

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF ENGINEERING AND TECHNOLOGY



AGILE BASED SOFTWARE ENGINEERING
(CSP 644)
LAB FILE
for
2nd SEMESTER

Submitted to:

Dr. Anuj Kumar (SET Assistant Professor)
Dept. of Computer Science and Engineering

Submitted by:

Bhaumik Tyagi (M. Tech CSE)
2021327470

EXPERIMENT 1:

Q.1 Design a workflow process to Streamline and automates the steps required to ensure custom statuses are completed.

Overall Description:

What is streamlining?

Streamlining is the process used to simplify or eliminate unnecessary work-related tasks to improve the efficiency of processes in businesses or organizations. Streamlining processes require the usage of modernizing techniques, technology, and other possible approaches to complete.

What are processes and workflows?

Processes and workflows are similar but they are not the same. Here are the correct definitive terms for processes and workflows:

- **Process:** A set of repeatable activities that need to be continued to complete a specific goal that an organization has set.
- **Workflow:** A series of repeatable activities that need to be continued to complete a specific task.

Steps to design a workflow process to Streamline and automate to ensure custom statuses are completed.

Streamlining processes and workflows may take some time and is best completed in small steps that contribute to your organization's improved efficiency goals. Businesses may streamline processes by reviewing the details of how they manage their individual challenges. Your organization needs to decide which areas to streamline. Here are 6 steps you may try to streamline processes and workflows to improve efficiency:

1.Assess existing processes and workflows-

When you assess your current processes and workflows it will be easier to get an overall idea of the exact way things are done before you can notice what areas could be greatly improved by streamlining. For instance, it may be helpful to list your processes and workflow in the most simplistic terms, write down what you believe the benefits of each process, and list each person involved in the process.

2. Rank processes-

It is likely surprising to find out exactly how many processes and workflows are actually carried out to complete goals or tasks. Many processes are interrelated and may not be as important. Write down a list of processes from most important to least important. This will help you to decide which process to streamline first, second, third, and so on.

3. Analyze outcomes-

Once you have assessed the processes and workflows, you may analyze the outcomes or results of those processes and workflows. This gives you a perspective on processes that are tedious, unnecessary, and not budget-friendly. For example, you may note in the assessment that there are five bins of paper shredded and taken to the trash each day. When you analyze the outcome, you should calculate the cost of the toner and paper, the time employees spend printing and shredding, and compare what a new software could do to improve a certain process or workflow in your organization or department.

4. Ask for feedback-

Colleagues, coworkers, or employees may have valuable feedback about the workflow chain or process. Many of these individuals likely have their own ideas on how processes are working and how they can be improved to save time and complete tasks and goals more efficiently. There are often small details that can be improved to streamline processes and they can be easily missed unless you ask for feedback from those with whom you work.

For instance, you may try holding a meeting to discuss areas of improvement or you may generate a survey based on various processes and workflows and have your employees submit answers anonymously to avoid bias.

5. Streamline and automate processes-

There are now many modern workflow software solutions that help with streamlining business processes and workflows. They are often referred to as business process management software. Each one has its own features and capabilities that will benefit different career fields and organizations. However, most BPMs are designed to have similar benefits such as making it easy to collect data in a central location, automating steps in business workflow, and generating better visibility with graphs, charts, and reports.

Try researching software that may work for your organization and check to see what your competitors are using to streamline their processes.

6. Adjust and refine-

There is no perfect process or workflow and they will likely require continuous adjustments or refinements. You may want to change the streamlining processes based on your results. It may also take time to train employees properly on the new processes and not every employee may get the process down correctly the first time. When you refine processes and workflows, it requires you to be patient and ask for additional feedback from clients and employees.

EXPERIMENT 2:

Q.2 Understand agile development practices.

Overall Description:

Best Practices for Agile Teams-

Using an iterative framework, the Agile methodology relies upon the interaction of self-organizing teams of people who have the cross-functional skillsets required to develop tested, working software. The most commonly used programming practices – test-driven development, code refactoring, continuous integration, simple code design, pair programming, a common codebase and a single coding standard – contribute to the quality, flexibility and sustainability of the software.

The people skills required for efficient and effective interaction on Agile teams are:

1. **Collaborate with the customer:** An Agile team is in near-constant communication with the customer, clarifying expectations, collaborating on fixes, and communicating options not previously considered.
2. **Work together daily:** According to the Agile Alliance, a common pitfall among Agile teams is to equate a group of people who work together with a “team.” Teams, and teamwork, contribute to successful projects when they collaborate as a cohesive unit. Organizational science researchers identified six components of Teamwork Quality:
 - Communication
 - Coordination
 - Balance of team member contributions
 - Mutual support
 - Effort
 - Cohesion
3. **Build projects around motivated individuals:** It takes motivation to push through an intense development cycle and get the work done right. Agile teams are passionate about their work, focused on the team goal, and supportive of each other. When there is trust and respect among peers, Agile teams establish a rhythm to their work that is fast-paced and predictable. They are willing to:
 - Take on the required roles
 - Form collaborative relationships
 - Adopt what the team deems to be the most efficient processes
 - Work reliably without management oversight
4. **Convey information face-to-face:** Whether working through a knotty problem with a teammate or reporting on the day’s accomplishments at a daily meeting, Agile team members prefer face-to-face communication. This brief, face-to-face encounter demands team members be present and forthcoming.
5. **Form self-organizing teams:** Self-organizing teams choose how they will execute the work, and who will do what. They divide the work into increments that can be completed within each iteration, and into tasks that can be completed each day.

EXPERIMENT 3:

Q.3 Understand the background and driving forces for taking an Agile Approach to Software Development.

Overall Description:

The meaning of Agile is swift or versatile. " **Agile process model**" refers to a software development approach based on iterative development. Agile methods break tasks into smaller iterations or parts that do not directly involve long-term planning. Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks. Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.

Agile Development works on four values and twelve principles.

1. Individual interactions are more important than processes and tools.
2. A focus on working software rather than thorough documentation.
3. Collaboration instead of contract negotiations.
4. Responding to change over following a plan.

The 12 principles of Agile-

The Agile Manifesto also outlined 12 core principles for the development process. They are:

1. Satisfy customers through early and continuous delivery of valuable work.
2. Break big work down into smaller tasks that can be completed quickly.
3. Recognize that the best work emerges from self-organized teams.
4. Provide motivated individuals with the environment and support they need and trust them to get the job done.
5. Create processes that promote sustainable efforts.
6. Maintain a constant pace for completed work.
7. Welcome changing requirements, even late in a project.
8. Assemble the project team and business owners on a daily basis throughout the project.
9. Have the team reflect at regular intervals on how to become more effective, then tune and adjust behavior accordingly.
10. Measure progress by the amount of completed work.
11. Continually seek excellence.
12. Harness change for a competitive advantage.

EXPERIMENT 4:

Q.4 Create sidebars and use the drag-and-drop option.

Overall Description:

ClickUp is a cloud-based collaboration and project management tool suitable for businesses of all sizes and industries. Features include communication and collaboration tools, task assignments and statuses, alerts and a task toolbar.

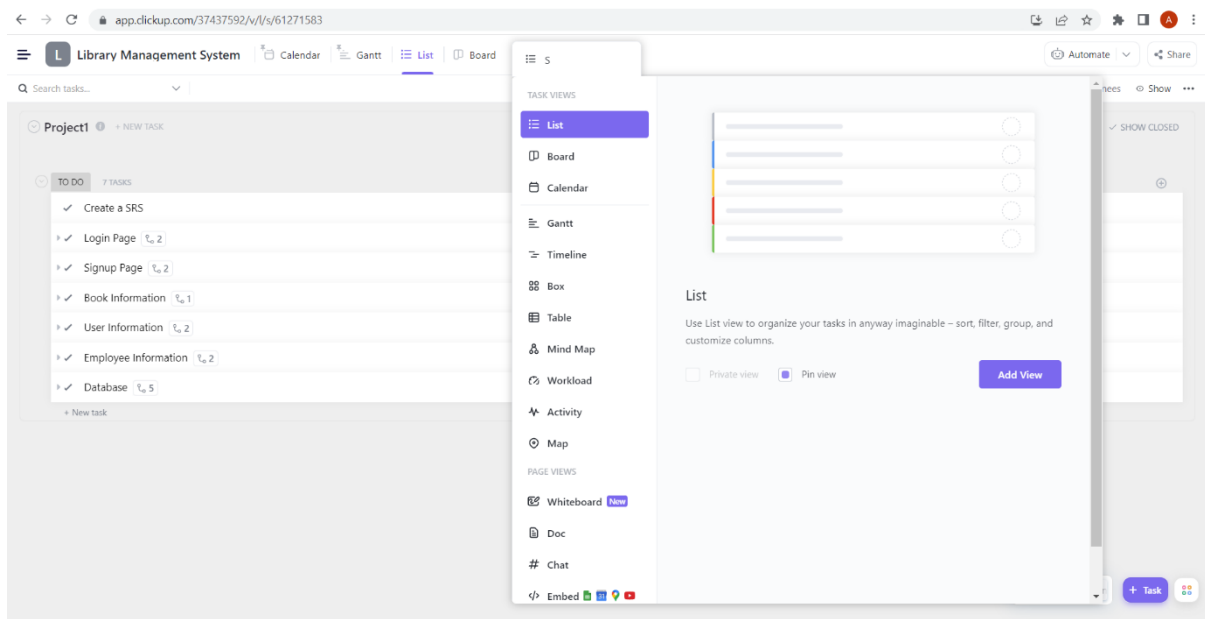
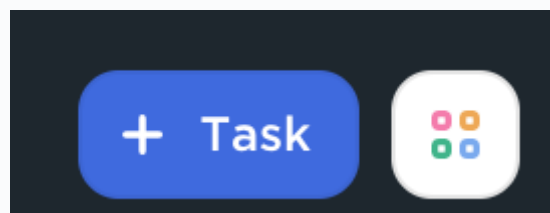


Fig: ClickUp Tool drag and drop option

The above figure shows of ClickUp project from where we can easily manage our project and use multiple features as per the requirement of software development.

With ClickUp, creating Tasks has never been easier.

Just click on the pretty purple button in the lower right corner of your screen.



Here's what you can do when creating tasks in ClickUp:



- Add Custom Task Statuses, Subtasks, and Checklists
- Organize using Tags and Priorities
- Assign tasks, Due Dates, and more

Want to save time? Rely on Task Templates.

Instead of wasting time creating things from scratch each time, just reuse a format you used before!

Save as Template ×

Template name


 Select Template or Create new... 

TEMPLATES

Create new Template

Big Plans

☐ Admins

☐ Select people: 

☐ All Members

EXPERIMENT 5:

Q.5 Figure out the schedule time, manage workforce capacity and organize important events.

1. Schedule Time:

- Divide the project into workable tasks and define the sprints' length: An Agile project starts with a list of features to be implemented and several iterations to implement those features. A project manager should identify these tasks prioritize them with the help of their team and input them into the schedule. Each feature will have a timeframe to be mapped to an iteration. Each feature and sprint will also require a status e.g. not started, in progress or completed.
- Add the tasks that should be completed as part of the sprint: Agile management software can help project managers and their teams view the tasks to be completed and mark them as 'done' when they're ready. Applications like Jira, Trello, Monday, and Zoho Sprints are just a few examples of Agile management software.
- Define the estimated time for each issue in the sprint: Each section or iteration is reviewed by the project team, which should include some of the project's various stakeholders. Insights gained from the feedback are used to determine the next steps.

Monday	Tuesday	Wednesday	Thursday	Friday
16	17	18 Sprint Planning (Sprint 1)	19 Daily standup	20 Daily standup
23 Daily standup	24 Daily standup	25 Daily standup Backlog Refinement	26 Daily standup	27 Daily standup
30 Daily standup	31 Daily standup	1 Daily standup Backlog Refinement	2 Daily standup	3 Daily standup
6 Daily standup	7 Sprint Review (Sprint 1) Sprint Retrospective (Sprint 1)	8 Sprint Planning (Sprint 2)	9 Daily standup	10 Daily standup

Fig: Schedule time

2. Manage Workforce Capacity:

- An agile workforce is simple in its definition. It is a workforce strategy that makes use of both permanent employees and non-employee (contingent) workers to improve the flexibility, scalability and agility of your business.

Agile projects typically break the work of implementing and validating requirements into a set of tasks (the task breakdown) associated with each requirement specification (the User Story). Most Agile projects estimate tasks in units of person-hours, so our goal is to determine how many person hours of work a Team can perform in a particular period of time, such as a Sprint or Release.

- Building an agile workforce will ensure your company has the capabilities to scale up or down depending on current demand, beat the skills gaps by accessing talented workers at good rates as-and-when they are needed, and make better staffing decisions depending on your organization's current workforce requirements.
- Multiply the number of workdays in the period by eight (hours per day) to get the total number of "Work Hours" hours in the period.
- Subtract the total time allocated for whole-team meetings. This result is the "Net Work Hours," and is smaller than the total "Work Hours."
- Get the availability and time off for each person. For each person, subtract time off from Net Work Hours, and multiply the result by his availability to get his individual capacity.
- Add up the individual capacities to get the Team capacity in person hours, and divide by eight to get the capacity in person-days.
- Divide the Team capacity in hours by the Work Hours to get the Net Team Resources, which is the effective number of full-time people on the Team.

Team Member			% Avail	Hrs Off	Hours
Jason Archer			25%		8.00
Don Bass			38%		12.16
Cheryl Lu			75%		24.00
Harry Smith			75%		24.00
Terry Pin			75%		24.00
Jack Bean			50%	16	8.00
Roy Jones			75%		24.00
Person-Days Available:	15.5	Net Team Resources:	3.10	Team Hours:	124.16

3. Organize Important Events:

- Agile Events, (formerly Agile Ceremonies), are several types of meetings prescribed within Agile frameworks. The most common Agile framework with such events is the Scrum. It is recommended for the Scrum framework to hold six key events during every Sprint to ensure efficient work from the team.

The six key events are:

- a) Daily Standup/Scrum :Usually a daily meeting, lasting no more than 15 minutes. Typical for Scrum and Kanban Agile framework.Team members describe what they worked in the previous day, and what are they intend to work on today. It should not be detailed.Where further discussion is merited, a "Parking Lot" discussion can be held afterwards with those concerned.
- b) Sprint Planning:A sprint is a term specific for a Scrum framework, and usually lasts two weeks (Some teams may opt for one or three week sprints).Other types of agile frameworks use “iteration” to define a time-boxed period of development.The team uses this meeting to decide what to do during the sprint. The team has to estimate how much work they can complete from the product backlog and the effort involved, also making a small allocation of effort for any defects that may emerge.
- c) Iteration/Sprint Review: At the end of a sprint or milestone, the team reviews their accomplishments against the planned sprint work. The team will often use this meeting to demonstrate the results of their work, and get feedback from stakeholders.
- d) Retrospective: At the end of an iteration, the team discusses what worked well, and what could be improved. Scrum and Kanban frameworks use this meeting for creating solutions and developing action plans.
- e) Product Backlog Refinement: In this meeting, the team assesses work in the Product Backlog, adds detail and estimates the effort required and priority for each job. The Backlog is updated and the items are prepared for the upcoming sprint. This meeting can be open ended, with the time spent dictated by the amount of outstanding work to be assessed.
- f) The Sprint: It is the time-boxed period that contains all the work planned during Sprint Planning, and all other work such as resolving urgent defects that occur.

The most common Agile framework with such events is the Scrum. It is recommended for the Scrum framework to hold six key events during every Sprint to ensure efficient work from the team. Usually a daily meeting, lasting no more than 15 minutes.

EXPERIMENT 6:

Q.6 Show the implementation of two-way calendar sync.

Two-way synchronization (or bi-directional sync) is all about mirroring the data between two business tools. It means all newly created records in one app should be instantly (or near-instantly) created in the other one, and all updates of already existing records should be reflected in the records of the other app.

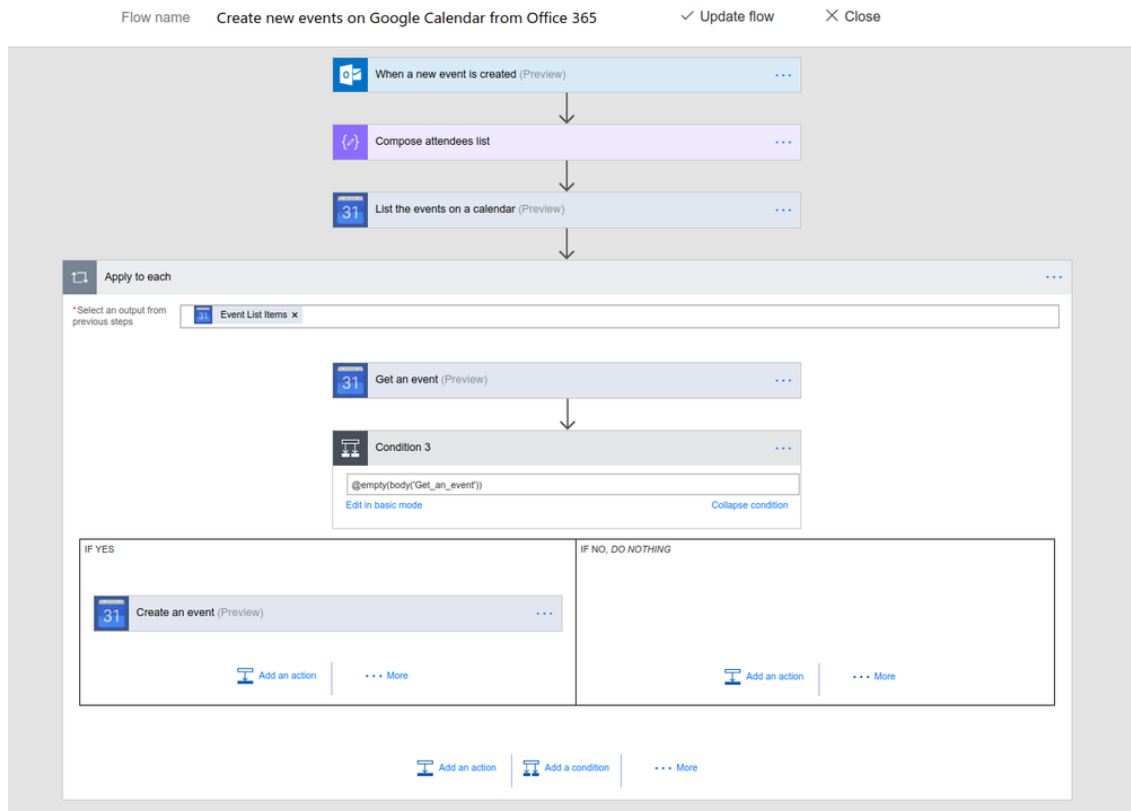


Fig: Two way calendar sync

EXPERIMENT 7:

Q.7. Show the implementation of Gantt charts.

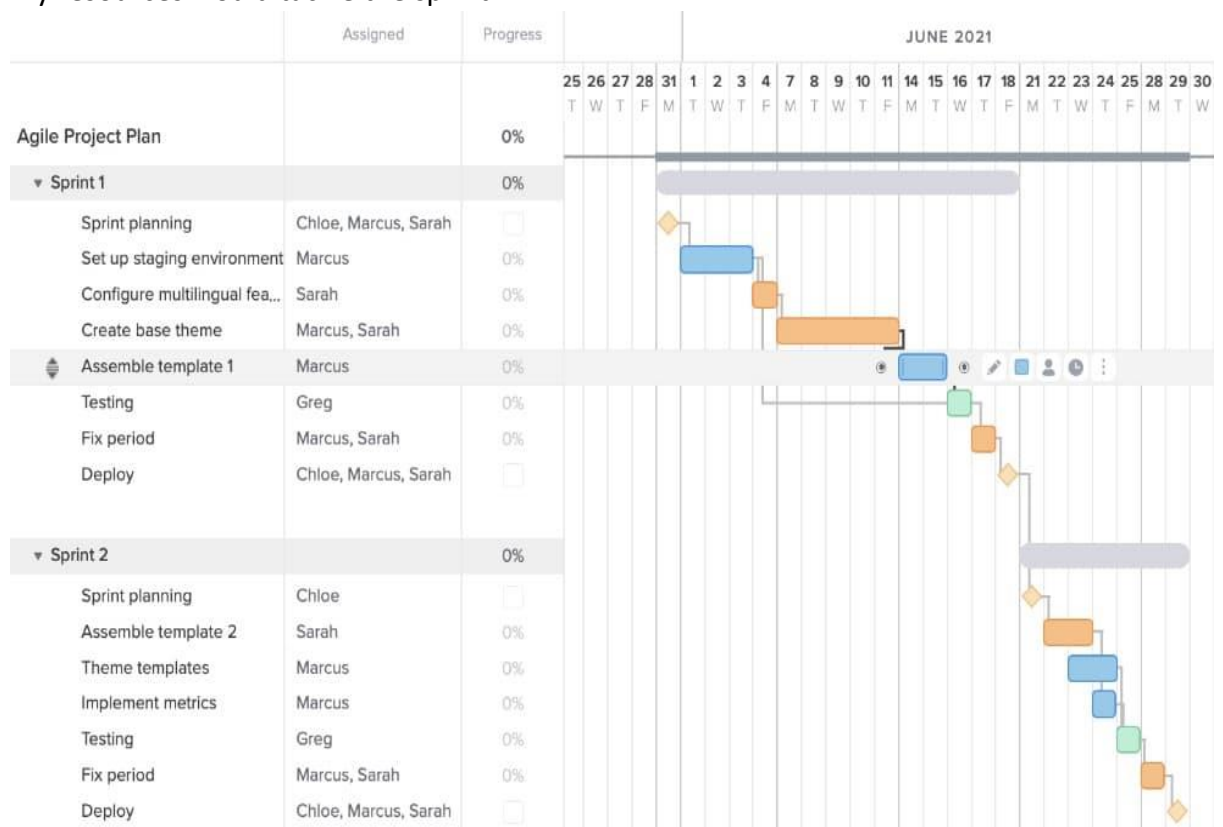
What is an Agile gantt chart?

An Agile gantt chart is a project planning tool that applies a Waterfall model to an Agile project by mapping sprint tasks and dependencies out on a visual timeline. Using a gantt chart for an Agile project makes it easy to track progress, manage workloads, and keep stakeholders up-to-date on the work.

How to use an Agile project framework with a gantt chart:

1. Create task items per feature-

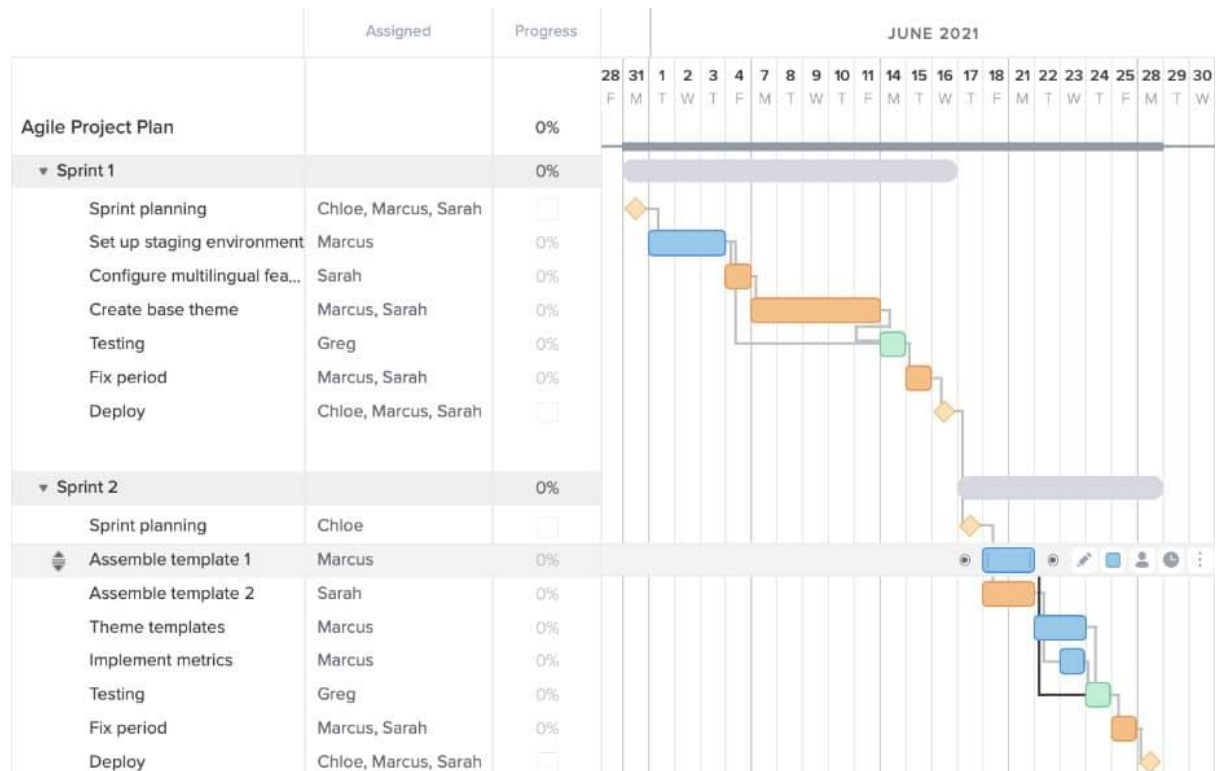
Under each anticipated iteration, I create one task item per feature. The key deviation from a typical gantt chart for a Waterfall-based project is that this chart relies heavily on dependencies. All items in each iteration have a start-to-finish dependency to the testing period for the iteration. Start-to-finish dependencies among tasks are only given if there is a technical dependency that would model how my resources would tackle the sprint.



2. Move and rearrange the dependencies-

Below, you can see that I've moved the task down and rearranged the dependencies. From my daily stand-ups, I've determined that my two resources can work on their assigned tasks simultaneously, but some of the other tasks that are also assigned will have to move back to accommodate this new addition.

So now, the gantt chart for this Agile sprint looks something like this:

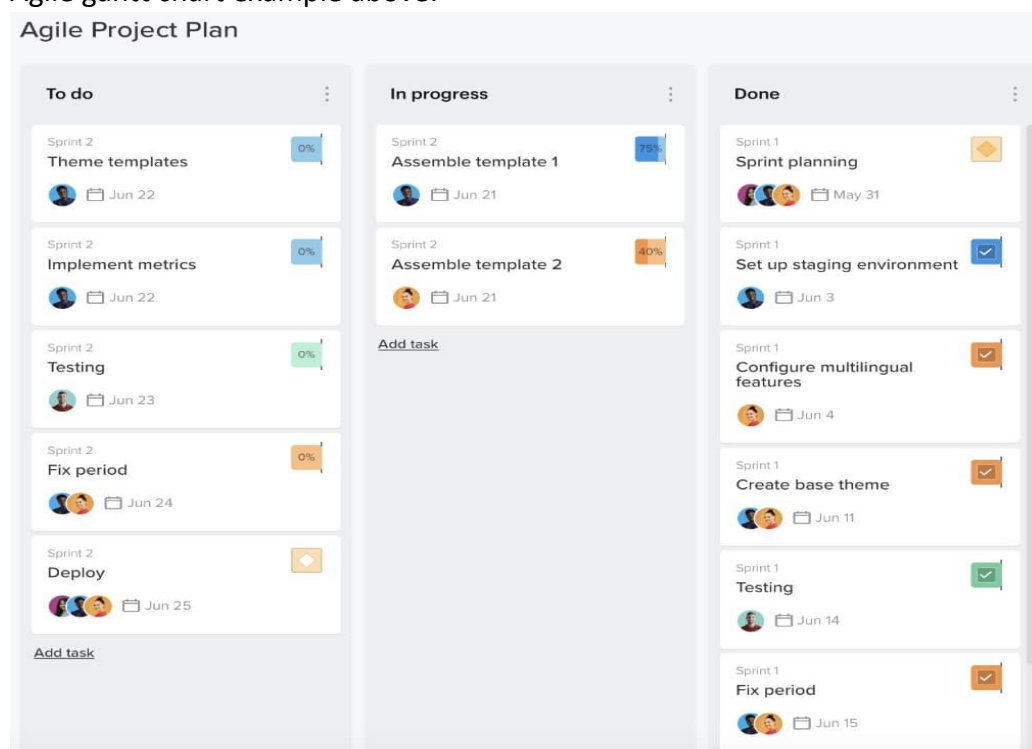


3. Use project boards to tackle daily tasks-

If your team doesn't want to work in a gantt chart, TeamGantt gives you the option to view and manage tasks in a kanban board without having to juggle multiple tools to tackle the project.

In Board view, cards are tied directly to tasks on your gantt chart. Team members can update and move cards across columns on your board as work progresses—and you can rest easier knowing your Agile timeline always reflects the current status of the project.

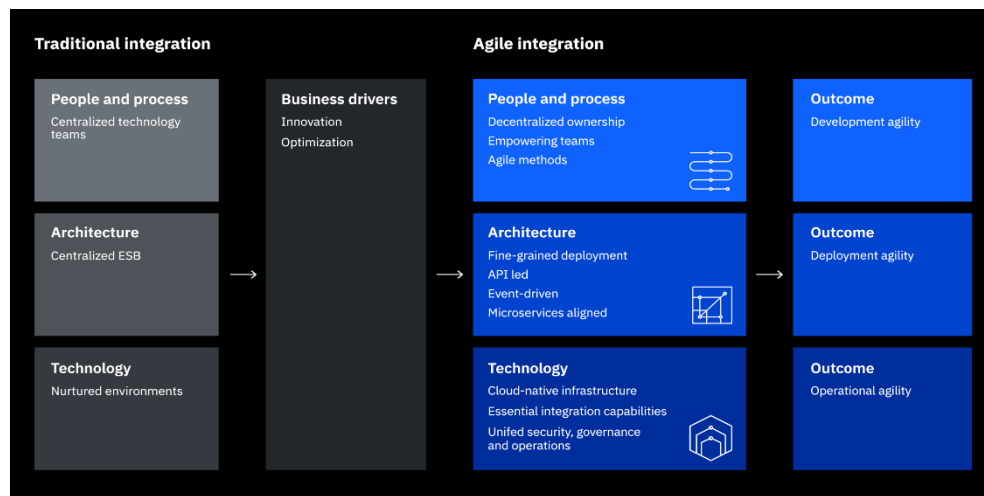
Here's what it would look like if we track our sprint tasks in Board view based on the Agile gantt chart example above:



EXPERIMENT 8:

Q.8. WAP to integrate more than one applications within the system.

Agile integration enables you to break free from heavily centralized integration architectures that cannot support the demand. Instead, integration becomes a critical enabler of innovation. This new approach spans from people and process to architecture and technology by leveraging modern practices and capabilities including APIs, microservices architecture, cloud-native design, DevOps, event-driven architecture and container-based infrastructure.



There are three related, but separate aspects to agile integration:

- **People and process:** Decentralized integration ownership. How should we adjust the organizational structure to better leverage a more autonomous approach and give application teams more control over the creation and exposure of their own integration, and how they are exposed as APIs, messages and events?
- **Architecture:** Fine-grained integration deployment. What might we gain by breaking out the integrations in the siloed ESB into separate deployments that could be maintained and scaled independently? What is the simplest way that these discrete integrations could be made available with consistency and enhanced security across and beyond the enterprise using APIs and events?
- **Technology:** Cloud-native integration infrastructure. How can we best leverage a cloud-native infrastructure such as containers to improve productivity, operational consistency and portability for both applications and integrations across a hybrid and multicloud landscape?

EXPERIMENT 9:

Q.9. To study the Continuous Integration tool.

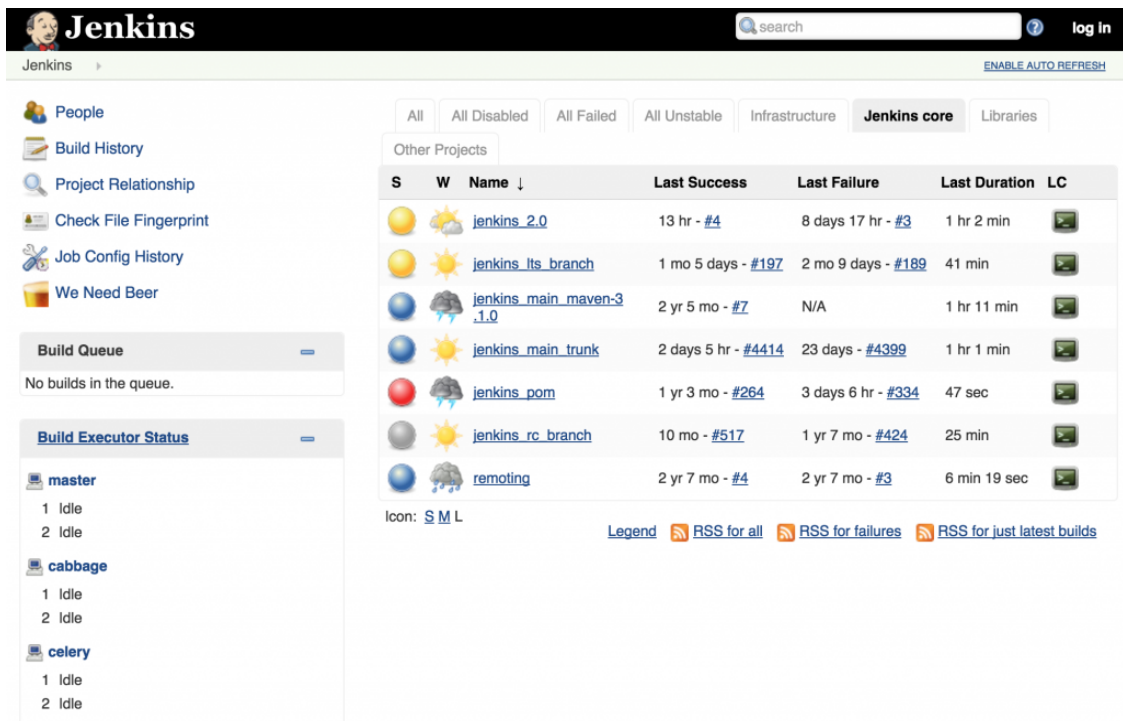
Many Agile development teams turn to CI solutions, as they not only automate code sharing and merging but also report if anything has gone wrong. This makes automated build tools in Agile tools indispensable in preventing a break of a ready build. Also, as these solutions minimize manual testing efforts, they are often considered test automation instruments, because they allow checking the code against a larger codebase.

The major advantage of CI tools is that they make a significant contribution to the software quality by revealing of possible defects earlier for their timely elimination. The users derive the following benefits as well:

- A quick feedback loop
- Improved collaboration within a team
- Reduced risk of code regression
- Version control within the various patch and product releases
- Reduced delivery time
-

Many companies wonder which continuous integration tools for agile software development are the best and why to opt for one instrument instead of another. Indeed, the choice depends heavily on a company's requirements, available budget and technology stack.

Jenkins:-



The screenshot displays the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and a 'log in' link. Below the navigation bar, there are tabs for 'All', 'All Disabled', 'All Failed', 'All Unstable', 'Infrastructure', 'Jenkins core' (selected), and 'Libraries'. The main content area shows a table of build jobs under the 'Jenkins core' tab. The table has columns for 'S' (Status), 'W' (Weather icon), 'Name', 'Last Success', 'Last Failure', 'Last Duration', and 'LC' (Link icon). The jobs listed are: 'jenkins_2.0', 'jenkins_its_branch', 'jenkins_main_maven-3.1.0', 'jenkins_main_trunk', 'jenkins_pom', 'jenkins_rc_branch', and 'remoting'. The left sidebar contains links for 'People', 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Job Config History', and 'We Need Beer'. Below these links are sections for 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing the status of 'master', 'cabbage', and 'celery' executors).

S	W	Name ↓	Last Success	Last Failure	Last Duration	LC
🟡	☀️	jenkins_2.0	13 hr - #4	8 days 17 hr - #3	1 hr 2 min	🔗
🟡	☀️	jenkins_its_branch	1 mo 5 days - #197	2 mo 9 days - #189	41 min	🔗
🟢	☁️	jenkins_main_maven-3.1.0	2 yr 5 mo - #7	N/A	1 hr 11 min	🔗
🟢	☀️	jenkins_main_trunk	2 days 5 hr - #4414	23 days - #4399	1 hr 1 min	🔗
🔴	☁️	jenkins_pom	1 yr 3 mo - #264	3 days 6 hr - #334	47 sec	🔗
🟡	☀️	jenkins_rc_branch	10 mo - #517	1 yr 7 mo - #424	25 min	🔗
🟢	☁️	remoting	2 yr 7 mo - #4	2 yr 7 mo - #3	6 min 19 sec	🔗

Icon: [S](#) [M](#) [L](#)

Legend [RSS for all](#) [RSS for failures](#) [RSS for just latest builds](#)

Jenkins is a cross-platform CI tool that was created after Oracle acquired Sun Microsystems. Now Jenkins is probably the most popular cross-platform solution because of its free-of-charge basis and high configurability and customization. It allows the extension of test features through multiple plugins, both via console commands and GUI interface. Also, Jenkins allows for developing custom plugins.

Besides the superior extensibility, the capability to distribute ready builds on several devices distinguishes this tool. Also, it supports a variety of tools for source code management, such as Git, ClearCase and more. In its performance, Jenkins focuses on continuous software development and testing and on the monitoring of external jobs.

However, on the minus side, Jenkins' user interface is outdated. Also, the absence of performance issues on the server is required for Jenkins' seamless maintenance.

EXPERIMENT 10:

Q.10. To study automated build tool.

Automation in Build:

Continuous Integration (CI): The aim of practicing Continuous Integration is to maintain a build environment that is able to generate “Production Ready” builds. As part of this practice, the following actions will be performed:

- Analyse the code for quality
- Build the code
- Execute unit test cases
- Upload build artifacts
- Post-build and test results
- Alert the team with results

This can be achieved effectively through automation. For further details on CI practice, please refer to the study material on ‘Continuous Integration and Associated Tools’ of this course.

To aid in this practice following support systems are required:

- **Source Version Control tools:** Through “hooking” (proactively notifying other tools when new code changes are made) or “polling” (other tools have to poll the Version Control tool to determine whether new code changes are made), any new code changes should be determined and communicated to other tools by the version control tool. Example – Git, ClearCase, SVN
- **Build tool:** A robust tool/framework that helps in maintaining and code build more easily. Example – Maven, ANT.
- **Static code analysis tool:** As described in the above section, these tools are used to check the code quality. Example – CAST, SONAR.
- **Build repositories:** A robust repository to maintain the builds that can be uploaded/downloaded/searched with ease. Example – Nexus (for maven builds)
- **CI server:** Interacts with all the above components and performs the actual build of the application along with the execution of tests either automatically or just by a click of a button. Example – Jenkins, Hudson.

CircleCI is a free hosted tool for code integrating into both a self-hosted repository or into a shared one. It provides the testing of mobile apps (Android and iOS, cross-platform and native) and Rails apps and has the following technical characteristics:

- Integrates with GitHub web- service
- Supports Java, Python, Ruby/Rails, Haskell, Node.js, PHP and Scala languages
- Provides dependencies from Postgres to Docker
- Utilizes Phantom JS and Cucumber

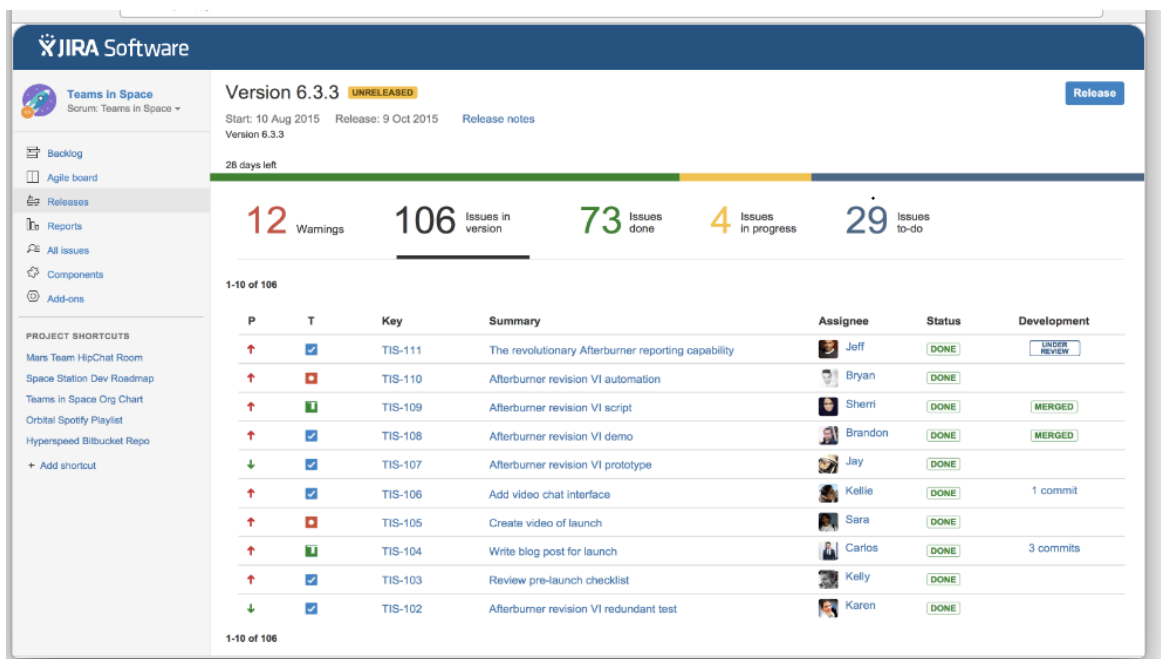
EXPERIMENT 11:

Q.11. Implement Team collaboration & Device agnostic.

Collaboration tools are crucial to enable teams to work together. These include:

- Collaborative dashboards to track work and provide details of sub-tasks as part of the bigger project picture.
- Communication tools that connect team members, including office-based and deskless workers wherever they are in the world. This is even more important with hybrid working and also means teams can work internationally with fewer barriers.
- File sharing tools to give every team member access to the documents they need a
- Integration tools to bring all software and applications together, so there's no wasting time when continually moving between them.
- Progress monitoring tools help project managers plan more effectively and keep teams on schedule by tracking progress in real time.

Your Agile team can stick to using one framework or combine a few as it suits their work process. In fact, in the State of Agile report 2021 the overwhelming majority of responders, namely 66%, reported using Scrum as their preferred framework and 15% use some combination of Scrum and another model.



Device agnosticism, or device-agnostic, is the capacity of a computing component to work with various systems without requiring any particular adaptations.

The term can apply to either hardware or software. In an IT context, agnosticism refers to anything designed to be compatible across the most common systems.

EXPERIMENT 12:

Q.12. Embed links and set permissions to increase productivity and teamwork between the workforces.

a) Embed Links:



b) Set Permissions to increase productivity:

Increased productivity is the key priority for many project managers. There are a few ways that can improve team productivity – you can adopt a new technology or try a new approach and improve the methods and skills your team uses to complete the work. The agile methodology can help achieve that through increased collaboration and being more responsive to customer requests.

Step 1: Show a prototype to the customer as early as possible: The ability to show a working prototype to the customer early is one of the key agile benefits.

Let's take for instance a six-month project:

- With the waterfall approach – it could take three months to see a prototype. Changes are undesirable and hard to implement.
- With the agile approach – it could take less than months to see a prototype. Changes are welcomed and seen as opportunities to make a product better.

The agile focus on face-to-face communication and close cooperation (inside the team and between the team and the customer) makes a great difference to productivity. You can sort all things out during the meeting and start developing the product immediately afterwards. You solve any problems easier and faster. You stay flexible to introduce changes whenever needed. As a result, you increase the overall productivity and create successful products that solve real needs.

Step 2: Create a supportive culture: Success with the agile methodology is 5% due to the tools used and 95% due to the culture. When it comes to the team, the agile approach can be described as collaboration-cultivation-competence.

- Collaboration – “We succeed by working together”: The organisation stands for diversity, teamwork, partners, trust, and interaction.
- Cultivation – “We succeed by growing people”: The organisation understands that its employees are the key, and can greatly contribute to success. Such companies emphasise flexibility, creativity, and growth among their employees.
- Competence – “We succeed by doing our best”: Experts, craftsmanship, professionalism, and efficiency – these are the words to describe agile team members.

They strive to do their best and always look for new techniques or approaches to make their processes even more efficient.

Step 3: Get ready for agile transition in advance: Agile transition is not an easy thing. Without a proper background and preparation, agile can be misunderstood and implemented in the wrong way. Here are a few points to consider:

- Determine your project challenges. Find out your organisation's current problems. Some problems – speed, efficiency and productivity – and think how they can be improved with agile.
- Find a group open to new ideas. Identify people in your organisation who are open to new ideas. These are people who regularly experiment with new software and invest effort in learning and training.
- Ask the group to experiment with agile. Ask the group of 'early adopters' to experiment with agile. But keep in mind that agile is not a silver bullet, so you may have to lower your expectations in the beginning. Try agile with one or two projects for a month and see what you can get. Then do the whole thing again for the next month until you get the phenomenal results.

c) Teamwork between the workforces:

It is nearly impossible to do a project without effective teamwork. The term "teamwork" means the process of acting collaboratively with a group of people in order to accomplish a shared objective. Teamwork creates and shapes the collaborative effort the team makes to move the project to success.

Following the Agile principles, there are three major areas that identify successful collaboration and teamwork, as follows:

- Project developers and customers must daily communicate and collaborate with each other throughout the entire development process.
- A better motivated team member will work better in the collaborative environment.
- Face-to-face conversation is the best way to convey information within a team.