

PROJET PROFESSIONNEL 2023



Share Event



TABLE DES MATIÈRES

01

INTRODUCTION

02

L'ORGANISTION ET CONCEPTION DU PROJET

03

PARTIE FRONT END

04

PARTIE BACK END

05

COMMUNICATION FRONT/BACK

06

CYBERSECURITÉ

07

TEST UNITAIRE

08

JEUX D'ESSAI

09

AXES D'AMELIORATIONS

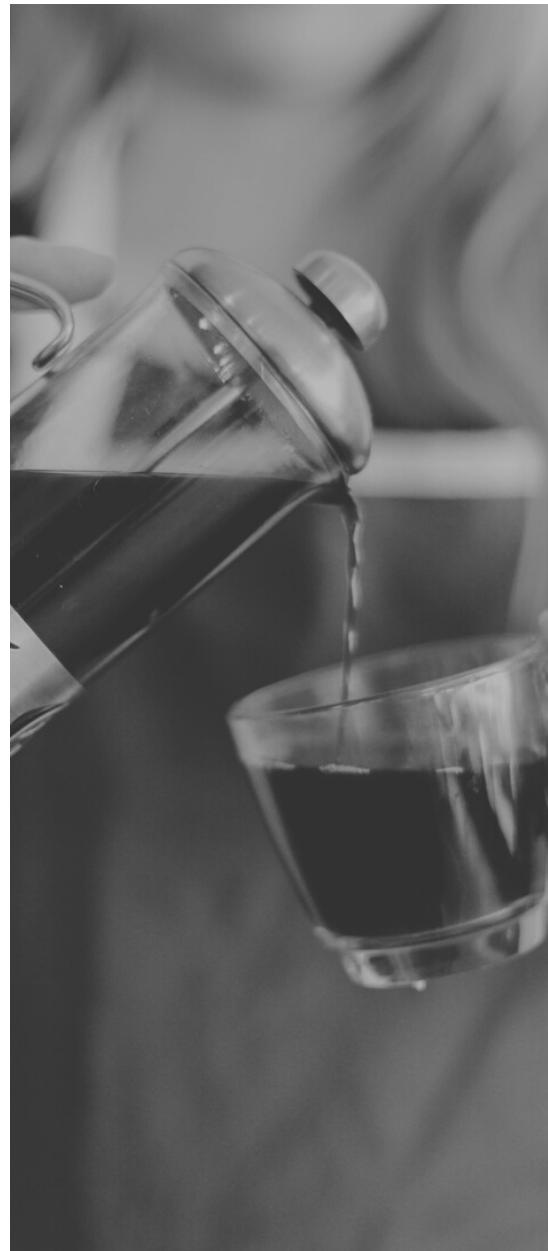
INTRODUCTION

Contexte et motivation

DANS UN MONDE DE PLUS EN PLUS CONNECTÉ ET OÙ LES ÉVÉNEMENTS SOCIAUX JOUENT UN RÔLE CENTRAL DANS NOS VIES, IL EST DEVENU ESSENTIEL DE DISPOSER D'UNE PLATEFORME PRATIQUE POUR CRÉER ET PARTAGER DES ÉVÉNEMENTS.

DE PLUS, LORS DE CES ÉVÉNEMENTS, LES PARTICIPANTS PRENNENT SOUVENT DE NOMBREUSES PHOTOS QU'ils SOUHAITENT PARTAGER AVEC LES AUTRES PARTICIPANTS. CEPENDANT, IL PEUT ÊTRE DIFFICILE DE CENTRALISER TOUTES CES PHOTOS EN UN SEUL ENDROIT.

C'EST DANS CE CONTEXTE QUE L'APPLICATION SHAREEVENT A ÉTÉ DÉVELOPPÉE.



OBJECTIFS DE L'APPLICATION SHAREEVENT :

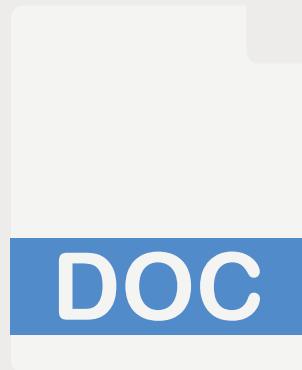
L'OBJECTIF PRINCIPAL DE SHAREEVENT EST DE SIMPLIFIER LA CRÉATION ET LE PARTAGE D'ÉVÉNEMENTS EN FOURNISANT UNE PLATEFORME CONVIVIALE ET INTUITIVE. EN UTILISANT SHAREEVENT, LES UTILISATEURS PEUVENT CRÉER DES ÉVÉNEMENTS, INVITER D'AUTRES PARTICIPANTS ET PARTAGER FACILEMENT LES PHOTOS PRISES LORS DE CES ÉVÉNEMENTS. L'APPLICATION VISE À OFFRIR UNE EXPÉRIENCE UTILISATEUR FLUIDE ET À RÉSOUDRE LE PROBLÈME DE DISPERSION DES PHOTOS SUR DIFFÉRENTES PLATEFORMES DE MÉDIAS SOCIAUX.

02 L'ORGANISTION ET CONCEPTION DU PROJET

Le projet fut réalisé avec la méthode agiles:

Le développement agile, appelé aussi développement adaptatif, centré sur l'autonomie des ressources humaines impliquées, la production et la validation d'une application intégrée et testée en continu.

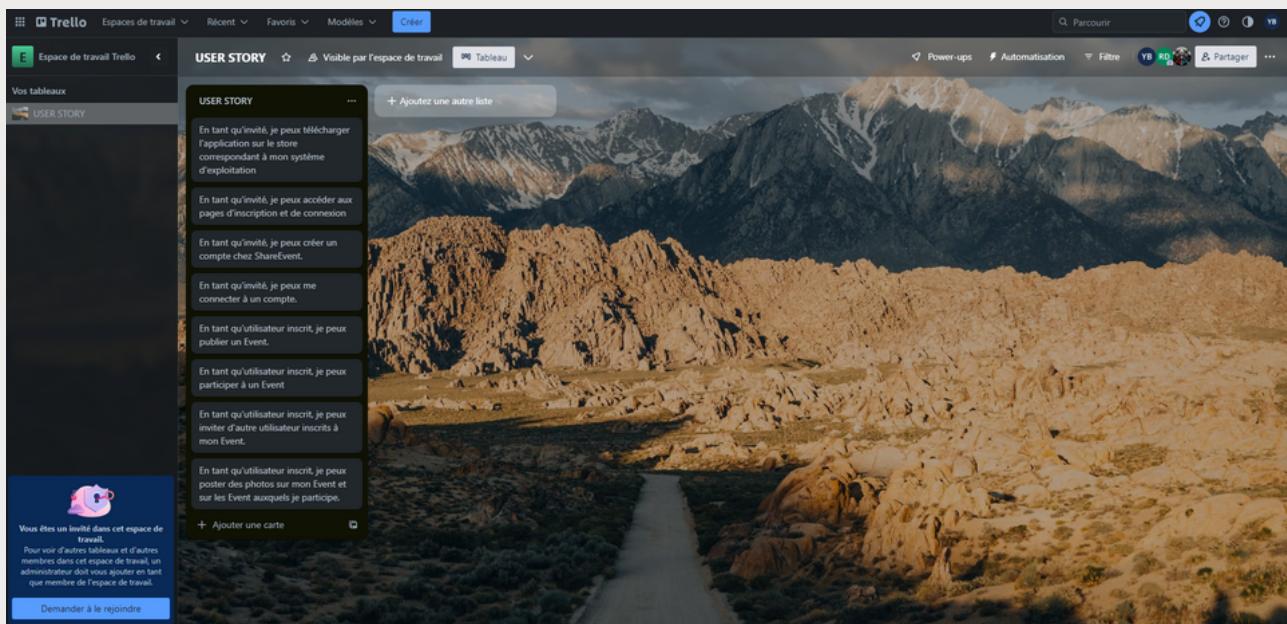
C'est à partir de ces réalités pratiques, que l'agilité progresse vers les sphères les plus hautes de l'organisation .



Pour l'organisation nous avons utilisés divers outils de communication tel que discord pour le travail à distance, trello pour l'organisation des ticket et taches à faire ainsi que google doc pour les résumé des réunions bi-mensuel,auquels on changé régulièrement de scrum master

02 L'ORGANISTION ET CONCEPTION DU PROJET

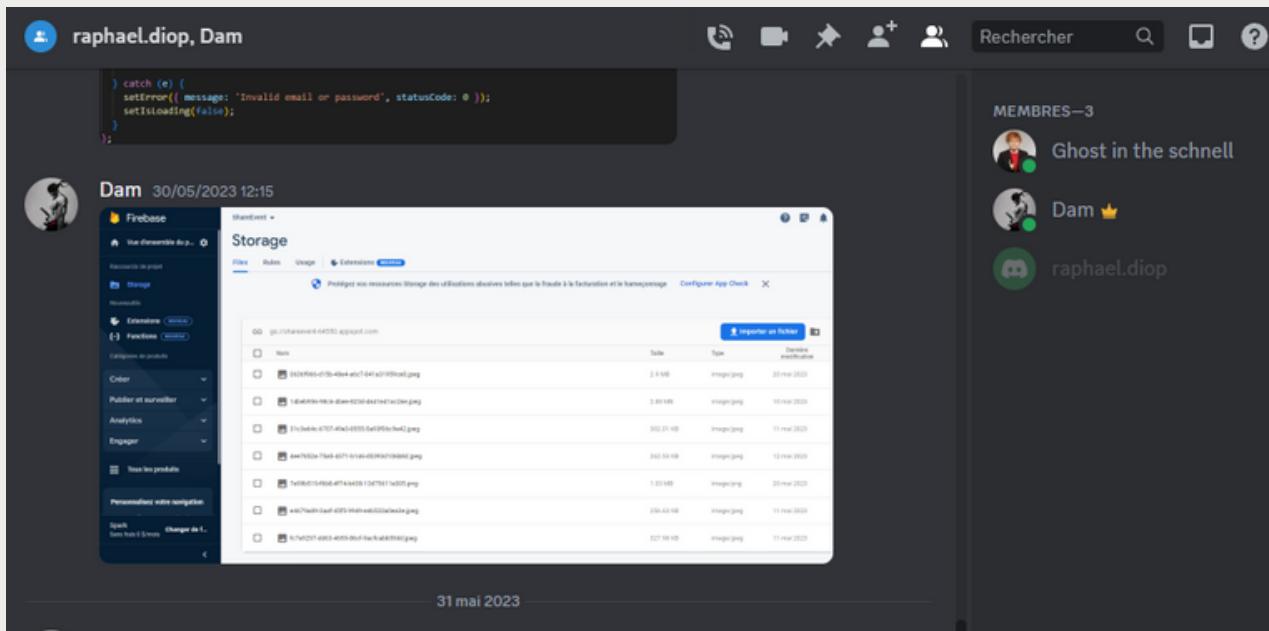
TRELLO



Trello est un outil de gestion de projet en ligne, lancé en septembre 2011 et inspiré par la méthode Kanban de Toyota. Il repose sur une organisation des projets en planches listant des cartes, chacune représentant des tâches. Les cartes sont assignable à des utilisateurs et sont mobiles d'une planche à l'autre, traduisant leur avancement.

02 L'ORGANISTION ET CONCEPTION DU PROJET

DISCORD



Discord est une plateforme de communication en ligne, créée à l'origine pour les joueurs, mais qui est maintenant utilisée par une grande variété de communautés en ligne. C'est un service gratuit qui permet aux utilisateurs de discuter, de partager des informations et de se connecter avec d'autres personnes partageant les mêmes centres d'intérêt.

Serveurs : Dans Discord, les utilisateurs peuvent créer ou rejoindre des "serveurs". Un serveur est essentiellement un groupe ou une communauté dédiée à un sujet spécifique. Par exemple, il pourrait y avoir un serveur pour un jeu en particulier, un groupe de discussion pour une série télévisée, une communauté d'artistes, etc.

Salons : À l'intérieur de chaque serveur, il y a des "salons" qui sont des canaux de discussion spécifiques. Ces salons peuvent être organisés par thématique (ex. : général, actualités, questions, etc.) pour faciliter les conversations ciblées.

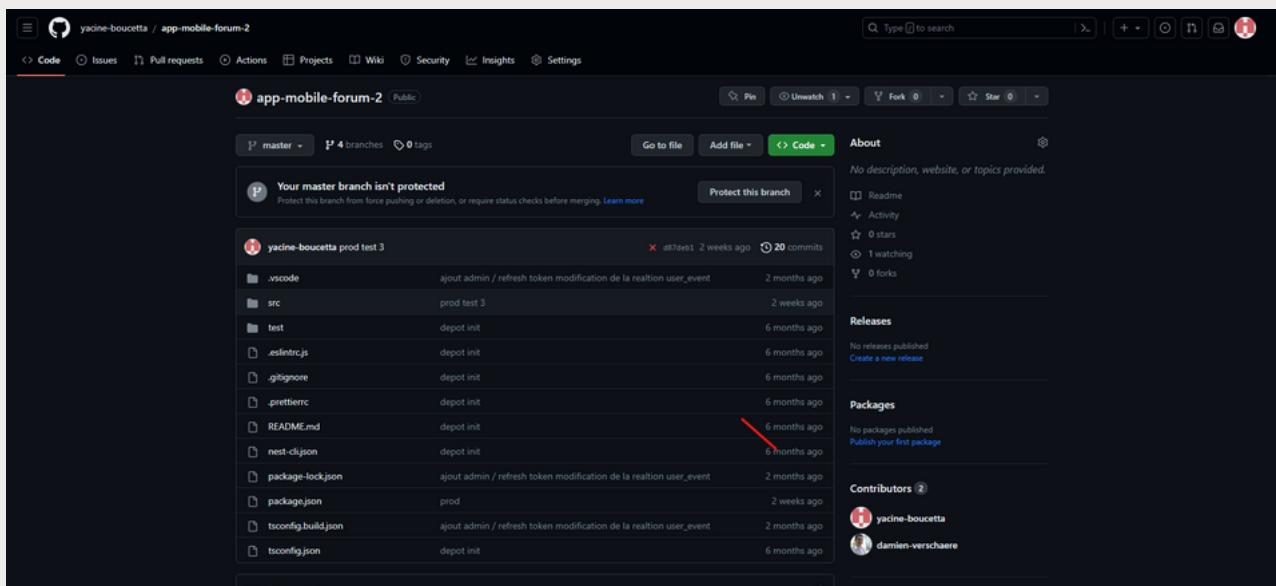
Chat vocal : Discord propose également des fonctionnalités de chat vocal. Les utilisateurs peuvent rejoindre des canaux vocaux pour discuter en temps réel avec d'autres membres du serveur. Cela est particulièrement utile pour les joueurs qui veulent communiquer pendant qu'ils jouent ensemble.

Messages textuels : En plus des discussions vocales, Discord permet également aux utilisateurs de communiquer par messages textuels dans les différents salons.

Rôles et permissions : Dans les serveurs Discord, les administrateurs peuvent attribuer des "rôles" aux membres, ce qui détermine leurs permissions au sein du serveur. Cela permet de gérer les droits d'accès et de modération.

02 L'ORGANISTION ET CONCEPTION DU PROJET

GIT HUB



GitHub est une plateforme en ligne qui permet aux développeurs de collaborer sur des projets informatiques et de gérer leur code source de manière efficace.

Repositories (Dépôts) : Sur GitHub, chaque projet est appelé un "repository" (ou "repo" en abrégé). C'est un espace où vous pouvez stocker tous les fichiers et dossiers de votre projet, y compris le code source, la documentation, les images, etc.

Commits : Lorsque vous apportez des modifications à votre code, vous effectuez ce qu'on appelle un "commit". Chaque commit est accompagné d'un message décrivant les changements que vous avez effectués.

Branches : Vous pouvez créer une "branche" pour travailler sur ces modifications en toute sécurité. Une fois que vos changements sont prêts, vous pouvez les fusionner avec la branche principale, appelée généralement "master".

Collaboration : GitHub facilite la collaboration entre plusieurs développeurs. D'autres personnes peuvent cloner (copier) votre repository sur leur propre ordinateur, apporter des modifications et vous soumettre ce qu'on appelle une "pull request". Cela vous permet de voir les modifications proposées et de les intégrer à votre projet si vous les trouvez utiles.

Suivi des problèmes : GitHub propose également un outil pour gérer les problèmes, les bugs et les demandes de fonctionnalités de votre projet. Cela vous aide à garder une trace des problèmes rencontrés et des améliorations à apporter.

02 L'ORGANISTION ET CONCEPTION DU PROJET

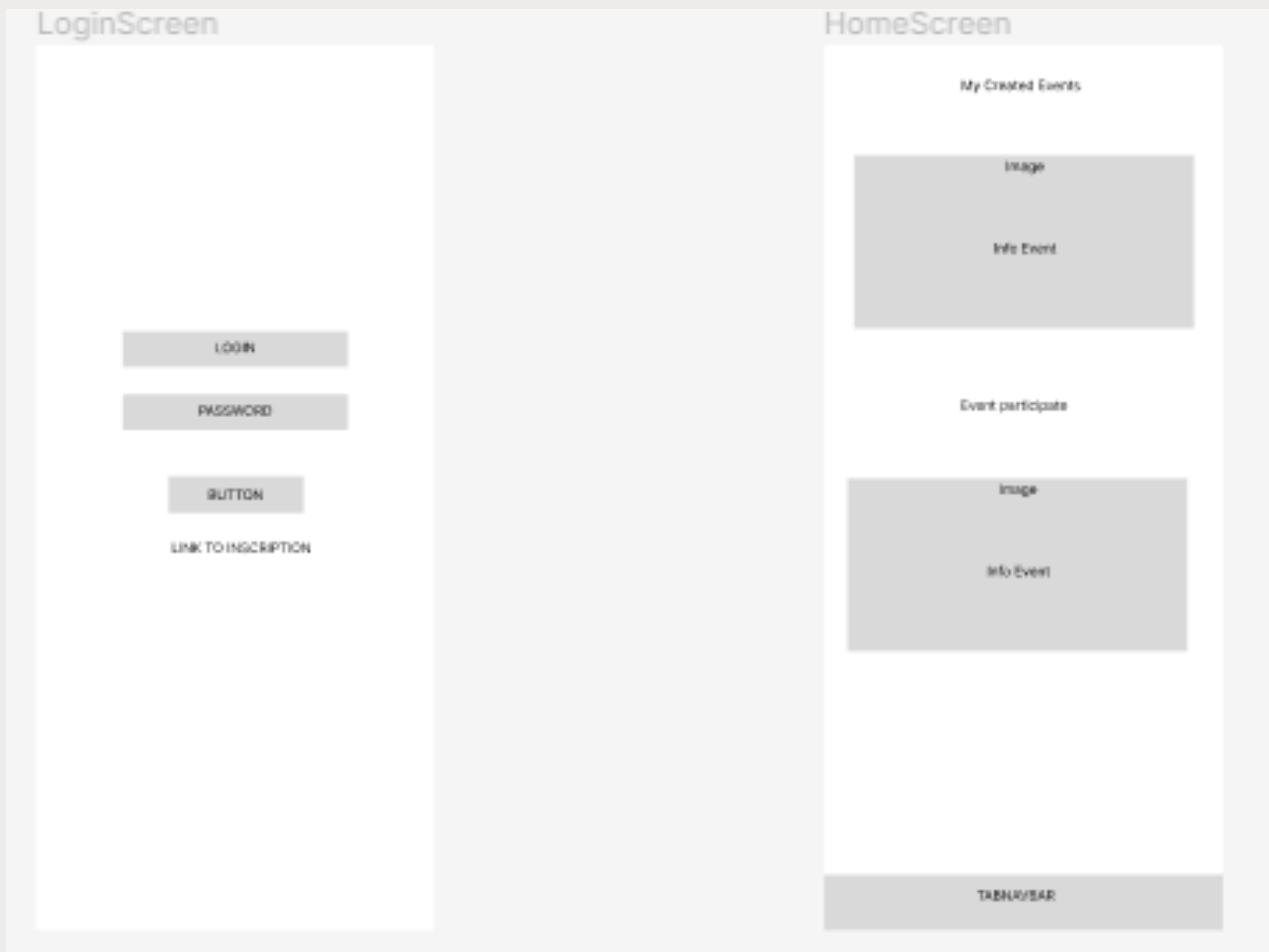
FIGMA



Figma est un outil de conception et de prototypage collaboratif en ligne. Il permet aux concepteurs et aux équipes de créer des interfaces utilisateur, des maquettes, des prototypes interactifs et bien plus encore. Figma est apprécié pour sa facilité d'utilisation, sa puissance et surtout sa capacité à permettre une collaboration en temps réel entre plusieurs utilisateurs.

02 L'ORGANISTION ET CONCEPTION DU PROJET

FIGMA

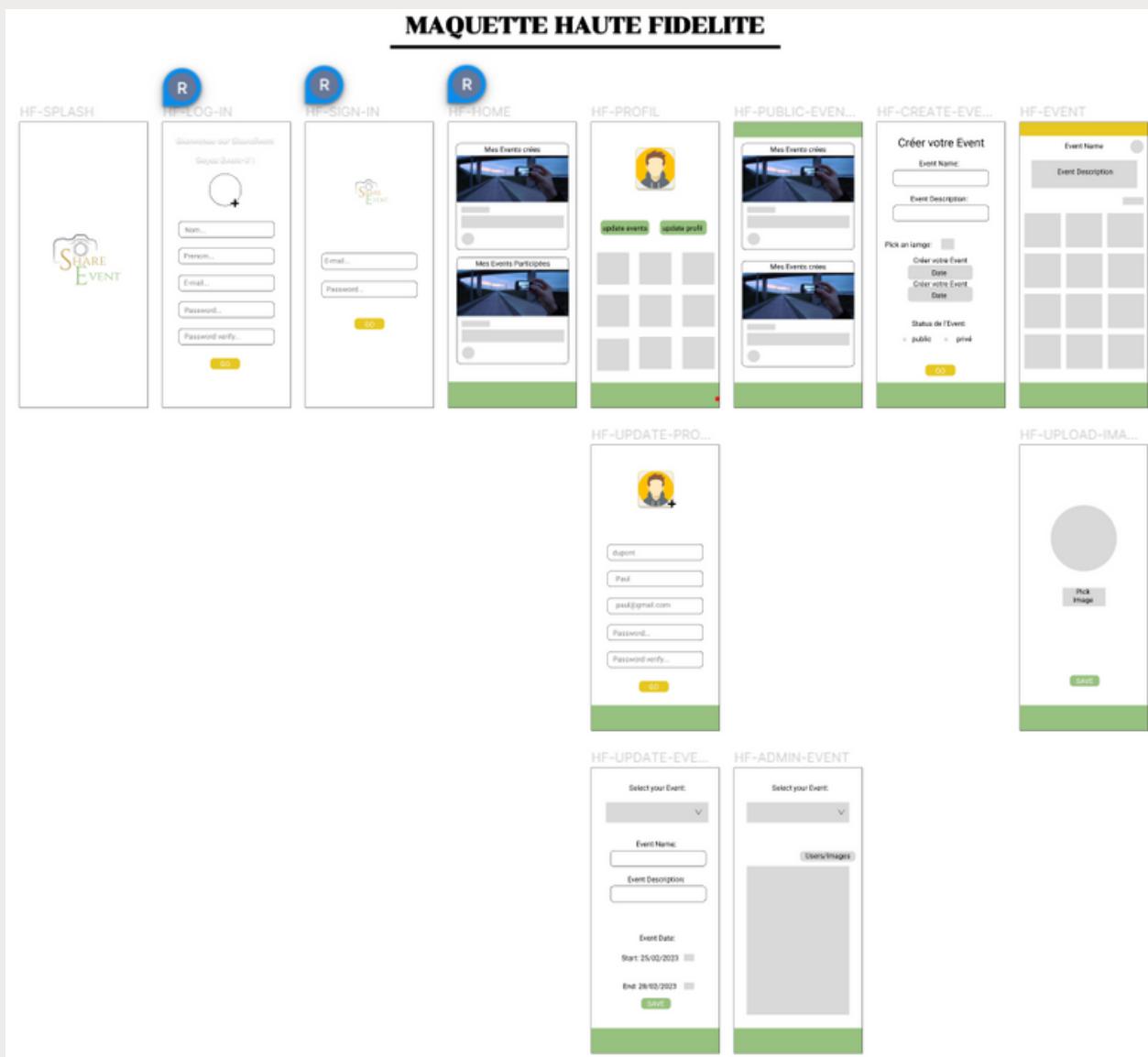


Divisez la maquette en zones fonctionnelles ou en sections distinctes pour représenter les différentes parties de l'interface ou du concept. Chaque zone devrait avoir un but spécifique.(zonning)

Le zonning (maquette basse fidélité) est une étape cruciale dans le processus de conception d'interface ou de concept. Cela permet de visualiser rapidement la structure et le flux global sans se soucier des détails visuels.

02 L'ORGANISTION ET CONCEPTION DU PROJET

FIGMA



Niveau de détail accru : Il représente une version plus précise de la disposition, des éléments et des interactions.

Éléments d'interface : Le wireframe inclut des éléments d'interface spécifiques tels que les boutons, les champs de texte, les menus déroulants, les icônes, les images, etc. Chaque élément est conçu avec plus de précision.

Disposition des éléments : Il définit la structure globale de la page ou de l'application.

Le wireframe est une étape essentielle pour détailler l'interface, ses interactions et son apparence générale avant de passer à la phase de conception plus avancée ou de développement. C'est un outil précieux pour communiquer la vision du projet aux membres de l'équipe et aux parties prenantes.

LUCIDCHART



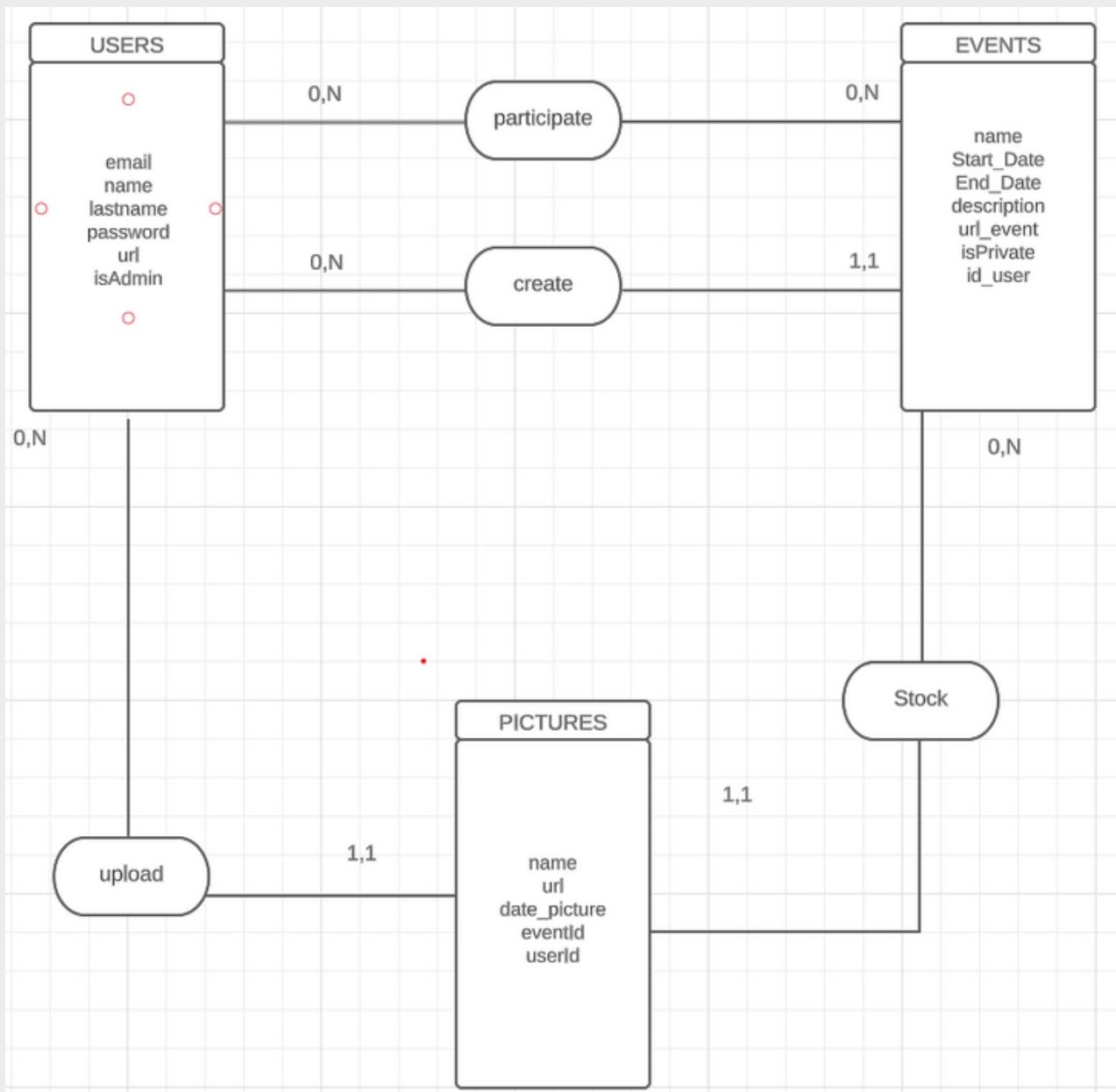
Nous avons utilisé lucidchart pour conceptualisé la base de donnée avec la methode merise

- Collaboration en temps réel
- Sauvegarde et accès dans le cloud
- Support et documentation complète
- Facilité d'utilisation

.

02 L'ORGANISTION ET CONCEPTION DU PROJET

MCD



Le MCD permet de représenter le système d'information indépendamment de son aspect informatique, il doit être compréhensible par tous: informaticiens, .

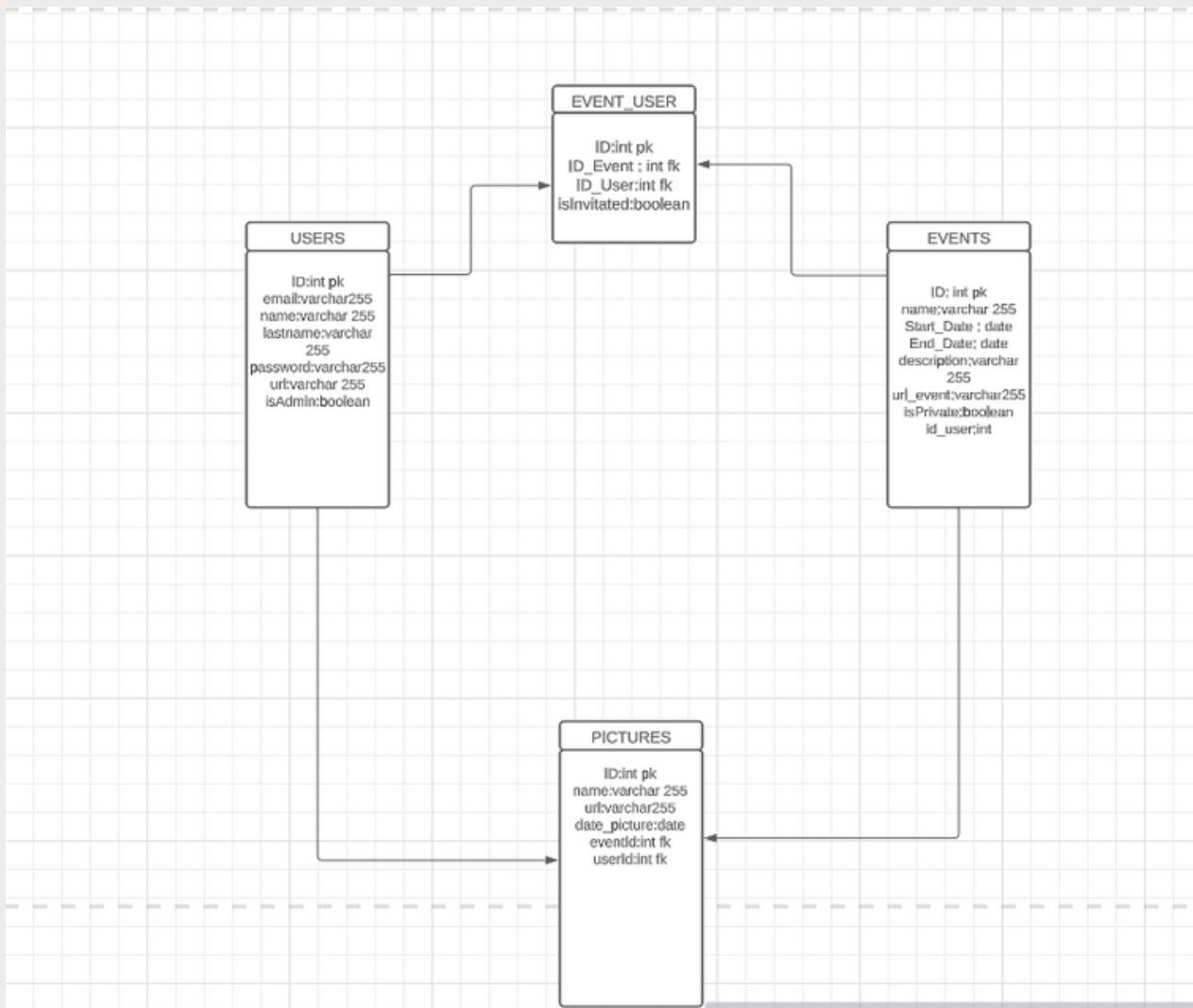
Pour ce faire la méthode Merise utilise 2 représentations:

l'entité: elle regroupe l'information statique et durable.

L'association (ou relation) est représentée par un verbe d'action ou d'état à l'infinitif.

02 L'ORGANISTION ET CONCEPTION DU PROJET

MLD



Les entités mises en relation deviennent des tables.

La clé dupliquée est appelée clé étrangère

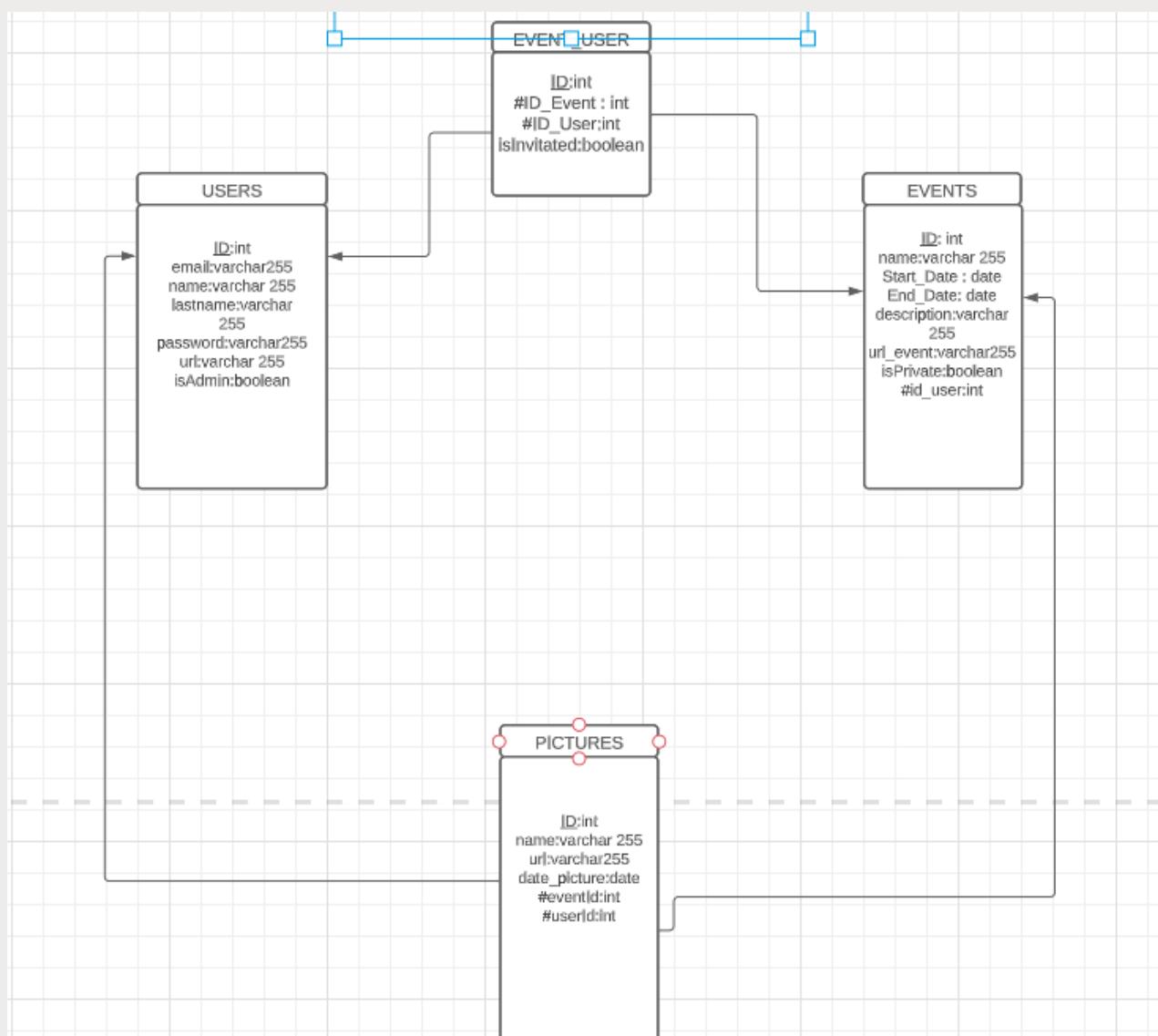
Association (1-n)(1-n)

L'association devient une table ayant comme clef primaire la concaténation des clefs primaires des 2 entités.

Une table issue d'une association peut avoir des attributs au même titre que n'importe quelle autre table

02 L'ORGANISTION ET CONCEPTION DU PROJET

MPD



cette étape permet de construire la structure finale de la base de données avec les différents liens entre les éléments qui la composent. Pour la peine, on change aussi de vocabulaire :

- Les entités se transforment en tables ;
- Les propriétés se transforment en champs (ou attributs) ;
- Les propriétés se trouvant au milieu d'une relation génèrent une nouvelle table ou glissent vers la table adéquate en fonction des cardinalités de la relation ;
- Les identifiants se transforment en clés et se retrouvent soulignés. Chaque table dispose d'au minimum 1 clé dite primaire ;
- Les relations et les cardinalités se transforment en champs parfois soulignés : il s'agit de créer des « clés étrangères » reliées à une « clé primaire » dans une autre table.

02 L'ORGANISTION ET CONCEPTION DU PROJET

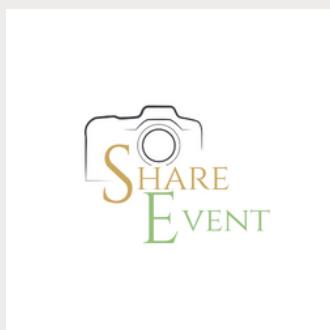
CAHIER DES CHARGES

DÉFINITION DU BESOIN

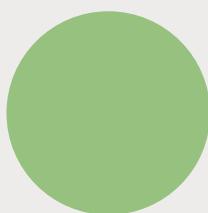
L'objectif principal de ShareEvent est de simplifier la création et le partage d'événements en fournissant une plateforme conviviale et intuitive. En utilisant ShareEvent, les utilisateurs peuvent créer des événements, inviter d'autres participants et partager facilement les photos prises lors de ces événements. L'application vise à offrir une expérience utilisateur fluide et à résoudre le problème de dispersion des photos sur différentes plateformes de médias sociaux.

CHARTE GRAPHIQUE

LOGO



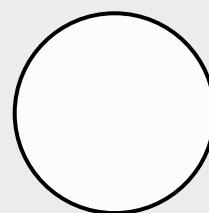
CODE COULEUR



#97C17E



#E6C821



#FCFCFC

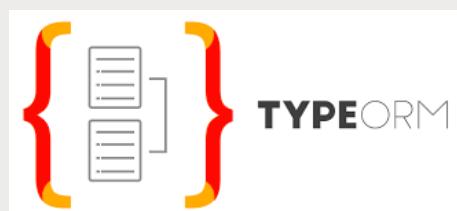
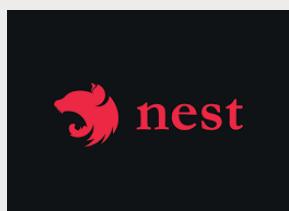
Typographie

CINZEL

OPEN SANS

CONTRAINTE TECHNOLOGIQUES

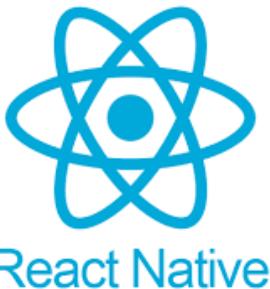
BACK END



02

L'ORGANISTION ET CONCEPTION DU PROJET

FRONT END



UTLITAIRE

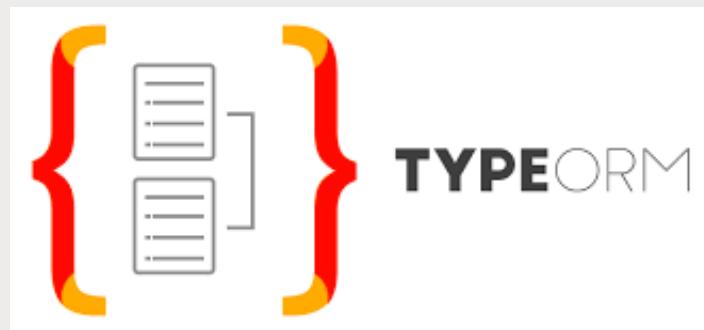
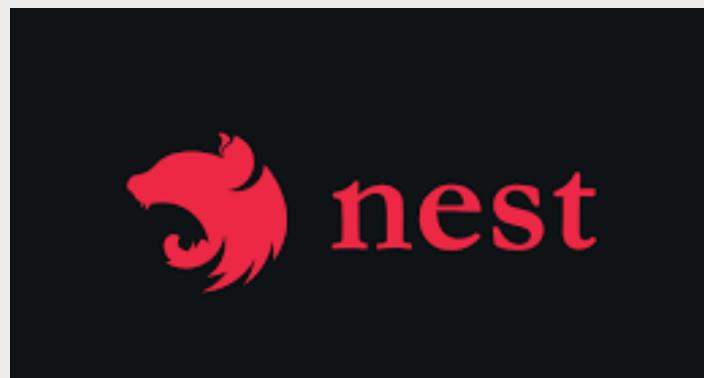


FIREBASE STORAGE



LANGUAGE







NestJS, en tant que framework back-end basé sur Node.js, facilite le développement d'applications serveur .

Basée sur l'architecture orientée vers les services (SOA) et la programmation orientée objet (POO), offre une base solide pour la gestion des routes, l'injection de dépendances et la gestion des requêtes HTTP. De plus, NestJS bénéficie d'une communauté active et de nombreuses ressources disponibles, ce qui facilite l'apprentissage et le développement.

Cependant, le choix de NestJS peut présenter une courbe d'apprentissage initiale plus élevée, en particulier pour les développeurs moins familiers avec TypeScript ou les frameworks back-end basés sur Node.js.

Cependant, une fois cette phase d'apprentissage terminée, NestJS offrent une base solide et cohérente pour développer et maintenir l'application ShareEvent.

03

PARTIE BACK END

```
@Module({
  imports: [TypeOrmModule.forFeature([User,UserEvent]),UsersModule],
  exports: [TypeOrmModule],
  providers: [UsersService],
  controllers: [UsersController],
})
export class UsersModule {}
```

```
import { Controller, Post, Body, Delete } from '@nestjs/common';
import { Get, Patch, Put } from '@nestjs/common/decorators';
import { User } from './model/entities/users.entity';
import { UsersService } from './users.service';
import { Param } from '@nestjs/common/decorators';
import { UpdateUserDto } from './dto/update-user.dto';
import { CreateUserDto } from './dto/create-user.dto';

@Controller('users')
export class UsersController {
  constructor(private readonly usersService: UsersService) {}

  @Post('register')
  async register(@Body() user: CreateUserDto) {
    return this.usersService.create(user);
  }
}
```

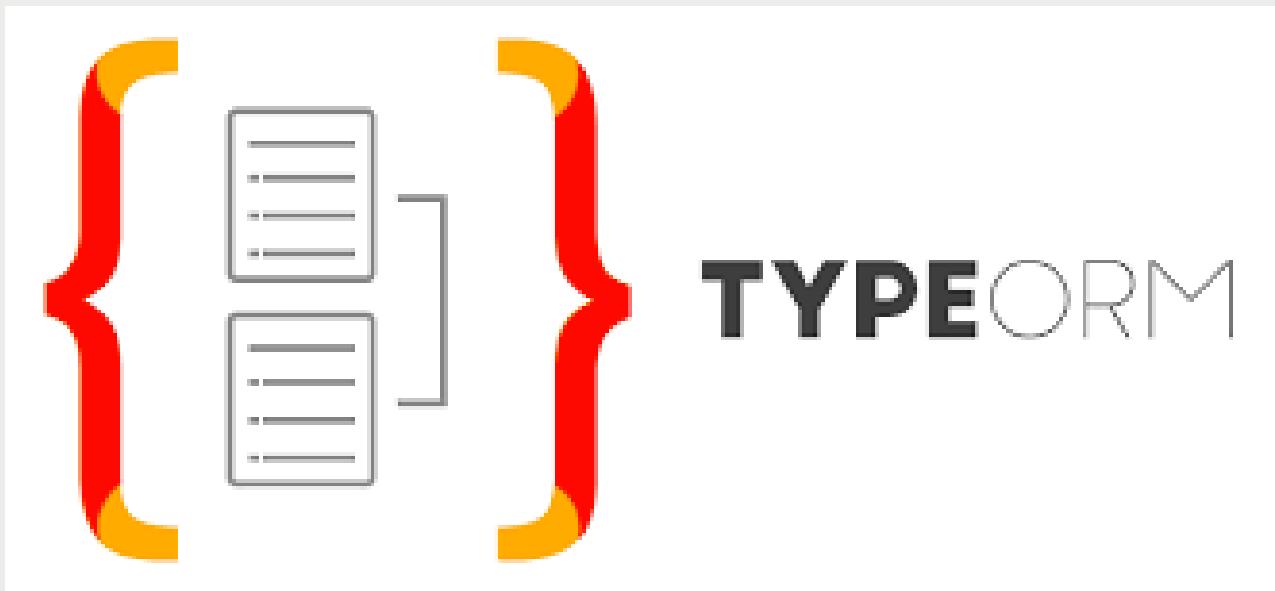
```
  @Get()
  findAll() {
    return this.usersService.findAll();
  }
```

```
  @Get()
  findAll(): Promise<User[]> {
    return await this.usersRepository.find();
  }

  async findOneByEmail(email:string){
    return await this.usersRepository.findOneBy({email:email});
  }
  async findOne(id: number) {
    return await this.usersRepository.findOneBy({ id });
  }

  async updateUser(id: number, user: UpdateUserDto): Promise<any> {
    var passwordHash= user.password;
    if(user.password){
      passwordHash = await bcrypt.hash(user.password,10);
    }
  }

  const updateuserDatabase = await this.usersRepository.findOneBy({id:id});
  updateuserDatabase.id= id;
```



TypeORM, en tant qu'ORM compatible avec NestJS, simplifie considérablement l'interaction avec la base de données en utilisant des objets et des requêtes en langage TypeScript.

Cela permet aux développeurs de modéliser et de manipuler les données de manière intuitive, en évitant la nécessité d'écrire des requêtes SQL manuellement. TypeORM offre également des fonctionnalités telles que la migration de base de données et la gestion des relations entre les entités, ce qui facilite la gestion de la persistance des données.

TYPEORM

```
import { IsEmail } from "class-validator";
import { Picture } from "src/pictures/model/entities/pictures.entity";

import { Entity, PrimaryGeneratedColumn, Column, JoinTable, ManyToMany, BeforeInsert, OneToMany, Unique, BaseEntity } from "typeorm"
import { Event } from "../../events/model/entities/events.entity"
@Entity()
@Unique(['email'])
export class User extends BaseEntity {
    @PrimaryGeneratedColumn()
    id: number;

    @IsEmail()
    @Column({ length: 255 })
    email: string;

    @Column({ length: 255 })
    name: string;

    @Column({ length: 255 })
    url: string;

    @Column({ length: 255 })
    lastname: string;

    @Column({ length: 255 })
    @BeforeInsert()
    password: string;

    @Column()
    isAdmin:boolean;

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

Le decorateur Unique par exemple Permet d interdire la creation de deux mail identique :

type orm vas verifié que l'email n'existe pas avant de l'implémenter en base de donnée

SQL



SQL (sigle de Structured Query Language, en français langage de requête structurée) est un langage informatique normalisé servant à exploiter des bases de données relationnelles. La partie langage de manipulation des données de SQL permet de rechercher, d'ajouter, de modifier ou de supprimer des données dans les bases de données relationnelles.

nous avons créer notre base de donnée via l'interface graphique de wamp phpmyadmin.

SQL

```
async findEventByIdAdmin(id_user: number):Promise<Event[]> {
    return this.eventsRepository.findBy({ id_user: id_user });
}

async findAllUserInEvent(id_event: number):Promise<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()
}

async FindAllEvent():Promise<Event[]> {
    return this.eventsRepository.findBy({isPrivate:false});
}

async findInvitation(id_user:number){
    return await this.eventsRepository
        .createQueryBuilder()
        .select("event.description,event.start_date,event.end_date,event.name,event.id,user_event.id")
        .innerJoin("user_event","user_event","user_event.eventId = event.id")
        .innerJoin("user","user","user_event.userId = user.id")
        .where("user.id = "+ id_user +" and user_event.isInvitated = false")
        .execute()
}

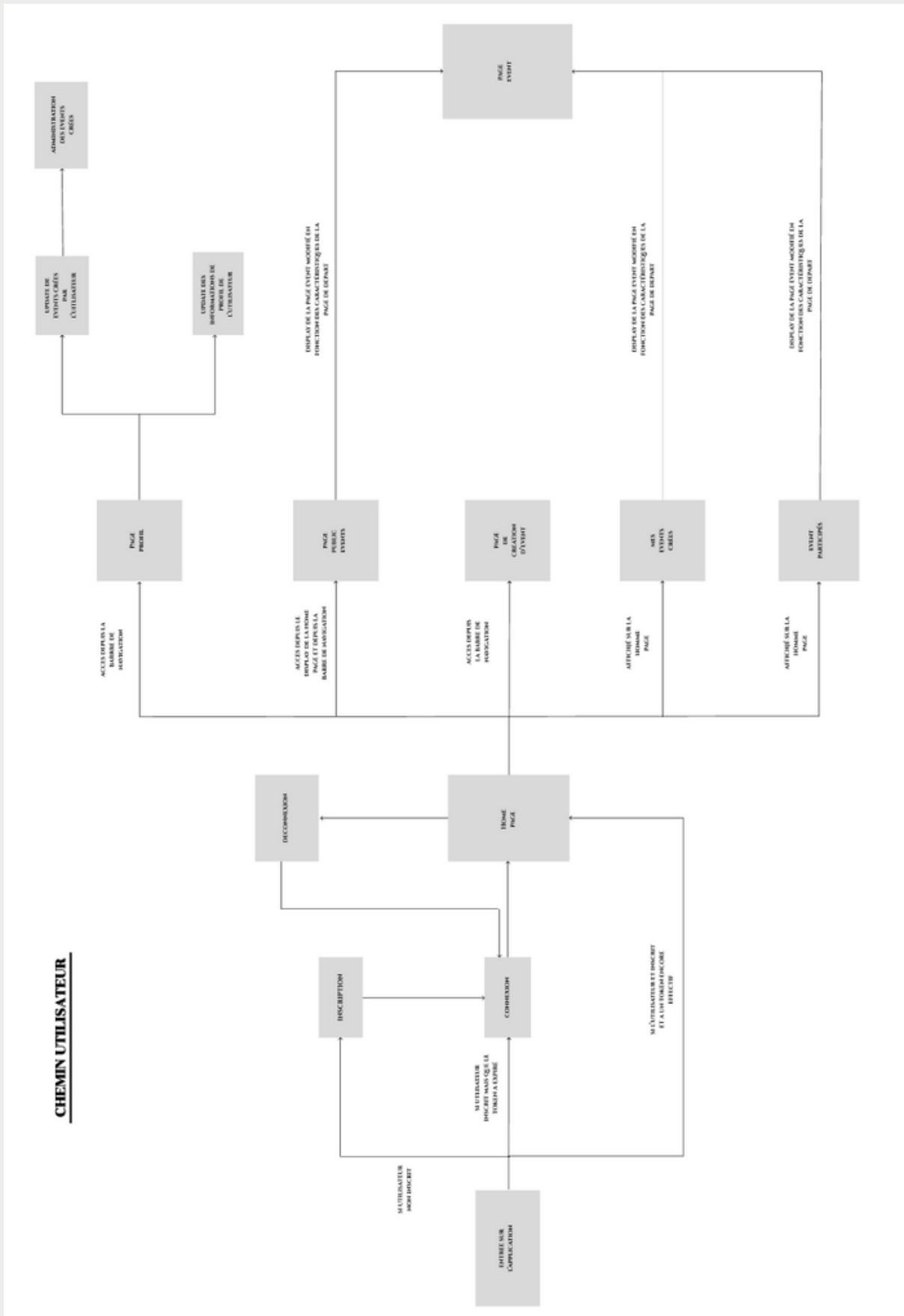
delete_Invitation(id_user_event:number){
    return this.userEventRepository.delete(id_user_event);
}
```

```
query: SELECT `Event`.`id` AS `Event_id`, `Event`.`name` AS `Event_name`, `Event`.`start_date` AS `Event_start_date`, `Event`.`end_date` AS `Event_end_date`, `Event`.`id_user` AS `Event_id_user`, `Event`.`description` AS `Event_description`, `Event`.`url_event` AS `Event_url_event`, `Event`.`isPrivate` AS `Event_isPrivate` FROM `event` `Event`
```

Les méthodes typeOrm & les méthodes de nestjs nous permettent de communiquer avec la base de donnée celles ci sont retranscrites en language sql.

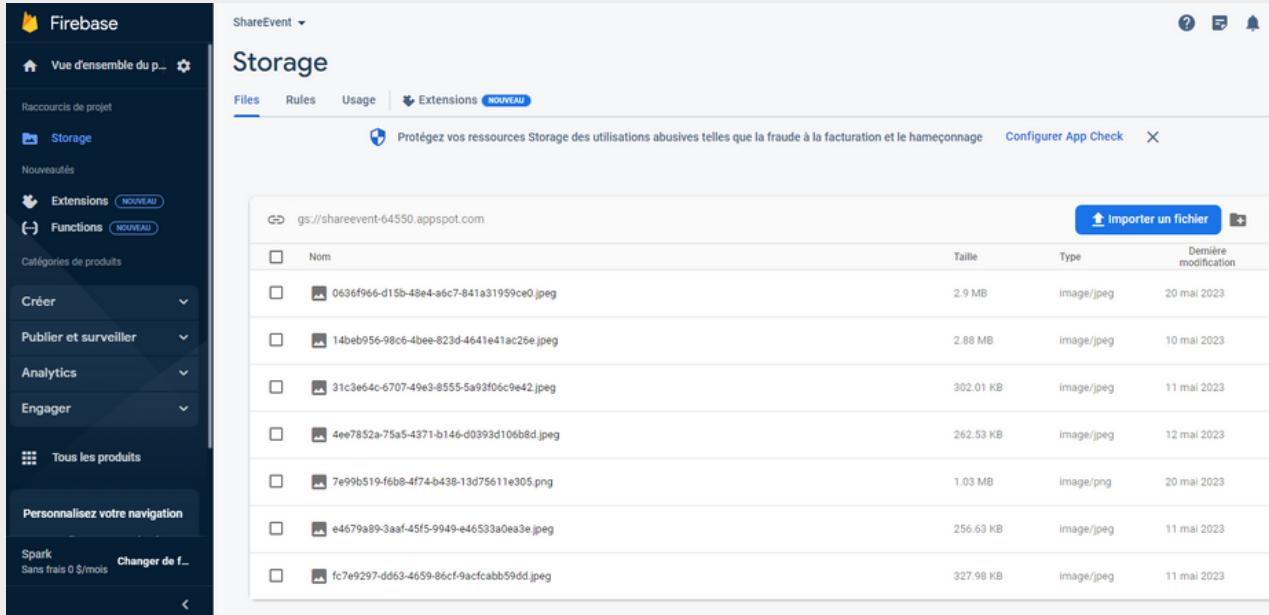
03 L'ORGANISTION ET CONCEPTION DU PROJET

CHEMIN UTILISATEUR



03

PARTIE BACK END



The screenshot shows the Firebase Storage interface for a project named "ShareEvent". On the left, there's a sidebar with navigation links like "Vue d'ensemble du p...", "Storage", "Extensions", "Functions", "Créer", "Publier et surveiller", "Analytics", "Engager", and "Tous les produits". The main area is titled "Storage" and has tabs for "Files", "Rules", "Usage", and "Extensions". A banner at the top says "Protégez vos ressources Storage des utilisations abusives telles que la fraude à la facturation et le hameçonnage". Below the banner is a table listing files:

| Nom | Taille | Type | Dernière modification |
|---|-----------|------------|-----------------------|
| 0636f966-d15b-48e4-a6c7-841a31959ce0.jpeg | 2.9 MB | image/jpeg | 20 mai 2023 |
| 14beb956-98c6-4bee-823d-4641e41ac26e.jpeg | 2.88 MB | image/jpeg | 10 mai 2023 |
| 31c3e64c-6707-49e3-8555-5a93f06c9e42.jpeg | 302.01 KB | image/jpeg | 11 mai 2023 |
| 4ee7852a-75a5-4371-b146-d0393d106b8d.jpeg | 262.53 KB | image/jpeg | 12 mai 2023 |
| 7e99b519-f6b8-4f74-b438-13d75611e305.png | 1.03 MB | image/png | 20 mai 2023 |
| e4679a89-3aa9-45f5-9949-e46533a0ea3e.jpeg | 256.63 KB | image/jpeg | 11 mai 2023 |
| fc7e9297-dd63-4659-86cf-9acfabb59dd.jpeg | 327.98 KB | image/jpeg | 11 mai 2023 |

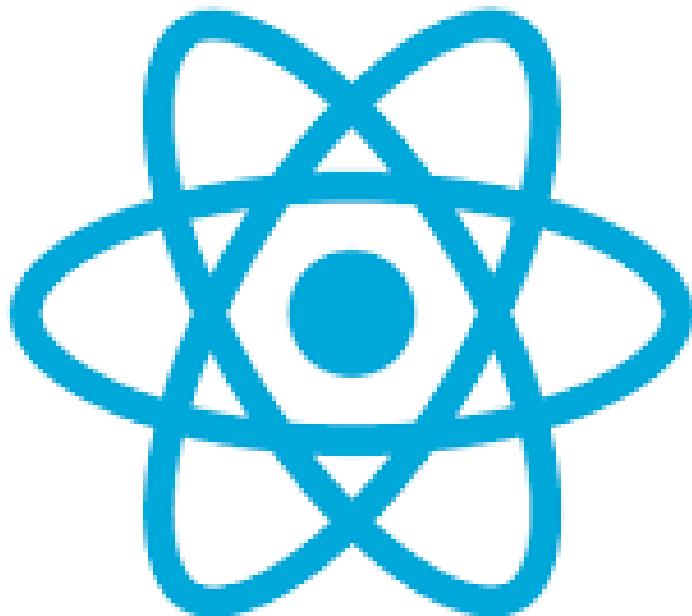
```
import firebase from 'firebase/compat/app';
import 'firebase/compat/auth';
import 'firebase/compat/firestore';
import 'firebase/compat/storage';

const firebaseConfig = {
  apiKey: "AIzaSyDozoDtF77kh6NyjLT5fZJNuSTro4faTUA",
  authDomain: "shareevent-64550.firebaseio.com",
  projectId: "shareevent-64550",
  storageBucket: "shareevent-64550.appspot.com",
  messagingSenderId: "291923698562",
  appId: "1:291923698562:web:ea626819e667b54cbee59b"
};

if (!firebase.apps.length) {
  firebase.initializeApp(firebaseConfig);
}

export { firebase };
```

Firebase storage nous permet de stocké les images de manière efficace cela nous évite de les stocké dans l appli ou dans la base de donnée



Le choix de React Native pour la partie front-end mobile de ShareEvent présente plusieurs avantages significatifs:

- il est possible de développer une application mobile multiplateforme
- la vaste bibliothèque de composants préconstruits et les nombreuses bibliothèques tierces disponibles facilitent la création d'interfaces utilisateur.

Cependant, l'utilisation de React Native peut présenter quelques inconvénients potentiels:

- Certaines fonctionnalités avancées spécifiques à chaque plateforme peuvent être plus difficiles à implémenter avec React Native,
- les performances de l'application peuvent être légèrement inférieures à celles d'une application entièrement native, bien que les améliorations constantes de React Native contribuent à atténuer cet écart.

```
function HomeUserScreen() {  
    const navigation = useNavigation()  
  
    const handleLogout = async () => {  
        // Delete the access token from the secure storage  
        await SecureStore.deleteItemAsync('acess_token');  
        navigation.navigate('Login');  
    };  
  
    return (  
        <View style={styles.container}>  
            <ImageCarousel navigation={navigation} />  
  
            <CarouselEvent navigation={navigation} />  
            <Pressable onPress={handleLogout}><Text style={styles.logout}>Logout</Text></Pressable>  
        </View>  
    );  
};
```

```
export const ImageCarousel = ({ navigation }: { navigation: any }) => {  
    const [data, setData] = useState<any[]>([]);  
    const [userID, setuserID] = useState<number>();  
  
    useEffect(() => {  
        // Fetch user's information from the server  
        const fetchUserData = async () => {  
            try {  
                const token: any = await SecureStore.getItemAsync('acess_token');  
                if (token) {  
                    const decoded: any = jwt_decode(token);  
                    // console.log(decoded)  
                    setuserID(decoded.id)  
                }  
  
            } catch (error) {  
                console.log('Invalid token:', error);  
                // Show message to the user here  
            }  
        };  
  
        fetchUserData();  
    }, [userID]);
```



Expo, est à la fois un framework et une plateforme qui simplifient la création et le déploiement d'applications mobiles avec React Native.

Expo embarque de nombreux outils utiles et des librairies natives pour React Native. Il gère aussi la mise à jour de ces librairies. C'est donc un moyen de démarrer facilement et rapidement son projet.

Expo a l'avantage d'être très simple à configurer et il simplifie grandement la vie des développeurs, à tous les stades du projet .

Expo : les inconvénients

le poids des applications générées par Expo est très élevé.

il manque des fonctionnalités cruciales à Expo : par exemple, votre application ne pourra pas intégrer d'achats in app, ni utiliser le Bluetooth.

le temps de build peut être très long en passant par les serveurs d'Expo,

FONCTIONNALITÉ

```
const {width} = Dimensions.get('window');
const SPACING = 5;
const ITEM_LENGTH = width * 1; // Item is a square. Therefore, its height and width are of the same length.
const EMPTY_ITEM_LENGTH = (width - ITEM_LENGTH) / 2;
const BORDER_RADIUS = 20;
const CURRENT_ITEM_TRANSLATE_Y = 20;

// interface ImageCarouselProps {
//   data: ImageCarouselItem[];
// }

export const CarouselEvent=()=> {

  const[data,setData]=useState<any>([]);
  useEffect(() => {
    async function GetAllEvent() {
      try {
        const response: any = await axios.get("http://10.10.22.116:3000/events");
        console.log(response.data);
        setData(response.data);
      }
      catch (error) {
        console.error(error);
      }
    }
    GetAllEvent(),
    []
  });

  const scrollX = useRef(new Animated.Value(0)).current;
```

Dans ShareEvent, plusieurs fonctionnalités clés ont été implémentées dans la partie front-end à l'aide de React Native. Cela inclut :

Création d'événements

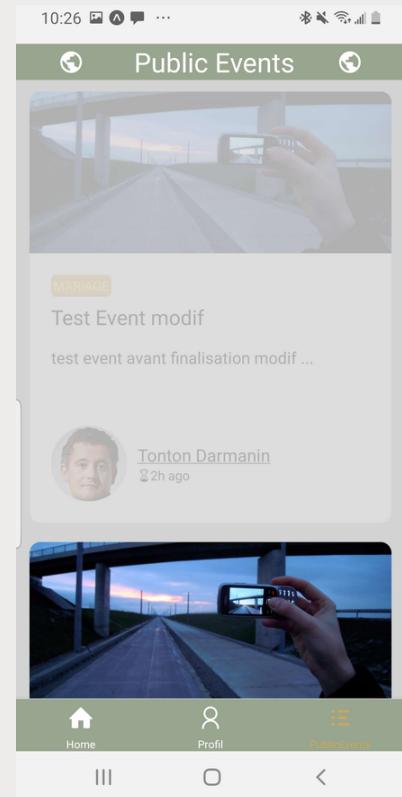
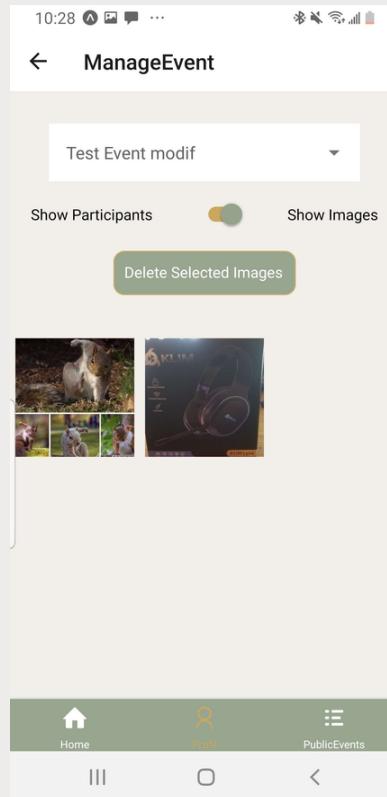
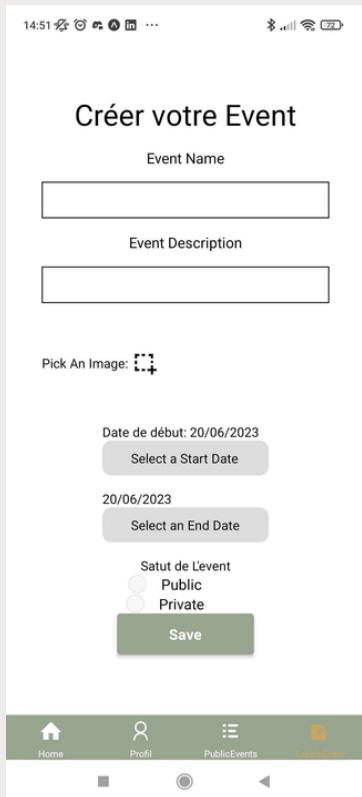
Affichage des événements

Invitation des participants

Partage de photos

04

PARTIE FRONT-END





Axios est une bibliothèque JavaScript utilisée pour effectuer des requêtes HTTP depuis un navigateur ou depuis un serveur utilisant Node.js. Elle simplifie et facilite les appels réseau en fournissant une interface simple et intuitive pour interagir avec des API et récupérer des données à partir de serveurs distants :

Interception des requêtes et des réponses : Axios offre la possibilité d'intercepter les requêtes et les réponses avant qu'elles ne soient envoyées ou traitées. Cela permet de manipuler les données, d'ajouter des en-têtes personnalisés, de gérer les erreurs de manière centralisée, ou d'authentifier les requêtes avec des jetons d'accès.

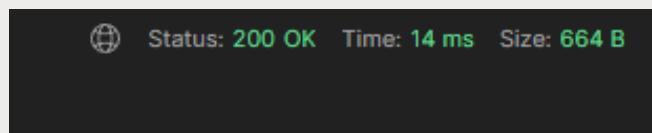
Gestion avancée des erreurs : Axios facilite la gestion des erreurs liées aux requêtes HTTP. Il capture les erreurs réseau, les erreurs de connexion, les erreurs HTTP et permet de les traiter de manière cohérente

axios peut être considéré comme un middleware.

05

COMMUNICATION FRONT/BACK

```
useEffect(() => {
  async function GetAllEvent() {
    if(events.length === 0){
      try {
        const response: any = await axios.get(route+"/events/FindAllEvent");
        setEvents(response.data);
      } catch (error) {
        console.error(error);
      }
    }else{
      return;
    }
  }
  GetAllEvent();
}, []);
```



Il désigne une catégorie de composants logiciels , englobant des environnements d'exécution d'applications, des solutions d'intégration d'applications d'entreprise et différents types de services cloud. La gestion des données, les services d'applications, la messagerie, l'authentification et la gestion des API sont des services communément gérés par les solutions de middleware.

- agit comme une couche intermédiaire entre différentes applications ou services
- intervient dans le traitement des requêtes entrantes et sortantes
- diverses tâches telles que la gestion des autorisations, la transformation de données



Open Web Application Security Project (OWASP), désormais Open Worldwide Application Security Project, est une communauté en ligne travaillant sur la sécurité des applications Web. Sa philosophie est d'être à la fois libre et ouverte à tous.

Elle a pour vocation de publier des recommandations de sécurisation Web et de proposer aux internautes, administrateurs et entreprises des méthodes et outils de référence permettant de contrôler le niveau de sécurisation de ses applications Web.



SANITIZEHTML

```
const cleanName = sanitizeHtml(name);
const cleanLastName = sanitizeHtml(lastname);
const cleanEmail = sanitizeHtml(email);
const cleanPassword = sanitizeHtml(password);
```

```
await axios.post(route+`/users/register`, {
    name: cleanName ,
    lastname : cleanLastName,
    email: cleanEmail,
    password: cleanPassword ,
    url
});
```

Nettoyage des entrées utilisateur pour prévenir les attaques XSS :

Les attaques XSS (Cross-Site Scripting) sont une menace courante pour la sécurité des applications web. Pour prévenir ces attaques, la bibliothèque sanitize-html pour nettoyer les données entrées par les utilisateurs avant de les envoyer au serveur.

Cela supprime tout contenu potentiellement dangereux et réduit considérablement le risque d'exécution de code malveillant sur le client.

Voici un exemple de code qui montre comment utiliser sanitize-html pour nettoyer les données avant de les envoyer au serveur

LES GUARDS



Les guards ont une responsabilité unique . Ils déterminent si une requête donnée sera traitée par le gestionnaire de route ou non, en fonction de certaines conditions (comme les autorisations, les rôles,, etc.) présentes au moment de l'exécution

LES GUARDS

```
import { Injectable, CanActivate, ExecutionContext } from '@nestjs/common';
import { Observable } from 'rxjs';
import { JwtService } from '@nestjs/jwt';

@Injectable()
export class AdminGuard implements CanActivate {
  constructor(private jwtService: JwtService) {}

  canActivate(context: ExecutionContext): boolean | Promise<boolean> | Observable<boolean> {
    const request = context.switchToHttp().getRequest();

    const token = request.headers.authorization; // assuming the token is sent in the "Authorization" header
    if (token == null || token === undefined) {
      return false;
    }

    try {
      const decodedToken = this.jwtService.decode(token) as { isAdmin: boolean };
      if(decodedToken.isAdmin == true){
        return true
      }
      else{
        return false
      }
    }
    catch{
      return false
    }
  }
}
```

extrait de code du guard sur le chemin admin

passwordHash

```
        "id": 37,
        "email": "yacine.boucetta@laplateforme.io",
        "name": "Yacine",
        "url": "",
        "lastname": "Boucetta",
        "password": "$2b$10$y7B8p3xUFhUKgZqsHN.cjuC6C7S7IbiuwHR0eYZV0if4aNmcPLOq",
        "isAdmin": true
```

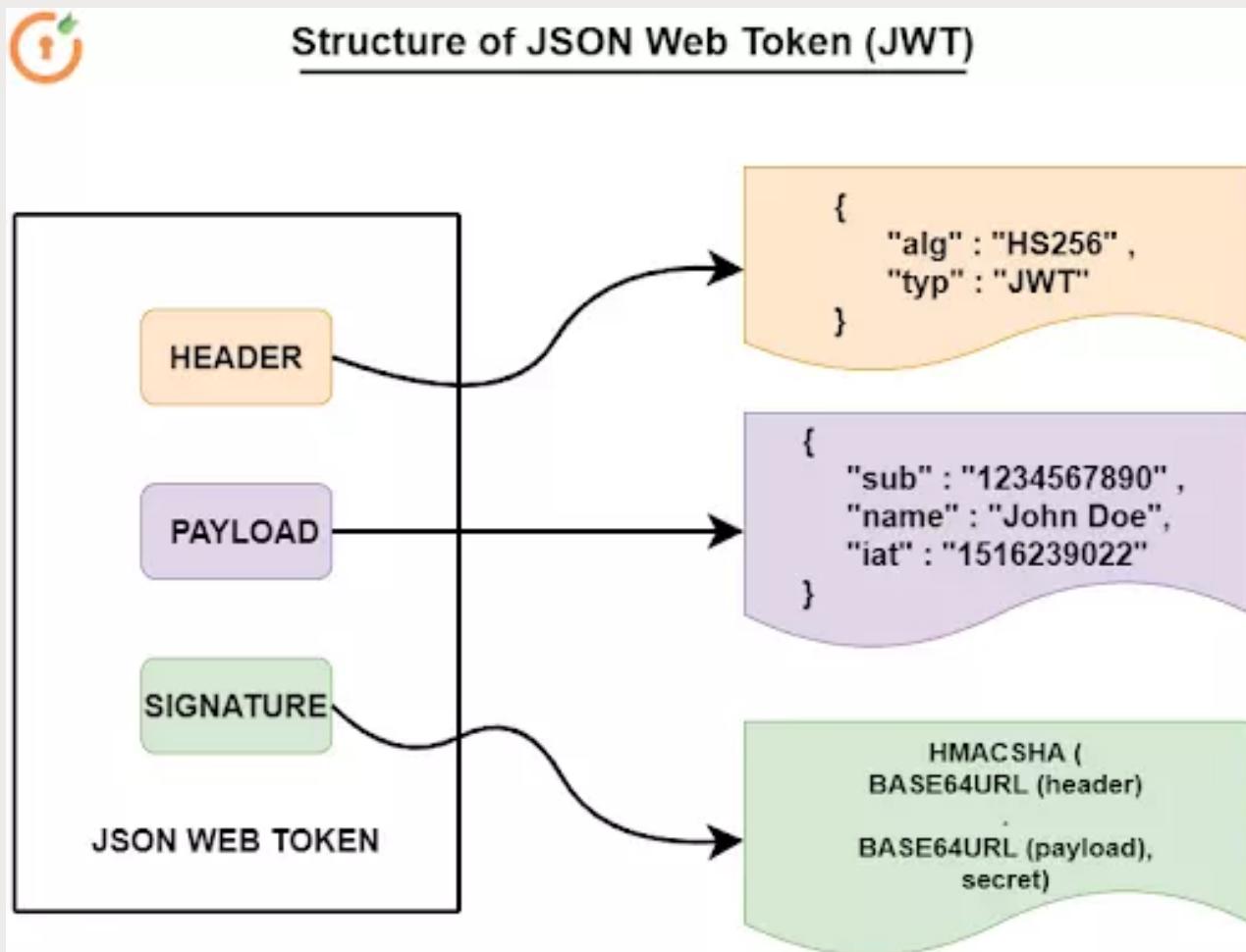
```
async create(user: CreateUserDto) {
    const passwordHash= await bcrypt.hash(user.password,10);
    const getUserbefore= await this.usersRepository.findAndCountBy({isAdmin:true});

    if(getUserbefore[1] >=1){
        return this.usersRepository.save(
            {
                email:user.email,
                name:user.name,
                url:user.url,
                lastname:user.lastname,
                password: passwordHash,
                isAdmin:false,
            }
        )
    }
}
```

Le hachage de mot de passe permet de chiffrer de manière irréversible les mots de passe via une fonction de hachage cryptographique

actuellement le hachage contient 60 caractère alphanumerique.

Jwt



Les « JSON Web Token » ou JWT sont des jetons générés par un serveur lors de l'authentification d'un utilisateur sur une application Web, et qui sont ensuite transmis au client.

Ils seront renvoyés avec chaque requête HTTP au serveur, ce qui lui permettra d'identifier l'utilisateur.

Il se compose de 3 parties:

Une partie “Header”, contenant l'algorithme utilisé pour la signature ainsi que le type de jeton (dans notre cas toujours “JWT”), en JSON encodé en Base64

Une partie “Payload” contenant les informations du jeton, comme par exemple le nom de l'utilisateur, la date d'émission du jeton ou sa date d'expiration le tout en JSON encodé en Base64

Une partie “Signature”, qui correspond à la concaténation des parties “Header” et “Payload” chiffrée avec la clé privée.

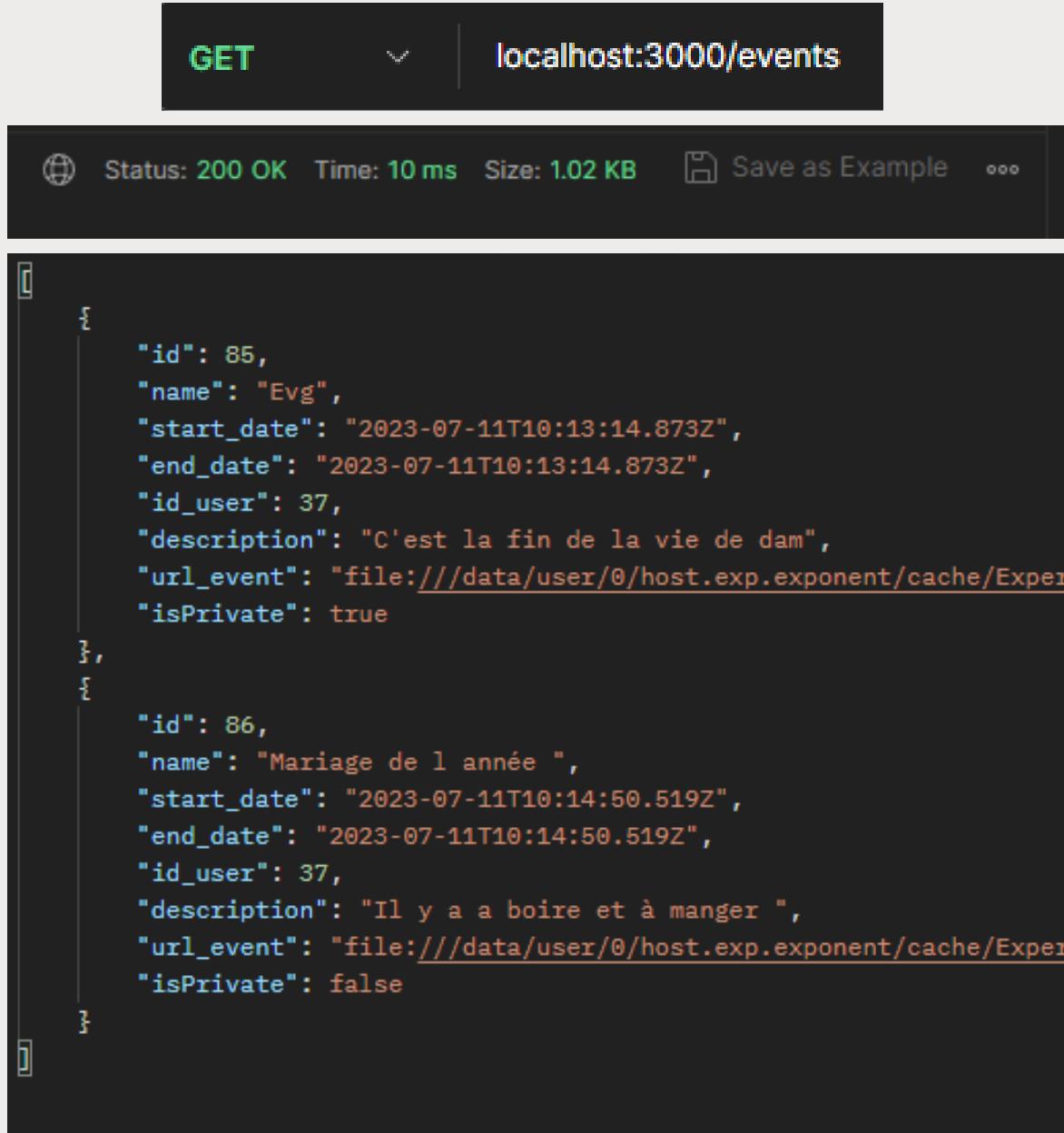


POSTMAN

En programmation informatique, le test unitaire est une procédure permettant de vérifier le bon fonctionnement d'une partie précise d'un logiciel ou d'une portion d'un programme (appelée « unité » ou « module »).

Pour ce faire plusieurs outils peuvent être utilisés : nous avons utilisés postman

Postman



The screenshot shows a Postman request for a GET operation on the URL `localhost:3000/events`. The response status is 200 OK, with a time of 10 ms and a size of 1.02 KB. The response body is a JSON array containing two event objects:

```
[{"id": 85, "name": "Evg", "start_date": "2023-07-11T10:13:14.873Z", "end_date": "2023-07-11T10:13:14.873Z", "id_user": 37, "description": "C'est la fin de la vie de dam", "url_event": "file:///data/user/0/host.exp.exponent/cache/Exper", "isPrivate": true}, {"id": 86, "name": "Mariage de l'année", "start_date": "2023-07-11T10:14:50.519Z", "end_date": "2023-07-11T10:14:50.519Z", "id_user": 37, "description": "Il y a à boire et à manger", "url_event": "file:///data/user/0/host.exp.exponent/cache/Exper", "isPrivate": false}]
```

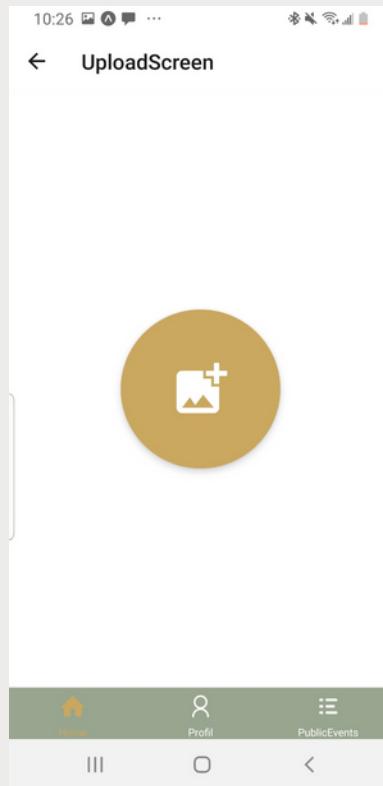
Le code 200 est le status ok c'est à dire que la requete c'est bien exécuté .

il existe plusiseur 'famille ' de code status les plus fréquents sont :

code 400 erreur coté client

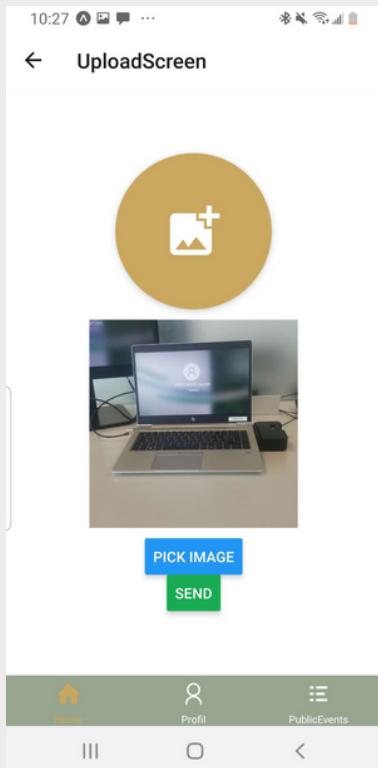
code 500 erreur coté serveur

code 200 status ok



```
return (
  <View>
    <View style={styles.buttonContainer}>
      <TouchableOpacity
        onPress={() => navigation.navigate('UploadScreen', { eventId: eventId })}
        style={styles.button}
      >
        <AntDesign name="pluscircle" size={24} color="black" />
      </TouchableOpacity>
    </View>
    <View style={styles.container}>
      {imageUri ? (
        <Image source={{ uri: imageUri }} style={styles.imagePreview} />
      ) : (
        <Text style={styles.noImage}>No image selected</Text>
      )}
      <Button title="Pick Image" onPress={pickImage} />
      <Button title="Send" onPress={sendImage} color="#19AC52" disabled={!imageUri} />
    </View>
  );
}

const styles = StyleSheet.create({
```



```

useEffect(() => {
  // Check for media library permissions on mount
  const checkMediaLibraryPermissions = async () => {
    const { status } = await ImagePicker.requestMediaLibraryPermissionsAsync();
    if (status !== "granted") {
      alert("Sorry, we need camera roll permissions to make this work!");
    }
  };

  if (Platform.OS !== "web") {
    checkMediaLibraryPermissions();
  }
}, []);

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);
    }
  }
}

```

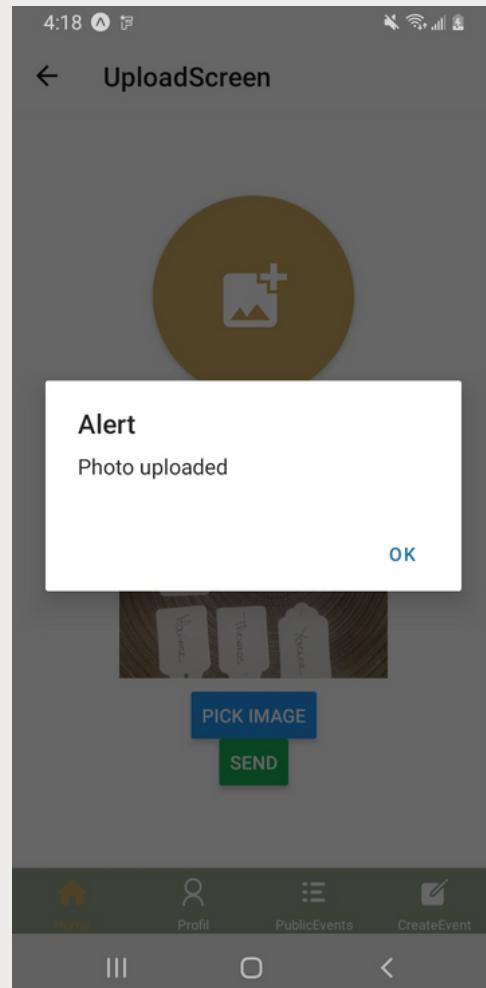
```

const saveImage2 = async (uri: string, eventId: number | null) => {
  try {
    const response = await fetch(imageUri);
    const blob = await response.blob();
    const filename = imageUri.substring(imageUri.lastIndexOf("/") + 1);
    var ref = firebase.storage().ref().child(filename).put(blob);

    try {
      ref.then(async (snapshot) => {
        const url = await snapshot.ref.getDownloadURL();
        console.log(url);
        console.log(userId);

        // Conditionally update the request data based on the uploadPurpose
        const requestData =
          uploadPurpose === "event"
            ? {
                name: filename,
                url: url,
                userId: userId,
                eventId: eventId,
              }
            : {
                name: filename,
                url: url,
                userId: userId,
                eventId: eventId,
              };
        ...
      });
    } catch (error) {
      console.error(error);
    }
  } catch (error) {
    console.error(error);
  }
}

```



```
https://firebasestorage.googleapis.com/v0/b/shareevent-64550.appspot.com/o/1b8bc966-f3b2-4a0f-9429-d416c7abb444.jpeg?alt=media&token=e8912581-5d8f-4228-a1bd-c1c5b76aafe7
40
Response from server: Object {
  "config": Object {
    "adapter": Array [
      "xhr",
      "http",
    ],
    "baseUrl": "http://192.168.1.29:3000",
    "data": "{\"name\":\"1b8bc966-f3b2-4a0f-9429-d416c7abb444.jpeg\",\"url\":\"https://firebasestorage.googleapis.com/v0/b/shareevent-64550.app
7abb444.jpeg?alt=media&token=e8912581-5d8f-4228-a1bd-c1c5b76aafe7\"}, {"userId":40, "eventId":70},
    "env": Object {
      "Blob": [Function Blob],
      "FormData": [Function FormData],
    },
  },
}
```



nouvelle fonctionnalités

- Bannir des utilisateurs
- système de reporting pour les signalements pour le non respect des cgu
- amélioration de la sécurité
- amélioré certaines fonctionnalités dans le cadre de la rgpd