COIMBATORE INSTITUTE OF TECHNOLOGY

CIVIL AERODROME POST, COIMBATORE, TAMILNADU, INDIA–641014

OBJECT ORIENTED PROGRAMMING LANGUAGE LABORATORY

(22MDC38)

CAT II - PROJECT

Tymzy – A time-banking app that rewards productivity with leisure.

SUBMITTED BY :

MITHRA OMPRAKASH

(2403717673722027)

SUBMITTED TO :

Dr. V.SAVITHRI

Ms. K.VANI

Ms. S.DHARANI

1. Abstract

Tymzy is a gamified productivity app that rewards users with "credits" for logging productive activities. Credits can be redeemed for short leisure sessions. The system uses a Java Swing UI for interaction, JDBC (Oracle) for persistent storage, and background threads (`SwingWorker`) for responsiveness. Core features:

- Log activities with timestamp
- Store persistent credits in a database
- Add and consume credits transactionally
- Responsive GUI with dark-themed design and icons

2. Requirement Analysis

2.1 Actors

- **User (Student/Player)**: Adds activities, views activity log, uses credits, views stats.
- **System (Tymzy)**: Persists activities and credits, enforces credit balance logic.

2.2 Functional Requirements

1. FR1 — **Add Activity**: User can add a named activity which is recorded with a timestamp.
2. FR2 — **View Activities**: User can view a chronological list of past activities.
3. FR3 — **Delete Activity**: User can delete an activity from the list.
4. FR4 — **Earn Credits**: Adding activities grants credits according to rules.
5. FR5 — **Use Credits**: User can spend credits if sufficient balance exists; atomic update.
6. FR6 — **View Credits**: User can view current total credits.
7. FR7 — **Responsive UI**: Long-running DB operations must not block the UI.

2.3 Non-functional Requirements

- NFR1 — Persistence: Use Oracle DB with JDBC.
- NFR2 — Usability: Simple Swing GUI, readable fonts, icons.
- NFR3 — Reliability: Transactions for credit updates.

- NFR4 — Performance: Background threads for DB operations.

2.4 Use Case Diagram

3. Design

3.1 Class Diagram

3.2 ER Diagram

3.3 Table Design (SQL)

```
-- activities table
CREATE TABLE activities (
id NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
activity_name VARCHAR2(400) NOT NULL,
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- user_credits table
CREATE TABLE user_credits (
id NUMBER PRIMARY KEY,
total_credits NUMBER DEFAULT 0
);

-- optional seed
INSERT INTO user_credits(id, total_credits) VALUES(1, 0);
```

