CSU2013 / CSU3012 - Group 4

DANCE - Detection of ANti-patterns in a C++ Environment - **Final Sprint Documentation**

Team Members - *Dylan Fitzpatrick - 20331794*
Elliot Lyons - 20333366
Daniel Penrose - 20331752
Sprina Chen - 21339184
Antoni Zapedowski - 21366133
Bryan Chikwendu - 21363862
Gráinne Ready - 20332706
John Wallace - 21364595
Liam Byrne - 21364304
Leon Byrne - 21365536

Updated Product Backlog

We slightly changed our product backlog for this sprint. We decided we wanted to have our program run as a separate executable program. This enhanced presentability of our program over the simple command line operated program we had at the start of this sprint.

You may also notice we came up with a catchy name for our software - 'DANCE'. We were also advised to add a licence to our repository so other developers could potentially build upon our software in the future.

Planning

Sprint Planning (20/03/23)

The third year students had a meeting in which we discussed our overall progress up until this point. We discussed what would be needed by the end of this sprint so we would have a final product. We had been happy with our progress up until this point, knew where we needed to point most of our attention and were ready to put in the work.

Overall Sprint Objectives

We outlined the objectives for this sprint to be:

- Build upon our third release to have a final product
- Get each antipattern completed and integrated with the new project architecture.
- Ensure continuous integration passes successfully and sufficient tests are written.
- Prepare for our final presentation.
- Prepare for project handoff.

Week by Week Breakdown

We broke the sprint down into a weekly structure. Our usual meeting schedule was slightly disrupted by the calendar again with Bank Holidays sometimes preventing us from meeting our demonstrator on occasion.

Week 1 (21/03/23 - 27/03/23)

This week was the first full week we had for about two weeks. We spent a lot of this week getting re-organised from where we left off in the weeks prior. We continued to work on our anti-patterns.

Week 2 and 3 (28/03/23 - 10/04/23)

These two weeks were seen as the important part of the sprint. We were too late in the project cycle to assign new tasks so those who had finished theirs helped other members who were struggling with their problems. Within these two weeks we got all, bar a few minor issues in one anti-pattern, working. We looked toward next week with the submissions in mind.

Week 4 (11/04/23 - 14/04/23)

We spent this week finalising our antipatterns in order to have a sufficient final product. This involved fixing some last minute bugs, resolving issues with tests not passing in the CI as well as merge conflicts. A few members also conducted a 'code tidy-up'. Chris expressed that he was very happy with the work we did during our final client meeting and he hopes that Cisco can use this project to help in future projects.

Please see 'Sprint Review' for our 'Sprint Backlog'

Project Organisation

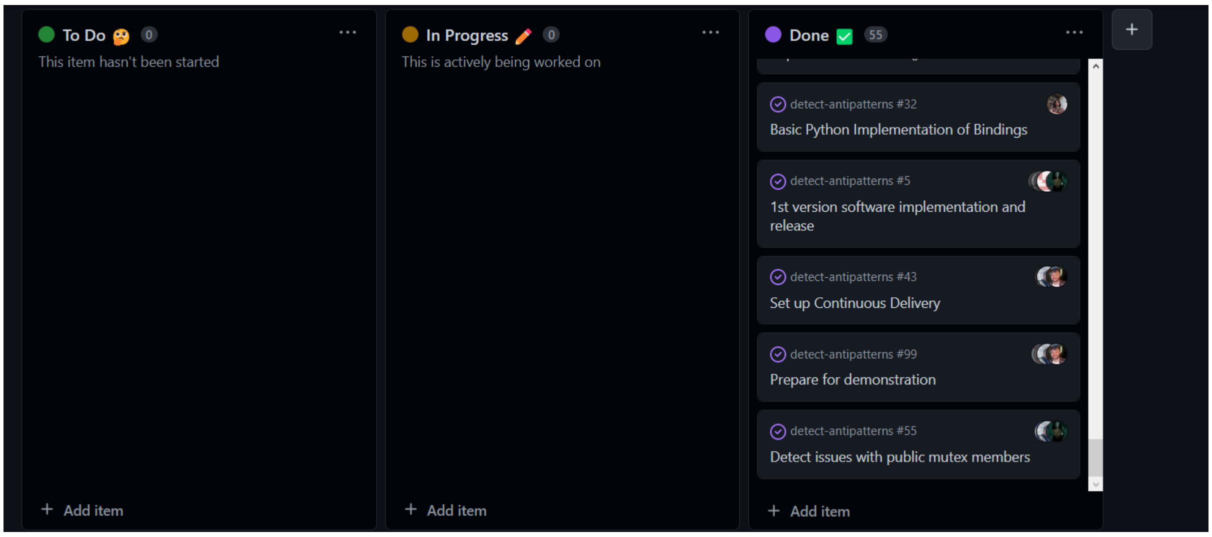
Staff Chart

Product Owner: Daniel Penrose - 20331752
SCRUM Master: Elliot Lyons - 20333366
Team Members:

Dylan Fitzpatrick - 20331794
John Wallace - 21364595
Sprina Chen - 21339184
Antoni Zapedowski - 21366133
Bryan Chikwendu - 21363862
Gráinne Ready - 20332706
Liam Byrne - 21364304
Leon Byrne - 21365536

Project Management with GitHub (14/04/23)

Kanban board



To Do 0
This item hasn't been started
+ Add item

In Progress 0
This is actively being worked on
+ Add item

Done 55

detect-antipatterns #32
Basic Python Implementation of Bindings

detect-antipatterns #5
1st version software implementation and release

detect-antipatterns #43
Set up Continuous Delivery

detect-antipatterns #99
Prepare for demonstration

detect-antipatterns #55
Detect issues with public mutex members

+ Add item

As you can see from the image above, we achieved all of our planned goals throughout the duration of the project with no tasks left in 'To Do'/'In Progress'.

Google Drive

We shared files, managed and collaborated through Google Drive. This can be found here: https://drive.google.com/drive/folders/1tKuXwD7NfoCil3Mx7pL30MKZi8zTGe_-?usp=share_link

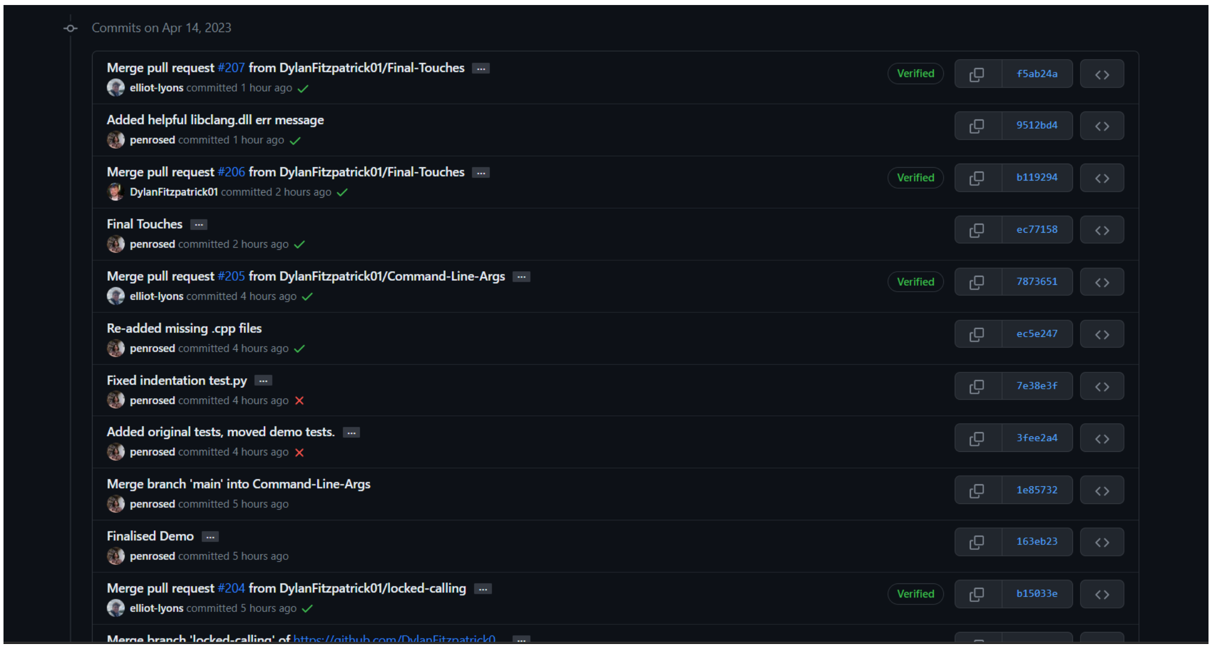
Development

Implementation

GitHub: https://github.com/DylanFitzpatrick01/detect-antipatterns

DockerHub: https://hub.docker.com/r/elliotlyons/detect-antipatterns

Commit Logs (14/04/23)



Commits on Apr 14, 2023

Merge pull request #207 from DylanFitzpatrick01/Final-Touches … Verified f5ab24a
elliot-lyons committed 1 hour ago ✓

Added helpful libclang.dll err message 9512bd4
penrosed committed 1 hour ago ✓

Merge pull request #206 from DylanFitzpatrick01/Final-Touches … Verified b119294
DylanFitzpatrick01 committed 2 hours ago ✓

Final Touches … ec77158
penrosed committed 2 hours ago ✓

Merge pull request #205 from DylanFitzpatrick01/Command-Line-Args … Verified 7873651
elliot-lyons committed 4 hours ago ✓

Re-added missing .cpp files ec5e247
penrosed committed 4 hours ago ✓

Fixed indentation test.py … 7e38e3f
penrosed committed 4 hours ago ✗

Added original tests, moved demo tests. … 3fee2a4
penrosed committed 4 hours ago ✗

Merge branch 'main' into Command-Line-Args 1e85732
penrosed committed 5 hours ago

Finalised Demo … 163eb23
penrosed committed 5 hours ago

Merge pull request #204 from DylanFitzpatrick01/locked-calling … Verified b15033e
elliot-lyons committed 5 hours ago ✓

Merge branch 'locked-calling' of https://github.com/DylanFitzpatrick0 …

Continuous Integration and Delivery (CI/CD) (14/04/23)

Continuous Integration:

Our repository automatically runs tests on our code with each push to main

Continuous Deployment:

Our program automatically pushes to DockerHub upon each release. The DockerHub repository can be found here:

https://hub.docker.com/repository/docker/elliotlyons/detect-antipatterns/general

Sprint Review (14/04/23)

In this sprint we successfully implemented the following:

- Detected Anti-Patterns in a Large C++ Codebase Using Clang/LLVM

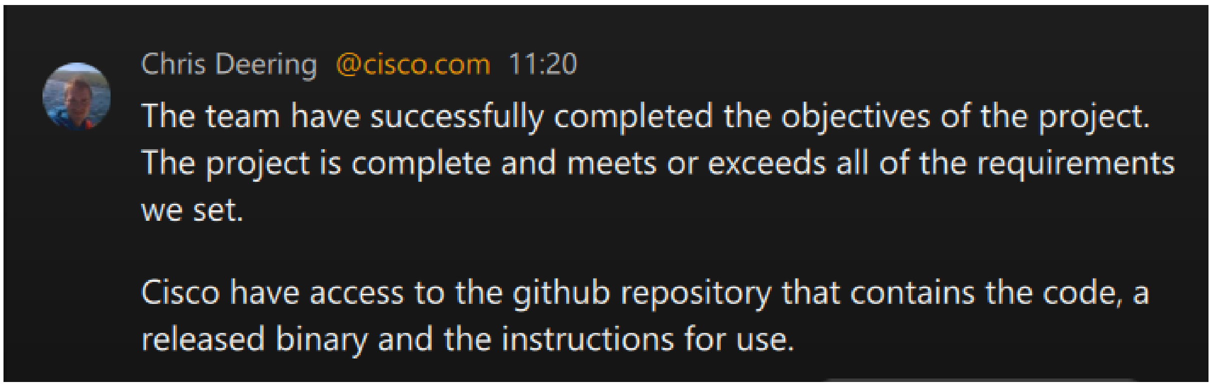
You'll notice this is our original project title. As per the client's handoff, we successfully implemented what we had originally set out to do. We are all proud of our work and we hope to get the opportunity to work on a similar project in the future.

Sprint Retrospective (14/04/23)

Student	Learnings
Daniel Penrose	Lesson learnt: When I try to really apply myself practically to programming and problem solving, I spend less time doing my managerial duties. This kind of time management between guiding and *doing* was something I wasn't expecting; It was very hard to do both at all times. I learned to trust my team more. Initially I spent hours and hours preparing branches with code for my teammates, because I wanted to "lighten their load", when I should've just trusted us to start developing together. Action to take: Get more acquainted with CMake, PyInstaller, and other build tools. It's an aspect of a project's lifecycle that I was really underprepared for.
Elliot Lyons	Lesson learnt: Looking broadly at the project as a whole, I learned the difficulty in solving errors when you're not coding yourself. In a managerial role you need to delegate, not solve at times. I learned the benefit in trusting your team to do so. The number of problems we faced seemed to increase at the end of the project's lifespan (as they always do). I learned the importance of leaning on the team to solve them. Action to take: Be more confident in delegating issues to others and not feel you have to solve everything yourself. This is especially applicable in this managerial role.
Dylan Fitzpatrick	Lesson learnt: The final stretch of a project can get very busy and it can become hard to balance other tasks along with the workload that comes with working on a large-scale project Action to take: Start working on finalising the project earlier, leaving less stress towards the very end of the project, especially coming to the end of the semester.
Sprina Chen	Lesson learnt: Testing is important as errors can occur when executing different kinds of files. This will cause a delay in producing the final product. Action to take: Working on projects earlier and working out times to make sure that there is progress made and not rushing it. Learning to make sure all testing works in all cases.
Antoni Zapedowski	Lesson learnt: The solution we create even if it seems fully working needs to be very carefully tested. One thing that we forgot to do is to test the antipatterns solutions on a broader range of files and we only focused on our own unit tests. This turned out to be a bad approach as once we tested the code on many even unrelated c++ files we noticed new errors and exceptions. Action to take: Broaden the range of test cases, test it on all available files and assume there could always be something wrong with the code, even if we can't currently find an error.

Bryan Chikwendu	Lesson learnt: Code can fail in unexpected ways, and you'll never be able to cover every single case with testing, so it's important to have systems in place to allow your program to still work even through errors Action to take: Develop robust error-handling procedures, so that even when an error is given, the program can potentially handle it or try again without completely crashing
Gráinne Ready	Lesson learnt: Your code is only as good as your unit tests, towards the end we faced issues where some antipatterns were running perfectly in their own tests, but acting irrationally and throwing exceptions in other antipatterns' test files. This could have been noticed sooner had we developed better testing and error-handling strategies Action to take: Produce more advanced, broader unit tests in future, to ensure code works in all situations and environments.
John Wallace	Lesson learnt: during the final stretch of the project even though all of our functions worked for the test files we had set up but there were some simple errors when the functions ran on other files, it showed me how important it is to test your functions not just against files that you didn't create. Action to take:To include more vigorous testing of my testing but also encourage/ask others for help on your code as i may not spot every error even the obvious ones
Liam Byrne	Lesson learnt: Testing is extremely important as even when code seems to be 100% working, you can't be certain until in depth testing is complete. Communication is extremely important in group projects and it's better to ask for help than wasting time trying to finish something you can't by yourself. Action to take: Continue to improve my communication and be less hesitant to ask for help. Test code as rigorously as I can to find edge case errors before they cause a problem. Don't just focus on solving an antipattern in a single scenario but in all scenarios.
Leon Byrne	Lesson learnt: Testing should extend more than just in isolation. How my code affected others was more important than I had previously considered and created far more work in order to fix it. Action to take: Do more rigorous testing before I assume that it will not break it. Attempt to find edge cases and check how my code will handle it.

Client Hand Over Proof and Sign Off of Requirements (13/04/23)



Chris Deering @cisco.com 11:20

The team have successfully completed the objectives of the project. The project is complete and meets or exceeds all of the requirements we set.

Cisco have access to the github repository that contains the code, a released binary and the instructions for use.