****

**North South University**

Department of Electrical and Computer Engineering

**Lab Project Report**

Semester : NSU Spring 2023

Course Code : CSE 215L

Section : 16

Group Name : D

Faculty : Dr. Shamim Al Mamun (SAM3)

Lab Instructor : A. S. M. Sabiqul Hassan

Project Topic : Metro Rail Ticket Booking System

GitHub Repo Link : https://github.com/AtikAkbar/Metro-Rail-Ticket-Booking-System.git

Submission Date : 16/06/2023

|  |  |
| --- | --- |
| Student Information | GitHub Account Links |
| 2211757642  Mohd Atik Akbar Rahat  Mohd.rahat@northsouth.com | https://github.com/AtikAkbar |
| 2222472042  Md Rayhan Sharif Rafsun | https://github.com/rsrrafsun |
| 2222742642  Md Radoanul Islam | https://github.com/Radoanul-Islam |

Package: metro.home

Class: Home

Description: This class creates the home screen for the Metro Ticketing System.

Methods:

* public Home(): This is the constructor for the Home class. It initializes the components of the home screen and sets the window properties.
* public void actionPerformed(ActionEvent ae): This method handles the button clicks on the home screen. It redirects the user to the appropriate screen depending on the button clicked.

Usage:

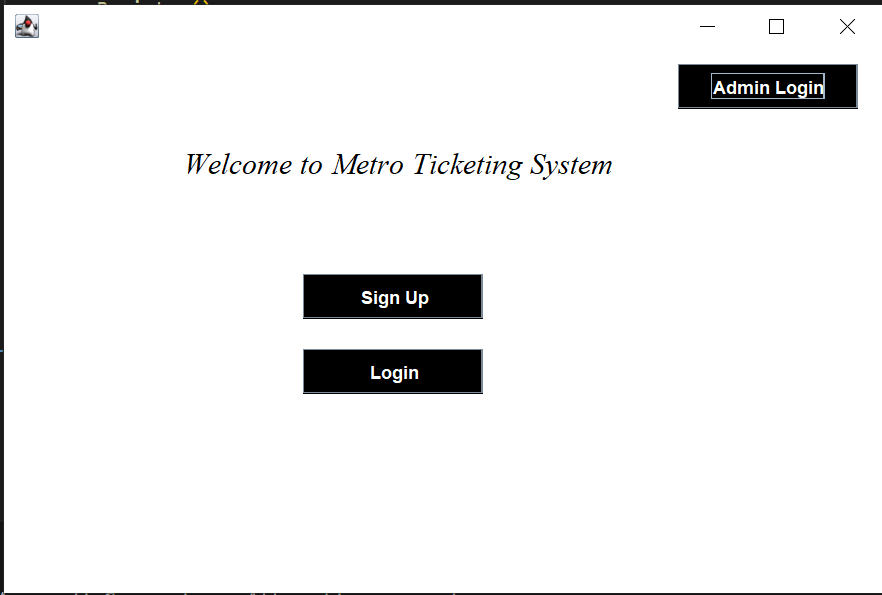
To use the Home class, you first need to create an instance of the class. You can then call the actionPerformed() method to handle the button clicks on the home screen.

Example:

Home home = new Home(); home.actionPerformed(ae);

Output:

The output of the Home class is the home screen of the Metro Ticketing System. The screen contains three buttons: Sign Up, Login, and Admin Login. The user can click on any of these buttons to be redirected to the appropriate screen.



Package: metro.home

Class: Register

Description: This class creates the registration screen for the Metro Ticketing System.

Methods:

* public Register(): This is the constructor for the Register class. It initializes the components of the registration screen and sets the window properties.
* public void writeToFile(String username, String mail, String password, String securityQuestion, String securityAnswer): This method writes the user's registration information to a file.
* public long generateUniqueCardNumber(): This method generates a unique card number for the user.
* @Override public void actionPerformed(ActionEvent ae): This method handles the button clicks on the registration screen. It redirects the user to the appropriate screen depending on the button clicked.

