

Assignment No 5.

Explain date & time filter in angular with example?

In angular you can implement date & time filtering using pipes.

Make sure you have imported date pipe from @angular/common in your components.

```
import { DatePipe } from '@angular/common';  
import { Component } from '@angular/core';  
import { DatePipe } from '@angular/common';
```

```
@Component({  
  selector: 'app-date-time-example',  
  template: '/date-time-example.component.html',  
  styleUrls: ['./date-time-example.component.css']  
})
```

```
export class DateTimeExampleComponent {  
  currentDate: Date = new Date();  
  currentDate: Date = new Date("2024-03-2  
  TTTT: 00:00");  
}
```

```
constructor(private datePipe: DatePipe) {}
```

```
formatCurrentDate(): string {  
  return this.datePipe.transform(this.currentDate,  
    'yyyy-MM-dd HH:mm:ss');  
}
```

html :-

<P> Current Date: {{ formatCurrentDate() }} </P>

<P> Current Date: {{ formatCurrentDate() }} </P>

=

Q 2) What is routing in angular?

→ Routing:-

Routing in angular allows you to navigate to between different component & views in your single page application (SPA) without reloading the entire page.

1) Setting up Router:-

In angular project define your routers in the app-routing-module.js file as in separate routing module.

2) Creating Components:-

Create the components referenced in the routers for example create 'home-component.js' about component & continuous component.

3) Navigation:-

Use angular directive such as router-link to navigate between routes in your HTML template. The router-outlet directive will dynamically load the component associated with current source.

4) Router Parameters:-

You can define with parameter to handle dynamic data.

Q3) Explain dependency injection in angular with example?

→ Dependency injection (DI) :-

Dependency injection (DI) in Angular is a design pattern used to provide & manage dependencies within your application. It helps in creating looser coupled components & facilitates testing, reusability & maintainability.

1) Generating service :-

First a service that you want to inject into your components. Services are typically used for sharing data or functionality across multiple components.

2) Injection the service :-

Next inject the service into your components constructors. Angulars system will take care of providing an instance of the service when components is created.

3) Using the service :

How you can use the service method or properties within your components.

4) Providing service in module or components :

providing service in module or components as you can also provide services of the module or components level.

5) Injecting other Dependencies:

There apart from services you can also.

inject other dependencies like other service, HTTP client, router etc.

Q4) Explain SPA in Angular :-

→ SPA (Single page application) :-

Angular is well-suited for building SPA's due to its powerful features like routing, data binding & dependency injection.

1) Routing setup:-

Angular's router allow you to define different router for your application. You can figure routing in the app-routing-module's file or in separate routing module.

2) Component structure:-

Organize your application into components each component represents a part of your application. UI Components can be merged to create complex UI structure.

3) Template Binding:-

Use Angular data binding system syntax bind data from your components class to its template. This allow you to dynamically update the UI or build on changes in your application data.

4) Service for Business logic:-

Use Angular service to encapsulate business logic, data access & other shared functionality. Service can be injected into components.

making them reusable & trustable.

5) HTTP Requests:-

Use Angular's Http client module to make http request to a server for testing data or interacting with APIs.

6) Testing:-

Write unit tests & end-to-end tests to ensure the reliability & correctness of your applications.

Q5.) Explain Typescript or angular with example?

→

```
import { Component } from '@angular/core';
import { HttpClient } from '@angular/CommonComponent / http';
import { Observable } from 'rxjs';
```

```
@Component({
```

```
  selector: 'app-root';
```

```
  templateUrl: './app.component.html';
```

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

```
})
```

```
export class AppComponent {
```

```
  users: Observable<any>;
```

```
  constructor (private http: HttpClient) {
```

```
    this.users = this.http.get('http://jsonplaceholder  
holder.typicode.com/users');
```

We import necessary module like 'Component', 'httpClient' & 'observable' from Angular. core modules.

We define Typescript class App. Component ~~for~~ represent a component in Angular within app Component class we declare a variable 'users' of type 'observable <any>' to hold data retrieved from API Synchronously.

Inside the constructor we make HTTP GET request using 'httpClient' service to fetch the data.