4. Code

4.1 DBHelper.java

```
import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class DBHelper {
    private static final String URL = "jdbc:oracle:thin:@localhost:1521:XE";
    private static final String USER = "system";
    private static final String PASSWORD = "orcl";
```

```
    static {
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        } catch (ClassNotFoundException e) {
            System.err.println("Oracle JDBC Driver not found. Add ojdbc jar to project
classpath.");
            e.printStackTrace();
        }
    }

    public static Connection connect() throws SQLException {
        return DriverManager.getConnection(URL, USER, PASSWORD);
    }

    public static void insertActivity(String activity) {
        String sql = "INSERT INTO activities(activity_name) VALUES(?)";
        try (Connection conn = connect();
             PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setString(1, activity);
            pstmt.executeUpdate();
        } catch (SQLException e) {
            System.err.println("Insert activity error:");
            e.printStackTrace();
        }
    }

    public static List<String> getAllActivities() {
        List<String> list = new ArrayList<>();
        String sql = "SELECT id, activity_name, created_at FROM activities ORDER BY id ASC";
```

```
    try (Connection conn = connect();
         PreparedStatement pstmt = conn.prepareStatement(sql);
         ResultSet rs = pstmt.executeQuery()) {
        while (rs.next()) {
            String activity = rs.getString("activity_name");
            Timestamp ts = rs.getTimestamp("created_at");
            list.add(activity + " (" + ts + ")");
        }
    } catch (SQLException e) {
        System.err.println("Fetch activities error:");
        e.printStackTrace();
    }
    return list;
}

// Delete an activity by exact name (as your UI does)
public static void deleteActivity(String activity) {
    String sql = "DELETE FROM activities WHERE activity_name = ?";
    try (Connection conn = connect();
         PreparedStatement pstmt = conn.prepareStatement(sql)) {
        pstmt.setString(1, activity);
        pstmt.executeUpdate();
    } catch (SQLException e) {
        System.err.println("Delete activity error:");
        e.printStackTrace();
    }
}

public static void addCredits(int minutes) {
```

```
        String sqlUpdate = "UPDATE user_credits SET total_credits = total_credits + ? WHERE id
= 1";
        String sqlInsert = "INSERT INTO user_credits(id, total_credits) VALUES(1, ?)";
        try (Connection conn = connect()) {
            conn.setAutoCommit(false);
            try (PreparedStatement pstmt = conn.prepareStatement(sqlUpdate)) {
                pstmt.setInt(1, minutes);
                int rows = pstmt.executeUpdate();
                if (rows == 0) {
                    try (PreparedStatement insert = conn.prepareStatement(sqlInsert)) {
                        insert.setInt(1, minutes);
                        insert.executeUpdate();
                    }
                }
                conn.commit();
            } catch (SQLException ex) {
                conn.rollback();
                throw ex;
            } finally {
                conn.setAutoCommit(true);
            }
        } catch (SQLException e) {
            System.err.println("Add credits error:");
            e.printStackTrace();
        }
    }

    public static boolean useCredits(int minutes) {
        String sql = "UPDATE user_credits SET total_credits = total_credits - ? WHERE id = 1 AND
total_credits >= ?";
```

```
        try (Connection conn = connect();
             PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setInt(1, minutes);
            pstmt.setInt(2, minutes);
            int rows = pstmt.executeUpdate();
            return rows > 0;
        } catch (SQLException e) {
            System.err.println("Use credits error:");
            e.printStackTrace();
            return false;
        }
    }

    public static int getCredits() {
        String sql = "SELECT total_credits FROM user_credits WHERE id = 1";
        try (Connection conn = connect();
             PreparedStatement pstmt = conn.prepareStatement(sql);
             ResultSet rs = pstmt.executeQuery()) {
            if (rs.next()) {
                return rs.getInt("total_credits");
            }
        } catch (SQLException e) {
            System.err.println("Get credits error:");
            e.printStackTrace();
        }
        return 0;
    }
}
```

