



Habit Tracker App

Submitted in partial fulfillment of the requirements of the degree of
Bachelor of Engineering (Information Technology)

By

Shreya Sawant Roll no -53



Department of Information Technology
VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF
TECHNOLOGY,

Chembur, Mumbai 400074

(An Autonomous Institute, Affiliated to University of
Mumbai)

April 2024

Contents :

Content	Page No.
Project Description	1-2
Requirement gathering	1-2
System requirements	2-3
Technologies used	2-3
Setup instructions	4-6
Project structure	6-7
Architectural diagrams	7-9
Features implemented	9-10
Screenshots of implementation	11-14
Future scope	14-15
Github link	14-15
Conclusion	15

Habit Tracker App

Name of student	Shreya Sawant
Class_Roll no	D15A_53
D.O.P	3/04/25
D.O.S	17/04/25
Sign and Grade	

Title : Habit Tracker App

Project Description : This web application is a habit tracking system developed to help users build and maintain consistent routines. Using **Flask with Jinja templating** for the frontend and **MongoDB** for the database, the app offers a user-friendly interface for managing daily habits effectively. It allows users to add custom habits, mark completion on a daily basis, and visualize their progress through streaks and dashboards. The system emphasizes simplicity, consistency, and motivation, making it easier for users to stick to their goals. With a responsive layout and smooth navigation, the application is accessible across devices and encourages positive behavior change.

Requirement gathering : The requirement gathering for this habit tracker application was carried out through informal user discussions, observation of routine-building challenges, and a comparative study of existing habit tracking tools. The aim was to uncover common frustrations such as cluttered interfaces, lack of motivation features, and limited flexibility in habit scheduling. Insights were gathered from individuals with varying goals—ranging from students aiming to improve study habits to professionals working on personal productivity.

The system was designed with simplicity and usability in mind, while also supporting essential habit tracking features. Functional requirements included the ability to create, update, and delete habits, track daily completions, and visualize progress through streaks and statistics. Non-functional requirements focused on ensuring a responsive design, smooth navigation using Flask with Jinja templating, secure data management through MongoDB, and overall performance optimization.

System Requirements :

1. Hardware Requirements:

- **Processor:** Intel Core i5 / AMD Ryzen 5 or higher (dual-core, 2.0 GHz or faster)
- **RAM:** Minimum 8GB (16GB recommended)
- **Storage:** At least 1GB free space (256GB SSD recommended)
- **Network:** Stable internet connection (especially for MongoDB Atlas users)

2. Software Requirements:

- **Operating System:** Windows 10/11, macOS 10.15+, or Ubuntu 20.04+
- **Code Editor:** Visual Studio Code or compatible IDE
- **Version Control:** Git 2.25+

Technologies Used :

Development	VS Code , Git
Frontend	Flask with Jinja Templating
Backend	Flask (Python 3.8+)
Database	MongoDB
Styling	CSS / Bootstrap

Setup Instructions :

- **Python 3.8+:** To set up the habit tracker application, begin by installing Python 3.8 or higher from the official Python website. During installation, especially on Windows, make sure to check the option "Add Python to PATH." Once installed, verify the setup by running `python --version` and `pip --version` in your terminal. For better environment management, create a virtual environment using `python -m venv venv`, then activate it using `venv\Scripts\activate` on Windows or `source venv/bin/activate` on macOS/Linux. Once activated, install the necessary dependencies using `pip install -r requirements.txt`.
- **MongoDB (Local or Cloud - MongoDB Atlas):** Set up MongoDB either locally or using MongoDB Atlas. For a local setup, download MongoDB Community Edition from the official MongoDB website and follow the installation steps specific to your operating system. If you prefer a cloud-based setup, visit <https://www.mongodb.com/cloud/atlas>, create a free cluster, set up a database, and generate a connection string. This string will be used in your Flask backend configuration to establish a database connection.
- **Flask with Jinja Templating:** The frontend and backend are developed using Flask along with Jinja templating for rendering dynamic content. After setting up the environment and database, run the application using `flask run`. Make sure to set the required environment variables beforehand by using `set FLASK_APP=app.py` and `set FLASK_ENV=development` on Windows or `export FLASK_APP=app.py` and `export FLASK_ENV=development` on macOS/Linux. Once launched, the application will be accessible at <http://localhost:5000>.

