Project: API Hit Tracking and Analytics Dashboard - Timeline 2 days

Requirement

Design and develop a web application that effectively tracks API hits, stores them in a PostgreSQL database, and presents insightful visualizations on a separate dashboard. This project will provide valuable insights into API usage patterns and trends, facilitating informed decision-making.

API Tracking fields -

For Each endpoint that hits the server, extract the below details from Request headers and save it to Postgress.

Ensure the captured data includes essential information like:

- Request timestamp
- Request type GET, POST, PUT, DELETE
- API endpoint accessed
- User agent (browser information)
- Request body (optional, depending on API design)
- Operating System
- IP Address

Sample Table in Database

Id	**RequestId**	**RequestType**	**RequestTime**	**Payload**	**Content-type**	**IP address**	**OS**	**UserAgent**
R001	get-items	GET	2024/04/05 12:30			192.168.1.1	Windows	Chrome
R002	get-items	GET	2024/04/05 12:35			192.168.1.2	Macos	Safari
R003	update-items	POST	2024/04/05 12:45	{“a”:””hello”}	application/json	192.168.1.1	Windows	Chrome
R004	delete-item	DELETE	2024/04/05 12:50			192.168.1.1	Postman	Postman

Dashboard

1. Create a user-friendly dashboard interface to display the collected API hit analytics.(Tabular format with filters)

2. Pie Chart - Number of Requests by Browser: Illustrate the breakdown of requests by different browsers used by users.

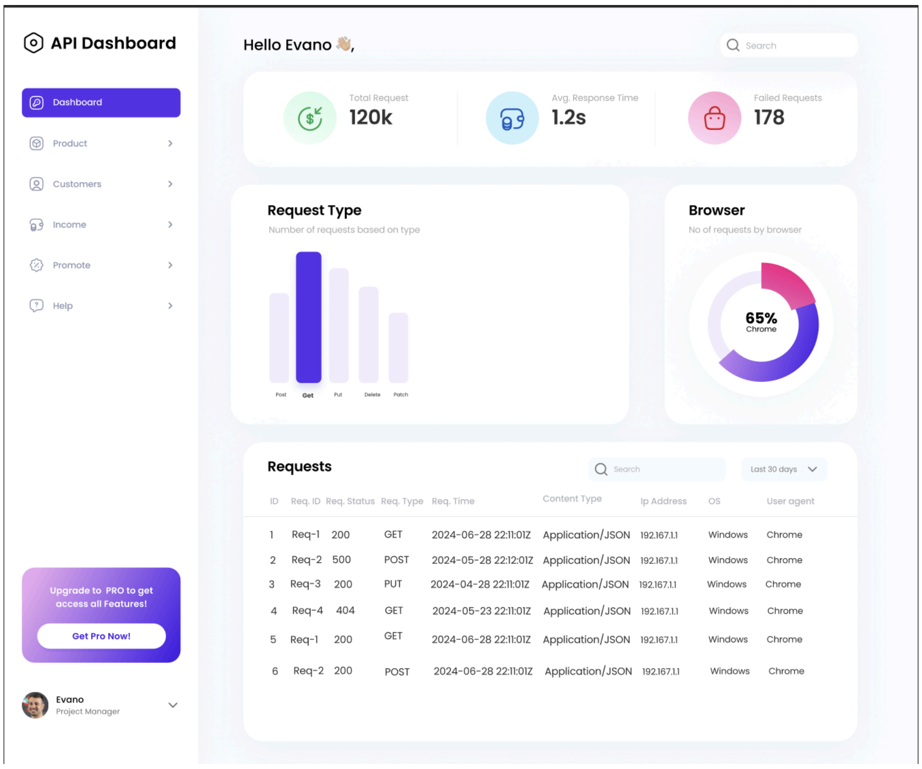
3. Bar Chart - Number of Requests by: Include an additional bar chart, allowing for flexibility to display requests based on user-selectable criteria (e.g., request status, duration, source IP address, etc.).

Backend TechStack: Python, Flask, Postgress, SQLAlchemy(ORM) - optional
Frontend TechStack - ReactJs, Html, CSS

External UI libs can be used for Doughnut, Bar chart and Table

Dashboard Layout:

Try to recreate the following using react and vanilla css and match the layout and functionality however approximate colour values and box shadows can be used to match the UI. Dashboard Link [Link]



ID	Req. ID	Req. Status	Req. Type	Req. Time	Content Type	Ip Address	OS	User agent
1	Req-1	200	GET	2024-06-28 22:11:01Z	Application/JSON	192.167.1.1	Windows	Chrome
2	Req-2	500	POST	2024-05-28 22:12:01Z	Application/JSON	192.167.1.1	Windows	Chrome
3	Req-3	200	PUT	2024-04-28 22:11:01Z	Application/JSON	192.167.1.1	Windows	Chrome
4	Req-4	404	GET	2024-05-23 22:11:01Z	Application/JSON	192.167.1.1	Windows	Chrome
5	Req-1	200	GET	2024-06-28 22:11:01Z	Application/JSON	192.167.1.1	Windows	Chrome
6	Req-2	200	POST	2024-06-28 22:11:01Z	Application/JSON	192.167.1.1	Windows	Chrome

Evaluation Criteria:

Code Quality: Clean, well-structured, maintainable, and well-documented code with adherence to best practices.

Functionality: Complete implementation of all project requirements, ensuring successful API hit capture, data storage, and comprehensive dashboard functionality.

Scalability: Design that considers future growth and potential expansion of API endpoints and usage patterns.

Error handling and validation mechanisms on both the front-end and back-end to ensure data integrity and user experience.

Ensure the UI/UX fully matches the required layout and design as closely as possible.