



Technische Documentatie - ASP GATAM   
ISW-IP  
Group 8

Applied Software Project

ondersteund door de

Mariën Sven

DePooterMarijn

AP Hogeschool

begeleid door het bedrijf

GATAM

Studenten: Van Vaerenbergh Nils, Makdesi Tony, Baqloul Haitam, Onya Lauren

Table of Contents

[Projectoverzicht 2](#_Toc180605108)

[Technische Stack 2](#_Toc180605109)

[Project Structure 2](#_Toc180605110)

[Setup en Installatie 3](#_Toc180605111)

[Belangrijkste Kenmerken 4](#_Toc180605112)

[Codeconventies en Architectuur 4](#_Toc180605113)

[Testen en Implementatie 4](#_Toc180605114)

[Toekomstige Verbeteringen 4](#_Toc180605115)

# Projectoverzicht

Projectnaam: ASP - GATAM ISW-IP  
Doel: dit project is gericht op het ontwikkelen van een kernapplicatie volgens de principes van Clean Architecture, met duidelijke lagen die applicatielogica, infrastructuurkwesties en gebruikersinterface beheren. Het doel is om een robuuste oplossing te bieden voor het effectief beheren van gegevens en het integreren van externe systemen.

# Technische Stack

- Programmeertaal: C#  
- Framework: .NET 8  
- Architectuur: Clean Architecture  
- Containerisatie: Docker  
- CI/CD: GitLab pipelines

# Project Structure

Het project is gestructureerd in duidelijke lagen volgens de principes van Clean Architecture. Elke laag heeft zijn eigen verantwoordelijkheden en maakt gebruik van specifieke technologieën en patronen.  
  
- **ApplicationCore/**

* **Beschrijving:** bevat de kernapplicatielogica en de business rules die de primaire functionaliteiten van de applicatie definiëren.
* **Submappen:**
  + **Gatam.Application/**: bevat de use cases en services die de bedrijfslogica implementeren en ook de interfaces die de communicatie met de infrastructuurlaag definiëren, zoals repositories en services.
  + **Gatam.Domain/**: bevat domeinmodellen en entiteiten,

- **Infrastructure/**

* **Beschrijving:** beheert infrastructuurkwesties, zoals toegang tot databases, externe API’s en andere services.
* **Submappen:**
  + **Gatam.Infrastructure.Cotexts/**: bevat de databasecontext en migraties voor SQL Server.
  + **Gatam.Infrastructure.Repositories/**: bevat implementaties van repositories die toegang bieden tot de gegevenslaag.

- **Presentation/**

* **Beschrijving:** beheert de gebruikersinterface en zorgt voor de interactie met de eindgebruikers.
* **Submappen:**
  + **Gatam. WebAppBegeleider/**: bevat de website toegankelijk voor de beheerders en de begeleiders. Deze is gebouwd op Blazor server framework.
  + **Gatam. WebAppVolger/**: bevat de website toegankelijk voor de gebruiker. Deze is gebouwd op Blazor server framework.

- **Solution Items/**

* **Beschrijving:** bevat gemeenschappelijke configuratiebestanden en hulpmiddelen voor de gehele oplossing.
* **Bestanden:**
  + **.gitignore**: definieert welke bestanden en mappen moeten worden genegeerd door Git.
  + **docker-compose.yml**: bevat configuraties voor het opzetten van de Docker-containers voor de verschillende lagen van de applicatie.
  + **README.md**: bevat algemene informatie en instructies voor het project.

# Setup en Installatie

Om de server efficiënt en foutloos op te zetten, is een setup-script ontwikkeld dat de benodigde dependencies installeert en handige feedback geeft tijdens het proces. Hieronder volgt een duidelijk stappenplan om de deployment-omgeving op te zetten.

## 1. Overzetten van de benodigde scripts

Er zijn twee opties om de scripts over te zetten naar de server:

Optie 1: Git installeren en de repository clonen  
Gebruik deze methode als je de volledige projectrepository nodig hebt.  
Als je kiest voor deze optie ga je ervoor moeten zorgen dat je kan clonen aan de hand van een token.

Optie 2: Bestanden overzetten via SFTP (aanbevolen)  
Deze methode maakt het mogelijk om alleen de essentiële bestanden over te zetten en bespaart serveropslag.

## 2. Stappen voor SFTP:

Maak een SFTP-verbinding naar de server.

Kopieer de benodigde bestanden uit de map asp-gatam-isw-ip/Server Setup naar een gewenste locatie op de server.

In dit voorbeeld kies ik de locatie: /etc/gatam/setup/.

