TESTPLAN

SOFTWARELICENTIEMODEL

Datum	02-01-2023
Versie	1.0
Status	Final
Auteur	Patrick van Nieuwburg

Versie

Versie	Datum	Auteur(s)	Wijzigingen	Status
0.1	02-01-2023	PvN	Toevoegen acceptatietest	Afgerond
1.0	06-01-2023	PvN	Toevoegen service- unittests	Afgerond

Verspreiding

Versie	Datum	Aan
1.0	10-01-2023	Jeroen van Rijckevorsel
1.0	10-01-2023	Inleveren portfolio – Mark Melstrom

Inhoudsopgave

1. Begrippenlijst

Afkorting/ begrip	Beschrijving
TBWB	Mijn stagebedrijf
API	Een Application Programming Interface, wordt gebruikt om op een geformaliseerde manier gegevens uit te wisselen tussen apps.
Https	Hypertext Transfer Protocol Secure is een uitbreiding op het HTTP-protocol met als doel een veilige uitwisseling van gegevens.
JWT	Een JSON-webtoken (JWT) is een JSON-object dat wordt gebruikt om informatie veilig via internet over te dragen tussen twee partijen. Het kan worden gebruikt voor een authenticatiesysteem en kan ook worden gebruikt voor informatie-uitwisseling.

2. Testplan

Voor het maken van mijn applicatie wil ik me focussen op drie verschillende testen. Allereerst wil ik de opgestelde functionele eisen controleren door een acceptatietest uit te voeren op de stories die zijn ontwikkeld. Hiermee kan ik uiteindelijk concluderen of mijn applicatie voldoet aan de eisen die ik met de stakeholder heb opgesteld.

De tweede test die ik wil gaan maken zijn servicetests. Deze testen zijn bedoeld om de werking van mijn API te controleren. Ik maak calls en ik controleer of deze calls het juiste antwoord teruggeven.

De derde test die ik wil gaan maken zijn de unittesten. Ik zeg heel eerlijk dat ik deze applicatie niet volledig TDD heb opgezet. Ik ben namelijk eerst begonnen met de grote bouwstenen en daarna pas heb gekeken welke stukjes code ik onder test wil hebben staan.

2.1. Acceptatietest

Voor de acceptatietest heb ik in mijn analysedocument gekeken welken eisen er zijn gesteld aan de applicatie. Deze heb ik hieronder benoemd en heb ik beschreven wat het gewenst gedrag is, en wat het daadwerkelijke gedrag is. In mijn conclusie kan ik dan bepalen of de applicatie goed genoeg is om in productie te gaan.

2.1.1. Functionele eisen

FE1	Admins moeten gebruikerslicenties kunnen toevoegen.
Gewenst gedrag	
1. Wanneer de admin inlogt op het licentiemanager dashboard moet de admin gepresenteerd worden met een scherm waarin de huidige licentie beschreven staat. Dit is de klantnaam, aantal gebruikerslicenties, laatste betaalde factuurdatum en wanneer de volgende factuurdatum verloopt. 2. Wanneer de admin op de knop 'Change license amount' klikt wordt de admin gepresenteerd met een scherm waarin hij de gebruikers kan toevoegen. 3. Als de admin de juiste gegevens invoert wordt hij teruggestuurd naar de licentie beheer pagina en de aantal gebruikers zijn aangepast. 4. Als de admin de onjuiste gegevens invoert wordt er een scherm getoond dat hij onjuiste gegevens heeft ingevoerd en dat hij het nog eens moet proberen	
Daadwerkelijk gedrag	
1. De admin wordt ingelogd op de pagina en ziet de juiste gegevens 2. Er wort een scherm geopend waarin de admin de gebruikersaantallen kan wijzigen. 3. De admin wordt teruggestuurd naar de pagina en de juiste aantallen zijn weergegeven 4. De admin wordt gewaarschuwd dat hij de verkeerde gegevens heeft ingevuld.	
Test geslaagd	**JA**

