

Solution Architecture

Revolutionizing Liver Care: Predicting Liver Cirrhosis using Advanced Machine Learning Techniques

Date: 28 June 2025

Team ID: LTVIP2025TMID45560

Project Name: Liver Cirrhosis Prediction System

Maximum Marks: 4 Marks

1. Solution Architecture Overview

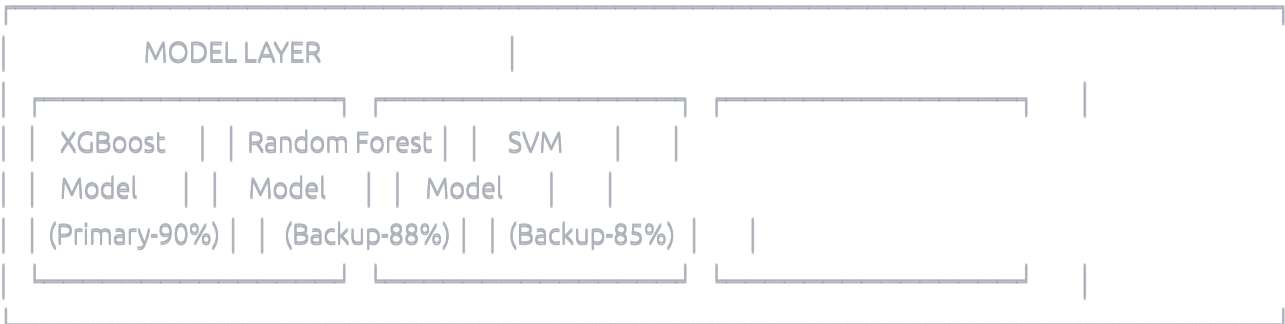
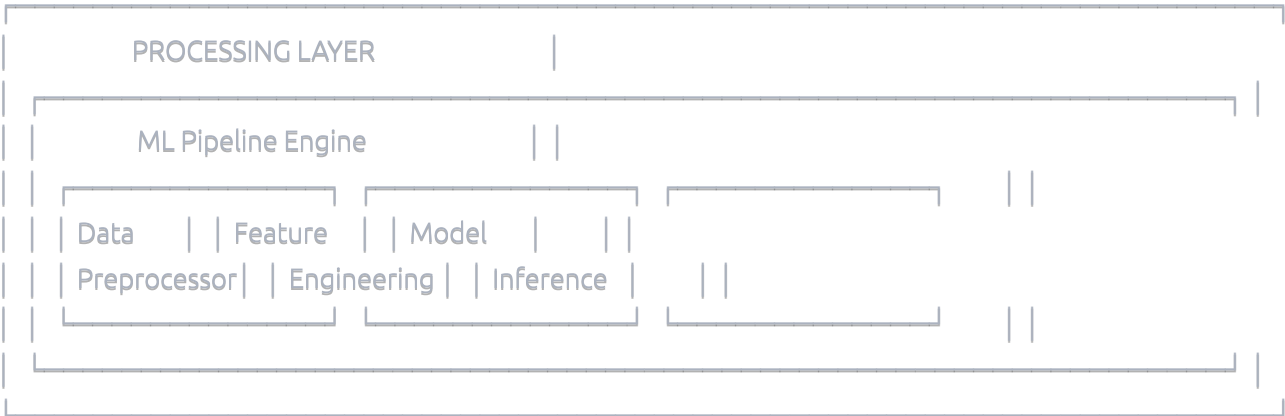
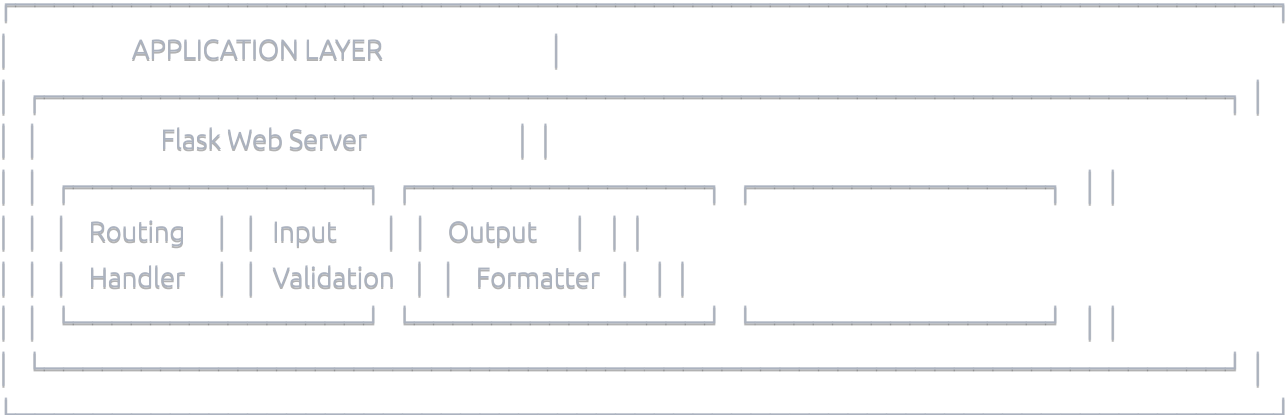
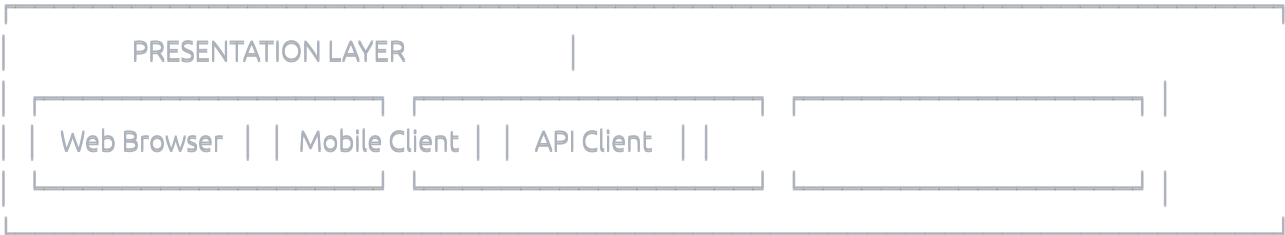
The solution architecture bridges the gap between the clinical need for early liver cirrhosis detection and advanced machine learning technology. This system provides a comprehensive, non-invasive approach to predict liver cirrhosis using clinical and biochemical data.

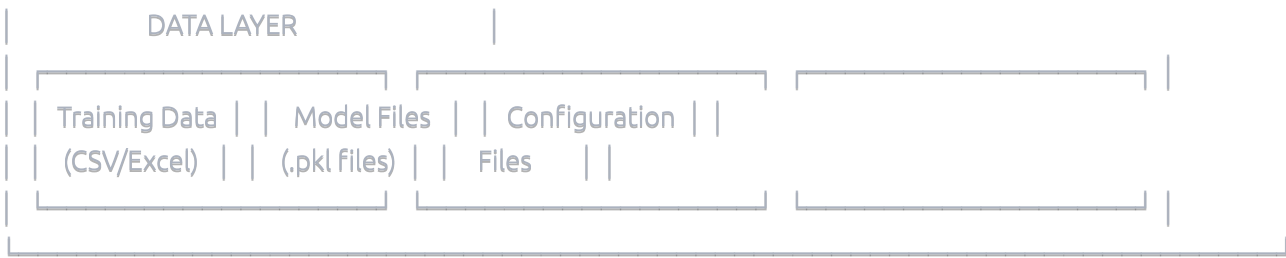
Goals:

- **Primary Goal:** Develop an accurate ML-based prediction system for early liver cirrhosis detection
 - **Technical Goal:** Create a scalable web application that integrates multiple ML models
 - **Clinical Goal:** Provide healthcare professionals with a cost-effective diagnostic support tool
 - **Business Goal:** Reduce healthcare costs by enabling early intervention and preventing advanced-stage treatments
-

2. System Architecture Components

2.1 High-Level Architecture





2.2 Detailed Component Architecture

A. Data Input Layer

- **Input Sources:** Web forms, API endpoints, file uploads
- **Data Types:**
 - Patient demographics (Age, Gender)
 - Clinical parameters (Bilirubin, Albumin, Hemoglobin)
 - Laboratory results (Liver enzymes, Prothrombin time)
- **Format Support:** JSON, CSV, Excel, Form data

B. Data Processing Pipeline

Raw Input → Data Validation → Missing Value Handling →
Feature Encoding → Normalization → Feature Selection → Model Input

Processing Steps:

1. **Data Validation:** Input format and range validation
2. **Missing Value Handling:** Imputation using median/mode
3. **Categorical Encoding:** Label encoding for gender and other categorical features
4. **Normalization:** Min-Max scaling or standardization
5. **Feature Selection:** Based on correlation analysis and domain knowledge

