Frontend

User Interface Laag (UI)

<<interface>>
WebappFrontend

+ get_date(): date
+ get_location(): latitude, longitude

<<interface>>
UserLocation

+ get_geolocation(current_position): tuple

<<interface>>
CookieManager

+ check_cookies(key): boolean
+ get_cookies(key): array
- set_cookies(key, value)

CookieManager is een interface voor de CookieManager Class. Deze class implementeert de functionaliteit om cookies op het lokale systeem uit te lezen en aan te passen. Code moet in de frontend mee worden gegeven, omdat daar in/via de browser van het lokale systeem de cookies opgeslagen staan.

WebappFrontend interface is niet per se 1 object (zoals een class o.i.d.), maar een verzameling van objecten (wss functies) die worden gebruikt door de frontend.
Voorbeeld hiervan is het ophalen van de datum of een implementatie van een class met complexere functionaliteit. De functie van de implementatie wordt dan wss iets als:

```
[js]
function get_user_location( ) {
    let user_location = new UserLocation();
    return user_location.get_geolocation(current_position);
}
```

Backend

Applicatie Logica Laag (AL)

<<interface>>
AdviceGenerator

+ get_personalized_advice(KS): array

<<interface>>
DataCollector

+ get_weather_report(date): array
- get_advice_data(gevoels_temp): array

<<interface>>
UserDataManager

- get_user_info(id): array
- set_user_info(id, new_user_data)
- register_user(new_user_data)

AdviceGenerator is voor alle functionaliteit m.b.t. de hoofdlijnen in het genereren van een advies. Specifiekere/complexere functionaliteit wordt uitbesteed aan geïsoleerde objecten (bijv: UserDataManager).

UserDataManager is een object voor het uitlezen/wegschrijven van/naar de database van data gerelateerd aan gebruikers.

DataCollector is een object of verzameling van functies die allemaal betrekking hebben op het ophalen van data van algemene informatiebronnen. Denk data van APIs en de database. Het uitlezen van gebruikers informatie en andere gevoelinge/beschermde data worden opgehaald door daarvoor besteemde objecten (zoals UserDataManager). DataCollector kan daarentegen wel (een deel van) de implementatie van deze objecten zijn.

DatabaseAccess en APIAccess vervullen theoretisch gezien beide dezelfde functionaliteiten, alleen met betrekking tot een ander bron. Deze andere bron maakt dat het beide om een verschillende implementatie vraagt.

Data Access Laag (DA)

<<interface>>
DatabaseAccess

+ get_data(query): array
+ set_data(data)

<<interface>>
APIAccess

+ get_data(date): array
- build_request(request_params): object

Database Laag (DB)

<u>MongoDB Database</u>

<u>Users</u>

<u>Advices</u>

MongoDB Database is in dit model geen object, maar presenteert de opdeling van de database.

Het idee is de database op te delen in twee hoofdsoorten data die opgeslagen gaan worden:
- Users, voor gebruikersinformatie
- Adivces, voor alle informatie/data m.b.t. de adviezen die gegeven worden.