Project Intermediate Deliverable Template

As a Boilermaker pursuing academic excellence, we pledge to be honest and true in all that we do. Accountable together – We are Purdue.

(On group submissions, have each team member type their name).

Your names: Parker Bushey, Katie Roberts, Ashwin Senthilkumar

Today's date: 11/15/2021

Assignment Goal

This is a "Deliverable" milestone. You must demonstrate your team's automated test and deployment pipeline, as well as one or more pieces of functionality as you indicated in your initial plan.

Relevant Course Outcomes

A student who successfully completes this assignment will have demonstrated the ability to

- Outcome i:
    - Identify and follow an appropriate software engineering process for this context.
- Outcome iii:
    - Experience social aspects of software engineering (communication, teamwork).

Resources

One or more intermediate deliverables is a normal component of many software engineering efforts. Such a deliverable gives you the opportunity to demonstrate preliminary functionality to a customer and get feedback. The specific tempo depends on the kind of product, the relationship with the customer, and the team's degree of experience with "continuous" development processes (Continuous Integration, Continuous Test, Continuous Deployment). At the extreme, some teams demo on a biweekly tempo aligned with their sprints, and even more frequently if they are following the "eXtreme Programming" practice of embedding a customer representative on the team (for obvious logistical reasons, this practice is rare in industry).

Assignment

Working functionality

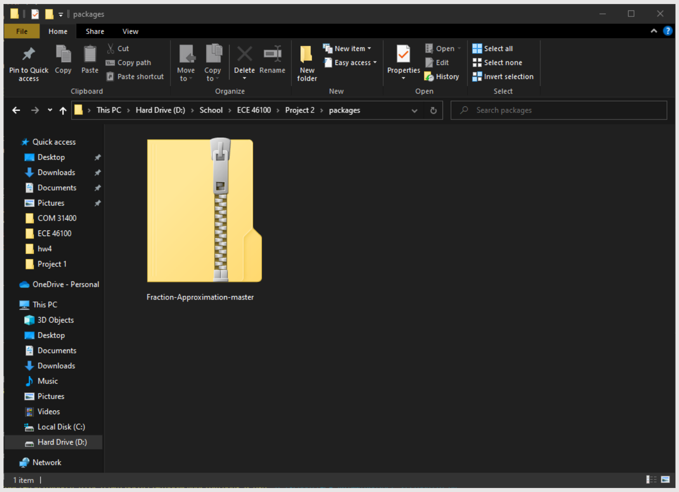
Provide prose and screenshot(s) demonstrating all elements of working functionality that were promised by your team in Milestone 1. For example, perhaps you've implemented the "*upload package*" feature, and can show a *wget* request or Python client interacting with this feature as deployed on a GCP resource or a local machine.

(To receive any credit on this part, you must demonstrate at least one working element.)
(Don't forget to describe any *planned technical debt*, i.e. aspects of the feature whose implementation you deferred in order to get a working demo. That technical debt should end up in your team's backlog.)

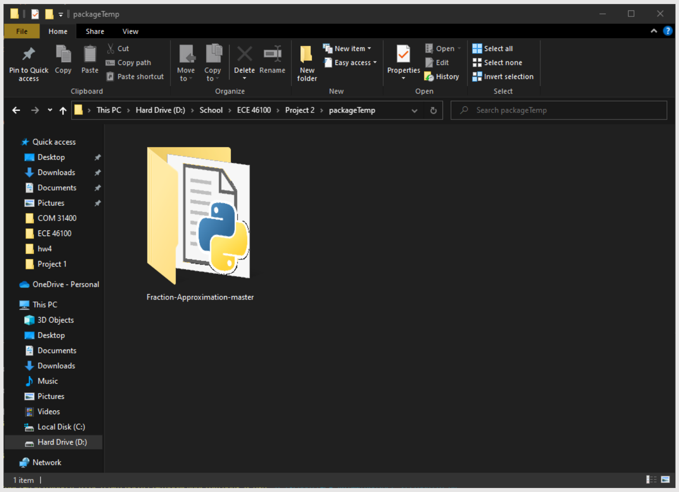
Completed Feature 1:
The zip and unzip element of the project is completed and working. The class can accept a zipped file and unzip it into a temporary directory. It can also rezip a file that is stored in the

temporary directory. After it has been unzipped, the given package from project 1 can be used to test the contents of the files stored in the temporary directory.

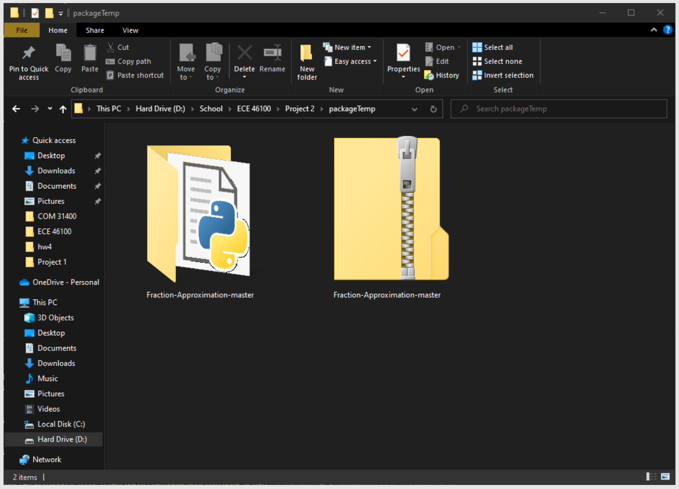
Completed Feature 1 Screenshots:
For zipping and unzipping the program accepts a zip or tar archive. This is the archive that will be used for this demo.



This is the temporary holding folder after the class has unzipped the archive. The contents of the archive are shown here.



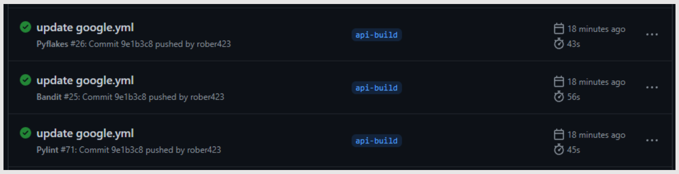
This is the temporary holding folder after the class has archived the contents again. The created archive is shown here.



For the future, a cleaning method has also been included which removes these contents from this folder so the system does not build up too much data.

Completed Feature 2:
The continuous integration testing using GitHub Actions has been implemented. It is currently running tests on the implementation given for project 1 and will soon be expanded to test the rest of the system. Three independent testers run each push and their output is used to determine if any major issues exist within the system.

Completed Feature 2 Screenshots:
Shown here is the successful screening of a recent push made to the project repository. Each testing software runs from its own YAML file and gives its report after completion.



update google.yml
Pyflakes #26: Commit 9e1b3c8 pushed by rober423
api-build
18 minutes ago
43s

update google.yml
Bandit #25: Commit 9e1b3c8 pushed by rober423
api-build
18 minutes ago
56s

update google.yml
Pylint #71: Commit 9e1b3c8 pushed by rober423
api-build
18 minutes ago
45s

---

feature 3: see CD below

For each incomplete element, explain why you are behind schedule.

Incomplete Feature 1: Explanation and progress to date

API will not deploy because it thinks the api.yaml file is not in the correct format, and the flask_restful module will not import

the base functionality of the api is all ready (objects with get/post etc methods), we just need to put in the calls to zip/unzip and ingest

---

Incomplete Feature 2: …

CI/CD via GitHub Actions

Provide prose and screenshot(s) demonstrating that your team is using GitHub actions to facilitate continuous integration (e.g. by running a *linter* and a *test suite* on every pull request).

What steps is your team following prior to accepting a code change? (e.g. git-hooks, code review, linting, test suite, etc.)
Upon each push to the GitHub repository, three independent code testers are run on the Python code to ensure there are no outstanding problems with its implementation. These three testers are Pylint, Pyflakes, and Bandit. The output should be checked for any recommended fixes with urgent status before accepting code changes.

Provide a link to an example in your GitHub repo where your team followed this CI process (e.g. a pull request):
https://github.com/Purdue-ECE-461/project-2-3/commit/8375c16a8c832c35e14fe648f7cea980dd6ff9ea

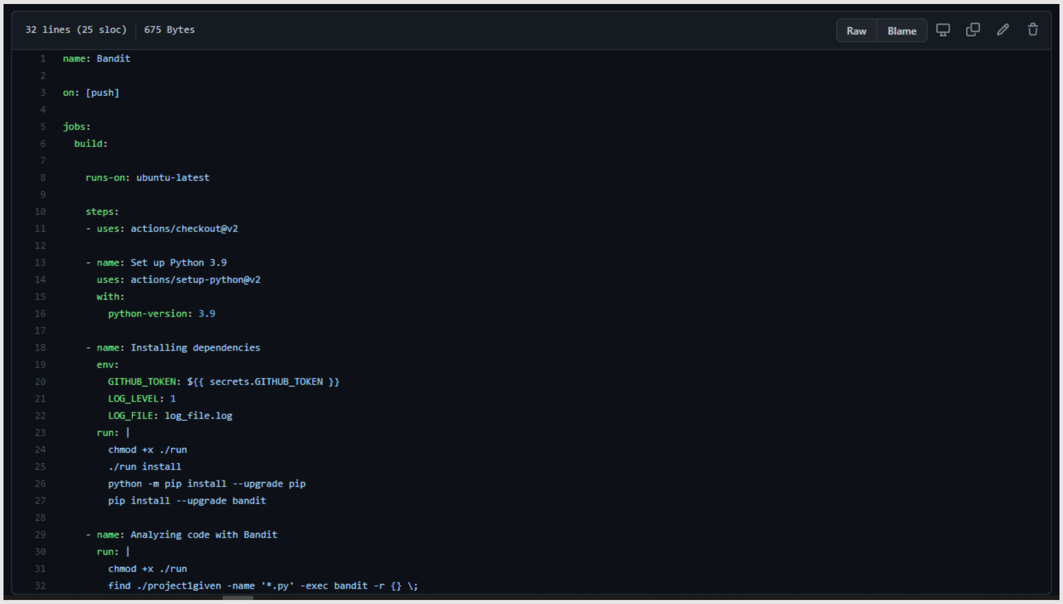
How consistent have you been with this process? What is keeping you from full consistency?
We have not been adding very much software recently, but when we begin to commit more heavily the process will be respected. As of now, there haven’t been any major flags raised by the software testers, so there hasn’t been any need to reject a commit, but when they do we will be sure to reject it until the file in question has been fixed. In general, we have not been using the CI process to its fullest potential, but plans have been made to make it a priority.

What aspects of your system are being tested automatically by your CI scheme?
Currently it is testing the implementation we were given from project 1. Any changes being made to this are tested automatically upon push. We plan to test the entirety of our system in the future however, the organization of the repository has not been finalized.

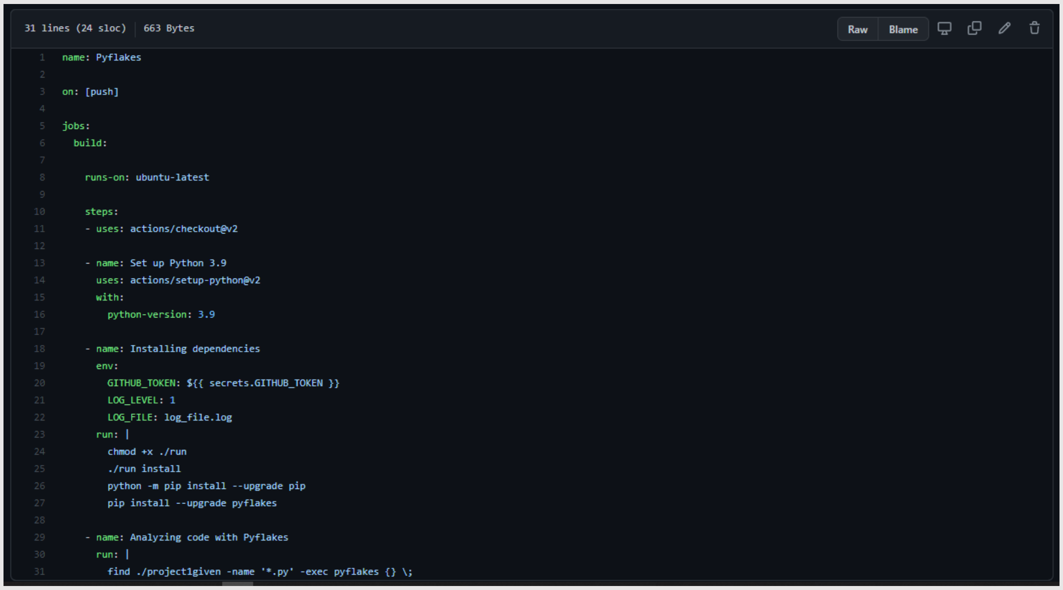
What kinds of defects might go uncaught, and how are you mitigating this risk?
The testers that we are using were chosen to try and address most kinds of defects. However there are still some things that might not be caught such as poor commenting or poor code

quality. To help with this, we are doing our best to think about maintainability when pushing code. If the code is not maintainable, it should be reworked in order to fix that.

