

Academie IT en Mediadesign

Stress-testing met Apache JMeter

DOEL

- Voorbeeld applicatie ASP.NET6 Razor/MVC/Blazor
- Testplan maken in Jmeter
- Stress testing de applicatie
- Een aantal performance verbeteringen implementeren
- Meten
- Documenteren

VOORBEELDAPPLICATIE

- Download van Onderwijsonline: PerformanceTestProject.zip
- Open solution: PerformanceTestProject\MV_PerfTest_WebApp\MV_PerfTest_WebApp.sln

APPLICATIE

Gebruik het performanceTest project OF gebruik een project wat je al hebt gemaakt (bv. in de workshop) OF je prototype AIRBNB applicatie.

Maak listings model

Scaffold CRUD Listing page:

- Voeg pages map toe,

- Add → new Scaffolded item → Razor Pages using Entity Framework (CRUD).

- Model class: Kies Listing model

- Data context class: naamproject.Data.ListingContext

APPLICATIE (VERVOLG)

Update Database connection string naar Airbnb db in appsettings.json

```
1 {  
2   "Logging": {  
3     "LogLevel": {  
4       "Default": "Information",  
5       "Microsoft": "Warning",  
6       "Microsoft.Hosting.Lifetime": "Information"  
7     }  
8   },  
9   "AllowedHosts": "*",  
10  "ConnectionStrings": {  
11    "ListingContext": "Server=DESKTOP-DSDR416;Database=AirBNB;Trusted_Connection=True;MultipleActiveResultSets=true"  
12  }  
13 }
```

Voeg pageroute optie toe in Startup.cs (service.AddRazorPages)

Remove old index and privacy pages

```
services.AddRazorPages()  
    .AddRazorPagesOptions(options =>  
    {  
        options.Conventions.AddPageRoute("/Listings/Index", "");  
    });
```

APPLICATIE (VERVOLG)

Add paginering

- Class PaginatedList toevoegen in root
- edit Index.cshtml.cs voor paginering
- edit index.cshtml voor paginering

Voorbeeldbestanden staan in BestandenLoadTestApp.zip (toevoegen door uit te pakken en in juiste map in VS: Add existing Item uitvoeren)

Wijzig aantal resultaten per pagina in Index.cshtml.cs naar 50

APACHE JMETER

Download via:

JAVA8+ SDK of JRE nodig.

Documentatie op:

De geconfigureerde Apache test kan gedownload worden via onderwijsonline.

