## BACKEND ENGINEERING

Project Report

Semester-V (Batch-2023)

**EXPENSE TRACKER**

A red and white sign

Description automatically generated with low confidence

**Supervised By: Submitted By:**

Mr. Rahul Tanisha Bhardwaj (2310991065).

Tushar Koundal (2310991071).Upasvi (2310991072).

**Department of Computer Science and Engineering**

## Chitkara University Institute of Engineering & Technology,

## Chitkara University, Punjab

**Abstract**

The Expense Tracker is an intuitive and efficient platform designed to simplify personal finance management and budgeting. It enables users to record, categorize, and monitor their daily expenses while providing real-time insights into spending patterns. Built with a scalable architecture, the system leverages a user-friendly frontend, a robust backend, and secure database management to deliver a seamless financial tracking experience.

The platform incorporates intelligent data visualization, automated categorization, and personalized recommendations to help users manage their finances effectively. By reducing manual effort, enhancing transparency, and enabling data-driven financial decisions, the Expense Tracker empowers individuals to optimize their savings and spending habits.

With its secure, scalable, and user-centric design, the Expense Tracker represents a modern solution for financial management, making expense tracking accessible, reliable, and impactful.

**Table of Contents**  
**1. Introduction**

* 1. Background
  2. Objectives
  3. Significance

**2. Problem Definition and Requirements**:

2.1. Problem Statement

2.2. Software Requirements

2.3. Hardware Requirements

2.4 Data Sets

**3. Proposed Design/Methodology** :

3.1. System Architecture

3.2 Core features and Functionalities

3.3 Technology Stack

3.4 Development Methodology

3.5 Security Measure

**4. Results**:

4.1. Code Snippets

4.2 Project Snippets

**5. References**:

**1. Introduction**

**1.1 Background**

Financial management plays a critical role in the daily life of individuals as well as organizations. The ability to monitor income and expenses is the foundation of sound financial planning. Traditionally, people have relied on manual methods such as maintaining diaries, spreadsheets, or simple notes to track their expenditure. While these methods are straightforward, they are often inefficient, prone to errors, and lack the ability to provide meaningful insights into spending habits.

In today’s digital era, where automation and real-time decision-making have become the norm, a modern solution is required to address this problem. The **Expense Tracker** is designed to meet this demand by providing an intelligent and user-friendly platform that simplifies financial tracking. It allows users to log their expenses, categorize them, and view analytical reports that highlight their financial patterns.

With the growth of personal finance applications worldwide, the demand for secure, scalable, and intuitive platforms has increased significantly. This project aims to fulfill that demand by combining robust backend functionality, a clean and interactive frontend interface, and a secure database system.

**1.2 Objectives**

The major objectives of the Expense Tracker project are:

1. **Expense Recording:** Provide users with a simple mechanism to record their daily transactions with details such as date, amount, category, and notes.
2. **Categorization:** Classify expenses into categories (e.g., food, transport, utilities, entertainment) for better organization.
3. **Visualization:** Offer real-time graphical representations of expenses, including pie charts and bar graphs, for easy understanding.
4. **Analytics and Reports:** Enable users to generate weekly, monthly, and yearly reports that highlight spending trends.
5. **User Management:** Include secure user authentication and profile management for personalized usage.
6. **Data Security:** Implement security measures like data encryption, input validation, and secure authentication mechanisms to safeguard sensitive data.
7. **Scalability:** Ensure the application can handle increasing data volume and a growing number of users without performance issues.

**1.3 Significance**

The Expense Tracker project is significant in multiple ways, both from a practical and academic perspective. It addresses the growing need for financial awareness and discipline while also showcasing the application of modern software development technologies. The following points highlight its importance:

1. **Improved Financial Discipline:**  
   The system encourages individuals to record their daily transactions consistently, which increases awareness of their spending habits. By identifying unnecessary expenses, users can plan their budgets more effectively and practice better financial discipline.
2. **Automation of Expense Tracking:**  
   Manual record-keeping in diaries or spreadsheets is often tedious and prone to mistakes. The Expense Tracker automates this process, allowing expenses to be added with just a few clicks. Categorization and reporting are also automated, saving time and reducing errors.
3. **Real-Time Insights and Analytics:**  
   The application provides real-time dashboards and charts that display expense distribution across categories. This allows users to immediately identify overspending areas and make corrective financial decisions without waiting until the end of the month.
4. **Accessibility and Portability:**  
   Unlike traditional methods, the digital Expense Tracker can be accessed anytime, anywhere, through devices connected to the internet. Cloud support ensures that the user’s financial data is always available, whether on a desktop, laptop, or mobile phone.
5. **Data Security and Privacy:**  
   Financial information is highly sensitive. The Expense Tracker incorporates modern security measures such as encryption, secure authentication, and access control, ensuring that user data is protected from unauthorized access or breaches.
6. **Scalability and Future Growth:**  
   The system is designed to be scalable, meaning it can handle increasing amounts of user data and a growing number of users without performance degradation. This scalability makes it suitable not only for individuals but also for small businesses and startups looking for lightweight financial management tools.
7. **Educational and Technological Relevance:**  
   From an academic standpoint, the project demonstrates the practical application of software engineering concepts such as requirement analysis, system design, architecture modeling, security considerations, and Agile methodology. It also incorporates trending technologies like **Node.js, MongoDB, and React/Next.js**, making it relevant to current industry practices.
8. **Contribution to Financial Literacy:**  
   By simplifying the process of expense tracking and making financial data easier to interpret, the project contributes to improving financial literacy among students, young professionals, and households. It helps users develop the habit of mindful spending and saving.

**2. Problem Definition and Requirements**

**2.1 Problem Statement**

In the modern world, financial management has become a critical skill for individuals, students, professionals, and even small businesses. However, a large number of people still struggle with managing their expenses effectively due to a lack of systematic tracking and analysis.

