```
/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package presenter;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.time.LocalDate;
import java.time.ZoneId;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import model.Admins;
import utility.DBConnection;

/**
 *
 * @author Adarsha
 */
public class AdminPersister {

    private Connection connection; // Connection object creation
    private PreparedStatement insertAdmin;
    private PreparedStatement getAllAdmins;
    private PreparedStatement deleteAdmin;
    private PreparedStatement loginAdminByUsernameAndPassword;
    private PreparedStatement findAllAdminsByName;
    private PreparedStatement findAllAdminsById;
    private PreparedStatement updateAdmin;

    public AdminPersister(){
        try {
                this.connection = DBConnection.getConnection(); //
database connection
                if (connection != null) {

                    insertAdmin = connection.prepareStatement("INSERT
INTO admins (admin_name, username, password, email, phone, address,
campus) "
                            + "VALUES(?, ?, ?, ?, ?, ?, ?)");
                    getAllAdmins = connection.prepareStatement("SELECT *
FROM admins");
                    deleteAdmin = connection.prepareStatement("DELETE
FROM admins WHERE admin_id = ?");
```

```
                    loginAdminByUsernameAndPassword =
connection.prepareStatement("SELECT * FROM admins WHERE username = ? AND
password = ?");
                    findAllAdminsById =
connection.prepareStatement("SELECT * FROM admins WHERE admin_id LIKE ?
");
                    findAllAdminsByName =
connection.prepareStatement("SELECT * FROM admins WHERE username LIKE ?
");

                  //  updateAdmin = connection.prepareStatement("UPDATE
admins SET username = ? WHERE admin_name = ? ");
                    updateAdmin = connection.prepareStatement("UPDATE
admins SET admin_name = ?, username = ?, password = ?, email = ?, phone =
?, address = ?, campus = ? WHERE admin_id = ?");
                }
            } catch (SQLException e) {
                System.out.println("Connection Failed!");
                System.out.println("SQLException : " + e.getMessage());
            }
    }

    boolean registerAdmin(Admins admin) {
        try {
            insertAdmin.setString(1, admin.getAdminName());
            insertAdmin.setString(2, admin.getAdminUsername());
            insertAdmin.setString(3, admin.getAdminPassword());
            insertAdmin.setString(4, admin.getAdminEmail());
            insertAdmin.setString(5, admin.getAdminPhone());
            insertAdmin.setString(6, admin.getAdminAddress());
            insertAdmin.setString(7, admin.getAdminCampus());
            insertAdmin.executeUpdate();  // execute the prepared
statement insert
            return true;

        } catch (SQLException e) {
            System.out.println("SQL Exception: " + e.getMessage());
            return false;
        }

    }

    List<Admins> getAllAdmins() {
        List<Admins> adminList = new ArrayList<>();
        try {
            ResultSet adminResult = getAllAdmins.executeQuery();

            System.out.println("Admin details reading from the
database.");
            while (adminResult.next()) {
                int adminId = adminResult.getInt("admin_id");
                String adminName = adminResult.getString("admin_name");
                String adminEmail = adminResult.getString("email");
                String adminPhone = adminResult.getString("phone");
                String adminUsername = adminResult.getString("username");
                String adminPassword = adminResult.getString("password");
                String adminAddress = adminResult.getString("address");
                String adminCampus = adminResult.getString("campus");
                Admins newAdmin = new Admins(adminName, adminEmail,
adminPhone, adminUsername, adminPassword, adminAddress, adminCampus);
                newAdmin.setAdmin_id(adminId);
              //  System.out.println("New added admin is : "+newAdmin);
                adminList.add(newAdmin);
            }
        } catch (SQLException e) {
            System.out.println("SQL Exception: " + e.getMessage());
        }
     //   System.out.println("Final admins list to be sent from persister
is :"+ adminList);
        return adminList;
    }

    String deleteAdmin(int admin_id) {
        String adminStatus = "";
        try {

            deleteAdmin.setInt(1, admin_id);
            int adminResult = deleteAdmin.executeUpdate();

            if (adminResult > 0) {
                adminStatus = "Admin deleted successfully.";
            } else {
                adminStatus = "Cannot delete the Admin.";
            }

        } catch (SQLException e) {
            adminStatus = "The admin cannot be deleted.";
            System.out.println("The admin cannot be deleted : " +
e.getMessage());
        }
        return adminStatus;
    }

    boolean isAdminExist(String username, String password) {
        try {
            loginAdminByUsernameAndPassword.setString(1, username);
            loginAdminByUsernameAndPassword.setString(2, password);
            ResultSet loginResult =
loginAdminByUsernameAndPassword.executeQuery();
            if (loginResult.next()) {
                return true;
            }
        } catch (SQLException e) {
            System.out.println("SQL Exception: " + e.getMessage());
        }
        return false;
    }

    void writeToFile(String notification) {
        try {

          File myObj = new File("AdminLog.txt");
          if (myObj.createNewFile()) {
            System.out.println("File created: " + myObj.getName());
          } else {
            System.out.println("File already exists.");
          }
        } catch (IOException e) {
          System.out.println("An error occurred.");
          e.printStackTrace();
        }

        Path p = Paths.get("AdminLog.txt");
        try (BufferedWriter writer = Files.newBufferedWriter(p,
StandardOpenOption.APPEND)) {
            writer.write(notification+"\n");

            System.out.println("Successfully wrote to the file.");
        } catch (IOException e) {
          System.out.println("An error occurred.");
          e.printStackTrace();
        }
  }

    public Date getDateFromLocalDate(LocalDate date) {
        Date newDate = null;
        if (date != null) {
            newDate =
Date.from(date.atStartOfDay(ZoneId.systemDefault()).toInstant());
        }
        return newDate;
    }

    public List<Admins> findAllAdminsById(int admin_id) {
        //String id = Integer.toString(adminId);
       Admins admin = new Admins();
       List<Admins> searchedAdmins = new ArrayList();
        try {
            findAllAdminsById.setInt(1, admin_id);
            ResultSet adminResult = findAllAdminsById.executeQuery();

            System.out.println("Admin details reading from the
database.");
            while (adminResult.next()) {
                int adminId = adminResult.getInt("admin_id");
                String adminName = adminResult.getString("admin_name");
                String username = adminResult.getString("username");
                String email = adminResult.getString("email");
                String phone = adminResult.getString("phone");
                String address = adminResult.getString("address");
                String campus = adminResult.getString("campus");
                String password = adminResult.getString("password");


                admin = new
Admins(adminName,email,phone,username,password,address,campus);
                admin.setAdmin_id(adminId);

                searchedAdmins.add(admin);
            }
        } catch (SQLException e) {
            System.out.println("SQL Exception: " + e.getMessage());
        }
        return searchedAdmins;
    }

    public List<Admins> findAllAdminsByName(String keyword) {
       Admins admin = new Admins();
       List<Admins> searchedAdmins = new ArrayList();
        try {
            findAllAdminsByName.setString(1, "%"+keyword+"%");
            ResultSet adminResult = findAllAdminsByName.executeQuery();

            System.out.println("Admin details reading from the
database.");
            while (adminResult.next()) {
                int adminId = adminResult.getInt("admin_id");
                String adminName = adminResult.getString("admin_name");
                String username = adminResult.getString("username");
                String email = adminResult.getString("email");
                String phone = adminResult.getString("phone");
                String address = adminResult.getString("address");
                String campus = adminResult.getString("campus");
                String password = adminResult.getString("password");


                admin = new
Admins(adminName,email,phone,username,password,address,campus);
                admin.setAdmin_id(adminId);

                searchedAdmins.add(admin);
            }
        } catch (SQLException e) {
            System.out.println("SQL Exception: " + e.getMessage());
        }
        return searchedAdmins;
    }
//
//    boolean updateAdmin(String column,String username, String
adminName) {
//        try{
//            updateAdmin.setString(1, username);
//            updateAdmin.setString(2, adminName);
//
//            updateAdmin.execute();
//        }catch (SQLException e) {
//            System.out.println("SQL Exception: " + e.getMessage());
//            return false;
//        }
//        return true;
//    }

    boolean updateAdmin(Admins admin) {
        try {
            int myID = admin.getAdmin_id();
            updateAdmin.setInt(8, myID);

            updateAdmin.setString(1, admin.getAdminName());
            updateAdmin.setString(2, admin.getAdminUsername());
            updateAdmin.setString(3, admin.getAdminPassword());
            updateAdmin.setString(4, admin.getAdminEmail());
            updateAdmin.setString(5, admin.getAdminPhone());
            updateAdmin.setString(6, admin.getAdminAddress());
            updateAdmin.setString(7, admin.getAdminCampus());
            updateAdmin.executeUpdate();  // execute the prepared
statement insert
            return true;

        } catch (SQLException e) {
            System.out.println("SQL Exception: " + e.getMessage());
            return false;
        }

      }

    }
```