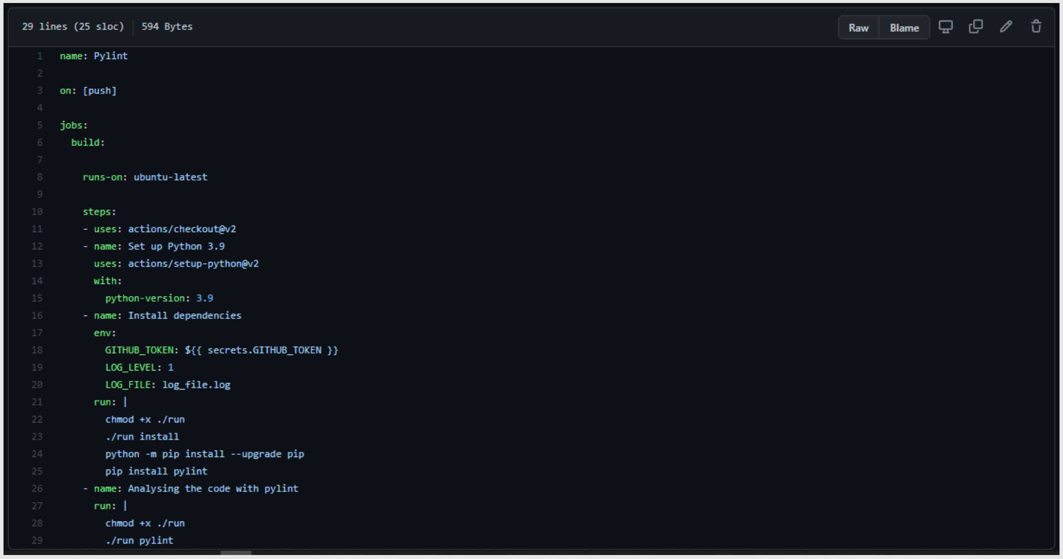
Provide screenshots of the GitHub action file (e.g. YAML) that defines the CI stages



```yaml
name: Bandit

on: [push]

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v2

    - name: Set up Python 3.9
      uses: actions/setup-python@v2
      with:
        python-version: 3.9

    - name: Installing dependencies
      env:
        GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
        LOG_LEVEL: 1
        LOG_FILE: log_file.log
      run: |
        chmod +x ./run
        ./run install
        python -m pip install --upgrade pip
        pip install --upgrade bandit

    - name: Analyzing code with Bandit
      run: |
        chmod +x ./run
        find ./project1given -name '*.py' -exec bandit -r {} \;
```

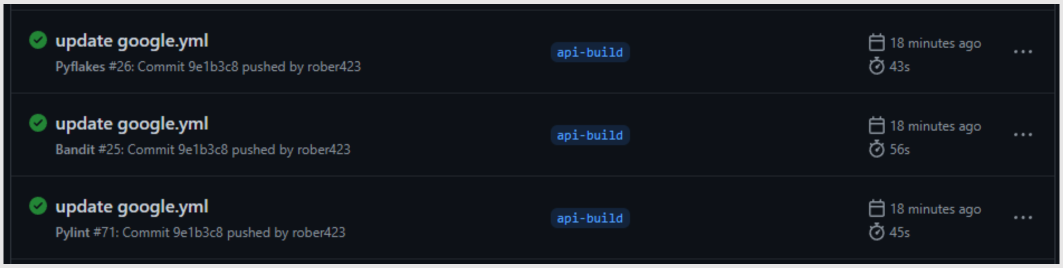


```yaml
name: Pyflakes

on: [push]

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v2

    - name: Set up Python 3.9
      uses: actions/setup-python@v2
      with:
        python-version: 3.9

    - name: Installing dependencies
      env:
        GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
        LOG_LEVEL: 1
        LOG_FILE: log_file.log
      run: |
        chmod +x ./run
        ./run install
        python -m pip install --upgrade pip
        pip install --upgrade pyflakes

    - name: Analyzing code with Pyflakes
      run: |
        find ./project1given -name '*.py' -exec pyflakes {} \;
```



```yaml
name: Pylint

on: [push]

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v2
    - name: Set up Python 3.9
      uses: actions/setup-python@v2
      with:
        python-version: 3.9
    - name: Install dependencies
      env:
        GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
        LOG_LEVEL: 1
        LOG_FILE: log_file.log
      run: |
        chmod +x ./run
        ./run install
        python -m pip install --upgrade pip
        pip install pylint
    - name: Analysing the code with pylint
      run: |
        chmod +x ./run
        ./run pylint
```

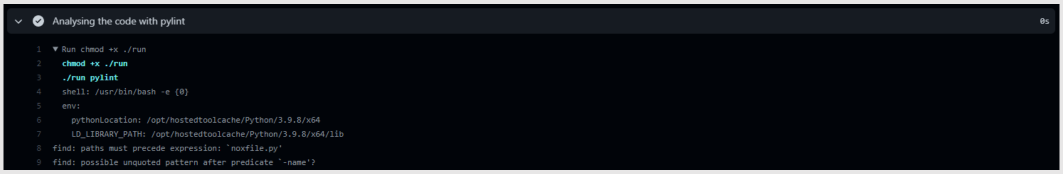
Provide screenshot(s) of the test suite in action, e.g. the reports from the various tools you have configured, as run on one of your team's code changes.



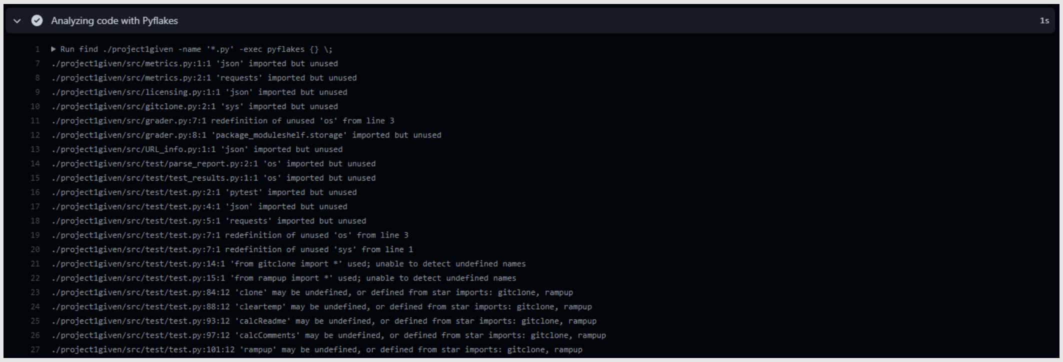
update google.yml
Pyflakes #26: Commit 9e1b3c8 pushed by rober423
api-build
18 minutes ago
43s

update google.yml
Bandit #25: Commit 9e1b3c8 pushed by rober423
api-build
18 minutes ago
56s

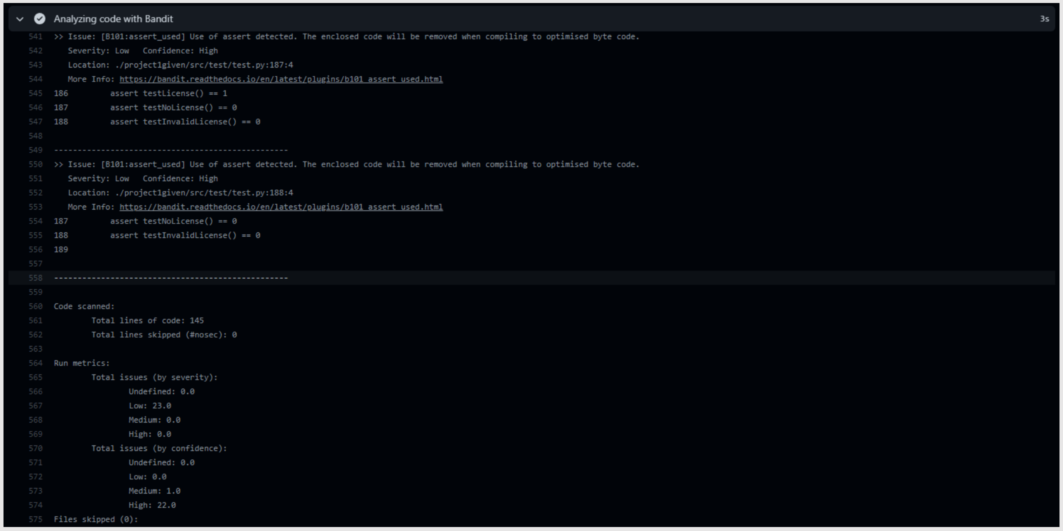
update google.yml
Pylint #71: Commit 9e1b3c8 pushed by rober423
api-build
18 minutes ago
45s



```
▼ Run chmod +x ./run
  chmod +x ./run
  ./run pylint
  shell: /usr/bin/bash -e {0}
  env:
    pythonLocation: /opt/hostedtoolcache/Python/3.9.8/x64
    LD_LIBRARY_PATH: /opt/hostedtoolcache/Python/3.9.8/x64/lib
find: paths must precede expression: `noxfile.py'
find: possible unquoted pattern after predicate `-name'?
```



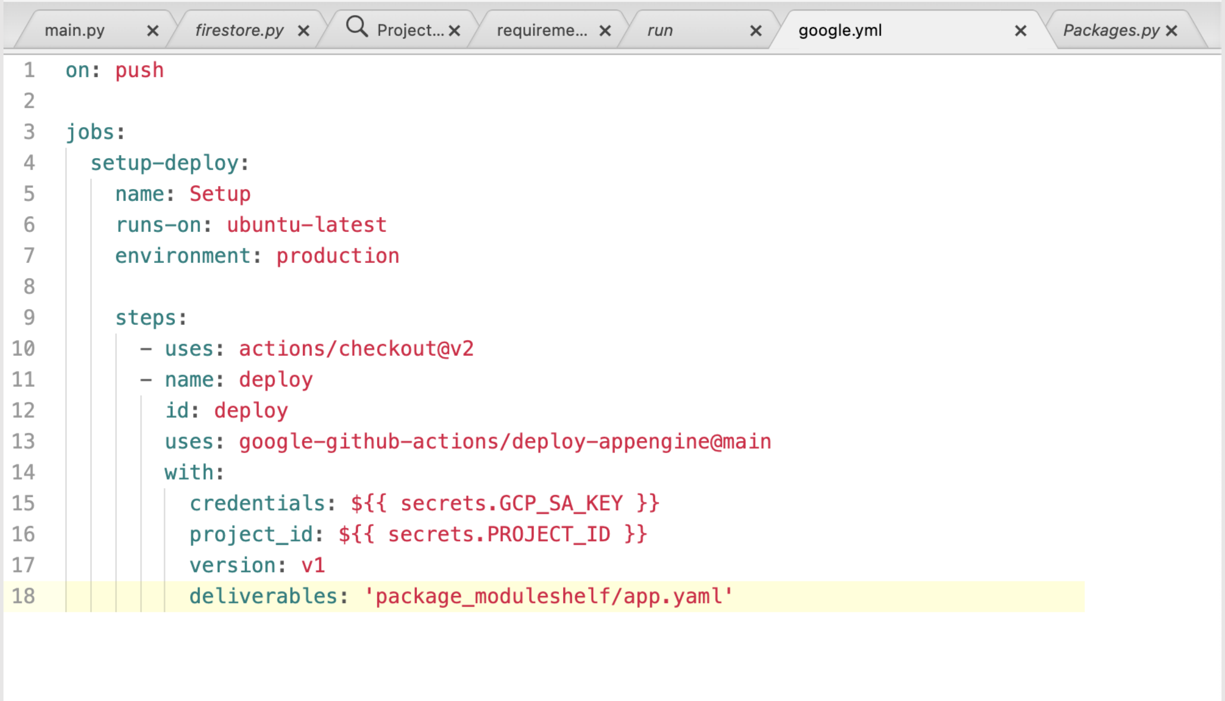
```
▶ Run find ./project1given -name '*.py' -exec pyflakes {} \;
./project1given/src/metrics.py:1:1 'json' imported but unused
./project1given/src/metrics.py:2:1 'requests' imported but unused
./project1given/src/licensing.py:1:1 'json' imported but unused
./project1given/src/gitclone.py:2:1 'sys' imported but unused
./project1given/src/grader.py:7:1 redefinition of unused 'os' from line 3
./project1given/src/grader.py:8:1 'package_moduleshelf.storage' imported but unused
./project1given/src/URL_info.py:1:1 'json' imported but unused
./project1given/src/test/parse_report.py:2:1 'os' imported but unused
./project1given/src/test/test_results.py:1:1 'os' imported but unused
./project1given/src/test/test.py:2:1 'pytest' imported but unused
./project1given/src/test/test.py:4:1 'json' imported but unused
./project1given/src/test/test.py:5:1 'requests' imported but unused
./project1given/src/test/test.py:7:1 redefinition of unused 'os' from line 3
./project1given/src/test/test.py:7:1 redefinition of unused 'sys' from line 1
./project1given/src/test/test.py:14:1 'from gitclone import *' used; unable to detect undefined names
./project1given/src/test/test.py:15:1 'from rampup import *' used; unable to detect undefined names
./project1given/src/test/test.py:84:12 'clone' may be undefined, or defined from star imports: gitclone, rampup
./project1given/src/test/test.py:88:12 'cleartemp' may be undefined, or defined from star imports: gitclone, rampup
./project1given/src/test/test.py:93:12 'calcReadme' may be undefined, or defined from star imports: gitclone, rampup
./project1given/src/test/test.py:97:12 'calcComments' may be undefined, or defined from star imports: gitclone, rampup
./project1given/src/test/test.py:101:12 'rampup' may be undefined, or defined from star imports: gitclone, rampup
```



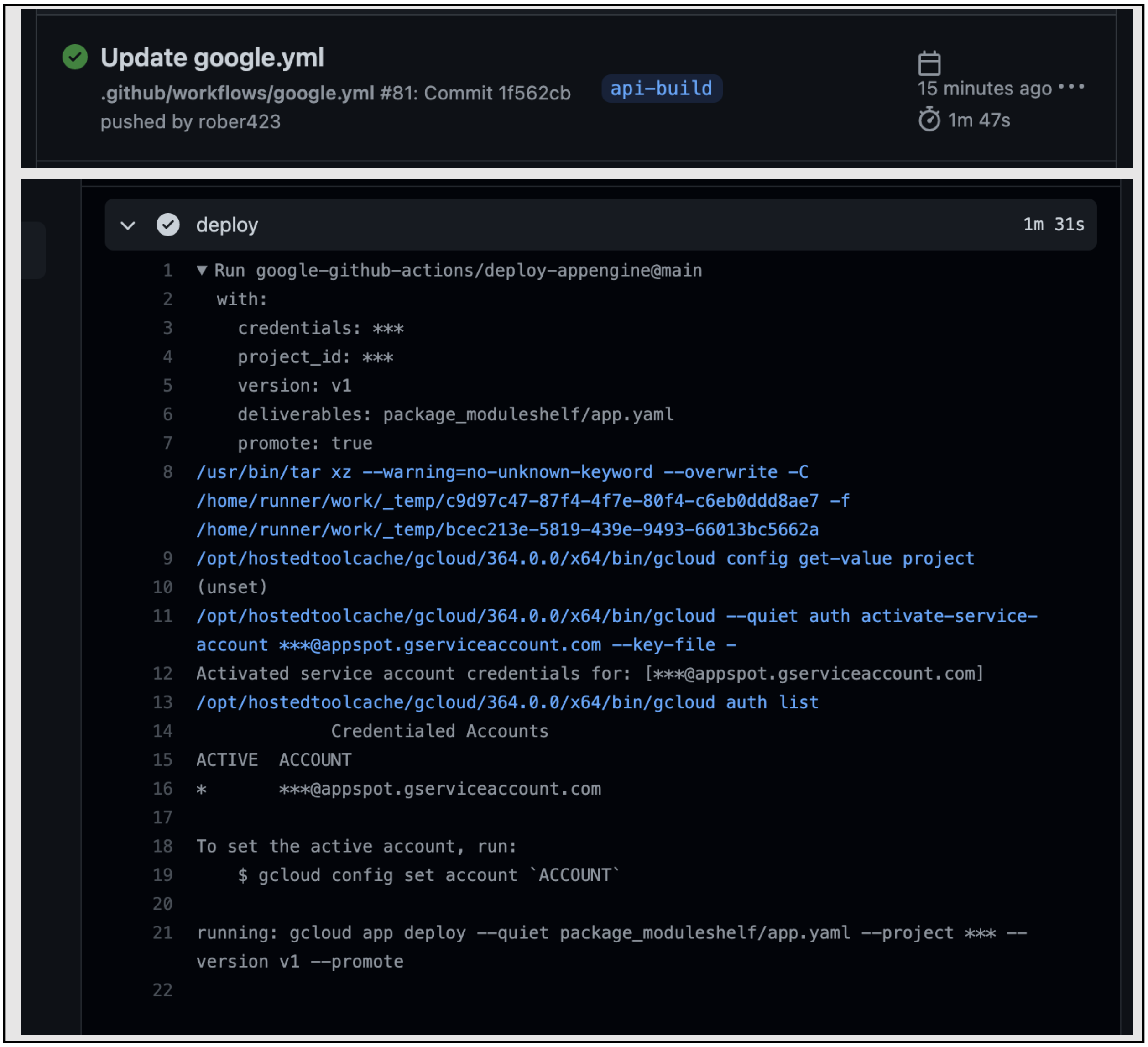
```
>> Issue: [B101:assert_used] Use of assert detected. The enclosed code will be removed when compiling to optimised byte code.
   Severity: Low   Confidence: High
   Location: ./project1given/src/test/test.py:187:4
   More Info: https://bandit.readthedocs.io/en/latest/plugins/b101_assert_used.html
186         assert testLicense() == 1
187         assert testNoLicense() == 0
188         assert testInvalidLicense() == 0

--------------------------------------------------
>> Issue: [B101:assert_used] Use of assert detected. The enclosed code will be removed when compiling to optimised byte code.
   Severity: Low   Confidence: High
   Location: ./project1given/src/test/test.py:188:4
   More Info: https://bandit.readthedocs.io/en/latest/plugins/b101_assert_used.html
187         assert testNoLicense() == 0
188         assert testInvalidLicense() == 0
189

--------------------------------------------------

Code scanned:
        Total lines of code: 145
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0.0
                Low: 23.0
                Medium: 0.0
                High: 0.0
        Total issues (by confidence):
                Undefined: 0.0
                Low: 0.0
                Medium: 1.0
                High: 22.0
Files skipped (0):
```

