Fontys HOGESCHOOL ICT

Research plan

Jordy Walraven

Table of Contents

Introduction

As we start on Semester 6, we're presented with a variety of new concepts. Each one requires delving into research, a process that can be incredibly rewarding. However, the most significant hurdle I encounter lies in architecting my system. Specifically, I'm fascinated by the most effective design patterns and approaches to tackling the inherent challenges that arise when creating a robust and scalable system.

Problem

The sheer number of design and architecture patterns available is overwhelming. Each seems to offer a perfect solution, but selecting the right one for a system designed to handle a million users feels like navigating a maze. The core challenge lies in striking a delicate balance for my non-functional requirements.

Through research, I aim to identify the ideal design patterns and architectural approaches that can overcome these challenges. This will help me build a system that is not only scalable but also remains understandable and manageable for future development and maintenance.

Research questions

Main question: How to design and deploy a web application that enables user-submitted system performance evaluation for game compatibility, ensuring scalability, security, reliability maintainability, and high performance?

Sub questions:

- What architectural styles best suit the requirements of my system?
    - Library (Literature Research): Research different architectural styles (e.g., microservices, monolithic, layered) and their strengths and weaknesses. Understand how they address common non-functional requirements.
    - Workshop (IT-Architecture- sketching): Sketch the basic architecture that my application will have.

- How do different services or components communicate with each other?
    - Library (Literature Research): Research common communication protocols used in distributed systems (e.g., REST APIs, message queues). Understand their pros and cons in different scenarios.
    - Workshop (multi-criteria decision making): Compare the different methods with each other and how they fit the non-functional requirements.
    - Workshop (Prototyping): Create a low-fidelity prototype to explore different communication mechanisms.

- How can I ensure that the application is secure:
    - Library (Literature Research): Research on best practices for securing web applications, including data encryption, user authentication, and secure coding practices.
    - Lab (Security test) : Find and prioritize vulnerabilities in the system and the impact on confidentiality, integrity and availability of information

- How to deploy a scalable application?
    - Literature (Available Product Analysis): Research cloud platforms and containerization technologies (e.g., Docker, Kubernetes) that can help scale your application efficiently. Explore existing solutions and best practices for deploying scalable applications.
    - Workshop (Prototyping): Create a simple environment where my application can be hosted.

- How do I make sure the application is reliable deployed without breaking issues?
    - Literature (Available Product Analysis): Research existing deployment tools and frameworks that can automate and streamline the deployment process. Analyze their features and see how they can help mitigate deployment risks.
    - Workshop (Prototyping): Create a simple environment that can reliable deploy an application

- How do I make sure the application is reliable running in the cloud?
    - Library (Literature Research): Research on best practices for cloud-based application reliability, including redundancy, load balancing, and disaster recovery strategies.
    - Workshop (Prototyping): Experiment with deploying the application in a cloud environment and monitoring its performance.

Deliverables

Main Deliverable:

- A web application that enables user-submitted system performance evaluation for game compatibility

Sub-question Deliverables:

- Non-functional requirements list: A documented list of identified non-functional requirements for the application.
- Architectural style comparison table: A table comparing different architectural styles (e.g., microservices, monolithic, layered) with their strengths, weaknesses, and how they address your non-functional requirements.
- Technical Documentation: Things like a C4 diagram and a description of the design choices.
- Communication protocol comparison table: A table comparing different communication protocols (e.g., REST APIs, message queues) with their pros, cons, and suitability based on non-functional requirements. And a basic prototype showing this interaction.
- Client interaction prototype: A low-fidelity prototype showcasing how the client interacts with different services in your application.
- CI: A pipeline that checks the quality of the product and makes sure it gets deployed reliably.
- Deployment: A deployment that is scalable, reliable and secure.