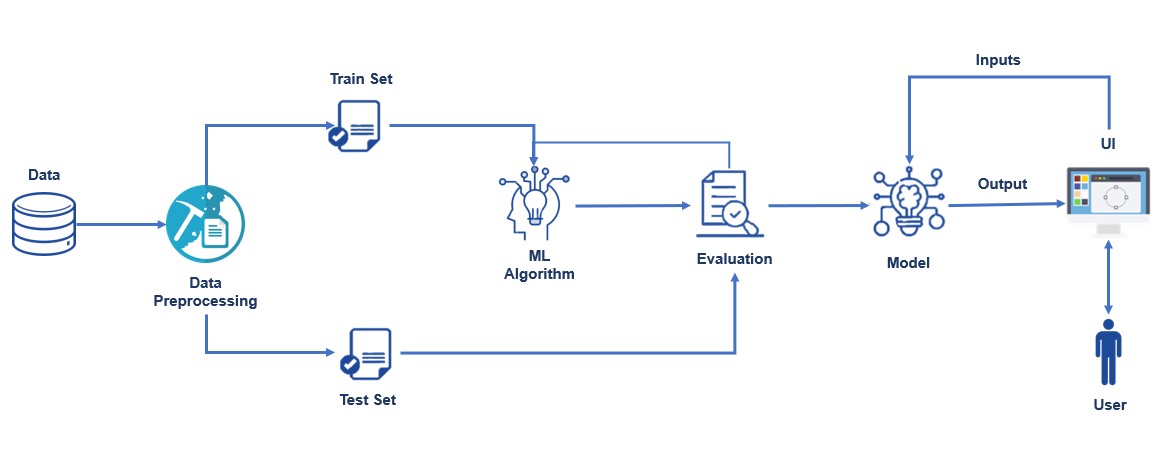
**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 17 October 2022 |
| Team ID | PNT2022TMID11612 |
| Project Name | Web Phishing Detection |
| Maximum Marks | 4 Marks |

**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 application Web UI | HTML, CSS |
| 2. | Application Logic-1 | Logic for a process in the application | Python |
| 3. | Application Logic-2 | Logic for a process in the application | Flask, IBM Cloud |
| 4. | Application Logic-3 | Logic for a process in the application | IBM Watson Assistant |
| 5. | Database | Data Type, Configurations etc. | MongoDB |
| 6. | Cloud Database | Database Service on Cloud | IBM DB2, IBM Cloudant |
| 7. | File Storage | File storage requirements | Local Filesystem |
| 8. | External API-1 | Purpose of External API used in the application | Google API, Apple REST API |
| 9. | Machine Learning Model | Purpose of Machine Learning Model | Logistic Regression, K means clustering, CNN etc |
| 10. | Infrastructure (Server / Cloud) | Application Deployment on Local System / Cloud Local Server Configuration:  Cloud Server Configuration: | Local, Cloud Foundry, Kubernetes, etc. |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | List the open-source frameworks used | Wifiphisher, Gophish, Zphisher, EvilginX |
| 2. | Security Implementations | List all the security / access controls implemented, use of firewalls etc. | Two factor verification |
| 3. | Scalable Architecture | Justify the scalability of architecture (3 – tier, Micro-services) | Flask API, Microservices for scalability |
| 4. | Availability | Justify the availability of application (e.g. use of load balancers, distributed servers etc.) | Auto scaling using IBM cloud |
| 5. | Performance | Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN’s) etc. | Using effective message, effect of message persistence, using queues of different lengths. |