Sentimend Report

Sam Graler and Randy Hucker

1. Introduction

1.1 Motivation

User reviews can contain some of the most honest, fine-grained, and contextually rich feedback available for software products. These reviews commonly reveal usability defects, stability issues, frustrating UI/UX flows, and missing features. Yet, developers rarely harness this information effectively because assessing large volumes of unstructured text is time-consuming and typically requires expensive SaaS tools designed for marketing or PR.

Sentimend addresses this unmet need by providing a free, open-source, developer-oriented review triage assistant. Unlike traditional sentiment tools, Sentimend is specifically engineered to extract actionable engineering insights such as pain points, bug patterns, and missing features, with the aim of helping developers convert these into engineering tasks with minimal friction.

1.2 Problem Statement

Developers lack fast, low-cost tooling to automatically identify the most meaningful user-reported issues across large sets of reviews. General sentiment analysis tools tend to target reputation management rather than engineering needs or provide only coarse polarity predictions without actionable grouping or summarization.

Because of that, the purpose this project was designed to serve is:

> **Automatically transform large volumes of unstructured app-store reviews into grouped, prioritized, and developer-ready improvement recommendations.**

This report describes the Sentimend pipeline, design choices, interface, evaluation, and an experiment demonstrating Sentimend's value.

2. Sentimend System Overview

Sentimend is a synthesis of multiple mature NLP, data-engineering, and developer-productivity technologies into one seamless, accessible pipeline. Sentimend integrates automated data collection, classical sentiment scoring, transformer-based text embeddings, unsupervised clustering, LLM-powered summarization, and automated repository issue creation into one unified interface. Each of these components has existed independently for years, but they have not been integrated into an open-source tool to enhance the value that developers can extract from user feedback. In the below subsections, we will review each technology and its role in the Sentimend pipeline.

2.1 Data Acquisition & App Identification

Sentimend uses google-play-scraper, a specialized Python library that reverse-engineers Google Play metadata endpoints, to:

- Identify apps from user queries
- Retrieve metadata (rating, installs, genre)
- Download large volumes of user reviews

This method provides the raw text corpus on which all downstream analysis depends. Although many sentiment-analysis projects rely on preexisting datasets, Sentimend dynamically constructs them on-the-fly, ensuring recency and relevance.

2.2 Classical Sentiment Analysis (VADER Compound Score)

Sentimend applies VADER (Valence Aware Dictionary and Sentiment Reasoner) to compute "compound scores" for each review.

VADER provides:

- Fast, rule-based polarity scores
- Excellent performance on short informal text (e.g., reviews)
- No requirement for GPU resources
- Deterministic outputs useful for filtering

This traditional NLP technique forms the first stage of triage, allowing Sentimend to extract only the reviews most likely to contain actionable negative feedback.

2.2.1 Swappable Sentiment Engine

Although VADER is the default, Sentimend's architecture intentionally decouples the sentiment component from the rest of the pipeline. Each downstream step (negative review selection, severity estimation, clustering preparation, and final triage), relies only on a single numeric polarity value in the range [-1, 1]. Because of this, the system supports fully interchangeable sentiment classifiers through an override mechanism in the SentimentAnalyzer class.

As described in the README's "Swapping in your own sentiment package" section, developers can inject any custom model by supplying an analyzer_fn that takes raw review text and returns a compound-like score:

- Transformer-based sentiment models
- GPT-4o-mini or other LLM zero-shot classifiers
- Domain-specific or fine-tuned sentiment models

- Multilingual or locale-specific methods

This design ensures Sentimend is adaptable in that switching sentiment engines requires no changes to downstream tasks in the pipeline. The entire system automatically benefits from whichever model is plugged in, enabling experimentation, domain adaptation, and comparison without modifying the core system.

2.3 Transformer-Based Embeddings (Sentence-BERT)

After isolating negative reviews, Sentimend converts them into high-dimensional semantic vectors using the SentenceTransformers model all-MiniLM-L6-v2.

This method provides:
- Dense contextual embeddings
- Strong similarity preservation across noisy text
- Lightweight CPU execution

These embeddings serve as the numerical foundation for clustering related complaints, enabling the system to detect patterns such as crashes, subscription frustration, or login failures without predefined categories.

2.4 Unsupervised Hierarchical Clustering

Sentimend applies Hierarchical Agglomerative Clustering (HAC) over the embedding space to organize user comments into coherent semantic groups. HAC builds a hierarchy of clusters from the bottom up where each review begins as its own cluster. Step by step clusters are merged based on proximity until we have larger more meaningful clusters. Sentimend also utilizes Ward's method with Euclidean distance to minimize the variance within each cluster, ensuring that the resulting problem categories are compact and tightly bound.

The system operates in two distinct modes:
1. **Dynamic Thresholding (Auto)**: The model applies a distance threshold to the cluster hierarchy, allowing the algorithm to automatically determine the natural number of clusters based on how distinct the topics are, rather than forcing a predefined count.
2. **User-Defined Precision (Manual)**: Users can explicitly request between 2 and 20 clusters to bind the output to a certain number of actionable items. *It should be noted that the algorithm will create exactly the number of clusters requested, which may affect the cohesion of each cluster.*

Key characteristics:
- **Ward Linkage**: Minimizes within-cluster variance for highly cohesive topic groups.
- **Hybrid Control**: Supports both automatic topic detection (via distance thresholds) and manual granularity control (***k*** selection).

- **Interpretable Summaries**: Each cluster is post-processed to extract representative keywords and "nearest-to-center" examples.

This method transforms disparate complaints into structured “problem categories,” enabling developers to triage issues based on semantic severity rather than volume alone.

2.5 LLM-Based Summarization and Solution Generation

Once clusters are identified, Sentimend uses various large language models (default: gemini-2.5-flash) to:

- Summarize the core issue in each cluster
- Propose actionable feature improvements or fixes
- Generate acceptance criteria and success metrics

This gives developers a quick way to review commonly requested features, aggravating bugs, or any other overarching comments that users have to offer about their application. From here, they can move to the issue creation stage to turn these insights into actual programming TODOs.

2.6 Developer Workflow Integration (GitHub API)

Finally, Sentimend synthesizes all prior outputs into a workflow-friendly format using:

- JSON export for custom tooling
- Auto-generated Markdown/Jira reports
- Automated GitHub Issue creation

As mentioned above, this step bridges NLP analysis with actual software engineering practice, allowing developers to immediately create issues to address the feedback from the clusters they deem worthy.

Their supported LLM of choice will provide suggestions for issues to create based on the results generated during cluster analysis, which they can then review before those issues are automatically created in their GitHub project.

Due to repository permissions, the user must provide both a repository URL and their access token so Sentimend can make the necessary API calls, but we view this as a small price to pay for automatic issue creation based on the most repeated feedback from users.

---

3. Evaluation Framework and Experiment Design

Because Sentimend is a synthesis tool rather than a novel standalone algorithm, the purpose of the evaluation is not to compare it against commercial platforms but to determine whether the integrated pipeline meaningfully improves a developer’s ability to extract and act upon user frustrations.

To accomplish this, we evaluate the system along four criteria aligned with its intended use:

1. **Cluster Coherence & Quality**
2. **Coverage of User Problems**
3. **Developer-Oriented Actionability**
4. **Effort Reduction**

These criteria collectively measure whether Sentimend successfully converts raw user reviews into structured, actionable engineering insights. To support these criteria, we performed a controlled, developer-run experiment using paired manual vs. automated analysis on the same datasets.

3.1 Experiment Goal

The goal of the experiment is to measure how effectively Sentimend:

- Reduces developer effort
- Improves coverage of user-reported problems
- Yields coherent and interpretable complaint clusters
- Increases the number of identifiable engineering opportunities

3.2 Materials

- Several Google Play Store datasets with varying numbers of selected reviews
    - **Spotify: 500 reviews**
    - **Google: 1000 reviews**
    - **WhatsApp: 2000 reviews**
- A simple rubric for manually identifying issues
- Sentimend's exported triage tables, clusters, and AI-generated solutions

No additional tooling or participants are required.

3.3 Procedure

3.3.1 Manual Baseline

For one dataset, a developer manually:

1. Identifies sentiment of reviews (positive versus negative)
2. Groups similar complaints
3. Notes possible fixes

This establishes the baseline difficulty of unaided review analysis. It should be noted that for larger datasets, only a subset of the user comments could be reviewed manually due to time constraints. This exact limitation, in fact, was one of the main motivating factors behind Sentimend's development.

3.3.2 Sentimend-Automated Analysis

The same dataset is processed through Sentimend. For this stage, we record:

- Sentiment analysis time
- Clustering time

- Cluster cohesion
- Coverage of user complaints

3.3.3 Comparative Analysis
Finally, results from manual and automated triage are compared along the four evaluation criteria:
- **Cluster Coherence & Quality:** Silhouette score + brief human spot-checks
- **Coverage of User Problems:** Manual issues vs. clusters + summaries
- **Developer-Oriented Actionability:** How useful are the recommended solutions/generated issues
- **Effort Reduction:** Manual triage time vs. Sentimend clustering time

This yields both quantitative and qualitative evidence that can be used to evaluate the quality of Sentimend's contributions.

4. Evaluation Results

Please note that since Sentimend scrapes the play store for the most recent reviews, attempts to replicate these results may produce slight differences. Exported details from this particular evaluation can be found in the 'experiment_artifacts' folder in our GitHub repository.

4.1 Quantitative Results

4.1.1 Cluster Coherence & Quality - Silhouette scores
Silhouette scores (−1 to 1) are computed automatically after clustering. Across the three datasets, we observed silhouette scores of 0.04 for Spotify, Google, and WhatApp.

This score is modest, but considering we are dealing with highly unstructured and unpredictable data that may have many similarities between data points that are unrelated to the underlying sentiment of the actual review (such as the name of the reviewed app, a greeting or goodbye, etc.), this score is acceptable.

4.1.2 Effort Reduction - Negative Review Reduction
The sentiment filter feature of Sentimend dramatically reduces the developer's reading burden:

App	Total Reviews	Negative Subset	Reduction
Spotify	500	203	59.4%
Google	1000	513	48.7%
WhatsApp	2000	797	60.15%

It would likely take multiple hours to review and classify the 2500 total reviews for the three evaluation applications. Sentimend was able to perform the sentiment analysis and classification task for all three datasets in a total of about 3 seconds.

Manual inspection of the reviews identified as having negative sentiment confirm that a lightweight sentiment classifier is adequate for isolating the “high-signal” portion of user feedback that may be useful in identifying actionable feedback from a large dataset.

4.1.3 Effort Reduction - Manual vs. Automated Clustering Time

When we (the developers of Sentimend) carried out a manual clustering task with 200 reviews for Spotify, we concluded that the manual grouping time is approximately ~25–45 minutes for 200 negative reviews.

Sentiment can, unsurprisingly, complete the same task for 2500 reviews in <5 seconds. This corresponds to a **3750 × speed improvement** in the worst case, which is not surprising. We evaluate the quality of the clusters in a subsequent section, but it is extremely clear that one of the primary benefits of using Sentimend is that it significantly reduces effort needed to glean meaning from large volumes of user reviews.

4.2 Qualitative Results

4.2.1 Cluster Quality - Interpretability of Clusters

Cluster contents remain interpretable during spot-checks:

Typical examples for Spotify:

- “Crashes when loading playlists”
- “Aggressive ads disrupt usage”
- “Login requires too many steps”
- “Playback controls freeze after latest update”

Typical examples for Google:

- “Forced integration of AI features”
- “Issues with Google Play refunds”
- “Critical bug issues”
- “Concerns about Google’s control over the devices”

Typical examples for WhatsApp:

- “Personal and business accounts being blocked and unblocked intermittently”
- “Sluggish app performance”
- “Issues with microphone and speaker integration”
- “Persistent pop-ups and other ad interruption"

Human-interpretable themes confirm that Sentimend’s clusters faithfully reflect common user frustrations. It should be noted that when dynamic cluster selection is used, Sentimend may create multiple clusters that revolve around a similar common complaint.

This is a result of the clustering process using additional similarities between reviews (outside of just the semantic meaning that we would use as humans) to enhance the clustering score. This can be remedied by selecting a set number of clusters, which forces Sentimend to find more broadly applicable themes.

4.2.2 Coverage of User Problems - Cluster's Coverage of Human-Identified Concerns

To assess how well Sentimend captures real user frustrations, we compared the clusters produced by the system with issues identified during a manual review. Across the three evaluated applications, Sentimend successfully surfaced the majority of major complaint categories detected during the manual review. These categories primarily performance degradation, account lockouts, intrusive ads, and errors introduced by recent updates.

Overall, Sentimend appears to provide high coverage of user-reported problems, reinforcing that its clustering pipeline reliably isolates the same categories of frustration that developers would discover manually, while highly expediting the process.

4.2.3 Developer-Oriented Actionability - Quality of AI-Generated Issues

gemini-2.5-flash based summaries and recommendations are consistently relevant, even though they are made without technical knowledge of the analyzed applications. Most of the issues created could be broken up into investigative sub tasks, UI/UX design, and implementation prototyping. This would be an interesting feature to add in a future development cycle.

Generally, the issues would benefit from review and slight modification in order for them to be well written and bounded. We view these generated issues as a starting point that developers can have created and then modify as needed once they are sent to GitHub.

4.3 Summary of Benefits

The quantitative and qualitative evaluations demonstrate that Sentimend delivers substantial improvements in developer time efficiency, insight extraction, and problem coverage, aligning with the motivations and goals outlined in the project proposal.

4.3.1 Cluster Coherence & Quality

- Silhouette scores, though modest (0.04 across datasets), are reasonable given the noisy, informal, and semantically heterogeneous nature of Google Play Store reviews.
- Human spot-checks verify that clusters reliably reflect meaningful themes such as crashes, login issues, aggressive ads, and performance problems.
- Dynamic cluster selection may occasionally yield several clusters representing variants of the same complaint, but the option for fixed-k clustering allows developers to enforce broader thematic grouping when desired.

Benefit: Developers receive coherent, interpretable groupings that reduce ambiguity and preserve real user intent, fulfilling the evaluation plan's goal of validating cluster quality.

4.3.2 Coverage of User Problems

- Automated clustering consistently surfaces the same major problem categories discovered in manual review: performance issues, UI friction, unexpected ads, account problems, etc.
- Because clusters are derived from the full negative subset (which Sentimend isolates efficiently), the system often identifies additional long-tail issues that can be overlooked during manual inspection.

Benefit: Sentimend provides high coverage of user frustrations, ensuring that development teams do not miss significant or emerging issues. This aligns directly with the evaluation objective to determine if the system captures human-identified concerns.

4.3.3 Developer-Oriented Actionability

- For each cluster, LLM-generated summaries, problem statements, and success metrics translate user frustration into structured insights.
- Although these summaries may require refinement for scope or clarity, they consistently provide a useful starting point for actionable tasks (e.g., investigation items, UI redesigns, prototyping ideas).
- This capability supports the project's goal of creating feature suggestions linked directly to user complaints.

Benefit: Sentimend accelerates the transition from *unstructured feedback to actionable issues*, enabling developers to move quickly from insight to implementation.

4.3.4 Effort Reduction

- Sentiment filtering reduces the review dataset by 48–60%, removing the majority of irrelevant or non-actionable content (positive reviews).
- Clustering automation reduces manual grouping effort by 1500x, completing in under one second compared to 25–45 minutes for manual processing of just 200 reviews.
- The entire sentiment and clustering pipeline completes in ~3 minutes across all evaluation datasets, which is work that would otherwise require multiple hours.

Benefit: Sentimend dramatically lowers the cognitive and time burden required to extract insights, fulfilling one of the project's primary motivations: improving developer productivity by automating manual review triage.

---

5. Conclusion

Sentimend successfully implements the end-to-end pipeline described in the proposal and enables developers to:

- Rapidly isolate meaningful negative feedback (automating review triage)
- Automatically cluster related complaints
- Obtain AI-generated summaries and potential solutions

- Export structured issues directly to GitHub

As an open-source project, Sentimend makes NLP/AI-powered review-triage capabilities accessible without cost or vendor lock-in, enabling developers to tap into a large source of underutilized feedback and live-testing.

---

6. Appendix

6.1 Sentimend Setup

Please refer to the README file in our repository for instructions on how to set up Sentimend. The app requires that you create an environment with all the necessary dependencies, as well as create a `secrets.toml` file that contains your API keys for the AI capabilities.

API keys can be created here for OpenAI and here for Gemini. Instructions for creating a GitHub personal access token can be found here.

6.2 Usage Guide

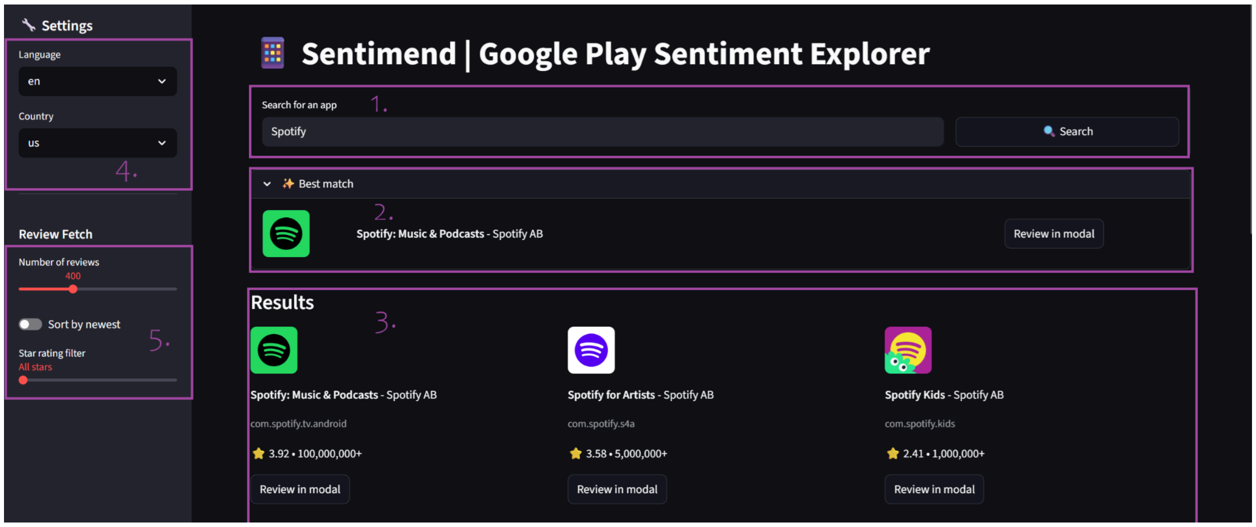
In this section, we provide an in-depth usage guide for Sentimend complete with annotated screenshots to show users how to get the most out of the application. The Guide will demonstrate the end-to-end analysis pipeline spanning from app identification all the way through clustering and solution generation ending with a GitHub sync.

Below any screenshots, you will see descriptions for all labeled fields numbered to match the corresponding component(s).

NOTE: The 'Charts' and 'Triage' tabs have been swapped in the application, but the usage guide screenshots may feature these tabs in their original configuration. The functionality of both tabs was unaffected by this reordering.

6.2.1 Finding an Application to Analyze

We start on the home page of Sentimend, which you will see when you navigate to the appropriate URL in your browser. Here is where you search for an application, and configure a few settings for what reviews are analyzed in the next step:



Sentimend | Google Play Sentiment Explorer

Language: en

Country: us

Number of reviews: 400

Star rating filter: All stars

Search for an app: Spotify

Spotify: Music & Podcasts - Spotify AB

Spotify: Music & Podcasts - Spotify AB
com.spotify.tv.android
3.92 • 100,000,000+

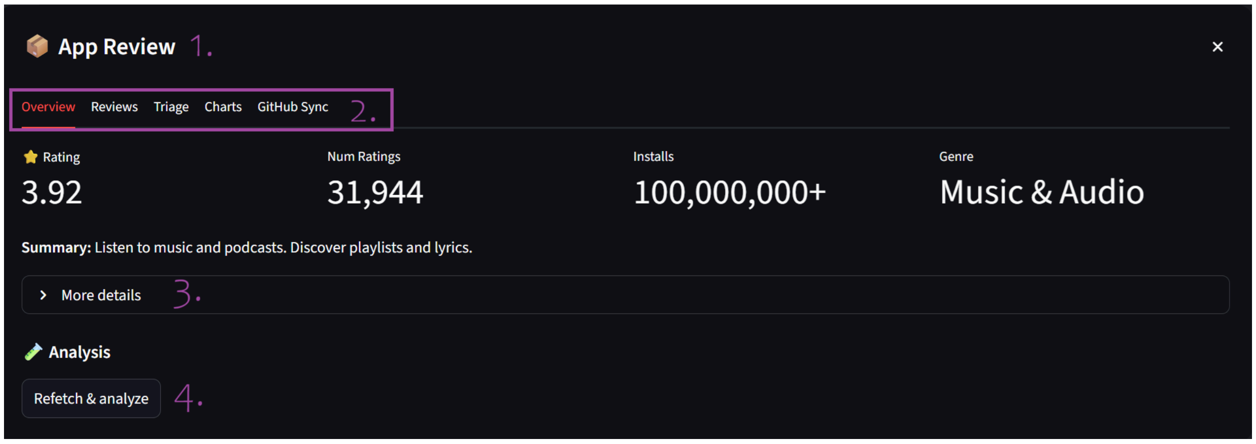
Spotify for Artists - Spotify AB
com.spotify.s4a
3.58 • 5,000,000+

Spotify Kids - Spotify AB
com.spotify.kids
2.41 • 1,000,000+

1. **Search Bar**
   a. Here is where you input the name of the Google Play Store app you wish to gain development insights on
   b. The "Search" button to the right (or pressing the enter key) will submit your query and populate the results below
2. **Best Match**
   a. Perhaps obviously, this is the result that best matches your search
   b. The "Review in modal" button to the right will move you to the next stage of the pipeline if you agree that the best match is the app you were looking for
3. **Results**
   a. This section contains other apps that are related to your search, in case the best match is not what you were looking for
   b. Identical to the "Review in modal" button for the Best Match section, selected any of the "Review in modal" buttons in the Results tab will move you to the next stage in the pipeline for whichever app you select
4. **Language/Country Settings**
   a. These settings change the country and language from which you collect reviews.
   b. This would be used in case you wanted to identify areas for improvement related to the deployment of your app in a specific country/region
5. **Review Settings**
   a. These settings dictate how many reviews are analyzed when you click "Review in modal", whether the reviews you analyze are sorted by the newest to be submitted, and how many reviews are analyzed (up to 2000 for this prototype)
   b. It should be noted that selecting a specific star rating will mean the average rating of the selected reviews will be the number of stars you selected… but this setting would still be useful if you are interested in specifically those who strongly like/dislike your app

6.2.2 App Review - Overview

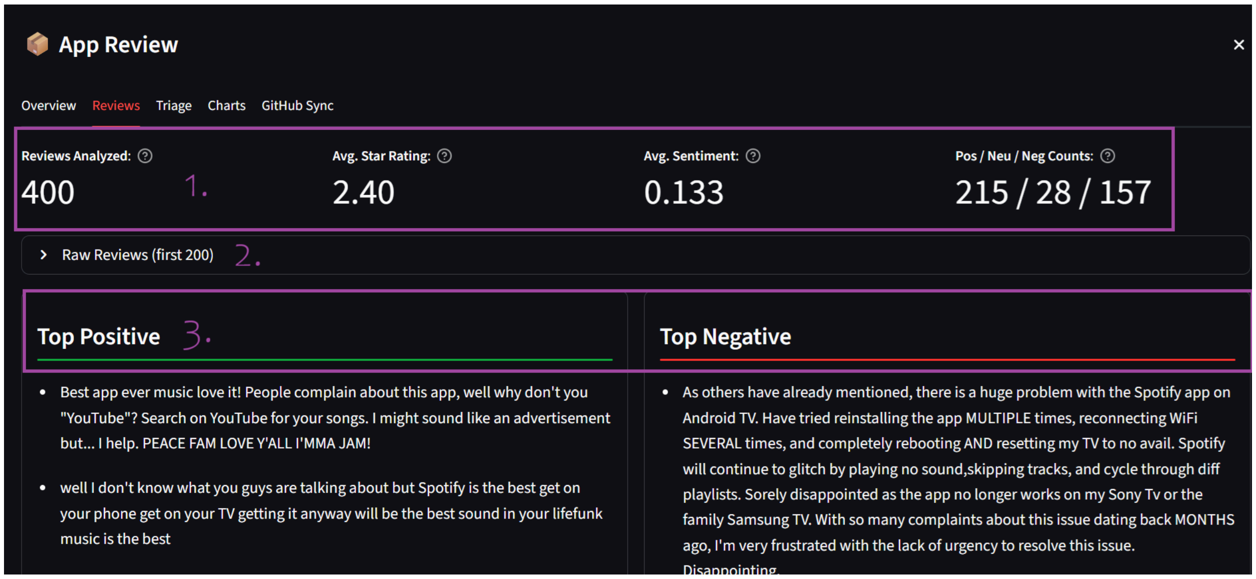
The next step in the pipeline is reviewing the app you have selected. There are several tabs to navigate through and explore, but immediately after clicking "Review in modal", you will find yourself on the Overview page:



1. **App Review**
   a. The App Review page is where the rest of the sentiment magic happens
2. **Tabs**
   a. Navigating through the tabs on the App Review page is how you access the various details and insights that Sentimend provides. In the coming sections, there will be more in depth descriptions of the different tabs' functions, but a brief overview is provided here
      i. **Reviews** - summary of the analyzed reviews sentiment, along with highlights positive and negative examples
      ii. **Charts** - Additional sentiment visualization
      iii. **Triage** - this is where the negative reviews are clustered to identify areas of improvement in the application based on user feedback. After the negative reviews are clustered, you may opt to receive possible solutions and GitHub issue drafts from and LLM
      iv. **GitHub Sync** - After you have submitted the identified clusters for LLM analysis, you can use this page to select, edit, and send your generated GitHub drafts straight to your desired repository for automatic creation.
3. **More details**
   a. Expanding this dropdown will display additional information about the analyzed app such as its developer, description, and appId
4. **Refetch & analyze**
   a. This is a forced refresh button to take new reviews or settings into account, but it shouldn't be needed much in practice

6.2.3 App Review - Reviews

Moving one tab to the right, we arrive at the Reviews tab. This page displays some general information about the sentiment analysis of the reviews, as well as some highlighted positive and negative examples:



Reviews Analyzed: 400

Avg. Star Rating: 2.40

Avg. Sentiment: 0.133

Pos / Neu / Neg Counts: 215 / 28 / 157

Raw Reviews (first 200)

Top Positive

- Best app ever music love it! People complain about this app, well why don't you "YouTube"? Search on YouTube for your songs. I might sound like an advertisement but... I help. PEACE FAM LOVE Y'ALL I'MMA JAM!
- well I don't know what you guys are talking about but Spotify is the best get on your phone get on your TV getting it anyway will be the best sound in your lifefunk music is the best

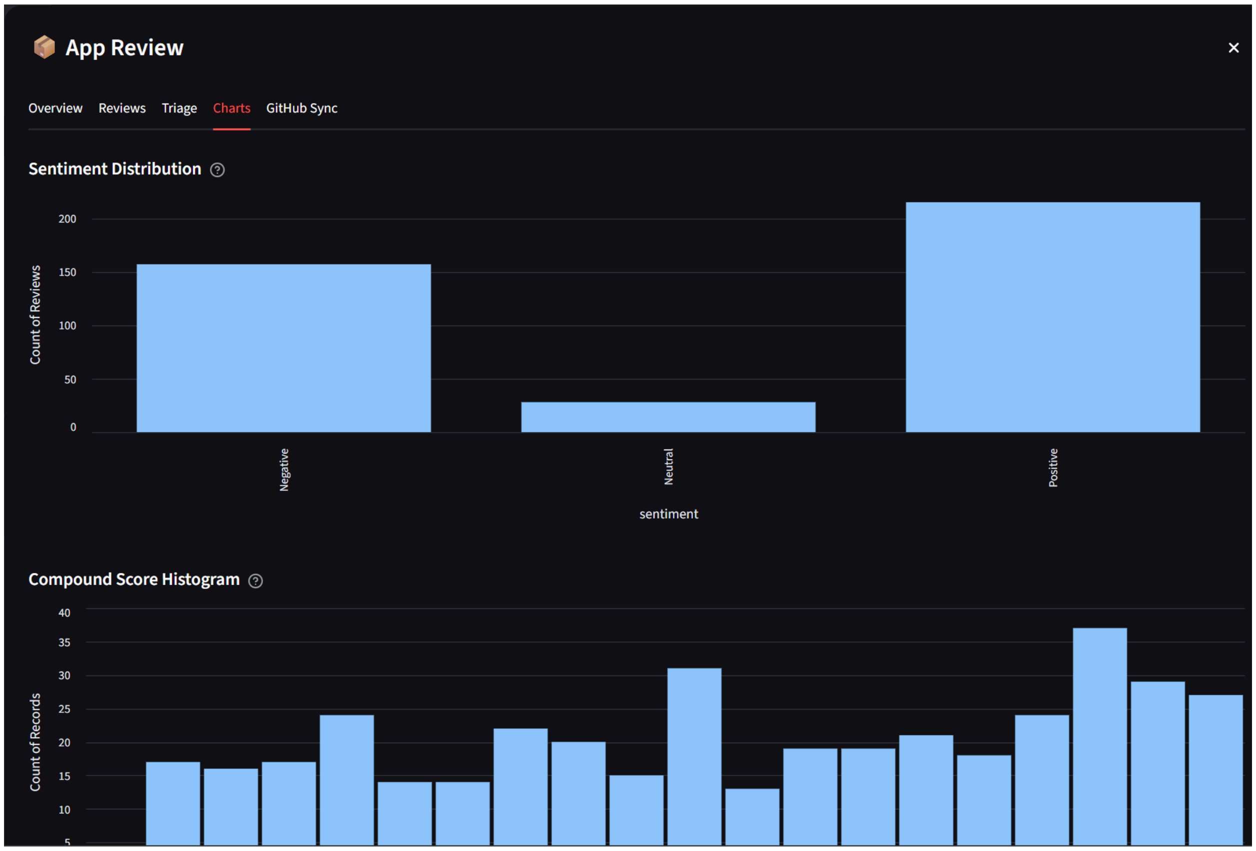
Top Negative

- As others have already mentioned, there is a huge problem with the Spotify app on Android TV. Have tried reinstalling the app MULTIPLE times, reconnecting WiFi SEVERAL times, and completely rebooting AND resetting my TV to no avail. Spotify will continue to glitch by playing no sound,skipping tracks, and cycle through diff playlists. Sorely disappointed as the app no longer works on my Sony Tv or the family Samsung TV. With so many complaints about this issue dating back MONTHS ago, I'm very frustrated with the lack of urgency to resolve this issue. Disappointing

1. **(Sentiment) Analysis Statistics**
    a. Information about the number of reviews analyzed, avg. star rating, avg. sentiment, and count of positive, neutral, and negative reviews
    b. The question mark icon to the right of each metric title will give additional information about what each metric is measuring
2. **Raw Reviews**
    a. Expanding this dropdown will reveal up to 200 raw reviews that were analyzed for manual inspection
3. **Top Positive/Top Negative**
    a. The most extreme positive and negative review examples

6.2.4 App Review - Charts

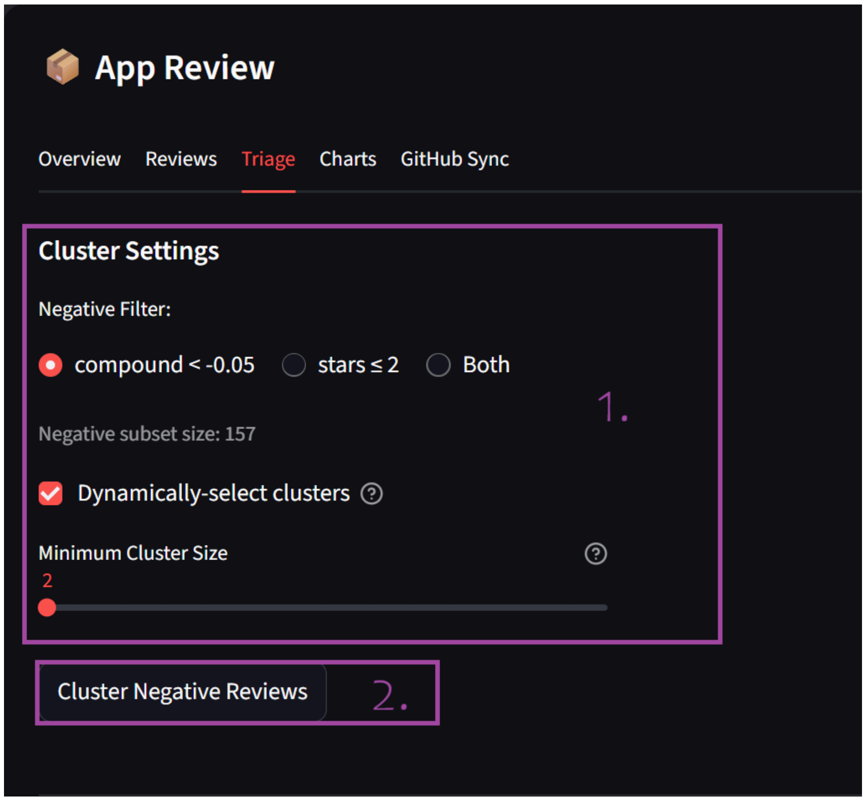
Directly to the right we see the Charts tab, which contains additional visualization for the sentiment analysis. There is not much additional description needed for this page, so just the screenshot will be provided below. This is intended to help developers understand the sentiment ratings and view how the positive/neutral/negative reviews are distributed:



6.2.5 App Review - Triage

The Triage page contains the next two definitive steps in the pipeline that follow app identification and initial/sentiment analysis. On this tab, users can cluster negative reviews to gain insight surrounding areas of improvement for the application they've chosen. Then, once the reviews have been clustered, they can submit the data for LLM analysis to generate reports and GitHub issues that can later be sent directly to their repository.

Since this tab contains functionality that is revealed in stages, it will be broken up into multiple screenshots:


App Review

Overview Reviews Triage Charts GitHub Sync

Cluster Settings

Negative Filter:

[✓ compound < -0.05] stars ≤ 2 Both

Negative subset size: 157

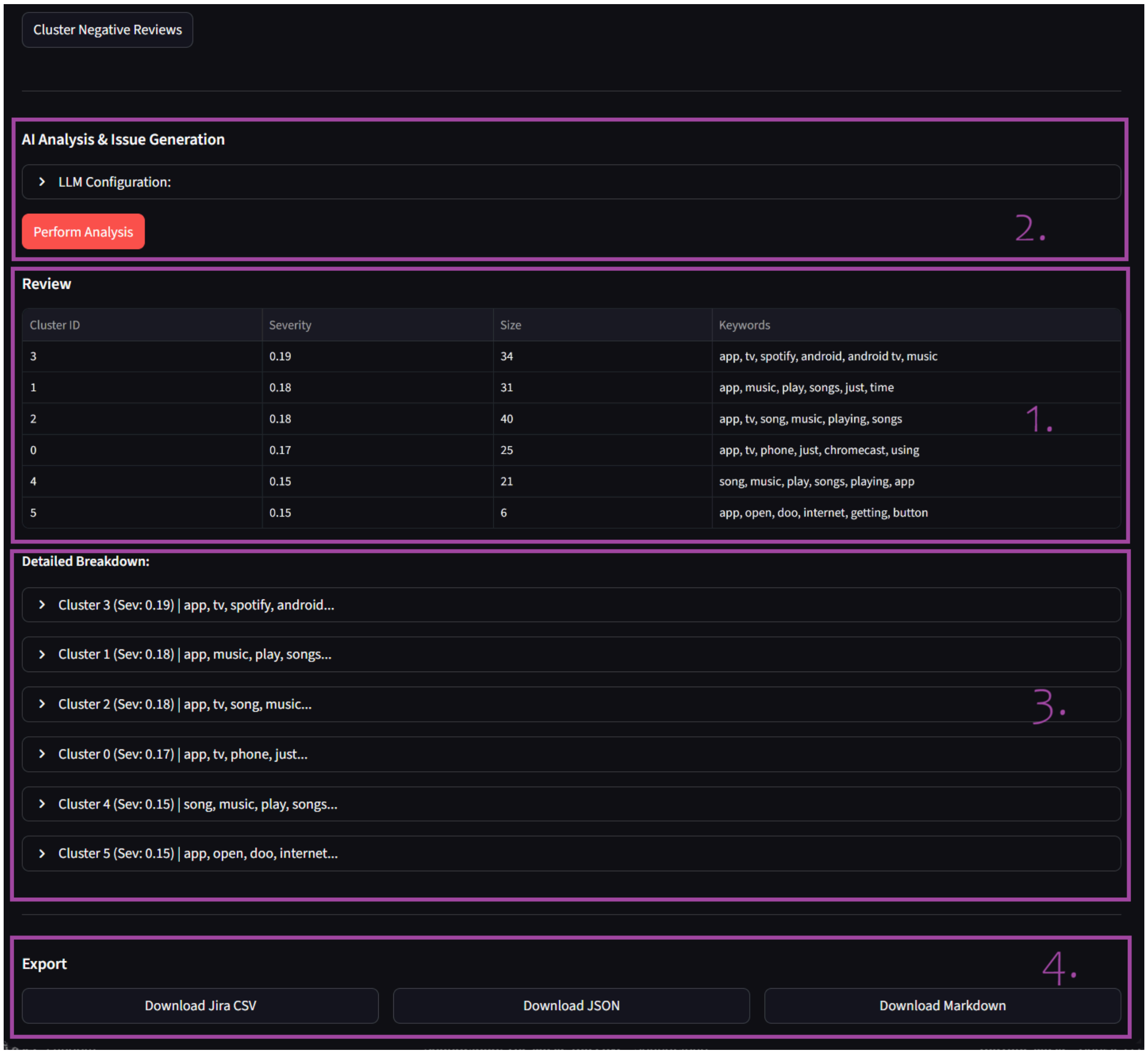
[✓] Dynamically-select clusters

Minimum Cluster Size

2

1. **Cluster Settings**
    a. The settings in this section allow you to configure the clustering process by specifying what constitutes a "negative" review, as well as the minimum number of reviews that must be in a cluster for it to be considered
    b. Additionally, by deselecting the "Dynamically-select clusters" option, the user can then use a slider to specify how many clusters should be created.
2. **Cluster Negative Reviews Button**
    a. Once the clustering options have been set, the operation is kicked off by clicking this button

After the clustering is complete, the tab will update and extend to show the additional content below:



Cluster ID	Severity	Size	Keywords
3	0.19	34	app, tv, spotify, android, android tv, music
1	0.18	31	app, music, play, songs, just, time
2	0.18	40	app, tv, song, music, playing, songs
0	0.17	25	app, tv, phone, just, chromecast, using
4	0.15	21	song, music, play, songs, playing, app
5	0.15	6	app, open, doo, internet, getting, button

