

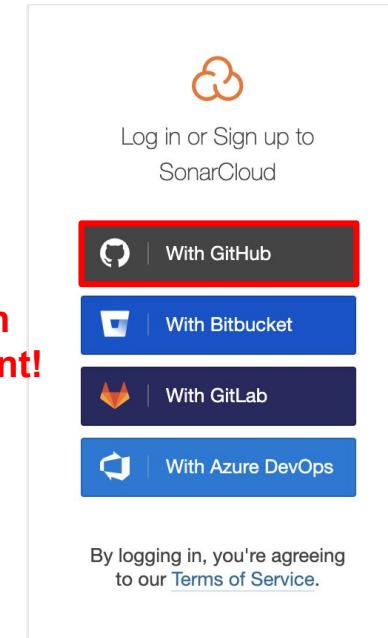
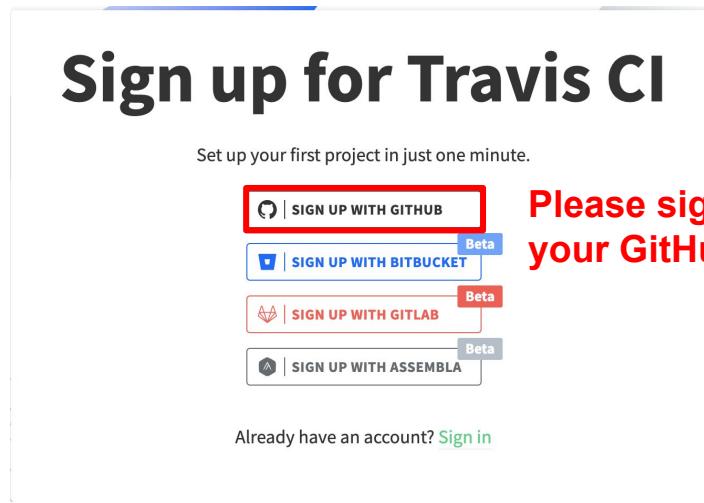
SWPP Practice Session #8

FE-BE Integration & CI Environment Setup

2022 Oct 26

Set Up

- Create your TravisCI account
 - <https://app.travis-ci.com/signup>
- Create your SonarCloud account
 - <https://sonarcloud.io/sessions/new>



Set Up

- Please fork the repository for today's practice:

<https://github.com/swpp22fall-practice-sessions/swpp-p8-integration-practice>

- Please Install travis cli:

- For Ubuntu:

- \$ sudo apt install ruby-full
\$ gem install travis

- For Mac:

- \$ brew install travis

- If you are using Mac with an M1 chip and failed to install travis, the link below may help troubleshooting:

<https://blog.dalso.org/article/m1-mac-brew-error-cannot-install-in-homebrew-on-arm-processor-in-intel-default-prefix>

Setup - docker container

- run docker container

```
$ docker run --rm -it \
    --ipc=host \
    --name "practice8" \
    -p 0.0.0.0:3000:3000 -p 0.0.0.0:8000:8000 \
    -v ${PWD}:/home \
    snuspl/swpp:practice8
```

Announcement

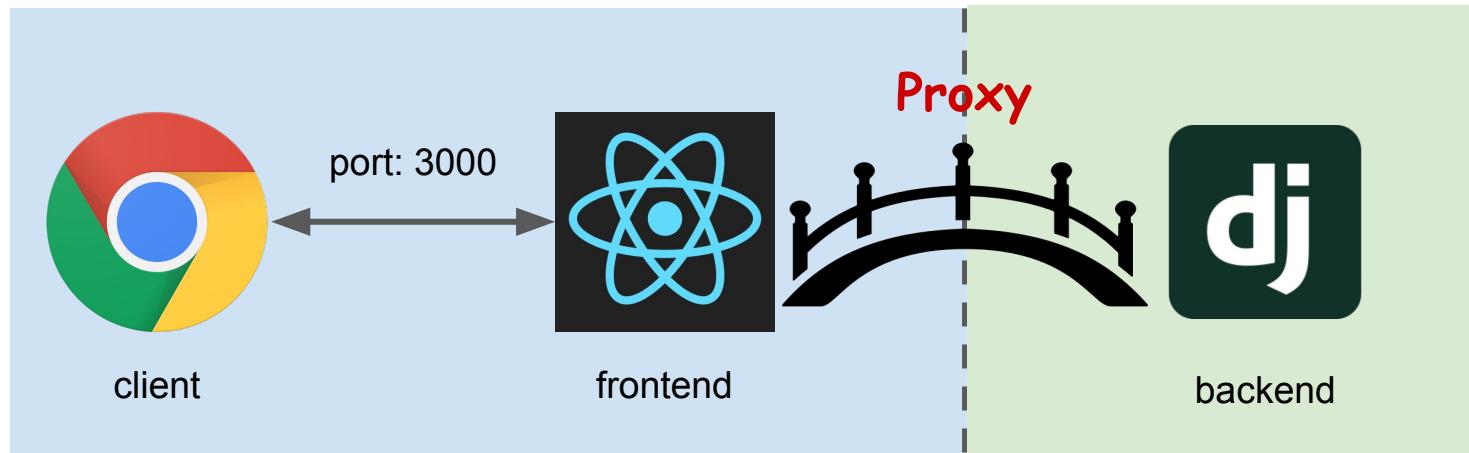
- Sprint #2
 - Don't forget to submit the following documents.
 - (1) Design and planning (wiki, pdf)
 - (2) Sprint #2 backlog (pdf)
 - Due: **10/29(Sat) 6pm**
 - **Send the materials to swpp.22.staff@spl.snu.ac.kr**
- Homework 4
 - Due: **11/2(Wed) 6pm**
 - Don't forget to add `swpp-tas` as your collaborator.

Today's Tasks

- **One Individual Task**
 - To submit today's practice assignment, fork the following repo.
 - <https://github.com/swpp22fall-practice-sessions/swpp-p8-integration-practice>
 - Implement today's tasks and send PR.
 - Due: 10/27 (Thu) 9pm
- **One Team Task**
 - Setup Travis CI and SonarCloud and create a pull request in your team repo.
 - Set jaewooMaeng as your reviewer.
 - Due: 10/27 (Thu) 9pm

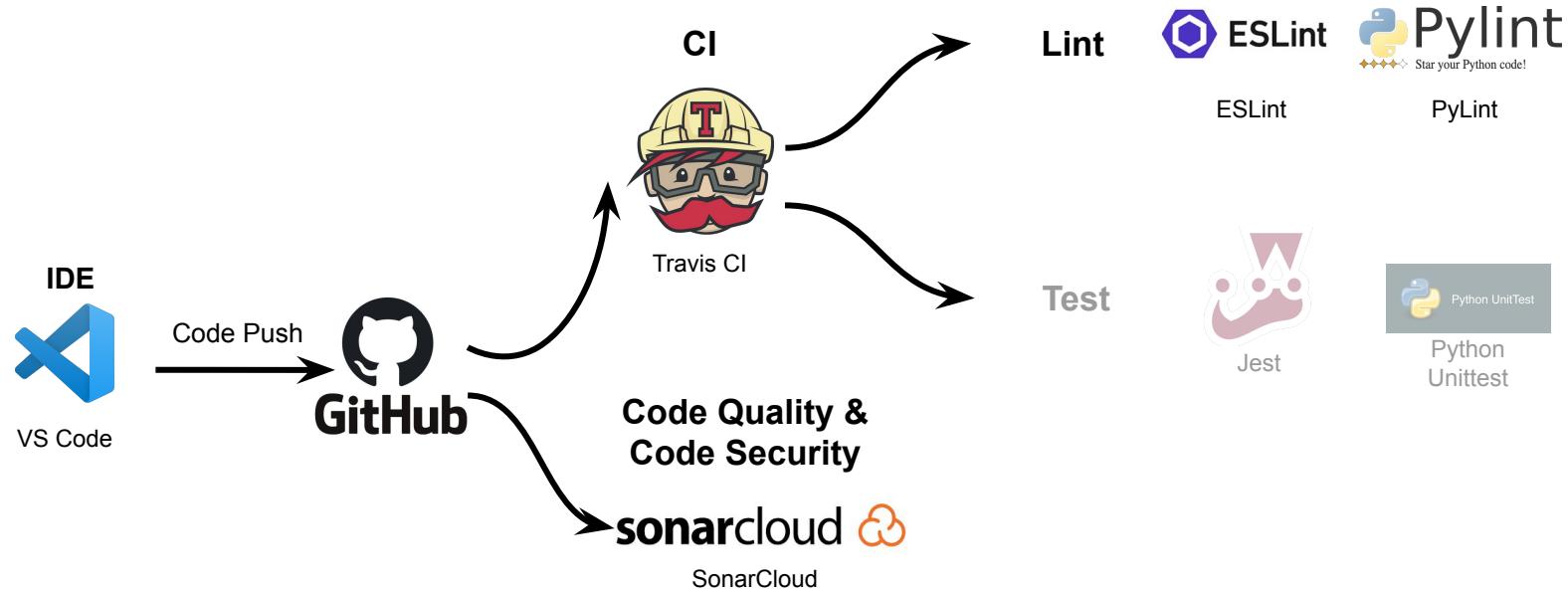
Today's Contents

1. Frontend-Backend Integration
 - a. Connect frontend and backend with Proxy server
 - b. Pass CSRF tokens from frontend to backend



Today's Contents

2. Continuous Integration(CI) and code analysis tool setup
 - a. Automate static code analysis using Lint tools through Travis CI
 - b. Automate static code quality analysis with SonarCloud



