FFA Platform Technical and Functional Specifications

1. Introduction
2. Background
3. Platform Objectives
4. Functional Requirements
   1. Frontend User Features
   2. Intervener Features
   3. Administrator Features
5. Core Features
   1. User Authentication and Role Management
   2. Responsive Design
   3. Security
   4. User Privacy and Data Protection
6. Non-Functional Requirements
   1. Performance Requirements
   2. Availability
   3. Maintainability
   4. Scalability
7. Technology Stack
   1. Frontend
   2. Backend
   3. Database
   4. Security and Encryption
   5. Other Tools
8. Security and Risk Management
   1. Security Vulnerability Prevention
   2. Data Backup and Recovery
   3. Privacy Protection and Compliance
9. **Introduction**
10. This document describes the technical and functional requirements of the FFA platform. The goal of this platform is to provide users with a complete system for project publication, application, and management. The platform involves multiple user roles: general users, interveners, and administrators, each with different permissions and functions.
11. **Background**

The FFA platform aims to improve project management processes through digital means. The platform will support project publication, application submission, project feedback management, and effectively connect various users, providing a secure and user-friendly project collaboration environment.

1. **Platform Objectives**

The objective is to develop a complete and responsive web application, covering the frontend, back-office, and backend, to achieve the following goals:

• Project publication and application functionality

• Interaction between interveners, general users and projects

• Administrator account and permissions management

• Secure and reliable user authentication and data protection mechanisms

1. **Functional Requirements**
   1. Frontend User Features

Frontend users are registered users who can browse and apply for projects. The main features include:

* + - User Registration, Login, and Logout:
* Users can register via email, phone number.
* Users can log in with the registered email or phone number and verify their identity via password.
* Users can log out of the current session at any time.
  + - Project Browsing:

Users can view all published projects on the platform without logging in. Each project will display brief information such as project name, description, and publication date.

* + - Project Application:

Registered users can apply for projects by uploading documents and filling out forms and can check the status of their applications at any time.

* + - Notifications and Message Management:
* Users will receive feedback notifications regarding their project applications.
* Users have a personal message inbox displaying application statuses and other notifications.
  + - Project Filtering:

Users can search and filter projects based on multiple criteria (e.g., country, language, project type, application deadline).

* 1. Intervener Features

Interveners are responsible for project management and have more permissions than general users. Their features include:

* + - Project Management:
* Interveners can securely submit projects via the platform and manage them, including editing and deleting projects.
* Interveners can view all application materials submitted by users and provide feedback on applications.
  + - Notifications and Message Management:
* Interveners can receive feedback from project applicants and respond to them.
* Interveners have a personal message inbox displaying all project-related notifications and messages from applicants.
  + - Department and Cooperation Management:

Interveners can specify their affiliated decentralized cooperation department and manage related projects.

* + - Statistics and Analytics:

Interveners can view statistics for each project, including view count, number of applications, and project completion rates.

* 1. Administrator Features

Administrators are responsible for managing the overall operation of the platform. Their features include:

* + - User and Permissions Management:
* Administrators can verify the authenticity of intervener accounts and enable or disable user accounts.
* Administrators can create, read, update, and delete information on countries, departments, embassies, and other entities.
  + - Intervener Management:

Administrators can manage intervener accounts, including deleting, modifying, and viewing their information.

* + - System Maintenance and Support:

Administrators can send system notifications, provide technical support, and assist platform users.

1. **Core Features**
   1. User Authentication and Role Management

* The platform uses role-based access control (RBAC) to ensure that different roles (general users, interveners, administrators) have distinct access rights.
* User authentication is performed using email/phone number and password.
  1. Responsive Design
* The platform adopts a responsive web design to ensure a consistent user experience across devices (desktop, tablet, mobile).
* The frontend is built with the Vue.js framework and uses UI components to ensure a modern and user-friendly interface.
  1. Security
* End-to-end encryption (SSL/TLS) is used to ensure secure communication between users.
* Data storage and transmission are encrypted using the AES-256 encryption standard.
* The platform supports secure session management and automatic logout to prevent unauthorized access.
  1. User Privacy and Data Protection
* The platform adheres to the principle of data minimization, collecting and processing only the necessary data for project operations.
* User data is anonymized in the database, and users have the right to access, modify, and delete their personal data.

