Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	15/10/2022
Team ID	PNT2022TMID23685
Project Name	Project – Nutrition Assistant Application
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

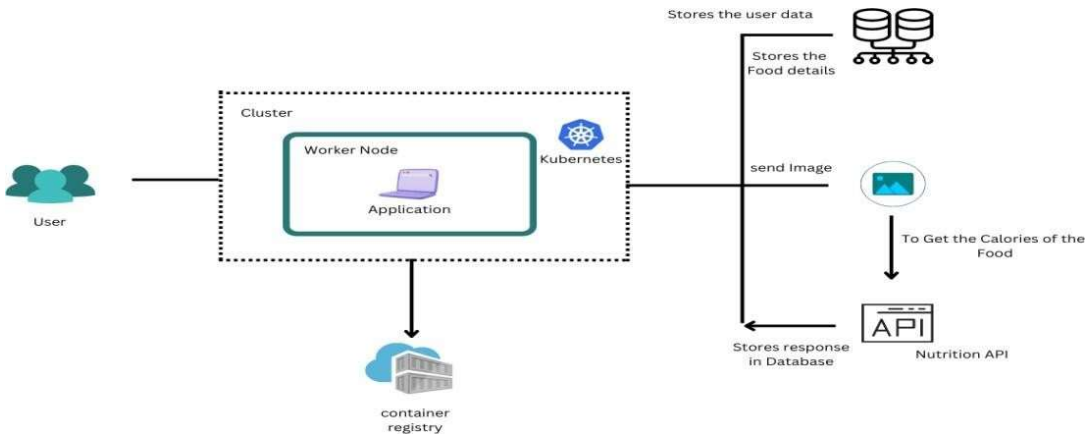

Table-1: Components & Technologies:

S.NO	Component	Description	Technology
1.	User Interface	Web UI	HTML, CSS, JavaScript
2.	To get the food nutrition and calorie value	The user will upload the food picture. Then the user will see the food nutrition value the process will compute	Python, Flask (web Framework), HTML, CSS, JavaScript.
3.	Database	Get the user's name, mail and stores the food calories value. Data types: integer, string, Float Number and etc.,	MySQL or PostgreSQL
4.	Cloud Deployment	Through is the application Will compose to the internet	Kubernetes, Docker
5.	External API-1	To predict the image that user will upload in the upload image page	Clarifai's AI-driven Food detection Model API
6.	External API-2	Food API's for to the nutritional value for the identified food	Food API
7.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration:	Local, Cloud Foundry, Kubernetes, etc. Docker.

Table-2: Application Characteristics:

S.NO	Characteristics	Description	Technology
1.	Open-Source Frameworks	We are using both front and back end here to runs the web application.	Flask (Microweb framework) Vue.js
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g., SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Presentation tier- HTML/ CSS/ JavaScript Application tier- Python (API) Data tier- MySQL, PostgreSQL
4.	Availability	Justify the availability of application (e.g., use of load balancers, distributed servers etc.)	working to reduce the severity and likelihood of problems, closely monitoring applications and infrastructure, keeping technical debt in check, automating recovering mechanisms, and regularly putting those recovery mechanisms to the test.
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Optimize image sizes, use a content delivery network, use website caching and adopt cloud-based website monitoring.