

**Project Acronym:**  
**Grant Agreement number:**  
**Project Full Title:**

NUTRISHIELD  
818110 (H2020-SFS-2018-IA)  
Fact-based personalised nutrition for the young



## DELIVERABLE

### D2.6 – NUTRISHIELD Platform technical requirements & use case design

<b>Dissemination level</b>	PU – Public
<b>Type of Document</b>	Report
<b>Contractual date of delivery</b>	30/06/2019
<b>Deliverable Leader</b>	INTRASOFT INTL (INTRA)
<b>Status &amp; version</b>	Final
<b>WP responsible</b>	WP2 (RU)
<b>Keywords:</b>	Use case, requirements, data management, design

<b>Deliverable Leader:</b>	INTRA
<b>Contributors:</b>	Miltiadis Kritikos (INTRA), Stylianos Georgoulas (INTRA)
<b>Reviewers:</b>	HUA, ALPES
<b>Approved by:</b>	ALPES

Document History			
Version	Date	Contributor(s)	Description
v0.1	20/05/2019	INTRA	Draft
v0.2	22/06/2019	INTRA	Addressed comments by HULAFE,QRT,SWB
v0.3	26/06/2019	INTRA	Internal review version
v0.4	27/06/2019	INTRA	Addressed comments by ALPES
v0.5	28/06/2019	INTRA	Addressed comments by HUA
v1.0	28/06/2019	INTRA	Final version

*This document is part of a project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 818110. It is the property of the NUTRISHIELD consortium and shall not be distributed or reproduced without the formal approval of the NUTRISHIELD Management Committee. The content of this report reflects only the authors' view. EC is not responsible for any use that may be made of the information it contains.*

## Executive Summary

This report presents the needs from the project partners regarding the NUTRISHIELD Platform and App. It analyses the requirements as documented in WP1, WP2 and WP9 deliverables as well as requirements from preliminary work performed on WP3. It then proceeds by elaborating on the use cases, functional and non-functional requirements. The requirements identified are prioritised as *must*, *should* and *could* with the view to allow set a roadmap for potential longer term extensions and enhancements to the platform and app. It concludes with a preliminary architecture of the NUTRISHIELD platform.



# Table of Contents

<b>Executive Summary .....</b>	<b>2</b>
<b>Table of Contents .....</b>	<b>3</b>
<b>Table of Figures .....</b>	<b>7</b>
<b>List of Tables .....</b>	<b>8</b>
<b>Definitions, Acronyms and Abbreviations.....</b>	<b>9</b>
<b>1. Introduction .....</b>	<b>10</b>
1.1. Methodology.....	10
1.2. Relation to other Work Packages .....	10
1.3. Document structure.....	11
<b>2. Requirements analysis .....</b>	<b>12</b>
2.1. Gap analysis .....	12
2.2. Laboratory data .....	13
2.2.1. Microbiome analysis.....	13
2.2.2. Genome analysis.....	13
2.2.3. Urine analysis .....	14
2.2.4. Milk analysis .....	14
2.2.5. Breath analysis .....	14
2.3. Sensor data .....	14
2.3.1. Urine analysis .....	15
2.3.2. Milk analysis .....	15
2.3.3. Breath analysis .....	15
2.4. Environment data .....	15
2.4.1. Urine analysis. ....	15
2.5. Data management .....	16
2.5.1. GDPR compliance .....	16
2.5.2. Anonymization .....	16
2.5.3. Data formats.....	16
2.5.4. Machine Learning analysis.....	16
<b>3. Actors .....</b>	<b>18</b>
3.1. Users.....	18
3.2. Components .....	18
<b>4. Use cases .....</b>	<b>19</b>
4.1. How to read this section .....	19
4.1.1. The Basic flow.....	19
4.1.2. Alternative and Exception flows.....	20
4.1.3. Use case summary.....	20
4.2. Administration .....	21
4.2.1. UC-ADMIN-1: View medical professionals list.....	21



4.2.2.	UC-ADMIN-2: Create health professional user .....	21
4.2.3.	UC-ADMIN-3: View medical professional details .....	22
4.2.4.	UC-ADMIN-4: Edit medical professional details .....	22
4.2.5.	UC-ADMIN-5: Delete medical professional .....	23
<b>4.3.</b>	<b>Account.....</b>	<b>23</b>
4.3.1.	UC-ACCOUNT-1: Login .....	23
4.3.2.	UC-ACCOUNT-2: Change password .....	24
4.3.3.	UC-ACCOUNT-3: Logout .....	25
4.3.4.	UC-ACCOUNT-4: Impersonate .....	26
4.3.5.	UC-ACCOUNT-5: Set nickname .....	26
<b>4.4.</b>	<b>Patients.....</b>	<b>27</b>
4.4.1.	UC-PATIENTS-1: View patients list.....	27
4.4.2.	UC-PATIENTS-2: Create patient .....	28
4.4.3.	UC-PATIENTS-3: View patient details .....	28
4.4.4.	UC-PATIENTS-4: Edit patient .....	29
4.4.5.	UC-PATIENTS-5: Delete patient .....	29
<b>4.5.</b>	<b>Measurements .....</b>	<b>30</b>
4.5.1.	UC-MSR-1: View patient measurements list .....	30
4.5.2.	UC-MSR-2: Add patient measurements .....	31
4.5.3.	UC-MSR-3: View patient measurement details.....	32
4.5.4.	UC-MSR-4: Edit patient measurements.....	32
4.5.5.	UC-MSR-5: Delete patient measurements .....	34
4.5.6.	UC-MSR-6: Associate QR code with patient .....	34
4.5.7.	UC-MSR-7: Add patient measurements from QR code .....	35
<b>4.6.</b>	<b>Foods .....</b>	<b>37</b>
4.6.1.	UC-FOOD-1: View list of foods.....	37
4.6.2.	UC-FOOD-2: Add food .....	37
4.6.3.	UC-FOOD-3: View food details .....	38
4.6.4.	UC-FOOD-4: Edit food.....	38
4.6.5.	UC-FOOD-5: Delete food .....	39
<b>4.7.</b>	<b>Nutrition algorithm.....</b>	<b>40</b>
4.7.1.	UC-NUTR-1: View suggestions.....	40
4.7.2.	UC-NUTR-2: Generate suggestion .....	40
4.7.3.	UC-NUTR-3: View suggestion details .....	41
4.7.4.	UC-NUTR-4: Delete suggestion.....	41
<b>4.8.</b>	<b>Dietary and activity plans.....</b>	<b>42</b>
4.8.1.	UC-PLAN-1: View plans list .....	42
4.8.2.	UC-PLAN-2: Create plan.....	42
4.8.3.	UC-PLAN-3: View plan details.....	43
4.8.4.	UC-PLAN-4: Edit plan .....	44
4.8.5.	UC-PLAN-5: Delete plan.....	44
4.8.6.	UC-PLAN-6: Send plan to patient .....	45
4.8.7.	UC-PLAN-7: Send notification to patient .....	46
<b>4.9.</b>	<b>Food and activity diary.....</b>	<b>46</b>
4.9.1.	UC-DIARY-1: View diary entries.....	46
4.9.2.	UC-DIARY-2: Add diary entry .....	47
4.9.3.	UC-DIARY-3: View diary entry.....	48
4.9.4.	UC-DIARY-4: Edit diary entry .....	48
4.9.5.	UC-DIARY-5: Delete diary entry .....	49



<b>4.10. Mobile phone app .....</b>	<b>49</b>
4.10.1. UC-APP-1: Login to mobile phone app .....	49
4.10.2. UC-APP-2: Change password .....	50
4.10.3. UC-APP-3: Receive plan .....	51
4.10.4. UC-APP-4: View plan .....	52
4.10.5. UC-APP-5: Report meal or activity from list .....	52
4.10.6. UC-APP-6: Report consumed meal from photograph .....	53
4.10.7. UC-APP-8: Get nutritional information from barcode .....	54
4.10.8. UC-APP-9: Logout from mobile phone app .....	55
<b>4.11. Common exception flows.....</b>	<b>55</b>
4.11.1. PE-1: Communication failure when accessing the backend .....	55
4.11.2. PE-2: Operation fails due to insufficient access rights.....	56
4.11.3. PE-3: Operation fails due to database errors .....	56
<b>5. Requirements.....</b>	<b>57</b>
<b>5.1. Dashboard.....</b>	<b>57</b>
5.1.1. Functional requirements .....	57
5.1.2. Non-functional requirements.....	61
<b>5.2. Backend .....</b>	<b>61</b>
5.2.1. Functional requirements .....	61
5.2.2. Non-functional requirements.....	65
<b>5.3. Mobile app .....</b>	<b>66</b>
5.3.1. Functional requirements .....	66
5.3.2. Non-functional requirements.....	67
<b>6. Architectural foundations.....</b>	<b>68</b>
<b>6.1. Applications .....</b>	<b>68</b>
6.1.1. Dashboard .....	68
6.1.2. Mobile application.....	69
<b>6.2. Backend components.....</b>	<b>69</b>
6.2.1. REST API.....	69
6.2.2. Databases .....	69
6.2.3. Nutrition Algorithm Service.....	69
6.2.4. Machine learning component .....	70
6.2.5. Measuring Devices .....	70
<b>6.3. Communication layer.....</b>	<b>70</b>
6.3.1. Cross streaming data processing.....	70
6.3.2. Data analysis.....	70
6.3.3. Interoperability.....	71
<b>6.4. Extensibility.....</b>	<b>71</b>
6.4.1. Product prototyping .....	71
6.4.2. Flexible business models .....	71
<b>7. Conclusion .....</b>	<b>72</b>
<b>References.....</b>	<b>73</b>
<b>Annex I – Urine analysis metabolites.....</b>	<b>74</b>
<b>Annex II – Vitamins measured in milk.....</b>	<b>75</b>



## Table of Figures

Figure 1 – D2.6's relation to other deliverables of the project .....	10
Figure 2 – A preliminary architecture of the NUTRISHIELD platform.....	68



## List of Tables

Table 1 – Gap analysis of NUTRISHIELD platform.....	13
Table 2 – System users .....	18
Table 3 – High level view of system components .....	18
Table 4 – Dashboard administration functional requirements .....	57
Table 5 – Dashboard account functional requirements.....	57
Table 6 – Dashboard Patients functional requirements.....	58
Table 7 – Dashboard measurements functional requirements.....	58
Table 8 – Dashboard Foods functional requirements .....	59
Table 9 – Dashboard nutrition algorithm functional requirements.....	59
Table 10 – Dashboard dietary plan functional requirements .....	60
Table 11 – Dashboard food diary functional requirements .....	60
Table 12 – Dashboard general functional requirements.....	61
Table 13 –Dashboard non-functional requirements .....	61
Table 14 – Backend administration functional requirements.....	62
Table 15 – Backend account functional requirements.....	62
Table 16 – Backend patient functional requirements .....	63
Table 17 – Backend measurements functional requirements .....	63
Table 18 – Backend Foods functional requirements .....	64
Table 19 – Backend nutrition algorithm functional requirements.....	64
Table 20 – Backend dietary and activity plan functional requirements.....	65
Table 21 – Backend food diary functional requirements .....	65
Table 22 – General Backend functional requirements.....	65
Table 23 – Mobile application account functional requirements .....	66
Table 24 – Mobile dietary and activity plan functional requirements.....	66
Table 25 – Mobile application food diary functional requirements .....	66
Table 26 – Mobile application nutrition information functional requirements.....	67
Table 27 – Mobile application general functional requirements.....	67
Table 28 – Mobile application non-functional requirements .....	67
Table 29 – List of metabolites and corresponding biomarkers measured in urine analysis.....	74



## Definitions, Acronyms and Abbreviations

Acronym	Title
AF	Alternative Flow
BF	Basic Flow
DoA	Description of Action
EF	Exception Flow
FFQ	Food Frequency Questionnaire
FTIR	Fourier transform infrared (spectroscopy)
HDM	Human Donor Milk
HTTP	Hypertext Transport Protocol
POPD	Protection of Personal Data
QCL	Quantum Cascade Laser
SIBO	Small Intestinal Bacterial Overgrowth
SSL	Secure Sockets Layer
STOMP	Simple Text Oriented Messaging Protocol
T2D	Type 2 Diabetes
TLS	Transport Layer Security
WP	Work Package

# 1. Introduction

This deliverable presents the NUTRISHIELD platform technical requirements and use case design.

This section presents the methodology used to define the technical requirements. It then illustrates how this deliverable relates to other deliverables in the project and concludes with a summary of what is presented in the following chapters.

## 1.1. Methodology

Requirements gathering was based on two inputs.

- Submitted deliverables** (including the project's DoA) were analysed in order to identify the NUTRISHIELD platform's technical requirements.
- Bilateral communication with partners** was carried out, either by teleconferences or e-mails to discuss the processes to be followed.

Business analysis techniques were employed to identify the system actors, components and their interactions. These were mapped to use cases, functional and non-functional requirements.

## 1.2. Relation to other Work Packages

This document is based on WP2's deliverables, WP1's first version data management plan, WP9's Protection of Personal Data deliverable and preliminary work performed on WP3. It will, in turn, be used as input to WP6 deliverables, to drive the specifications of the software framework. Figure 1 below shows this document's relation to the other deliverables of the project.

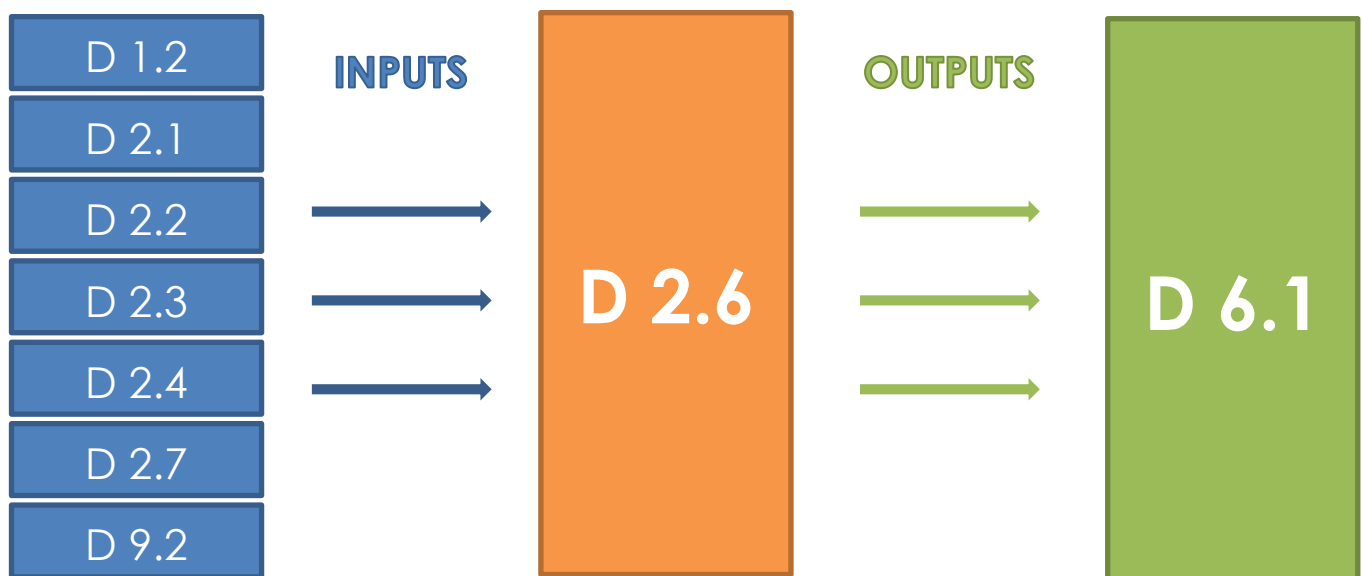


Figure 1 – D2.6's relation to other deliverables of the project



## 1.3. Document structure

The chapters in D2.6 are organised as follows.

- **Chapter 2** analyses the requirements gathered from WP1, WP2 and WP9 deliverables as well as preliminary work on WP3. It focuses on identifying any requirements stemming from the microbiome, urine, milk and breath analysis as well as their identified data needs.
- **Chapter 3** introduces the system users and components.
- **Chapter 4** documents the use cases. It presents the system users and components interactions as sequences of steps.
- **Chapter 5** presents the functional and non-functional requirements extracted from the use cases.
- **Chapter 6** presents a preliminary architecture to support the identified requirements.
- **Chapter 7** presents the conclusions of this document.



## 2. Requirements analysis

This section analyses the requirements documented in the following deliverables and how they relate to NUTRISHIELD's ambitions.

- D1.2 – Data Management Plan v1
- D2.1 – Report on requirements for microbiome analysis
- D2.2 – Report on requirements for urine analysis
- D2.3 – Report on requirements for breast milk analysis
- D2.4 – Report on requirements for breath analysis
- D2.7 – Clinical Protocols for Clinical Studies in NUTRISHIELD
- D9.2 – POPD - Requirement No.2

The analysis focuses on identifying the technical requirements that should be considered when designing the NUTRISHIELD platform. The requirements identified in the section are used in the following chapters to deduce system actors, use cases, functional and non-functional requirements.

### 2.1. Gap analysis

Section 1.4. on page 21 of DoA presents both the current state and the state beyond the state of art envisioned by NUTRISHIELD.

It elaborates on genome, microbiome, metabolome, vitamin, breath, urine and human milk analysis techniques, their relation to nutrition and monitoring in clinical validation settings.

It introduces quantum cascade laser-based measuring devices, how they are currently used and how they will be evolved through NUTRISHIELD to provide more accurate measurements to back up the analyses.

Finally, it introduces the NUTRISHIELD platform which gathers clinical measurements and laboratory analysis results, correlates them and provides personalised nutrition suggestions; and the accompanying NUTRISHIELD app to present this complex information in a user-friendly way.

The following table presents a summary gap analysis on the current and future states as envisioned by NUTRISHIELD with regards to the NUTRISHIELD app and platform.

<b>Current state</b>	<ul style="list-style-type: none"> <li>• There exist methods to measure and analyse the aforementioned biomarkers.</li> <li>• Some biomarkers are known to affect metabolism and food reactions.</li> </ul>
<b>Future state</b>	<ul style="list-style-type: none"> <li>• Novel measuring methodology will provide faster and more accurate measurements.</li> <li>• The correlation of the raw measured biomarkers or their analysed derivatives as well as other patient data will be used to offer personalised nutrition suggestions.</li> <li>• Decisions will be based on factors documented in the medical literature as well as factors identified during the project.</li> <li>• Nutrition suggestions will become available using the NUTRISHIELD platform</li> </ul>
<b>Gap description</b>	<ul style="list-style-type: none"> <li>• The NUTRISHIELD platform needs to be implemented.</li> <li>• The means to correlate and analyse the data needs to be implemented.</li> <li>• Lack of adequate patient data to support machine learning</li> </ul>



<b>Next steps and proposals</b>	<ul style="list-style-type: none"> <li>• The NUTRISHIELD platform must provide the means to input and store raw measurement data or laboratory results per patient.</li> <li>• These data will be further analysed on how they relate to nutrition. Analysis can be performed via Machine Learning and/or rule base algorithms combining a subset or all patient data. Analysis results must be used to provide nutrition suggestions.</li> <li>• The NUTRISHIELD app can be used by patients to receive the nutrition, as well as report on food habits back to the platform.</li> </ul>
---------------------------------	---

*Table 1 – Gap analysis of NUTRISHIELD platform*

The following sections present the technical requirements of the NUTRISHIELD platform per deliverable with a focus on the management of structured and unstructured streams of laboratory, sensor and environment/lifestyle data.

## 2.2. Laboratory data

Laboratory data can be raw data measured in clinical or laboratory settings or the results of raw data analysis. In the context of NUTRISHIELD the exact data that will contribute in the nutrition suggestion algorithm will be decided in task *T5.2 Optimization of NUTRISHIELD algorithm* (M32-M42) and documented in *D5.4 Report on Nutrition Suggestion Algorithms* (M46).

The following sections present the data that could contribute to the nutrition algorithm coming from preliminary work performed in WP3. The data have been partially reported as state-of-the-art in the corresponding WP2 deliverables. For sensor-based measurements please refer to section 2.3 Sensor data on page 14.

As a general requirement, the NUTRISHIELD platform should allow lab technicians to store the relevant clinical and laboratory data that will be used as inputs to the nutrition algorithm.

### 2.2.1. Microbiome analysis

*D2.1 – Report on requirements for microbiome* elaborates on a method to extract the relative abundance of the most relevant bacterial species from human milk and stool.

The laboratory analysis produces many intermediate results such as electropherograms and sets of rational numbers (derived from further analysis of electropherograms). The NUTRISHIELD platform should eventually offer the functionality to lab technicians to input the bacteria identified for a patient in a convenient way so that this information can be considered in the nutrition algorithm.

### 2.2.2. Genome analysis

*D1.2 – Data Management Plan* mentions that personalised nutrition will be based on integrating genomic profiling of patients. It elaborates on the raw input and the processed output formats and how they will be used to identify variants and correlate them with relevant disease groups. The NUTRISHIELD platform should allow uploading genomic profiles of patients. The results of the classification could be considered when generating personalised nutrition suggestions.



### 2.2.3. Urine analysis

*D2.2 - Report on requirements for urine analysis* elaborates on using traditional methods to determine urinary biomarkers (Ca, Phosphate, Urea, Na, K, Creatinine) concentrations from preterm infants. Calcium and Phosphate are reported to be related to preterm infant nutrition.

In addition, it is reported that, certain metabolites (Non-esterified fatty acids, Glycerol, Hormones, Cytokines, Pro-inflammatory markers) measured in urine predict obesity and T2D related diseases.

Preliminary work done on WP3 has produced a list of metabolites measured in urine analysis and their relations to certain biomarkers. This list is presented in *Annex I – Urine analysis metabolites* on page 74.

The NUTRISHIELD platform should be able to offer the lab technicians the functionality to input either the raw concentration of these values, or the assessment of the analysis of these values (e.g. high, normal, low). Such input could be considered when generating personalised nutrition suggestions.

### 2.2.4. Milk analysis

*D2.3 Report - Requirements for breast milk analysis* mentions several wet-chemical sample preparation techniques as well as commercial FTIR spectroscopy-based products used traditionally to analyse individual proteins, energy, fat, carbohydrate and total protein in human milk. The NUTRISHIELD platform could offer lab technicians to input the data analysed using these techniques until the sensors are implemented. These data could be used as input in the nutrition algorithm to provide nutrition suggestions to breastfeeding mothers and preterm infants.

In addition, preliminary work done in WP3 has produced a list of vitamins to be measured in milk. The list is presented in *Annex II – Vitamins measured in milk* on page 75. The measurements could be considered when generating personalised nutrition suggestions.

### 2.2.5. Breath analysis

*D2.4 Report - Requirements for breath analysis* elaborates on widely used methods for analysing exhaled breath in clinical and lab settings. These methods could be used (where available) until the sensor is implemented. Lab technicians could input raw measurements or analysed results into the NUTRISHIELD platform, which in turn could be used as input to the nutrition algorithm.

## 2.3. Sensor data

Sensor data are data that are measured from the various sensors developed in NUTRISHIELD. These sensors provide raw measurements which can be either inputted directly in NUTRISHIELD by lab technicians or analysed and input the analysis results if they are more meaningful. It is assumed that these sensors are used as novel measuring devices in a lab setting and do not have IP connectivity, thus they are not directly connected to the NUTRISHIELD platform. The information presented in the following sections comes from the corresponding WP2 deliverables.



### 2.3.1. Urine analysis

*D2.2 - Report on requirements for urine analysis* elaborates on using QCL to extract the concentration of phosphate and creatinine in urine to further assess the renal function. The NUTRISHIELD platform should be able to offer the lab technicians the functionality to input either the raw phosphate and creatinine concentrations or the renal function assessment results in a convenient way.

*D2.2 - Report on requirements for urine analysis* also elaborates on potentially using electrochemical sensors to measure a set of analytes (pH, glucose, uric acid, sodium, potassium) from urine. The NUTRISHIELD platform should be able to offer the lab technicians the functionality to input either the raw concentration of these values, or the assessment of the analysis of these values (e.g. low, medium, high, etc).

Both sets of inputs from urine analysis could be used as input in the nutrition suggestion algorithm.

### 2.3.2. Milk analysis

*D2.3 Report - Requirements for breast milk analysis* elaborates on using QCL to analyse the protein composition in human milk. The NUTRISHIELD platform should allow lab technicians to associate this information with preterm infants and their mothers (in case where HDM is not used). These data could be used as input in the nutrition suggestion algorithm for mothers and infants.

**Note:** Suggestions could be generated for preterm infants and mothers to evaluate the nutrition algorithm even though they are not to be followed given the experimental nature of the project.

### 2.3.3. Breath analysis

*D2.4 Report - Requirements for breath analysis* elaborates on using infrared absorption spectroscopy to detect concentrations of Methane and Hydrogen cyanide in exhaled breath to diagnose carbohydrate malabsorption syndromes and SIBO.

## 2.4. Environment data

Environment data are data related to the patients' environment. Such data mainly refer to food preferences, habits that affect their health, demographic and socioeconomic data and results from food questionnaires.

### 2.4.1. Urine analysis.

*D2.2 - Report on requirements for urine analysis* and *D2.7 - Clinical Protocols for Clinical Studies in NUTRISHIELD* report several environmental and pathophysiological factors that contribute to obesity (e.g. family lifestyle, medical problems, certain medications that lead to weight gain, social and economic issues, age, pregnancy, lack of sleep, quitting smoking). NUTRISHIELD could offer to store some of these factors as part of the patient profile. These factors could be used as inputs in the nutrition algorithm to provide personalised nutrition suggestions.



## 2.5. Data management

This section presents technical requirements extracted from various data management and privacy related documents.

### 2.5.1. GDPR compliance

*D9.2 – POPD - Requirement No. 2* elaborates on GDPR and data protection issues. In this context, the NUTRISHIELD platform, should only store and process anonymous personal data. Access to data should be allowed only to authorised users. Only the minimum data required should be collected. NUTRISHIELD users should have rights to information, access, rectification/deletion.

### 2.5.2. Anonymization

*D1.2 – Data Management Plan* suggests that laboratory data should be anonymised and conform to specific file formats. In particular an abstract identifier should replace any personally identifiable information.

Thus, users should be registered to the system using an abstract identifier. All personal data that relate patients to this id should be kept in a separate patient management system already installed at the clinic.

**Note:** Since no personally identifiable information is stored in NUTRISHIELD, emails will not be stored either. This means that in case users forget their passwords and cannot log into the system, there should be a contact person and a user verification procedure to reset a forgotten password.

### 2.5.3. Data formats

*D1.2 – Data Management Plan* provides information on the data formats used for uploading and processing genome, clinical and metabolic profile data files on the NUTRISHIELD platform. The NUTRISHIELD platform should be able to work with these files.

*D2.7 - Clinical Protocols for Clinical Studies in NUTRISHIELD* elaborates on the dietary assessment methodology. It mentions a Food Frequency Questionnaire (FFQ) used to calculate the MedDietScore [1]. NUTRISHIELD could provide the means to collect, store and process this information.

### 2.5.4. Machine Learning analysis

*T5.2 – Optimization of NUTRISHIELD algorithm* elaborates on combining genome analysis results, anthropometric data, clinical data, biochemical data, breath, gut microbiota markers and relevant biomarkers identified from the urinary metabolome, health parameters defined in the initial health screening to provide additional layers of personalised nutrition suggestions.

While providing nutrition suggestions backed up by rules documented in the medical literature can be considered accurate, nutrition suggestions based on machine learning algorithms require extensive testing before being adapted in real life. In addition, the accuracy in machine learning analysis solutions relies heavily on the volume of data being analysed.

One risk identified is that the number of patients who will participate in NUTRISHIELD's studies is not adequate to produce accurate results when employing machine learning techniques. Nevertheless, the NUTRISHIELD platform will implement the functionality to correlate biomarkers to nutrition using machine learning which potentially can be used to provide accurate personalised nutrition suggestions once adequate data are available.



### 3. Actors

The NUTRISHIELD system is operated by groups of users. These users interact with a set of components. Both users and components will be referred to as actors henceforth. The system actors can be grouped into two categories:

- Users, the people who interact with the platform
- Components, the software and hardware parts that make up the platform

#### 3.1. Users

The users who interact with the system components are presented in the following table. The users have been selected based on their role in the NUTRISHIELD platform.

Group	Role	Description
<b>Administration</b>	Administrator	Creates medical professional accounts, resetting passwords and overall managing system users
	Data manager	Manages the system's food, activity, nutrient, barcode and pricing data
<b>Medical professional</b>	Doctor	Manages patient records and their related data such as measurements and personalised nutrition.
	Lab Technician	Manages patient measurements and has basic access to patient data
<b>Patient</b>	Study I individual	Obesity and Diabetes Study patient (8-18 years old). The system records data from both patient and their parents/guardians. The patient or their parents/guardians interact with the NUTRISHIELD application.
	Study II individual	Lactating mother. The system records data from both mother and the infant.
	Study III individual	Study III young individual whose breath analysis results will be stored in the system

*Table 2 – System users*

#### 3.2. Components

The following table presents the components users interact with.

Component	Description
Dashboard	The web application medical professionals interact with
App	The mobile application patients interact with
Backend	The backend web service serving the dashboard and the mobile application

*Table 3 – High level view of system components*



## 4. Use cases

This section presents NUTRISHIELD's use cases. These use cases are grouped in logical entities and system processes. Each use case describes the interaction steps of actors with the system.

### 4.1. How to read this section

#### 4.1.1. The Basic flow

Each use case is presented as a sequence of **actions** carried out to achieve a goal. For example, the actions for a user to log in to the system are:

Action
Enter credentials
Press log in button

Each action is carried out by an **actor** which is also noted next to each step:

Actor	Action
User	Enters credentials
User	Presses log in button

When system **components** are involved in the sequence of steps, their actions are also listed

Actor / Component	Action
Dashboard	Displays the login screen
User	Enters credentials
User	Presses log in button
Dashboard	Authenticates user
Dashboard	Displays main dashboard

The list of actions that achieves our goal is called **the basic flow**. This flow assumes that everything is going well, e.g. correct credentials are given, the users backend database is up and running, etc.

The basic flow sequence of actions are titled as **Basic Flow** and **each action is assigned an ordinal prefixed with BF-** (e.g. BF-1, BF-2, etc) to indicate that this is a basic flow action.

Basic Flow		
ID	Actor / Component	Action
BF-1	Dashboard	Presents login screen
BF-2	User	Enters credentials
BF-3	User	Presses log in button
BF-4	Dashboard	Authenticates user
BF-5	Dashboard	Displays main dashboard



#### 4.1.2. Alternative and Exception flows

A **Notes & References** column might present additional comments on the actions. It can also be used to refer to other use cases, **alternative flows** or **exception flows** which handle alternative course of actions for the same use case (e.g. when invalid user credentials have been entered or the network is down).

Basic Flow			
ID	Actor / Component	Action	Notes & References
BF-1	Dashboard	Presents login screen	
BF-2	User	Enters credentials	
BF-3	User	Presses log in button	EF-A: Network error
BF-4	Dashboard	Authenticates user	AF-A: Authentication error
BF-5	Dashboard	Displays main dashboard	

Alternative flows are actions taken when an alternative branch of actions need to be taken to handle the use case. They are presented in a similar way below the basic flow.

AF-A: Authentication error			
ID	Actor / Component	Action	Notes & References
AF-A-1	Dashboard	Informs the user that invalid credentials have been entered	
AF-A-2		[Continue with BF-2]	

Exception flows are actions taken when the use case cannot be completed (e.g. the network is down)

EF-A: Network error			
ID	Actor / Component	Action	Notes & References
EF-A-1	Dashboard	Informs the user about the connectivity problem and suggests trying again later	
EF-A-2		[End use case]	

**Note:** Some exception flows are common to many use cases (e.g. action taken when database errors occur, or network connectivity is down). To avoid repetition, common exception flows are presented in section 4.11 *Common exception flows* on page 55. Their id prefixes are **PE-** which stands for platform error.

#### 4.1.3. Use case summary

Apart from its actions, a use case has an **Id**, a **description**, one or more **preconditions** and **postconditions**, and a **trigger** that starts it. These, along with the participating **actors** and **components**, are documented in a block before the basic flow as shown below.

<b>Description</b>	This use case describes the process by which a user logs into the system
<b>Actors</b>	User
<b>Components</b>	Dashboard
<b>Preconditions</b>	The user is already registered on the system
<b>Postconditions</b>	The user successfully logs in
<b>Triggers</b>	The user wants to access functionality that requires him to be logged in

The following sections follow this pattern to present NUTRISHIELD's use cases.



## 4.2. Administration

This section presents the use cases related to system and user administration requirements. These are actions performed by the system administrator actor.

### 4.2.1. UC-ADMIN-1: View medical professionals list

<b>Description</b>	This use case describes the process by which the system administrator views a list of medical professional users already registered in the system
<b>Actors</b>	Administrator
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The administrator is already logged into the system
<b>Postconditions</b>	A list of medical professional users is displayed on the dashboard
<b>Triggers</b>	The administrator wants to view all medical professional users in the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Administrator	Selects the <i>Users</i> option	
BF-2	Dashboard	Requests all users from the backend	PE-1: Communication error
BF-3	Backend	Retrieves all users	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Presents a list of users	

### 4.2.2. UC-ADMIN-2: Create health professional user

<b>Description</b>	This use case describes the process by which the system administrator creates a user account for a health professional
<b>Actors</b>	Administrator Health professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The administrator is already logged into the system
<b>Postconditions</b>	A new medical professional user is created in the system
<b>Triggers</b>	A medical professional wants to register to the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Administrator	Selects the Add option	Continue from UC-ADMIN-1
BF-2	Dashboard	Presents a view to fill in the new user details	
BF-3	Administrator	Fills in the user details and presses save	
BF-4	Dashboard	Sends the new user request to the backend	PE-1: Communication error
BF-5	Backend	Generates a unique username and a random first-time password which needs to be changed upon the first login, saves the user and returns the credentials to the dashboard	PE-2: Insufficient rights PE-3: Database error



BF-6	Dashboard	Presents the username and password on the screen indicating that the password will need to change upon the next login	
BF-7	Administrator	Hands the details to the medical professional	e.g. print or e-mail

#### 4.2.3. UC-ADMIN-3: View medical professional details

<b>Description</b>	This use case describes the process by which the system administrator views a medical professional user detail		
<b>Actors</b>	Administrator		
<b>Components</b>	Dashboard Backend		
<b>Preconditions</b>	The administrator is already logged into the system		
<b>Postconditions</b>	The medical professional details are presented on the dashboard		
<b>Triggers</b>	The administrator wants to view a medical professional user's details		

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Administrator	Presses on a user list item from the list	Continue from UC-ADMIN-1
BF-2	Dashboard	Requests user details from the backend	PE-1: Communication error
BF-3	Backend	Retrieves requested user details	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Presents the selected user's details	

#### 4.2.4. UC-ADMIN-4: Edit medical professional details

<b>Description</b>	This use case describes the process by which the system administrator edits a medical professional user's details		
<b>Actors</b>	Administrator		
<b>Components</b>	Dashboard Backend		
<b>Preconditions</b>	The administrator is already logged into the system		
<b>Postconditions</b>	The medical professional user's details are updated		
<b>Triggers</b>	The administrator wants to edit the medical professional user's details on the platform		

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Administrator	Presses the edit button	Continue from UC-ADMIN-3
BF-2	Dashboard	Enters edit mode	
BF-3	Administrator	Edits the details that need to be edited	e.g. password
BF-4	Administrator	Presses the save button	AF-A: Cancel editing
BF-5	Dashboard	Sends updated details	PE-1: Communication error
BF-6	Backend	Persists updated details	PE-2: Insufficient rights PE-3: Database error
BF-7	Dashboard	Indicates success	e.g. Show a toast
BF-8	Dashboard	Exits edit mode	



AF-A: Cancel editing			
ID	Actor / Component	Action	Notes & References
AF-A-1	Administrator	Presses the cancel button	
AF-A-2	Dashboard	Undoes any changes	
AF-A-3		[Continue with BF-8]	i.e. exit edit mode

#### 4.2.5. UC-ADMIN-5: Delete medical professional

<b>Description</b>	This use case describes the process by which the system administrator deletes a medical professional user from the system
<b>Actors</b>	Administrator
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The administrator is already logged into the system
<b>Postconditions</b>	The medical professional user is deleted from the system
<b>Triggers</b>	The administrator wants to delete a medical professional user from the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Administrator	Presses the delete button	Continue from UC-ADMIN-3
BF-2	Dashboard	Asks for confirmation	
BF-3	Administrator	Confirms deletion	AF-A: Cancel delete
BF-4	Dashboard	Sends delete request to platform	PE-1: Communication error
BF-5	Backend	Deletes medical professional	PE-2: Insufficient rights PE-3: Database error
BF-6	Dashboard	Indicates success	
BF-7	Dashboard	Navigates to users list	See UC-ADMIN-1

AF-A: Cancel delete			
ID	Actor / Component	Action	Notes & References
AF-A-1	Administrator	Presses the cancel button	
AF-A-2		[End use case]	

## 4.3. Account

This section presents all the use cases related to dashboard user account requirements. These are account related actions carried out by administrators and medical professionals from the dashboard web application.

#### 4.3.1. UC-ACCOUNT-1: Login

<b>Description</b>	This use case describes the process by which the system administrator or medical professional logs into the system
<b>Actors</b>	Administrator Medical professional
<b>Components</b>	Dashboard



	Backend
<b>Preconditions</b>	An account exist for the user who tries to log in
<b>Postconditions</b>	The user successfully logs into the system
<b>Triggers</b>	The user wants to use the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Administrator or Medical professional	Enters credentials	Assume a login screen
BF-2	Administrator or Medical professional	Presses login	
BF-3	Dashboard	Sends credentials to backend	PE-1: Communication error
BF-4	Backend	Authenticates user	AF-A: Invalid credentials PE-3: Database error
BF-5	Dashboard	Navigates to main dashboard page	

AF-A: Invalid credentials			
ID	Actor / Component	Action	Notes & References
AF-A-1	Backend	If the user has provided invalid credentials, the Backend returns an error code	Could be a standard HTTP code
AF-A-2	Dashboard	Presents an indicative message and prompts the user to try again	
AF-A-3		[Continue with BF-1]	

#### 4.3.2. UC-ACCOUNT-2: Change password

<b>Description</b>	This use case describes the process by which the system administrator or medical professional change their account password
<b>Actors</b>	Administrator Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The user is already logged in
<b>Postconditions</b>	The user successfully changes his password
<b>Triggers</b>	The user wants to change his password

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Administrator or Medical professional	Selects the account option from the dashboard	
BF-2	Dashboard	Displays the account options	Could be a dedicated page or options menu
BF-3	Administrator or Medical professional	Selects the change password option	
BF-4	Dashboard	Prompts for current, new and confirmation of new password	



BF-5	Administrator or Medical professional	Types current, new and confirmation of new password	
BF-6	Administrator or Medical professional	Presses the change button	AF-A: Invalid password
BF-7	Dashboard	Sends the password change request to the Backend	PE-1: Communication error
BF-8	Backend	Updates the current user's password	AF-B: Invalid password PE-3: Database error
BF-9	Dashboard	Indicates success to the user	

**AF-A: Invalid password**

ID	Actor / Component	Action	Notes & References
AF-A-1	Dashboard	If the user has provided a new password that does not meet certain restrictions e.g. minimum length, complexity, etc, or the password confirmation does not match, an indicative message is shown which prompts the user to try again	
AF-A-3		[Continue with BF-5]	

**AF-B: Invalid password**

ID	Actor / Component	Action	Notes & References
AF-B-1	Backend	If the user has provided a new password that does not meet certain restrictions e.g. minimum length, complexity, etc or the current password provided does not match the current user's password, the backend returns an error code	Could be a standard HTTP code
AF-B-2	Dashboard	Presents an indicative message and prompts the user to try again	
AF-B-3		[Continue with BF-5]	

**4.3.3. UC-ACCOUNT-3: Logout**

<b>Description</b>	This use case describes the process by which the system administrator or medical professional log out of the system
<b>Actors</b>	Administrator Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The user is already logged in
<b>Postconditions</b>	The user successfully logs out of the system
<b>Triggers</b>	The user wants to log out of the system

**Basic flow**

ID	Actor / Component	Action	Notes & References
BF-1	Administrator or Medical professional	Selects the log out option from the dashboard	



BF-2	Dashboard	Sends the log out request to the backend	PE-1: Communication error
BF-3	Backend	Logs the user out of the system	
BF-4	Dashboard	Navigates to the log in page	

#### 4.3.4. UC-ACCOUNT-4: Impersonate

<b>Description</b>	This use case describes the process by which the system administrator can impersonate a medical professional user to perform actions on the medical professional's behalf
<b>Actors</b>	Administrator
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The medical professional has asked the administrator to perform an action on their behalf The administrator is already logged in
<b>Postconditions</b>	The administrator successfully impersonates the medical professional user
<b>Triggers</b>	The administrator wants to perform actions on medical professional's behalf

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Administrator	Selects the impersonate option	Continue from UC-ADMIN-3
BF-2	Dashboard	Sends the impersonation request to the backend	PE-1: Communication error
BF-3	Backend	Impersonates the requested user	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Navigates to main dashboard page as impersonated user	

#### 4.3.5. UC-ACCOUNT-5: Set nickname

<b>Description</b>	This use case describes the process by which medical professionals set their nickname from the dashboard
<b>Actors</b>	Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The user is logged in
<b>Postconditions</b>	The user's nickname changes
<b>Triggers</b>	The user wants to change his/her nickname

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Selects the <i>Change nickname</i> option from the <i>Account options</i>	<i>Account options</i> could be a dedicated page or menu
BF-2	Dashboard	Provides a form to fill in the new nickname	
BF-3	Medical professional	Fills in a new nickname	AF-A: Autogenerate
BF-4	Medical professional	Presses the <i>Save</i> button	AF-B: Cancel change
BF-5	Dashboard	Sends new nickname to backend	PE-1: Communication error
BF-6	Backend	Saves the new nickname to the database	AF-C: Nickname exists PE-2: Database error



BF-7	Dashboard	Indicates success	
------	-----------	-------------------	--

AF-A: Autogenerate			
ID	Actor / Component	Action	Notes & References
AF-A-1	Medical personnel	Presses the autogenerate button	
AF-A-2	Dashboard	Fills in the nickname field with a random auto generated nickname	Codenamize <sup>1</sup> could be used for it

AF-B: Cancel change			
ID	Actor / Component	Action	Notes & References
AF-B-1	Medical personnel	Presses the <i>Cancel</i> button	
AF-B-2		[End use case]	

AF-C: Nickname exists			
ID	Actor / Component	Action	Notes & References
AF-C-1	Backend	In case the nickname has already been taken, the dashboard returns an error code	Could be a standard HTTP status code
AF-C-2	Dashboard	Provides the user with an indicative message and asks to provide a different name, optionally offering suggestions based on the initially provided name	
AF-C-3		[Continue at BF-3]	

## 4.4. Patients

This section presents use cases related to patient's management. The actions are carried out by doctors and cover managing patient entities and assigning other doctors and/or lab technicians responsible for each patient.

### 4.4.1. UC-PATIENTS-1: View patients list

<b>Description</b>	This use case describes the process by which medical professional view a list of patients assigned to them
<b>Actors</b>	Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The user is logged in
<b>Postconditions</b>	The user views a list of patients assigned to them
<b>Triggers</b>	The user wants to view patients assigned to them

<sup>1</sup> Codenamize is a JavaScript library which generates random names. For example: *happy\_noyce*, *sharp\_blackwell*, *admiring\_bell*. At the time of writing, it is available on this URL <https://github.com/stemail23/codenamize-js>



Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Selects the Patients option	
BF-2	Dashboard	Requests the patients list from the backend	PE-1: Communication error
BF-3	Backend	Retrieves the patients list for the current user	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Displays the user's list of patients	

#### 4.4.2. UC-PATIENTS-2: Create patient

<b>Description</b>	This use case describes the process by which the doctor registers a new patient in the system
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is already logged into the system
<b>Postconditions</b>	The patient is registered in the system
<b>Triggers</b>	The doctor wants to register a new patient in the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Selects the Add option	Continue from UC-PATIENTS-1
BF-2	Dashboard	Presents a view to fill in the new user details	
BF-3	Doctor	Fills in the patient details and presses save	
BF-4	Dashboard	Sends the new patient request to the backend	PE-1: Communication error
BF-5	Backend	Generates a unique username and a random first-time password which needs to be changed upon the first login, saves the user and returns the credentials to the dashboard	PE-2: Insufficient rights PE-3: Database error
BF-6	Dashboard	Presents the username and password on the screen indicating that the password will need to change upon the next login	
BF-7	Doctor	Hands the credentials to the patient	e.g. print or e-mail

#### 4.4.3. UC-PATIENTS-3: View patient details

<b>Description</b>	This use case describes the process by which medical professionals view patient details
<b>Actors</b>	Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The medical professional is already logged in
<b>Postconditions</b>	The patient details are presented on the dashboard
<b>Triggers</b>	The medical professional wants to view a patient user's details

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Presses on a patient list item from the list	Continue from UC-PATIENTS-1
BF-2	Dashboard	Requests patient details from the backend	PE-1: Communication error



BF-3	Backend	Retrieves the requested patient details	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Presents the selected patient's details	

#### 4.4.4. UC-PATIENTS-4: Edit patient

<b>Description</b>	This use case describes the process by which the doctor edits a patient's details. Part of this use case describes the optional process by which the doctor assigns other medical personnel to a patient so that other medical personnel can work with these patients		
<b>Actors</b>	Doctor		
<b>Components</b>	Dashboard Backend		
<b>Preconditions</b>	The doctor is already logged into the system		
<b>Postconditions</b>	The patient user's details are updated		
<b>Triggers</b>	The doctor wants to edit the patient user's details on the backend		

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the edit button	Continue from UC-PATIENTS-3
BF-2	Dashboard	Enters edit mode	
BF-3	Doctor	Edits the details that need to be edited	AF-A: Assign medical personnel
BF-4	Doctor	Presses the save button	AF-B: Cancel editing
BF-5	Dashboard	Sends updated details	PE-1: Communication error
BF-6	Backend	Persists updated details	PE-2: Insufficient rights PE-3: Database error
BF-7	Dashboard	Indicates success	e.g. Show a toast
BF-8	Dashboard	Exits edit mode	

AF-A: Assign medical personnel			
ID	Actor / Component	Action	Notes & References
AF-B-1	Doctor	Selects the medical personnel users that will be able to see the currently edited user in their list of users	Refer to UC-PATIENTS-1
AF-B-3		[Continue with BF-4]	

AF-B: Cancel editing			
ID	Actor / Component	Action	Notes & References
AF-B-1	Doctor	Presses the cancel button	
AF-B-2	Dashboard	Undoes any changes	
AF-B-3		[Continue with BF-8]	i.e. exit edit mode

#### 4.4.5. UC-PATIENTS-5: Delete patient

<b>Description</b>	This use case describes the process by which the doctor deletes a patient user from the system		
<b>Actors</b>	Doctor		
<b>Components</b>	Dashboard Backend		



<b>Preconditions</b>	The doctor is already logged into the system
<b>Postconditions</b>	The patient user is deleted from the system
<b>Triggers</b>	The doctor wants to delete a patient user from the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the delete button	Continue from UC-PATIENTS-3
BF-2	Dashboard	Asks for confirmation	
BF-3	Doctor	Confirms deletion	AF-A: Cancel delete
BF-4	Dashboard	Sends delete request to backend	PE-1: Communication error
BF-5	Backend	Deletes patient	PE-2: Insufficient rights PE-3: Database error
BF-6	Dashboard	Indicates success	
BF-7	Dashboard	Navigates to users list	See UC-ADMIN-1

AF-A: Cancel delete			
ID	Actor / Component	Action	Notes & References
AF-A-1	Doctor	Presses the cancel button	
AF-A-2		[End use case]	

## 4.5. Measurements

This section presents the use cases related to managing patient measurements. The actions are performed mainly by lab technicians and in some cases doctors. By measurements we refer to any data associated with the patient. These data could be as simple as anthropometric (weight, height, age) or as complex as the relative abundance of bacteria identified in the patient's gut.

### 4.5.1. UC-MSR-1: View patient measurements list

<b>Description</b>	This use case describes the process by which a medical professional view a list of a patient's measurements
<b>Actors</b>	Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The medical professional is logged in
<b>Postconditions</b>	The medical professional views a list of measurements for a patient
<b>Triggers</b>	The user wants to view a patient's measurements list

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Selects the measurements option	Continue from UC-PATIENTS-3
BF-2	Dashboard	Requests the patient's measurements list from the backend	PE-1: Communication error
BF-3	Backend	Retrieves the patient's measurement list for the current user	PE-2: Insufficient rights PE-3: Database error



BF-4	Dashboard	Displays the patient's measurements list	
------	-----------	--	--

#### 4.5.2. UC-MSR-2: Add patient measurements

<b>Description</b>	This use case describes the process by which a medical professional can add new measurements for a patient either manually or from a file
<b>Actors</b>	Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The medical professional is logged in
<b>Postconditions</b>	The medical professional adds a new set of measurements for a patient
<b>Triggers</b>	The medical professional wants to add new measurements for a patient

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Selects the add measurement option	Continue from UC-MSR-1
BF-2	Dashboard	Presents a list of measurement type options to choose from	e.g. <i>Basic</i> to record weight, etc or certain file types to import from
BF-3	Medical professional	Selects an option	AF-A: Cancel adding
BF-4	Dashboard	Displays a form to fill in the measurements depending on the type of measurements selected	AF-A: Cancel adding AF-B: Add from file
BF-5	Medical professional	Fills in fields and press save	AF-A: Cancel adding
BF-6	Dashboard	Sends new measurements to backend	PE-1: Communication error
BF-7	Backend	Persists new measurements	PE-2: Insufficient rights PE-3: Database error EF-A: Invalid file type
BF-8	Dashboard	Closes the form and indicate success	
BF-9	Dashboard	Updates the displayed list of patient measurements with the new entry	

AF-A: Cancel adding			
ID	Actor / Component	Action	Notes & References
AF-A-1	Medical professional	Presses the cancel button or click to hide popup menu	
AF-A-2		[End use case]	

AF-B: Add from file			
ID	Actor / Component	Action	Notes & References
AF-B-1	Dashboard	Displays a file selection dialog to choose a file from	AF-A: Cancel adding
AF-B-2	Medical professional	Selects the file and press save	AF-A: Cancel adding
AF-B-3		[Continue with BF-6]	

EF-A: Invalid file type			
ID	Actor / Component	Action	Notes & References



EF-A-1	Backend	If checking the file for consistency fails, the backend returns an error code	Could be a standard HTTP code
EF-A-2	Dashboard	Displays an indicative message asking the user to try uploading the file again	
EF-A-3		[End use case]	

#### 4.5.3. UC-MSR-3: View patient measurement details

<b>Description</b>	This use case describes the process by which a medical professional can view the details of a measurement either on the dashboard or by downloading the associated measurements file to view in a dedicated 3 <sup>rd</sup> party application		
<b>Actors</b>	Medical professional		
<b>Components</b>	Dashboard Backend		
<b>Preconditions</b>	The medical professional is logged in		
<b>Postconditions</b>	The medical professional views or downloads a file of the measurement details		
<b>Triggers</b>	The medical professional wants to view the details of a specific measurement		

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Selects a list item from the measurements list	Continue from UC-MSR-1
BF-2	Dashboard	Requests the measurements recorded for the selected item from the backend	PE-1: Communication error
BF-3	Backend	Retrieves measurements for selected item	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Displays the recorded measurements	AF-A: Download file

AF-A: Download file			
ID	Actor / Component	Action	Notes & References
AF-A-1	Dashboard	In case the measurement consists of a file, the dashboard displays the available file details providing the option to download it	Details might be file name, size, date, uploader
AF-A-2	Medical professional	Presses on download	
AF-A-3	Dashboard	Requests to download file	PE-1: Communication error
AF-A-4	Backend	Retrieves file to download	PE-2: Insufficient rights PE-3: Database error
AF-A-5	Dashboard	Download file to user's workstation (at predefined or user defined location)	

#### 4.5.4. UC-MSR-4: Edit patient measurements

<b>Description</b>	This use case describes the process by which a medical professional edits the details of an existing measurement either by filling in fields on the dashboard or by uploading a new measurements file		
<b>Actors</b>	Medical professional		
<b>Components</b>	Dashboard		



	Backend
<b>Preconditions</b>	The medical professional is logged in
<b>Postconditions</b>	The medical professional updates the details of a specific measurement
<b>Triggers</b>	The medical professional wants to update the details of a specific measurement

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Presses the edit button	Continue from UC-MSR-3 AF-A: Replace measurements file
BF-2	Dashboard	Enters edit mode	
BF-3	Medical professional	Edits the details that need to be edited	
BF-4	Medical professional	Presses the save button	AF-B: Cancel editing
BF-5	Dashboard	Sends updated details	PE-1: Communication error
BF-6	Backend	Persists updated details	PE-2: Insufficient rights PE-3: Database error
BF-7	Dashboard	Indicates success	e.g. Show a toast
BF-8	Dashboard	Exits edit mode	

AF-A: Replace measurements file			
ID	Actor / Component	Action	Notes & References
AF-A-1	Medical professional	In case the measurements are stored in a file, the medical personnel is offered a Replace file option to select instead of editing.	
AF-A-2	Dashboard	Prompts the user to select a new file	
AF-A-3	Medical professional	Selects the file to upload	
AF-A-4	Dashboard	Sends the new file to the backend	PE-1: Communication error
AF-A-5	Backend	Persists the new file	PE-2: Insufficient rights PE-3: Database error AF-C-1: Invalid file type

AF-B: Cancel editing			
ID	Actor / Component	Action	Notes & References
AF-B-1	Medical professional	Presses the cancel button	
AF-B-2	Dashboard	Undoes any changes	
AF-B-3		[Continue with BF-8]	i.e. exit edit mode

AF-C: Invalid file type			
ID	Actor / Component	Action	Notes & References
AF-C-1	Backend	If checking the file for consistency fails, the backend returns an error code	Could be a standard HTTP code
AF-C-2	Dashboard	Displays an indicative message asking the user to try uploading the file again	
AF-C-3		[Continue with AF-B-1]	i.e. exit edit mode



#### 4.5.5. UC-MSR-5: Delete patient measurements

<b>Description</b>	This use case describes the process by which medical professionals delete patient measurements from the system
<b>Actors</b>	Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The medical professional is already logged into the system
<b>Postconditions</b>	The patient's measurement record is deleted from the system
<b>Triggers</b>	The medical professional wants to delete a patients measurement from the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Presses the delete button	Continue from UC-MSR-3
BF-2	Dashboard	Asks for confirmation	
BF-3	Medical professional	Confirms deletion	AF-A: Cancel delete
BF-4	Dashboard	Sends delete request to backend	PE-1: Communication error
BF-5	Backend	Deletes measurement	PE-2: Insufficient rights PE-3: Database error
BF-6	Dashboard	Indicates success	
BF-7	Dashboard	Navigates to measurements list	See UC-MSR-1

AF-A: Cancel delete			
ID	Actor / Component	Action	Notes & References
AF-A-1	Doctor	Presses the cancel button	
AF-A-2		[End use case]	

#### 4.5.6. UC-MSR-6: Associate QR code with patient

<b>Description</b>	This use case describes the process by which a medical professional associates a pre-generated QR code label, stuck on a sample container (e.g. vial), with a system patient by scanning the code using a laptop or PC camera
<b>Actors</b>	Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The medical professional is already logged into the system The medical professional has a sample container with a QR code on it The medical professional knows the patient that will be associated with the sample container
<b>Postconditions</b>	The sample container is associated with the patient
<b>Triggers</b>	The medical professional wants to associate a sample container with a patient

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Presses the <i>Associate sample container</i> button	Continue from UC-PATIENTS-3



BF-2	Dashboard	Presents a dialog showing input from the PC/laptop/mobile camera	EF-A: No camera
BF-3	Medical professional	Positions the QR code in front of the PC/laptop/mobile camera and presses the <i>OK</i> button to capture the image	AF-A: Cancel capture
BF-4	Dashboard	Captures the image and send it to back end	PE-1: Communication error
BF-5	Backend	Reads the QR code from the image and associates it with the patient	AF-B: Invalid QR code PE-2: Insufficient rights PE-3: Database error
BF-6	Dashboard	Indicates success	
BF-7	Dashboard	Closes the camera capture dialog	

**AF-A: Cancel capture**

ID	Actor / Component	Action	Notes & References
AF-A-1	Medical professional	Presses the cancel button	
AF-A-2		[Continue with BF-7]	

**AF-B: Invalid QR code**

ID	Actor / Component	Action	Notes & References
AF-B-1	Backend	In case the uploaded QR code image cannot be parsed or the QR code has already been assigned with a patient, the backend returns an error code to the dashboard	Could be a standard HTTP code (e.g. 400, and 409 respectively)
AF-B-2	Dashboard	Presents an indicative error message to the user and prompts to try again	
AF-B-3		[Continue with BF-2]	

**EF-A: No camera**

ID	Actor / Component	Action	Notes & References
EF-A-1	Dashboard	In case the web application cannot access the devices camera, either because it does not exist, or the user denied access to the camera for this web application, the user is notified with an indicative error message	
EF-A-2		[End use case]	

#### 4.5.7. UC-MSR-7: Add patient measurements from QR code

<b>Description</b>	This use case describes the process by which a medical professional can find a patient on the dashboard to start filling in measurements by scanning a pre-generated QR code label stuck on the sample container.
<b>Actors</b>	Medical professional
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The medical professional is already logged into the system The medical professional has a sample container with a QR code on it The container has already been associated with a patient (see UC-MSR-7)



<b>Postconditions</b>	The medical professional finds the patient and starts filling in the measurements
<b>Triggers</b>	The medical professional wants to fill in the measurements for a patient by scanning a QR code labelled sample container

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Medical professional	Presses the <i>Find from AR code</i> button	Continue from UC-PATIENTS-1
BF-2	Dashboard	Presents a dialog showing input from the PC/laptop/mobile camera	EF-A: No camera
BF-3	Medical professional	Positions the QR code in front of the PC/laptop/mobile camera and presses the <i>OK</i> button to capture the image	AF-A: Cancel capture
BF-4	Dashboard	Captures the image and send it to back end	PE-1: Communication error
BF-5	Backend	Reads the QR code from the image and returns the associated patient	AF-B: Invalid QR code PE-2: Insufficient rights PE-3: Database error
BF-6	Dashboard	Navigates to patient screen	
BF-7	Dashboard	[Continue with steps in UC-MSR-2]	See UC-MSR-2

AF-A: Cancel capture			
ID	Actor / Component	Action	Notes & References
AF-A-1	Medical professional	Presses the cancel button	
AF-A-2		[Continue with BF-7]	

AF-B: Invalid QR code			
ID	Actor / Component	Action	Notes & References
AF-B-1	Backend	In case the uploaded QR code image cannot be parsed or the QR code has not been associated to a patient, the backend returns an error code to the dashboard	Could be a standard HTTP code (e.g. 400, and 404 respectively)
AF-B-2	Dashboard	Presents an indicative error message to the user and prompts to try again (if necessary)	
AF-B-3		[Continue with BF-2]	

EF-A: No camera			
ID	Actor / Component	Action	Notes & References
EF-A-1	Dashboard	In case the web application cannot access the devices camera, either because it does not exist, or the user denied access to the camera for this web application, the user is notified with an indicative error message	
EF-A-2		[End use case]	



## 4.6. Foods

This section presents the actions related to data management of various auxiliary entities used in the context of NUTRISHIELD. Namely, foods, nutrients, barcodes, activities and market prices. These actions are carried out by a data manager who is responsible for maintaining these datasets. The following use cases present food related interactions. Physical activity interactions follow a similar approach but are omitted for brevity.

### 4.6.1. UC-FOOD-1: View list of foods

<b>Description</b>	This use case describes the process by which the data manager views a list of the available foods.
<b>Actors</b>	Data manager
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The data manager is logged in to the system
<b>Postconditions</b>	The foods list is displayed on the dashboard
<b>Triggers</b>	The data manager wants to view the list of available foods

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Data manager	Selects the Foods option	
BF-2	Dashboard	Presents the list of available foods	

### 4.6.2. UC-FOOD-2: Add food

<b>Description</b>	This use case describes the process by which the data manager adds new food in the list of foods. Added food details might contain food description, nutrients, ingredients, service sizes, barcode, category, price, etc
<b>Actors</b>	Data manager
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The data manager is logged in to the system
<b>Postconditions</b>	The new food is added in the backend
<b>Triggers</b>	The data manager wants to add information about new food to the backend

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Data manager	Presses the add button	Continue from UC-FOOD-1
BF-2	Dashboard	Provides a form to fill in the new food details	AF-A: Upload from file(s)
BF-3	Data manager	Fills in food information	AF-B: Cancel adding
BF-4	Data manager	Presses save	
BF-5	Dashboard	Sends the new food to the backend	PE-1: Communication error
BF-6	Backend	Stores the new food data	PE-2: Insufficient rights PE-3: Database error



BF-7	Dashboard	Indicates success	
BF-8	Dashboard	Displays list of current food items	See UD-FOOD-1

AF-A: Upload from file			
ID	Actor / Component	Action	Notes & References
AF-A-1	Dashboard	Prompts for file(s) to upload	AF-B: Cancel adding
AF-A-2	Data manager	Selects food data file(s) to upload	
AF-A-3		[Continue with BF-4]	

AF-B: Cancel add			
ID	Actor / Component	Action	Notes & References
AF-B-1	Data manager	Presses the cancel button	
AF-B-2		[Continue at BF-8]	

#### 4.6.3. UC-FOOD-3: View food details

<b>Description</b>	This use case describes the process by which the data manager can view details about a specific food from the list of foods
<b>Actors</b>	Data manager
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The data manager is already logged in
<b>Postconditions</b>	The food details are presented on the dashboard
<b>Triggers</b>	The data manager wants to view a food's details

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Data manager	Presses on a food item from the list	Continue from UC-FOOD -1
BF-2	Dashboard	Requests food details from the backend	PE-1: Communication error
BF-3	Backend	Retrieves the requested food's details	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Presents the selected food details	

#### 4.6.4. UC-FOOD-4: Edit food

<b>Description</b>	This use case describes the process by which the data manager edits food details. Details contain description, nutrients, barcodes and indicative price in case of branded products, serving size, etc.
<b>Actors</b>	Data manager
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The data manager is already logged into the system
<b>Postconditions</b>	The food details are updated
<b>Triggers</b>	The data manager wants to edit food details on the backend

Basic flow			
------------	--	--	--



ID	Actor / Component	Action	Notes & References
BF-1	Data manager	Presses the edit button	Continue from UC-FOOD-3
BF-2	Dashboard	Enters edit mode	
BF-3	Data manager	Edits the details that need to be edited	
BF-4	Data manager	Presses the save button	AF-A: Cancel editing
BF-5	Dashboard	Sends updated details	PE-1: Communication error
BF-6	Backend	Persists updated details	PE-2: Insufficient rights PE-3: Database error
BF-7	Dashboard	Indicates success	
BF-8	Dashboard	Exits edit mode	

AF-A: Cancel editing			
ID	Actor / Component	Action	Notes & References
AF-B-1	Medical professional	Presses the cancel button	
AF-B-2	Dashboard	Undoes any changes	
AF-B-3		[Continue with BF-8]	i.e. exit edit mode

#### 4.6.5. UC-FOOD-5: Delete food

<b>Description</b>	This use case describes the process by which the data manager deletes food data from the system
<b>Actors</b>	Data manager
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The data manager is already logged into the system
<b>Postconditions</b>	The food is deleted from the system
<b>Triggers</b>	The data manager wants to delete food from the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Data manager	Presses the delete button	Continue from UC-FOOD-3
BF-2	Dashboard	Asks for confirmation	
BF-3	Data manager	Confirms deletion	AF-A: Cancel delete
BF-4	Dashboard	Sends delete request to backend	PE-1: Communication error
BF-5	Backend	Deletes food	PE-2: Insufficient rights PE-3: Database error
BF-6	Dashboard	Indicates success	
BF-7	Dashboard	Navigates to foods list	See UC-FOOD-1

AF-A: Cancel delete			
ID	Actor / Component	Action	Notes & References
AF-A-1	Data manager	Presses the cancel button	
AF-A-2		[End use case]	



## 4.7. Nutrition algorithm

### 4.7.1. UC-NUTR-1: View suggestions

<b>Description</b>	This use case describes the process by which the doctor views the nutrition and activity suggestions which have already been generated for a specific patient
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is logged in to the system
<b>Postconditions</b>	The dashboard displays a list of nutrition and activity suggestions
<b>Triggers</b>	The doctor wants to view a list of generated nutrition and activity suggestions for a patient

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the Suggestions options	Continue from UC-PATIENTS-3
BF-2	Dashboard	Requests the nutrition suggestions for the specified patient	PE-1: Communication error
BF-3	Backend	Retrieves the existing suggestions for the specified patient	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Displays the existing suggestions list for the specified patient	

### 4.7.2. UC-NUTR-2: Generate suggestion

<b>Description</b>	This use case describes the process by which the doctor uses NUTRISHIELD to generate nutrition and activity suggestions for a patient.
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is logged in to the system The patient has the required measurements added to the system
<b>Postconditions</b>	The backend generates nutrition and activity suggestions
<b>Triggers</b>	The doctor wants to generate nutrition and activity suggestions for a patient

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the add button	Continue from UC-NUTR-1
BF-2	Dashboard	Sends the request to the backend	PE-1: Communication error
BF-3	Backend	Works out the suggestions for the specific patient considering available patient data, persists it and returns it to the dashboard	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Adds the new suggestion to the list of suggestions of the currently viewed patient	



BF-5	Dashboard	Displays the suggestion details	See UC-NUTR-3
------	-----------	---------------------------------	---------------

#### 4.7.3. UC-NUTR-3: View suggestion details

<b>Description</b>	This use case describes the process by which the doctor can view details about a suggestion generated for a patient
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is already logged in
<b>Postconditions</b>	The nutrition and activity suggestion details are presented on the dashboard
<b>Triggers</b>	The doctor wants to view a nutrition and activity suggestion's details

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses on a suggestion item from the list	Continue from UC-NUTR-1
BF-2	Dashboard	Requests the suggestion details from the backend	PE-1: Communication error
BF-3	Backend	Retrieves the requested suggestion's details	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Presents the selected suggestion's details	

#### 4.7.4. UC-NUTR-4: Delete suggestion

<b>Description</b>	This use case describes the process by which the doctor deletes a nutrition and activity suggestion for a specific patient from the system
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is already logged into the system
<b>Postconditions</b>	The nutrition and activity suggestion is deleted from the system
<b>Triggers</b>	The doctor wants to delete a nutrition and activity suggestion from the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the delete button	Continue from UC-NUTR-3
BF-2	Dashboard	Asks for confirmation	
BF-3	Doctor	Confirms deletion	AF-A: Cancel delete
BF-4	Dashboard	Sends delete request to backend	PE-1: Communication error
BF-5	Backend	Deletes nutrition and activity suggestion	PE-2: Insufficient rights PE-3: Database error
BF-6	Dashboard	Indicates success	
BF-7	Dashboard	Navigates to suggestions list	See UC-NUTR-1

AF-A: Cancel delete			
ID	Actor / Component	Action	Notes & References



AF-A-1	Data manager	Presses the cancel button	
AF-A-2		[End use case]	

## 4.8. Dietary and activity plans

### 4.8.1. UC-PLAN-1: View plans list

<b>Description</b>	This use case describes the process by which the doctor views the dietary and activity plans which have been created for a specific patient
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is logged in to the system
<b>Postconditions</b>	The dashboard displays a list of dietary and activity plans for a patient
<b>Triggers</b>	The doctor wants to view a list of generated dietary and activity plans for a patient

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the <i>Meals and Activity Plans</i> option	Continue from UC-PATIENTS-3
BF-2	Dashboard	Requests the plans for the specified patient	PE-1: Communication error
BF-3	Backend	Retrieves the plans for the specified patient	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Displays the plans for the specified patient	

### 4.8.2. UC-PLAN-2: Create plan

<b>Description</b>	This use case describes the process by which the doctor creates a dietary and activity plan
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is logged in to the system
<b>Postconditions</b>	The dietary and activity plan is created and stored on the system The patient is notified on the mobile phone (optionally)
<b>Triggers</b>	The doctor wants to create a dietary and activity plan for a specific patient

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the add button	Continue from UC-PLAN-1 AF-A: Create from suggestion
BF-2	Dashboard	Presents a form to fill in the plan details	
BF-3	Doctor	Fills in the plan details	
BF-4	Doctor	Presses the <i>Save</i> button	AF-B: Cancel adding
BF-5	Dashboard	Sends the new plan to the backend	PE-1: Communication error
BF-6	Backend	Persists the new plan	PE-2: Insufficient rights PE-3: Database error AF-C: Notify patient



BF-7	Dashboard	Indicates success	
BF-8	Dashboard	Displays the current list of plans	See UC-PLAN-1

AF-A: Create from suggestion			
ID	Actor / Component	Action	Notes & References
AF-A-1	Doctor	Presses the <i>Create from suggestion</i> button	AF-B: Cancel adding
AF-A-2	Dashboard	Presents a list of nutrition suggestions to create a plan from	See UC-NUTR-1
AF-A-3	Doctor	Selects a suggestion	
AF-A-4	Dashboard	Requests a plan for the selected suggestion from the backend	
AF-A-5	Backend	Works out a plan based on patient data and nutrition suggestion and returns it to the dashboard	
AF-A-6	Dashboard	Presents a form prefilled with the proposed plan for further editing by the doctor	
AF-A-7		[Continue at BF-4]	

AF-B: Cancel add			
ID	Actor / Component	Action	Notes & References
AF-B-1	Doctor	Presses the cancel button	
AF-B-2		[Continue at BF-8]	

AF-C: Notify patient			
ID	Actor / Component	Action	Notes & References
AF-B-1	Backend	In case the doctor selected the <i>Notify patient</i> option, the backend notifies the patient's mobile application about the new plan	Could be an option or a <i>Save &amp; Notify</i> button
AF-B-2		[Continue at BF-7]	

#### 4.8.3. UC-PLAN-3: View plan details

<b>Description</b>	This use case describes the process by which the doctor can view details about a patient's dietary and activity plan
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is already logged in
<b>Postconditions</b>	The dietary and activity plan details are presented on the dashboard
<b>Triggers</b>	The doctor wants to view a dietary and activity plan's details

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses on a plan item from the list	Continue from UC-PLAN-1
BF-2	Dashboard	Requests the plan details from the backend	PE-1: Communication error
BF-3	Backend	Retrieves the persisted plan details	PE-2: Insufficient rights



			PE-3: Database error
BF-4	Dashboard	Presents the dietary and activity plan details	

#### 4.8.4. UC-PLAN-4: Edit plan

<b>Description</b>	This use case describes the process by which the doctor edits a dietary and activity plan
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is already logged into the system
<b>Postconditions</b>	The dietary and activity plan is updated The patient is notified on the mobile phone (optionally)
<b>Triggers</b>	The doctor wants to edit a dietary and activity plan on the backend

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the edit button	Continue from UC-PLAN-3
BF-2	Dashboard	Enters edit mode	
BF-3	Doctor	Edits the details that need to be edited	
BF-4	Doctor	Presses the <i>Save</i> button	AF-A: Cancel editing
BF-5	Dashboard	Sends updated details	PE-1: Communication error
BF-6	Backend	Persists updated details	PE-2: Insufficient rights PE-3: Database error AF-B: Notify patient
BF-7	Dashboard	Indicates success	
BF-8	Dashboard	Exits edit mode	

AF-A: Cancel editing			
ID	Actor / Component	Action	Notes & References
AF-A-1	Medical professional	Presses the cancel button	
AF-A-2	Dashboard	Undoes any changes	
AF-A-3		[Continue with BF-8]	i.e. exit edit mode

AF-B: Notify patient			
ID	Actor / Component	Action	Notes & References
AF-B-1	Backend	In case the doctor selected the <i>Notify patient</i> option, the backend notifies the patient's mobile application about the edited plan	Could be an option or a <i>Save &amp; Notify</i> button
AF-B-2		[Continue at BF-7]	

#### 4.8.5. UC-PLAN-5: Delete plan

<b>Description</b>	This use case describes the process by which the doctor deletes a dietary and activity plan for a specific patient
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard



	Backend
<b>Preconditions</b>	The doctor is already logged into the system
<b>Postconditions</b>	The dietary and activity plan is deleted from the system The patient is notified on the mobile phone
<b>Triggers</b>	The doctor wants to delete a dietary and activity plan from the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the delete button	Continue from UC-PLAN-3
BF-2	Dashboard	Asks for confirmation	
BF-3	Doctor	Confirms deletion	AF-A: Cancel delete
BF-4	Dashboard	Sends delete request to backend	PE-1: Communication error
BF-5	Backend	Deletes plan	PE-2: Insufficient rights PE-3: Database error
BF-6	Backend	Notifies the patient on the mobile phone	
BF-7	Dashboard	Indicates success	
BF-8	Dashboard	Navigates to plans list	See UC-PLAN-1

AF-A: Cancel delete			
ID	Actor / Component	Action	Notes & References
AF-A-1	Doctor	Presses the cancel button	
AF-A-2		[End use case]	

#### 4.8.6. UC-PLAN-6: Send plan to patient

<b>Description</b>	This use case describes the process by which the doctor sends a new or recently edited dietary and activity plan to the patient's mobile app
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is already logged into the system
<b>Postconditions</b>	The new dietary and activity plan is sent to the patient's mobile phone
<b>Triggers</b>	The doctor wants to notify the patient about changes in the dietary and activity plan

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the <i>Send</i> button	Continue from UC-PLAN-3
BF-2	Dashboard	Requests the backend to notify the patient about the specified plan	PE-1: Communication error
BF-3	Backend	Notifies the patient on the mobile phone	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Indicates success	
BF-5	Dashboard	Navigates to plans list	See UC-PLAN-1



#### 4.8.7. UC-PLAN-7: Send notification to patient

<b>Description</b>	This use case describes the process by which the backend sends a notification set by the doctor in the dietary and activity plan to the patient's mobile app
<b>Actors</b>	Patient
<b>Components</b>	Backend Mobile app
<b>Preconditions</b>	The doctor has created a dietary and activity plan and set notifications at specified hours
<b>Postconditions</b>	The notification is received at the mobile app
<b>Triggers</b>	The time set to send the notification has come

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Backend	Sends a notification to the patient's mobile phone informing the patient it's time to eat a food or perform an exercise from the plan	
BF-2	Mobile app	Notifies the patient (visual, audio and vibration) it's time to eat a food or perform an exercise from the plan	

## 4.9. Food and activity diary

The food and activity diary is an electronic log maintained at the backend. It contains logs of the food consumed and activities performed (active and sedentary ones) by patients. It can be updated either through the mobile app or manually by the doctors based on hand written food and activity diaries filled in by patients or their guardians.

#### 4.9.1. UC-DIARY-1: View diary entries

<b>Description</b>	This use case describes the process by which the doctor views the activity performed, and meals consumed by the patient and reported by the NUTRISHIELD mobile app, either by selecting from a list (UC-APP-5), sending a picture (UC-APP-6) or ticking a box.
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is logged into the system
<b>Postconditions</b>	The doctor views the activity performed, and the meals consumed by the patient
<b>Triggers</b>	The doctor wants to view the reported activity performed and meals consumed by the patient

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Selects the <i>Food and activity diary</i> option	Continue from UC-PATIENT-3
BF-2	Dashboard	Requests the foods and activity logged for the specified patient	PE-1: Communication error
BF-3	Backend	Retrieves the food and activity logged for the specified patient	PE-2: Insufficient rights PE-3: Database error



BF-4	Dashboard	Presents a list of the logged entries, both text and pictures, along with the time reported and any other relevant data.	Relevant data might be Nutrients, portion/number of items consumer, etc
------	-----------	--	---

#### 4.9.2. UC-DIARY-2: Add diary entry

<b>Description</b>	This use case describes the process by which the doctor can add entries to the patient's food and activity diary		
<b>Actors</b>	Doctor		
<b>Components</b>	Dashboard Backend		
<b>Preconditions</b>	The doctor is logged into the system		
<b>Postconditions</b>	A new entry is added to the patient's diary		
<b>Triggers</b>	The doctor wants to add an entry to the food and activity diary		

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Selects the <i>Add</i> option	
BF-2	Dashboard	Displays a text box for the doctor to start typing in keywords that describe a food or activity	
BF-3	Doctor	Types some characters	
BF-4	Dashboard	Requests foods and activities from the backend that match the typed characters	PE-1: Communication error
BF-5	Backend	Retrieves top 10 matches	PE-2: Insufficient rights PE-3: Database error
BF-6	Dashboard	Presents the matched foods and activities	
BF-7	Doctor	Selects the appropriate entry	AF-A: Entry not found
BF-8	Dashboard	Prompts for additional information	e.g. portion size, number of items consumed, hours, etc
BF-9	Doctor	Fills in additional details and presses <i>Save</i>	AF-B: Cancel add
BF-8	Dashboard	Sends the new diary entry to the backend	PE-1: Communication error
BF-9	Backend	Persists the new diary entry	PE-2: Insufficient rights PE-3: Database error
BF-10	Dashboard	Indicates success	
BF-11	Dashboard	Updates the diary entries to include the new entry	

AF-A: Entry not found			
ID	Actor / Component	Action	Notes & References
AF-A-1	Dashboard	In case the entry is not found on the list, the dashboard offers the doctor to fill it in.	
AF-A-2	Doctor	Selects the <i>Add missing entry</i> option	
AF-A-3	Dashboard	Presents the doctor the UI to fill in a missing food or activity entry	Mainly following the steps in UC-FOOD-2
AF-A-4	Dashboard	After the entry is filled in, it is automatically selected, and the process proceeds at step BF-8	



AF-B: Cancel add			
ID	Actor / Component	Action	Notes & References
AF-A-1	Doctor	The doctor doesn't want to proceed, and presses Cancel	
AF-A-2	Dashboard	Closes the prompt	
AF-A-3		[Continue with BF-7]	

#### 4.9.3. UC-DIARY-3: View diary entry

<b>Description</b>	This use case describes the process by which the doctor views an entry in the food and activity diary
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is already logged into the system
<b>Postconditions</b>	The dashboard displays the food and activity diary entry and its details
<b>Triggers</b>	The doctor wants to view more details about a food and activity diary entry

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses on a diary item from the list	Continue from UC-DIARY-1
BF-2	Dashboard	Requests the diary item details from the backend	PE-1: Communication error
BF-3	Backend	Retrieves the persisted diary item details	PE-2: Insufficient rights PE-3: Database error
BF-4	Dashboard	Presents the diary item details	

#### 4.9.4. UC-DIARY-4: Edit diary entry

<b>Description</b>	This use case describes the process by which the doctor edits a food and activity diary entry
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is already logged into the system
<b>Postconditions</b>	The diary entry is updated
<b>Triggers</b>	The doctor wants to current an existing food and activity diary entry for a patient

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the edit button	Continue from UC-DIARY-3
BF-2	Dashboard	Enters edit mode	
BF-3	Doctor	Edits the details that need to be edited	
BF-4	Doctor	Presses the Save button	AF-A: Cancel editing
BF-5	Dashboard	Sends updated details	PE-1: Communication error
BF-6	Backend	Persists updated details	PE-2: Insufficient rights PE-3: Database error
BF-7	Dashboard	Indicates success	
BF-8	Dashboard	Exits edit mode	



AF-A: Cancel editing			
ID	Actor / Component	Action	Notes & References
AF-A-1	Medical professional	Presses the cancel button	
AF-A-2	Dashboard	Undoes any changes	
AF-A-3		[Continue with BF-8]	i.e. exit edit mode

#### 4.9.5. UC-DIARY-5: Delete diary entry

<b>Description</b>	This use case describes the process by which the doctor deletes an entry from a specific patient's food and activity diary
<b>Actors</b>	Doctor
<b>Components</b>	Dashboard Backend
<b>Preconditions</b>	The doctor is already logged into the system
<b>Postconditions</b>	The food and activity diary entry is deleted from the system
<b>Triggers</b>	The doctor wants to delete a food and activity diary entry from the system

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Doctor	Presses the delete button	Continue from UC-DIARY-3
BF-2	Dashboard	Asks for confirmation	
BF-3	Doctor	Confirms deletion	AF-A: Cancel delete
BF-4	Dashboard	Sends delete request to backend	PE-1: Communication error
BF-5	Backend	Deletes food and activity diary entry	PE-2: Insufficient rights PE-3: Database error
BF-7	Dashboard	Indicates success	
BF-8	Dashboard	Navigates to food and activity diary list	See UC-DIARY-1

AF-A: Cancel delete			
ID	Actor / Component	Action	Notes & References
AF-A-1	Doctor	Presses the cancel button	
AF-A-2		[End use case]	

## 4.10. Mobile phone app

### 4.10.1. UC-APP-1: Login to mobile phone app

<b>Description</b>	This use case describes the process by which the patient logs into the NUTRISHIELD mobile phone app
<b>Actors</b>	Patient
<b>Components</b>	App Backend
<b>Preconditions</b>	The patient has been registered to the system by a doctor
<b>Postconditions</b>	The patient logs into the app



<b>Triggers</b>	The patient wants to use the application
-----------------	--

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Patient	Enters credentials	Assume a login screen
BF-2	Patient	Presses login	
BF-3	App	Sends credentials to backend	PE-1: Communication error
BF-4	Backend	Authenticates user	AF-A: Invalid credentials PE-3: Database error
BF-5	App	Navigates to main page	AF-B: First time login

AF-A: Invalid credentials			
ID	Actor / Component	Action	Notes & References
AF-A-1	Backend	If the user has provided invalid credentials, the backend returns an error code	Could be a standard HTTP code
AF-A-2	App	Presents an indicative message and prompts the user to try again	
AF-A-3		[Continue with BF-1]	

AF-B: First time login			
ID	Actor / Component	Action	Notes & References
AF-B-1	App	If this is the first time the patient logs in, the app asks the user to provide a new password	
AF-B-2	Patient	Types in the new and confirmation of new password	
AF-B-3	Patient	Presses the <i>Change Password</i> button	
AF-B-5	App	Sends the change password request to the backend	PE-1: Communication error AF-C: Password error
AF-B-6	Backend	Persists the password change	PE-3: Database error
AF-B-7	App	Indicates success	
AF-B-8		[Continue with BF-5]	

AF-C: Password error			
ID	Actor / Component	Action	Notes & References
AF-C-1	App	If the passwords provided do not match or they do not meet certain security standards the app presents the user with an indicative message and prompts to try again	
AF-B-2	Patient	[Continue with AF-B-2]	

#### 4.10.2. UC-APP-2: Change password

<b>Description</b>	This use case describes the process by which the patient changes their password from the NUTRISHIELD mobile app
<b>Actors</b>	Patient



<b>Components</b>	App Backend
<b>Preconditions</b>	The patient is logged in
<b>Postconditions</b>	The patient sets a new password
<b>Triggers</b>	The patient wants to change the password used to log into the NUTRISHIELD mobile app

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Patient	Selects the <i>Change password</i> option	
BF-2	App	Prompts for new and confirmation of new password	
BF-3	Patient	Types in new and confirmation of new password	
BF-4	Patient	Presses the <i>Change password</i> button	
BF-5	App	Sends the change password request to the backend	AF-A: Password error PE-1: Communication error
AF-B-6	Backend	Persists the password change	PE-3: Database error
AF-B-7	App	Indicates success	

AF-A: Password error			
ID	Actor / Component	Action	Notes & References
AF-C-1	App	If the passwords provided do not match or they do not meet certain security standards the app presents the user with an indicative message and prompts to try again	
AF-B-2	Patient	[Continue with AF-B-2]	

#### 4.10.3. UC-APP-3: Receive plan

<b>Description</b>	This use case describes the process by which the patient receives a new or updated dietary and activity plan notification on the mobile phone
<b>Actors</b>	Patient
<b>Components</b>	App Backend
<b>Preconditions</b>	The patient is logged in
<b>Postconditions</b>	The patient receives a notification about a new or updated dietary and activity plan
<b>Triggers</b>	The doctor sends a dietary and activity plan to the patient (see UC-PLAN-2, 4, 6)

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	App	Receives a notification from the backend	
BF-2	App	Requests the new plan from the backend	PE-1: Communication error
BF-3	Backend	Retrieves the new plan	PE-2: Insufficient rights PE-3: Database error
BF-4	App	Updates the plan on the mobile phone	
BF-5	App	Notifies the user about the new plan	



AF-6	Patient	Views the plan	See UC-APP-5
------	---------	----------------	--------------

#### 4.10.4. UC-APP-4: View plan

<b>Description</b>	This use case describes the process by which the patient views the dietary and activity plan and statistics on the goals set
<b>Actors</b>	Patient
<b>Components</b>	App Backend
<b>Preconditions</b>	The patient is logged in
<b>Postconditions</b>	The patient views the dietary and activity plan and statistics on the goals reached
<b>Triggers</b>	The patient wants to view the dietary and activity plan assigned by the doctor

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Patient	Presses the <i>Plan</i> option	
BF-2	App	Displays the plan along with statistics on the nutritional goals reached (or exceeded)	

#### 4.10.5. UC-APP-5: Report meal or activity from list

<b>Description</b>	This use case describes the process by which the patient uses the NUTRISHIELD mobile app to log consumed food or performed activity by selecting it from a list
<b>Actors</b>	Patient
<b>Components</b>	App Backend
<b>Preconditions</b>	The patient is logged in
<b>Postconditions</b>	The backend is updated with the patient's consumed food or performed activity
<b>Triggers</b>	The patient needs to log consumed meal or activity performed using the mobile app

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Patient	Presses the <i>Log food or activity from list</i> option	
BF-2	App	Displays a text box for the patient to start typing in keywords	
BF-3	Patient	Types some characters	
BF-4	App	Requests foods and activities from the backend that match the typed characters	PE-1: Communication error
BF-5	Backend	Retrieves top 10 matches	PE-2: Insufficient rights PE-3: Database error
BF-6	App	Presents the matched entries	
BF-7	Patient	Taps on the appropriate entry to select it	
BF-8	App	Prompts for additional information	e.g. portion size, number of items consumed, hours, etc
BF-9	Patient	Fills in additional details and presses <i>Save</i>	AF-A: Cancel add
BF-8	App	Sends the logged item to the backend	PE-1: Communication error
BF-9	Backend	Persists the logged item for the database	PE-2: Insufficient rights



			PE-3: Database error
BF-10	App	Indicates success and offers undo option	AF-B: Undo add
BF-11	App	Updates the statistics on the target goals based on the food selected	
BF-12	App	Clears the text box to allow logging a new food	

AF-A: Cancel add			
ID	Actor / Component	Action	Notes & References
AF-A-1	Patient	The patient doesn't want to proceed with adding details about the selected item and presses Cancel	
AF-A-2	App	Closes the prompt	
AF-A-3		[Continue with BF-7]	

AF-B: Undo add			
ID	Actor / Component	Action	Notes & References
AF-A-1	Patient	When the entry is logged, the App presents an option to Undo the action and the patient selects this <i>Undo</i> option	
AF-A-2	App	Sends a request to the backend to remove the logged item	PE-1: Communication error
AF-A-3	Backend	Deletes the recently logged item	PE-2: Insufficient rights PE-3: Database error
AF-A-4	App	Updates statistics on the target goals based on the item that was removed	
AF-A-5		[Continue with BF-7]	

#### 4.10.6. UC-APP-6: Report consumed meal from photograph

<b>Description</b>	This use case describes the process by which the patient uses the NUTRISHIELD mobile app to log consumed food by capturing a picture of it
<b>Actors</b>	Patient
<b>Components</b>	App Backend
<b>Preconditions</b>	The patient is logged in
<b>Postconditions</b>	The backend is updated with the patient's picture
<b>Triggers</b>	The patient needs to log consumed meal using the NUTRISHIELD mobile app

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Patient	Presses the <i>Log food from photo</i> option	
BF-2	App	Enters the picture taking mode	
BF-3	Patient	Takes a picture	
BF-4	App	Uploads the picture to the backend	PE-1: Communication error
BF-5	Backend	Persists the picture and the date taken for the patient who uploaded it	PE-2: Insufficient rights PE-3: Database error



BF-6	App	Indicates success and offers undo option	AF-A: Undo add
------	-----	--	----------------

AF-A: Undo add			
ID	Actor / Component	Action	Notes & References
AF-A-1	Patient	When the picture is uploaded, the App presents an option to Undo the action and the patient selects this <i>Undo</i> option	
AF-A-2	App	Sends a request to the backend to remove the uploaded picture	PE-1: Communication error
AF-A-3	Backend	Deletes the recently uploaded picture	PE-2: Insufficient rights PE-3: Database error
AF-A-4		[End use case]	

#### 4.10.7. UC-APP-8: Get nutritional information from barcode

<b>Description</b>	This use case describes the process by which the patient uses the NUTRISHIELD mobile app to scan a product and view the relevant information
<b>Actors</b>	Patient
<b>Components</b>	App Backend
<b>Preconditions</b>	The patient is logged in
<b>Postconditions</b>	The mobile app displays the product's relevant information
<b>Triggers</b>	The patient wants to know about a product's relevant information

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Patient	Presses the <i>Scan Barcode</i> option	
BF-2	App	Enters the barcode scanning mode	
BF-3	Patient	Scans the barcode	AF-A: Scan error
BF-4	App	Processes the barcode and request relevant information from the backend	PE-1: Communication error
BF-5	Backend	Retrieves relevant information about the scanned barcode	PE-2: Insufficient rights PE-3: Database error AF-B: Product not found
BF-6	App	Presents the relevant information retrieved	

AF-A: Scan error			
ID	Actor / Component	Action	Notes & References
AF-A-1	App	If the app cannot read the bard code for any reason an indicative message is presented to the user prompting to try again	
AF-A-2		[Continue with BF-3]	

AF-B Product not found			
ID	Actor / Component	Action	Notes & References



AF-B-1	Backend	If the product is not found on the backend, the backend returns an indicative code	Could be a standard HTTP code
AF-A-2	App	Presents an indicative message to the user informing that relevant information about the scanned product was not found on the backend.	
AF-A-3		[Continue at BF-3]	

#### 4.10.8. UC-APP-9: Logout from mobile phone app

<b>Description</b>	This use case describes the process by which the patient logs out of the NUTRISHIELD mobile application.
<b>Actors</b>	Patient
<b>Components</b>	App Backend
<b>Preconditions</b>	The patient is logged in
<b>Postconditions</b>	The patient is logged out
<b>Triggers</b>	The patient wants to log out of the mobile application

Basic flow			
ID	Actor / Component	Action	Notes & References
BF-1	Patient	Presses the <i>Log out</i> option	
BF-2	App	Logs the patient out of the application	

## 4.11. Common exception flows

This section presents common exception flows shared among several use cases. These flows would normally be placed below their corresponding use case. However, since their behaviour is the same in all use cases, they are presented here to avoid unnecessary repetition.

#### 4.11.1. PE-1: Communication failure when accessing the backend

<b>Description</b>	This exception flow describes the process by which the dashboard and the mobile application components handle a communication error when accessing the backend
<b>Actors</b>	Any user
<b>Components</b>	Dashboard Mobile application
<b>Preconditions</b>	-
<b>Postconditions</b>	User is notified about the communication error
<b>Triggers</b>	The dashboard or the mobile application tries to communicate with the NUTRISHIELD Backend

Exception Flow: Communication error			
ID	Actor / Component	Action	Notes & References
PE-1	Dashboard or Mobile application	If the application fails to communicate with the backend, then the user is informed and asked to try again later or contact the system administrator if the problem persists	



PE-2		[End use case]	
------	--	----------------	--

#### 4.11.2. PE-2: Operation fails due to insufficient access rights

<b>Description</b>	This exception flow describes the system behaviour when a user with insufficient rights tries to perform an action on the backend using the dashboard or the mobile application
<b>Actors</b>	Any user
<b>Components</b>	Backend Dashboard Mobile application
<b>Preconditions</b>	The user is logged in
<b>Postconditions</b>	User is notified about insufficient rights to perform the operation
<b>Triggers</b>	User tries to perform an action without having the necessary rights

Exception Flow: Insufficient rights			
ID	Actor / Component	Action	Notes & References
PE-1	Backend	If the user has no rights to perform the operation, the backend returns an error code	Could be a standard HTTP error code
PE-2	Dashboard or Mobile application	The user is notified with an indicative message and asked to contact the system administrator if they require access to this functionality	
PE-3		[End use case]	

#### 4.11.3. PE-3: Operation fails due to database errors

<b>Description</b>	This exception flow describes the system behaviour when a database operation fails at the Backend for any reason
<b>Actors</b>	-
<b>Components</b>	Backend
<b>Preconditions</b>	-
<b>Postconditions</b>	An error message is returned from the backend indicating failure
<b>Triggers</b>	The backend tries to read from or write to the backend database

Exception Flow: Database error			
ID	Actor / Component	Action	Notes & References
PE-1	Backend	If a database operation fails for any reason while the backend tries to access it, the backend returns an error code	Could be a standard HTTP error code
PE-2	Dashboard or Mobile application	A notification with an indicative message pops up and asks the user to contact the system administrator if the problem persists	
PE-3		[End use case]	



## 5. Requirements

This section presents the system functional and non-functional requirements. The requirements are presented per system component.

### 5.1. Dashboard

This section presents the functional and non-functional requirements of the dashboard.

#### 5.1.1. Functional requirements

This section presents the functional requirements of the dashboard component used by administrators and medical personnel. The requirements are presented in groups of common functionalities.

##### 5.1.1.1. Administration

Id	Description	Priority	Case
DA-1	The dashboard must allow administrators to view a list of registered medical professionals	Must	UC-ADMIN-1
DA-2	The dashboard must allow administrators to register new medical professionals on the platform	Must	UC-ADMIN-2
DA-3	The dashboard should allow administrators to view registered details for each medical professional	Should	UC-ADMIN-3
DA-4	The dashboard should allow administrators to edit the details of registered medical professionals	Should	UC-ADMIN-4
DA-5	The dashboard should allow administrators to delete registered medical professionals	Should	UC-ADMIN-5

*Table 4 – Dashboard administration functional requirements*

##### 5.1.1.2. Account

Id	Description	Priority	Case
DAC-1	The dashboard must provide a login form to ask for a username and password	Must	UC-ACCOUNT-1
DAC-2	The dashboard must provide users the option to change their password	Must	UC-ACCOUNT-2
DAC-3	The dashboard must provide users the option to log out	Must	UC-ACCOUNT-3
DAC-4	The dashboard must provide administrators the option to impersonate medical personnel users to perform actions on their behalf (such as resetting a forgotten password)	Must	UC-ACCOUNT-4
DAC-5	The dashboard should provide the option to medical personnel to define a nickname for easy identification in the context of anonymization	Should	UC-ACCOUNT-5
DAC-6	The dashboard could provide suggestions for alternative nicknames in case the nickname is used by another user	Could	UC-ACCOUNT-5

*Table 5 – Dashboard account functional requirements*

#### 5.1.1.3. Patients

Id	Description	Priority	Case
DP-1	The dashboard must display a list of patients assigned to the currently logged in medical professional.	Must	UC-PATIENTS-1
DP-2	The dashboard must allow doctors to create patient users by filling in the minimum data required in an anonymised way.	Must	UC-PATIENTS-2
DP-3	The dashboard must allow medical personnel to view the details of a patient. The details should present everything related to the patient with basic information being readily available and further information, such as measurements, nutrition suggestions, dietary plans and food diaries accessible through tabs	Must	UC-PATIENTS-3
DP-4	The dashboard should allow doctors to edit a patient's basic data. Doctors can only edit the data for patients they created	Should	UC-PATIENTS-4
DP-5	The dashboard must allow doctors to assign patients to other medical professionals.	Must	UC-PATIENTS-4
DP-6	The dashboard must allow doctors to delete patients they created	Must	UC-PATIENTS-5

*Table 6 – Dashboard Patients functional requirements*

#### 5.1.1.4. Measurements

Id	Description	Priority	Case
DM-1	The dashboard must display a list of a specific patient's measurements. Each entry must display at least the date the measurement was made and the type of measurement.	Must	UC-MSR-1
DM-2	The dashboard must provide medical professionals the options to input various types of measurements for a patient. For example: <ul style="list-style-type: none"> <li>Basic measurements such as weight, height, age.</li> <li>Assessments questionnaires such as FFQs and KIDMED [2].</li> <li>Laboratory results such as gut microbiome, etc.</li> </ul> The dashboard must provide a different form for each type of data to input. Some data must be provided by filling in fields on the screen, while other data must be provided by uploading files	Must	UC-MSR-2
DM-3	The dashboard must allow medical professionals to view measurements or download files of measurements for a specific patient assigned to them.	Must	UC-MSR-3
DM-4	The dashboard should allow medical professionals to edit measurements for patients they are assigned to, either by editing fields or uploading new measurement files.	Should	UC-MSR-4
DM-5	The dashboard could optionally display to doctors previous versions of edited or deleted measurements in the list of measurements for a patient they are assigned to.	Could	UC-MSR-1
DM-6	The dashboard must allow medical professionals to delete measurements for a patient they are assigned to.	Must	UC-MSR-5
DM-7	The dashboard could allow sample container QR codes labels to be associated with patients	Could	UC-MSR-6
DM-8	The dashboard could allow medical professionals to scan sample container QR code labels to start filling in measurements for the associated patient	Could	UC-MSR-7

*Table 7 – Dashboard measurements functional requirements*



#### 5.1.1.5. Foods and activities

Id	Description	Priority	Case
DF-1	The dashboard should display a list of foods and activities already in the system to data managers	Should	UC-FOOD-1
DF-2	The dashboard should allow data managers to add new entries to the existing food and activities. Food entries contain description, nutrients, portion sizes, units of measurement, bar code (for branded food), picture. Activities contain description, energy expenditure information, picture	Should	UC-FOOD-2
DF-3	The dashboard should display the details of a specific food or activity	Should	UC-FOOD-3
DF-4	The dashboard should allow data managers to edit the details of a specific food or activity	Should	UC-FOOD-4
DF-5	The dashboard should allow data managers to delete a specific food or activity from the backend	Should	UC-FOOD-5

*Table 8 – Dashboard Foods functional requirements*

#### 5.1.1.6. Nutrition algorithm

Id	Description	Priority	Case
DN-1	The dashboard must display to doctors a list of nutrition and activity suggestions generated for a patient that is assigned to them. The list should at least contain the date/time the suggestion was generated	Must	UC-NUTR-1
DN-2	The dashboard should allow doctors to request new suggestions from the backend for a specific patient that is assigned to them	Should	UC-NUTR-2
DN-3	The dashboard should display to doctors the details of a suggestion generated for a patient assigned to the doctor.	Should	UC-NUTR-3
DN-4	The dashboard could provide alternative machine learning based suggestions	Could	UC-NUTR-3
DN-5	The dashboard must allow doctors to delete suggestions for a patient assigned to them.	Must	UC-NUTR-4

*Table 9 – Dashboard nutrition algorithm functional requirements*

#### 5.1.1.7. Dietary and activity plan

Id	Description	Priority	Case
DMP-1	The dashboard must display a list of patient dietary and activity plans to the doctor for a patient that is assigned to this doctor. There must be an indication for plans that have not yet been sent to patients (see DMP-6)	Must	UC-PLAN-1
DMP-2	The dashboard must provide the forms and functionality to doctors to compile a dietary and activity plan manually	Must	UC-PLAN-2
DMP-3	The dashboard could provide the means to initialise a dietary and activity plan automatically by considering nutrition and activity suggestions for the current patient	Could	UC-PLAN-2
DMP-4	The dashboard must allow doctors to view the details of a dietary and activity plan.	Must	UC-PLAN-3
DMP-5	The dashboard must allow doctors to edit the details of an existing dietary and activity plan for a patient assigned to them.	Must	UC-PLAN-4



DMP-6	The dashboard should allow doctors to indicate whether they want the dietary and activity plan to be sent to the patient's mobile after successfully creating it or editing it.	Should	UC-PLAN-2 UC-PLAN-4
DMP-7	The dashboard could allow doctors to select from the foods and activities registered in the system when creating or editing the plan. Entries not available could be inserted by doctors using existing user interfaces for adding new data items to the system	Could	UC-PLAN-2 UC-PLAN-4 UC-FOOD-2
DMP-8	The dashboard must allow doctors to delete a dietary and activity plan for a patient assigned to them.	Must	UC-PLAN-5
DMP-9	The dashboard must allow doctors to send dietary and activity plans to patients on demand	Must	UC-PLAN-6
DMP-10	The dashboard should allow setting notifications to be sent to the patients regarding consumption of food or planned activity in the plan at specified hours.	Should	UC-PLAN-7

*Table 10 – Dashboard dietary plan functional requirements*

#### 5.1.1.8. Food and activity diary

Id	Description	Priority	Case
DD-1	The dashboard should be able to display the patient's food and activity diary to the doctor assigned to the patient.	Should	UC-DIARY-1
DD-2	The dashboard could allow doctors to add entries to the patient's diary	Could	UC-DIARY-2
DD-3	The dashboard should allow doctors to view details about an entry in the food and activity diary of a patient assigned to them	Should	UC-DIARY-3
DD-4	The dashboard could allow doctors to edit food and activity diary entries for the patients they are assigned to	Could	UC-DIARY-4
DD-5	The dashboard could allow doctors to delete food and activity diary entries for the patients they are assigned to	Could	UC-DIARY-5

*Table 11 – Dashboard food diary functional requirements*

#### 5.1.1.9. General

Id	Description	Priority	Case
DG-1	Lists in the dashboard should be sortable	Should	UC-ADMIN-1 UC-PATIENTS-1 UC-MSR-1
DG-2	Lists in the dashboard should be searchable	Should	
DG-3	List in the dashboard should be displayed in pages	Should	
DG-4	The number of items displayed per page could be set from the user interface	Could	UC-FOOD-1 UC-PLAN-1 UC-DIARY-1
DG-5	All operations that successfully create, edit or delete entities from the system, should indicate success via a visual feedback such as a toast	Should	UC-ADMIN-2,4,5 UC-ACCOUNT-2,4 UC-PATIENTS-2,4,5 UC-MSR-2,3,5 UC-FOOD-2,4,5 UC-NUTR-2,4 UC-PLAN-2,4,5,6 UC-DIARY-2,4,5



DG-6	All operations failures should show a message with either recovery steps or general information on the effect the error had (e.g. the dietary plan was not saved)	Should	All
DG-7	Pages and options in the user interface must be available depending on the role of the logged in user. Non-applicable pages and options must remain hidden to unauthorised users.	Must	All
DG-8	Operations that take more than 500 milliseconds to complete could indicate a progress indicator while waiting	Could	All
DG-9	Action buttons and form fields must be disabled while an action is performed to prevent users from issuing the same action again	Must	All
DG-10	Deleting multiple items from a list could be achieved by allowing multiple selection and delete functionality while in list view mode	Could	UC-ADMIN-1 UC-PATIENTS-1 UC-MSR-1 UC-FOOD-1 UC-NUTR-1 UC-PLAN-1 UC-DIARY-1
DG-11	All operations that create or edit entities should perform a data validation prior to sending the data to the backend to reduce unnecessary network overhead and server resources in case missing or invalid data is to be submitted	Should	UC-ADMIN-2,4 UC-ACCOUNT-2 UC-PATIENTS-2,4 UC-MSR-2,3 UC-FOOD-2,4 UC-NUTR-2,4 UC-PLAN-2,4 UC-DIARY-2,4
DG-12	The dashboard should support entering food and activity data in multiple languages to be available for selection by mobile app users speaking different languages	Should	UC-FOOD-2,4

*Table 12 – Dashboard general functional requirements*

### 5.1.2. Non-functional requirements

This section presents the non-functional requirements for the dashboard

Id	Description	Priority
DNF-1	Navigating the dashboard should be intuitive	Must

*Table 13 –Dashboard non-functional requirements*

## 5.2. Backend

This section presents the functional and non-functional requirements of the backend.

### 5.2.1. Functional requirements

This section presents the functional requirements of the backend which performs various backend operations and serves the dashboard and the mobile application.



#### 5.2.1.1. Administration

Id	Description	Priority	Case
BA-1	The backend must serve the dashboard a list of registered medical professionals	Must	UC-ADMIN-1 UC-PATIENTS-2 UC-PATIENTS-4
BA-2	The backend must provide system administrators the functionality to create new medical professionals (doctors, lab technicians)	Must	UC-ADMIN-2
BA-3	The backend should serve the dashboard the details of a specified medical professional	Should	UC-ADMIN-3
BA-4	The backend should provide the functionality to update the details of a medical professional	Should	UC-ADMIN-4
BA-5	The backend must provide the functionality to administrators for deleting a medical professional	Must	UC-ADMIN-5

*Table 14 – Backend administration functional requirements*

#### 5.2.1.2. Account

Id	Description	Priority	Case
BAC-1	The backend must handle login requests from the dashboard and the mobile application. Upon submitting the correct credentials, access to logins issued from the dashboard should only be granted to medical professionals while access to mobile applications is granted to patients.	Must	UC-ACCOUNT-1 UC-APP-1
BAC-2	The backend must process password change requests from the dashboard and the mobile application for the currently logged in user	Must	UC-ACCOUNT-2 UC-APP-2
BAC-3	The backend must handle logout requests from the dashboard and the mobile application	Must	UC-ACCOUNT-3 UC-APP-9
BAC-4	The backend must allow administrators to impersonate any medical professional user and execute any actions on their behalf	Must	UC-ACCOUNT-4
BAC-5	The backend should allow medical professionals to set a nickname for easier identification	Should	UC-ACCOUNT-5
BAC-6	The backend could provide suggestions of alternative nicknames based on the provided one in case the provided one is already used by another user	Could	UC-ACCOUNT-5

*Table 15 – Backend account functional requirements*

#### 5.2.1.3. Patients

Id	Description	Priority	Case
BP-1	The backend must serve a list of patients related to the user requesting it. This is a list of patients assigned to the medical professional requesting it.	Must	UC-PATIENTS-1
BP-2	The backend must allow doctors to save patient users created from the dashboard. The newly created patients are automatically assigned to the doctors that created them.	Must	UC-PATIENTS-2
BP-3	The backend must provide the basic patient details to the medical professionals requesting it. The patient must be assigned to the medical professional requesting the details	Must	UC-PATIENTS-3



BP-4	The backend must allow doctors to validate and save changes in patient's basic details made in the dashboard by doctors	Must	UC-PATIENTS-4
BP-4	The backend must allow doctors to assign/dissociate patients from/to other medical professionals. Doctors can only do that for patients that were either created by or assigned to them.	Must	UC-PATIENTS-4
BP-5	The backend must allow doctors to delete patients they created along with all their related data such as measurements, nutrition suggestions, dietary and activity plans and food diaries.	Must	UC-PATIENTS-5

*Table 16 – Backend patient functional requirements*

#### 5.2.1.4. Measurements

Id	Description	Priority	Case
BM-1	The backend must serve a list of measurements for a specified patient to display on the dashboard	Must	UC-MSR-1
BM-2	The backend must validate and save various types of new measurements send as fields and/or files from the dashboard by doctors and lab technicians	Must	UC-MSR-2
BM-3	The backend must serve the measurement details requested by medical professionals via the dashboard for a specific patient. Medical professionals can only request measurements for the patients they are assigned to.	Must	UC-MSR-3
BM-4	The backend should allow medical professionals to update existing measurements for patients they are assigned to either by uploading new values or files of measurements.	Should	UC-MSR-4
BM-5	The backend could server a list of measurements to a medical professional which also includes past versions of edited or deleted measurements for a patient the medical professional is assigned to.	Could	UC-MSR-1
BM-6	The backend must allow medical professionals to delete measurements for a patient the medical professionals are assigned to.	Must	UC-MSR-5

*Table 17 – Backend measurements functional requirements*

#### 5.2.1.5. Foods and activities

Id	Description	Priority	Case
BF-1	The backend should serve a list of foods and activities by specifying an optional search term to data managers and doctors for use in the dashboard and patients for use in the mobile application. The food data format could be compatible with the United States Department of Agriculture Food Composition Databases <sup>2</sup> which relate foods to nutrients per measuring units.	Should	UC-FOOD-1 UC-PLAN-2 UC-PLAN-4 UC-DIARY-2 UC-DIARY-4 UC-APP-5
BF-2	The backend could save new food and activity entries created by the data managers and doctors from the dashboard	Could	UC-FOOD-2

<sup>2</sup> <https://ndb.nal.usda.gov/ndb/>



BF-3	The backend should retrieve the details of a specific food or activity requested by the dashboard or the mobile app	Should	UC-FOOD-3 UC-APP-8
BF-4	The backend could update the details of a specific food or activity edited by a data manager and doctors from the dashboard	Could	UC-FOOD-4
BF-5	The backend could handle requests from doctors and data managers issued by the dashboard to delete specific food or activity items	Could	UC-FOOD-5

*Table 18 – Backend Foods functional requirements*

#### 5.2.1.6. Nutrition algorithm

Id	Description	Priority	Case
BN-1	The backend must retrieve a list of nutrition and activity suggestions already generated for a specific patient to display on the dashboard to the doctor	Must	UC-NUTR-1
BN-2	The backend must generate nutrition and activity suggestions upon request by the doctor for a patient that is assigned to the doctor. The suggestions must be generated by feeding the user measurements to the nutrition algorithm. The nutrition algorithm should partially be based on identified correlations between biomarkers and nutrition.	Must	UC-NUTR-2
BN-3	The backend must retrieve the nutrition and activity suggestion generated for a specific patient upon request by a doctor assigned to this patient through the dashboard	Must	UC-NUTR-3
BN-4	The backend must handle a doctor's request to delete a nutrition and activity suggestion for a patient that is assigned to this doctor	Must	UC-NUTR-4

*Table 19 – Backend nutrition algorithm functional requirements*

#### 5.2.1.7. Dietary and activity plan

Id	Description	Priority	Case
BMP-1	The backend must provide a list of dietary and activity plans for a specified patient to a doctor assigned to this patient.	Must	UC-PLAN -1
BMP-2	The backend must save a new dietary and activity plan created by a doctor for a patient that is assigned to this doctor.	Must	UC-PLAN-2
BMP-3	The backend could calculate a suggestion of a dietary and activity plan considering the latest nutrition and activity suggestions and offer it to the dashboard as a starting point for doctors to work on.	Could	UC-PLAN-2
BMP-4	There could be a consideration of patient's preferences when generating dietary and activity plans, or offer an alternative for suggested dietary and activity plan	Could	UC-PLAN-2
BMP-5	The backend must serve the dashboard the dietary and activity plan details for a specific patient as requested by the doctor assigned to the patient	Must	UC-PLAN-3
BMP-6	The backend must allow doctors to update the details of a dietary and activity plan they edited through the dashboard for a patient assigned to them	Must	UC-PLAN-4
BMP-7	The backend must handle requests for dietary and activity plans deletion issued by doctors through the dashboard for a patient that is assigned to them	Must	UC-PLAN-5



BMP-8	The backend should be able to send notifications to a patient's mobile app when a doctor requests this on demand, optionally when creating or editing a dietary and activity plan, and always when deleting a dietary and activity plan	Should	UC-PLAN-2 UC-PLAN-4 UC-PLAN-6 UC-APP-3
-------	---	--------	---

*Table 20 – Backend dietary and activity plan functional requirements*

#### 5.2.1.8. Food and activity diary

Id	Description	Priority	Case
BD-1	The backend should retrieve the food and activity diary of a patient as requested from the doctor assigned to this patient from the dashboard	Should	UC-DIARY-1
BD-2	The backend could allow a doctor to send new entries to save in the diary of a patient assigned with the doctor	Could	UC-DIARY-2
BD-3	The backend should retrieve a food and activity diary entry for a patient as requested by the dashboard from a doctor assigned to this patient	Should	UC-DIARY-3
BD-4	The backend could save updated details for a food and activity diary entry sent from a doctor through the dashboard for a patient assigned to the doctor	Could	UC-DIARY-4
BD-5	The backend could handle delete requests for food and activity diary entries issued by doctors for patients they are assigned to	Could	UC-DIARY-5

*Table 21 – Backend food diary functional requirements*

#### 5.2.1.9. General

Id	Description	Priority	Case
BG-1	For auditing purposes, all actions could be logged to the system along with information about the user and the date/time the action was performed. Any actions that create/modify entities data could also store the new state of the created/modified entity. Any changes made while a user was impersonated should also indicate who the impersonator user is.	Must	All
BG-2	The role of the currently logged in user must be checked prior to execution of each request and deny serving users who do not have the necessary rights.	Must	All
BG-3	The validity of the data sent must be checked prior to handling each request and, in case they data are not valid, the backend must communicate this to the client (dashboard or mobile app)	Must	All
BG-4	Data exchanged with the client applications (dashboard and mobile app) must be communicated over a secure channel	Must	All
BG-5	Changes of entities shared with the mobile application should be communicated to the mobile applications to be in sync with the backend data	Must	UC-PLAN-2,4,5,6 UC-DIARY-2,4,5
BG-6	The backend should be able to store and serve data in multiple languages	Should	-

*Table 22 – General Backend functional requirements*

### 5.2.2. Non-functional requirements

This section presents the non-functional requirements of the backend



Id	Description	Priority
BNF-1	Only the minimum required data must be requested and processed	Must
BNF-2	Personal data stored in the system should be anonymised	Must
BNF-3	The backend must be secure	Must

## 5.3. Mobile app

This section presents the functional and non-functional requirements of the NUTRISHIELD mobile app

### 5.3.1. Functional requirements

This section presents the functional requirements of the NUTRISHIELD mobile app.

