High-Level Design (HLD) Document

Spam Ham Classifier

**1. Introduction**

**1.1 Project Overview**

The **Spam Ham Classifier** is a machine learning application that classifies SMS messages as either "Spam" or "Not Spam." The classification is performed using a Naive Bayes Multinomial model. The application provides a user-friendly web interface deployed on Streamlit Cloud, allowing users to input text and receive real-time classification results.

**1.2 Objective**

The primary objective of this project is to develop an effective spam detection system using machine learning techniques. The system should be easy to use, provide accurate results, and be accessible online.

**2. System Architecture**

**2.1 High-Level Architecture**

The system architecture consists of the following components:

**1. User Interface (UI):**

- **Functionality**: Provides an interactive web interface for users to input SMS text and view classification results.

- **Technology**: Streamlit, hosted on Streamlit Cloud.

**2. Backend Processing:**

- **Functionality**: Handles text preprocessing, model prediction, and result generation.

- **Modules**:

Text Preprocessing: Tokenizes, stems, and cleans input text.

Model Prediction: Applies the Naive Bayes Multinomial model to classify the processed text.

- **Technology**: Python with NLTK for text processing, scikit-learn for machine learning.

**3. Machine Learning Model:**

- **Functionality**: Classifies SMS messages as spam or not spam.

- **Model Type**: Naive Bayes Multinomial.

- **Storage**: Model and vectorizer are stored in pickle files (`model.pkl`, `vectorizer.pkl`).

**4. Data Storage:**

- **Functionality**: Stores the dataset used for training and testing.

- **Technology**: No ongoing data storage required in the application; dataset used during model training only.

**5. Deployment Platform:**

- **Functionality**: Hosts the entire application and makes it accessible to users.

- **Platform**: Streamlit Cloud.

**6. Logging and Monitoring:**

- **Functionality**: Tracks application usage and errors for monitoring and debugging purposes.

- **Technology**: Streamlit Cloud's built-in logging features and Python’s logging library.

**2.2 Component Interaction**

- **User Interface (Streamlit App)**: Users input SMS text through the web interface. This input is passed to the backend processing modules.

- **Backend Processing**: The input text is preprocessed (tokenization, stemming, etc.) and then vectorized using the TF-IDF vectorizer.

- **Model Prediction**: The processed input is passed through the Naive Bayes model to predict whether it is spam or not spam.

- **Result Display**: The prediction result is sent back to the UI, where it is displayed to the user.

**3. Data Flow**

1. **User Input**: User enters SMS text into the Streamlit interface.

2. **Text Preprocessing**: The text is tokenized, lowercased, stripped of punctuation, and stemmed.

3. **Vectorization**: The processed text is transformed into a numerical format using the TF-IDF vectorizer.

4. **Model Prediction**: The vectorized text is classified as spam or not spam by the Naive Bayes model.

5. **Result Output**: The classification result is displayed on the Streamlit interface.

**4. Deployment Strategy**

**4.1 Hosting**

- **Platform**: Streamlit Cloud

- **Components**: Deploy the Streamlit app, including the Python scripts, model files (`model.pkl`, `vectorizer.pkl`), and any dependencies.

**4.2 Monitoring**

- **Tools**: Utilize Streamlit Cloud’s built-in monitoring features and Python logging to track app performance, errors, and usage patterns.

**4.3 Security**

- **Data Security**: Ensure secure handling of any input data and compliance with privacy regulations.

- **Model Security**: Protect the model and vectorizer files from unauthorized access.

**5. Security Considerations**

- **User Data Privacy**: No personal data is stored; input text is used only for real-time classification.

- **Secure Deployment**: The app is deployed on Streamlit Cloud, which provides HTTPS by default, ensuring encrypted data transmission.

**6. Conclusion**

This High-Level Design document outlines the architecture, key components, and interactions of the **Spam Ham Classifier** system. The design is focused on providing an efficient, secure, and user-friendly platform for SMS classification.