1. Project Title

Streamify – Unified Music & Video Streaming Platform

---

2. Introduction

Streamify is a web-based streaming application developed using the Django framework to provide a unified platform for both music and video streaming services. In the current digital ecosystem, streaming services are fragmented, with separate platforms dedicated to audio and video content. Streamify aims to bridge this gap by integrating both content types into a single centralized system.

The platform provides secure user authentication, cloud-based media storage, subscription-based content access, playlist management, payment record handling, and content search functionality. The system follows a modular architecture design, making it scalable, maintainable, and suitable for future commercial expansion.

The project was developed by a team of four members within a 10-day structured development cycle, emphasizing teamwork, modular design, and cloud integration.

---

3. Problem Statement

Modern digital entertainment platforms are segmented into separate services for music streaming and video streaming. This fragmentation requires users to maintain multiple subscriptions, manage separate accounts, and switch between platforms for different types of content consumption.

Additionally, many small-scale platforms lack integrated subscription control, secure authentication, and scalable cloud storage solutions.

There is a need for a unified streaming system that consolidates both music and video services into a single application while maintaining secure authentication, scalable storage, and modular backend architecture. Streamify addresses this need by providing a centralized and structured streaming solution.

---

4. Objectives

The primary objective of Streamify is to design and implement a unified streaming platform that supports both music and video content under one subscription ecosystem.

The system aims to:

- Implement secure OTP-based user authentication.
- Integrate cloud-based media storage using Cloudinary.
- Provide subscription-based access control for premium content.
- Enable users to create and manage playlists across media types.
- Record and track payment transactions.
- Demonstrate scalable and modular Django architecture.

Another key objective is to simulate real-world SaaS application development within a limited 10-day timeframe through efficient team collaboration.

---

5. Scope of the Project

The scope of Streamify includes the development of a fully functional web application capable of handling user authentication, media streaming, subscription validation, playlist management, and search functionality.

The platform supports both MP3 (audio) and MP4 (video) formats. Media files are uploaded to Cloudinary cloud storage, and secure URLs are stored in a MySQL database. The system ensures that premium content is accessible only to users with active subscriptions.

The project is limited to a web-based implementation and does not include mobile application development, AI recommendation systems, or live third-party payment gateway integration at this stage. However, the architecture is designed to support such enhancements in the future.

---

6. Technologies Used

Frontend:
HTML, CSS, JavaScript for user interface and interaction.

Backend:
Django (Python) following the Model-View-Template (MVT) architecture.

Database:
MySQL relational database for structured data storage.

Media Storage:
Cloudinary cloud-based storage for audio and video file management.

Other Tools & Concepts:
Git for version control
REST-style API views
Django Admin Panel for backend management
Virtual environment for dependency isolation

---

7. System Architecture

Streamify follows a client-server architecture built on Django's MVT design pattern. The frontend interacts with the backend through HTTP requests. The backend processes authentication, subscription validation, playlist operations, and content retrieval logic.

The database layer stores structured data such as user information, content metadata, subscription details, and payment records. Media files are stored separately in Cloudinary cloud storage to ensure scalability and reduce server load.

The overall system flow is:

User → Frontend Interface → Django Backend → MySQL Database
↘ Cloudinary
Media Storage

This layered approach ensures separation of concerns, improved maintainability, and future scalability.

---

8. Modules Description

Admin Module

The Admin module provides backend administrative control through Django's built-in admin interface. Administrators can manage users, upload and modify content, define subscription plans, and monitor payment records.

User Module

The User module handles registration, OTP-based login, session management, and profile control. It ensures secure access and validates subscription status before granting premium content access.

Content Module

The Content module manages audio and video metadata, handles Cloudinary upload integration, and facilitates media streaming via stored URLs.

Playlist Module

This module allows users to create, update, and manage personalized playlists containing both music and video content.

Subscription Module

The Subscription module tracks plan activation, expiry dates, and enforces access restrictions based on subscription validity.

Payment Module

The Payment module stores transaction details and links payments to subscription activation records for structured tracking.

Search Module

The Search module enables keyword-based content discovery and improves user experience by allowing efficient navigation across available media.

---

9. Database Design

The database design is relational and structured to maintain integrity across multiple entities.

The primary entities include User, Content, Playlist, Subscription, and Payment. A user can create multiple playlists, a subscription is linked to a user account, and payments are associated with subscription records.

Foreign key relationships ensure consistency and prevent orphan records. Media URLs generated by Cloudinary are stored within the Content model to enable secure streaming.

---

10. Implementation Plan

The project was completed within 10 days through structured phase-based development.

The first phase involved requirement analysis and architectural planning. The second phase focused on database schema implementation and model creation. Backend development, including authentication, subscription logic, and Cloudinary integration, followed next. Frontend integration and testing were completed in subsequent stages.

Task distribution among four team members allowed parallel development and efficient milestone achievement.

---

11. Testing Strategy

Testing included functional testing, integration testing, and manual scenario validation.

The OTP authentication flow was tested to ensure secure login. Subscription validation logic was verified to restrict unauthorized access. Cloudinary media streaming was tested to confirm correct file retrieval.

Database relationships were validated to ensure proper linking between users, subscriptions, playlists, and payments. End-to-end testing was performed to simulate real user interactions.

---

12. Expected Outcome

The expected outcome of Streamify is a secure, scalable, and unified streaming platform capable of handling both music and video services within one system.

The project demonstrates full-stack web development capabilities, cloud integration, database design proficiency, and modular SaaS architecture implementation.

It serves as a strong prototype for potential future commercialization.

---

13. Future Enhancements

Future enhancements include implementing AI-based personalized recommendations, integrating live payment gateways for real-time transactions, deploying the system on scalable cloud hosting platforms, and developing mobile applications.

Additional improvements may involve multi-tier subscription models, advanced analytics dashboards, CDN-based streaming optimization, and microservices-based deployment for high scalability.

---

14. Conclusion

Streamify successfully achieves its goal of building a unified music and video streaming platform within a limited 10-day timeline by a team of four members.

The system integrates secure authentication, Cloudinary cloud storage, MySQL database management, subscription validation, playlist functionality, and modular backend design.

The project reflects strong technical implementation, teamwork coordination, and scalable system architecture. With further enhancements and production-level deployment, Streamify has the potential to evolve into a commercially viable streaming platform.