4.2 MainUI.java

```
import javax.swing.*;
import javax.swing.border.LineBorder;
import java.awt.*;
import java.util.List;

public class MainUI extends JFrame {
    private JTextField txtActivity;
    private JList<String> activityList;
    private DefaultListModel<String> listModel;
    private JLabel lblCredits;

    public MainUI() {
        setTitle("Tymzy 🎯 Activity Logger");
        setSize(650, 500);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(null);
        getContentPane().setBackground(new Color(30, 30, 60));

        Font headerFont = new Font("Arial", Font.BOLD, 22);
        Font normalFont = new Font("Verdana", Font.PLAIN, 14);

        JLabel lblWelcome = new JLabel("Welcome to Tymzy !");
        lblWelcome.setBounds(20, 10, 400, 30);
        lblWelcome.setFont(headerFont);
        lblWelcome.setForeground(new Color(0, 102, 204));
        add(lblWelcome);

        txtActivity = new JTextField();
```

```
txtActivity.setBounds(20, 50, 250, 32);
txtActivity.setFont(normalFont);
add(txtActivity);

ImageIcon addIcon    = resizeIcon("/icons/tick.png", 22, 22);
ImageIcon deleteIcon = resizeIcon("/icons/cross.png", 22, 22);
ImageIcon useIcon    = resizeIcon("/icons/controller.png", 22, 22);
ImageIcon homeIcon   = resizeIcon("/icons/home.png", 22, 22);

JButton btnAdd = new JButton(" Add Activity", addIcon);
btnAdd.setBounds(290, 50, 150, 32);
styleButton(btnAdd, new Color(76, 175, 80));
add(btnAdd);

JButton btnDelete = new JButton(" Delete Activity", deleteIcon);
btnDelete.setBounds(450, 50, 160, 32);
styleButton(btnDelete, new Color(244, 67, 54));
add(btnDelete);

listModel = new DefaultListModel<>();
activityList = new JList<>(listModel);
activityList.setFont(new Font("Monospaced", Font.PLAIN, 14));

activityList.setCellRenderer((list, value, index, isSelected, cellHasFocus) -> {
    JLabel label = new JLabel(value.toString());
    label.setOpaque(true);
    if (isSelected) {
        label.setBackground(new Color(100, 149, 237)); // blue selection
        label.setForeground(Color.WHITE);
```

```
            } else {
                label.setBackground((index % 2 == 0) ? new Color(230, 240, 250) : new Color(255,
245, 238));
                label.setForeground(new Color(50, 50, 50));
            }
            label.setBorder(BorderFactory.createEmptyBorder(4, 6, 4, 6));
            return label;
        });

        JScrollPane scroll = new JScrollPane(activityList);
        scroll.setBounds(20, 100, 590, 250);
        add(scroll);

        lblCredits = new JLabel("Credits: 0 mins");
        lblCredits.setBounds(20, 370, 250, 32);
        lblCredits.setFont(new Font("Verdana", Font.BOLD, 15));
        lblCredits.setForeground(new Color(128, 0, 128));
        add(lblCredits);

        JButton btnUse = new JButton(" Use Credits", useIcon);
        btnUse.setBounds(290, 370, 160, 32);
        styleButton(btnUse, new Color(33, 150, 243));
        add(btnUse);

        JButton btnHome = new JButton(" Home", homeIcon);
        btnHome.setBounds(460, 370, 150, 32);
        styleButton(btnHome, new Color(255, 140, 0)); // orange
        add(btnHome);

        loadPastActivities();
```

```
updateCreditsLabel();

btnAdd.addActionListener(e -> {
    String activity = txtActivity.getText().trim();
    if (!activity.isEmpty()) {
        new SwingWorker<Void, Void>() {
            int earnedCredits = calculateCreditsFromTask(activity);

            @Override
            protected Void doInBackground() {
                DBHelper.insertActivity(activity);
                DBHelper.addCredits(earnedCredits);
                return null;
            }

            @Override
            protected void done() {
                loadPastActivities();
                updateCreditsLabel();
                JOptionPane.showMessageDialog(MainUI.this,
                        "Task added ✅ +" + earnedCredits + " mins credits earned!");
                txtActivity.setText("");
            }
        }.execute();
    } else {
        JOptionPane.showMessageDialog(this, "Please enter an activity!");
    }
});
```

```
btnDelete.addActionListener(e -> {
    String selected = activityList.getSelectedValue();
    if (selected != null) {
        String activityName = selected.split("\\(")[0].trim();
        new SwingWorker<Void, Void>() {
            @Override
            protected Void doInBackground() {
                DBHelper.deleteActivity(activityName);
                return null;
            }

            @Override
            protected void done() {
                loadPastActivities();
            }
        }.execute();
    } else {
        JOptionPane.showMessageDialog(this, "Please select an activity to delete.");
    }
});

btnUse.addActionListener(e -> {
    String minsStr = JOptionPane.showInputDialog(this, "Enter minutes to use:");
    if (minsStr == null) return;
    int mins;
    try {
        mins = Integer.parseInt(minsStr.trim());
    } catch (NumberFormatException nfe) {
        JOptionPane.showMessageDialog(this, "Enter a valid number!");
```

```
            return;
        }
        if (mins <= 0) {
            JOptionPane.showMessageDialog(this, "Enter a positive number!");
            return;
        }

        new SwingWorker<Boolean, Void>() {
            @Override
            protected Boolean doInBackground() {
                return DBHelper.useCredits(mins);
            }

            @Override
            protected void done() {
                try {
                    boolean ok = get();
                    if (ok) {
                        updateCreditsLabel();
                        JOptionPane.showMessageDialog(MainUI.this, "Enjoy your leisure! 🎉 -" +
mins + " mins");
                    } else {
                        JOptionPane.showMessageDialog(MainUI.this, "Not enough credits!");
                    }
                } catch (Exception ex) {
                    JOptionPane.showMessageDialog(MainUI.this, "Error using credits: " +
ex.getMessage());
                }
            }
        }.execute();
```

```
        });

        btnHome.addActionListener(e -> {
            dispose();
            HomePage home = new HomePage();
            home.setLocationRelativeTo(null);
            home.setVisible(true);
        });
    }

    private ImageIcon resizeIcon(String path, int width, int height) {
        java.net.URL imgURL = getClass().getResource(path);
        if (imgURL != null) {
            ImageIcon icon = new ImageIcon(imgURL);
            Image img = icon.getImage().getScaledInstance(width, height,
Image.SCALE_SMOOTH);
            return new ImageIcon(img);
        } else {
            System.err.println("⚠ Icon not found: " + path);
            return null;
        }
    }

    private void styleButton(JButton btn, Color bg) {
        btn.setBackground(bg);
        btn.setForeground(Color.WHITE);
        btn.setFocusPainted(false);
        btn.setBorder(new LineBorder(new Color(200, 200, 200)));
        btn.setFont(new Font("Tahoma", Font.BOLD, 13));
        btn.setOpaque(true);
```

```
        btn.setHorizontalAlignment(SwingConstants.LEFT);
    }

    private void loadPastActivities() {
        new SwingWorker<List<String>, Void>() {
            @Override
            protected List<String> doInBackground() {
                return DBHelper.getAllActivities();
            }

            @Override
            protected void done() {
                try {
                    List<String> pastActivities = get();
                    listModel.clear();
                    for (String act : pastActivities) {
                        listModel.addElement(act);
                    }
                } catch (Exception e) {
                    JOptionPane.showMessageDialog(MainUI.this, "Error loading activities: " +
e.getMessage());
                }
            }
        }.execute();
    }

    private void updateCreditsLabel() {
        new SwingWorker<Integer, Void>() {
            @Override
            protected Integer doInBackground() {
```

```
            return DBHelper.getCredits();
        }

        @Override
        protected void done() {
            try {
                int credits = get();
                lblCredits.setText("Credits: " + credits + " mins");
            } catch (Exception e) {
                JOptionPane.showMessageDialog(MainUI.this, "Error updating credits: " +
e.getMessage());
            }
        }
    }.execute();
  }

  private int calculateCreditsFromTask(String activity) {
    activity = activity.toLowerCase();
    if (activity.contains("hour")) {
        return 30;
    } else if (activity.contains("30 min") || activity.contains("30min")) {
        return 15;
    } else {
        return 10; // default
    }
  }

  public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        MainUI ui = new MainUI();
```

```
            ui.setLocationRelativeTo(null);
            ui.setVisible(true);
        });
    }
}
```

