**Name: Koushik Das, Reg:2347232, Exam:Java Practical**

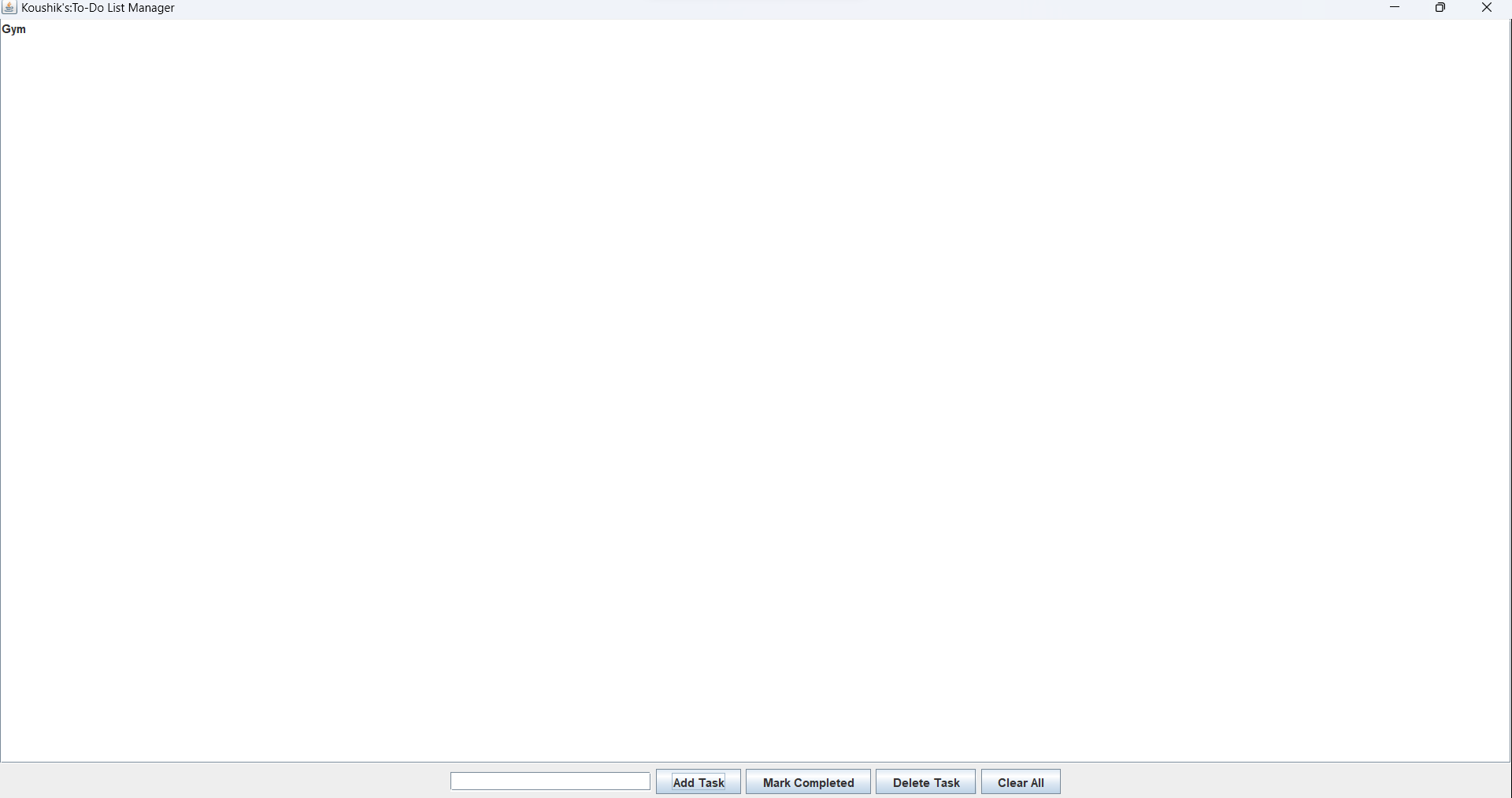
Q1:- Graphical User Interface (Swing):

Code:-

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
  
public class Q1 extends JFrame {  
 private DefaultListModel<String> taskListModel;  
 private JList<String> taskList;  
 private JTextField taskTextField;  
  
 public Q1() {  
 initializeUI();  
 }  
  
 private void initializeUI() {  
 setTitle("Koushik's:To-Do List Manager");  
 setSize(400, 300);  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 setLayout(new BorderLayout());  
  
 taskListModel = new DefaultListModel<>();  
 taskList = new JList<>(taskListModel);  
 JScrollPane scrollPane = new JScrollPane(taskList);  
 add(scrollPane, BorderLayout.*CENTER*);  
  
 JPanel inputPanel = new JPanel();  
 inputPanel.setLayout(new FlowLayout());  
  
 taskTextField = new JTextField(20);  
 JButton addButton = new JButton("Add Task");  
 JButton completeButton = new JButton("Mark Completed");  
 JButton deleteButton = new JButton("Delete Task");  
 JButton clearButton = new JButton("Clear All");  
  
 addButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 addTask();  
 }  
 });  
  
 completeButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 markCompletedTask();  
 }  
 });  
  
 deleteButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 deleteTask();  
 }  
 });  
  
 clearButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 clearAllTasks();  
 }  
 });  
  
 inputPanel.add(taskTextField);  
 inputPanel.add(addButton);  
 inputPanel.add(completeButton);  
 inputPanel.add(deleteButton);  
 inputPanel.add(clearButton);  
  
 add(inputPanel, BorderLayout.*SOUTH*);  
  
 setLocationRelativeTo(null);  
 setVisible(true);  
 }  
  
 private void addTask() {  
 String task = taskTextField.getText().trim();  
 if (!task.isEmpty()) {  
 taskListModel.addElement(task);  
 taskTextField.setText("");  
 }  
 }  
  
