A Software requirements specification

On

**Application for Nutrition Assistant**

Submitted in Partial Fulfillment for the Award of Degree of Bachelor of Technology in Information Technology from Rajasthan Technical University, Kota



**Project Guide: Prepared By:**

**Dr. Sunita Gupta Devanshu Soni (19ESKIT302)**

(Associate Professor) **Rishabh Panchal (19ESKIT078)**

**Ritwik Sharma (19ESKIT301)**

**Ritik Chourasiya (19ESKIT081)**

**Submitted To:**

**Ms. Sanju Choudhary**

(Associate Professor)

DEPARTMENT OF INFORMATION TECHNOLOGY

**SWAMI KESHWANAND INSTITUTE OF TECHNOLOGY, MANAGEMENT & GRAMOTHAN**

**Ramnagaria (Jagatpura), Jaipur – 302017**

**SESSION 2022-23**

**Table of Contents**

1. **Introduction……………………………………………………………………. 1**
2. Purpose…………………………………………………………………….... 1
3. Project Scope………………………………………………………………... 1
4. Technologies and Tools Used………………………………………………. 1
5. Overview……………………………………………………………………. 6
6. **Overall Description……………………………………………………………. 7**
   1. Product Perspective…………………………………………………………. 7
   2. Product Function……………………………………………………………. 7
   3. User Classes and Characteristics……………………………………………. 7
   4. Operating Environment……………………………………………………... 8
   5. Design and Implementation Constraints……………………………………. 8
   6. User Documentation……………………………………………………….. 10
   7. Assumption and Dependencies…………………………………………….. 10
7. **External Interface Requirements…………………………………………….. 11**
   1. User Interfaces……………………………………………………………… 11
   2. Software Interfaces…………………………………………………………. 11
   3. Communication Interfaces………………………………………………….. 12
8. **System Features……………………………………………………………….. 13**
   1. System Feature 1……………………………………………………………. 13
   2. System Feature 2……………………………………………………………. 13
9. **Other Non-Functional Requirements………………………………………... 15**
   1. Performance Requirements…………………………………………………. 15
   2. Safety Requirements………………………………………………………... 15
   3. Security Requirements…………………………………………………….... 15
   4. Software Quality Attributes……………………………………………….... 16
10. **UML Diagrams………………………………………………………………... 17**

6.1 E-R Diagram………………………………………………………………... 17

6.2 Use-Case Diagram………………………………………………………….. 18

6.3 Class Diagram……………………………………………………………..... 19

6.4 Activity Diagram…………………………………………………………..... 20

6.5 Data Flow Diagram……………………………………………………….... 21

**References……………………………………………………………………... 22**

**List of Figures**

* 1. E-R Diagram
  2. Use Case Diagram
  3. Class Diagram
  4. Activity Diagram
  5. Data Flow Diagram

1. **Introduction**
   1. **Purpose**

Due to the ignorance of healthy food habits, obesity rates are increasing at an alarming speed, and this is reflective of the risks to people’s health. People need to control their daily calorie intake by eating healthier foods, which is the most basic method to avoid obesity. However, although food packaging comes with nutrition (and calorie) labels, it’s still not very convenient for people to refer to App-based nutrient dashboard systems which can analyze real-time images of a meal and analyze it for nutritional content which can be very handy and improves the dietary habits, and therefore, helps in maintaining a healthy lifestyle.

* 1. **Project Scope**

This project aims at building a web App that automatically estimates food attributes such as ingredients and nutritional value by classifying the input image of food. Our method employs Clarita’s AI-Driven Food Detection Model for accurate food identification and Food APIs to give the nutritional value of the identified food. This application will be accessible in all web browsers and can be use by any user to check the food attributes. The main aim of this application is to provide nutrients value in food in order to live a healthy life.

