Facebook Post Status Prediction

Low Level Design

Domain: Networking

Creator: Vinayaka Uppar

Date: 10.01.2024

Document Version Control

Date issued	Version	Description	Author
DEC 22, 2023	1.1	First Draft	Vinayaka.uppar
JAN 02, 2024	1.2	Added the models	Vinayaka.uppar

Contents

Introduction

What is Low-Level Design Document?

A Low-Level Design (LLD) Document in software development is a comprehensive and detailed guide that provides an in-depth view of the technical aspects and internal workings of a system. It goes beyond the high-level architecture presented in High-Level Design (HLD) and breaks down the system into smaller, manageable components. In the context of the "Facebook Post Status Prediction" project, the LLD would elaborate on how each module functions, the specific algorithms used for sentiment analysis and emotion prediction, the detailed structure of the database, and the interactions between different components. It acts as a roadmap for developers, offering a clear understanding of the system's intricacies and facilitating the smooth implementation of the project.

Scope

The scope of the "Facebook Post Status Prediction" project encompasses the development of a robust and proactive system for identifying the emotional quotient in young individuals through analysis of their Facebook posts. It includes the implementation of advanced natural language processing (NLP) and machine learning techniques to predict emotional states accurately. The project's reach extends to creating a user-friendly interface for effective information dissemination to parents, teachers, and mental health professionals.

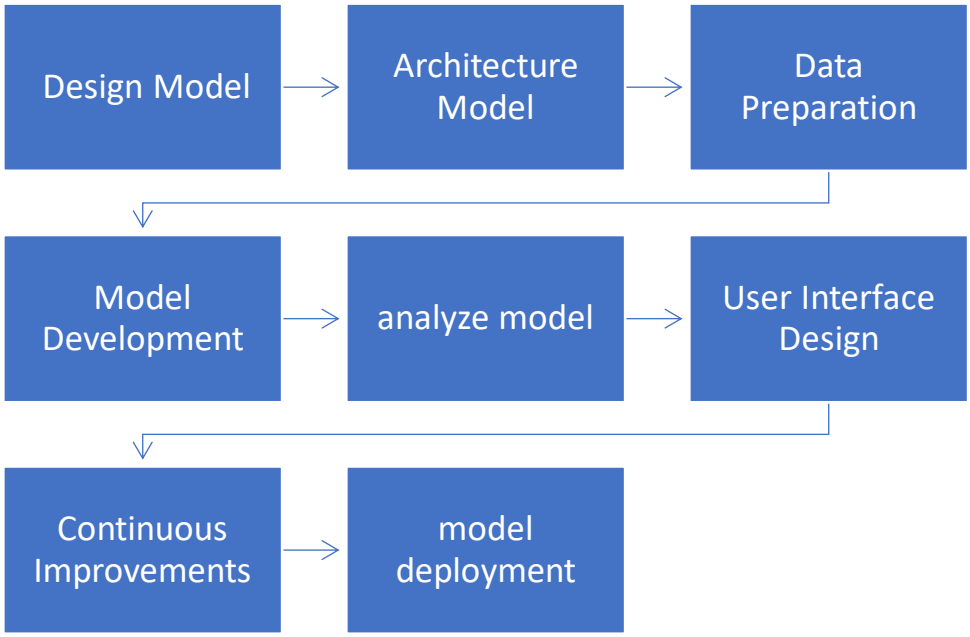
The scope also covers the collection and preprocessing of a diverse dataset of Facebook posts from young users, allowing the training and fine-tuning of machine learning models. The system aims to provide timely insights into the emotional states of young individuals during adolescence, facilitating early intervention and support systems. Overall, the scope is focused on leveraging social media data to contribute to the early detection and intervention of mental health challenges faced by teenagers.

Architecture

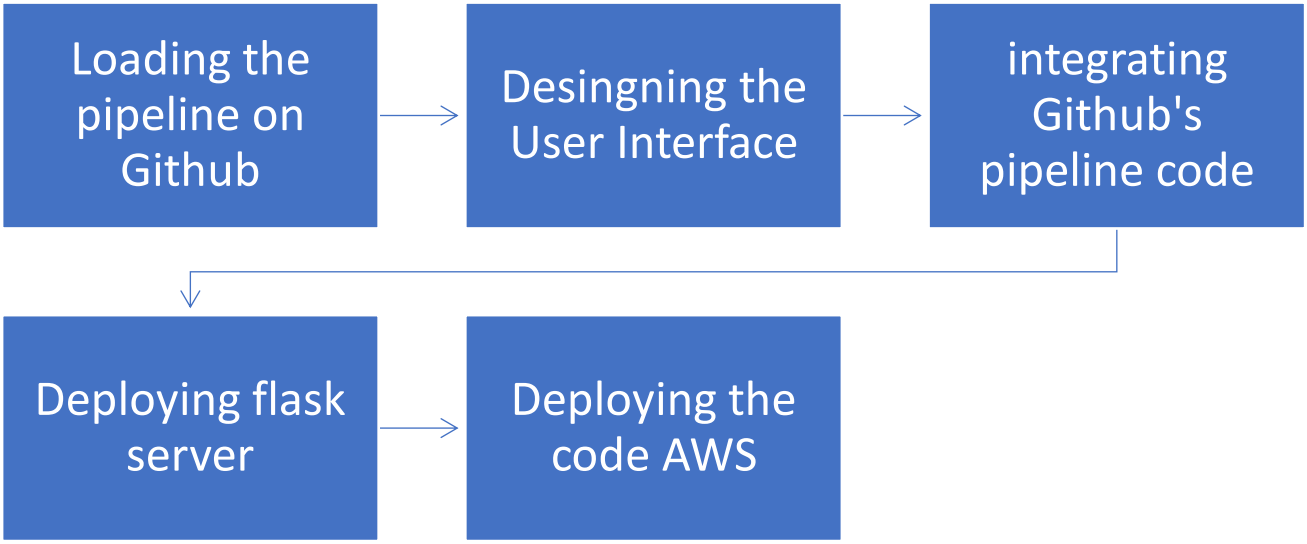
Data Preparation

Model Development

Deployment



Deployment



Architecture Description

Data Preparation

Data Description

Data Preprocessing

Data preprocessing in the "Facebook Post Status Prediction" project is a crucial phase that involves refining the raw Facebook post data to make it suitable for detailed analysis and machine learning model training.

In the initial step of text cleaning, irrelevant elements like special characters, HTML tags, and emojis are removed to ensure that the focus remains on the actual content of the posts. Following this, tokenization breaks down the text into individual words or tokens, providing a structured representation.

Stop word removal eliminates commonly occurring but less informative words, reducing noise and enhancing the accuracy of sentiment analysis. Stemming or lemmatization

Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a crucial phase in the "Facebook Post Status Prediction" project, focusing on understanding and extracting insights from the dataset before model development. This involves examining descriptive statistics to understand central tendencies and dispersion, visualizing the distribution of emotional states to identify potential imbalances, and analyzing word frequency to uncover prevalent themes. Correlation analysis helps identify relationships between features, while visualization techniques provide graphical representations of patterns and trends. Outlier detection ensures data integrity, and assessing feature importance guides the selection of relevant variables for model training. EDA is integral for informed decision-making, contributing to the overall effectiveness of the project.

Feature Engineering

Feature engineering in the context of the "Facebook Post Status Prediction" project involves creating new, meaningful features from existing ones to enhance the predictive power of the machine learning model. This process aims to extract relevant information, improve model performance, and contribute to accurate emotion prediction.

Model Development

Model implementation

Model implementation in the "Facebook Post Status Prediction" project involves translating the chosen machine learning algorithms and techniques into functional code. The implementation process can be broken down into several key steps:

Algorithm Selection: Choose appropriate machine learning algorithms based on the nature of the problem. For emotion prediction in text data, algorithms like Support Vector Machines (SVM), Random Forest, or neural networks may be considered.

Data Splitting: Divide the preprocessed dataset into training and testing sets. The training set is used to train the model, while the testing set evaluates its performance on unseen data.

Feature Scaling: Standardize or normalize the input features to ensure that they are on a similar scale. This is crucial for algorithms sensitive to feature magnitudes, such as SVM or neural networks.

Model Training: Train the selected machine learning model on the training dataset

Hyper-parameter Tuning

Optimize the model's hyperparameters to achieve the best performance. Techniques like grid search or random search can be employed to find the optimal combination of hyperparameters.

Model Evaluation

Assess the model's performance on the testing set using metrics such as accuracy, precision, recall, and F1-score. This step ensures that the model generalizes well to new, unseen data.

Deployment

Designing UI

Designing the User Interface (UI) for the "Facebook Post Status Prediction" project involves creating a visually appealing and user-friendly platform to facilitate effective communication of emotional insights.

Designing a server

Designing a server for the "Facebook Post Status Prediction" project involves creating a robust and scalable infrastructure to support the backend processes, handle user requests, and ensure the efficient functioning of the system. Here's an explanation of the server design process

Code deployment on cloud

Deploying code on the cloud for the "Facebook Post Status Prediction" project involves using cloud services to host and manage the application. Here's an explanation of the code deployment process on a cloud platform, using Amazon Web Services (AWS)

Deployment Process

The deployment process for the "Facebook Post Status Prediction" project is a carefully orchestrated series of steps aimed at seamlessly transitioning the application from the development environment to a live, operational state.

Firstly, meticulous preparation ensures that the application code is thoroughly tested and all dependencies are accurately specified for the intended production environment. Configuration files or environment variables are set up to tailor the application to the production environment's specific requirements.

If applicable, database migration is executed to synchronize the production database schema with the latest code changes. This is followed by the collection and organization of static files, ensuring that all necessary assets are available for production use.

Unit cases

Test Case Description	Pre-Requisite	Expected Result

Text-to-Audio Conversion	Application is accessible	The engine should successfully convert text to audio, and the user should hear the expected output
Confirm that speech recognition with speechrecognition accurately transcribes spoken words to text.	Application is accessible	The system should correctly interpret and display the transcribed text from the provided speech. Wikipedia Search
Validate the webbrowser library's functionality to open web pages in a browser	Application is accessible	The system should successfully launch the default web browser and open the specified webpage. Random Number Generation
Ensure that the random module generates random numbers effectively	Application is accessible	The system should produce random numbers with a uniform distribution
Verify that the pyautogui library captures screenshots as intended	Application is accessible	The system should take accurate screenshots of the current screen. API Requests and Data Parsing
Verify that the socket module enables communication over a network.	Application is accessible	the system should establish and maintain network communication successfully
Validate the qrcode library's ability to generate QR codes programmatically	Application is accessible	The system should create QR codes accurately based on provided input