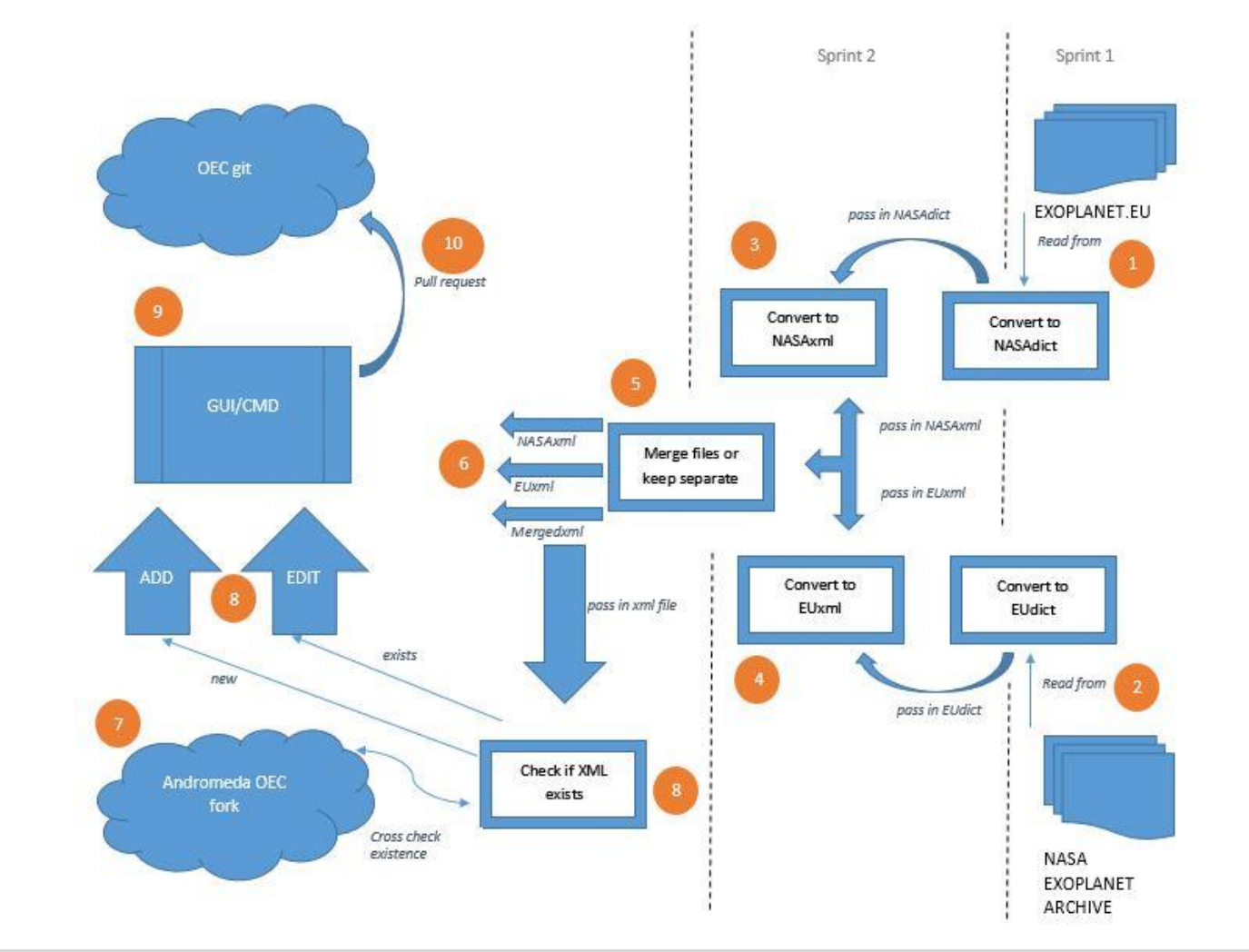
Team 15 – Andromeda

Deliverable 4: Report

[Sprint 2]

Table of Contents:

System Design



Subsystems:

1. Read from Exoplanet.eu and convert exoplanet data into a base dictionary
2. Read from NASA EXOPLANET ARCHIVE and convert exoplanet data into a base dictionary
3. Convert the base EU dictionary to valid OEC xml file
4. Convert the base NASA dictionary to valid OEC xml file
5. Merges XML files if they refer to the same system, or passes two different if they are not of the same system
6. Return valid OEC xml file to compare with subsystem 7 through subsystem 8
7. Fork the OEC database
8. Compare and denote if files from subsystem 6 exist in subsystem 7, thereby denoting them as an edit to an existing file or a new addition
9. The subsystem representing the backend and frontend of the CMD or GUI user interface, giving user access to designated features
10. If the user accepts an add or an edit create the corresponding pull request to the OEC

Completed Components of the System

Sprint 1:

- Read from Exoplanet.eu and convert exoplanet data into a base dictionary
- Read from NASA EXOPLANET ARCHIVE and convert exoplanet data into a base dictionary
- Subsystem 1 and 2 demo in driver.py

Sprint 2:

- Completed subsystems 3, 4, 8, 7 with major parts of subsystem 9 and 10
  - Sub-system3 and 4 utilized the same code, but is regarded as different subsystems to allow modularity of code for the developers and well as to allow additional features for the client such as: choose either EU or NASA or both for update
  - Sub-system 7 and 8 utilizes for the major part existing GIT commands and functionalities so as to complete the subsystems efficiently and so as to not “re-invent the wheel”
  - Sub-system 9 and 10 again utilized existing GIT commands and Shell script to allow users to have freedom of choice between updates in a meaningful and efficient way though interactions with a CMD interface.

Product Backlog

Cost: DH (Developer Hours)
Priority: 1 – lowest, 5 – highest

- **User Story # 1:** As Professor Smith, the client, I want the application to run on Unix environment by using software and packages that are UNIX compatible
  - Cost: 0 DH & Priority: 5

- **User Story #2:** As Professor Smith, the client, I want the application to fetch data from the NASA and EU catalogues and I want it to be organized by each star system.
  - Cost: 5 DH & Priority: 5

- **User Story #3:** As Professor Smith, the client, I would like the data that can representable in the OEC to be retrieved from the two catalogues.
  - Cost: 25 DH & Priority: 5

- **User Story #5**: As Professor Smith, the client, I want the application to produce data files in XML format that are usable as OEC XML files.
  - Cost: 25 DH & Priority: 4

- **User Story #6:** As Professor Smith, the client, I want the application to check for new data in the EU and NASA catalogues.
  - Cost: 20 DH & Priority: 3

- **User Story #7:** As Professor Smith, the client, I want the application to work on a fork version of the OEC master repository on GitHub.
  - Cost: 8 DH & Priority: 4

- ****User Story #7.5:** As Professor Smith, the client, I want the application to check if the new XML file created already exists in the forked OEC repository and report an edit or new addition respectively.
  - Cost: 20 DH & Priority: 4

- ** **User Story # 7.6:** As Professor Smith, the client, I want the application to let me manually choose which new files should go into the repository.
  - Cost: 15 DH & Priority: 4

- **User Story #8:** As Professor Smith, the client, I want the application to receive daily updates, if any, from the NASA and EU catalogues.
  - Cost: 7 DH & Priority: 3

- **User Story #9:** As Professor Smith, the client, I would like to be notified of all updates available when I log into the application
    - Cost: 13 DH & Priority: 3

- **User Story #10:** As Professor Smith, the client, I would to view the log of changes where I can analyze and approve of the generated updates.
    - Cost: 10 DH & Priority: 2

- **User Story #11:** As Alice, the graduate student, I want the application to let me view the permissions for updates and the log.
    - Cost: 7 DH & Priority: 2

- **User Story #12:** As Professor Smith, the client, I would like the application to allow human intervention by allowing myself to manually edit errors by modifying the updates directly brought through by the data from the other catalogues
    - Cost: 18 DH & Priority: 3

- **User Story #13:** As Professor Smith, the client, I would like the application to have a log of all changes made to my repository where I can select a certain date to see the changes made in each update on that day.
    - Cost: 10 DH & Priority: 2

Sprint Plan

- This sprint will end on November 14th 2016.

Sprint Backlog for Sprint #2:

[#5] As Professor Smith, the client, I want the application to produce data files in XML format that are usable as OEC XML files. [Cost: 20 DH & Priority: 4]

Tasks:

- To be done by: Ralph, Hajoon
- Get subsystem users input on number of planets to convert to xml, the destination folder name, and from which catalog.
- Convert the base dictionary with all the data into a dictionary of planet dictionaries that hold data for each respective key (data column)
- Convert the planet dictionaries into XML by adding the planet data to respective system xml

[#7] As Professor Smith, the client, I want the application to work on a fork version of the OEC master repository on GitHub. [Cost: 4 DH & Priority: 4]

Tasks:

- To be done by: Kelvin, Daniel
- Run a script to clone the OEC repository in a directory

[#7.5] As Professor Smith, the client, I want the application to check if the new XML file created already exists in the forked OEC repository and report an edit or new addition respectively. (Cost: 5 DH & Priority: 5)

Tasks:

- To be done by Kelvin, Daniel
- Check for new XML files in the OEC repository
- Report for changes (an edit or new addition)

[#7.6] As Professor Smith, the client, I want the application to let me manually choose which new files should go into the repository. [Cost: 6 DH & Priority: 4]

Tasks:

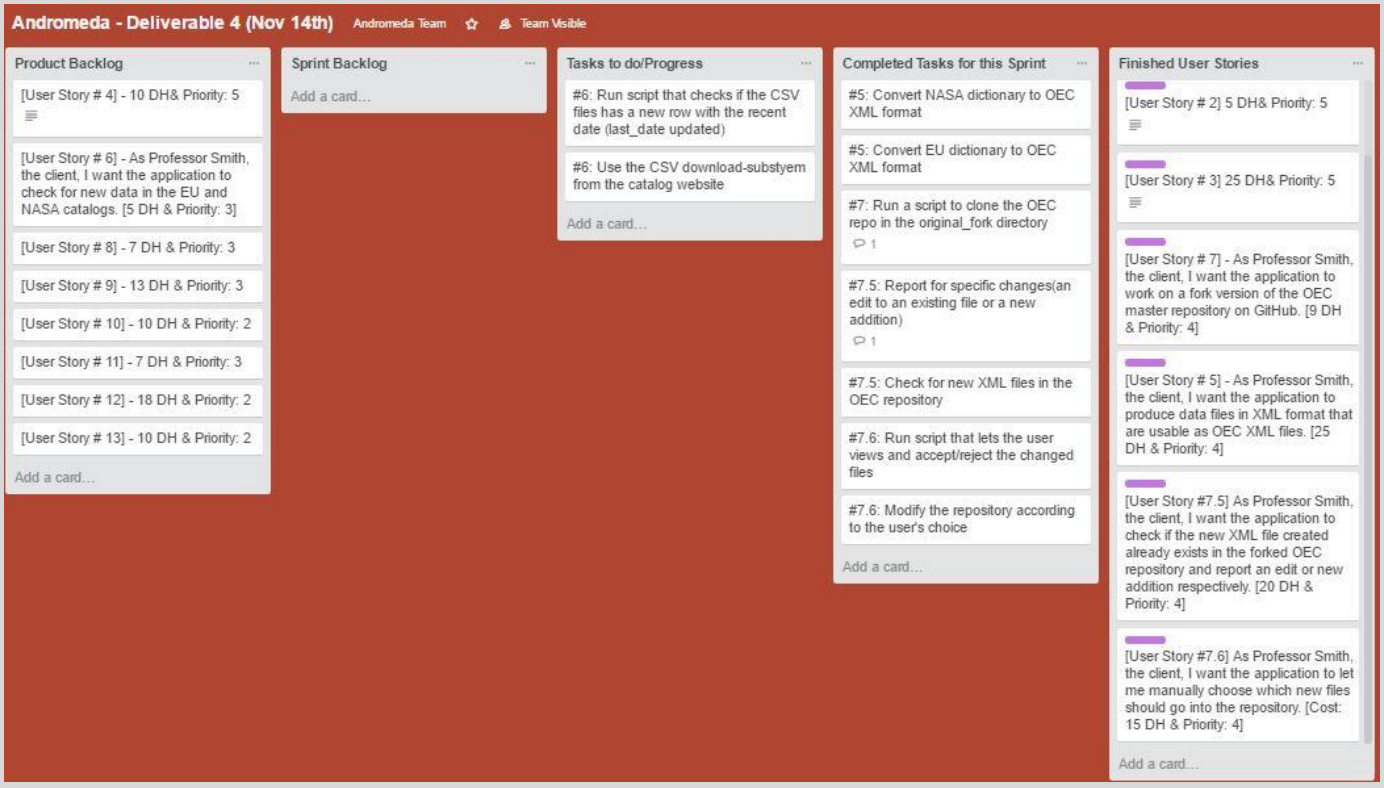
- To be done by Suhailah
- Run a script that lets the user views, accept/reject the files that are changed
    - Modify the repository as needed

Iteration Plan:

Story #	Priority Level 1 - lowest 5 - highest	Cost	User Story
5	4	20	As Professor Smith, the client, I want the application to produce data files in XML format that are usuable as OEC XML files
7	4	4	As Professor Smith, the client, I want the application to work on a fork version of the OEC master repository on GitHub.
7.5	4	5	As Professor Smith, the client, I want the application to check if the new XML file created already exists in the forked OEC repository and report an edit or new addition respectively.
7.6	4	6	As Professor Smith, the client, I want the application to let me manually choose which new files should go into the repository.

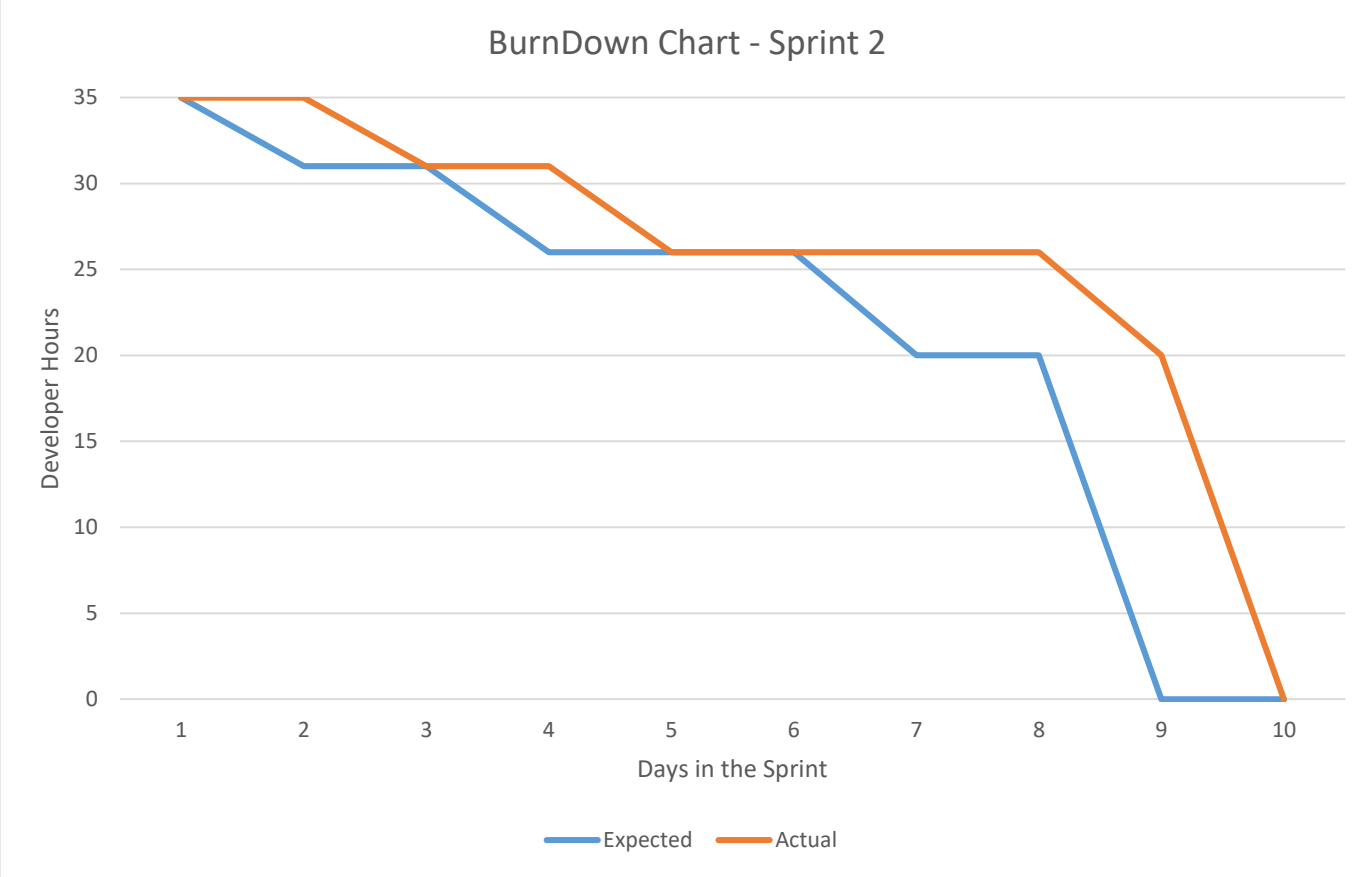
Tasks	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10
[#5] Convert NASA dictionary to OEC XML format	0	0	0	0	0	0	HC:4	RS: 2	HC: 4	RS: 2
[#5] Convert EU dictionary to OEC XML format	0	0	0	0	0	0	RS:4	RS: 2	RS: 2	0
[#7] Run a script to clone the OEC repo in the original_fork directory	0	DM:2	DM: 2	0	0	0	0	0	0	0
[#7.5] Report for specific changes(an edit to an existing file or a new addition)	0	DM: 1	0	0	0	0	0	0	0	0
[#7.5] Check for new XML files in the OEC repository	0	0	DM:2	0	KO: 2	0	0	0	0	0
[#7.6] Run script that lets the user views and accept/reject the changed files		0	0	0	0	SR: 1	SR: 2	0	0	0
[#7.6] Modify the repository according to the user's choice	0	0	0	0	0	0	0	0	SR:3	0
TOTAL HOURS [PER DAY]	0	3	4	0	2	1	10	4	9	2

Task Board:



Burn-down Chart:

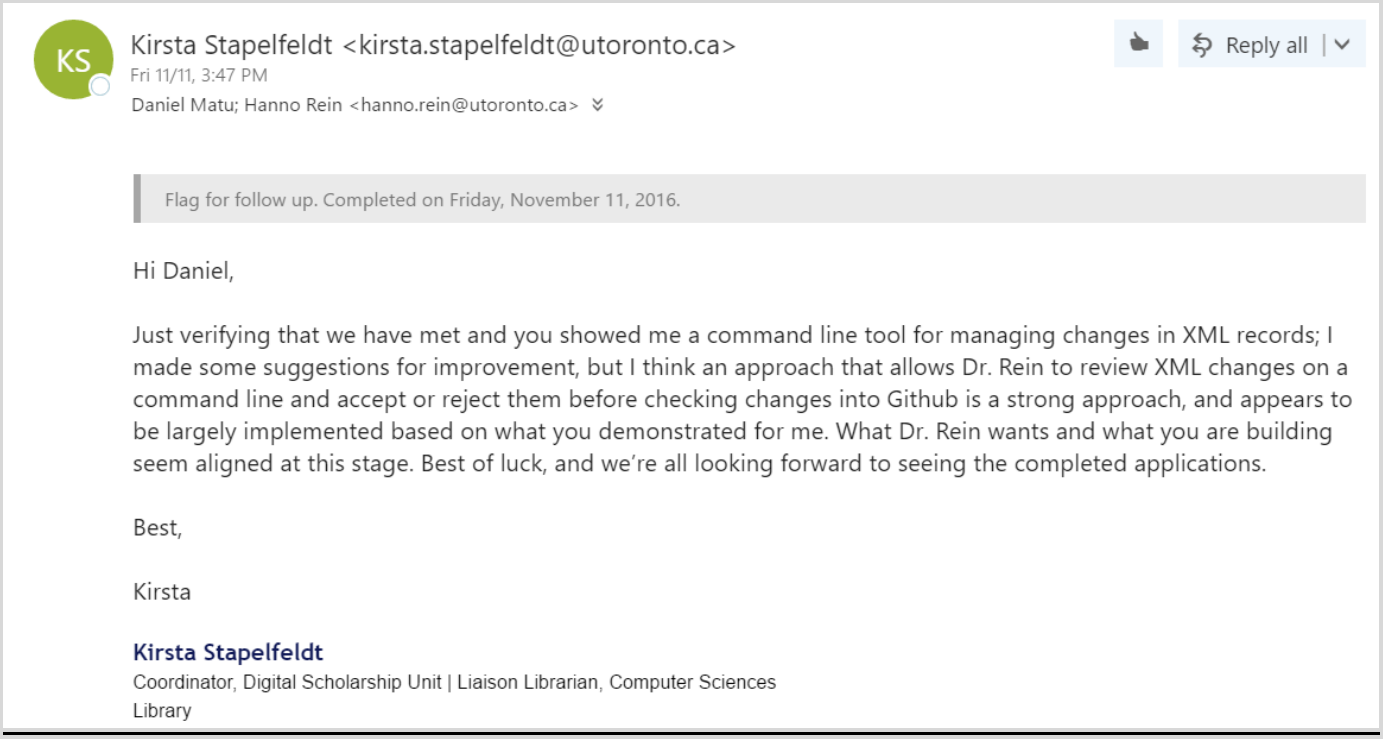
Sprint 2



System Validation Activities

We talked to one of the client's representatives Kirsta Stapelfeldt. We showed her the initial working version of the front end, which is application.py.

All of her responses are down below:



KS

Kirsta Stapelfeldt <kirsta.stapelfeldt@utoronto.ca>

Fri 11/11, 3:47 PM

Daniel Matu; Hanno Rein <hanno.rein@utoronto.ca>

Reply all

Flag for follow up. Completed on Friday, November 11, 2016.

Hi Daniel,

Just verifying that we have met and you showed me a command line tool for managing changes in XML records; I made some suggestions for improvement, but I think an approach that allows Dr. Rein to review XML changes on a command line and accept or reject them before checking changes into Github is a strong approach, and appears to be largely implemented based on what you demonstrated for me. What Dr. Rein wants and what you are building seem aligned at this stage. Best of luck, and we're all looking forward to seeing the completed applications.

Best,

Kirsta

Kirsta Stapelfeldt

Coordinator, Digital Scholarship Unit | Liaison Librarian, Computer Sciences Library

Brief Overview – State of the project

- **Transition from Deliverable 3 to Deliverable 4:**
  - From Deliverable 3, we continued to work with our ideas for the project and using the concept of Agile development, we used our experience from the past sprint to improve on our implementation and system design.
  - We realized that we were being ambitious with the user stories, so we split user stories up into more so that we can finish it (we split #7 up into 7.5 and 7.6)
  - We changed our product backlog to include the new user stories. It was decided that we need the new stories for a greater system integrity and efficient implementation.