* 1. **Technologies and Tools Used**

1. **Streamlit**

* Streamlit is a Python-based library that allows data scientists to easily create free machine learning applications. Streamlit allows you to display descriptive text and model outputs, visualize data and model performance and modify model inputs through the UI using sidebars.
* Streamlit is the easiest way especially for people with no front-end knowledge to put their code into a web application:
* No front-end (html, js, css) experience or knowledge is required.
* You don't need to spend days or months to create a web app, you can create a really beautiful machine learning or data science app in only a few hours or even minutes.
* It is compatible with the majority of Python libraries (e.g. pandas, matplotlib, seaborn, plotly, Keras, PyTorch, SymPy(latex)).
* Less code is needed to create amazing web apps.
* Data caching simplifies and speeds up computation pipelines.
* Streamlit is designed to streamline the process of building interactive data-driven applications. It allows users to create a web application using simple Python scripts that can be updated in real-time, making it easy to iterate on an application and quickly test new ideas.
* Another key feature of Streamlit is its ability to integrate with popular Python libraries such as Pandas, Numpy, and Scikit-learn. This allows users to leverage the power of these libraries to analyze and manipulate data, create machine learning models, and generate visualizations.
* Streamlit is a powerful and user-friendly library that simplifies the process of building web applications for machine learning and data science projects. Its easy-to-use interface, real-time updates, built-in widgets, and integration with popular Python libraries make it a popular choice for data scientists and developers alike.
* Streamlit provides a wide range of customization options that allow developers to create custom themes, layouts, and styles to match their branding or project requirements.
* Streamlit allows multiple users to work collaboratively on a project, making it easy to share code and work together on a single application.
* Streamlit provides built-in tools and integrations with popular machine learning libraries such as Scikit-learn and TensorFlow, making it easy to create and deploy machine learning models.

1. **Flask**

* Flask is a web application framework written in Python. It is developed by **Armin Ronacher**, who leads an international group of Python enthusiasts named Pocco. Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.
* A Web Application Framework or a simply a Web Framework represents a collection of libraries and modules that enable web application developers to write applications without worrying about low-level details such as protocol, thread management, and so on.
* Flask uses URL routing to map HTTP requests to specific Python functions, making it easy to create a URL structure for a web application.
* It is an acronym for web server gateway interface which is a standard for python web application development. It is considered as the specification for the universal interface between the web server and web application.
* Flask provides configuration and conventions, with sensible defaults, to get started. This section of the documentation explains the different parts of the Flask framework and how they can be used, customized, and extended. Beyond Flask itself, look for community-maintained extensions to add even more functionality.

1. **Pandas**

* Python Pandas is defined as an open-source library that provides high-performance data manipulation in Python.
* Pandas is defined as an open-source library that provides high-performance data manipulation in Python. The name of Pandas is derived from the word **Panel Data**, which means **an Econometrics from Multidimensional data**.
* Data analysis requires lots of processing, such as **restructuring, cleaning** or **merging**, etc. There are different tools are available for fast data processing, such as **Numpy, Scipy, Cython**, and **Panda**. But we prefer Pandas because working with Pandas is fast, simple and more expressive than other tools.
* Pandas is built on top of the **Numpy** package, means **Numpy** is required for operating the Pandas.
* Handle multiple operations of the data sets such as subsetting, slicing, filtering, groupBy, re-ordering, and re-shaping.
* Process a variety of data sets in different formats like matrix data, tabular heterogeneous, time series.

1. **Kaggle Dataset**

* Kaggle is one of the best sources for providing datasets for Data Scientists and Machine Learners. It allows users to find, download, and publish datasets in an easy way. It also provides the opportunity to work with other machine learning engineers and solve difficult Data Science related tasks.
* Kaggle provides a high-quality dataset in different formats that we can easily find and download.
* Kaggle provides powerful resources on cloud and allows you to use a maximum of 30 hours of GPU and 20 hours of TPU per week.
* Kaggle is suitable for different groups of people, from students interested in data science and [**artificial intelligence**](https://www.datacamp.com/learn/ai) to the most experienced data scientists in the world.
* In data science competitions like Kaggle’s or **[DataCamp](https://www.datacamp.com/data-science-competitions" \t "_blank)**’s, companies and organizations share a big amount of challenging data science tasks with generous rewards in which data scientists, from beginners to experienced, compete on their completion.
* Kaggle also provides the Kaggle Notebook, which, just like **[DataCamp Workspace](https://www.datacamp.com/workspace" \t "_blank)**, allows you to edit and run your code for data science tasks on your browser, so your local computer doesn't have to do all the heavy lifting and you don't need to set up a new development environment on your own.
* You can upload your datasets to Kaggle and download others' datasets as well. Additionally, you can check other people's datasets and notebooks and start discussion topics on them.
* Last but not least, when applying for jobs in data science, mentioning your Kaggle experience definitely makes a positive impact. It goes without saying that all these benefits also apply to highly experienced data scientists.

1. **VS Code**

* Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging.
* First and foremost, it is an editor that gets out of your way. The delightfully frictionless edit-build- debug cycle means less time fiddling with your environment, and more time executing on your ideas.
* Visual Studio Code is a free source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.
* Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including [Java,](https://en.wikipedia.org/wiki/Java_(programming_language)) [JavaScript,](https://en.wikipedia.org/wiki/JavaScript) [Go](https://en.wikipedia.org/wiki/Go_(programming_language)), [Node.js](https://en.wikipedia.org/wiki/Node.js) and [C++.](https://en.wikipedia.org/wiki/C%2B%2B) Here we can change the [theme,](https://en.wikipedia.org/wiki/Theme_(computing)) [keyboard shortcuts,](https://en.wikipedia.org/wiki/Keyboard_shortcut) preferences, and install extensions that add additional functionality.
* Visual Studio Code's source code comes from Microsoft's free and open-source Software **VSCode** project released under the permissive Expat License, and the compiled binaries are freeware for any use.
* Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language. It supports several programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface, but can be accessed via the command palette.
* Visual Studio Code can be extended via extensions available through a central repository. This includes additions to the editor and language support A notable feature is the ability to create extensions that add support for new languages, themes, and debuggers, perform static code analysis and using the Language Server Protocol

1. **TensorFlow**

* TensorFlow is a free and open-source software library for machine learning and artificial intelligence.
* It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.
* Develop ML models in JavaScript, and use ML directly in the browser or in Node.js
* The TensorFlow platform helps you implement best practices for data automation, model tracking, performance monitoring, and model retraining.
* TensorFlow offers better visualization, which allows developers to debug better and track the training process

1. **Python**

* Python is a high-level, general-purpose programming language.
* It supports multiple programming paradigms, including structured, object-oriented and functional programming
* Mature frameworks and development tools
* Clean Syntax and easy to read code
* Advanced libraries for deep learning, machine leaning, artificial intelligence
* Easy to Debug
* There are several libraries and frameworks in Python that can be used for machine learning
* Machine learning involves a computer to be trained using a given data set, and use this

training to predict the properties of a given new data.

* 1. **Overview**

By using Streamlit, Flask, Python and Kaggle Dataset a fully featured web application is created. Application uses signup by filling signup data, after filling all the data the application validate the data and it will be redirected to the home page. In the home page we can view services, we can see article of health, Reviews and likes. Users can also upload his image and it will be stored in Kaggle Dataset storage. User can also update his profile.

1. **Overall Description**
   1. **Product Perspective**

A Nutrition Assistant application database system stores the following information.

* User details:

The database consists of the username, password, contact, email id. These details provide valid login to the user after getting registered.

* Item details:

The DB consists of the description the items users have selected along with image and also it stores whether an item belongs to any type of food.

* 1. **Product Function**
* Application uses signup by filling signup data. After validation it will be redirected to the home page. In the home page we can view various categories of services. Application allows the user to upload the image and it will be stored in KaggleDB storage.
* Users also update their profile by adding extra details such as Bio, location, personal and professional website information. Once this information are submitted then they are validated.
* Users can choose their favorite products. They get two options i.e., either they can add the product directly to the Cart or they can add it to their Directory.
* Maintenance of details of various users.
  1. **User Classes and Characteristics**

The user can search for food nutrition’s of their choice once they login to the application. Also once they have gone through a few searches then the algorithm shows them more similar kinds of food. The Users should be able to do the following functions:

* Upload the image or profile that will be stored in Firebase storage.
* User can add their food to their directory.
* User can view his total intake of nutrients to date or between any specific dates.
* Users can provide feedback for services.
  1. **Operation Environment**

The Operating Environment used in the project is Windows

* 1. **Design and Implementation Constraints**

The project operates under a number of design and implementation constraints. Some of these are as outlined below:

**Hardware and Software Constraints:**

Since the project has been developed entirely using Streamlit ,python, flask and Kaggle DB for frontend and backend, it is largely independent. The project can be run on any platform.

