GitHub repo: https://github.com/ryan-montville/days-for-girls-solon
Demo: https://ryan-montville.github.io/days-for-girls-solon/

Table of Contents

List of Project Participants

Name: Ryan Montville
contribution to the project: The entire project

Abstract of the Project

When brainstorming concepts for my project, I asked my parents if they had any suggestions. My mom asked if I could build an app for her Days for Girls team in Solon Ohio. Days for Girls International provides sustainable supplies, shatters stigma, elevates menstrual health, and advocates for global policy change to create lasting impact. The Days for Girls team my mom is on sews reusable items to be distributed to communities around the globe. My mom helps keep track of the inventory her team has sewn and wanted a better way to manage it. The previous method included several Google forms and spreadsheets with basic formulas to calculate totals. The links to the forms and sheets were long strings of random characters and not easy to share. The old system was time consuming and provided less detail than desired. The app I created provides inventory management, event sign-ups, mailing list sign-ups, and general info about the Days for Girls Solon Ohio team all in one place that is easy to share with members and potential donors.

Project Narrative

The goal of the app was to combine the various Google forms and spreadsheets into a single app that would be easy to use and share with other members on the team. We identified 4 main pages and a few sub pages. The most desired page was the inventory page, where the user would be able to log entries of items created/donated and items distributed. The ability to generate a report for a given time frame was requested since the Solon team is required to report their monthly totals to the Days for Girls international team. The next page was a form to sign up for emails and newsletters from the Solon team to replace the Google form they previously used. The third page was a page to talk about what the Solon team's current fundraising goals are and to provide a link to the team’s donations page. The previous method was to share the link with a QR code, but even then they had to search for the QR code amongst their other files. The final page was a place to display upcoming events with the ability to sign up to attend the event.

Design considerations

Inspiration and Datasets

The main inspiration for the design of the app is the Days for Girls International website because we wanted to match their branding/design. I matched colors, fonts, and shapes of the page elements to stay consistent with their design. I used the limited data the Solon team had created from the Google forms to track their inventory as a starting point to model how the inventory management in the app would be structured.

Homepage Goals

The primary goal for the homepage of the app is to provide quick access to the app's core functions and establish the Solon team’s mission. Once the app goes live and the team learns how users are interacting with the app, it is possible that more user roles will be created in the future and the homepage could turn into quick links based on the user’s role.

Consistency considerations

The design of the app goes for a modern, responsive, and user-centric design with focus on accessibility and visual clarity. The app features both light and dark mode themes, high contrast between background and foreground colors, and accessibility features such as screen reader links and keyboard focus traps to improve user experience. The app does not include any graphics because none of the pages required them and we didn’t want to clutter the page with unnecessary elements. The icons throughout the app are Google’s Material Icons and were chosen to match the action the user will perform when clicking on the button or link.

The app maintains consistency across the pages by loading in the same header, footer, and modals from separate files through the deferred pattern. Every page uses a card layout to group content together to allow the user to quickly find what they are looking for on the page.

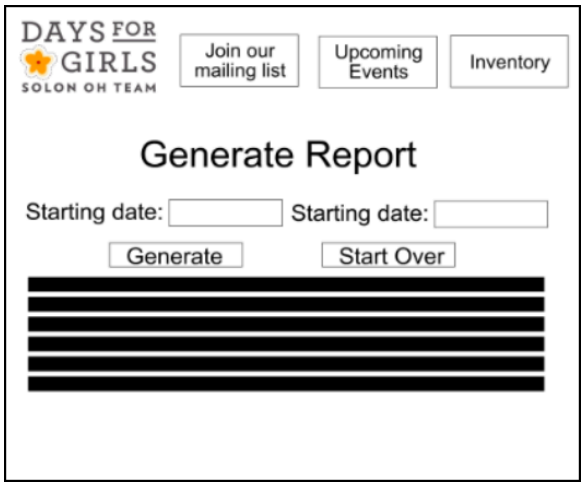
Interaction and design patterns

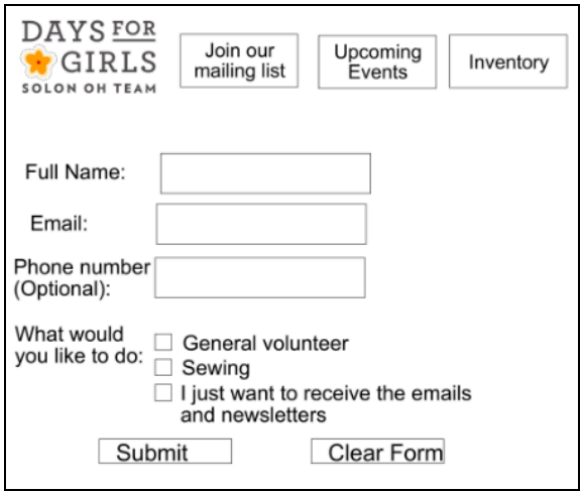
The inventory pages use a Promise chain to first authenticate the user to determine their role, then fetches the inventory data from Firebase to prevent unauthorized access. The events page is similar in that the Promise chain authenticates the user and, depending on their user role, allows them to manage events and sign-ups if they are an admin, otherwise the app only allows them to view the event details and sign up to attend the event. The donate page authenticates and checks the user’s role, then allows the user to edit the content on the donate page if they are an admin, otherwise, only allows the user to view the page.Every page except the home page interacts with Firebase through the deferred pattern and then updates the DOM.

The app uses a few design patterns. The app interacts with Firebase through the Facade design pattern using the firebaseService I created to simplify the Firebase SDK into exported functions that could be adapted to a different data storage method in the future if the Firebase free tier no longer meets the needs of the Solon team. The app uses the Adapter pattern to take the firebase documents and turn them into strongly-typed TypeScript objects for easier use throughout the app. The Command pattern is used when deleting items by decoupling the delete action from the components that trigger it. The Strategy pattern is used to generate the inventory report by separating the steps into separate functions such as filtering the entries within the date range, calculating the totals, and creating the various tables to be displayed. Once I implement storage location into the app, I can easily modify the generated report to display inventory totals, storage locations, or other filter options.

Design and specification

Wireframes





App Sitemap

Homepage
- |- Mailing List Sign Up page
- |- Events page
  - |- Event Sign Up page
  - |- Create New Event page
- |- Inventory page
  - |- Donated Inventory page
  - |- Distributed Inventory page
- |- Donate page

I used the existing Google forms and spreadsheets to create the layout of the wireframes. I am not much of a designer, and even less so someone who draws, so the design planning phase went from the wireframes to creating the html and css for the site to get a feel for the layout and colors rather than more sketches and mockups. Again, creating the design for the app was fairly straightforward and simple since the app follows the Days for Girls International website branding and design. Creating the sitemap for the app was based on the forms and spreadsheets the app will replace along with a few additional pages for events and info about the Solon team.

The app initially relied on local storage to handle the data manipulation and allowed me to develop the functions before proper data storage was implemented. Further on in the semester, once most of the functions to handle the C.R.U.D. operations were sorted out, I added Firebase to the app to allow for centralized data storage and user roles to control who could access the various functions in the app.

Another feature of the app that evolved over the course of the semester was the donate page. The original plan was to just include a link to the fundraising page on the Days for Girls International site, but then we had the idea to create a page to show the Solon team's current goals with fundraising and sewing items. This created the need for the Solon team to edit the content of the page without requiring any technical knowledge or coding experience. The first method I implemented was using Marked.js to add markdown code to the page, which is simpler than writing the HTML code, while providing more formatting options than just basic strings. I included a markdown cheat sheet for anyone who has never used markdown before. While this method worked, I wanted to make the experience better, which led me to discovering Quill.js, which creates a javascript text-editor that can easily be added to the page and is much easier to use than markdown. Testing and iterative design

User Testing

I conducted user testing over thanksgiving break with my family. The age range of the users who participated in user testing ranged from age 17 to 59. Everyone owned a smart phone and was very familiar with how mobile apps work, so there was not anyone who was "tech illiterate". The app was tested on several devices which included Pixel 9 Pro, Samsung S21, iPone 12,

iPhone 16, iPhone 16 Pro Max, iPad, Windows PC. The app was tested in several browsers which included Chrome, Firefox, and Safari on both desktop and mobile. The app loaded fine and there were no issues with the design layout on any device or browser. None of the users had any recommendations on changes needed to the design of the app. All users were able to complete all their tasks.

User 1 signed in as admin - on iPad
Task: Count the number of people attending the “Sewing Day” event on December 3rd **Outcome**: user was able to find the event and view the number attending without issue or guidance
Task: Create an event **Outcome**: easy to create the event. Commented that the event time field was a plain text box, while the event date used the IOS date selector. Maybe expected the time to also use the IOS time selector, but understands the time input is just plain text since the event time could be set as “3pm” or “3pm-5pm”. **Issues**: One bug observed, the events page displayed the date of the event as the 16th, when the set the date to the 17th. (This is an issue with how javascript handles dates and was easily fixed)
Task: Edit the event the user just created to change the time of the event and add a note in the description about the time change **Outcome:** User said it was easy to edit the data since the input fields already populated the event info for them. The user did comment that there are no formatting options such as making the text bold. We decided that was fine since the events page might become “too busy” if the text had different formats throughout the page
Task: Delete a sign up entry an event **Outcome:** The user was able to figure out how to delete the entry without any help/guidance.
Task: Delete an event from last month **Outcome:** Easily found the delete button and deleted the event. No issues. The user liked the confirmation window asking them if they wanted to delete the event.
Task: Generate a report for July 2025 **Outcome:** Easy to generate report, easy to understand report, no suggested changes to what info is included in the generated report. Slight confusion that they had to scroll down to the bottom of the page to view the report. As a result, I added some javascript code to scroll the report into view
Task: Asked the user to add socks as a new component to the inventory **Outcome:** User easily understood how to add the new component, no issues.
Task: Asked the user to add a new entry log to add some socks to the inventory **Outcome:** User was able to accomplish the task **Issues**: The user observed that the inventory log was not in descending order, but the new log entry was added to the top, out of order. (This was an issue with how the firestore ordered the entries and has been fixed).
Task: Delete the entry log that the user just entered **Outcome**: The user understood how to delete the entry log, but was not successful in deleting the log

Issues: The user received an error from the app when they tried to delete the item. (The issue was that I was not adding the entryId to the row before adding it to the inventory entries table. This has been fixed and the user was able to delete the log entry immediately after adding it.)
User 1 signed in as admin on windows desktop
Task: Distribute some tote bags **Outcome:** The user easily added a distribution entry log to distribute tote bags
Task: Update the contents of the donate page to add an image. **Outcome:** After following the markdown cheatsheet, they were able to add an image and update the donate page.Easy enough. They commented that the “Last update” line at the bottom of the page was a nice addition.
User 1 signed in (not as admin) on windows desktop
Task: Asked the user to sign up for the event in January **Outcome:** Since the user was signed in with their Google account, the form was already populated with their name and email address, which they found convenient.
Overall feedback I asked the user for feedback about the use of modals to add/edit information in the app compared to having the input fields on the page above or below the event/entry data and they said the modals were preferred. I also asked the user for feedback on the design of the app and to compare the colors and design in both light mode and dark mode. The user had no suggested changes or improvements, said it all looked great.

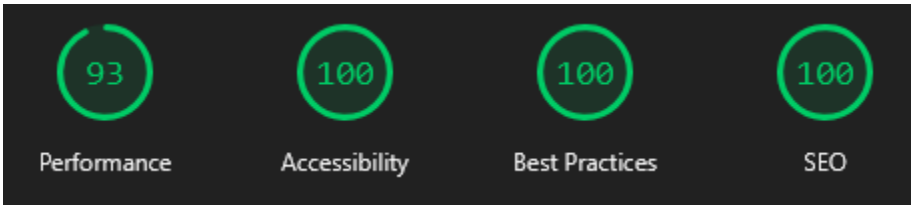
User 2 signed in as admin - on iPad
Task: Generate an inventory report for June 2025 **Outcome:** The user had some confusion using the IOS date selector (not with my code), but we got through it and had no other other issues. Easy to do
Task: Asked user to add shirts as a new component to the inventory **Outcome**: No issues. Mentioned the app was showing 0 shirts and asked if it was supposed to be like that (It is correct since they haven’t logged any shirts into the inventory yet).
Tasks: Asked user to make a new entry log to add some shirts to the inventory **Outcome:** Easy to do, no suggestions for changes
Task: Delete the shirt component from the inventory **Outcome:** The user found it easy to delete the component
Task: Intentionally told told the user to add a distribution log with more carry pouches than were currently in the inventory **Outcome:** The user received an error telling them to insert a quantity less than or equal to 77, which was the current quantity in the inventory. After they adjusted the quantity they were able to submit the entry log without any other issues.

User 3 not signed in on iPhone 16 Pro Max
Task: Sign up for an event in December **Outcome**: Since the user was not signed in, they had to fill out the fields themself. They received an error since they only entered their first name instead of their full name. After entering their last name, they were able to sign up for the event with any other issues

User 4 not signed in on Samsung S21
Task: Sign up for the mailing list **Outcome**: No issues, easy to find the page with the form and filled it out

User 5 not signed in on iPhone 16
Task: Sign up for an event in December **Outcome**: The user was able to navigate to the sign up form and signed up for the event with no issues

Lighthouse Testing



I also ran lighthouse testing with Unlighthouse and WebAIM to check color contrast. The colors I originally used were taken from the Days For Girls International site. After running the test, I found out that the colors didn't pass accessibility and contrast tests with certain font sizes. I used WebAIM to adjust the colors until they passed all the tests. Another failure was the lack of spacing in the footer on smaller screens. I adjusted the margin between links to make it more accessible. The lighthouse test mentioned that I should include meta tags for better SEO, so I updated every page to have unique titles and descriptions. I kept on tweaking the app until It got 100% in almost every category. The main reason performance is not at 100% is that the test noted that there were render blocking requests, mainly with my imports such as Google fonts. It recommended deferring them, but that only caused other issues. Overall the app still loads quickly and as the image above shows, 93% is passing.

While I did not have time to conduct formal unit testing on the app, I relied on functional testing and integration testing to validate the core functions and ensure that the logic performed correctly within the app flow.

Restrictions, limitations, and constraints

I wanted to design the app in a way that would allow the Solon team to be self reliant once I hand over the finished app. I wanted to make sure that technical experience would not be required to interact with the website in any way, both as an admin updating the site or any members viewing the site.Since Days for girls is a nonprofit, the Solon team does not have a budget to spend on services for the website, so I made sure all parts of the website could function without requiring any subscriptions, licenses, or paid accounts.

There are a few features I was unable to complete before the end of the semester. The first feature was the ability to state where the inventory items are being stored. While this would be possible to add to the app, this feature was not discussed at the beginning of the project, but a few weeks before the end of the semester and it was not a high enough priority given the limited time left to complete the app. I do intend on adding this feature before the actual app goes live. Another feature that is missing from the app is the ability to add photos using the Quill.js text editor. I added Quill.js fairly late in the semester and did not have enough time to learn all the features of the text editor. I would like to add css classes and possibly a link to view the full size image, but since Quill.js is creating the HTML code, I do not yet know how to add the class or link when inserting an image. I plan on learning more about Quill.js and how to manipulate the code it generates.

Updates since final presentation

I have taken the feedback I received from my presentation and have made a few updates to the app to improve the user experience.

1. The first piece of feedback was that with any form inputs, users might get lazy and not properly capitalize titles, names, or other proper nouns. As part of the form validation, I have added the function capitalizeFirstLetter() that capitalizes the first letter of every word in the string provided. This function is called before the data is stored in Firebase, removing the need to call the function every time the item is displayed.
2. I was asked if my app had a button to toggle dark mode. I have added a toggle button to the nav and added a script to the head of every page to make sure the page loads the theme before the DOM loads. I am storing the user's preference in session storage because then next time they visit the site, if their device dark mode preference has changed, the app applies that preference first, then the user can toggle if they choose.
3. Another comment I received was that I shouldn't reuse the same colors for different elements. The app had 3 colors, a shade of amber as the primary accent color, a shade of teal as the secondary accent color: used for buttons and in the navigation menu, and a shade of purple: used for the footer and for the table headers. The concern was that users might assume items that are the same color are the same thing, which is not the case in my app. I have since added more colors to the app to avoid any confusion.

Conclusion

Overall, I enjoyed building this app and learned a lot throughout the semester, both from the lectures and through my own exploration into the various technologies used in the app. Working with Firebase was very interesting since my only experience with databases before this class was SQL. I was impressed with how easy it was to integrate user authentication through Firebase. None of my previous projects have used authentication, mainly because I was concerned about how complex of a task it would be adding user authentication to my projects.

I am proud of what I was able to accomplish this semester working as the app's solo developer and am hopeful that this app will become an important tool in helping the Solon team achieve their mission to provide help to communities in need.