**Agile With Atlassian JIRA**

**Agile Approach:**

Agile is an approach to managing and working on projects that combines project management and product development.

It is a simple approach to managing complexity and is seen as an alternative to complex project management and heavy upfront planning.

Agile can be applied to any type of project, but it is most commonly used in software development.

Agile techniques can also be used in various departments within a company, such as human resources and marketing.

**Characteristics of Agile Projects:**

Agile projects are built incrementally, meaning that small valuable increments of the product are planned and released successively.

They are iterative, allowing for continuous feedback, learning, and improvement of both the product and the process.

Agile projects relentlessly focus on value, always working on the highest value parts of the project based on feedback.

They have an empowered team where members make decisions collectively and have the most current knowledge to make informed choices.

**Benefits of Agile Projects:**

From the customer's perspective, Agile projects deliver a desirable product by consistently incorporating customer feedback and delivering high-priority features early.

Agile projects lead to higher quality by continuously addressing mistakes and issues as they arise.

They foster higher job satisfaction among team members by empowering them to make decisions and utilize their skills and creativity.

Agile projects encourage innovation by allowing for experimentation and quick testing of ideas with real users.

They result in lower costs by focusing on value and avoiding wasteful activities, as well as resolving problems promptly.

Agile projects are safer as they continuously receive feedback, reducing the risk of building low-value features.

They offer predictable deliveries by ensuring shippable increments, even if the number of features may vary.

**The Scientific Method and Agile:**

Agile projects are based on the scientific method, which involves formulating hypotheses, conducting experiments, observing results, and learning from them.

Agile projects apply this iterative learning loop not only for discovery but also for accomplishing the work of a project.

The scientific method serves as a formalized description of problem-solving, which is an integral part of project work.

Various concepts and processes, such as "plan, do, check, act" or "think, build, ship, tweak," are derivatives of the scientific method and are used in Agile projects.

**Comparing Agile to the Waterfall Approach:**

The traditional waterfall approach manages projects by developing the entire product in distinct phases without looping or iterations.

The waterfall approach is similar to traditional mass production and relies on large batch sizes and economies of scale.

Waterfall projects face several downsides, including the upfront plan often being wrong, building unnecessary features, underestimated time and complexity, inability to adapt to market changes, difficulty and costliness of change, creation of obsolete documents, delayed feedback, and limited flexibility.

**Usage of Waterfall Approach:**

The waterfall approach is still used in certain situations where setup costs for each phase are high or when the work is relatively predictable.

For example, in the past, setting up computers and running tests required significant time and expense, leading to batching of testing in waterfall projects.

However, as the setup costs for project phases decrease and the work becomes less predictable, Agile approaches become more favorable and the waterfall approach becomes outdated.

**Introdução ao Jira:**

O Jira é um software usado para ajudar a gerenciar, desenvolver e comunicar projetos.

Pode ser usado em projetos individuais ou em equipe, sejam eles complexos ou simples.

O Jira é flexível e pode ser adaptado aos processos ágeis ou ao modelo cascata de gerenciamento de projetos.

**Hierarquia do Jira:**

Níveis de hierarquia: aplicação, projeto e problema (issue).

A aplicação Jira contém vários projetos.

Um projeto contém um ou mais problemas relacionados ao trabalho.

Um problema é uma tarefa ou trabalho a ser realizado no projeto.

**Navegação no Jira:**

Os projetos podem ser visualizados no menu suspenso "Projects" na barra de navegação superior.

Também é possível acessar projetos recentes ou problemas recentes por meio da opção "Your work".

Dentro de um projeto, há uma barra lateral com links específicos para o projeto atual.

**Tipos de projetos:**

O Jira oferece dois tipos de projetos: classic e next-gen.

Projetos classic são mais tradicionais, com mais funcionalidades e configurações compartilhadas.

Projetos next-gen são mais recentes, oferecendo maior facilidade de criação e configuração por membros da equipe do projeto.

**Administração no Jira:**

Há diferentes níveis de administração: site, Jira e projeto.

Um administrador de site tem acesso a toda a plataforma.

Um administrador do Jira pode controlar a criação de projetos e configurações que se aplicam a vários projetos.

Um administrador de projeto pode configurar especificamente um projeto em particular.

**Configurações e personalização:**

O Jira é altamente configurável e possui diferentes opções de personalização.

As configurações podem ser acessadas por meio do ícone de engrenagem na barra de navegação superior.

Existem configurações gerais do Jira, configurações de projeto e configurações pessoais.

**Conta e configurações pessoais:**

Cada conta do Atlassian Cloud tem configurações pessoais que podem ser personalizadas.

As configurações pessoais se aplicam a todos os sites associados à conta.

As configurações relacionadas a um site específico do Jira podem ser acessadas por meio das configurações pessoais.

**Recursos adicionais e aprendizado contínuo:**

Existem muitos recursos disponíveis para aprender mais sobre o Jira. Links para a página inicial do Jira, documentação, fóruns de discussão, treinamentos e canal do YouTube do Atlassian são fornecidos.

Pesquisas na web também podem fornecer informações úteis sobre tópicos específicos do Jira.

**Principles of Visualizing Work**

Visualizing work is an important principle of Agile.

A to-do list is a simple tool that visually reminds you of the work that needs to be done and helps you focus.

Prioritizing work items on a to-do list is a way to manage and organize tasks.

Tracking progress by checking off completed work items is rewarding and provides a sense of accomplishment.

Visuali zing work using tools like boards helps in organizing, focusing, and managing the work of a team.

Boards can be physical (e.g., whiteboard with sticky notes) or digital (e.g., software-based like JIRA).

Boards allow everyone to see the current state of the project, making it transparent to both the project team and stakeholders.

Visualizing work helps identify problems or bottlenecks in the process, facilitating continuous improvement.

Visualizing work can be applied to personal projects to increase productivity and effectiveness.

**Boards and Workflows**

Boards are Agile tools used to visualize and manage the work of a team.

Boards can be task boards, project boards, kanban boards, or scrum boards, depending on the context.

Boards can be physical or software-based, with examples like a whiteboard or JIRA.

Boards consist of columns that represent different stages or statuses of work items.

Work items (issues) are moved across columns as the team progresses through the project.

Boards provide a two-dimensional representation of work, allowing work items to go through multiple steps before completion.

Workflows are used to model the processes involved in a project, breaking down work into a series of steps.

Workflows and boards are closely related, with the board visualizing the workflow.

Boards and workflows can be customized and configured to match the team's needs and changing workflow over time.

**Configuring Boards and Workflows in JIRA**

JIRA automatically creates boards when projects are created using Kanban or Scrum templates.

Multiple boards can be created for a project, and a board can contain issues from multiple projects.

Workflows are associated with JIRA projects and define the available statuses for issues.

Changing an issue's status on a board involves moving it between columns, which represents a transition.

Boards can be configured to match the team's workflow by adding, removing, renaming, or modifying columns.

Column categories (to-do, in progress, done) help identify the status of an issue in its life cycle.

JIRA's simplified workflow is the default, but customized workflows can be managed to enforce step-by-step movement of issues.

Board cards display a small number of field values for an issue, such as type, priority, assignee, and additional custom fields.

Card layout and colors can be configured in JIRA to display relevant information and improve visual representation.

**Kanban Method Overview**

Agile methods are approaches used to achieve agility in project management. They are sometimes referred to as frameworks or methodologies. Agile is more of a mindset than a specific project management approach. Agile methods provide some structure to the agile ideas. Kanban and Scrum are common agile methods, along with others like Extreme Programming (XP). Although these methods differ, they all embody the core principles of agility, such as empowering the team, continuous improvement, working in small batches, and delivering value. Teams often combine ideas from different methods to create a customized approach for their specific needs.

The Kanban method is an agile approach commonly used to manage a continuous queue of work items or issues. It draws on ideas from the Toyota production system. Some key concepts in the Kanban method include limiting work in progress, ensuring the team only takes on a sustainable amount of work at a given time. It also emphasizes removing bottlenecks to improve workflow. While the goal is to achieve a steady flow of work, bottlenecks can occur due to process problems or the complexity of the issues. The team collaborates to identify and eliminate these bottlenecks, gaining a holistic view of the entire process. Addressing bottlenecks involves identifying the root cause and making process improvements.

The Kanban method promotes a "pull" approach to work, where individuals responsible for the next step in the process pull work from the previous step when they are ready, instead of having it pushed onto them. This helps maintain a smooth workflow.

**Why would a team choose to use the Kanban method?**

Kanban is a lightweight and efficient agile method. Compared to other methods like Scrum, it is even lighter and simpler. It provides a bare-bones approach to achieving agility, making it easy to understand and start using. Some teams find Kanban more effective than other agile methods. Additionally, Kanban can be adopted as an evolutionary approach to transitioning to agility. It allows teams to utilize existing roles and processes without requiring extensive reorganization or the introduction of new meetings or roles. Teams can begin implementing Kanban immediately and continually improve their agility over time.

The Kanban method works well for service-oriented workflows. It can be applied to various areas such as operations teamwork, support requests, maintenance development, and human resource departments' new hire processes. Anywhere there is a continuous flow of work can benefit from the Kanban method. However, Kanban can also be used for product development. It supports multi-team and multi-project workflows, allowing issues to be moved among teams using a single board or dedicated boards for each team. Each team has the flexibility to handle the work in their own preferred way.

**Kanban Boards:**

Like other boards, a Kanban board is a tool used to visualize a team's workflow. The default Kanban board in Jira consists of four columns: Backlog, Selected for Development, In Progress, and Done. When issues are created, they automatically appear in the Backlog column. Issues in the backlog may require further discussion or planning before they are ready to be worked on. When an issue is ready, it is moved to the Selected for Development column, which represents the team's prioritized to-do list. When a team member is ready to work on an issue, they pull it from the Selected for Development column into the In Progress column. Once the work on an issue is completed, it is moved to the Done column.

Kanban boards provide a simple and effective way to visually manage a team's work. While the default configuration is shown, the column layout can be customized to match the team's specific needs. Kanban boards focus on improving the flow of issues through the workflow. New issues continuously enter the backlog and are prioritized, usually by the business team. Once ready, they move through the workflow statuses. Work is completed before new work is started, ensuring the team always works on the most important issues as defined by the team.

**Separating the Kanban Backlog from the Kanban Board:**

In Jira, the backlog column on a Kanban board can be separated from the rest of the board, offering several advantages. It allows the development team to focus on issues they can work on, as backlog column issues are not ready yet. A separate backlog column is easier to manage, and its contents are not visible to the rest of the team.

To move a column from the Kanban board to the Kanban backlog, you can access the board's settings, go to the Columns tab, and drag the desired status or statuses to the Kanban backlog section. Once configured, the Kanban board no longer displays the backlog column. Instead, Jira adds a backlog tab to the sidebar, providing visibility and management of the backlog. The issues in the backlog are presented as a list, allowing for easier editing. Above the backlog, the first column of the board facilitates moving issues from the backlog onto the board, enabling the team to view and work on them. Modifying an issue's status from the backlog causes it to move out of the backlog and onto the Kanban board.

Kanban is a lightweight agile method that utilizes Kanban boards to manage the continuous flow of issues from backlog to done. In Jira, the Kanban backlog can be separated from the Kanban board, simplifying the board, and allowing separate backlog management.

**Work in Progress Limits (WIP):**

Limiting the amount of work in progress (WIP) on a Kanban board has several advantages. By setting minimum and/or maximum limits for the number of issues in specific columns, the flow of work improves. This means that fewer issues are being worked on simultaneously, resulting in a smoother and more efficient workflow. When work in progress is limited, the team focuses on completing the existing work before starting new issues. This approach reduces multitasking, which can be detrimental to productivity.

One of the benefits of limiting work in progress is faster issue delivery. With fewer issues in progress, they are less likely to pile up in certain columns, waiting for work to be restarted. As a result, the team can deliver completed work to the customer more promptly. Moving issues into the "done" status becomes a primary measure of the team's progress.

Moreover, limiting work in progress helps quickly identify any bottlenecks in the process. Since there are relatively few issues in progress at any given time, any problems or obstacles become more visible and can be addressed promptly. This continuous improvement of the team's work contributes to enhanced efficiency.

Another advantage of work in progress limits is the reduction of waste in the process. When there is a backlog or accumulation of issues in a particular status, it indicates a delay in working on those issues, resulting in wasted time and effort. Limiting work in progress ensures a more streamlined workflow and minimizes the need for rework caused by problems in earlier steps.

Additionally, work in progress limits foster teamwork. By restricting the number of issues in a particular status, team members are encouraged to collaborate and resolve any blockages together. This promotes a sense of shared responsibility and a collective effort to clear any obstacles.

To set up work in progress limits for a column on a Kanban board, access the Columns tab in the board settings. In Jira, these limits are referred to as Column constraints. You can define constraints based on the number of issues or exclude sub-tasks from the count. The column constraints are visible below the column name, indicating the current minimum and maximum constraints.

Determining the appropriate work in progress limits depends on the specific project and team dynamics. It is recommended to start without any limits initially and observe how the flow of issues progresses. If issues are not flowing smoothly or if there are instances where there are no items in a particular column, it might be necessary to introduce minimum constraints to ensure a continuous flow of work. Work in progress limits can also be set to discourage multitasking, such as having a limit of one issue per team member at a time. Additionally, limits can be placed on specific steps that tend to be neglected to ensure the team's attention is distributed evenly.

**Pulling versus Pushing work:**

In a workflow, work can be either pulled or pushed depending on the step and availability of resources. To illustrate this, let's consider an example of a restaurant taking and delivering orders. In some cases, the work is pushed forward, while in others, it is pulled.

For instance, after the wait staff takes an order, they usually push it to a queue, where it waits until they have the opportunity to start processing it. Once the cook becomes available, they pull the order from the queue and begin preparing it. After the food is ready, the cook pushes the order to the wait staff delivery queue. Finally, when the wait staff is ready, they pull the order from the delivery queue and deliver it to the customer:

Diagrama

Descrição gerada automaticamente

In this workflow, we can observe both push and pull mechanisms in action. The overall process follows a pull system since the cook only prepares orders upon receiving them. If there are no customers, the cook doesn't prepare any orders, thereby avoiding wastage. In contrast, a push system would involve preparing the orders ahead of time and pushing them onto the wait staff, even if there are no customers waiting.

Interface gráfica do usuário, Aplicativo, Teams

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

**Kanban reports and metrics:**

Kanban provides various reports and metrics to visualize project status, troubleshoot issues, and plan for improvements. Two important metrics to track are lead time and cycle time. Lead time measures the time it takes for an issue to move from the request stage to the completed stage. Cycle time, on the other hand, measures the actual time spent working on an issue.

Kanban boards can generate reports like cumulative flow diagrams and cycle time control charts. Cumulative flow diagrams show the number of issues in each column over time, highlighting any bottlenecks or areas of concern. Cycle time control charts depict the average cycle time for issues over a specific period, allowing teams to monitor their performance and identify opportunities for optimization.

Gráfico

Descrição gerada automaticamente

Linha do tempo

Descrição gerada automaticamente com confiança média

Jira, a popular project management tool, provides automatic real-time reporting to support Kanban efforts. These reports help teams assess their progress, identify areas for improvement, and make data-driven decisions to enhance their workflow.

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

**SCRUM OVERVIEW**

According to the Scrum Guide, which is written by the creators of scrum. Scrum is a framework for developing, delivering, and sustaining complex projects. It is a relatively simple framework for dealing with complex, unpredictable projects. By framework, we mean that scrum contains basic structures and ideas for completing a project. The scrum guide refers to scrum as a process framework for your project management and work techniques rather than a standalone process or definitive method. So, the basic ideas of Scrum can be customized to suit your specific project. This contrasts with a rigid methodology in which every project is executed the same way. Much of what we discussed in this video is discussed in the Scrum Guide. It is free, relatively short, and very well-written. It's highly recommended that you read it. Scrum is a way of achieving the idea of agility. In the previous videos, we have seen that Kanban is another option. You can think of agile as a mindset and the methods and frameworks such as Kanban, Scrum, and XP as ways of achieving agility.

A key component of scrum and agile in general is continuous learning. Scrum projects start with a vision. This is the initial desired end result of the project. The stronger the vision at the start of the project, the better. The product is then work done and iteration at a time.

After the first iteration, we have some of the features of the product. Even though the product only has a few features, it can be considered usable with respect to those features. That product should have value and potentially can be given to the customer. After the first iteration, notice that our vision has slightly changed. This is because we have learned from implementing the first product features and adapted our vision accordingly.

After our second iteration, the product contains the work of the first iteration as well as the work of the second iteration. It now has more features that the customer will value. Because of continuous feedback, we may have improved some of the features from the first iteration. Notice that our vision has again been adapted.

After the third iteration, our product is closer to our vision, but our vision has again changed slightly.

You can see that we are building the project incrementally because we are building it piece by piece. We are building it iteratively because the product gets improved as we are learning along the way. That is why Scrum is both an incremental and iterative approach to building projects. This process of building and improving parts of the product continues indefinitely, allowing the product to stay relevant in a changing marketplace.

If the product that is being built has a physical aspect, such as a rocket, the result of an iteration may be a prototype rather than a part of a completed product. At the end of every iteration, a product called an increment is ready. Here we see that three iterations have created three increments of the product. An increment is a usable product that may be given to the customer.

The organization always has the option to release the increment. Each increment must meet the organization's agreed upon definition of done. This includes quality and security standards, as well as other organizational requirements, such as requiring documentation for each feature. Each increment contains the work of the current iteration as well as the work of all prior iterations. In other words, an increment is the complete state of the product after an iteration.

In scrum, iterations are called sprints. A sprint is a time-boxed to period used to work on an increment. The time period of a sprint is fixed. In general, you do not shorten or lengthen the duration of the current sprint. Sprints usually have a duration of 1-4 weeks, with two-week sprints being typical. Shorter sprints create an opportunity for more adaptation. Longer sprints allow for more work to be done in a single increment. It's up to each team to decide on the appropriate sprint length.

There are three main parts of the scrum framework:

Artifacts are tools that allow for transparency of the project. They allow anyone with access to them to see the current state of the project.

The artifacts that we will talk about are the product backlog, the sprint backlog, the sprint goal, the sprint board and the sprint reports.

The second part of the Scrum framework are the roles related to scrum. We will discuss the roles of product owner, scrum master, development team members and stakeholders.

The third main part of the Scrum framework are the events related to scrum. These are also called ceremonies or meetings. The sprint guide considers the sprint as a container event for other events. In We will discuss the sprint planning meeting, daily standups, the sprint review, and sprint retrospective.

**Scrum artifacts:**

The main purpose of the artifacts is to provide project transparency. Anyone with proper access can use the artifacts to see the current state of the project, including the project's history and future plans. This enables the team to have a shared understanding of the project so that everybody is on the same page. The scrum artifacts are used to enable inspection and adaptation both inside and outside of scrum meetings.

**Product Backlog** is an ordered, ever-changing to do list for the project. It contains issues that are not yet part of any sprint. The scrum guide refers to issues as items. You might also hear them referred to as stories. Constant feedback means that the product backlog is always changing. The product backlog can include issues that represent features, improvements, bug fixes or any other type of issue that you would like. The product backlog is ordered. Issues near the top of the backlog are the closest to being worked on, so they usually have more details than the lower items.

Modifying the product backlog is called product backlog refinement. You might also hear this referred to as backlog grooming. According to the scrum guide, each scrum team decides how to do refinement, but it should consume no more than 10 percent of the development team's time.

In Jira when you are ready to plan a sprint, you navigate to the product backlog and click the "Create Sprint" button. After clicking the Create Sprint button, Jira creates an empty sprint. You can see here that Jira named the sprint using the project key, which is PRJ in this example, and the sprint member which is one in our case. You can see that Jira invites you to drag issues from your product backlog into the sprint.

The list of issues to be completed during the sprint is called the sprint backlog. The sprint backlog includes a plan on how to accomplish the work of the issues. In Jira, this means that before starting the actual sprint, more details are added to the issues in the sprint backlog. Those details describe how the work of the issues will be done.

As part of planning for sprints, it is common to estimate how much work an issue will take. **Story points** are the most common estimation statistic. In Jira you can use story points, hours, issue count or create your own estimation statistic. Story points are a relative measure of the amount of work required to complete an issue. For example, an issue that is assigned two story points is assumed to take about twice as long to complete as an issue that is assigned one story point.

In Jira, there's a field on each issue named story points. In the sprint backlog, you can see that the story points are shown in the gray boxes along with a total estimate of three points for the sprint backlog. Story points are used to help the team decide how many issues can be completed in a sprint.

When you want to start a sprint, you click the "Start Sprint" button associated with the sprint. The start sprint screen appears.

Jira starts by reminding you that you have added issues to the sprint backlog. You can modify the sprint name, specify the sprint's duration, and specify the start date for this sprint.

You can see that you can have the sprints started at a later date, so you don't have to actually click the start sprint button on the first morning of the sprint. You could also set up multiple sprints at one time.

The start sprint window also contains a place to enter what is called the sprint goal. The sprint goal represents the objective of the sprint’s increments. The sprint goal is reached by completing the issues in the sprint backlog.

A scrum role is that the sprint goal does not change during the sprint. The sprint is considered a success if the sprint goal is reached.

There are two major reasons to have a **Sprint Goal**. The first is that it provides a coherence to the product increment. This means that the features are related so that the product increment is valuable rather than building a collection of unrelated features. This also results in the Scrumteam working together to achieve the Sprint Goal.

The second reason is that it enables flexibility with the sprint backlog. Projects are complex and even though the sprint duration is relatively short, the team cannot predict the future and will learn and adjust during the sprint. There must be flexibility somewhere.

The sprint goal remains fixed during the sprint, but the issues that achieve the sprint goal can be modified as long as quality is not decreased. This means that there's flexibility in the makeup of the sprint backlog as the sprint is worked on. The sprint goal provides guidance for decisions as the team makes the adjustments.

A sprint has a **sprint board**. Notice that it only contains the issues in the sprint backlog. Issues in the product backlog or issues that are assigned to other sprints are not shown on the sprint board. Even in sprint projects, boards are often called Kanban boards, so don't be confused if you hear that term related to a sprint.

The **reports** in agile are tool to visualize the work, promote transparency, help with troubleshooting and continuous improvement, and help with planning and estimating.

In Jira, you access reports using the Reports tab in the sidebar. You can see that Jira automatically provides many reports related to your project.

Scrum has some common reports related to sprints. We will discuss the Burndown chart, spring report, and velocity chart.

A **Burndown chart** shows the progress that the team makes during a sprint.

The sprint backlog starts with a certain number of issues, each with an associated number of story points or other estimation statistic. The total number of starting points is shown on the left of the chart. This is the number of story points that the development team estimated that it would complete in the sprint. In our case, this sprint has three story points. The gray guideline shown is used to show the number of story points that should remain on a given day, assuming a linear Burndown of story points. On the last day of the sprint, the guideline reaches zero story points. This means that the work of the sprint should be finished on the last day. Notice that the non-working days are shown in the chart and the guidelines assume no progress will be made on those days.

As the sprint is underway, Jira will automatically update the Burndown chart as the status of the issues are updated by the team. The red line shows the actual number of remaining story points over time. You can see that about two days into the sprint, one story point was completed. On the last day of the sprint the remaining two-story points were completed.

Consulting this chart is an easy way to see if the team is on track for the current sprint. If the red line is below the gray line, your team is on track to complete all of the story points and reach the sprint goal. If not, the team may need to make some adjustments to reach the sprint goal.

The **sprint report** contains a nice summary of the sprint. It shows the Burndown chart as well as the current status of all of the issues in the sprint. This is an easy way to see how the sprint is progressing.

**Velocity** represents the rate at which the team accomplishes work. Usually, it is the number of story points completed per sprint.

Some teams use an estimation statistics other than story points, so in that case, velocity measures some other units completed per sprint.

You can see the team's velocity of a single sprint by looking at the Burndown chart. In this sprint, the team completed three story points, so its velocity is three. The velocity chart shows the estimated and the actual velocity of the team over time.

Scrum is an agile framework. An increment is a potentially shippable portion of the project that meets the "definition of done". A sprint is a time boxed period in which the increment is created. Scrum artifacts provide project transparency, enable shared understanding, and enable inspection and adaptation. Artifacts include the product backlog, the sprint backlog, the sprint goal, sprint boards and reports. Velocity is the rate at which the team accomplishes work, usually in story points per sprint.

**Scrum roles:**

A scrum team is made up of three roles: **product owner, scrum master, and development team**. The scrum team is made up of cross-functional team members allowing it to complete stories within the team. It is flexible and adaptable with members willing to help out where needed and continuously learning new things.

It is self-organizing: The team is responsible for deciding how to organize and do it's work.

Stakeholders are not members of the scrum team but are interested in the success of the project. There are internal stakeholders such as company managers, executives or other scrum teams that rely on the work of the scrum team. There are also stakeholders that are external to the organization such as customers, partners, and investors.

The **product owner** is the member of the scrum team who is responsible for communicating the product vision. The stakeholders and the scrum team need to have an understanding of the product vision in order to work effectively. The product owner is also responsible for maximizing the value of each increment. Each feature that the development team works on should be of high value.

The product owner is responsible for the product backlog. Others may help with the product backlog, but the product owner is responsible for it. Stakeholders primarily interact with the product owner. The product owner represents the stakeholders when they are not part of the discussion such as during scrum team meetings. The product owner is accountable to the stakeholders for the success of the project.

The **scrum master** is the member of the scrum team who is primarily responsible for promoting and supporting scrum, for scrum team members as well as stakeholders. It is up to the scrum master to ensure that everyone understands how and why things are done a certain way. The scrum master is also responsible for the day-to-day effectiveness of the scrum team. This includes ensuring that the team is reaching its goals and continuously improving.

The scrum master is responsible for protecting the focus of the team. This may mean helping to remove bottlenecks or ensuring that those outsides are interacting with the team in helpful ways. In general, scrum masters do what they need to do to allow team members to focus on their work.

The scrum master is also responsible for increasing the transparency of the project. There should be no surprises about the current status of the project.

The scrum master's tasks vary by project and from day-to-day. Typical tasks include coaching the scrum team and stakeholders on scrum and agile, helping to remove blocking issues, facilitating scrum events, configuring scrum artifacts, and monitoring sprint progress. This distinction of the product owner and scrum master roles should now be pretty clear

The **product owner** is responsible for the value of the product. The **scrum master** is primarily responsible for the effectiveness of the team. You can think of this both as a divide and conquer approach because the work of the roles is so different. Also, as a checks and balances approach because combining the roles could put too much weight on one of the responsibilities.

The separate roles lead to greater team success and sustainability. The development team is a cross-functional, adaptive team that does the work of the project.

Responsibilities of the development team include:

* Estimating issues – this is usually done using story points, but other estimation methods can be used;
* Deciding on how much work can be done in the sprint – only the development team can decide how much work to take out. It is assumed that the development team is in the best position to forecast how much work the issues take;
* Deciding how to organize to do the work of the team – this is because the team is an empowered, self-organizing team;
* Creating the increment of each sprint;
* The development team are the only members allowed to modify the sprint backlog during the sprint – This is sometimes necessary as the team learns and adjusts as it is building.

The Scrum Guide recommends having from three to nine members of the development team. Fewer than three members decreases the productivity and quality created by a cross-functional group of people working together. More than nine members tends to increase the amount of coordination required.

Jeff Bezos at Amazon refers to two pizza teams, meaning two pizzas should be able to feed the entire team.

To scale scrum to more than nine people, it is usually better to create multiple teams.

**Scrum events:**

A sprint contains four types of **events**. You may also hear events referred to as **ceremonies** or simply **meetings**. The events are the **sprint planning meeting, the daily standups, the sprint review, and the sprint retrospective**.

Scrum events occur at regular intervals and minimize the need for other meetings. These events are designed to maximize the opportunities for feedback, and continuous learning, and are a key part of achieving agility. In between these meetings is where the work of the issues of the sprint is completed.

All the **scrum meetings** have some common characteristics:

* The meetings have a fixed maximum time limit and no minimum time limit;
* Meetings can never go over their allotted time, but they can be ended early if the purpose of the meeting is achieved;
* The meetings are primarily to plan, inspect, and adapt.

In agile projects, the planning is distributed throughout the project rather than the mostly upfront planning of waterfall projects. These meetings are an important part of that distributed planning. The meetings are used to inspect the project and adapt with the team is doing based on that inspection. This is a key part of increasing transparency and continuous improvement.

The meetings are primarily about the team collaborating, not about updating status. The work is visualized, so you don't need a meeting to see the status. It is the responsibility of every team member to ensure that meetings have value and to help modify them to increase their value if necessary. If the discussion moves in a direction that becomes a value to only a portion of the participants, it should be moved to a later time outside of the meeting.

The **sprint planning meeting** is held at the start of a sprint. The entire scrum team attends the meeting. The duration of the meeting is typically four hours for a two-week sprint. If your sprints are four weeks long, you can expect this meeting to be twice as long. The purpose of the meeting is to plan the work of the sprint. The output of the meeting is an agreed upon sprint goal and a sprint backlog.

Before the meeting, the product owner usually has a proposed sprint goal and a minimum set of issues that accomplish the goal. These preliminary items often come from the product backlog that has been updated during the previous sprints, sprint review, and sprint retrospective meetings.

During the meeting, the team usually discusses the sprint goal, modifies the sprint backlog, places story point estimates on issues, and adds details to the issues to better describe the specific work to be done.

The **development team** is responsible for estimating the story points for the work and deciding on how much work can be done during the sprint. They need to do enough planning to have an accurate forecast for the amount of work that they will agree to.

The **development team** also creates subtasks for the first few days of the sprint. The subtasks are often a day or less of work. This is an example of an empowered team rather than a command and control-based team.

The purpose of the meeting has been met when there is an agreed upon sprint goal and sprint backlog. The sprint backlog contains the issues that the development team has agreed to complete during the sprint, as well as a plan for completing that work.

The **daily standup meeting** is also called the **daily scrum**. It is a planning meeting that occurs every day and the participants usually stand as a reminder that it's a short meeting.

The meeting usually takes place in the same location at the same time every day. The development team are the primary attendees for the daily standup. Others may attend the meetings but are usually asked to listen only.

The meeting usually last 15 minutes or less. The purpose of the meeting is to inspect recent progress toward the sprint goal, plan the day's work, and identify any impediments to the team.

The team usually makes plans to resolve the impediments, but the discussion often moves to after the meeting.

The output of the meeting is the plan for the day.

It's important that the daily standup is a collaborative meeting and not simply a status update. The team collaborates to make the best decisions for the day given the latest information. They usually decide on who will work on specific issues, to plan slightly changes after every daily standup. This is continuous improvement.

The **sprint review meeting** occurs near the end of the sprint. It is an informal meeting that includes the scrum team and interested stakeholders.

It is typically a two-hour meeting for two-week sprints.

The purpose of the meeting is to inspect the increment that was just created in the sprint and to collaboratively update the product backlog. This is a meeting with a lot of feedback on the project and includes a brainstorming session to help decide what to do next.

The output of the meeting is a first pass at the next sprint's backlog. By the time the sprint planning meeting happens for the following sprint, the team already has a good idea of what they will be working on.

The **sprint retrospective** is the last event of the sprint. The scrum team attends the retrospective.

The meeting typically takes 90 minutes for a two week sprint.

The purpose of the meeting is for the team to inspect itself including its processes, tools, and team interaction.

The retrospective is a positive meeting containing constructive feedback. Everyone should always remember that they are part of a team.

The scrum master usually helps make sure that this is a positive meeting. The team usually discusses what they should keep doing, what they should stop doing, and what they should start doing. The meeting is about continuously improving the team.

The output of the meeting is to add one or more improvement related issues to the next sprint's backlog. It's important that the team spent some of it's time on these issues, rather than exclusively building the product.

It's important to always make sure that the meetings have high value, and the team should focus on continuously improving their meetings.

Notice that the next sprint goal and sprint backlog start to form in the sprint review.

The retrospective usually adds one or more issues. This forms the starting sprint backlog for the sprint planning meeting.

* Scrum roles include the product owner, scrum master, development team members, and stakeholders;
* Scrum meetings include the sprint planning meeting, daily standups, the sprint review, and the sprint retrospective.