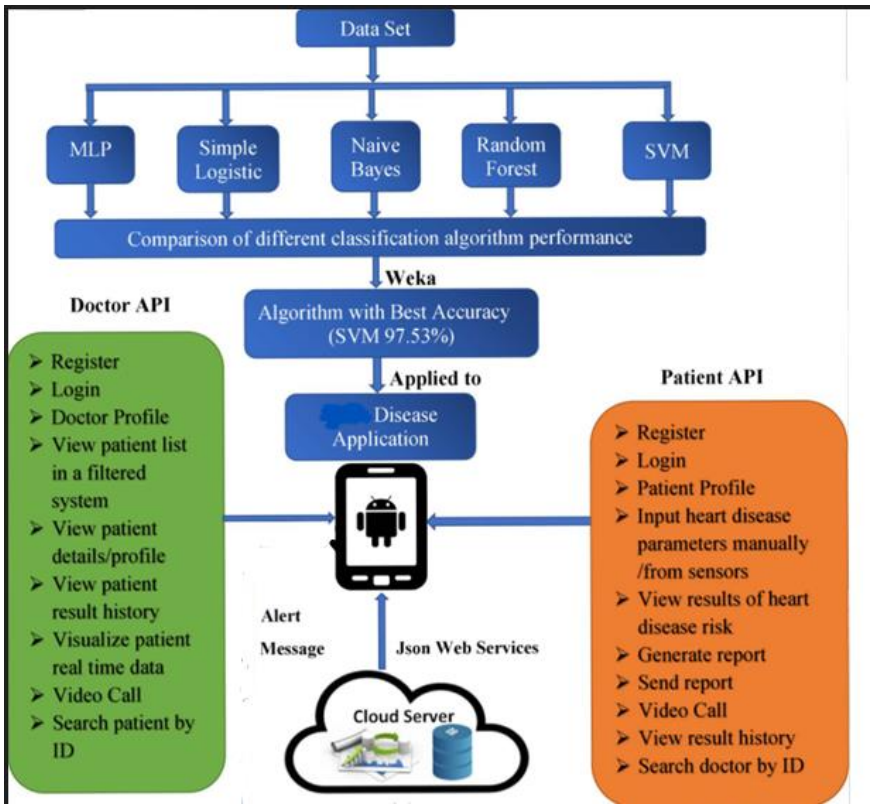
| | |
|---|---|
| Date | 27 October 2023 |
| Team ID | Team-592629 |
| Project Name | Project - Disease Prediction Using Machine Learning |
| Maximum Marks | 4 Marks |

# Technical Architecture

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

| S.No | Component | Description | Technology |
|---|---|---|---|
| 1 | User Interface | How user interacts with application I.e Web page | HTML, CSS, JavaScript / Angular Js / React Js/Python |
| 2 | Application Logic-1 | Logic for the main process in the application | Python(app.py) /ML model deployment |
| 3 | Application Logic-2 | Speech-to-Text service for audio to text | IBM Watson STT service |
| 4 | Application Logic-3 | Chatbot for user interaction | IBM Watson Assistant |
| 5 | Database | Data storage and management | MySQL/NOSQL/MongoDb |
| 6 | Cloud Database | Cloud-based database service for scalability | IBM Cloudant |
| 7 | File Storage | File storage for application requirements | IBM Block Storage or Other Storage Service or Local Filesystem |
| 8 | External API-1 (doctor api) | External API for fetching doctor information | IBM API |
| 9 | External API-1 (Patient api) | External API for fetching Patient information | IBM API |
| 10 | Machine Learning Model | Machine learning models for disease prediction | Random Forest, K-Nearest Neighbors (KNN),SVM |
| 11 | Infrastructure (Server / Cloud) | Application deployment on local system or cloud | Local Server Configuration, Cloud Foundry, Kubernetes, etc. |

- **Block Daigrame:**




- **Table-2: Application Characteristics:**

| S.NO | characteristics | Description | Technology |
|---|---|---|---|
| 1 | Open-Source Frameworks | List the open-source frameworks used | Python, Scikit-learn (for machine learning) |
| 2 | Security Implementations | List all the security / access controls implemented,use of firewalls etc. | SHA-256, Encryption, IAM Controls, Access Control Lists (ACLs), OWASP Best Practices |
| 3 | Scalable Architecture | Justify the scalability of architecture | Microservices, Load Balancers, (3-tier, Micro-services) \| Kubernetes, Cloud-Native Technologies |
| 4 | Availability | Justify the availability of application | Load Balancers, Distributed Servers, (e.g., use of load balancers,High Availability Architectures |

| | | | |
|---|---|---|---|
| 5 | Performance | Design considerations for application | Caching, Content Delivery Networks (CDNs performance (requests per second, use of Request Optimization, Efficient Algorithms cache, use of CDNs) |