4.3 HomePage.java

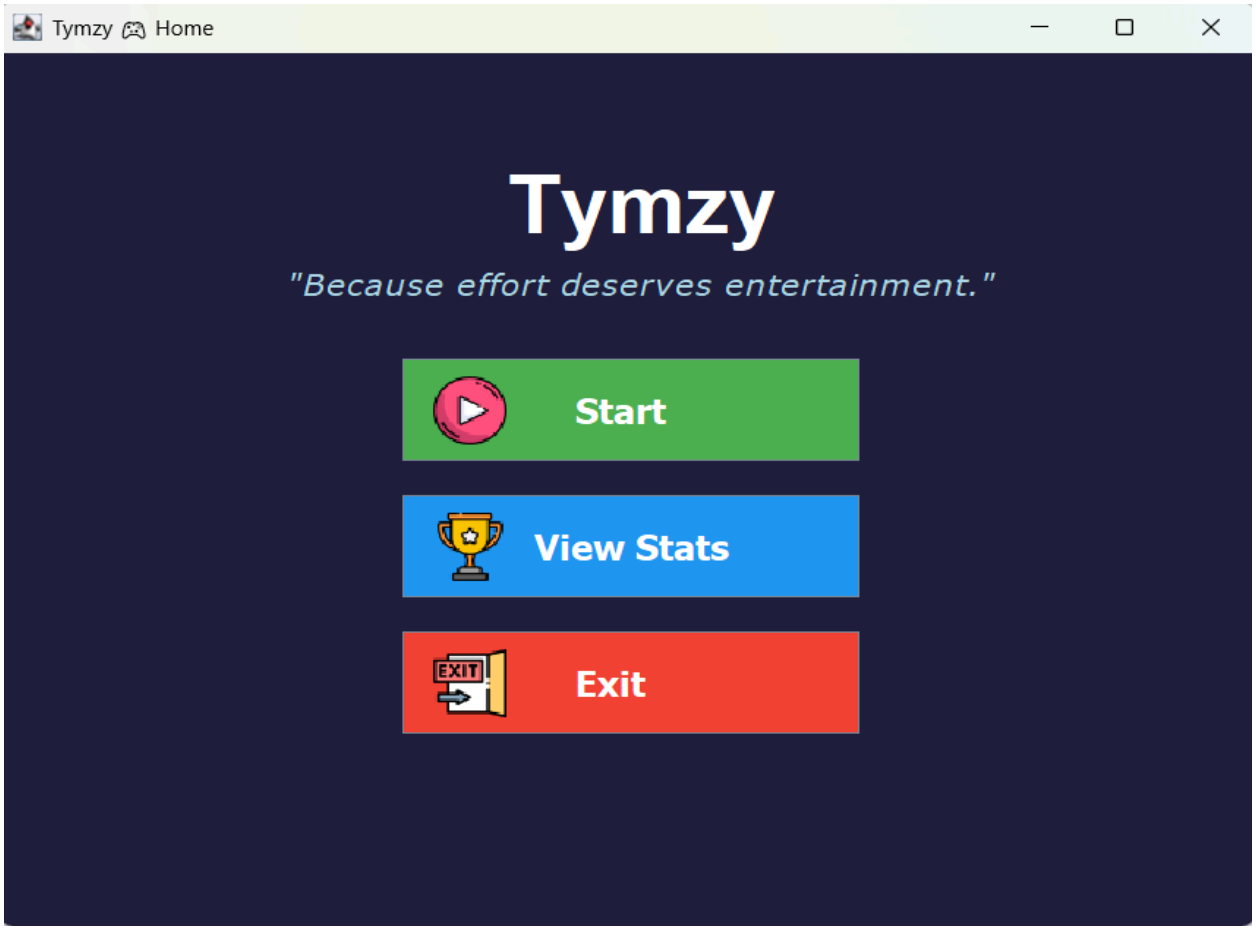
```
import javax.swing.*;
import java.awt.*;
public class HomePage extends JFrame {
    public HomePage() {
        setTitle("Tymzy 🎮 Home");
        setSize(700, 550);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(null);

        getContentPane().setBackground(new Color(30, 30, 60)); // dark gaming style

        JLabel title = new JLabel("Tymzy", SwingConstants.CENTER);
        title.setFont(new Font("Arial", Font.BOLD, 48));
        title.setForeground(Color.WHITE);
        title.setBounds(150, 60, 400, 60);
        add(title);

        JLabel tagline = new JLabel("\"Because effort deserves entertainment.\"",
SwingConstants.CENTER);
        tagline.setFont(new Font("Verdana", Font.ITALIC, 18));
        tagline.setForeground(new Color(173, 216, 230));
        tagline.setBounds(100, 120, 500, 30);
```

