[ایجاد پروژه 1](#_Toc158285617)

[پیکربندی Sanity و ساخت Schema 3](#_Toc158285618)

[طراحی HomePage 6](#_Toc158285619)

[طراحی HeroBanner 8](#_Toc158285620)

[اتصال به Sanity و واکشی اطلاعات 8](#_Toc158285621)

[طراحی Button 10](#_Toc158285622)

[اضافه کردن دکمه ثبت نام به Menu 11](#_Toc158285623)

[منابع 12](#_Toc158285624)

# Sanity چیست

نحوه مدیریت محتوای یک وب سایت از اهمیت بالایی برخوردار است؛ زیرا نگهداری و بروزرسانی یک سایت به شکل نادرست، می­تواند هزینه­های قابل توجهی به همراه داشته باشد. برای رفع این مشکل، می‌توانید وبسایت خود را با یک Headless CMS که یک API برای مدیریت محتوا و بروزرسانی ارائه می‌دهد، ادغام کنید. ما برای این منظور از «Sanity» استفاده خواهیم کرد.

**Sanity** یک فریمورک Headless CMS است که ابزارهای لازم برای استفاده از APIها برای اتصال به وب اپلیکیشن و مدیریت محتوا در فضای ابری را فراهم می کند. با استفاده از Sanity، می‌توانید صفحات یا محتوایی که نیاز به بروزرسانی منظم دارند را به Studio متصل کنید و بدون نیاز به تغییر مداوم کدها، آنها را از طریق content lake مدیریت کنید. استفاده از Sanity موجب می­شود فرآیند تولید و مدیریت محتوا برای تمامی افراد، بدون دانش قبلی، امکان­پذیر باشد.

در ادامه قصد داریم یک سایت Portfolio را به کمک Sanity و Next.js 13 پیاده­سازی کنیم.

# نصب Next.js

برای نصب آخرین نسخه Next.js، terminal را اجرا کرده و دستور زیر را وارد کنید :

**npx create-next-app@latest**

به جز نام پروژه، تمامی گزینه­های پیش فرض را انتخاب کنید :

**What is your project named?... next-sanity**

**Would you like to use TypeScript?... Yes**

**Would you like to use ESLint?... Yes**

**Would you like to use Tailwind CSS?... Yes**

**Would you like to use `src/` directory?... No**

**Would you like to use App Router?(recommended)... Yes**

**Would you like to customize the default import alias?... No**

اکنون برای مشاهده خروجی، می­توانید از دستور زیر را استفاده کنید :

**cd next-sanity**

**npm run dev**

# راه اندازی Sanity Studio

sanity studio یک برنامه تک صفحه open source، برای مدیریت داده­ها و انجام عملیات مختلف است. شما از طریق این رابط، می­توانید داده­های خود را در sanity ایجاد، حذف و یا به روز کنید.

برای نصب sanity studio یک ترمینال جدید خارج از برنامه Next.js باز کرده و دستورات زیر را وارد کنید :

**mkdir sanity-studio**

**cd sanity-studio**

**npm create sanity@latest**

پس از وارد کردن دستور فوق؛ از شما خواسته می­شود که از بین گزینه­های موجود، یک روش را برای ورود به حساب کاربری انتخاب کنید. اگر از قبل حسابی در Sanity ایجاد نکرده­اید، می­توانید یک حساب جدید ایجاد کنید. پس از احراز هویت، تعدادی prompt برای پیکربندی پروژه نمایش داده می­شود :

**Project name: Next Sanity**

**Use the default dataset configuration? Yes**

**Project output path: C:\Users\Admin\Desktop\sanity-studio**

**Select project template: Clean project with no predefined schemas**

**Do you want to use TypeScript? Yes**

**Would you like to add the project ID and dataset to your .nv file? Yes**

**Package manager to use for installing dependencies? npm**

پس از تکمیل دستور، Sanity Studio به صورت محلی نصب می­شود. برای مشاهده آن دستور زیر را وارد کرده و خروجی را در localhost:3333 مشاهده کنید :

**npm run dev**

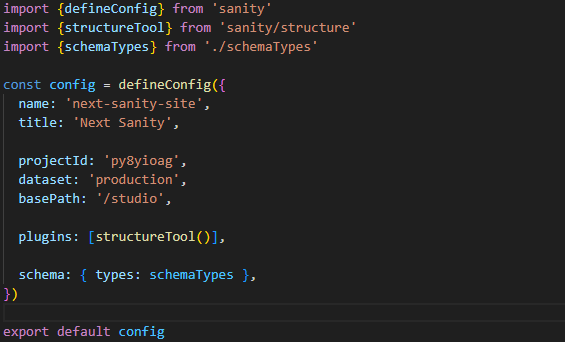
# افزودن Studio به Next.js

شما می­توانید Studio را در هاست جداگانه قرار دهید، اما در این آموزش با استفاده از next-sanity، آن را در برنامه Next.js خود نصب می­کنیم. با استفاده از دستور زیر **next-sanity** را نصب کنید :

**npm install sanity next-sanity**

سپس پوشه **schemaTypes** و فایل **sanity.config.ts** را کپی کرده و در ریشه برنامه Next.js خود قرار دهید.

در داخل فایل sanity.config.ts، یک فیلد به نام **'basePath'** با مقدار **'/studio'** یا هر URL معتبری که می‌خواهید Studio شما در آن اجرا شود، تعریف کنید.



