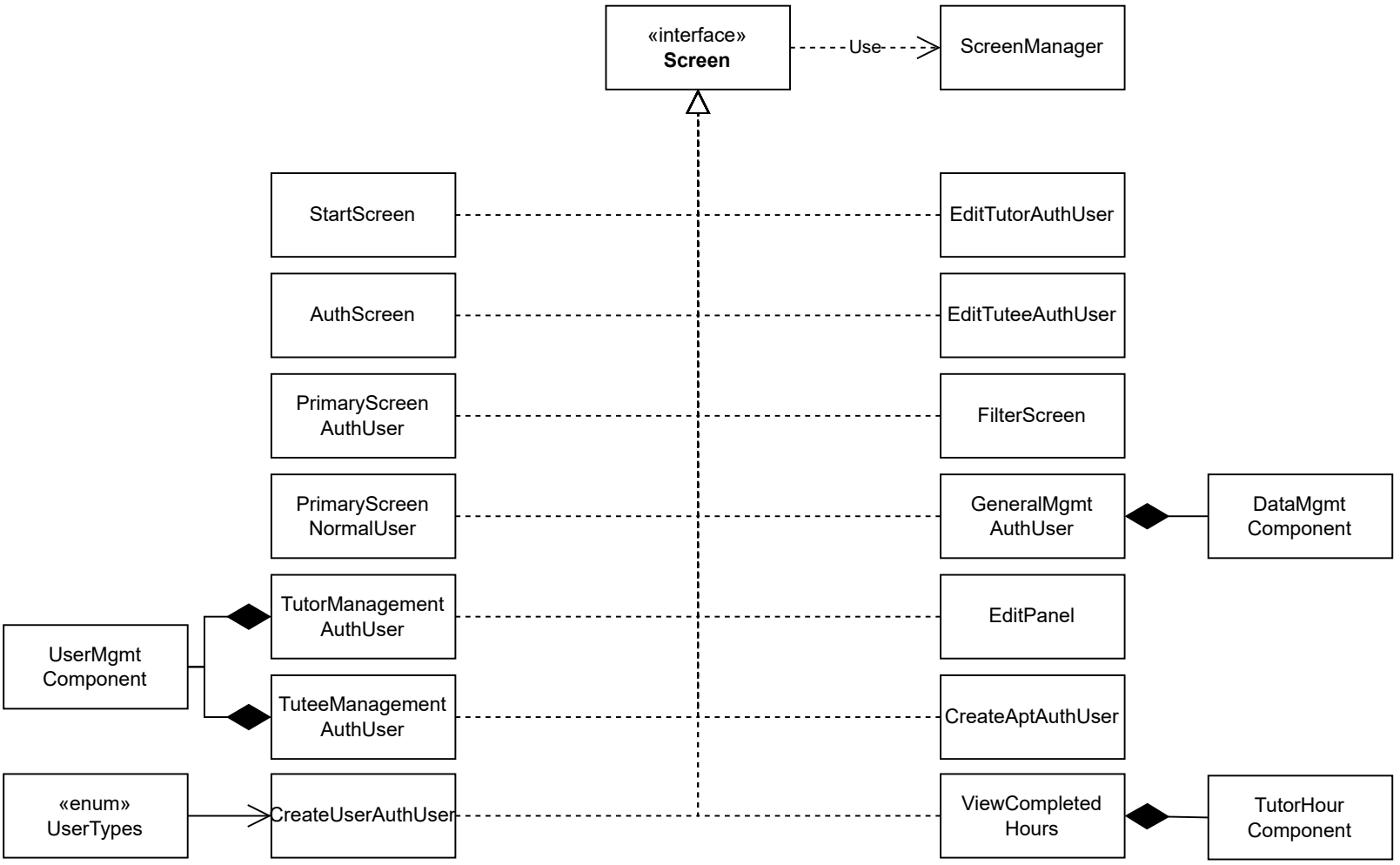

Tutor / Tutee Search Screen
Load classes, skills, proficiencies, and sessions
Tutor search?
YES
Load tutors
NO
Load tutees
Set filters to none
D
Filter list of tutees/tutors
Display filtered list
Display filterable attributes
Button clicked
Attribute clicked?
YES
Attribute is in filters
NO
Add attribute to filters
D
YES
Remove attribute from filters
D
NO
"Back" clicked?
YES
A
NO

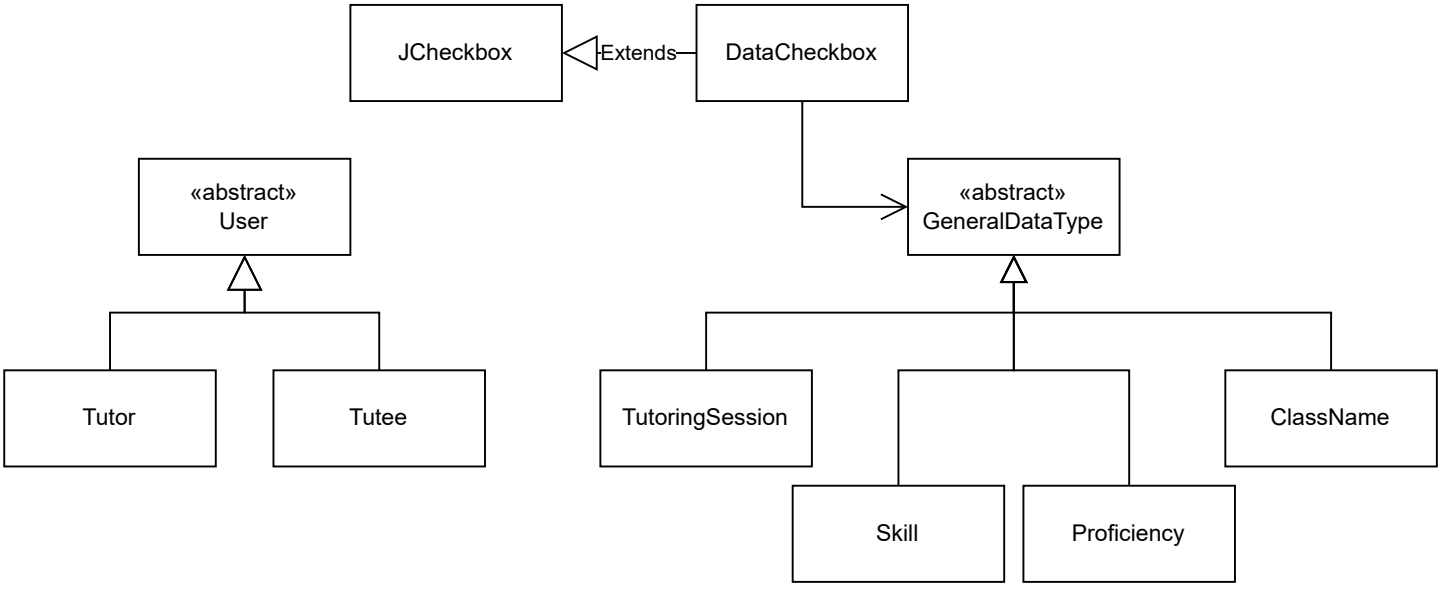

Primary Screen - Authenticated User
Display screen
E
Button clicked
Tutor management clicked?
YES
Tutor/tutee management - Authenticated user
NO
Tutee management clicked?
YES
Tutor/tutee management - Authenticated user
NO
General management clicked?
YES
General management - Authenticated user
NO
Create appointment clicked?
YES
Create appointment - Authenticated user
NO
View hours clicked?
YES
View hours - Authenticated user
NO
Back clicked?
YES
A
NO
E
View hours - Authenticated user
Load tutor data
Display screen
Button clicked
NO
Back clicked?
YES
Primary Screen - Authenticated User
Create appointment - Authenticated user
Display screen
F
Button clicked
Button is save?
YES
Inputs fields are filled?
YES
Save appointment to external file
Primary Screen - Authenticated User
NO
Display invalid input error
NO
Button is cancel?
F
YES
Primary Screen - Authenticated User


General management - Authenticated user
Display screen
G
Button clicked
Edit classes clicked?
YES
Edit data screen - classes
NO
Edit skills clicked?
YES
Edit data screen - skills
NO
Edit proficiencies clicked?
YES
Edit data screen - proficiencies
NO
Edit tutoring sessions clicked?
YES
Edit data screen - tutoring sessions
NO
Back clicked?
YES
A
NO
G
Edit data screen (classes, skills, proficiencies, or tutoring sessions)
Load specified data from external file
Display screen
H
Button clicked
Edit data button clicked?
Update specified data with edited data
NO
Remove data button clicked?
YES
Remove data from specified data
H
NO
Add data button clicked?
YES
Data field is filled?
YES
Add data in field to specified data
NO
Display invalid input error
H
NO
Save button clicked?
Save specified data to external file
NO
Back button clicked?
YES
NO
H
General management - Authenticated user

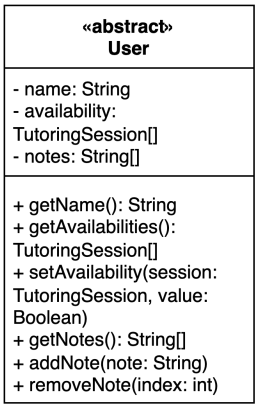
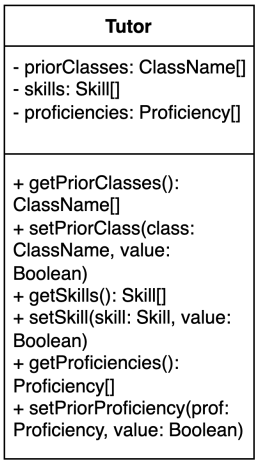