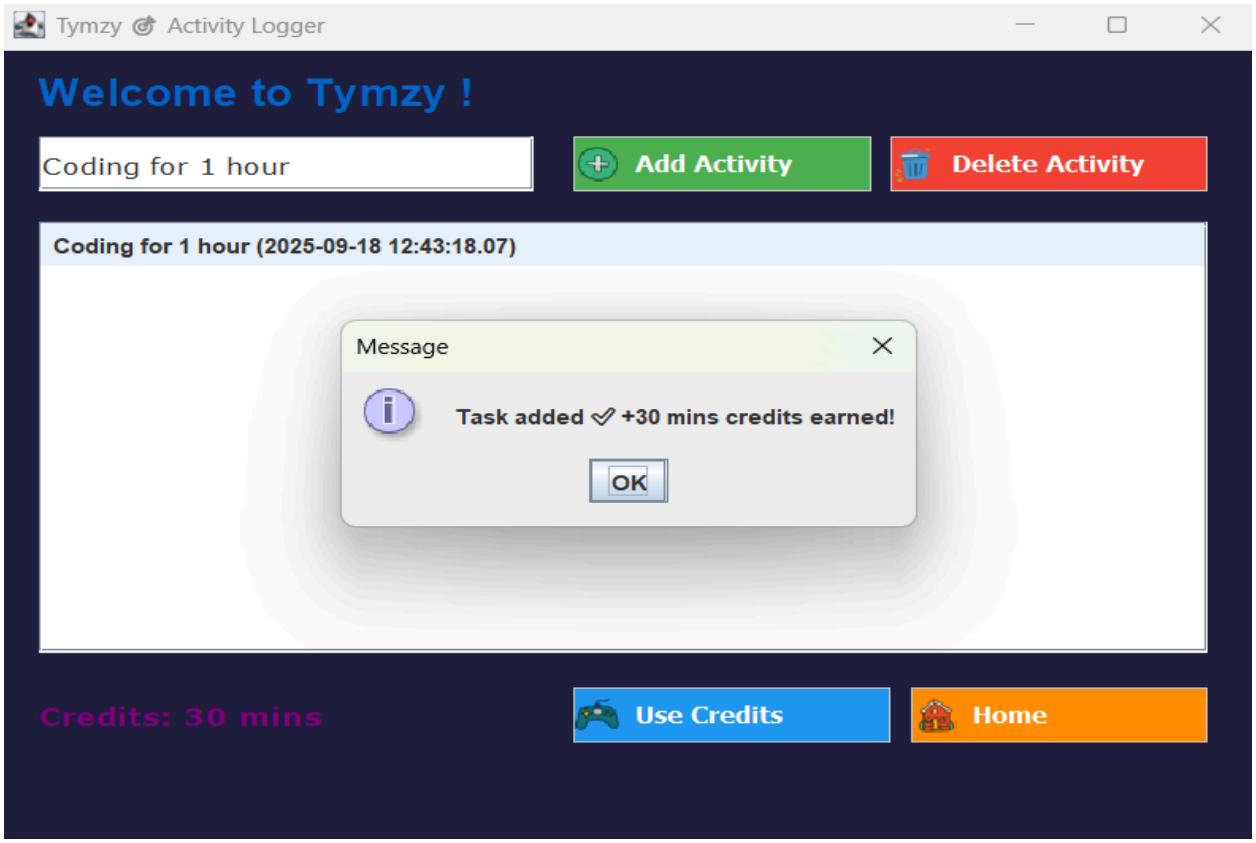
```
add(tagline);

ImageIcon startIcon = resizeIcon("/icons/play.png", 40, 40);
ImageIcon statsIcon = resizeIcon("/icons/trophy.png", 40, 40);
ImageIcon exitIcon  = resizeIcon("/icons/exit.png", 40, 40);

JButton btnStart = new JButton("      Start", startIcon);
btnStart.setBounds(220, 180, 250, 60);
styleButton(btnStart, new Color(76, 175, 80));
add(btnStart);

JButton btnStats = new JButton("  View Stats", statsIcon);
btnStats.setBounds(220, 260, 250, 60);
styleButton(btnStats, new Color(33, 150, 243));
add(btnStats);

JButton btnExit = new JButton("      Exit", exitIcon);
btnExit.setBounds(220, 340, 250, 60);
styleButton(btnExit, new Color(244, 67, 54));
add(btnExit);

btnStart.addActionListener(e -> {
    dispose(); // close home
    MainUI mainUI = new MainUI();
    mainUI.setLocationRelativeTo(null);
    mainUI.setVisible(true);
});

btnStats.addActionListener(e -> {
```

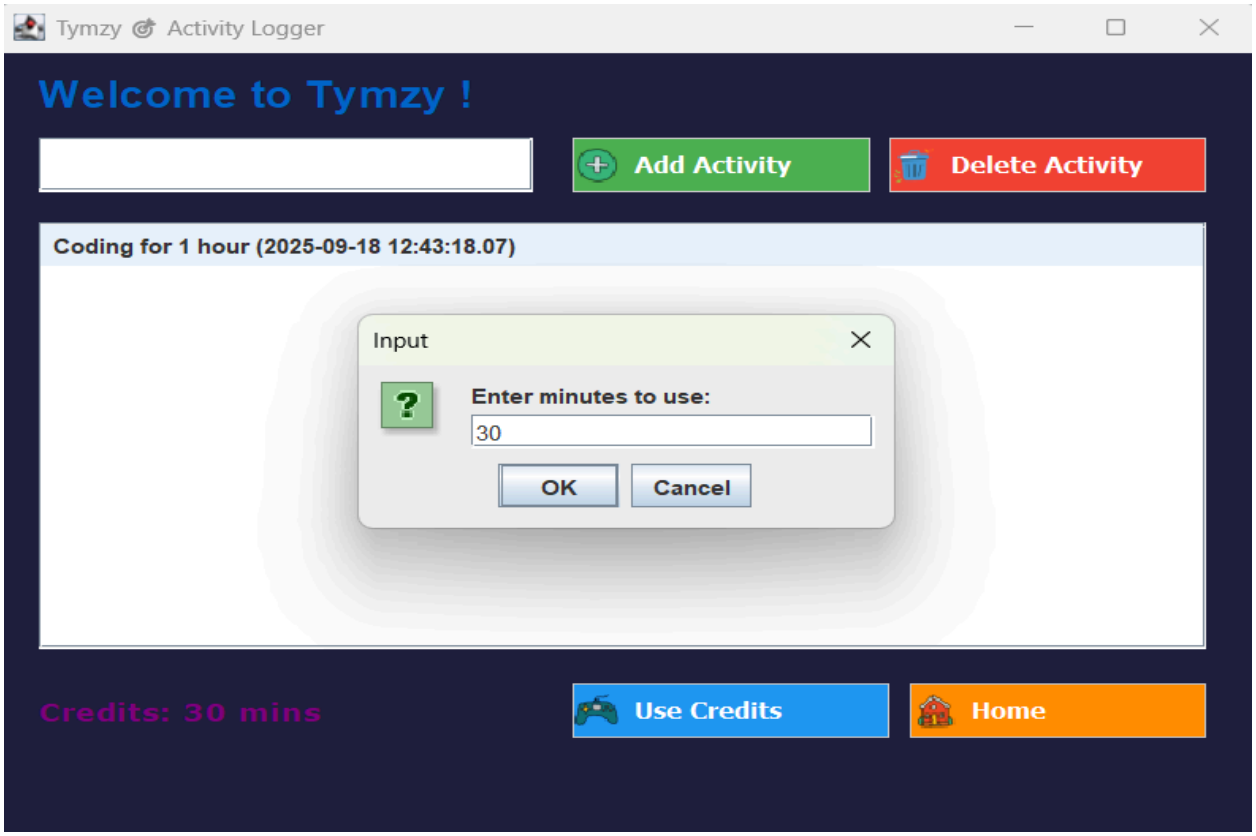
```
            int credits = DBHelper.getCredits();
            JOptionPane.showMessageDialog(this,
                    "🏆 Your current credits: " + credits + " mins",
                    "Player Stats", JOptionPane.INFORMATION_MESSAGE);
        });

        btnExit.addActionListener(e -> System.exit(0));
    }

    private ImageIcon resizeIcon(String path, int width, int height) {
        java.net.URL imgURL = getClass().getResource(path);
        if (imgURL != null) {
            ImageIcon icon = new ImageIcon(imgURL);
            Image img = icon.getImage().getScaledInstance(width, height,
Image.SCALE_SMOOTH);
            return new ImageIcon(img);
        } else {
            System.err.println("⚠ Icon not found: " + path);
            return null;
        }
    }

    private void styleButton(JButton btn, Color bg) {
        btn.setBackground(bg);
        btn.setForeground(Color.WHITE);
        btn.setFocusPainted(false);
        btn.setFont(new Font("Tahoma", Font.BOLD, 20));
        btn.setOpaque(true);
        btn.setHorizontalAlignment(SwingConstants.LEFT);     }
```

