Project Plan Document

for

< Music Streaming Web-App >

Version 1.0 approved

Prepared by <Group 5>

1. Introduction

1.1. Purpose

The digital revolution has transformed the way we experience music, bringing the vast universe of songs from around the globe to our fingertips. MusicStreaming stands at the forefront of this transformation, offering a seamless and personalized music streaming service to millions of users. This document delineates the requirements for MusicStreaming's software system, ensuring that it continues to meet the evolving needs of its users with efficiency, reliability, and innovation.

1.2. Document Conventions

Headings Times New Roman/15 font size/Bold

Sub Headings Times New Roman/14 font size/Bold

1.3. Intended Audience and Reading Suggestions

The intended audience for our music app includes music enthusiasts, casual listeners, and professionals alike, seeking a seamless and immersive music experience. Whether you're a dedicated audiophile looking for curated playlists or an artist interested in reaching new audiences, our app caters to your diverse needs. For those new to the platform, we recommend exploring our recommendations curated playlists to discover new music tailored to your taste. Additionally, artists can benefit from our platform's tools for promoting their music and connecting with fans. With something for everyone, our music app invites users to explore, discover, and enjoy music in all its forms.

1.4. Product Scope

The scope of this project encompasses the development and enhancement of the MusicStreaming application, a premier music streaming service. The project aims to deliver a user-friendly platform that provides access to a vast library of music, podcasts, and videos from creators all over the world.

The key objectives include:

- Delivering high-quality audio streaming to users globally.
- Providing personalized content recommendations based on user preferences.
- Ensuring seamless integration across various devices and platforms.
- Maintaining a robust and scalable infrastructure to support a growing user base.

- Incorporating social features that allow for sharing and discovering content within user communities.

1.5. References

Gantt chart:
https://app.smartsheet.com
Document base:
plan_document.docx (sharepoint.com)

2. Project Schedule

Conceptualization phase (1 week):

Define App's Purpose(2 days): Gather the team and brainstorm the core purpose of the music app. Determine whether it will focus on streaming, discovery, social features, or a combination.

Identify Target Audience(3 days): Conduct market research to identify demographics most likely to engage with the app. This could include age groups, music preferences, geographical location, etc.

Key Features Identification(3 days): Prioritize features that align with the app's value proposition and are feasible within the initial development timeline and budget.

Planning phase (1 weeks): Create a detailed project plan, including timelines, resources, and budget. Identify key milestones and potential risks.

Project Scope Definition(1 days): Review the outcomes of the conceptualization phase and define the scope of the project, including the features to be included in the initial release (MVP). Determine any constraints or limitations, such as budget, time, or technological requirements.

Create a Detailed Project Plan(2 days): Break down the project into smaller tasks and create a timeline for each, considering dependencies and resource availability. Use project management tools like Gantt charts or Kanban boards to visualize the project timeline and allocate tasks to team members.

Resource Allocation(3 days): Identify the resources required for each task, including human resources (developers, designers, testers, etc.) and technological resources (software, hardware, etc.).

Budget Estimation(1 days): Estimate the costs associated with each phase of the project, including development, design, testing, marketing, and ongoing maintenance.

Risk Assessment and Mitigation(3 days): Identify potential risks and challenges that may arise during the project, such as technical issues, resource constraints, scope creep, or market fluctuations.

Design phase(1 week): Develop wireframes and prototypes of the app. Finalize the user interface and user experience design.

User Experience (UX) Design(1 day): Conduct user research to understand user preferences, behaviors, and pain points related to music app usage.

User Interface (UI) Design(3 day): Translate wireframes into high-fidelity mockups, incorporating branding elements, color schemes, typography, and visual assets.

Visual Design Refinement(1 days): Refine the UI design based on feedback received during usability testing and stakeholder reviews. Pay attention to detail, optimizing visual elements for different screen sizes and resolutions to ensure a seamless experience across devices.

Interaction Design(1 days): Define interactive elements such as buttons, gestures, animations, and transitions to enhance user engagement and usability.

Development phase(3 weeks): Code the app's functionality. This stage often includes several sub-stages, such as setting up the database, developing the front-end and back-end, integrating APIs, and testing each feature.

Environment Setup(1 day): Set up development environments, version control systems, and project management tools.

Backend Development(3 days): Develop backend functionalities, including user authentication, database management, and API integrations.

Frontend Framework Selection(1 day): Choose a suitable frontend framework or technology stack based on project requirements and team expertise.

UI Implementation(2 days): Translate UI designs into code, using HTML, CSS, and JavaScript to build responsive and visually appealing user interfaces.

Core Feature Development(3 days):Develop core app features such as music discovery, playlist creation, search functionality, and user preferences.

Integration with Third-Party Services(3 days): Integrate with third-party services such as music streaming platforms, payment gateways, and analytics tools.

Unit and Integration Testing(2 days): Conduct unit tests to validate individual components and functions, ensuring they meet specifications and perform as expected.

Performance Optimization(2 days): Optimize code for performance, scalability, and efficiency, addressing bottlenecks and improving app responsiveness. Conduct load testing to simulate heavy user traffic and identify areas for optimization.

Testing phase(1 week): Conduct thorough testing to identify and fix bugs. This includes unit testing, integration testing, and user acceptance testing.

Test Case Creation(3 days): Develop comprehensive test cases based on functional requirements, user stories, and use cases.

Test Execution(3 days): Execute test cases according to the test plan, following predefined test procedures and documenting test results.

Performance and security Testing(3 days): Perform load testing, stress testing, and scalability testing to assess the app's performance under various conditions, including peak usage and high traffic volumes.security assessments and penetration testing to identify vulnerabilities, such as authentication flaws, data leaks, and injection attacks.

Launch phase(1 week): Prepare for launch by setting up app store accounts, creating promotional materials, and planning a marketing strategy.

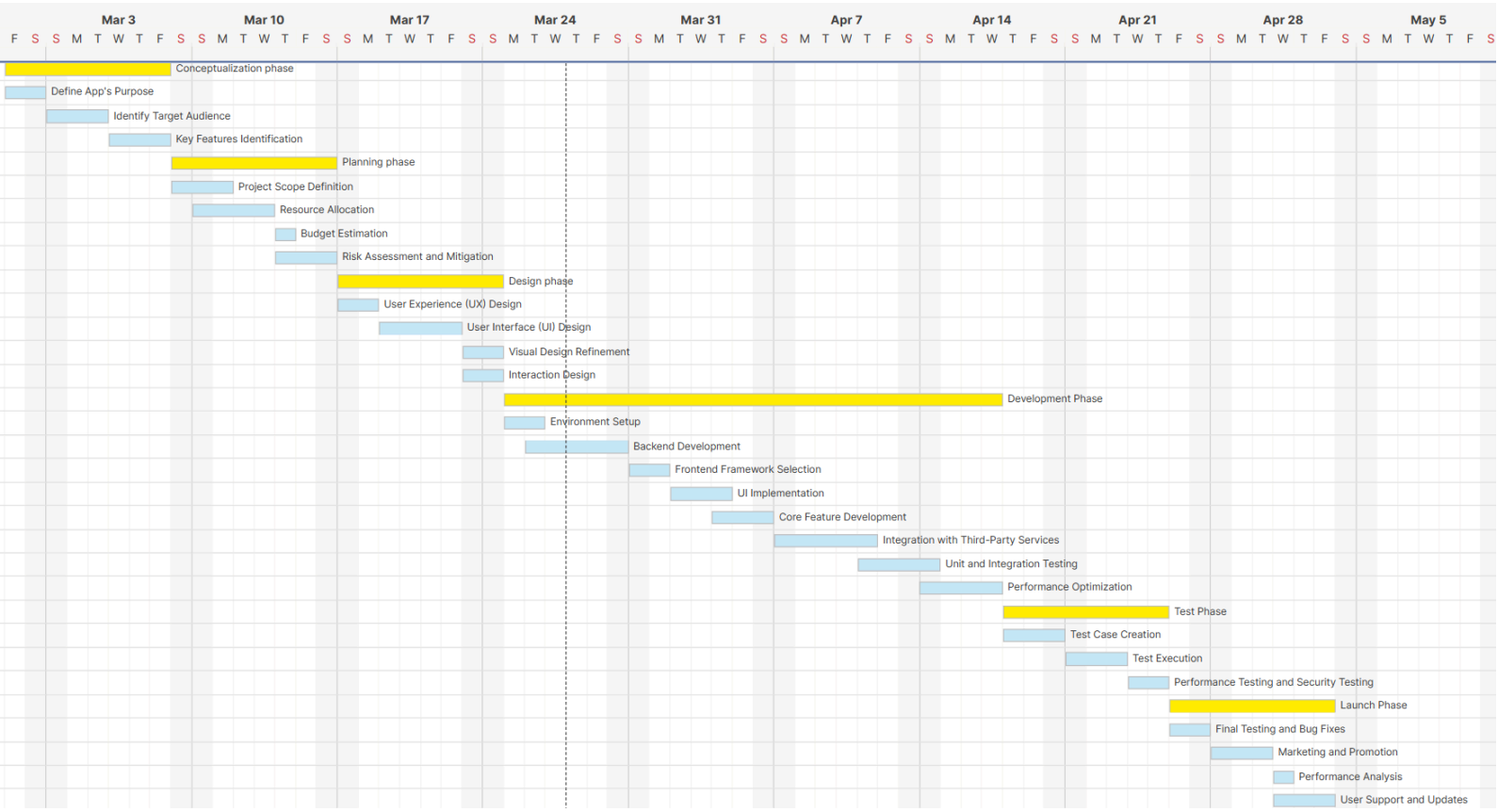
Final Testing and Bug Fixes(2 days): Conduct thorough testing of the app to identify and address any remaining bugs or issues.

Marketing and Promotion(3 days): Develop a marketing strategy to generate buzz and attract users to the app.

Performance Analysis(1 day): Analyze app performance metrics, including downloads, user engagement, retention rates, and revenue generation.

User Support and Updates(3 days): Provide ongoing customer support to address user inquiries, issues, and feature requests in a timely manner.

Post-Launch (Ongoing): Monitor the app’s performance, gather user feedback, and make necessary updates or improvements.



Gantt chart for project schedule

3. Resource Allocation

3.1. Software

Integrated Development Environment (IDE):

Git: Used for collaborative development, version control, and managing codebase changes.

Testing Frameworks:

Jest: JavaScript testing framework used for frontend testing.

XCTest: Testing framework for writing unit tests for iOS apps.

Backend Development:

Programming Languages:

Node.js: Used for building the backend server and API endpoints.

Python: Used for building the backend sever.

Database Management System (DBMS):

MySQL: Relational database management system used for storing user data, playlists, and metadata.

Frontend Development:

Programming Languages and Frameworks:

HTML, CSS, JavaScript: Languages used for building the frontend user interface.

React.js: JavaScript library used for building interactive UI components.

API Integration:

Third-Party APIs:

Spotify API: Used for integrating music playback, search, and metadata retrieval functionalities.

Apple Music API: Utilized for integrating with the Apple Music service on iOS devices.

Google Play Music API: Used for integrating with the Google Play Music service on Android devices.

3.2. hardware

Development Machines:

Desktop Computers: High-performance desktop computers equipped with powerful processors (e.g., Intel Core i7 or AMD Ryzen 7), ample RAM (e.g., 16GB or more), and SSD storage for fast compilation and development.

Operating Systems: Windows, macOS, or Linux, based on developers' preferences and platform-specific development requirements.

Laptops:

High-end laptops with specifications similar to desktop computers, enabling developers to work remotely or while traveling.

3.3. People

Development Team:

Lead Developer: Ngô Lê Hoàng

Frontend Developer: Nguyễn Đức Khánh, Trần Đức Anh, Đỗ Minh Quang

Backend Developer: Trần Thế Mạnh, Ngô lê Hoàng

UI/UX Designer: Đỗ Minh Quang

Quality Assurance (QA) Team: Nguyễn Đức Khánh

QA Lead: Ngô Lê Hoàng

QA Tester: Trần Thế Mạnh

3.4. Materials

3.4.1. Data Sets:

Music Metadata:

Comprehensive data sets containing information about songs, albums, artists, genres, release dates, and popularity rankings.

Used for populating the app's music library, providing accurate metadata for search, recommendation algorithms, and display purposes.

3.4.2. User Preferences and Behavior:

Anonymized data sets capturing user interactions, listening habits, preferences, and feedback.

Used for personalizing recommendations, improving user engagement, and optimizing app features based on user behavior.

3.4.3. Legal and Regulatory Documents:

Copyright Laws and Licensing Agreements:

Legal documents, regulations, and guidelines governing the use of copyrighted music, licensing agreements, and royalty payments in digital music distribution.

Used for ensuring compliance with intellectual property laws, securing necessary licenses, and mitigating legal risks associated with content distribution.

3.4.4. Data Privacy and Security Regulations:

Legislation, standards, and guidelines related to data privacy, user consent, and security measures for protecting user data.

Used for implementing data protection measures, obtaining user consent for data collection, and complying with regulatory requirements (e.g., GDPR, CCPA).

4. Task Breakdown and Roles

Task	Time(day)	Assign to
Conceptualization phase	7	
Define App's Purpose	2	Nguyễn Đức Khánh, Ngô Lê Hoàng
Identify Target Audience	3	Đỗ Minh Quang, Trần Thế Mạnh
Key Features Identification	3	Nguyễn Đức Khánh, Trần Đức Anh
Planning phase	8	
Project Scope Definition	3	Nguyễn Đức Khánh
Resource Allocation	4	Nguyễn Đức Khánh, Trần Đức Anh
Budget Estimation	1	Đỗ Minh Quang
Risk Assessment and Mitigation	3	Trần Thế Mạnh, Ngô Lê Hoàng, Trần Đức Anh
Design phase	7	
User Experience (UX) Design	2	Đỗ Minh Quang, Nguyễn Đức Khánh
User Interface (UI) Design	4	Đỗ Minh Quang, Nguyễn Đức Khánh
Visual Design Refinement	2	Nguyễn Đức Khánh, Trần Đức Anh
Interaction Design	2	Đỗ Minh Quang

Development Phase	23	
Environment Setup	2	Ngô Lê Hoàng
Backend Development	5	Ngô Lê Hoàng
Frontend Framework Selection	2	Ngô Lê Hoàng
UI Implementation	3	Ngô Lê Hoàng
Core Feature Development	3	Trần Thế Mạnh
Integration with Third-Party Services	5	Ngô Lê Hoàng
Unit and Integration Testing	4	Trần Thế Mạnh
Performance Optimization	4	Trần Thế Mạnh
Test Phase	7	Ngô Lê Hoàng
Test Case Creation	3	Ngô Lê Hoàng
Test Execution	3	Trần Thế Mạnh
Performance Testing and Security Testing	2	Ngô Lê Hoàng, Trần Thế Mạnh
Launch Phase	7	
Final Testing and Bug Fixes	2	Ngô Lê Hoàng, Trần Thế Mạnh
Marketing and Promotion	3	Đỗ Minh Quang, Nguyễn Đức Khánh

Performance Analysis	1	Trần Thế Mạnh
User Support and Updates	3	Nguyễn Đức Khánh, Trần Đức Anh
Post-Launch		
Iterative Updates and Maintenance		Nguyễn Đức Khánh
Strategic Planning and Future Roadmap		Đỗ Minh Quang

5. Meeting schedule

Frequency: Weekly meetings will be held every Monday to review project progress, discuss any challenges or roadblocks, and plan tasks for the upcoming week.

Format: Meetings will be conducted virtually via video conferencing tools such as Zoom to accommodate remote team members. An agenda will be circulated prior to each meeting to ensure discussions remain focused and productive.

Communication Tools:

Email: Used for official communication, sharing important updates, and circulating meeting agendas and minutes.

Decision-making Process:

Consensus Building: Decisions will be made collaboratively through consensus-building discussions during weekly meetings or via Slack channels. Input from all team members will be considered, and decisions will be based on a collective understanding of project requirements, goals, and constraints.

