

## Welcome to Chawkbazar

Navigation  
Supported platforms & ecosystem for local development  
Requirements  
Special Notice

## How it works

Tech We Have Used  
Admin Dashboard (GraphQL & REST)  
Shop Frontend (GraphQL & REST)  
API

## Getting Started

Installation Video  
Prerequisites  
Frontend  
Getting Started  
For MAC and Linux(with sail and docker)  
Prerequisites  
Installation  
Video  
Getting Started with Frontend  
For Admin :  
For Shop :  
Shop config  
Admin (At the root of the chawkbazar directory, you can run the below commands)  
Shop (At the root of the chawkbazar directory, you can run the below commands)

## Upgrade

ChawkBazar Update - Virutal Private Server  
Step 1: Setup Git - Server  
Install git  
Config for first time  
Prepare Git Repository  
Git & Github  
Step 2: Shut Down Current Process  
Step 3: Local Repository & Updated Code  
Step 4: Update API  
Step 5: FrontEnd Project Build  
Config Next Admin App For /admin Sub Directory  
Install & Build  
Build the project  
Upload to GitHub  
Step 6: Upload Frontend & Run  
Run frontend app  
Server migration guide  
How to upgrade the existing deployed laravel 8 server to laravel 9?  
Laravel version upgrade  
How to upgrade the existing deployed laravel 9 server to laravel 10?  
Breaking changes  
Version 2.2.2 to 3.0.0 update

## Admin dashboard

A portion of order management. Order status change.  
Coupon Management  
Customer Management  
Tax Management  
Shipping Management

[Shipping Management](#)  
[Settings Management](#)

### **Available Scripts:**

#### **Customization**

For customizing the template's default site settings:

#### **Styles and Assets**

CSS styles:

Icons:

For Adding a custom Icon:

#### **Menu**

[Data Structure](#)

#### **Banner**

[Data Structure](#)

[Data Mapping](#)

[homeOneBanner](#)

[homeOneHeroBanner](#)

[homeTwoHeroBanner](#)

[homeThreeBanner](#)

[homeThreeMasonryBanner](#)

[bannerGrid](#)

[promotionBanner](#)

[promotionBannerTwo](#)

[bannerDataOne](#)

[bannerDataTwo](#)

[homeThreeProductsBanner](#)

[homeFourBanner](#)

[homeFourGridBanners](#)

[homeFiveBanner](#)

#### **Brand**

#### **Utilities**

#### **Backend Integration**

API Integration:

Data Fetching

Uses Example:

#### **Deployment**

[AWS \(Amazon Web Service\)](#)

[How to create ec2 server?](#)

[Domain Setup](#)

[Login to Server](#)

[Change permission .pem](#)

[Virtual Private Server \(Automated Script\)](#)

[Prerequisite](#)

[At first login your server from terminal](#)

[Upload api and deployment project to Virtual Server form your PC - RUN on Local PC](#)

[Server Environment setup script - RUN on Virtual Server](#)

[Nginx Setup And Settings - RUN on Virtual Server](#)

[Backend build - RUN on Virtual Server](#)

[Frontend build script - RUN on Local PC](#)

[Frontend run script - RUN on Virtual Server](#)

[Vercel](#)

[vercel.com](#)

[API](#)

[Frontend](#)

[Virtual Private Server \(Manual\)](#)

[Access Server](#)

[Install NodeJS & Required Application](#)

- Install NodeJS
- Install Yarn
- Install Zip & Unzip
- Install PM2
- Setup Server
- Introduction
- Step 1 - Installing Nginx
- Step 2: Adjusting the Firewall
- Step 3 – Checking your Web Server
- Step 4 - Install MySQL
- Step 5 - Install PHP
- Step 6 - Create MySQL Database & User
- Step 7 - Change permission for the `www` folder
- Step 8 - Upload API to Server
- Step 9: Setting Up Server & Project
- Step 10 - Create New Nginx for the domain

Secure Server

- Step 1: Secure Nginx with Let's Encrypt

Install API

- Step 1: Build and Run `api`

FrontEnd Project Build

- Step 1 - Config Next Admin App For /admin Sub Directory

- Step 2 - Install & Build

- Step 3 - Build the project

  - Shop

  - Admin

  - Image Config

  - Build Project

  - Install FrontEnd And Run

  - Run frontend app

cPanel

- Access Server

- Create Subdomains

- Install API

- Install FrontEnd

- FrontEnd Project Build

- step 1 - Build Custom Server

  - shop

  - admin rest

- Step 2 - Install & Build

- Step 3 - Build the project

- Install NodeJs Project

## SEO and Analytics

- SEO

- Analytics

## Laravel API

- Introduction

- Getting Started

  - Installation Video

  - Prerequisites

  - Resources you might need

  - Packages we have used

- Installation

- For MAC and Linux(with sail and docker)

  - Prerequisites

- [Installation](#)
- [Configuration](#)
- [Console Commands](#)
- [Development](#)
- [REST API](#)
  - [Endpoints](#)
  - [Folder structure](#)
  - [config](#)
  - [database](#)
  - [Enums](#)
  - [Events](#)
  - [Listeners](#)
  - [Mail](#)
  - [Notifications](#)
  - [Providers](#)
  - [stubs](#)
- [GraphQL API](#)
  - [Laravel GraphQL API Endpoint](#)
  - [Alternatives](#)
  - [Folder Structure](#)
  - [Payment Gateway](#)
  - [Extending The Functionality](#)
  - [Deployment](#)
- Social Login**
  - [Google](#)
  - [Facebook](#)
- OTP (Mobile Number Verification)**
  - [Configuration](#)
    - [Twilio Configuration:](#)
    - [MessageBird:](#)
  - [FrontEnd Demo](#)
    - [Login with Mobile Number:](#)
    - [OTP Verification During Checkout Process:](#)
    - [OTP Verification For Updating Mobile Number:](#)
- Email Verification**
  - [How to turn on this feature](#)
  - [Admin view of this feature](#)
  - [Shop view of this feature](#)
  - [Configuration](#)
- Payment**
  - Stripe**
    - [Stripe integrate inside ChawkBazar.](#)
    - [How to create & setup Stripe information properly?](#)
    - [Special Notes for Stripe users.](#)
    - [How can I add card in my user profile for future payments in Stripe?](#)
  - PayPal**
    - [PayPal integrate inside ChawkBazar.](#)
    - [How to create & setup PayPal information properly?](#)
  - RazorPay**
    - [RazorPay integration inside PickBazar.](#)
    - [How to create & setup RazorPay information properly?](#)
    - [Special Notes for RazorPay.](#)
  - Mollie**
    - [Mollie integrate inside PickBazar.](#)
    - [Mollie Webhook settings](#)
    - [How to create & setup Mollie information properly?](#)

Special Notes for Mollie.

## Paystack

Paystack integrate inside PickBazar.

Paystack Webhook settings

How to create & setup Paystack information properly?

Special Notes for Paystack.

## Integratioin of new payment gateway

Getting Started with API

Step 1: Install and configure the payment gateway package

Step 2: Add payment gateway name in the Enum.

Step 3: Configure the Payment Facade for the new payment gateway.

Step 4: Using that payment gateway for submitting the order.

Step 5: How to Setup webhook in ChawkBazar-laravel follow the steps

Getting Started with admin dashboard.

Getting Started with shop front.

Redirect-base Payment Gateway

Non-redirect based payment gateway

## Multiple payment gateway

How to activate multiple payment gateways [ in Admin ] ?

Description [ sequence by number in the screenshot ]

How to use multiple payment gateways from customer end [ in Shop ] ?

1. Select a payment gateway during place order

2. Select different payment gateway ( if needed )

Description [ sequence by number in the screenshot ]

## Translation

Existing Language

For Rest

New Language

Default Language

## Multivendor

Create New Shop

Dashboard:

Attributes:

Products:

Order:

Staff:

WithDraw:

User Roles:

Super Admin:

Store Owner:

Staff:

Withdraw Payment:

FrontEnd Shop

## Export Import

Simple Product

variable products

## Settings Management

Change Logo

API

Change Site information

Shipping class, Tax class & Free Shipping

Payment Gateway Management

Customer

SEO

Add or Remove Delivery Schedule

Shop Settings & Google Map

## New Static Page

Shop,  
Admin,

### FAQ:

- How to change the default layout?
- Why showing client\_id error while Google social login
- How to configure Stripe payment gateway?
- Why am I facing "You may need an appropriate loader to handle this file type" during running shop rest?
- I am changing schema files but changes is not working
- Changing .env files but not getting the changes
- Changing route but not getting the changes.
- I have set `STRIPE_API_KEY` in .env but still getting error.
- Can I use it with my existing laravel?
- Why am I getting `Access denied for user`?
- Why am I getting permission issue during deployment?
- Why am I getting "The GET method is not supported for this route. Supported methods: HEAD"?
- I am getting "Cannot use 'Mixed' as class name as it is reserved" after access graphql endpoint.
- How to resolve the `Load More Infinity` loading issue?
- I'm trying to upload images, but the images are not displayed on the frontend?

API  
Admin  
Shop  
Build Frontend,

How to resolve the `502 Bad Gateway` error for frontend/admin?

Reason 1:

PORT:

Reason 2:

How to resolve `javascript heap out of memory` issue?

Image upload throw Internal Server Error; how to resolve that?

Sometimes my server shutdown or halts automatically. After restart, it works fine again; how to resolve that?

S3 uploading doesn't work after adding credentials to .env; how to resolve that?

How to rebuild the project?

How to increase upload size?

How to resolve `docker: invalid reference format: repository name must be lowercase. ?`

How to remove the existing payment gateway?

Why checkout `Place Order` button is disabled?

How do I log in to the admin panel as the main administrator?

How to disable product popup view and redirect to a single product page?

[Support](#)

[Thank You!](#)

# Welcome to Chawkbazar

---

Welcome to the ChawkBazar Laravel documentation! Fastest Ecommerce App built with React, NextJS, TypeScript, Laravel and Tailwind CSS.

## Navigation

---

You can find different topics in the table of contents. On desktop, you should see it in the left sidebar. On mobile, you should see it after pressing an icon with Hamberger in the top right corner.

# Supported platforms & ecosystem for local development

---

- Compatible Browsers (Firefox, Safari, Chrome, Edge)
- Node.js 16.15.0 or later
- PHP 8.1
- MySQL 8
- MacOS, Windows, and Linux are supported

## Requirements

---

- Node.js 16.15.0 or later
- yarn
- PHP 8.1
- MySQL 8
- ext\_curl
- ext\_gmp
- editor: [Visual Studio Code](#)(recommended)

## Special Notice

---

One new chapter named `Upgrade` was introduced along with some sub-chapters for easy migration process of app & server ecosystem. Please visit this chapter each time to check if any new changes are introduced. [Breaking changes](#)

Please follow this video playlist with the documentation, and it'll make installation and customization process relatively easy.

## How it works

---

### Tech We Have Used

---

We have used monorepo folder structure with Yarn Workspace. In our app we have three different services:

- api
- shop
- admin

Tech specification for specific part is given below:

### Admin Dashboard (GraphQL & REST)

---

- [Nextjs](#)
- [Typescript](#)
- [Tailwindcss](#)

- [React Hook Form](#)
- [React-Query](#)

## Shop Frontend (GraphQL & REST)

---

- [NextJs](#)
- [TypeScript](#)
- [Tailwindcss](#)
- [React Hook Form](#)
- [React-Query](#)

## API

---

- Laravel

## Getting Started

---

For getting started with the template you have to follow the below procedure. For quick guide you can check below videos for installation.

## Installation Video

---

*Please follow this video with the documentation, and it'll make the installation process on windows relatively easy.*

## Prerequisites

---

- PHP 8.1
- Composer
- Xamp/Wamp/Lamp/Mamp for any such application for apache, nginx, mysql
- PHP plugins you must need
  - simplexml
  - PHP's dom extension
  - mbstring
  - GD Library

## Frontend

- node(16.15 or later)
- yarn
- editor: [Visual Studio Code](#)(recommended)

## Getting Started

---

1. First download the file from codecanyon.
2. Unzip the downloaded file and folder structure you get

```
chawkbazar
|-- chawkbazar-api
|-- admin
|-- shop
```

3. From the above folder structure you should notice that our app has three parts `chawkbazar-api`, `shop` and `admin`. So you have to run all the parts separately and sequentially.

## Getting started with api

- Run and start xamp/wamp/lamp/mamp and start all the services
- Create a database in your mysql and put those info in next step
- Rename `.env.example` file to `.env` and provide necessary credentials. Like database credentials stripe credentials, s3 credentials(only if you use s3 disk) admin email shop url etc.
  - Specially check for this `env` variables

```
DB_HOST=localhost
DB_DATABASE=chawkbazar_laravel
DB_USERNAME=root
DB_PASSWORD=
```

- Run `composer install`

```
▶ composer install
Installing dependencies from lock file (including require-dev)
Verifying lock file contents can be installed on current platform.
```

```
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi
Discovered Package: barryvdh/laravel-dompdf
Discovered Package: bensampo/laravel-enum
Discovered Package: cviebrock/eloquent-sluggable
Discovered Package: facade/ignition
Discovered Package: fideloper/proxy
Discovered Package: fruitcake/laravel-cors
Discovered Package: ignited/laravel-omnipay
Discovered Package: intervention/image
Discovered Package: laravel/legacy-factories
Discovered Package: laravel/sail
Discovered Package: laravel/sanctum
Discovered Package: laravel/tinker
Discovered Package: mll-lab/laravel-graphql-playground
Discovered Package: nesbot/carbon
Discovered Package: nunomaduro/collision
Discovered Package: nuwave/lighthouse
Discovered Package: pickbazar/shop
Discovered Package: prettus/l5-repository
Discovered Package: spatie/laravel-medialibrary
Discovered Package: spatie/laravel-permission
Package manifest generated successfully.
95 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
```

- run `php artisan key:generate`

```
▶ php artisan key:generate
Application key set successfully.
```

- Run `php artisan marvel:install` and follow necessary steps.

```

Installing Pickbazar Dependencies...
Do you want to migrate Tables? If you have already run this command or migrated tables then be aware, it will erase all of your data. (yes/no) [no]:
> yes

Migrating Tables Now....
Dropped all tables successfully.
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (79.70ms)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (58.57ms)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (79.56ms)
Migrating: 2019_12_14_000001_create_personal_access_tokens_table
Migrated: 2019_12_14_000001_create_personal_access_tokens_table (143.06ms)
Migrating: 2020_04_17_194830_create_permission_tables
Migrated: 2020_04_17_194830_create_permission_tables (839.79ms)
Migrating: 2020_06_02_051901_create_pickbazar_tables
Migrated: 2020_06_02_051901_create_pickbazar_tables (1,601.56ms)
Migrating: 2020_10_26_163529_create_media_table
Migrated: 2020_10_26_163529_create_media_table (71.72ms)
Tables Migration completed.

Do you want to seed dummy data? (yes/no) [no]:
> yes

Copying necessary files for seeding....
File copying successful
Seeding...
Seed completed successfully!

Do you want to create an admin? (yes/no) [no]:
> no

Copying resources files...
Installation Complete

```

- For image upload to work properly you need to run `php artisan storage:link`.

The `[/var/www/html/public/storage]` link has been connected to `[/var/www/html/storage/app/public]`.  
The links have been created.

- run `php artisan serve`

```

▶ php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Sat Apr 10 16:35:26 2021] PHP 8.0.0 Development Server (http://127.0.0.1:8000) started

```

NB: your frontend `NEXT_PUBLIC_REST_API_ENDPOINT` env value will be `localhost:8000/`

For details api doc and requirements details you can go to [Laravel API](#)

## For MAC and Linux(with sail and docker)

There is an alternate installation procedure for linux and mac. You can follow below procedure to getting started with `sail`

### Prerequisites

- Docker

### Installation

### Video

- Run Docker application first
- Now go to your chawkbazar-laravel root directory and run `bash install.sh`. It will guide you through some process. Follow those steps carefully and your app will be up and running
- Navigate to `api` then `sail down` to stop the container. If you want to remove the volumes then `sail down -v`

NB: your frontend `NEXT_PUBLIC_REST_API_ENDPOINT` env value will be `localhost/`

For details api doc and requirements details you can go to [Laravel API](#)

# Getting Started with Frontend

4. After configuring API & running it successfully you can choose the directory where you need to work

Below are the directories where you will choose to work for frontend stuffs

```
cd admin/rest  
cd shop
```

After choosing your working directory

**Go to specific folder and rename the `.env.template` => `.env` and put your api endpoint here. You will find `.env.template` file at the root of your `admin/{chosen-directory-name}` or `shop`**

5. Run yarn at the root directory.

```
# on chawkbazar/root directory  
yarn
```

## 6. Scripts To Run the fronted App

### For Admin :

For starting the admin dashboard part with corresponding api data run below commands.

- using workspace (At the root of the chawkbazar directory, you can run the below commands)

```
yarn dev:admin-rest
```

```
▶ yarn dev:admin-rest  
yarn run v1.22.5  
$ yarn workspace @pick-bazar/admin-rest dev  
$ next dev -p 3002  
ready - started server on 0.0.0.0:3002, url: http://localhost:3002  
info  - Loaded env from /Users/bashar/Codes/pickbazar-sail/frontend/admin-rest/.env.local  
event - compiled successfully
```

- without workspace(if you want to run the command within specific project root of `admin/{chosen-directory-name}`)

```
# for dev mode run below command  
# REST  
yarn dev
```

This command will run the app in development mode. Open the suggested url in your terminal. like => <http://localhost:3000> .

\*\* Note: \*\*

- The page will automatically reload if you make changes to the code. You will see the build errors and lint warnings in the console.
- If you saw any error while running make Sure you setup your API endpoint properly at `.env` file.

## For Shop :

### Shop config

For starting the shop part with corresponding api data run below commands.

- using workspace (At the root of the chawkbazar directory, you can run the below commands)

```
yarn dev:shop-rest
```

```
▶ yarn dev:shop-rest
yarn run v1.22.5
$ yarn workspace @pick-bazar/shop-rest dev
$ next dev -p 3003
ready - started server on 0.0.0.0:3003, url: http://localhost:3003
info  - Loaded env from /Users/bashar/Codes/pickbazar-sail/frontend/shop-rest/.env.local
```

- without workspace(if you want to run the command within specific project root of `shop`)

```
yarn dev
```

\*\* If you want to test your production build admin or shop in local environment then run the below commands. \*\*

### Admin (At the root of the chawkbazar directory, you can run the below commands)

```
# build admin for production
yarn build:admin-rest

#start admin in production mode
yarn start:admin-rest
```

### Shop (At the root of the chawkbazar directory, you can run the below commands)

```
# build shop for production
yarn build:shop-rest

# start shop in production mode
yarn start:shop-rest
```

```

▶ yarn build:shop-rest
yarn run v1.22.5
$ yarn workspace @pick-bazar/shop-rest build
$ next build
info - Loaded env from /Users/bashar/Codes/pickbazar-sail/frontend/shop-rest/.env.local
info - Creating an optimized production build
info - Compiled successfully
info - Collecting page data
info - Generating static pages (48/48)
info - Finalizing page optimization

Page          Size     First Load JS
7_app          0 B      176 kB
  • /[type]
    L css/7936e7aa084bd830e87c.css
      | grocery
      | makeup
      | bags
      [+2 more paths]
  o /404
  • /bakery (ISR: 60 Seconds)
    L css/0dcfb7860d56385c547.css
      | change-password
  λ /checkout
  o /contact
  o /example
  o /help
  o /logout
  o /offers
  λ /order
    L css/db04274d0e1caa11303a.css
      | order-received/[tracking_number]
  λ /orders
    L css/da2daebf602ff72a3fc0.css
  o /privacy
  • /products/[slug]
    L css/efb94a24c22d59f4d562.css
      | products/apples
      | products/baby-spinach
      | products/blueberries
      [+27 more paths]
  λ /profile
  o /terms
  o /ui
    8.82 kB   185 kB
+ First Load JS shared by all
  176 kB
  chunks/00682edc09922f9a0564bd51ebfe3fe9fe3f6bc..837c2b.js   3.14 kB
  chunks/03597eb9deef95bb8e6991ab21a5b170a1b2330ab.CSS.3bbe0c.js 68 B
  chunks/1c2031c8..c533da.js   23.5 kB
  chunks/2c96bea9ebf36e109a607a23c3ac04c26631eb28..f78ccb.js  4.76 kB
  chunks/57ad4ee9d685f4a85f01607d3121ddc66f6f331.d3e7fd.js  35.8 kB
  chunks/6bc2a055fc5fb5a7f770dd80e4657ccaa465f5acf5a..173fc5.js 7.14 kB
  chunks/78478e0567901f0d00137009cf6f751fab214ebc..855475.js  15.3 kB
  chunks/a0a9552a4f74729e02303abd0a7d42186859f2ec0..377727a.js 7.43 kB
  chunks/commons..f7fe9e.js   18.2 kB
  chunks/framework..f6e4f9.js  42.3 kB
  chunks/main..9c0e07.js   6.29 kB
  chunks/pages..app..558fda.js 9.76 kB
  chunks/webpack..d3d782.js  2.3 kB
  css/669ac54f0d96e08d6aC.css  4.38 kB
  css/f53121b36d6036aac827.css 11.2 kB

λ (Server) server-side renders at runtime (uses getInitialProps or getServerSideProps)
o (Static) automatically rendered as static HTML (uses no initial props)
• (SSG) automatically generated as static HTML + JSON (uses getStaticProps)
  (ISR) incremental static regeneration (uses revalidate in getStaticProps)

'+' Done in 96.20s.

▶ yarn start:shop-rest
yarn run v1.22.5
$ yarn workspace @pick-bazar/shop-rest start
$ next start -p 3003
ready - started server on 0.0.0.0:3003, url: http://localhost:3003
info - Loaded env from /Users/bashar/Codes/pickbazar-sail/frontend/shop-rest/.env.local

```

**\*\* Note \*\*:**

Please see `package.json` file for other builtin helper commands.

7. For development purpose we use `yarn workspace` if you want to use it then see the

`package.json` file at root, for various workspace specific command.

- if you prefer single template then just copy the required template folder and separate them. you'll find their `package.json` file within them and follow the command for dev, build, start.

8. For further development & customization check our [Frontend Customization](#) guide.

# Upgrade

## ChawkBazar Update - Virutal Private Server

If you follow this [Virtual Private Server](#) docs to host your site, then follow this documentation to update ChawkBazar to a new version.

To build the frontend you've to update the `API` first. But before that this time, we'll use git and GitHub to make upload and download relatively easy.

# Step 1: Setup Git - Server

This step is only for first update. From second update start from [Step 2](#)

At first, we've to install git on our server and config it.

## Install git

```
sudo apt install git
```

## Config for first time

```
git config --global user.email "you@example.com"  
git config --global user.name "Your Name"
```

Make sure you change `you@example.com` and `Your Name` with your `email` and `name`.

## Prepare Git Repository

At first, go to your `chawkbazar` directory on your server,

```
cd /var/www/chawkbazar
```

2. Then initialize a git on that folder,

```
git init
```

3. Create a new `.gitignore` file

```
nano .gitignore
```

and paste this code to that `.gitignore`,

```
# See https://help.github.com/articles/ignoring-files/ for more about ignoring  
files.  
  
# dependencies  
node_modules  
.pnp  
.pnp.js  
  
# testing  
/coverage  
# *~  
*.swp  
tmp/  
  
# misc  
.DS_Store  
  
# If Packages have their independent repo  
# /packages
```

```
# ignore log files
npm-debug.log*
yarn-debug.log*
yarn-error.log*
.idea/
.vscode/
node_modules/
.DS_Store
*.tgz
my-app*
lerna-debug.log
npm-debug.log*
yarn-debug.log*
yarn-error.log*
/.changelog
.npm/
packages/server/node_modules
packages/server/error.log
packages/server/debug.log
packages/web/.env
packages/reusecore
packages/cloud

.docz
.now
.vercel
```

After that, `save` that file and use this command for the initial commit,

```
git init
```

```
git add .
```

```
git commit -m "Initial commit"
```

Create a separate branch to maintain the updated code.

```
git branch chawkbazar
```

## Git & Github

1. At first, go to <https://github.com/> and create an account first. If you already have an account then `Sign In` on that account.
2. Then create a new repository,

3. After creating the repository you'll get a page like this and from this page copy the `second command block` and go to your `server` using `SSH & terminal`,

The screenshot shows the GitHub 'Quick setup' page for a new repository named 'chawkbazar-update'. It includes options for 'Set up in Desktop' (with 'HTTPS' selected), 'SSH', and a URL. A red arrow labeled '1' points to the 'HTTPS' button. Another red arrow labeled '2' points to the 'Import code' button.

And `paste` that copied command to `pickabazar` folder and press `enter`

It'll ask your GitHub `username` and `password`, provide your GitHub `username` and `password`

Your existing repository is successfully connected with GitHub.

## Step 2: Shut Down Current Process

At first use this command to shut down all the applications for update,

```
pm2 stop 'all'
```

## Step 3: Local Repository & Updated Code

In this step,

1. Clone that `GitHub repository` to your `local computer`
2. Download `update package` from `CodeCanyon`
3. Open your terminal and clone that repository to your computer
4. Checkout to `chawkbazar` branch

```
git checkout -b chawkbazar
```

5. replace `repository` file with downloaded `chawkbazar-laravel` folder content.
6. Then use this command to add all files to git

```
git add .
```

```
git commit -m "Update ChawkBazar API"
```

7. Merge new code with `main` branch

```
git checkout -b main
```

```
git merge chawkbazar
```

In this step, you'll face a merge conflict issue. Make sure you resolve all the conflicts to maintain your customization with the updated code. You can check this [video](#) about resolve merge conflict.

After `resolve` and `commit`, push the code to GitHub.

```
git push origin main
```

## Step 4: Update API

In this step, go to your server terminal and go to `/var/www/chawkbazar` directory and pull all updated code,

```
git pull origin main
```

After pull go to `api` folder,

```
cd api
```

and install composer package and optimize compiled file,

```
composer install
```

```
php artisan optimize:clear
```

With that updated API will be installed. To check go to your `YOUR_API_DOMAIN/products`

## Step 5: FrontEnd Project Build

TypeScript requires a huge chunk of memory to build the project, so if your server has at least 8gb+ of memory, then you can build the project on your server directly. If not, then build the project on your server, then move the folder to the server then serve the project. We'll do the second method in this tutorial.

We'll suggest you build the frontend part on your computer and then move the build file to the server using git and github.

Go to your `local git repository` using terminal.

## Config Next Admin App For `/admin` Sub Directory

Edit `admin/rest/next.config.js`,

add `basePath` for `'/admin'`

```
1 const withPWA = require("next-pwa");
2 const runtimeCaching = require("next-pwa/cache");
3 const { i18n } = require("./next-i18next.config");
4
5 module.exports = withPWA({
6   basePath: "/admin", ←
7   i18n,
8   pwa: {
9     disable: process.env.NODE_ENV === "development",
10    dest: "public",
11    runtimeCaching,
12  },
13
14  images: {
15    domains: [
16      "via.placeholder.com",
17      "res.cloudinary.com",
18      "s3.amazonaws.com",
19      "18.141.64.26",
20      "127.0.0.1",
21      "localhost",
22      "picsum.photos",
23      "pickbazar-sail.test",
24      "pickbazarlaravel.s3.ap-southeast-1.amazonaws.com",
25      "chawkbazarlaravel.s3.ap-southeast-1.amazonaws.com",
26      "lh3.googleusercontent.com",
27    ],
28  },
29}
```

## Install & Build

go to your `root` folder again

Clean previous build file,

```
yarn clean
```

To install all the npm packages run this command,

```
yarn
```

## Build the project

```
yarn build:shop-rest
yarn build:admin-rest
```

## Upload to GitHub

Use this command to add all files to git,

```
git add .
```

```
git commit -m "Build frontend"
```

```
git push origin main
```

## Step 6: Upload Frontend & Run

At first go to your server `chawkbazar` or `git` folder and use this command to pull all the build file,

```
git pull origin main
```

Then install all the node packages,

```
yarn
```

### Run frontend app

Use this command to run frontend app as `PM2` again.

```
pm2 start 'all'
```

Now go to Now, go to your `YOUR_DOMAIN` to access the shop page and `YOUR_DOMAIN/admin` for the access admin section.

## Server migration guide

### How to upgrade the existing deployed laravel 8 server to laravel 9?

At first, remove all the existing `php 7.4` and its extensions by using this command,

```
sudo apt purge php-fpm php-mysql  
sudo apt purge php-mbstring php-xml php-bcmath php-simplexml php-intl php-mbstring php7.4-gd php7.4-curl php7.4-zip composer
```

Then remove the composer,

```
sudo rm /usr/bin/composer
```

Then delete the `vendor` folder from the `chawkbazar-laravel -> chawkbazar-api` folder.

```
cd /var/www/chawkbazar-laravel/chawkbazar-api  
  
sudo chown -R $USER:www-data storage  
sudo chown -R $USER:www-data bootstrap/cache  
  
rm vendor -rf
```

Then install PHP 8, and its extensions,

```
sudo add-apt-repository ppa:ondrej/php  
sudo apt update
```

```
sudo apt install php8.0-fpm php8.0-mysql
```

```
sudo apt install php8.0-mbstring php8.0-xml php8.0-bcmath php8.0-simplexml  
php8.0-intl php8.0-gd php8.0-curl php8.0-zip php8.0-gmp
```

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"  
php -r "if (hash_file('sha384', 'composer-setup.php') ===  
'e21205b207c3ff031906575712edab6f13eb0b361f2085f1f1237b7126d785e826a450292b6cf1d  
64d92e6563bbde02') { echo 'Installer verified'; } else { echo 'Installer  
corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"  
php composer-setup.php  
php -r "unlink('composer-setup.php');"
```

```
sudo mv composer.phar /usr/bin/composer
```

Then update 7.4 fpm to 8.0 fpm from,

```
/etc/nginx/sites-enabled/chawkbazar
```



```
charset utf-8;  
  
# For API  
location /backend {  
    alias /var/www/pickbazar/api/public;  
    try_files $uri $uri/ @backend;  
    location ~ \.php$ {  
        include fastcgi_params;  
        fastcgi_param SCRIPT_FILENAME $request_filename;  
        fastcgi_pass unix:/run/php/php8.0-fpm.sock;  
    }  
}  
  
location @backend {  
    rewrite /backend/(.*)$ /backend/index.php?/$1 last;  
}  
  
# For FrontEnd -> Rest  
location / {  
    proxy_pass http://localhost:3000;  
    proxy_http_version 1.1;  
    proxy_set_header Upgrade $http_upgrade;  
    proxy_set_header Connection 'upgrade';  
    proxy_set_header Host $host;  
    proxy_cache_bypass $http_upgrade;  
}  
  
location /admin {  
    proxy_pass http://localhost:3002/admin;  
    proxy_http_version 1.1;  
    proxy_set_header Upgrade $http_upgrade;  
    proxy_set_header Connection 'upgrade';  
    proxy_set_header Host $host;  
    proxy_cache_bypass $http_upgrade;  
}  
  
error_page 404 /index.php;  
  
location ~ \.php$ {  
    fastcgi_pass unix:/var/run/php/php8.0-fpm.sock;  
    fastcgi_index index.php;  
    fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;  
    include fastcgi_params;  
}  
  
location ~ /.(?!well-known).* {  
    deny all;
```

Then install composer packages,

```
cd /var/www/chawkbazar-laravel/chawkbazar-api
```

```
composer install
```

```
php artisan optimize:clear
```

```
php artisan marvel:install
```

php artisan marvel:install will remove all of your existing data. Ensure you export or backup your data before using that command.

```
sudo chown -R www-data:www-data storage  
sudo chown -R www-data:www-data bootstrap/cache
```

## Laravel version upgrade

### How to upgrade the existing deployed laravel 9 server to laravel 10?

At first, remove all the existing `php 7.4` and its extensions by using this command,

```
sudo apt purge php8.0-fpm php8.0-mysql  
sudo apt purge php8.0-mbstring php8.0-xml php8.0-bcmath php8.0-simplexml php8.0-intl  
php8.0-gd php8.0-curl php8.0-zip php8.0-gmp composer
```

Then remove the composer,

```
sudo rm /usr/bin/composer
```

Then delete the `vendor` folder from the `chawkbazar-laravel -> chawkbazar-api` folder.

```
cd /var/www/chawkbazar-laravel/chawkbazar-api  
  
sudo chown -R $USER:www-data storage  
sudo chown -R $USER:www-data bootstrap/cache  
  
rm vendor -rf
```

Then install PHP 8.1, and its extensions,

```
sudo add-apt-repository ppa:ondrej/php  
sudo apt update
```

```
sudo apt install php8.1-fpm php8.1-mysql
```

```
sudo apt install php8.1-mbstring php8.1-xml php8.1-bcmath php8.1-simplexml  
php8.1-intl php8.1-gd php8.1-curl php8.1-zip php8.1-gmp
```

```
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"  
php -r "if (hash_file('sha384', 'composer-setup.php') ===  
'e21205b207c3ff031906575712edab6f13eb0b361f2085f1f1237b7126d785e826a450292b6cf1d  
64d92e6563bbde02') { echo 'Installer verified'; } else { echo 'Installer  
corrupt'; unlink('composer-setup.php'); } echo PHP_EOL;"  
php composer-setup.php  
php -r "unlink('composer-setup.php');"
```

```
sudo mv composer.phar /usr/bin/composer
```

Then update 8.0 fpm to 8.1 fpm from,

```
/etc/nginx/sites-enabled/chawkbazar
```

```
# For API
location /backend {
    alias /var/www/chawkbazar-laravel/chawkbazar-api/public;
    try_files $uri $uri/ @backend;
    location ~ \\.php$ {
        include fastcgi_params;
        fastcgi_param SCRIPT_FILENAME $request_filename;
        fastcgi_pass unix:/run/php/php8.1-fpm.sock;
    }
}

location @backend {
    rewrite /backend/(.*)$ /backend/index.php?/$1 last;
}

# For FrontEnd
location /{
    proxy_pass http://localhost:3003;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
}

location /admin{
    proxy_pass http://localhost:3002/admin;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
}

error_page 404 /index.php;

location ~ \\.php$ {
    fastcgi_pass unix:/var/run/php/php8.1-fpm.sock;
    fastcgi_index index.php;
    fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
}
```

Then install composer packages,

```
cd /var/www/chawkbazar-laravel/chawkbazar-api
```

```
composer install
```

```
php artisan optimize:clear
```

```
php artisan marvel:install
```

php artisan marvel:install will remove all of your existing data. Ensure you export or backup your data before using that command.

```
sudo chown -R www-data:www-data storage
sudo chown -R www-data:www-data bootstrap/cache
```

## Breaking changes

---

In this chapter the breaking changes will be listed here version wise sequentially. So that our customers can easily track down the changes & take necessary steps during their migration process.

### Version 2.2.2 to 3.0.0 update

- Laravel version 8 to 9 upgradation process was introduced. [Laravel 8 to 9 upgrade](#)
- There are changes coming in database section. Before update & run any database migration command, please keep the database backup.
  - Order table was updated in the database.
  - Order-Status table was deleted. Now, order statuses are managed dynamically
- Before update please keep your existing code backup. Because there are changes inside order code architecture & payment gateway integration system.
- Product inventory system was managed based on new order & payment status.

## Admin dashboard

---

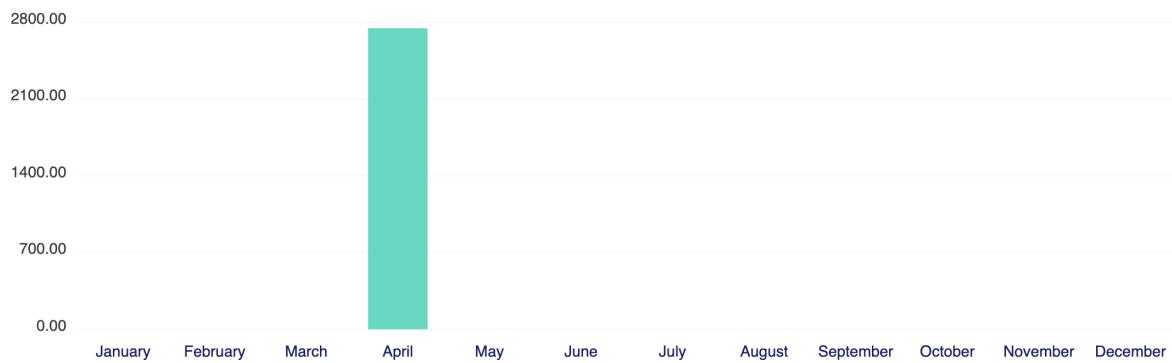
### • Analytics Dashboard

---

You will get a complete analytics dashboard to know the overview of your shop.

Total Revenue (Last 30 Days)		Total Order (Last 30 Days)		New Customer (Last 30 Days)		Todays Revenue	
\$7,844.05		25		16		\$0.00	

Sale History



### Recent Orders

Tracking Number	Total	Order Date	Status
GkszijsvAcae	\$2,744.33	5 days ago	Order Received
Irrdghqshkys	\$1,748.30	14 days ago	Order Dispatched
iTUVhT87QdE8	\$526.98	17 days ago	Order Dispatched
A9mvTnzhGxkc	\$294.49	17 days ago	Order Received
B1OnTtmsHHHE	\$12.80	17 days ago	Order Received
rulb1kz4UIIN	\$2.45	19 days ago	Shipment Refused by Consignee
NqlykelW6awe	\$97.80	18 days ago	Delivered
TfzYmld1AR32	\$18.00	18 days ago	Ready To Dispatch
PXqs8RXQBGM5	\$7.20	19 days ago	Delivered
K2isul12Cetf	\$540.00	19 days ago	Ready To Dispatch

## Popular Products

ID	Name	Type	Price/Unit	Quantity
4	Brussels Sprout		\$5.00	17
3	Blueberries		\$3.00	30
2	Baby Spinach		\$0.60	10
6	Clementines		\$3.00	50
5	Celery Stick		\$6.00	18
7	Sweet Corn		\$5.00	50
8	Cucumber		\$2.50	25
103	Armani Purse		\$80.00	50
9	Dates		\$10.00	50
102	Armani Leather Purse		\$50.00	50

## • Manage Product Type

In **Types** menu you will get the product types and you can add, remove or modify product type from there.

Types			
		<input type="text"/> Type your query and press enter	<b>+ Add Types</b>
ID	Name	Icon	Actions
6	Furniture	🛋	
5	Clothing	👚	
4	Bags	💼	
3	Makeup	💄	
2	Bakery	🍰	
1	Grocery	🍎	

## • Manage Product Category

In **Categories** menu you will get the product types and you can add, remove or modify product categories from there.

Categories						
ID	Name	Details	Image	Icon	Slug	Actions
8	Women				women	
7	Watch				watch	
6	Sunglass				sunglass	
5	Sports				sports	
4	Sneakers				sneakers	
3	Men				men	
2	Kids				kids	
1	Bags				bags	

(1)

## • Product Management

In **Products** menu you will get the products and you can add, remove or modify products from there.

Products							
Image	Name	Brand	Product Type	Price/Unit	Quantity	Status	Actions
	Zara Monte Carlo	Fania Fashion	variable	\$90.00 - \$100.00	240		
	Zara Miss Chase	Vintgae	simple	\$100.00	120		
	White Oxford Shirt	Fusion	variable	\$35.00 - \$40.00	80		
	The Horse Original	Hoppister	simple	\$200.00	250		
	Roadster Women Round Neck	Hairstore VIntage	variable	\$150.00 - \$200.00	200		
	Reyban Havana Phantos Sunglasses	Hipster	simple	\$100.00	50		
	Philip Lim Leather Shoulder Bag	Phonix Bags	simple	\$260.00	100		
	Nike Pro Mesh Top with Leggins	Vintgae	variable	\$30.00 - \$35.00	100		
	Nike Comfy Vapor Maxpro	AB Shoes	variable	\$220.00 - \$250.00	2000		

## A portion of product form

Gallery  
Upload your product image gallery here

Brand & Categories  
Select product group and categories from here

Brand\*  
Fania Fashion

Categories  
Kids

Tags  
Gift Collection, Kids Collection

Description  
Edit your product description and necessary information from here

Name\*  
Zara Monte Carlo

## • Order Status

In **Order Status** menu you will get the order status list and you can add, remove or modify order status from there.

ID	Name	Serial	Actions
7	Delivered	10	
9	Failed To Contact Consignee	9	
8	Failed To collect payment	8	
10	Shipment Refused By Consignee	7	
6	Out For Delivery	6	
5	At Local Facility	5	
4	Order Dispatched	4	
3	Ready to Dispatch	3	
2	Order Processing	2	
1	Order Received	1	

## Order Management

In **Order** menu you will get the order list and you can add, remove or modify order from there.

Orders

Type your query and press enter

Tracking Number	Delivery Fee	Total	Order Date	Status	Shipping Address	Download	Actions
a1bKQKf4na1K	\$50.00	\$243.80	2 hours ago	Order Received	3886 Froe Street, Mill Creek, WV, 26280, United States	Download	🔗
0DGIGYdkMSw8	\$50.00	\$2,804.00	2 days ago	Order Received	f, fsdfs, fsdf, fsdf, fsdf	Download	🔗
4yQV4m38mbBe	\$50.00	\$845.60	3 days ago	Order Received	Cumilla, Cumilla, Cumilla, 1600, Cumilla	Download	🔗
iZndjRHpKCc3	\$50.00	\$1,136.30	5 days ago	Order Received	5/1-A, SBC Lane, Holiday Park, Gandaria, Dhaka, Dhupkhola maath, Pukur par, Dhaka, Dhaka, 1204, Bangladesh	Download	🔗
PGCb83dPpUc3	\$50.00	\$4,181.00	5 days ago	Order Received	5/1-A, SBC Lane, Holiday Park, Gandaria, Dhaka, Dhupkhola maath, Pukur par, Dhaka, Dhaka, 1204, Bangladesh	Download	🔗

< 1 >

## A portion of order management. Order status change.

Order ID - PGCb83dPpUc3

Select...

Change Status

Order Received

Order Processing

Ready to Dispatch

Order Dispatched

At Local Facility

Out For Delivery

5 Local Facility

Total

Sub total \$4,050.00

Tax \$81.00

Delivery fee \$50.00

Discount \$0.00

**Total** \$4,181.00

Billing Address

Shipping Address

## Coupon Management

In **Coupon** menu you will get the Coupon list and you can add, remove or modify Coupon from there.

Coupons

Type your query and press enter

+ Add Coupon

ID	Banner	Code	Amount	Active	Expired	Action
10	20OFF	20OFF	\$20.00	22 days ago	22 days ago	trash edit
9	CHRISTMAS18	CHRISTMAS18	\$18.00	22 days ago	in a month	trash edit
8	SUMMER15	SUMMER15	\$15.00	22 days ago	in 3 months	trash edit
7	12OFF	12OFF	\$12.00	22 days ago	in 14 days	trash edit
6	WINTER10	WINTER10	\$10.00	22 days ago	in 2 months	trash edit
5	SUMMER8	SUMMER8	\$8.00	22 days ago	in a month	trash edit
4	6OFF	6OFF	\$6.00	22 days ago	in 2 months	trash edit
3	RAMADANS	RAMADANS	\$5.00	22 days ago	in a month	trash edit
2	4OFF	4OFF	\$4.00	22 days ago	in a month	trash edit

# Customer Management

In **Customer** menu you will get the Customer list and you can add, remove or modify Customer from there.

The screenshot shows a list of customers with columns for Avatar, Name, Email, Status, and Actions. A modal window titled "Block Customer" is open over the list, containing a trash icon and the text "Are you sure you want to block this customer?". At the bottom are "Cancel" and "Block" buttons. Red arrows point to the "Block" button and the "Actions" column header.

# Tax Management

In **Tax** menu you will get the Tax list and you can add, remove or modify Tax from there.

The screenshot shows a list of taxes with columns for ID, Name, Rate (%), Country, City, State, ZIP, and Actions. A search bar and a "+ Add Tax" button are at the top. Red arrows point to the "Rate (%)" column header, the "Actions" column header, and the edit and delete icons in the Actions column.

# Shipping Management

In **shipping** menu you will get the shipping list and you can add, remove or modify shipping from there.

The screenshot shows a list of shippings with columns for ID, Name, Amount, Global, Shipping Type, and Actions. A search bar and a "+ Add Shipping" button are at the top. Red arrows point to the "Name" column header, the "Amount" column header, the "Actions" column header, and the edit and delete icons in the Actions column.

# Shipping Management

In `shipping` menu you will get the shipping list and you can add, remove or modify shipping from there.

The screenshot shows a table titled "Shipments" with columns: ID, Name, Amount, Global, Shipping Type, and Actions. A single row is visible with ID 1, Name "Global", Amount 50, Global checked, Shipping Type "fixed", and Actions (trash and edit icons). To the right of the table is a search bar with placeholder text "Type your query and press enter" and a button "+ Add Shipping". Red arrows point to the "Name" column header, the "Amount" column header, the search bar, the "Actions" column header, and the "+ Add Shipping" button.

# Settings Management

In `settings` menu you will get the settings management form there.

The screenshot shows two forms. The top form is for "Logo" with a placeholder image showing the word "CHAWK" and a file upload area. A red arrow points to the logo image. The bottom form is for "Information" with fields for Site Title ("ChawkBazar"), Site Subtitle ("Your next ecommerce"), Currency ("US Dollar"), Minimum Order Amount ("0"), Tax Class ("Global"), and Shipping Class ("Global"). Three red arrows point to the "Site Title", "Tax Class", and "Shipping Class" input fields.

# Available Scripts:

You can run below commands in the root folder for your need.

