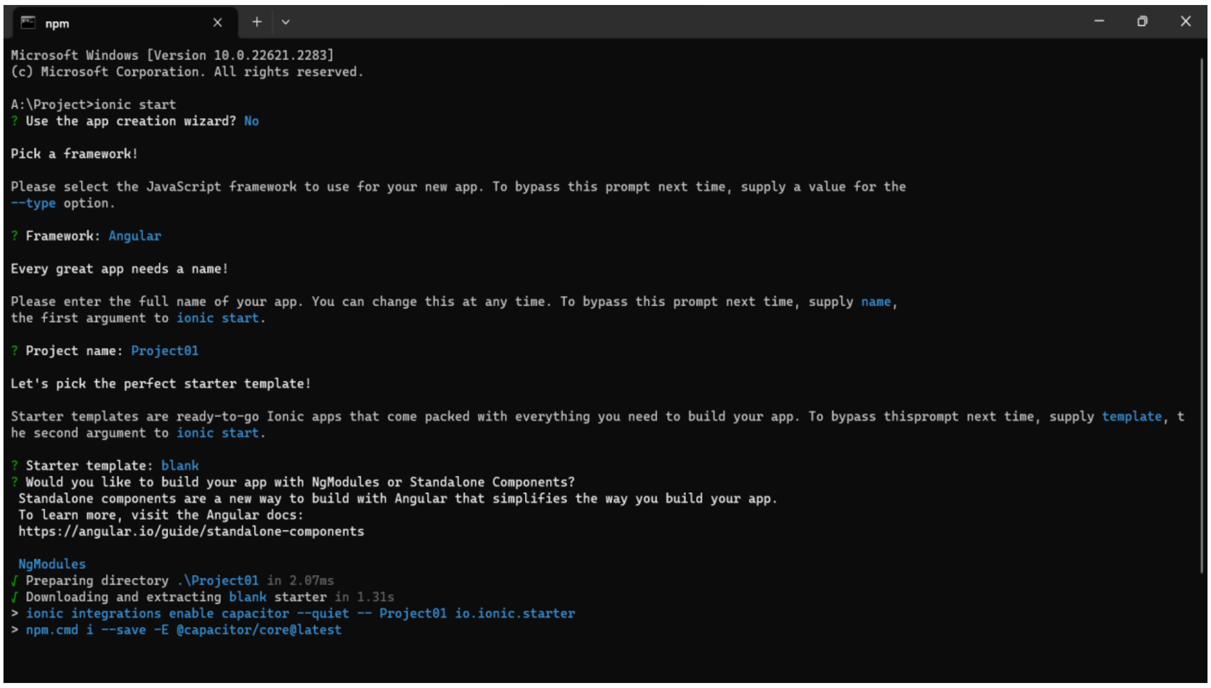
## Step 1: Set Up My Angular Application

1. Open a terminal/command prompt and navigate to Angular application directory.

2. Make sure your Angular application is ready for deployment. should have already built your application using ng build.



```
Microsoft Windows [Version 10.0.22621.2283]
(c) Microsoft Corporation. All rights reserved.

A:\Project>ionic start
? Use the app creation wizard? No

Pick a framework!

Please select the JavaScript framework to use for your new app. To bypass this prompt next time, supply a value for the
--type option.

? Framework: Angular

Every great app needs a name!

Please enter the full name of your app. You can change this at any time. To bypass this prompt next time, supply name,
the first argument to ionic start.

? Project name: Project01

Let's pick the perfect starter template!

Starter templates are ready-to-go Ionic apps that come packed with everything you need to build your app. To bypass thisprompt next time, supply template, t
he second argument to ionic start.

? Starter template: blank
? Would you like to build your app with NgModules or Standalone Components?
 Standalone components are a new way to build with Angular that simplifies the way you build your app.
 To learn more, visit the Angular docs:
 https://angular.io/guide/standalone-components

 NgModules
√ Preparing directory .\Project01 in 2.07ms
√ Downloading and extracting blank starter in 1.31s
> ionic integrations enable capacitor --quiet -- Project01 io.ionic.starter
> npm.cmd i --save -E @capacitor/core@latest
```

