# Software Requirement Specification
for
# **FreshFuel**

Project name: FreshFuel.com
Document: SRS (Software Requirement Specification)
Author: Sruti Ranjan Pradha
Published on:

Name: Sruti Ranjan Pradhan
Enrollment No: 230160203066

# Table of Contents

# 1. Introduction

### 1.1 Purpose
The purpose of this software is to help the user to maintain a healthy lifestyle.
### 1.2 Product Scope
- The application allows users to:
- Users get access 24/7 from anywhere.
- The software suggests a healthier alternative to whatever they search for.
- So that they can maintain a healthier lifestyle.
### 1.3 Product Value
It helps people maintain a healthier lifestyle, decreases the unconditional death ratio, and decreases various types of severe diseases.
### 1.4 Intended Audience
Gym lovers, who are health conscious and love to maintain decent healthy lifestyles.
### 1.5 Intended Use
The user will search based on their wants and requirements to obtain the desired and best results.
### 1.6 General Description
Our software helps people to find healthier alternatives whatever they search for.

## 2. Functional Requirements

### 2.1 Constraints of the product:
Name, Price, Nutrition Facts, etc. are mentioned on our website.

1. **Organic Foods**
    - Fruits and vegetables
    - Nuts and seeds
    - Grains (quinoa, brown rice, oats)
    - Dairy products (organic milk, yogurt)
2. **Whole Foods**
    - Fresh produce
    - Whole grains
    - Lean meats and fish
3. **Superfoods**
    - Chia seeds

    - Goji berries
    - Spirulina
    - Matcha powder
4. **Healthy Snacks**
    - Protein bars
    - Dried fruits
    - Organic popcorn
    - Kale chips
5. **Beverages**
    - Herbal teas
    - Cold-pressed juices
    - Kombucha
    - Smoothies

**Supplements and Vitamins**

1. **Vitamins and Minerals**
    - Multivitamins
    - Vitamin D
    - Omega-3 supplements
2. **Herbal Supplements**
    - Echinacea
    - Turmeric
    - Ashwagandha
3. **Protein Supplements**
    - Whey protein(Isolate & concentrate)
    - Plant-based protein

**Personal Care Products**

1. **Skincare**
    - Natural moisturizers
    - Organic sunscreens
    - Essential oils
2. **Haircare**
    - Sulfate-free shampoos
    - Natural conditioners
    - Hair masks with natural ingredients
3. **Oral Care**
    - Fluoride-free toothpaste

    - Bamboo toothbrushes
    - Natural mouthwash

## 3. External Interface Requirements

### 3.1 External interface requirements

Homepage:

- Clear and visually appealing design.

- Prominent search bar for products.

- Login/Sign-up options.

- Our Logo.

**Navigation:**

- Intuitive navigation with a well-organized menu.

- Clear categories of products.

- Easy-to-use filters for refining search results.

**Search and Order place:**

- User-friendly search functionality with auto-suggestions.

- Advanced search options, including filters for price range, amenities, etc.

**Product Pages:**

- Detailed information about all the products we have.

- High-quality images and interactive UI.

- Clear pricing details, including taxes and fees.

- Rating and reviews from another user.

**User Accounts:**

- Simple and secure login and registration process.

- User profiles with booking history and preferences.

- Option to wishlist your favorite products.

**Notifications:**

- Email or SMS notifications for order confirmation, reminders, and updates.

- Alters for special promotions or discounts.

**Customer Support:**
- Easily accessible help or support section.
- Live chat support for real-time assistance.

## 3.2 Hardware interface requirements

### 3.2.1 Supported Devices

**Desktop Computers:**
- Windows PCs (Running recent versions of Windows)

Macintosh computers (running recent versions of macOS).

Linux-based systems.

All smartphones: IOS and Android

iOS devices (iPhones and iPads).

**Tablets:**
Compatibility with popular tablets, including iPads and Androids tablets.

**Browsers:**

Google Chrome, Mozilla Firefox, Apple Safari, Microsoft Edge, and other major browsers.

### 3.2.2 Network Requirements

**Internet Connectivity:**
The software should be designed to operate over standard internet connections.
Support for both high-speed broadband and mobile data connections.
**Bandwidth:**
Optimize the application for varying levels of bandwidth to accommodate users with slower internet connections.
**Redundancy and Failover:**
Implement redundancy and failover mechanisms to ensure service availability even during network disruptions.
**CDN Integration:**
Use Content Delivery Networks (CDNs) to optimize content delivery and reduce latency.

### 3.2.3 Communication Protocols:

**HTTP/HTTPS:**

Use the Hypertext Transfer Protocol (HTTP) or its secure variant (HTTPS) for communication between clients and servers.

**3.3 Software Interface Requirements**
**3.3.1 Frontend Components:**
**HTML/CSS/JavaScript:**
Foundation for building the user interface and handling client-side interactivity.
11
Frameworks and libraries like React, Angular, or Vue.js can be employed for a structured frontend architecture.
Frontend Framework (e.g., React):
React components for building a modular and dynamic user interface.

Integration with state management libraries like Redux or Context API for managing application state.
·
**CSS Framework (e.g., Bootstrap):**
Utilize a CSS framework for responsive and consistent styling across different devices.
·
**JavaScript Libraries (e.g., jQuery):**
If necessary, use JavaScript libraries for DOM manipulation and event handling.
**WebSockets (e.g., Socket.IO):**
Employ WebSockets for real-time communication, enabling features like live chat or updates.
·
**GraphQL (optional):**
Implement GraphQL for more efficient and flexible data retrieval, allowing the front end to request only the data it needs.
**3.3.2 Backend Components:**
Backend Framework (e.g., Node.js, Django, Flask, Ruby on Rails):

Use a backend framework to handle server-side logic, routing, and database interactions.
12
**API Endpoints:**
Define RESTful or GraphQL API endpoints to facilitate communication between the front end and back end.
·

**Database (e.g., MongoDB, PostgreSQL, MySQL):**
Store and retrieve data from a database to manage user information, bookings, and other relevant data.
·
**Authentication (e.g., OAuth, JWT):**
Implement secure authentication mechanisms to ensure user data and transactions are protected.
·
**Payment Gateway Integration (e.g., Stripe, PayPal):**
Connect with payment gateways to handle financial transactions securely.
·
**Server-Side Caching (e.g., Redis):**
Implement server-side caching for frequently accessed data to improve performance.
Middleware:
Use middleware components for request processing, logging, and security.
·
**Server:**
Deploy the backend on a server, which could be a cloud-based solution like AWS, Azure, or Google Cloud.
13

## 3.4 Communication Interface Requirements

### 3.4.1 Email Communication:

**User Registration:**
Send a welcome email upon user registration, providing information about the account and services.
·
**Order Placed Confirmations:**
Instantly email users with detailed confirmations upon successful order placed.
**Updates and Alerts:**
Notify users about any changes to their product shipment.
**Promotions and Deals:**
Periodically send promotional emails featuring special deals, discounts, or exclusive offers to encourage repeat business.
**Feedback and Reviews:**
Request user feedback and reviews post-order, contributing to improving services and building customer trust.

**Password Recovery:**
Provide a secure mechanism for users to reset their passwords through email verification. 14
**Newsletter Subscriptions:**
Allow users to subscribe to newsletters for updates on travel trends, destinations, and exclusive offers.
### 3.4.2 Embedded Forms and Communication on the Website:
**Contact Forms:**
Implement user-friendly contact forms on the website for general inquiries and support requests.
·
**Customer Support Chat:**
Integrate a live chat feature for real-time customer support, allowing users to ask questions or get assistance while browsing the site.
·
**Feedback Forms:**
Include feedback forms on various pages to collect user opinions about the website's usability and content.
·
**Cancellation and Refund Requests:**
Provide an embedded form for users to submit cancellation or refund requests with necessary details.
·
**Subscription Forms:**
If applicable, offer subscription forms for users to sign up for newsletters or alerts directly on the website.
### 3.4.3 SMS Communication:
**Booking Confirmations and Updates:**
Send instant SMS notifications for booking confirmations, updates, and important product-related information.
**Two-Factor Authentication (2FA):**
Implement SMS-based two-factor authentication for enhanced account security.
### 3.4.4 Social Media Communication:
**Social Sharing:**
Enable users to share their product buying experiences on social media platforms directly from the website.
**Customer Engagement:**
Use social media channels to engage with customers, address queries, and promote special offers.

**3.4.5 Push Notifications:**
**Mobile App Alerts:**
If there is a mobile app, implement push notifications for timely alerts, updates, and personalized offers.
·
**Browser Push Notifications:**
Allow users to opt-in for browser push notifications for updates even when they are not actively using the website.

## 4. Non-functional requirements
### 4.1 Security
**Privacy and Data Protection Regulations:**
**General Data Protection Regulation (GDPR):**
Obtain explicit user consent before collecting personal data.

Clearly communicate the purpose of data processing and provide users with control over their data.

Implement measures for data portability, access, and deletion upon user request.

Ensure the security of user data through encryption and other security measures.
**Payment Card Industry Data Security Standard (PCI DSS):**
If handling payment information, comply with PCI DSS standards to ensure the secure processing of credit card data.
**Electronic Communications Privacy Act (ECPA):**
Comply with ECPA regulations when intercepting, disclosing, or using electronic communications data.
**CAN-SPAM Act:**
Adhere to CAN-SPAM requirements for commercial emails, including providing clear opt-out mechanisms.

**Data Breach Notification Laws:**
Comply with data breach notification laws by promptly notifying users and authorities in the event of a data breach.
**Privacy by Design:**
Integrate privacy considerations into the design and development process of the website.

Minimize data collection to what is necessary for the intended purpose.
**Cookie Consent (ePrivacy Directive):**

Obtain user consent for the use of cookies and similar tracking technologies as required by the ePrivacy Directive.

**Children's Online Privacy Protection Act (COPPA):**

If the website is accessible to children, comply with COPPA regulations regarding the collection of personal information from children under 13 years of age.

**Data Retention Policies:**

Establish clear data retention policies and only retain user data for the necessary duration.

## 4.2 Capacity

### 4.2.1 Current Storage Needs:

**User Profiles:**

User account information, including personal details, preferences, and order history.

**Media Content:**

High-resolution images of products.

**Search and Recommendation Data:**

History search queries and user interaction to enhance personalization.

Data related to user preference and recommendation.

**Reviews and Ratings:**

User-generated content, such as reviews and ratings for products etc.

**Logs and Analytics:**

Server logs, error logs, and analytics data to monitor website performance and user behavior.

**Third-Party Integrations:**

Data related to third-party services, such as payment gateways, APIs, and external systems.

### 4.2.2 Future Storage Needs:

**User Growth:**

Expansion of user base may lead to increased storage requirements for user profiles and associated data.

**Additional Services:**

Introduction of new services may require storage for additional types of data, such as car rentals, vacation packages, or travel insurance.

**Enhanced Personalization:**

As personalization efforts grow, more data related to user preferences and behavior may need to be stored.

**Big Data Analytics:**
If implementing advanced analytic or machine learning models, there may be an increase in storage needs for large datasets.
**Improved Media Content:**
Enhanced visual content, including 3D image  of products, virtual reality experiences.
**Regulatory Compliance:**
Compliance with evolving data protection regulations may necessitate additional storage for audit logs and data governance.
**Backup and Disaster Recovery:**
Enhanced backup and disaster recovery mechanisms may require additional storage capacity.

## 4.3 Compatibility

**Hardware Requirement for the Software**
Processor: Minimum 1 GHz; Recommended 2GHz or more

Ethernet connection (LAN) OR a wireless adapter (Wi-Fi)

Hard Drive: Minimum 32 GB; Recommended 64 GB or more

Memory (RAM): Minimum 1GB; Recommended 4GB or above

Sound card speaker

Some classes require a camera andmicrophone

A modern wa or application that support accesing online services.

## 4.4 Reliability

Critical failure time would depend on various factors:
**System Maintenance:** Regular maintenance and updates to ensure optimal performance and security.

**Infrastructure Stability:** The reliability of the infrastructure hosting the AI model, including servers, networks, and data centers.

**Software Stability**: The robustness of the software implementation, including error handling, redundancy, and failover mechanisms.

**Usage Patterns:** The demand on the system and the frequency of interactions with users.

## 4.5 Scalability

**Concurrent Users:** The number of users interacting with the model simultaneously can impact performance. Generally, the more concurrent users, the higher the workload on the servers.

**Complexity of Requests:** Some requests may require more computational resources than others. For example, generating longer texts or processing complex queries may increase the workload.

**Resource Allocation:** Adequate server resources, such as CPU, memory, and bandwidth, are essential for handling peak workloads. Insufficient resources can lead to performance degradation or even system failures.

22

**Load Balancing:** Distributing incoming requests across multiple servers can help balance the workload and prevent any single server from becoming overwhelmed.

**Scaling Strategies:** Implementing auto-scaling mechanisms allows the infrastructure to dynamically adjust resources based on demand, ensuring optimal performance during peak usage periods.

**Optimization:** Continuous optimization of algorithms, caching mechanisms, and other performance-enhancing techniques can improve the efficiency of the system under high workloads.

## 4.6 Maintainability

**Automated Builds:** Developers commit their code changes to a shared repository multiple times a day. Each commit triggers an automated build process, where the code is compiled and packaged into a deployable artifact (such as a binary or container image).

**Automated Tests**: As part of the CI pipeline, automated tests are run against the built artifacts to ensure that the changes haven't introduced regressions or bugs. These tests can include unit tests, integration tests, and even end-to-end tests depending on the complexity of the application.

**Immediate Feedback:** If the automated tests fail, developers receive immediate feedback about the issues introduced by their code changes. This encourages them to fix problems early in the development cycle, reducing the likelihood of bugs creeping into production.

23

**Continuous Deployment:** Once the code changes pass all tests in the CI pipeline, they are automatically deployed to a staging or production environment. Continuous deployment allows features and bug fixes to be released to users quickly and frequently.

**Rollback Mechanisms:** In case a deployment causes unexpected issues or errors in production, CI systems typically include rollback mechanisms that allow the

previous version of the software to be quickly restored. This minimizes downtime and mitigates the impact of failed deployments.
**Monitoring and Feedback Loop:** Continuous monitoring of the deployed applications helps identify performance issues, errors, and other anomalies in real-time. This feedback loop informs future development efforts and ensures that the software remains stable and reliable over time.
**Version Control Integration:** CI systems integrate seamlessly with version control systems like Git, allowing developers to track changes, collaborate effectively, and revert to previous versions if necessary.

## 4.7 Usability

To make software user-friendly:
**Intuitive User Interface (UI):** The user interface should be intuitive and easy to navigate. This involves organizing features logically, providing clear labels and instructions, and minimizing clutter.
**Simplified Onboarding:** A smooth onboarding process helps new users get started quickly. Tutorials, tooltips, and guided tours can assist users in understanding the core features and functionalities of the software.
**Consistent Design Patterns:** Consistency in design elements, such as color schemes, typography, and layout, helps users build familiarity with the software and reduces cognitive load.
**Accessibility**: Designing software with accessibility in mind ensures that users with disabilities can access and use the application effectively. This includes features like keyboard navigation, screen reader compatibility, and adjustable font sizes.
**Responsive Design:** In today's multi-device landscape, software should be responsive and adapt seamlessly to various screen sizes and resolutions, whether users access it from a desktop computer, tablet, or smartphone.
**Minimal User Input:** Minimize the need for manual data entry by automating repetitive tasks and leveraging smart defaults wherever possible. This reduces user effort and the likelihood of errors.
**Help and Support Resources:** Provide easily accessible help resources, such as documentation, FAQs, and user forums, to assist users in troubleshooting issues and learning how to use the software effectively.
**Feedback Mechanisms:** Incorporate feedback mechanisms, such as surveys, feedback forms, and support tickets, to gather user input and identify areas for improvement. Actively responding to user feedback demonstrates a commitment to user satisfaction and continuous improvement.

**Error Handling:** Implement clear and user-friendly error messages that provide actionable guidance on how to resolve issues. Avoid technical jargon and provide users with options to recover from errors gracefully.

**Regular Updates:** Regular updates and feature enhancements based on user feedback keep the software relevant and competitive in the market, while also addressing usability issues and addressing user needs over time.