

Dossier de projet professionnel



Jarod Rossini

Sommaire

| | |
|--|-----------|
| I – Présentation | 4 |
| 1. Présentation personnelle | 4 |
| 2. Présentation du projet en Anglais | 4 |
| 3. Liste des compétences visées | 4 |
| II – Organisation | 6 |
| 1. Fonctionnalités | 6 |
| 2. Répartition des tâches | 7 |
| 3. Outils utilisés | 7 |
| III – Conception | 8 |
| 1. Base de données | 8 |
| 2. Frontend | 8 |
| 3. Backend | 15 |
| 3.1 Backend de l'Application | 15 |
| IV - Charte graphique | 22 |
| 1. Couleurs | 22 |
| 2. Maquette | 23 |
| V – Cahier des charges | 24 |
| 1. Présentation d'ensemble du projet | 24 |
| 1.1. Les objectifs de l'application : | 24 |
| 1.2. La cible adressée par l'application : | 24 |
| 1.3. Objectifs quantitatifs : | 25 |
| 1.4. Périmètre du projet : | 25 |
| Questions les plus courantes : | 25 |
| 2. Description graphique et ergonomique | 26 |
| 2.1. Charte graphique : | 26 |
| 3. Description fonctionnelle et technique | 27 |
| 3.1 Arborescence du de l'application : | 27 |
| 3.2. Contraintes techniques : | 28 |
| VI – Sources et inspirations | 29 |
| 1. Recherche d'informations | 29 |

| | |
|--|-----------|
| 2. Recherche sur des sites anglophones | 29 |
| VII – Conclusion | 29 |
| 1. Conclusion personnelle | 29 |
| 2. Conclusion de projet | 29 |
| Annexes | 30 |
| Wireframe | 30 |

I – Présentation

1. Présentation personnelle

Je m'appelle Jarod, j'ai 23 ans et je suis passionné de développement web depuis plusieurs années. J'ai mis un pied dedans en allant en faculté d'informatique. C'est la raison pour laquelle je suis actuellement en formation de développeur web et application mobile à La Plateforme. Aujourd'hui je compte sur mes projets professionnels et personnels pour développer cette passion et en faire mon métier.

2. Présentation du projet en Anglais

I don't think that I am wrong if I say that a lot of people want to plan their travel. What could be better than an event finder under your hands. More than an accessory, NetEvent is a tool to manage your life as a traveler. Even companies can take advantage of NetEvent.

We've built NetEvent during our educational background at La Plateforme. Our motivation is based on doing a group project with the same goals than each others :

Fully achieve multi purpose application under a year project followed by instructors.

3. Liste des compétences visées

- Maquetter une application
- Développer une interface utilisateur de type desktop
- Développer des composants d'accès aux données
- Développer la partie front-end d'une interface utilisateur web
- Développer la partie back-end d'une interface utilisateur web
- Concevoir une base de données
- Mettre en place une base de données
- Développer des composants dans le langage d'une base de données
- Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement

- Concevoir une application
- Développer des composants métier
- Construire une application organisée en couches
- Développer une application mobile
- Préparer et exécuter les plans de tests d'une application
- Préparer et exécuter le déploiement d'une application

II – Organisation

1. Fonctionnalités

| Pages | Description |
|-----------------------|--|
| Accueil | La page d'accueil intègre tous les événements non terminés, triés par catégories. |
| Création d'événement | L'application dispose d'une page avec un formulaire permettant de créer des événements. Cette page sera exclusivement réservée aux utilisateurs partenaires ou agréés. |
| Profil | Une page de profil simple et épurée disposant d'une possibilité de modification du pseudo, de la photo de profil et du mot de passe. |
| Connexion | La page de connexion est régie par un formulaire avec deux entrées : le pseudo et le mot de passe. |
| Inscription | La page d'inscription contient un formulaire avec trois entrées : le pseudo, le mot de passe et la confirmation du mot de passe. |
| Recherche d'événement | La page recherche d'événement est constituée par un formulaire avec la ville recherchée, le rayon de recherche et un arsenal de filtre pour préciser la recherche |

2. Répartition des tâches

Nous avons répartis les tâches de façon à tirer le meilleur des capacités de chacun.

