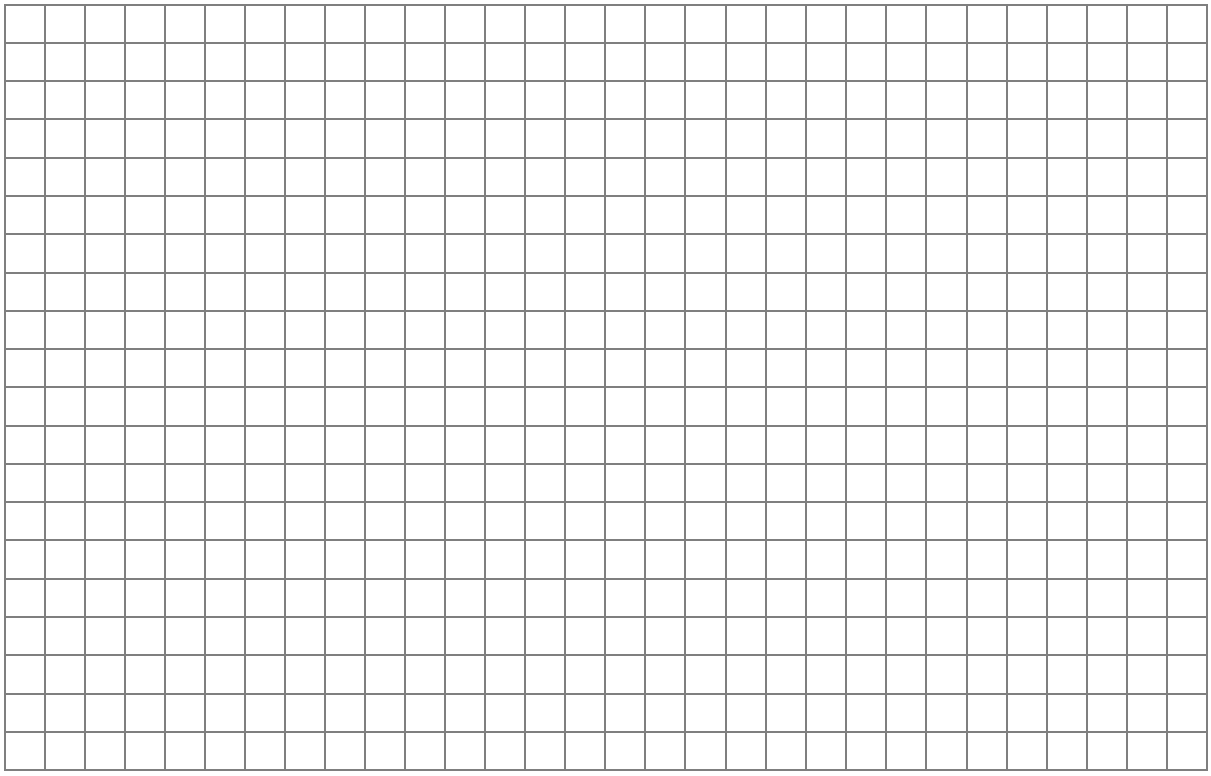


DFP50043 INTEGRATIVE PROGRAMMING AND TECHNOLOGIES

PRACTICAL ACTIVITY

TOPIC 4: JAVA DATABASE CONNECTIVITY

JABATAN TEKNOLOGI MAKLUMAT & KOMUNIKASI
POLITEKNIK MUKAH

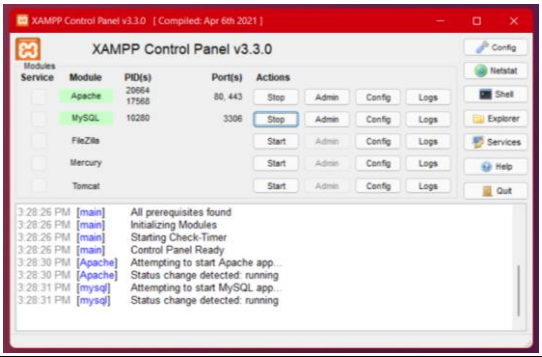
Lab Activity **4.1**: Creating a database

Description: In this *lab activity 4.1*, you will be creating a new database in XAMPP. Follow the step below to complete the activity.

Step 1:

Open XAMPP and start Apache & MySQL module.

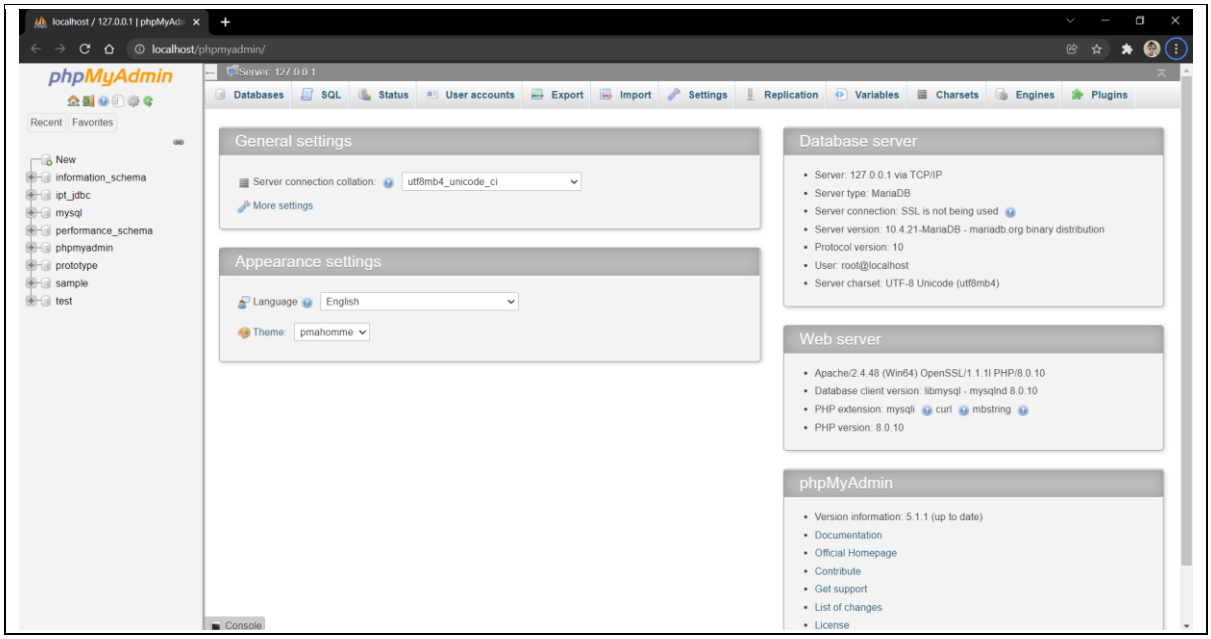
Output:



Step 2:

Click the Admin button on the MySQL module, you will be navigated to phpMyAdmin page as shown below.

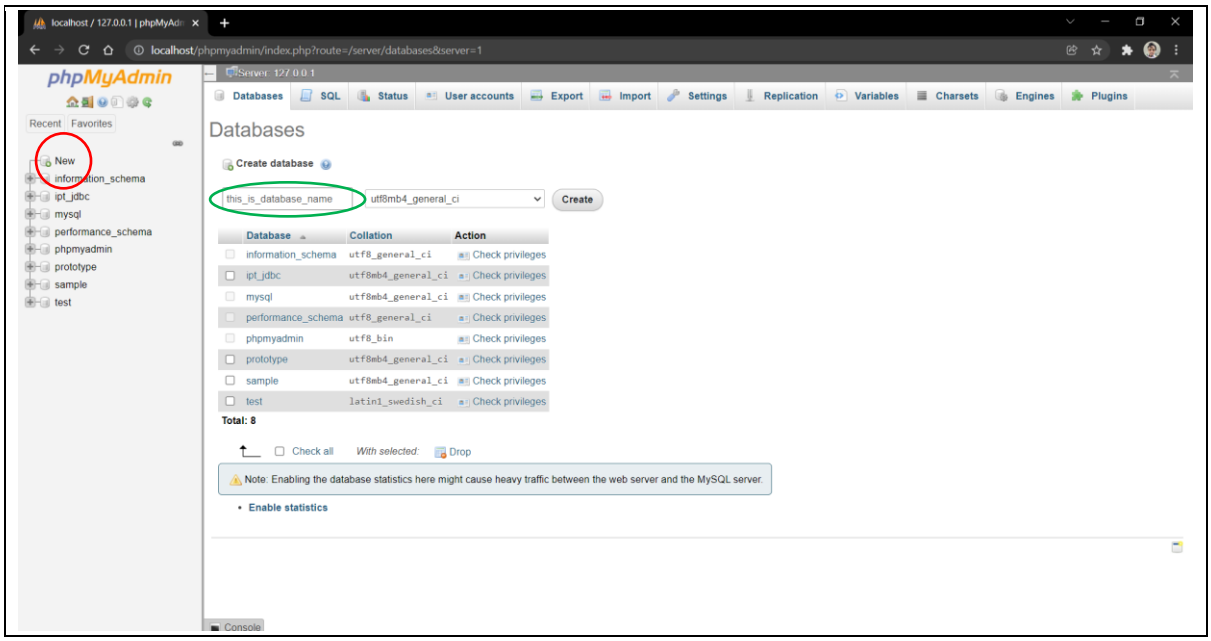
Output:



Step 3:

To create a new database, click New (red circle) and named your database (green circle) appropriately as shown below.

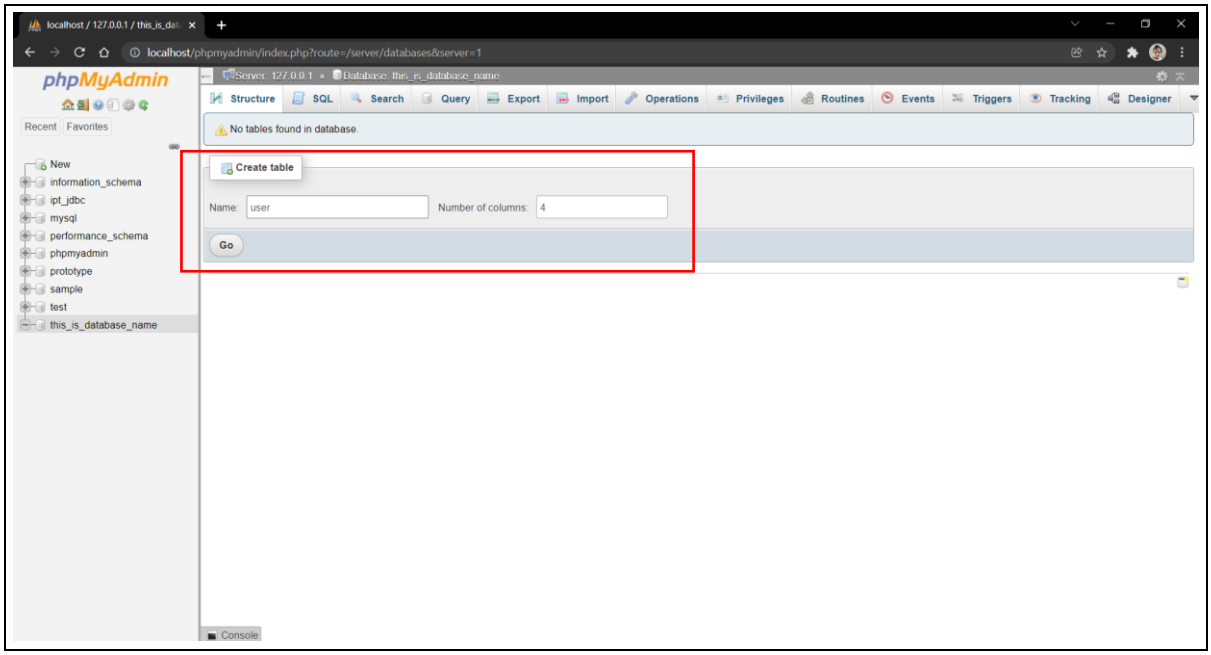
Output:



Step 4:

Once done you will be asked to create your first database table, make sure you named your table appropriately and just click Go button. The number of table column can be modified later.

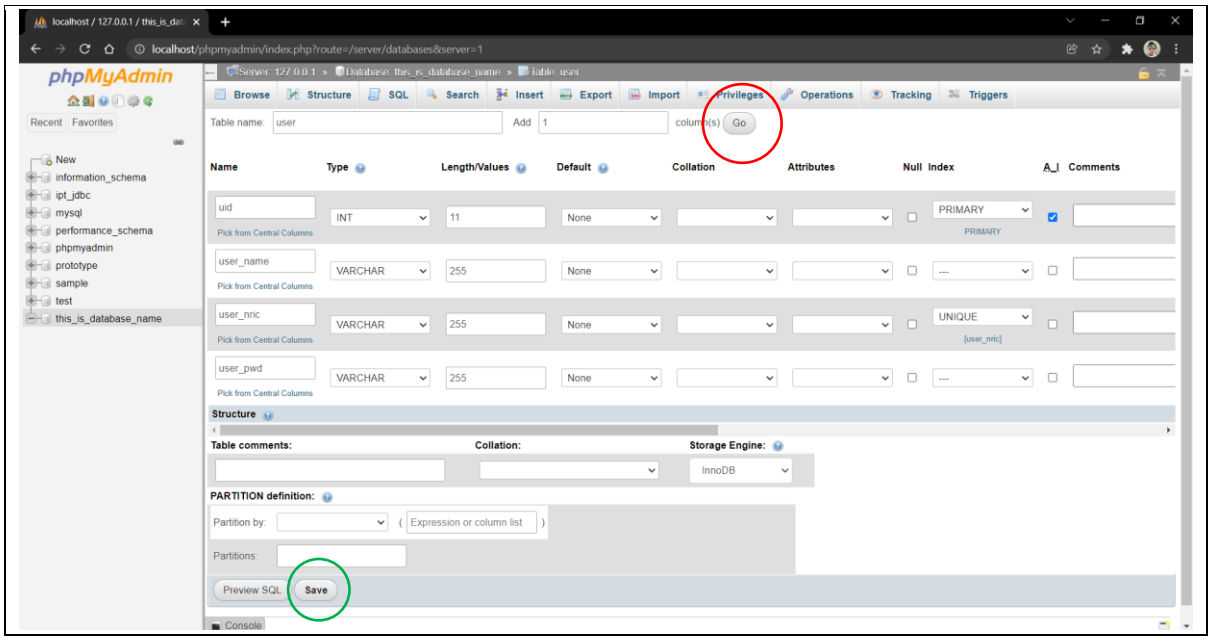
Output:



Step 3:

Once table created, fill in the field, if there is redundant field just let it empty but if there is not enough field just click the Go button (red circle) to add another column. Once finished, just click the Save button (green circle).

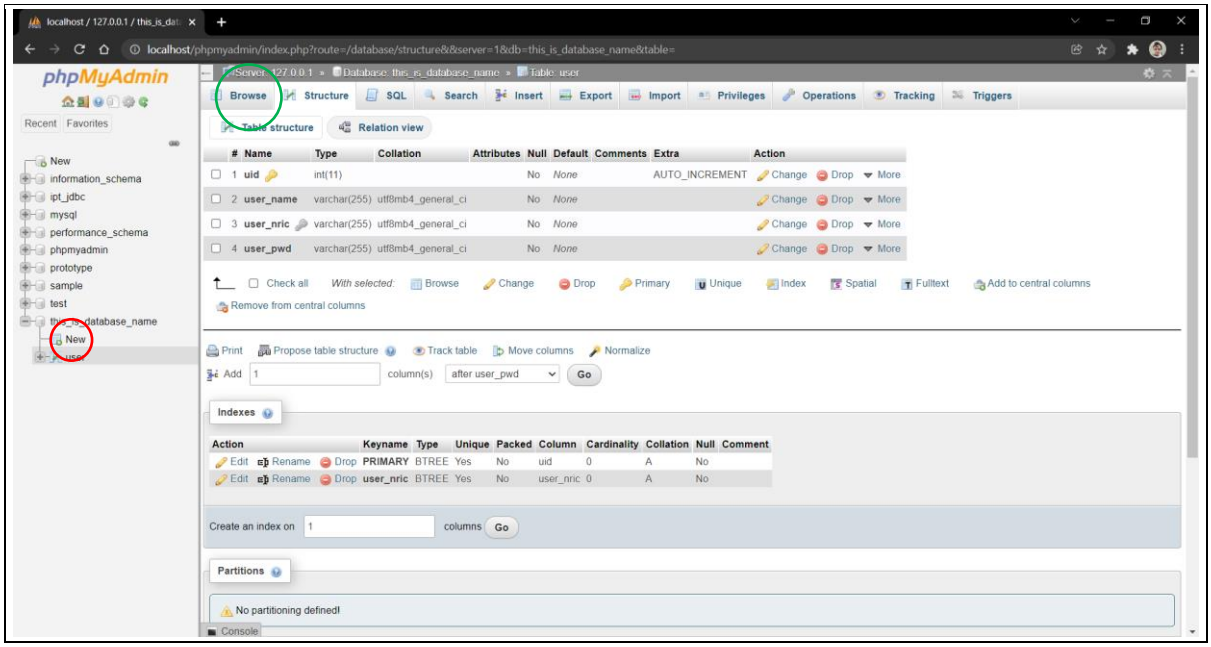
Output:



Step 4:

Once saved, you will be displayed the structure of the table created. To view the records, just click the browse tab (red circle). To create another table, click the New (green circle) button.

Output:



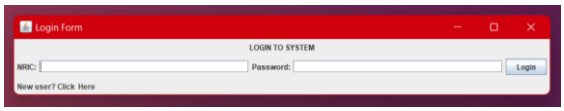
Lab Activity **3.2**: Creating Login Form GUI

Description: In this lab activity, you will be creating a simple login form interface to authenticate the user ID before the user can access the system dashboard. The code below demonstrates how we can create it.

Code:

```
package labactivitytopic4;

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class LoginForm {

    JFrame frame;
    JPanel pnl_1, pnl_2, pnl_3;
    JLabel lbl_title, lbl_nric, lbl_pwd, lbl_register, lbl_here;
    JTextField txtf_nric;
    JPasswordField pwd_1;
    JButton btn_login;

    LoginForm() {
        frame = new JFrame("Login Form");

        pnl_1 = new JPanel(new FlowLayout(FlowLayout.CENTER));
        lbl_title = new JLabel("LOGIN TO SYSTEM");
        pnl_1.add(lbl_title);
        frame.add(pnl_1, BorderLayout.NORTH);

        pnl_2 = new JPanel(new FlowLayout(FlowLayout.CENTER));
        lbl_nric = new JLabel("NRIC:");
        lbl_pwd = new JLabel("Password:");
        txtf_nric = new JTextField(30);
        pwd_1 = new JPasswordField(30);
        btn_login = new JButton("Login");
        pnl_2.add(lbl_nric);
        pnl_2.add(txtf_nric);
        pnl_2.add(lbl_pwd);
        pnl_2.add(pwd_1);
        pnl_2.add(btn_login);
        frame.add(pnl_2, BorderLayout.CENTER);

        pnl_3 = new JPanel(new FlowLayout(FlowLayout.LEADING));
        lbl_register = new JLabel("New user? Click");
        lbl_register.setForeground(Color.DARK_GRAY);
        lbl_here = new JLabel("Here");
        lbl_here.setForeground(Color.DARK_GRAY);
        pnl_3.add(lbl_register);
        pnl_3.add(lbl_here);
        frame.add(pnl_3, BorderLayout.SOUTH);

        frame.pack();
        frame.setVisible(true);
        frame.setDefaultCloseOperation(frame.DISPOSE_ON_CLOSE);
    }

    public static void main(String[] args) {
        LoginForm apps = new LoginForm();
    }
}
```

Output:



Lab Activity **3.3**: Implementing event – handling & JDBC (SELECT SQL Operation)

Description: In this lab activity, you will be implementing event – handling to check either the form is empty or not and to handle the new user registration, so when the Here is clicked, a registration form will be displayed. Then if and only if the form is not empty, we will trigger the JDBC action to cross – check the user input with the database records. In this lab activity we will separate the action by using Function so our code will be more readable.

Code:

```
//implement the following Listener Inteface
public class LoginForm implements MouseListener, ActionListener {
//fill in the auto - generated code with the following code
@Override
public void mouseClicked(MouseEvent e) {
    if (e.getSource() == lbl_here) {
        RegistrationForm obj = new RegistrationForm();
    }
}

@Override
public void mousePressed(MouseEvent e) {
}

@Override
public void mouseReleased(MouseEvent e) {
}

@Override
public void mouseEntered(MouseEvent e) {
    lbl_here.setForeground(Color.BLUE);
}

@Override
public void mouseExited(MouseEvent e) {
    lbl_here.setForeground(Color.DARK_GRAY);
}
@Override
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == btn_login) {
        if (is_field_empty() == false) {
            //JOptionPane.showMessageDialog(null, "Registered");
            is_authenticated();
        } else {
            JOptionPane.showMessageDialog(null, "Make Sure you fill up the form!", "Error", JOptionPane.ERROR_MESSAGE);
        }
    }
}
//create a function to check either the field is empty or not
//make sure to put this function inside the class and outside the constructor
//function to check field empty or not
private boolean is_field_empty() {
    boolean empty = false;
    String nric = txtf_nric.getText();
    char[] pwd = pwd_1.getPassword();

    if (nric == "" || pwd.length == 0) {
        empty = true;
    } else {
        empty = false;
    }
    return empty;
}
```

```
//create a function to trigger JDBC action
//make sure to put this function inside the class and outside the constructor
//function to check data with database
private void is_authenticated() {
    String nric = txtf_nric.getText();
    char[] pwd = pwd_1.getPassword();
    String pwd_from_db = null;

    String db_username = "root";
    String db_pwd = "";
    String db_url = "jdbc:mysql://localhost:3306/ipt_jdbc";
    Connection conn = null;
    ResultSet rs = null;

    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        conn = DriverManager.getConnection(db_url, db_username, db_pwd);
        String sql = "SELECT * FROM user WHERE user_nric = ?";

        PreparedStatement statement = conn.prepareStatement(sql, ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CONCUR_UPDATABLE);
        statement.setString(1, nric);
        rs = statement.executeQuery();
        if (rs.next()) {
            rs.beforeFirst();
            while (rs.next()) {
                pwd_from_db = rs.getString("user_pwd");
                char[] temp_pwd = pwd_from_db.toCharArray();
                if (Arrays.equals(pwd, temp_pwd)) {
                    JOptionPane.showMessageDialog(null, "User authenticated!");
                    Dashboard obj = new Dashboard();
                    frame.dispose();
                } else {
                    JOptionPane.showMessageDialog(null, "Wrong password!", "Error", JOptionPane.ERROR_MESSAGE);
                }
            }
        } else {
            JOptionPane.showMessageDialog(null, "User " + nric + " not found. Please register first.", "Error", JOptionPane.ERROR_MESSAGE);
        }
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Error: " + ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}
//register the Here JLabel and Login JButton to the ListenerInterface
    pnl_1 = new JPanel(new FlowLayout(FlowLayout.CENTER));
    lbl_title = new JLabel("LOGIN TO SYSTEM");
    pnl_1.add(lbl_title);
    frame.add(pnl_1, BorderLayout.NORTH);

    pnl_2 = new JPanel(new FlowLayout(FlowLayout.CENTER));
    lbl_nric = new JLabel("NRIC:");
    lbl_pwd = new JLabel("Password:");
    txtf_nric = new JTextField(30);
    pwd_1 = new JPasswordField(30);
    btn_login = new JButton("Login");
    btn_login.addActionListener(this);
    pnl_2.add(lbl_nric);
    pnl_2.add(txtf_nric);
    pnl_2.add(lbl_pwd);
    pnl_2.add(pwd_1);
    pnl_2.add(btn_login);
    frame.add(pnl_2, BorderLayout.CENTER);

    pnl_3 = new JPanel(new FlowLayout(FlowLayout.LEADING));
    lbl_register = new JLabel("New user? Click");
    lbl_register.setForeground(Color.DARK_GRAY);
    lbl_here = new JLabel("Here");
    lbl_here.addMouseListener(this);
    lbl_here.setForeground(Color.DARK_GRAY);
    pnl_3.add(lbl_register);
    pnl_3.add(lbl_here);
    frame.add(pnl_3, BorderLayout.SOUTH);
```

Lab Activity **3.4**: Creating Registration Form GUI

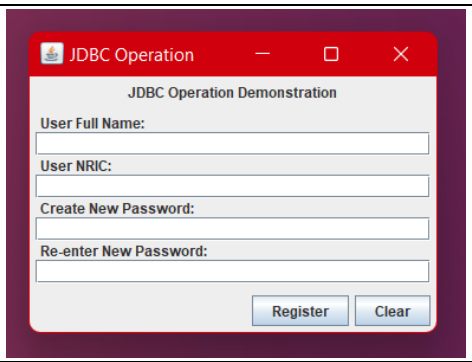
Description: In this lab activity, you will be creating a user registration form interface. This form will be used to insert new record to the database.

Code:

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class RegistrationForm {

    JFrame frame;
    JPanel pnl_1, pnl_2, pnl_3, pnl_4;
    JLabel lbl_title, lbl_name, lbl_nric, lbl_pwd, lbl_pwd_check;
    JTextField txtf_name, txtf_nric;
    JPasswordField pwd_1, pwd_2;
    JButton btn_register, btn_clear;

    RegistrationForm() {
        frame = new JFrame("JDBC Operation");

        //NORTH SECTION
        pnl_1 = new JPanel(new FlowLayout(FlowLayout.CENTER));
        lbl_title = new JLabel("JDBC Operation Demonstration");
        pnl_1.add(lbl_title);

        //CENTER SECTION
        pnl_2 = new JPanel();
        pnl_2.setLayout(new BoxLayout(pnl_2, BoxLayout.Y_AXIS));

        pnl_3 = new JPanel(new FlowLayout(FlowLayout.LEADING));

        lbl_name = new JLabel("User Full Name:");
        lbl_nric = new JLabel("User NRIC:");
        lbl_pwd = new JLabel("Create New Password:");
        lbl_pwd_check = new JLabel("Re-enter New Password:");

        txtf_name = new JTextField(30);
        txtf_nric = new JTextField(30);

        pwd_1 = new JPasswordField(30);
        pwd_2 = new JPasswordField(30);

        pnl_2.add(lbl_name);
        pnl_2.add(txtf_name);
        pnl_2.add(lbl_nric);
        pnl_2.add(txtf_nric);
        pnl_2.add(lbl_pwd);
        pnl_2.add(pwd_1);
        pnl_2.add(lbl_pwd_check);
        pnl_2.add(pwd_2);
        pnl_3.add(pnl_2);

        //SOUTH SECTION
        pnl_4 = new JPanel(new FlowLayout(FlowLayout.TRAILING));
        btn_register = new JButton("Register");
        btn_clear = new JButton("Clear");
        pnl_4.add(btn_register);
        pnl_4.add(btn_clear);

        frame.add(pnl_1, BorderLayout.NORTH);
        frame.add(pnl_3, BorderLayout.CENTER);
        frame.add(pnl_4, BorderLayout.SOUTH);

        frame.pack();
```

```
        frame.setVisible(true);
        frame.setDefaultCloseOperation(frame.DISPOSE_ON_CLOSE);
    }
}
```

Output:



Lab Activity **3.5**: Implementing event – handling & JDBC (INSERT SQL Operation)

Description: In this lab activity, you will be implementing event – handling to check either the form is empty or not, to check either the password entered is similar or not and to trigger the JDBC action to insert the data to the database records. In this lab activity we will separate the action by using Function so our code will be more readable.

Code:

```
//make sure to import util and sql package
//implement the action listener interface to handled the Register button

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.sql.*;
import java.util.*;

public class RegistrationForm implements ActionListener {
//fill in the auto - generated code with the following code
@Override
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == btn_register) {
        if (is_field_empty() == false) {
            if (is_password_same() == false) {
                JOptionPane.showMessageDialog(null, "Password did not match!", "Error", JOptionPane.ERROR_MESSAGE);
            } else {
                register_action();
                clear_form();
                frame.dispose();
            }
        } else {
            JOptionPane.showMessageDialog(null, "Make Sure you fill up the form!", "Error", JOptionPane.ERROR_MESSAGE);
        }
    } else if (e.getSource() == btn_clear) {
        clear_form();
    }
}
```

```
//create a function to check either the field is empty or not
//make sure to put this function inside the class and outside the constructor
```

```
//function to check field empty or not
private boolean is_field_empty() {
    boolean empty = false;
    String name = txtf_name.getText();
    String nric = txtf_nric.getText();
    char[] new_pwd = pwd_1.getPassword();
    char[] re_pwd = pwd_2.getPassword();

    if (name == "" || nric == "" || new_pwd.length == 0 || re_pwd.length == 0) {
        empty = true;
    } else {
        empty = false;
    }
    return empty;
}
```

```
//create a function to check either the password entered is similar or not
//make sure to put this function inside the class and outside the constructor
```

```
//function to check password same or not
private boolean is_password_same() {
    boolean same = false;
    char[] new_pwd = pwd_1.getPassword();
    char[] re_pwd = pwd_2.getPassword();

    if (Arrays.equals(new_pwd, re_pwd)) {
        same = true;
    } else {
        same = false;
    }
    return same;
}
```

```
//create a function to trigger JDBC action
//make sure to put this function inside the class and outside the constructor
```

```
private void register_action() {
    String name = txtf_name.getText();
    String nric = txtf_nric.getText();
    char[] new_pwd = pwd_1.getPassword();
    String pwd = String.valueOf(new_pwd);

    String db_username = "root";
    String db_pwd = "";
    String db_url = "jdbc:mysql://localhost:3306/ipt_jdbc";
    Connection conn = null;
    int status = 0;
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        conn = DriverManager.getConnection(db_url, db_username, db_pwd);
        String sql = "INSERT INTO user (user_name, user_nric, user_pwd) VALUES (?, ?, ?)";

        PreparedStatement statement = conn.prepareStatement(sql);
        statement.setString(1, name);
        statement.setString(2, nric);
        statement.setString(3, pwd);

        status = statement.executeUpdate();
        if (status > 0) {
            JOptionPane.showMessageDialog(null, "Registration completed. Please login using again...");
        } else {
            JOptionPane.showMessageDialog(null, "Unable to complete the registration process", "Error", JOptionPane.ERROR_MESSAGE);
        }
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Error: " + ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

```
//create a function to clear the form after any action (if needed)
//make sure to put this function inside the class and outside the constructor
 //function to clear the form
 private void clear_form() {
     txtf_name.setText("");
     txtf_nric.setText("");
     pwd_1.setText("");
     pwd_2.setText("");
 }
//register the JButton to the ListenerInterface
    //SOUTH SECTION
    pnl_4 = new JPanel(new FlowLayout(FlowLayout.TRAILING));
    btn_register = new JButton("Register");
    btn_register.addActionListener(this);
    btn_clear = new JButton("Clear");
    btn_clear.addActionListener(this);
    pnl_4.add(btn_register);
    pnl_4.add(btn_clear);

    frame.add(pnl_1, BorderLayout.NORTH);
    frame.add(pnl_3, BorderLayout.CENTER);
    frame.add(pnl_4, BorderLayout.SOUTH);

    frame.pack();
    frame.setVisible(true);
    frame.setDefaultCloseOperation(frame.DISPOSE_ON_CLOSE);
}
```

Lab Activity **3.6**: Creating Dashboard GUI

Description: In this lab activity, you will be creating dashboard of your system. This GUI will be divided into 3 sections. The first section is used to fetch the data from the database, the second section is used to display the fetched data and the third section is used to display the data list (summary) from the database.

