Final Report

● Ungraded

1 Day, 23 Hours Late

Group

James Baker
Jasper Zhu
Chanya Thanglerdsumpan
...and 3 more
View or edit group

Total Points

- / 15 pts

Question 1

Report 15 pts

Question assigned to the following page: 1

NETS 2130 Project Final Report: Loopify

James Baker, Grace Thanglerdsumpan, Angie Cao, Jasper Zhu, Namita Sajai, Avi Upadhyaula

---

Table of Contents

Question assigned to the following page: 1

Basic Project Information

1. **Name of your project**
    a. Loopify
2. **Name of your teammates**
    a. James Baker, Grace Thanglerdsumpan, Angie Cao, Jasper Zhu, Namita Sajai, Avi Upadhyayula
3. **One sentence description of your project:**
    a. Loopify is a music discovery platform that uses crowdsourcing to match songs with similar vibes, allowing users to curate and vote on theme-based playlists like “road trip anthems” or “late-night thoughts,” with AI-generated playlists to enhance recommendations.
4. **Logo for your project:**
    a. 

5. **What problem does it solve?**
    a. This platform addresses the challenge of discovering songs that capture a specific vibe or mood. Traditional music streaming services like Spotify and Apple Music do have similar services, but they use algorithms that we feel often fall short of matching the exact “feel” or thematic intention users want in their playlists (e.g., songs perfect for a rainy day, a workout, or a nostalgic road trip). By combining crowdsourcing with AI, this app allows users to collaboratively define vibes and curate thematic playlists with accuracy, resulting in a more personalized and community-driven music discovery experience.
6. **What similar projects exist?**
    a. One similar project is Spotify’s Discover Weekly & Playlist Recommendations, which use algos that analyze user listening patterns to give users new playlists each week. However, it’s a one-way process without crowd feedback, so it often lacks the exact "vibe-matching" that community input could provide. Another similar one is 8tracks which is a website that allows users to create vibe-based playlists, but it is not up anymore.
7. **What type of project is it?**
    a. This project is **a hybrid AI + crowdsourcing tool**.
8. **What was the main focus of your team’s effort**
    a. The main focus of our team’s effort is somewhere in between engineering a complex system and conducting an in-depth analysis of data. We are building a

Question assigned to the following page: 1

platform that integrates crowdsourcing functionality, music, and people's personal music taste while relying heavily on user-contributed data for vibe-matching and playlist curation. While building this project, this combination requires both advanced system design and thorough data analysis to ensure the platform is engaging and accurate in delivering personalized music discovery experiences.

9. **How does your project work? Describe each of the steps involved in your project. What parts are done by the crowd, and what parts will be done automatically.**
    a. Crowd workers pick a playlist they want to help curate. They vote on whether songs fit the playlist description or not.
    b. Parts that are done automatically for them is voting aggregation part to create the actual playlist. The songs are also randomly recommedned to them using a Random Song Generator API.
10. **Provide a link to your final presentation video. Give the full URL to your YouTube video or your Google Drive video and make sure it is publicly available (OK to keep it unlisted).**
    a. https://drive.google.com/file/d/1QBJYsUSHxUdOgEjlalDMsAToayKIugxp/view?usp=sharing
11. **Which two sections below did you pick for your in-depth analysis?**
    a. **The two sections we picked for our in-depth analysis is looking into the quality of the votes and explori**
    b. Technical challenges and Quality Control/Aggregation

The Crowd

1. **Who are the members of your crowd?**
    a. The members of our crowd are anyone who enjoys listening to music, or discovering new music. The age demographic can be anyone, but most likely young teens to older adults – anyone that listens to music.
2. **For your final project, did you simulate the crowd or run a real experiment?**
    a. We used a real crowd.
3. **If the crowd was real, how did you recruit participants?**
    a. We reached out to friends and acquaintances, and asked them to participate! We also solicited ideas for new features and UX improvements from them. One common ask was the ability to create new playlists and open them up to voting, which was a feature we'd already planned to implement. It was great to see the interest in the platform, and to know that there was real enthusiasm for the product.
4. **How many unique participants did you have?**
    a. We had 120 unique voters in our database.

Incentives

Question assigned to the following page: 1

1. **What motivation does the crowd have for participating in your project?**
    a. For our project, the crowd's motivation was **altruism** and **enjoyment.**
2. **How do you incentivize the crowd to participate? Please write 1-3 paragraphs giving the specifics of how you incentivize the crowd. If your crowd is simulated, then what would you need to do to incentivize a real crowd?**
    a. In this project, our primary incentive for participants was the natural enjoyment derived from discovering and sharing music that resonates with them. Rather than offering tangible rewards, we relied on participants' intrinsic motivation to explore new songs, contribute their favorites to themed playlists, and help create a vibrant, community-driven music ecosystem. Given that many of our early users were friends and acquaintances, there was an element of altruism at play, as they were eager to support our initiative and help improve the platform by offering honest feedback, exploring new features, and providing new song suggestions.
    b. No formal incentives like payment, reputation points, or badges were introduced at this stage. Instead, the draw was the excitement of collaborating in a shared music discovery experience, knowing that their input could shape playlists that other enthusiasts would enjoy. This approach leaned heavily on the goodwill of our crowd and their natural interest in music, rather than extrinsic rewards.
3. **Did you perform any analysis comparing different incentives?**
    a. Because our project was in its early phase and relied on friends and acquaintances, we did not formally compare different incentive structures. Our recruitment was informal and voluntary, with participants motivated by their personal enjoyment of music and the desire to help improve the platform. Had we implemented alternative incentives—such as gamification elements or status recognition—future analyses could measure changes in participation, the number of new playlist submissions, or increased engagement in voting and commenting. However, at this stage, no comparative analysis was conducted since our primary focus was simply on creating a fun, community-driven environment that friends were happy to support.
4. **If you compared different incentives, what analysis did you perform? If you have a graph analyzing incentives, include a graph here.**
    a. Not applicable

What the crowd gives you

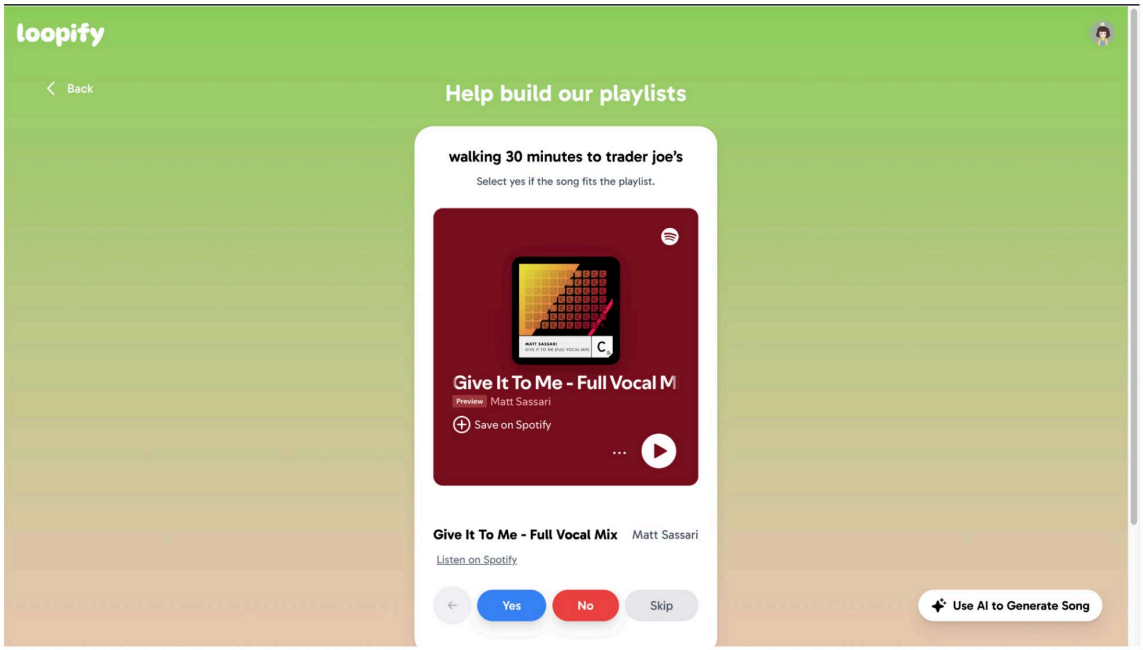
1. **What does the crowd provide for you?**
    a. The crowd gives us insights into what kind of songs fit into people's tastes and preferences for playlists with specific moods or vibes. They do so by clicking on the different playlists, or generating their own ones, and voting on whether songs match the vibes of the current playlist or not.
2. **Is this something that could be automated?**

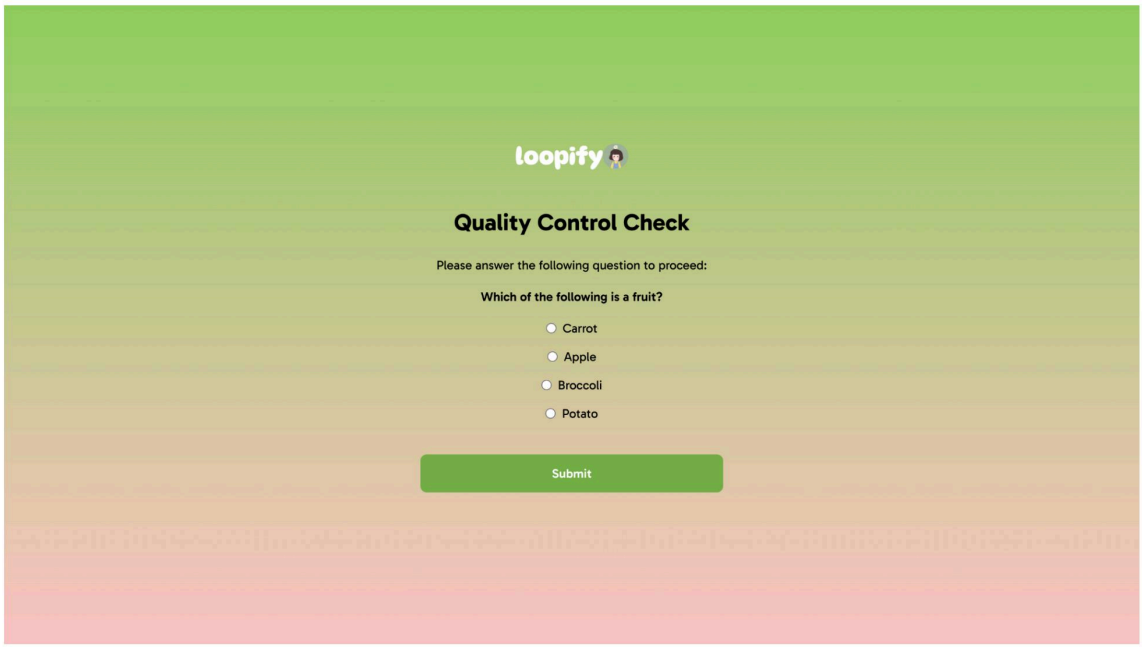
Question assigned to the following page: 1

    a. We actually tested this out during one of our milestones to realize that curating playlists is better if it is sourced or created by humans. From our analysis, we had people vote on whether they liked a playlist generated from crowd source results or from AI-generated results more. From our vote, we found out that the majority of the voters preferred the playlist that was generated by workers.

3. **If it could be automated, say how. If it is difficult or impossible to automate, say why.**
    a. It actually would not be too difficult to automate the playlist curation process. Most of the hard work would be working on a prompt to input this kind of LLM, then having a detailed and structured description of what songs we are looking for in the specific playlist.
4. **Did you train a machine learning component from what the crowd gave you?**
    a. Yes, mostly in the analysis part.
5. **If you trained a machine learning component, describe what you did.**
    a. We initially planned to use generative AI tools in music to create the "ultimate" song that represents a playlist using all the songs that were aggregated from users' votes. Hence, we tried implementing that with Suno AI, and we also used GPT to generate a playlist to compare with what our crowd "gave" us in our data analysis part as well.
6. **Did you analyze the quality of the machine learning component? For instance, did you compare its quality against crowd workers using an n-fold cross validation?**
    a. We analyzed it by having crowd workers vote for the best playlist without knowing which is which.
7. **If you have a graph analyzing a machine learning component, include the graph here.**
    a. N/A
8. **Did you create a user interface for the crowd workers? Answer yes even if it's something simple like a HTML form on MTurk.**
    a. Yes, we coded our own user interface using React and Spotify's API to pull songs that users could listen to and then vote on.
9. **If yes, please include a screenshot of the crowd-facing user interface in your report. You can include multiple screenshots if you want.**

Question assigned to the following page: 1


loopify
Back
Help build our playlists
walking 30 minutes to trader joe's
Select yes if the song fits the playlist.
Give It To Me - Full Vocal M
Preview
Matt Sassari
Save on Spotify
Give It To Me - Full Vocal Mix
Matt Sassari
Listen on Spotify
Yes
No
Skip
Use AI to Generate Song


loopify
Quality Control Check
Please answer the following question to proceed:
Which of the following is a fruit?
Carrot
Apple
Broccoli
Potato
Submit

Question assigned to the following page: 1

loopify

Hitting The Gym

High-energy tracks to fuel your workouts.

Last updated: Dec 10, 2024 • 15 songs, 2 hours 10 minutes

Play

#	TITLE	ALBUM	
1	Eye of the Tiger Survivor	Rocky III	4:05
2	Lose Yourself Eminem	8 Mile	5:20
3	Stronger Kanye West	Graduation	5:11
4	We Will Rock You Queen	News of the World	2:02
5	Can't Hold Us Macklemore & Ryan Lewis	The Heist	4:18

loopify

loopify

discover new loops of sound

Search for playlist category

Playlist Results →

Quality Control Check

Help us crowdsource more vibes. Select the best songs for...

hitting the gym

getting over a three week-long situationship

walking 30 minutes to trader joe's

locking in before 11:59 PM deadline

loopify

Playlists

hitting the gym

getting over a three week-long situationship

walking 30 minutes to trader joe's

locking in before 11:59 PM deadline

Question assigned to the following page: 1

10. **Describe your crowd-facing user interface. This can be a short caption for the screenshot. Alternatively, if you put a lot of effort into the interface design, you can give a longer explanation of what you did.**

We implemented a whole website to support the crowd-facing user interface. This includes the home page, voting interface, playlist result page, and so on. We first studied how users would usually vote for a playlist if it was on a Google form. Then, we turned that idea into a website so that the interface allows users to submit their votes at a much greater ease. As a result, we were able to develop a full-working website with a very user-friendly frontend interface.

Ethics

1. **Should my application exist at all?**
    a. While it may not be strictly necessary for the application to exist, its presence can be genuinely beneficial to users, especially those seeking a convenient tool for curating or discovering music. By providing a more accessible and organized way to find relevant songs, the application can help save time and streamline the listening experience. Thus, even though the concept isn't indispensable, it delivers enough value through improved user convenience and enjoyment to justify its existence.
2. **Does this task potentially expose workers to harm (for example, content moderation)? What effect can it have on them?**
    a. The likelihood of harm to workers is minimal, as the content they review will generally be typical, everyday music tracks without severe or disturbing themes. However, there is a small possibility of encountering songs with explicit lyrics that could be uncomfortable or triggering for some. To address this, introducing a content filtering feature in the future could help prevent workers from having to process R-rated material. By proactively offering tools to filter out such content, we can further reduce the risk of negative effects on their mental well-being and ensure a safer, more positive working environment.
3. **Are you fairly compensating the workers for their time?**
    a. Currently, no direct monetary compensation is provided. Instead, the interface is designed to be gamified, encouraging participation through a points or level-based system rather than traditional pay. Although this approach deviates from conventional compensation models, workers benefit indirectly: the more they contribute, the more accurate the results become, and the richer their access to well-curated playlists. Over time, they gain value in the form of improved user experience, convenient music discovery, and a sense of community involvement. This non-monetary form of compensation may appeal to users who enjoy contributing to and benefiting from a collective effort, even if they do not receive financial payment.
4. **If you are creating a dataset for machine learning:**
    a. We did not create a dataset for ML.

Question assigned to the following page: 1

5. **Is your evaluation sound? Do the conclusions you reach stand up to scientific scrutiny?**
    a. The evaluation methods employed do not aim to meet the strict standards of scientific research or academic rigor. Instead, the primary measure of success is user satisfaction and enjoyment. While some techniques or metrics might be used to improve the user experience, the ultimate objective is not to produce statistically significant results or publish scientific findings. The conclusions drawn are therefore more aligned with practical, real-world usability and subjective user feedback rather than standing up to formal scientific scrutiny.

