

Module Interface Specification for Utrition

Team 16, Durum Wheat Semolina

Alexander Moica

Yasmine Jolly

Jeffrey Wang

Jack Theriault

Catherine Chen

Justina Srebrnjak

April 5, 2023

1 Revision History

Date	Version	Notes
January 18, 2023	1.0	Initial Document
March 7, 2023	1.1	Added New Modules - VNV Report
April 3, 2023	1.2	Final Document Revision

2 Symbols, Abbreviations and Acronyms

See SRS Documentation, Semolina (2022b), at <https://github.com/jeff-rey-wang/utrition/blob/3c91ed8d891c50d14bab9dd2f7ddcd5d3d465f56/docs/SRS/SRS.pdf>

Contents

1	Revision History	i
2	Symbols, Abbreviations and Acronyms	ii
3	Introduction	1
4	Notation	1
5	Module Decomposition	1
6	MIS of Application Path Module	3
6.1	Module	3
6.2	Uses	3
6.3	Syntax	3
6.3.1	Exported Constants	3
6.3.2	Exported Access Programs	3
6.4	Semantics	3
6.4.1	State Variables	3
6.4.2	Environment Variables	3
6.4.3	Assumptions	3
6.4.4	Access Routine Semantics	3
6.4.5	Local Functions	3
7	MIS of Home Page Module	4
7.1	Module	4
7.2	Uses	4
7.3	Syntax	4
7.3.1	Exported Types	4
7.3.2	Exported Access Programs	4
7.4	Semantics	4
7.4.1	State Variables	4
7.4.2	Environment Variables	4
7.4.3	Assumptions	4
7.4.4	Access Routine Semantics	4
7.4.5	Local Functions	4
8	MIS of Upload Page Module	5
8.1	Module	5
8.2	Uses	5
8.3	Syntax	5
8.3.1	Exported Types	5
8.3.2	Exported Access Programs	5

8.4	Semantics	5
8.4.1	State Variables	5
8.4.2	Environment Variables	5
8.4.3	Assumptions	5
8.4.4	Access Routine Semantics	5
8.4.5	Local Functions	6
9	MIS of Profile Page Module	7
9.1	Module	7
9.2	Uses	7
9.3	Syntax	7
9.3.1	Exported Types	7
9.3.2	Exported Access Programs	7
9.4	Semantics	7
9.4.1	State Variables	7
9.4.2	Environment Variables	7
9.4.3	Assumptions	8
9.4.4	Access Routine Semantics	8
9.4.5	Local Functions	9
10	MIS of BMI Page Module	10
10.1	Module	10
10.2	Uses	10
10.3	Syntax	10
10.3.1	Exported Types	10
10.3.2	Exported Access Programs	10
10.4	Semantics	10
10.4.1	State Variables	10
10.4.2	Environment Variables	11
10.4.3	Assumptions	11
10.4.4	Access Routine Semantics	11
10.4.5	Local Functions	12
11	MIS of Settings Page Module	13
11.1	Module	13
11.2	Uses	13
11.3	Syntax	13
11.3.1	Exported Types	13
11.3.2	Exported Access Programs	13
11.4	Semantics	13
11.4.1	State Variables	13
11.4.2	Environment Variables	13
11.4.3	Assumptions	13

11.4.4	Access Routine Semantics	13
11.4.5	Local Functions	14
12	MIS of Upload Container Module	15
12.1	Template Module	15
12.2	Uses	15
12.3	Syntax	15
12.3.1	Exported Types	15
12.3.2	Exported Access Programs	15
12.4	Semantics	15
12.4.1	State Variables	15
12.4.2	Environment Variables	15
12.4.3	Assumptions	15
12.4.4	Access Routine Semantics	15
12.4.5	Local Functions	16
13	MIS of Image Upload Module	17
13.1	Module	17
13.2	Uses	17
13.3	Syntax	17
13.3.1	Exported Types	17
13.3.2	Exported Access Programs	17
13.3.3	State Variables	17
13.3.4	Environment Variables	17
13.3.5	Assumptions	17
13.3.6	Access Routine Semantics	17
13.3.7	Local Functions	18
14	MIS of Text Upload Module	19
14.1	Module	19
14.2	Uses	19
14.3	Syntax	19
14.3.1	Exported Types	19
14.3.2	Exported Access Programs	19
14.4	Semantics	19
14.4.1	State Variables	19
14.4.2	Environment Variables	19
14.4.3	Assumptions	19
14.4.4	Access Routine Semantics	19
14.4.5	Local Functions	20

