FRONTEND DEVELOPMENT

PROJECT DOCUMENTATION- **rhythmictunes: your melodic companion**

## 1.Introduction :

* **PROJECT TITLE:rhythmictunes: your melodic compani on**
* TEAM ID: NM2025TMID43218
* TEAM SIZE: 4
* TEAM LEADER: santhiya.B
* ROLE: CODING AND DEVELOPMENT
* TEAM MEMBER:sandhiya.G
* ROLE: CODING AND DEVELOPMENT
* TEAM MEMBER: sandhana .s
* ROLE: CODING AND DEVELOPMENT
* TEAM MEMBER: Roshini.S
* ROLE: CODING AND DEVELOPMENT

## 1. Project Overview

Project Title: [Insert Project Name]

Description:

This project is developed to [state the main purpose, e.g., build a web/mobile application that solves a particular problem]. It provides users with [briefly explain core functionality, e.g., an efficient way to manage tasks, share resources, or track progress].

# Objectives:

To create a user-friendly solution that addresses [the problem].

To improve efficiency and accuracy in [domain].

To provide scalable and reliable software that can be expanded in the future.

## Stakeholders:

Project Owner / Sponsor: [Name/Organization]

Development Team: [Developers, Designers, Testers]

End Users: [Students, Businesses, General Public, etc.]

## 2. Project Scope

In Scope:

Design and development of the application.

Implementation of key features [list major features].

Integration with database and APIs.

User interface design and usability testing.

Out of Scope:

Third-party integrations not specified in requirements.

Support for legacy systems beyond current compatibility.

## 3. Requirements

Functional Requirements:

User registration and authentication.

Data storage and retrieval.

[Feature 1: e.g., Task management, Payment processing].

[Feature 2: e.g., Notifications, Reports, Analytics].

Non-Functional Requirements:

Application must be secure and protect user data.

System should handle at least [X] concurrent users.

Availability of 99% uptime.

Easy navigation and responsive design.

# 4. Architecture & Technologies

System Architecture:

Frontend: [React / Angular / Flutter / HTML, CSS, JS]

Backend: [Node.js / Java / Python / PHP]

Database: [MySQL / MongoDB / PostgreSQL]

APIs: [REST / GraphQL]

Tools & Platforms:

Version Control: GitHub / GitLab

IDE: VS Code, IntelliJ, PyCharm

Deployment: AWS / Azure / Google Cloud / On-Premise

# 5. Design

UI / UX:

Clean, minimal, and responsive design.

Accessibility features included.

Database Design:

ER Diagram with relationships (Users, Transactions, Activities, etc.).

Integration Points:

Payment Gateway (if applicable).

External APIs (e.g., maps, weather, SMS/email).

# 6. Immpleentation Plan

Phases & Milestones:

1. Requirement Analysis – Collect and finalize requirements.

2. Design Phase – Prepare wireframes, architecture diagrams.

3. Development Phase – Code the application in modules.

4. Testing Phase – Conduct unit, integration, and user acceptance testing.

5. Deployment – Deploy on production server.

6. Maintenance – Continuous monitoring and updates.

Roles & Responsibilities:

Developer: Coding and module integration.

Tester: Writing and executing test cases.

Project Manager: Monitoring timeline and deliverables

# 7. Testing & Quality Assurance

Testing Types:

Unit Testing

Integration Testing

System Testing

User Acceptance Testing (UAT)

Success Criteria:

All modules perform as expected.

No major bugs or crashes.

Meets performance benchmarks.

# 8. Deployment & Maintenance

Deployment on [Cloud / Local Server / Mobile App Stores].

Separate environments for Development, Testing, and Production.

Regular updates and bug fixes.

Backup and recovery strategy in place.

# 9. Risks & Mitigation

Risks:

Requirement changes mid-development.

Integration failures with third-party APIs.

Security vulnerabilities.

Mitigation:

Clear requirement documentation before development.

Regular sprint reviews.

Conduct penetration and vulnerability testing

# 10. Timeline & Budget

Timeline (example):

Requirement Analysis – 2 weeks

Design – 3 weeks

Development – 8 weeks

Testing – 3 weeks

Deployment – 1 week

Budget:

Development Costs

Testing & QA

Hosting and Infrastructure

Maintenance & Support

# 11. Appendices

Glossary of technical terms.

References to APIs, frameworks, and libraries used.

Screenshots / diagrams / mockups.