|  |
| --- |
| Educom|Detacom |
| Testcase software development v.1.2. |
| Project EducomOpdracht |

|  |
| --- |
| Justin Muris  6-24-2021 |

Contents

[Opdracht 2](#_Toc75443501)

[Stappenplan 3](#_Toc75443502)

[Eisen 4](#_Toc75443503)

[Algemeen idee 5](#_Toc75443504)

[Uitvoering 5](#_Toc75443505)

# Opdracht

Buienradar.nl stelt een API-koppelvlak ter beschikking van derden. Deze API kan geraadpleegd worden via **<https://data.buienradar.nl/1.1/feed/json>.**

* Bouw twee afzonderlijke applicaties:
  + De eerste communiceert periodiek, via een configuratiebestand in te stellen, met de API van Buienradar. De opgehaalde gegevens worden in een lokale database opgeslagen. Deze serverapplicatie heeft een specifiek API endpoint dat door de tweede applicatie geraadpleegd wordt. De API kent drie parameters:
    - Het betreffende weerstation;
    - De startdatum van de periode die we willen raadplegen;
    - De einddatum van de periode, bij default 7 dagen later dan de startdatum.
  + De tweede applicatie toont vervolgens een grafiek met de geselecteerde meeteenheden. Deze meeteenheden kunnen via checkboxes geselecteerd worden en zijn de meeteenheden zoals ze in de buienradar-API beschreven worden.

Voor de opdracht mag zelf bepaald worden waarin het ontwikkeld wordt, ook is een combinatie van programmeertalen toegestaan. Wees vooral creatief.

Het project wordt in een private repository opgeleverd en vervolgens gedeeld met “ReneKrewinkel” op GitHub.

Een werkende online versie met een goede gebruikershandleiding is een pre.

# Stappenplan

Als eerste stap heb ik een stappenplan opgesteld, om een soort van planning op te kunnen bouwen over de twee weken heen.

1. Creëer een Visual Studio solution met drie projecten erin.
   1. EducomOpdracht, ASP.NET Core Web App (Model-View-Controller);
   2. EducomOpdrachtAPI, ASP.NET Core Web API;
   3. EducomOpdrachtTaskScheduler, Console Application.
2. Maak een GitHub repository aan en upload deze (initial commit).
3. Eerst wordt de API in zijn basis opgezet, dit houdt in:
   1. GET-functie – haalt alles in de database op;
   2. POST-functie – plaatst een entry in de database;
   3. PUT-functie – updatet een entry in de database.
4. Als tweede wordt de console application gebouwd, dat als trigger wordt gebruikt om de database van de API te updaten met data uit de feed van Buienradar. Het enige dat de console applicatie doet is de API pingen/triggeren zodat het de functie uitvoert.   
   De console application wordt dan later ingesteld om periodiek uitgevoerd te worden.
5. Nu wordt de API verder uitgewerkt terwijl over tijd de database gevuld wordt met meer data uit Buienradar:
   1. Specifieke endpoints voor de GET-functie, een mét start- en eindperiode en een met alleen een startperiode;
   2. Een redirect van POST naar PUT zodra bekend is dat een entry al bestaat met dezelfde ID en/of datum;
   3. Een DELETE/wipe functie voor testing.
6. Wanneer de API gereed is, wordt aan de site gewerkt:
   1. Verbinding en dataretrieval met API;
   2. Data verwerken naar Model voor grafiekplotting;
   3. Grafiekplotting op basis van geselecteerde data;
   4. Meeteenheden togglen/filtreren.
7. Testen of alles goed werkt en een beetje idiot-proof is.
8. Achteraf wordt ruim gedocumenteerd wat gedaan is, wat er nog eventueel gedaan moet worden en wat eventuele problemen kunnen zijn.
9. Proberen te hosten, wat waarschijnlijk niet goed lukt want ik ben geen server-persoon.

# Eisen

Voor de eisen heb ik een tabel opgesteld. De eisen krijgen een unieke code, beschrijving en prioriteit. Dit fungeert ook als een soort-van to-do list.

De prioriteringswijze is MoSCoW – Must haves, should haves, could haves en won’t haves.

* Must haves: moeten erin, essentieel: topprioriteit;
* Should haves: zou erin moeten, maar niet essentieel: hoog prioriteit;
* Could haves: zou fijn zijn om erin te hebben, maar niet essentieel: laag prioriteit;
* Won’t have: zal niet erin zijn, verre van essentieel: geen prioriteit.

|  |  |  |
| --- | --- | --- |
| Code | Beschrijving | Prioriteit |
| SITE-001 | De site toon een grafiek met opgehaalde data uit de API. | Must |
| SITE-002 | De site heeft een verbinding met de API. | Must |
| API-001 | De API heeft een achterliggende database. | Must |
| API-002 | De API heeft een GET-functie voor weerberichten/weerstations. | Must |
| API-003 | De API heeft een POST-functie voor weerberichten/weerstations. | Must |
| API-004 | De API heeft een PUT-functie voor weerberichten/weerstations. | Should |
| API-005 | De API heeft een DELETE-functie voor weerberichten/weerstations. | Could |
| APP-001 | De console applicatie heeft toegang tot de API. | Must |
| APP-002 | De console applicatie triggert de API voor datacollectie. | Must |
| APP-003 | De console applicatie wordt periodiek uitgevoerd. | Must |

# Algemeen idee

De website toont de grafiek. De API levert data aan de website vanuit zijn database. De API verzamelt data uit de Buienradar-feed en verwerkt en plaatst deze in zijn database. De console applicatie triggert de API om data te verzamelen, want het is niet echt mogelijk om een periodieke task uit te voeren op een MVC pagina. De console applicatie wordt periodiek uitgevoerd door middel van Windows Task Scheduler, en de interval wordt hier ook geconfigureerd.

# Installatie

Het project gebruikt veel externe tools en frameworks en dergelijke, dus er wordt veel gedownload.

1. Download en installeer de dotnet hosting bundle installer: <https://dotnet.microsoft.com/download/dotnet/thank-you/runtime-aspnetcore-5.0.4-windows-hosting-bundle-installer>
   1. Dit bevat allerlei frameworks en libraries die nodig zijn voor het hosten van de server.
2. Download SQL Express 2019 <https://www.microsoft.com/en-us/Download/details.aspx?id=101064> en SSMS <https://aka.ms/ssmsfullsetup>.
   1. Dit zal de primaire database zijn inclusief zijn management-tool.
3. Installeer SQL Express 2019, gebruik de Basic profiel. Laat alle instellingen op standaard staan. Klik op het eind op ‘Connect Now’ om te testen of het werkt. Als het goed werkt, verschijnt een CMD venster met ‘1 row affected’ onderaan.
4. Installeer SSMS, houd alle instellingen standaard.  
   Eenmaal opgestart, verbind met ‘[COMPUTER]\SQLEXPRESS’ via   
   Windows Authentication. Dit staat waarschijnlijk allemaal al vooringesteld.
5. Creëer een database door links met de rechtermuisknop op Databases te klikken en dan ‘New database...’. Vul als database naam ‘EducomOpdrachtAPI’ in, en klik op OK.
6. Open het bijgeleverde ‘SQL Script.sql’ bestand. Dit zou moeten openen in SSMS. Voer deze uit door op F5 te drukken. Als goed is krijg je geen errors terug en staan er nu twee nieuwe Tables in de database.
7. Houd de Windows knop en R in, zodat het venster Uitvoeren verschijnt. Type hier ‘optionalfeatures’ in, zodat je het venster Windows-onderdelen ziet. Klik op de checkbox bij ‘Internet Information Services’. Open vervolgens de map en ga naar ‘World Wide Web-services’, ‘Toepassingontwikkelingsfuncties’ en klik op de checkboxes voor ‘ASP’, ‘ASP.NET 3.5’ en ‘ASP.NET 4.8’. Klik op OK en laat Windows zijn werk doen.
8. Wanneer het klaar is, houd de Windows knop en R in voor het venster Uitvoeren en type hier ‘InetMgr’ in. Klap links de folder uit en klik met de rechtermuisknop op ‘Sites’ en vervolgens op ‘Website toevoegen...’. Vul een sitenaam in, dit maakt niets uit. Ik heb zelf gekozen voor ‘Educom Opdracht Site.  
   Kies een fysiek pad om de server in op te slaan. Zet de binding-type op https en kies onderaan een SSL-certificaat (de IIS Express Development Certificate is prima).  
   Laat de rest van de instellingen standaard en klik op OK. Herhaal dit voor de andere API, maar zet de port één verder en kies natuurlijk een andere sitenaam.
9. Open de Visual Studio solution met de drie projecten. Klik met de rechtermuisknop op EducomOpdracht en klik op ‘Publish’. Kies als ‘Publish method’ File System, of de optie die je gewoon laat kiezen in welke folder het komt te staan. Kies als Target location het fysiek pad dat je eerder hebt gebruikt om de site in op te slaan. Zet de configuration op Release en laat de rest standaard staan. Klik na de setup boven op ‘Publish’ als het dit nog niet vanzelf heeft gedaan.  
   Herhaal dit voor de API.
10. ‘Publish’ de console application op dezelfde wijze, alleen dan niet in de IIS folder, via ‘Folder’ en dan weer ‘Folder’. Sla deze op een andere veilige, permanente plek op. Klik na de setup boven op ‘Publish’ als het dit nog niet vanzelf heeft gedaan.