**Prerequisites:**

The best practices in this article assume that you have:

* Node.js and npm installed.
* An existing Node.js app.
* A free Heroku account.
* The Heroku CLI.

**Deployment Steps:**

1. **Declare app dependencies**

Heroku Node.js support will only be applied when the application has a package.json file in the root directory.

Make sure that you are not relying on any system-level packages. Missing dependencies in your package.json file will cause problems when you try to deploy to Heroku. To troubleshoot this issue, on your local command line, type rm -rf node\_modules; npm install --production and then to try to run your app locally by typing heroku local web. If a dependency is missing from your package.json file, you should see an error that says which module cannot be found.

1. **Specify the version of node :**

The version of Node.js that will be used to run your application on Heroku, should also be defined in your package.json file.

1. **Specifying a start script:**

To determine how to start the app, Heroku first looks for a Procfile. If no Procfile exists for a Node.js app, we will attempt to start a default web process via the start script in the package.json. The command in a web process type must bind to the port number

1. **Build your app and run it locally:**

Run the npm install command in your local app directory to install the dependencies that you declared in your package.json file. Start the app locally using the heroku local command, which is installed as part of the Heroku CLI.

1. **Deploy the application to Heroku:**

Execute following commands after committing changes to git:

$ git add .

$ git commit –m “Adding a procfile”

$ heroku login

Enter your login credentials

$ heroku create

$ git push heroku master

To open the app in your browser, type heroku open