TP 9 – Test Cypress E2E sur GitLab

Brahim HAMDI

Préparation du projet sur GitLab.com

1. Git cloner le dépôt suivant sur votre machine locale : https://github.com/brahimhamdi/cypress-gitlab-react

```
brahim@Training:~/TP9$ git clone https://github.com/brahimhamdi/cypress-gitlab-react
Clonage dans 'cypress-gitlab-react'...
remote: Enumerating objects: 40, done.
remote: Counting objects: 100% (40/40), done.
remote: Compressing objects: 100% (32/32), done.
remote: Total 40 (delta 6), reused 40 (delta 6), pack-reused 0
Réception d'objets: 100% (40/40), 184.07 Kio | 331.00 Kio/s, fait.
Résolution des deltas: 100% (6/6), fait.
brahim@Training:~/TP9$
```

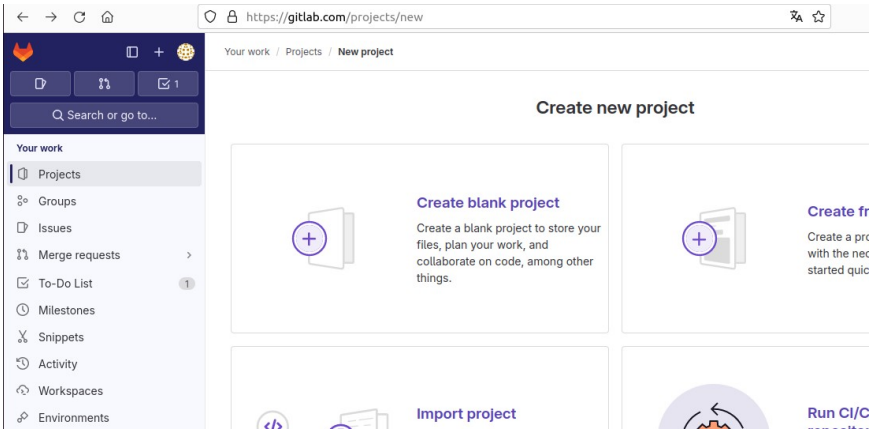
   - Découvrir le dépôt cloné (fichiers, branches, commits, …)

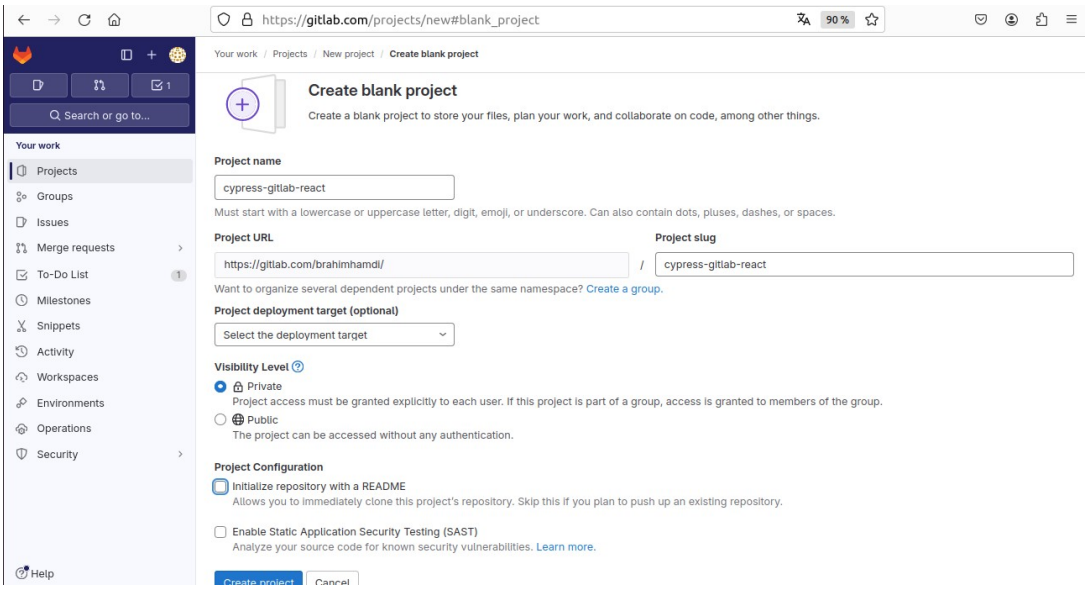
```
brahim@Training:~/TP9$ cd cypress-gitlab-react/
brahim@Training:~/TP9/cypress-gitlab-react$ ls -a
.  ..  cypress  cypress.config.js  .git  .gitignore  package.json  package-lock.json  public  README.md  src
brahim@Training:~/TP9/cypress-gitlab-react$
brahim@Training:~/TP9/cypress-gitlab-react$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/master
brahim@Training:~/TP9/cypress-gitlab-react$
brahim@Training:~/TP9/cypress-gitlab-react$ git log
commit e0e234d19a6840d78cbdb2566e221f54711d8057 (HEAD -> master, origin/master, origin/HEAD)
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Tue Jan 9 23:37:19 2024 +0100

    modif cypress.config.js

commit b40092facfd39eba49e8b9e0afcf19881f75ce18
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Sun Jan 7 07:41:09 2024 +0100

    cypress test create & cypress config add

commit b08b6ac252d11f9f2e9856b47ffff24e044bac01
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Sun Jan 7 07:40:18 2024 +0100

    modif package*

commit b79b310f283d0caf7b2b2474ca21678907551107
Author: Brahim HAMDI <brahim.hamdi.consult@gmail.com>
Date:   Sun Jan 7 07:05:10 2024 +0100

    Initialize project using Create React App
brahim@Training:~/TP9/cypress-gitlab-react$
```

   - Supprimer le lien origin

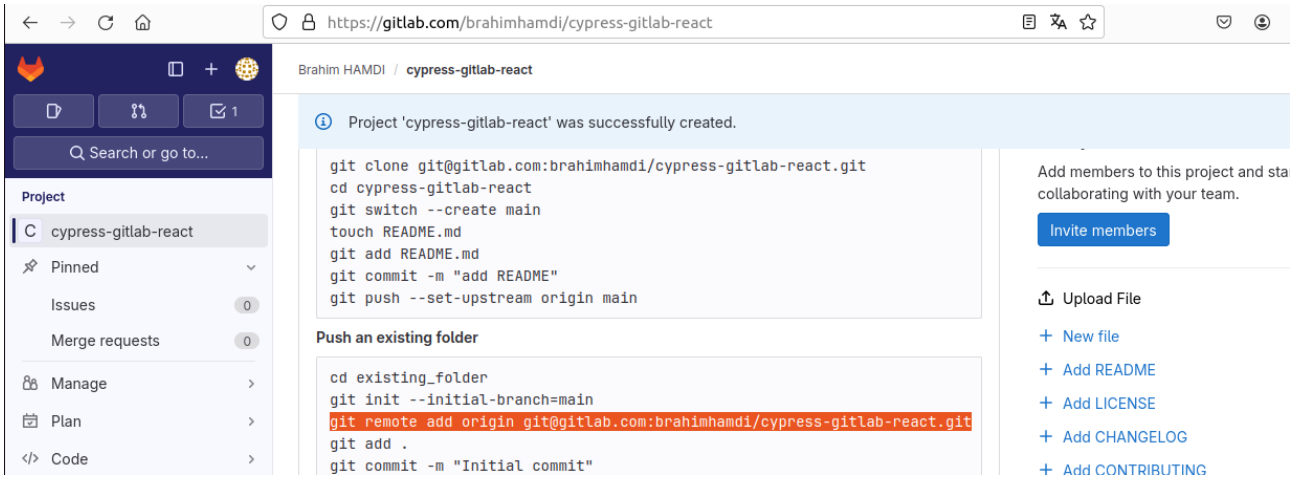
```
brahim@Training:~/TP9/cypress-gitlab-react$ git remote -v
origin  https://github.com/brahimhamdi/cypress-gitlab-react (fetch)
origin  https://github.com/brahimhamdi/cypress-gitlab-react (push)
brahim@Training:~/TP9/cypress-gitlab-react$
brahim@Training:~/TP9/cypress-gitlab-react$ git remote remove origin
brahim@Training:~/TP9/cypress-gitlab-react$
brahim@Training:~/TP9/cypress-gitlab-react$ git remote -v
brahim@Training:~/TP9/cypress-gitlab-react$
```

2. Créer un nouveau projet blanc nommé « ***cypress-gitlab-react*** » sur http://gitlab.com en décochant la case « ***Initialize repository with a README*** »





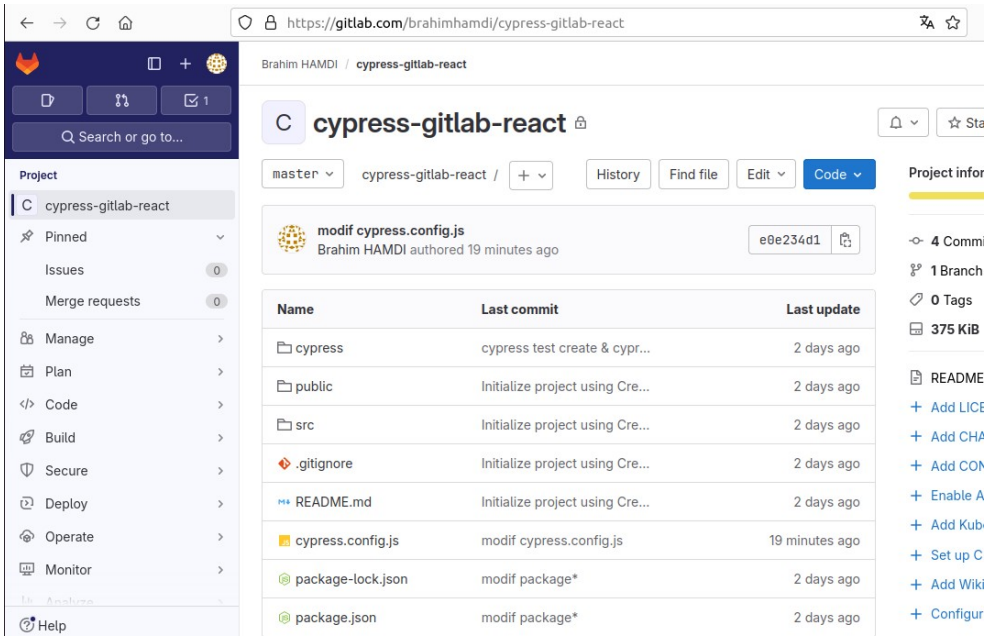
3. Lier le nouveau projet à votre dépôt git local avec le nom origin (utiliser le lien ssh)




```
brahim@Training:~/TP9/cypress-gitlab-react$ git remote -v
brahim@Training:~/TP9/cypress-gitlab-react$ git remote add origin git@gitlab.com:brahimhamdi/cypress-gitlab-react.git
brahim@Training:~/TP9/cypress-gitlab-react$
brahim@Training:~/TP9/cypress-gitlab-react$ git remote -v
origin  git@gitlab.com:brahimhamdi/cypress-gitlab-react.git (fetch)
origin  git@gitlab.com:brahimhamdi/cypress-gitlab-react.git (push)
brahim@Training:~/TP9/cypress-gitlab-react$
```

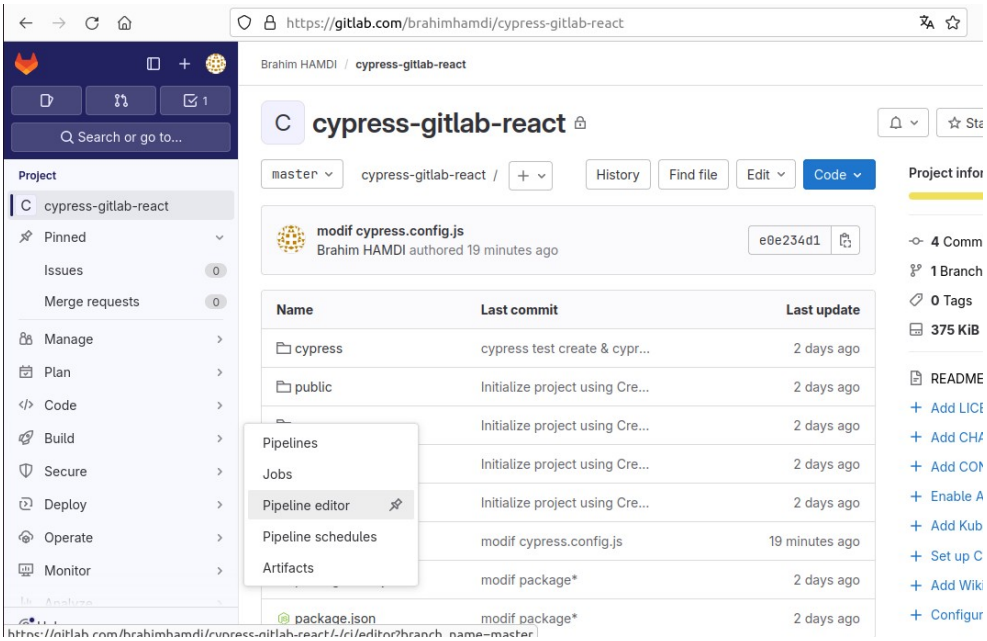
4. Pousser la branche master locale vers le projet distant sur gitlab.com, et vérifier par la suite que le push a bien passé.

