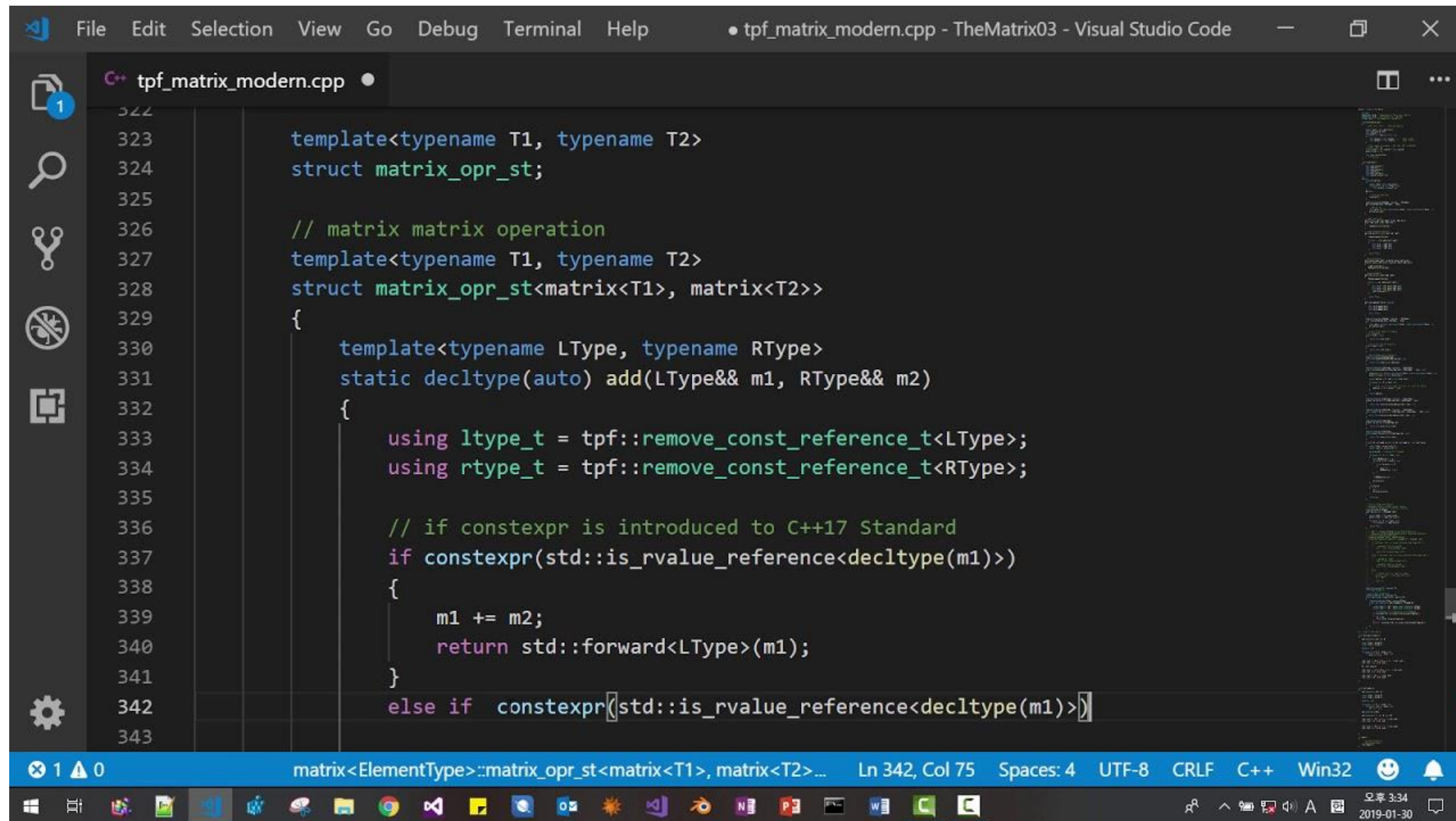


The Missing Semester: Metaprogramming



```
File Edit Selection View Go Debug Terminal Help • tpf_matrix_modern.cpp - TheMatrix03 - Visual Studio Code

tpf_matrix_modern.cpp
323 template<typename T1, typename T2>
324 struct matrix_opr_st;
325
326 // matrix matrix operation
327 template<typename T1, typename T2>
328 struct matrix_opr_st<matrix<T1>, matrix<T2>>
329 {
330     template<typename LType, typename RType>
331     static decltype(auto) add(LType&& m1, RType&& m2)
332     {
333         using ltype_t = tpf::remove_const_reference_t<LType>;
334         using rtype_t = tpf::remove_const_reference_t<RType>;
335
336         // if constexpr is introduced to C++17 Standard
337         if constexpr(std::is_rvalue_reference<decltype(m1)>>)
338         {
339             m1 += m2;
340             return std::forward<LType>(m1);
341         }
342         else if constexpr(std::is_rvalue_reference<decltype(m1)>>)
343
```

matrix<ElementType>::matrix_opr_st<matrix<T1>, matrix<T2>... Ln 342, Col 75 Spaces: 4 UTF-8 CRLF C++ Win32 2019-01-30



What is Metaprogramming?

- Imagine a guy who builds cars. Say it's the same thing as using a computer.
- At some point he realizes he's always doing the same thing, more or less. So he builds factories to build cars, and it's much better.
- He's now programming!
- Nevertheless, once again, at some point, he realizes he's always doing the same thing, to some extent. Now he decides to build factories that build factories that build cars.
- That's metaprogramming.

Simply put , Metaprogramming is the practice of writing code that writes code for us .



Sri Venkateswara
College of Engineering



The Missing Semester:

An Introduction to CI/CD with Travis CI and Python



Sri Venkateswara
College of Engineering



Part I - Choosing a continuous integration service

What to consider when creating a Python3 CI/CD pipeline :

- **Python Community on GitHub** - support for all python3 🐍 versions.
- **Choose your testing framework** - Use it with any popular testing framework: pytest, unittest, etc.
- **Open vs. closed source** - whenever possible choose an open source option. The below providers generally offer zero cost options for new open source projects with reasonable build 🔧 constraints. If you go closed source, you're going to need to fork over some 💰.



Sri Venkateswara
College of Engineering



Part I - Choosing a continuous integration service

Common Providers

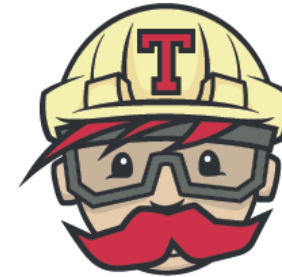
- **Travis CI**: The one used in today's session
- **Circle CI**: Seems to have a little saner syntax and finer configuration relative to Travis CI
- **GitLab**: Leans open source where the others are closed source
- **GitHub Actions**: Newer, native to GitHub, and offers Linux, Windows, and MacOS builds.



Part II - Prerequisites

You should have basic knowledge about

- GitHub
- Python
- Testing frameworks
- How to use the Terminal.



Travis CI

Knowledge of Pipenv (Pipfiles) and pytest is a plus, but not required. CI/CD tools are a great way to learn both.

YAML Basics

Most CI services interact with your repositories using a YAML configuration file.

YAML - [YAML Syntax, from Red Hat Ansible](#)



Sri Venkateswara
College of Engineering



Sample Travis CI YAML Program

```
# .travis.yml

dist: xenial

language: python

cache: pip

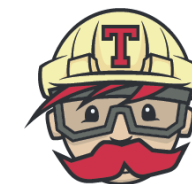
python:
  - "3.6"
  - "3.7"
  - "3.8"
  - "nightly"

matrix:
  allow_failures:
    - python: "nightly"

install:
  - pip install pipenv --upgrade-strategy=only-if-needed
  - pipenv install --dev

script:
  - bash scripts/test.sh

after_script:
  - bash <(curl -s https://codecov.io/bash)
```



Travis CI



Sri Venkateswara
College of Engineering



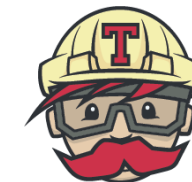
CI/CD SETUP

STEP I - Get setup to install my python package project template

Ensure you have a compatible python 🐍 environment on your computer.

```
python3 --version  
python3 -m pip --version  
python3 -m pytest --version
```

If you are missing any of the above, you can add the packages to your main python install.



Travis CI



Sri Venkateswara
College of Engineering

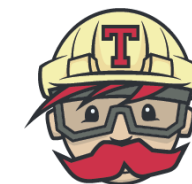


Step I - Get setup to install my python package project template

```
sudo apt-get update  
sudo apt-get install python3-pip  
sudo apt-get install python3-pytest
```

Now that we have confirmed your base python setup, let's go ahead and ensure the pipenv and cookiecutter python packages are installed.

```
python3 -m pip install --user pipenv  
python3 -m pip install --user cookiecutter
```



Travis CI



Sri Venkateswara
College of Engineering



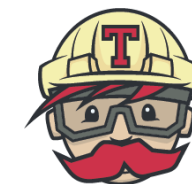
Step II – Create a Python project and setup the Pipenv virtual environment

```
# make sure your path finds --user installs

## add `export PATH="$HOME/.local/bin:$PATH"`
## to your ~/.bashrc, ~/.zshrc file on linux

cookiecutter https://github.com/iancleary/pypackage
# For the sake of brevity, repos on GitHub can just use the 'gh' pre
cookiecutter gh:iancleary/pypackage
```

- Use `cd new-directory` to change into the new directory you just created (replace new-directory)
- Run `pipenv` to display it's options. Take a look at what's offered.
- Next, run `pipenv install --dev` to install the production and development dependencies specified in the Pipfile.
- Run `pipenv shell` to load the virtual environment.



Travis CI



Sri Venkateswara
College of Engineering



Step III- Run the tests locally and make sure they pass

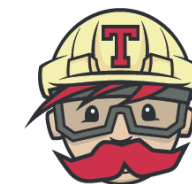
1. Execute `./scripts/tests.sh` from within the directory's Pipenv.

This command executes a bash script that does several things:

- runs pytest to check test cases and check test coverage
- checks formatting with the black package
- runs static type checking against your code base with mypy
- checks import sorting from standard lib, your application, and custom packages .

2. Push the directory to a remote git repo

- Git website: [Adding an existing project to GitHub using the command line](#)



Travis CI



Sri Venkateswara
College of Engineering

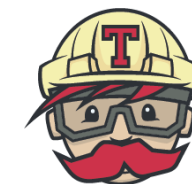


Step IV – Connect Travis-CI to the repo

Head over to [Travis CI](https://travis-ci.com/) and sign in with your GitHub account:

- <https://help.github.com/en/enterprise/2.16/admin/developer-workflow/continuous-integration-using-travis-ci>

Once you login to Travis CI and Enable GitHub Access, you'll want to configure Travis from your GitHub Settings page.