Usage:

To use the Register class, you first need to create an instance of the class. You can then call the actionPerformed() method to handle the button clicks on the registration screen.

Example:

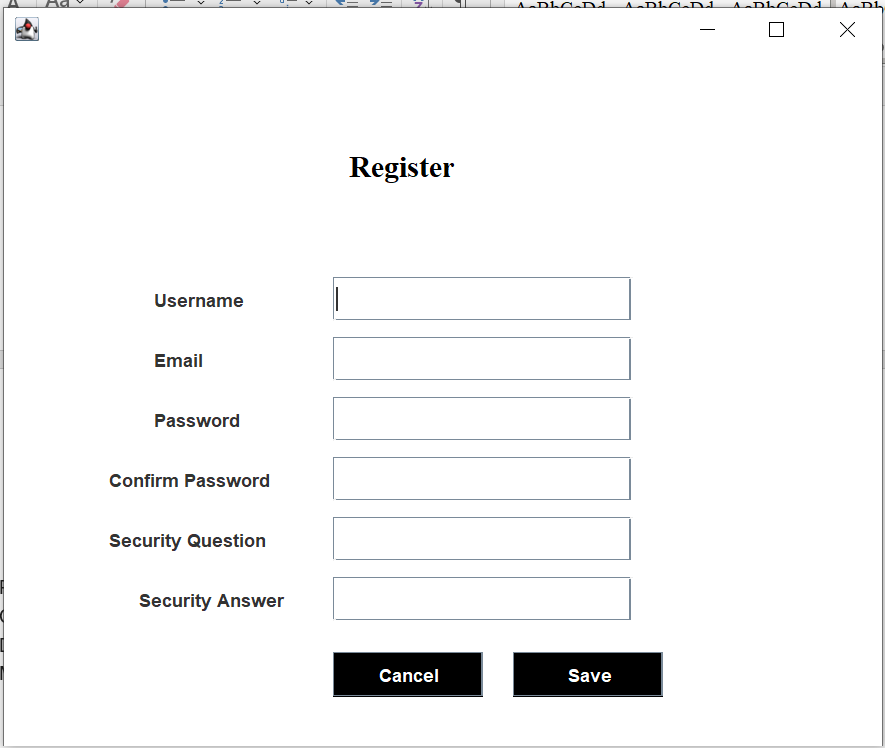
Register register = new Register(); register.actionPerformed(ae);

Output:

The output of the Register class is the registration screen of the Metro Ticketing System. The screen contains six text fields and two buttons: Cancel and Save. The user can enter their username, email, password, security question, and security answer in the text fields. The Cancel button redirects the user to the Home screen. The Save button saves the user's registration information to a file and redirects the user to the Login screen.

Additional notes:

* The writeToFile() method writes the user's registration information to a file called "userInfo.txt". The file is stored in the target/files/userInfo directory.
* The generateUniqueCardNumber() method generates a unique card number for the user. The card number is a 10-digit number that is randomly generated.
* The actionPerformed() method checks the source of the action event and redirects the user to the appropriate screen.



Package: metro.home

Class: Login

Description: This class creates the login screen for the Metro Ticketing System.

Methods:

* public Login(): This is the constructor for the Login class. It initializes the components of the login screen and sets the window properties.
* public boolean verifyCredentials(String enteredUsername, String enteredPassword): This method verifies the entered username and password against the data stored in the signUpFilePath file.
* public void actionPerformed(ActionEvent ae): This method handles the button clicks on the login screen. It redirects the user to the appropriate screen depending on the button clicked.

Usage:

To use the Login class, you first need to create an instance of the class. You can then call the actionPerformed() method to handle the button clicks on the login screen.

