P04 - Interactive Scripting

Antonella's library

7th August 2020

Student
Antonella Testa

Index

Scripting requirements document

toggleView() :

Use: I need to toggle the view between user and admin with a variable provided in $scope.adminView. This is achieved by using a boolean which will enable each view.

calculateRaiting() :

Use: I need to calculate the number of stars to display in the HTML.
This function is called when Angular renders each row, returning an array of booleans which will render stars grey or yellow. To find out the rating I pass all reviews from a given book, add all review's ratings together and divide by the amount of review, applying Math.round() to obtain a number without decimals. Once I have the number I use it to iterate over the array of booleans, turning TRUE as many as the number indicates and I return the array to the HTML.

toggleModal() :

Use: this function is used to open every modal in the application.
This is called every time the user clicks a button that requires to open a modal. It expects a modal key and a book.
The modal key is used to toggle the boolean that hides or show the modal. In the $scope I have a boolean for each modal, toggling the corresponding boolean will display the corresponding modal.
Besides this, the modal receives a book. This is for those cases in which I need to open a modal that displays book info. I store a copy of the book in a new variable called "currentBook" in case the user modifies the info but wants to cancel the action, otherwise, the object in the list is updated as well. If the modal does not require a book I store NULL instead of a book.

confirmBooking() :
Use: the application needs to submit the info when the user wants to take a book.
This function is called when the user clicks on the confirmation button, on the Booking Modal, and submits the info using the db.php file.
When the user opens the modal, the info for the selected book is stored in an object in the $scope called “currentBook”, which contains a copy of the book selected. When the user confirms I create the JSON info necessary to send to db.php and then I make the POST request.
On a success response, I call “updateTable()” to populate the table with new info. In case the request fails, I display an Alert with the error.

createTable() :
Use: this function creates the tables in the database.
I have two tables, one to store the books and another one the store the reviews, for each table I make a request to db.php with the corresponding JSON config.
This function is triggered when the user clicks, in the admin view, the action to create the table, confirming in the modal.
When the response of the books table is success I call ”toggleModal()” to hide the modal. In case the request fails, I display an Alert with the error.

uploadList() :
Use: this function uploads a new copy of the book’s list and resets the database.
The JSON file with all the books is located in mcr76_hidden folder in our server and is used to populate the table.
This function is triggered when the user clicks, in the admin view, the action to upload list, confirming in the modal.
When the response of the books table is success I call “updateTable()” to populate the table and after I call ”toggleModal()” to hide the modal. In case the request fails, I display an Alert with the error.

updateTable() :
Use: this function is used to retrieve the info from the database, to populate the application table.
This function is called when the app starts, or every time I do a request that updates the database info.
In this case, I will retrieve the info from the books table. When the response of the books table is success I store the books in $scope.Inventory to render the rows in the table using a ng-repeat, after this I call "getReviews()" function to get the reviews from the other table in the database. In case the request fails, I display an Alert with the error.

getReviews() :
Use: this function is used to retrieve the info from the database, to update the Inventory books with the corresponding reviews.
This function is called every time "updateTable()" is triggered.
In this case, I will retrieve the info from the reviews table. When the response of the books table is success I iterate over $scope.Inventory to get each book. Once I have the book.id I iterate over the reviews I got and check if the book.id matches the review.bookId, if it matches I store the review in the book in the book.Reviews array. In case the request fails, I display an Alert with the error.

deleteTable() :
Use: this function deletes both tables, reviews and book, from the database.
This function is triggered when the user clicks, in the admin view, the action to delete the tables, confirming in the modal.
When the response of the books table is success I call "toggleModal()" to hide the modal and call "updateTable()" to clear the list. In case the request fails, I display an Alert with the error.

deleteRow() :
Use: this function deletes a book from the database.
This function is triggered when the user clicks on the garbage icon displayed in each row of the list. This icon is only present in the admin view, opening a modal to confirm the action.

When the response of the books table is success I call ”toggleModal()” to hide the modal and call “updateTable()” to clear the list. In case the request fails, I display an Alert with the error.

addNewBook() :
Use: this function creates a new book in the books table from the database.
This function is triggered when the user clicks on add book action present in the admin view, opening a modal to complete the info and confirm the action.
Before making the request I populate the JSON from the $scope.currentBook object, and call “uploadImage()” if the user provided an image.
When the response of the books table is success I call ”toggleModal()” to hide the modal and call “updateTable()” to clear the list. In case the request fails, I display an Alert with the error.

modifyBook() :
Use: this function updates an existing book in the books table from the database.
This function is triggered when the user clicks on the pencil icon displayed in each row of the list. This icon is only present in the admin view, opening a modal to complete the info and confirm the action.
Before making the request I populate the JSON from the $scope.currentBook object, and call “uploadImage()” if the user provided an image.
When the response of the books table is success I call ”toggleModal()” to hide the modal and call “updateTable()” to clear the list. In case the request fails, I display an Alert with the error.

uploadImage():
Use: this function is called when I need to upload an image in the server.
This function is called every time I create or modify a book, and the user provides an image. I send the file as a parameter and create a FormData() to send it. The config in the JSON file is pointing to public_html/mcr76/library/images/ to be able to display the file when the user opens a modal that displays a cover.
This is the only request I didn’t know how to make it work with Angular HTTP, and because of this I implemented jQuery ajax.

returnBook() :
Use: this function changes the status of a book back to available, deleting the date in which the book was taken.
This function is triggered when the user clicks on the return button. This button is present in each row of those books that are taken and is only visible in the user view.
When the user clicks in the button a modal appears in which is possible to assign a rating and a review comment. When the user clicks on the confirm action this function is triggered, storing the review in the reviews table, with the bookId, and also updates the book in the books table.
When the response of the books table is success I call ”toggleModal()” to hide the modal and call “updateTable()” to clear the list. In case the request fails, I display an Alert with the error.

calculateFee() :
Use: I need to calculate the current fees to display in the HTML in those books taken.
This function is called when Angular renders each row, returning a value based on the difference between the book.DateRental and the current date.
Once I find the number of days I divide them by 7, implementing Math.floor() to find the minimum number removing decimals, and multiply this by .50 to find the ongoing fees, then I return this value to the HTML where I implement currency pipe from Angular to display the currency symbol and format.

sumFees() :
Use: I need to calculate the total amount of fees collected for all books.
This function is called when Angular renders fees modal, returning the sum of all books with fees.
To do this I iterate over the inventory and add the fees when the books have, then I return this value to the HTML where I implement currency pipe from Angular to display the currency symbol and format.

checkErrorResponse():
Use: I wanted to have a function to contain the logic to display errors and avoid repeat some code.
This is a generic function passed in each HTTP request in places where I need to listen for

errors and nothing else.
The function checks the response status and displays an alert with the message if necessary.

Testing validation plan

Test Case	Area of Input	Expected Outcome	Actual Outcome	√ or X	Action Taken
Delete table button	Click delete button and confirm modal.	The table should be deleted in the database.	Work as expected.	√	
Create table button	Click create table button and confirm modal.	The table should be created in the database.	Work as expected.	√	
Upload content button	Click upload list button and confirm modal.	The table should be populated in the database.	Work as expected.	√	
Add book button	Click add book button, complete modal and submit.	The book should be added in the table, the image should be stored in the folder, and list should reload.	Work as expected.	√	
Delete book button	Click delete icon button in row and confirm modal.	The book should be removed from database and list should update.	Work as expected.	√	
Modify book button	Click pencil icon button in row and update information in the modal.	The book should be updated in the database and list should update.	Work as expected.	√	
Toggle view button	Click in toggle button in the footer.	The interface should toggle users view.	Work as expected.	√	

Inputs to filter content	Complete inputs to filter results.	The list should filter the results matching criteria.	Work as expected.	√	
Book checkout	Click book button in row and confirm modal.	The book should be updated in the database and list should be updated.	Work as expected.	√	
Book return	Click return button in row, complete review and confirm modal.	The book should be updated in the database, and review should be stored in review table and list should be updated to reflect status with the rating.	Work as expected.	√	
Fees collected	Click fees collected button and see total of fees collected.	All books fees should be listed in the modal, showing the total.	Work as expected.	√	

Maintenance plan

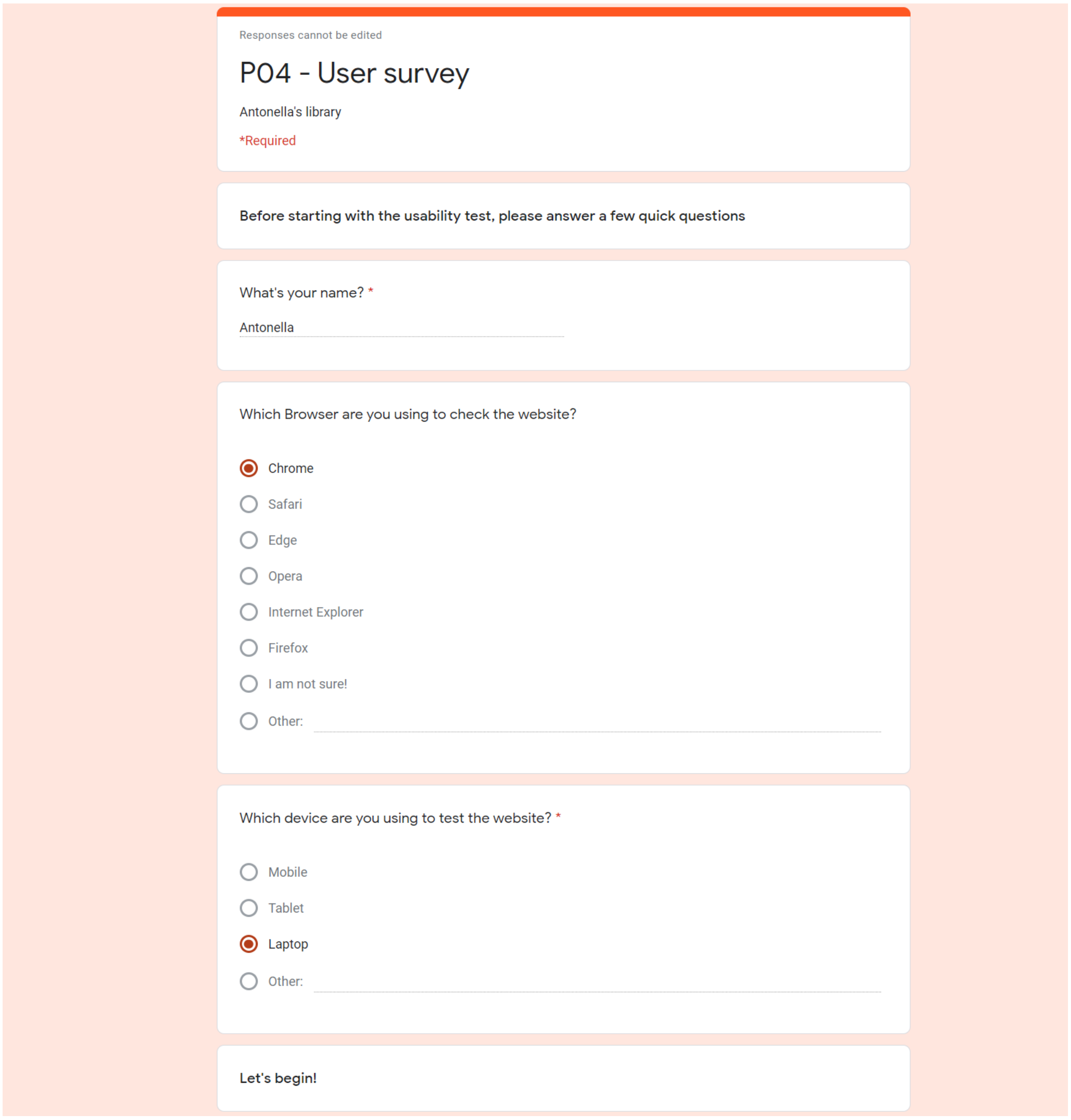
In order to keep my application Antonella's library working, I will check it for updates and modifications every 3 months.

I will ensure to:

- Regular clean up of database to ensure storage capacity is not reached out.
- Assessment database every month to ensure data quality.
- Check for new libraries realises to be up to date with latest security improvements.
- Provide surveys to new users to ensure usability and performance in the app.
- Update UI to make it mobile-friendly.

Survey

User 1:



Responses cannot be edited

P04 - User survey

Antonella's library

*Required

Before starting with the usability test, please answer a few quick questions

What's your name? *

Antonella

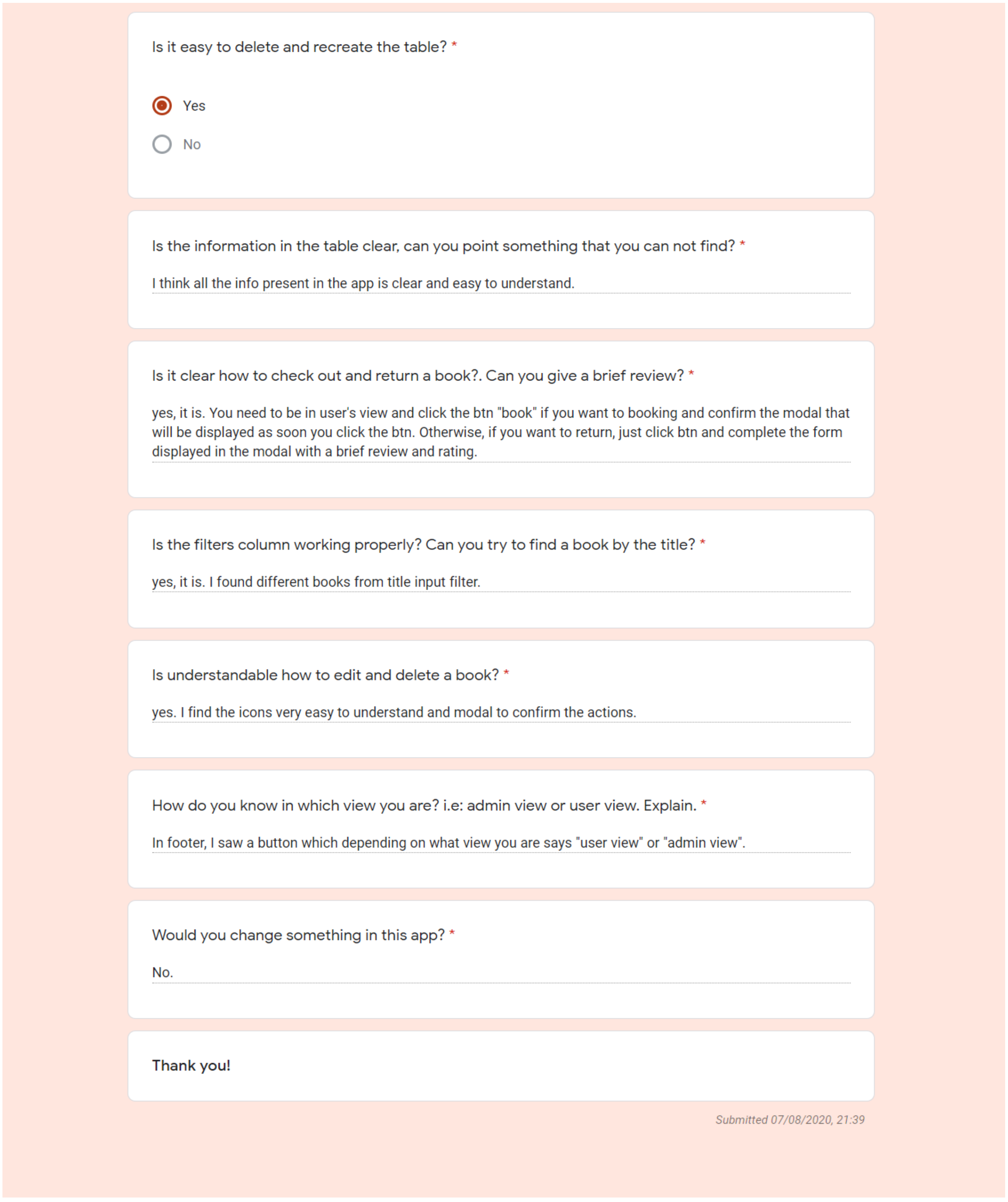
Which Browser are you using to check the website?

[✓ Chrome] Safari Edge Opera Internet Explorer Firefox I am not sure! Other:

Which device are you using to test the website? *

Mobile Tablet [✓ Laptop] Other:

Let's begin!


Is it easy to delete and recreate the table? *

[✓ Yes] No

Is the information in the table clear, can you point something that you can not find? *

I think all the info present in the app is clear and easy to understand.

Is it clear how to check out and return a book?. Can you give a brief review? *

yes, it is. You need to be in user's view and click the btn "book" if you want to booking and confirm the modal that will be displayed as soon you click the btn. Otherwise, if you want to return, just click btn and complete the form displayed in the modal with a brief review and rating.

Is the filters column working properly? Can you try to find a book by the title? *

yes, it is. I found different books from title input filter.

Is understandable how to edit and delete a book? *

yes. I find the icons very easy to understand and modal to confirm the actions.

How do you know in which view you are? i.e: admin view or user view. Explain. *

In footer, I saw a button which depending on what view you are says "user view" or "admin view".

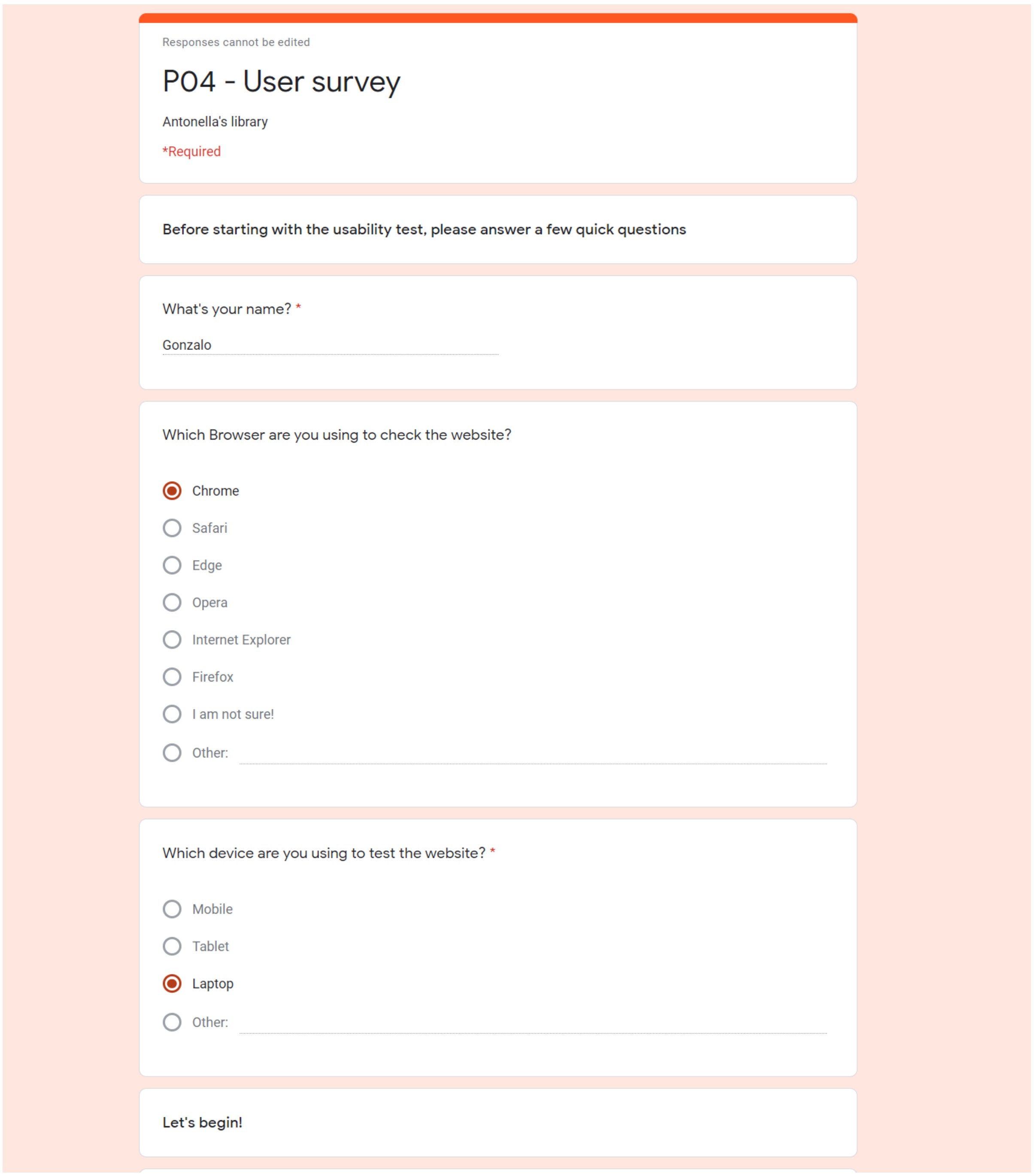
Would you change something in this app? *

No.

Thank you!

Submitted 07/08/2020, 21:39

User 2:



Responses cannot be edited

P04 - User survey

Antonella's library

*Required

Before starting with the usability test, please answer a few quick questions

What's your name? *

Gonzalo

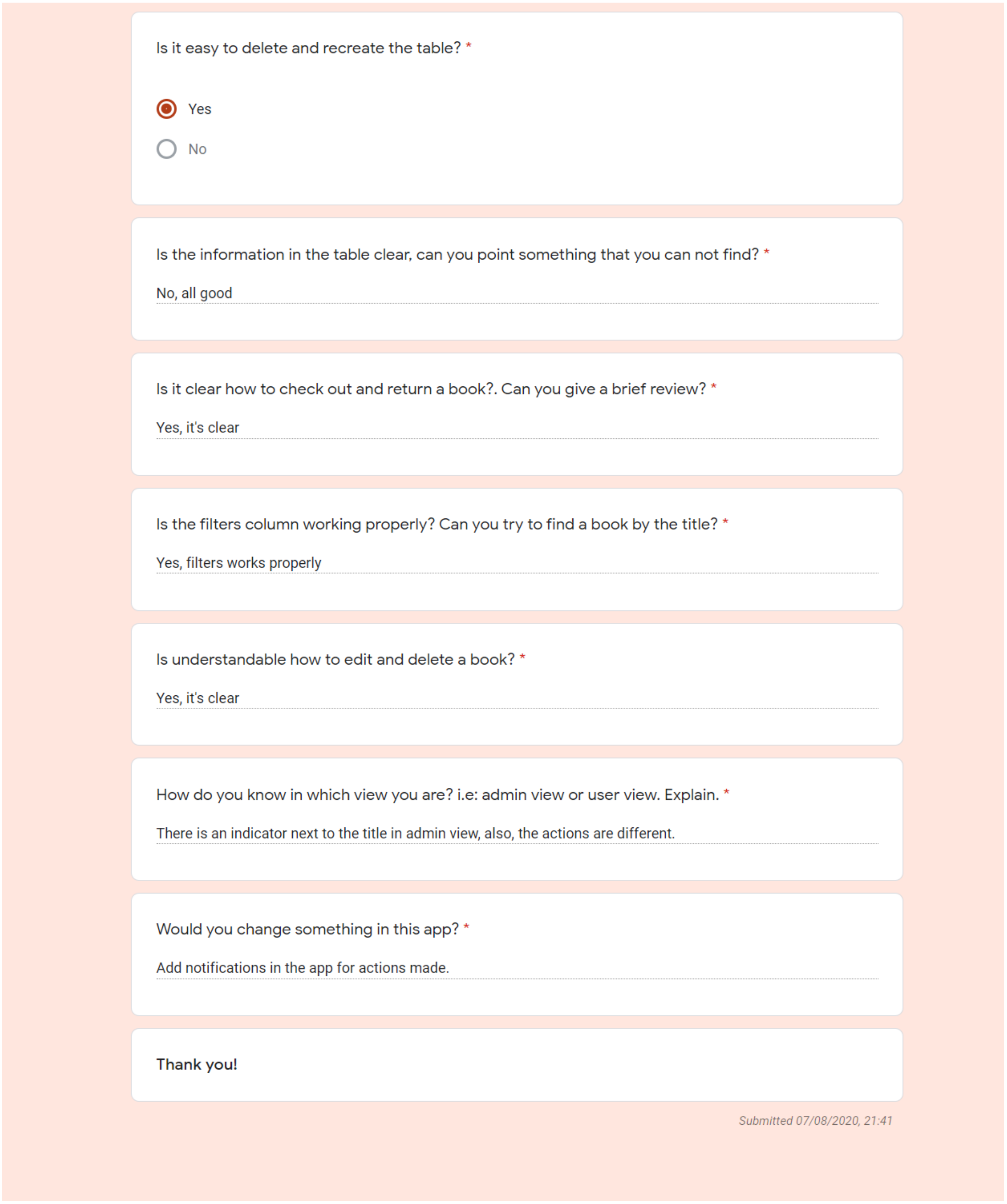
Which Browser are you using to check the website?

[✓ Chrome]
Safari
Edge
Opera
Internet Explorer
Firefox
I am not sure!
Other:

Which device are you using to test the website? *

Mobile
Tablet
[✓ Laptop]
Other:

Let's begin!



Is it easy to delete and recreate the table? *

[✓ Yes] No

Is the information in the table clear, can you point something that you can not find? *

No, all good

Is it clear how to check out and return a book?. Can you give a brief review? *

Yes, it's clear

Is the filters column working properly? Can you try to find a book by the title? *

Yes, filters works properly

Is understandable how to edit and delete a book? *

Yes, it's clear

How do you know in which view you are? i.e: admin view or user view. Explain. *

There is an indicator next to the title in admin view, also, the actions are different.

Would you change something in this app? *

Add notifications in the app for actions made.

Thank you!

Submitted 07/08/2020, 21:41