Software Requirements Specification

for

CROWDFUNDING PLATFORM

Prepared by

BHARANIDHARAN K(23CSR034)

BHUPESH SANTHA KUMAR(23CSR036)

DHARSAN SP(23CSL254)

Kongu Engineering College,Perundurai,Erode.  
  
18/08/2025

Table of Contents

1. Introduction  
   1.1 Purpose …………………………………………………….……………. 2  
   1.2 Intended Audience ………………………………………………………. 2  
   1.3 Project Scope ……………………………………………………………. 2  
   1.4 Definitions, Acronyms and Abbreviations………………………………. 3  
   1.5 References ……………………………………….……………………….3
2. Overall Description  
   2.1 Product Perspective…………………………………………...…………. 3  
   2.2 Product Features…………………………………………………………. 3  
   2.3 User Classes and Characteristics…………………………………………. 4  
   2.4 Operating Environment……..……………………………………………. 4  
   2.5 Design and Implementation Constraints…………………………………. 4
3. System Features  
   3.1 User Authentication………………………………………………………..5  
   3.2 Campaign Creation and Listing……………………………..……………. 5  
   3.3 Donation Management…………………………………………………… 5  
   3.4 Payment Gateway Integration……………………………………………. 5  
   3.5 Notifications and Updates…………………………………………..……. 5  
   3.6 Analytics Dashboard…………………………………………………..…. 6
4. External Interface Requirements  
   4.1 User Interfaces……………………………………………………………. 6  
   4.2 Hardware Interfaces………………………………………………..…..…. 6  
   4.3 Software Interfaces…………………………………………………….…. 6  
   4.4 Communication Interfaces………………………………………..………..6
5. Other Nonfunctional Requirements  
   5.1 Performance Requirements…………………..…………………..………..7  
   5.2 Security Requirements…………………..………………………..…….....7  
   5.3 Usability Requirements…………………..…………………..……..……..7  
   5.4 Scalability Requirements…………………..……………………..………..7
6. Other Requirements…………………..……………………..…………………8

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 1. Introduction

# 1.1 Purpose

# The purpose of this document is to define the functional and non-functional requirements of the Crowdfunding Platform.

# This system will enable project creators to launch fundraising campaigns, donors to contribute securely, and administrators to monitor platform activities.

# 1.2 Intended Audience

# Project Creators (Fundraisers): Individuals or organizations creating campaigns to raise funds.

# Donors (Backers): Users contributing financially to campaigns.

# Administrators: Personnel managing platform operations, user accounts, and fraud detection.

# Developers & Testers: Teams responsible for implementing, testing, and maintaining the system.

# Stakeholders & Investors: Individuals overseeing project success and business growth.

# 1.3 Project Scope

# The crowdfunding platform is a web-based application that provides a secure and user-friendly environment for fundraising.

# Key features include:

# User registration and authentication.

# Campaign creation, management, and listing.

# Donation processing with integrated payment gateways.

# Real-time progress tracking and notifications.

# Analytics dashboard for creators.

# Administrative tools for fraud detection and platform management.

# 1.4 Definitions, Acronyms, and Abbreviations

# Donor/Backer: A user who contributes funds to a campaign.

# Creator/Fundraiser: A user who creates and manages campaigns.

# Admin: A privileged user responsible for managing the platform.

# Campaign: A fundraising initiative created by a project owner.

# Payment Gateway: A service (e.g., Stripe, PayPal, UPI) that processes online transactions securely.

# MERN Stack: MongoDB, Express.js, React.js, Node.js – technologies used in development.

# 1.5 References

# IEEE Software Requirements Specification (SRS) Template.

# Stripe and PayPal Payment Gateway API Documentation.

# OWASP Security Guidelines for Web Applications.

# MongoDB and Express.js Documentation.

# React.js and Node.js Official Documentation.

# 2.Overall Description

# 2.1 Product Perspective

# The crowdfunding platform is a web application built using the MERN stack (MongoDB, Express, React, Node.js). It works like an online marketplace where creators can raise money for their projects, and supporters (donors) can contribute safely.

# 2.2 Product Features

# User Login and Signup: Secure registration and login for all users.

# Campaign Creation: Creators can start a fundraising campaign with details like title, description, goal, and deadline.

# Campaign Listing: Donors can view all available campaigns.

# Payment System: Donors can make payments easily using online methods.

# Progress Tracking: Real-time updates on how much money has been raised.

# Dashboard for Creators: Simple charts and stats for creators to see campaign performance.

# 2.3 User Classes and Characteristics

# Donor: Should have a simple interface with only a few steps to donate.

# Creator: Should be able to manage multiple campaigns easily.

# Admin: Has control to check campaigns, prevent fraud, and monitor users.

# 

# 2.4 Operating Environment

# Can be accessed through common web browsers like Chrome, Edge, and Firefox.

# Works on both computers and mobile devices (mobile-friendly design).

# Backend uses Node.js and Express.

# Database uses MongoDB to store campaign, user, and payment details.

# 2.5 Design and Implementation Constraints

# Must follow payment security rules to keep transactions safe.

# Should be able to handle many users at the same time.

# All personal and payment data must be stored securely.

# Must follow privacy and data protection rules.

# 3.System Features

**3.1 User Authentication**

* + Users can **sign up, log in, and recover their password**.
  + **Role-based access control :** Donor, Creator, and Admin have different levels of access.
  + Authentication is handled using **JWT (JSON Web Token)** for secure sessions.

