Chapter II

Objectives

This project is a surprise.

As you are reaching the end of your Common Core, you have developed strong adaptation and problem-solving skills. This project will confront you with -maybe- unknown technologies, intentionally. Once again, you will have to adapt, discover, explore, experiment to create the expected software.

The project includes a mandatory part and a series of modules on various topics, as detailed below in this subject. You will be able to choose the modules you want among a large list, but each module and mandatory element contains technical constraints you cannot bypass. So you can select the topics you like, but not technologies you like. This is a deliberate pedagogical choice.

This project is not intended to be a portfolio for an incoming internship or other professional experience. Its purpose is to reveal your ability to become acquainted with and complete a complex task using an unfamiliar technology. This situation will inevitably be faced during your career, and we aim to develop your self-confidence in front of such situations.

Especially on this big and long project, we encourage you to carefully read the entire subject, consider several possible strategies, think about your design, before starting coding anything! Some modules may depend on others, some modules may conflict with others. Ft_transcendence will bring many doubts and requires a lot of difficult decisions! Act wisely :-)

Also, this project is definitely a long run, and a wrong path will lead you to a huge loss of time. Your project management and team management choices will strongly impact your timeline and results. Many approaches and tools exist to support you on these topics.

Good luck, and have fun playing Pong!

Chapter IV

Mandatory part

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

IV.1 Overview

Your software will offer a nice user interface and real-time multiplayer capabilities allowing to play Pong with all your friends!

- At first, your project needs to adhere to the mandatory guidelines as a minimum requirement (see next section), which will represent only a small portion to the final grade.
- The second part of this subject will offer additional modules that can replace or complement the mandatory part.

In this subject, you will face words that are highlighted in green. They represent technology choices that will evolve over time. Be sure to pay close attention to the version of the subject.

- The use of libraries or tools that provide an immediate and complete solution for an entire feature or a module is prohibited.
- Any direct instruction regarding the use (can, must, can't) of a third-party library or tool must be followed.
- The use of a small library or tool that solves a simple, unique task representing a subcomponent of a larger feature or module, is allowed.
- During the evaluation, the team will justify any use of library or tool that is not explicitly approved by the project guidelines and is not in contradiction with the project's constraints.
- During the evaluation, the evaluator will take responsibility for determining whether the use of a specific library or tool is legitimate (and allowed) or if it essentially solves an entire feature or module (and is therefore prohibited).

IV.2 Minimal technical requirement

Your project must comply with the following rules:

`Some of these constraints may be overridden 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 `PHP` without frameworks. 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 `Typescript` as base code. However, this requirement can be modified 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 *Mozilla Firefox*. Of course, it can be compatible with other web browsers!
- The user should encounter no unhandled errors or warnings when browsing the website.
- You must use `Docker` to run your website. Everything must be launched with a single command line to run an autonomous container.

`Several container technologies exist: Docker, containerd, podman, etc. On the computers of your campus, you may access the container software in rootless mode for security reasons. This could lead to the following extra constraints:`

- `Your runtime needs to be located in /goinfre or /sgoinfre.`
- `You are not able to use "bind-mount volumes" between the host and the container if non-root UIDs are used in the container.`

`Depending on the current requirements of the subject (highlighted in green above) and the local configuration in clusters, you may need to adopt different strategies, such as: container solution in virtual machine, rebuild your container after your changes, craft your own image with root as unique UID.`

IV.3 Game

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

- Users must be able 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, and a **tournament** system should also be available. 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 play.
- A **registration system** is required: at the start of a tournament, each player must input their alias. The aliases will be reset when a new tournament begins. However, this requirement can be modified using the **Standard User Management module**. **Note:** This does *not* imply user account creation.
  In the default (mandatory) version, users simply enter an alias manually.
- There must be a **matchmaking system**: the tournament system should organize the matchmaking of the participants, and announce the next match.
- All players must adhere to the same rules, including having identical paddle speed. This requirement also applies when using AI; the AI must exhibit the same speed as a regular player.
- The game must adhere to the default frontend constraints (as outlined above), or you may choose to use the **FrontEnd module**, or override it with the **Graphics module**. While the visual aesthetics can vary, the game must still capture the essence of the **original Pong** (1972).