Backend Setup:

1. Navigate to the **backend** folder:
`cd project`

(Optional) Create a virtual environment:

```
python -m venv venv  
venv\Scripts\activate # For Windows  
# or  
source venv/bin/activate # For macOS/Linux
```

Install dependencies:

Run the following command to install the required libraries:
`pip install -r requirements.txt`

Start the Flask server:

Navigate to the api directory and start the Flask server with:

```
cd api  
python app.py
```

2. The **Backend** will run at:
`http://localhost:5000`

Frontend Setup:

1. Navigate to the **frontend** folder:
`cd project`

Install dependencies:

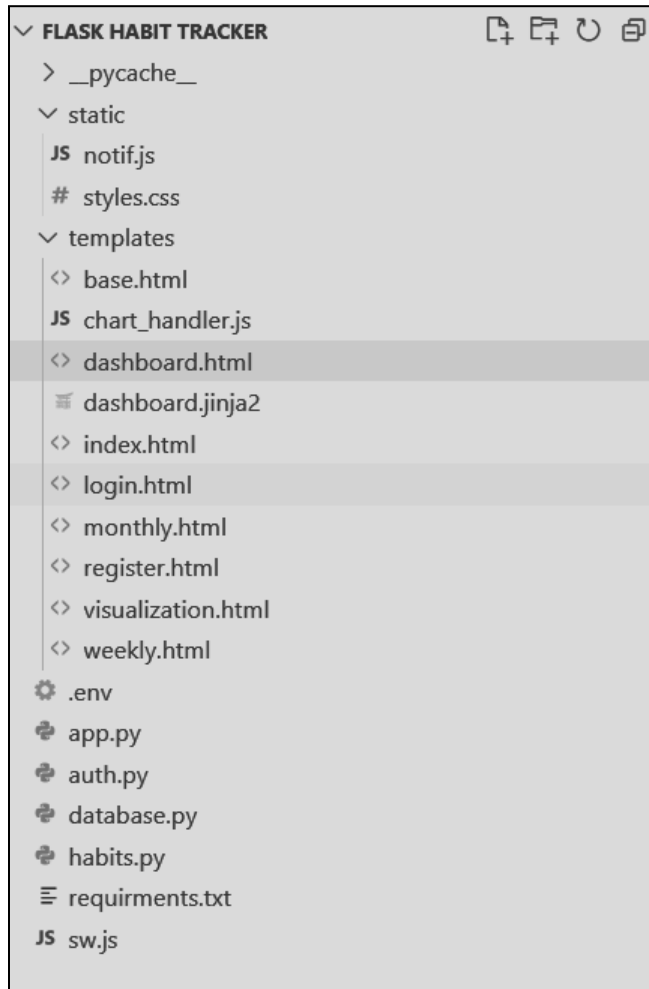
Run the following command to install required frontend dependencies:
`npm install`

Start the Angular development server:

Start the server with:
`ng serve`

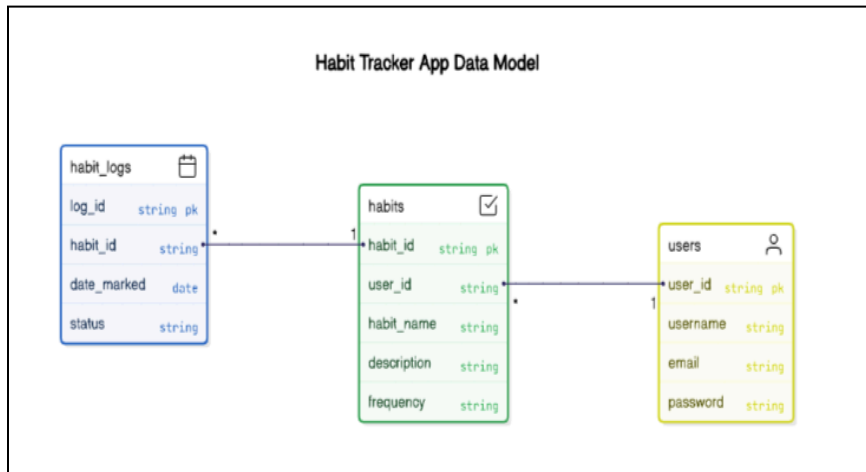
2. The **Frontend** will run at:
<http://localhost:4200>

