

Tech Workshop #7

Braden Guliano

Director of Technical Development

Kappa Theta Pi

Before we start...



Prerequisites



**Your first deploy
is just a sign-up away.**

Plan Type

☒ I'm working on personal projects Hobby

☐ I'm working on commercial projects Pro

Your Name

Continue

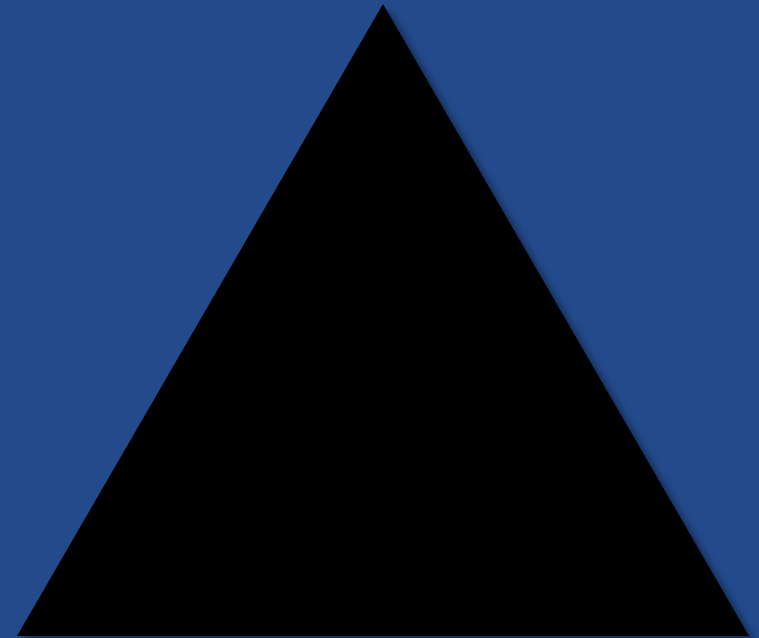
By joining, you agree to our [Terms of Service](#) and [Privacy Policy](#)

Have a complex company use case? [Get Enterprise grade assistance](#) ⓘ

Vercel account

Vercel

- What is it?
 - Cloud platform for deploying web apps
 - Vercel built Next.js, so compatibility right out of the gate
 - Handles hosting, builds, domains, etc.
- But why?
 - Instant deploys
 - Serverless functions (`/api`)
 - Edge network (fast loading)
 - Dynamic content (not static like GH pages)
 - Built-in environment var management



Neon

- What is it?
 - Serverless, Postgres database (SQL)
 - Built for modern cloud apps
 - Fully compatible with Vercel
- But why?
 - Serverless compute (pay-as-you-go)
 - Branching (like GH for databases)
 - Connection pooling



Prisma

- What is it?
 - Object-Relational Mapper (ORM) for working with databases
 - Translates TypeScript into SQL queries (no manual SQL required)
- But why?
 - Works seamlessly with Vercel and Neon
 - Autocomplete and type-safe querying
 - Migrations for version control

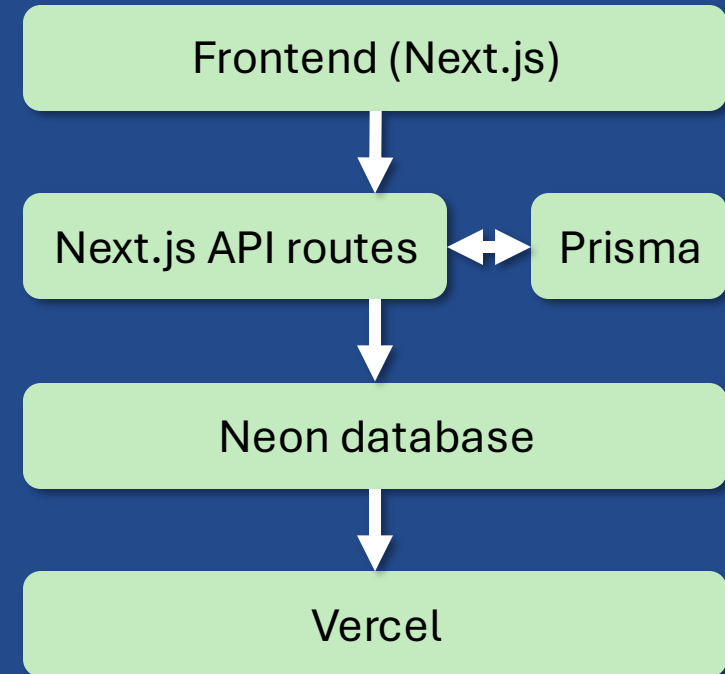
```
// Create a new todo
await prisma.todo.create({
  data: { title: 'Learn Prisma' }
})

// Get all todos
const todos = await prisma.todo.findMany()
```



How do they all work together?

1. You deploy your app to Vercel
2. You connect Neon through Vercel Integrations
3. Vercel automatically sets up a `DATABASE_URL` env var
4. You build custom API routes that use Prisma to talk to Neon



Vercel + Neon setup

The screenshot shows the Vercel dashboard for a project named "tech-workshop-7". The interface is dark-themed and includes a top navigation bar with links for Overview, Deployments, Analytics, Speed Insights, Logs, Observability, Firewall, AI, Storage, Flags, and Settings. A search bar and utility icons are also present in the top right.

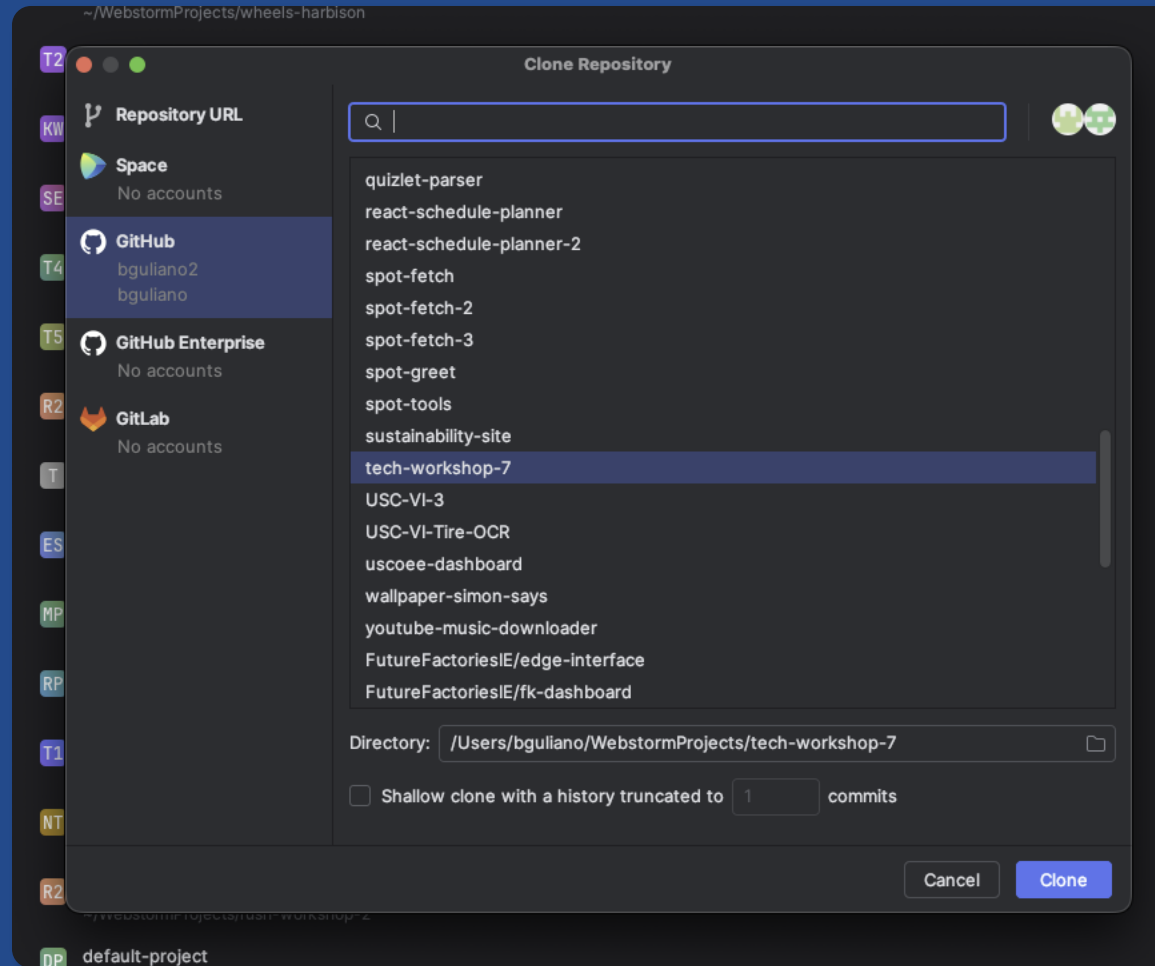
The main section is titled "tech-workshop-7" and includes buttons for Repository, Usage, Domains, and Visit. Below this, the "Production Deployment" section is active, showing a deployment of "nextjs-boilerplate-qbjteoqns-bgulianos-projects.vercel.app". The deployment status is "Ready" and was created 18m ago by "bguliano". The source is "main" with commit "ce11f81".

