Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER PROFESSIONNEL (DP)

Nom de naissance - *Verschaere*
Nom d'usage - *Verschaere*
Prénom - Damien
Adresse - 51 Rue de Crimée 13003 Marseille

Titre professionnel visé

Concepteur Développeur d'Applications

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

Sommaire

Exemples de pratique professionnelle

EXEMPLES DE PRATIQUE PROFESSIONNELLE

DOSSIER PROFESSIONNEL (DP)

Activité-type 1

Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité

Exemple n°1 - *Share event - Application mobile*

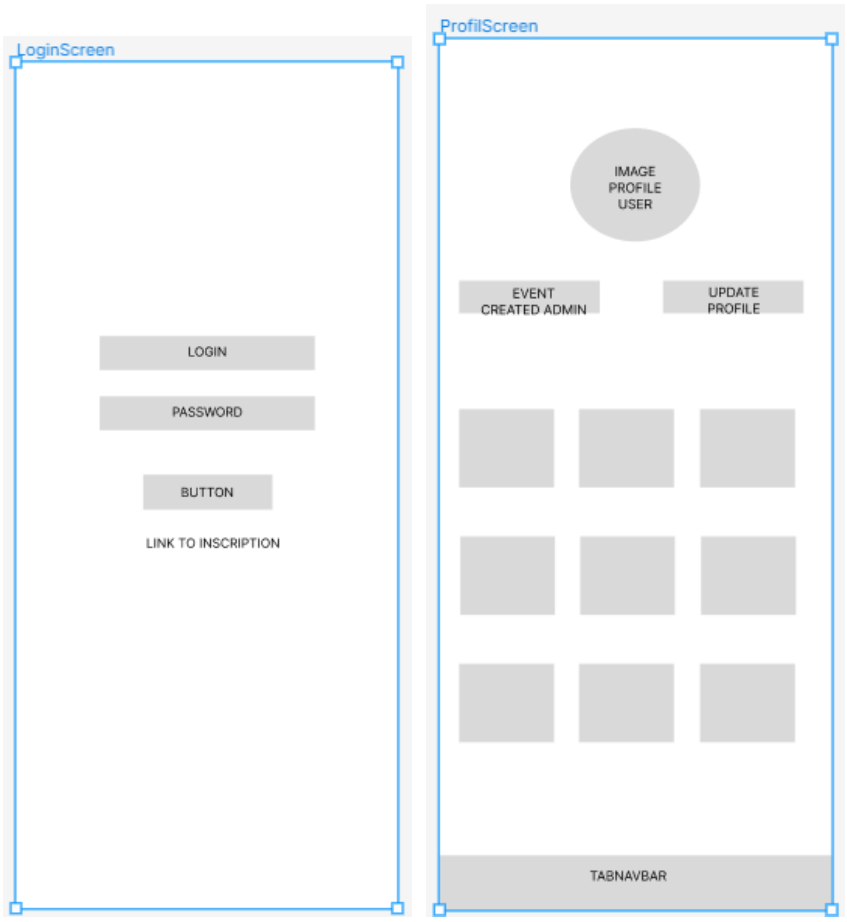
1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Ce projet a été réalisé dans le cadre de ma formation pour passer mon titre :
il a été réalisé en équipe et en télétravail

2. Précisez les moyens utilisés :

Figma a été notre outil pour cette tâche :
Dans un premier temps nous avons réalisé le wireframe de notre application qui est un version basique du design de l'app permettant de placer les différents composants dans l'écran.

Ci-dessous le wireframe du screen Login ainsi que le screen User profil



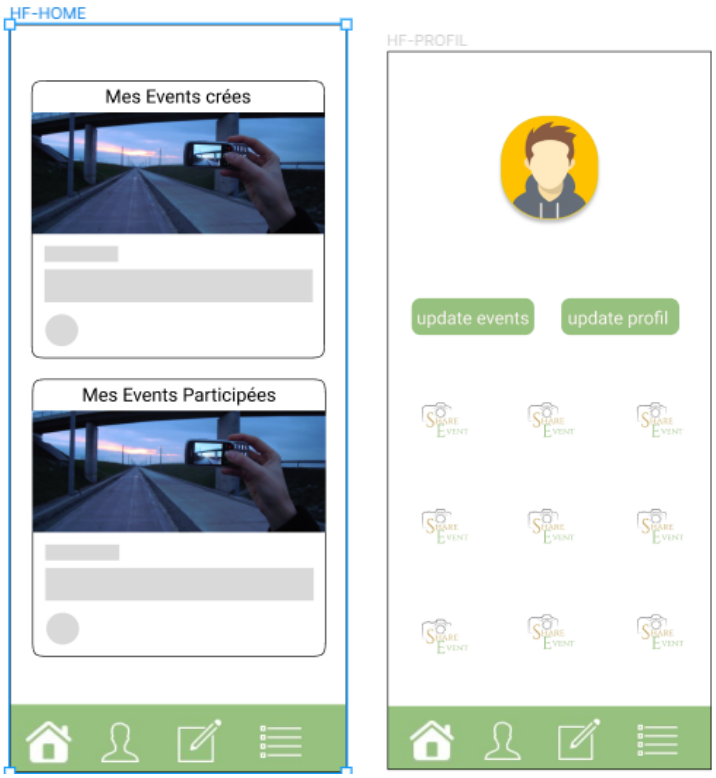
Le wireframe réalisé nous avons défini une charte graphique pour notre app

Couleur principale : #97C17E Couleur secondaire : #E6C821

Couleur de fond : #FCFCFC

Police pour les titres et entête : CINZEL Police pour le reste du texte : open sans

La charte graphique définie nous passons à la maquette haute fidélité qui sera le rendu finale de l'app



Cette étape dans la conception de l'application ma permis de valider la compétence

-**Maquetter une application**

3. Avec qui avez-vous travaillé ?

4. Contexte

Nom de l'entreprise, organisme ou association ▸ La plateforme

Chantier, atelier, service ▸ *Cliquez ici pour taper du texte.*

Période d'exercice ▸ Du : *02/01/2023i* au : *31/08/2023a*

5. Informations complémentaires *(facultatif)*

Activité-type 1

Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité

Exemple n° 2 - *Share event - Application mobile*

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

Ce projet est réalisé dans le cadre de ma formation en équipe :

2. Précisez les moyens utilisés :

Au cours de notre projet Share Event, nous avons travaillé sur le développement de la partie front-end de l'interface utilisateur en utilisant React Native et TypeScript. Notre but était de fournir une interface utilisateur claire et intuitive pour nos utilisateurs.

Pour commencer, nous avons utilisé le système de navigation de React Navigation pour mettre en place la navigation entre les différents écrans de notre application. Nous avons principalement utilisé deux types de navigation : le Stack Navigator et le Bottom Tab Navigator.

```
const Tab = createBottomTabNavigator();
const Stack = createStackNavigator();
```

```
import { createStackNavigator } fro
import { createBottomTabNavigator }
```

Le Stack Navigator nous a permis de gérer une pile d'écrans où chaque nouvel écran est placé sur le dessus de la pile. Par exemple, nous l'avons utilisé pour la navigation entre l'écran d'accueil, l'écran de l'événement et l'écran de téléchargement :
createStackNavigator nous a permis de mettre en place une navigation entre les différents écrans de l'application. Cette bibliothèque a rendu possible la création de piles de navigation, où l'utilisateur peut naviguer en avant vers un nouvel écran, puis en arrière vers l'écran précédent.

```
function ProfilStack() {
  return (
    <Stack.Navigator initialRouteName="Profile">
      <Stack.Screen name="Profile" component={ProfilScreen} options={{ headerShown: false }} />
      <Stack.Screen name="MyEvents" component={MyEventsScreen} />
      <Stack.Screen name="MyProfil" component={UpdateProfilScreen} />
      <Stack.Screen name='ManageEvent' component={AdminEventScreen} />
    </Stack.Navigator>
  );
}
```

D'autre part, le Bottom Tab Navigator a été utilisé pour faciliter la navigation entre les différents écrans principaux de notre application. Voici un exemple de comment nous avons mis en place le Bottom Tab Navigator :

```
function TabNavigator() {
  return (
    <Tab.Navigator
      initialRouteName="Home"
      screenOptions={({ route }) => ({
        headerShown: false,
        tabBarIcon: ({ focused, color, size }) => {
          let iconName;
          if (route.name === 'Home') {
            iconName = (
              <Entypo name="home" size={24} color={focused ? '#CBA85F' : 'white'} />
            );
          } else if (route.name === 'Profil') {
            iconName = (
              <AntDesign name="user" size={24} color={focused ? '#CBA85F' : 'white'} />
            );
          } else if (route.name === 'PublicEvents') {
            iconName = (
              <Entypo name="list" size={24} color={focused ? '#CBA85F' : 'white'} />
            );
            } else if (route.name === 'CreateEvent') {
              iconName = (
                <Ionicons name="create" size={24} color={focused ? '#CBA85F' : 'white'} />
              );
          }
          return iconName;
        },
        tabBarActiveTintColor: '#CBA85F',
        tabBarInactiveTintColor: 'white',
        tabBarStyle: {
          backgroundColor: '#98A68F',
        },
      })}
    >
      <Tab.Screen name="Home" component={EventsStack} />
      <Tab.Screen name="Profil" component={ProfilStack} />
      <Tab.Screen name="PublicEvents" component={PublicEventsScreen} />
        {/* <Tab.Screen name="CreateEvent" component={CreateEventScreen} /> */}
    </Tab.Navigator>
  );
}
```

Nous avons également personnalisé les icônes de la barre de navigation en utilisant des bibliothèques d'icônes externes, à savoir AntDesign, Entypo, et Ionicons. Chaque onglet de la barre de navigation a été associé à une icône spécifique, rendant l'interface utilisateur plus attrayante et intuitive.

