# Workshop: Part I

Workshop for the "Angular" course @ SoftUni

In this workshop we will start developing a fully functional Post Blog application using Angular. The application will use Firebase and MongoDB to perform CRUD operations. It will also include user authentication, using REST API endpoints, and access control for public and private sections of the app.

## Setting Up the Project

### Step 1: Install Angular CLI

If you still haven't done it, install Angular globally using this command:

|  |
| --- |
| **npm install -g @angular/cli** |

### Step 2: Create a New Project

Now, let's generate a new project using the command below:

|  |
| --- |
| **ng new post-blog cd post-blog** |

### Step 3: Start the Development Server

Now, let's start the Angular development server and open the project in your browser:

|  |
| --- |
| **ng serve --open** |

This command starts the development server and opens the project in your default web browser. The **--open** flag automatically launches the browser with the project URL.

## Setting Up Firebase for Database Integration

### Step 1: Register and Create a Firebase Project

Firebase provides a scalable, real-time database service that makes backend integration seamless for front-end developers.

We will now use it to set up a cloud database to store blog post data, enabling real-time updates and scalability without manual server management.

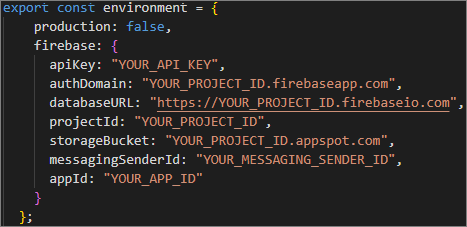
* Visit [Firebase Console](https://console.firebase.google.com/).
* Create a new project.
* Navigate to **Build → Realtime Database** and set up a new database.
* Create a collection named **Posts** with fields like **title** and **content**.

### Step 2: Integrate Firebase with our Project

We will now integrate Firebase with our project so that we are able to make the application communicate with the database, so that we can execute CRUD operations.

Use the command below to do that:

|  |
| --- |
| **npm install firebase @angular/fire** |

Usually, there is a file **src/environments/environment.ts**, which contains configuration details for connecting the application to Firebase project. We need to set it up like in the picture below:  


Replace the placeholders with the actual values from the Firebase project.

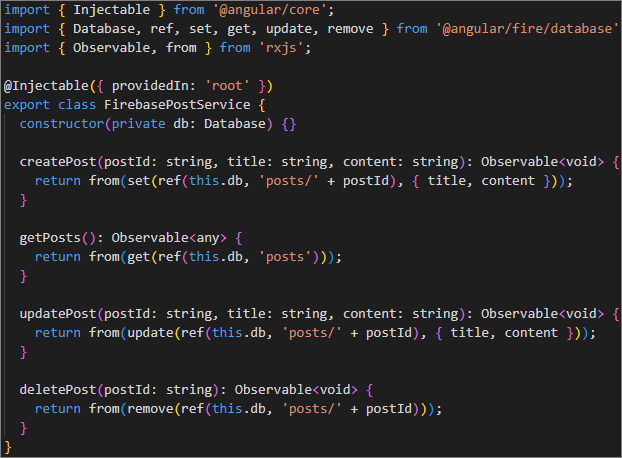
**NOTE: If there is no such file or directory in your project, you have to create it.**

Now, after we are done with the **environment.ts** file we have to initialize the file **src/app/app.module.ts**, in order to allow access to Firebase's database functions:

## Create Firebase Services for CRUD Operations

Now, let's create a service, which provides methods to create, retrieve, update, and delete posts in Firebase’s Realtime Database, handling all CRUD operations directly from Angular.

Create a file called **firebase-post.service.ts** in the **src/app/services** directory:



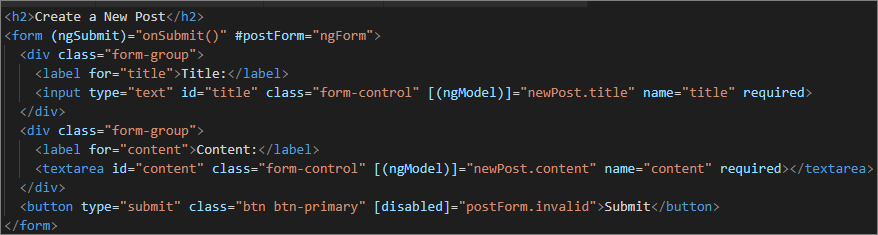
## Create Components and Integrate Firebase in Angular

### Step 1: Create a Component for Creating New Posts

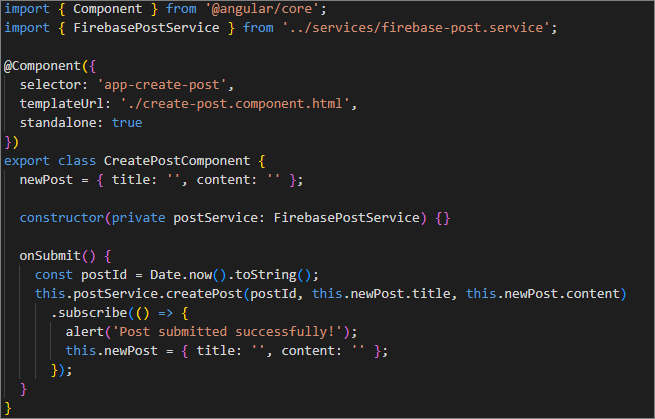
First, generate a new component using the command below:

|  |
| --- |
| **ng generate component create-post --standalone** |

Then, set up the form in the **create-post.component.html** file like shown in the image below:



Now, we have to add the logic for displaying the posts to use the Firebase service:

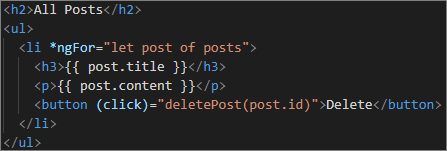


### Step 2: Create a Component for Displaying Posts

Next, let's generate a **post-list** component by using the command below:

|  |
| --- |
| **ng generate component post-list --standalone** |

As we did with the previous component, set up the **post-list.component.html** to display the posts:

****

Don't forget to add the logic in the **post-list.component.ts**:

