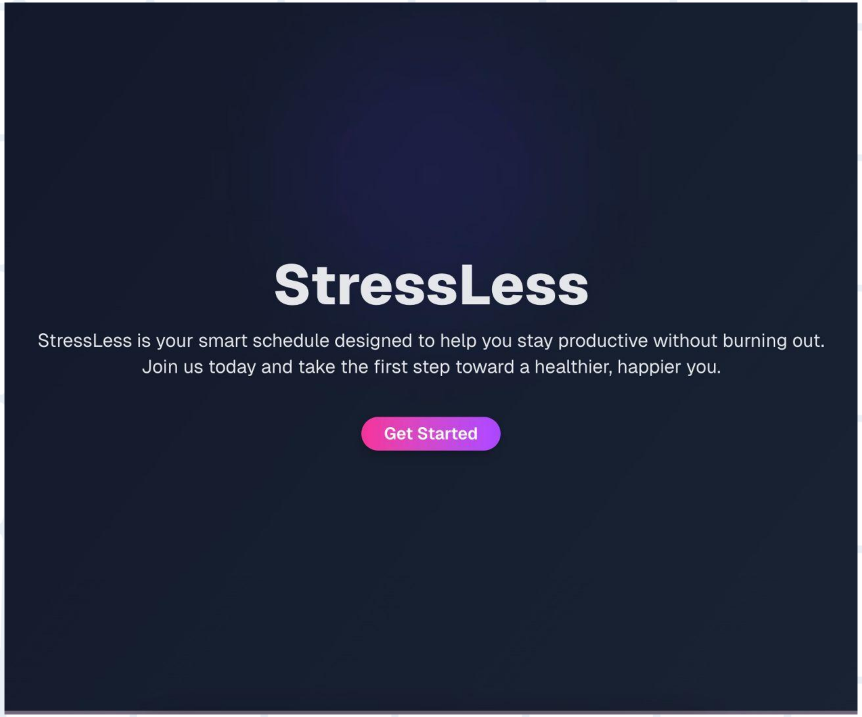
StressLess

Demo 2

Khanh Do, Nifemi Ogunmesa, Cheyanne Vinson, Madel Sibal, Alyssa Trapp, Tim Yu

Our Product

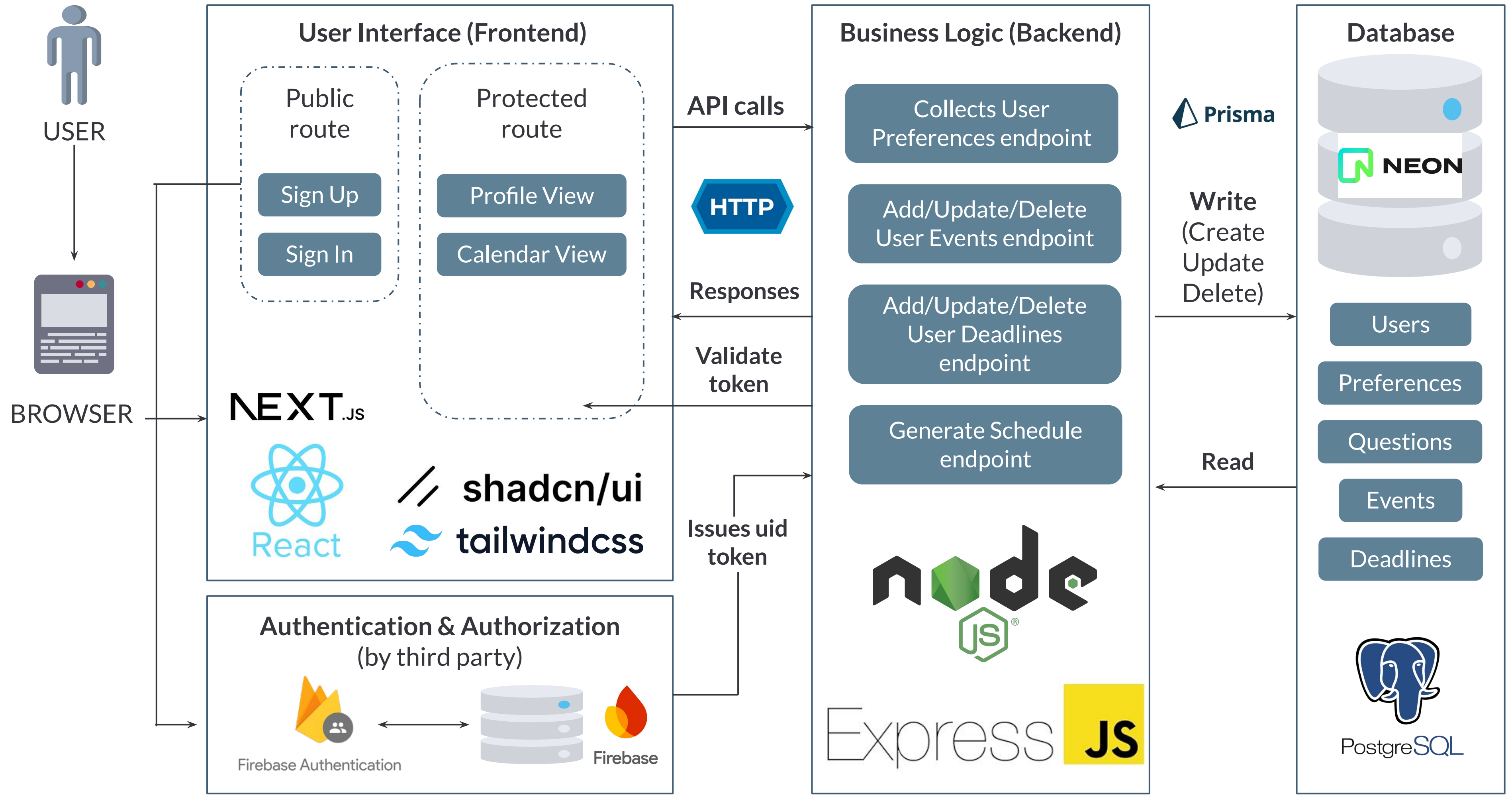
- A customized calendar tailored to your productivity preferences, break times, and scheduled events
- Creates a calendar for an individual based on their survey answers and event plans

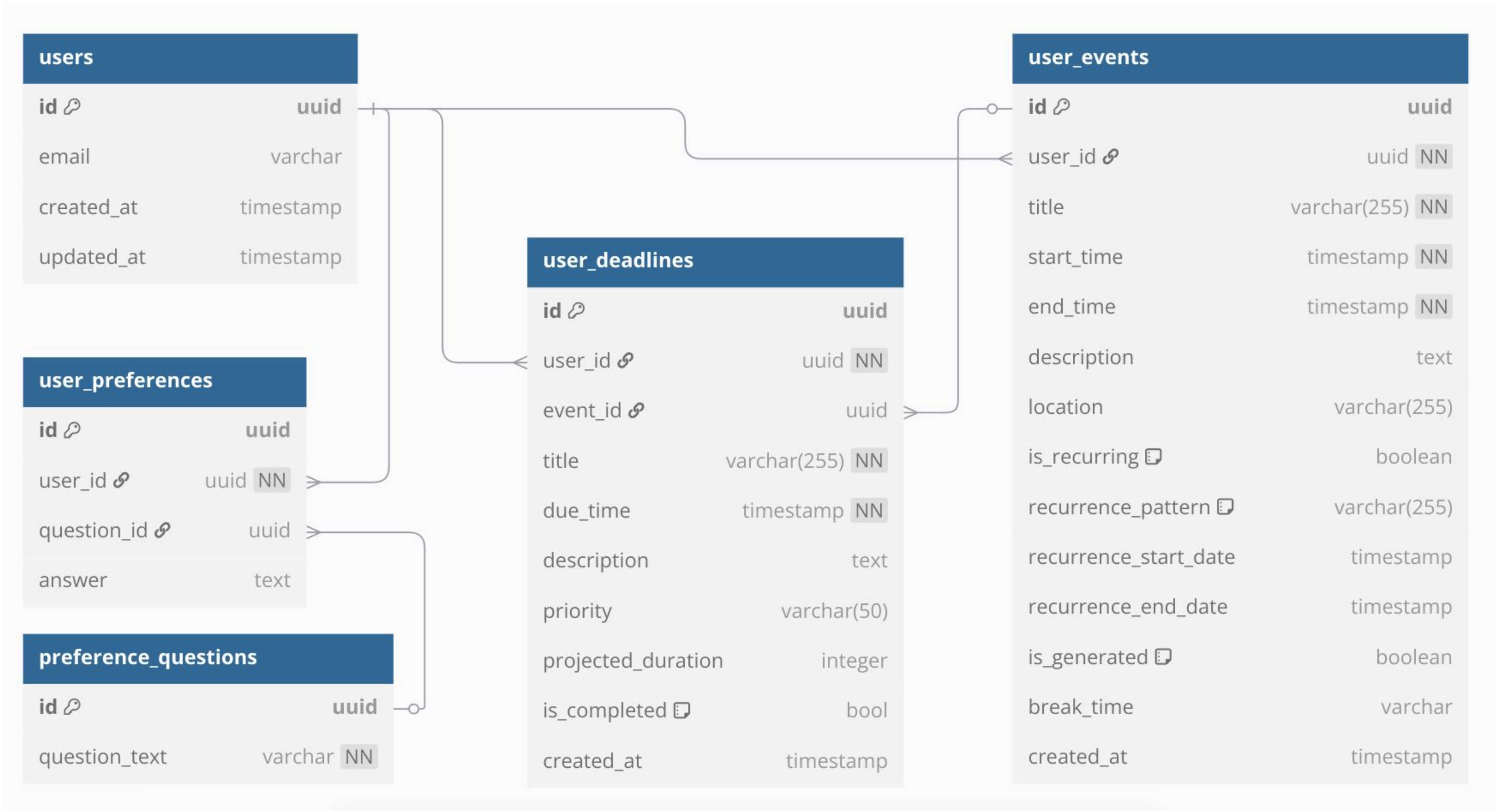


StressLess

StressLess is your smart schedule designed to help you stay productive without burning out. Join us today and take the first step toward a healthier, happier you.

StressLess Software Architecture





users		
id 🔑	uuid	
email	varchar	
created_at	timestamp	
updated_at	timestamp	

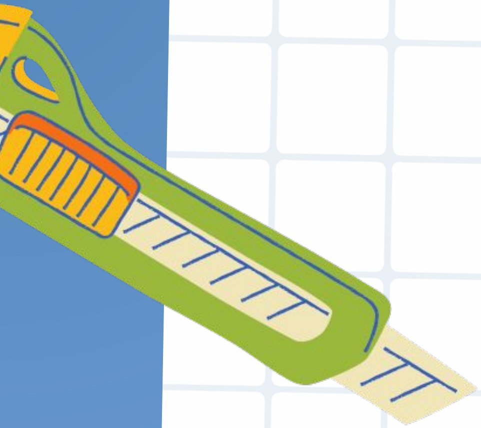
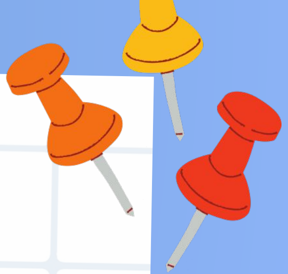
user_preferences		
id 🔑	uuid	
user_id 🔗	uuid	NN
question_id 🔗	uuid	
answer	text	

preference_questions		
id 🔑	uuid	
question_text	varchar	NN

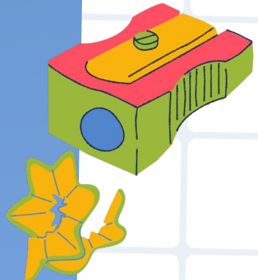
user_deadlines		
id 🔑	uuid	
user_id 🔗	uuid	NN
event_id 🔗	uuid	
title	varchar(255)	NN
due_time	timestamp	NN
description	text	
priority	varchar(50)	
projected_duration	integer	
is_completed	bool	
created_at	timestamp	

user_events		
id 🔑	uuid	
user_id 🔗	uuid	NN
title	varchar(255)	NN
start_time	timestamp	NN
end_time	timestamp	NN
description	text	
location	varchar(255)	
is_recurring	boolean	
recurrence_pattern	varchar(255)	
recurrence_start_date	timestamp	
recurrence_end_date	timestamp	
is_generated	boolean	
break_time	varchar	
created_at	timestamp	

Demo and Code Explanation

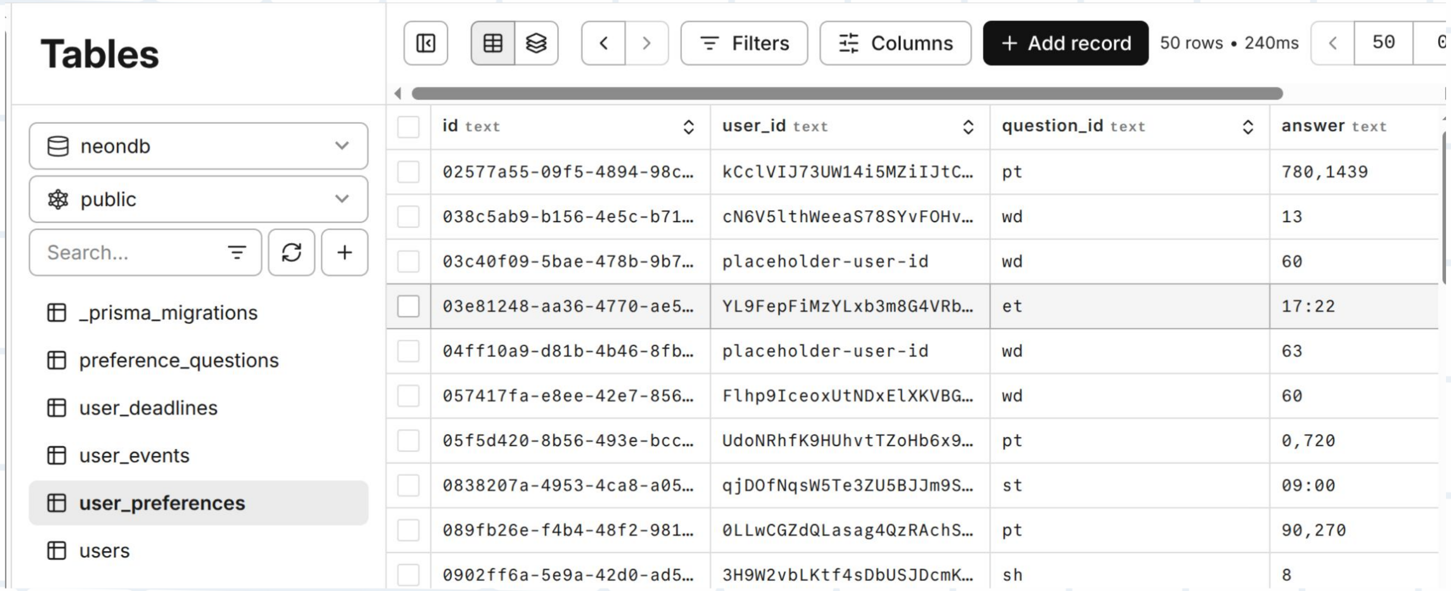



New Features



User Preferences

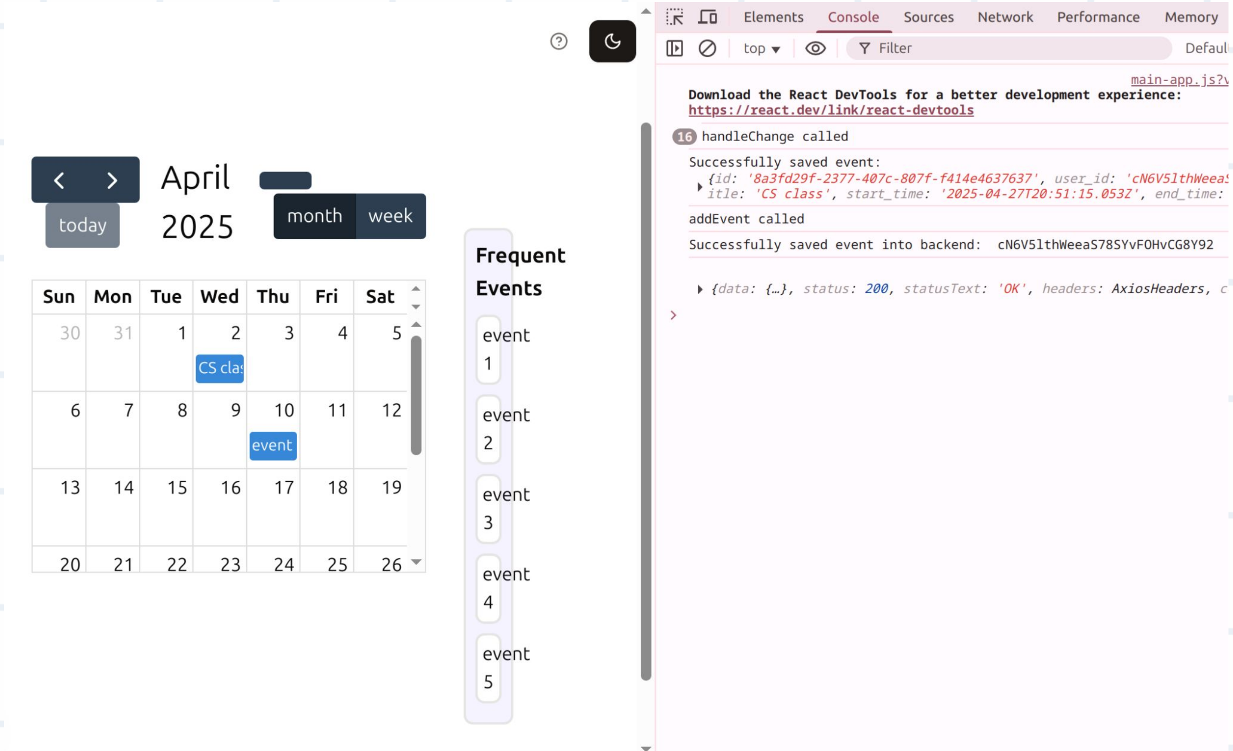
- User’s preferences answers are shown on Profile Page. Before, it uses default values. Now, it **fetches the data from the backend** to show on the frontend.
- When an user edit their preferences and save their new preferences, the database is **updated with the new answers.** When the users come back to their Profile Page, that new information replaced the old one.



id text	user_id text	question_id text	answer text
02577a55-09f5-4894-98c…	kCclVIJ73UW14i5MZiIJtC…	pt	780,1439
038c5ab9-b156-4e5c-b71…	cN6V5lthWeeaS78SYvFOHv…	wd	13
03c40f09-5bae-478b-9b7…	placeholder-user-id	wd	60
03e81248-aa36-4770-ae5…	YL9FepFiMzYLxb3m8G4VRb…	et	17:22
04ff10a9-d81b-4b46-8fb…	placeholder-user-id	wd	63
057417fa-e8ee-42e7-856…	Flhp9IceoxUtNDxElXKVBG…	wd	60
05f5d420-8b56-493e-bcc…	UdoNRhfK9HUhvtTZoHb6x9…	pt	0,720
0838207a-4953-4ca8-a05…	qjDOfNqsW5Te3ZU5BJJm9S…	st	09:00
089fb26e-f4b4-48f2-981…	0LLwCGZdQLasag4QzRAchS…	pt	90,270
0902ff6a-5e9a-42d0-ad5…	3H9W2vbLKtf4sDbUSJDcmK…	sh	8

Adding Events

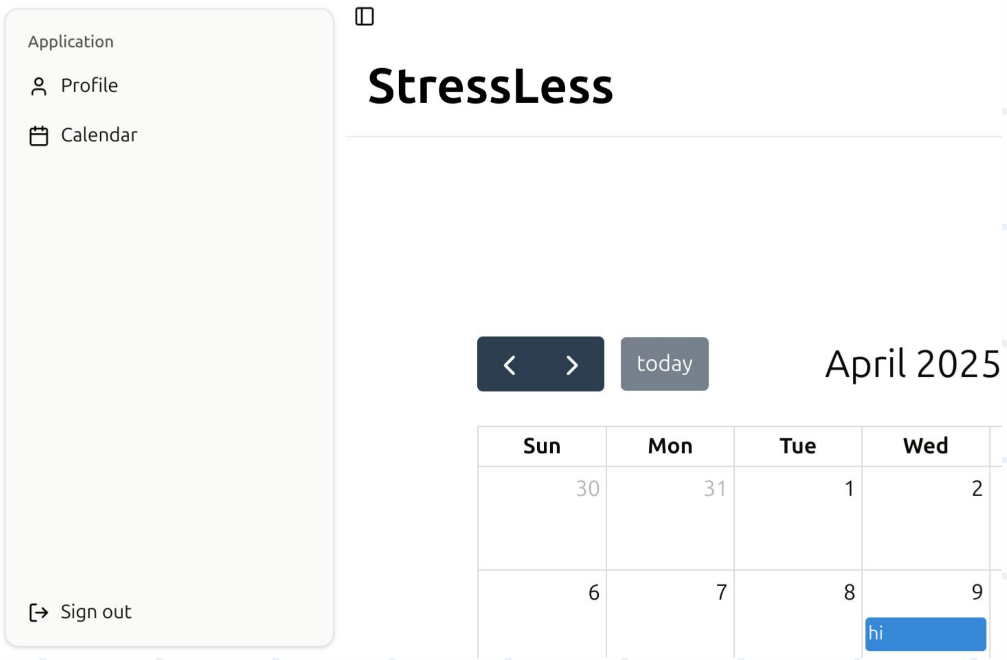
- When an user add an event (either by dragging or submitting a form), that event is added to the database.



```
                                                              main-app.js?v
Download the React DevTools for a better development experience:
https://react.dev/link/react-devtools
16 handleChange called
Successfully saved event:
 ▸ {id: '8a3fd29f-2377-407c-807f-f414e4637637', user_id: 'cN6V5lthWeeaS
   itle: 'CS class', start_time: '2025-04-27T20:51:15.053Z', end_time:
addEvent called
Successfully saved event into backend:  cN6V5lthWeeaS78SYvFOHvCG8Y92

 ▸ {data: {…}, status: 200, statusText: 'OK', headers: AxiosHeaders, c
>
```

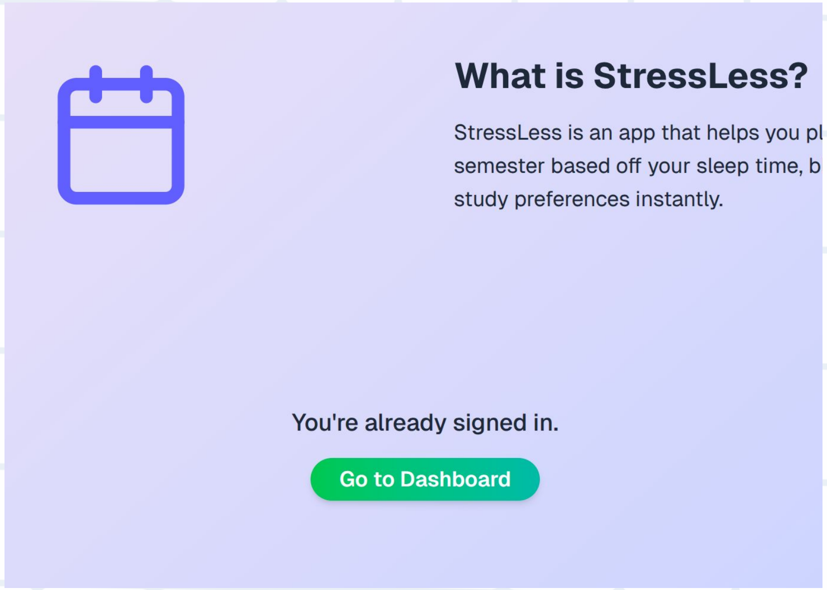
Sidebar

- There is now a side-bar (or called hamburger menu) that centralizes all routes that we have: Calendar, Profile Page. Upon signed-in, user is routed to `/dashboard` where they can click on the side-bar button to show navigations to other places of the app.



Logout & Protected Routes

- There is now a Log out button that log the user out.
- If the user is not signed-in (or not authenticated), they can only route to `/` . Meaning if they try to go to `/dashboard`, they will be automatically redirected.
- If the user is signed-in (or is authenticated), they will not see the Sign up, Sign in form on the landing page `/`, but they will have the option to click on a button that says `Go to dashboard`.



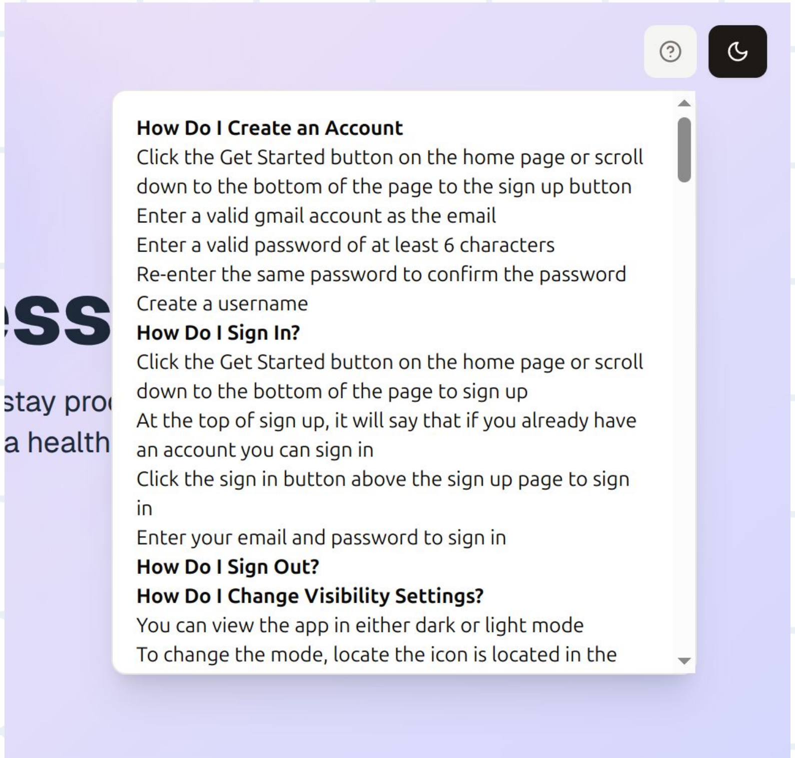
What is StressLess?

StressLess is an app that helps you pl
semester based off your sleep time, b
study preferences instantly.

You're already signed in.

Help Button

- There is now a Help button on the top right corner of the app. Upon clicking, a pop-up will appear that says how to use the app in detailed.

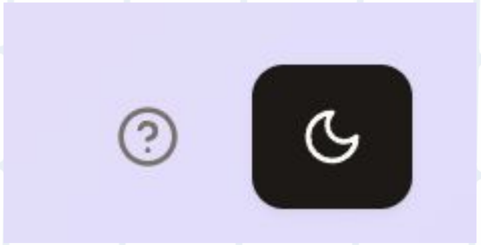


ess

stay pro

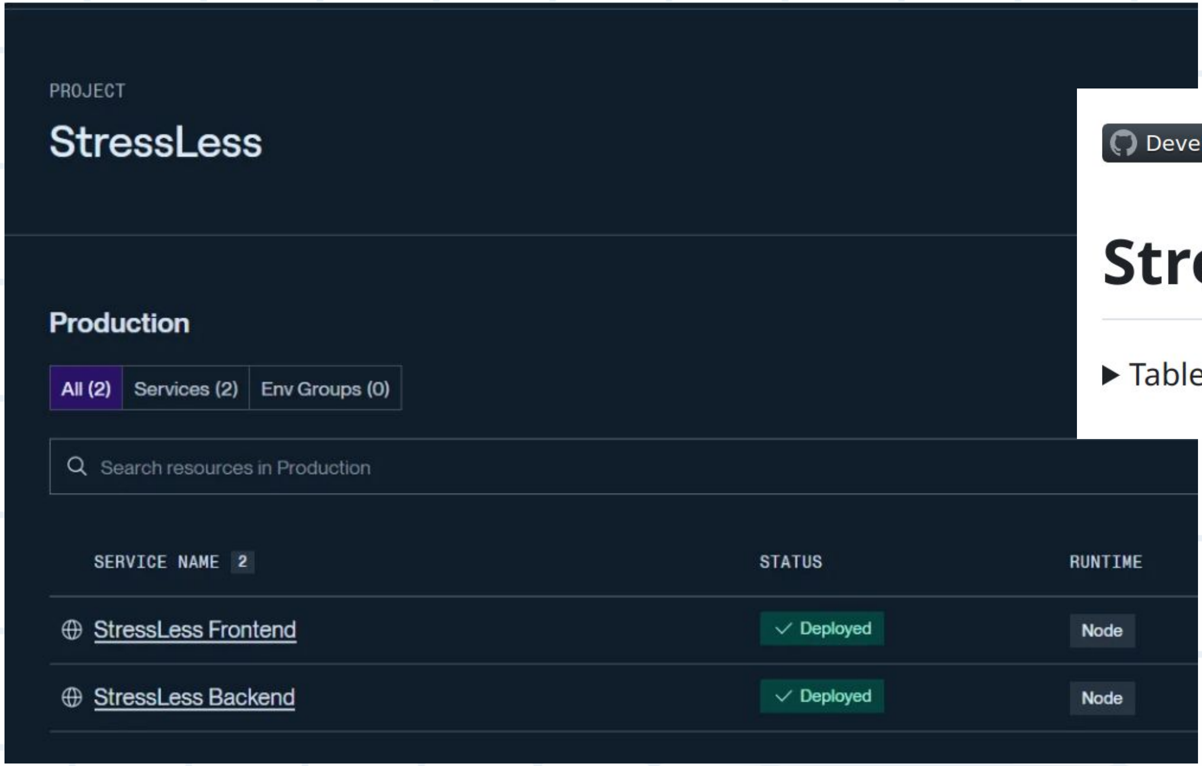
a health

How Do I Create an Account
Click the Get Started button on the home page or scroll down to the bottom of the page to the sign up button
Enter a valid gmail account as the email
Enter a valid password of at least 6 characters
Re-enter the same password to confirm the password
Create a username
How Do I Sign In?
Click the Get Started button on the home page or scroll down to the bottom of the page to sign up
At the top of sign up, it will say that if you already have an account you can sign in
Click the sign in button above the sign up page to sign in
Enter your email and password to sign in
How Do I Sign Out?
How Do I Change Visibility Settings?
You can view the app in either dark or light mode
To change the mode, locate the icon is located in the



Continuous Deployment

- All commits to the `main` branch automatically trigger re-deployment via **GitHub Actions**.
- Both frontend and backend are hosted on **Render**, ensuring live updates with every merge.



PROJECT

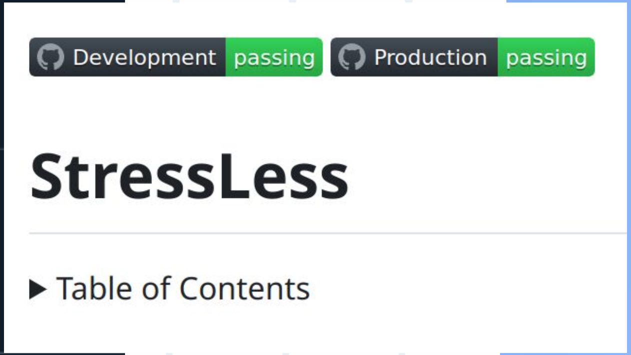
StressLess

Production

All (2) | Services (2) | Env Groups (0)

Search resources in Production

SERVICE NAME 2	STATUS	RUNTIME
StressLess Frontend	✓ Deployed	Node
StressLess Backend	✓ Deployed	Node



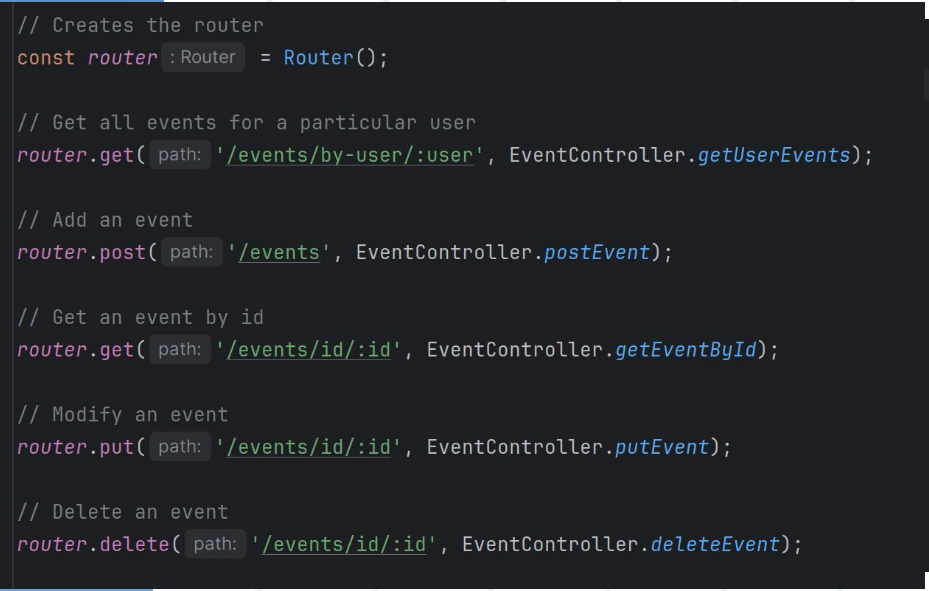
Development passing | Production passing

StressLess

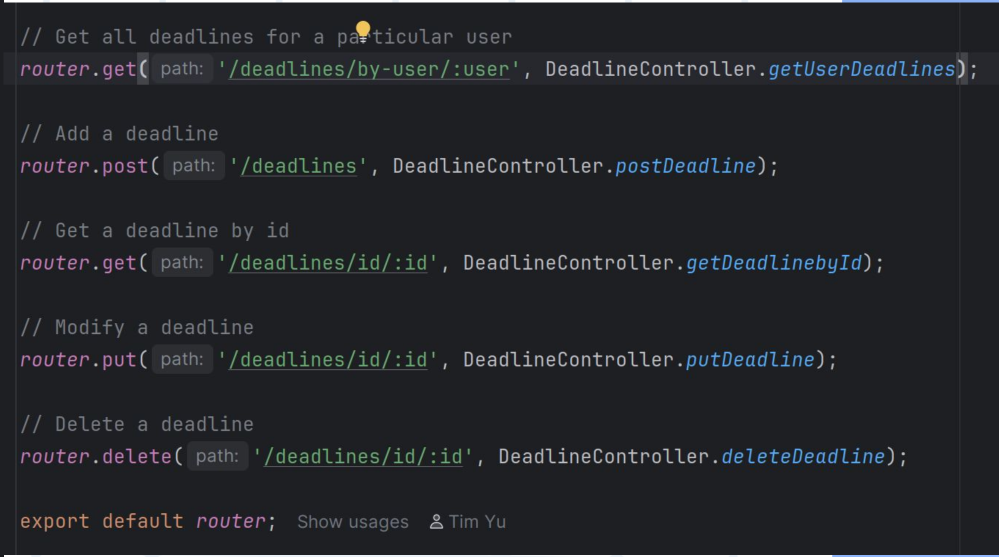
▶ Table of Contents

Backend: Events & Deadlines

API Endpoints Create-Read-Update-Delete (CRUD) operation for events and deadlines, ready to be integrated end-to-end.

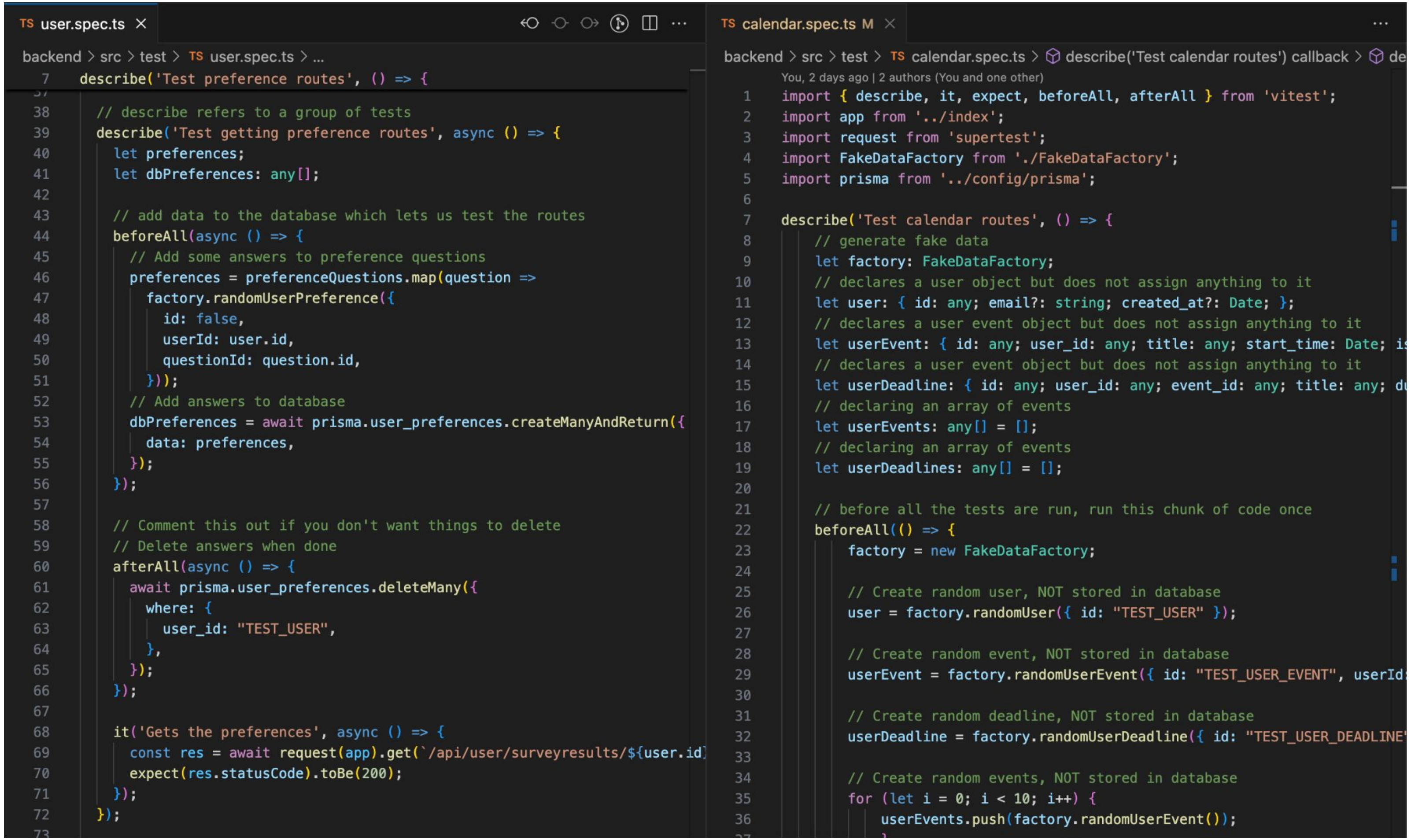


```
// Creates the router
const router : Router  = Router();

// Get all events for a particular user
router.get( path: '/events/by-user/:user', EventController.getUserEvents);

// Add an event
router.post( path: '/events', EventController.postEvent);

// Get an event by id
router.get( path: '/events/id/:id', EventController.getEventById);

// Modify an event
router.put( path: '/events/id/:id', EventController.putEvent);

// Delete an event
router.delete( path: '/events/id/:id', EventController.deleteEvent);
```



```
// Get all deadlines for a particular user
router.get( path: '/deadlines/by-user/:user', DeadlineController.getUserDeadlines);

// Add a deadline
router.post( path: '/deadlines', DeadlineController.postDeadline);

// Get a deadline by id
router.get( path: '/deadlines/id/:id', DeadlineController.getDeadlinebyId);

// Modify a deadline
router.put( path: '/deadlines/id/:id', DeadlineController.putDeadline);

// Delete a deadline
router.delete( path: '/deadlines/id/:id', DeadlineController.deleteDeadline);

export default router;  Show usages   Tim Yu
```

Backend Unit Test



TS user.spec.ts ×

backend > src > test > TS user.spec.ts > ...

```ts
describe('Test preference routes', () => {

  // describe refers to a group of tests
  describe('Test getting preference routes', async () => {
    let preferences;
    let dbPreferences: any[];

    // add data to the database which lets us test the routes
    beforeAll(async () => {
      // Add some answers to preference questions
      preferences = preferenceQuestions.map(question =>
        factory.randomUserPreference({
          id: false,
          userId: user.id,
          questionId: question.id,
        }));
      // Add answers to database
      dbPreferences = await prisma.user_preferences.createManyAndReturn({
        data: preferences,
      });
    });

    // Comment this out if you don't want things to delete
    // Delete answers when done
    afterAll(async () => {
      await prisma.user_preferences.deleteMany({
        where: {
          user_id: "TEST_USER",
        },
      });
    });

    it('Gets the preferences', async () => {
      const res = await request(app).get(`/api/user/surveyresults/${user.id}
      expect(res.statusCode).toBe(200);
    });
  });
```

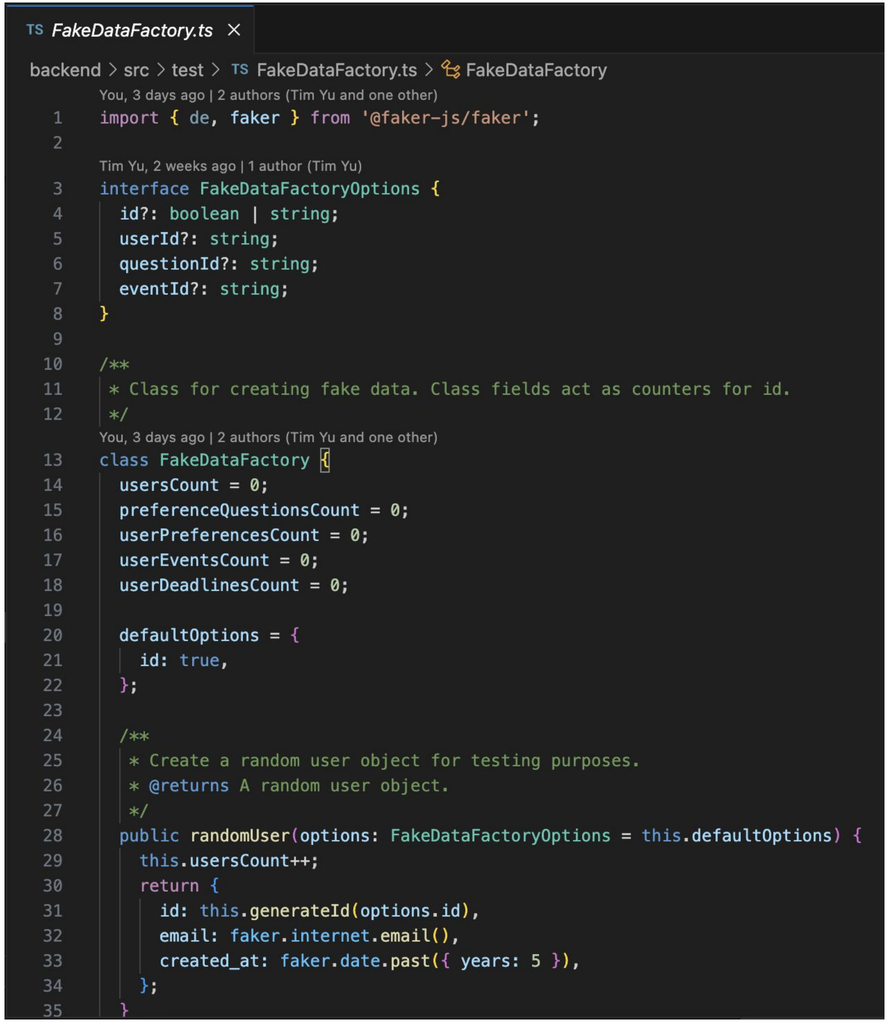
TS calendar.spec.ts M ×

backend > src > test > TS calendar.spec.ts > describe('Test calendar routes') callback > de

You, 2 days ago | 2 authors (You and one other)

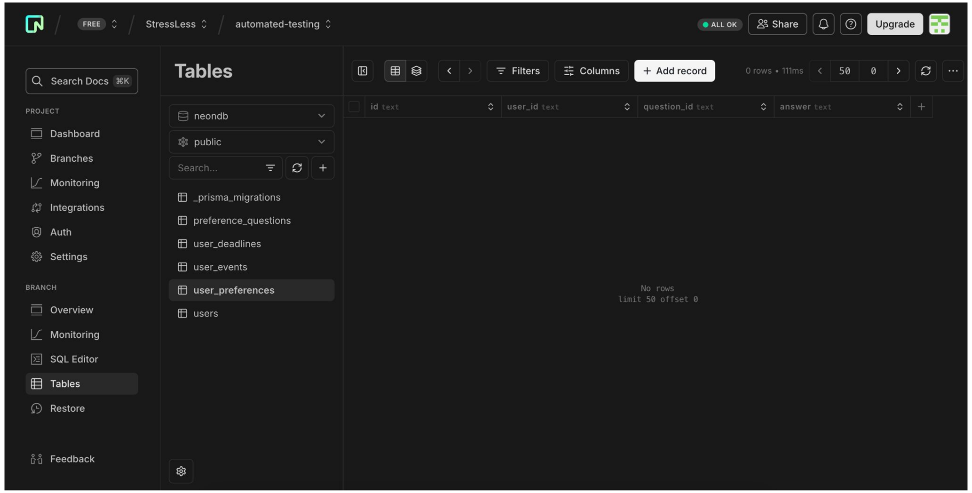
```ts
import { describe, it, expect, beforeAll, afterAll } from 'vitest';
import app from '../index';
import request from 'supertest';
import FakeDataFactory from './FakeDataFactory';
import prisma from '../config/prisma';

describe('Test calendar routes', () => {
  // generate fake data
  let factory: FakeDataFactory;
  // declares a user object but does not assign anything to it
  let user: { id: any; email?: string; created_at?: Date; };
  // declares a user event object but does not assign anything to it
  let userEvent: { id: any; user_id: any; title: any; start_time: Date; is
  // declares a user event object but does not assign anything to it
  let userDeadline: { id: any; user_id: any; event_id: any; title: any; d
  // declaring an array of events
  let userEvents: any[] = [];
  // declaring an array of events
  let userDeadlines: any[] = [];

  // before all the tests are run, run this chunk of code once
  beforeAll(() => {
    factory = new FakeDataFactory;

    // Create random user, NOT stored in database
    user = factory.randomUser({ id: "TEST_USER" });

    // Create random event, NOT stored in database
    userEvent = factory.randomUserEvent({ id: "TEST_USER_EVENT", userId

    // Create random deadline, NOT stored in database
    userDeadline = factory.randomUserDeadline({ id: "TEST_USER_DEADLINE"

    // Create random events, NOT stored in database
    for (let i = 0; i < 10; i++) {
      userEvents.push(factory.randomUserEvent());
```

Backend Unit Test



TS FakeDataFactory.ts
backend > src > test > TS FakeDataFactory.ts > FakeDataFactory

```ts
You, 3 days ago | 2 authors (Tim Yu and one other)
import { de, faker } from '@faker-js/faker';

Tim Yu, 2 weeks ago | 1 author (Tim Yu)
interface FakeDataFactoryOptions {
  id?: boolean | string;
  userId?: string;
  questionId?: string;
  eventId?: string;
}

/**
 * Class for creating fake data. Class fields act as counters for id.
 */
You, 3 days ago | 2 authors (Tim Yu and one other)
class FakeDataFactory {
  usersCount = 0;
  preferenceQuestionsCount = 0;
  userPreferencesCount = 0;
  userEventsCount = 0;
  userDeadlinesCount = 0;

  defaultOptions = {
    id: true,
  };

  /**
   * Create a random user object for testing purposes.
   * @returns A random user object.
   */
  public randomUser(options: FakeDataFactoryOptions = this.defaultOptions) {
    this.usersCount++;
    return {
      id: this.generateId(options.id),
      email: faker.internet.email(),
      created_at: faker.date.past({ years: 5 }),
    };
  }
```



```
id text | user_id text | question_id text | answer text

No rows
limit 50 offset 0
```

Q & A