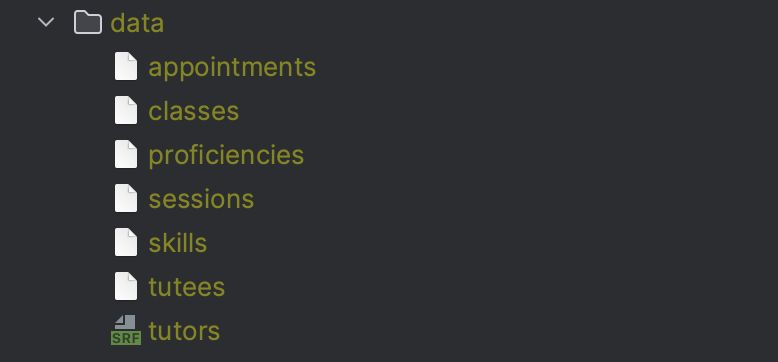
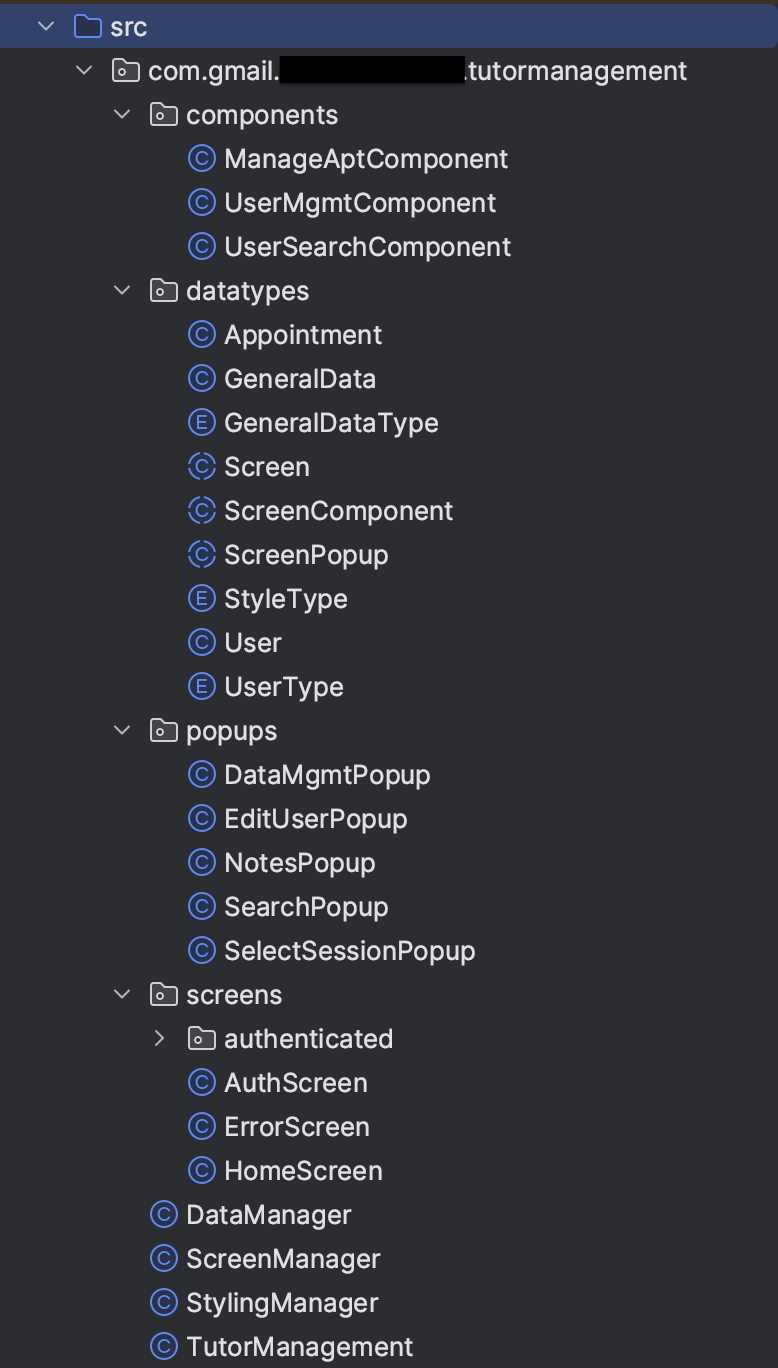
**Criterion C: Development**

This program was developed using the language Java.

The program aims to allow a user to easily create and manage tutors, tutees, and tutoring sessions between tutors and tutees from a graphical user interface (GUI). The user can add, remove, and edit both tutors and tutees and customize them with certain attributes. A tutor can be defined by the following: tutoring sessions they are available for, prior classes, skills, and proficiencies or certifications. A tutee can also be defined by the same attributes, but instead for classes they are seeking help in, skills they are looking to improve, etc. Preset lists of sessions, classes, skills, and proficiencies can managed by the user. After all the data is inputted, the user can look to schedule tutoring sessions between a tutor and tutee by filtering each by their attributes, finding the most appropriate combination based on several criteria. Appointments can be added, removed, and marked as completed, giving credit to the tutor for their work. Completed tutoring hours can be viewed by the user as well.

# File Structure

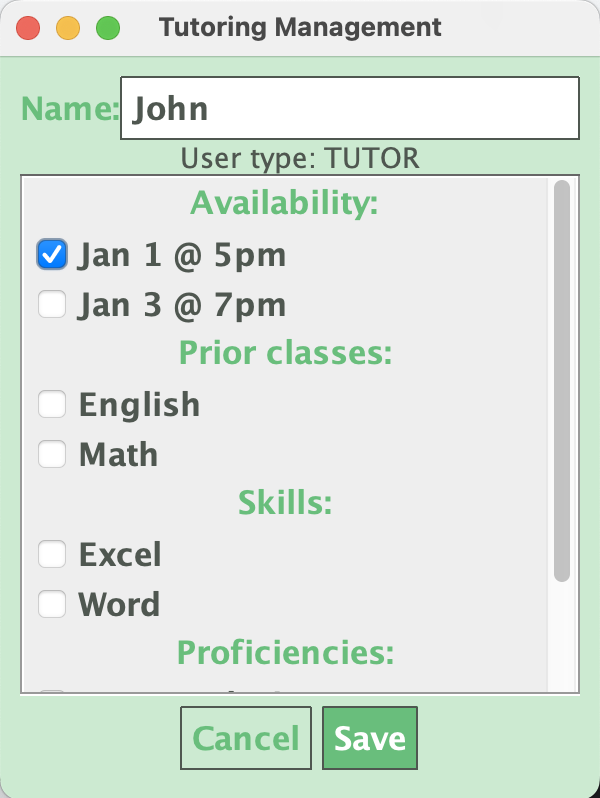


# Input

## Input Functions Table

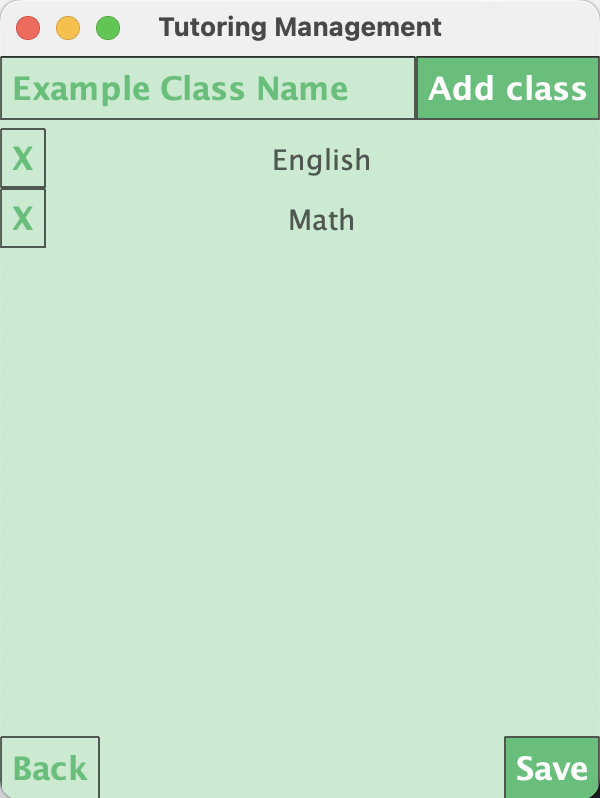
| **Function** | **Name** | **Usage** |
| --- | --- | --- |
| Create/edit tutor/tutee (#1) | EditUserPopup#SaveBtnPressed#actionPerformed() | The purpose of this function is to receive user input to create or update the data of a tutor or tutee. The output screen displays current (or blank, in the case of a new tutor or tutee) data which can be edited. Once the save button is pressed, this method is called and will save the updates. This can be used for both tutors and tutees |
| Create session, class, skill, or proficiency (#2) | DataMgmtPopup#AddBtnPressed#actionPerformed() | The purpose of this function is to add a new GeneralData object to the appropriate stored list by reading an input from a text box. GeneralData represents a session, class, skill, or proficiency. This function can be used for all of these attributes |
| Create appointment (#3) | CreateAptScreen#CreateAptBtnPressed#actionPerformed() | The purpose of this function is to add a new Appointment object to the stored appointment list. This function reads from inputs for a tutor, tutee, session, and duration and creates a new appointment |
| Delete tutor/tutee (#4) | UserMgmtComponent#RemoveBtnPressed#actionPerformed() | The purpose of this function is to remove a User object (tutor or tutee) from the appropriate stored list. This function can be used for both tutors and tutees |
| Delete session, class, skill, or proficiency (#5) | DataMgmtPopup#RemoveButtonPressed#actionPerformed() | The purpose of this function is to remove a GeneralData object from the appropriate stored list. This function can be used for sessions, classes, skills, or proficiencies |
| Delete appointment (#6) | ManageAptComponent#CancelBtnPressed#actionPerformed() | The purpose of this function is to remove an Appointment object from the stored list of appointments |
| Mark appointment as completed (#7) | ManageAptComponent#CompletedBtnPressed#actionPerformed() | The purpose of this function is to mark an Appointment object as complete. To do this, the program will add tutoring hours to the designated Tutor object of an Appointment object and then remove the Appointment object from the stored list of appointments |
| Input list of tutor/tutee filters (#8) | SearchPopup#FilterUpdated#actionPerformed() | The purpose of this function is to keep track of which filters the user wants to apply when searching for a tutor or tutee. This function will add and remove filters a list of active filters when their designated checkbox is toggled |
| Authenticate (#9) | AuthScreen#LoginBtnPressed#actionPerformed() | The purpose of this function is to authenticate the user by reading from the inputted username and password credentials. This function ensures the credentials are valid and allows the user to continue, otherwise will show an error message |

## Input #1 (Create/edit tutor/tutee)

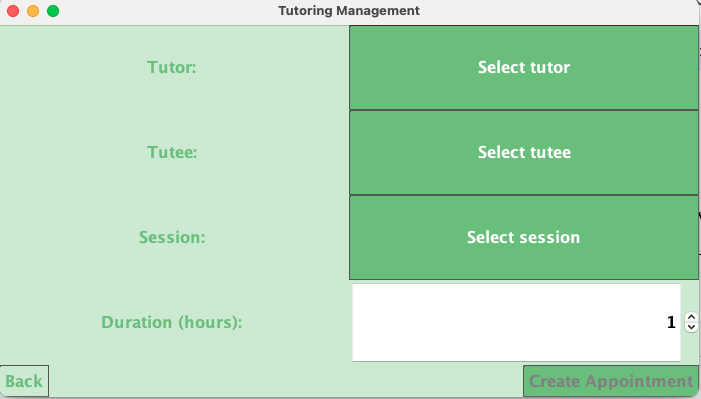


## 

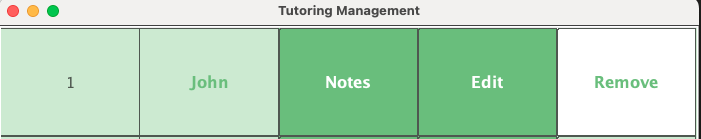
## Input #2 (Create session, class, skill, or proficiency)



## Input #3 (Create appointment)



## Input #4 (Delete tutor/tutee)



## 

## Input #5 (Delete session, class, skill, or proficiency)



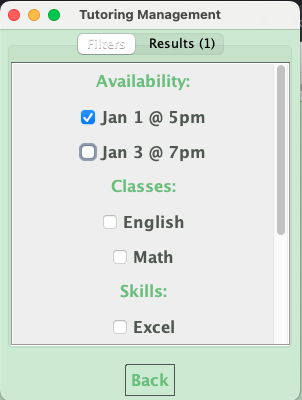
## Input #6 (Delete appointment)



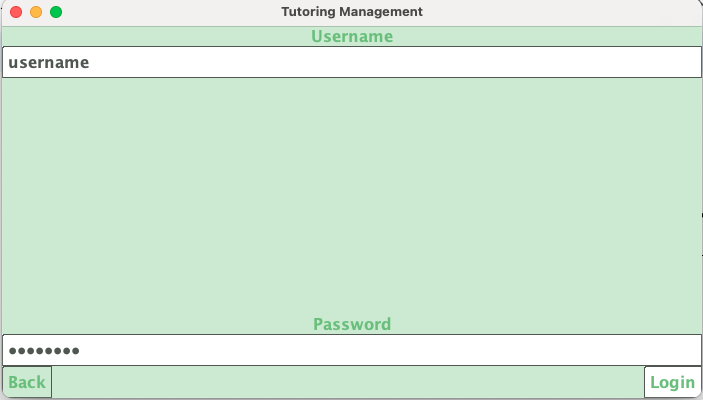
## Input #7 (Mark appointment as completed)