- The use of libraries or tools that provide an immediate and complete solution for an entire feature or module is prohibited.
- Any direct instruction regarding the use (can, must, can't) of a third-party library or tool must be followed.
- The use of a small library or tool that solves a simple, unique task, representing a subcomponent of a larger feature or module, is allowed.
- During the evaluation, the team will justify any use of a library or tool that is not explicitly approved by the project guidelines and is not in contradiction with the project's constraints.
- During the evaluation, the evaluator will determine whether the use of a specific library or tool is legitimate (and allowed) or if it essentially solves an entire feature or module (and is therefore prohibited).
- The tournament system must work with or without user registration.
  - Without the **Standard User Management** module: users manually input an alias.
  - With the module: aliases are linked to registered accounts, allowing persistent stats and friend lists.

  The module extends the tournament logic; it does not replace it.

IV.4 Security concerns

In order to create a functional website, there are several security concerns you must address:

- Any password stored in your database, if applicable, must be **hashed**.
- Your website must be protected against `SQL injections`/`XSS` attacks.
- If you have a backend or any other features, it is mandatory to enable an HTTPS connection for all aspects (use wss instead of ws for example).
- You must implement validation mechanisms for forms and any user input, either on the base page if no backend is used, or on the server side if a backend is employed.
- Regardless of whether you choose to implement the JWT Security module with 2FA, it's essential to prioritize the security of your website. For instance, if you choose to create an API, ensure your routes are protected. Even if you decide not to use JWT tokens, securing the site remains critical.

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 cause your project to fail.

Chapter V

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

- The use of libraries or tools that provide an immediate and complete solution for an entire feature or module is prohibited.
- Any direct instructions regarding the use (can, must, can't) of a third-party library or tool must be followed.
- The use of a small library or tool that solves a simple, unique task, representing a subcomponent of a larger feature or module, is allowed.
- During the evaluation, the team will justify any usage of library or tool that is not explicitly approved by the subject, and that does not contradict the subject's constraints.
- During the evaluation, the evaluator will determine whether the use of a specific library or tool is legitimate (and allowed) or if it essentially solves an entire feature or module (and is therefore prohibited).

Two Minor Modules count as one Major Module.

V.1 Overview

- **Web**
  - **Major module**: Use a framework to build the backend.
  - **Minor module**: Use a framework or a toolkit to build the frontend.
  - **Minor module**: Use a database for the backend.
  - **Major module**: Store the score of a tournament in the Blockchain.
- **User Management**
  - **Major module**: Standard user management, authentication, users across tournaments.
  - **Major module**: Implementing a remote authentication.

  Gameplay and user experience
  - **Major module**: Remote players
  - **Major module**: Multiplayer (more than 2 players in the same game).
  - **Major module**: Add another game with user history and matchmaking.
  - **Minor module**: Game customization options.
  - **Major module**: Live chat.
- **AI-Algo**
  - **Major module**: Introduce an AI opponent.
  - **Minor module**: User and game stats dashboards
- **Cybersecurity**
  - **Major module**: Implement WAF/ModSecurity with a hardened configuration and HashiCorp Vault for secrets management.
  - **Minor module**: GDPR compliance options with user anonymization, local data management, and Account Deletion.
  - **Major module**: Implement Two-Factor Authentication (2FA) and JWT.
- **Devops**
  - **Major module**: Infrastructure setup for log management.
  - **Minor module**: Monitoring system.
  - **Major module**: Designing the backend as microservices.

- **Graphics**
  - **Major module**: Use advanced 3D techniques.
- **Accessibility**
  - **Minor module**: Support on all devices.
  - **Minor module**: Expanding browser compatibility.
  - **Minor module**: Supports multiple languages.
  - **Minor module**: Add accessibility features for visually impaired users.
  - **Minor module**: Server-Side Rendering (SSR) integration.
- **Server-Side Pong**
  - **Major module**: Replace basic Pong with server-side Pong and implement an API.
  - **Major module**: Enabling Pong gameplay via CLI against web users with API integration.

V.2 Web

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

- **Major module**: Use a framework to build the backend.

  In this major module, you are required to use a specific web framework for backend development: Fastify with Node.js.

  

  You can create the backend without using the constraints of this module by using the default backend language (as specified above in the mandatory part). However, this module will only be valid if you follow its requirements.

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

  Your frontend development must use the Tailwind CSS in addition of the Typescript, and nothing else.

  

  You can create a front-end without using the constraints of this module by using the default front-end directives (as specified above in the mandatory part). However, this module will only be valid if you follow its requirements.

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

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

- **Major module**: Store the score of a tournament in the Blockchain.

  This Major module focuses on implementing a feature within the Pong website to securely store tournament scores on a blockchain. It is essential to clarify that for development and testing purposes, we will use a testing blockchain environment. The chosen blockchain for this implementation is Avalanche, and Solidity will be the programming language used for smart contract development.

