

Team 11 Report

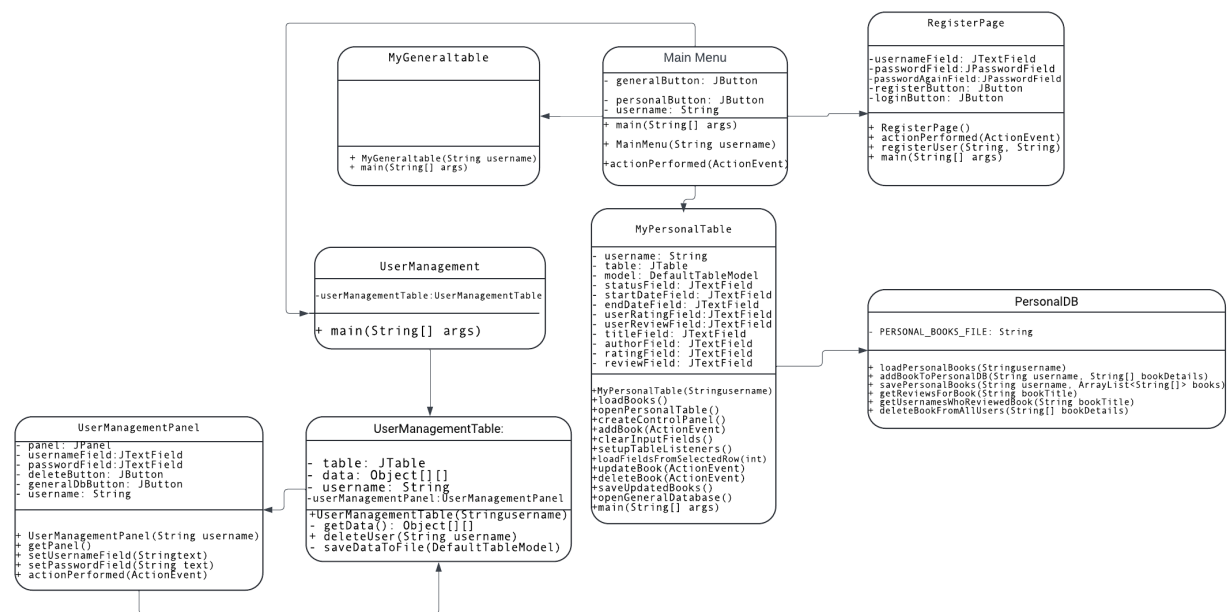
Work allocation.

Full Name	Student ID	Approximation % of Work Done	What was done
Leyla Eynullazada	18033	25%	Wrote part of readme file, wrote some part of code
Rahman Karimzadeh	17969	25%	Wrote part of readme file, wrote some part of code
Riad Baghishov	17368	25%	Wrote part of report, wrote some part of code
Aykhan Asgarov	18128	25%	Wrote part of report, wrote some part of code


Github Link:

[team-11 · ADA-SITE-CSCI-1202 Team \(github.com\)](https://github.com/team-11-ADA-SITE-CSCI-1202-Team)

Youtube Link: https://youtu.be/5SG_R3Q8Zuw?feature=shared



Book Library



Welcome To You


Username:

Password:


Login

Register

Main Menu



General DB



Personal DB

Reviews for Parallel Lives

user555 - Add rating - Add review

Author	Rating	Review
thor	No rating	No reviews
known	5.00 (1)	No reviews
known	4.00 (1)	No reviews
known	5.00 (1)	user555
mer	3.00 (1)	user555
mer	No rating	No reviews
rodotus	No rating	No reviews
phocles	5.00 (1)	user555
schyles	5.00 (1)	user555
ripides	No rating	No reviews
ripides	5.00 (1)	user555
ripides	No rating	No reviews
ripides	No rating	No reviews
ucydides	No rating	No reviews
ato	5.00 (1)	user555
istotle	No rating	No reviews
istotle	4.00 (1)	asiman
istotle	No rating	No reviews
istotle	3.00 (1)	user555
known	No rating	No reviews
icretius	No rating	No reviews
utarch	No rating	No reviews
Aeneid	Virgil	1.00 (1)
Bucolics	Virgil	No rating
Georgics	Virgil	No rating
Annals	Tacitus	No rating
Metamorphoses	Ovid	No rating
Heroides	Ovid	No rating
Amores	Ovid	No rating
The New Testament	Unknown	1.00 (1)
The Twelve Caesars	Suetonius	No rating
Meditations	Marcus Aurelius	No rating
Poems	Catullus	No rating
Poems	Horace	No rating
Discourses	Epictetus	No rating
Plays	Aristophanes	5.00 (1)
Historical Miscellany	Aelianus	No rating
On the Nature of Animals	Aelianus	No rating
Argonautica	Apollonius Rhodius	No rating
Fourteen Byzantine Rulers	Michael Psellus	No rating
The Rise and Fall of the Roman Empire	Edward Gibbon	No rating
The Enneads	Plotinus	No rating
Ecclesiastical History	Eusebius	No rating
Consolations of Philosophy	Boethius	No rating

Search:

Title:

Parallel Lives

Author:

Plutarch

Rating:

No rating

Review:

No reviews

Add to Personal Library

Personal DB

Personal Database - user

Title	Author	Rating	Review	Status	Time Spent	Start Date	End Date	User Rating	User Review
Hamlet	William Shakesp...	No rating	No reviews	Completed	367	10/09/2023	11/09/2024	4	Good Book
Aeneid	Virgil	No rating	No reviews	Ongoing	0	10/08/2023	Not Finished	1	Add review
The New Testa...	Unknown	No rating	No reviews	Not Started	0	Not Started	Not Started	1	Add review

Search:

Add

Delete

Update

General DB

Title:

Author

Rating:

Review:

Status:

Not Started


UserRating:

1

UserReview:

StartDate:

EndDate:



Title	Author	Rating	Review
Title	Author	No rating	No reviews
Mahabharata Bhagavad Gita	Unknown	5.00 (1)	No reviews
Epic of Gilgamesh	Unknown	4.00 (1)	No reviews
The Old Testament	Unknown	5.00 (1)	user555
Iliad	Homer	3.00 (1)	user555
Odyssey	Homer	No rating	No reviews
Histories	Herodotus	No rating	No reviews
Plays	Sophocles	5.00 (1)	user555
Plays	Aeschyles	5.00 (1)	user555
Hippolytus	Euripides	No rating	No reviews
Bacchants	Euripides	5.00 (1)	user555
Electra	Euripides	No rating	No reviews
The Phornician Women	Euripides	No rating	No reviews
The Peloponesian War	Thucydides	No rating	No reviews
Dialogues	Plato	5.00 (1)	user555
Poetics	Aristotle	No rating	No reviews
Physics	Aristotle	4.00 (1)	asiman
Ethics	Aristotle	No rating	No reviews
De Anima	Aristotle	3.00 (1)	user555
Greek Anthology	Unknown	No rating	No reviews
On The Nature of Things	Lucretius	No rating	No reviews
Parallel Lives	Plutarch	No rating	No reviews
Aeneid	Virgil	1.00 (1)	No reviews

Search:

Title:

Author:

Rating:

Review:

Add

Update

Delete

User Mana...

```

import javax.swing.*;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.BufferedImage;

public class MainMenu extends JFrame implements ActionListener {

    JButton generalButton;
    JButton personalButton;
    private static String username;

    public static void main(String[] args) {
        MainMenu mainMenu = new MainMenu(username);
        mainMenu.setVisible(true);
    }

    MainMenu(String username) {
        this.username=username;
        setTitle("Main Menu");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(450, 300);
        setLayout(null);
        setLocationRelativeTo(null);
        getContentPane().setBackground(Color.GRAY);

        // Load images (assuming the images are in the same directory as the Java class)
        ImageIcon generalIcon = new ImageIcon("images\\personaldatabase.png");
        ImageIcon personalIcon = new ImageIcon("images\\personaldatabase.png");

        Image img = generalIcon.getImage();

        // Create a BufferedImage to modify the image
        BufferedImage bufferedImage = new BufferedImage(img.getWidth(null), img.getHeight(null),
            BufferedImage.TYPE_INT_ARGB);
        Graphics2D g2d = bufferedImage.createGraphics();
        g2d.drawImage(img, 0, 0, null);
        g2d.dispose();

        // Modify the color to red
        for (int y = 0; y < bufferedImage.getHeight(); y++) {
            for (int x = 0; x < bufferedImage.getWidth(); x++) {
                int rgb = bufferedImage.getRGB(x, y);
                int alpha = (rgb >> 24) & 0xff;
                // int red = (rgb >> 16) & 0xff;
                // int green = (rgb >> 8) & 0xff;
                // int blue = rgb & 0xff;
                int newRGB = (alpha << 24) | (255 << 16) | (0 << 8) | 0;
                bufferedImage.setRGB(x, y, newRGB);
            }
        }

        // Create a new ImageIcon with the modified image
        ImageIcon generalIconRed = new ImageIcon(bufferedImage);

        // Create labels for the images
        JLabel generalLabel = new JLabel();
        generalLabel.setIcon(generalIconRed);
        generalLabel.setBounds(10, 20, 210, 210);
        add(generalLabel);

        JLabel personalLabel = new JLabel();
        personalLabel.setIcon(personalIcon);
        personalLabel.setBounds(230, 20, 210, 210);
        add(personalLabel);

        // create button of general and personal database
        generalButton = new JButton("General DB");
        generalButton.setBounds(10, 200, 200, 25);
        generalButton.setFont(new Font("Times New Roman", Font.BOLD, 15));
        generalButton.addActionListener(this);
        generalButton.setFocusPainted(false);
        generalButton.setForeground(Color.BLACK);
        add(generalButton);

        personalButton = new JButton("Personal DB");
        personalButton.setBounds(230, 200, 200, 25);
        personalButton.setFont(new Font("Times New Roman", Font.BOLD, 15));
        personalButton.addActionListener(this);
        personalButton.setFocusPainted(false);
        personalButton.setForeground(Color.BLACK);
        add(personalButton);
    }

    @Override
    public void actionPerformed(ActionEvent e){
        if (e.getSource() == generalButton){
            new MyGeneralTable(username);
        }
        else if (e.getSource() == personalButton)
        {
            new MyPersonalTable(username);
        }
    }
}

```

```

import javax.swing.*;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class UserManagementPanel extends JFrame implements ActionListener {
    private JPanel panel;
    private JTextField usernameField, passwordField;
    private JButton deleteButton, generalDbButton;
    String username;

    public UserManagementPanel(String username) {
        this.username = username;
        panel = new JPanel();
        panel.setLayout(null);
        usernameField = new JTextField();
        passwordField = new JTextField();
        deleteButton = new JButton("Delete User");
        generalDbButton = new JButton("GeneralDB");

        JLabel usernameLabel = new JLabel("Username: ");
        usernameLabel.setBounds(40, 390, 150, 30);
        usernameLabel.setFont(new Font("Arial", Font.BOLD, 14));
        panel.add(usernameLabel);
        usernameField.setBounds(140, 390, 200, 30);
        panel.add(usernameField);

        JLabel passwordLabel = new JLabel("Password: ");
        passwordLabel.setBounds(40, 430, 150, 30);
        passwordLabel.setFont(new Font("Arial", Font.BOLD, 14));
        panel.add(passwordLabel);
        passwordField.setBounds(140, 430, 200, 30);
        panel.add(passwordField);

        generalDbButton.setBounds(110, 665, 150, 50);
        generalDbButton.setFont(new Font("Arial", Font.BOLD, 14));
        generalDbButton.addActionListener(this);
        panel.add(generalDbButton);

        deleteButton.setBounds(110, 620, 150, 35);
        deleteButton.setFont(new Font("Arial", Font.BOLD, 14));
        deleteButton.addActionListener(this);
        panel.add(deleteButton);
    }

    JPanel getPanel() {
        return panel;
    }

    public void setUsernameField(String text) {
        usernameField.setText(text);
    }

    public void setPasswordField(String text) {
        passwordField.setText(text);
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == deleteButton) {
            String selectedUsername = usernameField.getText();
            if (!selectedUsername.isEmpty()) {
                UserManagementTable userManagementTable = new UserManagementTable(selectedUsername);
                userManagementTable.deleteUser(selectedUsername);
                this.dispose();
            }
        } else if (e.getSource() == generalDbButton) {
            this.dispose();
            MyGeneralTable generalTable = new MyGeneralTable("admin");
            generalTable.setVisible(true); // Make sure to set the visibility of your General DB table
            this.dispose();
        }
    }
}

```

```

import java.swing.*;
import java.swing.event.ListSelectionEvent;
import java.swing.event.ListSelectionListener;
import java.swing.table.DefaultTableModel;

import java.awt.*;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.PrintWriter;
import java.util.ArrayList;

public class UserManagement {
    public static UserManagementTable userManagementTable;

    public static void main(String[] args) {
        UserManagementTable userManagementTable = new UserManagementTable("admin");
        userManagementTable.setLocationRelativeTo(null); // Center on screen
        userManagementTable.setVisible(true);
    }

    class UserManagementTable extends JFrame {
        private JTable table;
        private Object[][] data;
        private String username;
        private UserManagementPanel userManagementPanel;

        public UserManagementTable(String username) {
            this.username = username;
            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            setBounds(500, 200, 600, 500);
            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            setVisible(true);
            data = getData();
            String[] columnName = {"Username", "Password"};
            DefaultTableModel defaultTableModel = new DefaultTableModel(data, columnName);
            table = new JTable(defaultTableModel);

            Font headerFont = table.getTableHeader().getFont();
            Font newHeaderFont = headerFont.deriveFont(Font.BOLD, 14);
            table.getTableHeader().setFont(newHeaderFont);

            userManagementPanel = new UserManagementPanel(username);
            userManagementPanel.getPanel().setPreferredSize(new Dimension(400, getHeight()));
            add(userManagementPanel.getPanel(), BorderLayout.EAST);

            add(new JScrollPane(table));
            setExtendedState(JFrame.MAXIMIZED_BOTH);
            pack();

            table.getSelectionModel().addListSelectionListener(new ListSelectionListener() {
                @Override
                public void valueChanged(ListSelectionEvent e) {
                    int selectedRow = table.getSelectedRow();
                    if (selectedRow == -1) {
                        userManagementPanel.setUsernameField((String) table.getValueAt(selectedRow, 0));
                        userManagementPanel.setPasswordField((String) table.getValueAt(selectedRow, 1));
                    }
                }
            });
        }

        Object[][] getData() {
            try {
                BufferedReader br = new BufferedReader(new FileReader("csvfiles\\users.csv"));
                ArrayList<Object[]> list = new ArrayList<>();
                String str;
                while ((str = br.readLine()) != null) {
                    String[] parts = str.split(",");
                    if (parts.length > 0) {
                        String username = parts[0];
                        String password = parts[1];
                        list.add(new Object[]{username.trim(), password});
                    }
                }
                br.close();

                Object[][] data = new Object[list.size()][2];
                for (int i = 0; i < list.size(); i++) {
                    data[i] = list.get(i);
                }
                return data;
            } catch (Exception x) {
                x.printStackTrace();
                return null;
            }
        }

        public void deleteUser(String username) {
            DefaultTableModel model = (DefaultTableModel) table.getModel();
            for (int i = 0; i < model.getRowCount(); i++) {
                if (model.getValueAt(i, 0).equals(username)) {
                    model.removeRow(i);
                    break;
                }
            }

            // PersonalLib.deleteUserPersonalLibray(username);
            saveDataToFile(model);
            if (userManagementPanel != null) {
                userManagementPanel.dispose(); // Dispose the panel if it exists
            }
        }

        private void saveDataToFile(DefaultTableModel model) {
            try {
                PrintWriter writer = new PrintWriter(new FileWriter("csvfiles\\users.csv"));
                for (int i = 0; i < model.getRowCount(); i++) {
                    StringBuilder sb = new StringBuilder();
                    for (int j = 0; j < model.getColumnCount(); j++) {
                        sb.append(model.getValueAt(i, j));
                        if (j < model.getColumnCount() - 1) {
                            sb.append(",");
                        }
                    }
                    writer.println(sb.toString());
                }
                writer.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
}

```

```

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;

import javax.swing.*;

public class RegisterPage extends JFrame implements ActionListener {
    private JTextField usernameField;
    private JPasswordField passwordField;
    private JButton registerButton;
    private JButton loginButton;

    public RegisterPage() {
        setTitle("Register");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        // Set frame size to follow our
        setSize(400, 400);
        setLayout(null); // Set the frame to the screen
        getContentPane().setBackground(Color.GRAY);

        // Load the image
        ImageIcon imageIcon = new ImageIcon("images\\login.png");
        JLabel imageLabel = new JLabel(imageIcon, JLabel.CENTER);
        imageLabel.setBounds(100, 50, 200, 100);
        // Add the image label to the frame
        add(imageLabel);

        JLabel firstNameLabel = new JLabel("First Name");
        firstNameLabel.setBounds(10, 100, 100, 30);
        JTextField firstNameField = new JTextField(20);
        firstNameField.setBounds(10, 130, 100, 30);
        add(firstNameLabel);
        add(firstNameField);

        JLabel usernameLabel = new JLabel("Username");
        usernameLabel.setBounds(10, 160, 100, 30);
        JTextField usernameField = new JTextField(20);
        usernameField.setBounds(10, 190, 100, 30);
        add(usernameLabel);
        add(usernameField);

        JLabel passwordLabel = new JLabel("Password");
        passwordLabel.setBounds(10, 220, 100, 30);
        JPasswordField passwordField = new JPasswordField(20);
        passwordField.setBounds(10, 250, 100, 30);
        add(passwordLabel);
        add(passwordField);

        JLabel confirmPasswordLabel = new JLabel("Confirm Password");
        confirmPasswordLabel.setBounds(10, 280, 100, 30);
        JPasswordField confirmPasswordField = new JPasswordField(20);
        confirmPasswordField.setBounds(10, 310, 100, 30);
        add(confirmPasswordLabel);
        add(confirmPasswordField);

        JButton registerButton = new JButton("Register");
        registerButton.setBounds(100, 340, 100, 30);
        JButton loginButton = new JButton("Login");
        loginButton.setBounds(100, 370, 100, 30);
        add(registerButton);
        add(loginButton);

        // Add the text area
        JTextArea textArea = new JTextArea(10, 20);
        textArea.setBounds(10, 400, 100, 30);
        add(textArea);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == registerButton) {
            String username = usernameField.getText().trim();
            String password = passwordField.getText().trim();
            String confirmPassword = confirmPasswordField.getText().trim();

            if (username.isEmpty() || password.isEmpty() || confirmPassword.isEmpty()) {
                JOptionPane.showMessageDialog(this, "Please fill in all fields.");
            } else if (!password.equals(confirmPassword)) {
                JOptionPane.showMessageDialog(this, "Passwords do not match.");
            } else {
                // Save the user information
                saveUser(username, password);
            }
        } else if (e.getSource() == loginButton) {
            // Handle login action
        }
    }

    private void saveUser(String username, String password) {
        // Save the user information to a file
        try {
            FileWriter writer = new FileWriter("users.txt", true);
            writer.write(username + "," + password + "\n");
            writer.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        RegisterPage registerPage = new RegisterPage();
        registerPage.setVisible(true);
    }
}

```



```

import javax.swing.*;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;

public class LoginFrame extends JFrame implements ActionListener {

    public static void main(String[] args) {
        LoginFrame loginFrame = new LoginFrame();
        loginFrame.setVisible(true);
    }

    private JPasswordField usernameTextField;
    private JPasswordField passwordPasswordField;
    private JButton loginButton;
    private JButton registerButton;

    public LoginFrame() {
        setTitle("Book Library");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(480, 520);
        setLayout(null);
        setLocationRelativeTo(null);
        getContentPane().setBackground(Color.GRAY);

        // Load the image
        ImageIcon imageIcon = new ImageIcon("images\\library.png");
        Image image = imageIcon.getImage();
        ImageIcon scaledIcon = new ImageIcon(image.getScaledInstance(60, 60, Image.SCALE_SMOOTH));
        // Create a label to display the image
        JLabel imageLabel = new JLabel(scaledIcon);
        imageLabel.setBounds(120, 20, 60, 60);

        // Add the image label to the frame
        add(imageLabel);

        JLabel label = new JLabel("Welcome to You!");
        label.setBounds(200, 30, 200, 30);
        label.setFont(new Font("Times New Roman", Font.BOLD, 18));
        label.setForeground(Color.BLACK);
        add(label);

        JLabel usernameLabel = new JLabel("Username:");
        usernameLabel.setBounds(110, 90, 80, 30);
        usernameLabel.setFont(new Font("Times New Roman", Font.BOLD, 14));
        usernameLabel.setForeground(Color.BLACK);
        add(usernameLabel);

        usernameTextField = new JPasswordField();
        usernameTextField.setBounds(200, 90, 165, 30);
        usernameTextField.setForeground(Color.BLACK);
        add(usernameTextField);

        JLabel passwordLabel = new JLabel("Password:");
        passwordLabel.setBounds(110, 150, 80, 30);
        passwordLabel.setFont(new Font("Times New Roman", Font.BOLD, 14));
        passwordLabel.setForeground(Color.BLACK);
        add(passwordLabel);

        passwordPasswordField = new JPasswordField();
        passwordPasswordField.setBounds(200, 150, 165, 30);
        passwordPasswordField.setForeground(Color.BLACK);
        add(passwordPasswordField);

        loginButton = new JButton("Login");
        loginButton.setBounds(120, 220, 100, 45);
        loginButton.setFont(new Font("Times New Roman", Font.BOLD, 14));
        loginButton.addActionListener(this);
        loginButton.setForeground(Color.BLACK);
        add(loginButton);

        registerButton = new JButton("Register");
        registerButton.setBounds(170, 220, 100, 45);
        registerButton.setFont(new Font("Times New Roman", Font.BOLD, 14));
        registerButton.addActionListener(this);
        registerButton.setForeground(Color.BLACK);
        add(registerButton);

        // Create users.csv file if it doesn't exist
        File file = new File("csvfiles\\users.csv");
        if (!file.exists()) {
            try {
                file.createNewFile();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

        @Override
        public void actionPerformed(ActionEvent e) {
            if (e.getSource() == loginButton) {
                String username = usernameTextField.getText();
                String password = new String(passwordPasswordField.getPassword());
                if (authenticate(username, password)) {
                    JOptionPane.showMessageDialog(this, "Login Successful");
                    dispose();
                    if (username.equals("admin")) {
                        new MyGeneralTable(username); // pass username to general table
                    } else {
                        MainMenu mainMenu = new MainMenu(username); // pass username to main menu
                        mainMenu.setVisible(true);
                    }
                } else {
                    JOptionPane.showMessageDialog(this, "Invalid Username or Password");
                }
            } else if (e.getSource() == registerButton) {
                this.dispose();
                new RegisterPage().setVisible(true);
            }
        }

        private boolean authenticate(String username, String password) {
            try {
                BufferedReader bufferedReader = new BufferedReader(new FileReader("csvfiles\\users.csv"));
                String line;
                while ((line = bufferedReader.readLine()) != null) {
                    String[] parts = line.split(",");
                    if (parts.length >= 2 && parts[0].equals(username) && parts[1].equals(password)) {
                        return true;
                    }
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
            return false;
        }
    }
}

```

```

import java.io.*;
import java.util.*;
import javax.swing.*;

public class BrodskyDataParser {
    private static final String CSV_FILE_PATH = "csvfiles\\brodsky.csv";

    Object[][] getData() {
        try {
            BufferedReader br = new BufferedReader(new FileReader(CSV_FILE_PATH));
            BufferedWriter bw = new BufferedWriter(new FileWriter("csvfiles\\generalDatabaseUpdated.csv"));
            bw.write("Title" + "Author" + "Rating" + "Review");
            ArrayList<Object[]> list = new ArrayList<>();
            String str;
            int count = 0;
            while ((str = br.readLine()) != null) {
                if (count != 0) {
                    String[] parts;
                    String[] authorBooks = str.split("\\\\");
                    if (authorBooks.length > 1) {
                        String author = (authorBooks[authorBooks.length - 1].trim().length() > 0) ? authorBooks[authorBooks.length - 1].trim() : "Unknown";
                        if (author.contains(",")) {
                            author = author.replace(",", "");
                        }
                        parts = authorBooks[1].split(",");
                        String rating = "No rating";
                        String review = "No review";
                        for (String title : parts) {
                            title = title.replace("[", "");
                            title = title.replace("]", "");
                            list.add(new Object[]{title.trim(), author, rating, review});
                            bw.write(title.trim() + "," + author + "," + rating + "," + review);
                            bw.newLine();
                        }
                    } else {
                        parts = str.split(",");
                        String title = parts.length > 0 && !parts[0].trim().isEmpty() ? parts[0].trim() : "Unknown";
                        String author = parts.length > 1 ? parts[1].trim() : "Unknown";
                        String rating = "No rating";
                        String review = "No review";
                        title = title.replace("[", "");
                        title = title.replace("]", "");
                        list.add(new Object[]{title.trim(), author, rating, review});
                        bw.write(title.trim() + "," + author + "," + rating + "," + review);
                        bw.newLine();
                        System.out.println(str);
                    }
                }
                count++;
            }
            br.close();
            bw.close();
            Object[][] data = new Object[list.size()][4];
            for (int i = 0; i < list.size(); i++) {
                data[i] = list.get(i);
            }
            return data;
        } catch (Exception x) {
            x.printStackTrace();
            return null;
        }
    }

    public static void main(String[] args) {
        BrodskyDataParser parser = new BrodskyDataParser();
        Object[][] data = parser.getData();
        // Assume you have some method to handle the display of this data in your UI
        System.out.println("Data loaded successfully.");
    }
}

```

[illegible][illegible][illegible]