**Notes Manager**

**Bachelor of Science in Information**

**Technology**

**Christian James G. Francisco**

**Rodny Mendoza**

**Ritzmer Alminaza**

**Reyster Caberte**

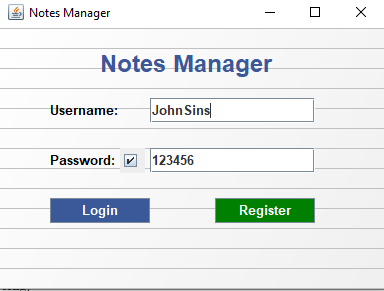
**June 12, 2024**

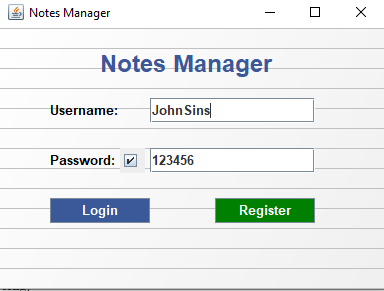
**Introduction**

**Demonstration**

To start with our application, run the JAR file of our application, the first scene of our application will be user authentication. If you’re not registered with our application, you can first register an account with your username and password.

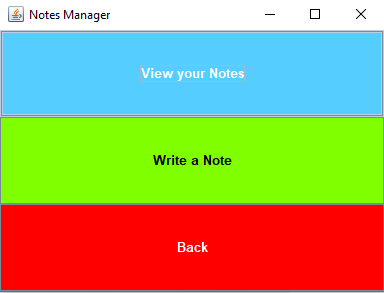
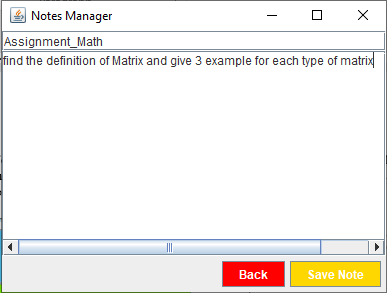
**Step 1.**  Create an account with your desired username and passsword(password must be atleast 6 characters long) by filling up the 2 text box and then press the “register” button.



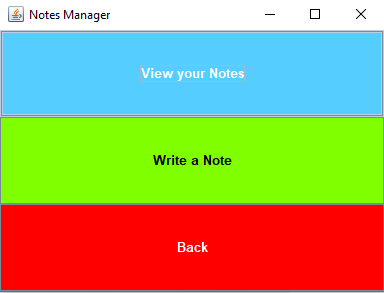
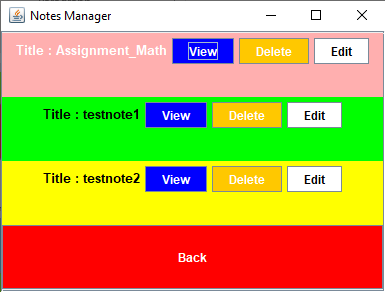
**Step 2.** As you have finished creating your account press the “Login” button to login into the application using the credentials you’ve provided.



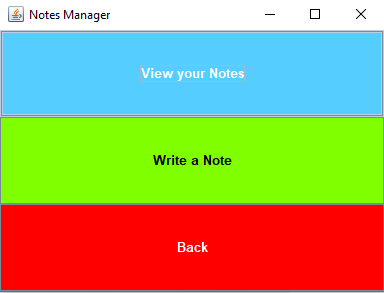
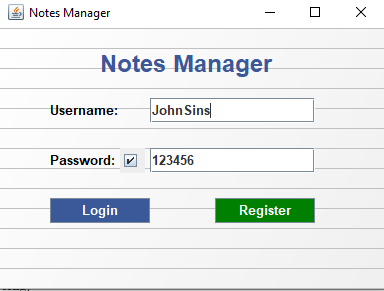
**Step 3.** To write notes press the “Write a Note” button on the main menu and it will take you to the Note writer window. In the note writer window, there will be two text box that you need to fill, the title box and the content box(Note: you can’t save a note if you don’t fill the title box). After filling up the text boxes press the “Save Notes” button to save it.



**Step 4.** In the view notes scene, your saved notes will be displayed alongside with 4 buttons, “View”, “Delete”, “Edit”, and “Back”. The functions of these buttons are self explanatory. If you done managing your notes, pressing the “Back” button will take you back into the main menu scene.



