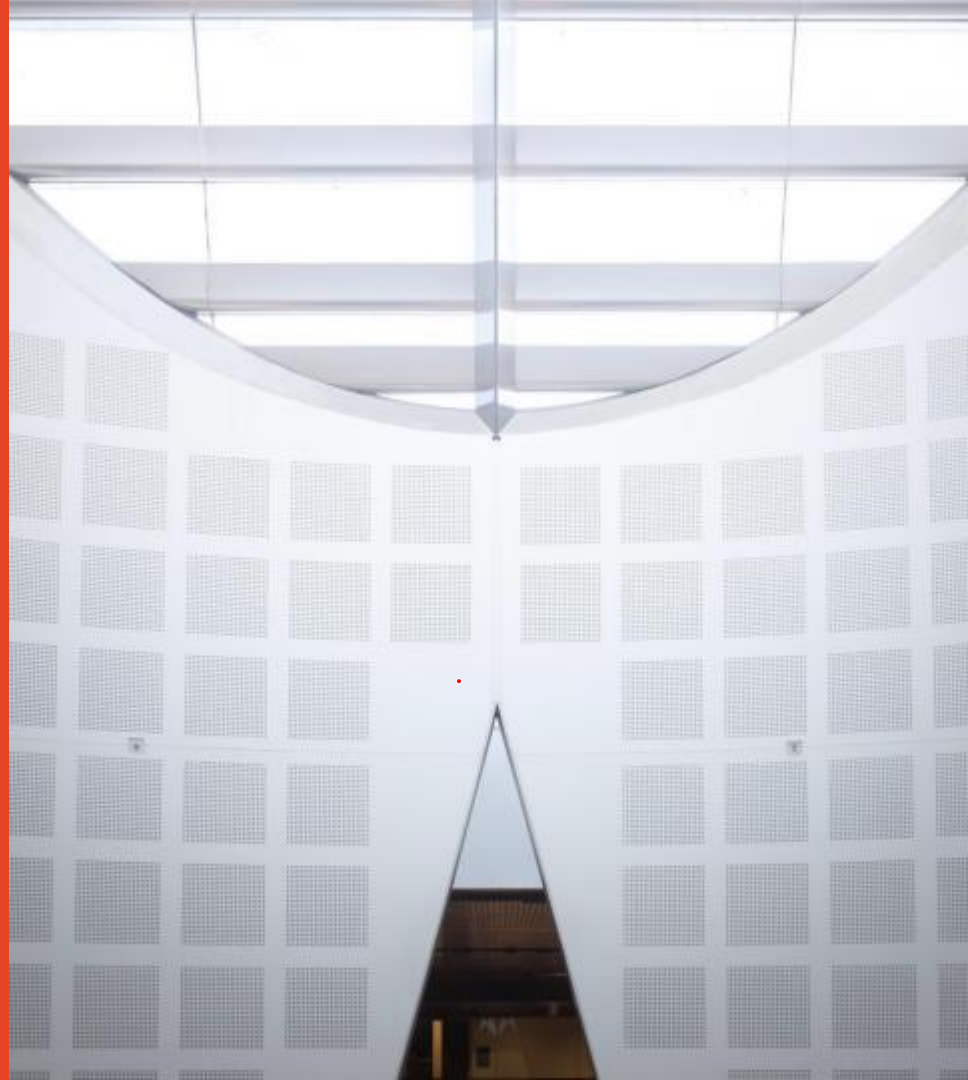


Agile Software Development Practices SOFT2412 / COMP9412

Team Dynamics; Tools and
Technologies for Teamwork

Xinyi Sheng

School of Computer Science

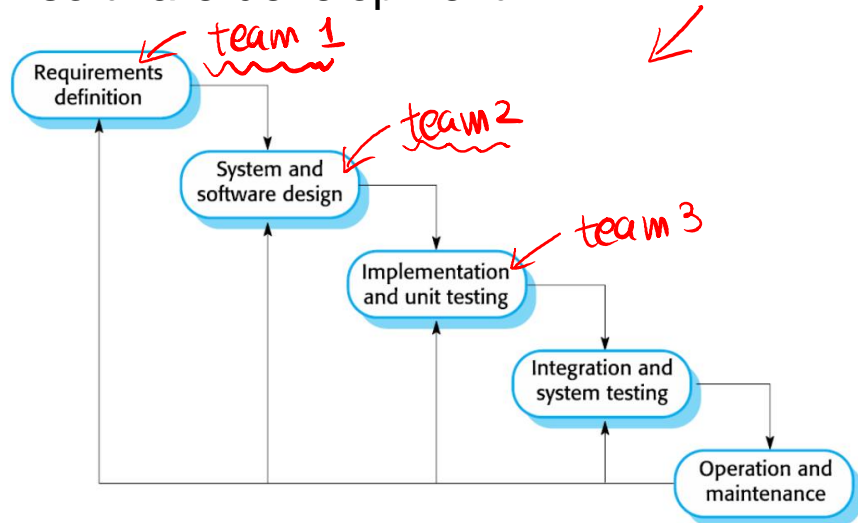


Agenda

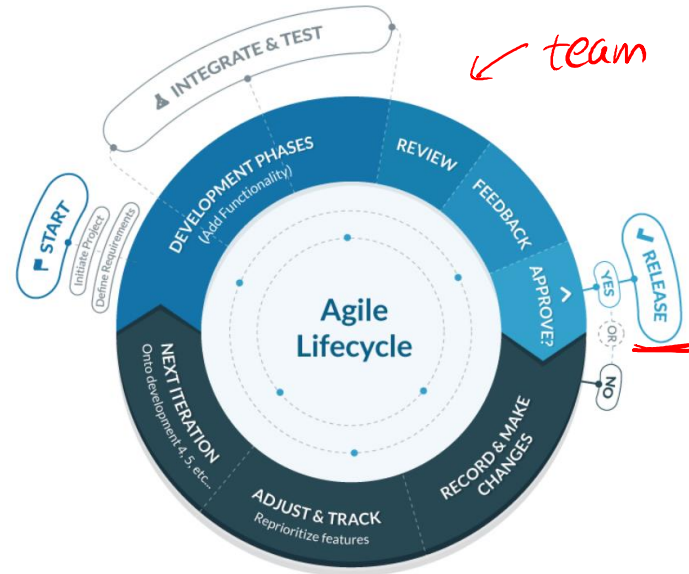
- Teams in Software Development Models
- Teams in Agile Development
- Team Dynamics ✓
- Effective Teams and Teamwork ✓
- Tools and Technologies for Teamwork

Software Development Models – Teams

Do software development models influence team structure and interactions in software development?



Waterfall model
plan-driven development



Agile model
incremental & iterative development

<https://blog.capterra.com/agile-vs-waterfall/>

Waterfall Model – Teams

Development activities ✓	Teams ✓
Divide the work into stages	A separate team of <u>specialists</u> for each stage
At each stage, the work is passed from one team to another	Some coordination is required for the handoff from team to team – using <u>“documents”</u>
At the end of all of <u>the stages</u> , you have a software product ready to ship	<u>As each team finishes, they are assigned to a new product</u>

Teams under different SDLC models

- In a traditional structure how do teams work?
 - As work is planned and allocated, it can be divided into pieces that should be more-or-less independent
 - Specialist teams
 - Project management and resource reallocation
 - Clear authority lines, so disagreements can be resolved
 - Problems?
 - Single points of failure
 - Inflexibility
 - Lack of feedback
- And in Agile teams?

Agile Manifesto – Revisit

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Agile Manifesto – Revisit

- Individuals and interactions over processes and tools
- Why Agile values individuals and interactions over processes and tools? Discuss

Agile Manifesto – Why Individuals and Interactions?

- Why Agile values individuals and interactions over processes and tools?
 - People tend to follow processes blindly, and make mistakes
 - “A great tool can sometimes help people to do the wrong thing faster”
 - Tools or best practices are not enough - people who need to use it should buy into it to realize its benefits
 - People needs to see the value of following certain practices
- It is important to recognize that you are working with a group of people who have different motivations, ideas and preferences

Agile Principles 1 – People

- Build projects around **motivated individuals**. Give them the **environment and support** they need, and **trust** them to get the job done
- The most efficient and effective method of **conveying information** to and within a development team is **face-to-face conversation**
- At regular intervals, the **team reflects** on how to become more **effective**, then **tunes and adjusts its behavior** accordingly

Agile Principles 1 – People

- *In-class Discussion:*

- *Briefly discuss how the following agile principles perceive teams and teamwork in agile development?*

- ① Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done
- ② The most efficient and effective method of conveying information to and within a development team is face-to-face conversation
- ③ At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly

Agile Principles 2 – People

- Business **people** and **developers** must **work together daily** throughout the project
 - The best architectures, requirements, and designs emerge from **self-organizing teams**
 - Our highest priority is to **satisfy the customer** through early and continuous delivery of valuable software
- feedback*
changes

Agile Principles 2 – People

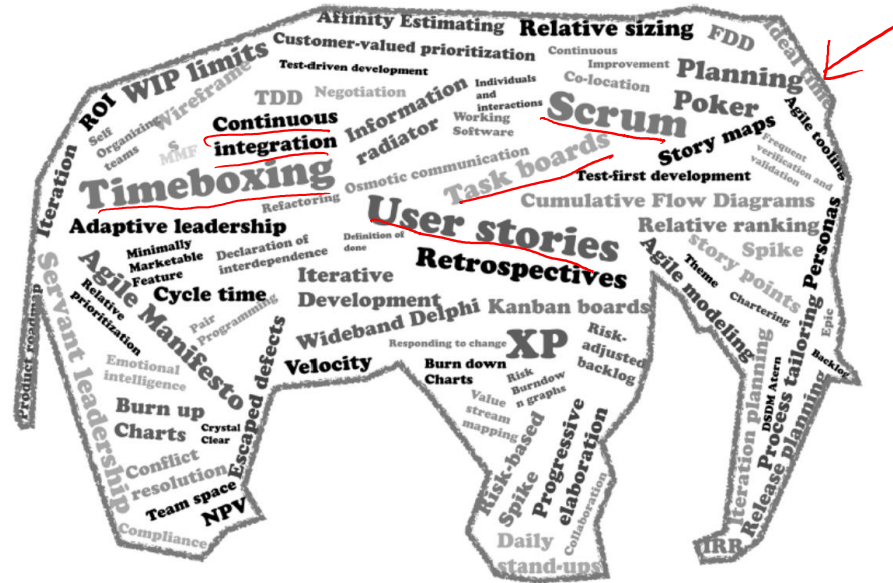
- *In-class discussion*
 - *briefly discuss how the following agile principles perceive teams and teamwork in agile development?*
- ④ Business people and developers must work together daily throughout the project
- ⑤ The best architectures, requirements, and designs emerge from self-organizing teams
- ⑥ Our highest priority is to satisfy the customer through early and continuous delivery of valuable software

Teams – Individuals and Collaboration

- Common problem experienced in software development teams
 - “throw it over the wall”*
 - Team members are busy thinking about their own project work and problems
 - Different views/perspectives
 - Teams are divided, and collaboration is killed

Agile Practices – The Agile Elephant

- The agile elephant is made up of many practices



*Agile practices and Scrum method (most of these presented in this figure) will be discussed in more details in future lectures. The focus of the discussion here is on teams and the adoption of Agile practice

Agile Teams – Individual Practices

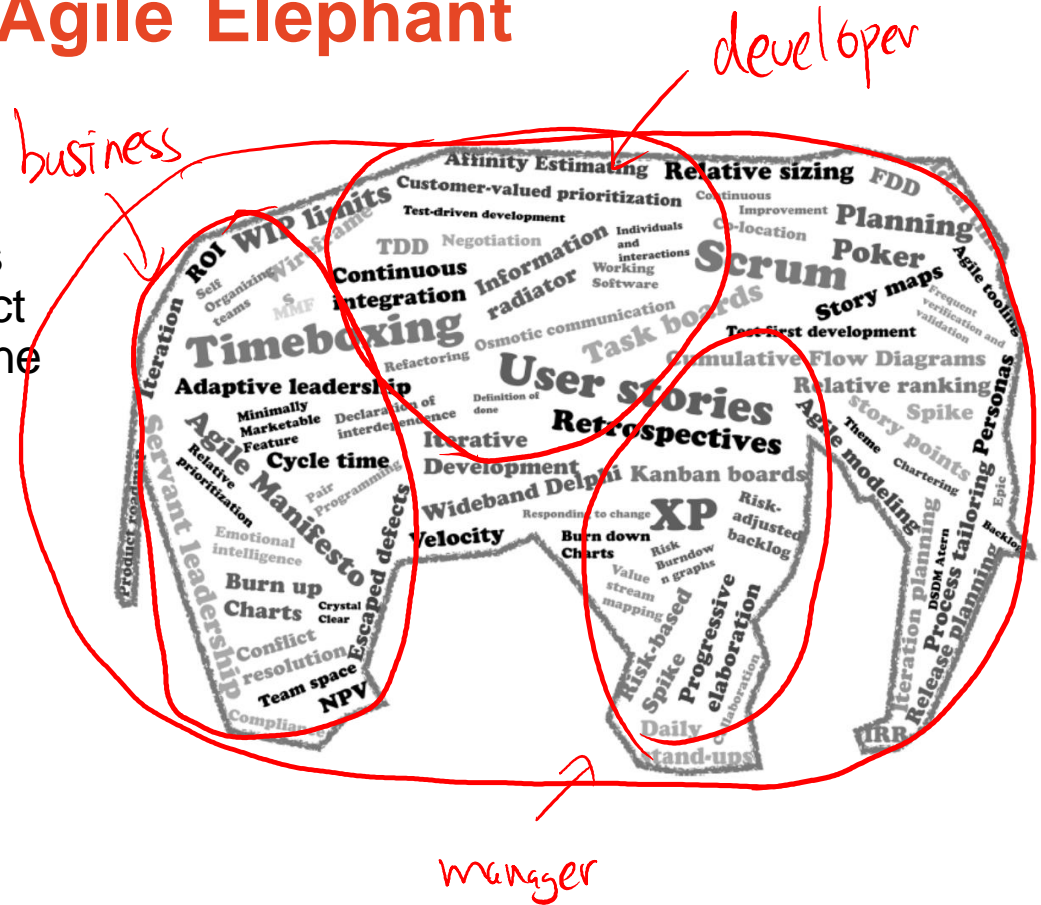
- When adopting agile practices, team members may adopt practices individually:
 - Each person uses only the practices that impact their work; developers focus on automated tests and build, team leads on task boards, project velocity and burn-down charts, business users on user stories
 - Adopting practices individually will improve things, but this may lead to a self-contradictory effect

Teams – Individual Practices

- When adopting agile practices, team members may follow the same thinking:
 - Each person uses only the practices that impact their work; developers focus on automated tests and build, team leads on task boards, project velocity and burn-down charts, business users on user stories
- Adopting practices individually will improve things, but this may lead to a self-contradictory effect
 - Each person sees the part of agile that affects their specific work – (attitudes: “see! I was right all along”)
 - Agile is made up of day-to-day practices, but it’s much bigger than those practices

Understanding the Agile Elephant

- If you only see the practices that directly affect your project work, then you will see the one small piece of agile



Team Dynamics



Team Dynamics

- “*Team dynamics are the unconscious, psychological forces that influence the direction of a team’s behaviour and performance*”
- Factors that lead to team dynamics:
 - Personalities and work styles
 - Knowledge and skills
 - Organization culture and structure
 - Cultural differences, background

Team Dynamics – Good or Bad?

In Agile development, is team dynamics a good or bad thing?

- Discuss

Team Dynamics – Pros and Cons

- Can be good
 - E.g., Improve overall team performance (productive conflict, different perspectives)
- Can be bad
 - Can lead to unproductive conflict can demotivate and prevent team from achieving its goals

Team Dynamics – Tuckman Team Development Model



Image: <https://www.atlassian.com/agile/teams>

Tuckman's stages of group development - https://en.wikipedia.org/wiki/Tuckman%27s_stages_of_group_development

Team Dynamics – Identification and Resolution

- Result from the interaction of many factors
 - E.g., Personalities, work style, roles, culture, organizational structure
- Investigate the root causes of conflict or poor team performance
 - Structured interviews or informal chats in a private and confidential
- Identify potential improvements
 - E.g., change in office layout, team development workshops (practices, personality dynamics, cultural change programs)

Source: <https://mysoftwarequality.wordpress.com/2014/09/04/cross-dysfunctional-teams>

Agile Teams – Skills

- It's claimed that agile teams work with the best developers. However, this is not necessary the case;
 - Every project needs at least one experienced and competent lead person (Critical Success Factor)
 - Each experienced and competent person on the team permits the presence of 4-5 “average” or learning people
 - With that skill mix, agile techniques have been shown to work many times

Teams Culture

- Consider the following scenarios:
 - *Developer*: “It works in our environments, it’s operations responsibility to make it work in production”
 - *Tester*: “Listen, it worked in User Acceptance Testing, it must be a configuration issue, or a missing firewall hole and nothing I could have spotted during testing ...”
 - *Customer*: “Hello! Nothing works here ...”
- Are these statements signs of team dynamics?
- Should this kind of culture exist in agile teams?

Source: <https://mysoftwarequality.wordpress.com/2014/09/04/cross-dysfunctional-teams>

Effective Teamwork

- Teamwork comprises of the right tools, the right people and the right practices
- Effective teamwork is everyone's shared responsibility
- Large software organizations, teams involve many roles across different departments (engineering, design, sales/marketing, legal)
- Use team building activities to build effective teams

Team Building

- “Various types of activities used to enhance social relations and define roles within teams, often involving collaborative tasks”
- There is an evidence how team building affect positively team effectiveness

https://en.wikipedia.org/wiki/Team_building

Team Building Activities

- **Goal setting:** emphasizes the importance of clear objectives and individual and team goals
 - E.g., objectives and key results (Atlassian)
- **Interpersonal relations:** focus on teamwork skills such as giving and receiving support, communication and sharing information
- **Standup meetings**
- **Roles and responsibilities** (Atlassian)

https://en.wikipedia.org/wiki/Team_building

Team Building – Roles and Responsibilities

- Define the roles and responsibilities that will make your team successful
- Clarify expectations as a team
- Helps to move a team from “storming” to “norming”, or help “performing” teams to get back on track

Roles and Responsibilities – How?

- Create a table of roles and responsibilities
 - Responsibilities from own perspective
 - Responsibilities from team member's perspective



Name	Role	Responsibility (own)	Responsibility (other's)

The table is a 4x4 grid. The first column is grey and labeled 'Name'. The next three columns are red and labeled 'Role', 'Responsibility (own)', and 'Responsibility (other's)' respectively. There are three empty rows below the header. Above the 'Role' header is one red checkmark. Above the 'Responsibility (own)' header are three red checkmarks. Above the 'Responsibility (other's)' header are three red checkmarks.

Roles and Responsibilities – How?

- Identify roles
 - E.g., team lead, developers, designers
 - Coarse-grained
 - Add to the role's column
- Clarify own responsibility
 - Think of top 3-5 tasks in priority order
 - Write on sticky notes
- Think of teammate's responsibilities
 - Write 1-2 responsibilities for each role from your perspective
 - Write responsibilities you may think that don't have a clear owner

Roles and Responsibilities – How?

- Refine and consolidate (optional)
 - Talk teammates with similar roles and refine responsibilities
- Discuss all roles
 - Role owner(s) describe their role and place their sticky notes in own responsibility column
 - Other teammate's role description in the other column
 - Owner to accept/decline the responsibilities by other teammates (suggest role to own it). Define primary owner for overlapping roles
 - Add “unassigned responsibilities” to
- Summarize roles and responsibilities
 - All to agree
 - Owner to document it and how to fill skill gaps

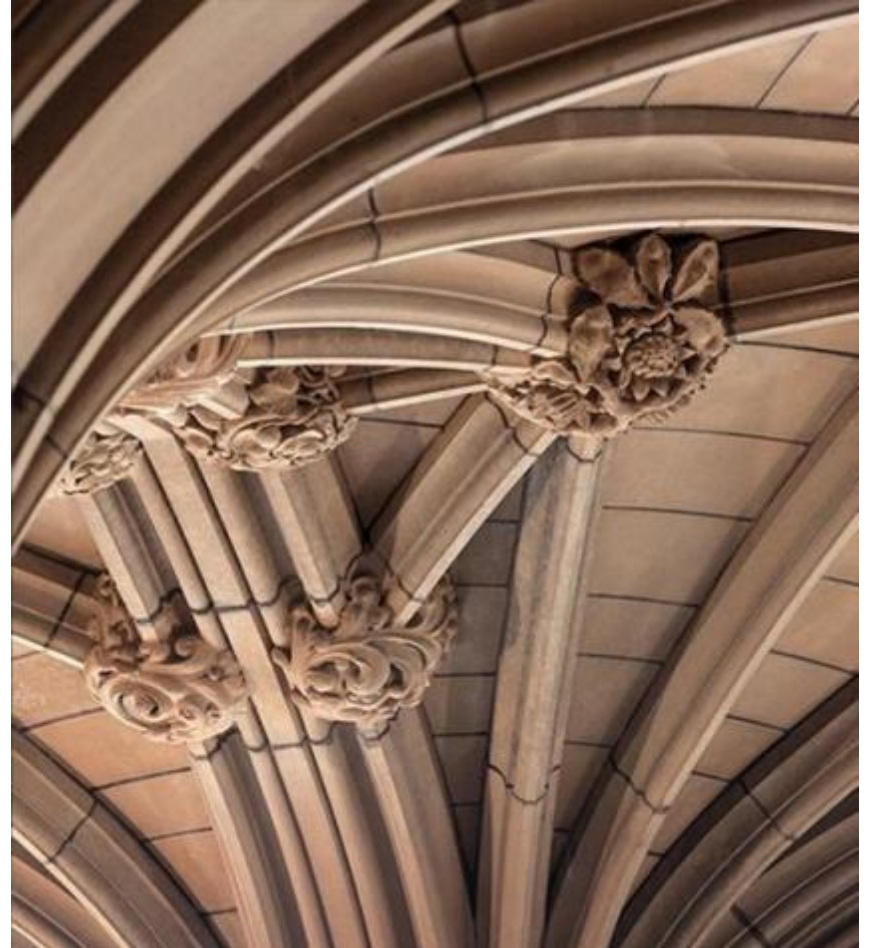
High-Performing Agile Teams

- Cross-functional; engineers, designers, architects, sales
- Mutual respect and mutual responsibility
 - Not blaming culture, and/or “throw it over the wall”
- Sound engineering practices (tools and automation)
- Value and belief of agile practices and principles
- Apply agile practices effectively as individuals and as a team
- Receive continuous training (technical and non-technical) and team monitoring/coaching

How much can you find out?

- Search for:
 - Team effectiveness
 - Self-managed teams
 - Group conflict
 - Team efficacy ...
- Learn about the theory of teams!

Tools and Technologies for Teamwork



Issue Tracking Systems

- A software that manages and maintains lists of issues
- Used to create, update and resolve reported issues internally or externally
- **Bug (defect) Tracking System:** keeps track of reported software bugs in software development projects
 - Centralized overview of development requests and their states
 - May assigned a priority, status, severity and/or complexity
 - Prioritized list of pending items (Backlog)
 - Typically integrated with other tools or software management systems

Bug/Issue Tracking Part of other Systems

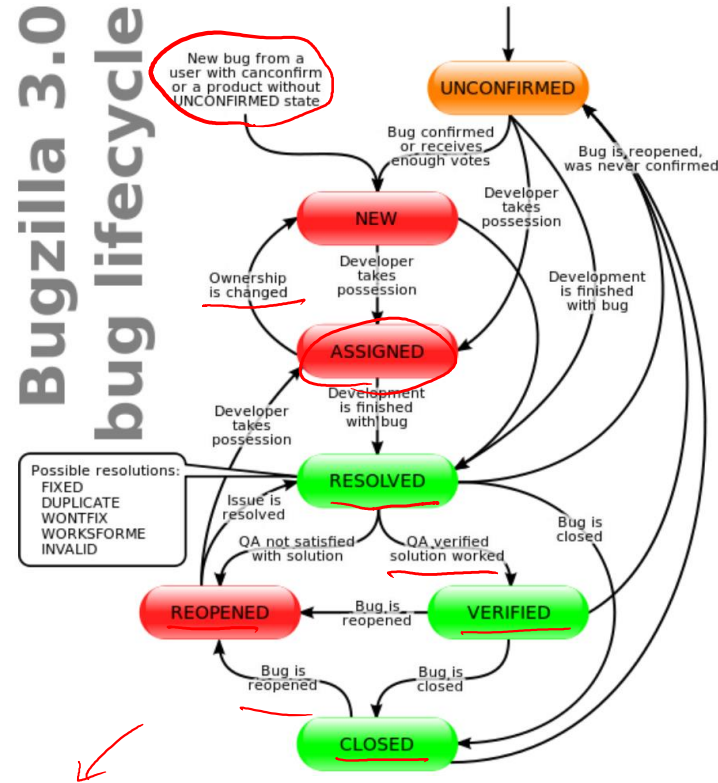
- Part of integrated project/software development management systems
- It helps integrating issue/bug tracking with other activities
- Distributed bug tracking tools are designed to be used with distributed revision control software

Bugzilla – Bug Tracking Tool

- Open-source web-based bug tracker and testing tool by Mozilla project
- Bug (or feature) requests can be submitted by anyone and will be assigned to a particular developer
- Various status updates for each bug
 - E.g., Bugzilla itself allows the public to file bugs – it assigns all bugs to a gatekeeper whose job is to assign responsibility and priority level

<https://en.wikipedia.org/wiki/Bugzilla>

Bugzilla – Bug Lifecycle



By Nyco [GFDL (<http://www.gnu.org/copyleft/fdl.html>) or CC BY-SA 3.0 (<https://creativecommons.org/licenses/by-sa/3.0/>)], from Wikimedia Commons

Issue Tracking – GitHub

SOFT2412-Agile-Software-Development / Test Private

Watch 0 Star 0 Fork 0

<> Code Issues 2 Pull requests 0 Projects 1 Wiki Insights Settings

Login bug

Write Preview

When a user enters a username or password with non-permitted characters such as a semicolon

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

Submit new issue

Assignees

bsul6138

asan0483

Labels

bug

Projects

Test repo

Milestone

No milestone

GitHub Issue Tracking – Edit Labels

SOFT2412-Agile-Software-Development / Test Private Watch 0 Star 0 Fork 0

Code Issues 2 Pull requests 0 Projects 1 Wiki Insights Settings

Search all labels Labels Milestones New label

8 labels Sort

bug	Something isn't working	Edit Delete
duplicate	This issue or pull request already exists	Edit Delete
enhancement	New feature or request	Edit Delete
good first issue	Good for newcomers	Edit Delete
help wanted	Extra attention is needed	Edit Delete
invalid	This doesn't seem right	Edit Delete
question	Further information is requested	Edit Delete
wontfix	This will not be worked on	Edit Delete

SOFT2412-Agile-Software-Development/Test/labels

GitHub Issue Tracking – Project Management

The screenshot displays the GitHub Projects interface for a project named "SOFT2412 - Agile Software Development Practices". The top navigation bar includes links for "Pull requests", "Issues", and "Explore". Below the project name, there are tabs for "Repositories", "People", "Teams", "Projects", and "Settings". The "Projects" tab is selected, showing a Kanban board with three columns: "To do", "In progress", and "Done".

- To do column:** Contains a list of tasks with checkboxes, including "Search filter user story", "Product Search User Story", "Automation", and "Welcome to GitHub Projects". A red box highlights this column.
- In progress column:** Contains one card titled "Test 2 - added" with the subtitle "Added by hoso5448". A red box highlights this column.
- Done column:** Contains two cards: "test" (Test#2 opened by ffar6831) and "Complete User Sign-up" (Test#3 opened by bsul6138). A red box highlights this column.

On the right side of the board, there is a button labeled "+ Add column" with a red box around it. At the top right of the board, there is a search bar labeled "Filter cards" and buttons for "+ Add cards (1 new)", "Fullscreen", and "Menu".

Version Control Systems – GitHub Revisit

- GitHub allows groups of people to collaborate across many projects at the same time in organizations account
- Organization's members can be:
 - **Owner**: complete administrative access to the organization
 - **Member**: default role for everyone else
- Owners can manage members' access to the organization's repos. and projects with fine-grained permission controls
- Can add collaborators from outside of the organization (consultant) to have access to one or more organization repos. without bring a member of the organization

GitHub – Organization Access Control (Revisit)

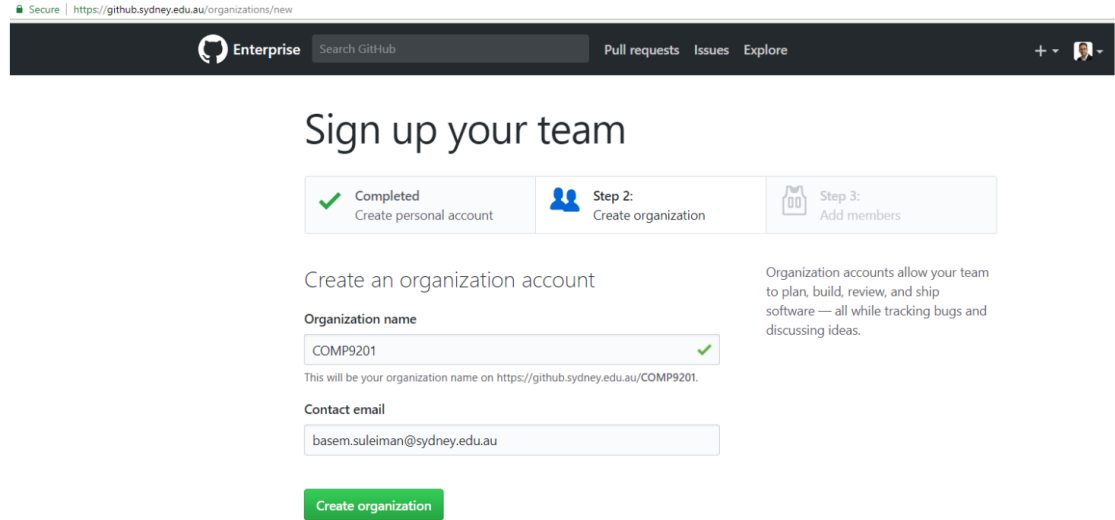
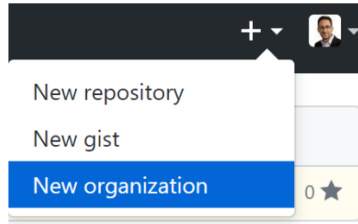
Organization action	Owners	Members
Invite people to join the organization	X	
Edit and cancel invitations to join the organization	X	
Remove members from the organization	X	
Reinstate former members to the organization	X	
Add and remove people from all teams	X	
Promote organization members to <i>team maintainer</i>	X	
Add collaborators to all repositories	X	
Access the organization audit log	X	
Delete all teams	X	
Delete the organization account, including all repositories	X	

Organization action	Owners	Members
Create teams	X	X
See all organization members and teams	X	X
@mention any visible team	X	X
Can be made a <i>team maintainer</i>	X	X
Transfer repositories	X	
View a project board and add or reorganize its cards and columns	X	X
Create or delete a project board and edit its description	X	X
Automate actions for project boards	X	X
View and post private team discussions to all teams (see " About team discussions " for details)	X	
Edit and delete team discussions in all teams (for more information, see " Managing disruptive comments ")	X	

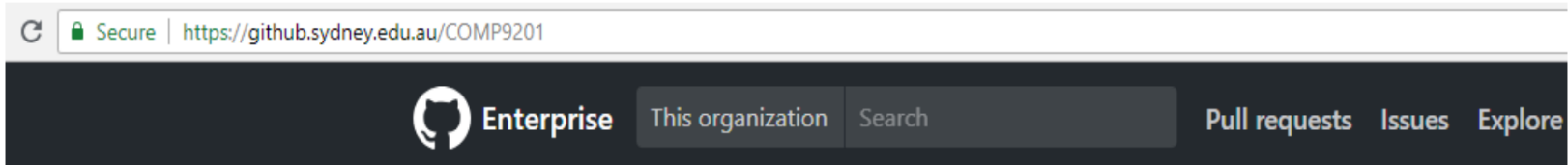
- Examples of access permissions for organization's owners and members

<https://help.github.com/enterprise/2.13/user/articles/permission-levels-for-an-organization/>

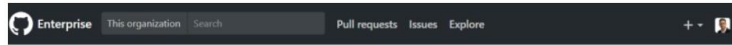
GitHub – Creating Organization (Revisit)



Organizational accounts have a namespace where all their projects exist



GitHub – Add Members to Organization (Revisit)



Add organization members

✓ Completed
Create personal account

Step 2:
Create organization

Step 3:
Add members

Search by username, full name or email address

hoso5448

hoso5448 Hamzah Bin Osoy

hoso5448

Famnaz Farid

Finish

Organization members

✓ See all repositories

✓ Create repositories

✓ Organize into teams

✓ Review code

✓ Communicate via @mentions

As an organization owner, you'll have complete access to all of the organization's repositories and have control of what members have access using fine-grained permissions.

- Note: when you create a new repo you can create them under your personal account or under any of the organizations that you're owner in

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

Repository name

COMP9201

/

WebStoreApp

Choose another owner

bsul6138

✓ COMP9201

INFO3220-Object-Oriented-Design

PROFESSIONAL-Software-Engineering

SOFT2201-Software-Constr-and-Design-1

SOFT2412-Agile-Software-Development

Need inspiration? How about

choose who can commit.

repository.

GitHub Organization – Manage Repos. (Revisit)

The screenshot shows the GitHub Organization page for 'COMP9201'. The top navigation bar includes 'Enterprise', 'This organization', 'Search', 'Pull requests', 'Issues', and 'Explore'. The main content area has a header with the organization name 'COMP9201' and a sub-header with tabs for 'Repositories', 'People', 'Teams', 'Projects', and 'Settings'. A message states 'This organization has no repositories.' with a 'Create a new repository' button. Below this, there is a search bar for repositories, a 'Type: All' dropdown, and a 'Customize pinned repositories' button. A list of repositories is shown, including 'Front-end', 'Designs', 'Back-end', and 'WebStoreApp'. On the right side, there is a 'People' section with a list of members: 'bsu6138 Basem Fathi Suleiman', 'ffar6831 Farnaz Farid', and 'hoso5448 Hamzah Bin Osoop'.

The screenshot shows the GitHub Repository page for 'COMP9201 / WebStoreApp'. The top navigation bar includes 'Enterprise', 'This repository', 'Search', 'Pull requests', 'Issues', and 'Explore'. The main content area has a header with the repository name 'COMP9201 / WebStoreApp' and a sub-header with tabs for 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. A message states 'This repository has no issues.' with a 'Create a new issue' button. Below this, there is a search bar for issues, a 'Type: All' dropdown, and a 'Customize pinned issues' button. A list of issues is shown, including 'Front-end', 'Designs', 'Back-end', and 'WebStoreApp'. On the right side, there is a 'Teams' section with a list of teams: 'FrontEndDeve', 'BackEndDeve', and 'Designers'. Each team has a dropdown menu for permissions (Read, Admin) and a button to 'Add someone'.

GitHub Organization – Manage People (Revisit)

COMP9201

Repositories 0 People 3 Teams 0 Projects 0 Settings

Find a member... Members Outside collaborators Add member

Select all	2FA	Role
Basem Fathi Suleiman bsul6138	2FA X Private	Owner 0 teams
Farnaz Farid ffar6831	2FA X Private	Member 0 teams
Hamzah Bin Osop hoso5448	2FA X Private	Member 0 teams

- Manage
- Change role...
- Convert to outside collaborator
- Remove from organization

Enterprise This organization Search Pull requests Issues Explore

COMP9201

Repositories 4 People 3 Teams 3 Projects 0 Settings

ffar6831 Farnaz Farid

Role: Member

3 repositories

1 team

Membership private

Two-factor security disabled

Convert to outside collaborator




Remove from organization

ffar6831 has access to 3 repositories

Find a repository they have access to...

Repository	Access Level	Manage access
COMP9201/WebStoreApp	Read on this repository	Manage access
COMP9201/Designs	Write on this repository	Manage access
COMP9201/Front-end	Admin on this repository	Manage access



GitHub Organization – Manage Teams (Revisit)

Find a team...	Import teams	New team
<input type="checkbox"/> Select all	Visibility ▾	Members ▾
<input type="checkbox"/> BackEndDeve Back-end Development Team	 2 members	0 teams
<input type="checkbox"/> Designers Web Application Designers	 1 member	0 teams
<input type="checkbox"/> FrontEndDeve Front-end development team	 2 members	0 teams

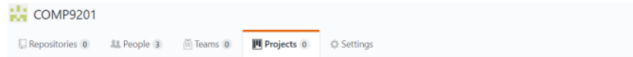
You may have 3 repos; Designs, Front-end and Back- end. You want FrontEndDeve to work on the Front-end and Designs repos, Designers team to work on Designs repo and BackEndDeve to work on Back-end repo

COMP9201 / BackEndDeve

Discussions Members 2 Teams 0 Repositories 2

Find a repository...	Add repository
<input type="checkbox"/> Select all	
<input type="checkbox"/> COMP9201/Back-end Private updated 21 minutes ago	 Admin ▾
<input type="checkbox"/> COMP9201/WebStoreApp Private updated an hour ago	 Admin ▾

GitHub Organization – Manage Projects (Revisit)



Organize your issues with project boards

Did you know you can manage projects in the same place you keep your code? Set up a project board on GitHub to streamline and automate your workflow.

[Learn More](#) [Create a project](#)



Sort tasks

Add issues and pull requests to your board and prioritize them alongside note cards containing ideas or task lists.



Plan your project

Sort tasks into columns by status. You can label columns with status indicators like "To Do", "In Progress", and "Done".



Automate your workflow

Set up triggering events to save time on project management—we'll move tasks into the right columns for you.



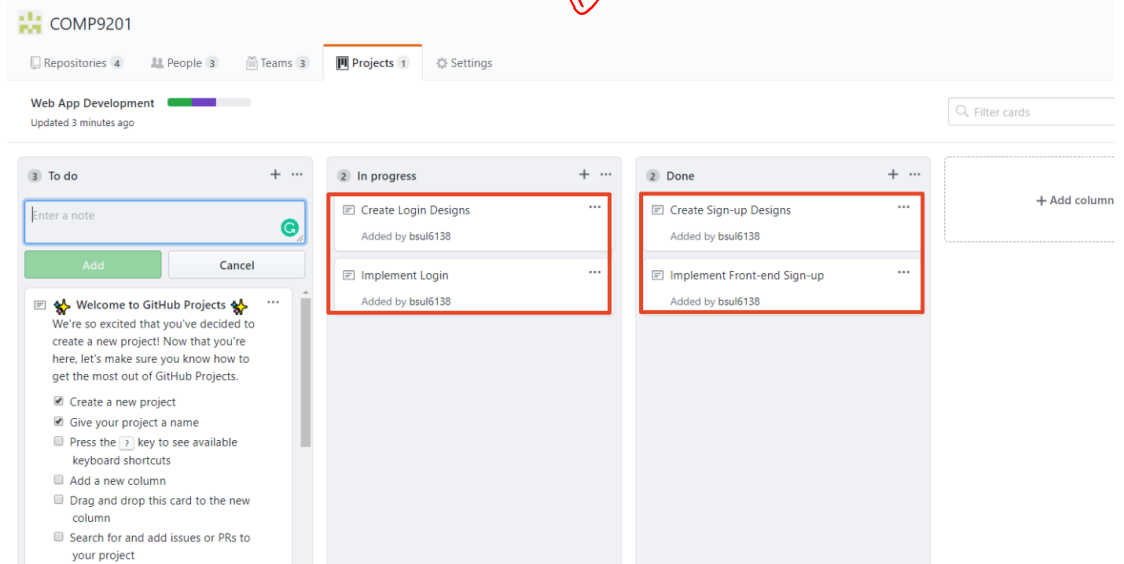
Track progress



Share status

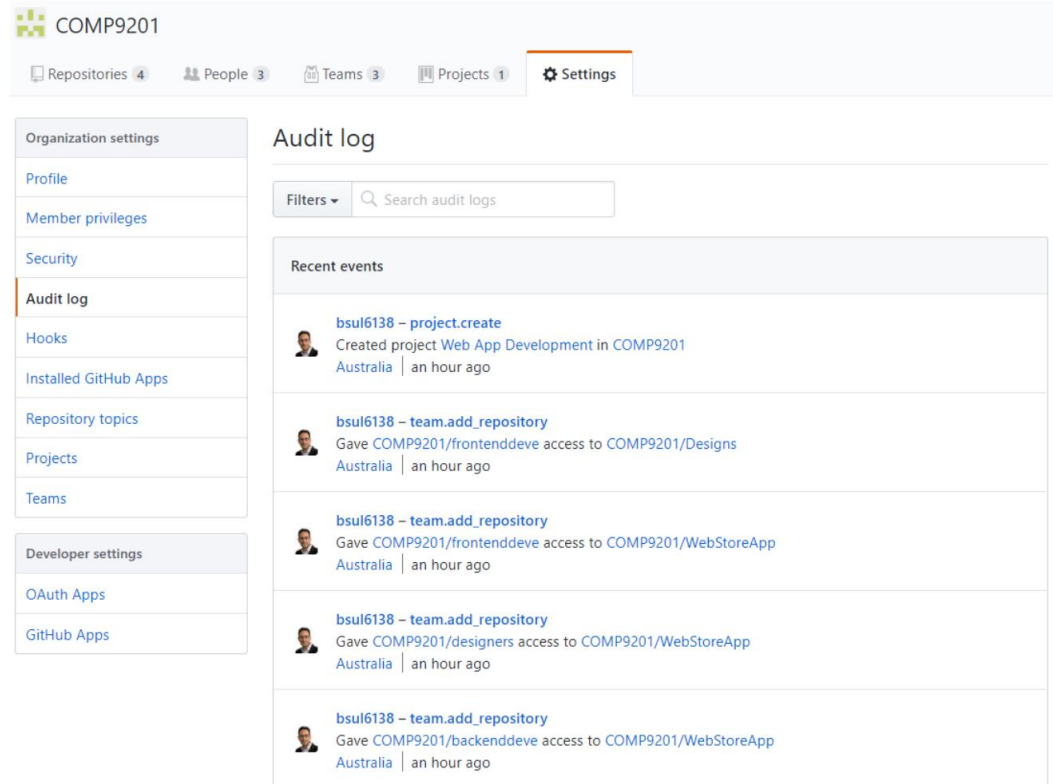


Wrap up



GitHub Organization – Audit Log (Revisit)

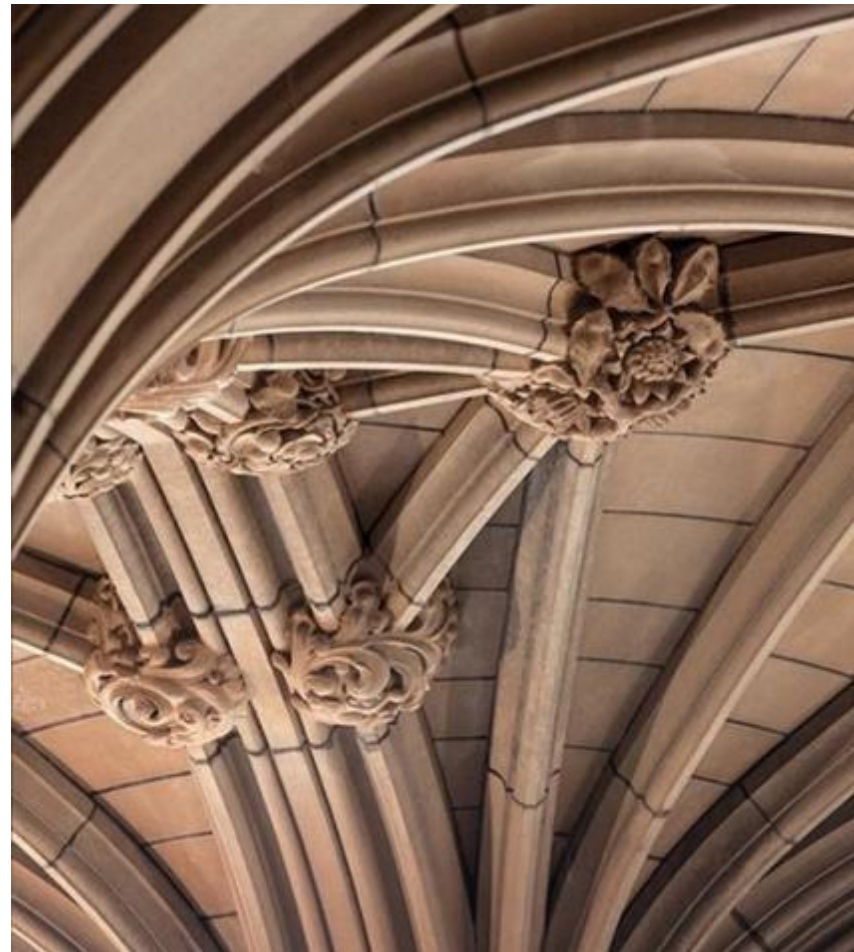
- Audit log records all events that have happened at the organization level, who did them and where in the world they were done



The screenshot shows the GitHub interface for the organization COMP9201. The top navigation bar includes tabs for Repositories (4), People (3), Teams (3), Projects (1), and Settings (selected). On the left sidebar, the 'Audit log' option is highlighted under the 'Organization settings' section. The main content area is titled 'Audit log' and features a search bar and a 'Filters' dropdown. Below this, a list of 'Recent events' is displayed, showing actions performed by user bsul6138:

- bsul6138 – project.create**
Created project Web App Development in COMP9201
Australia | an hour ago
- bsul6138 – team.add_repository**
Gave COMP9201/frontenddeve access to COMP9201/Designs
Australia | an hour ago
- bsul6138 – team.add_repository**
Gave COMP9201/frontenddeve access to COMP9201/WebStoreApp
Australia | an hour ago
- bsul6138 – team.add_repository**
Gave COMP9201/designers access to COMP9201/WebStoreApp
Australia | an hour ago
- bsul6138 – team.add_repository**
Gave COMP9201/backenddeve access to COMP9201/WebStoreApp
Australia | an hour ago

Jenkins

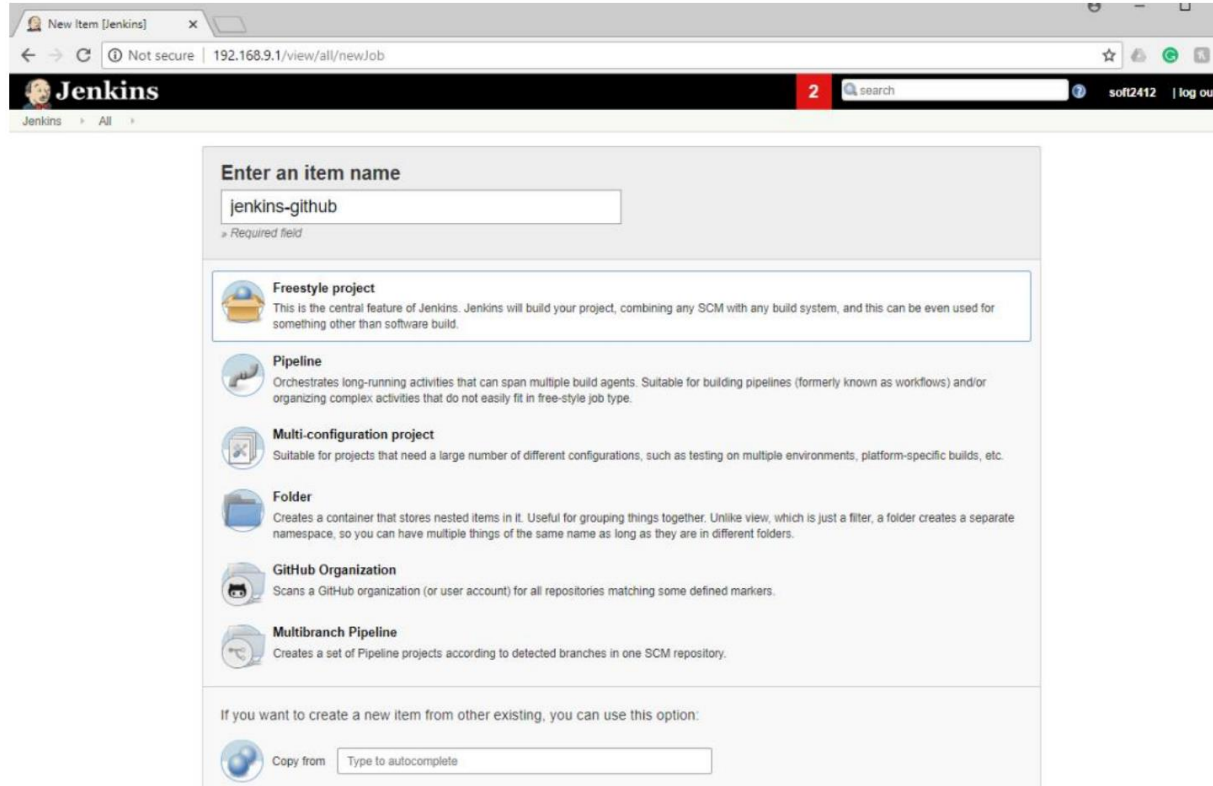


Jenkins – CI /CD



- “Jenkins is a self-contained, open source automation server which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.”
- Jenkins pipeline is a suite of plugins which supports implementing and integrating *continuous delivery pipelines* into Jenkins
 - A *continuous delivery pipeline* is an automated expression of your process for getting software from version control right through to end users/customers
 - Typically written in *Jenkinsfile* which is checked in a project's source code repository

Jenkins – Integration with GitHub (1)



The screenshot shows the Jenkins web interface in a browser. The address bar indicates the URL is 192.168.9.1/view/all/newJob. The Jenkins logo and name are visible in the top left. A search bar and a 'log out' link are in the top right. The main content area is titled 'Enter an item name' and contains a text input field with the value 'jenkins-github'. Below this, there is a list of project types, each with an icon and a description:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- GitHub Organization**: Scans a GitHub organization (or user account) for all repositories matching some defined markers.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.

At the bottom, there is a section titled 'If you want to create a new item from other existing, you can use this option:' with a 'Copy from' label and an input field with the placeholder text 'Type to autocomplete'.

Jenkins – Integration with GitHub (2)

The image shows two screenshots of the Jenkins configuration interface. The top screenshot is the 'Source Code Management' section, and the bottom screenshot is the 'Build Triggers' section.

Source Code Management

- ☐ None
- ☒ Git

Repositories

Repository URL:

Credentials:

Branches to build

Branch Specifier (blank for 'any'):

Repository browser:

Additional Behaviours:

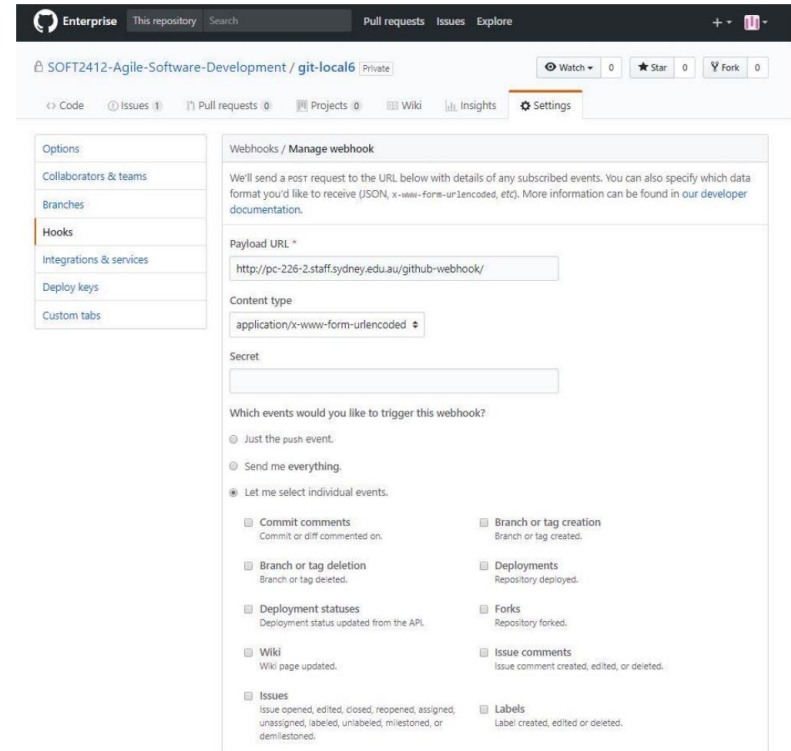
☐ Subversion

Build Triggers

- ☐ Trigger builds remotely (e.g., from scripts)
- ☐ Build after other projects are built
- ☐ Build periodically
- ☐ GitHub Branches
- ☐ GitHub Pull Requests
- ☒ GitHub hook trigger for GITScm polling
- ☐ Poll SCM

Jenkins – Integration with GitHub (3)

- Webhooks to set up GitHub applications to subscribe to certain events on GitHub
- Events is triggered, HTTP POST payload will be sent to the webhook's configured URL
- Webhooks can be used to update an external issue tracker, trigger CI builds, update a backup mirror



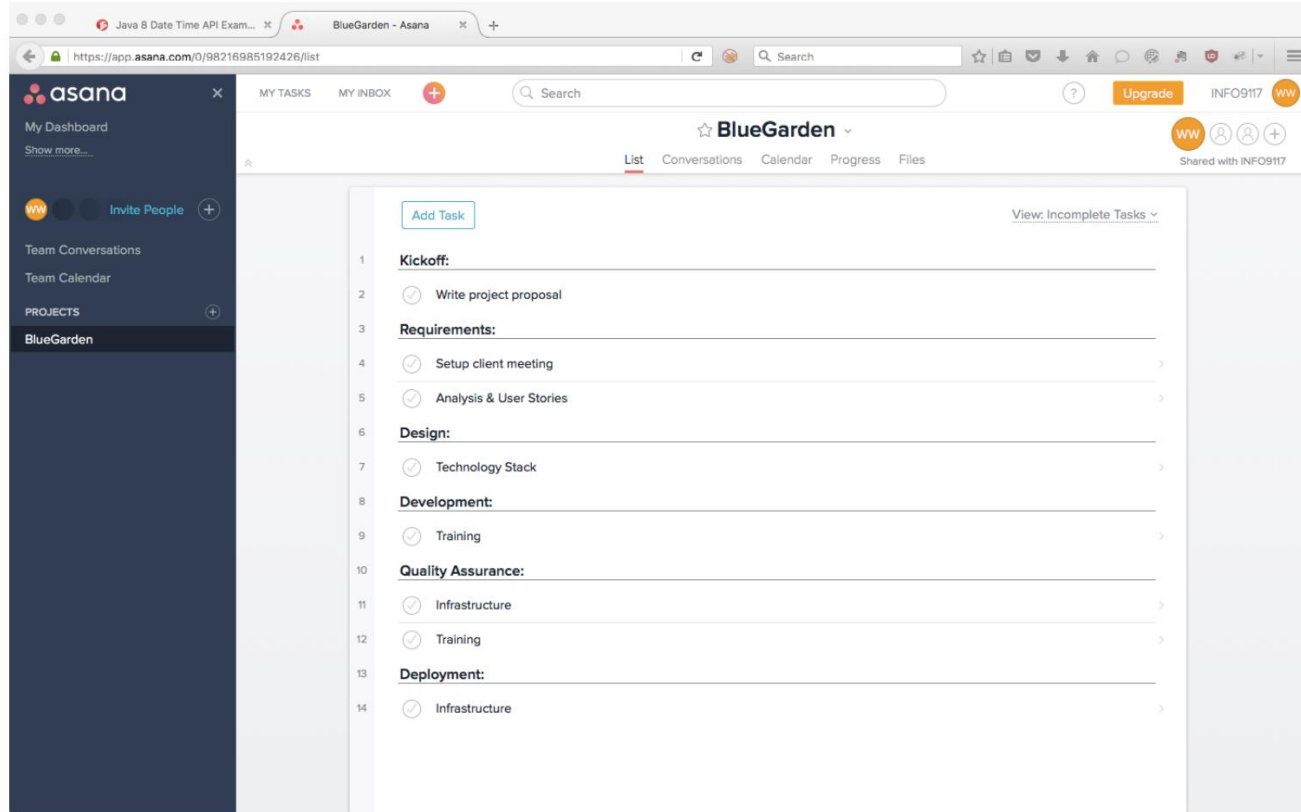
The screenshot shows the GitHub 'Manage webhook' interface. On the left, a sidebar contains links for 'Options', 'Collaborators & teams', 'Branches', 'Hooks' (which is highlighted), 'Integrations & services', 'Deploy keys', and 'Custom tabs'. The main content area is titled 'Webhooks / Manage webhook' and includes an introductory paragraph about sending POST requests. Below this, there are input fields for 'Payload URL *' (containing 'http://pc-226-2.staff.sydney.edu.au/github-webhook/'), 'Content type' (set to 'application/x-www-form-urlencoded'), and a 'Secret' field. A section titled 'Which events would you like to trigger this webhook?' contains three radio buttons: 'Just the push event.', 'Send me everything.', and 'Let me select individual events.' (which is selected). Under the selected option, there are two columns of checkboxes for various events: 'Commit comments', 'Branch or tag deletion', 'Deployment statuses', 'Wiki', 'Issues', 'Branch or tag creation', 'Deployments', 'Forks', 'Issue comments', and 'Labels'. Each checkbox is accompanied by a brief description of the event.

<https://developer.github.com/webhooks/>

Teamwork Collaboration Tools

- What tools would you use to help support your team?
- Examples:
 - Dropbox ✓
 - Google Docs ✓
 - Skype ✓
 - Trello ✓
 - Slack ✓
 - Basecamp ✓
 - Asana ✓
 - ... other

Other Tools – Asana – Project Management



References

- Andrew Stellman, Margaret C. L. Greene 2014. Learning Agile: Understanding Scrum, XP, Lean and Kanban (1st Edition). O'Reilly, CA, USA.
- S. P. Myers, 2013. [<https://www.teamtechnology.co.uk/team/dynamics/definition/>]
- Cross Dysfunctional Teams. [<https://mysoftwarequality.wordpress.com/2014/09/04/cross-dysfunctional-teams>]
- Atlassian, Agile Teams. [<https://www.atlassian.com/agile/teams>]
- Tuckman's stages of group development. [https://en.wikipedia.org/wiki/Tuckman%27s_stages_of_group_development]
- Atlassian, Teamwork. [<https://www.atlassian.com/teamwork>]
- Team building. [https://en.wikipedia.org/wiki/Team_building]
- Issue tracking system. [https://en.wikipedia.org/wiki/Issue_tracking_system]
- Bug tracking system. [https://en.wikipedia.org/wiki/Bug_tracking_system]

References

- Further Readings:
 - Hackman J. R., “*Leading Teams: Setting the stage for great performances*”, Harvard Business Press 2002
 - Hackman J. R., Katz N. “*Group behavior and performance*”. In Fiske ST, Gilbert DT, Lindzey G Handbook of social psychology (5th ed.) New York: Wiley; 2010. pp. 1208-1251. DOI: 10.1002/9780470561119.socpsy002032

Tutorial: Continuous Integration with Jenkins / Docker

Team Dynamics – Team Building Activities

Next Week: Agile Methods, Scrum

- Scrum team ✓
- Scrum events ✓
- Scrum artifacts ✓
- Scrum estimation ✓

