**Project Design Phase-II**

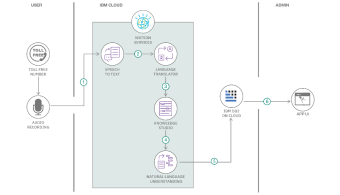
**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 26 October 2023 |
| Team ID |  |
| Project Name | Project -**ECOMMERCE SHIPPING PREDICTION USING MACHINE LEARNING** |
| 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: 

1. Include all the processes (As an application logic / Technology Block)

2. Provide infrastructural demarcation (Local / Cloud)

3. Indicate external interfaces (third party API’s etc.)

4. Indicate Data Storage components / services

5. Indicate interface to machine learning models (if applicable)

**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | The User utilizes the application with the help of  a web page. In the provided URL the User can read the description of the application. | HTML, CSS |
| 2. | Application Logic-1= To input the necessary fields for the prediction and submitting it. | In the “predict” html page, the user inputs the necessary fields. The data is requested to a variable using Flask “request” and “POST” methods when the submit button is clicked. | HTML, Python(Flask) |
| 3. | Application Logic-2 = To calculate the probability of the package arriving on time. | The best machine learning model is imported to the flask file using pickle extension.The data requested using the “POST” function is  passed to the model to predict the output. The output is passed to the “output” html file. | HTML, Python(Flask,pickle) |
| 4. | Application Logic-3 = To display the  output | The output is stored in a variable “output” and passed to “output” html file. In the “output” html file the predicted output is displayed. | HTML, Python(Flask) |
| 5. | Database | Data Type, Configurations etc. | —- |
| 6. | Cloud Database | Database Service on Cloud | —- |
| 7. | File Storage | File storage requirements | Files stored in local system |
| 8. | External API | Purpose of External API used in the application | —- |
| 9. | Machine Learning Model | Purpose of Machine Learning Model is the predict the whether the given package will arrive on time | Classification supervised ML Model |
| 10. | Infrastructure (Server / Cloud) | Application Deployment on Local System | Local System |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | Python micro web framework | Flask |
| 2. | Security Implementations | List all the security / access controls implemented, use of firewalls etc. | — |
| 3. | Scalable Architecture | Using flask and html we can scale the architecture using the session management technique, static file serving and horizontal scaling techniques. | Flask, Redis, Docker, Nginx |

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 4. | Availability | Load balancers distribute incoming traffic across multiple Flask application instances. Offloading static file serving to a dedicated web server or CDN reduces the load on the Flask application.Running multiple Flask instances and using horizontal scaling allows the application to handle increased traffic. | Static file serving, Horizontal scaling,  Load balancers. |
| 5. | Performance | Performance of the flask application will improve drastically. There will be even distribution of incoming traffic, ability to handle increased loads, faster response times and reduced workload on the flask app. | Static file serving, Horizontal scaling,  Load balancers. |

**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**