## Fairmont Community Website

Members: Andrew Sunde, Veronica Breguta, Axel Beltran

## Project Proposal/Description

The Fairmont Community website offers members of the community a hub where an individual can view the following page features and be informed on news pertaining to Fairmont. On the home page users can read a brief introduction about the Fairmount Community and can donate through PayPal, as well as visit the school district website, and subscribe to the newsletter. The social work page offers users information about Fairmont social work and what services are offered including contact details. On the history page users can familiarize themselves with history of the Fairmount Community. On the services page a user can fill out a suggest service form, as well as view services provided by other members of the community. Finally, the events page displays what upcoming events are happening in the Fairmont community.

## Features

- Users can donate and access school district website
- Users can subscribe to receive newsletters
- Users can be redirected to Facebook page of the Fairmount Community
- Users can view social worker services available.
- Users can view and learn about the history of the Fairmont community.
- Users can suggest services for the community
- Users can view a list of services provided by other members
- Users can view an event calendar detailing the events relevant to the Fairmont community.

## Software Environment

Software tools used for development:

- Microsoft Visual Studio Code
- Google's Firebase: Realtime Database
- Github: Source code management
- Google's Calendar API v3 for events
- XMind for flowchart and organization of nodes

## Framework Specification

- JavaScript
- Google Calendar API v3
- HTML/CSS

## Team Responsibility

- Andrew Sunde: Provide detail on node graphs and create the initial merged version. Collaborate on the creation of the data model. Provide feedback on project write up. Work on Events feature. Contribute and provide feedback on project presentation and final project manual.
- Veronica Breguta: Provide detail and feedback on node graphs. Provide feedback for the project write up. Collaborate on the creation of the class diagrams and data model. Work on Subscribe feature. Contribute and provide feedback on project presentation and final project manual.
- Axel Beltran: Create the initial project write up. Provide detail and feedback on the node graphs. Collaborate on the creation of the data model. Work on Services feature. Contribute and provide feedback on project presentation and final project manual.

## Test Plan

| Feature | Input Tested |
|---|---|
| **Newsletter** | **Test123@gmail.com** |
| **Service Suggestion** | **First Name: Axel**<br>**Last Name: Beltran**<br>**Service: Cleaning**<br>**Description: Excellent Cleaner!** |
| **Events** | **Title: Fairmont Festival**<br>**Date: April 28th, 2021**<br>**Description: Fairmont Festival is starting April 28th** |
| **Events** | **Title: Event Example**<br>**Date: April 30th, 2021, 9:30 A.M - 10:30 A.M**<br>**Description: (no input)** |
| **Events** | **Title: Longer Example Event**<br>**Date: May 21st 4:00 P.M - May 25th 10:00 P.M**<br>**Description: This event lasts for several days!** |
| **Events** | **Title: Two Day Event**<br>**Date: May 11th - May 12th**<br>**Description: May 11- May 12 event description** |

- For feature “Subscribe to Newsletter” we use MailerLite service. The test was done in two parts. First part was to subscribe to receive a newsletter on our website by filling out the email address text field and confirm on MailerLite that user was added to subscribers list. This part of the test was done several times to confirm that feature was working as expected. Second part was to create a test email and send it to subscribers and to confirm that the test email was received.

- For feature “Service Suggestion,” the test data for the database was generated by filling out the text fields name, last name, service, and description. The first obstacle was ensuring that the database was creating the fields and nesting appropriately since the database saved information in JSON format. Reading over the firebase documentation the “push()” method was needed to ensure no data was overwritten and instead creates a new record. The second obstacle was pulling the data from the database and formatting the data. To pull from the database, we needed to review the documentation once more and follow the guidelines of reading from the database. Once we were able to pull and log the data to the console, JavaScript was used to dynamically create HTML tags along with styling to format the database contents. Since no character restrictions are a rule, we did not focus on setting any character input restrictions. A development feature that was not addressed was the number of times a user can post a submission.
- For feature “Events,” the test data for the database was generated by inputting event titles, start-dates, end-dates and descriptions to the Fairmont public google calendar created on the Gmail, fairmontlewisc@gmail.com. This would then be pulled onto the events page and informed us of needed formatting changes through the creation of additional conditional statements. There were multiple types of events that were dictated by date-length. As such, after each test another conditional statement was created to handle the possible input. A single-day event test prompted a conditional statement to remove the end-date from being displayed when formatting these specific events. A multiple-day event that didn’t specify a specific hour start-time or end-time needed a different display from events where the hour start-finish time was specified,

prompting additional conditional statements to handle different formatting for these specified inputs. A single-day event with a specified hour start-date and/or end-date needed different formatting as well, prompting another conditional statement. Events with no description input were tested and then handled to dynamically create the description, “description to be determined” when no description was added during the creation of the event on the Google Calendar. These tests were all done through the creation of various events on the Google Calendar.

