**DataBase TermProject 4조 중간발표자료**

202235002 강지원

202135733 김서영

# MySQL script

DROP DATABASE Twitter;

CREATE DATABASE Twitter;

USE Twitter;

CREATE TABLE User (

UserID varchar(20),

Name varchar(12),

Email varchar(20),

Password varchar(23),

PRIMARY KEY (UserID)

) DEFAULT CHARACTER SET UTF8;

CREATE TABLE Tweet (

TweetID varchar(20),

WriterID varchar(20),

Content varchar(255),

Timestamp varchar(25),

PRIMARY KEY (TweetID),

Foreign key (writerID) references User(UserID)

) DEFAULT CHARACTER SET UTF8;

CREATE TABLE Timeline (

UserID varchar(20),

timestamp varchar(25),

TweetID varchar(20),

Primary key (UserID),

Foreign key(TweetID) references Tweet(TweetID)

) DEFAULT CHARACTER SET UTF8;

CREATE TABLE BlackList (

FollowID varchar(20),

Timestamp varchar(25),

PRIMARY KEY (FollowID)

) DEFAULT CHARACTER SET UTF8;

CREATE TABLE Following (

UserID varchar(20),

FollowerID varchar(20),

Primary key (UserID,FollowerID),

Foreign key (FollowerID) references User(UserID),

Foreign key(UserID) references User(USerID)

) DEFAULT CHARACTER SET UTF8;

CREATE TABLE Follower (

UserID varchar(20),

FollowingID varchar(20),

Primary key (UserID,FollowingID),

Foreign key (FollowingID) references User(UserID),

Foreign key(UserID) references User(USerID)

) DEFAULT CHARACTER SET UTF8;

CREATE TABLE Comment (

CommentID varchar(20),

TweetID varchar(20),

WriterID varchar(20),

Content varchar(50),

PRIMARY KEY (CommentID),

foreign key(TweetID) references Tweet(TweetID),

foreign key(WriterID) references Timeline(UserID)

) DEFAULT CHARACTER SET UTF8;

/\*User 추가\*/

insert into User(UserID, Name, email, password) values('202235040', 'ParkGeonwoo','yourkik@gachon.ac.kr', '12345');

select \*from User;

/\*Timeline\*/

insert into Timeline(UserID, timestamp) values('202235040', '2023.11.21.10:40');

select \*from Timeline;

/\*Tweet 추가\*/

insert into Tweet values('0','202235040','Test','2023.11.21');

select \*from Tweet;

/\*update table Timeline add '202235040', \*/

update Timeline set TweetID = '0' where Timeline.UserID='202235040';

select \*from Timeline;

# Client GUI application

텍스트, 도표, 스크린샷, 평면도이(가) 표시된 사진

자동 생성된 설명

*User management*

## Login/Membership (Initial Window)

텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

자동 생성된 설명텍스트, 소프트웨어, 멀티미디어 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

public class SimpleTwitterApp extends JFrame {

private Connection connection;

private JTextField useridField, usernameField, useremailField, passwordField, followField, tweetField;

private JTextArea feedArea;

private boolean isLoggedIn = false;

public SimpleTwitterApp() {

//Initialize GUI components

useridField = new JTextField(10);

usernameField = new JTextField(10);

useremailField = new JTextField(20);

passwordField = new JPasswordField(10);

followField = new JTextField(10);

tweetField = new JTextField(30);

feedArea = new JTextArea(10, 30);

JButton createUserButton = new JButton("Create User");

JButton followButton = new JButton("Follow");

JButton postButton = new JButton("Post");

//Connect to the database

connectToDatabase();

//Set up the GUI layout

setLayout(new BorderLayout());

JPanel userPanel = new JPanel();

userPanel.setBackground(new Color(126, 210, 255)); //배경 색상

if (!isLoggedIn) {

//로그인 필드 및 버튼

JTextField loginIdField = new JTextField(10);

JPasswordField loginPasswordField = new JPasswordField(10);

JButton loginButton = new JButton("Login");

loginButton.setBackground(new Color(255, 255, 255)); //버튼 색상

loginButton.setForeground(Color.***BLUE***); //텍스트 색상

userPanel.add(new JLabel("UserID:"));

userPanel.add(loginIdField);

userPanel.add(new JLabel("Password:"));

userPanel.add(loginPasswordField);

userPanel.add(loginButton);

