Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER PROFESSIONNEL (DP)

Nom de naissance - *Collet.*
Nom d'usage - *Collet.*
Prénom - *Julien.*
Adresse - *95 rue des Sablons 73230 Saint Alban Leysse Savoie.*

Titre professionnel visé

Développeur Web et Web Mobile

MODALITÉ D'ACCÈS :

☑ Parcours de formation
☐ Validation des Acquis de l'Expérience (VAE)

Présentation du dossier

Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel. **Ce titre est délivré par le Ministère chargé de l'emploi.**

Le DP appartient au candidat. Il le conserve, l'actualise durant son parcours et le présente **obligatoirement à chaque session d'examen**.

Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.

Il est consulté par le jury au moment de la session d'examen.

Pour prendre sa décision, le jury dispose :

1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l'entretien professionnel ou de l'entretien technique ou du questionnement à partir de productions.

2. du **Dossier Professionnel** (DP) dans lequel le candidat a consigné les preuves de sa pratique professionnelle.

3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d'un parcours de formation

4. de l'entretien final (dans le cadre de la session titre).

[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels du ministère chargé de l'Emploi]

Ce dossier comporte :

- pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ;
- un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d'un titre, d'un diplôme, d'un certificat de qualification professionnelle (CQP) ou des attestations de formation ;
- une déclaration sur l'honneur à compléter et à signer ;
- des documents illustrant la pratique professionnelle du candidat (facultatif)
- des annexes, si nécessaire.

Pour compléter ce dossier, le candidat dispose d'un site web en accès libre sur le site.

http://travail-emploi.gouv.fr/titres-professionnels

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER PROFESSIONNEL (DP)

Sommaire

Exemples de pratique professionnelle

EXEMPLES DE PRATIQUE PROFESSIONNELLE

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER PROFESSIONNEL (DP)

Activité-type 1

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

CP 1 *Maquetter une application.*

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Les wireframes et user-stories présentés ont été réalisés dans le cadre d'un exercice simulant une demande client fictive à l'aide d'un enregistrement audio et un croquis.

Une **wireframe** est une représentation visuelle de la structure d'une page Web selon le support ciblé. Qui a pour but visualiser l'architecture de la page sans contrainte de design afin d'intégrer au mieux les fonctionnalités, la navigation, l'agencement des parties diverses (texte, image, liens, vidéo) pour toutes les personnes participant au développement.

Une **user story** est une phrase rédigée dans un langage simple sous forme de demande fonctionnelle exprimée par les types d'utilisateurs " `En tant que ...|Je veux ...|Afin de …`" qui a pour but de décrire avec suffisamment de précision le contenu d'une fonctionnalité. Les user-stories sont issues de la méthode Agile.

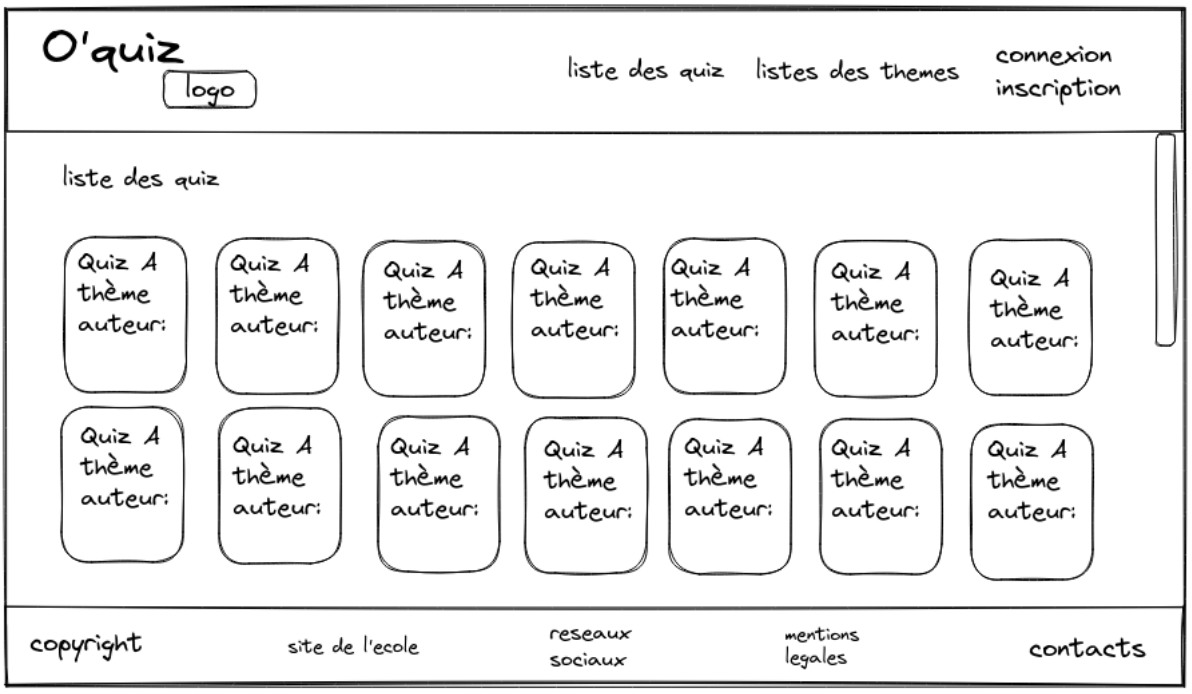
Exemple de user stories crée pour l'exercice:

En tant que ...	Je veux ...	Afin de ...
visiteur	consulter la page d'accueil	prévisualiser le contenu du site
visiteur	visualiser la liste des quizzes	pouvoir en sélectionner un
visiteur	accéder à un formulaire d'inscription	m'inscrire
visiteur	accéder à un formulaire de login	m'identifier
...	...	...
membre	supprimer mon compte	redevenir visiteur
membre	consulter mes informations personnelles	les vérifier/les modifier
membre	répondre aux quizzes	vérifier mes connaissances
...	...	...
admin	avoir un espace dédié	avoir accès aux opérations de créatioin/maintenance
admin	gérer les quizzes	en ajouter, en modifier, en supprimer
admin	gérer les thèmes	en ajouter, en modifier, en supprimer

Types d'utilisateurs de l'appli

- visiteur : utilisateur qui n'est pas connecté
- membre : utilisateur connecté avec un login/password
- admin : utilisateur connecté avec tous les pouvoirs sur le site

Exemple de wireframe créé lors de l'exercice:



DOSSIER PROFESSIONNEL (DP)

2. Précisez les moyens utilisés :

Les wireframes ont été réalisées sur le site https://excalidraw.com et les user-stories en Markdown via l'éditeur de code Visual studio code.

3. Avec qui avez-vous travaillé ?

Travail effectué en autonomie.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *O'clock promo Uther.*

Chantier, atelier, service ▸ *Exercice d'apprentissage.*

Période d'exercice ▸ Du : *03 Mai 2021* au : *15 Octobre 2021*

5. Informations complémentaires (facultatif)

Il est courant que lors de la conception de wireframes des commentaires soient ajoutés pour simplifier la compréhension des interactions possibles et souvent au sein de la méthode Agile c'est le Product Owner (chef de projet) qui se charge de les rédiger aidé soit du scrum master ou des lead developers.

Le maquettage d'une application ne s'arrête pas aux user stories et au wireframes, il existe divers principes de schématisation. Maquetter une application comme des modèles conceptuels de données MCD, la charte graphique, les mockups (prototype d'interface) et bien d'autres.

Activité-type 1

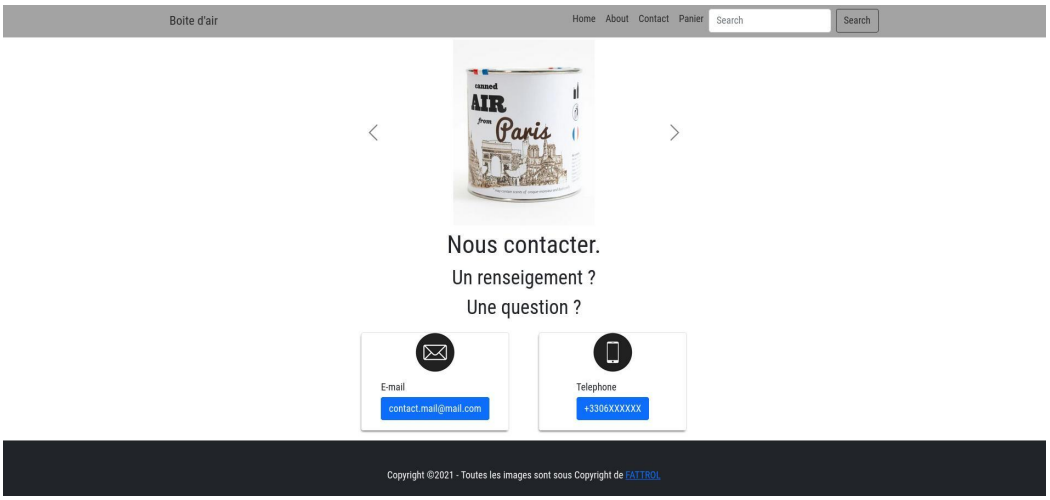
Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

CP 2 ▸ *Réaliser une interface web statique et adaptable.*

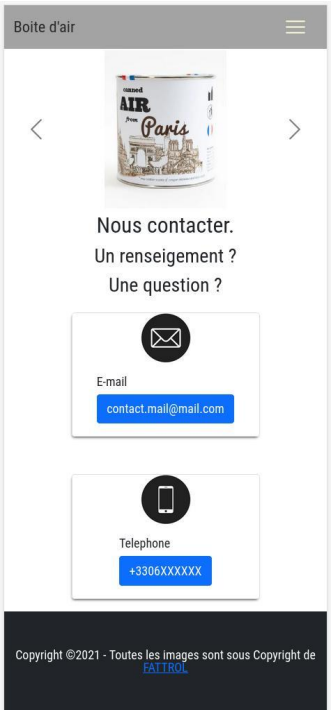
1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le cadre d'un projet personnel j'ai conçu une page web statique de contact entièrement responsive respectant la charte du site. Pour cela j'ai utilisé la bibliothèque Bootstrap pour sa prise en main relativement simple grâce à sa documentation, sa modularité et sa compatibilité avec de nombreux navigateurs ainsi que certaines bonnes pratiques web grâce au référentiel Opquast et W3C

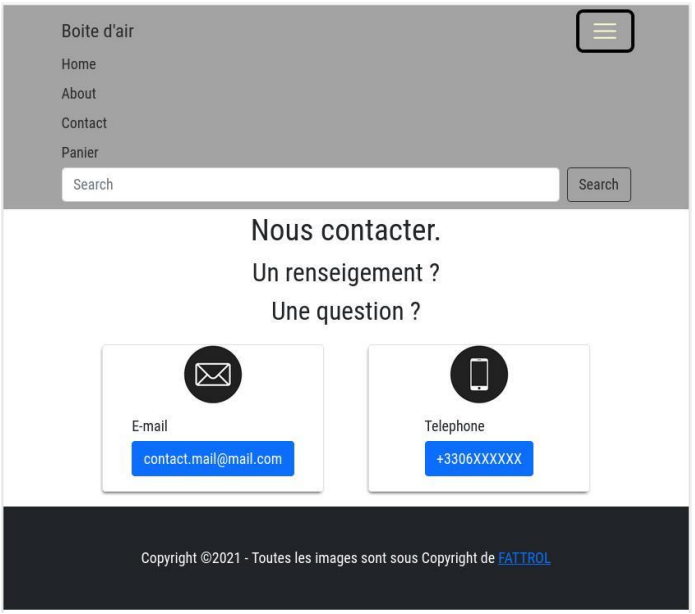
Version desktop :



Version mobile :



Version tablette :



Extrait de code HTML:

```
<main>
  <div class="row align-items-center justify-content-center">
    <h1 class="text-center fs-1 lh-base" >Nous contacter.</h1>
    <span class="text-center fs-2 lh-base">Un renseigement ?</span>
    <span class="text-center fs-2 lh-base">Une question ?</span>
  </div>

  <div class="row align-items-center justify-content-center gap-5 my-3">
    <div class="card col-md-6">
      <img src="images/Screenshot_2021-11-17_13-49-33 1mailIcon.jpg" class="card-img-top rounded
          mx-auto d-block" alt="logo e-mail">
      <div class="card-body mx mx-auto">
        <address>
          <h2 class="card-title">E-mail</h2>
          <a href="mailto:contact.mail@mail.com" class="btn btn-primary">contact.mail@mail.com</a>
        </address>
      </div>
    </div>
    <div class="card col-md-6">
      <img src="images/Screenshot_2021-11-17_13-50-24 1phoneIcon.jpg" class="card-img-top rounded
          mx-auto d-block" alt="logo de telephone">
      <div class="card-body mx-auto">
        <address>
          <h2 class="card-title">Telephone</h2>
        <a href="tel:+3306XXXXXX" class="btn btn-primary ">+3306XXXXXX</a>
        </address>
      </div>
    </div>
  </div>
</main>
```

Le HTML a été écrit pour se rapprocher au plus des exigences W3C en utilisant la balise <main> pour le contenu principal, les balises <address> pour contenir les adresse de contact et l'ordre des titres <h1> puis <h2>.

De nombreuses et class sont utilisées pour le fonctionnement de Bootstrap comme la class <div class=”col-md-6” qui permet de placer cette div en colonnes au point de rupture ≥768px avec sa valeur d'encapsulage ce qui rend le tout responsive grâce la parente qui applique le concept flexbox.

Extrait de code CSS :

```
body {
  padding-top: 65px;
  font-family: 'Roboto Condensed', sans-serif;
}

:focus-visible {
  outline: 1px solid black;
  border-radius: 5px;
}

.form-control:focus-visible, .btn:focus-visible, :focus-visible {
  border-color: #030303;
  box-shadow: inset 0 1px 1px rgba(0, 0, 0, 0.267), 0 0 8px rgba(0, 0, 0, 0.267);
}

.navbar {
  background-color:rgb(163, 163, 163);
}

.navbar a, .card-name, .link-add-bookmarks {
  color :rgb(43, 43, 43);
}
```

Les couleurs de la navbar et du texte ont été choisies pour respecter un ratio de contraste convenable de 5,61.

Un des défauts de Bootstrap est son design générique et son focus clavier qui est difficilement visible en termes d'accessibilité au focus clavier. les sélecteurs .form-control:focus-visible et btn:focus-visible permet de modifier uniquement le focus du clavier de façon plus claire .

Avant:

Après:

2. Précisez les moyens utilisés :

La bibliothèque Bootstrap a été utilisée principalement en complément de HTML5 et d'un peu de CSS.
Pour les bonnes pratiques du web, le référentiel Opquast, Web Content Accessibility Guidelines (WCAG) 2.1, Accessible Rich Internet Applications (WAI-ARIA) 1.1 et divers guides W3C ont servi de support.

3. Avec qui avez-vous travaillé ?

Travail effectué en autonomie.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *O'clock promo Uther.*

Chantier, atelier, service ▸ *Travail personnel*

Période d'exercice ▸ Du : *03 Mai 2021* au : *15 Octobre 2021*

5. Informations complémentaires (facultatif)

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ DE L'EMPLOI

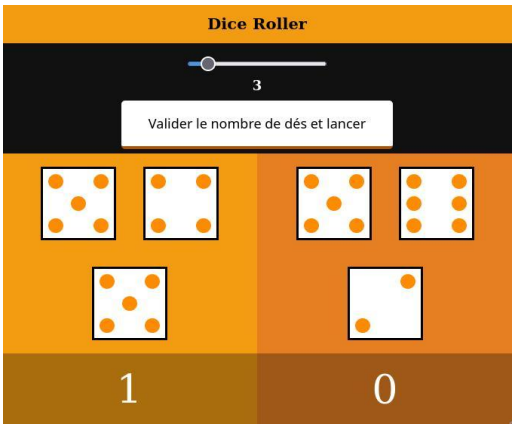
DOSSIER PROFESSIONNEL (DP)

Activité-type 1

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

CP 3 - *Développer une interface utilisateur web dynamique.*

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :



Dice Roller est un jeu de lancer de dés, créé dans le cadre de ma formation.

Le but est d'effectuer des lancers contre un script JavaScript. L'utilisateur a le choix du nombre de dé à lancer a chaque tour grâce au curseur mis à disposition allant de 1 à 20 dès maximum et le score est comptabilisé à chaque tour.

Le choix du curseur a été choisi pour réduire les vulnérabilités de l'application pour éviter une saisie donnée malveillante par un tiers (attaque XXS).

La méthode init est le point névralgique de l'application.
Composé de trois duos de méthodes de ciblage d'éléments du DOM et d'écoute événement.

Le premier duo déclenche l'apparition l'interface utilisateur permettant de commencer le jeu et de sélectionner le nombre de dés à lancer.

Le second récupère cette sélection.

le troisième et le déclenchement d'un tour.

```
init: function() {
  // déclenchement de l'interface utilisateur.
  const playBtn = document.getElementById('play');
  playBtn.addEventListener('click', game.start);

  // ecoute sur le changement de valeur du curseur.
  game.diceNumberInput = document.getElementById('dice-number-input');
  game.diceNumberInput.addEventListener('input', game.changeNumber);

  // on ecoute l'envoie du formulaire pour déclencher un tour
  const gameForm = document.getElementById('game-form');
  gameForm.addEventListener('submit', game.play);
},
```

```
start: function() {
    document.getElementById('welcome').classList.add('hidden');
    document.getElementById('app').classList.remove('hidden');
  },

changeNumber: function() {
    const diceNumberElement = document.getElementById('dice-number');
    game.nbDices = game.diceNumberInput.value;
    diceNumberElement.textContent = game.nbDices;
  },

play: function(event) {
    event.preventDefault();
      if(!game.ingame) {
      game.ingame = true;
      game.reset();
      game.playerScore = game.createAllDices('player');
      game.dealerPlay();
    }
  },

createAllDices: function(player) {
    let score = 0;
    for (let nbDice = 0; nbDice < Number(game.nbDices); nbDice++) {
      const diceScore = game.createDice(player);
      score += diceScore;
    }
    return score;
  },

...
```

Lors de l'écriture le principe de séparation des concepts (SOC) a été appliqué.

Chaque action est découpée en fonction au sein d'un objet principal qui est nommé app. Une fonction définie dans un objet devient une méthode.