Jarod :

- Création de la page d'accueil
- Affichage des événements sous forme de carrousel
- Création de l'algorithme de recherche
- Mise en place du token d'authentification à partir de l'api

Emmanuelle :

- Création de la page de connexion
- Gestion du token à travers les différentes pages de l'application
- Création de la page profil
- Mise en place de la fonctionnalité de modification du profil

Jeremy :

- Création de la page inscription
- Création du template d'affichage d'un événement
- Création de la page de recherche d'événements
- Gestion des données des événements par le biais d'un fichier json

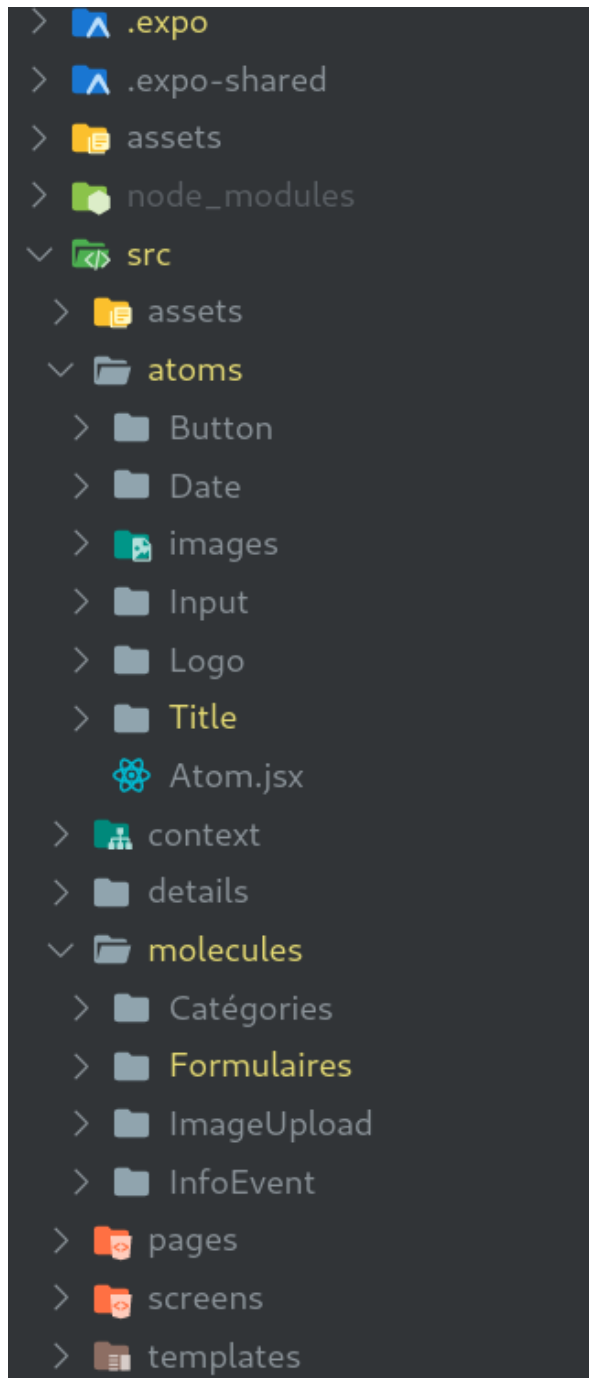
3. Outils utilisés

III – Conception

1. Base de données

2. Frontend

Développement du Front-End de l'Application



Arborescence

l'Arborescence correspond à l'architecture Atomic Design.

L'atomic design est une méthodologie composée de cinq étapes distinctes travaillant ensemble pour créer des systèmes de conception d'interface de manière plus délibérée et hiérarchique.

Les cinq étapes de la conception atomique sont :

- Atoms
- Molécules
- Organism
- Templates
- Pages

Les **Atoms** un élément HTML de base tel que des boutons, des titres, qui ne peuvent plus être décomposés en plus petit.

Les **Molécules** sont un groupement d'atomes qui fonctionnent ensemble, par exemple: un formulaire (Input, Label, Bouton). Le résultat un composant simple est utilisable de partout où on en a besoin.

Les **Organism**, sont des composants plus complexes composés soit de groupes d'atomes, soit de plusieurs molécules et peuvent même être composés d'autres organismes. C'est organism forme une fonction distinctes de l'interface, exemple le Header d'une page web.

Les **Templates**, sert de mise en page des composants précédents et articule la structure et non pas le contenu exact de la page.