  - Blockchain Integration: The primary goal of this module is to seamlessly integrate blockchain technology, specifically Avalache, into the Pong website. This integration ensures the secure and immutable storage of tournament scores, providing players with a transparent and tamper-proof record of their gaming achievements.
  - Solidity Smart Contracts: To interact with the blockchain, we will develop Solidity smart contracts. These contracts will be responsible for recording, managing, and retrieving tournament scores.
  - Testing Blockchain: As mentioned earlier, a testing blockchain will be used for development and testing purposes. This ensures that all blockchain-related

functionalities are thoroughly validated without any risks associated with a live blockchain.

- Interoperability: This module may have dependencies on other modules, particularly the Backend Framework module. Integrating blockchain functionality might require adjustments in the backend to accommodate interactions with the blockchain.

By implementing this module, we aim to enhance the Pong website by introducing a blockchain-based score storage system. Users will benefit from the added layer of security and transparency, ensuring the integrity of their gaming scores. The module emphasizes the use of a testing blockchain environment to minimize risks associated with blockchain development.

V.3 User Management

This module delves into the realm of **User Management**, addressing key 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 and users across tournaments.
  - Users can securely subscribe to the website.
  - Registered users can securely log in.
  - Users can select a unique display name to participate in 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.

`The management of duplicate usernames/emails is at your discretion; please ensure a logical solution is provided.`

- **Major module**: Implement remote authentication.

  In this major module, the goal is to implement a secure external authentication system using OAuth 2.0.

- **You are free to choose any OAuth-compatible provider (e.g., Google, GitHub, etc.)**.

Key features and objectives include:

- Integrate the authentication system, allowing users to securely sign in.
- Obtain the necessary credentials and permissions from the authority to enable 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 provide a remote user authentication, offering users a secure and convenient way to access the web application.

V.4 Gameplay and user experience

These modules are designed to enhance the overall gameplay experience of the project.

- **Major module**: Remote players

  It should be possible for two players to play remotely. Each player is located on a separated computer, accessing the same website and playing the same Pong game.

  

  Consider network issues, such as unexpected disconnections or lag.
  You must offer the best user experience possible.

- **Major module**: Multiple players

  It should be possible to have more than two players. Each player needs live control (so the "remote players" module is strongly recommended). It's up to you to decide how the game could be played with 3, 4, 5, 6 or more players. Along with the regular 2 players game, you can define a specific number of players, greater than 2, for this multiplayer module. Ex: 4 players could play on a square board, with each player controlling one unique side of the square.

- **Major module**: Add another game with user history and matchmaking.

  The goal of this major module, is to introduce a new game, distinct from Pong, and incorporate features such as user history tracking and matchmaking. Key features and objectives include:

  - Develop a new, engaging game to diversify the platform's offerings and entertain users.
  - Implement user history tracking to record and display individual users' gameplay statistics.
  - Create a matchmaking system to allow users to find opponents and participate in fair and balanced matches.
  - Ensure that user game history and matchmaking data are stored securely and remain up-to-date.
  - Optimize the performance and responsiveness of the new game to provide an enjoyable user experience. Regularly update and maintain the game to fix bugs, add new features, and enhance gameplay.

  This major module aims to expand your platform by introducing a new game, enhancing user engagement with gameplay history, and facilitating matchmaking for an enjoyable gaming experience.

- **Minor module**: Game customization options.

  In this minor module, the goal is to provide customization options for all available games on the platform. Key features and objectives include:

- Offer customization features, such as power-ups, attacks, or different maps, that enhance the gameplay experience.
- Allow users to choose a default version of the game with basic features if they prefer a simpler experience.
- Ensure that customization options are available and applicable to all games offered on the platform.
- Implement user-friendly settings menus or interfaces for adjusting game parameters.
- Maintain consistency in customization features across all games to provide a unified user experience.

This module aims to give users the flexibility to tailor their gaming experience across all available games by providing a variety of customization options while also offering a default version for those who prefer a straightforward gameplay experience.

- **Major module**: Live Chat.

  In this module, you need to create a chat feature for users:

  - The user should be able to send **direct messages** to other users.
  - The user should be able to block other users, preventing them from seeing any further messages from the blocked account.
  - The user should be able to invite other users to play a Pong game through the chat interface.
  - The tournament system should be able to notify users about the next game.
  - The user should be able to access other players' profiles through the chat interface.

