To-do list Web application

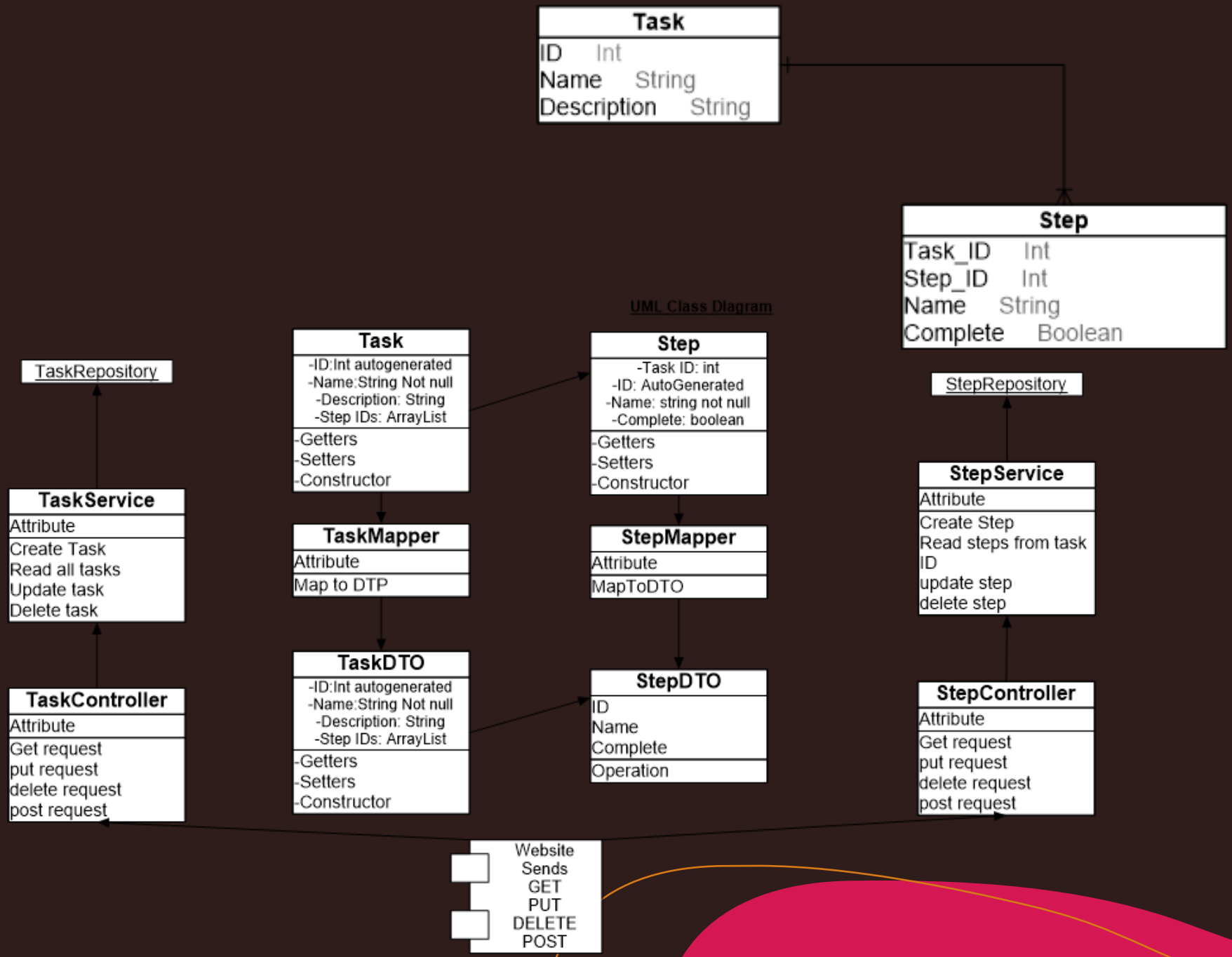
Ross Cowie

The Specification

- The specification asked for:
  - A to-do list web application created using an API for the back-end and a website front end connecting to it.

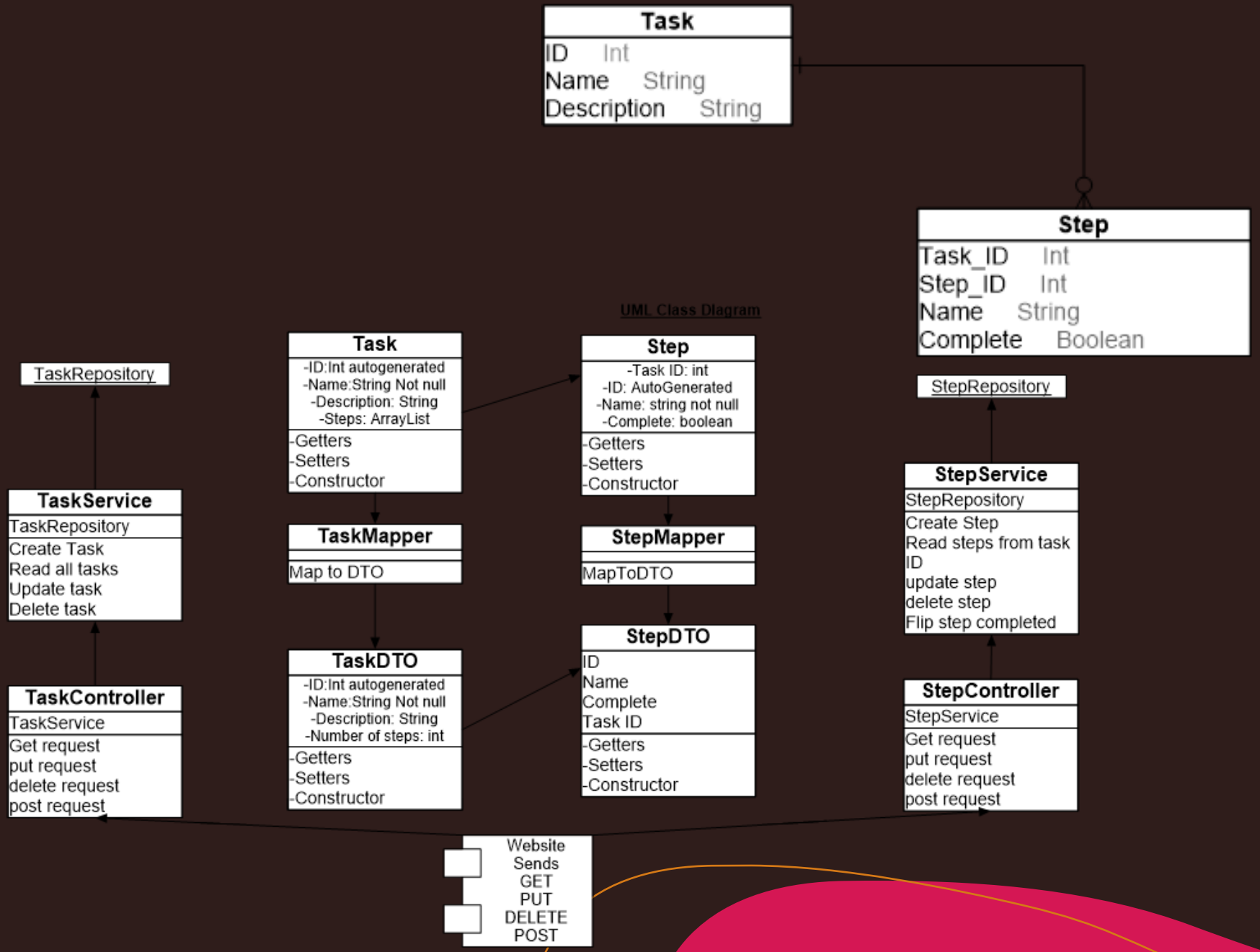
Design

- The initial design of the program was of a task being created with one or more steps. With full CRUD functionality for both.



Design

- After the first sprint the database was changed so a task could also have zero steps to it, and the step service had a new put request added to handle changing the completion status of the step only.



Sprints

- Each sprint was 3-4 days in length, and three sprints were completed over the course of the project.
- Sprint 1

This sprint handled creation and testing of the API. It lasted about 4 days, from the first project day till the end of tuesday week two.

- Sprint 2

This sprint was spent on the front end of the program. There were also several changes made to the API to clear up some issues associated with the front end, such as cross origin issues. This sprint finished at closing on thursday week two.

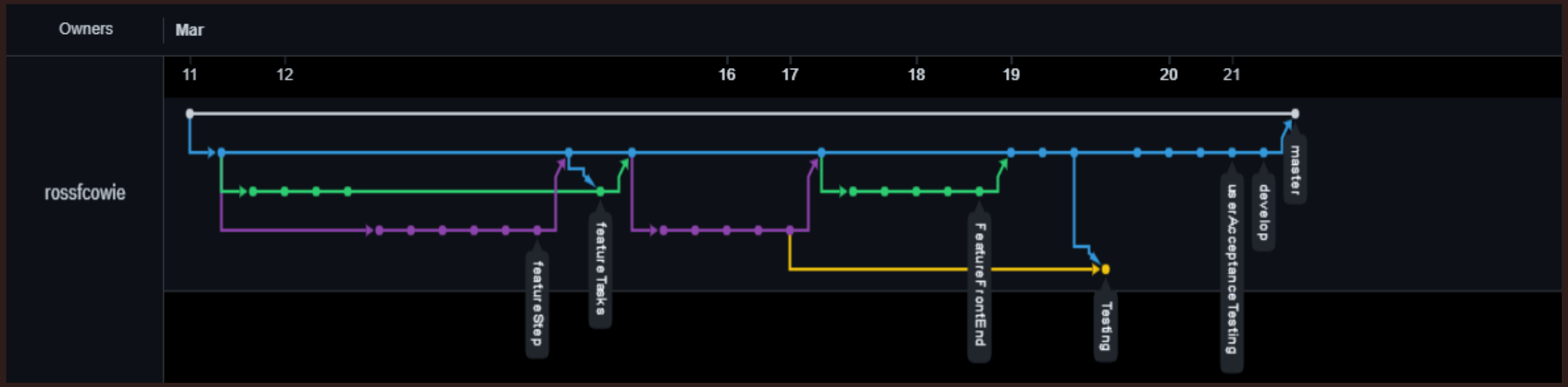
- Sprint 3

This sprint was spent on user-acceptance testing and the user guide. It started on friday of week two and ends at closing the following monday.

Technologies used

- Spring
- Javascript
- Selenium
- Extent reports
- SonarQube

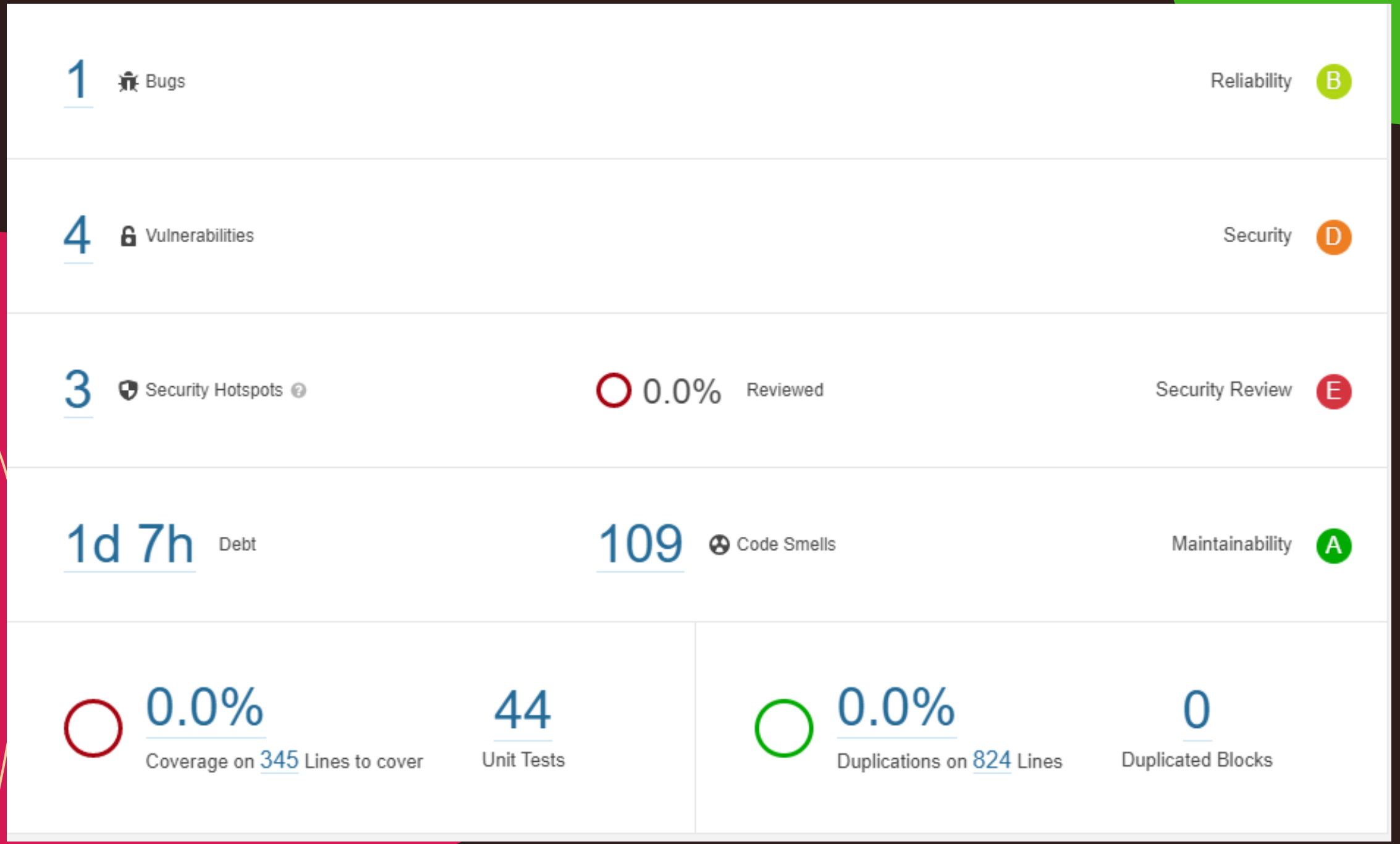
Version control

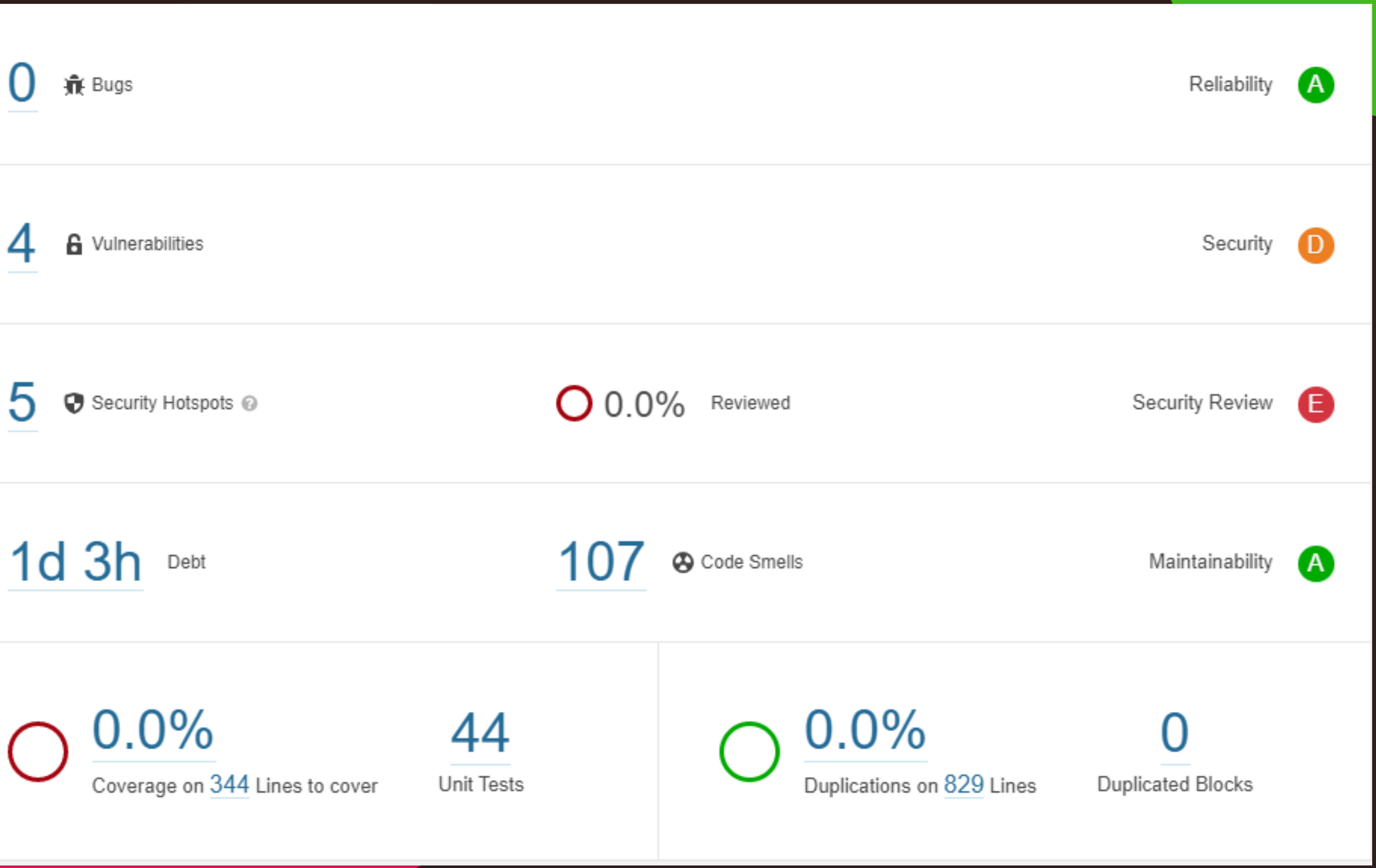


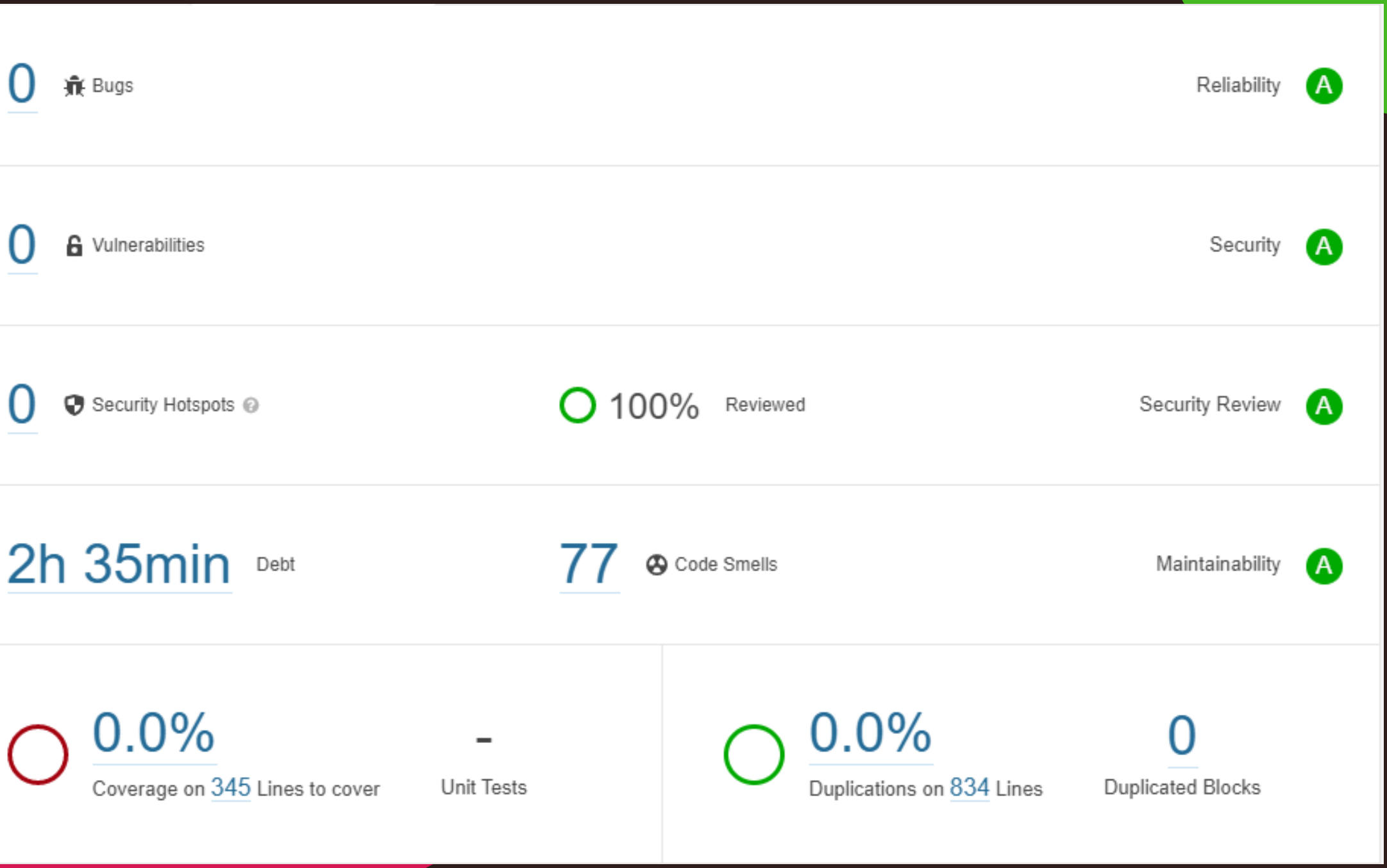
Testing

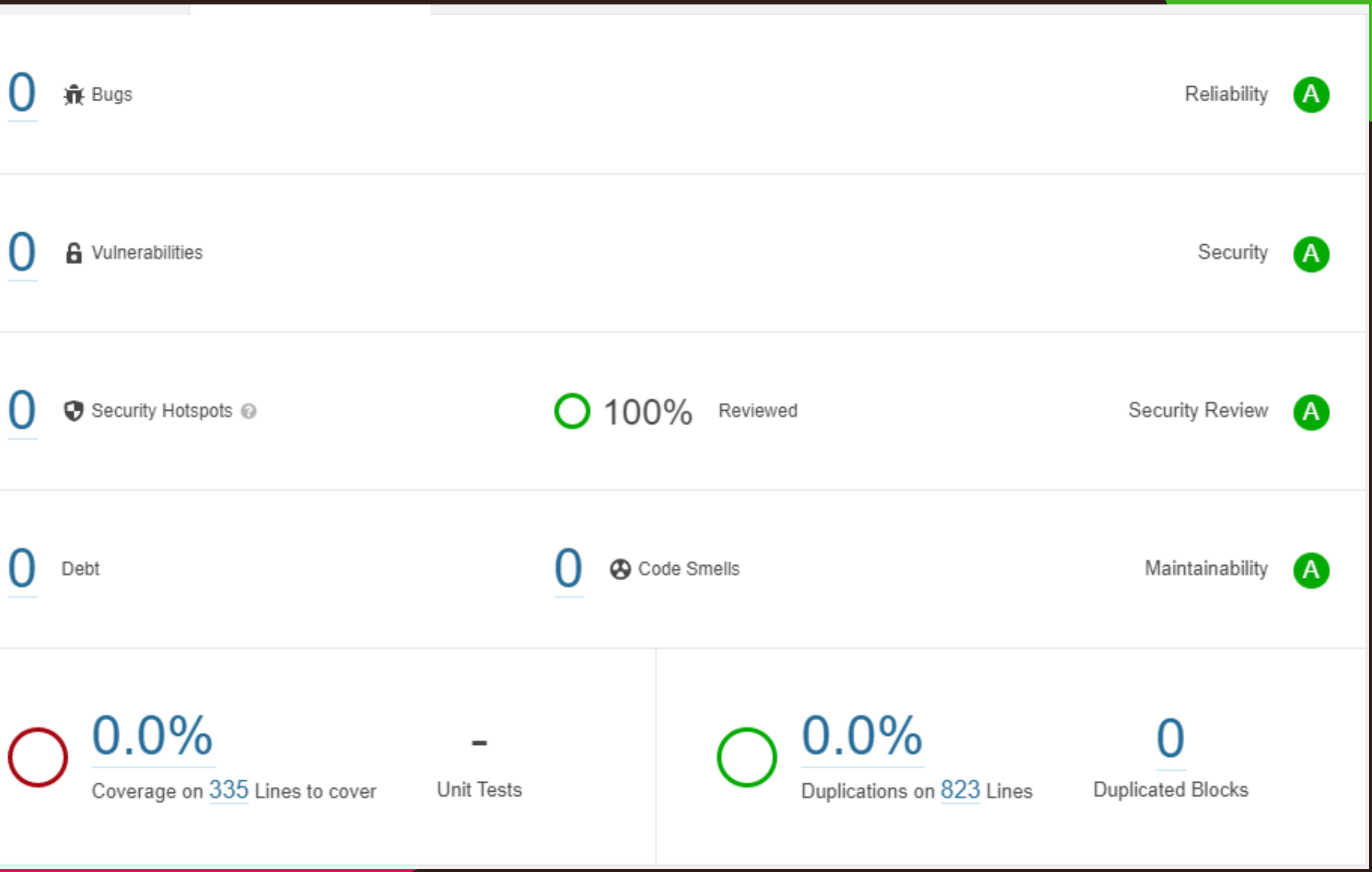
Name	Size	Date modified
Step_Controller_Integration_Report.html	22.9 kB	3/21/21, 10:30:44 PM
Step_Controller_Unit_Report.html	22.9 kB	3/21/21, 10:30:45 PM
Step_Service_Integration_Report.html	22.9 kB	3/21/21, 10:30:41 PM
Step_Service_Unit_Report.html	22.9 kB	3/21/21, 10:30:43 PM
Task_Controller_Integration_Report.html	21.1 kB	3/21/21, 10:30:45 PM
Task_Controller_Unit_Report.html	21.1 kB	3/21/21, 10:30:46 PM
Task_Service_Integration_Report.html	21.1 kB	3/21/21, 10:30:43 PM
Task_Service_Unit_Report.html	21.1 kB	3/21/21, 10:30:42 PM
User_Acceptance_Report.html	31.2 kB	3/21/21, 10:36:12 PM

ToDoListAPI	96.3 %	3,840	147	3,987
src/main/java	89.1 %	1,188	145	1,333
com.qa.todolistapi.mapper	48.7 %	37	39	76
com.qa.todolistapi.model.dto	92.0 %	382	33	415
com.qa.todolistapi.service	89.1 %	213	26	239
com.qa.todolistapi.model.data	94.4 %	403	24	427
com.qa.todolistapi.exceptions	0.0 %	0	14	14
com.qa.todolistapi	37.5 %	3	5	8
com.qa.todolistapi.configuration	42.9 %	3	4	7
com.qa.todolistapi.controller	100.0 %	147	0	147
src/test/java	99.9 %	2,652	2	2,654


1
Bugs
Reliability
B
4
Vulnerabilities
Security
D
3
Security Hotspots
0.0%
Reviewed
Security Review
E
1d 7h
Debt
109
Code Smells
Maintainability
A
0.0%
Coverage on 345 Lines to cover
44
Unit Tests
0.0%
Duplications on 824 Lines
0
Duplicated Blocks


0
Bugs
Reliability
A
4
Vulnerabilities
Security
D
5
Security Hotspots
0.0%
Reviewed
Security Review
E
1d 3h
Debt
107
Code Smells
Maintainability
A
0.0%
Coverage on 344 Lines to cover
44
Unit Tests
0.0%
Duplications on 829 Lines
0
Duplicated Blocks


0 Bugs
Reliability A
0 Vulnerabilities
Security A
0 Security Hotspots
100% Reviewed
Security Review A
2h 35min Debt
77 Code Smells
Maintainability A
0.0%
Coverage on 345 Lines to cover
-
Unit Tests
0.0%
Duplications on 834 Lines
0
Duplicated Blocks


0
Bugs
Reliability
A
0
Vulnerabilities
Security
A
0
Security Hotspots
100%
Reviewed
Security Review
A
0
Debt
0
Code Smells
Maintainability
A
0.0%
Coverage on 335 Lines to cover
-
Unit Tests
0.0%
Duplications on 823 Lines
0
Duplicated Blocks

Demonstration

- As a user I want to be able to Create a task and add it to my to-do list so I can keep track of things I need to do.
- As a user I want to be able to Create a step and add it to a task so I can track progress through a single task.
- As a user I want to be able to View all of the tasks I have to do so I can track what I have to do.
- As a user I want to be able to View all of the steps associated with a single task so I can track what needs to be done to complete it.
- As a user I want to be able to Delete a task so I can remove tasks that no longer need to be completed.

Sprint review

- Sprint 1

API was completed in 4 days which was the total time alloted to this sprint. All goals for it were completed, but they did need some adjustment when the front end came into it. This was mostly as both sides of the connection were unclear in what exactly they needed from each other initially.

- Sprint 2

Website was created in the allotted time, Whilst it does not look the best, appealing visuals was not high on the priorities list as functionality needed to be ensured for the minimum viable product. User-acceptance testing for the site could have been placed into this sprint, but it was delayed to sprint three since it was not on the Jira board when the sprint was started.

- Sprint 3

User testing was finished in time and the documentation had about a day of time to be cleaned up in. At the time of this presentation all that is missing is the user guide.

Sprint Retrospectives

- Sprint 1
- This sprint lasted longer than intended. This was mostly down to the increased amount of time spent on creating the integration tests. The actual functionality was done in two days and the unit tests took about half a day total; however, the integration tests threw up several issues that took a bit longer to sort out.
- Sprint 2
- This sprint also took a bit longer than I would have liked. Taking up the better part of three days. The main issues came from the limited amount of data I was sending back in each fetch request. To keep down the amount of wasted space from the program, this caused logic problems for things like updating the steps in a task, which took a while to solve.
- Sprint 3
- This sprint was quite well handled with the user-acceptance testing being done in a day and a half, with most of the first day being spent on getting the form modal to work in selenium. Once that was handled the tests just took a bit of time to write and get working. The rest of the duration of this sprint was spent on the documentation, and static analysis to clean up the code. I feel this sprint went far better than the prior two but that is mostly as it had the fewest headaches from overthinking problems.

Conclusion

- This project was:
- Challenging
- Engaging

- What I would do if I had more time:
- Rewrite much of the API.
- Clean up the javascrpit.
- Add comments throughout the code.

Questions?