Les **Pages** sont des exemples spécifiques de modèles qui montrent à quoi ressemble une interface utilisateur avec un contenu représentatif réel en place. C'est là que vous voyez tous ces composants se rassembler pour former une interface utilisateur belle et fonctionnelle. Les pages sont essentielles pour tester l'efficacité du système de conception sous-jacent. C'est au stade de la page que nous sommes en mesure d'examiner comment tous ces modèles résistent lorsque du contenu réel est appliqué au système de conception. Est-ce que tout est beau et fonctionne comme il se doit ? Si la réponse est non, nous pouvons revenir en arrière et modifier nos molécules, organismes et modèles pour mieux répondre aux besoins de notre contenu.

Atoms et Molécules:

Voici un exemple de plusieurs atoms (Bouton, Input, Logo, Title...) :

```
MyInput.jsx M X
src > atoms > Input > MyInput.jsx > ...
You, 8 seconds ago | 2 authors (jeremy and others)
1 import React from 'react';
2 import { TextInput } from 'react-native';
3
4 const MyInput = ({inputValue, placeholder, placeHoldeTextColor, inputSet, inputKeyboardType, inputStyle}) => {
5   return (
6     <TextInput placeholder={placeholder} style={inputStyle} value={inputValue} onChangeText={inputKeyboardType}
7   );
8 };
9 export default MyInput;
```

Le but de cette atom est d'afficher un Input, on peut voir que l'atom est une fonction avec des paramètres, chaque paramètres (props) correspond à une caractéristique spéciales du composant TextInput, c'est paramètre seront définis lors de l'appelle final de l'atom.

```
Atom.jsx X
src > atoms > Atom.jsx
jeremy, last month | 1 author (jeremy)
1 import MyImage from "../images/MyImage";
2 import MyTitle from "../Title/MyTitle";
3 import MyDate from "../Date/MyDate";
4 import MyLogo from "../Logo/MyLogo";
5 import MyInput from "../Input/MyInput";
6 import MyButton from "../Button/MyButton";
7
8 export {MyImage, MyTitle, MyDate, MyLogo, MyInput, MyButton}
```

Pour faciliter l'import des atoms dans les molécules j'ai créé un fichier atoms qui importe chaque atoms un par un et j'exporte les atoms ensemble, cela évite de faire plusieurs lignes pour importer plusieurs atoms.

```
Formulaires.jsx M X
src > molecules > Formulaires > Formulaires.jsx > ...
You, 27 seconds ago | 2 authors (jeremy and others)
1 import { View, Text, Button, StyleSheet, ScrollView } from 'react-native'
2 import React, {useState, useEffect} from 'react'
3 import { MyButton, MyInput, MyTitle } from "../../atoms/Atom"
4 import DateTimePicker from '@react-native-community/datetimepicker'
5 import { Picker } from "@react-native-picker/picker";
6 import * as SecureStore from "expo-secure-store";
7
```

Molécules Formulaires

On peut voir sur la ligne 3 la façon dont j'importe tous les atoms sur une ligne depuis le fichiers atoms que j'ai créé **[Voir document n°1]**.

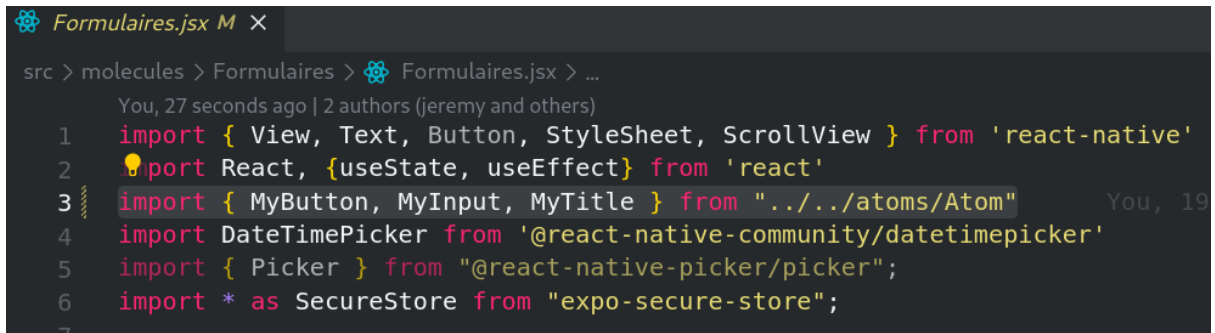
```
<MyInput
  placeholder={'Nom Event'}
  placeHolderTextColor={'white'}
  inputStyle={styles.myInput}
  inputValue={name}
  inputSet={setName}
/>

<MyInput
  placeholder={'Description'}
  placeHolderTextColor={'white'}
  inputStyle={styles.myInput}
  inputValue={content}
  inputSet={setContent}
/>
```

On peut voir que j'appelle MyInput (Atom) dans la molécule Formulaire et que je défini les paramètres que je lui ai passés ; Tous les paramètres ne sont pas forcément à définir

React Native est simple d'utilisation, cependant il ne s'utilise pas exactement comme React , car il ne manipule pas le DOM via un DOM virtuel. Il s'exécute en arrière-plan (interprète le code JavaScript écrit par les développeurs) directement sur le terminal et communique avec la plateforme native via une passerelle de sérialisation, asynchrone et par lots.