در ادامه کاربرد هر یک از خصوصیت­ها ذکر شده است :

**: name** برای تفکیک فضاهای کاری استفاده می­شود. برای راه اندازی یک فضای کاری اجباری نیست.

**: title** عنوان پروژه که در Studio نشان داده می شود.

**: projectId** یک شناسه منحصر به فرد است که به پروژه Sanity جاری، اشاره دارد.

**: dataset** نام dataset که برای Studio استفاده می­کنید؛ نام­های رایج عبارتند از: 'production' و 'development' است.

**: basePath** مسیری که Studio در آن نصب می­شود.

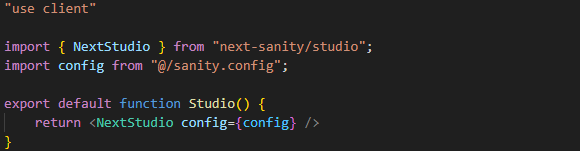
**: schema** شئ که فایل­های schema در آن تعریف می­شوند.

پس از انجام تغییرات فوق؛ باید یک کامپوننت برای Studio تعریف کنید. بنابراین، یک فایل به نام **page.tsx** در مسر زیر ایجاد کنید :

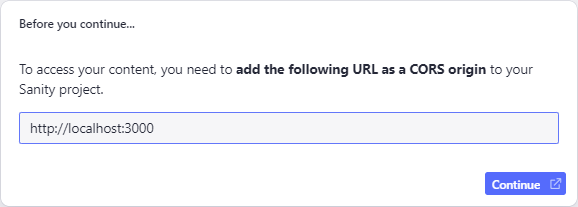
**app/studio/[[…index]]**

با این کار، در هر مسیری که با '/studio' مطابقت داشته باشد، کامپوننت page.tsx استودیو رندر می­شود.

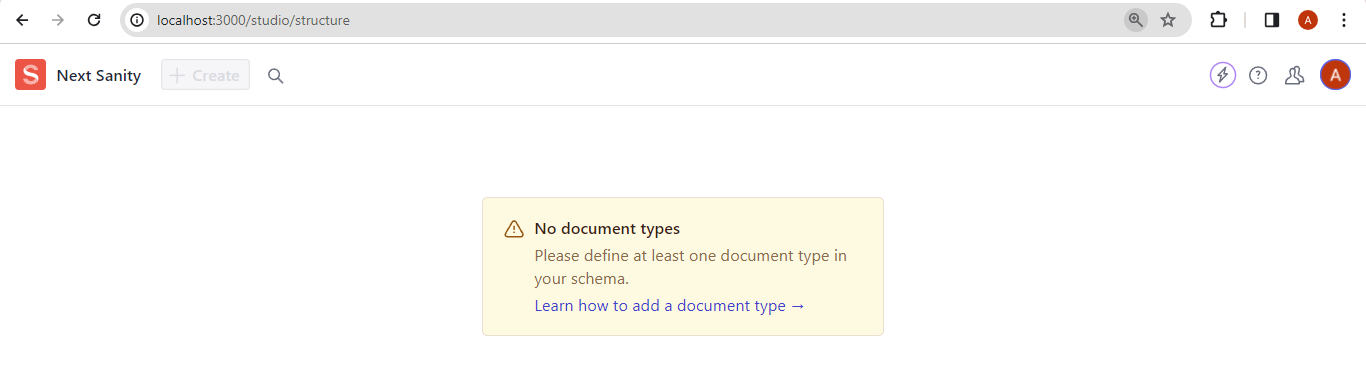
اکنون کدهای زیر را در این کامپوننت قرار دهید :



اکنون دستور **npm run dev** را اجرا کرده و به localhost:3000/studio مراجعه کنید. شما یک پیغام برای اضافه کردن localhost:3000 به عنوان CORS origin به پروژه Sanity خود دریافت خواهید کرد. با کلیک بر روی 'continue' URL را اضافه کنید.



سپس با همان روشی که در ایجاد حساب کاربری خود استفاده کردید، وارد حساب Sanity خود شوید و همانطور که در تصویر زیر نشان داده شده است، Studio در برنامه Next.js نصب شده است :



با توجه به اینکه اکنون Studio در برنامه Next.js اجرا می­شود، لذا به پوشه sanity-studio نیاز ندارید و می­توانید آن را حذف کنید یا ببندید.

# تعریف Schema

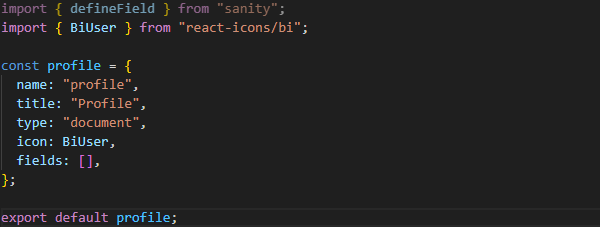
به‌طور پیش‌فرض، محیط Studio خالی می­باشد و باید فایل­های schema به آن اضافه شود. یک **الگو (schema)** اساساً راهی برای سازماندهی مجموعه داده­ها در یک پایگاه داده با توجه به نوع محتوای مورد نیاز شما است.

از آنجایی که ما در حال ساختن یک سایت نمونه کار (Portfolio) هستیم، الگوهای زیر را برای پروژه ایجاد خواهید کرد :

* profile
* project
* work

## الگوی پروفایل (Profile Schema)

الگوی پروفایل، برای تعریف اطلاعات شخصی شما مانند نام و نام خانوادگی، درباره من، مهارت­ها و ... است. یک فایل به نام **profile.ts** به پوشه 'schemas' اضافه و کدهای زیر را در آن وارد کنید :



ابتدا با خصوصیات پایه یک فایل schema شروع می­کنیم. هر فایل باید دارای خصوصیت­های name، title و type باشد. در ادامه کاربرد هر کدام را به طور مختصر توضیح می­دهیم :

**: name** خصوصیتی است که برای ارجاع به الگو در زبان پرس­وجو استفاده می­شود و مقدار آن باید منحصربه‌ فرد باشد تا الگوها با هم ترکیب نشوند.

**: title** نام الگو را در رابط کاربری Studio مشخص می­کند.

**: type** نوع الگو ای را مشخص می­کند. مقدار 'document' امکان ساخت اسناد جدید در Studio را فراهم می­کند.

**: icon** یک خصوصیت اختیاری است که می­توانید در کنار عنوان اضافه کنید. برای استفاده از icon، کتابخانه react-icons را با دستور **npm install -D react-icons** نصب کنید.

**: fields** آرایه­ای است که فیلدهای ورودی در آن تعریف می­شوند.

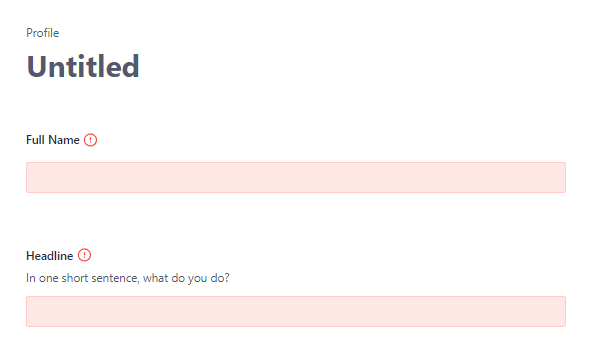
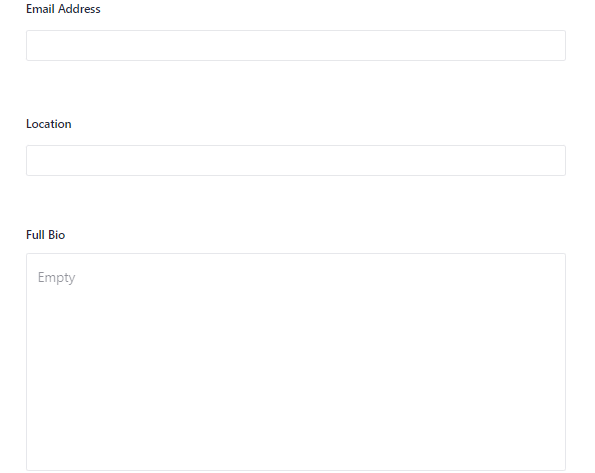
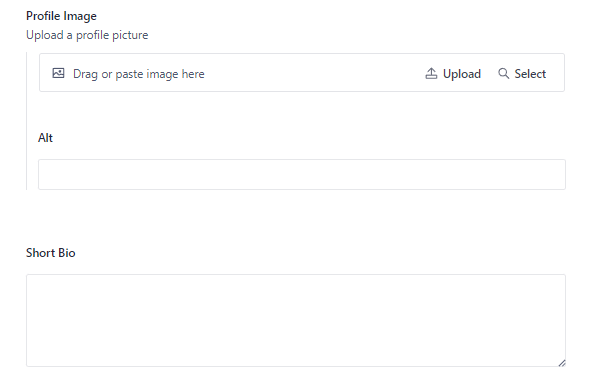
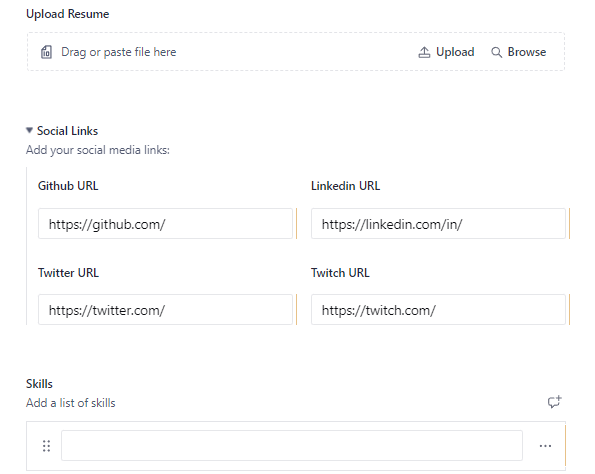
برای درک نحوه کار فیلدها، هر فیلد را مانند یک ورودی فرم در HTML در نظر بگیرید، این فیلدها در Studio در دسترس خواهند بود. مقدار هر ورودی به یک شی JSON صادر می­شود که می­توانید از آن برای تزریق داده­های خود استفاده کنید. می­توانید هر تعداد فیلد اضافه کنید، اما هر کدام باید دارای یک name، title و type باشد.

Sanity از انواع داده­ها برای تعریف یک الگو پشتیبانی می­کند که می­توانید لیست کامل آنها را در این [لینک](https://www.sanity.io/docs/schema-types) مشاهده کنید.

برای نمایش این الگو در Studio، باید آن را به آرایه schemas در فایل schemas/index.ts اضافه کنید:



اکنون می­توانید از این الگو برای ایجاد محتوا در Studio (localhost:3000/studio) استفاده کنید :

پس از تکمیل فیلدها، بر روی گزینه **"publish"** کلیک کنید.

پروژه: فایل الگو برای پروژه های شما.

work: فایل الگو ای برای تعریف تجربه کاری شما.

بیایید با طرح نمایه شروع کنیم.

**گام دوم : افزودن Sanity به پروژه**

**Sanity** یک پلتفرم واحد برای مدیریت محتوای ساختاریافته است. Sanity به ما این امکان را می­دهد که بدون نگرانی در مورد محتوا، ذخیره سازی فایل­ها و پایگاه داده، روی توسعه برنامه تمرکز کنیم. در این پروژه، جزئیات تمام محصولات موجود در فروشگاه و نیز بنرها را به کمک Sanity مدیریت می­کنیم.

برای پیکربندی Sanity، ابتدا باید یک حساب در [Sanity](https://www.sanity.io/) ایجاد نمایید، برای این منظور می­توانید از حساب Google خود استفاده نمایید. پس از ورود به حساب کاربری، دستور زیر را در Terminal وارد کنید :

**npm create sanity@latest**

پس از وارد کردن دستور، promptهای زیر را وارد کنید :

**login type Google**

**Use the default dataset configuration? yes**

**Would you like to add configuration files for a sanity project in this Next.js folder? Yes**

**Do you want to use TypeScript? No**

**Would you like to an embedded Sanity Studio? Yes**

**Would you like to use the Next.js app directory for routes? Yes**

**What route do you want to use for the Studio? /studio**

**Select project template to use Clean project with no predefined schemas**

**Project output path: C:\Users\Admin\Project\ecommerce\ecommerce\sanity**

**Would you like to add the project ID and dataset to your .nv file? Yes**

**Package manager to use for installing dependencies? npm**

پس از اجرای موفقیت آمیز دستور فوق؛ می­توانید خروجی Sanity را با استفاده از دو دستور زیر مشاهده کنید :

**cd C:\Users\Admin\Project\ecommerce\ecommerce\sanity**

**npm run dev**

منبع :

<https://youtu.be/bDVAQZVeebw?si=aaoXs83SBzHXv7Gp>

<https://www.sanity.io/docs/create-a-sanity-project>

**گام سوم : اضافه کردن پکیج­ها به پروژه**

برای نصب پکیج­های مورد نیاز پروژه، دستورات زیر را در Terminal وارد کنید :

**npm i @sanity/client**

**npm i --save @sanity/image-url**

**npm i @stripe/stripe-js**

**npm i canvas-confetti**

**npm i next-sanity-image**

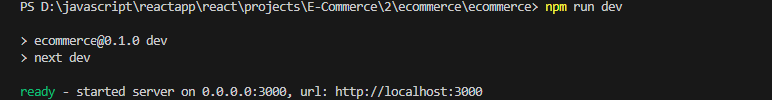
**npm i react-hot-toast**

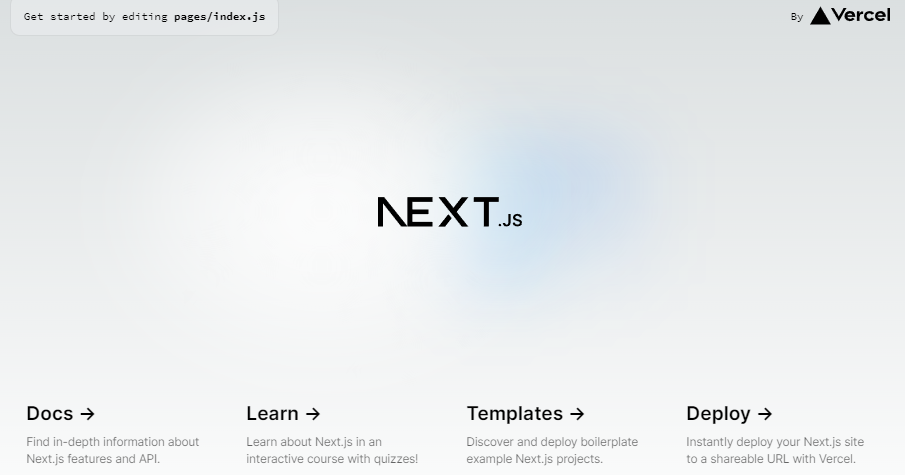
**npm i react-icons**

**npm i stripe**

اکنون با استفاده از دستور زیر می­توانید خروجی پروژه را مشاهده نمایید :

**npm run dev**





# پیکربندی Sanity و ساخت Schema

در بخش قبلی، Sanity را نصب کردیم. اکنون می­خواهیم قابلیت­های این پلتفرم را بررسی نماییم. ابتدا با استفاده از دستور زیر، Sanity/Cli را نصب کنید :

**npm install -g @sanity/cli**

دستور زیر مستندات Sanity را در مرورگر باز کرده و یک راهنمای کاربردی را در اختیار شما می­گذارد:

**sanity docs**

دستور زیر شما را به صفحه **'sanity manager'** هدایت می‌کند :

**sanity manage**

برای دسترسی به محتوا (Schema) وارد پوشه **'\sanity'** می­شویم و سپس دستور زیر را اجرا می­کنیم :

اگر دستور sanity اجرا نشد :

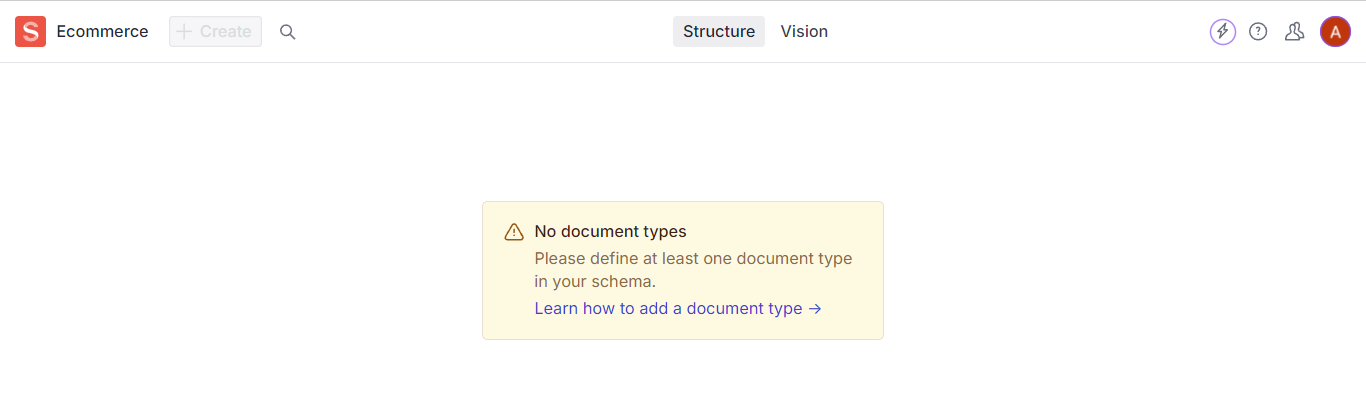
<https://www.makeuseof.com/windows-powershell-cannot-loaded-running-scripts-disabled>

**sanity start**

یا

**npm run dev**

با اجرای دستور فوق؛ میزکار (**Sanity desk)** در **localhost:3333** نمایش داده می­شود.



