Squeezing the model

How to create a full stack app generator

Mihovil Ilakovac

Full-stack developer who likes building stuff

I like contributing to open source (Directus, Caprover)

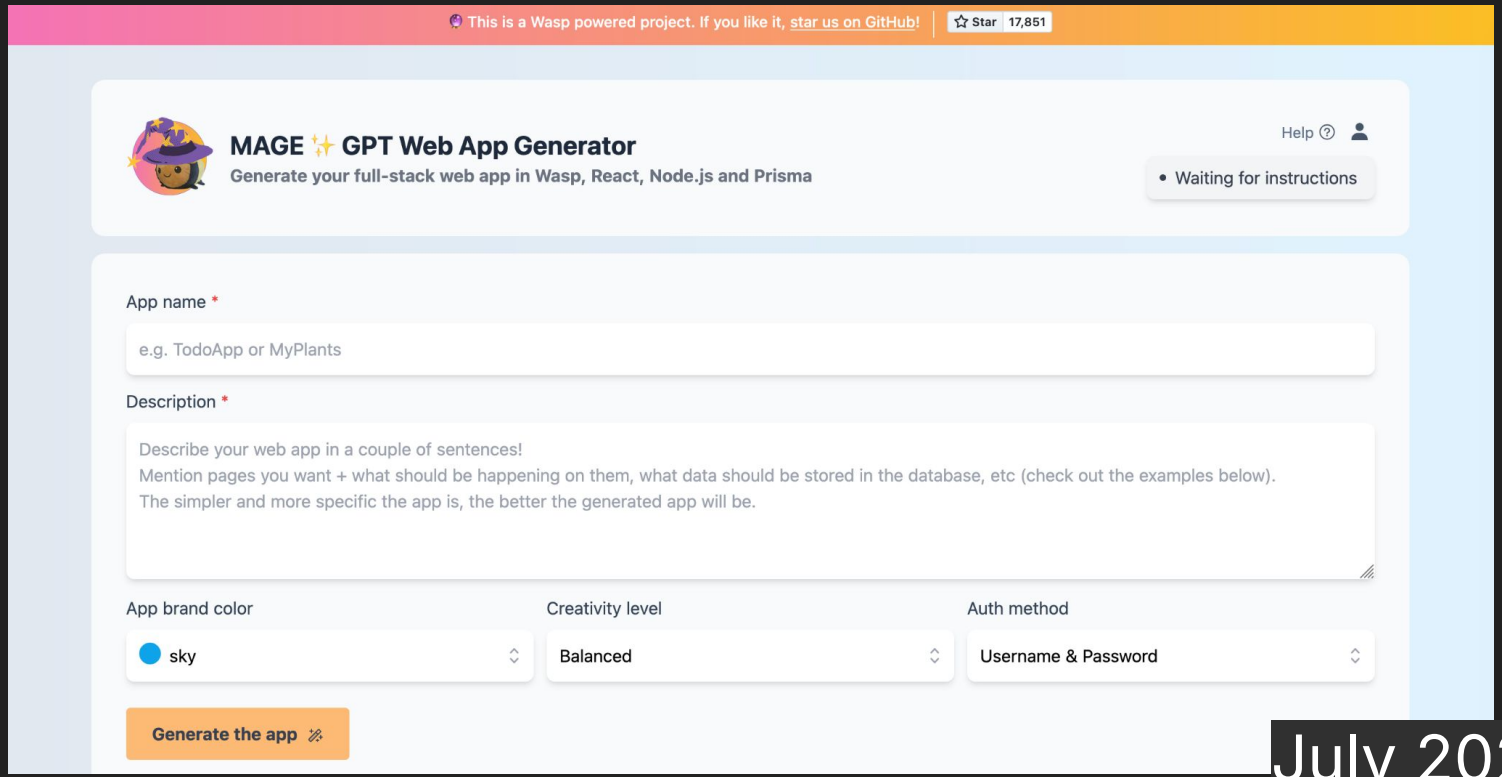
Freelanced, worked at enterprises and startups

Wasp =}

ilakovac.com

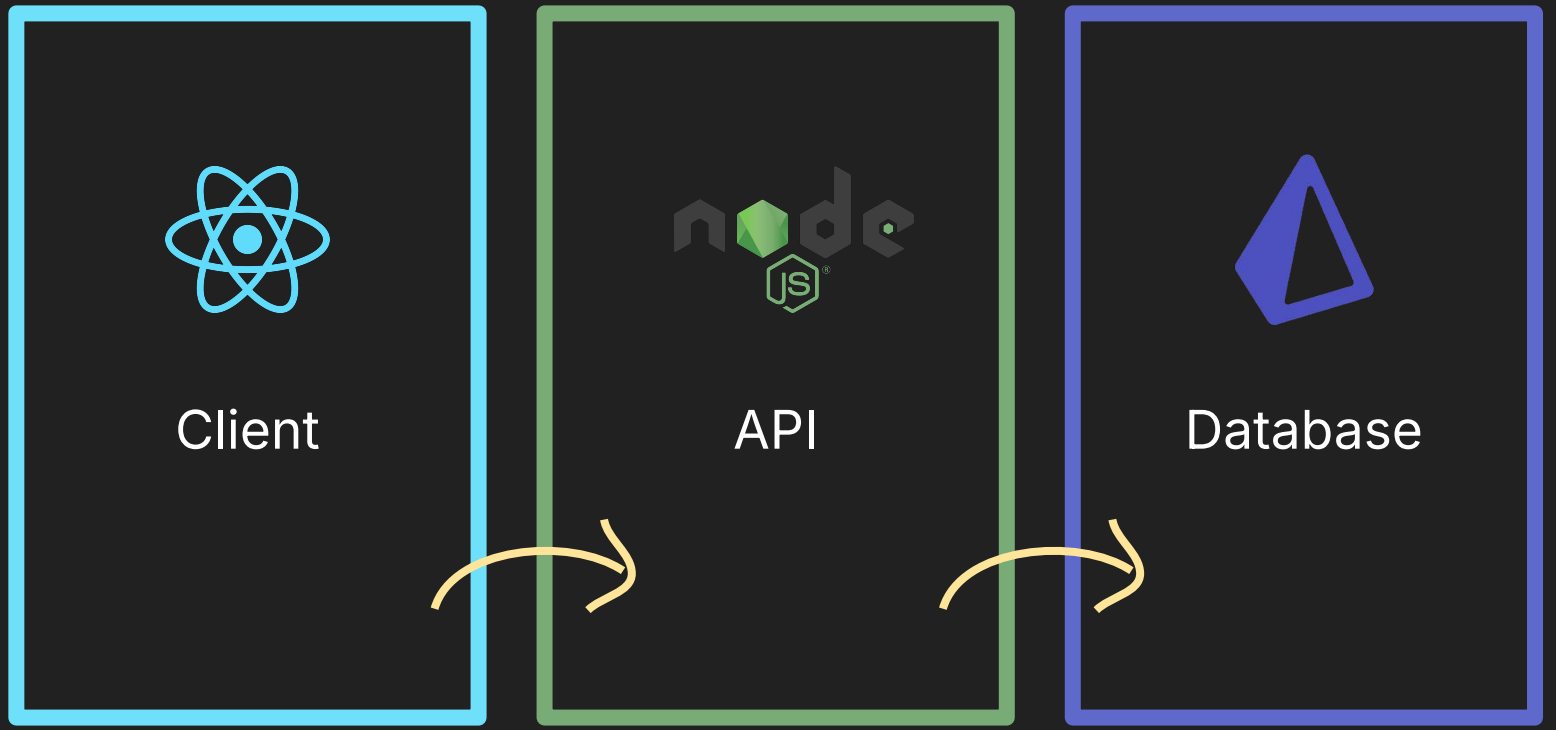
@infomiho

Let’s talk about the Magic App Generator



July 2023

We wanted to generate full-stack apps



When GPT 3.5 was all the rage

ChatGPT: Optimizing Language M | Web Dev Help Offered

chat.openai.com/chat

+ New chat

Web Dev Help Offered

Explaining Node JS Popularity

Clear conversations

Dark mode

OpenAI Discord

Updates & FAQ

Log out

V Generate code for a page with a play button over a picture

Sure! Here is some HTML code that will create a page with a play button over an image:

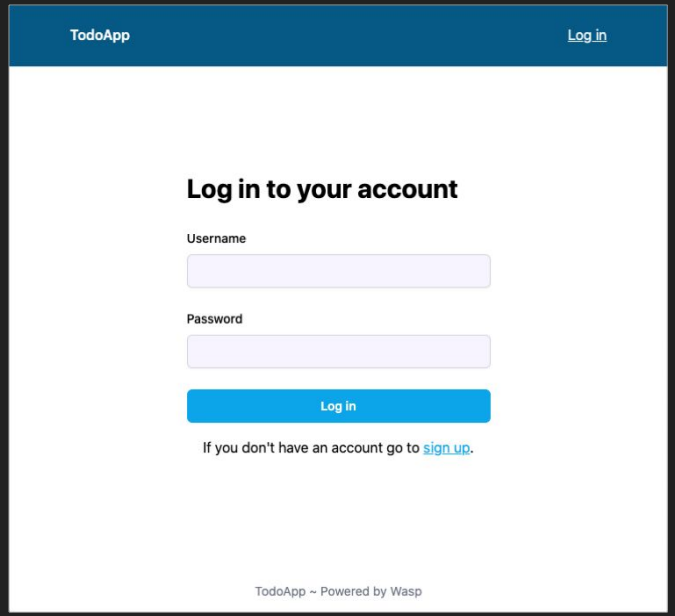
Copy code

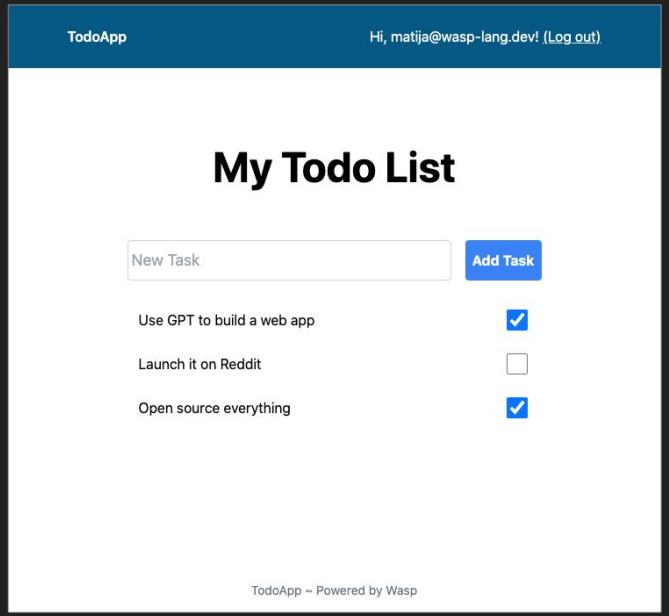
```
<!DOCTYPE html>
<html>
<head>
  <title>Play Button Over Image</title>
</head>
<body>
  <style>
    /* Style the play button */
    .play-button {
      position: absolute;
      top: 50%;
      left: 50%;
      transform: translate(-50%, -50%);
      background-color: #fff;
      border: none;
      outline: none;
      cursor: pointer;
      box-shadow: 0 3px 6px rg
px rgba(0,0,0,0.23);
      border-radius: 50%;
```

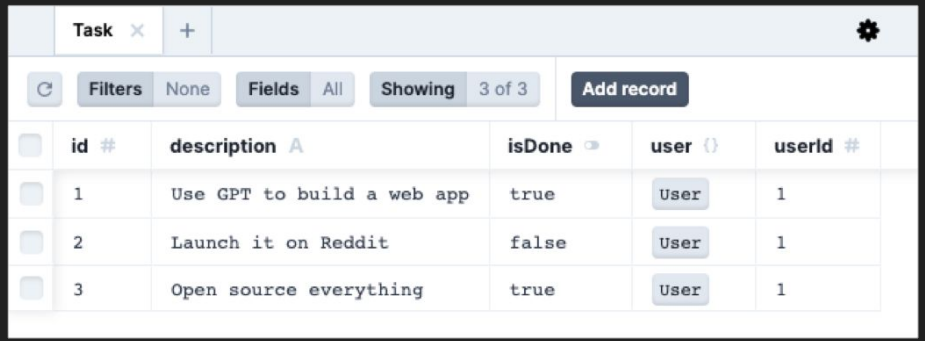
Regenerate response

ChatGPT Dec 15 Version. Free Research Preview. Our goal is to make AI systems more natural and safe to interact with. Your feedback will help us improve.

It wasn’t looking good at first…


TodoApp
Log in
Log in to your account
Username
Password
Log in
If you don't have an account go to sign up.
TodoApp ~ Powered by Wasp


TodoApp
Hi, matija@wasp-lang.dev! (Log out)
My Todo List
New Task
Add Task
Use GPT to build a web app
Launch it on Reddit
Open source everything
TodoApp ~ Powered by Wasp


Task
Filters None
Fields All
Showing 3 of 3
Add record
id
description
isDone
user
userId
1 Use GPT to build a web app true User 1
2 Launch it on Reddit false User 1
3 Open source everything true User 1

FIFA2024 Hi, John Doe! (Log out)

Tournament Name

Description of the tournament

Home Team vs Away Team Match Date

FIFA2024 ~ Powered

snarlOff Log in

Powered by Wasp

Nexus John Doe! (Log out)

