Software Requirements Specification (SRS)

Parkinsons Disease Detection with Model Comparison

Document Version: 1.0

Date: May 4, 2025

Prepared by: Ritwik Mittal & Yashas Raina

Table of Contents

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) document provides a comprehensive description of the Parkinson's Disease Detection Platform. It details the functional and non-functional requirements, system architecture, and design specifications for a software system that enables healthcare professionals to analyze patient data for Parkinson's disease detection using machine learning techniques.

This document adheres to the IEEE 830-1998 standard for software requirements specifications and is intended for use by developers, project managers, quality assurance personnel, and other stakeholders involved in the development and deployment of the system.

1.2 Scope

The Parkinson's Disease Detection Platform is a web-based application designed to assist healthcare professionals in the early detection and analysis of Parkinson's disease using machine learning algorithms. The system allows users to upload patient data, process it through three different machine learning models (KNN, Random Forest, and SVM), and compare their performance using various metrics, with a particular focus on confusion matrices and accuracy measures.

The system includes:

- Data upload and preprocessing capabilities
- Integration with three pre-trained machine learning models
- Comparative analysis of model performance
- Visualization of results through interactive charts and tables
- Data export functionality for further analysis

The system does not include:

- Electronic health record (EHR) integration
- Real-time patient monitoring
- Treatment recommendation functionality

- Automated diagnosis without human oversight

1.3 Definitions, Acronyms, and Abbreviations

Term/Acronym	Definition
PD	Parkinson's Disease
ML	Machine Learning
KNN	K-Nearest Neighbors algorithm
RF	Random Forest algorithm
SVM	Support Vector Machine algorithm
UI	User Interface
API	Application Programming Interface
CSV	Comma-Separated Values (file format)
TXT	Text File (file format)
SRS	Software Requirements Specification
DFD	Data Flow Diagram
ERD	Entity Relationship Diagram

1.4 References

1. IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications
2. Parkinson's Disease Detection - Data Collection Protocol Document
3. Machine Learning Model Documentation for KNN, Random Forest, and SVM
4. Streamlit Documentation - https://docs.streamlit.io/
5. Python Scientific Stack Documentation (NumPy, Pandas, Scikit-learn, Matplotlib, Seaborn)

1.5 Overview

The remainder of this document is organized as follows:

- **Section 2** provides an overall description of the system, including product perspective, functions, user characteristics, constraints, and assumptions.
- **Section 3** outlines the specific system features and requirements.
- **Section 4** details the external interface requirements.
- **Section 5** describes the non-functional requirements.
- **Section 6** covers additional requirements not addressed in previous sections.
- **Section 7** contains appendices with diagrams and supplementary information.

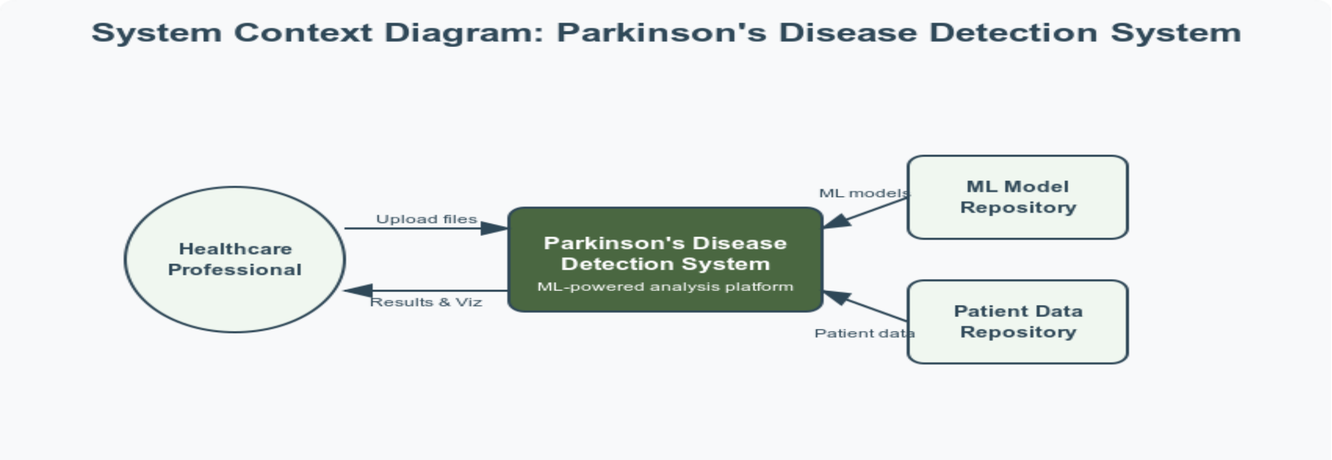
2. Overall Description

2.1 Product Perspective

The Parkinson's Disease Detection Platform is a standalone web application that can be deployed in clinical settings to assist healthcare professionals in early detection and analysis of Parkinson's disease. The system builds upon existing machine learning models that have been pre-trained on Parkinson's disease datasets.

The system operates within the broader clinical diagnostic workflow, serving as a supportive tool for healthcare professionals rather than as an autonomous diagnostic system. Results from the platform are intended to supplement, not replace, clinical judgment and standard diagnostic procedures.

System Context Diagram:


System Context Diagram: Parkinson's Disease Detection System

2.2 Product Functions

The Parkinson's Disease Detection Platform provides the following core functions:

1. **Data Management**
   - Upload patient data files in CSV or TXT formats
   - Configure data parsing options (delimiters, headers, etc.)
   - Display data preview and basic statistics
   - Validate data for format compliance and completeness
2. **Model Management**
   - Upload pre-trained machine learning models (KNN, RF, SVM)
   - Name and identify models for comparison purposes
   - Configure model display settings
3. **Analysis Functions**
   - Process patient data through multiple ML models
   - Generate performance metrics (accuracy, etc.)
   - Create confusion matrices for each model
   - Compare model performance side-by-side
4. **Visualization Functions**
   - Display interactive performance metric charts
   - Render comparative confusion matrices
   - Provide detailed analysis views
   - Allow for export of visualizations
5. **Utility Functions**
   - Download analysis results
   - Configure display settings
   - Access usage instructions and documentation

2.3 User Classes and Characteristics

The system is designed for the following user classes:

1. **Primary Users: Healthcare Professionals**
   - Neurologists, movement disorder specialists, and general practitioners
   - May have limited technical knowledge of machine learning algorithms
   - Need clear, interpretable results for clinical decision support
   - Concerned with the accuracy and reliability of diagnostic aids
   - Typical Usage: Evaluating patient data for potential early signs of Parkinson's disease
2. **Secondary Users: Clinical Researchers**
   - Researchers in neurodegenerative diseases and related fields
   - Higher technical proficiency with data analysis and machine learning concepts
   - Need detailed performance metrics and comparison capabilities
   - Typical Usage: Evaluating model performance on specific patient populations or comparing detection methodologies
3. **Tertiary Users: IT Support Staff**
   - Technical support personnel within healthcare facilities
   - Responsible for system deployment and maintenance
   - Need clear documentation and error reporting
   - Typical Usage: Installing the system, uploading trained models, and troubleshooting issues

2.4 Operating Environment

The Parkinson's Disease Detection Platform is designed to operate in the following environment:

- **Hardware**: Standard desktop or laptop computers with minimum 8GB RAM and 2.5GHz processor
- **Operating System**: Cross-platform (Windows, macOS, Linux)

- **Web Browser**: Modern web browsers (Chrome, Firefox, Safari, Edge) with JavaScript enabled
- **Back-end**: Python 3.8+ environment with necessary libraries installed
- **Network**: Local network deployment or secured internet connection
- **Screen Resolution**: Minimum 1366x768, optimized for 1920x1080
- **Storage**: Minimum 500MB for application, additional storage based on data volume
- **Concurrent Users**: Designed for single-user operation at a time

2.5 Design and Implementation Constraints

The development and deployment of the system are subject to the following constraints:

1. **Regulatory Compliance**
    - Must comply with relevant healthcare data protection regulations (e.g., HIPAA)
    - Cannot store patient data beyond the current session without appropriate safeguards
    - Must include appropriate disclaimers regarding the assistive (non-diagnostic) nature of the tool
2. **Technical Constraints**
    - Built using Streamlit framework for the front-end interface
    - Uses Python scientific stack (NumPy, Pandas, scikit-learn, etc.) for processing
    - Compatible with .pkl and .joblib model formats only
    - Limited to analysis of structured data in CSV or TXT formats
3. **Security Constraints**
    - No external API calls permitted to maintain data privacy
    - Local processing of all data to prevent network exposure of sensitive information
    - No persistent storage of patient data beyond the current session

4. **Deployment Constraints**
    - Must be deployable in healthcare environments with varying IT infrastructures
    - Should function in environments with limited internet connectivity
    - Installation should not require advanced system administration knowledge

2.6 User Documentation

The following user documentation will be provided with the system:

1. **Installation Guide**
    - System requirements
    - Installation procedures
    - Configuration instructions
    - Troubleshooting common issues
2. **User Manual**
    - Complete walkthrough of system features
    - Step-by-step instructions for all operations
    - Explanation of result interpretation
    - Data format requirements and examples
3. **Quick Start Guide**
    - Basic operation instructions
    - Common use cases
    - Quick reference for key functions
4. **In-application Help**
    - Context-sensitive help indicators
    - Tool tips and explanations
    - Expandable information sections

2.7 Assumptions and Dependencies

Assumptions

1. Users have access to properly formatted patient data containing relevant features for Parkinson's disease detection
2. Users have appropriate pre-trained ML models or access to the default provided models
3. The system will be used as a supplementary tool, not as the sole basis for diagnosis
4. Users have basic computer literacy and understanding of data formats (CSV, TXT)
5. The operating environment meets the minimum technical requirements specified

Dependencies

1. **External Libraries**
    - Streamlit (v1.20.0 or higher)
    - Pandas (v1.5.0 or higher)
    - NumPy (v1.22.0 or higher)
    - Scikit-learn (v1.1.0 or higher)
    - Matplotlib (v3.5.0 or higher)
    - Seaborn (v0.12.0 or higher)
    - Joblib (v1.2.0 or higher)
2. **Pre-trained Models**
    - Availability of trained KNN, Random Forest, and SVM models for Parkinson's detection
    - Compatibility between the models' expected input format and the system's preprocessing capabilities
3. **Reference Data**

- Access to standard feature definitions for Parkinson's disease detection
- Normative ranges for features when applicable

3. System Features

3.1 Data Management

3.1.1 Description

The Data Management feature allows users to upload, preview, and verify patient data before processing it through the ML models. It supports various file formats and configuration options to ensure flexibility with different data sources.

3.1.2 Functional Requirements

3.1.2.1 File Upload

- The system shall allow users to upload data files in CSV and TXT formats.
- The system shall support files up to 100MB in size.
- The system shall validate the uploaded file for readability and format compliance.
- The system shall display file metadata (file name, size, type) upon successful upload.

3.1.2.2 Data Parsing Configuration

- The system shall allow users to specify the delimiter used in the data file (comma, semicolon, tab, space, or pipe).
- The system shall allow users to specify whether the file contains a header row.
- The system shall allow users to specify whether the first column should be treated as an index.
- The system shall apply these configurations when parsing the uploaded file.

3.1.2.3 Data Preview and Validation

- The system shall display a preview of the parsed data (first 5 rows) in a tabular format.

- The system shall display the dimensions (rows and columns) of the dataset.
- The system shall detect and warn about missing values in the dataset.
- The system shall validate that the dataset contains the minimum required features for Parkinson's detection.

3.1.2.4 Data Statistics

- The system shall calculate and display basic statistics for numerical columns (mean, median, etc.).
- The system shall display the distribution of the target variable (Parkinson's diagnosis).
- The system shall identify and report potential data anomalies or quality issues.

3.2 Model Deployment

3.2.1 Description

The Model Deployment feature enables users to upload and configure pre-trained machine learning models for Parkinson's disease detection. These models will be used to analyze the uploaded patient data.

3.2.2 Functional Requirements

3.2.2.1 Model Upload

- The system shall allow users to upload up to three pre-trained machine learning models.
- The system shall support models saved in .pkl or .joblib formats.
- The system shall validate that uploaded models are compatible with scikit-learn.
- The system shall provide error messages for incompatible or corrupted model files.

3.2.2.2 Model Identification

- The system shall allow users to assign a name to each uploaded model.
- The system shall provide default names (Model 1, Model 2, Model 3) if the user does not specify a name.

- The system shall display the assigned names throughout the interface for consistency.

3.2.2.3 Model Configuration

- The system shall allow users to enable/disable detailed metrics for model evaluation.
- The system shall allow users to adjust visualization settings for model results.
- The system shall persist these configuration settings during the current session.

3.3 Analysis and Visualization

3.3.1 Description

The Analysis and Visualization feature processes patient data through the deployed models and generates comprehensive visual representations of the results, focusing on performance metrics and confusion matrices.