Tutor/tutee management - Authenticated user
Load tutor/tutee data from external file
Button clicked
I
Filter list of tutors/tutees
Display screen with data
Button clicked
View notes clicked?
YES
Display user notes
I
NO
Edit user clicked?
YES
Edit tutor/tutee - Authenticated user
NO
Remove user clicked?
YES
Remove user from user list
I
NO
Filter clicked?
YES
Tutor / Tutee Search Screen
NO
Save clicked?
YES
Save tutor/tutee data to external file
NO
Back clicked?
YES
Tutor/tutee management - Authenticated user
NO
Add user clicked?
YES
Field is filled?
YES
Add user to user list
NO
Display invalid field error
NO
I
I
Filter list of tutors/tutees
Load classes, proficiencies, skills, and tutoring sessions from external file
Display screen with checkbox for each filter
J
Button clicked
Button is checkbox?
YES
Toggle checkbox
J
NO
Button is filter?
YES
Filter tutor/tutees
Return filtered list
NO
Button is back?
YES
Return unfiltered list
J

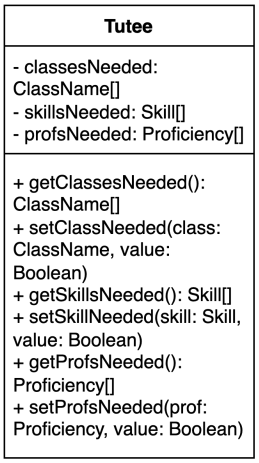
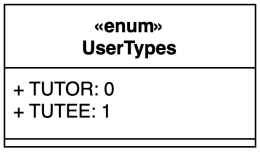
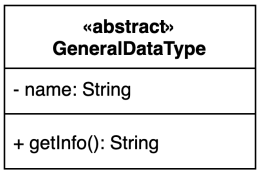
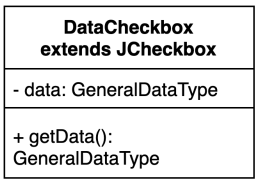
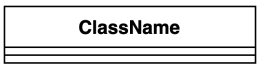

Edit tutor/tutee - Authenticated user
Load tutor/tutee data from external file
Load classes, proficiencies, skills, and tutoring sessions from external file
K
Display screen
Button clicked
Button is checkbox?
YES
Toggle checkbox
NO
K
Button is add note?
YES
Display note text field
NO
Button is save?
YES
Save tutor/tutee data to external file
NO
Button is back?
YES
Tutor/tutee management - Authenticated user
NO
K

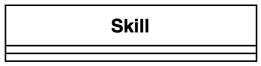
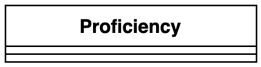
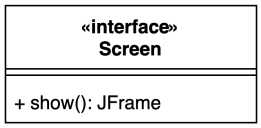
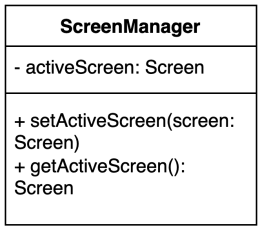
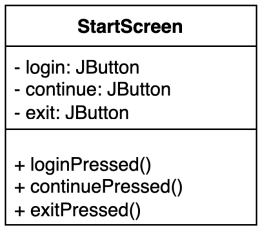
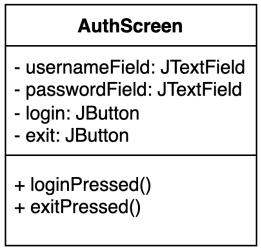

«interface»
Screen
Use
ScreenManager
StartScreen
AuthScreen
PrimaryScreen
AuthUser
PrimaryScreen
NormalUser
TutorManagement
AuthUser
TuteeManagement
AuthUser
CreateUserAuthUser
EditTutorAuthUser
EditTuteeAuthUser
FilterScreen
GeneralMgmt
AuthUser
EditPanel
CreateAptAuthUser
ViewCompleted
Hours
DataMgmt
Component
TutorHour
Component
UserMgmt
Component
«enum»
UserTypes

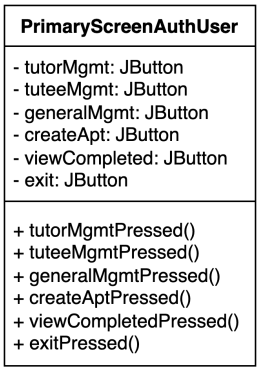
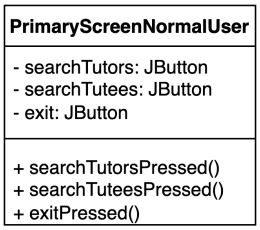
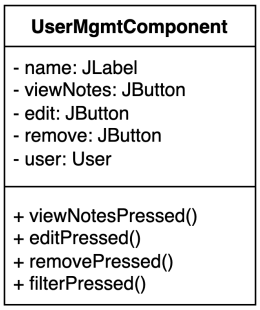

JCheckbox
Extends
DataCheckbox
«abstract»
User
«abstract»
GeneralDataType
Tutor
Tutee
TutoringSession
Skill
Proficiency
ClassName