FE2	Admins moeten gebruikerslicenties kunnen verwijderen.
Gewenst gedrag	
1. Wanneer de admin inlogt op het licentiemanager dashboard moet de admin gepresenteerd worden met een scherm waarin de huidige licentie beschreven staat. Dit is de klantnaam, aantal gebruikerslicenties, laatste betaalde factuurdatum en wanneer de volgende factuurdatum verloopt. 2. Wanneer de admin op de knop 'Change license amount' klikt wordt de admin gepresenteerd met een scherm waarin hij de gebruikers kan verwijderen. 3. Als de admin de juiste gegevens invoert wordt hij teruggestuurd naar de licentie beheer pagina en de aantal gebruikers zijn aangepast. 4. Als de admin de onjuiste gegevens invoert wordt er een scherm getoond dat hij onjuiste gegevens heeft ingevoerd en dat hij het nog eens moet proberen	
Daadwerkelijk gedrag	
1. De admin wordt ingelogd op de pagina en ziet de juiste gegevens 2. Er wort een scherm geopend waarin de admin de gebruikersaantallen kan wijzigen. 3. De admin wordt teruggestuurd naar de pagina en de juiste aantallen zijn weergegeven 4. De admin wordt gewaarschuwd dat hij de verkeerde gegevens heeft ingevuld.	
Test geslaagd	**JA**

FE3	Alleen de hoeveelheid beschikbare gebruikerslicenties mag toegelaten worden op het systeem.
Gewenst gedrag	
1. De medewerker probeert in te loggen op StockControl 2. Het systeem controleert het certificaat en laat de gebruiker toe als het gebruikers limiet niet wordt overschreden. 3. De medewerken probeert in te loggen op StockControl 4. Het systeem controleert het certificaat en laat de gebruiker niet toe wanneer het gebruikers limiet wordt overschreden.	
Daadwerkelijk gedrag	
1. De medewerker probeert in te loggen op StockControl 2. Hij wordt toegelaten op het systeem 3. De medewerker probeert in te loggen op StockControl 4. Hij wordt niet toegelaten op het systeem.	
Test geslaagd	**JA**

FE7	De licentiemanager moet de eerste twee weken na het niet betalen van de factuur een melding genereren in de UI die de werknemer elke dag moet wegklikken.
Gewenst gedrag	
1. De medewerker logt in op StockControl met een licentie die al 2 weken na de factuurdatum loopt. 2. Het systeem controleert het certificaat op de laatst betaalde datum. Wanneer dit 2 weken na de huidige datum is wordt er een melding in de UI getoond die de klant elke dag moet wegklikken	
Daadwerkelijk gedrag	
1. De medewerker logt in op StockControl 2. Het systeem genereerd een melding in de UI getoond	
Test geslaagd	**JA**

FE8	De licentiemanager moet, na de eerste twee weken, en voor de eerste maand, na het niet betalen van de factuur, een melding genereren in de UI die de werknemer elk uur moet wegklikken.
Gewenst gedrag	
3. De medewerker logt in op StockControl met een licentie die al meer dan 2 weken na de factuurdatum loopt. 4. Het systeem controleert het certificaat op de laatst betaalde datum. Wanneer dit 2 weken na de huidige datum is wordt er een melding in de UI getoond die de klant elk uur moet wegklikken	
Daadwerkelijk gedrag	
3. Nog niet geimplementeerd	
Test geslaagd	**NEE**

FE9	TBWB moet na de eerste maand van het niet betalen van de factuur de mogelijkheid hebben de orderverwerking omlaag te zetten
Gewenst gedrag	
5. De medewerker logt in op StockControl met een licentie die al 1 maand na de factuurdatum loopt. 6. Het systeem controleert het certificaat op de laatst betaalde datum. Wanneer dit 1 maand is na de huidige datum is wordt de huidige orderverwerking verlaagd.	
Daadwerkelijk gedrag	
4. Nog niet geimplementeerd	
Test geslaagd	**NEE**