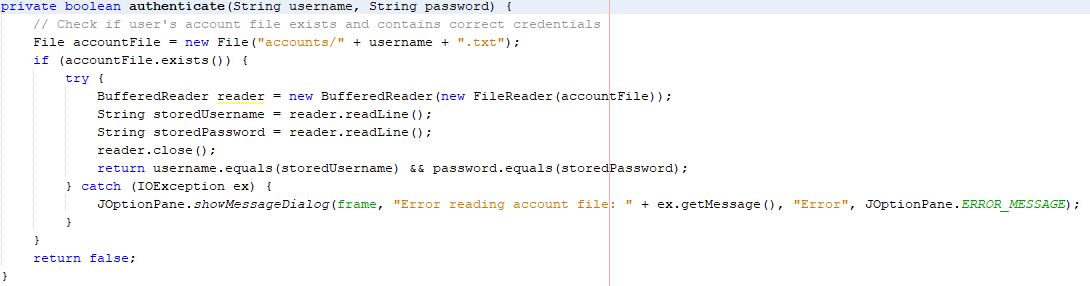
**Step 5.** To log out of your account press the “Back” button on the main menu scene and it will take you back into the user authentication scene.



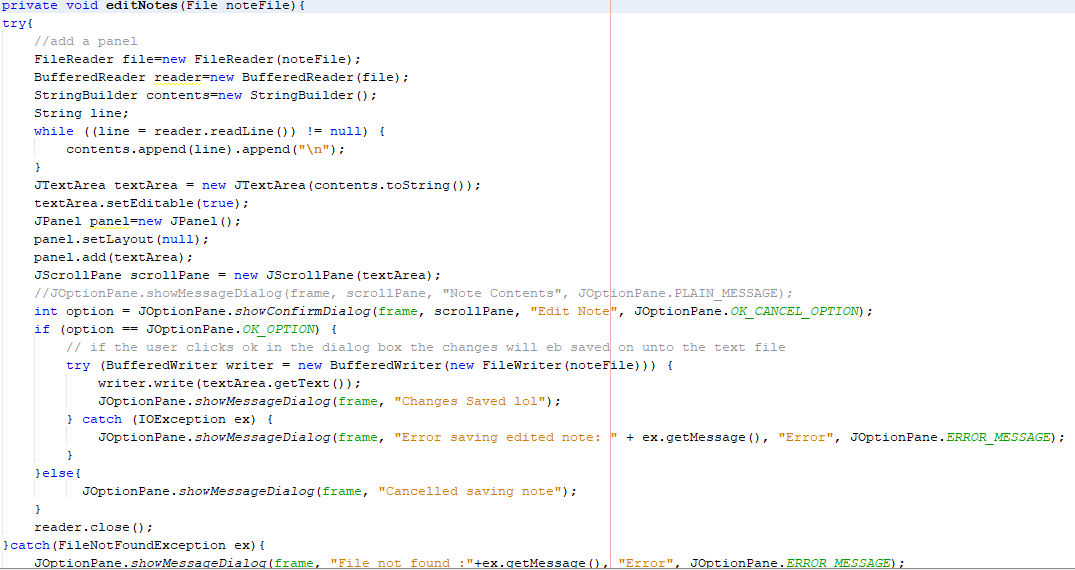
**List of Exceptions**

In our application we only used two exceptions and they are built-in type of Exceptions, The exceptions that we used are the FileNotFoundException, and the IOException. The IOException is used to handle potential errors that may occur during the reading of the account file, displaying an error message if an issue arises. In our code, the FileNotFoundException is used to handle the case where the specified note file is not found, displaying an error message if the file is not found.