```
"clean": "yarn workspaces run rimraf \"\n{.next,node_modules,__generated__,.cache,src/graphql/*.d.ts,src/framework/graphql\n/**/*.d.ts}\\" && rimraf node_modules",\n"dev:shop-rest": "yarn workspace @chawkbazar/shop dev",\n"build:shop-rest": "yarn workspace @chawkbazar/shop build",\n"start:shop-rest": "yarn workspace @chawkbazar/shop start",\n"dev:admin-rest": "yarn workspace @marvel/admin-rest dev",\n"build:admin-rest": "yarn workspace @marvel/admin-rest build",\n"start:admin-rest": "yarn workspace @marvel/admin-rest start",\n"prepare": "husky install",
```

\*\* Note: \*\* Also, individual Scripts are available under every individual package.  
You can check out them from their individual package.json file.

## Customization

### For customizing the template's default site settings:

> [your-frontend-project] = admin [ rest ] or shop

If you want to customize the site settings such as site info, default(logo, language , currency code),navigation etc, you can easily change those settings from

```
[your-frontend-project]/src/settings/site.settings.ts
```

In this file, find your required settings key and change it's value according to your need.

For example, If you want to change the currencyCode : please go to [your-frontend-project]/src/settings/site.settings.ts and find the following portions and change it.

```
currencyCode: "USD";
```

NOTE \*\* Some of these options are customizable through ADMIN Dashboard.

## Styles and Assets

### CSS styles:

[your-frontend-project] = admin [ rest ] or shop

We use tailwindcss framework with some customization which you find at :

```
open [your-frontend-project]/tailwind.config.js
```

For tailwindcss documentation:

Go to [Tailwindcss](#)

# Icons:

for our icons

```
open [your-frontend-project]/src/components/icons
```

## For Adding a custom Icon:

To add a custom icon please follow this procedure.

1. Open your custom SVG icon file in the code editor and copy all the code.
2. Then Go to src -> components -> icons folder and create a new .tsx file.
3. Then here create a function component and paste your copied SVG code inside the return statement.
4. Then convert all the SVG's kebab-cases properties into camelCase format except the data-name property. For ex. change the stroke-width and fill-color into strokeWidth and fillColor. (for reference you can see one of our icon.)
5. If your custom SVG code has any single custom color then change them into fillColor.

# Menu

Please follow this video with the documentation, and it'll make the menu build process relatively easy.

[https://www.youtube.com/watch?v=ewNGG\\_i3r-Y&feature=emb\\_title](https://www.youtube.com/watch?v=ewNGG_i3r-Y&feature=emb_title)

Mega menu for the `shop` are coming from a JSON file which is located at,

```
shop -> public -> data -> menus.json
```

The screenshot shows a code editor with the following details:

- File Explorer:** Shows the project structure: `CHAWKBAZAR-LARAVEL` with subfolders like `husky`, `idea`, `admin`, `chawkbazar-api`, `node_modules`, `shop` (highlighted with a red arrow), `sro` (highlighted with a red arrow), and `data/static`. Inside `data/static`, there are files like `banners.ts`, `collection.ts`, `exclusive-block.ts`, and `feature-block.ts`. The file `menus.ts` is also listed in this folder and is highlighted with a red arrow.
- Code Editor:** The file `menus.ts` is open. It contains the following TypeScript code:

```
You, 22 minutes ago | 1 author (You)
You, 2 days ago • Restructure menu data and menu components
1 export const menu = [
2   {
3     id: 1,
4     path: "/",
5     label: "menu-demos",
6     subMenu: [
7       {
8         id: 1,
9         path: "/",
10        label: "menu-modern"
11      },
12      {
13        id: 2,
14        path: "/standard",
15        label: "menu-standard"
16      },
17      {
18        id: 3,
19        path: "/minimal",
20        label: "menu-minimal"
21      },
22      {
23        id: 4,
24        path: "/vintage",
25        label: "menu-vintage"
26      },
27      {
28        id: 5,
29        path: "/classic",
30        label: "menu-classic"
31      }
32 ]
```

A large red arrow points from the bottom right towards the code editor area.

You've to update this JSON to build the mega menu.

## Data Structure

```
id -> ID for menu or submenu  
path -> After clicking the menu in frontend which path will be redirect  
label -> Label of that men  
subMenu -> JSON for submenu  
mobileMenu -> Different menu for mobile.
```

## Banner

Please follow this video with the documentation, and it'll make the banner update process relatively easy.

<https://www.youtube.com/watch?v=AMOexuHTKa0>

The banner data are coming from a JSON file which is located at,

```
shop -> public -> data -> banner.json
```

The screenshot shows the VS Code interface with the Explorer sidebar on the left and the code editor on the right. The Explorer sidebar shows a project structure under 'CHAWKBAZAR-LARAVEL' with several folders like 'husky', 'idea', 'admin', 'chawkbazar-api', 'node\_modules', 'public', 'src', 'assets', 'components', 'containers', 'contexts', and 'data'. Inside 'data', there is a 'static' folder containing files like 'banners.ts', 'collection.ts', 'exclusive-block.ts', 'feature-block.ts', and 'menus.ts'. The 'banners.ts' file is open in the editor, showing TypeScript code defining a 'masonryBanner' object with two entries (id: 1 and id: 2) each having 'title', 'slug', 'image', and 'mobile' and 'desktop' properties. A pink arrow points from the 'data' folder in the sidebar to the 'data' folder in the code editor's title bar. Another pink arrow points from the 'static' folder in the sidebar to the 'static' folder in the code editor's title bar.

```
You, 2 days ago | 1 author (You)  
1 // Common data for all demo  
2 export const masonryBanner = [  
3   {  
4     id: 1,  
5     title: "Men's Collection",  
6     slug: "mens-collection",  
7     image: {  
8       mobile: {  
9         url: "/assets/images/banner/masonry/banner-mobile-1.jpg",  
10        width: 470,  
11        height: 232,  
12      },  
13      desktop: {  
14        url: "/assets/images/banner/masonry/banner-1.jpg",  
15        width: 1078,  
16        height: 425,  
17      },  
18    },  
19    You, 2 days ago · Restructure banner data and banner components  
20    type: "medium",  
21  },  
22  {  
23    id: 2,  
24    title: "New Sports",  
25    slug: "new-sports",  
26    image: {  
27      mobile: {  
28        url: "/assets/images/banner/masonry/banner-mobile-2.jpg",  
29        width: 232,  
30        height: 232,  
31      },  
32      desktop: {  
33        url: "/assets/images/banner/masonry/banner-2.jpg",  
34      },  
35    },  
36  },  
37];
```

You've to update this JSON to build the mega menu.

## Data Structure

```
id -> ID for a banner  
title -> Title for a banner  
slug -> This is a tag slug
```

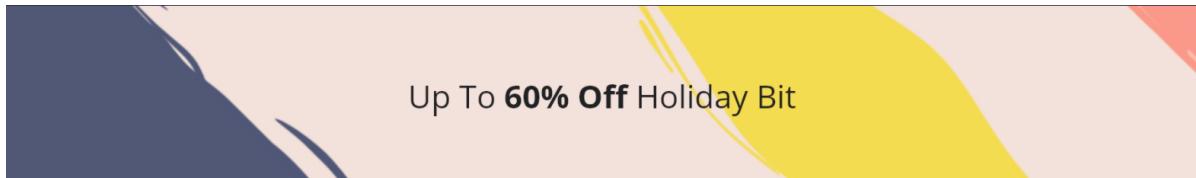
At first, create the tag and add that tag slug on the `slug` path.

For example, if you want to start a special campaign for some products, just create a tag for that campaign in admin, add that tag to their related products, and on `banners.json`, add that tag as a slug.

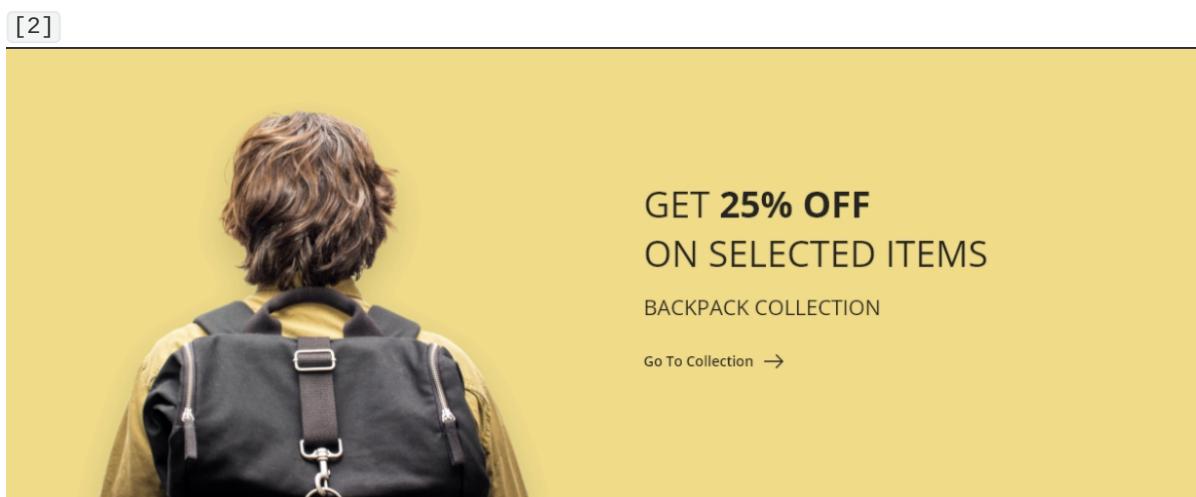
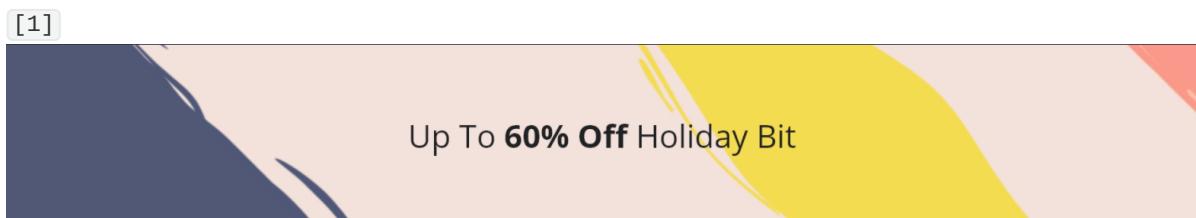
```
image -> Image path
mobile -> JSON object for mobile image
desktop -> JSON object for desktop image
url -> Path of the image
width -> Width of the image for next/image optimization
height -> Height of the image for next/image optimization
```

## Data Mapping

### homeOneBanner

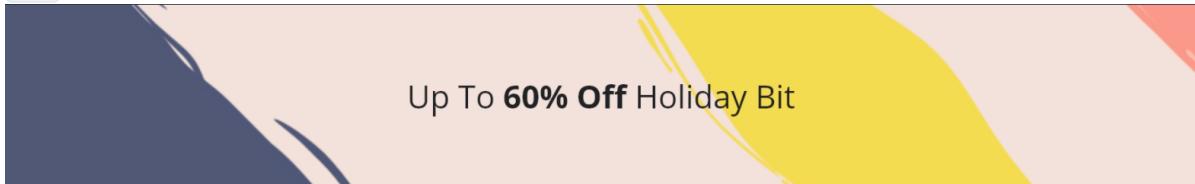


### homeOneHeroBanner

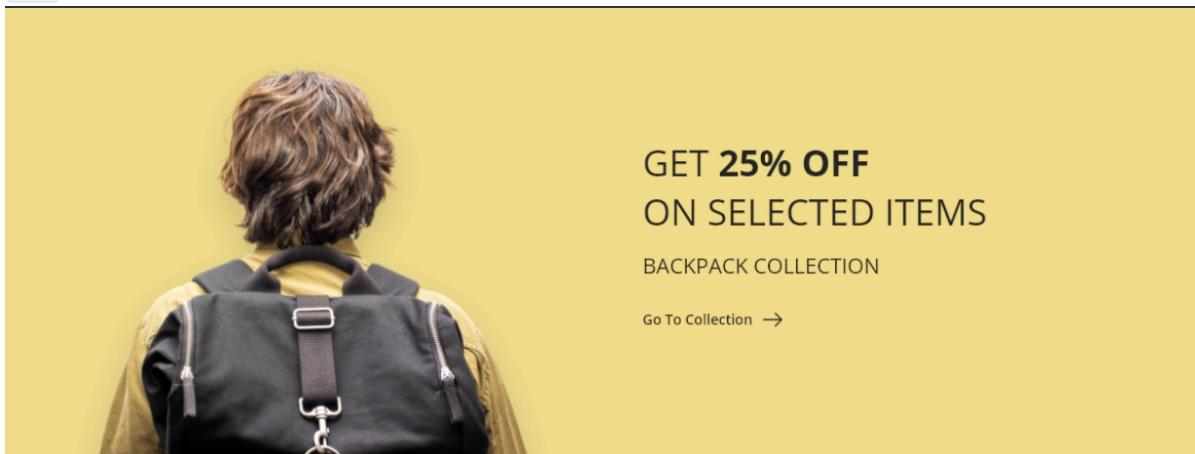


## homeTwoHeroBanner

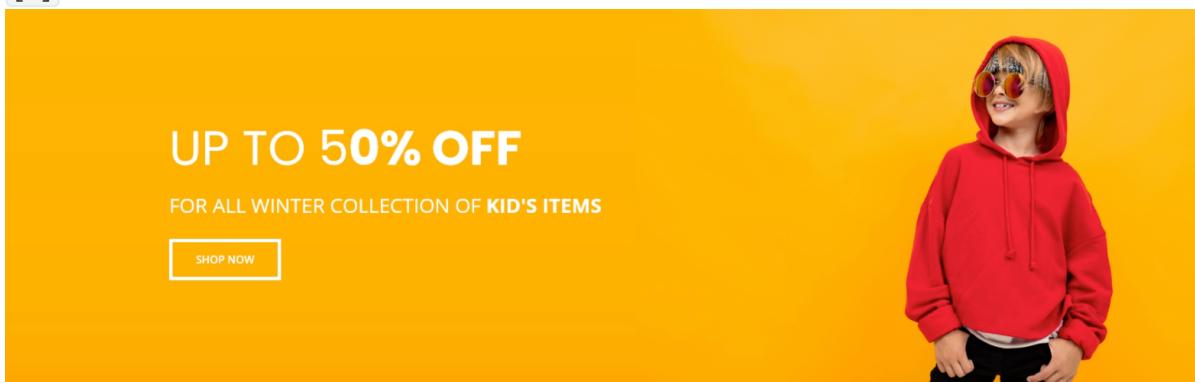
[1]



[2]

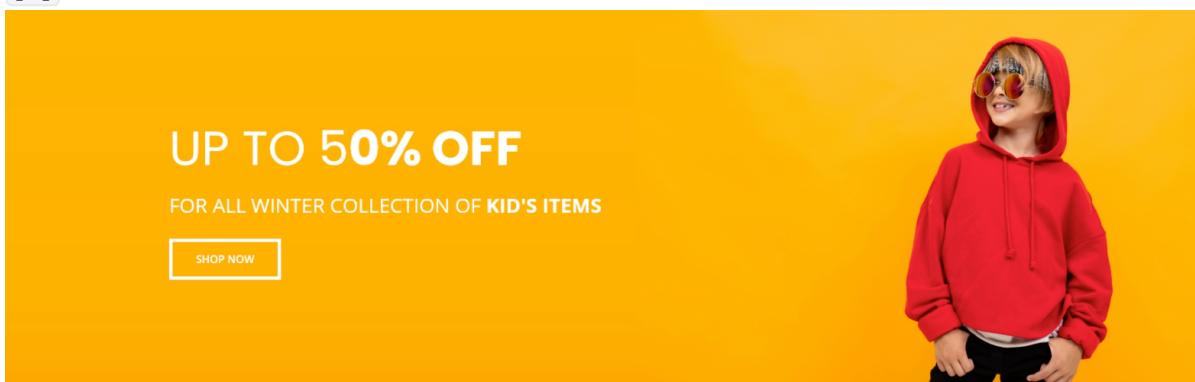


[3]



## homeThreeBanner

[1]

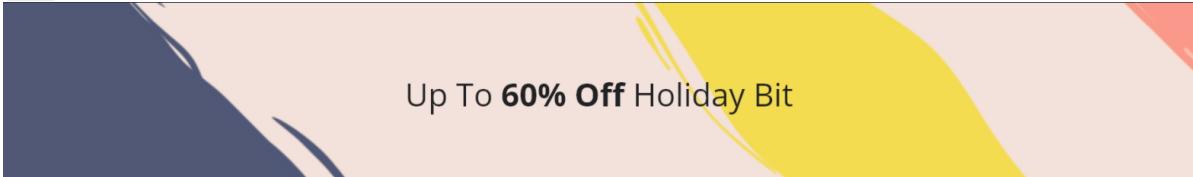


[2]

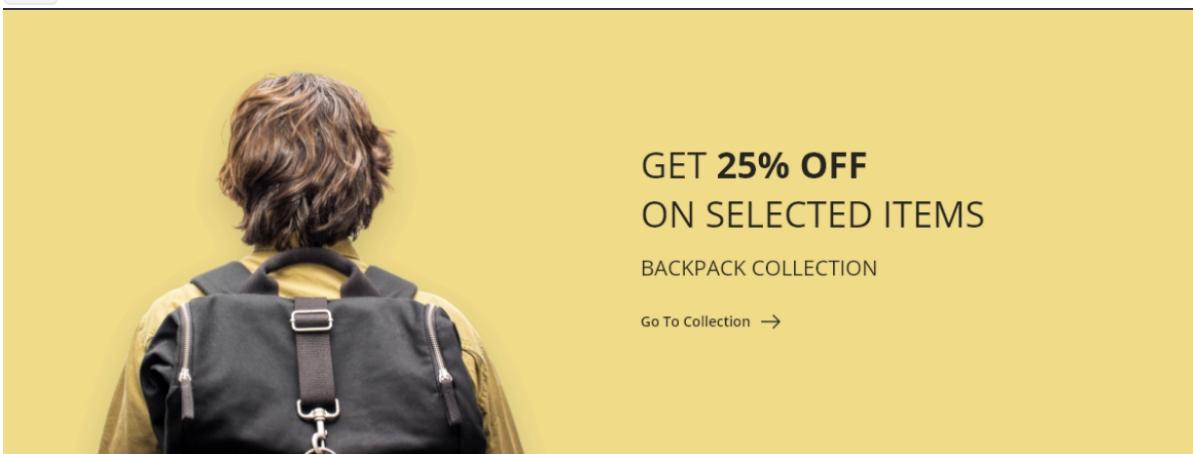


## homeThreeMasonryBanner

[1]



[2]



[3]



[4]

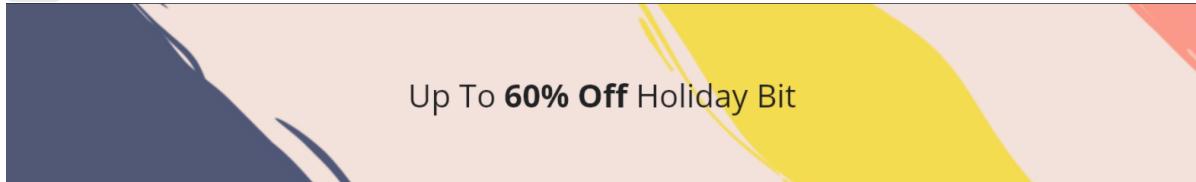


[5]

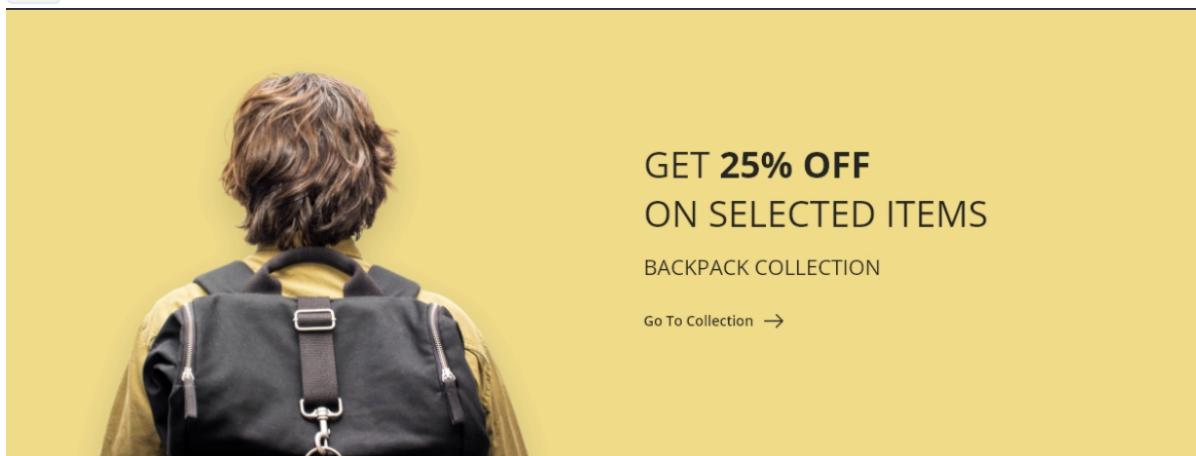


## bannerGrid

[1]



[2]

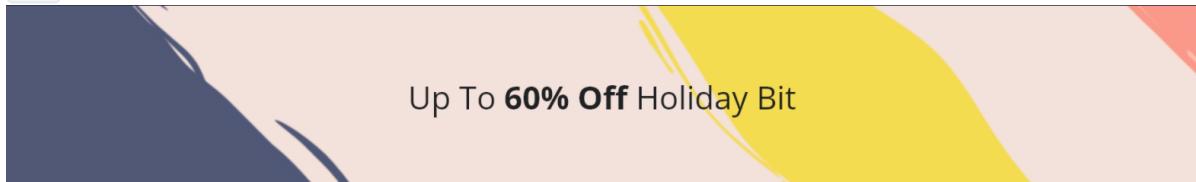


[3]

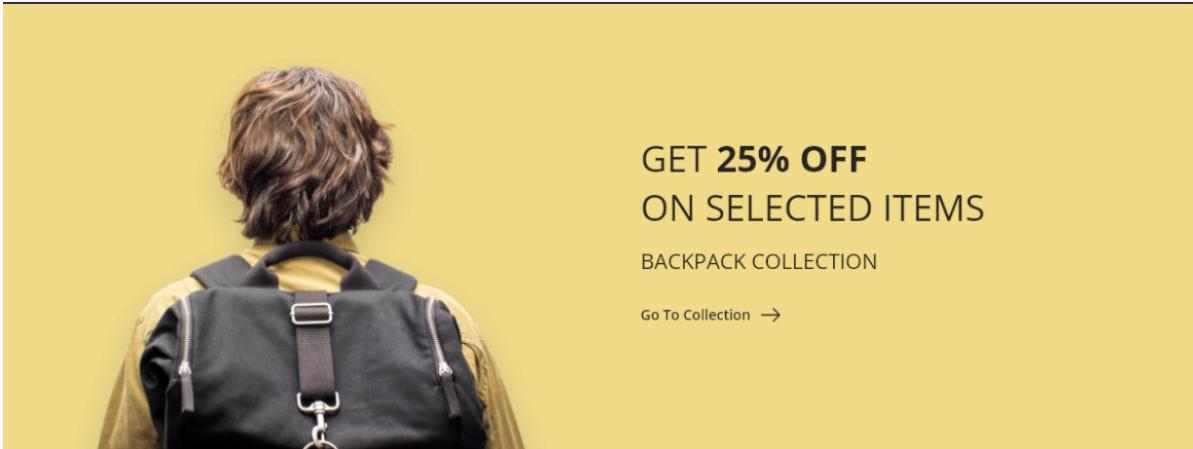


## promotionBanner

[1]



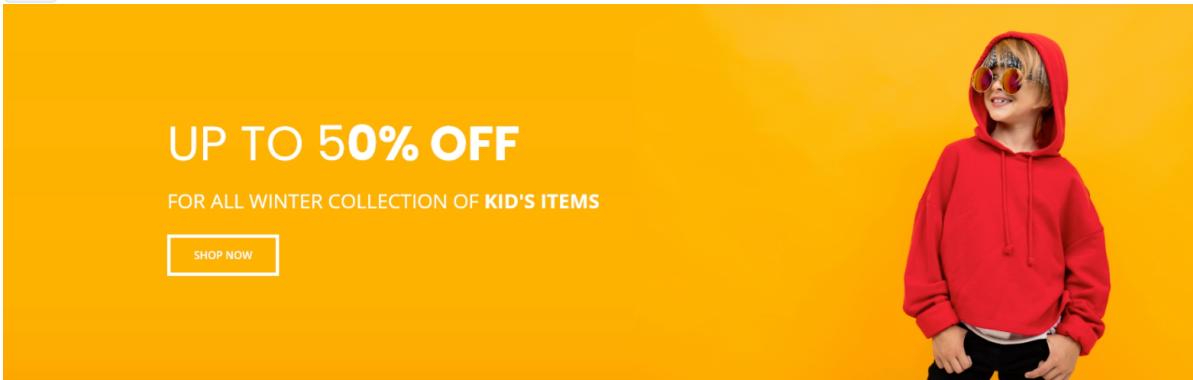
[2]



GET 25% OFF  
ON SELECTED ITEMS  
BACKPACK COLLECTION

[Go To Collection →](#)

[3]

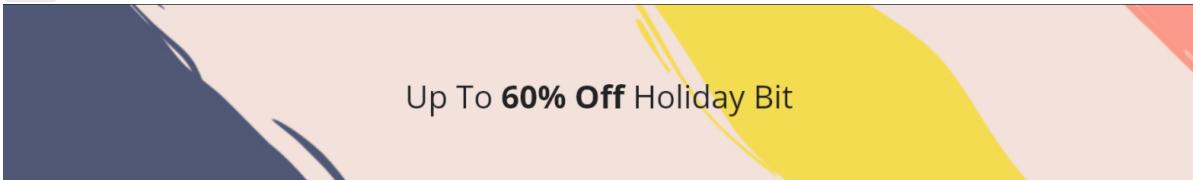


UP TO 50% OFF  
FOR ALL WINTER COLLECTION OF KID'S ITEMS

[SHOP NOW](#)

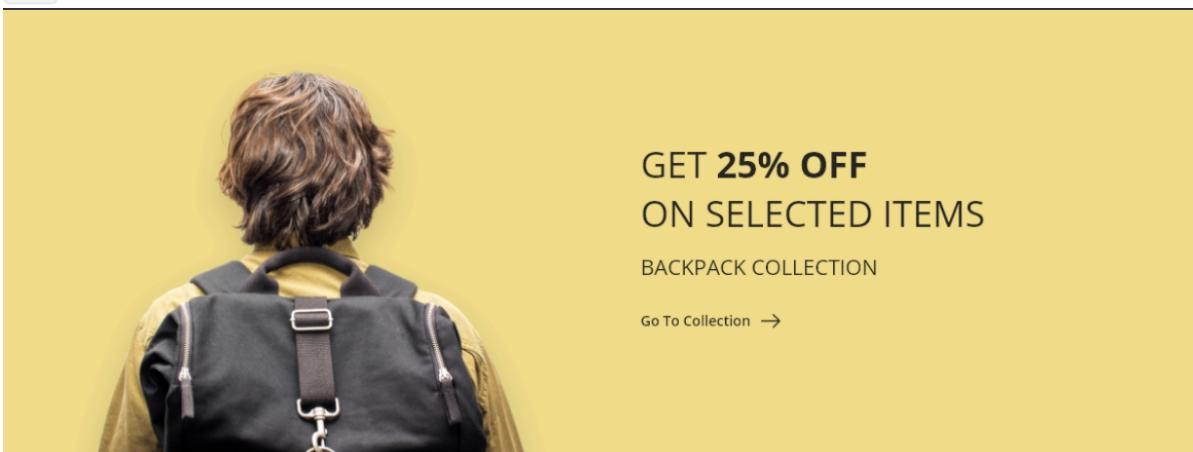
## promotionBannerTwo

[1]



Up To 60% Off Holiday Bit

[2]



GET 25% OFF  
ON SELECTED ITEMS  
BACKPACK COLLECTION

[Go To Collection →](#)

[3]



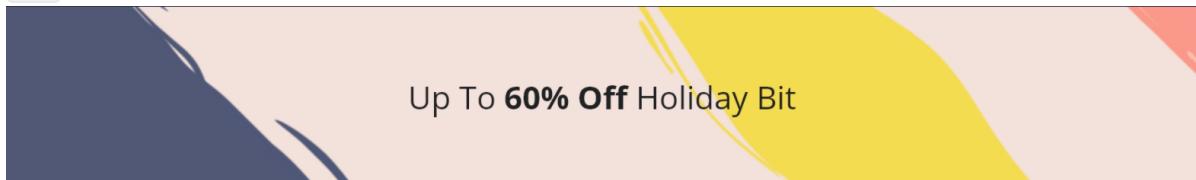
**UP TO 50% OFF**

FOR ALL WINTER COLLECTION OF KID'S ITEMS

[SHOP NOW](#)

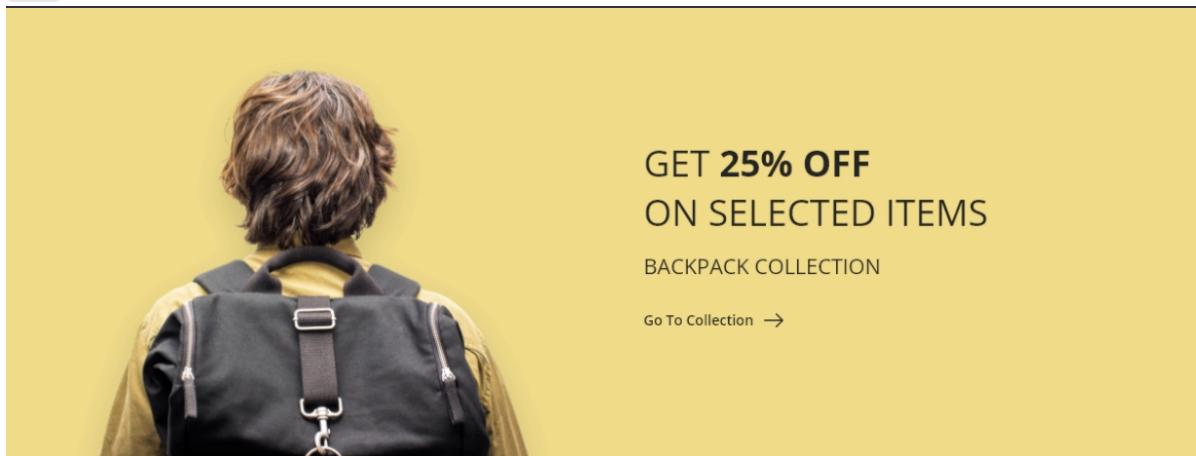
**bannerDataOne**

[1]



Up To **60% Off** Holiday Bit

[2]



**GET 25% OFF  
ON SELECTED ITEMS**

BACKPACK COLLECTION

[Go To Collection →](#)

[3]



**UP TO 50% OFF**

FOR ALL WINTER COLLECTION OF KID'S ITEMS

[SHOP NOW](#)

## bannerDataTwo

[1]



[2]



[3]

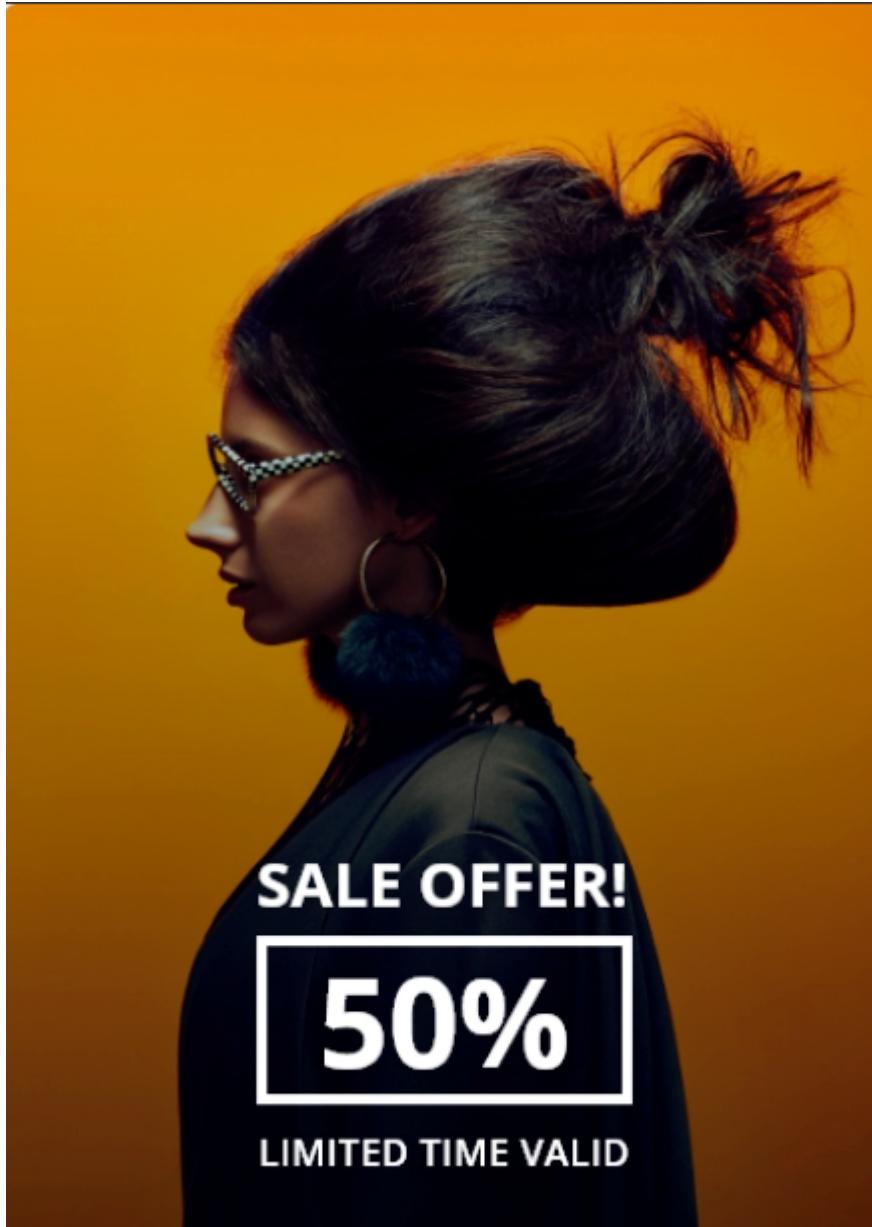


### homeThreeProductsBanner

[1]



[2]



homeFourBanner

[1]



[2]

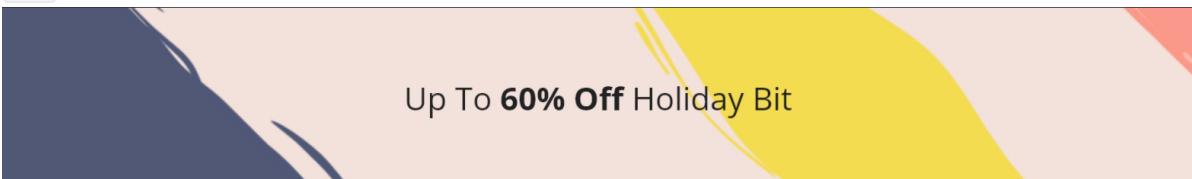


[3]

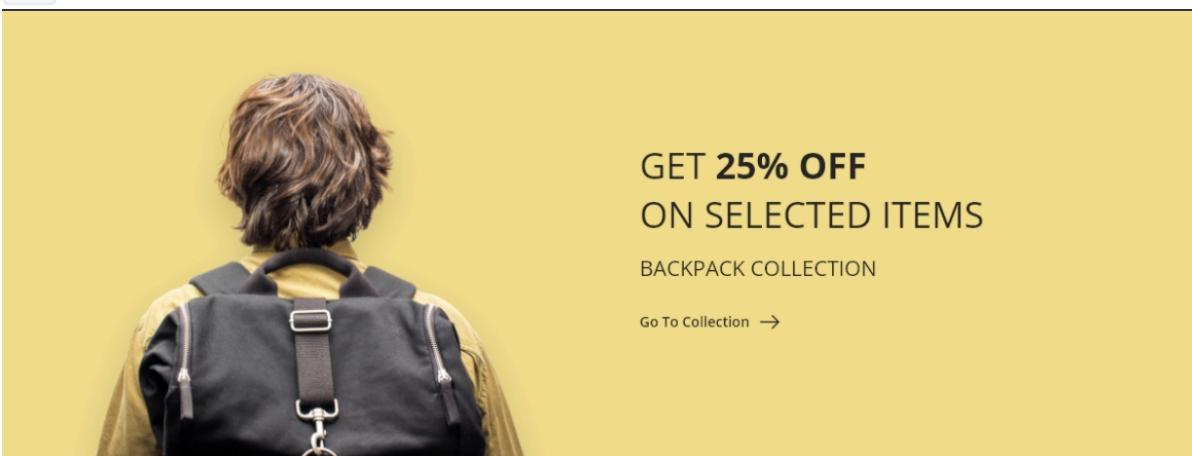


## homeFourGridBanners

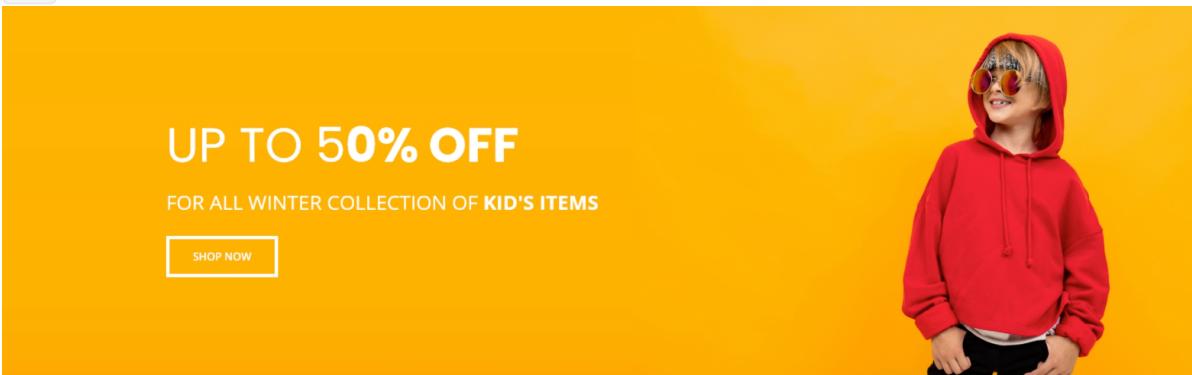
[1]



[2]



[3]



## homeFiveBanner



## Brand

We have multiple types of brand layouts.

1. Slider Layout -> For Minimal Demo
2. Grid Layout -> For Others Demo

For slider layout, you've to upload a single image, and for grid layout, you've to upload two images; one is a cover, and another is a logo.

You'll get a layout option when you create a brand,

CHAWK

Dashboard

Shops

My Shops

Products

Attributes

Brands

Categories

Tags

Orders

Order Status

Users

Coupons

Taxes

Shipments

Withdraws

Settings

Description  
Add type-description-help-text

Brand Images  
Add your brand images for different layout

Name  
CHAWK

Select Icon

Select...

Select layout type

Grid Layout

Grid Layout

Slider Layout

Upload an image or drag and drop  
PNG, JPG

Add Image

Remove

Add Brand

If you use a minimal home page, then add a slider layout; otherwise, use Grid Layout.

## Utilities

In this template, We have used some custom helper functions which is located in