V.5 AI-Algo

These modules serve to introduce data-driven elements to the project. The major module introduces an AI opponent for enhanced gameplay, while the minor module focuses on user and game statistics dashboards, offering users a minimalistic yet insightful glimpse into their gaming experiences.

- **Major module**: Introduce an AI opponent.

  In this major module, the objective is to incorporate an AI player into the game. Notably, the use of the **A* algorithm** is not permitted for this task. Key features and goals include:

  - Develop an AI opponent that provides a challenging and engaging gameplay experience for users.
  - The AI must replicate human behavior, which means that in your AI implementation, you must simulate keyboard input. The constraint here is that the AI can only refresh its view of the game once per second, requiring it to anticipate bounces and other actions.

The AI must utilize power-ups if you have chosen to implement the **Game customization options** module.

  - Implement AI logic and decision-making processes that enable the AI player to make intelligent and strategic moves.
  - Explore alternative algorithms and techniques to create an effective AI player without relying on A*.
  - Ensure that the AI adapts to different gameplay scenarios and user interactions.

You will need to explain in detail how your AI works during your evaluation. Creating an AI that does nothing is strictly prohibited; it must have the capability to win occasionally.

  This major module aims to enhance the game by introducing an AI opponent that adds excitement and competitiveness without relying on the A* algorithm.

- **Minor module**: User and Game Stats Dashboards.

  In this minor module, the goal is to introduce dashboards that display statistics for individual users and game sessions. Key features and objectives include:

  - Create user-friendly dashboards that provide users with insights into their gaming statistics.

- Develop a separate dashboard for game sessions, showing detailed statistics, outcomes, and historical data for each match.
- Ensure that the dashboards offer an intuitive and informative user interface for tracking and analyzing data.
- Implement data visualization techniques, such as charts and graphs, to present statistics in a clear and visually appealing manner.
- Allow users to access and explore their own gaming history and performance metrics conveniently.
- Feel free to add any metrics you deem useful.

This minor module aims to empower users with the ability to monitor their gaming statistics and game session details through user-friendly dashboards, providing a comprehensive view of their gaming experience.

V.6 Cybersecurity

These cybersecurity modules are designed to enhance the security posture of the project. The major module focuses on robust protection through Web Application Firewall (WAF) and ModSecurity configurations, as well as HashiCorp Vault for secure secrets management. The minor modules complement this effort by adding features for GDPR compliance, user data anonymization, account 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 WAF/ModSecurity with Hardened Configuration and HashiCorp Vault for Secrets Management.

  The objective of this major module is to enhance the security infrastructure of the project by implementing several key components. Key features and goals include:

  - Configure and deploy a Web Application Firewall (WAF) and ModSecurity with a strict and secure configuration to protect against web-based attacks.
  - Integrate HashiCorp Vault to securely manage and store sensitive information, such as API keys, credentials, and environment variables, ensuring that these secrets are properly encrypted and isolated.

  This major module aims to bolster the project's security infrastructure by implementing robust security measures, including WAF/ModSecurity for web application protection and HashiCorp Vault for secrets management to ensure a safe and secure environment.

- **Minor module**: GDPR compliance options with user anonymization, local data management, and account deletion.

  The goal of this minor module is to introduce GDPR compliance options that allow users to exercise their data privacy rights. Key features and objectives include:

  - Implement GDPR-compliant features that enable users to request anonymization of their personal data, ensuring that their identity and sensitive information are protected.
  - Provide tools for users to manage their local data, including the ability to view, edit, or delete their personal information stored within the system.
  - Offer a streamlined process for users to request the permanent deletion of their accounts, including all associated data, ensuring compliance with data protection regulations.
  - Maintain clear and transparent communication with users regarding their data privacy rights, with easily accessible options to exercise these rights.

  This minor module aims to enhance user privacy and data protection by offering GDPR compliance options that empower users to control their personal information and exercise their data privacy rights within the system.

  If you are not familiar with the General Data Protection Regulation (GDPR), it

is essential to understand its principles and implications, especially regarding user data management and privacy. The GDPR is a regulation that aims to protect the personal data and privacy of individuals within the European Union (EU) and the European Economic Area (EEA). It sets out strict rules and guidelines for organizations on how they should handle and process personal data.

To gain a better understanding of the GDPR and its requirements, it is strongly recommended to visit the official website of the European Commission on data protection[1]. This website provides comprehensive information about the GDPR, including its principles, objectives, and user rights. It also offers additional resources to delve deeper into the topic and ensure compliance with the regulation.

