



Python and SQL: intro / SQL platforms [2400-DS1SQL]


ELECTRONIC JOURNAL

Marcin Miszkiel
Katarzyna Mocio



Project presentation 25.01.2024



A black and white photograph of a young girl with pigtails, sitting at a desk in a classroom. She is looking up and to the left, with her right arm raised high, hand open, as if wanting to answer a question. In the background, another child is visible, also at a desk, looking towards the camera. The background consists of a wall with many small, square tiles.

The application was written as a final project for the **Python and SQL: intro / SQL platforms** class by students **Marcin Miszkiel** and **Katarzyna Mocio**.

Electronic Journal, a Flask-based Python web app with an SQLite database.

It streamlines school-related tasks for teachers, students, and administrators, offering grade input, statistics, and class management features.

Idea description



Short description:

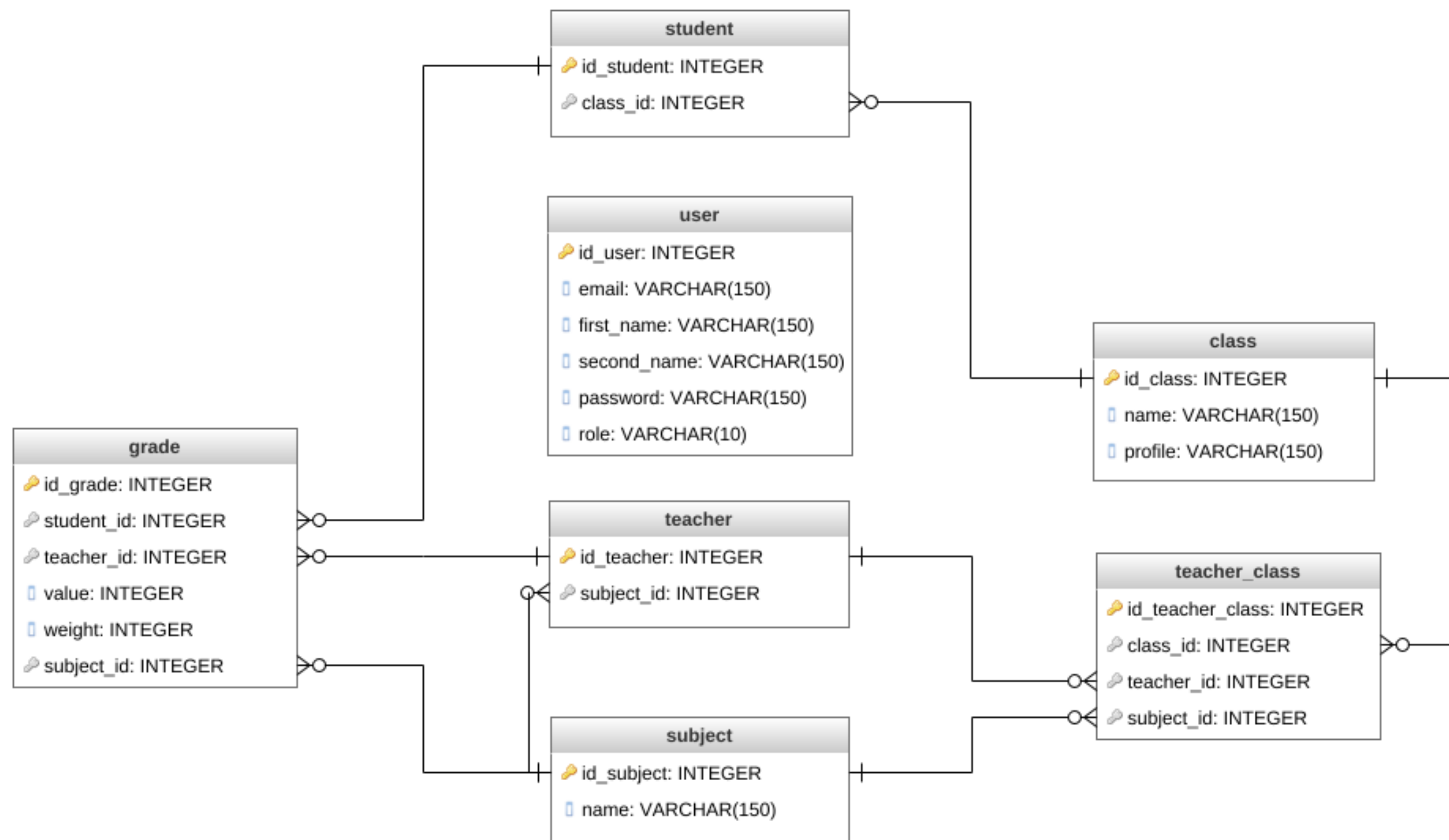
Our database consists of 7 tables connected by multiple references. It contains basic information about *students*, *classes*, *teachers*, *subjects* and *grades*, keeping the information resulting from their connections.

Note:

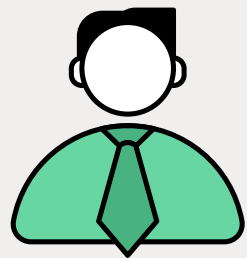
The **role** column in the **user** table takes values of **admin/teacher/student**. Based on this, the **teacher** and **student** tables are associated with the **user** through inheritance. We didn't create an additional **admin** table as it was not needed.