Frontend-Backend Integration

Frontend-Backend Integration

What we will learn

1. How to connect frontend and backend through proxy server
2. How to handle CSRF token in frontend side.
 - a. React / Axios
 - b. React / Fetch
 - c. React Form

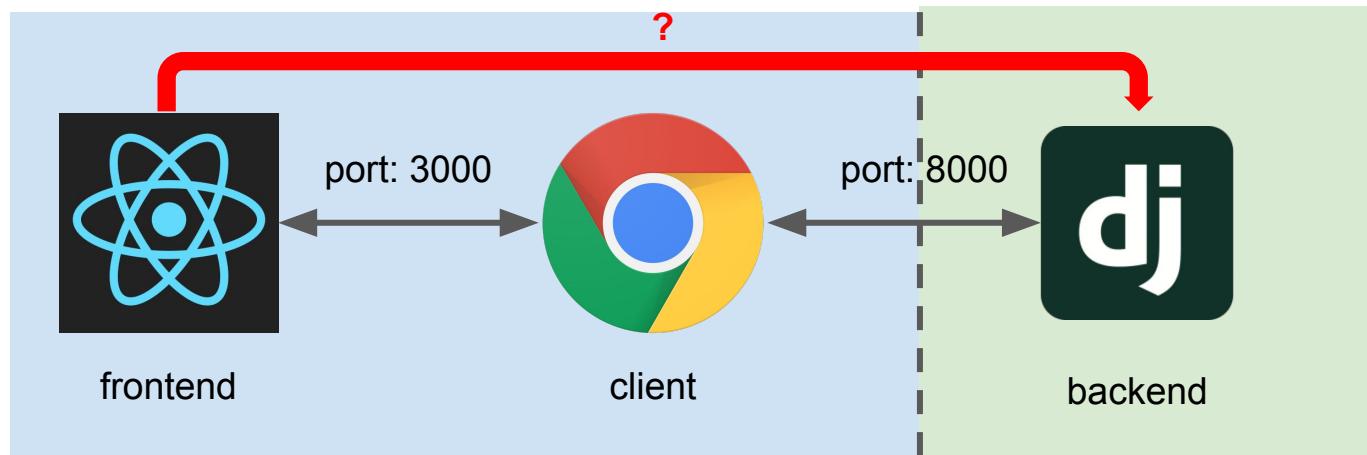
Frontend-Backend Integration

What we will learn

1. How to connect frontend and backend through proxy server
2. How to handle CSRF token in frontend side.
 - a. React / Axios
 - b. React / Fetch
 - c. React Form

How to connect React and Django?

- Now you can make both frontend and backend of a website.
- How to use backend API in frontend side?
 - Django → port: 8000
 - React → port: 3000



Deployment Layout

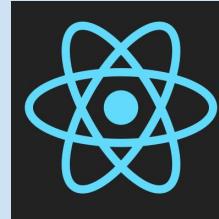
1. Client sends the HTTP request



2. Nginx handles all requests



3. React (+α) sends internal API request via Nginx

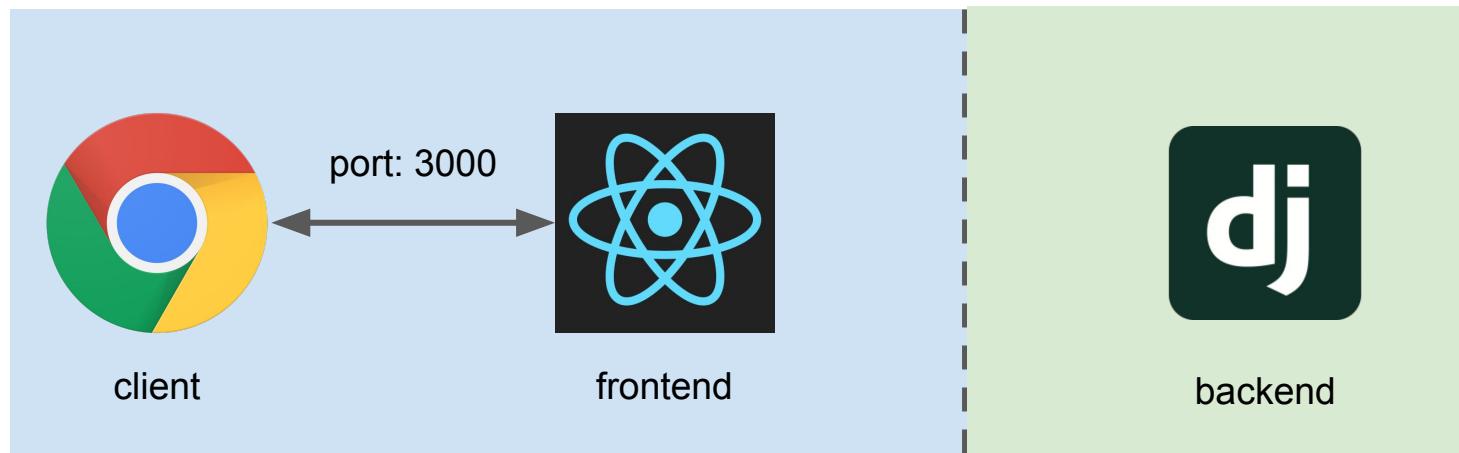


4. uWSGI sends request to Django

We will cover the deployment settings in the future lecture.

How to connect React and Django?

- Let's try what happens when we access the frontend server without any connection.



How to connect React and Django?

- Let's try what happens when we access the frontend server without any connection.

Terminal 1

```
cd frontend  
yarn  
yarn start
```

Terminal 2

```
cd backend  
python manage.py makemigrations  
python manage.py migrate  
python manage.py runserver 0.0.0.0:8000
```

Use Proxy

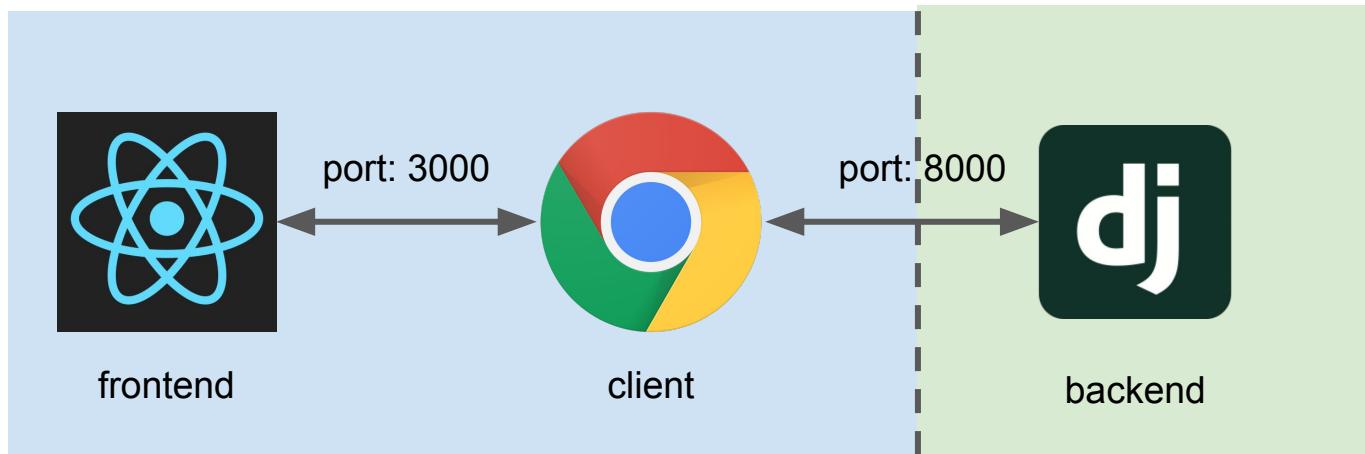
- When you try to send a request (`/hero/info/`) with axios, it raise a 404 error.

```
✖ ▶ POST http://localhost:3000/hero/info/ 404 (Not Found)
✖ ▶ Uncaught (in promise) Error: Request failed with status code 404
    at createError (createError.js:16)
    at settle (settle.js:17)
    at XMLHttpRequest.handleLoad (xhr.js:69)
```

- Add the following line at the end of `frontend/package.json`.
 - `"proxy": "http://localhost:8000"`
 - This will enable proxy server to bridge frontend and backend.
 - Run `'yarn start'` again when you add proxy settings in `package.json`.

