Southern Luzon State University
College of Engineering
CPE Department

CPE18 - SOFTWARE DESIGN & DEVELOPMENT

Health and Fitness Companion

Title of Software

Name:
CASIDO, Orven P.

Section:
BSCpE III-IF

Date of Submission:
June 6, 2024

Introduction

In the evolving landscape of digital health and wellness, the fitness application serves as a tool designed to address diverse fitness needs. The application encompasses five features that synergistically facilitate a fitness experience.

The application offers an exercise feature segmented into ten distinct modes, each specified to varying difficulty levels: easy, medium, and hard. These modes are meticulously designed to accommodate users across the fitness spectrum, ensuring progressive and adaptive training routines. Users have the flexibility to pause, restart, skip, and resume exercises, providing a customizable workout experience that aligns with individual preferences and schedules.

The Body Mass Index (BMI) counter is an integral tool within the app, offering precise BMI calculations to help users gauge their weight status. The inclusion of a unit conversion feature enhances its utility, allowing seamless transitions between different measurement units. This functionality not only broadens the app's accessibility but also ensures accuracy and ease of use for a global audience.

A balanced diet is fundamental to any fitness regime. The advanced meal planner feature empowers users to address their meals based on personal dietary preferences and nutritional requirements. Users can input ingredients, and design meal plans. The daily planner component further organizes meals according to the days of the week, promoting consistency and adherence to dietary goals.

Time management is also significant for maintaining a disciplined fitness routine. The integrated alarm clock feature is designed to help users schedule their workouts, meals, and other daily activities. By setting alarms, users can ensure they adhere to their planned routines, thereby enhancing the consistency and effectiveness of their fitness regimen.

The application supports a comprehensive library of exercises, providing users with a diverse range of workout options. Each exercise is accompanied by detailed instructions and visual aids to ensure proper execution and maximize benefits. This feature serves as an educational resource, enabling users to expand their knowledge of different exercises and incorporate variety into their fitness routines.

Table of Contents

SLDC	g. 4
HOMEPAGE DESIGN / MENU & LOGO	pg. 5
SOFTWARE DATA MODELING	pg.5 - 6
SOFTWARE LAYOUT DESIGN	pg. 7 - 8
SUMMARY OF CODES	pg. 9 - 108
SCREENSHOT OF ACTUAL USAGE	pg. 109 - 112

I. SLDC

Concept Creation

Concept: Personalized Health and Fitness Companion

Abstract: In response, to the prevalent issue of sedentary lifestyle and declining fitness levels among many individuals, the software aims the development of a personalized health and fitness companion application. The software aims to address these concerns by providing tailored daily exercise routines to promote physical activity and improve overall fitness. Additionally, it will feature a meal planner, offering customized meal plans based on user/s body com[position and nutritional needs.

Introducing the innovative software, your personalized health and fitness companion. Designed to empower you wellness journey, it offers guidance based on your unique goals and preferences. With insights at your fingertips, you'll receive customized recommendations and support every step of the way. From tracking your progress to providing innovation, the companion is there to help you achieve you health and fitness aspirations with ease and confidence.

Title

Orbs: Personalized Health and Fitness Companion

The software name "Orbs"is derived from the developer's name, "Orven". As the application is an AI type companion for fitness. Thus the author want to name it after himself as a coach or an instructor.

Intended Audience

The application is for anyone looking to improve their health and fitness. Whether you're a beginner of experienced enthusiast, busy professional, or health-conscious senior, the software offers personalized solutions to help you achieve your goals. Mainly, the target audience were persons who live sedentary lifestyle.

Features

The software offers these possible features:

1.) Meal Planner - Plan personalized meals based on dietary preferences and health goals.

2.) Dedicated Exercise - Access guided workout routines tailored to fitness levels and goals.

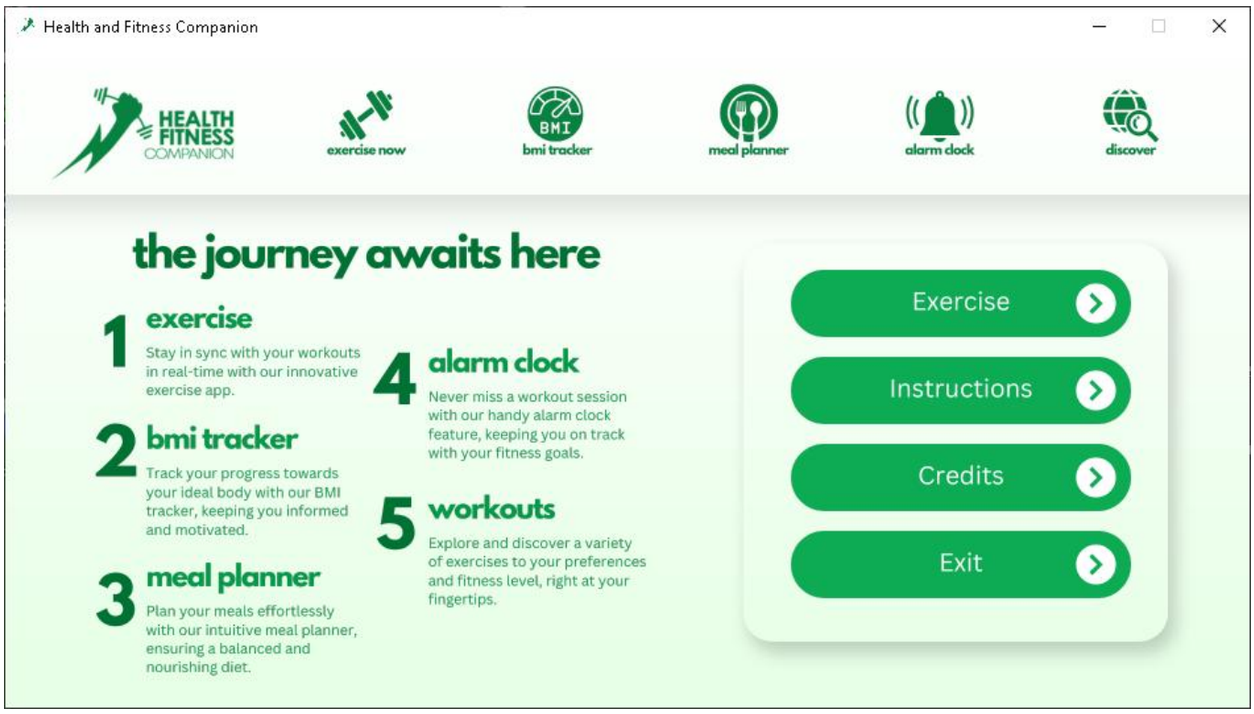
3.) Alarm Clock System - Set reminders for meals, and workouts

4.) BMI Tracking - Monitor body changes in composition to access progress and adjust plans.

5.) Library of Exercises - Browse diverse library of exercises.

II. HOMEPAGE DESIGN / MENU & LOGO

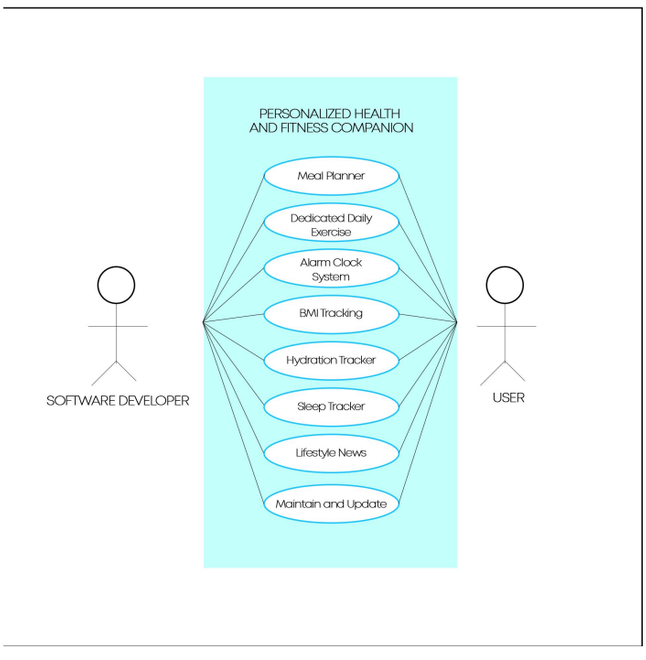
HOMEPAGE DESIGN



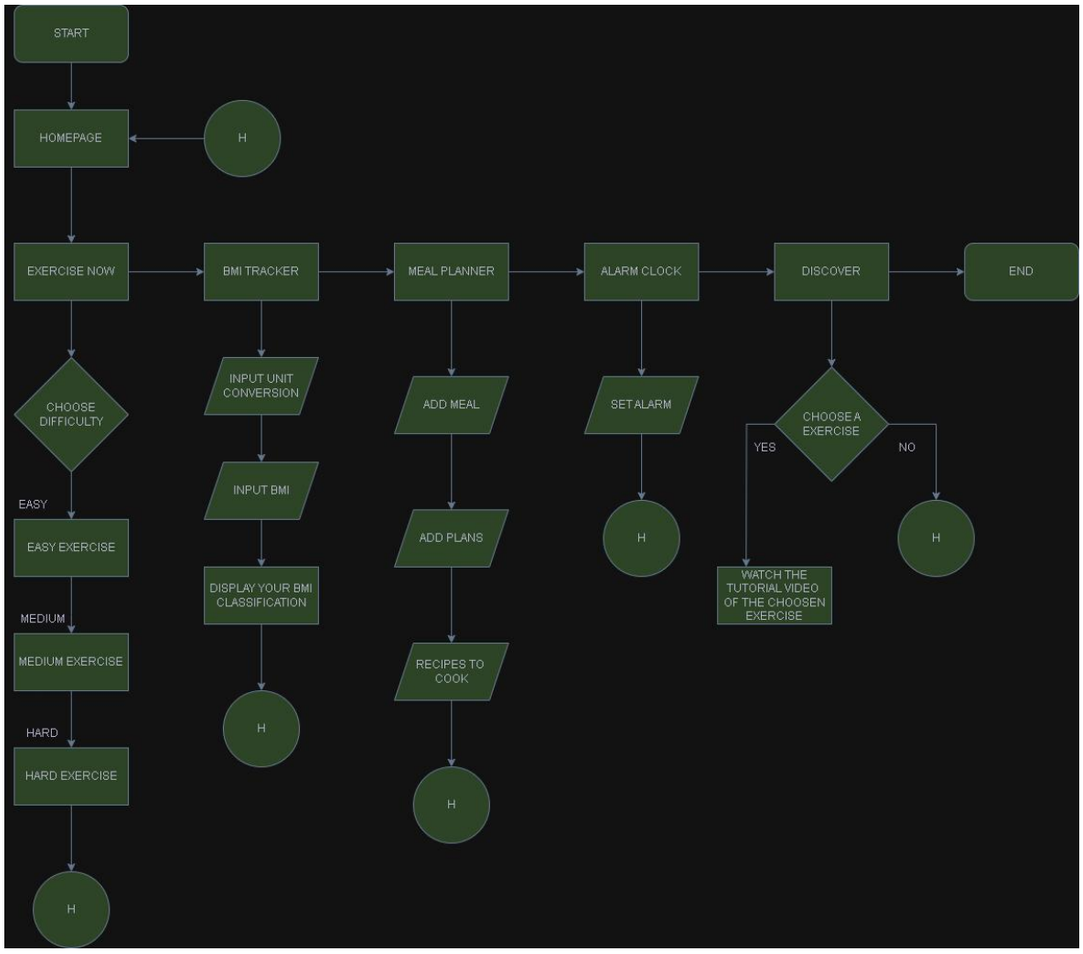
LOGO

III. SOFTWARE DATA MODELING

USE CASES



FLOWCHART

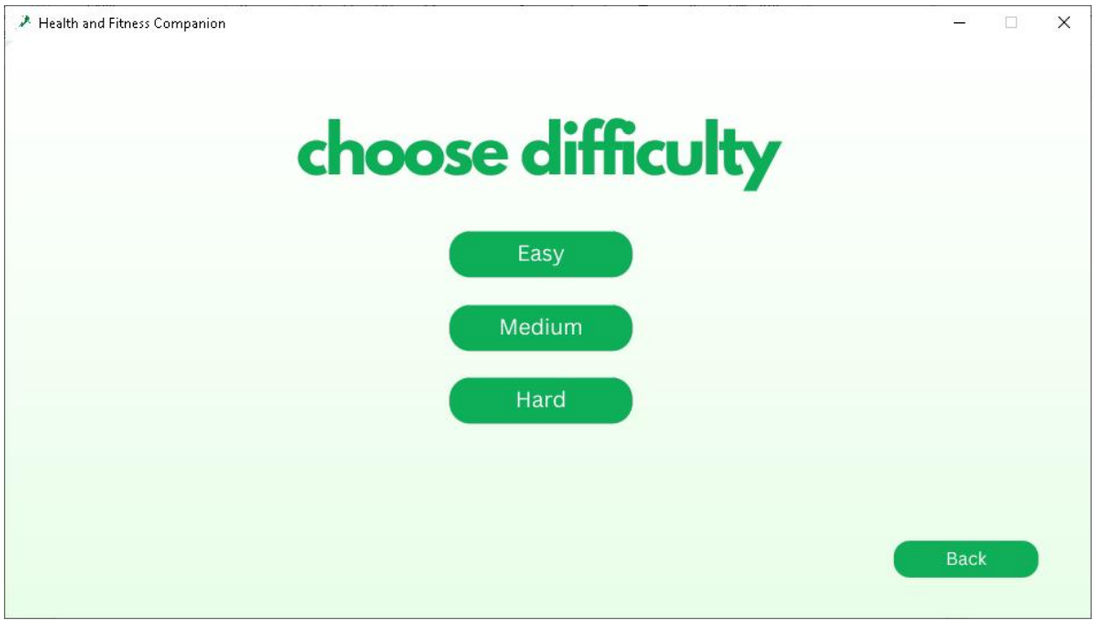


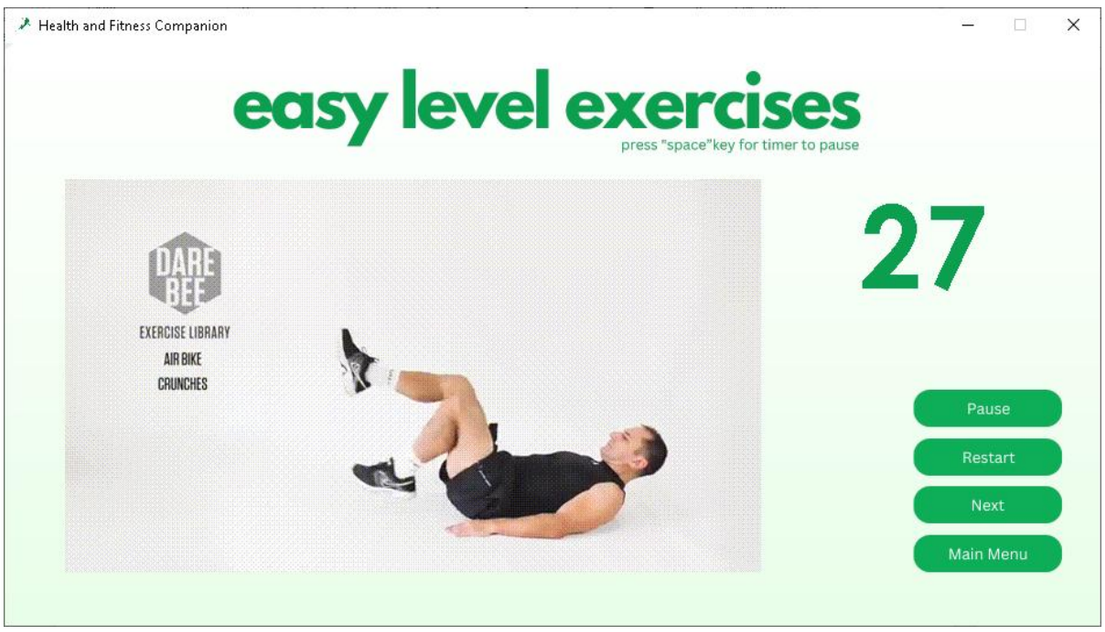
DISCUSSION

The flowchart outlines a user journey through a fitness and wellness application. It begins at the homepage and branches into several features, including "Exercise Now," "BMI Tracker," "Meal Planner," "Alarm Clock," and "Discover." Users can select the difficulty level for exercises, input BMI details, add and plan meals, set alarms, and watch exercise tutorials. Each feature leads to specific actions or displays, culminating in various end points marked by "H" (possibly denoting home or another section) or "END." This organized structure provides a clear, step-by-step guide to using the app's diverse functionalities.