3.3.2 Functional Requirements

3.3.2.1 Data Processing

- The system shall preprocess the uploaded data to match the format expected by the models.
- The system shall scale/normalize features according to standard practices for Parkinson's detection.
- The system shall split the data into features and target variables.
- The system shall handle any necessary data transformations required by the models.

3.3.2.2 Model Execution

- The system shall pass the preprocessed data through each uploaded model.
- The system shall collect predictions from each model.
- The system shall execute this process in an optimized manner to minimize processing time.
- The system shall handle exceptions that may occur during model execution.

3.3.2.3 Performance Metrics

- The system shall calculate accuracy scores for each model.
- The system shall identify the best-performing model based on accuracy.
- The system shall visualize accuracy comparisons through bar charts.
- The system shall highlight the best-performing model in visualizations.

3.3.2.4 Confusion Matrix Generation

- The system shall generate confusion matrices for each model.
- The system shall appropriately label the confusion matrices for Parkinson's detection (No Parkinson's/Parkinson's).
- The system shall visualize confusion matrices using color-coded heatmaps.
- The system shall provide downloadable versions of the confusion matrix visualizations.

3.4 Comparative Analysis

3.4.1 Description

The Comparative Analysis feature enables detailed side-by-side comparison of model performance, helping users identify strengths and weaknesses of different approaches for Parkinson's disease detection.

3.4.2 Functional Requirements

3.4.2.1 Summary Comparison

- The system shall display a comparative summary of all model metrics in a single view.
- The system shall highlight key differences between model performances.
- The system shall provide observations about comparative strengths and weaknesses.

3.4.2.2 Detailed Comparison

- The system shall provide sample-by-sample comparison of model predictions.
- The system shall indicate correct/incorrect predictions for each model.
- The system shall support filtering and sorting of comparison results.

3.4.2.3 Agreement Analysis

- The system shall calculate and visualize the level of agreement between different models.
- The system shall identify cases where all models agree or disagree.
- The system shall analyze patterns in agreement/disagreement relative to true labels.

3.4.2.4 Result Export

- The system shall allow users to download comparative analysis results.
- The system shall support export of visualization images.
- The system shall provide options for exporting data in tabular formats.

---

4. External Interface Requirements

4.1 User Interfaces

4.1.1 General UI Requirements

- The UI shall follow a responsive design approach to adapt to different screen sizes.
- The UI shall use a professional color scheme appropriate for clinical settings.
- The UI shall organize content into logical sections with clear navigation.
- The UI shall provide visual feedback for user actions and system processing.

4.1.2 Main Dashboard

- The main dashboard shall display a header with the system title.
- The main dashboard shall include a sidebar for model configuration options.
- The main dashboard shall provide a central area for data upload and result display.

- The main dashboard shall include expandable sections for additional information.

4.1.3 Data Upload Interface

- The data upload interface shall include a file selection control.
- The data upload interface shall display file details after upload.
- The data upload interface shall provide controls for data parsing configuration.
- The data upload interface shall display a data preview after parsing.

4.1.4 Model Configuration Interface

- The model configuration interface shall include file upload controls for each model.
- The model configuration interface shall provide text input fields for model naming.
- The model configuration interface shall include toggle controls for display preferences.
- The model configuration interface shall provide feedback on model loading status.

4.1.5 Results Display Interface

- The results interface shall be organized into tabs for different types of analysis.
- The results interface shall display performance metrics in a visually appealing manner.
- The results interface shall present confusion matrices with appropriate labeling.
- The results interface shall include controls for downloading visualizations.

4.1.6 Help and Documentation

- The interface shall include expandable help sections with usage instructions.
- The interface shall provide tooltips for controls and visualizations.

- The interface shall include contextual information relevant to Parkinson's detection.

4.2 Hardware Interfaces

The system does not directly interface with specialized hardware beyond standard computing equipment. It operates on standard input devices (keyboard, mouse) and output devices (display).

4.3 Software Interfaces

4.3.1 Operating System Interface

- The system shall run on Windows, macOS, and Linux operating systems.
- The system shall utilize the host operating system's file system for file operations.
- The system shall adapt to the host operating system's window management system.

4.3.2 Browser Interface

- The system shall run in modern web browsers (Chrome, Firefox, Safari, Edge).
- The system shall utilize browser local storage for temporary session data.
- The system shall utilize browser rendering capabilities for displaying visualizations.

4.3.3 Python Runtime Interface

- The system shall run on Python 3.8 or higher.
- The system shall utilize the Python scientific stack for data processing and analysis.
- The system shall interface with scikit-learn for model loading and prediction.

4.3.4 File Format Interfaces

- The system shall read CSV files according to the CSV RFC 4180 specification.
- The system shall read TXT files with various delimiter options.

- The system shall load model files in .pkl format (via pickle) and .joblib format.

4.4 Communications Interfaces

The system operates primarily as a local web application and does not require external communication interfaces for its core functionality. All processing occurs within the local environment.

- The system shall utilize HTTP/HTTPS for communication between the browser and the local Streamlit server.
- The system shall not transmit data outside the local environment.

5. Non-functional Requirements

5.1 Performance Requirements

5.1.1 Response Time

- The system shall load and display the initial interface within 3 seconds.
- The system shall parse and display uploaded data (up to 10MB) within 5 seconds.
- The system shall load machine learning models within 3 seconds per model.
- The system shall process data and generate all visualizations within 10 seconds for datasets up to 1000 records.

5.1.2 Throughput

- The system shall support processing of datasets with up to 10,000 patient records.
- The system shall handle up to 100 concurrent file operations (upload, parse, analyze).

5.1.3 Capacity

- The system shall support uploading files up to 100MB in size.
- The system shall support model files up to 500MB each.

- The system shall efficiently manage memory to prevent crashes during large data processing.

5.1.4 Resource Utilization

- The system shall not consume more than 2GB of RAM during normal operation.
- The system shall not consume more than 70% of CPU resources during analysis tasks.
- The system shall release resources promptly after processing tasks complete.

5.2 Safety Requirements

5.2.1 Clinical Safety

- The system shall clearly indicate that it is a decision support tool, not a diagnostic device.
- The system shall include disclaimers about the interpretative nature of results.
- The system shall not make definitive diagnostic claims in any outputs.
- The system shall maintain audit trails of analysis operations.

5.2.2 Data Safety

- The system shall validate input data to prevent corruption during processing.
- The system shall not modify original input files under any circumstances.
- The system shall handle unexpected input gracefully without system failures.
- The system shall provide clear error messages that do not expose sensitive information.

5.3 Security Requirements

5.3.1 Data Privacy

- The system shall process all data locally without external transmission.