```
[your-frontend-project] = admin [rest] or shop
```

```
cd [your-frontend-project]/src/utils/
```

You can use or customize these helper fuctions according to your needs.

# Backend Integration

## API Integration:

For API Integration:

```
> [your-frontend-project] = admin [rest] or shop
```

We have used env variables using `.env` file format. You have to put your API url within this file.

For example:

Put that url in the `shop/.env` and `admin/rest/.env`

```
NEXT_PUBLIC_REST_API_ENDPOINT= '{put_your_api_url_here}'
```

And run your frontend project

## Data Fetching

```
[your-frontend-project] = admin [rest] or shop
```

We have used React Query and Axios which is defined in

```
[your-frontend-rest-project]/src/data/
```

For details documentation:

- [React-Query](#)

You can check and customize it as your need.

## Uses Example:

- Query

```
import { useProductsQuery } from "@data/product/use-products.query";

const {
  isFetching: loading,
  isFetchingNextPage: loadingMore,
  fetchNextPage,
  hasNextPage,
  isError,
  data,
  error,
} = useProductsQuery({
  type: "grocery",
  text: "",
  category: "grocery",
```

```
});
```

- Mutation

```
import { useCreateProductMutation } from "@data/product/product-create.mutation";

const {
  mutate: createProduct,
  isLoading: creating,
} = useCreateProductMutation();

function handleSubmit(inputValues) {
  createProduct(
    {
      ...inputValues,
    },
    {
      onError: (error: any) => {
        Object.keys(error?.response?.data).forEach((field: any) => {
          setError(field, {
            type: "manual",
            message: error?.response?.data[field][0],
          });
        });
      },
    },
  );
}
```

# Deployment

## AWS (Amazon Web Service)

If you want to use all the scripts (`shop`, `admin`, `api`) on the same server as this tutorial, then we recommend creating a blank ubuntu-based server with at least 2+ CPU cores and 2GB+ memory.

### How to create ec2 server?

In this AWS tutorial, we're going to create an ec2 server. To do that at first, login to your AWS account and then click,

```
ec2 -> Instance -> Launch Instance
```

# AWS Management Console

The screenshot shows the AWS Management Console homepage. On the left, the navigation menu is open, with a red arrow pointing to the 'Compute' section under 'All services'. The 'Instances' option is highlighted with a red arrow. The main content area shows the 'Instances (1)' page for EC2. A red arrow points to the 'Launch instances' button at the top right of the table. The table lists one instance: i-00fb9dda581f15817, which is terminated.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
-	i-00fb9dda581f15817	Terminated	t2.micro	-	1 alarm	us-east-1e	-

Then select a ubuntu 20.04 server

After that, click **Next** -> **Next** -> **Next** -> **Next**

And on security pages, add a rule for **HTTP**, **HTTPS** and **SSH**,

Our automation scripts setup **HTTPS** in your domain so you should open **HTTPS**

The screenshot shows the AWS Security Groups page. It displays a table of security group rules:

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom ::/0	e.g. SSH for Admin Desktop
SSH	TCP	22	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Custom ::/0	e.g. SSH for Admin Desktop

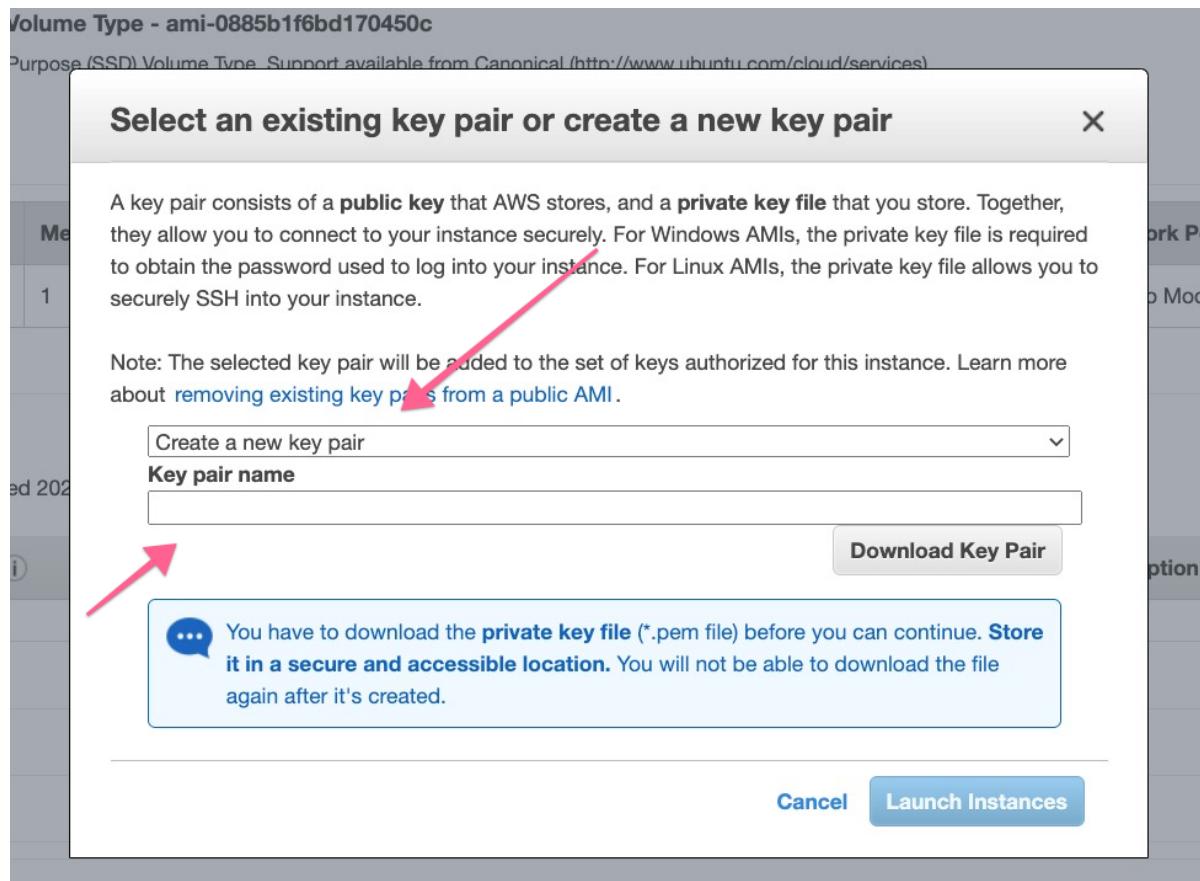
**Add Rule**

NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.

**Cancel** **Save**

After review, click **Launch**, and you'll get and popup for KeyPair, which will be required to login to the server using ssh.

If you already have a previous KeyPair, you can use that; otherwise, you can create a new one. After completing that, make sure you download that KeyPair.



After launching the instance, you'll get the server IP, which will be required to login into ssh.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
-	i-00fb9dda581f15817	Terminated	t2.micro	-	2 alarms	us-east-1e	-
<input checked="" type="checkbox"/> pickbazar	i-0b3ad96ed56b60940	Running	t2.micro	-	No alarms	us-east-1d	ec2-54-90-111-167

Instance: i-0b3ad96ed56b60940 (pickbazar)

Public IPv4 address: 54.90.111.167

## Domain Setup

Now copy the server IP and connect it with your domain name.

Instances (1/2) [Info](#)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
-	i-00fb9dda581f15817	<span>Terminated</span>	t2.micro	-	<span>2 alarms</span>	us-east-1e	-
<b>pickbazar</b>	i-0b3ad96ed56b60940	<span>Running</span>	t2.micro	-	No alarms	us-east-1d	ec2-54-90-111-167

**Instance: i-0b3ad96ed56b60940 (pickbazar)**

Details Security Networking Storage Status Checks Monitoring Tags

Instance summary [Info](#)

Instance ID i-0b3ad96ed56b60940 (pickbazar)	Public IPv4 address 54.90.161.167, <a href="#">open address</a>	Private IPv4 addresses
--	--	------------------------

© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)

Please contact your domain provider for detailed explanation of how to do that.

## Login to Server

At first, login to your AWS server using ssh. to do that, go to the folder from the terminal where KeyPair is downloaded.

then click Connect

Instances (1/2) [Info](#)

[Connect](#) Instance state Actions Launch instances

From the Connect dashboard, go to SSH Client and copy the example line and paste it to your terminal.

EC2 > Instances > i-0b3ad96ed56b60940 > Connect to instance

Connect to instance [Info](#)

Connect to your instance i-0b3ad96ed56b60940 (pickbazar) using any of these options

EC2 Instance Connect Session Manager **SSH client**

Instance ID: i-0b3ad96ed56b60940 (pickbazar)

- Open an SSH client.
- Locate your private key file. The key used to launch this instance is tareqmahmud.pem
- Run this command, if necessary, to ensure your key is not publicly viewable.  
`chmod 400 tareqmahmud.pem`
- Connect to your instance using its Public DNS:  
`compute-1.amazonaws.com`

Example:  
`ssh -i "tar...d.pem" ubuntu@compute-1.amazonaws.com`

**Note:** In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

Cancel

With this command, you will successfully connect to your server through ssh.

## Change permission .pem

You've to change the permission of the downloaded .pem file to 400 to access the server. To do that, at first go to the location where .pem store then run,

```
chmod 400 chawkbazar.pem
```

Change the `chawkbazar.pem` filename if you use a different name during generate the key.

Now go to the `VPS Server` section for deploy the `Chawkbazar Laravel`

## Virtual Private Server (Automated Script)

If you want to deploy the app using automated script then [follow this](#)

With this tutorial, you can install ChawkBazar to any type of blank or empty ubuntu server. For example, `Digital Ocean Droplets, Amazon Lightsail, AWS, Google Cloud Virtual Private Server, Azure Ubuntu Virtual Private Server`, etc.

If you want to use all the scripts (`shop`, `admin`, `api`) on the same server as this tutorial, then we recommend creating a blank ubuntu-based (`v20.0.4 lts`) server with at least 2+ CPU cores and 2GB+ memory.

Please connect your `domain` with `server`. We don't recommend/support deployment the project via `IP`.

Please follow this video with the documentation, and it'll make the installation process relatively easy.

## Prerequisite

This automated script is for the \*nix system. So if you are using mac or Linux, then you're good to go. But if you are using windows, then install WSL on your computer and use this script using [WSL](#) or follow this [manual installation](#)

Before starting, the procedure ensures that NodeJS 16 (the latest) is installed on your computer.

```
npm i -g yarn zx
```

Now you can follow the script installation procedure,

## At first login your server from terminal

```
ssh SERVER_USERNAME@SERVERIP
```

Make sure that you are logged in your server then follow the next step and run suggested command.

## Upload api and deployment project to Virtual Server form your PC - RUN on Local PC

To upload the zipped `chawkbazar-api` and `deployment` files to server you need to run the below command form your chawkbazar project root

while running below command you will be asked to enter your server `username` and `ip address` by entering and a successful connection you will also be asked to enter your `chawkbazar-api.zip` and `deployment.zip` files path and the path will look like `/home/your_project_folder_path/chawkbazar-laravel/chawkbazar-api.zip` for `chawkbazar-api.zip` file so forth for `deployment.zip`

```
bash deployment/deployment.sh
```

Then login your server using `ssh` and,

## Server Environment setup script - RUN on Virtual Server

```
bash /var/www/chawkbazar-laravel/deployment/nodesetup.sh
```

## Nginx Setup And Settings - RUN on Virtual Server

```
zx /var/www/chawkbazar-laravel/deployment/setenv.mjs
```

## Backend build - RUN on Virtual Server

```
sudo zx /var/www/chawkbazar-laravel/deployment/backendbuildscript.mjs
```

## Frontend build script - RUN on Local PC

Run the below command from your chawkbazar project root

```
zx deployment/frontendbuildscript.mjs
```

## Frontend run script - RUN on Virtual Server

```
zx /var/www/chawkbazar-laravel/deployment/frontendlaunchscript.mjs
```

## Vercel

[vercel.com](https://vercel.com)

If you want to host the template in vercel.com then follow the below command

## API

It's not possible to host the API to vercel. Vercel doesn't support laravel API deployment. So you've to host the API on a separate server. We suggest you create a VPS server and host the API there. For more details, follow this [VPS Deployment](#) docs.

After the host, you'll get your API URL.

## Frontend

Now for frontend add API URL and other necessary config details to,

```
shop -> vercel.json
```

```
admin -> rest -> vercel.json
```

after that, install vercel-cli on your computer using this command,

```
npm i -g vercel-cli
```

After that, log in to vercel using this command,

```
vercel login
```

Then go to the `shop` directory and use this command to deploy,

```
vercel
```

Similarly, go to the `admin -> rest` directory and use this command to deploy,

```
vercel
```

For more details,

Please follow [nextjs deployment docs](#):

## Virtual Private Server (Manual)

With this tutorial, you can install ChawkBazar to any type of blank or empty ubuntu server. For example, `Digital Ocean Droplets, Amazon Lightsail, AWS, Google Cloud Virtual Private Server, Azure Ubuntu Virtual Private Server`, etc.

If you want to use all the scripts (`shop`, `admin`, `api`) on the same server as this tutorial, then we recommend creating a blank ubuntu-based server with at least 2+ CPU cores and 2GB+ memory.

Please follow this video with the documentation, and it'll make the installation process relatively easy.

<https://www.youtube.com/watch?v=cESkbAmmDPc>

## Access Server

At first login your server using `SSH` and `Terminal`

## Install NodeJS & Required Application

### Install NodeJS

At first, we've to install NodeJS and npm to run the chawkbazar app. To install NodeJS and npm, run this command on your terminal,

```
sudo apt-get update
curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash -
sudo apt-get install -y nodejs
```

### Install Yarn

Chawkbazar is highly dependent on `yarn`, it would be best to handle all the script parts using `yarn`. So to install yarn, use this command,

```
sudo npm i -g yarn
```

If you face any permission issue, then please check this official doc to resolve that,

[Npm Permission Issue](#)

### Install Zip & Unzip

```
sudo apt install zip unzip
```

### Install PM2

Now we will install `PM2`, which is a process manager for Node.js applications. `PM2` provides an easy way to manage and daemonize applications (run them in the background as a service). To install `PM2` use this command,

```
sudo npm install -g pm2
```

After restarting the server or if the server crash, then pm2 will halt the process. To prevent that, we'll add pm2 as a `startup` process to run automatically after restart the server.

```
pm2 startup systemd
```

# Setup Server

## Introduction

Nginx is one of the most popular web servers in the world. In this deployment tutorial, we're going to use Nginx to host our website. In this tutorial, we're going to use ubuntu 20.04 to host chawkbazar

## Step 1 - Installing Nginx

After creating the server, make sure the apt library is up to date. To update the apt library, use this command,

```
sudo apt update
```

After the update apt, we're going to install Nginx. To do that, use this command

```
sudo apt install nginx
```

## Step 2: Adjusting the Firewall

Before testing Nginx, the firewall software needs to be adjusted to allow access to the service. Nginx registers itself as a service with `ufw` upon installation, making it straightforward to allow Nginx access.

To check the `ufw` list, use this command,

```
sudo ufw app list
```

You will get a listing of an application list like this,

```
Available applications:
  Nginx Full
  Nginx HTTP
  Nginx HTTPS
  OpenSSH
```

At first, add ssh to the firewall,

```
sudo ufw allow ssh
sudo ufw allow OpenSSH
```

After that, to enable Nginx on the firewall, use this command,

```
sudo ufw allow 'Nginx HTTP'
```

Now enable the firewall,

```
sudo ufw enable  
sudo ufw default deny
```

You can verify the change by typing:

```
sudo ufw status
```

The output will be indicated which HTTP traffic is allowed:

To	Action	From
--	-----	-----
Nginx HTTP 22/tcp	ALLOW	Anywhere
Nginx HTTP (v6) 22/tcp (v6)	ALLOW	Anywhere (v6)
	ALLOW	Anywhere (v6)

## Step 3 - Checking your Web Server

Now check the status of the Nginx web server by using this command,

```
systemctl status nginx
```

You'll get an output like this,

```
● nginx.service - A high performance web server and a reverse proxy server  
  Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)  
  Active: active (running) since Tue 2021-01-12 07:35:20 UTC; 7min ago  
    Docs: man:nginx(8)  
   Main PID: 2940 (nginx)  
     Tasks: 2 (limit: 1164)  
    Memory: 5.1M  
      CGrou... /system.slice/nginx.service  
          └─2940 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;  
              ├─2941 nginx: worker process
```

## Step 4 - Install MySQL

```
sudo apt install mysql-server
```

## Step 5 - Install PHP

```
sudo apt install php-fpm php-mysql  
sudo apt install php-fpm php-mysql php-mbstring php-xml php-bcmath php-simplexml  
php-mbstring php7.4-gd php7.4-curl composer
```

## Step 6 - Create MySQL Database & User

```
sudo mysql
CREATE DATABASE chawkbazar;

CREATE USER 'chawkbazar_user'@'%' IDENTIFIED WITH mysql_native_password BY
'chawkbazar1';

GRANT ALL ON chawkbazar.* TO 'chawkbazar_user'@'%';

FLUSH PRIVILEGES;
```

We use MySQL user name `chawkbazar_user` and MySQL password `chawkbazar1`. Make sure you change at least `MySQL` password for security.

## Step 7 - Change permission for the `www` folder

```
sudo chown -R $USER:$USER /var/www/
```

## Step 8 - Upload API to Server

At first, use this command to create a directory on `/var/www/`

```
mkdir /var/www/chawkbazar
```

Then, go to your `local computer`

1. Extract the `chawkbazar` package that you download from `CodeCanyon`.
2. On that folder, you'll get another `folder` called `chawkbazar-laravel`.
3. On that folder, you'll get a folder called `chawkbazar-api`

Now upload this `chawkbazar-api` folder to the server `/var/www/chawkbazar/`

## Step 9: Setting Up Server & Project

In this chapter, we'll set up our server and also will set up Reverse Proxy to host all of our sites from the same server.

At first, we'll disable the default configuration.

```
sudo rm /etc/nginx/sites-enabled/default
```

## Step 10 - Create New Nginx for the domain

```
sudo touch /etc/nginx/sites-available/chawkbazar
sudo nano /etc/nginx/sites-available/chawkbazar
```

Add this Nginx config file to that edited file,

```
server {
    listen 80;
```

```

server_name YOUR_DOMAIN.com;

add_header X-Frame-Options "SAMEORIGIN";
add_header X-XSS-Protection "1; mode=block";
add_header X-Content-Type-Options "nosniff";

index index.html index.htm index.php;

charset utf-8;

# For API
location /api {
    alias /var/www/chawkbazar/chawkbazar-api/public;
    try_files $uri $uri/ @api;
    location ~ \.php$ {
        include fastcgi_params;
        fastcgi_param SCRIPT_FILENAME $request_filename;
        fastcgi_pass unix:/run/php/php7.4-fpm.sock;
    }
}

location @api {
    rewrite /api/(.*)$ /api/index.php?/$1 last;
}

# For FrontEnd -> Rest
location /{
    proxy_pass http://localhost:3003;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
}

location /admin{
    proxy_pass http://localhost:3002/admin;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
}

error_page 404 /index.php;

location ~ \.php$ {
    fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
    fastcgi_index index.php;
    fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
    include fastcgi_params;
}

location ~ /\.(?!well-known).* {
    deny all;
}

```

```
}
```

Make sure you change `YOUR_DOMAIN.com` to your specific `domain name`

You can change `api` path, if you want to change the domain path for the laravel application

You can change `admin` path, if you want to change the domain path for the frontend admin

Save and close the file by typing `CTRL` and `X`, then `Y` and `ENTER` when you are finished.

Then enable the config

```
sudo ln -s /etc/nginx/sites-available/chawkbazar /etc/nginx/sites-enabled/
```

Make sure you didn't introduce any syntax errors by typing:

```
sudo nginx -t
```

Next, restart Nginx:

```
sudo systemctl restart nginx
```

## Secure Server

### Step 1: Secure Nginx with Let's Encrypt

```
sudo apt install certbot python3-certbot-nginx
sudo ufw status

sudo ufw allow 'Nginx Full'
sudo ufw delete allow 'Nginx HTTP'

sudo ufw status
sudo certbot --nginx -d YOUR_DOMAIN
```

After this command, you'll get several command prompt. Make sure you take the necessary steps and provide information on that command prompt.

## Install API

### Step 1: Build and Run `api`

At first, go to the `chawkbazar-api` folder, then copy `.env.example` to `.env`,

```
cd /var/www/chawkbazar/chawkbazar-api
cp .env.example .env
```

Edit `.env`

```
nano .env
```

And add MySQL, stripe, mail or others configuration.

Also, add https://YOUR\_DOMAIN.COM/api to APP\_URL. Without this, the upload function will be broken.

```
1 APP_NAME=ChawkBazar
2 APP_ENV=production ←
3 APP_KEY=base64:dladdfdsfdfdsfssdf/sdfdsfer
4 APP_DEBUG=true
5 APP_URL=https://YOUR_DOMAIN.COM/backend ←
6 APP_SERVICE=chawkbazar.test
7 APP_NOTICE_DOMAIN=CHAWKBAZAR_
8
9 DUMMY_DATA_PATH=chawkbazar
10
11 LOG_CHANNEL=stack
12 LOG_LEVEL=debug
13
```

Then install all the packages and install api

```
composer install

php artisan key:generate

php artisan marvel:install
```

You'll get several confirmations for migration, dummy data, and admin account. Make sure you check the confirmation step and take the necessary actions based on your requirement.

Enable laravel storage,

```
php artisan storage:link
```

Then give proper permission for laravel folder,

```
sudo chown -R www-data:www-data storage
sudo chown -R www-data:www-data bootstrap/cache
```

Now, when you go to the YOUR\_DOMAIN/api you'll get a welcome page like this

## FrontEnd Project Build

TypeScript requires a huge chunk of memory to build the project, so if your server has at least 8gb+ of memory, then you can build the project on your server directly. If not, then build the project on your server, then move the folder to the server then serve the project. We'll do the second method in this tutorial.

We'll suggest you build the frontend part on your computer and then upload the build file to the server.

Go to your `chawkbazar-laravel` folder from your `local computer`.

## Step 1 - Config Next Admin App For /admin Sub Directory

Edit `admin/rest/next.config.js`,

add `basePath` for '/admin'

```
1 const withPWA = require("next-pwa");
2 const runtimeCaching = require("next-pwa/cache");
3 const { i18n } = require("./next-i18next.config");
4
5 module.exports = withPWA({
6   basePath: "/admin", ←
7   i18n,
8   pwa: {
9     disable: process.env.NODE_ENV === "development",
10    dest: "public",
11    runtimeCaching,
12  },
13
14  images: {
15    domains: [
16      "via.placeholder.com",
17      "res.cloudinary.com",
18      "s3.amazonaws.com",
19      "18.141.64.26",
20      "127.0.0.1",
21      "localhost",
22      "picsum.photos",
23      "pickbazar-sail.test",
24      "pickbazarlaravel.s3.ap-southeast-1.amazonaws.com",
25      "chawkbazarlaravel.s3.ap-southeast-1.amazonaws.com",
26      "lh3.googleusercontent.com",
27    ],
28  },
```

## Step 2 - Install & Build

go to your `chawkbazar-laravel` folder again

To install all the npm packages run this command,

```
yarn
```

## Step 3 - Build the project

At first, we've to copy the sample `.env.template` to production `.env` for the shop and admin first.

### Shop

Go to `shop` and copy `.env.template` to `.env`

Then edit `.env` and add your `API` url to that file

```
NEXT_PUBLIC_REST_API_ENDPOINT="https://YOUR_DOMAIN/api"
```

Also, update all the necessary configurations like `OTP`, `Mailchimp Token`, `Instagram Token`, `Google MAP API`, etc.

### Admin

After that, Go to `admin -> rest` and copy `.env.template` to `.env`

Then edit `.env` and add your `API` url to that file

```
NEXT_PUBLIC_REST_API_ENDPOINT="https://YOUR_DOMAIN/api"
```

Also, update all the necessary configurations like `Google MAP API`, etc.

### Image Config

Then open `shop -> next.config.js` and `admin -> rest -> next.config.js`

and add your domain to `images` object

```
1 const withPWA = require("next-pwa");
2 const runtimeCaching = require("next-pwa/cache");
3 const { i18n } = require("./next-i18next.config");
4
5 module.exports = withPWA({
6   i18n,
7   pwa: {
8     disable: process.env.NODE_ENV === "development",
9     dest: "public",
10    runtimeCaching,
11  },
12
13  images: {
14    domains: [
15      "YOUR_API_DOMAIN", ↓
16      "via.placeholder.com",
17      "res.cloudinary.com",
18      "s3.amazonaws.com",
19      "18.141.64.26",
20      "127.0.0.1",
21      "localhost",
22      "picsum.photos",
23      "pickbazar-sail.test",
24      "pickbazarlaravel.s3.ap-southeast-1.amazonaws.com",
25      "chawkbazarlaravel.s3.ap-southeast-1.amazonaws.com",
26      "lh3.googleusercontent.com",
27    ],
28  },
29}
```

If your API is hosted on a subdomain, then add that subdomain with root domain on `next.config.js`

## Build Project

Now go to the `chawkbazar-laravel` folder,

run this for build `admin`

```
yarn build:admin-rest
```

And for the `shop`,

at first, run the shop in one terminal

```
yarn dev:shop-rest
```

and then build the shop,

```
yarn build:shop-rest
```

Now zip `admin`, `shop`, `package.json`, `babel.config.js` and `yarn.lock` files and upload them to the server `/var/www/chawkbazar`

Now go to the server `/var/www/chawkbazar` using terminal

Then `unzip` that `zip` file.

## Install FrontEnd And Run

Then install all the node packages,

```
yarn
```

## Run frontend app

For `shop` app, use this command,

```
pm2 --name shop-rest start yarn -- run start:shop-rest
```

Then to run the `admin` app, use this command,

```
pm2 --name admin-rest start yarn -- run start:admin-rest
```

Now go to Now, go to your `YOUR_DOMAIN` to access the shop page and `YOUR_DOMAIN/admin` for the access admin section.

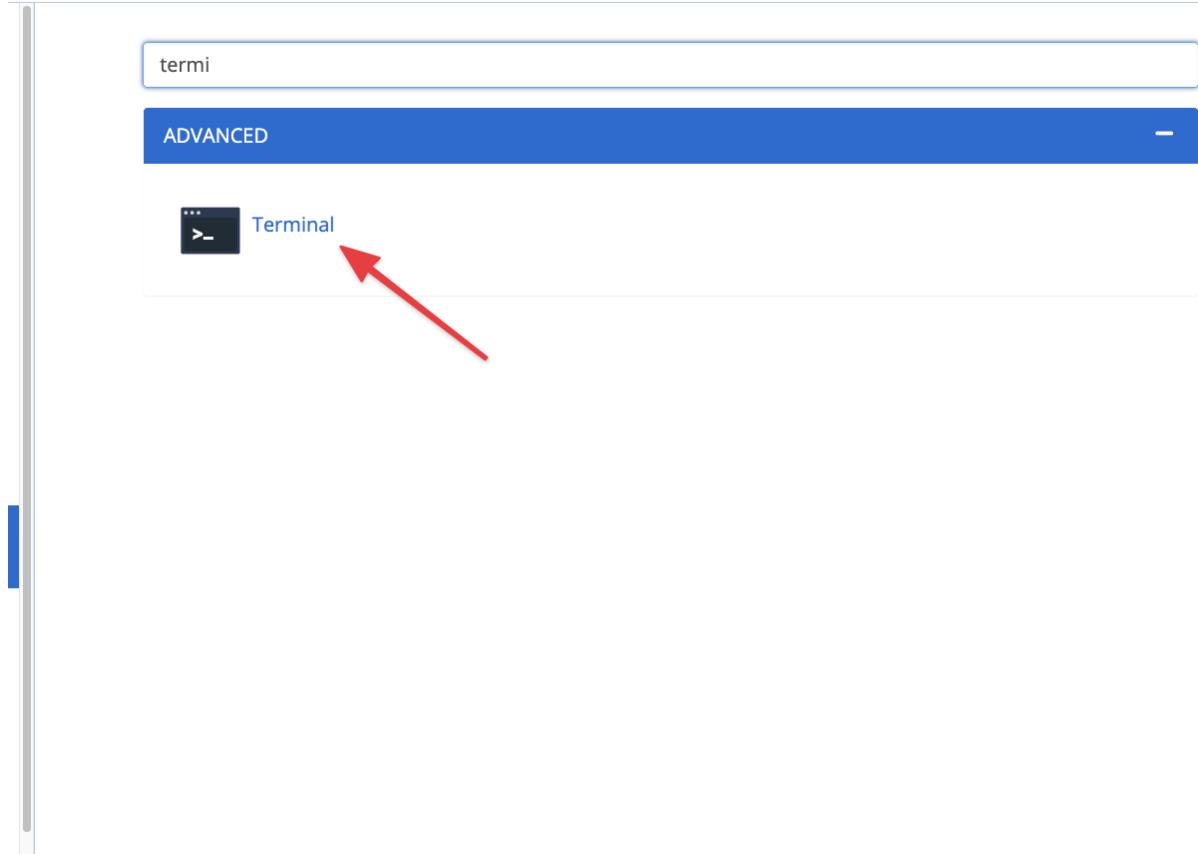
## cPanel

It's quite hard to debug any deployment issue on the cPanel or any managed server as the provider manages this type of server, and they've complete control of the server. And for that, We don't recommend Cpanel or any managed server for deployment. We suggest you use any VPS server where you have complete control of it. you can purchase any \$5 - \$10/mo server from amazon lightsail, ec2 or digitalocean or any ubuntu server

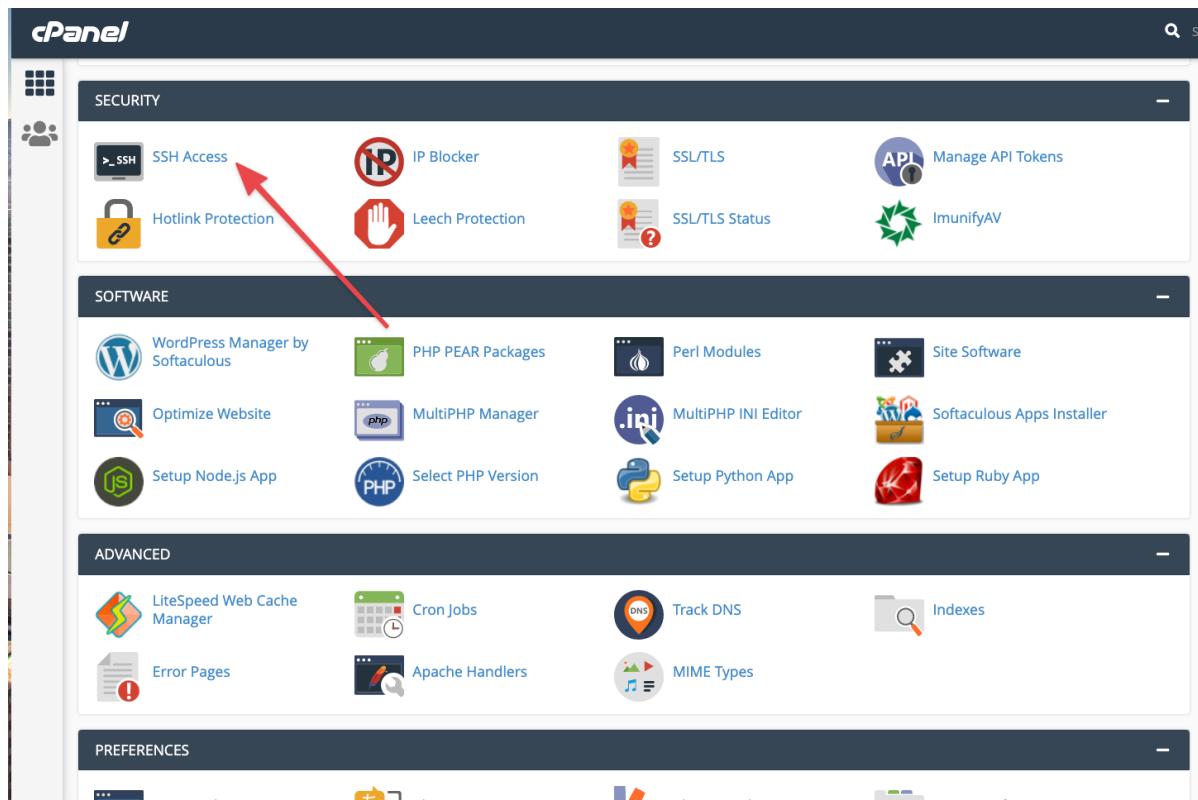
If you still decide to proceed with panel based server, our support team won't be able to help you. We have put some resources for Cpanel in this documentation section to help our users to get started but other than that, we don't have much to offer with Cpanel or any Managed Server.

## Access Server

To install the API, access the server using the cPanel terminal first,



If you don't find the terminal, then login to your local computer terminal or [putty](#) for Windows using SSH.



After enabling the ssh login to your server using ssh,

If you don't see any option, then contact your hosting provider as cPanel control by hosting provider.

After logging in, Check if the composer is already installed or not using this command,

```
composer -v
```



```
[~]# composer --version
Composer version 2.0.6 2020-11-07 11:21:17
```

If composer is not installed then, install `composer` to your server.

Check this [YouTube Video](#) for install `composer` on your server,

After that, check the PHP version using,

```
php -v
```

make sure it's `8.1`

## Create Subdomains

Now create two subdomains, for example,

```
-> your_domain.com -> host frontend store.
-> api.your_domain.com -> host laravel API.
-> admin.your_domain.com -> host admin dashboard.
```

Or if you want to host all the script on subdomains, then create subdomains like this,

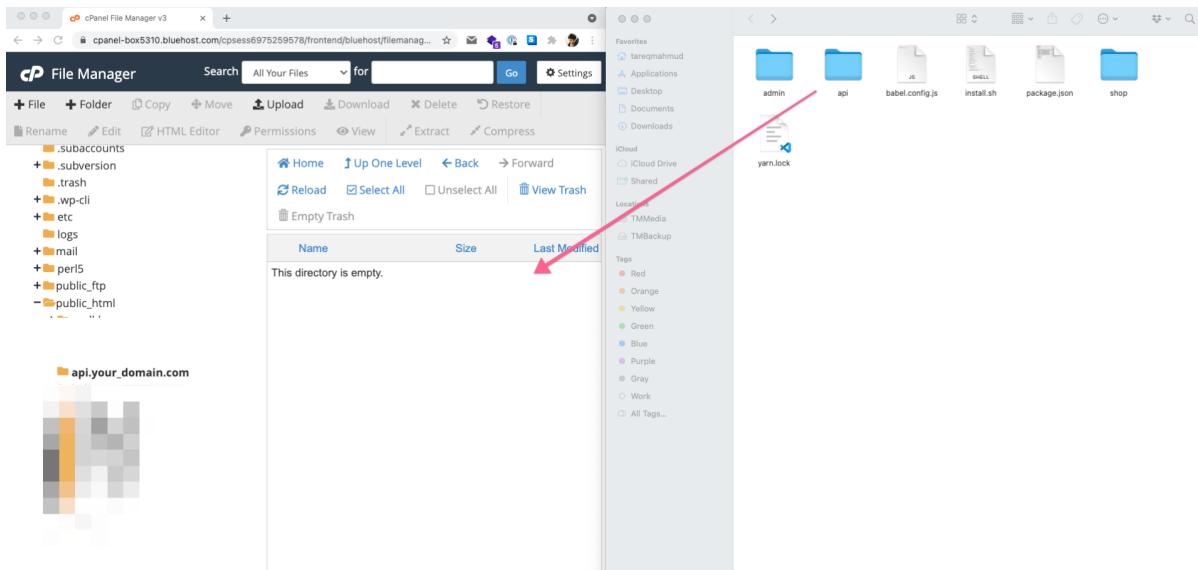
```
-> store.your_domain.com -> host frontend store.
-> api.your_domain.com -> host laravel API.
-> admin.your_domain.com -> host admin dashboard.
```

After creating domain/subdomains, make sure all the domain/subdomains are HTTPS enabled. Please contact your hosting provider to enable this, as most hosting providers provide some sort of free SSL.

## Install API

1. Extract the `chawkbazar` package that you download from [CodeCanyon](#).
2. On that folder, you'll get another `zip` called `chawkbazar-laravel.zip`.
3. Now extract this `chawkbazar-laravel.zip` file.
4. On that file, you'll get a folder called `api`

Now upload this `api` folder to the `api.your_domain.com` folder in your server



Make sure your `api.your_domain.com` subdomain Document Root points to that `api/public` folder.

The screenshot shows the cPanel Subdomains section. It displays instructions for creating a subdomain as a new website. The 'Create a Subdomain' form is filled out with 'api' in the Subdomain field, a specific domain from the Domain dropdown, and the Document Root set to `/api/public`. Three red arrows highlight the 'Subdomain' field, the 'Domain' dropdown, and the 'Document Root' field.

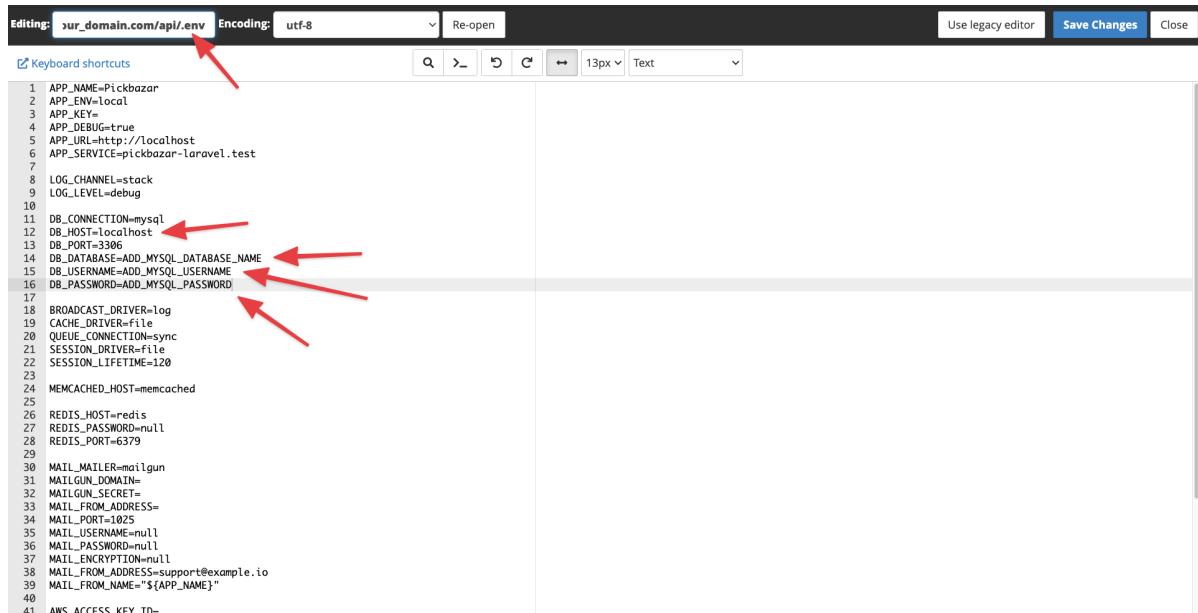
Now create a MySQL database and user from MySQL wizard

The screenshot shows the cPanel main dashboard. It features several icons for different services like Web Disk, FTP Accounts, Backup, and JetBackup. Below these are sections for DATABASES, DOMAINS, METRICS, and SECURITY. A prominent red arrow points to the 'MySQL® Databases' link under the DATABASES section.

After creating the MySQL database, go to your `api` folder from your cPanel file manager and copy `.env.example` to `.env`.

Name	Size	Last Modified	Type	Permissions
app	82 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
bootstrap	34 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
config	300 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
database	74 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
packages	95 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
public	106 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
resources	52 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
routes	75 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
storage	46 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
tests	83 bytes	Apr 4, 2021, 4:53 PM	httpd/unix-directory	0775
.editorconfig	220 bytes	Apr 4, 2021, 4:53 PM	text/x-generic	0664
.env	871 bytes	Today, 9:33 PM	text/x-generic	0664
.env.example	871 bytes	Apr 4, 2021, 4:53 PM	text/x-generic	0664
.gitattributes	111 bytes	Apr 4, 2021, 4:53 PM	text/x-generic	0664
.gitignore	216 bytes	Apr 4, 2021, 4:53 PM	text/x-generic	0664
.styleci.yml	181 bytes	Apr 4, 2021, 4:53 PM	text/x-generic	0664
artisan	1.65 KB	Apr 4, 2021, 4:53 PM	text/x-generic	0755
composer.json	2.38 KB	Apr 4, 2021, 4:53 PM	text/x-generic	0664
composer.lock	390.9 KB	Apr 4, 2021, 4:53 PM	text/x-generic	0664
docker-compose.yml	1.23 KB	Apr 4, 2021, 4:53 PM	text/x-generic	0664

After the copy, edit `.env` and add MySQL credentials,



The screenshot shows a text editor window with the following configuration:

- Editing: `sur_domain.com/api/.env`
- Encoding: `utf-8`
- Buttons: Re-open, Use legacy editor, Save Changes, Close
- Keyboard shortcuts: Enabled
- Text area content:

```

1 APP_NAME=Pickbazar
2 APP_ENV=local
3 APP_KEY=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6 APP_SERVICE=pickbazar-laravel.test
7
8 LOG_CHANNEL=stack
9 LOG_LEVEL=debug
10
11 DB_CONNECTION=mysql
12 DB_HOST=localhost
13 DB_PORT=3306
14 DB_DATABASE=ADD_MYSQL_DATABASE_NAME
15 DB_USERNAME=ADD_MYSQL_USERNAME
16 DB_PASSWORD=ADD_MYSQL_PASSWORD
17
18 BROADCAST_DRIVER=log
19 CACHE_DRIVER=file
20 QUEUE_CONNECTION=sync
21 SESSION_DRIVER=file
22 SESSION_LIFETIME=120
23
24 MEMCACHED_HOST=memcached
25
26 REDIS_HOST=redis
27 REDIS_PASSWORD=null
28 REDIS_PORT=6379
29
30 MAIL_MAILER=mailgun
31 MAILGUN_DOMAIN=
32 MAILGUN_SECRET=
33 MAIL_FROM_ADDRESS=
34 MAIL_PORT=1025
35 MAIL_USERNAME=null
36 MAIL_PASSWORD=null
37 MAIL_ENCRYPTION=null
38 MAIL_FROM_ADDRESS=support@example.io
39 MAIL_FROM_NAME="${APP_NAME}"
40
41 AWS_ACCESS_KEY_ID=

```

Red arrows point to the following lines in the code:

- `DB_HOST=localhost`
- `DB_DATABASE=ADD_MYSQL_DATABASE_NAME`
- `DB_USERNAME=ADD_MYSQL_USERNAME`
- `DB_PASSWORD=ADD_MYSQL_PASSWORD`

Also, add `https://YOUR_DOMAIN.COM/api` to `APP_URL`. Without this, the `upload` function will be broken.



The screenshot shows a text editor window with the following configuration:

- Editing: `sur_domain.com/api/.env`
- Text area content:

```

1 APP_NAME=ChawkBazar
2 APP_ENV=production
3 APP_KEY=base64:dLaddfd$fd$fsdfssdf/sdfdsfer
4 APP_DEBUG=true
5 APP_URL=https://YOUR_DOMAIN.COM/backend
6 APP_SERVICE=chawkbazar.test
7 APP_NOTICE_DOMAIN=CHAWKBAZAR_
8
9 DUMMY_DATA_PATH=chawkbazar
10
11 LOG_CHANNEL=stack
12 LOG_LEVEL=debug
13

```

Red arrows point to the following lines in the code:

- `APP_ENV=production`
- `APP_URL=https://YOUR_DOMAIN.COM/backend`

Then go to your `ssh terminal` again and,

go to `api` folder and run,

```
composer install
```

If `composer` installs all the packages successfully, then run this command on the `api` folder,

```
php artisan key:generate
```

```
php artisan marvel:install
```

You'll get several confirmations for migration, dummy data, and admin account. Make sure you check the confirmation step and take the necessary actions based on your requirement.

After that, run this command to link storage,

```
php artisan storage:link
```

After install, go to your `api.your_domain_name.com`, and you'll get a webpage like this,

## Marvel Laravel

GRAPHQL PLAYGROUND DOCUMENTATION SUPPORT CONTACT

## Install FrontEnd

Before proceeding next step, make sure you already create two subdomains like this,

```
-> your_domain.com -> host frontend store.  
-> admin.your_domain.com -> host admin dashboard.
```

OR

```
-> store.your_domain.com -> host frontend store.  
-> admin.your_domain.com -> host admin dashboard.
```

# FrontEnd Project Build

TypeScript requires a huge chunk of memory to build the project, so if your server has at least 8gb+ of memory, then you can build the project on your server directly. If not, then build the project on your server, then move the folder to the server then serve the project. We'll do the second method in this tutorial.

We'll suggest you build the frontend part on your computer and then upload the build file to the server.

## step 1 - Build Custom Server

go to your `chawkbazar-laravel` folder

**shop**

Create custom server for `shop`,

create a `server.js` file on,

shop/server.js

and paste this code,

```
// server.js
const { createServer } = require('http')
const { parse } = require('url')
const next = require('next')

const dev = process.env.NODE_ENV !== 'production'
const app = next({ dev })
const handle = app.getRequestHandler()

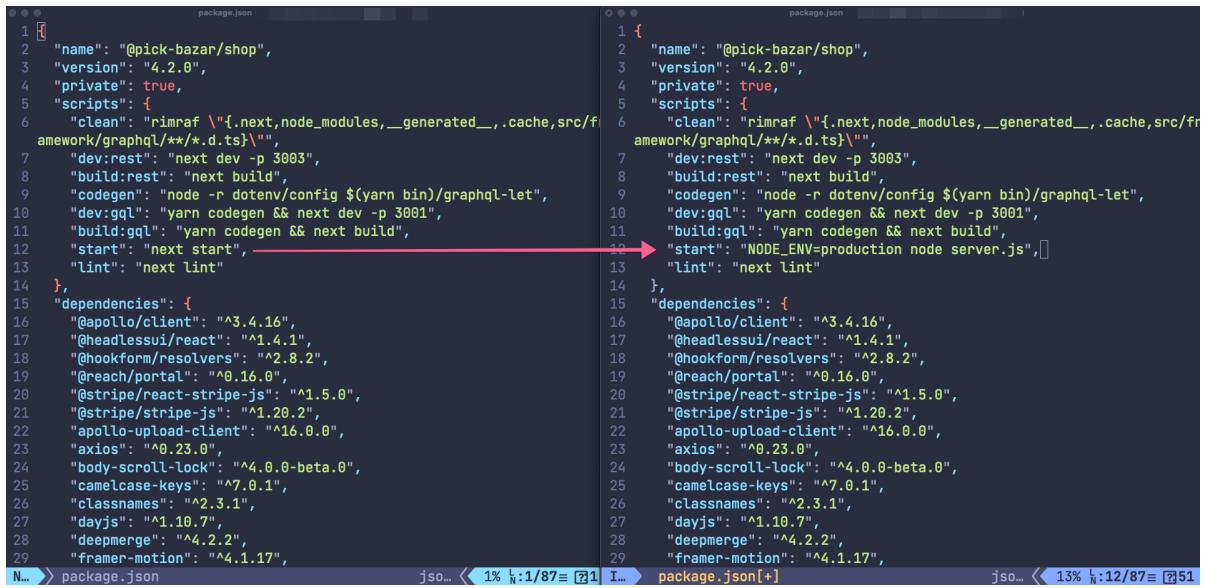
app.prepare().then(() => {
  createServer((req, res) => {
    // Be sure to pass `true` as the second argument to `url.parse`.
    // This tells it to parse the query portion of the URL.
    const parsedUrl = parse(req.url, true)
    const { pathname, query } = parsedUrl

    if (pathname === '/a') {
      app.render(req, res, '/a', query)
    } else if (pathname === '/b') {
      app.render(req, res, '/b', query)
    } else {
      handle(req, res, parsedUrl)
    }
  }).listen(3003, (err) => {
    if (err) throw err
    console.log('> Ready on http://localhost:3003')
  })
})
```

Now update package.json for `shop/package.json`,

and replace `start` script with this,

```
"start": "NODE_ENV=production node server.js"
```



```
1 { package.json
2   "name": "@pick-bazar/shop",
3   "version": "4.2.0",
4   "private": true,
5   "scripts": {
6     "clean": "rimraf '{.next,node_modules,__generated__,.cache,src/framework/graphql/**/*.d.ts}'",
7     "dev:rest": "next dev -p 3003",
8     "build:rest": "next build",
9     "codegen": "node -r dotenv/config $(yarn bin)/graphql-let",
10    "dev:gql": "yarn codegen && next dev -p 3001",
11    "build:gql": "yarn codegen && next build",
12    "start": "next start",
13    "lint": "next lint"
14  },
15  "dependencies": {
16    "@apollo/client": "^3.4.16",
17    "@headlessui/react": "1.4.1",
18    "@hookform/resolvers": "2.8.2",
19    "@reach/portal": "^0.16.0",
20    "@stripe/react-stripe-js": "^1.5.0",
21    "@stripe/stripe-js": "1.20.2",
22    "apollo-upload-client": "16.0.0",
23    "axios": "0.23.0",
24    "body-scroll-lock": "4.0.0-beta.0",
25    "camelcase-keys": "7.0.1",
26    "classnames": "2.3.1",
27    "dayjs": "1.10.7",
28    "deepmerge": "4.2.2",
29    "framer-motion": "4.1.17",
30  }
31 } package.json js... ↵ 1% ↵ 1/87 ↵ 51 I... package.json[+]
32 { package.json
33   "name": "@pick-bazar/shop",
34   "version": "4.2.0",
35   "private": true,
36   "scripts": {
37     "clean": "rimraf '{.next,node_modules,__generated__,.cache,src/framework/graphql/**/*.d.ts}'",
38     "dev:rest": "next dev -p 3003",
39     "build:rest": "next build",
40     "codegen": "node -r dotenv/config $(yarn bin)/graphql-let",
41     "dev:gql": "yarn codegen && next dev -p 3001",
42     "build:gql": "yarn codegen && next build",
43     "start": "NODE_ENV=production node server.js",
44     "lint": "next lint"
45   },
46   "dependencies": {
47     "@apollo/client": "3.4.16",
48     "@headlessui/react": "1.4.1",
49     "@hookform/resolvers": "2.8.2",
50     "@reach/portal": "0.16.0",
51     "@stripe/react-stripe-js": "1.5.0",
52     "@stripe/stripe-js": "1.20.2",
53     "apollo-upload-client": "16.0.0",
54     "axios": "0.23.0",
55     "body-scroll-lock": "4.0.0-beta.0",
56     "camelcase-keys": "7.0.1",
57     "classnames": "2.3.1",
58     "dayjs": "1.10.7",
59     "deepmerge": "4.2.2",
60     "framer-motion": "4.1.17",
61   }
62 } package.json js... ↵ 13% ↵ 12/87 ↵ 51 I...
```

## admin rest

Similarly, create custom server for `admin rest`,

create another `server.js` file on,

```
admin/rest/server.js
```

and paste this code,

```
// server.js
const { createServer } = require('http')
const { parse } = require('url')
const next = require('next')

const dev = process.env.NODE_ENV !== 'production'
const app = next({ dev })
const handle = app.getRequestHandler()

app.prepare().then(() => {
  createServer((req, res) => {
    // Be sure to pass `true` as the second argument to `url.parse`.
    // This tells it to parse the query portion of the URL.
    const parsedUrl = parse(req.url, true)
    const { pathname, query } = parsedUrl

    if (pathname === '/a') {
      app.render(req, res, '/a', query)
    } else if (pathname === '/b') {
      app.render(req, res, '/b', query)
    } else {
      handle(req, res, parsedUrl)
    }
  }).listen(3002, (err) => {
```

```
    if (err) throw err
    console.log('> Ready on http://localhost:3002')
  })
}
```

Now update package.json for `admin/rest/package.json`,

and replace `start` script with this,

```
"start": "NODE_ENV=production node server.js"
```

## Step 2 - Install & Build

go to your `chawkbazar-laravel` folder

To install all the npm packages run this command,

```
yarn
```

## Step 3 - Build the project

At first, we've to copy the sample `.env.template` to production `.env` for the `shop` and `admin` first.

Go to,

```
cd shop
```

then use this command to copy,

```
cp .env.template .env
```

Now edit `.env` and add you API url to `.env`

```
nano .env
```

and use

```
NEXT_PUBLIC_REST_API_ENDPOINT=https://api.YOUR_DOMAIN.com/
```

After that, go to the `admin -> rest` folder,

```
cd ../admin/rest
```

then use this command to copy,

```
cp .env.template .env
```

```
nano .env
```

and use

```
NEXT_PUBLIC_REST_API_ENDPOINT=https://api.YOUR_DOMAIN.com/
```

Now go to the `chawkbazar-laravel` folder,

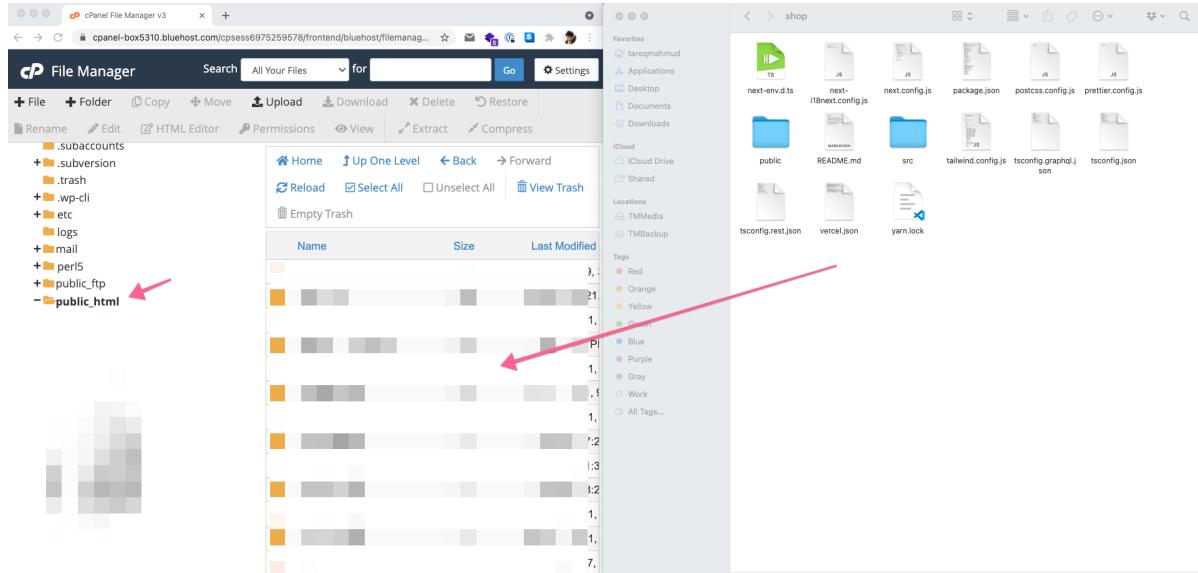
and run,

```
yarn build:shop-rest  
yarn build:admin-rest
```

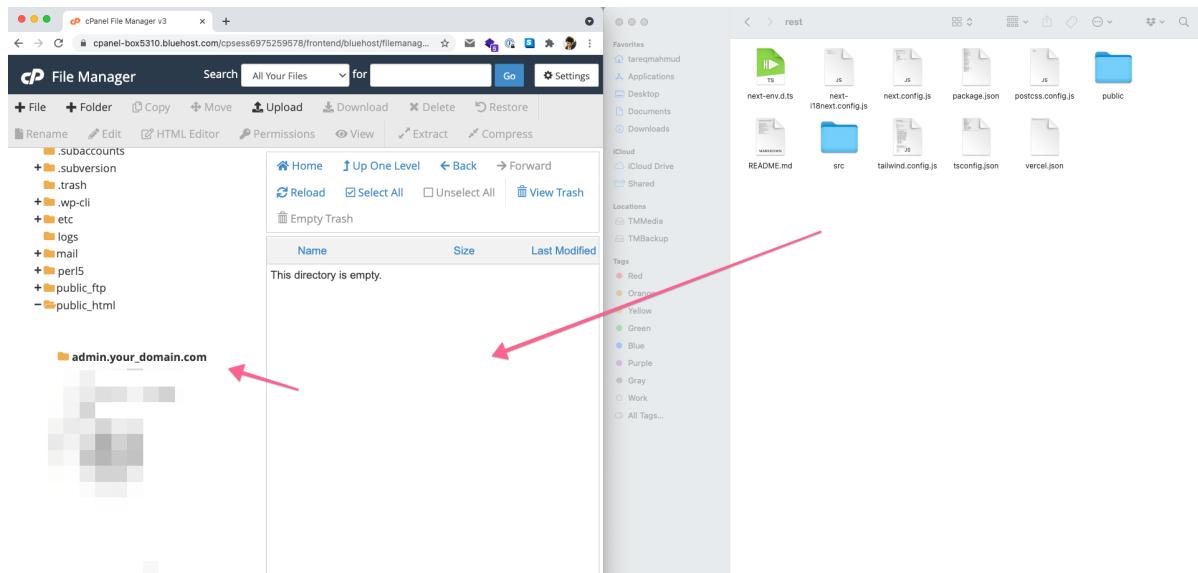
After build the project upload the

- `shop` to `root_domain -> public_html` folder
- `admin-rest` to `admin.your_domain.com` folder

`shop`,



`admin-rest`,



# Install NodeJs Project

We'll run both `shop` and `admin` using the cPanel NodeJs application in this step.

To do that at first go to the NodeJS section from your cPanel,

For `shop`,

The screenshot shows the cPanel software section. At the top, there are several icons for various services like SSH Access, IP Blocker, SSL/TLS, and Manage API Tokens. Below this, the 'SOFTWARE' section is visible, containing icons for WordPress Manager by Softaculous, PHP PEAR Packages, Perl Modules, Site Software, Optimize Website, MultiPHP Manager, MultiPHP INI Editor, Softaculous Apps Installer, Setup Node.js App (which has a red arrow pointing to it), Select PHP Version, Setup Python App, and Setup Ruby App. A second red arrow points to the 'ADVANCED' section below, which includes LiteSpeed Web Cache Manager, Cron Jobs, Track DNS, Indexes, Error Pages, Apache Handlers, and MIME Types. The bottom section is labeled 'PREFERENCES' with icons for Password & Security, Change Language, Change Style, and Contact Information.

The screenshot shows the cPanel Node.js application section. At the top, there is a header with the cPanel logo, a search bar, and user information. Below this, the 'WEB APPLICATIONS' section is shown, featuring a 'Node.js' icon. A red arrow points to the 'CREATE APPLICATION' button in the top right corner of this section. The main area displays the message 'NO APPLICATIONS FOUND'. At the bottom, there is footer information including the cPanel version (96.0.8) and links to Home, Trademarks, Privacy Policy, and Documentation.

Now,

- Select NodeJS version
- Make environment `production`.
- Set Application Root
- And application startup file as `server.js`

**cPanel**

WEB APPLICATIONS [CREATE APPLICATION](#)

Node.js version: 12.22.1 (red arrow)

Application mode: Production (red arrow)

Application root: /home/tareqma3/public\_html (red arrow)

Application URL: [REDACTED] (red arrow)

Application startup file: server.js (red arrow)

Passenger log file: /home/tareqma3/logs/passenger.log

**Environment variables** [ADD VARIABLE](#)

NO RESULT FOUND

You can get the Application Path from your cPanel file manager

**cPanel File Manager**

Search: All Your Files for: Go Settings

File Manager: Home Up One Level Back Forward Reload Select All Unselect All View Trash Empty Trash

Name	Size	Last Modified	Type	Permissions
.cagefs	39 bytes	Feb 12, 2021, 2:10 PM	httpd/unix-directory	0771
.cl.selector	48 bytes	May 13, 2021, 1:44 PM	httpd/unix-directory	0755
.cpanel	195 bytes	Today, 5:16 PM	httpd/unix-directory	0700
.cphorde	70 bytes	Mar 24, 2021, 9:28 PM	httpd/unix-directory	0700
.gnupg	79 bytes	Feb 13, 2021, 10:31 PM	httpd/unix-directory	0700
.htpasswdws	6 bytes	Jan 23, 2021, 5:05 PM	httpd/unix-directory	0750
.razor	231 bytes	Feb 23, 2021, 10:19 PM	httpd/unix-directory	0755
.softaculous	78 bytes	Jan 23, 2021, 5:06 PM	httpd/unix-directory	0711
.spamassassin	24 bytes	Jan 23, 2021, 5:05 PM	httpd/unix-directory	0700
.subaccounts	28 bytes	Mar 24, 2021, 9:28 PM	httpd/unix-directory	0700
.trash	217 bytes	Feb 13, 2021, 2:09 AM	httpd/unix-directory	0700
etc	80 bytes	Today, 3:05 PM	httpd/unix-directory	0750
logs	243 bytes	Yesterday, 6:28 PM	httpd/unix-directory	0700
Iscache	177 bytes	Feb 14, 2021, 2:24 AM	httpd/unix-directory	2770
IscmData	79 bytes	Feb 12, 2021, 7:02 PM	httpd/unix-directory	0700
mail	239 bytes	Mar 24, 2021, 9:28 PM	mail	0751
public_ftp	22 bytes	Jan 23, 2021, 5:05 PM	publicftp	0750
public_html	162 bytes	Today, 3:55 PM	publhtml	0750
ssl	77 bytes	May 12, 2021, 1:20 PM	httpd/unix-directory	0755

Collapsible sidebar: /home/tareqma3 (red arrow), public\_ftp (red arrow)

After create NodeJS app, install all the packages and restart the app,

**cPanel**

WEB APPLICATIONS [STOP APP](#) [RESTART](#) [DESTROY](#) CANCEL SAVE

Node.js: RESTART (red arrow)

Node.js version: 12.22.1

Application mode: Production

Application root: /public\_html (red arrow)

Application URL: [REDACTED]

Application startup file: server.js

Passenger log file: /home/tareqma3/

**Detected configuration files** [Run NPM Install](#) (red arrow)

For admin,

Similarly, create another NodeJS application for admin with `admin subdomain` and `admin subdirectory`

After installing and run both NodeJS application, you can access your domain to check ChawkBazar,

Thank You!

## SEO and Analytics

### SEO

For SEO, we use `Next SEO` packages, and we provide two boilerplates at

```
shop -> src -> components -> seo
```

With that boilerplate, check `Next SEO` docs to use SEO on your chawkbazar site,

<https://github.com/garmeeh/next-seo>

### Analytics

We have not used any analytics integration in this template yet. But, you can easily integrate any analytics using [Next JS examples](#).

## Laravel API

### Introduction

Chawkbazar is a laravel multi api package for ecommerce. This package is for building dynamic ecommerce site using chawkbazar package.

### Getting Started

For getting started with the template you have to follow the below procedure. For quick guide you can check below videos for installation.

### Installation Video

### Prerequisites

- PHP 8.1
- Composer
- Xamp/Wamp/Lamp for any such application for apache, nginx, mysql
- PHP plugins you must need
  - simplexml
  - PHP's dom extension

- mbstring
- GD Library

## Resources you might need

1. <https://laravel.com/docs/10.x>
2. <https://lighthouse-php.com/5/getting-started/installation.html>
3. <https://github.com/spatie/laravel-medialibrary>
4. <https://github.com/andersao/l5-repository>
5. <https://spatie.be/docs/laravel-permission/v3/introduction>

## Packages we have used

```

"require": {
    "php": "^8.0|^8.1",
    "barryvdh/laravel-dompdf": "2.0.1",
    "doctrine/dbal": "3.6.0",
    "guzzlehttp/guzzle": "7.5.0",
    "laravel/framework": "10.1.5",
    "laravel/socialite": "5.6.1",
    "laravel/tinker": "2.8.1",
    "marvel/shop": "dev-master",
    "messagebird/php-rest-api": "3.1.4",
    "openai-php/client": "^0.4.1",
    "psr/log": "2.0.0",
    "srmklive/paypal": "3.0.19",
    "stevebauman/purify": "5.1.1",
    "symfony/http-client": "6.2.6",
    "symfony/mailgun-mailer": "6.2.5",
    "twilio/sdk": "6.44.4"
},
"require-dev": {
    "spatie/laravel-ignition": "1.3.1",
    "fakerphp/faker": "1.20.0",
    "laravel/sail": "1.15.1",
    "mockery/mockery": "1.5.0",
    "nunomaduro/collision": "6.2.1",
    "phpunit/phpunit": "9.5.21",
    "squizlabs/php_codesniffer": "3.7.1"
},

```

## Installation

- Make sure you have run xamp/mamp/wamp/lamp for mysql and php
    - Create a database in your mysql and put those info in next step
    - Rename .env.example file to .env and provide necessary credentials. Like database credentials stripe credentials, s3 credentials(only if you use s3 disk) admin email shop url etc.
- Specially check for this `env` variables

```
DB_HOST=localhost
DB_DATABASE=chawkbazar_laravel
DB_USERNAME=root
DB_PASSWORD=
```

- o Run `composer install`

```
▶ composer install
Installing dependencies from lock file (including require-dev)
Verifying lock file contents can be installed on current platform.

Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi
Discovered Package: barryvdh/laravel-dompdf
Discovered Package: bensampo/laravel-enum
Discovered Package: cviebrock/eloquent-sluggable
Discovered Package: facade/ignition
Discovered Package: fideloper/proxy
Discovered Package: fruitcake/laravel-cors
Discovered Package: ignited/laravel-omnipay
Discovered Package: intervention/image
Discovered Package: laravel/legacy-factories
Discovered Package: laravel/sail
Discovered Package: laravel/sanctum
Discovered Package: laravel/tinker
Discovered Package: mll-lab/laravel-graphql-playground
Discovered Package: nesbot/carbon
Discovered Package: nunomaduro/collision
Discovered Package: nuwave/lighthouse
Discovered Package: pickbazar/shop
Discovered Package: prettus/l5-repository
Discovered Package: spatie/laravel-medialibrary
Discovered Package: spatie/laravel-permission
Package manifest generated successfully.
95 packages you are using are looking for funding.
Use the `composer fund` command to find out more!
```

- o run `php artisan key:generate`

```
▶ php artisan key:generate
Application key set successfully.
```

- o Run `php artisan marvel:install` and follow necessary steps.

```
Installing Pickbazar Dependencies...
Do you want to migrate Tables? If you have already run this command or migrated tables then be aware, it will erase all of your data. (yes/no) [no]:
> yes

Migrating Tables Now....
Dropped all tables successfully.
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table (79.70ms)
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table (58.57ms)
Migrating: 2019_08_19_000000_create_failed_jobs_table
Migrated: 2019_08_19_000000_create_failed_jobs_table (79.56ms)
Migrating: 2019_10_14_000001_create_personal_access_tokens_table
Migrated: 2019_10_14_000001_create_personal_access_tokens_table (143.06ms)
Migrating: 2020_04_17_194830_create_permission_tables
Migrated: 2020_04_17_194830_create_permission_tables (839.79ms)
Migrating: 2020_06_02_051901_create_pickbazar_tables
Migrated: 2020_06_02_051901_create_pickbazar_tables (1,601.56ms)
Migrating: 2020_10_26_163529_create_media_table
Migrated: 2020_10_26_163529_create_media_table (71.72ms)
Tables Migration completed.

Do you want to seed dummy data? (yes/no) [no]:
> yes

Copying necessary files for seeding...
File copying successful
Seeding...
Seed completed successfully!

Do you want to create an admin? (yes/no) [no]:
> no

Copying resources files...
Installation Complete
```

- o For image upload to work properly you need to run `php artisan storage:link`.

```
The [/var/www/html/public/storage] link has been connected to [/var/www/html/storage/app/public].
The links have been created.
```

- o run `php artisan serve`

```
▶ php artisan serve
Starting Laravel development server: http://127.0.0.1:8000
[Sat Apr 10 16:35:26 2021] PHP 8.0.0 Development Server (http://127.0.0.1:8000) started
```

NB: You must need to run `php artisan marvel:install` to finish the installation. Otherwise your api will not work properly. Run the command and follow the necessary steps.

NB: your frontend `NEXT_PUBLIC_GRAPHQL_API_ENDPOINT` env value will be `localhost:8000/graphql`

NB: your frontend `NEXT_PUBLIC_REST_API_ENDPOINT` env value will be `localhost:8000/`

## For MAC and Linux(with sail and docker)

There is an alternate installation procedure for linux and mac. You can follow below procedure to getting started with `sail`

### Prerequisites

- Docker

### Installation

- Run Docker application first
- Now go to your chawkbazar-laravel root directory and run `bash install.sh`. It will guide you through some process. Follow those steps carefully and your app will be up and running
- Navigate to `api` then `sail down` to stop the container. If you want to remove the volumes then `sail down -v`

NB: your frontend `NEXT_PUBLIC_GRAPHQL_API_ENDPOINT` env value will be `localhost/graphql`

NB: your frontend `NEXT_PUBLIC_REST_API_ENDPOINT` env value will be `localhost/`

## Configuration

All the configurations files are in `packages/marvel/config` folder. You can change any necessary configuration from these files. You can also publishes the shop configuration using `artisan vendor:publish --provider="Marvel\ShopServiceProvider" --tag="config"` command in your root folder.

- Create `.env` file from our `example.env` file and put necessary configuration
- By default s3 is using for media storage but you can also use local folder. Change `MEDIA_DISK` IN `.env` file as your need. Supported options are `public` and 's3'
- Set Payment related configuration to `STRIPE_API_KEY` `.env` variable
- Set `ADMIN_EMAIL`, `SHOP_URL` and necessary Database credentials.

## Console Commands

- `php artisan marvel:install` complete installation with necessary steps
- `php artisan marvel:seed` seeding demo data
- `php artisan marvel:copy-files` copy necessary files

- `php artisan vendor:publish --provider="Marvel\ShopServiceProvider" --tag="config"` published the configuration file

All of the above custom command you will find in `packages/marvel/src/Console` folder.

## Development

---

We have provided below two api.

- REST API
- GraphQL API

## REST API

---

All the rest routes is resides in `packages/marvel/src/Rest/Routes.php` file and you can easily navigate to corresponding controller and necessary files.

## Endpoints

<https://documenter.getpostman.com/view/11693148/UVC5Fo9R>

## Folder structure

### config

The `packages/marvel/config` folder contains all the `config` for our app.

### database

The `packages/marvel/database` folder contains all the `factories` and `migrations`.

- **Http:**

Contains two folders. `Controllers` and `Requests`. All the necessary controllers and requests are in this two folder.

- **Database:**

Contains `Models` and `Repositories`. For repositories we have used `I5-repository` (<https://github.com/andersao/I5-repository>).

### Enums

All the `enums` that are used throughout the app is in `packages/marvel/src/Enums` folder.

### Events

All the events are in `packages/marvel/src/Events` folder.

### Listeners

All the listeners corresponding to the above events are in `packages/marvel/src/Listeners` folder

## Mail

All the mailables are in `packages/marvel/src/Mails` folder.

## Notifications

Notifications related to order placed is reside `packages/marvel/src/Notifications`. Currently we have provided mail notification but you can easily add others notification system using laravel conventions.

## Providers

All the secondary service providers that we have used in our app resides in `packages/marvel/src/Providers` folder. The main `ShopServiceProviders` reside in `packages/marvel/src/` folder.

## stubs

The `packages/marvel/stubs` folder contains all the necessary email templates and demo data related resources for the app.

## GraphQL API

### Laravel GraphQL API Endpoint

`your_domain/graphql`

To access the graphql api you will need a graphql client. GraphiQL is one of them. You can install it from below link(<https://www.electronjs.org/apps/graphiql>)

## Alternatives

- GraphQL Playground
- <https://altair.sirmuel.design/>

## Folder Structure

All the code specifically related to GraphQL reside in `packages/marvel/src/GraphQL/` folder.

- **Mutations:**

Folder contain necessary mutations files which is connected to rest Controller file for code reusability.

- **Queries:**

Folder contain necessary queries files which is connected to rest Controller file for code reusability.

## • Schema

This is the most important part of the graphql api. Check the lighthouse-php(<https://lighthouse-php.com/4.18/getting-started/installation.html>) doc for understanding how schema works. We have provided schema for all the models in our app. If you check the above `lighthouse-php` doc you will understand how schema works and how you can modify it to your need.

## Before Finishing up

Before you finishes the installation process make sure you have completed the below steps.

- Copied necessary files and content to your existing laravel projects(if using existing projects)
- Installed all the necessary dependencies.
- Ran `marvel:install` commands and followed the necessary steps.
- Created a .env file with all the necessary env variables in the provided projects.
- Put `DISK_NAME` configuration for `public` or 's3'
- Set Payment related configuration to `STRIPE_API_KEY`

## Payment Gateway

We have used `omnipay` for payment and given `stripe` and `cash_on_delivery` default. We have used `ignited/laravel-omnipay` by forking it in our packages due to some compatibility issue with Laravel 8.

## Extending The Functionality

If you want to extend the functionality of the app you can easily do it on your app. You would not need to modify code from our packages folder. Like you can add any `routes` and corresponding `controller` in your laravel app as your need. We highly suggest you to do all the modification in your app so you can update the package easily.

## Deployment

Its a basic laravel application so you can deploy it as any other laravel application. Make sure you have installed all the php required plugins we mentioned above.

## Social Login

You can use a social network like `Google` or `Facebook` as a login provider like email/username and password. To set up social login, follow the below procedure,

### Google

For Google, follow this procedure,

1. At first go to Google Console dashboard (<https://console.cloud.google.com/>)
2. From the console, create a new project

3. After creating the project, go to **APIs & Services**

-> OAuth consent screen.

The screenshot shows the 'APIs & Services' section of the Google Cloud Platform console. On the left, a sidebar lists several options: Dashboard, Library, Credentials, OAuth consent screen (which is selected and highlighted in blue), Domain verification, and Page usage agreements. A red arrow points from the text 'OAuth consent screen' to the 'OAuth consent screen' option in the sidebar. The main content area is titled 'OAuth consent screen' and contains a message: 'because your OAuth request includes additional scopes that haven't been approved.' Below this is a progress bar showing '0 users / 100 user cap'. A 'SHOW LESS' link is present. The next section is 'OAuth rate limits', which displays token grant rates over time. It shows a chart with two data series: '5 minutes' (blue line) and '1 day' (grey line). The '1 day' series has a value of 10,001. A red arrow points to the value '10,000' on the chart. A note below the chart states: 'No data is available for the selected time frame.' At the bottom, there is a link 'Let us know what you think about our OAuth experience' and a 'SHOW LESS' link.

4. Create an **External OAuth Consent**.

5. Then, go to the **credentials** section and create **OAuth Client ID**.

The screenshot shows the 'Credentials' section of the Google Cloud Platform console. The sidebar on the left shows the same list of options as the previous screenshot, with 'Credentials' selected. The main content area is titled 'Credentials' and includes a 'CREATE CREDENTIALS' button. Below it, there are three options: 'API key', 'OAuth client ID', and 'Service account'. A red arrow points from the text 'OAuth client ID' to the 'OAuth client ID' option. The 'OAuth client ID' section contains a table with one row. The table has columns for 'Name' (checkbox), 'Browser' (checkbox), 'Restrictions' (None), and 'Key' (A1zaSyA1np...-hFhc-rots). There are edit and delete icons for the key.

Add an **Authorized redirect URIs**. The format of the URIs should be:

`http://yourdomain.com/api/auth/callback/google`

API APIs & Services

Client ID for Web application

DOWNLOAD JSON RESET SE

Enabled APIs & services

Library

Credentials

OAuth consent screen

Domain verification

Page usage agreements

Name \* Web client 1

The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.

The domains of the URIs you add below will be automatically added to your [OAuth consent screen](#) as [authorized domains](#).

Authorized JavaScript origins ?

For use with requests from a browser

+ ADD URI

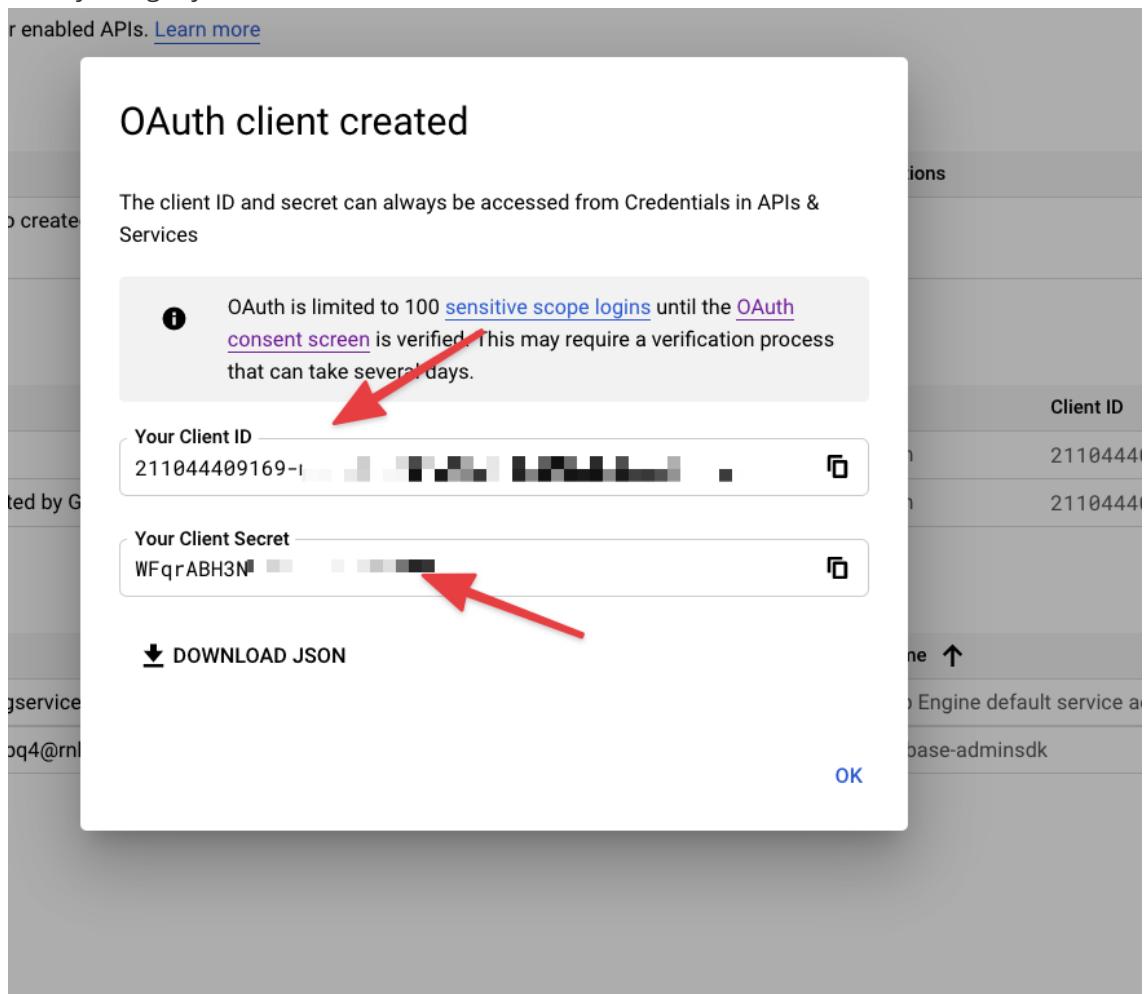
Authorized redirect URIs ?

For use with requests from a web server

URIs 1 \* https://yourdomain.com/api/auth/callback/google

+ ADD URI

6. Then you'll get your credentials,



7. Then open `api/.env` and add API ID and credentials,

```
51 LIGHTHOUSE_CACHE_ENABLE=false
52 MEDIA_DISK=public
53
54 SHOP_URL=
55 DASHBOARD_URL=
56 ADMIN_EMAIL=support@example.com
57
58 GOOGLE_CLIENT_ID= ← 2
59 GOOGLE_CLIENT_SECRET= ← 3
60 GOOGLE_REDIRECT_URI=http://127.0.0.1/login/google/callback
61
62 FACEBOOK_CLIENT_ID=
63 FACEBOOK_CLIENT_SECRET=
64 FACEBOOK_REDIRECT_URI=https://127.0.0.1/login/facebook/callback
65
66 ACTIVE_OTP_GATEWAY=twillio
67
68 TWILIO_AUTH_TOKEN=
69 TWILIO_ACCOUNT_SID=
70 TWILIO_VERIFICATION_SID=
71
72 MESSAGEBIRD_API_KEY=
73 MESSAGEBIRD_ORIGINATOR=Marvel
74
```

8. Then go to this `https://generate-secret.now.sh/32` site and generate a secret key.

9. Then open `shop/.env` and add credentials to `GOOGLE_CLIENT_ID` and `GOOGLE_CLIENT_SECRET` also add a `SECRET` code.

```
1 NEXT_PUBLIC_REST_API_ENDPOINT=http://localhost/
2
3 NEXT_PUBLIC_SITE_URL=http://localhost:3000
4
5 NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY=
6
7 # Social Authentication
8 NEXTAUTH_URL=http://localhost:3003
9
10 SECRET= # Linux: `openssl rand -hex 32` or go to https://generate-secret.now.sh/32 ←
11
12
13 GOOGLE_ID= ←
14 GOOGLE_SECRET= ←
15
16 FACEBOOK_ID=
17 FACEBOOK_SECRET=
18
19 NEXT_PUBLIC_GOOGLE_MAP_API_KEY=ADD_GOOGLE_MAP_API_KEY
20
21 NEXT_PUBLIC_INSTAGRAM_BASIC_DISPLAY_USER_TOKEN=your_instagram_basic_display_user_token_here
```

After configuration, make sure you rebuild your project using this command,

For REST API

```
yarn build:shop-rest
yarn build:admin-rest
```

## Facebook

1. Go to Facebook developer dashboard (<https://developers.facebook.com/apps/>)

2. Then create a new app for chawkbazar

3. After that, set up Facebook Login.

The screenshot shows the PickBazar dashboard with the 'Facebook Login' product selected. A red arrow points to the 'Set Up' button in the top-left corner of the 'Facebook Login' card. Below the cards, there is a message: 'Use the Quickstart to add Facebook Login to your app. To get started, select the platform for this app.' Four platform icons are shown: iOS (blue), Android (green), Web (dark blue), and Other (grey).

4. Then provide all the necessary information.

5. After that, go to the settings page, and on that page, you'll get all the credentials.

6. Then open `api/.env` and add API ID and credentials,

The screenshot shows two windows side-by-side. On the left is the 'Settings' page for the 'Basic' configuration of the 'PickBazar' app. It displays the App ID (220859506492814) and App Secret (redacted). A red arrow points to the App ID field. On the right is a code editor showing the contents of the `shop/.env` file. The file contains several environment variables, including `FACEBOOK_ID` and `FACEBOOK_SECRET`, which are highlighted with red arrows.

