Carpooling

Projectopdracht Tweede zit (augustus 2018)

Integratieproject 2

IAO en ISM

2017-2018

Inleiding

Het toenemende verkeersinfarct op onze wegen heeft verregaande ecologische en economische gevolgen. De in dit document beschreven applicatie, tracht carpooling zo sterk mogelijk te faciliteren met behulp van moderne ICT technologie.

Functionele vereisten

De applicatie laat carpoolers toe om via een web browser:

- met elkaar in contact te komen door routes aan te bieden en hier naar op zoek te gaan
- concrete afspraken te maken over samen rijden

Dit vertaalt zich in volgende features:

1. **Registreren**. Om info te bekijken hoeft men niet aangemeld te zijn. Om info toe te voegen aan het platform dient men zich te registreren met username (=e-mail adres) en paswoord.

2. Beheer van een persoonlijk **profiel** (naam, adres, leeftijd, man/vrouw, roker, wagens (type, verbruik, #passagiers)).

3. Beheer[1] van een of meerdere ***routes*** met per route minimaal volgende gegevens:
   - De route definitie (vertrek, bestemming, heen- en terug of enkele rit, passagepunten[2]) via een integratie met (google) maps (aanduiden van locaties op kaart)
   - Type wagen (uit het profiel).
   - Tijdstip van vertrek voor heen- en eventueel terugreis
   - Aantal personen die kunnen meerijden

4. **Zoeken** naar andere carpoolers op basis van minimaal volgende criteria
   - Vertrekpunt en bestemming met opgave van radius. Deze zoekactie houdt ook rekening met passagepunten in de doorzochte routes.
   - Tijdstip van vertrek (met een instelbare tijdsmarge)
   - Profielkenmerken (man, vrouw, roker,…)

5. **Communicatie** tussen carpoolers door het versturen/accepteren van een aanvraag tot het samen rijden van een bepaald traject. In de aanvraagform kan het pickup en -dropoff punt worden aangegeven en is een vrij tekstveld ter beschikking voor het maken van verdere afspraken.

[1] Beheer betekent zowel het aanmaken, bewerken, verwijderen als oplijsten (CRUD)
[2] Dit zijn punten waar iemand passeert en die handig zijn om mensen op te pikken of af te zetten (bv. carpool parkings)

Opmerking
Het systeem dient ondersteuning te bieden voor afspraken tussen meerdere carpoolers (>2) om samen te rijden, waarbij bv. bestuurder a persoon b op plaats x en persoon c op plaats y oppikt (en ook op verschillende plaatsen afzet).

Algemene opmerkingen

- Er hoeft geen admin interface (bv. voor het beheren van user accounts) voorzien te worden
- E-mail activatie van het account of het resetten van het password hoeft niet voorzien te worden
- Bijkomende features zijn uiteraard altijd toegestaan (ook de bovenstaande)

Niet-functionele vereisten

Technologieën

De applicatie dient gebouwd te worden met behulp van volgende technologieën

Back-end	**Spring Boot** Spring framework met database mapping (bv. Hibernate) Database naar keuze
Web client	**HTML-CSS-JS** Javascript SPA framework (Angular[3] / React of Vue) eventueel in combinatie met MPA (bv. Thymeleaf)

Security

Alle onderdelen van de applicatie (clients, back-end, databases,....) worden beveiligd met behulp van de standaard beveiligingsmechanismen van de gebruikte technologieën.

Gebruiksvriendelijkheid

De toepassingen dienen zonder vorm van gebruikersdocumentatie of opleiding gebruikt te kunnen worden. De interface moet dus bijzonder intuïtief te zijn. Alle functionaliteiten moeten zichzelf uitwijzen en indien nodig van een korte info tekst en (eenmalig verschijnende) gebruik hints te worden voorzien.

De usability dient expliciet getest te worden. De client heeft een responsive lay-out.

[3] versie 2 of hoger

Onderhoudbaarheid

De gehele oplossing dient geschreven te worden met het oog op onderhoudbaarheid en uitbreidingen. Refactor de code steeds zodat er een duidelijke structuur aanwezig is die gemakkelijk te begrijpen en uit te breiden is.

Zowel back-end als front-end moeten **test driven** ontwikkeld te worden. Er worden op zijn minst op alle niveaus een aantal zinvolle testen voorzien.

Error handling en logging (zowel naar gebruiker als developer toe) dienen voorzien te worden.

Configuration management gebeurt geautomatiseerd, met behulp van de voor de verschillende platformen gangbare tools zoals Gradle, Maven, NPM,....

Alle sources en resources worden in git bewaard.

Performantie

Deze dient te voldoen aan de gangbare normen voor performantie van dynamische webapplicaties.

Documentatie

Een gebruikershandleiding dient niet voorzien te worden (zie hoger).
Alle speciale constructies (gebouwd ter verbetering van de performantie, ter omzeiling van een bug,...) dienen te worden gedocumenteerd
Verder moet er “verstandig” gedocumenteerd worden. Bijvoorbeeld: getters en setters hoeven niet gedocumenteerd te worden, en velden dienen namen te krijgen waaruit hun betekenis gemakkelijk af te leiden valt. Package en class documentatie is een noodzakelijk instrument om het design van een package en de verantwoordelijkheden van de classes in een package toe te lichten. De naamgevingsconventies van de gebruikte programmeertalen worden steeds gevolgd.