## Input #8 (Input list of tutor/tutee filters)



## Input #9 (Authenticate)

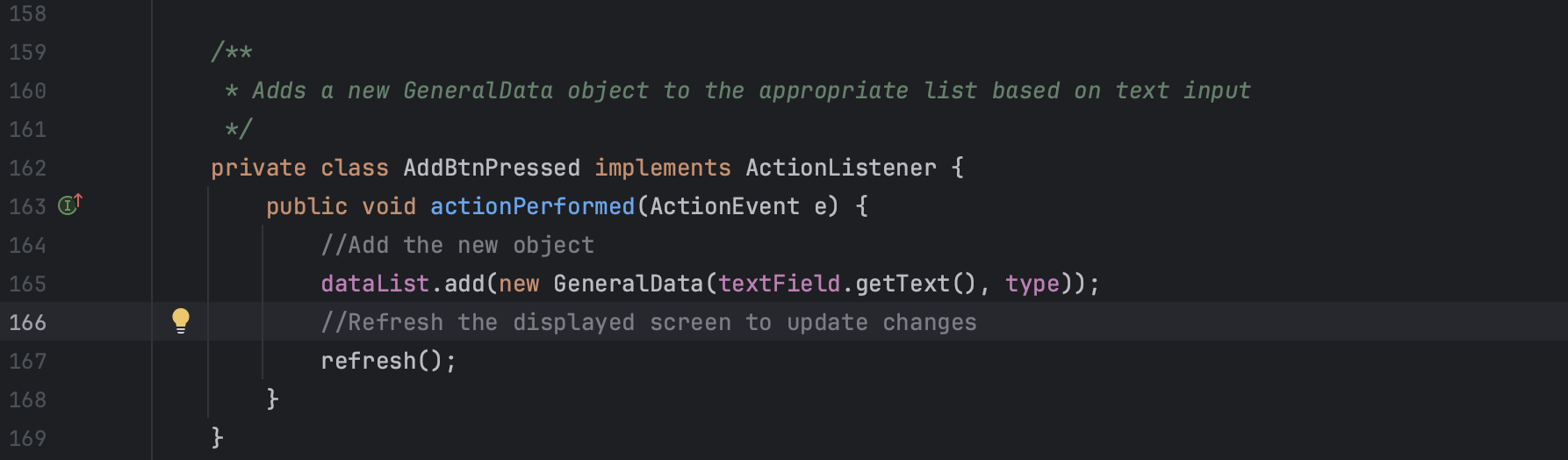


## Function #1 (Create/edit tutor/tutee)



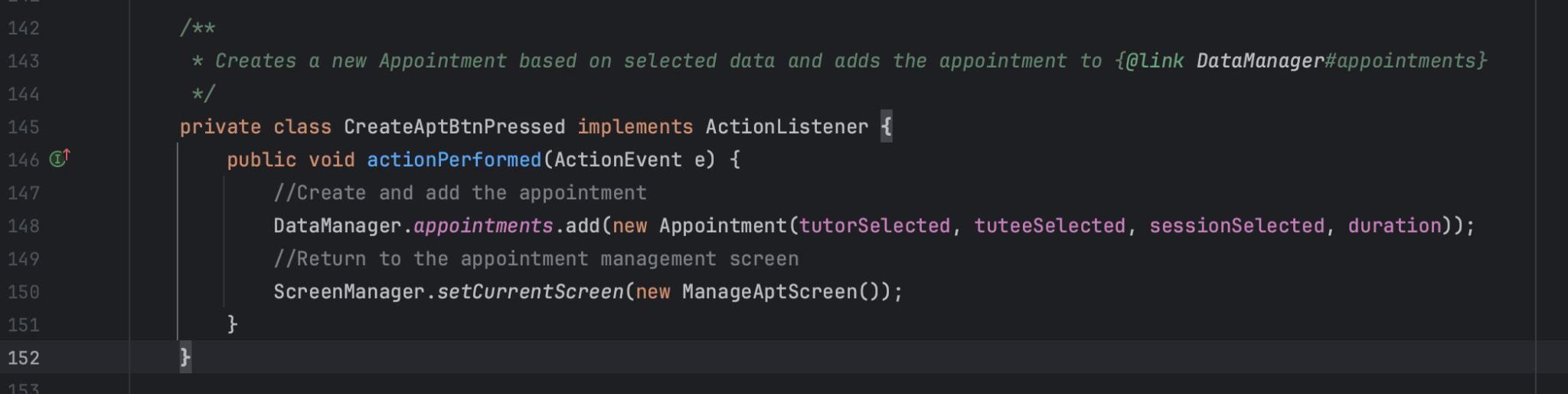
## 

## Function #2 (Create session, class, skill, or proficiency)



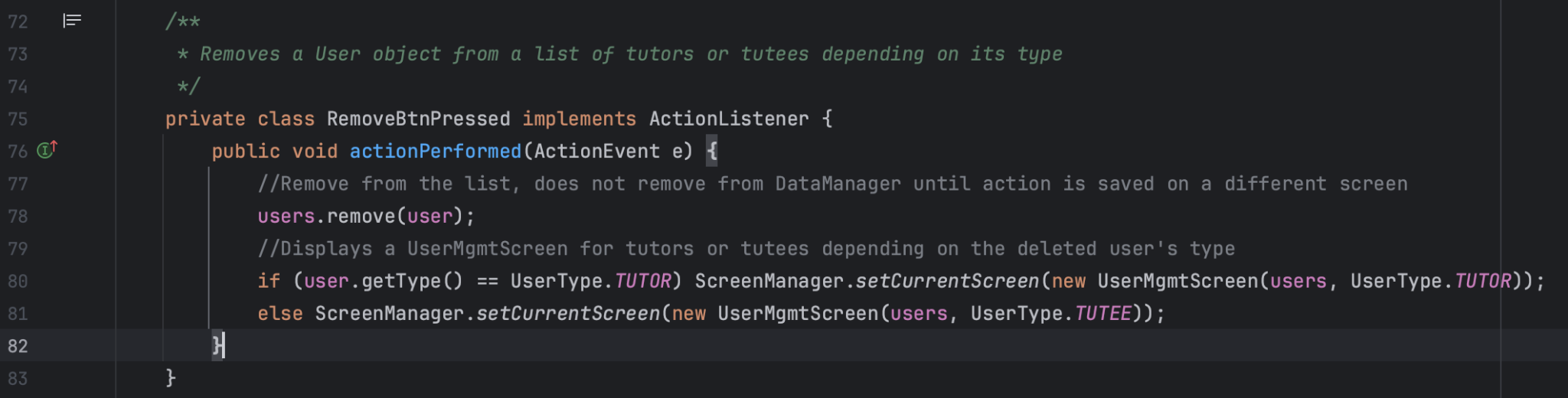
## 

## Function #3 (Create appointment)



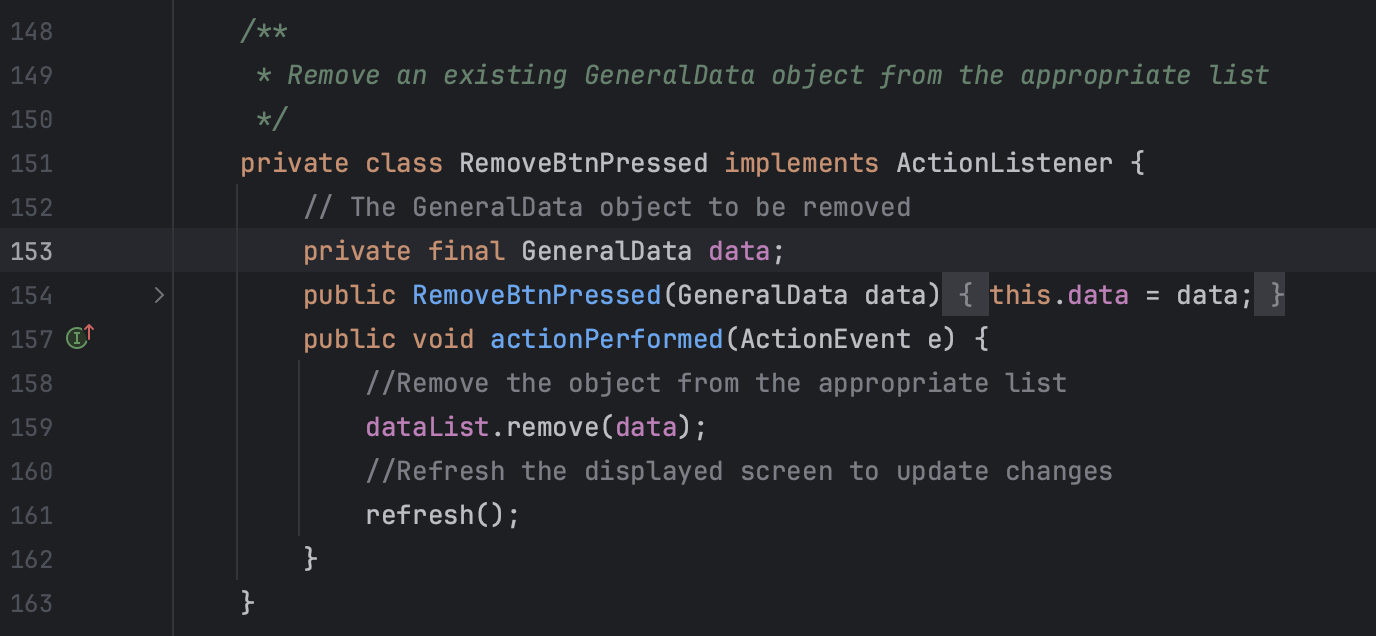
## 

## Function #4 (Delete tutor/tutee)



## 

## Function #5 (Delete session, class, skill, or proficiency)



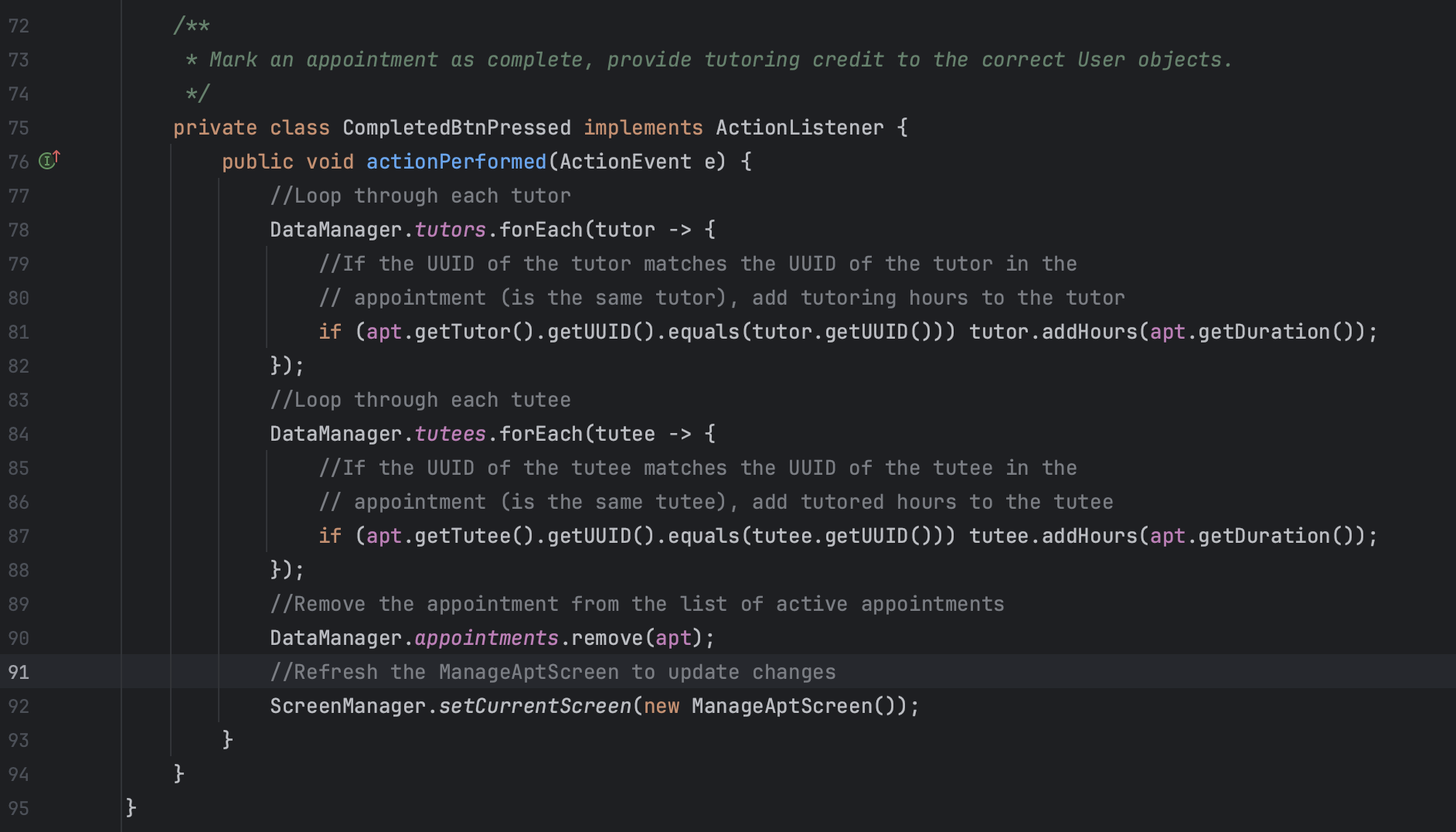
## 

## Function #6 (Delete appointment)

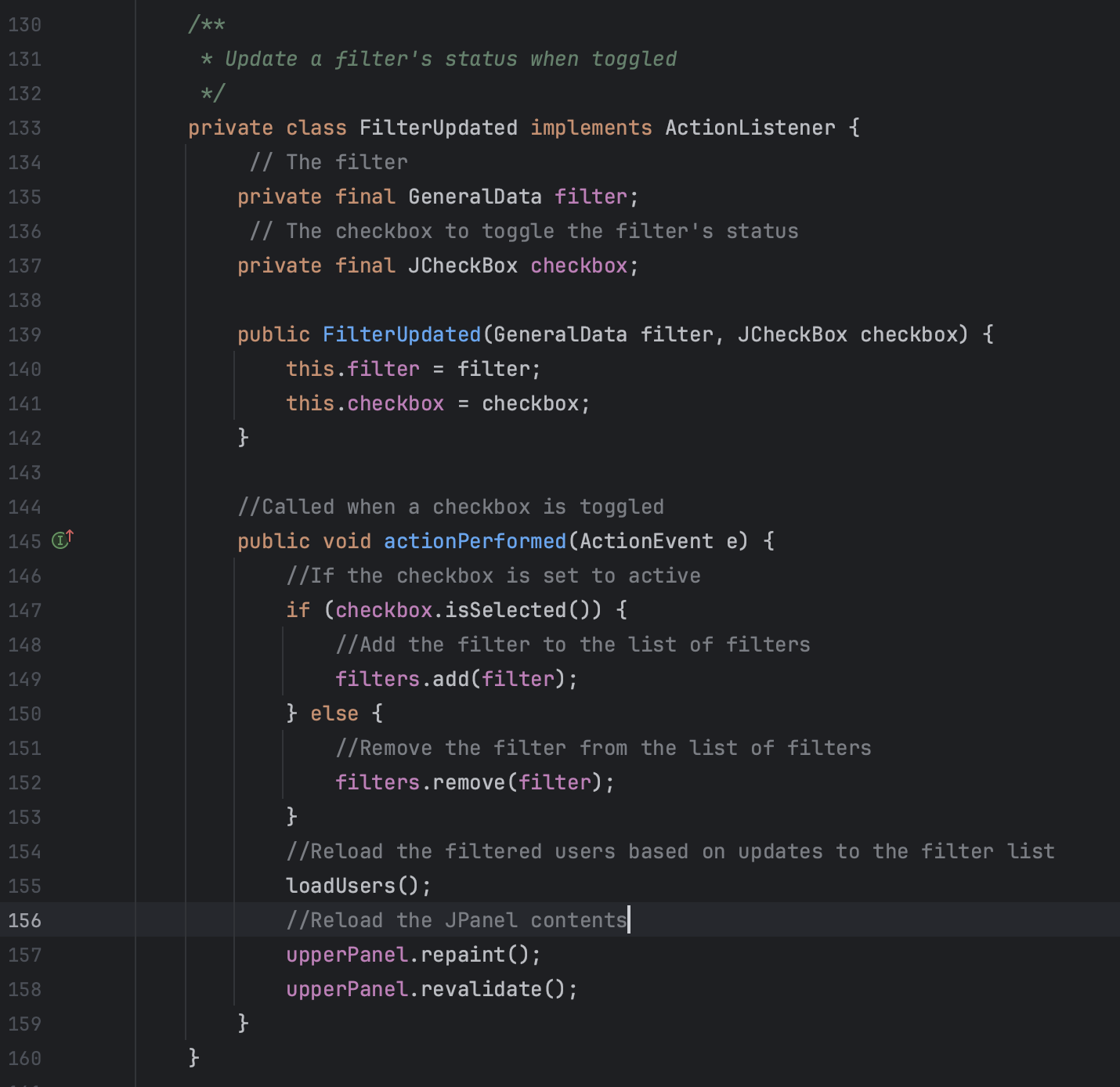


## 

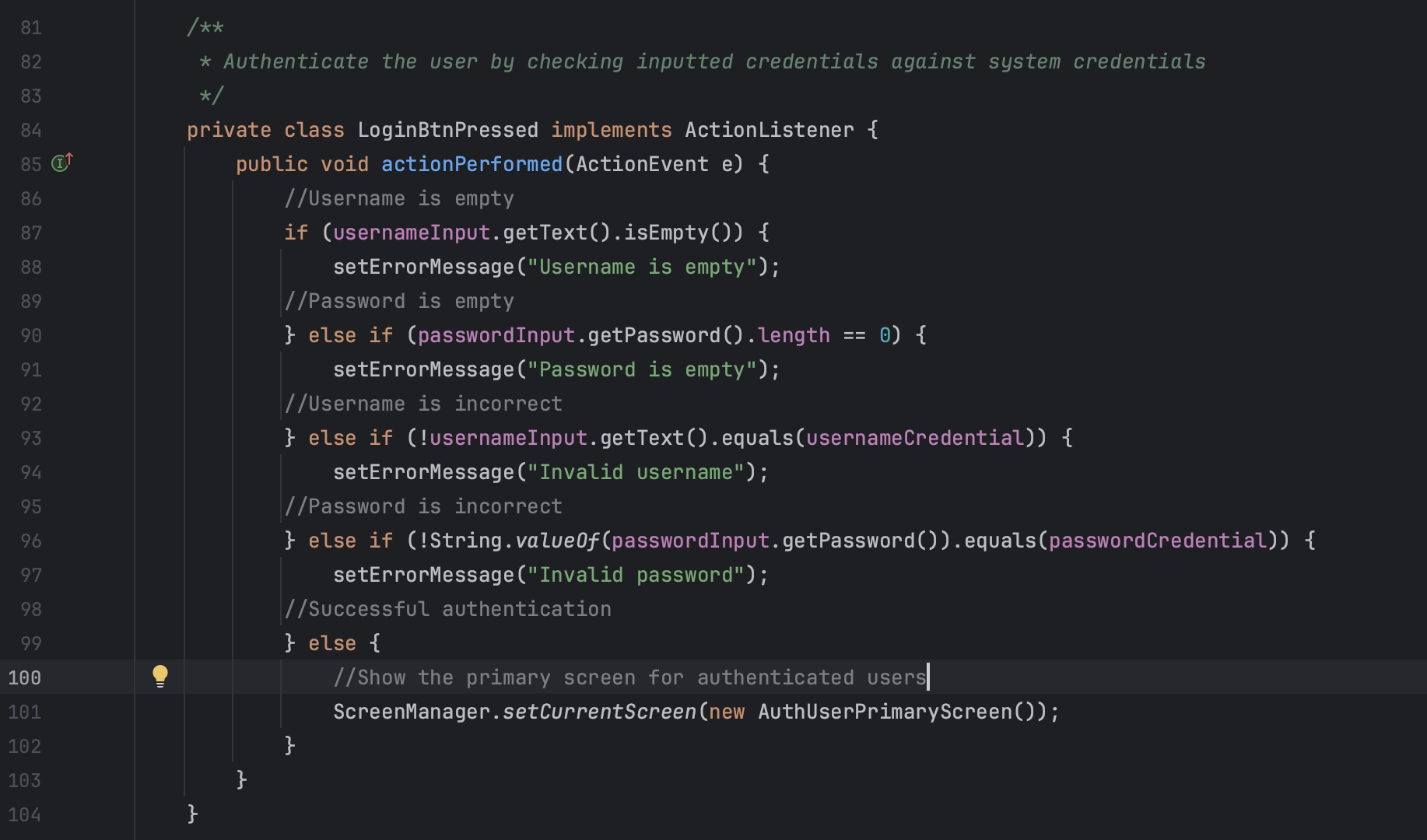
## Function #7 (Mark appointment as completed)



## Function #8 (Input list of tutor/tutee filters)



## Function #9 (Authenticate)



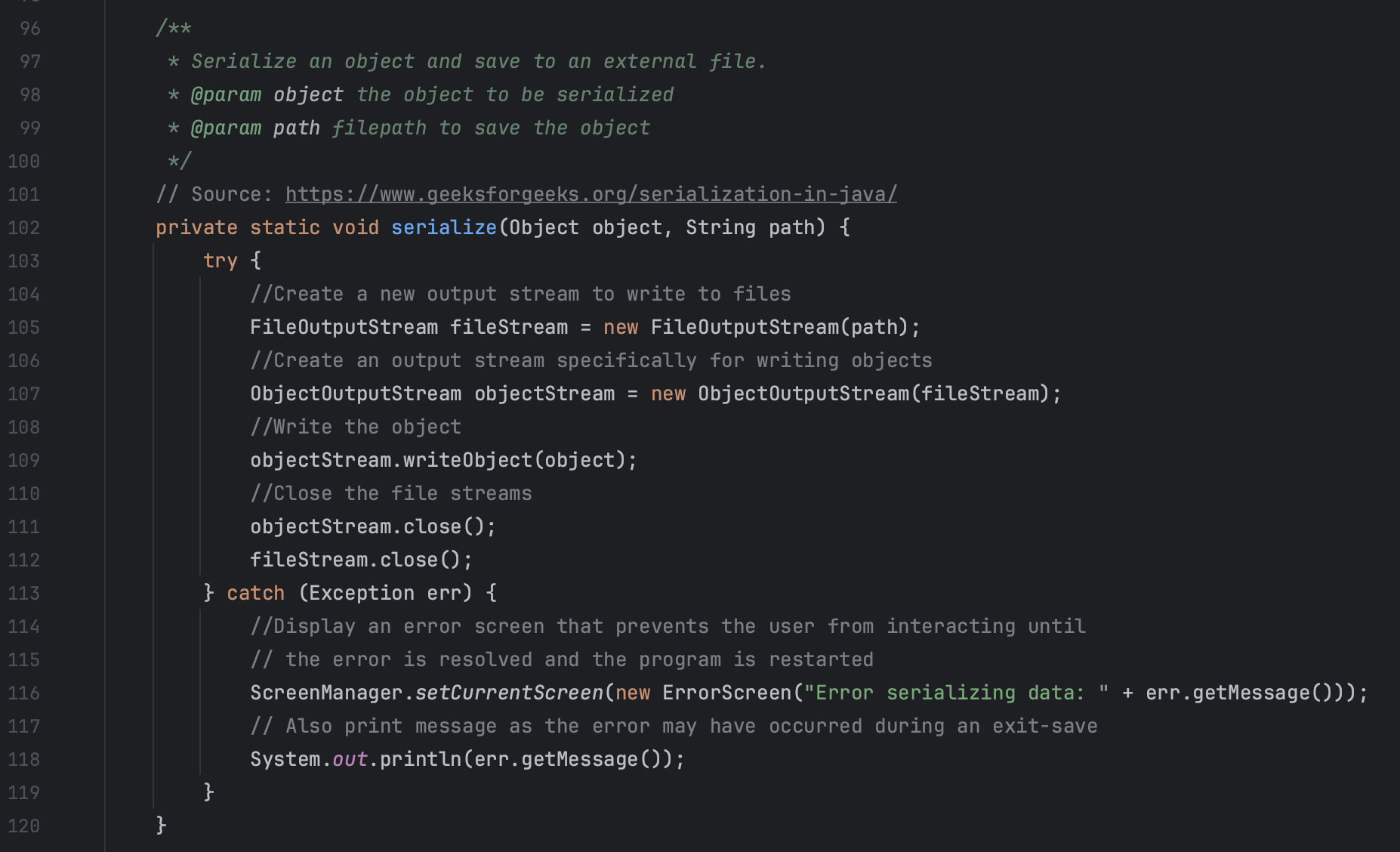
# 

# Processing

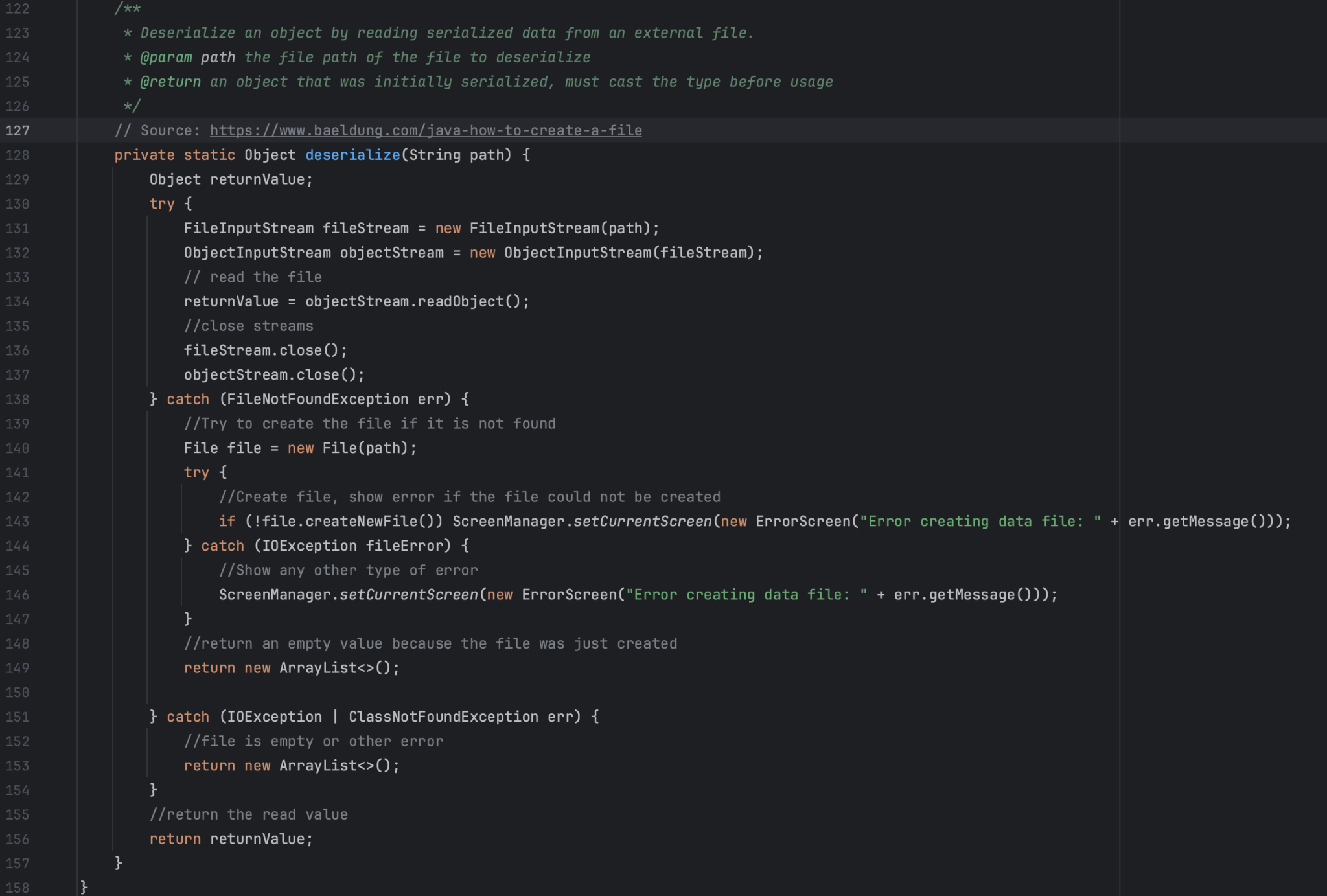
## Processing Functions Table

| **Function** | **Name** | **Usage** |
| --- | --- | --- |
| Serialize data (#10) | DataManager#serialize() | The purpose of this function is to serialize an object and save it to an external file. This serialization method is used to save objects directly and easily |
| Deserialize data (#11) | DataManager#deserialize() | The purpose of this function is to deserialize an object and read it from an external file. This deserialization method is used to convert serialized data into usable objects |
| Filter user (#12) | User#isIncluded() | The purpose of this function is to check if a user is included in a list of filters. The user must contain all of the attributes in the list to be included |
| Show screen (#13) | ScreenManager#setCurrentScreen() | This function takes a Screen object and displays it to the user on top of the program’s JFrame. Will remove the previous screen and close any outstanding popups |
| Show popup (#14) | ScreenManager#showPopup() | Similar to showing a screen, this function will take in a ScreenPopup object and display it on top of the current screen. Will remove the previous popup (if applicable) |
| Stylize component (#15) | StylingManager#stylize() | Stylizes a component according to a StyleType (primary or secondary). Will set various backgrounds, fonts, and text colors for each type of component. Method overloading is used to implement each type of component needed |

## Function #10 (Serialize data)

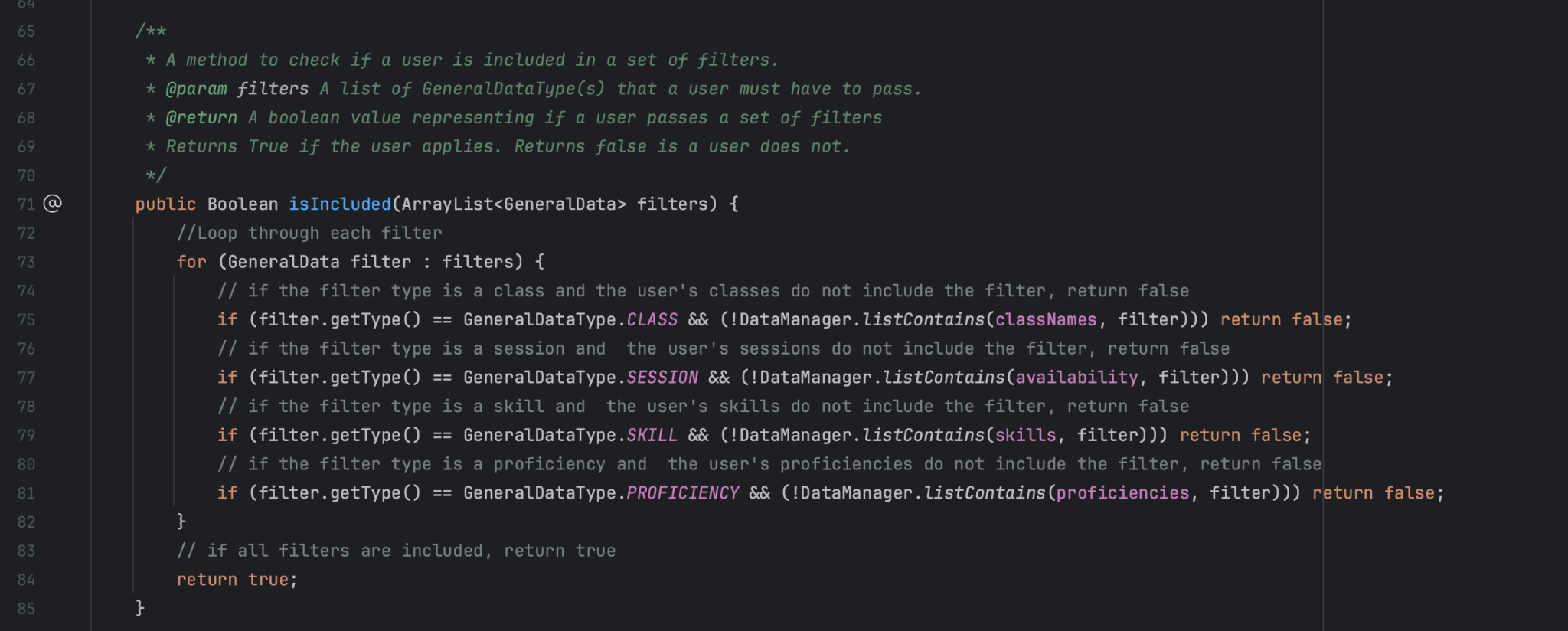


## Function #11 (Deserialize data)



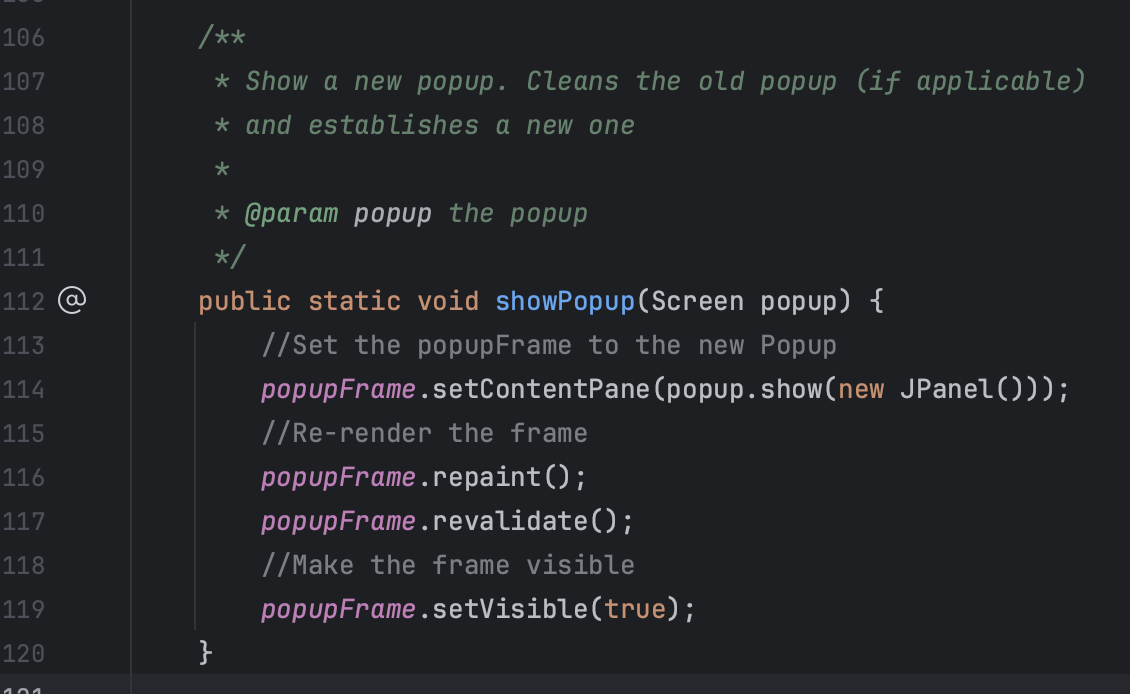
## 

## Function #12 (Filter user)

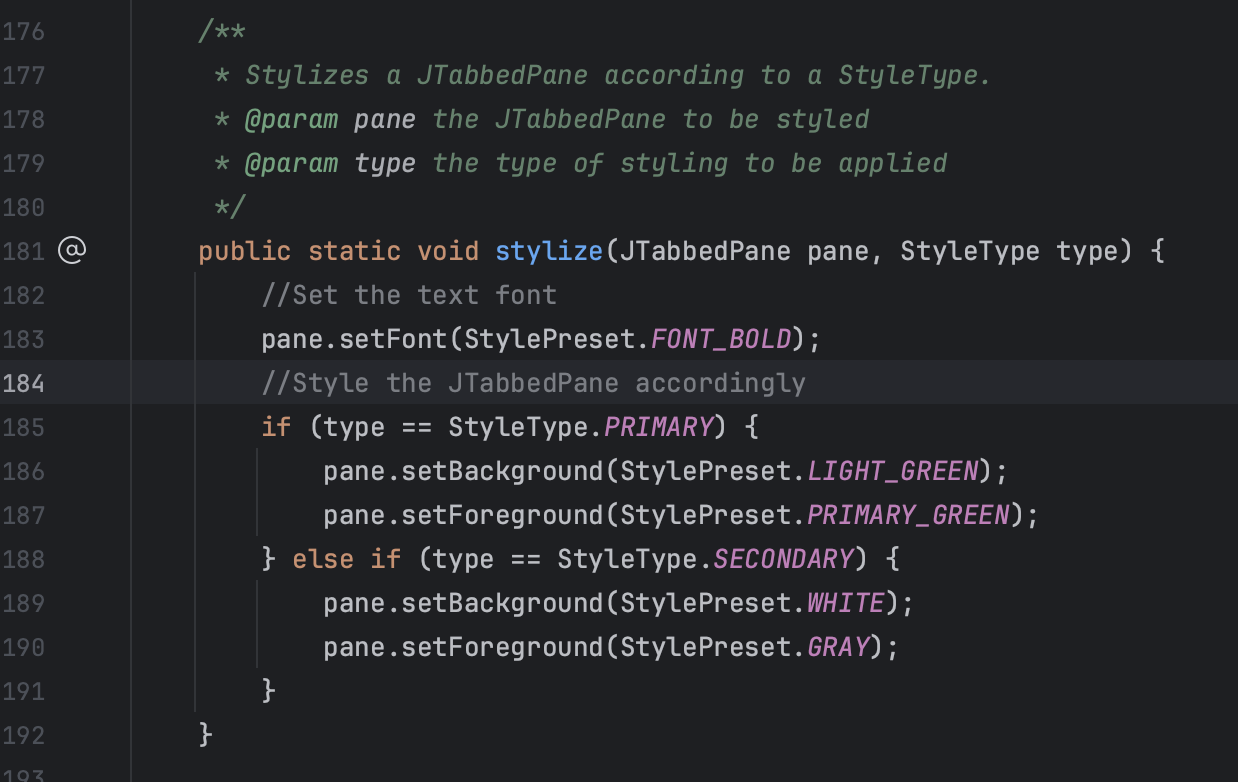
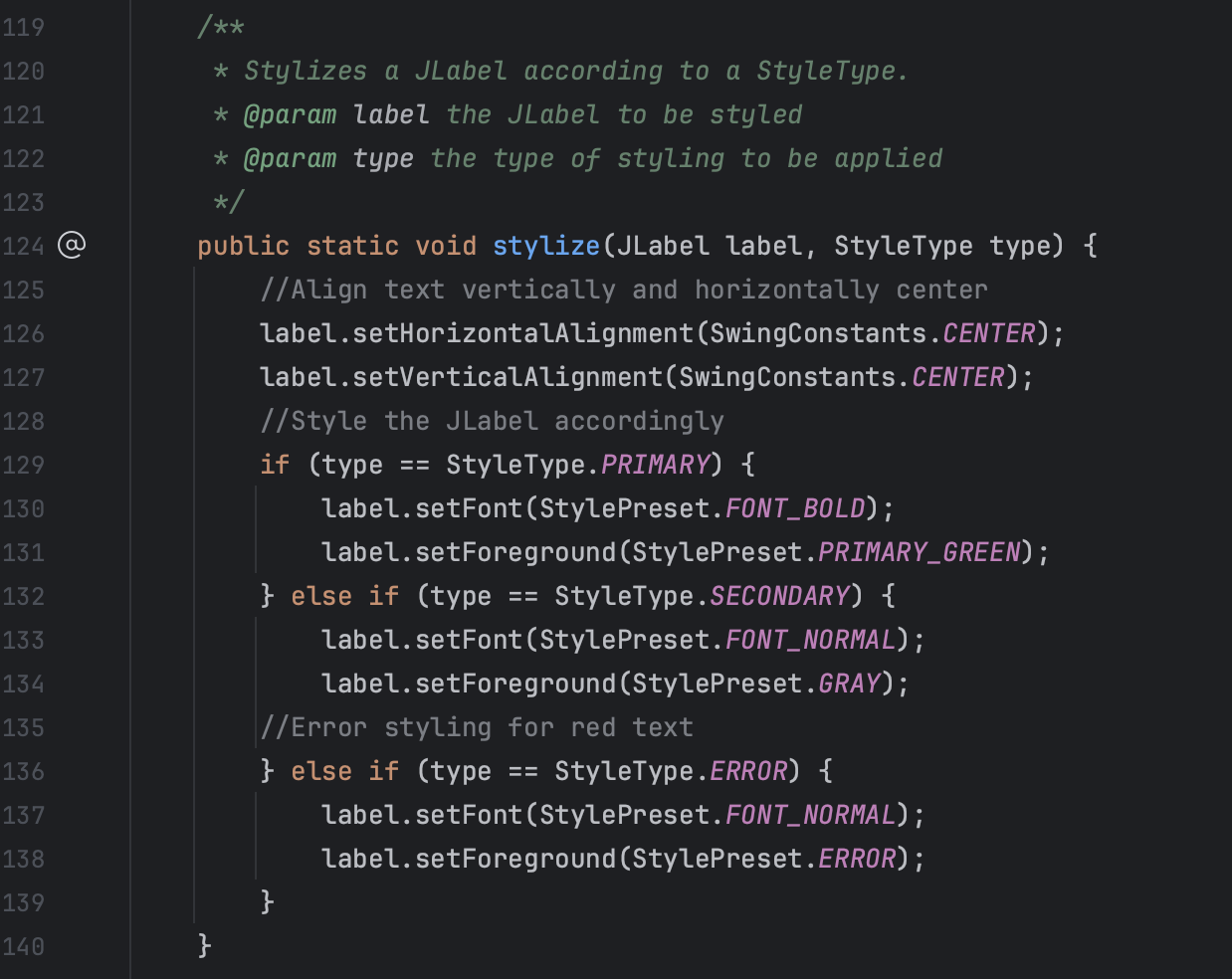
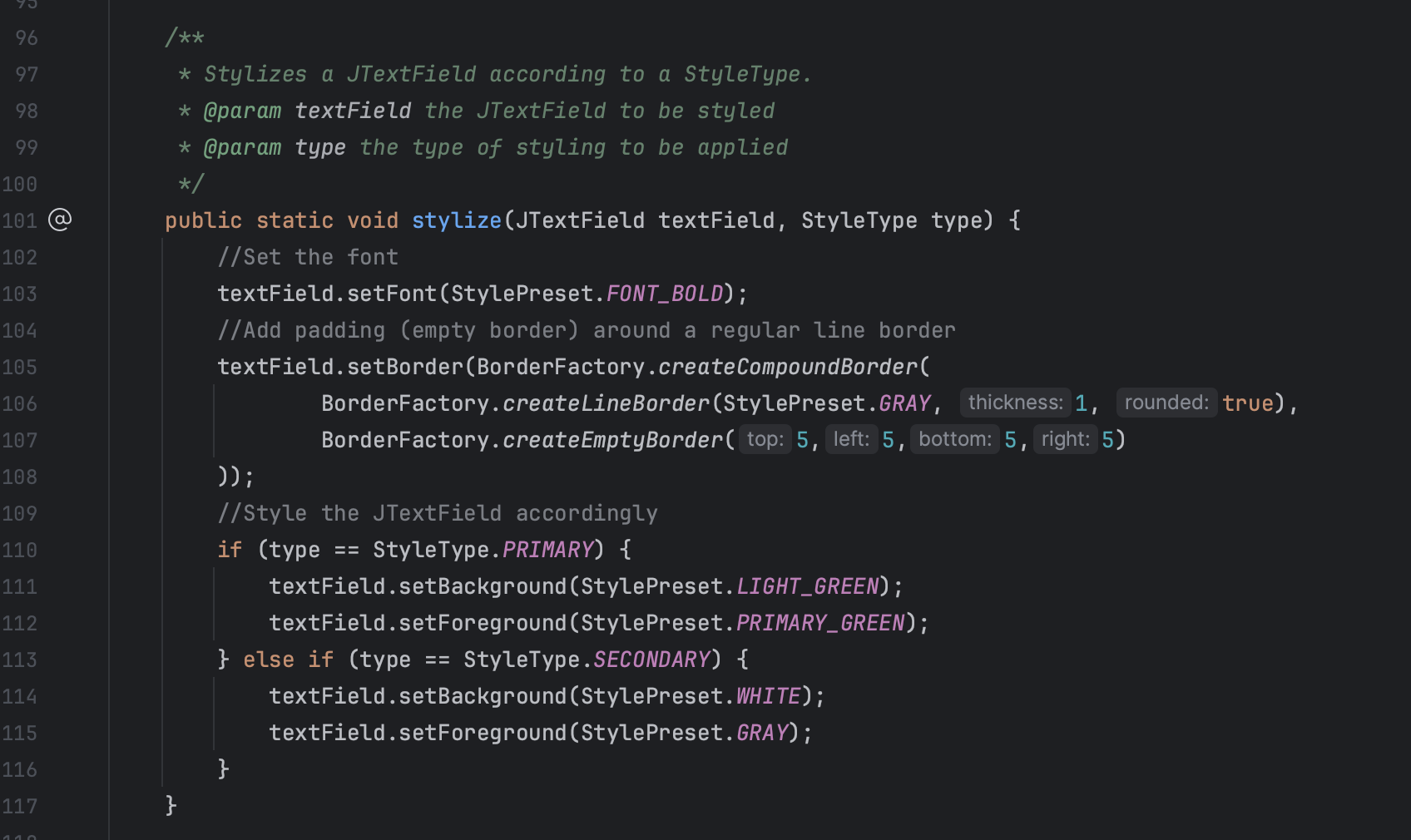
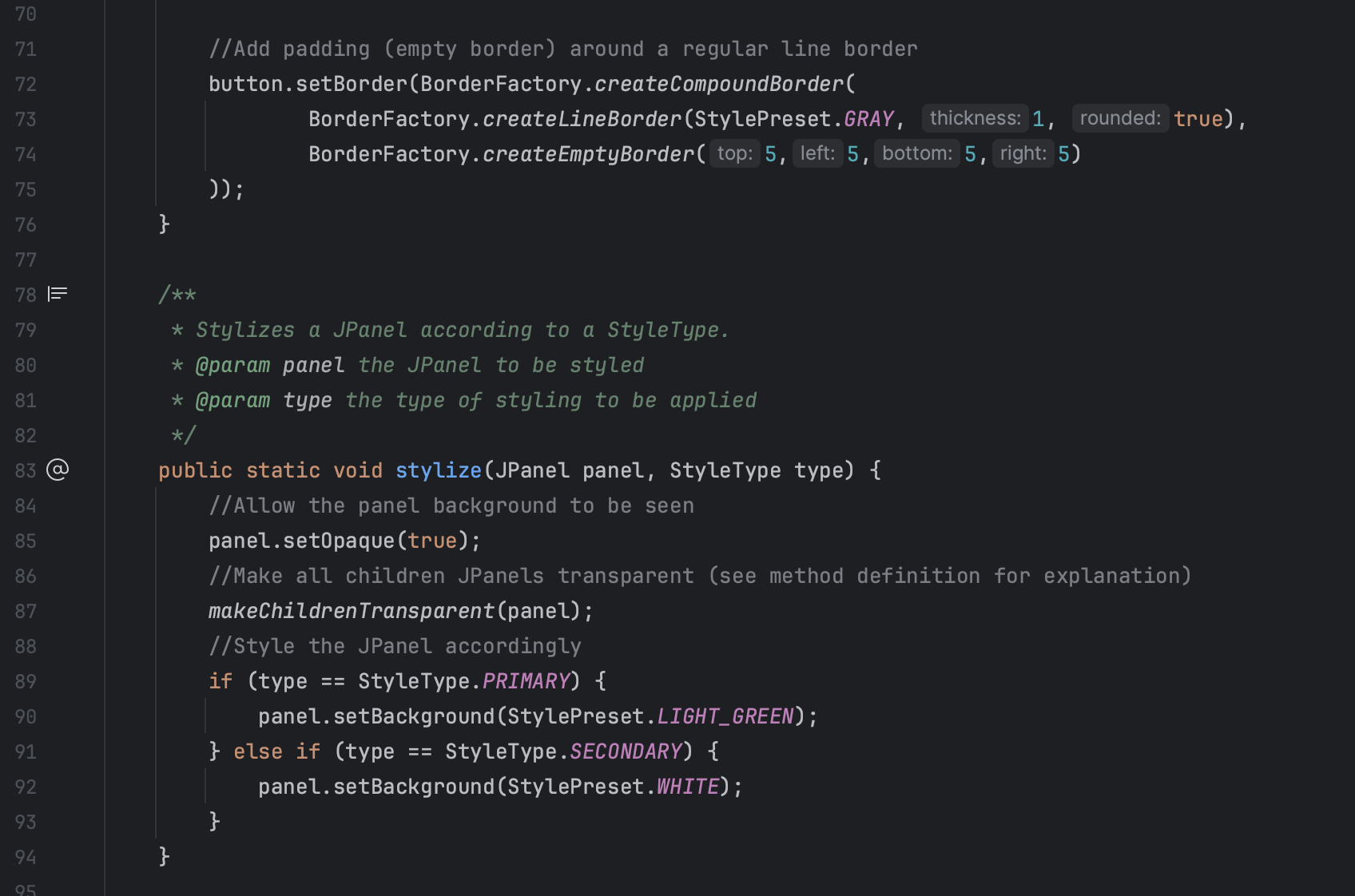


## Function #13 (Show screen)

## Function #14 (Show popup)



## Function #15 (Stylize component)

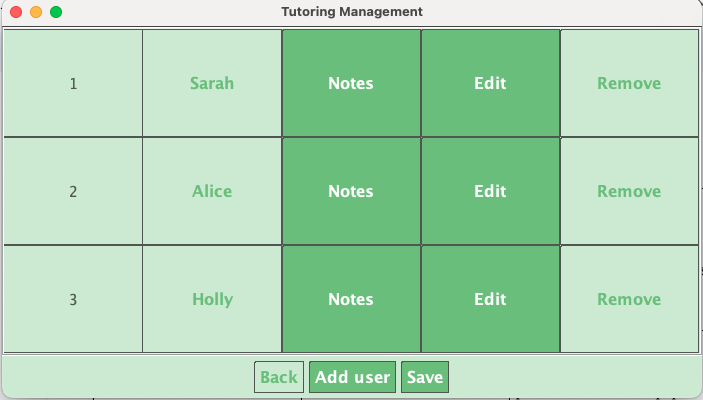


# Output

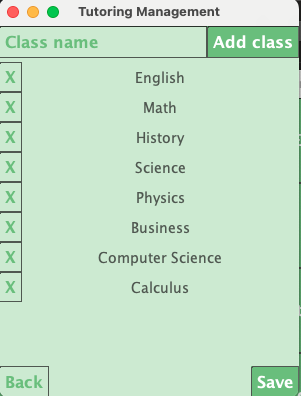
## Output Functions Table

| **Function** | **Name** | **Usage** |
| --- | --- | --- |
| Display list of tutors/tutees (#16) | UserMgmtScreen#show() | The purpose of this function is to display the list of tutors or tutees. This function can render the management screen for tutors and tutees depending on its inputs. The function allows the user to interact with other functions of the program to manage tutors and tutees as well |
| Display list of sessions/classes/skills/proficiencies (#17) | DataMgmtPopup#show() | The purpose of this function is to display the preset list of GeneralDataType(s). This function can render the management popup for a Session, ClassName, Skill, or Proficiency depending on its inputs. The function allows the user to interact with other functions of the program to manage these as well |
| Display list of appointments (#18) | ManageAptScreen#show() | The purpose of this function is to display the currently created Appointment objects and allow the user to interact with them |
| Display list of completed tutoring hours (#19) | ViewHoursScreen#show() | The purpose of this function is to display the completed tutoring hours each for each tutor and the number of hours of tutoring each tutee has received |
| Display list of filtered tutors or tutees (#20) | SearchPopup#loadUsers() | The purpose of this function is to populate the results panel of the SearchPopup screen with all users that are included in the currently selected filters. Filters can be managed on the same screen and results are updated in real-time |

## Output #16 (Diplay list of tutors/tutees)



## Output #17 (Display list of sessions/classes/skills/proficiencies)



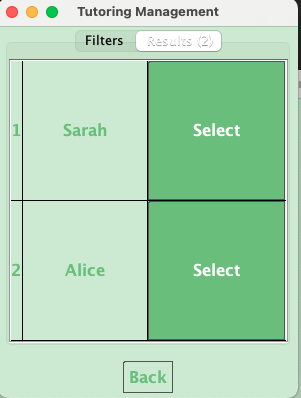
## Output #18 (Display list of appointments)



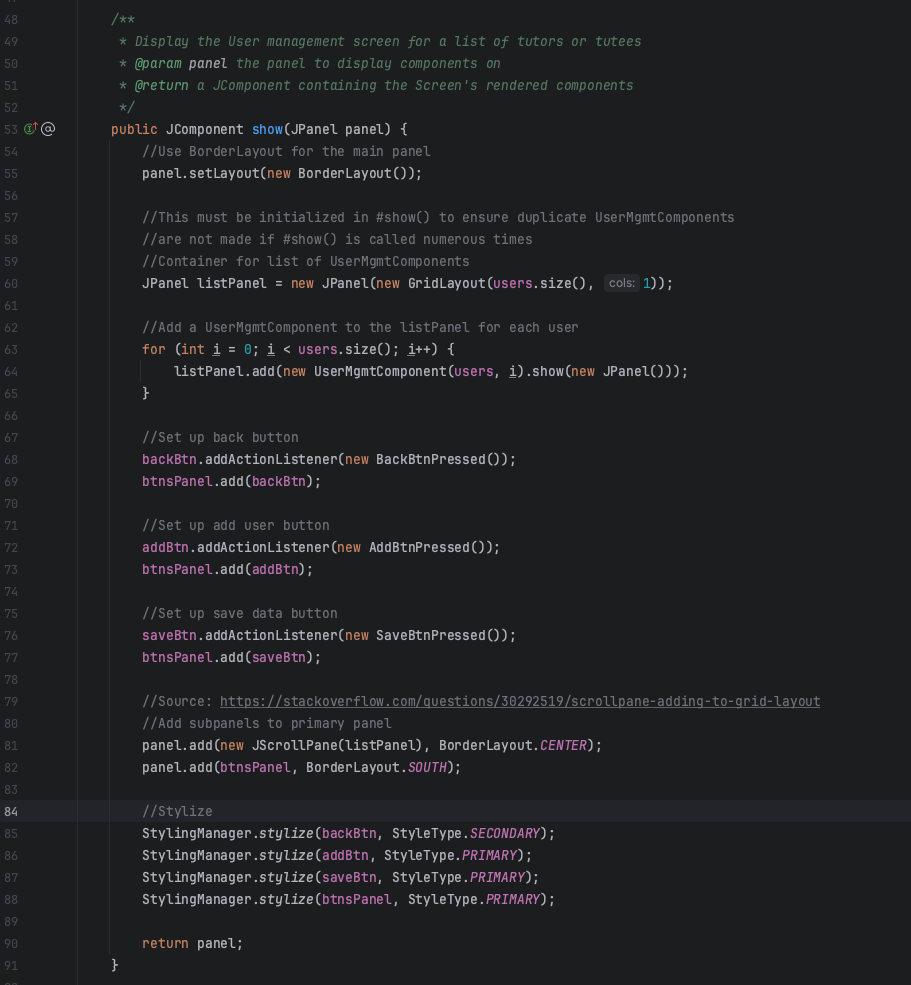
## Output #19 (Display list of completed tutoring hours)



## Output #20 (Display list of filtered tutors or tutees)



## Function #16 (Display list of tutors/tutees)



## Function #17 (Display list of sessions/classes/skills/proficiencies)



# 

# 

## Function #18 (Display list of appointments)

## Function #19 (Display list of completed tutoring hours)

## Function #20 (Display list of filtered tutors or tutees)



# 

# 

# Data Structures

| **Name** | **Purpose** |
| --- | --- |
| Appointment (#21) | The purpose of the Appointment data class is to store the data a created appointment. This class stores the appointment’s tutor, tutee, session, and duration. This data type is serializable |
| GeneralData (#22) | The purpose of GeneralData is to store the information for a session, class, proficiency, or skill. GeneralData can be used to represent each type, with GeneralDataType being used to differentiate between each. This data type is serializable |
| GeneralDataType (#23) | GeneralDataType is an enum that represents each type of GeneralData. The values are: CLASS, PROFICIENCY, SKILL, and SESSION |
| Screen (#24) | Screen is an abstract class and serves the purpose of being a model for all other screens displayed throughout the program |
| ScreenComponent (#25) | ScreenComponent is an abstract class that extends Screen and serves the same purpose as screen, a different class was created to have a logical separation between a Screen and ScreenComponent which serve different purposes |
| ScreenPopup (#26) | ScreenPopup is an abstract class that extends Screen and serves the same purpose as screen, a different class was created to have a logical separation between a Screen and ScreenPopup which serve different purposes |
| StyleType (#27) | StyleType is an enum which represents the different stylization presets that can be used with StyleManager. The values are: PRIMARY, SECONDARY, and ERROR |
| User (#28) | The purpose of the User object is to store the data of a tutor or tutee. User objects can be used for both tutors and tutees by specifying a UserType. A user object holds ArrayLists of each type of GeneralData (sessions, classes, skills, and proficiencies) and a unique user identifier (UUID) to ensure similar User objects are unique. This data type is serializable |
| UserType (#29) | UserType is an enum which represents the type of user that a User object is. The values are: TUTOR and TUTEE |

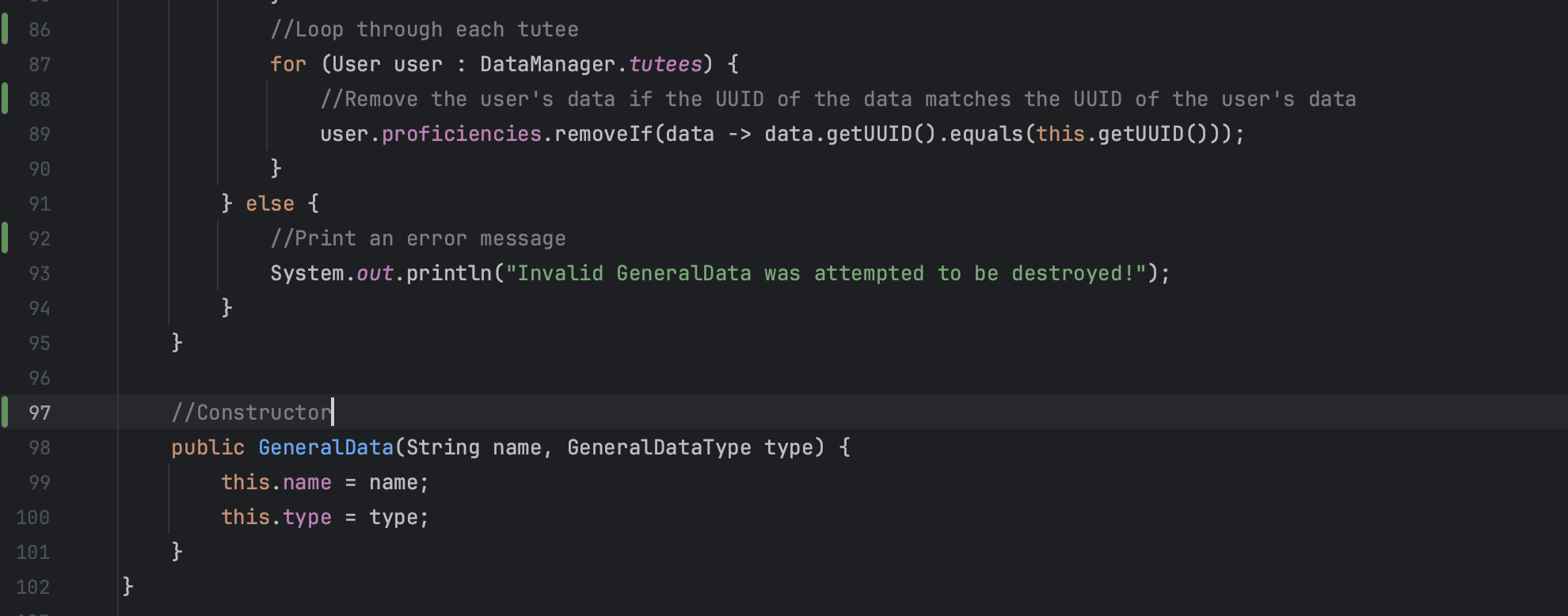
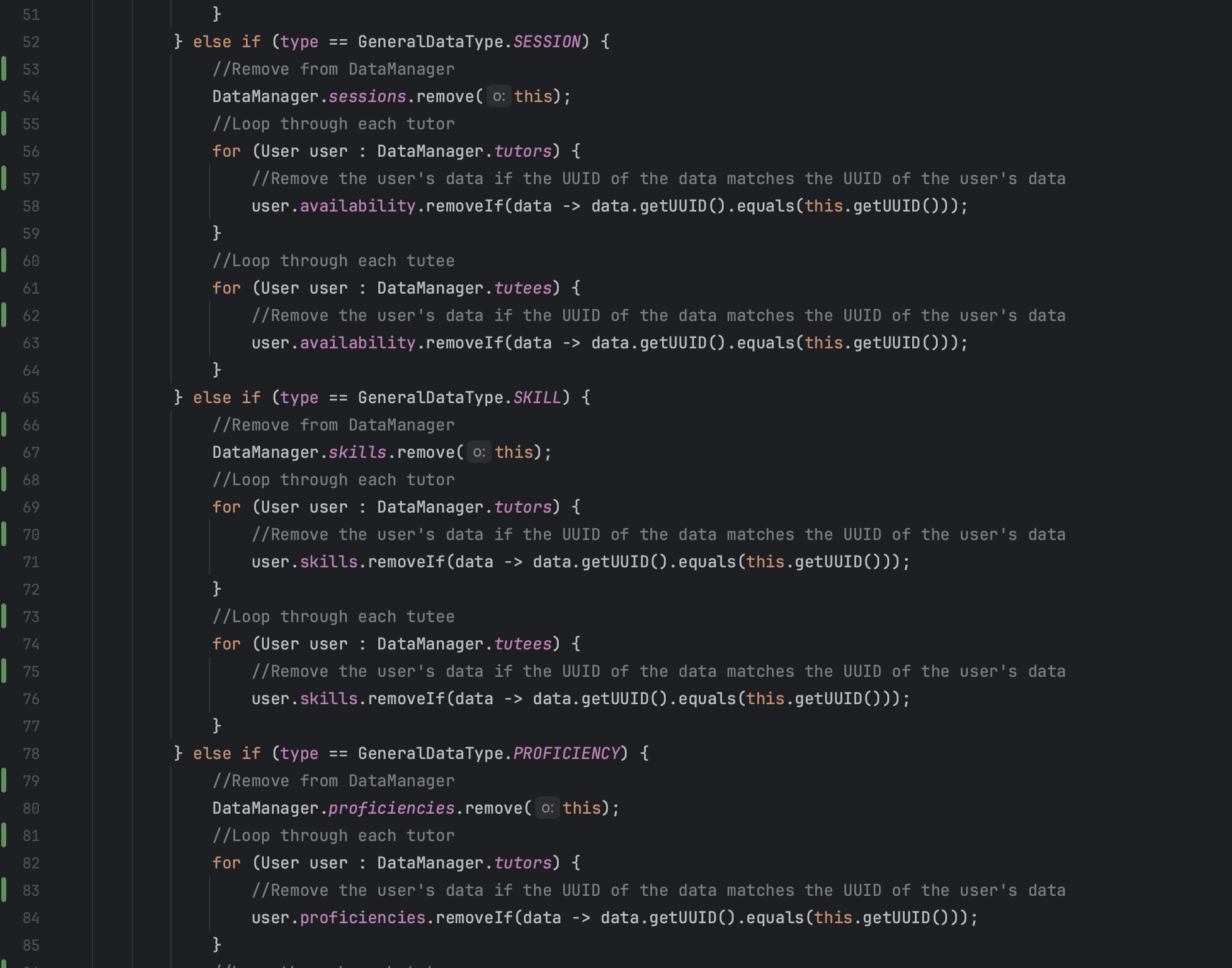
*\*Note that sample data files are not provided as data is stored in a serialized format. This format is not human-readable and the programmatic structure of their classes is provided instead.*

## 

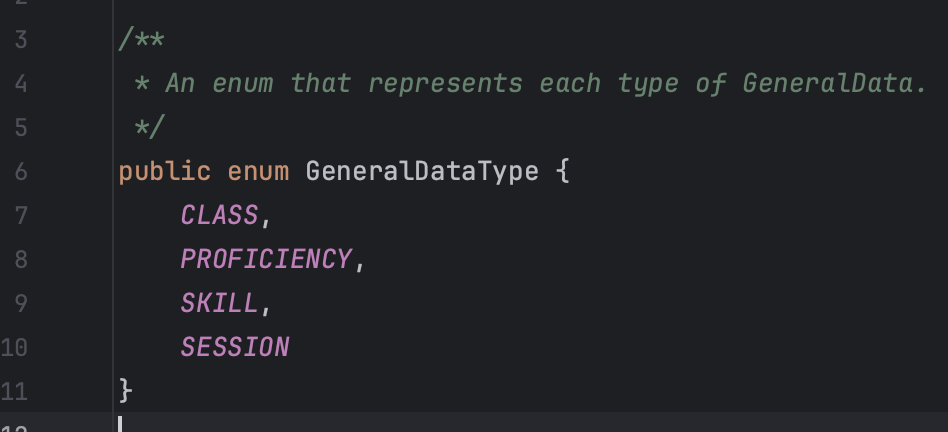
## Data Structure #21 (Appointment)



## Data Structure #22 (GeneralData)

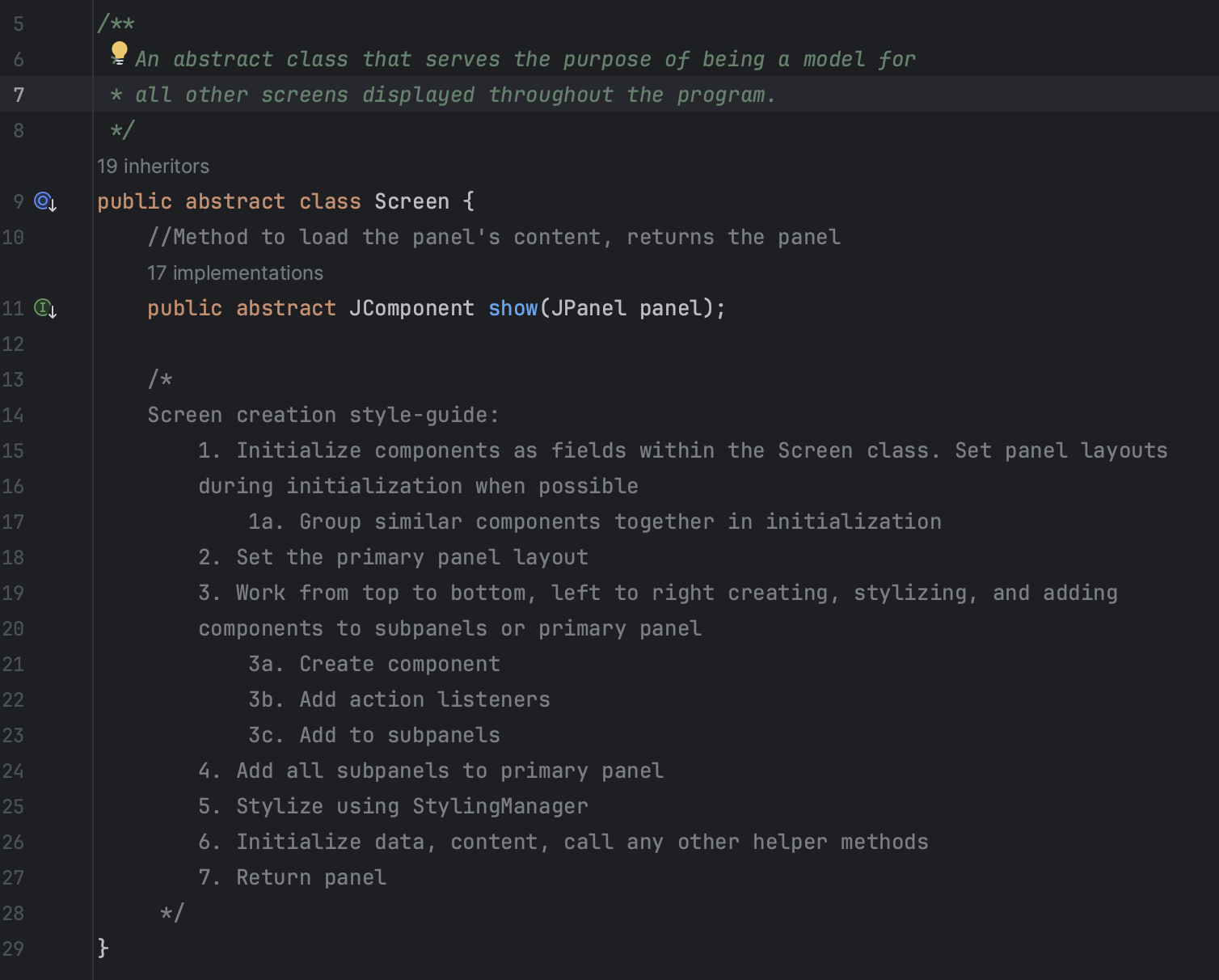


## Data Structure #23 (GeneralDataType)

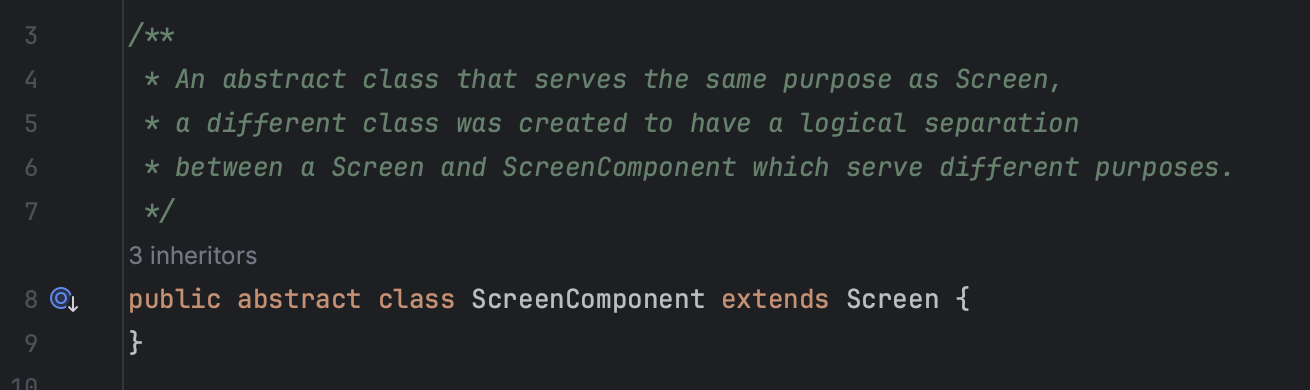


## 

## Data Structure #24 (Screen)

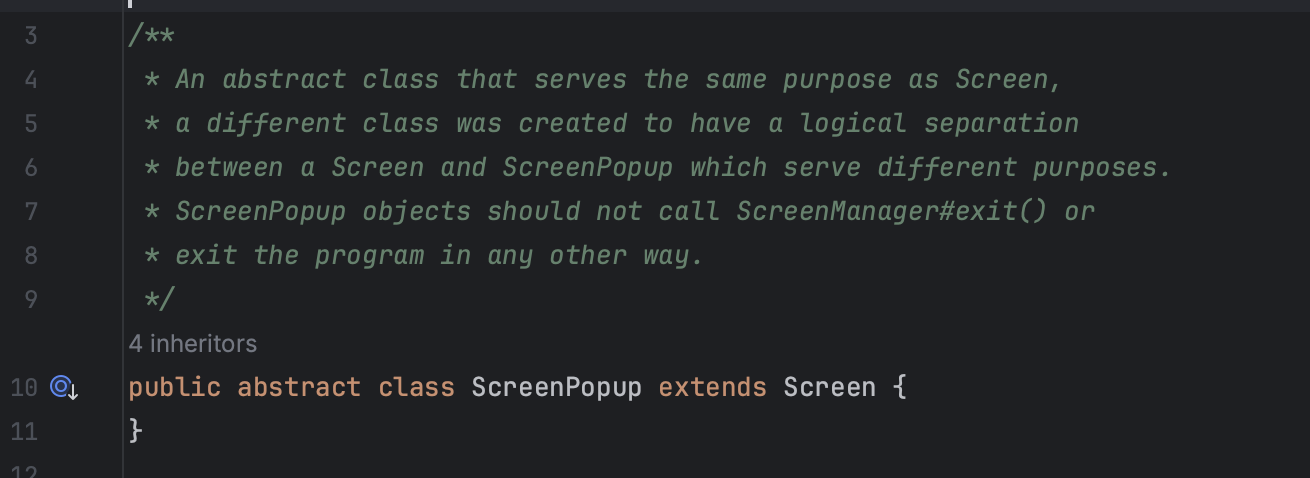


## Data Structure #25 (ScreenComponent)



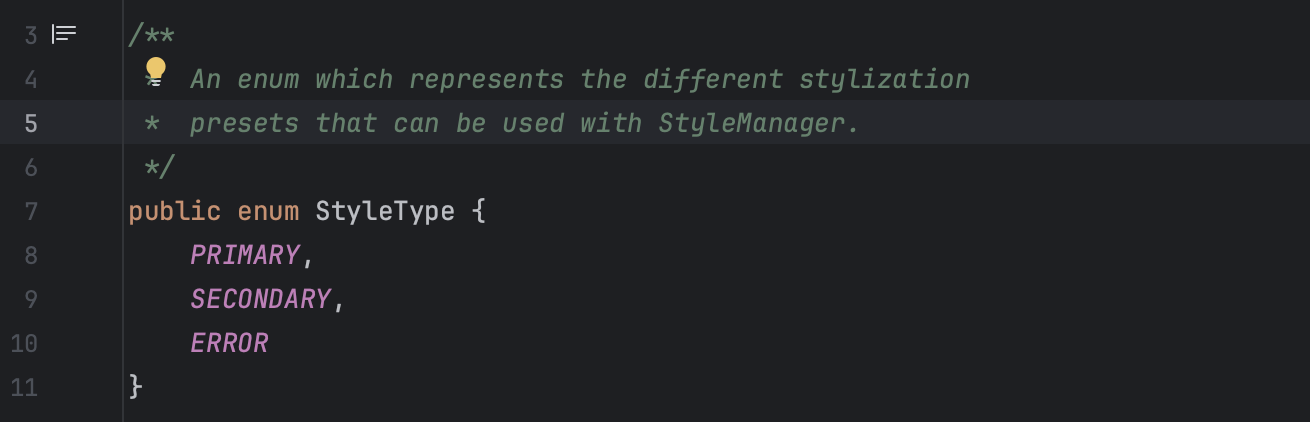
## 

## Data Structure #26 (ScreenPopup)

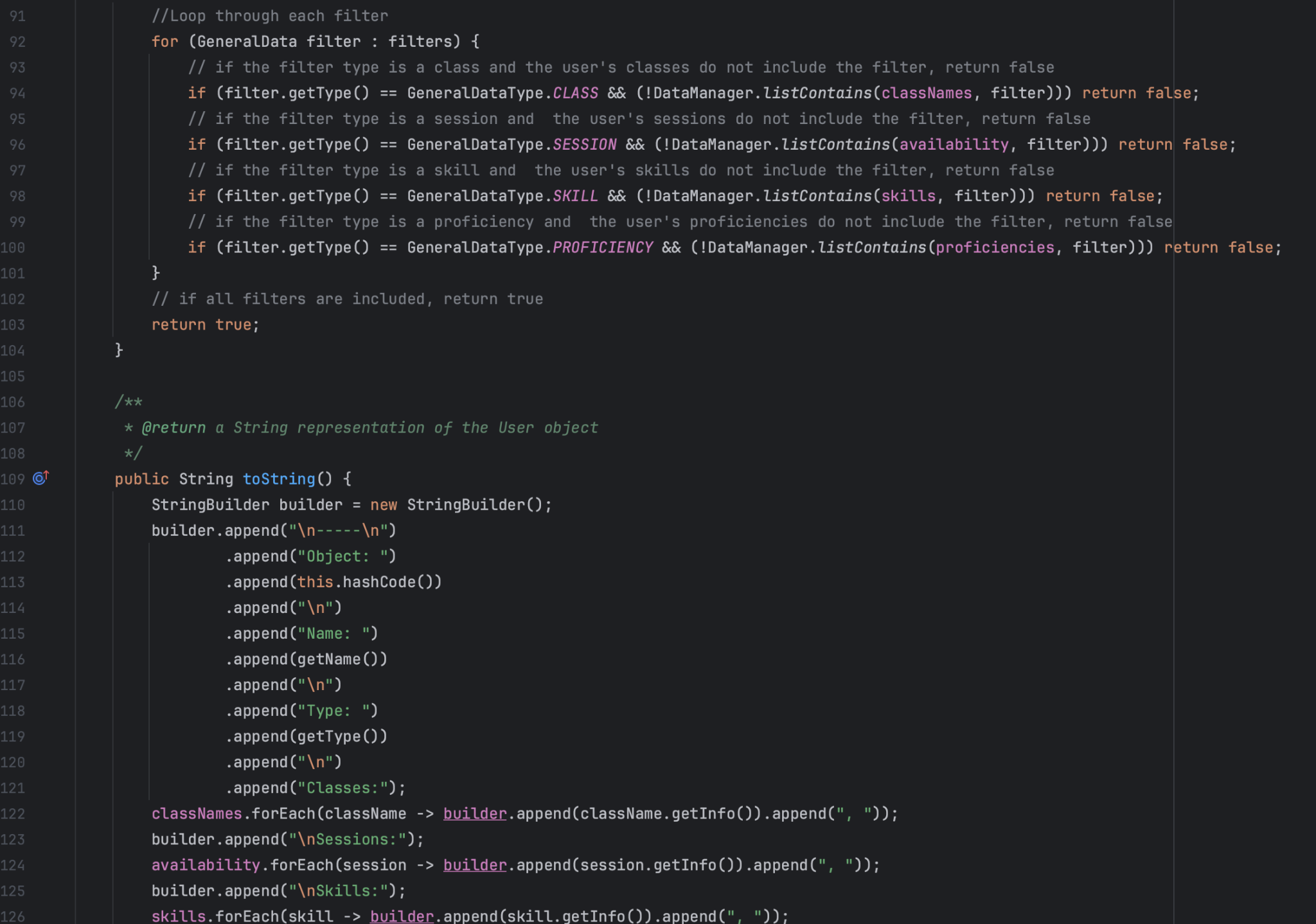
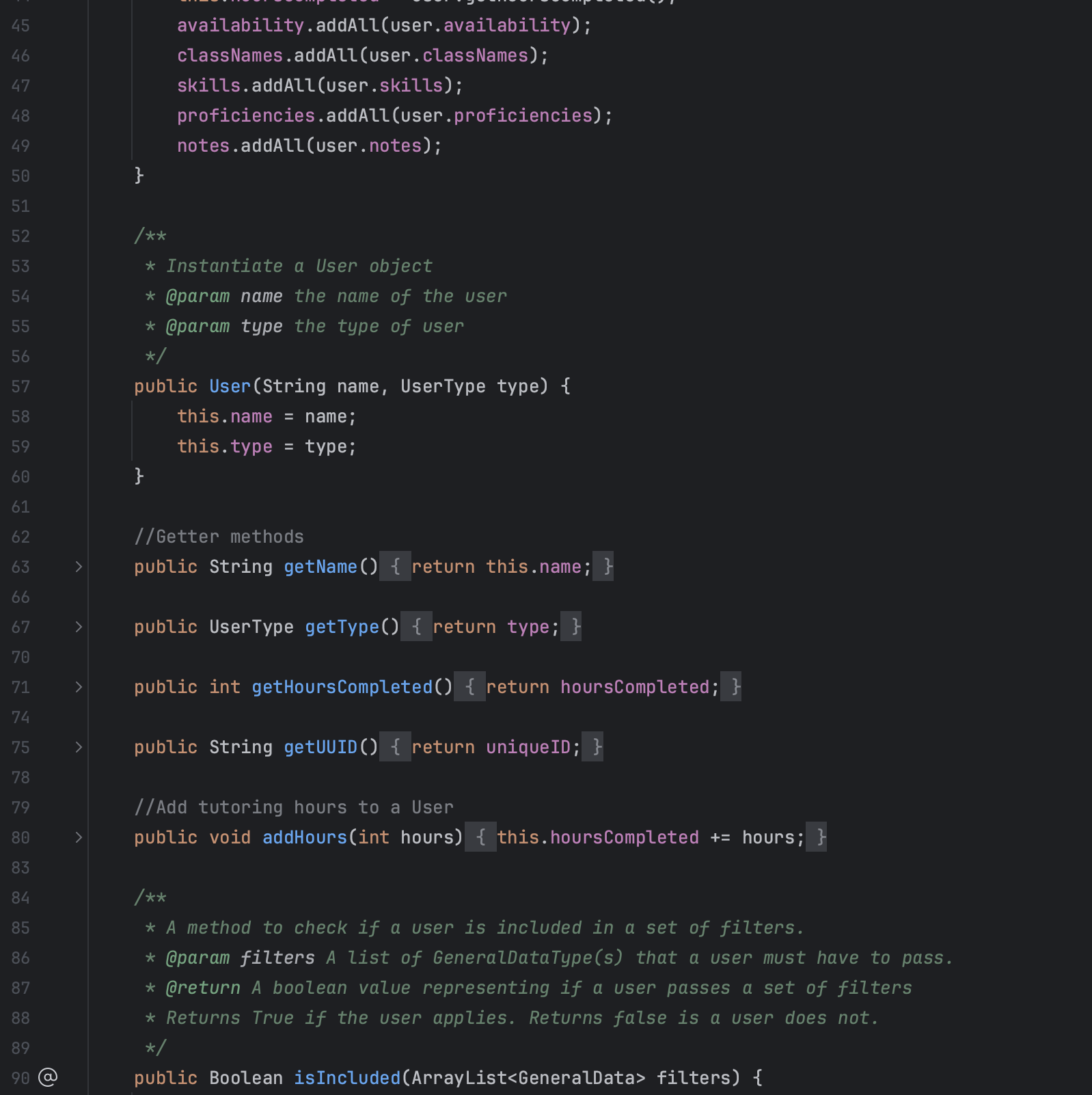


## 

## Data Structure #27 (StyleType)

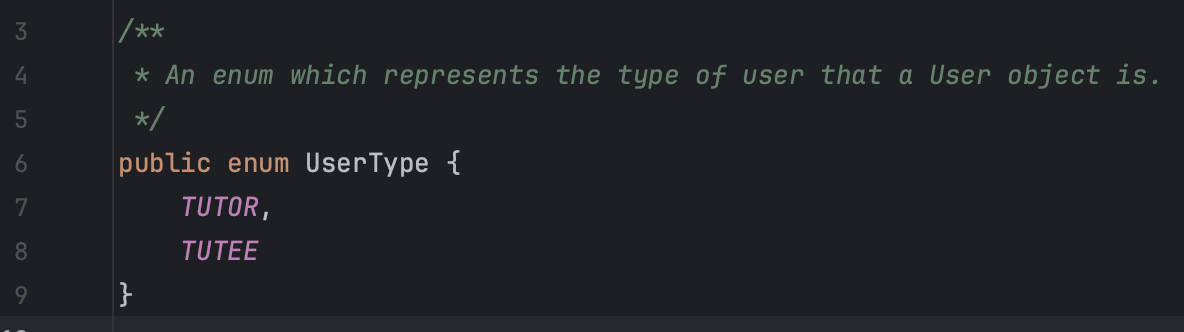


## Data Structure #28 (User)



## 

## Data Structure #29 (UserType)



# 

# Existing Libraries Used

## Swing (javax.swing)

Used for Screen GUI components (JFrame, JPanel, JLabel, JTextField, JPasswordField, JButton, JSpinner, JTabbedPane, JScrollPane, and JCheckBox) used throughout every Screen object in the program.

## 

## Awt (java.awt)

Used to handle various events (primarily button click) for JComponents.

## 

## Io (java.io)

Used to make objects Serializable and to create File objects and BufferReaders for serialization and deserialization.

## 

## Util (java.util)

Used to create ArrayList objects, HashMap objects, UUIDs, and a Scanner object to read from credentials file.

## String (java.lang)

Used number times to display text throughout the program.

# 

# Programming Techniques

## Serialization

Used to quickly and easily store custom objects in a datafile without a manual conversion method. Reliable and repeatable and easy to implement.

## Deserialization

Used to quickly and easily read custom serialized objects stored in external datafiles. Reliable and repeatable and easy to implement.

## Graphical User Interface (GUI)

Used (with the java.swing library) to display content to the user and receive user input in various forms (buttons, text boxes, dropdown menus, etc).

## Custom classes

Used to easily create and manage data programmatically. Custom classes allow the program to be much more organized and easily editable as they package related data together.

## Manager classes

Used to simply the higher-level processes of the program. Three managers were created and used: ScreenManager, DataManager, and StyleManager. **ScreenManager** allows the program to easily display different Screen and ScreenPopups while reusing the same content pane. This manager class also sets up the initialization and shutdown of the program’s screens. **DataManager** allows the program to easily interface with externally stored data and handles serialization and deserialization. This manager class allows other parts of the program to easily access data through a field on the class. **StyleManager** allows the program to easily be styled in a consistent manner with minimal boilerplate code. The entire program’s look and feel can be updated in one location without updates to every file needed.

## Method overloading

Method overloading was used within the StyleManager class to overload the #stylize() method for each type of component to be styled. Different components have different stylization requirements and sub-components to be stylized, however, by using method overloading. This method allows the program to be much more readable and reusable by using the same method name for similar functions with different arguments

# Justification

The choices made when developing this program were specifically made to ensure reliability, readability, and usability in the future. The program opts to be more verbose and extensive for the sake of a high level of organization. The file structure is organized into 5 main sections, each serving its own purpose. Custom classes are specifically important to the program’s organization to ensure that when writing new code or editing existing code, no unexpected data is passed to functions not designed for that type of data. Each manager class serves a distinct purpose to simplify major functions of the program. Besides manager classes and data classes, all other classes are Screen, ScreenPopup, or ScreenComponent objects, and hold the structural “design” of each screen as well as the appropriate button actions

**Modifications**

Minor modifications were made to the look and feel of the program to create a consistent overall look for the program. The most notable changes made were the removal of a purpose for a tutoring session and the removal of the non-authenticated user panel. The tutoring session purpose was removed as the client communicated that this information did not need to be tracked. The non-authenticated user panel was removed for the sake of time management to ensure the rest of the product was not rushed and the client thought it was unlikey that the feature would be used.

# Bibliography

“Creating Jbutton with Customized Look.” *Stack Overflow*, 1 Oct. 2013, stackoverflow.com/questions/14159536/creating-jbutton-with-customized-look.

“How to Center the Text in a Jlabel?” *Stack Overflow*, May 2012, stackoverflow.com/questions/6810581/how-to-center-the-text-in-a-jlabel.

“How to Convert a Char Array Back to a String?” *Stack Overflow*, July 2012, stackoverflow.com/questions/7655127/how-to-convert-a-char-array-back-to-a-string.

“How to Dispose a Jpanel - Jpanel1.Dispose() or Equivalent.” *Stack Overflow*, Nov. 2014, stackoverflow.com/questions/21365570/how-to-dispose-a-jpanel-jpanel1-dispose-or-equivalent.

“How to Duplicate an Array List in Java?” *Stack Overflow*, Feb. 2016, stackoverflow.com/questions/30074736/how-to-duplicate-an-array-list-in-java.

“How to Reload a JPanel?” *Stack Overflow*, 1 Apr. 2014, stackoverflow.com/questions/17608421/how-to-reload-a-jpanel.

“How to Set a Background Color to a Jframe Containing Multiple JPanels, JButtons and JLabels?” *Stack Overflow*, Jan. 2015, stackoverflow.com/questions/29460600/how-to-set-a-background-color-to-a-jframe-containing-multiple-jpanels-jbuttons.

“How to Set Background Color of a Button in Java Gui?” *Stack Overflow*, 1 Aug. 2011, stackoverflow.com/questions/4172940/how-to-set-background-color-of-a-button-in-java-gui.

“How to Set Fix Size of Jlabel?” *Stack Overflow*, Jan. 2013, stackoverflow.com/questions/22920046/how-to-set-fix-size-of-jlabel.

“How to Use Spinners.” *How to Use Spinners*, Oracle, docs.oracle.com/javase/tutorial/uiswing/components/spinner.html. Accessed 8 Apr. 2024.

“Java - Serialization.” *TutorialsPoint*, TutorialsPoint, www.tutorialspoint.com/java/java\_serialization.htm. Accessed 8 Apr. 2024.

“JPanel Padding in Java.” *Stack Overflow*, Jan. 2010, stackoverflow.com/questions/5328405/jpanel-padding-in-java.

*JPasswordField (Java Platform SE 8 )*, docs.oracle.com/javase/8/docs/api/javax/swing/JPasswordField.html. Accessed 8 Apr. 2024.

*JTextField*, Oracle, docs.oracle.com/javase/8/docs/api/javax/swing/JTextField.html. Accessed 8 Apr. 2024.

Paraschiv, Eugen. “Java - Create a File.” *Baeldung*, 5 Jan. 2024, www.baeldung.com/java-how-to-create-a-file.

“Scrollpane Adding to Grid Layout.” *Stack Overflow*, Mar. 2016, stackoverflow.com/questions/30292519/scrollpane-adding-to-grid-layout.

“Serialization and Deserialization in Java with Example.” *GeeksforGeeks*, GeeksforGeeks, 27 Oct. 2023, www.geeksforgeeks.org/serialization-in-java/.

“Setdefaultcloseoperation to Show a Jframe Instead.” *Stack Overflow*, June 2013, stackoverflow.com/questions/12210972/setdefaultcloseoperation-to-show-a-jframe-instead.

“Test.Java.” *Github Gist*, Anonymous, 16 Oct. 2016, gist.github.com/anonymous/ceab52cc7082c34baf1c.