If you are unfamiliar with the GDPR, please take the time to visit the provided link and familiarize yourself with the regulations before proceeding with this project.

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

  The goal of this major module 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).

[1] https://commission.europa.eu/law/law-topic/data-protection/data-protection-eu_en

V.7 Devops

These modules focus on enhancing the project's infrastructure and architecture. The major modules address infrastructure setup for efficient log management using ELK (Elasticsearch, Logstash, Kibana), designing the backend as microservices for flexibility and scalability, and implementing Prometheus/Grafana for comprehensive system monitoring.

- **Major module**: Infrastructure Setup with ELK (Elasticsearch, Logstash, Kibana) for Log Management.

  The objective of this major module is to establish a robust infrastructure for log management and analysis using the ELK stack (Elasticsearch, Logstash, Kibana). Key features and goals include:

  - Deploy Elasticsearch to efficiently store and index log data, ensuring it is easily searchable and accessible.
  - Configure Logstash to collect, process, and transform log data from various sources, sending it to Elasticsearch.
  - Set up Kibana for visualizing log data, creating dashboards, and generating insights from log events.
  - Define data retention and archiving policies to manage log data storage effectively.
  - Implement security measures to protect log data and access to the ELK stack components.

  This major module aims to establish a powerful log management and analysis system using the ELK stack, enabling effective troubleshooting, monitoring, and insights into the system's operation and performance.

- **Minor module**: Monitoring system.

  The goal of this minor module is to set up a comprehensive monitoring system using Prometheus and Grafana. Key features and goals include:

  - Deploy Prometheus as the monitoring and alerting toolkit to collect metrics and monitor the health and performance of various system components.
  - Configure data exporters and integrations to capture metrics from different services, databases, and infrastructure components.
  - Create custom dashboards and visualizations using Grafana to provide real-time insights into system metrics and performance.
  - Set up alerting rules in Prometheus to proactively detect and respond to critical issues and anomalies.
  - Ensure proper data retention and storage strategies for historical metrics data.
  - Implement secure authentication and access control mechanisms for Grafana to protect sensitive monitoring data.

This minor module aims to establish a robust monitoring infrastructure using Prometheus and Grafana, enabling real-time visibility into system metrics and proactive issue detection for improved system performance and reliability.

- **Major module**: Designing the Backend as Microservices.

  The goal of this major module is to architect the backend of the system using a microservices approach. Key features and objectives include:

  - Divide the backend into smaller, loosely-coupled microservices, each responsible for specific functions or features.
  - Define clear boundaries and interfaces between microservices to enable independent development, deployment, and scaling.
  - Implement communication mechanisms between microservices, such as RESTful APIs or message queues, to facilitate data exchange and coordination.
  - Ensure that each microservice is responsible for a single, well-defined task or business capability, promoting maintainability and scalability.

  This major module aims to enhance the system's architecture by adopting a microservices design approach, enabling greater flexibility, scalability, and maintainability of the backend components.

V.8 Graphics

- **Major module**: Implementing Advanced 3D Techniques

  This major module,"Graphics," focuses on enhancing the visual aspects of the Pong game. It introduces the use of advanced 3D techniques to create a more immersive gaming experience. Specifically, the Pong game will be developed using Babylon.js to achieve the desired visual effects.

  - Advanced 3D Graphics: The primary goal of this module is to implement advanced 3D graphics techniques to elevate the visual quality of the Pong game. By utilizing Babylon.js, the goal is to create stunning visual effects that immerse players in the gaming environment.
  - Immersive Gameplay: The incorporation of advanced 3D techniques enhances the overall gameplay experience by providing users with a visually engaging and captivating Pong game.
  - Technology Integration: The chosen technology for this module is Babylon.js. These tools will be used to create the 3D graphics, ensuring compatibility and optimal performance.

  This major module aims to revolutionize the Pong game's visual elements by introducing advanced 3D techniques. Through the use of Babylon.js, we aim to provide players with an immersive and visually stunning gaming experience.

V.9 Accessibility

These modules are designed to enhance the accessibility of our web application, with a focus on ensuring compatibility across all devices, expanding browser support, offering multi-language capabilities, providing accessibility features for visually impaired users, and integrating Server-Side Rendering (SSR) for improved performance and user experience.

- **Minor module**: Support on all devices.