```

1 NEXT_PUBLIC_REST_API_ENDPOINT=http://localhost/
2
3 NEXT_PUBLIC_SITE_URL=http://localhost:3000
4
5 NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY=
6
7 # Social Authentication
8 NEXTAUTH_URL=http://localhost:3003
9
10 SECRET= # Linux: `openssl rand -hex 32` or go to https://generate-secret.now.sh/32
11
12 GOOGLE_ID=
13 GOOGLE_SECRET=
14
15 FACEBOOK_ID= ←
16 FACEBOOK_SECRET= ←
17
18
19 NEXT_PUBLIC_GOOGLE_MAP_API_KEY=ADD_GOOGLE_MAP_API_KEY
20
21 NEXT_PUBLIC_INSTAGRAM_BASIC_DISPLAY_USER_TOKEN=your_instagram_basic_display_user_token_here

```

7. Then go to this <https://generate-secret.now.sh/32> site and generate a secret key.

8. Then open `shop/.env` and add credentials to `FACEBOOK_CLIENT_ID` and `FACEBOOK_CLIENT_SECRET` also add a `SECRET` code.

This screenshot shows a code editor with two tabs: `api/.env` and `shop/.env`. The `api/.env` tab is active, displaying environment variables like `LIGHTHOUSE_CACHE_ENABLE` and `MEDIA_DISK`. The `shop/.env` tab shows the same variables as the previous screenshot, plus `FACEBOOK_CLIENT_ID` and `FACEBOOK_CLIENT_SECRET`, which are highlighted with red arrows. Both tabs have red arrows pointing to their respective `SECRET` variables.

```

51 LIGHTHOUSE_CACHE_ENABLE=false
52 MEDIA_DISK=public
53
54 SHOP_URL=
55 DASHBOARD_URL=
56 ADMIN_EMAIL=support@example.com
57
58 GOOGLE_CLIENT_ID=
59 GOOGLE_CLIENT_SECRET=
60 GOOGLE_REDIRECT_URI=http://127.0.0.1/login/google/callback
61
62 FACEBOOK_CLIENT_ID= ←
63 FACEBOOK_CLIENT_SECRET= ←
64 FACEBOOK_REDIRECT_URI=https://127.0.0.1/login/facebook/callback
65
66 ACTIVE_OTP_GATEWAY=twilio
67
68 TWILIO_AUTH_TOKEN=
69 TWILIO_ACCOUNT_SID=
70 TWILIO_VERIFICATION_SID=
71
72 MESSAGEBIRD_API_KEY=
73 MESSAGEBIRD_ORIGINATOR=Marvel
74

```

For Facebook, make sure you apply verification for the APP. Otherwise, the Facebook login won't work. (<https://developers.facebook.com/docs/app-review/>)

After configuration, make sure you rebuild your project using this command,

For REST API

```
yarn build:shop-rest  
yarn build:admin-rest
```

## OTP (Mobile Number Verification)

We implement the `OTP` feature on ChawkBazar `v2.2.0`. So if you want to use `OTP` with `chawkBazar`, then make sure your ChawkBazar is `v2.2.0` or later.

With the OTP feature, you can do,

- Login with Mobile Number
- OTP verification for updating Mobile Number in profile update section
- OTP verification during checkout process in customer contact section

## Configuration

As of today, we implement two service providers for OTP. One is `Twilio`, and another is `MessageBird`.

### Twilio Configuration:

If you want to use `Twilio` as your OTP service provider, then follow this procedure,

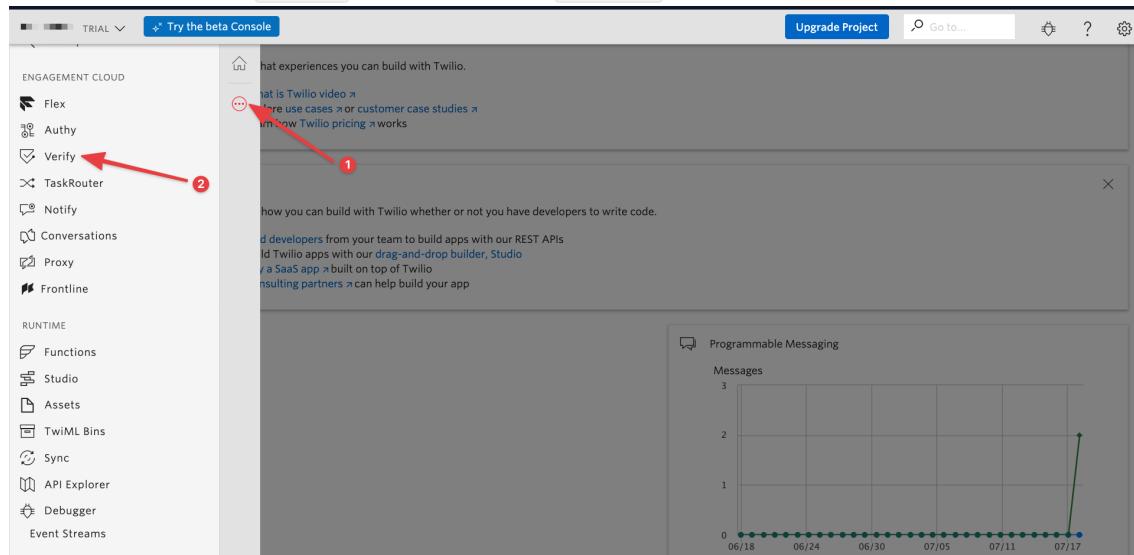
1. Create an account at [Twilio](#)
2. After that, go to the `Twilio Console Panel` and copy the `ACCOUNT SID`, and `AUTH TOKEN` key and add this key to your `api/.env` file

```

LICHTHOUSE_CACHE_ENABLE=false
MEDIA_DISK=public
SHOP_URL=
DASHBOARD_URL=
ADMIN_EMAIL=support@example.com
GOOGLE_CLIENT_ID=
GOOGLE_CLIENT_SECRET=
GOOGLE_REDIRECT_URI=http://127.0.0.1/login/google/callback
FACEBOOK_CLIENT_ID=
FACEBOOK_CLIENT_SECRET=
FACEBOOK_REDIRECT_URI=https://127.0.0.1/login/facebook/callback
ACTIVE_OTP_GATEWAY=twilio
TWILIO_AUTH_TOKEN=
TWILIO_ACCOUNT_SID=
TWILIO_VERIFICATION_SID=
MESSAGEBIRD_API_KEY=
MESSAGEBIRD_ORIGINATOR=Marvel

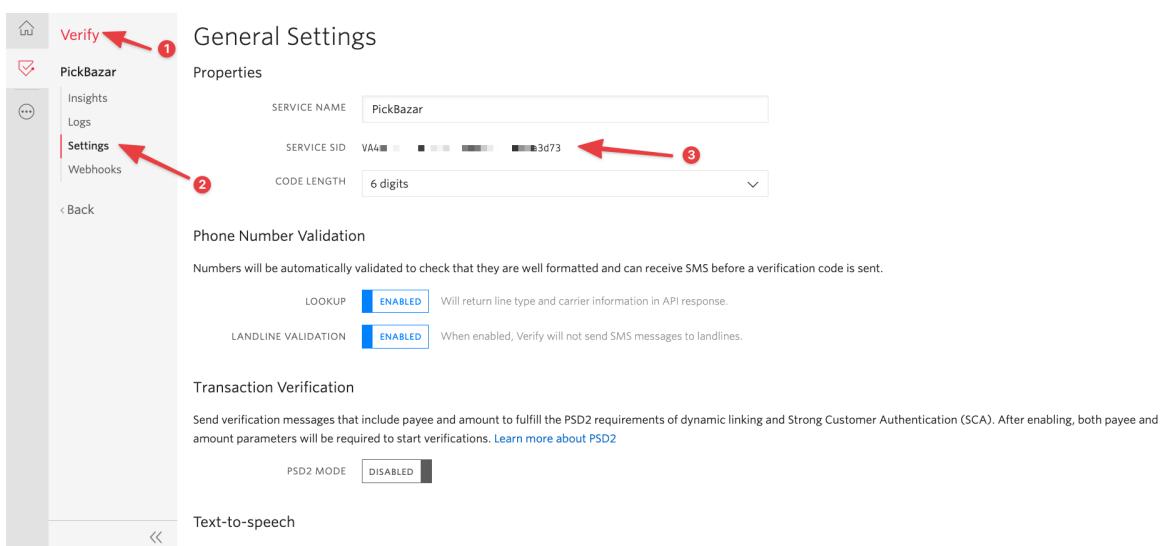
```

3. After that, go to the `Verify` option from the `Sidebar` panel,



4. And from `verify dashboard`, create a service.

5. Add add that service key to `api/.env -> TWILIO_VERIFICATION_SID`



```

LICHTHOUSE_CACHE_ENABLE=false
MEDIA_DISK=public
SHOP_URL=
DASHBOARD_URL=
ADMIN_EMAIL=support@example.com
GOOGLE_CLIENT_ID=
GOOGLE_CLIENT_SECRET=
GOOGLE_REDIRECT_URI=http://127.0.0.1/login/google/callback
FACEBOOK_CLIENT_ID=
FACEBOOK_CLIENT_SECRET=
FACEBOOK_REDIRECT_URI=https://127.0.0.1/login/facebook/callback
ACTIVE_OTP_GATEWAY=twilio ←
TWILIO_AUTH_TOKEN= ←
TWILIO_ACCOUNT_SID= ←
TWILIO_VERIFICATION_SID=
MESSAGEBIRD_API_KEY=
MESSAGEBIRD_ORIGINATOR=Marvel

```

With that, your configuration is complete.

## MessageBird:

Use only one service provider for OTP service, so use this `MessageBird` configuration only if you want to use `MessageBird` instead of `Twilio`.

To configure `MessageBird`, follow this procedure,

1. Create an account from [MessageBird](#)
2. After creating the account, go to the [API Getting Started](#) and follow all the procedures step by step.

**API Getting Started**

**Your getting started guide**

SMS API • 4 steps left 20%

- 1 Create your account
- 2 Get started with free test credits
- 3 Install the official SDK for SMS
- 4 Send your first SMS
- 5 Go live with SMS

3. After completing the profile, copy the `API` key from `MessageBird` and paste it to `API/.env`

The screenshot shows the Twilio developer console interface. On the left, there's a sidebar with navigation links like Contacts, Numbers, SMS (API Getting Started, SMS Overview, Campaign Builder, Quick Send, Bulk Messages), Voice, WhatsApp, Verify, Channels, Flow Builder, Inbox, Solutions, and Knowledge Bases. Below the sidebar are cookie settings and a "Cookie Settings" link.

The main area is titled "Your getting started guide" and shows a progress bar for the SMS API with 4 steps left and 20% completion. The steps are:

- 1 Create your account
- 2 Get started with free test credits
- 3 Install the official SDK for SMS
- 4 Send your first SMS
- 5 Go live with SMS

To the right, there's a section titled "Your API Keys" with two entries: "Live" and "Test". Each entry has a "Show" button. A red arrow points to the "Show" button for the "Live" key. Below the keys are links for "Manage API Keys" and "Resources" (SMS API Quickstarts and SMS API Tutorials).

The screenshot shows a code editor with a file named ".env" open. The file contains environment variables for a Laravel application, including:

```

LIVESHOW_CACHE_ENABLE=false
MEDIA_DISK=public
SHOP_URL=
DASHBOARD_URL=
ADMIN_EMAIL=support@example.com
GOOGLE_CLIENT_ID=
GOOGLE_CLIENT_SECRET=
GOOGLE_REDIRECT_URI=http://127.0.0.1/login/google/callback
FACEBOOK_CLIENT_ID=
FACEBOOK_CLIENT_SECRET=
FACEBOOK_REDIRECT_URI=https://127.0.0.1/login/facebook/callback
ACTIVE_OTP_GATEWAY=messagebird ←
TWILIO_AUTH_TOKEN=
TWILIO_ACCOUNT_SID=
TWILIO_VERIFICATION_SID=
MESSAGEBIRD_API_KEY= ←
MESSAGEBIRD_ORIGINATOR=Marvel
    
```

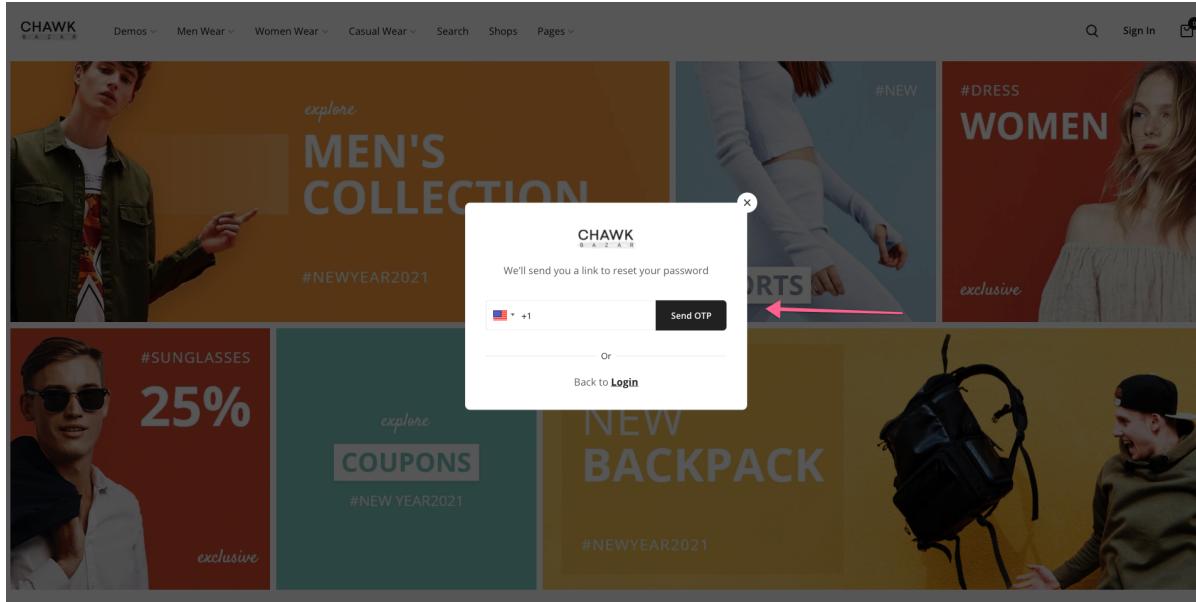
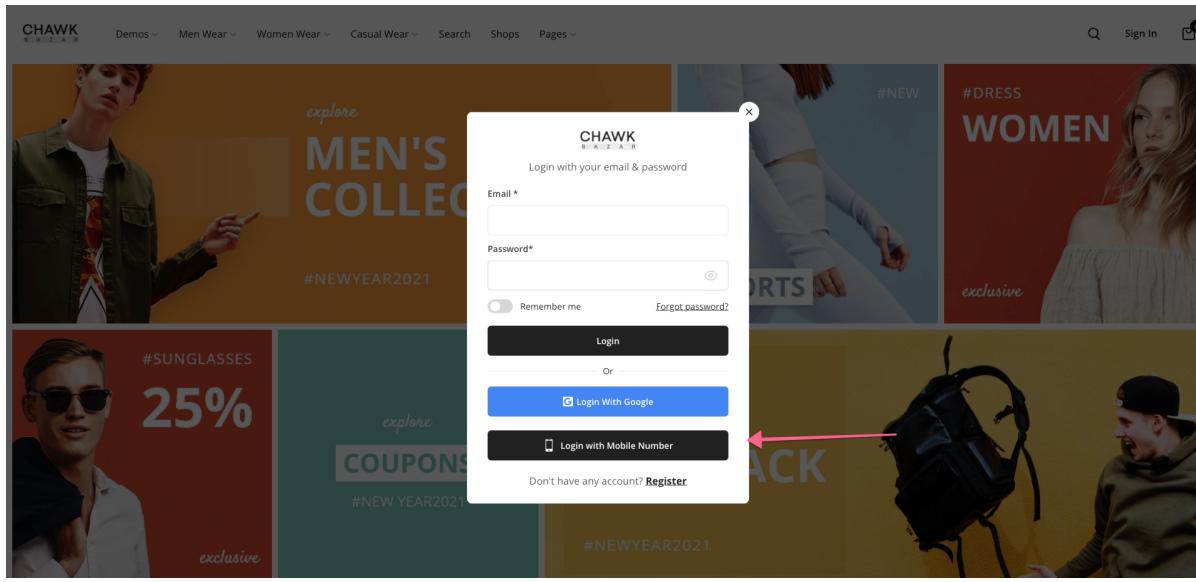
Red arrows point to the environment variables for the OTP gateway, Twilio authentication token, and MessageBird API key.

With that, `MessageBird` is appropriately configured for the OTP Service.

## FrontEnd Demo

### Login with Mobile Number:

After configuration, you will be able to use `mobile number` as a login provider. When users select the `Join` button, they'll get an option `Login with mobile` to use a mobile number as a login provider.



## OTP Verification During Checkout Process:

Similarly, when users go to the checkout page, they'll get a form to verify their mobile number.

For the front end, you don't have to do anything. When a user goes to the `checkout` page, they get a phone number box to add their number and verify it.

Product	Subtotal
3 x Zara Army Bag   1 pc	\$780.00
4 x Pior Womes Bangles   1 pcs	\$4,600.00
<b>Subtotal</b>	<b>\$5,380.00</b>
Tax	Calculated at checkout
Shipping	Calculated at checkout

## OTP Verification For Updating Mobile Number:

Users also can change the number from their profile.

The screenshot shows the Chawkbazar dashboard with a sidebar on the left containing links: Dashboard, Orders, Address, Contact Number (highlighted with a red arrow), Change Password, and Logout. The main area is titled 'Contact Number' and displays the mobile number '12365141641631'. A red arrow points to the '+ Update' button in the top right corner of this section.

## Email Verification

We implement the `Email Verification` feature on Chawkbazar `v5.1.0`. So if you want to use `Email Verification` with Chawkbazar, then make sure your Chawkbazar is `v5.1.0` or later.

With the Email Verification feature, you can do,

- Get Email After Registration
- Email verification for updating Email in profile update section
- When Email verification is on in settings then users will be forced to verify their email.

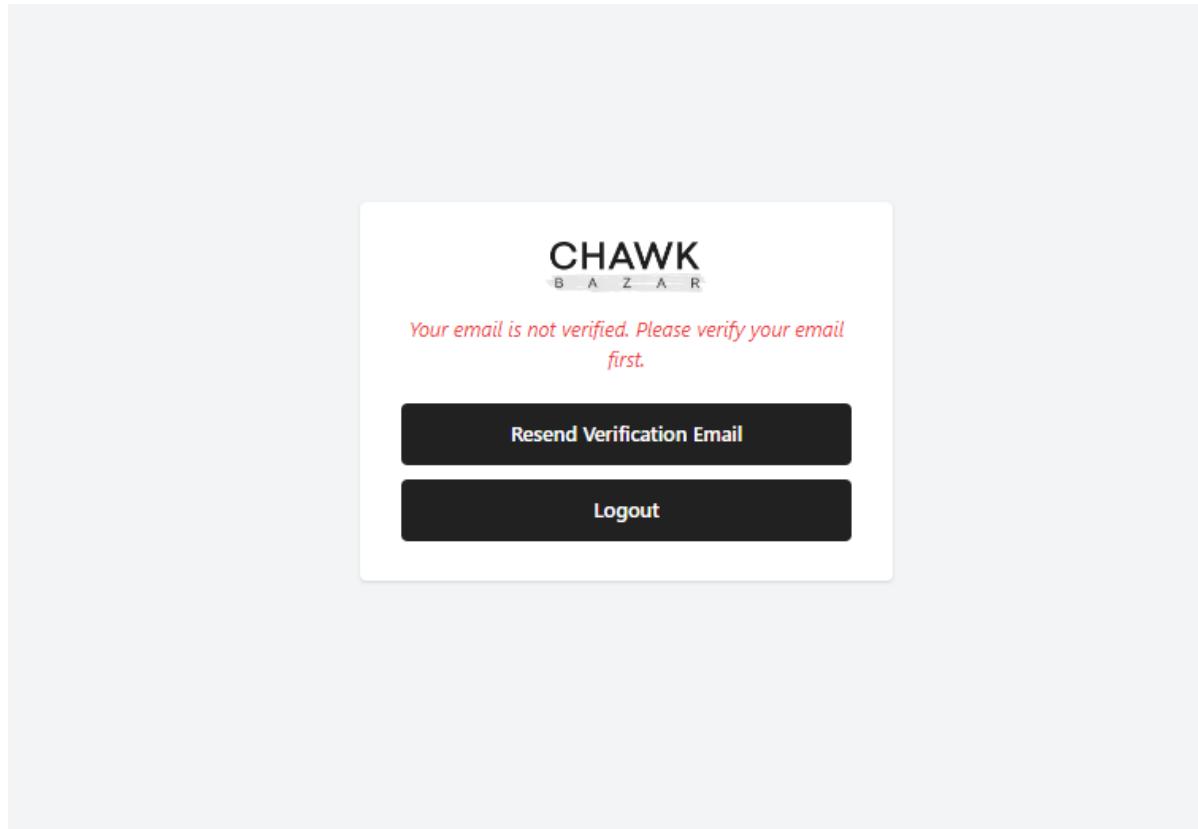
## How to turn on this feature

To turn on this feature go to settings of your site then find `Enable Must Verify Email` switch.

The screenshot shows the Chawkbazar site settings page. It includes fields for Site Title (ChawkBazar), Site Subtitle (Your next ecommerce), Minimum Order Amount (0), and three toggle switches: 'Use OTP at checkout' (off), 'Enable Must Verify Email' (off, highlighted with a red box), and 'Enable AI' (off). There is also a 'Select AI' dropdown menu set to 'openai'.

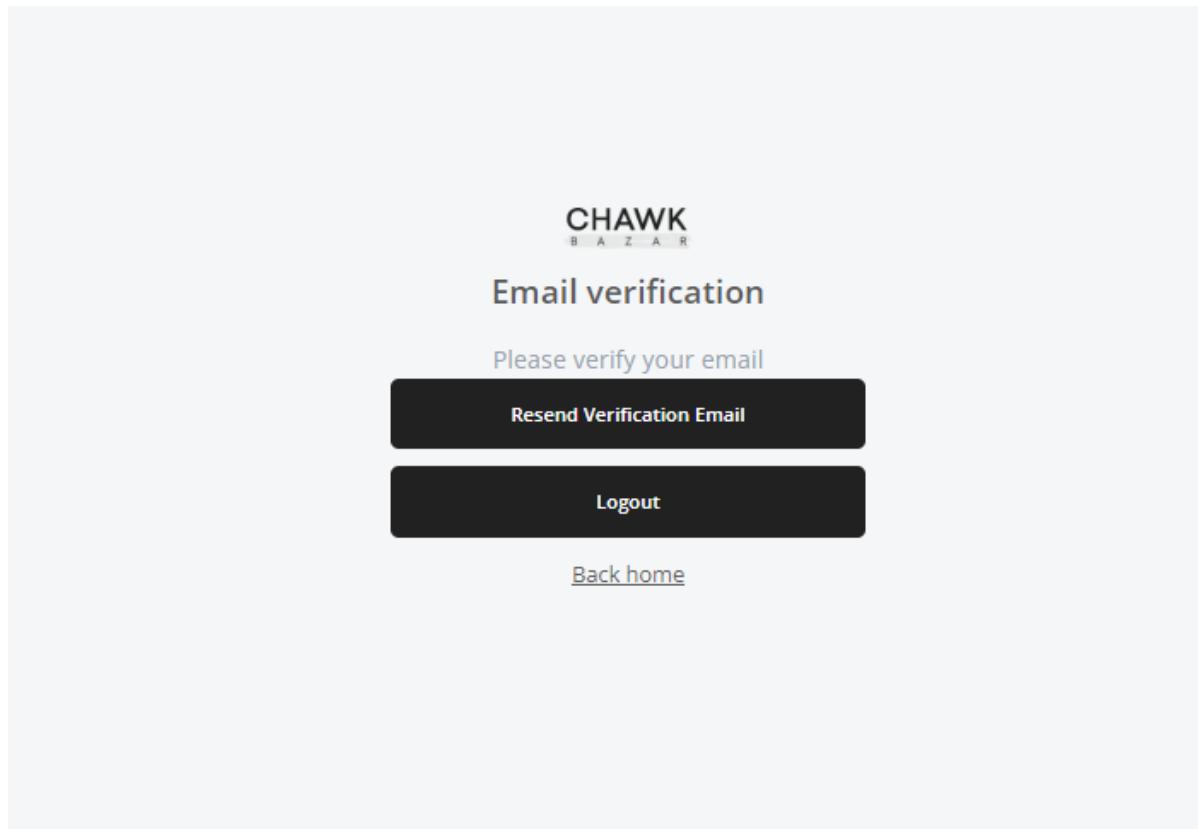
## Admin view of this feature

---



## Shop view of this feature

---



## Configuration

---

To use this feature all you need to configure mail in your application.

```
MAIL_MAILER=mailgun
MAILGUN_DOMAIN=
MAILGUN_SECRET=
MAIL_FROM_ADDRESS=
MAIL_PORT=1025
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_ENCRYPTION=null
ADMIN_EMAIL=support@example.com
```

# Payment

We have introduced the new `Payment` architecture feature on ChawkBazar `v3.0.0`. So if you want to use `Payment` with `ChawkBazar`, then make sure your ChawkBazar is `v3.0.0` or later.

We have plan to enrich this feature area by integrating more payment gateways in future. List of available payment gateways now.

- **Stripe**
- **PayPal**
- **Rezorpay**
- **Mollie**
- **Paystack**

Let's discuss those sequentially.

## Stripe

Stripe is a financial service & Software as a service (SaaS) company. It offers payment processing software & API for e-commerce applications. In ChawkBazar we have integrated Stripe API for payment system. Though currently `Card` based features are available only, `Stripe Element` for other payment options will be integrate in future updates.

### Stripe integrate inside ChawkBazar.

Please follow & complete this steps for stripe integration for your e-commerce system.

- Inside api there is `.env` file. You have to copy & paste this line of codes inside the `.env` file. We will discuss later about how to create those API keys in stripe.

```
STRIPE_API_KEY=[YOUR_STRIPE_API_SECRET_KEY]
STRIPE_WEBHOOK_SECRET_KEY=[YOUR_STRIPE_API_WEBHOOK_SECRET_KEY]
```

- To activate Stripe go to settings from ChawkBazar admin dashboard. Inside settings you will find configure payment option.(e.g. webhook URL is coming from local development. This static link will dynamically generated in live environment)

- Add Stripe publishable key inside `.env` file of ChawkBazar shop.

```
NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY=[YOUR_STRIPE_PUBLISHABLE_KEY]
```

## How to create & setup Stripe information properly?

- Go to the stripe website and login <https://stripe.com> If you aren't a registered user, the complete the stripe registration first.
- After logged in into stripe dashboard, follow the developer link to create the API keys.

- Then create API keys from there. for more details follow up this official documentation [http://stripe.com/docs/keys](https://stripe.com/docs/keys)

The screenshot shows the Stripe Developers API keys page. The sidebar on the left has tabs for Overview, API keys (which is selected and highlighted with a red arrow), Webhooks, Events, Logs, and Apps. The main content area is titled 'API keys' and contains a table with several rows of API key information. At the top right of the main area, there are buttons for 'Learn more about API authentication' and 'Viewing test data'.

- Create Webhook secret key if you decide to up & running webhooks in your App.

The screenshot shows the Stripe Developers Webhooks page. The sidebar on the left has tabs for Overview, API keys, Webhooks (selected and highlighted with a red arrow), Events, Logs, and Apps. The main content area is titled 'WEBHOOK' and shows a single webhook configuration. It includes fields for Status (Disabled), Listening for (2 events), API version (2018-08-23), Signing secret (with a red arrow pointing to it), Configuration, and View logs. The URL for the webhook is https://a6ce-103-60-175-8.ngrok.io/webhooks/stripe.

- Create this two webhook events for monitoring the payment flow.

- **payment\_intent.succeeded**
- **payment\_intent.payment\_failed**

## Special Notes for Stripe users.

If we have used any third party system/plugin/packages, then we have always encouraged our respected customers to follow the official documentation for detailed & in-depth knowledge.

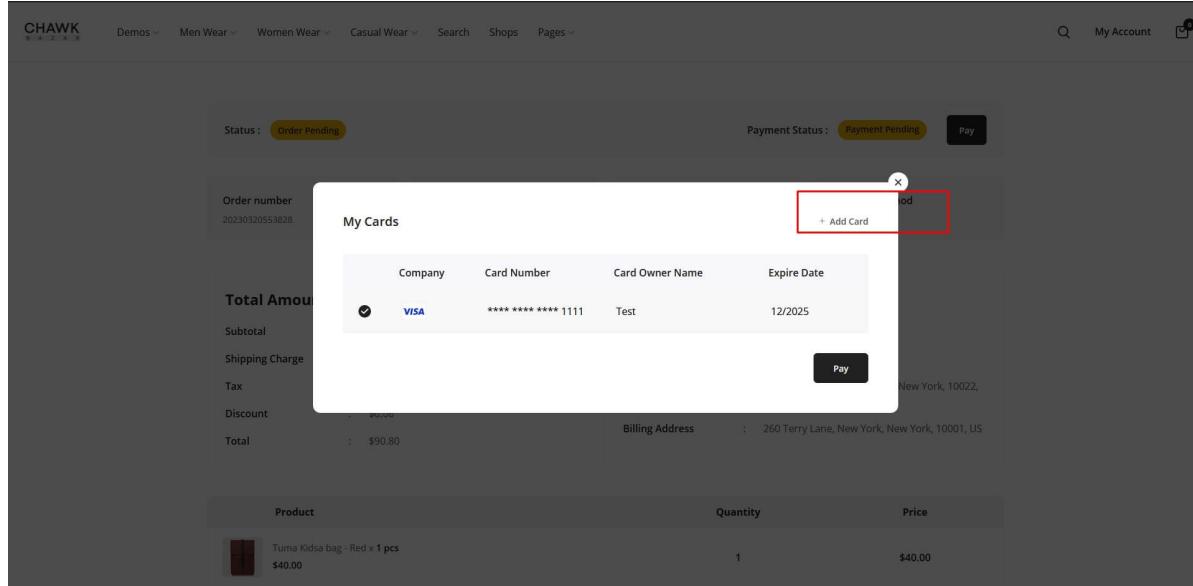
- Stripe DOCS [home page] -- <https://stripe.com/docs>
- Stripe API Docs -- <https://stripe.com/docs/api>
- Stripe JavaScript Docs -- <https://stripe.com/docs/js>

# How can I add card in my user profile for future payments in Stripe?

In ChawkBazar we have provided an feature for saving card in case of future usages. This is an `on-session` process. So, if a customer wants to pay via Cards, s/he must have present in the application lively. No `off-session` payment was applied here.

A customer can save a card via two process.

- Save card during checkout process.

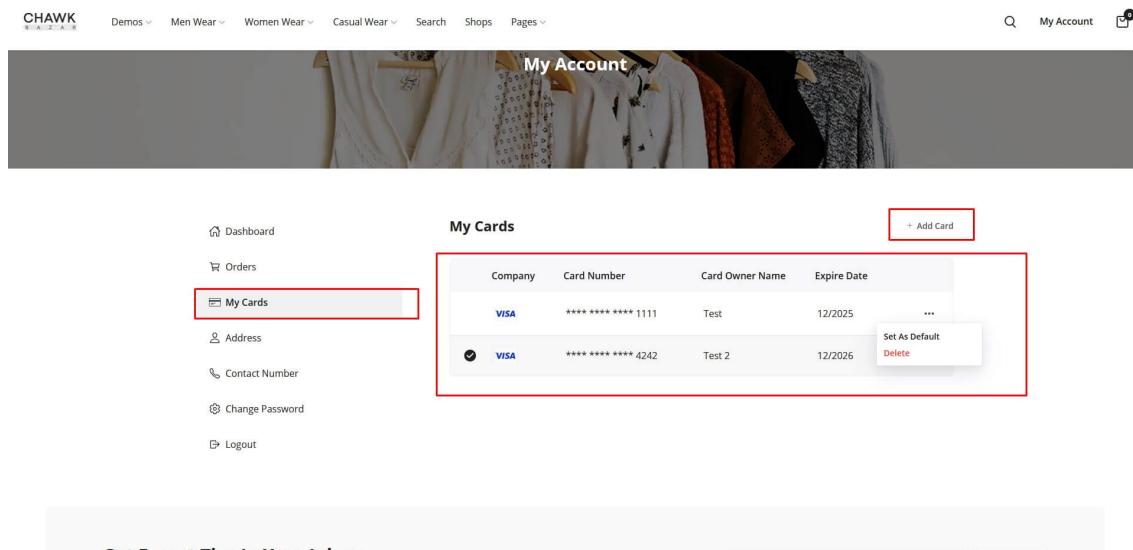


The screenshot shows a checkout page for an order with an order number of 20230320553828. The total amount is \$90.80. A modal window titled "My Cards" is displayed, containing a table with one card entry:

Company	Card Number	Card Owner Name	Expire Date
VISA	***** * 1111	Test	12/2025

There is a red box around the "Add Card" button in the top right corner of the modal. Below the modal, there is a "Pay" button.

- Save card in his user profile `My Card` section.



The screenshot shows the "My Account" page with a sidebar menu. The "My Cards" option is highlighted with a red box. The main content area shows a table of saved cards:

Company	Card Number	Card Owner Name	Expire Date	Actions
VISA	***** * 1111	Test	12/2025	... Set As Default Delete
VISA	***** * 4242	Test 2	12/2026	Set As Default Delete

There is a red box around the "Delete" link for the second card entry.

There are something needs to keep in mind.

## Please note

- No confidential information was saved in this features. By maintaining the guideline only available information which are permitted to save via Stripe is implemented here.
- Guest user can't save card for future payment.

# PayPal

PayPal is an American multinational financial technology company operating an online payments system. In ChawkBazar we have integrated PayPal APIs which may cover a vast area of PayPal supported region. It will help your business to grow and reach to a wide region.

## PayPal integrate inside ChawkBazar.

Please follow & complete this steps for PayPal integration for your e-commerce system.

- Inside api there is `.env` file. You have to copy & paste this line of codes inside the `.env` file. We will discuss later about how to create those API keys in PayPal.

```
# Values: sandbox or live (Default: live)
PAYPAL_MODE=sandbox

# Add currency like USD
PAYPAL_CURRENCY=USD

# Change this accordingly for your application.
PAYPAL_NOTIFY_URL=

# force gateway language i.e. it_IT, es_ES, en_US ... (for express checkout
only)
PAYPAL_LOCALE=en

#Validate SSL when creating api client.
PAYPAL_VALIDATE_SSL=

# PayPal Setting & API Credentials -> sandbox

PAYPAL_SANDBOX_CLIENT_ID=[YOUR_PAYPAL_SANDBOX_CLIENT_ID]
PAYPAL_SANDBOX_CLIENT_SECRET=[YOUR_PAYPAL_SANDBOX_CLIENT_SECRET_KEY]

# PayPal Setting & API Credentials -> live

PAYPAL_LIVE_CLIENT_ID=[YOUR_PAYPAL_LIVE_CLIENT_ID]
PAYPAL_LIVE_CLIENT_SECRET=[YOUR_PAYPAL_LIVE_CLIENT_SECRET_KEY]

# PayPal Webhook settings

PAYPAL_WEBHOOK_ID=[YOUR_PAYPAL_WEBHOOK_URL]
SHOP_URL=[YOUR_SHOP_URL]
```

**SHOP\_URL=[YOUR\_SHOP\_URL]** This parameter is must have in `.env` file when PayPal is using. Otherwise the payment redirection will be broken.

- To activate Paypal go to settings from ChawkBazar admin dashboard. Inside settings you will find configure payment option. (e.g. webhook URL is coming from local development. This static link will dynamically generated in live environment)

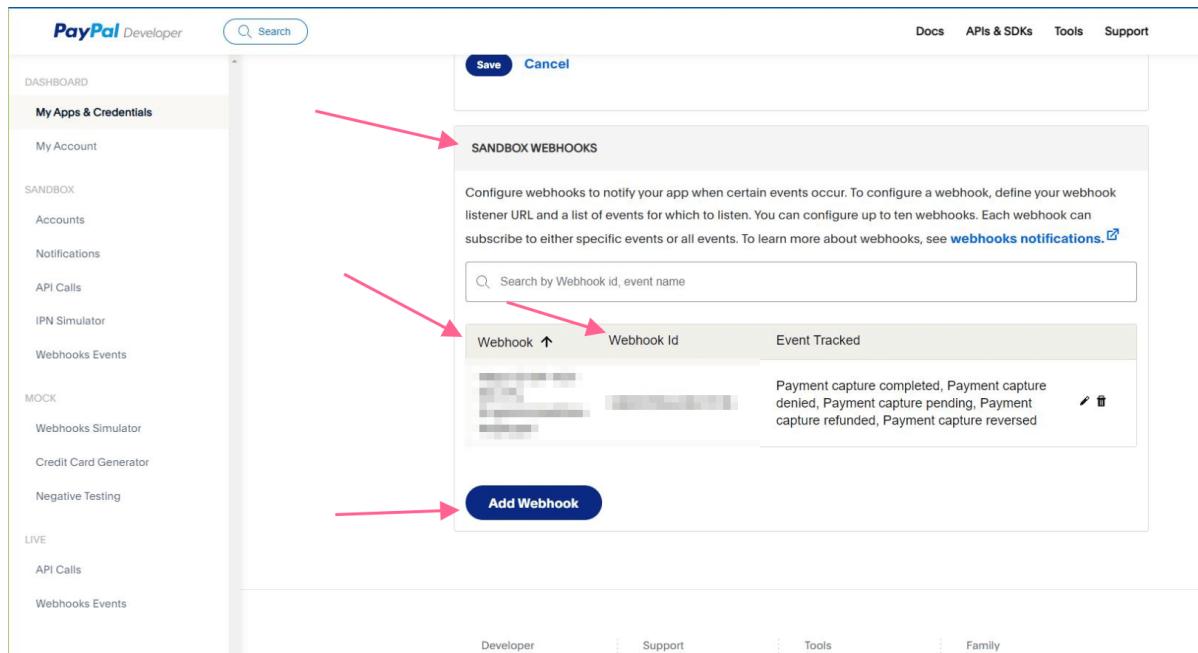
The screenshot shows the ChawkBazar dashboard with a sidebar containing various settings like Dashboard, Shops, My Shops, Products, Attributes, Brands, Categories, Tags, Orders, Users, Coupons, Taxes, Shipments, Withdrawals, and Settings. The 'Settings' tab is highlighted with a red box. In the main content area, there are four cards: Total Revenue (\$0.00), Total Order (9), Today's Revenue (\$0.00), and Total Shops (2). Below these is a Sales History chart from January to December. Under the 'Recent Orders' and 'Recent Withdrawals' sections, there is sample data. The 'Payment' section is expanded, showing a toggle for 'Enable Cash On Delivery', a dropdown for 'Select Payment Gateway' (set to 'Paypal'), and a text input for 'Webhook URL' with the value 'https://marvelmockapi.redq.io/chawkbazar-qa/webhooks/paypal'.

## How to create & setup PayPal information properly?

- Create an account in <https://developer.paypal.com>
- Choose sandbox if your are testing your development environment.
- Click on `Default Application`. or create a new App.
- You will get your Sandbox API credentials. Copy & paste those inside ChawkBazar app .env file like mentioned above.

The screenshot shows the PayPal Developer dashboard. On the left, there's a sidebar with links like DASHBOARD, My Apps & Credentials (which is highlighted with a red arrow), My Account, SANDBOX (Accounts, Notifications, API Calls, IPN Simulator, Webhooks Events), MOCK (Webhooks Simulator, Credit Card Generator, Negative Testing), and LIVE (API Calls, Webhooks Events). The main panel is titled 'Default Application' and shows 'App display name: Default Application'. It contains sections for 'SANDBOX API CREDENTIALS' (highlighting 'Sandbox Account', 'Client ID', and 'Secret' with red arrows), 'SANDBOX APP SETTINGS' (highlighting 'Return URL' and 'App feature options' with red arrows), and a note about accepting payments.

- You will find the Webhook ID and link too.



- Then Add the webhook events if you want to up & running the services.
  - **PAYMENT.CAPTURE.COMPLETED**
  - **PAYMENT.CAPTURE.PENDING**
  - **PAYMENT.CAPTURE.CANCELLED**
  - **PAYMENT.CAPTURE.REVERSED**
- At last, For going live with your application please follow this official documentation. <https://developer.paypal.com/docs/archive/paypal-here/sdk-dev/going-live/>

## RazorPay

Razorpay is the only payments solution in India that allows businesses to accept, process and disburse payments with its product suite. It gives you access to all payment modes including credit card, debit card, netbanking, UPI and popular wallets including JioMoney, MobiKwik, Airtel Money, FreeCharge, Ola Money and PayZapp.

### RazorPay integration inside PickBazar.

Please follow & complete this steps for RazorPay integration with your e-commerce system.

- Inside api there is `.env` file. You have to copy & paste this line of codes inside the `.env` file. We will discuss later about how to create those API keys in RazorPay.

```
RAZORPAY_KEY_ID=[YOUR_RAZORPAY_KEY_ID]
RAZORPAY_KEY_SECRET=[YOUR_RAZORPAY_KEY_SECRET]
RAZORPAY_WEBHOOK_SECRET_KEY=[YOUR_RAZORPAY__WEBHOOK_URL]
```

- Activate RazorPay from PickBazar admin dashboard. (e.g. webhook URL is coming from local development. This static link will dynamically generated in live environment)

The screenshot shows the CHAWK BAZAAR dashboard. On the left, there's a sidebar with links for Dashboard, Shops, My Shops, Products, Attributes, Brands, Categories, Tags, Orders, Users, Coupons, Taxes, Shipments, Withdrawals, and Settings. The 'Settings' link is highlighted with a red box. The main area displays four cards: 'Total Revenue (Last 30 Days)' with '\$0.00', 'Total Order (Last 30 Days)' with '9', 'Today's Revenue' with '\$0.00', and 'Total Shops' with '2'. Below these are sections for 'Sales History' (a line chart from January to December) and 'Recent Orders' (a table with one row: Tracking Number 20230318657888, Total \$611.00, Order Date 2 days ago, Status Order Processing). Another section shows 'Recent Withdrawals' (a table with one row: Shop Name Chawkbazar Vendor shop, Amount \$500.00, Status Approved, Created a year ago). A large central box contains settings for payment: 'Enable Cash On Delivery' (switch on), 'Select Payment Gateway' (RazorPay selected), and 'Webhook URL' (http://localhost:8000/webhooks/razorpay).

- Add RazorPay publishable key inside `.env` file of pickbazar shop.