Dans l'ensemble, notre travail sur le développement de l'interface utilisateur a impliqué l'utilisation de nombreux aspects de React Native et de TypeScript, nous permettant de fournir une application avec une expérience utilisateur de qualité.
Nous avons également accordé une attention particulière à la gestion des données utilisateur, en particulier en ce qui concerne l'authentification des utilisateurs.

Pour ce faire, nous avons utilisé **SecureStore**, une fonctionnalité d'Expo qui fournit un moyen sécurisé de stocker des données sensibles localement sur l'appareil de l'utilisateur. En utilisant **SecureStore**, nous avons pu stocker le jeton d'accès d'un utilisateur après son authentification réussie. Ce jeton est ensuite utilisé pour valider les requêtes subséquentes de l'utilisateur, permettant ainsi une expérience utilisateur fluide tout en maintenant la sécurité.

Voici un exemple de comment nous avons utilisé **SecureStore** pour stocker le jeton d'accès :

```
const [isLoggedIn, setIsLoggedIn] = useState(false);

useEffect(() => {

  const checkIfLoggedIn = async () => {
    const token = await SecureStore.getItemAsync('acess_token');

    if (token) {
      setIsLoggedIn(true);
      console.log(token);
    } else {
      setIsLoggedIn(false);
    }
  };
  checkIfLoggedIn();

}, []);

return (
```

Parlons de la gestion de l'état de l'application. Dans notre projet, nous avons utilisé les **hooks** de React, qui sont des fonctions spéciales intégrées dans la bibliothèque React, pour gérer l'état de l'application. En particulier, nous avons fait un usage intensif de **useState** et **useEffect.**
Dans cet exemple, **isLoggedIn** est une variable d'état que nous avons initialisée à ***false***. **setIsLoggedIn** est la fonction que nous avons utilisée pour modifier l'état de **isLoggedIn**.
Nous avons également utilisé le **hook useEffect** pour vérifier si un utilisateur est connecté ou non. Cela a été réalisé en vérifiant la présence d'un **token stocké localement**. Si un **token** est présent, cela signifie que l'utilisateur est connecté, et par conséquent, nous avons utilisé **setIsLoggedIn** pour mettre à jour l'état **isLoggedIn** à ***true***. Sinon, nous avons réinitialisé l'état **isLoggedIn** à ***false***.
La gestion de **l'état** est cruciale dans une application React car elle permet à l'interface utilisateur de réagir dynamiquement aux changements d'état. Grâce à l'utilisation efficace des **hooks** de React, nous avons pu créer une interface utilisateur dynamique et réactive qui améliore considérablement

l'expérience utilisateur.
Ce projet m'a permis de valider la compétence :
Développer la partie front-end d'une interface utilisateur web

3. Avec qui avez-vous travaillé ?

J'ai travaillé en équipe avec

Raphael Diop
Yacine BOUCETTA
Rémi Garguilo

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *La plateforme*

Chantier, atelier, service ▸ *Cliquez ici pour taper du texte.*

Période d'exercice ▸ Du : *02/01/2023i* au : *31/08/2023*

5. Informations complémentaires *(facultatif)*

Activité-type 1

Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité

Exemple n° 3 - *Sokoban - Python*

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

Dans le cadre de ma formation, j'ai réalisé un projet de jeu de Sokoban en utilisant Python avec la bibliothèque Pygame.

Le jeu de Sokoban est un jeu de puzzle dans lequel le joueur doit pousser des caisses sur des emplacements cibles en utilisant un personnage. Dans cet exemple, je vais mettre l'accent sur **la compétence développer une interface utilisateur de type desktop** en utilisant la bibliothèque Pygame pour ce jeu.

Pour développer l'interface utilisateur de mon jeu de Sokoban, j'ai utilisé la bibliothèque Pygame, qui fournit des fonctionnalités pour créer des graphiques et gérer les interactions utilisateur dans les jeux. Voici quelques exemples concrets de mon travail dans le cadre de cette compétence :

Affichage graphique avec Pygame :
J'ai utilisé les fonctionnalités de Pygame pour créer un affichage graphique attrayant et interactif pour le jeu Sokoban. Voici un exemple de code qui montre comment j'ai utilisé Pygame pour afficher la grille de jeu :

```
# Définir les dimensions de la fenêtre du jeu
largeur_fenetre = 900
hauteur_fenetre = 480
fenetre = pygame.display.set_mode((largeur_fenetre, hauteur_fenetre))
pygame.mixer.music.load('ff7.ogg')
pygame.mixer.music.play(-1)
pygame.mixer.music.set_volume(0.2)
```

```
# Charger les images
image_mur = pygame.image.load('mur.png')
image_case = pygame.image.load('case.png')
image_caisse = pygame.image.load('woodbox.png')
image_caisse_ok = pygame.image.load('teleport.png')
image_personnage = pygame.image.load('personnage.png')
image_grass = pygame.image.load('grass.png')
```

```
# Dessiner la grille de jeu
for x in range(largeur_grille):
    for y in range(hauteur_grille):
        if grille.est_mur(x, y):
            fenetre.blit(image_mur, (marge_x + x * taille_case, marge_y + y * taille_case))
        elif grille.est_case(x, y):
            fenetre.blit(image_case, (marge_x + x * taille_case, marge_y + y * taille_case))
        elif grille.est_caisse(x, y):
            fenetre.blit(image_caisse, (marge_x + x * taille_case, marge_y + y * taille_case))
        elif grille.est_caisse_ok(x, y):
            fenetre.blit(image_caisse_ok, (marge_x + x * taille_case, marge_y + y * taille_case))
        elif grille.est_grass(x, y):
            fenetre.blit(image_grass, (marge_x + x * taille_case, marge_y + y * taille_case))
        else:
            fenetre.blit(image_grass, (marge_x + x * taille_case, marge_y + y * taille_case))
```

Dans cet exemple, j'ai utilisé Pygame pour créer une fenêtre de jeu, charger les images des différents éléments du jeu (murs, cases, caisses, caisses cibles, personnage), et les afficher sur la fenêtre en fonction de l'état de chaque cellule de la grille.

Interaction avec la grille de jeu :
J'ai créé une classe Grille pour représenter la grille de jeu et gérer son état. Voici un exemple de code qui montre comment j'ai interacted avec la grille :

Gestion des événements avec Pygame :
J'ai utilisé les fonctionnalités de gestion des événements de Pygame pour détecter et réagir aux actions de l'utilisateur. Voici un exemple de code qui montre comment j'ai utilisé Pygame pour gérer les événements au clavier :

```
# Boucle principale du jeu
while True:
    # Gérer les événements
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
            quit()
        elif event.type == pygame.KEYDOWN:
            if event.key == pygame.K_LEFT:
                personnage.deplacer('gauche')
            elif event.key == pygame.K_RIGHT:
                personnage.deplacer('droite')
            elif event.key == pygame.K_UP:
                personnage.deplacer('haut')
            elif event.key == pygame.K_DOWN:
                personnage.deplacer('bas')
            elif event.key == pygame.K_r:  # Ajout de la touche 'r' pour reset
                grille, personnage = init_level(level)  # Réinitialiser le niveau
```

Dans ce code, j'ai utilisé la boucle for pour parcourir tous les événements détectés par Pygame. J'ai ensuite utilisé des conditions pour vérifier le type d'événement, par exemple event.type == pygame.KEYDOWN, et réagir en conséquence. J'ai appelé les méthodes de la classe Personnage pour déplacer le personnage en fonction des touches directionnelles pressées par le joueur.

