If you don't already have an Angular project, you can create one by running the following commands:

bash

Copy code

ng new angular-form-app

cd angular-form-app

ng serve

In this example, I'll create a login form with username and password fields using Angular's template-driven form approach.

Make sure you import FormsModule from @angular/forms to use template-driven forms in Angular.

app.module.ts

import { NgModule } from '@angular/core';

import { BrowserModule } from '@angular/platform-browser';

import { FormsModule } from '@angular/forms'; // Import FormsModule

import { AppComponent } from './app.component';

@NgModule({

declarations: [

AppComponent

],

imports: [

BrowserModule,

FormsModule // Add FormsModule here

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

Now, let's create the form in the HTML template. We'll have two fields: one for the username and one for the password.

App.component.html

html

Copy code

<div style="text-align:center;">

<h1>Login</h1>

<form #loginForm="ngForm" (ngSubmit)="onSubmit(loginForm)">

<div>

<label for="username">Username:</label>

<input

type="text"

id="username"

name="username"

[(ngModel)]="model.username"

#username="ngModel"

required

/>

<div \*ngIf="username.touched && username.invalid">

<small \*ngIf="username.errors?.['required']">Username is required.</small>

</div>

</div>

<div>

<label for="password">Password:</label>

<input

type="password"

id="password"

name="password"

[(ngModel)]="model.password"

#password="ngModel"

required

/>

<div \*ngIf="password.touched && password.invalid">

<small \*ngIf="password.errors?.['required']">Password is required.</small>

</div>

</div>

<button type="submit" [disabled]="loginForm.invalid">Submit</button>

</form>

<div \*ngIf="submitted">

<h2>Form Submitted</h2>

<p>Username: {{ model.username }}</p>

<p>Password: {{ model.password }}</p>

</div>

</div>

**3. Create the component logic**

The component will handle the form submission and store the values from the form. We can also track whether the form was submitted or not.

app.component.ts

typescript

Copy code

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

model = {

username: '',

password: ''

};

submitted = false;

onSubmit(form: any) {

this.submitted = true;

console.log('Form Data: ', form.value); // Log the form data

}

}

You can style the form with basic CSS to make it look nicer. You can add styles in app.component.css:

css

Copy code

form {

max-width: 400px;

margin: 0 auto;

padding: 20px;

border: 1px solid #ccc;

border-radius: 8px;

}

input {

width: 100%;

padding: 8px;

margin: 8px 0;

border: 1px solid #ccc;

border-radius: 4px;

}

button {

width: 100%;

padding: 10px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 4px;

cursor: pointer;

}

button:disabled {

background-color: #ccc;

}

small {

color: red;

}

After setting everything up, you can run your Angular application using the following command:

bash

Copy code

ng serve

Visit http://localhost:4200 in your browser to see the form.

**Explanation:**

1. **ngModel**: This directive is used for two-way data binding. It binds the form fields (username and password) to the model object in the component.
2. **#loginForm="ngForm"**: This creates a reference to the form, which allows you to access its state and validate it (e.g., checking if the form is valid).
3. **Form Validation**: The form has basic validation, ensuring that both fields are required. If a user tries to submit the form without filling out the fields, they are shown an error message.
4. **ngSubmit**: The onSubmit() method is triggered when the form is submitted.

**Conclusion:**

This is a simple example of how to create a template-driven form in Angular for login with a username and password. You can expand this by adding more fields, validations, or integrating it with a backend API for user authentication.