Software Requirements Specification

for

Startup Idea Validator

Version 1.0 approved

Institution : Keshav Memorial Institute of Technology

Project Id : G566

Date Created: August 30, 2025

Table of Contents

Table of Contents

1. Introduction

1.1 Purpose

This document outlines the software requirements for the Startup Idea Validator, a web-based platform designed to help entrepreneurs and students evaluate startup ideas based on algorithmic scoring and feedback.

1.2 Document Conventions

All requirement IDs will be prefixed with REQ-, and all sections follow IEEE SRS standards.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers, testers, project managers, startup mentors, and students who will interact with the system either as users or contributors.

1.4 Product Scope

The system provides a structured interface for submitting startup ideas, scoring them automatically based on viability criteria, and returning feedback and suggestions.

1.5 References

Presentation: startup_idea_validator.pptx
Architecture Diagram and Class Diagram (submitted images)

2. Overall Description

2.1 Product Perspective

Startup Idea Validator is a self-contained full-stack web application developed using the MERN stack (MongoDB, Express.js, React.js, Node.js). It integrates AI-based scoring logic using OpenAI's API and allows users to validate their startup ideas in a structured and objective manner.

2.2 Product Functions

- User registration and authentication
- Guided startup idea submission form
- Automated scoring based on key viability criteria
- Instant feedback and suggestions
- Viewing and managing previously submitted ideas
- Chat-based interactions for refining ideas

2.3 User Classes and Characteristics

- Students and aspiring entrepreneurs: Use system to validate ideas.
- Mentors or evaluators: (Future) Can comment on or score ideas.
- Admins: Oversee system, manage users and ideas.

2.4 Operating Environment

- Web-based application
- Runs on any modern browser (Chrome, Firefox, Edge)
- Backend server (Node.js) on cloud infrastructure
- MongoDB Atlas (or local MongoDB instance)
- Llama 3 API for idea analysis

2.5 Design and Implementation Constraints

- Must use MERN stack
- Must use Llama 3 API for scoring logic
- JWT for authentication
- Responsive web design for desktop and mobile users

2.6 User Documentation

- Online user manual
- Onboarding tutorial in app
- Help section accessible through UI

2.7 Assumptions and Dependencies

- Llama3 API remains accessible and affordable
- MongoDB cloud availability
- Users will have access to internet and modern browsers

3. External Interface Requirements

3.1 User Interfaces

The system will provide a responsive web-based user interface designed using React.js. Main screens include:

- Login/Register screen
- Idea submission form (Problem, Market, USP, Business Model)
- Score and feedback display page
- Idea history and details page
- Optional mentor feedback chat interface (future enhancement)

3.2 Hardware Interfaces

This application is designed to operate on standard web-enabled devices (PCs, tablets, smartphones). No special hardware interfaces are required.

3.3 Software Interfaces

- MongoDB (cloud/local): Stores user, idea, and scoring data
- OpenAI API: Used for analyzing and generating feedback
- JWT/Bcrypt: Used for secure user authentication
- Node.js/Express: Backend API services

-PineconeDB : It’s a managed vector database

3.4 Communications Interfaces

All data transfer between frontend and backend uses secure HTTPS connections. JWT tokens are passed in HTTP headers. External communications include RESTful calls to Llama3 APIs or any other API versions if needed.

4. System Features

4.1 User Registration and Authentication

4.1.1 Description and Priority
High priority. Provides basic access control and user management.

4.1.2 Stimulus/Response Sequences
User submits registration/login form → System validates → JWT issued on success.

4.1.3 Functional Requirements
REQ-1: System shall allow new users to register.
REQ-2: System shall allow existing users to log in using email and password.
REQ-3: Passwords shall be hashed using bcrypt.
REQ-4: JWT tokens shall be used for session management.

4.2 Idea Submission and Scoring

4.2.1 Description and Priority
High priority. Core feature for evaluating startup ideas.

4.2.2 Stimulus/Response Sequences
User fills idea form → System submits to backend → Score and feedback returned.

4.2.3 Functional Requirements
REQ-5: System shall allow users to submit startup idea details.
REQ-6: System shall call OpenAI API with structured data.
REQ-7: System shall generate and display score and feedback.

4.3 View Idea History

4.3.1 Description and Priority
Medium priority. Enables users to view previously submitted ideas.

4.3.2 Stimulus/Response Sequences
User clicks 'History' → System fetches and displays previous ideas and scores.

4.3.3 Functional Requirements
REQ-8: System shall display all past ideas submitted by the user.
REQ-9: System shall allow viewing each idea’s score and feedback.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

System shall respond to user requests within 2 seconds for 95% of operations under normal load.

5.2 Safety Requirements

No specific safety requirements as this is a non-critical system.

5.3 Security Requirements

- User passwords must be encrypted using bcrypt.
- All API endpoints should require authentication (JWT).
- Secure HTTPS must be used for all communication.

5.4 Software Quality Attributes

The application must be:
- Usable and responsive
- Maintainable (modular code)
- Scalable for concurrent users
- Reliable with fallback for OpenAI failures

5.5 Business Rules

- Users can only access and modify their own ideas.
- Admin can view all user data for moderation.

6. Other Requirements

- Database must store each idea with timestamp and user ID.
- Future version may include:

    ->export to PDF and team collaboration features.

    ->allowing professors and researchers access to user idea by giving permissions and adding a rating system.

Appendix A: Glossary

JWT - JSON Web Token
USP - Unique Selling Proposition
API - Application Programming Interface

Appendix B: Analysis Models

Refer to the attached class diagram and architecture diagram for visual models.

Appendix C: To Be Determined List

TBD-1: Integration with mentor feedback
TBD-2: Multi-user collaboration feature