Jogging Coach

**A brief description of the system(goal, used sensordata)**

Met de jogging coach kan de gebruiker meten hoeveel afstand hij heeft afgelegd van punt A naar B. De gebruiker stelt een target in en kan beginnen met joggen. De coach helpt de gebruiker met feedback. Keep running!!! Als de gebruiker halverwege stopt. De gebruiker kan ook breaks inplannen binnen de target. Voorbeeld: de gebruiker stelt 5 KM in en op 2,5 KM een break. Als de gebruiker op 1 KM stopt dan moedigt de coach de gebruiker aan om door te gaan. “Still 4 KM to GO”.

Vooruitgang wordt in een database opgeslagen. En de prestaties van de gebruikers worden met elkaar vergeleken. Eventueel wordt er bijgehouden hoe vaak de gebruiker stopt.

**A desciption of how your system will achieve this goal**

Ons doel is om met de applicaties de joggers te ondersteunen door de joggers de mogelijkheid te bieden de jog sessie in te plannen. De jogger kan instellen hoeveel km hij/zij wil joggen en hoe vaak er gepauzeerd wordt. Na het joggen krijgt de jogger een status update waarin staat hoe hij/zij heeft gepresteerd en hoe het beter kan.

We gaan een android applicatie maken en een Java servlet voor het afhandelen van sensor data. De jogging app gebruikt sensor data van de accelerometer. De data wordt opgeslagen in een database met behulp van de servlet.

**What services will be exposed to the outside world [REST API]**

Wij gaan de REST (REpresentational State Transfer) API gebruiken.

Informatie en data in REST wordt geadresseerd via een URI(Uniform Resource Identifier).

URI kan in verschillende formats gaan, elke Platform heeft een andere format nodig.

In onze geval voor onze android applicatie gaan wij REST gebruiken in combinatie met JSON.

Om een RESTful web services te schrijven is kennis over HTTP van belang.

De REST API bevat een methode GET, GET(specified), POST, PUT, DELETE , HEAD, OPTIONS en de Response message van de server bijna hetzelfde als die van http.

Voor een werkende RESTful services heb je nodig een Class met getters en setters.

**What format the app will use to communicate with the backend**

De app gaat HTTP gebruiken om met de backend te communiceren. Android heeft verschillende libraries waaronder de standaard HttpClient, retrofit en volly. Voor de backend gebriuken wij JAX-RS service implementeren zodat de remote client kan communiceren met de database. Voor JAX-RS kunnen we componenten modellen gebruiken zoals JavaBeans(EJB), Web Beans, JBoss Seam en Spring. De JAX-RS services kan gebruik maken van een singletons or per-request object:

-Bij singleton wordt er alleen één Java object services http request wordt uitgevoerd.

-Bij per-request wordt er een java object gemaakt die elke inkomende request verwerkt en wordt aan het eind wordt het weer verweiderd.

Vervolgens heeft elke Get en Post methode een notatie nodig om te binden met een HTTP Post en Get operaties.

**UserCase**

-Use cases for the system:

Title: Joggen.

Goal: Kilometers vastleggen.

Actor: Gebruiker

Pre-conditions: Applicatie opstarten

Success scenario:

-Voert aantal kilometers in.

-jogged de aantal kilometers.

-De app zegt goed gedaan.

Post Condition:

-Start application

-voert aantal kilometers in.

-Voert de sensor data’s in de database

-Geeft feedback bij drastische verandering in de in komende snesor data.

Domein Model

Sensor ruw

X

Y

z

Ranking

Id

rank

Prestaties

Start tijd

Stop tijd

Afstand

Gebruiker

Id

Naam

leeftijd