**Technology Stack**

|  |  |
| --- | --- |
| Date | 31 January 3035 |
| Team ID | LTVIP2025TMID59149 |
| Project Name | A Collage Food Choices Case Study |
| Maximum Marks | 4 Marks |

## **Technology Stack for Food Choices Application**

### **🧩 1. Frontend (User Interface)**

Used for building interactive user-facing apps:

* **Web App:**
  + **React.js** or **Vue.js** – Component-based UI, fast rendering
  + **Tailwind CSS** or **Material UI** – For modern, responsive design
* **Mobile App:**
  + **React Native** – Cross-platform for iOS & Android
  + **Flutter** – Another cross-platform choice with great UI

### **🧠 2. Backend (Server-side Logic)**

Used for managing logic, APIs, and database interaction:

* **Node.js** + **Express.js** – Lightweight and scalable
* **Python** + **Django/Flask** – Great if you're doing data analytics or ML
* **Ruby on Rails** – For faster MVP development

### **📦 3. Database**

To store user data, food items, nutritional info, etc.:

* **PostgreSQL** – Relational DB for structured data (nutritional tables, user accounts)
* **MongoDB** – NoSQL, great for flexible schemas like meal logs, food photos
* **Firebase Firestore** – Real-time database for mobile apps

### **🌍 4. APIs / Integrations**

Enhance features by integrating external data sources:

* **Nutritionix API**, **Edamam API** – For food and nutritional data
* **Open Food Facts API** – For barcode scanning and ingredient info
* **Google Maps API** – To show nearby healthy food options
* **AI/ML model** – For recommending meals or analyzing dietary habits

### **☁️ 5. Cloud / Hosting**

For deploying your backend, database, and web frontend:

* **Heroku** – Quick deployment (good for MVPs)
* **Vercel / Netlify** – Frontend hosting (great with React/Vue)
* **AWS / Google Cloud / Azure** – Scalable infrastructure with storage, compute, and AI services

### **🛡️ 6. Authentication & Security**

For user login, data protection, etc.:

* **Firebase Authentication** – Easy email/social login
* **OAuth 2.0** – For third-party sign-ins (Google, Apple)
* **JWT (JSON Web Tokens)** – Secure session handling

### **📊 7. Analytics & User Feedback**

To track user behavior and improve UX:

* **Google Analytics / Mixpanel** – User behavior tracking
* **Hotjar / FullStory** – Session replays and heatmaps
* **Sentry / Log Rocket** – Error logging and debugging

**8. Optional: AI/ML Integration**

If your app includes personalization or dietary analysis:

* **TensorFlow.js / PyTorch** – For running models
* **OpenAI GPT** – For recipe suggestions, chatbot integration
* **Scikit-learn / pandas / NumPy** – For analyzing user food patterns

**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** |
|  | User Interface | How user interacts with application e.g.  Web UI, Mobile App, Chatbot etc. | HTML, CSS, JavaScript / Angular Js / React Js etc. |
|  | Application Logic-1 | Logic for a process in the application | Java / Python |
|  | Application Logic-2 | Logic for a process in the application | IBM Watson STT service |
|  | Application Logic-3 | Logic for a process in the application | IBM Watson Assistant |
|  | Database | Data Type, Configurations etc. | MySQL, NoSQL, etc. |
|  | Cloud Database | Database Service on Cloud | IBM DB2, IBM Cloudant etc. |
|  | File Storage | File storage requirements | Local Filesystem |
|  | External API-1 | Purpose of External API used in the application | IBM Weather API, etc. |
|  | External API-2 | Purpose of External API used in the application | Aadhar API, etc. |
|  | Machine Learning Model | Purpose of Machine Learning Model | Object Recognition Model, etc. |
|  | Infrastructure (Server / Cloud) | Application Deployment on Local System / Cloud  Local Server Configuration:  Cloud Server Configuration : | Local, Cloud Foundry, Kubernetes, etc. |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
|  | Open-Source Frameworks | List the open-source frameworks used | Technology of Opensource framework |
|  | Security Implementations | List all the security / access controls implemented, use of firewalls etc. | e.g. SHA-256, Encryptions, IAM Controls, OWASP etc. |
|  | Scalable Architecture | Justify the scalability of architecture (3 – tier, Micro-services) | Technology used |
|  | Availability | Justify the availability of application (e.g. use of load balancers, distributed servers etc.) | Technology used |
|  | Performance | Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN’s) etc. | Technology used |