```
NEXT_PUBLIC_RAZORPAY_KEY=[YOUR_RAZORPAY_KEY_ID]
```

## How to create & setup RazorPay information properly?

- Go to the RazorPay website & login <https://razorpay.com> If you aren't a registered user, Complete the RazorPay registration first.
- After logged in into RazorPay dashboard, Click on Settings

The screenshot shows the RazorPay dashboard. The sidebar has options: Payment Button, Route, Subscriptions, QR Codes (NEW), Smart Collect, Customers, Offers, Checkout Rewards, Reports, My Account (NEW), and Settings (highlighted with a red arrow). The main area shows 'YOU'RE IN TEST MODE' and 'Test Mode'. It includes sections for 'Theme Color' (set to #528FF0), 'Your Logo' (choose file), 'Default Language' (set to English), and 'SELECT A PAYMENT METHOD' (Card, Netbanking, Wallet, Upi, Emi, Qr). A note at the bottom says 'Changes will reflect on Checkout page, Payment Links, Invoices & Payment pages.'

- Select API Keys option & click on Generate Test Key.

The screenshot shows the Razorpay dashboard with the 'API Keys' tab selected. The page displays a table with one row containing the Key Id 'rzp\_test\_5NQGiVrDXbrP8V', Created At 'Dec 31st, 2022 09:12:07 AM', Expiry 'Never', and Action 'Regenerate Test Key'. A red arrow points to the 'Regenerate Test Key' button.

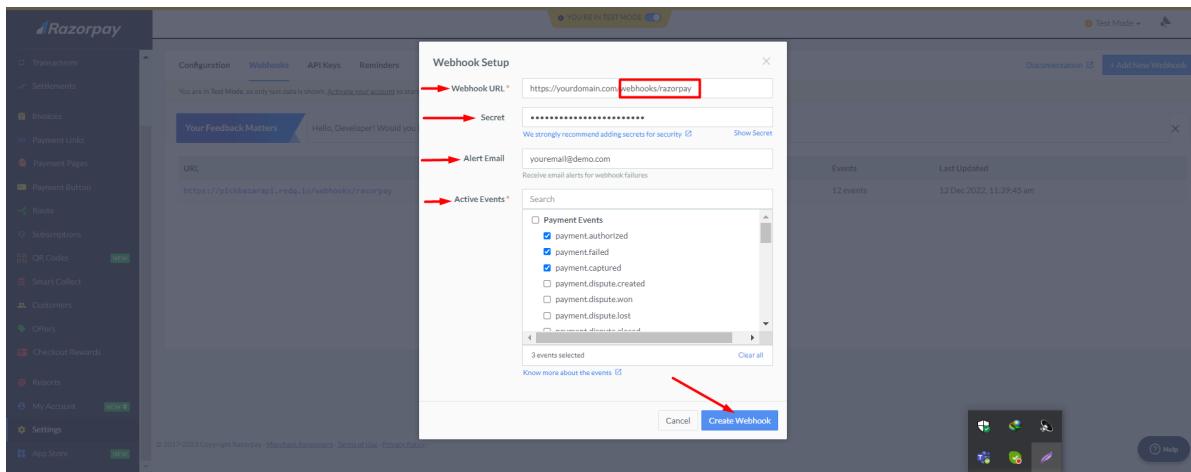
- The Client Id and Secret Key will appear. Download these keys and click on OK

The screenshot shows the 'New Key' modal dialog. It contains fields for 'Key Id' (rzp\_test\_FpKKOVu8jOdI) and 'Key Secret' (yPqipD1NfVSTSxbG23C). A red arrow points to the 'Download Key Details' button, and another red arrow points to the 'OK' button.

- Select Webhooks option & click on Add New Webhooks.

The screenshot shows the 'Webhooks' page with the 'Webhooks' tab selected. A red arrow points to the '+ Add New Webhook' button.

- Enter Webhook URL , Secret key , Alert Email and tick mark the checkbox in Active Event then click on create Webhook



- Copy & paste those inside PickBazar app .env file like mentioned above. after that test your RazorPay Payment Status

Payment Id	Razorpary Order Id	Amount	Email	Contact	Created At	Status
pay_KzaX5rehTtoAUP	order_KzawXR85J715b	\$ 53.06	customer@demo.com	+19365141641631	03 Jan 2023, 11:25:35 am	Captured
pay_KzaJu3D4Pk1CLe	order_KzaJ6rkG0tSLYD	\$ 54.90	customer@demo.com	+19365141641631	03 Jan 2023, 11:13:08 am	Captured
pay_KxcEcmyvDA3ikN	order_KxcEBpxZwKzex9	€ 30.45	customer@demo.com	+19365141641631	29 Dec 2022, 11:47:29 am	Captured
pay_KxcCMYveu8DQpw	order_KxcBpJDJu6J0ho	€ 57.75	customer@demo.com	+19365141641631	29 Dec 2022, 11:45:21 am	Captured
pay_KzbmIBiwL9qqQp	order_KzbImqCzs1Mx5q	€ 30.45	customer@demo.com	+19365141641631	29 Dec 2022, 11:20:40 am	Captured
pay_Kbcbtk0ivvRrq1H	order_KbcbtdaPDZ1qdg	€ 19.95	customer@demo.com	+19365141641631	29 Dec 2022, 11:19:40 am	Captured
pay_KzbWPQwSH6k2nU	order_KzbVDOFLAxCvor	€ 51.45	customer@demo.com	+19365141641631	29 Dec 2022, 11:19:40 am	Captured

## Special Notes for RazorPay.

If we have used any third party system/plugin/packages, then we have always encouraged our respected customers to follow the official documentation for detailed & in-depth knowledge.

- RazorPay DOCS -- <https://razorpay.com/docs/#home-payments>
- RazorPay Webhooks -- <https://razorpay.com/docs/webhooks/setup-edit-payments>

## Mollie

Mollie is a Payment Service Provider (PSP) that processes online payments for companies. If you buy something online from one of our merchants, we make sure that your money is transferred safely from your bank to the merchant's bank. Since we arrange the payment process, you may see Mollie or Stg Mollie Payments on your bank statement.

## Mollie integrate inside PickBazar.

Please follow & complete this steps for Mollie integration for your e-commerce system.

- Inside api there is .env file. You have to copy & paste this line of codes inside the .env file. We will discuss later about how to create those API keys in Mollie.

```
MOLLIE_KEY=[YOUR_MOLLIE_API_KEY]
```

## Mollie Webhook settings

```
SHOP_URL=[YOUR_SHOP_URL]  
MOLLIE_WEBHOOK_URL=[YOUR_MOLLIE_WEBHOOK_URL]
```

**SHOP\_URL=[YOUR\_SHOP\_URL]** This parameter is must have in .env file when Mollie is using. Otherwise the payment redirection will be broken.

- To activate Mollie go to settings from PickBazar admin dashboard. Inside settings you will find configure payment option. (e.g. Mollie Webhook URL is coming from local development. This static link will dynamically generated in live environment)

The screenshot shows the PickBazar admin dashboard with a sidebar on the left containing links like Dashboard, Shops, My Shops, Products, Attributes, Brands, Categories, Tags, Orders, Users, Coupons, Taxes, Shipments, Withdrawals, and Settings. The Settings link is highlighted with a red box. The main content area displays various metrics such as Total Revenue (\$0.00), Total Order (9), Today's Revenue (\$0.00), and Total Shops (2). Below these are Sales History and Recent Orders tables. A modal window titled 'Payment' is open, showing options for enabling Cash On Delivery, selecting a payment gateway (Mollie), and entering a Webhook URL (http://localhost:8000/webhooks/mollie).

- Copy & paste those inside PickBazar app .env file like mentioned above.
- If you want to use webhook during development on localhost, you must use a tool like ngrok to have the webhooks delivered to your local machine.

```
- ngrok http http://localhost:8000
```

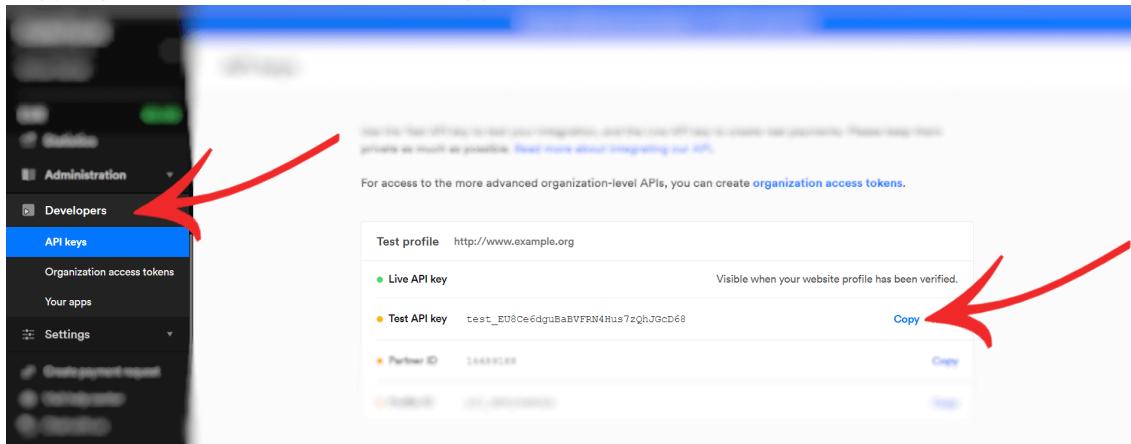
The terminal window shows the command '- ngrok http http://localhost:8000' being run. The output includes session details (Status: online, Account: [REDACTED], Version: 3.1.0, Region: India (in), Latency: 116ms, Web Interface: http://127.0.0.1:4040, Forwarding: https://c852-37-111-243-195.in.ngrok.io -> http://localhost:8000) and connection statistics (Connections: ttl 0, open 0, rtt1 0.00, rtt5 0.00, p50 0.00, p90 0.00).

- Copy the forwarding https link & paste .env file after link add line- /webhooks/mollie

```
84 # Payment -> Mollie
85 MOLLIE_KEY=test_EU8Ce6dguBaBVFRN4Hus7zQhJGcD68
86 MOLLIE_WEBHOOK_URL=https://c852-37-111-243-195.in.ngrok.io/webhooks/mollie
```

## How to create & setup Mollie information properly?

- Go to the Mollie website & login <https://www.mollie.com> If you aren't a registered user, the complete the Mollie registration first.
- After logged in into Mollie dashboard, follow the developer link to copy the API keys.
- Choose Test API key your are testing your development environment.
- Copy & paste those inside PickBazar app .env file like mentioned above.



## Special Notes for Mollie.

If we have used any third party system/plugin/packages, then we have always encouraged our respected customers to follow the official documentation for detailed & in-depth knowledge.

- Mollie DOCS -- <https://docs.mollie.com/>
- Mollie Webhooks -- <https://docs.mollie.com/overview/webhooks>

## Paystack

Paystack is a Nigerian fintech company that provides online payment solutions for businesses. It offers a range of services, including online and offline payments, subscriptions, and invoicing. Paystack's platform enables businesses to easily and securely accept payments from customers via credit card, debit card, and mobile money.

## Paystack integrate inside PickBazar.

Please follow & complete this steps for Paystack integration for your e-commerce system.

- First go to settings from PickBazar admin dashboard. Inside settings you will find Currency option. Select your Currency. If you want to Test development on localhost.then Select South African Rand(ZAR).

Information  
Change your site information from here

Site Title  
Pickbazar

Site Subtitle  
Your next ecommerce

Currency  
South African Rand

Minimum Order Amount  
0

Here is a breakdown of countries and the currencies they can accept payments in.

Country	Available Currencies
Ghana	GHS
Nigeria	NGN, USD
South Africa	ZAR

For more information: [Paystack currency is available.](#)

- Inside api there is `.env` file. You have to copy & paste this line of codes inside the `.env` file. We will discuss later about how to create those API keys in [Paystack official website](#).

```
PAYSTACK_PUBLIC_KEY=[YOUR_PAYSTACK_PUBLIC_KEY]
PAYSTACK_SECRET_KEY=[YOUR_PAYSTACK_SECRET_KEY]
PAYSTACK_PAYMENT_URL=https://api.paystack.co
MERCHANT_EMAIL=[YOUR_PAYSTACK_MERCHANT_EMAIL]
```

- Add Paystack publishable key inside pickbazar shop.

```
NEXT_PAYSTACK_PUBLIC_KEY=[YOUR_YOUR_PAYSTACK_PUBLIC_KEY]
```

**SHOP\_URL=[YOUR\_SHOP\_URL]** This parameter is must have in `.env` file when Paystack is using. Otherwise the payment redirection will be broken.

## Paystack Webhook settings

- To activate Paystack go to settings from PickBazar admin dashboard. Inside settings you will find configure payment option. (e.g. Paystack Webhook URL is coming from local development. This static link will dynamically generated in live environment)

The screenshot shows the CHAWK BAZAAR dashboard. On the left sidebar, under the 'Settings' section, there is a red box highlighting the 'Payment' option. The main content area displays various metrics: Total Revenue (\$0.00), Total Order (9), Today's Revenue (\$0.00), and Total Shops (2). Below these are sections for Sales History (a line chart from January to December) and Recent Orders (listing one order: Tracking Number 20230318657888, Total \$611.00, Order Date 2 days ago, Status Order Processing). Another section shows Recent Withdrawals (Shop Name Chawkbazar Vendor shop, Amount \$500.00, Status Approved, Created a year ago).

- Copy the webhook url & registered in Paystack Dashboard.

The screenshot shows the Paystack Dashboard settings page. It has two main input fields: 'Test Callback URL' and 'Test Webhook URL'. The 'Test Webhook URL' field contains the value 'https://pickbazarapi.com/webhooks/paystack'. A red arrow labeled '1' points from the 'Test Callback URL' field to the 'Test Webhook URL' field. A red arrow labeled '2' points from the 'Save Changes' button at the bottom right.

- If you want to use webhook during development on localhost, you must use a tools like ngrok to have the webhooks delivered to your local machine.

```
- ngrok http http://localhost:8000
ngrok
Check which logged users are accessing your tunnels in real time https://ngrok.com/s/app-users
Session Status      online
Account
Version          3.1.0
Region           India (in)
Latency          116ms
Web Interface
Forwarding       http://127.0.0.1:4040
https://c852-37-111-243-195.in.ngrok.io  > http://localhost:8000
Connections
  ttl     opn      rt1      rt5      p50      p90
    0       0     0.00     0.00     0.00     0.00
```

- Copy the forwarding https link & paste on Test Webhook URL, after paste forwarding https link, add line- /webhooks/paystack.

Test Callback URL

Test Webhook URL   

## How to create & setup Paystack information properly?

- Go to the [Paystack official website](#) & login If you aren't a registered user, Complete the Paystack registration first.
- After logged in into Paystack dashboard, Click on Settings.

The screenshot shows the Paystack dashboard. On the left, there's a sidebar with various navigation options like Get Started, Compliance, Home, Payments (Transactions, Customers, Refunds, Payouts, Disputes, Transaction Splits, Subaccounts, Terminals), Recurring (Audit Logs, Settings), and Support. A red arrow points to the 'Settings' option under the Recurring section. The main dashboard area shows a chart titled 'ZAR Revenue' with a value of 8,725.68, a bar chart for 'Last 30 Days' showing daily revenue, and a 'Next Payout' section indicating a payout of 8,725.68 due tomorrow.

- After That Click on API Keys & Webhooks for Paystack Secret Key & Paystack Public Key.

The screenshot shows the Paystack Settings page with the 'API Keys & Webhooks' tab selected. The sidebar remains the same as the previous screenshot. The main area contains fields for 'Test Secret Key' (with a red arrow pointing to it) and 'Test Public Key' (also with a red arrow pointing to it). Below these are fields for 'Test Callback URL' and 'Test Webhook URL' (containing the value 'https://yourapi.com/webhooks/paystack'), and a 'Save Changes' button.

- Copy & paste those inside PickBazar API and PickBazar Shop .env file like mentioned above.

## Special Notes for Paystack.

If we have used any third party system/plugin/packages, then we have always encouraged our respected customers to follow the official documentation for detailed & in-depth knowledge.

- [Paystack official documentation](#)
- [Paystack Webhooks documentation](#)
- [Accept Payments in US Dollars \(USD\)](#)

# Integratioin of new payment gateway

To integrate a payment gateway in ChawkBazar-Laravel, you will need to follow these general steps:

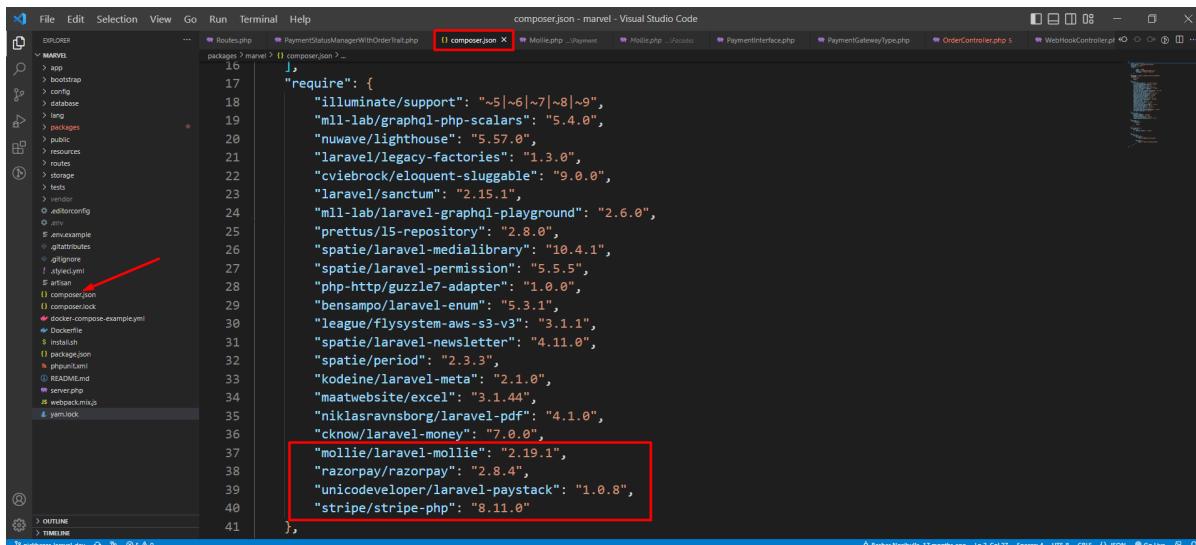
## Getting Started with API

### Step 1: Install and configure the payment gateway package

First, you will need to install the payment gateway package for Laravel (if there are any available package). There are several packages available that provide integration with various payment gateways. Example is given in the screenshot.

Once you have chosen and installed a package, you will need to configure it by adding your payment gateway credentials and any other required settings.

To check the installed dependencies of payment gateway in ChawkBazar-Laravel, you can open the composer.json file in a text editor and find that.



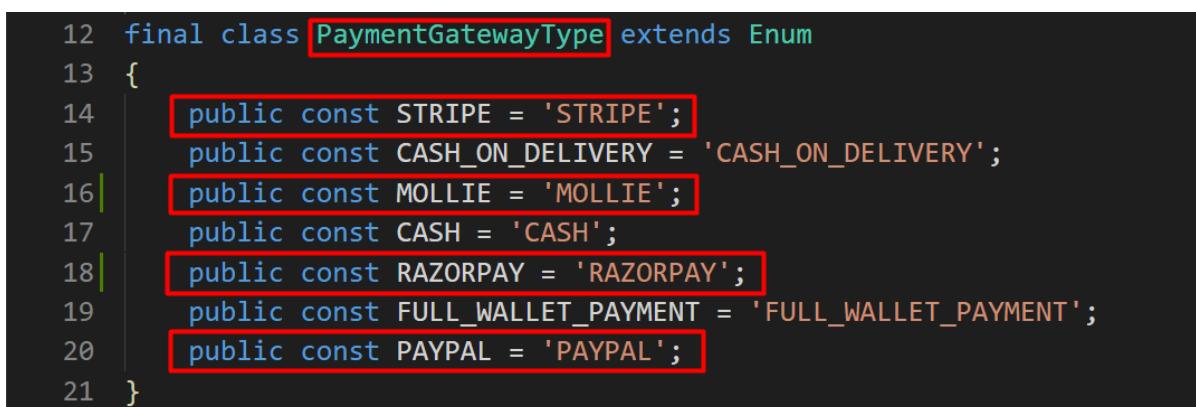
The screenshot shows the Visual Studio Code interface with the 'composer.json' file open. The 'require' section of the JSON file is highlighted with a red box, showing various Laravel packages and their versions. An arrow points from the left margin to the 'require' section. The code block below shows the contents of the highlighted area.

```
    "require": {  
        "illuminate/support": "~5|~6|~7|~8|~9",  
        "mll-lab/graphql-php-scalars": "5.4.0",  
        "nuwave/lighthouse": "5.57.0",  
        "laravel/legacy-factories": "1.3.0",  
        "cviebrock/eloquent-sluggable": "9.0.0",  
        "laravel/sanctum": "2.15.1",  
        "mll-lab/laravel-graphql-playground": "2.6.0",  
        "prettus/l5-repository": "2.8.0",  
        "spatie/laravel-medialibrary": "10.4.1",  
        "spatie/laravel-permission": "5.5.5",  
        "php-http/guzzle7-adapter": "1.0.0",  
        "bensampo/laravel-enum": "5.3.1",  
        "league/flysystem-aws-s3-v3": "3.1.1",  
        "spatie/laravel-newsletter": "4.11.0",  
        "spatie/period": "2.3.3",  
        "kodine/laravel-meta": "2.1.0",  
        "maatwebsite/excel": "3.1.44",  
        "niklasrvnsborg/laravel-pdf": "4.1.0",  
        "cknow/laravel-money": "7.0.0",  
        "mollie/laravel-mollie": "2.19.1",  
        "razorpay/razorpay": "2.8.4",  
        "unicodeveloper/laravel-paystack": "1.0.8",  
        "stripe/stripe-php": "8.11.0"  
    },
```

### Step 2: Add payment gateway name in the Enum.

Add PaymentGateway Enum API -> package -> marvel -> src -> Enums ->

PaymentGatewayType.php



The screenshot shows a code editor with the 'PaymentGatewayType.php' file open. A red box highlights several enum values: 'STRIPE', 'CASH\_ON\_DELIVERY', 'MOLLIE', 'CASH', 'RAZORPAY', 'FULL\_WALLET\_PAYMENT', and 'PAYPAL'. The code block below shows the contents of the highlighted area.

```
12 final class PaymentGatewayType extends Enum  
13 {  
14     public const STRIPE = 'STRIPE';  
15     public const CASH_ON_DELIVERY = 'CASH_ON_DELIVERY';  
16     public const MOLLIE = 'MOLLIE';  
17     public const CASH = 'CASH';  
18     public const RAZORPAY = 'RAZORPAY';  
19     public const FULL_WALLET_PAYMENT = 'FULL_WALLET_PAYMENT';  
20     public const PAYPAL = 'PAYPAL';  
21 }
```

## Step 3: Configure the Payment Facade for the new payment gateway.

Now go to API -> package -> marvel -> src -> Payment then create new payment Class (e.g. Stripe, PayPal, Razorpay, Mollie) for implements the PaymentInterface. The Class must implements all of the methods defined in the PaymentInterface.

The screenshot shows the Visual Studio Code interface with the file 'Razorpay.php' open. The code implements the 'PaymentInterface'. Several other files in the 'Payment' directory are listed in the Explorer sidebar, including 'Base.php', 'Mollie.php', 'Payment.php', 'PaymentInterface.php', 'Paypal.php', 'Paystack.php', 'Razorpay.php', and 'Stripe.php'. Red arrows point from the sidebar to each of these files, indicating they are part of the payment gateway implementation.

```
class Razorpay extends Base implements PaymentInterface
```

Methods defined in the PaymentInterface

The screenshot shows the Visual Studio Code interface with the file 'PaymentInterface.php' open. It defines the 'PaymentInterface' with several methods: getIntent, verify, handleWebHooks, createCustomer, attachPaymentMethodToCustomer, detachPaymentMethodToCustomer, retrievePaymentIntent, confirmPaymentIntent, setIntent, and retrievePaymentMethod. Red boxes highlight each of these method names. The sidebar shows the 'Payment' directory containing 'Base.php', 'Mollie.php', 'Payment.php', 'PaymentInterface.php', 'Paypal.php', 'Paystack.php', 'Razorpay.php', and 'Stripe.php'. A red arrow points from the sidebar to the 'PaymentInterface.php' file.

```
interface PaymentInterface
```

```
public function getIntent(array $data): array;
```

```
public function verify(string $id): mixed;
```

```
public function handleWebHooks(object $request): void;
```

```
public function createCustomer(object $request): array;
```

```
public function attachPaymentMethodToCustomer(string $retrieved_payment_method, object $request): object;
```

```
public function detachPaymentMethodToCustomer(string $retrieved_payment_method): object;
```

```
public function retrievePaymentIntent(string $payment_intent_id): object;
```

```
public function confirmPaymentIntent(string $payment_intent_id, array $data): object;
```

```
public function setIntent(array $data): array;
```

```
public function retrievePaymentMethod(string $method_key): object;
```

Here's an example of a class that implements that PaymentInterface:

```
1 <?php
2
3 namespace Marvel\Payments;
4
5 use Exception;
6 use Marvel\Database\Models\Order;
7 use Marvel\Database\Models\PaymentIntent;
8 use Marvel\Exceptions\MarvelException;
9 use Marvel\Traits\PaymentTrait;
10 use Razorpay\Api\Api;
11 use Marvel\Enums\OrderStatus;
12 use Marvel\Enums\PaymentStatus;
13 use Razorpay\Api\Errors\SignatureVerificationError;
14 use Str;
15 use Throwable;
16
17 class Razorpay extends Base implements PaymentInterface
18 {
19     use PaymentTrait;
20
21     public Api $api;
22
23     public function __construct()
24     {
25         parent::__construct();
26         $this->api = new Api(config('shop.razorpay.key_id'), config('shop.razorpay.key_secret'));
27     }
28
29 /**
30 * Get payment intent for payment
31 *
32 * @param $data
33 * @return array
34 * @throws MarvelException
35 */
36 public function getIntent($data): array
37 {
38     try {
39         extract($data);
40         $order = $this->api->order->create([
41             'receipt' => $order_tracking_number,
42             'amount' => round($amount, 2) * 100,
43             'currency' => $this->currency,
44         ]);
45
46         return [
47             'payment_id' => $order->id,
48             'order_tracking_number' => $order->receipt,
49             'currency' => $order->currency,
50             'amount' => $order->amount,
51             'is_redirect' => false,
52         ];
53     } catch (Exception $e) {
54         throw new MarvelException(SOMETHING_WENT_WRONG_WITH_PAYMENT);
55     }
56 }
57
58 /**
59 * Verify a payment
60 *
61 * @param $id
62 * @return false|mixed
63 * @throws MarvelException
64 */
65 public function verify($id): mixed
66 {
67     try {
68         $order = $this->api->order->fetch($id);
69         return isset($order->status) ? $order->status : false;
70     } catch (Exception $e) {
71         throw new MarvelException(SOMETHING_WENT_WRONG_WITH_PAYMENT);
72     }
73 }
74
75 /**
76 * handleWebHooks
77 *
78 * @param mixed $request
79 * @return void
80 * @throws Throwable
81 */
82 public function handleWebHooks($request): void
83 {
84     $webhookSecret = config('shop.razorpay.webhook_secret');
85     $webhookBody = @file_get_contents('php://input');
86     $webhookSignature = $request->header('X-Razorpay-Signature');
87
88     try {
89         if ($webhookBody && $webhookSignature && $webhookSecret) {
90             $this->api->utility->verifyWebhookSignature($webhookBody, $webhookSignature, $webhookSecret);
91         } else {
92             // Invalid request
93             http_response_code(400);
94             exit();
95         }
96     } catch (SignatureVerificationError $e) {
97         // Invalid signature
98         http_response_code(400);
99         exit();
99 }
```

```

100     }
101
102     $eventStatus = (string) Str::of($request->event)->replace('payment.', '', $request->event);
103 * Note : * It is important to note that each payment gateway has its own set of requirements and
104 may have different fields for processing payments. You will need to follow the official
105 documentation for your specific payment gateway.
106
107     switch ($eventStatus) {
108         case 'dispute.won':
109             break;
110         case 'dispute.created':
111             $this->updatePaymentOrderStatus($request, OrderStatus::PENDING, PaymentStatus::PROCESSING);
112             break;
113         case 'captured':
114             $this->updatePaymentOrderStatus($request, OrderStatus::PENDING, PaymentStatus::SUCCEEDED);
115             break;
116         case 'failed':
117             $this->updatePaymentOrderStatus($request, OrderStatus::PENDING, PaymentStatus::FAILED);
118         }
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

```

Go to `API -> package -> marvel -> src -> Http -> Controllers -> OrderController.php` -

> `submitPayment` then you've to add your PaymentGatewayType and function name in switch

```

121
122 /**
123 * Update Payment and Order Status
124 * @throws Exception
125 */
126 public function submitPayment(Request $request): void
127 {
128     $tracking_number = $request->tracking_number ?? null;
129     try {
130         $order = $this->repository->with(['products', 'children.shop', 'wallet_point', 'payment_intent'])
131             ->findOneByFieldOrFail(['field' => 'tracking_number', 'value' => $tracking_number]);
132
133         switch ($order->payment_gateway) {
134             case PaymentGatewayType::STRIPE:
135                 $this->stripe($order, $request, $this->settings);
136                 break;
137
138             case PaymentGatewayType::PAYPAL:
139                 $this->paypal($order, $request, $this->settings);
140                 break;
141
142             case PaymentGatewayType::MOLLIE:
143                 $this->mollie($order, $request, $this->settings);
144                 break;
145
146             case PaymentGatewayType::RAZORPAY:
147                 $this->razorpay($order, $request, $this->settings);
148                 break;
149
150         }
151     }
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

```

After That go to `API -> package -> marvel -> src -> Traits ->`

`PaymentStatusManagerWithOrderTrait.php` for verify your payment status update, you've to add

function like Stripe, PayPal, Razorpay or Mollie.

```

155 /**
156 * @return object
157 */
158 public function attachPaymentMethodToCustomer(string $retrieved_payment_method, object $request): object
159 {
160     $this->repository->attachPaymentMethod($request, $retrieved_payment_method);
161
162     return $request;
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

```

Step 5: How to setup webhook in ChawkBazar-laravel follow

the steps

To create a payment gateway webhook, you would first configure the webhook URL in your payment gateway account. Then, whenever the specified events occur, the payment gateway will send an

HTTP POST request to the webhook URL with a payload of data about the event.

```

201 /**
202 * API -> package -> marvel -> src -> Routes
203 *
73 Route::post('webhooks/razorpay', action: [WebHookController::class, 'razorpay']]);
74 Route::post('webhooks/stripe', action: [WebHookController::class, 'stripe']);
75 Route::post('webhooks/paypal', action: [WebHookController::class, 'paypal']]);
76 Route::post('webhooks/mollie', action: [WebHookController::class, 'mollie']));
211}
212

```

Add your function in WebHookController

```

File Edit Selection View Go Run Terminal Help
WebHookController.php - marvel - Visual Studio Code
Routes.php PaymentStatusManagerWithOrderTrait.php Mollie.php Razorpay.php PaymentInterface.php PaymentGatewayType.php OrderController.php WebHookController.php
8 class WebHookController extends CoreController
9 {
10
11     public function stripe(Request $request)
12     {
13         return Payment::handleWebHooks($request);
14     }
15
16     public function paypal(Request $request)
17     {
18         return Payment::handleWebHooks($request);
19     }
20
21     public function razorpay(Request $request)
22     {
23         return Payment::handleWebHooks($request);
24     }
25
26     public function mollie(Request $request)
27     {
28         return Payment::handleWebHooks($request);
29     }

```

To handle **webhook events** follow your payment gateway official webhook documentation

```

File Edit Selection View Go Run Terminal Help
Razorpay.php - marvel - Visual Studio Code
Request.php Response.php
82     public function handleWebhooks($request): void
83     {
84         $webhookSecret = config('shop.razorpay.webhook_secret');
85         $webhookBody = @file_get_contents('php://input');
86         $webhookSignature = $request->header('X-Razorpay-Signature');
87
88         try {
89             if ($webhookBody && $webhookSignature && $webhookSecret) {
90                 $this->xpl-utility->verifyWebhookSignature($webhookBody, $webhookSignature, $webhookSecret);
91             } else {
92                 // Invalid request
93                 http_response_code(400);
94                 exit();
95             }
96         } catch (SignatureVerificationError $e) {
97             // Invalid signature
98             http_response_code(400);
99             exit();
100        }
101
102        $eventStatus = (string) Str::of($request->event)->replace('payment.', '', $request->event);
103
104        switch ($eventStatus) {
105            case 'dispute.won':
106            case 'dispute.created':
107            case 'authorized':
108                $this->updatePaymentOrderStatus($request, OrderStatus::PENDING, PaymentStatus::PROCESSING);
109                break;
110            case 'captured':
111                $this->updatePaymentOrderStatus($request, OrderStatus::PROCESSING, PaymentStatus::SUCCESS);
112                break;
113            case 'failed':
114                $this->updatePaymentOrderStatus($request, OrderStatus::PENDING, PaymentStatus::FAILED);
115        }
116
117        // To prevent loop for any case
118        http_response_code(200);
119        exit();

```

\* Note : \* For locally webhook testing you can use ngrok tools for that. Please follow their official documentation.

## Getting Started with admin dashboard.

First go to `marvel-admin -> rest -> types -> index.ts -> export enum PaymentGateway`  
then add PaymentGateway Name.

```

24 export enum PaymentGateway {
25   STRIPE = 'STRIPE', ←
26   COD = 'CASH_ON_DELIVERY',
27   CASH = 'CASH',
28   FULL_WALLET_PAYMENT = 'FULL_WALLET_PAYMENT',
29   PAYPAL = 'PAYPAL', ←
30   MOLLIE = 'MOLLIE', ←
31   RAZORPAY = 'RAZORPAY', ←
32 }

```

after that go to `marvel-admin -> rest -> src -> components -> settings -> payment.ts` add name & Title.

```

1 export const PAYMENT_GATEWAY = [
2   { name: 'stripe', title: 'Stripe' },
3   { name: 'paypal', title: 'Paypal' },
4   { name: 'razorpay', title: 'RazorPay' },
5   { name: 'mollie', title: 'Mollie' },
6 ];

```

Then it will automatically add that payment gateway in the marvel-admin settings.

## Getting Started with shop front.

There are two types of payment gateway system can be integrated here.

- Redirect based payment gateway (e.g PayPal). Where the customer will redirect to that payment gateway site during order checkout. Complete the payment there. And then comeback to the application.
- Non redirect based payment gateway. Where the customer will stay on the application and complete the whole payment process here. Here we consider Stripe as a non-redirect based payment gateway. Though Stripe has features too similar to redirect based payment gateway.

## Redirect-base Payment Gateway

if you want to integrate redirect based payment gateway (e.g. PayPal, Mollie). follow the steps

First go to `shop -> src -> types -> index.ts -> export enum PaymentGateway` then add PaymentGateway Name.

```

24 export enum PaymentGateway {
25   STRIPE = 'STRIPE', ←
26   COD = 'CASH_ON_DELIVERY',
27   CASH = 'CASH',
28   FULL_WALLET_PAYMENT = 'FULL_WALLET_PAYMENT',
29   PAYPAL = 'PAYPAL', ←
30   MOLLIE = 'MOLLIE', ←
31   RAZORPAY = 'RAZORPAY', ←
32 }

```

after that go to `shop -> src -> components -> checkout -> payment -> Payment-grid.tsx` then add your PaymentGateway object.

```

File Edit Selection View Go Run Terminal Help
EXPLORER ... payment-grid.tsx
src > components > checkout > payment > payment-grid.tsx > isEnableCashOnDelivery
cart > categories > checkout > contact > digital > item > payment
cash-on-delivery.tsx
payment-grid.tsx
payment-online.tsx
schedule
wallet
address-grid-guest.tsx
address-grid.tsx
check-availability-action.tsx
coupons
rate-of-update-guest.tsx
quest-rain.tsx
place-order-action.tsx
cta
dashboard
item
layouts
manufacturer
orders
otp
payment
razorpay
stripe-payment-modal.tsx
stripe
add-new-payment-modal.tsx
pay-now-button.tsx
payment-modal.tsx
saved-card-view-header.tsx
products
profile
promotions
outline
timeline
77 > = {
    STRIPE: {
        name: 'Stripe',
        value: PaymentGateway.STRIPE,
        icon: '/payment/stripe.png',
        component: PaymentOnline,
    },
    PAYPAL: {
        name: 'Paypal',
        value: PaymentGateway.PAYPAL,
        icon: '',
        // icon: '/payment/paypal.png',
        component: PaymentOnline,
    },
    RAZORPAY: {
        name: 'RazorPay',
        value: PaymentGateway.RAZORPAY,
        icon: '/payment/razorpay.png',
        component: PaymentOnline,
    },
    MOLLIE: {
        name: 'Mollie',
        value: PaymentGateway.MOLLIE,
        icon: '/payment/mollie.png',
        component: PaymentOnline,
    }
}

```

## Non-redirect based payment gateway

First, complete the redirect-based payment gateway steps mentioned above. Because that two steps is universal for all payment gateway to apply.

After That go to `shop -> src -> components -> payment` then add a folder like Stripe, Razorpay. Inside your payment folder you can create your required typescript files with related functionalities for your Payment Gateway. For example, you can checkout the Stripe folder. You can find all the necessary indication, components guide, payment-method (card) saving options etc in the Stripe folder.

