Curated For You By You

Team : Cache Me If You Can

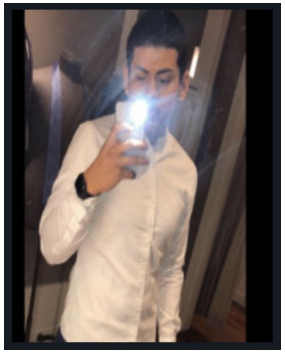
Agenda

- Team Member Roles and Responsibilities **
- Improvements from Feedback
- Project Description **
- Team Working Agreement**
- Personas
- MVP
- Algorithms
- Rest API/Examples
- Technologies
- Diagrams
- Product Backlog
- Sprint 2 Backlog
- Burndown Chart
- User Stories / Test Cases
- Project Schedule
- Retrospective
- AI Disclosure
- Github Link

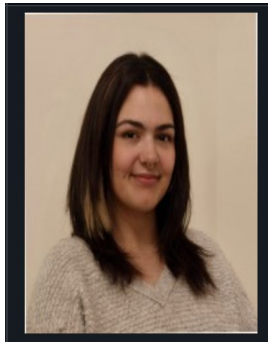
Team Members

Emmet Allen- Scrum Master / Systems Architect

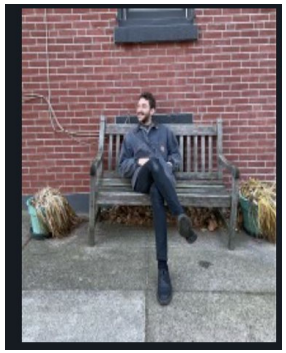
Ben Smith - Backend Dev / Integrations



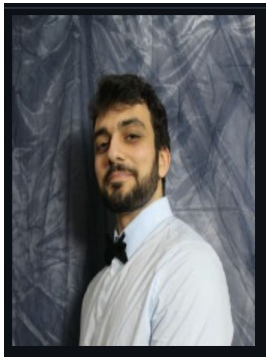
Brian Moran - Full Stack Dev



Front End Dev Lead



Pete Adams - Back End Lead / Data Team



Christos Markakis - Data Team / Full Stack Dev

Jalen Gill - Front End Dev / Integrations

Improvements from Feedback

- Reviewed peer and instructor feedback on project presentation and group wiki page.
- Added relevant photos throughout the slideshow more specifically the Team Member page.
- Incorporated links into the wiki page to meet syllabus standard.
- Added download links for current sprint and previous ones
- Adjusted agenda slides to be accurate
    - User stories
    - Burndown chart
    - Test cases

Project Description

Curated By You, For You

Curated By You, For You is a web application that helps people discover music and artists based on location.

- **For:** musicians, fans, and music industry professionals
- **Who:** want to be discovered, discover others, and stay connected with trends in local music scenes

What It Does
Curated By You, For You searches for information about local artists and organizes it by genre and location, so users can make meaningful local discoveries.

Benefit Outcomes
By using this app, people will become more knowledgeable about local music scenes, discover new bands they enjoy, and help smaller musicians grow larger fanbases.

Team Working Agreement

- Agreed to meet twice a week Tuesday @ 6:00PM EST
- Pairing is necessary to work on a story
- If one person in pair is unavailable the other person in the pair is expected to pick up the work
- Pairs switch for either each story or each sprint depending on need for expertise if necessary
- Stories will be groomed and pointed before opening a sprint
- Stories need a kickoff ceremony before work is started on them consisting of 1 or 2 other members

Primary User : DJ Two Tone

- Age, 23
  Location : New York
  Occupation: DJ
- DJ Two Tone is a local DJ who performs at clubs and parties. While he often mixes popular, well-known tracks into his sets, he's now looking to incorporate more local cultural music genres to create unique experiences that celebrate the community and connect audiences with the sounds of their own scene.
- **Issue:** Difficulty accessing and integrating unfamiliar cultural music genres—such as Jakarta folk—into performance sets, requiring extensive independent research through scattered sources like record stores, forums, and music databases.
- **Solution:** The platform enables DJs and creators to easily discover and explore local and niche genres by searching specific locations and styles (e.g., Jakarta + Folk Music), instantly connecting them with relevant artists, bands, and tracks to blend seamlessly with their own sets.

Primary User 2 : Tobi

- Age, 29
  Location : West Nyack, NY
  Occupation: Musician
- Tobi is a musician looking for his big break into the music industry. He plans on starting in the local scene to build his experience and connections in the industry.
- **Issue:** Music creators face barriers in networking, collaboration, and expanding their creative expertise, limiting growth and visibility.
- **Solution:** Our platform empowers creators by fostering meaningful connections, driving community-driven promotion, and offering tools and resources to continuously expand their skill set.

Primary User 3: Glenn

- Age, 58
  Location : Chattanooga, TN
  Occupation: Store Manager

- Glenn is a store manager that often listens to music in his spare time. He uses apps like Spotify and Apple Music to listen to his preferred music and popular music of today. He wants to immerse himself in new music by discovering new artists locally.

- **Issue:** As a music fan, it can be hard to discover new local performers beyond the big-name acts. Relying on word of mouth or chance encounters often means missing out on fresh talent, especially younger or emerging artists who aren't promoted widely.

- **Solution:** The platform makes it easy for fans to discover new and local artists tailored to their tastes. By combining personal preferences with location-based recommendations, fans can explore performers they might not otherwise find—helping them connect with fresh music while supporting their local scene.

MVP Feature Prioritization

- **Feature Name:** Location and Genre Based Search
    - Purpose: Searching Capabilities for Music Artist
    - User story: As a User, I want to search for artists using a specific location and music genre, so that when I complete the search, I'm provided with a list of artists who are from that location and perform that genre.
- **Feature Name:** Artist Music Listing
    - Purpose: Provide Artist Music Collection
    - User story: As a User, when I search an artist using the application, I want to be able to click on an artist, so that I can see their discography

CM

Algorithms:

Entity Resolution / Deduplication: cleaning and removing dupes from our corpus of data

Hybrid Ranking (RRF): merging tables of similar genres on location proximity and popularity

Inverted Index: classic search engine implementation, used to search for band name, genre, location, and/or description

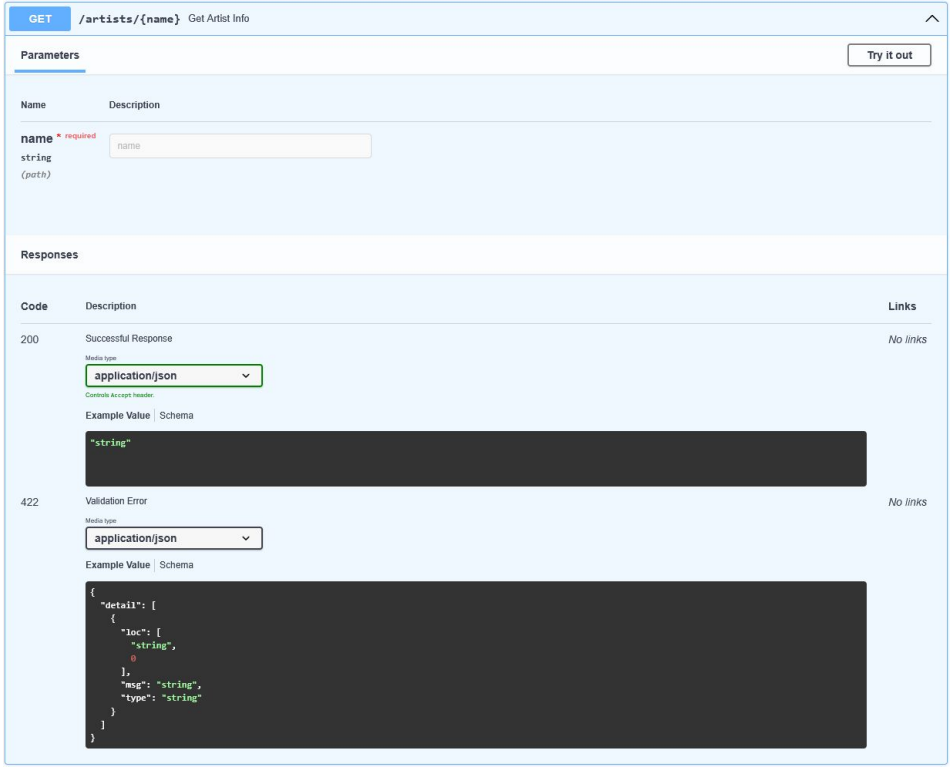
Backend REST API

The heart of the backend is a REST API that's queryable through HTTP GET requests. Data is accessed from our local repository of artists and then transformed into a JSON payload.

We've expanded the dataset and endpoints considerably since Sprint 1.

/ - (root that returns the current API spec and version)
/artists
/artists/{name}
/artists/{name}/description
/artists/{name}/image
/artists/{name}/albums
/artists/genre
/artists/city
/albums{title}/description

API Request Example

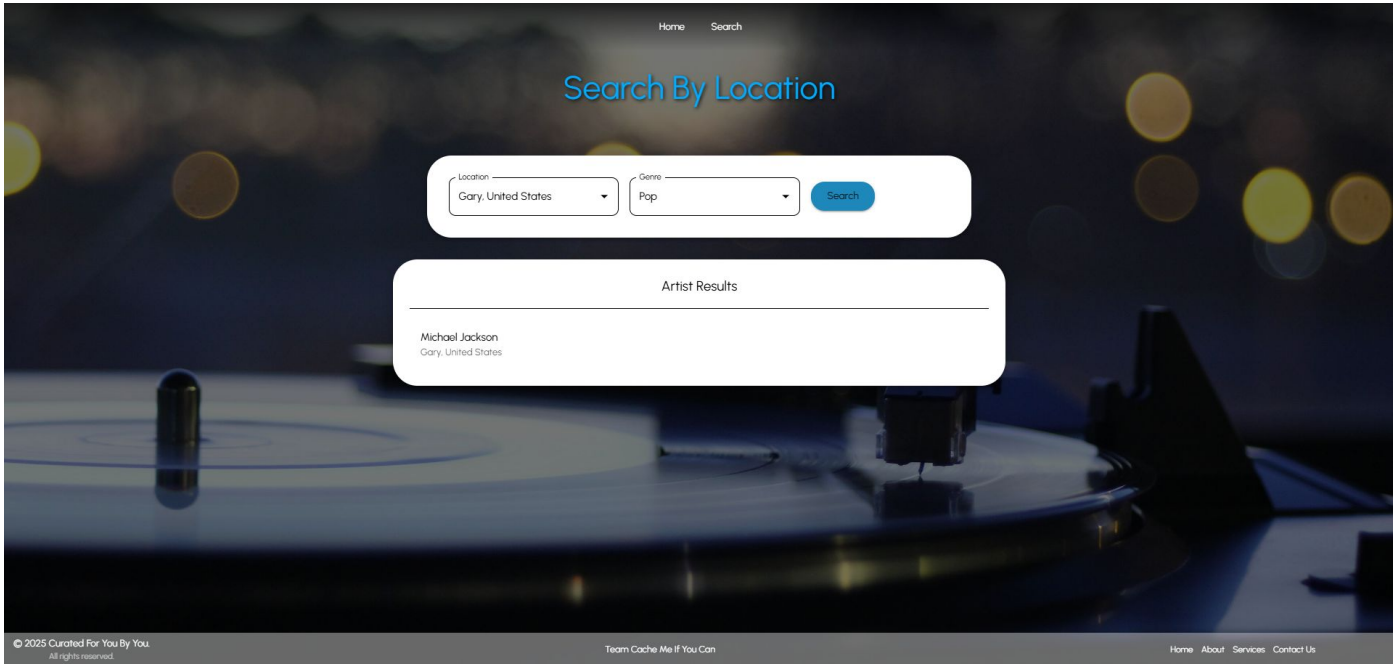


Updated Dataset Example


```
{
  "rock": [
    {
      "name": "Bruce Springsteen",
      "country": "United States",
      "city": "Long Branch",
      "summary": "Bruce Frederick Joseph Springsteen (born September 23, 1949) is an American singer, songwriter, and guitarist. Nicknamed \"the Boss\",
      "image": "https://iconicimages.net/app/uploads/2021/07/BRSNewsHeader.jpg",
      "albums": [
        {
          "title": "Born to Run",
          "year": 1975,
          "image": "https://upload.wikimedia.org/wikipedia/en/thumb/7/7a/Born_to_Run_%28Front_Cover%29.jpg/250px-Born_to_Run_%28Front_Cover%29.jpg",
          "rating": 5,
          "tracks": [
            {
              "title": "Jungleland",
              "duration": 240
            },
            {
              "title": "Thunder Road",
              "duration": 210
            }
          ]
        }
      ]
    }
  ],
```

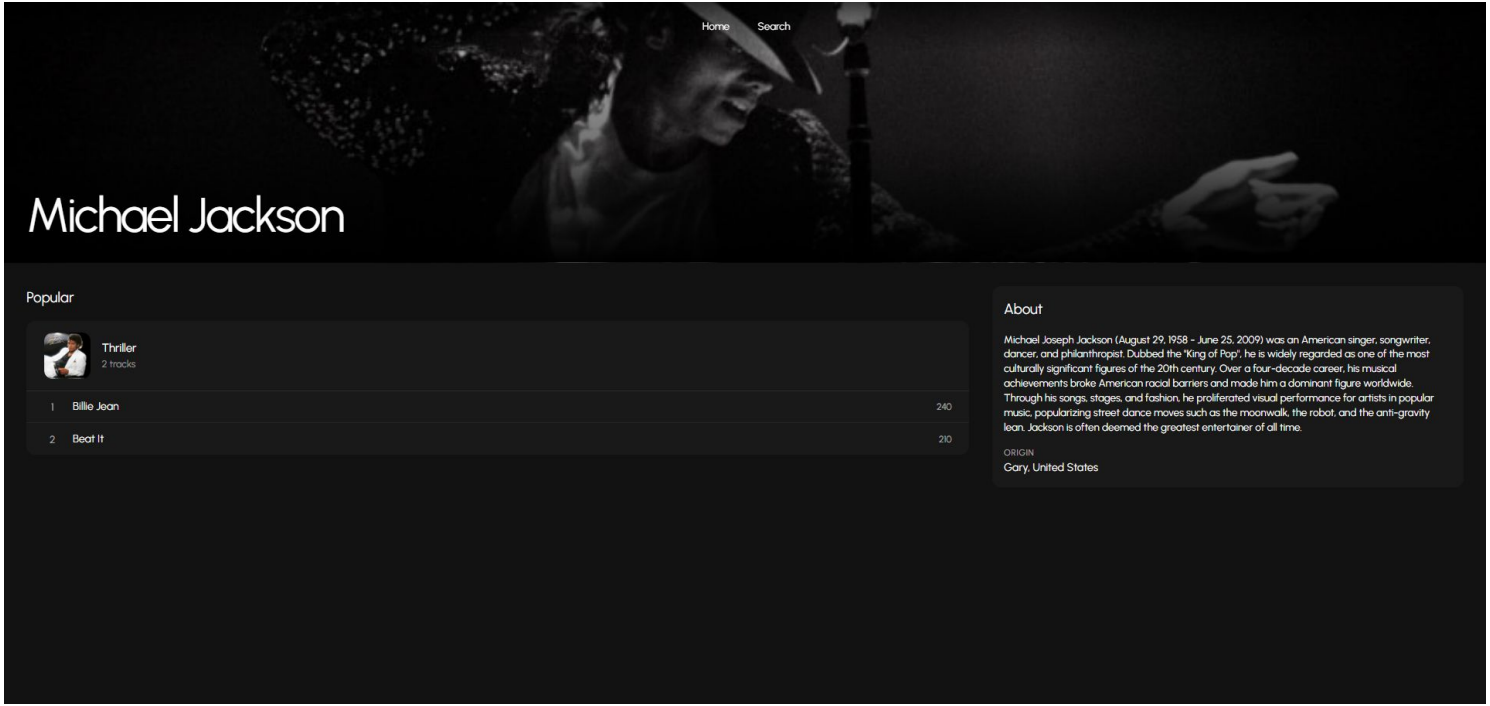
Frontend Search

The search page has been updated with a new layout, allowing for search results to be focused more.



Artist Page

After clicking an artist, the user is taken to a details page that shows more information about the artist.



