

Admin Infra : Exercices

1 Exercice 1 : Un premier pipeline

Nous allons créer un premier pipeline permettant de déployer le site Web des bonnes nouvelles. Ce site Web utilisera une base de données PostgreSQL.

Le pipeline est le suivant :



1. Le dépôt Github est composé des sources PHP du site des bonnes nouvelles.
2. L'environnement de Développement sera composé d'une application PHP (phpapppdev) hébergée par Heroku contenant une base de données Heroku PostgreSQL.
3. L'environnement de Production sera composé d'une application PHP (phpappprod) hébergée par Heroku contenant une base de données Heroku PostgreSQL.
4. Les bases de données dev et prod sont distinctes.
5. A chaque commit/push sur le dépôt, les tests seront effectués et si les tests passent l'environnement dev sera déployé automatiquement.
6. L'environnement prod est déployé manuellement à partir de l'environnement de dev.

Nous allons travailler étape par étape pour réaliser ce pipeline.

1.1 Première étape – Sources & GitHub

La première étape consiste à placer les sources du site des bonnes nouvelles dans un repository GitHub.

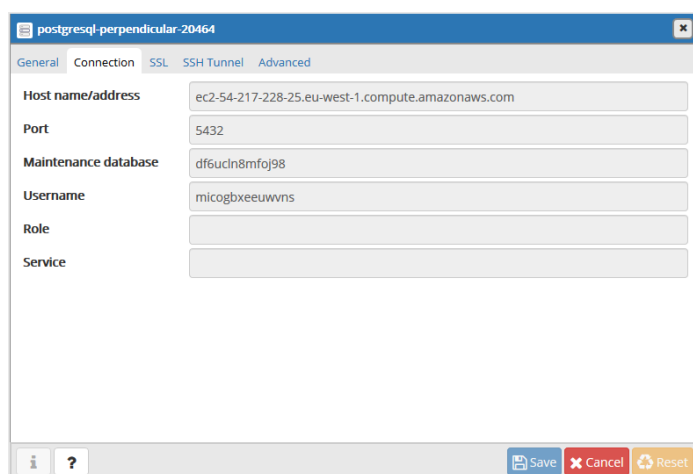
1. Récupérer sur Moodle les sources du site des bonnes nouvelles
 1. Prenez la version du site des bonnes nouvelles de cette séance, les fichiers ont été quelque peu modifiés pour PostgreSQL.
2. Créez un dépôt sur GitHub et ajoutez-y les sources du site des bonnes nouvelles
 1. Vous pouvez faire du drag and drop des sources dans l'interface Web de GitHub.
 2. Vous pouvez également faire des « push » en ligne de commande si vous préférez.

1.2 Deuxième étape – Env Dev / Prod & Heroku

1. Créez-vous un compte sur Heroku (si vous n'en avez pas déjà un)
2. Créer une application phpdev
 1. Connectez-là à votre dépôt github
 2. Activer le déploiement automatique
3. Créer une application phpprod
4. Créer un pipeline
 1. lier l'application phpdev à « staging » (env dev)
 2. lier l'application phpprod à production (env prod)
5. Vérifier que votre pipeline fonctionne bien
 1. Modifier la page d'accueil (views → accueil.php)
 2. Vous pouvez modifier le code directement dans Github
 3. Faites un commit/push
 4. Remarquer que pour l'instant la page « les livres » ne fonctionne pas

1.3 Troisième étape – Base de données & Heroku PostgreSQL

1. Ajouter dans l'application phpprod l'addon Heroku Postgres
 1. Onglet Ressources dans l'application.
 2. Formule « Free »
2. Ajouter dans l'application phpdev l'addon Heroku Postgres
 1. Onglet Ressources dans l'application.
 2. Formule « Free »
3. Connectez-vous à base de données depuis PgAdmin 4 v4
 1. Les informations de connexion se trouvent dans Heroku Postgres → Settings → Database Credentials
 2. Cochez bien « SSL require » dans l'onglet SSL
 3. Un exemple :



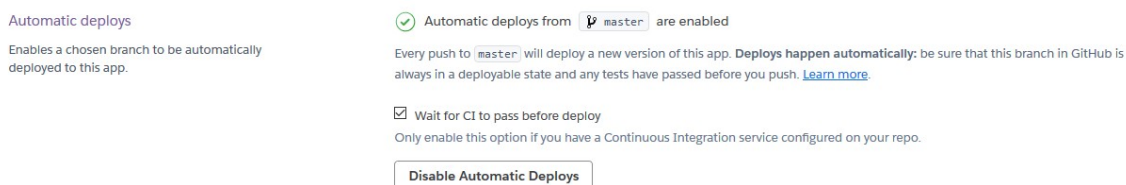
4. Dans le schéma public de votre base de données → créer la table « livres »
 1. colonnes : no (bigserial), titre (character varying) , auteur (character varying)
5. Insérer quelques données dans la table livres
6. Modifier la classe « DB.class.php » pour utiliser la base de données Heroku
 1. <https://devcenter.heroku.com/articles/heroku-postgresql>
7. Tester

1.4 Quatrième étape – Tests & Travis CI

Travis CI s'intègre très bien à GitHub.

1. Faites en sorte que le déploiement automatique dans Heroku ne se fasse maintenant uniquement si les tests passent.

1. Dans votre application phpdev, cocher « Wait for CI to pass before deploy »



2. Suivez le tutoriel de Travis CI pour connecter Travis CI à Github
 1. <https://docs.travis-ci.com/user/tutorial/>
3. Préparation Tests simples
 1. Récupérez sur Moodle le fichier .travis.yml
 2. Placez le fichier .travis.yml à la racine du dépôt GitHub
 3. Testez
 4. Qu'utilise Travis comme environnement pour réaliser les tests ?
4. Tests simples
 1. Remplacez le fichier .travis.yml par celui présent dans « travis phpunit » sur MooVin
 2. Ajoutez le fichier « test.php »
 3. Testez
5. Bonus : Commencez à réfléchir au pipeline de développement pour votre PFE.