1. **IOException**



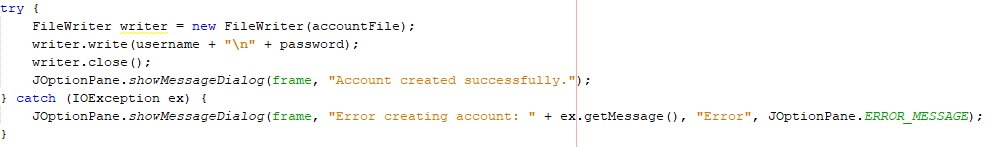
1. **FileNotFoundException**

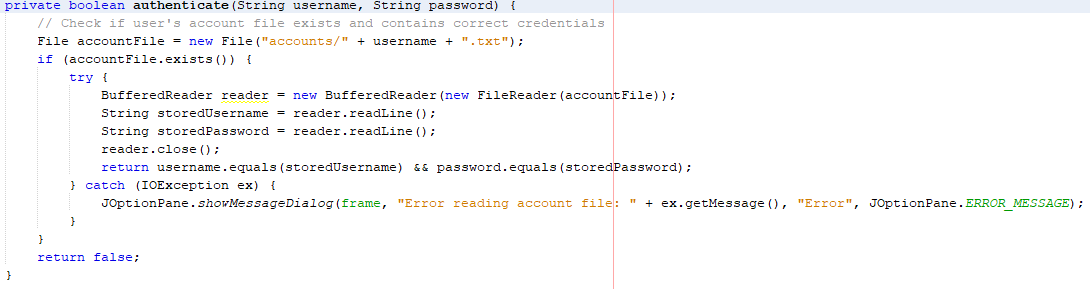


**File Input and Output Procedure**

In our appication, file input and output procedures are utilized for managing user accounts and notes. FileWriter is employed to create new files and store user account details during registration, while BufferedReader is used to retrieve stored credentials during authentication. Similarly, for notes, FileWriter facilitates the creation of new note files, and BufferedReader is employed to read and display existing note contents, with BufferedWriter used to save edits back to the note files.

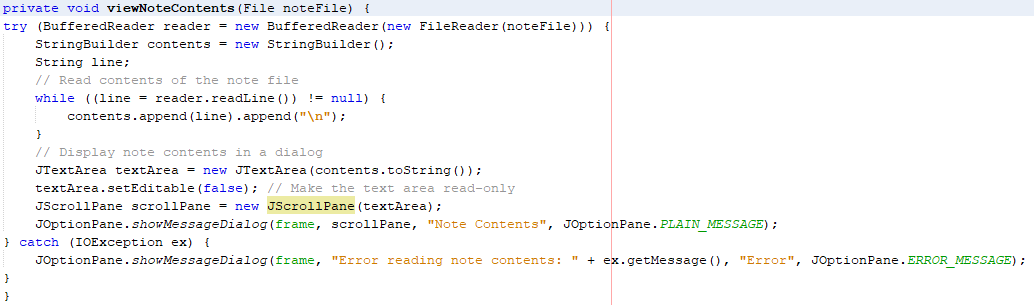
1. In the register method, FileWriter is utilized to create a new user account file and store the provided username and password. The "\n" character is used to ensure that the username and password are stored on separate lines within the file. If an IOException occurs during this process, indicating a failure in file writing, an error message is displayed, alerting the user of the issue. This ensures that any potential errors in file input and output operations are appropriately handled during user registration.



1. In the authenticate method, BufferedReader is employed to read the stored username and password from the user's account file. Each credential is read from a separate line within the file. If an IOException occurs during this process, indicating a problem with file reading, an error message is displayed to inform the user about the issue. This ensures that any potential errors in file input operations are handled gracefully during user authentication.
2. In the saveNoteAndReturn method, FileWriter is used to create a new note file and save the provided note title and content. The note title is appended with ".txt" to create the filename, ensuring it is stored as a text file. If an IOException occurs during this process, indicating a failure in file writing, an error message is displayed to notify the user about the problem. This ensures that any potential errors in file output operations are appropriately managed during note creation.



1. In the viewNoteContents method, BufferedReader is utilized to read the contents of the specified note file. Each line of the note content is read sequentially and appended to a StringBuilder to display in a text area.If an IOException occurs during this process, indicating a problem with file reading, an error message is displayed to alert the user about the issue. This ensures that any potential errors in file input operations are handled effectively during note viewing.



1. In the editNotes method, FileReader and BufferedReader are used to read the existing contents of the specified note file, ensuring that the text is displayed for editing. BufferedWriter is employed to write the updated note content back to the file, ensuring that changes made by the user are saved. If any IOException occurs during these operations, indicating issues with file reading or writing, appropriate error messages are displayed to inform the user about the problem, ensuring robust handling of file input and output operations during note editing.

Top of Form

Bottom of Form