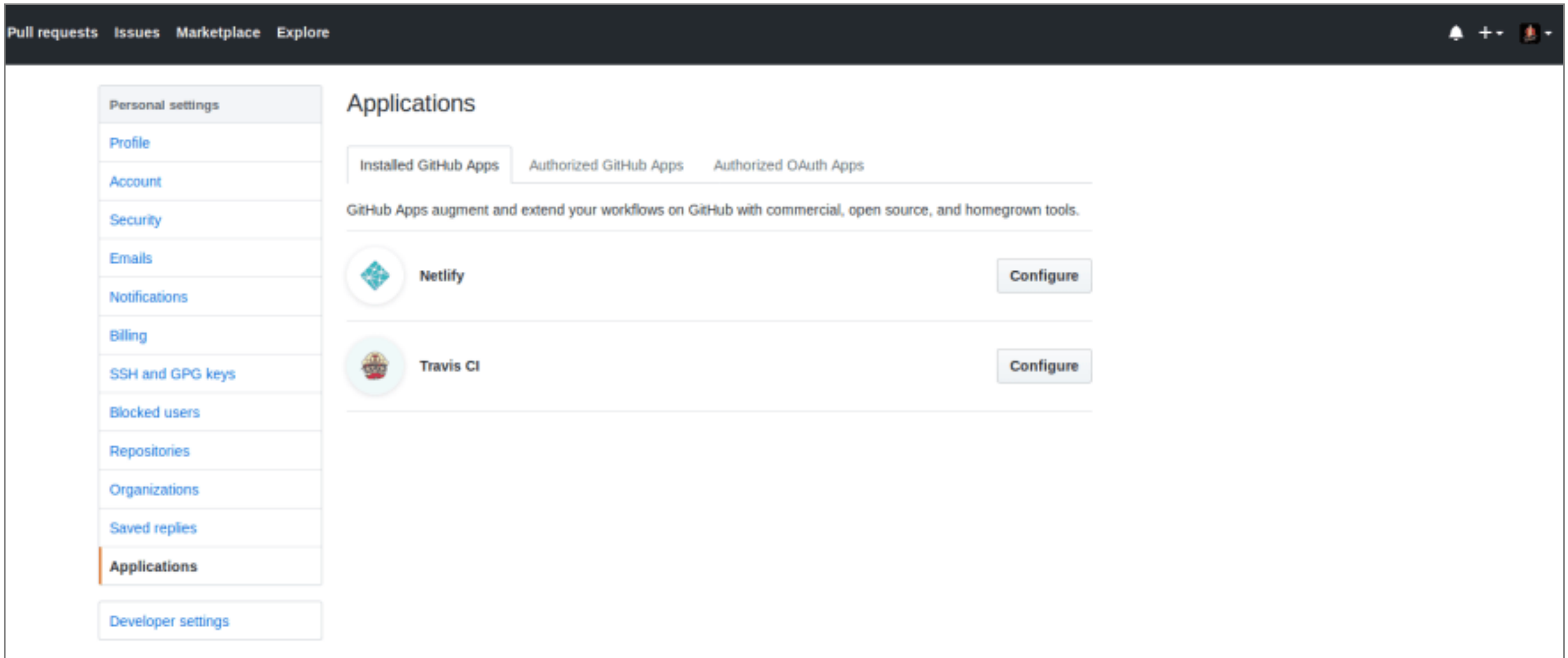
Travis CI



Sri Venkateswara
College of Engineering



Step IV – Connect Travis-CI to the repo



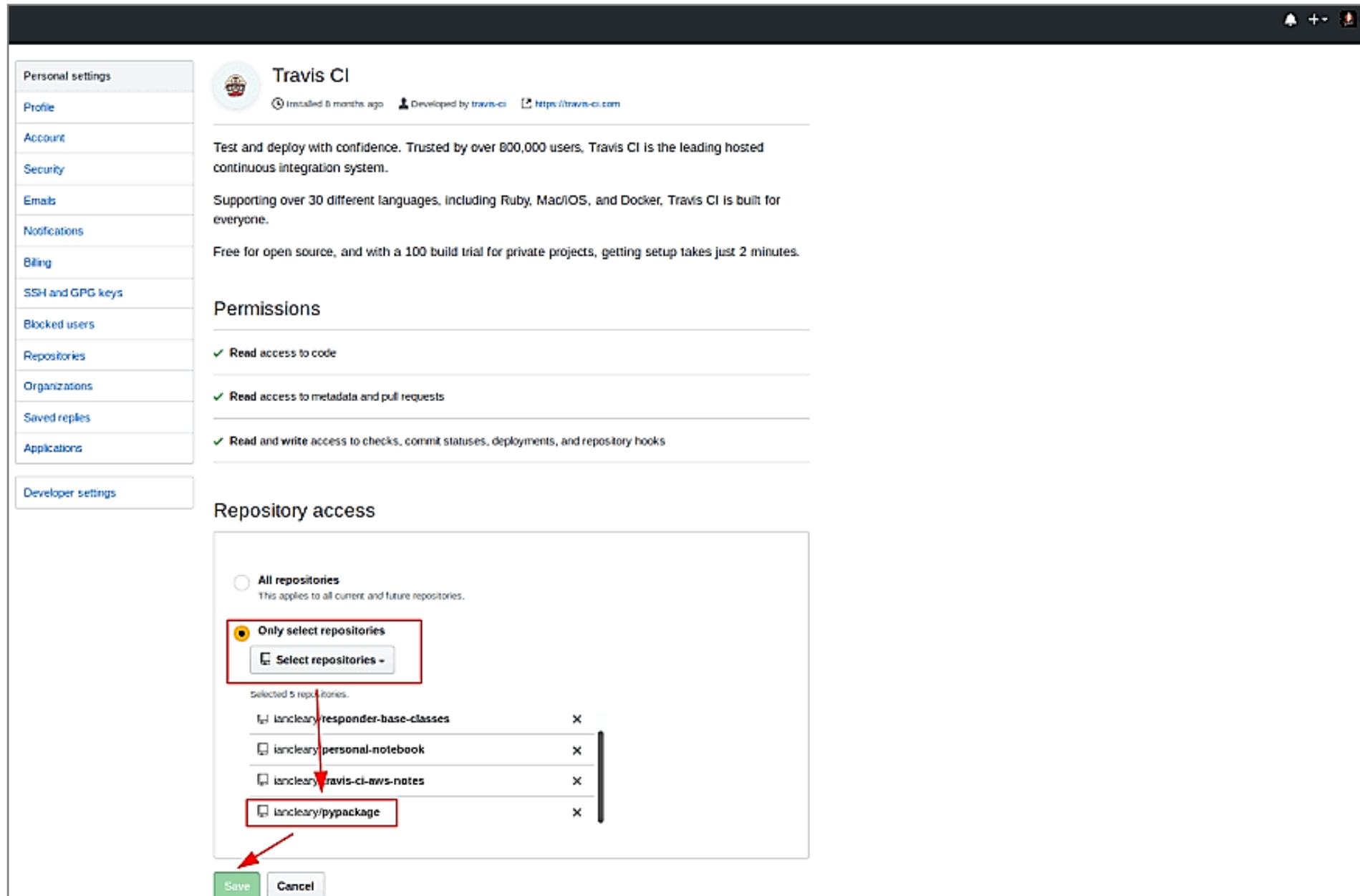
The screenshot shows the GitHub 'Applications' page. On the left is a sidebar with navigation links: Personal settings, Profile, Account, Security, Emails, Notifications, Billing, SSH and GPG keys, Blocked users, Repositories, Organizations, Saved replies, Applications (highlighted), and Developer settings. The main content area is titled 'Applications' and has three tabs: 'Installed GitHub Apps' (selected), 'Authorized GitHub Apps', and 'Authorized OAuth Apps'. Below the tabs, a message states: 'GitHub Apps augment and extend your workflows on GitHub with commercial, open source, and homegrown tools.' Two applications are listed: 'Netlify' and 'Travis CI'. Each application has a 'Configure' button to its right.



