Technical Architecture:

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

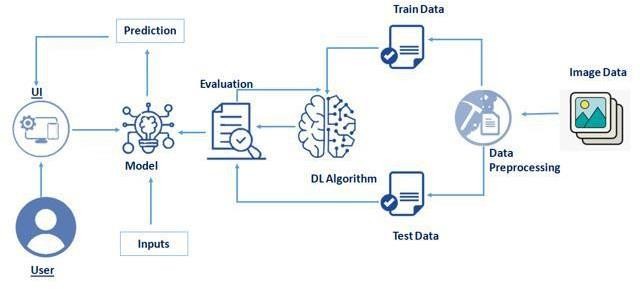


Table-1 : Components & Technologies:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | How user interacts with User interface to upload  image | Anaconda, Jupyter notebooks, Spyder, Python. |
| 2. | Model analyses | Once model analyses the uploaded image, the prediction is showcased on the UI | Kaggle.com, data.gov, UCI |
| 3. | Data collection | Create the dataset | Python, Keras, Numpy |
| 4. | Data Preprocessing-1 | Import the ImageDataGenerator library | Python, Keras, Numpy |
| 5. | Data Preprocessing-2 | Configure ImageDataGenerator class | Python, Numpy, Keras |
| 6. | Data Preprocessing-3 | Apply ImageDataGenerator functionality to Trainset and Testset | Python, Numpy, Keras |
| 7. | Model Building-1 | Import the model building libraries and Initializing The model | Python, Numpy, Keras, |
| 8. | Model Building-2 | Adding layers and configure | Python, Numpy, Keras |
| 9. | Model Building-3 | Training and testing the model, Optimize and save the model | Python, Numpy, Keras |
| 10. | Application Building | Purpose of create an HTML file and Building Python code | HTML, Python |
| 11. | Train the model on IBM | CNN Development and integrate it with the flask Application | IBM Watson |

Table-2: Application Characteristics:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | Open source software is that by which the source code or the base code is usually available for  modification or enhancement. | Flask(Python) |
| 2. | Security Implementations | By placing a filtration barrier between the targeted server and the attacker, the WAF is able to protect against attacks like cross site forgery, cross site  scripting and SQL injection. | e.g. SHA-256, Encryptions, IAM Controls, OWASP etc. |
| 3. | Scalable Architecture | Does not affect the performance even though used by many users. | Technology used |
| 4. | Availability | Justify the availability of application (e.g. use of  load balancers, distributed servers etc.) | Technology used |
| 5. | Performance | Design consideration for the performance of the  application (number of requests per sec, use of Cache, use of CDN’s) etc. | Technology used |

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>