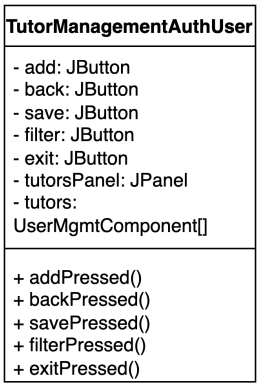
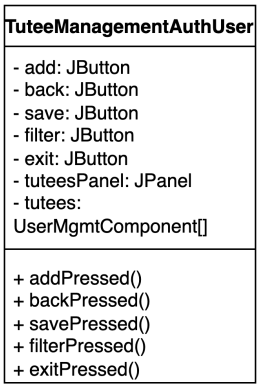
Data Detail Spreadsheet

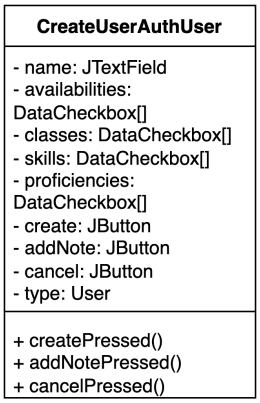
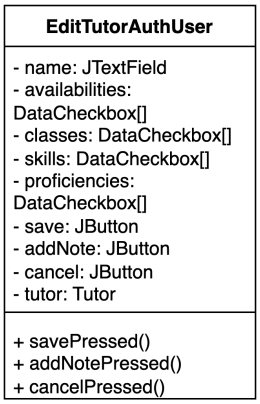
Item	Description	Data Type	Sample Data
User	An object that holds user data for tutors or tutees Overarching data structure for Tutor and Tutee		N/A - abstract
Tutor	An object that holds user data and tutor-specific information		Tutor: name: “John” availability: TutoringSession[] notes: [“Speaks Spanish”, “Fast worker”] priorClasses: ClassName[] skills: Skill[] proficiencies: Proficiency[]

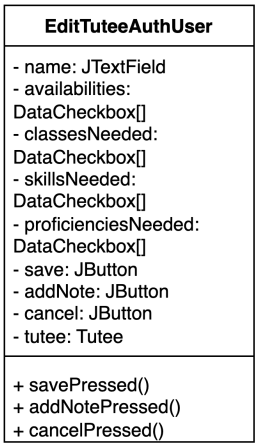
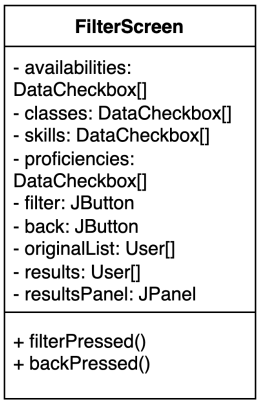
Tutee	An object that holds user data for tutee-specific information		Tutee: name: “Sarah” availability: TutoringSession[] notes: [“Needs help with time management”] classesNeeded: ClassName[] skillsNeeded: Skill[] proficienciesNeeded: Proficiency[]
UserTypes	Enum for a type of user (tutor or tutee), used when creating a new user		UserTypes.TUTOR = 0
GeneralDataType	Overarching data structure for TutoringSession, ClassName, Skill, and Proficiency		N/A - abstract
DataCheckbox	JCheckbox that holds information on a GeneralDataType		DataCheckbox data: new TutoringSession()
TutoringSession	An object that holds information about a tutoring session	 Extends GeneralDataType	TutoringSession: name: “Wed. Jan 5, 2024”
ClassName	An object that holds information about a class (course)	 Extends GeneralDataType	ClassName: name: “Algebra 2 HN”

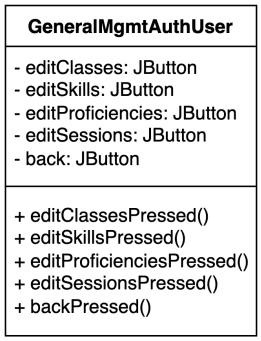
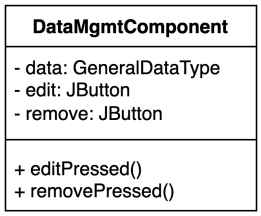
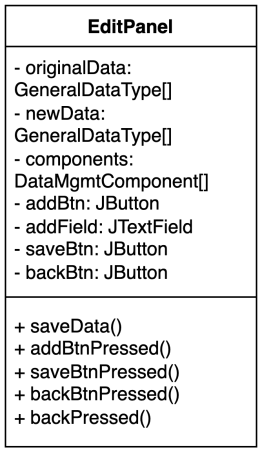
Skill	An object that holds information about a skill	 Extends GeneralDataType	Skill: name: “Essay Editing”
Proficiency	An object that holds information about a proficiency	 Extends GeneralDataType	Proficiency: name: “AWS Certified Cloud Practitioner”
Screen	An interface that holds the primary content of a panel or screen		N/A - interface
ScreenManager	A static class that manages the rendering of screens and the control flow of the program		ScreenManager: activeScreen: StartScreen
StartScreen	The initial screen of the program		StartScreen: login: JButton continue: JButton exit: JButton
AuthScreen	The authentication screen of the program which prompts for a username and password		AuthScreen: usernameField: JTextField passwordField: JTextField login: JButton exit: JButton

