

Retrospective Sprint 4 of Group AttackFlow1

Building a dataset of real-world cyber-attacks with Attack Flow

Team members:

Se Jin Yoon: a1706219

Ting-Wei Chin: a1782423

Faisal Hawsawi: a1822781

Lina Nehme: a1802697

Ran Qi: a1675122

Joseph Toubia: a1753547

Zemin Wong: a1780385

Jixing Ye: a1798631

Yu Zheng: a1739446

Snapshots (Group):

1. Client meeting: *Wednesday-18-10-2023.*
2. Retrospective meeting: *Thursday-19-10-2023.*

I attended both the above meetings with team members and the client.

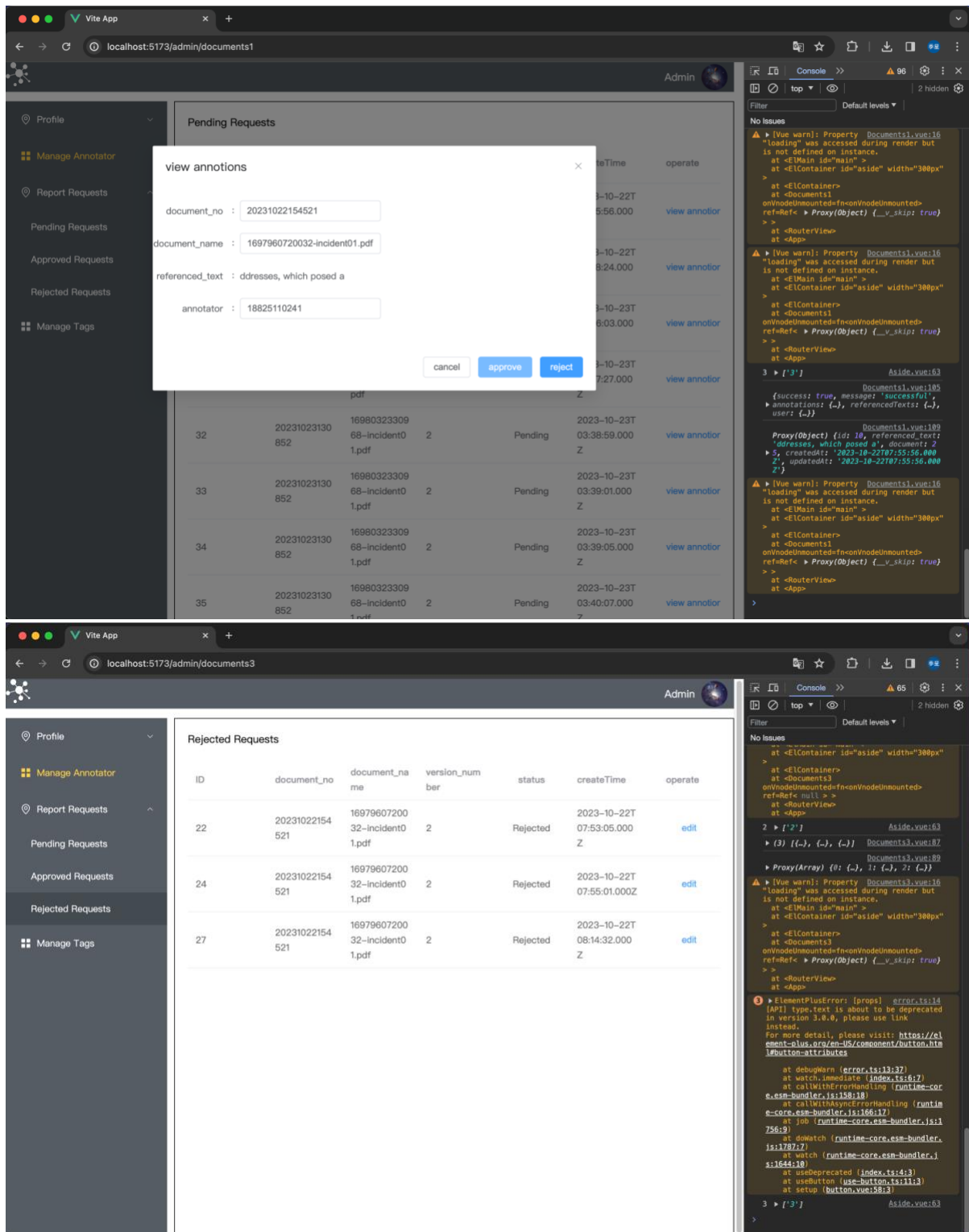
*For snapshots see the end of this report

Comment on your progress this sprint (Individually Written)

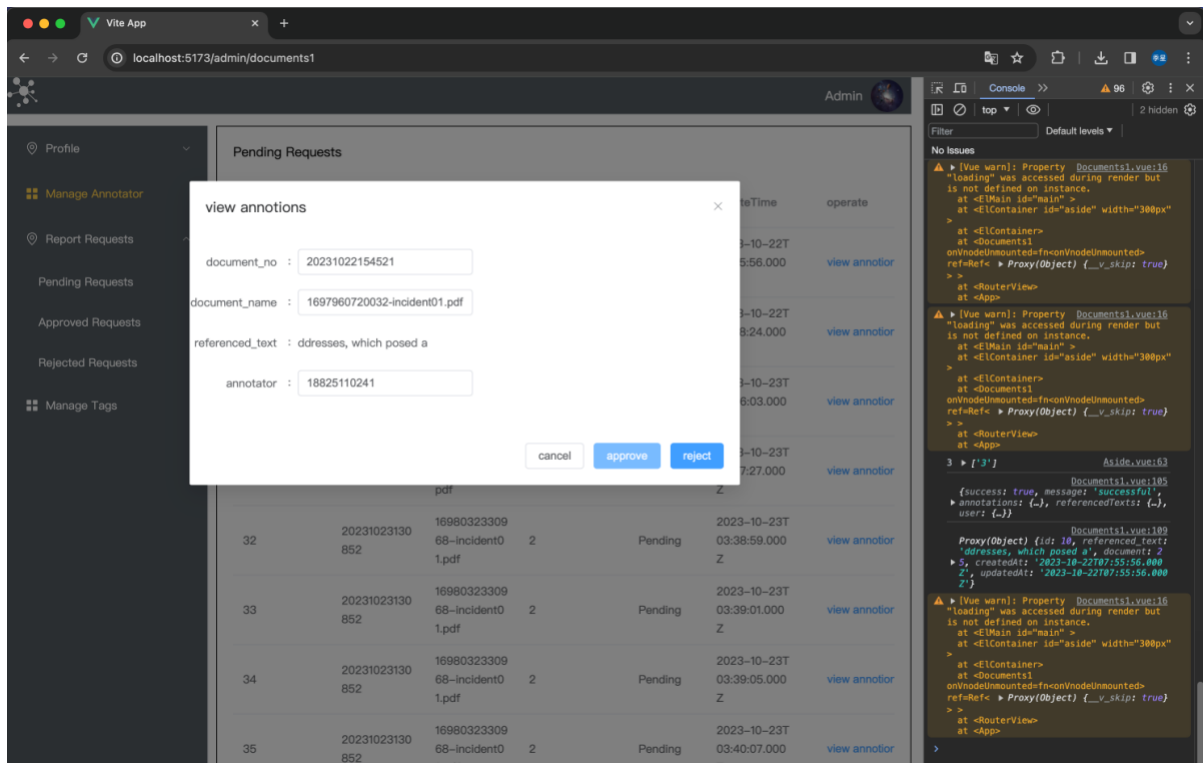
What I did:

Since our last update, I've been deep in the trenches with Yu Zheng, and I'm pleased with the progress we've made. Here's a brief overview:

Implemented an Administrative Review Feature: Admins can now oversee and moderate user uploads, marking them as "Pending," "Approved," or "Rejected."

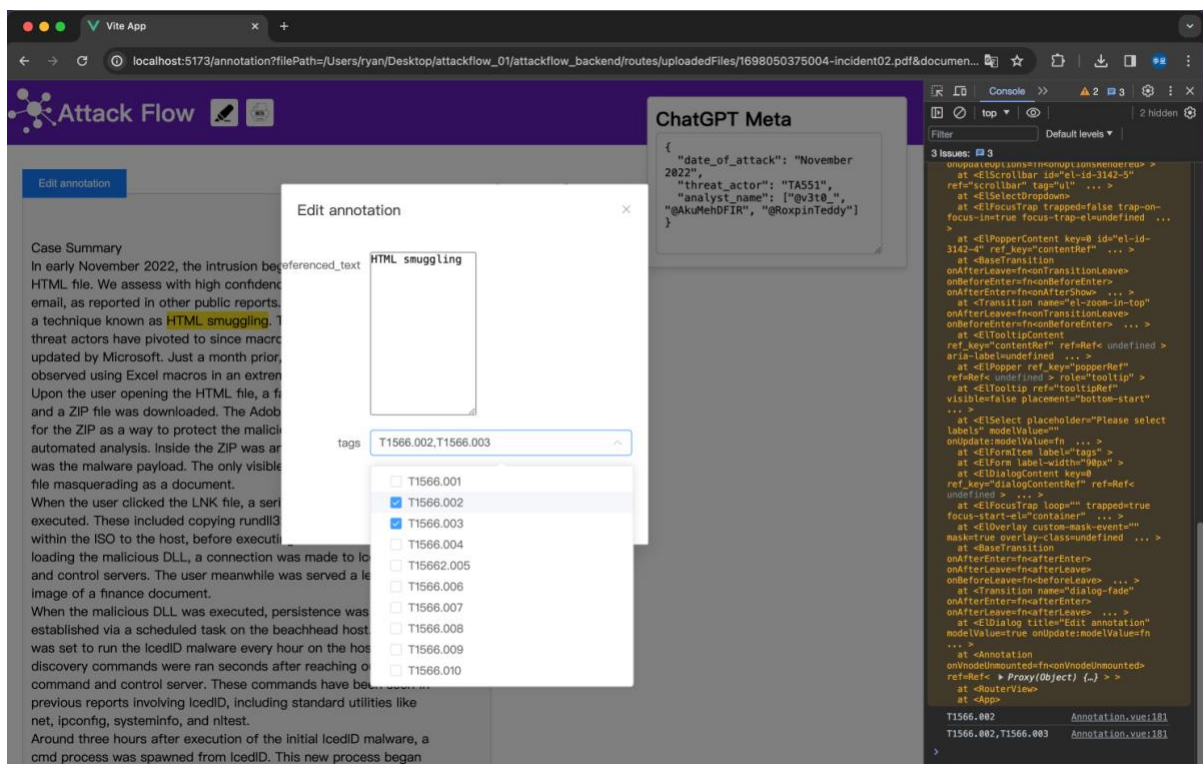


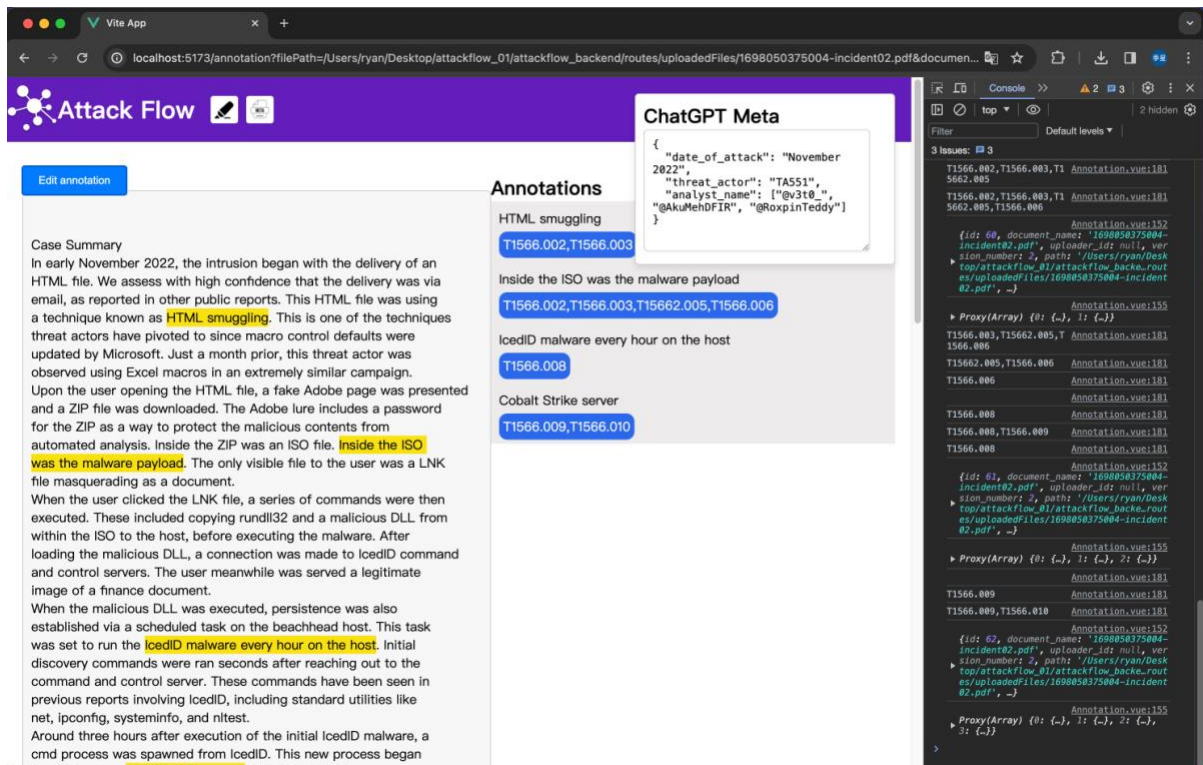
Added User Profile Modification Capabilities: Users now have the flexibility to update their email addresses, usernames, and other personal details.



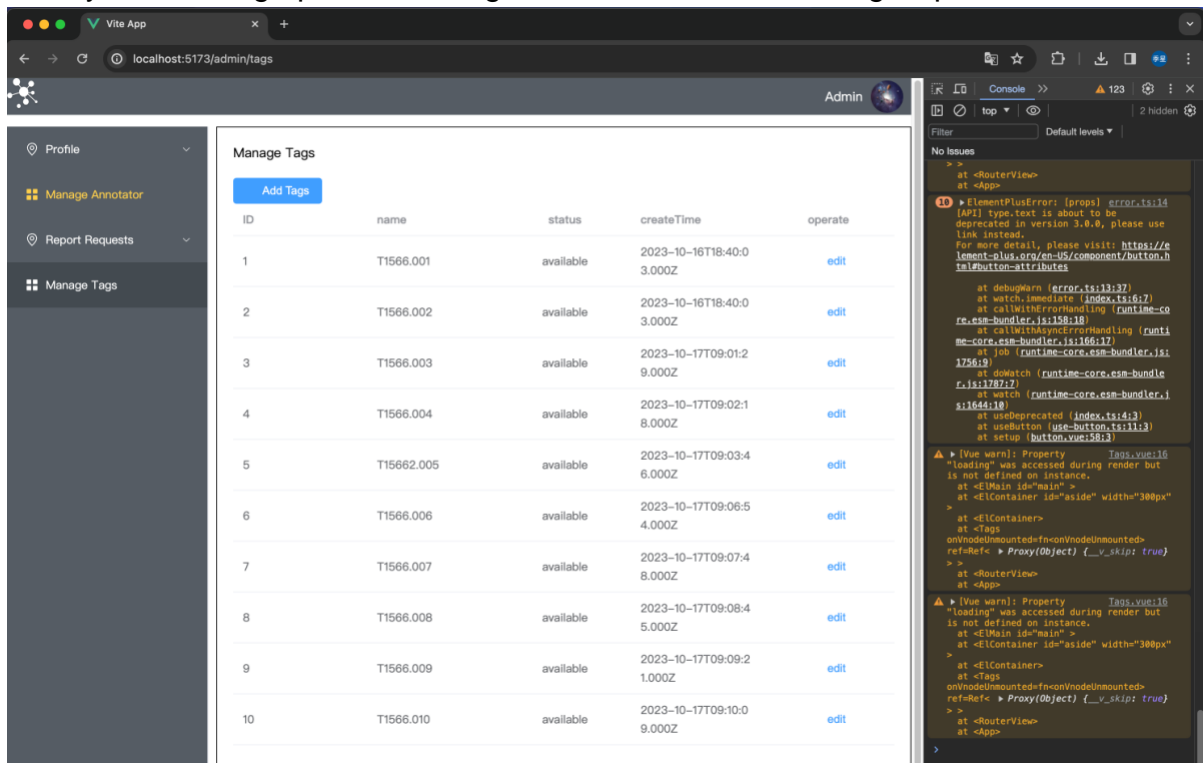
Enhanced the Annotation Interface: Incorporated a feature in the right panel of the Annotation interface to display tags added to specific fields.

Introduced a Dynamic Document Processing Tool: Once a document is uploaded in the Annotation interface, the system leverages the ChatGPT API to generate files in JSON format.



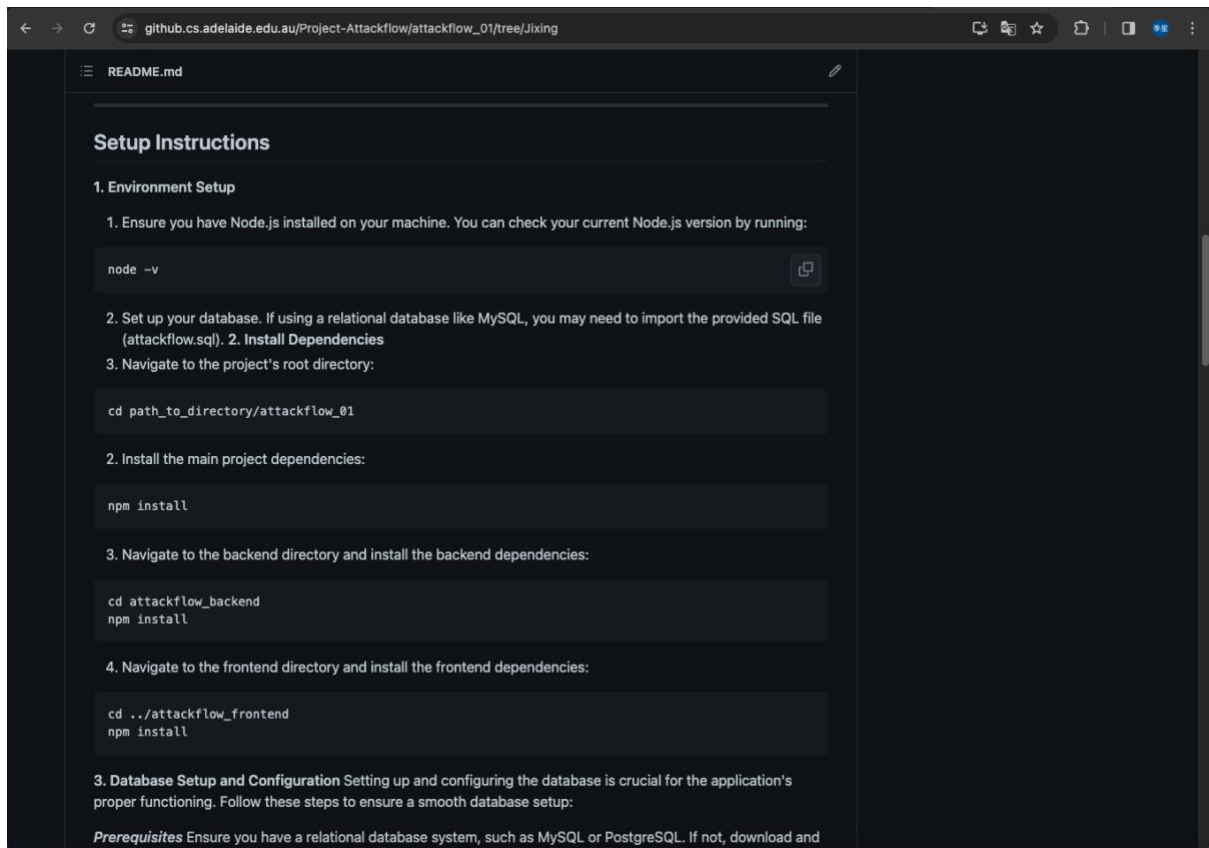


Integrated Editable Predefined Tags in Admin Interface: Administrators can now modify and manage predefined tags to better cater to evolving requirements.



Resolved Multiple Known Issues: Undertook diligent debugging to identify and rectify several known bugs, enhancing the platform's stability and performance.

Enriching the README.md:



Discussed with Jin to merge his work on keywords search:



In conclusion, the advancements observed in the current phase of development represent a synthesis of rigorous technical exploration and the application of user-centric design principles. Our endeavors, manifested through the integration of iterative feedback and the meticulous refinement of system functionalities, underscore our commitment to pioneering a platform that stands at the confluence of functionality and academic rigour. As we navigate the complexities of cybersecurity, it becomes imperative to acknowledge the continual need for evolution, informed by both empirical evidence and theoretical underpinnings. The present state of our project, while commendable, serves as a foundation for more profound scholarly inquiry and technological innovation. Each progressive step we undertake is not merely a testament to our immediate objectives but an embodiment of our overarching academic and research-driven aspirations. Looking forward, I anticipate our continued efforts will further bolster the platform's academic and practical contributions to the field.

What went well in the sprint (Individually Written)?

- **Clarity of Role:** The tasks were distinctly outlined for me. This clear demarcation of duties eliminated any ambiguity, allowing me to channel my efforts efficiently.
- **Integration of Frontend and Backend:** Managing both frontend and backend allowed me to ensure seamless integration. It not only sped up the development process but also minimized potential integration bugs.
- **Adaptive Approach:** Facing the pivot from "using chatgpt to extract techniques" to "using chatgpt to extract metadata only", I quickly adjusted my coding and integration techniques. This adaptability ensured we met client requirements without compromising on delivery timelines.
- **Technological Leverage:** I utilized a range of tools and frameworks to streamline the development process. This ensured that our product was not only functionally robust but also scalable and maintainable.
- **Collaborative Dynamics:** Working closely with my code mate Michael, we established a swift feedback loop. This collaborative spirit ensured that any hitches were immediately addressed, leading to a more refined end product.
- **Feedback Implementation:** Post-client meetings, I was proactive in implementing the feedback on the coding side. This not only improved the product's efficiency but also its alignment with the client's vision.
- **Challenge Management:** There were instances when we faced unexpected bugs or integration issues. Leveraging my dual-role expertise, I was able to troubleshoot and resolve these challenges, ensuring the sprint stayed on track.
- **Continuous Learning Curve:** This sprint was a journey of continuous learning for me. Whether it was adopting a new framework or devising an innovative solution to a persistent problem, I continually expanded my skill set.
- **Forward Thinking:** As we approached the end of the sprint, I already began strategizing for potential optimizations and refinements for the next phase, ensuring we remain ahead of the curve.
- **Client Interaction:** Engaging with the client provided me with firsthand insights into their expectations and vision. This direct communication allowed me to tailor my coding approach more aligning with the client's objectives.

What could be improved (Individually Written)?

- **Standardization and Quality Assurance:** The importance of consistent code quality can't be overstated. This would not only make the codebase more maintainable but would also foster better collaboration among team members.

- **Version Control and Branch Management:** While our commit system was functional, introducing a more descriptive and structured versioning system (e.g., v1, v1.2) could further optimize our development workflow. This, combined with rigorous code reviews before merging into the main branch, would ensure that only thoroughly vetted features get integrated, reducing potential bugs and inconsistencies.

- **Robust Testing Framework:** Our testing methods need refinement. Instituting a comprehensive testing plan would be pivotal. This would involve:

- **Fault Tolerance:** Ensuring the system gracefully handles errors, leading to a more reliable and robust application.

- **Unit Testing:** The use of assertions in unit tests can catch a majority of the bugs during the development phase itself, saving time and effort in the long run.

- **Automated Testing:** Implementing automated testing tools would enable us to detect issues in real-time, leading to quicker resolutions and a more efficient development cycle.

- **Feedback Loop with Non-Technical Stakeholders:** While we maintained a strong feedback mechanism within the technical team, broadening this loop to include non-technical stakeholders (like UI/UX designers or business analysts) earlier in the process might bring different perspectives and catch potential user experience or business logic issues.

- **Enhanced Collaboration Tools:** Adopting tools that foster better communication and task management, like Trello or Jira, could streamline our workflows and ensure everyone is aligned on priorities.

- **Documentation:** While our focus was rightly on development, we sometimes overlooked comprehensive documentation. Prioritizing this would help onboard new team members faster and serve as a reference during future phases of the project.

In essence, while we've achieved significant milestones, these improvements could streamline our processes, bolster our product quality, and foster a more cohesive team environment.

Snapshots:

Snapshot Week 10 of Group AttackFlow1

Building a dataset of real-world cyber-attacks with Attack Flow

Se Jin Yoon: a1706219

Ting-Wei Chin: a1782423

Faisal Hawsawi: a1822781

Lina Nehme: a1802697

Ran Qi: a1675122

Joseph Toubia: a1753547

Zemin Wong: a1780385

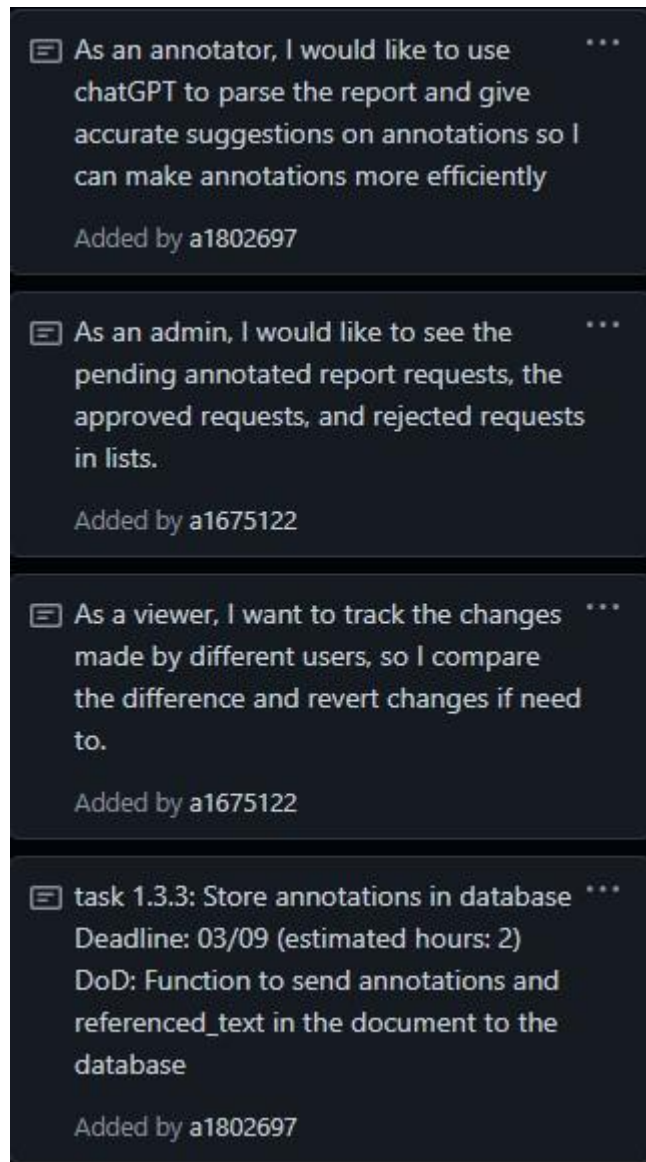
Jixing Ye: a1798631


Yu Zheng: a1739446


October 16, 2023

Product Backlog

Figure 1: Product Backlog (Sprint 5) 1 of 5



 task 1.3.4: Retrieve data from database to***
website
Deadline: 04/09 (estimated hours: 2)
DoD: Implement function to retrieve data from database (version control, document, annotations) for users to view on website
Added by a1802697

 task 4.2: implement different user roles in***
the database
DoD: admin/annotator roles added to the database schema
Estimated Due Date: 1/Sep/23
Added by a1675122


 task 3.1: implement version control ***
function in front end
DoD: a javascript function that allows users to see different versions of annotation of the same document
Estimated Due Date: 4/Sep/23
Added by a1675122

Figure 2:
Product
Backlog
(Sprint 5)
2 of 5

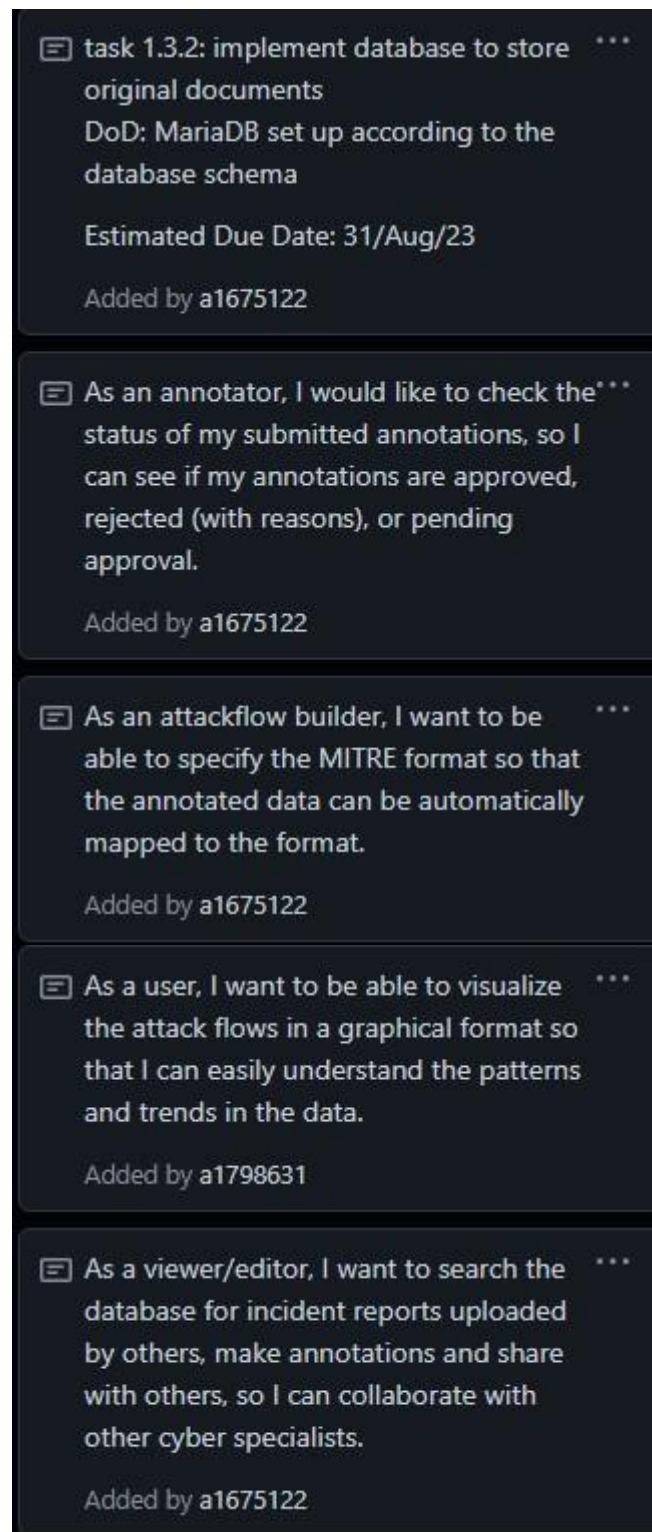


Figure 3: Product Backlog (Sprint 5) 3 of 5

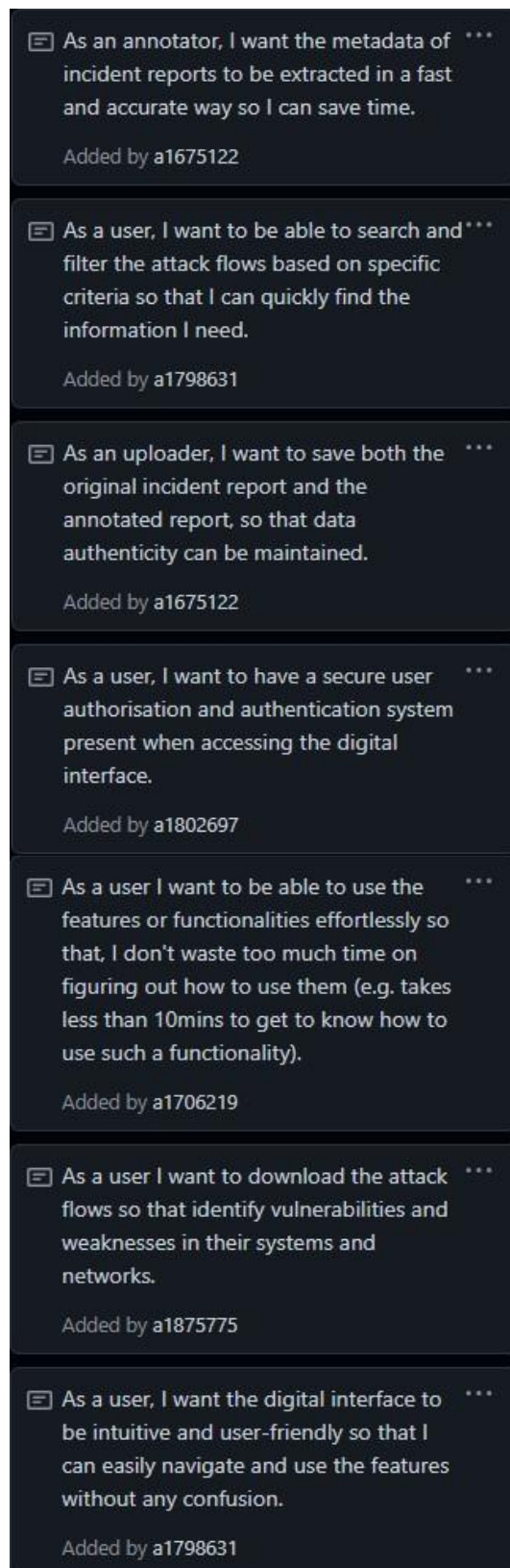


Figure 4: Product Backlog (Sprint 5) 4 of 5

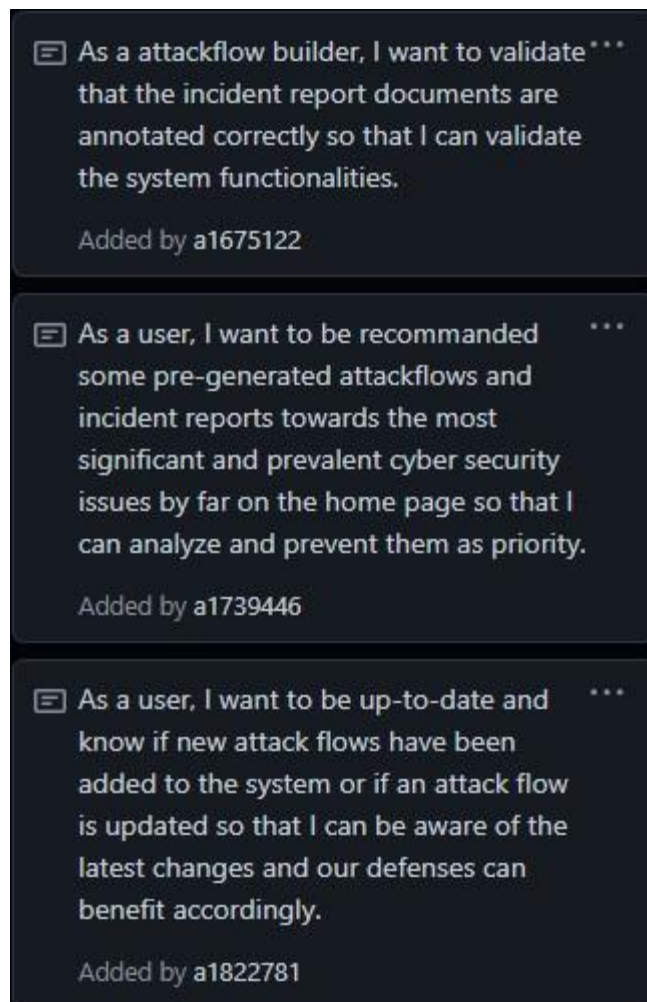


Figure 5: Product Backlog (Sprint 5) 5 of 5

Task Board

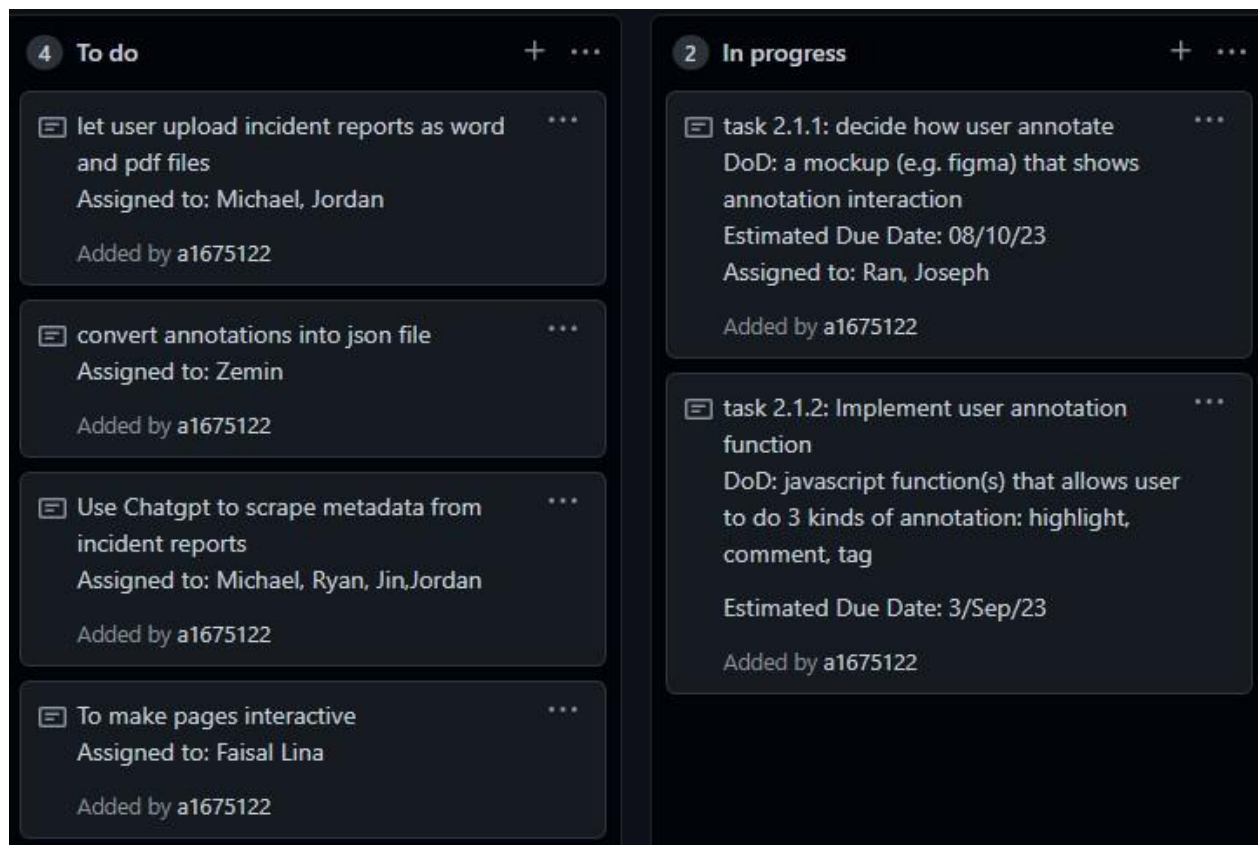
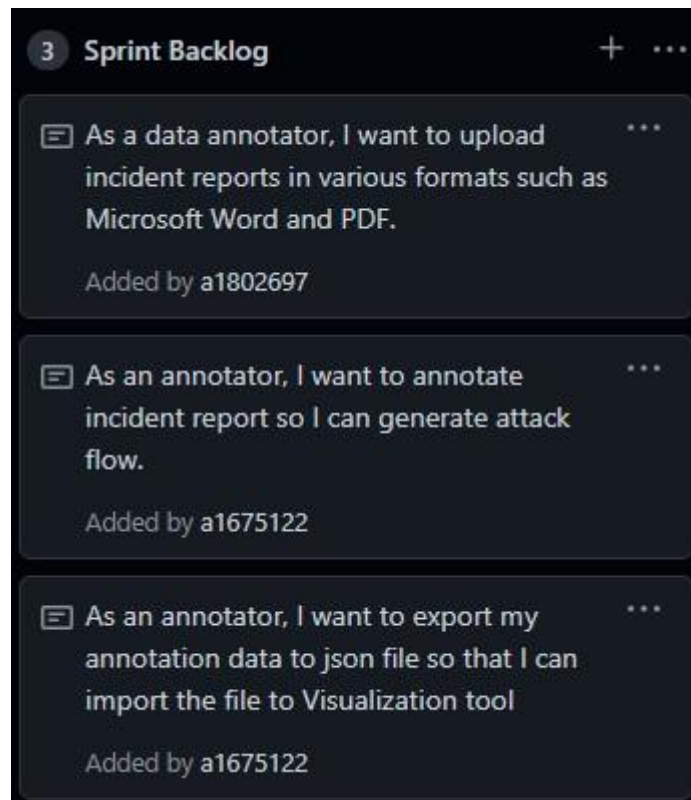


Figure 6: Task board (Sprint 5)

Sprint Backlog

Figure 7: Sprint Backlog (Sprint 5)



User

Stories

Nearing the end of the project and some user stories have become imperative to complete. One such task is to improve the annotation process with ChatGPT, which was ruled out last snapshot but we are trying to utilise it as much as possible given the short time left. We will be adding the pre-defined tags to the annotation page as well as a highlighting function to help the annotator. Finally, we need to implement the annotator functionality, which involves viewing and editing profile, and incident report management including pending requests, rejected requests and approved requests, and Administrator functionality including annotator management and incident report management similar to the annotator functionality.

- “As a data annotator, I want to upload an incident report and save it so that I can further edit the file.”

– Related tasks:
task 1.2: upload function

task 1.3.2: implement database to store original documents

- “As an annotator, I want to annotate incident report so I can generate attack flow.”

– Related tasks:

task 2.1.1: decide how user annotate

task 2.1.2: implement user annotation function

task 2.1.3: integrate ChatGPT to parse report and give accurate annotation suggestions

- “As an admin, I would like to see the pending annotated report requests, the approved requests, and rejected requests in lists.”

– Related tasks:

task 1.3.3: Store annotations in database

task 4.2: Implement different user roles in the database

Definition of Done

The individual DoD of each task can be seen on the screenshot of the task board.
In summary, the goals are:

- Testing the integrated backend functions works as we expected.
- Code is well-documented and adheres to our coding standards.
- Explain and ensure that all the scrum members understand the specifics of the tasks and how to construct the implementation.

Summary of Changes

This snapshot will still focus on the upload and annotate functions but the focus will now be on implementing the 'keywords/tags' function as well as the highlight function. ChatGPT, however, will be implemented as much as possible without compromising the end goal of the project which is a backflip on last snapshot. The annotator and administrator user roles are now also in the main focus.

- **Annotation function to be completed including the ability to view and edit profile.** We will continue with the annotate functionality including how to annotate? How to use keywords/tags? How to use predefined keywords/tags? How to incorporate keywords from Mitre Att&ck? Must identify tags and be able to manually search from drop down menu to define relevant tag as well as profile functionality.
- **This snapshot re-enstates the major role of ChatGPT in parsing the report and providing accurate annotation suggestions.** We have now decided to utilise the ChatGPT as much as possible in our web application to extract metadata and as much information as possible with regard to extracting techniques.
- **Administrator functionality to be implemented.** The administrator functionality needs to be implemented including annotator management (delete annotator from the database and else) incident report management (pending request, approved request, rejected request).

Snapshot Week 11 of Group AttackFlow1

Building a dataset of real-world cyber-attacks with Attack Flow

Se Jin Yoon: a1706219

Ting-Wei Chin: a1782423

Faisal Hawsawi: a1822781

Lina Nehme: a1802697

Ran Qi: a1675122

Joseph Toubia: a1753547

Zemin Wong: a1780385

Jixing Ye: a1798631

Yu Zheng: a1739446

October 23, 2023

Product Backlog

Figure 1: Product Backlog (Sprint 6) 1 of 5

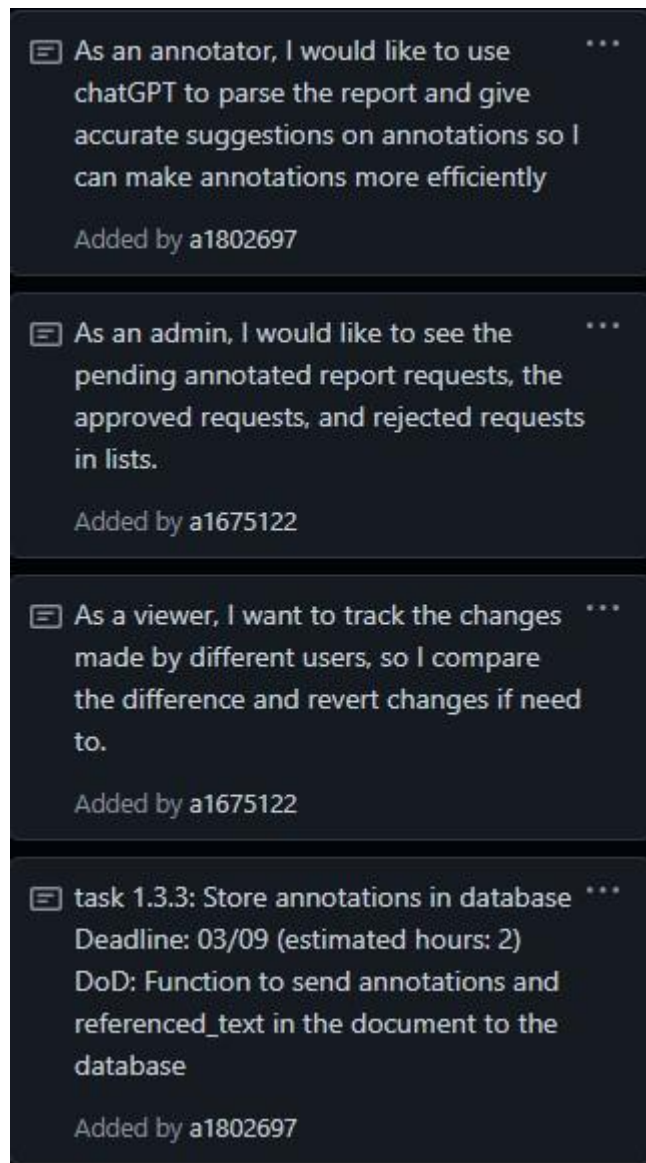




Figure 2: Product Backlog (Sprint 6) 2 of 5

 task 1.3.4: Retrieve data from database to*** website
Deadline: 04/09 (estimated hours: 2)
DoD: Implement function to retrieve data from database (version control, document, annotations) for users to view on website
Added by a1802697

 task 4.2: implement different user roles in*** the database
DoD: admin/annotator roles added to the database schema
Estimated Due Date: 1/Sep/23
Added by a1675122





 task 3.1: implement version control function in front end ***
DoD: a javascript function that allows users to see different versions of annotation of the same document
Estimated Due Date: 4/Sep/23
Added by a1675122

Figure 3: Product Backlog (Sprint 6) 3 of 5

 task 1.3.2: implement database to store original documents ***
DoD: MariaDB set up according to the database schema
Estimated Due Date: 31/Aug/23
Added by a1675122

 As an annotator, I would like to check the status of my submitted annotations, so I can see if my annotations are approved, rejected (with reasons), or pending approval.
Added by a1675122

 As an attackflow builder, I want to be able to specify the MITRE format so that the annotated data can be automatically mapped to the format. ***
Added by a1675122

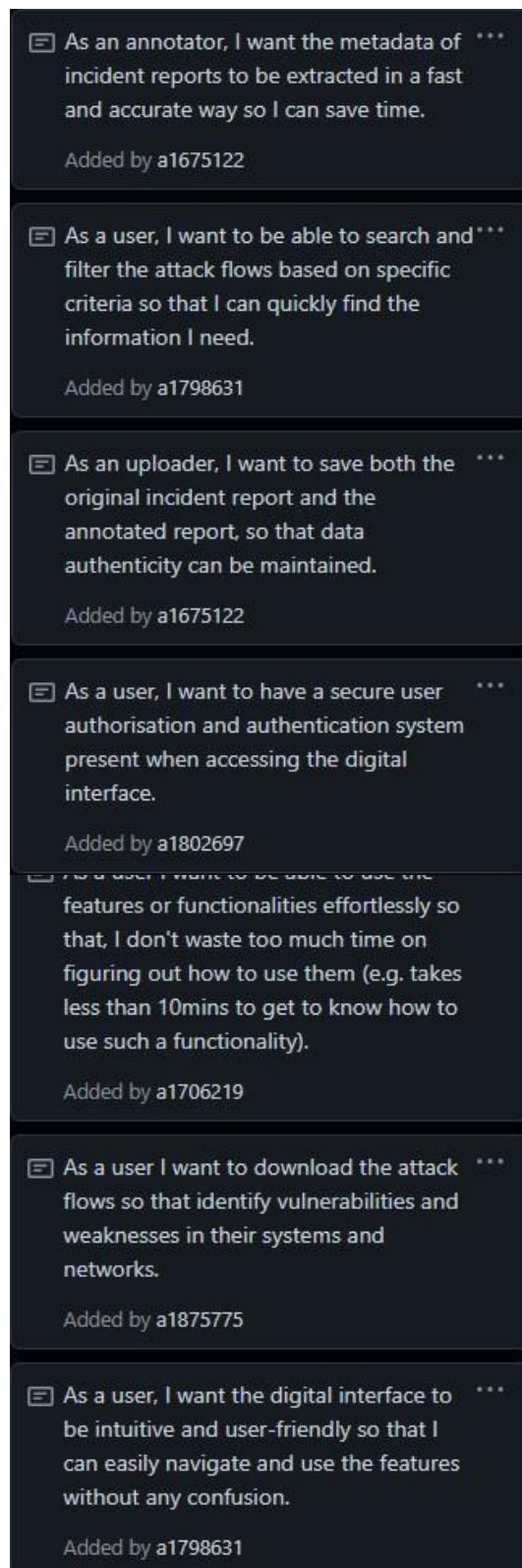


Figure 4: Product Backlog (Sprint 6) 4 of 5

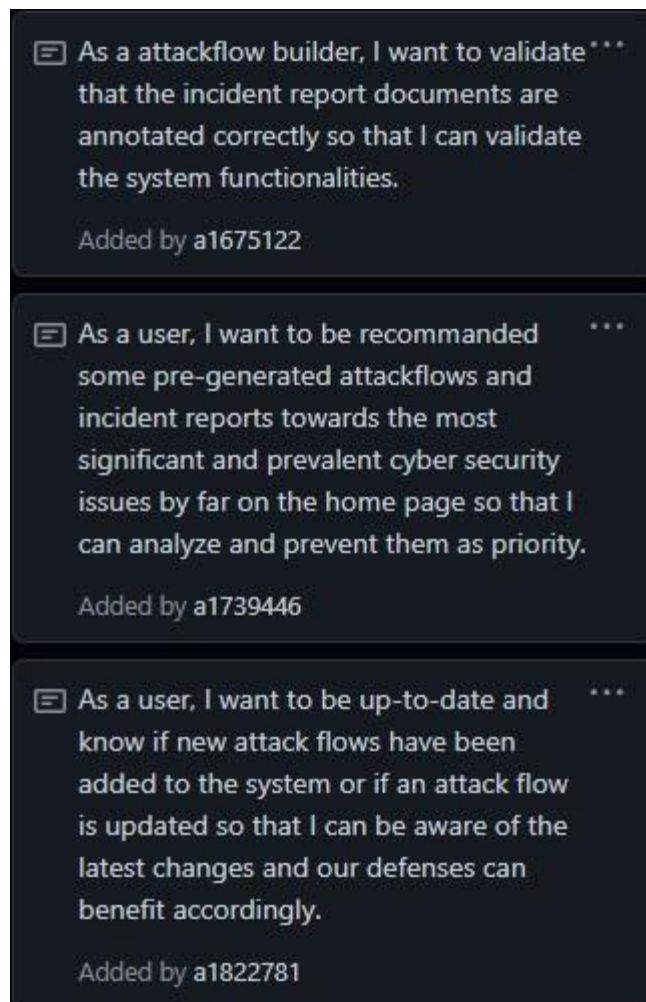


Figure 5: Product Backlog (Sprint 6) 5 of 5

Task Board

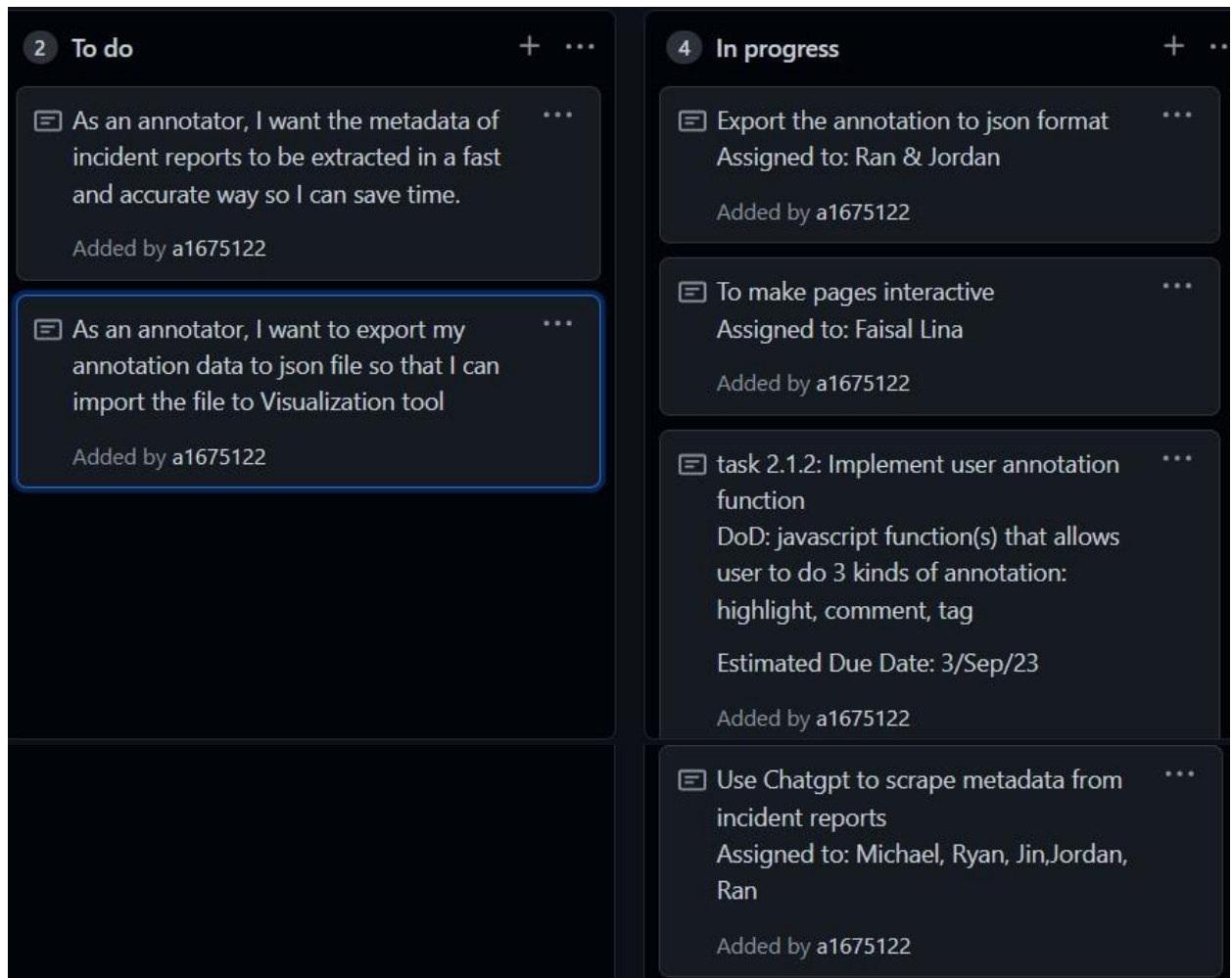


Figure 6: Task board (Sprint 6)

Sprint Backlog

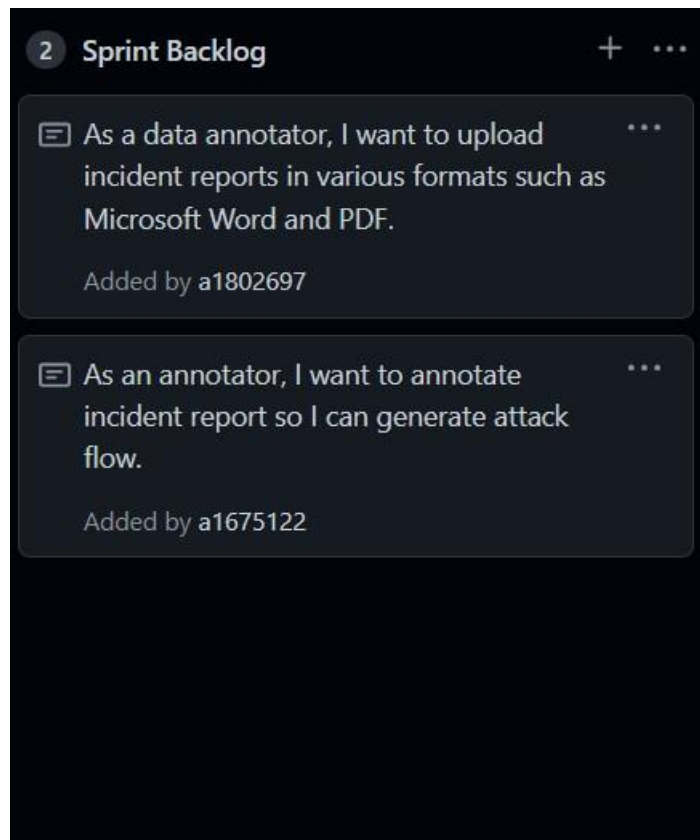


Figure 7: Sprint Backlog (Sprint 6)

User Stories

In the final weeks now and all our focus and resources are on the annotation process with ChatGPT, we are trying to utilise it as much as possible given the time frame. We have added the pre-defined tags to the annotation page as well as a highlighting function to help the annotator. Finally, we need to implement the annotator functionality including incident report management pending requests, rejected requests and approved requests, and Administrator functionality including annotator management and incident report management as with the annotator functionality.

- “As a data annotator, I want to upload an incident report and save it so that I can further edit the file.”
 - Related tasks:
 - task 1.2: upload function
 - task 1.3.2: implement database to store original documents
- “As an annotator, I want to annotate incident report so I can generate attack flow.”
 - Related tasks:
 - task 2.1.1: decide how user annotate
 - task 2.1.2: implement user annotation function
 - task 2.1.3: integrate ChatGPT to parse report and give accurate annotation suggestions
- “As an admin, I would like to see the pending annotated report requests, the approved requests, and rejected requests in lists.”
 - Related tasks:
 - task 1.3.3: Store annotations in database
 - task 4.2: Implement different user roles in the database

Definition of Done

The individual DoD of each task can be seen on the screenshot of the task board. In summary, the goals are:

- Testing the integrated backend functions works as we expected.
- Code is well-documented and adheres to our coding standards.

- Explain and ensure that all the scrum members understand the specifics of the tasks and how to construct the implementation.

Summary of Changes

The upload and annotate functions are still under the spotlight to get the best possible functionality and we are still trying to get the most functionality out of ChatGPT. Both Administrator and Annotator user roles are still the main focus as suggested above, however, some functionality has been implemented.

- **Annotation function to be completed.** We will continue with the annotate functionality including how to annotate? How to use keywords/tags? How to use predefined keywords/tags? How to incorporate keywords from Mitre Att&ck? Must identify tags and be able to manually search from drop down menu to define relevant tag as well as profile functionality.
- **This snapshot re-enstates the major role of ChatGPT to extract metadata.** We have now decided to utilise the ChatGPT as much as possible in our web application to extract metadata.
- **Administrator functionality to be implemented.** The administrator functionality needs to be implemented including annotator management (delete annotator from the database and else) incident report management (pending request, approved request, rejected request).