Info

Database scheme

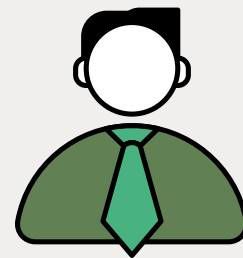


Login panel



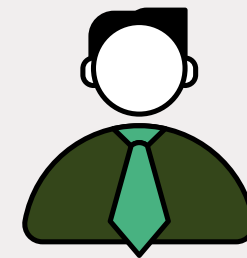
ADMIN

- login/logout
- adding a new user
- adding a new subject
- assigning teacher to class



TEACHER

- login/logout
- entering grades
- grades summary/dashboard



STUDENT

- login/logout
- grades summary/dashboard

Roles functions

Welcome to the Electronic Journal!



Source: Google.com

Thank you for choosing our electronic journal platform. This system is designed to provide a seamless experience for students, teachers, and administrators to manage and access academic information efficiently.

Who is this platform for?

This platform caters to the needs of students, teachers, and administrators. Whether you are here to view grades, manage classes, or oversee the academic progress of students, we've got you covered.

Login Instructions

To get started, please use the navigation bar above and click on the appropriate login type:

- **Student Login:** Access your grades and academic information. Click on "Student's view" in the left up corner.
- **Teacher Login:** Manage classes, enter grades, and interact with students. Click on "Teacher's view" in the left up corner.
- **Admin Login:** Oversee the overall functioning of the electronic journal. Click on "Admin's view" in the left up corner.

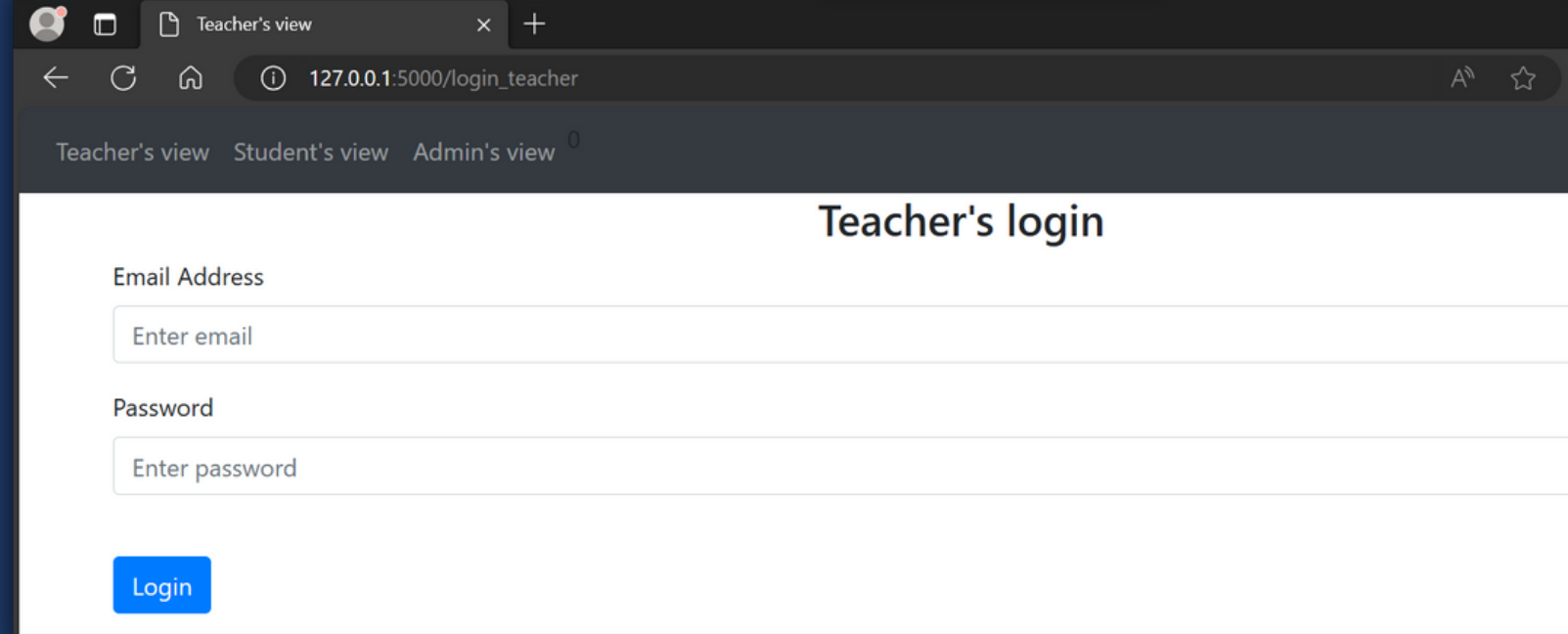
If you don't have an account, please contact the administration to get the necessary credentials.

