# Deploy Documentation

## Dev server:

Running the backend server on dev is simple. In the route directory, navigate to config/config.js and modify the ‘development’ property to match your needs. For example:

development: {

dialect: "postgres",

username: “postgres”,

password: “postgres”,

database: “api”,

host: “localhost”,

saltRounds: 2,

jwtSecret: 'yo-its-a-secret',

tokenExpireTime: '6h',

},

Dialect, username, password, and database references your database connection, and the saltRounds, jwtSecret, and tokenExpireTime creates the JSON Web Token layer.

In Frontend/dashboard, adjust your “proxy” settings to reflect the localhost that your dev backend server is running on.

## Production Server:

Setting up the production server is the same (make sure you adjust the “proxy” to match). The current production server is running on an Azure VM in two separate Docker containers. In the VM there is a build\_prod.sh bash script that runs a docker-compose.yml command to build up the docker containers under the same network connection. To start up the docker containers,

1. Run sudo docker build -t crown-prod . in the crownCenter directory
2. Run sudo docker build -t crown-client . in the crownCenter/Frontend/dashboard directory
3. Run ./build\_prodh.sh in the crownCenter directory to start the docker containers

On subsequent server side builds, you need to restart the crown-prod docker container for changes to register and update. Changes you make to the UI do not require rebuilds, unless you install a new node module.