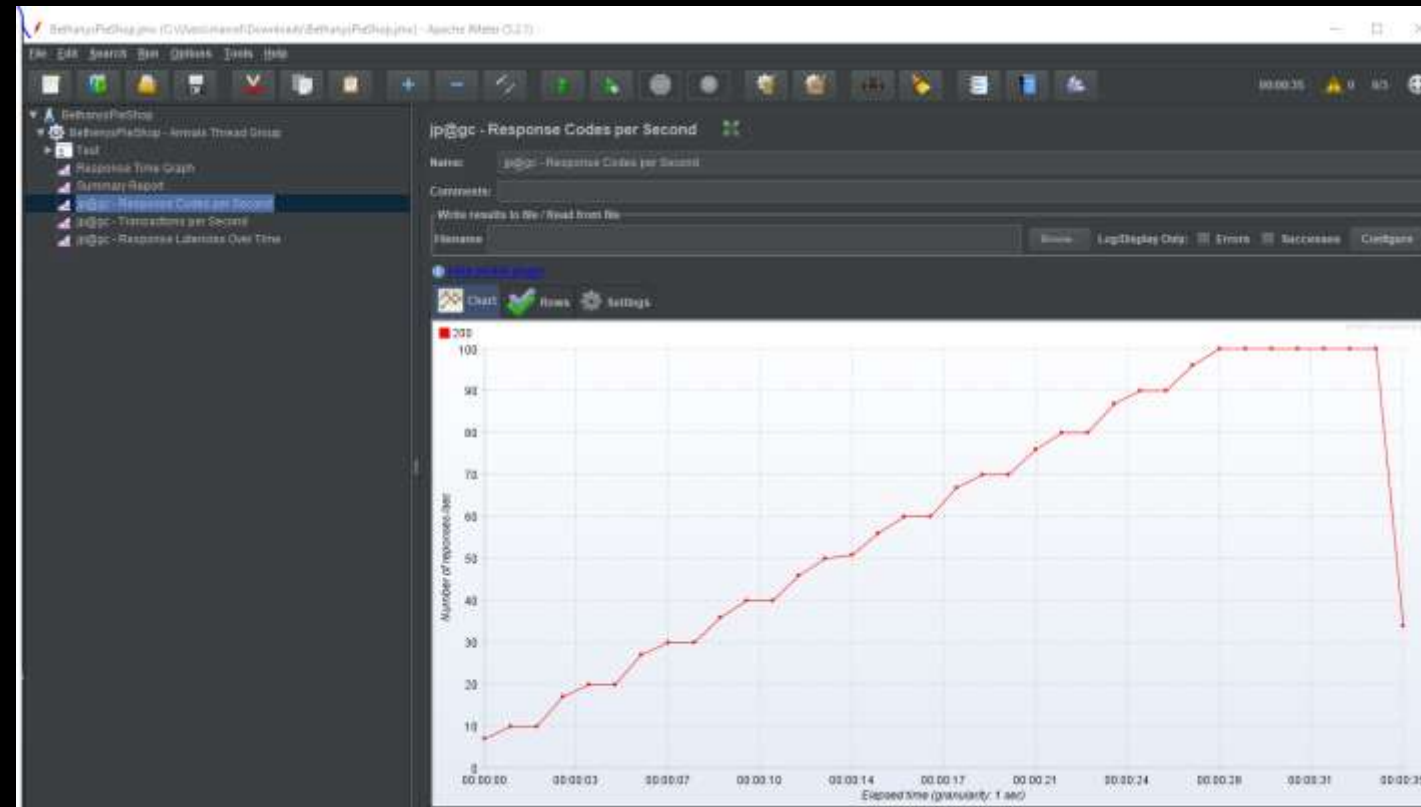
TESTPLAN

Jmeter Plugins via Jmeter-plugins.org (download en plaats in Jmeter map lib/ext), via Options toegang tot plugins:

- Plugins manager

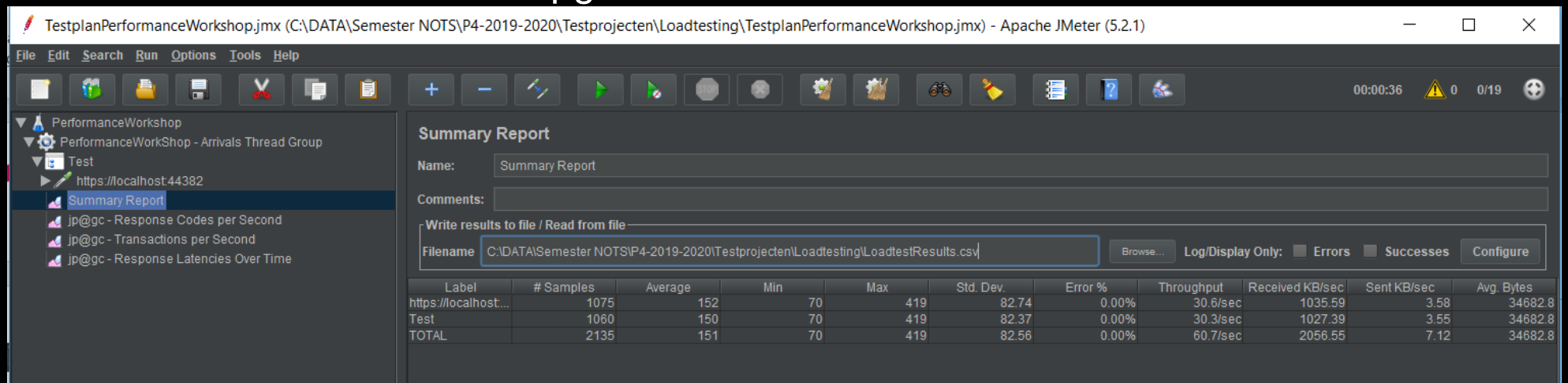
De rest kan geïnstalleerd worden via de Plugin-manager:

- Custom Thread groups
- Parallel Controller & Sampler
- 3 basic graphs
- 5 additional graphs
- Testplan:
 - Arrivals Thread Group
 - Transaction controller
 - Sampler – HTTP request
 - Graphs
 - Reports



PREPARE TEST

In Summary report → Configure → Check alle opties behalve de XML
Bij filename het resultatenbestand als csv opgeven.



Kopieer in Jmeter/Bin map de inhoud van reportgenerator.properties naar user.properties
(notepad++ is het eenvoudigst)

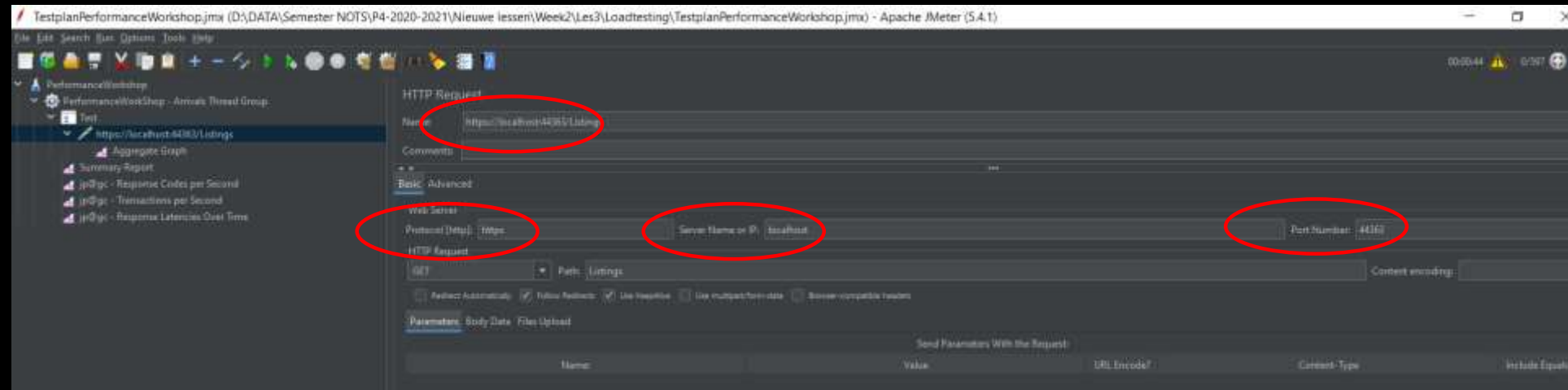
Wijzig in user.properties de
Granularity naar 1500

```
227 # Defines the overall granularity for over time graphs
228 # Granularity must be higher than 1000 (1second) otherwise Throughput graphs will be incorrect
229 # see Bug 60149
230 jmeter.reportgenerator.overall_granularity=1500
231
```

PREPARE TEST

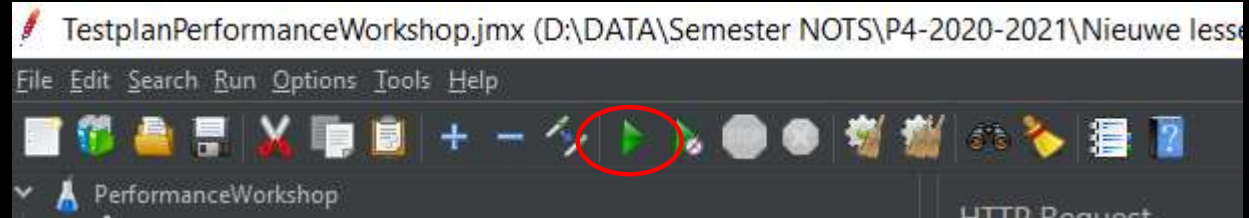
Bij HTTP Request de settings controleren en zonodig wijzigen:

- Protocol
- Server Name of IP
- Port number
- Path
- In mijn geval:



RUN TEST

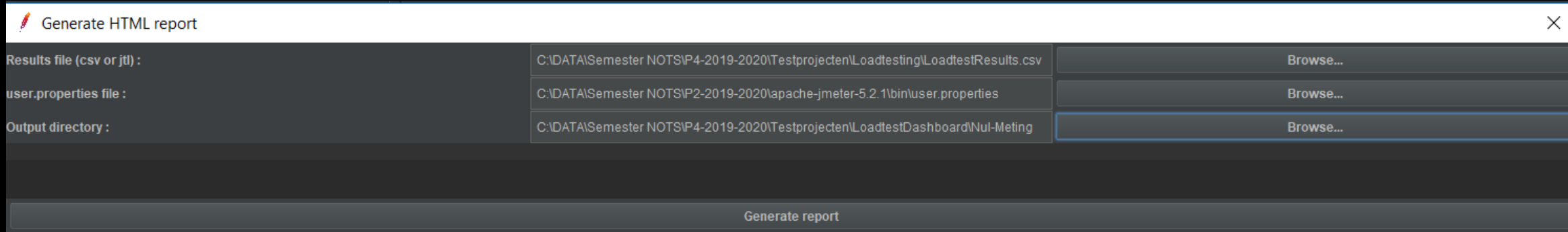
Klik in de menubalk op de groene pijl:



De test zal nu starten met de settings zoals gedefinieerd in de Arrivals Thread Group. Deze settings niet maar aanpassen nadat de nulmeting is gedaan, anders kun je de resultaten niet vergelijken.

GENERATE HTML REPORT

Dan in Jmeter: Tools → Generate HTML report



The screenshot shows the 'Generate HTML report' dialog box in JMeter. It has a title bar with a pencil icon and a close button. The dialog contains three rows of input fields and buttons:

Field Label	Field Value	Button
Results file (csv or jtl) :	C:\DATA\Semester NOTS\IP4-2019-2020\Testprojecten\Loadtesting\LoadtestResults.csv	Browse...
user.properties file :	C:\DATA\Semester NOTS\IP2-2019-2020\apache-jmeter-5.2.1\bin\user.properties	Browse...
Output directory :	C:\DATA\Semester NOTS\IP4-2019-2020\Testprojecten\LoadtestDashboard\Nul-Meting	Browse...

At the bottom of the dialog is a large button labeled 'Generate report'.

Open index.html

In Charts kies Response Times Over Time, Active Threads Over Time, Latencies Over Time
Download de charts door klik rechts op steeksleutel en Save as PNG.

Gebruik deze in Document.

EERSTE PERFORMANCE IMPROVEMENT: RESPONSE COMPRESSION

Voeg aan startup.cs toe:

Eerste regel in ConfigureServices

```
//First performance action: Register Response in DI container  
Service.AddResponseCompression();
```

Eerste regel in Configure

```
//First performance action: Add response compression to Middleware  
app.UseResponseCompression();
```

Voer nieuwe meting uit met exact dezelfde settings. Sla resultaten op in nieuw csv bestand.
Genereer nieuw HTML report in nieuwe map.

Vergelijk en documenteer dit.

TWEEDE PERFORMANCE ACTIE: RAZOR CACHEHELPER TAGHELPER

Voeg in de index.cshtml toe:

zet het foreachblok waarin de data gedisplayed wordt in een cache blok:

```
<cache expires-after="@Timespan.FromSeconds(60)">
```

```
</cache>
```

En meet nog een keer.

OPDRACHTEN

Wijzig het testplan in Jmeter na de laatste meting zodanig zodat een nieuwe operational ceiling bepaald kan worden.

Documenteer dit.

Dezelfde aanpak moet uitgevoerd worden voor de AirbBNB applicatie