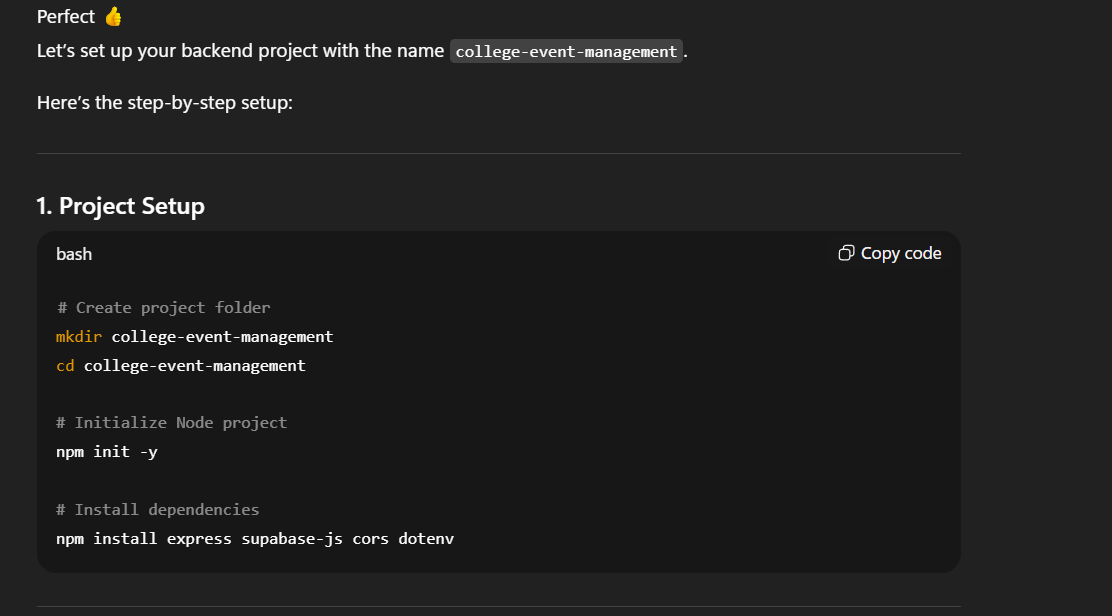
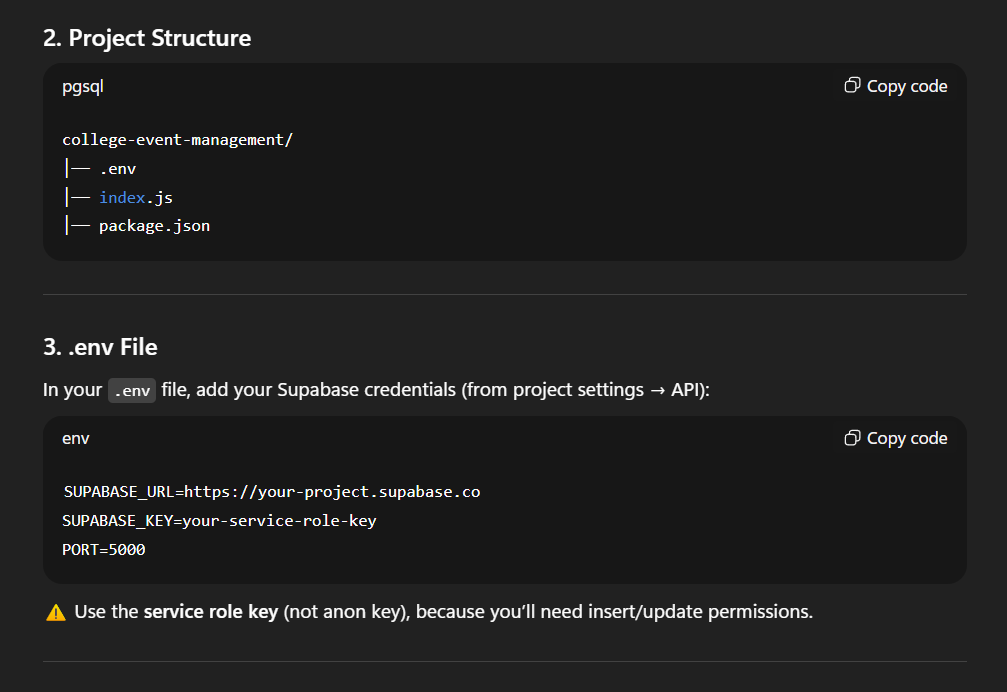
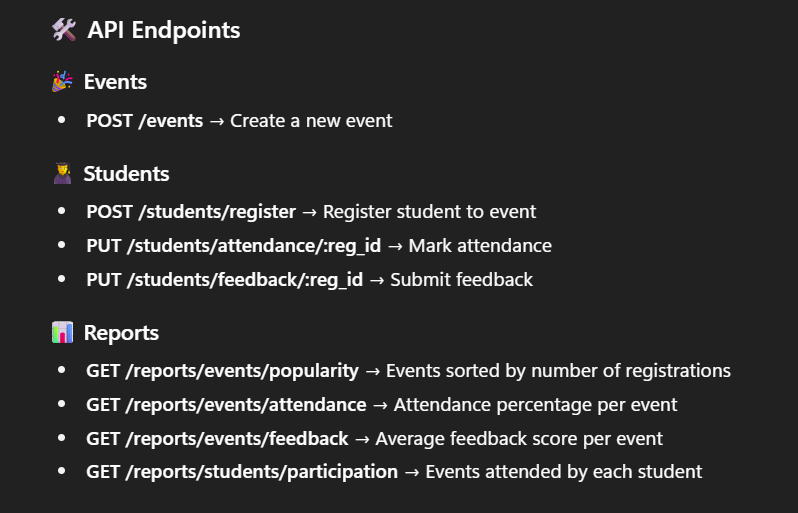
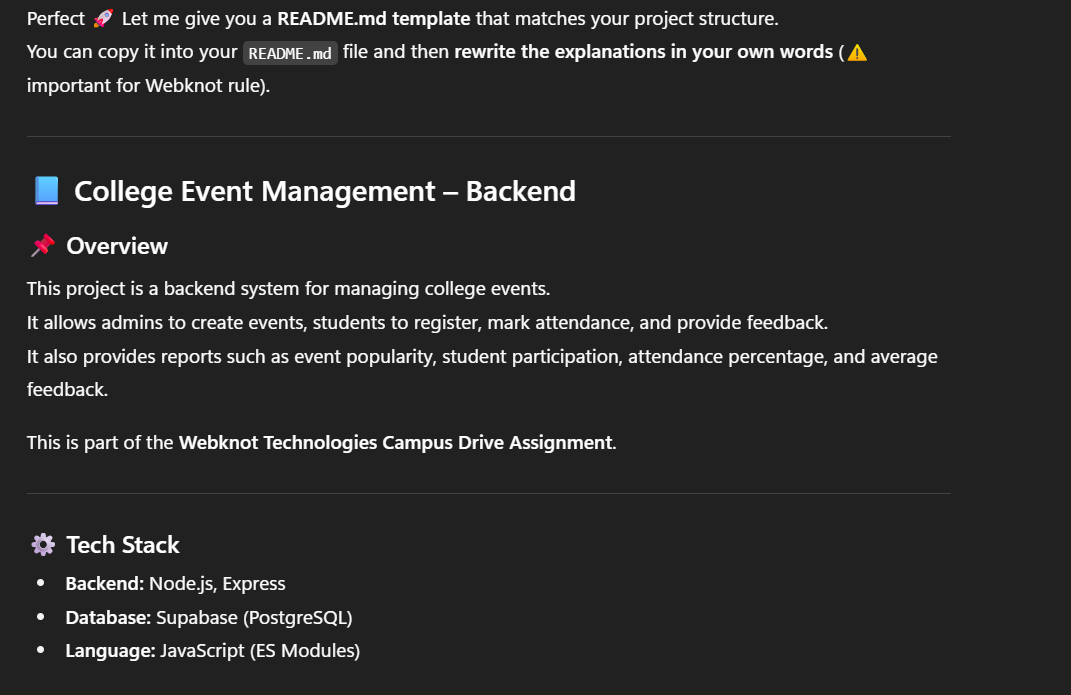