Inve

Item Remove

Item Remove

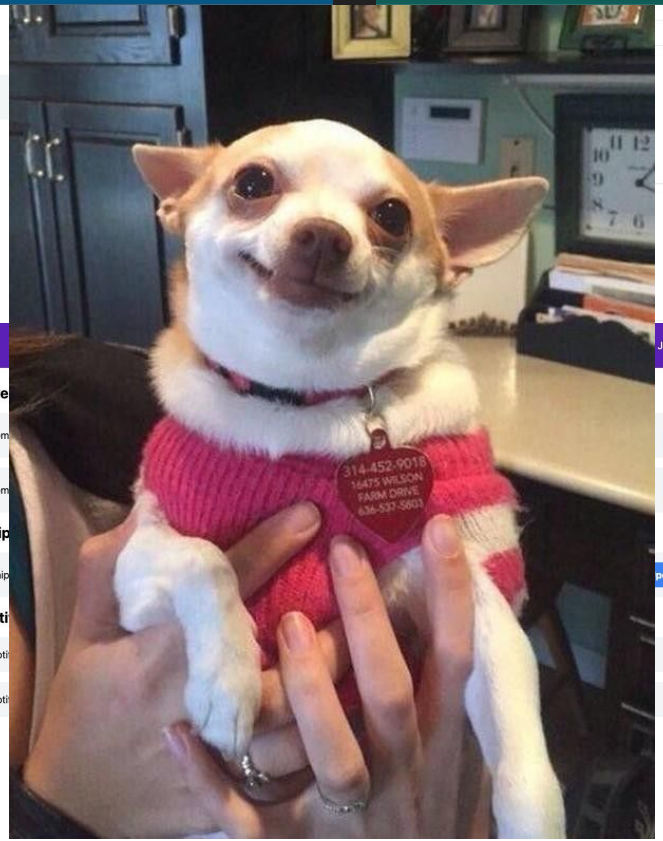
Ship

Ship pdate Status

Noti

Noti

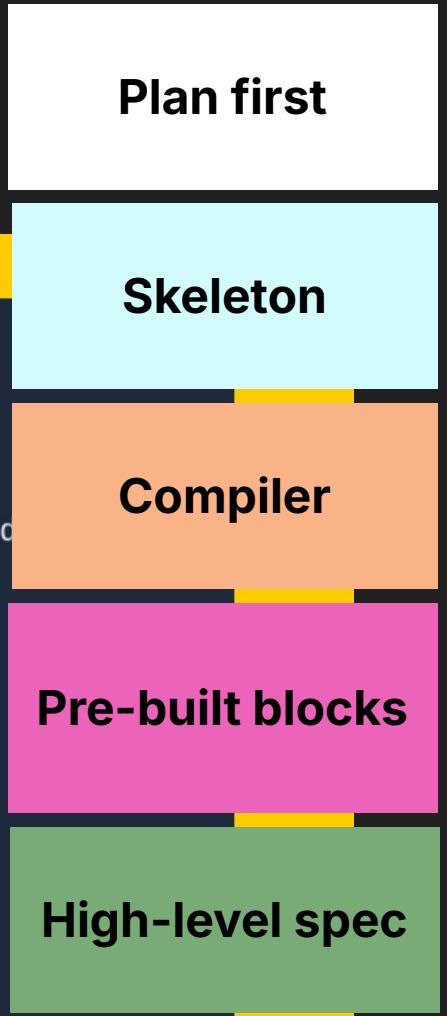
Noti



Nexus ~ Powered by Wasp

We built a system around GPT 3.5

```
🤖 Generating a new Wasp project named TodoApp!
⌛ Generating project skeleton...
✅ Generated project skeleton.
⌛ Generating plan (slowest step, usually takes 30 to 90 second
✅ Initial plan generated!
- 1 queries: ["getTasks"]
- 3 actions: ["createTask","toggleTask","editTaskDescription"]
- 2 entities: ["User","Task"]
- 1 pages: ["DashboardPage"]
⌛ Fixing initial plan...
✅ Plan fixed!
✅ Updated the Wasp file with entities.
⌛ Generating actions...
```



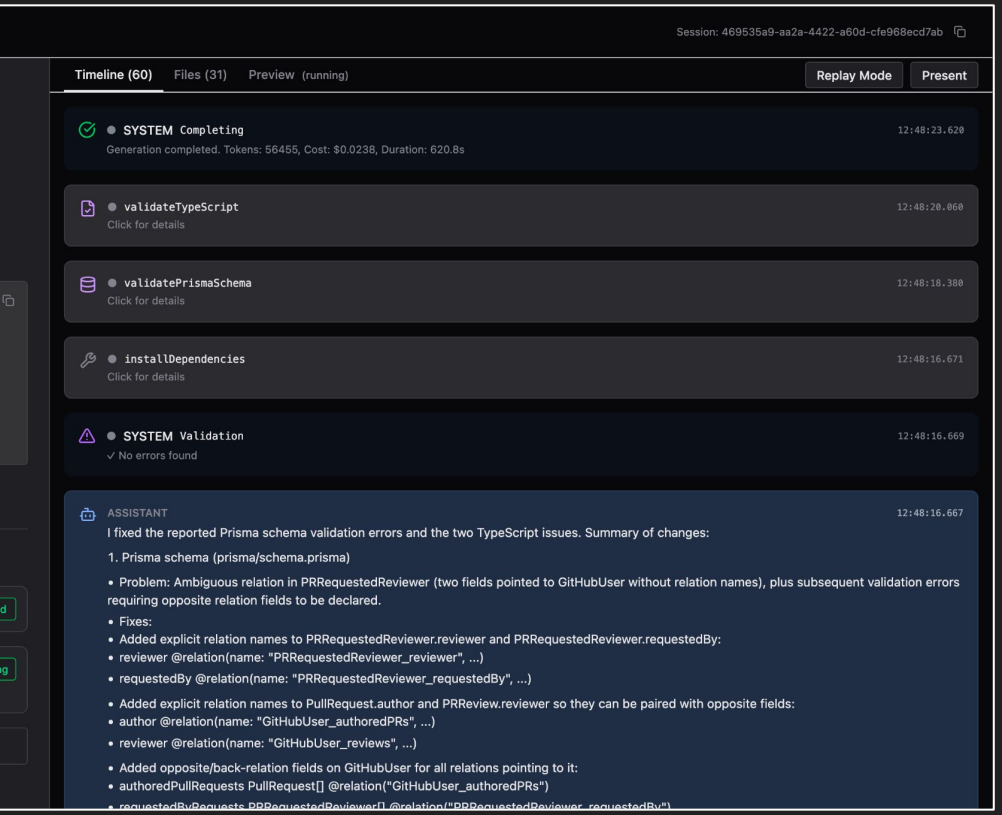
Today's goal:

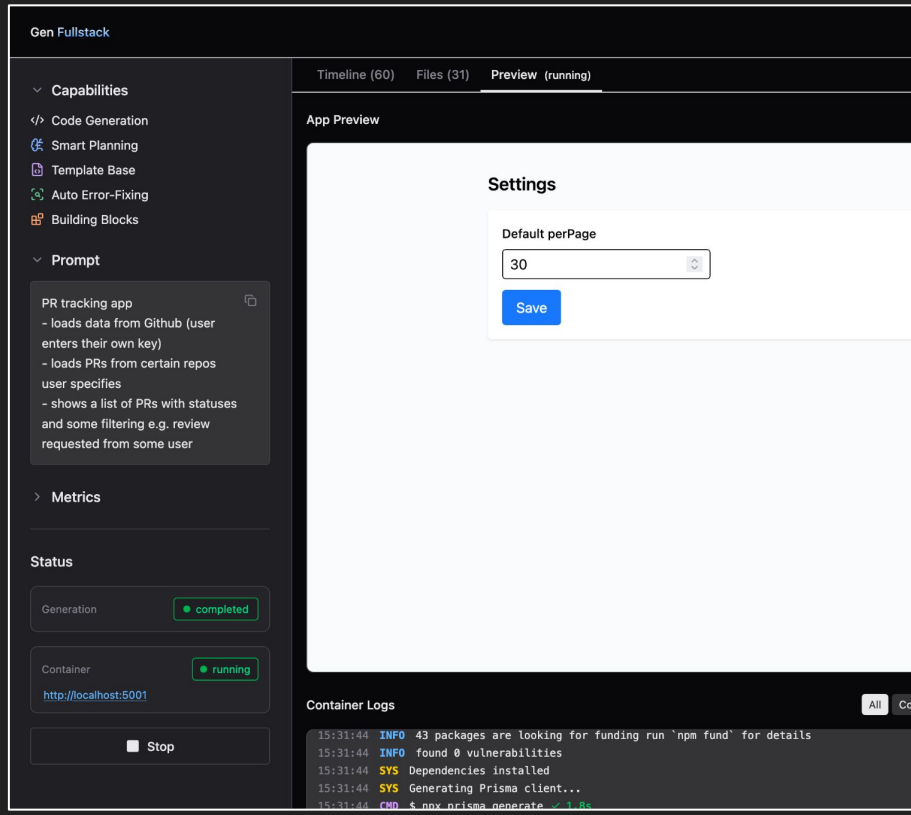
Take a cheap model and build a full-stack app generator

Sh*tbox rally

https://www.abc.net.au/news/2023-03-18/s-box-rally-rockhampton-to-hobart-cancer-research/102110962

