Systemtestplan

Project: PharmaPartners

Project team: S3-DB03T group 3

Team members:

Name	E-mail
Bart Blaak	b.blaak@student.fontys.nl
Sibe van Etten	sibe.vanetten@student.fontys.nl
Hugo Mkandawire	h.mkandawire@student.fontys.nl
Maarten Reimus	m.reimus@student.fontys.nl
Kacper Serewiś	k.serewis@student.fontys.nl
Kevin van der Wouw	kevin.vanderwouw@student.fontys.nl

Client: Pharmapartner
Version: 1.0
Version date:
Status: Concept

Documenthistory

Version	Changes	Author	Date
1.0			27-10-2020

Content

C1 Preface

The goal of this application is to make its user more aware of the possible side effects of taking multiple medications Once the user has 2 medicines on their list that can cause problems, the application will show a warning which medication does not go well together.

The function of this document is to get an understanding wich parts of the application we have to test and what the importance is. If the importance is high there are more test that need to secure that everything works. The parts that we are going to test are listed in the analysis document in the Functional requirements.

There are 2 applications that we have to test: the back- and the frontend also, we have to test the communication between those applications.

From all the results that we get from the test we write a separate report.

C2 Baseline situation

In this chapter we describe the baseline situation for the systemtest. The different versions from the software are described below in table 1.
Also all the test data we use to make the test successful is described in a simple and efficient way. We do this to make all our tests reproducible.

The different technologies that we use with the corresponding versions are:

Technology:	**Version:**
Angular	10
Spring boot	2.3.4.RELEASE
Java JDK	11.0.7
Junit	5

Table 1

The test data that we use is described below in table 2.

Model:	**Number:**	**Variable (type):**	**Value:**
Patient	1	Id (UUID)	550e8400-e29b-41d4-a716-446655440000
		Username (String)	Test-Patient
		Email (String)	Test-Patient@hotmail.com
		Password (String)	Test123!
		Emailverified (Boolean)	False
	2	Id (UUID)	5c0e175d-8979-4052-8ea8-4860923365d6
		Username (String)	admin
		Email (String)	admin@admin.nl
		Password (String)	Admin1!
		Emailverified (Boolean)	True
	3	Id (UUID)	5c43175d-8979-4052-8ea8-4852333365r6
		Username (String)	Jan_van_Dijk
		Email (String)	janvandijk@upcmail.com
		Password (String)	kBe(};X+<J_+ba3!>7"hY~!- vV_$/XA(+Swp}vM!,Y`Cw^c@#&/H\=k~\Seg"6RVz>PM2 {jQUpcP:*>9mp!?Hjvv!>xGE6+zZGg_}hBZ<$!YC7xJ3XY
		Emailverified (Boolean)	False
Medicine	1	Id (UUID)	907bf236-c514-441e-a90b-28da1ce5b340

		Name (String)	Paracetamol
		Description (String)	Painkiller in tablet shape.
		Type (int)	2

Table 2

C3 Product Risk Analysis

In this chapter, the product risk analysis is described. The time available for testing is limited; not everything can be tested equally heavily. So choices have to be made. The aim is to distribute the test capacity as effectively and efficiently as possible over the overall test trajectory.

The test strategy records what is going to be tested with which heaviness and aims to find the most important errors as early as possible at the least cost, i.e. with optimal use of available capacity and time.

The first step in drawing up the test strategy is to carry out a product risk analysis. In consultation with the client and other stakeholders, the product risks are identified. The level of risk (the risk class) depends on the failure rate on one side (what are the chances of it going wrong?) and on the other side of the damage for the organization if it does indeed go wrong.

The risk class (RC) then determines the severity of the test. Risk class A is the highest risk class and C is the lowest. The test strategy aims to cover the risks with the highest risk class as early as possible in the test process.

Product risk analysis (PRA) is determined by determining per test target: failure probability, damage, risk class and severity class. The risk class is equal to damage * failure rate. The severity class is based on the risk class. Specify how the severity class is determined. Such as:

Damage: 1 = Low, 2 = Middle, 3 = High
Failure rate: 1 = Low, 2 = Middle, 3 = High
Risk class = Damage * Failure chance
Severity Class C: Risk Class ≤ 2
Severity class B: 2 < risk class ≤ 6 Severity Class A: risk class > 6

Test target	Damage	Failure chance	Risk class	Severity class
Action F1	3	1	3	B
Action F2	3	1	3	B
Action F3	2	1	2	C
Action F4.1	1	1	1	C
Action F4.2	1	1	1	C
Action F5	3	1	3	B
Action F6	3	1	3	B
Action F7	1	1	1	C
Action F8	2	1	2	C
Action F9	1	1	1	C
Action F10	3	1	3	B
Action F11	3	2	6	A

Action F12	3	1	3	B
Action F13	2	1	2	C
Action F14	2	1	2	C
Action F15	1	1	1	C
Action F16	1	1	1	C
Action F17	1	1	1	C
Action F18	1	1	1	C
Action F19	1	1	1	C
Action F20	2	1	2	C
Action NF1	1	1	1	C
Action FN2	1	1	1	C
Rule R1	2	1	2	C
Rule R2	2	1	2	C
Rule R3	2	1	2	C
Rule R4	1	1	1	C
Rule R5	1	1	1	C
Qual.attr. Q1	1	1	1	C
Qual.attr. Q2	1	1	1	C
Qual.attr. Q3	1	1	1	C
Qual.attr. Q4	1	1	1	C
Qual.attr. Q5	1	1	1	C
Qual.attr. Q6	1	1	1	C
Qual.attr. Q7	3	1	3	B
Qual.attr. Q8	1	1	1	C
Qual.attr. Q9	1	1	1	C
Qual.attr. Q10	1	1	1	C
Qual.attr. Q11	1	1	1	C
Qual.attr. Q12	1	1	1	C

C4 Test strategy

In this chapter, on the basis of the product risk analyst, the test strategy (the what) is drawn up and translated into a concrete test suit (the how).
Indicate how you intend to test or hedge the risks per Test goal (see chapter 3). Just describe how you're going to handle this. Running is not part of the command.
For each risk from the product risk analysis, the risk class determines the severity of the test. Risk class A is the highest risk class and C is the lowest. The test strategy is then aimed at covering the risks with the highest risk class as early as possible in the test process.
Design: With a heavier risk class, you want to use more test types and a heavier test per test type. Indicate this in the table below with a ball or asterisk. Such as:
Risk class A must have □□□ in at least one test type, risk class B in at least one test type □□ and risk class C in at least one test type □
Note: This table lists a number of test types, but only as an example. In your project there may be more/less and/or other and/or differently mentioned test types. For 3rd semester, assume minimum unit testing and automatic and manual system testing. Preferably also Component Testing, which allows you to test the individual tests in a distributed system.

Test target	**PRA**	**UT**	**CT**	**IT**	**ST manual**	**ST automatic**
Functionaliteit						
Action F1	B	*		*		
Action F2	B		*	*		
Action F3	C				*	
Action F4.1	C	*				
Action F4.2	C			*		
Action F5	B	*		*		
Action F6	B	*		*		
Action F7	C	*				
Action F8	C	*				
Action F9	C	*				
Action F10	B	*		*		
Action F11	A	*	*	*		
Action F12	B	*		*		
Action F13	C	*				
Action F14	C	*				
Action F15	C				*	
Action F16	C			*		
Action F17	C				*	
Action F18	C				*	
Action F19	C				*	

Action F20	C	*				
Action NF1	C				*	
Action FN2	C	*				
Rule R1	C	*				
Rule R2	C	*				
Rule R3	C			*		
Rule R4	C	*				
Rule R5	C	*				
Qual.attr. Q1	C					*
Qual.attr. Q2	C					*
Qual.attr. Q3	C	n/a				
Qual.attr. Q4	C				*	
Qual.attr. Q5	C					*
Qual.attr. Q6	C					*
Qual.attr. Q7	B			*		*
Qual.attr. Q8	C				*	
Qual.attr. Q9	C		*			
Qual.attr. Q10	C					*
Qual.attr. Q11	C					*
Qual.attr. Q12	C					*
- totaal						
Gebruiksvriendelijkheid						
Performance						
- online						
- batch						
Beveiliging						
Inpasbaarheid						

Explanation of the table above:
PRA-RC Risk class from product risk analysis (PRA): risk table

Keys	Review/review of the various intermediates such as requirements, functional design, technical design
Development test	Unittest en Unitintegratietest
ST	System test
IT	Integration test
CT	Component test
UT	Unit Test
■	Limited dynamic test
■■	Average dynamic test
■■■	Heavy dynamic test

C5 Logical test cases

This chapter describes the logical testcases. A logical testcase is derived from the requirements of the analysis document. In general, multiple logical testcases can be defined for a requirement.

