Project Design Phase-II

Technology Stack (Architecture & Stack)

Date: 28 June 2025
Team ID: LTVIP2025TMID45560
Project Name: Revolutionizing Liver Care: Predicting Liver Cirrhosis using Advanced Machine Learning Techniques
Maximum Marks: 4 Marks

Technical Architecture:

System Overview

The Liver Cirrhosis Prediction System is a web-based machine learning application that analyzes clinical and biochemical patient data to predict the likelihood of liver cirrhosis. The system follows a three-tier architecture with a web-based user interface, Flask backend processing, and machine learning model integration.

Architecture Diagram Description:

```
[User Interface] → [Flask Web Server] → [ML Model Layer] → [Data Processing] →
[Prediction Results]
      ↓                    ↓                     ↓
[HTML/CSS/JS]    [Python/Flask]    [XGBoost/Scikit-learn]
```

Table-1: Components & Technologies

S.No	Component	Description	Technology
1.	User Interface	Web-based form for clinical data input and result display	HTML5, CSS3, JavaScript, Bootstrap
2.	Application Logic-1	Web server and routing logic for handling user requests	Python Flask Framework
3.	Application Logic-2	Data preprocessing and validation logic	Python (Pandas, NumPy)
4.	Application Logic-3	Machine learning model inference and prediction logic	Python (Scikit-learn, XGBoost)
5.	Database	Patient data storage and model persistence	SQLite (Development), PostgreSQL (Production)
6.	Cloud Database	Scalable database service for production deployment	AWS RDS / Google Cloud SQL
7.	File Storage	Model storage and static file management	Local Filesystem / AWS S3
8.	External API-1	Healthcare data validation and	Medical Data Validation API

S.No	Component	Description	Technology
		normalization	
9.	External API-2	Patient identity verification (if required)	Healthcare Identity Verification API
10.	Machine Learning Model	Liver cirrhosis prediction using clinical features	XGBoost Classifier, Random Forest
11.	Infrastructure	Application deployment environment	Local Server: Python 3.8+, Flask Cloud: AWS EC2, Heroku, Docker

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Machine learning and web development frameworks	Flask, Scikit-learn, XGBoost, Pandas, NumPy, Matplotlib
2.	Security Implementations	Data encryption, secure file handling, input validation	SSL/TLS encryption, Input sanitization, CSRF protection, Secure headers
3.	Scalable Architecture	Modular design supporting horizontal scaling	3-tier architecture: Presentation (Flask), Business Logic (ML Pipeline), Data Layer (Database)
4.	Availability	High availability through load balancing and redundancy	Load Balancers (NGINX), Docker containers, Health monitoring
5.	Performance	Optimized for fast prediction response times	Model caching, Efficient data preprocessing, Compressed model serialization (Pickle)

Detailed Technical Specifications:

Frontend Layer:

- **Technology:** HTML5, CSS3, JavaScript, Bootstrap
- **Functionality:** User-friendly form for clinical data input, responsive design
- **Features:** Real-time validation, progress indicators, result visualization

Backend Layer:

- **Technology:** Python Flask
- **Functionality:** REST API endpoints, request handling, model integration
- **Features:** Input validation, error handling, logging, session management

Machine Learning Layer:

- **Technology:** XGBoost, Scikit-learn, Pandas, NumPy
- **Models:** XGBoost Classifier (Primary), Random Forest (Backup)
- **Features:** Feature engineering, model ensemble, prediction confidence scoring

Data Layer:

- **Technology:** SQLite (Development), PostgreSQL (Production)
- **Functionality:** Patient data storage, model versioning, audit logs
- **Features:** Data encryption, backup and recovery, query optimization

Deployment Architecture:

- **Development:** Local Flask development server
- **Staging:** Docker containers with automated testing
- **Production:** Cloud deployment (AWS/Heroku) with CI/CD pipeline

Security Considerations:

1. **Data Privacy:** HIPAA-compliant data handling
2. **Encryption:** End-to-end encryption for sensitive medical data
3. **Authentication:** Secure user authentication (if multi-user)
4. **Input Validation:** Comprehensive input sanitization
5. **Audit Logging:** Complete audit trail for medical predictions

Performance Metrics:

- **Response Time:** < 2 seconds for prediction requests
- **Throughput:** 100+ concurrent users
- **Accuracy:** 90.1% prediction accuracy (XGBoost model)
- **Availability:** 99.9% uptime target

Scalability Features:

1. **Horizontal Scaling:** Containerized deployment for easy scaling
2. **Load Distribution:** NGINX load balancer for traffic distribution
3. **Database Scaling:** Read replicas for improved performance
4. **Caching:** Redis for model result caching
5. **CDN Integration:** Static content delivery optimization

References:

- Flask Documentation: https://flask.palletsprojects.com/
- XGBoost Documentation: https://xgboost.readthedocs.io/
- Scikit-learn Documentation: https://scikit-learn.org/
- AWS Architecture Center: https://aws.amazon.com/architecture/
- Docker Documentation: https://docs.docker.com/
- Healthcare Data Security: https://www.hhs.gov/hipaa/