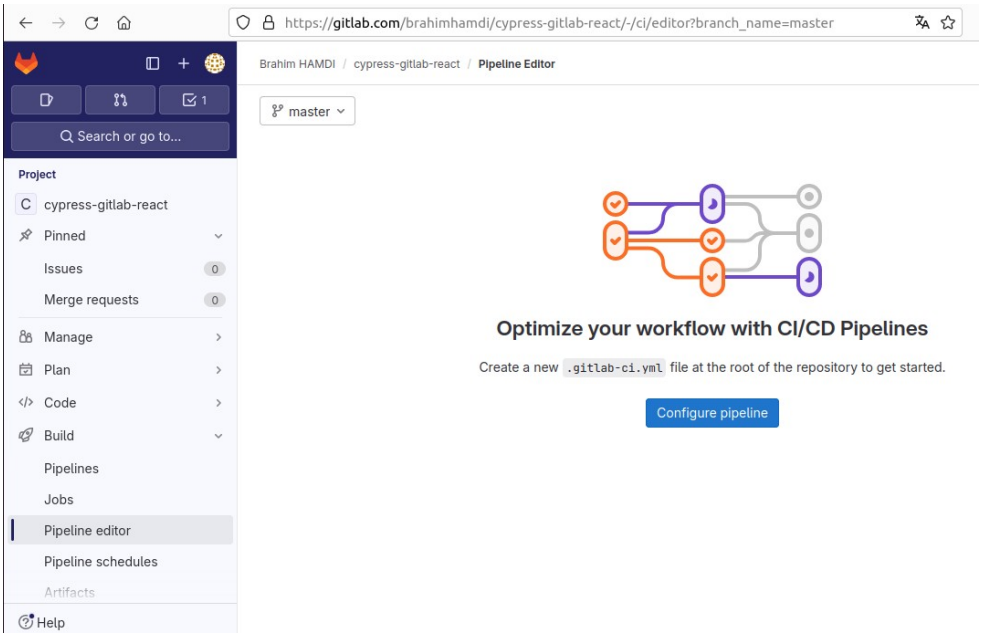
```
brahim@Training:~/TP9/cypress-gitlab-react$ git push origin master
Énumération des objets: 40, fait.
Décompte des objets: 100% (40/40), fait.
Compression par delta en utilisant jusqu'à 8 fils d'exécution
Compression des objets: 100% (32/32), fait.
Écriture des objets: 100% (40/40), 184.07 Kio | 184.07 Mio/s, fait.
Total 40 (delta 6), réutilisés 40 (delta 6), réutilisés du pack 0
To gitlab.com:brahimhamdi/cypress-gitlab-react.git
 * [new branch]      master -> master
brahim@Training:~/TP9/cypress-gitlab-react$
```

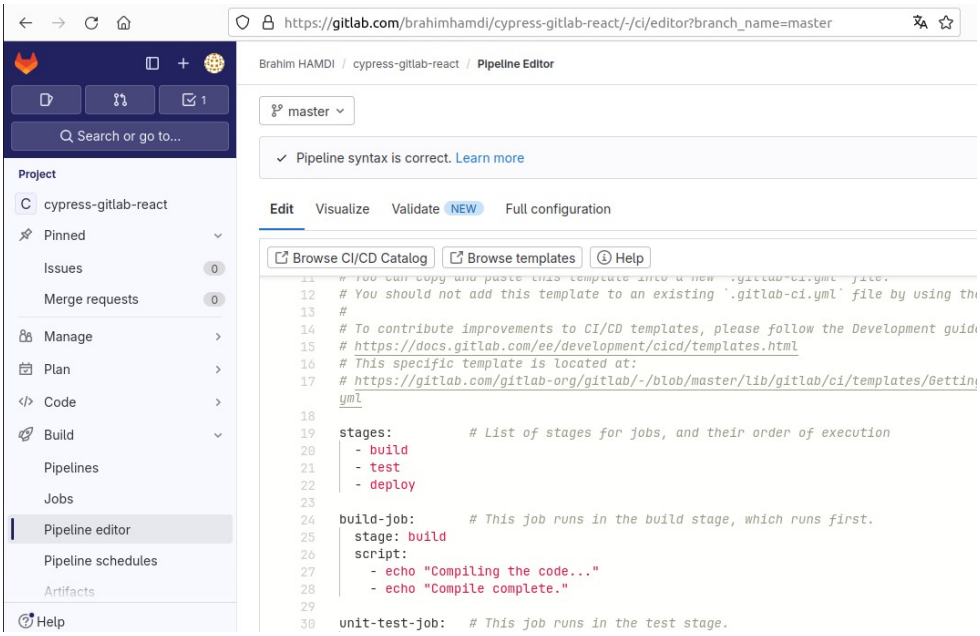


Un simple pipeline GitLab CI

5. Sur l'interface de gitlab.com, créer le fichier ***.gitlab-ci.yml*** sur la racine de votre projet.







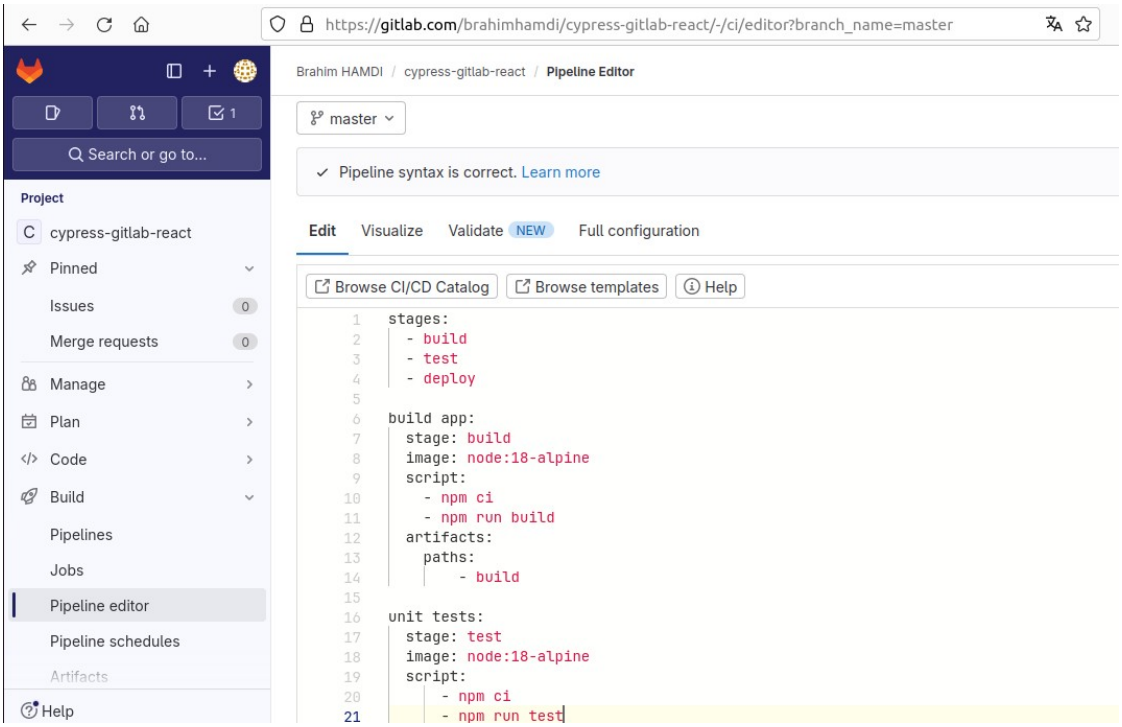
6. Pour le moment, on va créer 2 stages dans le manifest ***.gitlab-ci.yml*** :
   - **build** : Nous allons installer les dépendances npm et « builder » l'application Web. Il s'agit de la version de test. A la fin de build, nous allons la sauvegarder en tant qu'artefact pour plus tard. Cet artefact sera disponible pour les stages suivants.

   - **test** : Au cours de ce stage, nous allons exécuter tous les tests unitaires. Dans le cas de cette application, il s'agit simplement d'un test de composant React.js (cela n'a pas vraiment d'importance).

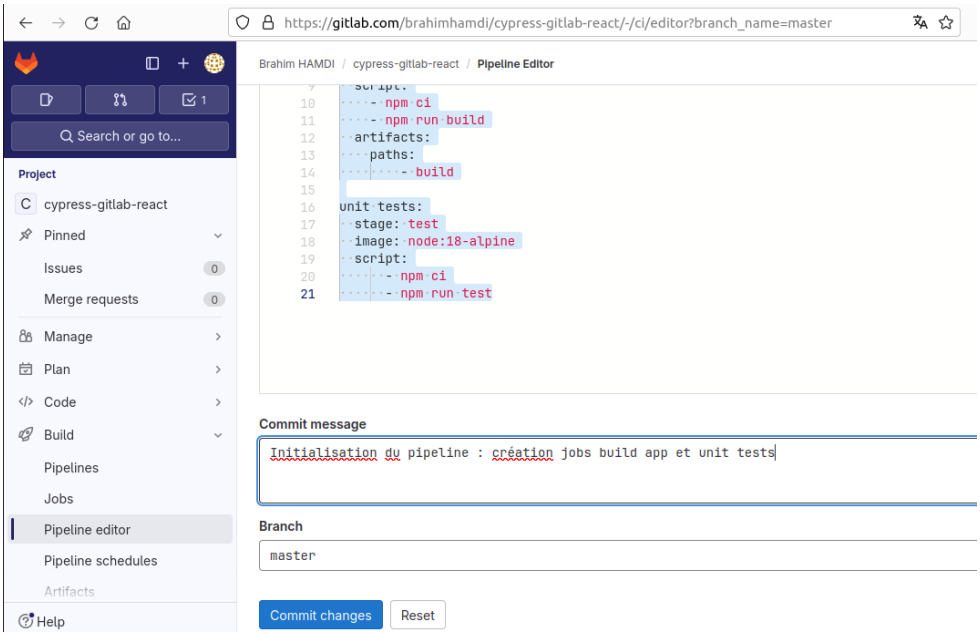
   Copier le contenu suivant dans le fichier ***.gitlab-ci.yml*** :

```
stages:
  - build
  - test

build app:
  stage: build
  image: node:18-alpine
  script:
    - npm ci
    - npm run build
  artifacts:
    paths:
       - build

unit tests:
  stage: test
  image: node:18-alpine
```

script:
- npm ci
- npm run test



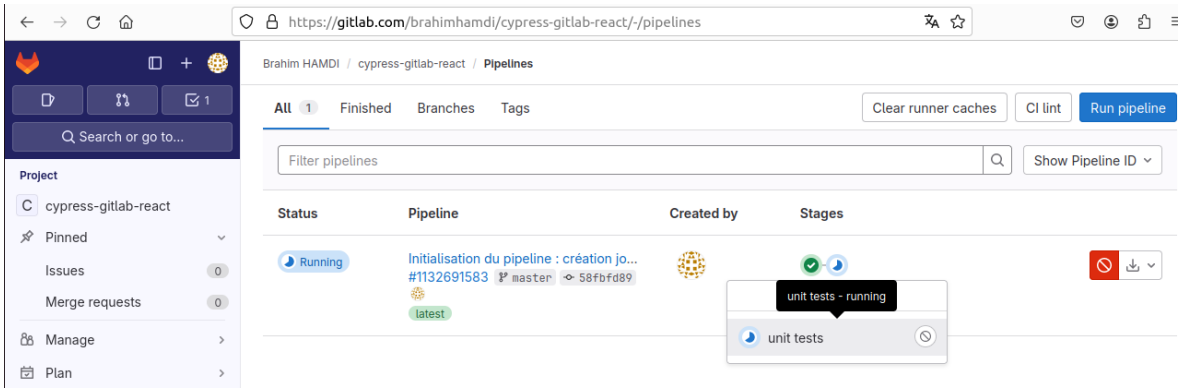
- Créer un commit avec le message « Initialisation du pipeline : création jobs build app et unit tests »

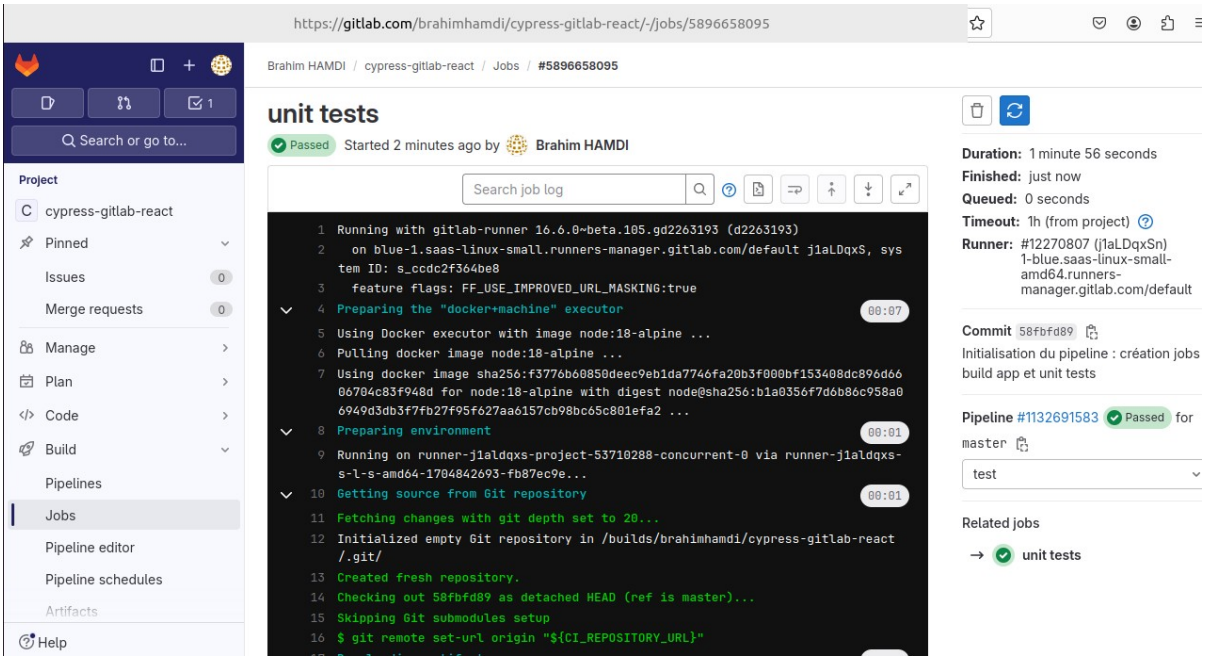


En cliquant sur le bouton « Commit changes », l'exécution du pipeline se lance automatiquement.

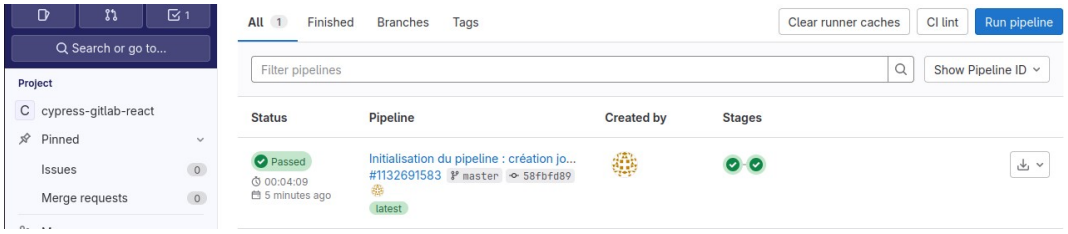
- Cliquer sur le menu (à gauche) ***Build > Pipelines*** pour observer l'avancement de l'exécution des jobs.

- Cliquer sur le job de tests unitaire pour voir la sortie de console et voir les détails (runner, pull image docker, création conteneur, exécution du script, ...)

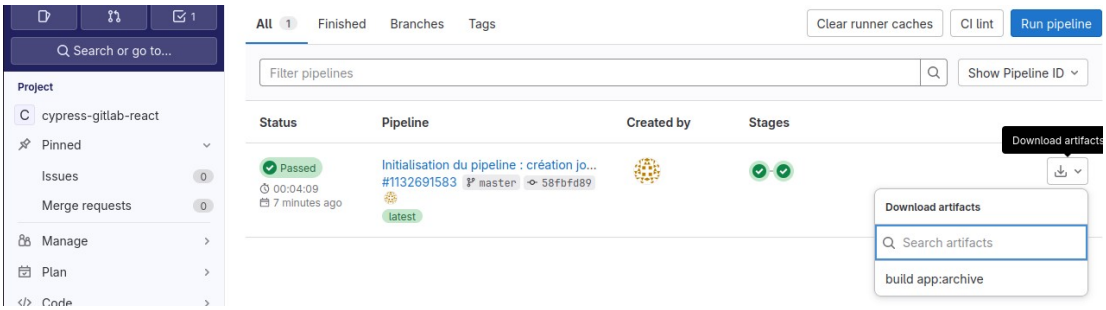




- Retourner à l'interface des pipelines et vérifier si l'exécution du pipeline (les 2 stages jusqu'à maintenant) a passé.



- Sur la même interface, vérifier s'il y a des artifacts générés.



Télécharger les artifacts.

Ajout du test e2e de Cypress au pipeline GitLab CI

7. Dans cette partie nous allons ajouter le job « ***e2e tests*** » au stage **test** pour exécuter les tests e2e de cypress.

   Ce que vous devez retenir, c'est qu'avant de pouvoir exécuter les tests Cypress, nous aurons besoin d'un serveur d'exécution avec l'application. Nous allons donc installer et démarrer le serveur en ajoutant les 2 scripts suivants :
   npm install -g serve
   serve -s build &

   La commande suivante avec les tests Cypress peut démarrer avant que le serveur ne soit en cours d'exécution. Pour nous assurer que le serveur a démarré, nous utiliserons le ***wait-on***.

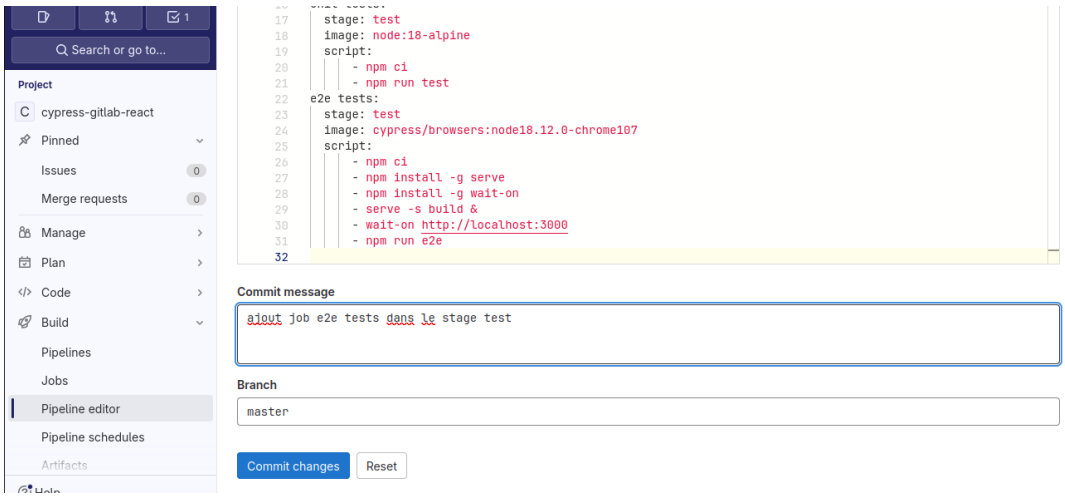
   L'image ***cypress/base*** possède les dépendances du système d'exploitation requises par Cypress.

   Donc le nouveau job « ***e2e tests*** » peut être comme suit :

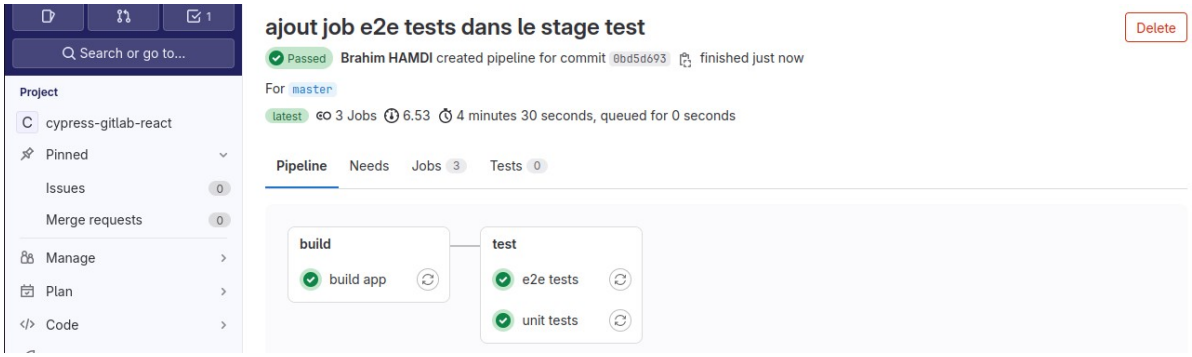
```
e2e tests:
  stage: test
  image: cypress/browsers:node18.12.0-chrome107
  script:
     - npm ci
     - npm install -g serve
```

- npm install -g wait-on
- serve -s build &
- wait-on http://localhost:3000
- npm run e2e

- Ajouter ces lignes au fichier ***.gitlab-ci.yml***, et créer un commit avec le message « ajout job e2e tests dans le stage test »

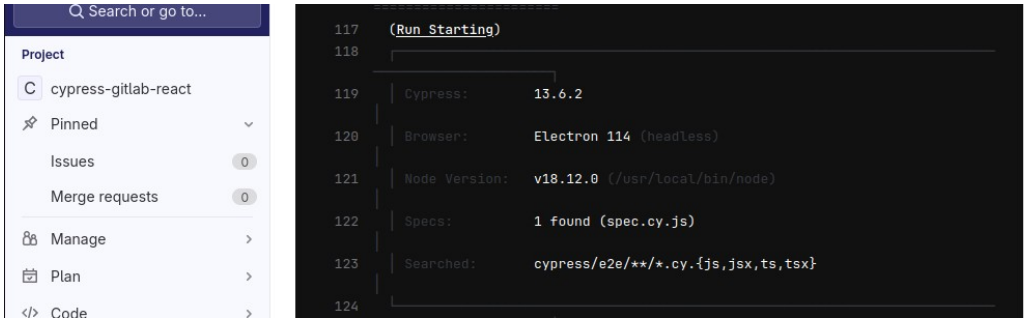


- Suivez l'exécution du pipeline, et vérifier que le nouveau job a passé avec succès.



Utiliser Chrome ou Firefox pour exécuter les tests Cypress

8. Si vous examinez attentivement les journaux d'exécution des tâches, vous remarquerez peut-être que les tests Cypress ont été exécutés avec Electron 114.



Bien qu'Electron fasse du bon travail en termes de rendu de notre application Web, nous souhaitons la tester dans un vrai navigateur que les utilisateurs réguliers pourraient utiliser, comme **Chrome** ou **Firefox**.

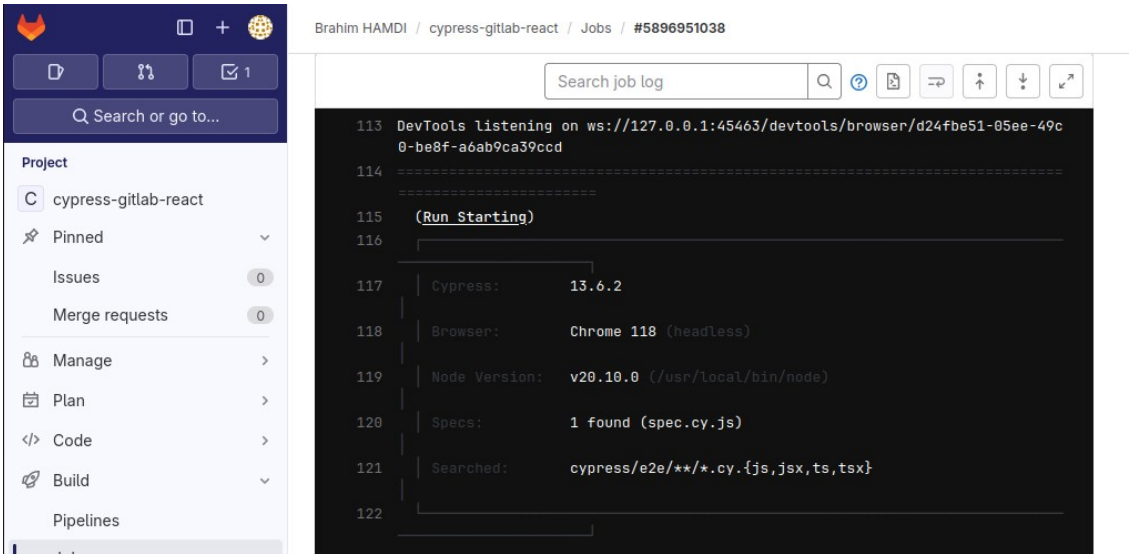
- Editer le fichier ***package.json*** et spécifier Chrome comme navigateur. Faire le changement sur votre dépôt local puis pousser vers gitlab.com

```
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject",
    "e2e": "cypress run --browser chrome --headless"
  },
  "eslintConfig": {
    "extends": [
      "react-app"
```

```
brahim@Training:~$ cd TP9/cypress-gitlab-react/
brahim@Training:~/TP9/cypress-gitlab-react$ vim package.json
brahim@Training:~/TP9/cypress-gitlab-react$ git status
Sur la branche master
Modifications qui ne seront pas validées :
  (utilisez "git add <fichier>..." pour mettre à jour ce qui sera validé)
  (utilisez "git restore <fichier>..." pour annuler les modifications dans le répertoire de travail)
        modifié :         package.json

aucune modification n'a été ajoutée à la validation (utilisez "git add" ou "git commit -a")
brahim@Training:~/TP9/cypress-gitlab-react$
brahim@Training:~/TP9/cypress-gitlab-react$ git commit -am "package.json: exécuter test e2e sur navigateur Chrome"
[master 654ff89] package.json: exécuter test e2e sur navigateur Chrome
 1 file changed, 1 insertion(+), 1 deletion(-)
```

```
brahim@Training:~/TP9/cypress-gitlab-react$ git pull origin master
Depuis gitlab.com:brahimhamdi/cypress-gitlab-react
 * branch            master     -> FETCH_HEAD
astuce: Vous avez des branches divergentes et vous devez spécifier comment
astuce: les réconcilier. Vous pouvez le faire en lançant une des
astuce: commandes suivantes avant votre prochain tirage :
astuce:
astuce:   git config pull.rebase false  # fusion (stratégie par défaut)
astuce:   git config pull.rebase true   # rebasage
astuce:   git config pull.ff only       # avance rapide seulement
astuce:
astuce: Vous pouvez remplacer "git config" par "git config --global" pour que
astuce: ce soit l'option par défaut pour tous les dépôts. Vous pouvez aussi
astuce: passer --rebase, --no-rebase ou --ff-only sur la ligne de commande pour
astuce: remplacer à l'invocation la valeur par défaut configurée.
fatal: Besoin de spécifier comment réconcilier des branches divergentes.
brahim@Training:~/TP9/cypress-gitlab-react$ git config pull.rebase false
brahim@Training:~/TP9/cypress-gitlab-react$
brahim@Training:~/TP9/cypress-gitlab-react$ git pull origin master
Depuis gitlab.com:brahimhamdi/cypress-gitlab-react
 * branch            master     -> FETCH_HEAD
Merge made by the 'ort' strategy.
 .gitlab-ci.yml | 31 ++++++++++++++++++++++++++++++
 1 file changed, 31 insertions(+)
 create mode 100644 .gitlab-ci.yml
brahim@Training:~/TP9/cypress-gitlab-react$ git push origin master
Énumération des objets: 9, fait.
Décompte des objets: 100% (8/8), fait.
Compression par delta en utilisant jusqu'à 8 fils d'exécution
Compression des objets: 100% (5/5), fait.
Écriture des objets: 100% (5/5), 615 octets | 615.00 Kio/s, fait.
Total 5 (delta 3), réutilisés 0 (delta 0), réutilisés du pack 0
To gitlab.com:brahimhamdi/cypress-gitlab-react.git
   0bd5d69..3b3a6d4  master -> master
brahim@Training:~/TP9/cypress-gitlab-react$
```

- Suite au push, le pipeline sur gitlab.com s'exécute automatiquement. Observer l'exécution du pipeline et vérifier que l'exécution du test a été fait sur le navigateur Chrome.



N.B. : Si test e2e n'a pas exécuté sur navigateur Chrome, c'est l'image docker ne supporte pas la version Chrome installée. Dans ce cas, chercher sur dockerhub.com une autre version de l'image docker (par exemple : cypress/browsers:node-20.10.0-**chrome-118**.0.5993.88-1-ff-118.0.2-edge-118.0.2088.46-1)

Publication du rapport JUnit de Cypress

Générer un rapport **JUnit** à partir de Cypress est possible, mais nous devons apporter quelques modifications aux paramètres passés à Cypress.

9. En utilisant l'option **--reporter**, nous pouvons spécifier les rapports que nous souhaitons générer.
   Adaptez le script e2e dans le fichier **package.json** comme suit, puis faire un commit :

   "e2e": "cypress run --browser chrome --headless --reporter junit"

```
    "web-vitals": "^2.1.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject",
    "e2e": "cypress run --browser chrome --headless --reporter junit"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
```

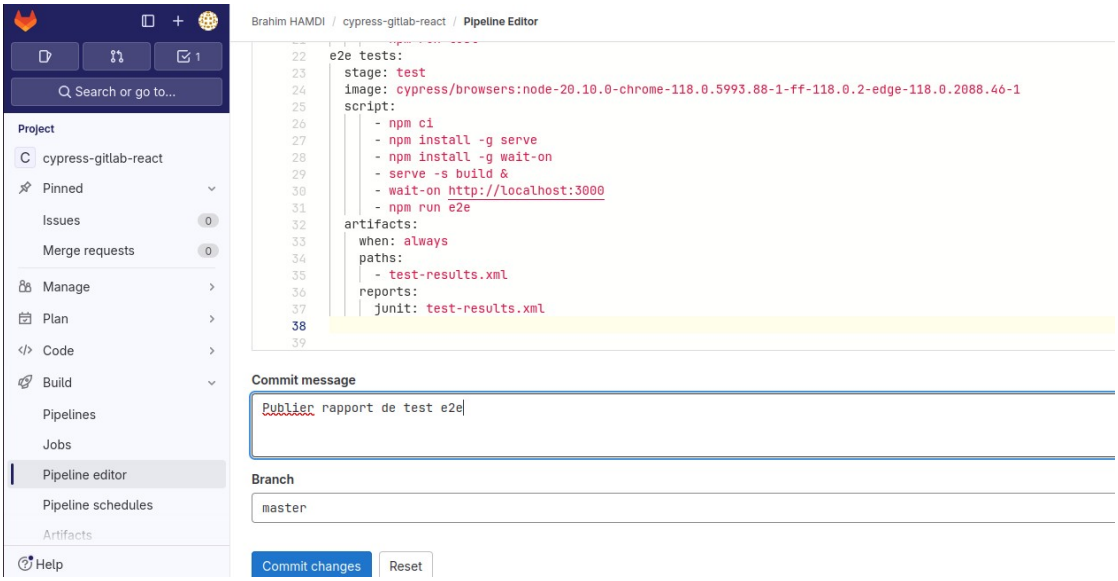
```
brahim@Training:~/TP9/cypress-gitlab-react$ vim package.json
brahim@Training:~/TP9/cypress-gitlab-react$ git pull origin master
Depuis gitlab.com:brahimhamdi/cypress-gitlab-react
 * branch            master     -> FETCH_HEAD
Déjà à jour.
brahim@Training:~/TP9/cypress-gitlab-react$ git commit -am "package.json: générer rapport json"
[master de6af68] package.json: générer rapport json
 1 file changed, 1 insertion(+), 1 deletion(-)
brahim@Training:~/TP9/cypress-gitlab-react$
brahim@Training:~/TP9/cypress-gitlab-react$ git push origin master
Énumération des objets: 5, fait.
Décompte des objets: 100% (5/5), fait.
Compression par delta en utilisant jusqu'à 8 fils d'exécution
Compression des objets: 100% (3/3), fait.
Écriture des objets: 100% (3/3), 341 octets | 341.00 Kio/s, fait.
Total 3 (delta 2), réutilisés 0 (delta 0), réutilisés du pack 0
To gitlab.com:brahimhamdi/cypress-gitlab-react.git
   74d39e7..de6af68  master -> master
brahim@Training:~/TP9/cypress-gitlab-react$
```

Avec cette option, Cypress générera un fichier nommé ***test-results.xml*** sur la racine du projet.

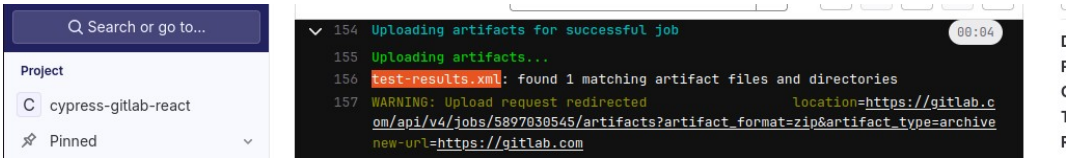
10. Puisque nous connaissons le nom et le chemin du fichier, il nous suffit également de l'indiquer à GitLab.

    Publier le fichier **test-results.xml** en tant qu'artefact et indiquer à GitLab qu'il s'agit d'un rapport Junit en ajoutant les lignes suivantes au job **e2e tests** :

```
artifacts:
  when: always
  paths:
    - test-results.xml
  reports:
    junit: test-results.xml
```



- Observer l'exécution du pipeline et vérifier que la génération du rapport de test à la fin.



Exécution des tests Cypress sur un environnement de test

Jusqu'à présent, nous n'avons exécuté des tests que sur une version locale de notre application. Très souvent, les applications Web dépendent d'autres services et nous ne pouvons pas recréer l'environnement nécessaire pour exécuter l'application dans GitLab CI. Nous devons déployer l'application quelque part.
Très probablement, vous utilisez déjà différents environnements de développement, de staging et de production.
Dans cette partie, on va déployer et tester l'application sur un environnement de staging représenté par un serveur sur cloud.

11. Étant donné que GitLab, CI dispose déjà d'une prise en charge intégrée des environnements et expose déjà la variable d'environnement **CI_ENVIRONMENT_URL**.
    Remplaçer ***baseUrl*** dans le fichier de configuration ***cypress.config.js*** par ce qui suit :

    baseUrl: process.env.CI_ENVIRONMENT_URL || 'http://localhost:3000',

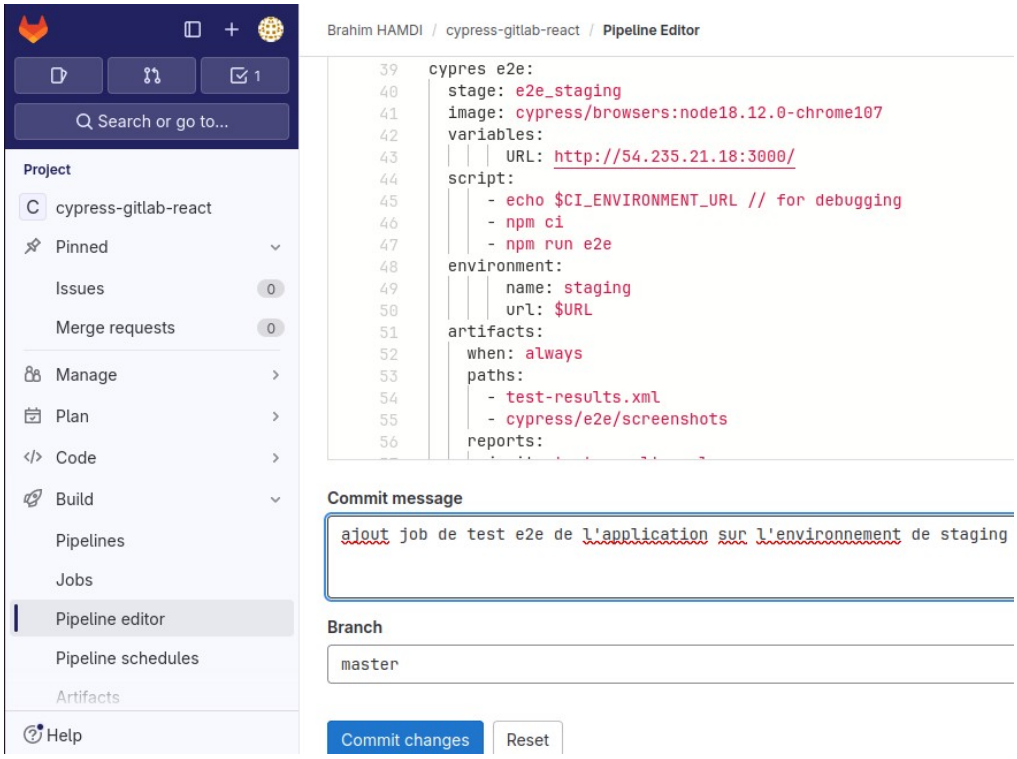
```
const { defineConfig } = require("cypress");

module.exports = defineConfig({
  e2e: {
    baseUrl: process.env.CI_ENVIRONMENT_URL || 'http://localhost:3000',
    video: false,
    setupNodeEvents(on, config) {
    },
  },
});
```

```
brahim@Training:~/TP9/cypress-gitlab-react$ vim cypress.config.js
brahim@Training:~/TP9/cypress-gitlab-react$ git commit -am "cypress.config.js: définir baseURL"
[master 7de0df2] cypress.config.js: définir baseURL
 1 file changed, 1 insertion(+), 1 deletion(-)
```

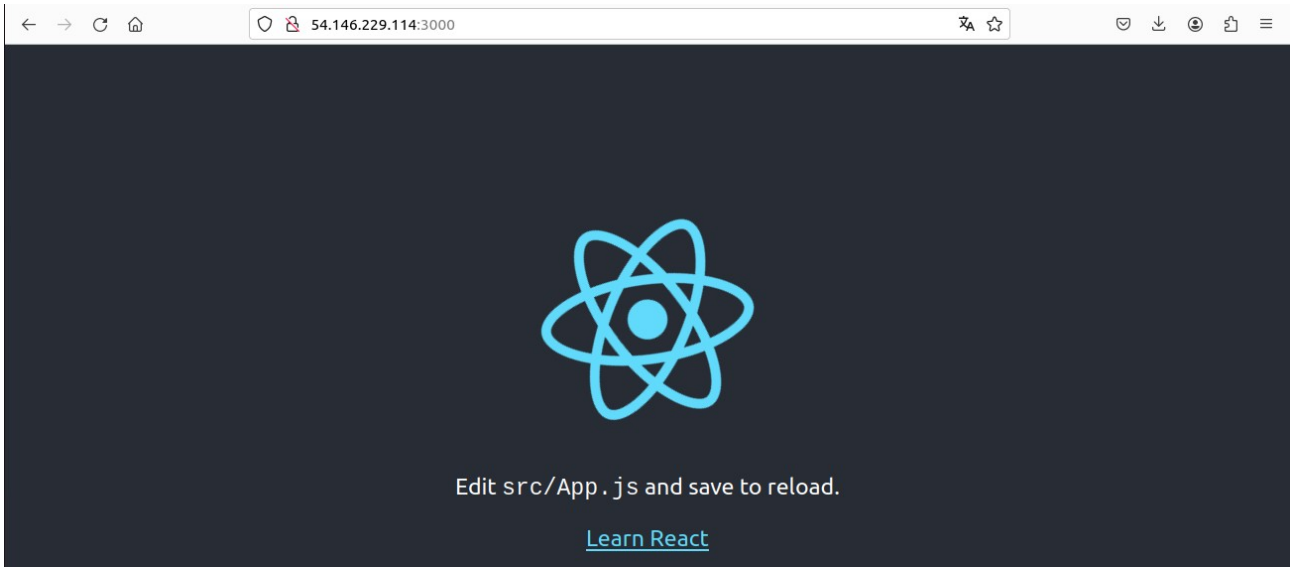
    Cela nous permettra de remplacer la baseUrl de GitLab CI. Si la variable d'environnement CI_ENVIRONMENT_URL n'est pas définie, la valeur par défaut restera http://localhost:3000.

- Ajouter le contenu suivant au nouveau job **Cypress e2e** qui appartient au nouveau stage **e2e_staging** qui va exécuter le test cypress e2e sur l'environnement de staging représenté par l'instance Cloud où l'application tourne.

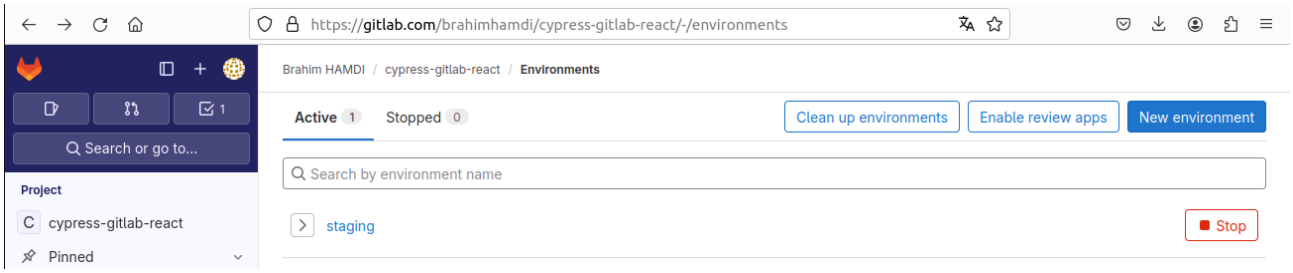
```
stages:
  - build
  - test
  - deploy_staging
  - e2e_staging
…
cypres e2e:
  stage: e2e_staging
  image: cypress/browsers:node18.12.0-chrome107
  variables:
      URL: http://54.235.21.18:3000/
  script:
     - echo $CI_ENVIRONMENT_URL // for debugging
     - npm ci
     - npm run e2e
  environment:
      name: staging
      url: $URL
  artifacts:
    when: always
    paths:
      - test-results.xml
      - cypress/e2e/screenshots
    reports:
      junit: test-results.xml
```



- En utilisant l'URL définit dans le fichier yaml, vérifier que l'application tourne sur l'environnement de staging.



- Accéder à **Operate > Environments** et vérifier qu'un nouveau environnement appelé **staging** a été crée.



- Finalement vérifié que le dernier job de test e2e a été bien fait et a passé sur l'environnement de staging.