**3.2 Campaign Creation and Listing**

* + Creators can start campaigns by adding:

Title, description, funding goal, deadline, media (images/videos).

* + Campaigns are stored in the **MongoDB database** and displayed dynamically on the frontend.
  + Search and filter options are available for donors.

**3.3 Donation Management**

* + Donors can contribute using multiple **payment methods (UPI, cards, net banking, PayPal, Stripe).**
  + Contributions are tracked in the **database** with unique transaction IDs.
  + Transparent donation history is available for both creators and donors.

**3.4 Payment Gateway Integration**

* + Integrated with **Stripe / PayPal API / UPI** for processing payments.
  + All transactions are **encrypted (SSL/TLS)** to ensure security.
  + System automatically generates **digital receipts**.

**3.5 Notifications and Updates**

* + Real-time notifications via **email (SMTP)** and **SMS (API integration)**.
  + Alerts are sent for:
    - New donations
    - Campaign milestones
    - Campaign completion or expiry

**3.6 Analytics Dashboard**

* + Built with **charts and graphs (using libraries like Chart.js or D3.js)**.
  + Shows:
    - **Funding progress** (raised vs. target).
    - **Donor statistics** (number of donors, amount per donor).
    - **Engagement metrics** for creators.

**4. External Interface Requirements**

**4.1 User Interfaces**

* + The system will provide a **responsive user interface (UI)** designed with **React.js**.
  + Role-based dashboards:
    - **Donor Dashboard:** Browse campaigns, donate easily, and view donation history.
    - **Creator Dashboard:** Create campaigns with title, description, goal, and media; track progress in real time.
    - **Admin Dashboard:** Monitor campaigns, users, and detect/report fraudulent activities.

**4.2 Hardware Interfaces**

* + Accessible on **standard desktops, laptops, tablets, and smartphones**.
  + Requires **basic internet connectivity**; no specialized hardware is needed.
  + Optimized for **touchscreen devices** to enhance mobile experience.

**4.3 Software Interfaces**

* + Backend: Node.js with Express.js framework.
  + Database: MongoDB for storing user details, campaigns, and transactions.
  + Payment Integration: APIs such as Stripe, PayPal, and UPI.
  + Notification Services: Email and SMS APIs (e.g., SMTP, Twilio).
  + Operating System Compatibility: Works across Windows, macOS, Linux, Android, and iOS via browsers.

**4.4 Communication Interfaces**

* + HTTPS protocol will ensure secure communication between client and server.
  + RESTful APIs will handle requests and responses between frontend and backend.

**5. Other Nonfunctional Requirements**

**5.1 Performance Requirements**

* The system should support 10,000+ concurrent users without performance degradation.
* Response time for critical actions (e.g., donations, campaign creation) should be less than 2 seconds under normal load.

5.2 Security Requirements

# All sensitive data (user credentials, payment details) must be encrypted using industry standards (e.g., AES, HTTPS/TLS).

# Secure authentication mechanisms such as JWT (JSON Web Tokens) for session handling.

# Fraud detection and prevention mechanisms for suspicious activities (e.g., multiple fake campaigns, unusual donation patterns).

# Regular data backups to ensure recovery in case of failures.

# 5.3 Usability Requirements

# The platform must provide a mobile-friendly and responsive UI.

# Support for accessibility features (screen readers, proper color contrast, keyboard navigation).

# Simple navigation flow with intuitive forms, menus, and notifications for all user roles (Donor, Creator, Admin).

# 5.4 Scalability Requirements

# The system should support horizontal scaling to handle increasing user traffic.

# Deployment should be compatible with cloud platforms (AWS, Azure, or GCP) for elastic resource management.

# Ability to add new features or integrate third-party APIs with minimal architectural changes.

**6. Other Requirements**

* Admin Panel with Fraud Detection  
  The system must include an admin panel equipped with tools for monitoring campaigns, detecting fraudulent activities, and managing suspicious user accounts.
* Legal and Tax Compliance  
  The platform must comply with applicable taxation laws, donation regulations, and financial reporting standards to ensure transparency and legal operation.