## Feature Table

| Feature | Software Used | Added Feature | Improved Upon | Existed Feature | Add Later Feature |
|---|---|---|---|---|---|
| **Newsletter** | **MailerLite** | **X** | | | |
| **Service Suggestion** | **FireBase: Realtime Database** | | **X** | **X** | |
| **Events** | **Google Calendar API Gmail Google Cloud Platform** | | **X** | **X** | |

**Potential List of Features to Consider:**

- Activity guide
    - A guide for unfamiliar community members or existing members of the community to seek points of interest in Fairmont.
- Contact Us
    - A way to directly communicate with the Fairmont community.

- Volunteer
    - Offers an opportunity for individuals to become involved with the Fairmont community.
- Gallery
    - Showcases community involvement and previous events that took place in Fairmont.
- Admin
    - Create a login page where admins could log in and add, update, or modify calendar events, services and other features that will be added later.

- For feature “Service Suggestion,” protection from mass submission needs to be guarded against. This feature currently posts any submission entered the form and is saved to the database. The flow of this feature was intended to send a notification is to some “admin” who verifies the post and then is approved to be submitted to the site. There is currently no error checking for any submission, we believe there is no restrictions on input.
- For feature “Events,” implementation for better handling of repeating events is needed. One could potentially format the repeating events to be displayed in a single element. As of now it will display a repeating event as separate upcoming events. This is purely a formatting decision, as the same information is portrayed regardless. This feature could also display events up-to an arbitrary time, but as of now it just displayed the ten most recent upcoming events.

## Requirements: Software Installation

- Microsoft Visual Studio Code – https://code.visualstudio.com/download
    - Using the link provided download the software.
    - Installed, but not required, libraries include LiveServer, Colorize, Prettier, Bracket Pair, Auto Rename.

Microsoft Visual Studio Code allows for easily readable code. This software has the ability for extensions making any type of source code manageable through extensions or through the software's native language/file recognition. The primary usefulness comes with LiveServer reflecting any changes you make to code instant with no need for page refreshes. Live Server launches a development local Server with live reload feature for static & dynamic pages. Colorize is a vscode extension to help visualize CSS colors in files. Bracket Pair is a customizable extension for colorizing matching brackets. Auto rename auto renames paired HTML/XML tags. Finally, Prettier is an opinionated code formatter. It enforces a consistent style by parsing your code and re-printing it with its own rules that take the maximum line length into account, wrapping code when necessary.

- Github – https://www.github.com
    - Create an account on Github.
    - Create repository on Github.
    - Clone repository to your device.
    - Use Github documentation to understand commands.

Github is a very useful tool for source code management. With Github we can review any changes made by team members. Any unwanted changes can be reverted to its original

code state. Commits made to code can be reviewed before they are permanent. Github tool is great for observing the complete development life cycle of a software application.

- XMind – https://www.xmind.net/download/
    - Using the link provided download the software.

XMind is a useful tool for creating diagrams. With XMind we were able to easily build out the scope of our website. XMind offers various components and styles to format the best prototypes for your next development project.

- MailerLite - https://www.mailerlite.com/
    - Below are the URL to log into MailerLite and the needed credentials:

      URL: https://www.mailerlite.com/

      Username: fairmontlewisc@gmail.com

      Password: Lewisu2020

MailerLite is an email marketing service provider with focus on simplicity, excellent customer support and beautiful email designs. MailerLite's user friendly software allows you to quickly create beautiful email campaigns, track results, manage subscribers, and individuals who may unsubscribe from the service. While MailerLite offers many features, the ones used by us are managing subscribers and sending newsletters with community updates.

- Google Service Firebase - https://firebase.google.com/
    - For testing, please connect the Fairmont Google account. In our case, a personal Google account was used for testing (note: this was a mistake). For future testing please connect the Fairmont Google account to this service.
    - Go to Console
    - Add a project
    - Create Realtime Database
    - Using the documentation provided by Google, add the necessary scripts and lines of code as described per documentation.