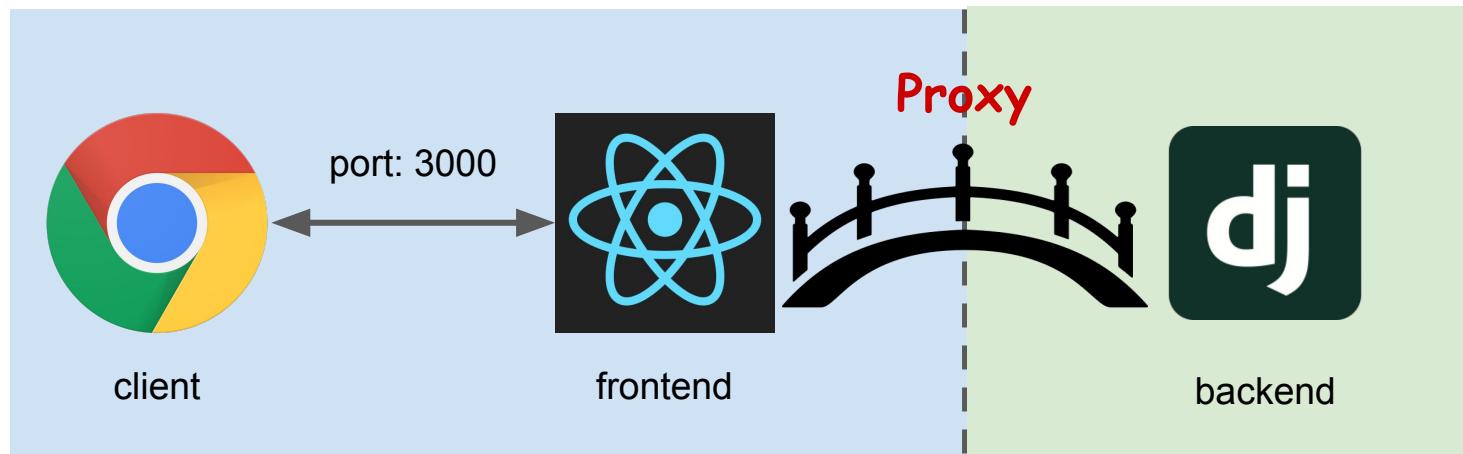
Use Proxy

- React frontend server cannot directly communicate with Django backend server.
 - `axios.get('hero/info/')`
 - `fetch('hero/info/')`



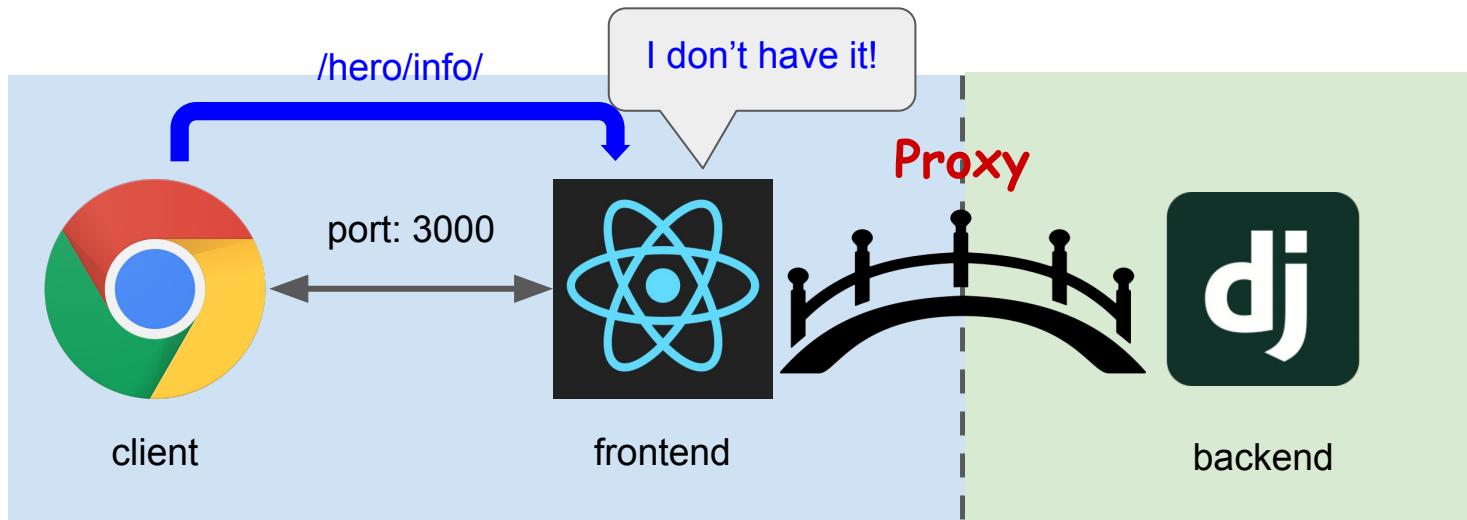
Use Proxy

- The web browser only communicates with a frontend server (localhost:3000).



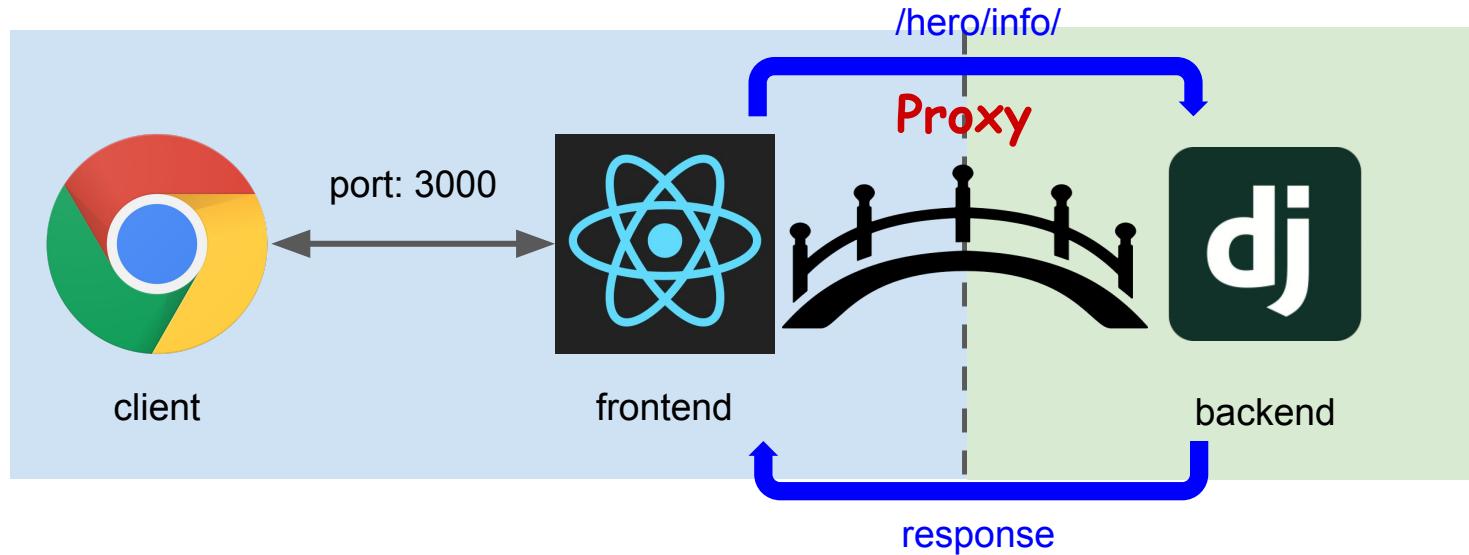
Use Proxy

- User sends a request (`localhost:3000/hero/info/`) to React server.
- React server cannot find the identical URL from itself.



Use Proxy

- Instead of raising an error instantly, React forwards the request to backend server. (Django)
- A response is returned from backend server to frontend.



Frontend-Backend Integration

What we will learn

1. How to connect frontend and backend through proxy server
2. How to handle CSRF token in frontend side.
 - a. React / Axios
 - b. React / Fetch
 - c. React Form

How to handle CSRF token?

- Now we have a proxy to bridge FE and BE.
- But we still have a problem :(
- When you try to send a POST request, 403 error occurs due to the absence of CSRF token.

Add a New Hero!

Name

Batman

Age

45

Submit



```
Forbidden (CSRF cookie not set.): /hero/info/  
[24/Oct/2020 16:24:13] "POST /hero/info/ HTTP/1.1" 403 2864
```

Review: CSRF token

- Whenever server responds to a request, it generates a new token.
 - Store it to session, and send it to client.
- Client send form(or request) with that CSRF token.
 - The token is embedded as a hidden property of the form.
- If server cannot find client's CSRF token or provided the wrong token, just reject the request.

Handling CSRF Token: Django setting

- React renders component dynamically.
 - Form is dynamically added to the page.
- Django does not set the CSRF token cookie if the view is not rendering a template.
- We need to enforce a view to send the CSRF cookie.
 - Django provides a view decorator for this purpose:
 - `ensure_csrf_cookie(view)`

```
from django.views.decorators.csrf import ensure_csrf_cookie
```

```
@ensure_csrf_cookie
```

```
def hero_list(request):
```

```
    if request.method == 'GET':
```

```
        hero_all_list = [hero for hero in Hero.objects.all().values()]
```

```
        return JsonResponse(hero_all_list, safe=False)
```

```
    elif request.method == 'POST':
```

Handling CSRF Token: (1) React / Axios

- Attach CSRF token in the header of Axios call.
- You can automatically attach the CSRF token with the following code.

```
frontend/src/store/slices/hero.ts
```

```
import axios from 'axios';

axios.defaults.xsrfCookieName = 'csrftoken';
axios.defaults.xsrfHeaderName = 'X-CSRFToken';
```

Handling CSRF Token: (1) React / Axios

- Now it works!

```
[26/Oct/2020 07:48:32] "POST /hero/info/ HTTP/1.1" 201 40
```

Add a New Hero!

Name

Age

Submit



MY FAVORITE HEROS	
Ironman	
Superman	
Batman	
New Hero	

Handling CSRF Token: (2) React / Fetch

- [Django docs](#) provides a JS function to get the CSRF token from the cookie.
- Retrieve the CSRF token by calling `getCookie('csrftoken')`.

```
function getCookie(name) {  
    let cookieValue = null;  
    if (document.cookie && document.cookie !== '') {  
        const cookies = document.cookie.split(';');  
        for (let i = 0; i < cookies.length; i++) {  
            const cookie = cookies[i].trim();  
            if (cookie.substring(0, name.length + 1)  
===(name + '=')) {  
                cookieValue =  
decodeURIComponent(cookie.substring(name.length + 1));  
                break;  
            }  
        }  
    }  
    return cookieValue;  
}
```

```
const csrftoken = getCookie('csrftoken');
```

Handling CSRF Token: (2) React / Fetch

- [Django docs](#) provides a JS function to get the CSRF token from the cookie.
- Retrieve the CSRF token by calling `getCookie('csrftoken')`.
- Use the retrieved CSRF token when sending a request with fetch API.
 - Embed it to X-CSRFToken header.