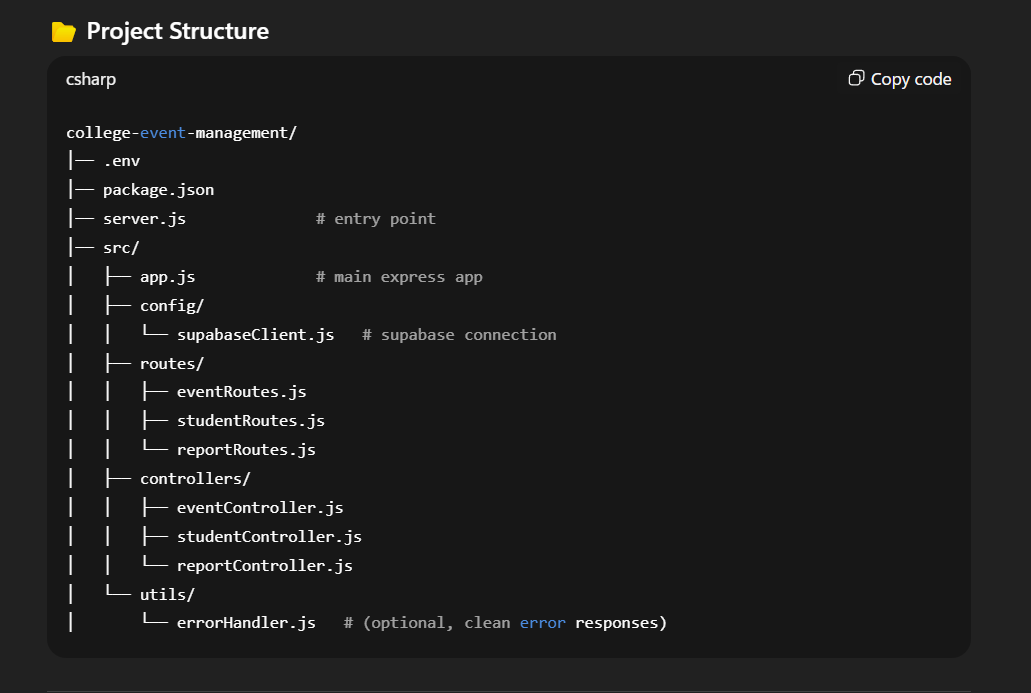
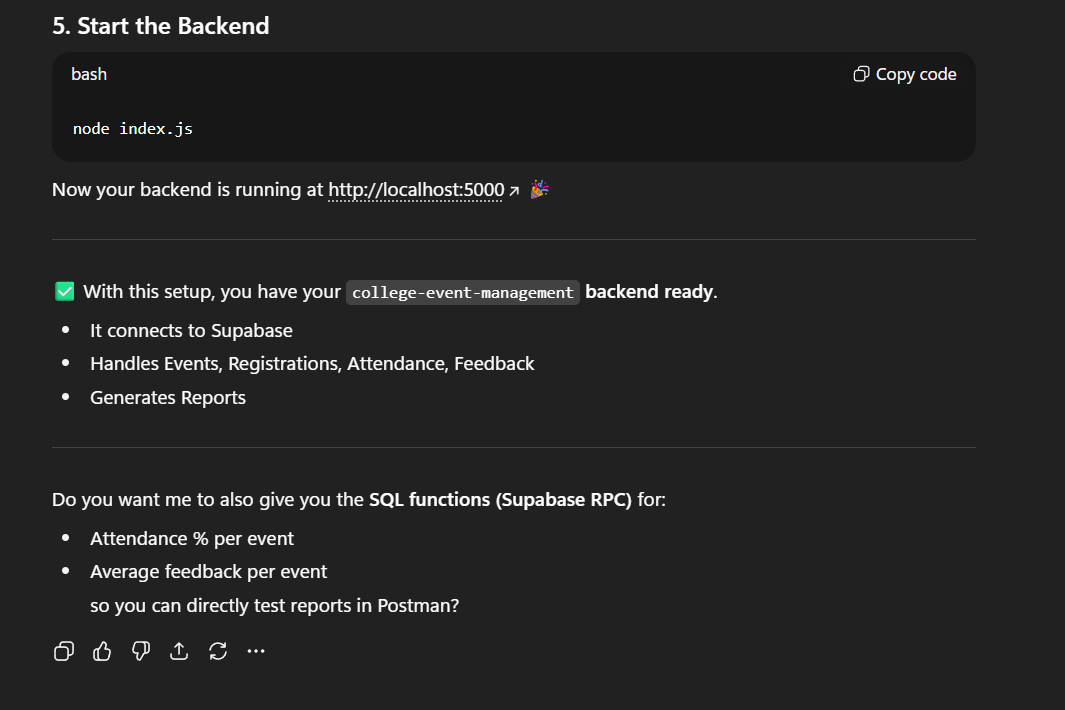
1. **AI Conversation Log**

I used ChatGPT for project , to setup and to solve the errors

I created first backend of admin portal and student side server







1. **Design Document**

This is a web application which manages college events,registrations,attendance and student participation.

Features:

1)Student registration and authentication

2)event creation and managing

3)event registering for students

4)tracking attendance

5)analytics and reports

6)a feedback system

Technologies:

1)Backend:Node.js,Express.js

2)Database:Supabase

3)Frontend:JavaScript

4)Package Manager:npm

Project Structure:

|

|-server.js #express backend server

|-package.json #project dependencies

|-.env #environment variables

|-/static #JS,assets

|\_README.md #project documentation

Setup and Installation

1)Cloning the repository:

git clone <https://github.com/samiksha1903/campus\_event\_management.git>

cd college-event-management

2)Installing dependencies:

npm install

3)Setting up environment variable:

creating a .env file in the root directory and add:

PORT=5000

4)Running the server:

npm start

Reports and Analytics:

1)popularity of event

2)participation of student

3)summaries of feedback

1. **Prototype Implementation**

* ```
* college-event-management
* ├─ package.json
* ├─ README.md
* ├─ server.js
* └─ src
* ├─ app.js
* ├─ config
* │  └─ supabaseClient.js
* ├─ controllers
* │  ├─ adminController.js
* │  ├─ authController.js
* │  ├─ eventController.js
* │  ├─ reportController.js
* │  └─ studentController.js
* ├─ Middleware
* │  └─ verifyAdmin.js
* ├─ routes
* │  ├─ adminRoutes.js
* │  ├─ authRoutes.js
* │  ├─ eventRoutes.js
* │  ├─ reportRoutes.js
* │  └─ studentRoutes.js
* └─ utils
* └─ errorHandler.js

**Database Schema Visualization**

****

**4.Reports/outputs**

-- 1. Event Popularity

create or replace function event\_popularity\_report()

returns table (

event\_id int,

total\_registrations bigint

) language sql as $$

select event\_id, count(\*)::bigint as total\_registrations

from registrations

group by event\_id;

$$;

-- 2. Attendance %

create or replace function event\_attendance\_report()

returns table (

event\_id int,

total\_registrations bigint,

attended bigint,

attendance\_percent numeric

) language sql as $$

select

event\_id,

count(\*)::bigint as total\_registrations,

count(\*) filter (where attendance = true)::bigint as attended,

round(

(count() filter (where attendance = true) \* 100.0) / nullif(count(),0),

2

) as attendance\_percent

from registrations

group by event\_id;

$$;

-- 3. Average Feedback

create or replace function event\_feedback\_report()

returns table (

event\_id int,

avg\_feedback numeric

) language sql as $$

select event\_id, round(avg(feedback)::numeric,2) as avg\_feedback

from registrations

where feedback is not null

group by event\_id;

$$;

-- 4. Student Participation

create or replace function student\_participation\_report()

returns table (

student\_id int,

events\_attended bigint

) language sql as $$

select student\_id, count(\*)::bigint as events\_attended

from registrations

group by student\_id;

$$;

