****

Projet ACLIA

Outil d’annotation

Documentation technico-fonctionnelle

29 janvier 2021



Table des matières

[**I.** **Description** 2](#_Toc62834635)

[**II.** **Démarrage** 2](#_Toc62834636)

[**A.** **Prérequis** 2](#_Toc62834637)

[**B.** **Installer le projet** 2](#_Toc62834638)

[**C.** **Execution** 2](#_Toc62834639)

[**III.** **Déploiement** 3](#_Toc62834640)

[**A.** **Workflow** 3](#_Toc62834641)

[**IV.** **Documentation technique** 5](#_Toc62834642)

[**V.** **Finalité et fonctionnement de l’application** 5](#_Toc62834643)

[**VI.** **Description des tables utilisées par l’application** 5](#_Toc62834644)

[**A.** **Table des annotations réalisées : app\_annotation** 5](#_Toc62834645)

[**B.** **Table d’historisation des actions réalisées sur l’application : app\_user\_log** 5](#_Toc62834646)

[**C.** **Table des informations utilisateurs : app\_users** 6](#_Toc62834647)

[**D.** **Table des actes présentés dans l’interface : test\_annotations** 6](#_Toc62834648)

[**VII.** **Contribution** 6](#_Toc62834649)

[**A.** **Processus de Pull Request** 7](#_Toc62834650)

[**B.** **Code de conduite** 7](#_Toc62834651)

[1. Notre engagement 7](#_Toc62834652)

[2. Nos Standards 7](#_Toc62834653)

[3. Nos responsabilités 7](#_Toc62834654)

[4. Portée 7](#_Toc62834655)

[5. Mise en vigueur 8](#_Toc62834656)

[**C.** **Comment contribuer** 8](#_Toc62834657)

[1. Bases 8](#_Toc62834658)

[2. Rebasing 8](#_Toc62834659)

[**VIII.** **Version** 9](#_Toc62834660)

# **Description**

@ctes-ia est une application NLP, construite avec Reactjs et NodeJs.

L’outil d’annotation est accessible sur l’adresse https://dgcl-aclia.starclay.fr

Le code est versionné sur le serveur Git de Starclay à l’adresse <https://gitlab.starclay.fr/ms9/dgcl/actes-webapp>

# **Démarrage**

## **Prérequis**

**IDE**

Vous pouvez choisir l'IDE que vous voulez, mais vous devez avoir certains packages installés avec comme ESLint pour le linter en direct, et certains packages React pour obtenir l'auto-complétion, ... Voici un exemple des packages nécessaires avec Visual Studio Code :

* Reactjs code snippets
* ESLint

#### **Node / Npm / Yarn**

Comme il s'agit d'un projet basé sur JS, en raison de ReactJS & express, vous devez installer des éléments sur votre ordinateur :

Version :

* NodeJs version 10.6 et plus.
* Npm version 6.9 et plus.
* macOS: Installer [Homebrew](https://brew.sh) comme gestionnaire de package et installer les dépendances suivantes: brew install node et brew install npm
* Windows:  Installer  [Chocolatey](https://chocolatey.org) comme gestionnaire de package et installer les dépendances suivantes: choco install -y nodejs.install python2 jdk8 npm
* Linux: Suivez les instructions suivantes pour installer Node 8 ou une version plus récente et npm :  [install instructions for your linux distribution](https://nodejs.org/en/download/package-manager/)
* Yarn: npm install -g yarn

## **Installer le projet**

Vous devez d'abord cloner le projet Git disponible à l’adresse [https://gitlab.starclay.fr/ms9/dgcl/actes-webapp](https://gitce.conseil-etat.fr/externe/juradinfo-ia/juradinfo-ia-www). N'oubliez pas de télécharger votre clé SSH dans git et d'avoir le bon accès : cd actes-webapp/client && yarn && cd ../server && yarn

## **Execution**

Lorsque tout est installé, si vous souhaitez l'exécuter, procédez comme suit:

Serveur : yarn start:dev

client: yarn start

**NOTE :**

Avant d'exécuter le serveur et le client, assurez-vous de:

* 1. configurez le fichier "default.json" dans ' actes-webapp/server /config/default.json' pour pointer vers votre base de données postgresql (prod :192.168.222.204 :5432)
  2. Si vous souhaitez pointer vers la base de données de production, assurez-vous d’être dans le réseau de Starclay.

Par défaut, le serveur pointe vers la base de données de production (192.168.222.204 :5432).

Par défaut le client en mode DEV pointe sur localhost:3011 mais lors d’un « build » sur l’url de production ou de pré-production. Vous trouverez l’information dans actes-webapp/client/src/core/config

Vous trouverez un exemple des deux configurations dans la section déploiement.

# **Déploiement**

Lors du déploiement sur le serveur, assurez-vous que tout a été testé et fonctionne correctement en local. Soyez prudent et n'ayez pas peur de demander de l'aide si vous en avez besoin !

Veuillez noter que le workflow est simple en cas de problème pour quelque raison que ce soit, vous devez être capable de le gérer et essayer d'adapter autant que possible le workflow.

## **Workflow**

**NOTE:**

Après avoir cloné le projet, assurez-vous que vous avez le bon accès à la base de données postgresql et au serveur de Starclay via ssh.

#### **Part 1 - Test développement**

1. Une fois le nouveau patch prêt et fusionné dans la branche develop. Clonez le projet dans un dossier tmp.
2. Installez maintenant le projet et testez-le avec la base de données de production. Vérifiez que tout fonctionne bien et que les modifications ne créent pas de nouveaux bugs et bien sûr résolvent l'existant. Cf. section “Execution”.

#### **Part 2 - Package**

1. Vérifier le fichier default.json dans actes-webapp/server/config, ça doit être comme ça:

{

« port » :3011

« logLevel » : « info »,

« db » :{

« database » : CHANGE\_ME

« user » : ghpsj

« host » : « 192.168.222.204 »

« port » : 5342

« password » : CHANGE\_ME

}

}

1. Créer le build du client, vérifiez le fichier client/src/core/config. Afin d’entrer l’url entre la prod ou la pré-prod. Néanmoins l’API tourne toujours sur le port 3011.

cd client && yarn build

1. Enfin zipper l’ensemble des sources zip -r package.zip client/build server

**NOTE :** Oui, nous compressons également le node\_module, vous pouvez le supprimer avant mais c n'est pas obligatoire.

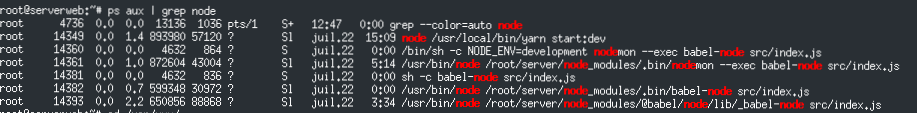
#### **Part 3 – Nettoyer le serveur et envoyer le fichier zip**

1. Ouvrez un nouveau terminal et connectez-vous au serveur Starclay (changez <username> par votre nom d'utilisateur): ssh <username>@192.168.6.70
2. Arrêter les processus existants :

* Premièrement le client web située hébergé via nginx.  
  cd /var/www/html && rm –rf ./\*
* Ensuite sauvegardons les logs et supprimons les sources de l’api:

Cd ~/server && cp –rf ./server/\*.out ../logs && cd .. && rm –rf ./server\*

* Maintenant, arrêtons les processus de l’api :
  + Premièrement, les trouver avec : ps aux | grep node un exemple de l’output:



* + Vous trouverez les PID dans le 2èm colonne à gauche ici : 14349, 14360, 14361, 14381, 14382, 14393pour arrêter les processus voici la commande :

kill -9 {PID} remplacez {PID} par le PID réel

* Retournez dans votre dossier local où se trouve votre fichier zip et envoyez-le au serveur. Bien sûr, ne fermez pas votre connexion ssh, utilisez un autre terminal !

scp package.zip <username>@192.168.222.207:/home/<username>

#### **Part 4 – Packager le serveur**

1. Retournez dans le terminal où votre connexion ssh a été établie : unzip `package.zip'
2. Copier le dossier du serveur et réinstallez les node\_modules :

cd /server && yarn

1. Rendre disponible le serveur :

Sudo sh –c « sudo nohup npm run start:prod 2>&1 > ./server\_actes-api.out & »

1. Et maintenant retournez dans /home/<username> et copiez le build:

Cp –a build/\* /var/www/html && chmod 755 /var/www/html/static

1. Puis relancez nginx (la gestion SSL est automatiquement gérer par nginx et letsencrypt) :

Sudo systemctl restart nginx

1. Ouvrez votre navigateur Web préféré et accédez à l'URL:

https://dgcl-aclia.starclay.fr essayez de vous connecter avec votre compte.

1. Testez à nouveau le site Web comme à l'étape 1. Si tout est OK, vous avez déployé avec succès le nouveau patch !

# **Documentation technique**

Si vous voulez tout savoir sur l'application et ses différents composants, rendez-vous à la racine du dossier client ou serveur et exécuter la commande :

Client / serveur : yarn doc

# **Finalité et fonctionnement de l’application**

* Le but de l’outil d’annotation est de permettre d’annoter les actes
* **L’annotation se réalise sur du** **texte brut issu du pipeline d’ingestion des requêtes** (voir projet https://gitlab.starclay.fr /ms9/dgcl/actes -ia-data). Ce texte est présenté dans une page html sur l’application.
* Le fonctionnement de l’application nécessite la présence des quatre tables suivantes dans la base de données ‘dgcl’ : app\_users, app\_annotation, app\_user\_log, dw\_actes.

# **Description des tables utilisées par l’application**

**Table des annotations réalisées : app\_annotation**

|  |  |  |
| --- | --- | --- |
| **Colonne** | **Type** | **Description** |
| date | datetime | Date de la dernière action |
| qcm | json | Contient l’ensemble des annotations sous un format json |
| iddocument | chaine de caractères | Identifiant de l’acte |
| filename | chaine de caractères | Nom de l’acte |
| username | chaine de caractères | Identifiant de l’utilisateur ayant réalisé l’annotation |

**Table d’historisation des actions réalisées sur l’application : app\_user\_log**

|  |  |  |
| --- | --- | --- |
| **Colonne** | **Type** | **Description** |
| event\_date | datetime | Date de l’action |
| message | chaine de caractères | Description de l’action |
| username | chaine de caractères | Identifiant de l’utilisateur réalisant l’action |

**Table des informations utilisateurs : app\_users**

|  |  |  |
| --- | --- | --- |
| **Colonne** | **Type** | **Description** |
| id | Nombre unique | Id de l’utilisateur |
| user | chaine de caractères | Nom de l’utilisateur |
| password | chaine de caractères | Mot de passe de l’utilisateur |
| Creation\_date | Datetime | Date de création |
| Userin | Chaine de caractères | Clé de la license. |

**Table des actes présentés dans l’interface : test\_annotations**

|  |  |  |
| --- | --- | --- |
| **Colonne** | **Type** | **Description** |
| noacte | chaine de caractères | Numéro de l’acte. Un numéro d’acte peut avoir plusieurs fichiers rattachés : l’acte et les pièces jointes |
| filename | chaine de caractères | Nom du fichier océrisé. Il s’agit de la clef primaire dans la table. |
| nature | chaine de caractères | Le code de la nature de l’acte. 1 : délibération, 2 : actes réglémentaires, 3 : actes individuels, 4 : contrats, conventions et avenants, 5 : documents budgétaires et financiers, 6 : autres |
| matiere | chaine de caractères | Le code de matière juridique de l’acte. Sous la forme « X. Y.Z… », X étant le code de la matière niveau 1, Y le code matière niveau 2, etc |
| objetacte | chaine de caractères | Le texte saisi par les collectivité locales pour renseigner l’objet de l’acte |
| texte | chaine de caractères | Texte océrisé issu de la pipeline data |
| pj\_acte\_principal | nombre | Est rempli lorsque le fichier est une pièce jointe. La valeur renseignée est le « filename » de l’acte « principal » |
| dept | chaine de caractères | Numéro du département d’origine de l’acte |
| iddocument | chaine de caractères | Identifiant d’un acte. Il s’agit d’une 2e déclinaison de « noacte » |
| is\_recrutement | chaine de caractères | Booléen. Vrai lorsque « objetacte » contient le mot « recrutement ». Faux sinon. |
| is\_avenant | nombre | Booléen. Vrai lorsque « objetacte » contient le mot « avenant ». Faux sinon. |
| statut | nombre | Statut de l’annotation. 0 : pas annoté, 1 : annoté, -1 : acte illisible (mauvaise océrisation) |
| step | nombre | Etat actuelle dans le processus d’annotation. 1 : 'transmissibilité', 2 : 'nature', 3 : 'matières\_1\_2', 4 : 'méta-Objet', 5 : 'Corps', 6 : 'pj' |

# **Contribution**

Lorsque vous contribuez à ce référentiel, veuillez d'abord discuter de la modification que vous souhaitez apporter via un ticket, un e-mail ou toute autre méthode avec les propriétaires de ce référentiel avant d'effectuer une modification. Veuillez noter que nous avons un code de conduite, veuillez le suivre dans toutes vos interactions avec le projet.

## **Processus de Pull Request**

* Test your code at 100%.
* Update unit tests.
* Don't forget to rebase your branch.
* Asking a review.

## **Code de conduite**

### Notre engagement

Dans l'intérêt de favoriser un environnement ouvert et accueillant, nous, en tant que contributeurs et responsables, nous engageons à faire de la participation à notre projet et à notre communauté une expérience sans harcèlement pour tout le monde, indépendamment de l'âge, de la taille du corps, du handicap, de l'ethnicité, de l'identité de genre et de l'expression, niveau d'expérience, nationalité, apparence personnelle, race, religion, ou identité et orientation sexuelles.

### Nos Standards

Voici des exemples de comportements qui contribuent à créer un environnement positif:

* Utiliser un langage accueillant et inclusif
* Être respectueux des différents points de vue et expériences
* Accepter gracieusement les critiques constructives
* Se concentrer sur ce qui est le mieux pour la communauté
* Faire preuve d'empathie envers les autres membres de la communauté

Voici des exemples de comportement inacceptable des participants:

* L'utilisation de langage ou d'images sexualisés et l'attention sexuelle importune
* Trolling, propos insultants / désobligeants et attaques personnelles ou politiques
* Harcèlement public ou privé
* Publier des informations privées d'autrui, telles qu'une adresse physique ou électronique, sans autorisation explicite
* Autre comportement pouvant raisonnablement être considéré comme inapproprié dans un cadre professionnel

### Nos responsabilités

Les responsables de projet sont responsables de clarifier les normes de comportement acceptable et doivent prendre des mesures correctives appropriées et équitables en réponse à tout cas de comportement inacceptable.

Les responsables de projet ont le droit et la responsabilité de supprimer, modifier ou rejeter les commentaires, les validations, le code, les modifications du wiki, les problèmes et autres contributions qui ne sont pas alignés sur ce code de conduite, ou d'interdire temporairement ou définitivement tout contributeur pour d'autres comportements qui ils jugent inapproprié, menaçant, offensant ou nuisible.

### Portée

Ce code de conduite s'applique à la fois dans les espaces du projet et dans les espaces publics lorsqu'une personne représente le projet ou sa communauté. Des exemples de représentation d'un projet ou d'une communauté incluent l'utilisation d'une adresse e-mail officielle du projet, la publication via un compte de médias sociaux officiel ou la fonction de représentant désigné lors d'un événement en ligne ou hors ligne. La représentation d'un projet peut être davantage définie et clarifiée par les responsables du projet.

### Mise en vigueur

Des cas de comportement abusif, de harcèlement ou autrement inacceptable peuvent être signalés en contactant l'équipe de projet à [INSERER L'ADRESSE EMAIL]. Toutes les plaintes seront examinées et examinées et donneront lieu à une réponse jugée nécessaire et appropriée aux circonstances. L'équipe de projet est tenue de maintenir la confidentialité à l'égard du journaliste d'un incident. De plus amples détails sur les politiques d'application spécifiques peuvent être publiés séparément.

Les responsables de projet qui ne respectent pas ou n'appliquent pas le Code de conduite de bonne foi peuvent faire face à des répercussions temporaires ou permanentes telles que déterminées par d'autres membres de la direction du projet.

## **Comment contribuer**

### Bases

Tout d'abord, passez votre commande sur dev et obtenez les dernières mises à jour:

$ git checkout develop && git pull origin develop

Maintenant, vous êtes configurés et vous pouvez créer votre propre branche :

$ git checkout -b "awesome-feature"

Oui, vous avez mis à jour et ajouté de nombreux fichiers, vous devez valider ces modifications :

$ git add $(UNTRACKED\_FILES) # Vous pouvez voir vos fichiers non suivis avec `git status`

$ git commit $(FILES\_TO\_COMMIT) -m "relevant commit message"

Vous pouvez maintenant, si vous le souhaitez, les pousser sur votre branche :

$ git push -u origin awesome-feature

Après de nombreuses validations, et peut-être quelques push, vous souhaitez fusionner cette fonctionnalité impressionnante! Tout d'abord, obtenez la dernière version de développement!

$ git fetch origin develop

Vous êtes maintenant prêt à rebaser!

### Rebasing

Vous devriez toujours rebaser avant de soumettre un PR, sinon, git vous avertira que votre branche de développement locale est en retard (par rapport au remote)

Pour cela :

$ git rebase develop

Vous pouvez également fournir l'indicateur -i (raccourci pour --interactive) pour adapter la rebase et avoir un historique propre (ce n'est pas nécessaire si vos commits ont du sens) Le mode interactif vous permet de créer une branche propre, et éventuellement de fusionner / supprimer les validations.

Si, lors de votre rebase, vous avez des conflits (c'est normal, ne vous inquiétez pas), vous devez corriger ces conflits de fusion. Une fois que vous avez terminé, vous pouvez faire:

$ git add $(FILES\_WITH\_RESOLVED\_CONFLICTS)

Une fois tous vos conflits résolus, vous pouvez:

$ git rebase –continue

Effectuez ces deux étapes autant de fois que nécessaire, vous pouvez avoir plusieurs conflits par rebase (si vous n'avez pas de chance). Quand git vous a dit que votre rebase est terminé, vous pouvez enfin push :

$ git push –f

L'indicateur -f force la télécommande (gitlab) à réécrire votre historique, ceci est nécessaire après un rebase. Maintenant, vous pouvez soumettre votre PR!

# **Version**

Actuellement nous utilisons gitlab comme système de versionning. Le produit actes-ia est livré à la version 1.0.0.