Name :- Dhola Drashti Ishwarbhai Date :- 17/10/23
Roll no :- 12
Sem :- 7th [MSc IT]

Assignment :- 3

☆ Write in details along with diagram, code and examples wherever applicable :

- Angular Introduction

- Angular is a popular open-source web application framework for building dynamic, single-page applications and enterprise-grade web applications.

- It is developed and maintained by Google and a community of individual developers and organizations.

- Angular is written in TypeScript, which is a superset of Javascript, and provides a comprehensive set of tools and features for developing complex and feature-rich web applications.

Key Features of Angular

- Angular offers a wide range of features and benefits that make it a powerful framework for web development.

1. Declarative UI :

Angular allows developers to build the user interface using HTML templates and declarative data bindings.

VISION

This simplifies the process of creating dynamic and responsive user interfaces.

2. Component - Based Architecture

- Angular applications are built using a components - based architecture.

- Components are self - contained, reusable building blocks that encapsulate HTML templates, styles, and behavior.

3. Dependency Injection :

Angular's built in dependency injection system makes it easy to manage and share application - wide services and data between different parts of the application.

4. Two- way Data Binding

Angular supports two - way data binding, allowing automatic synchronization of Data between the model and the view.

This simplifies the process of keeping the UI in sync with the underlying data.

5. Routing

Angular provides a powerful routing system that enables the creation of multi-page applications with support for deep linking, lazy loading and route guards.

6. Forms

Angular offers robust support for building complex forms, including form validations, form controls, and two-way data binding with form elements.

7. HTTP client

Angular includes an HTTP client module for making HTTP requests to fetch data from servers and interacts with APIs.

8. Testing

Angular has built in support for unit testing and end-to-end testing, making it easier to ensure the reliability and quality of your applications.

- Angular application structure.

- Angular applications follow a well-defined structur that helps organize your code and resources efficiently.

- This structure ensures a clear separation of concerns, making it easier to maintain, test and scale your application.

```
angular-app/
    ├ src/
        ├ app/
            ├ components/
                ├ my-component/
                    ├ my-component.component.ts
                    ├ my-component.component.htm
                    ├ my-component.component.css
                    ├ my-component.component.spec.ts
            ├ services/
            ├ modules/
        ├ assets/
        ├ environments/
        ├ index.html/
        ├ main.ts
    ├ angular.json
    ├ package.json
    ├ tsconfig.json
    ├ tsconfig.app.json
    ├ tslint.json
    ├ README.md
```

1. 'src' Directory

- The src directory is the heart of Angular application.
- it contains all the source code and resources required for application.

2. 'app' Directory

- The app directory is where will spend most of time.
- it contains the core components, services, and modules of application.

- Component directory

- The component directory is where you define your angular components.
- Each component is organized within its own subdirectory.
- For example, if you have component named 'my-component", you will have the following files:

my-component.component.ts : Typescript file that defines component class.

my-component.component.html : HTML template associated.

my-component.component.css : css / scss file for styling component

my-component.component.spec.ts : unit test for component

'Services' directory

- The services directory is where you define feature Angular services.
- Services are used for handling data, making HTTP requests, and sharing functionality across components.

'modules' directory

- The module directory is where you define feature modules.
Feature modules help you organize and encapsulate related components, services, and routes.
Each feature module has its own directory

'assets' directory

The assets directory is used to store static files, such as images, fonts, or configuration files.

'environment' Directory

- The environment directory is containing environment specific configuration files.
- By default Angular provides two files: environment.ts for development and environment.prod.ts for application production.

8) index.html and main.ts

index.html

- The main HTML file for your application
- it serves as the entry point for your Angular application.
- You can include metadata, css files, and the Angular application root component selector

main.ts

- The file is the entry point of your application's TypeScript code.
- it bootstraps the angular application and starts the execution.

4) 'configuration' files.

angular.json : This file contains configuration settings for your Angular project, including build options, assets, and dependencies.

package.json : This is your project's package.json file, which lists your projects dependencies and scripts

tsconfig.json : The TypeScript configuration file for your entire project.

ts config.app.json : and tsconfig.spec.json : specific TypeScript configurations for the app and tests, respectively.

tslint.json : Configuration for TSLint, a code analysis tool for typeScript

5) README.md

The readme.md file is used to provide documentation and instructions for your project.

drushti dholu-12

- Angular Architecture.

Angular architecture follows the MVC pattern, but in Angular terminology, it's refferd to as Model - View - View Model

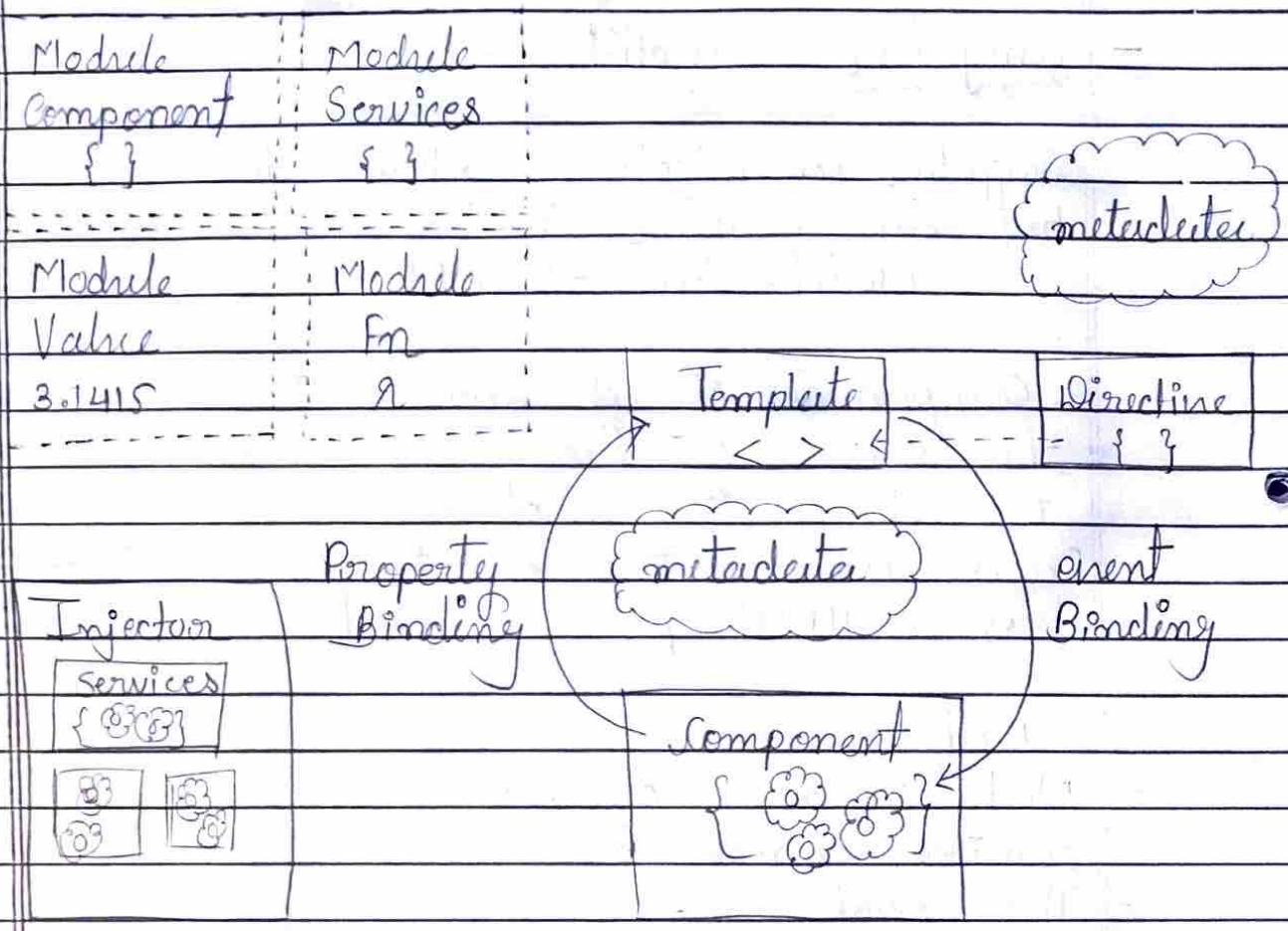
1. Components : A component encapsulates the user interface, its behavior, and the data it displays.
- Each component consist of a TypeScript class, HTML page, and associated styles.

2. Modules :
- Modules are containers for organizing related components, services and other code.
- The root module is 'App Module'.

3. Services :
- services are used for encapsulating business logic, data access and communication b/w components.

4. Directives :
- directives are used to manipulate DOM.
- Angular provides built-in services like ngIf and ngFor

vision



Component :

class : contains properties and methods to define the component's behaviour

Template : Angular specific syntax to bind data and control the view

Styles : css / scss styles for HTML template

Metadata : used to decorate the class and define the component's characteristics

```
import { component } from '@angular/core';

@component ({
    selector : 'app-greeting',
    template : '<h1> {{ message }} </h1>',
    styles : ['h1 {color : blue; }']
})

export class GreetingComponent {
    message = 'Hello , Angular';
}
```

Module example

```
import {NgModule } from '@angular/core';
import {BrowserModule} from '@angular/platform-
                                    browser';
import {GreetingComponent} from './greeting.component';

@NgModule ({
    declarations : [GreetingComponent],
    imports : [BrowserModule],
    bootstrap : [GreetingComponent]
})

export class AppModule { }
```

Services example

```
import { Injectable } from '@angular/core';

@Injectable()
export class DataServices {
    private data: string[] =
        ['Item 1', 'Item 2', 'Item 3'];

getData() {
    return this.data;
}

addData(newItem: string) {
    this.data.push(newItem);
}
}
```

Directives example

```
<ul>
  <li *ngFor = "let item of items">
    {{ item }}
  </li>
</ul>
```

Template example

```
<div>
    <h1> {{ title }} </h1>
    <p> {{ discription }} </p>
</div>
```

Metadata example

```
import {component} from '@angular/core';

@Component({
    selector: 'app-greeting',
    template: '<h1> {{ message }} </h1>',
    styles: ['h1 {color: blue;}']
})

export class GreetingComponent {
    message = 'Hello, Angular';
}
```

TypeScript introduction

- Typescript is an open-source, statically typed superset of javascript that compiles to plain JS.
- It was developed by Microsoft and is designed for building large-scale, maintainable, and robust web applications.
- Typescript add static typing, interfaces, classes and other features to JS, making it a powerful tool for developers.

Key features of Typescript

static typing, Interfaces, classes, Type Annotation, Modules, Enums, Type Interface, compatibility

Typescript	V/s	JavaScript
statically typed		dynamically typed
supports OOP.		doesn't support
must be transpiled to JS before execution		executed directly

Examples

Type Annotation

```
let name: string = 'John';
let age: number = 50;
let isStudent: boolean = false;

function greet(person: string): string {
    return `Hello, ${person}!`;
}

console.log(greet(name));
```

Interfaces

```
interface Person {
    name: string;
    age: number;

function introduce(person: Person): string {
    return `Hi, I'm ${person.name}`;
}

const user: Person = {name: 'Alice', age: 25};
console.log(introduce(user));
```

Classes

```
class Animal {
    constructor (public name : string ) {}

    makeSound (sound : string ) : void {
        console.log (`${this.name} makes a ${sound}`);
    }
}

Const cat = new Animal ('cat');
cat.makeSound ('meow');
```

Typescript Workflow

npm install -g typescript
write code
compile → tsc file.ts
Run.

- Angular CLI Commands

- The angular CLI is a powerful tool that simplifies the development, testing and deployment of angular applications.

Installation

npm install -g @ angular/cli

New project creation

ng new my-app

Change the project directory

cd my-app

Development Commands

Serve : starts development server and serves

ng serve --open

Generate : used to create various components

ng generate component my-component

Testing commands

Unit tests

ng test

End-to-End Tests

ng e2e

Building & deployment

ng build

Deploy

The deployment process typically involves copying the contents of the /dist directory generated by the 'ng build'.

Code linting & formatting

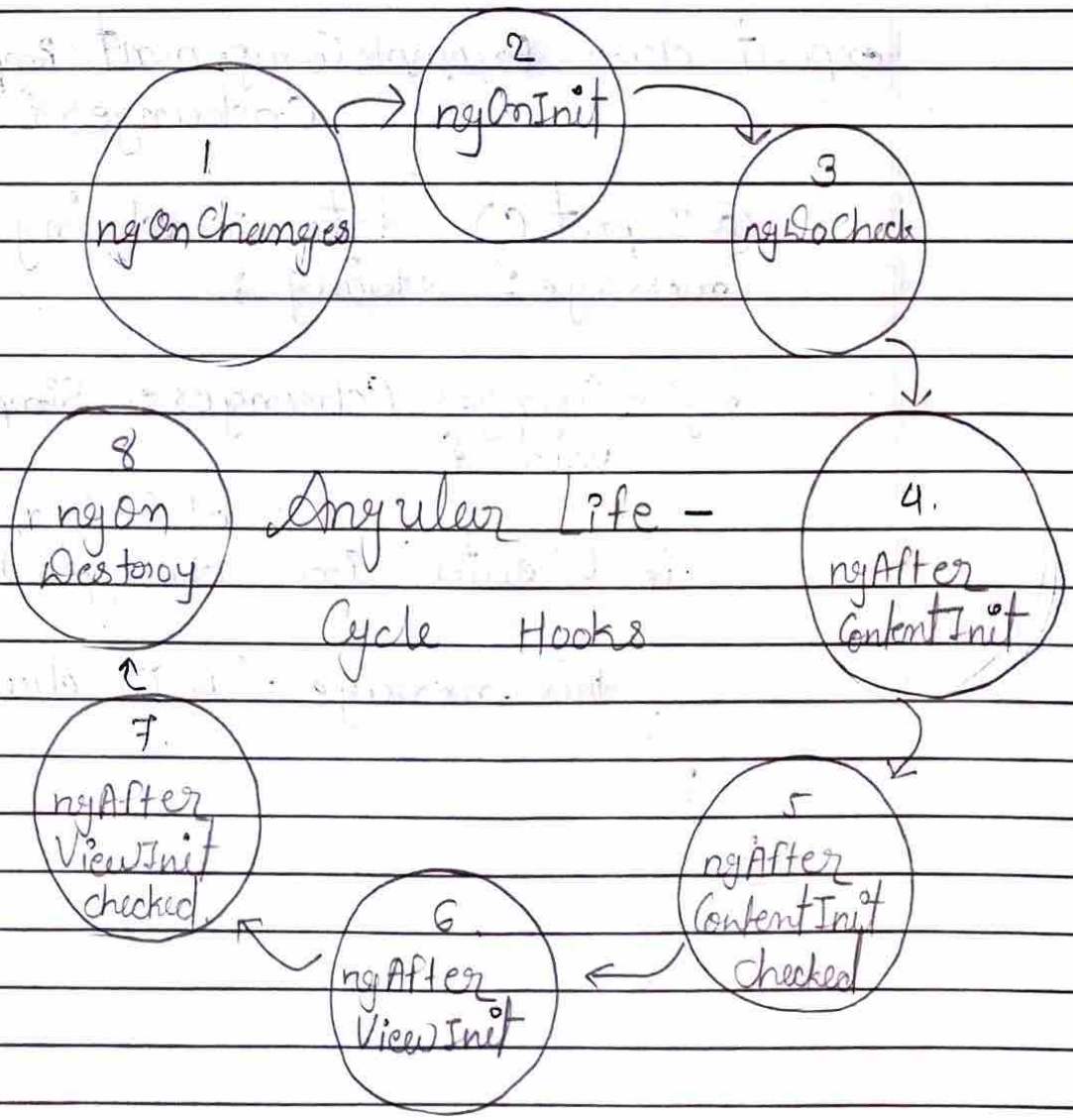
ng lint

ng format

- Component LifeCycle Hooks.

- components have a well-defined lifecycle with various phases during their existence

- angular provides a set of lifecycle hooks that allow you to tap into these phases and execute code at specific points in a component's lifecycle.



1. ngOnChanges

```
import {component, Input, OnChange, SimpleChange}
    from '@angular/core';

@component ({
    selector : 'app-example',
    template : '<p>{{ message }}</p>',
})

export class ExampleComponent implements
                    OnChanges {

    @Input() date1 : string ;
    message : string ;

    ngOnChanges (changes : SimpleChanges) :
        void {

        if ('date1' in changes) {

            this.message = 'Date1 changed' ;
        }
    }
}
```

2. ngOnInit

```
import { component , OnInit } from '@angular/core';

@ component ({
    selector : ' app - example',
    template : '<p> {{ message }} </p>'
})

export class ExampleComponent implements OnInit {

    message : string;

    ngOnInit () : void {
        this . message = 'compond initialized';
    }
}
```

3. ngAfterViewInit

```
export class ExampleComponent implements
                                AfterViewInit {

    message : string;

    ngAfterViewInit () : void {
        const messageElement =
            this . messageElement . nativeElement;
        messageElement . style . color = 'blue';
    }
}
```

4. ngOnDestroy

```
export class ExComponent implements OnDestroy{
    message: string;
    private subscription : Subscription ;

    constructor () {
        this. subscription =
    }

    ngOnDestroy (): void {
        this. subscription . unsubscribe ();
    }
}
```

★ Write details of following topics with example commands, Flow diagram, code fragements and at the end make application combining them.

- components , Templates

Components

- components are the building blocks of an Angular application.
- Each component encapsulates a part of the application's logic and user interface.
- Components are organized in a component tree, with one root component representing the entire application.

Creating a Component

ng generate component my-component

This command generates the component files in the project directory, including a TypeScript file, an HTML template, and CSS styles.

Component Structure

```
import {component} from '@angular/core';

@component ({
    selector : 'app-my-component',
    templateUrl : './my-component.component.html',
    styleUrls: ['./my-component.component.css']
})

export class MyComponentComponent {
    //
}
```

Templates

- Templates define the user interface of a component.
- Angular template use HTML enhanced with Angular-specific syntax and data binding expressions.

Building a Template

```
<div>
    <h1> {{ title }} </h1>
    <p> {{ description }} </p>
    <button (click)="onButtonClick()"> Click </button>
</div>
```

In above example

Interpolation (`{{ title }} ` and `{{ description }}`) to display component properties.

Event Binding (`(click) = "onButtonClick()"`) to bind to a button click event.

- Routing
- allows you to navigate b/w different views or components within your application.
- it helps create single-page application by loading content dynamically without full page reloads.

Setting Up Routing

Generate a Routing Module

ng generate module app-routing --flat --module=app

configure Routes

app-routing.module.ts

```
const routes: Routes = [
  { path: 'home', component: Home },
  { path: 'products', component: Product },
  { path: '', redirectTo: '/home',
    pathMatch: 'full' },
];
```

Use Router Outlet

```
<h1> welcome </h1>
<router-outlet> </router-outlet>
```

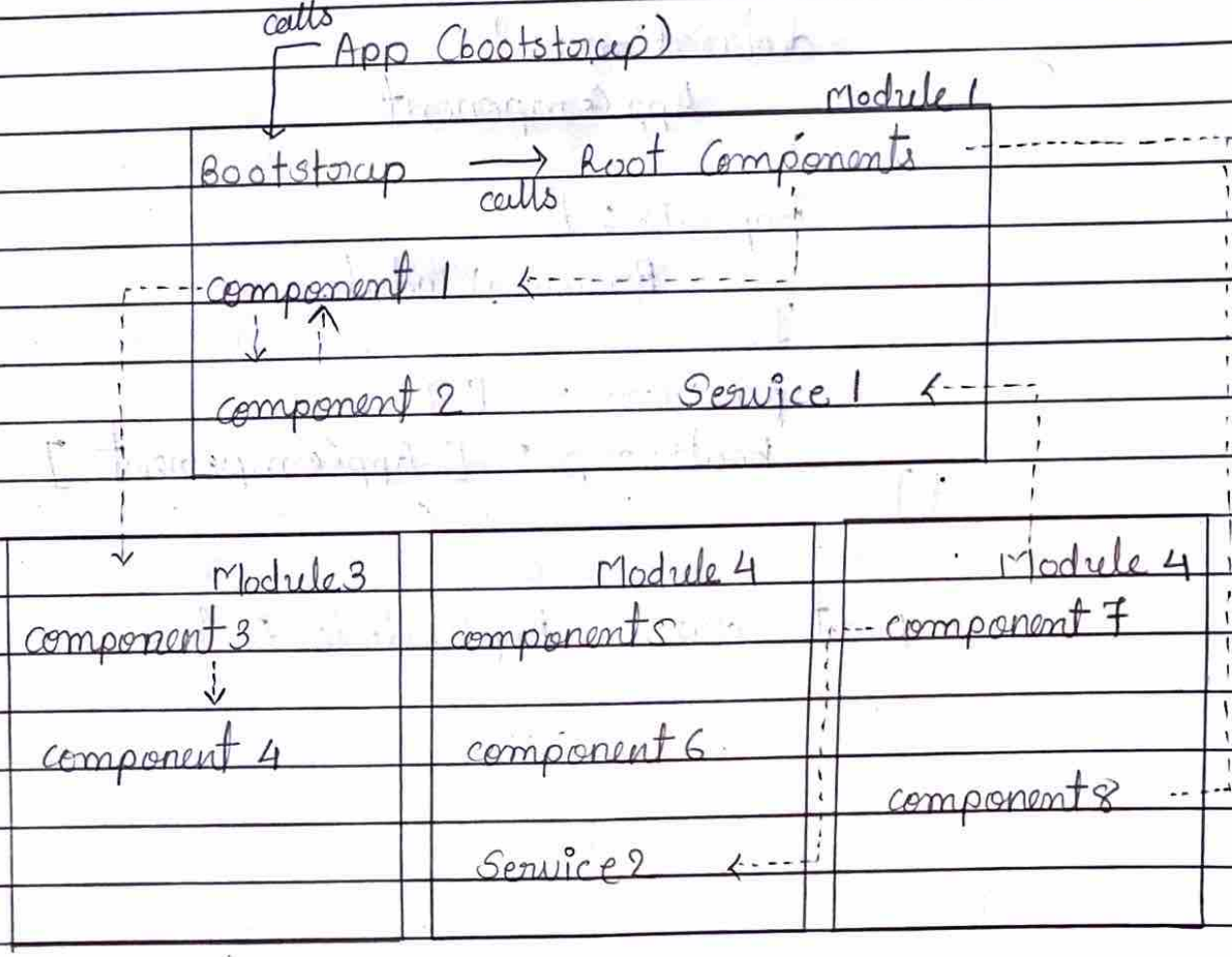
Navigation

```
<a routerLink = "/products"> go </a>
```

drashti dhoke-12

Modules

- Angular Module is basically a collection of related features / functionality.
- Angular Module groups multiple components and services under a single context.



```
import {BrowserModule } from '@angular/platform-
                              browser';
import { NgModule } from '@angular/core';
import { AppComponent } from './app.component';

@NgModule ({

    declarations: [
        AppComponent
    ],
    imports: [
        BrowserModule
    ],
    providers: [],
    bootstrap: [ AppComponent ]
})

export class AppModule { }
```

Pipes

- Pipes are referred as filters. It helps to transform data and manage data within interpolation, denoted by {{ | }}.
- It accepts data array, integers and strings as input which are separated by '|' symbol.

example

```
export class TestComponent {
    presentDate = new Date();
}
```

```
<div>
    Today's date :- {{ presentDate }}
</div>
```

adding pipe parameter

```
<div>
    Today's date :- {{ presentDate | date }}
</div>
```

Directive

- Angular directives begin with ng- where ng stands for: Angular and extends HTML attribute with @ directive decorator.

- directives enables logic to be included in the Angular templates.

- Three categories

attribute directive

used to add new attributes for the existing HTML elements to change its look and behavior

<HTMLTag [attrDirective] = 'value' />

structural Directive

<HTMLTag [structural Directive] = 'value' />

Component - based Directive

<Component - selector - name [input - reference] = "input - value" > ... </ component - sel. >

Forms

- Forms are used to handle user input data
- Two types of forms :-

Template driven forms

- it is created using directives in the template.
- mainly used for creating a simple form application.

Reactive forms

- Reactive forms are created inside a component class so it is also referred as model driven forms.
- Every control will have an object in the component and this provides greater control and flexibility in the form programming.

HttpClient

- powerful and convenient way to make HTTP requests in your Angular Application.
- simplifies the process of sending and receiving data from a server, whether it's fetching data from an API, posting data, or handling other HTTP request

Installation

npm install @angular/common/http

importing

import {HttpClientModule} from @angular/common/http

Use

constructor (private http: HttpClient) {}

- Combine all above code fragements in examples, it will make an Angular app.