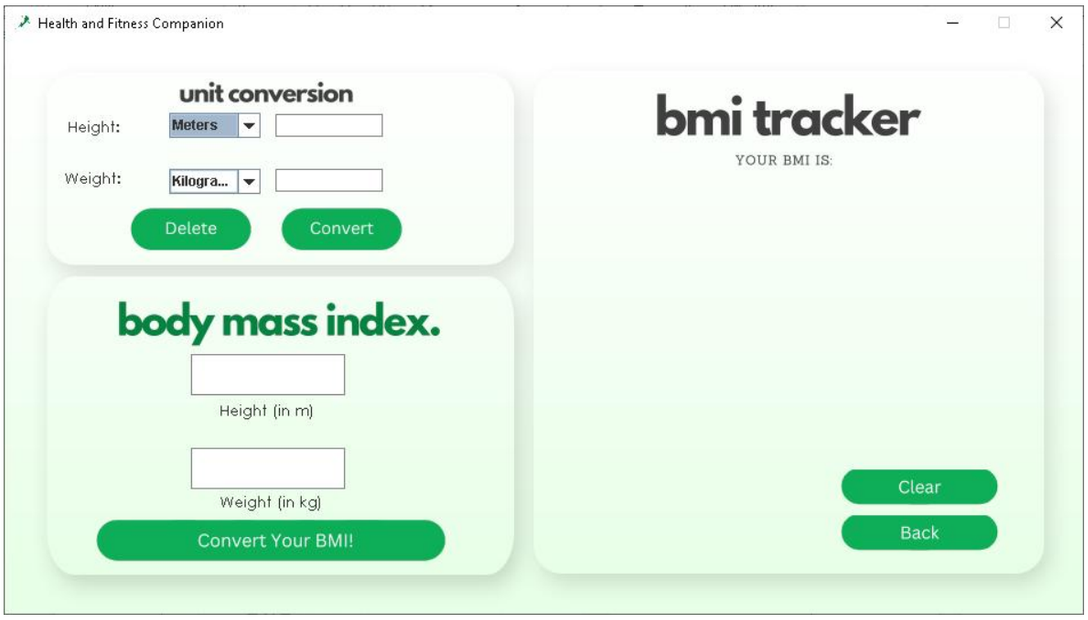
IV. SOFTWARE LAYOUT DESIGN

FEATURE 1

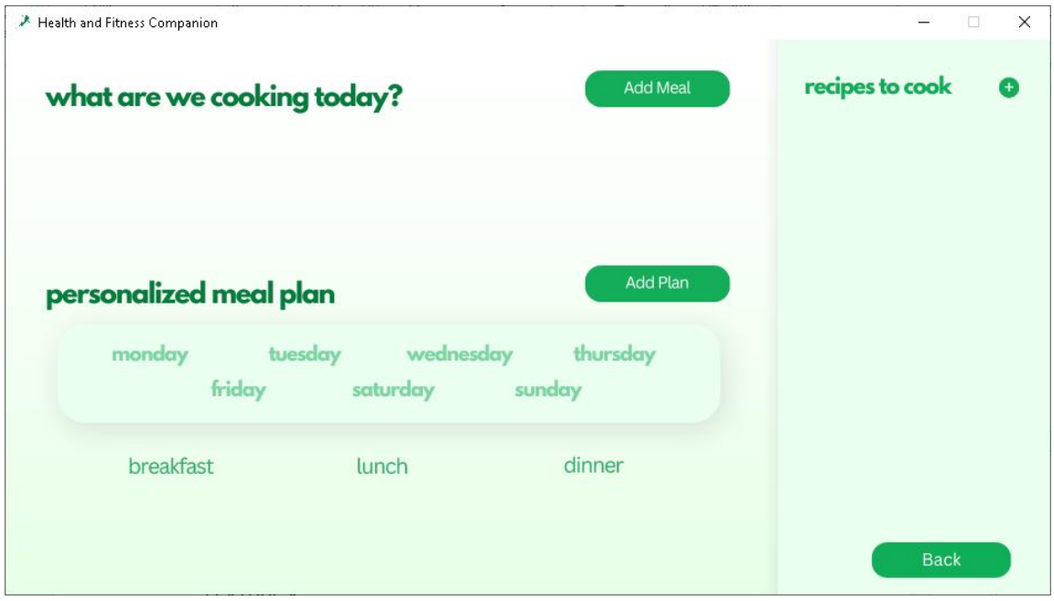




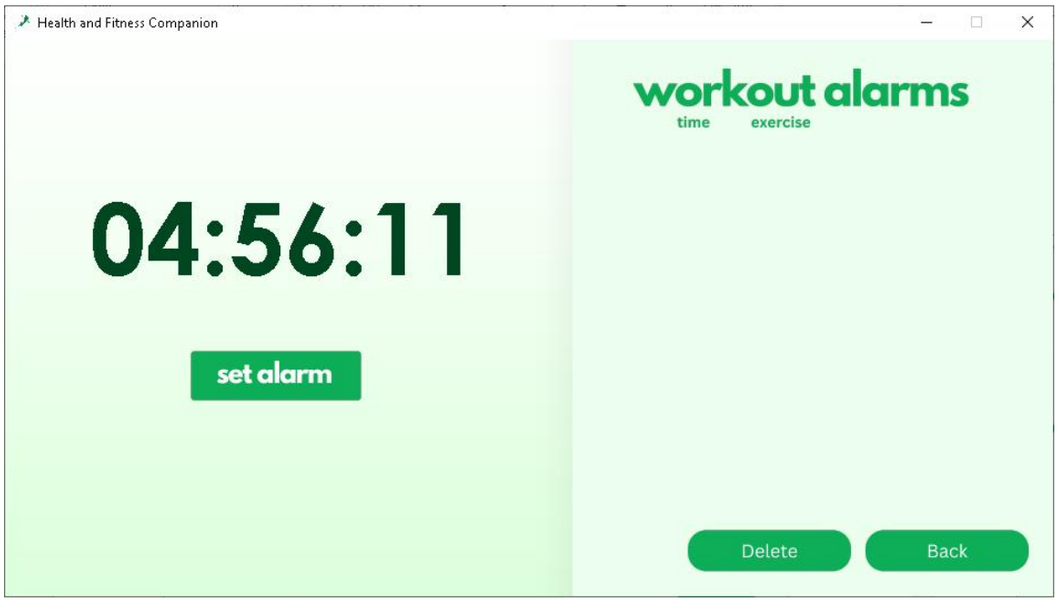
FEATURE 2



FEATURE 3



FEATURE 4



FEATURE 5

V. SUMMARY OF CODES

//ALARM_CLOCK_UI

```
package application;


import java.awt.EventQueue;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.time.LocalDateTime;
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;
import javax.swing.DefaultListCellRenderer;
import javax.swing.DefaultListModel;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JList;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.ListSelectionModel;
import javax.swing.SwingConstants;
import javax.swing.Timer;
import javax.swing.border.EmptyBorder;
import java.awt.Color;
import java.awt.Component;


public class ALARM_CLOCK_UI extends JFrame {
```

```
    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JLabel CLOCK_TIMER_LBL;
    private JList<String> alarmList;
    private DefaultListModel<String> listModel;
    private JLabel CLOCK_ICON;
    private String alarmEntry;


    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    LANDING_SIMPLIFIED_UI frame = new
LANDING_SIMPLIFIED_UI();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }


    /**
     * Create the frame.
     */
    public ALARM_CLOCK_UI() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 960, 540);
        contentPane = new JPanel();
        contentPane.setLayout(null);
        setTitle("Health and Fitness Companion");
```

```
        setResizable(false);

        ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
        setIconImage(mainIcon.getImage());

        ImageIcon workoutAlarm = new ImageIcon(getClass().getClassLoader().getResource("workout-alarm-3.png"));
        JLabel WORKOUT_ALARM = new JLabel(workoutAlarm);
        WORKOUT_ALARM.setBounds(543, 11, 366, 83);
        contentPane.add(WORKOUT_ALARM);

        // Initialize the list model
        listModel = new DefaultListModel<>();

        // Initialize the JList with the list model
        alarmList = new JList<>(listModel);
        alarmList.setBackground(new Color(236, 255, 239));
        alarmList.setOpaque(true);
        alarmList.setFont(new Font("Century Gothic", Font.PLAIN, 20));
        alarmList.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
        alarmList.setBorder(new EmptyBorder(0, 0, 0, 0));

        // Set cell renderer to wrap text
        alarmList.setCellRenderer(new MultilineCellRenderer());

        // Create a JScrollPane and add the JList to it
        JScrollPane scrollPane = new JScrollPane(alarmList);
        scrollPane.setBorder(new EmptyBorder(0, 0, 0, 0));
        scrollPane.setBounds(568, 115, 300, 200);
        contentPane.add(scrollPane);

        ImageIcon alarmIcon = new ImageIcon(getClass().getClassLoader().getResource("set-alarm.png"));
        JButton ALARM_BTN = new JButton(alarmIcon);
```

```
        ALARM_BTN.setBorder(null);
        ALARM_BTN.setOpaque(false);
        ALARM_BTN.setContentAreaFilled(false);
        ALARM_BTN.setBorderPainted(false);
        ALARM_BTN.setSize(alarmIcon.getIconWidth(), alarmIcon.getIconHeight());
        ALARM_BTN.setBounds(162, 275, 164, 55);
        contentPane.add(ALARM_BTN);
        ALARM_BTN.addActionListener(ActionListener -> {
          setAlarm();
        });

        ALARM_BTN.addMouseListener(new MouseAdapter() {
            public void mouseEntered(MouseEvent e) {
                // Change the opacity of the button icon when clicked
                ImageIcon clickedIcon = new
ImageIcon(getClass().getClassLoader().getResource("set-alarm-clicked.png"));
                ALARM_BTN.setIcon(clickedIcon);
            }

            public void mouseExited(MouseEvent e) {
                // Reset to normal icon when mouse exits
                    ALARM_BTN.setIcon(alarmIcon);
            }
        });

        CLOCK_TIMER_LBL = new JLabel("00:00:00");
        CLOCK_TIMER_LBL.setHorizontalAlignment(SwingConstants.CENTER);
        CLOCK_TIMER_LBL.setFont(new Font("Century Gothic", Font.BOLD, 90));
        CLOCK_TIMER_LBL.setForeground(Color.decode("#004721"));
        CLOCK_TIMER_LBL.setBounds(-14, 88, 533, 176);
        contentPane.add(CLOCK_TIMER_LBL);
        clock();
```

```
        ImageIcon backIcon = new ImageIcon(getClass().getClassLoader().getResource("back-alarm.png"));
        JButton BACK_BTN = new JButton(backIcon);
        BACK_BTN.setBounds(775, 440, 147, 40);
        BACK_BTN.setBorder(null);
        BACK_BTN.setOpaque(false);
        BACK_BTN.setContentAreaFilled(false);
        BACK_BTN.setBorderPainted(false);
        contentPane.add(BACK_BTN);
        BACK_BTN.addActionListener(ActionListener -> {
          LANDING_SIMPLIFIED_UI frame = new LANDING_SIMPLIFIED_UI();
           frame.setVisible(true);
          dispose();
        });
        BACK_BTN.addMouseListener(new MouseAdapter() {
           public void mouseEntered(MouseEvent e) {
              ImageIcon clickedIcon = new ImageIcon(getClass().getClassLoader().getResource("back-alarm-clicked.png"));
              BACK_BTN.setIcon(clickedIcon);
           }
           public void mouseExited(MouseEvent e) {
                 BACK_BTN.setIcon(backIcon);
           }
        });


        int topMargin = WORKOUT_ALARM.getY() + WORKOUT_ALARM.getHeight() + 10; // 10 pixels spacing
        int bottomMargin = BACK_BTN.getY() - 20; // 20 pixels spacing
        int listHeight = bottomMargin - topMargin;
        int listWidth = scrollPane.getWidth() + 40; // Extend 20 pixels to the right


        scrollPane.setBounds(568, topMargin, listWidth, listHeight);
        contentPane.add(scrollPane);
```

```
        ImageIcon clearIcon = new ImageIcon(getClass().getClassLoader().getResource("delete-alarm.png"));
        JButton CLEAR_BTN = new JButton(clearIcon);
        CLEAR_BTN.setBounds(615, 440, 147, 40);
        CLEAR_BTN.setBorder(null);
        CLEAR_BTN.setOpaque(false);
        CLEAR_BTN.setContentAreaFilled(false);
        CLEAR_BTN.setBorderPainted(false);
        contentPane.add(CLEAR_BTN);
        CLEAR_BTN.addActionListener(ActionListener->{
          listModel.removeAllElements();
        });
        CLEAR_BTN.addMouseListener(new MouseAdapter() {
            public void mouseEntered(MouseEvent e) {
                ImageIcon clickedIcon = new ImageIcon(getClass().getClassLoader().getResource("delete-alarm-clicked.png"));
                CLEAR_BTN.setIcon(clickedIcon);
            }
            public void mouseExited(MouseEvent e) {
                    CLEAR_BTN.setIcon(clearIcon);
            }
        });

        ImageIcon backgroundImageIcon = new ImageIcon(getClass().getClassLoader().getResource("ALARM-BG-3.png"));
        JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
        backgroundImageLabel.setBounds(0, 0, 960, 540);
        contentPane.add(backgroundImageLabel);

        contentPane.setOpaque(false);
        setContentPane(contentPane);
    }

    public void clock() {
```

```
        DateTimeFormatter dtf = DateTimeFormatter.ofPattern("HH:mm:ss");

        ActionListener updateClockAction = new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                LocalDateTime now = LocalDateTime.now();
                CLOCK_TIMER_LBL.setText(dtf.format(now));
                checkAlarm(now);
            }
        };

        Timer timer = new Timer(1000, updateClockAction);
        timer.start();
    }

    private void checkAlarm(LocalDateTime now) {
        for (int i = 0; i < listModel.getSize(); i++) {
            String alarmEntry = listModel.getElementAt(i);
            String[] parts = alarmEntry.split(" - ");
            String alarmTimeString = parts[0];
            LocalTime alarmTime = LocalTime.parse(alarmTimeString);

            int currentHour = now.getHour();
            int currentMinute = now.getMinute();
            int currentSecond = now.getSecond();

            int alarmHour = alarmTime.getHour();
            int alarmMinute = alarmTime.getMinute();
            int alarmSecond = alarmTime.getSecond();

            if (currentHour == alarmHour && currentMinute == alarmMinute &&
currentSecond == alarmSecond) {
                JOptionPane.showMessageDialog(contentPane, "Alarm! Time is up for
" + alarmEntry);
```

```
            listModel.removeElementAt(i);
            break;
        }
    }
}

private void setAlarm() {
    String[] hours = new String[24];
    String[] minutes = new String[60];
    String[] seconds = new String[60];

    for (int i = 0; i < 24; i++) {
        hours[i] = String.format("%02d", i);
    }

    for (int i = 0; i < 60; i++) {
        minutes[i] = String.format("%02d", i);
    }

    for (int i = 0; i < 60; i++) {
        seconds[i] = String.format("%02d", i);
    }

    JComboBox<String> hourComboBox = new JComboBox<>(hours);
    JComboBox<String> minuteComboBox = new JComboBox<>(minutes);
    JComboBox<String> secondComboBox = new JComboBox<>(seconds);

    JPanel panel = new JPanel();
    panel.add(new JLabel("Hour:"));
    panel.add(hourComboBox);
    panel.add(new JLabel("Minute:"));
    panel.add(minuteComboBox);
    panel.add(new JLabel("Second:"));
```

```
        panel.add(secondComboBox);


        int result = JOptionPane.showConfirmDialog(contentPane, panel, "Set Alarm Time", JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);


        if (result == JOptionPane.OK_OPTION) {
            String selectedHour = (String) hourComboBox.getSelectedItem();
            String selectedMinute = (String) minuteComboBox.getSelectedItem();
            String selectedSecond = (String) secondComboBox.getSelectedItem();


            String alarmTime = selectedHour + ":" + selectedMinute + ":" + selectedSecond;
            String alarmTitle = JOptionPane.showInputDialog(contentPane, "Enter alarm title:");


            if (alarmTitle == null || alarmTitle.trim().isEmpty()) {
                alarmTitle = "";
            }


            alarmEntry = alarmTime + " - " + alarmTitle;
            listModel.addElement(alarmEntry);
            JOptionPane.showMessageDialog(contentPane, "Alarm set for " + alarmEntry);
        }
    }




    public class MultilineCellRenderer extends DefaultListCellRenderer {
        @Override
        public Component getListCellRendererComponent(JList<?> list, Object value, int index, boolean isSelected, boolean cellHasFocus) {
            Component c = super.getListCellRendererComponent(list, value, index, isSelected, cellHasFocus);
            if (c instanceof JLabel) {
                JLabel label = (JLabel) c;
```

```
                label.setText("<html>" + value.toString().replace("\n", "<br>") +
        "</html>");
                }
                return c;
            }
        }
    }
```