Contact the administration at: admin@admin.uw.edu.pl

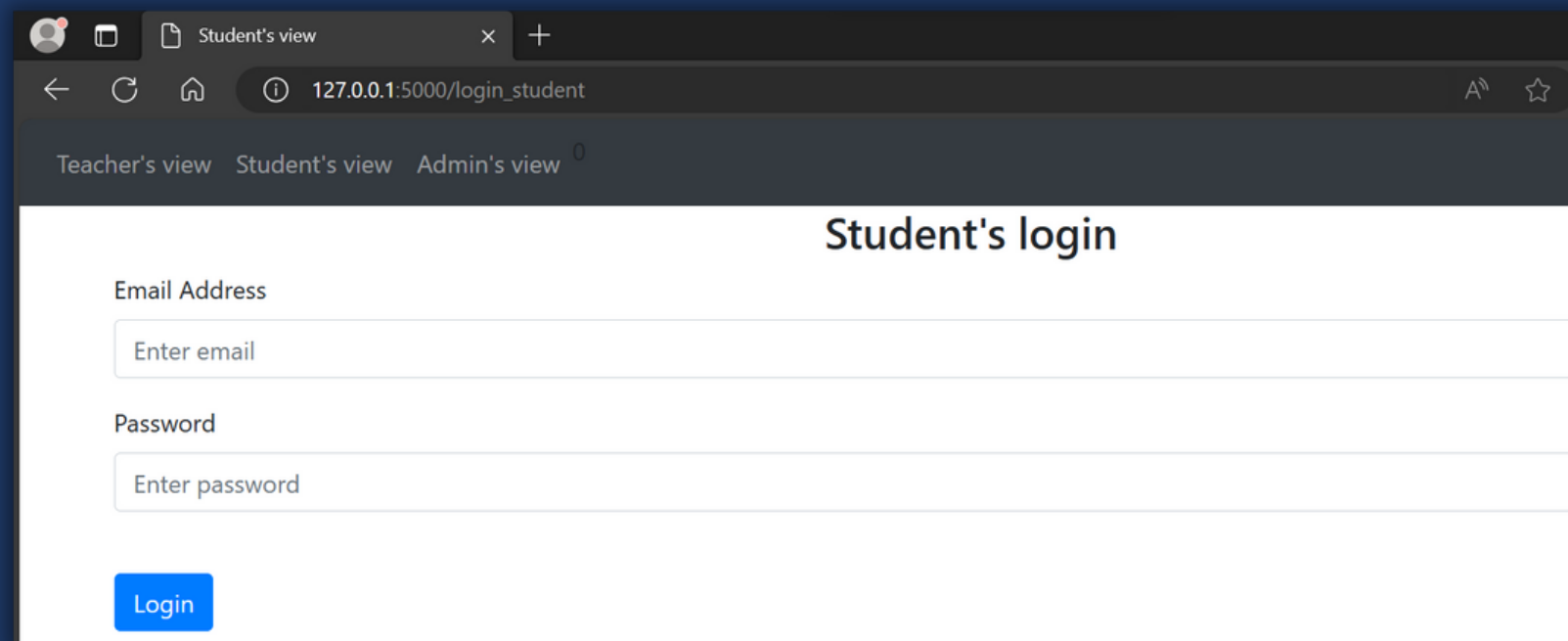
If you encounter any issues or have questions, feel free to contact our support team.

Contact our support team at: [Marcin Miszkief](#)

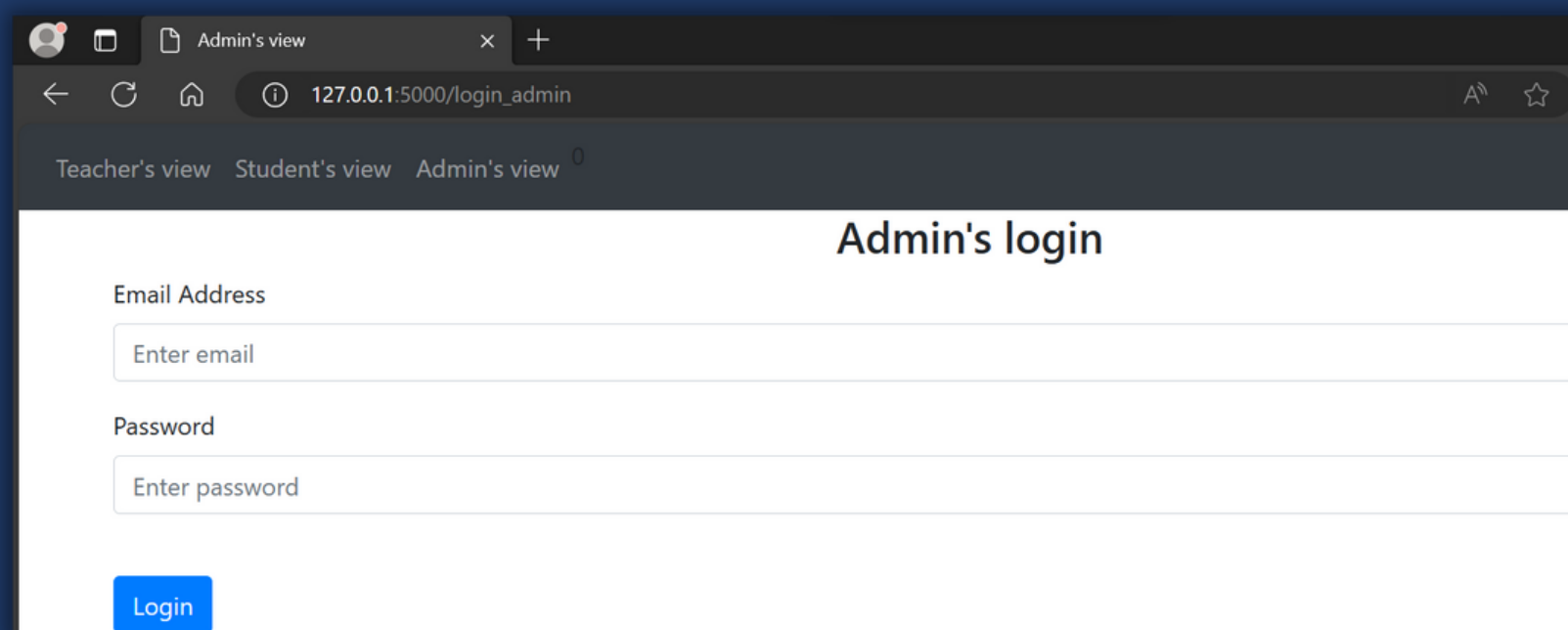
Contact our support team at: [Katarzyna Mocio](#)