  - We also added on to our sprint log, which includes the new stories and we also took out some stories. The reason why we took out some user stories is because we realized one of the stories was an epic and we couldn't have finished it off in this sprint.
  - We continued with our meetings every week and we are still using Slack as our communication tool.
  - A branch was created to test out and develop the conversions into planetary system XML files (user story #5).
  - We started on the front-end of the software.
- **Estimated Project Velocity: 35**
- **Actual Project Velocity: 35**
- **Did we follow with our plan?**
  - Compared to the previous deliverable, this sprint went a lot smoother. All the members finished their tasks slowly while being on time, and making sure they are doing what the client wants.
  - We had gone to the client's representative to confirm that our idea is what they want.
  - We deleted user story#4 as it was very similar to #5.
  - One of the user stories (#5) took a long time to develop but the members, who are responsible for the tasks, researched from the client's code (generate_systems_kepler.py) on the OEC master repository. They used that as reference to finish the user story.
    - We have to scrap some bit of the backend after looking at the client's code.
    - We used our backend code from the last deliverable as it helped developing the conversion subsystem. It was more efficient and had more functionality with less code.
- **How did the work done for deliverable 4 differ from deliverable 3?**
  - Since we are continuing working on the code, the work load didn't differ that much as we used the codes from last sprint. In terms of progress and end results, we got far into implementation of the software. We got started on the front-end which was helpful when visualizing the ideas and concept we had in the start of the project. There were much more to be done such as the unit tests, code reviews from all members, and the video for the inspection meeting.