



ContinuousCare: System Requirements and Architecture Informatics Project

André Pedrosa[85098], Filipe Pires [85122], João Alegria [85048]
Coordinator Professor Carlos Costa

Table of Contents

1. Requirements Elicitation.....	2
1.1. System Goals.....	2
1.2. State of the Art.....	2
1.2.1. Beyond Health Tracking	2
1.2.2. Pervasive Health Service System	3
1.2.3. UPHSM	3
2. Functional and Nonfunctional Requirements.....	4
2.1. Requirements.....	4
2.1.1. Functional	4
2.1.2. Nonfunctional	4
2.2. Target Audience.....	5
2.2.1. Actors	5
2.2.2. Use Cases	5
2.2.3. Domain Model	8
3. System Architecture.....	9
3.1. Deployment	9
3.2. Architecture	10
3.3. Mock-Up	11



1. Requirements Elicitation

This document is the result of a specification process that can be divided in three main parts. The first one, where the system's main goals were evaluated and its scope defined, is detailed in this chapter.

1.1. System Goals

In short words, our goal is to develop a system of web information that allows the automatic and transparent gather of clinical and environmental data sources and an intuitive and integrated form of visualization of the daily information of a citizen.

The solution, that we called ContinuousCare, will collect data such as weather conditions, air quality indicators, vital signs, physical activity, geolocation, etc. It will also allow a simplified form of registering basic physical and mental health states of a person such as local pains, depression, anxiety, fatigue, etc.

It is expected to observe the usage of our system anywhere a person is. Why is that? Since the clinical and environmental data collected helps to analyse several aspects of a person's life, to detect bad health habits and to understand symptoms and personal complaints, the applicability of the system extends to anyone concerned with their lifestyle quality.

When called for a regular check-up by the family doctor, the user no longer needs to try to remember relevant episodes for the doctor. Instead, the doctor simply requests access to the user's information collected by ContinuousCare and makes the analysis with the tools offered by the system itself. This makes the usage of the software by clinical offices very appealing.

The idea, however, is to increase the depth of applicability and build the system with great scalability and mechanisms to serve not only our daily routines, but also other softwares and machine learning models and make available rich data to more elaborate studies on long term diseases.

1.2. State of the Art

With our mindset consolidated, the following step was to make a research on projects, studies and technologies with similar purposes. This improved the system requirements firstly, by discovering the latest and most innovative works around this area; and secondly, by understanding what went right and wrong in those works.

1.2.1. Beyond Health Tracking

This article from University of Trento and Hospital of Milan is about a personal health and lifestyle platform.

To summarize the work done for this paper, based on the idea of personal health record (PHR), this team created a personal health and lifestyle record which made easier for the individual to keep track of and maintain a healthy lifestyle and helped him with personalized advice and ways he could improve through a platform that monitored and assessed all data collected.



Their GPS based monitoring application may bring useful contributions to our research since it tracks the cyclist's position in real time, computes his/her power output and overall energy consumption, shares that data with others, and provides nutrition advices.

The sharing feature for performances and result presented itself a very interesting characteristic of their solution, since it proved that people and professionals follow advices given by others with same goals. However, the application of this feature in ContinuousCare seems to be not as useful and a bit inappropriate.

1.2.2. Pervasive Health Service System

PHSS is a ubiquitous health service system, a grid-based personal health service system targeted to fulfill requirements of home-care. The main goal was to reduce medical management expenses and burdens of hospitals by making use of a home healthcare system, proposed by the authors.

The top layers of the infrastructure, that actually use the collected data (amongst which, a few new datatypes were noticed such as body temperature and fall detection), divided in two parts: a platform and the applications and resulted in a situation-aware, context-aware and environment-aware cyber mobile health-service software.

Even though the entire system was strongly based on enormous computing and resource sharing capacities of an underlying grid, some aspects of the authors' research may still be useful to us during the development phase. It was clear to us that this system was very complete and presented itself as a product prepared to be put to market production – it turned out to be a great example to follow more in a strategic manner than in a technical one.

1.2.3. UPHSM

The UPHSM (Ubiquitous Personal Health Surveillance and Management) system is in many ways similar to ContinuousCare. This project integrates health-related sensory devices to aid senior citizens and patients who need long-term attention to their illness, with the help of ubiquitous surveillance and remote management of the recorded data. It differs on the target audience, since it focuses on elderly patients and messages sent to family members and doctors, and on the absence of environmental data analysis.

