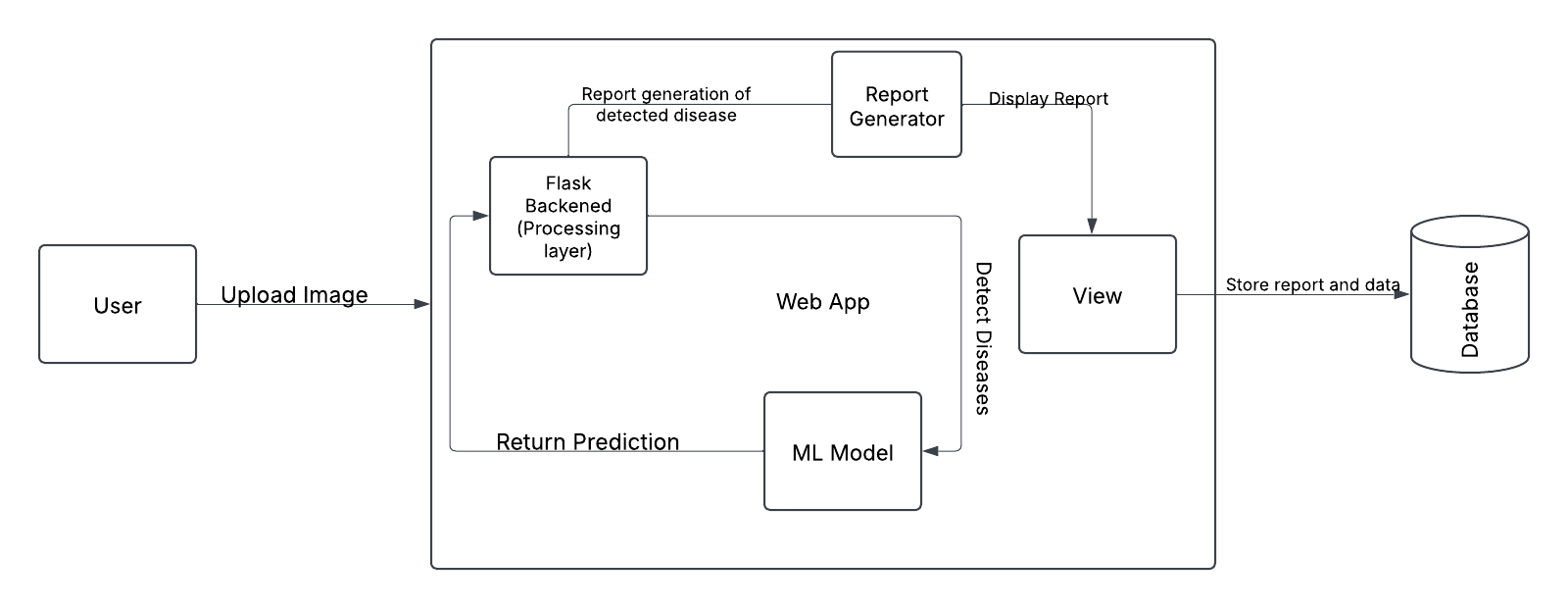
**Requirement Analysis**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 02 February 2025 |
| Team ID |  |
| Project Name | Eye Disease Detection Using Deep 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



----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Table 1: Components & Technologies**

| **S.No** | **Component** | **Description** | **Technology** |
| --- | --- | --- | --- |
| 1 | User Interface | Web UI where doctors can upload images, view results, and download reports. | HTML, CSS, JavaScript, React.js, Flask |
| 2 | Application Logic-1 | Backend logic that handles image upload, preprocessing, and database interactions. | Python, Flask |
| 3 | Application Logic-2 | Image preprocessing pipeline (resizing, normalization). | OpenCV, NumPy, TensorFlow/Keras |
| 4 | Application Logic-3 | Model inference logic – passing the image to a trained CNN model. | TensorFlow, Keras |
| 5 | Database | Stores user details, image metadata, and prediction results. | PostgreSQL / MongoDB |
| 6 | Cloud Database | Cloud-based database for scalability. | Firebase, AWS RDS, MongoDB Atlas |
| 7 | File Storage | Stores uploaded images and generated reports. | AWS S3 / Google Cloud Storage / Local Filesystem |
| 8 | External API-1 | API for user authentication (optional). | Google OAuth / Firebase Auth |
| 9 | External API-2 | API for report sharing via email (optional). | SendGrid / Twilio |
| 10 | Machine Learning Model | Deep learning model for eye disease detection. | VGG19-based model, TensorFlow/Keras |
| 11 | Infrastructure (Server/Cloud) | Hosting and deployment of the application. | AWS EC2 / GCP App Engine / Azure |

**Table 2: Application Characteristics**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
| 1 | Open-Source Frameworks | Frameworks used for model development and backend. | Flask, TensorFlow, OpenCV, React.js |
| 2 | Security Implementations | Encryption of user data, secure authentication, protection against injection attacks. | SHA-256, SSL/TLS, Firebase Auth, Role-based Access Control (RBAC) |
| 3 | Scalable Architecture | The system follows a modular and scalable design, supporting cloud scaling. | Microservices architecture, Kubernetes, Docker |
| 4 | Availability | Ensuring minimal downtime by distributing workload using cloud services. | Load Balancing, Auto-scaling (AWS/GCP), Multiple Database Replicas |
| 5 | Performance | Optimized model inference and caching for faster predictions. | Redis Cache, CloudFront (CDN), Model Quantization |