**User/Administrator Manual for**

**Team Pseudo’s**

**Password Manager**

Dennis Tran, Kyle Parker, Brandon Pearson, Addison Burkett

Capstone Project Fall 2022

The University of West Florida

29 November 2022

CIS4595

Dr. Owsnicki-Klewe

**Password Manager**

**Product Functions and Capabilities**

The purpose of this product is to provide individuals, teams, and organizations with an easy-to-use, secure credential sharing solution. There are two main focuses for the Password Manager.

* First, it will eliminate the hassle and security lapses of having to memorize/maintain passwords with a centralized vault for user credential storage and access.
* Secondly, it will provide credential sharing ability with real-time alerts on credential access.

With the Password Manager, users will be able to set up hierarchical systems in which specific users can be granted access to resources and credentials. When administrators change credential information, the software will automatically update passwords of any resources that are shared to other members. Additionally, administrators get privileged access to audit logs which display a record of credential access from all members.

**Product Features and Function**

The Password Manager includes many features that make storing, accessing, and sharing credentials easy. These features are:

1. **Account Creation (Register and Login)**

The application allows users to register an account. Since many credentials are saved under the user account, it is important that the user has a strong password linked to their account. The Password Manager application ensures that the user’s password is strong by enforcing it to have at least one capital letter, numbers, and a special symbol. Saved credentials are associated with accounts and therefore users will be able to access their saved credentials when they log in.

Basic Course of Events:

1. The user selects the “create an account” button
2. The user inputs their email address, username, and password
3. The application communicates with the database of profiles to confirm that the input email and username are not currently in use
4. The application calls for a new profile entry to be added to the database
5. A message confirming that the account was created successfully
6. The user is brought to the main screen of the application

1. **Store credentials in a vault**

Users can save their credentials in a vault. The process is very simple, as the user only needs to enter the website, user, and password that they want to save.

Basic course of events:

1. The User clicks on the "Store Credential" button.

2. The system responds by asking the user for the credentials to store and the domain in which the credential is stored for.

3. The User enters the credentials to store and clicks confirm.

4. The system validates the User's credentials for security and adds it to the database.

1. **Access saved credentials, through individual vault and teams**

Users can access their saved credentials by pressing the credentials button. By doing that, the application will display the titles of all credentials that the user has saved in their vault. Once a credential has been selected, the application will show the credential, which includes the strength of the password, creation time, modified time, notes, and more. Users also have the ability to access credentials through teams.

Basic course of events:

1. The User presses on the tab to show stored credentials.

2. The system displays the user's stored credentials in a hidden state.

3. The User chooses the credential to see and clicks on the reveal button.

4. The system prompts the user for their software password to unlock the credential.

5. The User enters the password.

6. The system displays the credential.

1. **Create teams to share credentials**

By clicking on the “new team” button, users are able to create teams. When teams are established, other users can be invited to the team and the team owner can administer the credentials that they wish to share. When other users access your credentials through a team, you will also be notified.

Basic course of events:

1. The User clicks on the "create a group" button.

2. The system responds by allowing the user to input information for their intended group.

3. The User enters the group name and other relevant information.

4. The system creates the group and adds it to the table of groups in the database.

1. **Establish roles within teams**

Team owners can create roles within their team. Roles serve the purpose of giving specific members of the team privilege/access to certain credentials. For example, the team owner can create a role called “admin” and indicate that the admin role has access to all shared credentials. Likewise, roles can limit the amount of credentials other members can view.

Basic course of events:

1. The user clicks the "create role" button

2. The system asks the user to name the role and define the credentials that the role has access to.

3. The user enters the information

4. The system saves it to the database.

1. **Audit Log that displays credential access**

The audit logs serve the purpose of allowing the user to know when any of their credential(s) have been accessed. The audit log will display the username of the user who has accessed their credential, the username of the credential being accessed, and the time at which it occurred. This feature gives credential owners the ability to keep track of their credentials, especially when their credential is being shared through a team.

