Backend Configuration

1. Rename .env.example to .env (use .env not .env.local it's only for backend).

2. Configure your MongoDB database, watch this video MongoDB, after configuring you will find a mongo URI just put that on your .env file MONGO_URI variable.

3. The JWT_SECRET is just a random value for creating a secret token, you can use whatever you want but make sure it is secret.

4. You need an email and password for using email verification and forget the password option. Use an email that you want to send messages to others when they register or request to forget the password. We use Nodemailer and the default email server for this. watch this video to create an app password for email app-password. After that put your email in the .env file EMAIL_USER and app password in the EMAIL_PASS variable.
Also, need to **Allow less secure apps to be ON, and access captcha for using this in production environment, see this doc**

4. Use your local server URL in STORE_URL variable, when you run on the local server your URL will be http://localhost:3000,

Finally, your .env file will look like this:

```
PORT=5055
MONGO_URI=your mongodb uri
JWT_SECRET=alamsfdfsdfsdfsdrafdar!@#$0dlfgjgsdfdsfdsfds
JWT_SECRET_FOR_VERIFY=lfjfjasjfr09ri09wrilfdjdj

SERVICE=gmail
EMAIL_USER=your email  //change with your sender email
EMAIL_PASS=you email app password  //change with your email app password
HOST=smtp.gmail.com
EMAIL_PORT=465

//use this when in dev/local server but when you will run on production/ live server
then use that URL/domain in here and put that live URL on environment variable when
hosting this backend

STORE_URL= http://localhost:3000
```

Once you successfully connect with MongoDB and configure .env then run "npm run data:import", it will run seed.js file and will import all demo data on the database. (You will find all demo data in the utils folder, change that data according to your need, also use staff email with real email for use of the forgetting password option) If everything is okay, then the backend configuration is done. Now you will find all demo data in your MongoDB database.

Store Configuration

1. Rename .env.example to .env.local

2. Please watch this video for Cloudinary configuration Cloudinary configuration, (We use Cloudinary for profile image upload).

3. This video for google client id Google Client ID (For google sign in),

4. Also need a stripe API key, for using the stripe payment option. If you want this, then go to react-stripe and create an account, get your test stripe API key, and put that on the .env.local variable. But if you do not add this value on the .env.local file then stripe will not work, others will work fine.

After Configure your .env.local file will look like this:

NEXT_PUBLIC_STRIPE_KEY="your stripe key" //for use stripe, change with your stripe API key

NEXT_PUBLIC_API_BASE_URL=http://localhost:5055/api //base API URL, when run on localhost/dev server.

NEXT_PUBLIC_CLOUDINARY_URL=https://api.cloudinary.com/v1_1/**your-cloudinary-user-name**/image/upload

NEXT_PUBLIC_CLOUDINARY_UPLOAD_PRESET=fg1vfge //your cloudinary upload preset

NEXT_PUBLIC_GOOGLE_CLIENT_ID=72898gfgdf0628gf79-jvugigp1d16rr0nf5hmvugfgtkiuogfgfh1ch.apps.goofgleusercontent.com //this one for google sign in, change with your google client API key

Deploy On Vercel

As a next.js project vercel is recommended for deployment because If you deploy on vercel then it will automatically do everything for us and there will be no need for customization.

Here is your guide for deploying KachaBazar

1. Create a GitHub account, go to vercel and sign up with that GitHub account.

2. Create three private repositories on GitHub, then push your store client code in one and backend code in another repository.

3. Watch this video deploy on vercel, do according to.

4. When you import your GitHub repository on vercel by creating a project, you will see an option for Environment Variables, just click on that and give you a local .env all variable with the value. then click on deploy.

First you have to deploy a backend project.

5. After the backend is deployed successfully, you will find a URL for your API route that will like this https://kachabazar.vercel.app/, and now change that like this https://kachabazar.vercel.app/api and use this as a NEXT_PUBLIC_API_BASE_URL when you deploy your store project.

6. Now create another project for store same as the backend project and put all the .env.local variables before clicking on deploy button, then click deploy, it will take some time for build and after that build, you will see your live version of Kacha Bazar store

7. If you do accordingly, then everything will be okay, for now when you make any changes on your local file, you just need to push your code on GitHub, vercel will automatically detect those changes and will redeploy your project with updated features.

You will find many videos on youtube and also articles on google about how to deploy next.js and express apps on vercel.