* There is the secondary memory, which is where your files are stored, and a computer can use a Hard Disk to store memory, or a Solid-State Drive (As well as other kinds of Flash Memory). Modern Hard Disks (HDs) can store from around 500 GB to 2TB of data.
* There is also primary memory, which is the memory that stores information that you are manipulating with immediately, when the computer is ON.
* To better understand the difference between primary and secondary, let me give you an example: Suppose you want to edit a photo. Your photo is stored in your Secondary Memory. When you open up your photo in a photo editor, it is loaded on your primary memory, so that you manipulate it. Then, you apply your favorite vintage filter to it, as well as colors and whatever you want. The edited photo is still on your primary memory.

Then, you are happy with the result, so you click on "Save", that's when your new version of the photo is copied back to your secondary memory.

* Computers use CPU Memory registers, RAM (Random Access Memory), and other resources as Primary Memory. A modern laptop has around 8GB of RAM.
* Another big difference between Primary and Secondary Memory is that Primary memory is volatile. That means if your battery dies while your editing your photo, you'll lose all of your changes, because the Primary Memory can only keep information as long as there is energy. Secondary Memory, on the other hand, can keep your photo for years, even if the computer is OFF.
* An interesting thing I'd like to point up is Swap Space. Let's suppose you have 25 tabs of

YouTube videos open on your browser. All of those videos are being downloaded into your RAM memory. Let's suppose you have 2GB, and then all of the videos combined occupy 3GB of memory. You don't have enough memory to hold all of that! When the computer reaches it's RAM limit, it starts using "Swap Space", which is basically storing the excessive data into a temporary location on your HD. HDs are too slow to hold data we are momentarily using, that's why your computer seems sluggish after your open a lot of pages and apps.

* Another place full of constraints in hardware is the CPU (Central Processing Unit). It is where everything in the computer is calculated. Every core of a CPU can only do one operation at a time. Think of operation as something like 1- Get two numbers from memory, 2- Sum two numbers, 3- Show numbers on screen. It has a stack, which you can think of as a line of all the operations wanting to get in. It works with cycles, and with each cycle, one simple operation is done. The CPU Clock is a little device that sends electrical pulses to the CPU, and each pulse, a cycle.
* It seems quite straight forward then: Make the clock tick faster, so that we have more cycles per second, and work on all of the operations on the stack faster. There is a problem, however: You run the risk of overheating your processor, potentially melting down components and having a big headache. Many hardware hackers experiment with this, called "Over-clocking", and they generally use more expensive cooling systems, like liquid cooling, faster fans, bigger heat dissipaters, etc.
* An approach that has been widely used on the industry is to increase the number of "Cores" on a processor. You probably heard the terms dual-core, quad-core somewhere, since now even smartphones can be quad-core (four cores). So, that way, you multiply your processing power. That doesn't come without obstacles, of course. You need to carefully craft your software to work well with Parallel computing (AKA The art of using multiple processing units).
* One of the biggest challenges for computing today is the size of the transistor, a fundamental component of a processor. Transistors are getting so small, that Quantum Effects are starting to be relevant, as they will play out bigger interference with the functioning of the component. That's a very important constraint for today's industry.
* I'd like to briefly talk about I/O (Input/Output). I/O is basically communication across devices (AKA the thing behind USB stuff). Suppose you have just bought a super cool webcam, with 20 megapixels, 60 frames per second. Of course, the amount of data it generates is large. In that same scenario, your computer is old, and the USB port can only transport 10 frames per second of your beautiful camera. You have a problem there. You won't be able to enjoy 60fps Skype Calls (and remember there are internet constraints as well, to slow it even further). Your computer's I/O is slow, that's a constraint, and even a 4GHZ, Octa-Core Processor, or a 50GB of RAM (I'm exaggerating values here) wouldn't solve that issue.
* As you can see, there can be many constraints coming from many places in a computer. Many of the design decisions that have to be taken will have to take into account many trade-offs of speed, capacity, price, compatibility, etc. You can think of it as a huge equation with thousands, millions of variables, and your job is to find the optimal value for each of those variables.

