Ministry of Programming

Corona Tracking

Albin Didić
Emin Laletović
Erdem Redžepi
Dino Trnka
Dženan Šljivo
Faris Začina
Rešad Začina
Sedin Đugum

Introduction - Project Purpose

Since the outbreak of COVID-19, governments around the world have implemented a range of digital tracking measures in order to slow the spread of the Coronavirus.

This project has a purpose to implement a similar solution for Bosnia and Herzegovina in order to achieve the same goal in our country.

Objective

The objective is to provide a digital system for efficient tracking of citizens in order to carry out self-isolation measures, as advised by medical institutions. The goal is to reduce the number of new Coronavirus cases, as well as the number of deaths caused.

Success Metrics

The number of Coronavirus cases in Bosnia and Herzegovina has been steadily growing since the first recorded case on March 6, 2020, resulting in a total number of 230 cases on March 27, 2020.

Although many are conforming to self-isolation, there are still people who don't obey these measures, as the global civic consciousness around this issue is not strong enough.

According to this, the main success metrics are:

- Reduced number of new Coronavirus cases.
- Flattening the curve of the total number of Coronavirus cases in Bosnia and Herzegovina.
- Fewer reports (in the media or in other forms of communication) on self-isolation breaches.

Assumptions

There are several assumptions about users of the system:

- Many citizens are senior people, and/or not very technically proficient, and they would benefit from more simplistic solutions.
- People have a tendency to not follow the law and regulations in general, and will go around any measures which leave room for bypassing and cheating.
- A large number of citizens are not educated enough or not socially aware enough in order to understand the importance of the self-isolation measures enforced by the government, and voluntarily follow them.

Technical constraints include:

- Simple, easy to use design and user experience so the system can be used by everyone.
- Tight deadlines as the subject matter is very time-critical and every new day can mean new infections and deaths.
- The pandemic has slowed all industries, including iOS AppStore and Android Play Store, which means creating and releasing completely new mobile apps will probably take more time than usual and is thus not an ideal approach. A better solution would be to use resources that are already available, if possible.
- In case of a physical solution, the product would need to be distributed to a large number of people in a short time, so production needs to be efficient in terms of the resources used.

If possible, cooperation with the government and the telecommunications companies is highly beneficial due to the amount of useful data they possess. However all approaches should abide by the law.

Action Plan / Tasks

There are 3 proposed solutions, and each one will be described in detail below.

1. Viber chat bot

This solution can be implemented by using any of the available messaging apps, such as Viber, WhatsApp or Facebook Messenger. The process will be explained using Viber as an example (as most of the citizens in Bosnia and Herzegovina are using Viber), but similar effects can be achieved with other messaging apps.

One of the biggest benefits of this approach is familiarity and ease of use, as most people are already familiar with Viber and can start using it immediately.

1.1 Mobile end

Viber app will handle the mobile component of the system. It provides several ways to reach the audience, through **Public Accounts**, **Communities** and **Business Messages**.

Public Accounts are not supported anymore and they have limited access because they have been replaced by Communities. So this is not a possibility.

Communities are very similar to Viber group chats, but on a much bigger scale as they support unlimited members. The obvious disadvantage of this approach is that everyone is able to write and read everyone else's messages, which is unnecessary.

Business Messages are the preferred approach, as they provide a way to personally interact with each member and drive them to perform actions in a native environment.

Therefore, the optimal approach for reaching the users are **Business Messages**. They can be set up by first creating a **Bot Account** through the Admin Panel. After a successful account creation, a private token will be generated, which can be used to connect to the Viber Chat API.

1.2 Backend

Viber Developer Console provides NodeJS, Python and REST API solutions for developing chat bots.

Regardless of the chosen API solution, prerequisites for backend are:

- An active Viber account to use as administrator
- Active bot account
- Authentication token
- Account webhook

The backend is responsible for sending automated Viber bot messages - for example twice a day and at different times during the day, so the users can't predict when they will need to report their location.

It is also responsible for checking if the users responded on time, parsing their response locations and checking if they are self-isolated by comparing locations.

Finally, the backend is responsible for storing all data in the database and reporting any irregularities such as user failure to respond on time or receiving a location that shows that the user has violated self-isolation.