The research team used several sensors that could be interesting to be integrated with ContinuousCare, such as: electromyography (EMG) sensor for muscle activity, electroencephalography (EEG) sensor for brain activity, blood pressure sensor, blood glucose sensor, breathing sensor for respiration, etc. Though we focus on wearables already common in regular citizens, these sensors would enrich the diversity of datatypes and improve the diagnoses of doctors connected to our system. However, the need for additional hardware presented a big challenge to the UPHSM team. Their modularity and use of wireless communication technologies may be useful during our development phase, taking in consideration the success of the software.



2. Functional and Nonfunctional Requirements

Once the first part of the specification process was completed, it became possible to define all of the requirements needed to be met for the system to work as intended. These requirements are listed in this chapter and the target audience is described with greater detail along with the actors' use cases.

2.1. Requirements

2.1.1. Functional

There are three main components in the system, these are: the devices used by the actors, the user interface (UI) from where they can interact with the system, and the online server, where most of the activity happens. In a typical scenario, an account will have associated two devices, 1 non-portable for air quality measurements, and 1 smart bracelet for the remaining features. The following list covers all functional requirements that must be fulfilled. It serves as a description of what the software should do.

1. The home device must send data to the server whenever an actor is indoors.
2. The personal device (bracelet) must send data to the server whenever an actor uses it.
3. The server must store all data collected from the devices in a database.
4. The system must keep track of the actor's geolocation at all times.
5. The server must collect environment data based on the actor's geolocation.
6. The system must still collect data even when the devices are not being currently used (worn).
7. The UI must make available to the actor all data collected.
8. The UI must present the latest data and allow the access to the progress data.
9. The system must support multiple accounts.
10. The system must have to separate user profile types.
11. The system must be prepared to allow access to the information to other actors (e.g. medical personnel) with the owner's permission.
12. The system must be prepared to receive and store health states according to the actor.
13. The system must support accounts with several home devices.
14. The system must support several accounts with the same home device.
15. The system must not allow the use of several personal devices by the same account.
16. The system must allow the association and deassociation of devices (e.g. device update).
17. The UI must inform the actor which devices are associated.

2.1.2. Nonfunctional

Since usability and data processing are key aspects for the success of the software, a list of constraints that it must respect was created for future reference during the development phase of the project. This list also includes the properties/characteristics the it must exhibit.

1. The system must be available at all times (24/7).
2. The devices must not stop producing data (unless if they run out of battery or stop functioning).
3. All information must be accessible by the actor without exceptions.
4. The system must always persist all data of all accounts.
5. The system must be prepared to expand in case of a large increase of number of users.
6. All system functionalities must be well documented.
7. The system's database must store all data in the least amount of space possible.
8. All actors' personal data must be confidential to the owners alone.
9. The user interface must be supported by the most common web browsers.
10. All requests must be responded within a timespace no longer than 1 second.



2.2. Target Audience

2.2.1. Actors

The target user for the services provided by the system belongs to one of 2 groups (with the possible existence of a third group): either he/she is a common user (a client) or a medical user (a doctor). The third group refers to a developer that integrates the system with his/her own application.

- **Client:** Has access to all features of the web interface; He is able to see historical data, give permission to doctors to see it and manage the associated account devices.
- **Doctor:** also referred as medic; Only has features available when a permission is given by a client; Assuming it is, he can analyse client's historical data as a helping tool to make a better diagnostic of the clients health state.

It is expected to see medical personnel using the system with high frequency. The levels of experience with technology needed for both these actors to be comfortable with the system are relatively low. An actor only needs to be used to work with the basic tools of a computer and a web browser in order to properly use our tools.

Regarding the set up of the external devices and their maintenance, these are tasks that may make actors encounter difficulties that require some more experience. However, it is a matter taken in consideration and we intend to make available everything necessary for any kind of person to be able to use the system without much entropy.

Both actors need to be authenticated before any communication with the server, since sensitive and personal data is stored in our server.

2.2.2. Use Cases

Figure 1 represents the use case model of the system, having only one main package: the web application. As said before, both the client and doctor use the web application but with different purposes. For this reason, we grouped their use cases in different packages within the web app package.

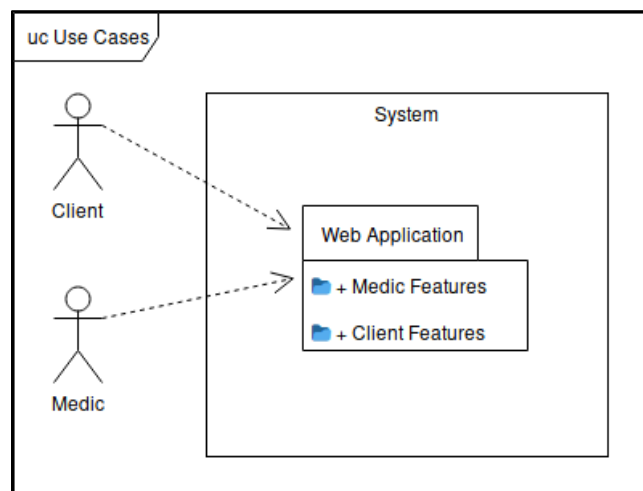


Figure 1: Use case model in UML format.

The web application use case diagram is present in figure 2. The diagram shows the activities that both the client and the doctor can do, with each package linked to the respective use cases. The doctor package is mostly characterized by its dependency upon the client package.

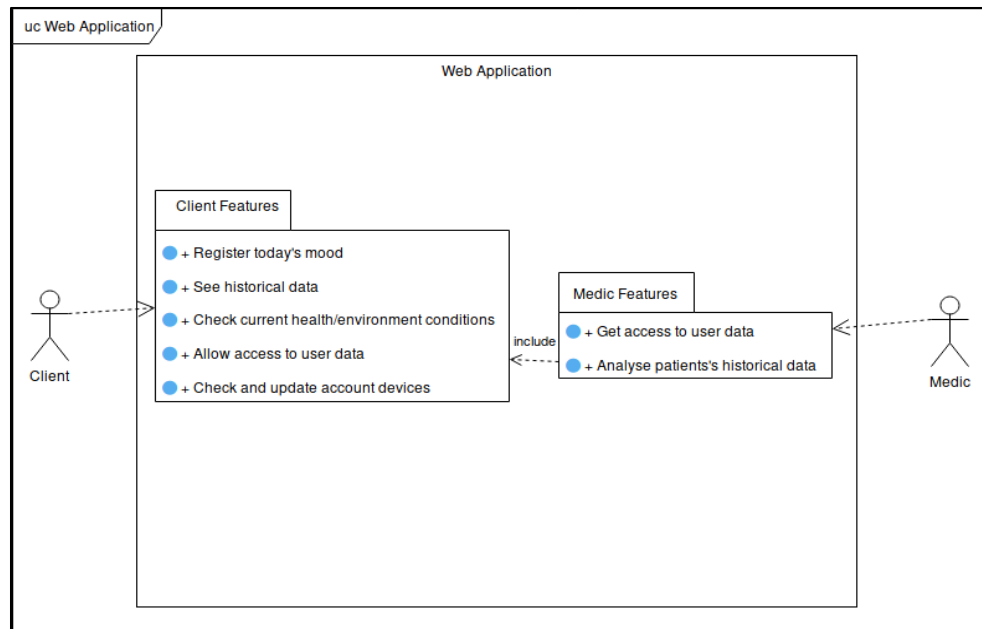


Figure 2: Web application use case diagram.

All operations that a client can do on the web application are presented and described bellow. All use cases on figure 3's diagram assume that the client is already authenticated.

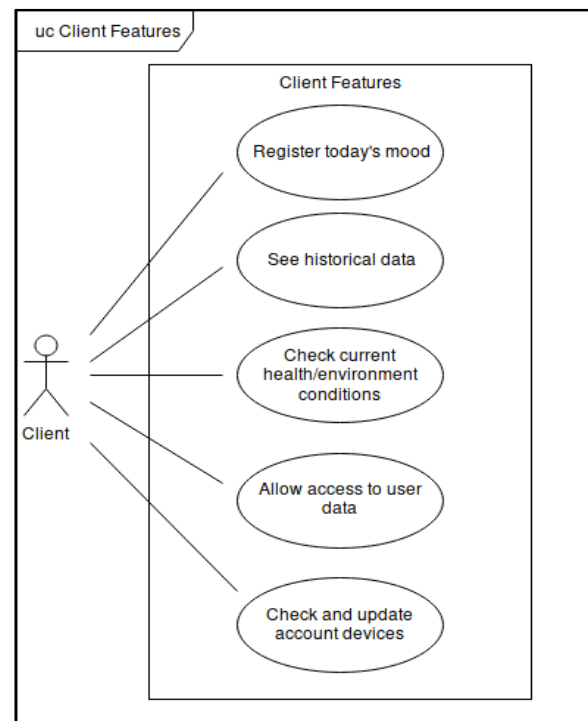


Figure 3: Client features.



- **Register today's mood:** Allow the client to register his health status during the current day by selecting options from a predefined set. This makes possible to, later on, use the data retrieved from the external devices and already labelled by the system to be fed to machine learning algorithms applicable according to the labels. Table 1 presents a list of possible daily emotions and health issues made available as options for the user.
Priority: Medium
- **See historical data:** Allow the user to browse through the personal historical data so that he can keep track of his health and progress.
Priority: High
- **Check current health/environment conditions:** Although the user can see this type of data on the external devices, a second option is given to the client where he can access the web application to consult the latest values.
Priority: Low
- **Allow access to user data:** Allow a client to grant temporary access to a medic for professional analysis purposes. This activity is aimed to be used during medical consultations.
Priority: High
- **Check and update account devices:** Make possible the association and dissociation of new/old external devices to the user's account.
Priority: High

All operations that a doctor can do on the web application are presented and described next. All use cases on figure 4's diagram assume that the medic is already authenticated.

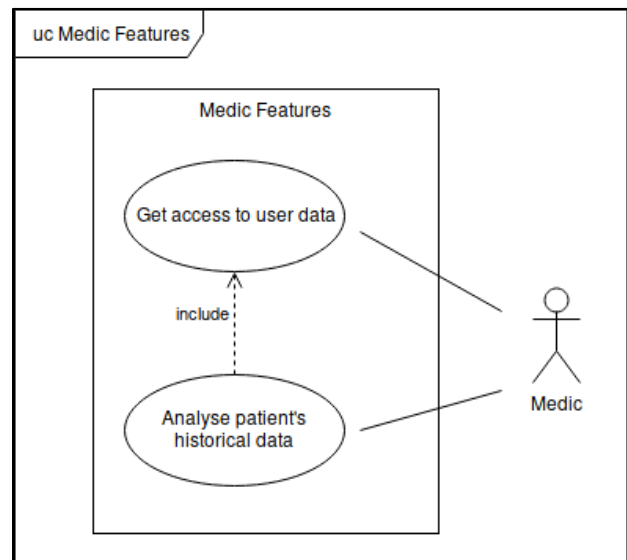


Figure 4: Doctor Features.

- **Get access to user data:** The same way a client can grant access to doctor accounts, a doctor is also able to ask permission himself. The acceptance process must involve the presence of both entities in the same space.
Priority: High
- **Analyse patient's historical data:** Allows the medic to analyse the historical data during a consultation with a client. This activity depends on the medic having the permission to access the data of the patient that he is currently attending.
Priority: High



Amused	Angry	Cold / Respiratory Pain
Calm	Annoyed / Irritated	Insomnia / Difficult Night
Cheerful	Apathetic	Migraine / Headache
Dreamy	Bad	Muscle Soreness or Injury
Excited	Cranky / Grumpy	Allergies
Flirty	Depressed	Back or Joint Pain
Good	Envious	Skin Conditions
Happy	Frustrated	Stomache or Intestinal Discomforts
Joyful	Gloomy	Overstress / Heart Discomfort
Loving	Guilty	
Mellow	Indifferent	
Optimistic	Melancholy	
Peaceful	Pessimistic	
Silly	Rejected	
Sympathetic	Restless	
	Sad	
	Stressed	
	Weird	

Table 1: List of possible options made available for the user to describe his/her day.

2.2.3. Domain Model

The domain model is used in this phase of the project to help us developers keep an abstraction of the system model, describing in an intuitive way the entities, roles and relationships. Our domain model is present in figure 5.

The first entity is the user, divided into the two actors specified in section 2.2.1. The client can have external devices associated that can be of two types: bracelet (Fitbit), home device (Foobot) and Public API (In case the client isn't home). A record of a specific metric is recorded by an external device and is associated with a client.

Regarding permissions both the medic and the client can ask or grant permissions, respectively, many times. This implies the existence of a one-to-many relationship. Each permission can cover one or more records. Hence the presence of a many-to-many connection between record and permissions.

Each client has a set of external devices associated, each external device can be associated with more than one client. For this, the relation between the client and the external devices is many-to-many. At last, a record is associated with only one client and only one external device so for both connections one-to-many relationships were used.

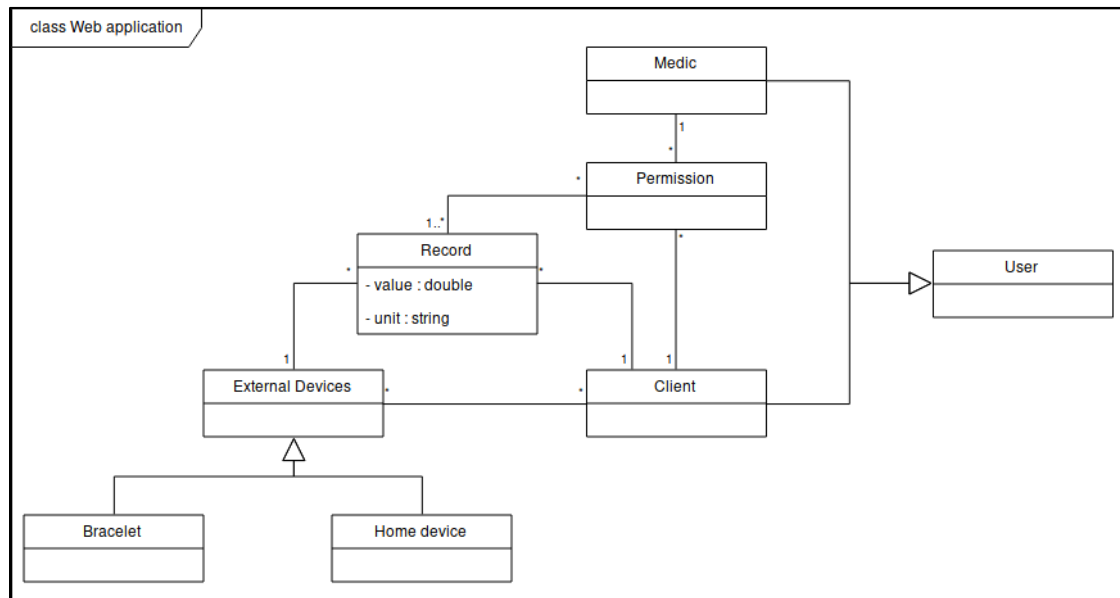


Figure 5: Domain model.

3. System Architecture

3.1. Deployment

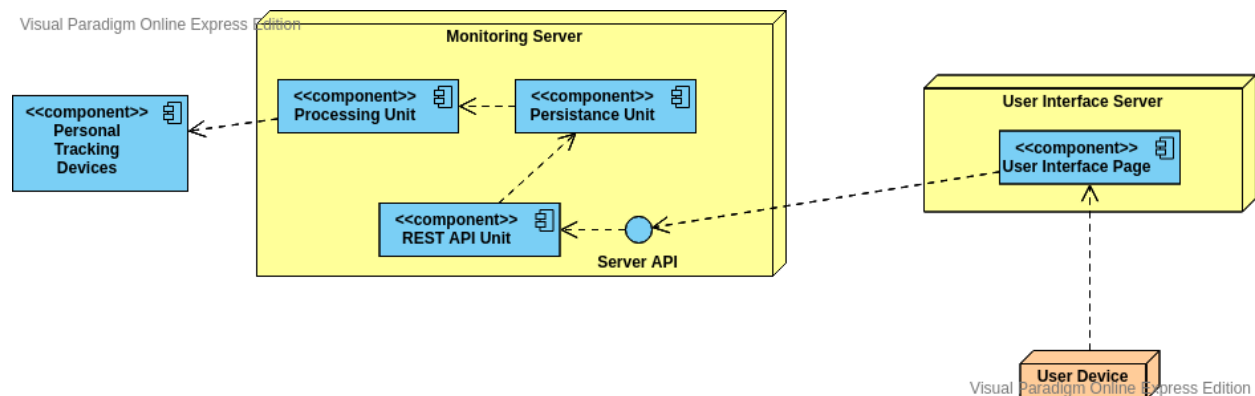


Figure 6: Deployment model.

Our current plan for the deployment model is susceptible to future adaptations. However, it already offers a visual description of how we intend to deploy the system, taking in consideration the machines required, the separation of components and other necessary external devices.

The system will be distributed over two machines, one for the logical server responsible for the processing and persistence units and the REST API described in the next subsection, and one for the web application server.

Other devices are taken into consideration like the personal and house devices and the personal computers used by the client to connect to the user interface machine to access the system.



3.2. Architecture

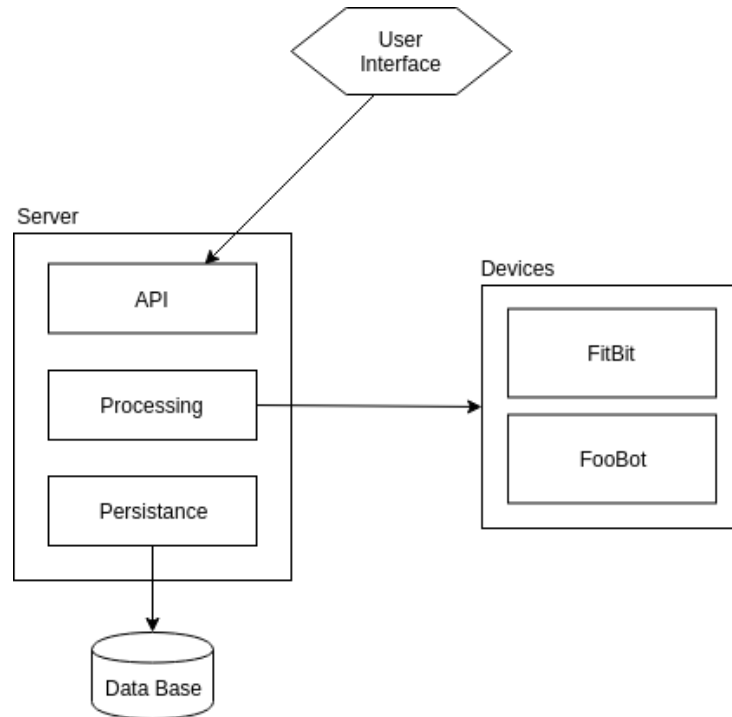


Figure 7: Architecture model.

The image represents our view of the system's architecture also a target of future changes if found necessary. To better understand the architecture, we make available a brief description of each component and of the overall construction. The idea is, given that we need to collect the information from personal and home devices and present it, a server available 24/7 is required for doing all the core procedures and supply the service, the devices themselves as data sources and the client application.

The server itself is divided into different units: processing unit, persistence unit and REST API unit.

The processing unit is responsible for gathering the information from all the devices and other external sources like public APIs; This operation is scheduled, which means that it is executed many times along the day; The unit is responsible for doing this scheduling for processing the data collected, like filtering it, assigning labels, etc. It is our plan to implement this unit with the use of Python libraries such as Flask. The persistence unit is the component responsible for storing all the data after processing; It is this component that will interact with the database interface and make sure every single field is persisted correctly. We intend to use a polyglot persistence, with the help of InfluxDB (time series database) and MySQL (relational database).

The REST API unit is the component responsible for supporting a REST (Representational State Transfer) interface for the entire system, serving as the point where every client application will connect to.

The last component is the client interface. We found that the most appropriate solution would be a web application that allows the support of every device, given that it has access to the internet. It is this app that communicates with the server API and presents in a simple and intuitive way the information gathered along with other useful features.



3.3. Mock-Up

The following mockup is the result of a more profound planning of the user interface. Though susceptible to further changes and not completely alike to the final result, it will serve as a solid basis to what the system will look like to the user.

WELCOME!

About xxxx...

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

Who We Are?

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

Register Now!

Create New Account

Name

Surname

Password

Confirm Password

Date of Birth

Height


Weight


Public Health Personal Number

Upload Image(Optional)

Register





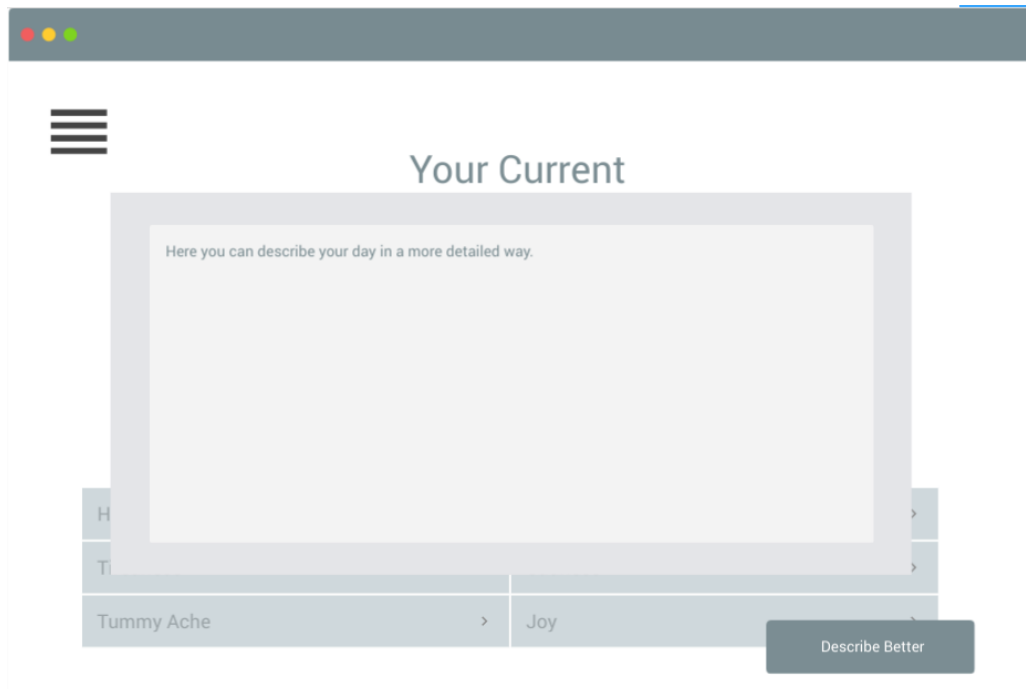
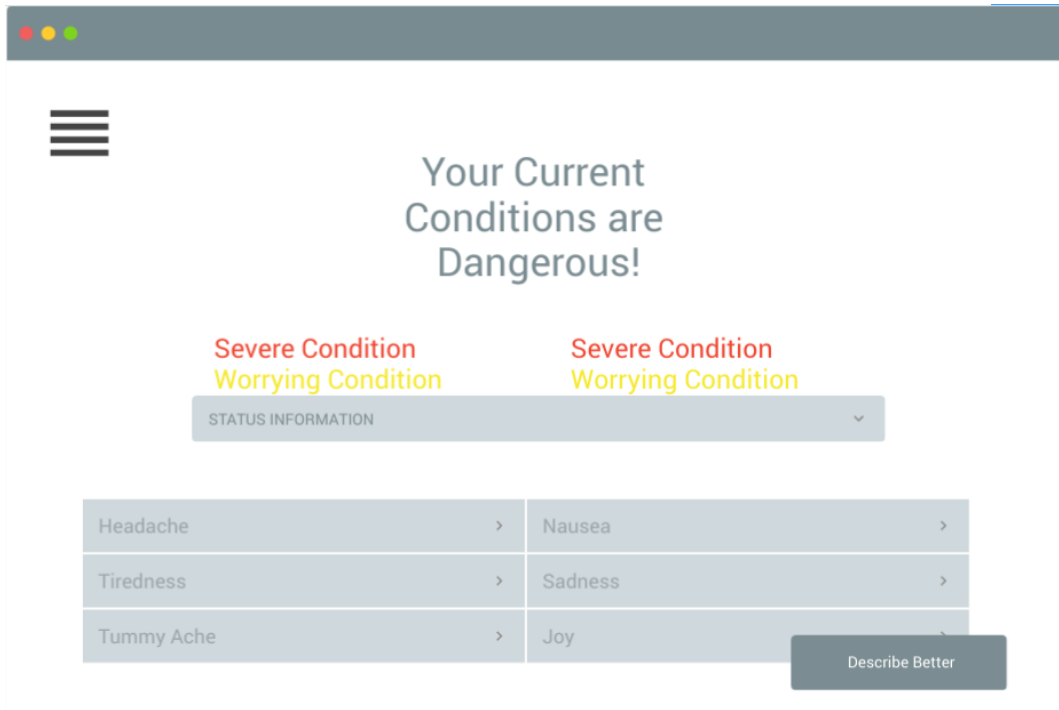


Your Current Conditions are Healthy!

STATUS INFORMATION

Headache	>	Nausea	>
Tiredness	>	Sadness	>
Tummy Ache	>	Joy	>

Describe Better





Your Current

Please update the following fields:

Height

Weight

Tummy Ache

Joy

Describe Better

HOME

HISTORY

DEVICES

ABOUT

SETTINGS

LOG OUT

Your Current Conditions are Healthy!

