**SAAS Application**

1. **npx create-next-app @latest**
2. **shadcn.UI**

npx shadcn-ui@latest init

* default
* slate
* yes
* copy theme code and paste inside global.css

1. for button

* npx shadcn-ui@latest add button

1. **for mobile menu**

* npx shadcn-ui@latest add sheet

1. **Kinde authentication**

* npm i @kinde-oss/kinde-auth-nextjs
* configure environment variable
* Set up Kinde Auth Route Handlers Create the following file app/api/auth/[kindeAuth]/route.js inside your Next.js project. Inside the file route.js put this code:

import {handleAuth} from "@kinde-oss/kinde-auth-nextjs/server";

export const GET = handleAuth();

This will handle Kinde Auth endpoints in your Next.js app.

Important! Our SDK relies on this file existing in this location specified above.

* Add <LoginLink>Login </LoginLink> and <RegisterLink>Register</RegisterLink> inside Navbar
* Add new component inside components named as UserNav.tsx to handle logout and other functionality

1. **Dropdown menu**

* Install dropdown menu to handle logout functionality

npx shadcn-ui@latest add dropdown-menu

* Install avatar

npx shadcn-ui@latest add avatar

* set avatar image based on the name dynamically

<https://avatar.vercel.sh/rauchg>

1. **Create ne component so added new folder inside app named as sell and create new file inside it named as page.tsx and linked in UserNav asChild component**

* Insatll card inside SellRote

npx shadcn-ui@latest add card

* Install input

npx shadcn-ui@latest add input

* Install label

npx shadcn-ui@latest add label

and implement the card inside **page.tsx**

1. **Add new file inside component named as SelectCategory.tsx**
2. Create new folder inside app folder named as **lib** and add new file inside named as **categoryItems.tsx**

Insatll npm install lucide for icons

Now bind the categoryItem.tsx into SelectedCategory.tsx

Add new section for price in SelectCategory

Add new section for summary section in SelectCategory

npx shadcn-ui@latest add textarea

1. **Install tiptap kit for editor**

* npm install @tiptap/react @tiptap/pm @tiptap/starter-kit
* create new component inside components named as Editor.tsx implement the TipTap editor and import inside page.tsx of sell folder
* add tailwind css plugin npm install -D @tailwindcss/typography
* Import inside tailwindcss.config and add the plugins require('@tailwindcss/typography'),

1. **Login to uploadthing and create app to upload files**

* Install npm install uploadthing @uploadthing/react
* Add the API key from dashboard to env file
* Setup a FileRouter. Create new folder inside app/api folder named as uplaodthing and create new file inside uplaodthing named as core.ts and implement file upload functionality in core.ts
* Create a Next.js API route using the FileRouter. Add new file inside uplaodthing folder named as route.ts
* Now Add UploadThing style import import { withUt } from "uploadthing/tw"; in tailwind.config.ts and wrap with withUT
* Now create The Uploadthing Components (Recommended). Add new file inside lib folder named as uploadthing.tsx
* Mount A Button And upload. Now import inside page.tsx of sell component
* Now UploadThings SSR Plugins inside layout.tsx

1. **Configgre the Supabase database**

* Create a project in Supabase and configure the **DatabaseUrl** and DirectUrl into env to save the data into database.
* Configure the Prisma an open source ORM
* Then, install the Prisma CLI as a development dependency in the project: npm i -D prisma
* Then insatll prisma client: npm i @prisma/client
* Then initialize the prisma: npx prisma init
* Now add **DATABASE\_URL** and **DIRECT\_URL** inside env
* Now add Transaction and session mode url inside env for DATABASE\_URL and DIRECT\_URL
* Now create User model inside Prisma.schema
* Now synch the database by running command npx prisma db push

1. **Now create new folder inside app/auth named as creation and create new file inside named as route.ts to handle the API request**

* Implement the api handler inside route.ts.
* Now goto nextjs prisma best practices(<https://www.prisma.io/docs/orm/more/help-and-troubleshooting/help-articles/nextjs-prisma-client-dev-practices>) and copy the solution and create new file inside app/lib named as db.ts and paste the solution.
* Now defined the API handler and response inside route.ts
* Then modify .env *KINDE\_POST\_LOGIN\_REDIRECT\_URL=http://localhost:3000* **to** *KINDE\_POST\_LOGIN\_REDIRECT\_URL=http://localhost:3000/api/auth/creation*
* Now restart the development server and login again and the run npx prisma studio to check database details.

1. **Now let’s create product model inside prisma schema**

* Created product model
* Created category type
* And added one to many relationship between Product and User
* Now run npx prisma db push to sync the database