En ReactJS nous utilisons du JSX pour les écrans contrairement à React Native, où nous devrions utiliser des composants déjà créés.



```
Formulaires.jsx M X
src > molecules > Formulaires > Formulaires.jsx > ...
You, 27 seconds ago | 2 authors (jeremy and others)
1 import { View, Text, Button, StyleSheet, ScrollView } from 'react-native'
2 import React, {useState, useEffect} from 'react'
3 import { MyButton, MyInput, MyTitle } from "../../atoms/Atom"
4 import DateTimePicker from '@react-native-community/datetimepicker'
5 import { Picker } from "react-native-picker/picker";
6 import * as SecureStore from "expo-secure-store";
7
```

Reprenons cette image.

On peut voir tous les composants importés pour créer ma vue à la **ligne 1**, le principal problème de React native c'est qu'il faut ajouter de nombreuses bibliothèques pour accéder au maximum de fonctionnalités et d'éléments qui constitueront notre vue.

Par exemple :

- DateTimePicker est un composant permettant de choisir une date et une heure créés par la communauté React native, le composant de base étant déprécié.

DateTimePicker Function

```
src > molecules > Formulaires > Formulaires.jsx > Formulaires > showModeEnd
95
96  const [dateStart, setDateStart] = useState(new Date());
97  const [dateEnd, setDateEnd] = useState(new Date());
98  const [mode, setMode] = useState('date');
99  const [showStart, setShowStart] = useState(false);
100 const [showEnd, setShowEnd] = useState(false);
101 const [textDateStart, setTextDateStart] = useState('Empty');
102 const [textDateEnd, setTextDateEnd] = useState('Empty');
103
104
105  const onChangeDateStart = (event, selectedDate) => {
106    const currentDate = selectedDate || dateStart;
107    setDateStart(currentDate);
108    setShowStart(false);
109    let tempDate = new Date(currentDate);
110    let fullDate = tempDate.getDate() + '/' + (tempDate.getMonth() + 1) + '/' + tempDate.getFullYear();
111    let fullTime = 'Hours: ' + tempDate.getHours() + ' | Minutes: ' + tempDate.getMinutes();
112
113    setTextDateStart(fullDate + '\n' + fullTime);
114  }
115
116  const onChangeDateEnd = (event, selectedDate) => {
117    const currentDate = selectedDate || dateEnd;
118    setDateEnd(currentDate);
119    setShowEnd(false);
120    let tempDate = new Date(currentDate);
121    let fullDate = tempDate.getDate() + '/' + (tempDate.getMonth() + 1) + '/' + tempDate.getFullYear();
122    let fullTime = 'Hours: ' + tempDate.getHours() + ' | Minutes: ' + tempDate.getMinutes();
123
124    setTextDateEnd(fullDate + '\n' + fullTime);
125  }
126
127  const showModeStart = (currentMode) => {
128    setShowStart(true);
129    setMode(currentMode);
130  }
131  const showModeEnd = (currentMode) => {
132    setShowEnd(true);
133    setMode(currentMode);
134  }
```

Pour mon application lorsque que je voulais créer un event il fallait définir la date de début de l'événement [`setDateStart`](#) et la date de fin de l'événement [`setDateEnd`](#) ainsi que l'heure de début [`setTextDateStart`](#) et de fin [`setTextDateEnd`](#).

Date time picker nous permet comme je l'ai dit précédemment de choisir une date ou une heure pour savoir lequel il faut on définit un mode [`setMode`](#). Ensuite pour afficher la vue du composant il faut aussi définir quand on veut l'afficher ou pas [`setShowModeStart`](#) / [`setShowModeEnd`](#)

Après avoir défini tous les besoins du composant, il reste à créer la fonction qui vont afficher et changer la date lors de l'appuie sur un bouton.