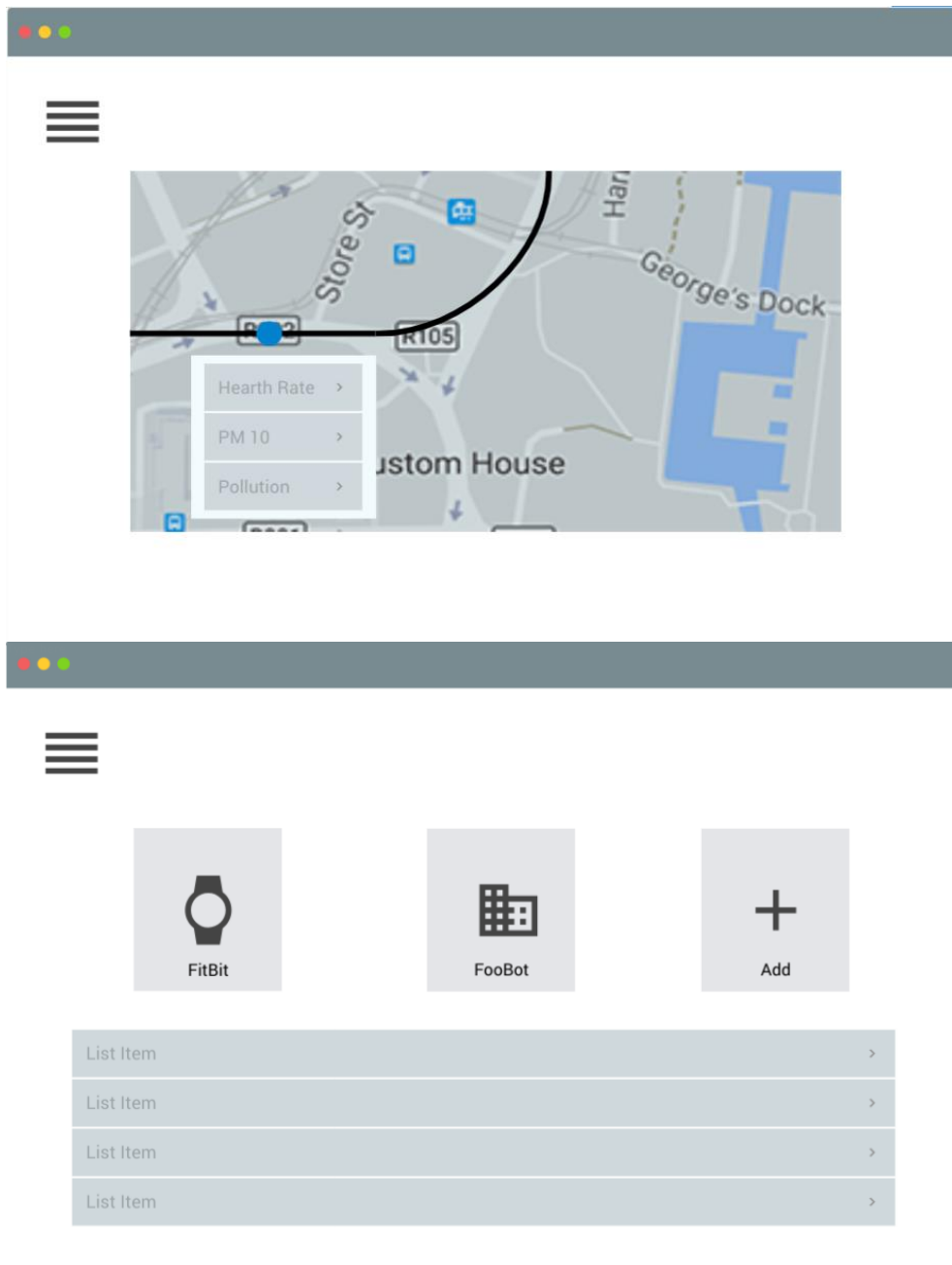
FORMATION

Nausea

Sadness

Joy

Describe Better



Update Photo

Update Password

Name

Surname

Password

Confirm Password

Date of Birth

Height

Weight

Public Health Personal Number

User XXX has used the system for 3 years and 2 months!

Update Info

The screenshot shows a web application interface. At the top, there is a dark grey header bar with three colored circles (red, yellow, green) on the left. Below the header, on the left side, is a sidebar with a hamburger menu icon (three horizontal lines) and a list of items, each with a right-pointing chevron. A large, light grey modal dialog is centered on the screen. The dialog has a title 'Add Device' and two input fields labeled 'Name' and 'Type'. Below the inputs, the text 'Syncing...' is displayed.