- The system shall not store uploaded patient data beyond the current session.
- The system shall not include identifying patient information in any exported results.
- The system shall clear all temporary data when the application is closed.

5.3.2 Access Control

- The system shall operate as a single-user application with no login requirements.
- The system shall rely on host system security for access control.

5.3.3 Data Integrity

- The system shall verify file integrity during upload and processing.
- The system shall ensure that analysis results correctly correspond to the input data.
- The system shall prevent injection attacks through input validation.

5.4 Software Quality Attributes

5.4.1 Reliability

- The system shall operate continuously without failure for 8-hour periods under normal use.
- The system shall gracefully handle unexpected inputs without crashing.
- The system shall provide meaningful error messages for exceptional conditions.
- The system shall maintain data integrity throughout all operations.

5.4.2 Availability

- The system shall be available whenever the host machine is operational.
- The system shall not require internet connectivity to function.
- The system's startup time shall not exceed 30 seconds.

5.4.3 Maintainability

- The system shall follow modular design principles for easy component updates.

- The system shall include comprehensive comments in the codebase.
- The system shall log operations and errors to facilitate troubleshooting.
- The system shall be structured to allow easy addition of new features or models.

5.4.4 Usability

- The system shall be usable by healthcare professionals with minimal technical training.
- The system shall provide clear guidance for all operations through the interface.
- The system shall use domain-appropriate terminology and visualizations.
- The system shall provide informative feedback during all operations.

5.4.5 Portability

- The system shall operate consistently across Windows, macOS, and Linux platforms.
- The system shall function across different browser environments.
- The system shall be installable with minimal configuration requirements.

5.5 Business Rules

5.5.1 Regulatory Compliance

- The system shall include appropriate medical software disclaimers.
- The system shall comply with relevant data protection regulations by design.
- The system shall maintain clear separation between analytical results and diagnostic conclusions.

5.5.2 Ethical Considerations

- The system shall present results in a manner that avoids bias in interpretation.
- The system shall clearly indicate confidence levels in predictions where applicable.

- The system shall not make recommendations outside the scope of its design parameters.

---

6. Other Requirements

6.1 Database Requirements

The system does not utilize a persistent database. All data is processed in memory during the user session and cleared upon application termination. This design choice supports data privacy and simplifies deployment.

For future versions that may incorporate persistent storage:

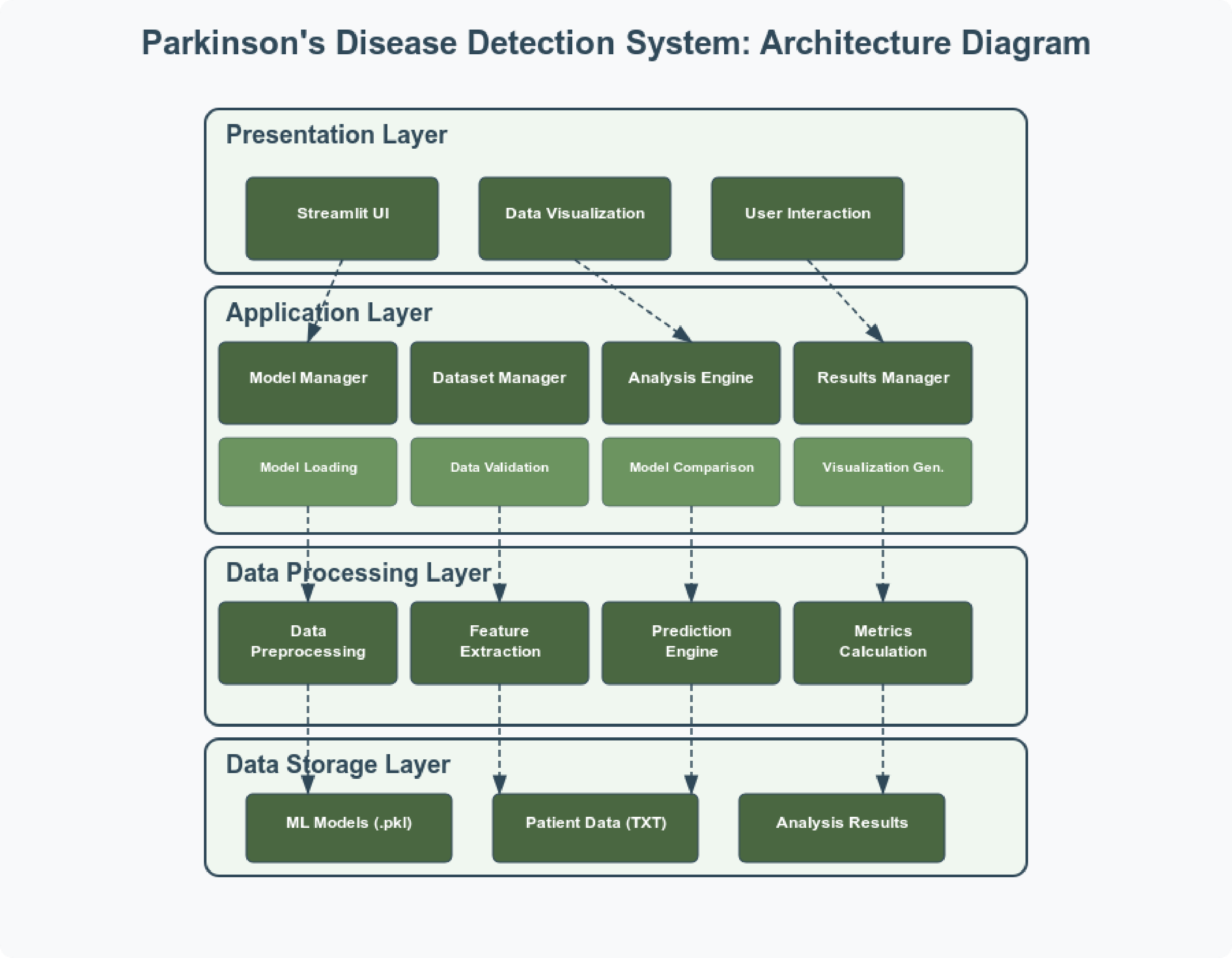
- Any database implementation shall encrypt patient data at rest.
- Database access shall be restricted to authorized users only.
- Database operations shall be logged for audit purposes.

6.2 System Architecture

The system follows a simple three-tier architecture:

1. **Presentation Layer**: Streamlit-based web interface
2. **Application Layer**: Python backend with data processing and model execution logic
3. **Data Layer**: Temporary in-memory data storage during user session

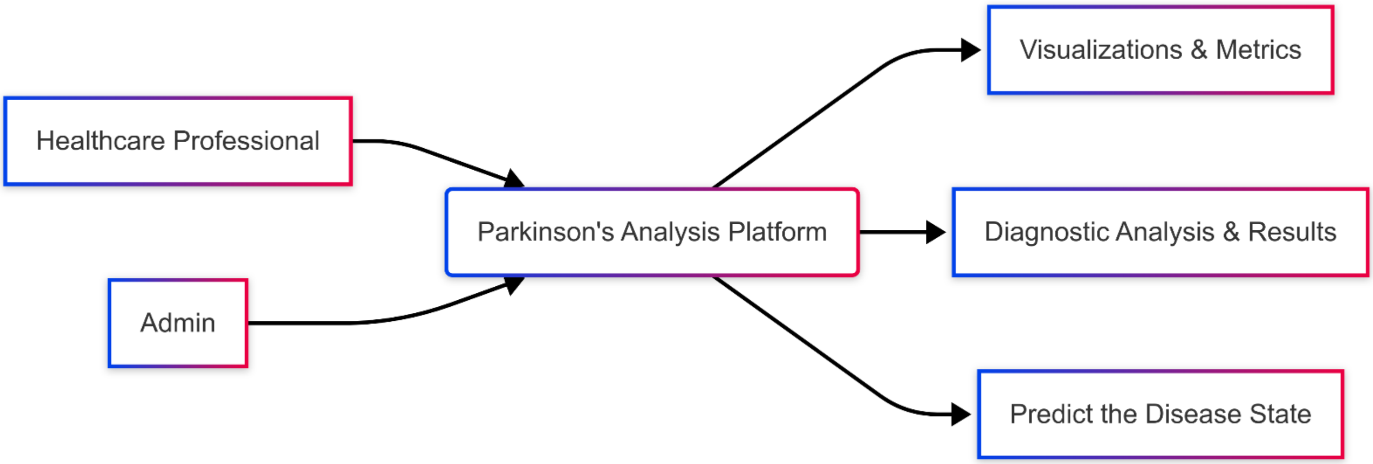
Architecture Diagram:



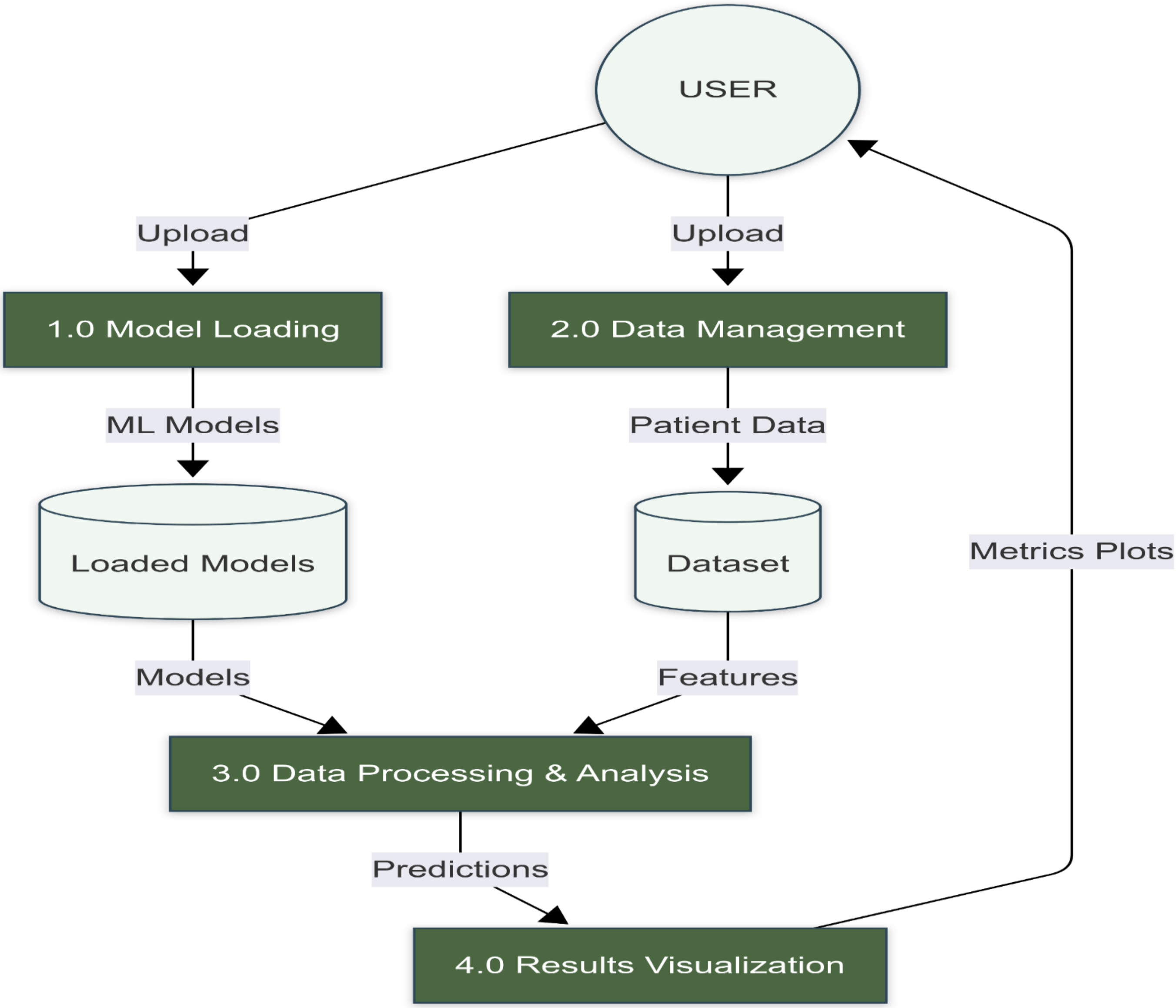
7. Appendix

7.1 Data Flow Diagrams

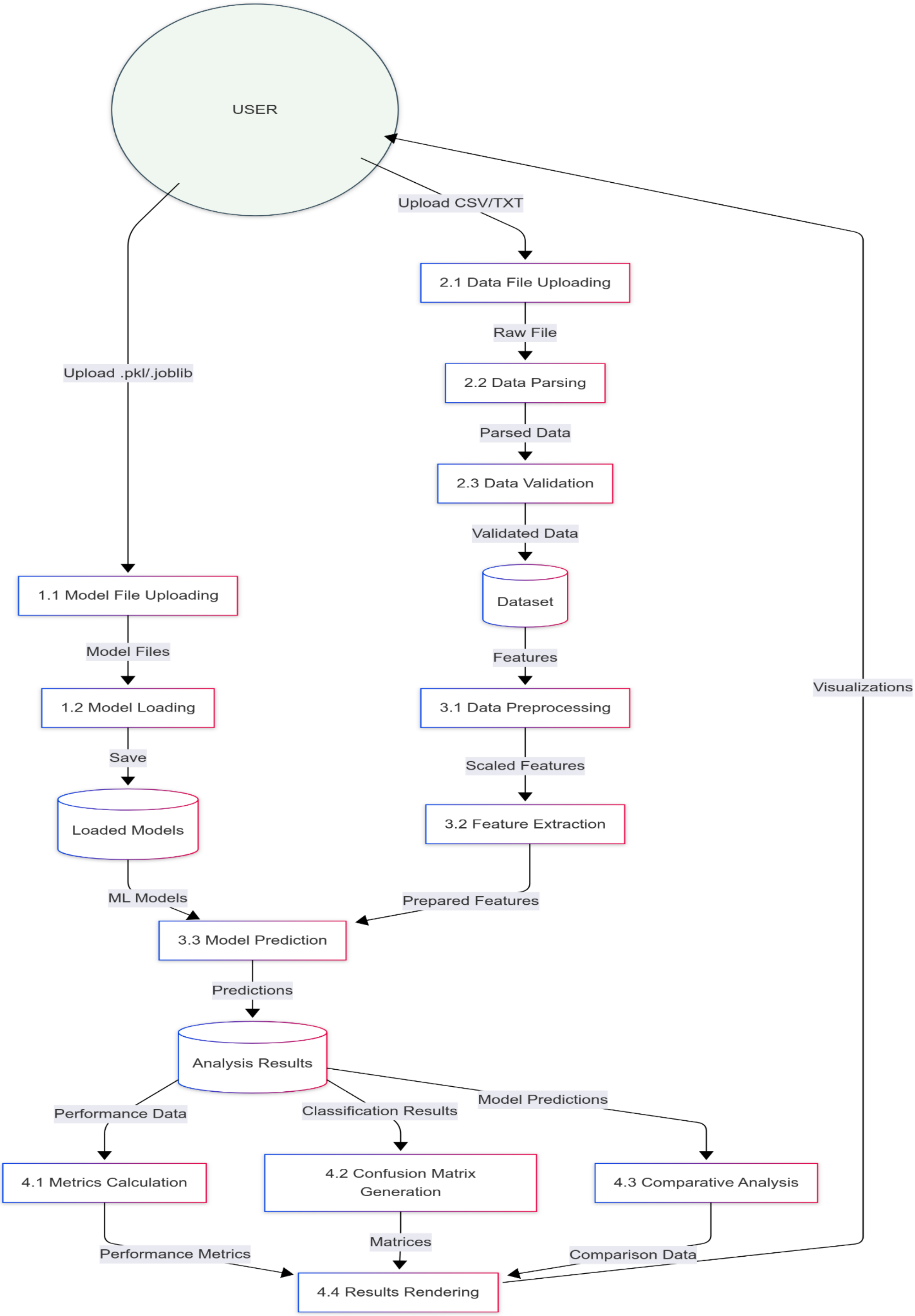
7.1.1 Level 0 DFD (Context Diagram)


Healthcare Professional
Admin
Parkinson's Analysis Platform
Visualizations & Metrics
Diagnostic Analysis & Results
Predict the Disease State

7.1.2 Level 1 DFD

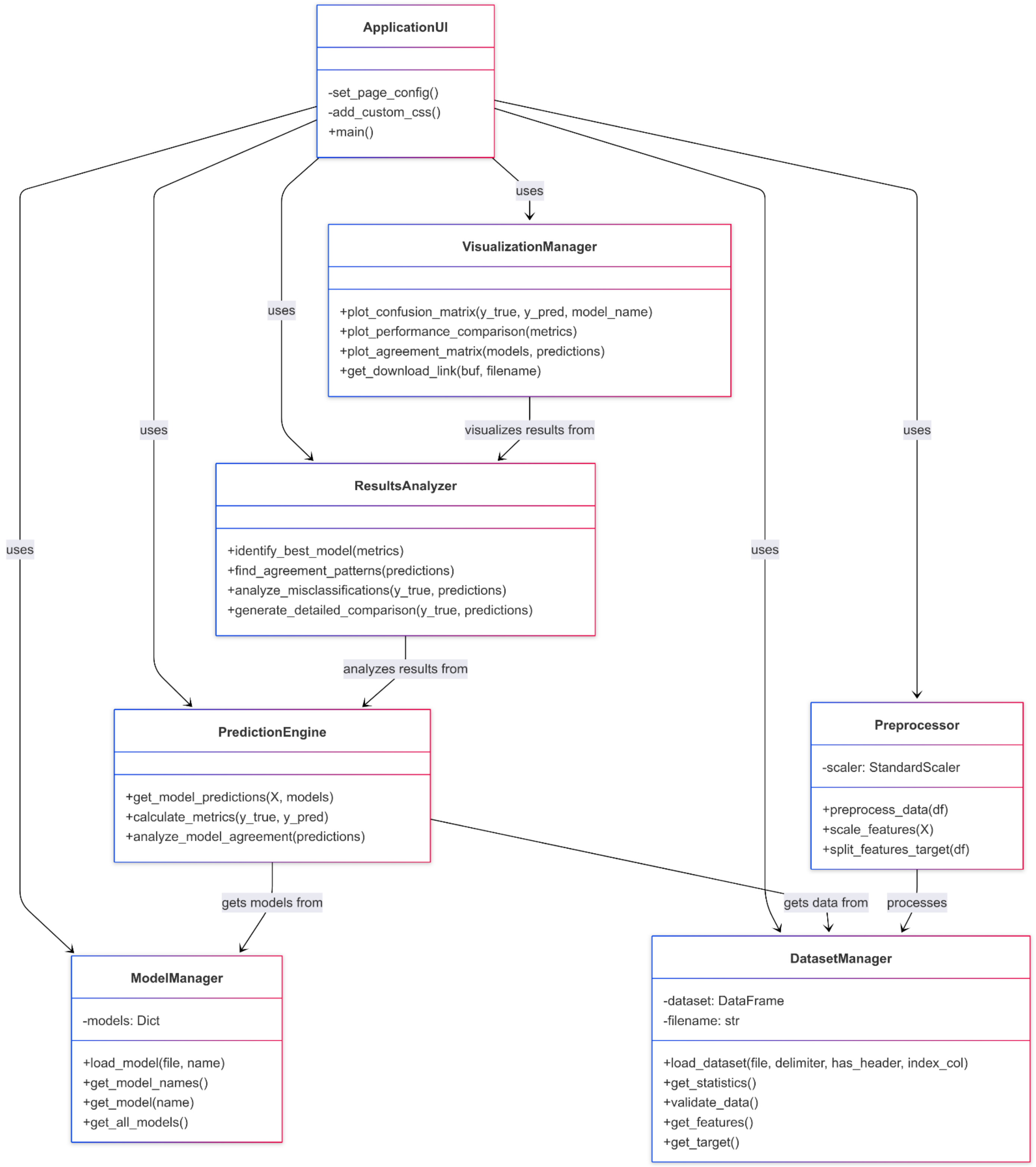


7.1.3 Level 2 DFD: Data Management Process (1.0)

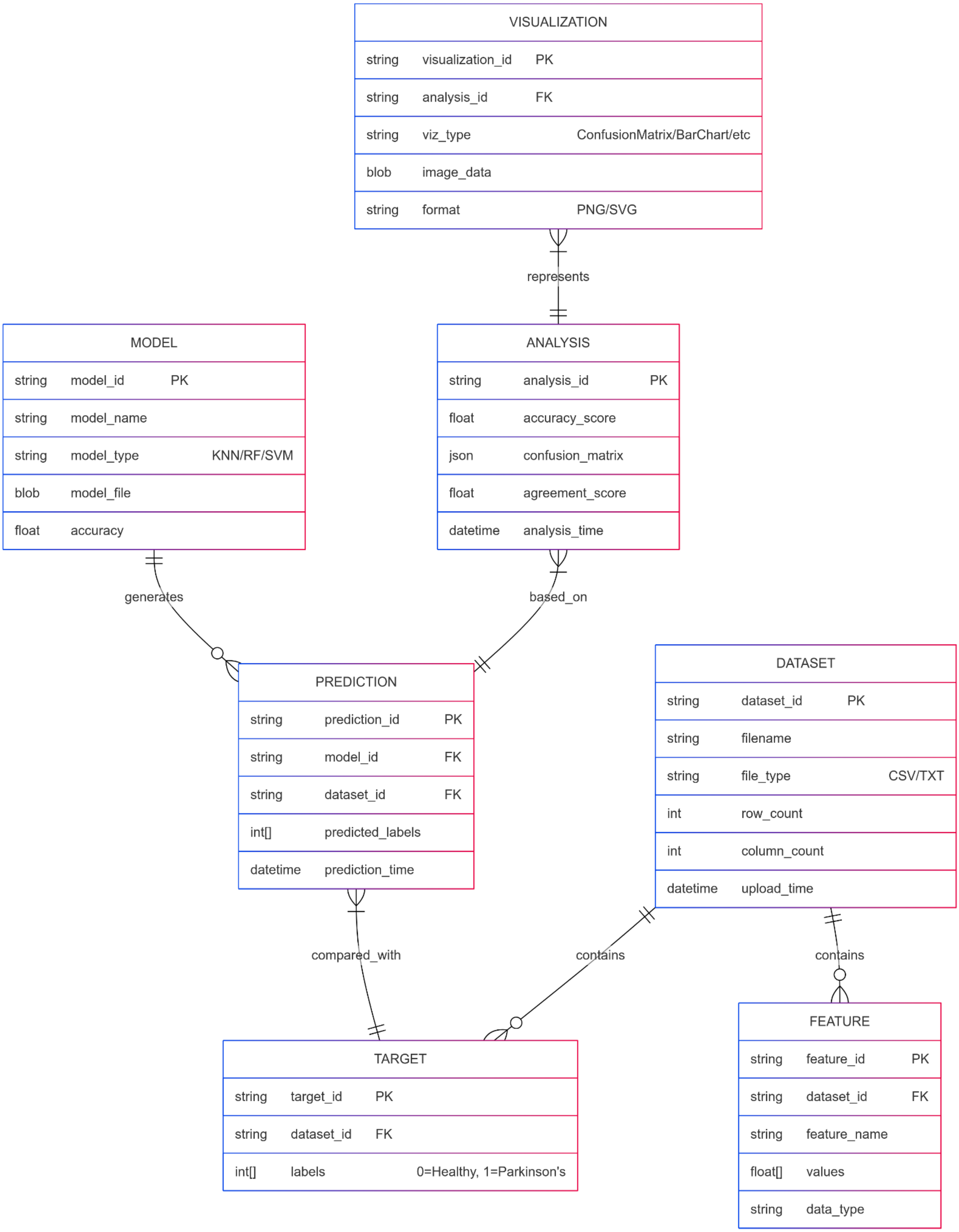


7.2 System Models

7.2.1 Class Diagram


ApplicationUI
-set_page_config()
-add_custom_css()
+main()
uses
VisualizationManager
+plot_confusion_matrix(y_true, y_pred, model_name)
+plot_performance_comparison(metrics)
+plot_agreement_matrix(models, predictions)
+get_download_link(buf, filename)
visualizes results from
ResultsAnalyzer
+identify_best_model(metrics)
+find_agreement_patterns(predictions)
+analyze_misclassifications(y_true, predictions)
+generate_detailed_comparison(y_true, predictions)
analyzes results from
PredictionEngine
+get_model_predictions(X, models)
+calculate_metrics(y_true, y_pred)
+analyze_model_agreement(predictions)
gets models from
ModelManager
-models: Dict
+load_model(file, name)
+get_model_names()
+get_model(name)
+get_all_models()
Preprocessor
-scaler: StandardScaler
+preprocess_data(df)
+scale_features(X)
+split_features_target(df)
gets data from
processes
DatasetManager
-dataset: DataFrame
-filename: str
+load_dataset(file, delimiter, has_header, index_col)
+get_statistics()
+validate_data()
+get_features()
+get_target()

7.2.2 Entity Relationship Diagram(ERD)



VISUALIZATION		
string	visualization_id	PK
string	analysis_id	FK
string	viz_type	ConfusionMatrix/BarChart/etc
blob	image_data	
string	format	PNG/SVG

MODEL		
string	model_id	PK
string	model_name	
string	model_type	KNN/RF/SVM
blob	model_file	
float	accuracy	

ANALYSIS		
string	analysis_id	PK
float	accuracy_score	
json	confusion_matrix	
float	agreement_score	
datetime	analysis_time	