Skills

1. **Do your crowd workers need specialized skills?**
    a. Our crowd workers need the ability to hear. They need to listen and analyze music, extract emotional context from the song, and decide whether a given feeling matches the song.
2. **What sort of skills do they need?**
    a. The crowd workers need musical sensitivity, emotional intelligence, analytical thinking skills, and attention to detail. Workers need the ability to perceive and interpret musical elements like rhythm, melody, and harmony to determine emotional impact. They need skill in associating songs with specific feelings or moods, as well as capacity to compare songs and evaluate whether they fit a given vibe or theme. Finally, they need the focus to discern subtle differences between songs and accurately match them to appropriate moods or themes.
3. **Do the skills of individual workers vary widely?**
    a. Skills of individual workers vary based on musical knowledge, cultural background, and experiences. Some workers may have formal training in music, while others rely on personal experience and exposure. Workers who frequently engage with music may better identify and classify its emotional undertones. Depending on cultural context, musical interpretation can differ significantly.
4. **If skills vary widely, what factors cause one person to be better than another?**
    a. Workers familiar with a wide range of musical styles might be able to make more nuanced judgments. Someone who's favorite genre is rock music might be really good at filtering rock songs but really bad at filtering other types of songs. Also, some individuals may have a more stable sense of how music correlates to emotions.
5. **Did you analyze the skills of the crowd?**
    a. Since the results are very subjective, it was difficult to analyze the skills of the crowd. The way we analyzed the skills of the crowd was by verifying against an AI model.
6. **If you analyzed skills, what analysis did you perform? How did you analyze their skills? What questions did you investigate? Did you look at the quality of their results? Did you analyze the time it took individuals to complete the task? What conclusions did you reach?**

Question assigned to the following page: 1

   a. First, we collected a crowd-chosen playlist of songs based on a specific vibe. Then we asked AI to create a playlist of songs based on the same vibe. Since the AI (ChatGPT) was trained on a large amount of popular data and music trends, it is likely that its results show a general consensus. This way, we can analyze how well an individual performs against the playlist. We looked at the quality of other user's votes by judging them ourselves and aggregating the team's judgements to form an average.
   b. We came to an interesting conclusion about the skills of our crowd. Agreement rates were higher for common emotions like "getting over a three week long situationship" but lower for nuanced feelings / difficult vibes like "walking 30 minutes to trader joes". It seems that the crowd was better at identifying more common emotions, indicative of the human psyche. And it also seems that everyone can interpret music very differently.

Quality Control

1. **Is the quality of what the crowd gives you a concern?**
   a. Initially, we definitely had some doubts about the crowd's quality. The question our product tries to solve (which songs belong to which playlists) is a very subjective one. There was also the potential for miscommunication (a playlist titled "songs to get over a situationship" might not be understood by everybody).
2. **How do you ensure the quality that the crowd provides?**
   a. We implemented two quality control mechanisms: worker quality control, and vote validation. We identify and filter out low-quality workers by comparing their votes against the majority consensus. Workers with agreement rates below 30% are excluded. (This threshold can be adjusted.) Then, only songs that receive positive votes above a 50% threshold are accepted, and each song is assigned a confidence score based on the ratio of positive votes. (Presenting these confidence scores to the user allows them to further filter recommendations as needed.)
3. **If quality is a concern, then what did you do for quality control? If it is not a concern, then what about the design of your system obviates the need for explicit QC? This answer should be substantial (several paragraphs long).**
   a. (This question was partially covered by the previous answer.) In short, we had two quality control mechanisms in place. After collecting votes and logging worker IDs, we first filtered out low-quality workers by comparing their votes against the majority consensus. Workers with agreement rates below 30% are excluded. (This threshold can be adjusted programmatically, and we never overwrite raw data. We chose this design to keep some flexibility: as we get more data and the quality distribution changes, we can go back and tweak our algorithm.) In the second stage, we decide the membership of each song in a playlist. Only songs that receive positive votes above a 50% threshold are accepted, and each song is assigned a confidence score based on

Question assigned to the following page: 1

the ratio of positive votes. We also plan on presenting these confidence scores to the end user, which will allow them to further filter out recommendations.

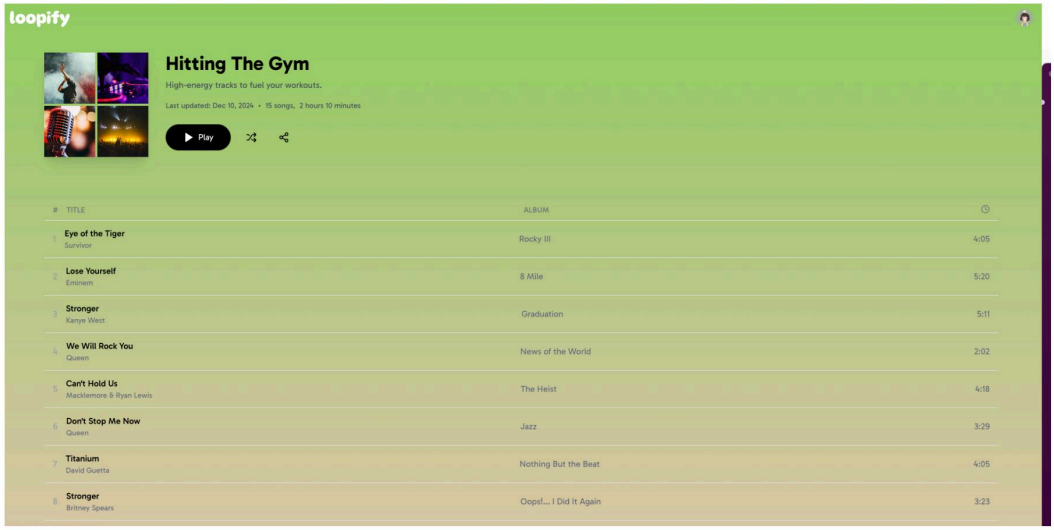
4. **Did you analyze the quality of what you got back? For instance, did you compare the quality of results against a gold standard? Did you compare different QC strategies?**
    a. It was difficult to apply some sort of objective function on the data we collected, since the question we were trying to answer (does a song belong to a playlist or not) is pretty subjective. To ensure the data was reasonable, we sampled randomly from the votes we collected and had team members review them. (One challenge with this strategy was that not all our team members knew the songs they were reviewing!)
    b. Another sanity check we considered was having an LLM review votes. In this plan, a closed-source LLM (like GPT-4o) would be presented with a vote, and asked whether it agreed with the decision or not. We really liked the idea behind this, since it's reasonable to expect that the LLM has consumed a lot of music reviews in its training data. After some experimentation, we found this sanity check to be quite reasonable. However, some problems arose when songs that weren't in the training data (such as songs released after GPT-4o's data cutoff) were presented.
5. **What analysis did you perform on quality?**
    a. (Refer to the previous question for a more in-depth discussion!) We performed a couple sanity checks on the data we got back. Overall, applying a rigorous analysis was somewhat difficult given the subjective sense of song recommendations. There's no widely-accepted heuristic to determine whether a song belongs to a playlist or not (especially when the playlists are hyper-specific, like those on our platform). As a substitute, we applied some mass data verification techniques (like random sampling with a gold standard, or automated data review with an LLM).waa
6. **What questions did you investigate? What conclusions did you reach?**
    a. Our primary question was to figure out whether crowdsourced recommendations made sense. We were worried that several factors (like a language barrier, subjective taste profiles, and miscommunication) may introduce structural issues with the data. As described in the previous answers, we applied some data verification techniques to make sure the results were up to par.
    b. Overall, we found crowdsourced recommendations to be pretty effective, especially once quality control mechanisms were added! Filtering out low quality workers and setting an acceptance threshold kept our results from spiraling out of control. (As an example, these quality control mechanisms filtered down the pool of songs in a playlist by 65%.)

Aggregation

1. **How do you aggregate the results from the crowd?**
    a. We aggregate results from the crowd by storing individual votes in a MongoDB database. These votes contain a couple pieces of information: the song in question, the playlist in question, and whether the song belongs to the playlist or not.

Question assigned to the following page: 1

   b. In our API, we expose a route that allows users to view the songs "accepted" to a user-specified playlist. The system then aggregates crowd results by iterating through all votes, identifying ones relevant to the specified playlist, and then doing some quality control on the set of songs. (Refer to the previous section for further discussion on the quality control mechanisms.)

2. **Did you analyze the aggregated results?**
   a. Yes, but we weren't too concerned about the aggregation step of our project. (In our opinion) the UX for inputting voting decisions was quite straightforward, and we implemented a lot of error checking both on the frontend and the backend. By the time the information was written to our database, we were sure that it was structurally correct.
3. **What analysis did you perform on the aggregated results? What questions did you investigate? Did you compare aggregated responses against individual responses? What conclusions did you reach?**
   a. We decided to sanity check our aggregation method by looking at all votes for a particular song. Given multiple playlists and our chosen song, we expected to see relatively similar voting patterns across a large distribution. (For example, *Despacito* probably shouldn't be in a romantic playlist.) Although our analysis lacked some rigor, we found that most of the results lined up with what we were expecting.
4. **Did you create a user interface for the end users to see the aggregated results? If yes, please include a screenshot of the user interface for the end user in your final report. You can include multiple screenshots, if you want.**
   a. Yes. After users vote on songs, they can also see which songs have been "accepted" to which playlists. On our backend, this functions as two routes: one to fetch all playlists, and another to fetch which songs belong to a certain playlist.
   b. 

5. **Describe what your end user sees in this interface. This can be a short caption for the screenshot. Alternatively, if you put a lot of effort into the interface design, you can give a longer explanation of what you did.**

Question assigned to the following page: 1

a. The end user sees an aggregated list of the top songs, very similar to a playlist. The idea is that they can scroll through the playlist and find songs related to that specific vibe / playlist name and discover new songs.
b. We put a lot of effort into the interface design, first going through inspiration photos and then drafting up the design mockup in Figma. Here is the link to our Figma link, which we then coded in frontend: https://www.figma.com/design/a7JFwhZObAKBDDq1o6gX9g/LOOPIFY?node-id=0-1&node-type=canvas&t=lu5SipEH53GStpwZ-0

Scaling Up

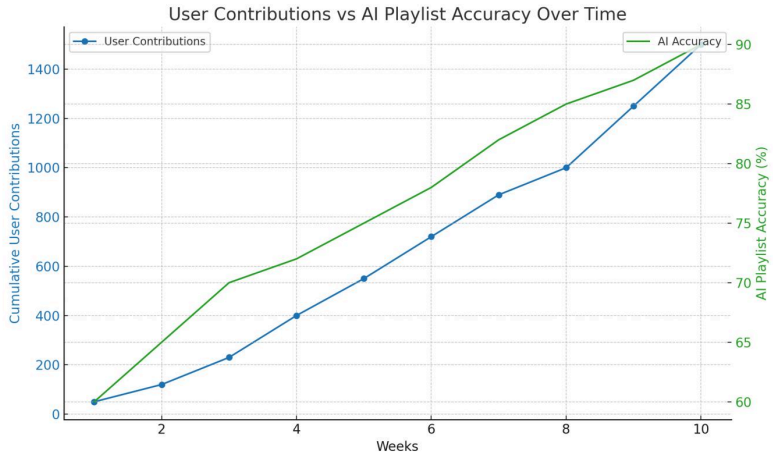
1. **What is the scale of the problem that you are trying to solve?**
    a. The problem is global, as millions of music listeners struggle to discover songs and playlists that perfectly match specific moods or themes. It is not just limited to one demographic or even language, as anyone can contribute.
2. **Would your project benefit if you could get contributions from thousands of people?**
    a. Yes of course, contributions from thousands of people would enhance the quality and diversity of playlists, improve AI recommendations, and ensure better vibe accuracy through crowd consensus.
3. **If it would benefit from a huge crowd, how would it benefit?**
    a. A large crowd would bring in varied perspectives, making playlists more inclusive and relatable to a broader audience. More user votes and tags improve the precision of AI-generated recommendations and validate song-vibe matches. Also, the scale would definitely bring in more community engagement, with users collaborating to create high-quality playlists that reflect shared preferences.
4. **What challenges would scaling to a large crowd introduce?**
    a. Scaling could introduce challenges like maintaining quality control, preventing misuse of the voting system, and ensuring the platform remains user-friendly and engaging.
5. **Did you perform an analysis about how to scale up your project? For instance, a cost analysis?**
    a. We considered challenges like the cost of maintaining servers to handle large-scale data, implementing AI models for efficient aggregation, and incentivizing user engagement at scale.
6. **What analysis did you perform on the scaling up?**
    a. We analyzed the impact of increasing users on server capacity, database performance, and data aggregation systems. We also evaluated user engagement strategies like gamification and moderation tools. However, because of the short time limit, we were not able to implement scaling up features although we would love to in the future.
7. **What questions did you investigate? What conclusions did you reach?**
    a. We explored how to maintain quality control with a large user base and concluded that community voting and AI-assisted moderation are critical. We also assessed the

Question assigned to the following page: 1

trade-offs between real-time data processing costs and user experience. Some conclusions that we reached were that AI cannot offer the same value that human music taste input can, and people prefer authentic user recommendations over AI generated ones.

Project Analysis

1. **Did your project work? How do you know? Analyze some results, discuss some positive outcomes of your project.**
    a. Yes, our project worked in its initial implementation. We know this because users successfully tagged songs, voted on playlists, and received AI-generated recommendations that closely matched the intended vibes. Positive outcomes include increased user engagement, where participants reported enjoying the collaborative playlist-building process, and a noticeable improvement in AI-generated playlists as more crowd data was collected and analyzed.
2. Do you have a graph analyzing your project? If you have a graph analyzing your project, include the graph here.



    a.
    b. We touched on this in our Milestone 4 as well, but this graph shows the relationship between cumulative user contributions and AI playlist accuracy over 10 days. The blue line represents the growth in user contributions, while the green line illustrates the corresponding improvement in AI playlist accuracy. (the weeks is a typo)
3. **What were the biggest challenges that you had to deal with?**
    a. The biggest challenges were ensuring quality control in user contributions, preventing spam or irrelevant votes, and achieving sufficient engagement to train the AI models effectively. Additionally, balancing computational costs for AI processing and real-time data aggregation were definitely a technical challenge.
4. **Were there major changes between what you originally proposed and your final product?**

Question assigned to the following page: 1

    a. Yes, there were some changes. Initially, we planned to rely heavily on AI for playlist generation, but we realized early on that crowd input was more effective for accurately matching niche vibes. We shifted focus to making the platform more collaborative, with features like voting and playlist sharing taking precedence.
5. **If so, what changed between your original plan and your final product?**
    a. The original plan emphasized automated AI-generated playlists, but the final product became more crowd-driven. Features like the skip button and pulling songs from top 1000 Spotify songs (to ensure people would at least likely know them) were introduced later to address engagement challenges that we hadn't initially anticipated.
6. **What are some limitations of your product? If yours is an engineering-heavy project, what would you need to overcome in order to scale (cost/incentives/QC…)? If yours was a scientific study, what are some sources of error that may have been introduced by your method.**
    a. Some limitations of our product include scaling issues, as maintaining quality control becomes more challenging with larger crowds, requiring advanced moderation tools and AI filters. Additionally, the platform faces a cold start problem, where new playlists or vibes may struggle to gain traction without an active initial user base to contribute and vote. There is also a risk of bias in contributions, with popular songs or genres potentially dominating playlists and overshadowing lesser-known tracks that align with a specific vibe. To address these challenges and scale effectively, we would need to optimize computational costs for AI processing, implement robust spam detection systems, and incentivize user participation through rewards or strategic partnerships.
7. **Did your results deviate from what you would expect from previous work or what you learned in the class?**
    a. Yes, our results partially deviated. We expected the AI model to handle vibe matching effectively from the start, but it underperformed without substantial crowd contributions. Crowd consensus proved to be a stronger driver of playlist quality than initially anticipated.
8. If **your results deviated, why might that be?**
    a. The deviation occurred because AI models require significant training data to produce accurate results. Early-stage user data was insufficient for the AI to recognize nuanced vibes, so crowd input became the primary driver of playlist quality during the initial phase. This highlighted the need for a hybrid approach where AI and crowd contributions complement each other over time.

Technical Challenges

1. **Did your project require a substantial technical component? Did it require substantial software engineering? Did you need to learn a new language or API?**
    a. Yes, our project was very tech-heavy. We used React Native for the frontend, Vercel and MongoDB for backend services, and GitHub for collaboration and version

Question assigned to the following page: 1

control. While some of our team members were familiar with these tools, we had to learn and implement several new APIs and frameworks for integrating the database and deploying the platform on Vercel.

b. Github repository: https://github.com/acao22/loopify

c. Vercel deployment link: https://loopify-sand.vercel.app/

2. **If the project required a substantial technical component, describe the largest technical challenge you faced.**
    a. Our largest technical challenge was definitely synchronizing real-time user interactions with the database. For example, when users tagged songs, voted on playlists, or provided feedback, these updates had to be reflected in the app immediately while maintaining data consistency across sessions. This required implementing real-time database operations and ensuring scalability to handle potential large-scale user activity.
3. **How did you overcome this challenge? What new tools or skills were required? Feel free to nerd out a bit, to help us understand the amount of work that was required.**
    a. Another significant challenge was securely managing the Spotify API secret keys, which are required for accessing Spotify's music data. Handling these credentials safely while integrating the API into our app posed a risk of accidental exposure, particularly during development and deployment stages. To address this, we used environment variables to store the API keys securely and configured Vercel to manage them in production. This ensured that the keys were not exposed in our codebase or version control.
    b. Also, we learned best practices for API integration, such as setting up rate-limiting mechanisms to avoid exceeding Spotify's API usage quotas and caching responses to reduce unnecessary API calls. By combining these strategies with tools like React Query for frontend state management and MongoDB Atlas for backend operations, we were able to maintain both security and efficiency. These steps required a deep understanding of API security protocols and modern deployment practices.
4. **Do you have any screenshots or flow diagrams to illustrate the technical component you described? If so, include the graph here.**