Example:

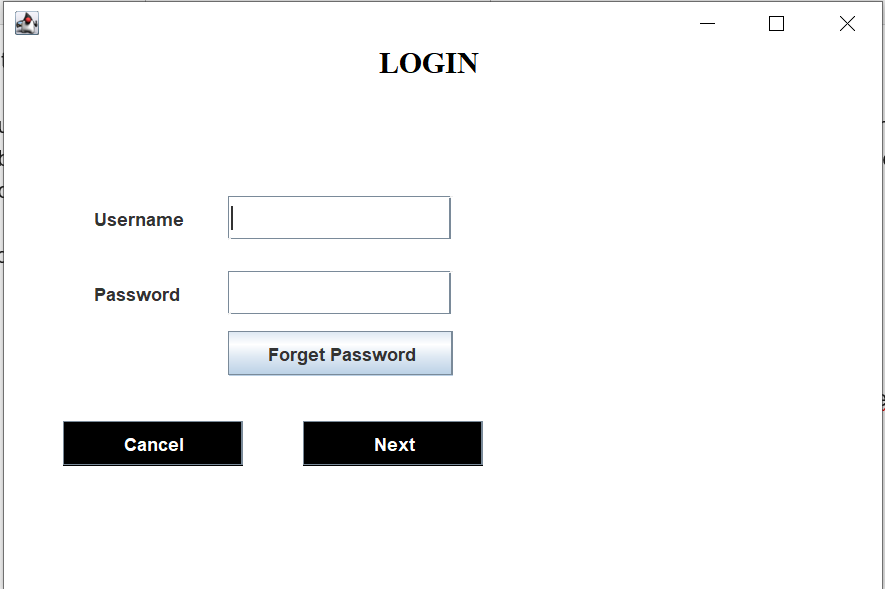
Login login = new Login(); login.actionPerformed(ae);

Output:

The output of the Login class is the login screen of the Metro Ticketing System. The screen contains three buttons: Cancel, Next, and Forget Password. The user can click on any of these buttons to be redirected to the appropriate screen.

Additional notes:

* The verifyCredentials() method uses the signUpFilePath file to store the username and password information for all registered users.
* The actionPerformed() method uses the verifyCredentials() method to verify the entered username and password. If the credentials are valid, the user is redirected to the UserHome screen. Otherwise, the user is shown an error message.



Class: AdminLogin

Description: This class creates the admin login screen for the Metro Ticketing System.

Methods:

* public AdminLogin(): This is the constructor for the AdminLogin class. It initializes the components of the admin login screen and sets the window properties.
* public void actionPerformed(ActionEvent ae): This method handles the button clicks on the admin login screen. It redirects the user to the appropriate screen depending on the button clicked.

Usage:

To use the AdminLogin class, you first need to create an instance of the class. You can then call the actionPerformed() method to handle the button clicks on the admin login screen.

Example:

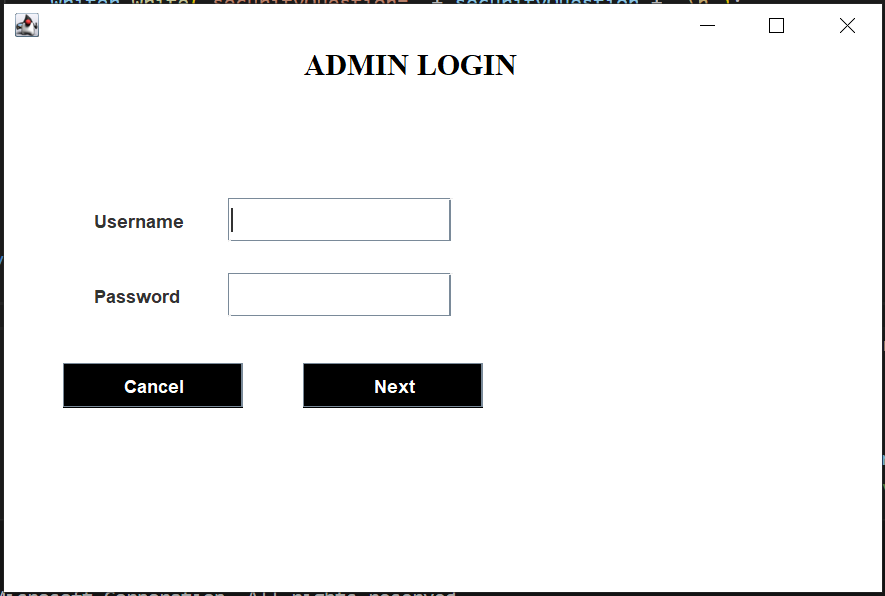
AdminLogin adminLogin = new AdminLogin(); adminLogin.actionPerformed(ae);

Output:

The output of the AdminLogin class is the admin login screen of the Metro Ticketing System. The screen contains two text fields and two buttons: Cancel and Next. The user can enter their username and password in the text fields. The Cancel button redirects the user to the Home screen. The Next button validates the username and password and redirects the user to the AdminHome screen if the credentials are valid. Otherwise, the user is shown an error message.

