**SURVEY SYSTEM**

**A Mini Project Report**

**Submitted by**

**NAME : GUNASEELAN N**

**ROLL NO : 21CSR059**

***in partial fulfillment of the requirements***

***for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

****

**KONGU ENGINEERING COLLEGE**

**(Autonomous)**

**PERUNDURAI ERODE-638060**

**JANUARY 2023**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**KONGU ENGINEERING COLLEGE**

**(Autonomous)**

**PERUNDURAI ERODE-638060**

**JANUARY 2023**

**BONAFIDE CERTIFICATE**

This is to certify that the Project Report entitled **SURVEY SYSTEM** is the bonafide record of project work done by **GUNASEELAN N(21CSR059)** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in **Computer Science and Engineering** of Anna University, Chennai during the year 2022-2023.

**COURSE INSTRUCTOR HEAD OF THE DEPARTMENT**

**Date:**

Submitted for the viva voce examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ABSTRACT**

In Today’s generation, We are engaged in highly computerized technology aiming to make individual lifestyle more comfortable and easier most especially in the world of business This proposed online project is an implementation of Java programming language for software generation that is important in college or organization to carry survey. In this system of survey, only the users authenticated by admin from the database system can drop their vote or express their viewpoint regarding the issue. Being online software, it can be logged on from anywhere with internet access.The proposed Online Survey System is easy and comfortable to use. In this software, the answers or viewpoints of the participants are collected using a radio buttonThe system plays a vital role in minimizing the budget of the survey. The implementation of the project avoids the programs such as meetings, conferences etc.to take any decision or research. With the help of this online system, one can easily forward ideas and viewpoints to the officials.

**CHAPTER 1**

**JAVAFX**

1. **INTRODCUTION**

* **OVERVIEW :**

The JavaFX Platform is a rich client platform for cross-screen rich internet application and content. It consists of common elements and device specific elements for desktop, mobile and TV. The JavaFX Runtimes targeted for different devices will ensure consistency and fidelity for content created based on the JavaFx Common APIs.

* **EVOLUTION OF JAVAFX :**

JavaFx is the evolution of the Java rich client platform, designed to address the needs of today’s and tomorrow’s customers. JavaFX was originally developed by Chris Oliver at SeeBeyond and it was called F3 (Form Follows Function). F3 was a Java scripting language for easily developing GUI applications. It offered declarative syntax, static typing, type inference, data binding, animation, 2D graphics, and Swing components. SeeBeyond was bought by Sun Microsystems and F3 was renamed JavaFX in 2007. Oracle acquired Sun Microsystems in 2010. Oracle then open JavaFX was originally developed by Chris Oliver at SeeBeyond and it was called F3 (Form Follows Function). F3 was a Java scripting language for easily developing GUI applications. It offered declarative syntax, static typing, type inference, data binding, animation, 2D graphics, and Swing components. SeeBeyond was bought by Sun Microsystems and F3 was renamed JavaFX in 2007. Oracle acquired Sun Microsystems in 2010. Oracle then open sourced JavaFX in 2013.

The first version of JavaFX was released in the fourth quarter of 2008. The current release for JavaFX is version 8.0. The version number jumped from 2.2 to 8.0sourced JavaFX in 2013.

The first version of JavaFX was released in the fourth quarter of 2008. The current release for JavaFX is version 8.0. The version number jumped from 2.2 to 8.0.

* **BENEFITS :**

JavaFX is basically a standard compatible class library. It is included with the Eclipse IDE. JavaFX also allows developers to define, preview, and test user interfaces for web, mobile, and desktop.

1. **LIFECYCLE OF JAVAFX APPLICATION**

The JavaFX Application class has three life cycle methods, which are −

* **start()** − The entry point method where the JavaFX graphics code is to be written.
* **stop()** − An empty method which can be overridden, here you can write the logic to stop the application.
* **init()** − An empty method which can be overridden, but you cannot create a stage or scene in this method.

The entry point for JavaFX applications is the Application class. The JavaFX runtime does the following, in order, whenever an application is launched:

1. Constructs an instance of the specified Application class
2. Calls the init() method
3. Calls the start(javafx.stage.Stage) method
4. Waits for the application to finish, which happens when either of the following occur:

* the application calls Platform.exit()
* the last window has been closed and the implicitExit attribute on Platform is true

1. Calls the stop() method

Note that the start method is abstract and must be overridden. The init and stop methods have concrete implementations that do nothing.

* Calling Platform.exit() is the preferred way to explicitly terminate a JavaFX Application. Directly calling System.exit(int) is an acceptable alternative, but doesn't allow the Application stop() method to run.
* A JavaFX Application should not attempt to use JavaFX after the FX toolkit has terminated or from a ShutdownHook, that is, after the stop() method returns or System.exit(int) is called **Parameters.**
* Application parameters are available by calling the getParameters() method from the init() method, or any time after the init method has been called **Threading.**
* JavaFX creates an application thread for running the application start method, processing input events, and running animation timelines. Creation of JavaFX Scene and Stage objects as well as modification of scene graph operations to live objects (those objects already attached to a scene) must be done on the JavaFX application thread.
* The Java launcher loads and initializes the specified Application class on the JavaFX Application Thread. If there is no main method in the Application class, or if the main method calls Application.launch(), then an instance of the Application is then constructed on the JavaFX Application Thread.
* The init method is called on the launcher thread, not on the JavaFX Application Thread. This means that an application must not construct a Scene or a Stage in the init method. An application may construct other JavaFX objects in the init method.
* All the unhandled exceptions on the JavaFX application thread that occur during event dispatching, running animation timelines, or any other code, are forwarded to the thread's uncaught exception handler.

The following example will illustrate a simple JavaFX application.

importjavafx.application.Application;

import javafx.scene.Group;

import javafx.scene.Scene;

import javafx.scene.shape.Circle;

import.javafx.stage.Stage;

public class MyApp extends Application {

public void start(Stage stage) {

Circle circ = new Circle(40,40,30);

Group root = new Group(circ);

Scene scene = new Scene(root,400,300);

Stage.setTitle(“My JavaFX Application”);

Stage.setScene(scene);

Stage.show();

}

1. **JAVAFX PACKAGES AND UI COMPONENTS**

**UI COMPONENTS:**

| Label | Label is a component that is used to define a simple text on the screen.Typically, a label is placed with the node, it describes. |
| --- | --- |
| [Button](https://www.javatpoint.com/javafx-button) | Button is a component that controls the function of the application. Button class is used to create a labelled button. |
| RadioButton | The Radio Button is used to provide various options to the user. The user can only choose one option among all. A radio button is either selected or deselected. |
| CheckBox | Check Box is used to get the kind of information from the user which contains various choices. User marked the checkbox either on (true) or off(false). |
| TextField | Text Field is basically used to get the input from the user in the form of text. javafx.scene.control.TextField represents TextField |
| PasswordField | PasswordField is used to get the user's password. Whatever is typed in the passwordfield is not shown on the screen to anyone. |
| HyperLink | HyperLink are used to refer any of the webpage through your appication. It is represented by the class **javafx.scene.control.HyperLink** |
| Slider | Slider is used to provide a pane of options to the user in a graphical form where the user needs to move a slider over the range of values to select one of them. |
| ProgressBar | Progress Bar is used to show the work progress to the user. It is represented by the class **javafx.scene.control.ProgressBar**. |
| ProgressIndicator | Instead of showing the analogue progress to the user, it shows the digital progress so that the user may know the amount of work done in percentage. |
| ScrollBar | JavaFX Scroll Bar is used to provide a scroll bar to the user so that the user can scroll down the application pages. |
| Menu | JavaFX provides a Menu class to implement menus. Menu is the main component of any application. |
| ToolTip | JavaFX ToolTip is used to provide hint to the user about any component. It is mainly used to provide hints about the text fields or password fields being used in the application. |

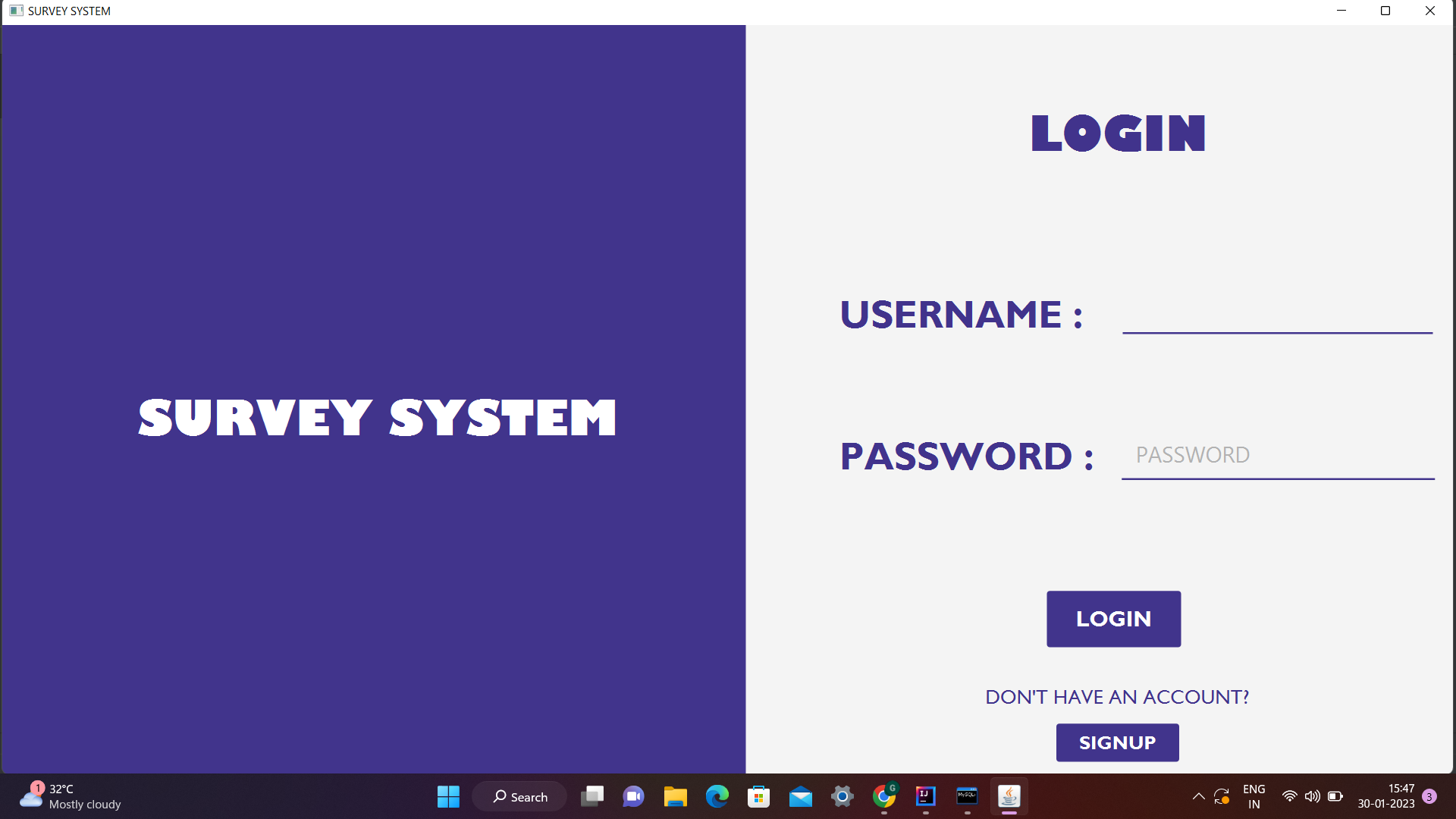
**CHAPTER 2**

**MODULES DESCRIPTION**

1. **INTRODCUTION**

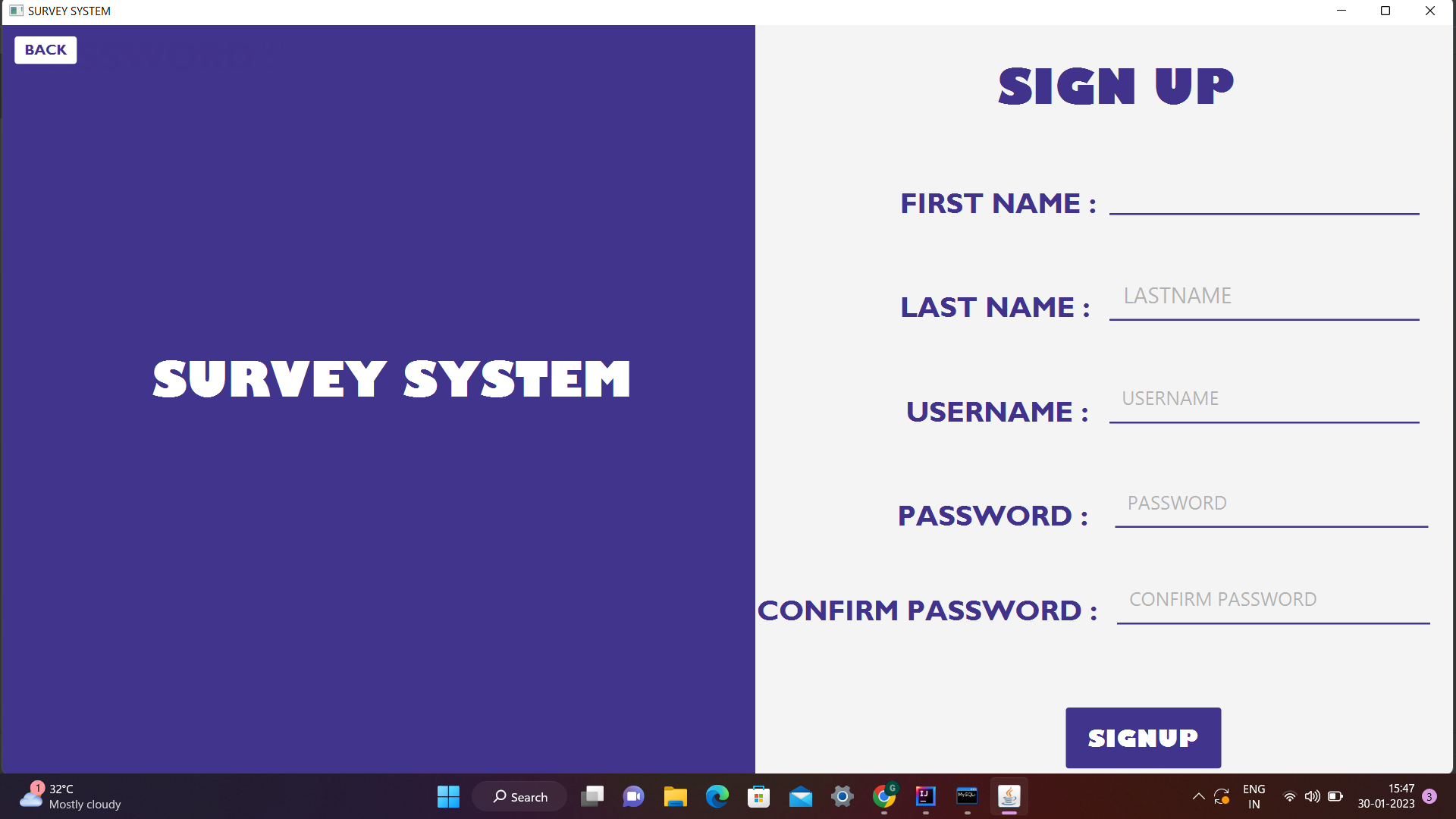
This proposed online project is an implementation of Java programming language for software generation that is important in college or organization to carry survey. In this system of survey, only the users authenticated by admin from the database system can drop their vote or express their viewpoint regarding the issue. Being online software, it can be logged on from anywhere with internet access.The proposed Online Survey System is easy and comfortable to use. In this software, the answers or viewpoints of the participants are collected using a radio buttonThe system plays a vital role in minimizing the budget of the survey. The implementation of the project avoids the programs such as meetings, conferences etc.to take any decision or research. With the help of this online system, one can easily forward ideas and viewpoints to the officials.

1. **(LOGIN) MODULE 1**



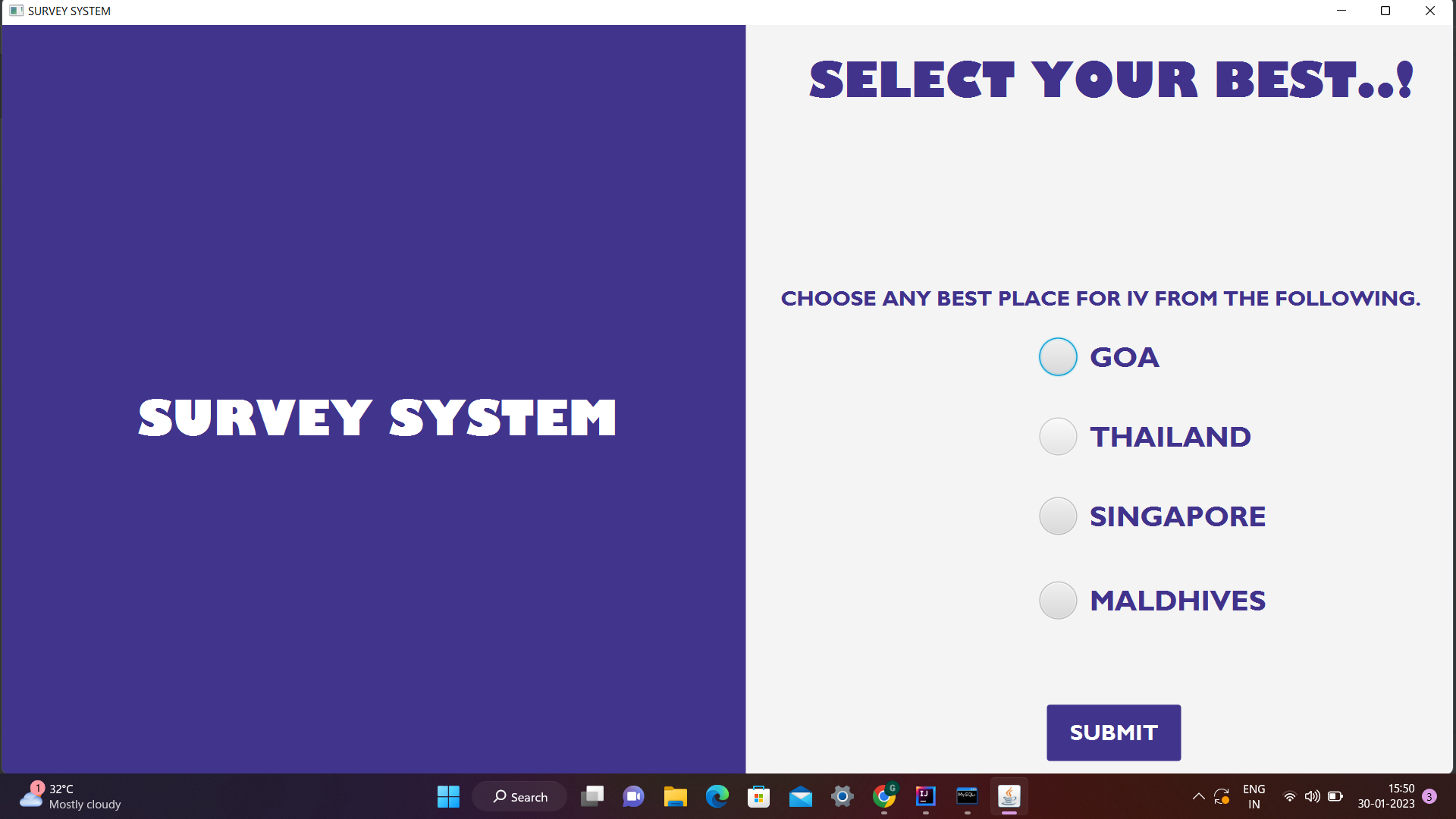
1. **(SIGNUP)MODULE 2**

New user can sign up and create an new account.



1. **(SURVEY)MODULE 3**

This module is used to choose the answer form the following options is given by radio buttons.



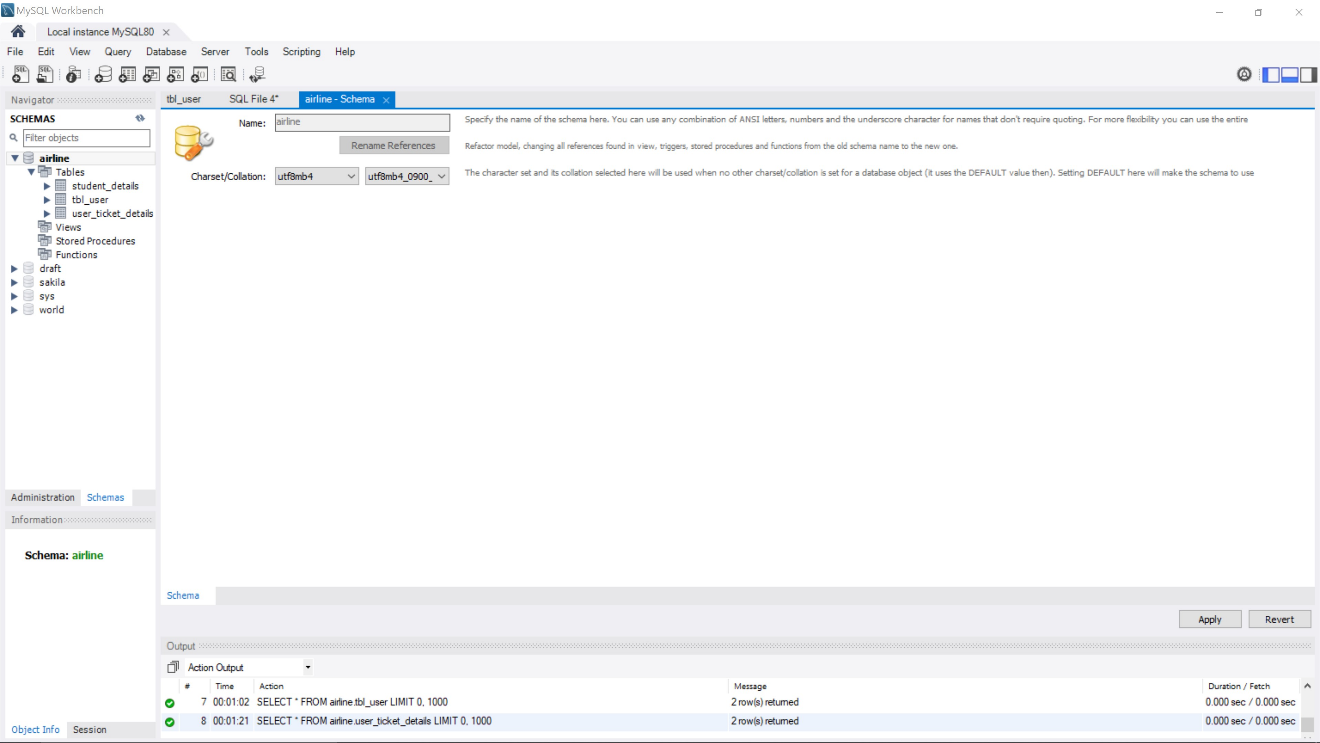
1. **DATABASE CREATION**

**Purpose of a database for any real-time application**:

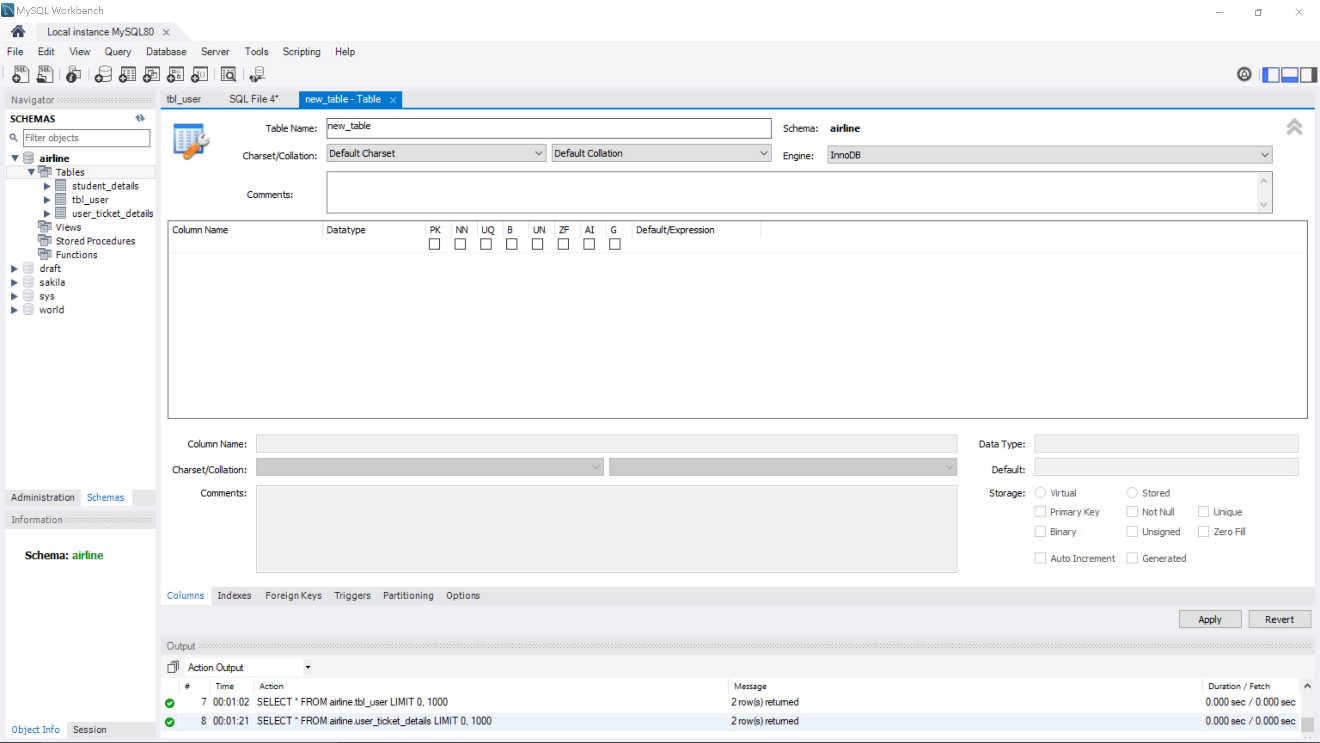
Real-time databases can move data from purpose-built app to purpose-built app in real time, so they can serve as the backbone for sharing data and messaging in microservices architectures, which are becoming more common

**Steps for Creating a Database and table:**

1. Download MySQL WorkBench.
2. Create Schema,

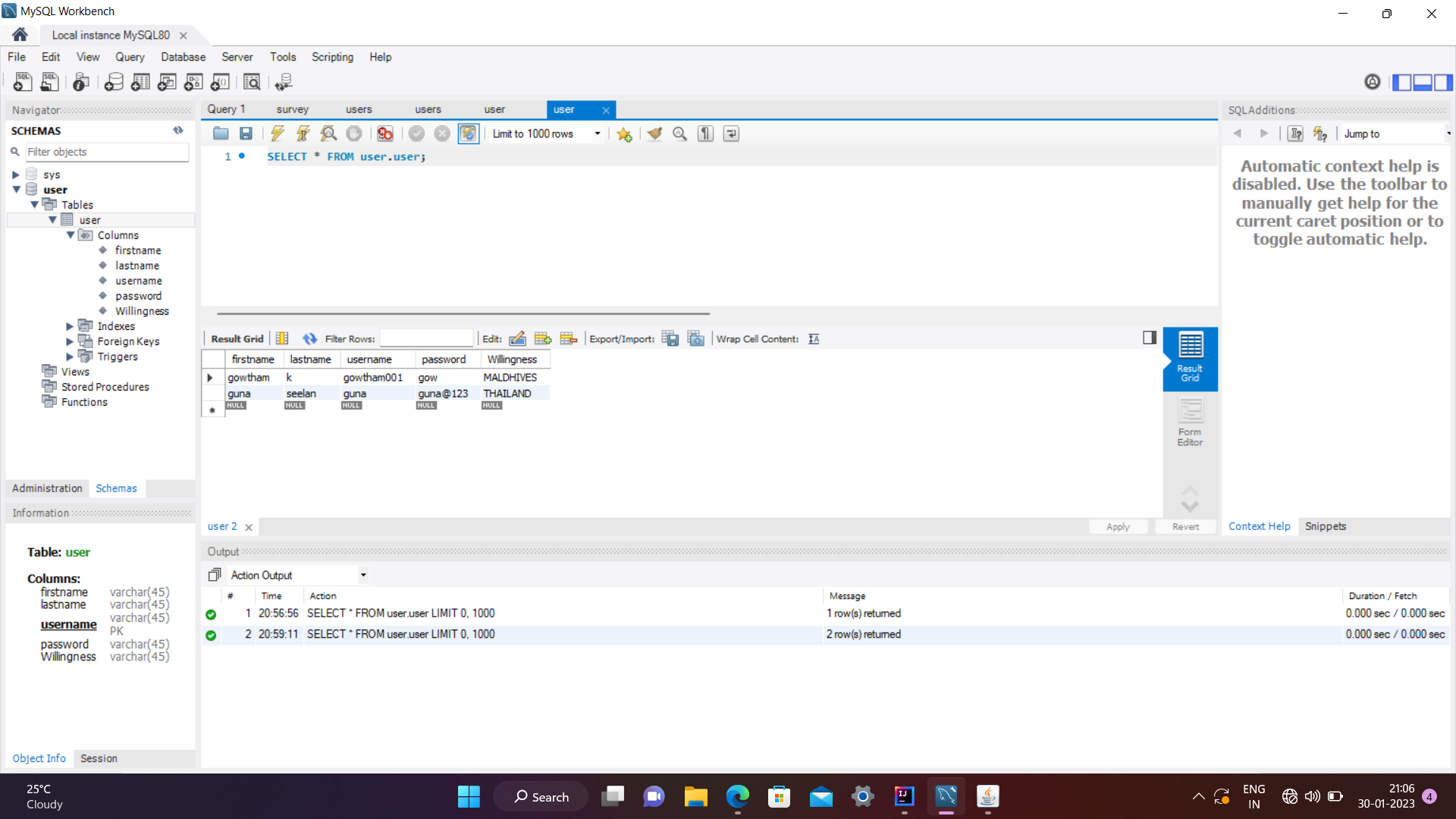
****

1. After creating the Database, create table by clicking the create table option and create as required.



1. Here, in this project two tables are created,

User Details Table,



1. Establish connection between the project and the Database by using mySQL connector.jar file and importing the module in intellij.
2. Establish connection between the project and the Database by using mySQL connector.jar file and importing the module in intellij.
3. Finally,import the connection in the project and establish the database link with the project.

**CHAPTER 2**

**CONCLUSION**

In conclusion, Online Survey System Project in Java is an excellent software to conduct online survey with minimized economy. The result of the project is accurate and totally error free. With this system, the whole survey process can be conducted secretly by hiding the identities of the people surveyed. The growing use of internet and computers confirms the good scopes of project.

**APPENDIX**

// LOGIN PAGE

import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Scene;

import javafx.stage.Stage;

import java.io.IOException;

public class login\_page extends Application {

@Override

public void start(Stage stage) throws IOException {

FXMLLoader fxmlLoader = new FXMLLoader(login\_page.class.getResource("login.fxml"));

Scene scene = new Scene(fxmlLoader.load(),1530 , 790);

stage.setTitle("SURVEY SYSTEM");

stage.setScene(scene);

stage.show();

}

public static void main(String[] args) {

launch();

}

}package com.example.survey\_system;

//LOGIN CONTROLLER

package com.example.survey\_system;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.Label;

import javafx.scene.control.PasswordField;

import javafx.scene.control.TextField;

import javafx.stage.Stage;

import java.io.IOException;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

public class login\_controller {

@FXML

private Button loginbutton;

public static String str;

public String getname(){

String name=str;

return name;

}

@FXML

private PasswordField password;

@FXML

private Button signupbutton;

@FXML

private TextField username;

@FXML

private Label altmsg;

private static String ename;

@FXML

void login(ActionEvent event) throws IOException {

str = username.getText();

DB\_file connectnow = new DB\_file();

Connection connectDB = connectnow.getConnection();

String verifylogin = "SELECT count(1) FROM user WHERE username = '"+username.getText()+"' and password = '"+password.getText()+"';";

try {

ename=username.getText();

Statement statement = connectDB.createStatement();

ResultSet queryResult = statement.executeQuery(verifylogin);

while (queryResult.next())

{

if(queryResult.getInt(1)==1){

FXMLLoader fxmlLoader = new FXMLLoader(login\_page.class.getResource("mainpage.fxml"));

Stage window = (Stage) loginbutton.getScene().getWindow();

window.setScene(new Scene(fxmlLoader.load(), 1530, 790));

}

} catch (Exception e)

{

altmsg.setText("INVALID USERNAME OR PASSWORD..!");

e.printStackTrace();

e.getCause();

}

}

@FXML

void signup(ActionEvent event) throws IOException {

FXMLLoader fxmlLoader = new FXMLLoader(login\_page.class.getResource("signup.fxml"));

Stage window = (Stage)signupbutton.getScene().getWindow();

window.setScene(new Scene(fxmlLoader.load(),1530 , 790));

}

}

//SIGNUP CONTROLLER

package com.example.survey\_system;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.Label;

import javafx.scene.control.PasswordField;

import javafx.scene.control.TextField;

import javafx.stage.Stage;

import java.io.IOException;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

public class signup\_controller {

@FXML

private Button backbutton;

@FXML

private PasswordField cpword;

@FXML

private TextField fname;

@FXML

private TextField lname;

@FXML

private PasswordField pword;

@FXML

private Button sbutton;

@FXML

private TextField uname;

@FXML

private Label msg;

@FXML

private Label msg;

@FXML

void gosignup(ActionEvent event) {

if((pword.getText().equals(cpword.getText())) && fname.getText()!=null && lname.getText()!=null && uname.getText()!=null){

DB\_file connectnow = new DB\_file();

Connection connectDB = connectnow.getConnection();

String verifylogin = "INSERT INTO user (firstname,lastname,username,password) VALUES('" + fname.getText() + "','" + lname.getText() + "','" + uname.getText() + "','" + pword.getText() + "');";

try {

Statement statement = connectDB.createStatement();

statement.executeUpdate(verifylogin);

msg.setText("NEW USER REGISTER SUCCESSFULLY..!!");

} catch (Exception e)

{

e.printStackTrace();

e.getCause();

}

}

else if(!pword.getText().equals((cpword.getText()))){

msg.setText("PASSWORD MISS MATCH..!!");

}

else if(pword.getText()==null || cpword.getText()==null || fname.getText()==null || lname.getText()==null || uname.getText()==null){

msg.setText("SOME FIELDS ARE MISSING..!!");

}

}

@FXML

void goback(ActionEvent event) throws IOException {

FXMLLoader fxmlLoader = new FXMLLoader(login\_page.class.getResource("login.fxml"));

Stage window = (Stage)backbutton.getScene().getWindow();

window.setScene(new Scene(fxmlLoader.load(),1530 , 790));

}

}

//MAIN PAGE CONTROLLER

package com.example.survey\_system;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.scene.control.\*;

import java.io.IOException;

import java.sql.Connection;

import java.sql.Statement;

public class mainpage\_controller {

@FXML

private RadioButton b1;

@FXML

private RadioButton b2;

@FXML

private RadioButton b3;

@FXML

private RadioButton b4;

@FXML

private Button submitform;

@FXML

private ToggleGroup toggle;

@FXML

private Label erms;

@FXML

private Button repbut;

@FXML

public TextField setname;

public String usname;

public String setname (){

mainpage\_controller mc = new mainpage\_controller();

usname=mc.setname();

return "";

}

String str;

@FXML

void butsubmit(ActionEvent event) {

if(b1.isSelected()){

str = "GOA";

}

else if(b2.isSelected()){

str = "THAILAND";

}

else if(b3.isSelected()){

str = "SINGAPORE";

}

else if(b4.isSelected()){

str = "MALDHIVES";

}

if(str!=null){

DB\_file connectnow = new DB\_file();

Connection connectDB = connectnow.getConnection();

login\_controller ak = new login\_controller();

usname = ak.getname();

String verifylogin = "update user set Willingness ='"+str+"' where username='"+usname+"';";

try {

Statement statement = connectDB.createStatement();

statement.executeUpdate(verifylogin);

erms.setText("THANKS FOR YOUR ANSWER..!!");

} catch (Exception e)

{

e.printStackTrace();

e.getCause();

}

}

}

@FXML

void showrep(ActionEvent event) throws IOException {

}

}

**DATABASE CONNECTION**

package com.example.survey\_system;

import java.sql.Connection;

import java.sql.DriverManager;

public class DB\_file {

public Connection databaseLink;

public Connection getConnection(){

String databaseName = "user";

String databaseUser = "root";

String databasePassword = "guna@2004";

String url = "jdbc:mysql://localhost/"+databaseName;

try{

Class.forName("com.mysql.cj.jdbc.Driver");

databaseLink = DriverManager.getConnection(url,databaseUser,databasePassword);

}catch (Exception e)

{

e.printStackTrace();

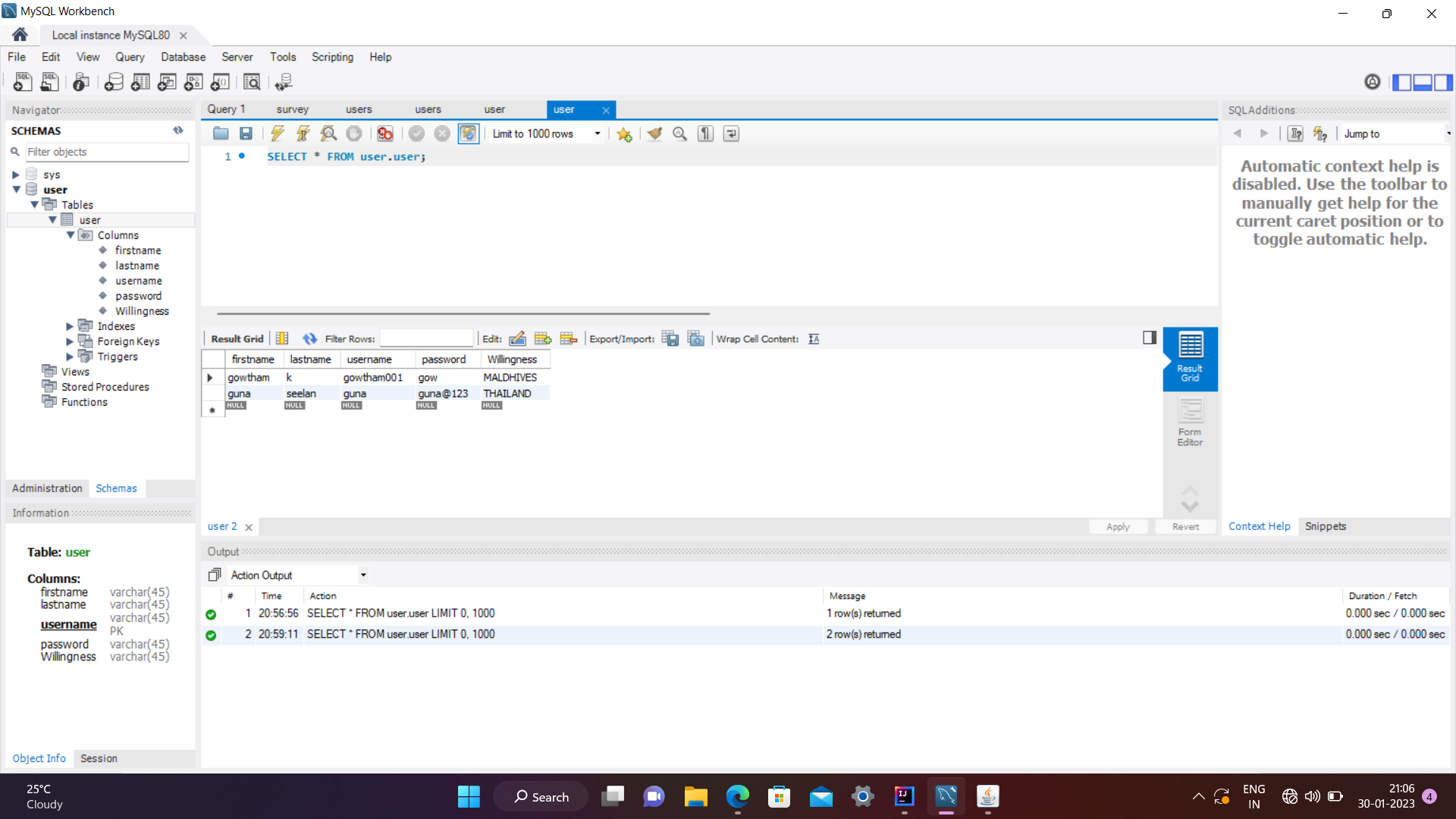
e.getCause();

}

return databaseLink;

}

}

****