Don't send csrftoken to another domain.

```
fetch(url, {
  credentials: 'include',
  method: 'POST',
  mode: 'same-origin',  
    ←
  headers: {
    'Accept': 'application/json',
    'Content-Type': 'application/json',
    'X-CSRFToken': csrftoken
  },
  body: {
    // ...
  }
}).then(function(response) {
  // ...
});
```

retrieved csrftoken

Handling CSRF Token: (3) React Form

- Django template use `{ % csrf_token % }` tag to render a CSRF token inside a form. → React cannot use this tag.
- Need to create a frontend component to retrieve the CSRF token.

```
// csrftoken.js
import React from 'react';

const csrftoken = getCookie('csrftoken');

const CSRFToken = () => {
    return (
        <input type="hidden" name="csrfmiddlewaretoken" value={csrftoken} />
    );
};

export default CSRFToken;
```

Handling CSRF Token: (3) React Form

- Django template use `{ % csrf_token % }` tag to render a CSRF token inside a form. → React cannot use this tag.
- Need to create a frontend component to retrieve the CSRF token.

```
import React, { useState } from 'react';
import CSRFToken from './csrftoken';

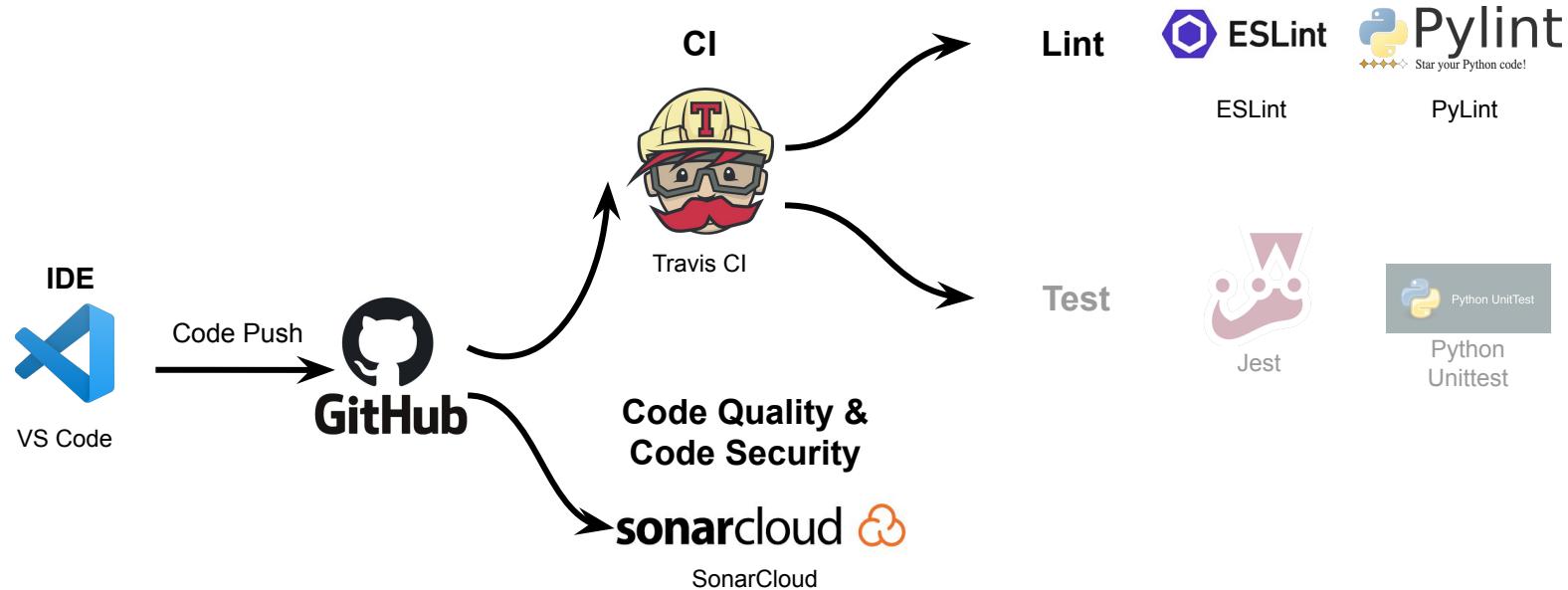
function ExampleForm() {
    return (
        <form action="/endpoint" method="post">
            <CSRFToken />
            <button type="submit">Send</button>
        </form>
    );
}
export default ExampleForm;
```

Today's Task (Individual)

1. Set proxy server in your package.json
2. Apply Axios CSRF settings
3. Create a pull request
 - Due: 10/27 (Thu) 9pm

Today's Contents

2. Continuous Integration(CI) and code analysis tool setup
 - a. Automate static code analysis using Lint tools through Travis CI
 - b. Automate static code quality analysis with SonarCloud



Lint (ESLint + Pylint)

Lint

What we will learn

1. What is Linter? Purpose?
2. How to apply ESLint
3. How to apply PyLint

What is your coding style?

- Someone in your team may prefer `snake_case`, while you prefer `camelCase`.
- For the ease of maintenance and cooperation, we need a unified coding convention.
- Linter checks the coding style along with error-prone codes.



What is Linter?

- A static analysis tool that analyzes source code to flag programming errors, bugs, stylistic errors, and suspicious constructs.
- For our projects, we will use
 - ESLint for Javascript
 - PyLint for Python

Dynamic analysis is not everything

- **Dynamic analysis:** testing a program by executing data in real-time.
 - Unit test, integration test, acceptance test, etc.
- But as you may have realized...
 - You can achieve high coverage with incomprehensive test cases.
 - Even a good dynamic test does not guarantee soundness of your code.
- **Static analysis:** analysis is performed without executing program.
 - It shows potential problems of your code.
 - It also doesn't always guarantee soundness of your code, but still very useful.

Dynamic analysis vs. Static analysis

Dynamic analysis

- **Can test developer's intent.**
 - Hey, `multiply(3,4)` should have returned 12, not 7!
- int multiply(int a, int b) {
 return a + b;
}
- Only reports **actual** failure in execution.
- No false positive / Possibly many false negative
 - If unit test fails, the code is guaranteed to be wrong.
 - No failure in unit test does not guarantee soundness of your code.
- Takes quite a lot of effort.
 - Changing feature often requires changing test code.

Static analysis

- Cannot test developer's intent
 - No possible crash (e.g. segfault). It's fine!
- int multiply(int a, int b) {
 return a + b;
}
- Only reports **possible** failure in execution.
- Trade-off between false positive and false negative
 - You can configure the balance.
 - For example, you NEVER want a false negative in an autopilot program for aeroplanes.
- No extra effort
 - Changing feature does not require extra effort.

ESLint

- Install ESLint: `$ yarn add eslint --save-dev`
- Initialize ESLint: `$./node_modules/.bin/eslint --init`

Choose whatever you like

```
*[practice/3-lint] [/home/frontend]$ ./node_modules/.bin/eslint --init
You can also run this command directly using 'npm init @eslint/config'.
npx: installed 41 in 4.396s
✓ How would you like to use ESLint? · style
✓ What type of modules does your project use? · esm
✓ Which framework does your project use? · react
✓ Does your project use TypeScript? · No / Yes
✓ Where does your code run? · browser, node
✓ How would you like to define a style for your project? · guide
✓ Which style guide do you want to follow? · standard-with-typescript
✓ What format do you want your config file to be in? · Javascript
Checking peerDependencies of eslint-config-standard-with-typescript@latest
Local ESLint installation not found.
The config that you've selected requires the following dependencies:

eslint-plugin-react@latest eslint-config-standard-with-typescript@latest @typescript-eslint/eslint-plugin@5.0.0 eslint@^8.0.1 eslint-plugin-import@^2.25.2 eslint-plugin-n@^15.0.0 eslint-plugin-promise@^6.0.0 typescript@*
✓ Would you like to install them now? · No / Yes
✓ Which package manager do you want to use? · yarn
```

ESLint

- Open .eslint.js and add the highlighted configurations:

```
// project_root/frontend/.eslintrc.js
//...
env: {
  //...
  jest: true // This is for jest
},
extends: [
  'plugin:react/recommended', // This is for React
  'standard-with-typescript'
],
parserOptions: {
  //...
  project: ['./tsconfig.json'] // This is for Typescript
}
//...
```

ESLint

- Run ESLint: \$ `./node_modules/.bin/eslint yourfile.tsx (.js)`
- You can see ESLint complain about your syntax, coding style, etc. :)

```
*[practice/3-lint] [/home/frontend]$ ./node_modules/.bin/eslint src/App.tsx
Warning: React version not specified in eslint-plugin-react settings. See https://github.com/jsx-eslint/eslin
t-plugin-react#configuration .

/home/frontend/src/App.tsx
  1:19  error  Extra semicolon          @typescript-eslint/semi
  3:54  error  Extra semicolon          @typescript-eslint/semi
  4:60  error  Extra semicolon          @typescript-eslint/semi
  5:60  error  Extra semicolon          @typescript-eslint/semi
  7:74  error  Extra semicolon          @typescript-eslint/semi
10:1   error  Missing return type on function      @typescript-eslint/explicit-function-return-type
       pe
  10:13 error  Missing space before function parentheses @typescript-eslint/space-before-function-paren
  12:5   error  'React' must be in scope when using JSX react/react-in-jsx-scope
  13:7   error  'React' must be in scope when using JSX react/react-in-jsx-scope
  14:9   error  'React' must be in scope when using JSX react/react-in-jsx-scope
  15:11  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  15:41  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  15:58  error  Strings must use singlequote @typescript-eslint/quotes
  16:11  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  16:45  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  17:11  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  17:44  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  17:56  error  Multiple spaces found before '/' no-multi-spaces
  18:11  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  18:36  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  19:11  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  19:36  error  'React' must be in scope when using JSX react/react-in-jsx-scope
  23:4   error  Extra semicolon          @typescript-eslint/semi
  26:19  error  Extra semicolon          @typescript-eslint/semi
  26:20  error  Newline required at end of file but not found eol-last

* 25 problems (25 errors, 0 warnings)
  11 errors and 0 warnings potentially fixable with the `--fix` option.
```

PyLint

- Install PyLint: `$ sudo apt install pylint`
 - Or `$ pip install pylint`
- Move to your project directory: `$ cd your/python/project`
- Run PyLint: `$ pylint **/*.py`
- You can config your pylint setting in `.pylintrc`.