La méthode "play" qui était au préalable appelée dans la méthode "init", rappelle elle-même trois autres méthodes "reset","createAllDices","dealerPlay" qui à leur tour en appellent d'autres.

De cette manière chaque méthode a sa propre utilité et améliore la lisibilité et la réutilisation du code.

2. Précisez les moyens utilisés :

JavaScript, HTML, CSS et de la manipulation du DOM sont les moyens utilisé pour cette exercice

3. Avec qui avez-vous travaillé ?

Travail effectué en autonomie.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *O'clock promo Uther.*

Chantier, atelier, service ▸ *Exercice d'apprentissage.*

Période d'exercice ▸ Du : *03 Mai 2021* au : *15 Octobre 2021*

5. Informations complémentaires (facultatif)

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ DE L'EMPLOI

DOSSIER PROFESSIONNEL (DP)

Activité-type 2

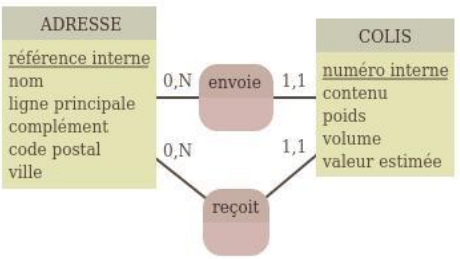
Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

CP 5 - *Créer une base de données.*

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le cadre d'un exercice effectué lors de ma formation, j'ai eu pour mission de concevoir une base de données avec la méthode **Merise** pour une start-up fictive de livraison de colis et de s'occuper de son évolution.

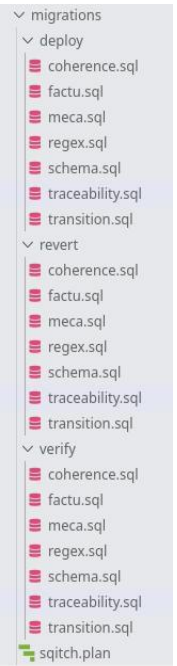
Pour débuter le processus de conception j'ai fait un premier **modèle conceptuel de données (MCD).**



Nous avons 2 **entités** : ADRESSE et COLIS.
Chaque entité possède des **attributs** et 1 **déterminant** chacun qui permet de les identifier de manière unique.

Nos 2 **entités** ont besoin de 2 **associations** et de leurs **cardinalités**:

- une ADRESSE envoie au minimum 0 colis et au maximum plusieurs (N) COLIS.
- une ADRESSE reçoit au minimum 0 colis et au maximum plusieurs (N) COLIS.
- un COLIS s'envoie au minimum à 1 et au maximum 1 ADRESSE.
- un COLIS se reçoit au minimum à 1 et au maximum 1 ADRESSE.



Ensuite nous écrivons le **modèle logique de données (MLD)**

Adresse (référence interne, nom, ligne principale, complément, code postal, ville).
Colis (numéro interne, contenu, poids, volume, valeur estimée,#adresseEnvoie, #adresseReçoit).

Étant donné que cette BDD a pour but d'évoluer un outil de versionning est utilisé, Sqitch.

Pour chaque nouvelle modification du **Data Definition Language** (DDL) avec Sqitch nous devons créer trois scripts SQL.

1. **Deploy** : la nouvelle définition de données.
2. **Revert** : script SQL permettant de supprimer les modifications effectuées.
3. **Verify** : script SQL pour test que les modifications du deploy ont fonctionné.

Par précaution nous pouvons faire un export de la base de données avant toute modifications:

-pg_dump -d <nom_base> --column-inserts [-U <nom_user>]

Exemple de **deploy** ou je crée une nouvelle table, j'ajoute une clef-étrangère dans la table "package", j'effectue les transferts de données et supprime la colonne qui n'est plus utile.

```
BEGIN;

CREATE TABLE expedition (
    id int GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    driver_name text NOT NULL,
    vehicle_plate text NOT NULL,
    starting_time TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    ending_time TIMESTAMPTZ

);

ALTER TABLE package
    ADD COLUMN expedition_id INT REFERENCES expedition(id);

INSERT INTO expedition (driver_name, vehicle_plate, starting_time)
    SELECT DISTINCT 'inconnu', 'inconnu', expedition_time FROM package WHERE expedition_time IS NOT
NULL;

UPDATE package SET expedition_id =
    (SELECT id FROM expedition WHERE starting_time = package.expedition_time);

ALTER TABLE package
    DROP expedition_time;


COMMIT;
```

Exemple du **Revert,** on recrée la colonne supprimée avec sa contrainte, on transfère les données à l'intérieur et je supprime la colonne et la table du deploy.

```
BEGIN;

ALTER TABLE package
    ADD COLUMN expedition_time TIMESTAMPTZ;

UPDATE package SET expedition_time =
    (SELECT starting_time FROM expedition WHERE id=package.expedition_id);


ALTER TABLE package
    DROP expedition_id;

DROP TABLE expedition;

COMMIT;
```

Exemple du **Verify** ou je fait des requête à la base de donnée avec en condition “WHERE false” pour qu’elle ne retourne aucun donnée:

```
BEGIN;

SELECT id FROM expedition WHERE false;

SELECT expedition_id FROM package WHERE false;


ROLLBACK;
```

2. Précisez les moyens utilisés :

J’ai utilisé Mocodo pour le MCD, Sqitch comme outils de versionning et Postgresql comme SGBD.

3. Avec qui avez-vous travaillé ?

Travail effectué en autonomie.

4. Contexte

Nom de l’entreprise, organisme ou association ▸ *O’clock promo Uther.*

Chantier, atelier, service ▸ *Exercice d’apprentissage.*

Période d’exercice ▸ Du : *03 Mai 2021* au : *15 Octobre 2021*

5. Informations complémentaires (facultatif)

Activité-type 2

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

CP 6 ▸ *Développer les composants d'accès aux données.*

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Lors d'un exercice en équipe, j'ai eu pour mission de concevoir une partie d'une application back-office pour la maintenance d'attractions.

La communication avec la base de données est assurée par la bibliothèque **node-postgres** au sein d'un fichier database.js.

```
const { Pool } = require('pg');


const pool = new Pool({
    connectionString: process.env.DATABASE_URL
});

module.exports = pool;
```

Les requêtes à la base de données sont assurées par les models sous pattern Active Record comme ORM (Object Relational Mapping).

Créations la class "Issue" avec son constructeur qui permet lors de son instanciation de créer autant de paires propriété/valeur souhaitées. Ensuite je crée une Méthode "static" asynchrone "findOne" qui permet de faire une promesse qui se charge d'une requête SQL avec en paramètre un "id" qui sera la query parametre de la requête HTTP .

Dans cette méthode on utilise l'instruction "try...catch"
pour la gestion d'erreurs. Le try est constitué d'une déstructuration de la réponse à notre promesse pour récupérer uniquement les infos voulues. "rows" est un tableau d'objet, chaque objet correspond à un enregistrement fourni par la requête SQL donc je "return" le premier index de ce tableau au controller sinon je renvoie "null".

```
class Issue {

    constructor(obj = {}) {
        for (const propName in obj) {
            this[propName] = obj[propName];
        }
    }

  static async findOne(id) {
        try {
            const query_db = `
                SELECT issue.id, issue.type_issue, issue.operator,
                issue.report_time, issue.resolution_date,
                attraction.name, issue.attraction_id
                FROM support_schema.issue
                JOIN support_schema.attraction
                ON attraction.id = issue.attraction_id
                WHERE issue.id = $1
            `;
            const { rows } = await db.query(query_db, [id])
            if (rows[0]) {
                return new Issue(rows[0])
            }
            return null;
        } catch (error) {
            if (error.detail) {
                throw new Error(error.detail)
            } else {
                throw error;
            }
        }
    }
```

La méthode "findOne" possède une requête SQL préparée pour se prémunir d'une injection SQL.

```
SELECT * FROM support_schema.issue
JOIN support_schema.attraction
ON attraction.id = issue.attraction_id
WHERE resolution_date IS NULL;
```

Dans cette requête je demande à Postgres de me fournir toute les colonnes de la table virtuelle qu'il va créer. "JOIN" me permet d'associer plusieurs tables par rapport a une valeur commune."WHERE" est une condition, je lui indique de me donner uniquement les lignes qui n'a pas de valeur dans la colonne "resolution_date".

Les tables utilisées dans cette requête on été lors de leur création placées dans des schémas, pour créer une liste de contrôle d'accès (**Access Control List - ACL)** ce qui permet de contrôler l'accès et l'écriture aux données selon le client.

2. Précisez les moyens utilisés :

La librairie node-postgres, Javascript, Postgresql et SQL.

3. Avec qui avez-vous travaillé ?

Seul pour cette partie du projet.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *O'clock promo Uther.*

Chantier, atelier, service ▸ *Exercice test des connaissances.*

Période d'exercice ▸ Du : *03 Mai 2021* au : *15 Octobre 2021*

5. Informations complémentaires (facultatif)

Activité-type 2

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité.

CP 7 ▸ *Développer la partie back-end d'une application web ou web mobile.*

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans le même exercice présenté dans le CP-6 j'ai conçu une modèle **MVC** (model, view, controller) en **programmation orienté objet** (POO) dans une **application, interface de programmation** (API).

Le **point d'entrée** de l'API est situé à la racine du projet index.js. À l'intérieur, j'importe toutes les méthodes, librairies ou bibliothèques dont j'ai besoin.