**End User Constraints:**

The major constraints for the end user are having a proper internet connection on his computer or mobile. Also he should be familiar with the operation of the application to a certain extent.

* 1. **User Documentation**

User manual and GitHub link will be available for troubleshooting and help. The user manual will contain detailed information about the usage of the application from a layman’s perspective to an expert network/system administrator. The manual shall also be made available online.

* 1. **Assumptions and Dependencies**

The proposed solution will be designed to work in an enterprise environment. The target environment may consist of wired and wireless links inside the network. The solution has to be self-sufficient and free from any unfamiliar dependencies.

1. **External Interface Requirements**
   1. **User Interfaces**

We have tried to keep the User interface as simple as possible so that users can use the application without putting much of their efforts. Like any other Food Nutrition Assistant Application this application also shows a variety of categories to the user once they fall on the homepage of the application.

The project contains different kinds of services. Services like to search Nutrition in food by providing images of food. Admin can add new products into the existing system with all its details including an image.

* 1. **Software Interfaces**

Following is the software used for the flight management online application.

|  |  |
| --- | --- |
| **Software used** | **Description** |
| Operating system | We have chosen a Window operating system for its best support and user-friendliness. |
| Database | To save the food nutrition which have been searched by the users and to store user information |
| Tools/IDE | To implement the project we have chosen Visual Studio for its more interactive support. |
| Platform | Web Application |
| Technologies and Tools Used | Streamlit, Flask, Python, KaggleDB, etc. |

* 1. **Communications Interfaces**

As the application is a Food Nutrition Assistant application so it requires high speed internet modem for the use of this application. For the suitable use there must be a correct internet connection among the users.

The users can directly interact with the application and use its functionalities.

1. **System Features**
   1. **System Feature 1 - User Registration**
      1. **Description and Priority**

When the application is browsed for the very first time the user is presented with an initial registration/welcome screen. This screen prompts the user to create an account on the Nutrition

Assistance Application using the email address and username associated with their account. The priority of this feature is highest because completing this process will create and store an account for the user on the food nutrition assistant application, enabling all of the application’s synchronization capabilities.

* + 1. **Stimulus/Response Sequence**
* Step 1: The user is prompted to enter an email address and his/her name. If all the information is valid then it will be redirected to the home page.
* Step 2: This information is stored in the Firebase database.
* Step 3: Registration is completed and the user is taken to the home page.
* Step 4: In the nutrition check, the user can upload the image.
  + 1. **Functional Requirements**

The application will ask for a valid email address, username, and password at the time of registration. If your details are not valid then registration will not be successful.

* 1. **System Feature 2**
     1. **Description and Priority**

The next thing that has to be done after registration and login is post the new profile or may like existing profiles.

* + 1. **Stimulus/Response Sequence**
* Step 1: After login you will be able to see all the services that are posted by the admin.
* Step 2: Now you must select any one / more of the services and if you like to search nutrient values then there is an option to add it to your personal directory.
* Step 3: You can also see total nutrition intake between specific date in your track activity.
* Step 4: Lastly the application will also provide a logout button if you want to exit from the application.
* Step 5: It also provides a BMI calculator , Weight Gain calculator to help users track their weight.
* Step 6: Press the logout button.
  + 1. **Functional Requirements**

When you create an application, it’s not all about the product range, pricing, and service, as it may seem. Your web application platform should satisfy and attract buyers both visually and in terms of functionality traits. Let’s talk about the most relevant functional requirements for the website up to date.

Formally, all functional requirements can be subdivided into two broad categories: those that allow the user to enjoy the site at the stage of product selection and those that “push” them to make a purchase.

The first category includes the development of search tools, sorting, filtering, navigation, as well as the visual component of the site.

The second category includes the study of user interactions, the application form for the return of goods, all kinds of elements of social proof, etc.

Generally speaking, UX requirements for web application sites can be very different depending on the region a particular site targets. If we are talking about the international food nutrition website scope, such sites should necessarily have several localizations and comply with regional shipping restrictions.

Comparing functional web store requirements, there are certainly more and less important points among them. Of utter importance are global things such as several localizations, integration with payment systems, the ability to make purchases without registration, sped up page loading, etc.

1. **Other Non-Functional Requirements**
   1. **Performance Requirements**

* **Normalized data-** data redundancy should be minimal which in turn reduces the chances of insert, delete and update anomalies.
* **Response time-** the overall time beginning with the user action (click on sign up button after filling up the details), the request going to the server, the response received from the server, and finally the response processing by the application will not take more than 10 seconds.
* **Scalability-** according to intended number of users and the projected load scenarios, the system should be able to serve 50 queries / day (in large part during the peak hours).
  1. **Safety Requirements**

Some securities measures are provided to the application account holders such as account holder must give his/her account id and password to login. Other than that security to user’s personnel details and photos galleries.

If there is extensive damage to a wide portion of database due to catastrophic failure, like disk crash, recovery method restores a past copy of the database that was backed up to archival storage (typically tape) and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed-up log, up to the time of failure.

* 1. **Security Requirements**
* If a user tries to log into a non-existing account, then the user should not be logged in.
* User login account security-the User should not be able to login if the email id, username or password is incorrect.
* Users create account security-if a user wants to create an account and the desired name is occupied, the user should be asked to enter a different name.
  1. **Software Quality Attributes**
* **Availability-** All the services should be available to the user.
* **Correctness-** The list of the products related to a user should be stored correctly.
* **Usability-** The details of products should be self-explanatory.
* **Maintainability-** User should maintain the database and store in updated form.
* **Portability-** The application should be portable to mobile.
* **Reliability-** The system should give 98% correct search results out of 1000 searches during testing.
* **Extendibility-** The application should be easy to extend, code should be written in such a way that it favor’s implementation of new functions.

**6. UML Diagrams**

**6.1 E-R Diagram**

This entity relationship diagram represents the model of Food Nutrition Assistant entity. The entity relationship diagram of Food Nutrition Assistant shows all the information about the data and relationship between users’ food id and photo with its nutrition’s value.

The main entities of Food Nutrition Assistant are food id, food photo, user id, nutrition.

* User attributes: User Id, Login Name, Email Id, mob.no.
* Food name: Photo, Description, Nutrient, Food Id , Attributes.
* After successful login users can perform user activity and use the functionalities of the application.
* Database manages all the products.

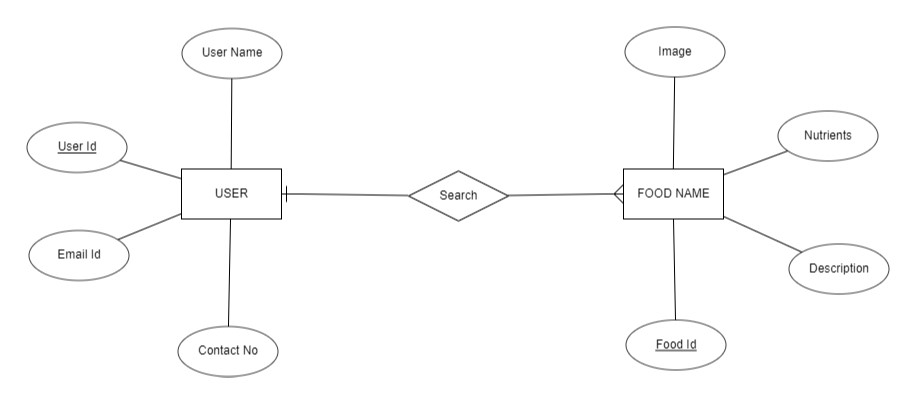


Fig.1.1 E-R Diagram

**6.2 Use-Case Diagram**

* This diagram describes a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable results to the actors or other stakeholders of the system.
* This use-case diagram is a graphic depiction of the interaction among the elements of the Nutrition Assistance Application. It represents the methodology used in system analysis to identify, clarify and organize system requirements of Nutrition Assistance Application.
* The main actors in the Nutrition Assistance Application are the admin and user.

