Chapter II

Essential Points

This project is a complex undertaking, requiring decision-making within the specified constraints. You have some flexibility in implementing certain modules, and it is left to your discretion **within the scope of the subject.** All your choices must be justifiable.

If you believe it's necessary to use *nginx* to set up your website, there's no issue, but ask yourself first, is it truly necessary? Can I do without it? Similarly, when faced with a library that could assist you, it's crucial to understand whether it will fulfill your tasks. You're not expected to rework uninteresting sub-layers but rather to make the proposed features function.

It's crucial to understand that you'll encounter decisions where doubts about implementing certain features will arise. Initially, it is **STRONGLY recommended** to comprehend the project requirements thoroughly. Once you've grasped what needs to be accomplished, it is necessary to stay within the framework of the project. When we mention an imposed technology, it explicitly means that everything officially related to the requested framework/language is allowed.

However, we emphasize that when you wish to implement a module, all restrictions apply to that module. For instance, if you want to realize the project with the Backend module as specified in the subject, you can no longer use the default language and must adapt your project accordingly. If you still want to create a backend using the default language, it's also possible, but since you're not using the requested language/framework, this module will not be considered valid.

Before concluding, it's important to note that some modules intentionally have strong dependencies on others.

Your choices are significant and must be justified during your evaluation. Exercise caution.

Take the time to contemplate the design of your application with your choices before delving into the code – it's crucial.

Have a fun ! :)

Chapter III

Mandatory part

This project is about creating a website for the mighty **Pong** contest!

The use of libraries or frameworks or tools to replace your job is strictly prohibited. Each part of the subject will explicitly present the authorized third party software you can use. Nevertheless, it is allowed and even recommended to use anything possible to simplify certain actions. It is important to note that any tools or resources utilized must be justified. Please be aware that simplifying does not equate to completing your work.

III.1 Overview

Thanks to your website, users will play Pong with others. You have to provide a nice user interface and real-time multiplayer online games!

- Your project needs to adhere to the following guidelines as a minimum requirement, contributing only a small portion to the final grade.
- The second part of this subject will offer additional modules that can replace or complete the following rules.

In this Subject, certain words are highlighted in green. These represent technology choices that will evolve over time. Pay close attention to the version of the subject.

III.2 Minimal technical requirement

Your project has to comply with the following rules:

Again, some of these contraints could be overriden by the choice of specific modules.

- You are free to develop the site, with or without a backend.
  - If you choose to include a backend, it must be written in pure `Ruby`. However, this requirement can be overridden by the **Framework module**.
  - If your backend or framework uses a database, you must follow the constraints of the **Database module**.
- The frontend should be developed using pure vanilla `Javascript`. However, this requirement can be altered through the **FrontEnd module**.
- Your website must be a single-page application. The user should be able to use the `Back` and `Forward` buttons of the browser.
- Your website must be compatible with the **latest stable up-to-date version** of *Google Chrome*.
- The user should encounter no unhandled errors and no warnings when browsing the website.
- Everything must be launched with a single command line to run an autonomous container provided by `Docker`. Example : `docker-compose up --build`

If your container solution is Docker:
When your computers in clusters run under Linux, you will use Docker in rootless mode for security reasons. This comes with 2 sideways:

- Your Docker runtime files must be located in /goinfre or /sgoinfre.
- You can't use so called "bind-mount volumes" between the host and the container if non-root UIDs are used in the container.

Depending on the project, your situation and the context, several fallbacks exist: Docker in a VM, rebuild you container after your changes, craft your own docker image with root as unique UID.

III.3 Game

The main purpose of this website is to play Pong versus other players.

- Therefore, users must have the ability to participate in a live Pong game against another player directly on the website. Both players will use the same keyboard. The **Remote players module** can enhance this functionality with remote players.
- A player must be able to play against another player, but it should also be possible to propose a **tournament**. This tournament will consist of multiple players who can take turns playing against each other. You have flexibility in how you implement the tournament, but it must clearly display who is playing against whom and the order of the players.
- A **registration system** is required: at the start of a tournament, each player must input their alias name. The aliases will be reset when a new tournament begins. However, this requirement can be modified using the **Standard User Management module**.
- There must be a **matchmaking system**: the tournament system organize the matchmaking of the participants, and announce the next fight.
- All players must adhere to the same rules, which includes having identical paddle speed. This requirement also applies when using AI; the AI must exhibit the same speed as a regular player.
- The game itself must be developed in accordance with the default frontend constraints (as outlined above), or you may choose to utilize the **FrontEnd module**, or you have the option to override it with the **Graphics module**. While the visual aesthetics can vary, it must still capture the essence of the **original Pong** (1972).

The use of libraries or frameworks or tools to replace your job is strictly prohibited. Each part of the subject will explicitly present the authorized third party software you can use. Nevertheless, it is allowed and even recommended to use anything possible to simplify certain actions. It is important to note that any tools or resources utilized must be justified. Please be aware that simplifying does not equate to completing your work.

III.4 Security concerns

In order to create a basic functional website, here are a few security concerns that you have to tackle:

- Any password stored in your database, if applicable, must be **hashed**.
- Your website must be protected against `SQL injections/XSS`.
- If you have a backend or any other features, it is mandatory to enable an HTTPS connection for all aspects (Utilize wss instead of ws...).
- You must implement some form of validation for forms and any user input, either within the base page if no backend is used or on the server side if a backend is employed.

`Please make sure you use a strong password hashing algorithm`

`For obvious security reasons, any credentials, API keys, env variables etc... must be saved locally in a .env file and ignored by git. Publicly stored credentials will lead you directly to a failure of the project.`

Chapter IV

Modules

Now that you've accomplished 25% of the project, congratulations!

With a functional basic website in place, the next step is to choose modules for further improvement.

