Requirements Workshop

Non-Functional Requirements

1. Provide an example of five hypothetical non-functional requirements for this system. Be sure to include the specific type of requirement discussed in class, with each requirement coming from a unique category.

1. Usability: KarmaTracker should include a user guide for newly onboarded engineers as they join Slack.
2. Reliability: When awarding points, KarmaTracker should have a 12 hour MTTR (mean time to recover).
3. Performance: The server response time is <5 seconds for 85% of the actions.
4. Supportability: The system should be able to easily adapt and fit to a specific team's needs.
5. Implementation/Constraints: System to be built with React and TypeScript.

Functional Requirements

2. Provide an example of five hypothetical functional requirements for this system.

1. KarmaTracker must keep track of users point values for one-week periods (short-term) as well as an all-time cumulative point total (long-term).
2. KarmaTracker must have a help command listing out all commands available and how to use each.
3. KarmaTracker must track engineers' points week by week to identify the top point-jumper each period.
4. KarmaTracker must send a message into the "Weekly Winner" Slack channel to congratulate the identified top point-jumper for the period.
5. KarmaTracker must notify the awarded engineer every time points are given to them.
    a. If the person who awarded the engineer would like to remain anonymous, then only the reason why points were given will be displayed. Otherwise, point awarder and reason for points will be shown in the notification to the engineer.

Tasks

3. Think of a specific task required to complete each of the functional requirements and non-functional requirements mentioned above (10 total). Estimate the amount of effort needed to complete this task using function points (i.e., using the values here). Briefly explain your answer.

Non-Functional tasks:

1. Write up user documentation for how to use KarmaTracker - should include all available commands, how to use each, and explain the “Weekly Winner” Slack channel and associated React leaderboard application. Estimated effort: Low as it shouldn’t be too technical and provides a general overview. (2 points)
2. Implement continuous monitoring of the server with processes in place to reboot the system back into working state when needed. Estimated effort: Medium, depends on how the server will be implemented which may lead to difficulties. (5 points)
3. Implement command validation to validate commands sent to KarmaTracker to ensure the formatting and given arguments are correct.
    a. Taking the “Fail Fast” approach will help in reducing KarmaTracker errors that can create downtime.

    Estimated effort: Low, have to implement error handling dependent on command arguments. (2 points)
4. Implement role access controls to allow managers to add new engineers on their team into their line of KarmaTracker visibility.
    a. Only managers will be able to see who awarded points to a specific engineer, when this was done, and for what reason.
    b. All others can only see an engineer’s total points. Breakdowns are visible only to managers.

    Estimated effort: Low-Medium, have to use Slack API to enable access to certain functionalities for specified users. (3 points)

5. Set up a development environment with React and TypeScript. Estimated effort: Low, only have to install required dependencies. (1 point)

Functional tasks:

1. Implement a notification system that tells the awarded engineer whenever points are provided to them. This task entails message production, routing, and notification settings. Estimated effort: Medium, because it involves message handling and notification settings. (8 points)
2. Create a feature that sends a message to the "Weekly Winner" Slack channel to congratulate the top point-jumper of the week. This assignment entails connecting with the Slack API and formatting messages. Estimated effort: Low to Medium, depending on the integration's complexity. (5 points)
3. Create an algorithm to track engineers' points week by week in order to find the top point-jumper for each period. This entails creating and implementing a scoring system

that operates on the collected data. Estimated effort: Medium, because algorithm development is required. (3 points)

4. Within KarmaTracker, create a command that shows all accessible commands and provides instructions on how to use each one. This includes writing text responses that are easy to read and properly formatted. Estimated effort: Low, since it just requires returning the accepted commands. (3 points)
5. Create and implement a system that tracks user point values across one-week periods and keeps a cumulative point total. It would necessitate database design, data storage, and logic for updating and retrieving user points. Estimated effort: Medium, due to database design and sophisticated logic. (8 points)

User Stories

4. Write three user stories from the perspective of at least two different actors. Provide the acceptance criteria for these stories.

1. User Story 1 (Actor: Engineer):
    a. Title: Awarding Points to a Colleague
    b. Story: As an engineer, I want to be able to award points to my colleagues in recognition of their efforts and achievements.
    c. Acceptance Criteria:
        i. I can enter the name of the receiving engineer into the system.
        ii. I can specify why I'm awarding points.
        iii. I need to be able to define how many points to award.
        iv. The points should be successfully awarded to the selected engineer after submission.
        v. To validate the successful point award, an appreciation message is presented.
2. User Story 3: Viewing My Karma Points (Actor: Team Member)
    a. Story: As a team member, I'd like to see my karma points and learn how they're dispersed so that I can track my contributions and achievements.
    b. Acceptance Criteria:
        i. I can access KarmaTracker dashboard
        ii. On the dashboard, the system should show my entire karma points.
        iii. I should be able to see a breakdown of my karma points, displaying points earned for various reasons.
        iv. The system should allow users to filter and examine points earned during a given time period (for example, weekly or monthly).
        v. I can view details such as who awarded the points and why by clicking on each point entry.
3. User Story 3 (Manager Perspective):
    a. As a Manager, I want to be able to add new engineers to our team in KarmaTracker, so I can ensure accurate tracking of their performance.

   b. Acceptance Criteria:
      i. I should have access to a "Manage Team" section in KarmaTracker.
      ii. Within the "Manage Team" section, there should be an option to add new engineers to our team.
      iii. When adding an engineer, I should provide their name and Slack username.
      iv. The new engineer should be added to our team's visibility, allowing us to see who awarded points to them and for what reasons.
      v. The engineer should receive a notification that they've been added to our team in KarmaTracker.

Risks

5. Provide two examples of risk that could potentially impact this project. Explain how you would mitigate these risks if you were implementing your project as a software system.

1. Risk 1: Integration Challenges with Slack API
   a. Description:
      i. The project relies on integrating with the Slack API for notifications and user interactions. Any changes or issues with the Slack API can impact the project's functionality and reliability.
   b. Mitigation:
      i. Continuous Monitoring: Set up a way to monitor the integration with the Slack API for any disruptions or changes. This will aid in the early detection of problems.
      ii. Controlling Versions: Keep track of the Slack API version used in the integration and update it to the most recent stable version on a regular basis to avoid compatibility concerns.
      iii. Create a backup communication system in case of lengthy Slack API outages, such as email notifications, to guarantee crucial messages are still sent.
2. Risk 2: Data Security and Privacy Concerns
   a. Description:
      i. The project involves tracking and recording user interactions and points, which could raise data security and privacy concerns among users.
   b. Mitigation:
      i. Data Encryption: To protect user information, ensure that all user data and interactions are encrypted both in transit and at rest.
      ii. Implement user privacy options, which allow users to restrict the visibility of their point awards and interactions.
      iii. Compliance with legislation: Depending on the nature of the data acquired and the user base, stay in compliance with relevant data protection legislation such as GDPR, HIPAA, and so on.

Process

6. Describe which process your team would use for requirements elicitation from clients or customers, and explain why.

The team would use a combination of Agile and user-centered design methodologies for requirements elicitation from clients or customers.

Explanation:

- Iterative Approach: Agile allows for iterative development, which is well-suited for requirements elicitation as it allows for feedback and changes throughout the project.
- Client Involvement: Agile methodologies encourage active client involvement in the development process, ensuring that the client's needs and expectations are consistently met.
- User-Centered Design: User-centered design principles would be employed to ensure that the system is user-friendly.
- Regular Feedback: Frequent meetings and reviews of the project's progress and user feedback allow for ongoing adjustments.
- Adaptability: Agile methods enable the team to adapt to changing client needs and evolving project requirements.

By using Agile and user-centered design, the team ensures that the client's requirements are thoroughly understood, and the resulting software system is aligned with the client's expectations and user needs.

Requirements Analysis

1. Use Case: Peer wants to award points to engineer for scheduling debugging meeting
    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. <u>Subflows</u>
        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. <u>Alternative Flows</u>
        i. [E1] The engineer themself do not attend the meeting.
        ii. [E2] The peer reports a wrong person
        iii. [E3] No meeting is created

2. Use Case: Peer wants to award points to engineer for answering question outside of own team
    a. <u>Actors</u>
        i. Peer: 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. <u>Preconditions</u>
       Peer and engineer have a slack account and are registered to a channel. Moreover, an engineer should answer a question asked by another team.
    c. <u>Main Flow</u>
        i. [S1] Engineer will answer a question outside their own team on the general channel where all teams ask questions.
        ii. [S2] The teammate or same group peer notices this and reports it to the KarmaTracker bot, that will track this action
        iii. [S3] The KarmaTracker bot checks if the engineer answered outside the team, and adds points to the engineer’s individual leaderboard
    d. <u>Subflows</u>
        i. [S1] KarmaTracker lets engineer know which peer asked to award them points
        ii. [S2] The peer checks the engineer’s performance for displaying and sharing knowledge outside their own team
        iii. [S3] The engineer answers a question in the general channel or the channel where other teams ask questions
    e. <u>Alternative Flows</u>
        i. [E1] The peer notices the action but does not report it
        ii. [E2] The engineer does not answer the question

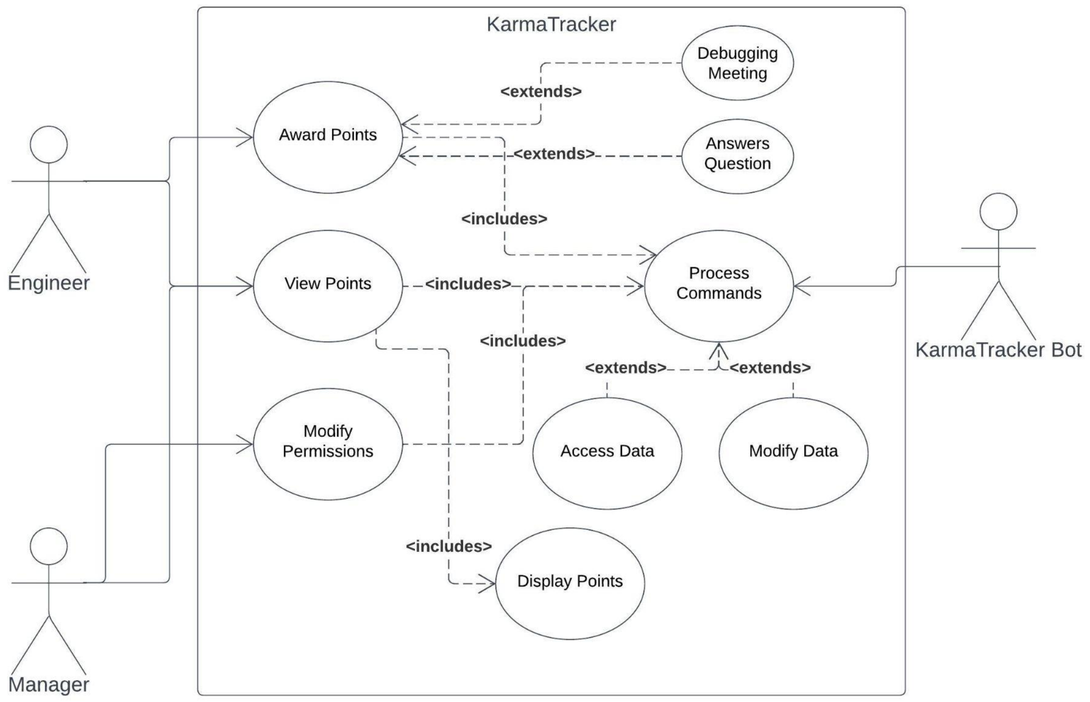
3. Peer wants to award points to engineer for performing a code review
    a. <u>Actors:</u>

        i. Peer: 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. <u>Preconditions</u>
        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. <u>Main Flow</u>
        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. The KarmaTracker bot adds points to the engineer's individual leaderboard. [S2]
    d. <u>Subflows</u>
        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. <u>Alternative Flows</u>
        i. [E1] The peer notices the action but does not report it.

4. Use Case: Manager wants to view engineer's cumulative points for employee evaluation.
    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.

5. Use Case: Peer Wants to View Engineer's Cumulative Karma Points
    a. Actors:
        i. Peer (the person interested in viewing an engineer's karma points)
        ii. Engineer (the target engineer whose karma points the peer wants to view)
    b. Preconditions:
        i. Peer and Engineer have active Slack accounts.
        ii. Peer and Engineer are registered to the same Slack channel.
        iii. The KarmaTracker system is operational and integrated with the Slack channel.
    c. Main Flow:
        i. [S1] Peer, who wants to view an engineer's cumulative karma points, opens the Slack channel.
        ii. [S2] Peer sends a message in the channel expressing their intent to view an engineer's karma points.
        iii. [S3] The KarmaTracker bot in the Slack channel recognizes the request.
    d. Subflows: Bot Response:
        i. [S1] The bot identifies the engineer whose karma points the peer wants to view based on the message
        ii. [S2] The bot retrieves the cumulative karma points for the identified engineer
        iii. [S3] The bot displays the engineer's cumulative karma points to the peer in the Slack channel, indicating the total points achieved by the engineer.
    e. Alternative Flows:
        i. [E1] Engineer Not Found: If the bot cannot identify the engineer mentioned in the peer's request, it will inform the peer that the engineer's name was not recognized.
        ii. [E2] The bot may suggest asking for the engineer's name again or provide instructions on how to properly request an engineer's karma points.



Process I Deliverable

Notes from most recent Scrum meeting (from each team mate)
<u>Most recent meeting: 10/12</u>
<u>Topic: finishing work for PM2</u>

Morgan

- Since last meeting, Aiden and I have worked on the requirements workshop together and completed the functional and non-functional requirements.
- Also had a chance to work on the tasks for the functional requirements
- Agreed to take on a use case, will be out of town but will finish it before the Friday deadline
- Clear to do: 1 use case for PM2

Aiden

- Since last meeting, Morgan and I started to work on PM2
- Aim to complete one example use case, use case diagram and estimated point allocation for 10 tasks
- Currently, don't have any blockers or anything in the way

Vatsa

- Since last meeting, I've been working on two example use cases, and thinking about the overall functionality of KarmaTracker.
- We divided the work equally based on what each of us wanted to do.
- No blockers, and have been able to finish my work before the deadline

Eunice

- Since last meeting, I've been working on the requirements workshop remaining bits
- We decided to divide the work, therefore I committed to fulfill functional tasks, risks, and user stories before the deadline.
- We then chose to divide the requirement analysis and I focused on one of the use cases.