Team Jelly: Deliverable 5

CSCC01

December 1st, 2016

Team 25

Members: Dennis Tra, Henry Liu, Angelina Choi, Kelly Mo, Kris Lai

TABLE OF CONTENTS

OVERVIEW OF PROGRESS FROM DELIVERABLE 4 TO DELIVERABLE 5

Estimated project velocity was initially twenty. With each deliverable, a considerable number of extra tasks were added, which slowed down the project velocity. As more and more tasks were added, team members had to increase their story points to catch up to meet the expected remaining story points. With each successive deliverable, the project velocity changed in proportion to the number of tasks and enhancements that had to be implemented.

The initial plans and the end results were seldom matching and the team did not expect it to. In the early deliverables, the team focused extensively on planning before starting code development, which lead to a lot of chaos as the coding process did not work as we had anticipated and rushed near the end of each sprint to complete our tasks. There were always several elements that had to be re-planned multiple times throughout several sprints. Solutions had to be modified, workarounds had to be reached, and though we anticipated challenges as part of our plans, there were some difficulties that the plan didn’t anticipate at all. One user story where we had to compare two database files was thought to have been finished early on. In was a deliverable after that we belatedly realized that we missed a key component of the user story in comparing binary blocks that were in certain files, which lead to the team scrambling to finish the user story we thought done a while ago. Deliverable 5 was the last deliverable, so all of the work had to be completed by the deadline.

Progress on this deliverable was significantly faster than previous deliverables in an effort to accomplish all the tasks at hand. Extra story points were pitched in by each member to finish all the tasks specified. Compared to the work done on previous deliverables, this final deliverable involved a lot more progress and the end result was a fully working project, in contrast to the separate components and pieces that were presented in previous deliverables.

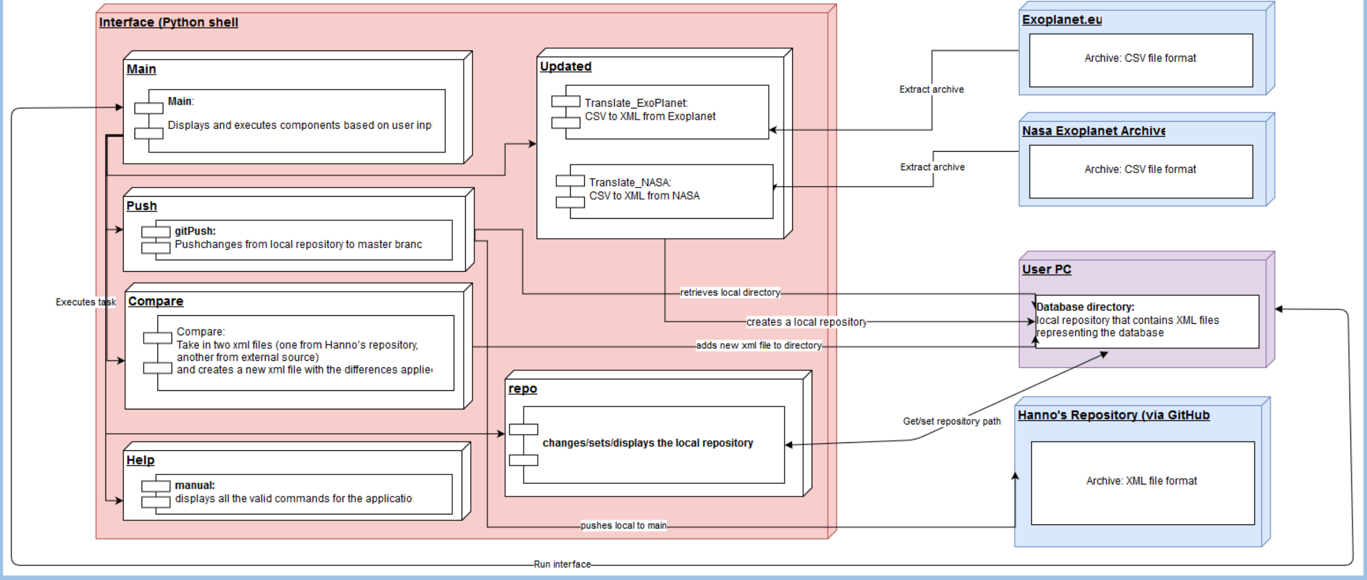
RELEASE PLAN (CUMULATIVE)

Our sprint will be one workweek long with 1 story point complete by each team member each day for a total of 20 points (Combined from all team members) each 7 days. Each story point is worth 30 minutes of development time.

We will have a release planned every week with a goal of 20 complete points each week.

Date	User stories scheduled to be completed by this release	Stories started but not expected to be completed in this release	Estimated Remaining Story Points
October 10th	Creating User Stories	N/A	96
October 17th	Planning User stories	N/A	96
October 24th	1, 2, 3, 4	4	71
October 31st	4, 5, 6, 7	4, 7	57
November 7th	4, 7, 8, 9, 10, 11	7, 11	57
November 14th	7, 11, 12, 13, 14, 15	15	40
November 21st	15, 16, 17, 18, 19	N/A	20

SYSTEM DESIGN


Interface (Python shell
Main
Main:
Displays and executes components based on user inp
Push
gitPush:
Pushchanges from local repository to master branc
Compare
Compare:
Take in two xml files (one from Hanno's repository, another from external source)
and creates a new xml file with the differences applie
Help
manual:
displays all the valid commands for the applicatio
Updated
Translate_ExoPlanet:
CSV to XML from Exoplanet
Translate_NASA:
CSV to XML from NASA
repo
changes/sets/displays the local repository
Executes task
Extract archive
Extract archive
retrieves local directory
creates a local repository
adds new xml file to directory
Get/set repository path
pushes local to main
Run interface
Exoplanet.eu
Archive: CSV file format
Nasa Exoplanet Archive
Archive: CSV file format
User PC
Database directory:
local repository that contains XML files representing the database
Hanno's Repository (via GitHub
Archive: XML file format

System Design Overview

1) Extraction: accesses and extracts records from external sources and stores them in XML format in a directory for updated systems
2) Git Component: Syncs and pushes changes from remote repository and local repository
3) Compare: Retrieves files from local repository and updated systems directory , compares differences in systems and modifies XML files in the local repository with the updates
4) Interface: Parses user input commands and runs Extraction, Git, Compare functions accordingly
5) Display: This was a functionality to have visual display of differences in compare stage but was removed because product owner prefers seeing difference via GitHub.

PRODUCT BACKLOG

SPRINT 1,2,3,4

At the beginning of sprint 2, we had originally planned to compare using CSV files, but decided to change the comparison to two XML files as oppose to two CSV files. As a result, we saw a dramatic increase in story points needed for story 7 from the original value of 3 story points to 24 story points used.

Note:

-1 Story point = 0.5 developer hours = 30 minutes

Priority: P **Story points: SP** **Developer Hours: H**	**User Story**
P: 1 **SP: 4** **H: 2**	Story #1: As Anne (an admin), I want a command that will output a manual of all commands and how to use them
P: 2 **SP: 10** **H: 5**	Story #2: As Hanno (an admin), I want a command to list all, line by line, all planetary systems that have been updated in other catalogues since the last commit
P: 3 **SP: 11** **H: 5.5**	Story #3: As Hanno (an admin), I want to be able to manually push and commit updates from other catalogues (Exoplanet.eu, exoplanetarchive.ipac.caltech.edu) to OEC so that I can immediately add updated information to my catalogue
P: 4 **SP: 12** **H: 5**	Story #4: As Hanno (an admin), I want to be able to open a file, stored on my computer to see the information of a system that has been updated from another catalogue
P: 5 **SP: 4** **H: 2**	Story 5: As Hanno, I want to pull a XML file of the updated system corresponding to the table columns of my repository
P: 6 **SP: 4** **H: 2**	Story #6: As Hanno, I want to push an XML file
P: 7 **SP: 32** **H: 12**	Story #7: As Hanno, I want to be able to compare two XML files representing the initial catalogue information versus the updated one.
P: 8 **SP: 5** **H: 2.5**	Story 8: As Hanno (an admin), I want to change the units of measurements that updates from other catalogues should be converted into before committing to my repository.

From our meeting with our TA, we established that the original user story 1 was a non-feature requirement. That allowed us to remove it from our Product backlog for the current sprint and move all other stories up by one priority.

Our original Story 8 has been removed. While working on our story, we noticed that we would be unable to modify existing files on disk to track changes due to security issues. After consulting with Hano, we established that he wants to see the changes via pull request on git.

Priority: P **Story points: SP** **Developer Hours: H**	**User Story**
P: 1 (now removed) **SP: 0** **H: 0**	Story #1: As Anne (an admin), I want a command that will output a manual of all commands and how to use them
P: 8 (Now removed) **SP: 10** **H: 5**	Story #8(Removed): As Hanno, I want to be able to view the description of a system that has been updated from another catalogue in a table format in an html page

SPRINT 5, 6

In Sprint 5, a user story was added to enhance the file comparison functionality. Also, many completed stories in the previous sprints had to be modified to work with the new features. As a result of this, the development took much more story points than expected.

Note:

-Changed Values represented by (OLD -> NEW)

-1 Story point = 0.5 developer hours = 30 minutes

Priority: P **Story points: SP** **Developer Hours: H**	**User Story**
P: 9 **SP: 10 -> 44** **H: 5**	Story #6: As Hanno, I want to push an XML file to a given repo path.
P: 10 **SP: 4 -> 49** **H: 2**	Story #7: As Hanno, I want to be able to compare two XML files representing the initial catalogue information versus the updated one.
P: 12 **SP: 7** **H: 12**	Story #2: As Hanno (an admin), I want a command to list of all planetary systems that have been updated in other catalogues since the last commit.
P: 13 **SP: 67** **H: 2.5**	Story #3: As Hanno (an admin), I want to be able to manually push and commit updates from other catalogues (Exoplanet.eu, exoplanetarchive.ipac.caltech.edu) to my git repository so that I can immediately add updated information to my catalogue

After extracting Hanno's database into XML files, we noticed that there is an additional block called "Binaries" that we have to deal with. A new user story is created to add new features to the XML comparison component. User story #7.5 is added to handle of the new block during the comparison.

Priority: P **Story points: SP** **Developer Hours: H**	**User Story**
P: 11 **SP: 60** **H: 2**	User Story #7.5 (Newly Added): As Hanno, I want to compare the binary content of two files.

SPRINT PLAN + BACKLOG

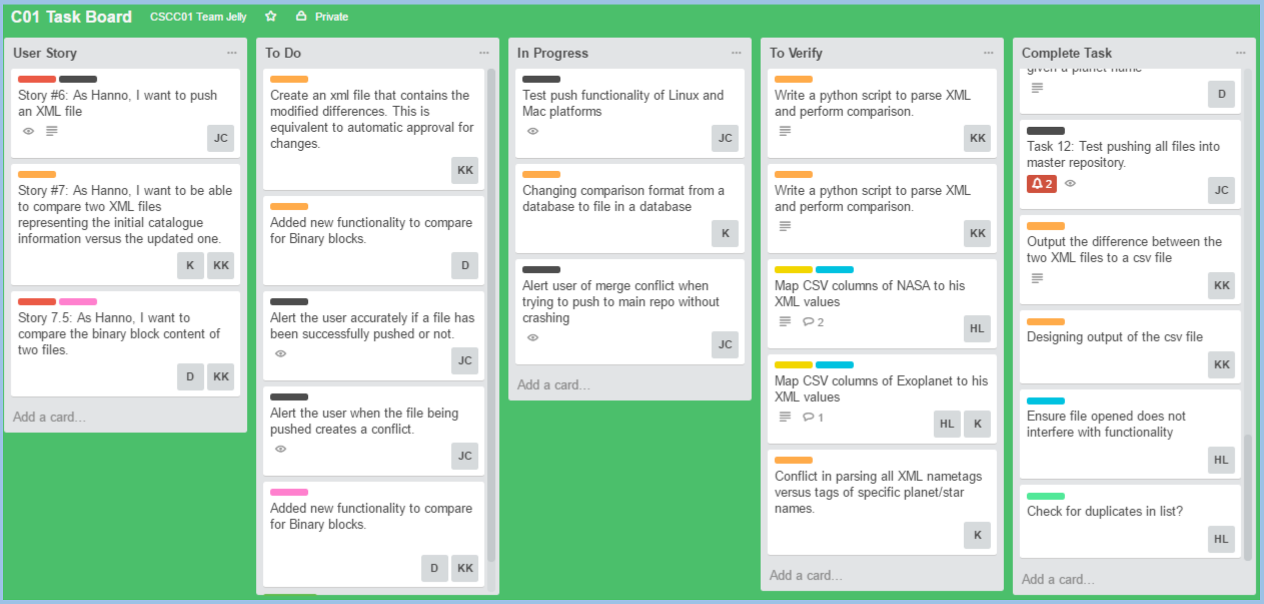
SPRINT 05

In Sprint 5, there were multiple modifications in the points allotted to certain user stories due to requiring more time than expected. We saw that some user stories needed to be divided into simpler tasks. With planning readjustments, the team had to re-assign the new stories to members and story points. Due to the challenges faced, story points were increased by extended research and development.

Another user story was added. The team decided to enhance the file comparison functionality by comparing files on a binary level. During the extraction process of files, we discovered that some files contained blocks of binary that could not be compared with other tags.

Angelina – A [Sprint] – JC [Task Board]
Dennis – D [Sprint] + [Task Board]
Henry – H [Sprint] – HL [Task Board]
Kelly – KM [Sprint] – K [Task Board]
Kris – KL [Sprint] – KK [Task Board]

BEFORE SPRINT 05:



C01 Task Board CSCC01 Team Jelly Private

User Story
- Story #6: As Hanno, I want to push an XML file — JC
- Story #7: As Hanno, I want to be able to compare two XML files representing the initial catalogue information versus the updated one. — K KK
- Story 7.5: As Hanno, I want to compare the binary block content of two files. — D KK

Add a card...

To Do
- Create an xml file that contains the modified differences. This is equivalent to automatic approval for changes. — KK
- Added new functionality to compare for Binary blocks. — D
- Alert the user accurately if a file has been successfully pushed or not. — JC
- Alert the user when the file being pushed creates a conflict. — JC
- Added new functionality to compare for Binary blocks. — D KK

In Progress
- Test push functionality of Linux and Mac platforms — JC
- Changing comparison format from a database to file in a database — K
- Alert user of merge conflict when trying to push to main repo without crashing — JC

Add a card...

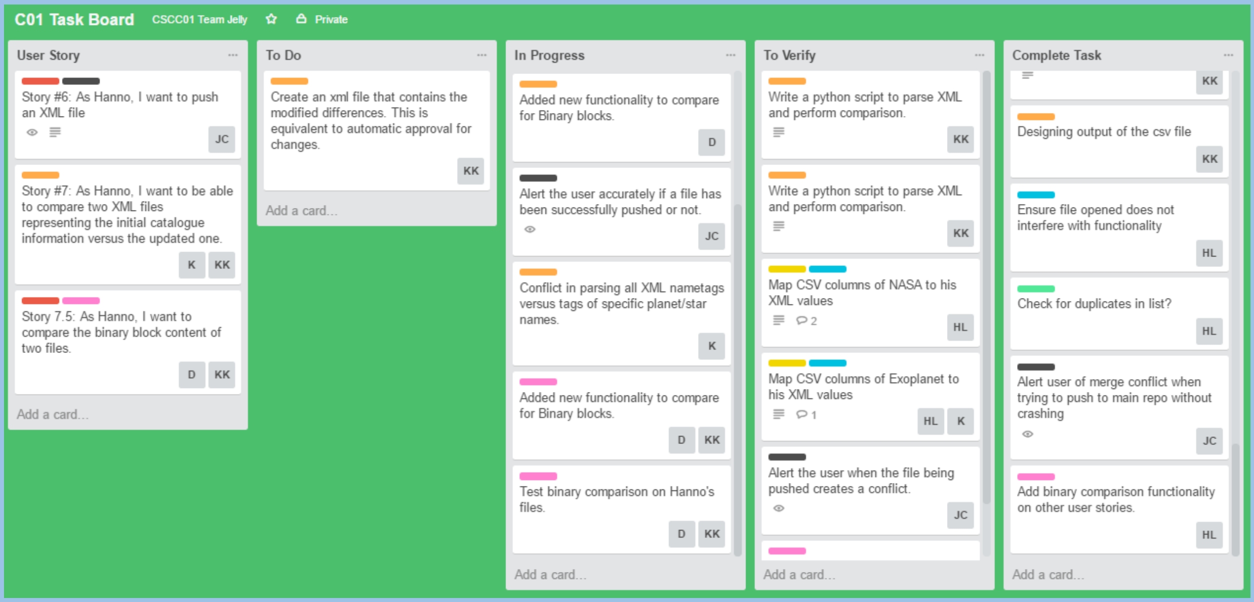
To Verify
- Write a python script to parse XML and perform comparison. — KK
- Write a python script to parse XML and perform comparison. — KK
- Map CSV columns of NASA to his XML values — 2 — HL
- Map CSV columns of Exoplanet to his XML values — 1 — HL K
- Conflict in parsing all XML nametags versus tags of specific planet/star names. — K

Add a card...

Complete Task
- given a planet name — D
- Task 12: Test pushing all files into master repository. — 2 — JC
- Output the difference between the two XML files to a csv file — KK
- Designing output of the csv file — KK
- Ensure file opened does not interfere with functionality — HL
- Check for duplicates in list? — HL

Add a card...

AFTER SPRINT 05:



C01 Task Board CSCC01 Team Jelly Private

User Story

- Story #6: As Hanno, I want to push an XML file — JC
- Story #7: As Hanno, I want to be able to compare two XML files representing the initial catalogue information versus the updated one. — K KK
- Story 7.5: As Hanno, I want to compare the binary block content of two files. — D KK
- Add a card...

To Do

- Create an xml file that contains the modified differences. This is equivalent to automatic approval for changes. — KK
- Add a card...

In Progress

- Added new functionality to compare for Binary blocks. — D
- Alert the user accurately if a file has been successfully pushed or not. — JC
- Conflict in parsing all XML nametags versus tags of specific planet/star names. — K
- Added new functionality to compare for Binary blocks. — D KK
- Test binary comparison on Hanno's files. — D KK
- Add a card...

To Verify

- Write a python script to parse XML and perform comparison. — KK
- Write a python script to parse XML and perform comparison. — KK
- Map CSV columns of NASA to his XML values — 2 — HL
- Map CSV columns of Exoplanet to his XML values — 1 — HL K
- Alert the user when the file being pushed creates a conflict. — JC
- Add a card...

Complete Task

- KK
- Designing output of the csv file — KK
- Ensure file opened does not interfere with functionality — HL
- Check for duplicates in list? — HL
- Alert user of merge conflict when trying to push to main repo without crashing — JC
- Add binary comparison functionality on other user stories. — HL
- Add a card...

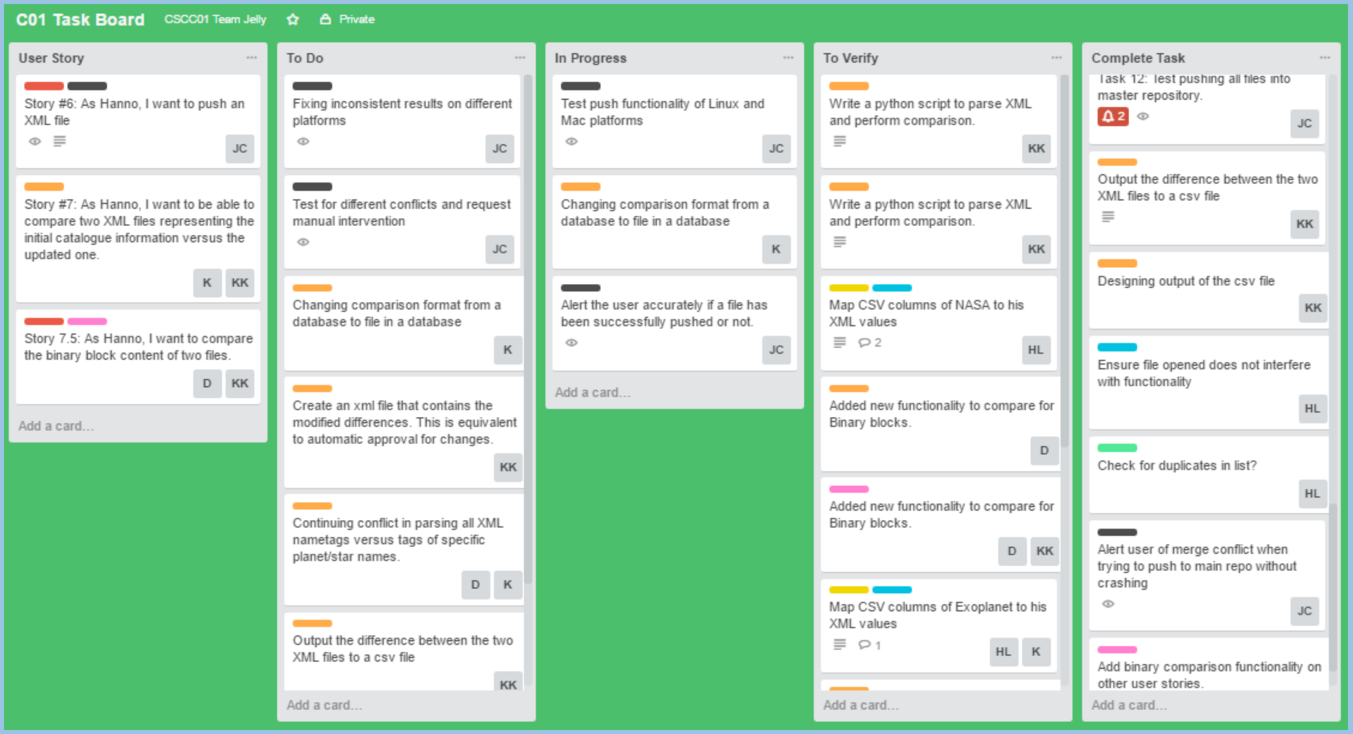
User Story	Tasks	Day 1	Day 2	Day 3	Day 4	Day 5
Story #6: As Hanno, I want to push an XML file to the main repository. **Need to Fix: Crashes when a conflict is detected, along with inaccurate confirmations of file being pushed.**	Test functionality on Linux and Mac platforms					A(1)
	Alert the user accurately if a file has been successfully pushed.				A(4)	A(3)
	Alert the user when the file being pushed creates a conflict.	A(1)	A(6)	A(2)		
Story #7: As Hanno, I want to be able to compare two XML files representing the initial catalogue info vs the updated one. ***This has been pushed to this sprint due to unanticipated difficulties.**	Changing comparison format from a database to file in a database	KM(1)	KM(1)			KM(1)
	Conflict in parsing all XML nametags versus tags of specific planet/star names.			KM(2)	KM(2)	
	Output the difference between the two XML files to a csv file	KL(1)	KL(1)			
New User Story #7.5: As Hanno, I want to compare the binary content of two files.	Added new functionality to compare for Binary blocks.		D(1)	KL(1) + D(1)	KL(1) + D(1)	KL(1)
Story #2: As Hanno, I want a command to list of all planetary systems that have been updated in other catalogues since the last commit.	Accommodate new user story's functionality into existing user story	H(1) + D(1)	H(1)	H(1)	H(1)	H(1) + D(1)

SPRINT 06

Sprint 6 was the merging of individual components into a single cohesive project. During the testing phase there were a great deal of bugs discovered. The team had to backtrack to ensure that the mandatory requirements of the program were functional. This was also the last sprint; therefore, it was crucial that the deliverable would be able to do the necessary functionalities. Most of the story points were spent on that rather than completing other user stories that were add-ons that would improve the deliverable but not absolutely essential.

Many of the user stories completed in previous sprints had to be modified to work with the new user story. Adding the additional feature to our existing code required more story points.

BEFORE SPRINT 06:


C01 Task Board · CSCC01 Team Jelly · Private

User Story

- Story #6: As Hanno, I want to push an XML file (JC)
- Story #7: As Hanno, I want to be able to compare two XML files representing the initial catalogue information versus the updated one. (K, KK)
- Story 7.5: As Hanno, I want to compare the binary block content of two files. (D, KK)

Add a card...

To Do

- Fixing inconsistent results on different platforms (JC)
- Test for different conflicts and request manual intervention (JC)
- Changing comparison format from a database to file in a database (K)
- Create an xml file that contains the modified differences. This is equivalent to automatic approval for changes. (KK)
- Continuing conflict in parsing all XML nametags versus tags of specific planet/star names. (D, K)
- Output the difference between the two XML files to a csv file (KK)

Add a card...

In Progress

- Test push functionality of Linux and Mac platforms (JC)
- Changing comparison format from a database to file in a database (K)
- Alert the user accurately if a file has been successfully pushed or not. (JC)

Add a card...

To Verify

- Write a python script to parse XML and perform comparison. (KK)
- Write a python script to parse XML and perform comparison. (KK)
- Map CSV columns of NASA to his XML values · 2 (HL)
- Added new functionality to compare for Binary blocks. (D)
- Added new functionality to compare for Binary blocks. (D, KK)
- Map CSV columns of Exoplanet to his XML values · 1 (HL, K)

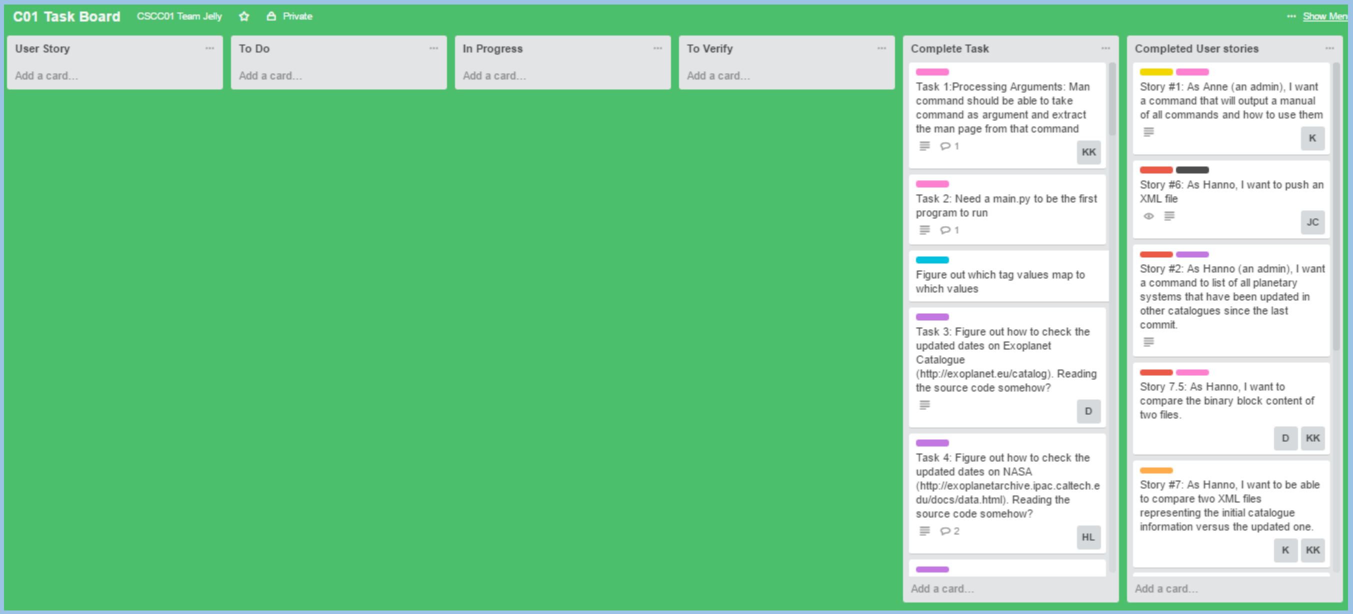
Add a card...

Complete Task

- Task 12: Test pushing all files into master repository. · 2 (JC)
- Output the difference between the two XML files to a csv file (KK)
- Designing output of the csv file (KK)
- Ensure file opened does not interfere with functionality (HL)
- Check for duplicates in list? (HL)
- Alert user of merge conflict when trying to push to main repo without crashing (JC)
- Add binary comparison functionality on other user stories.

Add a card...

AFTER SPRINT 06:



User Story

Add a card...

To Do

Add a card...

In Progress

Add a card...

To Verify

Add a card...

Complete Task

- Task 1:Processing Arguments: Man command should be able to take command as argument and extract the man page from that command — 1 — KK
- Task 2: Need a main.py to be the first program to run — 1
- Figure out which tag values map to which values
- Task 3: Figure out how to check the updated dates on Exoplanet Catalogue (http://exoplanet.eu/catalog). Reading the source code somehow? — D
- Task 4: Figure out how to check the updated dates on NASA (http://exoplanetarchive.ipac.caltech.edu/docs/data.html). Reading the source code somehow? — 2 — HL

Add a card...

Completed User stories

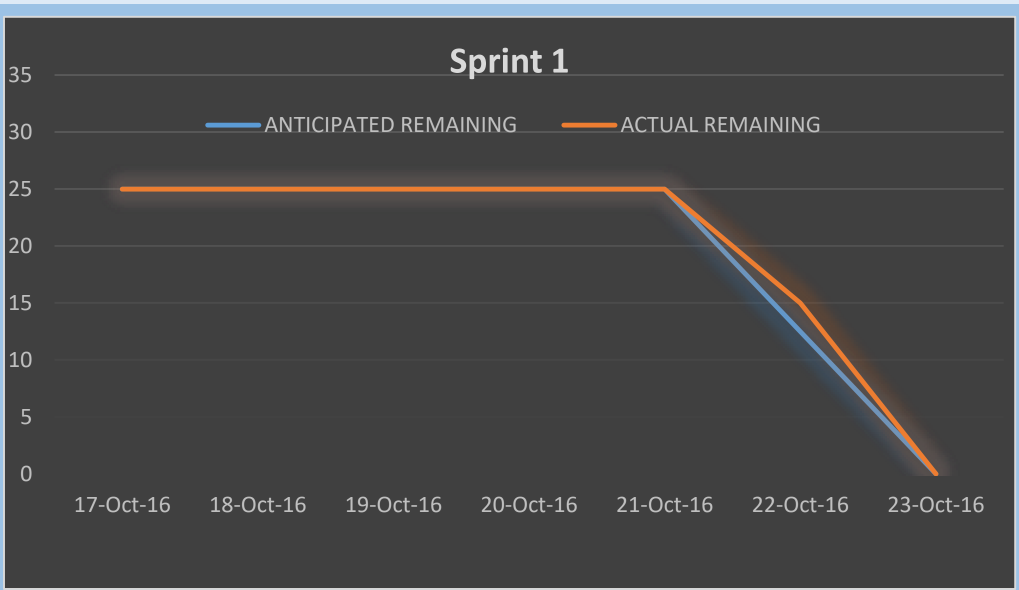
- Story #1: As Anne (an admin), I want a command that will output a manual of all commands and how to use them — K
- Story #6: As Hanno, I want to push an XML file — JC
- Story #2: As Hanno (an admin), I want a command to list of all planetary systems that have been updated in other catalogues since the last commit.
- Story 7.5: As Hanno, I want to compare the binary block content of two files. — D KK
- Story #7: As Hanno, I want to be able to compare two XML files representing the initial catalogue information versus the updated one. — K KK

Add a card...

User Story	Tasks	Day 1	Day 2	Day 3	Day 4	Day 5
Story #6: As Hanno, I want to push an XML file to the main repository	Fixing inconsistent results on different platforms	A(4)		A(4)		
	Test for different conflicts and request manual intervention		A(8)		A(7)	A(4)
Story #7: As Hanno, I want to be able to compare two XML files representing the initial catalogue info vs the updated one. *Tasks from the previous sprint required attention during the merging process to ensure consistency.	Changing comparison format from a database to file in a database			KM(9)	KM(6)	KM(16)
	Continuing conflict in parsing all XML nametags versus tags of specific planet/star names.	KM(7)	KM(4)			
	Output the difference between the two XML files to a csv file		KL(7)			
Story #7.5: As Hanno, I want to compare the binary content of two files.	Modify binary block comparisons	KL(12)				KL(13)
	Create an xml file that contains the modified differences. This is equivalent to automatic approval for changes.			KL(6)	KL(8)	D(15)
Story #3: As Hanno, I want to manually push and commit updates so that I can immediately add updated information to my catalogue	Add the binary comparison feature in pushing and committing updates.	H(4) + D(4)	H(6)			
	Test feature for bugs and fix accordingly		D(15)	H(7) + D(7)	H(6) + D(8)	H(10)

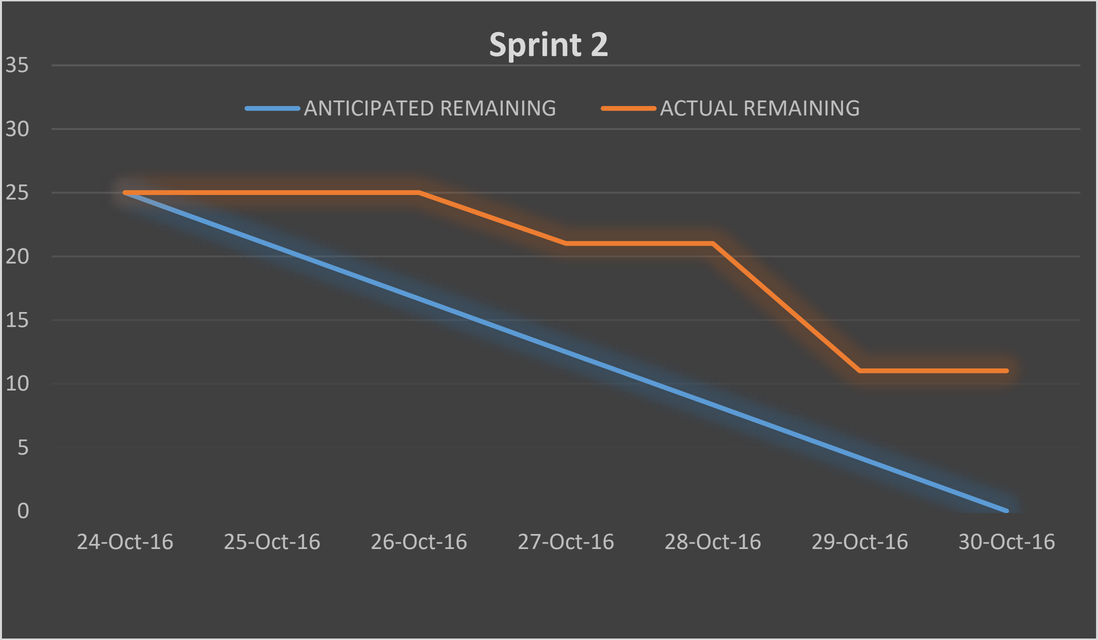
BURNDOWN CHARTS

SPRINT 1

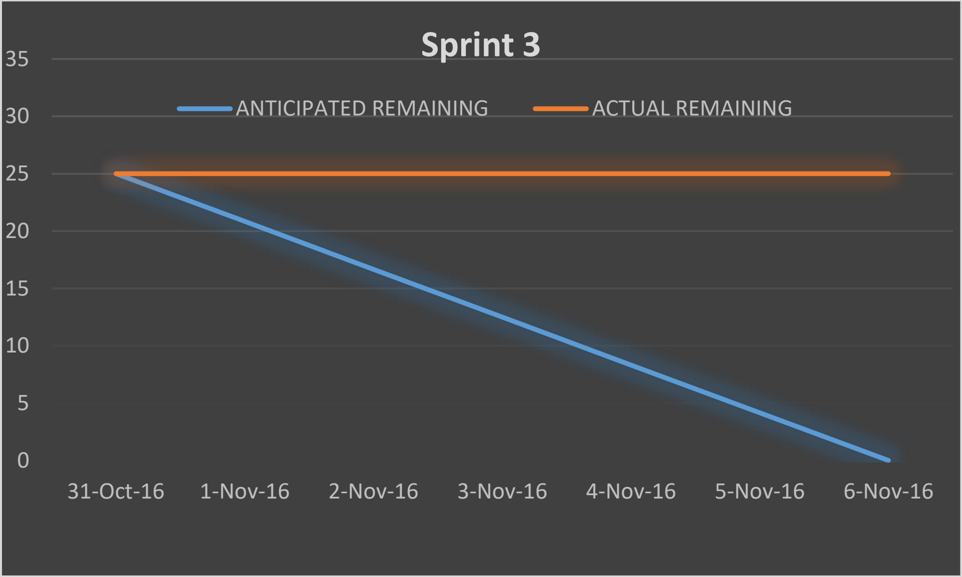


After our first sprint, we have collectively completed 5 story points per developer but over a 2-day period as oppose to over a 5-day period. As a result, our burn down chart has a straight line the first 5-days and a much steeper slopes the last 2-days

SPRINT 2



SPRINT 3



User story completion NOT ACCOMPLISHED IN THIS SPRINT, to be accomplished in next sprint

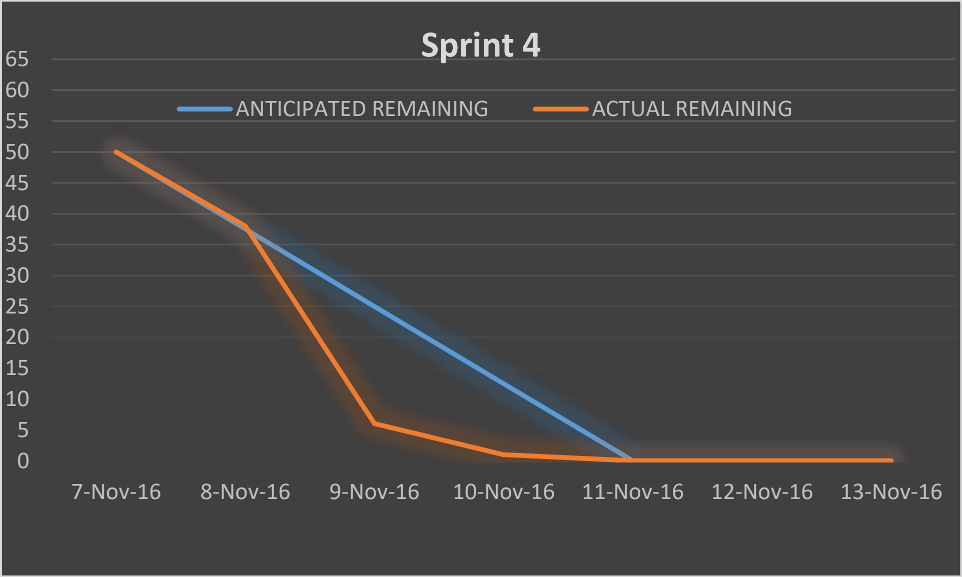
All tasks delayed

Heavy course load for some members, burndown negligible this sprint

due to clarification with client:

Story 7 taking an extremely unexpectedly long amount of time

SPRINT 4



Story point reassignment becoming major issue

-due to clarification with client:

Story 7 took an extremely unexpectedly long amount of time

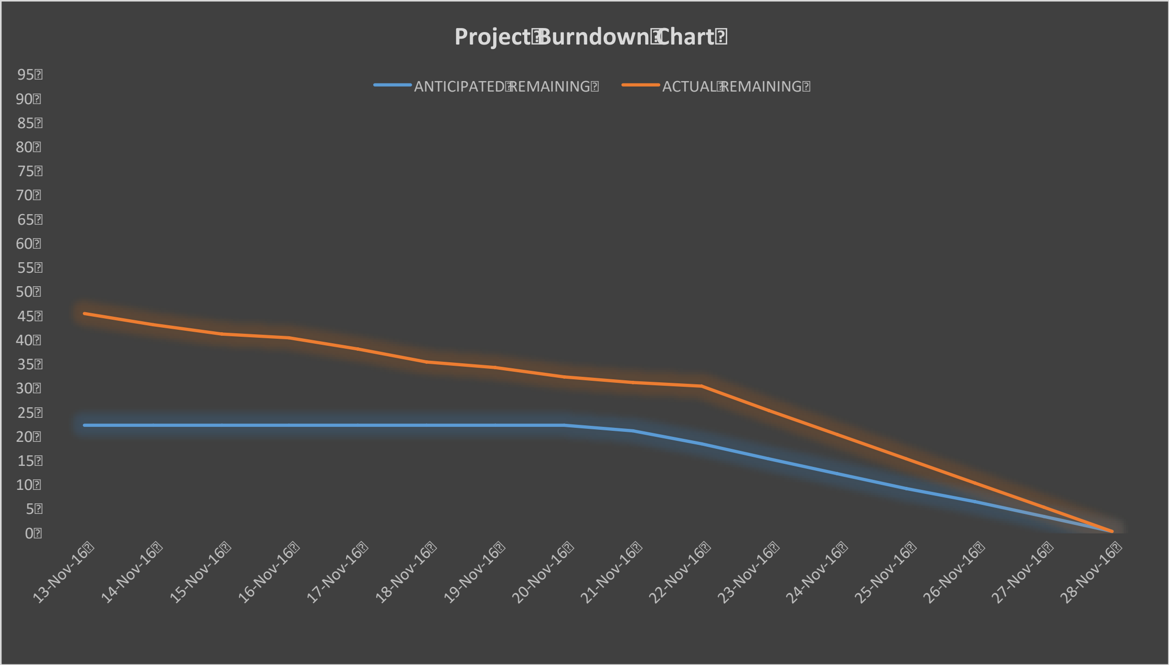
Story 8 removed

-Story 4,7,8 (formerly 9) complete

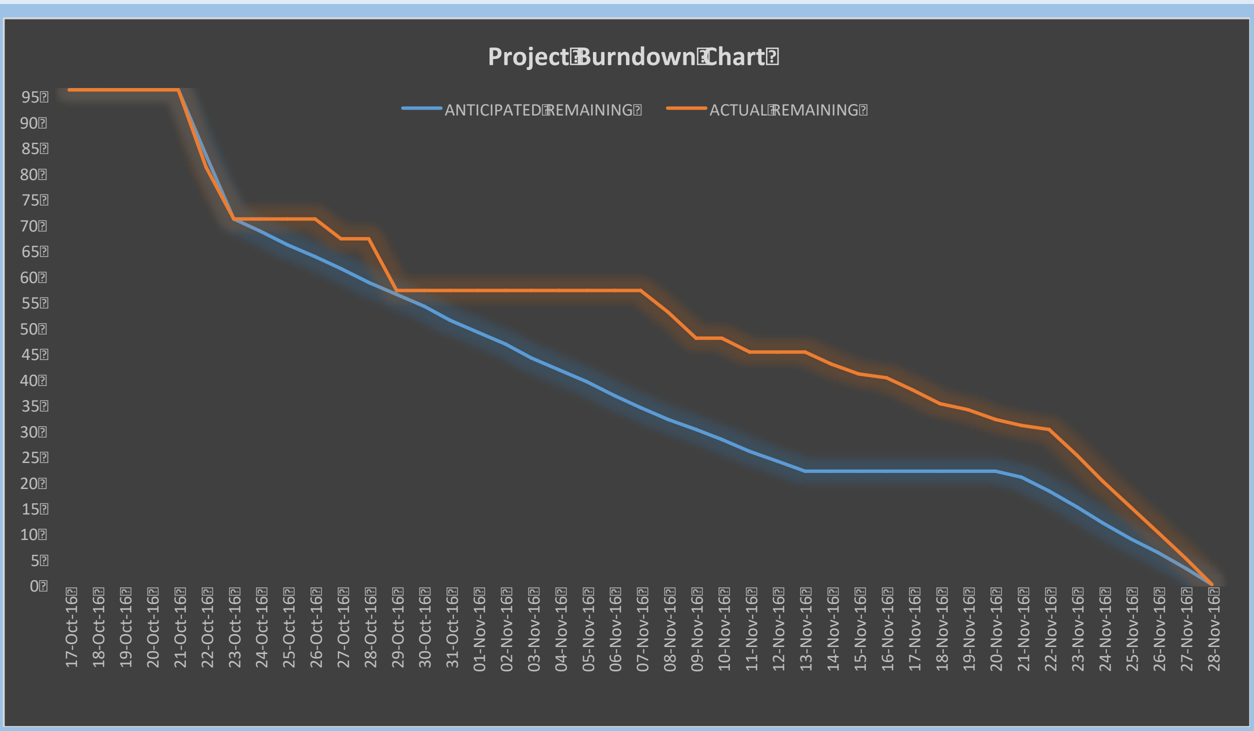
-Code freeze on the 12th for code review session

Summary: The estimated project velocity for deliverable 4 was 75 in total, with 25 per sprint. The actual being 14 with sprint 2, 0 with sprint 3, and 49 with sprint 4. On average the actual project velocity was 21.

SPRINT 5



PROJECT BURNDOWN



Summary:

Looking back on this project, one thing we will need to improve on is developing more accurate predictions on story cost. We saw many of our initial predicted cost increase substantially due to actual higher research time than expected. The most prominent increases of story cost came from extracting from external databases such as exoplanet.eu and Nasa archive. Another major story increase came from comparing two difference XML files (story 7) due to a higher than expected research time required for this user story. Although we faced setbacks stalling development, such as periods where there was no progress made due to exams and other commitments, we were able to rally together near the last two weeks of the project and complete all user stories.

VALIDATION – DEMO WITH CLIENT

Throughout the sprint, we performed validation of our application by having demos with the Product Owner and asking him how he prefers the process flow. We also had to resolve blockages from non-technical issues, such as questions relating to astronomy. Through the validation process, we were able to determine how he would want us to handle binary systems, how he wanted to compare and commit changes, and how he wanted certain data to look. For example: when asking about how he wanted to see changes, the product owner was satisfied with viewing these changes via pull request. He pointed us towards a script to clean up xml files. This can be used to modify XML files to verify that the data is to his liking.

VERIFICATION

CODE INSPECTION

Code Inspection 1:

-Reviewer: Dennis Tra
-Code under review: repo.py
-Date of review: November 30th, 2016
-Code Author: Henry Liu

Categories:

-Bugs:
-storeSystemMatch crashes when given a file that doesn't exist. Simple
-Poor code logic:
-No problems
-Poor coding style:
-Code not in compliance with PEP8 standards. Run code through PEP8 style check and fix accordingly
-Missing documentation:
-Good documentation
-Unreadable code:
-No problems
-Vulnerabilities in code
-None found
-Passed all unit test in testrepo.py

Code Inspection 2:

-Name of Reviewer: Kris Lai
-Code under review: compare.py
-Date of review: December 1st, 2016
-Author of the code: Dennis Tra

Categories:

-Missing Documentation:
-Incomplete Docstrings missing for each function.
-Issues:
1) Don't know what types of parameter
2) Don't know what the function returns
This led to difficulties in writing unit test.
-Unclear comments:
1) Comment typos, hard to understand (Binary_Check Function it's -> its)
2) "TO DOs" comment should be taken away after implementation

Code Inspection 3:

-Code Reviewer: **Angelina Choi**
-Code Under Review: **main.py**
-Date of Review: November 30th , 2016
-Code Author: **Henry Liu**

Categories:

Bugs

No bugs have been detected. Code does not crash with unreasonable input and all output is as expected.

Poor Code Logic

Redundant code with functions get_names2 and get_names. Insufficient commenting as to explain why both functions are needed.

Poor Coding Style

Minor issues with lack of whitespace on certain lines. Inconsistencies with commenting as function compare is the only function to have a docstring and no commenting.

Missing Documentation

No documentation for function get_names2.

Unreadable Code

N/A

Vulnerabilities in Code

Data inputs are not checked when comparing databases but is unnecessary as the input is automated and is always the correct type.

Poor Testing

Testing was done with group members with multiple cases.

Code Inspection 4

-Code Reviewer: Kelly Mo
-Code Under Review: main.py [date and extract functionality]
-Date of Review: November 30th, 2016
-Code author: Henry Liu

Categories

Missing Documentation

- missing command in help function for date function
-missing optional commands (-l, etc) in help function

Poor coding style

- Needs to follow PEP-8

Some manual test:

Date:

```
input: ["date -c 2016-12-12", "date"]
output:  [Last commit date has been changed to: 2016-12-12", "2016-12-12"]
input: ["date -c 1/1/1", "date"]
output: ["Could not parse 1/1/1 as date. Please try again with date format YYYY-MM-DD", "2016-12-12"]
input: ["date -c 1-1-1", "date"]
output: ["Could not parse 1-1-1 as date. Please try again with date format YYYY-MM-DD", "2016-12-12"]
ALL TESTS PASSED
```

Extract:

Input: ["date -c 0000-00-00", "date"]
Output: ["Could not parse 0000-00-00 as date. Please try again with date format YYYY-MM-DD", "2016-12-12"]
Input: ["date -c 1999-00-00", "date"]
Output: ["Could not parse 1999-00-00 as date. Please try again with date format YYYY-MM-DD", "2016-12-12"]
Input: ["date -c 1999-19-00", "date"]
Output: ["Could not parse 1999-19-00 as date. Please try again with date format YYYY-MM-DD", "2016-12-12"]
Input: ["date -c 1999-11-01", "date"]
Output: ["Last commit date has been changed to: 1999-11-01", "1999-11-01"]
ALL TESTS PASSED

Code Inspection 4

-Code Reviewer: Henry Liu
-Code Under Review: main.py [date and extract functionality]
-Date of Review: November 30th, 2016
-Code author: Kelly Mo

Categories

Bugs

–Did not match some pairs of XML files even though they were the same systems

-Program crashes – some XML files crash the program due to encoding

Poor code logic

– some confusion on get_names() on how it works? Not easy to follow.

-redundant or duplicate code – Don't need to remember "none" for line 7

Vulnerabilities in code

-data inputs not checked -None

-where third-party code is used, not all errors are caught –XML files needed to be opened in UTF-8.

-invalid parameters not handled –Test blocked from encoding error