The following documentation is a pictorial guide of how to make and how to review a pull request.

Making a Pull Request

# Select the pull request tab

# 

# Select open new pull request

A screenshot of a computer

Description automatically generated

# Select your branch from the drop down list in “compare”

# A screenshot of a computer Description automatically generated

# Select “create pull request”

# A screenshot of a computer Description automatically generated

# Add title and description for your pull request

Title must include which team you are (DATA SCIENCE or WEBDEV)

Name of what you are working on (This should match what is on your Trello card)

Percentage of completion of your use case from Trello

Description must include reason for pull, e.g. use case completed, Sprint 1 pull request.

Description must include a list of the changes that have been made to your folder since last making a pull request.

# 

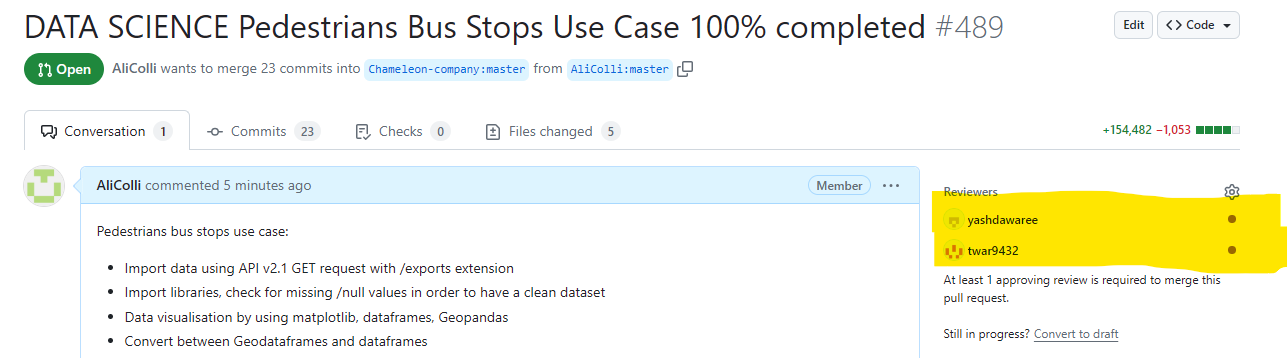
# Select reviewers and choose 2 reviewers

One reviewer can be any member of your team, the other MUST be a leadership student.

You should message students fist to check if they are happy to review for you. You can find a list of GitHub user names to add reviewers on the Melbourne Open Playground team list in Teams.

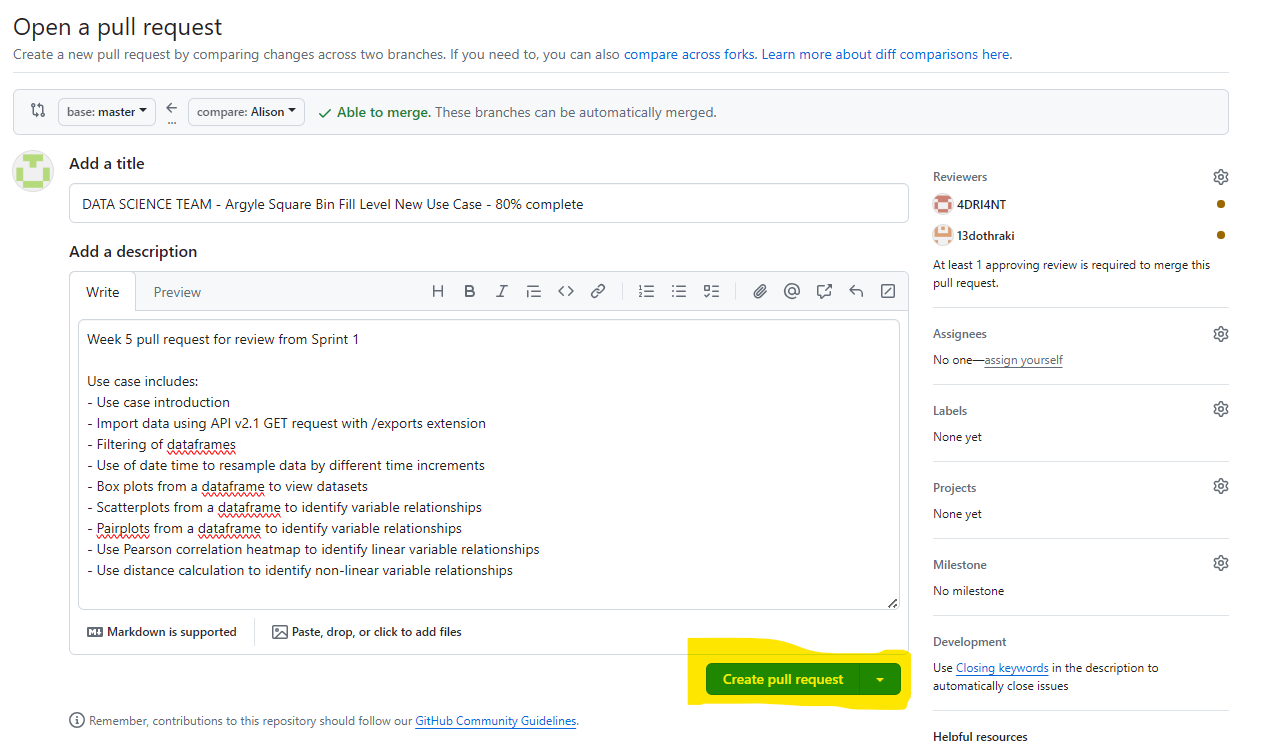
# A screenshot of a computer Description automatically generated