Below the deployment details, there are three tabs: "Deployment Settings", "3 Recommendations", and "Deployments". A message states: "To update your Production Deployment, push to the main branch."

The bottom section features three cards: "Firewall 24h" (with "Enable Bot Protection" and "Firewall is active"), "Observability 6h" (showing "Edge Requests: 43", "Function Invocations: 0", and "Error Rate: 0%"), and "Analytics" (with a toggle for "Track visitors and page views" set to "Enable").

The "Active Branches" section is visible at the bottom but is currently empty.

Vercel integration



Setup

```
npm install -g vercel
```

```
vercel link
```

```
vercel env pull .env.development.local
```

```
npm install @neondatabase/serverless @prisma/client prisma
```

```
npx prisma init
```

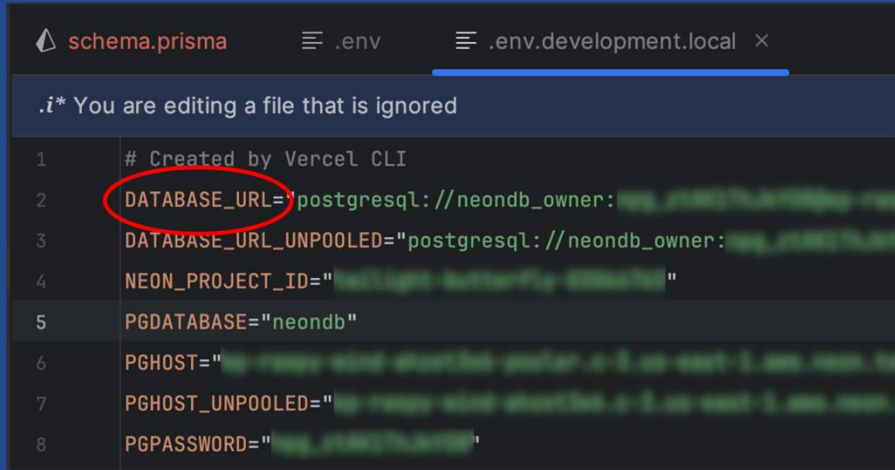
Setup

```
vercel env pull .env.development.local
```

**DO NOT COMMIT
ENV FILES!**



Setup



```
schema.prisma  .env  .env.development.local x
.i* You are editing a file that is ignored
1 # Created by Vercel CLI
2 DATABASE_URL="postgresql://neondb_owner:neondb_password@localhost:5432/neondb"
3 DATABASE_URL_UNPOOLED="postgresql://neondb_owner:neondb_password@localhost:5432/neondb"
4 NEON_PROJECT_ID="neondb"
5 PGDATABASE="neondb"
6 PGHOST="localhost"
7 PGHOST_UNPOOLED="localhost"
8 PGPASSWORD="neondb_password"
```

Copy DATABASE_URL env var from .env.development.local to .env



```
schema.prisma  .env x  .env.development.local
.i* You are editing a file that is ignored
1 DATABASE_URL="postgresql://neondb_owner:neondb_password@localhost:5432/neondb"
2
```

Prisma init output

Next steps:

1. Install `dotenv`, and add `import "dotenv/config";` to your `prisma.config.ts` file to load environment variables from `.env`.
2. Run `prisma dev` to start a local Prisma Postgres server.
3. Define models in the `schema.prisma` file.
4. Run `prisma migrate dev` to migrate your local Prisma Postgres database.
5. Tip: Explore how you can extend the ORM with scalable connection pooling, global caching, and a managed serverless Postgres database. Read: <https://pris.ly/cli/beyond-orm>

Prisma init output

Next steps:

1. ~~Install `dotenv`, and add `import "dotenv/config";` to your `prisma.config.ts` file to load environment variables from `.env`.~~
2. ~~Run `prisma dev` to start a local Prisma Postgres server.~~
3. Define models in the `schema.prisma` file.
4. Run `prisma migrate dev` to migrate your local Prisma Postgres database.
5. ~~Tip: Explore how you can extend the ORM with scalable connection pooling, global caching, and a managed serverless Postgres database. Read: <https://pris.ly/cli/beyond-orm>~~

What is a migration?

A **migration** is a set of instructions that tell your database **how to change its structure** (tables, columns, relations) to match your current Prisma schema.

4. Run `prisma migrate dev` to migrate your local Prisma Postgres database.

```
npx prisma migrate dev --name init
```

What is a migration?

```
npx prisma migrate dev --name init
```

```
model Todo {  
  id      String  @id @default(cuid())  
  title   String  
  completed Boolean @default(false)  
  createdAt DateTime @default(now())  
}
```



```
-- CreateTable  
CREATE TABLE "Todo" (  
  "id" TEXT NOT NULL,  
  "title" TEXT NOT NULL,  
  "completed" BOOLEAN NOT NULL DEFAULT false,  
  "createdAt" TIMESTAMP(3) NOT NULL DEFAULT CURRENT_TIMESTAMP,  
  
  CONSTRAINT "Todo_pkey" PRIMARY KEY ("id")  
);
```


Prisma migrate

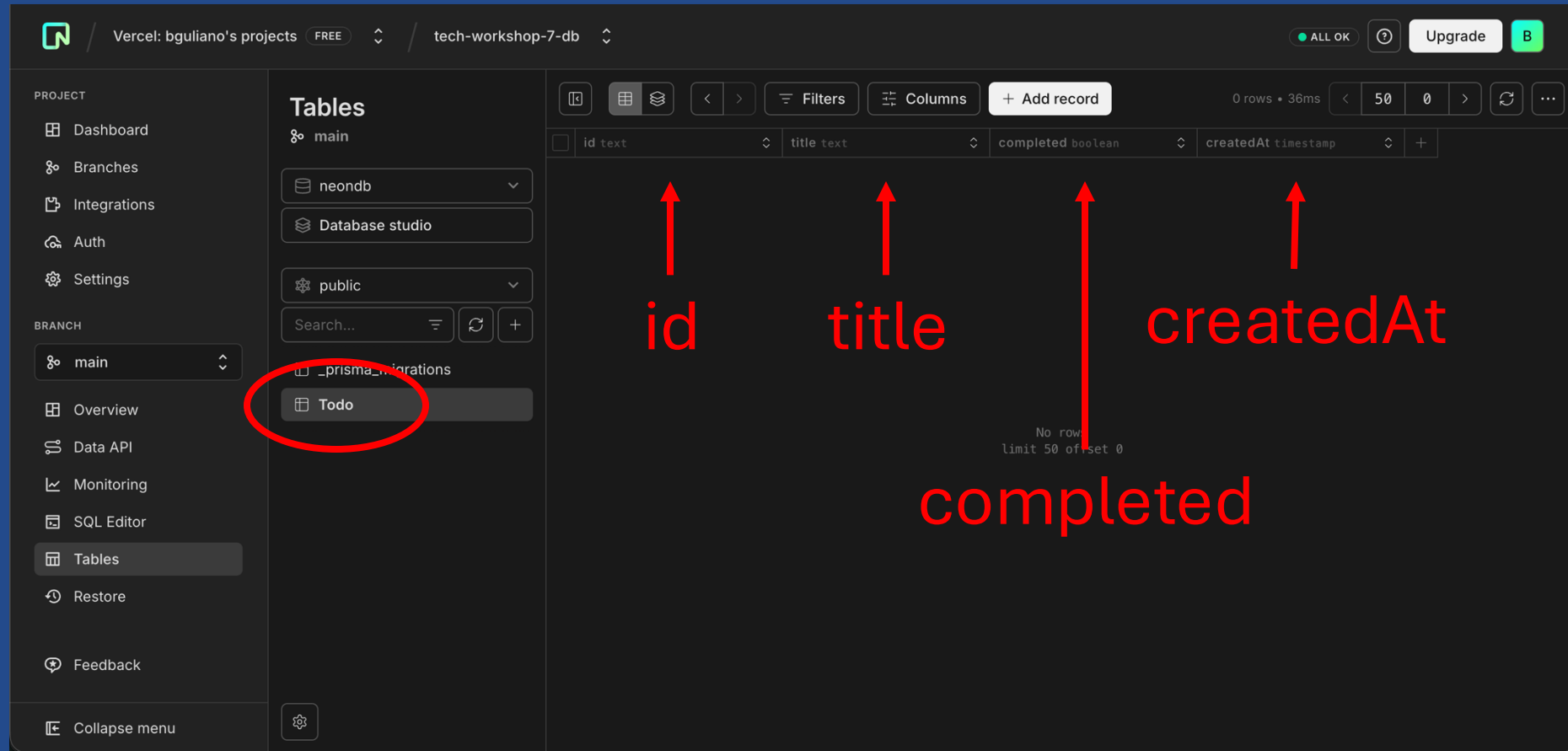
- Looks at your models in `schema.prisma`
- Creates a SQL migration to match your Neon database
- Runs that migration on your Neon instance
- Updates your local Prisma history in `prisma/migrations/`

4. Run `prisma migrate dev` to migrate your local Prisma Postgres database.

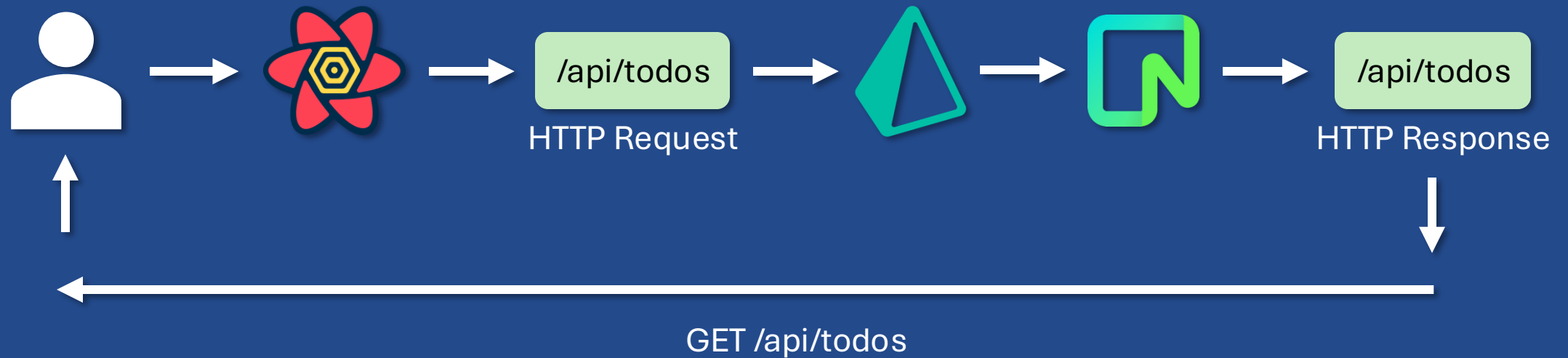
```
npx prisma migrate dev --name init
```

Your database is now in sync with your schema.

Prisma migrate



API routing



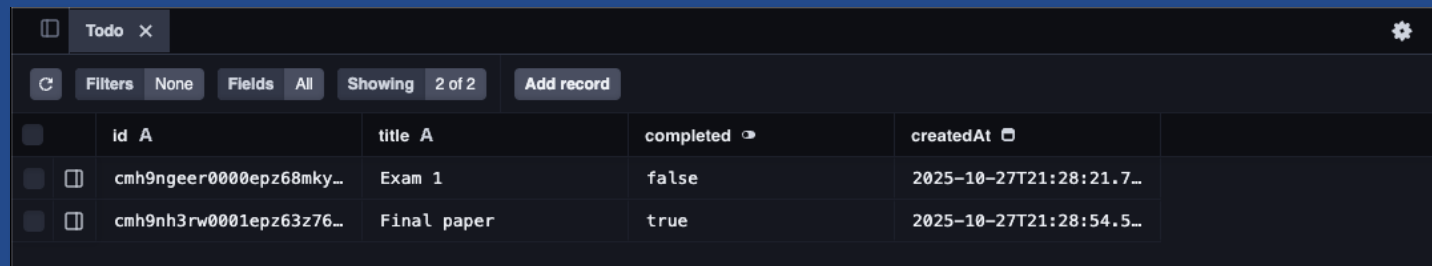
UI connection

```
npm install @tanstack/react-query
```



Local Prisma studio

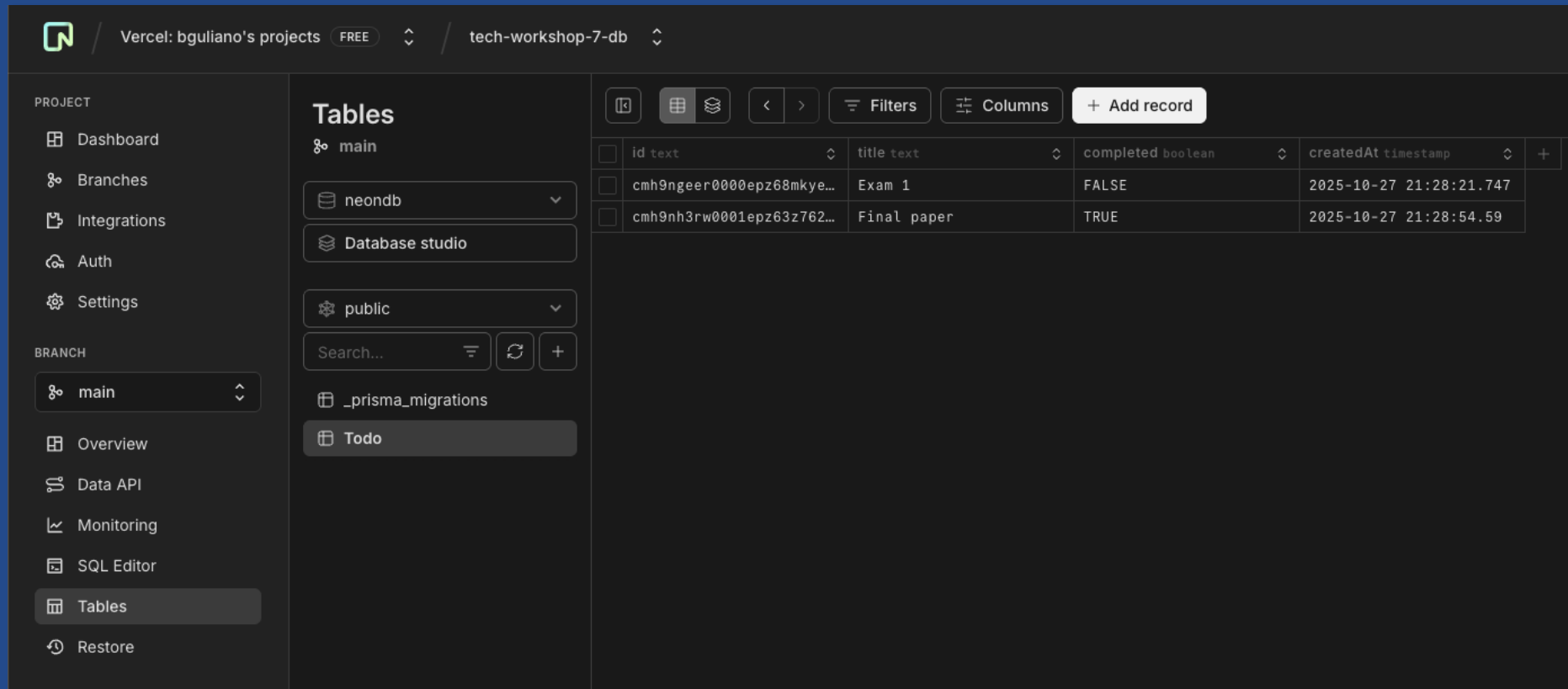
```
npx prisma studio
```



The screenshot shows the Prisma Studio web interface. At the top, there's a tab labeled 'Todo' with a close button. Below the tab, there's a toolbar with a refresh icon, 'Filters', 'None', 'Fields', 'All', 'Showing 2 of 2', and an 'Add record' button. The main area displays a table with two columns: 'id' and 'title'. The first row has 'cmh9ngeer0000epz68mky...' and 'Exam 1'. The second row has 'cmh9nh3rw0001epz63z76...' and 'Final paper'. The 'completed' column shows 'false' and 'true' respectively. The 'createdAt' column shows timestamps.

	id A	title A	completed ▾	createdAt ▾
<input type="checkbox"/>	cmh9ngeer0000epz68mky...	Exam 1	false	2025-10-27T21:28:21.7...
<input type="checkbox"/>	cmh9nh3rw0001epz63z76...	Final paper	true	2025-10-27T21:28:54.5...

Neon cloud storage



The screenshot displays the Neon Cloud Storage interface. The top navigation bar shows the Neon logo, the project name "Vercel: bguliano's projects" with a "FREE" label, and the database name "tech-workshop-7-db". The left sidebar contains a "PROJECT" section with links to Dashboard, Branches, Integrations, Auth, and Settings, and a "BRANCH" section with a dropdown for "main" and links to Overview, Data API, Monitoring, SQL Editor, Tables (highlighted), and Restore. The main content area is titled "Tables" and shows the "main" branch. It includes a dropdown for "neondb", a "Database studio" button, a "public" schema dropdown, a search bar, and a list of tables: "_prisma_migrations" and "Todo" (highlighted). The "Todo" table is expanded, showing a table view with columns: id (text), title (text), completed (boolean), and createdAt (timestamp). The table contains two records: one with id "cmh9ngeer0000epz68mkye..." and title "Exam 1", and another with id "cmh9nh3rw0001epz63z762..." and title "Final paper".

PROJECT

- Dashboard
- Branches
- Integrations
- Auth
- Settings

BRANCH

- main
- Overview
- Data API
- Monitoring
- SQL Editor
- Tables
- Restore

Tables

main

neondb

Database studio

public

Search...

_prisma_migrations

Todo

id text	title text	completed boolean	createdAt timestamp
cmh9ngeer0000epz68mkye...	Exam 1	FALSE	2025-10-27 21:28:21.747
cmh9nh3rw0001epz63z762...	Final paper	TRUE	2025-10-27 21:28:54.59

How does this work in production?

- Yes, the DATABASE_URL env var is needed to talk to the Neon database
- But...in production, Vercel automatically (and securely) injects it into the deployed backend environment
- All API functions run on Vercel's servers, not on the client's device



Questions?