"dotenv" pour les variables d'environnements qui contiennent les données sensibles (adresse BDD, port, mots de passe, identifiant), placées dans le .gitignore pour ne pas être publiées en ligne avec l'outil Git.

On configure un moteur de rendu avec PUG pour générer les **views**.

Trois middlewares sont mis en place:

- activation du **CORS** pour limiter l'accès à l'API aux adresses souhaitées.
- le body-parser d'express pour lire les body des requêtes POST/PUT
- middleware json pour reconnaître le format JSON dans les requêtes HTTP

```
require('dotenv').config();
const express = require('express');
const router = require('./app/router');
const app = express();
const cors = require('cors');

app.set("views", "./view");
app.set("view engine", "pug");

app.use(cors({
  'Access-Control-Allow-Origin':"http://localhost"
  }));

app.use(express.urlencoded({ extended: true }));
app.use(express.json());

app.use(router);

const port = process.env.PORT;

app.listen(port, () => {
  console.log(`Server started on http://localhost:${port}`);
});
```

Ensuite j'appelle le routeur qui a est factorisé et la méthode "app.listen" qui lance la connexion d'Express.

```
const { Router } = require('express');
const router = Router();
const issueController = require('./controllers/issueController');


router.get('/', issueController.findAll);

router.route('/incident/:id(\\d+)')
    .get(issueController.findOne)
    .post(issueController.save);

router.route('/incident/new')
    .get(issueController.displayFormCreateIssue)
    .post(issueController.createIssue);

module.exports = router;
```

Le **routeur** sert à diriger chaque requête faite à l'API au bon controller avec la méthode souhaitée.

Une fonction router est constituée de l'adresse URL dans laquelle on peut inclure une expression régulière pour ajouter un niveau de sécurité à l'API. Du Verbe HTTP qui lui est destiné et le controller/méthode à qui il doit transmettre la requête.

Le **controller** est la partie maîtresse du modèle MVC. C'est lui qui traite les requêtes, les données reçues et les renvoie au bon endroit.

La méthode asynchrone "findOne" reçoit dans son try la "request" (requête HTTP) dans laquelle on utilise que le paramètre ID de sa Query qui est envoyé dans le **modèle** comme vu lors de la CP 6.

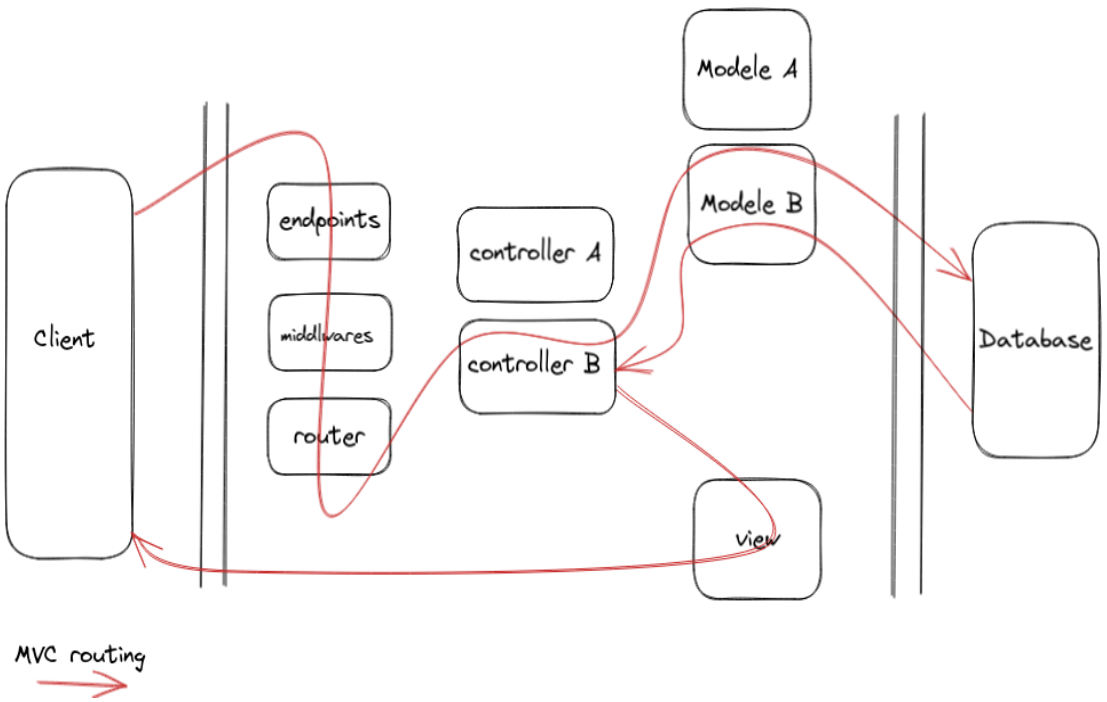
Une fois la promesse résolue (await), la réponse du modèle stockée dans la constante "issue" est rendue à PUG préalablement configuré pour générer une **view** avec les données récupérées en base.

```
const { json } = require('express');
const Issue = require('../models/issue');
const Attraction = require('../models/attraction')

const issueController = {
    findAll: async (_, response) => {
        try {
            const issues = await Issue.findAll();
            response.render('listing-issue', { issues })
        } catch (error) {
            response.status(500).send(error.message);
        }
    },

    findOne: async (request, response) => {
        try {
            const issue = await Issue.findOne(
              parseInt(request.params.id, 10));
            response.render('detail-issue', { issue });
        } catch (error) {
            response.status(500).send(error.message);
        }
    },

  ...
```