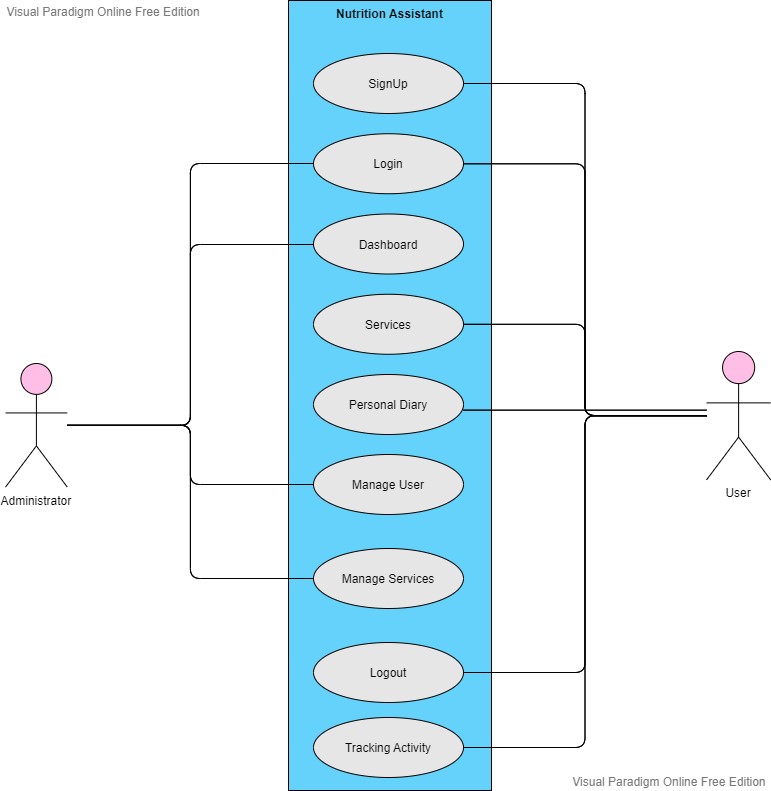


Fig.1.2 Use Case Diagram

* In Nutrition Assistance Applications, users are the actors who can interact with the frontend of the application and MongoDB handles the database (back-end) and Express Js is used for the communication between frontend and backend.

**6.3 Class Diagram**

* A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.
* Classes of social media application are:

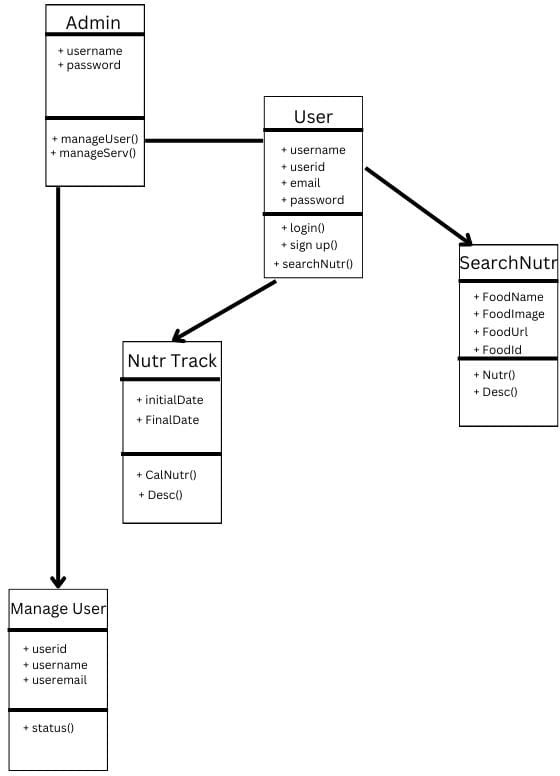
1. User’s class: Manage all the operations of the user.
2. Admin class: Manage all the services and users.
3. Search Nutrition’s class: Manages details about all the nutrients present in food items.
4. Nutr Track Class: Track all the details about nutrients within given dates.