ID	Requirement ID	Description	Expected outcome
LTC-1.1	F1	User tries to register without an email address.	False result
LTC-1.2	F1	User registers with a password of 256 characters.	False result
LTC-1.3	F1	User tries to register with an email that already exists in the system.	False result
LTC-1.4	F1	User tries to register without a password.	False result
LTC-1.5	F1	User registers with a valid email address (that's not already in the DB) and a valid password.	Good result
LTC-1.6	F1	User registers without a username.	False result
LTC-1.7	F1	User registers with an invalid email address.	False result
LTC-2.1	F2	The user tries to login with an invalid email address.	False result
LTC-2.2	F2	The user logs in with the correct email address and password.	Good result
LTC-3.1	F3	The user tries to reset their password by entering the exact same password they already had.	Good result
LTC-3.2	F3	The user resets their password with a new password.	Good result
LTC-3.3	F3	The user tries to reset their password with an invalid password.	False result
LTC-4.1.1	F4.1	The user goes to profile to see their data.	Good result
LTC-4.2.1	F4.2	The user changed his records but forgets leaves an empty email address.	False result
LTC-4.2.2	F4.2	The user changed his personal data all correctly.	Good result
LTC-5.1	F5	The user registrates a new medication.	Good result

LTC-5.2	F5	The user tries to register a new medication but forgets to fill in a few fields.	False result
LTC-5.3	F5	The user tries to register a new medication but this medicine already exists.	False result
LTC-6.1	F6	The user deletes an medicine.	Good result
LTC-7.1	F7	The user edits an medicine.	Good result
LTC-7.2	F7	The user tries to edit a medication but forgets to fill in a few fields.	False result
LTC-8.1	F8	The user goes to medicines to see her/his own medicine list.	Good result
LTC-9.1	F9	The user goes to medicine history to see her/his medicine history.	Good result
LTC-10.1.1	F10.1	The user goes to medical personal data and add their blood pressure, weight and length.	Good result
LTC-10.1.2	F10.1	The user goes to medical personal data and leaves fields open.	False result
LTC-10.1.3	F10.1	The user goes to medical personal data and fills in negative values.	False result
LTC-10.2.1	F10.2	The user goes to medical personal data and changes their blood pressure, weight and length.	False result
LTC-10.2.2	F10.2	The user goes to medical personal data and leaves fields open.	False result
LTC-10.2.3	F10.2	The user goes to medical personal data and fills in negative values.	False result
LTC-11.1	F11	The user goes to check their compatibility. Person 1 is good result and person 2 is a false result.	Good result
LTC-12.1	F12	The user adds one disease from the list of diseases.	Good result
LTC-12.2	F12	The user adds all available diseases.	Good result
LTC-13.1	F13	The user changes one disease into a different one.	Good result.

LTC-14.1	F14	The user removes a disease from their diseases.	Good result
LTC-14.2	F14	The user removes all of their diseases.	Good result

C6 Physical test cases

This chapter describes the physical testcases. A physical testcase is derived from the corresponding logical testcase. Also give the expected result. The "result" column is intended for the test report. This column uses colors to indicate different results. Below is an overview of the colors and their meaning:

	The test has been passed and there is nothing to note.
	The test seems to be going well, but there is something that can do better.
	The test failed.
	This test does not apply to the current version of the application and has not been performed.
	This is not a test but an action that needs to be taken in order to be able to perform subsequent tests (initialization action).

ID	Description	Result
TC-1.1	Register an account. Use the Username and Password from Model ‘Patient 1’. Don’t fill in an email address. Expected result: A message will appear, saying “Please fill in a password.”	
TC-1.2	Register an account. Use the Email, Username and Password from Model ‘Patient 3’. Expected result: a message will pop up saying that the password is too long.	
TC-1.3	Register an account. Use the Email, Username and Password from Model ‘Patient 2’. Expected result: a message will pop up saying that the Email address is already used.	
TC-1.4	Register an account. Use the Email and Username from Model ‘Patient 1’, but don’t fill in a password. Expected result: a message will appear saying “Please fill in a password.”	
TC-1.5	Register an account. Use the Email, Username and Password from Model ‘Patient 1’. Expected result: new user account is registered and directly logged in.	
TC-1.6	Register an account. Use the Email and Password from Model ‘Patient 1’. Don’t fill in a username. Expected result: a message will appear, saying “Please fill in a username”.	
TC-1.7	Register an account. Use the Email, Password and Username from Model ‘Patient 1’. Leave out the ‘@’ from the email address.	

	Expected result: a message appears saying the used email address is invalid.	
TC-2.1	Log in. Use Email, Username and Password from Model ‘Patient 1’, but leave out the ‘@’. Expected result: Message gets displayed, saying “Used email address is invalid.”	
TC-2.2	Log in. Use Email, Username and Password from Model ‘Patient 2’. Expected result: user is logged in.	
TC-3.1	Reset password: Use Model ‘Patient 1’. Make sure the account’s password is already set to the Password of Model ‘Patient 1’. Reset the password and change the password to the exact same password as in Model ‘Patient 1’. Expected result: Password is successfully changed.	
TC-3.2	Reset password. Use Model ‘Patient 1’. Reset the password to: R0tterdam1. Expected result: Password is successfully changed.	
TC-3.3	Try to reset password. Use Model ‘Patient 1’. Reset the password to: rotterdam. Expected result: Message appears, saying “Please use at least 8 characters, 1 number and 1 special character”.	
TC-4.1.1	Go to personal records. Expected result: all records are shown.	
TC-4.2.1	Go to edit and fills in the following values: Username: Henk Email: Password: Henk And the user then clicks on edit. Expected result: Personal records of patient 1 are not changed.	
TC-4.2.2	Go to edit and fills in the following values: Username: Henk Email: HenkieeTankie@gmail.com Password: Henk-Tank! And the user then clicks on edit. Expected result: Your personal records are changed.	
TC-5.1	User registrates medicine. Fill in values of medicine 1. Expected result: New medicine registered.	
TC-5.2	User registrates medicine. Fill in values of medicine 1. Name: Paracetamol Description: Type: 1 Then press the edit button. Expected result: There is a field empty and the medicine is not	

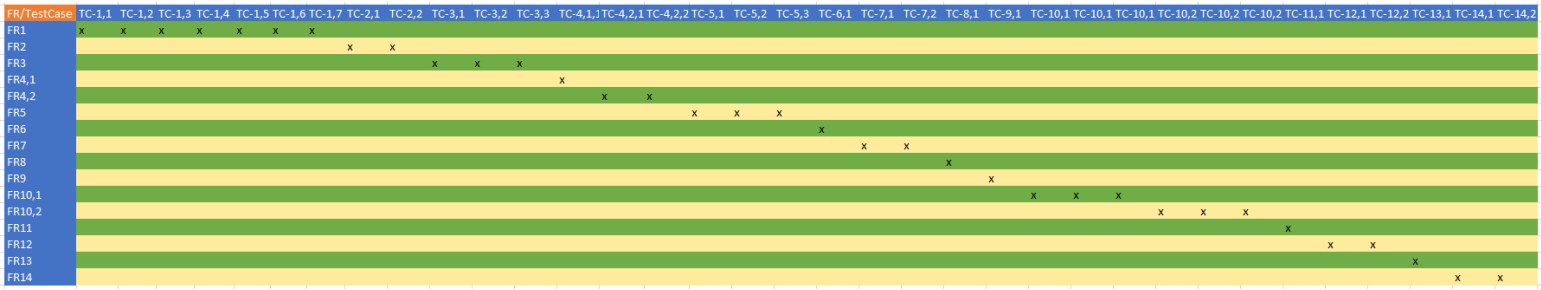
	registered.	
TC-5.3	User registrates medicine. Fill in values of medicine 1 and then press the save button. Expected result: The medicine already exists and is not added to your list of medicines.	
TC-6.1	Go to your list of medicines. After you see the list of medicines search the medicine with the values of medicine 1 and then press delete. Expected result: The medicine is deleted from your list.	
TC-7.1	User edits an medicine. Go to your list of medicines. After you see the list of medicines search the medicine with the values of medicine 1 and then press edit. Then fill in values of medicine 1. Name: Diclofinac, Description: pijnstiller, Type: 3 Then press the edit button. Expected result: The medicine is changed.	
TC-7.2	User edits an medicine. Go to your list of medicines. After you see the list of medicines search the medicine with the values of medicine 1 and then press edit. Then fill in values of medicine 1. Name: Diclofinac, Description: “ ”, Type: “ ” Then press the edit button. Expected result: The medicine is not changed because a few fields don`t have values.	
TC-8.1	Go to your own list of medicines. Expected result: You see a list of your own medicines.	
TC-9.1	Go to your own history of medicines. Expected result: You see a list of your own medicines history.	
TC-10.1.1	Go to add medical personal data. Add the following values to the fields: Blood pressure: 100, Weight: 86, Height: 178 And then press on add data. Expected result: Medical personal data is added.	
TC-10.1.2	Go to add medical personal data. Add the following values to the fields: Blood pressure: 100, Weight: “ ”, Height: 178	

	And then press on change data. Expected result: Medical personal data is added.	
TC- 10.1.3	Go to add medical personal data. Add the following values to the fields: Blood pressure: -11, Weight: -3, Height: -65 And then press on add data. Expected result: Medical personal data is changed.	
TC- 10.2.1	Go to change medical personal data. Add the following values to the fields: Blood pressure: 90, Weight: 96, Height: 175 And then press on change data. Expected result: Medical personal data is changed.	
TC- 10.2.2	Go to change medical personal data. Add the following values to the fields: Blood pressure: 90, Weight: “ ”, Height: “ ” And then press on change data. Expected result: Medical personal data is changed.	
TC- 10.2.3	Go to change medical personal data. Add the following values to the fields: Blood pressure: -11, Weight: -3, Height: -65 And then press on change data. Expected result: Medical personal data is added.	
TC-11.1	Go to check compatibility. Expected result for patient 1: you get a warning. Expected result for patient 2: you don`t get a warning.	
TC-12.1	Go to personal diseases. Add disease “Kidney cancer”. Expected result: “Kidney cancer” is added to diseases.	
TC-12.2	Go to personal diseases. Add all available diseases. Expected result: all diseases are added to you account’s diseases.	
TC-13.1	Go to your account’s added personal diseases. Make sure you have “Kidney cancer” added. Change “Kidney cancer” into “Lung cancer”. Expected result: “Kidney cancer” is changed to “Lung cancer”.	
TC-14.1	Go to your account’s added personal diseases. Make sure “Lung cancer” is added. Remove “Lung cancer”. Expected result: “Lung cancer” is removed.	
TC-14.2	Go to your account’s added personal diseases. Make sure you have at least 5 diseases added. Remove all personal diseases from the account.	

	Expected result: All diseases are successfully removed.	

C7 Test coverage

This chapter describes which requirements are covered with which physical test cases. You do this using a Requirements Traceabilty Matrix.



FR/TestCase	TC-1,1	TC-1,2	TC-1,3	TC-1,4	TC-1,5	TC-1,6	TC-1,7	TC-2,1	TC-2,2	TC-3,1	TC-3,2	TC-3,3	TC-4,1,1	TC-4,2,1	TC-4,2,2	TC-5,1	TC-5,2	TC-5,3	TC-6,1	TC-7,1	TC-7,2	TC-8,1	TC-9,1	TC-10,1	TC-10,1	TC-10,1	TC-10,2	TC-10,2	TC-10,2	TC-11,1	TC-12,1	TC-12,2	TC-13,1	TC-14,1	TC-14,2
FR1	x		x		x		x																											
FR2								x																										
FR3										x		x		x																				
FR4,1													x																					
FR4,2														x		x																		
FR5																	x		x		x													
FR6																			x															
FR7																				x		x												
FR8																						x												
FR9																							x											
FR10,1																								x		x		x						
FR10,2																											x		x		x			
FR11																															x			
FR12																																x	x	
FR13																																		x
FR14																																		x

C8 Unittesten and code coverage

This chapter describes the strategy you have used when writing the unit tests. Describe which classes and methods you will write the unit tests for and why you chose these classes/methods. Show the test results and code coverage in the test report, for example by taking a screenshot. Also describe your conclusions based on the test results and code coverage.

C9 Static code analyse

This chapter describes how you will investigate static code analysis. This can be done with the help of SonarQube. Show the results of the analysis in the test report, for example by taking a screenshot. Also describe your conclusions based on the results and indicate where you see opportunities to further improve the code.

C10 Conclusion *(alleen van toepassing in testrapport)*

Describe here the overall conclusion based on the results of the implementation of the system test, unit tests and static code analysis. Also indicate whether the application can be delivered, i.e. indicate whether the application meets the requirements (functional and non-functional requirements) as described in the analysis document. If not, please indicate what needs to be done before the application can be delivered.