Describe the extent to which you are able to "continuously deploy". What is your team's process to get your current prototype into a deployment on GCP?

We are able to continuously deploy our app via our google.yml workflow, which deploys the app to the cloud App Engine after a push request.



```yaml
on: push

jobs:
  setup-deploy:
    name: Setup
    runs-on: ubuntu-latest
    environment: production

    steps:
      - uses: actions/checkout@v2
      - name: deploy
        id: deploy
        uses: google-github-actions/deploy-appengine@main
        with:
          credentials: ${{ secrets.GCP_SA_KEY }}
          project_id: ${{ secrets.PROJECT_ID }}
          version: v1
          deliverables: 'package_moduleshelf/app.yaml'
```



Update google.yml

.github/workflows/google.yml #81: Commit 1f562cb pushed by rober423

```
▼ Run google-github-actions/deploy-appengine@main
  with:
    credentials: ***
    project_id: ***
    version: v1
    deliverables: package_moduleshelf/app.yaml
    promote: true
/usr/bin/tar xz --warning=no-unknown-keyword --overwrite -C
/home/runner/work/_temp/c9d97c47-87f4-4f7e-80f4-c6eb0ddd8ae7 -f
/home/runner/work/_temp/bcec213e-5819-439e-9493-66013bc5662a
/opt/hostedtoolcache/gcloud/364.0.0/x64/bin/gcloud config get-value project
(unset)
/opt/hostedtoolcache/gcloud/364.0.0/x64/bin/gcloud --quiet auth activate-service-
account ***@appspot.gserviceaccount.com --key-file -
Activated service account credentials for: [***@appspot.gserviceaccount.com]
/opt/hostedtoolcache/gcloud/364.0.0/x64/bin/gcloud auth list
             Credentialed Accounts
ACTIVE  ACCOUNT
*       ***@appspot.gserviceaccount.com

To set the active account, run:
    $ gcloud config set account `ACCOUNT`

running: gcloud app deploy --quiet package_moduleshelf/app.yaml --project *** --
version v1 --promote

```

Security requirement

The customer requires that you conduct a STRIDE analysis (link to a notable writeup of this process). I have advised you to not defer this until the end of the project. Discuss your efforts in this direction so far, and any changes or adaptations you've made in your design as a result of your analysis.

Design changes

This week there were no major design changes. Everything is proceeding according to the design document or any changes made in previous milestones. Some of us fell slightly behind on this milestone so we will be spending this week catching up.

Teammate contributions

Team member	Task(s) worked on	Task(s) completed	Hours spent
Parker Bushey	- GitHub Actions continuous integration	- GitHub Actions continuous integration	- 6 hours
Ashwin Senthilkumar	- Ingesting - Updating	Ingestion in progress (pushed back due to interviews/exams)	- 3 hours
Katie Roberts	- api template - fixing CD action	- api template - CD action	- 12 hours

Progress relative to plan

Task targeted for this week	Hours for this task: Planned and Actual Actual (Planned)	Is it complete? (If yes, how did you confirm?)	If incomplete: Estimated hours of remaining work
GitHub Actions continuous integration	12 (15)	Yes, tests from each software tester successfully run on the software delegated to them at this time.	N/A
Updating/Ingestion	3 (5)	No, it will be complete by next milestone	2-3 hours
version parsing	10 (0)	no	3 hours

Are you on, ahead of, or behind schedule? If behind, what is your mitigation plan?

Ashwin: I am currently behind schedule due to an extremely busy month with interviews/exams every week. I will be spending this week and Thanksgiving catching up on my part. My mitigation plan will be to attend office hours and get some guidance tomorrow after class (Tuesday).

Parker: This week I have technically been on schedule, however I wished to have done more than I actually did. This next week will be more focused on getting ahead when possible instead of completing only what is required. The milestones we have defined will help me organize my time for the next few weeks.