Fig.1.3 Class Diagram

**6.4 Activity Diagram**

* This diagram describes the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another.
* This is the Activity UML diagram of a Shopping application which shows the flow between the activity of users and application.
* The main activity involved in this UML Activity diagram of social media application are as follows:-

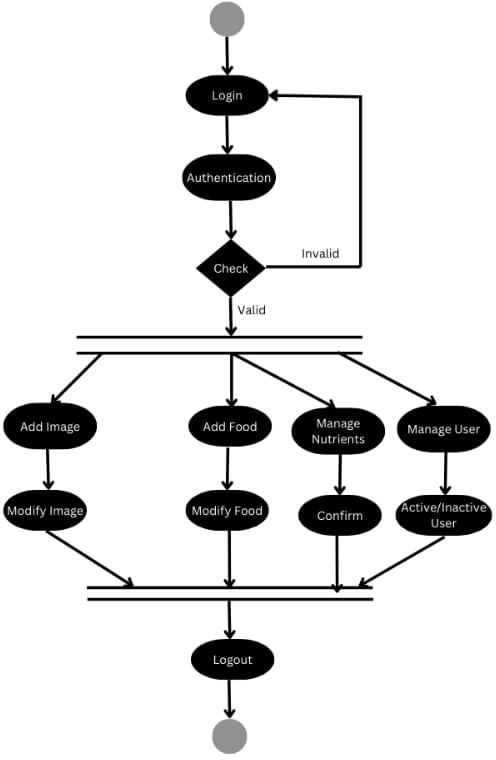
1. User activity
2. Tracking activity
3. Personal diary activity

Fig.1.4 Activity Diagram

**6.5 Data Flow Diagram**

* A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.
* Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one.
* Like all the best diagrams and charts, a DFD can often visually “say” things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That’s why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems.

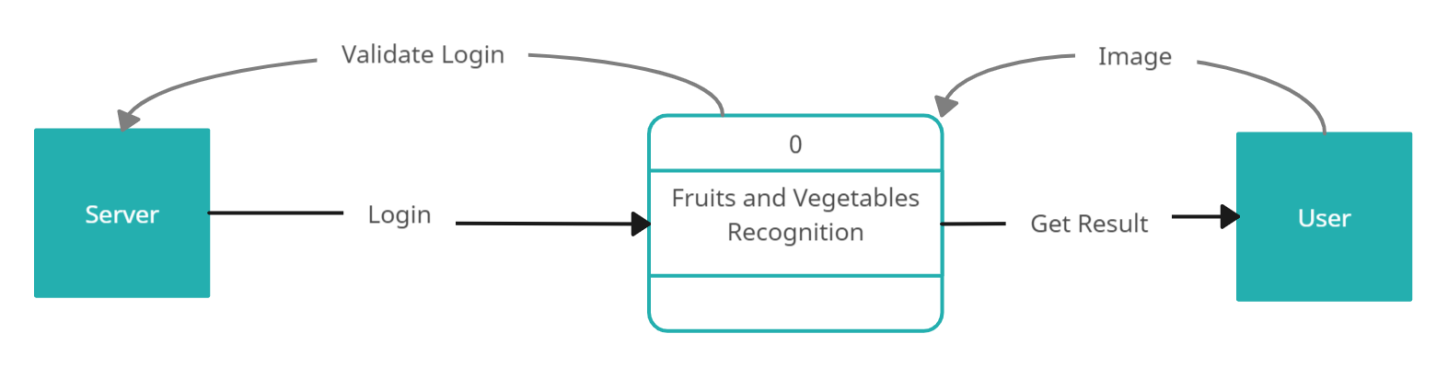


Fig. 1.5 Data Flow Diagram

**References**

* Object Oriented Modeling and Design with UML-Michael Blaha, James Rambaugh.
* Software Engineering, Seventh Edition, Ian Sommerville.
* Wikipedia - [*www.wikipedia.com*](http://www.wikipedia.com)
* Streamlit*-* [*https://docs.streamlit.io/*](https://docs.streamlit.io/)
* Python*-* [*https://machine-learning-with-python.readthedocs.io/en/latest/*](https://machine-learning-with-python.readthedocs.io/en/latest/)
* Flask *-* [*https://flask.palletsprojects.com/en/2.3.x/tutorial/*](https://flask.palletsprojects.com/en/2.3.x/tutorial/)
* Kaggle Dataset *-* [*https://www.kaggle.com/docs/datasets*](https://www.kaggle.com/docs/datasets)