Deployment Guide

The system is being developed at this GitHub repo: https://github.com/charliewang95/plm

Basic Set-Up and Assumptions

- The system is supposed to be deployed on a Linux Ubuntu server. Current release is developed and tested on Ubuntu 16.04 LTS. Support for other versions of Ubuntu has not been verified yet.
- The following software are needed for deploying and running the project: git, npm, Node.JS, and MongoDB.

Set up your server

- We assume that the user have an account with root previlege on the server. We further assume that the user has either terminal or ssh access to the server.

Set up your server following these steps (adopted from https://poweruphosting.com/blog/initial-server-setup-ubuntu-16-04/):

1. Log in to your account with root previlege either with ssh or on the terminal. Note:
2. If the user account you have is `root`, we recommend creating another user and log in with that user instead. If you do not wish to do this, go to step 5
3. Create another user by typing `adduser <username>`. Follow the prompt on the command line to set up a new user. **Caveate: if for some reason your server uses Kerberos and you do not have the password for it, you can turn it off and modify the password of your account by entering these commands (adopted from https://unix.stackexchange.com/questions/116028/how-could-i-eliminate-kerberos-for-passwd:**
   ```
   sudo -s
   pam-auth-update
   passwd <username>
   ```
   During `pam-auth-update`, use the space bar to disable Kerberos authentification.
4. Grant your account administrator previleges by typing `sudo gpasswd -a <username> sudo`
5. If you wish, you can also add public key authentication and configure ssh daemon on your server by following Step 4 and 5 of this link. However, since they are not necessary for server setup we will not detail the steps here.

Configure the firewall for your server (adopted from https://www.digitalocean.com/community/tutorials/how-to-set-up-a-firewall-with-ufw-on-ubuntu-14-04)

- You may also need to configure the firewall of your server, either to allow the server to communicate to the outside world or enhance its security. The following steps allow you to configure the UFW, or Uncomplicated Firewall, for your server.

1. If UFW is not installed, install it by running `sudo apt-get install ufw`
2. If your server has IPv6 enabled, ensure that UFW is configured to support IPv6 so that it will manage firewall rules for IPv6 in addition to IPv4. Open the UFW configuration by typing `sudo`

`nano /etc/default/ufw`. Then, make sure the value of `IPV6` is equal to `yes`. It should look like this:
```
...
IPV6=yes
...
```
Save and quit.
3. Check the status of UFW by typing `sudo ufw status verbose`
4. Configure your UFW by specifying the type of connection and its policy. Some examples are given below
```
sudo ufw allow ssh   # allow ssh connections
sudo ufw allow http  # allow http connections on port 80
sudo ufw allow https # allow https connections on port 443
```
5. Enable the UFW by typing `sudo ufw enable`

- For more options regarding setting your firewall, refer to this link.

Set up the development environment

Install Node.js (adopted from http://mean.io/blog/)

- In order to manage Node.js we will use the **Node Version Manager** (NVM). Follow these steps to install the NVM (adopted from the README at https://github.com/creationix/nvm#installation):

1. Run `curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.8/install.sh | bash`
2. Run this command: `export NVM_DIR="$HOME/.nvm"`
```
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh" # This loads nvm
```
3. Verify that nvm is installed by typing `command -v nvm`. On Linux if you get `nvm: command not found` or see no feedback from the terminal, close the current terminal, open a new terminal, and try verify again. For more troubleshooting tips and information see this link
4. Install Node.js 6.11.4 by running this command `nvm install 6.11.4`. After the installation completes, use installed version of Node 6.11.4 by running `nvm use 6.11.4`.

Install MongoDB (adopted from http://mean.io/blog/)

1. Import the GPG keys for the official MongoDB repository by running `sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv EA312927`
2. Add MongoDB repository details to your system by running `echo "deb http://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/3.2 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.2.list`
3. Update the package list by running `sudo apt-get update`
4. Install the MongoDB package itself: `sudo apt-get install -y mongodb-org`
5. Add MongoDB as a service in Ubuntu by typing this command `sudo nano /etc/systemd/system/mongodb.service`
Paste the following instructions to the file you just opend by running the above command, save it, and close it.
```
[Unit]
Description=High-performance, schema-free document-oriented database
After=network.target

[Service]
```

```
User=mongodb
ExecStart=/usr/bin/mongod --quiet --config /etc/mongod.conf

[Install]
WantedBy=multi-user.target
```

6. Start the MongoDB instance: `sudo systemctl start mongodb`
7. Check the status of MongoDB by running this command: `sudo systemctl status mongodb`
8. If you wish to start MongoDB on system boot, use the following command: `sudo systemctl enable mongodb`

