Ce se intampla in spate cand Angular porneste aplicatia?

The JavaScript file references get injected into our index.html page by a utility that Angular uses to build her application. The utility is named Web Pack.

**Components** ofera datele pentru view in browser.

Angular foloseste componente. In componenta avem un decorator, TypeScript suporta decorators (this is a way of giving a normal class some extra powers). De exemplu giving a class the ability to be an angular component.

**Interpolation** – folosim proprietati din componenta in template file( file.html ). We can get data from our component into our template.

**Module** – declara componentele care sunt ‘available’ in aplicatie intr-un array de declaratie. Putem importa alte module angular in alte module folosindu-le in aplicatia noastra.

Implements OnInit- Pentru a folosi diferite functii dupa constructor.

Pentru a utiliza **cod asincron** Angular utilizeaza **Observables**.(Not a feature of JavaScript but they are available in typescrip).\

**Observables** are lazy. Nu fac nimic daca cineva nu ii da subscribe.

-they can be streams of data, but not in the case of an HTTP request going to an API.

**--End of Chapter 3--**

To display data in html we use **structural directive**. They are something that modifies the domain object model in our HTML in some way.

Angular manipulates the DOM(domain object model), jQuery also manipulates the DOM. But angular add its own change detection to the DOM. Daca folosim si angular si jQuery putem sa avem probleme in cod.

In this application DatingApp we are using SSL security to encrypt the data on the way from the client to the server. We are using a certificate.

JSON Web Token – They are self-contained and they can contain credentials, claims and other. We don’t need to call to the database every time for authentication because API server is going to verify that it’s a valid token based on the signature that is signed it with.

Un string foarte lung. Care contine un algoritm care cripteaza o semnatura in prima parte din token. In partea a 2-a din token avem payloud-ul. Iar in a 3-a parte verificarea semnaturi. Semnatura tokenului este criptata de server utilizand o cheie securizata care niciodata nu paraseste serverul.

Benefits – No session to manage, small and lightweight.

* No cookies required – mobile friendly
* Portable – a single token can be used with multiple backends.
* Performance – once the token is issued, there is no need to make a database request to verify a user authentication.

**--End of Chapter 4--**

Create a new component – ng g c nameofcomponent - -skip-tests.

Create a service – ng g s nameofservice - -skip-tests

**Two way binding**- takes data entered in a form and update something in our component.

**Services** are injectable, singleton. The data stored in the service doesn’t get destroyed until our application is closed down. Usually used for **http request.**

**Components** are destroyed as soon as they’re not in use.

**What are Observables?**

New standard for managing async data.

They are lazy collection of multiple values over time.

**--End of Chapter 5--**

Handle Exception – We create a API middleware class. With a RequestDelegate (what’s coming up next in the middleware pipeline).

-We use angular Interceptors. Check the responses for every response that comes back from our API.

What is a middleware?

When we have middleware we have access to the http request that’s coming in.

Format text shift+alt+f

De ce folosim virtual?

In Entity Framework, the virtual keyword is used to enable lazy loading of related entities.

By marking a navigation property as virtual, you tell Entity Framework to create a proxy object for the related entity. This proxy object can be used to load the related entity data only when it is accessed for the first time, and not when the parent entity is loaded from the database. This can improve the performance of your application by reducing the amount of data that needs to be loaded from the database.

In the example I provided for the Appointment class, the User and Doctor properties are marked as virtual because they are navigation properties that represent a relationship between the Appointment entity and the ApplicationUser and Doctor entities. By marking them as virtual, we allow Entity Framework to lazy load these related entities when they are accessed.

It's important to note that lazy loading can have some performance implications, especially if you are loading a large number of related entities. You should carefully consider the performance implications of lazy loading and choose the appropriate loading strategy for your application.