Exemple de la view PUG :

```
doctype html
html(lang="fr")
  head
    title= DetailIncident
    script(type='text/javascript').
  body
    h1 Back office /GET detail un incident

    h2 Incident numéro #{issue.id} :  #{issue.type_issue}

    if (issue.resolution_date)
        p Incident fermé le #{issue.resolution_date}
    else
        p Incident en cours de réparation

    p Attraction concernée: #{issue.name}
    p Problème repéré le : #{issue.report_time}

    form(role='form' method='POST' action=`/incident/${issue.id}`)
        div
                | Opérateur:
                input(type='text', value=issue.operator name='operator')

        if (!issue.resolution_date)
            div
                    | Fermer l'incident :
                    input(type='checkbox' name='closedIssue' checked=false)


        button(type="submit") Submit
```

Représentation du cheminement de la donnees :



Sécurité:

Lors du développement de cette API quelques règles de sécurité ont été appliquées.

CORS:

Les navigateurs interdisent les requêtes d'origines différentes que celles du site de d'origine par défaut pour des raisons de sécurité. CORS permet de contourner cette interdiction et offre un mécanisme qui trie les entêtes HTTP et déterminent s'il faut ou non bloquer les requêtes.

Injections SQL :

L'injection SQL est une faille de sécurité très basique. Elle consiste lors d'une requête d'introduire un code SQL qui peut compromettre la sécurité de la base de données ou déclencher un comportement non désiré. Pour éviter cela nous pouvons échapper tout caractère spéciaux en amont de la requête SQL ou utiliser des requêtes préparées car la requête est compilée avant de recevoir son paramètre ce qui rend le code malveillant inefficace.

Il existe d'autre faille de sécurité bien connu comme les attaques **XSS** (Cross-site scripting)

- attaques XSS stockées - contenu malicieux envoyé dans une app web, qui va le stocker et retourné dans le navigateur des autres utilisateurs.
- attaques XSS reflétées - contenu malicieux envoyé via l'url pour permettre d'attaquer directement le serveur.
- attaques XSS basées sur le DOM- similaire à l'attaque XSS reflétées mais qui vise directement le DOM.

Les moyens de se prémunir de ces attaques est de nettoyer et valider chaque donnée entrante voir même les encoder.

Les attaques **CSRF** (Cross Site Request Forgery) sont très courantes et ont pour principe "d'usurper votre identité" mais avec votre vraie identité par exemple forcer le navigateur de la victime à exécuter des tâches indésirables. Pour contrer ces attaques en tant que développeurs nous pouvons mettre en place un système d'authentification supplémentaire pour les actions sensibles.

- Double authentification
- Nouvelle saisie des identifiants
- Validation captcha
- Token à usage unique
- Token de synchronisation

2. Précisez les moyens utilisés :

Javascript, node.js, SQL, PUG, Express.js sont les moyens utilisés dans cette compétence.

3. Avec qui avez-vous travaillé ?

Seul pour cette partie du projet.

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *O'clock promo Uther.*

Chantier, atelier, service ▸ *Exercice test des connaissances..*

Période d'exercice ▸ Du : *03 Mai 2021* au : *15 Octobre 2021*

5. Informations complémentaires (facultatif)

Titres, diplômes, CQP, attestations de formation

(facultatif)

Intitulé	Autorité ou organisme	Date
Cliquez ici.	Cliquez ici pour taper du texte.	Cliquez ici pour sélectionner une date.

Déclaration sur l'honneur

Je soussigné(e) [prénom et nom] ,

déclare sur l'honneur que les renseignements fournis dans ce dossier sont exacts et que je suis l'auteur(e) des réalisations jointes.

Fait à le

pour faire valoir ce que de droit.

Signature :

Documents illustrant la pratique professionnelle

(facultatif)

Intitulé
Cliquez ici pour taper du texte.

ANNEXES

(Si le RC le prévoit)