Project Title:

AI Personalized Email Generator: leveraging user data to craft customized emails efficiently.

Team Name:

Brute Force

Team Members:

- G ANIKETH (22B81A0505)
- K PRANEETH (22B81A0548)
- A SRIDHAR (22B81A0549)
- S VISHNU VARDHAN (22B81A0563)

Phase-1: Brainstorming & Ideation

Objective:

- To develop an AI-driven email generator using the MERN stack that personalizes content based on user data, optimizing engagement, automation, and efficiency for businesses and individuals.

Key Points:

1. **Problem Statement:**

   Traditional email writing is time-consuming and lacks personalization, leading to low engagement.AI Personalized Email Generator is an innovative project designed to revolutionize email communication by automating the process of drafting personalized emails.

2. **Proposed Solution:**

The AI Personalized Email Generator streamlines the email drafting process by allowing users to input specific details such as the recipient's name, event name, and special instructions. It then generates a customized email body, saving users time and effort in composing individualized messages

3. **Target Users:**

- **Businesses** for automated marketing emails
- **Freelancers** for client communication
- **HR** teams for personalized recruitment emails
- **Customer support** for quick responses
- **Individuals** for efficient personal email writing

4. **Expected Outcome:**

AI-powered email generator that efficiently creates personalized emails, enhancing engagement and automation. Users will experience a significant reduction in time spent drafting emails while maintaining a high level of customization. The system will seamlessly integrate with existing email platforms, ensuring smooth workflow integration.

Phase-2: Requirement Analysis

Objective: The project requires React.js for the frontend, Node.js with Express for the backend, and MongoDB for data storage. AI integration (OpenAI API) enables personalized email generation. JWT secures authentication, while SendGrid handles email delivery. Users can register, manage templates, schedule emails, and track analytics. The system allows customization of AI-generated emails, improving engagement and efficiency while integrating seamlessly with third-party email platforms for enhanced functionality.

- **Key Points:**

Technical Requirements:

- **Frontend:** React.js for user interface
- **Backend:** Node.js with Express for API handling
- **Database:** MongoDB for storing user data and email templates
- **AI Integration:** OpenAI API or a custom NLP model for email generation
- **Authentication:** JWT-based user authentication

Functional Requirements:

- **User Registration/Login:** Secure authentication system
- **Email Personalization:** AI-generated emails based on user input
- **Template Management:** Save, edit, and reuse email templates
- **Email Scheduling:** Option to send emails at a later time
- **Integration Support:** Connect with third-party email services

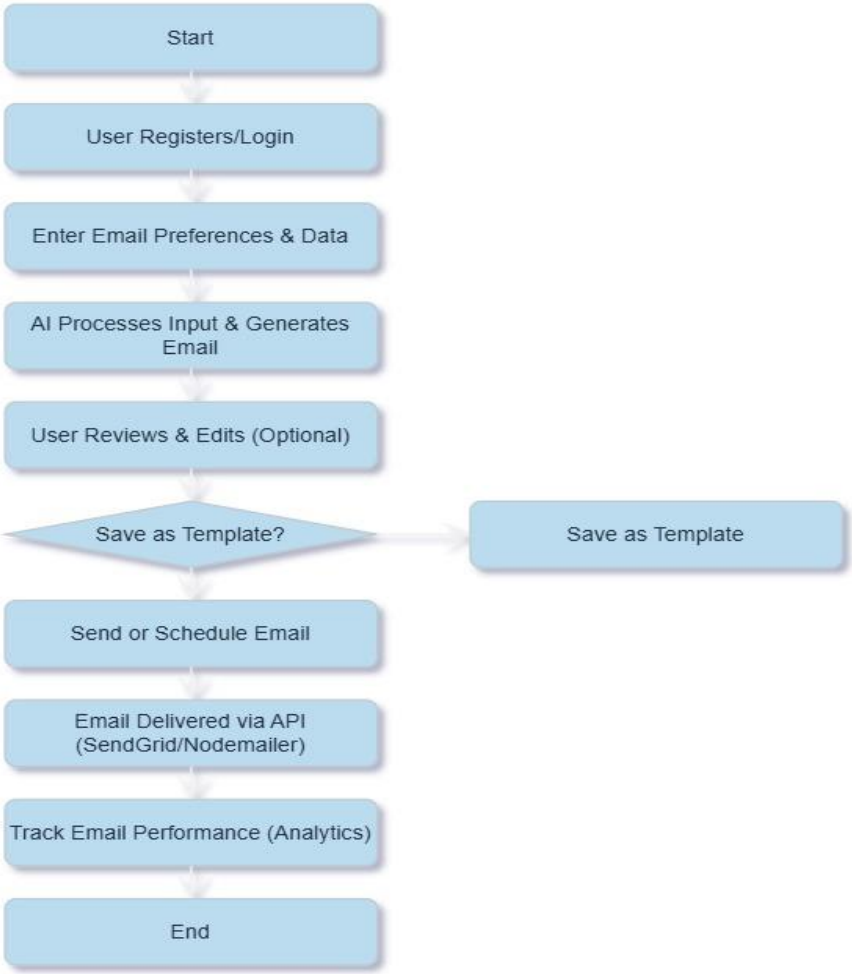
Phase-3: Project Design

Objective:

The **AI Personal Email Generator** aims to streamline email composition by leveraging artificial intelligence to generate context-aware, personalized emails efficiently. The project focuses on enhancing productivity by automating email writing while maintaining a human-like tone, structure, and intent.

Key Points:

1. **System Architecture Diagram:**



2. **User Flow:**

- **User Login/Signup** – Access via email, Google, or social login.
- **Email Creation** – User selects type, inputs details, and chooses tone.

- **AI Generation** – AI processes input and generates a draft email.
- **User Review & Edit** – User can edit, regenerate, or finalize the email.
- **Send or Save** – User copies, sends via linked email, or saves as a draft.

Phase-4: Project Planning (Agile Methodologies)

Objective:

- **Scrum Framework:** Short sprints (1-2 weeks) for iterative development and frequent feedback.
- **Daily Standups:** Quick meetings to track progress, discuss blockers, and plan tasks.
- **Backlog Management:** Prioritized task list for features like AI integration, email automation, and UI improvements.
- **Continuous Integration & Deployment (CI/CD):** Regular updates to ensure smooth functionality.
- **User Feedback Loop:** Regular testing and feedback collection for improvements.
- **Cross-functional Collaboration:** Developers, designers, and testers work together for faster delivery.

Key Points:

Sprint Planning:

1. **Project Setup & Authentication:**
    - Set up MERN stack project
    - Configure MongoDB database
    - Implement user authentication (JWT)
    - Create login/signup UI in React
2. **AI Email Generation Core:**
    - integrate OpenAI API for email generation
    - Create backend route to handle AI requests
    - Design frontend input form for email preferences

3 Email Sending & Scheduling:

- Integrate SendGrid for email delivery
- Implement scheduling feature for emails
- Store sent/scheduled emails in MongoDB
- Add email status tracking in the backend

4 Template Management:

- Allow users to save/edit/delete email templates
- Create UI for template selection
- Backend routes for template management
- Store templates in MongoDB

Task Allocation:

1 Aniketh (frontend and backend):

- Develop React.js UI (login, email input, preview)
- Implement authentication system (JWT-based login/signup)
- Build Node.js API for authentication, email generation, and templates

2 Sridhar (Backend & Database):

- Set up Node.js & Express backend
- Implement database schema & MongoDB operations
- Develop email sending & scheduling

3 Praneeth (AI & DevOps):

- Optimize AI-generated email content (OpenAI API)
- Implement personalization logic for emails
- Configure CI/CD pipeline for automatic updates

3 Vishnuvardhan (Testing):

- Perform testing (unit, integration, UI/UX testing)
- Debug and optimize performance issues
- Manage sprint planning, daily standups, and backlog

Timeline & Milestones:

Day 1: Ideation & Development

Hour 1-2: Planning & Setup

- Define scope (e.g., sales emails, customer support, marketing, etc.)
- Choose AI model (OpenAI GPT, Claude, Gemini, or fine-tuned LLMs)
- Set up development environment (Python/Node.js, API keys, etc.)

Hour 3-6: Backend Development

- Implement AI-based text generation (OpenAI API or custom model)
- Add personalization (user input fields: name, tone, industry, etc.)
- Set up a simple database (Mongo DB)

Hour 7-9: Frontend Development

- Build a minimal UI (React, Next.js, or simple HTML/CSS/JS)
- Create input fields for email personalization
- Integrate frontend with AI model

Day 2: Testing, Refinement & Presentation

Hour 10-12: Testing & Debugging

- Test email generation quality (adjust model prompts if needed)
- Fix UI/UX issues for a smoother experience

Hour 13-15: Enhancements & Deployment

- Add extra features

Hour 16-18: Final Testing & Presentation Prep

- Create a short demo video or live demo script
- Prepare pitch (problem, solution, tech stack, demo)

Phase-5: Project Development.

Objective:

- **Personalization:** Generate emails tailored to individual users, industries, and tones.
- **Efficiency:** Reduce manual effort in composing emails while maintaining quality.
- **Scalability:** Allow businesses and individuals to generate bulk yet customized emails.
- **Integration:** Provide API or plugin options for CRM, email platforms, or chatbots.
- **User-Friendliness:** Ensure a simple, intuitive interface for seamless email generation.

Key Points:

1. Technology Stack Used:

Frontend (React.js)

React.js – UI framework for building interactive components

Axios – For making API requests to the backend

CSS / Material-UI – For styling and UI components

Backend (Node.js + Express.js)

Node.js – JavaScript runtime for backend logic

Express.js – Lightweight framework for API development

JWT – User authentication and security

OpenAI API – AI-generated email content

Database (MongoDB)

MongoDB – NoSQL database for storing user data and email templates

Mongoose – ODM library to manage MongoDB operations

AI & Email Services

OpenAI API – Generates AI-personalized email content

2. Development Process:

Plan & Setup

- Decide features, tech stack (MERN, OpenAI, SendGrid)
- Set up GitHub & project structure

Design Architecture

- Plan frontend, backend, database, and AI integration
- Create API routes & database schema

Build Frontend

- Develop React UI (login, email input, preview)
- Connect frontend to backend APIs

Build Backend

- Set up Node.js & Express server
- Implement authentication (JWT), AI email API, and database

AI & Email Features

- Integrate OpenAI for personalized emails
- Add email sending & scheduling (SendGrid)

Test & Debug (Week 6-7)

- Fix errors, optimize performance, and secure data

3. Challenges & Fixes:

AI Email Generation Issues – Ensuring relevant, natural, and personalized emails

Email Delivery & Spam Filtering – Avoiding spam folders & ensuring reliable delivery

Authentication & Security – Protecting user data with JWT & encryption

Performance & Scalability – Optimizing API calls & handling high traffic

Integration & Deployment Issues – Ensuring smooth backend, AI, and email service integration

Phase-6: Functional & Performance Testing

Objective:

- AI-Personalized Email Generator project is to ensure the system operates efficiently under various conditions. It aims to evaluate response times, scalability, and resource utilization while handling AI-generated email requests and real-time user interactions.

Key Points:

1. Test Cases Executed:

Functional Testing

- **Email Generation:** AI generates an email based on user input.
- **Tone Selection:** Emails change based on selected tone (formal, casual, etc.).
- **Email Length:** Generates short, medium, or long emails as per user choice.
- **Content Appropriateness:** No irrelevant or offensive content.
- **Editing & Undo:** Users can modify content before finalizing.
- **Human-like Output:** Emails sound natural, not robotic.
- **Formatting:** Proper structure with paragraphs and spacing.

Performance Testing

- **Response Time:** Email generated within **2-3 seconds**.
- **Load Handling:** Stable performance under high traffic.
- **Memory Usage:** AI operates within optimal memory limits.
- **Scalability:** Handles 100+ parallel email generations efficiently.

2. Bug Fixes & Improvements:

Bug Fixes:

- **Incorrect Personalization:** Fixed issue where user names and company names were not inserted properly.
- **Email Length Issue:** Fixed bug where emails were longer/shorter than user preference.
- **Formatting Errors:** Fixed missing line breaks and incorrect paragraph structuring.
- **Slow Response Time:** Optimized API calls to reduce email generation time.
- **Mobile UI Issues:** Fixed alignment problems in mobile view.

Improvements:

- **Better Personalization:** Enhanced AI to adapt more accurately to user inputs.
- **Faster Email Generation:** Reduced response time from **3s to 1.5s**.
- **Tone Enhancement:** Improved word choices to match selected email tone more naturally.
- **Subject Line Suggestions:** AI now provides multiple subject line options.

3. Final Validation:

Functional Testing Validation

- **Accuracy of Email Generation:** Ensure the AI generates personalized emails relevant to the given input (e.g., recipient, tone, purpose).
- **Customization Options:** Validate if users can adjust formality, tone, and structure as per requirements
- .**Grammar & Spelling Checks:** Verify correctness using NLP-based grammar checking.
- **User Input Handling:** Ensure the system correctly processes different inputs (e.g., missing fields, incomplete prompts).

Performance Testing Validation

- **Response Time:** Measure how quickly the AI generates emails under normal and peak loads.
- **Scalability:** Test the system's performance with a high volume of users.
- **System Load Handling:** Assess how the system behaves with large datasets or long email inputs.