**forum**

**Objectives**

This project consists in creating a web forum that allows :

* communication between users.
* associating categories to posts.
* liking and disliking posts and comments.
* filtering posts.

**SQLite**

In order to store the data in your forum (like users, posts, comments, etc.) you will use the database library SQLite.

SQLite is a popular choice as an embedded database software for local/client storage in application software such as web browsers. It enables you to create a database as well as controlling it by using queries.

To structure your database and to achieve better performance we highly advise you to take a look at the [entity relationship diagram](https://www.smartdraw.com/entity-relationship-diagram/) and build one based on your own database.

* You must use at least one SELECT, one CREATE and one INSERT queries.

To know more about SQLite you can check the [SQLite page](https://www.sqlite.org/index.html).

**Authentication**

In this segment the client must be able to register as a new user on the forum, by inputting their credentials. You also have to create a login session to access the forum and be able to add posts and comments.

You should use cookies to allow each user to have only one opened session. Each of this sessions must contain an expiration date. It is up to you to decide how long the cookie stays "alive". The use of UUID is a Bonus task.

Instructions for user registration:

* Must ask for email
  + When the email is already taken return an error response.
* Must ask for username
* Must ask for password
  + The password must be encrypted when stored (this is a Bonus task)

The forum must be able to check if the email provided is present in the database and if all credentials are correct. It will check if the password is the same with the one provided and, if the password is not the same, it will return an error response.

**Communication**

In order for users to communicate between each other, they will have to be able to create posts and comments.

* Only registered users will be able to create posts and comments.
* When registered users are creating a post they can associate one or more categories to it.
  + The implementation and choice of the categories is up to you.
* The posts and comments should be visible to all users (registered or not).
* Non-registered users will only be able to see posts and comments.

**Likes and Dislikes**

Only registered users will be able to like or dislike posts and comments.

The number of likes and dislikes should be visible by all users (registered or not).

**Filter**

You need to implement a filter mechanism, that will allow users to filter the displayed posts by :

* categories
* created posts
* liked posts

You can look at filtering by categories as subforums. A subforum is a section of an online forum dedicated to a specific topic.

Note that the last two are only available for registered users and must refer to the logged in user.

**Docker**

For the forum project you must use Docker. You can read about docker basics in the [ascii-art-web-dockerize](https://github.com/01-edu/public/blob/master/subjects/ascii-art-web/dockerize/README.md) subject.

**Instructions**

* You must use **SQLite**.
* You must handle website errors, HTTP status.
* You must handle all sort of technical errors.
* The code must respect the [**good practices**](https://github.com/01-edu/public/blob/master/subjects/good-practices/README.md).
* It is recommended to have **test files** for [unit testing](https://go.dev/doc/tutorial/add-a-test).

**Allowed packages**

* All [standard Go](https://golang.org/pkg/) packages are allowed.
* [sqlite3](https://github.com/mattn/go-sqlite3)
* [bcrypt](https://pkg.go.dev/golang.org/x/crypto/bcrypt)
* [gofrs/uuid](https://github.com/gofrs/uuid) or [google/uuid](https://github.com/google/uuid)

You must not use any frontend libraries or frameworks like React, Angular, Vue etc.

This project will help you learn about:

* The basics of web :
  + HTML
  + HTTP
  + [Sessions](https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#session-management-waf-protections) and [cookies](https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies)
* Using and [setting up Docker](https://docs.docker.com/get-started/)
  + Containerizing an application
  + Compatibility/Dependency
  + Creating images
* SQL language
  + Manipulation of databases
* The basics of encryption