Interaction avec la grille de jeu :
J'ai créé une classe Grille pour représenter la grille de jeu et gérer son état. Voici un exemple de code qui montre comment j'ai interacted avec la grille :

```
# Vérifier si toutes les caisses sont sur des cases cibles
if grille.toutes_caisses_ok():
    # Augmenter le score
    score_obj.increase_score(100)
    # Afficher le message de victoire
    fenetre.blit(victory_text, (largeur_fenetre // 2 - victory_text.get_
    pygame.display.flip()
    pygame.time.delay(6000)
    # Initialiser le niveau suivant
    level += 1
    grille, personnage = init_level(level)
```

2. Précisez les moyens utilisés :

Dans ce code, j'ai utilisé la méthode **toutes_caisses_ok()** de la classe Grille pour vérifier si toutes les caisses sont sur des cases cibles. Si c'est le cas, j'ai augmenté le score en utilisant la méthode **increase_score()** de la classe Score et affiché un message de victoire à l'écran.

Ce projet m'a permis de valider la compétence :
développer une interface utilisateur de type desktop en utilisant la bibliothèque **Pygame**. J'ai été en mesure de créer un affichage graphique attrayant pour la grille de jeu, de gérer les événements utilisateur, d'interagir avec la grille de jeu en fonction des actions du joueur et de gérer le score du joueur.

3. Avec qui avez-vous travaillé ?

4. Contexte

Nom de l'entreprise, organisme ou association ▸ *Cliquez ici pour taper du texte.*

Chantier, atelier, service ▸ *Cliquez ici pour taper du texte.*

Période d'exercice ▸ Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires *(facultatif)*

Activité-type 1

Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité

Exemple n°5 - Share event app mobile

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

Dans le cadre de notre projet d'application mobile ShareEvent, nous avons été amenés à développer plusieurs composants d'accès aux données. L'objectif était de permettre une interaction fluide et sécurisée entre notre application et notre base de données, ainsi que la gestion sécurisée des sessions utilisateurs.

Pour commencer, nous avons interagi avec une API RESTful pour manipuler les données. Pour cela, nous avons utilisé diverses requêtes **HTTP** (**GET, POST, DELETE, etc.)** afin de créer, lire, mettre à jour et supprimer des données sur notre serveur. Par exemple, en utilisant la bibliothèque axios pour envoyer une requête **GET** à notre **API** et récupérer les données correspondantes.

Pour illustrer ce point, prenons comme exemple notre utilisation d'**Axios** pour interagir avec l'API de notre application. Nous avons notamment fait des appels GET pour récupérer les données des événements depuis le backend. Voici un exemple :

```
const getEvents = async () => {
  try {
    const response = await axios.get(
      'http://localhost:3000/api/events',
      {
        headers: {
          'Content-Type': 'application/json',
          Authorization: `Bearer ${token}`,
        },
      },
    );
    setEvents(response.data);
  } catch (error) {
    console.error(error);
  }
};
```

Ensuite, nous avons utilisé le package **SecureStore** d'Expo pour gérer de manière sécurisée les sessions utilisateurs. Dans notre application, nous avons mis en place une méthode pour vérifier si un utilisateur est connecté en cherchant un token de session dans le **SecureStore** :

```
const [isLoggedIn, setIsLoggedIn] = useState(false);

useEffect(() => {
  const checkIfLoggedIn = async () => {
    const token = await SecureStore.getItemAsync('acess_token');
    if (token) {
      setIsLoggedIn(true);
    } else {
      setIsLoggedIn(false);
    }
  };
  checkIfLoggedIn();
}, []);
```

Ces différentes pratiques nous ont permis de développer des composants d'accès aux données efficaces et sécurisés. Par cette approche, nous avons assuré une interaction sécurisée et fluide entre l'utilisateur et l'application, en intégrant les recommandations de sécurité dans notre processus de développement. Ce projet m'a permis d'acquérir la compétence :

Développer des composants d'accès aux données

2. Précisez les moyens utilisés :

J'ai utilisé pour ce projet :
-Vs code (IDE)
-React Native(framework)
-ThunderClient(extension Vs code pour tester API)
-SecureStore (API fournie par Expo)

3. Avec qui avez-vous travaillé ?

J'ai travaillé en équipe avec

Raphael Diop
Yacine BOUCETTA
Rémi Garguilo

4. Contexte

Nom de l'entreprise, organisme ou association ▸ La plateforme

Chantier, atelier, service ▸ *Cliquez ici pour taper du texte.*

Période d'exercice ▸ Du : *02/01/2023i* au : *31/08/2023*

5. Informations complémentaires *(facultatif)*

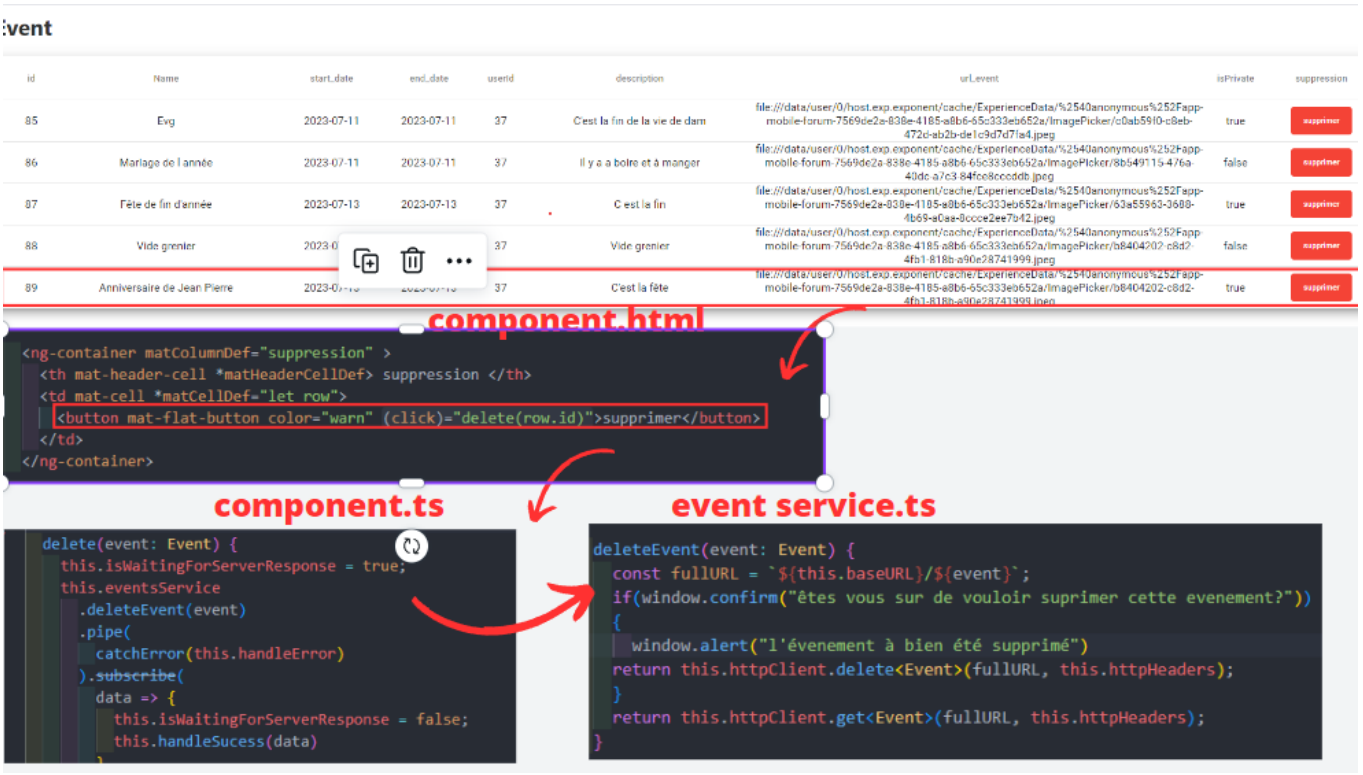
Activité-type 1

Concevoir et développer des composants d'interface utilisateur en intégrant les recommandations de sécurité

Exemple n°1

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

Pour le projet share event nous avons développé une interface web pour l'admin de l'application qui lui permet de gérer les différentes données utilisateurs .Cette interface a été développé en **Angular**.