Additional notes:

* The AdminLogin class inherits from the JFrame class.
* The actionPerformed() method uses the enteredUsername and enteredPassword variables to validate the user's credentials.
* The actionPerformed() method also uses the JOptionPane class to show an error message if the user's credentials are invalid.



Class: UserHome

Description: This class creates the user home screen for the Metro Ticketing System.

Methods:

* public UserHome(): This is the constructor for the UserHome class. It initializes the components of the user home screen and sets the window properties.
* public UserHome(String username): This constructor takes a username as input and saves it to a file.
* public void actionPerformed(ActionEvent ae): This method handles the button clicks on the user home screen. It redirects the user to the appropriate screen depending on the button clicked.

Usage:

To use the UserHome class, you first need to create an instance of the class. You can then call the actionPerformed() method to handle the button clicks on the user home screen.

Example:

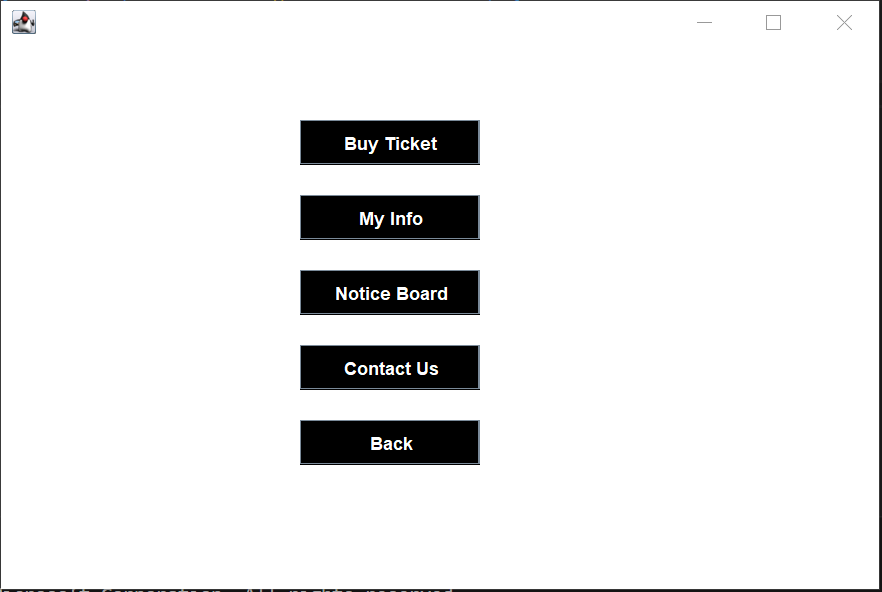
UserHome userHome = new UserHome(); userHome.actionPerformed(ae);

Output:

The output of the UserHome class is the user home screen of the Metro Ticketing System. The screen contains five buttons: Buy Ticket, My Info, Notice Board, Contact Us, and Back. The user can click on any of the buttons to be redirected to the appropriate screen.

Additional notes:

* The UserHome class inherits from the JFrame class.
* The actionPerformed() method uses the ae.getSource() variable to determine which button was clicked.
* The actionPerformed() method then redirects the user to the appropriate screen depending on the button clicked.



Class: ShowUserInfo

Description: This class displays the user's information in a text area.

Methods:

* public ShowUserInfo(): This is the constructor for the ShowUserInfo class. It initializes the components of the ShowUserInfo screen and sets the window properties.
* private void readFile(String filename, String usearnameFromFile) throws IOException: This method reads the user's information from a file and displays it in a text area.

