Project Functional Specification Document

Project name: Timeless Empire

Project team: **Quentin Redt-Zimmer** – Time manager

Yiru Xiong – Project manager

Rory Bueno - Technicians

Mathis Constant - Technicians

Luka Lesueur – Communications

Website: *https://timeless-empire.fr/*

Version: V3 – 10/01/2026

1. Context and Objectives

Brief Presentation of the Initial Need

This project is a computer science project that consists of the creation of a video game developed in Python.
The game is designed to provide an entertaining and strategic experience for multiple players.

Problem Statement to Be Addressed

The entire game must be fully coded in Python, using the Pygame library.
No external game engines or advanced frameworks will be used.
The challenge is to design a complete and functional game only with Python programming and basic game development tools.

Main Objectives of the Project

The main objective is to create a strategy game that allows users to:

- Develop their strategic thinking
- Manage their own empire
- Interact and manage relationships with other players

The game aims to entertain users and offer a meaningful way to spend their free time while training their logical and critical thinking skills.

---

2. Project Scope

Moving on to the Project Scope. To deliver a working prototype on time, we had to be very clear about what to build and what to leave out.

First, what did we include? We focused on the Core Gameplay Loop.

- We built a Resource and Development System to drive the economy.
- We created a Battle and Trading System to force player interaction.
- And technically, we implemented a Turn Management System to keep all 5 players synchronized on the server.

Second, what did we exclude? We made strategic cuts to ensure stability.

- No Manual Battles: We made combat automatic to focus on strategy rather than clicking speed.
- No Terrain Effects: Mountains don't slow you down. This kept our pathfinding code simple and bug-free.
- No Real History: We ignored strict historical events to give us more creative freedom with the game balance.

Conclusion: In short, we chose to prioritize a stable multiplayer experience over complex, unnecessary features.

---

3. Stakeholders and Target Users

Description of End Users and Their Needs

To develop *Timeless Empire*, we first analyzed our Target Audience. We are targeting players aged 16 and up, because our game relies on complex strategic thinking and historical evolution rather than just reflexes.

We identified three main User Needs and translated them into code features:

- **The need to Relax**: We built a Turn-Based engine so players aren't stressed by time. We also want to code a Ghost Mode, ensuring that 'Game Over' doesn't mean the fun stops—users stay connected to the server to chat and influence the game.
- **The need for Strategy**: We implemented a resource economy (Gold, Wood, Food) that forces players to fight over territory, rather than just camping.
- **The need for Social Interaction**: This is why we chose a 5-player architecture with a custom Chat System, allowing for alliances and betrayal.

Finally, regarding the Technical Actors, the game operates on a Client-Server model. This requires stable internet connectivity, as our Python server synchronizes the game state for all 5 computers simultaneously.

---

4. General Description of the Expected Product

Overall Vision of How It Works

The game is a turn-based multiplayer strategy game.
Each player manages an empire and takes turns to perform actions such as:

- Developing resources
- Expanding their empire
- Interacting with other players through battles or exchanges

The game progresses over time following a timeline until a victory condition is reached.

Main Use Cases or Typical Scenarios

- A group of **3 to 5 players** competes against each other in a strategic environment.
- Players play together for fun and entertainment.
- Players may try to disrupt or annoy other players through strategic actions.

Global Constraints

- Each player must have a computer with internet access.
- The game is free and requires no financial investment from players.
- The interface must be simple and accessible.
- Players need a minimum level of critical thinking to fully enjoy the game.

---

5. Functional Requirements

Function 1: Turn-Based Game System

Description of the Function

The game features a turn-based system in which players take their turns individually, each with a 2-minute timer.

Input and Output Conditions

- **Inputs**: Player actions (move, build, exchange, attack …)
- **Outputs**: Updated game state (resources, player status, map changes …)

Priority Level

- Mandatory

Function 2: Resource System

Description of the Function

The resources are given randomly each turn and are updated for each building you have (if you don't have a farm you can’t receive wheat).
Resources are important for buildings and village management.
People need food to live and building resources to be created.

Input and Output Conditions

- **Inputs**: Player actions (buildings) and system equilibre (base resources for every players)
- **Outputs**: Resources

Priority Level

- Mandatory

Function 3: Exchange System

Description of the Function

One of the main functions of the game is the different exchange between each of the players. In fact they have to interact to make alliances, trade resources or declare war.

Input and Output Conditions

- **Inputs**: Player actions (resources, alliances, war …)
- **Outputs**: Player actions (alliance, war), map modification (travel of resources)

Priority Level

- High

Function 4: Skill Tree

Description of the Function

Each player has to improve his skill tree to be the first to obtain a new weapon, building… and be the first to pass an age. All of this to be the most powerful empire on the map.

Input and Output Conditions

- **Inputs**: Player action, Ressources.
- **Outputs**: Status and Time

Priority Level

- High

Function 5: Combat mode

Description of the Function

Two players can fight each other to gain resources or terrain with troops. The combat is automatic and if two troops have the same power, a random variable defines the winner.

Input and Output Conditions

- **Inputs**: Player action, System equilibre
- **Outputs**: Map change, Ressources, Update health of troops

Priority Level

- Mandatory

6. Non-Functional Requirements

Expected Performance

- The game shall respond quickly to player actions.
- The interface shall remain responsive during gameplay.
- Turn transitions shall occur without noticeable delays.

Reliability, Safety, and Maintainability

- The system shall remain stable during extended play sessions without crashes or critical errors.
- Game state data (units, resources, player actions) shall be consistently updated and saved to prevent data loss.
- Errors shall be handled gracefully without breaking gameplay.
- The code shall be organized and maintainable.

Usability and Accessibility

- The user interface shall be intuitive and responsive.
- The game shall provide clear visual and audio feedback for player actions and events.
- Controls shall be configurable to match player preferences.
- The game shall be playable without advanced technical knowledge or prior experience with RTS games.

Possible Regulatory Constraints

- The game shall comply with intellectual property laws.
- Only original or properly licensed assets shall be used.
- The game shall not include offensive, discriminatory, or illegal content.

---

7. Assumptions and Dependencies

Initial Assumptions Made by the Project Team

- A domain name may be purchased to host the official game website.
- An online server may be rented to host multiplayer services and online features.
- The development team will have continuous access to the required development tools (game engine, IDE, version control system).
- All team members are assumed to remain available throughout the duration of the project.
- The project scope will remain stable and will not require major redesigns of core mechanics.
- The game will be developed primarily for desktop platforms (PC).
- The target audience is assumed to not have basic familiarity with strategy games.Sufficient documentation and learning resources for the chosen technologies will remain available.
- Version control services (e.g., Git repositories) are required for collaboration and code management.
- The project may depend on third-party networking or hosting services.
- External libraries or assets may be used.
- The development timeline depends on the academic schedule and project deadlines.
- No critical external dependency has been identified that would currently block or significantly delay development.

---

8. Acceptance and Validation Criteria

The validation of the project will be based on several testing and evaluation methods to ensure that the game meets the initial objectives and requirements.

The game will be tested at different stages of development through beta testing sessions. Family members, friends, or classmates will play early, intermediate, and final versions of the game. Their feedback will help identify bugs, gameplay issues, and usability problems, as well as measure the overall enjoyment of the game.

Functional testing will be carried out to verify that all core systems work correctly, including the turn-based system, resource management, battles, development, exchanges. These systems must function both independently and together without causing errors.

The game will also be tested for stability and performance during extended play sessions. The objective is to ensure that the game does not crash, remains responsive, and handles multiple players correctly throughout a full game.

Usability will be evaluated by observing how easily new players understand the game rules and interface. The game should be intuitive and playable without requiring external explanations or advanced technical knowledge.

The project will be considered successful if:

- The game runs correctly without critical errors
- A complete game session with 3 to 5 players can be played
- The main features are fully functional
- Players enjoy the game
- Only minor bugs remain that do not affect gameplay

If all these conditions are met, the project will be validated and considered successful.

---

9. Schedule and Estimation

Major Steps of the Project

- Map creation
- Completion of basic code and core mechanics
- First test of the base game
- Addition of features
- Testing phase
- Final project completion
- Bug fixing if necessary
- Final testing

Key Milestones

- Playable prototype
- First complete multiplayer session
- Final validated version

Estimated Completion Times

- Map creation: **1 week**
- Basic mechanics: **1.5 months**
- First test: **1 week**
- Feature additions: **1–2 months**
- Testing: **1 week**
- Final version: **1 month**
- Bug fixing and final test: **2 weeks**

Preliminary Cost Estimation

- Total budget: approximately **60 €**
- Development time: around **2 hours per person per week**
- Required resources:
    - Computer with internet access
    - Discord for communication
    - Git for version control
    - Coffee for night work or rush

Appendices

Glossary of Specific Terms Used

- **Turn-based game**: A game where players act one after another.
- **Empire**: A player-controlled civilization that manages cities, resources, and military forces to expand its influence and achieve victory.
- **Pygame**: A Python library for video game development.

References

- Python documentation
- Pygame documentation