//BMI_FEATURE_UI

```
package application;

import java.awt.Dimension;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.Image;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

import javax.swing.BorderFactory;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.SwingConstants;
import javax.swing.border.EmptyBorder;
import javax.swing.border.LineBorder;
import javax.swing.JComboBox;
import java.awt.Color;
```

```
import java.awt.Cursor;

public class BMI_FEATURE_UI extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTextField HEIGHT_TEXTFIELD;
    private JTextField WEIGHT_TEXTFIELD;
    private JLabel BMI_COUNTER_LBL;
    private JLabel BMI_LABELER_LBL_1;
    private JTextField HEIGHT_CONVERSION_TEXTFIELD;
    private JTextField WEIGHT_CONVERSION_TEXTFIELD;

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                                        LANDING_SIMPLIFIED_UI landingSimplified = new
LANDING_SIMPLIFIED_UI();
                                        landingSimplified.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the frame.
     */
    public BMI_FEATURE_UI() {
```

```
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 960, 540);
        contentPane = new JPanel();
        contentPane.setBackground(new Color(235, 235, 235));
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);
        setTitle("Health and Fitness Companion");
        setResizable(false);

        ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
        setIconImage(mainIcon.getImage());

        JComboBox HEIGHT_COMBOBOX = new JComboBox<>();
        HEIGHT_COMBOBOX.setBounds(144, 63, 80, 22);
        HEIGHT_COMBOBOX.setBorder(BorderFactory.createEmptyBorder());
        // Set background color to white
        HEIGHT_COMBOBOX.setBackground(Color.WHITE);
        contentPane.add(HEIGHT_COMBOBOX);
        // Adding values to the combo box
        HEIGHT_COMBOBOX.addItem("Meters");
        HEIGHT_COMBOBOX.addItem("Feet");
        HEIGHT_COMBOBOX.addItem("Centimeters");
        HEIGHT_COMBOBOX.addItem("Inches");

        ImageIcon convertIcon = new ImageIcon(getClass().getClassLoader().getResource("convert-icon.png")); // Load the image for the button
        JButton CONVERSION_CONVERT_BTN = new JButton(convertIcon); // Create the delete button
        CONVERSION_CONVERT_BTN.setBorder(null); // Remove the default border of the button
        CONVERSION_CONVERT_BTN.setOpaque(false); // Remove Background
        CONVERSION_CONVERT_BTN.setContentAreaFilled(false);
        CONVERSION_CONVERT_BTN.setBorderPainted(false);
```

```
CONVERSION_CONVERT_BTN.setSize(convertIcon.getIconWidth(), convertIcon.getIconHeight()); // Set the size of the button to match the size of the icon

CONVERSION_CONVERT_BTN.setLocation(234, 145); // Set the position of the button

contentPane.add(CONVERSION_CONVERT_BTN); // Add the button to the content pane


CONVERSION_CONVERT_BTN.addMouseListener(new MouseAdapter() {

    public void mouseEntered(MouseEvent e) {

        // Change the opacity of the button icon when clicked

        ImageIcon clickedIcon = new ImageIcon(getClass().getClassLoader().getResource("convert-icon-clicked.png"));

        CONVERSION_CONVERT_BTN.setIcon(clickedIcon);

    }


    public void mouseExited(MouseEvent e) {

        // Reset to normal icon when mouse exits

        CONVERSION_CONVERT_BTN.setIcon(convertIcon);

    }

});


ImageIcon deleteIcon = new ImageIcon(getClass().getClassLoader().getResource("delete-icon.png")); // Load the image for the button

JButton DELETE_CONVERT_BTN = new JButton(deleteIcon); // Create the delete button

DELETE_CONVERT_BTN.setBorder(null); // Remove the default border of the button

DELETE_CONVERT_BTN.setOpaque(false); // Remove Background

DELETE_CONVERT_BTN.setContentAreaFilled(false);

DELETE_CONVERT_BTN.setBorderPainted(false);

DELETE_CONVERT_BTN.setSize(deleteIcon.getIconWidth(), deleteIcon.getIconHeight()); // Set the size of the button to match the size of the icon

DELETE_CONVERT_BTN.setLocation(102, 145); // Set the position of the button

contentPane.add(DELETE_CONVERT_BTN); // Add the button to the content pane


DELETE_CONVERT_BTN.addMouseListener(new MouseAdapter() {
```

```
            public void mouseEntered(MouseEvent e) {
                // Change the opacity of the button icon when clicked
                ImageIcon clickedIcon = new ImageIcon(getClass().getClassLoader().getResource("delete-icon-clicked.png"));
                DELETE_CONVERT_BTN.setIcon(clickedIcon);
            }

            public void mouseExited(MouseEvent e) {
                // Reset to normal icon when mouse exits
                    DELETE_CONVERT_BTN.setIcon(deleteIcon);
            }
        });

        JComboBox WEIGHT_COMBOBOX = new JComboBox<>();
        WEIGHT_COMBOBOX.setBounds(144, 112, 80, 22);
        contentPane.add(WEIGHT_COMBOBOX);
        WEIGHT_COMBOBOX.setBorder(BorderFactory.createEmptyBorder());
        // Set background color to white
        WEIGHT_COMBOBOX.setBackground(Color.WHITE);
        // Adding values to the combo box
        WEIGHT_COMBOBOX.addItem("Kilograms");
        WEIGHT_COMBOBOX.addItem("Pounds");
        WEIGHT_COMBOBOX.addItem("Tons");
        WEIGHT_COMBOBOX.addItem("Grams");

        JLabel WEIGHT_CONVERSION_LBL = new JLabel("Weight: ");
        WEIGHT_CONVERSION_LBL.setHorizontalAlignment(SwingConstants.CENTER);
        WEIGHT_CONVERSION_LBL.setFont(new Font("Century Gothic", Font.PLAIN, 13));
        WEIGHT_CONVERSION_LBL.setBounds(27, 99, 105, 40);
        contentPane.add(WEIGHT_CONVERSION_LBL);

        JLabel HEIGHT_CONVERSION_LBL = new JLabel("Height:");
        HEIGHT_CONVERSION_LBL.setHorizontalAlignment(SwingConstants.CENTER);
```

```
        HEIGHT_CONVERSION_LBL.setFont(new Font("Century Gothic", Font.PLAIN, 13));
        HEIGHT_CONVERSION_LBL.setBounds(26, 54, 105, 40);
        contentPane.add(HEIGHT_CONVERSION_LBL);

        // Load the image
        ImageIcon unitConversionIcon = new ImageIcon(getClass().getClassLoader().getResource("unit-conversion.png"));
        // Create a JLabel with the image icon
        JLabel CONVERSION_LABEL = new JLabel(unitConversionIcon);
        // Set the size of the JLabel to match the size of the image
        CONVERSION_LABEL.setBounds(144, 28, 170, 35);
        // Add the JLabel to the content pane
        contentPane.add(CONVERSION_LABEL);

        HEIGHT_TEXTFIELD = new JTextField();
        HEIGHT_TEXTFIELD.setToolTipText("");
        HEIGHT_TEXTFIELD.setHorizontalAlignment(SwingConstants.CENTER);
        HEIGHT_TEXTFIELD.setFont(new Font("Century Gothic", Font.BOLD, 22));
        HEIGHT_TEXTFIELD.setBounds(163, 274, 135, 36);
        HEIGHT_TEXTFIELD.setBorder(new LineBorder(Color.GRAY, 1));
        HEIGHT_TEXTFIELD.setColumns(10);
        contentPane.add(HEIGHT_TEXTFIELD);

        WEIGHT_TEXTFIELD = new JTextField();
        WEIGHT_TEXTFIELD.setToolTipText("");
        WEIGHT_TEXTFIELD.setHorizontalAlignment(SwingConstants.CENTER);
        WEIGHT_TEXTFIELD.setFont(new Font("Century Gothic", Font.BOLD, 22));
        WEIGHT_TEXTFIELD.setColumns(10);
        WEIGHT_TEXTFIELD.setBounds(163, 356, 135, 36);
        WEIGHT_TEXTFIELD.setBorder(new LineBorder(Color.GRAY, 1));
        contentPane.add(WEIGHT_TEXTFIELD);

        JLabel HEIGHT_LBL = new JLabel("Height (in m)");
```

```
HEIGHT_LBL.setFont(new Font("Century Gothic", Font.PLAIN, 13));
HEIGHT_LBL.setBounds(188, 311, 94, 22);
contentPane.add(HEIGHT_LBL);

JLabel WEIGHT_LBL_1 = new JLabel("Weight (in kg)");
WEIGHT_LBL_1.setFont(new Font("Century Gothic", Font.PLAIN, 13));
WEIGHT_LBL_1.setBounds(188, 391, 94, 22);
contentPane.add(WEIGHT_LBL_1);

ImageIcon convertBMIIcon = new ImageIcon(getClass().getClassLoader().getResource("convert-bmi-2.png")); // Load the image for the button
JButton CONVERT_BTN = new JButton(convertBMIIcon); // Create the delete button
CONVERT_BTN.setBorder(null); // Remove the default border of the button
CONVERT_BTN.setOpaque(false); // Remove Background
CONVERT_BTN.setContentAreaFilled(false);
CONVERT_BTN.setBorderPainted(false);
CONVERT_BTN.setSize(convertBMIIcon.getIconWidth(), convertBMIIcon.getIconHeight()); // Set the size of the button to match the size of the icon
CONVERT_BTN.setBounds(56, 418, 377, 38); // Set the position of the button
contentPane.add(CONVERT_BTN); // Add the button to the content pane

CONVERT_BTN.addMouseListener(new MouseAdapter() {
    public void mouseEntered(MouseEvent e) {
        // Change the opacity of the button icon when clicked
        ImageIcon clickedIcon = new ImageIcon(getClass().getClassLoader().getResource("convert-bmi-2-clicked.png"));
        CONVERT_BTN.setIcon(clickedIcon);
    }

    public void mouseExited(MouseEvent e) {
        // Reset to normal icon when mouse exits
            CONVERT_BTN.setIcon(convertBMIIcon);
    }
```

```
        });


        // Load the image
        ImageIcon iconBMI = new ImageIcon(getClass().getClassLoader().getResource("bmi-logo.png"));
        // Create a JLabel with the image icon
        JLabel BMI_LBL = new JLabel(iconBMI);
        // Set the size of the JLabel to match the size of the image
        BMI_LBL.setBounds(553, 42, 264, 43);
        // Add the JLabel to the content pane
        contentPane.add(BMI_LBL);


        // Load the image
        ImageIcon iconBMIIs = new ImageIcon(getClass().getClassLoader().getResource("bmi-is.png"));
        // Create a JLabel with the image icon
        JLabel BMI_LOWER_LBL = new JLabel(iconBMIIs);
        // Set the size of the JLabel to match the size of the image
        BMI_LOWER_LBL.setBounds(635, 94, 94, 22);
        // Add the JLabel to the content pane
        contentPane.add(BMI_LOWER_LBL);


        BMI_COUNTER_LBL = new JLabel("");
        BMI_COUNTER_LBL.setHorizontalAlignment(SwingConstants.CENTER);
        BMI_COUNTER_LBL.setFont(new Font("Century Gothic", Font.BOLD, 45));
        BMI_COUNTER_LBL.setBounds(624, 129, 122, 71);
        contentPane.add(BMI_COUNTER_LBL);


        BMI_LABELER_LBL_1 = new JLabel("");
        BMI_LABELER_LBL_1.setHorizontalAlignment(SwingConstants.CENTER);
        BMI_LABELER_LBL_1.setFont(new Font("Century Gothic", Font.PLAIN, 23));
        BMI_LABELER_LBL_1.setBounds(590, 186, 190, 50);
        contentPane.add(BMI_LABELER_LBL_1);
```

```
JLabel ULTIMATE_LABEL_LBL = new JLabel(" ");
ULTIMATE_LABEL_LBL.setHorizontalAlignment(SwingConstants.RIGHT);
ULTIMATE_LABEL_LBL.setFont(new Font("Century Gothic", Font.PLAIN, 14));
ULTIMATE_LABEL_LBL.setBounds(492, 233, 401, 124);
contentPane.add(ULTIMATE_LABEL_LBL);

ImageIcon clearIcon = new ImageIcon(getClass().getClassLoader().getResource("clear.png")); // Load the image for the button
JButton CLEAR_BTN = new JButton(clearIcon); // Create the delete button
CLEAR_BTN.setBorder(null); // Remove the default border of the button
CLEAR_BTN.setOpaque(false); // Remove Background
CLEAR_BTN.setContentAreaFilled(false);
CLEAR_BTN.setBorderPainted(false);
CLEAR_BTN.setSize(convertBMIIcon.getIconWidth(), convertBMIIcon.getIconHeight()); // Set the size of the button to match the size of the icon
CLEAR_BTN.setBounds(716, 375, 170, 31);; // Set the position of the button
contentPane.add(CLEAR_BTN); // Add the button to the content pane

CLEAR_BTN.addMouseListener(new MouseAdapter() {
    public void mouseEntered(MouseEvent e) {
        // Change the opacity of the button icon when clicked
        ImageIcon clickedIcon = new ImageIcon(getClass().getClassLoader().getResource("clear-pressed.png"));
        CLEAR_BTN.setIcon(clickedIcon);
    }

    public void mouseExited(MouseEvent e) {
        // Reset to normal icon when mouse exits
            CLEAR_BTN.setIcon(clearIcon);
    }
});

ImageIcon backIcon = new ImageIcon(getClass().getClassLoader().getResource("back.png")); // Load the image for the button
```

```
        JButton BACK_BTN = new JButton(backIcon); // Create the delete button

        BACK_BTN.setBorder(null); // Remove the default border of the button

        BACK_BTN.setOpaque(false); // Remove Background

        BACK_BTN.setContentAreaFilled(false);

        BACK_BTN.setBorderPainted(false);

        BACK_BTN.setSize(convertBMIIcon.getIconWidth(),
convertBMIIcon.getIconHeight()); // Set the size of the button to match the size of the
icon

        BACK_BTN.setBounds(716, 415, 170, 31); // Set the position of the button

        contentPane.add(BACK_BTN); // Add the button to the content pane



        BACK_BTN.addMouseListener(new MouseAdapter() {

            public void mouseEntered(MouseEvent e) {

                // Change the opacity of the button icon when clicked

                ImageIcon clickedIcon = new
ImageIcon(getClass().getClassLoader().getResource("back-pressed.png"));

                BACK_BTN.setIcon(clickedIcon);

            }



            public void mouseExited(MouseEvent e) {

                // Reset to normal icon when mouse exits

                    BACK_BTN.setIcon(backIcon);

            }

        });



        HEIGHT_CONVERSION_TEXTFIELD = new JTextField();


HEIGHT_CONVERSION_TEXTFIELD.setHorizontalAlignment(SwingConstants.CENTE
R);

        HEIGHT_CONVERSION_TEXTFIELD.setBounds(237, 64, 94, 20);

        HEIGHT_CONVERSION_TEXTFIELD.setColumns(10);

        HEIGHT_CONVERSION_TEXTFIELD.setBorder(new  LineBorder(Color.gray,  1));
// Adjust color and thickness as needed

        contentPane.add(HEIGHT_CONVERSION_TEXTFIELD);



        WEIGHT_CONVERSION_TEXTFIELD = new JTextField();
```

```
WEIGHT_CONVERSION_TEXTFIELD.setHorizontalAlignment(SwingConstants.CENT
ER);
        WEIGHT_CONVERSION_TEXTFIELD.setColumns(10);
        WEIGHT_CONVERSION_TEXTFIELD.setBounds(237, 112, 94, 20);
        WEIGHT_CONVERSION_TEXTFIELD.setBorder(new  LineBorder(Color.gray,  1));
// Adjust color and thickness as needed
        contentPane.add(WEIGHT_CONVERSION_TEXTFIELD);


        JLabel CONVERTED_HEIGHT_LBL = new JLabel("");
        CONVERTED_HEIGHT_LBL.setFont(new  Font("Century  Gothic",  Font.PLAIN,
13));
        CONVERTED_HEIGHT_LBL.setBounds(357, 63, 67, 18);
        contentPane.add(CONVERTED_HEIGHT_LBL);


        JLabel CONVERTED_WEIGHT_LBL = new JLabel("");
        CONVERTED_WEIGHT_LBL.setFont(new  Font("Century  Gothic",  Font.PLAIN,
13));
        CONVERTED_WEIGHT_LBL.setBounds(357, 113, 67, 18);
        contentPane.add(CONVERTED_WEIGHT_LBL);


        // Load the image
        ImageIcon  icon  =  new  ImageIcon(getClass().getClassLoader().getResource("fill-
in.png"));
        // Create a JLabel with the image icon
        JLabel BMI_IMG_LBL = new JLabel(icon);
        // Set the size of the JLabel to match the size of the image
        BMI_IMG_LBL.setBounds(56, 219, 369, 50);
        // Add the JLabel to the content pane
        contentPane.add(BMI_IMG_LBL);


        CONVERT_BTN.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                // Get height and weight input
                String heightStr = HEIGHT_TEXTFIELD.getText().trim();
                String weightStr = WEIGHT_TEXTFIELD.getText().trim();
```

```
// Check if height and weight inputs are not empty
if (!heightStr.isEmpty() && !weightStr.isEmpty()) {
    try {
        // Parse height and weight to double
        double height = Double.parseDouble(heightStr);
        double weight = Double.parseDouble(weightStr);

        // Calculate BMI
        double bmi = calculateBMI(height, weight);

        // Display BMI
        BMI_COUNTER_LBL.setText(String.format("%.1f", bmi));

        // Determine BMI label
        String bmiLabel = determineBMILabel(bmi);

        // Display BMI label
        BMI_LABELER_LBL_1.setText(bmiLabel);

        // Determine BMI description
        String bmiDescription = descriptionBMI(bmi);

        // Display BMI description
        ULTIMATE_LABEL_LBL.setText(bmiDescription);
    } catch (NumberFormatException ex) {
      HEIGHT_TEXTFIELD.setText(" ");
      WEIGHT_TEXTFIELD.setText(" ");
    }
} else {
                HEIGHT_TEXTFIELD.setText(" ");
                WEIGHT_TEXTFIELD.setText(" ");
                BMI_COUNTER_LBL.setText(" ");
```

```
                    BMI_LABELER_LBL_1.setText(" ");
                    ULTIMATE_LABEL_LBL.setText(" ");
                JOptionPane.showMessageDialog(contentPane, "Error Input! Retry", "No Input Integer", JOptionPane.ERROR_MESSAGE);
            }
        }
    });

    CLEAR_BTN.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
            HEIGHT_TEXTFIELD.setText(" ");
            WEIGHT_TEXTFIELD.setText(" ");
            BMI_COUNTER_LBL.setText(" ");
            BMI_LABELER_LBL_1.setText(" ");
            ULTIMATE_LABEL_LBL.setText(" ");

      }
    });

    CONVERSION_CONVERT_BTN.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            // For HEIGHT
            // Get the selected unit
            String selectedUnit = (String) HEIGHT_COMBOBOX.getSelectedItem();
            // Get the value to convert
            String inputValue = HEIGHT_CONVERSION_TEXTFIELD.getText().trim();

            // Check if the input value is not empty
            if (!inputValue.isEmpty()) {
                try {
                    double valueToConvert = Double.parseDouble(inputValue);

                    // Convert to meters based on the selected unit
                    double convertedValue = 0.0; // Initialize to a default value
```

```
                if (selectedUnit.equals("Centimeters")) {
                    convertedValue = valueToConvert / 100; // Convert to meters
                } else if (selectedUnit.equals("Inches")) {
                    convertedValue = valueToConvert / 39.3701; // Convert to meters
                } else if (selectedUnit.equals("Feet")) {
                    convertedValue = valueToConvert * 0.3048; // Convert to meters
                } else if (selectedUnit.equals("Meters")) {
                    convertedValue = valueToConvert;
                }

                // Display the converted value
                CONVERTED_HEIGHT_LBL.setText(String.format("%.2f m", convertedValue));
            } catch (NumberFormatException ex) {
                // Handle the case where the input value is not a valid number
                CONVERTED_HEIGHT_LBL.setText("Invalid input");
            }
        } else {
            // If the input value is empty, display an appropriate message or clear the label
            CONVERTED_HEIGHT_LBL.setText("");
        }

        // For WEIGHT
        // Get the selected unit
        String selectedUnitWeight = (String) WEIGHT_COMBOBOX.getSelectedItem();
        // Get the value to convert
        String inputValueWeight = WEIGHT_CONVERSION_TEXTFIELD.getText().trim();

        // Check if the input value is not empty
        if (!inputValueWeight.isEmpty()) {
            try {
                double valueToConvertWeight = Double.parseDouble(inputValueWeight);
```

```
                // Convert to meters based on the selected unit
                double convertedValueWeight = 0.0;
                if (selectedUnitWeight.equals("Pounds")) {
                    convertedValueWeight = valueToConvertWeight * 0.45359237;
                } else if (selectedUnitWeight.equals("Tons")) {
                    convertedValueWeight = valueToConvertWeight * 1000;
                } else if (selectedUnitWeight.equals("Grams")) {
                    convertedValueWeight = valueToConvertWeight / 1000;
                } else if (selectedUnitWeight.equals("Kilograms")) {
                    convertedValueWeight = valueToConvertWeight;
                }

                // Display the converted value
                CONVERTED_WEIGHT_LBL.setText(String.format("%.2f kg",
convertedValueWeight));
            } catch (NumberFormatException ex) {
                // Handle the case where the input value is not a valid number
                CONVERTED_WEIGHT_LBL.setText("Invalid input");
            }
        } else {
            // If the input value is empty, display an appropriate message or clear the
label
            CONVERTED_WEIGHT_LBL.setText("");
        }
      }
    });

    DELETE_CONVERT_BTN.addActionListener (new ActionListener() {
      public void actionPerformed(ActionEvent e) {
            WEIGHT_CONVERSION_TEXTFIELD.setText(" ");
            HEIGHT_CONVERSION_TEXTFIELD.setText(" ");
            CONVERTED_HEIGHT_LBL.setText(" ");
            CONVERTED_WEIGHT_LBL.setText(" ");
```

```
        }
    });

    BACK_BTN.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
                                LANDING_SIMPLIFIED_UI landingSimplified = new LANDING_SIMPLIFIED_UI();
                                landingSimplified.setVisible(true);
                dispose();
        }
    });

    // Load the image
    ImageIcon backgroundImageIcon = new ImageIcon(getClass().getClassLoader().getResource("BACKGROUND-BMI-12.png"));

    // Create a JLabel with the image icon
    JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);

    // Set the size of the label to match the size of the JFrame
    backgroundImageLabel.setSize(getSize());

    // Set the bounds of the label to cover the whole content pane
    backgroundImageLabel.setBounds(0, 0, getWidth(), getHeight());

    // Add the JLabel to the content pane (at the back, so it doesn't cover other components)
    contentPane.add(backgroundImageLabel, new Integer(Integer.MIN_VALUE));

    // Make sure the content pane is transparent
    ((JPanel) contentPane).setOpaque(false);
  }

  private double calculateBMI(double height, double weight) {
    return weight / (height * height);
```

```
    }


    private String determineBMILabel(double bmi) {
        String text;
       if (bmi < 18.5) {
          text = "Underweight";
          return text;
       } else if(bmi >= 18.5 && bmi <= 24.9) {
          text = "Normal";
          return text;
       } else if (bmi >= 25 && bmi <= 29.9) {
          text = "Overweight";
          return text;
       } else if (bmi >= 30) {
          text = "Obese";
          return text;
       }
       return "";
    }


    private String descriptionBMI(double bmi) {
        return (bmi < 18.5) ? "<html>Individuals with a BMI below the normal range are considered underweight. They may have insufficient body fat, "
                                + "which can indicate potential health risks such as weakened immune function, nutrient deficiencies, and osteoporosis.</html>"
                            : (bmi <= 24.9) ? "<html>Falling within the healthy BMI range indicates a normal weight. Individuals in this category typically have a "
                                + "balanced body composition, which is associated with lower risks of chronic diseases and overall good health.</html>"
                            : (bmi <= 29.9) ? "<html>Those with a BMI above the normal range but below the obese range are classified as overweight. Excess weight "
                                + "can increase the risk of various health conditions, including heart disease, type 2 diabetes, hypertension, and certain cancers.</html>"
                            : "<html>Individuals with a BMI in the obese range have an excessive amount of body fat, which can significantly increase "
```

```
                                                    + "the risk of developing
serious health problems. These may include cardiovascular diseases, stroke, type 2
diabetes. </html>";

    }

}
```