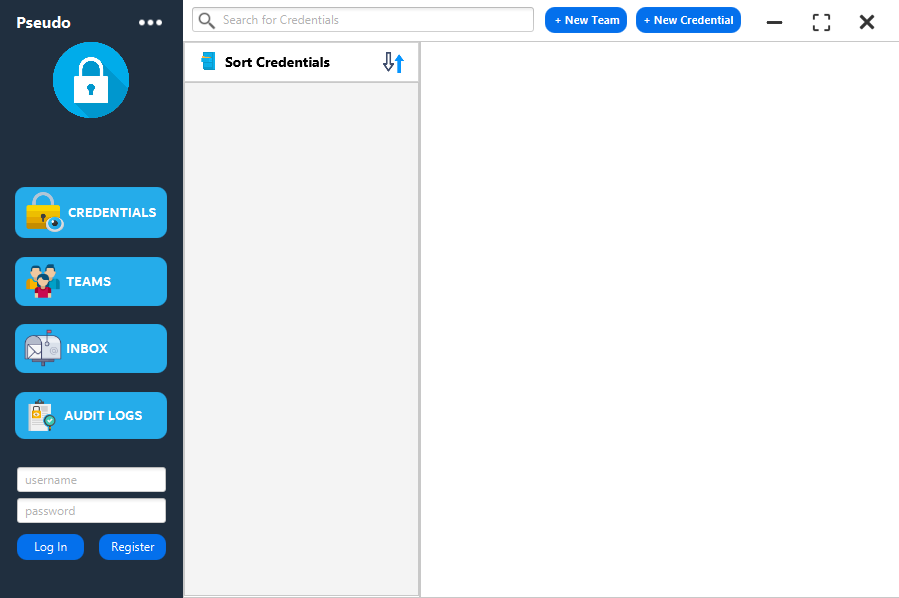
Basic course of events

1. The User or other Users get access to a credential

2. The system responds by updating the audit log for that user/group

**Product Walkthrough**

The first thing that you will want to do is launch the application. Once the project is running, the user is presented with the graphical user interface. All views will be empty since the user is not logged in yet. The application will look like this:

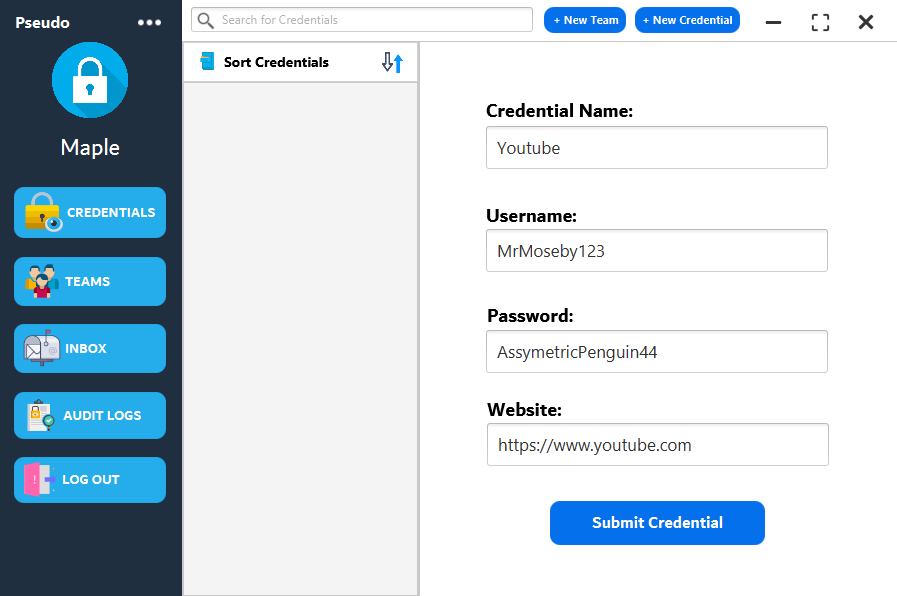


In order to have access to the application features, the user must first register an account. Users can do this by entering a username of their choice and a password that has at least one capital letter, numbers, and a symbol, then press the “register” button. If the username is already taken, the application will indicate that to the user. Successful registration is indicated when the application says “registration complete”.

When an account has been registered, the user can simply log in with their newly created account by entering the appropriate credentials and pressing log in.

Users will know that they have successfully logged in when the credential fields at the bottom of the interface have disappeared. Furthermore, the username of the user is presented at the top-left of the application under the application logo. When logged in, the user can click on the side button and top buttons to interact with the application.

At this point, users can choose to add credentials to save, create a team, and more.

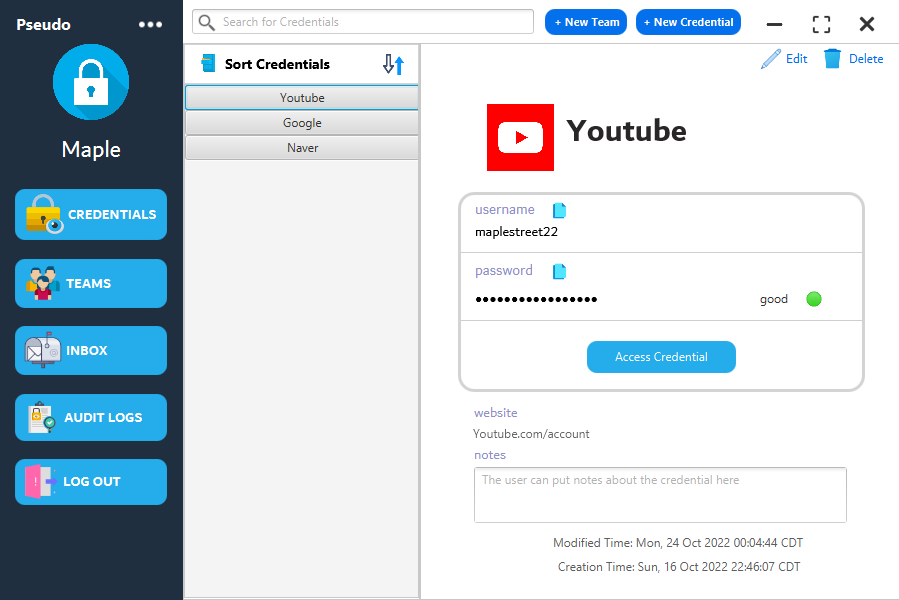


To save a credential, users can press the “+ New Credentials” button at the top right of the application. Once clicked, it will pop up a new view on the right side of the application. In this view, the user has to enter the name of the credential. In the example shown in the image, the credential name is “Youtube”.

Afterward, the user can now enter the username and password of the credential that they want to save.

The website field is reserved for the website URL. Since the credential in the example is youtube, I have entered the URL of Youtube for that field.

Clicking on the “Submit Credential” button will then save the credential and link it to the current user. To view the credential, the user should click on the “Credentials” button and the saved credential should appear.