Angular est un framework robuste et complet, idéal pour créer des interfaces complexes et riches en fonctionnalités, comme c'est souvent le cas pour les interfaces d'administration. Il offre également de nombreux outils intégrés, tels que la gestion des formulaires, la validation, le routing, et l'injection de dépendances, qui peuvent simplifier le développement.

```
export class EventService {
  private baseURL = 'http://localhost:3000/events';
  private httpHeaders = {
    headers: new HttpHeaders({ 'Content-Type': 'application/json' }),
  };

  constructor(private httpClient: HttpClient) {}

  getEvent() {
    const fullURL = `${this.baseURL}`;
    return this.httpClient.get<Event>(fullURL, this.httpHeaders);
  }

  deleteEvent(event: Event) {
    const fullURL = `${this.baseURL}/${event}`;
    if(window.confirm("êtes vous sur de vouloir suprimer cette evenement?"))
    {
      window.alert("l'évenement à bien été supprimé")
    return this.httpClient.delete<Event>(fullURL, this.httpHeaders);
    }
    return this.httpClient.get<Event>(fullURL, this.httpHeaders);
  }
```

ce projet m'a permis d'acquérir les compétences :

Développer la partie front-end d'une interface utilisateur web & Développer la partie back-end d'une interface utilisateur web

2. Précisez les moyens utilisés :

3. Avec qui avez-vous travaillé ?

4. Contexte

DOSSIER PROFESSIONNEL (DP)

Nom de l'entreprise, organisme ou association ▸ La plateforme

Chantier, atelier, service ▸ *Cliquez ici pour taper du texte.*

Période d'exercice ▸ Du : *02/01/2023i* au : *31/08/2023a*

5. Informations complémentaires *(facultatif)*

Activité-type 2

Concevoir et développer la persistance des données en intégrant les recommandations de sécurité

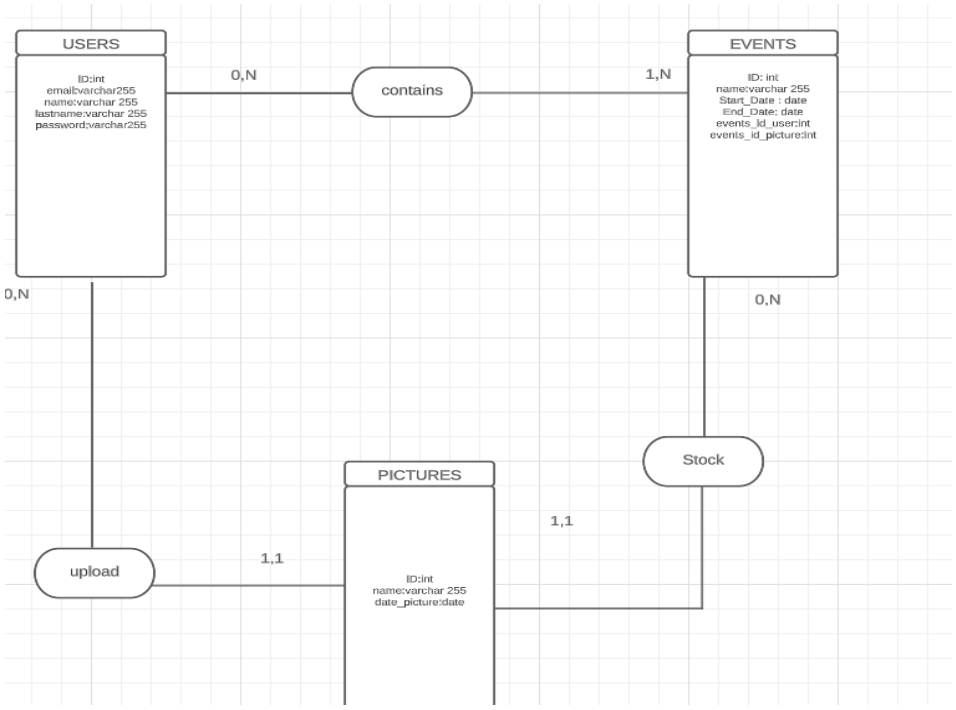
Exemple n°1 - *Share event application mobile*

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

Dans le cadre de notre projet d'application mobile ShareEvent, nous avons utilisé l'outil de modélisation Lucidchart pour créer nos **modèles conceptuels de données (MCD)**, **modèles logiques de données (MLD) et modèles physiques de données (MPD)**.

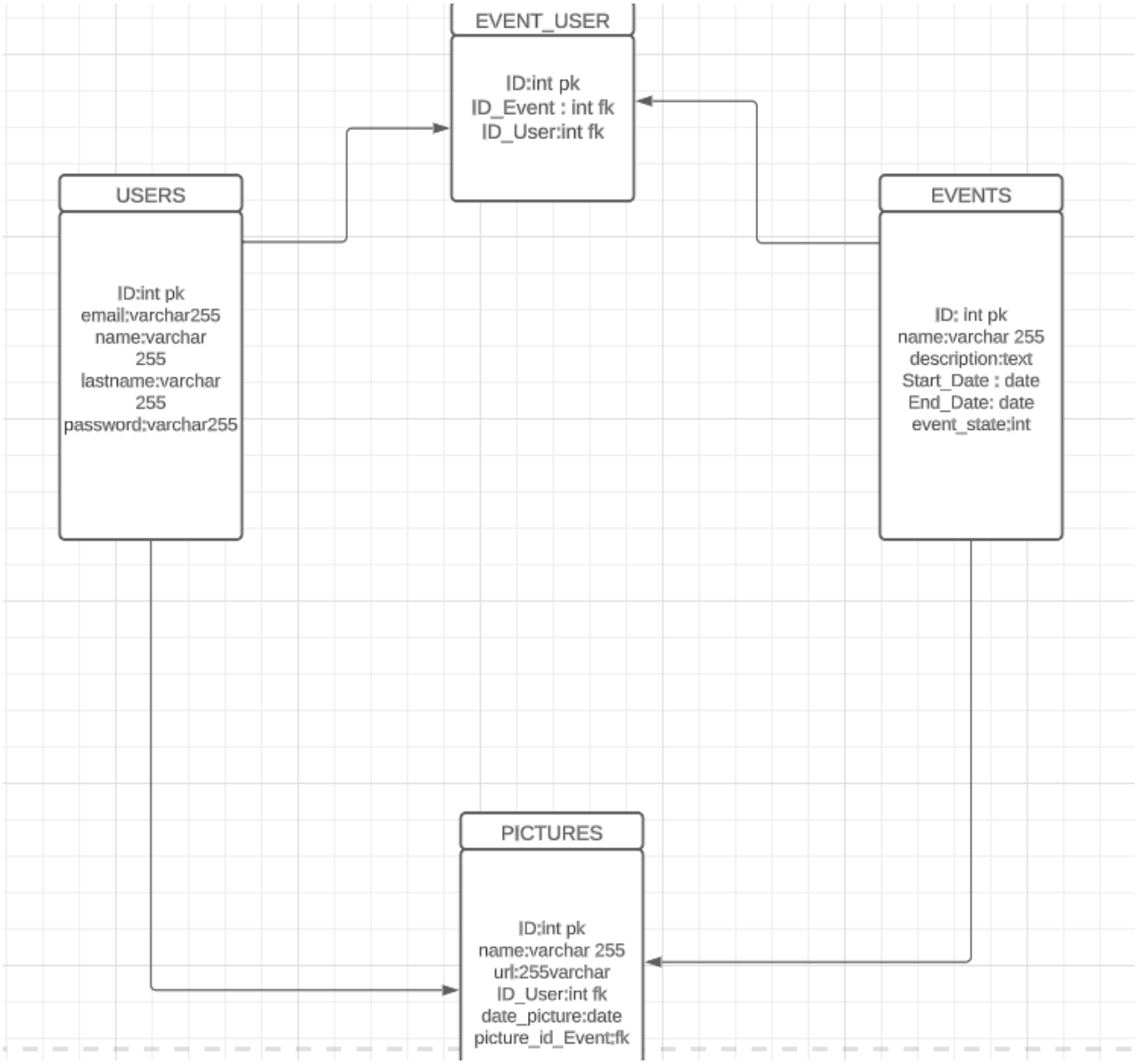
Modèle Conceptuel de Données (MCD):

Le Modèle Conceptuel de Données est la première étape de la conception d'une base de données. Il permet de représenter de manière abstraite et simplifiée la structure des données que l'on souhaite stocker. Dans un MCD, nous avons identifié les différentes entités de notre projet (par exemple, Utilisateur, Événement, etc.), ainsi que les relations entre ces entités.



