Web applicatie workflow

De behandelde web technieken spelen ieder een rol in het creëren van een web applicatie. De typische flow van een web applicatie is als volgt:

1. Gebruiker opent in de browser de applicatie
2. De browser stuurt een request naar de server
3. Aan de server kant wordt de request afgehandeld (bijv. een C# controller)
4. In de controller wordt eventueel de database geraadpleegd
5. De server geeft een resultaat terug in de vorm van een pagina (HTML en JavaScript)
6. De gebruiker heeft nu een applicatie in de browser in de vorm van een pagina met JavaScript
7. De gebruiker verricht handelingen, sommige handelingen zullen communiceren met de server. Om niet steeds een nieuwe pagina te hoeven laden en makkelijk met de applicatie te kunnen werken, zullen deze handelingen asynchroon gedaan worden (AJAX).
8. De server handelt de AJAX requests af, maar geeft het resultaat terug in de vorm van een JSON object zodat deze makkelijk in de JavaScript gebruikt kan worden. Dit proces vindt plaats in de C# controller, hier kan ook gecommuniceerd worden met de database.

Web applicaties zijn dus verdeeld over een client en een server kant. Gebruikers interacteren uitsluitend met de client kant. De client communiceert met de server en de server communiceert met de database.

Gebruiker 🡨🡪 Client 🡨🡪Server🡨🡪Database

De client staat nooit direct in contact met de database, elke actie waarbij de database nodig is verloopt dus altijd via de server.

Behandelde web technologieën

# Client

* HTML - <www.w3schools.com/html/>
* CSS - [www.w3schools.com/css/](http://www.w3schools.com/css/)
* JavaScript - [www.w3schools.com/js/](http://www.w3schools.com/js/)
* jQuery - [www.w3schools.com/jquery/](http://www.w3schools.com/jquery), <http://api.jquery.com/>  
  Een JavaScript library die heel veel handelingen sneller en makkelijker maakt.
* Typescript - <typescriptlang.org>, <https://github.com/borisyankov/DefinitelyTyped>

Superset van JavaScript waarbij je classes en types kunt gebruiken, de compiler kan dan checks doen op de geschreven code. Veel gebruikte libraries zijn JavaScript geschreven en beschikken dus niet over type informatie. Voor deze libraries bestaan meestal ‘typings’, dit zijn .d.ts files die je in je project kunt opnemen zodat type informatie wel beschikbaar is. Deze zijn te vinden door de naam van de library + .d.ts te googlen.

* Bootstrap - <getbootstrap.com>

Een CSS/JavaScript library voor opmaak van de HTML elementen, mooie dialoog schermen etc.

* Angular - <www.w3schools.com/angular/>, <https://angularjs.org>

Een JavaScript library waarmee een koppeling tussen JavaScript code en HTML elementen gemaakt kan worden. Bijv. als een input veld ingevuld wordt dan wordt het bijbehorende JavaScript variabele ook aangepast. Andersom verandert de inhoud van het input veld als de JavaScript variabele wordt aangepast.

* Jasmine - <http://jasmine.github.io/>

Unit testing framework

* Karma - <http://karma-runner.github.io>

Automatische runner van unit tests

# Webserver

* C# - <https://msdn.microsoft.com/en-us/library/67ef8sbd.aspx>
* ASP MVC - <http://www.asp.net/mvc>
* NewtonSoft.Json - <http://www.newtonsoft.com/json>
* Entity Framework - <https://msdn.microsoft.com/en-us/data/ef.aspx>
* Code First - <https://www.asp.net/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application>
* Database First - <https://www.asp.net/mvc/overview/getting-started/database-first-development/setting-up-database>