WEBSITE ENHANCEMENT REPORT

CHURCH OF CHRIST WEBSITE

PRESENTED BY

Cristian Betancourt

COURSE

INTRODUCTION TO JAVASCRIPT

NORTH ISLAND COLLEGE

COMOX VALLEY, APRIL 9-2024

Table of Content

Website Background Information

URL Original: https://churchofchristcourtenay.org/
HTML & CSS Course: https://nic-dgl103-f23-cvs1.github.io/project-crisbux/

Purpose of the Website

This site was created with the aim to show the Church, its thoughts, and positions known, and also to be useful in attracting people seeking a place for spiritual growth.

Why this site?

First of all, the site is not available for different devices, the page appears as a common desktop site even on responsive tablet and mobile pages. Then, the site is old and uses basic elements with limitations to be a scalable website.

Also, the content is accumulated on the same screen, so it could also improve the organization and distribution in a better information architecture.

Target audience on this website

The site of this Christian church seeks to provide the necessary information for people, believers or not, interested in spiritual growth, telling their thoughts, spiritual foundations, creed, ministries and activities.

Additionally, it seeks to offer contact information and location, as a means to respond to inquiries, offer help or simply to inform about meeting times and spaces.

It would be ideal if people had the desire to understand and know more about the church and/or the faith. That they can feel comfortable and see that the church can be a safe and comfortable place to explore spirituality.

Features

Verses Animation
System of verses that will show different verses all the time.

JavaScript events
By responding to user actions, to change and navigate Hero images. Included animation.

Form Validation

Validate data entered by users in forms. This helps ensure that correct data is provided before sending the information to the server.

JavaScript Techniques Used

Verses Animation

1. **Arrays Creation:** I decided to create an array that stores the verses, each one of them are text for a verse and also for a reference.

2. **Index Tracking:** I added the variable *currentVerseIndex* to track which verse is currently being displayed. It starts at 0, indicating the first verse in the array.

3. **DOM Elements:** The *verseElement* and *referenceElement* variables are references to HTML elements with the IDs "verse" and "reference". These elements are where the verse text and its reference will be displayed.

4. **Display a Verse:** The *showVerse* function takes an index as an argument. It first sets the opacity of *VerseElement* to 0, making it invisible. And after the timeOut it sets the opacity back to 1, making the verse visible, including the reference.

5. **Navigating Verses:** The showNextVerse and showPreviousVerse functions are used to navigate through the verses.

   showNextVerse increments currentVerseIndex by 1
   showPreviousVerse decrements currentVerseIndex by 1

6. **TimeOut:** Using setTimeOut function to create a delay before displaying a new verse. This works as a smooth transition effect between the different verses and references, included in the array.

Hero Animation and user Interaction

This feature includes 2 different tasks, an automatic animation but also controls to navigate the images hero.

1. **Selecting Elements:** Using document.getElementById the solution selects the html elements, in this case (heroContainer) and also buttons for navigation to the previous and next buttons:

```
const prevBtn = document.getElementById("prevBtn");
const nextBtn = document.getElementById("nextBtn");
```

2. **Image Array:** I create an Array named *images,* to store the URL's (location) of the hero images to be displayed in the carousel.

3. **Current Image:** Using the variable *currentImageIndex* the solution tracks which image is currently being displayed.

4. **Change Image:** With the *changeImage* function updates the background of the *hereContainer* to the image specified by *currentImageIndex.*

5. **Navigating Images:** The showPreviousImage and showNextImage functions are used to navigate through the images.

6. **Automatic Image Change:** The *autoChangeImage* function is set to be called every 5000 milliseconds using *setInterval.*

7. **Event Listeners:** as a *prevBtn* and *nextBtn* buttons call *showPreviousImage* or *showNextImage* when user click in buttons.

Form Validation

1. **Selecting Elements:** Using *document.getElementById('contactForm');* JavaScrit call to the HTML form by ID to add an event listener for the *submit* event (Button).

2. **Form Submission Event:** When the form is submitted, the function defined within the event listener is executed. This function performs several checks to validate the form inputs (mandatory).

3. **Input Validations:** The decision validates if the name, the email, and the message inputs field are empty. If it is, an error message is displayed as a feedback to the user who needs to fill the inputs to do the submission.

   As well, I added a validation for the phone, just in case that the checkbox "Contact me by phone" is checked by the user.

4. **Event Preventing:** If the before validation error is true, the form submission stops until all errors are corrected.

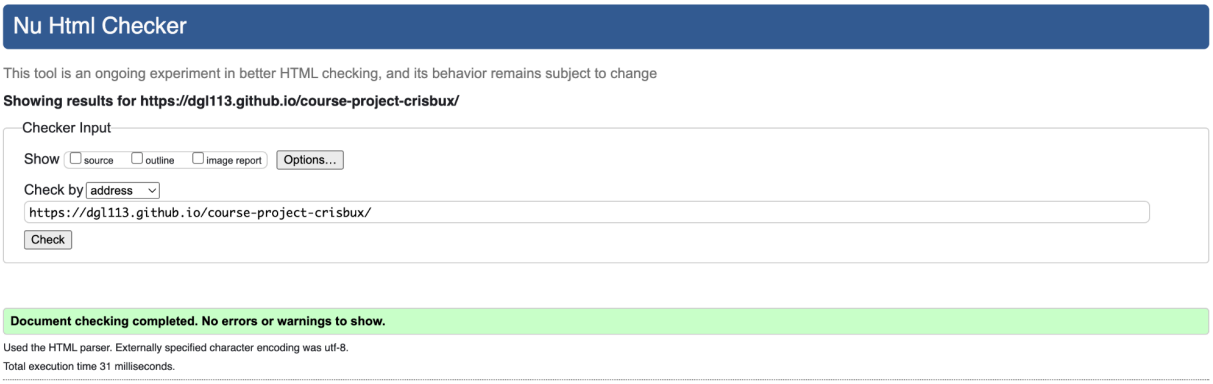
5. **Successful Submission:** if there are no errors, an alert is shown to the user indicating the submission was successful. At this point, I had to add a reset to clear the inputs

because when I successfully sent, the fields remained complete. And also, another *event.preventDefault();* because when the form was restarted it was throwing *Error 405*.

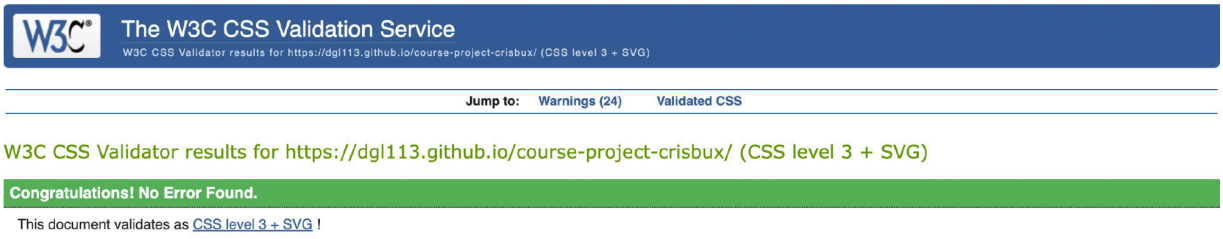
HTML and CSS Validation Result

Both HTML and CSS validation were successful. Here are the results:

HTML



CSS

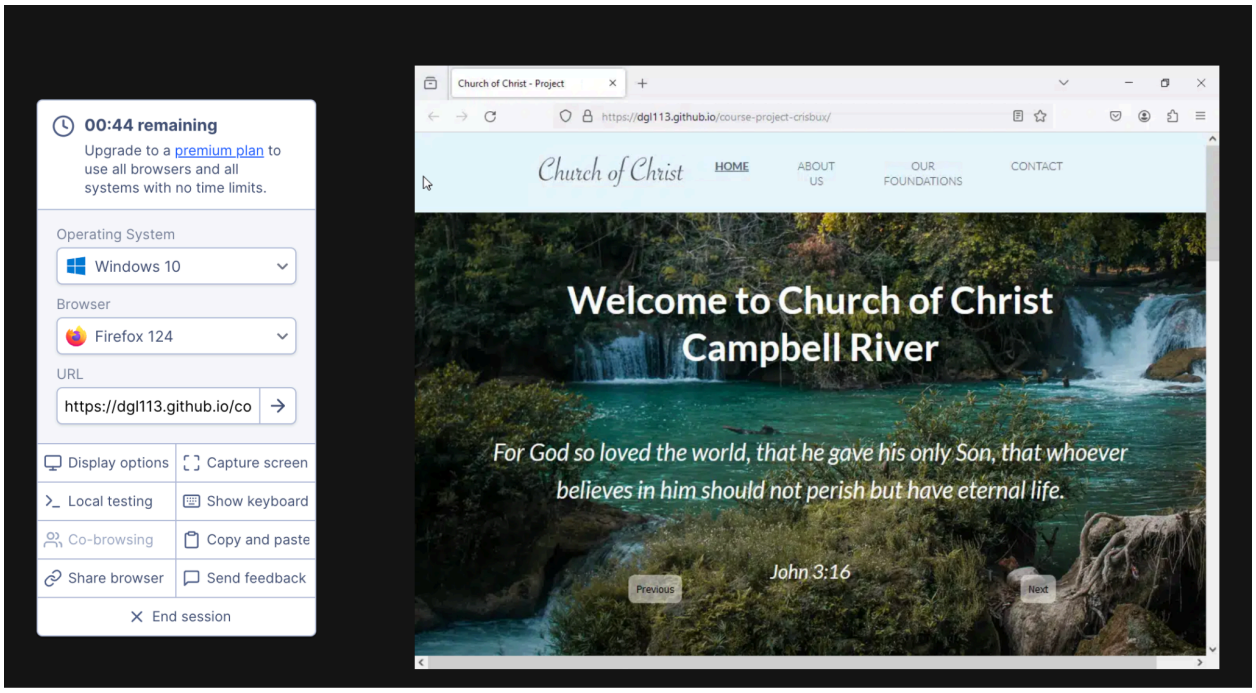


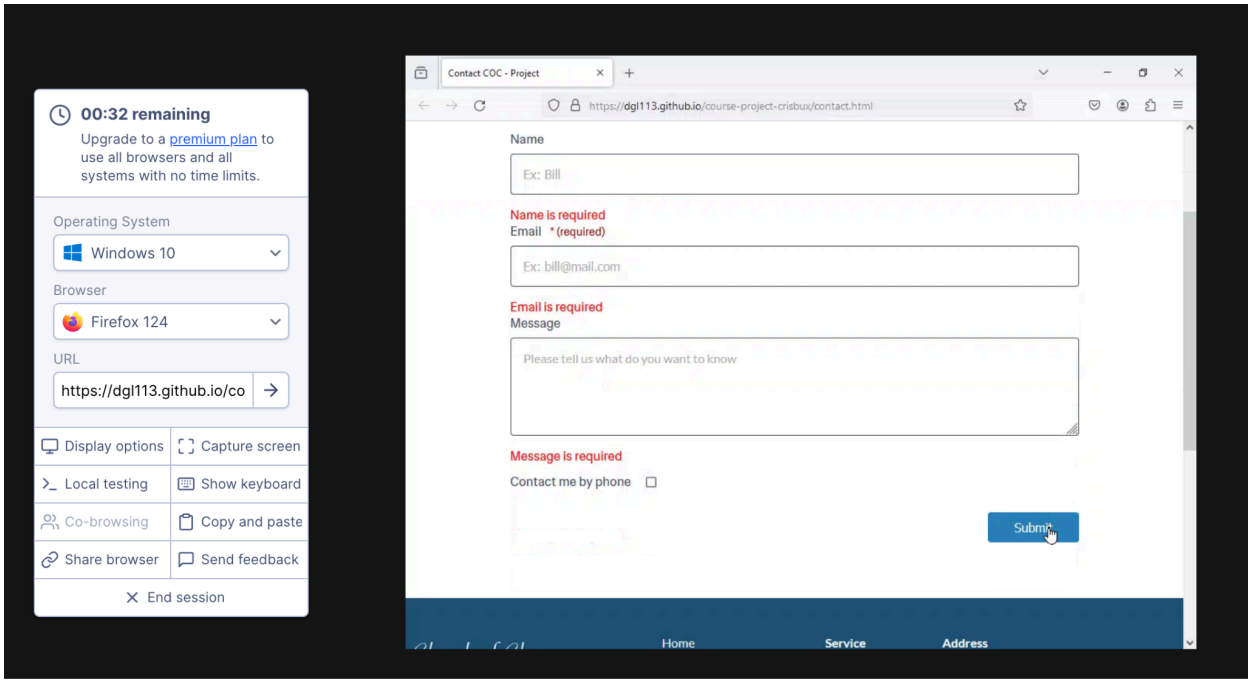
Browser Testing Result

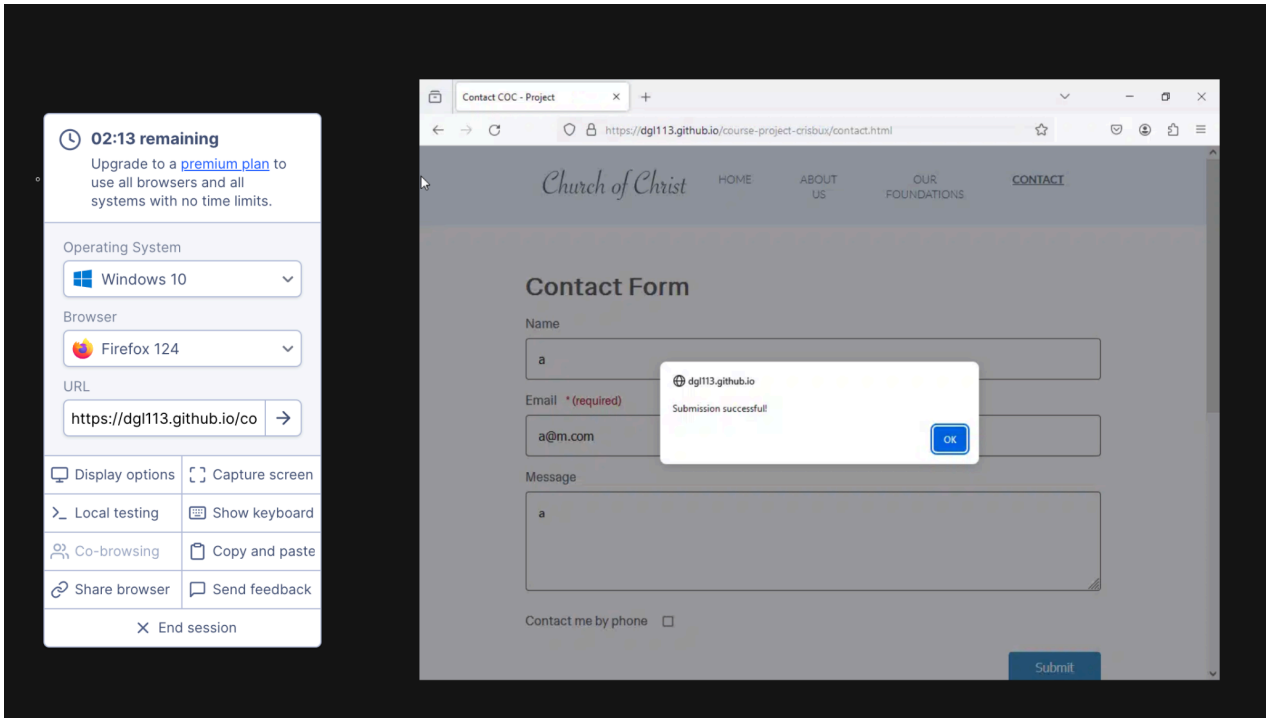
Browser validation was performed using https://www.browserling.com/ and https://app.lambdatest.com/, which was satisfactory in FireFox. Additionally I did a manual validation in Chrome and Safari, where the results were also good. The site and the components worked according to the expected behavior.

Attached are screenshots of the validations:

Using Web Tool - Windows10/FireFox

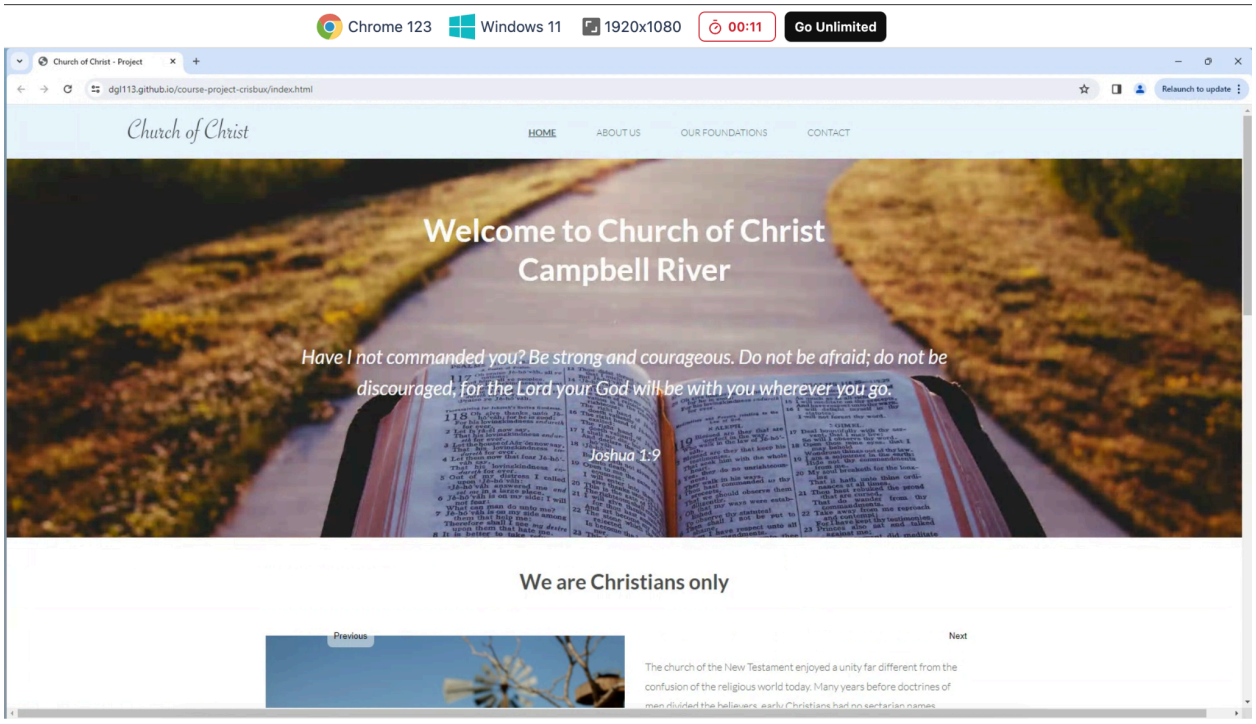




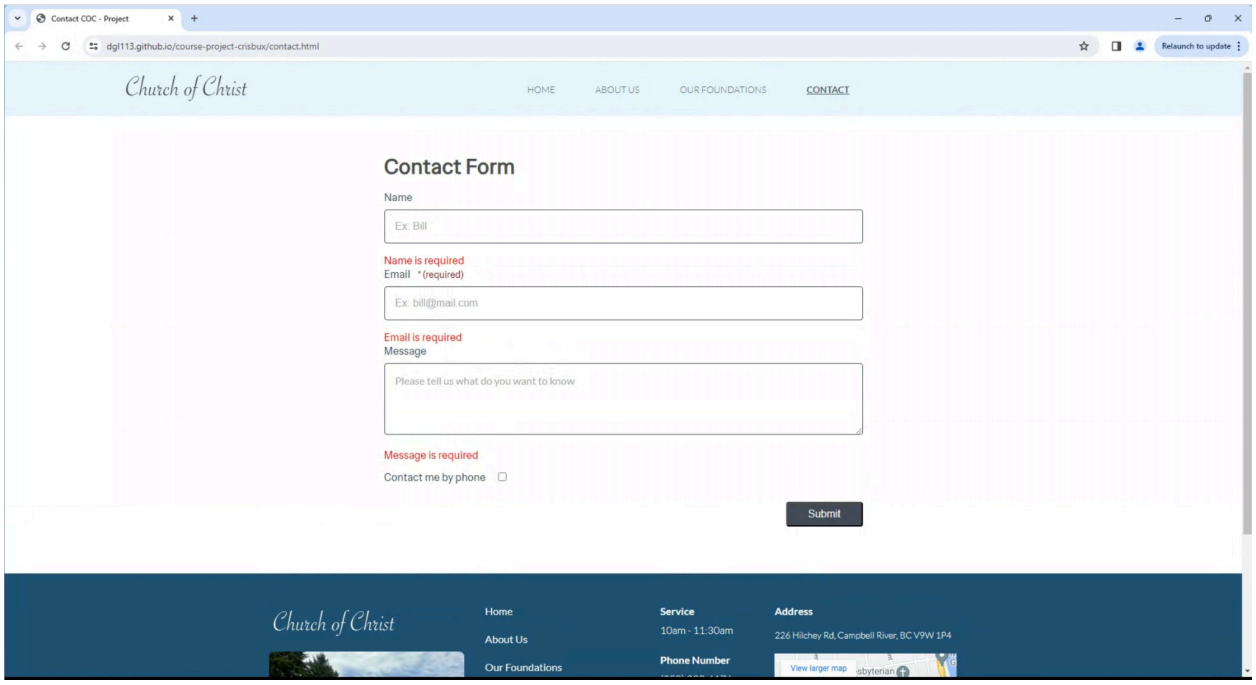


Using Web Tool - Windows11/Chrome

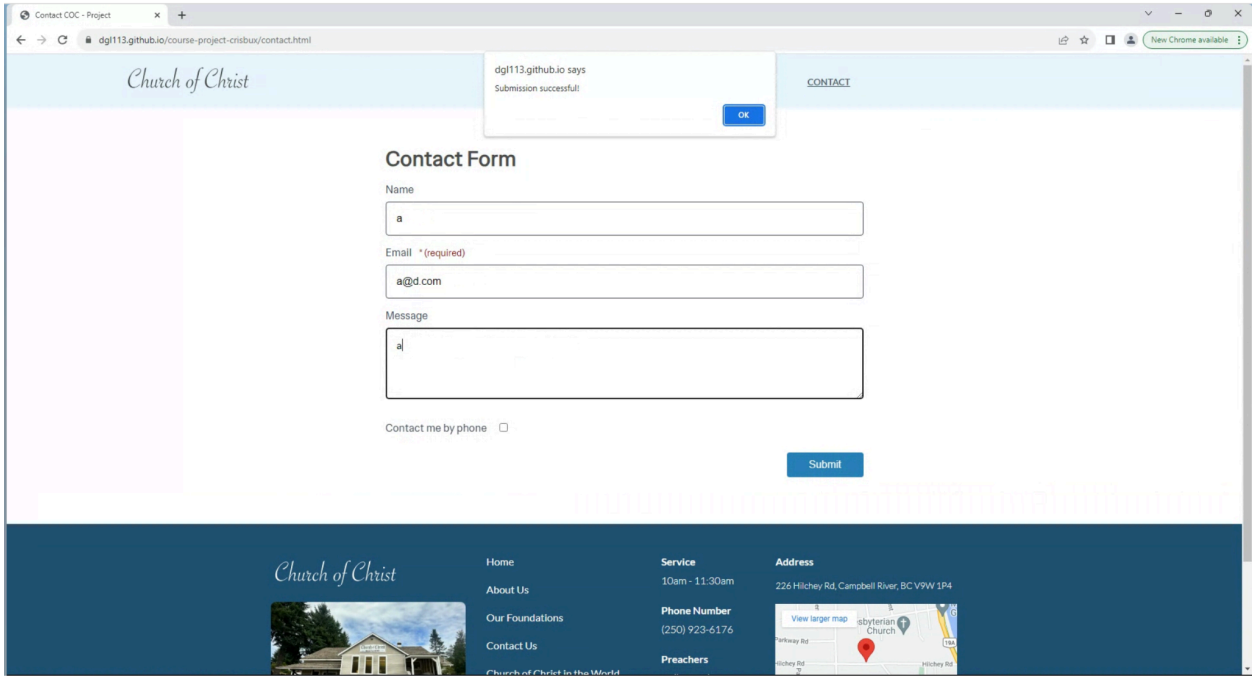
Manual Validation: Safari



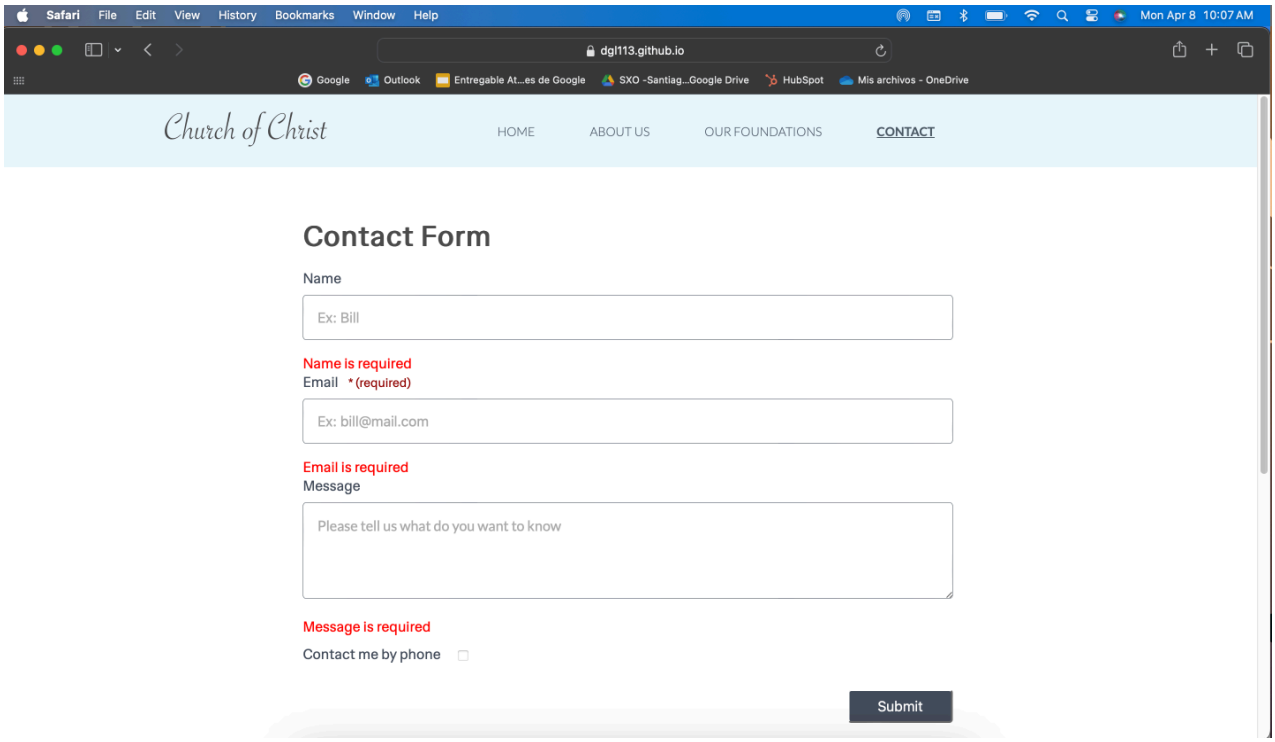
Chrome 123 Windows 11 1920x1080 00:31 Go Unlimited

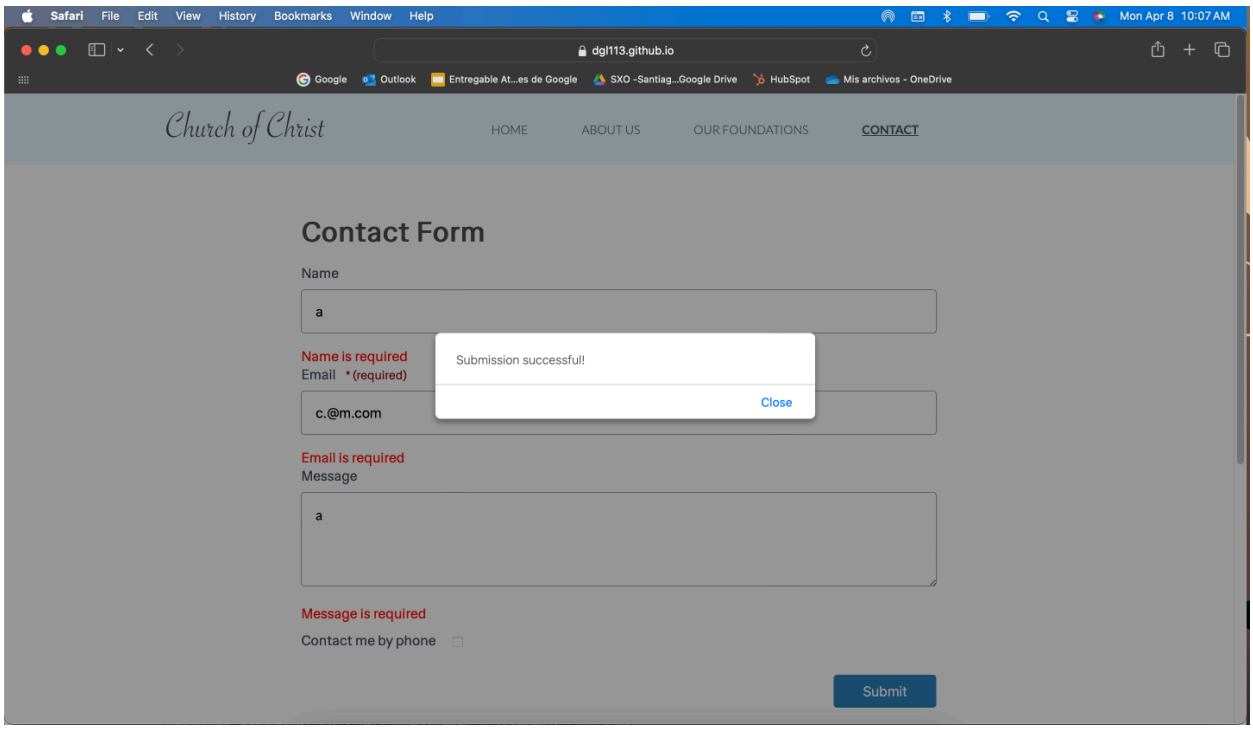


Chrome 121 Windows 11 1920x1080 01:31 Go Unlimited

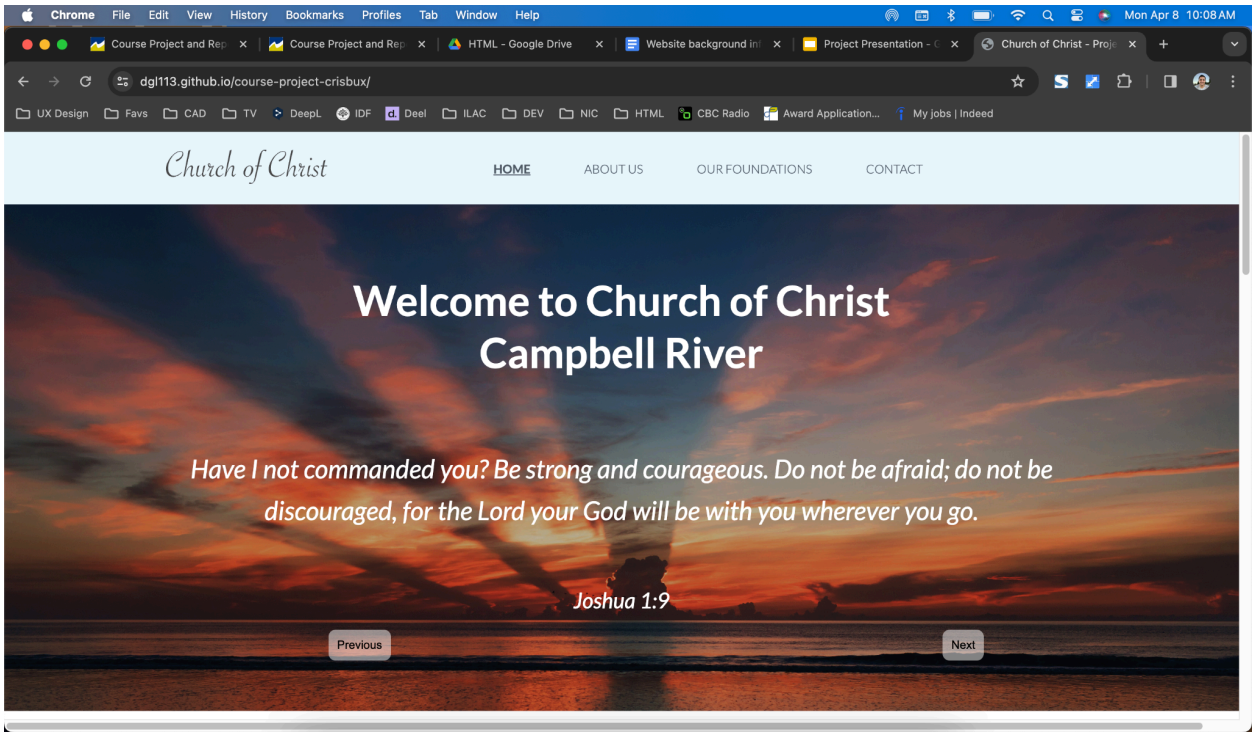


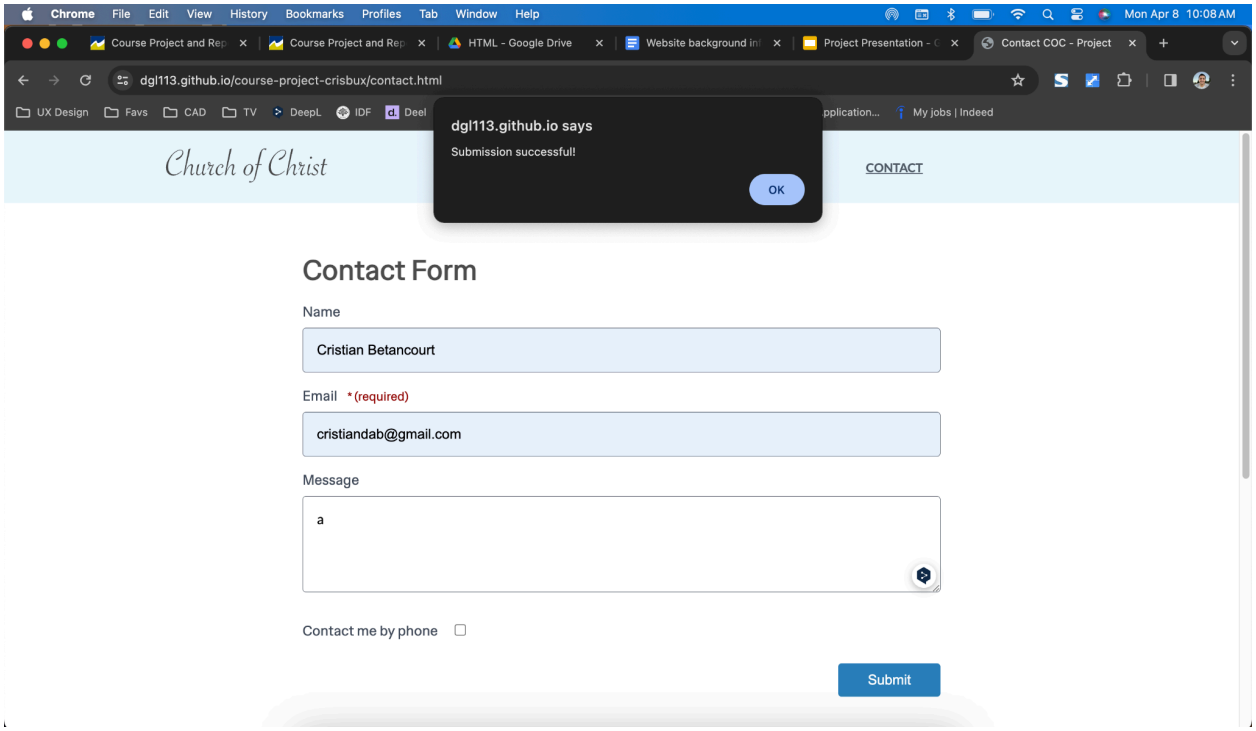
Manual Validation: Safari





Manual Validation: Chrome





Project Work Experience

Time spent: 24 - 32h

Knowledge and skills acquired:

Although throughout the course we were applying the knowledge in the different exercises, it was interesting to see how I started to think about what elements, techniques or concepts to use to achieve the tasks I proposed to work on.

At this stage the problems/objectives were set by myself so it was different to start the initiatives from the previous knowledge of the classes. Being able to mix and use different techniques to solve the problems, even if it was basic, was something that surprised me about the work I did.

In conclusion, I think that the knowledge acquired is related to being able to implement all the concepts and techniques proposed in the class, to solve the tasks of the exercise.

Skills: Analysis, understanding of the problem, problem solving, in-depth analysis of the documentation and contents of the class.

Challenges Encountered:

The validation of the form really was a big challenge, because I made several attempts with different forms but the solution did not work. In this I invested many hours without understanding what the problem was. Then I went to read the documentation and the class contents, and I found that it could work better if I created two separate .js files, indeed, I tried it and it worked. The interesting thing about this, is that I was browsing solutions of different techniques looking for the clearest and simplest one.

Lessons:

The main one is that it is not bad to look in the documentation to find the answers, it is impossible, in my case, to master the language by heart, but if I have doubts about how to solve problems I can go to the documentation and class notes to find the answer. It is also valid.

Also, in this case ChatGPT helped me solve a problem that I do not know if I had to do with js. It was not something we had seen in class or that I could know previously, the problem with error 405 after submitting the form. There I found the solution and it worked.