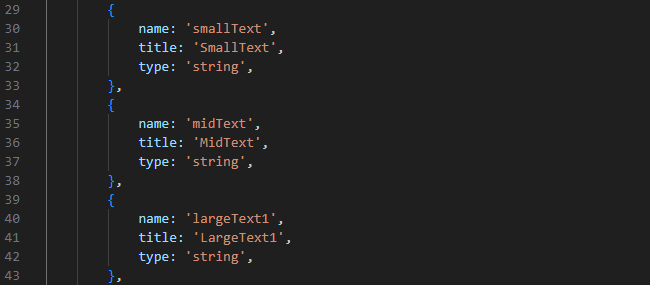
اکنون باید Schema مورد نظر خود را تعریف کنیم. برای این منظور؛ در پوشه **'\sanity\schemas'** یک فایل به نام **product.js** ایجاد کرده و کدهای زیر را به آن اضافه می­کنیم :



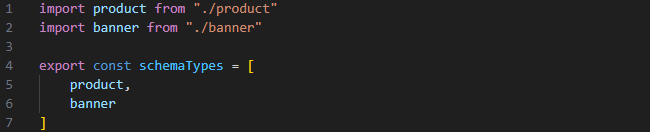


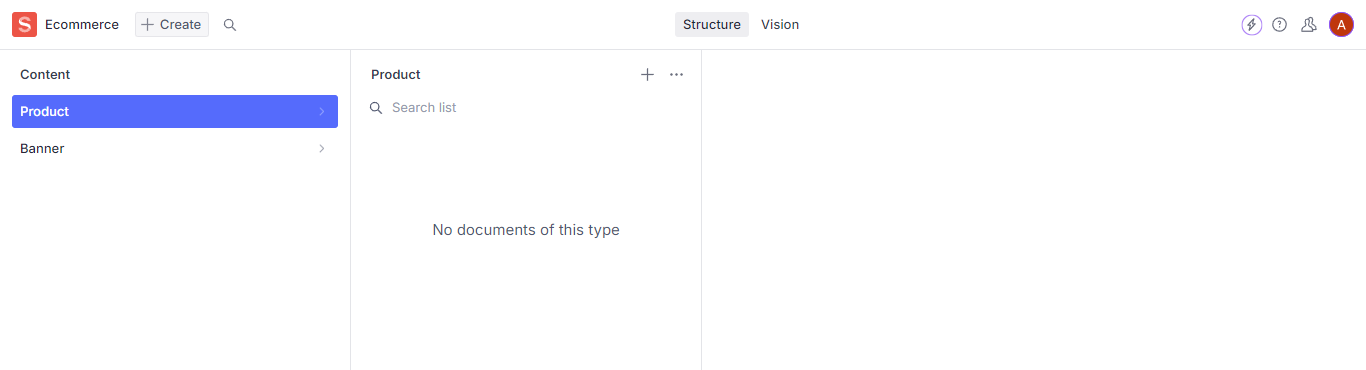
فایل **banner.js** را هم به پوشه **'schemas'** اضافه کرده و کدهای زیر را در آن وارد می­کنیم :



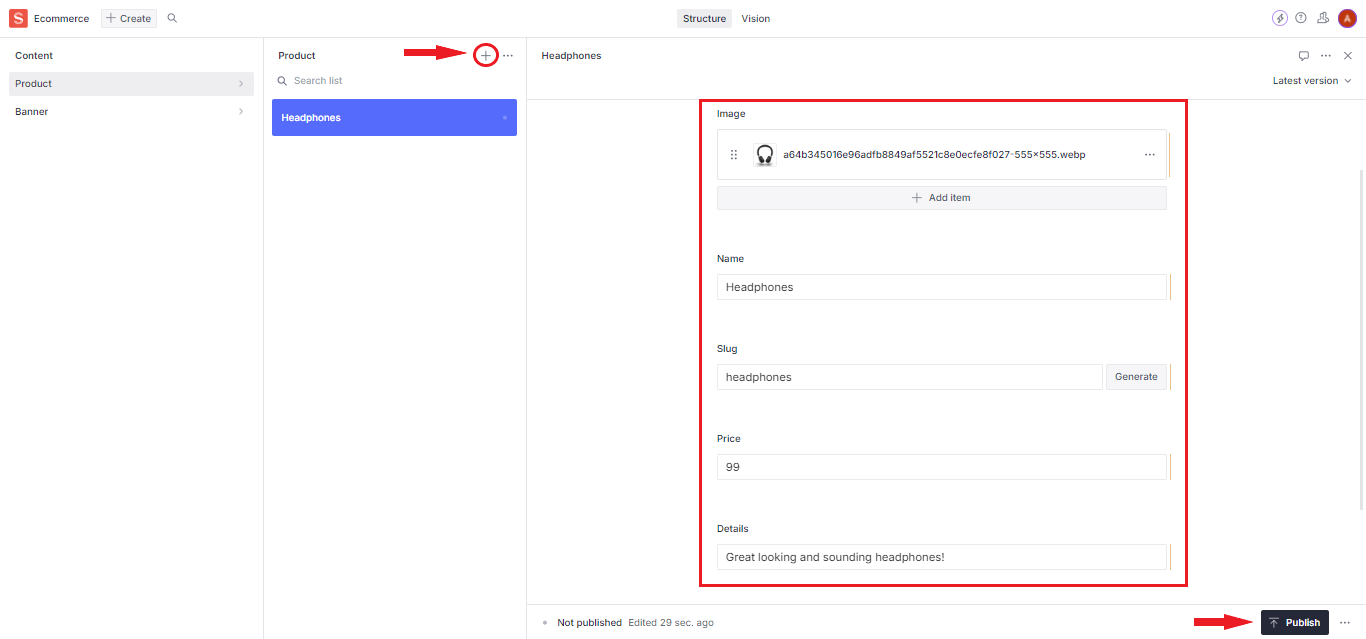


اکنون تنها کاری که باید انجام دهیم، این است که schemaهای تعریف شده را به فایل **index.js** اضافه کنیم:





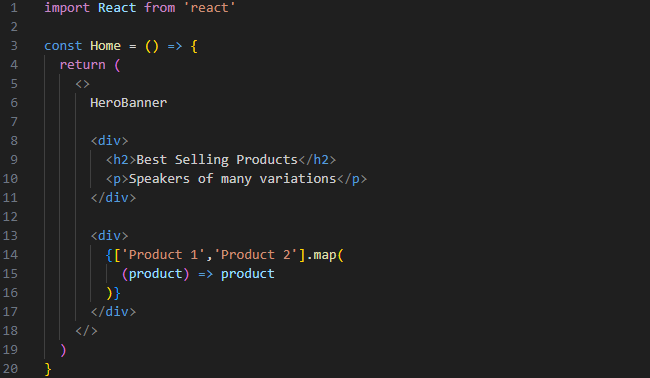
پس از تعریف schema، می­توانیم با استفاده از رابط کاربری Sanity desk نمونه­هایی از آن ایجاد کنیم :



همانطور که مشاهده می­کنید؛ به راحتی توانستیم با تعریف یک schema، یک پایگاه داده با قابلیت افزودن محتوا به آن را پیاده­سازی نماییم. با تکمیل پیکربندی sanity، توسعه بخش Back-end به پایان می­رسد و اکنون باید به سراغ توسعه بخش Front-end برویم. برای این منظور؛ در بخش بعدی Home Page را طراحی خواهیم کرد.

# طراحی HomePage

برای طراحی HomePage؛ فایل **index.js** در پوشه **'\pages\api'** را باز کرده و تمامی کدهای موجود در آن را حذف کنید، سپس عبارت **rface** را وارد کرده و Enter را بزنید. اکنون کدهای زیر را در آن بنویسید:



HomePage ما از بخش­های **HeroBanner**، **پرفروش­ترین محصولات**، **سایر محصولات** و **Footer** تشکیل می­شود.



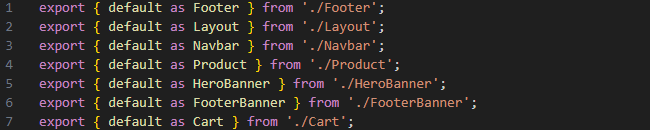
در ادامه به جای بخش HeroBanner و پرفروش­ترین محصولات، یک بنر زیبا طراحی خواهیم کرد. در بخش محصولات؛ به جای آرایه ['product1', 'product2'] روی محصولات واقعی تعریف شده در Sanity حلقه زده و آنها را نمایش می­دهیم. به جای 'Footer' نیز کامپوننت **Footer** را رندر می­نماییم.

پیش از طراحی کامپوننت­ها، قصد داریم استایل­های خود را به پروژه اضافه کنیم. برای این منظور؛ فایل **Home.module.css** را از پوشه **Styles** حذف کرده، همچنین کدهای موجود در فایل **global.css** را با استایل­های مورد نظر خود جایگزین می­کنیم. اکنون می­توانیم این فایل را در **\_app.js** وارد کرده و از کلاس­های آن برای استایل­دهی عناصر خود استفاده کنیم.

یک پوشه به نام **components** به پروژه اضافه کنید. تمامی کامپوننت­های مورد نیاز پروژه، در این پوشه قرار می­گیرند. فایل­های زیر را به این پوشه اضافه کنید :

* **Cart.jsx**
* **Footer.jsx**
* **FooterBanner.jsx**
* **HeroBanner.jsx**
* **Layout.jsx**
* **Navbar.jsx**
* **Product.jsx**

با استفاده از **'rafce'** می­توانید ساختار اولیه فایل­ها را ایجاد کنید. پس از تعریف تمامی کامپوننت­ها، می­توانیم آنها را درون فایلی با نام **index.js** صادر کنیم :

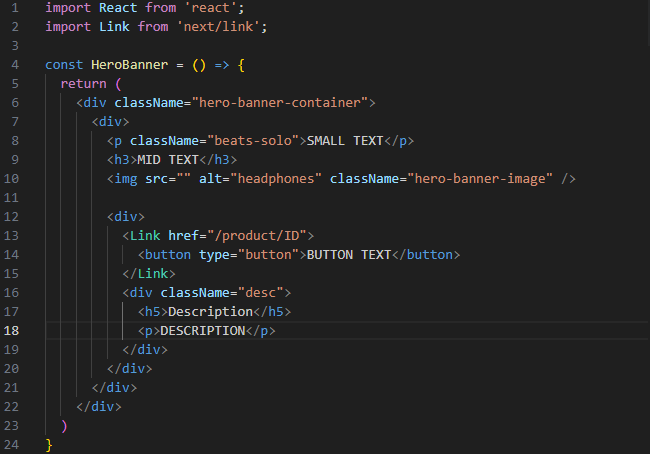


اکنون می­توانیم با یک خط کد، کامپوننت­های مورد نیاز خود را در هر فایلی در پروژه، وارد کنیم. به عنوان مثال :



# طراحی HeroBanner

اولین کامپوننتی که طراحی می­کنیم، **HeroBanner** است :



این کامپوننت، از **دو متن** **ساده** (SMALL TEXT و MID TEXT)، **یک تصویر** (hero-banner-image)، **یک دکمه** و **توضیحات** (desc) تشکیل شده می­شود. تمام داده­هایی که با حروف بزرگ مشخص شده­اند، مستقیماً از پلتفرم Sanity گرفته می­شوند؛ بنابراین در گام بعدی به Sanity متصل شده و داده­ها را از آن واکشی می­کنیم. Sanity این امکان را فراهم کرده است که محتوای سایت را به صورت پویا مدیریت نماییم.

