

# Progress Report II

COSC 4P02

Software Engineering II

Naser Ezzati-Jivan

Winter 2023

April 2nd, 2023

Brock University

---

**Alec Ames**

**Team Leader**

**Product Owner**

*aa19vl@brocku.ca*

*6843577*

**Ibrahim Hashmi**

**Developer**

*ih17px@brocku.ca*

*6352926*

**Tommy Pham**

**Developer**

*tp19if@brocku.ca*

*6733646*

**Justin Stickel**

**Scrum Master**

*js19ge@brocku.ca*

*6718404*

**Matthew Benson**

**Developer**

*mb19hz@brocku.ca*

*6729388*

**Francis Monwe**

**Developer**

*fm19pe@brocku.ca*

*6724355*

**Abhijeet Prajapati**

**Developer**

*ap15qm@brocku.ca*

*5987722*

**Haaris Yahya**

**Developer**

*hy20ao@brocku.ca*

*7054984*

# Table of Contents

<b>Table of Contents</b>	<b>2</b>
<b>1. Overview</b>	<b>2</b>
1.1 Contributions	3
<b>2. Design &amp; Implementation</b>	<b>4</b>
2.1 Staff Authentication	5
2.2 Event Editor	8
2.3 Cursor Component	11
2.4 Search Bar	12
2.5 PWA Support	14
<b>3. Sprints</b>	<b>14</b>
3.1 Meetings	14
3.2 Past Sprints (Excluding those covered in Report 1)	15
3.2.1 Sprint 3	15
3.2.2 Sprint 4	15
3.3 Current Sprint (Sprint 5)	16
<b>4. Issues</b>	<b>16</b>
<b>5. Links</b>	<b>17</b>

---

## 1. Overview

Since our last report, our software engineering team has continued to make progress on our interactive timeline product for the Niagaa-on-the-Lake Museum. We have finalized incomplete items from earlier sprints as well as integrated new features to build upon the previous version of the product. In this progress report, we will provide further insight into our team's development progress as well as our continued adherence to the agile

methodology. As we are entering the final phases of development, we will cover our recently completed sprints as well as our current in-progress items that are still being worked on. Additionally, we will identify problems and challenges that have been uncovered and how we have been addressing them.

## 1.1 Contributions

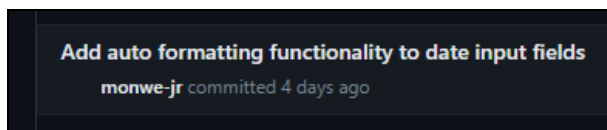
In addition to the contributions each member outlined in the previous report, the following contributions have been completed:

<i>Member</i>	<i>Contributions/Achievements</i>
<b>Alec Ames</b>	Created cursor element to visualize current position on timeline. Added scrolling/zooming to timeline bar. Styled event editor, searchbar. Added upload file function. Added PWA support.
<b>Matthew Benson</b>	Fixed issue with themes not persisting on refresh, added transitions on timeline for event changes.
<b>Ibrahim Hashmi</b>	Finalized staff login page and authentication with Supabase.
<b>Francis Monwe</b>	Created and developed the search bar. Implemented a drop-down menu for search bar results and implemented functionality to change page components after selection.
<b>Tommy Pham</b>	Added functionality to support years that contain multiple events.
<b>Abhijeet Prajapati</b>	Created edit buttons for the staff-edit view on the Timeline page.
<b>Justin Stickel</b>	Continued chairing and facilitating team meetings; Delegated and organized tasks among members.
<b>Haaris Yahya</b>	Worked on adding image full screen and zoom-in/out

	functionality and implementing bug fixes related to and in addition to zooming in/out.
--	--

**Note:**

The GIT configuration for Francis' was under a different email, resulting in all commits being authored under his username 'monwe-jr' but not linked with GitHub. However, fixing this issue did not retroactively link the prior commits to his account. For reference, the following screen capture is what the unlinked contributions look like.



## 2. Design & Implementation

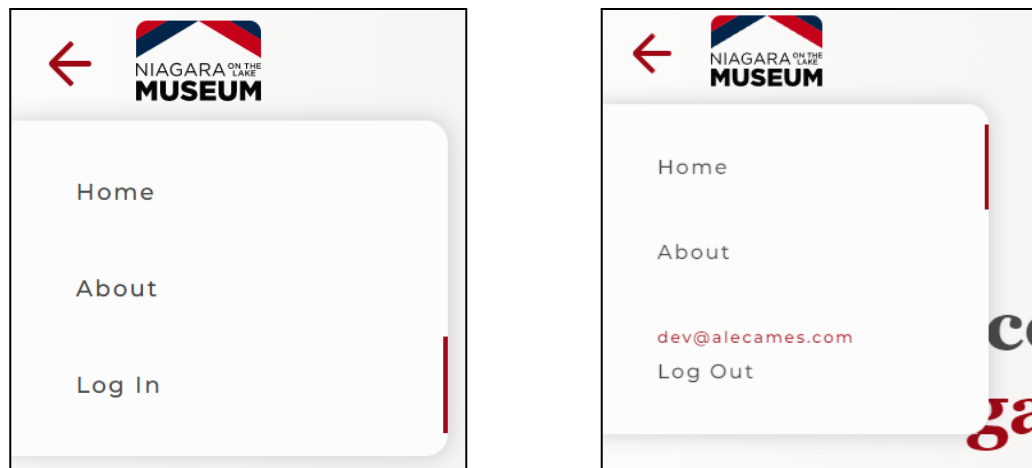
In this section of the report, we further discuss our design choices and our implementation methods of core features added and Our team has largely kept our designs and themes showcased in our last progress report, with several new features: a timeline cursor, staff authentication page, and editing and adding menus. These components work in conjunction with the pre-established timeline and database to provide the Niagara-on-the-Lake staff with an easy and professional method of adding/removing/changing displayed data.

## 2.1 Staff Authentication

### Design

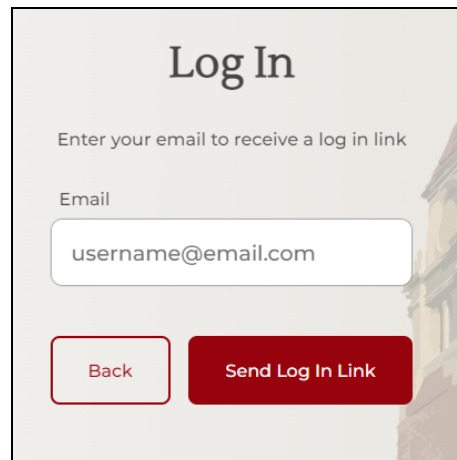
Based on the client requirements discussed in our prior progress report, we have implemented a user authentication feature. Our sign-in process is streamlined and user-friendly, with the login page consisting of a submit button and a single text field for entering authenticated emails. While logged in, the authenticated user's email is on display in the navbar on the left.

**Left: appearance of navigation menu when user is not signed in; Right: appearance of navigation menu when user is authenticated.**



Our focus on user-friendliness meets the client's requirements as it ensures that user authentication is straightforward and easy to use, providing staff members with an efficient and convenient login process.

## Login window

A UI mockup of a login window. It features a light gray background with a faint image of a classical building on the right side. At the top, the text "Log In" is centered in a large, dark serif font. Below it, a smaller line of text reads "Enter your email to receive a log in link". Underneath this is a label "Email" followed by a white text input field with a thin gray border. The input field contains the placeholder text "username@email.com". At the bottom, there are two buttons: a light gray button with a red border labeled "Back", and a solid dark red button labeled "Send Log In Link".

Log In

Enter your email to receive a log in link

Email

username@email.com

Back Send Log In Link

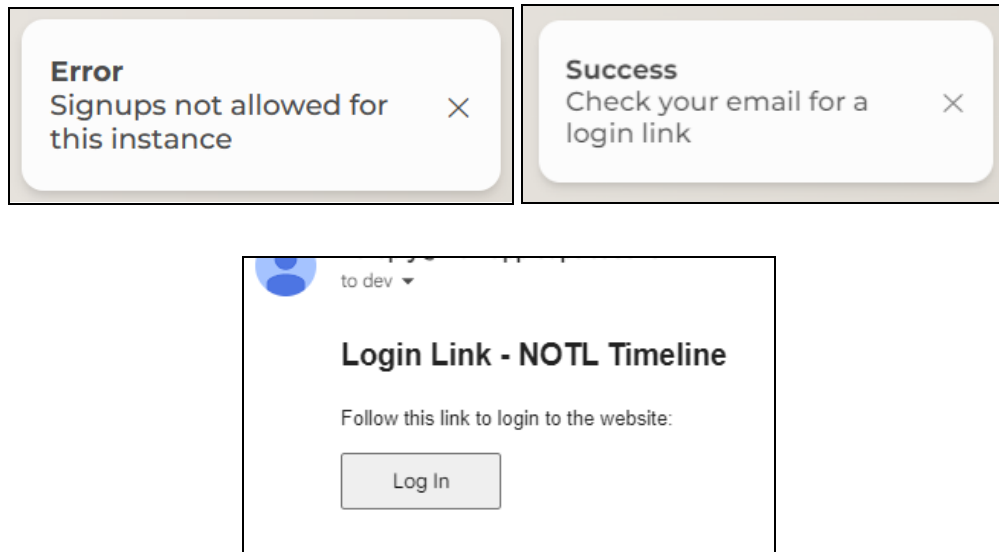
## Implementation

Upon entering their registered email into the given textfield, museum staff are granted access to the timeline via a secure magic-link sent directly to their email. This Magic Link authentication method is provided by Supabase, our backend solution. It allows us to provide a secure and passwordless entry into the account for verified users only. Login attempts through unregistered emails will simply prompt an alert with an error message stating that magic-link sign ups are disabled.

**Top-left: toast notification when an unregistered user attempts to sign in;**

**Top-right: toast notification when a registered user attempts to sign in;**

**Bottom: resulting email sent to the user's inbox.**



## 2.2 Event Editor

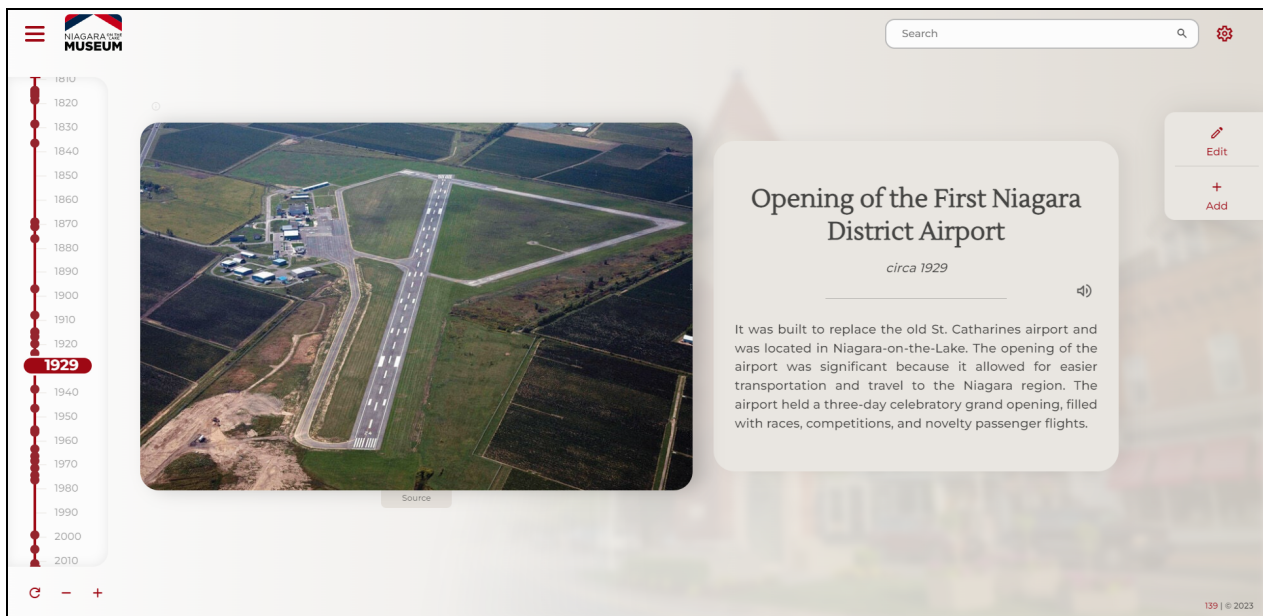
### Design

Authenticated users are presented with additional menu options on the top right of the screen that allows them to update, add, and remove timeline items. On each successful action – whether it's saving, adding, or deleting items – the user is shown a toast notification with the same utility for the authentication process. Deleting an item also prompts the user with confirmation dialog.

When editing or adding an event, the layout changes to show input boxes and text areas for each property of the timeline entry. It provides fields

for the URL of the image, with an option to drag-and-drop or pick a file to upload an image to the Supabase storage bucket. The editing page also lets the user know if they are editing or adding to the timeline with a notice above the image portion of the editor.

**View of the entire webpage with the additional editor menu to the right.**





**Left: editor menu shown to authenticated users; Right: window for editing timeline event (image property).**



**Window for adding a new timeline event (text properties).**

## Implementation

In edit mode, the event editor populates the input fields with the data for the selected entry and allows the user to edit. The 'title' and 'date' fields are mandatory, and the application will show an error toast notifying the user. The 'date' field will auto format the entered digits into the ISO standard date that entries are saved as in our database. The image frame allows drag-and-drop, pasting an image address, and picking a file from disk, along with a live preview of the selected image. Changes are not saved to the server until the user clicks 'save' or optionally 'delete'.

In add mode, the actions the user can perform are much the same as discussed in the event editor. The exceptions are that the text fields are populated with placeholders to guide the user input in addition to the menu options allowing for only saving and canceling changes.

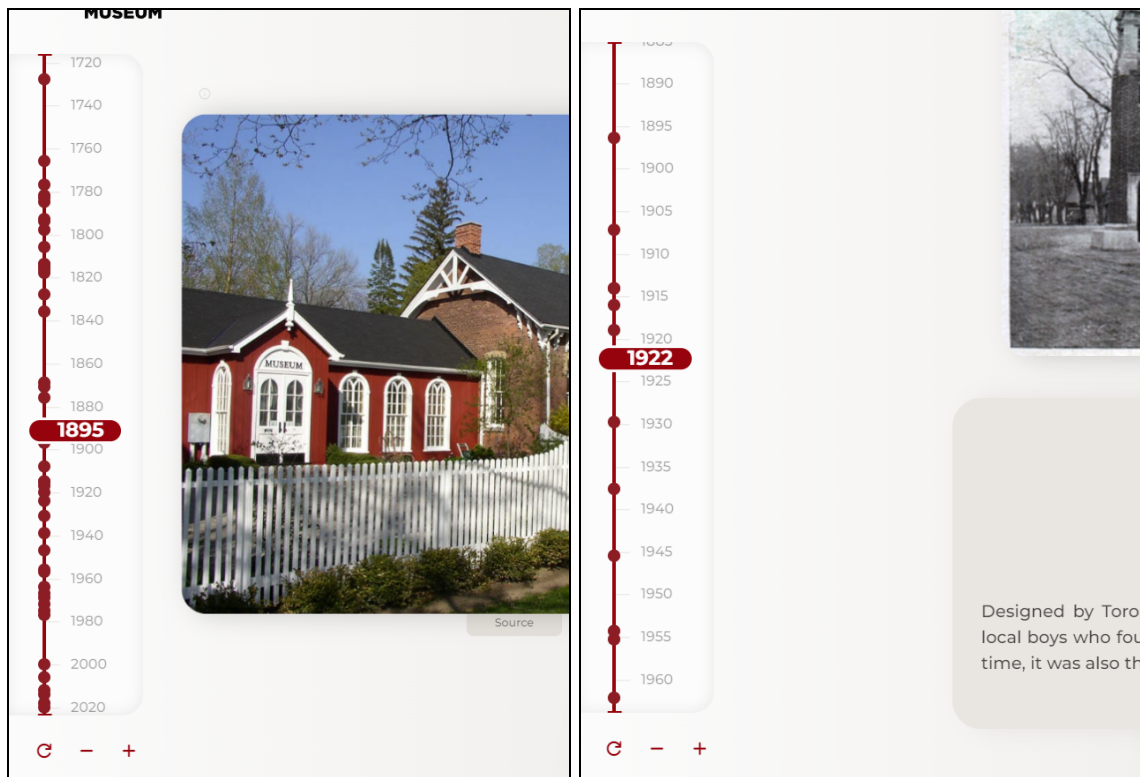
For increased usability, the selected form item border increases, showing the user which item they have selected. Additionally, the form styling is tested to work in each of the three themes as discussed in our first report.

## 2.3 Cursor Component

The cursor component was created to visualize the current selected timeline item. Clicking on an item will trigger the cursor to animate towards the desired item, while easing both in motion and the date displayed on the cursor. When adding the cursor, the timeline bar was also overhauled to allow

for zooming in, zooming back out, and resetting the scale. While the timeline bar is zoomed in, the user can use their scroll wheel (desktop) or drag to change the current position. This provides a way to spread out the timeline items without losing the positions of the dots relative to the timescale markers to the right.

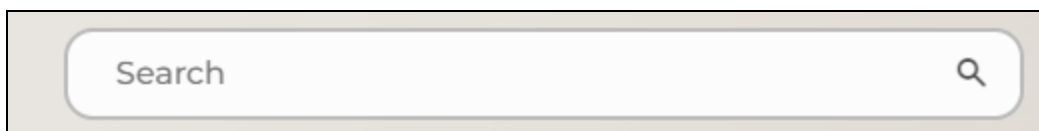
**Left: fully zoomed out view of the timeline; Right: zoomed in view of the timeline, offset to position year 1922 near the middle.**



## 2.4 Search Bar

The search bar component works by saving the list of dates and titles as an object and using the `filter()` function to filter the list with the desired search keywords. It allows for searching partial words, full sentences, and years. Clicking on an item in the search bar will navigate to that item in the list.

**Top: search bar with no search query entered; Bottom: list of results in drop down portion of search bar.**



A search bar with a light gray background and a rounded rectangle. Inside, there is a white input field with the placeholder text "Search" in a light gray font. To the right of the input field is a magnifying glass icon.



A search bar with a light gray background and a rounded rectangle. The input field contains the text "Niagara". Below the input field is a dropdown menu with a white background and a red border. The dropdown menu has a close button (X) in the top right corner. It contains a list of search results, each with a year and a title.

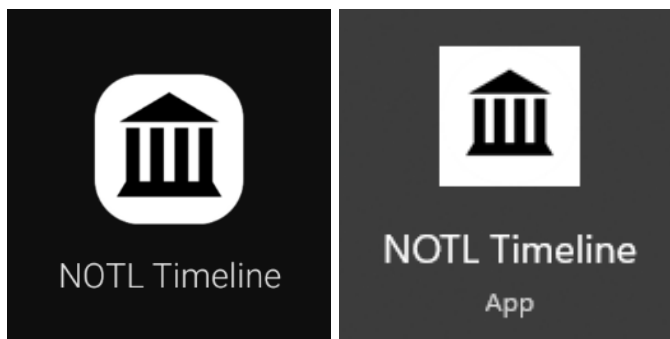
Year	Title
1726	Fort Niagara Built
1764	Treaty of Niagara
1780	Peter Secord, Sr. Became First European to Set...
1781	Niagara Purchase (Treaty 381)
1791	Niagara Became District Capital for the Niagar...
1792	Niagara-on-the-Lake Chosen as First Capital of...
1816	Niagara Volunteer Fire Department Built First ...
1826	Niagara Volunteer Fire Department Established
1867	Former President of the Confederate States of ...
1895	Niagara Historical Society Formed

## 2.5 PWA Support

An additional feature implemented was progressive web application (PWA) support, by registering a service-worker and adding the required metadata and cache policies to the application. A simple addition, but with PWA support, it can be installed as an application on all platforms for ease of use, and an improved, more immersive experience.

**Left: the application installed as an application on Android;**

**Right: the application installed as an application on Windows 11.**



## 3. Sprints

### 3.1 Meetings

Weekly scheduled meetings have continued to take place without issue. All members provide their personal updates in weekly stand-ups, whilst also participating in bi-weekly sprint planning and sprint retrospective meetings.

These meetings are an integral part for fostering group understanding and organizational coherence among members. When we complete our final sprint, we will continue to have stand-up meetings to monitor the testing and overall stability of the product deployment, however we will naturally cease sprint planning and sprint retrospective meetings as they will become obsolete.

### 3.2 Past Sprints (Excluding those covered in Report 1)

The following subsections feature our two most recently completed sprints, and their notable items, which primarily focused on the addition of new features to the interactive timeline application.

#### 3.2.1 Sprint 3

- Connect data from Supabase database with timeline application
- Create text-to-speech controls to read timeline text
- Timeline item search functionality
- Fix responsiveness of the timeline screen across different screen sizes
- Fix image overlap with logo when scrolling up on the timeline screen

#### 3.2.2 Sprint 4

- Create edit buttons on timeline page
- Finalize login page and create authentication flag to use in C4SWE-24
- Make themes persist on reload
- Set up secure user authentication with Supabase
- Finalize transition on timeline event change

- Fill 'About' page on website with relevant information about the town, the museum and our project

### **3.3 Current Sprint (Sprint 5)**

As of this report we are midway through our final planned sprint for this project. As such, this sprint includes all remaining items from the product backlog, however bugs will continue to be added and addressed in our formal testing and deployment phase.

- Creating zoom in/out functionality on TimelineBar component
- Implement editing of timeline event data through Supabase
- Implement functionality for event edit buttons
- Create timeline cursor component
- Add functionality to handle years with several events

## **4. Issues**

Since we're heading into April and team members have other university obligations, such as projects, it has become more difficult to hit sprint goals. Simply due to overlapping time constraints, some items had to be pushed to future sprints as a result. This being the case, for our final sprint we have planned less items overall to ensure we will be ready to move into the testing phase of our product without issue.

In our testing phase, we aim to test multiple devices, browsers, and screen sizes using Playwright. We will record common interactions and test them to ensure the experience is equal on most platforms.

## 5. Links

GitHub Repository: <https://github.com/SWE-2023/COSC-4P02-Project>

Vercel Deployment: <https://museumtimeline.vercel.app/timeline>