  In this module, the main focus is to ensure that your website works seamlessly on all types of devices. Key features and objectives include:

  - Ensure the website is responsive, adapting to different screen sizes and orientations, providing a consistent user experience on desktops, laptops, tablets, and smartphones.
  - Ensure that users can easily navigate and interact with the website using different input methods, such as touchscreens, keyboards, and mice, depending on the device they are using.

  This module aims to provide a consistent and user-friendly experience on all devices, maximizing accessibility and user satisfaction.

- **Minor module**: Expanding Browser Compatibility.

  In this minor module, the objective is to enhance the compatibility of the web application by adding support for an additional web browser. Key features and objectives include:

  - Extend browser support to include an additional web browser, ensuring that users can access and use the application seamlessly.
  - Conduct thorough testing and optimization to ensure that the web application functions correctly and displays correctly in the newly supported browser.
  - Address any compatibility issues or rendering discrepancies that may arise in the added web browser.
  - Ensure a consistent user experience across all supported browsers, maintaining usability and functionality.

  This minor module aims to broaden the accessibility of the web application by supporting an additional web browser, providing users with more choices for their browsing experience.

- **Minor module**: Multiple language support.

  In this minor module, the objective is to ensure that your website supports multiple languages to cater to a diverse user base. Key features and goals include:

  - Implement support for a minimum of three languages on the website to accommodate a broad audience.

- Provide a language switcher or selector that allows users to easily change the website's language based on their preferences.
- Translate essential website content, such as navigation menus, headings, and key information, into the supported languages.
- Ensure that users can navigate and interact with the website seamlessly, regardless of the selected language.
- Consider using language packs or localization libraries to simplify the translation process and maintain consistency across different languages.
- Allow users to set their preferred language as the default for subsequent visits.

This minor module aims to enhance the accessibility and inclusivity of your website by offering content in multiple languages, making it more user-friendly for a diverse international audience.

- **Minor module**: Add accessibility for Visually Impaired Users.

  In this minor module, the goal is to make your website more accessible for visually impaired users. Key features include:

  - Support for screen readers and assistive technologies.
  - Clear and descriptive alt text for images.
  - High-contrast color scheme for readability.
  - Keyboard navigation and focus management.
  - Options for adjusting text size.
  - Regular updates to meet accessibility standards.

  This module aims to improve the website's usability for individuals with visual impairments and ensure compliance with accessibility standards.

- **Minor module**: Server-Side Rendering (SSR) Integration.

  In this minor module, the focus is on integrating Server-Side Rendering (SSR) to enhance the performance and user experience of your website. Key objectives include:

  - Implement SSR to improve the website's loading speed and overall performance.
  - Ensure that content is pre-rendered on the server and delivered to users' browsers for faster initial page loads.
  - Optimize SEO by providing search engines with pre-rendered HTML content.
  - Maintain a consistent user experience while benefiting from the advantages of SSR.

  This module aims to boost website performance and SEO by integrating Server-Side Rendering for faster page loads and improved user experience.

V.10 Server-Side Pong

- **Major module**: Replace 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.

- **Major module**: Enabling Pong Gameplay via CLI against Web Users with API Integration.

  In this major module, the goal is to develop a Command-Line Interface (CLI) that allows users to play Pong against players using the web version of the game. The CLI should connect to the web application seamlessly, enabling CLI users to join and interact with web players. Key features and objectives include:

  - Create a robust CLI application that replicates the Pong gameplay experience available on the website, providing CLI users with the ability to initiate and participate in Pong matches.
  - Utilize the API to establish communication between the CLI and the web application, enabling CLI users to connect to the site and interact with web players.
  - Develop a user authentication mechanism within the CLI, allowing CLI users to log in to the web application securely.
  - Implement real-time synchronization between the CLI and web users, ensuring that gameplay interactions are seamless and consistent.
  - Enable CLI users to join and create Pong matches with web players, facilitating cross-platform gameplay.

- Provide comprehensive documentation and guidance on how to use the CLI effectively for Pong matches against web users.

This major module aims to enhance the Pong gaming experience by creating a CLI that seamlessly connects CLI users to web players through API integration, offering a unified and interactive gameplay environment.

If you wish to complete this module, we strongly recommend that you do the previous one.

Chapter VI

Bonus part

For this project, the bonus section is designed to be straightforward. You are required to include additional modules.

- Five points will be awarded for each **minor module**.
- Ten points will be awarded for each **major module**.

The bonus part will only be assessed if the mandatory part is PERFECT. "Perfect" means the mandatory part has been completed fully and works without any issues. If you do not meet ALL the mandatory requirements, your bonus part will not be evaluated at all.