

Fermata

song. style. send.

The modern letter.

Table of Contents

| | |
|---|-----------|
| <i>Project Two Title: A Brand New Project!</i> | 2 |
| Team Members: | 2 |
| Logo | 2 |
| Project Board Website Link and Screenshots | 2 |
| Project Description (High Level): | 3 |
| Motivation | 3 |
| User Story | 4 |
| APIs to be Used: | 5 |
| Libraries to be Used: | 5 |
| Packages Required: | 5 |
| Minimum Viable Product (MVP) Requirements | 5 |
| Stretch Goals | 6 |
| Breakdown of Tasks (Ownership by Group Member): | 6 |
| Schedule for Completion of Tasks: | 6 |
| Schema | 7 |
| Associations: | 8 |
| Validations: | 8 |
| Migrations and Seed Information | 8 |
| Wireframes: | 9 |
| Website “Components” and/or “Sections” | 11 |
| Git Workflow, Website Directory Structure: | 11 |
| GIT Workflow | 11 |
| Code Review and Merge Requests | 11 |
| Directory Naming and Structure; Filenames and descriptions | 12 |
| “Stuck Time” Agreement: | 13 |
| Retrospectives – To Be Handed In At the Project Midpoint | 13 |
| GitHub Repository Link, Production Website Link Submission Dates | 14 |
| Additional Notes: | 14 |

Project Two Title: **Fermata**

| | |
|---------------------|---------------------|
| Team Number: | 2 |
| Team Name: | Fermata Team |

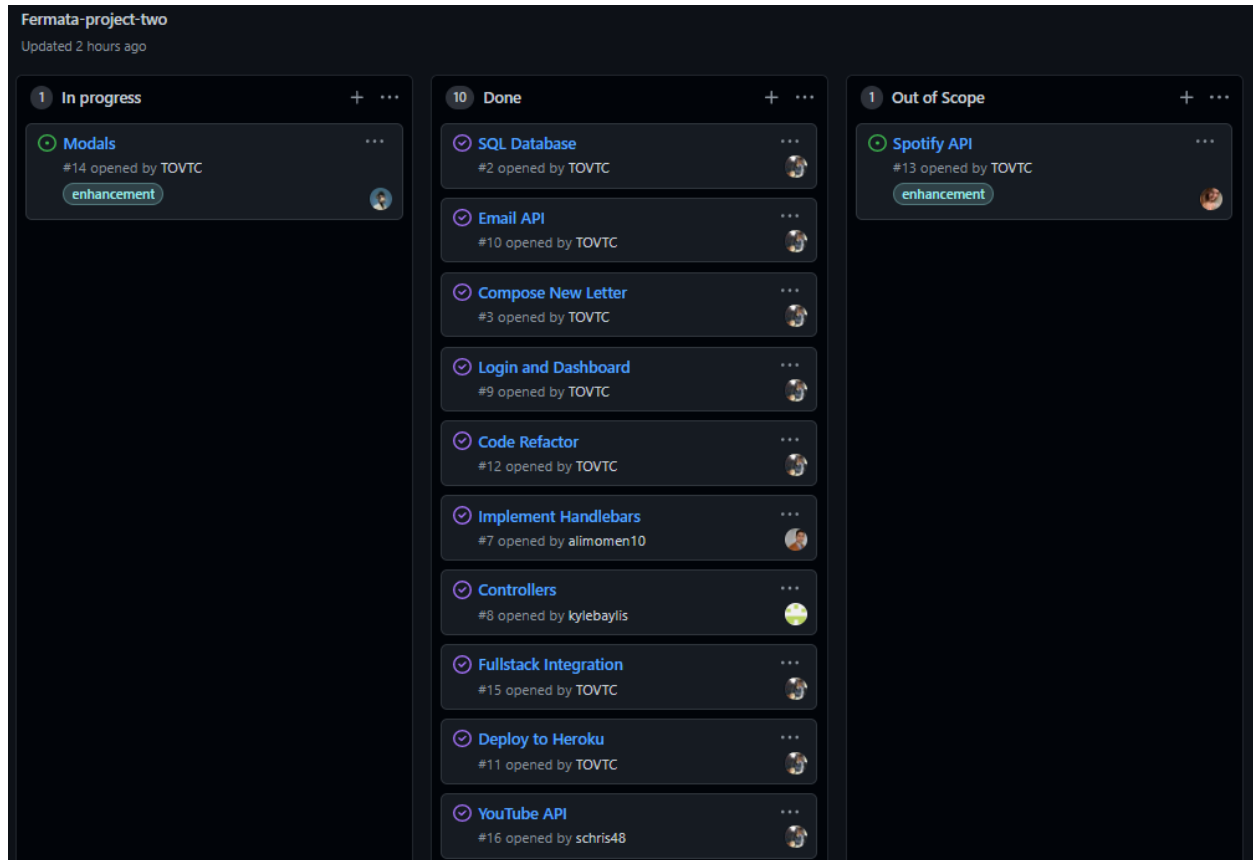
Team Members:

| <u>Name</u> | <u>Email Contact</u> |
|----------------------------|--|
| Kyle Baylis | kylejmbaylis@gmail.com |
| Sharon Christensen | sharonchristensen@live.com |
| Iwhere Ifeanyichukwu Lewis | thiszlewis@gmail.com |
| Ali Momen | ali.momen@me.com |
| Veronica Tai Chi To | veronica.tc.to@outlook.com |
| | |
| | |
| | |

Logo



Project Board Website Link and Screenshots



Our project board at the end of the project

The initial breakdown of tasks was as follows:

Ali: Creative Direction, HTML, CSS, Handlebars, Presentation

Veronica: Project Management, MySQL/Sequelize Database, Compose Letter Frontend JS

Kyle: API Routes

Sharon: 3rd Party APIs (Spotify + Email)

Lewis: Frontend Login/Logout JS, Dashboard JS

But tasks were reassigned and created as the project was developed

<https://github.com/SCScbc-Projects2022/Fermata-project-two>

Project Description (High Level):

Fermata allows you to compose a beautiful digital letter that is presented to a recipient with the background song of your choice.

Motivation

The first hand-written letter was sent in 500 BC by the Persian Queen Attosa. Since then the letter has been the source and glue to many love affairs, friendships, and bonds. Hand-writing a letter denotes care and consideration and receiving that letter breeds excitement and encourages a focus. Today, however, our modern forms of letter writing - tweets, DMs, Slack messages and even email - while convenient and instant, lack inspiration.

When you want to express love, remorse, joy, or sorrow, you want your recipient to not just read your words, but feel, see and hear them. Is there a way to make our modern letters feel special? There is, and that answer is Fermata.

Fermata is a musical term for a *pause* or a *suspension* and that's exactly what we want these letters to feel like.

User Stories

AS a USER who wishes to send someone a letter with music playing in the background

I WANT a website that connects with a streaming music service like Spotify, and creates a user profile for me

SO THAT I can compose and send a letter with a song, view my letter history and see my drafts

ACCEPTANCE TEST FOR USER STORY

GIVEN a homepage

WHEN I visit the site for the first time

THEN I am presented with options to log-in or sign-up

WHEN I click sign-up

THEN I am presented with a page with inputs for my name, email and an input field to choose a password

WHEN I click next

THEN a user profile is generated and I am sent to a dashboard with a button to compose a new letter and links to my letter history and drafts of letters I have not completed.

WHEN I revisit the site then I am prompted to enter in my email and password
 THEN I am presented with a dashboard with a button to compose a new letter ,
 links to my letter history, and drafts of letters I have not completed.
 WHEN I click compose
 THEN I am presented with a page titled YOU with inputs for my name and email
 WHEN I click next
 THEN I am presented with a page titled THEM with inputs for the intended
 recipients name and email
 WHEN I click next
 THEN I am presented with a page titled SONG with a search bar to choose a song
 from Spotify
 WHEN I select my song and click next
 THEN I am presented with a page titled STYLE with a list of fonts that can be
 selected to use in my letter
 WHEN I click next
 THEN I am presented with a page titled SEND with a text-field to compose the
 letter
 WHEN I click next
 THEN I am presented with a page titled PREVIEW with a preview of the letter
 with the song playing in the background
 WHEN I click send
 THEN the letter is sent to the recipient's email.