C. Machine Learning Engine

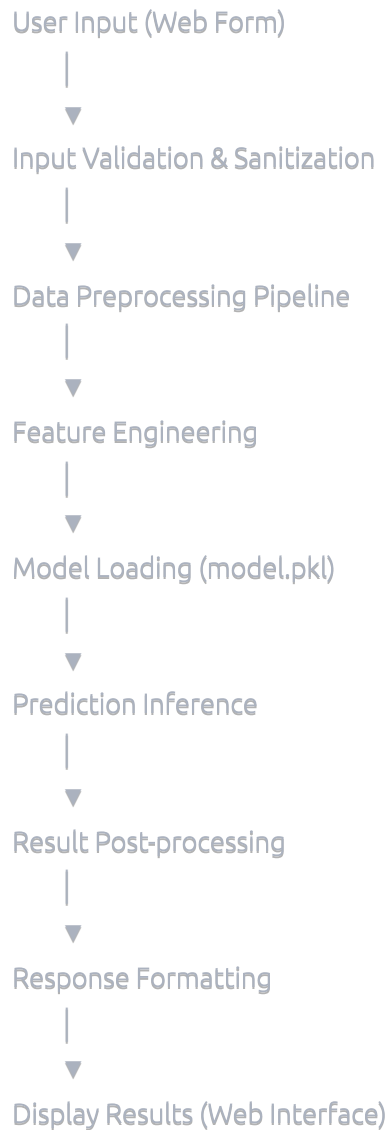
- **Primary Model:** XGBoost (90.1% accuracy)
- **Backup Models:** Random Forest (88.6%), SVM (85.2%)
- **Model Selection Logic:** Automatic fallback mechanism
- **Ensemble Option:** Voting classifier for critical cases

D. Web Application Framework

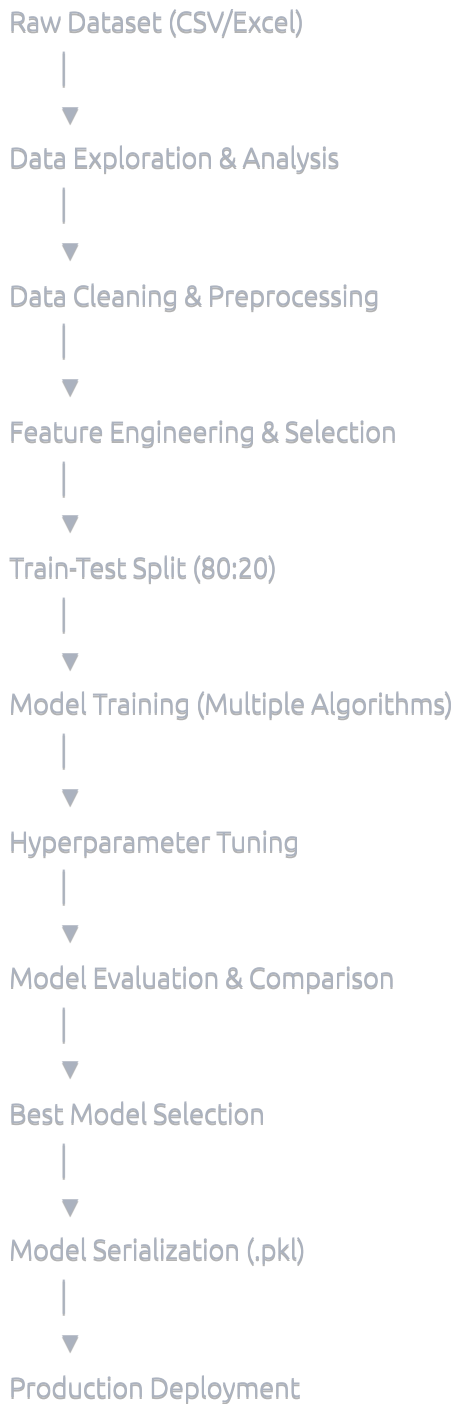
- **Backend:** Python Flask
 - **Frontend:** HTML/CSS/JavaScript
 - **Templates:** Jinja2 templating engine
 - **Static Assets:** CSS, JavaScript, images
-