Project Manager Authority: In cases where consensus cannot be reached or immediate decisions are required, the project manager will have the authority to make final decisions, ensuring the project stays on track and deadlines are met.

Documentation: Decisions made and rationale behind them will be documented in meeting minutes and shared with the team to ensure transparency and accountability.

6. Risk Assessment and Mitigation Strategies

6.1. technical risk

Risk: Integration challenges with third-party APIs for music streaming and payment processing.

Mitigation:

Conduct thorough API compatibility testing during the development phase.

Establish direct communication channels with API providers to troubleshoot and resolve integration issues promptly.

Implement fallback mechanisms and error handling procedures to minimize service disruptions.

6.2. Resource Risks:

Risk: Key team members leaving the project unexpectedly.

Mitigation:

Cross-train team members to ensure knowledge sharing and redundancy in critical roles.

Maintain documentation and clear task assignments to facilitate smooth transition in case of personnel changes.

Consider hiring freelancers or contractors as backup resources to mitigate workload impact.

6.3. Schedule Risks:

Risk: Delays in development due to unforeseen technical challenges or scope changes.

Mitigation:

Implement Agile methodologies such as Scrum or Kanban to adapt to changing requirements and mitigate schedule risks.

Break down project tasks into smaller, manageable chunks with defined deadlines to maintain momentum and track progress.

Regularly review and adjust project timelines based on ongoing assessments of progress and potential risks.

6.4. Security Risks:

Risk: Data breaches or unauthorized access compromising user privacy and security.

Mitigation:

Implement robust authentication and authorization mechanisms to protect user accounts and sensitive data.

Encrypt data transmission using secure protocols (e.g., HTTPS) and encrypt stored data to prevent unauthorized access.

Conduct regular security audits and penetration testing to identify vulnerabilities and proactively address security risks.

6.5. Market Risks:

Risk: Changes in market trends, user preferences, or competitive landscape affecting app adoption and retention.

Mitigation:

Conduct market research and stay updated on industry trends to anticipate changes and adapt the app's features and strategies accordingly.

Monitor competitor activities and user feedback to identify emerging opportunities and areas for differentiation.

Implement agile development practices to iterate quickly, pivot when necessary, and stay responsive to market dynamics.

6.6. Legal and Compliance Risks:

Risk: Non-compliance with data protection regulations (e.g., GDPR, CCPA) or copyright infringement issues related to music licensing.

Mitigation:

Consult legal experts to ensure compliance with relevant laws and regulations governing data privacy, intellectual property rights, and digital content distribution.

Obtain necessary licenses and permissions for music streaming and ensure adherence to copyright laws and licensing agreements.

Implement robust terms of service, privacy policies, and user consent mechanisms to protect user rights and mitigate legal risks.

7. Review and Approval

- Project Scope and Requirements:

Approval: Stakeholders (Project Sponsor, Product Owner)

Purpose: Ensure alignment with project goals and stakeholder expectations.

- Design Documents and UI/UX Prototypes:

Approval: Design Lead, Project Manager
Purpose: Validate design concepts and usability before development.

- Development Progress and Feature Completion:

Approval: Project Manager, Product Owner
Purpose: Confirm that developed features meet requirements and expectations.

- Test Plan and Test Cases:

Approval: QA Lead, Project Manager
Purpose: Ensure comprehensive test coverage and alignment with project scope.

- Change Requests and Scope Adjustments:

Approval: Project Manager
Purpose: Assess impact on project timelines, resources, and budget before implementation.

- Deployment Readiness and Release Approval:

Approval: Project Manager, Operations Team
Purpose: Confirm readiness for production deployment and release to end-users.

- Post-Release Evaluation and Feedback Analysis:

Approval: Project Manager, Stakeholders
Purpose: Review user feedback, performance metrics, and analytics data to inform future iterations and improvements.

- Project Closure and Lessons Learned:

Approval: Project Manager, Steering Committee
Purpose: Formal acceptance of project deliverables and discussion of lessons learned for future projects.