**Prerequisites:**

* An Azure account with an active subscription <https://azure.microsoft.com/en-us/pricing/>
* Node.js and npm (or yarn) installed on your development machine <https://nodejs.org/en/learn/getting-started/how-to-install-nodejs>
* An Angular 17 project (if you don't have one, create a new one using the Angular CLI: ng new my-angular-app)

**Steps:**

1. **Build your Angular application for production:**
   * Open your terminal or command prompt and navigate to your Angular project's root directory.
   * Run the following command to build the app in production mode:

Bash

ng build --prod

Use code [with caution.](https://gemini.google.com/faq#coding)

content\_copy

1. This will create an optimized production-ready build in the dist folder within your project directory.
2. **Create an Azure App Service:**
   * Sign in to the Azure portal (<https://azure.microsoft.com/en-us/get-started/azure-portal>).
   * Click on "Create a resource" and search for "App Service."
   * In the "Create App Service" blade, provide the following details:
     + **Subscription:** Select your Azure subscription.
     + **Resource group:** Create a new resource group or choose an existing one. Resource groups help you manage related Azure resources.
     + **App name:** Choose a unique name for your app service. This will be part of the URL where your app is accessible.
     + **Publish:** Select "Code" as the publishing option.
     + **Runtime stack:** Choose a runtime stack that supports Node.js. A common choice is "Node.js 16 LTS" or a later compatible version.
     + **Region:** Select a region closest to your target audience for optimal performance.
     + **Pricing tier:** Choose a pricing tier that suits your application's needs. Free and shared tiers are available for testing purposes.
   * Once you've filled in these details, click "Review + create" and then "Create" to provision the App Service.
3. **Configure Deployment Credentials (Optional):**
   * If you plan to deploy using manual FTP or Git deployment, you'll need to configure deployment credentials.
     + In the Azure portal, navigate to your App Service.
     + Go to "Deployment Center" in the left menu.
     + Choose the deployment method you'll use (e.g., FTP, Local Git) and configure the credentials accordingly.
4. **Deploy your Angular application:**
   * You have several deployment options:
     + **Manual Deployment (FTP or Local Git):**
       - Use an FTP client or the Azure portal's "Deployment Center" to upload the contents of your Angular project's dist folder to the wwwroot directory of your App Service.
       - **Important:** If your Angular application doesn't use the default index.html file, you'll need to configure path mappings in your App Service settings. Navigate to "Configuration" -> "Path mappings" and set the appropriate virtual path and physical path.
     + **Continuous Deployment from GitHub or Azure DevOps:**
       - Set up a continuous integration/continuous delivery (CI/CD) pipeline in GitHub Actions or Azure DevOps to automate building and deploying your Angular application whenever you push code changes to your version control repository. Search for specific instructions for your chosen platform.
5. **Access your deployed Angular application:**
   * Once your deployment is complete, navigate to the URL of your App Service (e.g., https://<your-app-name>.azurewebsites.net). Your Angular application should be running!

**Additional Considerations:**

* **Environment Variables:** If your Angular application relies on environment variables (e.g., API endpoints), you can configure them in your App Service settings under "Configuration" -> "App settings."
* **SSL/TLS Certificates:** If you require secure communication (HTTPS), you can add an SSL/TLS certificate to your App Service.

By following these steps, you'll successfully deploy your Angular 17 application to Azure App Service and make it accessible to the worl