Module 10
Agile Teams

CS 169A: Software Engineering

1 Overview

Becoming a proficient software engineer is itself a continuous, lifelong feat. However, what sometimes is more challenging is learning to work with other people. Whether its a group project, a sports team, or a student organization, chances are, you've already run into the nuances that come with working with people!

Module 10 is completely dedicate to the processes and workflows that make the human component of the Agile workflow successful. Specifically, we will look at processes and tools for debugging, checking in code, and deploying to production. Just remember that the core to great team dynamics is ensuring *everyone feels psychologically safe* to foster an environment where anyone can speak their mind without fear of failure or humiliation.

2 Version Control

From prior classes, hopefully you are familiar with version control as a tool for taking snapshots of your work and backing them up. *Branches* are a feature that allow you to simultaneously work collaboratively on the same codebase with multiple engineers! Answer the questions below about how to resolve different scenarios in Git version control:

2.1

Lay out the steps and commands that you would use to resolve a merge conflict in Git.

Your answer may be more comprehensive, but the bare essentials are as follows:

1. Identify the files that have caused the conflict.
2. Make the necessary changes in the files to resolve the conflict. (i.e. choose the correct version, manually rewrite existing code)
3. Add these files with `git add`
4. Commit the changed files with `git commit`.

2.2

What is the difference between Git and Github?

Git is a distributed version control system that tracks changes to source code during the development process. On the other hand, GitHub is a Git repository hosting service with multiple additional features and a UI for viewing changes.

2.3

What is the difference between a fork, a branch, and a clone of a Git repository?

- A fork is a copy of the repository that still holds the reference to the original repository, but allows one to freely experiment without affecting the original project. Changes on forks can be submitted as a PR to propose a change.
- A clone makes a direct copy of a repository. When code is edited, the changes would be added and pushed to the original repository.

- As discussed in the textbook, a branch is a copy of source code at a particular commit of an existing branch.

2.4

What is the difference between `rebase` and `merge` in Git?

The two commands perform a very similar action of integrating changes from one branch into another. They differ in how this action is performed. With `merge`, the changes are added to the master branch as one or more commits. On the other hand, `rebase` compresses all the changes into a single "patch." Then it integrates the patch onto the target branch. In other words, `rebase` "flattens" history and eliminates unwanted history when transferring work from one branch to another.

2.5

True or false: If you attempt `git push` and it fails with a message such as `Non-fast-forward (error): failed to push some refs,` this means some file contains a merge conflict between your repo's version and the origin repo's version. Not necessarily. It just means that your copy of the repo is missing some commits that are present in the origin copy, and until you merge in those missing commits, you won't be allowed to push your own commits. Merging in these missing commits may lead to a merge conflict, but frequently does not.

3 Five R's of Bug Fixes

Debugging is an inevitable part of software development, but luckily, there's a team workflow in place for resolving them when they come up. The 5 R's are:

- Reporting a bug
- Reproducing the problem, or else Reclassifying it as "not a bug" or "won't be fixed"
- Creating a Regression test that demonstrates the bug
- Repairing the bug
- Releasing the repaired code

3.1

Assume you a part of a software engineering team building a ride-sharing application. Determine and justify if and why the following backlog items should be classified as bugs:

- A beta tester of a limited edition of the app reports a button that is not redirecting properly when clicked. It has been decided that the limited edition version of the app will not be deployed.

  No, this bug is in a part of the code that is being undeployed or is otherwise no longer supported
- A user with an iPhone 4s and an outdated version of iOS reports that certain UI elements are not rendering properly.

  No, this bug occurs only with an unsupported user environment, such as a very old browser lacking necessary features for this SaaS app.
- A duplicate of a previous issue that has been fixed in the upcoming to be released version of the app.

  No, This bug is already fixed in the latest version (uncommon in SaaS, whose users are always using the latest version)
- A user does not like that the app doesn't include information about the driver and requests it to be displayed.

  No, this is not a bug but a request to make an enhancement or change behavior that's working as designed.
- When a user requests a ride from an airport, the designated driver receives an incorrect pick up location.

  Yes. The app is not behaving as expected.

3.2

Why do you think “bug fix” stories are worth zero points in Tracker even though they follow the same lifecycle as regular user stories?

A team’s velocity would be artificially inflated by fixing bugs, since they’d get points for implementing the feature in the first place and then more points for actually getting the implementation right

3.3

True or false: a bug that is triggered by interacting with another service (for example, authentication via Twitter) cannot be captured in a regression test because the necessary conditions would require us to control Twitter’s behavior.

False: integration-level mocking and stubbing, for example using the `FakeWeb gem` or the techniques described in Section 8.4, can almost always be used to mimic the external conditions necessary to reproduce the bug in an automated test.

3.4

True or false: a bug in which the browser renders the wrong content or layout due to JavaScript problems might be reproducible manually by a human being, but it cannot be captured in an automated regression test.

False: tools such as Jasmine and Webdriver can be used to develop such tests.