

# Rapport Programmation WEB

Jessy FALLAVIER

## Introduction

L'objet du projet de programmation web était de créer un site de scrutin.

J'ai nommé ce projet **AVotey** rappelant l'annonce "a voté" présente lors du dépôt d'un bulletin de vote dans une urne.

## Réalisation

### Généralités

J'ai utilisé le framework bulma CSS.

Le site AVotey est une monopage sans rechargement avec des requêtes Ajax. Dès l'entrée sur la page principale, on ouvre une session avec `session_start()`.

Généralement, chaque fichier php renvoie un json avec un champ status (succes | error), un message et éventuellement un champ data.

Il y a deux fichiers JSON. On encode depuis php chaque fichier JSON avec le drapeau `JSON_PRETTY_PRINT` pour que le JSON reste lisible.

## Fichiers JSON

users.json contient un tableau associatif avec un email comme clé et des champs :

- password, représentant le mot de passe haché
- lists, représentant les listes gérées par l'utilisateur
  - Ce champ est une liste associative avec un nom de liste comme clé, et comme valeur : une autre liste associative avec un email comme clé et un nombre de votes permis comme valeur.
- notificationToken, le token de notification de l'utilisateur

scrutins.json contient un tableau associatif avec l'identifiant du scrutin comme clé et des champs :

- organizer, représentant l'email de l'organisateur
- question, représentant la question posée
- choices, une liste représentant les choix disponibles
- systemeVote, représentant le système de vote utilisé (uninominal ou jugementMajoritaire)
- voters, une liste représentant la liste des votants
- votesPermis, un tableau associatif avec comme clé l'email d'un votant et comme valeur le nombre de votes permis (1+procurations)
- votesUtilises, un tableau associatif avec comme clé l'email d'un votant et comme valeur le nombre de votes utilisés.
- compteurNotifs, un compteur du nombre de notifications envoyées
- open un booléen représentant si le scrutin est ouvert ou non
- votes, la liste des bulletins de votes chiffrés
- publicKey, la clé publique associée au scrutin (générée par JSEncrypt de taille 1024)
- resultats, un tableau associatif trié par ordre de succès, avec comme clé un choix et comme valeur le nombre de choix en faveur si c'est un scrutin uninominal ou la réparation si c'est un scrutin à jugement majoritaire.

## Fonctionnalités AJAX :

### Compte

- Inscription
- Connexion
- Choix automatique entre inscription et connexion
- Déconnexion

### Listes

- Récupérer les listes
- Définir les listes

### Scrutin

- Créer un scrutin
- Consulter les scrutins disponibles
- Récupérer un scrutin
- Voter pour un scrutin
- Clôturer un scrutin
- Publier les résultats
- Consulter les résultats
- Détruire un scrutin

### Notification – Pdf

- Définir le token de notification
- Envoyer des notifications
- Obtenir le fichier pdf de résultats

# Implémentation pratique de chaque fonctionnalité AJAX

## Inscription

On récupère la valeur des champs email et mot de passe puis on l'envoie via une requête ajax à register.php

On vérifie que :

- La personne n'est pas déjà connectée
- L'email est valide avec le filtre FILTER\_VALIDATE\_EMAIL
- Le mot de passe doit faire au moins 8 caractères
- L'email n'est pas déjà utilisé

Le mot de passe est haché avec la fonction password\_hash et le drapeau PASSWORD\_DEFAULT.

Ensuite, on écrit dans le fichier users.json pour ajouter le nouvel utilisateur.

Puis on en cas de succès, définit le champ uuid de la session avec la valeur de l'email et on renvoie l'uuid pour qu'un affichage puisse être fait.

## Connexion

On récupère la valeur des champs email et mot de passe puis on l'envoie via une requête ajax à login.php

On vérifie que :

- La personne n'est pas déjà connectée
- Si on trouve une paire email/password correspondante dans le fichier users.json.

Le mot de passe est dé-haché avec la fonction password\_verify.

Puis en cas de succès, on définit le champ uuid de la session avec la valeur de l'email et on renvoie l'uuid pour qu'un affichage puisse être fait.

## Choix automatique entre inscription et connexion

On récupère le champ email puis on l'envoi via une requête ajax à emailExists.php.

On fait cela à chaque changement de champs email grâce à onchange.

Si le mail existe déjà dans users.json, on renvoie data 1, sinon data 0. On active ou désactive les boutons login/register en fonction.

## Déconnexion

Lors d'un clic sur le bouton "Se déconnecter", on fait un appel Ajax vers `logout.php`.

On vérifie que l'utilisateur n'est pas déjà connecté.

On met le champs `uuid` de la session en chaîne vide.

## Récupérer les listes

Grâce à un appel Ajax à `getLists.php` on peut récupérer les listes du fichier `users.json` de l'utilisateur.

Il faut donc que l'utilisateur soit connecté et que son compte existe.

Il y a deux cas de figure pour utiliser ces listes :

- Pour les gérer :
  - Dans ce cas, on va créer une interface pour gérer les listes : en ajouter / supprimer des listes, modifier le nom de la liste, ajouter / supprimer des votants, modifier l'email / le nombre de procurations des votants.
- Pour un ajout rapide :
  - Lors de la création d'un scrutin on pourra utiliser ces listes prédéfinies.

## Définir les listes

A chaque changement dans l'interface de gestion des listes, on va faire un appel ajax vers `setLists.php`.

Pour être sauvegardés dans le fichier `users.json`, ces listes doivent être :

- Avec un titre non-vide
- Chaque votant doit avoir un email valide avec le filtre `FILTER_VALIDATE_EMAIL`
- Chaque votant doit avoir un nombre de procurations `[0, 2]` (nombre de votes permis `[1, 3]`)

## Créer un scrutin

On peut créer un scrutin en renseignant la question, le système de vote parmi un select (uninominal ou jugementMajoritaire), la liste des choix et la liste des votants.

Un système de choix rapide est mis en place avec des choix prédéfinis pour les choix (binaire => Oui / Non) et les listes pour les votants.

Au clic du bouton créer un scrutin, on va effectuer un appel AJAX vers createScrutin.php.

Pour être sauvegardé, ce scrutin doit :

- Être créé par une personne connectée
- Avoir une question non vide
- Au moins 2 choix, non vides et uniques
- Au moins 1 votant unique (avec un email valide et un nombre de procurations [0,2])
- Une clé publique non vide

L'identifiant du scrutin est défini grâce à la fonction uniqid() et avec un préfixe "S-".

La clé privée du scrutin est enregistrée grâce au local storage avec comme nom scrutin-ID.

## Consulter les scrutins disponibles

Pour consulter les scrutins disponibles, un appel ajax vers consulter.php est effectué.

L'utilisateur a accès à les scrutins qu'il a créé ainsi qu'à ceux dont il est désigné votant.

On renvoie donc :

- Un indicateur pour savoir si le scrutin est votable, ou s'il est créé par lui-même.
- L'ID, la question, l'organisateur, le statut ouvert du scrutin
- Le nombre de votes restants à l'utilisateur
- Le nombre de votes permis et le nombre de votes utilisés totaux pour pouvoir afficher un taux de participation

On laisse la possibilité à l'utilisateur de filtrer ces scrutins avec ces choix :

- Votable
- Avec résultats
- Créé par lui-même

## Récupérer un scrutin

Lorsque l'on veut voter pour un scrutin, nous faisons un appel AJAX avec comme paramètre l'ID de ce scrutin.

Pour récupérer ce scrutin, il faut que :

- L'utilisateur soit connecté
- Le scrutin avec cet id existe et soit ouvert
- L'utilisateur soit votant
- Qu'il lui reste des votes ( $\text{votesPermis} - \text{votesUtilises} > 0$ )

Dans ce cas, on lui renvoie la question, les choix, la clé publique, le système de vote et une petite phrase récapitulative à afficher.

## Voter pour un scrutin

Une fois le choix choisi, on fait un appel AJAX avec comme paramètre l'id du scrutin et le choix chiffré à voter.php

Pour que le vote soit enregistré, il faut que :

- L'utilisateur soit connecté
- Le scrutin avec cet id existe et soit ouvert
- L'utilisateur soit votant
- Qu'il lui reste des votes ( $\text{votesPermis} - \text{votesUtilises} > 0$ )
- Le choix soit non vide

Ensuite, on décompte un vote à l'utilisateur et on ajoute ce vote à la liste des votes.

## Clôturer un scrutin

Lorsque l'organisateur le décide, il peut décider de clôturer le scrutin. L'id du scrutin est envoyé via un appel AJAX à clôturer.php

Pour que le scrutin soit clôturé, il faut que :

- L'utilisateur soit connecté
- Le scrutin avec cet id existe et soit ouvert
- L'utilisateur soit organisateur

Ensuite, on revoie une liste mélangée des votes à l'organisateur pour qu'il puisse faire le dépouillement.

## Publier les résultats

Dans le javascript, en récupérant les votes, on les déchiffre avec la clé privée contenue dans le local storage.

Si le système de vote est uninominal, il suffit de compter chaque choix et de publier ces résultats {choix : nbDeVotes } dans l'ordre décroissant

Si le système de vote est à jugement Majoritaire, il suffit d'enregistrer chaque mention votée pour chaque choix. Puis de calculer la mention majoritaire. On publiera ces résultats sous la forme {choix : {repartition : [1, 0 ..., 2], mention : "TB"}}

On effectue donc un appel AJAX à publierResultats.php avec l'id du scrutin et les résultats.

Les résultats sont publiés si :

- L'utilisateur soit connecté
- Le scrutin avec cet id existe et soit ouvert
- L'utilisateur soit organisateur

Ensuite, on enregistre les résultats dans le fichier scrutin et passer le scrutin à clos.

On peut ainsi supprimer la clé privée du local storage.

## Consulter les résultats

Pour consulter les résultats, un appel AJAX est effectué à resultatsScrutins.php avec l'id du scrutin.

Il faut que :

- L'utilisateur soit connecté
- Le scrutin avec cet id existe et soit fermé
- L'utilisateur soit organisateur ou votant

Ensuite, on renvoie à l'utilisateur la question, l'organisateur, le système de vote, les résultats, ainsi que les totaux permis et utilisés du scrutin pour afficher un taux de participation.

## Détruire un scrutin

Pour détruire un scrutin on effectue un appel AJAX à detruire.php.

Il faut que :

- L'utilisateur soit connecté
- Le scrutin avec cet id existe
- L'utilisateur soit organisateur

Ensuite, le scrutin est supprimé du fichier scrutins.json

## Définir le token de notification

Pour pouvoir recevoir les notifications, un token doit être créé. On utilise ici l'API Firebase de Google.

Si on veut recevoir des notifications, le token est créé, sinon le token envoyé est vide.

Un appel AJAX est effectué à setNotificationToken.php avec le token.

Il faut que :

- L'utilisateur soit connecté

Si on ajoute un token, on l'enregistre dans le fichier users.json, sinon, on supprime le token de l'utilisateur et dans les autres comptes là où le même token a été utilisé. Dans le cas d'une désactivation, l'appareil entier ne recevra donc plus de notifications.



## Envoyer des notifications

Pour envoyer des notifications, un appel AJAX à `envoyerNotifications.php` est effectué à chaque création et publication de résultats de scrutins, avec l'id du scrutin et le type de notification.

Il faut que :

- L'utilisateur soit connecté
- Le scrutin existe
- L'utilisateur soit l'organisateur
- 0 (resp. 1) notification maximum ait été déjà envoyée si c'est une ouverture (resp. clôture)

On envoie donc les notifications avec l'API Firebase depuis une API de mon site personnel.

## Obtenir le fichier pdf de résultats

Pour obtenir un pdf récapitulatif de résultat, on peut effectuer un appel AJAX à `getPdfResultats.php` avec l'id du scrutin.

Il faut que :

- L'utilisateur soit connecté
- L'utilisateur soit l'organisateur ou votant
- Le scrutin existe et est clôturé

Ensuite un pdf contenant les informations essentielles est généré grâce à la librairie `tcpdf` depuis mon site personnel. Ce PDF est encodé en base64 puis un blob est créé depuis javascript pour pouvoir afficher ce PDF.

## Fonctionnalités HTML

- Affichage des messages
- Toggle login
- Toggle scrutins

## Implémentation pratique de chaque fonctionnalité HTML

### Affichage des messages

Des messages informatifs peuvent être affichés. Il y a un affichage sticky, comment à tous les AJAX. Le message disparaît au bout de 5 secondes. Il est coloré différemment selon si c'est un succès ou une erreur.

### Toggle login

Pour basculer du statut connecté ou non, on affiche certains objets ou non. Un choix par défaut est effectué par PHP au chargement de la page.

### Toggle scrutins

Pour pouvoir basculer entre chaque panel d'action de chaque scrutin (Gérer mes listes, Créer un scrutin, Voir mes scrutins), on va afficher certaines div ou non.

## Expérience Utilisateur :

### Bienvenue sur AVotey

Une plateforme de scrutins sécurisés

#### Compte

Veuillez vous connecter.

Email :

jessy.fallavier@gmail.com

Email valide

Password :

\*\*\*\*\*

Minimum 8 caractères

Login

Register

#### Scrutins

Vous devez être connecté pour interagir avec les scrutins

On se connecte à son compte en cliquant sur le bouton login.

Activer Notifications

Désactiver Notifications

#### Scrutins

Gérer mes listes

Créer un scrutin

Voir mes scrutins

Titre de la liste :

Liste Normale

Titre de la liste.

Votants :

jessy.fallavier@gmail.com

1

Retirer le votant

jessy.fallavier@universite-paris

0

Retirer le votant

Ces votants pourront voter au scrutin. Maximum 2 procurations.

Ajouter un votant

Retirer la liste

Ajouter une liste

AVotey par JESSY FALLAVIER. Projet universitaire. Licence : CC BY NC SA 4.0.

On peut gérer ses listes

#### Scrutins

Gérer mes listes

Créer un scrutin

Voir mes scrutins

Question :

Question ... ?

Cette question sera posée aux votants.

Système de vote :

Uninominal

Choix :

Ces choix seront proposés aux votants.

Ajouter un choix

Choix prédéfinis :

Ajouter depuis une liste

Votants :

Ces votants pourront voter au scrutin. Maximum 2 procurations.

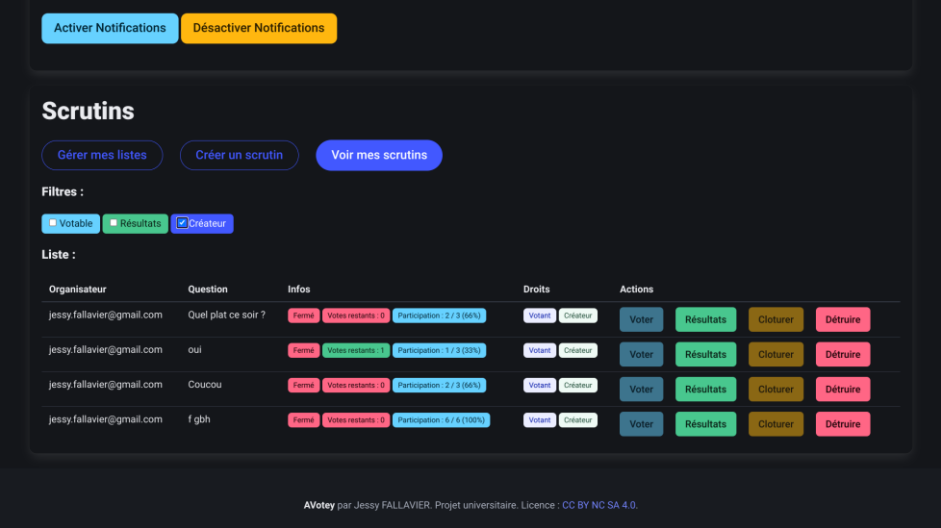
Ajouter un votant

Liste de votants :

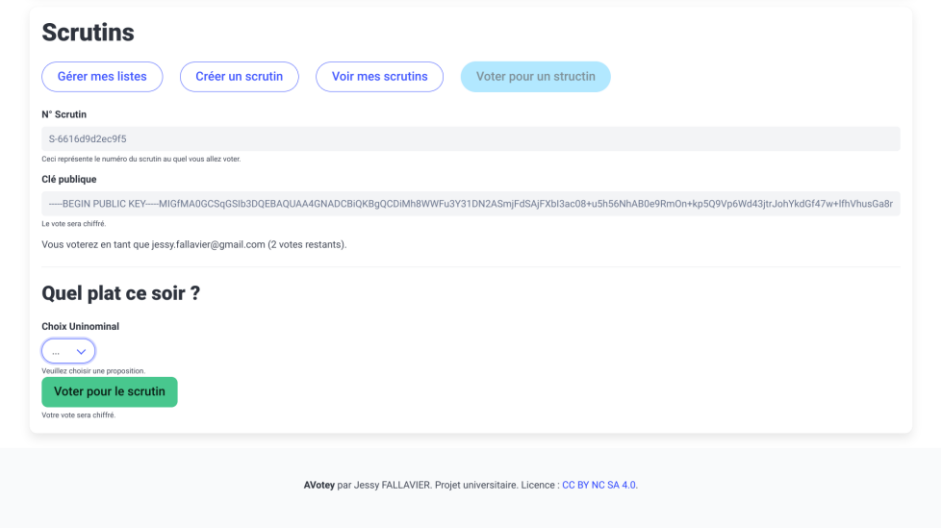
Ajouter depuis une liste

Créer le scrutin

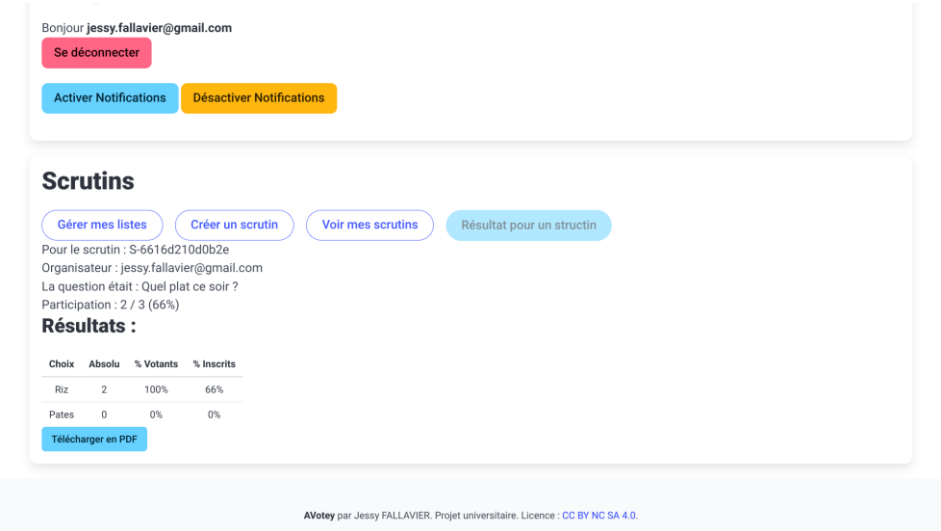
On peut créer un scrutin.



On peut consulter et trier les scrutins disponibles. Les boutons permettent d'effectuer les actions sur les scrutins. Ici le thème est sombre.



On peut voter pour un scrutin



Affichage des résultats


Notifications activées pour ce compte



Titre vide



## Exemples de messages

 **Résultats du scrutin** jessy.fallavier@gmail.com (AVotey)  
**S-6616d210d0b2e**

Scrutin n°: S-6616d210d0b2e  
Organisateur: jessy.fallavier@gmail.com

Question: Quel plat ce soir ?

Choix disponibles:  
- Pates  
- Riz

Système de vote: uninominal

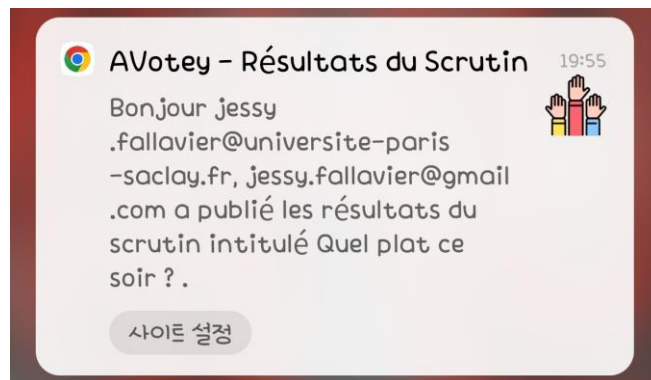
Votes permis:  
- jessy.fallavier@gmail.com: 2 votes  
- jessy.fallavier@universite-paris-saclay.fr: 1 votes

Nombre de votes permis: 3  
Nombre de votes utilisés: 2

Clé publique:  
-----BEGIN PUBLIC KEY-----  
MIGfMA0GCsGqSb3DQEBAAQAA4GNADCBiQKBgQDGO0nUbu4KvYB9SKG1Th4xlGip  
MzWBWJUQXyMpgC8jeVMnqjD/GJ5FaCO4os4PygN59S+gx1piczEb+Y/ISSOhDG  
p/6z8l/b8yLlF/4ipeVjD08C7PWZGUEOfnA/CHWKZyQHg6VlnZlzxVpFZKd0fS5  
+/Vo5xbWafKT+Yqm8wIDAQAB  
-----END PUBLIC KEY-----

Résultats:  
-> Riz : 2  
-> Pates : 0

## Document PDF récapitulatif du scrutin



Exemple de notification d'ouverture ou de clôture de scrutin

## Résultats et Conclusion :

J'ai beaucoup apprécié ce projet. Il m'a permis d'aborder le concept de développement back-end, qui m'était, dans l'implémentation, inconnu jusqu'à présent.

Le résultat me convient, j'ai pu implémenter toutes les fonctionnalités monôme et certaines supplémentaires.

J'ai également pu y intégrer deux fonctionnalités web dont je connaissais sur certains sites : les blobs et les notifications avec service workers.