IP Major opdracht 19-20

Tom Eversdijk & Wannes Fransen

March 2, 2020

1 Inleiding

- Deadline: **Woensdag 20 mei 2020, 08u00 CEST**.
- Je werkt individueel aan de volledige opgave.
- Er wordt gebruik gemaakt van een github classroom. Accepteer hiervoor onderstaande uitnodigingslink.
  `https://classroom.github.com/a/3tdEhD5W`
- Tijdens het examen zul je een extra functionaliteit moeten toevoegen aan je applicatie, zorg er dus voor dat je applicatie eenvoudig uitgebreid kan worden en dat je perfect weet hoe je applicatie werkt. Dit komt bovendien ook van pas tijdens de mondelinge verdediging.
- Dit project telt voor 30% mee voor je uiteindelijke score van dit vak. Deze 30% geldt enkel voor het project, de uitbereiding die tijdens het examen wordt toegevoegd is onderdeel van de 70% van het examen.
- Omdat het project een onderdeel is voor de evaluatie van dit vak, is het examenreglement hierop van toepassing. Schrijf dus je eigen oplossing! Wanneer je hulp krijgt of geeft aan een mede-student, zorg er dan voor dat het altijd over algemeen advies gaat en deel geen code. Vermeld ook altijd je bron. Dit geldt zowel voor contact in persoon maar ook voor advies via online forums.
- Heb je problemen met je project kun je altijd naar het monitoraat komen. Daarnaast is het ook mogelijk om je vraag op het Toledo forum te plaatsen. Mails met vragen zullen onbeantwoord blijven.

2 Opgave

De opgave is om een applicatie te ontwikkelen waar dat gebruikers informatie kunnen bijhouden welke honden en katten ze zijn tegen gekomen. We willen onthouden wat de naam van het dier was, weten of het een kat of een hond is, wanneer het is geboren en wie dat het baasje is. Daarnaast willen we dat

een third party application gebruik kan maken van onze API om een overzicht te krijgen welke honden en katten we zijn tegengekomen, nieuwe honden en katten kan toevoegen, reeds gekende honden en katten kan wijzigen en zelfs kan verwijderen. Bovendien moet er een onderscheid gemaakt worden tussen een standaard gebruiker die zich aanmeld en een admin. Een admin moet naast de functionaliteiten die een standaard gebruiker heeft, een extra functionaliteit hebben om gebruikers te verwijderen en om het wachtwoord van een gebruiker aan te passen. Tot slot wordt er verwacht dat er tests geschreven worden. De tests mogen beperkt blijven tot de API werking.

3 API

Met de API is het de bedoeling dat een third party application honden en katten kan toevoegen, reeds gekende honden en katten kan wijzigen en bestaande honden en katten kan verwijderen **die van jou zijn**. Om een van deze acties uit te kunnen voeren moet in de https requests header een geldige API key meegegeven worden. Bovendien is het de bedoeling om voor iedere third party application een nieuwe API key mee te geven. Zo kunnen we eenvoudig de toegang van 1 third party application blokkeren door het intrekken van overeenkomstige API key.

3.1 Create

Voorzie een API endpoint om door middel van een POST request een dier toe te voegen.

3.2 Read (overview)

Voorzie een API endpoint om door middel van een GET request alle dieren (beperkt) te tonen. **LET OP: Dit zijn enkel jouw dieren** Voorbeeld output zou zijn:

```
[
    "1" : {"name" : "NAME", "dob" : "01/01/1994"},
    "2" : {"name" : "NAME", "dob" : "01/01/1994"}
]
```

3.3 Read (animal details)

Voorzie een API endpoint om door middel van een GET request info over een specifiek dier op te vragen. Voorbeeld output zou zijn:

```
{
    "id" : 1,
    "name" => "NAME",
    "dob" : "01/01/1994"
}
```

3.4 Update

Voorzie een API endpoint om door middel van een PUT of PATCH request een dier up te daten op basis van ID. Dit kan een response geven van 200, 400 met de errors van de verkeerde velden TODO: VERIFY

3.5 Delete

Voorzie een API endpoint om door middel van een DELETE request een dier te verwijderen op basis van ID.

3.6 EXTRA

Je kan enkel je eigen dieren beheren of toegang verlenen aan andere gebruikers door middel van API keys. Bij het binnenkomen v.e. request, ga je nakijken of de meegegeven API key geldig is voor de gebruiker van wie je dieren wilt aanpassen/lezen.

4 Pagina's

Om de opgave tot een goed einde te brengen is het nodig om verschillende pagina's met hun eigen functionaliteit te ontwerpen. In deze sectie wordt er dieper ingegaan op de verwachtingen van iedere pagina.

4.1 Header

Met uitzondering van de login pagina [4.2] en de raw-api [4.7] pagina wordt er verwacht dat iedere volgende pagina gebruik maakt van eenzelfde header. Deze header wordt gebruikt door de gebruiker om zichzelf te kunnen navigeren tussen de verschillende paginas. Meer concreet moet de header de volgende functionaliteiten bevatten: Veranderen van taal (minstens 2 verschillende talen), uitloggen uit de applicatie, navigeren naar de show profile [4.4] pagina en terug navigeren naar de home pagina [4.3]. Voor een admin gebruiker moet het ook mogelijk zijn om naar de user management pagina [4.8] te gaan.

4.2 Login pagina

Zoals de naam als doet vermoeden is dit de pagina waar dat een gebruiker kan inloggen. Daarnaast kan een nieuwe gebruiker op deze pagina zijn account aanmaken. Zowel het inloggen als het aanmaken van een nieuwe account gebeurd op basis van een gebruikersnaam & wachtwoord dat door de gebruiker zelf gekozen kan worden.

4.3 Dashboard

Het dashboard is de pagina waar dat de gebruik op terecht zal komen nadat deze zichzelf heeft aangemeld. Hier wordt een overzicht gegeven van al de honden en katten die reeds gekend zijn. Daarnaast moet de gebruiker door middel van javascript kunnen zien hoe laat het op dit moment is.

*Via de browser is het **nergens** mogelijk om katten / honden te kunnen aanmaken /aanpassen / verwijderen, **wel via REST.***

4.4 Show profile

Op deze pagina wordt er verwacht dat een gebruiker zijn eigen gebruikersnaam kan zien. Naast het zien van de eigen gebruikersnaam bevat deze pagina een knop om verder te gaan naar de edit profile[4.5] pagina en een knop om de gebruiker zijn wachtwoord te laten veranderen[4.6].

Daarnaast wordt er verwacht dat de gebruiker op deze pagina al zijn API-keys met de bijhorende naam kan zien [4.7], verwijderen en nieuwe API-keys kan genereren met een gegeven naam.

4.5 Edit username

De enige functionaliteit van deze pagina bestaat eruit dat een gebruiker zijn gebruikersnaam kan wijzigen. Zodra de gebruikersnaam is gewijzigd moet de gebruiker in staat zijn om met zijn gewijzigde gebruikersnaam in te loggen op de applicatie.

4.6 Change password

Deze pagina bevat de mogelijkheid om het wachtwoord van een gebruiker aan te passen. Om veiligheidsredenen wordt er verwacht dat de gebruiker zijn vorig wachtwoord nodig heeft om een nieuw wachtwoord in te stellen. Bovendien moet het nieuw wachtwoord twee maal ingevoerd worden om de kans op typefouten te minimaliseren.

4.7 API key displayen (raw text)

Deze pagina bevat geen layout, het enige wat er op deze pagina zichtbaar moet zijn is de API key van de geselecteerde key in de show profile pagina[4.4].

4.8 User management

Deze pagina is alleen beschikbaar als een admin zich heeft aangemeld. Er wordt een overzicht gegevens van al de gebruikers. Om het eenvoudig te houden is het de admin zelf die een nieuw wachtwoord voor een gebruiker kiest.

Per opgelijste user, zie je de knoppen

- Edit
- Delete

4.8.1 Edit user page

Op deze pagina kan een admin de gebruikersnaam & wachtwoord van een gebruiker wijzigen. Er is een save button en een cancel button om terug naar het overzicht te gaan.

5 Images

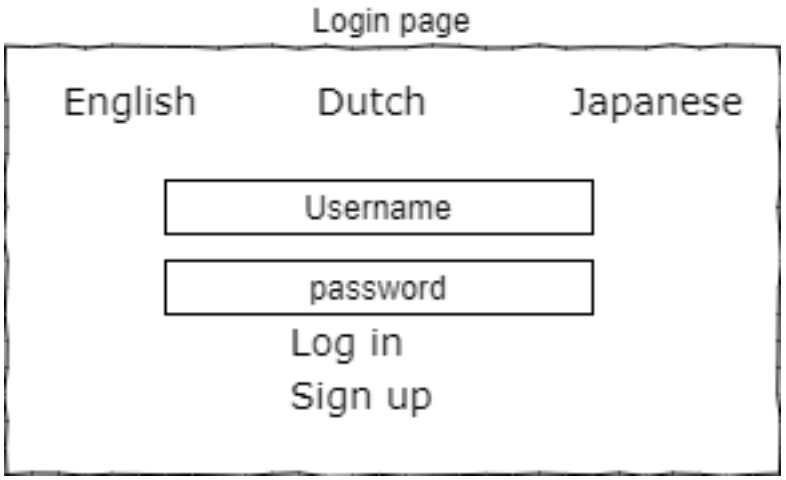

Figure 1: Login page

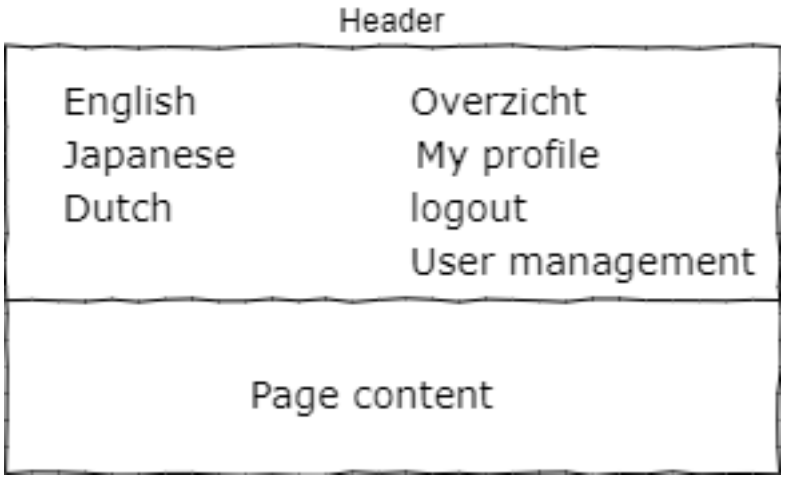

Figure 2: Header

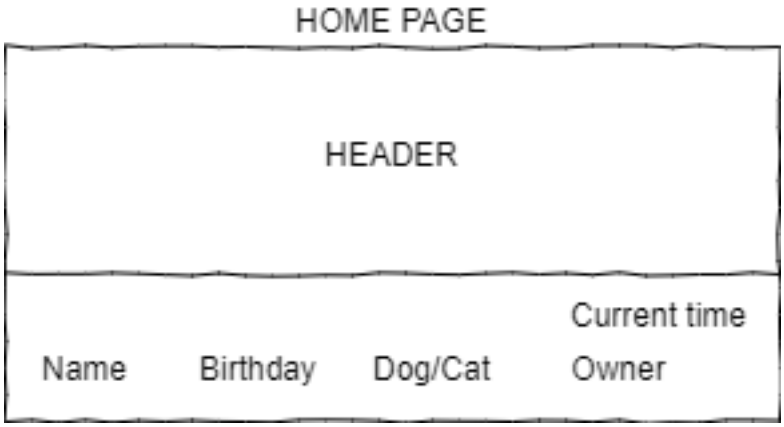

Figure 3: Home page

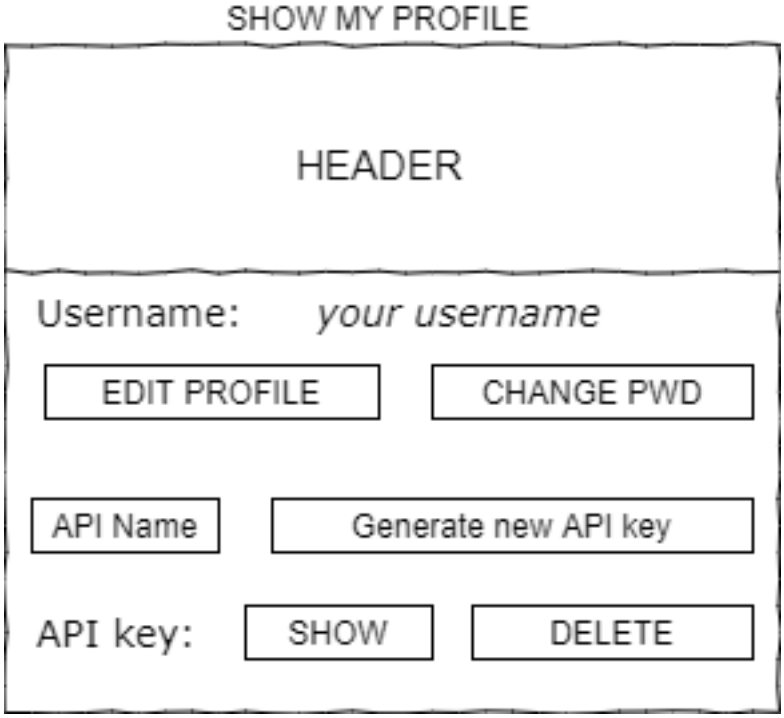

Figure 4: Show profile

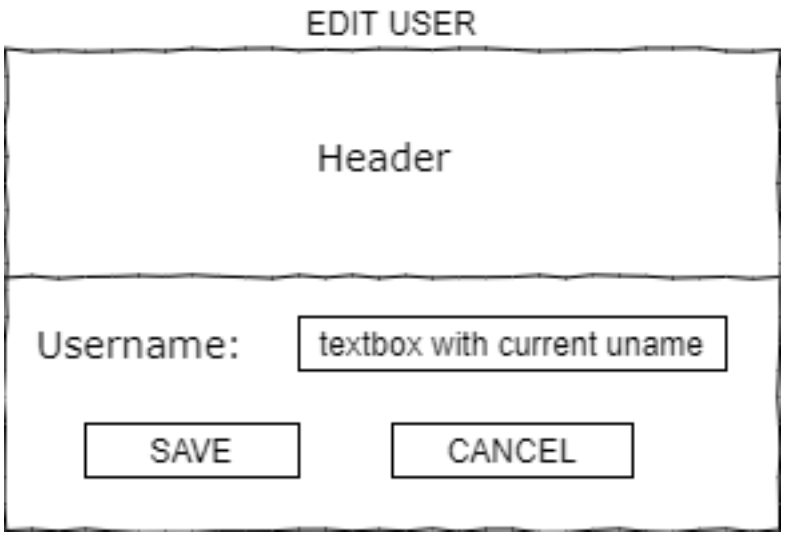

Figure 5: edit user

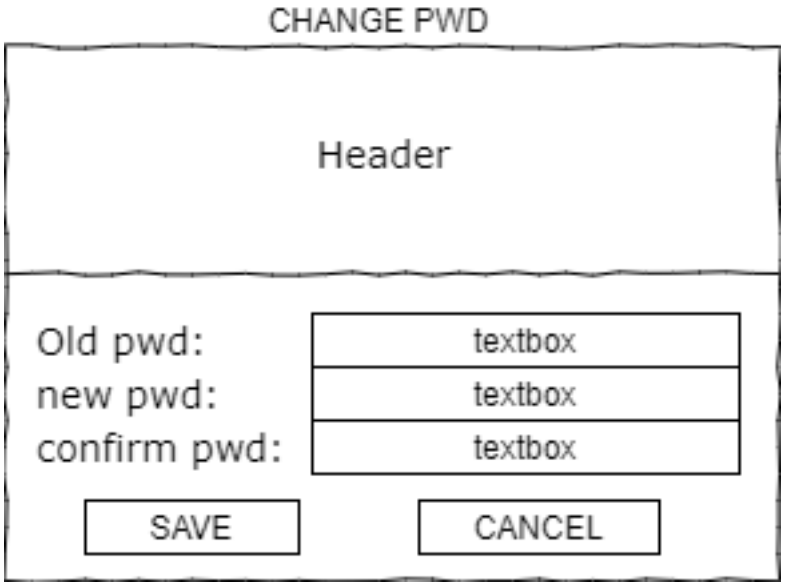

Figure 6: Change password

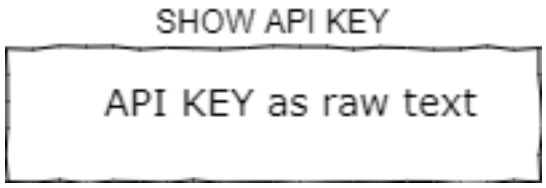

Figure 7: Raw api

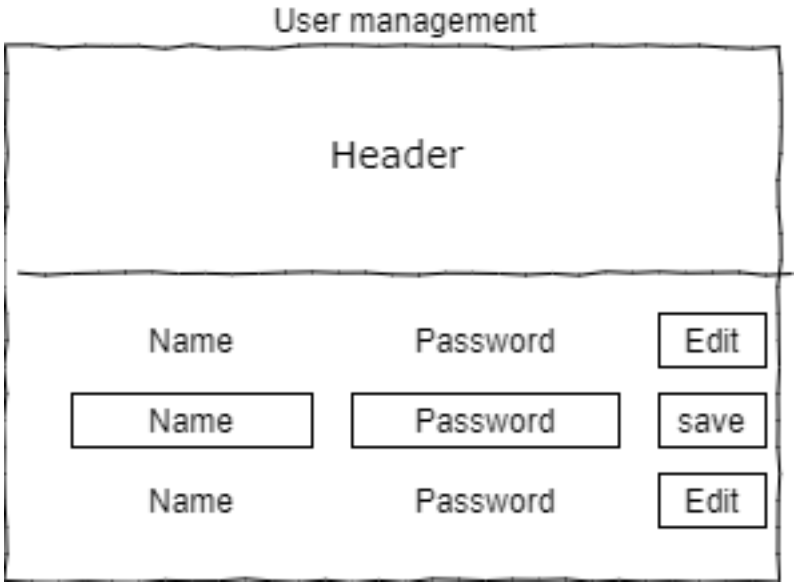

Figure 8: user management

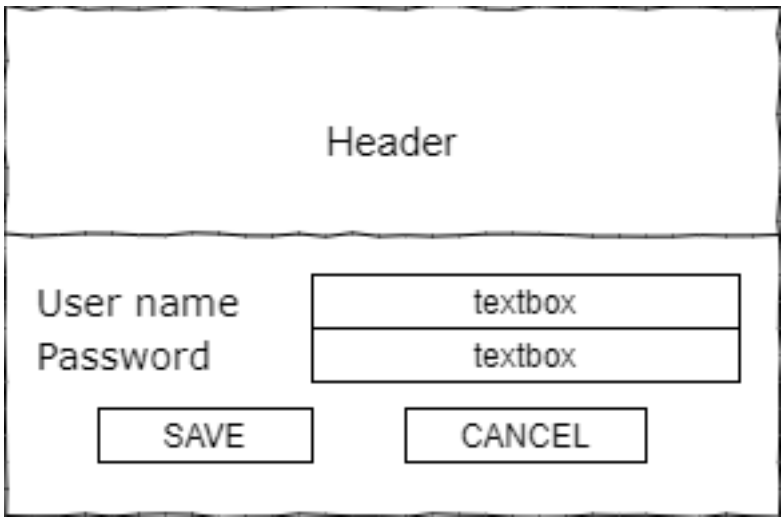

Figure 9: User edit as admin