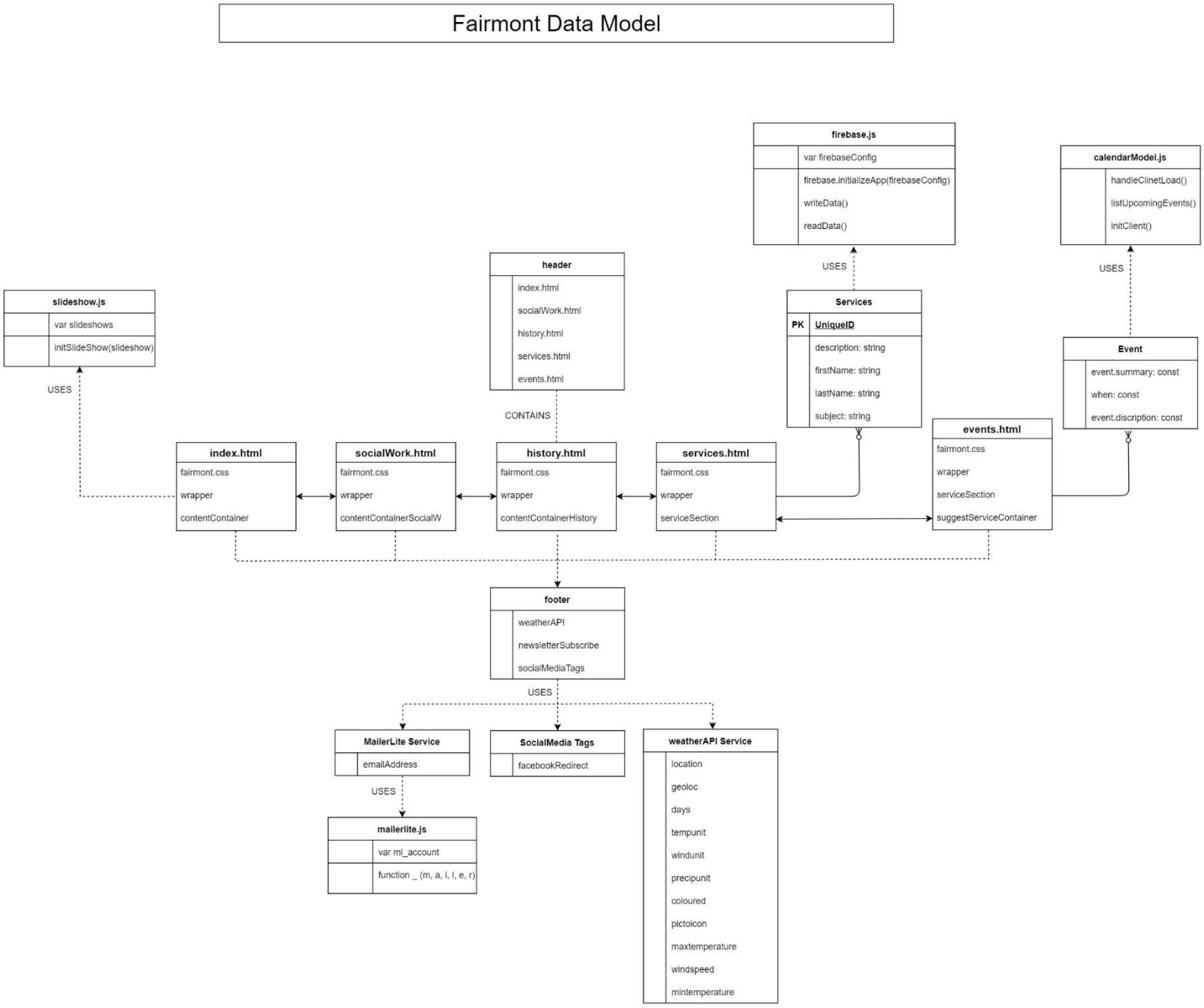
The Google Service Firebase allowed for database read and write functionality to save user input contained within the service suggestion form.

- Google Calendar API v3
    - For testing or future use, simply use the Fairmont Google account provided to us, as this was used in the creation of the Google Cloud Project “Fairmont Events” used to access the Google Calendar API. One just adds events to the Public Fairmont Google Calendar I created on this account to be automatically added to the website.
    - If clients plan on using a different Gmail for Google Calendar, they then can create a different Google Cloud Platform Project that connects to the Google Calendar API. Then replace the previous Calendar ID and API key located in the CalendarModel.js file with the ID of the Calendar they intend to use along with the new API key. Here is documentation on how to pull from the Google

Calendar API via JavaScript. https://developers.google.com/calendar/quick-start/js

- Alternatively, one can also just replace the Google Calendar ID with any public Calendar ID, and the current API key will pull events from it, regardless of who controls the Google Calendar.

## Data Model

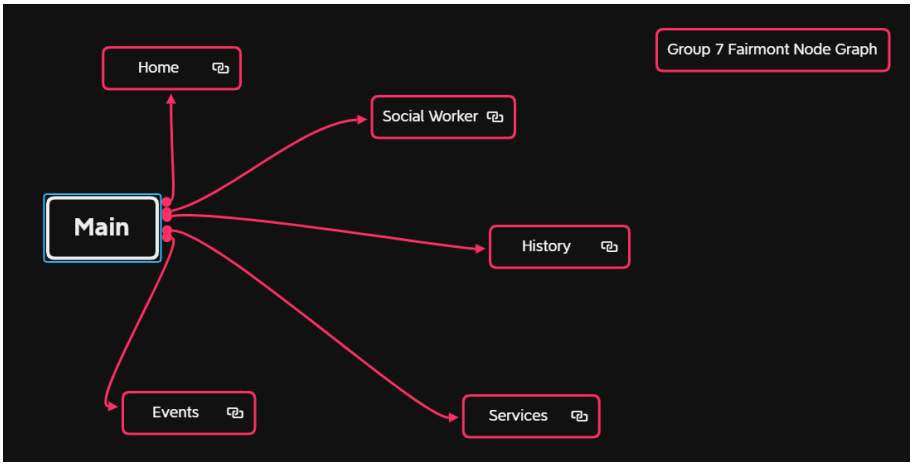

Fairmont Data Model
firebase.js
var firebaseConfig
firebase.initializeApp(firebaseConfig)
writeData()
readData()
calendarModel.js
handleClinetLoad()
listUpcomingEvents()
initClient()
USES
USES
header
index.html
socialWork.html
history.html
services.html
events.html
slideshow.js
var slideshows
initSlideShow(slideshow)
Services
PK UniqueID
description: string
firstName: string
lastName: string
subject: string
Event
event.summary: const
when: const
event.discription: const
USES
CONTAINS
index.html
fairmont.css
wrapper
contentContainer
socialWork.html
fairmont.css
wrapper
contentContainerSocialW
history.html
fairmont.css
wrapper
contentContainerHistory
services.html
fairmont.css
wrapper
serviceSection
events.html
fairmont.css
wrapper
serviceSection
suggestServiceContainer
footer
weatherAPI
newsletterSubscribe
socialMediaTags
USES
MailerLite Service
emailAddress
SocialMedia Tags
facebookRedirect
weatherAPI Service
location
geoloc
days
tempunit
windunit
precipunit
coloured
pictoicon
maxtemperature
windspeed
mintemperature
USES
mailerlite.js
var ml_account
function _ (m, a, i, l, e, r)

## Data Model Updates

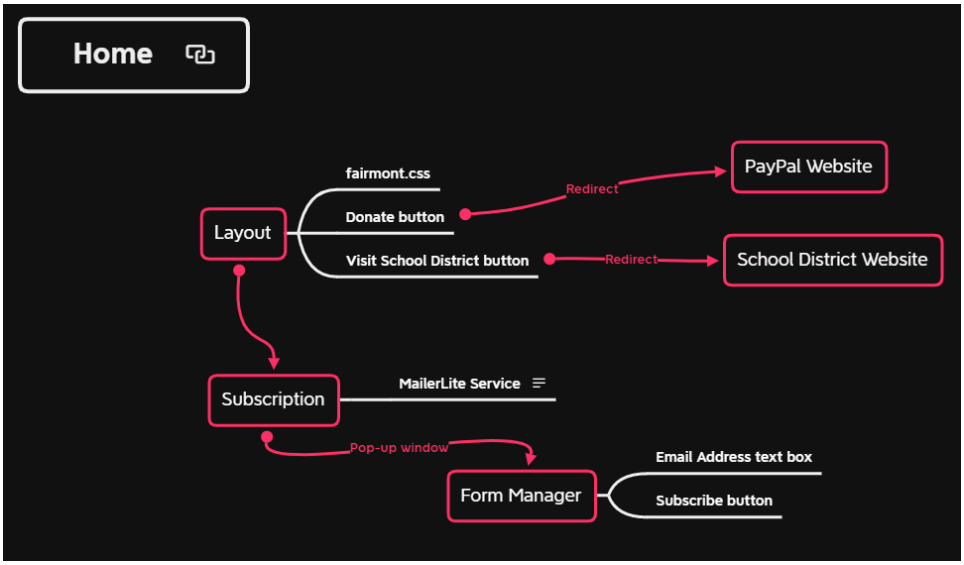
- Dropped previous data model(s) and started from scratch.
- Included a new design illustrating the flow of the website with relationships to components previously ignored.
- Included a zero to many relationship “crow” arrow for data component “services.html” to “Services.”
- Included a zero to many relationship “crow” arrow for data component “events.html” to “Event.”
- “Slideshow.js” interacts with “index.html” to create a slideshow of images.
- “mailerlite.js” interacts with “MailerLite Service” to respond to a user entered email addresses in the text box pop up submission form.
- “weatherAPI Service” pulls relevant weather data for the Fairmont location using variables defined in API documentation.
- “firebase.js” interacts with “Services” to perform networking with Firebase database to write and read data.
- “calendarModel.js” interacts with “Event” to format and retrieve relevant Google calendar information pertaining to Fairmont using the Google Calendar v3 API.