## 3. Uitvoeren van de setup-script

Het setup.sh-script is ontworpen om het volledige setup automatisch uit te voeren. Dit script zal:

* Alle vereiste dependencies installeren.
* Noodzakelijke bestanden en configuraties aanmaken.

Stappen:

1. Navigeer naar de map waar je de bestanden hebt overgezet (bijvoorbeeld /etc/gatam/setup/).
2. Voer het script uit met het volgende commando:   
   bash setup.sh
3. Volg de instructies die het script weergeeft.

A screenshot of a computer

Description automatically generated

## 4. Toevoegen van de benodigde credentials

Na het succesvol uitvoeren van de setup-script, worden er automatisch twee .env-bestanden gegenereerd in /etc/gatam/:

* .env.staging
* .env.production

Stappen:

1. Open het bestand .env.example uit het project.
2. Vul de gegenereerde .env-bestanden aan met de juiste keys en hun waarden zoals gedefinieerd in .env.example.
3. Sla de wijzigingen op.

## 5. Configureren van de reverse proxy

Voor de configuratie van domeinnamen en het correct routeren van inkomende verbindingen, wordt de tool Traefik gebruikt. Dit beheert de koppeling tussen inkomende requests en de juiste services.

Stappen:

* Voeg een .env-bestand toe in de setup-folder met de benodigde keys uit .env.example.
* Voer het script DeployProxySetup.sh uit met het volgende commando:   
  bash DeployProxySetup.sh
* Controleer of er een Docker-container draait met Traefik als image.
* Gebruik het commando docker ps om dit te verifiëren.

## 6. Aanmaken van runners

Voor het configureren van GitLab Runners kun je de officiële documentatie volgen:

GitLab Runner Registratie: <https://docs.gitlab.com/runner/register/>.

De CI/CD pipeline maakt gebruik van onderstaande runners. Het is dus handig om deze waarden over te nemen. De tag waaronder de runner is aangemaakt is zeer belangrijk.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Runner name** | **Executor** | **Tags** | **Runner ID** | **Stage** |
| ASP\_GATAM\_DEPLOYMENT\_MAIN | shell | production | hGvssUdEh | production-deployment |
| ASP\_GATAM\_STAGING | shell | staging | Bv5eQuYEm | staging-deployment |
| ASP\_GATAM\_GLOBAL | shell | global | 9s5yrgu6U | test, build |

# Belangrijkste Kenmerken

- **Gebruikersauthenticatie:** ondersteunt individuele authenticatiemechanismen voor inloggen en registreren  
- R**olgebaseerde Toegangscontrole:** beheert toegang tot specifieke delen van de app met rollen zoals admin, editor en gebruiker.  
- **Database-integratie:** SQL Server wordt gebruikt voor gegevensopslag.

## Codeconventies en Architectuur

Het project houdt zich aan de principes van Clean Architecture, waarbij verantwoordelijkheden in verschillende lagen worden gescheiden:  
1. **Application Core:** bevat de bedrijfslogica en applicatieregels.  
2. **Infrastructure:** beheert externe bronnen zoals databases en bestandsystemen.  
3. **Presentatielaag:** beheert de gebruikersinterface, waarschijnlijk met Blazor.  
  
Ontwerppatronen zoals CQRS en het Repository-patroon worden toegepast om de scheiding van verantwoordelijkheden en efficiënt gegevensbeheer te waarborgen.

# Testen en Implementatie

- **Unit- en Integratietests:** tests zijn te vinden in de tests/ map.  
- **CI/CD:** geautomatiseerd met GitLab-pijplijnen, die het project bouwen en implementeren in containerized omgevingen.

# Toekomstige Verbeteringen

* Bij het deployen van applicaties naar de staging- en productie-omgevingen worden oude versies van de containerimages niet automatisch verwijderd. Dit leidt tot een snelle opeenhoping van ongebruikte images op de server, waardoor de beschikbare opslagruimte aanzienlijk wordt beperkt. Als dit probleem niet wordt aangepakt, kan het resulteren in een volgelopen server, wat negatieve gevolgen heeft voor de stabiliteit en beschikbaarheid van de omgeving.

Daarnaast heeft deze situatie ook negatieve effecten op de test- en buildfasen van het ontwikkelproces. Het gebrek aan beschikbare opslagruimte kan ervoor zorgen dat builds mislukken of vertraging oplopen, wat de ontwikkelcyclus vertraagt en de efficiëntie van het team vermindert. Bovendien kunnen volgelopen servers tijdens het testen leiden tot inconsistente resultaten of downtime, wat de betrouwbaarheid van de testomgeving ondermijnt. Een proactieve strategie voor het beheren en opschonen van ongebruikte images is essentieel om deze problemen te voorkomen en de workflow soepel te laten verlopen.