

SOFTWARE DESIGN DESCRIPTION

for

Data Pulse

Version 1.0

By

Syed Abdul Aleem (2020F-SE-042)
Shaheer Khan Qureshi (2020F-SE-003)
Abdul Moiz Chishti (2020F-SE-022) (GL)

Supervisor

Mr. Sarfaraz Abdul Sattar Natha

Bachelor of Science in Software Engineering (2023-2024)

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this project is to develop an end-to-end platform that automates the process of data profiling and machine learning model selection. By doing so, the project aims to simplify and streamline the utilization of data-driven insights, empowering users to effectively harness the potential of their datasets.

1.2 Scope

The project encompasses the development of an automated system that conducts comprehensive data profiling using advanced techniques. It will analyze datasets, compare multiple machine learning models, evaluate their performance metrics, and generate detailed reports outlining the comparison results. Additionally, the system will provide users with the option to download the trained model in a deployable format for seamless integration into various applications.

1.3 Project Background

In today's data-driven world, efficiently understanding and utilizing vast amounts of information is a significant challenge. Automated data profiling and machine learning offer solutions to extract valuable insights. However, selecting the right machine learning model can be daunting due to the multitude of available algorithms and the need for in-depth analysis. This project aims to address this challenge by developing an automated system that streamlines the process of data profiling and model selection.

1.4 Motivation

The motivation behind this project lies in the growing need for effective data analysis and model selection in a data-rich environment. By simplifying these processes and making them more accessible, the project seeks to empower users to make informed decisions and maximize the potential of their datasets. The goal is to democratize machine learning and enable broader adoption across various industries and applications.

1.5 Project Objective

The primary objective of the project is to develop an end-to-end platform that automates data profiling and machine learning model selection. This platform will utilize advanced techniques to explore and analyze datasets, compare various machine learning models, evaluate their performance metrics, and generate comprehensive reports. Additionally, the project aims to enable users to download the trained model in a deployable format, facilitating seamless integration into diverse applications. Ultimately, the project seeks to simplify data analysis and model selection, making the power of machine learning accessible to a wider audience.

2. Design Methodology and Software Process Model

2.1 Design Methodology

The design methodology for this project would likely follow an iterative and incremental approach, such as Agile or a hybrid Agile methodology. This is because the project involves developing an end-to-end platform that requires frequent feedback from stakeholders and users to ensure that the automated data profiling and machine learning model selection components meet their needs effectively. Additionally, an iterative approach allows for flexibility in incorporating new features, refining existing functionalities, and adapting to changing requirements throughout the development process.

2.2 Design Pattern (Creational, Structural, Behavioral) (Also give reasoning)

Creational Pattern: Factory Method Pattern

Reasoning: The Factory Method Pattern can be utilized to encapsulate the creation of machine learning model objects. This pattern allows the system to delegate the instantiation of specific machine learning models to subclasses, thereby promoting loose coupling between the client code and the concrete classes representing different algorithms. By employing this pattern, the system can easily incorporate new machine learning models in the future without modifying existing client code, enhancing scalability and maintainability.

Structural Pattern: Facade Pattern

Reasoning: The Facade Pattern can be applied to simplify the complexity of the automated data profiling and machine learning model selection process. By providing a unified interface, the Facade Pattern hides the intricacies of the underlying subsystems from the client, enabling users to interact with the platform in a straightforward manner. This pattern promotes modularity and encapsulation, allowing the system to manage the interactions between various components efficiently while presenting a cohesive and intuitive interface to users.

Behavioral Pattern: Strategy Pattern

Reasoning: The Strategy Pattern can be employed to facilitate the comparison of multiple machine learning models based on specific dataset characteristics. By encapsulating each comparison strategy (e.g., evaluation metrics, feature selection techniques) into separate classes, the system can dynamically switch between different strategies at runtime, thereby enabling flexibility and customization. This pattern promotes extensibility and maintainability, allowing the system to accommodate diverse requirements for model selection and evaluation.

2.3 Software Process Models (Classical, Modern, or Agile)

Agile
Reasoning: Agile methodologies, such as Scrum or Kanban, would be well-suited for this project due to their emphasis on iterative development, continuous feedback, and collaboration with stakeholders. Given the dynamic nature of data analysis and machine learning model selection, Agile practices allow the development team to adapt to changing requirements, incorporate user feedback, and deliver incremental improvements to the platform.

3. System Overview

3.1 Architectural Design (Client Server, Distributed, Cloud, ….)

- **Client-Server Model:**

    1. **Client Side**: A web application interface for users to upload datasets, initiate data profiling, select machine learning models for comparison, and download the optimal model.

    2. **Server Side**: A server or a cluster of servers that handle data processing, profiling, machine learning model training, and comparison. This setup involves local or server-based storage systems for managing data and models.

- **Local Storage System:**

    Data and models are stored in a local storage system managed by the server. This could involve file systems, databases, or dedicated data storage solutions that are accessible by the server.

- **Modular Architecture:**

    The system may still benefit from a modular or microservices approach for handling different tasks (e.g., data upload, profiling, model training) even without cloud services. This aids in maintaining scalability and manageability within a local or server-based environment.

- **Machine Learning Model Repository:**

    A cloud-based repository that stores various pre-trained machine learning models for quick comparison and selection.

3.2 Process Flow (Functional Requirement):

1. Data Upload and Storage:

Users upload their datasets through the web interface. The system stores this data in the local storage system.

2. Data Profiling:

The uploaded data undergoes profiling to assess quality, structure, and other characteristics important for model training and selection.

3. Model Training and Comparison:

The system trains multiple machine learning models with the profiled data, compares their performance, and identifies the most suitable models based on predefined criteria.

4. Report Generation and Model Selection:

Generates a report detailing the performance of each model, allowing users to make informed decisions on the best model for their needs.

5. Model Download:

Users can download the selected model for local deployment, ensuring ease of integration into various applications without the necessity for cloud services.

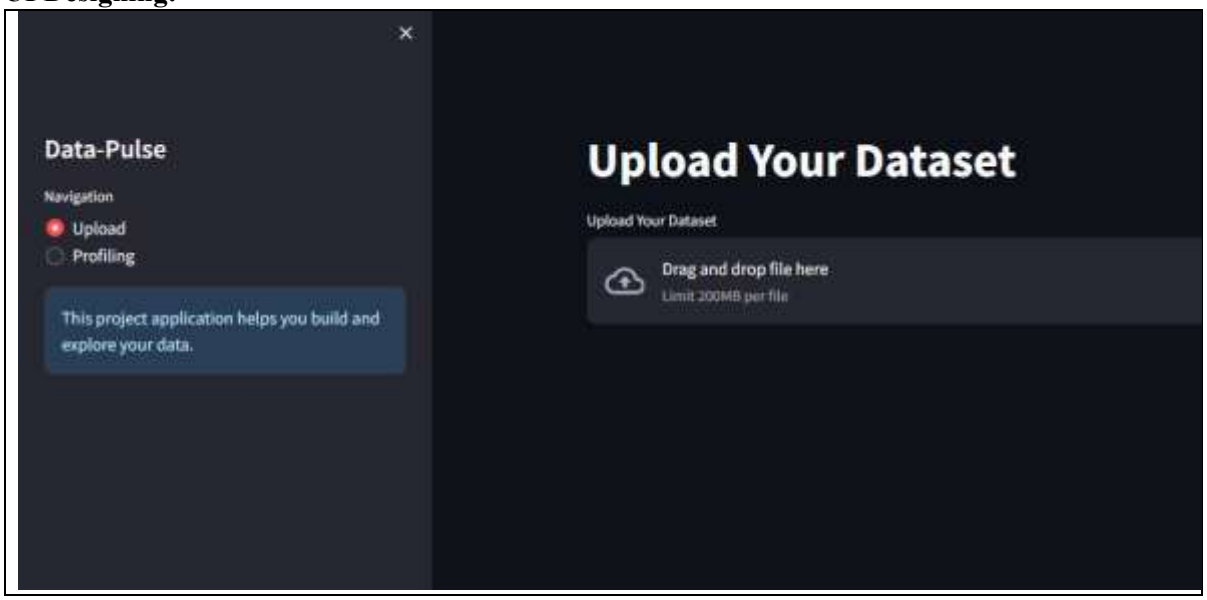
This architecture ensures that the system remains robust and scalable within a local or server-based environment, providing a comprehensive solution for automated data profiling and machine learning model comparison without relying on cloud storage.

4. Project Architecture

4.1 **Web/ Android module**

4.1.1 Functions (UI Designing – Figma, Canva) (UX based methodology):

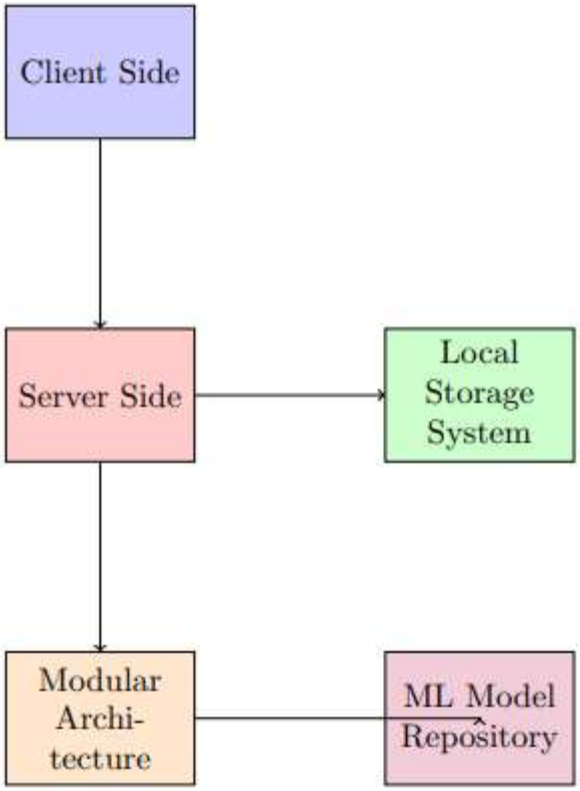
UI Designing:



UX Methodology:

The UX methodology for the project should prioritize user-centered design, focusing on understanding the needs, behaviors, and experiences of the users. This involves iterative design processes, including user research (surveys, interviews, user testing), creating personas, storyboarding, and journey mapping to identify key user interactions and pain points. Prototyping (using tools like Figma or Sketch) and usability testing should be conducted to refine the interface. The methodology should also incorporate accessibility standards and responsive design to ensure the product is usable across various devices and by all potential users. Feedback loops are essential, enabling continuous improvements based on user input and interaction data.

4.1.2 Architecture (Box and Line Diagram)



4.2 Database module (Justification), Type, DDL (Data Definition Table)

Justification:

A relational database is selected for structured data storage, efficient querying, and easy integration with data analysis tools. It supports complex queries and transactions, essential for managing the datasets, user information, and model details.

Type:

Relational Database Management System (RDBMS), such as PostgreSQL or MySQL, to manage structured data effectively.

DDL (Data Definition Table):

Users Table: Stores user account information.
Datasets Table: Contains details of uploaded datasets.
Models Table: Holds information about machine learning models, including performance metrics.
Reports Table: Stores generated reports for each dataset.

4.3 Administration module

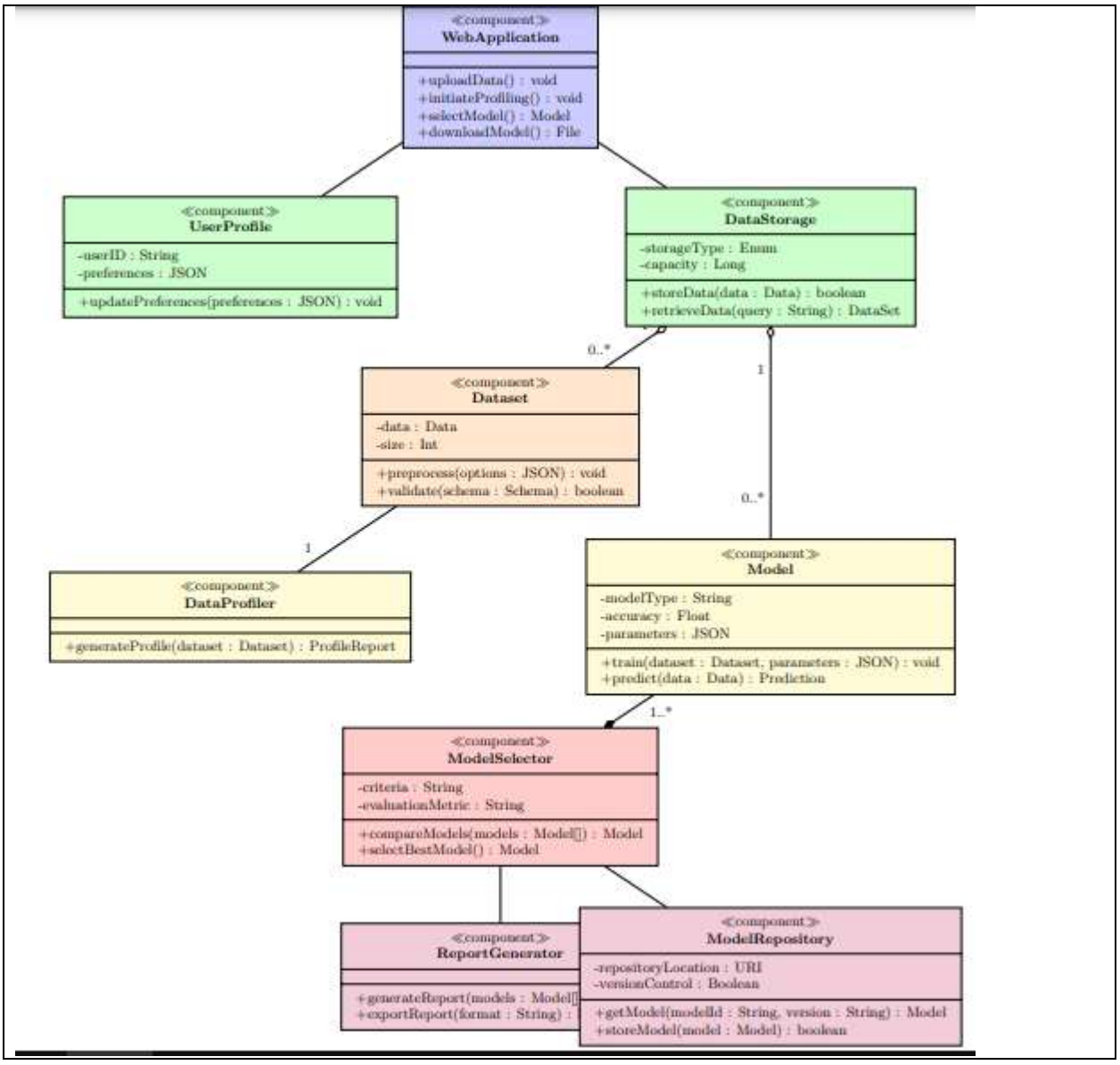
This module facilitates system management, user management, data oversight, and updates to machine learning models. It includes features for monitoring system health, managing access rights, and updating system components.

4.4 Data Dictionary

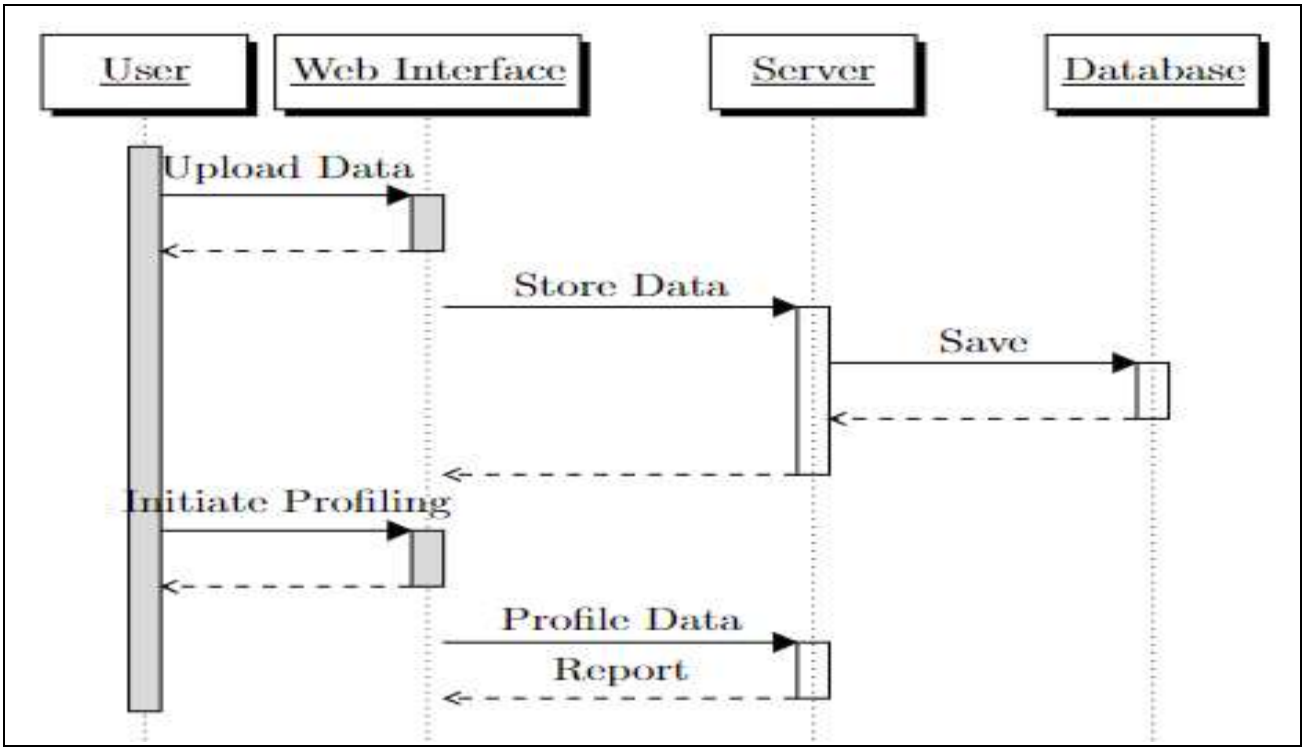
A comprehensive data dictionary would detail the structure of the database, defining each table and field, including data types, constraints, and descriptions of what they store (e.g., user IDs, dataset characteristics, model performance metrics). This dictionary ensures consistency and clarity in data management and system development.

5. System Analysis & Design Overview

5.1 Class Diagrams



5.2 System Sequence Diagrams



5.3 State Transition Diagram