# Introduction

# Architectural design

## Overview

The physical components that are involved in the Data4Help service are:

* Individual’s cell phone
* Individual’s Smartwatches
* Third party’s Computer
* TrackMe system (which contains databases and server)

For the service Automated SOS, the following service must be added :

* Ambulance service

Communications between components:

* Cell phone ↔ Server (The cell phone sends requests to the server and the server answers)
* Computer ↔ Server (The computer sends requests to the server and the server answers)
* Server ↔ Database (The server makes queries and the database answers)
* Smartwatch ↔ Cell phone (The cell phone sends queries to the smartwatch and the smartwatch sends data to the cellphone via Bluetooth)
* Cell phone → Ambulance service (The cell phone sends SMS to an ambulance service)

The server must send data to the different users (individuals or third parties) only when it’s necessary. Because the users use the app through devices that can be offline, the server must only send data when asked by the users. Otherwise the user could never receive the data if his device is offline. For those reasons, an event-based system would not be adapted

We need to design a system which involves many stakeholders such as individuals, third parties, ambulance services, track systems. Moreover, in all the interactions the system is providing a service to the users so we decided to use a client-server architectural approach.

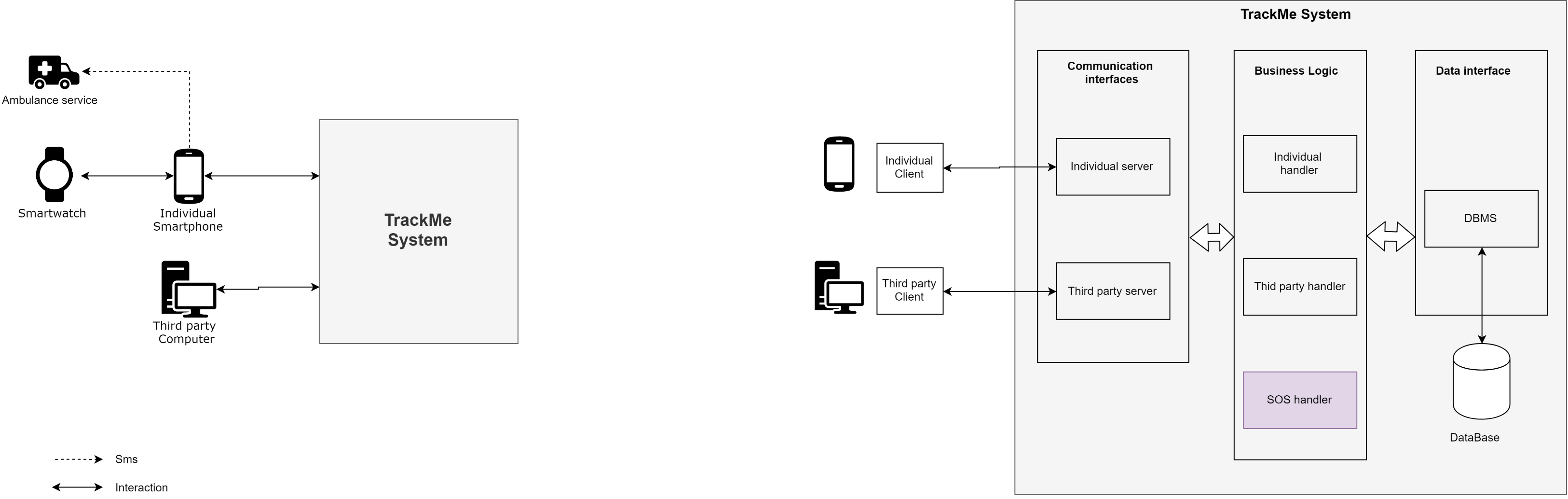


Figure 1 Overview 1

Going deeper in the analysis of the components of the TrackMe system, we are able to identify three different layers :

* **Communication Interfaces :**

This lawyer contains the interfaces components that allow the system to communicate with external agents (individual’s smartphones and third parties’ computers). As TrackMe system interacts with different external agents so it needs to have different communication Interfaces :

* A software module is needed in order to provide functionalities of the system to the individuals on their smartphone. This software module should allow individuals to send requests to the server using an API.
* A software module is needed to provide the functionalities of the system to the third parties. This module includes a website back -end and an API. The website back-end allows third parties to communicate with the server using a website ; the API allows the parties to download the data requested.
* **Business Logic :**

This lawyer focuses on the application logic of the TrackMe system. The business logic manages the individuals data, the third parties requests and their subscriptions ; for each of these functions, several software modules are necessary. Those modules will use the communication interfaces between the end user interface and the database. The business logic module receives from the communication interface orders to do specific actions; then the business logic ask to the data base interface the required data to execute these actions, and finally, returns the result to the communication interface.

* **Data Interface**

This lawyer contains all the modules that allow to store the data produced or retrieved from external resources. These modules allow interaction between the Business Logic modules and the System Databases.

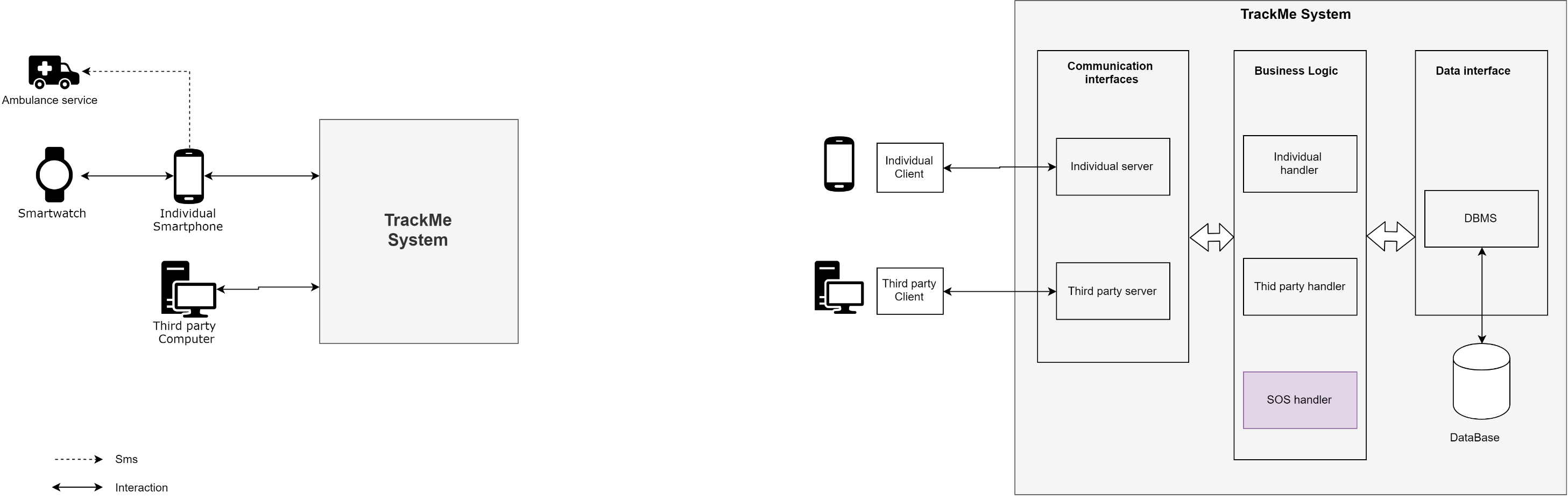


Figure 2 : Overview 2

## Component view

The following diagram, give a high-level representation of the components of the TrackMe system.

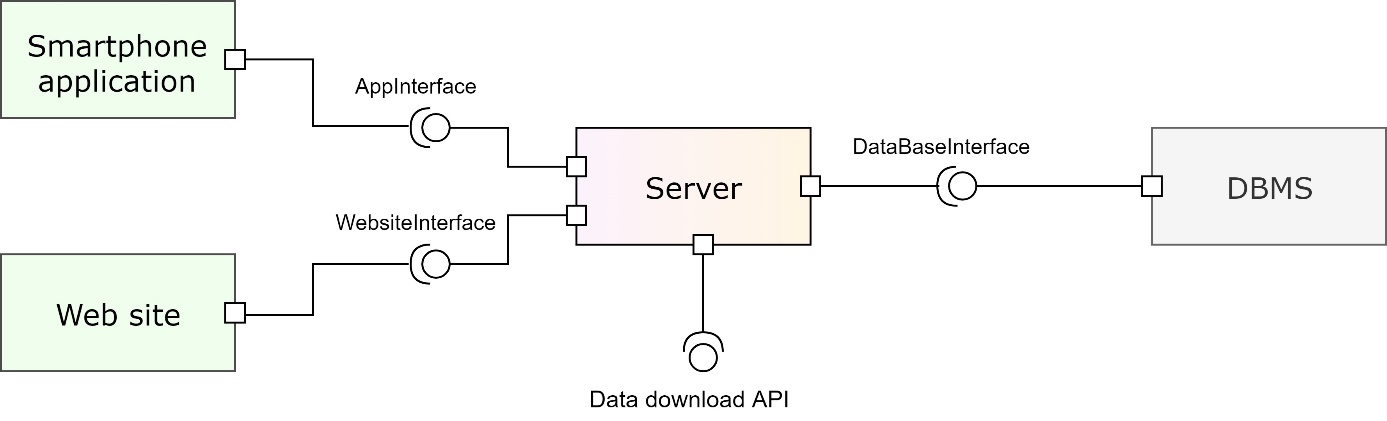


Figure 3: High level representation TrackMe system

As we can see, the system is composed of four main parts : A smartphone application, a Web site, a server and a DBMS. The smartphone application is intended for individuals and the Web site is intended for third parties. The server contains the business logic and make the connection between the others components and the DBMS.

The next component diagram contains a more low-level description of the server component :

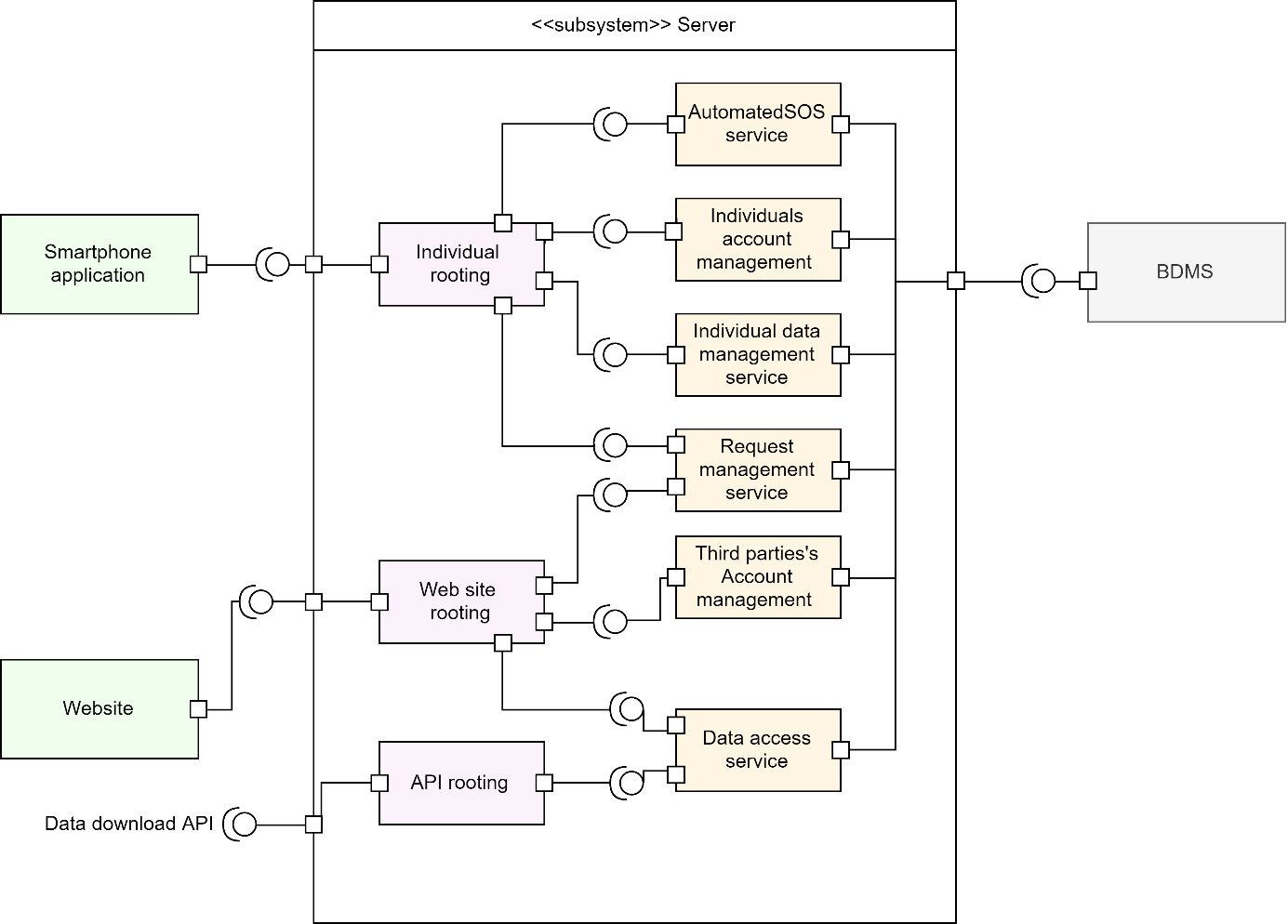


Figure 4: Low level representation trackMe system

As we can see, the server is composed of three main parts : Individual rooting, Web site rooting and API rooting. Those components determine whether a request that is received is valid or not and, if it is, to dispatch it to the relevant service component.

* **Individual rooting.**

This component manages all the functions the smartphone application has to provide to the individuals.

Individual rooting redirects the requests from the application to the corresponding component:

* *AutomatedSOS service* : manages all the functionalities of the AutomatedSOS service
* *Individuals account management* : manages the functionalities that allow individuals to register, login etc..
* *Individuals data management service :* manage the logic of the health and position data of individuals
* *Request management service :* manage the logic of the requests (for individuals : accepting or refusing request)
* **Web site rooting and API rooting.**

As explained in the RASD, the third parties can access all the functions of the Data4Help system through the Website and have also the possibility to download the data from accepted requests through an API. The API is aimed at accelerating the data acquisition process for third parties who need to acquire data regularly.

Web site rooting redirect the requests from the website to the corresponding component:

* *Third parties account management:* manages the functionalities that allow third parties to register, login etc..
* *Data access service:* manages the logic of formatting data from the Database to the third party.
* *Request management service:* manage the logic of the requests (for third parties : submitting individual or anonymized requests, subscribing to accepted requests)

The two following diagrams describe more precisely the business logic components, by showing their interactions with the data model. It is assumed that each component is able to invoke operations on the database

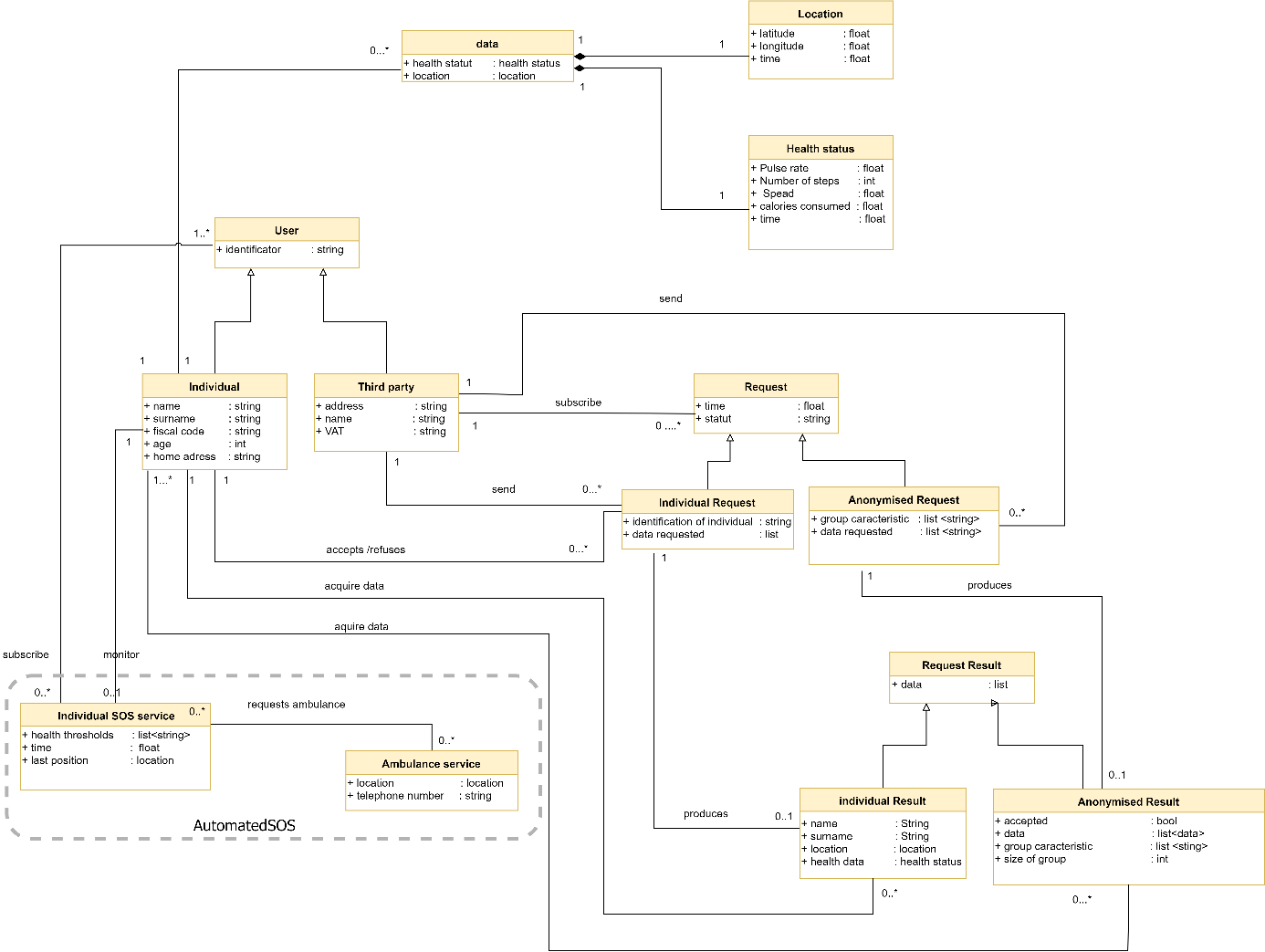


Figure 5: Class diagram of AutomatedSOS system

Some business components contain functions that requires the need to save data in the database. Those functions return a Boolean value : True if the data has been successfully saved, False otherwise. When a function is of this type, the return value is “bool\*”. The symbol \* make the distinction between functions of this type and other Boolean functions.

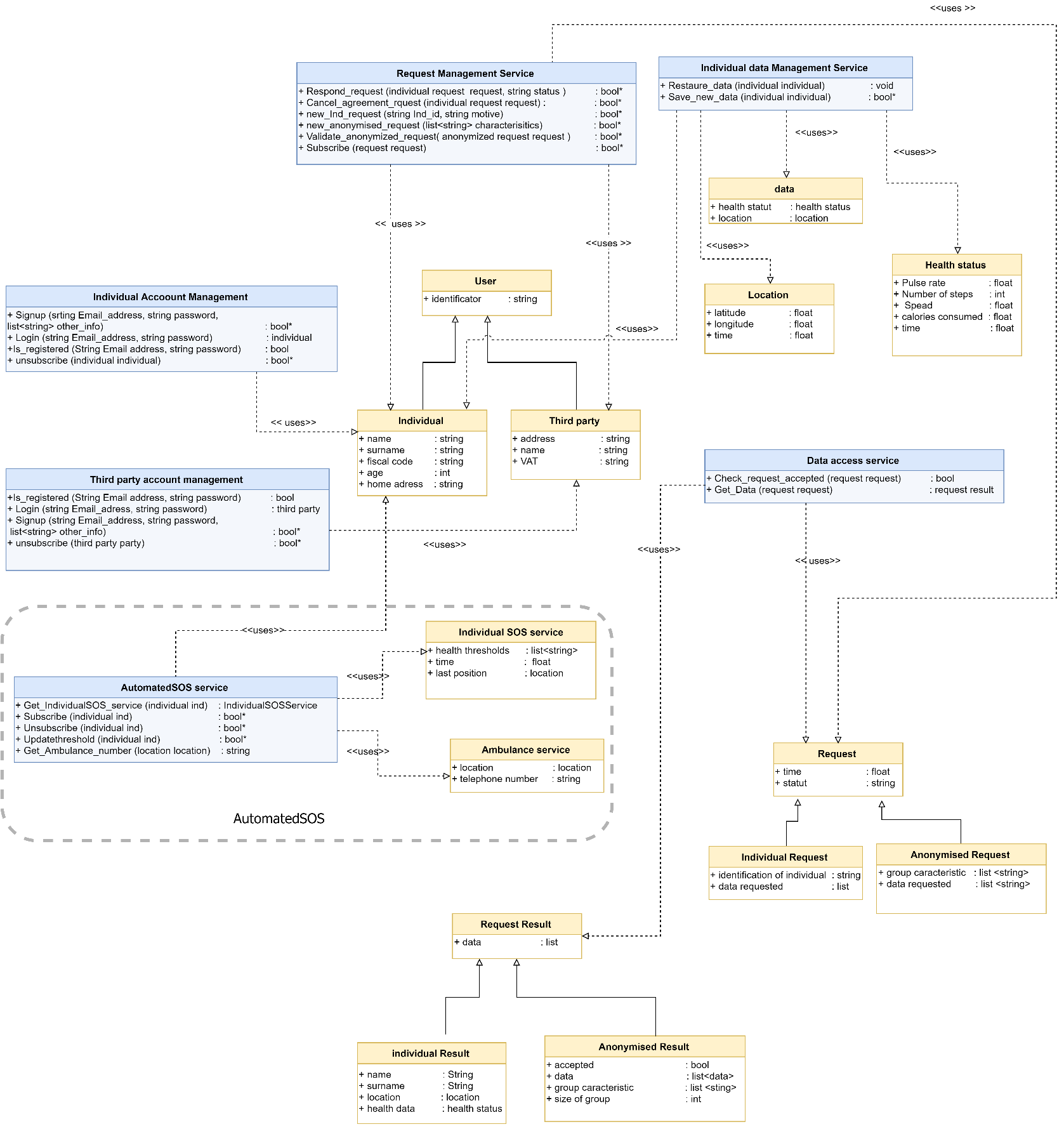
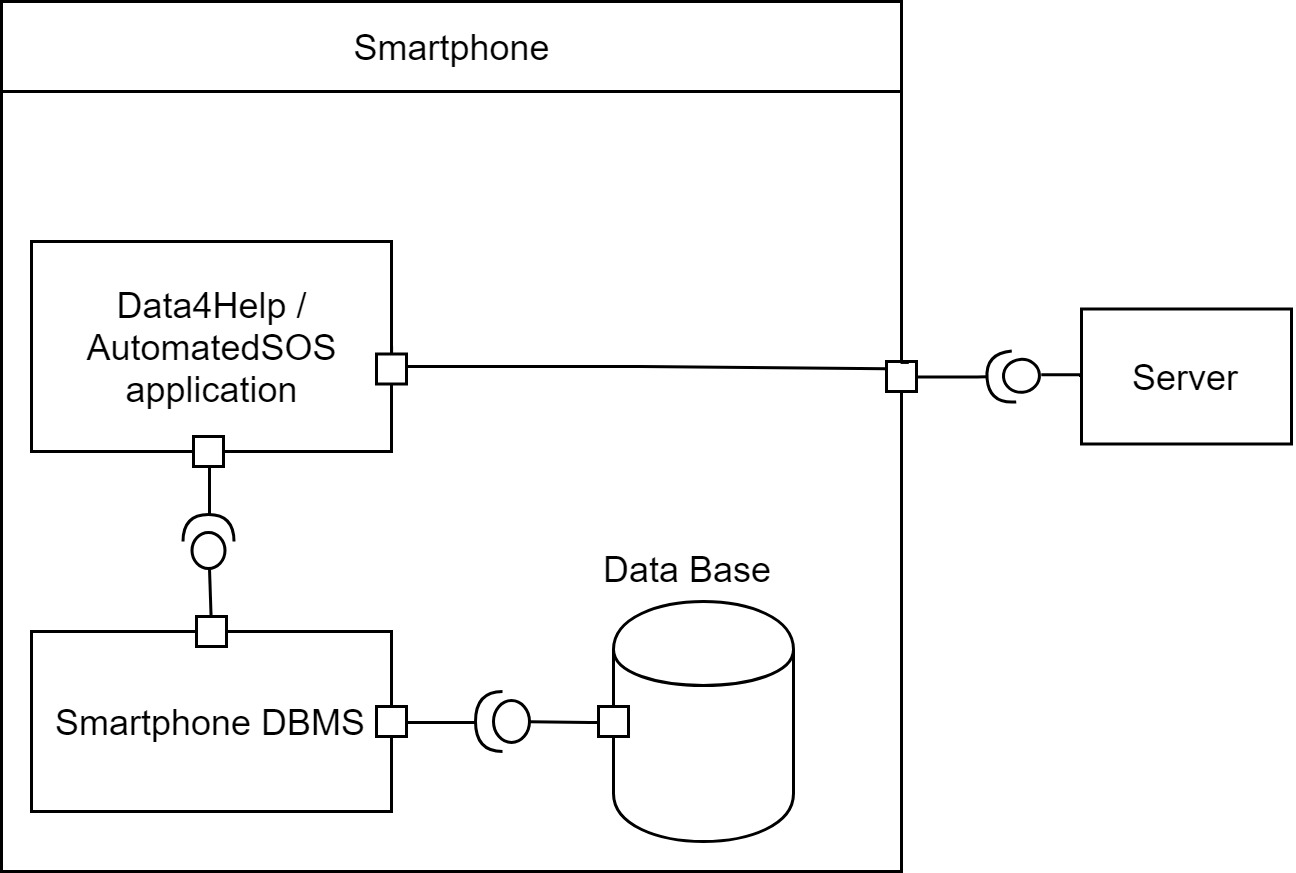
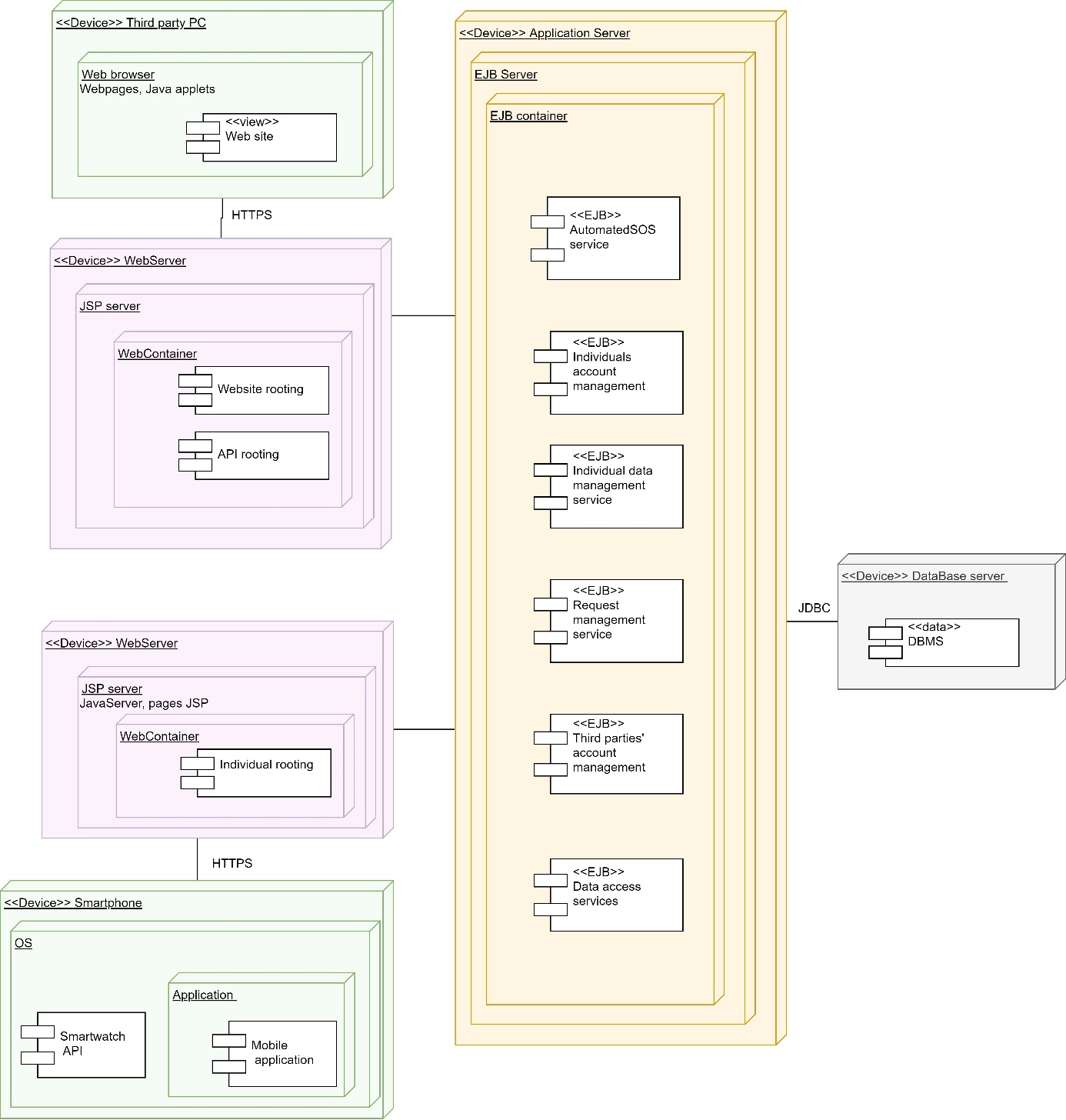


Figure 6: Description of logic components



## Deployment view



## Runtime view

## Component interfaces

## Selected architectural styles and patterns

# User interface design

# Requirements traceability

# Implementation and test plan

# Effort spent

# References