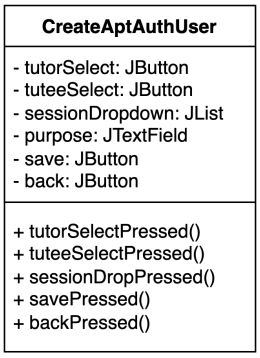
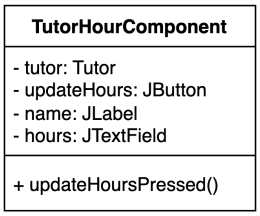
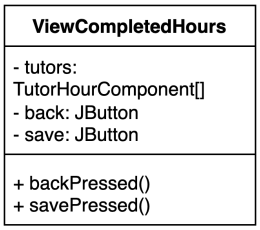
PrimaryScreenAuthUser	The primary home screen for an authenticated user after successful authentication		PrimaryScreenAuthUser: tutorMgmt: JButton tuteeMgmt: JButton generalMgmt: JButton createApt: JButton viewCompleted: JButton exit: JButton
PrimaryScreenNormalUser	The primary home screen for an unauthenticated user after pressing “continue” on the StartScreen		PrimaryScreenNormalUser: searchTutors: JButton searchTutees: JButton exit: JButton
UserMgmtComponent	A component holding the functionality buttons of each user (tutor/tutee) within a larger list		UserMgmtComponent: name: JLabel viewNotes: JButton edit: JButton remove: JButton user: User

TutorManagementAuthUser	The tutor management panel for an authenticated user allowing the user to add, edit, remove, and search for tutors		TutorManagementAuthUser: add: JButton back: JButton save: JButton filter: JButton exit: JButton tutorsPanel: JPanel tutors: [new UserMgmtComponent(new Tutor()), new UserMgmtComponent(new Tutor())]
TuteeManagementAuthUser	The tutee management panel for an authenticated user allowing the user to add, edit, remove, and search for tutees		TuteeManagementAuthUser: add: JButton back: JButton save: JButton filter: JButton exit: JButton tuteesPanel: JPanel tutees: [new UserMgmtComponent(new Tutee()), new UserMgmtComponent(new Tutee())]

CreateUserAuthUser	The screen to create a new user with input fields based on data for availabilities, classes, skills, and proficiencies		CreateUserAuthUser: name: JTextField availabilities: DataCheckbox[] classes: DataCheckbox[] skills: DataCheckbox[] proficiencies: DataCheckbox[] create: JButton addNote: JButton cancel: JButton type: UserTypes.TUTOR
EditTutorAuthUser	The edit panel for a tutor allowing name, availability, classes, skills, proficiencies, and notes to be edited		EditTutorAuthUser: name: JTextField availabilities: DataCheckbox[] classes: DataCheckbox[] skills: DataCheckbox[] proficiencies: DataCheckbox[] save: JButton addNote: JButton cancel: JButton Tutor: new Tutor()

EditTuteeAuthUser	The edit panel for a tutee allowing name, availability, classesNeeded, skillsNeeded, proficienciesNeeded, and notes to be edited		EditTuteeAuthUser: name: JTextField availabilities: DataCheckbox[] classesNeeded: DataCheckbox[] skillsNeeded: DataCheckbox[] proficienciesNeeded: DataCheckbox[] save: JButton addNote: JButton cancel: JButton Tutee: new Tutee()
FilterScreen	A panel allowing a list of User objects to be filtered by GeneralDataTypes		FilterScreen availabilities: DataCheckbox[] classes: DataCheckbox[] skills: DataCheckbox[] proficiencies: DataCheckbox[] filter: JButton back: JButton originalList: User[] results: User[] resultsPanel: JPanel

GeneralMgmtAuthUser	General management screen for an authenticated user to edit class, skill, proficiency, and session lists		GeneralMgmtAuthUser: editClasses: JButton editSkills: JButton editProficiencies: JButton editSessions: JButton back: JButton
DataMgmtComponent	A component holding the functionality buttons of each GeneralDataType within a list		DataMgmtComponent: data: new ClassName(“Math 8”) edit: JButton remove: JButton
EditPanel	A generic panel for editing different lists of GeneralDataTypes		EditPanel: originalData: [new ClassName(“Math 8”)] newData: [new ClassName(“Math 8”), new ClassName(“Math 9”)] components: [new DataMgmtComponent(new ClassName(“Math 8”))] addBtn: JButton addField: JTextField saveBtn: JButton backBtn: JButton

