**Whom?**

I have some friends who work in the exchange market. One of the hard problems is sharing information about forex exchange. People can use this website to share forex trading information confidentially. Using this website has the following advantages.

* Share information quickly without delay.
* Access is only available to authorized access.

**What?**

I used python flask and SQLiteStudio. This website has 3 pages (user login, user registration, and user profile) and it is connected to SQLiteStudio database.

* User login page:

You can first see a **user login** page when you visit the website. If you have access, you can visit the **user** **profile** page by entering your registered email address and password and clicking the **login** button. But if you don’t have access, you need to click on **register here**.

If you enter the wrong email or password, you will show “Please enter correct email / password !” message. If you enter the correct email or password, you will show “Logged in successfully !” message.

|  |
| --- |
|  |

* User registration page:

If you don’t have access, you need to fill in your name, email, and password. After it, you must click on the **Register** button. After it, your data will be added to the **user1** table of the database. If your data is saved successfully, you will show “You have successfully registered !” message. After that, go to the login page and enter your registered email and password to visit the user profile page.

|  |
| --- |
| Graphical user interface, application  Description automatically generated |

* User profile page

After visiting the **user** **profile** page, you will see the name of the user who is logged in. There is also a **log-out** button and clicking on it, it will take you to the **login** page. This page has three sections.

Section 1: After login, you will see a table of all requests for currency exchange. This table is saved as exchange1 in the database. You can use the information in this table to exchange the currency you need in a short time. Description of data in the table of all requests for currency exchange.

* + Name – the name of the person making the request.
  + Date – date of request
  + Count – the amount of currency
  + Exch – which currency exchange
  + Type – buy or sell currency
  + Phone – contact phone number

Section 2: from this table, you can see only your requests.

Section 3: In this section, you can enter a new request or delete an old request. To do this, you need to enter information such as Name, Date, Count, Currency, Type, and Phone. If you want to enter a new request, click the **save** button. But if you want to delete an old request, click the **delete** button.

|  |  |
| --- | --- |
| Table  Description automatically generatedSection1 | Table  Description automatically generatedSection2 |
| Graphical user interface, text, application, email  Description automatically generatedSection3 | |

**Which?**

I used SQLiteStudio (3.4.1) database in this project. And it works on my local server. Firstly, I created DB\_WEB schema. In this schema, there are two tables such as user1, and exchange1.

|  |
| --- |
| Graphical user interface  Description automatically generated with medium confidence |

**User1** table stores information about registered users. For example, name, email, and password.

|  |
| --- |
| Table  Description automatically generated with medium confidence |

**Exchange1** table stores requests. For example, name, date, count, exch, type, and phone.

|  |
| --- |
| Table  Description automatically generated |

**How?**

All materials of this project are in “login-register-app”. There are 3 files in it.

* templates folder – it includes HTML code of the website
* app.py – main python code
* DB\_WEB – it is our database file

There are 3 HTML files in the templates folder.

* login.html – HTML code of login page
* register.html – HTML code of register page
* user.html – HTML code of user page

|  |
| --- |
|  |
|  |
|  |

**Description of codes?**

* login HTML

All html code must be in between <html> and </html>. HTML code has two parts such as head and body.

In the head part, I wrote the character set, styles, scripts, and other information.

Text

Description automatically generated

Created class, its name is a container.



Created text output, text’s size is h2.



The post method is to send an email, and password data to a server to check these are the in user1 table. Action name is ‘login’. We will use this ‘login’ name in python code. Also, it will show mesage object. We created text output as mesage in python code.

Text

Description automatically generated

I created two inputs. Labels are Email: and Password:. Ids are email and password. You can see placeholder texts in input section as text. Also, Login button was created. Type is submit.

Also, I created link section using href. It saved as ‘register’. We will use it in python code.

Text

Description automatically generated

* register HTML

The structure of the code is like login.html. The action name is “register”, and we will use it in python code. Post is sent name, email, and password data to the database. There are 3 inputs such as name, email, and password. Each one has its own id. Also, there is a button it is called Register. Also, there is a link section, and its name is login. We will use it in python code.

Text

Description automatically generated

* User HTML

The head of user.html is like the previous one. There is a style of table, and it is created in between <style> and </style>. There are fonts, padding size, font weight, and many color codes.

Text

Description automatically generated

You will see the user profile text output and its size is h1. Also, you will see “Logged in:” text. And, after it, you will session.name. We will use it in python code. In other words, it will be shown who is logged in. After it, you will see logout link and its name is ‘logout’. We will use it in python code. Also, you will see text messages and their name is mesage. We will use it in python code. There is h3-sized text.

Text

Description automatically generated

This section has 2 tables. So, I must give them different id names. This table’s id is “table1”. In between <thead> and </thead>, I wrote the columns name of table1. And after it, I wrote code to create rows of table 1. It is a loop and it will be shown row in each row. We will use “rows” in python code.

Text

Description automatically generated

This action’s name is “update” and we will use it in python code. Also, this section sent the below data to the database. There are 6 inputs. And there are two buttons. These two buttons have different actions. Therefore, their names are different such as save\_ex and send\_ex.

Text

Description automatically generated

There is h2-sized text. Also, there is a table, its id is table2. We will use this id in python code. It will be shown a row in each entry. We will use entries in python code.

Graphical user interface, text

Description automatically generated

* app python

First, I imported libraries to use. I mentioned the directory where HTML codes are. After it, I connected python to the database, and it is saved as con.

Graphical user interface, text, application, email

Description automatically generated

Login() - In html, I wrote action="{{ url\_for('login') }}". This section will describe what to do in login. First, I created a login function and mesage object. Mesage is an empty stringer object, and it will be shown on the user and login pages. Email is the input value of email. And the password is the input value of the password. Con is a connection with the database. In cursor.execute, I select rows from user1 table whose email is the same as the input and whose password is the same as the input. It is saved as a user object. If the session’s name and email are the same as the user’s name and email, create rows which are select all rows from exchange1. Also, it will create rows1 table which is data from exchange1 table whose name is the same as the user’s name. After it, you will be user.html and rows are rows, rows1 is entries, mesage is mesage. In other words, rows and rows1 is the value of table1 and table2. The

If the user the enter wrong email or password, you will see login.html and you will receive ‘Please, enter correct email/password!’ mesage.

Update() – in this section, we will insert a new request to the exchange1 table, or delete an old request from exchange1 table. Each input has its own object names such as in\_name, and in\_date. If the user clicks on the save button, it means ‘save\_ex’ object is in request.form. if ‘save\_ex’ is in request.form, insert inputs to exchange1 table. If the user clicks on the delete button, delete a row from exchange1 that is the same as the inputs. You will see login.html.

Text

Description automatically generated with medium confidence

Logout() – If you click on logout, you will see login.html. the loggedin and email are removed from the object.

Register() – If you don’t fill in all 3 inputs such as name, password, and email, you will see “Please fill out the form!” mesage. If you didn’t use @ in email section, you will see “invalid email address” mesage. If you use the email which you used before, you will see “account already exists !” mesage.

After filling in all 3 inputs, insert these data into the user1 table. And you will see “you have successfully registered !”.

Text

Description automatically generated

Sugarbayar Enkhbayar

456296

s.enkhbayar@student.uw.edu.pl