Momentum - Multipy Active Dayz App

Architectural Design Document

Quinton Swanepoel	15245510
Azhar Patel	15052592
Tshepo Macebo Malesela	14211582
Keaton Pennels	14373018

Stakeholders

MMI Holdings: Phillip Kruger

Abstract
This document serves to describe the architecture used for the ActiveDayz application which is developed for Momentum (Multiply) as part of the Software Engineering Project (COS 301) at the University of Pretoria.
The document complies with the Architectural Design Document Software Engineering Standard, as set by the European Space Agency.

Contents

1 Introduction

1.1 Purpose

Active Dayz is part of Momentum's Multiply wellness and rewards programme. The Active Dayz programme tracks a user's steps taken and calories burned in order to provide users with relevant rewards. The programme currently only works with certain devices. An application is required that can log events such as marathons and other fitness events that users may attend, monitor users gym time, and track users steps taken and calories burned.

1.2 Scope

Active Dayz is aimed at providing momentum customers with a quick way to get rewards/active days by making use of location based services to track when a user goes to gym and how long they stay for.

The proposed system is to be integrated with a non-existing admin web interface. It should allow the integration of various services of which location based monitoring is the main objective. Other tools may include monitoring of steps take, calories burned and events attended.

1.3 Definitions and Abbreviations

1.3.1 Definitions

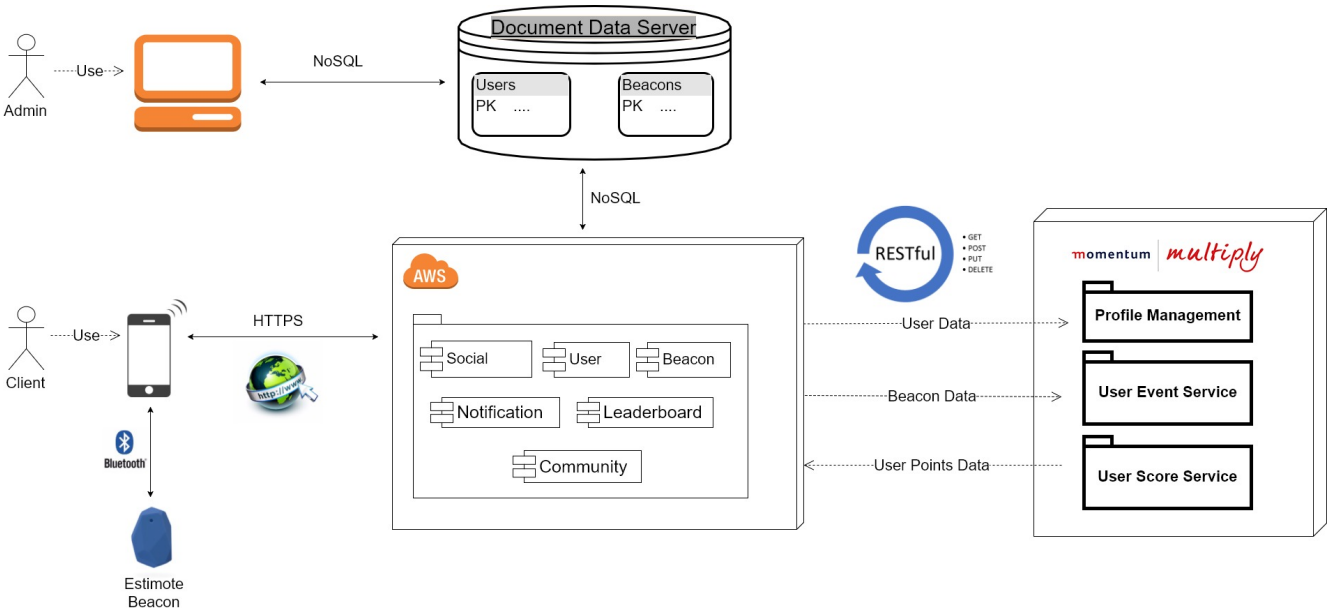
1.3.2 Abbreviations

1.4 References

1.5 Overview

2 Architectural Design

2.1 Deployment Diagram



2.2 Architectural Patterns of ActiveDayz

- Service oriented architecture patterns (Micro services)
- Layered architecture patterns
- Event-Driven patterns

The first two design patterns were chosen such that the modules within our application would have the as few as possible and also so that we could ensure we achieve low coupling. The event driven pattern will aid in the connecting and ranging of the event driven beacons

2.3 Non Functional Requirements

Listed below are the 3 non functional requirements which have been collectively identified by the us, the development team, and our client as those which are required during the development of the ActiveDayz application.

1. *Security Requirements:* This refers to the protection of our systems data and the system itself. In terms of protection of the system related data, identification and authentication of Multiply clients has been identified as a process of importance. This has been accommodated for in the Users Module. In terms of securing the system, the system will be deployed and hosted on a trusted COTS server, Amazon

Web Service Elastic Compute Cloud (EC2) and communication channels over the network will be secured using SSH

2. *Quality Requirements:*
   (a) Reliability - The system will be designed such that it will behave consistently in a user-acceptable manner when in operation. The EC2 server provides some of this capability through load balancing during periods of elevated requests for system resources.
   (b) Maintainability - The software of our application will be produced such that it requires minimal effort to update or modify components. The use of a micro service architecture will facilitate this requirement and in addition, the system will be designed to be testable through the use of a comprehensive testing structure in order to ensure accurate functionality after modification
   (c) Availability
3. *Interface Requirements* - Interfacing, and the interaction behavior of clients with the application was highlighted as an area of importance during development external entities to communicate with the system. The layout, look and feel will be designed according to the guidelines given by the client. The guidelines relate to how the above mentioned attributes have been applied in other Multiply applications/websites and are considerations which we as the design team need to heed of

[*Please note, this does not represent an exhaustive list of the non functional requirements. These are to be further updated*]

2.4 Design Requirements

- Our goal is to design a highly modular system which has minimal dependency between the modules
- The system is required to be designed such that if a module needs to be updated/modified or if a module needs to be added or removed. that this can be done with minimal effort

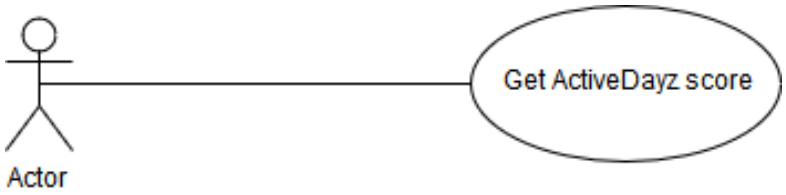
2.5 Design Constraints

The following design constraints are combination of those which were stipulated in the client's application for the call for projections and those which have been identified during meetings. These have been put it place in order to ensure seamless integration of the application into their already existing business structure

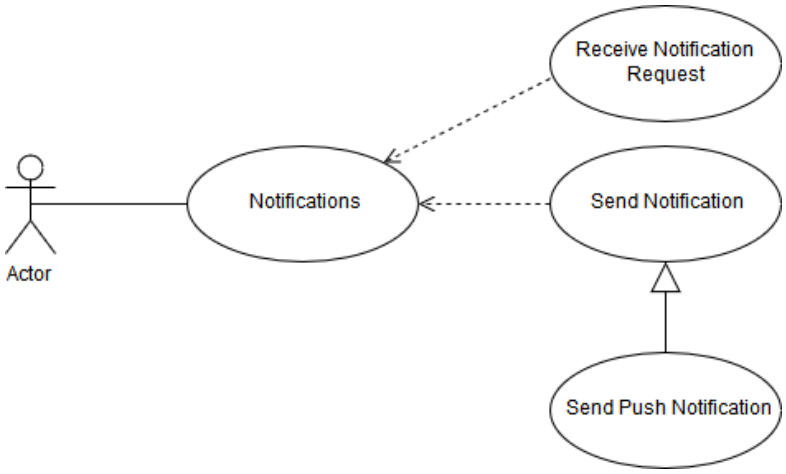
- Java EE to used for back-end development
- A non relational Database used to store data about the user (graph or document)
- Cloud Services in order to host the back-end services
- Containers in order to deploy the application
- The use of beacons in order to facilitate location based functionality

3 Component Descriptions

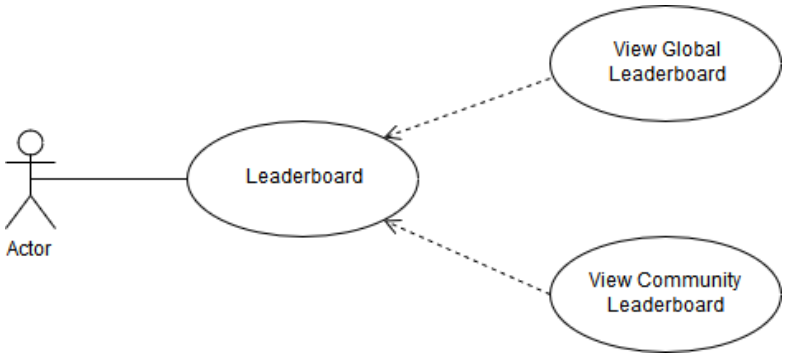
3.1 Beacon Module



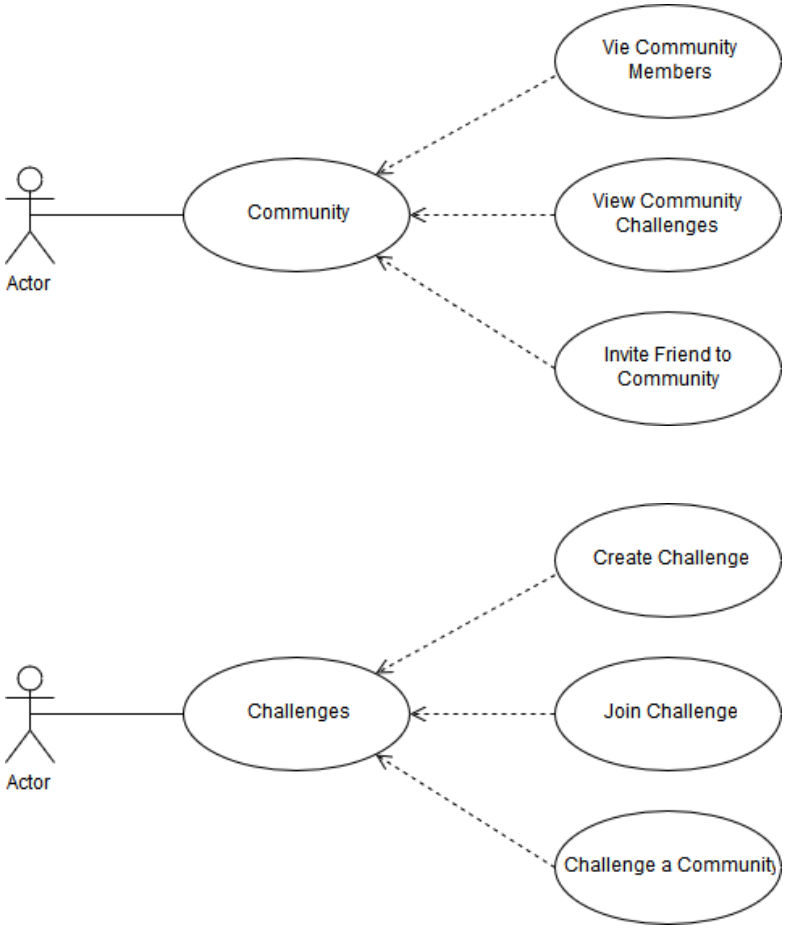
3.2 Notification Module



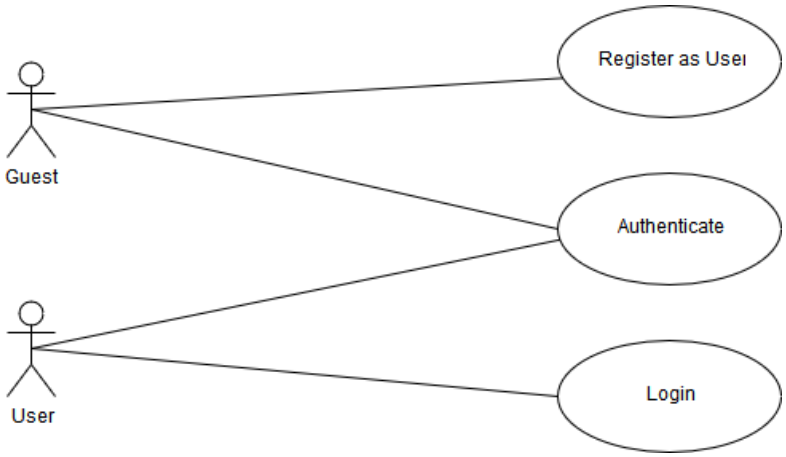
3.3 Leader Module



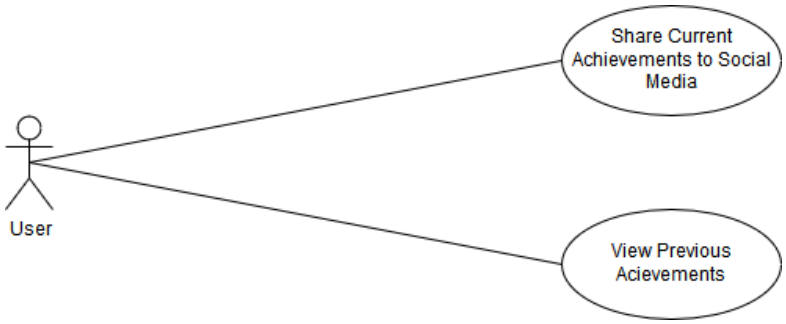
3.4 Community Module



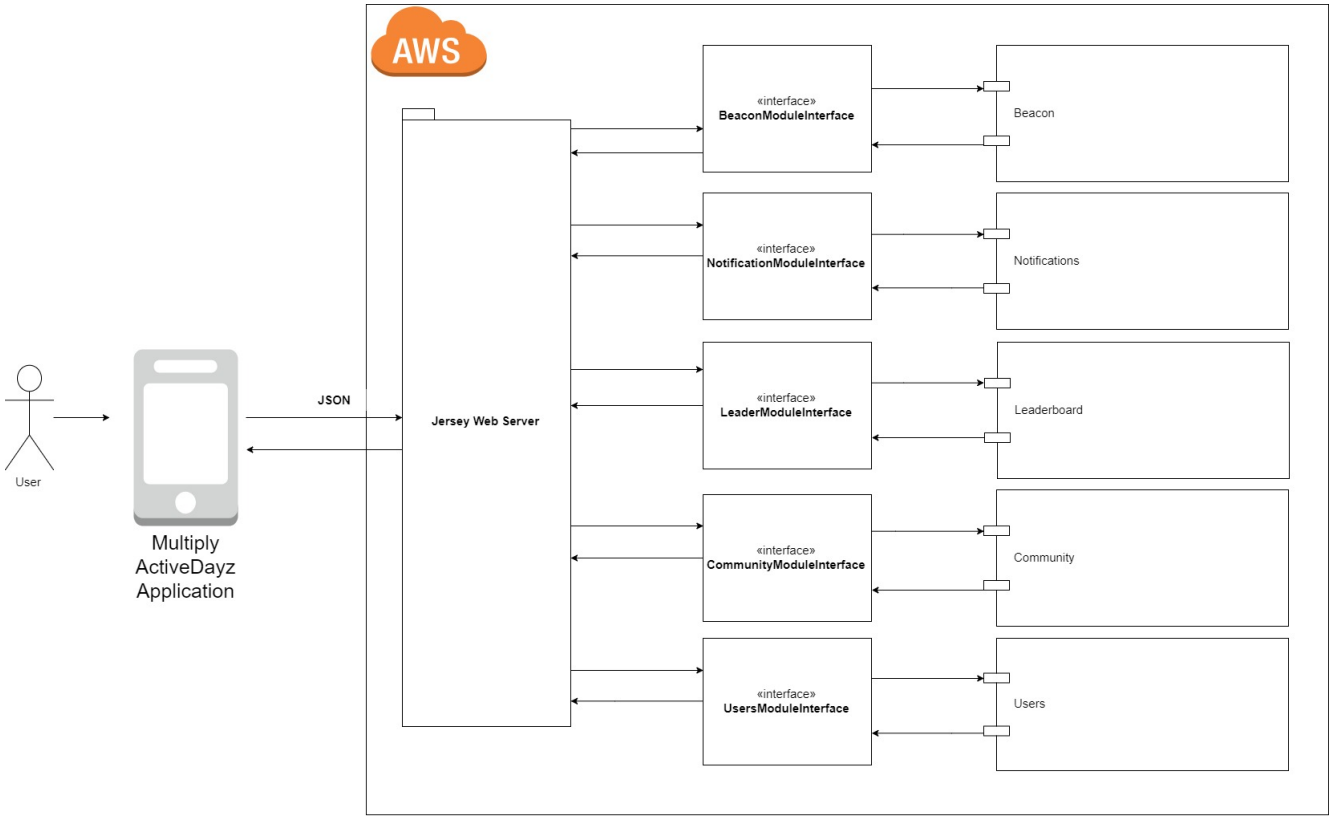
3.5 Users Module



3.6 Social Module



4 Integration Diagram



5 Feasibility and Resource Estimates

6 Technologies Used

6.1 Server

JavaEE (eclipse oxygen) will be used as the language to develop the server. The server will will implement **Jersey RESTful** web services as a means of accessing the various modules described above.
The server will be contained within a **Tomcat 8** application server and all of these components will be hosted on an **Amazon Web Service Elastic Cloud Computing** instance
Maven will be used for compilation, build and testing automation. Finally communication between modules will be structured using **JSON** in order to ensure low coupling between modules

6.2 Web develpement

polymer
node bower and polymer
js
mysql db
beacons estimotes
docker
wildfly