3. Data Flow Architecture

3.1 Prediction Workflow



3.2 Training Pipeline (Development Phase)



4. Technology Stack Specifications

4.1 Development Environment

- **Programming Language:** Python 3.8+
- **Web Framework:** Flask 2.0+
- **Machine Learning:** Scikit-learn, XGBoost
- **Data Processing:** Pandas, NumPy
- **Visualization:** Matplotlib, Seaborn

- **Model Serialization:** Pickle

4.2 Dependencies

python

Flask==2.0.1

scikit-learn==1.0.2

xgboost==1.5.0

pandas==1.3.3

numpy==1.21.2

matplotlib==3.4.3

seaborn==0.11.2

4.3 File Structure

```
liver_cirrhosis_prediction/
├── app.py          # Main Flask application
├── model.pkl       # Trained XGBoost model
├── requirements.txt # Python dependencies
├── templates/
│   ├── index.html  # Input form
│   └── result.html  # Prediction results
├── static/
│   ├── css/
│   ├── js/
│   └── images/
├── data/
│   └── training_data.csv
└── models/
    ├── xgboost_model.pkl
    ├── random_forest_model.pkl
    └── svm_model.pkl
```

5. Deployment Architecture

5.1 Local Development

- **Server:** Flask development server
- **Port:** 5000 (default)
- **Environment:** Local machine

5.2 Production Deployment Options

Option A: Cloud Deployment (Recommended)

- **Platform:** AWS/Google Cloud/Azure
- **Service:** Elastic Beanstalk/App Engine/App Service
- **Load Balancer:** Application Load Balancer
- **Auto Scaling:** Based on CPU/memory usage

Option B: Container Deployment

- **Containerization:** Docker
- **Orchestration:** Kubernetes
- **Registry:** Docker Hub/ECR

Option C: Traditional Server Deployment

- **Web Server:** Nginx/Apache
 - **WSGI Server:** Gunicorn/uWSGI
 - **Operating System:** Ubuntu/CentOS
-

6. Security Architecture

6.1 Data Security

- **Input Validation:** Prevent SQL injection and XSS
- **Data Sanitization:** Clean and validate all inputs
- **HTTPS:** SSL/TLS encryption for data transmission
- **Session Management:** Secure session handling

6.2 Model Security

- **Model Protection:** Pickle file integrity checks
 - **Access Control:** Restricted model file access
 - **Audit Logging:** Track prediction requests
-

7. Performance Specifications

7.1 Model Performance Metrics

Model	Accuracy	Precision	Recall	F1-Score
XGBoost (Primary)	90.1%	89.0%	90.7%	89.8%
Random Forest	88.6%	87.2%	89.0%	88.1%
SVM	85.2%	84.0%	85.0%	84.5%

7.2 System Performance Requirements

- **Response Time:** < 2 seconds for prediction
- **Throughput:** 100+ concurrent users
- **Availability:** 99.5% uptime
- **Scalability:** Horizontal scaling capability

8. Integration Points

8.1 External Integrations

- **Healthcare Systems:** HL7 FHIR compatibility
- **Electronic Health Records (EHR):** Integration APIs
- **Laboratory Information Systems:** Data import capabilities

8.2 API Specifications

- **REST API:** JSON request/response format
- **Authentication:** API key-based authentication
- **Rate Limiting:** Prevent API abuse

9. Monitoring and Maintenance

9.1 Application Monitoring

- **Health Checks:** Endpoint monitoring
- **Performance Metrics:** Response time, error rates
- **Logging:** Comprehensive application logs

9.2 Model Monitoring

- **Model Drift Detection:** Monitor prediction accuracy over time
- **Retraining Schedule:** Quarterly model updates
- **A/B Testing:** Compare model versions

10. Future Enhancements

10.1 Technical Improvements

- **Deep Learning Models:** Neural network implementation
- **Real-time Processing:** Streaming data support
- **Mobile Application:** Native iOS/Android apps

10.2 Feature Enhancements

- **Explainable AI:** SHAP/LIME integration
- **Multi-language Support:** Internationalization
- **Advanced Visualizations:** Interactive dashboards

11. Conclusion

This solution architecture provides a robust, scalable, and maintainable framework for the liver cirrhosis prediction system. The architecture supports the project's primary objectives of early disease detection, clinical decision support, and cost-effective screening while ensuring high performance and reliability.

The modular design allows for easy maintenance, future enhancements, and integration with existing healthcare systems, making it a comprehensive solution for revolutionizing liver care through advanced machine learning techniques.