//로그인 버튼 액션 처리

loginButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

//로그인 로직 수행

String inputId = loginIdField.getText();

String inputPassword = new String(loginPasswordField.getPassword());

if (checkLogin(inputId, inputPassword)) {

isLoggedIn = true;

JOptionPane.*showMessageDialog*(SimpleTwitterApp.this, "로그인 되었습니다.", "로그인 성공", JOptionPane.***INFORMATION\_MESSAGE***);

loginButton.setText("Logout");

} else {

JOptionPane.*showMessageDialog*(SimpleTwitterApp.this, "아이디 또는 비밀번호가 일치하지 않습니다.", "로그인 실패", JOptionPane.***ERROR\_MESSAGE***);

}

}

});

} else {

//로그인된 상태에서는 로그아웃 버튼 표시

JButton logoutButton = new JButton("Logout");

userPanel.add(logoutButton);

//로그아웃 버튼 액션 처리

logoutButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

isLoggedIn = false;

JOptionPane.*showMessageDialog*(SimpleTwitterApp.this, "로그아웃 되었습니다.", "로그아웃", JOptionPane.***INFORMATION\_MESSAGE***);

}

});

}

userPanel.add(createUserButton);

createUserButton.setBackground(new Color(255, 255, 255)); //버튼 색상

createUserButton.setForeground(Color.***BLUE***); //텍스트 색상

JPanel followPanel = new JPanel();

followPanel.add(new JLabel("Follow User:"));

followPanel.add(followField);

followPanel.add(followButton);

JPanel tweetPanel = new JPanel();

tweetPanel.add(new JLabel("Tweet:"));

tweetPanel.add(tweetField);

tweetPanel.add(postButton);

JPanel mainPanel = new JPanel();

mainPanel.setLayout(new BorderLayout());

mainPanel.add(new JScrollPane(feedArea), BorderLayout.***SOUTH***);

mainPanel.add(userPanel, BorderLayout.***CENTER***);

add(mainPanel, BorderLayout.***CENTER***);

//회원가입 다이얼로그 생성

createUserButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

//다이얼로그 생성

JDialog createUserDialog = new JDialog(SimpleTwitterApp.this, "Create User", true);

createUserDialog.setLayout(new BorderLayout());

//입력 필드 및 레이블 추가

JTextField userIdField = new JTextField(10);

JTextField userNameField = new JTextField(10);

JTextField userEmailField = new JTextField(20);

JPasswordField userPasswordField = new JPasswordField(10);

JPanel inputPanel = new JPanel(new GridLayout(4, 2));

inputPanel.add(new JLabel("UserID:"));

inputPanel.add(userIdField);

inputPanel.add(new JLabel("Username:"));

inputPanel.add(userNameField);

inputPanel.add(new JLabel("User Email:"));

inputPanel.add(userEmailField);

inputPanel.add(new JLabel("Password:"));

inputPanel.add(userPasswordField);

createUserDialog.add(inputPanel, BorderLayout.***CENTER***);

//확인 버튼 추가

JButton okButton = new JButton("OK");

okButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

//입력된 정보를 사용하여 사용자 생성

String userId = userIdField.getText();

String userName = userNameField.getText();

String userEmail = userEmailField.getText();

String userPassword = new String(userPasswordField.getPassword());

//사용자 생성 메서드 호출

createUser(userId, userName, userEmail, userPassword);

//다이얼로그 종료

createUserDialog.dispose();

}

});

//취소 버튼 추가

JButton cancelButton = new JButton("Cancel");

cancelButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

//다이얼로그 종료

createUserDialog.dispose();

}

});

JPanel buttonPanel = new JPanel();

buttonPanel.add(okButton);

buttonPanel.add(cancelButton);

createUserDialog.add(buttonPanel, BorderLayout.***SOUTH***);

//다이얼로그 크기와 위치 설정

createUserDialog.setSize(300, 150);

createUserDialog.setLocationRelativeTo(SimpleTwitterApp.this);

createUserDialog.setVisible(true);

}

});

}

private void connectToDatabase() {

try {

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

connection = DriverManager.*getConnection*("jdbc:mysql://127.0.0.1:3306/twitter?serverTimezone=UTC&useSSL=false", "root", "1234");

} catch (ClassNotFoundException | SQLException e) {

e.printStackTrace();

JOptionPane.*showMessageDialog*(this, "Error connecting to the database.", "Error", JOptionPane.***ERROR\_MESSAGE***);

}

}

private boolean checkLogin(String inputId, String inputPassword) {

try {

String query = "SELECT \* FROM User WHERE UserID = ? AND Password = ?";

try (PreparedStatement preparedStatement = connection.prepareStatement(query)) {

preparedStatement.setString(1, inputId);

preparedStatement.setString(2, inputPassword);

try (ResultSet resultSet = preparedStatement.executeQuery()) {

return resultSet.next();

}

}

} catch (SQLException ex) {

ex.printStackTrace();

return false;

}

}

private void createUser(String userID, String name, String email, String password) {

try {

String query = "INSERT INTO User (UserID, Name, Email, Password) VALUES (?, ?, ?, ?)";

try (PreparedStatement preparedStatement = connection.prepareStatement(query)) {

preparedStatement.setString(1, userID);

preparedStatement.setString(2, name);

preparedStatement.setString(3, email);

preparedStatement.setString(4, password);

preparedStatement.executeUpdate();

}

JOptionPane.*showMessageDialog*(this, "User created successfully.", "Success", JOptionPane.***INFORMATION\_MESSAGE***);

} catch (SQLException e) {

e.printStackTrace();

JOptionPane.*showMessageDialog*(this, "Error creating user.", "Error", JOptionPane.***ERROR\_MESSAGE***);

}

}

public static void main(String[] args) {

SwingUtilities.*invokeLater*(() -> {

SimpleTwitterApp app = new SimpleTwitterApp();

app.setSize(500, 400);

app.setTitle("Simple Twitter");

app.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

app.setVisible(true);

});

}

}

## Change password

*“Follow” mechanism*

## After logging in, personal TimeLine (Function: Comment upload, follow window, board window) ((로그인 후 개인 TimeLine (기능 : Comment upload, follow창 띄우기, board 창띄우기)))

텍스트, 전자제품, 스크린샷, 디스플레이이(가) 표시된 사진

자동 생성된 설명

import javax.swing.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

import java.util.ArrayList;

import java.util.List;

public class FollowFunc extends JFrame {

private static final String ***JDBC\_URL*** = "jdbc:mysql://127.0.0.1:3306/twitter";

private static final String ***USERNAME*** = "your\_username";

private static final String ***PASSWORD*** = "your\_password";

private JComboBox<String> followingComboBox;

private JComboBox<String> followerComboBox;

private JTextField targetUserIDField;

private static Connection *con*;

private void connectToDatabase() {

try {

// Load JDBC driver and establish a connection

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

*con* = DriverManager.*getConnection*("jdbc:mysql://127.0.0.1:3306/twitter?serverTimezone=UTC&useSSL=false", "root", "1234");

} catch (ClassNotFoundException | SQLException e) {

e.printStackTrace();

JOptionPane.*showMessageDialog*(this, "Error connecting to the database.", "Error", JOptionPane.***ERROR\_MESSAGE***);

}

}

public FollowFunc() {

// Connect to the database

connectToDatabase();

setTitle("FollowFunc");

setSize(400, 300);

setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

followingComboBox = new JComboBox<>();

followerComboBox = new JComboBox<>();

targetUserIDField = new JTextField(15);

JButton followButton = new JButton("Follow");

followButton.addActionListener(new ActionListener() {

*@Override*

public void actionPerformed(ActionEvent e) {

String targetUserID = targetUserIDField.getText();

if (!targetUserID.isEmpty()) {

*Follow*(loggedInUserID, targetUserID);

// Refresh the combo boxes after following

fetchUserIDs();

} else {

JOptionPane.*showMessageDialog*(FollowFunc.this, "Enter a valid target user ID.", "Error", JOptionPane.***ERROR\_MESSAGE***);

}

}

});

JPanel panel = new JPanel();

panel.add(new JLabel("Target User ID:"));

panel.add(targetUserIDField);

panel.add(followButton);

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

panel.add(followingComboBox);

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

panel.add(followerComboBox);

add(panel);

setLayout(new BoxLayout(getContentPane(), BoxLayout.***Y\_AXIS***));

setLocationRelativeTo(null);

setVisible(true);

fetchUserIDs(); // Fetch user IDs when the GUI is created

}

private void fetchUserIDs() {

try (Connection connection = DriverManager.*getConnection*(***JDBC\_URL***, ***USERNAME***, ***PASSWORD***)) {

