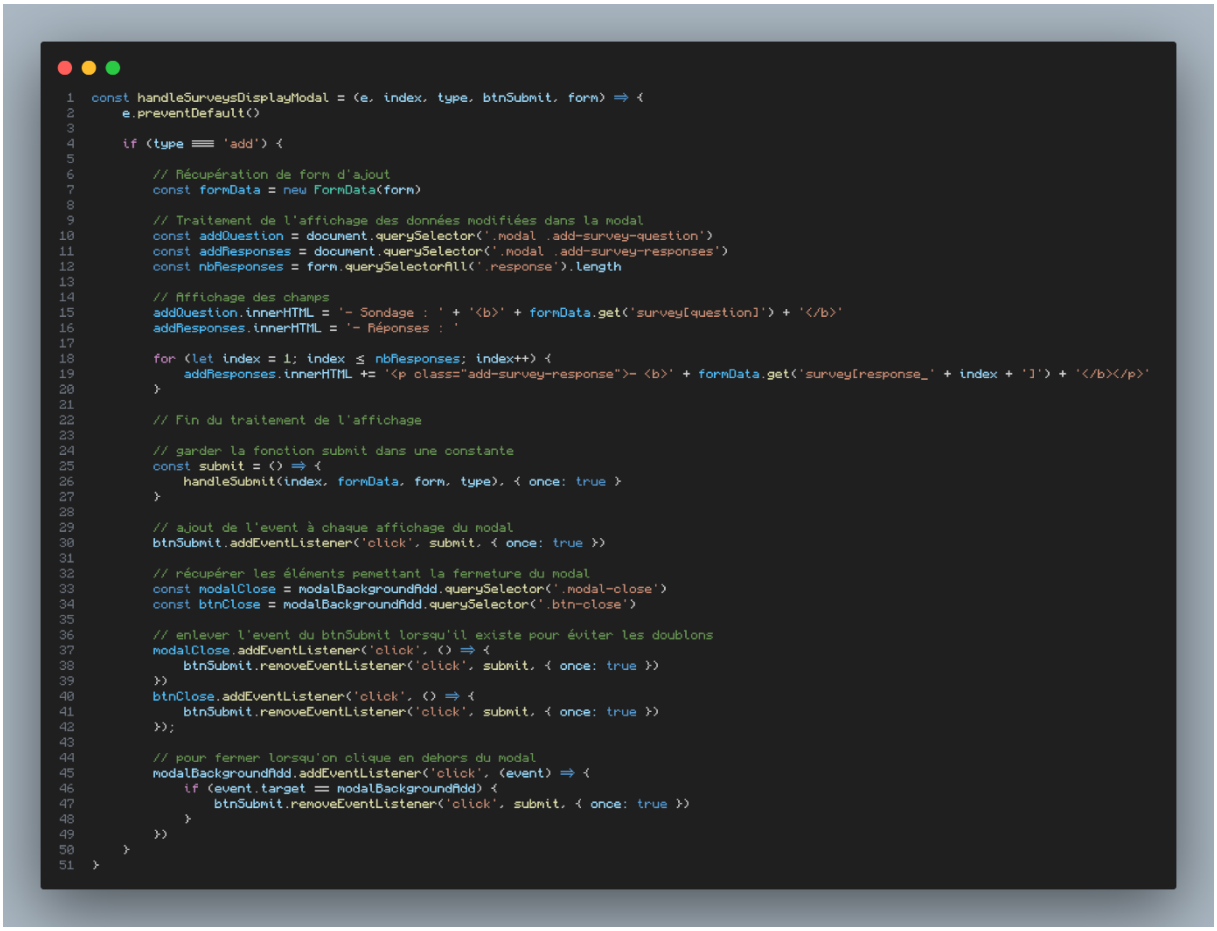
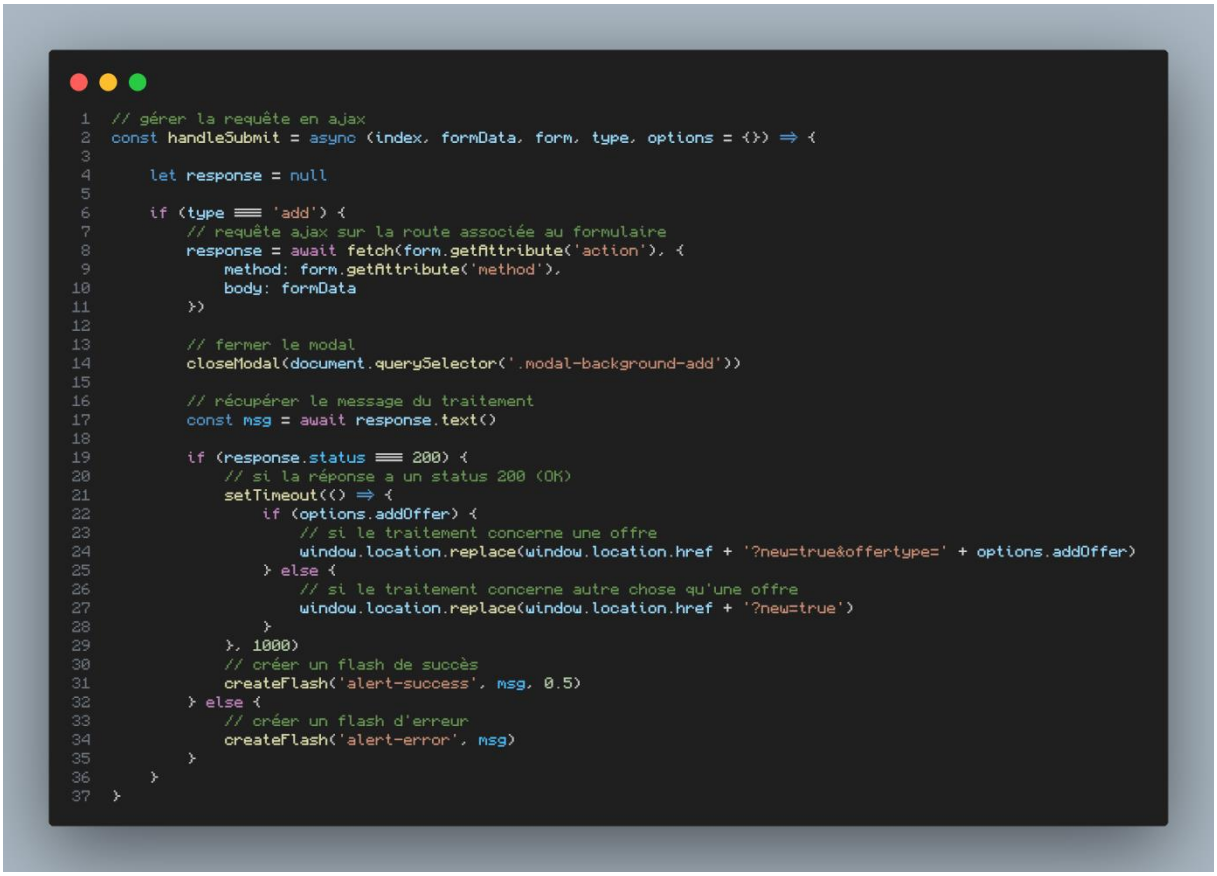
Details d’une tâche : ajout d’un nouveau sondage

Au clic de l’ajout d’un formulaire, ce code JavaScript s’exécute :



```javascript
const handleSurveysDisplayModal = (e, index, type, btnSubmit, form) => {
    e.preventDefault()

    if (type === 'add') {

        // Récupération de form d'ajout
        const formData = new FormData(form)

        // Traitement de l'affichage des données modifiées dans la modal
        const addQuestion = document.querySelector('.modal .add-survey-question')
        const addResponses = document.querySelector('.modal .add-survey-responses')
        const nbResponses = form.querySelectorAll('.response').length

        // Affichage des champs
        addQuestion.innerHTML = '- Sondage : ' + '<b>' + formData.get('survey[question]') + '</b>'
        addResponses.innerHTML = '- Réponses : '

        for (let index = 1; index <= nbResponses; index++) {
            addResponses.innerHTML += '<p class="add-survey-response">- <b>' + formData.get('survey[response_' + index + ']') + '</b></p>'
        }

        // Fin du traitement de l'affichage

        // garder la fonction submit dans une constante
        const submit = () => {
            handleSubmit(index, formData, form, type), { once: true }
        }

        // ajout de l'event à chaque affichage du modal
        btnSubmit.addEventListener('click', submit, { once: true })

        // récupérer les éléments permettant la fermeture du modal
        const modalClose = modalBackgroundAdd.querySelector('.modal-close')
        const btnClose = modalBackgroundAdd.querySelector('.btn-close')

        // enlever l'event du btnSubmit lorsqu'il existe pour éviter les doublons
        modalClose.addEventListener('click', () => {
            btnSubmit.removeEventListener('click', submit, { once: true })
        })
        btnClose.addEventListener('click', () => {
            btnSubmit.removeEventListener('click', submit, { once: true })
        });

        // pour fermer lorsqu'on clique en dehors du modal
        modalBackgroundAdd.addEventListener('click', (event) => {
            if (event.target == modalBackgroundAdd) {
                btnSubmit.removeEventListener('click', submit, { once: true })
            }
        })
    }
}
```

Ce code appelle une fonction commune à tous les CRUD (handleSubmit) dont voici le code :



```
// gérer la requête en ajax
const handleSubmit = async (index, formData, form, type, options = {}) => {

    let response = null

    if (type === 'add') {
        // requête ajax sur la route associée au formulaire
        response = await fetch(form.getAttribute('action'), {
            method: form.getAttribute('method'),
