Quality Management

*Team CAtS (Yihong Zou, Huairui Qi, Siqi Lu, Yun-Hsuan Chang)*

[**Quality Attributes (Table)**](#_s2gn7kshsyc7) **2**

[**Testing Roadmap**](#_5y4m4bpa0fct) **2**

[**Test procedures**](#_k1taw7kv5979) **3**

[Technical Review](#_l53e0vomg5dv) 3

[Design Review](#_rvef6hok4aas) 3

[Code Review](#_ce864ipshoe7) 3

[Unit testing](#_nqzt51feywrd) 4

[Integration Test](#_tx7p4qk4awgc) 4

[Static Analysis](#_quedb5ofzpo0) 5

[System Testing](#_pqgdoqlt7h2w) 6

[**Test towards Quality Attributes**](#_sx1d01tfe76u) **8**

[Performance Testing (QA-P01)](#_qwitqp2d8f6r) 8

[Figure 4. Jmeter report summary](#_hav30521yjkw) 9

[Performance Testing (QA-P02)](#_jdw1oknwzq0o) 9

[Availability Testing (QA-A01)](#_c3vjlx9wj4y2) 9

[Availability Testing (QA-A02)](#_gdqy6ftdnzc9) 10

[**Appendix**](#_vw8t3cz3z8go) **11**

# Quality Attributes (Table)

The quality attributes (table) of the system are as follows: (The detail of each quality attribute is defined in the appendix)

|  |  |  |  |
| --- | --- | --- | --- |
| QA Category | QA ID | Priority | Source |
| Performance | QA-P01 | Medium | CAS Web Service |
| Performance | QA-P02 | Medium | Jenkins |
| Modifiability | QA-M1 | High | Other Developers (Open source community) |
| Availability | QA-A01 | High | CAS service failure |
| Availability | QA-A02 | High | CAS service down |
| Maintainability | QA-M01 | Medium | Administration users |
| Usability | QA-U01 | High | End users |

# Testing Roadmap

The quality control and quality assurance activities should be carried out throughout the entire SDLC, including the following steps:

1. Conduct technical reviews of the content/interface/design model to uncover errors.
   1. Conduct **Design Review** to ensure the consistency and completeness of the project design.
   2. Conduct **Code Review** to ensure the correctness of coding standard, syntax and functionalities.
2. Conduct **Unit Testing** to ensure the correctness of each functional component.
3. Conduct **Integration Testing** to ensure the functional correctness of multiple interacting units through top down approaches.
4. Conduct **System Functionality Testing** to ensure no obvious defects in the workflow.
5. Conduct **System Quality Testing** to ensure the system can meet the specified quality attributes.
6. Conduct the **Acceptance Testing** on a population of end-users to evaluate the usability and reliability of the system to satisfy the client’s requirements.

# Test procedures

### Technical Review

#### *Design Review*

|  |  |
| --- | --- |
| Artifacts | Architecture/workflow diagram and documents. |
| Tools | NA |
| When | After the documents/diagrams have been finished. |
| Whom | The whole team. |
| Precondition | A new version of the design document has been discussed and documented. |
| Postcondition | All team members agree on the consistency and completeness of the new design version. |
| Defect Reporting & Tracking | The defects found should be added to the project’s backlog(Git issue). And be assigned to a specific member to fix. |

#### *Code Review*

|  |  |
| --- | --- |
| Artifacts | Code implementation of the critical functionalities. |
| Tools | NA |
| When | Code implementing the critical functionalities has been developed but has not to be delivered. |
| Whom | The whole team. |
| Precondition | The critical functional requirements have been implemented. |
| Postcondition | The code has been reviewed in the following aspects:   1. Coding standard (Readability/formatting/commenting/consistency/naming/dead code removal) 2. Correctness and efficiency of the algorithm. 3. Correctness of syntax. 4. The code implements the design correctly. |
| Defect Reporting & Tracking | 1. All defects and code improvement issues reported by the code review should be summarized and added to the issue tracker. 2. The defects should be marked with its severity, and all critical level defects should be fixed before delivering the functionality. |

### Unit testing

|  |  |
| --- | --- |
| Artifacts | Backend code of web service. |
| Tools | Django unittest |
| When | After implementation but before integration test. |
| Whom | Two developers test independently. |
| Precondition | The tested method has been implemented |
| Postcondition | 1. All critical functional requirements have been thoroughly tested. 2. All classes/methods should be thoroughly tested. 3. All the unit tests have been passed. |
| Defect Reporting & Tracking | 1. All defects reported by the unit testing were tracked using the issue tracker and were fixed as soon as possible. 2. All defects have to be resolved before the development enters the next stage. |

### Integration Test

|  |  |
| --- | --- |
| Artifacts | Interfaces, components. |
| Tools | Django unittest |
| When | Integration testing should be completed after all the components pass the unit test but before the system test. |
| Whom | At least two developers test independently. |
| Entry Criteria | Components have passed unit testing |
| Exit Criteria | 1. All interfaces (inner-system or cross-system) should be tested. 2. All critical components should be tested. 3. All tests should be passed. |
| Training | Integration test concepts and tools |
| Defect Reporting & Tracking | 1. All defects reported by the integration test shall be tracked using the issue tracker, and they shall be fixed as soon as possible. 2. All defects have to be resolved before the development enters the next stage. |

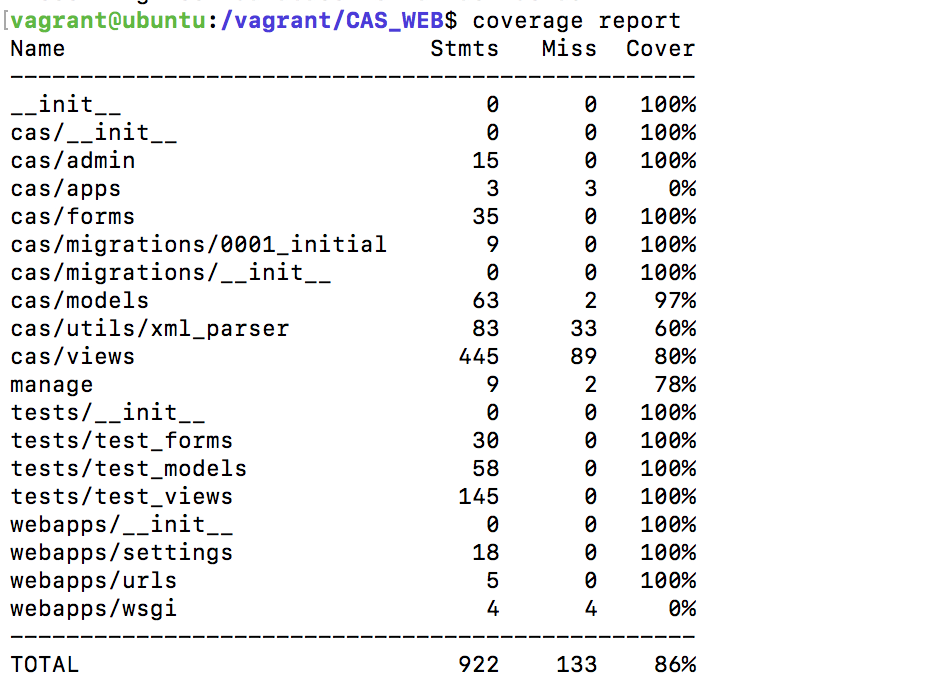


Figure 1. Unit test and Integration test coverage

### Static Analysis

|  |  |
| --- | --- |
| Artifacts | Code |
| Tools | SonarQube |
| When | Every build process. |
| Whom | The developer who is responsible for the piece of code. |
| Entry Criteria | Project is able to be built. |
| Exit Criteria | 1. All coding standard issues founded are fixed. 2. All bugs/vulnerabilities found by SonarQube are fixed. 3. All code smell issues found by SonarQube above critical level are fixed. |
| Training | Coding standard check process & SonarQube. |
| Defect Reporting & Tracking | 1. All defects reported by the integration test shall be tracked using the issue tracker, and they shall be fixed as soon as possible. 2. All defects have to be resolved before the development enters the next stage. |

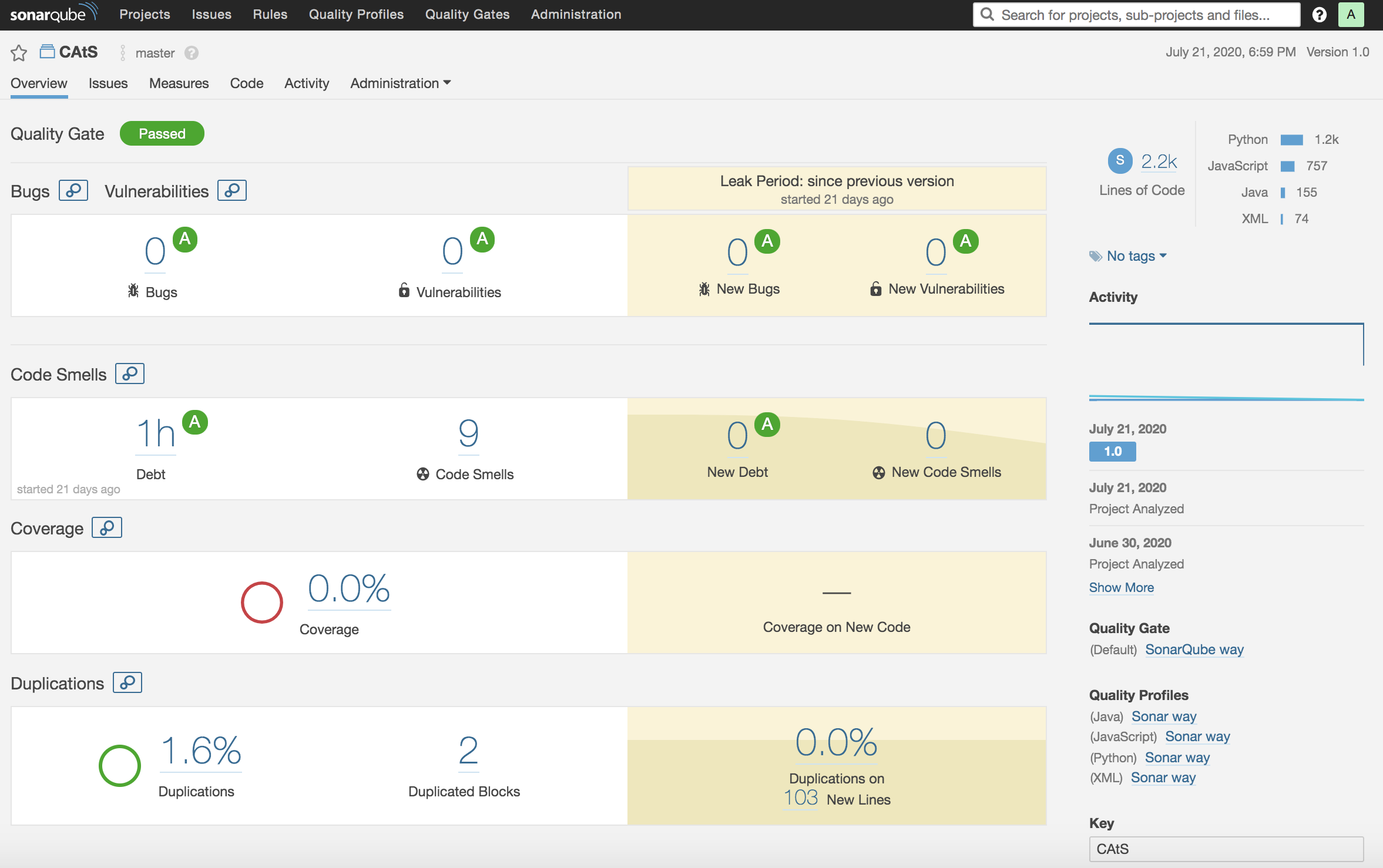


Figure 2. SonarQube static analysis

### System Testing

*Testing towards System Functionality*

The activity diagram showing the system functionality is presented as follows:

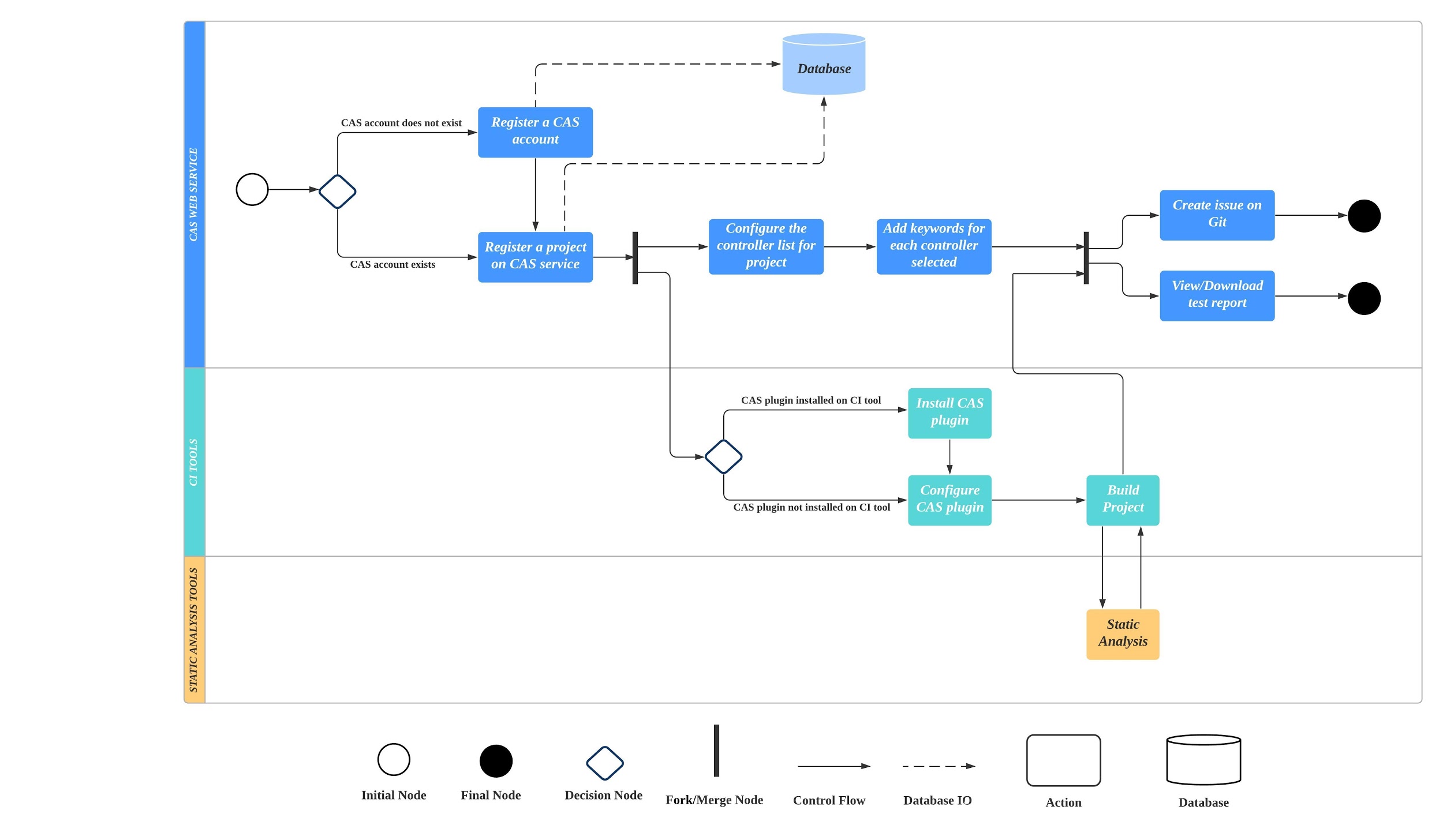


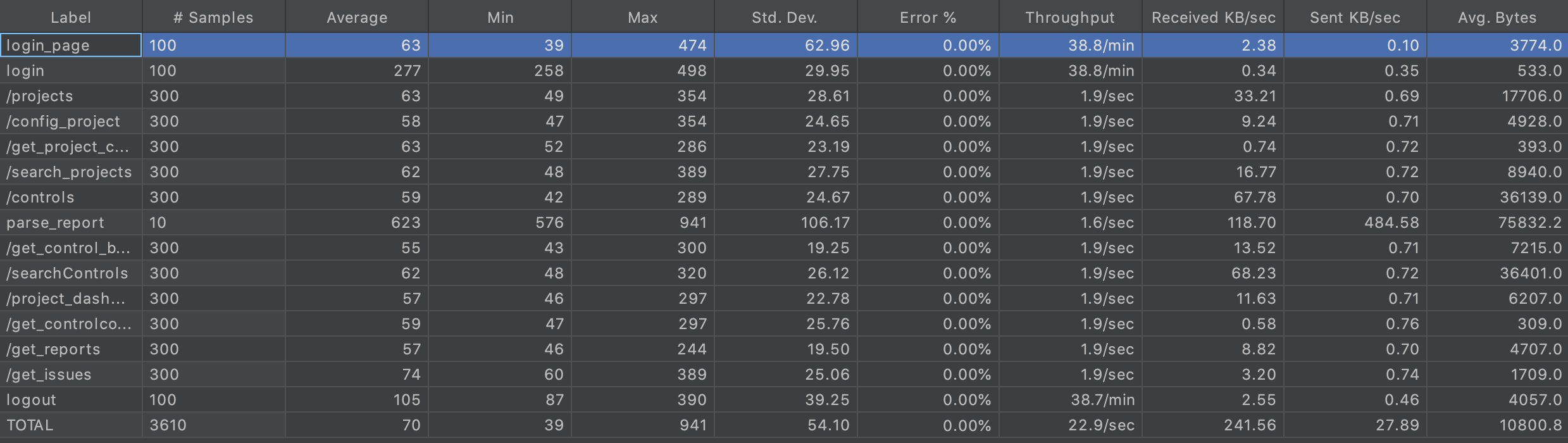
Figure 3. Activity Diagram

|  |  |
| --- | --- |
| Artifacts | The entire project |
| Tools | NA |
| When | Before each release |
| Whom | Developers |
| Entry Criteria | 1. All unit & integration testing has passed. 2. The project is about to be released. |
| Exit Criteria | 1. All action nodes and decision nodes are tested. 2. All issues related to the critical functionalities have been fixed and verified. |
| Effort Estimation | 10 hours/release |
| Training | NA |
| Defect Reporting & Tracking | 1. All defects reported by the system testing shall be tracked using the issue tracker, and they shall be fixed as soon as possible before releasing. 2. The status of all the issues should be updated in time. |

# Test towards Quality Attributes

#### *Performance Testing (QA-P01)*

|  |  |
| --- | --- |
| Artifacts | The web service |
| Tools | JMeter |
| When | 1. It is required to conduct performance testing before each release. 2. Additional performance testing can be conducted when the development is in a stage of tuning system performance. |
| Whom | One team member in charge of analyzing system performance. |
| Entry Criteria | The system has passed all unit tests and integration tests. |
| Exit Criteria | The system performance has met the required performance defined in the quality scenarios (ID: QA-P01) or has been accepted by the clients. |
| Effort Estimation | 3 hours/release |
| Training | System profiling techniques. The usage of JMeter. |
| Defect Reporting & Tracking | * The result of each performance test should be properly recorded with the issue tracker. Including the testing date, average response time and ‘95% response time’ for each RESTful API. |



#### Figure 4. Jmeter report summary

#### 

#### *Performance Testing (QA-P02)*

|  |  |
| --- | --- |
| Artifacts | The web service |
| Tools | NA |
| When | It is required to conduct performance testing after the core functionalities of both CAS and Jenkins plugin are ready to be released. |
| Whom | One team member in charge of analyzing system performance. |
| Entry Criteria | The system has passed all unit tests and integration tests. |
| Exit Criteria | 1. Conduct performance testing under normal load repeatedly (not less than 5 times). 2. Manually record the duration of the CAS to update the latest result on the dashboard after a new report is generated in Jenkins. This average duration should be less than 120 seconds. |
| Effort Estimation | 3 hours/release |
| Training | System profiling techniques. |
| Defect Reporting & Tracking | The result of each test should be properly recorded in the issue tracker. |

#### *Availability Testing (QA-A01)*

|  |  |
| --- | --- |
| Artifacts | CAS web service |
| Tools | JMeter |
| When | It is required to conduct performance testing before each release. Additional performance testing can be conducted when the development is in a stage of tuning system performance. |
| Whom | One team member in charge of analyzing system availability. |
| Entry Criteria | The system has passed all unit tests and integration tests. |
| Exit Criteria | 1. Use JMeter to generate HTTP requests to CAS service for 24 hours. Calculate the percentage of successful requests as the availability percentage. 2. The system met the required availability percentage defined in the quality scenarios (ID: QA-A01) or has been accepted by the clients. |
| Effort Estimation | 3 hours/release |
| Training | System profiling techniques. The usage of JMeter. |
| Defect Reporting & Tracking | The result of each availability test should be properly recorded in the issue tracker, including the availability percentage and testing date. |

#### *Availability Testing (QA-A02)*

|  |  |
| --- | --- |
| Artifacts | Jenkins |
| Tools | N/A |
| When | 1. It is required to conduct performance testing before each release. 2. Additional performance testing can be conducted when the development is in a stage of tuning system performance. |
| Whom | One team member in charge of analyzing system availability. |
| Entry Criteria | The system has passed all unit tests and integration tests. |
| Exit Criteria | 1. The configuration files and reports during the CA service shutdown time exists. 2. The content of the saved configuration files and reports are correct |
| Effort Estimation | 3 hours/release |
| Training | System profiling techniques. |
| Defect Reporting & Tracking | The result of each availability test should be properly recorded in the issue tracker, including the availability percentage and testing date. |

# Appendix

#### *Performance*

|  |  |
| --- | --- |
| ID | QA-P01 |
| Category | Performance |
| Priority | Medium |
| Source | CAS Web Service |
| Stimulus | Receives a new HTTP request from client |
| Artifact | CAS web service dashboard |
| Environment | CAS web service running with the normal condition (less than 50 concurrent users) |
| Response | The browser shows corresponding contents on the web page. |
| Response Measure | * Average response time should be less than 2 seconds. * The response time of 95% request should be less than 5 seconds. |

|  |  |
| --- | --- |
| ID | QA-P02 |
| Category | Performance |
| Priority | Medium |
| Source | Jenkins |
| Stimulus | A new testing result report generated by Jenkins. |
| Artifact | CAS web service dashboard |
| Environment | CAS web service running user’s jobs |
| Response | The dashboard shows the latest report of the result generated by Jenkins. |
| Response Measure | The report is updated on the dashboard within 2 minutes. |

#### *Modifiability*

|  |  |
| --- | --- |
| ID | QA-M1 |
| Category | Modifiability |
| Priority | High |
| Source | Developers in the open source community. |
| Stimulus | A directive to add new plugins for letting CAS compatible with other CI services (e.g. Travis). |
| Artifact | New plugin components |
| Environment | Other developers are trying to modify CAS with new CI services. |
| Response | The new plugin component should be compatible with CAS and tested. |
| Response Measure | * The response from other developers in the open source community shows that the development time for additional CI plugins is acceptable (Within a week). * The only part shall be affected for adding additional CI plugin is the CI plugin itself. |

#### *Availability*

|  |  |
| --- | --- |
| ID | QA-A01 |
| Category | Availability |
| Priority | High |
| Source | CAS web service failure |
| Stimulus | The CAS web service is down |
| Artifact | Continuous Authorization Service (CAS) |
| Environment | CAS web service running user’s jobs |
| Response | The CAS service should restart itself. |
| Response Measure | * The availability percentage of CAS service should be more than 90% |

### 

|  |  |
| --- | --- |
| ID | QA-A02 |
| Category | Availability |
| Priority | High |
| Source | The CAS web service is down |
| Stimulus | CA service is down or is closed by the administrator. |
| Artifact | Jenkins plugin |
| Environment | Run time |
| Response | The testing reports should not be dropped. The reports should be saved in Jenkins\_home and be sent to CA service after it is back to work. |
| Response Measure | 1. All the reports during the CA service shutdown time should be saved. 2. The reports should be sent to CA in 1 minute after CA recovering. |

*Maintainability*

|  |  |
| --- | --- |
| ID | QA-M01 |
| Category | Maintainability |
| Priority | Medium |
| Source | Administration users |
| Stimulus | Administration users maintain the Jenkins server, plugins, and configurations. |
| Artifact | CAS Web Service, Jenkins |
| Environment | Design time |
| Response | The admin user should be able to localize and correct the errors as well as preserve the modification and extension of functionality. |
| Response Measure | 1. The debug process should be able to complete in one-person day. 2. Admin users should be able to extend or modify the functions of CAS without affecting the existing configurations |

*Usability*

|  |  |
| --- | --- |
| ID | QA - U01 |
| Category | Usability |
| Priority | High |
| Source | End users |
| Stimulus | Try to learn and configure the CAS Web service. |
| Artifact | Runtime |
| Environment | CAS Web service, Jenkins |
| Response | Users can use the CAS web service productively. |
| Response Measure | Finishing configuration task with 20 minutes. |