1.3 Frontend

Frontend would not be normally required for a solution that already uses Viber, since Viber provides an option for sharing location through the app. The problem is that users can easily choose a fake location by choosing a different point in the map view. So in order to avoid this, instead of sending location directly through Viber, users can receive a link through Viber chat bot, which would lead to a custom mobile website that will track their exact location via GPS.

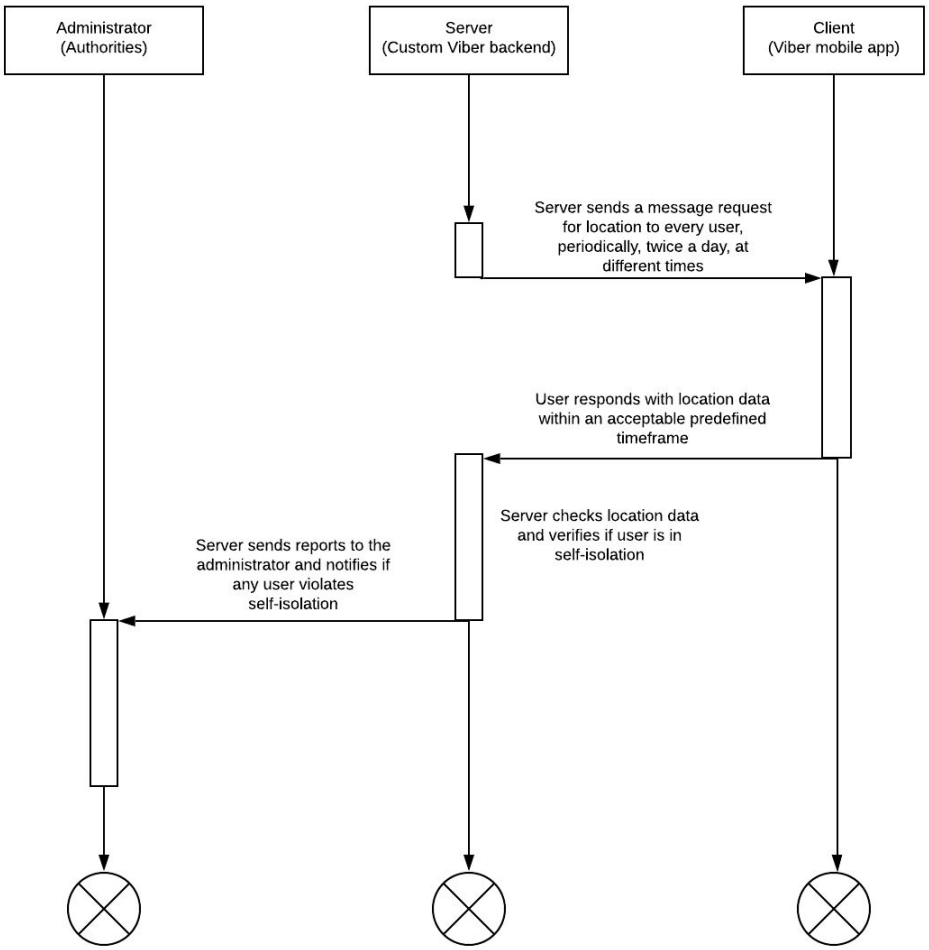
The technology for developing the frontend is irrelevant, as it will be very simple and only ask users for their location, so any of the frontend technologies (for example React) can be used.