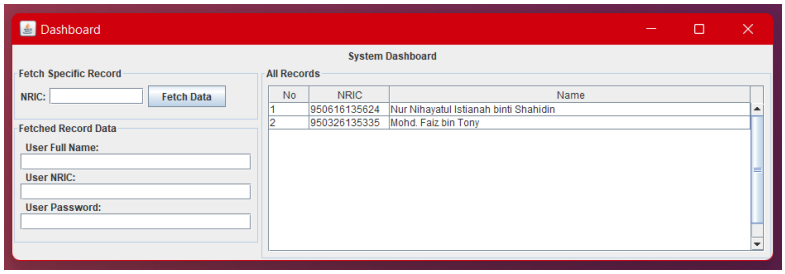
Code:

```
package labactivitytopic4;

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.sql.*;
import java.util.*;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.border.*;
import javax.swing.table.DefaultTableModel;

public class Dashboard implements ActionListener {

    JFrame frame;
    JPanel pnl_1, pnl_2, pnl_3, pnl_4, pnl_5, pnl_6, pnl_7, pnl_8, pnl_9,
pnl_east;
    JLabel lbl_title, lbl_name, lbl_nric, lbl_pwd, lbl_fetch_data,
lbl_edit_delete, lbl_fetched;
    JTextField txtf_name, txtf_nric, txtf_fetch;
    JPasswordField pwd_1;
    JButton btn_fetch, btn_update, btn_delete;
    TitledBorder title, title2, title3;
    JTable table;
    JScrollPane jsp;
```

```
    ResultSet rs = null;
    String nric_holder = null;

    Dashboard() {
        frame = new JFrame("Dashboard");

        title = BorderFactory.createTitledBorder("Fetched Record Data");
        title2 = BorderFactory.createTitledBorder("Fetch Specific Record");
        title3 = BorderFactory.createTitledBorder("All Records");

        pnl_1 = new JPanel(new FlowLayout(FlowLayout.CENTER));
        lbl_title = new JLabel("System Dashboard");
        pnl_1.add(lbl_title);
        frame.add(pnl_1, BorderLayout.NORTH);

        //base for west section
        pnl_2 = new JPanel();
        //hold component
        pnl_3 = new JPanel();
        pnl_3.setLayout(new BoxLayout(pnl_3, BoxLayout.Y_AXIS));

        pnl_5 = new JPanel(new FlowLayout(FlowLayout.LEADING));
        lbl_fetch_data = new JLabel("NRIC:");
        txtf_fetch = new JTextField(10);
        btn_fetch = new JButton("Fetch Data");
        pnl_5.add(lbl_fetch_data);
        pnl_5.add(txtf_fetch);
        pnl_5.add(btn_fetch);
        pnl_5.setBorder(title2);

        pnl_7 = new JPanel(new FlowLayout(FlowLayout.LEADING));
        pnl_8 = new JPanel();
        pnl_8.setLayout(new BoxLayout(pnl_8, BoxLayout.Y_AXIS));
        lbl_name = new JLabel("User Full Name:");
        lbl_nric = new JLabel("User NRIC:");
        lbl_pwd = new JLabel("User Password:");
        txtf_name = new JTextField(25);
        txtf_nric = new JTextField(25);
        pwd_1 = new JPasswordField(25);
        pnl_8.add(lbl_name);
        pnl_8.add(txtf_name);
        pnl_8.add(lbl_nric);
        pnl_8.add(txtf_nric);
        pnl_8.add(lbl_pwd);
        pnl_8.add(pwd_1);
        pnl_7.add(pnl_8);
        pnl_7.setBorder(title);

        pnl_9 = new JPanel(new FlowLayout(FlowLayout.TRAILING));
        btn_update = new JButton("Update");
        btn_delete = new JButton("Delete");
        pnl_9.add(btn_update);
        pnl_9.add(btn_delete);

        //pnl_3.add(pnl_4);
        pnl_3.add(pnl_5);
        pnl_3.add(pnl_7);
        pnl_3.add(pnl_9);

        frame.add(pnl_3, BorderLayout.WEST);
        hide_operation_button();

        frame.pack();
        frame.setVisible(true);
        frame.setDefaultCloseOperation(frame.DISPOSE_ON_CLOSE);
    }
}
```

Output:

Noted that your out may not have any table yet but in the following step we will add the table to the GUI step by step.



Lab Activity **3.7**: Implementing event – handling & JDBC (UPDATE, DELETE & SELECT SQL Operation)

Description: In this lab activity, you will be implementing event – handling to check either the form is empty or not, to hide/display action button, to clear the data from the form, to load data from database, to draw table on the GUI and to trigger the database operation such as UPDATE, DELETE & SELECT. Follow the instruction step by step to understand the procedure.

Code:

```
//create a common function first
private boolean is_field_empty() {
    boolean empty = false;
    String name = txtf_name.getText();
    String nric = txtf_nric.getText();
    char[] new_pwd = pwd_1.getPassword();

    if (name == "" || nric == "" || new_pwd.length == 0) {
        empty = true;
    } else {
        empty = false;
    }
    return empty;
}

//function to clear the form
private void clear_form() {
    txtf_name.setText("");
    txtf_nric.setText("");
    pwd_1.setText("");
}

private void show_operation_button(){
    btn_delete.setVisible(true);
    btn_update.setVisible(true);
}

private void hide_operation_button(){
    btn_delete.setVisible(false);
    btn_update.setVisible(false);
}
```

```
//create a function to set the data of the form
//this function will be trigger by the fetch button
//this function will select all data based on the nric no entered
private void set_data() {
    String nric = txtf_fetch.getText();
    nric_holder = nric;
    String db_username = "root";
    String db_pwd = "";
    String db_url = "jdbc:mysql://localhost:3306/ipt_jdbc";
    Connection conn = null;
    ResultSet rs = null;

    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        conn = DriverManager.getConnection(db_url, db_username, db_pwd);
        String sql = "SELECT * FROM user WHERE user_nric = ?";

        PreparedStatement statement = conn.prepareStatement(sql, ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CONCUR_UPDATABLE);
        statement.setString(1, nric);
        rs = statement.executeQuery();
        if (rs.next()) {
            rs.beforeFirst();
            while (rs.next()) {
                txtf_name.setText(rs.getString("user_name"));
                txtf_nric.setText(rs.getString("user_nric"));
                pwd_1.setText(rs.getString("user_pwd"));
            }
        } else {
            JOptionPane.showMessageDialog(null, "User " + nric + " not found. Please register first.", "Error", JOptionPane.ERROR_MESSAGE);
        }
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Error: " + ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

```
//create a function to update the records of the database
//this function will be trigger by the update button
//notice that the user_nric value in setString index 4, it used the nric_holder
//we the nric_holder hold the value of the original nric value fetched in the
set_data() function just in case we wanted to update the user_nric
//this is to prevent the SQL statement from referencing the newer value which does
not exist in the database
private void update_action() {
    String name = txtf_name.getText();
    String nric = txtf_nric.getText();
    char[] new_pwd = pwd_1.getPassword();
    String pwd = String.valueOf(new_pwd);

    String db_username = "root";
    String db_pwd = "";
    String db_url = "jdbc:mysql://localhost:3306/ipt_jdbc";
    Connection conn = null;
    int status = 0;
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        conn = DriverManager.getConnection(db_url, db_username, db_pwd);
        String sql = "UPDATE user SET user_name = ?, user_nric = ?, user_pwd = ? WHERE user_nric = ?";

        PreparedStatement statement = conn.prepareStatement(sql);
        statement.setString(1, name);
        statement.setString(2, nric);
        statement.setString(3, pwd);
        statement.setString(4, nric_holder);

        status = statement.executeUpdate();
        if (status > 0) {
            JOptionPane.showMessageDialog(null, "Update process completed");
            //fetch_data();
            loadData();
        } else {
            JOptionPane.showMessageDialog(null, "Unable to complete the update process", "Error", JOptionPane.ERROR_MESSAGE);
        }
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Error: " + ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

```
//create a function to delete the records of the database
//this function will be trigger by the delete button
private void delete_action() {
    String db_username = "root";
    String db_pwd = "";
    String db_url = "jdbc:mysql://localhost:3306/ipt_jdbc";
    Connection conn = null;
    int status = 0;
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        conn = DriverManager.getConnection(db_url, db_username, db_pwd);
        String sql = "DELETE FROM user WHERE user_nric = ?";

        PreparedStatement statement = conn.prepareStatement(sql);

        statement.setString(1, nric_holder);

        status = statement.executeUpdate();
        if (status > 0) {
            JOptionPane.showMessageDialog(null, "Delete process completed");
            //fetch_data();
            loadData();
        } else {
            JOptionPane.showMessageDialog(null, "Unable to complete the delete process", "Error", JOptionPane.ERROR_MESSAGE);
        }
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Error: " + ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}
//create a function to fetch all data from the database
//this function will be used to create our table
private void fetch_data() {
    String db_username = "root";
    String db_pwd = "";
    String db_url = "jdbc:mysql://localhost:3306/ipt_jdbc";
    Connection conn = null;

    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        conn = DriverManager.getConnection(db_url, db_username, db_pwd);
        String sql = "SELECT * FROM user";

        PreparedStatement statement = conn.prepareStatement(sql);

        rs = statement.executeQuery();
    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Error: " + ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
}
//create a function to create a table model
//this function will utilize the result set data fetched in the fetch_data()
function
public static DefaultTableModel buildTableModel(ResultSet rs)
        throws SQLException {
    ResultSetMetaData metaData = rs.getMetaData();

    Vector<String> columnNames = new Vector<String>();
    columnNames.add("No");
    columnNames.add("NRIC");
    columnNames.add("Name");

    Vector<Vector<Object>> data = new Vector<Vector<Object>>();
    int count = 0;
    while (rs.next()) {
        count++;
        Vector<Object> vector = new Vector<Object>();
        vector.add(count);
        vector.add(rs.getObject(3));
        vector.add(rs.getObject(2));

        data.add(vector);
    }
    return new DefaultTableModel(data, columnNames);
}
```

```
//create a function to create the table
//this function will call the fetch_data() first to set up the result set
//then it will call the modelling function and passed the result set data
//the build table model will return a table and displayed to the GUI
//we need to call loadData() in GUI to display the table
```

```
private void loadData() throws SQLException {
    fetch_data();
    table.setModel(buildTableModel(rs));
    table.setPreferredSize(new Dimension(600, 200));
    table.getColumnModel().getColumn(0).setPreferredWidth(50);
    table.getColumnModel().getColumn(1).setPreferredWidth(100);
    table.getColumnModel().getColumn(2).setPreferredWidth(450);
}
```

```
//complete the code by implement action listener and import any required package
```

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.sql.*;
import java.util.*;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.border.*;
import javax.swing.table.DefaultTableModel;

public class Dashboard implements ActionListener {
```

```
//fill in the auto - generated code with the following code
```

```
@Override
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == btn_update) {
        if (is_field_empty() == false) {
            update_action();
            clear_form();
            hide_operation_button();
        } else {
            JOptionPane.showMessageDialog(null, "Make Sure you fill up the form!", "Error", JOptionPane.ERROR_MESSAGE);
        }
    } else if (e.getSource() == btn_fetch) {
        set_data();
        show_operation_button();
    } else if (e.getSource() == btn_delete) {
        delete_action();
        hide_operation_button();
    }
}
```

```
//lastly in the GUI section add in the following code to display the table
```

```
    //EAST SECTION
    pnl_east = new JPanel();
    pnl_east.setBorder(title3);
    try {
        fetch_data();
        table = new JTable();
        loadData();
        jsp = new JScrollPane(table);
        jsp.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);
        jsp.setPreferredSize(new Dimension(600, 200));

    } catch (Exception ex) {
        JOptionPane.showMessageDialog(null, "Error: " + ex, "Error", JOptionPane.ERROR_MESSAGE);
    }
    pnl_east.add(jsp);
    frame.add(pnl_east, BorderLayout.EAST);
```

Your Task:

1. Add data validation on the NRIC part, limit the character to 12
2. Create a button in the Dashboard to INSERT new record, you may refer to the login form on how to call the Registration Form
3. Insert medias into the program
4. Change the layout of the program since this program will be the master code for your final assessment