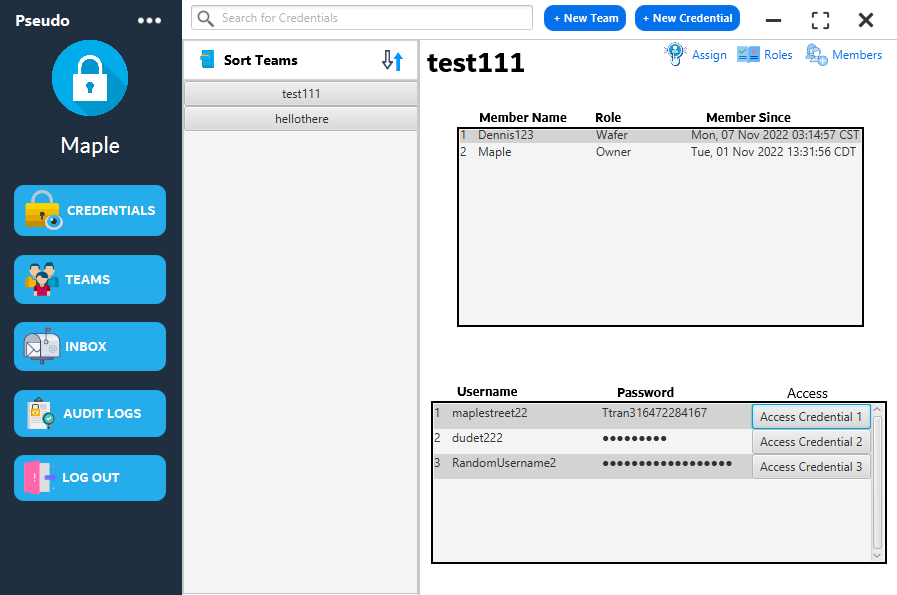


Once a saved credential is selected, the view on the right side of the application is updated to show the selected credential. In the example, I have selected the Youtube credential that I have previously created. The credential view shows the credential title with the website favicon next to it. Furthermore, the username and password is also displayed. Though, the password will be hidden until the user clicks on the “Access Credential” button.

The application will also calculate the strength of the saved password. In the example, my password that I saved has “good” strength.

At the bottom half of the credential view, it will display the website URL associated with the credential, any notes that the user chooses to enter, and the creation/modified time of the credential

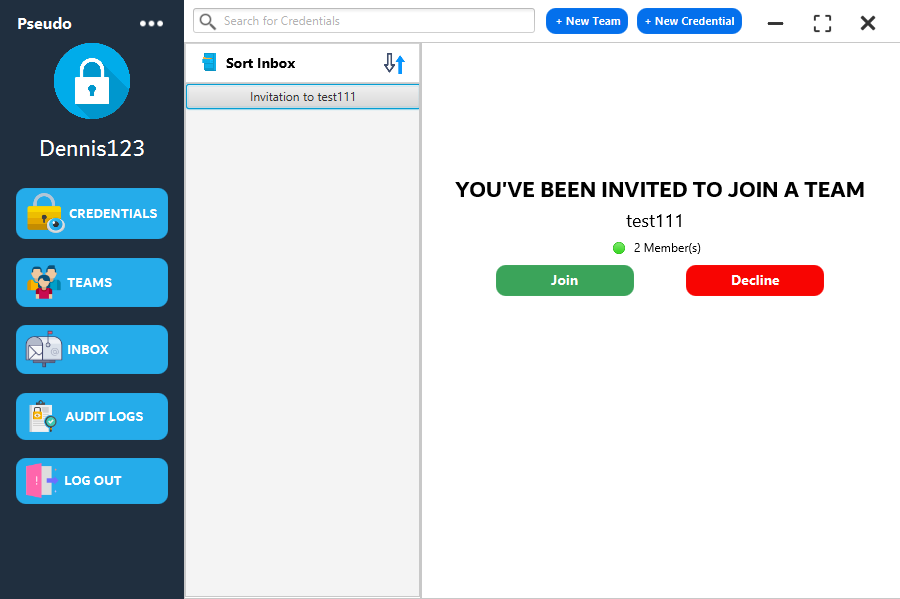
If the user wishes to modify the credential, the user can simply click on the “Edit” button at the top of the credential view to change any element of the credential. Modifying the credential will update it in the database, the user interface, and display the appropriate modification time. If the user no longer wants the credential, they can delete it with the “Delete” button at the top of the credential view.