 private void markCompletedTask() {  
 int[] selectedIndices = taskList.getSelectedIndices();  
 for (int i = selectedIndices.length - 1; i >= 0; i--) {  
 int selectedIndex = selectedIndices[i];  
 String completedTask = taskListModel.get(selectedIndex);  
 taskListModel.remove(selectedIndex);  
 taskListModel.addElement("[Completed] " + completedTask);  
 }  
 }  
  
 private void deleteTask() {  
 int[] selectedIndices = taskList.getSelectedIndices();  
 for (int i = selectedIndices.length - 1; i >= 0; i--) {  
 int selectedIndex = selectedIndices[i];  
 taskListModel.remove(selectedIndex);  
 }  
 }  
  
 private void clearAllTasks() {  
 taskListModel.removeAllElements();  
 }  
  
 public static void main(String[] args) {  
 SwingUtilities.*invokeLater*(new Runnable() {  
 @Override  
 public void run() {  
 new Q1();  
 }  
 });  
 }  
}

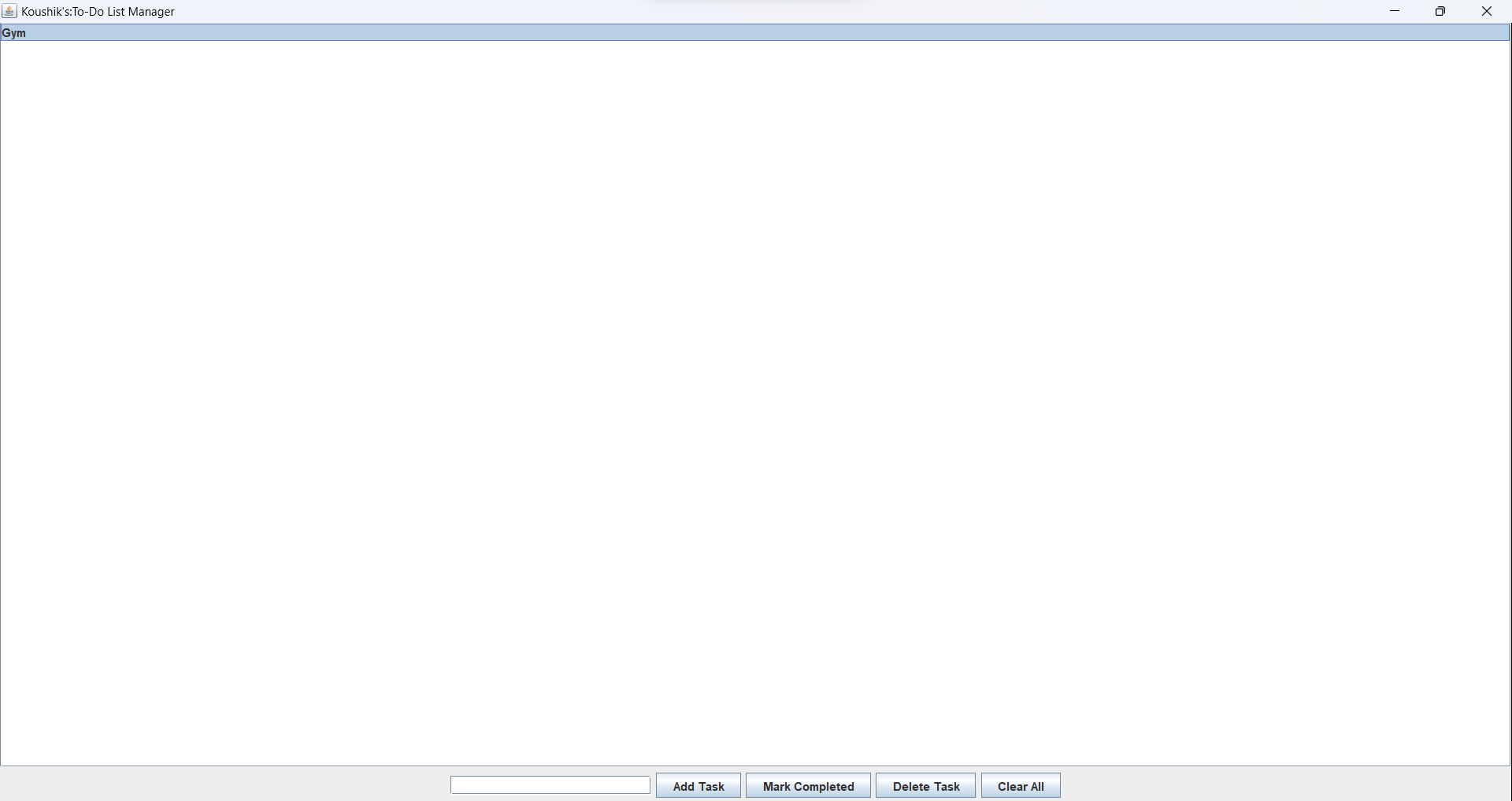
Outputs:-

For Adding:

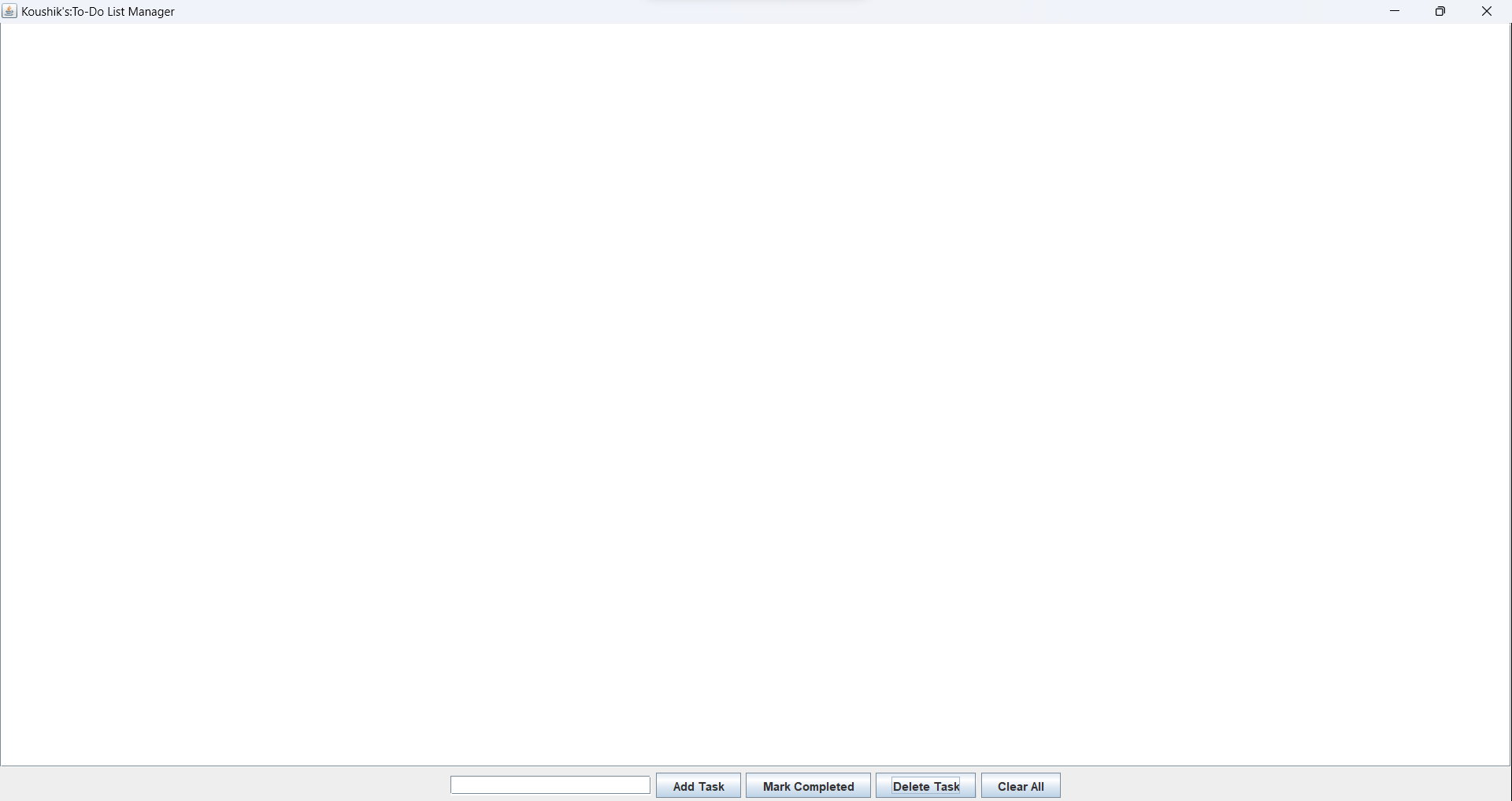


For Delete Task:

1.First Select the Task.

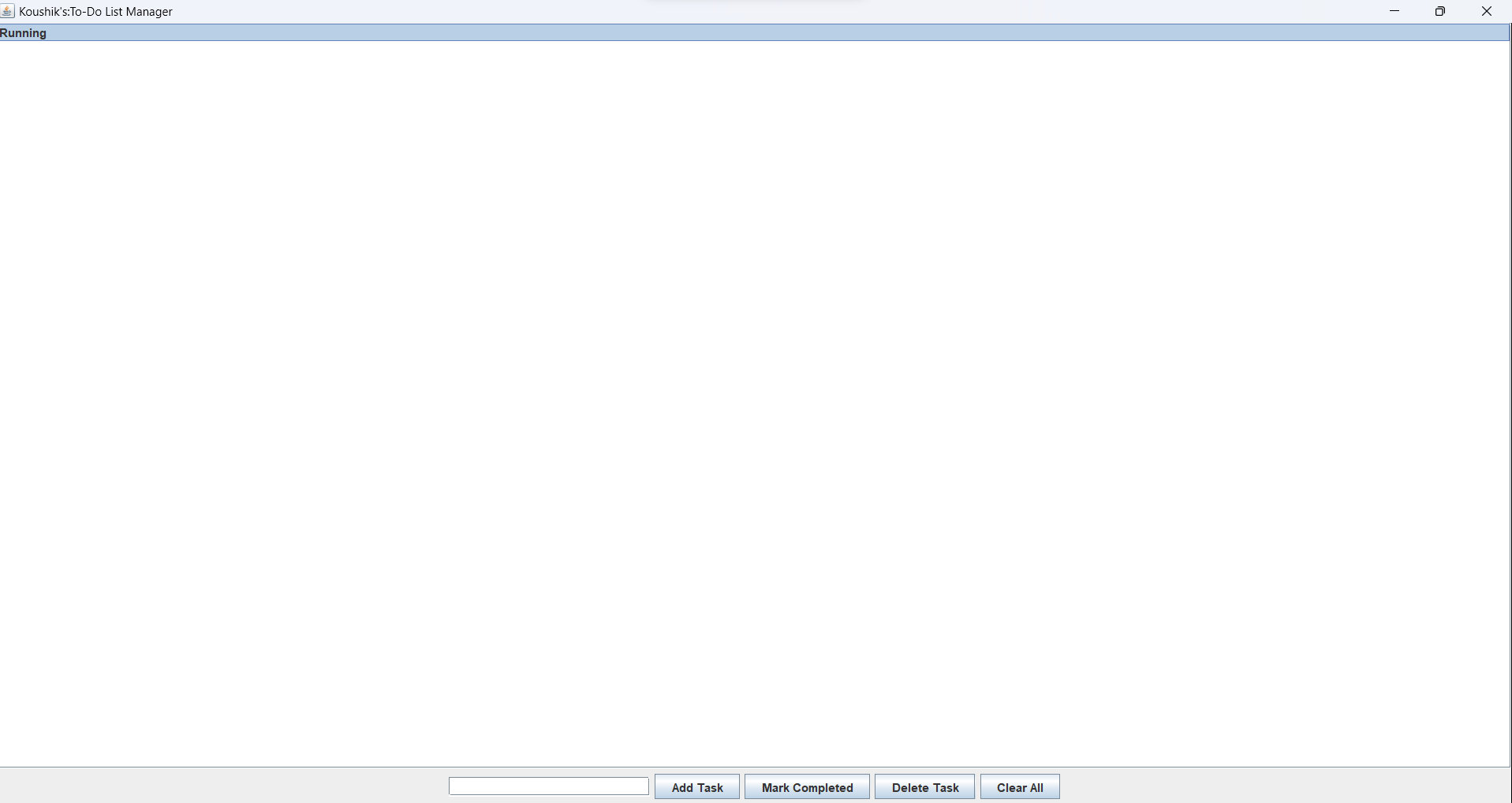


2.Then Delete the task.

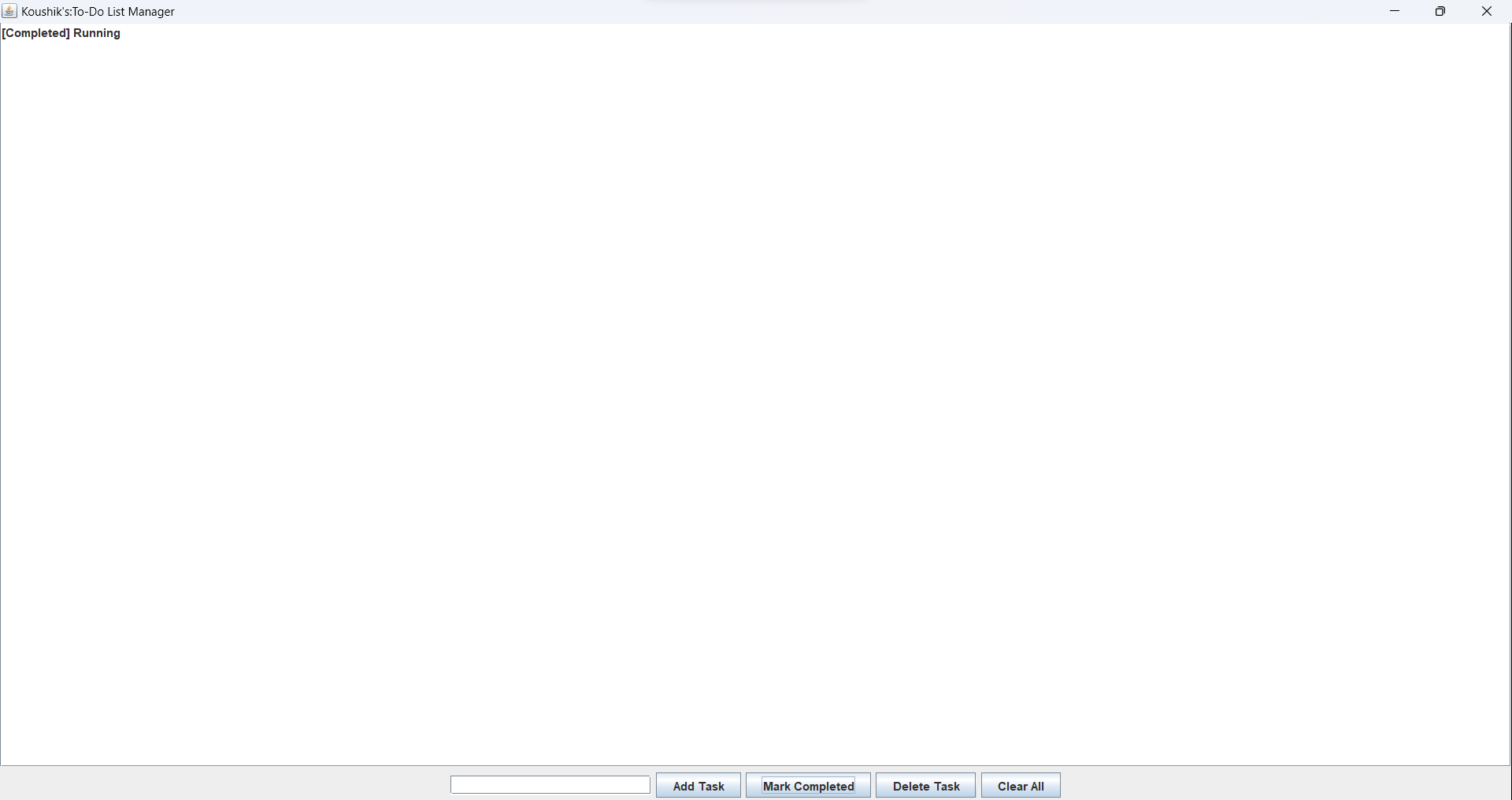


For Mark as Complete:

1.First select the task.

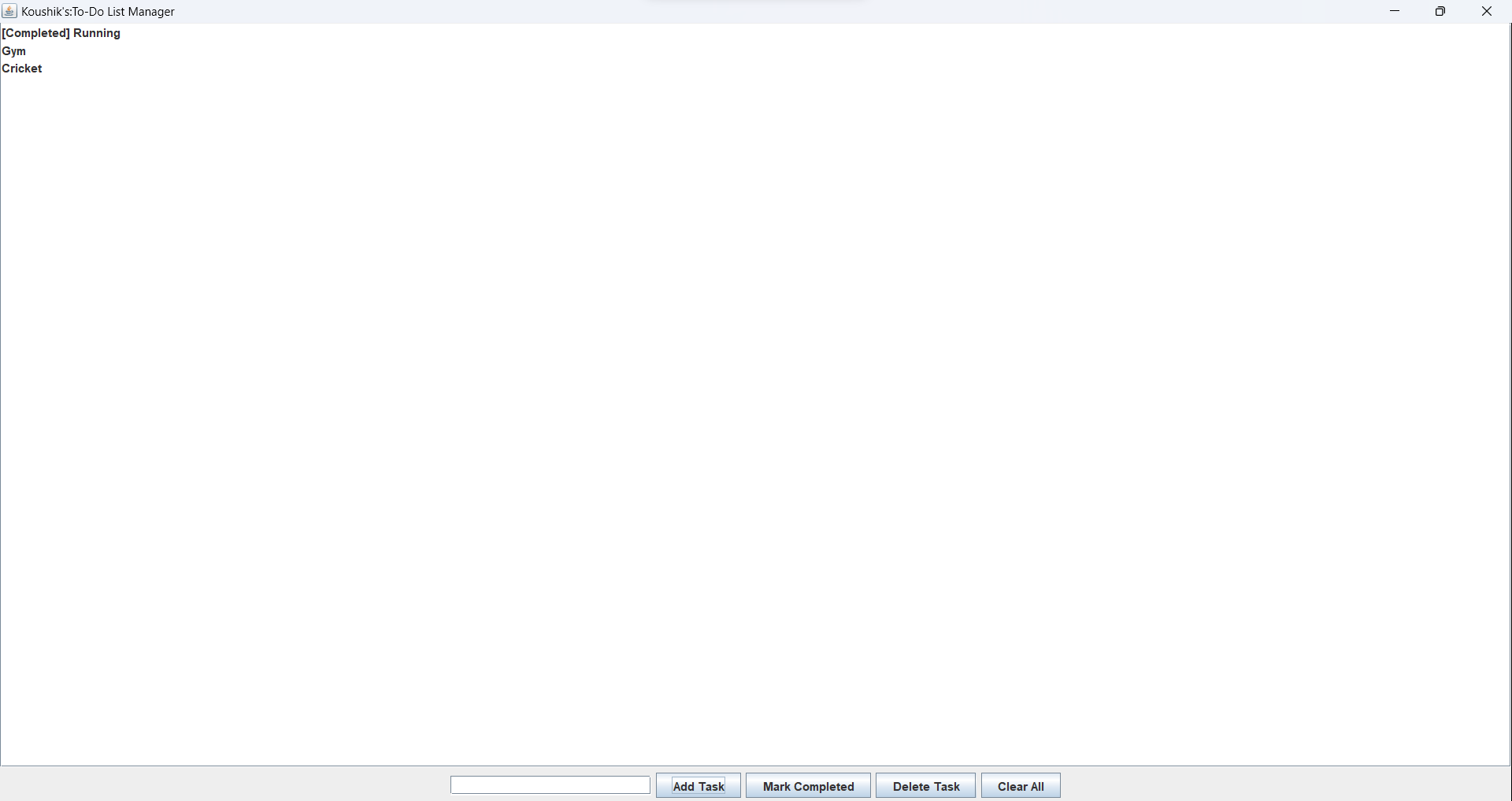


2.Then click Mark as completed.

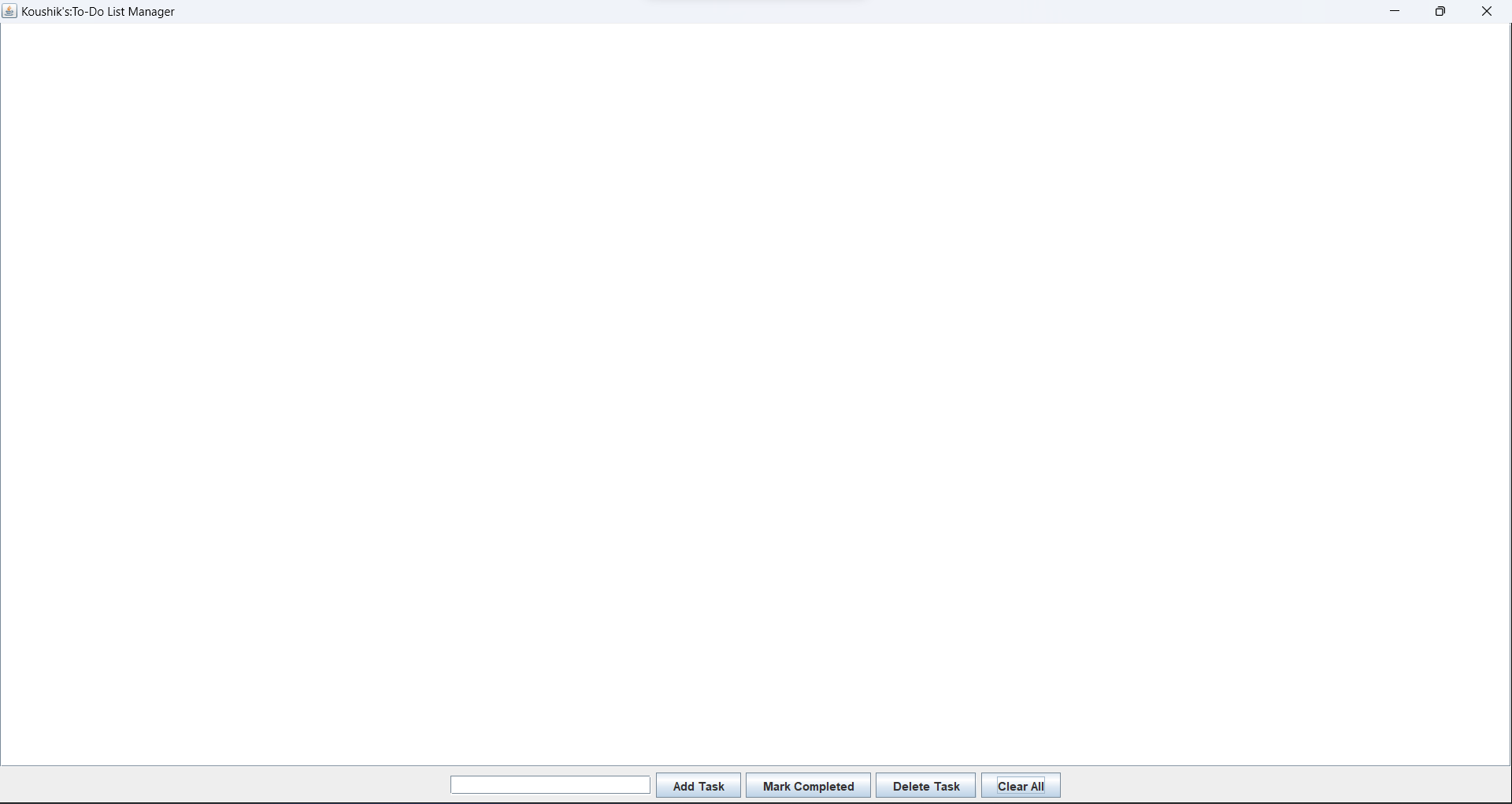


For Clearing the task;

1.list of the task are displayed.



2.Click Clear all.



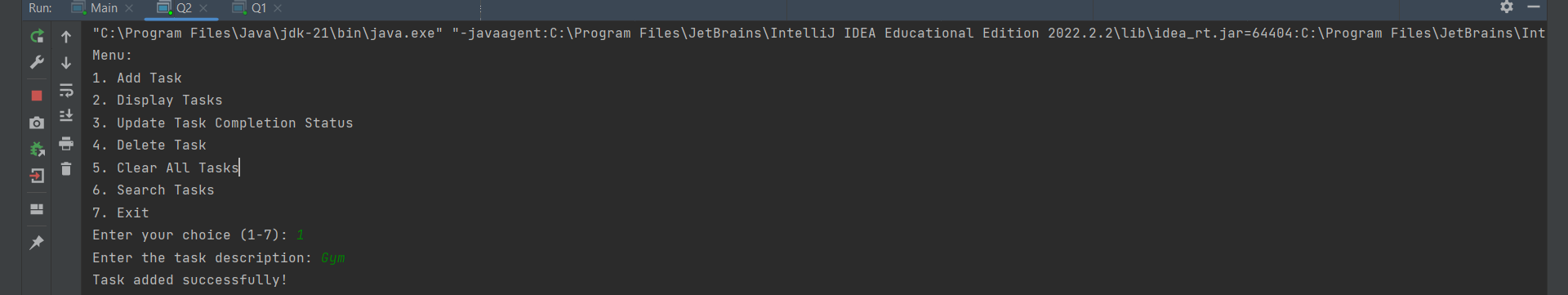
Q2. JDBC Connectivity

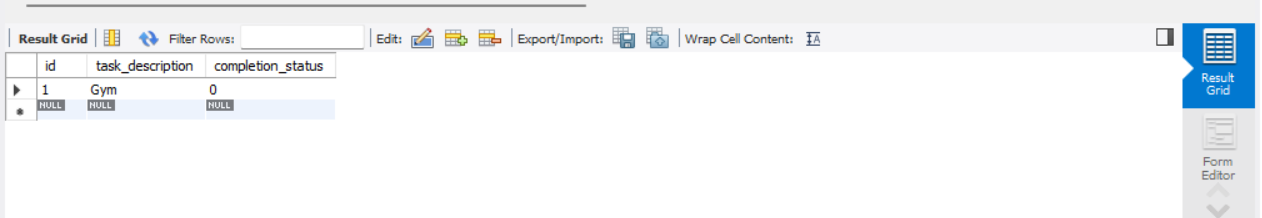
Code :-

import java.sql.\*;  
import java.util.Scanner;  
  
public class Q2 {  
  
 private static final String *JDBC\_URL* = "jdbc:mysql://localhost:3306/kd";  
 private static final String *USER* = "root";  
 private static final String *PASSWORD* = "Puchu9831!";  
 private static Connection *connection*;  
  
 static {  
 try {  
 *connection* = DriverManager.*getConnection*(*JDBC\_URL*, *USER*, *PASSWORD*);  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void addTask(String taskDescription) {  
 try {  
 String query = "INSERT INTO tasks (task\_description, completion\_status) VALUES (?, ?)";  
 try (PreparedStatement preparedStatement = *connection*.prepareStatement(query)) {  
 preparedStatement.setString(1, taskDescription);  
 preparedStatement.setBoolean(2, false); // Initialize completion status as false  
 preparedStatement.executeUpdate();  
 System.*out*.println("Task added successfully!");  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void getAllTasks() {  
 try {  
 String query = "SELECT \* FROM tasks";  
 try (Statement statement = *connection*.createStatement();  
 ResultSet resultSet = statement.executeQuery(query)) {  
 while (resultSet.next()) {  
 int id = resultSet.getInt("id");  
 String taskDescription = resultSet.getString("task\_description");  
 boolean completionStatus = resultSet.getBoolean("completion\_status");  
 System.*out*.println("ID: " + id + ", Task: " + taskDescription +  
 ", Completed: " + (completionStatus ? "Yes" : "No"));  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void updateTaskCompletionStatus(int taskId, boolean completionStatus) {  
 try {  
 String query = "UPDATE tasks SET completion\_status = ? WHERE id = ?";  
 try (PreparedStatement preparedStatement = *connection*.prepareStatement(query)) {  
 preparedStatement.setBoolean(1, completionStatus);  
 preparedStatement.setInt(2, taskId);  
 preparedStatement.executeUpdate();  
 System.*out*.println("Updated successfully!");  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void deleteTask(int taskId) {  
 try {  
 String query = "DELETE FROM tasks WHERE id = ?";  
 try (PreparedStatement preparedStatement = *connection*.prepareStatement(query)) {  
 preparedStatement.setInt(1, taskId);  
 preparedStatement.executeUpdate();  
 System.*out*.println("Deleted successfully!");  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void clearAllTasks() {  
 try {  
 String query = "DELETE FROM tasks";  
 try (PreparedStatement preparedStatement = *connection*.prepareStatement(query)) {  
 preparedStatement.executeUpdate();  
 System.*out*.println("All tasks cleared successfully!");  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void searchTasks(String keyword) {  
 try {  
 String query = "SELECT \* FROM tasks WHERE task\_description LIKE ?";  
 try (PreparedStatement preparedStatement = *connection*.prepareStatement(query)) {  
 preparedStatement.setString(1, "%" + keyword + "%");  
 try (ResultSet resultSet = preparedStatement.executeQuery()) {  
 while (resultSet.next()) {  
 int id = resultSet.getInt("id");  
 String taskDescription = resultSet.getString("task\_description");  
 boolean completionStatus = resultSet.getBoolean("completion\_status");  
 System.*out*.println("ID: " + id + ", Task: " + taskDescription +  
 ", Completed: " + (completionStatus ? "Yes" : "No"));  
 }  
 }  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void closeConnection() {  
 try {  
 if (*connection* != null && !*connection*.isClosed()) {  
 *connection*.close();  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 int choice;  
  
 do {  
 // Display menu  
 System.*out*.println("Menu:");  
 System.*out*.println("1. Add Task");  
 System.*out*.println("2. Display Tasks");  
 System.*out*.println("3. Update Task Completion Status");  
 System.*out*.println("4. Delete Task");  
 System.*out*.println("5. Clear All Tasks");  
 System.*out*.println("6. Search Tasks");  
 System.*out*.println("7. Exit");  
 System.*out*.print("Enter your choice (1-7): ");  
  
 // Get user input  
 choice = sc.nextInt();  
  
 // Process user input using switch statement  
 switch (choice) {  
 case 1:  
 // Add task  
 System.*out*.print("Enter the task description: ");  
 String taskDescription = sc.next();  
 *addTask*(taskDescription);  
 break;  
  
 case 2:  
 // Display tasks  
 *getAllTasks*();  
 break;  
  
 case 3:  
 // Update task completion status  
 System.*out*.print("Enter the task ID: ");  
 int taskId = sc.nextInt();  
 System.*out*.print("Enter the new completion status (true/false): ");  
 boolean completionStatus = sc.nextBoolean();  
 *updateTaskCompletionStatus*(taskId, completionStatus);  
 break;  
  
 case 4:  
 // Delete task  
 System.*out*.print("Enter the task ID: ");  
 taskId = sc.nextInt();  
 *deleteTask*(taskId);  
 break;  
  
 case 5:  
 // Clear all tasks  
 *clearAllTasks*();  
 break;  
  
 case 6:  
 // Search tasks  
 System.*out*.print("Enter keyword to search tasks: ");  
 String keyword = sc.next();  
 *searchTasks*(keyword);  
 break;  
  
 case 7:  
 System.*out*.println("Exiting the program. Goodbye!");  
 break;  
  
 default:  
 System.*out*.println("Invalid choice. Please enter a number between 1 and 7.");  
 }  
  
 // Add a line break for better readability  
 System.*out*.println();  
 } while (choice != 7);  
  
 // Close the database connection  
 *closeConnection*();  
 }  
}

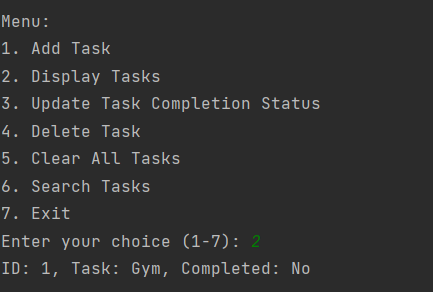
Outputs:-

For Adding:

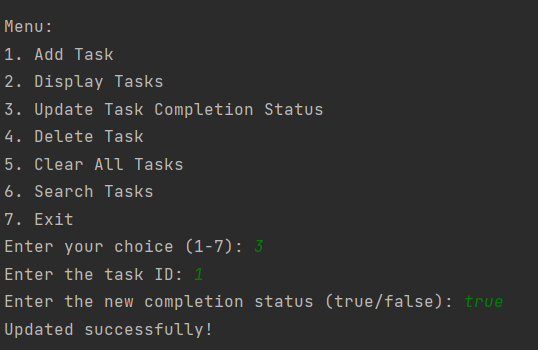


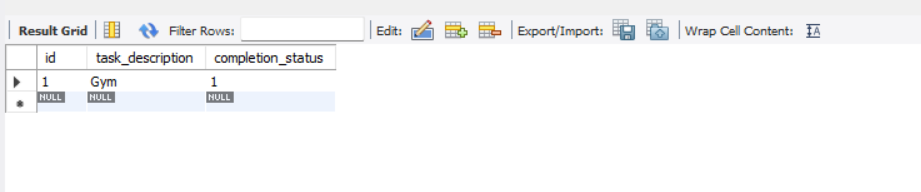


For Displaying:

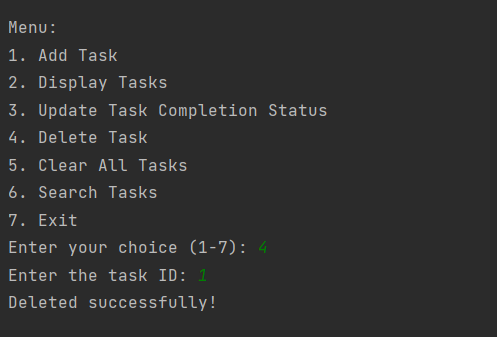


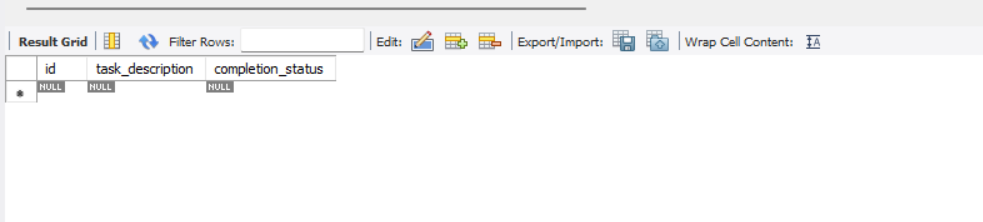
For Updating:





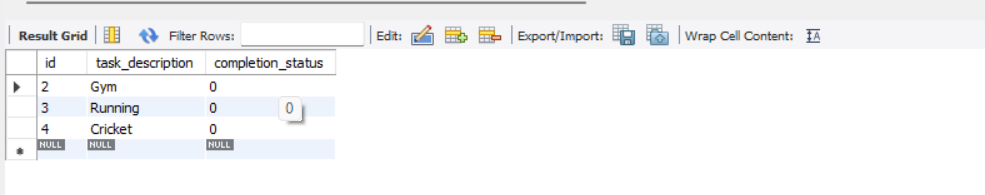
For Delete task:



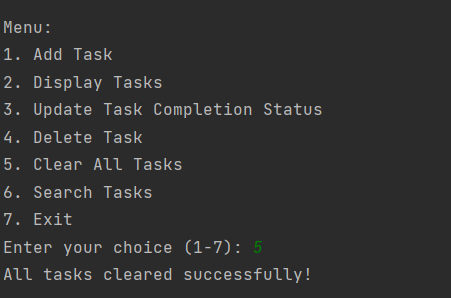


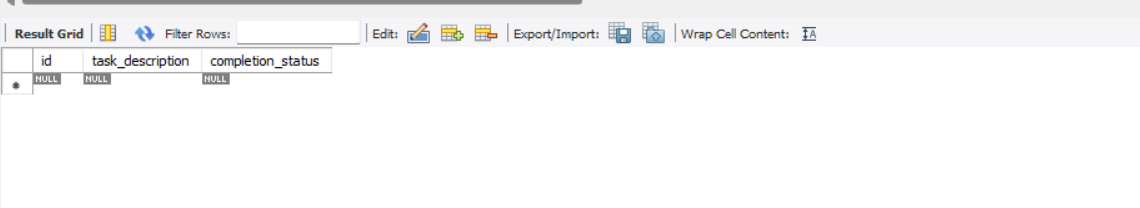
For Clearing task:

1. The tasks are there in the database as follows:



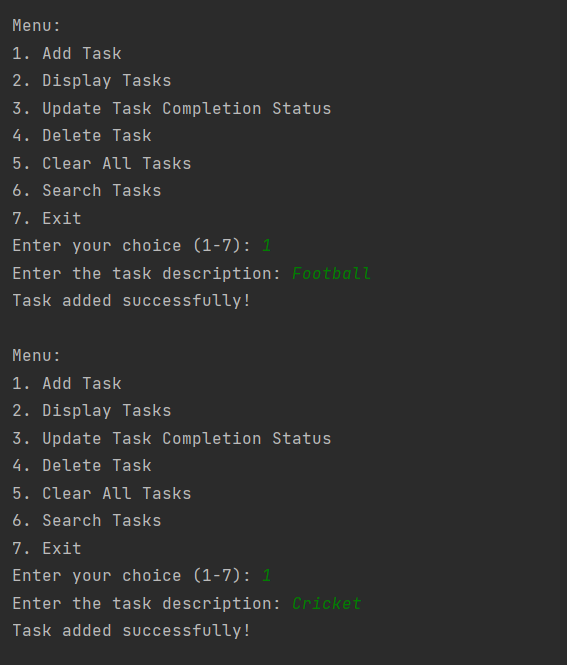
1. After Clearing:

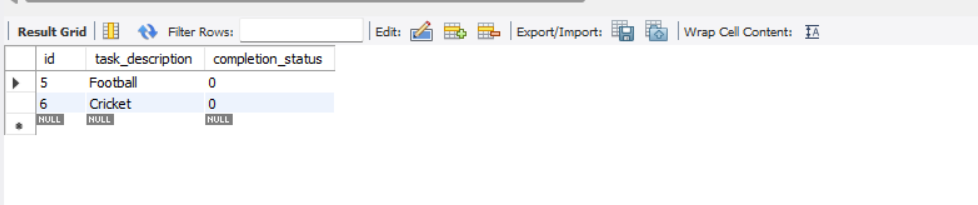




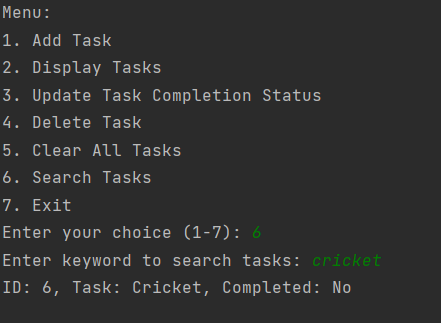
For Searching the task:

1. The tasks are as follows:





1. For searching:



To Exit:

