High-Level Design (HLD)

Visa Approval Prediction System

Version1.0

Last date of revision: 07/06/2024

JENNA RAVEENKUMAR

# Document Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Date Issued | Version | Description | Author |
| 07th June 2024 | 1.0 | First Version of Complete HLD | Jenna Raveenkumar |
|  |  |  |  |

**Contents**

Table of Contents

**1 Introduction4**

1.1 Purpose4

1.2 Scope4

**2. General Description4**

2.1 Product Perspective4

2.2 Tools Used4

2.3 Constraints5

2.4 Assumptions5

**3. Design Details6**

3.1 Functional Architecture

3.2 Event Logs6

3.3 Error Handling6

3.4 Performance7

3.5 Reusability7

**4. Reference7**

# Introduction

**1.1** **Purpose**:

The main aim is to outline a high-level design for a Machine Learning-based system that predicts visa approval chances for applicants. This system will assist in expediting the visa certification process by identifying candidates with a higher probability of approval.

`` **1.2 Scope:**

The system will analyze historical visa application data to identify the key factors that influence visa approval. It is going to use a classification model to predict the likelihood of visa approval for new applicants. The scope includes data pre-processing, model training, evaluation, and deployment as a web application for easy access.

### **General Description**

#### **2.1 Product Perspective:**

#### The Visa Approval Prediction System is a web application designed to predict the likelihood of visa approval for applicants. This system aims to streamline the visa certification process by helping the Office of Foreign Labor Certification (OFLC) in identifying candidates with higher chances of approval, based on historical data. The system leverages machine learning models to analyze and predict outcomes based on past visa application data. The data is stored and managed in MongoDB databases,. MongoDB queries are used to retrieve, insert, delete, and update the database.

#### **2.2 Tools Used:**



**Programming Language**: Python is the primary language used for developing the system, including machine learning models and backend logic.

**Frameworks**:

**Numpy** **and** **Pandas**: For data manipulation and preprocessing.

**Scikit-learn**: For building and evaluating machine learning models.

**Evidently AI :** Data Monitoring & Data Drift

**Visualization**:

**Matplotlib**, **Seaborn**, **and** **Plotly** For creating visual representations of data and model performance metrics.

**Deployment**:

**AWS**: Used for deploying the web application and machine learning models, ensuring scalability and reliability.

**Database**:

**MongoDB/AWS S3 BUCKET**: Databases used for storing visa application data, prediction results, and models-related data stored in aws s3 bucket for faster deployment.

**Frontend Development**:

**HTML/CSS**: Used to create the user interface of the web application.

**Backend Development**:

**Python FastApi**: The framework used for developing the backend of the application, handling business logic, and serving the web pages.

**Version Control**:

**GitHub**: Used as the version control system for managing the codebase, tracking changes, and collaborating among team members.

#### **2.3 Constraints**

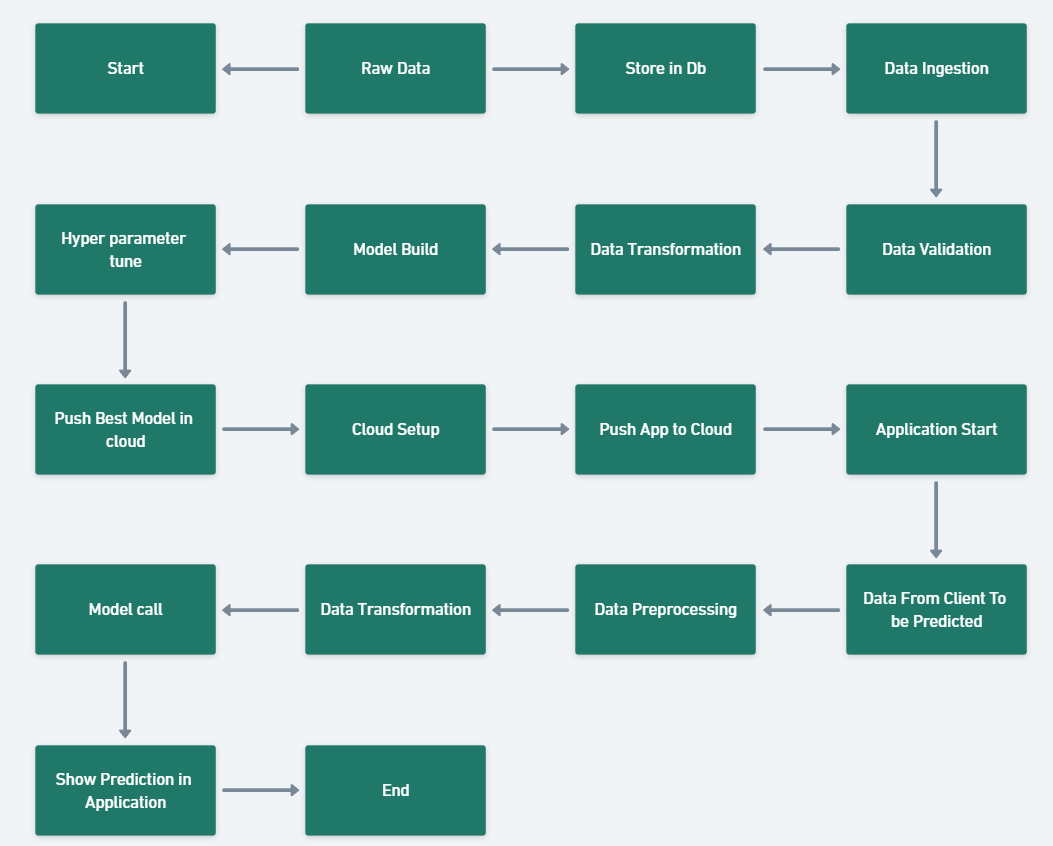
The Visa Approval Prediction System should be easy to use, as automated as possible, and not require users to have in-depth knowledge of its inner workings. It should provide real-time predictions.

**2.4 Assumptions**

The primary objective of the project is to predict the likelihood of visa approval for new applicants based on historical data using machine learning techniques. It is assumed that all components of the system, including data ingestion, model prediction, and user interaction, will work seamlessly together to provide accurate and timely predictions as expected.

**3 Design Details**

**3.1 Functional Architecture:**



**3.2 Event Log:**

The system must log every significant event to keep a detailed track of all processes and operations. This is crucial for both auditing and debugging purposes.

**Initial Step-By-Step Description**:

1. **Identify Logging Points**: The system must determine the key points at which logging is required.
2. **Comprehensive Logging**: The system should log all aspects of the system's flow, ensuring that every process is recorded.
3. **Logging Methods**: Developers can choose the logging method—either database logging or file logging.
4. **System Performance**: The system must remain responsive and should not hang due to extensive logging. Logging is mandatory to simplify issue debugging.

#### **3.3 Error Handling:**

The system should identify any errors encountered during operations and display a detailed explanation to the user regarding what went wrong. An error is defined as any condition that falls outside normal and intended usage.

**Error Handling Requirements**:

1. Capture and log all errors encountered.
2. Provide clear, user-friendly error messages.
3. Ensure the system can recover gracefully from errors when possible.

#### **3.4 Performance:**

The system's performance will be measured at every instance to ensure that the machine learning model remains effective over time. If the model's performance degrades, a retraining approach will be employed to maintain accuracy.

**Performance Considerations**:

1. Regular monitoring of model accuracy, precision, and other relevant metrics.

2. Use the threshold value of the f1 score to always store the best model in the S3 bucket.

3. Implement retraining procedures to restore performance..

#### **3.5 Reusability:**

The code and components developed for the system should be reusable in other projects with minimal modifications.

**Reusability Goals**:

1. Modular design to allow components to be easily reused.
2. Documentation to assist in the reuse of code and components.
3. Standardization of interfaces and protocols for integration.
4. **Reference**

1.https://www.kaggle.com/datasets/moro23/easyvisa-dataset