FE10	TBWB moet na de eerste maand van het niet betalen van de factuur de mogelijkheid hebben de gebruikerslicenties te verminderen
Gewenst gedrag	
1. De medewerker logt in op StockControl met een licentie die al 1 maand na de factuurdatum loopt. Op het systeem zijn op dit moment al drie andere gebruikers ingelogd. 2. Het systeem controleert het certificaat op de laatst betaalde datum. Wanneer dit 1 maand is na de huidige datum is wordt het aantal gebruikers dat toegalaten wordt verminderd naar drie gebruikers. 3. De medewerker krijgt de melding dat de aantal gebruikers dat toegelaten wordt op StockControl overschreden word en kan niet inloggen.	
Daadwerkelijk gedrag	
1. De medewerker logt in op StockControl 2. De medewerker krijgt een melding dat de aantal gebruikers wordt overschreden	
Test geslaagd	**JA**

FE11	De afdeling service heeft de mogelijkheid om een tijdelijke licentie aan te maken
Gewenst gedrag	
1. Een medewerker van de afdeling service van TBWB logt in op het licentiemanager dashboard 2. De medewerker wordt doorgestuurd naar de servicepagina. 3. De medewerker kiest een bedrijf uit het drop down menu waarvoor hij een certificaat wilt genereren 4. Hij drukt op de ‘genereer’ knop 5. Er wordt in de service folder een nieuw certificaat geplaatst	
Daadwerkelijk gedrag	
1. Een medewerker van de afdeling service van TBWB logt in op het licentiemanager dashboard 2. De medewerker wordt doorgestuurd naar de servicepagina. 3. De medewerker kiest TBWB uit het dropdown menu 4. Hij drukt op de ‘genereer’ knop 5. Er wordt in de service folder een nieuw certificaat geplaatst voor TBWB	
Test geslaagd	**JA**

FE12	De licentieservice stuurt een mail naar de klant van TBWB bij het aanmaken van een gebruiker
Gewenst gedrag	
1. De admin verhoogt het aantal gebruikerslicenties (FE1). 2. De admin krijgt een mail met de gemaakte wijzigingen.	
Daadwerkelijk gedrag	
1. Zodra er een wijziging plaatsvindt in het aantal gebruikerslicenties wordt de admin op de hoogte gehouden van deze wijzigingen via de mail.	
Test geslaagd	**JA**

FE13	De licentieservice stuurt een mail naar de klant van TBWB bij het verwijderen van een gebruiker
Gewenst gedrag	
1. De admin verlaagt het aantal gebruikerslicenties (FE2). 2. De admin krijgt een mail met de gemaakte wijzigingen.	
Daadwerkelijk gedrag	
1. Zodra er een wijziging plaatsvindt in het aantal gebruikerslicenties wordt de admin op de hoogte gehouden van deze wijzigingen via de mail.	
Test geslaagd	**JA**

FE14	De licentieservice stuurt de afdeling service een mail bij het wegvallen van de verbinding met de licentiemanager
Gewenst gedrag	
1. Zodra het watchdog component van de licentieservice in de gate krijgt dat er geen verbinding meer is tussen de licentieservice en de licentiemanager zal er via de mailserver een mail worden verstuurd naar de afdeling service van TBWB. Hierin staat dat de verbinding is weggevallen	
Daadwerkelijk gedrag	
1. Nog implementeren	
Test geslaagd	**NEE**

FE15	De licentiemanager geeft een pop-up aan de klant bij het wegvallen van de verbinding met de licentieservice
Gewenst gedrag	
1. Zodra het watchdog component van de licentiemanager in de gate krijgt dat er geen verbinding meer is tussen de licentieservice en de licentiemanager zal er in de UI van StockControl een pop up verschijnen met de melding of de klant contact wil opnemen met de afdeling service.	
Daadwerkelijk gedrag	
2. Nog implementeren	
Test geslaagd	**NEE**

2.1.2. Conclusie

De applicatie is nog niet helemaal af dus er moet nog wat getest worden. Hoe het er nu voorstaat is mijn applicatie zeker nog niet klaar voor in productie. Zodra ik deze eisen in mijn applicatie heb ingebouwd zal ik mijn documentatie bijwerken. Het eindresultaat acceptatietest zal ik dan ook presenteren bij mijn examenzitting om 2 februari 2023.

2.2. Servicetests

De services binnen mij applicatie moeten voorzien zijn van testen om te kijken of de API-calls de response geven die ik verwacht met de juiste content. Om dit te testen heb ik gebuik gemaakt van een testbase. Deze testbase wordt aangeroepen voordat elke test gedraaid wordt om te zorgen dat ik een actieve verbinding heb naar mijn swagger API. Deze testbase gebruik ik dan om een client te starten voor mijn testen.

2.2.1. Testbase

```
0 references
public ApiServiceTestBase()
{
    _webApplicationFactory = new WebApplicationFactoryServiceTests();
    _connectionString = GetConnectionString();
    _databaseName = GetDatabaseName(_connectionString);
    if (SystemTextJsonSerializerConfig.Options.PropertyNameCaseInsensitive)
    {
        SystemTextJsonSerializerConfig.Options.PropertyNameCaseInsensitive = false;
        SystemTextJsonSerializerConfig.Options.PropertyNamingPolicy = JsonNamingPolicy.CamelCase;
    }

    StartServer();
}

1 reference
private void StartServer()
{
    _ = _webApplicationFactory.Server;
}

2 references
public HttpClient CreateClient() => _webApplicationFactory.CreateClient();

1 reference
protected virtual void Dispose(bool disposing)
{
    if (!disposing)
    {
        return;
    }

    _webApplicationFactory.Dispose();
    NpgsqlConnection.ClearAllPools();
}
```

Het belangrijkste wat ik hier doe is het aanmaken van een WebApplicationFactoryServiceTest. Dit zorgt ervoor dat er een nieuwe http-verbinding gestart wordt waar mijn servicetesten tegen aan kunnen testen.

```
_webApplicationFactory = new WebApplicationFactoryServiceTests();
```

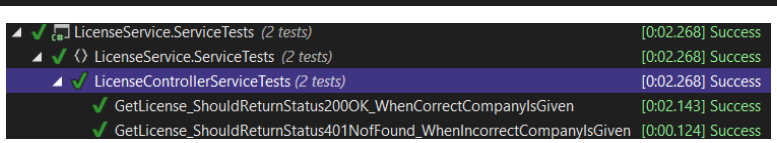
2.2.2. LicenseControllerServiceTests

In het voorbeeld dat ik ga geven wil ik mijn API aanroepen om een licentie op te vragen voor het bedrijf dat in de claim van de JWToken vermeld staat. Ik begin met het maken van een JWToken.

```
2 references
private void CreateJwToken(string company)
{
    var claims = new List<Claim>
    {
        new(type: "company", value: company)
    };

    var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(
        s: "NjOZM^2gc17sSCEbXwVKMzYn9$f&rjP%"));

    var token = new JwtSecurityToken(
        issuer: "TBWB/auth",
        audience: "TBWB clients",
        claims: claims,
        notBefore: DateTime.UtcNow,
        expires: DateTime.UtcNow.AddHours(8),
        signingCredentials: new SigningCredentials(key, algorithm: SecurityAlgorithms.HmacSha256)
    );

    JwtToken = new JwtSecurityTokenHandler().WriteToken(token);
}
```