Continuous Integration (CI)

Continuous Integration (CI)

What we will learn

1. What is CI?
2. Setting up Travis CI for your project

One common situation you will encounter...

- You push your implementation to the remote repository after you double-check that it builds well, works well, etc.
- ... but the build fails in the other teammates' machines!
- You: “Strange! It worked on my machine!”
- Your teammates:



Continuous Integration

- Keep merging your code frequently into the public repository
 - Can be done by Github Pull Request, etc...
- Continuous Integration (CI) service automatically builds and runs tests for you in a separate environment (not on your laptop!)
 - Build failure
 - Test case failure
 - Bad / suspicious code, etc.
- There are several free CI services (for public repo)
 - Travis CI
 - CircleCI
 - github actions

Setting up Travis CI

- <https://travis-ci.com/>
- Sign in with your Github account.
- You should be a member of the swsnu organization.
 - <https://app.travis-ci.com/account/repositories>
 - Let TAs know if the organization is not visible in Travis settings.

The screenshot shows the Travis CI dashboard for a user named Donghyun Kim (@swsnu). The top navigation bar includes links for Repositories, Settings, Plan, Migrate, and Plan usage. Below the navigation, there is a GitHub Apps Integration section with a button to Manage repositories on GitHub. A search bar contains the text "swpp2021-team". The main content area displays two organizations: "swsnu" and "SNU Software Platform Lab". A message at the bottom left says "MISSING AN ORGANIZATION? Review and add your authorized organizations." There are also "Settings" buttons for each organization listed.

MY ACCOUNT

Donghyun Kim

Sync account

ORGANIZATIONS

swsnu

SNU Software Platform Lab

MISSING AN ORGANIZATION?
Review and add your authorized organizations.

GitHub Apps Integration

Manage repositories on GitHub

swpp2021-team

swpp2021-team1

swpp2021-team10

Settings

Settings

Lint + CI Demo

- Check if Travis CI is installed in the repository
 - GitHub repo settings -> Integrations

Options

Manage access

Repository roles

Security & analysis

Webhooks

Notifications

Integrations

Deploy keys

Installed GitHub Apps

GitHub Apps augment and extend your workflows on GitHub with commercial, open source, and homegrown tools.

	Codecov	<button>Configure</button>
	Slack	<button>Configure</button>
	Travis CI	<button>Configure</button>

Lint + CI Demo

- Check if Travis CI can see your repository
 - Search for your team repo and check if you can access the Settings tab.

The screenshot shows a GitHub repository page for the organization "SWSnU". The top navigation bar includes links for "Repositories", "Settings", "Plan", "Migrate", and "Plan usage". Below the navigation is a section titled "GitHub Apps Integration" with a button to "Manage repositories on GitHub". At the bottom of the page is a search bar containing the text "ci-".

SWSnU

@swsnu

Repositories Settings Plan Migrate Plan usage

GitHub Apps Integration

Manage repositories on GitHub

ci-

swpp2021-ci-practice

Settings

Lint + CI Demo

- Check if Travis CI can see your repository
 - Search for your team repo and check if you can access the Settings tab.

 swsnu / swpp2021-ci-practice  build unknown

Current Branches Build History Pull Requests **Settings**

General **Manage when to run builds.**

Build pushed branches 

Build pushed pull requests 

Limit concurrent jobs 

User management

Lint + CI Demo

- Suppose we have a new React app and Django backend.
 - swpp2021-integration-practice:travis
 - We add backend dependencies in requirements.txt

```
$ cd backend  
$ echo django==4.1.2 > requirements.txt  
$ echo coverage==6.5 >> requirements.txt  
$ echo pylint==2.15.5 >> requirements.txt  
$ echo pylint-django==2.5.3 >> requirements.txt
```

```
$ tree -L 2  
.  
└── backend  
    └── backend  
        └── manage.py  
        └── requirements.txt  
└── frontend  
    ├── README.md  
    ├── node_modules  
    ├── package.json  
    ├── public  
    └── src  
        └── yarn.lock
```

Create travis config file

- File name should be `.travis.yml`
- Indentation is important!
- Build Lifecycle
 - `install` -> `script`
 - You can specify more phases such as `before_install`, `after_script`.
 - [More on Build Lifecycle](#)

```
# CI Environment
dist: bionic

# Job Configurations
jobs:
  include:
    - language: python
      python: "3.9.6"
      install:
        - cd backend
        - pip install -r requirements.txt
      script:
        - pylint **/*.py --load-plugins pylint_django
        - coverage run --source='.' manage.py test
        - coverage xml

    - language: node_js
      node_js: 14
      install:
        - cd frontend
        - yarn install
      script:
        - ./node_modules/.bin/eslint src
        - yarn test --coverage --watchAll=false
```

JOB

CI Demo

- YAML format
 - <http://yaml.org/>
- For more build configurations
 - <https://docs.travis-ci.com/user/customizing-the-build/>

CI Demo

- Now, let's see if our CI works.
- Push your changes and create a new pull request.
- And you'll see your build is on progress

 Some checks haven't completed yet Hide all checks

1 in progress check

 **Travis CI - Pull Request** *In progress — Build Started* Details

 **This branch has no conflicts with the base branch**
Merging can be performed automatically.

Merge pull request ▾ You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

CI Demo

- Your build fails if one of your `script` (defined in `.travis.yml`) fails.
 - ESLint or PyLint error
 - Failed unit test, etc.

A screenshot of a GitHub pull request interface. On the left, there's a yellow 'Merge' button. The main area shows a summary of CI status:

- All checks have failed** (red circle icon)
- 1 failing check
- Travis CI - Pull Request** Failing after 1m — Build Failed (with a red 'Details' button highlighted by a red box and arrow)
- This branch has no conflicts with the base branch** (green checkmark icon)
- Merging can be performed automatically.

At the bottom, there are buttons for "Merge pull request" and "Details". A red arrow points to the "Details" button for the Travis CI check.

CI Demo

- Detail of your test build

X # 16.1	AMD64	Bionic	</> Python: 3.7.9	no environment variables set	47 sec	
X # 16.2	AMD64	Bionic	</> Node.js: 14	no environment variables set	1 min 5 sec	

CI Demo

This build failed due to Pylint error

- Detail of your test build

```
272 **** Module hero.views
273 hero/views.py:51:9: C0303: Trailing whitespace (trailing-whitespace)
274 hero/views.py:1:0: C0114: Missing module docstring (missing-module-docstring)
275 hero/views.py:10:0: C0116: Missing function or method docstring (missing-function-docstring)
276 hero/views.py:14:0: C0103: Argument name "id" doesn't conform to snake_case naming style (invalid-name)
277 hero/views.py:14:0: C0116: Missing function or method docstring (missing-function-docstring)
278 hero/views.py:14:23: W0622: Redefining built-in 'id' (redefined-builtin)
279 hero/views.py:15:4: R1705: Unnecessary "elif" after "return" (no-else-return)
280 hero/views.py:24:8: C0103: Variable name "e" doesn't conform to snake_case naming style (invalid-name)
281 hero/views.py:24:8: W0612: Unused variable 'e' (unused-variable)
282 hero/views.py:36:0: C0116: Missing function or method docstring (missing-function-docstring)
283 hero/views.py:37:4: R1705: Unnecessary "elif" after "return" (no-else-return)
284 hero/views.py:38:0: R1721: Unnecessary use of a comprehension (unnecessary-comprehension)
285 hero/views.py:45:8: C0103: Variable name "e" doesn't conform to snake_case naming style (invalid-name)
286 hero/views.py:45:8: W0612: Unused variable 'e' (unused-variable)
287 hero/views.py:5:0: C0411: standard import "import json" should be placed before "from django.http import
    HttpResponseRedirect, JsonResponse, HttpResponseBadRequest" (wrong-import-order)
288 hero/views.py:6:0: C0411: standard import "from json.decoder import JSONDecodeError" should be placed before "from
    django.http import HttpResponseRedirect, JsonResponse, HttpResponseBadRequest" (wrong-import-order)
289
290 -----
291 Your code has been rated at 6.88/10
292
293 The command "pylint **/*.py --load-plugins pylint_django" exited with 28.
```

Setup CI on your team repository

- Please activate Travis CI on your team repository.
 - This is also included in our today's task!
- Check out this doc for your later sprint.
 - <https://docs.travis-ci.com/>

SonarCloud

SonarCloud

What we will learn

- What is SonarCloud?
- Setting up SonarCloud for your team project

SonarCloud

- Continuous code quality static analysis tool.
 - bugs, vulnerabilities, code smell
- SonarCloud vs lint
 - SonarCloud highlights issues found on new code
 - SonarCloud supports 24 languages
 - SonarCloud integrates better with Github
- SonarCloud can be used an add-on of Travis CI
- Sign in with Github
 - <https://sonarcloud.io/about>

Create your SonarCloud Organization

The screenshot shows the SonarCloud dashboard interface. At the top, there is a navigation bar with links for "My Projects" and "My Issues". A banner at the top right indicates "NEW PHP 8 now supported!". Below the navigation bar, there are sections for "Filters", "Quality Gate", and "Reliability". The "Quality Gate" section shows counts for Passed (0), Warning (0), and Failed (0) projects. The "Reliability" section shows counts for A (0), B (0), and C (0) bugs. In the center, a message states "You don't have any favorite projects yet." with a link to "Here is how to add projects to this page". At the bottom, there are buttons for "Analyze new project", "Favorite projects from your orgs", and "Favorite public projects". On the far right, a sidebar contains buttons for "Analyze new project" and "Create new organization", with the "Create new organization" button being specifically highlighted by a red box and an arrow pointing to it.

sonarcloud

My Projects My Issues

NEW PHP 8 now supported! X

Explore ?