1. **Non-Functional Requirements**

Non-functional requirements define the quality attributes of the FFA platform, ensuring the system meets essential criteria for performance, reliability, security, scalability, and maintainability.

* 1. Performance Requirements

The FFA platform is expected to perform efficiently, even under high user load. Performance goals include:

* Handling Concurrent Users: The platform must be able to support multiple users performing operations simultaneously without degradation in service. This includes managing both logged-in and anonymous users accessing the platform.
* Response Time: The platform should maintain fast response times, ensuring that user actions, such as submitting applications or viewing projects, are completed within a reasonable timeframe (e.g., under 2 seconds for major actions).
* Optimized API Performance: APIs should process requests and deliver responses quickly, using caching mechanisms where appropriate to reduce latency.
  1. Availability

The FFA platform must be highly available to ensure continuous operation and minimal downtime. Specific requirements include:

* Uptime Target: The platform must have a 99.9% uptime, ensuring it is accessible for most of the time with minimal interruptions.
* Redundant Infrastructure: Redundancy should be built into the infrastructure to prevent single points of failure. This includes mirrored databases and backup servers to take over in case of a failure.
* Disaster Recovery: A well-defined disaster recovery plan should be in place to restore operations within a predefined time frame (e.g., recovery point objective [RPO] and recovery time objective [RTO] of less than 1 hour).
  1. Maintainability

The platform should be designed for long-term maintenance and easy future upgrades. Key aspects include:

* Modular Code Structure: The code should be structured in a modular way, allowing specific components to be updated, modified, or replaced without impacting the entire system.
* Documentation: All components, from the frontend to the backend and database interactions, should be thoroughly documented. This ensures that future developers can easily understand and work on the platform.
* Automated Testing and CI/CD: Continuous integration and deployment pipelines should be established to automate testing and ensure that new updates do not introduce regressions or bugs.
  1. Scalability
* The platform must be scalable to accommodate increasing user demands and future growth. Requirements include:
* Horizontal and Vertical Scaling: The system should be capable of scaling both horizontally (adding more servers to handle load) and vertically (upgrading the capacity of existing servers) as demand increases.
* Database Scalability: The PostgreSQL database should be optimized for scalability through partitioning, replication, and load balancing to handle larger volumes of data and queries.
* Cloud Infrastructure Support: If necessary, the platform should be deployable to cloud infrastructure like AWS or Azure, enabling elastic scaling to meet fluctuating demand.

1. **Technology Stack**
   1. **Code versionning**

GitHub will be used to share and manage the different verrsions of the source code.

Back office : <https://github.com/Maurras/PFP_2425_FFAPlatform_BO.git>

Back-End : <https://github.com/Maurras/PFP_2425_FFAPlatform_BE.git>

Front-End : <https://github.com/Maurras/PFP_2425_FFAPlatform_FE.git>

* 1. Frontend

The frontend of the FFA platform is built using modern, flexible technologies to deliver an interactive and responsive user interface. The key technologies include:

* Vue.js: A progressive JavaScript framework for building reactive user interfaces. Vue.js allows for dynamic content updates without full page reloads, improving the overall user experience.
* Primefaces: This library provides a rich set of UI components like forms, calendars, and tables that enhance the platform’s functionality while maintaining a consistent, polished look.
* HTML5, CSS3, and JavaScript (ES6+): These foundational web technologies are used to structure the content, style the platform, and add client-side interactivity. CSS frameworks like Bootstrap can be used to ensure responsive layouts.
  1. Backend

The backend is the core engine of the platform, handling business logic, data processing, and communications with the database. The backend architecture is based on the following technologies:

* Spring Boot: A powerful Java-based framework used to build the backend services. It simplifies the development of RESTful APIs, facilitates dependency injection, and provides robust security features.
* Apache Shiro and Spring Security: These frameworks are used to secure the platform, managing authentication and authorization across different user roles (administrators, interveners, users).
* REST API: The backend communicates with the frontend and external systems through REST APIs, providing data and processing requests via HTTP.
  1. Database

The FFA platform uses PostgreSQL, a high-performance, open-source relational database management system (RDBMS) that offers:

* Data Integrity and ACID Compliance: PostgreSQL ensures that transactions are processed reliably, maintaining data integrity even in cases of system failure.
* Advanced Data Types: PostgreSQL supports advanced data types such as JSONB, allowing the platform to store and query semi-structured data efficiently.
* Indexing and Query Optimization: PostgreSQL’s indexing and query optimization features ensure that large datasets can be searched quickly, minimizing response times.
* Database Tools: Hibernate or MyBatis-Plus may be used as the ORM (Object Relational Mapping) layer, abstracting the database interactions and ensuring efficient query generation.
  1. Security and Encryption

Security and encryption are critical components of the FFA platform’s infrastructure. The platform must ensure that all user data is handled securely and that communications between the server and client are protected from unauthorized access and interception.

* Encryption Protocols: All data transferred between the client and the server is encrypted using SSL/TLS protocols to prevent data breaches and eavesdropping. The platform will support the latest versions of these encryption protocols to ensure maximum security.
* Data at Rest Encryption: Sensitive data stored within the database is encrypted using advanced encryption algorithms, ensuring that even in the event of unauthorized access to the database, the data remains protected.
* Password Management: User passwords are hashed using secure hashing algorithms such as bcrypt, which include salt and multiple iterations to prevent password cracking in the event of a data leak.
* Multi-Factor Authentication (MFA): The platform supports multi-factor authentication (MFA) for user accounts, especially for administrative roles, to add an additional layer of security during the login process.
* Regular Security Audits: The platform will undergo regular security audits and vulnerability testing to ensure that encryption protocols are functioning properly, and no new vulnerabilities are introduced during updates or changes.
  1. Other Tools

In addition to the core technologies described in previous sections, the FFA platform relies on various additional tools to ensure smooth development, deployment, and monitoring processes.

* Monitoring and Logging Tools: Tools like Prometheus and Grafana are used to monitor the platform's health and performance, including tracking user activity, load times, and server status. ELK Stack (Elasticsearch, Logstash, Kibana) is used for logging and analyzing system events to troubleshoot and debug issues efficiently.
* Version Control: The platform uses Git for version control, enabling collaborative development and ensuring that all changes to the platform are tracked and can be rolled back if necessary.
* CI/CD Pipelines: Continuous integration and deployment pipelines are set up using tools like Jenkins or GitLab CI/CD, automating the process of testing and deploying new code to the production environment. This ensures that updates can be delivered quickly and without introducing bugs.
* API Documentation Tools: Tools like Swagger are used to document the platform’s APIs, ensuring that both developers and external integrators have clear, up-to-date information on how to interact with the platform’s services.

1. **Security and Risk Management**
   1. Security Vulnerability Prevention

Security is integral to the FFA platform to protect sensitive data and maintain user trust. The security model is built around preventing unauthorized access, ensuring data confidentiality, and securing user communications. Key features include:

* End-to-End Encryption: All communications between the client and server are encrypted using SSL/TLS protocols to protect against eavesdropping or man-in-the-middle attacks.
* Multi-Factor Authentication (MFA): To protect user accounts, particularly those of interveners and administrators, MFA is enforced, requiring users to verify their identity through a secondary method, such as SMS codes or authentication apps.
* Role-Based Access Control (RBAC): Permissions are tightly controlled based on user roles (admin, user, intervener), ensuring that users can only access the data and features necessary for their roles.
  1. Data Backup and Recovery

The FFA platform implements a comprehensive data backup and recovery strategy to prevent data loss:

* Regular Backups: Automated backups are performed daily, and the backup system is designed to retain multiple versions of data, allowing restoration to a specific point in time.
* Offsite Storage: Backup data is stored both locally and offsite in secure, encrypted storage, ensuring it is safe from local disasters like hardware failure or natural calamities.
* Disaster Recovery Plan: In the event of a major system failure, a disaster recovery plan ensures minimal disruption to users. This plan includes defined RPO (Recovery Point Objective) and RTO (Recovery Time Objective) values to guarantee data recovery within hours
  1. Privacy Protection and Compliance

The FFA platform strictly adheres to privacy protection regulations and best practices, ensuring users’ data is handled with care and compliance with relevant legal frameworks like GDPR.

* Data Minimization: Only the necessary data required for the platform’s operations is collected, processed, and stored. Unnecessary personal information is anonymized or deleted to protect user privacy.
* User Data Rights: Users are provided with tools to control their personal data, including the ability to request access, make corrections, or permanently delete their information from the platform. All such requests are processed in accordance with applicable data protection laws.
* Regular Audits: Regular audits are conducted to ensure compliance with data privacy regulations and to identify any areas for improvement. These audits include reviewing data handling processes, access logs, and security measure