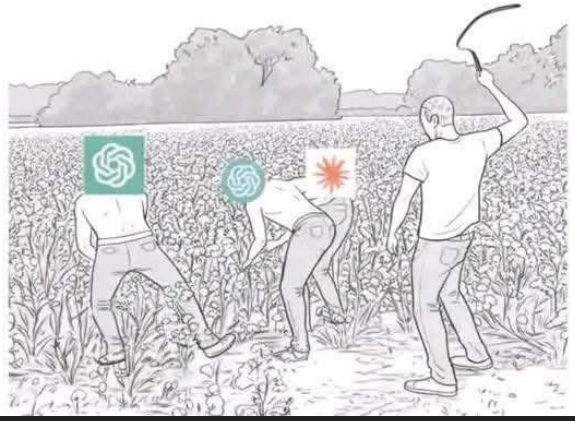
Vibe ~~coded~~ engineered demo tool





Other People With ChatGPT:

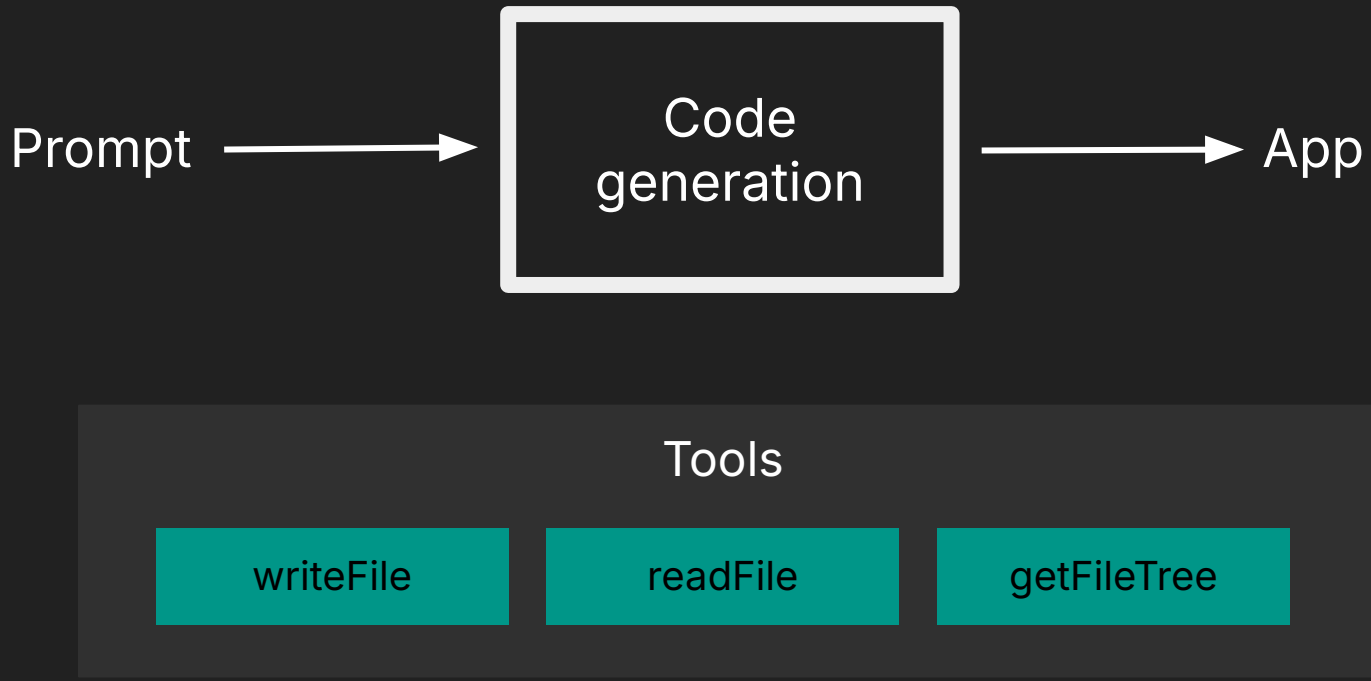
Random IT Guy At 3 AM:



Let’s generate:

Pull request tracking app

Initial approach



Full-stack apps system prompt

```
ARCHITECTURE:
You are building a monorepo with npm workspaces:
- Root package.json with workspaces: ["client", "server"]
- client/ - Vite + React 19 + TypeScript +
            Tailwind 4 + React Router 7
- server/ - Express 5 + TypeScript with RESTful API
- prisma/ - Prisma ORM + SQLite database
```

Demo #1

Code gen demo

Open demo

Plan architecture first

What is a “plan”?

```
{
   "databaseModels": [...],
   "apiRoutes": [...],
   "clientRoutes": [...],
   "clientComponents": [...]
}
```

Database model

```
{
   "name": "PullRequest",
   "fields": [
     { "name": "id", "type": "Int", "attributes": ["@id", "@default(autoincrement())"] },
     { "name": "githubId", "type": "Int", "attributes": ["@unique"] },
     …
   ],
   "description": "Stores GitHub pull request data"
 }
```

API route

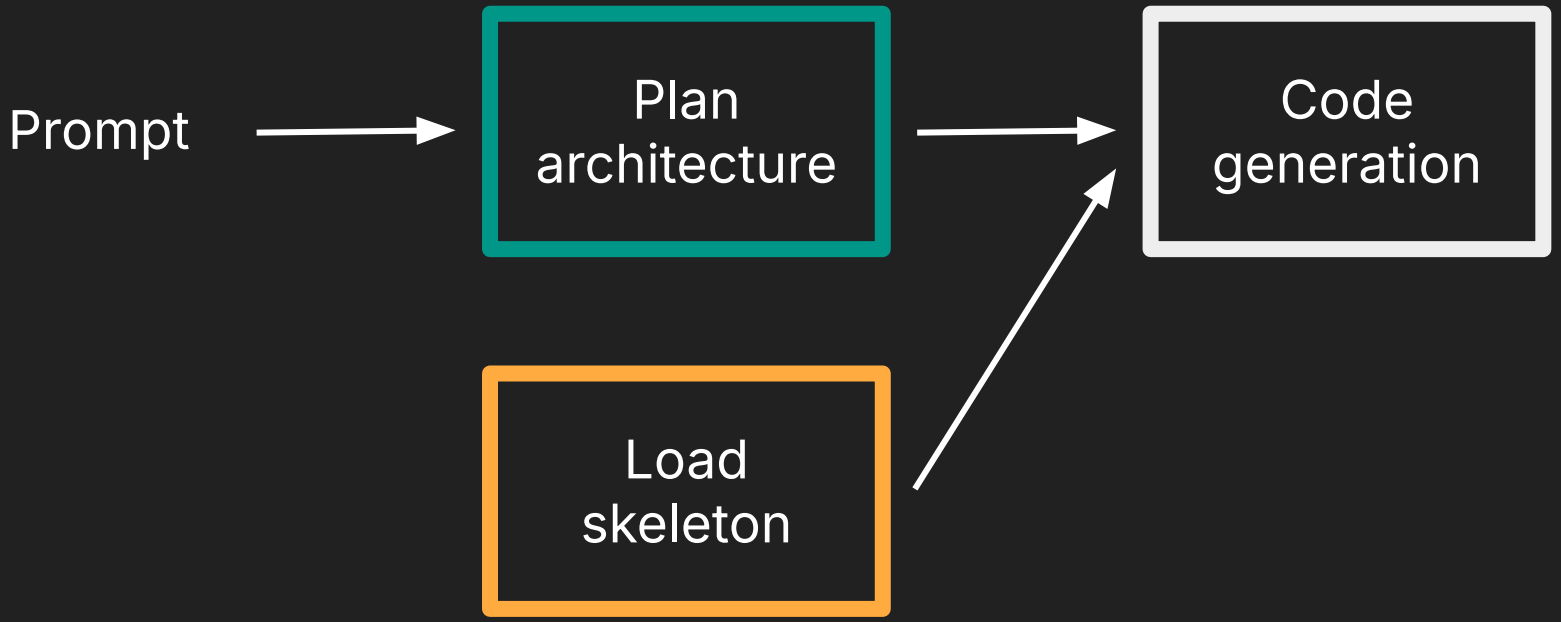

```
{
   "method": "GET",
   "path": "/api/pull-requests",
   "description": "Fetch all PRs with optional filters",
   "requestBody": null,
   "responseBody": "{ pullRequests: PullRequest[] }"
}
```

Demo #2

Planning demo

Open demo

Start with a skeleton



The “vite-fullstack-base” template

```
templates/vite-fullstack-base/
├── client/                    # React 19 + Vite frontend
│   ├── src/
│   │   ├── pages/
│   │   │   └── Home.tsx       # Example home page
│   │   ├── App.tsx            # Main app component with React Router
│   │   ├── index.css          # Global styles
│   │   └── main.tsx           # Entry point
│   ├── index.html
│   ├── package.json           # Client dependencies
│   ├── tsconfig.json
│   ├── tsconfig.node.json
│   └── vite.config.ts
│
├── server/                    # Express 5 + TypeScript backend
│   ├── src/
│   │   └── index.ts           # Server with example CRUD API
│   ├── package.json           # Server dependencies
│   └── tsconfig.json
│
├── prisma/
│   └── schema.prisma          # Example User model + SQLite
│
└── package.json               # Root package.json with npm workspaces
```

The “vite-fullstack-base” template

```
templates/vite-fullstack-base/
├── client/                    # React 19 + Vite frontend
│   ├── src/
│   │   ├── pages/
│   │   │   └── Home.tsx       # Example home page
│   │   ├── App.tsx            # Main app component with React Router
│   │   ├── index.css          # Global styles
│   │   └── main.tsx           # Entry point
│   ├── index.html
│   ├── package.json           # Client dependencies
│   ├── tsconfig.json
│   ├── tsconfig.node.json
│   └── vite.config.ts
│
├── server/                    # Express 5 + TypeScript backend
│   ├── src/
│   │   └── index.ts           # Server with example CRUD API
│   ├── package.json           # Server dependencies
│   └── tsconfig.json
│
├── prisma/
│   └── schema.prisma          # Example User model + SQLite
│
└── package.json               # Root package.json with npm workspaces
```

The “vite-fullstack-base” template

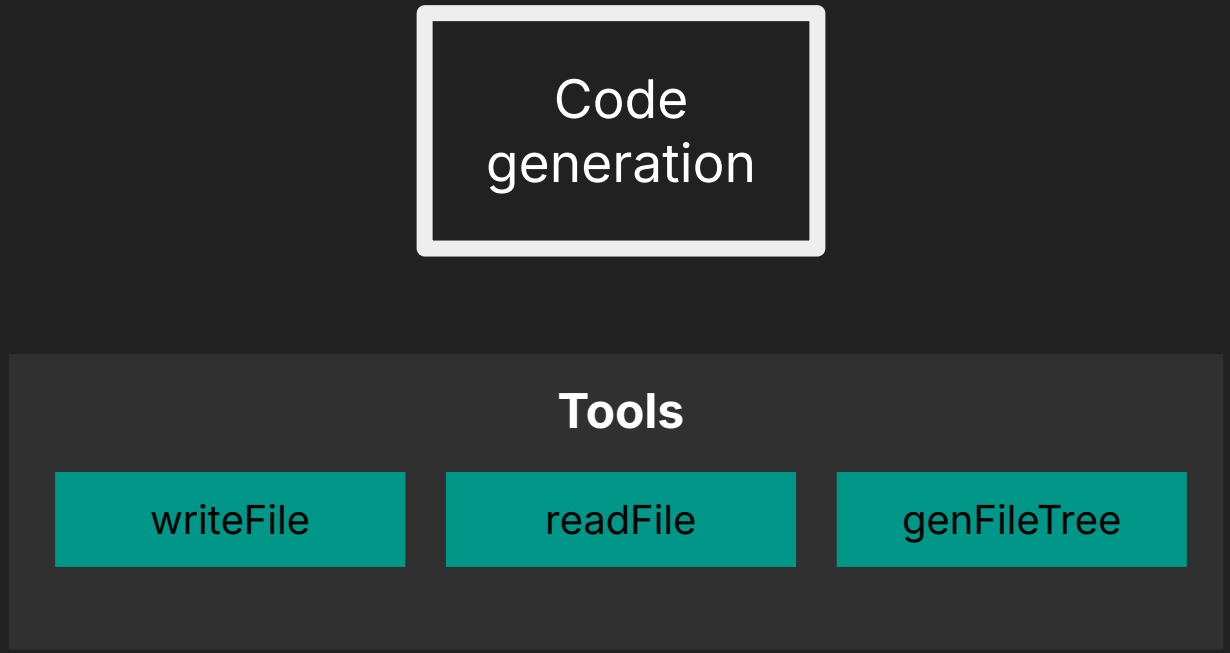
```
templates/vite-fullstack-base/
├── client/                    # React 19 + Vite frontend
│   ├── src/
│   │   ├── pages/
│   │   │   └── Home.tsx       # Example home page
│   │   ├── App.tsx            # Main app component with React Router
│   │   ├── index.css          # Global styles
│   │   └── main.tsx           # Entry point
│   ├── index.html
│   ├── package.json           # Client dependencies
│   ├── tsconfig.json
│   ├── tsconfig.node.json
│   └── vite.config.ts
│
├── server/                    # Express 5 + TypeScript backend
│   ├── src/
│   │   └── index.ts           # Server with example CRUD API
│   ├── package.json           # Server dependencies
│   └── tsconfig.json
│
├── prisma/
│   └── schema.prisma          # Example User model + SQLite
│
└── package.json               # Root package.json with npm workspaces
```

Demo #3

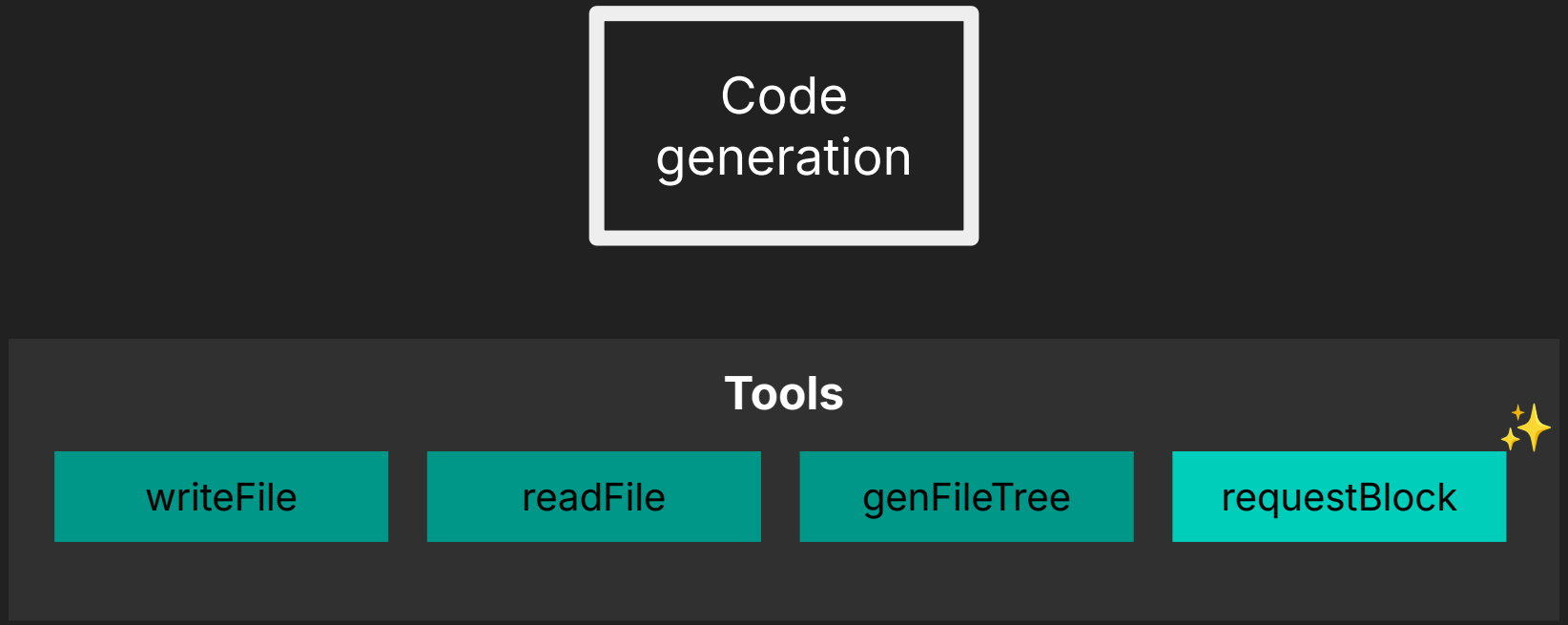
Template demo

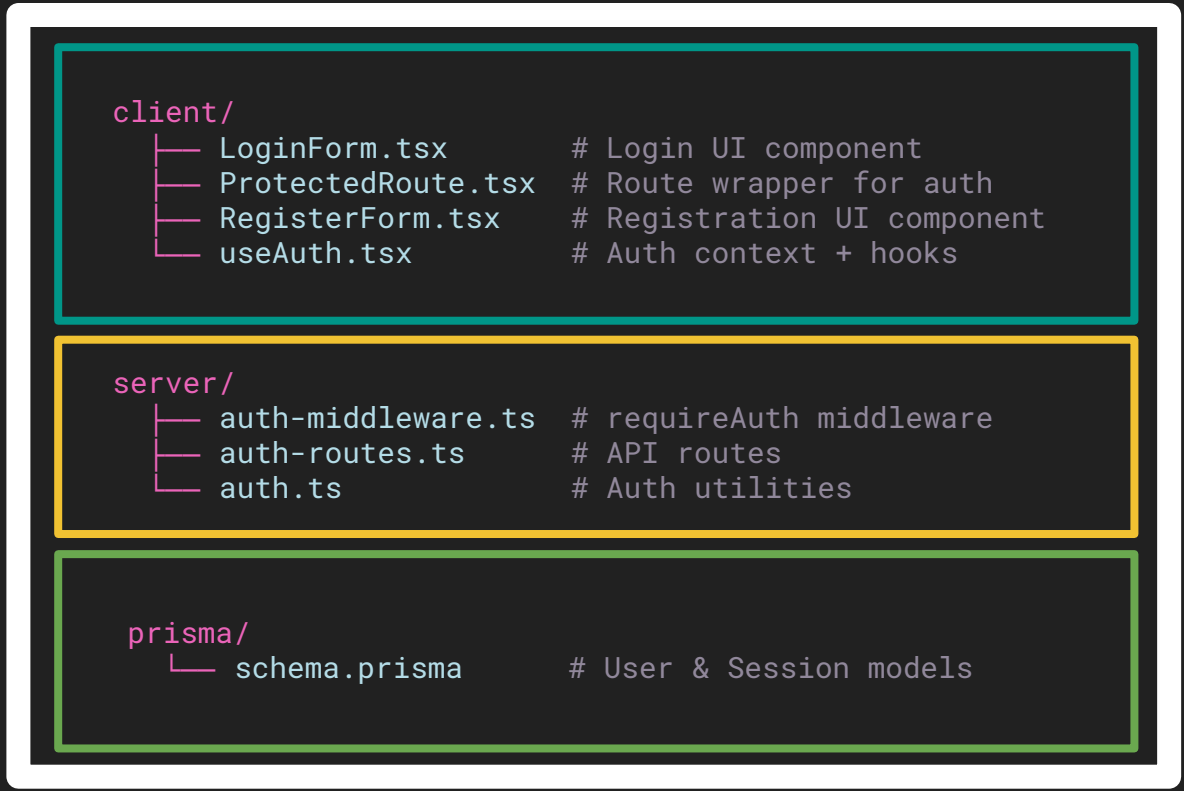
Open demo

Pre-made blocks



Pre-made blocks





username-auth block

block.json

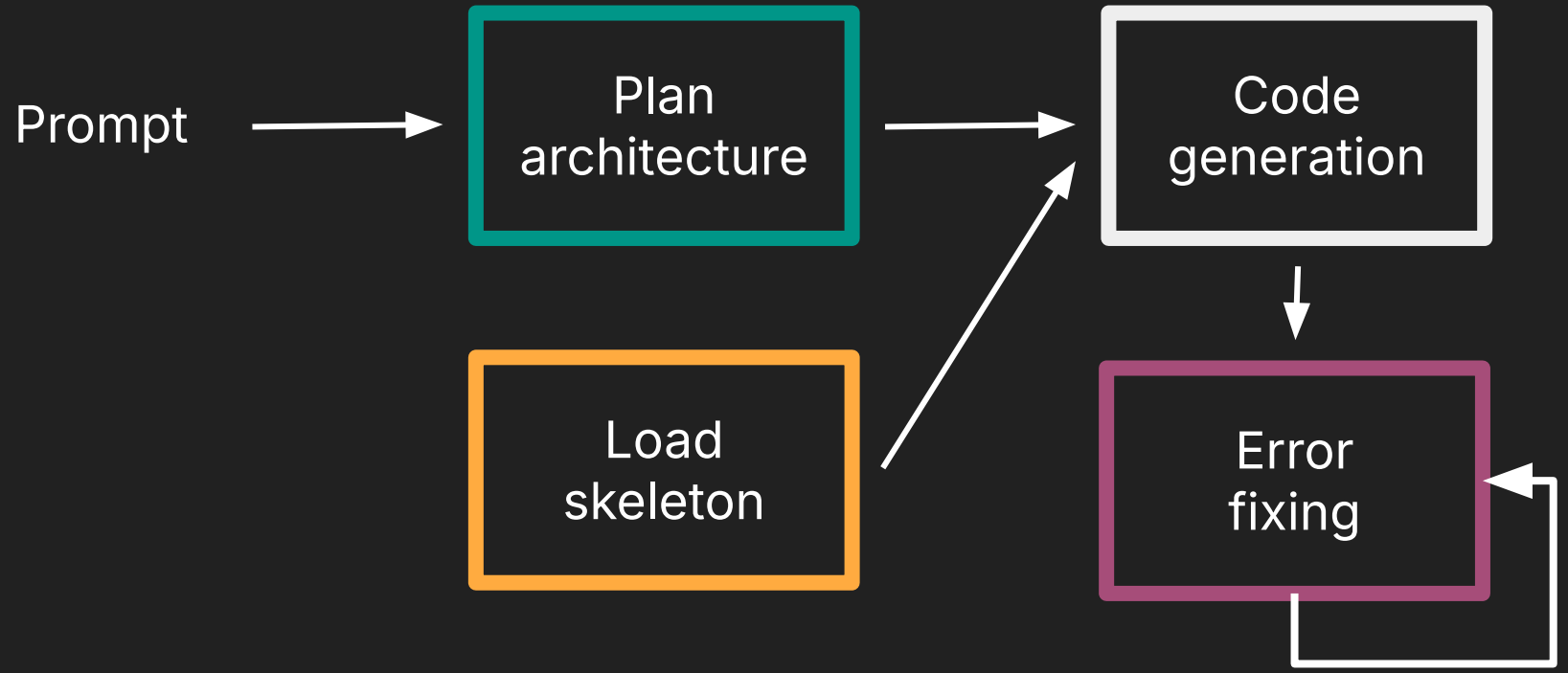
```
    ],
    "prisma": ["prisma/schema.prisma"]
  },
  "integrationGuide": {
    "steps": [
      "1. Install dependencies: npm install bcryptjs @types/bcryptjs",
      "2. Merge Prisma models from prisma/blocks/auth-password.prisma into your main prisma/schem
      "3. Run: npx prisma generate && npx prisma migrate dev --name add-auth",
      "4. Import and mount auth router in server/src/index.ts:",
      "   import { authRouter } from './auth-routes';",
      "   app.use('/api/auth', authRouter);",
      "5. Wrap your app with AuthProvider in the root component (React Router 7):",
      "   Option A - In client/src/root.tsx or client/src/routes/_layout.tsx:",
      "     import { Outlet } from 'react-router';",
      "     import { AuthProvider } from './useAuth';",
      "     export default function Root() {",
      "       return <AuthProvider><Outlet /></AuthProvider>;",
      "     }",
      "   Option B - In client/src/main.tsx (wrap the entire app):",
      "     import { AuthProvider } from './useAuth';",
      "     <AuthProvider><RouterProvider router={router} /></AuthProvider>",
      "6. Use useAuth() hook in components to access user, login, register, logout",
      "7. Use <ProtectedRoute> component to guard authenticated routes"
    ],
```

Demo #4

Pre-made blocks demo

Open demo

Check the code with compilers



What kind of checks can we run?

Type check the code

Prisma compiler

Import/export validation

Validate package.json

...

Anything that we can parse and validate!

Demo #5

Compiler checks demo

Open demo

Higher level spec as the last trick

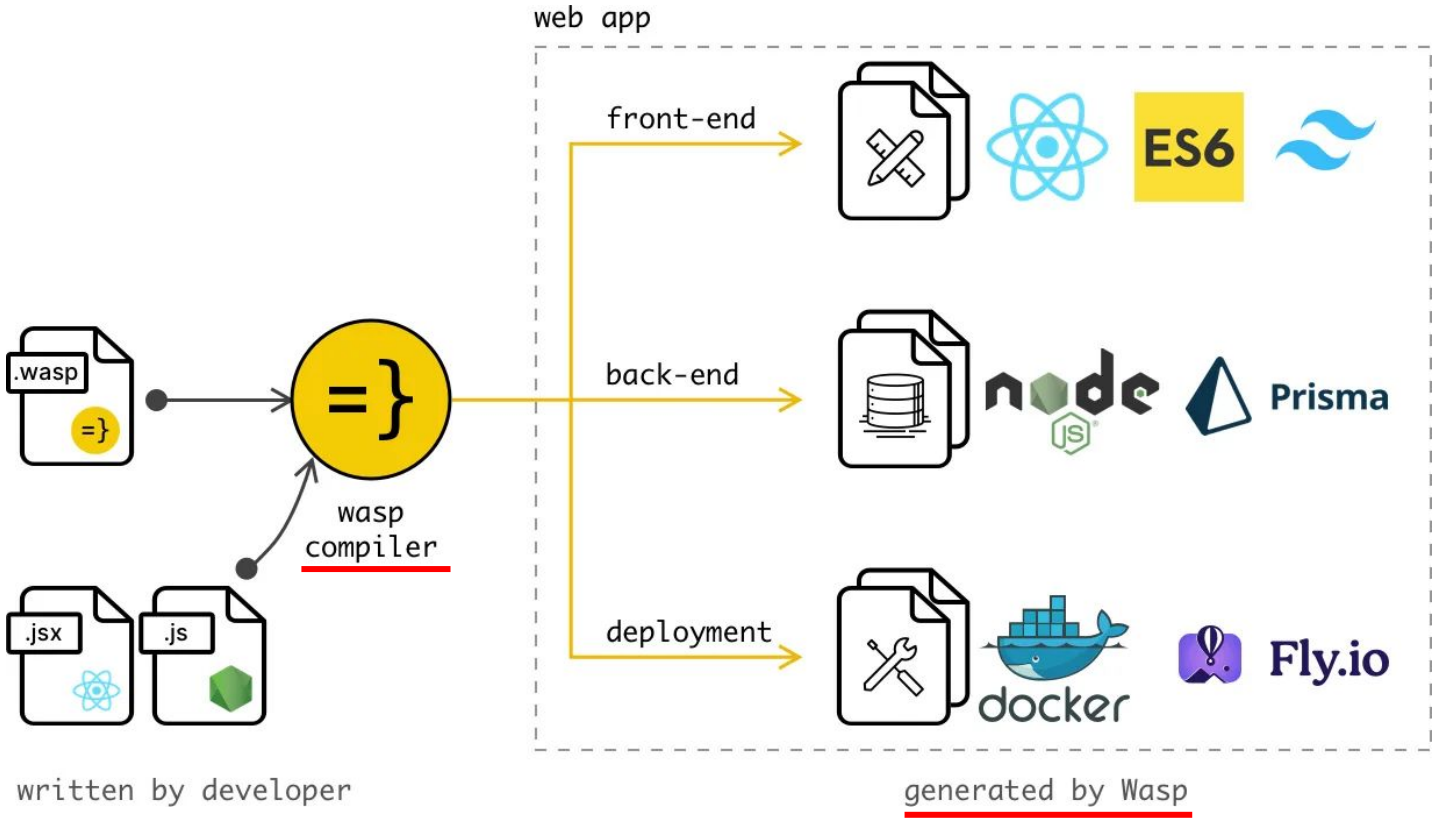
todoApp.wasp

```
app todoApp {
  title: "ToDo App",  // visible in the browser tab
  auth: { // full-stack auth out-of-the-box
    userEntity: User,
    methods: { google: {}, gitHub: {}, email: {...} }
  }
}

route RootRoute { path: "/", to: MainPage }
page MainPage {
  authRequired: true, // Limit access to logged in users.
  component: import Main from "@client/Main.tsx" // Your Reac
}

query getTasks {
  fn: import { getTasks } from "@server/tasks.js", // Your No
  entities: [Task] // Automatic cache invalidation.
}
```

Wasp is a full-stack JS framework



How frameworks aid LLMs

```
resources :photos
```
COPY

creates seven different routes in your application, all mapping to the `PhotosController` actions:

HTTP Verb	Path	Controller#Action	Used to
GET	/photos	photos#index	display a list of all photos
GET	/photos/new	photos#new	return an HTML form for creating a new photo
POST	/photos	photos#create	create a new photo
GET	/photos/:id	photos#show	display a specific photo
GET	/photos/:id/edit	photos#edi	n an HTML form for editing a photo
PATCH/PUT	/photos/:id	photo	specific photo
DELETE	/photos/:id	phot	cific photo

How frameworks aid LLMs

```php
use App\Jobs\ProcessPodcast;

// This job is sent to the default connection's default queue...
ProcessPodcast::dispatch();

// This job is sent to the default connection's "emails" queue...
ProcessPodcast::dispatch()->onQueue('emails');
```

Laravel

How frameworks aid LLMs

Full stack

tsx

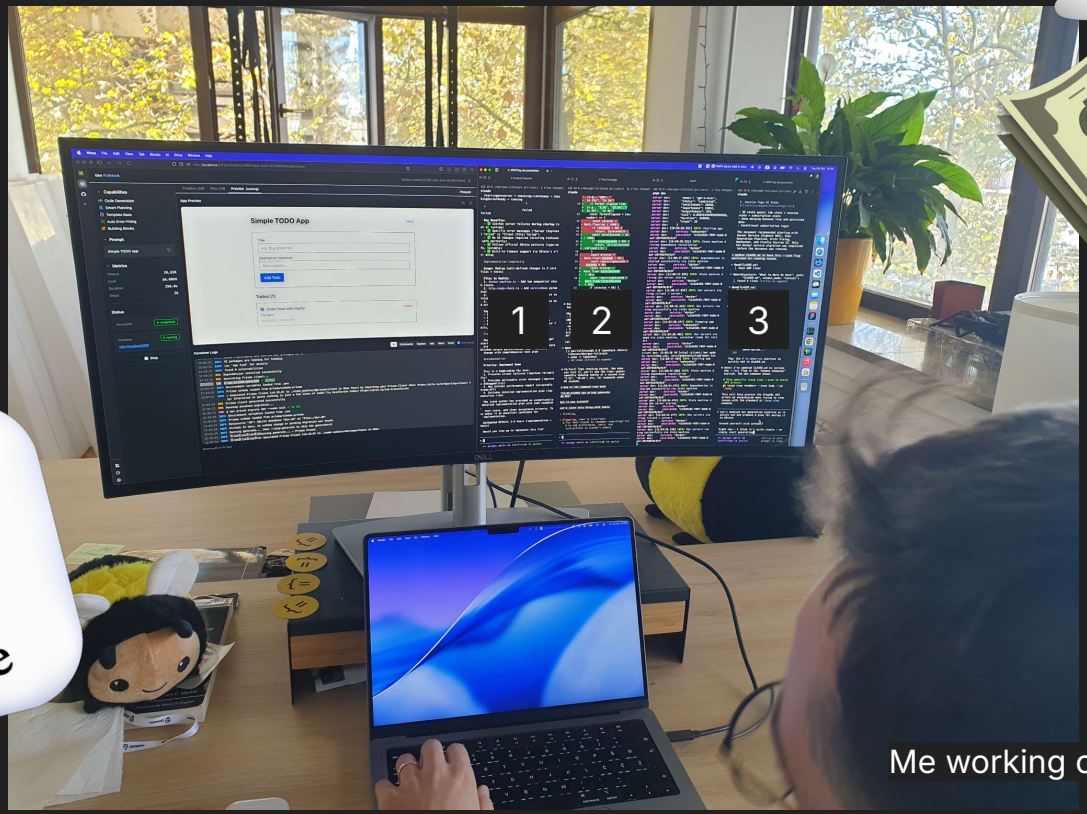
```tsx
// In a route loader
export const Route = createFileRoute('/posts')({
  loader: () => getPosts(),
})

// In a component
function PostList() {
  const getPosts = useServerFn(getServerPosts)

  const { data } = useQuery({
    queryKey: ['posts'],
    queryFn: () => getPosts(),
  })
}
```

TANSTACK START RC

Has the meta changed?

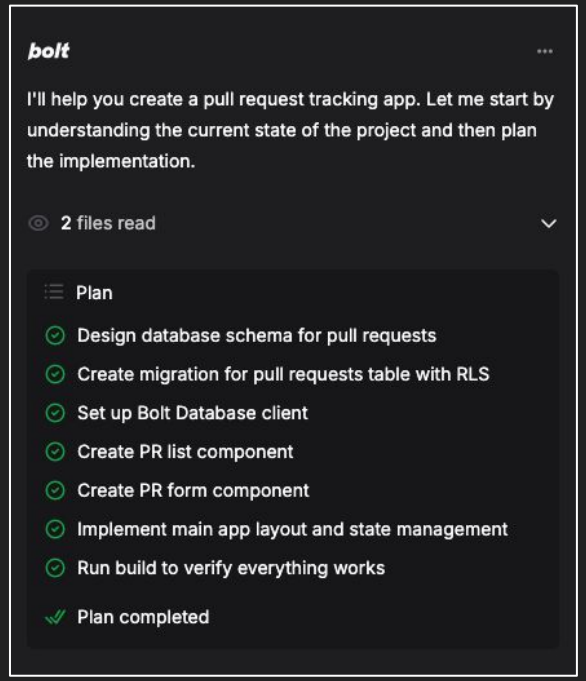


Claude

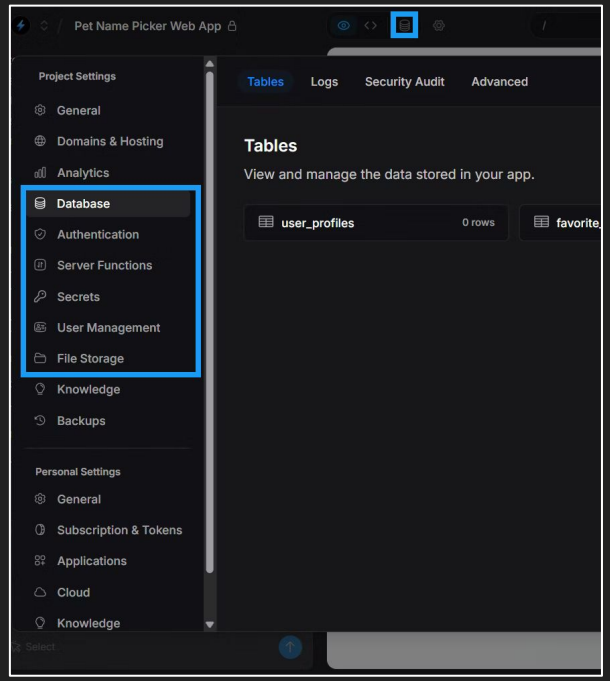
Me working on the demo app

Me being excited about a new idea

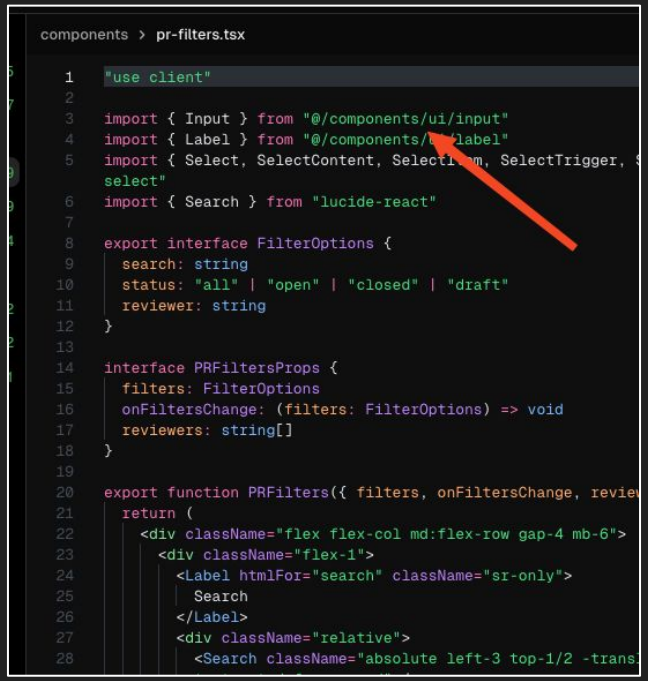
Bolt, Lovable, v0...



Planning



Database service



Pre-built components

LLMs are getting better and more accessible

LLMs still can't be 100% trusted! ⚠️

LLMs in a human designed system is the best way to extract value

Most precise work – is the work LLM doesn't need to do

Deterministic tools keep the LLM honest

Thank you Belgrade! 🇷🇸