String followingQuery = "SELECT FollowingID FROM Following WHERE UserID = ?";

String followerQuery = "SELECT FollowerID FROM Follower WHERE UserID = ?";

// Fetch following user IDs

fetchUserIDsFromDatabase(connection, followingComboBox, followingQuery);

// Fetch follower user IDs

fetchUserIDsFromDatabase(connection, followerComboBox, followerQuery);

} catch (SQLException ex) {

ex.printStackTrace();

}

}

private void fetchUserIDsFromDatabase(Connection connection, JComboBox<String> comboBox, String query) throws SQLException {

try (PreparedStatement preparedStatement = connection.prepareStatement(query)) {

preparedStatement.setString(1, loggedInUserID);

ResultSet resultSet = preparedStatement.executeQuery();

comboBox.removeAllItems(); // Clear existing items

while (resultSet.next()) {

comboBox.addItem(resultSet.getString(1));

}

}

}

public static void Follow(String userID, String followID) { // 유저1이 유저2를 팔로우하는 상황 가정

String insertFollowingQuery = "INSERT INTO Following (UserID, FollowerID) VALUES (?, ?)"; // 유저1의 팔로잉 목록에 유저2를 업데이트

String insertFollowerQuery = "INSERT INTO Follower (UserID, FollowingID) VALUES (?, ?)"; // 유저2의 팔로워 목록에 유저1을 업데이트

try {

PreparedStatement preparedStatement = *con*.prepareStatement(insertFollowingQuery);

preparedStatement.setString(1, userID);

preparedStatement.setString(2, followID);

PreparedStatement preparedStatement2 = *con*.prepareStatement(insertFollowerQuery);

preparedStatement2.setString(1, followID);

preparedStatement2.setString(2, userID);

preparedStatement.executeUpdate();

preparedStatement2.executeUpdate();

System.***out***.println(userID + "(이)가 " + followID +"(을)를 팔로우했습니다.");

} catch (SQLException e) {

e.printStackTrace();

System.***err***.println("팔로우에 실패했습니다.");

}

}

public static void FollowingList(String UserID) { // 유저의 팔로잉 목록 확인

String selectFollowingQuery = "SELECT FollowerID FROM Following WHERE UserID=?";

try {

PreparedStatement preparedStatement = *con*.prepareStatement(selectFollowingQuery);

preparedStatement.setString(1, UserID);

ResultSet resultSet = preparedStatement.executeQuery();

if (resultSet.next()) {

System.***out***.println(UserID + "의 팔로잉 목록");

do {

String FollowerID = resultSet.getString("FollowerID");

System.***out***.println(FollowerID);

} while(resultSet.next());

} else {

System.***out***.println(UserID + "사용자의 팔로잉 목록이 존재하지 않습니다.");

}

} catch (SQLException e) {

e.printStackTrace();

System.***err***.println("팔로잉 목록 오류가 발생했습니다.");

}

}

public static void FollowerList(String UserID) { // 유저의 팔로워 목록 확인

String selectFollowerQuery = "SELECT FollowingID FROM Follower WHERE UserID=?";

try {

PreparedStatement preparedStatement = *con*.prepareStatement(selectFollowerQuery);

preparedStatement.setString(1, UserID);

ResultSet resultSet = preparedStatement.executeQuery();

if (resultSet.next()) {

System.***out***.println(UserID + "의 팔로워 목록");

do {

String FollowingID = resultSet.getString("FollowingID");

System.***out***.println(FollowingID);

} while(resultSet.next());

} else {

System.***out***.println(UserID + "사용자의 팔로워 목록이 존재하지 않습니다.");

}

} catch (SQLException e) {

e.printStackTrace();

System.***err***.println("팔로워 목록 오류가 발생했습니다.");

}

}

public static void main(String[] args) {

SwingUtilities.*invokeLater*(new Runnable() {

*@Override*

public void run() {

new FollowFunc();

}

});

}

}

“Article”

## TimeLine Comment

## Viewing follower/following information and other user follow functions ((Follower / Following정보 열람 및 other user follow기능))

## A user can write a short message on his/her board