APIs to be Used (Projects One, Two and Three):

{list the Names, descriptions and URLs of the APIs being used for your project below, along with the reasons on why you are using it }

| Resource | URL | HTTP Verb (GET, POST) | Action | Used For |
|-------------|---|--------------------------|-----------------|-----------------------------------|
| Spotify API | https://developer.spotify.com/documentation/web-api/ | | | |
| | https://developer.spotify.com/console/get-search-item/ | GET | Search for item | Search for track to add to letter |

| | | | | |
|-------------|---|-----|---|--------------------|
| | https://developer.spotify.com/documentation/web-api/reference/#/operations/get-track | GET | Gets spotify ID for track | Add song to letter |
| YouTube API | https://kaeruct.github.io/posts/how-to-use-the-youtube-js-api-to-play-music-in-the-background.html | | Embeds a player that auto plays a video with specified video id | Spotify Failsafe |

Libraries and Packages Used (Projects Two and Three):

- Bootstrap
- Materialize CSS
- MySQL
- Node.js
- Express.js
- Handlebars.js
- Bcrypt
- Express-session
- Connect-session-sequelize
- Express-handlebars
- Dotenv
- Mysql2
- Sequelize
- Uniqid
- Insomnia
- Heroku
- Nodemon

Minimum Viable Product (MVP) Requirements

- Ability to authenticate - sign-up, sign-in
- Have a dashboard with links to create a new letter, view history, and open drafts
- Ability to compose a letter
- Ability to choose between a series of fonts
- Ability to pick a song with the Spotify API
- Ability to play the song in the background when composing and when recipient reads the letter
- Ability for recipient to send a new letter

Stretch Goals

- Include other streaming services
- Allow people to access their playlists and favourites
- Include more fonts
- Include Spotify and add working

Breakdown of Tasks (Ownership by Group Member):

Initial Breakdown:

Kyle: API routes
Sharon: Spotify and Email API
Iwhere: Front end JavaScript (Login/Dashboard)
Ali: Front end design - CSS, HTML, Handlebars, presentation
Veronica: Project Management, Database, Frontend JavaScript (Compose Letter)

Final Breakdown:

Ali: Creative Direction, Concept + Branding, UI/UX + Mobile Responsiveness, Handlebars (Formatting), Custom CSS, Pitch + Presentation, Project Documentation

Veronica: Project Management, Frontend JavaScript, MySQL/Sequelize Database + Seeds, API Routes (Rendering: Final + Login/Logout), Handlebars (Rendering), Fullstack Integration, Heroku Deployment, YT Failsafe, Documentation

Sharon: Spotify API (she got authentication and refresh tokens working!!)

Kyle: API Routes (Core), API Routes (Rendering: Initial)

Lewis: Modals

Schedule for Completion of Tasks:

| Date | Task | Notes |
|------|--|--------------------------|
| 5/24 | Brainstorming, Rough Project Outline | Logos, Mockup, MCV logic |
| 5/25 | Project Proposal | |
| 5/26 | Project Board | |
| 5/27 | Independent Coding | |
| 5/28 | Independent Coding | |
| 5/29 | Code Check In | 13:00 |
| 5/30 | Independent Coding, Asynchronous Code Check In | Reassign email API* |
| 5/31 | <u>SITE SKELETON COMPLETE</u> | <u>19:00</u> |
| 6/01 | Debugging + Middleware | |
| 6/02 | <u>Presentation Draft Ready</u> | |
| 6/02 | READY MVP | |
| 6/05 | PROJECT POLISH | |

(Schedule was unfortunately very quickly abandoned, especially with the deadline change - the basic functionality of site - except modals and Spotify was complete June 5)