Sri Venkateswara
College of Engineering



Step IV – Connect Travis-CI to the repo



The screenshot shows the Travis CI web interface. On the left is a sidebar with navigation links: Personal settings, Profile, Account, Security, Emails, Notifications, Billing, SSH and GPG keys, Blocked users, Repositories, Organizations, Saved replies, Applications, and Developer settings. The main content area is titled 'Travis CI' and includes a description: 'Test and deploy with confidence. Trusted by over 800,000 users, Travis CI is the leading hosted continuous integration system. Supporting over 30 different languages, including Ruby, MacOS, and Docker, Travis CI is built for everyone. Free for open source, and with a 100 build trial for private projects, getting setup takes just 2 minutes.'

Below the description is the 'Permissions' section, which lists three permissions with green checkmarks: 'Read access to code', 'Read access to metadata and pull requests', and 'Read and write access to checks, commit statuses, deployments, and repository hooks'.

The 'Repository access' section is highlighted with a red box. It contains two radio buttons: 'All repositories' (unselected) and 'Only select repositories' (selected). Below the 'Only select repositories' option is a dropdown menu labeled 'Select repositories -'. A red arrow points from this dropdown to a list of selected repositories. The list shows four repositories: 'lanceary/responder-base-classes', 'lanceary/personal-notebook', 'lanceary/travis-ci-aws-notes', and 'lanceary/pyppackage'. The last repository, 'lanceary/pyppackage', is highlighted with a red box. At the bottom of the 'Repository access' section are 'Save' and 'Cancel' buttons. A red arrow points from the 'Save' button to the bottom of the page.



Travis CI



Sri Venkateswara
College of Engineering



Step V –Now we are configured, let's start a build

```
182 $ python --version
183 Python 3.6.7
184 $ pip --version
185 pip 10.0.2 from /home/travis/virtualenv/python3.6.7/lib/python3.6/site-packages/pip (python 3.6)
186 $ pip install cookiecutter
187 $ pip install pipenv --upgrade-strategy=only-if-needed
188 $ bash ./test.sh
189
190 Courtesy Notice: Pipenv found itself running within a virtual environment, so it will automatically use that environment, instead of creating its own for any project. You can set PIPENV_IGNORE_VIRTUALENVS=1 to force pipenv to ignore that environment and create its own instead. You can set PIPENV_VERBOSITY=-1 to suppress this warning.
191
192 Pipfile.lock not found, creating.
193 Lacking [dev-packages] dependencies.
194 Lacking [packages] dependencies.
195 Updated Pipfile.lock (e61c74)
196 Installing dependencies from Pipfile.lock (e61c74)
197
198 ===== ss/ss - 00:01:12 =====
199 +pytest --cov=sampled --cov-tests --cov-report=term-missing
200 ===== test session starts =====
201 platform linux -- Python 3.6.7, pytest-3.2.2, py-1.8.0, pluggy-0.10.0
202 rootdir: /home/travis/build/iancleary/pypackage, inifile: pytest.ini
203 plugins: cov-2.0.1
204 collected 1 item
205
206 tests/test_module.py . [100%]
207
208 ===== warnings summary =====
209 /home/travis/virtualenv/python3.6.7/lib/python3.6/distutils/_init_.py:4
210 /home/travis/virtualenv/python3.6.7/lib/python3.6/distutils/_init_.py:4: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses
211     import imp
212
213 -- Docs: https://docs.pytest.org/en/latest/warnings.html
214
215 ----- coverage: platform linux, python 3.6.7-final-0 -----
216
217 Name Stmts Miss Cover Missing
218 -----
219 sampled/_init_.py 1 0 100%
220 sampled/module.py 3 0 100%
221 tests/_init_.py 0 0 100%
222 tests/conftest.py 0 0 100%
223 tests/test_module.py 4 0 100%
224
225 TOTAL 8 0 100%
226
227 ===== 1 passed, 1 warnings in 0.13s =====
228
229 +bash ./scripts/lint.sh
230 +mypy sampled --disallow-untyped-defs
231 Success: no issues found in 2 source files
232 +black sampled tests --check
233 All done! 🎉 🍌 🍌
234 5 files would be left unchanged.
235 +isort --multi-line=3 --trailing-comma --force-grid-wrap=8 --combine-as --line-width 88 --recursive --check-only --thirdparty sampled sampled tests
236 Courtesy Notice: Pipenv found itself running within a virtual environment, so it will automatically use that environment, instead of creating its own for any project. You can set PIPENV_IGNORE_VIRTUALENVS=1 to force pipenv to ignore that environment and create its own instead. You can set PIPENV_VERBOSITY=-1 to suppress this warning.
237
238 INFO - cleaning site directory
239 INFO - building documentation to directory: /home/travis/build/iancleary/pypackage/appackage/site
240 The command "bash ./test.sh" exited with 0.
241
242 store build cache
243
244 Done. Your build exited with 0.
```

1.1: Make a simple change to the repo, either a commit, or open a new branch to start a build! Grab your 🏠 and let's watch the CI process start and complete!



Sri Venkateswara
College of Engineering



Travis CI Dashboard for **iancleary / pypackage** (build: passing)

My Repositories | Running (0/0) | +

- ✓ **iancleary/pypackage** # 30
 - Duration: 13 min 50 sec
 - Finished: 3 days ago
- ✓ ianclary/responder-base-class # 125
 - Duration: 9 min 51 sec
 - Finished: 17 days ago
- ✓ ianclary/personal-notebook # 21
 - Duration: 14 min 43 sec
 - Finished: about a month ago
- ✓ ianclary/travis-ci-aws-notes # 26
 - Duration: 34 sec
 - Finished: 2 months ago

Current | Branches | Build History | Pull Requests

✓ **master** Merge pull request #9 from ianclary/bug/repo-name
 Bug/repo name
 ↳ Commit r3d7982
 ↳ Compare 51781d2...f3d7982

no environment variables set
 ↳ Ram for 7 min 15 sec
 ↳ Total time 13 min 50 sec
 ↳ 3 days ago

Build jobs

Build	Platform	OS	Python	Environment	Duration	Details
✓ # 30.1	AMD64	Linux	Python: 3.6	no environment variables set	2 min 45 sec	View details
✓ # 30.2	AMD64	Linux	Python: 3.7	no environment variables set	2 min 23 sec	View details
✓ # 30.3	AMD64	Linux	Python: 3.8	no environment variables set	4 min 16 sec	View details
✓ # 30.4	AMD64	Linux	Python: nightly	no environment variables set	4 min 26 sec	View details

Allowed Failures ?

I.2: New project added to Travis CI

The three red boxes above are:

1. The connected repo that is currently selected.
2. The action that started a build (commit or pull request)
3. The build matrix visualized, with links to view more details.



Sri Venkateswara
College of Engineering

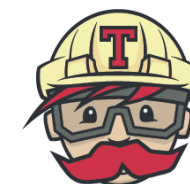


Where to go from here?

- Start offloading and automating your testing, deployments, and other activities. There is so many opportunities to tie in different hooks or features, this is just the beginning.

Some examples of what else can you automate:

- **PyPi Publishing**
- **Jupyter Notebook image deployment with CI/CD to Docker Hub**
- **Amazon EC2 instance deployments**



Travis CI



Sri Venkateswara
College of Engineering

