REVIEW TYPE AND GOAL

- First time review
- Someone asking to upgrade one of the components (dependencies). There is not so much code inside the PR.

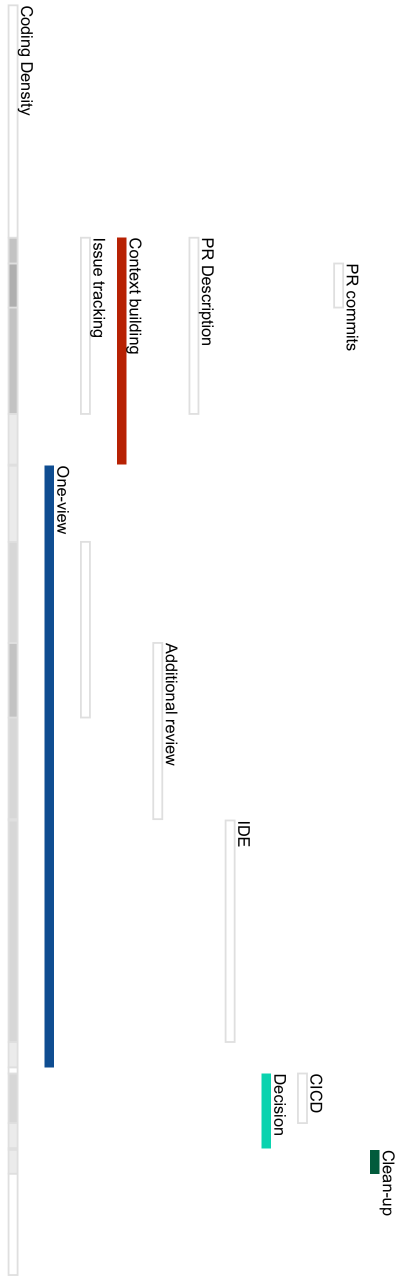
CONTEXT

TRANSCRIPT

- Checks first if the PR title is written according to their processes - for example, add Jira issue ID in the title and in the commit
- Switches to the commits tab to check the commit title
- Goes back to Conversations. Opens the Jira issue linked in the PR description - has all the information they need to perform the review
- There is not much description in the Jira as it is just a component upgrade, but it says the component is part of the test suite
- The component matters because if it would be related to security, someone else would be reviewing that component
- Some PRs are tied to a very specific component and then only the project lead of the component (or component lead) might be able to review it
- This review is an easy one - only an upgrade, not a lot of code changes. Therefore, needs to check what is new with this component.
- Looks at which version upgrades to which version in the diff of Files Changed
- Opens the linked Jira issue to check the differences in the versions, there is link in Description field of the issue that directly takes to the comparison of the versions on GitHub - follows the link
- Scrolls through the listed commits twice
- Sees it just to have an overview because the component is just part of the test suite so it is not a component that they put to the server or affects the server so the rules to verify the upgrade are more relaxed
- Then they want to verify where that component is used in the project
- Opens the project in their local fork
- checks in the IDE where the component is used and think about what needs to be affected with this upgrade
- Sees where the dependency is used which can show which parts of the project can be affected.
- Goes to GitHub to copy the dependency name from the diff
- Checks in the IDE - only one module is using it and it affects just the test suite
- Goes back to GitHub
- Since the PR is passing all the integration jobs they can just easily approve and there is nothing to discuss
- Submits the review as “Approve” without a further comment
- Adds a label “ready-to-merge’

GENERAL STRATEGY

Looks at the documentation, looks at the code, checks the comparison



of changes and checks what is affected in the IDE, then checks the CI/CD

INTERVIEW
I: can I maybe ask, now, because you mentioned that the process in your project is somehow specific for the code reviews. Well, what do you mean by that?

P4: for example, we have to verify the pull request commit contains valid information. For example, the Jira issue is in the commit itself because, later, one day someone is reviewing why these changes was applied and the way to get more information about the reason of that change is to navigate to the Jira issue. So if you check the Git history, you can check the Jira issue ticket number and you can open Jira and get more information about
the PR: what was the problem it was trying to resolve, what is the additional intention of the of the pull request, if the pull request [not understandable;maybe "bad" or "bug"] or whatever

Coding Density
One-view
Issue tracking
Context building
Additional review
PR Description
IDE
Decision
CICD
PR commits
Clean-up