This screenshot shows the 'Teacher's login' panel in a web browser. The browser's address bar displays '127.0.0.1:5000/login_teacher'. The page has a dark header with navigation links for 'Teacher's view', 'Student's view', and 'Admin's view'. The main content area is white and titled 'Teacher's login'. It contains two input fields: 'Email Address' with a placeholder 'Enter email' and 'Password' with a placeholder 'Enter password'. A blue 'Login' button is positioned at the bottom left of the form.



This screenshot shows the 'Student's login' panel. The browser address bar shows '127.0.0.1:5000/login_student'. The header navigation links are 'Teacher's view', 'Student's view', and 'Admin's view'. The main content area is white and titled 'Student's login'. It features the same login form structure as the teacher panel, with 'Email Address' and 'Password' input fields and a 'Login' button.



This screenshot shows the 'Admin's login' panel. The browser address bar shows '127.0.0.1:5000/login_admin'. The header navigation links are 'Teacher's view', 'Student's view', and 'Admin's view'. The main content area is white and titled 'Admin's login'. It contains the standard login form with 'Email Address' and 'Password' input fields and a 'Login' button.

Login panel

Login panel consists of 3 subpages, each for different role

- teacher
- student
- admin

The app login is an email address, unique to each user.

Passwords are encrypted for security reasons.



Logged in successfully!

Welcome to the Admin Panel, Katarzyna!

Admins can create user accounts. This includes students and teachers. They can also assign teachers to classes based on their profession (subject).

List of Users

ID	Email	First Name	Last Name	Role
1	admin@admin.uw.edu.pl	admin	admin	admin
2	m.miszkiel2@student.uw.edu.pl	Marcin	Miszkiel	admin
3	k.mocio@student.uw.edu.pl	Katarzyna	Mocio	admin
4	jkowalski@uw.edu.pl	Jan	Kowalski	teacher
5	anowak@uw.edu.pl	Adam	Nowak	teacher
6	bking@uw.edu.pl	Bryan	King	teacher
7	tholland@uw.edu.pl	Tom	Holland	teacher
8	mgreen@uw.edu.pl	Matt	Green	teacher
9	wpers@uw.edu.pl	Witold	Pers	teacher
10	kwysoki@uw.edu.pl	Kacper	Wysoki	teacher
11	pstrong@uw.edu.pl	Peter	Strong	teacher
12	jbatman@uw.edu.pl	Joanna	Batman	teacher
13	hwielki@uw.edu.pl	Hubert	Wielki	teacher
14	asmok@uw.edu.pl	Anna	Smok	teacher
15	gface@uw.edu.pl	Gabriela	Face	teacher
16	skwiat@uw.edu.pl	Sebastian	Kwiat	teacher
17	hwolt@uw.edu.pl	Halina	Wolt	teacher
18	mstar@uw.edu.pl	Maciej	Star	teacher

Admin Panel

#admin



Browser window showing the "Add new User" page. The page title is "Adding new User". A green notification bar at the top says "Logged in successfully!". The form includes:

- Role: Student ☒ Teacher ☐
- Email Address:
- First Name:
- Second Name:
- Password:
- Class:
-

Adding a new user

#admin

1.

Choose a role: student or teacher

2.