CreateAptAuthUser	Screen used to create an appointment between and tutor and tutee		CreateAptAuthUser: tutorSelect: JButton tuteeSelect: JButton sessionDropdown: JList purpose: JTextField save: JButton back: JButton
TutorHourComponent	A component holding the functionality buttons to edit each tutor's completed hours		TutorHourComponent: Tutor: new Tutor() updateHours: JButton name: JLabel hours: JTextField
ViewCompletedHours	Screen to view the total completed hours of all tutors		ViewCompletedHours: tutors: TutorHourComponent[] back: JButton save: JButton
JPanel startPanel	The JPanel holding the components of the start screen	JPanel object	JPanel startPanel = new JPanel()
JPanel authPanel	The JPanel holding the components of the authentication screen	JPanel object	JPanel authPanel = new JPanel()
JPanel primaryScreenAuthUserPanel	The JPanel holding the components of the primary authenticated screen	JPanel object	JPanel primaryScreenAuthUserPanel = new JPanel()

JPanel primaryScreenNormalUserPanel	The JPanel holding the components of the primary unauthenticated screen	JPanel object	JPanel primaryScreenNormalUserPanel = new JPanel()
JPanel tutorMgmtPanel	The JPanel holding the components of the tutor management screen	JPanel object	JPanel tutorMgmtPanel = new JPanel()
JPanel tuteeMgmtPanel	The JPanel holding the components of the tutee management screen	JPanel object	JPanel tuteeMgmtPanel = new JPanel()
JPanel addTutorPanel	The JPanel holding the components of the add tutor screen	JPanel object	JPanel addTutorPanel = new JPanel()
JPanel addTuteePanel	The JPanel holding the components of the add tutee screen	JPanel object	JPanel addTuteePanel = new JPanel()
JPanel editTutorPanel	The JPanel holding the components of the edit tutor screen	JPanel object	JPanel editTutorPanel = new JPanel()
JPanel editTuteePanel	The JPanel holding the components of the edit tutee screen	JPanel object	JPanel editTuteePanel = new JPanel()
JPanel searchPanel	The JPanel holding the components of the search screen	JPanel object	JPanel searchPanel = new JPanel()
JPanel generalManagementPanel	The JPanel holding the components of the general management screen	JPanel object	JPanel generalManagementPanel = new JPanel()
JPanel createAppointmentPanel	The JPanel holding the components of the create appointment screen	JPanel object	JPanel createAppointmentPanel = new JPanel()
JPanel editClassPanel	The JPanel holding the components of the edit class screen	JPanel object	JPanel editClassPanel = new JPanel()
JPanel editSkillPanel	The JPanel holding	JPanel object	JPanel editSkillPanel

	the components of the edit skill screen		= new JPanel()
JPanel editProficiencyPanel	The JPanel holding the components of the edit proficiency screen	JPanel object	JPanel editProficiencyPanel = new JPanel()
JPanel editSessionsPanel	The JPanel holding the components of the edit sessions screen	JPanel object	JPanel editSessionsPanel = new JPanel()
JPanel viewHoursPanel	The JPanel holding the components of the view hours screen	JPanel object	JPanel viewHoursPanel = new JPanel()
Scanner credentialsFileScanner	The scanner object to read login credentials	Scanner object	Scanner credentialsFileScanner = new Scanner(“credentials.txt”)
Scanner tutorsFileScanner	The scanner object to read tutor data	Scanner object	Scanner tutorFileScanner = new Scanner(“tutors.txt”)
Scanner tuteesFileScanner	The scanner object to read tutee data	Scanner object	Scanner tuteeFileScanner = new Scanner(“tutee.txt”)
Scanner classesFileScanner	The scanner object to read class data	Scanner object	Scanner classesFileScanner = new Scanner(“classes.txt”)
Scanner proficienciesFileScanner	The scanner object to read proficiencies data	Scanner object	Scanner proficienciesFileScanner = new Scanner(“proficiencies.txt”)
Scanner skillsFileScanner	The scanner object to read skills data	Scanner object	Scanner skillsFileScanner = new Scanner(“skills.txt”)
Scanner	The scanner object to	Scanner object	Scanner

sessionsFileScanner	read sessions data		sessionsFileScanner = new Scanner(“sessions.txt”)
JButton backButton	The button to go to the previous screen	JButton object	JButton backButton = new JButton(“Back”)
JButton saveButton	The button to save the current data	JButton object	JButton saveButton = new JButton(“Save”)
JButton filterButton	The button to filter tutors	JButton object	JButton filterButton = new JButton(“Filter”)
JButton addTutorButton	The button to add tutors	JButton object	JButton addTutorButton = new JButton(“Add tutor”)
JButton addTuteeButton	The button to edit tutees	JButton object	JButton addTuteeButton = new JButton(“Add tutee”)
JButton createAppointmentButton	The button to create an appointment	JButton object	JButton createAppointmentButton = new JButton(“Create appointment”)
JButton editTutorButton	The button to edit a tutor	JButton object	JButton editTutorButton = new JButton(“Edit tutor”)
JButton editTuteeButton	The button to edit a tutee	JButton object	JButton editTuteeButton = new JButton(“Edit tutee”)
JButton editClassesButton	The button to edit classes	JButton object	JButton editClassesButton = new JButton(“Edit classes”)
JButton editSkillsButton	The button to edit skills	JButton object	JButton editSkillsButton = new JButton(“Edit skills”)