//CREDITS_UI

```
package application;


import java.awt.EventQueue;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;


import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.border.EmptyBorder;


public class CREDITS_UI extends JFrame {


    private static final long serialVersionUID = 1L;

    private JPanel contentPane;


    /**

     * Launch the application.

     */

    public static void main(String[] args) {

        EventQueue.invokeLater(new Runnable() {

            public void run() {

                try {

                    LANDING_SIMPLIFIED_UI frame = new LANDING_SIMPLIFIED_UI();

                   frame.setVisible(true);
```

```
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 */
public CREDITS_UI() {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 960, 540);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPane.setLayout(null);
    setTitle("Health and Fitness Companion");
    setResizable(false);

    // Use getResource to load the image icon
    ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
    setIconImage(mainIcon.getImage());

    JButton BACK_BTN = new JButton();
    BACK_BTN = buttonSetter("back-exercises.png",
            "back-exercises-clicked.png",
            "back-exercises.png");
    BACK_BTN.setBounds(780, 450, 133, 40);
    BACK_BTN.addActionListener(ActionLister -> {
        LANDING_SIMPLIFIED_UI frame = new LANDING_SIMPLIFIED_UI();
        frame.setVisible(true);
        dispose();
    });
```

```
        // Use getResource to load the background image
        ImageIcon backgroundImageIcon = new ImageIcon(getClass().getClassLoader().getResource("credits-ui.png"));
        JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
        backgroundImageLabel.setBounds(0, 0, 960, 540);
        contentPane.add(backgroundImageLabel);


        setContentPane(contentPane);
    }


    public JButton buttonSetter(String imageLocation, String enteredIconLocation, String exitedIconLocation) {
        ImageIcon icon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));
        JButton button = new JButton(icon);
        button.setBorder(null);
        button.setOpaque(false);
        button.setContentAreaFilled(false);
        button.setBorderPainted(false);
        button.setSize(icon.getIconWidth(), icon.getIconHeight());
        button.addMouseListener(getMouseListener(enteredIconLocation, exitedIconLocation));
        contentPane.add(button);
        return button;
    }


    private MouseAdapter getMouseListener(String enteredIconLocation, String exitedIconLocation) {
        return new MouseAdapter() {
            public void mouseEntered(MouseEvent e) {
                ImageIcon enteredIcon = new ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));
                ((JButton) e.getSource()).setIcon(enteredIcon);
            }
```

```
            public void mouseExited(MouseEvent e) {
                ImageIcon exitedIcon = new ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));
                ((JButton) e.getSource()).setIcon(exitedIcon);
            }
        };
    }
}
```

//EXERCISE_DIFFICULTY_UI

```
package application;

import java.awt.EventQueue;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.SwingConstants;
import javax.swing.JButton;

public class EXERCISE_DIFFICULTY_UI extends JFrame {

	private static final long serialVersionUID = 1L;
	private JPanel contentPane;

	/**
	 * Launch the application.
	 */
	public static void main(String[] args) {
		EventQueue.invokeLater(new Runnable() {
```

```
                public void run() {
                        try {
                                LANDING_SIMPLIFIED_UI landingSimplified = new LANDING_SIMPLIFIED_UI();
                                landingSimplified.setVisible(true);
                        } catch (Exception e) {
                                e.printStackTrace();
                        }
                }
        });
    }

    /**
     * Create the frame.
     */
    public EXERCISE_DIFFICULTY_UI() {
            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            setBounds(100, 100, 960, 540);
            contentPane = new JPanel();
            contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
            contentPane.setLayout(null);
    setTitle("Health and Fitness Companion");
    setResizable(false);

    ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
    setIconImage(mainIcon.getImage());

    JButton EASY_BTN = buttonSetter(
            "easy.png",
            "easy-clicked.png",
            "easy.png");
    EASY_BTN.setBounds(382, 163, 168, 45);
    EASY_BTN.addActionListener(e -> {
```

```
    EXERCISE_EASY_UI easy = new EXERCISE_EASY_UI();
    easy.setVisible(true);
    dispose();
});

JButton MEDIUM_BTN = buttonSetter(
            "medium.png",
            "medium-clicked.png",
            "medium.png");
MEDIUM_BTN.setBounds(382, 227, 168, 45);
MEDIUM_BTN.addActionListener(ActionListener -> {
    EXERCISE_MEDIUM_UI medium = new EXERCISE_MEDIUM_UI();
    medium.setVisible(true);
    dispose();
});

JButton HARD_BTN = buttonSetter(
            "hard.png",
            "hard-clicked.png",
            "hard.png");
HARD_BTN.setBounds(382, 290, 168, 45);
HARD_BTN.addActionListener(ActionListener -> {
    EXERCISE_HARD_UI hard = new EXERCISE_HARD_UI();
    hard.setVisible(true);
    dispose();
});

JButton BACK_BTN = new JButton();
BACK_BTN = buttonSetter("back-exercises.png",
                        "back-exercises-clicked.png",
                        "back-exercises.png");
BACK_BTN.setBounds(770, 430, 133, 40);
BACK_BTN.addActionListener(ActionListener -> {
```

```
                    LANDING_SIMPLIFIED_UI landingSimplified = new
LANDING_SIMPLIFIED_UI();
                    landingSimplified.setVisible(true);
      dispose();
    });

    JLabel DIFFICULTY_LBL = imageSetter("choose-difficulty.png");
    DIFFICULTY_LBL.setBounds(90, 60, 748, 75);

    ImageIcon backgroundImageIcon = new
ImageIcon(getClass().getClassLoader().getResource("LIBRARY-PICS.png"));
    JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
    backgroundImageLabel.setBounds(0, 0, 960, 540);
    contentPane.add(backgroundImageLabel);

        setContentPane(contentPane);
      }

      public JLabel imageSetter(String imageLocation) {
    ImageIcon imageIcon = new
ImageIcon(getClass().getClassLoader().getResource(imageLocation));
    JLabel label = new JLabel(imageIcon);
    contentPane.add(label);
    return label;
  }

  public JButton buttonSetter(String imageLocation, String enteredIconLocation, String
exitedIconLocation) {
    ImageIcon icon = new
ImageIcon(getClass().getClassLoader().getResource(imageLocation));
    JButton button = new JButton(icon);
    button.setBorder(null);
    button.setOpaque(false);
    button.setContentAreaFilled(false);
    button.setBorderPainted(false);
    button.setSize(icon.getIconWidth(), icon.getIconHeight());
```

```
        button.addMouseListener(getMouseListener(enteredIconLocation,
exitedIconLocation));

        contentPane.add(button);

        return button;

    }


    private MouseAdapter getMouseListener(String enteredIconLocation, String
exitedIconLocation) {

        return new MouseAdapter() {

            public void mouseEntered(MouseEvent e) {

                ImageIcon enteredIcon = new
ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));

                ((JButton) e.getSource()).setIcon(enteredIcon);

            }



            public void mouseExited(MouseEvent e) {

                ImageIcon exitedIcon = new
ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));

                ((JButton) e.getSource()).setIcon(exitedIcon);

            }

        };

    }

}
```

//EXERCISE_EASY_UI

```
package application;


import java.awt.Color;

import java.awt.EventQueue;

import javax.swing.ImageIcon;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.border.EmptyBorder;
```

```
import javax.swing.SwingConstants;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.Timer;
import java.util.ArrayList;
import java.util.List;

public class EXERCISE_EASY_UI extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JLabel TIMER_LBL;
    private JLabel GIF_LBL;
    private Timer timer;
    private int timeRemaining = 30;
    private boolean isPaused = false;
    private List<String> exercises;
    private int currentExerciseIndex = 0;
    private JDialog restDialog = null; // Keep track of the rest dialog

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
```

```
                                    LANDING_SIMPLIFIED_UI landingSimplified = new
LANDING_SIMPLIFIED_UI();
                                    landingSimplified.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 */
public EXERCISE_EASY_UI() {
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 960, 540);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    contentPane.setLayout(null);
    setTitle("Health and Fitness Companion");
    setResizable(false);

    ImageIcon mainIcon = new
ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
    setIconImage(mainIcon.getImage());

    TIMER_LBL = new JLabel("30");
    TIMER_LBL.setFont(new Font("Century Gothic", Font.BOLD, 99));
    TIMER_LBL.setHorizontalAlignment(SwingConstants.CENTER);
    TIMER_LBL.setBounds(689, 110, 206, 124);
    TIMER_LBL.setForeground(Color.decode("#0B9F4F")); // Set text color
    contentPane.add(TIMER_LBL);

    JLabel TITLE_LBL = imageSetter("easy-level.png");
```

```
        TITLE_LBL.setBounds(72, 15, 791, 84);


        GIF_LBL = new JLabel(new ImageIcon(getClass().getClassLoader().getResource("AIR-CRUNCH-ANIMATION.gif")));
        GIF_LBL.setBounds(52, 117, 600, 338);
        contentPane.add(GIF_LBL);


        JButton MENU_BTN = buttonSetter(
                "main-menu-exercise.png",
                "main-menu-exercise-clicked.png",
                "main-menu-exercise.png");
        MENU_BTN.setBounds(782, 419, 130, 41);
        MENU_BTN.addActionListener(e -> {
            // Stop the timer and dispose of any open dialogs
            if (timer != null) {
                timer.stop();
            }
            if (restDialog != null) {
                restDialog.dispose();
            }
                        LANDING_SIMPLIFIED_UI landingSimplified = new LANDING_SIMPLIFIED_UI();
                        landingSimplified.setVisible(true);
            dispose();
        });


        JButton NEXT_BTN = buttonSetter(
                "next-exercise.png",
                "next-exercise-clicked.png",
                "next-exercise.png");
        NEXT_BTN.setBounds(782, 377, 130, 41);


        JButton RESTART_BTN = buttonSetter(
                "restart-exercise.png",
```

```
                "restart-exercise-clicked.png",
                "restart-exercise.png");
        RESTART_BTN.setBounds(782, 336, 130, 41);

        JButton PAUSE_BTN = buttonSetter(
                "pause-exercise.png",
                "pause-exercise-clicked.png",
                "pause-exercise.png");
        PAUSE_BTN.setBounds(782, 294, 130, 41);

        ImageIcon backgroundImageIcon = new ImageIcon(getClass().getClassLoader().getResource("LIBRARY-PICS.png"));
        JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
        backgroundImageLabel.setBounds(0, 0, 960, 540);
        contentPane.add(backgroundImageLabel);

        setContentPane(contentPane);

        // Initialize the list of exercises
        exercises = new ArrayList<>();
        exercises.add("CALF-RAISE-ANIMATION.gif");
        exercises.add("JUMPING-JACK-ANIMATION.gif");
        exercises.add("PLANK-ANIMATION.gif");
        exercises.add("PUSH-UP-ANIMATION.gif");
        exercises.add("SIT-OUTS-ANIMATION.gif");
        exercises.add("SIT-UP-ANIMATION.gif");
        exercises.add("SQUATS-ANIMATION.gif");
        exercises.add("WALKING-LUNGE-ANIMATION.gif");

        // Initialize and start the timer
        initializeTimer();

        // Add action listener for PAUSE_BTN
        PAUSE_BTN.addActionListener(new ActionListener() {
```

```
            public void actionPerformed(ActionEvent e) {
                toggleTimer();
            }
        });

        // Add action listener for RESTART_BTN
        RESTART_BTN.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                restartTimer();
            }
        });

        // Add action listener for NEXT_BTN
        NEXT_BTN.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                nextExercise();
            }
        });
    }

      public JLabel imageSetter(String imageLocation) {
        ImageIcon imageIcon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));
        JLabel label = new JLabel(imageIcon);
        contentPane.add(label);
        return label;
    }

    public JButton buttonSetter(String imageLocation, String enteredIconLocation, String exitedIconLocation) {
        ImageIcon icon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));
        JButton button = new JButton(icon);
        button.setBorder(null);
        button.setOpaque(false);
```

```
        button.setContentAreaFilled(false);
        button.setBorderPainted(false);
        button.setSize(icon.getIconWidth(), icon.getIconHeight());
        button.addMouseListener(getMouseListener(enteredIconLocation,
exitedIconLocation));
        contentPane.add(button);
        return button;
    }

    private MouseAdapter getMouseListener(String enteredIconLocation, String
exitedIconLocation) {
        return new MouseAdapter() {
            public void mouseEntered(MouseEvent e) {
                ImageIcon enteredIcon = new
ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));
                ((JButton) e.getSource()).setIcon(enteredIcon);
            }

            public void mouseExited(MouseEvent e) {
                ImageIcon exitedIcon = new
ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));
                ((JButton) e.getSource()).setIcon(exitedIcon);
            }
        };
    }

    private void initializeTimer() {
        timer = new Timer(1000, new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                if (timeRemaining > 0) {
                    timeRemaining--;
                    TIMER_LBL.setText(String.valueOf(timeRemaining));
                } else {
                    timer.stop();
                    showRestDialog();
```

```
            }
        }
    });
    timer.start();
}

private void toggleTimer() {
    if (isPaused) {
        timer.start();
        isPaused = false;
    } else {
        timer.stop();
        isPaused = true;
    }
}

private void restartTimer() {
    timer.stop();
    timeRemaining = 30;
    TIMER_LBL.setText(String.valueOf(timeRemaining));
    timer.start();
    isPaused = false;
}

private void nextExercise() {
    currentExerciseIndex++; // Increment first
    if (currentExerciseIndex < exercises.size()) {
        String exerciseGifPath = exercises.get(currentExerciseIndex);
        GIF_LBL.setIcon(new
ImageIcon(getClass().getClassLoader().getResource(exerciseGifPath)));
        restartTimer();
    } else {
        JOptionPane.showMessageDialog(contentPane, "Well done!");
```

```
            // Stop the timer and dispose of any open dialogs
            if (timer != null) {
                timer.stop();
            }
            if (restDialog != null) {
                restDialog.dispose();
            }

            // Navigate to main menu
            LANDING_SIMPLIFIED_UI landingSimplified = new LANDING_SIMPLIFIED_UI();
            landingSimplified.setVisible(true);
            dispose();
        }
    }

    private void showRestDialog() {
        final int restTime = 10; // Rest time in seconds
        restDialog = new JDialog(this, "Congratulations!", true);
        restDialog.getContentPane().setLayout(null);
        restDialog.setSize(423, 200);
        restDialog.setDefaultCloseOperation(JDialog.DO_NOTHING_ON_CLOSE);

        JLabel messageLabel = new JLabel("Well Done! Take a rest for the next " + restTime + " seconds");
        messageLabel.setHorizontalAlignment(SwingConstants.CENTER);
        messageLabel.setBounds(50, 30, 300, 30);
        restDialog.getContentPane().add(messageLabel);

        JButton okButton = new JButton("Skip");
        okButton.setBounds(150, 100, 100, 30);
        restDialog.getContentPane().add(okButton);

        // Create a timer for the dialog
```

```
        Timer dialogTimer = new Timer(1000, new ActionListener() {
            int timeLeft = restTime;

            @Override
            public void actionPerformed(ActionEvent e) {
                timeLeft--;
                messageLabel.setText("Well Done! Take a rest for the next " + timeLeft + "
seconds");
                if (timeLeft <= 0) {
                    restDialog.dispose();
                    nextExercise();
                    ((Timer) e.getSource()).stop();
                }
            }
        });
        dialogTimer.start();

        okButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                restDialog.dispose();
                nextExercise();
                dialogTimer.stop();
            }
        });

        restDialog.setLocationRelativeTo(contentPane);
        restDialog.setVisible(true);
    }
}
```

//EXERCISE_HARD_UI

```
package application;
```

```
import java.awt.Color;
import java.awt.EventQueue;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.SwingConstants;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.Timer;
import java.util.ArrayList;
import java.util.List;

public class EXERCISE_HARD_UI extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JLabel TIMER_LBL;
    private JLabel GIF_LBL;
    private Timer timer;
    private int timeRemaining = 90;
    private boolean isPaused = false;
    private List<String> exercises;
    private int currentExerciseIndex = 0;
    private JDialog restDialog = null; // Keep track of the rest dialog
```

```
    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                                        LANDING_SIMPLIFIED_UI landingSimplified = new
LANDING_SIMPLIFIED_UI();
                                        landingSimplified.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }


    /**
     * Create the frame.
     */
    public EXERCISE_HARD_UI() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 960, 540);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        contentPane.setLayout(null);
        setTitle("Health and Fitness Companion");
        setResizable(false);


        ImageIcon mainIcon = new
ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
        setIconImage(mainIcon.getImage());
```

```
        TIMER_LBL = new JLabel("90");
        TIMER_LBL.setFont(new Font("Century Gothic", Font.BOLD, 99));
        TIMER_LBL.setHorizontalAlignment(SwingConstants.CENTER);
        TIMER_LBL.setBounds(689, 110, 206, 124);
        TIMER_LBL.setForeground(Color.decode("#0B9F4F")); // Set text color
        contentPane.add(TIMER_LBL);

        JLabel TITLE_LBL = imageSetter("hard-level.png");
        TITLE_LBL.setBounds(72, 15, 791, 84);

        GIF_LBL = new JLabel(new ImageIcon(getClass().getClassLoader().getResource("AIR-CRUNCH-ANIMATION.gif")));
        GIF_LBL.setBounds(52, 117, 600, 338);
        contentPane.add(GIF_LBL);

        JButton MENU_BTN = buttonSetter(
                "main-menu-exercise.png",
                "main-menu-exercise-clicked.png",
                "main-menu-exercise.png");
        MENU_BTN.setBounds(782, 419, 130, 41);
        MENU_BTN.addActionListener(e -> {
            // Stop the timer and dispose of any open dialogs
            if (timer != null) {
                timer.stop();
            }
            if (restDialog != null) {
                restDialog.dispose();
            }
                        LANDING_SIMPLIFIED_UI landingSimplified = new LANDING_SIMPLIFIED_UI();
                        landingSimplified.setVisible(true);
            dispose();
        });
```

```
JButton NEXT_BTN = buttonSetter(
        "next-exercise.png",
        "next-exercise-clicked.png",
        "next-exercise.png");
NEXT_BTN.setBounds(782, 377, 130, 41);

JButton RESTART_BTN = buttonSetter(
        "restart-exercise.png",
        "restart-exercise-clicked.png",
        "restart-exercise.png");
RESTART_BTN.setBounds(782, 336, 130, 41);

JButton PAUSE_BTN = buttonSetter(
        "pause-exercise.png",
        "pause-exercise-clicked.png",
        "pause-exercise.png");
PAUSE_BTN.setBounds(782, 294, 130, 41);

ImageIcon backgroundImageIcon = new ImageIcon(getClass().getClassLoader().getResource("LIBRARY-PICS.png"));
JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
backgroundImageLabel.setBounds(0, 0, 960, 540);
contentPane.add(backgroundImageLabel);

setContentPane(contentPane);

// Initialize the list of exercises
exercises = new ArrayList<>();
exercises.add("CALF-RAISE-ANIMATION.gif");
exercises.add("JUMPING-JACK-ANIMATION.gif");
exercises.add("PLANK-ANIMATION.gif");
exercises.add("PUSH-UP-ANIMATION.gif");
exercises.add("SIT-OUTS-ANIMATION.gif");
exercises.add("SIT-UP-ANIMATION.gif");
```

```
        exercises.add("SQUATS-ANIMATION.gif");
        exercises.add("WALKING-LUNGE-ANIMATION.gif");

        // Initialize and start the timer
        initializeTimer();

        // Add action listener for PAUSE_BTN
        PAUSE_BTN.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                toggleTimer();
            }
        });

        // Add action listener for RESTART_BTN
        RESTART_BTN.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                restartTimer();
            }
        });

        // Add action listener for NEXT_BTN
        NEXT_BTN.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                nextExercise();
            }
        });
    }

      public JLabel imageSetter(String imageLocation) {
        ImageIcon imageIcon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));
        JLabel label = new JLabel(imageIcon);
        contentPane.add(label);
        return label;
```

```
    }

    public JButton buttonSetter(String imageLocation, String enteredIconLocation, String exitedIconLocation) {
        ImageIcon icon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));
        JButton button = new JButton(icon);
        button.setBorder(null);
        button.setOpaque(false);
        button.setContentAreaFilled(false);
        button.setBorderPainted(false);
        button.setSize(icon.getIconWidth(), icon.getIconHeight());
        button.addMouseListener(getMouseListener(enteredIconLocation, exitedIconLocation));
        contentPane.add(button);
        return button;
    }

    private MouseAdapter getMouseListener(String enteredIconLocation, String exitedIconLocation) {
        return new MouseAdapter() {
            public void mouseEntered(MouseEvent e) {
                ImageIcon enteredIcon = new ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));
                ((JButton) e.getSource()).setIcon(enteredIcon);
            }

            public void mouseExited(MouseEvent e) {
                ImageIcon exitedIcon = new ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));
                ((JButton) e.getSource()).setIcon(exitedIcon);
            }
        };
    }

    private void initializeTimer() {
```

```
    timer = new Timer(1000, new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            if (timeRemaining > 0) {
                timeRemaining--;
                TIMER_LBL.setText(String.valueOf(timeRemaining));
            } else {
                timer.stop();
                showRestDialog();
            }
        }
    });
    timer.start();
}

private void toggleTimer() {
    if (isPaused) {
        timer.start();
        isPaused = false;
    } else {
        timer.stop();
        isPaused = true;
    }
}

private void restartTimer() {
    timer.stop();
    timeRemaining = 90;
    TIMER_LBL.setText(String.valueOf(timeRemaining));
    timer.start();
    isPaused = false;
}

private void nextExercise() {
```

```
        currentExerciseIndex++; // Increment first
        if (currentExerciseIndex < exercises.size()) {
            String exerciseGifPath = exercises.get(currentExerciseIndex);
            GIF_LBL.setIcon(new
ImageIcon(getClass().getClassLoader().getResource(exerciseGifPath)));
            restartTimer();
        } else {
            JOptionPane.showMessageDialog(contentPane, "Well done!");

            // Stop the timer and dispose of any open dialogs
            if (timer != null) {
                timer.stop();
            }
            if (restDialog != null) {
                restDialog.dispose();
            }

            // Navigate to main menu
            LANDING_SIMPLIFIED_UI landingSimplified = new
LANDING_SIMPLIFIED_UI();
            landingSimplified.setVisible(true);
            dispose();
        }
    }

    private void showRestDialog() {
        final int restTime = 10; // Rest time in seconds
        restDialog = new JDialog(this, "Congratulations!", true);
        restDialog.getContentPane().setLayout(null);
        restDialog.setSize(423, 200);
        restDialog.setDefaultCloseOperation(JDialog.DO_NOTHING_ON_CLOSE);

        JLabel messageLabel = new JLabel("Well Done! Take a rest for the next " +
restTime + " seconds");
        messageLabel.setHorizontalAlignment(SwingConstants.CENTER);
```

```
        messageLabel.setBounds(50, 30, 300, 30);
        restDialog.getContentPane().add(messageLabel);

        JButton okButton = new JButton("Skip");
        okButton.setBounds(150, 100, 100, 30);
        restDialog.getContentPane().add(okButton);

        // Create a timer for the dialog
        Timer dialogTimer = new Timer(1000, new ActionListener() {
            int timeLeft = restTime;

            @Override
            public void actionPerformed(ActionEvent e) {
                timeLeft--;
                messageLabel.setText("Well Done! Take a rest for the next " + timeLeft + "
seconds");
                if (timeLeft <= 0) {
                    restDialog.dispose();
                    nextExercise();
                    ((Timer) e.getSource()).stop();
                }
            }
        });
        dialogTimer.start();

        okButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                restDialog.dispose();
                nextExercise();
                dialogTimer.stop();
            }
        });
```

```
        restDialog.setLocationRelativeTo(contentPane);
        restDialog.setVisible(true);
    }
}
```

//EXERCISE_LIBRARY_UI

```
package application;

import java.awt.BorderLayout;
import java.awt.EventQueue;
import java.awt.Image;

import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.SwingConstants;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.io.File;
import java.awt.event.ActionEvent;

public class EXERCISE_LIBRARY_UI extends JFrame {

        private static final long serialVersionUID = 1L;
        private JPanel contentPane;

        /**
         * Launch the application.
         */
```

```
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    LANDING_SIMPLIFIED_UI frame = new LANDING_SIMPLIFIED_UI();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the frame.
     */
    public EXERCISE_LIBRARY_UI() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 960, 540);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        contentPane.setLayout(null);
    setTitle("Health and Fitness Companion");
    setResizable(false);

    ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
    setIconImage(mainIcon.getImage());

    JLabel EXERCISE_LABEL = new JLabel(new ImageIcon(getClass().getClassLoader().getResource("exercises-library.png")));
    EXERCISE_LABEL.setBounds(127, 11, 710, 40);
    contentPane.add(EXERCISE_LABEL);
```

```
JButton BACK_BTN = new JButton();
BACK_BTN = buttonSetter("back-exercises.png",
                        "back-exercises-clicked.png",
                        "back-exercises.png");
BACK_BTN.setBounds(780, 450, 133, 40);
BACK_BTN.addActionListener(ActionLister -> {
  LANDING_SIMPLIFIED_UI frame = new LANDING_SIMPLIFIED_UI();
   frame.setVisible(true);
  dispose();
});

JButton PUSHUP_BTN = new JButton();
PUSHUP_BTN = buttonSetter("push-up-icon.png",
                                    "push-up-icon-clicked.png",
                                    "push-up-icon.png");
PUSHUP_BTN.setBounds(131, 67, 156, 97);
PUSHUP_BTN.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        ExerciseWindow exer = new ExerciseWindow(
            "PUSH-UP.gif",
            "push-up-text.png");
    }
});

JButton SITUP_BTN = new JButton();
SITUP_BTN = buttonSetter("sit-up-icon.png",
                        "sit-up-icon-clicked.png",
                        "sit-up-icon.png");
SITUP_BTN.setBounds(403, 67, 156, 97);
SITUP_BTN.addActionListener(ActionListener -> {
  ExerciseWindow exer = new ExerciseWindow(
                    "SIT-UP.gif",
                    "sit-up-text.png");
```

```
});

JButton BODYSQUAT_BTN = new JButton();
BODYSQUAT_BTN = buttonSetter("squats-icon.png",
                             "squats-icon-clicked.png",
                             "squats-icon.png");
BODYSQUAT_BTN.setBounds(678, 67, 156, 97);
BODYSQUAT_BTN.addActionListener(ActionListener -> {
  ExerciseWindow exer = new ExerciseWindow(
                   "AIR-SQUAT.gif",
                   "body-squats-text.png");
});

JButton WALKINGLUNGE_BTN = new JButton();
WALKINGLUNGE_BTN = buttonSetter("walking-lunge-icon.png",
                             "walking-lunge-icon-clicked.png",
                             "walking-lunge-icon.png");
WALKINGLUNGE_BTN.setBounds(131, 197, 156, 97);
WALKINGLUNGE_BTN.addActionListener(ActionListener -> {
  ExerciseWindow exer = new ExerciseWindow(
                   "WALKING-LUNGES.gif",
                   "walking-lunge-text.png");
});

JButton PLANK_BTN = new JButton();
PLANK_BTN = buttonSetter("plank-icon.png",
                             "plank-icon-clicked.png",
                             "plank-icon.png");
PLANK_BTN.setBounds(403, 197, 156, 97);
PLANK_BTN.addActionListener(ActionListener -> {
  ExerciseWindow exer = new ExerciseWindow(
                   "FOREARM-PLANK.gif",
                   "plank-text.png");
```

```
});

JButton JUMPING_JACK_BTN = new JButton();
JUMPING_JACK_BTN = buttonSetter("jumping-jack-icon.png",
                                "jumping-jack-icon-clicked.png",
                                "jumping-jack-icon.png");
JUMPING_JACK_BTN.setBounds(678, 198, 156, 97);
JUMPING_JACK_BTN.addActionListener(ActionListener -> {
  ExerciseWindow exer = new ExerciseWindow(
                        "JUMPING-JACKS.gif",
                        "jumping-jack-text.png");
});

JButton BIKECRUNCH_BTN = new JButton();
BIKECRUNCH_BTN = buttonSetter("bike-crunch-icon.png",
                                "bike-crunch-icon-clicked.png",
                                "bike-crunch-icon.png");
BIKECRUNCH_BTN.setBounds(131, 330, 156, 97);
BIKECRUNCH_BTN.addActionListener(ActionListener -> {
  ExerciseWindow exer = new ExerciseWindow(
                        "BICYCLE-CRUNCH.gif",
                        "bike-crunch-text.png");
});

JButton WALLSIT_BTN = new JButton();
WALLSIT_BTN = buttonSetter("wall-sit-icon.png",
                                "wall-sit-icon-clicked.png",
                                "wall-sit-icon.png");
WALLSIT_BTN.setBounds(403, 331, 156, 97);
WALLSIT_BTN.addActionListener(ActionListener -> {
  ExerciseWindow exer = new ExerciseWindow(
                        "WALL-SIT.gif",
                        "wallsit-text.png");
```

```
        });


        JButton CALFRAISE_BTN = new JButton("standing calf raise");
        CALFRAISE_BTN = buttonSetter("calfraise-icon.png",
                                     "calfraise-icon-clicked.png",
                                     "calfraise-icon.png");
        CALFRAISE_BTN.setBounds(678, 331, 156, 97);
        CALFRAISE_BTN.addActionListener(ActionListener -> {
          ExerciseWindow exer = new ExerciseWindow(
                           "STANDING-CALF-RAISE.gif",
                           "calfraise-text.png");
        });


        ImageIcon backgroundImageIcon = new ImageIcon(getClass().getClassLoader().getResource("LIBRARY-PICS.png"));
        JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
        backgroundImageLabel.setBounds(0, 0, 960, 540);
        contentPane.add(backgroundImageLabel);


                setContentPane(contentPane);
          }


    public JButton buttonSetter(String imageLocation, String enteredIconLocation, String exitedIconLocation) {
        ImageIcon icon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));
        JButton button = new JButton(icon);
        button.setBorder(null);
        button.setOpaque(false);
        button.setContentAreaFilled(false);
        button.setBorderPainted(false);
        button.setSize(icon.getIconWidth(), icon.getIconHeight());
        button.addMouseListener(getMouseListener(enteredIconLocation, exitedIconLocation));
        contentPane.add(button);
```

```
        return button;
    }

    private MouseAdapter getMouseListener(String enteredIconLocation, String exitedIconLocation) {
        return new MouseAdapter() {
            public void mouseEntered(MouseEvent e) {
                ImageIcon enteredIcon = new ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));
                ((JButton) e.getSource()).setIcon(enteredIcon);
            }

            public void mouseExited(MouseEvent e) {
                ImageIcon exitedIcon = new ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));
                ((JButton) e.getSource()).setIcon(exitedIcon);
            }
        };
    }
}

class ExerciseWindow extends JFrame {
        private static final long serialVersionUID = 1L;
        private JPanel contentPane;

        public ExerciseWindow(String gifPath, String description) {
        setTitle("Exercise Library");
        setBounds(100, 100, 960, 540);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setVisible(true);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        contentPane.setLayout(null);

        EXERCISE_LIBRARY_UI lib = new EXERCISE_LIBRARY_UI();
```

```
        JButton btnNewButton = new JButton("back");
        btnNewButton = lib.buttonSetter("back-exercises.png",
                                                        "back-exercises-clicked.png",
                                                        "back-exercises.png");
        btnNewButton.setBounds(780, 450, 133, 40);
        contentPane.add(btnNewButton);
        btnNewButton.addActionListener(ActionListener -> {
            lib.setVisible(true);
            dispose();
        });

        JLabel lblNewLabel = new JLabel(new ImageIcon(getClass().getClassLoader().getResource("exercise_label.png")));
        lblNewLabel.setHorizontalAlignment(SwingConstants.CENTER);
        lblNewLabel.setBounds(178, 11, 550, 40);
        contentPane.add(lblNewLabel);

        JLabel gifLabel = new JLabel(new ImageIcon(getClass().getClassLoader().getResource(gifPath)));
        gifLabel.setBounds(40, 71, 600, 338);
        gifLabel.setHorizontalAlignment(SwingConstants.CENTER);
        contentPane.add(gifLabel);

        JLabel descriptionLabel = new JLabel(new ImageIcon(getClass().getClassLoader().getResource(description)));
        descriptionLabel.setBounds(656, 71, 235, 338);
        descriptionLabel.setHorizontalAlignment(SwingConstants.CENTER);
        contentPane.add(descriptionLabel);

        ImageIcon backgroundImageIcon = new ImageIcon(getClass().getClassLoader().getResource("LIBRARY-PICS.png"));
        JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
        backgroundImageLabel.setBounds(0, 0, 960, 540);
        contentPane.add(backgroundImageLabel);
```

```
        setContentPane(contentPane);
          }
}
```

//EXERCISE_MEDIUM_UI

```
package application;

import java.awt.Color;
import java.awt.EventQueue;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.SwingConstants;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.Timer;
import java.util.ArrayList;
import java.util.List;

public class EXERCISE_MEDIUM_UI extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JLabel TIMER_LBL;
```

```
    private JLabel GIF_LBL;
    private Timer timer;
    private int timeRemaining = 60;
    private boolean isPaused = false;
    private List<String> exercises;
    private int currentExerciseIndex = 0;
    private JDialog restDialog = null; // Keep track of the rest dialog

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                                        LANDING_SIMPLIFIED_UI landingSimplified = new
LANDING_SIMPLIFIED_UI();
                                        landingSimplified.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
     * Create the frame.
     */
    public EXERCISE_MEDIUM_UI() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 960, 540);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        contentPane.setLayout(null);
```

```
        setTitle("Health and Fitness Companion");
        setResizable(false);

        ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
        setIconImage(mainIcon.getImage());

        TIMER_LBL = new JLabel("60");
        TIMER_LBL.setFont(new Font("Century Gothic", Font.BOLD, 99));
        TIMER_LBL.setHorizontalAlignment(SwingConstants.CENTER);
        TIMER_LBL.setBounds(689, 110, 206, 124);
        TIMER_LBL.setForeground(Color.decode("#0B9F4F")); // Set text color
        contentPane.add(TIMER_LBL);

        JLabel TITLE_LBL = imageSetter("medium-level.png");
        TITLE_LBL.setBounds(72, 15, 791, 84);

        GIF_LBL = new JLabel(new ImageIcon(getClass().getClassLoader().getResource("AIR-CRUNCH-ANIMATION.gif")));
        GIF_LBL.setBounds(52, 117, 600, 338);
        contentPane.add(GIF_LBL);

        JButton MENU_BTN = buttonSetter(
                "main-menu-exercise.png",
                "main-menu-exercise-clicked.png",
                "main-menu-exercise.png");
        MENU_BTN.setBounds(782, 419, 130, 41);
        MENU_BTN.addActionListener(e -> {
            // Stop the timer and dispose of any open dialogs
            if (timer != null) {
                timer.stop();
            }
            if (restDialog != null) {
                restDialog.dispose();
```

```
        }
                        LANDING_SIMPLIFIED_UI landingSimplified = new
LANDING_SIMPLIFIED_UI();
                        landingSimplified.setVisible(true);
        dispose();
    });

    JButton NEXT_BTN = buttonSetter(
            "next-exercise.png",
            "next-exercise-clicked.png",
            "next-exercise.png");
    NEXT_BTN.setBounds(782, 377, 130, 41);

    JButton RESTART_BTN = buttonSetter(
            "restart-exercise.png",
            "restart-exercise-clicked.png",
            "restart-exercise.png");
    RESTART_BTN.setBounds(782, 336, 130, 41);

    JButton PAUSE_BTN = buttonSetter(
            "pause-exercise.png",
            "pause-exercise-clicked.png",
            "pause-exercise.png");
    PAUSE_BTN.setBounds(782, 294, 130, 41);

    ImageIcon backgroundImageIcon = new
ImageIcon(getClass().getClassLoader().getResource("LIBRARY-PICS.png"));
    JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
    backgroundImageLabel.setBounds(0, 0, 960, 540);
    contentPane.add(backgroundImageLabel);

    setContentPane(contentPane);

    // Initialize the list of exercises
```

```
exercises = new ArrayList<>();
exercises.add("CALF-RAISE-ANIMATION.gif");
exercises.add("JUMPING-JACK-ANIMATION.gif");
exercises.add("PLANK-ANIMATION.gif");
exercises.add("PUSH-UP-ANIMATION.gif");
exercises.add("SIT-OUTS-ANIMATION.gif");
exercises.add("SIT-UP-ANIMATION.gif");
exercises.add("SQUATS-ANIMATION.gif");
exercises.add("WALKING-LUNGE-ANIMATION.gif");

// Initialize and start the timer
initializeTimer();

// Add action listener for PAUSE_BTN
PAUSE_BTN.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        toggleTimer();
    }
});

// Add action listener for RESTART_BTN
RESTART_BTN.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        restartTimer();
    }
});

// Add action listener for NEXT_BTN
NEXT_BTN.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        nextExercise();
    }
});
```

```
    }


        public JLabel imageSetter(String imageLocation) {

        ImageIcon imageIcon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));

        JLabel label = new JLabel(imageIcon);

        contentPane.add(label);

        return label;

    }


    public JButton buttonSetter(String imageLocation, String enteredIconLocation, String exitedIconLocation) {

        ImageIcon icon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));

        JButton button = new JButton(icon);

        button.setBorder(null);

        button.setOpaque(false);

        button.setContentAreaFilled(false);

        button.setBorderPainted(false);

        button.setSize(icon.getIconWidth(), icon.getIconHeight());

        button.addMouseListener(getMouseListener(enteredIconLocation, exitedIconLocation));

        contentPane.add(button);

        return button;

    }


    private MouseAdapter getMouseListener(String enteredIconLocation, String exitedIconLocation) {

        return new MouseAdapter() {

            public void mouseEntered(MouseEvent e) {

                ImageIcon enteredIcon = new ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));

                ((JButton) e.getSource()).setIcon(enteredIcon);

            }


            public void mouseExited(MouseEvent e) {
```

```
            ImageIcon exitedIcon = new ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));
            ((JButton) e.getSource()).setIcon(exitedIcon);
        }
    };
}

private void initializeTimer() {
    timer = new Timer(1000, new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            if (timeRemaining > 0) {
                timeRemaining--;
                TIMER_LBL.setText(String.valueOf(timeRemaining));
            } else {
                timer.stop();
                showRestDialog();
            }
        }
    });
    timer.start();
}

private void toggleTimer() {
    if (isPaused) {
        timer.start();
        isPaused = false;
    } else {
        timer.stop();
        isPaused = true;
    }
}

private void restartTimer() {
    timer.stop();
```

```
        timeRemaining = 60;
        TIMER_LBL.setText(String.valueOf(timeRemaining));
        timer.start();
        isPaused = false;
    }

    private void nextExercise() {
        currentExerciseIndex++; // Increment first
        if (currentExerciseIndex < exercises.size()) {
            String exerciseGifPath = exercises.get(currentExerciseIndex);
            GIF_LBL.setIcon(new
ImageIcon(getClass().getClassLoader().getResource(exerciseGifPath)));
            restartTimer();
        } else {
            JOptionPane.showMessageDialog(contentPane, "Well done!");

            // Stop the timer and dispose of any open dialogs
            if (timer != null) {
                timer.stop();
            }
            if (restDialog != null) {
                restDialog.dispose();
            }

            // Navigate to main menu
            LANDING_SIMPLIFIED_UI landingSimplified = new
LANDING_SIMPLIFIED_UI();
            landingSimplified.setVisible(true);
            dispose();
        }
    }

    private void showRestDialog() {
        final int restTime = 10; // Rest time in seconds
```

```
        restDialog = new JDialog(this, "Congratulations!", true);
        restDialog.getContentPane().setLayout(null);
        restDialog.setSize(423, 200);
        restDialog.setDefaultCloseOperation(JDialog.DO_NOTHING_ON_CLOSE);

        JLabel messageLabel = new JLabel("Well Done! Take a rest for the next " + restTime + " seconds");
        messageLabel.setHorizontalAlignment(SwingConstants.CENTER);
        messageLabel.setBounds(50, 30, 300, 30);
        restDialog.getContentPane().add(messageLabel);

        JButton okButton = new JButton("Skip");
        okButton.setBounds(150, 100, 100, 30);
        restDialog.getContentPane().add(okButton);

        // Create a timer for the dialog
        Timer dialogTimer = new Timer(1000, new ActionListener() {
            int timeLeft = restTime;

            @Override
            public void actionPerformed(ActionEvent e) {
                timeLeft--;
                messageLabel.setText("Well Done! Take a rest for the next " + timeLeft + " seconds");
                if (timeLeft <= 0) {
                    restDialog.dispose();
                    nextExercise();
                    ((Timer) e.getSource()).stop();
                }
            }
        });
        dialogTimer.start();

        okButton.addActionListener(new ActionListener() {
```

```
            @Override
            public void actionPerformed(ActionEvent e) {
                restDialog.dispose();
                nextExercise();
                dialogTimer.stop();
            }
        });

        restDialog.setLocationRelativeTo(contentPane);
        restDialog.setVisible(true);
    }
}
```

//INSTRUCTIONS_UI

```
package application;

import java.awt.EventQueue;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;

public class INSTRUCTIONS_UI extends JFrame {

            private static final long serialVersionUID = 1L;
            private static JPanel contentPane;
```

```
	/**
	 * Launch the application.
	 */
	public static void main(String[] args) {
		EventQueue.invokeLater(new Runnable() {
			public void run() {
				try {
		LANDING_SIMPLIFIED_UI frame = new LANDING_SIMPLIFIED_UI();
		frame.setVisible(true);
				} catch (Exception e) {
					e.printStackTrace();
				}
			}
		});
	}

	/**
	 * Create the frame.
	 */
	public INSTRUCTIONS_UI() {
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		setBounds(100, 100, 960, 540);
		contentPane = new JPanel();
		contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
		contentPane.setLayout(null);
	setTitle("Health and Fitness Companion");
	setResizable(false);

	ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
	setIconImage(mainIcon.getImage());

	JButton BACK_BTN = new JButton();
```

```
BACK_BTN = buttonSetter("back-exercises.png",
                        "back-exercises-clicked.png",
                        "back-exercises.png");
BACK_BTN.setBounds(780, 450, 133, 40);
BACK_BTN.addActionListener(ActionLister -> {
  LANDING_SIMPLIFIED_UI frame = new LANDING_SIMPLIFIED_UI();
   frame.setVisible(true);
  dispose();
});

JButton FEATURE_1_BTN = new JButton();
FEATURE_1_BTN = buttonSetter("exercise-instruction.png",
                        "exercise-instruction-clicked.png",
                        "exercise-instruction.png");
      FEATURE_1_BTN.setBounds(363, 159, 216, 40);
      FEATURE_1_BTN.addActionListener(ActionLister -> {
            showImage("feature1-instruction.png");
            dispose();
});

JButton FEATURE_2_BTN = new JButton();
FEATURE_2_BTN = buttonSetter("bmi-instruction.png",
                        "bmi-instruction-clicked.png",
                        "bmi-instruction.png");
      FEATURE_2_BTN.setBounds(364, 214, 216, 40);
      FEATURE_2_BTN.addActionListener(ActionLister -> {
            showImage("feature2-instruction.png");
            dispose();
});

JButton FEATURE_3_BTN = new JButton();
FEATURE_3_BTN = buttonSetter("meal-instruction.png",
                        "meal-instruction-clicked.png",
```

```
                                "meal-instruction.png");
        FEATURE_3_BTN.setBounds(365, 269, 216, 40);
        FEATURE_3_BTN.addActionListener(ActionLister -> {
            showImage("feature3-instruction.png");
            dispose();
    });

    JButton FEATURE_4_BTN = new JButton();
    FEATURE_4_BTN = buttonSetter("alarm-instruction.png",
                                "alarm-instruction-clicked.png",
                                "alarm-instruction.png");
        FEATURE_4_BTN.setBounds(365, 324, 216, 40);
        FEATURE_4_BTN.addActionListener(ActionLister -> {
            showImage("feature4-instruction.png");
            dispose();
    });

    JButton FEATURE_5_BTN = new JButton();
    FEATURE_5_BTN = buttonSetter("library-instruction.png",
                                "library-instruction-clicked.png",
                                "library-instruction.png");
        FEATURE_5_BTN.setBounds(365, 379, 216, 40);
        FEATURE_5_BTN.addActionListener(ActionLister -> {
            showImage("feature5-instruction.png");
            dispose();
    });

    ImageIcon backgroundImageIcon = new
ImageIcon(getClass().getClassLoader().getResource("instructions-new-bg.png"));
    JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
    backgroundImageLabel.setBounds(0, 0, 960, 540);
    contentPane.add(backgroundImageLabel);

            setContentPane(contentPane);
```

```
        }


    public JButton buttonSetter(String imageLocation, String enteredIconLocation, String exitedIconLocation) {

        ImageIcon icon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));

        JButton button = new JButton(icon);

        button.setBorder(null);

        button.setOpaque(false);

        button.setContentAreaFilled(false);

        button.setBorderPainted(false);

        button.setSize(icon.getIconWidth(), icon.getIconHeight());

        button.addMouseListener(getMouseListener(enteredIconLocation, exitedIconLocation));

        contentPane.add(button);

        return button;

    }


    private MouseAdapter getMouseListener(String enteredIconLocation, String exitedIconLocation) {

        return new MouseAdapter() {

            public void mouseEntered(MouseEvent e) {

                ImageIcon enteredIcon = new ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));

                ((JButton) e.getSource()).setIcon(enteredIcon);

            }



            public void mouseExited(MouseEvent e) {

                ImageIcon exitedIcon = new ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));

                ((JButton) e.getSource()).setIcon(exitedIcon);

            }

        };

    }


    private void showImage(String imagePath) {
```

```
        JFrame imageFrame = new JFrame();
        imageFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        imageFrame.setBounds(100, 100, 960, 540);
        imageFrame.setTitle("Health and Fitness Companion");
        imageFrame.setLayout(null);  // Use null layout to manually set bounds

        ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
        imageFrame.setIconImage(mainIcon.getImage());

        // Create and add BACK_BTN
        JButton BACK_BTN = new JButton();
        BACK_BTN = buttonSetter("back-exercises.png",
                                "back-exercises-clicked.png",
                                "back-exercises.png");
        BACK_BTN.setBounds(780, 450, 133, 40);
        BACK_BTN.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                INSTRUCTIONS_UI ins = new INSTRUCTIONS_UI();
                ins.setVisible(true);
                imageFrame.dispose();
            }
        });
        imageFrame.add(BACK_BTN);

        ImageIcon imageIcon = new ImageIcon(getClass().getClassLoader().getResource(imagePath));
        JLabel imageLabel = new JLabel(imageIcon);
        imageLabel.setBounds(0, 0, 960, 540);
        imageFrame.add(imageLabel);

        imageFrame.setVisible(true);
    }
```

```
}
```

//LANDING_SIMPLIFIED_UI

```
package application;

import java.awt.Dimension;
import java.awt.EventQueue;
import java.awt.Image;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import java.awt.Font;
import java.awt.Graphics;

import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.ActionEvent;

import java.awt.Color;
import java.awt.Dimension;
import java.awt.Graphics;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.SwingConstants;

public class LANDING_SIMPLIFIED_UI extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
```

```
/**
 * Launch the application.
 */
public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                //Initialize the Frame
                LANDING_SIMPLIFIED_UI frame = new LANDING_SIMPLIFIED_UI();
               frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

/**
 * Create the frame.
 */
public LANDING_SIMPLIFIED_UI() {
       /*
        *  Frame Customization
        */
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setBounds(100, 100, 960, 540);
    contentPane = new JPanel();
    contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
    setContentPane(contentPane);
    getContentPane().setLayout(null);
    setTitle("Health and Fitness Companion");
    setResizable(false);
```

```
        ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
        setIconImage(mainIcon.getImage());

        JButton VOLUME_UP_BTN = buttonSetter(
                "exercise-simplified.png",
                "exercise-simplified-clicked.png",
                "exercise-simplified.png");
        VOLUME_UP_BTN.setBounds(594, 167, 262, 55);
        VOLUME_UP_BTN.addActionListener(e -> {
          EXERCISE_DIFFICULTY_UI exercise = new EXERCISE_DIFFICULTY_UI();
          exercise.setVisible(true);
          dispose();
        });


        JLabel INFO = imageSetter("info-main.png");
        INFO.setBounds(30, 137, 493, 353);

        JButton INSTRUCTIONS_BTN = buttonSetter(
                "instructions-simplified.png",
                "instructions-simplified-clicked.png",
                "instructions-simplified.png");
        INSTRUCTIONS_BTN.setBounds(594, 233, 262, 55);
        INSTRUCTIONS_BTN.addActionListener(e -> {
          INSTRUCTIONS_UI ins = new INSTRUCTIONS_UI();
          ins.setVisible(true);
          dispose();
        });

        JButton CREDITS_BTN = buttonSetter(
                "credits-simplified.png",
                "credits-simplified-clicked.png",
```

```
            "credits-simplified.png");
CREDITS_BTN.setBounds(594, 299, 262, 55);
CREDITS_BTN.addActionListener(e -> {
  CREDITS_UI credits = new CREDITS_UI();
  credits.setVisible(true);
  dispose();
});

JButton EXIT_BTN = buttonSetter(
            "exit-simplified.png",
            "exit-simplified-clicked.png",
            "exit-simplified.png");
EXIT_BTN.setBounds(594, 365, 262, 55);
EXIT_BTN.addActionListener(e -> {
  System.exit(0);
});

JButton LATEST_NEWS_BTN = buttonSetter(
            "latest-news-main-ui.png",
            "latest-news-main-ui-pressed.png",
            "latest-news-main-ui.png");
LATEST_NEWS_BTN.setBounds(794, 21, 120, 71);
contentPane.add(LATEST_NEWS_BTN);
LATEST_NEWS_BTN.addActionListener(ActionListener -> {
  EXERCISE_LIBRARY_UI exercise_lib = new EXERCISE_LIBRARY_UI();
  exercise_lib.setVisible(true);
  dispose();

});

JButton ALARM_CLOCK_BTN = buttonSetter(
            "alarm-clock-main-ui.png",
            "alarm-clock-main-ui-pressed.png",
```

```
            "alarm-clock-main-ui.png");
ALARM_CLOCK_BTN.setBounds(649, 21, 120, 71);
contentPane.add(ALARM_CLOCK_BTN);
ALARM_CLOCK_BTN.addActionListener(e -> {
  ALARM_CLOCK_UI alarmFeatureUI = new ALARM_CLOCK_UI();
  alarmFeatureUI.setVisible(true);
  dispose();
});

JButton MEAL_PLANNER_BTN = buttonSetter(
            "meal-planner-main-ui.png",
            "meal-planner-main-ui-pressed.png",
            "meal-planner-main-ui.png");
MEAL_PLANNER_BTN.setBounds(504, 21, 120, 71);
contentPane.add(MEAL_PLANNER_BTN);
MEAL_PLANNER_BTN.addActionListener(ActionListener -> {
  MEAL_PLANNER_UI mealPlanner = new MEAL_PLANNER_UI();
  mealPlanner.setVisible(true);
  dispose();
});

JButton BMI_TRACKER_BTN = buttonSetter(
            "bmi-tracker-main-ui.png",
            "bmi-tracker-main-ui-pressed.png",
            "bmi-tracker-main-ui.png");
BMI_TRACKER_BTN.setBounds(359, 21, 120, 71);
contentPane.add(BMI_TRACKER_BTN);

BMI_TRACKER_BTN.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent e) {
      BMI_FEATURE_UI bmiFeatureUI = new BMI_FEATURE_UI();
      bmiFeatureUI.setVisible(true);
      dispose();
```

```
        }
    });

    JButton EXERCISE_NOW_BTN = buttonSetter(
            "exercise-now-main-ui.png",
            "exercise-now-main-ui-pressed.png",
            "exercise-now-main-ui.png");
    EXERCISE_NOW_BTN.setBounds(214, 21, 120, 71);
    contentPane.add(EXERCISE_NOW_BTN);
    EXERCISE_NOW_BTN.addActionListener(e -> {
      EXERCISE_DIFFICULTY_UI diff = new EXERCISE_DIFFICULTY_UI();
      diff.setVisible(true);
      dispose();
    });

    JLabel labelLogo = imageSetter("logo-green.png");
    labelLogo.setBounds(10, 21, 194, 87);

    ImageIcon backgroundImageIcon = new ImageIcon(getClass().getClassLoader().getResource("main-ui-bg-simplified.png"));
    JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
    backgroundImageLabel.setSize(getSize());
    backgroundImageLabel.setBounds(0, 0, getWidth(), getHeight());
    contentPane.add(backgroundImageLabel, new Integer(Integer.MIN_VALUE));
    ((JPanel) contentPane).setOpaque(false);
  }

    public JLabel imageSetter(String imageLocation) {
    ImageIcon imageIcon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));
    JLabel label = new JLabel(imageIcon);
    contentPane.add(label);
    return label;
  }
```

```
    public JButton buttonSetter(String imageLocation, String enteredIconLocation, String
exitedIconLocation) {

        ImageIcon icon = new
ImageIcon(getClass().getClassLoader().getResource(imageLocation));

        JButton button = new JButton(icon);

        button.setBorder(null);

        button.setOpaque(false);

        button.setContentAreaFilled(false);

        button.setBorderPainted(false);

        button.setSize(icon.getIconWidth(), icon.getIconHeight());

        button.addMouseListener(getMouseListener(enteredIconLocation,
exitedIconLocation));

        contentPane.add(button);

        return button;

    }


    private MouseAdapter getMouseListener(String enteredIconLocation, String
exitedIconLocation) {

        return new MouseAdapter() {

            public void mouseEntered(MouseEvent e) {

                ImageIcon enteredIcon = new
ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));

                ((JButton) e.getSource()).setIcon(enteredIcon);

            }



            public void mouseExited(MouseEvent e) {

                ImageIcon exitedIcon = new
ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));

                ((JButton) e.getSource()).setIcon(exitedIcon);

            }

        };

    }

}
```

//MEAL_PLANNER_UI

```
package application;

import java.awt.Color;
import java.awt.Component;
import java.awt.Dimension;
import java.awt.EventQueue;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.Image;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.io.File;
import java.util.HashMap;
import java.util.Map;

import javax.swing.ButtonGroup;
import javax.swing.DefaultListCellRenderer;
import javax.swing.DefaultListModel;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JList;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
```

```
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.JToggleButton;
import javax.swing.SwingConstants;
import javax.swing.border.EmptyBorder;

public class MEAL_PLANNER_UI extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JPanel BUTTON_PANNEL;
    private JScrollPane scrollPane;
    private JLabel BREAKFAST_LBL;
    private JLabel LUNCH_LBL;
    private JLabel DINNER_LBL;
    private DefaultListModel<String> recipeListModel;
    private JToggleButton MONDAY_BTN;
    private JToggleButton TUESDAY_BTN;
    private JToggleButton WEDNESDAY_BTN;
    private JToggleButton THURSDAY_BTN;
    private JToggleButton FRIDAY_BTN;
    private JToggleButton SATURDAY_BTN;
    private JToggleButton SUNDAY_BTN;

    private String mondayBreakfast = "";
    private String mondayLunch = "";
    private String mondayDinner = "";
    private String tuesdayBreakfast = "";
    private String tuesdayLunch = "";
    private String tuesdayDinner = "";
    private String wednesdayBreakfast = "";
    private String wednesdayLunch = "";
```

```
    private String wednesdayDinner = "";
    private String thursdayBreakfast = "";
    private String thursdayLunch = "";
    private String thursdayDinner = "";
    private String fridayBreakfast = "";
    private String fridayLunch = "";
    private String fridayDinner = "";
    private String saturdayBreakfast = "";
    private String saturdayLunch = "";
    private String saturdayDinner = "";
    private String sundayBreakfast = "";
    private String sundayLunch = "";
    private String sundayDinner = "";

    private Map<String, String[]> mealPlans = new HashMap<>();

    /**
     * Launch the application.
     */
    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                                        LANDING_SIMPLIFIED_UI landingSimplified = new
LANDING_SIMPLIFIED_UI();
                                        landingSimplified.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    /**
```

```
     * Create the frame.
     */
    public MEAL_PLANNER_UI() {
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 960, 540);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);
        setTitle("Health and Fitness Companion");
        setResizable(false);

        ImageIcon mainIcon = new ImageIcon(getClass().getClassLoader().getResource("iconLogoPoster.png"));
        setIconImage(mainIcon.getImage());

        JLabel COOKING_TODAY_LBL = imageSetter("what-are-we-cooking.png");
        COOKING_TODAY_LBL.setBounds(32, 32, 345, 40);

        JLabel PERSONALIZED_LBL = imageSetter("personalized-meal-plan.png");
        PERSONALIZED_LBL.setBounds(32, 210, 345, 40);

        JLabel RECIPES_LBL = imageSetter("recipes-to-cook.png");
        RECIPES_LBL.setBounds(718, 22, 159, 40);

        recipeListModel = new DefaultListModel<>(); // Initialize the model
        JList<String> recipeList = new JList<>(recipeListModel); // Create JList with model
        recipeList.setBackground(Color.decode("#eaffef")); // Set background color of JList
        JScrollPane recipeScrollPane = new JScrollPane(recipeList); // Put JList in JScrollPane
        recipeScrollPane.setBounds(718, 90, 200, 300);
        recipeScrollPane.setBorder(null); // Remove border from JScrollPane
        recipeScrollPane.setBackground(Color.decode("#eaffef")); // Set background color of JScrollPane
        contentPane.add(recipeScrollPane);
```

```
        String ADDMEAL_ICON_BTN = "add-meal.png";

        String ADDMEAL_ICON_ENTERED = "add-meal-clicked.png";

        JButton ADDMEAL_BTN = buttonSetter(ADDMEAL_ICON_BTN, ADDMEAL_ICON_ENTERED, ADDMEAL_ICON_BTN);

        ADDMEAL_BTN.setBounds(514, 24, 150, 40);



        String ADDPLANL_ICON_BTN = "add-plan.png";

        String ADDPLANL_ICON_ENTERED = "add-plan-clicked.png";

        JButton ADDPLAN_BTN = buttonSetter(ADDPLANL_ICON_BTN, ADDPLANL_ICON_ENTERED, ADDPLANL_ICON_BTN);

        ADDPLAN_BTN.setBounds(514, 200, 150, 40);



        String BACK_ICON_BTN = "back-exercises.png";

        String BACK_ICON_ENTERED = "back-exercises-clicked.png";

        JButton BACK_BTN = buttonSetter(BACK_ICON_BTN, BACK_ICON_ENTERED, BACK_ICON_BTN);

        BACK_BTN.setBounds(780, 450, 133, 40);

        BACK_BTN.addActionListener(ActionListener -> {

                        LANDING_SIMPLIFIED_UI landingSimplified = new LANDING_SIMPLIFIED_UI();

                        landingSimplified.setVisible(true);

          dispose();

        });



        String PLUS_ICON_BTN = "plus.png";

        String PLUS_ICON_ENTERED = "plus-clicked.png";

        JButton ADDRECIPE_BTN = buttonSetter(PLUS_ICON_BTN, PLUS_ICON_ENTERED, PLUS_ICON_BTN);

        ADDRECIPE_BTN.setBounds(887, 32, 40, 23);

        ADDRECIPE_BTN.addActionListener(new ActionListener() {

            @Override

            public void actionPerformed(ActionEvent e) {

                // Display input dialog to get recipe name

                String recipeName = JOptionPane.showInputDialog(MEAL_PLANNER_UI.this, "Enter Recipe Name:");
```

```
            if (recipeName != null && !recipeName.isEmpty()) {
                // Add the recipe to the list model
                recipeListModel.addElement(recipeName);
            }
        }
    });


    String MONDAY_ICON_BTN = "monday.png";
    String MONDAY_ICON_ENTERED = "monday-clicked.png";
    MONDAY_BTN = buttonSetterToggle(MONDAY_ICON_ENTERED, MONDAY_ICON_BTN, MONDAY_ICON_ENTERED, MONDAY_ICON_BTN);
    MONDAY_BTN.setBounds(70, 275, 121, 23);


    String TUESDAY_ICON_BTN = "tuesday.png";
    String TUESDAY_ICON_ENTERED = "tuesday-clicked.png";
    TUESDAY_BTN = buttonSetterToggle(TUESDAY_ICON_ENTERED, TUESDAY_ICON_BTN, TUESDAY_ICON_ENTERED, TUESDAY_ICON_BTN);
    TUESDAY_BTN.setBounds(210, 275, 121, 23);


    String WEDNESDAY_ICON_BTN = "wednesday.png";
    String WEDNESDAY_ICON_ENTERED = "wednesday-clicked.png";
    WEDNESDAY_BTN = buttonSetterToggle(WEDNESDAY_ICON_ENTERED, WEDNESDAY_ICON_BTN, WEDNESDAY_ICON_ENTERED, WEDNESDAY_ICON_BTN);
    WEDNESDAY_BTN.setBounds(350, 275, 121, 23);


    String THURSDAY_ICON_BTN = "thursday.png";
    String THURSDAY_ICON_ENTERED = "thursday-clicked.png";
    THURSDAY_BTN = buttonSetterToggle(THURSDAY_ICON_ENTERED, THURSDAY_ICON_BTN, THURSDAY_ICON_ENTERED, THURSDAY_ICON_BTN);
    THURSDAY_BTN.setBounds(490, 275, 121, 23);


    String FRIDAY_ICON_BTN = "friday.png";
    String FRIDAY_ICON_ENTERED = "friday-clicked.png";
    FRIDAY_BTN = buttonSetterToggle(FRIDAY_ICON_ENTERED, FRIDAY_ICON_BTN, FRIDAY_ICON_ENTERED, FRIDAY_ICON_BTN);
```

```
        FRIDAY_BTN.setBounds(150, 307, 121, 23);


        String SATURDAY_ICON_BTN = "saturday.png";
        String SATURDAY_ICON_ENTERED = "saturday-clicked.png";
        SATURDAY_BTN = buttonSetterToggle(SATURDAY_ICON_ENTERED,
SATURDAY_ICON_BTN, SATURDAY_ICON_ENTERED, SATURDAY_ICON_BTN);
        SATURDAY_BTN.setBounds(290, 307, 121, 23);


        String SUNDAY_ICON_BTN = "sunday.png";
        String SUNDAY_ICON_ENTERED = "sunday-clicked.png";
        SUNDAY_BTN = buttonSetterToggle(SUNDAY_ICON_ENTERED,
SUNDAY_ICON_BTN, SUNDAY_ICON_ENTERED, SUNDAY_ICON_BTN);
        SUNDAY_BTN.setBounds(430, 307, 121, 23);


        ButtonGroup buttonGroup = new ButtonGroup();
        buttonGroup.add(MONDAY_BTN);
        buttonGroup.add(TUESDAY_BTN);
        buttonGroup.add(WEDNESDAY_BTN);
        buttonGroup.add(THURSDAY_BTN);
        buttonGroup.add(FRIDAY_BTN);
        buttonGroup.add(SATURDAY_BTN);
        buttonGroup.add(SUNDAY_BTN);


        BREAKFAST_LBL = new JLabel("");
        BREAKFAST_LBL.setFont(new Font("Century Gothic", Font.PLAIN, 12));
        BREAKFAST_LBL.setHorizontalAlignment(SwingConstants.CENTER);
        BREAKFAST_LBL.setBounds(70, 419, 159, 59);
        contentPane.add(BREAKFAST_LBL);


        LUNCH_LBL = new JLabel("");
        LUNCH_LBL.setFont(new Font("Century Gothic", Font.PLAIN, 12));
        LUNCH_LBL.setHorizontalAlignment(SwingConstants.CENTER);
        LUNCH_LBL.setBounds(261, 419, 159, 59);
        contentPane.add(LUNCH_LBL);
```

```
        DINNER_LBL = new JLabel("");
        DINNER_LBL.setFont(new Font("Century Gothic", Font.PLAIN, 12));
        DINNER_LBL.setHorizontalAlignment(SwingConstants.CENTER);
        DINNER_LBL.setBounds(452, 419, 159, 59);
        contentPane.add(DINNER_LBL);

        BUTTON_PANNEL = new JPanel();
        BUTTON_PANNEL.setOpaque(false); // Make the panel transparent
        BUTTON_PANNEL.setLayout(new FlowLayout(FlowLayout.LEFT, 5, 5));

        scrollPane = new JScrollPane(BUTTON_PANNEL);
        scrollPane.setOpaque(false); // Make the scroll pane transparent
        scrollPane.getViewport().setOpaque(false); // Make the viewport transparent
        scrollPane.setBorder(null); // Remove the border
        scrollPane.setBounds(30, 78, 634, 100);

scrollPane.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_
NEEDED);
        contentPane.add(scrollPane);

        JLabel BREAKFAST_TXT_LBL = imageSetter("for-breakfast.png");
        BREAKFAST_TXT_LBL.setBounds(70, 366, 159, 42);

        JLabel LUNCH_TXT_LBL = imageSetter("for-lunch.png");
        LUNCH_TXT_LBL.setBounds(261, 366, 159, 42);

        JLabel DINNER_TXT_LBL = imageSetter("for-dinner.png");
        DINNER_TXT_LBL.setBounds(452, 365, 159, 42);

        ADDMEAL_BTN.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Display input dialogs to get food name, ingredients, and notes
```

```
        JTextField foodNameField = new JTextField();
        JTextArea ingredientsArea = new JTextArea();
        JTextField notesField = new JTextField();

        JPanel panel = new JPanel(new GridLayout(0, 1));
        panel.add(new JLabel("Food Name:"));
        panel.add(foodNameField);
        panel.add(new JLabel("Ingredients:"));
        panel.add(new JScrollPane(ingredientsArea)); // Use JScrollPane for multi-line input
        panel.add(new JLabel("Notes:"));
        panel.add(notesField);

        int result = JOptionPane.showConfirmDialog(MEAL_PLANNER_UI.this, panel, "Enter Food Details",
            JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
        if (result == JOptionPane.OK_OPTION) {
            // Get input values
            String foodName = foodNameField.getText();
            String ingredients = ingredientsArea.getText();
            String notes = notesField.getText();

            // Load the default image icon
            String defaultImagePath = "add-ulam.png";
            ImageIcon icon = new ImageIcon(getClass().getClassLoader().getResource(defaultImagePath));
            // Resize the icon to 90x70 pixels
            Image image = icon.getImage().getScaledInstance(90, 70, Image.SCALE_SMOOTH);
            icon = new ImageIcon(image);

            // Create a new button with food name and icon
            JButton newButton = new JButton(foodName);
            newButton.setPreferredSize(new Dimension(90, 70));
            newButton.setContentAreaFilled(false); // Make button transparent
```

```
                newButton.setBorderPainted(false); // Remove border
                if (icon != null) {
                    newButton.setIcon(icon); // Set the icon
                }

                newButton.addActionListener(new ActionListener() {
                    @Override
                    public void actionPerformed(ActionEvent e) {
                        // Display food details and delete option in a message dialog when the button is clicked
                        String message = "Food Name: " + foodName + "\n\nIngredients: \n" + ingredients + "\n\nNotes: " + notes;
                        int option = JOptionPane.showConfirmDialog(MEAL_PLANNER_UI.this, message + "\n\nDo you want to delete this meal?", "Meal Details", JOptionPane.YES_NO_OPTION);

                        if (option == JOptionPane.YES_OPTION) {
                            // Remove the button from BUTTON_PANNEL
                            BUTTON_PANNEL.remove(newButton);
                            BUTTON_PANNEL.revalidate();
                            BUTTON_PANNEL.repaint();
                        }
                    }
                });

                // Add the new button to BUTTON_PANNEL
                BUTTON_PANNEL.add(newButton);
                BUTTON_PANNEL.revalidate();
                BUTTON_PANNEL.repaint();
            }
        }
    });

    ADDPLAN_BTN.addActionListener(new ActionListener() {
        @Override
```

```
        public void actionPerformed(ActionEvent e) {

            String[] daysOfWeek = {"Monday", "Tuesday", "Wednesday", "Thursday",
"Friday", "Saturday", "Sunday"};

            String selectedDay = (String)
JOptionPane.showInputDialog(MEAL_PLANNER_UI.this,

                    "Select the day of the plan:", "Day Selection",

                    JOptionPane.QUESTION_MESSAGE, null, daysOfWeek,
daysOfWeek[0]);


            if (selectedDay != null && !selectedDay.isEmpty()) {

                JTextField breakfastField = new JTextField();

                JTextField lunchField = new JTextField();

                JTextField dinnerField = new JTextField();


                JPanel panel = new JPanel(new GridLayout(0, 1));

                panel.add(new JLabel("Breakfast:"));

                panel.add(breakfastField);

                panel.add(new JLabel("Lunch:"));

                panel.add(lunchField);

                panel.add(new JLabel("Dinner:"));

                panel.add(dinnerField);


                int result = JOptionPane.showConfirmDialog(MEAL_PLANNER_UI.this,
panel,

                        "Enter Meals for " + selectedDay,
JOptionPane.OK_CANCEL_OPTION);


                if (result == JOptionPane.OK_OPTION) {

                    String breakfast = breakfastField.getText();

                    String lunch = lunchField.getText();

                    String dinner = dinnerField.getText();


                    // Store meal plan for Monday

                    if (selectedDay.equals("Monday")) {

                        mondayBreakfast = breakfast;

                        mondayLunch = lunch;
```

```
                mondayDinner = dinner;
            } else if (selectedDay.equals("Tuesday")) {
                tuesdayBreakfast = breakfast;
                tuesdayLunch = lunch;
                tuesdayDinner = dinner;
            } else if (selectedDay.equals("Wednesday")) {
                wednesdayBreakfast = breakfast;
                wednesdayLunch = lunch;
                wednesdayDinner = dinner;
            } else if (selectedDay.equals("Thursday")) {
                thursdayBreakfast = breakfast;
                thursdayLunch = lunch;
                thursdayDinner = dinner;
            } else if (selectedDay.equals("Friday")) {
                fridayBreakfast = breakfast;
                fridayLunch = lunch;
                fridayDinner = dinner;
            } else if (selectedDay.equals("Saturday")) {
                saturdayBreakfast = breakfast;
                saturdayLunch = lunch;
                saturdayDinner = dinner;
            } else if (selectedDay.equals("Sunday")) {
                sundayBreakfast = breakfast;
                sundayLunch = lunch;
                sundayDinner = dinner;
            }
        }
    }
}
});

MONDAY_BTN.addActionListener(new ActionListener() {
    @Override
```

```
    public void actionPerformed(ActionEvent e) {
        if (MONDAY_BTN.isSelected()) {
            BREAKFAST_LBL.setText(mondayBreakfast);
            LUNCH_LBL.setText(mondayLunch);
            DINNER_LBL.setText(mondayDinner);
        }
    }
});


TUESDAY_BTN.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if (TUESDAY_BTN.isSelected()) {
            BREAKFAST_LBL.setText(tuesdayBreakfast);
            LUNCH_LBL.setText(tuesdayLunch);
            DINNER_LBL.setText(tuesdayDinner);
        }
    }
});

WEDNESDAY_BTN.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if (WEDNESDAY_BTN.isSelected()) {
            BREAKFAST_LBL.setText(wednesdayBreakfast);
            LUNCH_LBL.setText(wednesdayLunch);
            DINNER_LBL.setText(wednesdayDinner);
        }
    }
});

THURSDAY_BTN.addActionListener(new ActionListener() {
```

```
    @Override
    public void actionPerformed(ActionEvent e) {
        if (THURSDAY_BTN.isSelected()) {
            BREAKFAST_LBL.setText(thursdayBreakfast);
            LUNCH_LBL.setText(thursdayLunch);
            DINNER_LBL.setText(thursdayDinner);
        }
    }
});

FRIDAY_BTN.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if (FRIDAY_BTN.isSelected()) {
            BREAKFAST_LBL.setText(fridayBreakfast);
            LUNCH_LBL.setText(fridayLunch);
            DINNER_LBL.setText(fridayDinner);
        }
    }
});

SATURDAY_BTN.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if (SATURDAY_BTN.isSelected()) {
            BREAKFAST_LBL.setText(saturdayBreakfast);
            LUNCH_LBL.setText(saturdayLunch);
            DINNER_LBL.setText(saturdayDinner);
        }
    }
});

SUNDAY_BTN.addActionListener(new ActionListener() {
```

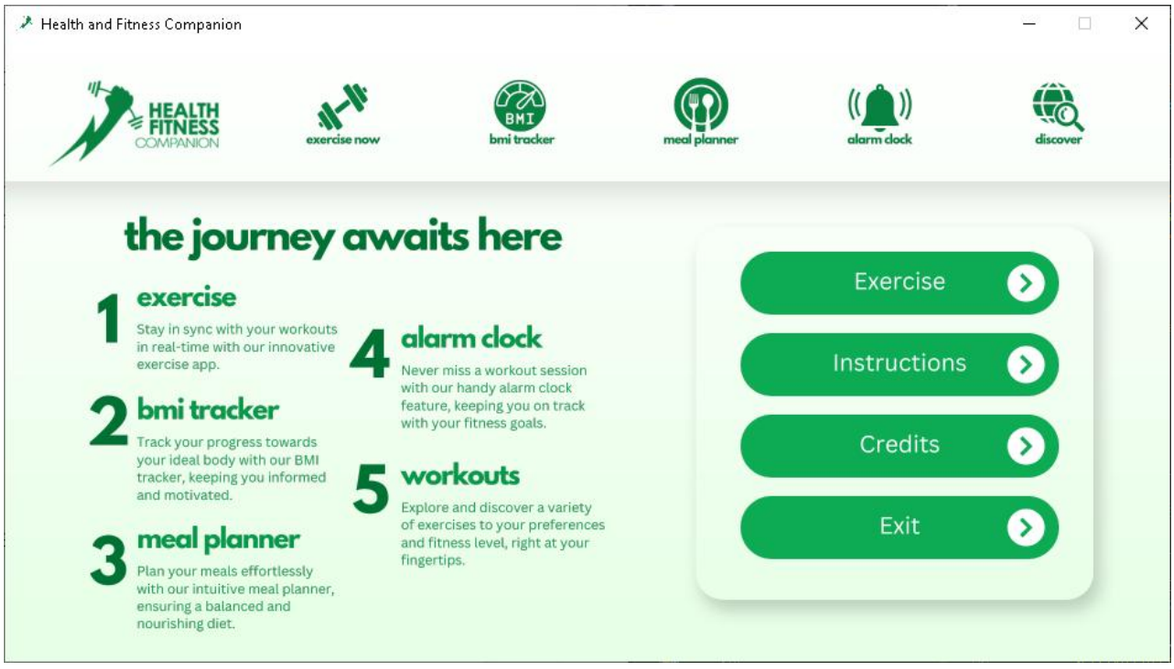
```
            @Override
            public void actionPerformed(ActionEvent e) {
                if (SUNDAY_BTN.isSelected()) {
                    BREAKFAST_LBL.setText(sundayBreakfast);
                    LUNCH_LBL.setText(sundayLunch);
                    DINNER_LBL.setText(sundayDinner);
                }
            }
        });

        ImageIcon backgroundImageIcon = new ImageIcon(getClass().getClassLoader().getResource("MEAL-PLANNER-BG.png"));
        JLabel backgroundImageLabel = new JLabel(backgroundImageIcon);
        backgroundImageLabel.setBounds(0, 0, 960, 540);
        contentPane.add(backgroundImageLabel);
    }

      public JLabel imageSetter(String imageLocation) {
        ImageIcon imageIcon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));
        JLabel label = new JLabel(imageIcon);
        contentPane.add(label);
        return label;
    }

    public JButton buttonSetter(String imageLocation, String enteredIconLocation, String exitedIconLocation) {
        ImageIcon icon = new ImageIcon(getClass().getClassLoader().getResource(imageLocation));
        JButton button = new JButton(icon);
        button.setBorder(null);
        button.setOpaque(false);
        button.setContentAreaFilled(false);
        button.setBorderPainted(false);
        button.setSize(icon.getIconWidth(), icon.getIconHeight());
```

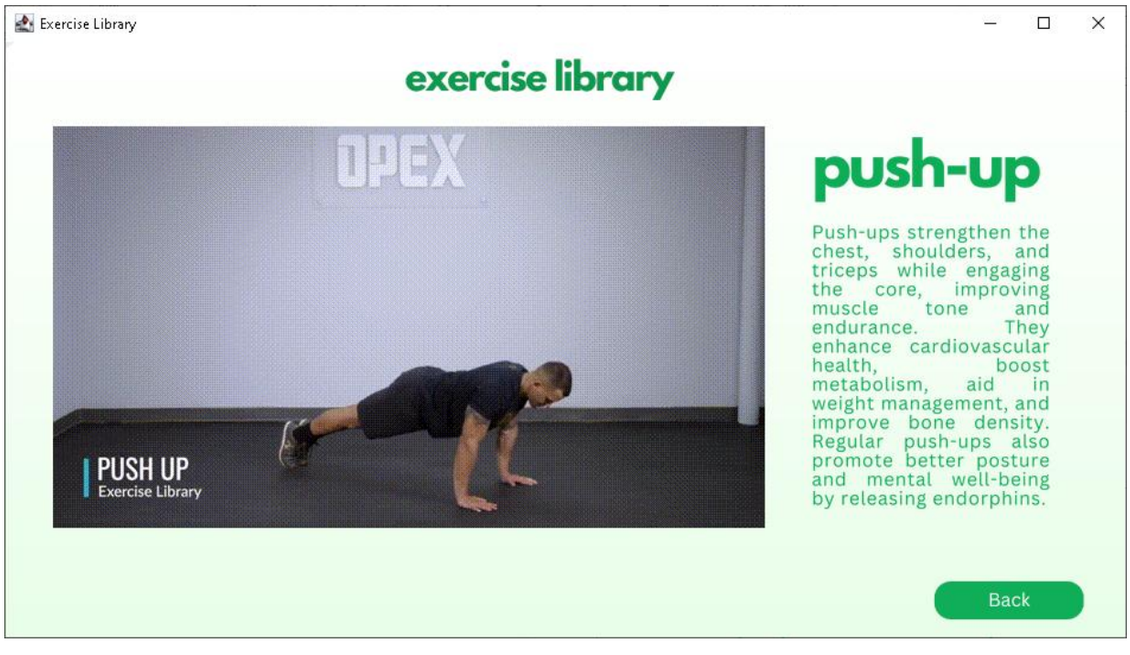
```
        button.addMouseListener(getMouseListener(enteredIconLocation,
exitedIconLocation));

        contentPane.add(button);

        return button;

    }


    private MouseAdapter getMouseListener(String enteredIconLocation, String
exitedIconLocation) {

        return new MouseAdapter() {

            public void mouseEntered(MouseEvent e) {

                ImageIcon enteredIcon = new
ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));

                ((JButton) e.getSource()).setIcon(enteredIcon);

            }



            public void mouseExited(MouseEvent e) {

                ImageIcon exitedIcon = new
ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));

                ((JButton) e.getSource()).setIcon(exitedIcon);

            }

        };

    }


    public JToggleButton buttonSetterToggle(String imageLocation, String
enteredIconLocation, String exitedIconLocation, String toggledIconLocation) {

        ImageIcon icon = new
ImageIcon(getClass().getClassLoader().getResource(imageLocation));

        JToggleButton button = new JToggleButton(icon);

        button.setBorder(null);

        button.setOpaque(false);

        button.setContentAreaFilled(false);

        button.setBorderPainted(false);

        button.setSize(icon.getIconWidth(), icon.getIconHeight());



        // Set the default icon

        button.setIcon(new
ImageIcon(getClass().getClassLoader().getResource(imageLocation)));
```

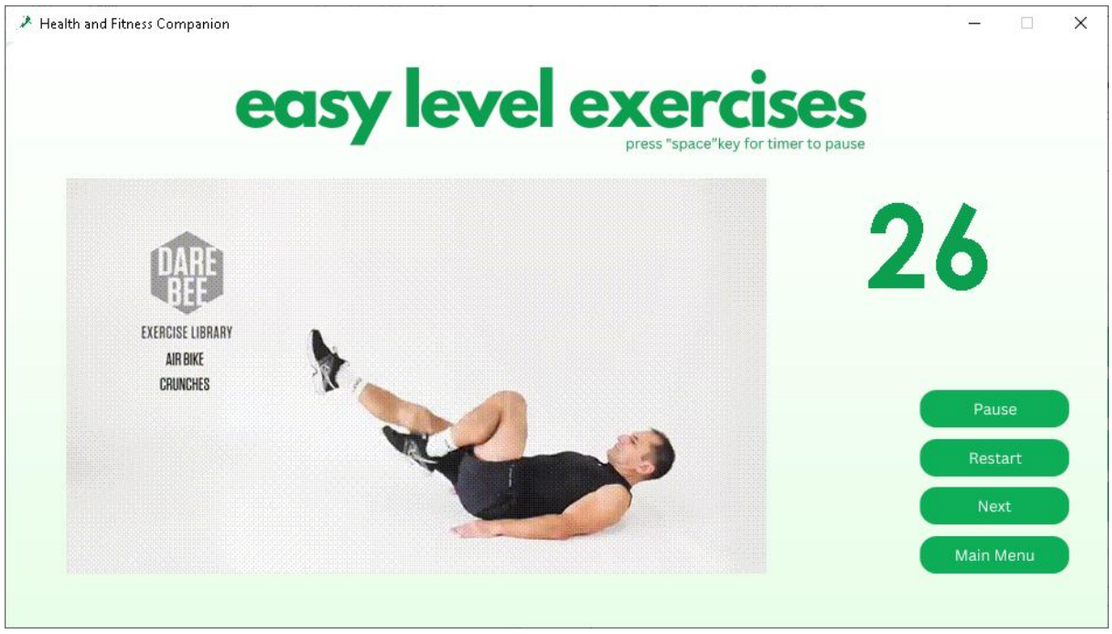
```
        // Create an ItemListener to listen for state changes in the toggle button
        button.addItemListener(new ItemListener() {
            @Override
            public void itemStateChanged(ItemEvent e) {
                if (e.getStateChange() == ItemEvent.SELECTED) {
                    // Button is toggled
                    button.setIcon(new
ImageIcon(getClass().getClassLoader().getResource(toggledIconLocation)));
                } else {
                    // Button is untoggled
                    button.setIcon(new
ImageIcon(getClass().getClassLoader().getResource(imageLocation)));
                }
            }
        });

        // Add mouse listener for hover effect
        button.addMouseListener(getMouseListenerToggle(enteredIconLocation,
exitedIconLocation, toggledIconLocation));

        contentPane.add(button);
        return button;
    }

    private MouseAdapter getMouseListenerToggle(String enteredIconLocation, String
exitedIconLocation, String toggledIconLocation) {
        return new MouseAdapter() {
            public void mouseEntered(MouseEvent e) {
                JToggleButton button = (JToggleButton) e.getSource();
                if (button.isSelected()) {
                    ImageIcon enteredIcon = new
ImageIcon(getClass().getClassLoader().getResource(toggledIconLocation));
                    button.setIcon(enteredIcon);
                } else {
```

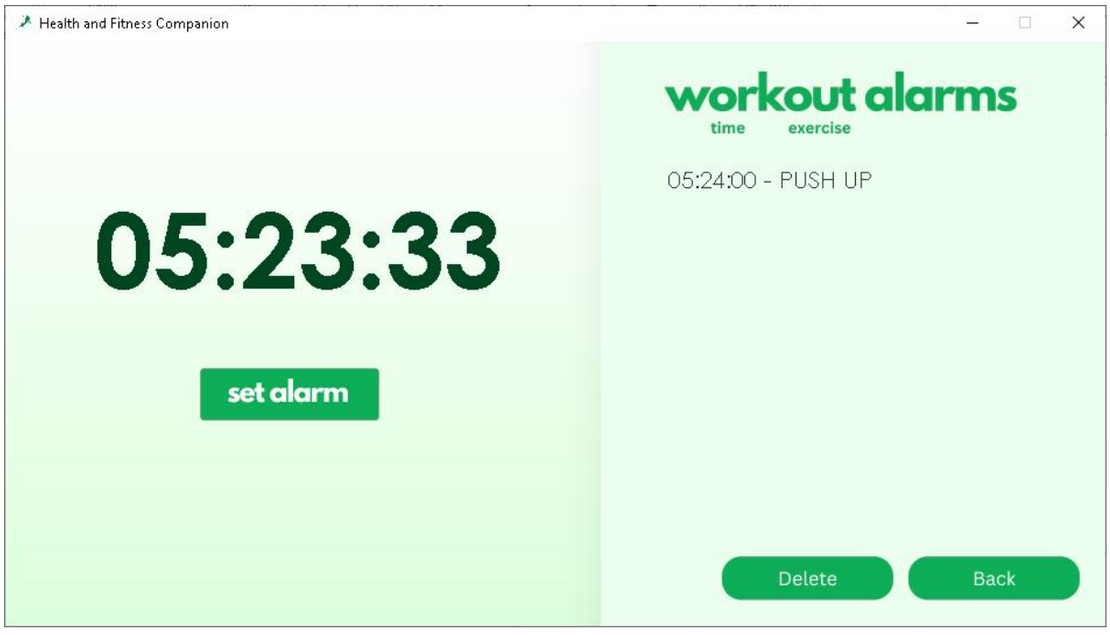
```
                ImageIcon enteredIcon = new
ImageIcon(getClass().getClassLoader().getResource(enteredIconLocation));
                button.setIcon(enteredIcon);
            }
        }

        public void mouseExited(MouseEvent e) {
            JToggleButton button = (JToggleButton) e.getSource();
            if (button.isSelected()) {
                ImageIcon exitedIcon = new
ImageIcon(getClass().getClassLoader().getResource(toggledIconLocation));
                button.setIcon(exitedIcon);
            } else {
                ImageIcon exitedIcon = new
ImageIcon(getClass().getClassLoader().getResource(exitedIconLocation));
                button.setIcon(exitedIcon);
            }
        }
    };
  }
}
```

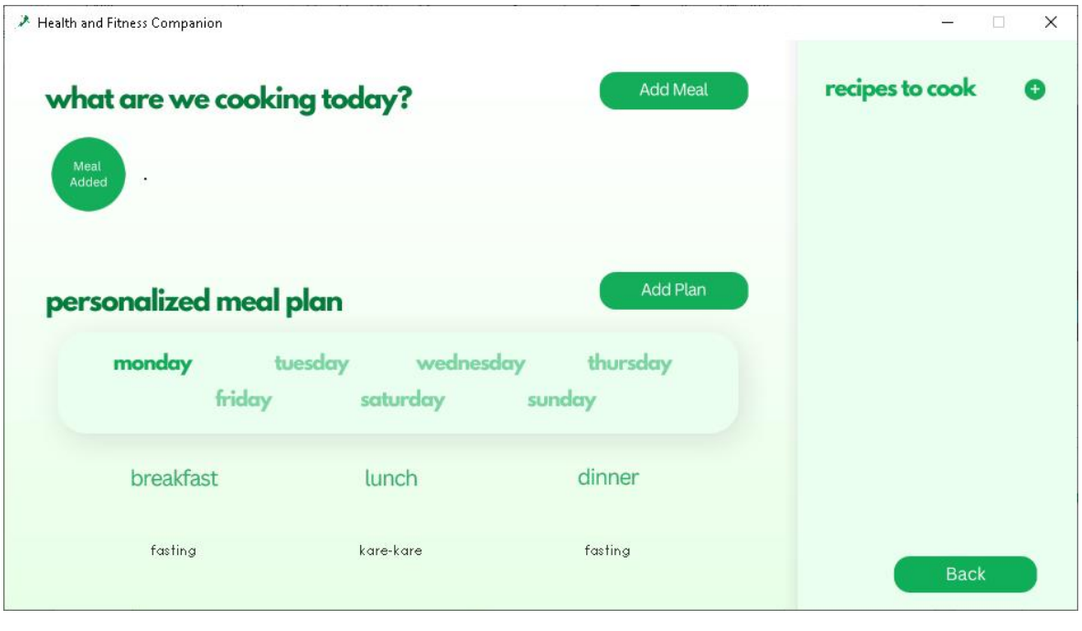
VI. SCREENSHOT OF ACTUAL USAGE

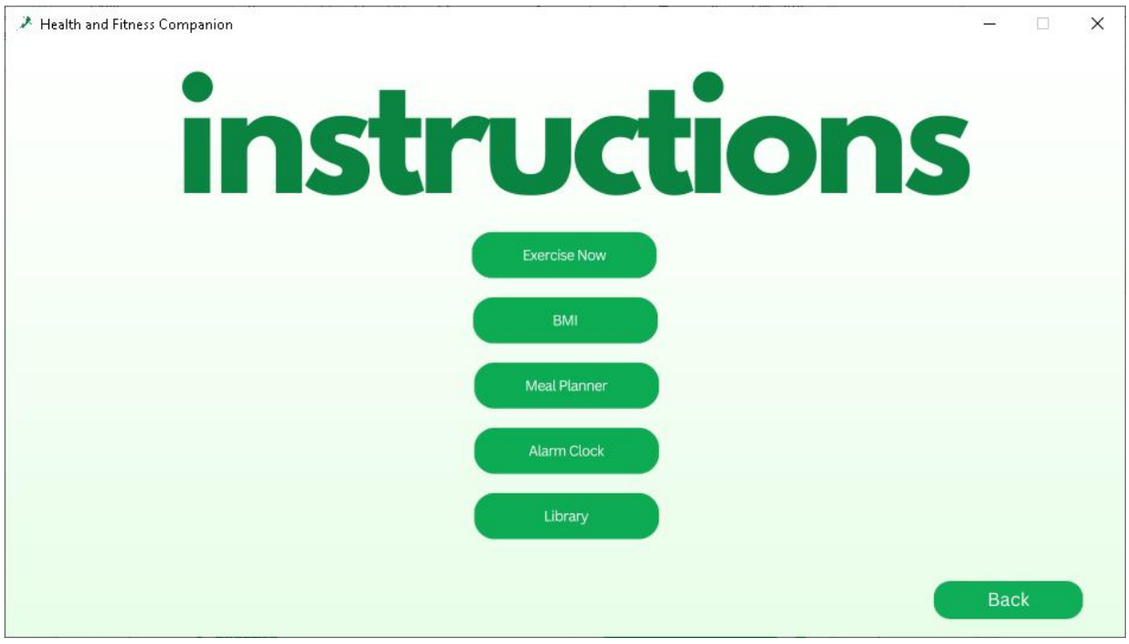


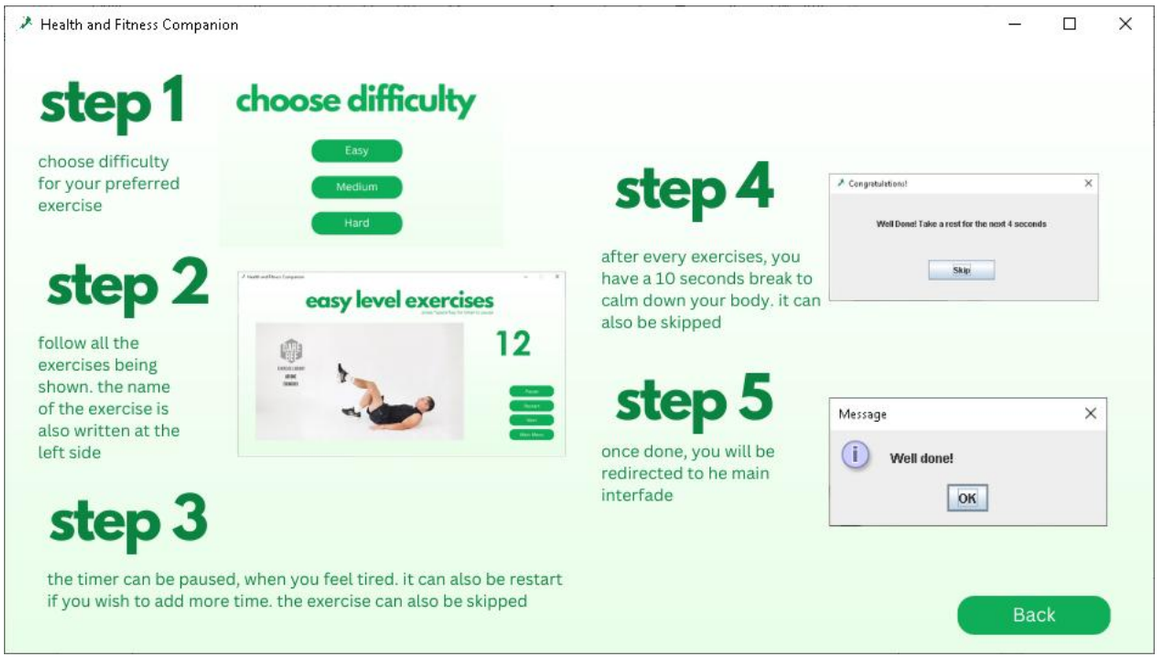


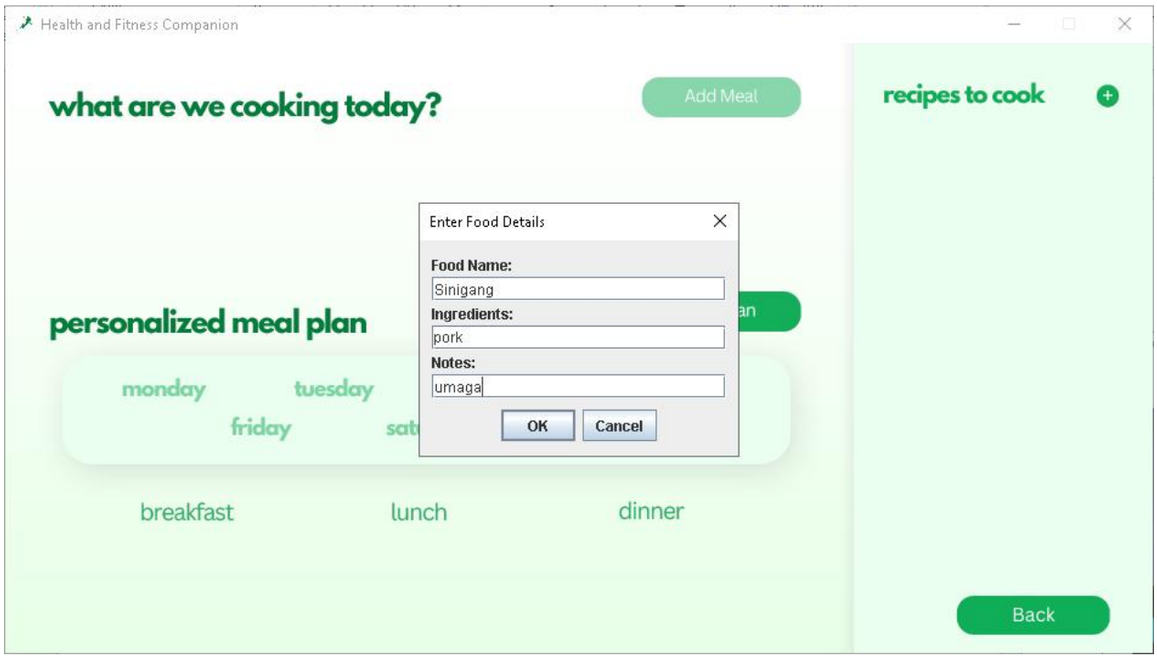

Health and Fitness Companion
easy level exercises
press "space"key for timer to pause
DARE BEE
EXERCISE LIBRARY
AIR BIKE
CRUNCHES
26
Pause
Restart
Next
Main Menu


Health and Fitness Companion
05:23:33
set alarm
workout alarms
time
exercise
05:24:00 - PUSH UP
Delete
Back


Health and Fitness Companion
what are we cooking today?
Add Meal
Meal Added
recipes to cook
personalized meal plan
Add Plan
monday
tuesday
wednesday
thursday
friday
saturday
sunday
breakfast
lunch
dinner
fasting
kare-kare
fasting
Back


Health and Fitness Companion
instructions
Exercise Now
BMI
Meal Planner
Alarm Clock
Library
Back


Health and Fitness Companion
step 1
choose difficulty
choose difficulty for your preferred exercise
Easy
Medium
Hard
step 2
follow all the exercises being shown. the name of the exercise is also written at the left side
easy level exercises
12
step 3
the timer can be paused, when you feel tired. it can also be restart if you wish to add more time. the exercise can also be skipped
step 4
after every exercises, you have a 10 seconds break to calm down your body. it can also be skipped
Congratulations!
Well Done! Take a rest for the next 4 seconds
Skip
step 5
once done, you will be redirected to he main interfade
Message
Well done!
OK
Back


Health and Fitness Companion
what are we cooking today?
Add Meal
recipes to cook
personalized meal plan
monday
tuesday
friday
breakfast
lunch
dinner
Enter Food Details
Food Name:
Sinigang
Ingredients:
pork
Notes:
umaga
OK
Cancel
Back