Cases - VI Company



Made by: Oscar Veldman & Jeffrey Hoogland

Table of Contents

[Case 1- Meertaligheid 3](#_Toc461713463)

[Scoop 3](#_Toc461713464)

[Puntsgewijs 3](#_Toc461713465)

[Wat kan het niet 3](#_Toc461713466)

[Software 3](#_Toc461713467)

[Login rollen 3](#_Toc461713468)

[User stories 4](#_Toc461713469)

[Case 2 – Identity Provider 5](#_Toc461713470)

[Case 3 – Kleinere projecten 6](#_Toc461713471)

# Case 1- Meertaligheid

## Waarom?

De programmeurs van VI vinden soms lastig om alle labels in verschillende talen te vertalen. Als je bij de ene taal toevoegt, dan moet je dat bij de andere ook doen. Soms wordt heel onoverzichtelijk. Daarom is er een programma nodig die het allemaal overzichtelijk houdt en alles makkelijk kan aanpassen.

## Scoop

De software moet makkelijk labels kunnen veranderen voor meerdere talen. Als je bij de ene taal een label toevoegt, dan herkent het programma gelijk dat je bij de andere talen ook moet vertalen. Deze worden veranderd in productie en op de lokale machines. Dit geld ook voor het verwijderen van labels. Deze labels moeten ook steeds opnieuw gesorteerd worden. Deze worden ingelezen en weggeschrijven in json. Alleen de bevoegde personen mogen erbij kunnen overal. De bevoegde personen kunnen klanten en werknemers van VI-Company.

Het programma moet in 1 opslag duidelijk zijn voor de gebruiker. Alle labels die kunnen gevonden door een goed werkende zoekbalk bijvoorbeeld. De labels moeten op meerdere manieren gevonden worden, bijvoorbeeld op naam of op route.

### Puntsgewijs

* Toevoegen van labels
* Vertaling veranderen van labels
* Verwijderen van labels
* Sorteren van labels
* Inlezen van json
* Wegschrijven van json
* Mark down
* Routes makkelijk vertalen
* Zoekbalk
* Duidelijk hoe het werkt.
  + Na 1 of 2 keer gebruiken
* Runtime inlezen

## Wat kan het niet

## Software

Hieronder staan meerdere mogelijkheden voor het maken van software

* Een webapplicatie (ASP.NET Core)
  + Json
* Een webapplicatie (ASP.NET Core)
  + Database
* \*.Exe voor Windows. (Programma local)

## Login rollen

* Iemand die alles mag aanpassen
* Iemand die alleen mag kijken?

## User stories/Requirements

Hieronder komen al een paar voorbeelden van user stories. Dit worden er later meer.

1. Een gebruiker kan een label kunnen toevoegen, omdat het label onthouden moet worden
2. Een gebruiker kan een label kunnen verwijderen, omdat het label niet meer nodig is
3. Een gebruiker kan een label vertalen, omdat je meerdere vertalingen hebt
4. Een gebruiker kan een label naam veranderen, omdat hij met een betere naam komt.
5. Een gebruiker kan een module aanmaken, zodat het netjes gesorteerd blijft.
6. Een gebruiker kan een route vertalen, omdat je meerdere vertalingen hebt
7. Een gebruiker kan makkelijk zien welke labels nog niet vertaald zijn, omdat dat dan overzichtelijk blijft.
8. Een gebruiker kan een label veranderen on run time, omdat dat veel minder tijd kost voor het opnieuw deployen
9. Een gebruiker kan inloggen, zodat hij kan laten zien wie hij is
10. Een gebruiker kan uitloggen, zodat niet iemand anders zomaar er in kan
11. Een gebruiker kan labels opzoeken, zodat hij snel bij het label kan zijn.
12. Een gebruiker kan een nieuwe taal invoegen, omdat er meer talen bij kunnen komen
13. De gebruiker kan switchen van taal, zodat hij bij de talen kan om ze te vertalen
14. Een gebruiker kan een nieuw onderwerp van labels toevoegen, omdat het overzichtelijk en snel blijft voor het programma.
15. De administrator kan nieuwe accounts aanmaken, omdat er meerdere mensen kunnen inloggen
16. De gebruiker kan makkelijk zien welke labels niet meer in de site worden gebruikt, zodat de gebruiker makkelijk kan beslissen of hij het niet meer nodig heeft
17. De gebruiker kan de labels sorteren op verschillende tabelkolommen, zodat de gebruiker het makkelijker kan opzoeken

# Case 2 – Identity Provider

## Waarom?

De developers bij VI Company moeten bij elk project die zij maken een login systeem maken en daarbij moeten zij ook vaak rollen aanmaken. Omdat dit vaak hetzelfde is bij elk project is dit zonde om dit telkens weer opnieuw te ontwikkelen. Als het een standaard package zou worden zou dit veel tijd schelen voor de developers.

## Scoop

De identity provider houdt in dat er een gebruiker aangemaakt kan worden en rollen aan deze gebruiker toegevoegd kan worden. Omdat dit nu elk project handmatig gebeurd moet er een package gemaakt worden die ingeladen kan worden in elk project en hierdoor automatisch een identity provider in het project zit. Deze identity provider moet dan een registratie en inlogsysteem bevatten. Ook moet het mogelijk zijn om rollen toe te voegen en deze aan gebruikers toe te wijzen.

## Software

Er moet een package gemaakt worden van de identity provider. Het moet een package worden die door de projecten ingeladen kan worden.

## User stories

1. De gebruiker kan zich registreren
2. Het systeem stuurt een activatiemail naar de gebruiker.
3. De gebruiker kan zich inloggen
4. Het systeem stuurt een link naar de gebruiker, als de gebruiker op de knop wachtwoord vergeten drukt.
5. De administrator kan gebruikers aanmaken
6. De administrator kan gebruikers wijzigen
7. De administrator kan gebruikers verwijderen
8. De administrator kan rollen aanmaken
9. De administrator kan gebruikers rollen toewijzen
10. De administrator kan gebruikers rollen afnemen
11. Het systeem moet real-time valideren als de gebruiker zich wilt registreren.

# Case 3 – Kleinere projecten

* Charts
* Prijs informatie
  + Binnenhalen en weergave quotes
* Validatie
* E-mail versturen
* Bestanden importeren
* Genereren van bestanden
* Formatting
* Security
* Caching
* SignalR
* Unittesten in een packets
* Tabellen en sorteren daarvan
* Contact formulieren
* Nui-Company