

# ShopNx Documentation

---

## Installation Instructions

### Softwares

- NodeJS (Web Server) (<https://nodejs.org/en/>)
- MongoDB (Database) (<https://www.mongodb.com/>)

### Start mongodb in a separate shell

In Windows operating system we can start it by opening the following file

```
C:/Program Files/MongoDB/Server/3.2/bin/mongod.exe
```

### Run the following 2 commands

This will install the required node dependencies and start the Server at <http://localhost:4200>

```
npm i  
npm start
```

### Building files for production server

This will generate both client and server files inside dist directory which can be directly copied to production server

```
npm run prod
```

# Client Settings

## Path:

*client/app/settings.ts*

## Enable / Disable demo mode

Restricts users from saving data if enabled

```
static demo: boolean = false;
```

## List of order status for Order Management

```
static orderStatus: string[] = ['Payment Pending', 'Order Placed', 'Paid', 'Order
```

## Enabled Payment Methods for ecommerce

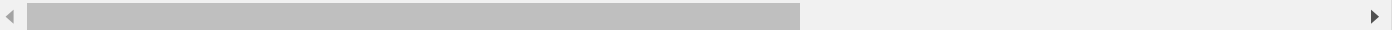
```
static paymentMethods: any = ['PayPal', 'COD'];
```

## Regional Settings

```
static country: any = { name: 'India', code: 'IN' };
static currency: any = {
  symbol: 'Rs', //Required only for sort icons at homepage
  code: 'INR', // Shop currency
  paypal: 'USD', // Paypal currency code *** Please choose from https://develo
  exchange_rate: '0.015' // Paypal currency code(USD) / Shop currency (INR) *
};
```

## Menu for Dashboad and Header

```
static menu: any[] = [
  { name: 'Products', url: '/admin/product', icon: 'store', authenticate: 'ad
  { name: 'My Orders', url: '/admin/my-orders', icon: 'watch_later', authenti
  { name: 'Manage Orders', url: '/admin/manage-orders', icon: 'history', auth
  { name: 'Address', url: '/admin/address', icon: 'location_city', authentica
  { name: 'Reviews', url: '/admin/reviews', icon: 'stars', authenticate: 'mar
  { name: 'My Reviews', url: '/admin/my-reviews', icon: 'star_rate', authenti
  { name: 'My Wishlist', url: '/admin/wishlist', icon: 'favorite', authentica
  { name: 'Media Library', url: '/admin/media-library', icon: 'perm_media', a
  { name: 'Brands', url: '/admin/brands', icon: 'wb_auto', authenticate: 'mar
  { name: 'Categories', url: '/admin/categories', icon: 'view_comfy', authent
  { name: 'Features', url: '/admin/features', icon: 'check_circle', authentic
  { name: 'Coupons', url: '/admin/coupons', icon: 'style', authenticate: 'adm
  { name: 'Shippings', url: '/admin/shippings', icon: 'local_shipping', authe
  { name: 'Users', url: '/admin/users', icon: 'face', img: 'auth.png', auther
  { name: 'Media Library', url: '/admin/media-library', authenticate: 'user',
  { name: 'Profile', url: '/account/edit-profile', authenticate: 'user', icon
  { name: 'Change Password', url: '/account/change-password', authenticate: '
  { name: 'Logout', url: '/account/logout', authenticate: 'user', icon: 'vpn_
];
```



# Server Settings

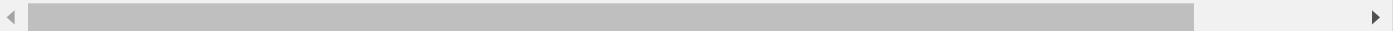
Path: *server/config.ts*

## Website Settings

```
smsEnabled = true;  
emailEnabled = true;
```

## Review Settings

```
reviewSettings = {  
  enabled: true, // Enables review for products  
  moderate: false // If enabled, the review will be visible to public after admin  
};
```

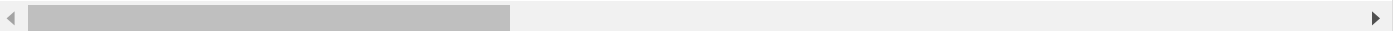


## Product Settings

```
product = { moderate: false };
```

## User Roles

```
userRoles = ['user', 'manager', 'admin']; // This should be in ascending order of
```



## Forgot Password Email Settings

```
forgotPasswordEmail = (body) => { // Expects email id and password reset token
  return {
    from: 'passwordreset@codenx.com',
    to: body.email,
    subject: 'ShopNx Password Reset Request',
    text: 'You are receiving this because you (or someone else) have requested th
    'Please click on the following link, or paste this into your browser to compl
    process.env.DOMAIN + '/account/reset-password/' + body.token + '\n\n' +
    'If you did not request this, please ignore this email and your password will
  }
}
```

## Reset Password Email Settings

```
resetPasswordEmail = (body) => { // Expects email id and name
  return {
    from: 'passwordreset@codenx.com',
    to: body.email,
    subject: 'ShopNx Password Changed',
    text: 'Hello,\n\n' +
    'This is a confirmation that the password for your account ' + body.to + ' ha
  };
}
```

## Order Placed Email Settings

```
orderPlacedEmail = (body) => { // Expects email id, orderNo, ...
  return {
    from: 'CodeNx <admin@codenx.com>',
    to: body.to,
    subject: 'Order Placed Successfully',
    text: 'Order No: ' + body.orderNo
    + '\n Status: ' + body.status
    + '\n\n Payment Method: ' + body.payment_method
    + '\n\n Payment ID: ' + body.id
    + '\n Amount: ' + body.amount.currency + ' ' + Math.round(body.amount.total)
    + '\n\n Name: ' + body.address.recipient_name
    + '\n Address: ' + body.address.line1
    + '\n City: ' + body.address.city
    + '\n Zip: ' + body.address.postal_code
  };
}
```

## Order Updated Email Settings

```
orderUpdatedEmail = (body) => {
  return {
    from: 'CodeNx <admin@codenx.com>',
    to: body.to,
    subject: 'Your Order Status Updated',
    text: 'Order No: ' + body.orderNo
    + '\n Status: ' + body.status
    + '\n\n Payment Method: ' + body.payment_method
    + '\n\n Payment ID: ' + body.id
    + '\n Amount: ' + body.amount.currency + ' ' + Math.round(body.amount.total)
    + '\n\n \n Name: ' + body.address.recipient_name
    + '\n Address: ' + body.address.line1
    + '\n City: ' + body.address.city
    + '\n State: ' + body.address.state
    + '\n Zip: ' + body.address.postal_code
  };
}
```

# Environment Settings

Rename `.env.sample` to `.env` and change according to your generated credentials

Generate the following APIs and input the credentials into `.env` file present at the root of the project

## Email Client

[Sendgrid](#)

## Payment Gateway

[Paypal Developer](#)

## Login through Facebook

[Facebook Developer](#)

## Google Auth Login

[Google Developer Console](#)

## Use twitter account to login to ShopNx

[Twitter Developer](#)

```
MONGODB_URI=mongodb://localhost:27017/shopnx-dev
DOMAIN=http://localhost:4200
SESSION_SECRET=shopnx-secret
PAYPAL_MODE=sandbox
PAYPAL_CLIENT_ID=YOUR_PAYPAL_CLIENT_ID
PAYPAL_CLIENT_SECRET=YOUR_PAYPAL_CLIENT_SECRET
STRIPE_APIKEY=sk_test_REST_OF_YOUR_KEY
SENDGRID_APIKEY=YOUR_SENDGRID_API_KEY
FACEBOOK_ID=YOUR_FACEBOOK_ID
FACEBOOK_SECRET=YOUR_FACEBOOK_SECRET
TWITTER_ID=YOUR_TWITTER_ID
TWITTER_SECRET=YOUR_TWITTER_SECRET
GOOGLE_ID=YOUR_GOOGLE_ID
GOOGLE_SECRET=YOUR_GOOGLE_SECRET
GOOGLE_MAPS_API=YOUR_GOOGLE_MAPS_API
```

# Project Root Directory Structure

```
1  +---client
2  +---node_modules
3  +---server
4  +---uploads
5  |   .angular-cli.json
6  |   .editorconfig
7  |   .env
8  |   .gitignore
9  |   karma.conf.js
10 |   package.json
11 |   protractor.conf.js
12 |   proxy.conf.json
13 |   tsconfig.json
14 |   tslint.json
```



# Client Directory Structure

```
1  |──app
2  |   |──account
3  |   |   |──cp
4  |   |   |──login
5  |   |   |──password
6  |   |   |──profile
7  |   |──admin
8  |   |   |──address
9  |   |   |──brands
10 |   |   |   |──brand-detail
11 |   |   |──categories
12 |   |   |──coupons
13 |   |   |   |──coupon-detail
14 |   |   |──dashboard
15 |   |   |──features
16 |   |   |   |──feature-detail
17 |   |   |──media
18 |   |   |──my-orders
19 |   |   |──my-reviews
20 |   |   |──order
21 |   |   |   |──order-content
22 |   |   |──product
23 |   |   |   |──features
24 |   |   |   |──product-detail
25 |   |   |   |   |──variants
26 |   |   |──reviews
27 |   |   |   |──review-detail
28 |   |   |──shipping
29 |   |   |   |──shipping-detail
30 |   |   |──user
31 |   |   |   |──user-detail
32 |   |   |──wishlist
33 |   |──home
34 |   |   |──banner
35 |   |   |──checkout
36 |   |   |──featured-products
37 |   |   |──home
38 |   |   |──megamenu
39 |   |   |──news-banner
```

```
40 | | | | owl-carousel
41 | | | | product
42 | | | | product-card
43 | | | | shop
44 | | | | success
45 | | | | wish-button
46 | | | modal
47 | | | shared
48 | | | | 404
49 | | | | address
50 | | | | cart-buttons
51 | | | | dialogs
52 | | | | edit
53 | | | | export
54 | | | | footer
55 | | | | guards
56 | | | | header
57 | | | | list
58 | | | | list-image
59 | | | | media
60 | | | | oauth-buttons
61 | | | | pipes
62 | | | | search
63 | | | | services
64 | | | | submit-button
65 | | assets
66 | | | fashion
67 | | | img
68 | | environments
69 | | hmr.ts
70 | | index.html
71 | | main.ts
72 | | output.txt
73 | | polyfills.ts
74 | | test.ts
75 | | tsconfig.app.json
76 | | tsconfig.spec.json
77 | | typings.d.ts
```

# Server Directory Structure

```
1  |──api
2  |   |──address
3  |   |──brand
4  |   |──category
5  |   |──contact
6  |   |──coupon
7  |   |──customer
8  |   |──feature
9  |   |──media
10 |   |──order
11 |   |──orderHistory
12 |   |──pay
13 |   |──payment-method
14 |   |──product
15 |   |──review
16 |   |──sendmail
17 |   |──shipping
18 |   |──upload
19 |   |──user
20 |   └──wishlist
21 |──auth
22 |   |──facebook
23 |   |──google
24 |   |──local
25 |   |   └──.git
26 |   |       |──hooks
27 |   |       |──info
28 |   |       |──objects
29 |   |       |   |──40
30 |   |       |   |──97
31 |   |       |   |──info
32 |   |       |   └──pack
33 |   |       └──refs
34 |   |           |──heads
35 |   |           └──tags
36 |   └──twitter
```

```
37  └── components
38      └── errors
39  |    app.ts
40  |    config.ts
41  |    routes.ts
42  |    seed.ts
43  |    tsconfig.json
```

# Usage Instructions

## Home

This is the main page of our Angular e-commerce store. Here we get

- List of all products
- Filter Products: based on Price (Price Slider), Brand, Features (Color, Type, Fit, Fabric, Neck)
- Sort: Based on Price and Name

Each product contains a add to cart button. Once the product is added into the cart, we get the increase or decrease cart

quantity option By clicking each product we arrive at the product detail page

## Product Details

This page presents the complete details of the product

- Product name
- Description
- Price, MRP
- Product Image (Including additional images)
- Brand
- Category
- Quantity in cart
- Size
- Features

## Search

The top navigation bar of the website has a search box which autocompletes with product info while user starts typing. By clicking a suggested item in the search bar, the page navigates to the product details page of the selected product.

## Category

Get the current category name with all products under it  
This page also has all the filter and sort options.

## Shopping Cart

This store is featured with a shopping cart facility which is easy to use and fast.

- Get quick summary of what is there in Cart
- Modify the cart quantity
- Checkout using Paypal

## Login / Signup

Features like Signup / SignIn / Change Password / Logout is integrated into this application already with high level of security,  
so that you no longer need to be worry about implementing all those features into the application

A user need **not** have to navigate to a separate page to login or signup. It comes as a popup which is a huge ui improvement.

This login popup has a advantage of popping out for any route when a guest user tries to access a restricted page Both the  
login and signup page has the option for connect using facebook, twitter, google as well

## Checkout

- The checkout page Displays the Order Amount + ShipCharge
- This also has an option discount coupons which is appif valid.
- Here the user can choose the Payment options (CasDelivery, Paypal)
- This page automatically choose the best available ShipOptions based on the total order weight and the shipper availability.
- While checkout the user can choose from any saddress.

## Address Management

The address management is integrated into the checkout page to make the checkout experience single view and easy. Here the user can store and manage different addresses.

## Order Management

The user has the facility to view the order history. Administrators can change order status + payment status

**Users:** All the orders placed by the logged in user is available in this view.

**Administrators:** This view presents all orders placed by users with the option to change order status and shipping

## User Management

**Users:**

Change Password

Forgot Password

**Administrators:**

User role management

## Media Library

- Now the shop has a new media library where the managers can upload any image that is to be used in the shopping application
- Clicking on each image displays the details about it as well as an option to delete it

## Products (Role: Managers, Administrators)

Product details can be added, modified and deleted using this page. Each product can be associated into a single Brand, Category A product can have

- Multiple features
- Multiple key features
- Multiple product images
- The list contains all the available products with a search box to filter the list.
- Clicking on a product at the product list will pop up the details of the product at the right sidebar
- The right sidebar has option to change product details, brand, category
- This sidebar also contains a module to manage product images

## Manage Brands

Administrators can add, edit, delete, filter brands of their store from this view

## Manage Categories

- Categories are presented in Parent-Child manner in store for better organisation of products.
- Store's navigation bar at top contains all the categories arranged in parent-child fashion.
- This view provides facility to add both parent and child categories, re-arrange category association according to their requirement.