# GitHub Actions Verification Overview

## ****Submission File Verification****

## **All repositories will come with a set of GitHub Actions which can be used by contestants to verify the fidelity of their submissions for each of the three challenges.**

## **For any questions/issues with the below process please reach out through our Slack channel.**

## ****Steps to Verify Submission File on Main Branch****

At the onset of the HTM competition, each team will be provided a GitHub repository for all code and submission files to be checked into. Any time that code is pushed to the “main” branch of this repository, a set of GitHub Actions will be kicked off to validate any submission files that have been placed in the repository. (For more information on the naming convention and repository structure that should be followed, refer to the READMEs for each of the challenges)

Any time that you have checked in a submission file to the “main” branch for any of the challenges, it is recommended to verify that the submission file has passed the validation checker within GitHub. Steps to validate submission file:

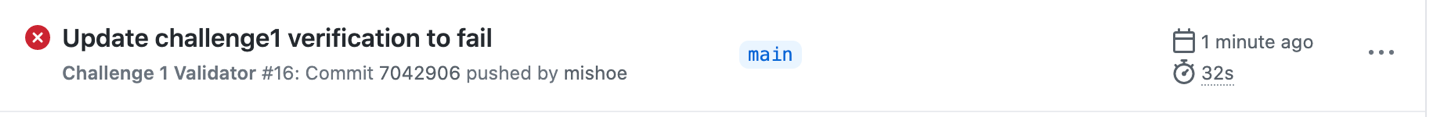
* Navigate to you team’s GitHub Repository
  + Navigate to <https://github.com/TRACK2-DATA-SCIENCE-2021> and click on your team’s repository.
* Click on `Actions` to see recent GitHub Actions that have run on your repository
  + Graphical user interface, text, application

    Description automatically generated
* You will now see all GitHub Actions that have been run on your repository. There will be one GitHub Action that will run with every commit or pull request made to the “main” branch. If you click on that GitHub Action you will see three jobs running, one for each of the different challenges (named “ChallengeN\_validation”, where N is the challenge number).
  + An example of a successful verification for Challenge 1 submission data can be seen below.
    - Note: the check mark notes that it was a successful verification, the GitHub Action job is titled “Challenge1\_validation”, and it was run on the “main” branch

Graphical user interface, application, Teams

Description automatically generated

* An example of an unsuccessful verification for Challenge 1 submission data can be seen below:
  + Note: The red X mark notes that was an unsuccessful verification



* + In order to see more information about the verification issue, click on the failed verification execution
  + By clicking on `build`, and then the red X next to the `execute Challenge N Verification script`, a descriptive message surrounding the issue will appear.
    - In this case, the column “scenario\_id” was misspelled.

Graphical user interface, text

Description automatically generated

## ****Using GitHub Pull Request (PR) Verification****

Pull requests (PR) are supported in GitHub and allow a team member to tell others about changes pushed to a branch in a repository on GitHub. Once a pull request is opened, this allows a discussion and review process to occur surrounding the potential changes with collaborators and add follow-up commits before your changes are merged into the base branch.

Pull requests also allow teams to perform Continuous Integration / Continuous Deployment (CI/CD) tests and validations prior to merging code into the repository’s main branch with CI/CD tools such GitHub Actions or Jenkins.

When a PR has been created with a feature branch that is merging into the “main” branch, the GitHub Actions will be executed upon creation of the PR. Once a PR has been created, you can verify the CI/CD GitHub Action verifications as seen below:

* At the bottom of the PR will be a CI/CD overview, which contains all of the verification results

Graphical user interface, text, application, email

Description automatically generated

* Click `details` beside the verifications that have failed to see the reason of failure and a screen will be brought up in the `Actions` tab that describes the reason that the submission csv did not pass the verification test
* Graphical user interface, text, application

  Description automatically generated