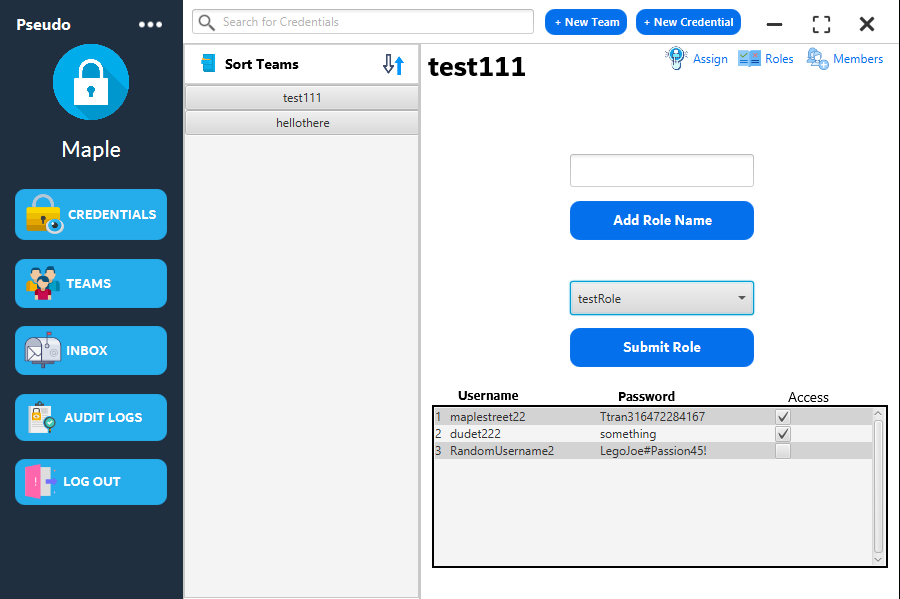
Creating a team is very simple. All the user needs to do is press on the “+ New Team” button at the top of the application. Upon doing so, the user is presented with a prompt that asks for the team name. Once the team name is entered, the team is created. In the example shown above, the team called “test111” was created and there is already a member in the team that was invited. This shows the role of the user and the date of when they were added to the team. At the bottom of the team view, the user can see the credentials that they are allowed to access. This would differentiate between members of the team depending on their assigned role.

Since the example shows the team owner, they have access to all of the saved credentials. Clicking on the “Access Credential” button next to the credential will display the password and notify the credential owner that their credentials have been accessed.

To invite users to the team, the team owner can press the “+ Member” button and enter the username of a user that they would like to add.



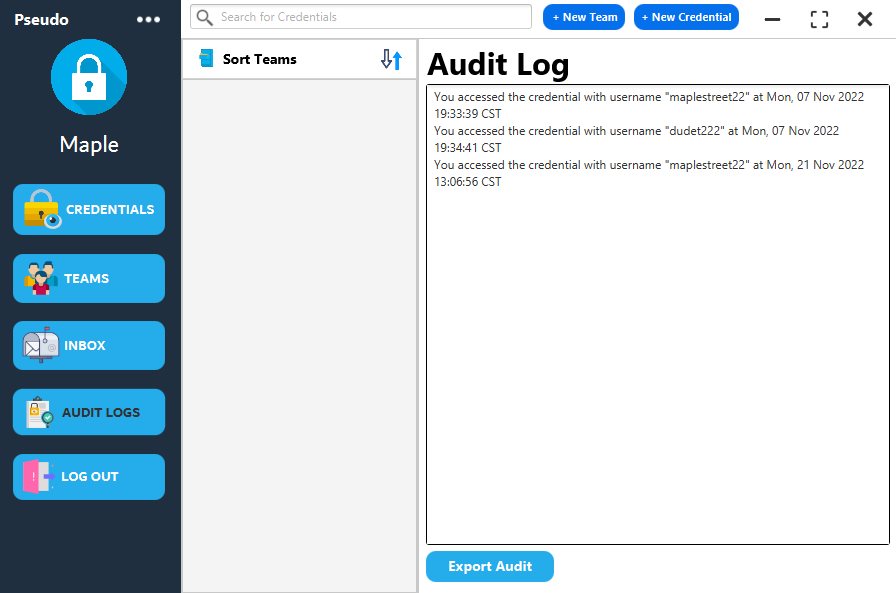
In the image shown above, I have logged into a different user that has been invited to the “test111” team. By clicking on the “Inbox” button on the left side of the application, I am able to see all of my invitations. Clicking on the invitation shows that I have been invited to the team and that I can either Accept or Decline the invitation. The invitation will display the team name and the number of users within the team.



On the team owner side, the owner can now create a role. In the example above, the owner has created a role called “testRole” and checked two credentials. That means that the testRole role has access to credentials one and two. Clicking on the “Submit Role” button will successfully create the role.

Once a role has been created, it is the team owner’s responsibility to assign the role to members of the team. By clicking on the “Assign” button at the top of the team view, it will pop up a view that allows the owner to choose the member and role that they want to assign.

Once the role has been assigned to a member of the team, that specific member will have access to the credentials that are within the privilege of their role.



