Project Design Phase-II

Technology Stack (Architecture & Stack)

Date	02 Nov 2023
Team ID	NM2023TMID02288
Project Name	Creating An Email Campaign In MailChimp

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

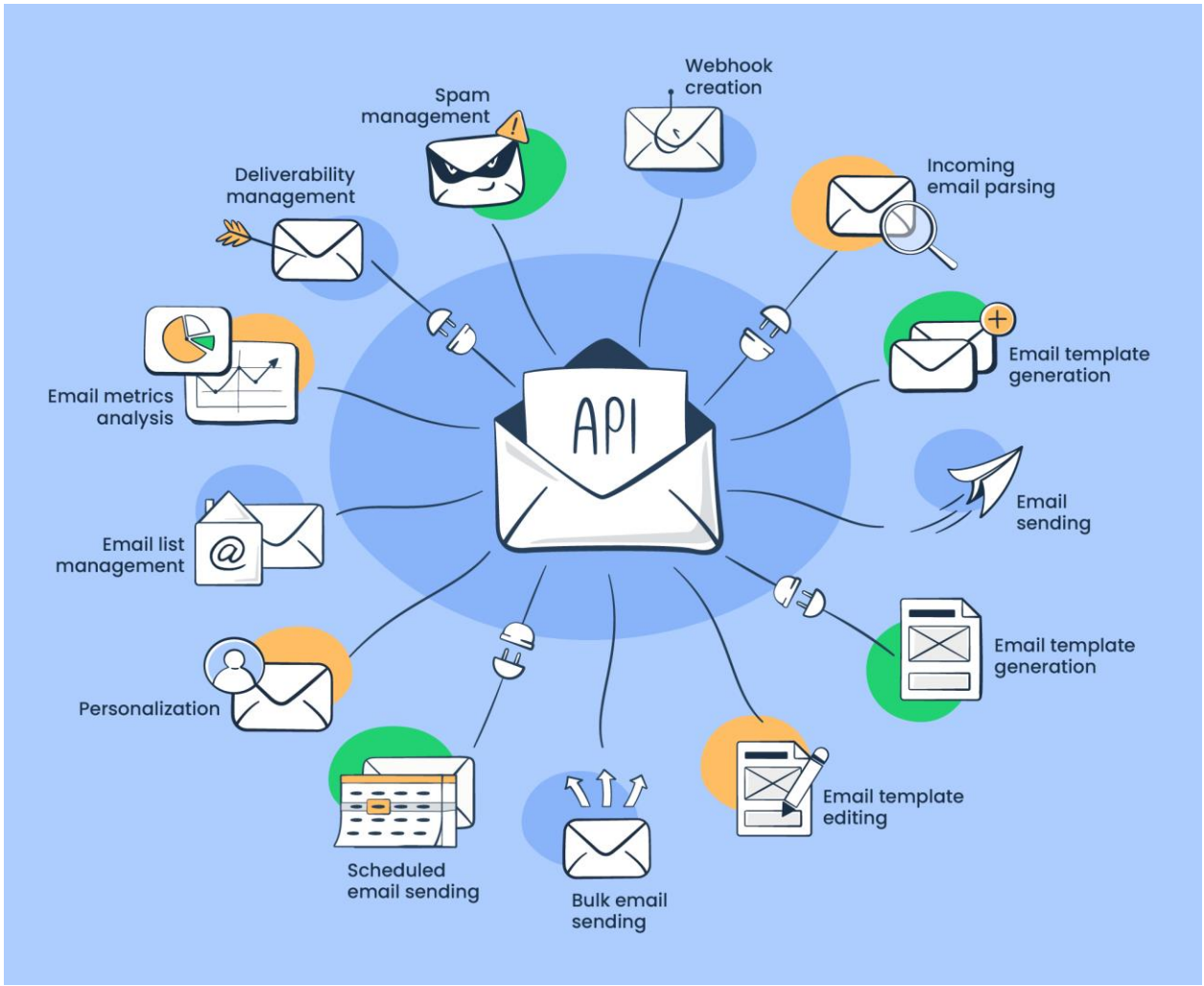

Table-1: Components & Technologies:

S. No	Component	Description	Technology
01.	User interface	Developing a user-friendly interface where users can interact with the system.	HTML, CSS, JavaScript, Python Web Framework
02.	Dataset collection	Gathering a comprehensive dataset of eye images containing examples of Normal, Cataract, Diabetic Retinopathy, and Glaucoma cases.	Web scraping, Openaccess datasets, Data annotation tools
03.	Preprocessing	Preprocessing and preparing the collected eye images for input into the deep learning models.	Python, OpenCV, PIL (Python Imaging Library)
04.	Deep learning model	Utilizing deep learning models for image analysis and classification.	Python, TensorFlow, Keras, PyTorch
05.	Model Training	Training the deep learning models using the prepared dataset.	Python, TensorFlow, Keras, PyTorch

06.	Model Evaluation	Assessing the performance of the trained models using various evaluation metrics.	Python, scikit-learn, TensorFlow, Keras, PyTorch
07.	Deployment	Integrating the trained model into the user interface where users can upload eye images and obtain predictions for the corresponding eye disease category.	Python-Flask, Python Web

Table-2: Application Characteristics:

S. No	Characteristics	Description	Technology
01.	Open-source frameworks	Open-source frameworks provide a foundation for building applications and come with pre-built components, libraries, and tools.	TensorFlow, Keras, PyTorch, Flask, Django, scikit-learn
02.	Security implementations	Security measures to protect the application and its data from unauthorized access, breaches, or attacks.	User authentication mechanisms, security frameworks like OWASP

03.	Scalable architecture	Design that allows the application to handle increasing workloads and accommodate growth without sacrificing performance.	Microservices architecture, Docker, Kubernetes
04.	Availability	Ensuring the application remains accessible and functional, minimizing downtime or interruptions.	APIs for data access, data collection frameworks like Apache Nutch or Scrapy
05.	Performance	Optimizing the application for fast response times, handling large numbers of requests, and efficient resource utilization.	Identify bottlenecks, performance optimization