Remote Work and Collaboration

AI Engineering - Recitation 2

Outline

- Introduction to Git
- Git Best Practices
- Branching Strategies
- Pull Requests & Merge conflicts
- Things to consider for remote collaboration

Introduction to Git

- Distributed version control system
- Centralized code repository
- Users can have local copies of the code
- Local updates get pushed to the central repo
- Commits to the central repo are authenticated
- Git is line-based
- Version control provides traceability benefits
- Desktop/GUI support

Common Git Operations

Clone	Make a local copy of a remote repository
Pull	Download latest code from a remote repository to a local repository
Commit	Update code/ create a new revision in the local repository
Push	Update the remote repository with the changes on the local repository
Checkout	Switch to a branch / make a branch as your working branch
Merge	Merge code from two branches

Git: Best Practices

- Use meaningful, descriptive commit messages
- Commit frequently (for completion of a small logical unit of code)
- Avoid committing generating files
- Use branches wisely
- Make use of pull requests (add description to help the reviewer)
- Have a defined process (or) Git workflow
- Aim to version control all code (IaaC, configurations, etc.)

- Cheatsheet to fix git screw-ups (SFW version) - https://dangitgit.com/

Source: DevOps lecture, Summer 2020

Branching Strategies

- Many strategies out there, it’s important to pick one and stick with it
    - Even if it is something simple, having one is a must
- My recommendation is to use GitLab flow:
    - Cut feature branches from master, and never make direct commits to master
    - Pull Requests made should have a code review before merging to master
    - Run unit tests before merging to master
    - Releases made by tags
    - Advantages: Git history is clean, CI/CD is easier, ideal for single version in production
- Generally speaking,
    - Branch names must indicate a purpose
    - Branches should be deleted upon merging

Source: 4 branching workflows for Git

Pull Requests

- We looked at branching strategies earlier
- Well, how would you merge code from a child branch back to the parent?
    - Do you merge directly?
    - Ideally, there should be a ‘process’ in place
    - Pull request is a widely adopted way to do this
    - Pull requests have their own set of good practices

Pull Requests: Best Practices

- One pull request is a complete update to the code
  - May be a feature or bugfix, should be a complete unit
- It should not have multiple features intertwined
  - While delivering feature A, avoid committing code related to feature B
  - Use stash/branching strategies to avoid this
- Always sync with target branch before raising a pull request
  - Always check for other open pull requests, and only open one if your changes are higher priority
  - Any conflicts in the code should be resolved before raising a pull request

Resolving merge conflicts

- A merge conflict happens when two or more commits contradict each other
  - For example, the commits editing the same line in the code
- Even with outstanding communication, conflicts are going to happen
- Easiest way to avoid is to pull from remote repository regularly
  - Typically done at the start of a workday
- With clean git practices, it may not be as hard as you think to fix
- Typical process is as follows-
  - Git highlights the conflicted area, and retains the lines from both branches
  - You decide which line to keep (you may keep both, one or none)
  - Remove the highlighted portion inserted by git
  - After all conflicts are resolved, commit the changes

Demo - Pull requests and merge conflicts

Activity - Remote Work Barriers & Mitigation Tactics

- In your breakout rooms
  - Think about potential barriers to remote collaboration and teamwork
  - What are potential strategies that you can adopt to mitigate these problems?
- After 5 minutes, come back to the main room, and
  - Explain what problems you identified
  - What strategies did you come up with to resolve those problems

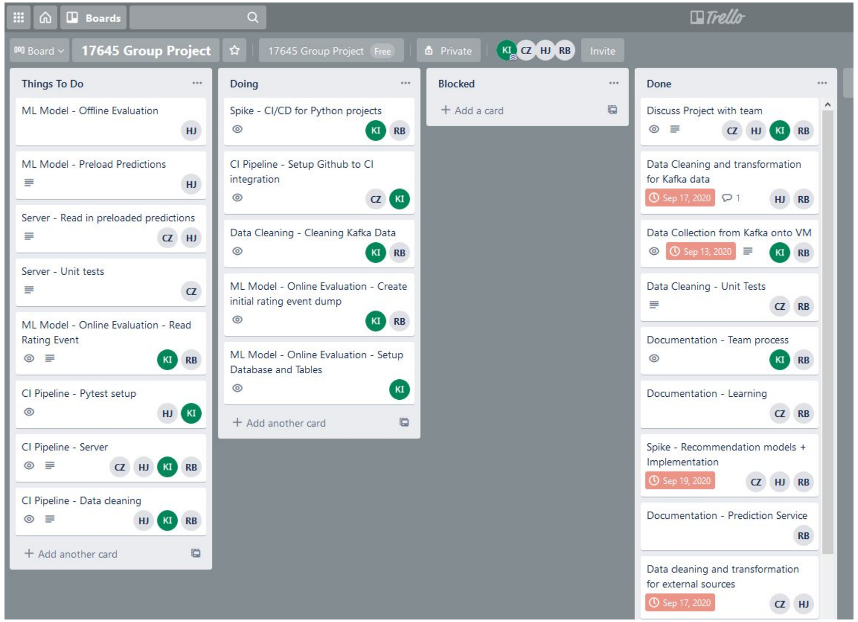
Things to Consider

- Communication [why? what? how? When?]
  - Consider availability of team members
  - Document discussions, key decisions, work to be done, commitments, etc.
- Areas of expertise
  - Get to know the strengths of each team member
- Representation of work
  - Break down your work into smaller tasks
  - Spend some time to identify dependencies
  - Is everyone aware of what they need to work on, and what is the status of the team as a whole?
- Pair programming can actually be fun to try

Things to Consider

- Plan for the future
- Help each other
- Don’t rush to implementation
  - Spend adequate time to design your system
  - Consider current and future requirements
- Clarify any assumptions you make
- Make choices considering the time available, expertise in the technology used, etc.

Tool recommendation - Trello

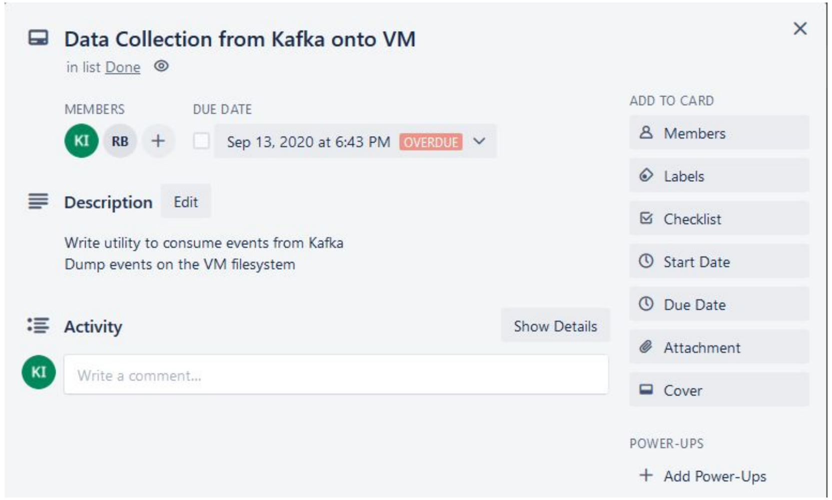


Things To Do
- ML Model - Offline Evaluation (HJ)
- ML Model - Preload Predictions (HJ)
- Server - Read in preloaded predictions (CZ, HJ)
- Server - Unit tests (CZ)
- ML Model - Online Evaluation - Read Rating Event (KI, RB)
- CI Pipeline - Pytest setup (HJ, KI)
- CI Pipeline - Server (CZ, HJ, KI, RB)
- CI Pipeline - Data cleaning (HJ, KI, RB)
- + Add another card

Doing
- Spike - CI/CD for Python projects (KI, RB)
- CI Pipeline - Setup Github to CI integration (CZ, KI)
- Data Cleaning - Cleaning Kafka Data (KI, RB)
- ML Model - Online Evaluation - Create initial rating event dump (KI, RB)
- ML Model - Online Evaluation - Setup Database and Tables (KI)
- + Add another card

Blocked
- + Add a card

Done
- Discuss Project with team (CZ, HJ, KI, RB)
- Data Cleaning and transformation for Kafka data — Sep 17, 2020 — 1 (HJ, RB)
- Data Collection from Kafka onto VM — Sep 13, 2020 (KI, RB)
- Data Cleaning - Unit Tests (CZ, RB)
- Documentation - Team process (KI, RB)
- Documentation - Learning (CZ, RB)
- Spike - Recommendation models + Implementation — Sep 19, 2020 (CZ, HJ, RB)
- Documentation - Prediction Service (RB)
- Data cleaning and transformation for external sources — Sep 17, 2020 (CZ, HJ)



Data Collection from Kafka onto VM

in list Done

MEMBERS: KI RB

DUE DATE: Sep 13, 2020 at 6:43 PM OVERDUE

Description

Write utility to consume events from Kafka
Dump events on the VM filesystem

Activity

KI Write a comment...