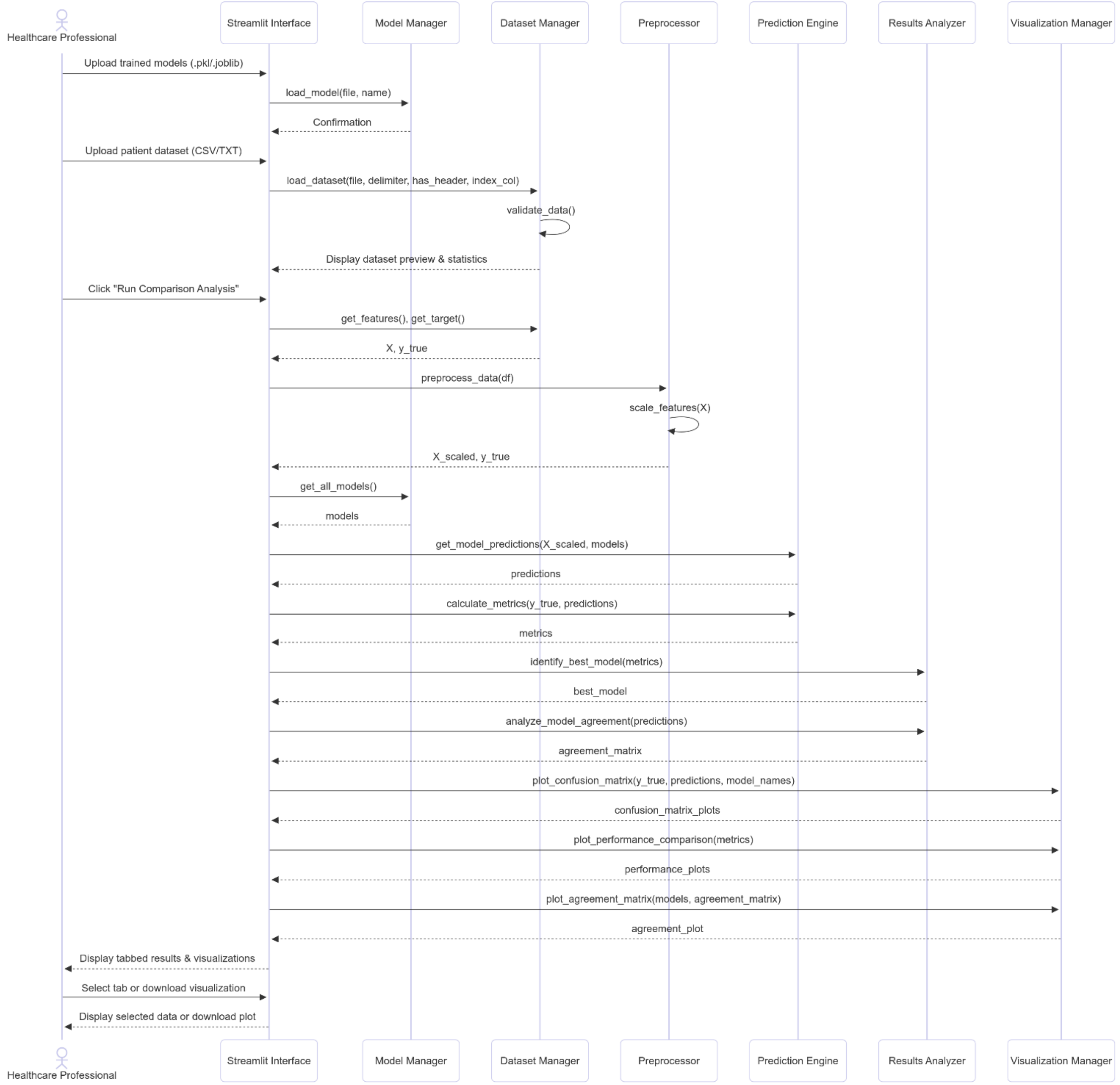
PREDICTION		
string	prediction_id	PK
string	model_id	FK
string	dataset_id	FK
int[]	predicted_labels	
datetime	prediction_time	

DATASET		
string	dataset_id	PK
string	filename	
string	file_type	CSV/TXT
int	row_count	
int	column_count	
datetime	upload_time	

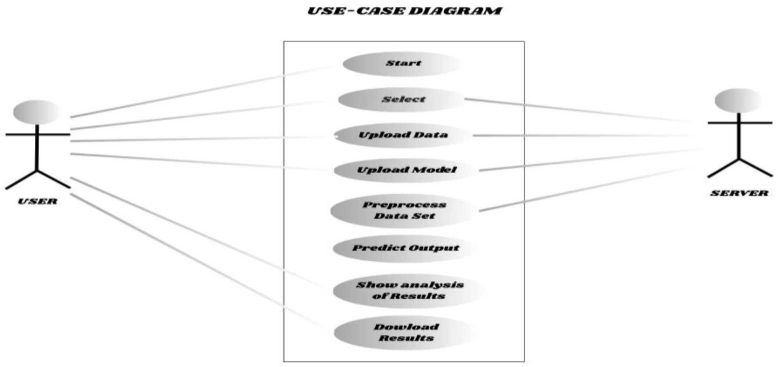
TARGET		
string	target_id	PK
string	dataset_id	FK
int[]	labels	0=Healthy, 1=Parkinson's

FEATURE		
string	feature_id	PK
string	dataset_id	FK
string	feature_name	
float[]	values	
string	data_type	

7.2.3 Sequence Diagram


Healthcare Professional
Streamlit Interface
Model Manager
Dataset Manager
Preprocessor
Prediction Engine
Results Analyzer
Visualization Manager
Upload trained models (.pkl/.joblib)
load_model(file, name)
Confirmation
Upload patient dataset (CSV/TXT)
load_dataset(file, delimiter, has_header, index_col)
validate_data()
Display dataset preview & statistics
Click "Run Comparison Analysis"
get_features(), get_target()
X, y_true
preprocess_data(df)
scale_features(X)
X_scaled, y_true
get_all_models()
models
get_model_predictions(X_scaled, models)
predictions
calculate_metrics(y_true, predictions)
metrics
identify_best_model(metrics)
best_model
analyze_model_agreement(predictions)
agreement_matrix
plot_confusion_matrix(y_true, predictions, model_names)
confusion_matrix_plots
plot_performance_comparison(metrics)
performance_plots
plot_agreement_matrix(models, agreement_matrix)
agreement_plot
Display tabbed results & visualizations
Select tab or download visualization
Display selected data or download plot
Healthcare Professional
Streamlit Interface
Model Manager
Dataset Manager
Preprocessor
Prediction Engine
Results Analyzer
Visualization Manager

7.2.4 Use Case Diagram



7.3 Document Revision History

Version: 1.0

Date: 4th May 2025

Description: 1st Draft with Full Stack working prototype

Author(s): Ritwik Mittal & Yashas Raina

x - End of Document - x