Ik geef als parameter mee, de naam van het bedrijf dat ik wil ophalen uit mijn backend. In dit voorbeeld is dat tbwb. In mijn test roep ik dan de createJwToken() aan, en gebruik ik de testbase om een client te creëren. Op deze client roep ik mijn https request aan en verwacht ik dat ik een Httpsstatuscode 200 terugkrijg. Daarnaast bevestig ik of ik wel de juiste data terug krijg in de response van het request.

```
[Fact]
0 references
public async Task GetLicense_ShouldReturnStatus200OK_WhenCorrectCompanyIsGiven()
{
    // Arrange
    CreateJwToken(company: "tbwb");
    using var client = CreateClient();
    client.DefaultRequestHeaders.Authorization =
        new AuthenticationHeaderValue(scheme: "Bearer", parameter: JwtToken);

    // Act
    var response = await client.GetAsync(requestUri: "https://localhost:44363/license");

    // Assert
    response.Should() // HttpResponseMessageAssertions
        .Be200Ok()
        .And.Satisfy(
            givenModelStructure: new[]
            {
                new
                {
                    company = default(string), actualAmount = default(int),
                    lastPayment =
                        JsonConvert.DeserializeAnonymousType(value: response.ToString(),
                            anonymousTypeObject: default(string)),
                    expiryDate =
                        JsonConvert.DeserializeAnonymousType(value: response.ToString(),
                            anonymousTypeObject: default(string)),
                }
            },
            assertion: model: {company,actualAmount,…}[] => model.Should()
                .ContainEquivalentOf(expectation: new {company = "tbwb"}));
}
```



2.3. Unittesten

Om de functionaliteit van de klassen te testen maak ik gebruik van unittests. Door kleine stukjes code onder test te zetten zorg je ervoor dat je bij eventuele wijzigingen in de toekomst snel alle test kan draaien om te zien of je niks af hebt gebroken.

Ik het voorbeeld hieronder heb ik ervoor gekozen om mijn TokenGenerator onder test te zetten. Dit stuk code is ervoor verantwoordelijk om een geldige JWT-token te genereren die verstrekt kan worden aan de front-end gebruiker.

Als eerste heb ik een mock of een vervanger van de IConfiguration nodig. Deze interface gebruik ik in mijn applicatie om bepaalde gegevens op te vragen uit mijn appsettings.json bestand. Ik gebruik NSubstitude om een vervanger te maken en zorg ervoor dat deze vervanger bij het vragen naar de configuratie, data teruggeeft die ik wil voor deze test.

Daarna maak ik een TokenGenerator aan met deze configuratie en roep ik GenerateAccessToken aan met een vooraf aangemaakte user.

Ik controleer of deze token niet leeg is en daarna controleer ik de data die ik mee heb gegeven aan de claims van de token.

```
public class TokenGeneratorUnitTests
{
    [Fact]
    0 references
    public void CreateBearerToken_ShouldCreateValidToken()
    {
        // Arrange
        var configuration = Substitute.For<IConfiguration>();
        configuration["JwtBearer:TokenSecret"].Returns(returnThis: "testTokenSecretThatIs128BitsLong");
        configuration["JwtBearer:Issuer"].Returns(returnThis: "testIssuer");
        configuration["JwtBearer:Audience"].Returns(returnThis: "testAudience");
        var tokenGenerator = new TokenGenerator(configuration);
        var user = new User(name: "testName", lastName: "testLastName", email: "testEmail", company: "testCompany", role: "tes

        // Act
        var token :string = tokenGenerator.GenerateAccessToken(user);

        // Assert
        token.Should().NotBeNullOrEmpty();

        var handler = new JwtSecurityTokenHandler();
        var jwtToken = handler.ReadJwtToken(token.Replace(oldValue: "Bearer ", newValue: ""));
        jwtToken.Claims.Should().Contain(c :Claim => c.Type == "name" && c.Value == "testName");
        jwtToken.Claims.Should().Contain(c :Claim => c.Type == "lastname" && c.Value == "testLastName");
        jwtToken.Claims.Should().Contain(c :Claim => c.Type == "email" && c.Value == "testEmail");
        jwtToken.Claims.Should().Contain(c :Claim => c.Type == "company" && c.Value == "testCompany");
        jwtToken.Claims.Should().Contain(c :Claim => c.Type == "role" && c.Value == "testRole");
        jwtToken.ValidTo.Should().BeAfter(DateTime.UtcNow);
    }
}
```