JButton editProficienciesButton	The button to edit proficiencies	JButton object	JButton editProficienciesButton = new JButton(“Edit proficiencies”)
JButton editSessionsButton	The button to edit sessions	JButton object	JButton editSessionsButton = new JButton(“Edit sessions”)
JButton viewHoursButton	The button to view tutoring hours	JButton object	JButton viewHoursButton = new JButton(“View hours”)

Pseudocode

Figure #1: Module - Main

```
Initiate ScreenManager
Display StartScreen
```

Figure #2: Module - Start Screen

Algorithm: Show Screen

```
Display prompt asking for user selection
Input selection
If (selection is “login”)
        Display AuthScreen
Else if (selection is “continue”)
        Display PrimaryScreenNormalUser
Else if (selection is “exit”)
        Exit program
End if
```

Figure #3: Module - Auth Screen

Algorithm: Authenticate

```
Display prompt asking for user selection
Input selection

Declare input file “credentials.txt”
Load username from “credentials.txt”
Load password from “credentials.txt”

If (selection is “login”)
        Display prompt asking for username
        Input usernameInput
        If (usernameInput is not “”)
                Display prompt asking for password
                Input passwordInput
                If (passwordInput is not “”)
                        If (username is usernameInput and password is passwordInput)
                                Display PrimaryScreenAuthUser
                        Else
                                Display “Invalid credentials”
                        End if
                Else
                        Display “Empty password”
                End if
        Else
                Display “Empty Username”
```

```
        End if
Else if (selection is “back”)
        Display StartScreen
End if
```

Figure #4: Module - Primary Screen Normal User

Algorithm: Show Screen

```
Display prompt asking for user selection
Input selection
If (selection is “search tutors”)
        Declare input file “tutors.txt”
        Load tutorList from “tutors.txt”
        Display FilterScreen(tutorList)
Else if (selection is “search tutees”)
        Declare input file “tutees.txt”
        Load tuteeList from “tutees.txt”
        Display FilterScreen(tuteeList)
Else if (selection is “back”)
        Display StartScreen
End if
```

Figure #5: Module - Filter Screen

Algorithm: Update Filters

```
Declare input file “classes.txt”
Declare input file “skills.txt”
Declare input file “proficiencies.txt”
Declare input file “sessions.txt”
Load classesList from “classes.txt”
Load skillsList from “skills.txt”
Load proficienciesList from “proficiencies.txt”
Load sessionsList from “sessions.txt”

Display prompt asking for list of users
Input userList

Set filters to none
Set displayedUsers to userList
Display displayedUsers

Display prompt asking for user button click
Input buttonClicked
If (buttonClicked is “class”)
        add class to filters
Else if (buttonClicked is “skills”)
```

```
        add skill to filters
Else if (buttonClicked is “proficiencies”)
        add proficiency to filters
Else if (buttonClicked is “sessions”)
        add session to filters
Else if (buttonClick is “back”)
        Close FilterScreen popup
End if

Set displayedUsers to filter(userList, filters)
Display displayedUsers
```

Algorithm: filter

```
Input userList
Input filters
Loop user in userList
        Loop filter in filters
                // The user does not apply to the specified filters
                if (user[filter.type] does not contain filter)
                        Delete userList[user]
                        Exit loop
                End if
        End loop
End loop
Return userList
```

Figure #6: Module - Primary Screen Auth User

Algorithm: Show Screen

```
Display prompt asking for user selection
Input selection
If (selection is “tutor management”)
        Display TutorManagement()
Else if (selection is “tutee management”)
        Display TuteeManagement()
Else if (selection is “general management”)
        Display GeneralManagement()
Else if (selection is “create appointment”)
        Display CreateAppointment()
Else if (selection is “view hours”)
        Display ViewHours()
Else if (selection is “back”)
        Display StartScreen()
End if
```

Figure #7: Module - View Hours Screen

Algorithm: Show Screen

```
Declare input file “tutors.txt”
Load tutorData[] from “tutors.txt”
Loop tutor in tutorData[]
        Display tutor.hoursCompleted
End loop

Display prompt asking for user selection
Input selection
If (selection is “back”)
        Display PrimaryScreenAuthUser()
Else if (selection is “update hours”)
        Display prompt asking for updated hours using a textbox
        Input updatedHours from textbox
        Set tutorData[tutor].hoursCompleted to updatedHours
        Save tutorData
End if
```

Figure #8: Module - Create Appointment Screen

Algorithm: Create Appointment

```
Declare input file “tutorSessions.txt”
Load tutorSessions from “tutorSessions.txt”
Input tutor from FilterScreen
Input tutee from FilterScreen

Display prompt asking for session using dropdown
Input session from dropdown

Display prompt asking for purpose using textfield
Input purpose from textfield

Display prompt asking for user selection
Input buttonClicked

If (selection is “create”)
        If (tutor, tutee, session, and purpose are set)
                Add new TutorSession(tutor, tutee, session, purpose) to tutorSessions
                Save tutorSessions to “tutorSessions.txt”
        Else
                Display “Error: Inputs must be set”
        End if
Else if (selection is “back”)
        Display PrimaryScreenAuthUser()
```

```
End if
```

Figure #9: Module - General Management Screen

Algorithm: Show Screen

```
Display prompt asking for user selection
Input selection

Declare input file “classes.txt”
Declare input file “skills.txt”
Declare input file “proficiencies.txt”
Declare input file “sessions.txt”
Load classes from “classes.txt”
Load skills from “skills.txt”
Load proficiencies from “proficiencies.txt”
Load sessions from “sessions.txt”

If (selection is “edit classes”)
        Display EditPanel(classes)
Else if (selection is “edit skills”)
        Display EditPanel(skills)
Else if (selection is “edit proficiencies”)
        Display EditPanel(proficiencies)
Else if (selection is “edit sessions”)
        Display EditPanel(sessions)
Else if (selection is “back”)
        Display PrimaryScreenAuthUser()
End if
```

Figure #10: Module - Edit Panel

Algorithm: Save Data

```
Display prompt asking for user selection
Input selection

Input originalData
Input editedData
Input newData

If (selection is “update data”)
        Set originalData[editedData] to newData
Else if (selection is “remove data”)
        Delete originalData[editedData]
Else if (selection is “add data”)
        If (newData is not null)
                Add newData to originalData
```

```
        Else
                Display “Error: data is empty”
        End if
Else if (selection is “save data”)
        Save originalData to “${data.name}.txt”
        Display GeneralManagement()
Else if (selection is “back”)
        Display GeneralManagement()
End if
```

Figure #11: Method - Tutor Management Screen

Algorithm: Show Screen

```
Declare input file “tutors.txt”
Load tutors from “tutors.txt”

Display prompt asking for user selection
Input selection

If (selection is “view notes”)
        Display NotesPanel(buttonClicked.tutor)
Else if (selection is “edit tutor”)
        Display EditUser(buttonClicked.tutor)
Else if (selection is “remove tutor”)
        Loop tutor in tutors
                If (selection.tutor is tutor)
                        Delete tutors[tutor]
                End if
        End loop
Else if (selection is “filter”)
        Display FilterScreen(tutors)
Else if (selection is “save”)
        Save tutors to “tutors.txt”
Else if (selection is “back”)
        Display PrimaryScreenAuthUser()
Else if (selection is “add user”)
        Display prompt asking for newUsername
        Input newUsername
        If (newUsername is not null)
                add new Tutor(newUsername) to tutors
        Else
                Display “Error: Invalid field”
        End if
End if
```

Figure #12: Method - Tutee Management Screen

Algorithm: Show Screen

```
Declare input file “tutees.txt”
Load tutors from “tutees.txt”

Display prompt asking for user selection
Input selection

If (selection is “view notes”)
        Display NotesPanel(selection.tutee)
Else if (selection is “edit tutee”)
        Display EditUser(selection.tutee)
Else if (selection is “remove tutee”)
        Loop tutee in tutees
                If (selection.tutee is tutee)
                        Delete tutees[tutee]
                End if
        End loop
Else if (selection is “filter”)
        Display FilterScreen(tutees)
Else if (selection is “save”)
        Save tutors to “tutees.txt”
Else if (selection is “back”)
        Display PrimaryScreenAuthUser()
Else if (selection is “add user”)
        Display prompt asking for newUsername
        Input newUsername
        If (newUsername is not null)
                add new Tutee(newUsername) to tutees
        Else
                Display “Error: Invalid field”
        End if
End if
```

Figure #13: Method - Edit User Screen

Algorithm: User Edited

```
If (typeof User is Tutor)
        Declare input file “tutors.txt”
        Load userFiles from “tutors.txt”
Else
        Declare input file “tutees.txt”
        Load userFiles from “tutees.txt”
End if
Declare input file “classes.txt”
```

```
Declare input file “proficiencies.txt”
Declare input file “skills.txt”
Declare input file “sessions.txt”
Load classes from “classes.txt”
Load proficiencies from “proficiencies.txt”
Load skills from “skills.txt”
Load sessions from “sessions.txt”

Display prompt asking for user selection
Input selection
If (selection is “toggleCheckbox”)
        If (selection.checkbox is true)
                Set selection.checkbox to false
        Else
                Set selection.checkbox to true
        End if
        Set userFiles[user][buttonClicked.property] to buttonClicked.checkbox
Else if (selection is “add note”)
        Display user.notes
Else if (selection is “save”)
        If (typeof User is Tutor)
                Save userFiles to “tutors.txt”
        Else
                save userFiles to “tutees.txt”
        End if
End if
```