Schema

| Model | Properties / Fields | Type | Details |
|-------|---------------------|---------|----------------|
| User | id | Integer | Auto Increment |
| | username | String | |
| | email | String | Is unique |
| | password | String | Bcrypt |
| Font | id | Integer | Auto Increment |

| | | | |
|--------|-----------------|---------|--|
| | style_tag | String | Inline style code snippet for Handlebars template |
| Letter | id | Integer | Generated using npm uniqid |
| | user_id | Integer | |
| | sign_off | String | Name entered to sign off in mailto |
| | recipient_name | String | Name entered to greet recipient in mailto |
| | recipient_email | Email | |
| | letter_body | String | |
| | song_id | String | Clipped from YouTube URL |
| | font_id | Integer | |
| | readonly | Boolean | Determines whether letter is a draft or sent history |

Associations (For Projects Two and Three Only):

A user has many letters

A letter belongs to one user

A letter has one font

A font belongs to many letters

Validations:

No fields can ever be empty upon submission

New User

- First name (string)
- Last name (string)
- Email (email - REGEX)
 - Non-duplicate in database

New Letter (on send)

- Recipient name (string)

- Recipient email (email REGEX)
- Song choice (string - YT song ID)
- Font choice (integer - name + value element)
- Letter body (max characters 255)

Access (read letter)

- Letter ID (uniqid for requested letter)
- Song ID (song id entered must match that of the requested letter)
- If a sent letter is accessed directly by link, it will redirect if the song id in localStorage is incorrect

Draft (readonly false)

- Users cannot access draft letters if not logged in

Migrations and Seed Information (Projects Two and Three Only)

Weirdly, users seeded into the database always return “incorrect password” but new users POSTed to the database can login just fine - unfortunately, there is not enough time to debug

User

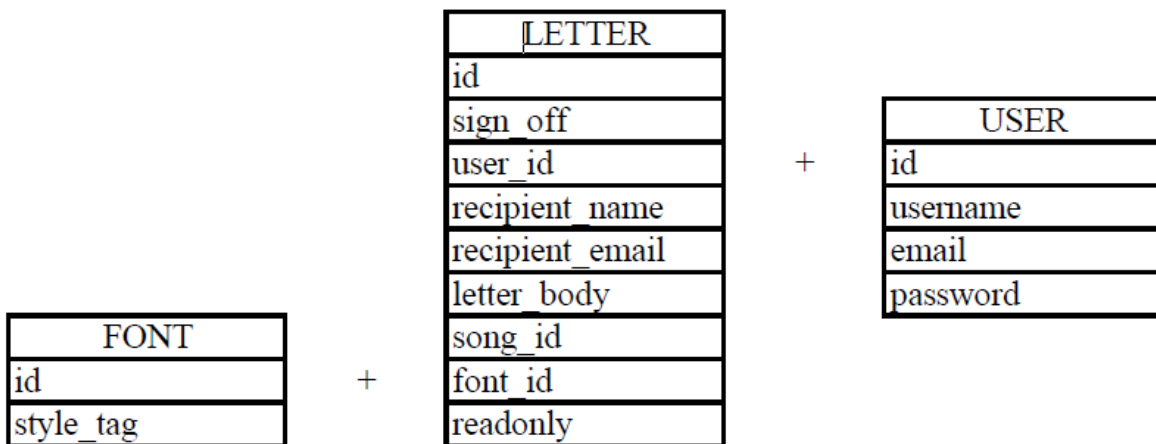
| id | username | email | password |
|----|-----------|---------------------|----------|
| 1 | username1 | username1@email.com | 123456 |
| 2 | username2 | username2@email.com | abcdef |
| 3 | username3 | username3@email.com | 654321 |
| 4 | username4 | username4@email.com | ABCDEF |

Fonts

| id | style_tag |
|----|-----------------------|
| 1 | "Caveat", cursive |
| 2 | "Lato", sans-serif |
| 3 | "Merriweather", serif |

