# Quality Assurance plan

1. **Abstract**
   1. **Introduction**

While developing a dashboard for integration in industrial settings, it is vital that we ensure that there are no issues that could cause damage to the product, environment, or equipment involved. To prevent this, we must implement a consistent process of testing and review that we’ll ensure we detect potential issues before they are introduced into a production environment.

* 1. **Purpose**

The follow is a plan to establish guidelines to Assure the quality of any code developed for this project. Defining the expectations for the testing of any developed feature or code will assist in ensuring that any problems that are found will be consistent and efficient.

* 1. **Scope**

The Quality Assurance Plan focuses on the testing of features as they are developed along with the documentation of those tests along with the process and documentation of code reviews and Q.A. Evaluation.

This plan is necessary to ensure that no feature we develop will cause damage to the warehouse environment that the system is to be deployed into and the machinery that it is implemented with.

* 1. **Policy Statement**

We will strive to produce an effective and quality product that will modernize Jadcup’s production line.  
In our endeavor to achieve this we will:

* Nurture a team culture that strives for quality.
* Ensure internal testing standards.
* Create an environment where feedback is encouraged.
* Record testing results for internal evaluation of current Q.A. procedures.
* Review of QA Metric results to identify common issues in development.
* Ensure that any requirements proved by the client are fulfilled.

1. **Management** 
   1. **Organizational Structure**

Members of the Quality Assurance and Technical Team report to Senior Management who further reports to the client. However, it is the responsibility of the Q.A./ Technical Team to develop, plan, and execute Testing and Reviews as they occur.

Because we are utilizing a Kanban methodology we will not be utilizing a QA Manager who would focus on developing acceptance tests and test code. Instead a member of the QA team will take on that role for a specific card on our Kanban board.

* 1. **Roles and Responsibilities**
     1. **Senior Management**

Matthew Kuo will act as Senior Management for Quality Assurance, however in this role his main responsibilities will be acting as a third party for any issues that could arise in the process of Q.A. disputes.

* + 1. **Task Lead**

Any task listed on the Kanban board will be assigned a task leader who is responsible for delivering the item for testing and review, this member will not be able to act as a member of the QA Team for that specific item.

* + 1. **Quality Assurance/ Technical Team**

The Q.A. and Technical Team will be a single team made up of the Development team.

* + - 1. **Quality Assurance Team**

Quality Assurance Team members are responsible for the testing of cards and recording the results.

* + - 1. **Technical Team**

Technical Team members undertake the process of code walkthroughs and review the tests that have been done.

1. **Q.A. procedure**

The Key to Ensuring our Q.A. Procedures are moving smoothly is using our Kanban Board on Trello to visualize the process and make it easy to detect at a glance what should be prioritized. Utilizing the Trello environment, we are able to assign developers and add reports and other documents for ease of communication as well as allowing us to work remotely.

While typically the Q.A. process is underdone when members of the respective teams are available, should a test need to be expedited then the card can be colored red so it is visually marked as urgent additionally members of the team will be alerted by the Task Lead that this card must be done quickly. A card may require an expedited timeline due to its importance to the stability of the project or because it is holding up further development of other cards. Any card that is marked to be expedited will not be counted against the flow rate of the systems.

* 1. **Testing**

Once a developer has finished working on a given card the developer utilizes the Kanban Board to move the card from the *In Development* column to the *Waiting for Testing* column,adding a link to the corresponding push request through Github. Once the card has been moved to the new column a member of the Q.A. Team can be added to create unit tests, integration tests, and/or, Functional tests depending on the requirements of a given card. Any test that is used should be run several times to ensure consistency.

Should the tests fail, the card will be moved back into the In Development Column, the push request will be declined, and the Task Lead will be notified. Should the Testing succeed, the card will be moved into the *In Review* column.

Once testing has concluded then a Testing Results matrix will be attached to the card as outlined in *4.1 Testing Matrices.* And records any applicable data into the Metric Record as outlined in *4.3. Q.A. Metric Record.*

* 1. **Code Walkthrough**

The Code walkthrough is key in ensuring that testing and coding standards are being upheld but also can act as a teaching tool where members of the development team they are less knowledgeable about a topic can learn.

When a Card has been tested and has moved into the *In Review* Column then a review of both the card and code is undertaken, the review process is underdone by members of the technical team that did not work on developing the card, this includes the member of the Q.A. team that produced the tests.

In the case where there are no eligible members of the technical team remaining, the walkthrough will be done with a selection of the Developer team that feels they did not work on enough amount of the feature to be biased in any way.

During the Walkthrough process the Technical Team will review both the code of the feature and testing that has been done. Should the team find that more tests are required then the card will return to the *In Testing* Column. If the team determines that the code overreaches the scope of a given card, then they will reach out to the Task Lead for justification. If this Justification is not sufficient then the card will be moved back to the *In Development* column. Otherwise once review is completed the Technical Team will move the card to the *Release* column.

Once the code walkthrough has concluded for any of the above reasons a Code Review document will be attached to the given card as outlined in *4.2. Code Review Reports*.

* 1. **Q.A. Evaluations**

Evaluations are the processes we utilize to ensure the products overall health, through the review of the Q.A. Metric Record asan overview of testing results and any key or repeating issues can be found and a discussion on possible changes can be undertaken.

Evaluation reviews of the Metric Record, along with any key issues that are detected and collated into an Evaluation Report such that it can be used as evidence and or justification in a change request.

1. **Required Documentation**
   1. **Testing Matrices**

Testing Matrices are utilized to record and display the testing that has occurred on a given card. These matrices are designed so that anyone who views them can quickly review the tests that have occurred, test description, the expected outcomes, and the actual outcome. Along with dates and who developed the tests.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test type | Test # | Test Description | Expected outcome | Actual Outcome | Test Pass |
| Unit Test |  | Testing for correct read of machine output |  |  | Test fail |
|  | 1 |  | 159 | 28 | Fail |
|  | 2 |  | 159 | 28 | Fail |
|  | 3 |  | 159 | 28 | Fail |

* 1. **Code Review Reports**

Code Review Documents are used to identify issues and determine any weak points in our testing and development cycles. The document records any notes that the Technical Team has regarding any aspect of either tests used or code written. The document is key in detailing any reason that a card has failed the walkthrough and may be shorter if there are no such issues.

* 1. **Q.A. Metric Record.**

The Metric Record is used to collate and record results from Testing matrices and Review Reports. The document acts as a living document that allows us to at a glance determine any issues that occur. The metrics recorded and the reasons for them can be found in *6.0 Quality Assurance Metrics.*

* 1. **Evaluation Reports**

Evaluation Reports are Documents that record evaluation sessions and document what metrics and reoccurring issues have been detected, detailing specific instances if required. The document is detailed just that it can be used as evidence in a Change Request Form and should be comprehensive enough to not require someone to review individual Review Reports and Test Matrices.

1. **Problem Reporting Procedures**
   1. **Noncompliance Reporting procedure**

Should a member of the Q.A. Team or Technical Team find that cards are being repeatedly submitted without the required changes having been resolved then that member should bring this to attention in the following meeting and a discussion must be made regarding why the changes are not being made.

Should this become a recurring issue the problem will be escalated to Senior Management members for assistance in finding an amicable resolution.

1. **Quality Assurance Metrics**

The following Section outlines what Metrics we will use to gage the effectiveness of our QA process and assist in identifying repeating issues.

* 1. **Test Coverage**
     1. **Test Execution**

Test Execution measures the number of tests we run and their results. Results are classified as pass or fail. Can be used alongside other metrics or alone.

* + 1. **Tests Per Card**

TPC Measures how many tests we are running on a given card, this metric allows us to identify if we are under/over testing our cards. Can be used with other metrics or alone.

* + 1. **Bug per card**

This metric allows us to find how effective our methods are at detecting bugs in a system. It is important to know that this metric does not need to be high but should act as a tool alongside other metrics such as test Execution and TPC to measure quality in our testing.

* 1. **Bug distribution**

Bug Distribution Allows us to identify common sources of issues such as integration, performance, security, unit level, eta. By measuring this distribution, we can identify weak points in our skills and determine where we should be putting extra focus during our testing.

1. **Quality Standards**
   1. **Definition of Ready (DoR)**

The Definition of Ready defines how we decide if a user story is viable to start developing on. By using this metric, we can guarantee that we are not starting work that we cannot finish. For example, if a user story requires a database to be running and the story that develops the database has not been finished, then that user story should not start.

* 1. **Definition of Done (DoD)**  
     The Definition of Done is a metric used to ensure that all developers understand what an outcome should look like when deciding if their systems are ready to be considered finalized. By using this metric, we can standardize the both the testing and review stages to ensure that there are no discrepancies between user stories as well as ensure that user stories are not entering testing and review stages before they should.  
     When considering if a user story meets the DoD, then we must consider the following for the following stages:
* Development:
  + The code only affects necessary systems.
  + Developers have reasonable expectations that the code will pass testing and review.
  + Code achieves user story.
* Testing:
  + The code passes unit tests.
  + The code passes integration tests.
  + Test matrix filled and attached to card.
  + Matrix added to metric record.
* Review:
  + Code fulfills programming standards.
  + The code is well documented.
  + Testing done is relevant and effective.
  + Evaluation report written and attached to card.