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

Date	30 October 2022
Team ID	NM2023TMID03207
Project Name	Indian Food EDA
Maximum Marks	4 Marks

Technical Architecture:

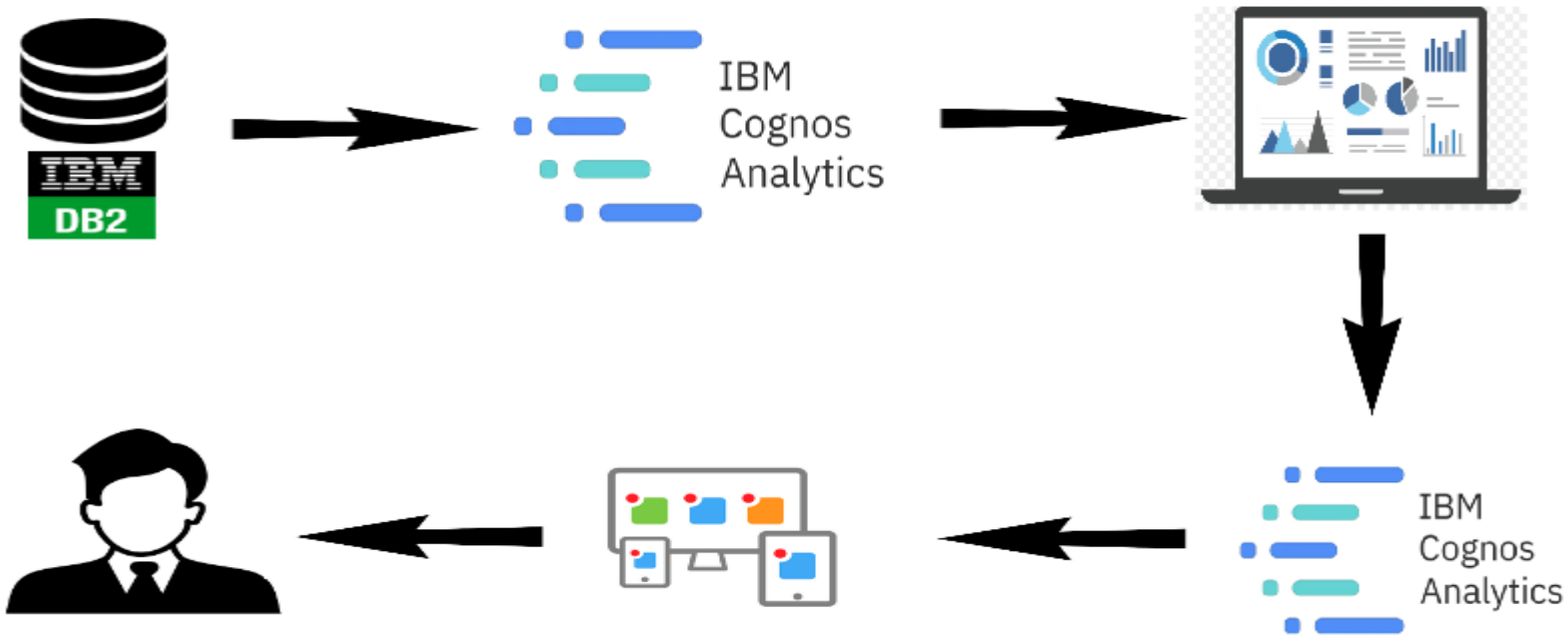

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Data Processing and Analysis	Responsible for handling HTTP requests, processing data, and serving responses	Python
3.	Voice-Based Data Collection	This service enables the collection of voice-based data from users.	IBM Watson STT service
4.	Chatbot Interface	An intelligent conversational agent designed to enhance the user experience and provide valuable insights into the Indian food domain.	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL
6.	Cloud Database	Database Service on Cloud	IBM Cloudant.
7.	File Storage	File storage requirements	IBM Block Storage or Local Filesystem
8.	IBM Weather API	Provide real-time weather data and historical weather trends for various regions across India	IBM Weather API
9.	Aadhar API Service	Integrated to verify and authenticate user identities using Aadhar numbers .	Aadhar API.
10.	Object Recognition Model	An image analysis component that automatically identifies and categorizes Indian food items and ingredients.	Object Recognition Model
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Kubernetes

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Technology of Opensource framework
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: - Frontend: React.js or Angular - Web Server: Nginx or Apache Application Tier: - Microservices: Node.js, Spring Boot, or Django - Message Queue: Apache Kafka or RabbitMQ - Container Orchestration: Kubernetes Data Tier: - Relational Database: PostgreSQL or MySQL - NoSQL Database: MongoDB or Cassandra
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	1. Load Balancers - AWS ELB 2. Distributed Servers/Clusters-Kubernetes 3. Database Replication - MySQL Replication

			4. Content Delivery Networks (CDNs) - Amazon CloudFront 5. Auto-Scaling-AWS Auto Scaling 6. Monitoring and Alerting-Prometheus, Grafana 7. Disaster Recovery 8. High Availability Databases -MySQL Group Replication
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	1. High-Performance Database -MySQL 2. Caching-Redis 3. Content Delivery Networks (CDNs)-Amazon CloudFront. 4. Load Balancing- Nginx 5. Horizontal Scaling-Kubernetes for container orchestration. 6. Caching of Query Results-In-memory databases like Redis. 7. Query Optimization-Database query optimization tools. 8. Asynchronous Processing Message queues like RabbitMQ 9. Content Compression-Gzip for content compression. 10. Content Minification-Minification tools for HTML, CSS, and JavaScript. 11. Database Sharding- Database sharding techniques. 12. Monitoring and Profiling-Monitoring tools like Prometheus, New Relic, or Datadog.