Software Requirements Specification

for

Dvinder

Version 1.0 approved

Prepared by

Name - Manan Nileshkumar Desai
Roll No - CE029
ID - 18CEUOS128
Batch - C2
Semester - 6
Subject - Software Engineering

DDU

Prepared on 13-12-2020

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this SRS document is to give a detailed description of ***Dvinder.*** *Dvinder is like tinder but for only developers or a person who is related to computer programming. This will help developers to find project partners or some good friends. This will connect developers with each other and help to make a good community.*

1.2 Document Conventions

This document is formed using IEEE SRS format, headings are in bold capital letters and wherever a necessary diagram is provided.

1.3 Intended Audience and Reading Suggestions

Intended audience will be developers or any one who is related to computer programming across the globe. Also the developer of this project, testers.

1.4 Product Scope

Dvinder is a ***P****rogressive* ***W****eb* ***A****pplication(PWA). The main goal is to create a good community for developers across the globe. Developers can share their code snippets, project ideas, other good images etc. and like or dislike or to others. This will help them to find project partners or some good friends. Developers who liked each other's posts are matched so they can talk with each other.*

1.5 References

IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

2. Overall Description

2.1 Product Perspective

The system will be developed in Client-Server architecture. The project stack will be MERN Stack(MongoDB, Express.js, React.js, Node.js). Client side will be React.js, server side API will be created in Node.js + Express.js and MongoDB as a NoSQL database.

2.2 Product Functions

The functionalities of Dvinder are as follow:

- *Authentication module*
- *User profile module*
- *Feeds module*
- *Chatting module*
- *Offline Availability*
- *Admin module*

2.3 User Classes and Characteristics

User	*Users can be developer or any person who is related to computer programming*
Admin	*Admin is the lead or manager of the system who can manage all the users.*

2.4 Operating Environment

Since this is a Progressive Web Application, it should work on major browsers like Google Chrome, Mozilla FireFox, Microsoft Edge or IE, Opera mini, Safari etc..

2.5 Design and Implementation Constraints

Proposed Progressive Web Application can be developed in MERN stack, in which React.js will be used to create UI, Node.js and Express.js will be used to create API and MongoDB as a database to store user data. Also as this is a Prograssive Web App you need an internet connection to use with up to date data and also you need modern browsers like Google Chrome, Mozilla FireFox, Microsoft Edge or IE, Opera mini, Safari.

2.6 User Documentation

To provide help to users for help center, reports or any other queries we will provide a user manual and documentation link.

2.7 Assumptions and Dependencies

As this is a Prograssive Web Application we assume that the user has a good internet connection to use it and also modern browsers like Google Chrome, Mozilla FireFox, Microsoft Edge or IE, Opera mini, Safari. to open it.

3. External Interface Requirements

3.1 User Interfaces

Since this is a Prograssive Web application so it should provide a very User friendly and responsive interface. It should be easy to navigate without any learning curve involved. A decent and pleasant appearance with ease of navigation should help users.

3.2 Hardware Interfaces

The hardware requirement at the user end is really simple and the Web App will be available on the hardware that can run a basic simple browser, provided the hardware should be competent enough during peak times for the web servers.

3.3 Software Interfaces

The software interfaces at the user ends are following. Users can run this web app using any Operating System like Windows, MacOS or any Linux based system. Users need modern browsers with good internet connection. It will use a cloud based NoSQL database like MongoDB.

3.4 Communications Interfaces

For communication It will use https for basic communication. For emails and help center users can use SMTP. Also for user chatting we will use web sockets.

4. Functional Requirement

4.1 Users

4.1.1 User Authentication

Users need to be authenticated to access all the features of the Dvinder.
Input: User Authentication Details
Output: System verify the details and provide particular Message

4.1.2 Update profile

Users can update their profiles like username, profile photo etc.
Input: Profile detail that need to be updated
Output: System verify user and update the particular details of the user.

4.1.3 Show feeds

Users can see posts according to the location data that user provided.
Input: User ask for more feeds to see
Output: System filters out feeds based on user location and shows it to the user.

4.1.4 Upload feed

Users can upload code snippets, project ideas or other good resources or images.
Input: User provide feeds data to system
Output: System takes all the data and creates feed for the user.

4.1.5 Subscribe for notifications

Users can subscribe for notifications for particular information like feeds,matches etc.
Input: User ask for notification subscription
Output:System creates subscription based on user browser and device combination.

4.1.6 React on feed

Users can react on any feed i.e- like or dislike the feed.
Input: React to any feed
Output: System add that feed in particular category.

4.1.7 Get all match

Users can get all the users that are matched with them.
Input: User ask for all matches
Output:System provides that data.

4.1.8 Chat with match

Users can chat with their matches.
Input: Users can ask to chat with their match
Output: System makes socket connection between them and allows them to chat.

4.1.9 Ask for help or report

Users can ask for help or report anything.
Input: Provide help or report data
Output: System sends data to admin and does according admin response.

4.2 Admin

4.2.1 Manage feeds

Admin can manage feeds like remove vulnerable feeds..
Input: Find feed vulnerable to policy and ask to delete
Output: System check the policy and do according.

4.2.2 Checks reports and provide help

Admin can check reports and help that are asked by users.
Input: Do some actions based on reports and help
Output: System does particular help to the user.

4.2.3 Make other admin for specific roles

Admin can make other admin for a particular role.
Input: Provide data of other users and roles.
Output: System Give the user a particular role.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

As mentioned earlier, the PWA should be able to operate on all major web-browsers with all of its fundamental functions . And it should not shut-down in any condition

5.2 Security Requirements

The system should provide a secure login to the users by using advanced secure login algorithms and provide access only to the authorized users as security is the key requirement of this system.

5.3 Software Quality Attributes

System should be-

- ➢ *Consistent in performance*
- ➢ *Safe and Secure*
- ➢ *Robust*
- ➢ *Scalable*
- ➢ *Flexible*
- ➢ *User friendly*
- ➢ *Efficient*
- ➢ *Inter-operable*
- ➢ *Upgradable*
- ➢ *Available all the time.*

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

Developer: A person related to computer programming.

Appendix B: Analysis Models

Appendix C: To Be Determined List