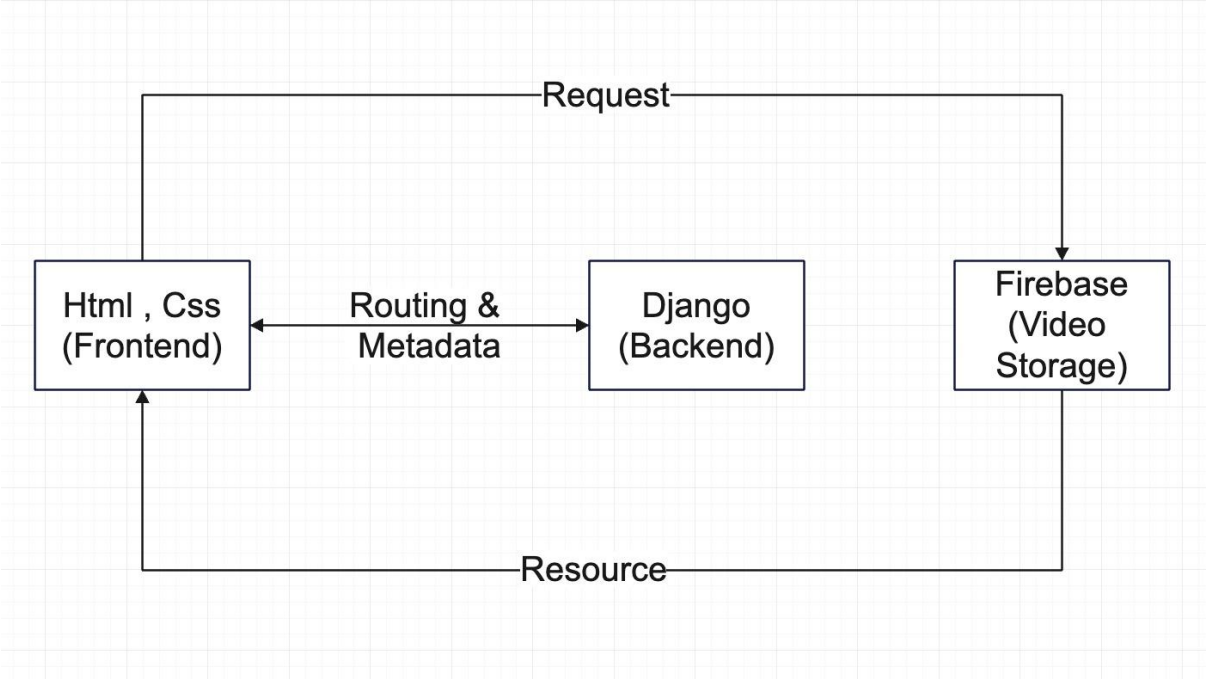
Software Requirements Specification for Video Streaming Django Project with Firebase Database

(Team Members Name and SRN)
Adnan - PES1UG213xx
Nagasaketh - PES1UG213xx
Nandan - PES1UG21361



0. Intended Audience and Reading Suggestions

0.1 Intended Audience

The primary audience for this document includes:

- **Development Team:** Developers and programmers responsible for implementing the video streaming testing Django project.
- **Project Stakeholders:** Individuals and groups with a vested interest in the successful development and deployment of the video streaming testing application.

0.2 Reading Suggestions

Readers are advised to review the entire document to gain a comprehensive understanding of the project requirements. Key sections for specific stakeholders are as follows:

- **Development Team:** Focus on sections related to system components, features, and functional requirements to guide the development process.
- **Project Stakeholders:** Pay special attention to the purpose, scope, and non-functional requirements to understand the goals and constraints of the project.

1. Introduction

1.1 Purpose

The purpose of this document is to outline the software requirements for the development of a video streaming testing Django project with a Firebase database. This application will focus on features such as auto-quality adaptation, cross-browser compatibility, corrupt video detection, internet disconnection detection, Synchronisation of audio and video with consistent quality, and dynamic resizing.

1.2 Scope

The video streaming testing Django project will cover the following key features:

- Auto quality adaptation for optimal user experience.
- Cross-browser compatibility to ensure consistent performance across different web browsers.
- Corrupt video detection to identify and handle corrupted video files.
- Internet disconnection detection to manage interruptions during video playback.
- Synchronisation of audio and video with consistent quality.
- Dynamic resizing to adapt video dimensions based on device and screen size.

1.3 Definitions, Acronyms, and Abbreviations

- Django: High-level Python web framework.
- Firebase: Cloud-based platform for mobile and web applications.

- API: Application Programming Interface.
- UI: User Interface.

2. System Overview

2.1 System Description

The video streaming testing Django project with Firebase database will consist of the following main components:

1. Django Web Application:
Responsible for handling user requests, managing the database, and rendering the user interface.
2. Firebase Database:
Cloud-based database to store video streaming-related data and configurations.
3. A modern browser:
In which we provide the way to stream as well as implement auto quality adaptation, cross-browser compatibility, corrupt video detection, internet disconnection detection, Synchronisation of audio and video, and dynamic resizing.

2.2 System Features

2.2.1 Auto Quality Adaptation

- The system shall automatically adapt video quality based on the user's internet speed and device capabilities.

2.2.2 Cross-Browser Compatibility

- The system shall ensure consistent video streaming performance across major web browsers, including but not limited to Chrome, Firefox, Safari, and Edge.

2.2.3 Corrupt Video Detection

- The system shall detect and handle corrupted video files to prevent playback issues.

2.2.4 Internet Disconnection Detection

- The system shall detect internet disconnections during video playback and handle them gracefully.

2.2.5 Synchronise Audio and Video

- The system shall ensure synchronisation between audio and video elements for a seamless viewing experience.

2.2.6 Dynamic Resizing

- The system shall dynamically resize video dimensions to adapt to different devices and screen sizes.

3. Functional Requirements

3.1 Django Web Application

3.1.1 User Authentication

1. The system shall implement user authentication to control access to video streaming features.

3.1.2 Database Management

1. The Django application shall interact with the Firebase database to store and retrieve video streaming-related data.

3.2 Video Streaming Engine

3.2.1 Auto Quality Adaptation

1. The system shall implement algorithms for adaptive streaming based on available bandwidth and device capabilities.

3.2.2 Cross-Browser Compatibility

1. The system shall conduct thorough testing to ensure compatibility with major web browsers.

3.2.3 Corrupt Video Detection

1. The system shall include mechanisms to identify and handle corrupted video files during playback.

3.2.4 Internet Disconnection Detection

1. The system shall monitor internet connectivity and handle disconnections during video streaming.

3.2.5 Synchronise Audio and Video

1. The system shall Synchronise audio and video elements to avoid latency and playback issues.

3.2.6 Dynamic Resizing

1. The system shall dynamically resize video dimensions based on device and screen size.

4. Non-Functional Requirements

4.1 Performance

1. The system shall provide low-latency video streaming under varying network conditions.
2. The auto-quality adaptation algorithm shall respond quickly to changes in network speed.

4.2 Reliability

1. The system shall handle video playback interruptions gracefully, providing a seamless user experience.
2. Corrupt video detection mechanisms shall be reliable and minimise false positives.

4.3 Usability

1. The user interface shall be intuitive and user-friendly for configuring video streaming settings.
2. Error messages shall be clear and guide users in resolving issues.

4.4 Security

1. User authentication mechanisms shall be secure to prevent unauthorised access.
2. Video streaming data stored in the Firebase database shall be protected with appropriate security measures.

5. Constraints

1. The system must comply with Django and Firebase compatibility requirements.
2. Cross-browser compatibility may have limitations based on browser specifications.