Project Structure

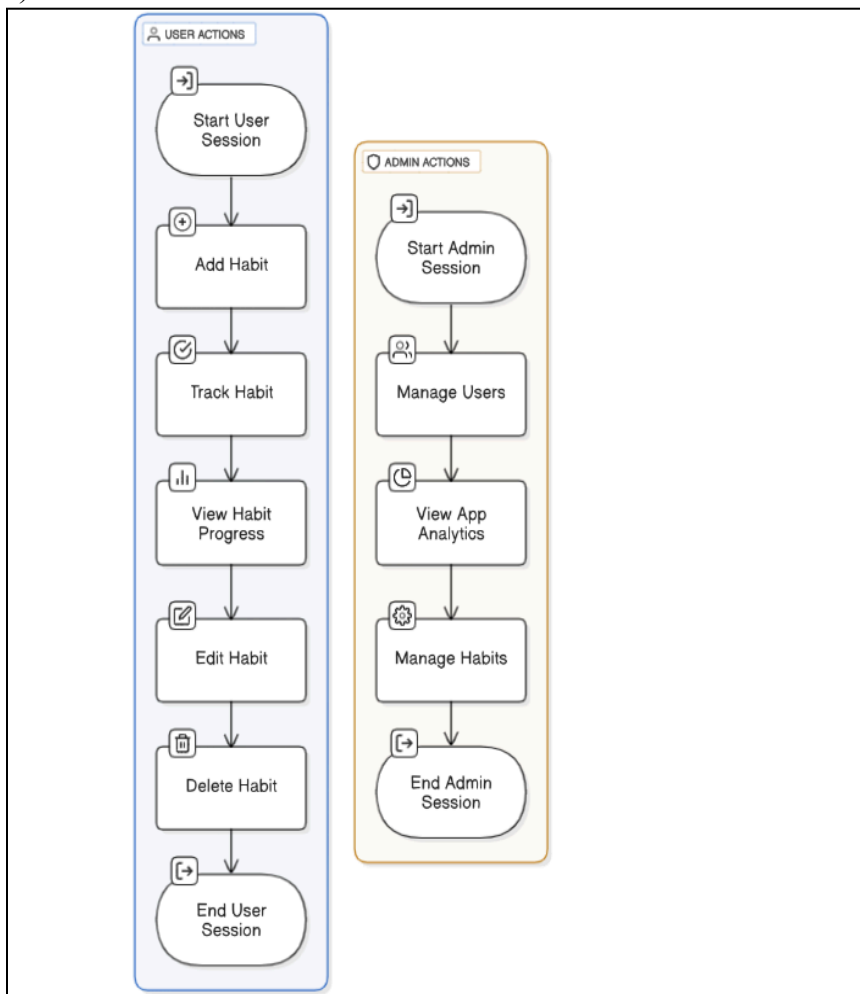


Architectural Diagrams :

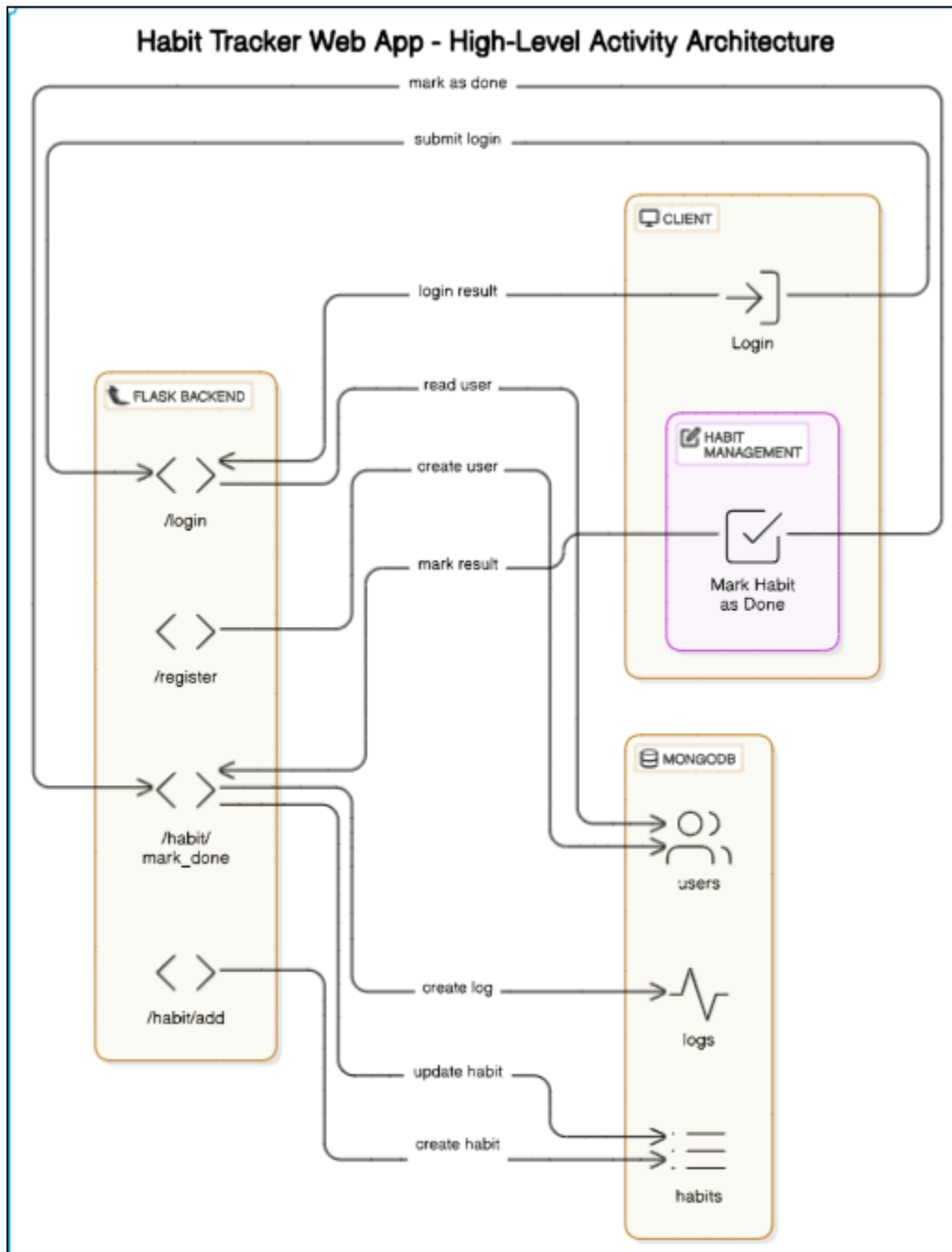
a) Class Based Diagram -



b) Flowchart -



Architecture Diagram :

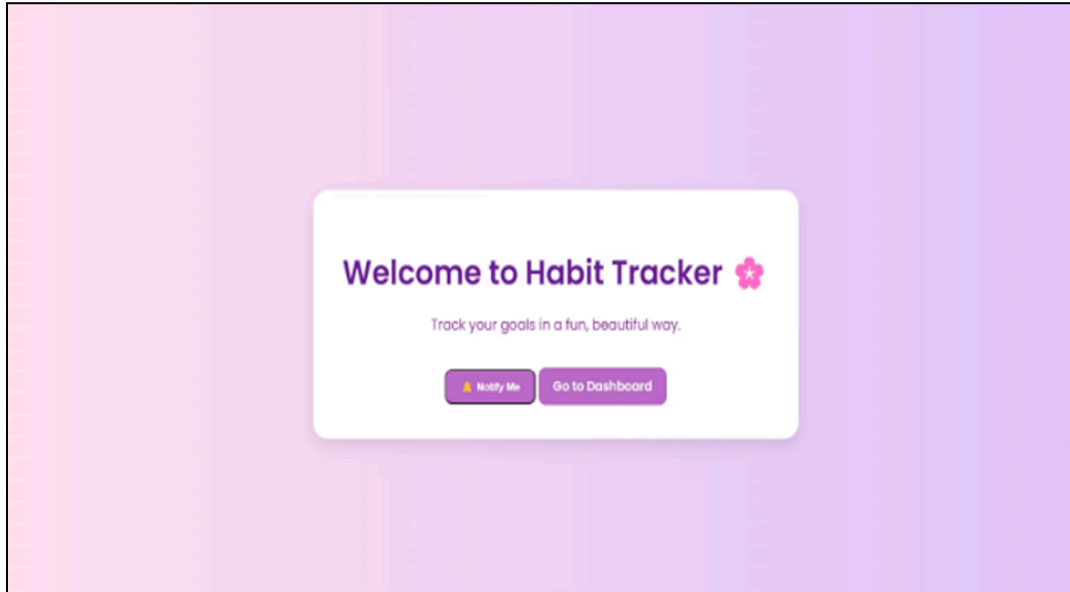


Features Implemented :

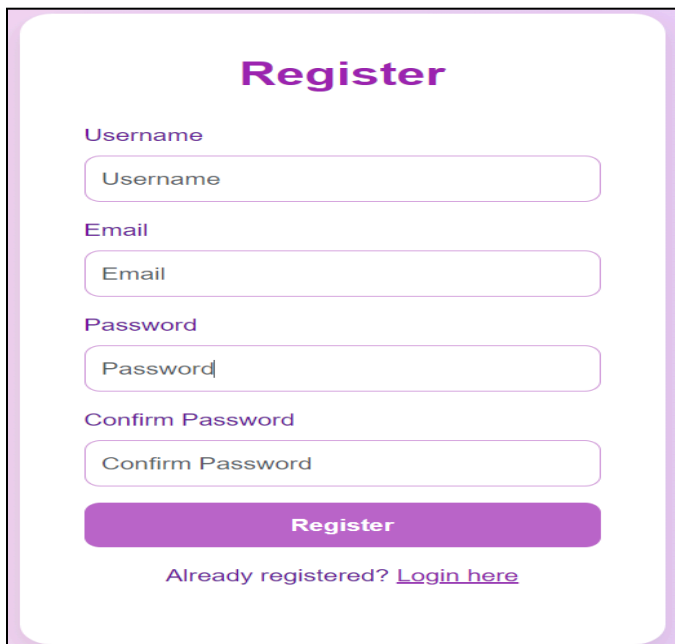
1. **Habit Management:**
Create, update, delete, and organize habits for users to track their daily routines.
2. **Progress Tracking:**
Visual progress indicators (such as completion percentages or streaks) for tracking habit completion over time.
3. **Responsive Design:**
A mobile-first UI design using SCSS and Bootstrap, ensuring a seamless experience across all devices, from mobile phones to desktops.
4. **RESTful APIs:**
Communication between the Angular frontend and Flask backend through REST APIs to handle user requests and data interactions.
5. **User Authentication:**
Secure login/registration using email and password authentication, along with optional one-time password (OTP) for added security during the registration process.
6. **Habit Reminders:**
Notifications to remind users about their habits and encourage consistency, ensuring they stay on track.
7. **Habit Analytics:**
Integration of Google Analytics to track user engagement, behavior, and activity within the app for continuous improvement.
8. **Task Breakdown for Habits:**
Breaking down larger habits into smaller tasks or milestones, helping users track their progress in a more structured way.
9. **Email Notifications for Habit Tracking:**
Sending email notifications to users to remind them of their habits, motivating them to complete their daily tasks and keep up with their progress.

Screenshots of implementation :

Welcome Page -



Register Page -

A screenshot of a registration form titled 'Register' in a bold, dark purple font. The form is set against a light purple background with a subtle gradient. It contains four input fields, each with a label above it: 'Username', 'Email', 'Password', and 'Confirm Password'. Each input field has a light purple border and a placeholder text matching the label. Below the input fields is a prominent purple button with the text 'Register' in white. At the bottom of the form, there is a link that says 'Already registered? [Login here](#)'.

Login page -

Login

Email

Password

Login

Don't have an account? [Register here](#)

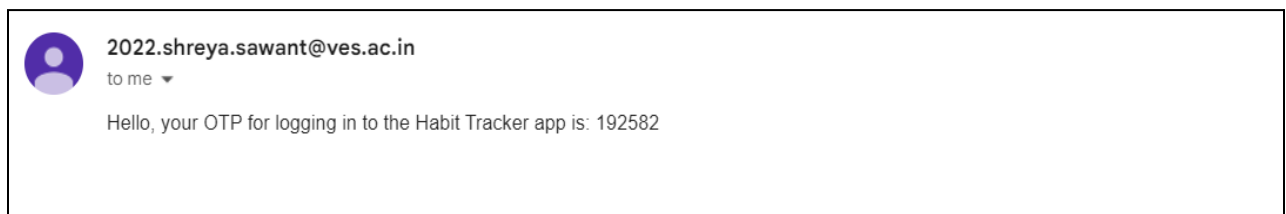
Login

OTP

OTP sent to your email.

Verify OTP

Don't have an account? [Register here](#)



Dashboard /homepage -

Habit Dashboard

Monthly Tracker Weekly Tracker

Add New Habit

Enter a new habit...

Add a short description (optional)

+ Add Habit

singing
Added on: 16 April 2025

water the plants
Added on: 18 April 2025


meditation
Added on: 18 April 2025

 **Habit Summary**

- 2025-04-16: 1 habit(s)
- 2025-04-18: 2 habit(s)

[← Back to Home](#)


monthly tracker page -

 **Monthly Habit Tracker**

water the plants

1 <input type="checkbox"/>	2 <input type="checkbox"/>	3 <input type="checkbox"/>	4 <input type="checkbox"/>	5 <input type="checkbox"/>	6 <input type="checkbox"/>	7 <input type="checkbox"/>
8 <input type="checkbox"/>	9 <input type="checkbox"/>	10 <input type="checkbox"/>	11 <input type="checkbox"/>	12 <input type="checkbox"/>	13 <input type="checkbox"/>	14 <input type="checkbox"/>
15 <input type="checkbox"/>	16 <input type="checkbox"/>	17 <input type="checkbox"/>	18 <input type="checkbox"/>	19 <input type="checkbox"/>	20 <input type="checkbox"/>	21 <input type="checkbox"/>
22 <input type="checkbox"/>	23 <input type="checkbox"/>	24 <input type="checkbox"/>	25 <input type="checkbox"/>	26 <input type="checkbox"/>	27 <input type="checkbox"/>	28 <input type="checkbox"/>
29 <input type="checkbox"/>	30 <input type="checkbox"/>	31 <input type="checkbox"/>				

weekly tracker page-

 **Weekly Habit Tracker**

water the plants

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Mon	Tue	Wed	Thu	Fri	Sat	Sun

meditation

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Mon	Tue	Wed	Thu	Fri	Sat	Sun

[← Back to Dashboard](#)

Set reminder page -

127.0.0.1:5000 says

Reminder set for habit: Give water to the plants at 03:01.

OK

Set Habit Reminder

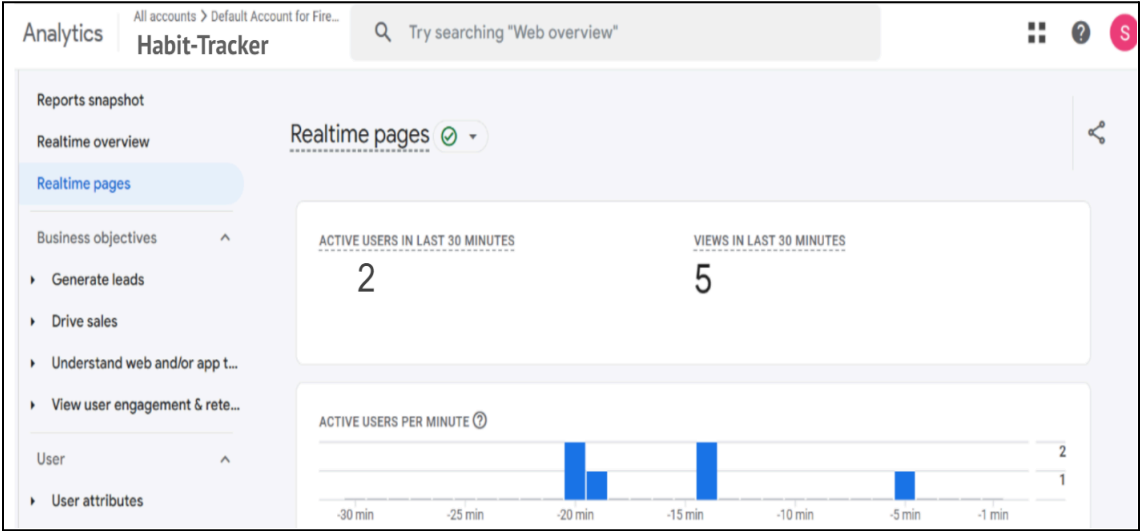
Give water to the plants

03 : 01

⌚

Set Reminder

Google Analytics :

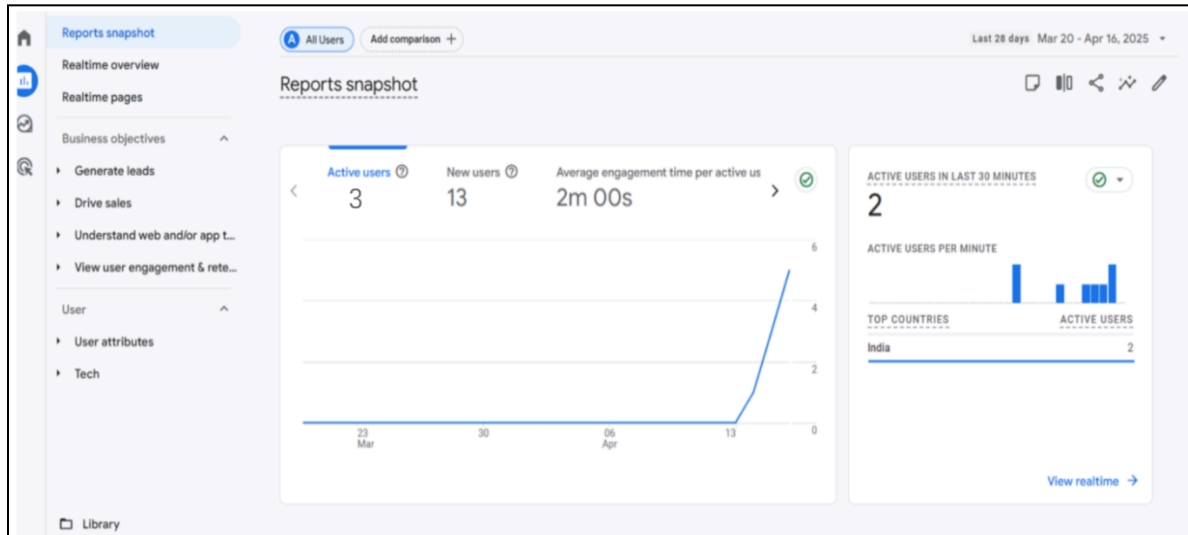


Page path and screen class in last 30 minutes

Search...

Rows per page: 10 1-6 of 6

	Page path and screen class	↓ Active users	Views
1	/ Dashboard	2	2
2	/Add Reminder	2	1
3	/login	2	1
4	/Month/67ffe964b2c8d0233813c134	1	1
5	/ Monthly	1	0
6	/ Weekly	1	0



MongoDb (Database) -

The screenshot shows the MongoDB Atlas interface for a project named 'SHREYA'S ORG'. The left sidebar contains navigation links for Overview, DATABASE, Clusters, SERVICES, and SECURITY. The main panel displays the 'habit_tracker.user' collection, showing storage size (4KB), logical data size (0B), total documents (0), and index size (4KB). Below this, there's a section for 'Generate queries from natural language in Compass' with fields for Filter, Project, Sort, and Collation. The 'Filter' field contains a query: { field: 'value' }. The 'Project' field contains { field: 0 }. The 'Sort' field contains { field: -1 } or ['field', -1]. The 'Collation' field contains { locale: 'simple' }. There are buttons for 'Reset', 'Apply', and 'Options'.

habit_tracker.user

STORAGE SIZE: 4KB LOGICAL DATA SIZE: 0B TOTAL DOCUMENTS: 0 INDEXES TOTAL SIZE: 4KB

Filter: { field: 'value' }

Project: { field: 0 }

Sort: { field: -1 } or ['field', -1]

Collation: { locale: 'simple' }

Future Scope : The habit tracker has the potential to grow into a holistic wellness and self-improvement platform. In the future, it can incorporate AI-driven insights such as personalized habit suggestions, progress predictions, and adaptive goal-setting based on user behavior. Integration with natural language processing can enable users to log and manage habits through voice commands, making the app more intuitive and accessible. Cross-platform compatibility with health apps, fitness trackers, and smartwatches can enable real-time habit monitoring and automated updates. Additionally, social and collaborative features like shared goals, group challenges, and community support can foster motivation and accountability, making the habit tracker ideal for both individual users and wellness-focused groups.

Github Link : <https://github.com/sawantshreya11/Flask-habit-tracker->

Conclusion: The habit tracker streamlines personal growth and daily routine monitoring through a well-integrated tech stack comprising React, Node.js, and MongoDB. The setup process involves installing key tools such as Node.js, npm, MongoDB, and configuring React for a responsive front-end experience. With an intuitive interface and features like customizable habit creation, daily tracking, and insightful progress visualization, the application encourages consistency and goal achievement. The successful implementation of seamless CRUD operations and real-time updates showcases effective full-stack development skills. This habit tracker serves as a practical and motivating solution for individuals aiming to build and maintain positive habits in their everyday lives.