# اتصال به Sanity و واکشی اطلاعات

برای اتصال به Sanity، کارهای زیر را انجام دهید :

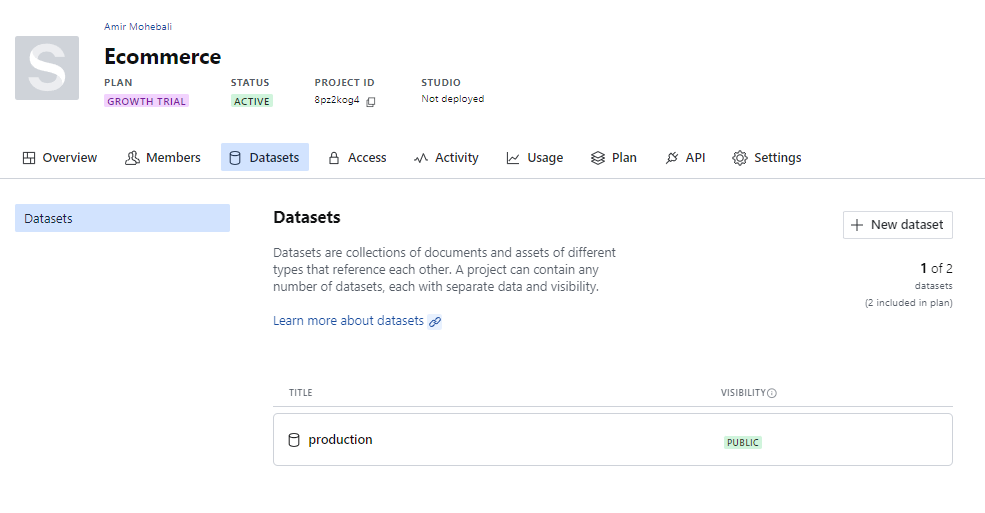
یک پوشه به نام **'lib'** به پروژه اضافه کرده و فایل **client.js** را داخل آن تعریف کنید. اکنون کدهای زیر را در این فایل وارد کنید :

ابتدا توابع لازم را import می­کنیم. برای ایجاد Sanity Client، یک متغیر به نام **client** تعریف کرده و سپس تابع **createClient()** را برای آن فراخوانی می­کنیم. این تابع، تنها یک پارامتر دارد که یک شی است و باید برای آن، چند خصوصیت (property) مشخص کنیم: **projectId** که پروژه مورد نظر ما را برای اتصال مشخص می­کند، **dataset** که دو مقدار 'production' و 'development' دارد، **apiVersion** یا نسخه api، **useCdn** با دو مقدار 'true' و 'false' و یک **token**.

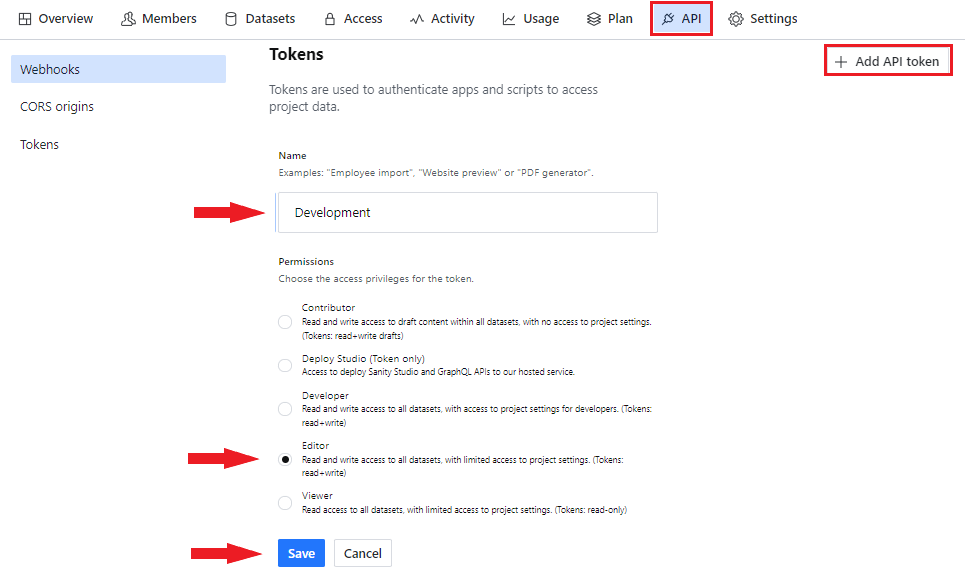
برای دسترسی به مقادیر فوق؛ باید وارد Sanity Management شویم. برای این منظور Terminal را باز کرده و دستور زیر را وارد می­کنیم :

**cd ./sanity**

**sanity manage**



برای مقدار token مشابه تصویر زیر به بخش **API** 🡪 **Tokens** رفته یک token جدید ایجاد می­کنیم :



سپس کلید تولید شده را کپی می­کنیم اما به دلایل امنیتی، مقدار آن را برای خصوصیت token در sanityClient قرار نمی­دهیم. ابتدا یک فایل env. در پوشه ecommerce ایجاد می­کنیم. در این فایل، عبارت **NEXT\_PUBLIC\_SANITY\_TOKEN** را نوشته و مقدار کلید را به آن انتساب می­دهیم :

در آخر برای دسترسی به تصاویر sanity از تابع **imageUrlBuilder()** استفاده کرده و client تعریف شده را به آن پاس می­دهیم. اکنون از طریق **builder** و تابع image() آن می­توانیم به تصاویر ذخیره شده در sanity دسترسی داشته باشیم :

مراحل فوق؛ یک بار در پروژه انجام می­شود و پس از آن می­توانیم از sanity client در هر جایی از کد استفاده کنیم.