Traditional methods such as **manual diaries, notebooks, and spreadsheets** are often inefficient, time-consuming, and prone to human error. These methods lack automation, data visualization, and analytical insights, which makes it difficult for users to identify overspending habits or make informed financial decisions. As a result, individuals are often unable to control unnecessary expenses, plan budgets accurately, or achieve their savings goals.

Additionally, existing expense-tracking applications in the market come with their own set of limitations, such as:

* **Privacy Concerns:** Many third-party apps store sensitive financial data without sufficient encryption, raising risks of data misuse or breaches.
* **High Costs:** Premium features in existing tools are locked behind expensive subscription models, making them inaccessible for students or average users.
* **Complexity:** Some applications are overloaded with features, which confuses users who only need basic and effective expense tracking.
* **Limited Customization:** Users are unable to customize categories, payment modes, or reports according to their personal financial needs.

Therefore, there is a need for a **secure, user-friendly, and scalable Expense Tracker system** that addresses these gaps by offering:

* Easy recording of daily expenses with minimal effort.
* Automatic categorization of expenses for better clarity.
* Real-time insights and visual dashboards for quick analysis.
* Strong security mechanisms to protect financial data.
* Scalability to support both individual and small-scale business usage.

By solving these problems, the proposed Expense Tracker will not only simplify the process of financial management but also promote **financial literacy, discipline, and better decision-making** among users.

### ****2.2 Software Requirements****

1. **Frontend Development**
   * **Framework:** Next.js (for server-side rendering, SEO optimization, and high performance)
   * **Languages:** JavaScript, TypeScript (optional for better scalability and type safety)
   * **Styling:** Tailwind CSS / Styled Components for responsive and modern UI design
2. **Backend Development**
   * **Runtime Environment:** Node.js
   * **Framework:** Express.js (for handling API requests and routing)
   * **Authentication:** Firebase/Auth0 or JWT-based authentication for secure user login
3. **Database Management**
   * **Database:** MongoDB (NoSQL, scalable storage for user and expense data)
   * **ODM:** Mongoose (for MongoDB object modeling and schema management)
4. **Data Analytics & Insights**
   * **Visualization Libraries:** Chart.js / D3.js for dynamic charts and expense dashboards
   * **Optional AI/ML:** TensorFlow / PyTorch (for predictive analytics or expense pattern recommendations in advanced versions)
5. **Deployment & Hosting**
   * **Frontend Hosting:** Vercel (optimized for Next.js apps)
   * **Backend Hosting:** AWS EC2 / DigitalOcean / Render
   * **Database Hosting:** MongoDB Atlas (cloud-based, scalable storage)
6. **DevOps & Version Control**
   * **Version Control:** GitHub (for collaborative development and code management)
   * **CI/CD:** GitHub Actions / Jenkins (for automated deployment)
   * **Containerization (Optional):** Docker (for easier deployment and scaling)

These software requirements ensure the Expense Tracker is secure, scalable, and capable of handling real-time expense tracking with modern development standards.

### ****2.3 Hardware Requirements****

1. **Development System (Local Machine)**
   * **Processor:** Minimum Intel Core i5 / AMD Ryzen 5 (Recommended i7/Ryzen 7 or higher)
   * **RAM:** 8GB (Recommended 16GB+ for smooth development)
   * **Storage:** 256GB SSD (Recommended 512GB+ SSD for faster read/write speeds)
   * **GPU (Optional):** NVIDIA GTX 1650 / RTX 3060+ (if running optional ML analytics)
   * **Operating System:** Windows 10/11, macOS, or Linux (Ubuntu preferred for backend development)
2. **Server Requirements (For Hosting & Analytics Processing)**
   * **Processor:** Minimum Quad-Core CPU (Recommended Intel Xeon / AMD EPYC)
   * **RAM:** 16GB (Recommended 32GB+ for multiple users and analytics processing)
   * **Storage:** 500GB SSD (Recommended NVMe SSD for high-speed database access)
   * **GPU (Optional):** NVIDIA A100 / RTX 3090 (for advanced ML analytics)
   * **Cloud Infrastructure:** AWS EC2 / DigitalOcean / Google Cloud for scalable hosting
3. **Network & Connectivity**
   * **Internet Speed:** Minimum 50 Mbps (Recommended 100+ Mbps for smooth API calls and data transfer)
   * **Cloud Database Support:** MongoDB Atlas or Firebase for secure remote access

These hardware specifications ensure smooth development, reliable hosting, and potential scalability for analytics and multi-user access.

### ****2.4 Data Sets****

The Expense Tracker relies on several datasets to provide analytics, reports, and insights. These include:

1. **User-Generated Expense Data (Primary Dataset)**
   * Individual expense entries: Amount, category, date, payment mode, and notes
   * Historical expenses for trend analysis
   * Example:
   * {
   * "userId": "U101",
   * "amount": 450,
   * "category": "Food",
   * "paymentMode": "UPI",
   * "date": "2025-09-23",
   * "note": "Lunch at cafe"
   * }
2. **Predefined Categories & Reference Data**
   * Expense categories: Food, Transport, Entertainment, Utilities, Health, Shopping
   * Payment modes: Cash, Card, UPI, Wallets
   * Helps in automatic categorization and reporting
3. **Analytical & Trend Data (Optional / Advanced)**
   * Aggregated spending patterns for personalized insights
   * Machine learning datasets for predictive analysis (if advanced analytics is implemented)
4. **User Interaction Logs (Post-Deployment)**
   * Frequency of app usage, category preferences, report downloads
   * Used to enhance dashboard recommendations and UX improvements

Leveraging these datasets allows the Expense Tracker to provide accurate, real-time insights, improve user experience, and enable scalable expense management for multiple users.

## ****3. Proposed Design / Methodology****

### ****3.1 System Architecture****

The Expense Tracker follows a **modular, scalable, and data-driven architecture** to ensure smooth expense management, secure user interactions, and real-time analytics.