#### 5.3.1.1. Account

Id	Description	Priority	Case
MA-1	The mobile application must provide the means for patients to log in using a username and password	Must	UC-APP-1
MA-2	The mobile application must allow logged in users to change their password	Must	UC-APP-2
MA-3	The mobile application must provide users the option to log out	Must	UC-APP-9

*Table 23 – Mobile application account functional requirements*

#### 5.3.1.2. Dietary and activity plan

Id	Description	Priority	Case
MP-1	Upon receiving a notification from the backend about a dietary and activity plan change, the mobile app must notify the user and update the user interface to reflect the changes	Must	UC-PLAN-2 UC-PLAN-4 UC-PLAN-5 UC-PLAN-6 UC-APP-3
MP-2	The mobile application should display the dietary and activity plan sent from the backend.	Should	UC-APP-4
MP-3	The mobile application could display notifications sent from the backend to remind patients to eat foods and perform activities found in the dietary and activity plan	Could	UC-PLAN-7

*Table 24 – Mobile dietary and activity plan functional requirements*

#### 5.3.1.3. Food diary

Id	Description	Priority	Case
MD-1	The mobile application could allow patients to record consumed food and performed activity by selecting items from a predefined list and setting related attributes (e.g. consumed quantity, hours of activity, etc)	Could	UC-APP-5
MD-2	The mobile application must allow patients to record consumed food by taking pictures of the food (before consumption of course)	Must	UC-APP-6

*Table 25 – Mobile application food diary functional requirements*



#### 5.3.1.4. Nutrition information

Id	Description	Priority	Case
MN-1	The mobile application could provide information on why a food was suggested	Could	UC-APP-3
MN-2	The mobile application could provide information on implications choices have	Could	UC-APP-5 UC-APP-8
MN-3	The mobile application should display the results of a healthy diet	Should	UC-APP-5
MN-4	The mobile application could communicate scientific measurements and health assessment results to patients	Should	UC-APP-4

*Table 26 – Mobile application nutrition information functional requirements*

#### 5.3.1.5. General

Id	Description	Priority	Case
MG-1	When an entity changes on the server, and the server notifies the mobile application about the change (BG-5), the mobile application must reflect the changes in its user interface (e.g. by updating statistics)	Must	UC-PLAN-2,4,5,6 UC-DIARY-2,4,5
MG-2	While waiting for the backend to respond, a visual progress indicator should be displayed, and any form controls should be disabled	Should	UC-APP-(all)
MG-3	The mobile app should support multiple languages	Could	-
MG-4	The mobile app must be able to provide nutrition information by scanning the bar codes of branded produces	Must	UC-APP-8

*Table 27 – Mobile application general functional requirements*

### 5.3.2. Non-functional requirements

This section presents the non-functional requirements of the NUTRISHIELD mobile app.

Id	Description	Priority
MNF-1	It must engage the end user in a pleasant and non-intrusive/judgemental way	Must
MNF-2	It must explain complex notions in an easy to understand manner	Must
MNF-3	It must have a user friendly and intuitive user interface	Must

*Table 28 – Mobile application non-functional requirements*

## 6. Architectural foundations

This section presents a preliminary architecture of the NUTRISHIELD platform. Figure 2, presents a preliminary system architecture diagram based on the users, components and applications identified in the previous chapters.

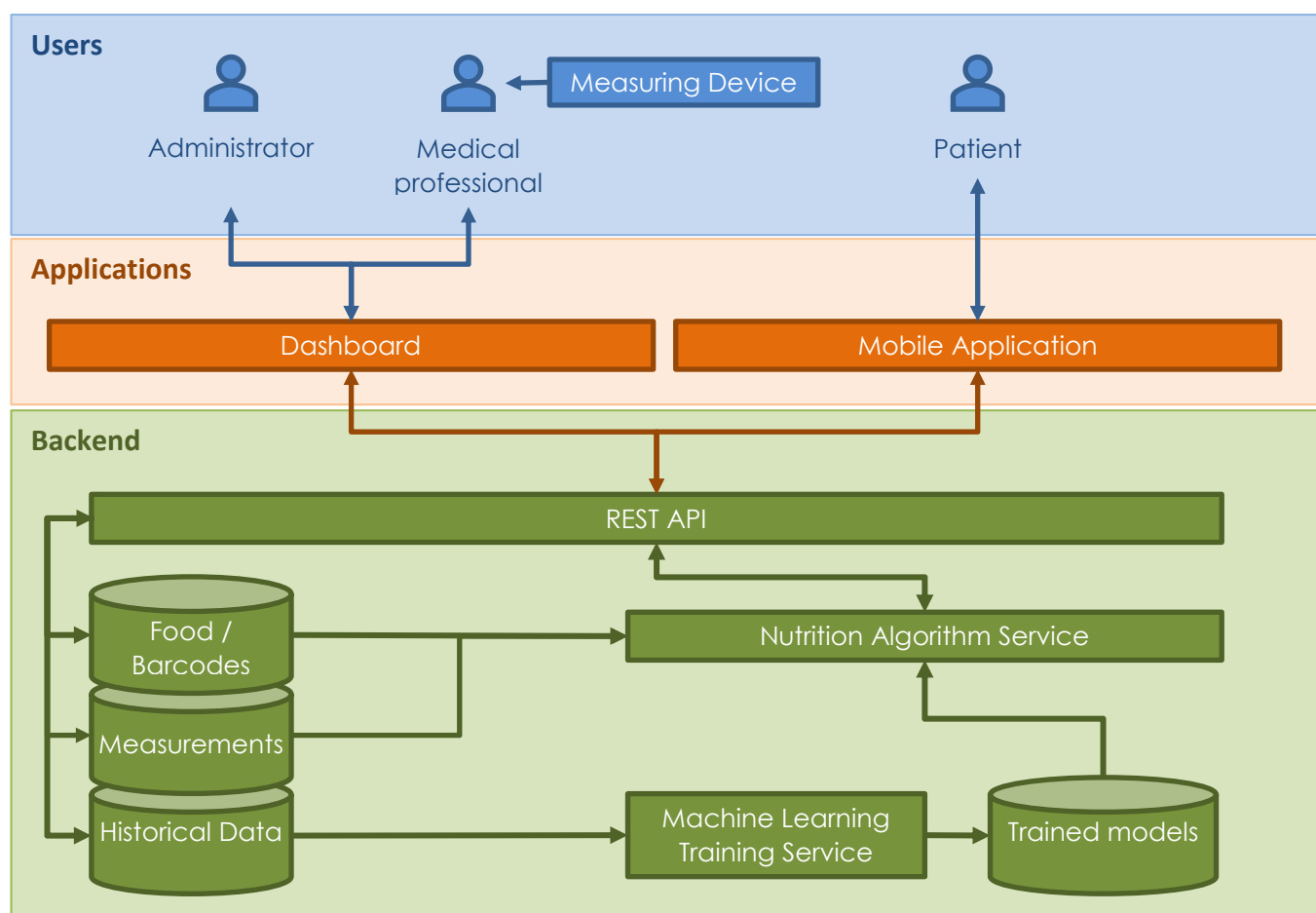


Figure 2 – A preliminary architecture of the NUTRISHIELD platform

### 6.1. Applications

The NUTRISHIELD platform consists of two applications. A web application and a mobile application.

#### 6.1.1. Dashboard

**The Dashboard** is a web application used primarily by medical personnel to monitor patients, upload measurements and prepare the dietary and activity plans. It interacts directly with the backend over the internet. It can be accessed from a web browser.



### 6.1.2. Mobile application

**The Mobile application** is used by patients or their guardians. It is used for receiving notifications regarding their dietary and activity plan and logging the food consumed. It interacts directly with the backend over the internet. It runs on a smartphone which must be connected to the internet, have a camera and enough storage to store the photographs taken.

## 6.2. Backend components

At a high-level view, the NUTRISHIELD system consists of three major components. The dashboard and the mobile application, which are described in the section above, and the backend system.

**The Backend** is a wrapper of many other components such as databases, web services and machine learning modules. It can be considered as NUTRISHIELD's brain. This is where the nutrition algorithm analyses the patient data and offers personalised nutrition suggestions. It interacts with both the dashboard and the mobile application.

### 6.2.1. REST API

This is a web service which exposes the endpoints used by the dashboard and the mobile application. These endpoints can be used to transfer data and commands between the applications and the backend. This service interacts directly with the applications and other backend components.

### 6.2.2. Databases

The backend must store various types of data. These data are stored in different types of databases. Tabular data could be stored in relational SQL databases, while unstructured data could be stored in NoSQL databases.

The data in the databases are populated from the data sent by the dashboard and the mobile applications. They can be patient measurements, user account data, food diary entries, etc. In general, anything that needs to be persisted is stored in the system databases.

Part of the data is used to feed the nutrition algorithm which produces the personalised nutrition suggestions.

### 6.2.3. Nutrition Algorithm Service

This is the service that implements the NUTRISHIELD nutrition algorithm. It processes relevant data stored in the databases and produces personalised nutrition suggestions for the patients. It communicates with the databases to retrieve the relevant data, the REST API which initiates the suggestion generation process and returns the result to the dashboard.

It could also be connected to the classification models of the machine learning component to provide nutritional suggestions based on data classification techniques.



#### 6.2.4. Machine learning component

This component analyses the data collected from the measurements to generate classification models. These classification models could be used by the nutrition algorithm to add an extra layer of personalization in the suggestions. Ideally, it could be trained based on historical data to increase the accuracy of the classification.

#### 6.2.5. Measuring Devices

Measuring devices are used in clinical or lab settings and do not have IP connectivity, thus they are not directly connected to the NUTRISHIELD platform. Raw measurements may be either inputted directly in NUTRISHIELD by lab technicians or analysed and have the analysis results inputted.

### 6.3. Communication layer

All components could exchange data using encrypted communication over the internet. In particular

- **HTTP** could be used by the dashboard and the mobile applications to communicate with the backend over a secure channel TLS (former SSL)
- **Stomp** could be used to issue notifications from the backend to the dashboard
- **Push notifications** could be used to issue notifications from the backend to the mobile application

Data exchange among internal components of the backend could be achieved using specific technology data adapters. This will depend on the technologies that will eventually be used and is something that will be documented on M12 in *D6.1 Specifications of NUTRISHIELD software framework* along with the rest of the system design.

#### 6.3.1. Cross streaming data processing

Data will be input to the system manually by medical professionals using the dashboard. Data can be submitted by typing in values in forms appearing on the dashboard or uploaded to the system as sets of files.

Although a wide range of data will be collected, they will not require real time processing. Thus, cross streaming data processing is handled by storing the different *streams* of data as they are submitted to by analysed at a later time on demand.

#### 6.3.2. Data analysis

The nutrition algorithm service supports two ways of processing data. A rule-based analysis backed up by strong scientific evidence and a statistical analysis of data supported by machine learning algorithms. Both algorithms will digest a subset of the data stored in the system's databases to contribute to personalised nutrition suggestion generation.



### 6.3.3. Interoperability

The NUTRISHIELD platform could implement well documented REST API services to support importing and exporting data. Data could range from anonymised patient data to food databases.

*D1.2 – Data Management Plan* suggests such API/REST services could allow import/export of genomic sequence data to/from public repositories such as the European Nucleotide Archive (EBI-ENA) and the Sequence Read Archive (SRA).

## 6.4. Extensibility

NUTRISHIELD should be modular to allow fast product prototyping and extensible to allow the flexibility to adapt to different business models.

### 6.4.1. Product prototyping

The CI/CD infrastructure proposed supports production grade product prototyping. Using well defined development workflows, the CI/CD infrastructure can support rapid application development and painless feature experimentation.

### 6.4.2. Flexible business models

The modular nature of the NUTRISHIELD platform in conjunction with compliance to web standards and good practices allows new features to be easily adapted. This allows the NUTRISHIELD platform the flexibility to be used in different business models.



## 7. Conclusion

This deliverable analysed the requirements gathered from the DoA, the deliverables submitted in M06 and the discussions among partners. A gap analysis revealed that, in terms of a platform, there is a need to implement the analysis algorithms and the software infrastructure to support them.

The requirements analysis identified a set of users who will use the NUTRISHIELD platform as well as the components it consists of.

Business analysis techniques were used to derive an extensive list of use cases which describe the interactions of the end users with the various system components as well as the interactions between different system components.

The use cases were further broken down to functional and non-functional requirements for the dashboard, the mobile application and the backend of the NUTRISHIELD platform.

Finally, a preliminary architecture of the NUTRISHIELD platform was presented providing more information on the front-end applications and the backend components as well as the data processing and interoperability.

The proposed modular approach and the production grade product prototyping CI/CD techniques used allow extending the platform thus providing flexible opportunities for NUTRISHIELD to be adapted to different business models.



## References

- [1] **Panagiotakos, Demosthenes & Miliadis, George & Pitsavos, Christos & Stefanadis, Christodoulos. (2006).** *MedDietScore: A computer program that evaluates the adherence to the Mediterranean dietary pattern and its relation to cardiovascular disease risk.* Computer methods and programs in biomedicine. 83. 73-7. 10.1016/j.cmpb.2006.05.003.
- [2] **Serra-Majem, Lluís & Ribas, Lourdes & Ngo, Joy & Ortega, Rosa & García, Alicia & Perez-Rodrigo, Carmen & Aranceta, Javier. (2004).** *Food, Youth and the Mediterranean diet in Spain. Development of KIDMED, Mediterranean Diet Quality Index in children and adolescents.* Public health nutrition. 7. 931-5. 10.1079/PHN2004556.
- [3] **NUTRISHIELD, D1.2 – Data Management Plan,** April 2019
- [4] **NUTRISHIELD, D2.1 – Report on requirements for microbiome analysis,** April 2019
- [5] **NUTRISHIELD, D2.2 – Report on requirements for urine analysis,** April 2019
- [6] **NUTRISHIELD, D2.3 – Report on requirements for breast milk analysis,** April 2019
- [7] **NUTRISHIELD, D2.4 – Report on requirements for breath analysis,** April 2019
- [8] **NUTRISHIELD, D2.7 - Clinical Protocols for Clinical Studies in NUTRISHIELD,** June 2019
- [9] **NUTRISHIELD, D9.2 – POPD - Requirement No. 2,** April 2019

## Annex I – Urine analysis metabolites

Metabolite	Biomarker of
Galactitol	Dairy products intake
trimethyl-N-aminovalerate	
Isovalerylglycine	Cheese intake
Tiglylglycine	
Isobutyrylglycine	
Isoflavones	Soy intake (Seeds)
Proline betaine	Citrus fruits intake
Flavonoids	Fruits and vegetables intake
Alkylresorcinol metabolites	Whole grain intake (Cereals)
1-Methylhistidine	Meat intake
3-Methylhistidine	
Anserine	
TMAO	Fish intake
4-O-Methylgallic acid	Coffee and tea intake
Gallic acid	
Chlorogenic acid	
Citrulline	Sugar sweetened beverages intake
Taurine	
Isocitrate	
Resveratrol metabolites	Wine intake
Indoxyl sulfate	Microbiota
p-cresol	
p-cresol sulfate	
Ferulic acid sulfate	
Caffeic acid sulfate	
3-Hydroxyphenyl-propionic acid sulfate	
4-Ethylphenyl sulfate	
Putrescine	
phenylacetylglycine (PAG)	
p-Hydroxyphenyl acetic acid	
3-Indoleacetic acid (3-IAA)	
Phenylpropionylglycine	
Hippuric acid	
L-Tyrosine	
Phenylacetic acid	
Trigonelline	
Phenylacetylglutamine	
Kynurenine	
Bile acids	
SCFA	
BCAA	

*Table 29 – List of metabolites and corresponding biomarkers measured in urine analysis*

## Annex II – Vitamins measured in milk

Vitamin	Molecular List
Vitamin A	Retinol
Vitamin D	Cholecalciferol (D3), Ergocalciferol (D2), 25-hydroxy-D3
Vitamin E	alpha-tocopherol, tocotrienols
Carotenoids	alpha-carotene, beta-carotene, lutein, zeaxanthin
Vitamin C	Ascorbic acid
Vitamin B1	Thiamine
Vitamin B6	Pyridoxal