1. **Getting start**

* OS: Recommend Linux system
* Guide for ubuntu system
* Need 4 domains and https for all. You should register wild card for SSL. Go to your DNS manager and point 3 domains above to your server
* Web user: <https://yourapp.com>
* Admin: <https://admin.yourapp.com>
* API: <https://api.yourapp.com>

1. **Softwares**
2. [**NodeJS**](https://nodejs.org)

* Version >= 8.1
* More information you can check [here](https://nodejs.org/en/download/package-manager/)
* Install

Open your terminal and run these commands

* $ curl -sL https://deb.nodesource.com/setup\_8.x | sudo -E bash -
* $ sudo apt-get install -y nodejs

After that you can check install

* $ node -v

1. [**MongoDB**](https://www.mongodb.com/)

* Version >= 3.6
* More information you can check [here](http://alendar.google.com/calendar/r?pli=1#main_7)
* Install

Open your terminal and run these commands

* $ sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 9DA31620334BD75D9DCB49F368818C72E52529D4
* $ echo "deb [ arch=amd64 ] https://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/4.0 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.0.list
* $ sudo apt-get update
* $ sudo apt-get install -y mongodb-org

After that you can check install

* $ mongo -v

1. [**Graphicsmagick**](http://www.graphicsmagick.org/)

* More information you can check [here](http://www.graphicsmagick.org/README.html)
* Install

Open your terminal and run these commands

* $ sudo apt-get update
* $ sudo apt-get install libpng12-0
* $ sudo apt-get install libjpeg-dev
* $ sudo apt-get install ghostscript
* $ sudo apt-get install libtiff5-dev
* $ sudo apt-get install libfreetype6
* $ sudo apt-get install libfreetype6-dev
* $ wget ftp://[ftp.graphicsmagick.org/pub/GraphicsMagick/1.3/GraphicsMagick-1.3.26.tar.gz](http://ftp.graphicsmagick.org/pub/GraphicsMagick/1.3/GraphicsMagick-1.3.26.tar.gz)
* $ tar xzvf GraphicsMagick-1.3.26.tar.gz
* $ cd GraphicsMagick-1.3.26
* $ ./configure
* $ make
* $ sudo make install

After that you can check install

* $ gm

1. [**Redis**](https://redis.io/)

* Install

Open your terminal and run these commands

* $ sudo add-apt-repository ppa:chris-lea/redis-server
* $ sudo apt-get update
* $ sudo apt-get install redis-server

After that you can check install

* $ redis-cli

1. [**Nginx**](https://www.nginx.com/)

* Install

Open your terminal and run these commands

* $ sudo apt-get install nginx

After that you can check install

* $ nginx -f

1. [**PM2**](http://pm2.keymetrics.io/)

* Install

Open your terminal and run these commands

* $ sudo npm install pm2 -g

After that you can check install

* $ pm2

1. [**Angular**](http://calendar.google.com/calendar/r?pli=1#main_7) **(you don’t need to install angular in your server, just use to build the app)**

* Your will need NodeJS version > 9.x to setup angular app. Please check step 1 but insall with Node JS 9.x
* Run these commands to install the app
* $ npm install -g @angular/cli

1. **Server**
2. **API**

* CD to API folder
* Create **[environment]**.json file in the server/config folder by copy development.json file (change [environment] by your environment)
* Create file .env in the root api folder. Then copy all content in to env.example file.
* Change all config by your config

# If live, it should production

NODE\_ENV=development

APP\_NAME=Escort

ALLOW\_CORS=true

# For live env, it should be 8080 or any port you want

PORT=8080

# This name will show in the Email content

SITE\_NAME=Genstore

# Mongo DB url

MONGO\_URI=mongodb://localhost/genstore-dev

## mailing

SENDGRID\_API\_KEY=xxxxx

SPARKPOST\_API\_KEY=xxx

# Just need spart post or sendgrid. Depend on your choosen service

MAIL\_SERVICE=sparkpost

# Sender email

mailFrom=admin@mail.com

# Email for contact page

ADMIN\_EMAIL=admin@genstore.com

# Information from Twilio config

SMS\_FROM=+13232xxxxx

SMS\_SID=ACaxxxxxxxxx

SMS\_AUTH\_TOKEN=950cd6xxxxxxxxx

* Save all information
* Start app by command **$ node server/www**
* If you installed pm2, you can use pm2 to start the app and keep in the background. Use command **$ pm2 start server/www --name=”api.yourapp.com”**
* You can check IV for API folder if needed

1. **Build web**

* Create your application folder in www like this
* Web: $ mkdir /var/www/yourapp.com/web
* Admin: $ mkdir /var/www/yourapp.com/admin
* API: $ mkdir /var/www/yourapp.com/api

1. **Web**

* Note: Must use node js 9 to build the app
* CD to web app
* Edit file src > environments > environment.prod.ts file
* Change all config by your api url and services key we have. It looks like

*export const environment = {*

*production: true,*

*version: '0.0.1',*

*build: 15,*

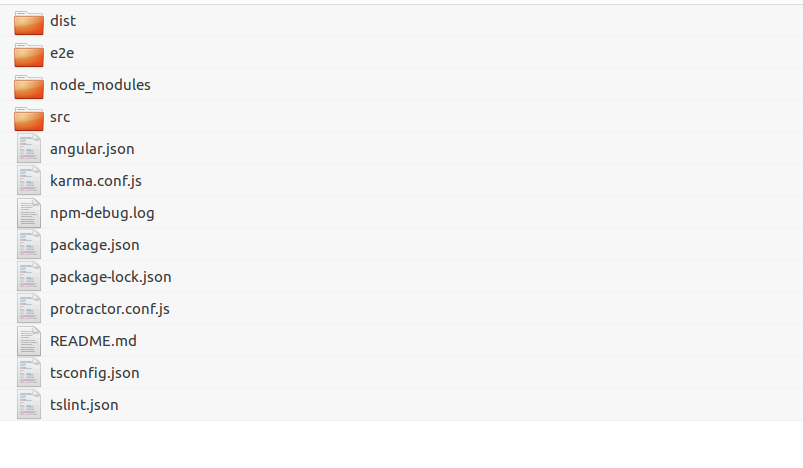
*// change this to API url*

*apiBaseUrl: '*[***https://api.yourapp.com****/v1*](https://api.yourapp.com/v1)*',*

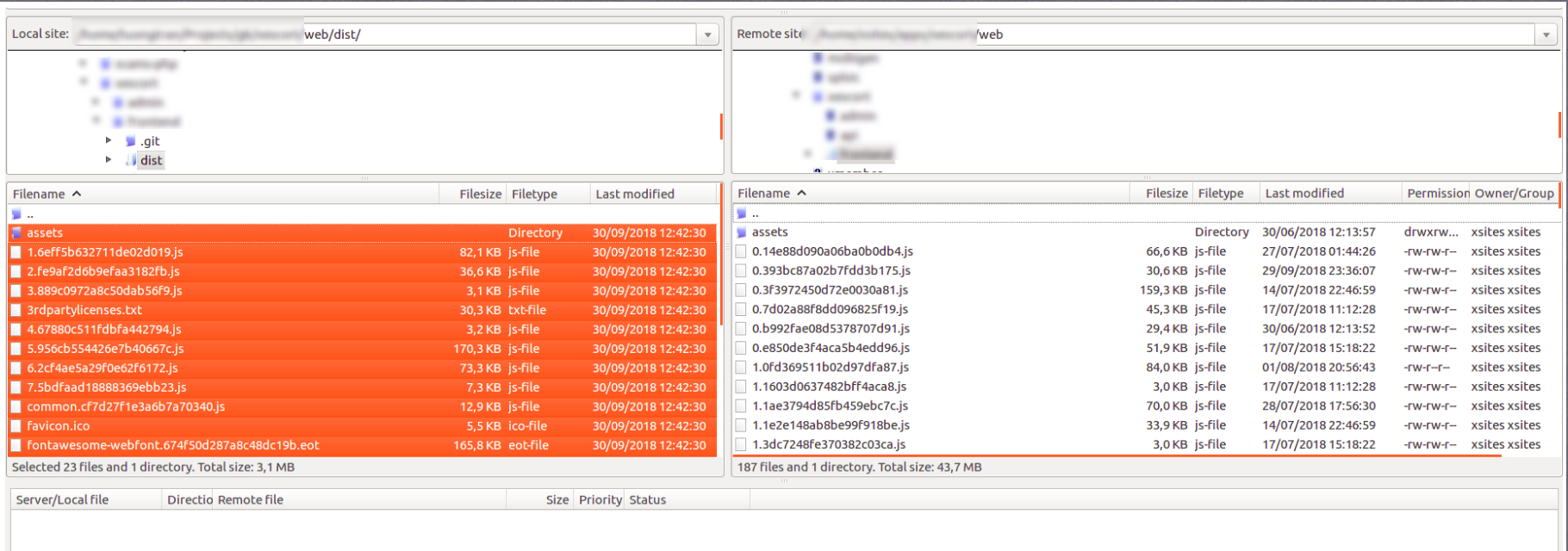
*platform: 'web'*

*};*

* Run command **$ npm run build**
* After that you will have a dist folder



* Upload all content in the dist folder to your web app folder in the server



* You can use [Filezilla](https://filezilla-project.org/) to upload

1. **Admin**

* Same Web app
* Upload all content in the dist folder to your seller app folder in the server

1. **Config nginx**

* Create new config file for your app in the /etc/nginx/sites-enabled folder. Like **/etc/nginx/sites-enabled/yourapp.com.conf**
* Edit your config look like this

*server {*

*listen 443;*

*ssl on;*

*ssl\_certificate /path/to/ssl/web/fullchain.pem;*

*ssl\_certificate\_key /path/to/ssl/web/privkey.pem;*

*server\_name yourapp.com;*

*location / {*

*root /var/www/yourapp.com/web;*

*index index.html index.htm;*

*try\_files $uri $uri/ /index.html?$query\_string;*

*}*

*location /api {*

*proxy\_pass http://localhost:8080;*

*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;*

*}*

*}*

*server {*

*listen 443;*

*ssl on;*

*ssl\_certificate /path/to/ssl/api/fullchain.pem;*

*ssl\_certificate\_key /path/to/ssl/api/privkey.pem;*

*server\_name api.yourapp.com;*

*client\_max\_body\_size 200m;*

*location / {*

*proxy\_pass http://localhost:8080;*

*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;*

*proxy\_redirect off;*

*proxy\_set\_header Host $host;*

*proxy\_set\_header X-Real-IP $remote\_addr;*

*proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;*

*proxy\_set\_header Proxy "";*

*}*

*proxy\_set\_header Host $http\_host;*

*proxy\_set\_header X-Forwarded-For $remote\_addr;*

*proxy\_set\_header X-Forwarded-Proto $scheme;*

*}*

*server {*

*listen 443;*

*ssl on;*

*ssl\_certificate /path/to/ssl/admin/fullchain.pem;*

*ssl\_certificate\_key /path/to/ssl/admin/privkey.pem;*

*server\_name admin.yourapp.comtop;*

*location / {*

*root /var/www/yourapp.com/admin;*

*index index.html index.htm;*

*try\_files $uri $uri/ /index.html?$query\_string;*

*}*

*}*

*server {*

*listen 80;*

*server\_name yourapp.com www.yourapp.com;*

*return 301 https://yourapp.com$request\_uri;*

*}*

*server {*

*listen 80;*

*server\_name api.yourapp.com www.api.yourapp.com;*

*return 301 https://api.yourapp.com$request\_uri;*

*}*

*server {*

*listen 80;*

*server\_name admin.yourapp.comtop www.admin.yourapp.comtop;*

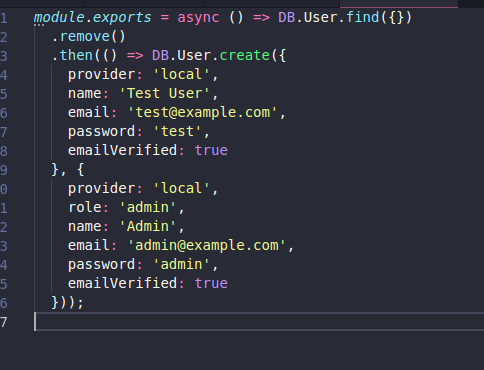
*return 301 https://admin.yourapp.comtop$request\_uri;*

*}*

* Change 8080 to your API port. Change domain, root directory in your app
* Restart your nginx **$ sudo service nginx restart**

1. **Seed data**

* CD to API folder
* Run $ export NODE\_ENV=production (or environment you are using)
* Open file server > migrations > user.js and change default account you want



* Run $ node server/migrate
* After that default data will be generate.

1. **Testing**

* Open your browser and test
* Web user: <https://yourapp.com>
* Admin: <https://admin.yourapp.com>
* Default admin account: [admin@example.com](mailto:admin@example.com) with password admin

1. **Other**
2. Use pm2

* Check pm2 services **$ pm2 ls**
* Reload your api app **$ pm2 reload [app-id]**

1. **Pre-render (for SEO)**

* Download source [here](https://www.dropbox.com/s/3gz5hnqkcrymowt/pre-render.tar.gz?dl=0)
* Run **$ npm install** to install
* Edit index.js file and change your default port
* Use pm2 to start prerender server. Like **$ pm2 start index.js --name=”pre-render-server”**
* Modify your nginx config (for web only) to content below (you can see all content in the README.MD file)

*server {*

*listen 443;*

*ssl on;*

*ssl\_certificate /path/to/fullchain.pem;*

*ssl\_certificate\_key /path/to/privkey.pem;*

*server\_name app;*

*set $prerender 0;*

*if ($http\_user\_agent ~\* "baiduspider|twitterbot|facebookexternalhit|rogerbot|linkedinbot|embedly|quora link preview|showyoubot|outbrain|pinterest|slackbot|vkShare|W3C\_Validator") {*

*set $prerender 1;*

*}*

*if ($args ~ "\_escaped\_fragment\_") {*

*set $prerender 1;*

*}*

*if ($http\_user\_agent ~ "Prerender") {*

*set $prerender 0;*

*}*

*if ($uri ~ "\.(js|css|xml|less|png|jpg|jpeg|gif|pdf|doc|txt|ico|rss|zip|mp3|rar|exe|wmv|doc|avi|ppt|mpg|mpeg|tif|wav|mov|psd|ai|xls|mp4|m4a|swf|dat|dmg|iso|flv|m4v|torrent|ttf|woff)") {*

*set $prerender 0;*

*}*

*location / {*

*root /var/www/app/web;*

*index index.html index.htm;*

*if ($prerender = 1) {*

*#setting prerender as a variable forces DNS resolution since nginx caches IPs and doesnt play well with load balancing*

*#resolve using Google's DNS server to force DNS resolution and prevent caching of IPs*

*# resolver 8.8.8.8;*

*rewrite .\* /$scheme://$host$request\_uri? break;*

*proxy\_pass http://localhost:8081;*

*}*

*try\_files $uri $uri/ /index.html?$query\_string;*

*}*

*}*