Letters

| id | user_id | sign_off | recipient_email | letter_body | song_id | font_id | read only |
|------------------|---------|----------|----------------------|-----------------------------|-------------|---------|-----------|
| 1tsl810l3wa2sf9 | 1 | user1 | recipient1@email.com | This is a letter from user1 | 8L0XuwPzmgU | 2 | True |
| 1tslfrsl3wa3y5x | 2 | user2 | recipient2@email.com | This is a letter from user2 | jUh0l8iPPxl | 1 | False |
| 31tslfn8l3wa1ltr | 1 | user1 | recipient3@email.com | This is a letter from user1 | 8AwjyOFiEQ0 | 3 | True |
| 1tsl30sl3wa0un8 | 3 | user3 | recipient4@email.com | This is a letter from user3 | tG7wLK4aAOE | 2 | False |
| 1tslah4l4bu29h9 | 4 | user4 | recipient5@email.com | This is a letter from user4 | DUPsleQLNU0 | 2 | False |



Wireframes:



sign in

sign up



song. style. send.

Sign Up

Name: _____

Email: _____

Password: _____

next



song. style. send.

Sign In

Email: _____

Password: _____

sign-in



song. style. send.

Compose

Drafts

March 24, 2022 - 8:45pm - To Jennifer

June 7, 2022 - 7:21pm - To Jennifer

History

March 24, 2022 - 8:45pm - To Jennifer

June 7, 2022 - 7:21pm - To Jennifer



song. style. send.

You

Your name: _____

Your email: _____

next



song. style. send.

Them

Their name: _____

Their email: _____

next



song. style. send.

Song

Search: _____

next





song. style. send.

Style

Caveat - airy, playful, heartfelt

Lato - elegant, simple, touching

Merriweather - strong, classic, refined

next



song. style. send.

Send

Dear Kimberly,

*Since the day we met, I have woken
up with a feeling of the deepest
gratitude for life.*

*I know this time away has been
difficult but I wanted to share with
you just how much you mean to me...*

next

Now PLaying :

Out Loud - Syd feat. Kehlani



song. style. send.

Preview

Dear Kimberly,

Since the day we met, I have woken up with a feeling of the deepest gratitude for life.

I know this time away has been difficult but I wanted to share with you just how much you mean to me.

I never thought that I could feel with someone even when they're not with me. So even though we are apart, please know that I never feel away from you. You're always here with me, a thousand times a day, in my heart.

Love you,

Jeff

send

Now PLaying :

Out Loud - Syd feat. Kehlani



song. style. send.

Dear Kimberly,

Since the day we met, I have woken up with a feeling of the deepest gratitude for life.

I know this time away has been difficult but I wanted to share with you just how much you mean to me.

I never thought that I could feel with someone even when they're not with me. So even though we are apart, please know that I never feel away from you. You're always here with me, a thousand times a day, in my heart.

Love you,

Jeff

reply

Now PLaying :

Out Loud - Syd feat. Kehlani

Website “Components” and/or “Sections”

Common Components:

- Header with logo on left side that acts as a “home” button and tag line on the right side
 - Logo redirects to dashboard if logged in, if not redirects to homepage.
- If/Else a greeting and logout button if logged in, if not, add an href to access a Letter

Primary Views

- Root
 - Homepage
 - Option to go to login or signup page
 - Sign-Up
 - Fields for username, email, password
 - Sign-In
 - Fields for email and password
 - Dashboard
 - Option to logout
 - List of Draft items that redirect to that letter (editable) - GET request/res.render()
 - Draft items can be deleted (button has value of letter id)
 - Each Draft has a data value of letter id and song id
 - List of History items that redirect to that letter (readonly) - GET request/res.render()
 - Each Sent letter has a data value of letter id and song id
- Compose Letter Views
 - You
 - Field for your name - saved to sessionStorage
 - Them
 - Fields for their name and email - saved to sessionStorage
 - Song
 - Field to add a YT URL - saved to localStorage
 - Style
 - Radio buttons to select a font - saved to sessionStorage
 - Send
 - Textbox to capture letter body - POST request when done
 - Plays music
 - Confirm
 - A confirmation page with a button to redirect to dashboard
- Letter Views - plays music
 - Draft - autofills with data from database - GET request/res.render()
 - Displays the recipient's name, email, and the sender's sign off name (to be used in mailto template literal)
 - Displays the recipient's letter in a large textarea in the middle of the page

- Options to save or send message - PUT request
 - Draft becomes Sent when PUT changes readonly from false to true
- Drafts cannot be accessed by users that are not logged in
- Sent - autofills with data from database - GET request/res.render()
- Access
 - Fields to input letter id and song id
 - Access page will only redirect to requested Sent letter if song id matches that of the requested Letter
 - If a user attempts to directly access a letter by entering a link, it will redirect if the song id in localStorage does not match

Git Workflow, Website Directory Structure:

GIT Workflow

Production Branch Name: Main

Development Branch Name: develop

All developers will create branches off develop according to the naming convention:

feature/title-initials

bug/title-initials

- Developers will create branches off develop according to the naming convention
- Developers will Git Pull before beginning to code
- Developers will create a new branch to work on one feature/bug
- Developers will add a comment of the branch name they will be using to code an issue in the issue conversation thread
- All developers will create a pull request from their branch into develop
- Developers will not make changes to branches that they did not initiate unless a ticket issue is being transferred to another developer
- Developers must notify the team when they are prepared to make a pull request
- Pull requests must be made before merging with develop branch
- Once branches have been merged to develop, all developers must git pull
- Pull requests MUST be reviewed by review pair
- Final merge from develop to main will be reviewed by all group members

Code Review and Merge Requests

Due to the nature of how the project was developed and tasks reassigned, code reviews were not completed

Identify Name(s) of people doing code review and approving Merge Requests: (lists names here)

- Team members should “pair up” to perform code reviews and merge requests
- At the halfway point, new pairs should be recreated
- If there is an odd number of participants in the team, then the extra member can float between different pairs to code review,

| | | |
|----------|-------------|-------------|
| Week One | Team Member | Team Member |
| | | |
| Week Two | | |
| | | |

Directory Naming and Structure; Filenames and descriptions

Only list the important files and folders.

| <i>File Name</i> | <i>Function</i> | <i>Comments</i> |
|-------------------|--|------------------------------|
| server.js | Hosts express js and syncs database | |
| README.md | | |
| package-lock.json | | |
| package.json | Describes application dependences and commands | |
| LICENSE | | |
| .gitignore | Prevents pushing files to GitHub | |
| sample.env | Sample .env file to host sensitive variables | Fill with personal variables |

| | | |
|---------------------|---|---|
| .vscode | | |
| settings.json | | |
| config | | |
| connection.js | Connects to MySQL | |
| controllers | | |
| index.js | Links all the different routes using router and defines pathnames for each as well has hosts homepage route | Routes at this level render views in handlebars |
| compose-routes.js | | |
| dashboard-routes.js | | |
| homepage-routes.js | | |
| letter-routes.js | | |
| controllers/api | | |
| index.js | Links all the metadata routes using router and defines pathnames for each | Routes at this level handle metadata and API requests/queries |
| font-routes.js | | |
| letter-routes.js | | |
| user-routes.js | | |
| db | | |
| schema.sql | Establishes MySql database | |
| models | | |
| index.js | Defines the associations between models and exports them for use throughout the application | |
| Font.js | | |
| Letter.js | | |
| User.js | | Contains bcrypt methods |

| | | |
|---------------------------|---|---|
| public | | |
| public/img | | |
| fermata.png | Image for README | |
| fermata-2.png | Image for README | |
| Fermata-logos | | Different variations, including blue, black, white, etc. |
| Fermata-logos_favicon.png | Favicon for site | |
| public/javascript | | |
| main.js | Dynamically generates greeting text and handles logout logic | |
| sign-in.js | Handles validation and POSTs new user | |
| sign-up.js | Handles validation and POSTs a new sign in to start express.sessions | |
| draft.js | Is a GET request for a specific letter and can perform PUT requests to save changes or to change a Draft to readonly, triggering the mailto option | Autoplays music - cannot access drafts if not logged in |
| sent.js | Is a GET request for a specific letter and renders those fields on the page | Autoplays music - cannot access sent letter if localStorage song id does not match that of the requested letter |
| access.js | A buffer page between a recipient and a sent letter to increase engagement (for autoplay to work) and to set the song id in local storage (so the iframe will load) | |
| dashboard-draft.js | Handles delete draft functionality and redirects to draft, while setting the song id to local storage | |