1. **Client-Side (Frontend – Next.js / React.js)**
   * **User Interface (UI):** Built with Next.js for fast rendering, responsive design, and SEO optimization.
   * **API Integration:** Communicates with the backend via RESTful APIs for fetching, updating, and deleting expense records.
   * **Authentication & Authorization:** Handled using Firebase/Auth0 or JWT for secure user login and session management.
2. **Server-Side (Backend – Node.js & Express.js)**
   * **API Layer:** Express.js handles user requests, business logic, and expense processing.
   * **Expense Processing Module:** Calculates totals, categorizes expenses, and generates analytical reports.
   * **Authentication & Security:** Implements JWT/OAuth and input validation to ensure secure access and prevent attacks.
3. **Database Layer (MongoDB)**
   * **User Data Management:** Stores user profiles, preferences, and authentication details.
   * **Expense Data Storage:** Saves all expense entries, categories, and historical data.
   * **Fast Query Execution:** NoSQL schema ensures high-speed retrieval of data for dashboard analytics.
4. **Analytics & Reporting Layer**
   * **Data Aggregation:** Aggregates expenses by category, month, and payment method.
   * **Visual Analytics:** Generates charts and dashboards for spending patterns.
   * **Recommendations (Optional Advanced Feature):** Provides tips for saving or highlights overspending categories using ML algorithms.
5. **Deployment & Hosting**
   * **Frontend Hosting:** Vercel (optimized for Next.js apps).
   * **Backend Hosting:** AWS EC2 / DigitalOcean / Render for scalable performance.
   * **Database Hosting:** MongoDB Atlas for secure cloud storage.
6. **Monitoring & Optimization**
   * **Performance Tracking:** Monitors API performance and dashboard load times.
   * **Logging & Error Handling:** Uses Winston & Morgan for backend error logging and monitoring.
   * **Continuous Deployment (CI/CD):** GitHub Actions for automated deployment and updates.

This architecture ensures **scalability, high performance, security, and easy maintainability**, making the Expense Tracker reliable for multiple users and large datasets.

### ****3.2 Core Features and Functionalities****

1. **User Authentication & Management**
   * Secure login and registration using Firebase/Auth0 or JWT.
   * Role-based access control for personal accounts.
2. **Expense Management**
   * Add, edit, delete, and view expenses.
   * Automatic categorization of expenses (Food, Transport, Utilities, etc.).
   * Record payment mode and notes for each transaction.
3. **Real-Time Dashboard & Analytics**
   * Dynamic pie charts, bar charts, and line graphs for visual representation of expenses.
   * Category-wise and monthly expense summaries.
4. **Reports & Exporting**
   * Generate weekly, monthly, or yearly reports.
   * Export data to CSV or PDF for offline use.
5. **Alerts & Notifications (Optional Advanced Feature)**
   * Budget limit notifications for categories.
   * Alerts for unusual spending patterns.
6. **Responsive & Mobile-Friendly Design**
   * Fully responsive UI for desktop, tablet, and mobile devices.
   * Smooth and intuitive user experience.
7. **Cloud-Based & Scalable Architecture**
   * Frontend hosted on Vercel, backend on AWS/DigitalOcean, database on MongoDB Atlas.
   * Ensures performance under high traffic and multiple user access.
8. **Optional Predictive Insights (Advanced)**
   * ML models predict spending trends based on historical data.
   * Personalized recommendations for savings and budget optimization.

These features make the Expense Tracker a **powerful, user-friendly, and scalable financial management tool** for individuals and small businesses.

### ****3.3 Technology Stack****

1. **Frontend**
   * **Framework:** Next.js / React.js (for SSR, SEO optimization, and fast UI rendering)
   * **Languages:** JavaScript, TypeScript (optional for scalability and maintainability)
   * **Styling:** Tailwind CSS / Styled Components (responsive and modern UI)
2. **Backend**
   * **Runtime Environment:** Node.js (event-driven, non-blocking architecture)
   * **Framework:** Express.js (lightweight and efficient API handling)
   * **Authentication:** Firebase/Auth0 or JWT for secure access control
3. **Database**
   * **Primary Database:** MongoDB (NoSQL, flexible, and scalable)
   * **ODM:** Mongoose (for schema modeling and database interaction)
4. **Analytics & Optional ML Layer**
   * **Visualization Libraries:** Chart.js / D3.js for expense dashboards
   * **Optional Machine Learning:** TensorFlow / PyTorch for predictive spending insights
5. **Deployment & Hosting**
   * **Frontend Hosting:** Vercel (optimized for Next.js apps)
   * **Backend Hosting:** AWS / DigitalOcean / Render
   * **Database Hosting:** MongoDB Atlas (managed cloud database)
6. **DevOps & Version Control**
   * **Version Control:** GitHub for collaborative development
   * **CI/CD:** GitHub Actions / Jenkins for automated deployment
   * **Containerization (Optional):** Docker for microservices architecture and scalability

This modern, modular, and scalable tech stack ensures **high performance, security, and maintainability**, making the Expense Tracker a robust solution for daily financial management.

### ****3.4 Development Methodology****

The **Expense Tracker** project follows the **Agile Development Methodology** to ensure flexibility, continuous improvement, and efficient project execution. Agile practices help manage iterative development, allowing rapid feature updates and user-driven enhancements.

#### **1. Agile & Scrum Approach**

* **Sprint-Based Development:** The project is divided into small, manageable sprints of 1–2 weeks, focusing on developing specific features such as expense logging, dashboards, and reports.
* **Daily Standups:** Regular team meetings (or personal review sessions for solo developers) to track progress, discuss challenges, and plan tasks.
* **Iterative Development:** Continuous refinement and improvement of features based on feedback from users or mentors.