            body: formData
        })

        // fermer le modal
        closeModal(document.querySelector('.modal-background-add'))

        // récupérer le message du traitement
        const msg = await response.text()

        if (response.status === 200) {
            // si la réponse a un status 200 (OK)
            setTimeout(() => {
                if (options.addOffer) {
                    // si le traitement concerne une offre
                    window.location.replace(window.location.href + '?new=true&offertype=' + options.addOffer)
                } else {
                    // si le traitement concerne autre chose qu'une offre
                    window.location.replace(window.location.href + '?new=true')
                }
            }, 1000)
            // créer un flash de succès
            createFlash('alert-success', msg, 0.5)
        } else {
            // créer un flash d'erreur
            createFlash('alert-error', msg)
        }
    }
}
```

Avant d'afficher un flash afin de renseigner l'utilisateur sur l'état de l'ajout, handleSubmit demande de manière asynchrone ce résultat à la route associée au formulaire concerné (ici la route comprenant le traitement de l'ajout d'un sondage en PHP) :

```php
#[Route(path: 'ajout-sondage', name: 'post-add-survey', methods: ['POST'])]
    public function addSurvey(SurveyRepository $sureveyRepo, ResponseRepository $respRepo, Request $request, Validator $validate): Response
    {
        try {
            // créer un nouveau sondage
            $survey = new Survey();

            // formater la question pour la base de données (htmlspecialchars et trim)
            $question = $validate->transformInputString($request->get('survey')['question']);

            // vérifier si le champ question (text) est valide
            if ($validate->checkInputString($question)) {
                // renseigner la date du sondage, la question et le rendre actif
                $survey->setQuestion($question);
                $survey->setDatetime(new DateTime());
                $survey->setIsActive(true);

                // désactiver le sondage actif
                if ($sureveyRepo->findActiveSurvey() !== null) {
                    $sureveyRepo->findActiveSurvey()->setIsActive(false);
                }

                // vérification que le sondage possède au moins 2 réponses
                if (count($request->get('survey')) - 1 >= 2) {
                    $responses = [];

                    // Pour le nombre de réponse
                    for ($i = 1; $i < count($request->get('survey')); $i++) {
                        // créer une nouvelle réponse
                        $response = new SurveyResponse();
