Definition of Done

Team: Diego, Issai, Brian, and Anthony

Product: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |
| --- |
| As a team, before saying that an item of the Sprint Backlog is Done, we agree that it will meet the following:  Highlighted Yellow – Agree  Red – Disagree  Highlighted Gray - Neutral   * The code is complete and according to developers’ standards. * The code is refactored. * Technical Debt is removed. * Meet the acceptance criteria. * Code checked-in to the repository. * Unit tests are written and green. * Test coverage: \_\_ %. * Pair programming. * Peer review. * Code merge and tagged. * Deployed to the development environment. * Integration tests are written and green. * Deployed to the QA environment. * Bugs, improvements, and changes are done. * Changes were communicated and updated in the ticket description. * Regression tests. * Performance tests. * Approved by QA. * Deployed to UAT. * Reviewed and accepted by the Product Owner. * Reviewed and accepted by the customer or user(s). * Final approval in the Sprint Review or demo by the Product Owner and customer. * Deployed to production. * Production tests are successful. * Make sure the things that take priority are done before going to the more minimal things * A 100% agreement from the team that it is done, no add ons or bugs it is good to go. * Make sure the team agrees on the state of something before showing it on a sprint. * When someone tests our product or a piece of it for the sprint and if they have no notes. We can consider that piece done. * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ * \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

https://lh6.googleusercontent.com/v9ZuOVmv15szbq5H6gAsok75mQbjLh5Ktzq2TUptv4eG1kwKPmspeuVOyQAeX5m-6StU3zldeRLjmnh0KXFVY1Xczs8WeH3xe0MeQ1BWghkVWolB39FsqY-5knI7KOFAmYevqpQMryHPNIija3G1kA

This material is licensed under Creative Commons Attribution 3.0 License, except that the reference to the author and all links included here must be referenced ([Ignacio Paz](https://www.linkedin.com/in/ignaciopaz)).

Definition of Ready

Team: #TeamNiceDynamite\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Product: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Before adding a product backlog item to a Sprint backlog, as a team, we agree that it will be refined until:

Independent

* Story can be developed **Independent** from other stories.
* **Refined** to get a **functional cohesion**.
* If there are **dependencies**, they were refined and separated.
* **One team** can work on it with minimal or no dependencies on other teams.
* **One team member** can complete the work with minimal or no dependencies on others.
* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Negotiable

* The story was **discussed** with the team and they understand what to do.
* The **Acceptance criteria** is defined.
* There is a high level **technical solution**.
* The team has what they need to **start working** on the story but some details will be discovered during the development.
* **A/B** variants are defined.
* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Value

* The **priority** in the backlog is clear.
* Prioritized using **MoSCoW** method.
* The **Value** of the story is quantified.
* The team understands **Why** we do this and the expected **impact** on the user
* The **Role** who obtain the value of the story is clear.
* The **stakeholders** (would) agree with the **Acceptance criteria** and **value**.
* It is clear if should be part of the next **Product increment** or **MVP** (minimum viable product).
* **Hypothesis/experiment**, expected outcome, metrics and measurement period is defined.
* **Return of Investment**: Stakeholders and PO know how much they will obtain with this story.
* As <**Role**> I want <**Feature**> so that <**Benefit**> title template.
* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Estimated**

* **Planning poker** session done with the team.
* Estimated in **Tee-shirt sizes**.
* Estimated in **Story points** by the team**.**
* All stories have **similar size**.
* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**SMALL**

* **Small**: Estimation size less than or equal to \_\_ points.
* It is not an **Epic**.
* It can be done in **One Sprint**.
* **Refined** to obtain **80% of its value** using 20% of its effort: The minimum size that can reach value was achieved**.**
* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Testable**

* The team understands how the story can be **tested**.
* **Acceptance tests** and examples are defined.
* Scenario-oriented Acceptance criteria Given-When-Then.

Develop an initial Product Backlog – an ordered list of everything that might be needed in the product

* + Have at least 3 user stories
  + Have at least 5 technical requirements
  + Suggest an ordering of your items …
    - 1. Define items you can do in about 2 weeks at top of list
      2. Make those items adhere to DoR.
      3. Meet with me to review

What We Agree On

* Integration tests are written and green.

We agreed on this one because of course we need to make sure that our app and its functions work. If it's in the red then we need to keep working on it.

* Bugs, improvements, and changes are done.

We agreed on this one because through our sprints and user feedback from tests. We know what needs to be changed and improved on. Also, any bug that is discovered by us or the user needs to be dealt with immediately.

* Final approval in the Sprint Review or demo by the Product Owner and customer.

We agreed on this because we need approval on any direction we are going with on the application by the owners. Otherwise, we could be straying away from the original idea that it could be seen as unrecognizable. We also need the final say from the customer to say that we finished the feedback we were given from them.

What We Disagreed On

* Pair Programming

We disagreed with this one because even though the backend and frontend will be working together in some capacity. Both working together in the same space would probably cause priority issues/

* Performance tests

We disagreed on this one because none of us know how to test performance. This could change somewhere during the semester but as of now, we don't know how.

* Deployed to the QA environment.

We disagreed on this one because our product would not be fully ready for a QA environment to test the application.

User Stories

* 1. As an employee of Aldi, I want to be able to sign in so that I can request my PTO days.
  2. As an employee of Aldi, I would like to view how many PTO days I have so I can correctly see how much time I have left before they expire.
  3. As the supervisor I would like to be able to approve or deny the days off an employee sends off to me.

Technical Requirements

1. On the login page, the user will input their employee Id but here they will also create a password for the app.
2. The system will have a message pop up on the user’s screen when they have submitted their days off. This notification will appear in 2 or 3 seconds. This will stay on the screen until the user clicks the x button on the corner of the notification.
3. When HR approves or denies the days off from the employee. The employee should be notified of this within 2 or 3 seconds. It will probably be 5 seconds if things get slow on the servers.
4. Lower level and higher level employees, the higher level being managers, HR, and bosses. See their specific views on the application. Regular employees see the employee view and HR and higher see the admin view.
5. Should an employee forget their password, they can click on the "Forget Password?" button on the log-in screen. From there they can be taken to a page where they will be prompted to enter their employee ID and a brand new password, as well as an input box to retype the password to verify it.

Ordering of Items (Top 3 can get done in 2 weeks)

1. Authentication and Authorization because we can get a sign-in going
2. Accessibility is something that seems very easy to get done
3. Human error we can figure out in two weeks because we can get co-workers to test and see what errors they can come up with
4. Data Quality can be handled within two weeks.
5. Privacy
6. Internal controls.