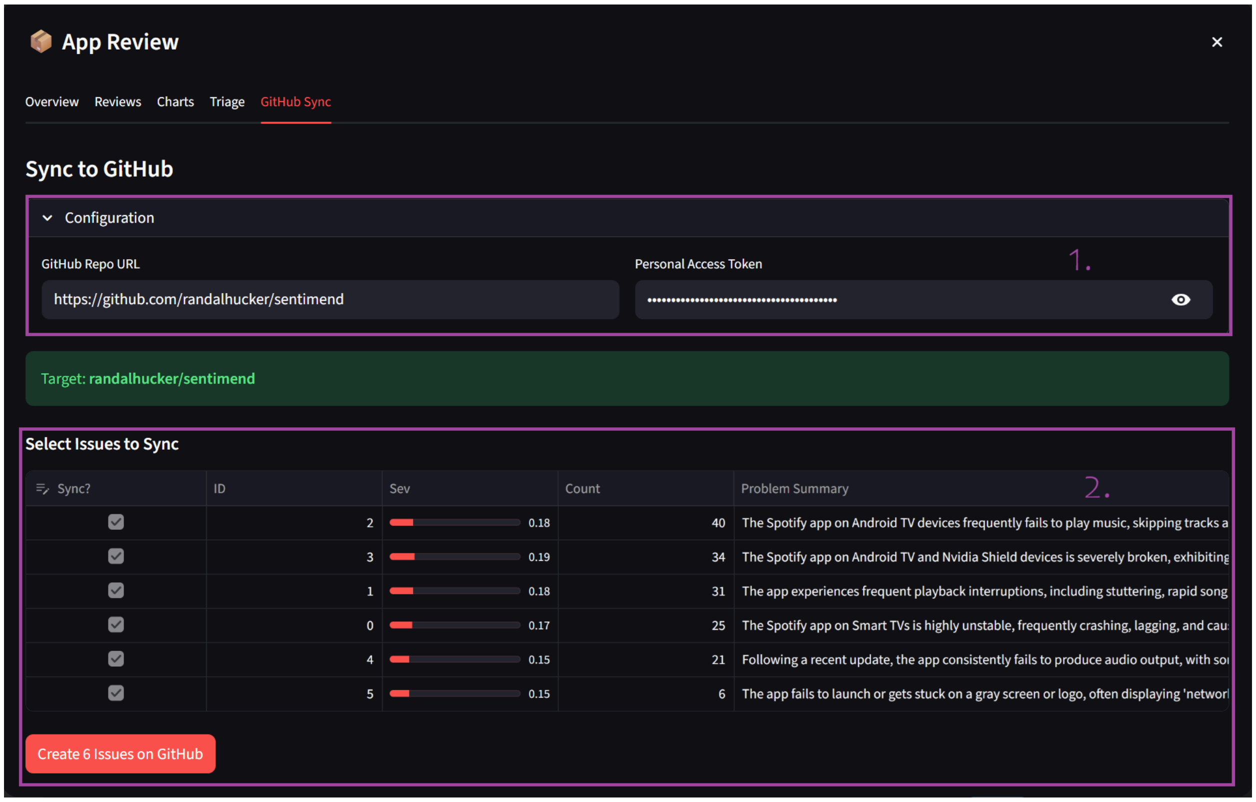
- Cluster 3 (Sev: 0.19) | app, tv, spotify, android...
- Cluster 1 (Sev: 0.18) | app, music, play, songs...
- Cluster 2 (Sev: 0.18) | app, tv, song, music...
- Cluster 0 (Sev: 0.17) | app, tv, phone, just...
- Cluster 4 (Sev: 0.15) | song, music, play, songs...
- Cluster 5 (Sev: 0.15) | app, open, doo, internet...

1. **Cluster Review**
   a. In the Review section, you can see a brief overview of each cluster including the severity, size, and keywords common among the reviews in the cluster
2. **AI Analysis & Issue Generation**
   a. This section allows the user to submit the identified clusters to an LLM for additional analysis and issue generation
   b. The LLM Configuration dropdown allows the user to provide additional context to the LLM, as well as select a support model of their choice
   c. Once any additional configuration has been entered, the user can initiate the process by clicking the "Perform Analysis" button.
3. **Detailed Breakdown**
   a. Once the LLM analysis has finished, the Detailed Breakdown dropdowns will be populated with the reports and issues that were generated
   b. These tabs will also feature example reviews from that particular cluster

4. **Export**
    a. The clustering data and analysis returned by the LLM can be exported in a variety of different formats using the buttons at the bottom of the page

6.2.6 App Review - GitHub Sync

The GitHub Sync page features the final stage in the pipeline: Allowing users to export the AI generated reports/issues directly to their GitHub repository:



Sync to GitHub

Configuration

GitHub Repo URL: https://github.com/randalhucker/sentimend

Personal Access Token: ••

Target: randalhucker/sentimend

Select Issues to Sync

Sync?	ID	Sev	Count	Problem Summary
✓	2	0.18	40	The Spotify app on Android TV devices frequently fails to play music, skipping tracks a
✓	3	0.19	34	The Spotify app on Android TV and Nvidia Shield devices is severely broken, exhibiting
✓	1	0.18	31	The app experiences frequent playback interruptions, including stuttering, rapid song
✓	0	0.17	25	The Spotify app on Smart TVs is highly unstable, frequently crashing, lagging, and cau
✓	4	0.15	21	Following a recent update, the app consistently fails to produce audio output, with so
✓	5	0.15	6	The app fails to launch or gets stuck on a gray screen or logo, often displaying 'networ

1. **Sync Configuration**
    a. This section allows you to enter the GitHub Repo URL for your project, as well as your Personal Access Token to allow Streamlit to create issues on your behalf
    b. It should be noted that this is intended for use on projects by the developers of that project, so they should have the proper access to the repository they provide
    c. Once the URL and access token are entered, the modal will extend and display the "Select Issues to Sync" section
2. **Issue Selection and Submission**
    a. Here users can select which issues they wish to sync to Github
    b. The LLM integration will generate one issue for each cluster, and from there the user can select which issues they believe are worthy to be entered and prioritized
    c. Once the desired issues have been selected/deselected, the user can select the "Create # issues on GitHub" to automatically create the desired issues

6.2.7 Usage Flow Summary

Now that the functionality of the app has been explained, we provide one final summary of the overall flow of the Sentimend pipeline.

1. Search for your desired app on Sentimend's main landing page
    a. Set any additional configurations for review language, quantity, recency, and star rating
2. Select "Review in modal" to begin the analysis
3. Review the information on the "Overview", "Review", and "Charts" tab
4. Cluster negative reviews
    a. Navigate to the "Triage" tab
    b. Enter review negativity and cluster configuration
    c. Select "Cluster Negative Review" to being clustering
5. Perform AI analysis for cluster reports and issue generation
    a. Scroll down and enter any additional LLM configuration
    b. Select "Perform Analysis" to begin the process
6. Review the Detailed Breakdown and export as needed
7. Send any generated issues desired to GitHub
    a. Navigate to the "GitHub Sync" tab
    b. Enter the GitHub configuration info and press Enter
    c. Select the issues you wish to create on Github
    d. Select "Create # Issues on GitHub" to send the selected issues

---