یادت باشه ساختار index.js برای قرارگیری کامپوننت ها رو هم قرار بدی.

# طراحی Button

اکنون قصد داریم یک دکمه با قابلیت استفاده مجدد، طراحی کنیم :



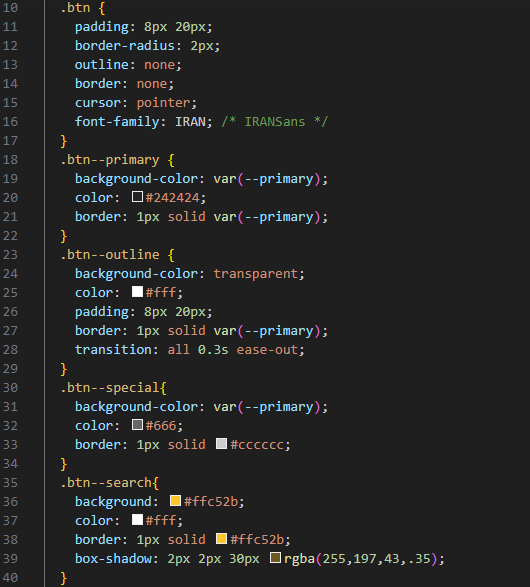
کامپوننت Button **'props'های** زیر را دریافت می­کند :

**: to** لینک مربوط به کامپوننتی که با کلیک بر روی دکمه، رندر می­شود.

**: children** عنوان دکمه

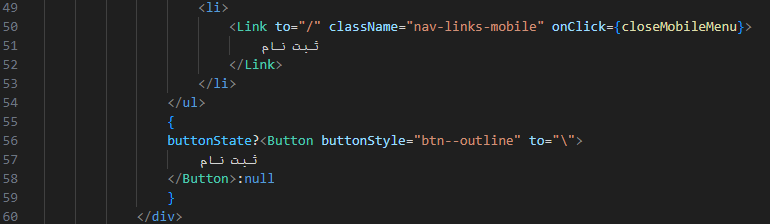
**: buttonStyle - buttonSize** دکمه طراحی شده با توجه به محل قرارگیری آن، دارای استایل­ها و اندازه­های متفاوتی می­باشد؛ بنابراین نوع دکمه باید در هنگام استفاده از آن تعیین شود.

* استایل مربوط به دکمه­های مختلف، در کلاس­های مجزا در فایل Button.css تعریف شده است.

## اضافه کردن دکمه ثبت نام به Menu

پس از طراحی Button؛ اکنون باید دکمه **" ثبت نام "** را به منو اضافه کنیم. برای این منظور، در کامپوننت **Navbar** به صورت زیر از کامپوننت **Button** استفاده می­کنیم :



دکمه **" ثبت نام "** تنها زمانی به منو اضافه می­شود که عرض صفحه کمتر از 960px باشد. بنابراین؛ ابتدا یک state به نام **'buttonState'** تعریف می­کنیم، دکمه **" ثبت نام "** با توجه به مقدار این state نمایش داده می­شود. پس از تعریف state، کنترل کننده رویداد **'resize'** را به شی window اضافه می­کنیم. در تابع **handleButton()** هنگامی که عرض پنجره مرورگر (window.innerWidth) کمتر از 960px باشد، مقدار state **'buttonState'** را برابر false و در غیر این صورت، مقدار آن را برابر true قرار می­دهیم :

# منابع

بخش سوم

* [LogRocket](https://blog.logrocket.com/react-router-v6-guide/)
* [digitalocean](https://www.digitalocean.com/community/tutorials/how-to-handle-routing-in-react-apps-with-react-router#step-3-accessing-route-data-with-hooks)

بخش چهارم

* [stackabuse](https://stackabuse.com/get-http-request-in-react/)
* [stackabuse](https://stackabuse.com/post-http-request-in-react/)
* [freecodecamp](https://www.freecodecamp.org/news/axios-react-how-to-make-get-post-and-delete-api-requests/)
* [medium](https://medium.com/@vdsnini/understanding-interceptors-in-axios-intercepting-and-enhancing-http-requests-5cdfb53a6b51)
* [medium](https://medium.com/@barisberkemalkoc/axios-interceptor-intelligent-db46653b7303)

بخش پنجم

* [freecodecamp](https://www.freecodecamp.org/news/context-api-in-react/)
* [virgool](https://virgool.io/iran-react-community/%D9%86%D8%AD%D9%88%D9%87-%DB%8C-%D8%A7%D8%B3%D8%AA%D9%81%D8%A7%D8%AF%D9%87-%D8%A7%D8%B2-context-%D8%AF%D8%B1-react-xsmrpa8uybym)
* [freecodecamp](https://www.freecodecamp.org/news/usereducer-hook-react/)
* [react.dev](https://react.dev/reference/react/useReducer)

بخش ششم

* [jest](https://jestjs.io/docs/getting-started)
* [enzyme](https://enzymejs.github.io/enzyme/)
* [npmjs](https://www.npmjs.com/package/@cfaester/enzyme-adapter-react-18)
* [sinonjs](https://sinonjs.org/)
* [github](https://github.com/testing-library)
* [testing-library](https://testing-library.com/docs/react-testing-library/intro/)

بخش هشتم

* [razzlejs](https://razzlejs.org/getting-started)
* [github](https://github.com/jaredpalmer/razzle)