Test Documentation

April 25, 2021

Main Component Unit Tests

The Main Component Table (Table 2) contains the following information regarding the unit tests:

- the *tested function* tells which function (in most of the cases) is tested, otherwise the value of the given *attribute* is under test
- the *test ID* gives a unique reference number to the test, using that we can find the implementation of the test
- the *test scope* is a brief summary of the test
- the *expected output* tells us what should be the result in the case of the *test input*

Other Components Unit Tests

The Other Components Table (Table 1) contains the following information regarding the unit tests:

- the *component* defines the component that it tests
- the *test ID* gives a unique reference number to the test, using that we can find the implementation of the test
- the *test scope* is a brief summary of the test
- the *expected output* tells us what should be the result in the case of the *test input*

E2E Tests

The E2E Table (Table 3) contains tests regarding the following GUI (Figure 1):

Figure 1: GUI

- the *form input* tells which form element is tested
- the *test ID* gives a unique reference number to the test, using that we can find the implementation of the test
- the *test scope* is a brief summary of the test
- the *expected output* tells us what should be the result after the test actions

Table 1: Other Components

Form Input	Test ID	Test Scope	Expected Output
age form	E2E1	age form input is empty	error: You must enter a value!
age form	E2E2	age form input get string	error: Age should be a number!
age form	E2E3	age form input get lower value than 1	error: Age should be a number and bigger than 0!
Subscribe button	E2E4	submit button should be disabled	-
Add glasses button	E2E5	add one glasses	glassesArray size: 1
Add glasses button, removeButtons	E2E6	add one glasses and remove it	glasssArray size: 0
Add glasses button	E2E7	add one glasses with input: asd	error: Price should be a valid number
Add glasses button	E2E8	add one glasses with input: 0	error: Price should be a number bigger than 0!

Table 2: Main Component

Tested Function/Attribute	Test ID	Test Scope	Test Input	Expected Output
component creates	M1	Angular component creates succesfully	-	Angular component is created succesfully
glassesArray	M2	Test the case when there is no glass	-	glassesArray is empty
addGlasses	M3	Add one glasses	-	glassesArray has length 1
addGlasses deleteGlasses	M4	Add one glasses and remove it	-	glassesArray is empty
addGlasses resetGlassesForm	M5	Add 3 glasses to glassesArray, add age and VIPcard value to form and reset form	-	glassesArray is empty, values related are set to initial value
addGlasses resetGlassesForm	M6	Add two glasses and reset form	-	glassesArray is empty, values related are set to initial value
deleteGlasses	M7	Add 5 elements to glassesArray, remove two	-	glassesArray is 3 elements long
deleteGlasses	M8	Add 5 elements, remove the 1st and the 3rd element	indexes 1, 3 to be removed	glassesArray 1st and 3rd element are removed
deleteGlasses	M9	Add 1 element, remove 1 element		glassesArray is empty
getGlassesError	M10	If price is negative, returns error message.	price: -1	returns "Price should be a number bigger than 0!"

getGlassesError	M11	If price is not a number, returns error message.	price: NotANumber	returns "Price should be a valid number!"
getGlassesError	M12	If price is correct, returns empty string.	price: 100	returns "" (no error)
getAgeError	M13	If age is valid, returns empty string.	age: 52	returns "" (no error)
getAgeError	M14	If age is a negative number, returns error message.	-	returns "Age should be a number and bigger than 0!"
getAgeError	M15	If age is not a number, returns error message.	-	returns "Age should be a number!"
getAgeError	M16	If age is missing, returns error message.		returns "You must enter a value!"
ApplyAgeDiscount	M17	apply the given discount for the given age	age:35 price:100	100
ApplyAgeDiscount	M18	apply the given discount for the given age	age:40 price:100	80
ApplyAgeDiscount	M19	apply the given discount for the given age	age:59 price:100	80
ApplyAgeDiscount	M20	apply the given discount for the given age	age:60 price:100	60
ApplyAgeDiscount	M21	apply the given discount for the given age	age:80 price:100	40
ApplyAgeDiscount	M22	apply the given discount for the given age	age:100 price:100	20
ApplyNumberDiscount	M23	apply discount for given nr of glasses with given price	glasses: 1 x 100 eur	100
ApplyNumberDiscount	M24	apply discount for given nr of glasses with given price	glasses: 2 x 100 eur	185
ApplyNumberDiscount	M25	apply rule: -30% from the cheapest glasses	glasses:3, price: 300	270
ApplyNumberDiscount	M26	apply rule: -50% from the cheapest glasses	glasses: 5 x 100	450
ApplyNumberDiscount	M27	apply rule: -100% from the cheapest glasses	glasses: 10 x 100 eur	900
ApplyNumberDiscount	M28	apply rule: -100% from the cheapest glasses	glasses: 12 x 100eur	1100

ApplyNumberDiscount	M29	apply rule: -120% from the cheapest glasses	glasses: 12 x 100 eur	1080
ApplyNumberDiscount	M30	apply rule: -15% from the cheapest glasses	glasses: 1 x 100 eur, 1 x 80 eur	168
ApplyNumberDiscount	M31	apply rule: -120% from the cheapest glasses	glasses: 11 x 100 eur, 1 x 80 eur	1100
logic testing	M32	with 0 glasses the array's size should be 0	-	with 0 glasses the array's size is 0
logic testing	M33	with 0 glasses the price should be 0	-	with 0 glasses the price is 0
logic testing of ApplyAgeDiscount and ApplyNumberDiscount	M34	applies no discount	age: 35, glasses: 1 x 100eur	100
logic testing of ApplyAgeDiscount and ApplyNumberDiscount	M35	applies age and apply discount for given nr of glasses with given price	age: 40, glasses: 2 x 100eur	145
submitForm	M36	Modifies the price according to prices and discount, submits the form.	age: 88 VIPcard: no glasses: 80, 90, 100, 110 eur	120
submitForm	M37	Modifies the price according to prices and discount, submits the form.	age: 65 VIPcard: yes glasses: 80, 90, 100 eur	165
submitForm	M38	Modifies the price according to prices and discount, submits the form.	age: 20 VIPcard: no glasses: 80, 90, 100 eur	246
submitForm	M39	Not counts those glasses form values, that are undefined.	-	Skips those glasses form values, that are undefined.
submitForm	M40	Not counts those glasses form values, that are not a number.	-	Skips those glasses form values, that are not a number.

Table 3: Other Components Table

Component Tested	Test ID	Test Scope	Expected Output
AppComponent	App1	Angular component creates succesfully	Angular component is created succesfully
AppComponent	App2	The app title is always rendered.	The title renders as "glass-store-app".
PageHeaderComponent	Head1	Angular component creates succesfully	Angular component is created succesfully

PageHeaderComponent	Head2	The app title is rendered in the header as html element.	The app title "Glasses Store" is rendered in the header as html element.

Coverage Achieved

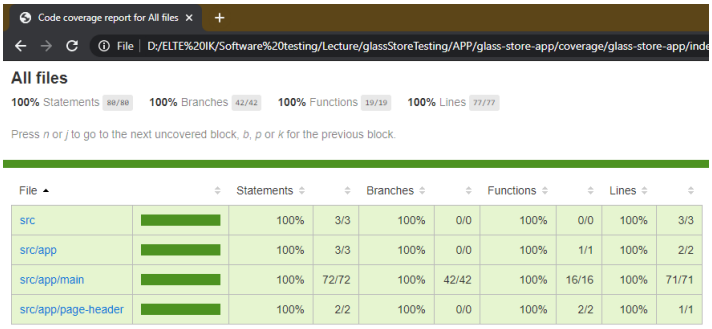

Figure 2: Coverage