## Step 2: Initialize a Git Repository Locally

1. Inside Angular application directory, run the following command to initialize a local Git repository: **git init**

2. This command will initialize Git for My project.

## Step 3: Create a .gitignore File

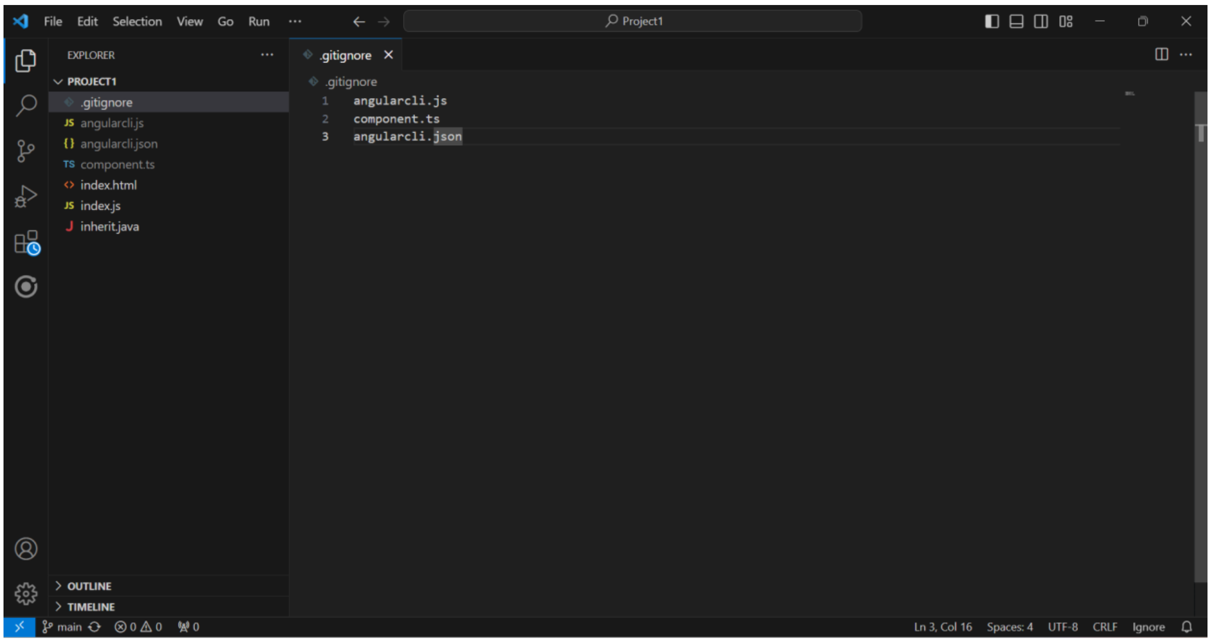
1. Create a .gitignore file in  project directory. This file will specify which files and directories should be ignored by Git during commits.

```
touch .gitignore
```

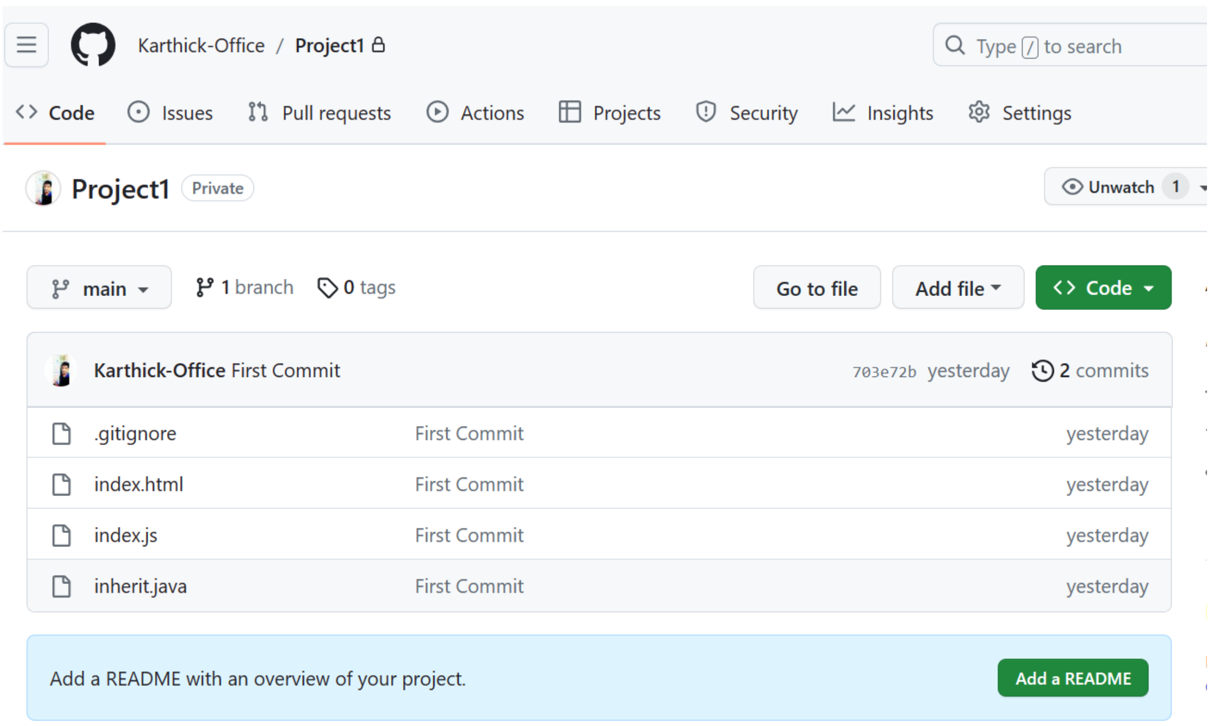
2. Open the .gitignore file using a nano text editor and specify files and directories that should be ignored.

```
# Ignore node_modules directory
node_modules/
# Ignore build output
dist/
# Ignore any environment-specific files
src/environments/*.ts
```

**In The Live Virtual Class, I Only Created A Few Files Based On The Instructions Given By The Instructor**



```
angularcli.js
component.ts
angularcli.json
```



Karthick-Office First Commit 703e72b yesterday 2 commits

| | | |
|---|---|---|
| .gitignore | First Commit | yesterday |
| index.html | First Commit | yesterday |
| index.js | First Commit | yesterday |
| inherit.java | First Commit | yesterday |

Add a README with an overview of your project.

## Step 4: Add Files to the Staging Area

1. Use the following command to add all files to the staging area:

   git add This stages all the changes we made.

## Step 5: Commit Changes

1. Commit the staged changes with a descriptive message:

   **git commit -m "Initial commit"**

## Step 6: Create a GitHub Repository

1. Go to GitHub and log in to My account.
2. Click the "+" icon in the upper right corner and select "New Repository."
3. Fill out the repository name, and other settings as needed.

   Repository name : Project1

   I'm create Private Repository
4. Click the "Create repository" button.

**Step 7: Link Local Repository to the GitHub Repository**

Step 1: Open a Terminal/Command Prompt

Step 2: Generate SSH Key Pair ssh-keygen

Run the ssh-keygen command to generate a new SSH key pair. By default, this command will generate an RSA key pair Open your SSH public key file in the `nano` text editor. In the terminal, run: nano ~/.ssh/id_rsa.pub This will open the `nano` text editor with the contents of your SSH public key.

Step 3: Copy the Public Key

Step 4: Add the Public Key to GitHub Account

i. Go to the GitHub website and log in to your GitHub account.
ii. Click on your profile picture in the top-right corner and select "Settings."
iii. In the left sidebar, click on "SSH and GPG keys."
iv. Click the "New SSH key" button.
v. Give your SSH key a title
vi. After pasting the key, double-check to ensure it's correct and complete.
vii. Click the "Add SSH key" button.

SSH key is now added to GitHub account settings, to securely authenticate when interacting with GitHub repositories using SSH. This provides a more secure and convenient way to connect to GitHub compared to using a username and password.

```
git remote add origin git@github.com:Karthick-Office/Project1.git
git branch -M main
git push origin main
```

**Step 8: Push Code to GitHub**

1. Finally, push My code to the GitHub repository using:

   ```
   git push origin main
   ```

2. You'll be prompted to enter your GitHub credentials.

**Step 9: Verify on GitHub**

1. Visit your GitHub repository in a web browser to ensure that code has been pushed successfully.

**During Live Class**

**GitHub repository Link:** https://github.com/Karthick-Office/Project1.git

**My Work:** Using Ionic Framework Angular Project

**Public GitHub repository Link:** https://github.com/Karthick-Office/Project01.git