Technical Implementation

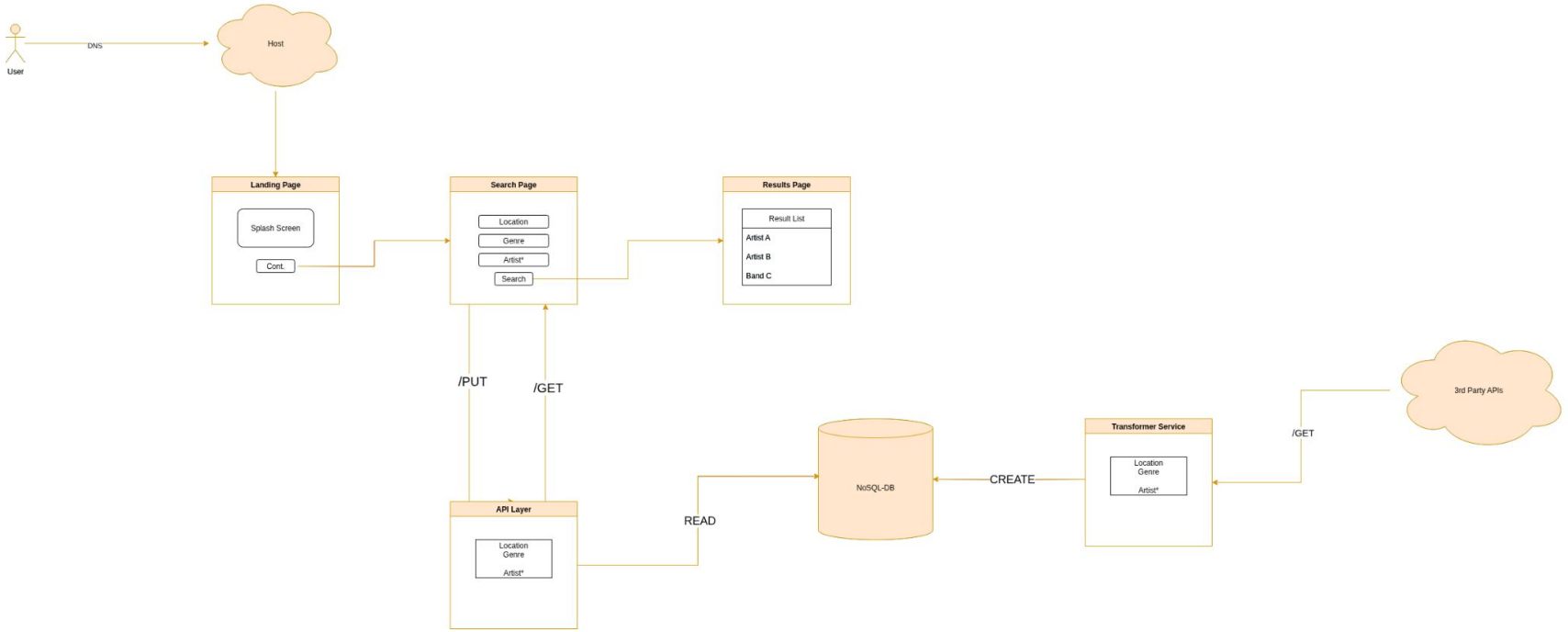
Platform / Technology:

Backend- Python, FastAPI, uvicorn, AWS

Frontend- React, TypeScript, Material.UI (CSS)

3rd Party APIs : Spotify API, Music Brainz API, EveryNoise.com

Architecture Diagram



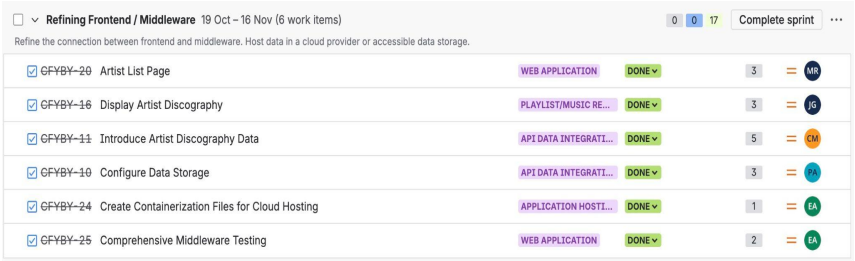
Product Backlog

Key	Issue Type	Parent	Summary	Status	Assignee	Sprint	Start date	Inferred start dat	Due date	Inferred due date
CFYBY-1	Epic		Web Application	To Do			2025/09/21	2025/09/21	2025/10/21	2025/10/21
CFYBY-6	Task	CFYBY-1	Location Search / Search - Web page	Done	Jalen Gill	Minimum Viable Product		2025/09/21		2025/10/21
CFYBY-7	Task	CFYBY-1	Genre / Search Web page	Done	Christos Markakis	Minimum Viable Product		2025/09/21		2025/10/21
CFYBY-5	Task	CFYBY-1	User Landing Page	Done	Maya Rivera	Minimum Viable Product		2025/09/21		2025/10/21
CFYBY-20	Task	CFYBY-1	Artist List Page	Done	Maya Rivera	Refining Frontend / Middleware		2025/10/19		2025/11/16
CFYBY-25	Task	CFYBY-1	Comprehensive Middleware Testing	Done	Emmet Allen	Refining Frontend / Middleware		2025/10/19		2025/11/16
CFYBY-9	Task	CFYBY-1	Artist Information Page	To Do						
CFYBY-23	Task	CFYBY-1	Improve Accessibility and Responsiveness to Frontend	To Do						
CFYBY-2	Epic		API Data Integration	To Do			2025/10/16	2025/10/16	2025/11/12	2025/11/12
CFYBY-19	Task	CFYBY-2	3rd Party API Discovery	Done	Pete Adams	Minimum Viable Product		2025/09/21		2025/10/21
CFYBY-8	Task	CFYBY-2	3rd Party API Ingestion	Done	Pete Adams	Minimum Viable Product		2025/09/21		2025/10/21
CFYBY-11	Task	CFYBY-2	Introduce Artist Discography Data	Done	Christos Markakis	Refining Frontend / Middleware		2025/10/19		2025/11/16
CFYBY-10	Task	CFYBY-2	Configure Data Storage	Done	Pete Adams	Refining Frontend / Middleware		2025/10/19		2025/11/16
CFYBY-3	Epic		Location Based Artist Data	To Do			2025/11/07	2025/11/07	2025/12/04	2025/12/04
CFYBY-12	Task	CFYBY-3	Location Search	To Do						
CFYBY-13	Task	CFYBY-3	Location Based Artist Search Data Retrieval	To Do						
CFYBY-4	Epic		Playlist/Music Recommendation	To Do			2025/11/12	2025/11/12	2025/12/06	2025/12/06
CFYBY-16	Task	CFYBY-4	Display Artist Discography	Done	Jalen Gill	Refining Frontend / Middleware		2025/10/19		2025/11/16
CFYBY-14	Task	CFYBY-4	Spotify OAuth Integration	To Do						
CFYBY-15	Task	CFYBY-4	Spotify API Integration	To Do						
CFYBY-17	Task	CFYBY-4	Playlist Creation	To Do						
CFYBY-18	Task	CFYBY-4	Playlist Integration	To Do						
CFYBY-26	Epic		Application Hosting	To Do			2025/11/17	2025/11/17	2025/12/03	2025/12/03
CFYBY-24	Task	CFYBY-26	Create Containerization Files for Cloud Hosting	Done	Emmet Allen	Refining Frontend / Middleware		2025/10/19		2025/11/16
CFYBY-22	Task	CFYBY-26	Configure MiddleWare Hosting	To Do						
CFYBY-21	Task	CFYBY-26	Configure FrontEnd Hosting	To Do						

Sprint 2 Backlog

These were the stories completed in Sprint 2. Each story is estimated using story points, which are reflected in the number column.

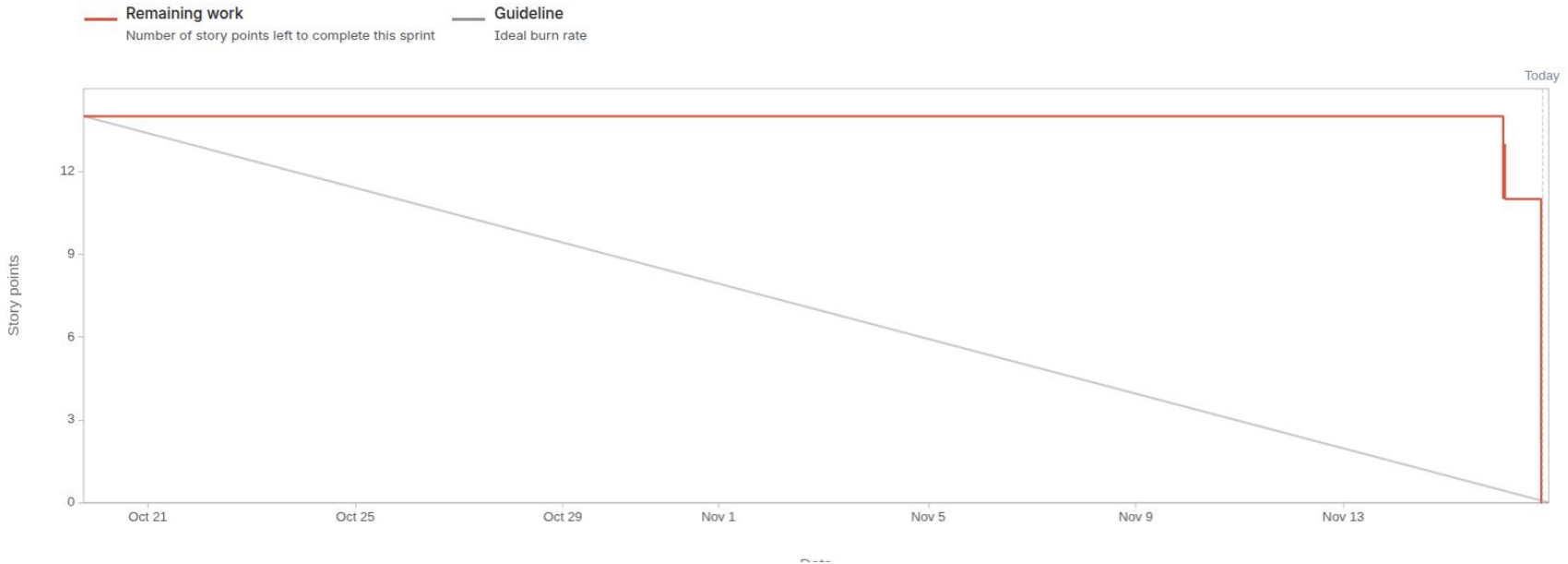
- Story points are a measurement used within Agile to describe the complexity of a given task
- For example, a task can be given either a fibonacci number (1,2,3,5) or T-shirt sizes (S,M,L,XL)
- It's used to ensure that the team members know what the task may entail before taking up the story or feature



Burndown Chart

Date - October 19th, 2025 - November 16th, 2025

Sprint goal - Refine the connection between frontend and middleware. Host data in a cloud provider or accessible data storage.



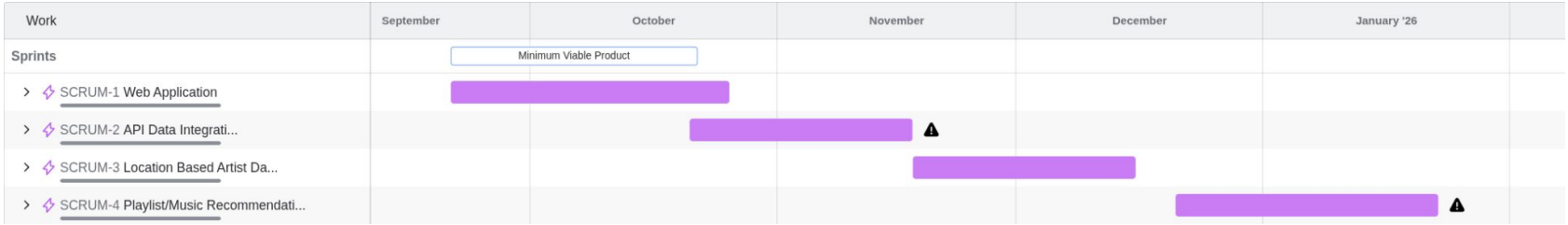
User Stories / Test Cases

ID	Epic	User Story	Goals	Acceptance Criteria	Test Cases	Points	Sprint
Artist List Page	Web Application	As a user, When I submit my search inputs, I want to be shown a webpage that has the results for that search.	Should have artist information cards presented on the page. Should integrate with the Middleware API to retrieve the search re	Acceptance Criteria User can See artists info on the webpage. Artist info card must include: Name City Country Should integrate with the Middleware AP	Happy Path: Successful search displays multiple artists with complete information Given I am on the search results page And the Middleware API is available and returns a list of artists When the search inputs are successfully submitted Then I should see the search results webpage And the webpage should display multiple artist information cards And for each artist, I should see their Name, City, and Country Sad Path: Search returns no results Given I am on the search results page And the Middleware API is available but returns an empty list of artists When the search inputs are successfully submitted Then I should see the search results webpage And I should see a clear message indicating "No artists found for your search" And I should not see any artist information cards Neutral Path: Artist data is partially complete Given I am on the search results page And the Middleware API returns an artist where the "City" or "Country" field is null When the search inputs are successfully submitted Then I should see the artist's Name displayed on an information card And for the missing fields, I should see a clear indicator like "N/A"	3	2
Display Artist Discography	API Data Integration	As a user, When I am on the artist information page, I want to be able to see the artist discography if available.	Should have a list of albums / songs made by the artist Should display the artist information in a subsection of the page List should contain no more than 5 songs	User can see A small list of songs Songs need to pertain to the artist An Artist's core information (name, etc.) i	Happy Path: Artist has 5 or more songs/albums and displays the maximum allowed Given I am on the artist information page And the API provides 10 songs/albums for the artist When the page loads the discography section Then I should see the artist's core information displayed And I should see a list of songs/albums And the list should contain exactly 5 songs/albums Sad Path: Artist has no discography data available Given I am on the artist information page And the API provides an empty list for the artist's discography When the page loads the discography section Then I should see the artist's core information displayed And I should see a message indicating "No discography available" or similar And I should not see any song or album items listed	3	2
Introduce Artist Discography Data	API Data Integration	As a middleware, When I make a call to the CFYBY database I want to be able to pull artist info objects that include artist discography	Data should be in the artist object Data is artist specific If no discography data, data should be an empty JSON object	Acceptance Criteria Middleware Should be able to query discography Should recieve artist discography with ar	Happy Path: Middleware successfully pulls an artist with a full discography Given the CFYBY database is available And the requested artist has existing discography data When the Middleware makes a single query for the artist object Then the Middleware should receive a response with a 200 (OK) status And the response body should contain the main artist object And the artist object should contain a field named "albums" And the "albums" field should be a list of song/album objects Sad Path: Middleware handles a database error during retrieval Given the CFYBY database is unavailable or the query is malformed When the Middleware attempts to query the artist object Then the Middleware should log a database connection or query error And the Middleware should return an appropriate error status (e.g., 500 Internal Server Error) to the call	5	2

User Stories / Test Cases

ID	Epic	User Story	Goals	Acceptance Criteria	Test Cases	Points	Sprint
Configure Data Storage	API Data Integration	As a middleware, When I receive a request, I want to be able to access the data that is stored in a cloud service.	Should be easily accessible Data should hold its integrity and structure Data should be hosted in a cloud service	Must be reachable by API key or token Data must have same JSON structure Data should be hosted on an accessible	Happy Path: Middleware successfully retrieves and validates cloud data structure Given the cloud data service is reachable via its endpoint And the Middleware has a valid authentication token/key When the Middleware makes a request to the cloud service Then the response status code should be 200 (OK) And the retrieved data should adhere to the predefined JSON schema And the retrieved data should contain the expected primary fields Sad Path: Middleware fails to access data with invalid credentials Given the cloud data service is reachable via its endpoint And the Middleware attempts to access the service with an invalid or expired token/key When the Middleware makes a request to the cloud service Then the response status code should be 401 (Unauthorized) or 403 (Forbidden) And the Middleware should log the access failure securely And the Middleware should return an appropriate error message to the calling client	3	2
Create Containerization Files for Cloud Host	Application Hosting	As a developer When I want to run the full application I want to be able to do so by running the application and its components	The application is able to be hosted within docker The README is updated to support initialization with both docker	Developers can Run the application by following the Rea Interact with the application without any r	Happy Path: Developer successfully starts the entire application stack using Docker Compose Given the developer has Docker installed on their machine And the developer has pulled the latest repository code When the developer executes the documented "docker-compose up" command Then all required application services (e.g., middleware, database) should start successfully And the developer should see logs indicating all services are "healthy" or "running" And the application should be reachable via the documented local host port	1	2
Comprehensive Middleware Testing	Web Application	As a developer When I run the test for the middleware application I want to be shown all test cases for the middleware including happy an	All endpoints of the middleware should be assigned a proper api t All test should validate data functionality and test case (happy/sac	Developers can Run pyunit test for middleware, where al Can quickly understand the happy/sad p	Happy Path: Middleware Test Suite successfully runs Happy Path tests Given the test environment is correctly configured (API keys valid, external services available) And the test runner executes all tests for the "/search" endpoint When the test case named "test_search_returns_multiple_artists_200" is run Then the test should assert that the response status code is 200 (OK) And the test should assert the response body contains a valid JSON list of artists And the test result should be "PASS" (green) Sad Path: Middleware Test Suite validates Sad Path for Invalid Request Given the test runner executes all tests for the "/artist/{id}" endpoint When the test case named "test_get_artist_invalid_id_returns_400" is run with a non-integer or malform Then the test should assert that the response status code is 400 (Bad Request) And the test should assert that the error message clearly states the input validation failure And the test result should be "PASS" (green)	2	2

Project Schedule



- Sprint 0 : September 2 - September 22
- Sprint 1 : September 23 - October 20
- Sprint 2 : October 21 - November 17
- Sprint 3 : Completion of MVP: November 18 - December 15

Weekly Meetings: Tuesdays at 6:00PM

Team Retro

What went well	What didn't go well	Action Items
• All task completed on Jira • Was able to complete additional task • Cross team occurred • More team meetings to follow up • Feature Branching • Improved Communication • Improved past barriers • Level of commitment	• Challenges dealing with class and other responsibilities	• Maintaining same level of commitment

AI Disclosure

These AI Tools were used to aid us in brainstorming and project development:

- Gemini CLI
- ChatGPT

Thank You!

Wiki Page/Github Repo: https://github.com/htmw/F2025-Async/wiki