15 MIS of Voice Upload Module	21
15.1 Module	21
15.2 Uses	21
15.3 Syntax	21
15.3.1 Exported Types	21
15.3.2 Exported Access Programs	21
15.4 Semantics	21
15.4.1 State Variables	21
15.4.2 Environment Variables	21
15.4.3 Assumptions	21
15.4.4 Access Routine Semantics	21
15.4.5 Local Functions	22
16 MIS of Navigation Bar Module	23
16.1 Module	23
16.2 Uses	23
16.3 Syntax	23
16.3.1 Exported Types	23
16.3.2 Exported Access Programs	23
16.4 Semantics	23
16.4.1 State Variables	23
16.4.2 Environment Variables	23
16.4.3 Assumptions	23
16.4.4 Access Routine Semantics	23
16.4.5 Local Functions	24
17 MIS of Input Pre-Processing Module	25
17.1 Module	25
17.2 Uses	25
17.3 Syntax	25
17.3.1 Exported Constants	25
17.3.2 Exported Access Programs	25
17.4 Semantics	25
17.4.1 State Variables	25
17.4.2 Environment Variables	25
17.4.3 Assumptions	25
17.4.4 Access Routine Semantics	25
17.4.5 Local Functions	26
18 MIS of Training Dataset Module	27
18.1 Module	27
18.2 Uses	27
18.3 Syntax	27

18.3.1	Exported Constants	27
18.3.2	Exported Access Programs	27
18.4	Semantics	27
18.4.1	State Variables	27
18.4.2	Environment Variables	27
18.4.3	Assumptions	27
18.4.4	Access Routine Semantics	27
18.4.5	Local Functions	28
19	MIS of Image Classification Module	29
19.1	Module	29
19.2	Uses	29
19.3	Syntax	29
19.3.1	Exported Constants	29
19.3.2	Exported Access Programs	29
19.4	Semantics	29
19.4.1	State Variables	29
19.4.2	Environment Variables	29
19.4.3	Assumptions	29
19.4.4	Access Routine Semantics	29
19.4.5	Local Functions	30
20	MIS of Nutritional Data Retriever Module	31
20.1	Module	31
20.2	Uses	31
20.3	Syntax	31
20.3.1	Exported Constants	31
20.3.2	Exported Access Programs	31
20.4	Semantics	31
20.4.1	State Variables	31
20.4.2	Environment Variables	31
20.4.3	Assumptions	31
20.4.4	Access Routine Semantics	32
20.4.5	Local Functions	32
21	MIS of Profile Data Calculation Module	33
21.1	Module	33
21.2	Uses	33
21.3	Syntax	33
21.3.1	Exported Constants	33
21.3.2	Exported Access Programs	33
21.4	Semantics	33
21.4.1	State Variables	33

21.4.2	Environment Variables	33
21.4.3	Assumptions	34
21.4.4	Access Routine Semantics	34
21.4.5	Local Functions	35
22	Appendix	37

3 Introduction

The following document details the Module Interface Specifications for Utrition. Utrition is an application that will provide the nutritional facts for an inputted food item. Users can provide input through text, voice, or image. Utrition will also log past input food data for users to easily view their eating habits and nutritional intake.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at <https://github.com/jeff-rey-wang/utrition>.

4 Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol $:=$ is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | \dots | c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by Utrition.

Data Type	Notation	Description
character	char	a single symbol or digit
integer	\mathbb{Z}	a number without a fractional component in $(-\infty, \infty)$
natural number	\mathbb{N}	a number without a fractional component in $[1, \infty)$
real	\mathbb{R}	any number in $(-\infty, \infty)$

The specification of Utrition uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. Utrition also uses user frontend events to signal some function executions. The type JSON is heavily used to transport data to be displayed in the application interface. In addition, Utrition uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Module Decomposition

The following table is taken directly from the Module Guide document, Semolina (2022a), for this project.

Level 1	Level 2
Hardware-Hiding Module	N/A
Behaviour-Hiding Module	Application Path Module Home Page Module Upload Page Module Profile Page Module BMI Page Module Settings Page Module Upload Container Module Image Upload Module Text Upload Module Voice Upload Module Navigation Bar Module
Software Decision Module	Input Pre-Processing Module Training Dataset Module Image Classification Module Nutritional Data Retriever Module Profile Data Calculation Module

Table 1: Module Hierarchy

6 MIS of Application Path Module

6.1 Module

App

6.2 Uses

NavBar, Home, Upload, Profile, BMI, Settings

6.3 Syntax

6.3.1 Exported Constants

None

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
App	-	App	-

6.4 Semantics

6.4.1 State Variables

None

6.4.2 Environment Variables

path: String

6.4.3 Assumptions

Users will not try to purposefully edit the site path to a nonexistent page.

6.4.4 Access Routine Semantics

App():

- transition: $path := \text{"/"}$
- output: $out := self$
- exception: None

6.4.5 Local Functions

None

7 MIS of Home Page Module

7.1 Module

Home

7.2 Uses

N/A

7.3 Syntax

7.3.1 Exported Types

Home = ?

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
Home	-	Home	-

7.4 Semantics

7.4.1 State Variables

None

7.4.2 Environment Variables

None

7.4.3 Assumptions

None

7.4.4 Access Routine Semantics

Home():

- transition: Page rendered with general information about Utrition
- output: $out := self$
- exception: None

7.4.5 Local Functions

None

8 MIS of Upload Page Module

8.1 Module

Upload

8.2 Uses

UploadContainer

8.3 Syntax

8.3.1 Exported Types

Upload = ?

8.3.2 Exported Access Programs

Name	In	Out	Exceptions
Upload	-	Upload	-
changeDisplay	\mathbb{N}	-	-

8.4 Semantics

8.4.1 State Variables

currentUpload : \mathbb{N}

8.4.2 Environment Variables

None

8.4.3 Assumptions

None

8.4.4 Access Routine Semantics

Upload():

- transition: Page rendered with component from UploadContainer
- output: *out* := *self*
- exception: None

changeDisplay(selected):

- transition: *currentUpload* := selected
- exception: None

8.4.5 Local Functions

None

9 MIS of Profile Page Module

9.1 Module

Profile

9.2 Uses

N/A

9.3 Syntax

9.3.1 Exported Types

Profile = ?

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
Profile	-	Profile	-
handleExplanationClick	-	-	-
getData	-	-	BadResponseError
leftClickForward	-	-	-
leftClickBackward	-	-	-
rightClickForward	-	-	-
rightClickBackward	-	-	-
handleDeleteEntry	N	-	-
handleCancelDelete	-	-	-
handleConfirmDelete	-	-	-
renderConfirmationDialog	-	HTML	-

9.4 Semantics

9.4.1 State Variables

totalcal : JSON

showConfirmationDialog : Boolean

entryToDelete : N

showTable : Boolean

isImageLoaded : Boolean

9.4.2 Environment Variables

None

9.4.3 Assumptions

None

9.4.4 Access Routine Semantics

Profile():

- transition: Page rendered with information from NutritionalData
- output: $out := self$
- exception: None

handleExplanationClick():

- transition: $showTable := \neg showTable$
- exception: None

getData():

- transition: Listen for response from backend, $totalcal := response$
- exception: $(responseData == error) \Rightarrow BadResponseError$

leftClickForward():

- transition: $totalcal.index := totalcal.index + 4$
- exception: None

leftClickBackward():

- transition: $totalcal.index := totalcal.index - 4$
- exception: None

rightClickForward():

- transition: $totalcal.right_{index} := totalcal.right_{index} + 7$
- exception: None

rightClickBackward():

- transition: $totalcal.right_{index} := totalcal.right_{index} - 7$
- exception: None

handleDeleteEntry(entry):

- transition: *entryToDelete* := entry, *showConfirmationDialog* := true
- exception: None

handleCancelDelete(entry):

- transition: *entryToDelete* := null, *showConfirmationDialog* := false
- exception: None

handleConfirmDelete():

- transition: Send deleted index to backend, wait for response, *totalcal* := response, *entryToDelete* := null, *showConfirmationDialog* := false
- exception: (*responseData* == error) \Rightarrow BadResponseError

renderConfirmationDialog():

- transition: Confirmation Dialog is shown
- output: *out* := Confirmation Dialog HTML
- exception: None

9.4.5 Local Functions

None

10 MIS of BMI Page Module

10.1 Module

BMI

10.2 Uses

N/A

10.3 Syntax

10.3.1 Exported Types

BMI = ?

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
BMI	-	BMI	-
handleBirthSexChange	Event	-	-
handleWeightChange	Event	-	-
handleWeightUnitChange	Event	-	-
handleHeightCmChange	Event	-	-
handleHeightFeetChange	Event	-	-
handleHeightInchesChange	Event	-	-
handleHeightUnitChange	Event	-	-
handleAgeChange	Event	-	-
handleActivityLevelChange	Event	-	-
getData	-	-	BadResponseError
handleSubmit	Event	-	-

10.4 Semantics

10.4.1 State Variables

birthSex : String

weight : \mathbb{R}

weightUnit : String

heightCm : \mathbb{R}

heightFeet : \mathbb{R}

heightInches : \mathbb{R}

heightUnit : String

age : \mathbb{N}

activityLevel : String
errorMessage : String

10.4.2 Environment Variables

None

10.4.3 Assumptions

None

10.4.4 Access Routine Semantics

BMI():

- transition: Page rendered with components relevant to editing user settings
- output: *out* := *self*
- exception: None

handleBirthSexChange(e):

- transition: *birthSex* := e.target.value
- exception: None

handleWeightChange(e):

- transition: *weight* := e.target.value
- exception: None

handleWeightUnitChange(e):

- transition: *weightUnit* := e.target.value
- exception: None

handleHeightCmChange(e):

- transition: *heightCm* := e.target.value
- exception: None

handleHeightFeetChange(e):

- transition: *heightFeet* := e.target.value

- exception: None

handleHeightInchesChange(e):

- transition: *heightInches* := e.target.value
- exception: None

handleHeightunitChange(e):

- transition: *heightUnit* := e.target.value
- exception: None

handleAgeChange(e):

- transition: *age* := e.target.value
- exception: None

handleActivityLevelChange(e):

- transition: *activityLevel* := e.target.value
- exception: None

getData():

- transition: Send state variable values to backend, return to Settings Page
- exception: (*responseData* == error) ⇒ `BadResponseError`

handleSubmit(e):

- transition: if state variable(s) empty: *errorMessage* := "Please fill out all fields.",
else: `getData()`
- exception: None

10.4.5 Local Functions

None

11 MIS of Settings Page Module

11.1 Module

Settings

11.2 Uses

N/A

11.3 Syntax

11.3.1 Exported Types

Settings = ?

11.3.2 Exported Access Programs

Name	In	Out	Exceptions
Settings	-	Settings	-
getData	-	-	BadResponseError

11.4 Semantics

11.4.1 State Variables

usersettings : JSON

11.4.2 Environment Variables

None

11.4.3 Assumptions

None

11.4.4 Access Routine Semantics

Settings():

- transition: Page rendered displaying current User Settings values
- output: $out := self$
- exception: None

getData():

- transition: Get currently stored user settings from the backend, *userSettings* := response
- exception: (*responseData* == error) \Rightarrow BadResponseError

11.4.5 Local Functions

None

12 MIS of Upload Container Module

12.1 Template Module

UploadContainer

12.2 Uses

ImageUpload, TextUpload, VoiceUpload

12.3 Syntax

12.3.1 Exported Types

UploadContainer = ?

12.3.2 Exported Access Programs

Name	In	Out	Exceptions
UploadContainer	\mathbb{N}	UploadContainer	-
selectComponent	-	ImageUpload \vee VoiceU- pload \vee TextUpload	-

12.4 Semantics

12.4.1 State Variables

displayedUpload : \mathbb{N}

12.4.2 Environment Variables

None

12.4.3 Assumptions

None

12.4.4 Access Routine Semantics

UploadContainer():

- transition: Component renders one of either ImageUpload, TextUpload, or VoiceUpload
- output: $out := self$
- exception: None

selectComponent():

- output: if *displayedUpload* == 0 : *out* := ImageUpload, if *displayedUpload* == 1 : *out* := VoiceUpload, if *displayedUpload* == 2 : *out* := TextUpload
- exception: None

12.4.5 Local Functions

None

13 MIS of Image Upload Module

13.1 Module

ImageUpload

13.2 Uses

N/A

13.3 Syntax

13.3.1 Exported Types

ImageUpload

13.3.2 Exported Access Programs

Name	In	Out	Exceptions
ImageUpload	-	ImageUpload	-
handleImage	Event	-	-
getData	-	-	BadResponseError

13.3.3 State Variables

image: String

responseData: JSON

13.3.4 Environment Variables

None

13.3.5 Assumptions

The input file is of an appropriate type and not empty. The backend of Utrition will always send a response.

13.3.6 Access Routine Semantics

ImageUpload():

- transition: *image, responseData* := "", ""
- output: *out* := *self*
- exception: None

handleImage(e):

- transition: *image* := path of uploaded image via *setImage(e)*
- exception: None

getData():

- transition: send image path, then listen for a response from backend
setResponseData(response)
- exception: (*responseData* == error) \Rightarrow BadResponseError

13.3.7 Local Functions

setImage(s)

- transition: *image* := *s*
- exception: None

setResponseData(r)

- transition: *responseData* := *r*
- exception: None

14 MIS of Text Upload Module

14.1 Module

TextUpload

14.2 Uses

None

14.3 Syntax

14.3.1 Exported Types

TextUpload = ?

14.3.2 Exported Access Programs

Name	In	Out	Exceptions
TextUpload	-	TextUpload	-
handleFoodItem	Event	-	-
getData	-	-	BadResponseError

14.4 Semantics

14.4.1 State Variables

foodDesc : (String, \mathbb{Z})

responseData: JSON

14.4.2 Environment Variables

None

14.4.3 Assumptions

None

14.4.4 Access Routine Semantics

TextUpload():

- transition: *foodDesc*, *responseData* := "", ""
- output: *out* := *self*
- exception: None

handleFoodItem(e)

- transition: *foodDesc* := the contents of the text fields via *setFoodDesc*(e)
- exception: None

getData():

- transition: send food item, then listen for a response from backend
 setResponseData(response)
 display nutritional output
- exception: (*responseData* == error) \Rightarrow BadResponseError

14.4.5 Local Functions

setFoodDesc((foodName, servings))

- transition: *foodDesc* := (foodName, servings)
- exception: None

setResponseData(r)

- transition: *responseData* := *r*
- exception: None

15 MIS of Voice Upload Module

15.1 Module

VoiceUpload

15.2 Uses

None

15.3 Syntax

15.3.1 Exported Types

VoiceUpload = ?

15.3.2 Exported Access Programs

Name	In	Out	Exceptions
VoiceUpload	-	VoiceUpload	-
handleVoiceInput	Event	-	-
getData	-	-	BadResponseError

15.4 Semantics

15.4.1 State Variables

detectSpeech: String

responseData: JSON

15.4.2 Environment Variables

None

15.4.3 Assumptions

None

15.4.4 Access Routine Semantics

VoiceUpload():

- transition: *detectSpeech*, *responseData* := "", ""
- output: *out* := *self*
- exception: None

handleVoiceInput(e)

- transition: *detectSpeech* := the detected speech input via setDetectSpeech(e)
- exception: None

getData():

- transition: send voice input, then listen for a response from backend
setResponseData(response)
display nutritional output
- exception: (*responseData* == error) \Rightarrow BadResponseError

15.4.5 Local Functions

setDetectSpeech(s)

- transition: *detectSpeech* := s
- exception: None

setResponseData(r)

- transition: *responseData* := r
- exception: None

16 MIS of Navigation Bar Module

16.1 Module

NavBar

16.2 Uses

N/A

16.3 Syntax

16.3.1 Exported Types

NavBar = ?

16.3.2 Exported Access Programs

Name	In	Out	Exceptions
NavBar	-	NavBar	-
changePage	Event		-

16.4 Semantics

16.4.1 State Variables

None

16.4.2 Environment Variables

None

16.4.3 Assumptions

Users will not try to purposefully change the paths for each button.

16.4.4 Access Routine Semantics

NavBar():

- output: $out := self$
- exception: None

changePage():

- transition: $path := "/" \vee "/profile" \vee "/upload"$
- exception: None

16.4.5 Local Functions

None

17 MIS of Input Pre-Processing Module

17.1 Module

InputPreProcess

17.2 Uses

ImageClassification

17.3 Syntax

17.3.1 Exported Constants

None

17.3.2 Exported Access Programs

Name	In	Out	Exceptions
open	String	String	-

17.4 Semantics

17.4.1 State Variables

filePath: String

foodIdentified: String

17.4.2 Environment Variables

None

17.4.3 Assumptions

It is assumed that there exists a valid image file at the provided image file path.

17.4.4 Access Routine Semantics

open(path):

- transition: $filePath := path$
- output: $out := foodIdentified$
- exception: None

17.4.5 Local Functions

None

18 MIS of Training Dataset Module

18.1 Module

TrainingDataset

18.2 Uses

N/A

18.3 Syntax

18.3.1 Exported Constants

None

18.3.2 Exported Access Programs

Name	In	Out	Exceptions
loadData	seq of (seq of \mathbb{Z}), \mathbb{Z}	Dictionary	-

18.4 Semantics

18.4.1 State Variables

imageArray: seq of (seq of \mathbb{Z})
flag: \mathbb{Z}

18.4.2 Environment Variables

None

18.4.3 Assumptions

It is assumed the file path and file type of the CIFAR-100 datasets are respectively constant and standard.

18.4.4 Access Routine Semantics

loadData(array, f):

- transition: *imageArray*, *flag* := *array*, *f*
- output: *out* := Dictionary consisting of image labels and classes used in the machine learning model
- exception: None

18.4.5 Local Functions

- `unpickle(file)`: takes in a file path and opens it into bytestream. Specific dictionary entries are retrieved and returned depending on the filepath that was passed as an argument
- `main()`: used for debugging a single file. Calls `loadData(None, None)` and prints the resulting retrieved dictionary entries.

19 MIS of Image Classification Module

19.1 Module

ImageClassification

19.2 Uses

TrainingDataset

19.3 Syntax

19.3.1 Exported Constants

None

19.3.2 Exported Access Programs

Name	In	Out	Exceptions
startModel	seq of (seq of \mathbb{Z})	String	-

19.4 Semantics

19.4.1 State Variables

weights: seq of (seq of \mathbb{Z})

imageArray : seq of (seq of \mathbb{Z})

foodItem: String

19.4.2 Environment Variables

None

19.4.3 Assumptions

It is assumed that there is a relationship between the uploaded image and the image labels that the machine learning model is aware of. It is also assumed that the food in an uploaded image has a one to one relation with a label that the machine learning model is aware of.

19.4.4 Access Routine Semantics

startModel(array):

- transition: *imageArray* := *array*
- output: *out* := *foodItem*
- exception: None

19.4.5 Local Functions

`tf.compat.v1.train.GradientDescentOptimizer(learning_rate).minimize(loss)`: Execute `GradientDescentOptimizer` and tries to minimize loss by computing the gradients of its trainable variables. Optimizes weights system variable on pass.

20 MIS of Nutritional Data Retriever Module

20.1 Module

NutritionalData

20.2 Uses

N/A

20.3 Syntax

20.3.1 Exported Constants

None

20.3.2 Exported Access Programs

Name	In	Out	Exceptions
getNutritionalData	String	tuple of (food_name: String, calories: String, total_fat: String, saturated_fat: String, cholesterol: String, sodium: String, total_carbohydrate: String, dietary_fiber: String, sugars: String, protein: String, potassium: String)	IllegalArgumentException

20.4 Semantics

20.4.1 State Variables

result: tuple of Strings

20.4.2 Environment Variables

None

20.4.3 Assumptions

None

20.4.4 Access Routine Semantics

getNutritionalData(*food_item*):

- output: *result* := tuple of (food_name: String, calories: String, total_fat: String, saturated_fat: String, cholesterol: String, sodium: String, total_carbohydrate: String, dietary_fiber: String, sugars: String, protein: String, potassium: String)
- exception: (*food_item* \Rightarrow *result* := NULL) \Rightarrow IllegalArgumentException

20.4.5 Local Functions

None

21 MIS of Profile Data Calculation Module

21.1 Module

ProfileData

21.2 Uses

os, datetime InputPreProcess, NutritionalData

21.3 Syntax

21.3.1 Exported Constants

None

21.3.2 Exported Access Programs

Name	In	Out	Exceptions
logData	JSON	-	-
calculateTotalNutrients	JSON	seq of String	-
readFile	-	seq of JSON	-
deleteEntry	\mathbb{N}	-	-
readFileAsJson	-	JSON	-
toMetricWeight	$(\mathbb{R}, \text{String})$	-	-
toMetricHeight	$(\mathbb{R}, \mathbb{R}, \mathbb{R}, \text{String})$	-	-
readUserSettings	-	JSON	-
updateUserSettings	JSON	-	-
totalCaloriesPerDay	Date	\mathbb{R}	-
totalFoodsPerDay	Date	\mathbb{Z}	-
totalCaloriesPerDaySummaryList	-	seq of JSON	-
mostEatenFood	-	String	-
calculateBmi	-	\mathbb{R}	-
calculateRecommendedCalories	-	\mathbb{R}	-

21.4 Semantics

21.4.1 State Variables

None

21.4.2 Environment Variables

None

21.4.3 Assumptions

None

21.4.4 Access Routine Semantics

logData(foodData):

- transition: $\text{nutritionLog}+ = \text{csvRow} : \text{String}$
- exception: None

calculateTotalNutrients(foodData):

- output: $\text{out} := \{\text{foodName} : \text{String}, \text{calories} : \mathbb{R}, \text{totalFat} : \mathbb{R}, \text{saturatedFat} : \mathbb{R}, \text{cholesterol} : \mathbb{R}, \text{sodium} : \mathbb{R}, \text{totalCarbohydrate} : \mathbb{R}, \text{dietaryFibre} : \mathbb{R}, \text{sugars} : \mathbb{R}, \text{protein} : \mathbb{R}, \text{potassium} : \mathbb{R}\}$
- exception: None

readFile():

- output: $\text{out} := \text{seq of Row} : \text{String}$
- exception: None

deleteEntry(index):

- output: $\text{out} := \text{seq of Row} : \text{String}$
- exception: None

readFileAsJson():

- output: $\text{out} := \{\text{timeStamp} : \text{String}, \text{foodName} : \text{String}, \text{calories} : \mathbb{R}, \text{totalFat} : \mathbb{R}, \text{saturatedFat} : \mathbb{R}, \text{cholesterol} : \mathbb{R}, \text{sodium} : \mathbb{R}, \text{totalCarbohydrate} : \mathbb{R}, \text{dietaryFibre} : \mathbb{R}, \text{sugars} : \mathbb{R}, \text{protein} : \mathbb{R}, \text{potassium} : \mathbb{R}\}$
- exception: None

totalCaloriesPerDay(day):

- output: $\text{out} := \sum_{i=0}^{|\text{foods}|-1} \text{food}_i.\text{calories}$
- exception: None

totalFoodsPerDay(day):

- output: $\text{out} := \sum_{i=1}^{|\text{foods}|} 1$

- exception: None

totalCaloriesPerDaySummaryList():

- output: $out := \text{seq of } \{data, sumPerDay, foodsPerDay\}$
- exception: None

mostEatenFood():

- output: $out := mode(\text{seq of foods})$
- exception: None

21.4.5 Local Functions

None

References

- Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.
- Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL <http://citeseer.ist.psu.edu/428727.html>.
- Durum Wheat Semolina. Module guide. <https://github.com/jeff-rey-wang/utrition/blob/39b1a9bda7ec90db43221e17b035e57b7fa29650/docs/Design/MG/MG.pdf>, 2022a.
- Durum Wheat Semolina. System requirements specification. <https://github.com/jeff-rey-wang/utrition/blob/39b1a9bda7ec90db43221e17b035e57b7fa29650/docs/SRS/SRS.pdf>, 2022b.

22 Appendix