Usage:

To use the ShowUserInfo class, you first need to create an instance of the class. You can then call the readFile() method to read the user's information from a file and display it in a text area.

Example:

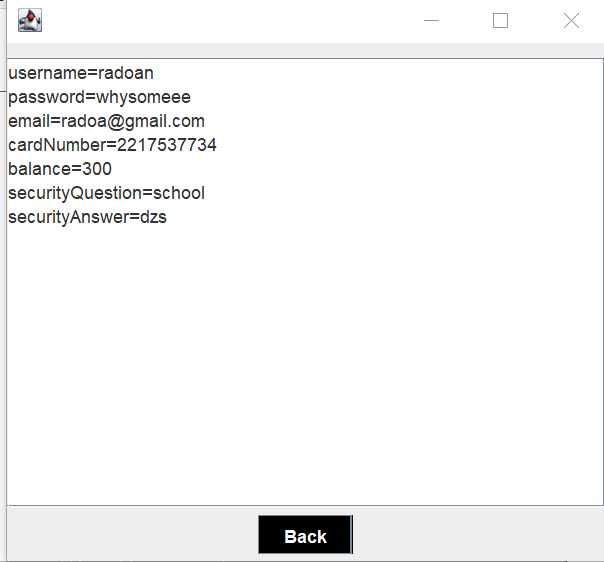
ShowUserInfo showUserInfo = new ShowUserInfo();

Output:

The output of the ShowUserInfo class is a text area that displays the user's information. The information includes the user's username, email, password, security question, and security answer.

Additional notes:

* The ShowUserInfo class inherits from the JFrame class.
* The readFile() method uses the usearnameFromFile variable to find the user's information in the file.
* The readFile() method then displays the user's information in a text area.



Class: NoticeBoard

Description: This class displays a list of notices in a JList.

Methods:

* public NoticeBoard(): This is the constructor for the NoticeBoard class. It initializes the components of the NoticeBoard screen and sets the window properties.
* private void openFile(String filePath): This method opens a file in the user's default text editor.

Usage:

To use the NoticeBoard class, you first need to create an instance of the class. You can then call the openFile() method to open a file in the user's default text editor.

Example:

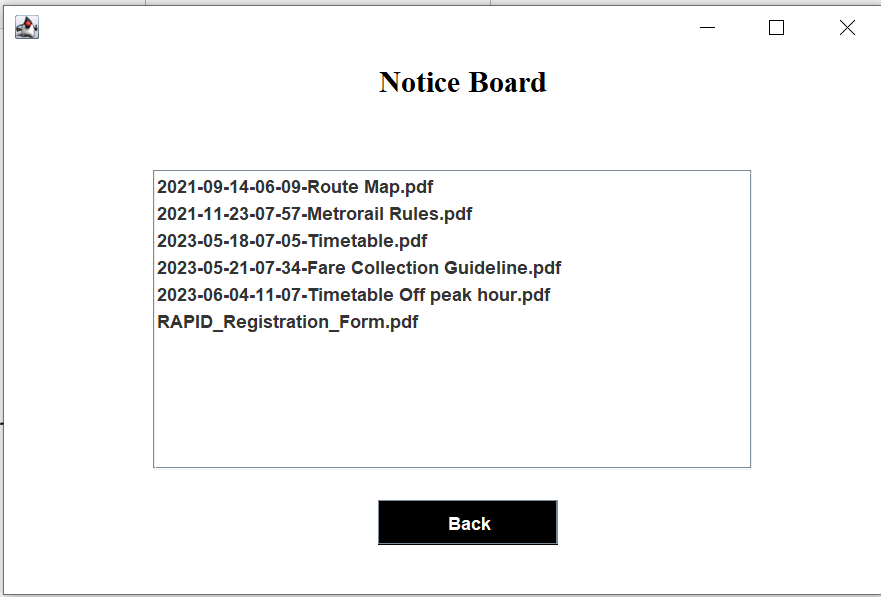
NoticeBoard noticeBoard = new NoticeBoard(); noticeBoard.openFile("target/files/notice/notice1.txt");

Output:

The output of the NoticeBoard class is a JList that displays a list of notices. The user can double-click on a notice to open it in the user's default text editor.

Additional notes:

* The NoticeBoard class inherits from the JFrame class.
* The openFile() method uses the Desktop class to open a file in the user's default text editor.



Class: ContactUs

Description: This class displays the contact information for the Dhaka Mass Transit Company Limited.

Methods:

* public ContactUs(): This is the constructor for the ContactUs class. It initializes the components of the ContactUs screen and sets the window properties.

Usage:

To use the ContactUs class, you first need to create an instance of the class. You can then view the contact information for the Dhaka Mass Transit Company Limited.

Example:

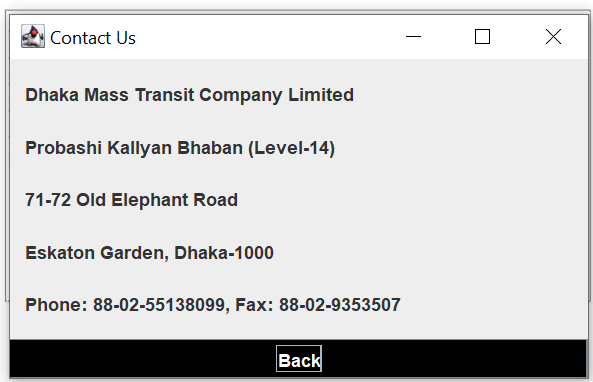
ContactUs contactUs = new ContactUs();

Output:

The output of the ContactUs class is a JFrame that displays the contact information for the Dhaka Mass Transit Company Limited. The contact information includes the company name, address, phone number, and fax number.

Additional notes:

* The ContactUs class inherits from the JFrame class.
* The ContactUs class uses a JPanel to arrange the contact information.
* The ContactUs class uses a JButton to allow the user to return to the UserHome screen.



Class Name: TicketInfo

Description:

The `TicketInfo` class represents a graphical user interface (GUI) component that displays ticket information to the user. It extends the `javax.swing.JFrame` class, indicating its window-like behavior.

Methods:

1. `TicketInfo()`: Constructor method that initializes class variables, reads the target string from a login file, and generates a random number.

Usage:

The `TicketInfo` class can be used to create an instance of a JFrame window that displays ticket information to the user. It is typically used in conjunction with other classes or components in a larger application.

Output:

When the `TicketInfo` window is displayed, it shows the following information:

- "Thank you for your purchase" label.

- The target string read from the login file.

- Ticket ID label followed by a randomly generated number.

Additional Information:

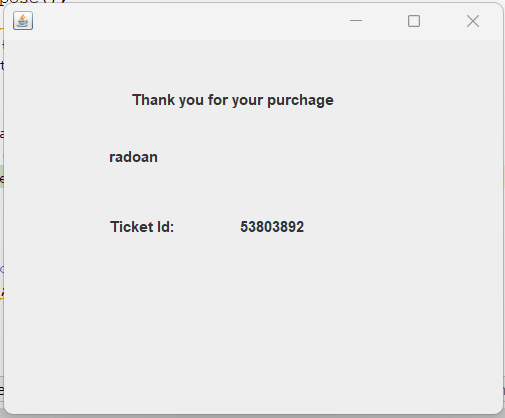
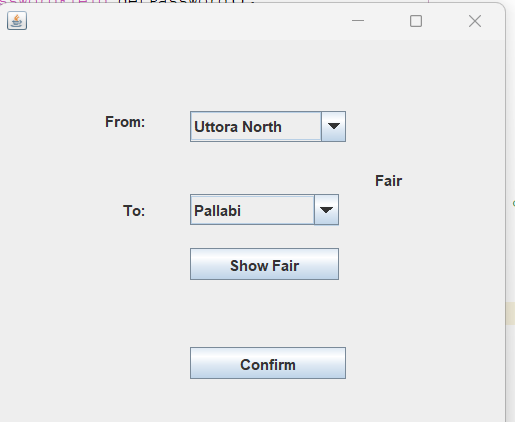
- The code suggests that the `TicketInfo` class is part of a larger application related to a metro or transportation system, as indicated by the package name "metro.home".