To attain 100% project completion, a minimum of **7 major modules is required**. It's crucial to carefully review each module as it may necessitate modifications to your baseline website. Therefore, we strongly recommend reading this entire subject thoroughly.

The use of libraries or frameworks or tools to replace your job is strictly prohibited. Each part of the subject will explicitly present the authorized third party software you can use. Nevertheless, it is allowed and even recommended to use anything possible to simplify certain actions. It is important to note that any tools or resources utilized must be justified. Please be aware that simplifying does not equate to completing your work.

Two Minor Modules are equivalent to one Major Module.

IV.2 Web

These modules enable the integration of advanced web features into your Pong game.

- **Major module**: Use a Framework as backend.

  In this major module, you are required to utilize a specific web framework for your backend development, and that framework is Django.

You can create a backend without using the constraints of this module by using the default language/framework. However, this module will only be valid if you use the associated constraints.

- **Minor module**: Use a front-end framework or toolkit.

  Your frontend development will utilize the Bootstrap toolkit.

You can create a front-end without using the constraints of this module by using the default language/framework. However, this module will only be valid if you use the associated constraints.

- **Minor module**: Use a database for the backend -and more.

  The designated database for all DB instances in your project is PostgreSQL. This choice guarantees data consistency and compatibility across all project components and may be a prerequisite for other modules, such as the **backend Framework module.**

IV.3 User Management

This module delves into the realm of **User Management**, addressing crucial aspects of user interactions and access control within the Pong platform. It encompasses two major components, each focused on essential elements of user management and authentication: user participation across multiple tournaments and the implementation of remote authentication.

- **Major module**: Standard user management, authentication, users across tournaments.
  - Users can subscribe to the website in a secure way.
  - Registered users can log in in a secure way.
  - Users can select a unique display name to play the tournaments.
  - Users can update their information.
  - Users can upload an avatar, with a default option if none is provided.
  - Users can add others as friends and view their online status.
  - User profiles display stats, such as wins and losses.
  - Each user has a **Match History** including 1v1 games, dates, and relevant details, accessible to logged-in users.

`Be carefull, the management of duplicate usernames/emails is at your discretion. You must provide a justification for your decision.`

- **Major module**: Implementing a remote authentication.

  In this major module, the goal is to implement the following authentication system: OAuth 2.0 authentication with 42 . Key features and objectives include:

Be carefull, the management of duplicate usernames/emails is at your discretion. You must provide a justification for your decision.

- Integrate the authentication system, allowing users to securely sign in.
- Obtain the necessary credentials and permissions from the authority to enable a secure login.
- Implement user-friendly login and authorization flows that adhere to best practices and security standards.
- Ensure the secure exchange of authentication tokens and user information between the web application and the authentication provider.

This major module aims to get a remote user authentication, providing users with a secure and convenient way to access the web application.

IV.4 Gameplay and user experience

These modules are designed to enhance the general gameplay of the project.

- **Major module**: Remote players

  It is possible to have two distant players. Each player is located on a separated computer, accessing the same website and playing the same Pong game.

Think about network issues, like unexpected disconnection or lag.
You have to offer the best user experience possible.

IV.6 Cybersecurity

These cybersecurity modules are designed to bolster the security posture of the project, with the major module focusing on robust protection through Web Application Firewall (WAF) and ModSecurity configurations and HashiCorp Vault for secure secrets management. The minor modules complement this effort by adding options for GDPR compliance, user data anonymization, a ccount deletion, two-factor authentication (2FA), and JSON Web Tokens (JWT), collectively ensuring the project's commitment to data protection, privacy, and authentication security.

- **Major module**: Implement Two-Factor Authentication (2FA) and JWT.

  In this major module, the goal is to enhance security and user authentication by introducing Two-Factor Authentication (2FA) and utilizing JSON Web Tokens (JWT). Key features and objectives include:

  - Implement Two-Factor Authentication (2FA) as an additional layer of security for user accounts, requiring users to provide a secondary verification method, such as a one-time code, in addition to their password.
  - Utilize JSON Web Tokens (JWT) as a secure method for authentication and authorization, ensuring that user sessions and access to resources are managed securely.
  - Provide a user-friendly setup process for enabling 2FA, with options for SMS codes, authenticator apps, or email-based verification.
  - Ensure that JWT tokens are issued and validated securely to prevent unauthorized access to user accounts and sensitive data.

  This major module aims to strengthen user account security by offering Two-Factor Authentication (2FA) and enhancing authentication and authorization through the use of JSON Web Tokens (JWT).

IV.11 Server-Side Pong

- **Major module**: Replacing Basic Pong with Server-Side Pong and Implementing an API.

  In this major module, the goal is to replace the basic Pong game with a server-side Pong game, accompanied by the implementation of an API. Key features and objectives include:

  - Develop server-side logic for the Pong game to handle gameplay, ball movement, scoring, and player interactions.
  - Create an API that exposes the necessary resources and endpoints to interact with the Pong game, allowing partial usage of the game via the Command-Line Interface (CLI) and web interface.
  - Design and implement the API endpoints to support game initialization, player controls, and game state updates.
  - Ensure that the server-side Pong game is responsive, providing an engaging and enjoyable gaming experience.
  - Integrate the server-side Pong game with the web application, allowing users to play the game directly on the website.

  This major module aims to elevate the Pong game by migrating it to the server side, enabling interaction through both a web interface and CLI while offering an API for easy access to game resources and features.

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your files to ensure they are correct.

The use of libraries or frameworks or tools to replace your job is strictly prohibited. Each part of the subject will explicitly present the authorized third party software you can use. Nevertheless, it is allowed and even recommended to use anything possible to simplify certain actions. It is important to note that any tools or resources utilized must be justified. Please be aware that simplifying does not equate to completing your work.