Check you have added two reviewers



# Create your pull request

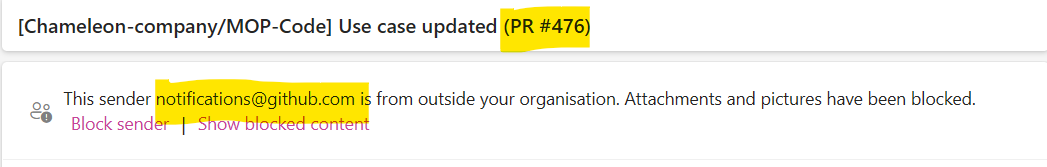
Create your pull request by pressing the green “create pull request” button



**After Making a Pull Request**

# Check your emails

You will be emailed when your pull request receives comments or approvals.



# Check GitHub Conversation for Feedback

After reviewing, your team members will leave you feedback in GitHub.

Open your pull request, and scroll down to the comments section to see this. If you need to make changes you will need to do so in your IDE and then push to GitHub again.

A screenshot of a computer

Description automatically generated

A close up of a computer screen

Description automatically generated

# After making changes

Once you have updated your code and pushed again you will see a new comment automatically generated

A yellow square with black text

Description automatically generated

You will need to also add a comment to email your reviewers so that they know you have updates your work and that it is ready to be re-reviewed.

Be sure to write @ next to their GitHub name so they receive an email notification.

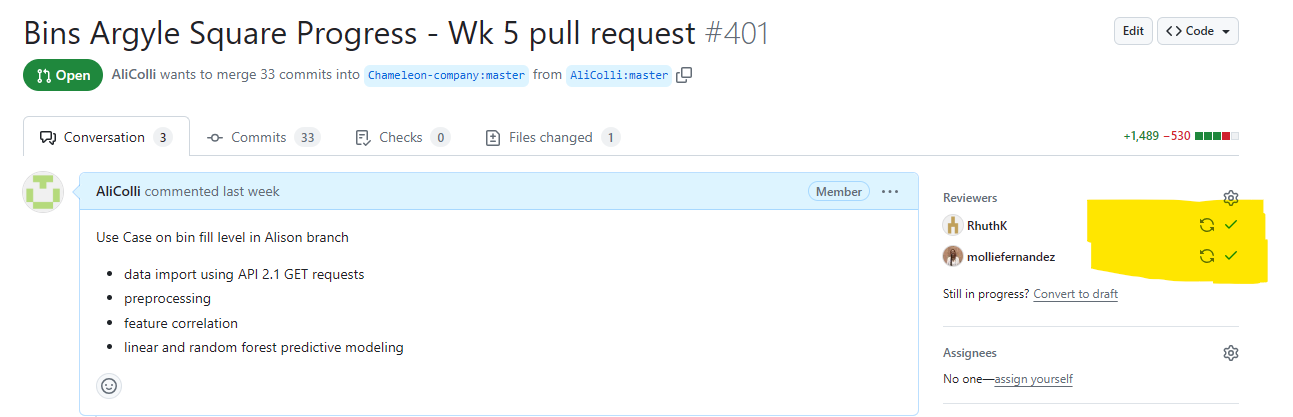
Be sure to press the green “comment” option to keep your pull request open.

A screenshot of a computer

Description automatically generated

# Check review status

Check both reviewers have approved your pull request. Look for green ticks against reviewers names



# Merge pull request

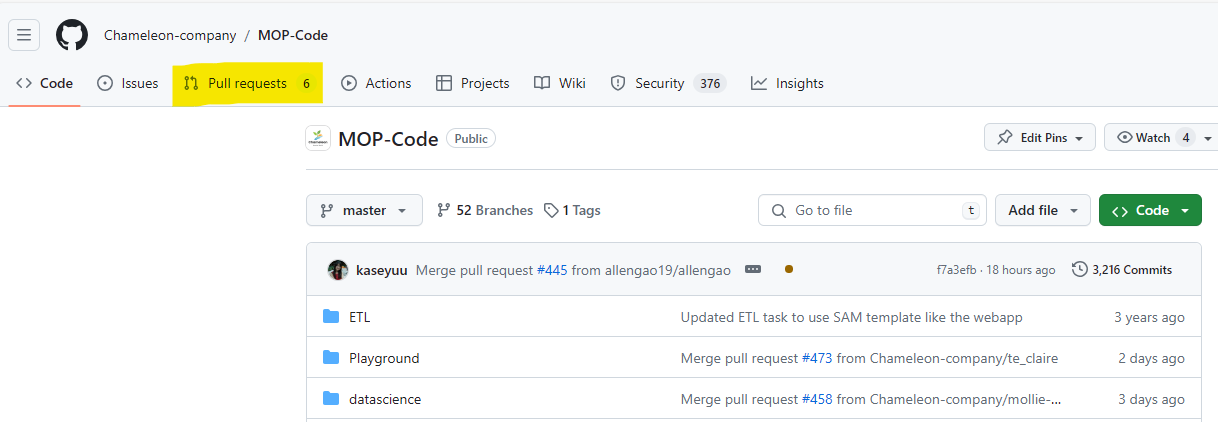
A screenshot of a computer

Description automatically generatedOnce you have two approvals, Merge your pull request!

**Reviewing a Pull request**

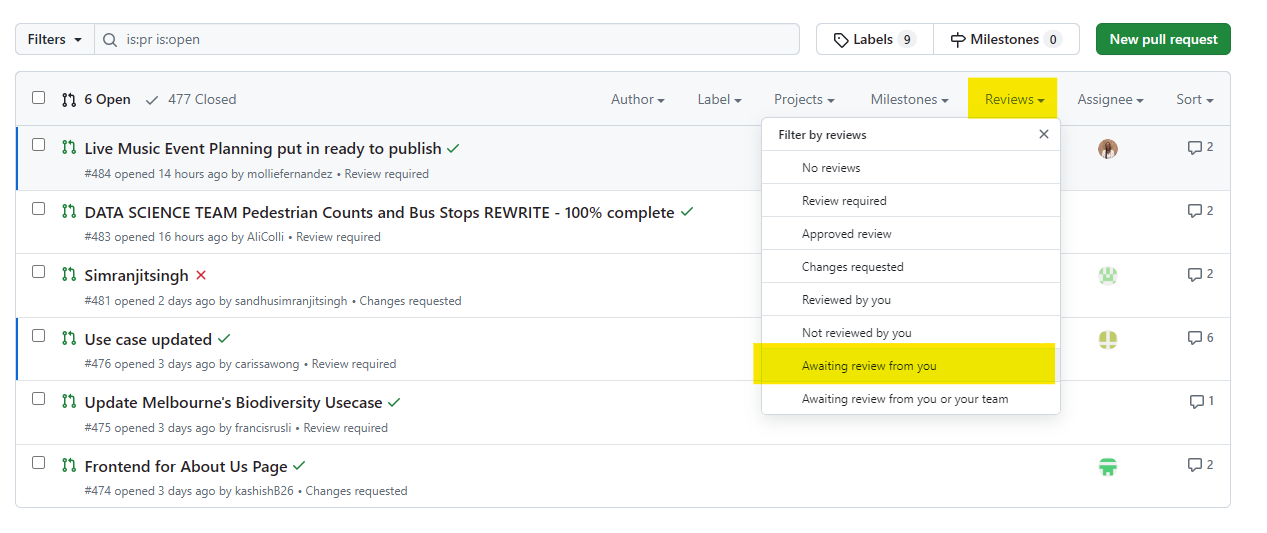
# View open pull requests

On GitHub, select “Pull requests”