## Node Graph

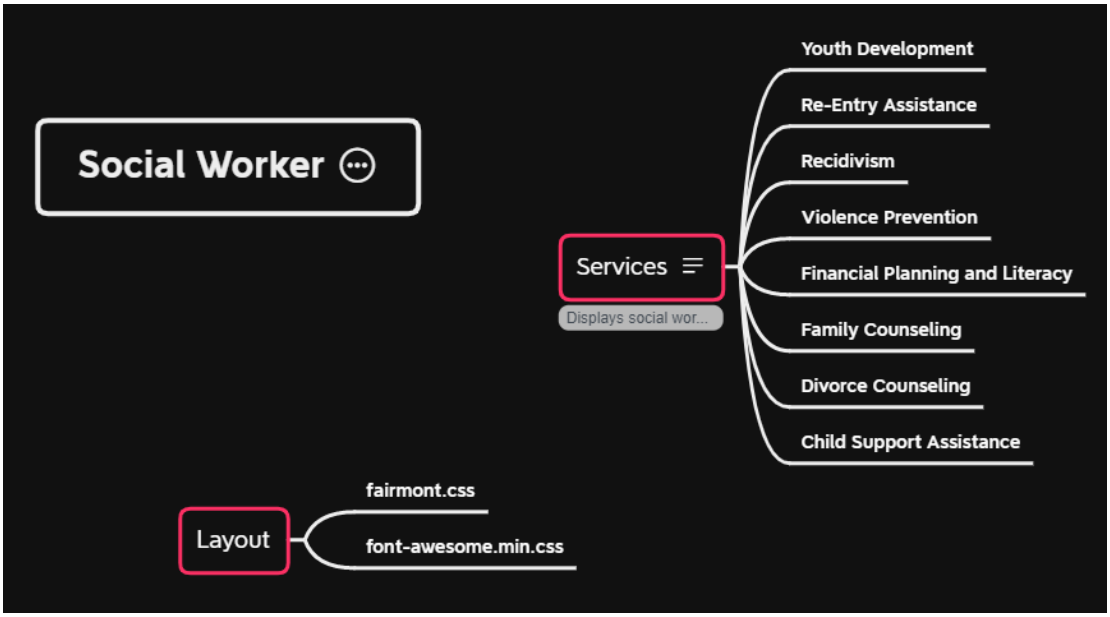
Main tab that shows the website pages. The updates done here is that we removed the details for Home page and created a new tab for that to be more organized.



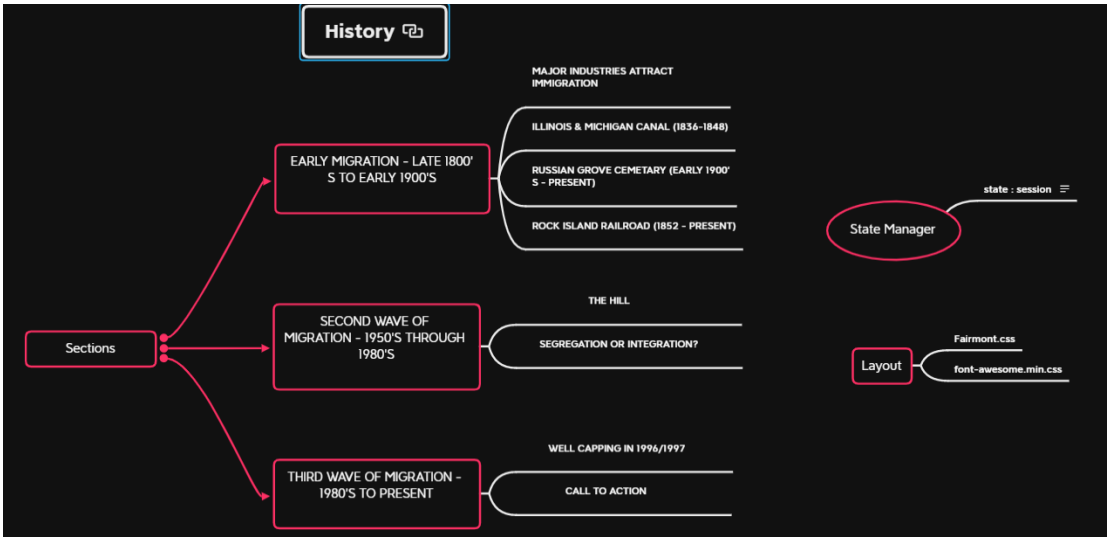

Home tab is where we can see how user interacts with the provided features. We show relationships where user is redirected to different URLs based on action. This tab was completely updated since first node graph submission.



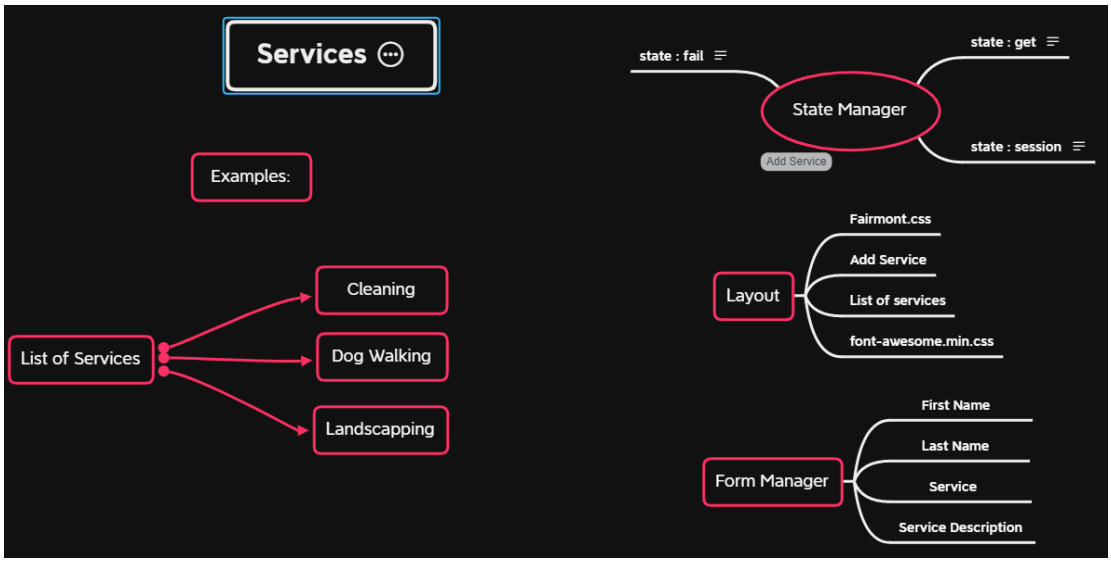
No updates were made to Social Work tab apart from adding one more item to Layout.



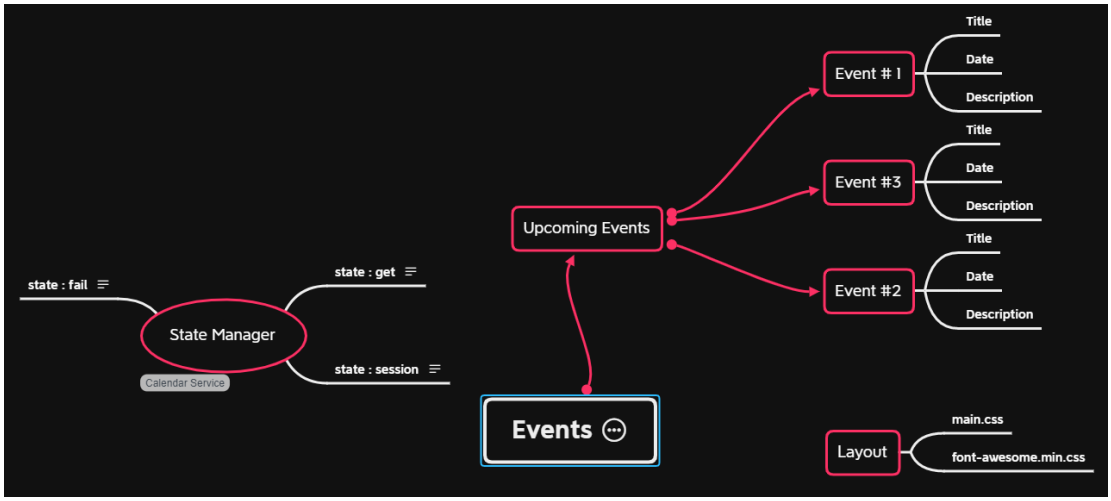

For the history tab we added more details to give a better understanding of what this page provides, such as sections to divide the content into several sub-topics.



On Services tab we added List of Services component as an example of what the user can expect to see on this page. We also added details to Layout and State Manager components.

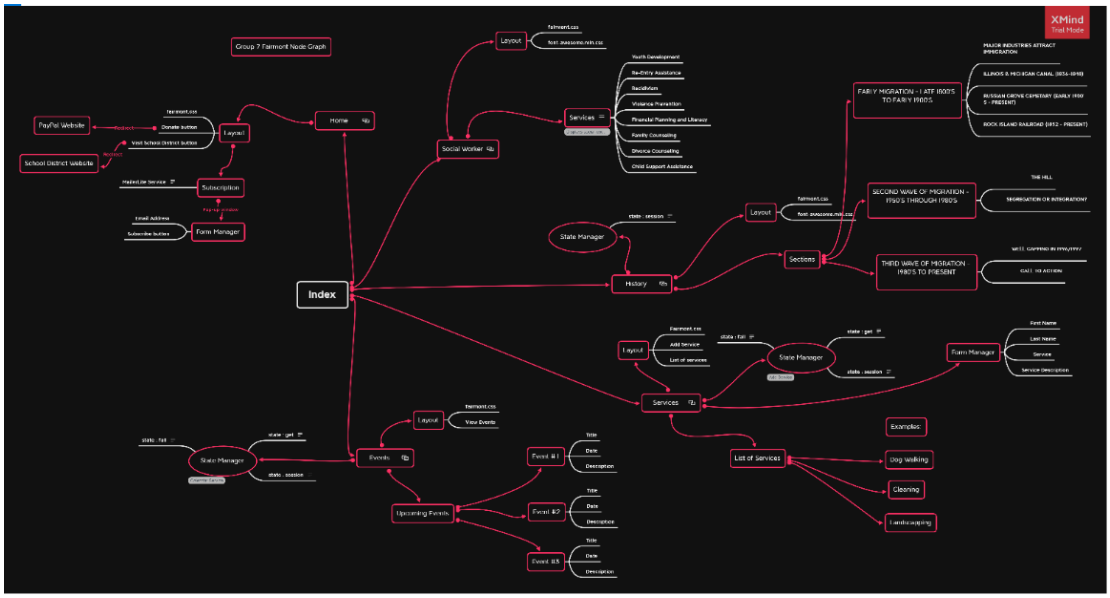



Last tab is Events page. Again, we added an Upcoming Events component to provide a clearer understanding of this feature. We also added one more item to Layout component. State sessions include retrieving from the Google Calendar API, and then dynamically creating the page on load. A failed state would be under the circumstance the API key no longer works.



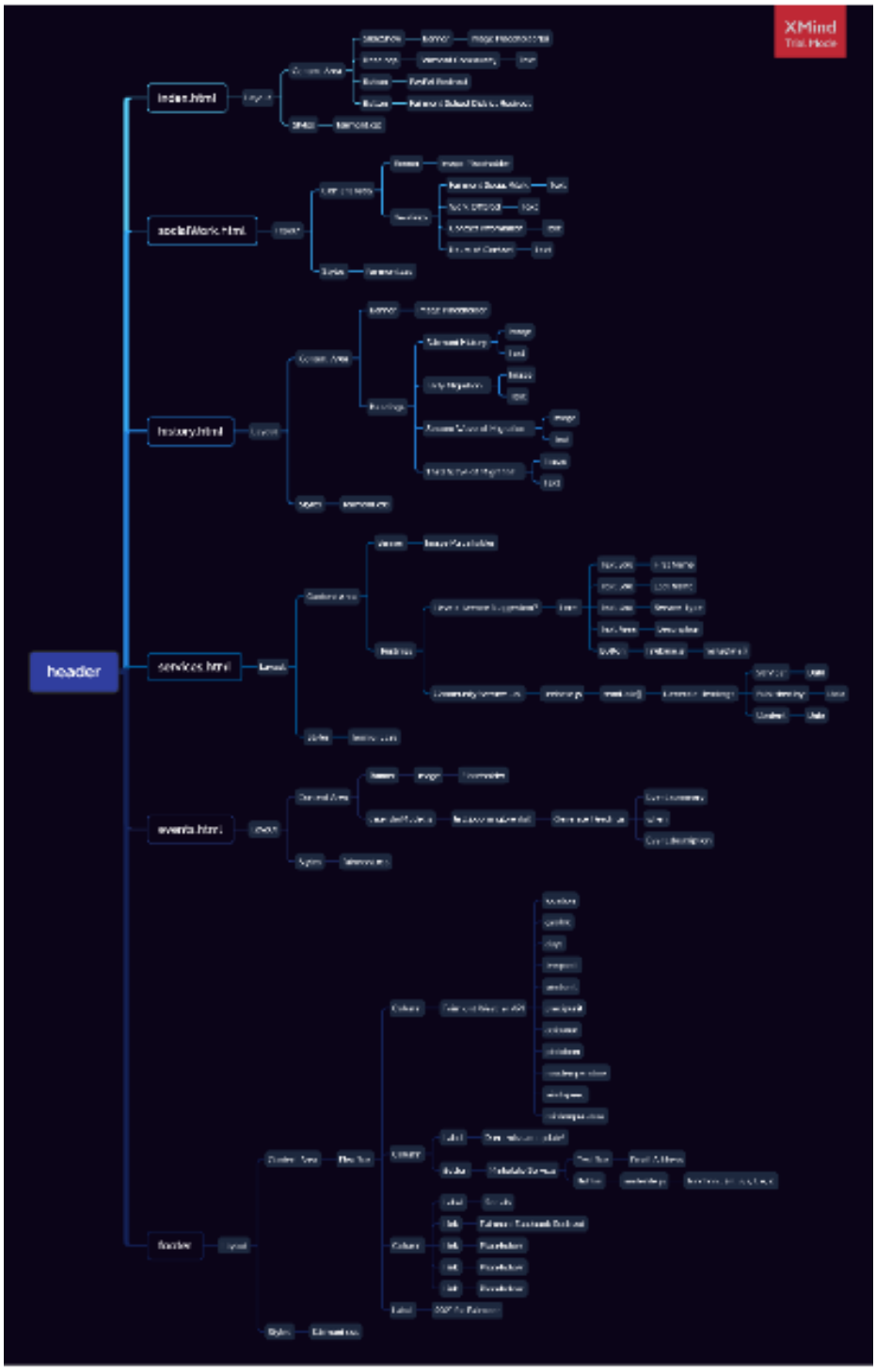
Fairmount Community tab provides a merged version of all other tabs in the workbook. It shows the relationship between all pages and components of the website.

Note: A clearer version of this can be seen in the .xmind file that will be part of the project manual submission folder.



Below is an alternative version of the merged version of node graph.

Note: A clearer version of this can be seen in the XMIND file that will be part of the project manual submission folder.



## Node Graph Updates

- Removed previously created tabs that were not relevant to the project anymore after meeting with partner
- Modified remaining tabs with more details as per previously received feedback
- Created a merged version of all pages and features of the website
- Corrected some details that were previously overlooked
- Added some details, modifications that became relevant after meeting with partners.

## Project Manual Updates

- Updated “Proposal” section refining the scope of the website and added descriptions to each section of the website.
- Added “Test Plan” section per project manual submission requirements.
- Added “Feature Table” section per project manual submission requirements.
- Updated “Requirements” sections per comments received on previous project manual submission to specify more in detail the setup process.
- Updated “Data Model” section with a new data model and added a “Data Model Updates” section.
- Updated “Node Graph” section with a node graph and added a “Node Graph Updates” section.
- Added “Conclusion” section per project manual submission requirements.

## Conclusion

As a group we believe we accomplished more than what we thought we were capable of. A lot of effort went into self-teaching web development concepts and becoming familiar with framework specific syntax. The initial website we were given was poorly formatted and sections did not seem correct. There were many unfinished sections for the website and features like the events and add a service that did not work. As a team we remade the website from scratch and communicated what we believed would fit the scope of a community website like colors, fonts, layout, accessibility, and page responsiveness.

**Potential List of Features to Consider:**
**(These features are reviewed in "Feature Table" section)**

- Activity guide
- Contact Us
- Career Opportunities
- Forms
- Volunteer
- Gallery
- Admin login