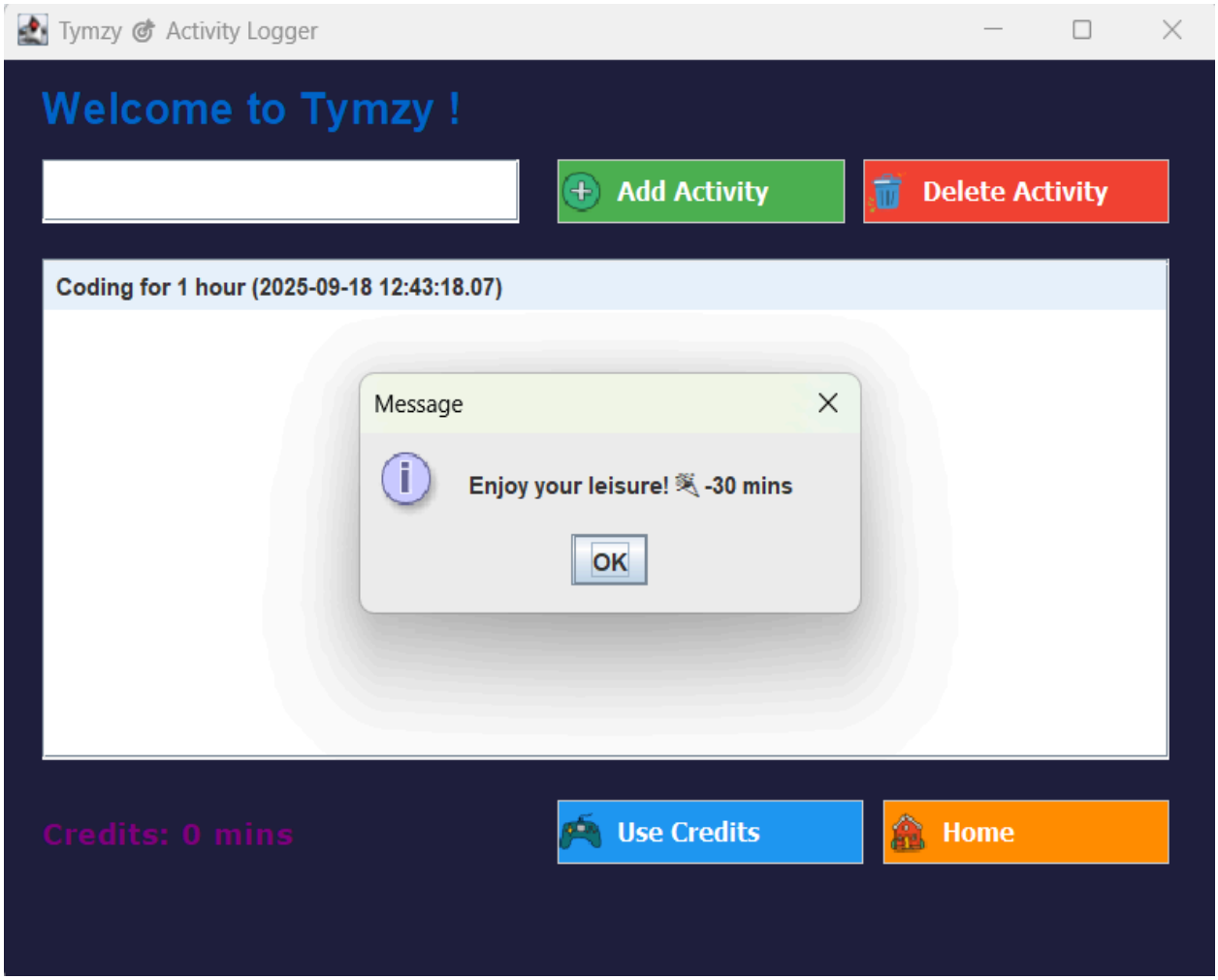
```
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            HomePage home = new HomePage();
            home.setLocationRelativeTo(null);
            home.setVisible(true);
        });
    }
}
```

6. Output




Tymzy Activity Logger
Welcome to Tymzy !
Coding for 1 hour
Add Activity
Delete Activity
Coding for 1 hour (2025-09-18 12:43:18.07)
Message
Task added ✓ +30 mins credits earned!
OK
Credits: 30 mins
Use Credits
Home


Tymzy Activity Logger
Welcome to Tymzy !
Add Activity
Delete Activity
Coding for 1 hour (2025-09-18 12:43:18.07)
Input
Enter minutes to use:
30
OK
Cancel
Credits: 30 mins
Use Credits
Home


Tymzy Activity Logger
Welcome to Tymzy !
Add Activity
Delete Activity
Coding for 1 hour (2025-09-18 12:43:18.07)
Message
Enjoy your leisure! -30 mins
OK
Credits: 0 mins
Use Credits
Home