Deploy our system

1. Clone the repository from GitHub by running `git clone https://github.com/charliewang95/plm.git`
2. Check out the production server by running `git fetch` `git checkout production-server`
3. Run `sudo npm install` to install all dependencies
4. If you wish to run the server locally and access it as localhost, remove the `.env` file. If you wish the system to be accessed on the Internet, open the `.env` file (create one if does not exist), and add the following entry: `HOST=<your-domain-url>`
5. Start the server by running the command `sudo npm run start-https`. This will start the system while the terminal is open and running. Addtionally, if you want to just start the system and have it operate after you close the terminal, run `sudo nohup npm run start-https >/dev/null 2>&1 &`. **Note: It is strongly recommended thta you run `sudo npm run start-https` first to make sure the system operates correctly, as the second command will produce no output on the terminal.**

- Additionally, 2 scripts are provided to ease shutting down and starting the server. Running `./startNonStoppingServer.sh` will start the server that will not be shut down when the terminal is closed; Running `./shutdownServer.sh` shuts down the server by killing all processes of the system

Migrate backed-up data to the system (Optional)

After completion of the above steps that specify how to deploy our system, if you have a copy of backed up data (i.e. bson file produced by our backup system using mongodump) and wish to migrate the data to your newly deployed system, follow the steps below. This portion is adopted largely from our backup guide.

Configuring the database

1. `ssh` into the server, and start the mongo shell by typing `mongo`.
2. Create the user administrator.

We want to create a user with the `userAdminAnyDatabase` role in the `admin` database. This user has privileges over creating users in other databases. Switch to the `admin` database by typing `use admin` in the mongo shell. Then, type the following command in the mongo shell:

```
db.createUser(
  {
    user: <username>,
    pwd: <password>,
    roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]
  }
)
```

`<username>` is a string such as `"myUserAdmin"`, and `<password>` is a string such as `"abc123"`.

(Note: You can check for existing users in the database by typing `db.getUsers()`. If you would like to delete any existing user, use `db.dropUser(<username>)`)

1. Create a user for your database. In our case, we want to add a user `plmUser` to our database `plm`. The user will have `readWrite` role in the `plm` database.

```
use plm
db.createUser(
  {
    user: "plmUser",
    pwd: <plmPassword>,
    roles: [ { role: "readWrite", db: "plm" } ]
  }
)
```

`<plmPassword>` is a the password for your user, and it is a string such as `"abc123"`.

1. Enable Auth and Open MongoDB access

Edit your MongoDB config file for the mongodb service. First, find out the the path of your config file by viewing the script to start your mongodb service. On our system, we typed `vi /etc/init.d/mongodb` . Inside the file you will see a line `CONF=<path-to-config-file>`. In our case `<path-to-config-file>` is `/etc/mongodb.conf`. Open the file by typing `sudo vi <path-to-config-file>`. Inside the file, uncomment the line `auth = true`, and change the line specifying `bind_ip` to `bind_ip = 0.0.0.0`.

1. Open port 27017 on your server by typing `sudo ufw allow 27017`
2. Restart the mongodb service by typing `sudo systemctl restart mongodb`. Make sure you can still log in with mongo while ssh'd into the server.

Note: you can try to log in as the newly created `plmUser` by typing `mongo -u "plmUser" -p <plmPassword> --authenticationDatabase "plm"` on the server.

Restoring data from backup

If the backup is stored in the backup system, and the system is configured correctly according to our backup guide, you can follow section **Restoring from a backup** in our backup guide. Otherwise, follow the steps listed below:

1. Copy the backup folder to the server, possibly using `scp`
2. Log in to the server. Run the command `mongorestore --username "plmUser" --password "<plmUserPassword>" --authenticationDatabase plm <path-to-backup-folder> --drop --objcheck`. The `--drop` opiton will drop the database before attempt to restore, and `objecheck` will check for validity while attempting to restore.
3. You will see command line outputs that indicate whether the restore was successful. In case of schema unmatch, you will also be notified on the command line.
4. Further validate your data by running `db.plm.validate({full:true})` in your mongo shell. If you see `valid: true` in the output, the data is valid to use.

Troubleshooting

- If when starting the server you get the following error`Module not found: Can't resolve 'react-transition-group/CSSTransition' in '/home/plm/node_modules/material-ui/transitions'`
  Please try running the this command `sudo npm install react-transition-group@^1`. This will resolve the error. (from https://github.com/reactjs/react-transition-group/issues/36)