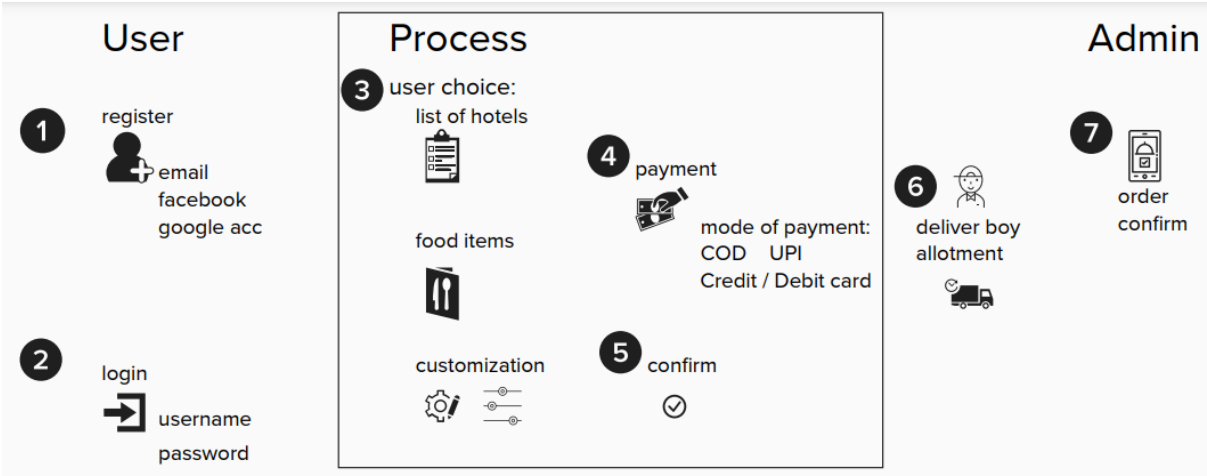
Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	25th October, 2023
Team ID	Team-590932
Project Name	**Snack Squad- A customisable snack ordering and delivery app.**

Team member name:	**Member details:**
Prasanna Gudivada	prasanna.21bce7104@vitapstudent.ac.in
Tishita Godavarthi	tishita.21bce7110@vitapstudent.ac.in
Varshitha Mattupalli	varshitha.21bce7327@vitapstudent.ac.in

Technical architecture:



Guidelines:

1. **Include All Processes (Application Logic / Technology Block**): Ensure that all essential processes within the application are defined and accounted for, including user interfaces, business logic, and any additional technology blocks. This should cover all user interactions, application functionality, and backend processes.
2. **Provide Infrastructural Demarcation (Local / Cloud):** Clearly specify the infrastructure demarcation, including whether components or services are deployed locally for development and testing or in the cloud for production. Outline the server configurations for each environment.
3. **Indicate External Interfaces (Third-Party APIs, etc.):** Identify and document all external interfaces, such as third-party APIs, that the Snack Squad app interacts with. Describe the purpose and usage of each external interface and ensure that their integration is well-documented.
4. **Indicate Data Storage Components/Services:** Define the data storage components and services used for different types of data. Specify whether you are using databases (e.g., MySQL or NoSQL), cloud-based data storage solutions (e.g., AWS DynamoDB), or other file storage mechanisms (e.g., IBM Block Storage).

5. **Indicate Interface to Machine Learning Models (If Applicable**): If the Snack Squad app utilizes machine learning models, clearly outline the interface between the application and these models. Describe the purpose and functionality of these models, as well as the technology and tools used for integration. By following these guidelines, you can ensure that the Snack Squad app project is well-documented and organized, with clear demarcation of processes, infrastructure, external interfaces, data storage components, and machine learning integration, ultimately leading to a well-structured and efficiently managed project.

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	web UI mobile app chat bot	HTML, CSS, JavaScript React native. dialogue flow for chatbot interaction
2.	Application Logic-1	business logic server logic	kotlin for Android, swift for iOS node.js for web backend
3.	Application Logic-2	voice recognition	IBM Watson speech to text service
4.	Application Logic-3	chatbot interaction	IBM Watson Assistant
5.	Database	user profiles snack inventory	MySQL NoSQL
6.	Cloud Database	cloud based storage	AWS DynamoDB
7.	File Storage	user content application files	Amazon S3 for cloud storage local file system and cloud storage
8.	External API-1	weather data	IBM Weather API, etc.
9.	External API-2	user verification	Aadhar API local file system and cloud storage for secure user identification

10.	Machine Learning Model	snack recommendation	custom machine learning model for personalised recommendations.
11.	Infrastructure (Server / Cloud)	application deployment local server configuration cloud server configuration cloud deployment orchestration	AWS cloud for scalability N/A (development and testing) AWS EC 2 instances for production KUBERNETES for containerization and scaling

Table-2: Application Characteristics:

S. No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	react for the user interface (web and mobile app) node JS for server logic. mongo DB as an NoSQL database
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)	micro services architecture for components scalability Uber needs for container orchestration and auto scaling
4.	Availability	Justify the availability of application (e.g., use of load balancers, distributed servers etc.)	Use of load balancers for even distribution of traffic AWS elastic load balancing for high availability
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Catching mechanisms to reduce database queries. content delivery networks for faster content delivery And reduce latency.