# Select a Use Case to review

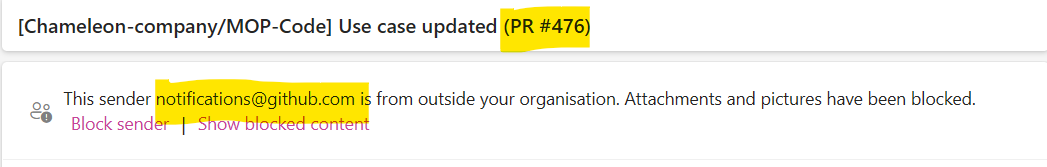
Select the “Reviews” tab and the “Awaiting review from you”



OR

# Check your emails for review requests

Follow the link to the use case from your email



# View the Use Case

When the use case is open, first thing, and the most important thing is to check is whether there is an infinity number of files being changed. If the request is for an infinity number of files to be merged, please do not approve this pull request, and let the requestor know to review this and redo the pull again.

A screenshot of a computer

Description automatically generated



A correct pull should look like below, showing a reasonable number of files changed.

A screenshot of a computer

Description automatically generated

When the use case is open, select the files changed tab. If the use case is large, you will get a load diff alert. In this case, select the three dots on the right hand side and go to view file.

A screenshot of a computer

Description automatically generated

**Pull Request Review Checklist**

This checklist is designed to ensure consistency, quality, and adherence to standards when reviewing GitHub pull requests. Each item below should be reviewed carefully before approving a pull reuest. Use the heading of each checklist item and write down your comments if the use case has met requirements from the checklist.

**Dataset Import**

– The dataset is being accessed via API v2.1 to pull data from the Melbourne Open Data portal.

– API key must NOT be visible

– If an external dataset is used, a csv version of the file must also be uploaded to the repository

[Instruction on API Key](https://github.com/Chameleon-company/MOP-Code/blob/master/datascience/documentation/Instructions%20for%20ODSQL_API_v2.1_2024/Instructions%20for%20ODSQL_API_v2.1_2024.docx)

A screenshot of a computer code

Description automatically generated

**Code Quality & Readability**

– Code must be well-structured and easy to understand

– Comments must have a space and capital letter for first word

– Variable names are meaningful and self-explanatory.

– Appropriate inline comments and documentation are included where necessary.

– Required libraries are imported at the beginning of the notebook.

**Use Case Naming Conventions**

– Use case name must follow the correct standard. Use this below tool and follow the instructions to rename the code file and your use case on MS Planner:

[Use Case Naming Tool](https://github.com/Chameleon-company/MOP-Code/blob/master/datascience/usecases/New_Use_Case_Tool.xlsm)

**Pull Request Location**

– Use case has been submitted under the correct folder (MOP-Code – Playground)

**Language**

– The content is free from major spelling or grammar issues.

– Australian English spelling is used throughout (e.g., “standardisation” instead of “standardization”).

**Visualisation**

– All visualisations must have proper legends, labels and interpretations

**Tutorial Quality**

– The use case is written as a clear, step-by-step guide.

– Insights, libraries, and tools used are clearly described.

– The content is educational and accessible to new users exploring open data.

**Use Case Template**

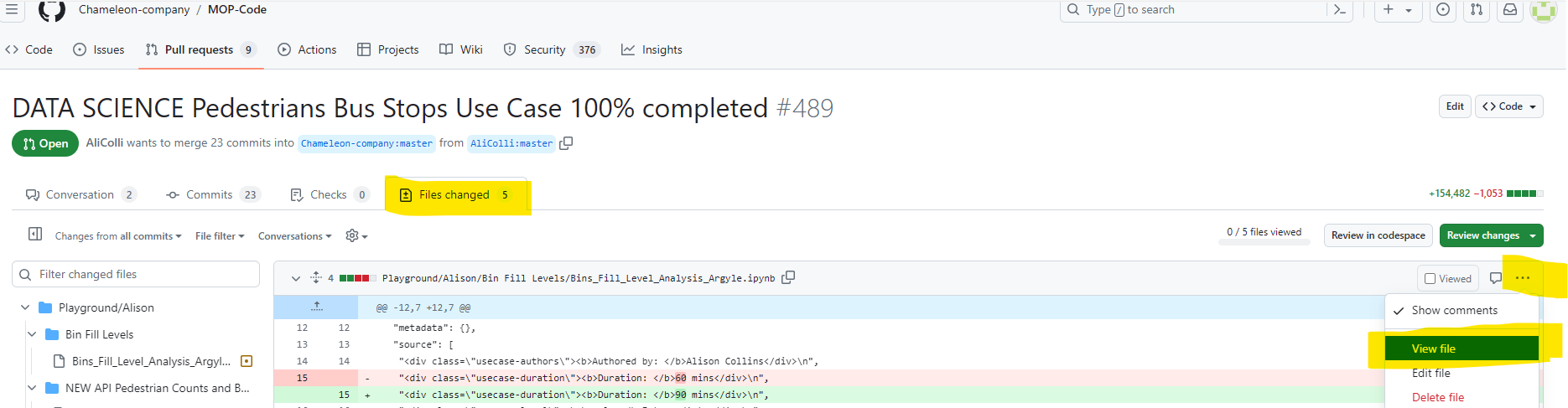
– The correct Use Case Template has been used:

[Use Case Template](https://github.com/Chameleon-company/MOP-Code/blob/master/datascience/usecases/usecase_TEMPLATE.ipynb)

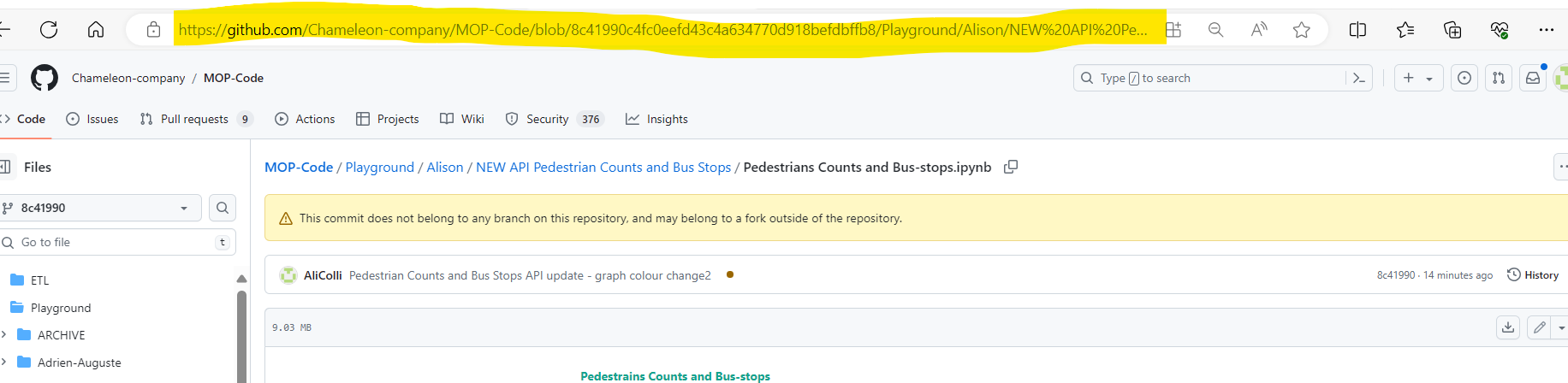
# Troubleshooting viewing:

If the Use Case or parts of it (graphs, tables, folium maps) will not render on GitHub, copy the link from the GitHub and open in <https://nbviewer.org/>

Select the files changed tab, three dots on the right and then view file



Copy the address bar

****

Paste into <https://nbviewer.org/> and press go



# Review statuses:

Post your review of the pull request in one of these 3 statuses:

# Comment: Submit general feedback without explicitly approving the changes or requesting additional changes.

# Approve: Submit feedback and approve merging the changes proposed in the pull request.

# Request changes: Submit feedback that must be addressed before the pull request can be merged.

## Further reading

Capstone Pull Requests: [5. Navigating Pull Requests on GitHub | Capstone (verdant-raindrop-f3e404.netlify.app)](https://verdant-raindrop-f3e404.netlify.app/capstonetraining/github/5-pull-requests/)

Capstone Reviewing Pull Requests: [6. Reviewing Pull Requests | Capstone (verdant-raindrop-f3e404.netlify.app)](https://verdant-raindrop-f3e404.netlify.app/capstonetraining/github/6-rewviewing-pull-requests/).

## Author

Alison Collins, 2024

Katrine Chan, 2024 v2

Jenny Nguyen, 2025 v3