#### **2. Phases of Development**

1. **Planning & Requirement Gathering**
   * Identify key features such as expense tracking, categorization, analytics, and reports.
   * Determine target users (students, professionals, small businesses) and technology stack (Next.js, Node.js, MongoDB).
   * Define project milestones and sprint timelines for development and testing.
2. **System Design & Architecture**
   * Design a modular, scalable architecture for frontend, backend, and database layers.
   * Create API structures, database schemas, and define deployment strategy for cloud hosting.
3. **Development & Implementation**
   * **Frontend:** Build responsive UI components using Next.js and Tailwind CSS.
   * **Backend:** Develop RESTful APIs with Node.js and Express.js for CRUD operations.
   * **Analytics Integration:** Implement charts, dashboards, and optional predictive insights.
4. **Testing & Quality Assurance**
   * **Unit Testing:** Validate individual components like API endpoints and UI forms.
   * **Integration Testing:** Ensure seamless communication between frontend, backend, and database.
   * **Load & Performance Testing:** Test scalability and optimize response times.
5. **Deployment & Monitoring**
   * Deploy frontend on Vercel and backend on AWS/DigitalOcean with MongoDB Atlas.
   * Implement CI/CD pipelines for automated updates and deployment.
   * Use monitoring tools for performance tracking and error logging.
6. **Continuous Improvement & Updates**
   * Collect user feedback and analyze usage metrics.
   * Optimize performance, UI/UX, and analytical features iteratively.
   * Roll out regular updates with enhancements and bug fixes.

This Agile-based methodology ensures **efficient development, rapid iterations, and continuous enhancement**, resulting in a reliable, user-friendly Expense Tracker application.

### ****3.5 Security Measures****

The Expense Tracker incorporates robust security measures to ensure **data protection, user privacy, and secure operations**.

#### **1. Authentication & Access Control**

* **JWT Authentication:** Secure login and session management for users.
* **Role-Based Access:** Differentiates access for standard users versus admin features (optional for multi-user systems).
* **Optional Multi-Factor Authentication (MFA):** Adds an extra layer of login security.

#### **2. Data Security & Encryption**

* **End-to-End Encryption (E2EE):** Encrypts sensitive data during transmission.
* **AES-256 & SHA-256 Encryption:** Secures data at rest and protects passwords via hashing.
* **SSL/TLS Certificates:** Encrypts all client-server communication.

#### **3. Secure API & Backend Protection**

* **Rate Limiting & Throttling:** Prevents API abuse and DDoS attacks.
* **Input Validation & Sanitization:** Protects against SQL injection and XSS attacks.
* **CSRF & CORS Protection:** Ensures safe cross-origin requests.

#### **4. Database & Infrastructure Security**

* **MongoDB Atlas Security Features:** Automated backups, access control, and encrypted cloud storage.
* **Regular Security Audits:** Routine checks to identify vulnerabilities.
* **Logging & Monitoring:** Backend logs track activity for real-time threat detection.

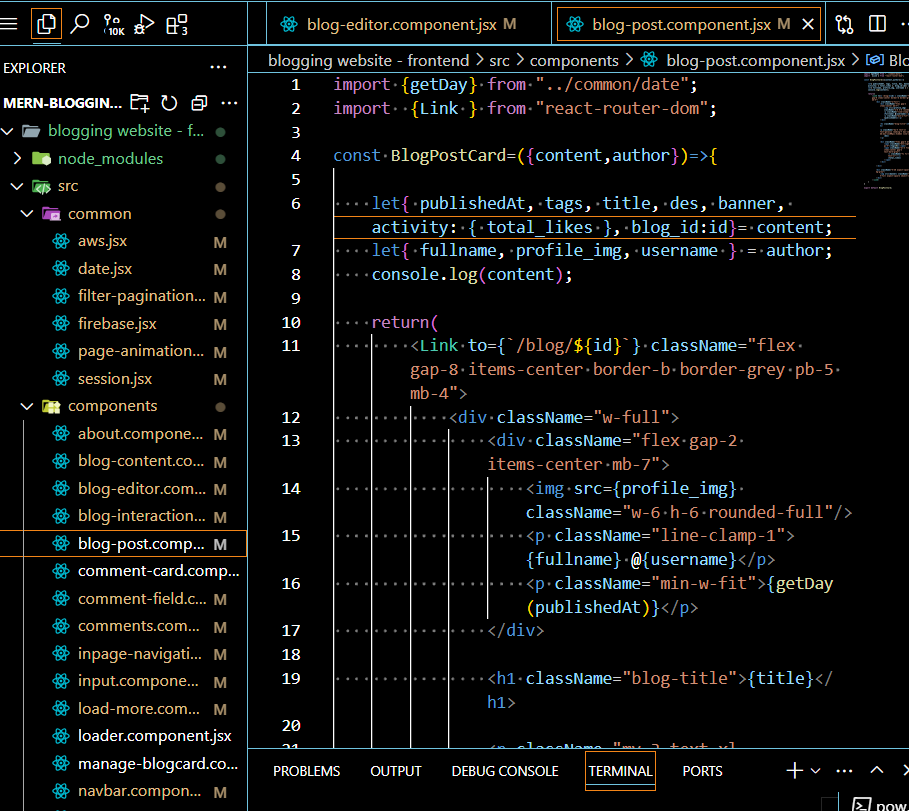
#### **5. Compliance & Data Privacy**

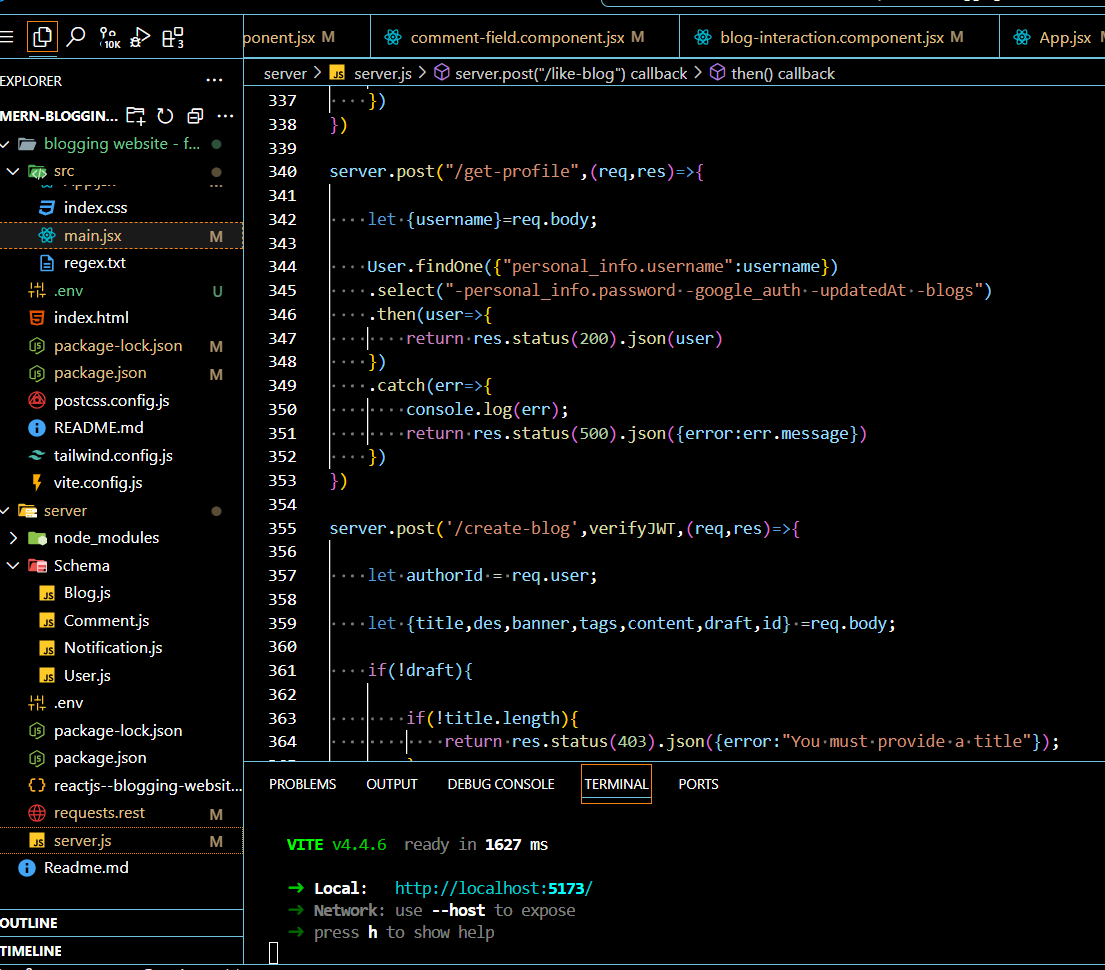
* **GDPR & CCPA Compliance:** Adheres to international data privacy regulations.
* **User Data Anonymization:** Protects sensitive personal information.
* **Secure Data Retention & Deletion Policies:** Ensures responsible handling and removal of data.

These security measures ensure **high-level protection, data integrity, and compliance**, making the Expense Tracker a safe and trustworthy platform for personal financial management.

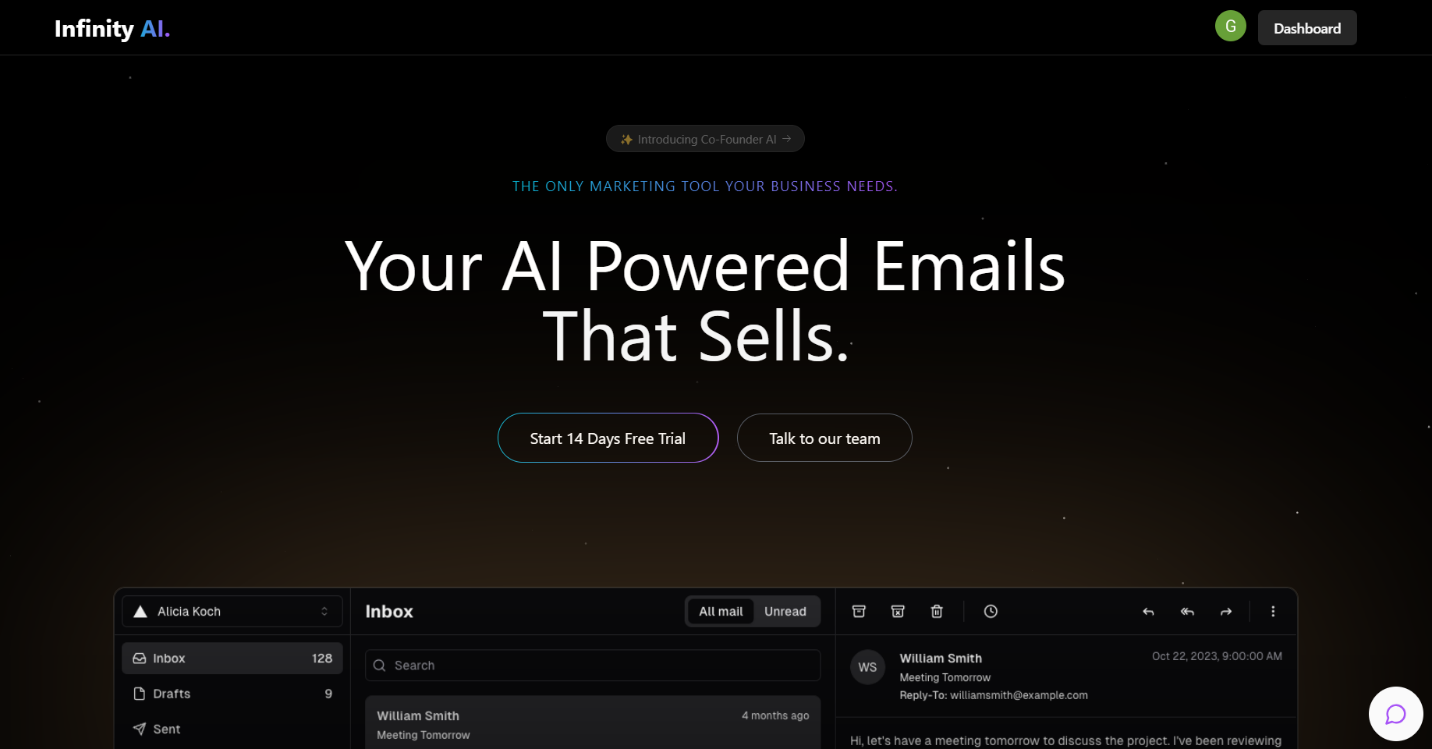
**4. Result**

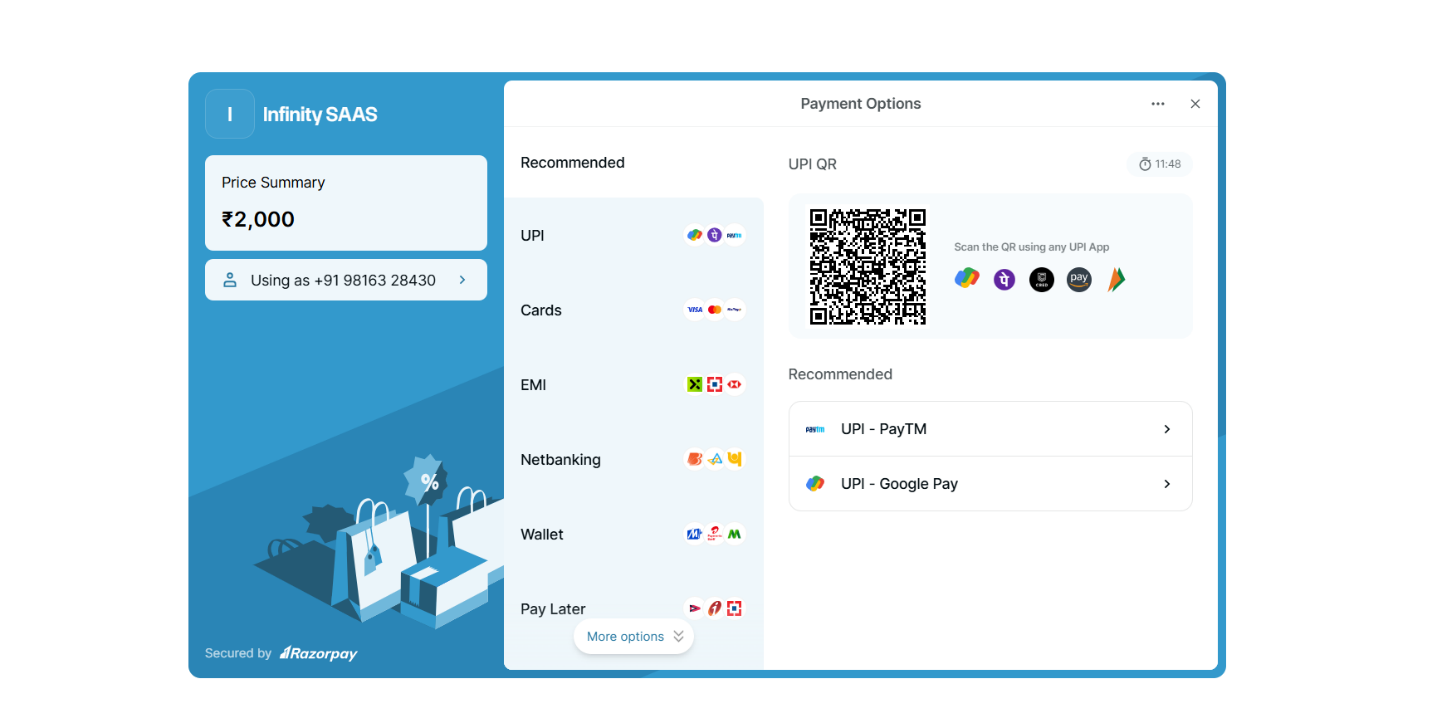
**4.1 Code snippets:**

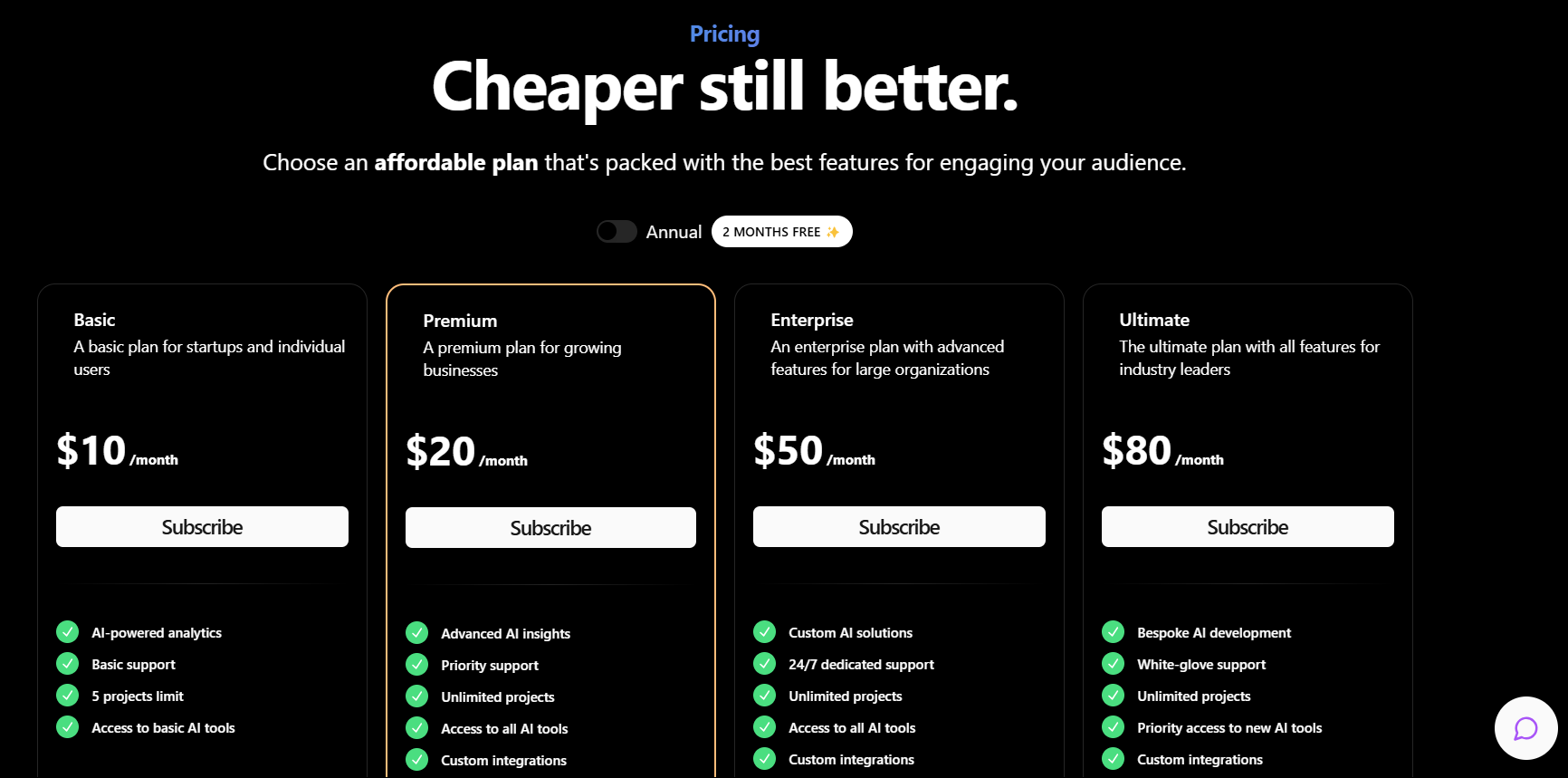


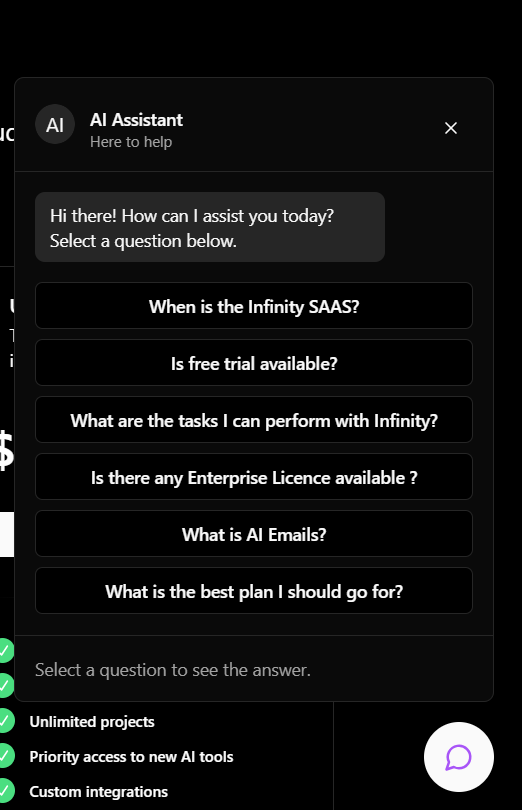


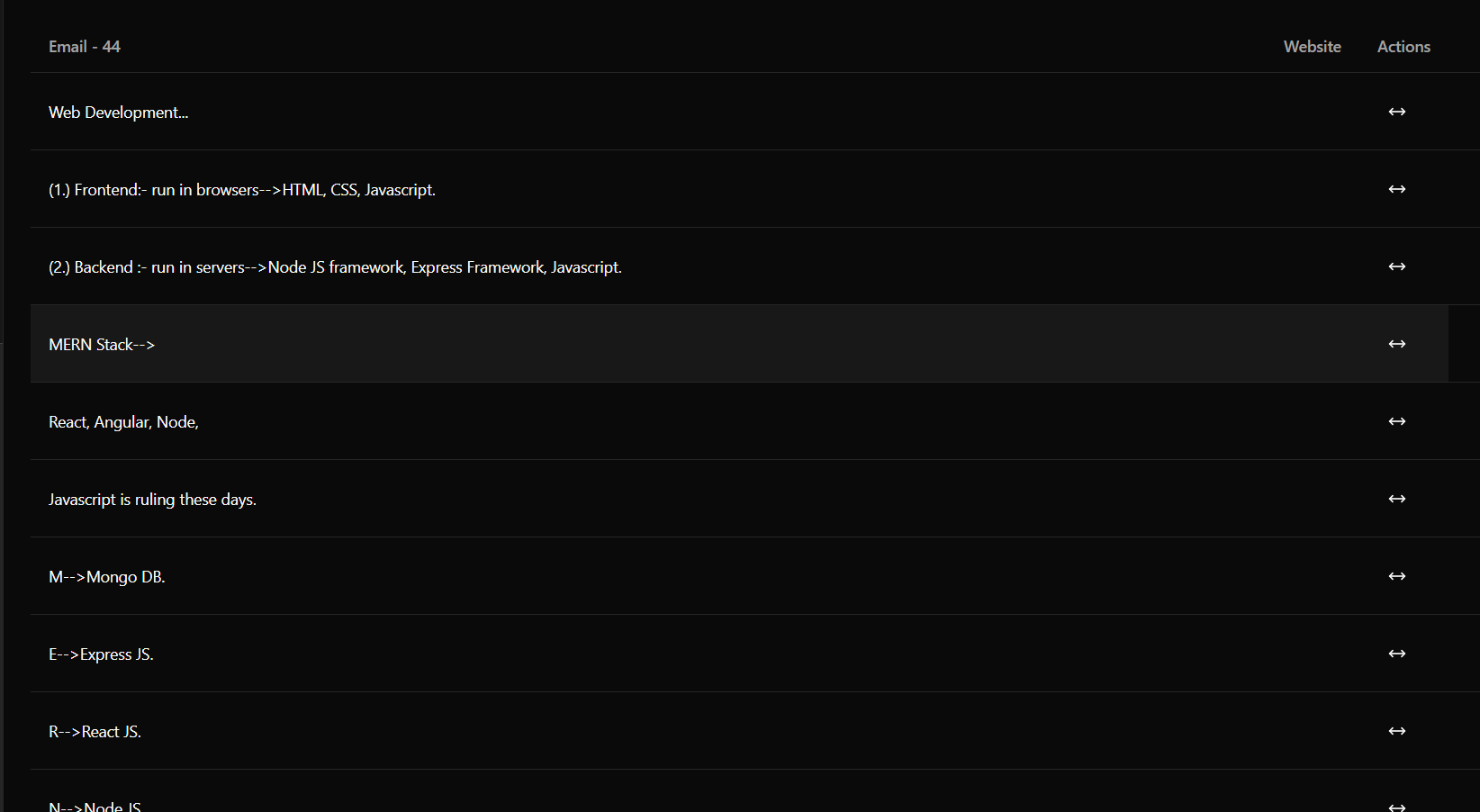
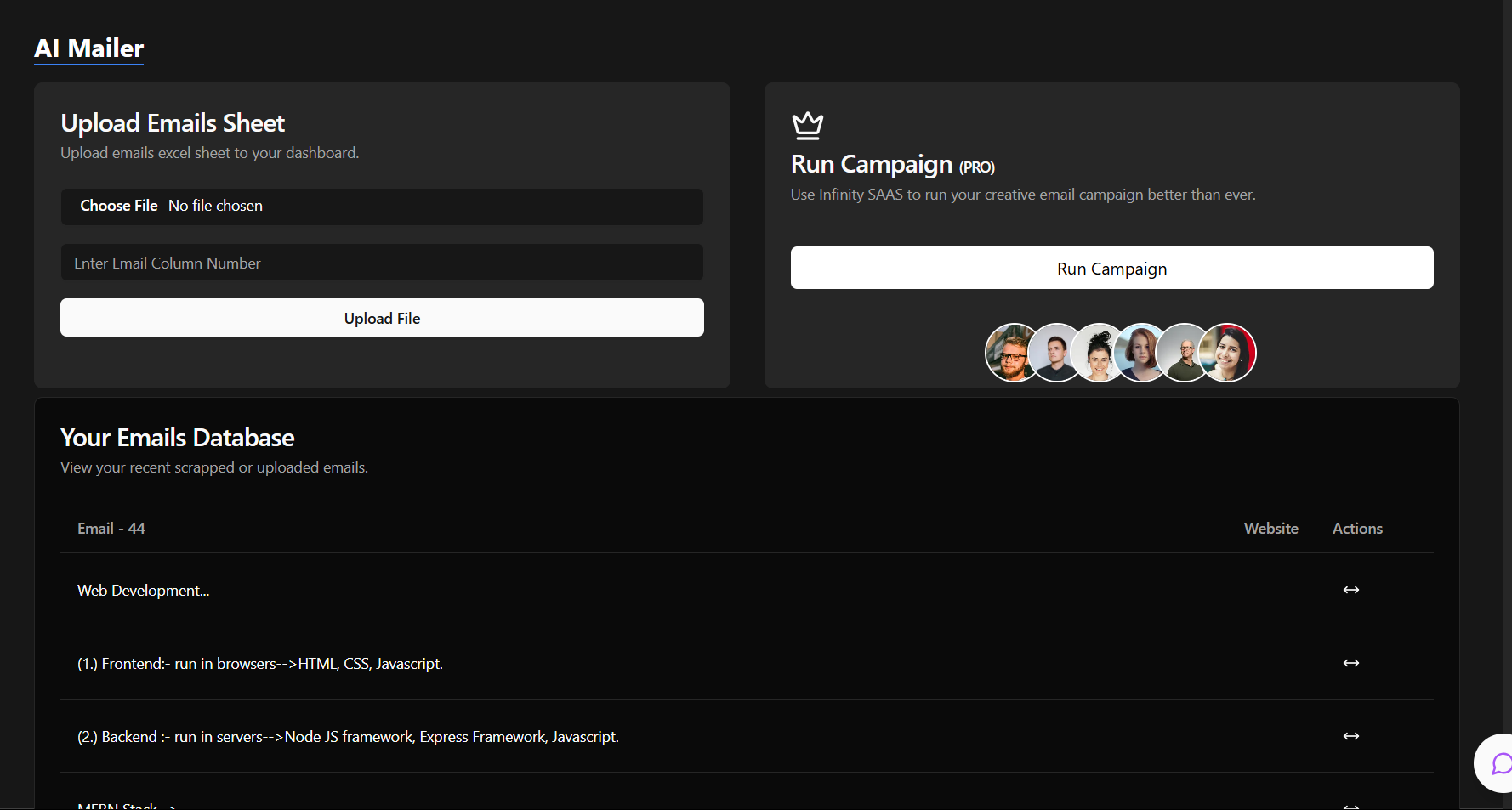
**4.2 Project Snippets:**

****

****

****

****



**5. References:**

1. “Run JavaScript everywhere,” Nodejs.org. [Online]. Available: <https://nodejs.org/en>. [Accessed: 18-Sep-2024].
2. “React,” React.dev. [Online]. Available: <https://react.dev/>. [Accessed: 18-Sep-2024].
3. “Express – Node.js web application framework,” Expressjs.com. [Online]. Available: <https://expressjs.com/>. [Accessed: 18-Sep-2024].
4. “Documentation – Tailwind CSS,” Tailwindcss.com. [Online]. Available: <https://v2.tailwindcss.com/docs>. [Accessed: 18-Sep-2024].
5. “GitHub Docs,” GitHub.com. [Online]. Available: <https://docs.github.com/en>. [Accessed: 18-Sep-2024].