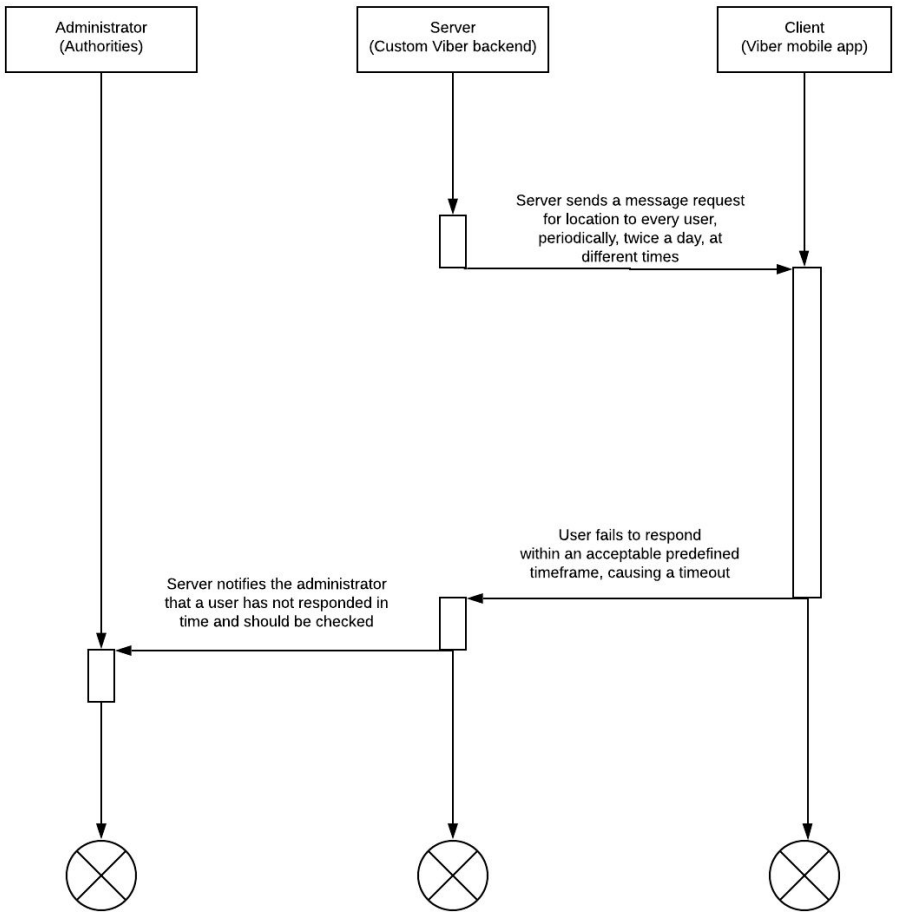
Note: This should be investigated more thoroughly, and if Viber APIs provide a way of tracking precise user location directly and without the possibility of inputting wrong data, then there is no need for a custom website.

1.4 External Services

There are many out-of-the-box solutions for backend implementation such as Rocketbots and Dialogflow, which can be integrated to the Viber chat bot, but that level of complexity is probably not needed for the scope of this project, since the focus is on simplicity and only fetching a single type of data (user location), instead of high level of interactivity in communication.

1.5 Usage diagrams





1.6 Milestones and tasks

Since the project is very time-sensitive with strict deadlines, it doesn't allow for a large number of milestones. However, the implementation can be split in the following tasks:

1. Create and set up a Viber business account
2. Set up and deploy a custom backend by integrating one of the Viber Developer libraries
3. Design, or integrate if there is an already existing database that stores user data such as phone numbers and location data
4. Implement Viber Bot solution that sends periodic messages to users at different times during the day
5. Set up web hooks for handling user location responses
6. Implement an algorithm that checks if received locations correspond to their self-isolation coordinates
7. Implement the reporting mechanism that notifies the administrator of self-isolation violations

2. SMS Messaging

This solution can be realized by sending SMS messages with a link that opens a mobile website which tracks a user's location and sends it to the server.

The main advantage of this approach is that it doesn't require external frontend apps (besides a simple website), and every mobile phone has SMS capabilities.

The main prerequisite is acquiring the citizens' phone numbers, which should be achieved by the medical staff when a citizen tests positive for Coronavirus.

2.1 Mobile end

Mobile component of the system is the native SMS capability of any mobile phone. The only function of this component is to receive an SMS with a link that will lead to a website.

2.2 Backend

Backend component will be a simple server whose purpose is to send automated SMS messages to phone numbers taken from the database. Any backend technology can be used for this purpose. It is recommended to integrate a platform that is able to perform this task easily, such as Twilio.

Similar to the first solution (Viber bot), the backend also has a task of receiving, checking and storing users' location responses from the website, and tracking if the users that received the SMS responded in a timely manner.

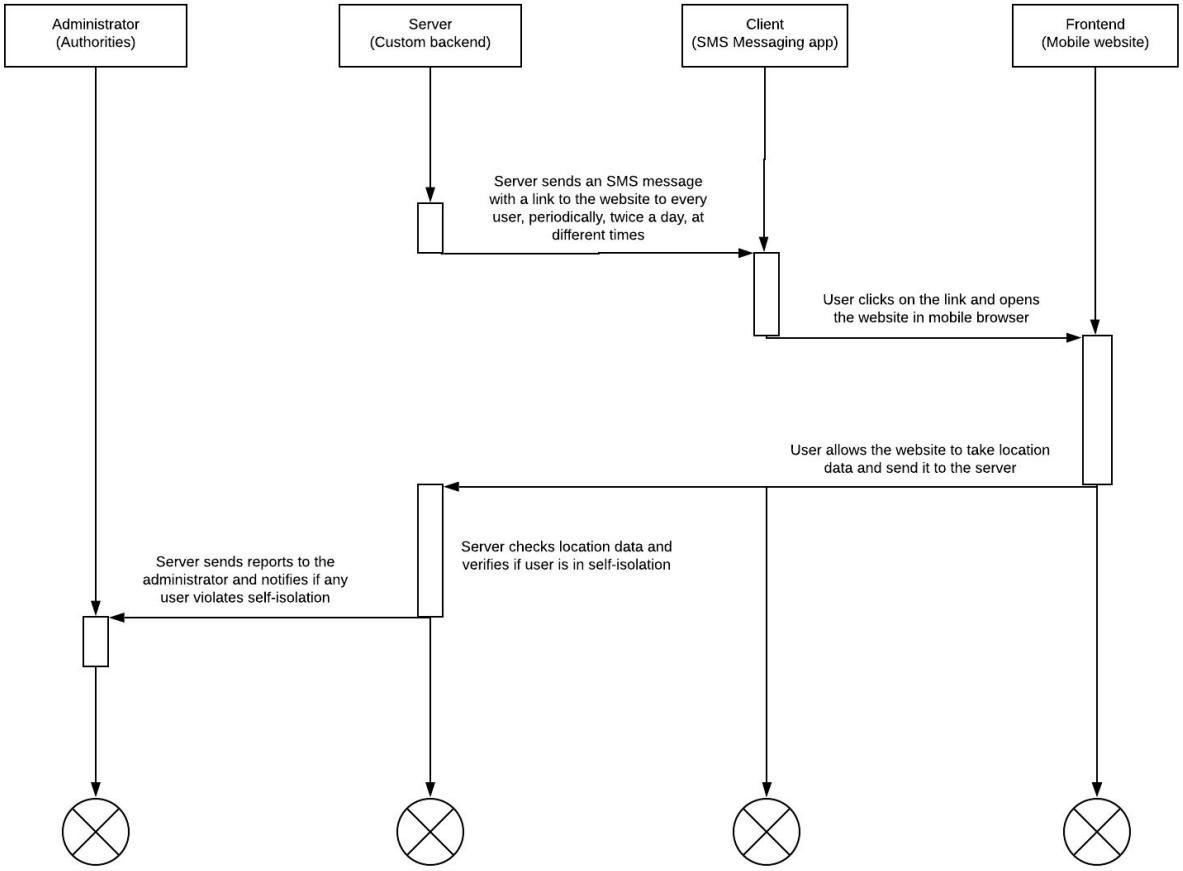
2.3 Frontend

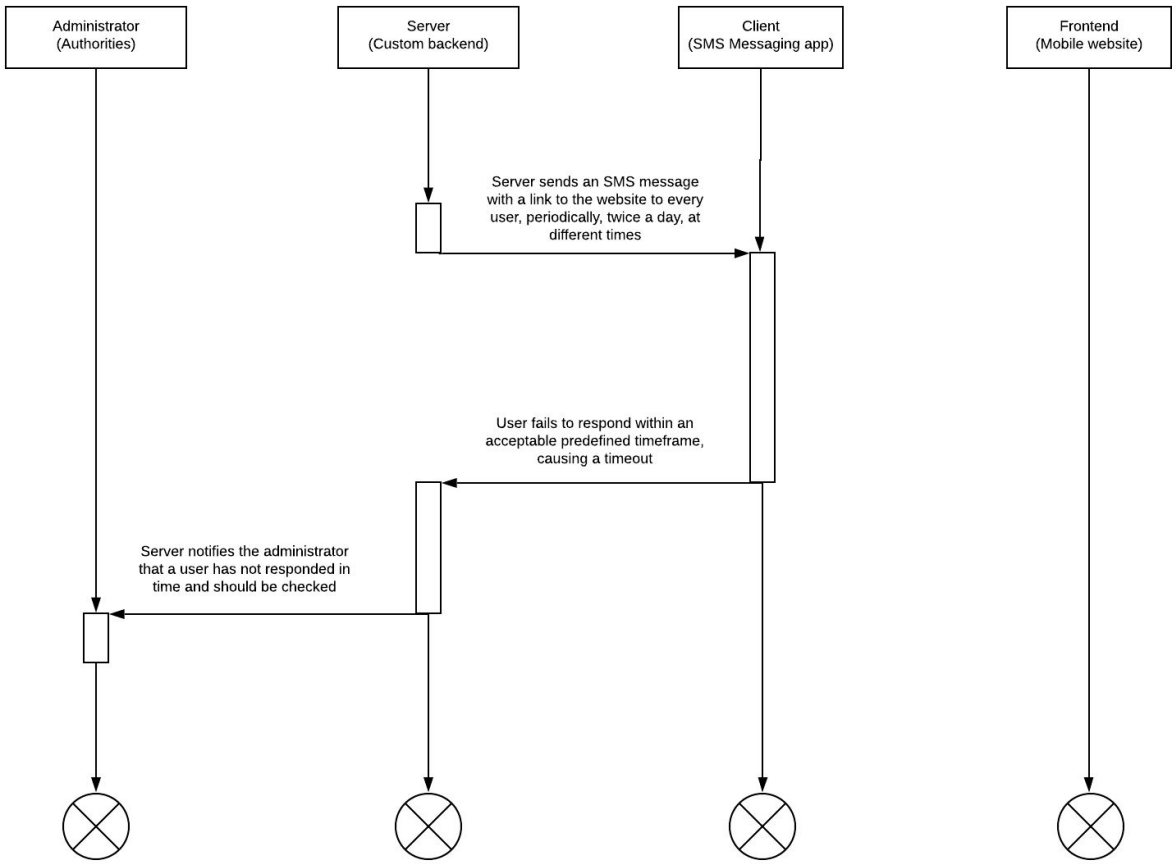
The frontend component is a website whose only function is to track users' location and send it to the server. It should be straightforward and simple.

2.4 External services

The only external service that needs to be used is Twilio, as it provides an easy solution for integrating support for programmable SMS messaging.

2.5 Usage diagrams





2.6 Milestones and tasks

The implementation can be split in the following tasks:

1. Set up and deploy a custom backend
2. Integrate and set up Twilio so it’s possible to send programmable SMS messages with a link to the website
3. Implement a mobile website that tracks users’ location and sends it to the server
4. Implement an algorithm that checks if received locations correspond to their self-isolation coordinates
5. Implement the reporting mechanism that notifies the administrator of self-isolation violations

3. Location-tracking wristbands

This solution is realized by rolling out electronic tracker wristbands that alert authorities if a citizen in self-isolation leaves the quarantine area. The concept is similar to digital handcuffs used in correctional / detention facilities, such as prisons and jails. Citizens in self-isolation must wear their wristbands that send location data to the backend in order to ensure they do not leave their quarantine zone.

3.1 Hardware component

Unlike the previous two proposed solutions, this solution doesn't require a mobile phone. Location-tracking wristbands are used as a hardware component instead.

Wristbands can be produced by using the 3D printing technology to create the bracelet body and then pairing it with a microcontroller with a GPS chip that will be responsible for tracking the location, or re-using handcuffs and infrastructure provided by detention facilities. Regardless of the approach, it is necessary to produce a large number of wristbands, so all citizens in quarantine can be tracked.

There are many possible choices for the microcontroller component, such as ESP8266, or Arduino, who have specifically stated on their website that they are prioritizing the fastest possible delivery of any hardware or software to facilitate the prototyping and production of any equipment or solution to lessen the impact of Covid-19.

