Vision and Scope Document
for «la Recette»

Prepared by developer’s team

TABLE OF CONTENT

1. SYSTEM VISION

1.1. Vision Statement

«la Recette» software is aimed to provide information about the most suitable recipes gathered worldwide according to the customer data. It's designed to empower the users with flexible recipe choices based on ingredients available to them.

In addition, the software will allow end-customers to create individual dietary plans. It doesn't matter whether the user is a student with a limited budget, the chief cook, nutritionist or a regular householder, in all these cases he will benefit from using such a tool.

1.2. Major Features

The major features include:

1. **Personal account:**
   - Standard user information (login, password, email).
   - Body data storage (age, weight, height, sex, activity)

     Used to calculate calorie intake.
   - History of cooked recipes

     Required for providing non-repeating recipes in a certain period.
   - Product Blacklist

     Automatically hides recipes with unwanted ingredients.

2. **Recipe:**
   - Product (calorie content, proteins, fats, carbohydrates)

     Ingredient for recipes. Contains information for calculating the cost and nutritional value of a dish.

- Time

  Total cooking time.
- Difficulty

  Level of difficulty from 1 to 5.
- World cuisine

  Characteristic style of cooking practices and traditions, often associated with a specific region, country or culture.

3. **Ration:**
   - Combining recipes for some period, taking into account user restrictions (calorie, diet).
4. **Filters and search:**
   - Selection of recipes with specific ingredients:

     only these products or with additional ingredients.
   - Selection of recipes by criteria:

     time, calorie, difficulty, world cuisine.
   - Full Text Search.

1.3. Terms, abbreviation and definitions

Abbreviation	Definition
HTTP	HyperText Transfer Protocol
JSON	JavaScript Object Notation
GCP	Google Cloud Platform

1.4. Background

Perhaps, each householder was puzzled with a challenge of unique meal creation out of products he has. Surely we have also experienced these feelings. That’s why we’ve decided to create a universal helper which could propose the number of recipe choices based on our requirements.

In a process of refining the main product idea we decided to go 1 step further and add a possibility of dietary plans creation. The combination of a set of unique filters, smart search, and personalized approach is going to make our product unique in the market. Also, the wide range of products and recipes database which will be updated by our clients in real time guarantees discovery of unique recipes and diversification of dietary plans.

1.5. Document purpose

The main goal of writing this document is to define high-level needs and features of the «la Recette». It shows why the product should be developed and emphasizes the primary needs of clients and target users. Common details about work principles of the «la Recette» are described in the attachment.

2. SCOPE AND LIMITATIONS

In the sections that follow, the scope of this project will be defined in terms of major features, that will be implemented.

The main goal of this development cycle is to implement those features that satisfy our immediate needs.

2.1. Scope of Release

The major features that will be implemented in our project are:

1. Search for a recipe:
    - By keywords and description;
    - By selected criteria.
2. Account authorization:
    - Sign in;
    - Sign up;
    - Filling in personal information.
3. The opportunity to see a history of already cooked recipes.
4. The ability to create a list of prohibited products.
5. Selection of a suitable diet according to the entered parameters.

2.3. Limitations and Exclusions

This section describes the functionality of the application, in which stakeholders may be interested, but the implementation of these functionalities is not planned yet.

Due to the lack of time, there are several features that won’t be implemented by the time of the initial release:

- Localization.

- Chat for users.
- Blog with articles on the subject of our project.
- Block users violating terms of service.
- Recipe ratings.
- Recipe moderation.

3. BUSINESS CONTEXT

This section summarizes some of the business issues around the project.

3.1. Project Priorities

- Recipes search and selection by flexible system of filters.
- User account.
- Additional search filtration according to the user’s special needs and already cooked recipes.
- Ration, which is based on available recipes.

3.2. Operating Environment

This section describes the environment in which the system will be used and defines the basic availability, reliability, performance, and integrity requirements.

The important operating environment information is following:

1. The application is based on the microservice architecture, that enables the rapid, frequent and reliable delivery of large, complex applications.
2. The client side of the application will work in any operating system where the browser is installed.
3. The server side of the application will be embedded in cloud (Java + Tomcat) and connected to two databases: embedded in docker container Elasticsearch and Spanner from Google Cloud Platform.
4. All users will be in the same time zone with the one in which the server is located.
5. The application will be available at any time.
6. The output data will be generated based on client input data.

4. STACK OF TECHNOLOGIES

Our team will apply many technologies to achieve the project's goals:

- JRE 8
- Tomcat
- Spring stack
  - *Spring Framework 5*
  - *Spring Boot*
  - *Spring Cloud*
  - *Spring Security*
- Google Cloud Platform
  - *Compute*

    Compute Engine
  - *Databases*

    Cloud Spanner
- Elasticsearch
- Apache Kafka
- Docker
- Jenkins
- Gradle

JRE 8

The Java Runtime Environment (JRE) is a set of software tools for development of Java applications. It combines the Java Virtual Machine (JVM), platform core classes and supporting libraries.

Tomcat 9.0

Apache Tomcat is an open-source implementation of the Java Servlet, JavaServer Pages, Java Expression Language and WebSocket technologies. Tomcat provides a "pure Java" HTTP web server environment in which Java code can run. Tomcat is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation, released under the Apache License 2.0 license.

Spring Framework 5

The Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications - on any kind of deployment platform.

A key element of Spring is infrastructural support at the application level: Spring focuses on the "plumbing" of enterprise applications so that teams can focus on application-level business logic, without unnecessary ties to specific deployment environments.

Spring Boot

Spring Boot provides a good platform for Java developers to develop a stand-alone and production-grade spring application that you can just run. You can get started with minimum configurations without the need for an entire Spring configuration setup.

Spring Cloud

Spring Cloud provides tools for developers to quickly build some of the common patterns in distributed systems (e.g. configuration management, service discovery, circuit breakers, intelligent routing, micro-proxy, control bus, one-time tokens, global locks, leadership election, distributed sessions, cluster state).

Coordination of distributed systems leads to boiler plate patterns, and using Spring Cloud developers can quickly stand up services and applications that implement those patterns.

Spring Security

Spring Security is a powerful and highly customizable authentication and access-control framework. It is the de-facto standard for securing Spring-based applications.

Spring Security is a framework that focuses on providing both authentication and authorization to Java applications. Like all Spring projects, the real power of Spring Security is found in how easily it can be extended to meet custom requirements

GCP Compute engine

Compute Engine delivers configurable virtual machines running in Google's data centers with access to high-performance networking infrastructure and block storage.

GCP Cloud Spanner

Cloud Spanner is the first scalable, enterprise-grade, globally-distributed, and strongly consistent database service built for the cloud specifically to combine the benefits of relational database structure with non-relational horizontal scale.

Elasticsearch

Elasticsearch is a search engine based on the Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents.

Elasticsearch is developed in Java. Following an open-core business model, parts of the software are licensed under various open-source licenses (mostly the Apache License), while other parts fall under the proprietary (source-available) Elastic License.

Apache Kafka

Apache Kafka is a distributed streaming platform.

A streaming platform has three key capabilities:

- Publish and subscribe to streams of records, similar to a message queue or enterprise messaging system.
- Store streams of records in a fault-tolerant durable way.
- Process streams of records as they occur.

Kafka is generally used for two broad classes of applications:

- Building real-time streaming data pipelines that reliably get data between systems or applications.
- Building real-time streaming applications that transform or react to the streams of data.

Docker

Docker is a set of platform as a service (PaaS) products that use OS-level virtualization to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, libraries and configuration files; they can communicate with each other through well-defined channels. All containers are run by a single operating-system kernel and are thus more lightweight than virtual machines.

Gradle

Gradle is an open-source build-automation system that builds upon the concepts of Apache Ant and Apache Maven and introduces a Groovy-based domain-specific language (DSL) instead of the XML form used by Apache

Maven for declaring the project configuration. Gradle uses a directed acyclic graph ("DAG") to determine the order in which tasks can be run.

Gradle was designed for multi-project builds, which can grow to be quite large. It supports incremental builds by intelligently determining which parts of the build tree are up to date; any task dependent only on those parts does not need to be re-executed.

Jenkins

Jenkins is a self-contained, open source automation server which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

5. NON-FUNCTIONAL REQUIREMENTS

Non-functional Requirements define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs.

- The response time must not take more than 1,5 seconds.
- The application is available at any time.
- Privacy of information, the export of restricted technologies, intellectual property rights, etc. should be audited.
- All sensitive data should be transferred in an encrypted way.
- Each request from client to server must contain all of the information necessary to understand the request, and cannot take advantage of any stored context on the server.
- Requests(messages) shall be self-descriptive.