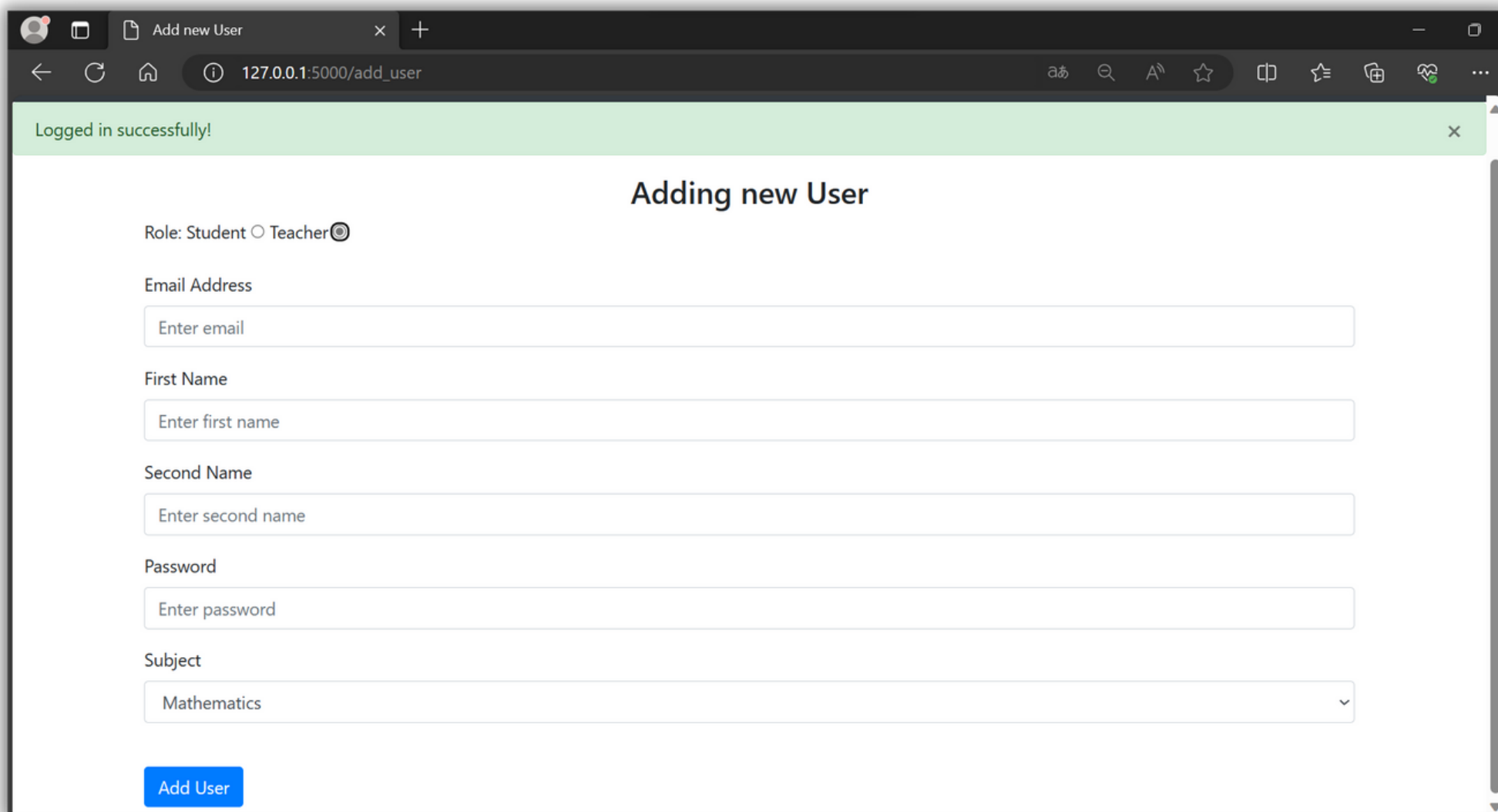
Enter email address, first and second name and password

3.

For student select class /
For teacher select subject

4.

Press "Add User" button



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000/add_user'. A green notification bar at the top says 'Logged in successfully!'. The main heading is 'Adding new User'. Below it, there is a role selection section with 'Student' selected. The form includes input fields for 'Email Address', 'First Name', and 'Second Name', each with a placeholder text. There is a 'Password' field with a placeholder text. A 'Subject' dropdown menu is set to 'Mathematics'. At the bottom left, there is a blue 'Add User' button.

Adding a new user

#admin

1.

Choose a role: student or teacher

2.

Enter email address, first and second name and password

3.

For student select class /
For teacher select subject

4.

Press “Add User” button

List of existing subjects

ID	Name
1	Mathematics
2	Physics
3	Chemistry
4	Biology
5	Computer Science
6	English Language
7	History
8	Geography
9	Literature
10	Physical Education
11	Art
12	Music
13	Spanish Language
14	Economics
15	Psychology
16	Ethics
17	Civic Sciences
18	Religious Education
19	Social Sciences
20	Natural Sciences

New Subject Name:

Add Subject

1.

Enter subject name

2.

Press “Add Subject” button

Adding a new subject

#admin



The screenshot shows a web browser window with the title 'Assign Teacher to Class'. The address bar shows '127.0.0.1:5000/assign_teacher_to_class'. The page has a dark header with navigation links: 'Adding new user', 'Assign Teacher to Class', and 'Logout'. The main content area is titled 'Assign Teacher to Class'. It contains two dropdown menus: 'Select Class:' with '3B' selected, and 'Select Subject:' with 'Chemistry' selected. Below these is a blue button labeled 'Choose this Class and Subject'.

The screenshot shows the same web browser window, but the form has changed. It now has a dropdown menu labeled 'Select Teacher' with 'Tom Holland' selected. Below this is a blue button labeled 'Save Assignment'.

Assign teacher to class

#admin

1.

Select class and subject from list of all classes and all subjects

2.

Press “Choose this Class and Subject” button

3.

Select teacher from list of teachers teaching selected subject

4.

Press “Save Assignment” button

Enter Grades

Select Class:

1A

Enter grade

Enter Grades

Weight:

Enter weight

Student Name	Enter Grade
Adam Adams	
Adrian Allen	
Arthur Crawford	

Submit Grades

Entering grades

#teacher

1.

Select class from list of all classes
2.

Press “Enter Grade” button
3.

Enter grades for all or selected students from choosen class
4.

Press “Submit Grades” button

Welcome, Adam Nowak!

Teached subject: Mathematics

List of classes: 1A - Information Technology

Selected class: 1A (Information Technology)

Class 1A average

Class average is **3.41** from Mathematics

Class 1A median

Class median is **3.0** from Mathematics

Class 1A mode

Class mode is **2** from Mathematics

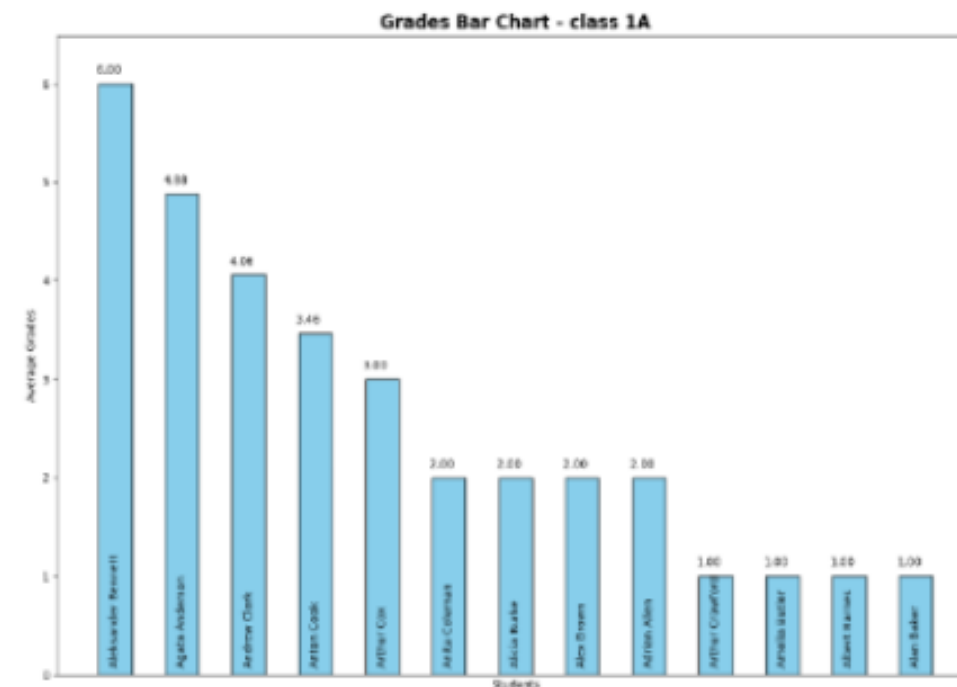
Highest Average in Mathematics

The student with the highest average is **Aleksander Bennett** with an average of 6.0

Lowest Average in Mathematics

The student with the lowest average is **Alan Baker** with an average of 1.0

Below you can find ranking based on average grade from Mathematics



Electronic Journal

1.

Select class from list of classes taught by logged in teacher

2.

Press "Select Class" button

Summary dashboard

#teacher

Welcome Louise Long!

You are assigned to class 2D, Linguistic profile

All subjects panel

Your recent grade

You got **1** on **Biology** by Matt Green

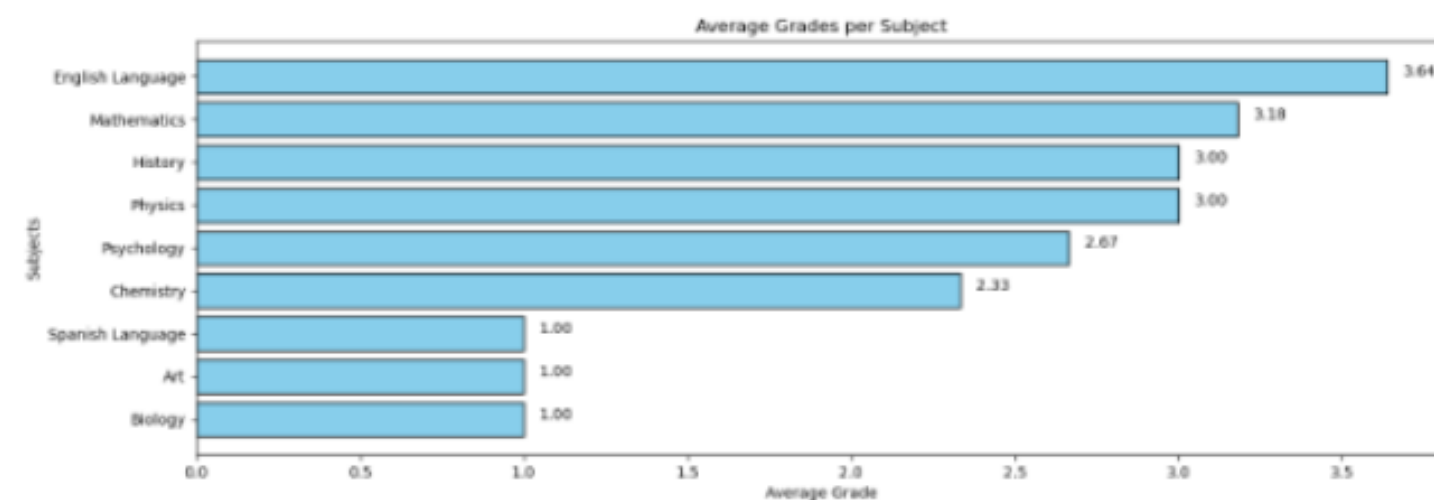
Your total average

Your total average is **2.63**

Your position in class ranking

Your position is **20** out of **20** students

Below you can find chart presenting your average grade from each subject



List of subjects:

Art



Select Subject

Your average from Art

Your average is **1.0**

Your class average from Art

Class average is **3.7**



Electronic Journal

1.

Select subject from list of subjects for which logged in student is enrolled

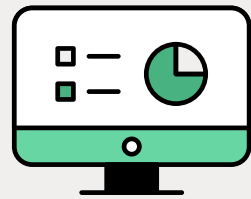
2.

Press "Select Subject" button

Summary dashboard

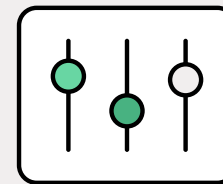
#student

Security features



ONLY LOGGED-IN USERS ACCESS

The website accepts only logged-in users. A user who is not logged in cannot view further panels, authorization is required. While logged in, has access only to the panels assigned to his/her role.



PASSWORD ENCRYPTION

Passwords encrypted with werkzeug.security using the 'pbkdf2:sha256' method
- passwords are not shown without encryption even in database file.



AUTOMATIC LOGOUT OF INACTIVE USERS

Inactive users are automatically logged out after a specified period of time.

Used SQL queries

(example INSERT)

```
if role == 'student': # If it's student, get selected class and add information to database
    class_id = request.form.get('classes')
    cursor.execute("INSERT INTO user(email, first_name, second_name, password, role) VALUES (?, ?, ?, ?, ?)",
        (email, firstname, secondname, generate_password_hash(password, method='pbkdf2:sha256'), role))
    # Get our student's ID and add information to second database
    user_id = cursor.lastrowid
    cursor.execute("INSERT INTO student(id, class_id) VALUES (?, ?)", (user_id, class_id))
elif role == 'teacher': # If it's teacher, get selected subject and add information to database
    subject_id = request.form.get('subjects')
    cursor.execute("INSERT INTO user(email, first_name, second_name, password, role) VALUES (?, ?, ?, ?, ?)",
        (email, firstname, secondname, generate_password_hash(password, method='pbkdf2:sha256'), role))
    # Get our teacher's ID and add information to second database
    user_id = cursor.lastrowid
    cursor.execute("INSERT INTO teacher(id, subject_id) VALUES (?, ?)", (user_id, subject_id))
else:
    flash('Invalid role', category='error')
    return render_template("add_user.html", subjects=subjects, classes=classes)
```

Used SQL queries

(example VIEWS)

```
# Changing ID numbers to values in Grade View
self.cursor.execute('''
    CREATE VIEW IF NOT EXISTS vw_grades AS
    SELECT g.id AS ID, g.value AS Value, g.wage AS Wage,
           s.id || ": " || u_s.first_name || " " || u_s.second_name AS "ID: Student",
           t.id || ": " || u_t.first_name || " " || u_t.second_name AS "ID: Teacher"
    FROM grade g
    LEFT JOIN student s ON g.student_id = s.id
    LEFT JOIN user u_s ON s.id = u_s.id
    LEFT JOIN teacher t ON g.teacher_id = t.id
    LEFT JOIN user u_t ON t.id = u_t.id;
''')
```

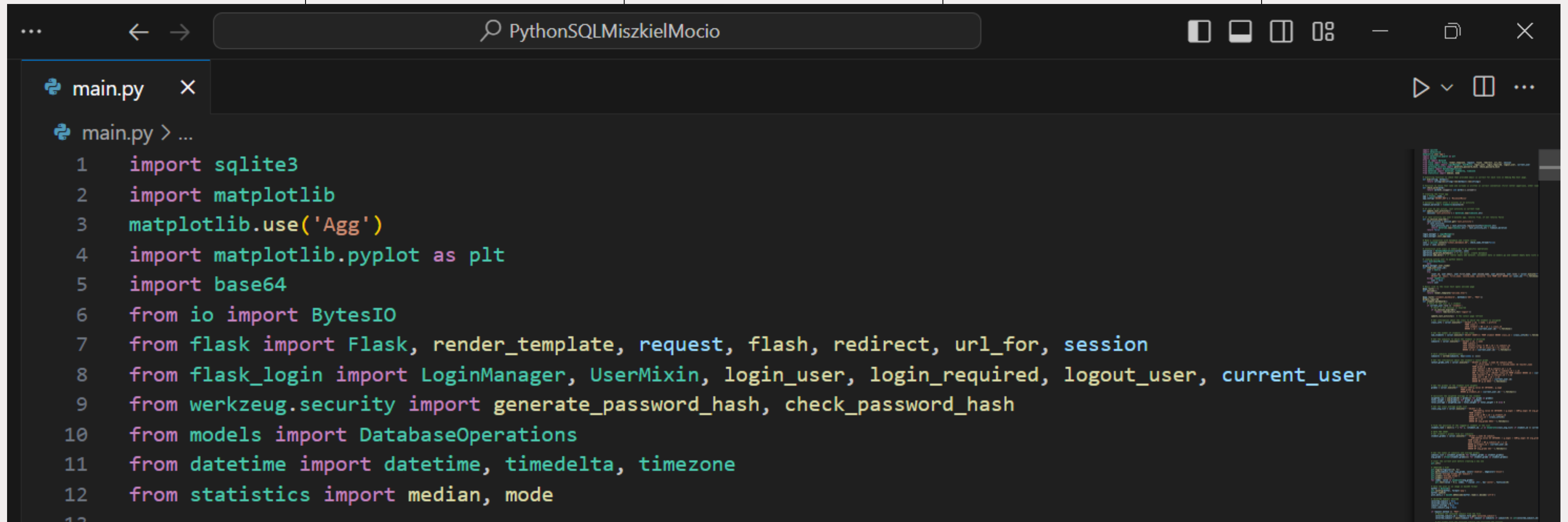
Used SQL queries

(example SUBQUERIES and COMPLEX QUERIES)

```
# Get the information about the student's latest grade
latest_grade_info = cursor.execute(f'''SELECT g.value, s.name AS subject_name,
    (u.first_name || ' ' || u.second_name) AS teacher_name
FROM grade g
JOIN subject s ON g.subject_id = s.id
JOIN teacher_class tc ON tc.subject_id = s.id
AND tc.class_id = (SELECT class_id FROM student WHERE id = {current_user.id})
JOIN teacher t ON tc.teacher_id = t.id
JOIN user u ON t.id = u.id
WHERE g.student_id = {current_user.id}
ORDER BY g.id DESC''').fetchone()
```

```
# Query for students ranking based on average grades
students_rank = cursor.execute(f'''SELECT u.id, u.first_name, u.second_name,
    SUM(CAST(g.value AS INTEGER) * g.wage) / SUM(g.wage) AS avg_grade
FROM user u
JOIN student s ON u.id = s.id
LEFT JOIN grade g ON g.student_id = s.id
WHERE s.class_id = {selected_class[0]} AND g.subject_id = {teacher_info[0]}
GROUP BY u.id
ORDER BY avg_grade DESC''').fetchall()
```


Python packages



The image shows a code editor window with a dark theme. The title bar at the top reads "PythonSQLMiszkielMocio". The editor has a file explorer on the left showing "main.py" and a breadcrumb "main.py > ...". The main area displays Python code with line numbers 1 through 13. The code imports various packages: sqlite3, matplotlib (with 'Agg' backend), Flask, flask_login, werkzeug.security, models, datetime, and statistics. A right-hand pane shows a preview of the rendered HTML output, which includes a login form with fields for username and password, and a 'Loguj się' button.

```
1 import sqlite3
2 import matplotlib
3 matplotlib.use('Agg')
4 import matplotlib.pyplot as plt
5 import base64
6 from io import BytesIO
7 from flask import Flask, render_template, request, flash, redirect, url_for, session
8 from flask_login import LoginManager, UserMixin, login_user, login_required, logout_user, current_user
9 from werkzeug.security import generate_password_hash, check_password_hash
10 from models import DatabaseOperations
11 from datetime import datetime, timedelta, timezone
12 from statistics import median, mode
13
```

Work distribution			Marcin Miszkiel ● Katarzyna Mocio ●
● • Login/Logout ● • Security features -> only logged-in users access -> Password encryption -> automatic logout of inactive users	● • Database scheme in app.genmymodel ● • Database in SQLite	● • Adding a new user ● • Admin Panel ● • Adding a new subject ● • Entering grades ● • Assign teacher to class ● • Home Page ● • Teacher dashboard ● • Student dashboard	• Project proposal, ● ● presentation and final report (relevant parts as we shared the code and database)

Note:

“Adding a new user” means preparing a route code in Python, writing SQL queries and template in HTML with JS and similar way with other functionalities.

work distribution



Electronic Journal

The application was written as a final project for the Python and SQL: intro / SQL platforms class by students Marcin Miszkiel and Katarzyna Mocio.

Thank you for your attention

