

Project Name: Medsy

About

This project includes Google Spreadsheet and serverless API routes. On visiting, the app calls get products method, which stores the product's info . On order, the app calls /api/order to place a order on google spreadsheet.

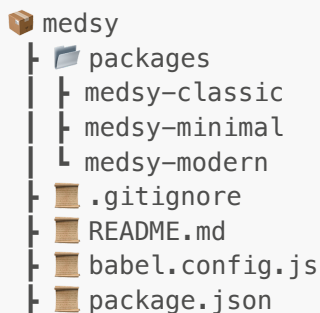
Requirement:

1. Node Version: > 12.13.0
2. Yarn Version: 1.22.24 [yarn 2 isn't tested yet]
3. Google Account

Technology we use

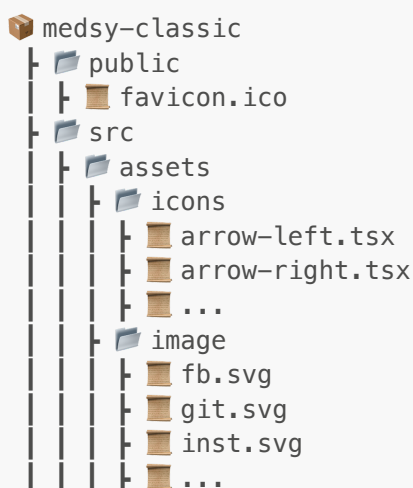
1. [React](#) (A JavaScript library for building user interfaces)
2. [NextJs](#) (React SSR and SSG Framework)
3. [Create Next App](#)
4. [yarn workspaces](#)
5. [Google Sheet](#) (As Database) with a serverless API
6. [API routes](#) by NextJs
7. [Instagram Oembed](#)

Folder Structure



```
medsy
├── packages
│   ├── medsy-classic
│   ├── medsy-minimal
│   └── medsy-modern
├── .gitignore
├── README.md
├── babel.config.js
└── package.json
```

within individual packages



```
medsy-classic
├── public
│   └── favicon.ico
├── src
│   ├── assets
│   │   ├── icons
│   │   │   ├── arrow-left.tsx
│   │   │   ├── arrow-right.tsx
│   │   │   └── ...
│   │   └── image
│   │       ├── fb.svg
│   │       ├── git.svg
│   │       ├── inst.svg
│   │       └── ...
└── ...
```

- └─ styles
 - └─ index.css
 - └─ scrollbar.css
- └─ components
 - └─ carousel
 - └─ carousel.tsx
 - └─ active-link.tsx
 - └─ button.tsx
 - └─ cart-item.tsx
 - └─ cart-modal.tsx
 - └─ counter.tsx
 - └─ customer-contact-form.tsx
 - └─ feature-block.tsx
 - └─ input.tsx
 - └─ item-card.tsx
 - └─ search.tsx
 - └─ section-title.tsx
 - └─ success-modal.tsx
- └─ containers
 - └─ drawer
 - └─ views
 - └─ cart.tsx
 - └─ checkout.tsx
 - └─ menus.tsx
 - └─ no-item.tsx
 - └─ product-details.tsx
 - └─ drawer.tsx
 - └─ layout
 - └─ footer.tsx
 - └─ header.tsx
 - └─ layout.tsx
 - └─ call-to-action.tsx
 - └─ hero-block.tsx
 - └─ how-it-works.tsx
 - └─ products.tsx
- └─ contexts
 - └─ cart
 - └─ cart.provider.tsx
 - └─ cart.reducer.tsx
 - └─ drawer
 - └─ drawer.provider.tsx
 - └─ search
 - └─ use-search.tsx
- └─ helpers
 - └─ constants.tsx
 - └─ get-initials-or-option.tsx
 - └─ use-ref-scroll.tsx
 - └─ use-searchable.tsx
 - └─ ...
- └─ pages
 - └─ api
 - └─ order.tsx
 - └─ _app.tsx
 - └─ _document.tsx
 - └─ index.tsx
-
- └─ .env.local.sample
- └─ .gitignore
- └─ README.md

```
├── babel.config.js
├── tsconfig.json
├── next.config.js
├── package.json
├── postcss.config.js
└── tailwind.config.js
```

Getting Started

Go to project root like:

```
cd medsy
```

open this project with your favorite editor (recommended : [Visual Studio code](#))

configure google sheet:

BUT FIRST -- Set up your google project & enable the sheets API 📁

1. Go to the [Google Developers Console](#)
2. Select your project or create a new one (and then select it)
3. Enable the Sheets API for your project: _ In the sidebar on the left, select APIs & Services > Library _ Search for "sheets" _ Click on "Google Sheets API" _ Click the blue "Enable" button

Setup Instructions

1. Follow steps above to set up project and enable sheets API
2. Create a service account for your project
 - In the sidebar on the left, select APIs & Services > Credentials
 - Click blue "+ CREATE CREDENTIALS" and select "Service account" option
 - Enter name, description, click "CREATE"
 - You can skip permissions, click "CONTINUE"
 - Click "+ CREATE KEY" button
 - Select the "JSON" key type option
 - Click "Create" button
 - your JSON key file is generated and downloaded to your machine (it is the only copy!)
 - Click "DONE"
 - note your service account's email address (also available in the JSON key file)
3. Create two [Google Spreadsheet](#) as **products** and **orders** Give it those column headers, give it some content for **products** sheet:

id	name	image	description	price	manufacturer	type	quantity	dosage	substance
value	value	value	value	value	value	value	value	value	value

for **orders** sheet:

items	address	phone_number	email	bill_amount
-------	---------	--------------	-------	-------------

items	address	phone_number	email	bill_amount
value	value	value	value	value

- Share the doc (or docs) with your service account using the **email** noted above.
 - Click the Share link in the upper right-hand corner
 - Click the very pale Advanced button
 - Click Save
 - Copy the Link to Share. Your URL should look something like
<https://docs.google.com/spreadsheets/d/1lo6W5XitNvifEXER9ECTsbHhAjXsQLq6VEz7kSPDPiQ/edit?usp=sharing>. It should not have a /d/e in it.
 - note your doc (or docs) key (spreadsheet key is the long id in the sheets URL as:
 1lo6W5XitNvifEXER9ECTsbHhAjXsQLq6VEz7kSPDPiQ)
- Copy the contents of **.env.local.sample** into a new file called **.env.local**(Which is define within every individual package like: **project/packages/medsy-modern/.env.local.sample**)
- Get your account credentials from the downloaded **json** file, It will contain keys such as **project_id**, **client_email** and **private_key**. Set them as environment variables in the **.env.local** file **at the root of your required project**.
- if your demo uses instagram oembed then you also need to set those env to getting those value please follow [the link](#)

At the end your **.env.local** file should be look like:

```
GOOGLE_SPREADSHEET_ID_PRODUCT=1zg-EQsksjruriY5KisfrsYdMZc8o3g4BygcAu906aHUA
GOOGLE_SPREADSHEET_ID_ORDER=1CG0y7qQLCrY5ZKHVbt4nm7v5XInW9jdhdueeUrrcf
GOOGLE_SERVICE_ACCOUNT_CLIENT_EMAIL=example@example.iam.gserviceaccount.com
GOOGLE_SERVICE_ACCOUNT_PRIVATE_KEY= "-----BEGIN PRIVATE KEY-----....-----END
PRIVATE KEY-----\n"
```

NOTE:

for demo medsy-modern you need to

- create another google sheet for like **product** sheet called **category** which contain **id**, **name**, **slug**, **image_icon_url** columns.
- also add one more column at your **product** google sheet which is **category_ids** (it's contain the category ids from your category sheet. for multiple category use **comma(,) as separator**)

Second, run the development server:

step-2: within the project root run

```
yarn
# or
yarn install
```

step-3: after successful installation of required packages run you required project or package like:

```
yarn dev:medsy-classic # to run the medsy-classic package or project
# or
yarn dev:medsy-minimal # to run the medsy-minimal package or project
# or
yarn dev:medsy-modern # to run the medsy-modern package or project
```

Open <http://localhost:3000> with your browser to see the result.

You can start editing the page by modifying `pages/index.tsx`. The page auto-updates as you edit the file.

NOTE:

please check the `package.json` file for other useful scripts like `clean`

if you want to clean node_modules and build's folder run

```
yarn clean
```

Basic Customization

- Edit the menu: If you want to add/delete menu items in the drawer menu, then please open `containers -> drawer -> views -> menus.tsx` file and add or delete menu items inside the `menus array`.
Note: You will find all the drawer views codes in `containers -> drawer -> views` folder.
- LOGO / Phone number: If you want to change the logo, then please open `containers -> layout -> header.tsx` file and put your logo / phone number here
- Add Custom data column: According to your needs you can add custom data column at your google sheet and the column title would be the data property/field.[empty column isn't supported(mean each column just have a name/title)]

Learn More

To learn more about Next.js, take a look at the following resources:

- [Next.js Documentation](#) - learn about Next.js features and API.
- [Learn Next.js](#) - an interactive Next.js tutorial.

Email Integration and Base Email template:

1. nodemailer
2. mailgun

We provide integration code for nodemailer and mailgun you should find them at ``src/helpers`` folder called ``nodemailer.ts`` and ``mailgun.ts`` to use them go to the specific file and add your credentials then go to ``src/pages/api/order.tsx`` file and uncomment the commented code.

```
// import { sendMail } from 'helpers/nodemailer';
...
```

```
// await sendMail('support@redq.io', email, 'Order Received By Medsy', {  
//   items,  
//   bill_amount,  
// });
```

Note We use [mailgen](#) for basic template for more information please visit their repository at github.

Deployment

Deploy on Vercel(Previously known as ZEIT Now)

The easiest way to deploy your Next.js app is to use the [Vercel Now Platform](#) step-1: go to your project root like:

```
cd medsy
```

step-2: run deployment command like:

```
deploy-vercel:medsy-classic
```

which is promoted for some information fill your information or just stay with the defaults as

```
$ vercel  
? Set up and deploy “~/your-directory-path”? [Y/n] y  
? Which scope do you want to deploy to? your-scope  
? Link to existing project? [y/N] n  
? What's your project's name? your-project-name  
? In which directory is your code located? ./  
Auto-detected project settings (Next.js):  
- Build Command: `next build` or `build` from `package.json`  
- Output Directory: Next.js default  
- Development Command: next dev --port $PORT  
? Want to override the settings? [y/N] N
```

IMPORTANT:

before deployment you need to setup your env variable on vercel go to your project root and set your env values within the `vercel.json` file. (you can also setup env value by others method supported by vercel for more details [follow the link](#))

NOTE:

make sure you had the [vercel-cli](#) install and logged in your account.