Project Design Phase -II

Technology Stack (Architecture & Stack)

Date: 24-06-2024

ID: Swagath Nalla

Project Name: Online Payment Fraud Detection

Maximum Marks:4

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

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1 | User Interface | How user interacts with application | HTML, CSS, Bootstrap, JavaScript |
| 2 | Application Logic-1 | Main application logic for fraud detection | Python, Flask |
| 3 | Machine Learning | Fraud detection model | BalancedRandomForestClassifier, scikit-learn |
| 4 | Data Processing | Data preprocessing and feature engineering | pandas, numpy, imbalanced-learn |
| 5 | Database | Data storage and retrieval | Local CSV file (used in the example) |
| 6 | Model Persistence | Saving and loading trained models | joblib |
| 7 | Deployment Platform | Platform for deploying the web application | Render |
| 8 | Logging | Logging application activities and errors | Python logging module |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1 | Open-Source Frameworks | List the open-source frameworks used | Flask, scikit-learn, imbalanced-learn |
| 2 | Security Implementations | List all the security/access controls implemented, use of firewalls etc. | HTTPS via Render, form validation |
| 3 | Scalable Architecture | Justify the scalability of architecture  (3-tier, Micro-services) | Flask can be scaled with WSGI servers like  Gunicorn |
| 4 | Availability | Justify the availability of application  (e.g., use of load balancers, distributed servers) | Render provides scalable cloud infrastructure |
| 5 | Performance | Design consideration for the performance of the  application (number of requests per sec, use of Cache, use of CDN’s) etc. | Render's cloud platform, caching layers like Redis  if needed |