| | | |
|----------------------------------|--|--|
| dashboard-sent.js | Handles redirect to sent letter while setting the song id to local storage | |
| public/javascript/compose-letter | | |
| you.js | Captures sender's name | For mailto template literal |
| them.js | Capture's recipient's name and email | For mailto template literal |
| song.js | Capture's YT URL and clips the song id to save to localStorage | |
| style.js | Captures font id from selected radio input | |
| send.js | Captures letter body | Autoplays music |
| public/stylesheets | | |
| style.css | | |
| seeds | | |
| database-test.js | Tests the join logic of models and console logs SELECT | |
| index.js | Executes seeding of database on "npm run seed" | |
| font-seeds.js | | |
| letter-seeds.js | | |
| user-seeds.js | | |
| utils | | |
| auth.js | Custom middleware to redirect user to homepage if not logged in | Used for dashboard and compose letter routes |
| helpers.js | Contains date formatting for Handlebars | |
| views | | |
| access.handlebars | | |
| confirm.handlebars | | |

| | | |
|-------------------------------|---|---|
| dashboard.handlebars | Dynamically renders two lists of letters (drafts and sent) depending on readonly property | |
| draft.handlebars | | |
| homepage.handlebars | | |
| send.handlebars | | |
| sent.handlebars | | |
| sign-in.handlebars | | |
| sign-up.handlebars | | |
| song.handlebars | | |
| style.handlebars | | |
| them.handlebars | | |
| you.handlebars | | |
| views/layouts | | |
| main.handlebars | | |
| documentation-and-additionals | | |
| additional/feature | Contains Lewis's unintegrated modals | There are two versions, a jquery + bootstrap version and a CSS + JS version |
| additional/sandbox | Contains Ali's first draft of the site's layout in static HTML pages | |
| additional/spotify | Contains Sharon's first attempt at the Spotify API using PHP | |
| additional/spotify-node | Contains Sharon's second attempt at the Spotify API using npm spotify-web-api-node | |
| fermata-proposal.pdf | Contains the completed documentation for the project | First version on project start, final version on project end |

| | | |
|-------------------|--|---|
| additional images | Contains Ali's example music images for the demo | Demo video here: https://drive.google.com/drive/folders/1O9ipDI5nzW2gf0zexIAi3e8is9b8CXaN?usp=sharing |
|-------------------|--|---|



“Stuck Time” Agreement:

The team must identify and agree to a time frame that a team member must use as a limit before reaching out to (a) fellow team members and then (b) the instructional team (in-class) for support. That way, *no one should be stuck for ‘days’ or ‘many hours’ on a task.*

We the team members of (insert project team number or team name here) agree that the following period of time of investigation (with evidence) is allowed before identifying themselves as being “stuck” or “bottlenecked”, in which case, the team member can then seek the assistance of other team members, the instructional team, or AskBCS for assistance.

Period of time permitted before asking to seek assistance (hours): 2 hours

As agreed upon by the following team members (name and signature below)

| Name | Signature |
|----------------------------|--|
| Ali Momen |  |
| Sharon Christensen | S. Christensen |
| IWHERE IFEANYICHUKWU LEWIS | I.C.I |
| Kyle Baylis |  |

| | |
|---------------------|--|
| Veronica Tai Chi To |  |
|---------------------|--|

Retrospectives – To Be Handed in At the Project Midpoint

Each team is to hold a *retrospective* meeting at the half-way point of the project period.

Retrospective tentatively scheduled for: May 31st

Retrospective not completed.

GitHub Repository Link, Production Website Link Submission Dates

- Presentation Date - June 2
- Project Submission Due Date - June 10
- Ideal Submission Date - June 5

Additional Notes: