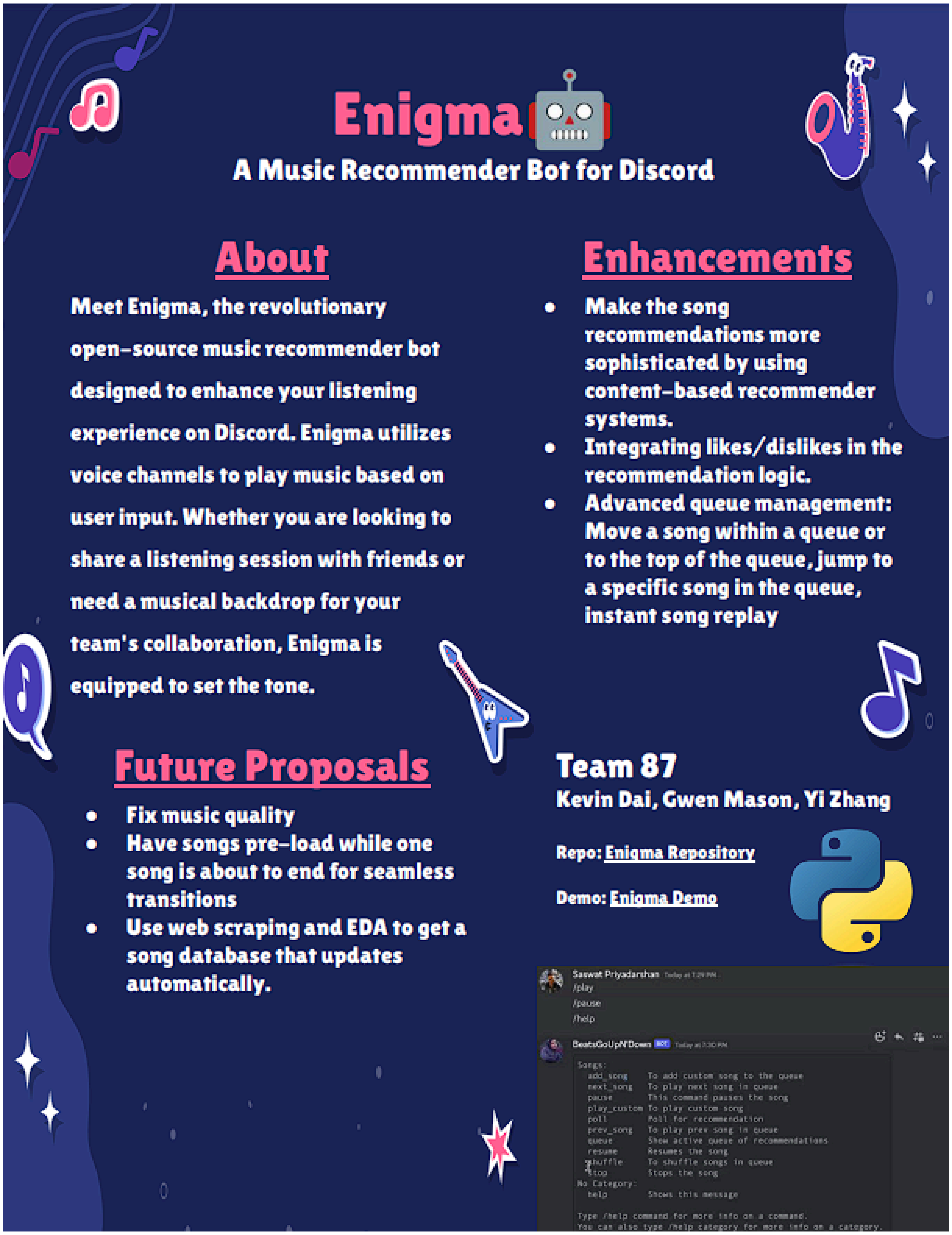
Project 2

Team 87: Yi Zhang (yzhan274), Gwen Mason (cpmason), Kevin Dai (kdai2)



Enigma

A Music Recommender Bot for Discord

About

Meet Enigma, the revolutionary open-source music recommender bot designed to enhance your listening experience on Discord. Enigma utilizes voice channels to play music based on user input. Whether you are looking to share a listening session with friends or need a musical backdrop for your team's collaboration, Enigma is equipped to set the tone.

Enhancements

- Make the song recommendations more sophisticated by using content-based recommender systems.
- Integrating likes/dislikes in the recommendation logic.
- Advanced queue management: Move a song within a queue or to the top of the queue, jump to a specific song in the queue, instant song replay

Future Proposals

- Fix music quality
- Have songs pre-load while one song is about to end for seamless transitions
- Use web scraping and EDA to get a song database that updates automatically.

Team 87

Kevin Dai, Gwen Mason, Yi Zhang

Repo: Enigma Repository

Demo: Enigma Demo

```
Saswat Priyadarshan  Today at 1:29 PM
/play
/pause
/help

BeatsGoUpN'Down BOT  Today at 1:30 PM
Songs:
  add_song    To add custom song to the queue
  next_song   To play next song in queue
  pause       This command pauses the song
  play_custom To play custom song
  poll        Poll for recommendation
  prev_song   To play prev song in queue
  queue       Show active queue of recommendations
  resume      Resumes the song
  shuffle     To shuffle songs in queue
  stop        Stops the song
No Category:
  help        Shows this message

Type /help command for more info on a command.
You can also type /help category for more info on a category.
```

Repo URL: https://github.com/CSC510-Team87/Enigma/tree/dev

Notes	Self-Assessment	Evidence
Workload is spread over the whole team (one team member is often Xtimes more productive than the others…		
but nevertheless, here is a track record that everyone is contributing a lot)		evidence in GH
Number of commits		in GH
Number of commits: by different people		in GH
Issues reports: there are **many**		in GH
Issues are being closed		evidence in GH
Docs: doco generated, format not ugly		in GH
Docs: what: point descriptions of each class/function (in isolation)		
Docs: how: for common use cases X,Y,Z mini-tutorials showing worked examples on how to do X,Y,Z		doc page entries

Docs: why: docs tell a story, motivate the whole thing, deliver a punchline that makes you want to rush out and use the thing		
Docs: short video, animated, hosted on your repo. That convinces people why they want to work on your code.		
Use of version control tools		
Test cases exist		dozens of tests and those test cases are more than 30% of the code base
Test cases are routinely executed		E.g. travis-com.com or github actions or something
Issues are discussed before they are closed		even if you discuss in slack, need a sumamry statement here
Chat channel: exists		Link or screenshots
Test cases: a large proportion of the issues related to handling failing cases.		If a test case fails, open an issue and fix it
Evidence that the whole team is using the same tools: everyone can get to		

all tools and files		
Evidence that the whole team is using the same tools (e.g. config files in the repo, updated by lots of different people)		
Evidence that the whole team is using the same tools (e.g. tutor can ask anyone to share screen, they demonstrate the system running on their computer)		
Evidence that the members of the team are working across multiple places in the code base		
Short release cycles		(hard to see in short projects) project members are committing often enough so that everyone can get your work
The file .gitignore lists what files should not be saved to the repo. See [examples]i(https://github.com/github/gitignore)		in GH
The file INSTALL.md lists how to install the code		in GH
The file LICENSE.md lists rules of usage for this repo		in GH

The file CODE-OF-CONDUCT.md lists rules of behavior for this repo; e.g. see example		in GH
The file CONTRIBUTING.md lists coding standards and lots of tips on how to extend the system without screwing things up; e.g. see example		in GH
The file README.md contains all the following		in GH
Video		2min video of new functionality, showing a significant delta from prior.
DOI badge: exists. To get a Digitial Object Indentifier, regiser the project at Zenodo. DOI badges look like this:		in GH
Badges showing your style checkers		config files in GH showing your config, badges in README
Badges showing your code formatters.		config files in GH showing your this formatter’s config, badges in README
Badges showing your syntax checkers.		config files iin GH showing this checker’s config, badges in README

Badges showing your code coverage tools		config files in GH, badges in README
Badges showing any other Other automated analysis tools		config files in GH, badges in README
Question 1.1: Does your website and documentation provide a clear, high-level overview of your software?		
Question 1.2: Does your website and documentation clearly describe the type of user who should use your software?		
Question 1.3: Do you publish case studies to show how your software has been used by yourself and others?		
Question 2.1: Is the name of your project/software unique?*		
Question 2.2: Is your project/software name free from trademark violations?		

Question 3.1: Is your software available as a package that can be deployed without building it?		
Question 3.2: Is your software available for free?		
Question 3.3: Is your source code publicly available to download, either as a downloadable bundle or via access to a source code repository?		

Question 3.4: Is your software hosted in an established, third-party repository likeGitHub (https://github.com), BitBucket (https://bitbucket.org),LaunchPad (https://launchpad.net) orSourceForge (https://sourceforge.net)?		
Question 4.1: Is your documentation clearly available on your website or within your software?		
Question 4.2: Does your documentation include a "quick start" guide, that		

provides a short overview of how to use your software with some basic examples of use?		
Question 4.3: If you provide more extensive documentation, does this provide clear, step-by-step instructions on how to deploy and use your software?		
Question 4.4: Do you provide a comprehensive guide to all your software’s commands, functions and options?		
Question 4.5: Do you provide troubleshooting information that describes the symptoms and step-by-step solutions for problems and error messages?		
Question 4.6: If your software can be used as a library, package or service by other software, do you provide comprehensive API documentation?		
Question 4.7: Do you store your documentation under revision control with your source code?		
Question 4.8: Do you publish your release history e.g. release data, version numbers, key features of each release etc. on your web site or in your documentation?		

Question 5.1: Does your software describe how a user can get help with using your software?		
Question 5.2: Does your website and documentation describe what support, if any, you provide to users and developers?		
Question 5.3: Does your project have an e-mail address or forum that is solely for supporting users?		
Question 5.4: Are e-mails to your support e-mail address received by more than one person?		
Question 5.5: Does your project have a ticketing system to manage bug reports and feature requests?		
Question 5.6: Is your project's ticketing system publicly visible to your users, so they can view bug reports and feature requests?		
Question 6.1: Is your software's architecture and design modular?*		
Question 6.2: Does your software use an accepted coding standard or convention?		
Question 7.1: Does your		

software allow data to be imported and exported using open data formats? e.g. GIF, SVG, HTML, XML, tar, zip, CSV, JSON, NetCDF, or domain specific ones		
Question 7.2: Does your software allow communications using open communications protocols? e.g. HTTP, FTP, XMPP, SOAP over HTTP, or domain-specific ones		

Question 8.1: Is your software cross-platform compatible?* e.g. does it run under two or more of Windows, Unix/Linux and Mac OS X, or can be used from within two or more of Internet Explorer, Chrome, Firefox and Safari? Yes No		
Question 9.1: Does your		

software adhere to appropriate accessibility conventions or standards?* Yes No		
Question 9.2: Does your documentation adhere to appropriate accessibility conventions or standards?* Yes No		
Question 10.1: Is your source code stored in a repository under revision control?* Yes No		
Question 10.2: Is each source code release a snapshot of the repository?* Yes No Not applicable		
Question 10.3: Are releases tagged in the		

repository?* Yes No Not applicable		
Question 10.4: Is there a branch of the repository that is always stable? (i.e. tests always pass, code always builds successfully)* Yes No Not applicable		
Question 10.5: Do you back-up your repository?* Yes No Not applicable		
Question 11.1: Do you provide publicly-available instructions for building your software from the source code?* Yes No		
Question 11.2: Can you build, or package, your software using an automated tool?*		

e.g. Make (https://www.gnu.org/software/make/), ANT (http://ant.apache.org/), Maven (https://maven.apache.org/), CMake (https://cmake.org/), Python setuptools (https://pypi.python.org/pypi/setuptools), or R package tools (https://cran.r-project.org/doc/manuals/r-devel/R-exts.html) Yes No		
Question 11.3: Do you provide publicly-available instructions for deploying your software?* Yes No		
Question 11.4: Does your documentation list all third-party dependencies?* Yes No		
Question 11.5: Does your documentation list the version number for all		

third-party dependencies?* Yes No Not applicable		
Question 11.6: Does your software list the web address, and licences for all third-party dependencies and say whether the dependencies are mandatory or optional?* Yes No Not applicable		
Question 11.7: Can you download dependencies using a dependency management tool or package manager?* e.g. Ivy (http://ant.apache.org/ivy/), Maven (https://maven.apache.org/), Python pip (https://pypi.python.org/pypi/pip) or setuptools (https://pypi.python.org/pypi/setuptools), PHP Composer (https://getcomposer.org/), Ruby gems		

(https://rubygems.org), or R PackRat (https://rstudio.github.io/packrat/) Yes No Not applicable		
Question 11.8: Do you have tests that can be run after your software has been built or deployed to show whether the build or deployment has been successful?* Yes No		
Question 12.1: Do you have an automated test suite for your software?* Yes No		
Question 12.2: Do you have a framework to periodically (e.g. nightly) run your tests on the latest version of the source code?* Yes No Not applicable		

Question 12.3: Do you use continuous integration, automatically running tests whenever changes are made to your source code?* Yes No Not applicable		
THERE IS MORE, COPY AND PASTE THEM FROM THE FORM		