# Question 3

## StudentInterface.java

```
package question_3;

import question_2.StudentDetails;
import java.rmi.RemoteException;

/**
 *
 * @author Donovan van Heerden | EL2014-0043
 */
public interface StudentInterface extends java.rmi.Remote {
  /**
   * Abstract function which returns an instace of the StudentDetails class
   *
   * @param id
   * @return
   * @throws RemoteException
   */
  public abstract StudentDetails readFromDatabase(int id) throws RemoteException;
}
```

## RMIImpInterface.java

```
package question_3;

import java.sql.*;
import question_2.StudentDetails;

/**
 * RMIImpInterfaces implements the StudentInterface which forces the
 * implementation of only one function "readFromDatabase"
 *
 * @author Donovan van Heerden | EL2014-0043
 */
public class RMIImpInterface implements StudentInterface {

  /**
   * Reads StudentData from the database, using the parameter id as a search to
   * select the data.
   *
   * @param id int
   * @return StudentDetails instance
   */
  @Override
  public StudentDetails readFromDatabase(int id) {
    StudentDetails student = null;
    // Try and initiate a connection to the database
    try (Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/PIHE2019", "root",
"root");
        Statement statement = conn.createStatement()) {

      // Create the SELECT query for selected the data from the database
      ResultSet rs = statement.executeQuery("SELECT * FROM details WHERE Id = " +
id + ";");

      // If we have a record or records, read the data and create an instance of
the
      // StudentDetails class
      while (rs.next()) {
        String firstname = rs.getString("FirstName");
        String lastname = rs.getString("LastName");
        String contactnumber = rs.getString("ContactNumber");
        String address = rs.getString("Address");

        student = new StudentDetails(id, firstname, lastname, contactnumber,
address);
      }
    } catch (SQLException e) {
      System.out.println(e.toString());
    }

    // Return the StudentDetails instance
    return student;
  }

}
```

## RMIClient.java

```
package question_3;

import java.awt.event.ActionEvent;
import java.net.MalformedURLException;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.SwingConstants;
import java.rmi.Naming;
import java.rmi.NotBoundException;
import java.rmi.RemoteException;
import java.util.logging.Level;
import java.util.logging.Logger;
import question_2.StudentDetails;
```

```
/**
 *
 * @author Donovan van Heerden | EL2014-0043
 */
public class RMIClient {

  private JFrame frmMain;
  private static StudentInterface rmi;

  // This labels string array is used when positioning and laying out the
  // components
  private final String[] labels = new String[] { "Student ID:", "First Name:",
"Last Name:", "Contact Number:",
      "Address:" };

  private JTextField txtId;
  private JTextField txtFirstName;
  private JTextField txtLastName;
  private JTextField txtContactNumber;
  private JTextField txtAddress;

  private JButton btnClear;
  private JButton btnSearch;

  /**
   * Entry point of Client, creates a new instance of the RMIClient class.
   *
   * @param args
   */
  public static void main(String[] args) {
    RMIClient client = new RMIClient();
  }

  /**
   * Constructor of RMIClient, creates and initialises the JFrame and components
   */
  RMIClient() {
    createForm();
  }

  /**
   * Initialises the JFrame, JTextFields and JButtons
   */
  private void initialise() {
    frmMain = new JFrame();

    txtId = new JTextField(5);
    txtFirstName = new JTextField(50);
    txtLastName = new JTextField(50);
    txtContactNumber = new JTextField(10);
    txtAddress = new JTextField(250);

    btnSearch = new JButton("Search");
    btnClear = new JButton("Clear");
  }

  /**
   * Creates and sets various values relating to each component on the JFrame.
   */
  private void createForm() {
    initialise();

    frmMain.setTitle("Student Details");
    frmMain.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frmMain.setResizable(false);
    frmMain.getContentPane().setLayout(null);

    frmMain.setBackground(new java.awt.Color(255, 255, 255));
    frmMain.setSize(296, 190);
    frmMain.setLocationRelativeTo(null); // Centers JFrame

    frmMain.setVisible(true);

    txtId.setBounds(140, 0, 140, 25);
    txtFirstName.setBounds(140, (1 * 25), 140, 25);
    txtLastName.setBounds(140, (2 * 25), 140, 25);
    txtContactNumber.setBounds(140, (3 * 25), 140, 25);
    txtAddress.setBounds(140, (4 * 25), 140, 25);

    btnSearch.setBounds(140, (5 * 25), 140, 25);
    // Add the action listener to the button for handling the search function
    btnSearch.addActionListener((ActionEvent event) -> {
      try {
        Search();
      } catch (NotBoundException | MalformedURLException | RemoteException ex) {
        Logger.getLogger(RMIClient.class.getName()).log(Level.SEVERE, null, ex);
      }
    });

    btnClear.setBounds(0, (5 * 25), 140, 25);
    // Add the action listener to the button for handling the clear function
    btnClear.addActionListener((ActionEvent event) -> {
      Clear();
    });

    // Iterate over the labels string array to set the label value and position
the
    // labels dynamically
    for (int index = 0; index < labels.length; index++) {
      JLabel lbl = new JLabel(labels[index], SwingConstants.RIGHT);
      lbl.setBounds(15, (index * 25), 120, 25);
      frmMain.add(lbl);
    }

    // Add all the components to the JFrame
    frmMain.add(txtId);
    frmMain.add(txtFirstName);
    frmMain.add(txtLastName);
    frmMain.add(txtContactNumber);
    frmMain.add(txtAddress);
    frmMain.add(btnSearch);
    frmMain.add(btnClear);
  }

  /**
   * Searches for the StudentDetails data that relate to the Id entered into the
   * JTextField. Uses RMI to fetch the data.
   *
   * @throws NotBoundException
   * @throws MalformedURLException
   * @throws RemoteException
   */
  public void Search() throws NotBoundException, MalformedURLException,
RemoteException {
    int id = Integer.parseInt(txtId.getText());

    // Casts the stub reference of the RMI object to the StudentInterface
    rmi = (StudentInterface) Naming.lookup("//localhost:4300/RMI");
    // Executes the readFromDatabase function, by passing in the Id of the student
    // you are looking for
    StudentDetails student = rmi.readFromDatabase(id);

    // If a student is found, populate the JTextFields with student data from the
    // StudentDetails instance
    if (student != null) {
      txtFirstName.setText(student.getFirstName());
      txtLastName.setText(student.getLastName());
      txtContactNumber.setText(student.getContactNumber());
      txtAddress.setText(student.getAddress());
    }
  }

  /**
   * Clears each JTextField
   */
  public void Clear() {
    txtId.setText("");
    txtFirstName.setText("");
    txtLastName.setText("");
    txtContactNumber.setText("");
    txtAddress.setText("");
  }
}
```

### RMIServer.java

```
package question_3;
```

```java
import java.net.MalformedURLException;
import java.rmi.AlreadyBoundException;
import java.rmi.registry.LocateRegistry;
import java.rmi.Naming;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

/**
 *
 * @author Donovan van Heerden | EL2014-0043
 */
public class RMIServer {
  /**
   * Constructor of RMIServer class
   */
  public RMIServer() {
  }

  /**
   * Entry point of RMIServer, creates a new instance of the RMIImpInterface
   * class, creates a RMI stub and binds the stub to localhost:4300/RMI
   *
   * @param args
   * @throws RemoteException
   * @throws MalformedURLException
   * @throws AlreadyBoundException
   */
  public static void main(String[] args) throws RemoteException,
MalformedURLException, AlreadyBoundException {
    try {
      RMIImpInterface obj = new RMIImpInterface();
      StudentInterface stub = (StudentInterface)
UnicastRemoteObject.exportObject(obj, 0);

      LocateRegistry.createRegistry(4300);
      Naming.rebind("//localhost:4300/RMI", stub);

    } catch (RemoteException e) {
      System.err.println(e.toString());
    }

  }
}
```