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

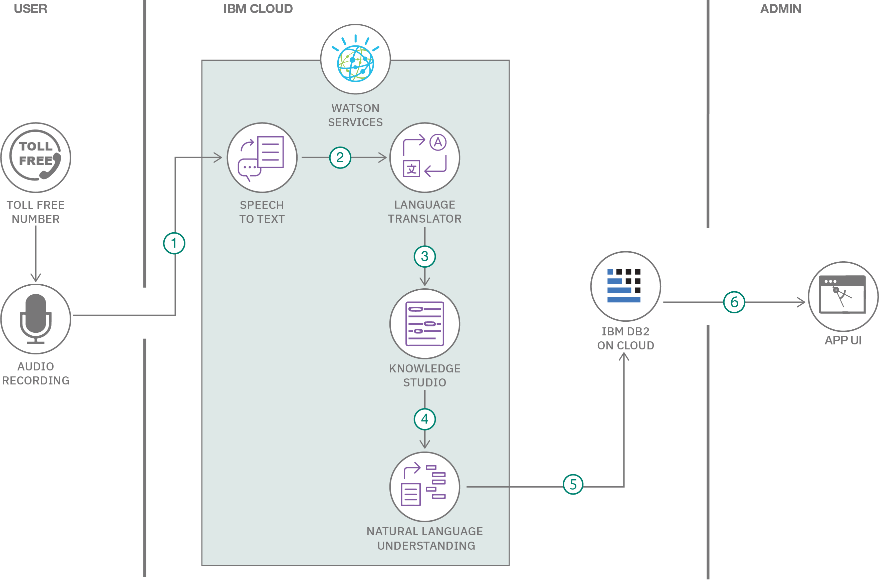
|  |  |
| --- | --- |
| Date | 24 June 2025 |
| Team ID | LTVIP2025TMID59024 |
| Project Name | Flight finder-navigating your air travel options |
| Maximum Marks | 4 Marks |

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/>

T

able-1 : Components & Technologies:

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 | Web interface to search and book flights | HTML, CSS, JavaScript / Angular Js / React Js etc. |
| 2. | Application Logic-1 | Backend logic for login, registration, booking, etc. | Java / Python |
| 3. | Application Logic-2 | model loading and prediction for flight price suggestions | IBM Watson STT service |
| 4. | Application Logic-3 | Authentication & session management | IBM Watson Assistant |
| 5. | Database | Store user data, flight data, and bookings | MySQL, etc. |
| 6. | Cloud Database | Optional cloud-based backup or live data | IBM DB2, IBM Cloudant etc. |
| 7. | File Storage | Local storage for CSV/Excel flight datasets | IBM Block Storage or Other Storage Service or Local Filesystem |
| 8. | External API-1 | Flight data APIs (if integrated live) | IBM Weather API, etc. |
| 9. | External API-2 | Email confirmation service | Aadhar API, etc. |
| 10. | Machine Learning Model | predict flight pricing or recommend best options | Object Recognition Model, etc. |
| 11. | Infrastructure (Server / Cloud) | Hosting the app for public access | Local, Cloud Foundry, Kubernetes, etc. |

Table-2: Application Characteristics:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | Use of open-source tools & libraries | Flask, Pandas, Scikit-learn, Bootstrap |
| 2. | Security Implementations | Secure login, password hashing, email verification | bcrypt, JWT, HTTPS, Flask-Login, Gmail OAuth |
| 3. | Scalable Architecture | Modular codebase, separation of frontend/backend, scalable DB | 3-tier architecture, Microservices used |
| 4. | Availability | High availability via cloud hosting | Load balancing (Render, Heroku Dynos, etc.) |

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 5. | Performance | Efficient queries, ML model caching, optimized routing | MongoDB indexing, Python caching, CDN (optional) |

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>