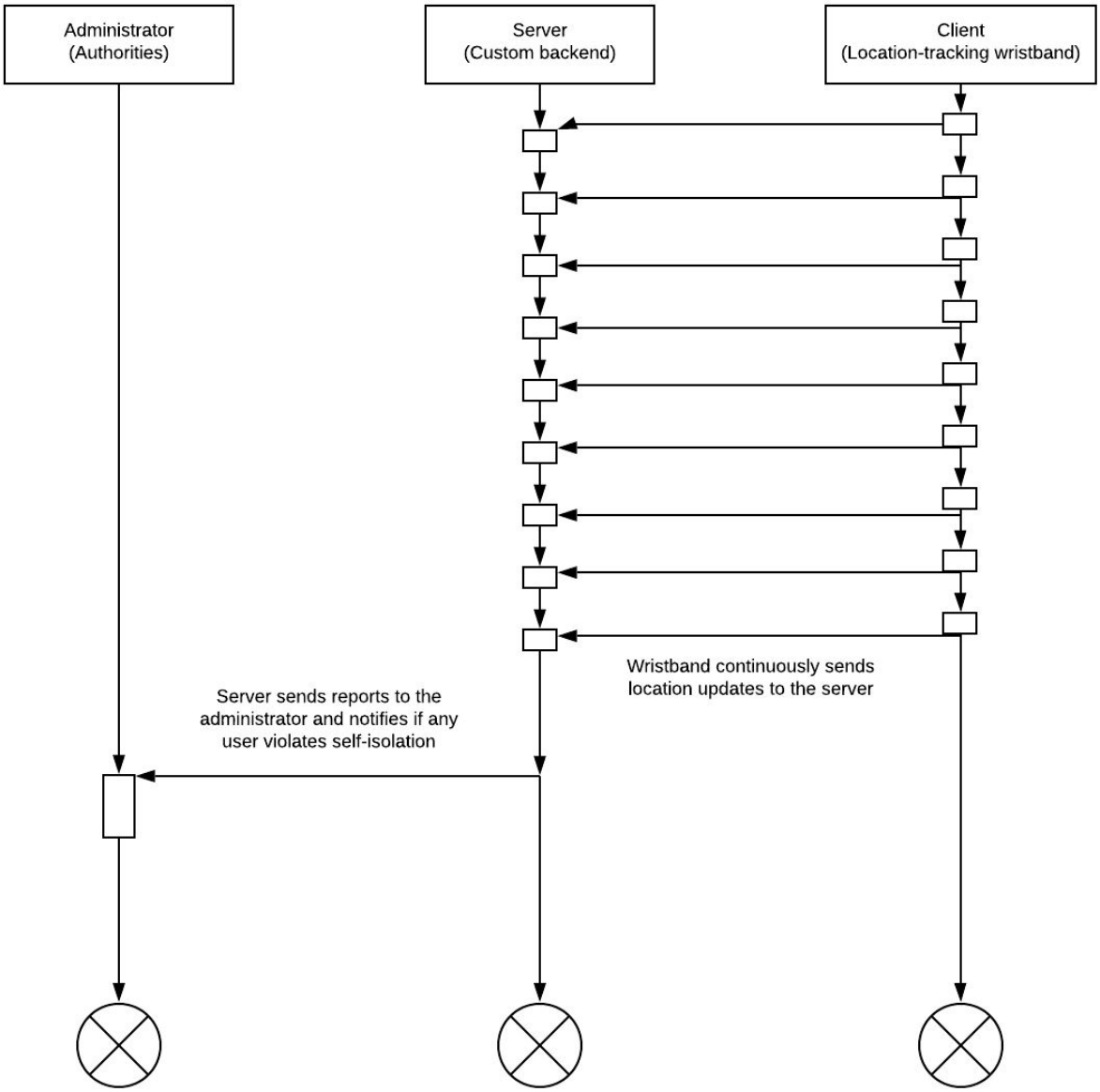
In case of choosing to 3D print the wristbands, due to limited time and resources, it is advised that the community helps in the production process as much as possible, similar to the engagement they showed in producing masks. Another important factor in the entire process is producing enough batteries for every wristband that could last, because they would need to be active all the time.

Besides production and supply requirements, the final requirement for this solution is the delivery of wristbands to end users who will be required to wear them at all times.

3.2 Backend

Backend needs to be able to continuously receive location updates from each user, regardless of the technology used. Other than that, it has the same requirements as in the previous two solutions, and that is checking if the reported locations are valid and reporting any irregularities.

3.3 Usage diagrams



3.4 Milestones and tasks

The implementation can be split in the following tasks:

1. Set up and deploy a custom backend
2. Ensure production of the necessary hardware: wristbands, microcontrollers and batteries
3. Configure and program the microcontroller that is responsible for GPS tracking and connect it to the backend

4. Implement an algorithm that checks if received locations correspond to their self-isolation coordinates
5. Implement the reporting mechanism that notifies the administrator of self-isolation violations