```

File Edit Selection View Go Run Terminal Help
EXPLORER ... stripe-payment-form.tsx
src > components > payment > stripe > stripe-payment-form.tsx > ...
payment
razorpay
razorpay-payment-modal.tsx
stripe
stripe-base-form.tsx
stripe-payment-form.tsx
stripe-payment-modal.tsx
stripe-saved-cards-list.tsx
add-new-payment-modal.tsx
pay-now-button.tsx
payment-modal.tsx
saved-card-view-header.tsx
products
profile
promotions
outline
timeline
16 interface Props {
17     paymentIntentInfo: PaymentIntentInfo;
18     trackingNumber: string;
19     paymentGateway: PaymentGateway;
20 }
21
22 const PaymentForm: React.FC<Props> = ({
23     paymentIntentInfo,
24     trackingNumber,
25     paymentGateway,
26 }) => {
27     const { t } = useTranslation('common');
28     const stripe = useStripe();
29     const elements = useElements();

```

That's all for today. If you need any more help you can always contact with our support agents in our support portal.

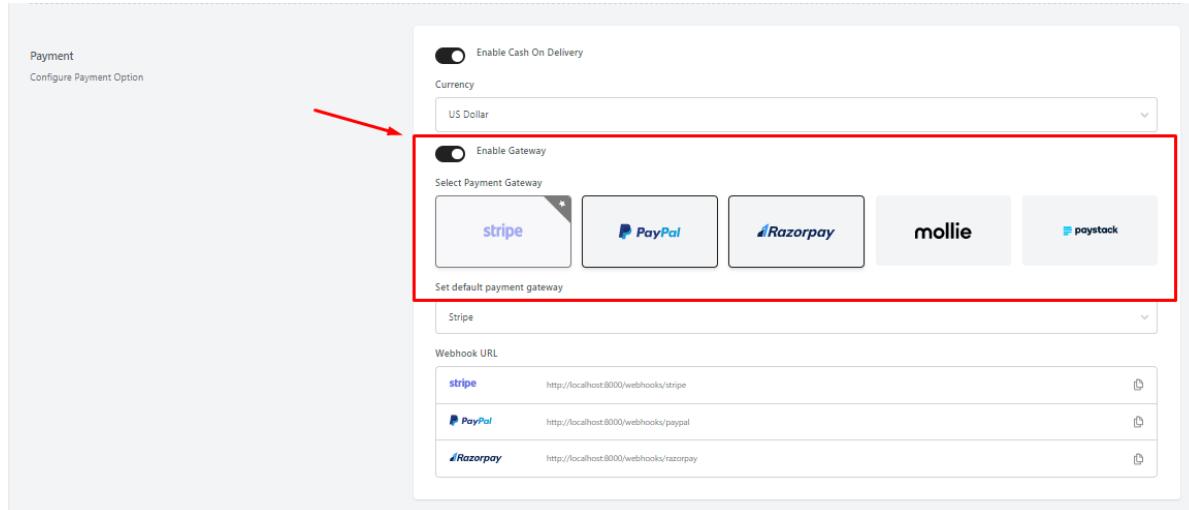
## Multiple payment gateway

In this chapter we are going to discuss about how you can use multiple payment gateway in our application. But before jumping into this feature user needs to keep some points in mind first.

1. Choose your desired payment gateway based on availability in your region.
2. Choose your currency correctly before activating that payment gateway.
3. Set up all the necessary .env keys and payment gateway internal settings properly before starting operations.
4. Configure webhooks properly if needed.

## How to activate multiple payment gateways [ in Admin ] ?

In admin dashboard there are new settings introduced for this feature. Please check this screenshot first,



## Description [ sequence by number in the screenshot ]

1. Choose the currency first
2. Enable gateway if you want to use any
3. Select the payment gateway based on your region and supported currency.
4. Select default payment gateway. Default gateway only available from selected ones.
5. Webhooks URL will be generated. You can use this URL for webhooks setting in payment gateways.

## How to use multiple payment gateways from customer end [ in Shop ] ?

Payment gateways activated by super-admin will be available in the checkout process. There are two steps of operation,

1. Select a payment gateway during place order
2. Select different payment gateway ( if needed ), after the place order.

This two points are cleared below with necessary screenshot.

## 1. Select a payment gateway during place order

Please check this screenshot first,

The screenshot shows the CHAWK order placement page. On the left, there are four steps: 1. Contact Number (input: 19365141641631), 2. Billing Address (input: Billing, 260 Terry Lane, New York, New York, 10001, US), 3. Shipping Address (input: Shipping, 1780 Angus Road, New York, New York, 10022, US), and 4. Delivery Schedule (options: Express Delivery, Morning 8:00 AM - 11:00 AM, Noon 11:00 AM - 2:00 PM, Afternoon, Evening). On the right, there is an order summary table:

Product	Subtotal
1 x Zara Army Bag   1 pc	\$260.00
Subtotal	\$260.00
Tax	\$5.20
Shipping	\$50.00
Total	\$315.20

Below the summary is a section titled "Choose Payment Method" with buttons for Stripe, PayPal, and Razorpay. A red arrow points from the "Place Order" button at the bottom right of this section to the "Place Order" button in the main content area.

Select a payment gateway just before placing order.

\*\* Notes for customer : \*\* Please choose this carefully. Otherwise you may need to re-select the payment procedure again.

## 2. Select different payment gateway ( if needed )

In order details page there is a option to re-select the payment gateway. If any customer select a gateway and placeorder but found out any issue of his/her selected gateway, then he/she will have a opportunity to re-select the payment gateway again and complete the order process.

Please check the screenshot below carefully,

The screenshot shows the CHAWK order details page. At the top, there are status buttons: "Order Pending" (highlighted in yellow), "Payment Pending", "Pay", and "Change gateway". A modal window titled "Choose Another Payment Gateway" is open, showing three options: PayPal, Razorpay, and stripe. A red arrow labeled "02" points to the "stripe" button. A red arrow labeled "01" points to the "Pay" button at the top right of the modal. A red arrow labeled "03" points to the "Submit Payment" button at the bottom of the modal. The background shows the order summary and product details.

## Description [ sequence by number in the screenshot ]

1. Stripe is the previous selected payment gateway. Which do not intend to use now. So he/she might change the gateway and submit the payment.
2. PayPal was selected payment gateway now.
3. Submit payment button need to be clicked, otherwise the operation will not be continued.

## Translation

If you're using `chawkbazar-laravel` after `v2.0.0` then you can translate static content for multilanguage. It doesn't matter if the language is RTL or not, you can do both. For translation we use this [next-i18next](#) package.

## Existing Language

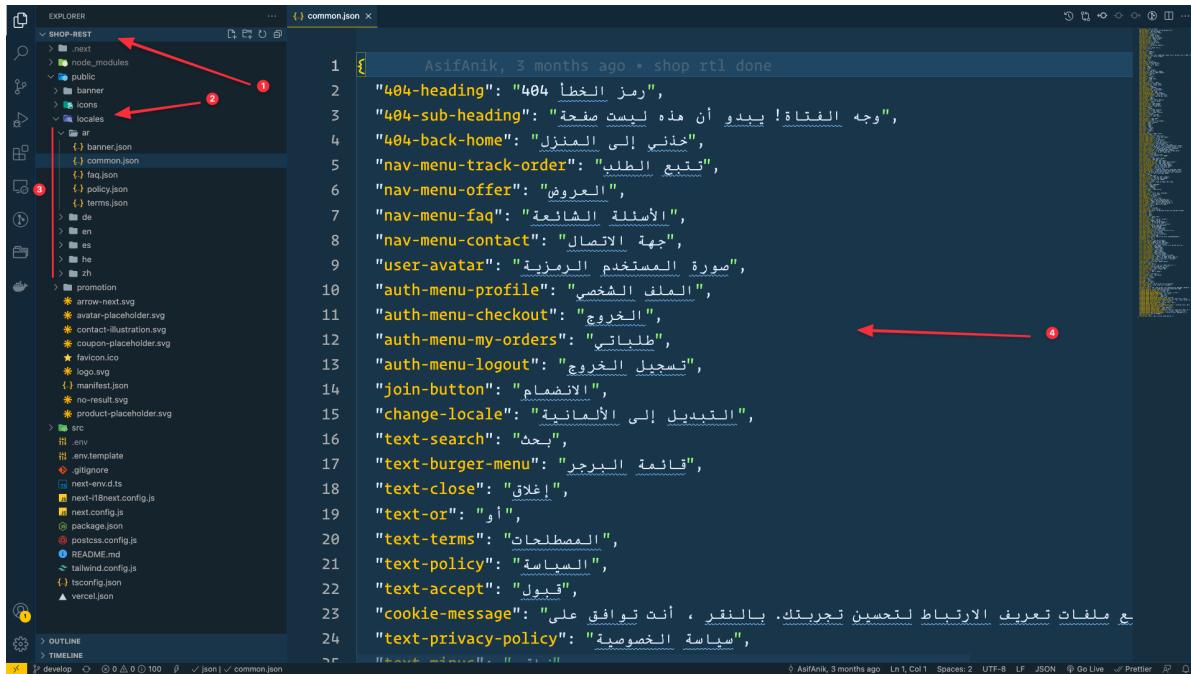
Right now, we provide translation files for `ar`, `de`, `en`, `es`, `he`, and `zh`.

If your translation language is from them then just edit the translation string from,

### For Rest

```
chawkbazar-laravel -> shop -> public -> locales -> YOUR LANGUAGE
```

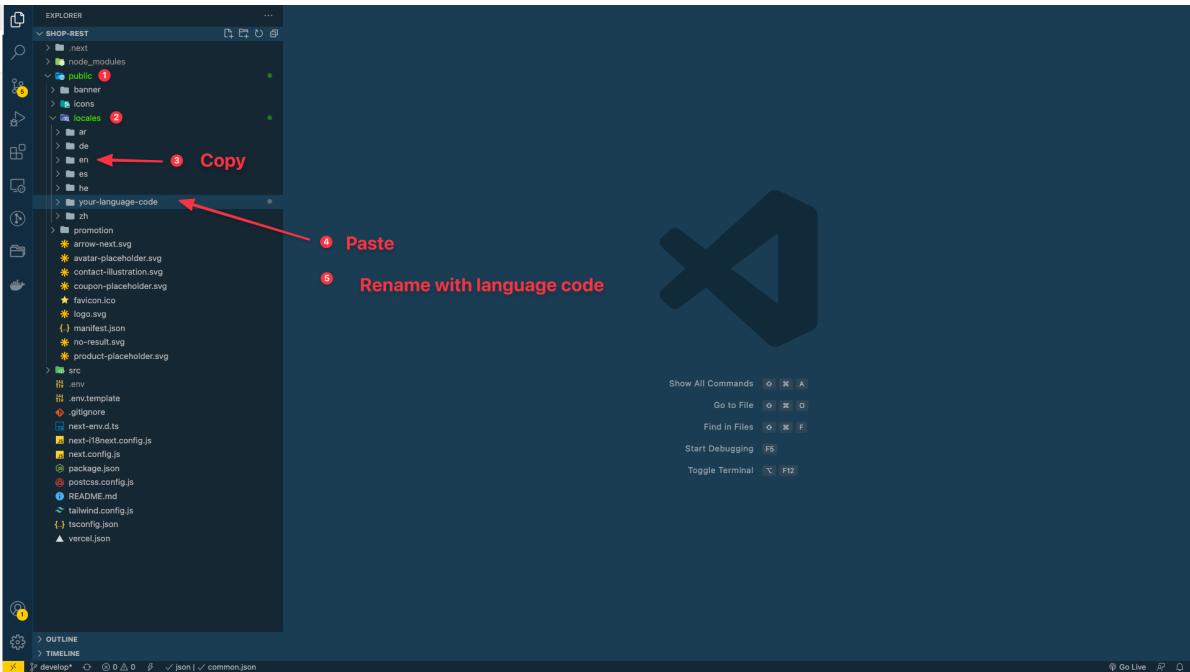
```
chawkbazar-laravel -> admin -> rest -> public -> locales -> YOUR LANGUAGE
```



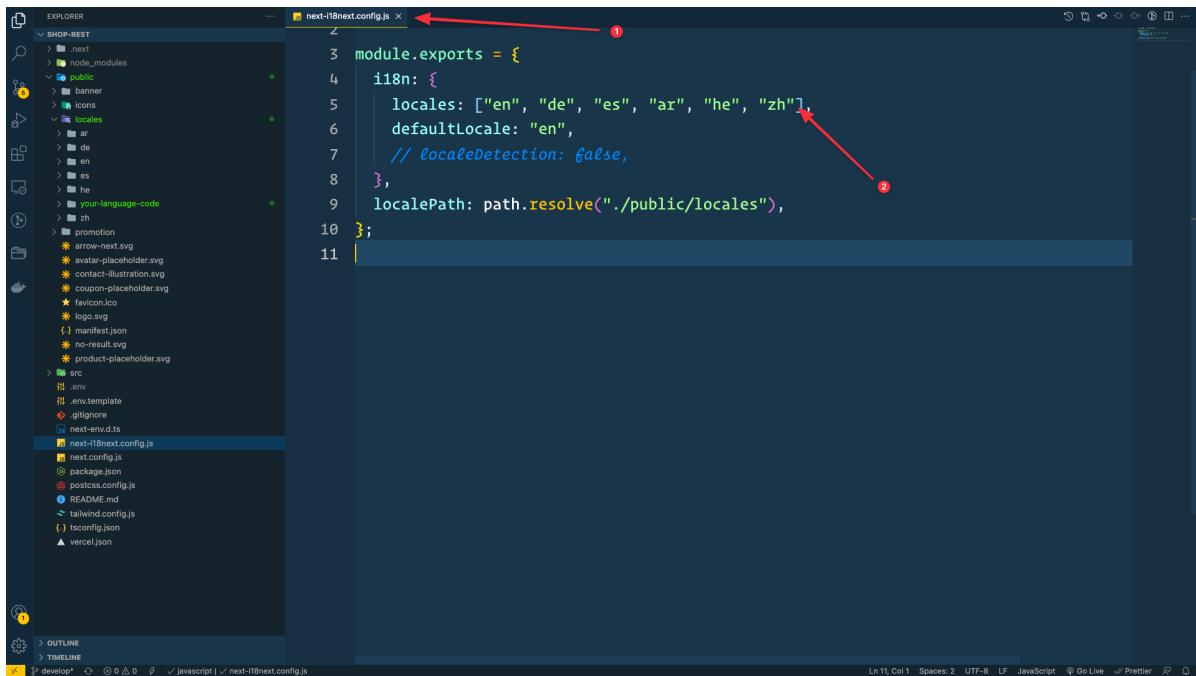
After translating `string` from `JSON` re-run or rebuild` project it'll translate the static string.

## New Language

If your language is not in the lists then just copy the `en` folder from `locale` and paste it in the same folder by rename your language code.



After creating that folder edit `next-i18next.config.js` and add the `language-code` at `locales` object.



## Default Language

By default, English is the default language. To change that edit,

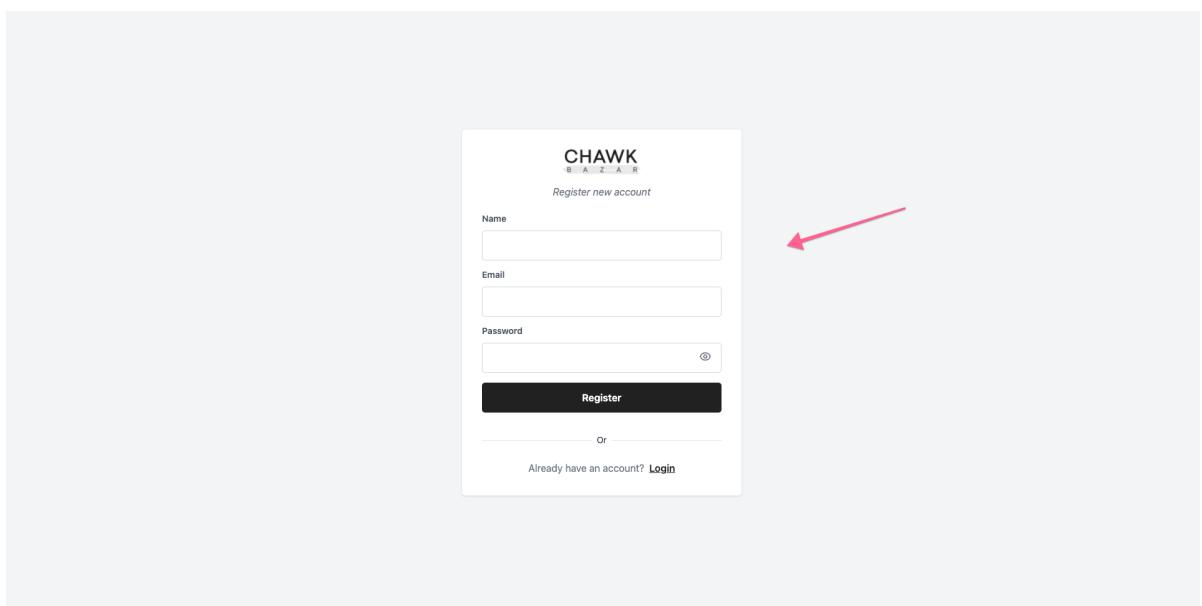
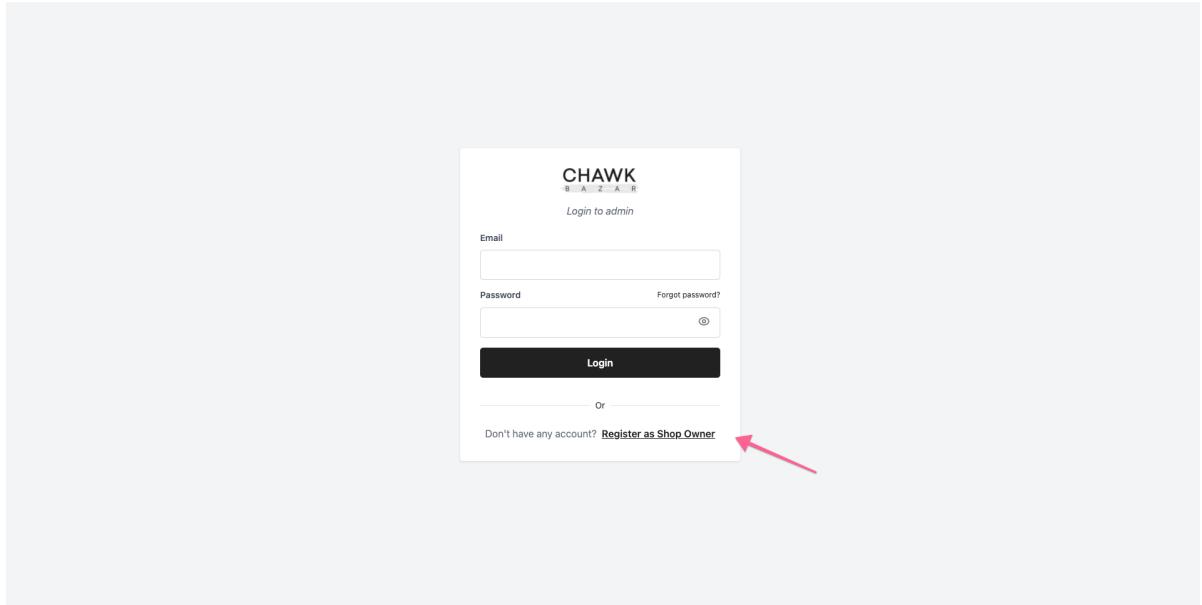
```
chawkbazar-laravel -> shop -> public -> locales -> YOUR LANGUAGE -> next-i18next.config.js
```

```
chawkbazar-laravel -> admin -> rest -> public -> locales -> YOUR LANGUAGE -> next-i18next.config.js
```

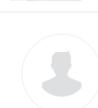
# Multivendor

## Create New Shop

To create new shop login as an [administrator](#) or create a new account for creating [shop](#)



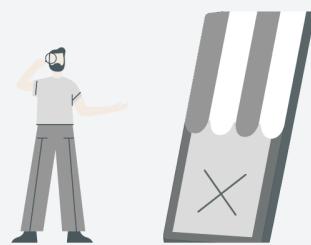
After creating the account you'll be redirected to this page,



Mahmud

mahmud@test.com  
Please add your [Profile](#) Enabled

## My Shops



No Shops Found

[Create Shop](#)After that, click [Create Shop](#)

Mahmud

mahmud@test.com  
Please add your [Profile](#) Enabled

## Create Shop

## Logo

Upload your shop logo from here

Upload an image or drag and drop  
PNG, JPG

## Cover Image

Upload your shop cover image from here  
Dimension of the cover Image should be 1170 x 435px

Upload an image or drag and drop  
PNG, JPG

## Basic Info

Add some basic info about your shop from here

Name	<input type="text"/>
Description	<input type="text"/>

[Create Shop](#)And provide all the information for the [store](#).

After creating the shop you'll redirect to this page,



Mahmud

mahmud@test.com  
Please add your [Profile](#) Enabled

## My Shops



Dummy Shop

inactive[Create Shop](#)

By default, the shop will be inactive. Only **administrator** can active a shop.

From **administrator** account go to **shop** and click **tick mark** to activate or deactivate a shop.

Logo	Name	Owner Name	Products	Orders	Status	Actions
	Dummy Shop	Mahmud	0	0	Inactive	
	name	Admin	0	0	Inactive	
	Chawkbazar Vendor shop	Vendor	57	5	Active	
	Chawkbazar Shop	Admin	0	0	Active	

After **activate** the shop by **administrator** the **vendor** dashboard will be like this,

CHAWK BAZAAR

**Mahmud**  
mahmud@test.com  
Please add your [Profile](#)

Enabled

**My Shops**

	Dummy Shop	Active
--	------------	--------

After click on **shop**, you'll be redirected to dashboard page,

- [Dashboard](#)
- [Attributes](#)
- [Products](#)
- [Orders](#)
- [Staffs](#)
- [Withdraws](#)



## Dummy Shop

This is a vendor shop of chawkbazar

📍 4360 Hampton Meadows, Boston,  
Massachusetts, 02210, USA  
📞 01234567852

[Visit Shop](#)[Edit Shop](#)

## Products

57  
Total Products

5  
Total Orders

## Revenue

\$7,906.50  
Gross Sales

\$7,828.50  
Current Balance

## Others

10 %  
Admin Commission Rate

Registered Since  
October 9, 2021

## Payment Information

Name:  
vendor  
Email:  
vendor@demo.com  
Bank:  
vendor bank  
Account No.:  
6523498651

From this dashboard, you can maintain your shop,

## Dashboard:

- [Dashboard](#)
- [Attributes](#)
- [Products](#)
- [Orders](#)
- [Staffs](#)
- [Withdraws](#)



## Dummy Shop

This is a vendor shop of chawkbazar

📍 4360 Hampton Meadows, Boston,  
Massachusetts, 02210, USA  
📞 01234567852

[Visit Shop](#)[Edit Shop](#)

## Products

57  
Total Products

5  
Total Orders

## Revenue

\$7,906.50  
Gross Sales

\$7,828.50  
Current Balance

## Others

10 %  
Admin Commission Rate

Registered Since  
October 9, 2021

## Payment Information

Name:  
vendor  
Email:  
vendor@demo.com  
Bank:  
vendor bank  
Account No.:  
6523498651

## Attributes:

- [Dashboard](#)
- [Attributes](#)
- [Products](#)
- [Orders](#)
- [Staffs](#)
- [Withdraws](#)

## Attributes

[+ Add Attribute](#)

⋮

ID	Name	Values	Actions
No data found			

## Products:

The screenshot shows the CHAWK BAZAAR dashboard. On the left, a sidebar menu lists: Dashboard, Attributes, Products (with a red arrow pointing to it), Orders, Staffs, and Withdraws. The main content area is titled "Products" with a red arrow pointing to it. It features a search bar, a "Add Product" button, and a "Filter" dropdown. A table below has columns for Image, Name, Brand, Product Type, Price/Unit, Quantity, Status, and Actions. The message "No data found" is displayed.

## Order:

The screenshot shows the CHAWK BAZAAR dashboard. On the left, a sidebar menu lists: Dashboard, Attributes, Products, Orders (with a red arrow pointing to it), Staffs, and Withdraws. The main content area is titled "Orders" with a red arrow pointing to it. It features a search bar, a "Download" button, and an "Actions" button. A table below has columns for Tracking Number, Delivery Fee, Total, Order Date, Status, Shipping Address, Download, and Actions. The message "No data found" is displayed.

## Staff:

The screenshot shows the CHAWK BAZAAR dashboard. On the left, a sidebar menu lists: Dashboard, Attributes, Products, Orders, Staffs (with a red arrow pointing to it), and Withdraws. The main content area is titled "Staff" with a red arrow pointing to the "+ Add Staff" button. It features a search bar, a "Download" button, and an "Actions" button. A table below has columns for Name, Email, Status, and Actions. The message "No data found" is displayed.

## WithDraw:

The screenshot shows the CHAWK BAZAAR dashboard. On the left, there's a sidebar with navigation links: Dashboard, Attributes, Products, Orders, Staffs, and Withdraws. A red arrow points to the 'Withdraws' link. The main content area is titled 'Withdraws' and contains a table with columns: Shop Name, Amount, Status, and Created. The table shows 'No data found'. In the top right corner of this section, there's a dark button labeled '+ Request Withdraw' with a red arrow pointing to it.

## User Roles:

### Super Admin:

Super admin can do everything. the admin can maintain and edit every store on the site.

### Store Owner:

Store owner can edit or main it's store, staff or payment.

### Staff:

The staff of a store has similar permission as store owner, but the staff can't update the store or withdraw payment.

## Withdraw Payment:

Only the store owner can withdraw its payment. To do that go to your shop dashboard -> withdraws -> Request a withdraw,

This screenshot is identical to the one above, showing the CHAWK BAZAAR dashboard with the 'Withdraws' section. It features the same sidebar navigation, the same 'Withdraws' page content with the '+ Request Withdraw' button, and the same red arrows indicating the 'Withdraws' link in the sidebar and the request button.

- Dashboard
- Attributes
- Products
- Orders
- Staffs
- Withdraws

## Create Withdraw

## Description

Add withdraw request from here

## Amount

## Payment Method

## Details

## Note

**Request Withdraw**

After request payment, the dashboard will be like this,

- Dashboard
- Attributes
- Products
- Orders
- Staffs
- Withdraws

## Withdraws

**+ Request Withdraw**

Shop Name	Amount	Status	Created
Chawkbazar Vendor shop	\$500.00	Pending	a few seconds ago
Chawkbazar Vendor shop	\$78.00	Approved	5 hours ago

&lt; 1 &gt;



After request, the admin has to be approved the admin.



- [Dashboard](#)
- [Shops](#)
- [My Shops](#)
- [Products](#)
- [Attributes](#)
- [Brands](#)
- [Categories](#)
- [Tags](#)
- [Orders](#)
- [Order Status](#)
- [Users](#)
- [Coupons](#)
- [Taxes](#)
- [Shipments](#)
- [Withdraws](#)
- [Settings](#)

Withdraws

Shop Name	Amount	Status	Created	Actions
Chawkbazar Vendor shop	\$500.00	Pending	a few seconds ago	
Chawkbazar Vendor shop	\$78.00	Approved	5 hours ago	

< 1 >



- [Dashboard](#)
- [Shops](#)
- [My Shops](#)
- [Products](#)
- [Attributes](#)
- [Brands](#)
- [Categories](#)
- [Tags](#)
- [Orders](#)
- [Order Status](#)
- [Users](#)
- [Coupons](#)
- [Taxes](#)
- [Shipments](#)
- [Withdraws](#)
- [Settings](#)

Withdrawal Information

Amount	: 500	Status	Pending	
Payment Method	N/A			
Status	Approved On Hold Processing Pending Rejected			

After approved,



- [Dashboard](#)
- [Shops](#)
- [My Shops](#)
- [Products](#)
- [Attributes](#)
- [Brands](#)
- [Categories](#)
- [Tags](#)
- [Orders](#)
- [Order Status](#)
- [Users](#)
- [Coupons](#)
- [Taxes](#)
- [Shipments](#)
- [Withdraws](#)
- [Settings](#)

Withdraws

Shop Name	Amount	Status	Created	Actions
Chawkbazar Vendor shop	\$500.00	Approved	a minute ago	
Chawkbazar Vendor shop	\$78.00	Approved	5 hours ago	

< 1 >

## FrontEnd Shop

From front end when customers click on the `shop` page they'll get all the shops as a card list,

## Super Shops Near You



Dummy Shop

NEW



Chawkbazar Vendor shop

NEW



Chawkbazar Shop

NEW

## Get Expert Tips In Your Inbox

Subscribe to our newsletter and stay updated.

## Social

-  Facebook
-  Twitter
-  Instagram

## Contact

-  Contact Us
- Email: admin@chawkbazar.demo
- Website: <https://redq.io>

## About

-  Support Center
-  Customer Support
-  Copyright

## Customer Care

-  FAQ & Help
-  Shipping & Delivery
-  Return & Exchanges

## Our Information

-  Privacy policy update
-  Terms & conditions
-  Return Policy

## Community

-  Announcements
-  Answer center
-  Discussion boards
-  Giving works

Also when customers click on a product they will get the seller link,

Home / Products / Dido Pilot Glass



## Dido Pilot Glass

Polarized sunglasses reduce glare reflected off of roads, bodies of water, snow and other horizontal surfaces. Restore true color. Vision lenses are 400UV rated, meaning it can block UVA and UVB radiation.

\$300.00 \$350.00

SKU: kjkjnjk894561230

Category: Sunglass, Sports

Tags: Kids Collection, Flash Sale, Sunglass, Sports, Men's Collection

Brand: Roseban

Shop: Chawkbazar Vendor shop



## Related Products

After clicking the **seller link** they will redirect to the seller shop page,



FASHION

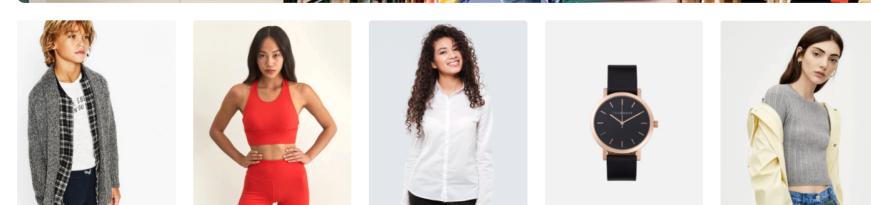
**Chawkbazar Vendor shop**

This is a vendor shop of chawkbazar

**Address:**  
4360 Hampton Meadows, Massachusetts, Boston, 02210, USA

**Phone:**  
01236547852

[Call Now](#)



# Export Import

Export Import works under a shop. So, You have to go to the `shop page -> products menu` for product or `shop page -> attributes menu` to import or export attributes.

The screenshot shows the CHAWK BAZAAR interface. On the left, there's a sidebar with various menu items: Dashboard, Attributes (highlighted with a red arrow), Products (highlighted with a red arrow), Orders, Staffs, and Withdraws. The main content area is titled "Attributes" and lists three attributes: Color (values: Red, Blue, Yellow), Size (values: Small, Medium, Large), and Shoe Size (values: 7, 8, 9, 10). At the top right of this section is a "+ Add Attribute" button and a three-dot menu icon.

Export Import work different way for simple and variable product

## Simple Product

For simple products, you've to export only products data. To do that, go to `Your Shop -> products`. Then click on `three dots` and export the simple products.

The screenshot shows the CHAWK BAZAAR interface. On the left, there's a sidebar with various menu items: Dashboard, Attributes, Products (highlighted with a red arrow), Orders, Staffs, and Withdraws. The main content area is titled "Products" and lists several products with columns for Image, Name, Brand, Product Type, Price/Unit, Quantity, Status, and Actions. A search bar and a "+ Add Product" button are at the top. A "Filter" dropdown and a three-dot menu icon are also present. A modal window titled "Export/Import" is open, showing four options: Import Products, Import Product Variations, Export Products (highlighted with a red arrow), and Export Product Variations. Each option has a cloud icon with an upward or downward arrow.

Similarly, for import `csv`, go to the same option and import exported `csv`

The screenshot shows the CHAWK BAZAAR dashboard. On the left sidebar, under the 'Products' section, there is a red arrow labeled '1' pointing to the 'Products' link. In the main content area, there is a 'Products' table listing several items. Overlaid on the table is a white box containing four cloud icons with labels: 'Import Products', 'Import Product Variations', 'Export Products', and 'Export Product Variations'. A red arrow labeled '2' points to the top right corner of this overlay. Another red arrow labeled '3' points to the 'Import Products' icon. The table columns include: Image, Name, Brand, Product Type, Price/Unit, Quantity, Status, and Action.

Image	Name	Brand	Product Type	Price/Unit	Quantity	Status	Action
	Zara Monte Carlo	Fania Fashion	variable	\$90.00 - \$100.00	240	<button>publish</button>	<span>trash</span> <span>edit</span>
	Reyban Havana Phantos Sunglasses	Hipster	simple	\$100.00	50	<button>publish</button>	<span>trash</span> <span>edit</span>
	Philip Lim Leather Shoulder Bag	Phonix Bags	simple	\$260.00	100	<button>publish</button>	<span>trash</span> <span>edit</span>
	Nike Pro Mesh Top with Leggings	Vintgae	variable	\$30.00 - \$35.00	100	<button>publish</button>	<span>trash</span> <span>edit</span>
	Nike Comfy Vapor Maxpro	AB Shoes	variable	\$220.00 - \$250.00	2000	<button>publish</button>	<span>trash</span> <span>edit</span>

## variable products

For variable products, you will need three different [csv](#).

- Attributes [csv](#)
- products [csv](#)
- variation options [csv](#)

Make sure to re-import you export these three csv.

This screenshot shows the 'Attributes' section of the CHAWK BAZAAR interface. On the left sidebar, the 'Products' option is highlighted with a red arrow labeled '1'. In the main content area, there is a table of attributes with columns for ID, Name, and Values. At the top right of this table is a button for '+ Add Attribute'. To the right of the table is a vertical 'Actions' column with icons for edit and delete. A large white callout box labeled 'Export/import' is overlaid on the page. It contains four options: 'Import Attributes' (with a cloud icon), 'Export Attributes' (with a cloud icon), 'Import Products' (with a person icon), and 'Export Products' (with a person icon). A red arrow labeled '3' points from the bottom of the 'Export/import' box towards the 'Export Attributes' button. Another red arrow labeled '2' points upwards from the 'Actions' column towards the 'Export/import' box.

This screenshot shows the 'Products' section of the CHAWK BAZAAR interface. On the left sidebar, the 'Products' option is highlighted with a red arrow labeled '1'. In the main content area, there is a table of products with columns for Image, Name, Brand, Product Type, Price/Unit, Quantity, Status, and Actions. At the top right of the table is a search bar, a '+ Add Product' button, and a 'Filter' dropdown. A large white callout box labeled 'Export/import' is overlaid on the page. It contains four options: 'Import Products' (with a person icon), 'Import Product Variations' (with a person icon), 'Export Products' (with a person icon), and 'Export Product Variations' (with a person icon). A red arrow labeled '3' points from the bottom of the 'Export/import' box towards the 'Export Products' button. Another red arrow labeled '2' points upwards from the 'Actions' column towards the 'Export/import' box.

This screenshot shows the 'Attributes' section of the CHAWK BAZAAR interface, identical to the one in the first screenshot. The 'Products' option is highlighted with a red arrow labeled '1'. The main content area shows a table of attributes with columns for ID, Name, and Values. At the top right is a '+ Add Attribute' button. To the right is an 'Actions' column with edit and delete icons. A large white callout box labeled 'Export/import' is overlaid. It contains two options: 'Import Attributes' (with a cloud icon) and 'Export Attributes' (with a cloud icon). A red arrow labeled '3' points from the bottom of the 'Export/import' box towards the 'Export Attributes' button. Another red arrow labeled '2' points upwards from the 'Actions' column towards the 'Export/import' box.

The screenshot shows the CHAWK BAZAAR interface. On the left sidebar, under the 'Products' section (marked with a red arrow 1), there are several items: Dashboard, Attributes, Products (selected), Orders, Staffs, and Withdraws. The main content area is titled 'Products' and contains a search bar and a table of products. The table columns include Image, Name, Brand, Product Type, Price/Unit, Quantity, Status, and Action (marked with a red arrow 2). Below the table is a modal titled 'Export/Import' with four options: Import Products, Import Product Variations, Export Products (marked with a red arrow 3), and Export Product Variations.

Now, during import variations product,  
At First, you have to import Attributes  
then Products and  
Then products variation.

The screenshot shows the CHAWK BAZAAR interface. On the left sidebar, under the 'Attributes' section (marked with a red arrow 1), there are several items: Dashboard, Attributes (selected), Products, Orders, Staffs, and Withdraws. The main content area is titled 'Attributes' and contains a table of attributes. The table columns include ID, Name, and Values. There are three attributes listed: Color (Values: Red, Blue, Yellow), Size (Values: Small, Medium, Large), and Shoe Size. Below the table is a modal titled 'Export/Import' with two options: Import Attributes (marked with a red arrow 3) and Export Attributes.

The screenshots illustrate the product management interface in the CHAWK BAZAAR dashboard. The top screenshot shows the main products list with a sidebar menu. The 'Products' item in the menu is highlighted with a red arrow (1). In the top right corner, there is a 'Filter' dropdown and a three-dot menu icon. A red arrow (2) points to the three-dot menu icon. Another red arrow (3) points to the 'Import Products' button within a modal titled 'Export/Import'. The bottom screenshot shows the same interface but with different highlighted areas. Red arrows point to the 'Actions' dropdown in the top right (2) and the 'Import Product Variations' button in the 'Export/Import' modal (3).

# Settings Management

In `settings` menu you will get the settings management form there. You can modify the basic information about your E-Commerce System, such as the Change Logo, Site Title, Site Information , Payment Information, SEO, Delivery Schedule, Shop Settings, etc.

**Dashboard**

**Shops**

**My Shops**

**Products**

**Attributes**

**Brands**

**Categories**

**Tags**

**Orders**

**Users**

**Coupons**

**Taxes**

**Shipments**

**Withdrawals**

**Settings**

**Settings**

**Logo**  
Upload your site logo from here.  
Dimension of the logo should be 128x40 Pixel  
Image size should not be more than 2 MB

**Information**  
Change your site information from here

**Site Title**  
ChawkBazar

**Site Subtitle**  
Your next ecommerce

**Minimum Order Amount (in currency)**  
0

Use OTP at checkout

Enable Must Verify Email

Enable AI

**Note:** If you make changes or update any fields, it's important to save those changes by clicking on the **Save Settings** button bottom of the page.

**Brands**

**Categories**

**Tags**

**Orders**

**Users**

**Coupons**

**Taxes**

**Shipments**

**Withdrawals**

**Settings**

**Select social platform** Twitter Add profile url https://twitter.com/home Remove

**Select social platform** Instagram Add profile url Remove

**Add New Social Profile**

**Save Settings**

## Change Logo

Click on Upload an image and select the file for your new logo from your Device. or Drag and Drop your new logo. (**Supported Formats JPG or PNG**)

**Logo**  
Upload your site logo from here.  
Dimension of the logo should be 128x40 Pixel  
Image size should not be more than 2 MB

**Upload an image or drag and drop**  
PNG, JPG

**CHAWK BAZAAR**

We support two types of file uploads; one is **local**, and another is **AWS S3**.

Both uploading systems follow this procedure,

## API

At first, add your API URL to your **api -> .env.**

```
1 APP_NAME=ChawkBazar
2 APP_ENV=production
3 APP_KEY=base64:dladdfdsfdfdsfssdf/sdfdsfer
4 APP_DEBUG=true
5 APP_URL=https://YOUR_DOMAIN.COM/backend
6 APP_SERVICE=chawkbazar.test
7 APP_NOTICE_DOMAIN=CHAWKBAZAR_
8
9 DUMMY_DATA_PATH=chawkbazar
10
11 LOG_CHANNEL=stack
12 LOG_LEVEL=debug
13
```

If you're using windows and your API is running with ports like `localhost:8000` or `127.0.0.1:8000`, then make sure you add the domain with port to `APP_URL`. Like this  
`API_URL=http://localhost:8000`

After that, clear your API cache by using this command from the `api` folder,

```
php artisan optimize:clear
```

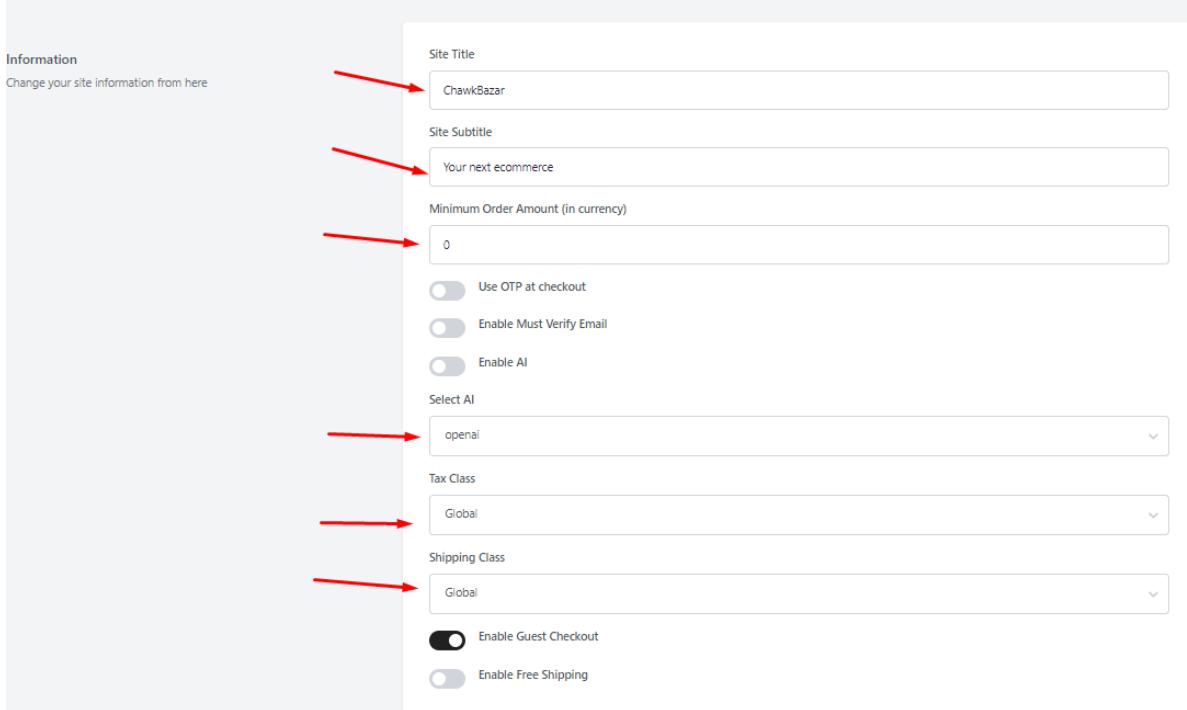
Then link the storage system by using this command,

```
php artisan storage:link
```

If you're using AWS S3, then update `MEDIA_DISK=s3` and add all `AWS_` credentials  
For AWS s3, make sure your properly setup permission of the bucket with ACL enable by follow this link -> <https://stackoverflow.com/a/70603995/2158023>

## Change Site information

This is the Site `Information` area of the admin settings. Here you can change your Site Title, Site Subtitle, Currency, Minimum Order Amount, Wallet Currency Ratio, Sign Up Points, Maximum Question Limit, OTP checkout, Tax Class, Shipping Class and handle Free Shipping.



The screenshot shows the 'Information' section of the admin settings. It includes fields for Site Title (ChawkBazar), Site Subtitle (Your next ecommerce), Minimum Order Amount (0), Select AI (openai), Tax Class (Global), Shipping Class (Global), and checkboxes for Use OTP at checkout, Enable Must Verify Email, Enable AI, Enable Guest Checkout, and Enable Free Shipping.

Setting	Value
Site Title	ChawkBazar
Site Subtitle	Your next ecommerce
Minimum Order Amount (in currency)	0
Select AI	openai
Tax Class	Global
Shipping Class	Global
Enable Guest Checkout	Enabled
Enable Free Shipping	Enabled

## Shipping class, Tax class & Free Shipping

You may find the shipping class at the conclusion of the information section. You may choose the precise class you wish to utilize from this menu. These classes were developed within the shipping category. The classes include a value for shipping that should be added to the total price bought.

For tax classes, the same applies. You may create tax classes in the admin settings' tax area by entering a dollar amount or percentage that will be added to the total amount of your order.

The option to allow free delivery is another one. It may be turned off to stop offering free delivery. Customers will receive free delivery if it is activated. If not, the cost of shipment must be paid.

The screenshot shows the 'Information' section of the admin settings. It includes fields for Site Title (ChawkBazar), Site Subtitle (Your next ecommerce), Minimum Order Amount (0), Select AI (openai), Tax Class (Global), and Shipping Class (Global). It also features toggle switches for Enable Guest Checkout and Enable Free Shipping.

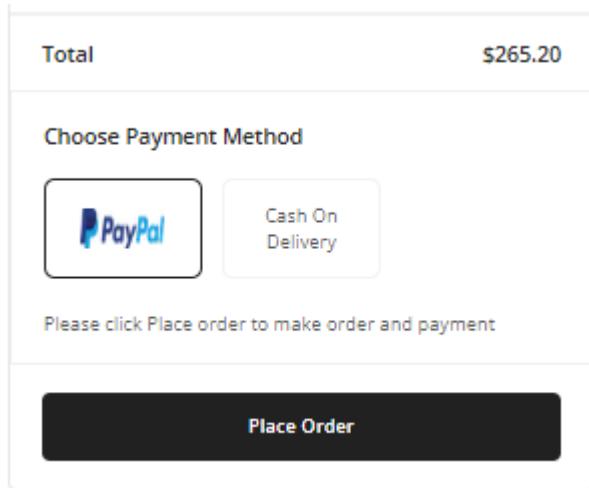
## Payment Gateway Management

This is the `payment` area of the admin settings. This will allow you to choose a payment gateway such as Stripe, Rezorpay, Paypal, Mollie Paystack etc. You can also enable or disable Cash On Delivery.

The screenshot shows the 'Payment' section of the admin settings. It includes a toggle for Enable Cash On Delivery, a dropdown for Currency (US Dollar), a toggle for Enable Gateway, and a list of payment gateways (stripe, PayPal, Razorpay, mollie, paystack). It also features a dropdown for Set default payment gateway (Stripe) and a field for Webhook URL (https://chawkbazarapi.redq.io/admin/webhooks/stripe).

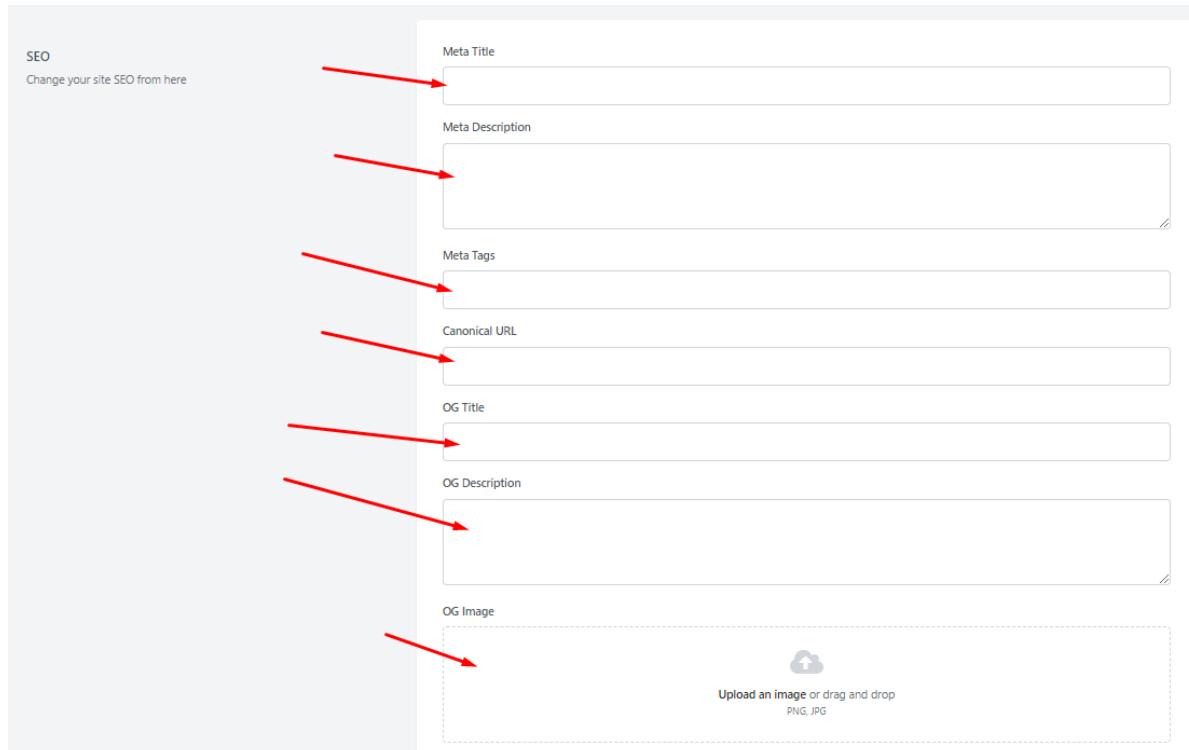
## Customer

When a client goes to a store to buy anything, he adds it to his cart. When he goes to pick a payment option, he will see `Cash on delivery` as well as payment gateways such as Stripe, Paypal, Razorpay, and Mollie, which are configured in the payment settings area of the dashboard.



## SEO

Search Engine Optimization (`SEO`) is a process of optimizing your website to rank higher in search engine results pages (SERPs) for specific keywords and phrases. you can add Meta Title, Meta Description, Meta Tags, Canonical URL, OG Title, OG Description etc.



## Add or Remove Delivery Schedule

This is the `Delivery Schedule` area of the settings. This will allow you to add new Delivery Schedule such as Morning, Noon, Afternoon, Evening etc. You can also remove any existing Delivery Schedule.

**Delivery Schedule**  
Add your delivery schedule time with proper description from here

Title/Time	<input type="text" value="Express Delivery"/>	Remove
Description	<input type="text" value="90 min express delivery"/>	

---

Title/Time	<input type="text" value="Morning"/>	Remove
Description	<input type="text" value="8.00 AM - 11.00 AM"/>	

---

Title/Time	<input type="text" value="Noon"/>	Remove
Description	<input type="text" value="11.00 AM - 2.00 PM"/>	

---

Title/Time	<input type="text" value=""/>	
------------	-------------------------------	--

## Shop Settings & Google Map

This is the **Shop Settings** area of the settings. This will allow you to change location from map, Contact Number, Website, Enable or Disable Google Map Address, or add new social platform account, etc.

**Shop Settings**  
Add your shop settings information from here

Set location from map	<input type="text" value="NY State Thruway, New York, USA"/>	
Contact Number	<input type="text" value="+129290122122"/>	
Website	<input type="text" value="https://redq.io"/>	
<input checked="" type="checkbox"/> Enable product review system before publish ?		

---

Select social platform	<input type="button" value="Facebook"/>	<input type="text" value="https://www.facebook.com/"/> Remove
------------------------	---	---

---

Select social platform	<input type="button" value="Twitter"/>	<input type="text" value="https://twitter.com/home"/> Remove
------------------------	--	--

---

Select social platform	<input type="button" value="Instagram"/>	<input type="text"/> Remove
------------------------	--	-----------------------------

## New Static Page

Both shop and admin are built using React NextJS framework. So all the existing pages are available to this location,

**Shop,**

```
/chawkbazar-laravel/shop/src/pages
```

**Admin,**

```
/chawkbazar-laravel/admin/rest/src/pages
```

You can use the NextJS routing feature for the new pages. Check these official NextJS docs for pages and routing,

<https://nextjs.org/docs/basic-features/pages>

<https://nextjs.org/docs/routing/introduction>

## **FAQ:**

### **How to change the default layout?**

Please go to,

```
chawkbazar-laravel/shop/src/pages
```

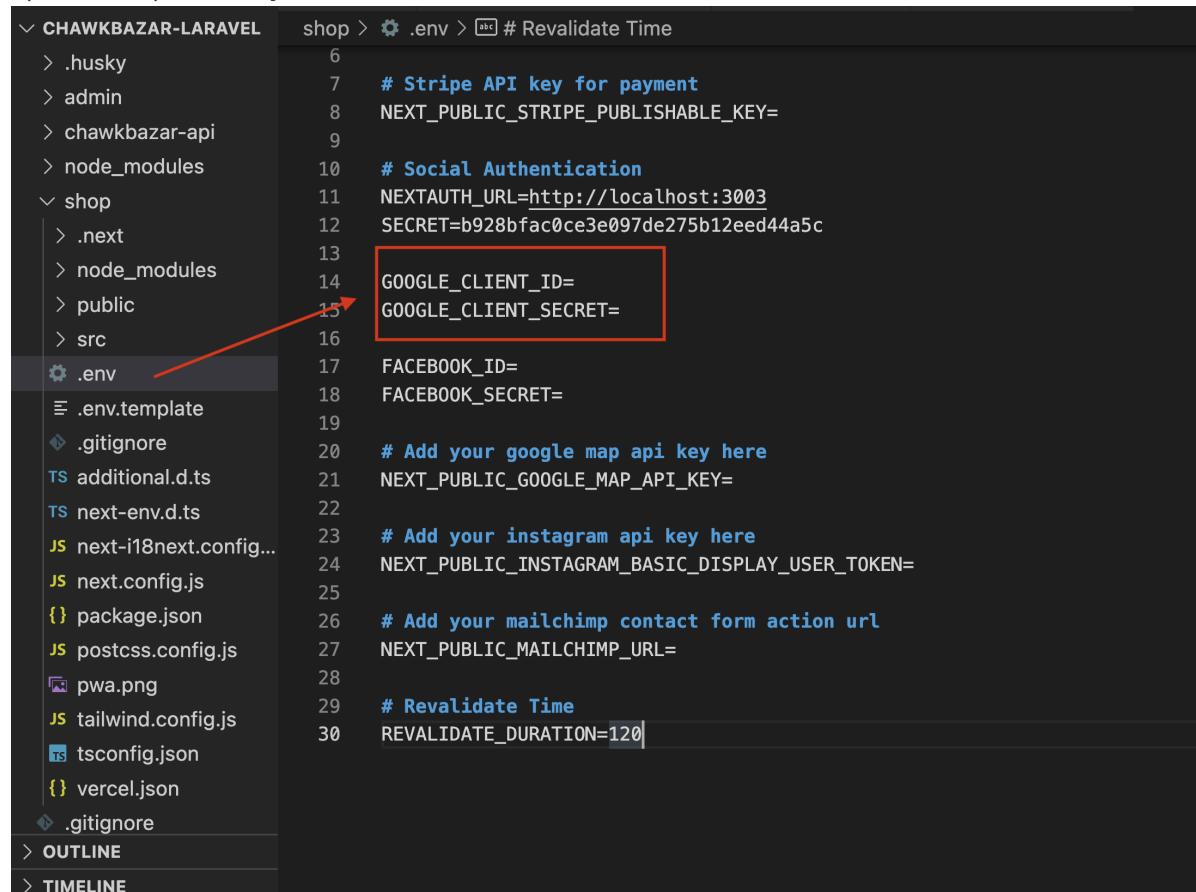
And delete or change the name of `index.tsx`

Then rename `your specific demo.tsx` to `index.tsx`

After that, make sure you rebuild your project to run in production.

# Why showing client\_id error while Google social login

Ans: Go to `.env` file and change the variables from `GOOGLE_ID=` to `GOOGLE_CLIENT_ID=` and `GOOGLE_SECRET=` to `GOOGLE_CLIENT_SECRET=`. Make this changes in the `.env` file of chawkbazar-api and shop directory.



```
shop > ⚙ .env > # Revalidate Time
6
7 # Stripe API key for payment
8 NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY=
9
10 # Social Authentication
11 NEXTAUTH_URL=http://localhost:3003
12 SECRET=b928bfa0ce3e097de275b12eed44a5c
13
14 GOOGLE_CLIENT_ID=
15 GOOGLE_CLIENT_SECRET=
16
17 FACEBOOK_ID=
18 FACEBOOK_SECRET=
19
20 # Add your google map api key here
21 NEXT_PUBLIC_GOOGLE_MAP_API_KEY=
22
23 # Add your instagram api key here
24 NEXT_PUBLIC_INSTAGRAM_BASIC_DISPLAY_USER_TOKEN=
25
26 # Add your mailchimp contact form action url
27 NEXT_PUBLIC_MAILCHIMP_URL=
28
29 # Revalidate Time
30 REVALIDATE_DURATION=120
```

## How to configure Stripe payment gateway?

To configure Stripe payment gateway,

1. At first, create a developer account from stripe developer dashboard (<https://dashboard.stripe.com/>)
2. log in to that account

3. Then go to the `Developer -> API Keys` section and on that section create an API key. After creating, you'll get one `Secret key` and one `Publishable key`.

The screenshot shows the Stripe Developers API keys page. On the left sidebar, under 'Developers', the 'API keys' option is highlighted with a red arrow. The main content area is titled 'API keys' and contains two sections: 'Standard keys' and 'Restricted keys'. The 'Standard keys' section has a heading 'These will allow you to authenticate API requests.' with a 'Learn more' link. It lists two entries:

NAME	TOKEN	LAST USED	CREATED	...
Publishable key	pk_tes... 26a0ZU... 9vNqrl...	Jul 18	Apr 11	...
Secret key	(REDACTED)	Jul 18	Jul 10	...

A red arrow labeled '1' points to the 'Secret key' row, and another red arrow labeled '2' points to the 'Publishable key' row. Below the table, there's a button 'Reveal test key'. The 'Restricted keys' section has a heading 'For greater security, you can create restricted API keys that limit access and permissions for different areas of your account data.' with a 'Learn more' link and a '+ Create restricted key' button.

4. Then copy `Secret key` to your `chawkbazar/.env`

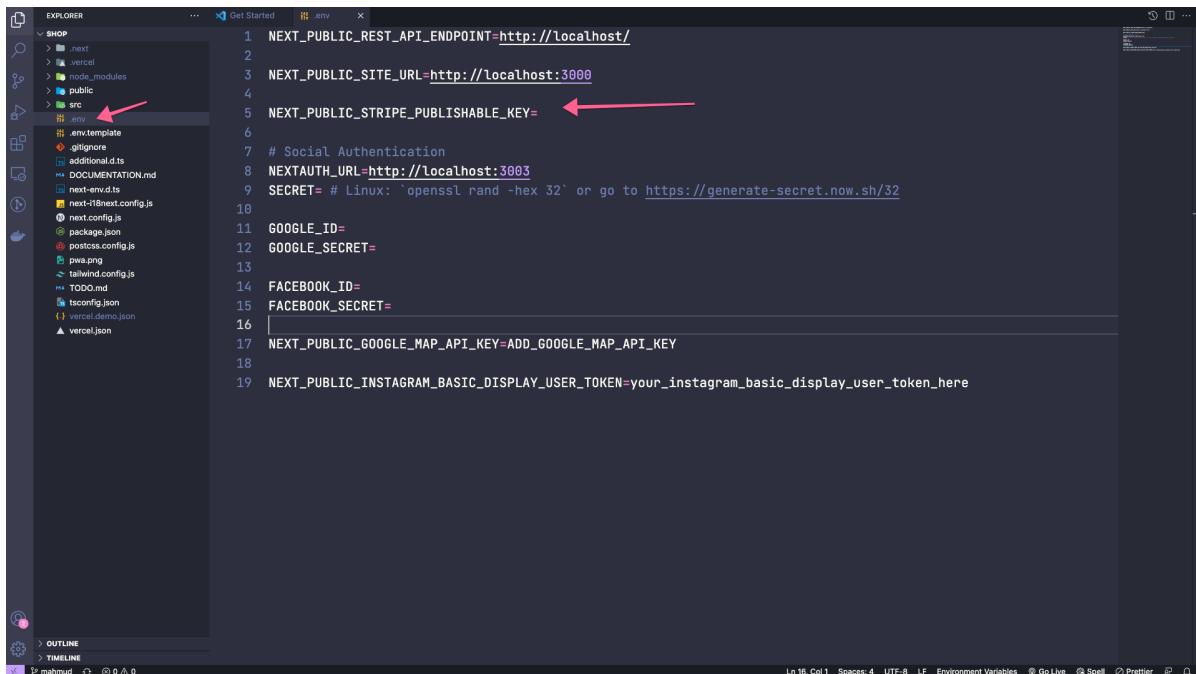
The screenshot shows a code editor displaying the `.env` file for the `CHAWKBAZAR-API` project. The file contains several environment variables, and the `STRIPE_API_KEY` variable is highlighted with a red arrow. The file also contains other variables like `MAIL_FROM_ADDRESS`, `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`, `AWS_DEFAULT_REGION`, `AWS_BUCKET`, `LIGHTHOUSE_CACHE_ENABLE`, `MEDIA_DISK`, `SHOP_URL`, `DASHBOARD_URL`, `ADMIN_EMAIL`, `GOOGLE_CLIENT_ID`, `GOOGLE_CLIENT_SECRET`, `GOOGLE_REDIRECT_URI`, `FACEBOOK_CLIENT_ID`, `FACEBOOK_CLIENT_SECRET`, `FACEBOOK_REDIRECT_URI`, `ACTIVE_OTP_GATEWAY`, `TWILIO_AUTH_TOKEN`, `TWILIO_ACCOUNT_SID`, `TWILIO_VERIFICATION_SID`, `MESSAGEBIRD_API_KEY`, and `MESSAGEBIRD_ORIGINATOR`.

```

MAIL_FROM_ADDRESS=
AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=
AWS_BUCKET=
STRIPE_API_KEY= ←
LIGHTHOUSE_CACHE_ENABLE=false
MEDIA_DISK=public
SHOP_URL=
DASHBOARD_URL=
ADMIN_EMAIL=support@example.com
GOOGLE_CLIENT_ID=
GOOGLE_CLIENT_SECRET=
GOOGLE_REDIRECT_URI=http://127.0.0.1/login/google/callback
FACEBOOK_CLIENT_ID=
FACEBOOK_CLIENT_SECRET=
FACEBOOK_REDIRECT_URI=https://127.0.0.1/login/facebook/callback
ACTIVE_OTP_GATEWAY=twilio
TWILIO_AUTH_TOKEN=
TWILIO_ACCOUNT_SID=
TWILIO_VERIFICATION_SID=
MESSAGEBIRD_API_KEY=
MESSAGEBIRD_ORIGINATOR=Marvel

```

5. And similarly, add `Publishable key` to your `frontend/shop/.env`



```
1 NEXT_PUBLIC_REST_API_ENDPOINT=http://localhost/
2
3 NEXT_PUBLIC_SITE_URL=http://localhost:3008
4
5 NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY= ←
6
7 # Social Authentication
8 NEXTAUTH_URL=http://localhost:3008
9 SECRET= # Linux: `openssl rand -hex 32` or go to https://generate-secret.now.sh/32
10
11 GOOGLE_ID=
12 GOOGLE_SECRET=
13
14 FACEBOOK_ID=
15 FACEBOOK_SECRET=
16
17 NEXT_PUBLIC_GOOGLE_MAP_API_KEY=ADD_GOOGLE_MAP_API_KEY
18
19 NEXT_PUBLIC_INSTAGRAM_BASIC_DISPLAY_USER_TOKEN=your_instagram_basic_display_user_token_here
```

After configuration, make sure you rebuild your project using this command,

For REST API

```
yarn build:shop-rest
yarn build:admin-rest
```

## Why am I facing "You may need an appropriate loader to handle this file type" during running shop rest?

Ans: Please run `yarn clean` then `yarn` then run `yarn dev:shop-rest`

## I am changing schema files but changes is not working

Your changes might not work because schema is cached. SO you might need to clear schema cache using the below command `php artisan lighthouse:clear-cache`.

## Changing .env files but not getting the changes

Run Below command `php artisan optimize:clear`

## Changing route but not getting the changes.

Run `php artisan optimize:clear` or `php artisan route:clear`

## I have set `STRIPE_API_KEY` in .env but still getting error.

In some cases `STRIPE_API_KEY` value can't read from .env in those cases you have to put the key in the config file directly in `api/packages/marvel/src/Config/laravel-omnipay.php`

## 6. Getting error on forget password email sending

Make sure you have run the `php artisan marvel:install` commands successfully and copied the necessary email templates to your resources folder. You can also do it by `php artisan marvel:copy-files` command.

NB: This same issue can occur during order creation.

# Can I use it with my existing laravel?

Yes, you can. Follow the below steps.

- Make sure you are using laravel 8.
- Copy `api/packages` folder from the downloaded files into your laravel root folder.
- Put below code in your laravel `composer.json` file into `require` section.

```
"ignited/laravel-omnipay": "dev-master",
"omnipay/common": "dev-master",
"omnipay/stripe": "dev-master",
"chawkbazar/shop": "dev-master"
```

- Put below code in bottom of your `composer.json`. If you already have an `repositories` section then put code inside `repositories` to your existing `repositories`

```
"repositories": {
    "chawkbazar/shop": {
        "type": "path",
        "url": "packages/marvel"
    },
    "ignited/laravel-omnipay": {
        "type": "path",
        "url": "packages/laravel-omnipay"
    },
    "omnipay/common": {
        "type": "path",
        "url": "packages/omnipay-common"
    },
    "omnipay/stripe": {
        "type": "path",
        "url": "packages/omnipay-stripe"
    }
}
```

- Now run `composer install`
- Copy necessary env variables from `.env.example` to your `env` file.
- Run `php artisan marvel:install` and follow necessary steps.
- To run server `php artisan serve`
- For image upload to work properly you need to run `php artisan storage:link`.

## Why am I getting Access denied for user?

navigate to `api` then run `./vendor/bin/php down -v`. It will delete any of your existing mysql volumes. Now run `./vendor/bin/php up -d` on same directory or run `bash install.sh` on root directory

## Why am I getting permission issue during deployment?

Run below commands for fixing permission issue in your laravel app during deployment

```
sudo chown -R $USER:www-data storage
sudo chown -R $USER:www-data bootstrap/cache

sudo chown -R www-data:www-data storage
sudo chown -R www-data:www-data bootstrap/cache
```

## Why am I getting "The GET method is not supported for this route. Supported methods: HEAD"?

Run `php artisan optimize:clear`

## I am getting "Cannot use 'Mixed' as class name as it is reserved" after access graphql endpoint.

You are using the application in PHP 8. Downgrade it to PHP 7.4. Mixed is a reserved word in PHP 8.

## How to resolve the Load More Infinity loading issue?

If you click on the `Load More` button and the button is spinning continuously, then check API request from the network tab. If it is HTTP and your API is hosted on HTTPS, then open,

```
/Users/tareqmahmud/tmbox/tmserver/larabox/work/redq/laravel-
ecommerce/api/app/Providers/AppServiceProvider.php
```

And add this code to `boot` method,

```
if (!\App::environment('local')) {
    $this->app['request']->server->set('HTTPS', true);
}
```

## I'm trying to upload images, but the images are not displayed on the frontend?

We support two types of file uploads; one is `local`, and another is `AWS S3`.

Both uploading systems follow this procedure,

## API

At first, add your API URL to your `chawkbazar-api -> .env.`

```
1 APP_NAME=ChawkBazar
2 APP_ENV=production ←
3 APP_KEY=base64:dLaddfdfdfdsfssdf/sdfdsfer
4 APP_DEBUG=true
5 APP_URL=https://YOUR_DOMAIN.COM/backend[] ←
6 APP_SERVICE=chawkbazar.test
7 APP_NOTICE_DOMAIN=CHAWKBAZAR_
8
9 DUMMY_DATA_PATH=chawkbazar
10
11 LOG_CHANNEL=stack
12 LOG_LEVEL=debug
13
```

If you're using windows and your API is running with ports like `localhost:8000` or `127.0.0.1:8000`, then make sure you add the domain with port to `APP_URL`. Like this  
`API_URL=http://localhost:8000`

After that, clear your API cache by using this command from the `chawkbazar-api` folder,

```
php artisan optimize:clear
```

Then link the storage system by using this command,

```
php artisan storage:link
```

If you're using AWS S3, then update `MEDIA_DISK=s3` and add all `AWS_` credentials

For AWS s3, make sure your properly setup permission of the bucket with ACL enable by follow this link -> <https://stackoverflow.com/a/70603995/2158023>

Amazon S3 > Buckets > pickbazarlaravel > Edit Object Ownership

### Edit Object Ownership Info

#### Object Ownership

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

##### ACLs disabled (recommended)

All objects in this bucket are owned by this account.  
Access to this bucket and its objects is specified using only policies.

##### ACLs enabled

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

#### Object Ownership

##### Bucket owner preferred

If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.

##### Object writer

The object writer remains the object owner.

Cancel

Save changes

## Admin

Then add your domain to your `chawkbazar-laravel -> admin -> rest -> next.config.js`

```
images: {
  domains: [
    "YOUR_DOMAIN.COM", ←
    "via.placeholder.com",
    "res.cloudinary.com",
    "s3.amazonaws.com",
    "18.141.64.26",
    "127.0.0.1",
    "localhost",
    "picsum.photos",
    "pickbazar-sail.test",
    "pickbazarlaravel.s3.ap-southeast-1.amazonaws.com",
    "chawkbazarlaravel.s3.ap-southeast-1.amazonaws.com",
    "lh3.googleusercontent.com",
  ],
},
```

If you're using `Aws S3`, then also add your root S3 domain like this,

```
images: {
  domains: [
    "YOUR_DOMAIN.COM", You, 7 minutes ago + Uncommitted changes
    "via.placeholder.com",
    "res.cloudinary.com",
    "s3.amazonaws.com",
    "18.141.64.26",
    "127.0.0.1",
    "localhost",
    "picsum.photos",
    "pickbazar-sail.test",
    "pickbazarlaravel.s3.ap-southeast-1.amazonaws.com", ← Like this
    "chawkbazarlaravel.s3.ap-southeast-1.amazonaws.com",
    "lh3.googleusercontent.com",
  ],
},
```

## Shop

Similarly, add your domain to your `chawkbazar-laravel -> shop -> next.config.js`

```
images: {
  domains: [
    "YOUR_DOMAIN.COM", ←
    "via.placeholder.com",
    "res.cloudinary.com",
    "s3.amazonaws.com",
    "18.141.64.26",
    "127.0.0.1",
    "localhost",
    "picsum.photos",
    "pickbazar-sail.test",
    "pickbazarlaravel.s3.ap-southeast-1.amazonaws.com",
    "chawkbazarlaravel.s3.ap-southeast-1.amazonaws.com",
    "lh3.googleusercontent.com",
  ],
},
```

If you're using `Aws S3`, then also add your root S3 domain like this,

```
images: {
  domains: [
    "YOUR_DOMAIN.COM",           You, 7 minutes ago • Uncommitted changes
    "via.placeholder.com",
    "res.cloudinary.com",
    "s3.amazonaws.com",
    "18.141.64.26",
    "127.0.0.1",
    "localhost",
    "picsum.photos",
    "pickbazar-sail.test",
    "pickbazarlaravel.s3.ap-southeast-1.amazonaws.com", ← Like this
    "chawkbazarlaravel.s3.ap-southeast-1.amazonaws.com",
    "lh3.googleusercontent.com",
  ],
},
```

## Build Frontend,

If you run in dev mode, then re-run will affect your update. But if you're running as a production mode, then you've to rebuild the project and re-run again. Otherwise, image upload will not work.

<https://chawkbazar-laravel-doc.vercel.app/vps-server#build-project>

After building the project, replace your server shop and admin folder using this rebuild folder and re-run the app using this command,

```
pm2 restart all
```

## How to resolve the `502 Bad Gateway` error for frontend/admin?

There can be several reasons that throw 502 errors for the frontend in the production server.

### Reason 1:

On your `Nginx` file, a specific port is set for shop or the admin, so when you run the script as a `PM2` instance`, the script has to be run that specific port; otherwise, your domain/subdomain will throw a 502 error.

#### PORT:

```
3003 -> Shop Rest
3002 -> Admin Rest
```

To check that at the first stop the PM2 instance by this command,

```
pm2 stop 'all'
```

then go to the `chawkbazar` folder from your server and try to run the script using `yarn` and check which port it is running,

Rest:

```
yarn start:admin-rest
```

```
yarn shop-rest
```

And, check if the port matched with the Nginx port or not? If not matched, then change the port to Nginx config.

## Reason 2:

To check that at the first stop the PM2 instance by this command,

```
pm2 stop 'all'
```

then go to the chawkbazar folder from your server and try to run the script using yarn and check which port it is running,

Rest:

```
yarn start:admin-rest
```

```
yarn shop-rest
```

With this command, it'll give you an error that no build file is found. To resolve that, rebuild your project and then rerun the scripts.

<https://chawkbazar-doc.vercel.app/vps-server#frontend-project-build>

## How to resolve javascript heap out of memory issue?

If you're using cPanel based server, then you maybe get an error like "call\_and\_retry\_last allocation failed - javascript heap out of memory".

To resolve that issue, you've to add a limit of the memory bound,

Please check this,

<https://stackoverflow.com/questions/38558989/node-js-heap-out-of-memory>

<https://support.snyk.io/hc/en-us/articles/360002046418-JavaScript-heap-out-of-memory>

<https://blog.openreplay.com/javascript-heap-out-of-memory-error>

## Image upload throw Internal Server Error; how to resolve that?

For uploading, the php8.1-gd library is required. So make sure you install the php8.1-gd library on your server.

## **Sometimes my server shutdown or halts automatically. After restart, it works fine again; how to resolve that?**

---

In general, this happens mainly when your upstream server fails to handle its processing power when there is lots of traffic or requires lots of operation. Please upgrade your server configuration to double to check the issue. Please check this video,

<https://www.youtube.com/watch?v=4pIqAz5bzX0>

If that doesn't solve the issue, then you've to redeploy again with a new ec2 instance.

## **S3 uploading doesn't work after adding credentials to .env; how to resolve that?**

---

For AWS s3, make sure your properly setup permission of the bucket with ACL enable by follow this link

<https://stackoverflow.com/a/70603995/2158023>

## **How to rebuild the project?**

---

When you upload the code to the server as a production, NextJS fetch its code from the build file. So after deployed, when you update or change code on scripts, the updated scripts don't work at the front end until you rebuild the project again. NextJs works this way.

Now there are two ways you can follow to rebuild the project,

If your server has more than 2+ CPU core and 6GB+ of memory, then you can directly edit code at your server then rebuild the project from the server directly using this command,

At first go to `/var/www/chawkbazar-laravel`

For `shop`,

```
yarn build:shop-rest
```

And for `admin`,

```
yarn build:admin-rest
```

After completing the build, restart the PM2 process using this command,

```
pm2 restart 'all'
```

But suppose your server doesn't have that processing power, and you try to build your scripts directly from the server. In that case, your server will be shut down during the build, and you've to restart the server.

In that case, you can rebuild the scripts at your local computer and upload them to your server. If you follow our [VPS Server Deployment](#) guide, then you already know the process. To build the scripts on your pc, follow this procedure,

At first, download the `chawkbazar-laravel` folder from the server to your computer, and customize the code at your computer. Then use this command from the `root` folder,

To install all the packages,

```
yarn
```

Then build the shop,

```
yarn build:shop-rest
```

Then build the admin,

```
yarn build:admin-rest
```

Then delete the shop and admin folder from the server and upload your build shop and admin folder to the server at the exact same location and use this command to restart the PM2,

```
pm2 restart 'all'
```

## How to increase upload size?

The API supports `10MB` of upload limit for a single file by default. If your upload file is more than `10MB`, then please follow this,

At first, login to the server using ssh, and then,

Edit,

```
/etc/php/1/fpm/php.ini
```

Then update,

```
upload_max_filesize = 256M  
post_max_size = 256M  
max_execution_time = 300
```

Then restart the PHP,

```
sudo service php8.1-fpm restart
```

Then update,

```
/etc/nginx/sites-enabled/chawkbazar
```

and Add `client_max_body_size 256M`

```
client_max_body_size 256M
```

```

server {
    server_name [REDACTED];
    client_max_body_size 256M; ←

    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-XSS-Protection "1; mode=block";
    add_header X-Content-Type-Options "nosniff";

    index index.html index.htm index.php;

    charset utf-8;
}

```

Then restart the server,

```

sudo nginx -t
sudo service nginx restart

```

After that update,

```
chawkbazar-api/packages/marvel/config/media-library.php
```

```
'max_file_size' => 1024 * 1024 * 256,
```

```

...
1  <?php
2
3  return [
4
5      /*
6          * The disk on which to store added files and derived images by default.
7          * one or more of the disks you've configured in config/filesystems.
8          */
9      'disk_name' => env('MEDIA_DISK', config('shop.media_disk')),
10
11     /*
12         * The maximum file size of an item in bytes.
13         * Adding a larger file will result in an exception.
14         */
15     'max_file_size' => 1024 * 1024 * 256, ←
16
17     /*
18         * This queue will be used to generate derived and responsive images.
19         * Leave empty to use the default queue.
20         */
21     'queue_name' => '',
22
23     /*

```

After that, run this command from the `chawkbazar-api` folder,

```
php artisan optimize:clear
```

And for `admin`, increase `timeout` from,

```
admin/rest/src/utils/api/http.ts
```

```
You, 1 second ago | 3 authors (explitan and others)
1 import { getAuthCredentials } from "@utils/auth-utils";
2 import { ROUTES } from "@utils/routes";
3 import axios from "axios";
4 import Cookies from "js-cookie";      Bashar Nozibulla, 8 months ago ·
5 import Router from "next/router";
6
7 const http = axios.create({
8     baseURL: process.env.NEXT_PUBLIC_REST_API_ENDPOINT, // TODO: take this from env
9     timeout: 600000, ←
10    headers: {
11        "Accept": "application/json",
12        "Content-Type": "application/json",
13    },
14});
15
16 // Change request data/error here
17 http.interceptors.request.use(
18    (config) => {
19        const { token } = getAuthCredentials();
20        config.headers = {
21            ...config.headers,
22            Authorization: `Bearer ${token}`,
23        };
24        return config;
25    }
26);
```

```
timeout: 600000
```

After that [Rebuild Your Project](#) and restart again.

## How to resolve docker: invalid reference format: repository name must be lowercase.?

Please open your terminal and use this command,

```
brew services stop mysql
```

It'll stop any MySQL instance that is installed using brew.

Then use this command to check is there MySQL running or not,

```
mysql -v
```

If no MySQL is running then, go to,

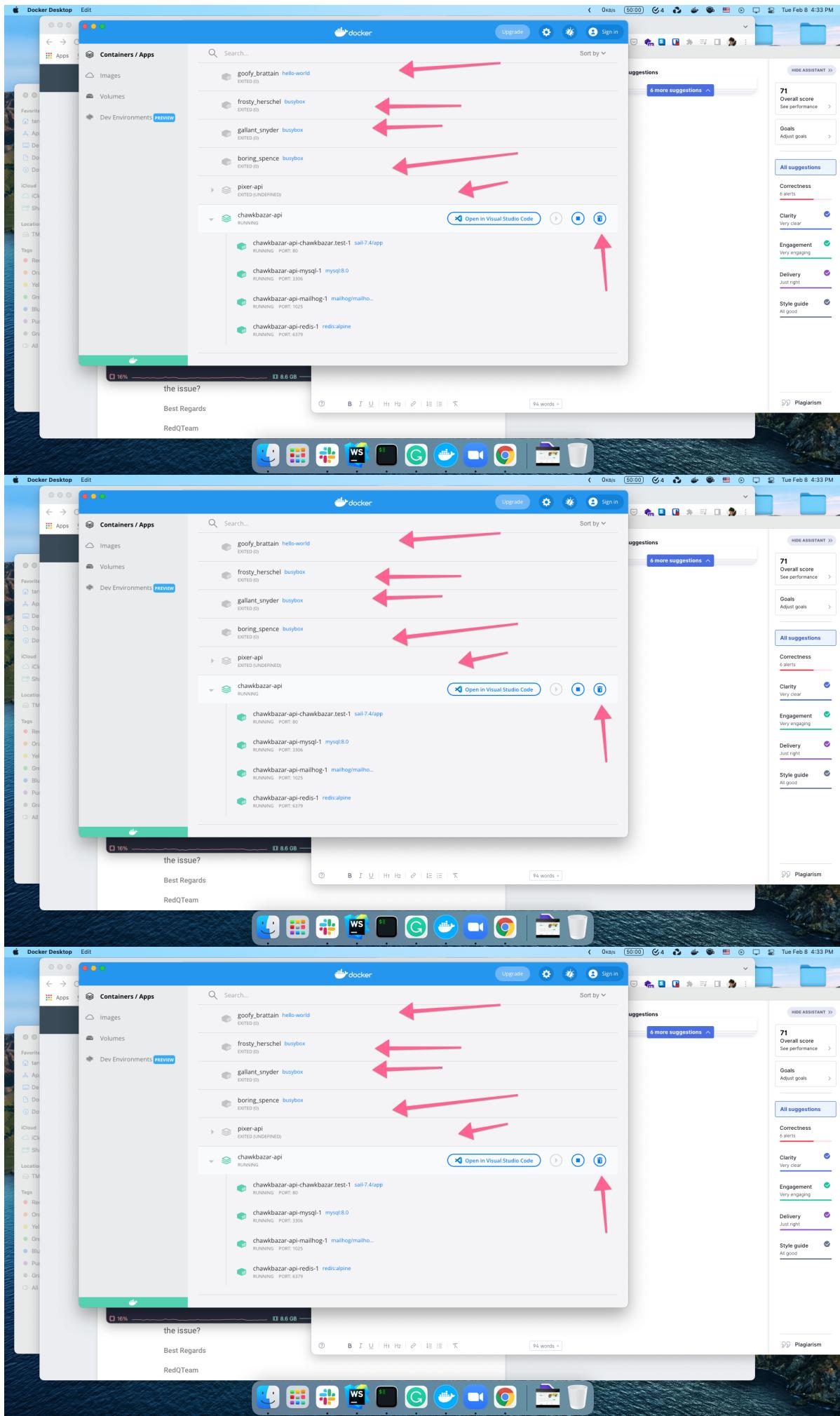
```
chawkbazar-laravel -> chawkbazar-api
```

and use this command,

```
./vendor/bin/sail down -v
```

After that,

Open docker dashboard and delete all the existing container and images,



And after that remove,

```
chawkbazar-laravel -> chawkbazar-api -> .env
```

And then go to your root `chawkbazar-laravel` folder and use this command again,

```
chmod +x install.sh  
bash install.sh
```

## How to remove the existing payment gateway?

You can remove the Stripe Payment from the available payment array,

```
shop/src/components/checkout/payment/payment-grid.tsx
```

## Why checkout Place Order button is disabled?

The place order button will be activated only when you fill in this information,

- OTP (Mobile number)
- Billing Address
- Shipping Address
- Delivery Schedule
- Payment Gateway

## How do I log in to the admin panel as the main administrator?

When you install the API using `php artisan marvel:install` or using `bash install.sh ( docker )`, you'll get a prompt to create an admin account, and you've to create admin credentials that time. And then, you can use that credentials for the main admin account.

## How to disable product popup view and redirect to a single product page?

At first edit `shop/src/components/product/product-card.tsx`

Import these two lines,

```
import { useRouter } from "next/router";  
import { ROUTES } from "@lib/routes";
```

```

product-card.tsx
1 import cn from "classnames";
2 import Image from "next/image";
3 import type { FC } from "react";
4 import { useUI } from "@contexts/ui.context";
5 import usePrice from "@lib/use-price";
6 import { Product } from "@framework/types";
7 import { siteSettings } from "@settings/site.settings";
8 import { useRouter } from "next/router";
9 import { ROUTES } from "@lib/routes";
10
11 interface ProductProps {
12   product: Product;
13   className?: string;
14   contactClassName?: string;
15   imageContentClassName?: string;
16   variant?: "grid" | "gridSlim" | "list" | "listSmall";
17   imgWidth?: number | string;
18   imgHeight?: number | string;
19   imgLoading?: "eager" | "lazy";
20 }
21
22 const ProductCard: FC<ProductProps> = ({(
23   product,
24   className = "",
25   contactClassName = "",
26   imageContentClassName = "",
27   variant = "list",
28   imgWidth = 340,
29   imgHeight = 440,
30   imgLoading,
31 )} => {
32   const router = useRouter();
33   const { openModal, setModalView, setModalData } = useUI();
34   const { name, image, min_price, max_price, product_type, description } =
35     product ?? {};
36
37   const { price, basePrice } = usePrice({
38     amount: product?.sale_price ? product?.sale_price : product?.price,
39     baseAmount: product?.price,
40   });
41
42   const { price: minPrice } = usePrice({
43     amount: min_price,
44   });
45
46   const { price: maxPrice } = usePrice({
47     amount: max_price,
48   });
49
50   function handlePopupView() {
51     router.push(`$${ROUTES.PRODUCT}/${product?.slug}`, undefined, {
52       locale: router.locale,
53     });
54   }
55
56   return (
57     <div
58       className={cn(
59         "group box-border overflow-hidden flex rounded-md cursor-pointer",
60       )}
61     >

```

Then call useRouter,

```
const router = useRouter();
```

```

product-card.tsx
23   product,
24   className = "",
25   contactClassName = "",
26   imageContentClassName = "",
27   variant = "list",
28   imgWidth = 340,
29   imgHeight = 440,
30   imgLoading,
31 )} => {
32   const router = useRouter();
33   const { openModal, setModalView, setModalData } = useUI();
34   const { name, image, min_price, max_price, product_type, description } =
35     product ?? {};
36
37   const { price, basePrice } = usePrice({
38     amount: product?.sale_price ? product?.sale_price : product?.price,
39     baseAmount: product?.price,
40   });
41
42   const { price: minPrice } = usePrice({
43     amount: min_price,
44   });
45
46   const { price: maxPrice } = usePrice({
47     amount: max_price,
48   });
49
50   function handlePopupView() {
51     router.push(`$${ROUTES.PRODUCT}/${product?.slug}`, undefined, {
52       locale: router.locale,
53     });
54   }
55
56   return (
57     <div
58       className={cn(
59         "group box-border overflow-hidden flex rounded-md cursor-pointer",
60       )}
61     >

```

Then update `handlePopupView` with this code,

```
router.push(`$${ROUTES.PRODUCT}/${product?.slug}`, undefined, {
  locale: router.locale,
});
```

```

product-card.tsx 1 x
    27 variant = "list",
    28 imgWidth = 360,
    29 imgHeight = 440,
    30 imgLoading,
    31 ) => {
    32   const router = useRouter();
    33   const { openModal, setModalView, setModalData } = useUI();
    34   const { name, image, min_price, max_price, product_type, description } =
    35     product ?? {};
    36
    37   const { price, basePrice } = usePrice({
    38     amount: product?.sale_price ? product?.sale_price : product?.price!,
    39     baseAmount: product?.price,
    40   });
    41
    42   const { price: minPrice } = usePrice({
    43     amount: min_price!,
    44   });
    45
    46   const { price: maxPrice } = usePrice({
    47     amount: max_price!,
    48   });
    49
    50   function handlePopupView() {
    51     router.push(`${ROUTES.PRODUCT}/${product?.slug}`, undefined, {
    52       locale: router.locale,
    53     });
    54   }
    55
    56   return (
    57     <div
    58       className={cn(
    59         "group box-border overflow-hidden flex rounded-md cursor-pointer",
    60         {
    61           "ltr:pr-0 rtl:pl-0 pb-2 lg:pb-3 flex-col items-start bg-white transition duration-200 ease-in-out transform h": variant === "grid",
    62           "ltr:pr-0 rtl:pl-0 md:pb-1 flex-col items-start bg-white": variant === "gridSlim",
    63           "items-center bg-transparent border border-gray-100 transition duration-200 ease-in-out transform hover:-trans": variant === "gridLarge"
    64         }
    65       )}
    66     </div>
  
```

Ln 57, Col 9 Spaces:2 UTF-8 LF () TypeScript React ✓ Spell ✓ Prettier R D

# Support

---

For support, Please open a ticket in the support forum

<https://redqsupport.ticksy.com/>

There are some basic requirements for our application so that support team can help you quick such as,

1. Asking queries regarding feature that is already implemented in the application
2. Following recommended configuration, environments & server which you need to met first before you proceed with installation, deployment in your server to receive support.
3. Support query need to be within Envato item support policy. ([https://themeforest.net/page/item\\_support\\_policy](https://themeforest.net/page/item_support_policy))
4. You should maintain only one support ticket at a time. Creating multiple ticket can cause unexpected delays.
5. Ticket should provide as much details as possible related to the issue such as screenshot, video explanation, access, how to reproduce, environments, if any changes or customisations are made etc

During working days our ticket response can take 24 hours to 48 hours depending on the volume of tickets pending prior to your ticket . We follow Envato Item support policy [https://themeforest.net/page/item\\_support\\_policy](https://themeforest.net/page/item_support_policy) to provide standard support for our items

# **Thank You!**

---