- The class makes use of file paths (`userInfoFilePath` and `loginFilePath`) to read information from text files. These files are expected to be present in specific locations.

- The code includes some commented-out code related to checking a string in a file, which is currently not being used.

- The user interface of the `TicketInfo` class is generated using an automated process, as evident from the `initComponents()` method, which is typically created by a graphical IDE like NetBeans.

- The `main()` method serves as the entry point of the program, setting the look and feel of the GUI and creating an instance of the `TicketInfo` class to display the ticket information window.



Class Name: EditNoticeBoard

Description: This class is a Java Swing application that allows users to manage the notice board.

Methods:

* public EditNoticeBoard(): This is the constructor for the class. It initializes the components of the application.
* public void actionPerformed(ActionEvent ae): This method handles the actions of the buttons in the application.
* private void openFile(String filePath): This method opens a file in the default application.

Usage:

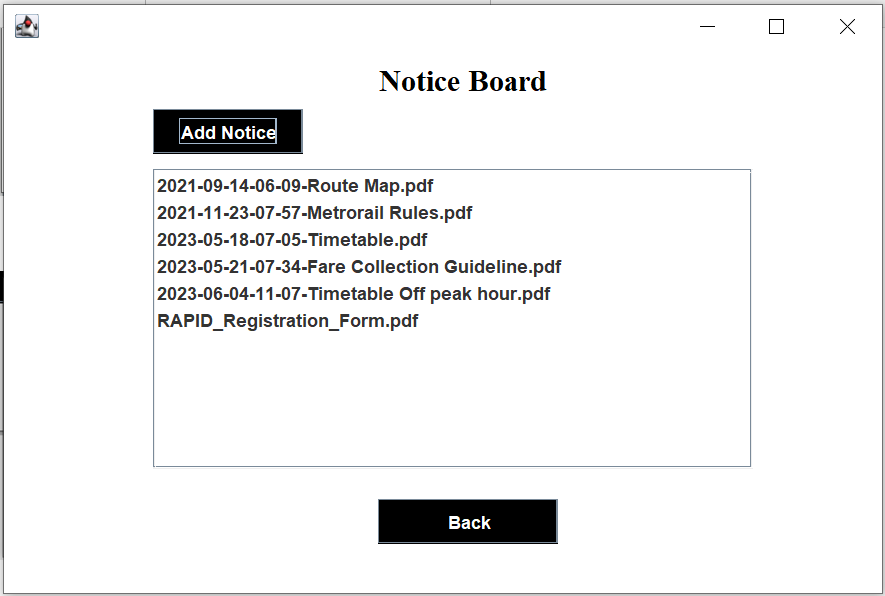
To run the application, you can compile the code and then run the EditNoticeBoard class.

Output:

The application will display a list of notices in a JList component. Users can add new notices by selecting files from their computer and clicking the Add Notice button. Users can open a notice by double-clicking on the name of the notice in the list.

Additional Information:

* The application uses the JList component to display a list of notices.
* The application uses the JFileChooser component to allow users to select files from their computer.
* The application uses the Files class to copy files from one location to another.
* The application uses the Desktop class to open files in the default application.



Class Name: RechargeMoney

Description: This class is a Java Swing application that allows users to recharge the balance of a metro card.

Methods:

* public RechargeMoney(): This is the constructor for the class. It initializes the components of the application.
* private void recharge(String cardNumber, int rechargeAmount): This method recharges the balance of the metro card with the specified card number.

Usage:

To run the application, you can compile the code and then run the RechargeMoney class.

Output:

The application will display a dialog box that asks for the card number and the amount to be recharged. If the card number is valid, the balance of the card will be updated. Otherwise, an error message will be displayed.

Additional Information:

* The application uses the JTextField component to get the card number and the amount to be recharged.
* The application uses the JButton component to handle the actions of the buttons in the application.
* The application uses the BufferedReader and BufferedWriter classes to read and write files.

