

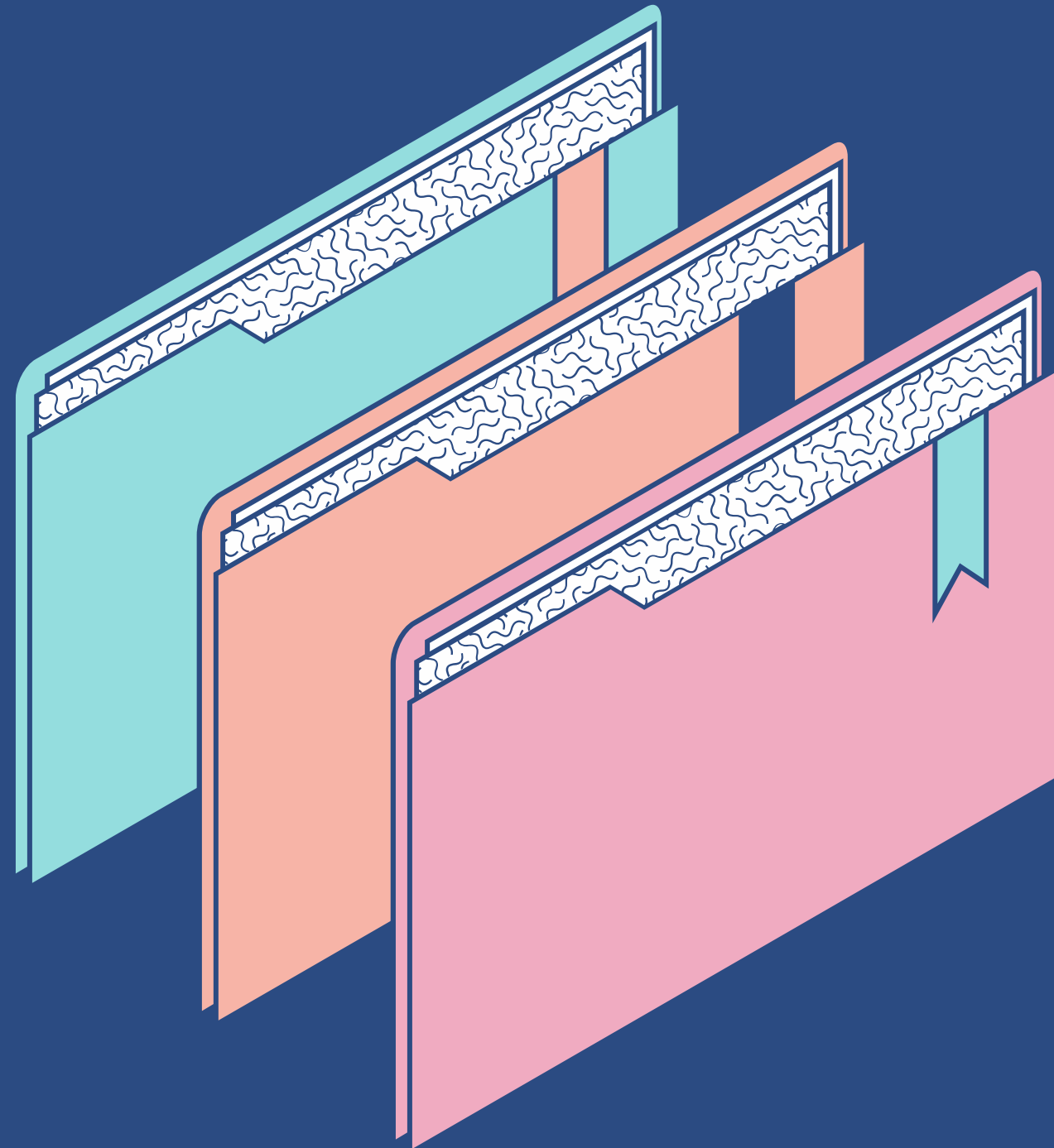


FINAL PROJECT PRESENTATION

KarmaTracker: Slack Reward Bot

Team VEM

Morgan Pham, Eunice Hong, Aiden Barker, Vatsa Bhatt



Agenda

KEY TOPICS DISCUSSED IN THIS PRESENTATION

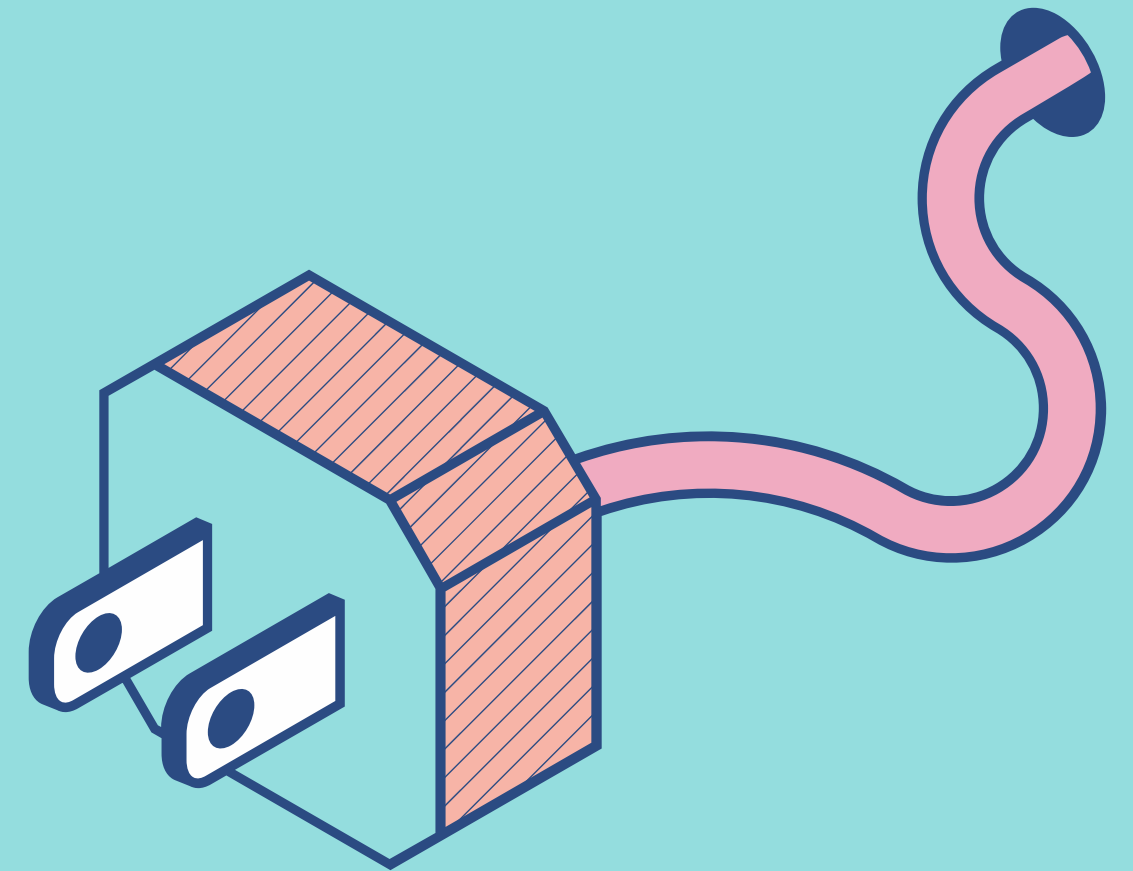
- Problem and solution
- Use cases and diagrams
- Visual representation of the project
- Discussion on limitations and future work
- Processes and tools used/to be used
- What we learned

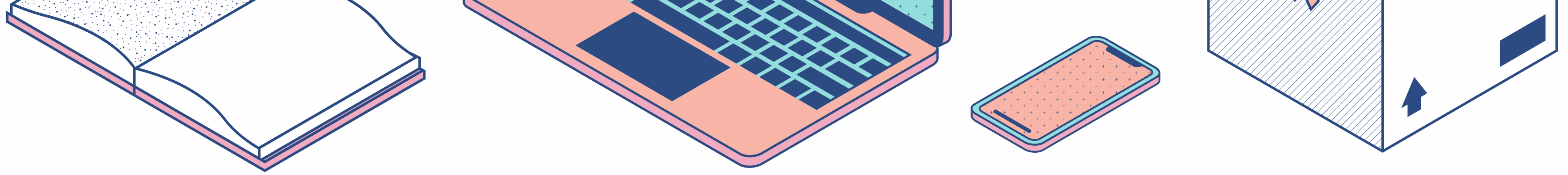
What's the problem?

Due to a lack of teamwork, collaboration, and communication, the quality and efficiency of software has fallen over time.

So who's to blame, or rather, *what's* to blame?

Software has become more inefficient and has dropped in quality over time, leading to unmanageable code bases and missing requirements that were agreed upon prior to development.





Proposed Solution

- **KarmaTracker:** Slack Bot rewards points to employees performing good deeds that aim to fix the problems that poor quality software creates
- Bot given commands by employees that recognize others performing helpful actions
- Slack channel: weekly-shoutouts
- React app to display leaderboard and individual collective points

Why it Works

- Promotes collaboration and teamwork between groups, specifically software engineers
 - Commands could be generalized to other groups than engineers
- More positive pressure or reinforcement to perform these actions as opposed to politely asking
- Social impact and managerial impact

Use Case 1

AWARD POINTS FOR DEBUGGING SESSION

- a. Actors
 - i. Peer: The engineer responsible for letting the bot know about awarding points to their peer
 - ii. Engineer: The engineer who took responsibility for setting up the debugging meeting
- b. Preconditions

Peer and engineer must have a slack account and are registered to a channel. Moreover, an engineer should set a meeting time for a debugging session.
- c. Main Flow
 - i. [S1] Engineer will create and request a debugging meeting, and provide a list of other engineers invited
 - ii. [S2] Peer notices this and reports it to the KarmaTracker bot, that will track the engineer's action
 - iii. [S3] The KarmaTracker bot updates awarded points to the engineer's individual leaderboard
- d. Subflows
 - i. [S1] KarmaTracker lets engineer know who awarded them with points
 - ii. [S2] The peer checks the engineer's performances
 - iii. [S3] The engineer creates the meeting
- e. Alternative Flows
 - i. [E1] The engineer themselves do not attend the meeting.
 - ii. [E2] The peer reports a wrong person
 - iii. [E3] No meeting is created

Use Case 2

AWARD POINTS FOR CODE REVIEW

- a. Actors
 - i. The engineer responsible for letting the bot know about awarding points to their peer
 - ii. Engineer: The engineer who took responsibility for answering a question outside their team
- b. Preconditions
 - i. Peer and engineer must have a slack account and are registered to a channel
 - ii. Engineer must have performed a code review for peer without being required to.
- c. Main Flow
 - i. Engineer will perform and complete a code review for their peer without being required to do so.
 - ii. The peer receives notice that the engineer has completed a code review for them and reports it to the KarmaTracker bot that will track this action. [S1]
 - iii. KarmaTracker bot adds points to the engineer's individual leaderboard. [S2]
- d. Subflows
 - i. [S1] User provides bot with command to award points to engineer.
 - ii. [S2] Bot will add to the engineer's respective points in the backend.
- e. Alternative Flows
 - i. [E1] The peer notices the action but does not report it.

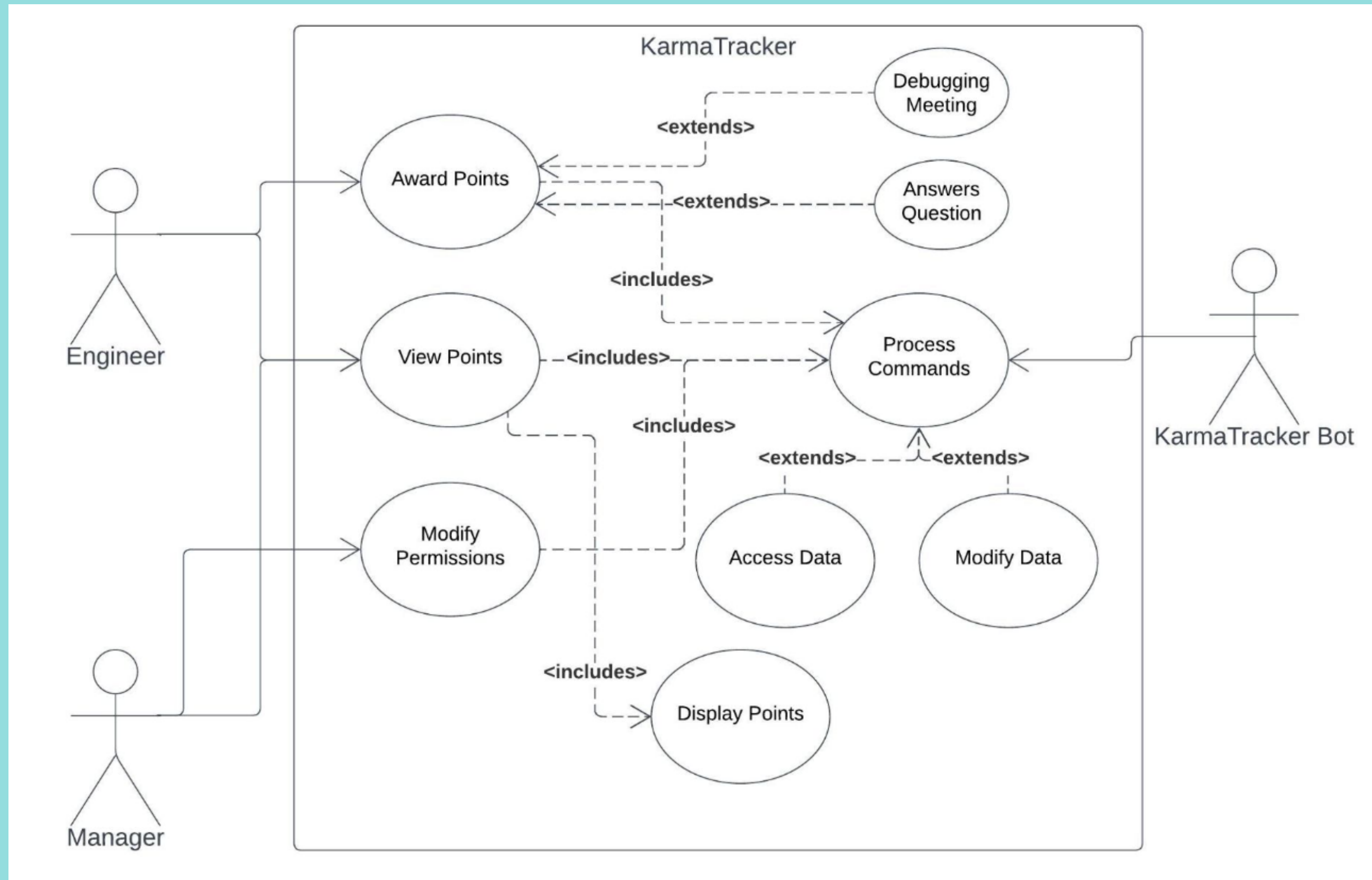
Use Case 3

MANAGER VIEWS

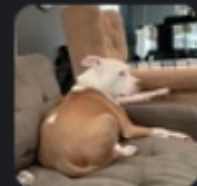
ENGINEER'S POINTS FOR
EMPLOYEE EVAL.

- a. Actors
 - i. Manager: The individual responsible for evaluating and making decisions based on engineer's performance
 - ii. Engineer: The individual whose cumulative points and commendable actions are being evaluated.
- b. Preconditions
 - i. Manager must have permissions to see history of awarded points in more detail.
 - ii. Engineers must have been previously using KarmaTracker to have a history of commendable actions and points awarded.
- c. Main Flow
 - i. [S1] Manager opens the private Slack channel to start evaluation.
 - ii. [S2] Manager sends message to the bot with a special command and then the engineer's username.
 - iii. [S3] KarmaTracker receives request and starts to process it.
- d. Subflows
 - i. [S1] KarmaTracker queries data for desired engineer.
 - ii. [S2] KarmaTracker will display awarded points on a weekly basis, cumulative points for the month, and year.
 - iii. [S3] Manager evaluates engineer's performance.
 - iv. [S4] Manager makes decision related to position, bonuses, etc.
- e. Alternative Flows
 - i. [E1] Engineer hasn't accumulated any points for commendable actions, thus there will be no history of points.

Use Case Diagram




Mock User Interface: Slack



Morgan Pham 10:23 PM

@KarmaTracker ++debug @EuniceHong

weekly-shoutouts

You created this channel today. This is the very beginning of the  weekly-shoutouts channel.

 Add description

 Add coworkers

Today ▾



KarmaBot 9:56 PM

joined weekly-shoutouts.

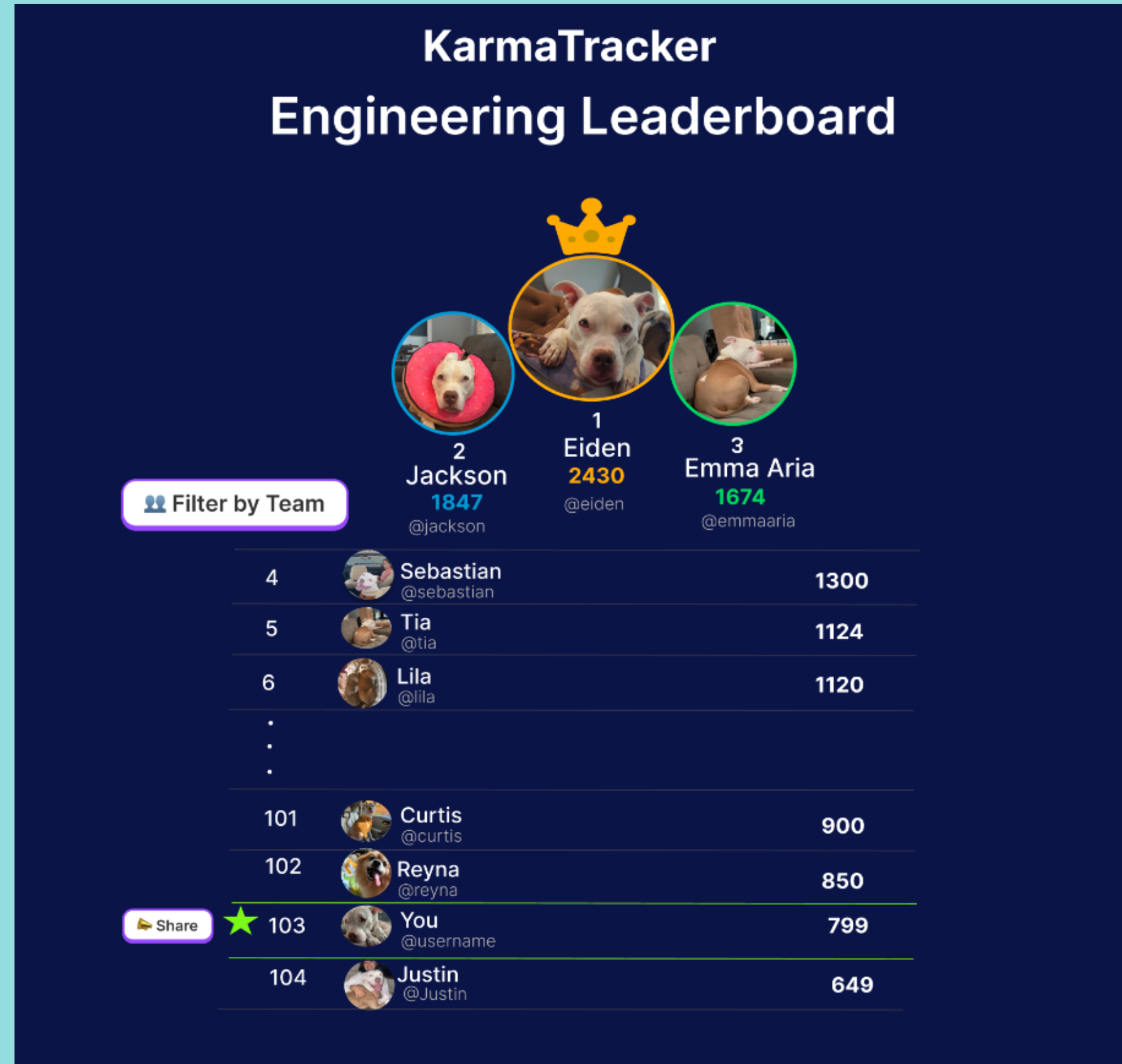


KarmaBot 9:57 PM

Oct 30 - Nov 3 : Congratulations to Peer1 for being our top point jumper of the week! Through their assistance in debugging code and code reviews we are able to better function as an organization. Thank you Peer1 for taking care of the team! 🥳 (edited)

Nov 6 - 10 : Congratulations to Peer5 for being our top point jumper of the week! Through their assistance in answering external team's questions and knowledge sharing we are able to better function as an organization. Thank you Peer1 for taking care of the team! 🎉

Mock User Interface: React





Limitations on KarmaTracker

- Subjectivity of recognition
 - different perspectives
- Gaming the system
 - insincere behavior only to gain points rather than authentic collaboration
- Risk of distraction
 - Too focused on gaining points or leaderboard

Future Implementations

- Machine learning for smart recognition
 - Recognize positive behavior without peer commands
- Anonymous recognition
 - Acknowledge behaviors without favoritism or bias
- Integration with project management tools
 - Connect KarmaTracker with other project management tools

Tools and Processes

RELATED TO PROJECT MANAGEMENT AND DEVELOPMENT



Embracing Agile's iterative approach

Helpful for continuous feedback and adaptable requirements. Utilized Scrum with daily standups, sprint retrospectives, etc.

Slack for communication and collaboration

Used Slack to plan and conduct frequent meetings regarding project reviews and feedback.

User feedback loop

Didn't get to implementation during this project, but would utilize user feedback loops to gather user input and adjust/update project.

Related Papers

CONVERSATIONAL AGENTS AND HYBRID TEAMWORK

Designing a Conversational Agent to Promote Teamwork and Collaborative Practices Using Design Thinking: An Explorative Study on User Experiences" by Sofie Smedegaard Skov, et. al.

- The study explores user experiences with this conversational agent, examining how it influences and supports collaborative efforts within a team. The research provides valuable insights into the effectiveness of employing conversational agents to enhance teamwork and the application of design thinking methodologies in this context.

"Hybrid Teamwork: Consideration of Teamwork Concepts to Reach Naturalistic Interaction between Humans and Conversational Agents" by Mathis Poser and Eva A. C. Bittner

- The study explores the integration of teamwork principles into the design and functionality of conversational agents. By considering teamwork concepts, the article aims to enhance the collaborative interaction between humans and conversational agents, providing insights into the potential of achieving more natural and effective communication in human-agent interactions.



Concepts Learned

- **User-Centered Design**
 - Useful for ensuring that system would be easy to use in all scenarios, which would encourage use amongst teams
- **Chain of Responsibility**
 - Useful for processing commands used in our system
 - Allows us to write clean and organized code as the command is passed along the “chain” accordingly
- **Layered Architecture**
 - Useful for handling communication between the different layers of the system
 - Strong framework for flexibility and scalability, to help easily add more features in the future

Resources Used

- <https://www.frontiersin.org/articles/10.3389/fpsyg.2022.903715/full>
- https://www.researchgate.net/profile/Mathis-Poser-2/publication/339797657_Hybrid_Teamwork_Consideration_of_Teamwork_Concepts_to_Reach_Naturalistic_Interaction_between_Humans_and_Conversational_Agents/links/5e6a1e9e299bf1b9f7ceb921/Hybrid-Teamwork-Consideration-of-Teamwork-Concepts-to-Reach-Naturalistic-Interaction-between-Humans-and-Conversational-Agents.pdf