



# Portfolio TNAH

Création d'un annuaire collectif de la  
promotion rassemblant toutes les pages  
personnelles des étudiants

# Déroulé de l'exercice

1

## CRÉATION D'UN SITE PERSO

Créer et publier votre  
page perso avec GitHub  
Pages

2

## CONTRIBUTION AU PORTFOLIO

Ajouter votre profil  
au *repository* commun  
via un *fork*

3

## CODE REVIEW

Examiner la *Pull  
Request* d'un·e camarade  
au portfolio de classe

4

## AUTOMATISATION AVEC LES ACTIONS

Découvrir comment  
vérifier  
automatiquement

github-page-template Public template

Pin

Watch 0

Fork 0

Star 0

Use this template

main

2 Branches

0 Tags

Go to file

t

Add file

Code

About

Settings



Segolene-Albouy [INDEX] better title

2cc3aad · 12 minutes ago

3 Commits

pages	[INIT] ajout des fichiers de template	17 minutes ago
.gitignore	[INIT] ajout des fichiers de template	17 minutes ago
LICENSE	Initial commit	1 hour ago
README.md	[INIT] ajout des fichiers de template	17 minutes ago
index.html	[INDEX] better title	12 minutes ago
script.js	[INIT] ajout des fichiers de template	17 minutes ago
style.css	[INIT] ajout des fichiers de template	17 minutes ago

README MIT license



## Publier sur GitHub Pages

### 1 Créer un repository à partir du template

1. Cliquez sur **Use this template** (en haut à droite)
2. Choisissez **Create a new repository**

Simple template for a GitHub Page website

Readme

MIT license

Activity

0 stars

0 watching

0 forks

### Releases

No releases published

[Create a new release](#)

### Packages

No packages published

[Publish your first package](#)

### Languages





## Template

Dupliquer le  
template du de  
page personnelle

## Username

Nommer le  
*repository* selon  
votre *username*

## Clone

Cloner le  
*repository* en  
local

## Modif

Mettre à jour les  
infos de votre  
page perso

## Commit

Sauvegarder les  
modifs dans un  
commit

## Push

Publier ses  
modifications sur  
GitHub



<> Code Issues 6 Pull requests Discussions Actions Projects Wiki Security Insights Settings

## General

### Access

Collaborators

Moderation options

### Code and automation

Branches

Tags

Rules

Actions

Models

Webhooks

Copilot

Environments

Codespaces

Pages

### Security

Advanced Security

Deploy keys

Secrets and variables

### Integrations

GitHub Apps

## GitHub Pages

[GitHub Pages](#) is designed to host your personal, organization, or project pages from a GitHub repository.

## Build and deployment

### Source

Deploy from a branch

### Branch

GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. [Learn more about configuring the publishing source for your site.](#)

None

Save

### Select branch

main

canvas\_label

v1.0

v2.0

✓ None

1

3

4

2

Get access to your GitHub Pages site by publishing it privately. You can host your documentation or knowledge base with members of your organization for 30 days. [Learn more about the visibility of your GitHub Pages](#)



# GitHub Page

Tout repository peut être transformé  
en site internet avec GitHub

Soit le *repository* le fichier  
[index.html](#) (à la racine), soit le  
[README.md](#) sert de page d'accueil

Portfolio-TNAH Public

Pin



Watch 0



Fork 0



Star 0



main



2 Branches



0 Tags

Go to file



&lt;&gt; Code



Segolene-Albouy [INDEX] add favicon ✓

1f2a22e · 3 hours ago 7 Commits



.github/workflows

[INIT] create basic project structure

12 hours ago



scripts

[SCRIPT] remove css generation

4 hours ago



templates

[INDEX] add favicon

3 hours ago



.gitignore

[INIT] create basic project structure

12 hours ago



README.md

[DOCS] fix link README

4 hours ago



students.json

[INIT] create basic project structure

12 hours ago



README



# Portfolio TNAH

**Annuaire collaboratif** des promotions du Master Technologies numériques appliquées à l'histoire de l'École des chartes.

## About



Un repository pour agréger les pages personnelles des étudiants TNAH

[segolene-albouy.github.io/Portfolio-T...](https://segolene-albouy.github.io/Portfolio-T...)

Readme

Activity

0 stars

0 watching

0 forks

## Releases

No releases published

[Create a new release](#)

## Packages

No packages published

[Publish your first package](#)

## Deployments 7

**github-pages** 3 hours ago



## Publier

Activer GitHub  
page sur la  
branche main

## Fork

Créer une copie  
du *repository* de  
portfolio

## Clone

Cloner le  
*repository* en  
local

## Commit

Ajouter vous à  
students.json avec  
une erreur

## Push

Publier votre  
code sur votre  
fork GitHub

## Pull Request

Créer une *pull*  
*request* de votre  
*fork*







## Review

Attribuer la  
*review* de la PR à  
un camarade

## Comment

Signaler l'erreur  
commise dans  
votre *review*

## Corriger

Pusher un commit  
de correction du  
JSON

## Valider

Valider la PR si  
l'erreur a bien  
été corrigée

## Merge

M'appeler pour  
achever la fusion  
de la PR





# GitHub Actions

Les actions permettent de définir des *workflows* automatiques sur le code du *repository* (***test***, ***build***, etc.)

Elles sont définies au format yaml  
dans **[.github/workflows](#)**

[.github/workflows/build.yml](#)



*name: Valider et Générer le Portfolio*

*on:*

*push:*

*branches: [ main ]*

*pull\_request:*

*branches: [ main ]*

*jobs:*

*validate-and-build: # Nom du job*

*runs-on: ubuntu-latest*

*steps:*

- name: Récupérer le code*
- name: Installer Python*
- ...*

<> Code Issues Pull requests **Actions** Projects Wiki Security Insights Settings

← Valider et Générer le Portfolio

✓ [DOCS] update README #3

Re-run all jobs



Summary

Jobs

✓ validate-and-build

Run details

Usage

Workflow file

## validate-and-build

succeeded 4 hours ago in 8s

Search logs



- > ✓ Set up job 1s
- > ✓ Récupérer le code 1s
- > ✓ Installer Python 0s
- > ✓ Valider le JSON 1s
- > ✓ Générer le HTML 0s
- > ✓ Déployer sur GitHub Pages 1s
- > ✓ Post Installer Python 0s
- > ✓ Post Récupérer le code 1s
- > ✓ Complete job 0s

# Autoriser une actions à modifier son repository

<> Code Issues Pull requests Actions Projects Wiki Security Insights **Settings**

General

Access

Collaborators

Moderation options

Code and automation

Branches

Tags

Rules

Actions

General

Runners

Models

Webhooks

Copilot

Environments

Codespaces

Pages

Security

## Actions permissions

### ☒ Allow all actions and reusable workflows

Any action or reusable workflow can be used, regardless of who authored it or where it is defined.

### ☐ Disable actions

The Actions tab is hidden and no workflows can run.

### ☐ Allow Segolene-Albouy actions and reusable workflows

Any action or reusable workflow defined in a repository within Segolene-Albouy can be used.

### ☐ Allow Segolene-Albouy, and select non-Segolene-Albouy, actions and reusable workflows

Any action or reusable workflow that matches the specified criteria, plus those defined in a repository within Segolene-Albouy, can be used. [Learn more about allowing specific actions and reusable workflows to run.](#)

### ☐ Require actions to be pinned to a full-length commit SHA

Save

## Workflow permissions

Choose the default permissions granted to the GITHUB\_TOKEN when running workflows in this repository. You can specify more granular permissions in the workflow using YAML. [Learn more about managing permissions.](#)

### ☒ Read and write permissions

Workflows have read and write permissions in the repository for all scopes.

### ☐ Read repository contents and packages permissions

Workflows have read permissions in the repository for the contents and packages scopes only.

Choose whether GitHub Actions can create pull requests or submit approving pull request reviews.

### ☐ Allow GitHub Actions to create and approve pull requests

Save

# Portfolio des promotions TNAH

Découvrez les sites personnels de vos camarades

1 étudiant dans l'annuaire



**Sékolène Albouy**

Promo 2019

Matière préférée : Git

**Voir le site →**

Généré automatiquement via GitHub Actions  
Pour vous ajouter, créez une Pull Request !



# { .. Partie 1

## Init

Un des deux crée un *repository* sur GitHub

## Collab

Il ajoute l'autre dans les collaborateurs

## Clone

Chacun *clone* le *repository* sur son ordinateur

## Branch

Depuis VSCode, chacun crée une branche

## Commit

Création de fichiers, ajout et *commit*

## Push

Publier sur GitHub la branche créée





# { .. Partie 2

## Pull

Récupération la  
branche de son·a  
camarade en local

## Switch

Passer sur la  
branche de son·a  
camarade

## Commit

Chacun effectue  
un commit dans la  
branche

## Push

Chacun *push* la  
branche modifiée  
sur le *repo*  
distant

## Pull request

Création d'une PR  
de la branche  
vers **main**

## Review

Chacun passe en  
revue la PR de  
l'autre et merge





# { .. Partie 3

## Fork

L'un crée un *fork* du *repository* d'un autre groupe

## Branch

Chacun crée une branche depuis main sur son *fork*

## Commit

Chacun effectue des modifs sur les mêmes lignes de fichiers

## Push

Chacun publie sa branche sur son *fork*

## Pull Request

Création d'une PR des branches des *fork* vers main du *repo* d'origine

## Review

Les binôme passe en revue leur PR et en merge **une seule**





CONFLIT



**L'autre *Pull Request* ne peut  
plus être mergée  
car elle présente  
un conflit !**



# { .. Partie 4

## Sync

Récupération du  
code *repo* original  
sur son *fork*

## Pull

Récupération de  
main du *fork* en  
local

## Merge

Fusion de main  
dans sa branche  
et résolution des  
conflits

## Push

Pousser la branche  
mise à jour sur le  
*fork*

## Fusion

Sur la page de  
*Pull Request*,  
finaliser la  
fusion





# Comment aborder un nouveau projet de développement ?

Tout projet commence par une  
phase de réflexion

Un bon plan = gain de temps à  
long terme

# Étapes au démarrage



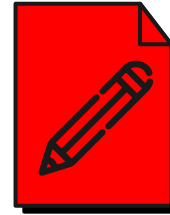
## COMPRENDRE LE BESOIN

Qui va utiliser ?  
Pour quoi faire ?  
Dans quel contexte ?



## PLANIFIER AVANT DE CODER

Découper en petites tâches  
Identifier les priorités  
Prévoir les difficultés



## DOCUMENTER LES DÉCISIONS

Choix techniques  
Planning  
prévisionnel  
Points à valider

# Cahier des charges

Permet de  
dérouter  
toutes les  
implications  
du projet

**Vision**

Base sur  
laquelle  
discuter avec  
les "clients"

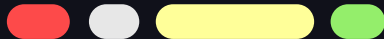
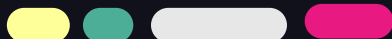
**Référence**

Permet de  
comparer les  
réalisations  
avec de qui  
était attendu

**Résultat**

- 1 Lister toutes les fonctionnalités
- 2 Découper en petites tâches
- 3 Identifier les dépendances
- 4 Prévoir les tests nécessaires
- 5 Valider avec le “client”

# Exemple : Export de données



## Format

Excel, CSV, JSON ? Structure des fichiers ?

## Interface

Page dédiée ? Bouton d'export ? Script ?

## Contenu

Quelles données inclure ? exclure ? Données personnelles ? Traçabilité ?





# Choisir une techno

Si on vous laisse choisir c'est  
que le client/manager/PI n'a pas  
d'avis : c'est une opportunité !

Pas de "mauvais" choix, que des  
compromis

# Les options

Plus rapide à  
développer  
Moins de  
surprises

**Je connais  
déjà**

Meilleur  
support  
communautaire  
& LLM bien  
entraînés

**Techno la  
plus répandue**

Plus  
d'inconnu  
mais plus  
intéressant  
pour vous

**Je veux  
apprendre**

# Exemple : créer un site



## Flask

Rapide à mettre en  
place  
Solution simple



## Wordpress

Très répandu  
Beaucoup de ressources  
en ligne



## Svelte

Nouveau et moderne  
Moins de  
documentation

## C'est quoi les technos populaires ?



- 1 Regarder les étoiles GitHub ★
- 2 Faire une recherche (Medium, etc.)
- 3 Demander à ChatGPT "Quelle stack pour un projet de..."



# **Dette technique**

Évaluer si un choix sera  
préjudiciable à l'avenir (e.g.  
Cobol)

C'est normal de réécrire du code  
qui a à peine 6 mois

# Technology Impact Decision Evaluation

Chaque critère est noté de 1 à 5 ★

**T**

**I**

**D**

**E**

## ***Technical Fit***

Compatible avec  
l'existant ?  
Répond aux  
besoins ?

## ***Implementation Cost***

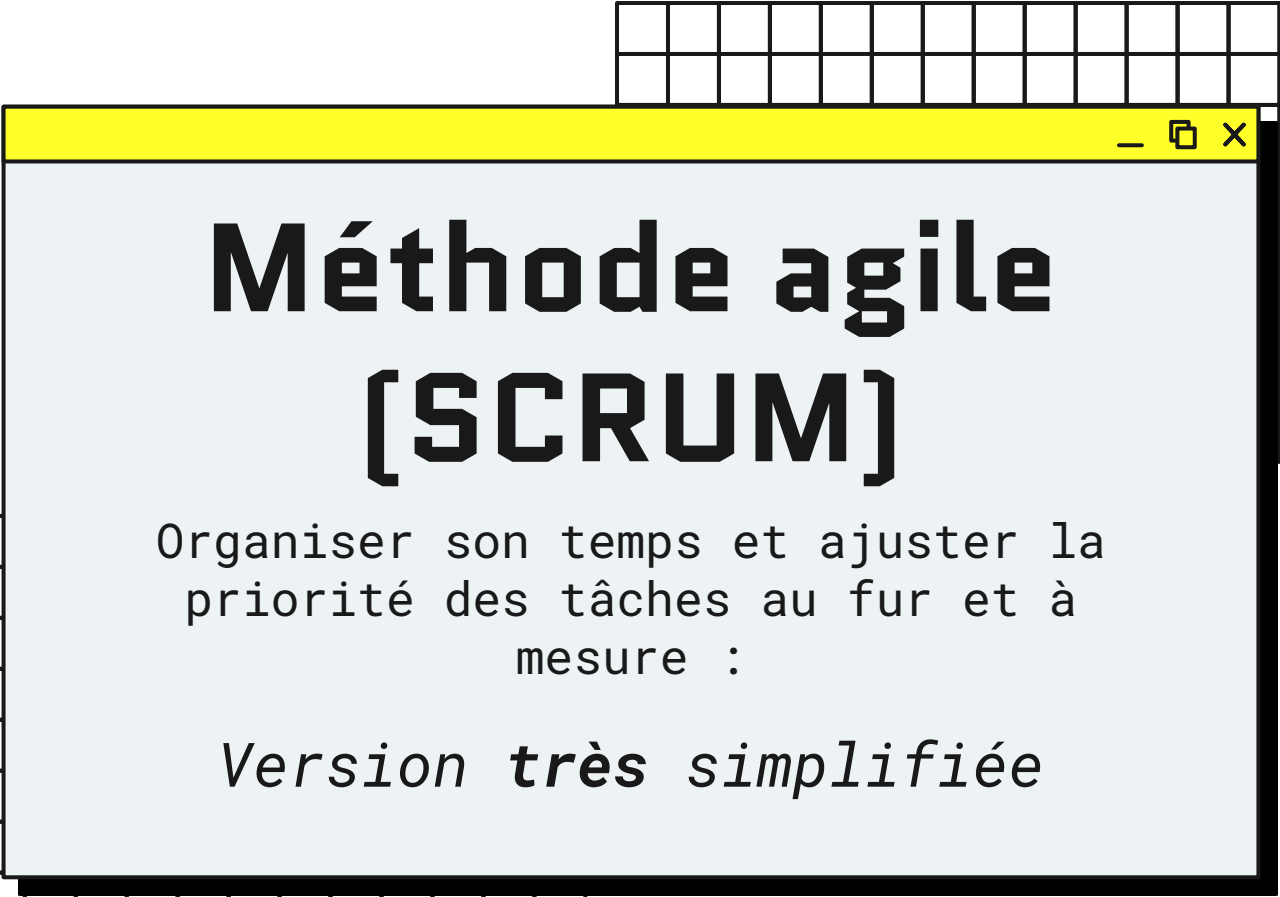
Directs licences,  
infrastructure, etc.  
Indirects formation,  
Maintenance, etc,

## ***Development Speed***

Compétence interne,  
temps de formation  
nécessaire,  
ressources dispo

## ***Evolution Capability***

Scalabilité,  
adaptabilité,  
facilité de  
maintenance



# Méthode agile [SCRUM]

Organiser son temps et ajuster la  
priorité des tâches au fur et à  
mesure :

*Version très simplifiée*



SPRINT (~semaine)



BACKLOG

POINT EN  
ÉQUIPE



Ken Schwaber & Jeff Sutherland

# Le Guide Scrum

Le guide de Référence de Scrum : Les règles du jeu

REVISED AND UPDATED



# SCRUM

## The Art of Doing Twice the Work in Half the Time

10<sup>th</sup>  
Anniversary  
Edition

**JEFF SUTHERLAND**

Co-creator of Scrum

**J.J. SUTHERLAND**

- 1 On met dans *pending* les tâches les plus prioritaires
- 2 On ne met pas trop de tâches : rester réaliste
- 3 On ne fait QUE les tâches dans *pending*
- 4 On fait un point chaque fin de sprint pour se réattribuer des tâches, mettre à jour le *backlog*, montrer ce qu'on a fait, etc.

# Importance du feedback



## Rapide

Tests automatiques  
Démonstration du code  
lors de réunion d'  
équipe



## Intermédiaire

Retour utilisateur,  
*Bug report*,  
Suggestions  
d'amélioration



## Long terme

Métriques globales  
d'utilisation,  
retour d'expérience