Users of the application are also equipped with an audit log to keep track of any time that their credentials are accessed. When users access their own credentials, it will display it in the audit log with the corresponding credential username and time that it was accessed. Likewise, any member that accesses the credential from within the team will also update the audit log of the credential owner.

Users also have the ability to export the audit log to a file on their desktop. For example, the user can choose to export the audit log as a spreadsheet or something as simple as a .txt file.

The final feature of the walkthrough is the “log out” button. Pressing the log out button the left side of the application will clear all application views, log the user out, and allow the user to relog or log in as a different user.

**Installation of the Software**

In order to install this application, you will first need to go to the github repository at this link (which should already be shared to you): <https://github.com/DennisTran00/Capstone_Pseudo>

Clone the repository into your IDE. The recommended IDE to use is Eclipse, as the software was developed with Eclipse and this entire installation tutorial will be based off of Eclipse.

Before Running the application there are several set-up steps that must be followed, starting with ensuring that all project dependencies are present and working. First, expand the project in the workspace to verify that all project files are present. If all project files are present, continue the installation tutorial. If not, then there may be a problem with cloning the repository.

**Database**

This project currently utilizes a local MySQL database for demonstration purposes. Therefore, it is necessary to correctly set up the database before the software can be launched. The database connection is established by the following code within DBConnection.java:

private DBConnection() {

uri = "jdbc:mysql://localhost/passwordmanager?useSSL=false&allowPublicKeyRetrieval=true";

try {

dbCon = DriverManager.getConnection(uri, "root", "password");

} catch (SQLException e) {

e.printStackTrace();

}

}

This means that the MySQL account must have a username of root and a password of password to get the database working correctly. If the database is being established on Windows OS, ensure that the MySQL service is running by pressing Win + R on your keyboard to open the Run window. Then type services.msc and hit enter. Scroll down until you see MySQL and run the service if it not doing so already. Once MySQL is running on your machine, launch the MySQL command line. The following commands will set up the database required for the software to work:

CREATE DATABASE passwordmanager;

use passwordmanager;

CREATE TABLE user\_info (

username VARCHAR(255),

password VARCHAR(255),

ID VARCHAR(255)

);

CREATE TABLE credentials (

UserID VARCHAR(255),

Username VARCHAR(255),

Password VARCHAR(255),

URL VARCHAR(255),

Notes VARCHAR(255),

Created VARCHAR(255),

CreationDate VARCHAR(255),

ModifiedDate VARCHAR(255),

Title VARCHAR(255)

);

CREATE TABLE teams (

GroupID VARCHAR(255),

GroupName VARCHAR(255),

OwnerID VARCHAR(255)

);

CREATE TABLE members (

GroupID VARCHAR(255),

Username VARCHAR(255),

UserID VARCHAR(255),

Role VARCHAR(255),

GroupName VARCHAR(255),

Joined VARCHAR(255)

);

CREATE TABLE roles (

GroupID VARCHAR(255),

RoleID VARCHAR(255),

RoleName VARCHAR(255),

Username VARCHAR(255),

Password VARCHAR(255)

);

CREATE TABLE invites (

GroupID VARCHAR(255),

UserID VARCHAR(255),

Username VARCHAR(255)

);

CREATE TABLE audit (

GroupID varchar(255),

Username VARCHAR(255),

CredentialUN varchar(255).

AccessTime VARCHAR(255)

);

Doing so, will finish the database set-up.

**Encryption**

This project utilizes AES symmetric encryption to secure the credentials that are stored in the database. Therefore, it is imperative that the setup for the encryption is done correctly. Not setting up the encryption for the project will result in several project runtime errors. Fortunately, this process is fairly simple. The Password Manager utilizes a Keystore that manages the keys used for encryption. To set up the Keystore follow these steps:

1. Assuming you are on Windows OS, open the Command Prompt by pressing Win + R and entering cmd.
2. Go into the database directory of the project. For example, doing cd C:\Users\YOUR\_USER\git\Pseudo\PassMngr\src\main\java\database will change directories to the database directory. Not being in the correct directory will most likely cause errors to occur, thus this step is very important.
3. Once in the database directory, paste the following into the command prompt: keytool -genseckey -keystore aes-keystore.jck -storetype jceks -storepass mystorepass -keyalg AES -keysize 256 -alias jceksaes -keypass mykeypass. If the cmd prompt asks for a password, enter password. It most likely will not. Doing this step will create the Keystore file inside of the database directory that will be used. Ensure that the file is there by refreshing the project workspace in the Eclipse IDE.
4. Done! The Keystore should work now.

**Testing Framework**

JUnit was used as the testing framework for this project. It is highly recommended that you use Eclipse IDE to run the project. The JUnit tests can be found under the test directory within src. To ensure that the tests work and that there are no compiling errors, make sure that you have the JUnit Library/Dependencies for the project. If you do not, follow these steps (in Eclipse IDE):

1. Right click the project file PassMngr.
2. Find and hover over Build Path, then click Configure Build Path....
3. In the window that pops up, find and click on the Libraries tab.
4. Click on ModulePath and click Add Library....
5. Select JUnit to add the library. The version should be JUnit 5. Click Finish.
6. JUnit should be imported as a library now. You can make sure of this by looking at the project files and see that JUnit is now in the project workspace.

**Running the Application**

Once you have finished setting up the database, keystore, and testing framework, you should now be able to run the application by navigating to the Main package. Within the Main package, you will find a .java file called PasswordManager.java. Simply right click on that file and hover over Run As then click Java Application. At this point, the application should be running without any error. If there are errors refer to the troubleshooting section below.

**Troubleshooting Problems**

Since this project uses JavaFX for the graphical user interface, some problems related to JavaFX dependencies may occur when trying to launch the project for the first time. Make sure to follow these fixes to common problems with our project. All of these fixes are through the assumption that the project is being imported into Eclipse IDE.

**Am I running the application with the correct JavaSE version?**

Make sure that the correct JavaSE version is being utilized. To do this, expand the project directory in the workspace. Once the project directory has been expanded, you should see JRE System Library. Right click on it and proceed to Properties. A new window will open with the title Properties for JRE Systerm Library. Here you can change the version by checking Execution Environment and change it to JavaSE-11 (jre) if it is not so already.

**JavaFX runtime components are missing**

If you run the project it will compile but you will get this error:

Error: JavaFX runtime components are missing, and are required to run this application

This error is shown since the Java 17 launcher checks if the main class extends javafx.application.Application. If that is the case, it is required to have the javafx.graphics module on the module-path.

A possible fix to many problems regarding JavaFX would be to use the JavaFX SDK instead of a build tool. Use this [link](https://gluonhq.com/products/javafx/) to download the correct JavaFX SDK. Our project uses javafx-sdk-17.0.0.1. Download the zip file and save it to a location on your drive.

Now, locate PasswordManager.java and right click it. Hover over Run As and select Run Configurations.... A new window titled Run Configurations should pop up. Navigate to the tab that says Arguments and under VM arguments add these VM arguments:

--module-path /path/to/javafx-sdk-17/lib --add-modules javafx.controls,javafx.fxml

**I am able to run the project without errors, but the GUI is empty**

If you are able to launch the project, but the GUI is just showing a white box, it means that is not correctly reading the .fxml file. Locate PasswordManagerGUI.fxml and make sure there is content in that file. If there is content in that file, make sure to refresh the workspace and relaunch the project. In most cases this will fix the problem of an empty GUI.

**The program launches, but I can’t register or log in**

If the project launches correctly and you are able to see the GUI, but unable to interact with the GUI (initial register and log in), then that means that the database is not set up correctly. This project heavily relies on a working database connection. Ensure to follow the instructions under the Database section to correctly set up the database.

**Launching the application fails because of JUnit**

For completion purposes, we have left the JUnit tests in the main branch of the github repository. If you cannot launch the application because there are errors in regards with the test, you can just delete them. Otherwise, make sure that you have correctly imported the JUnit libraries under the Testing Framework section of this manual.

For any assistance or further questions you can contact the project lead at this email:  
[dst13@students.uwf.edu](mailto:dst13@students.uwf.edu)

Or contact through Discord: Dennis1#7414