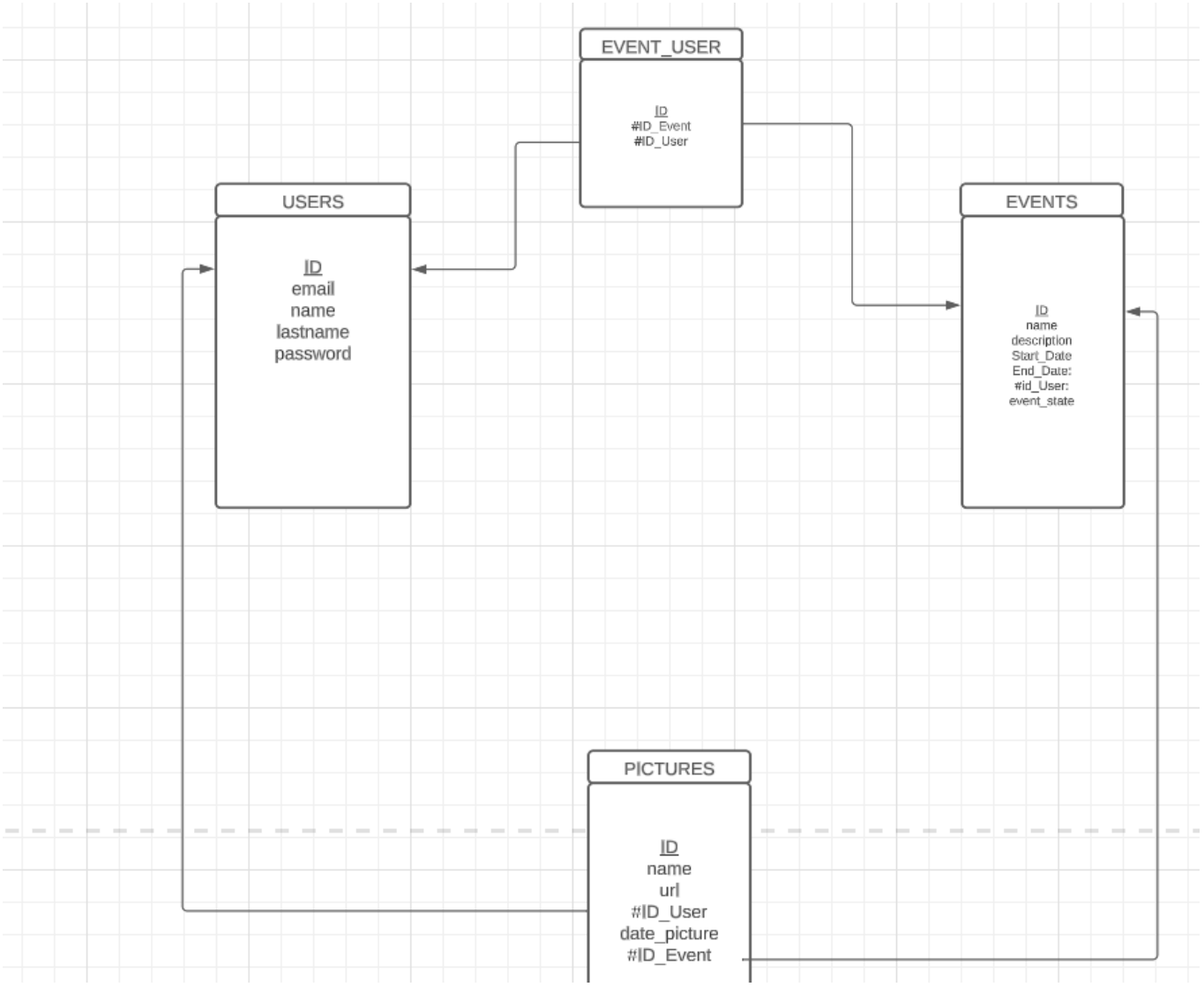
Modèle Logique de Données (MLD)

Une fois le **MCD** défini, nous sommes passés à l'étape du **Modèle Logique de Données**. **Le MLD** est une représentation plus technique du **MCD**, destinée à préparer l'implémentation physique de la base de données. Dans le **MLD**, nous avons traduit **les entités et les relations du MCD en tables**, et **les attributs en champ de ces tables**. Nous avons également défini **les clés primaires(PK) et étrangères(FK**) nécessaires pour assurer l'intégrité des données.



Modèle Physique de Données (MPD)

Enfin, nous avons élaboré le **Modèle Physique de Données**, qui est une représentation du **MLD** adaptée à un système de gestion de base de données spécifique, dans notre cas, MySQL. Le MPD a donc défini la structure exacte des tables de notre base de données, y compris les types de données à utiliser pour chaque champ, les contraintes à appliquer, les index à créer, etc.



En créant ces modèles, nous avons pu conceptualiser et structurer notre base de données de manière efficace et logique, assurant ainsi une gestion de la persistance des données dans notre application.

Ce projet m'a permis d'acquérir la compétence **Concevoir une base de données.**

2. Précisez les moyens utilisés :

Cette étape du projet a été réalisé à l'aide de **LucidChart**

3. Avec qui avez-vous travaillé ?

J'ai travaillé en groupe pour ce projet :
Raphael Diop
Yacine BOUCETTA
rémi Garguillo

4. Contexte

Nom de l'entreprise, organisme ou association ▸ La plateforme

Chantier, atelier, service ▸ *Cliquez ici pour taper du texte.*

Période d'exercice ▸ Du : *02/01/2023i* au : *31/08/2023a*

5. Informations complémentaires *(facultatif)*

2 **Concevoir et développer la persistance des données en intégrant les recommandations de sécurité**

Activité-type

Exemple n°1 - *Share event application mobile*

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

Suite à la création du MCD MLD MPD nous avons créer la base de donnée a l'aide de PHPmyadmin , le MPD et la representation finale de notre base de donnée donc nous appuyons sur celui- ci pour créer les tables et leurs attributs .
dans la capture ci-dessous nous pouvons voir la table user et ses attributs :

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
1	id	int			Non	*Aucun(e)*		AUTO_INCREMENT	Modifier Supprimer Plus
2	email	varchar(255)	latin1_swedish_ci		Non	*Aucun(e)*			Modifier Supprimer Plus
3	name	varchar(255)	latin1_swedish_ci		Non	*Aucun(e)*			Modifier Supprimer Plus
4	lastname	varchar(255)	latin1_swedish_ci		Non	*Aucun(e)*			Modifier Supprimer Plus
5	password	varchar(255)	latin1_swedish_ci		Non	*Aucun(e)*			Modifier Supprimer Plus
6	url	varchar(255)	latin1_swedish_ci		Non	*Aucun(e)*			Modifier Supprimer Plus
7	isAdmin	tinyint			Non	*Aucun(e)*			Modifier Supprimer Plus

puis la table Event :

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
1	id	int			Non	*Aucun(e)*		AUTO_INCREMENT	Modifier Supprimer Plus
2	name	varchar(255)	latin1_swedish_ci		Non	*Aucun(e)*			Modifier Supprimer Plus
3	start_date	datetime(6)			Non	CURRENT_TIMESTAMP(6)		DEFAULT_GENERATED	Modifier Supprimer Plus
4	end_date	datetime(6)			Non	CURRENT_TIMESTAMP(6)		DEFAULT_GENERATED	Modifier Supprimer Plus
5	description	varchar(255)	latin1_swedish_ci		Non	*Aucun(e)*			Modifier Supprimer Plus
6	url_event	varchar(255)	latin1_swedish_ci		Non	*Aucun(e)*			Modifier Supprimer Plus
7	isPrivate	tinyint			Non	*Aucun(e)*			Modifier Supprimer Plus
8	id_user	int			Non	*Aucun(e)*			Modifier Supprimer Plus

et comme vu lors de la présentation de la conception d'une bdd une table de relation entre la table user et event

table user_event :

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra	Action
1	eventId	int			Oui	*NULL*			Modifier Supprimer Plus
2	id	int			Non	*Aucun(e)*		AUTO_INCREMENT	Modifier Supprimer Plus
3	isInvitated	tinyint			Non	*Aucun(e)*			Modifier Supprimer Plus
4	userId	int			Oui	*NULL*			Modifier Supprimer Plus

Ce projet m'a permis d'acquérir la compétence **Mettre en place une base de données**

2. Précisez les moyens utilisés :

Pour la réalisation de ce projet nous avons utilisé **Phpmyadmin**

PhpMyAdmin fournit une interface web conviviale qui facilite la gestion des bases de données MySQL, même pour les utilisateurs non techniques. Il offre une navigation intuitive, des fonctionnalités d'édition de données, de création de tables, de requêtes SQL et d'administration des privilèges.

Gestion complète des bases de données : PhpMyAdmin permet de gérer tous les aspects d'une base de données, y compris la création, la modification et la suppression des bases de données, des tables, des colonnes, des index, des clés étrangères, etc. Il offre également des fonctionnalités avancées telles que l'importation et l'exportation de données, la sauvegarde et la restauration de bases de données.

3. Avec qui avez-vous travaillé ?

j'ai travaillé en groupe pour ce projet :
Raphael Diop
Yacine BOUCETTA
Rémi Garguillo

4. Contexte

Nom de l'entreprise, organisme ou association ▸ La plateforme

Chantier, atelier, service ▸ *Cliquez ici pour taper du texte.*

Période d'exercice ▸ Du : *02/01/2023* au : *31/08/2023*

5. Informations complémentaires *(facultatif)*

Activité-type 2

Exemple n°3 - **Concevoir et développer la persistance des données en intégrant les recommandations de sécurité**

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

Dans le cadre de notre travail sur une API de gestion d'événements, j'ai pu démontrer ma compétence à développer des composants métier en utilisant **NestJS** et **TypeORM**.

Injection de dépendances

Nous avons utilisé l'injection de dépendances pour intégrer les référentiels nécessaires à notre service. Par exemple, dans EventsService, nous avons injecté les référentiels pour Event, User, et UserEvent .

```
constructor(
  @InjectRepository(Event)
  private eventsRepository: Repository<Event>,
  @InjectRepository(User)
  private usersRepository: Repository<User>,
  @InjectRepository(UserEvent)
  private userEventRepository: Repository<UserEvent>
) {}
```

Gestion des opérations CRUD

Notre API gère un ensemble de scénarios de base de données. Par exemple, elle peut créer de nouveaux événements et utilisateurs (create), mettre à jour les informations des événements (updateEvent), récupérer la liste des événements (findAll), rechercher un événement spécifique (findOne) et supprimer des événements (remove).

```
async create(event: CreateEventDto) {

  const reqsUser=await this.usersRepository.findOneBy({id:event.id_user});
  const newUser =new User();
  newUser.id=reqsUser.id;
  this.usersRepository.save(newUser);

  const newEvent = new Event();
    newEvent.name = event.name,
    newEvent.url_event = event.url_event,
    newEvent.id_user = event.id_user,
    newEvent.start_date = new Date(),
    newEvent.end_date = new Date(),
    newEvent.isPrivate = event.isPrivate,
    newEvent.description = event.description,
  await this.eventsRepository.manager.save(newEvent);
}
```

Gestion des relations entre les tables

Notre API gère les relations entre les utilisateurs et les événements. Par exemple, elle peut ajouter des utilisateurs à un événement (updateEventUsers), supprimer des utilisateurs d'un événement (deleteEventUsers), et rechercher tous les utilisateurs participant à un événement (findAllUserInEvent).

```
async updateEventUsers(eventId: string, User:GetUserDto): Promise<any> {
  const user = await this.usersRepository.findOneBy({email:User.email});
  if (user) {
    const event = await this.eventsRepository.findOneBy({ id:+eventId});
    if (event) {
      const userEvent = new UserEvent();
  userEvent.user = user;
  userEvent.event = event;
  userEvent.isInvitated=false;
```

Utilisation de requêtes complexes

Nous utilisons des requêtes SQL complexes pour récupérer des informations spécifiques. Par exemple, pour obtenir la liste de tous les utilisateurs participant à un événement, j'utilise une jointure entre les tables user_event, event, et user :

```
return await this.usersRepository
  .createQueryBuilder()
  .select("user.name,user.email,user.url,user.id")
  .innerJoin("user_event","user_event","user_event.userId = user.id")
  .innerJoin("event","event","user_event.eventId = event.id")
  .where("user_event.eventId = "+ id_event)
  .execute()
```

En conclusion, le travail que j'ai accompli sur cette API démontre **ma capacité à développer des composants métier efficaces et robustes**.

2. Précisez les moyens utilisés :

Pour ce projet nous avons utilisés:

NestJS
TypeOrm
Visual studio Code
ainsi que la documentation officiel de nestJS

3. Avec qui avez-vous travaillé ?

J'ai travaillé en équipe avec

Raphael Diop
Damien Verschaere

4. Contexte

Nom de l'entreprise, organisme ou association

Laplateforme

Chantier, atelier, service

Période d'exercice Du : *02/01/2023* au : *31/08/2023*

5. Informations complémentaires *(facultatif)*

DOSSIER PROFESSIONNEL (DP)

Activité-type 3

Concevoir et développer une application multicouche répartie en intégrant les recommandations de sécurité

Exemple n°1 - *Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement*

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

Au cours de notre projet, nous avons adopté une approche collaborative pour gérer et organiser efficacement nos tâches et notre environnement de développement. Voici une liste de quelques outils que nous avons utilisés et comment ils ont contribué à notre processus de développement :

Google Docs : Nous avons utilisé Google Docs pour centraliser toutes les informations essentielles et les comptes-rendus de nos réunions. Cela nous a aidé à garder une trace des décisions importantes, des tâches attribuées et des prochaines étapes du projet. Cela nous a également permis de collaborer en temps réel sur les documents et de maintenir une source unique de vérité pour notre projet.

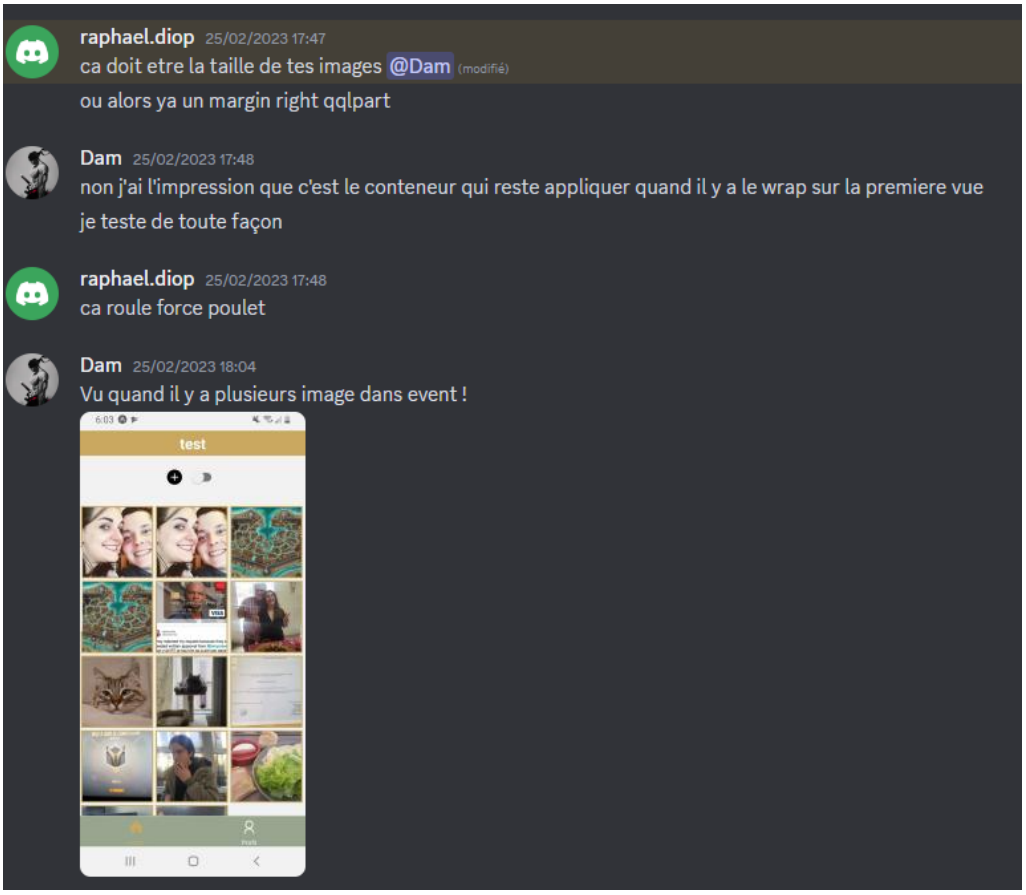
REUNION 09/01/2023
(Scrum master: Raphaël)

- Fix des pb de formats de date
- Poursuite de L'API :
  - Finition de routes (admin essentiellement)
  - Création de la vérification par JWT
  - Création des entités de liaison ?
- Finir de design le chemin utilisateur sur l'application

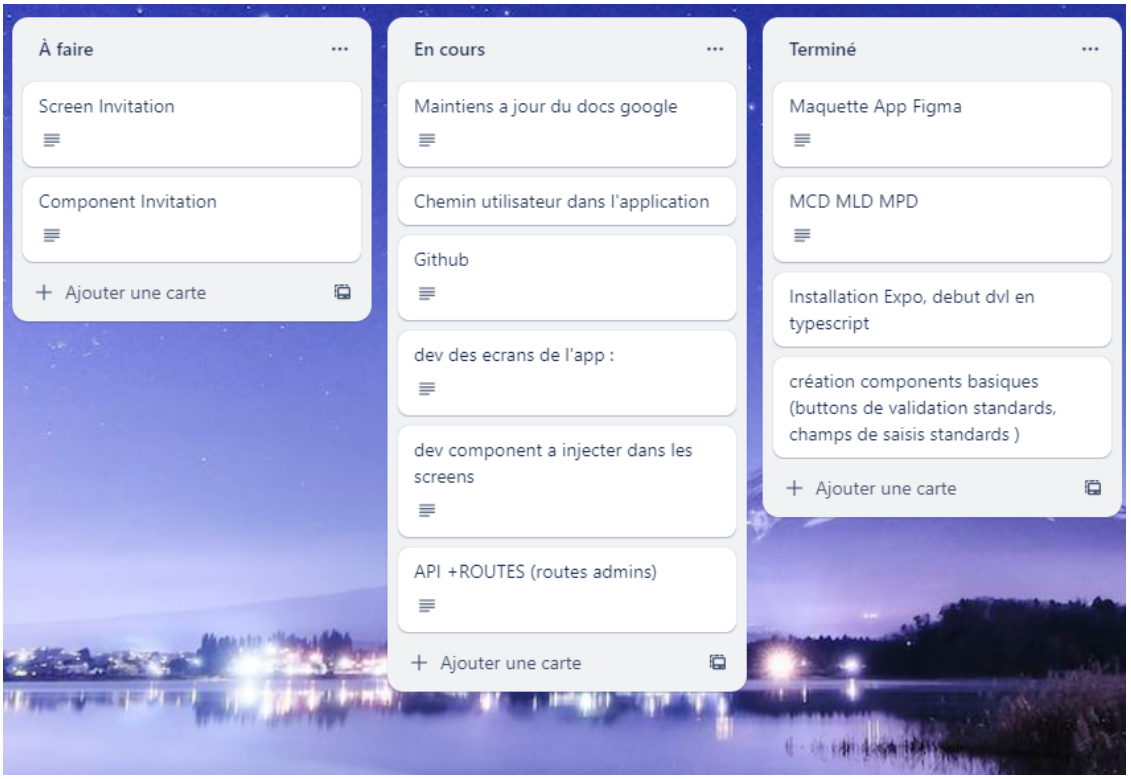
RÉUNION 10/01/2023
(Scrum master: Raphaël)

- Modification API, utilisation DTO.
- Debut dev application:
  - implémentation de la navigation dans l'appli via la librairie react-navigation (à installer depuis expo). Getting started | React Navigation
  - Dvl Écran SInIn.

Discord : Discord a été notre plateforme de choix pour les réunions à distance. En plus des discussions vocales et vidéo, nous avons utilisé les canaux de texte pour des mises à jour rapides, des clarifications et des discussions sur les problèmes ou tâches en cours. Discord a aidé à maintenir une communication fluide entre les membres de l'équipe malgré le travail à distance.



Trello : Nous avons utilisé Trello pour la gestion des tâches. Nous avons organisé nos tâches en listes et en cartes, en utilisant le système de statut "À faire", "En cours" et "Terminé" pour suivre l'avancement de chaque tâche. Trello nous a aidé à visualiser l'état d'avancement du projet et à identifier rapidement les goulets d'étranglement ou les problèmes potentiels.



GitHub : Pour le contrôle de version, nous avons utilisé GitHub. Cela nous a permis de travailler simultanément sur différents aspects du projet sans interférer les uns avec les autres.
En utilisant ces outils, nous avons pu travailler efficacement en tant qu'équipe, malgré les défis de la collaboration à distance. Nous avons appris à nous adapter à différents outils et processus, et à en tirer le meilleur parti pour notre projet.
ce projet m'a permis d'acquérir **Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement**

2. Précisez les moyens utilisés :

Pour ce projet nous avons utilisé different outils :
Google doc
Trello
Discord
Github

3. Avec qui avez-vous travaillé ?

J'ai travaillé en équipe avec

Raphael Diop
Yacine BOUCETTA
Rémi Garguilo

4. Contexte

Nom de l'entreprise, organisme ou association

Laplateforme

Chantier, atelier, service

Période d'exercice Du : *02/01/2023* au : *31/08/2023*

5. Informations complémentaires *(facultatif)*

Activité-type 3

Concevoir et développer une application multicouche répartie en intégrant les recommandations de sécurité

Exemple n°2 - *Développer des composants métier*

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

Dans notre application, nous avons développé un composant métier appelé UsePhotoUploadImage. Ce composant est responsable de la gestion de la sélection et de l'envoi d'images vers un serveur Firebase. Je vais vous présenter l'extrait de code du composant métier UsePhotoUploadImage et expliquer comment il est utilisé.

Le composant métier UsePhotoUploadImage est développé en utilisant des bibliothèques telles que Expo ImagePicker, SecureStore, **Axios** et Firebase. Il permet de sélectionner une image à partir de la galerie de l'appareil et de l'envoyer vers un serveur Firebase.
ci-dessous utilisation de **Image Picker** de expo

```
const pickImage = async () => {
  // Launch the image picker
  const { status } = await ImagePicker.requestMediaLibraryPermissionsAsync();
  if (status === "granted") {
    const result = await ImagePicker.launchImageLibraryAsync({
      mediaTypes: ImagePicker.MediaTypeOptions.Images,
      allowsEditing: true,
      aspect: [4, 3],
      quality: 1,
    });

    if (!result.canceled) {
      setImageUri(result.assets[0].uri);
      console.log(imageUri);
    }
  }
};
```

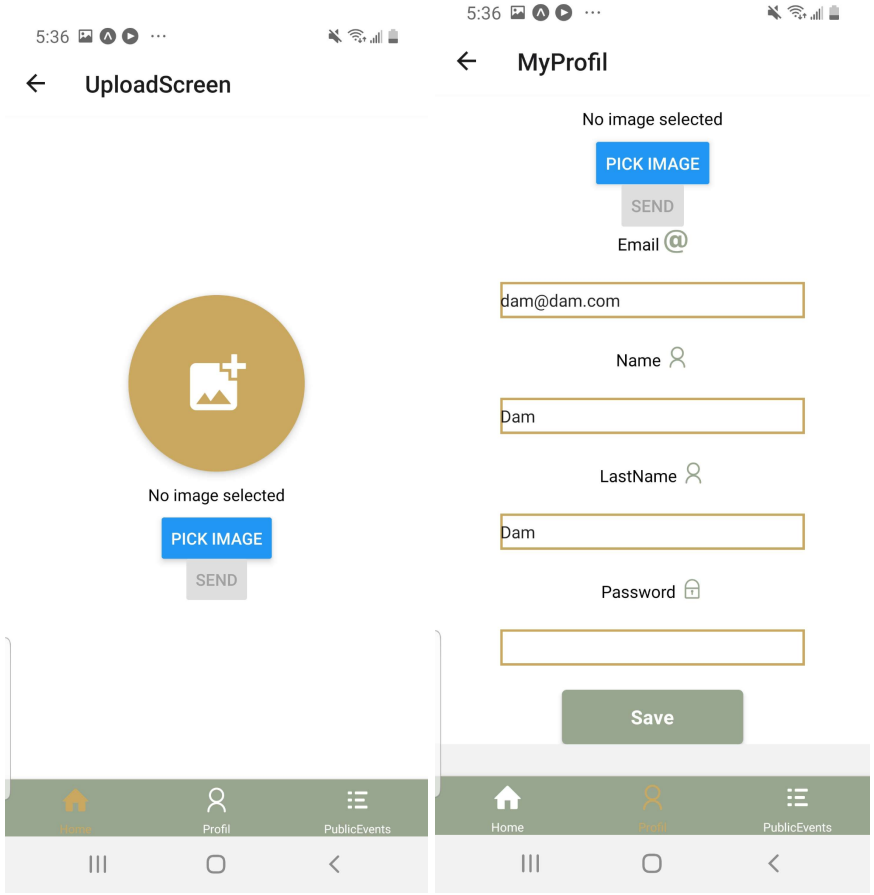
Le composant est utilisé a dans plusieur screen de l'application telle que le profil pour uploader une photo de profil pour l'utilisateur

```
<ScrollView>
  <View style={style.gen}>
    <Image source={{ uri: url }} style={style.pictureprofile} />
    <UsePhotoUploadImage eventId={null} onImageUploaded={setUpdatedImageUrl} uploadPurpose="profile" />

    <Text style={style.font}>
      Email{' '}
      <Entypo
        name="email"
        size={20}
        color="#98A68F"
```

Ainsi que sur le screen upload Image qui permet de poster une image sur un event

```
export default function UploadScreen({ navigation, route }: UploadScreenProps) {
  const [showCameraComponent, setShowCameraComponent] = useState(false);
  const [showLibraryComponent, setShowLibraryComponent] = useState(false);
  // const {titleLibrary = '+' } = route.params;
  const { eventId } = route.params;

  return (
    <View style={styles.container}>
      <Pressable style={styles.roundButton} onPress={() => setShowLibraryComponent(true)}>
        <MaterialIcons name="add-photo-alternate" size={60} color="white" />
      </Pressable>
      {showLibraryComponent && <UsePhotoUploadImage uri={""} eventId={eventId} />}
    </View>
  );
}
```

Ci dessous les screen de l'application utilisant le composant image upload



Le composant métier UsePhotoUploadImage est utilisé pour gérer la sélection et l'envoi d'images dans notre application. En l'utilisant dans un écran spécifique,nous permettons aux utilisateurs d'ajouter des images à leurs événements ou à leur photo de profil. Ce composant métier démontre notre capacité à développer des fonctionnalités spécifiques et réutilisables pour améliorer l'expérience utilisateur de notre application.
ce projet m'a permis d'acquérir la compétence ***Développer des composants métier***

2. Précisez les moyens utilisés :

React Native
Image picker expo
firebase

3. Avec qui avez-vous travaillé ?

J'ai travaillé en équipe avec

Raphael Diop
Yacine BOUCETTA
Rémi Garguilo

4. Contexte

Nom de l'entreprise, organisme ou association

Laplateforme

Chantier, atelier, service

Période d'exercice Du : *02/01/2023* au : *31/08/2023*

5. Informations complémentaires *(facultatif)*

Activité-type 3

Concevoir et développer une application multicouche répartie en intégrant les recommandations de sécurité

Exemple n°2 - *Share event*

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

Pour le développement de l'application mobile Share event ,il a été essentiel de structurer le code de manière logique et cohérente afin d'assurer la maintenabilité, l'évolutivité et la robustesse de l'application .
Pour cela nous avons utilisé un modèle de conception couramment utilisé pour atteindre ces objectifs est **l'architecture en couches (ou architecture n-tiers)**. Dans ce modèle, une application est divisée en plusieurs couches qui ont chacune des responsabilités distinctes.
Cette **séparation des responsabilités** permet une plus grande **modularité du code**, facilitant ainsi la **maintenance et l'évolution de l'application**. Elle permet également une meilleure réutilisabilité du code et peut améliorer la performance en distribuant les tâches entre différentes couches.

La couche de présentation de notre application est entièrement construite à l'aide de **React Native** et **Expo**, deux technologies modernes permettant de développer des applications mobiles multiplateformes (iOS et Android).
Dans cette section, nous allons examiner deux composants clés de notre application : le EventScreen et le EventImage.
1. Le composant EventScreen :
EventScreen est une page qui présente des informations sur un événement spécifique. Il reçoit l'identifiant (id) et le nom (nameevent) de l'événement à travers les paramètres de navigation (route).
Voici un extrait de code de ce composant :

```
const EventScreen = ({ route }: EventScreenProps) => {
  const { id, nameevent } = route.params;

  return (
    <View style={styles.view}>
      <View style={styles.header}>
        <Text style={styles.headerText}>{nameevent}</Text>
      </View>
      <EventImage eventId={id} />
    </View>
  );
};
```

Dans ce composant, nous utilisons le paramètre id pour rendre le composant EventImage qui affiche toutes les images liées à l'événement en question. Le nom de l'événement est affiché dans le header.
2. Le composant EventImage :

EventImage est un composant responsable de la récupération et de l'affichage des images associées à un événement spécifique. Il reçoit l'identifiant de l'événement et effectue une requête à l'API pour obtenir les images correspondantes. De plus, il propose une option d'affichage différente des images grâce à un bouton de switch, et un bouton pour naviguer vers une autre page d'upload d'images. Voici un extrait de code de ce composant :

```
const EventImage = ({ eventId }: EventImageProps) => {
  const [events, setEvents] = useState<any[]>([]);
  const [isEnabled, setIsEnabled] = useState(false);
  const toggleSwitch = () => setIsEnabled(previousState => !previousState);
  const navigation = useNavigation();

  useEffect(() => {
    async function getEvents() {
      try {
        const response = await axios.get(
          route+`/pictures/find/` + eventId
        );
        setEvents(response.data);
      } catch (error) {
        console.error(error);
      }
    }
    getEvents();
  }, [eventId]);
```

Ces deux composants montrent comment **la couche de présentation** est conçue pour **interagir avec l'utilisateur** et afficher les données pertinentes fournies par la couche de logique métier. Leur seul rôle est de présenter les données de manière lisible et facile à utiliser pour l'utilisateur. Ils n'ont pas à se soucier de savoir comment les données sont récupérées ou où elles sont stockées, ce qui illustre la séparation des responsabilités entre les différentes couches de notre application.

La couche logique métiers :
Elle abrite la logique opérationnelle de l'application, gère les interactions entre **la couche de présentation** et **la couche d'accès aux données**, et met en œuvre les règles métier. Dans le cadre de notre application, **cette couche est gérée par NestJS**.
Contrôleur des événements :
La classe **EventsController** est le point d'entrée de notre application pour toutes **les requêtes HTTP** liées à la gestion des événements. Elle définit les routes pour créer de nouveaux événements (register), mettre à jour des événements existants (updateUser), supprimer des événements (remove), et bien plus encore. Cette classe s'appuie sur EventsService pour traiter les requêtes reçues.

```
@Controller('events')
export class EventsController {
  constructor(private readonly eventsService: EventsService,
```

Service des événements :
La classe EventsService gère la logique métier spécifique aux événements. Elle interagit avec la base de données via le repository eventsRepository pour effectuer des opérations telles que la création d'un nouvel événement (create), la mise à jour d'un événement existant (updateEvent), ou la suppression d'un événement (remove). Cette classe est essentiellement une couche d'abstraction qui simplifie l'interaction avec la base de données pour le contrôleur.

```
@Injectable()
export class EventsService {
  constructor(
    @InjectRepository(Event)
    private eventsRepository: Repository<Event>,
    @InjectRepository(User)
    private usersRepository: Repository<User>,
    @InjectRepository(UserEvent)
    private userEventRepository: Repository<UserEvent>
  ) {}
```

Entité de l'événement :

L'entité Event représente un événement dans notre application. Elle définit la structure des événements dans notre base de données et fournit une interface pour interagir avec les données d'événements dans notre code. L'entité Event est utilisée par EventsService pour créer, mettre à jour et supprimer des événements.

```
@Entity()
export class Event extends BaseEntity {
  @PrimaryGeneratedColumn()
  id: number;
```

En structurant notre application de cette manière, nous parvenons à une séparation claire des responsabilités entre le contrôleur, le service et l'entité. Cela rend notre code plus lisible, plus maintenable et plus facile à tester. De plus, en définissant la logique métier dans le service plutôt que dans le contrôleur, nous pouvons facilement réutiliser cette logique dans d'autres parties de notre application si nécessaire.

La couche d'accès au données :
La couche d'accès aux données est une composante essentielle,permettant d'interagir avec le système de stockage de données, que ce soit une base de données SQL, NoSQL, un système de fichiers, un service cloud, etc.
Dans le cas de notre application, nous utilisons une base de données SQL et interagissons avec elle via un ORM (Object-Relational Mapping), en particulier TypeORM. TypeORM est un ORM très populaire pour TypeScript et Node.js, offrant une approche basée sur les entités et les décorateurs pour définir les

modèles de données et les relations entre eux.
Voici comment l'entité Event est définie :

```
  @OneToMany(() => User, (user) => user.events)
  @JoinTable({
    name: 'user_event',
    joinColumn: {
      name: 'user',
      referencedColumnName: 'id',
    },
    inverseJoinColumn: {
      name: 'event',
      referencedColumnName: 'id',
    },
  })
  users:User[];

  @OneToMany(() => Picture, (picture) => picture.event)
  picture: Picture[]
}
```

Chaque entité est mappée à une table dans la base de données. Ici, Event est une table qui contient des événements. Les décorateurs comme @Entity, @PrimaryGeneratedColumn, @Column, @OneToMany, etc., sont utilisés pour décrire la structure de la table et les relations entre les tables.

En plus des entités, nous utilisons également des **Data Transfer Objects** (DTOs) pour modéliser les données transférées entre le client et le serveur. Les DTO sont essentiels pour assurer la sécurité, la validation et la cohérence des données.

Par exemple, lorsque le client crée un nouvel événement, nous n'utilisons pas directement l'objet reçu dans la requête HTTP pour créer un nouvel enregistrement dans la base de données. Au lieu de cela, nous utilisons un DTO pour définir la forme que ces données devraient avoir. Le DTO correspondant est CreateEventDto:

```
export class CreateEventDto {
  name: string;
  url_event: string;
  id_user: number;
  isPrivate: boolean;
  description: string;
}
```

Cela garantit que seul un ensemble spécifique de champs peut être utilisé pour créer un nouvel événement, évitant ainsi l'injection de champs indésirables ou potentiellement malveillants.

En conclusion, la couche d'accès aux données en utilisant un **ORM** comme **TypeORM** et des **DTO**s permet de maintenir un code propre, sécurisé et maintenable. Elle offre également une manière abstraite de travailler avec la base de données, facilitant la transition vers un autre système de base de données si nécessaire.

ce projet m'a permis d'acquérir la compétence **Construire une application organisée en couches**

2. Précisez les moyens utilisés :

React Native et Expo pour **La couche de présentation**
NestJs pour **La couche logique métier**
BDD SQL type ORM et DTO **La couche d'accès au données**

3. Avec qui avez-vous travaillé ?

J'ai travaillé en équipe avec

Raphael Diop
Yacine BOUCETTA
Rémi Garguilo

4. Contexte

Nom de l'entreprise, organisme ou association ▸

Laplateforme

Chantier, atelier, service ▸

Période d'exercice ▸ Du : *02/01/2023* au : *31/08/2023*

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] Cliquez ici pour taper du texte. ,

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 Cliquez ici pour taper du texte. le Cliquez ici pour choisir une date

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)