                        // formater la réponse pour la base de données (htmlspecialchars et trim)
                        $responseSurvey = $validate->transformInputString($request->get('survey')['response_' . $i]);

                        // vérifier si le champ réponse (text) est valide
                        if ($validate->checkInputString($responseSurvey)) {
                            // renseigner le champ text de la réponse et lui assigner le nouveau sondage
                            $response->setText($responseSurvey);
                            $response->setSurvey($survey);

                            // ajout de la réponses dans le tableau contenant toutes les réponses du sondage
                            $responses[] = $response;
                        } else {
                            // si la réponse n'est pas valide
                            return new Response('La réponse ' . $i . ' doit comporter uniquement des lettres et des chiffres avec au moins deux caractères.', 400);
                        }
                    }
                    // persist et flush le sondage
                    $sureveyRepo->save($survey, true);

                    // persist et flush les réponses
                    foreach ($responses as $response) {
                        $respRepo->save($response, true);
                    }

                    return new Response('L\'ajout a bien été effectué !', 200);
                } else {
                    // s'il y a moins de 2 réponses précisées
                    return new Response('Un sondage doit au moins contenir 2 réponses.', 400);
                }
            } else {
                // si la question du sondage n'est pas valide
                return new Response('La question doit comporter uniquement des lettres et des chiffres avec au moins deux caractères.', 400);
            }
        } catch (\Throwable $th) {
            // s'il y a une erreur lors du traitement
            return new Response('Une erreur imprévue est survenue, veuillez recharger la page et réessayer.', 400);
        }
    }
```

Ce traitement appelle un service « Validator » qui s'occupe de vérifier la validité des champs et leur format :

```php
<?php

namespace App\Service;

class Validator
{
    public function checkInputString(string $name): bool
    {
        // limiter les caractères speciaux et avoir au moins 2 caractères
        $regexInputString = '/^[a-zñ-Z0-9^@&"().!_$£`~-°x#,%*µ§:{}+=\/;? #]{2,}$/i';
        if (preg_match($regexInputString, $name)) {
            return True;
        }
        return False;
    }

    public function transformInputString(string $string): string
    {
        // formater pour la base de données
        return trim(htmlentities($string));
    }
}
```