Search for projects and files... +

Filters

Perspective: Overall Status Sort by: Name

Search by project name or key

Analyze new project

Create new organization

Quality Gate

Status	Count
Passed	0
Warning	0
Failed	0

You don't have any favorite projects yet.

Here is how to add projects to this page

Analyze new project Favorite projects from your orgs Favorite public projects

Reliability (Bugs)

Category	Count
A	0
B	0
C	0

0 of 0 shown

Create your SonarCloud Organization

Create an organization

An organization is a space where a team or a whole company can collaborate across many projects.

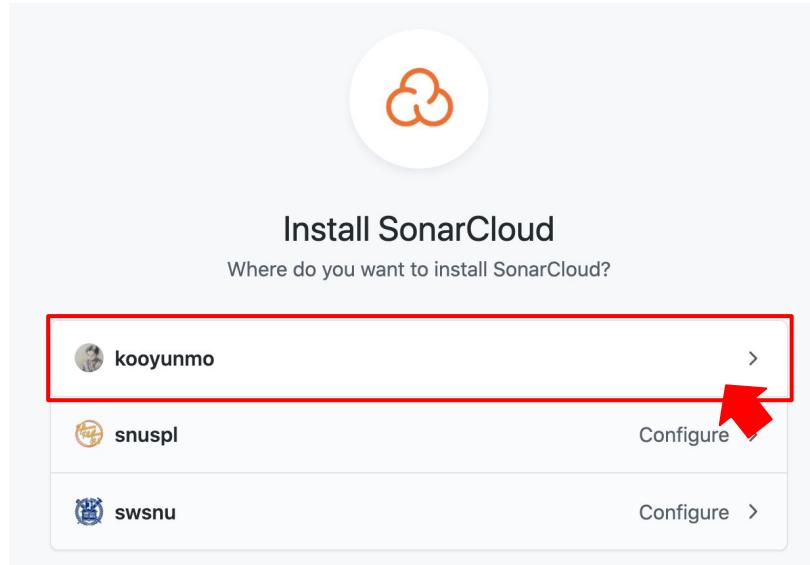


You will be asked to grant access to the SonarCloud application on your organization or user account, which will allow you to choose which repositories you want to analyze.

 Choose an organization on GitHub



Create your SonarCloud Organization



The screenshot shows a user interface for creating a SonarCloud organization. At the top center is a white circular icon containing an orange stylized 'S' logo. Below it, the text "Install SonarCloud" is centered. Underneath that, a question "Where do you want to install SonarCloud?" is displayed. A list of organizations is shown in a table format:

Organization	Action
kooyunmo	>
snuspl	Configure >
swnsu	Configure >

A red rectangular box highlights the first row ("kooyunmo"). A red arrow points from the bottom right towards the "Configure" link next to the "snuspl" organization.

Create your SonarCloud Organization

Install SonarCloud

Install on your personal account Yunmo Koo

All repositories
This applies to all current and future repositories.

Only select repositories
[Select repositories ▾](#)

Selected 1 repository.

kooyunmo/swpp-travis-sonarcloud [X](#)

...with these permissions:

Read access to code and metadata

Read and write access to checks, commit statuses, and pull requests

User permissions

SonarCloud can also request users' permission to the following resources. These permissions will be requested and authorized on an individual-user basis.

Read access to emails

[Install](#) [Cancel](#)

Next: you'll be redirected to the GitHub App's site to complete setup.



Create your SonarCloud Organization

Create an organization

An organization is a space where a team or a whole company can collaborate across many projects.

1 Import organization details

Import  **Yunmo Koo** into a SonarCloud organization X

Key* 

This key will be the unique identifier of your organization. You will have to include it as a parameter when configuring your analysis. It could be the name of your company or your team. Note that this key cannot be changed once created.

Add additional info ▾



All members from your GitHub organization Yunmo Koo will be added to your SonarCloud organization. As they connect to SonarCloud with their GitHub account, members will automatically have access to your SonarCloud organization and its projects.

[See all members on GitHub](#)

Continue



Create your SonarCloud Organization

Create an organization

An organization is a space where a team or a whole company can collaborate across many projects.

1 Import organization details

 kooyunmo

2 Choose a plan

Free plan

€0

All projects you analyze will be public.
Anyone can browse source code.

Paid plan

from €10

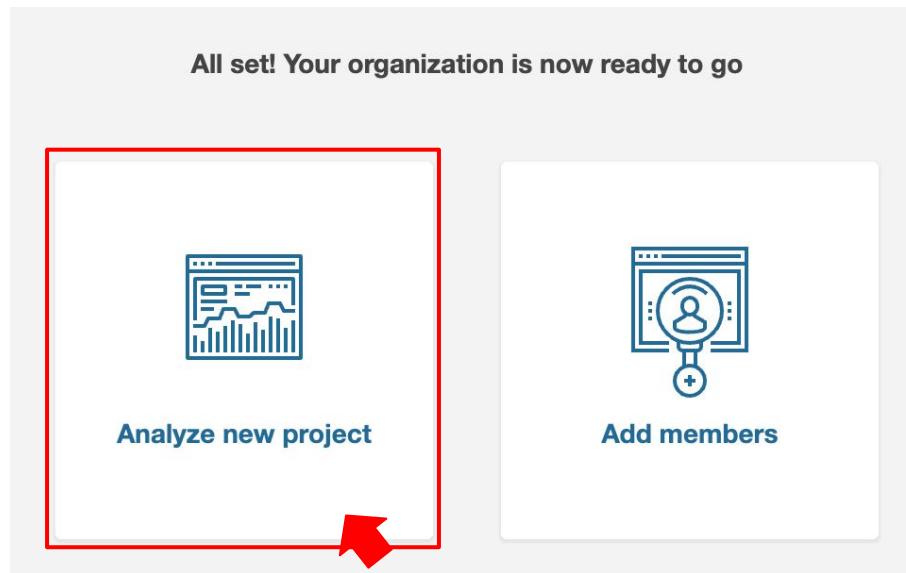
- ✓ Unlimited private projects
- ✓ Strict control over who can view your private data
- ✓ No commitments, cancel anytime
- ✓ 14 days free trial.

[Learn more](#)

[Create Organization](#)



Create your SonarCloud Organization



Create your SonarCloud Organization

Analyze projects - Select repositories

Organization*

 Yunmo Koo kooyunmo

[Import another organization](#)

 swpp-travis-sonarcloud

Don't see your repo? Check your [GitHub app configuration](#).

1 repository selected

1 repository will be created as a public project on SonarCloud

[Set Up](#)

Just testing? You can [create a project manually](#).

Setup a [monorepo](#).

Create your SonarCloud Organization

The screenshot shows the SonarCloud interface for managing analysis methods. On the left, there's a sidebar with project navigation and a red box highlighting the 'Administration' tab. The main content area is titled 'Analysis Method' and contains instructions for managing analysis settings. A prominent feature is a toggle switch for 'SonarCloud Automatic Analysis', which is currently set to 'Not recommended' (indicated by a red 'X'). A red arrow points to this toggle switch. Below it, there's a section for supported analysis methods, including GitHub Actions, Travis CI, Circle CI, and Other CI, each with a 'Follow the tutorial' link. At the bottom, there's a section for manually configuring analysis from local sources, also with a 'Follow the tutorial' link.

sonarcloud

swpp-p8-integration-practice

PUBLIC

Configure

Main Branch

Pull Requests 0

Branches 0

Information

Administration

My Projects My Issues Explore Q

swpp22fall-practice-sessions > swpp-p8-integration-practice > Administration - Analysis Method

Analysis Method

Use this page to manage and set-up the way your analyses are performed.

SonarCloud Automatic Analysis X Not recommended ↻ Recheck compatibility

SonarCloud automatically analyzes your default branch and Pull Requests. [Learn More ↗](#)

Supported analysis methods

GitHub Actions

Use a GitHub Action to configure your analysis.

[Follow the tutorial](#)

Travis CI

Use the SonarCloud add-on for Travis CI, and integrate analyses in your workflow

[Follow the tutorial](#)

Circle CI

Use the SonarCloud add-on for Circle CI, and integrate analyses in your workflow

[Follow the tutorial](#)

Other CI

SonarCloud integrates in your workflow no matter which tool you're using. Get generic instructions that you can adapt to your tool.

[Follow the tutorial](#)

Manually

Just testing? Configure your analysis from your local sources.

[Follow the tutorial](#)

your SonarCloud Organization

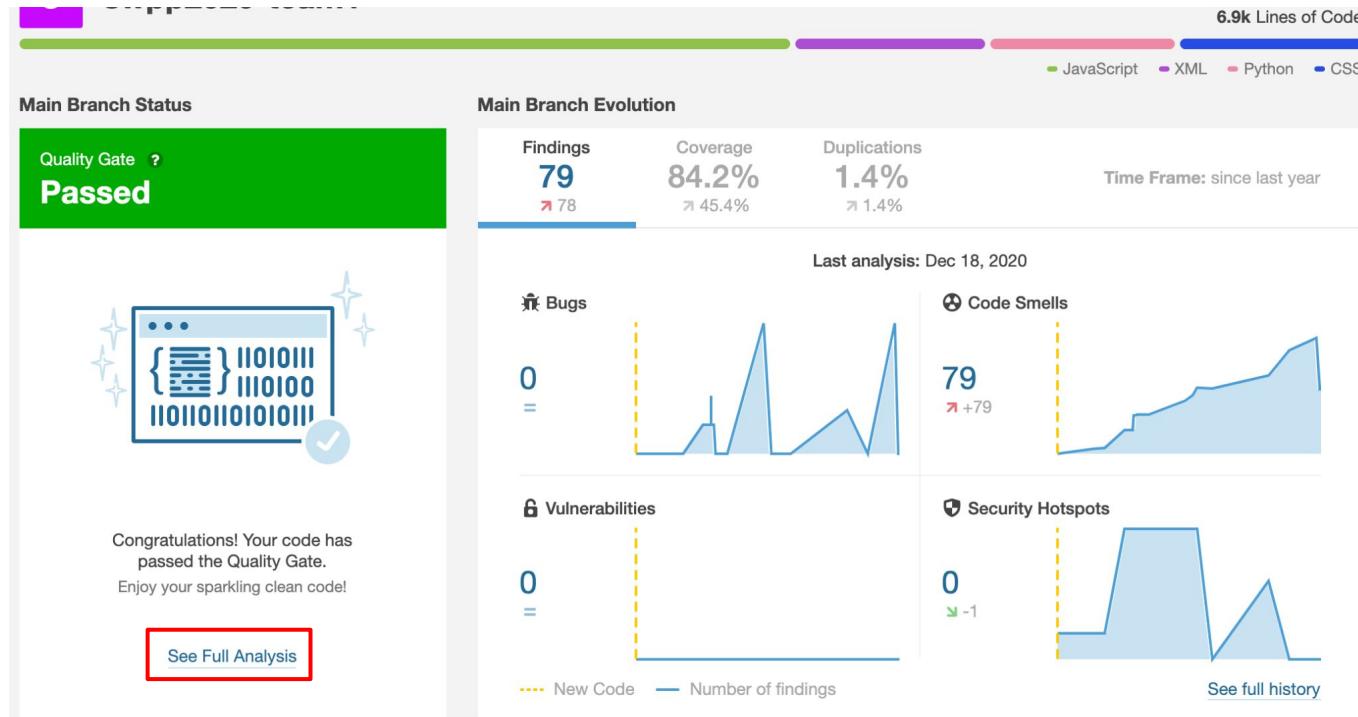
The SonarCloud projects for your team is in `swsnu`:

- <https://sonarcloud.io/organizations/swsnu/projects>

You can access your team sonarcloud page:

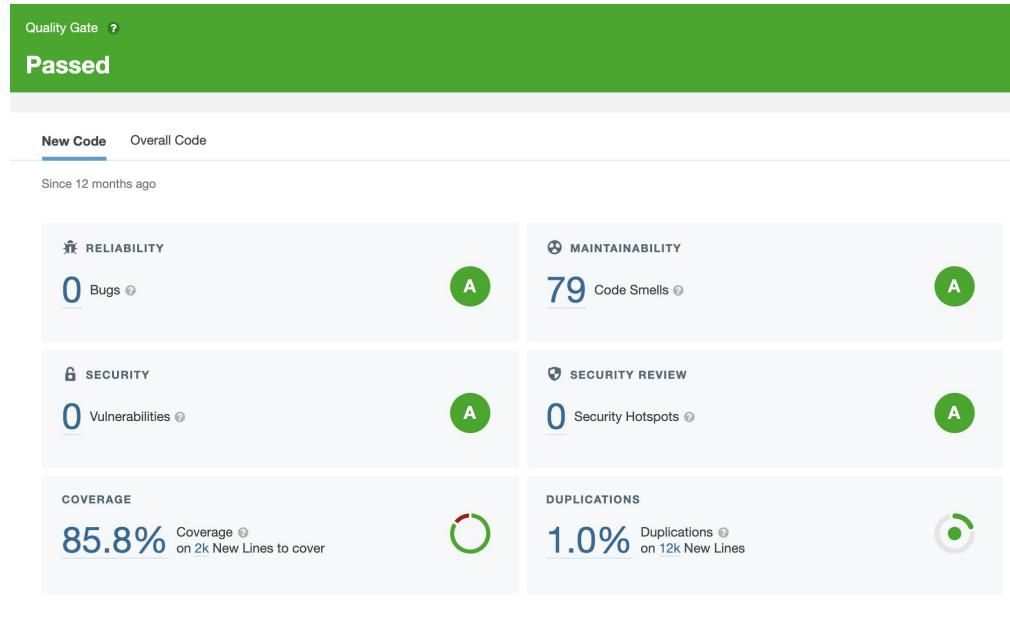
- https://sonarcloud.io/project/overview?id=swsnu_swppfall2022-teamX
- Replace `X` to your team number (1 ~ 20).

Code Analysis in SonarCloud



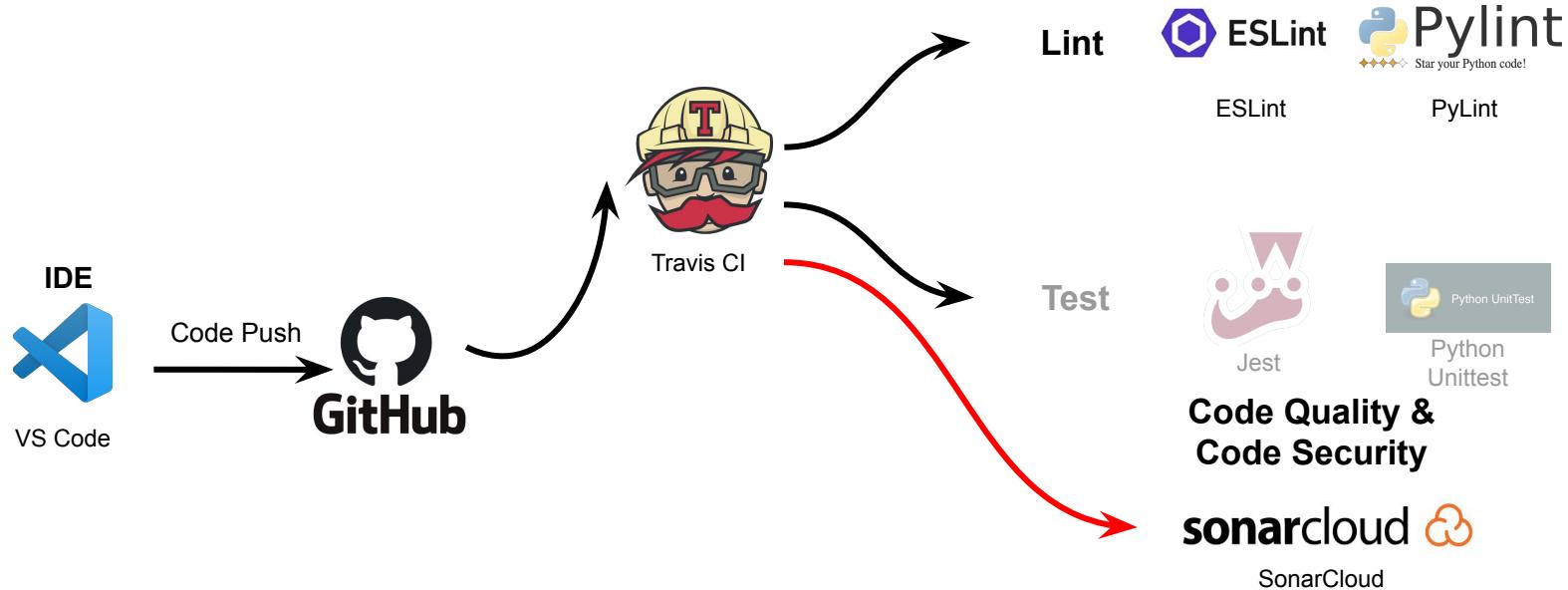
Code Analysis in SonarCloud

Should pass all the sections to pass the Quality Gate.



SonarCloud as an add-on of Travis CI

- Can trigger sonar-scanner with Travis CI
- Need connection between SonarCloud and Travis CI



Turn-off Automatic Analysis

Analysis Method

Use this page to manage and set-up the way your analyses are performed.

SonarCloud Automatic Analysis ✖ Not recommended ? [Recheck compatibility](#)

SonarCloud automatically analyzes your default branch and Pull Requests. [Learn More](#)



Supported analysis methods

GitHub Actions

Use a GitHub Action to configure your analysis.

[Follow the tutorial](#)

Travis CI

Use the SonarCloud add-on for Travis CI, and integrate analyses in your workflow

[Follow the tutorial](#)

Circle CI

Use the SonarCloud add-on for Circle CI, and integrate analyses in your workflow

[Follow the tutorial](#)

Other CI

SonarCloud integrates in your workflow no matter which tool you're using. Get generic instructions that you can adapt to your tool.

[Follow the tutorial](#)

Manually

Just testing? Configure your analysis from your local sources.

[Follow the tutorial](#)

Prepare Configuration Files

- Copy the output of `travis encrypt`
 - *token* is the token that you've just generated.
 - copy the result of encrypt command

```
$ sudo apt install ruby-full  
$ gem install travis
```

for mac users:

```
$ brew install travis
```

```
$ travis login --pro --github-token github_token  
$ travis encrypt --pro token
```

Please add the following to your `.travis.yml` file:

```
secure: "qerLSI0jhkQZrTtvUnkQMGSjQSS7v9TeghnzbQy2ipK/zbabHVBVDBSCAKEBROcP  
CB3nLV5GWPnHHaB5P6PFr9+WE5rLIpuqcqnAEVaKw+fww07e65YpG3gwH0D68DiX4jUve3fxI66  
R5zpxSbJPb7+D6YAjYo8ZoAzwrqUsFHR5deeJ65Apg+0TsyGhJJ0bWCmreAx/1Ljrr0ZWJmQhEu  
yidJfk7hvLT6qsiV6eHb0072BYj1YFTsJoqqhsVkvycQRJXLY7stNyXSs1yeX0/h1LYH9RZdk  
1QJ79J2IwP6PE6A1HavbргsrGzrCsFuaLN01RqXdk0vvFiURunj61xcWFZCZoqajox1R7/iHRtZ
```

Copy this

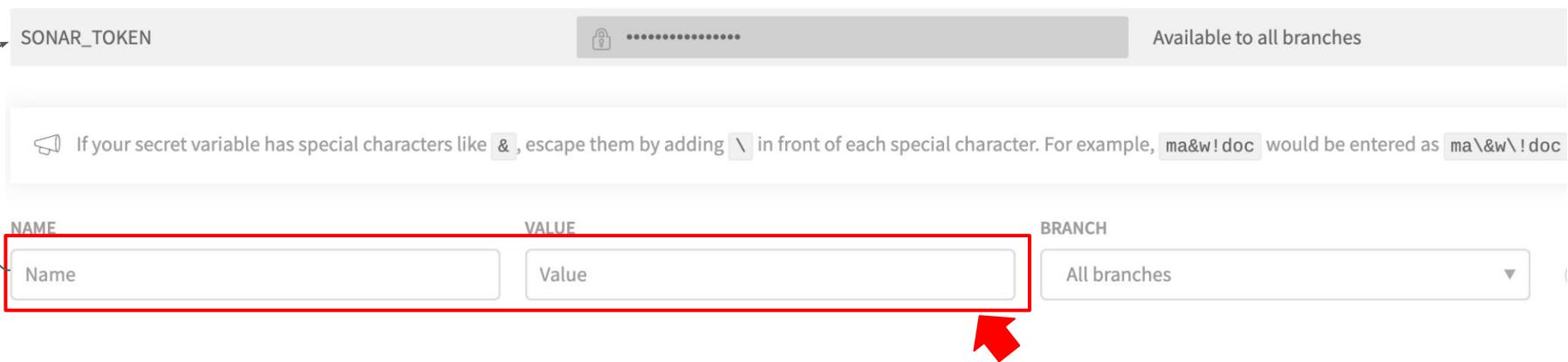
Prepare Travis Repo

Environment Variables

Customize your build using environment variables. For secure tips on generating private keys [read our documentation](#)

NAME	VALUE	BRANCH
Name	Value	All branches

If your secret variable has special characters like & , escape them by adding \ in front of each special character. For example, ma&w! doc would be entered as ma\&w\! doc



Prepare Configuration Files

- Open .travis.yml and add the highlighted lines.

```
# .travis.yml
#...
node_js:
- '10'
addons:
  sonarcloud:
    organization: "your SonarCloud organization key (e.g. swsnu)"
    token:
      secure: "encrypted value that you just copied"
before_install:
#...
script:
- coverage xml
- cd .. && sonar-scanner
```

Prepare Configuration Files

Your organization key

The screenshot shows the SonarCloud web interface. At the top, there's a navigation bar with links for 'My Projects' and 'My Issues'. A banner at the top right says 'NEW PHP 8 now supported!' with a close button. To the right of the banner is a search bar labeled 'Search for projects and files...' and a '+' icon for creating new projects. On the far right is a user profile icon with the name 'Yunmo Koo' and a dropdown arrow, followed by the URL 'https://kooyunmo.github.io/' and a red box highlighting the text 'Key: kooyunmo'.

The main content area displays a project analysis for 'swpp-travis-sonarcloud'. The analysis is marked as 'PUBLIC'. It was last analyzed on October 25, 2020, at 7:39 PM. The perspective is set to 'Overall Status' and sorted by 'Name'. There is a search bar for 'Search by project name or key'.

The analysis summary shows:

- Quality Gate:** Passed (0), Warning (0), Failed (0).
- Reliability (Bug):** 0 Bugs.
- Bugs:** 0 A (0 A)
- Vulnerabilities:** 0 A (0 A)
- Code Smells:** 0 A (0 A)
- Duplications:** 0.0%
- Files:** 213 (XS)
- Languages:** Python, JavaScript, ...

Prepare Configuration Files

- At the root of your project Create file with name: `sonar-project.properties`

```
sonar.projectKey=your-organization-key_your_project_name  
sonar.organization=your-organization-key  
sonar.projectName=your_project_name  
sonar.projectVersion=1.0  
sonar.sources=backend,frontend/src  
sonar.tests=backend,frontend/src  
sonar.exclusions=backend/**/migrations/*,frontend/src/index.tsx,frontend/src/serviceWorker.js  
sonar.test.inclusions=backend/**/tests.py,frontend/src/**/*.test.js  
sonar.python.coverage.reportPaths=backend/coverage.xml  
sonar.javascript.lcov.reportPaths=frontend/coverage/lcov.info
```

The configuration file snippet contains several placeholder values (`your-organization-key`, `your_project_name`) highlighted in yellow. Red arrows point from these placeholders to explanatory text below the code block.

[Link to the example file](#)

This has to be replaced with your SonarCloud organization key

- e.g. swsnu

These have to be replaced with your SonarCloud project name

- e.g. swppfall2022-teamX

Prepare Configuration Files

Your organization key

Your project name

The screenshot shows the SonarCloud web interface. At the top, there is a navigation bar with links for 'My Projects', 'My Issues', a 'PHP 8 now supported!' banner, 'Explore', and a search bar. Below the navigation bar, the user profile 'Yunmo Koo' is shown, along with the URL 'https://kooyunmo.github.io/' and a red box highlighting the 'Key: kooyunmo' link. The main content area displays a single project card for 'swpp-travis-sonarcloud'. The card includes a star icon, the project name, a 'NEW' badge, a 'PUBLIC' label, and a note about the last analysis date ('October 25, 2020, 7:39 PM'). The card also shows quality gate metrics: 0 bugs, 0 vulnerabilities, 0 code smells, and 0.0% duplications. A red box highlights the project name 'swpp-travis-sonarcloud'. On the left side, there are filters for 'Projects', 'Issues', 'Quality Profiles', 'Rules', 'Quality Gates', 'Members', and 'Administration'. The 'Quality Gate' section shows counts for 'Passed' (0), 'Warning' (0), and 'Failed' (0). The 'Reliability' section shows a count for 'Bugs'.

SonarCloud Demo

- Push all the changes and create a new pull request
- Travis CI will run sonar-scanner for you

A screenshot of a GitHub pull request merge button. The button is light blue with white text. At the top left is a green circular icon with a white checkmark and a pair of scissors. To its right, the text "All checks have passed" is displayed in bold, followed by "3 successful checks". On the far right is a link "Hide all checks". Below this, there are three items listed under "Details":

- ✓ SonarCloud Code Analysis Successful in 20s — Quality Gate passed
- ✓ continuous-integration/travis-ci/pr — The Travis CI build passed
- ✓ continuous-integration/travis-ci/push — The Travis CI build passed

At the bottom of the button, there is a section titled "This branch has no conflicts with the base branch" with the subtext "Merging can be performed automatically." A red box highlights the "Details" link next to the SonarCloud entry. To the right of the button, the word "Click" is written in red.

All checks have passed
3 successful checks

SonarCloud Code Analysis Successful in 20s — Quality Gate passed

continuous-integration/travis-ci/pr — The Travis CI build passed

continuous-integration/travis-ci/push — The Travis CI build passed

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request or view command line instructions.

Click

Hooray!

test #1

[Edit](#) [Open with ▾](#)

[Open](#) kooyunmo wants to merge 12 commits into `main` from `test`

Conversation 1 Commits 12 Checks 1 Files changed 8 +16,882 -18

init sonarcloud d743987

SonarCloud / SonarCloud Code Analysis succeeded 13 minutes ago in 36s

Quality Gate passed

Passed

Additional information

The following metrics might not affect the Quality Gate status but improving them will improve your project code quality.

0 Issues

- 0 Bugs
- 0 Vulnerabilities (and 0 Security Hotspots to review)
- 0 Code Smells

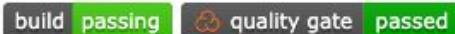
Coverage and Duplications

- No Coverage information
- No Duplication information (0.0% Estimated after merge)

[View more details on SonarCloud](#)

Add a badge to your team repo

- Copy and paste the following lines to your team repo README.md.
- Replace teamX to your team number.
 - `[![Build Status](https://travis-ci.com/swsnu/swppfall2022-teamX.svg?branch=main)](https://travis-ci.com/swsnu/swppfall2022-teamX)`
 - `[![Quality Gate Status](https://sonarcloud.io/api/project_badges/measure?project=s(wsnu)_swppfall2022-teamX&metric=alert_status)](https://sonarcloud.io/dashboard?id=s(wsnu)_swppfall2022-teamX)`
- You can attach status badges to your team repo README like this:



Today's Task (Team)

- Adopt to your team project Travis CI with ESLint, PyLint, and SonarCloud.
- Add status badges to your team repo README.
- Make a pull request and set jaewooMaeng as your reviewer.
 - Files to be included: .travis.yml, sonar-project.properties, README.md
 - Due: 10/29 (Fri) 9pm
 - example:

