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

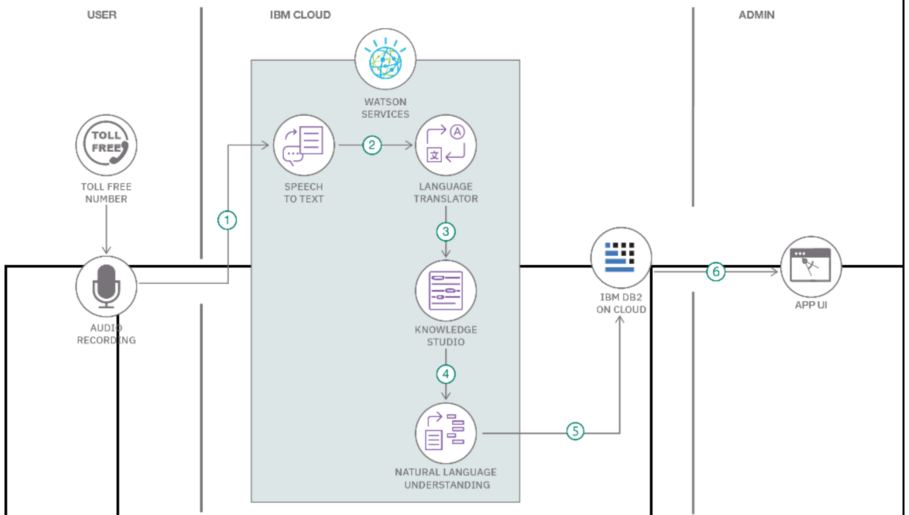
Date	6 April 2025
Team ID	SWTID1742834197
Project Name	Personal Expense Tracker App
Maximum Marks	4 Marks

Technical Architecture:

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

Example: Order processing during pandemics for offline mode

Reference: https://developer.ibm.com/patterns/ai-powered-backend-system-for-order-processing-during-pandemics/



Guidelines:

Include all the processes (As an application logic / Technology Block)
Provide infrastructural demarcation (Local / Cloud)
Indicate external interfaces (third party API's etc.)
Indicate Data Storage components / services
Indicate interface to machine learning models (if applicable)

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / React JS / Tailwind CSS

2.	Application Logic-1	Logic for adding, editing, and deleting expense entries	Node.js / Express.js
3.	Application Logic-2	Logic for managing user authentication (login, register)	JWT, bcrypt, Express.js
4.	Application Logic-3	Logic for category-wise filtering, search, and monthly limits	Node.js / Express.js
5.	Database	stores expense records, user info categories	MongoDB
6.	Cloud Database	Database Service on Cloud	MongoDB Atlas
7.	File Storage	Storing profile pictures or attachments (if any)	Cloudinary / Firebase Storage
8.	External API-1	Currency exchange rates (if supporting multi-currency)	ExchangeRate-API
9.	External API-2	Optional: Reminder API or email alert integration	SendGrid API

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Utilized open-source frameworks for both frontend and backend	React.js, Node.js, Express.js, MongoDB
2.	Security Implementations	Implemented user authentication and data protection mechanisms	JWT, bcrypt, HTTPS, SHA-256, OWASP guidelines
3.	Scalable Architecture	Modular codebase with clear separation of concerns; RESTful API design	MERN stack
4.	Availability	Deployed on reliable cloud services with automatic redeployments and versioning support	Vercel / Render / GitHub Actions
5.	Performance	Optimized data fetch using pagination, implemented caching at the client side	Browser Cache, Axios, Lazy Loading in React

References:

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d