Pour l'affichage:

```
const [showStart, setShowStart] = useState(false);
const [showEnd, setShowEnd] = useState(false);
```

On définit le mode par défaut à false donc ne pas afficher

```
const showModeStart = (currentMode) => {
  setShowStart(true);
  setMode(currentMode);
}
const showModeEnd = (currentMode) => {
  setShowEnd(true);
  setMode(currentMode);
}
```

On créer une fonction qui une fois appelée change l'état et définit le mode à afficher (Date ou Heure)

```
<View style={{margin:20}}>
  <Button title='Date de début' onPress={() => showModeStart('date')}></Button>
  <Button title='Heure de début' onPress={() => showModeStart('time')}></Button>
  <Text>{textDateStart}</Text>
  {showStart && <DateTimePicker
    testID='dateTimePicker'
    value={dateStart}
    mode={mode}
    is24Hour={true}
    display="default"
    onChange={onChangeDateStart}
  />}
</View>
```

Ensuite on crée deux boutons un pour pouvoir choisir la date qui va appeler la fonction `showMode` avec en paramètre le type de mode qu'il veut [`showModeStart\('date'\)`](#), cela nous affiche le bon mode de `DateTimePicker` et

on lui dit que à chaque changement de valeur on appelle la fonction `onChangeDateStart`.

```
const onChangeDateStart = (event, selectedDate) => {  
  const currentDate = selectedDate || dateStart;  
  setDateStart(currentDate);  
  setShowStart(false);  
  let tempDate = new Date(currentDate);  
  let fullDate = tempDate.getDate() + '/' + (tempDate.getMonth() + 1) + '/' + tempDate.getFullYear();  
  let fullTime = 'Hours: ' + tempDate.getHours() + ' | Minutes: ' + tempDate.getMinutes();  
  
  setTextDateStart(fullDate + '\n' + fullTime);  
}
```

La fonction récupère la date sélectionnée et change alors le Hook `dateStart` (`setDateStart`), passe le `showMode` à `false` pour ne plus afficher l'interface, et format la date et finit par renvoyer la date et l'heure dans le bon format et change le hook `textDateStart` pour ainsi pouvoir afficher la date et l'heure

```
<Text>{textDateStart}</Text>
```

- Picker => Composant permettant de créer une liste déroulante (un select en HTML) car il n'y a pas de composant existant.

Le problème est alors de devoir utiliser des composants créés par la communauté, ce qui nous fait perdre du temps dans la recherche du composant parfait.

3. Backend

3.1 Backend de l'Application

La base de données

Pour notre projet il était indispensable d'avoir une base de données, afin de sauvegarder les events et les utilisateurs. tout en ayant la possibilité d'avoir un suivi.

Comme j'avais déjà conçue dès base données avec Mysql l'année dernière je suis repartie sur la même technologie pour la sauvegarde de nos données. Ce projet m'a permis de renforcer mes connaissances sur le SQL et MySQL.

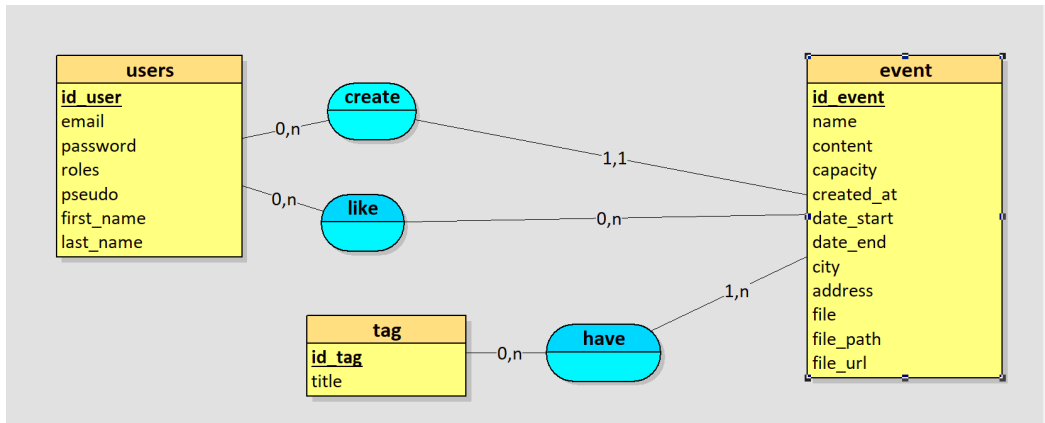
Conception de la base de données

Pour concevoir la base de données je me suis servie de la méthode Merise (MCD,MLD,MPD)

Première étape:

Modèle Conceptuel de Données

Tout d'abord il a fallu que je me demande de combien de tables j'avais besoins ensuite j'ai crée mes entités en fonction des besoins :



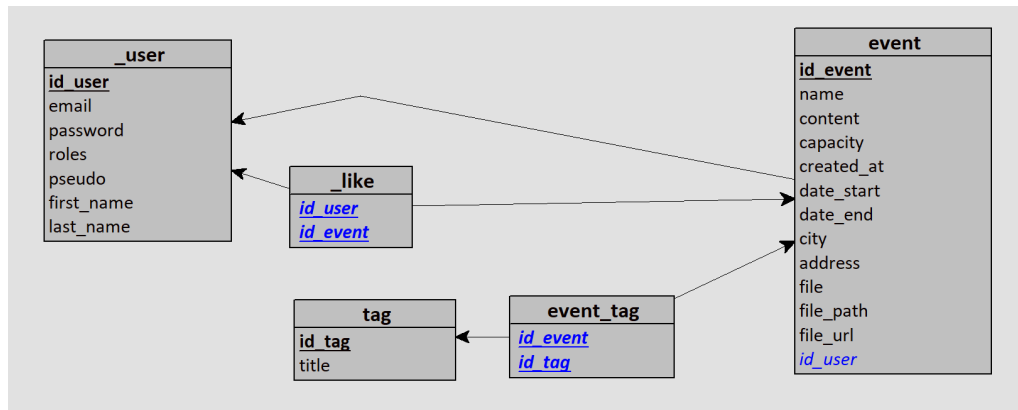
Modèle Logique de Données

Après avoir fait le MCD la suite logique est de faire un MLD, j'ai crée une clé primaire pour chaque entité (soulignage en noir) ce qui permet d'identifier sans ambiguïté chaque occurrence de l'entité.

Ensuite j'ai rempli chaque entité avec les bons attributs

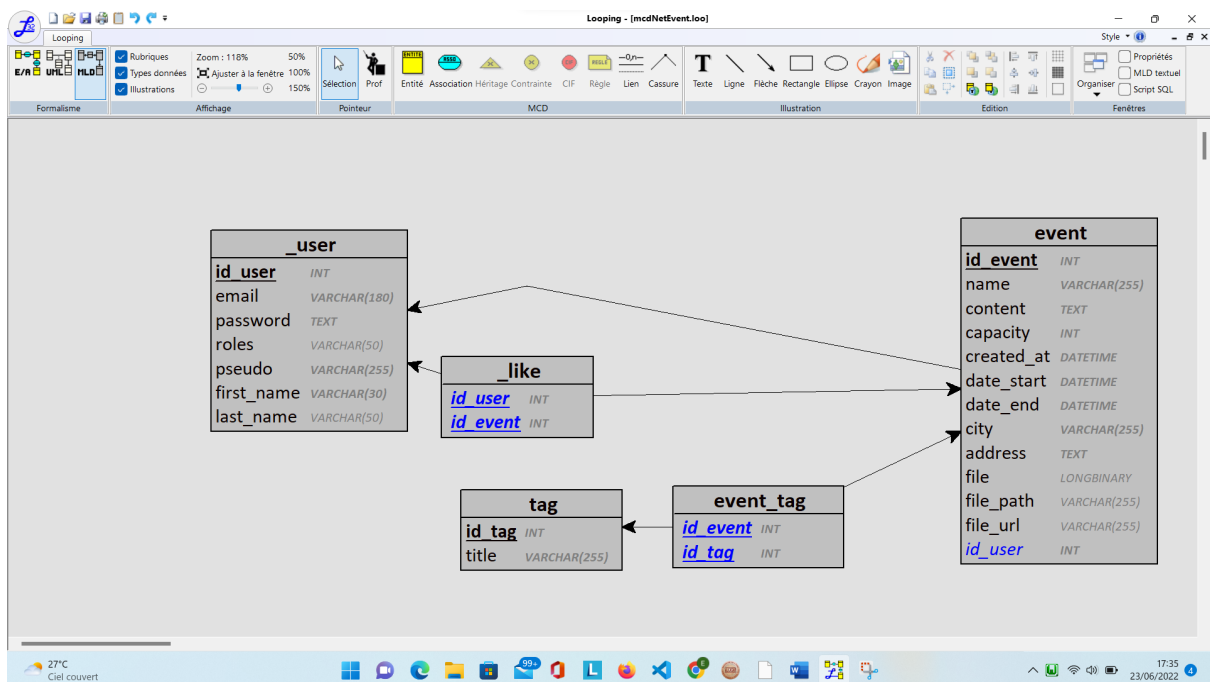
On peut aussi voir que dans le MCD certaines cardinalités sont de type N à N ce qui veut dire qu'il faut créer une table d'association, il nous en faut pour :

- Have => relation N à N entre tags et event (Création d'évènements)
- Like => relation N à N entre users et event (Like d'évènements)



Modèle Physique de Données

Pour cette dernière étape, il à fallut ajouter les types de chaque champ.

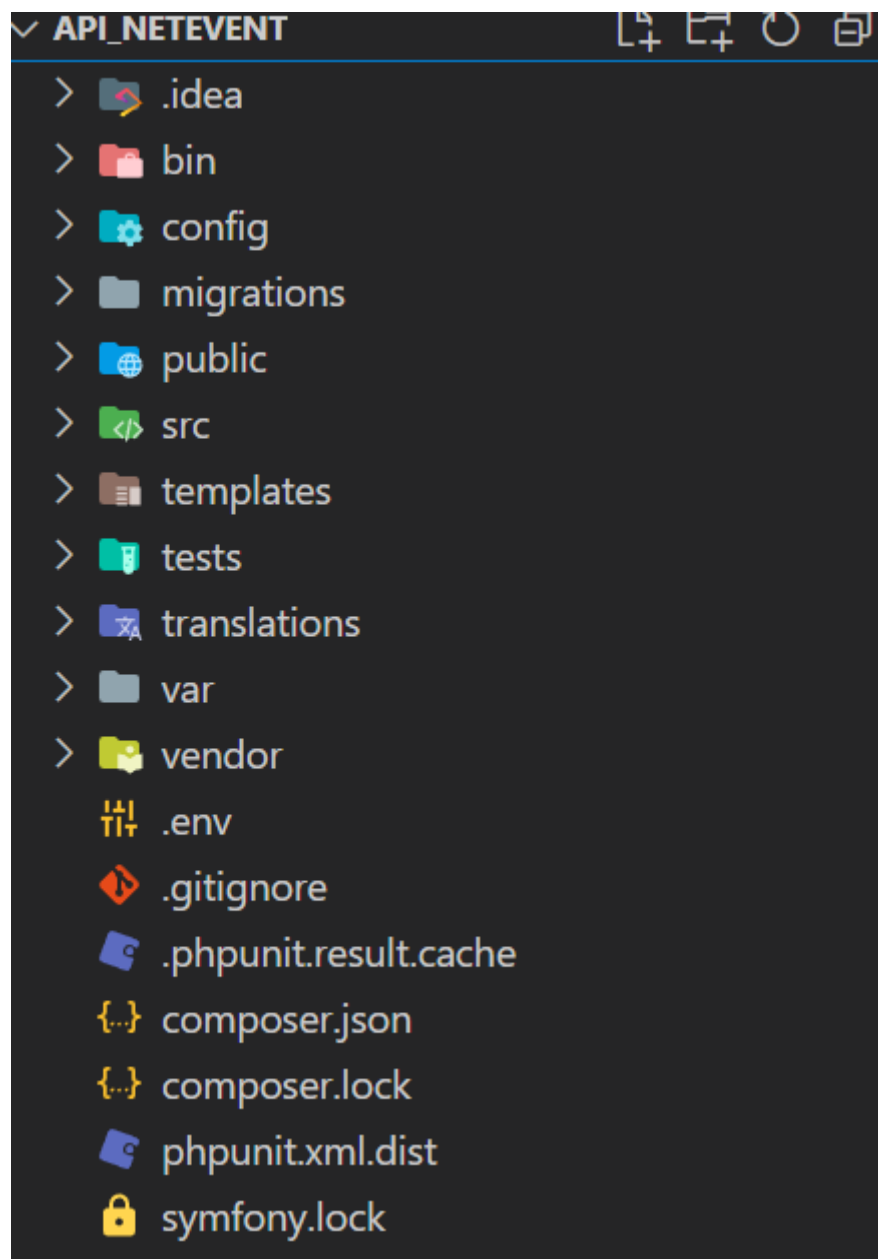


Développement du backend de l'application

Pour la création de l'api nous avons utilisé symfony avec le bundle api-platform.

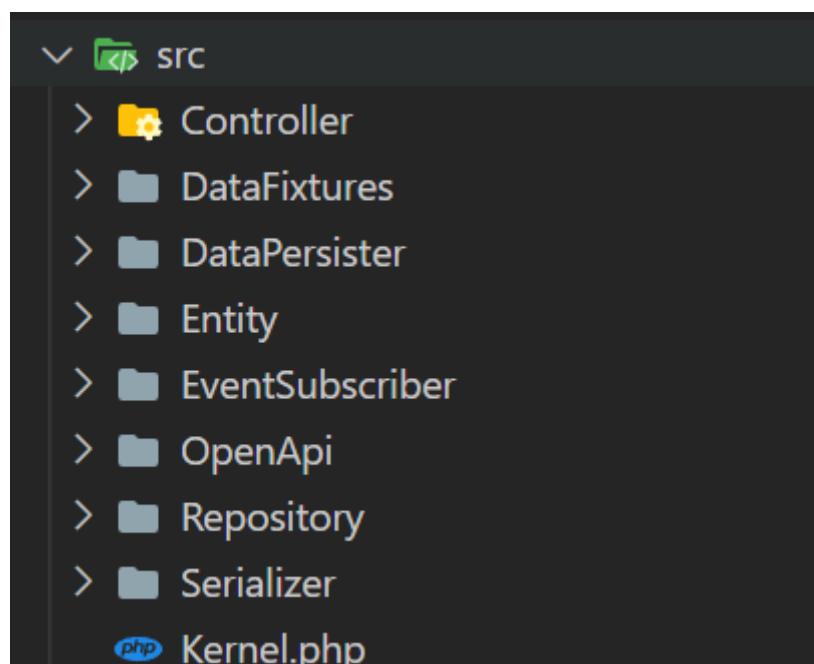
Un des avantages d'api-platform est de pouvoir aller piocher dans les bundles symfony qui nous intéressent.

Architecture global (symfony 5.4)



- bin/ : contient la console

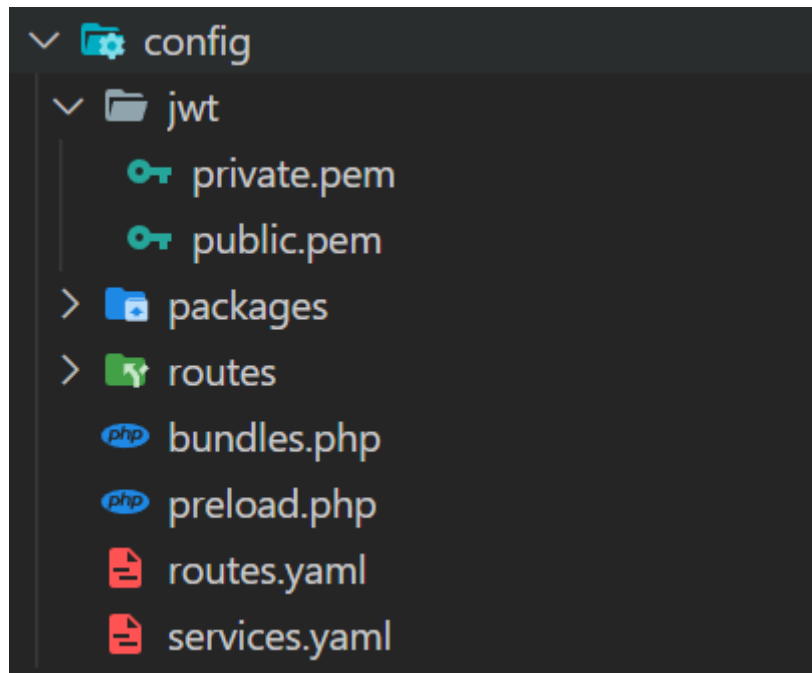
- config/ : contient les configurations des différents bundles, les routes, la sécurité et les services
- migration/ : historique des changements effectués sur la BDD
- public/ : c'est le répertoire public et accessible du site. Contient le contrôleur frontal "index.php" et les "assets" (css, js, images, ...)
- src/ : contient le projet (M et C du MVC)
- templates/ : contient les vues (V du MVC)
- tests/ : contient les tests unitaires et fonctionnels
- var/ : contient les logs, le cache
- vendor/ : contient les sources de bundles tiers et de



Dans ce répertoire se trouve toute la logique de l'application, les contrôleurs, les modèles, nos classes spécifiques, les services, ...

-
- Controller/ : Tous les contrôleurs de l'application, accessibles par des routes
- Entity/ : Les classes spécifiques pour la gestion des données et de l'API
- Repository/ : Est lié à une entité, et permet d'ajouter des requêtes spécifiques.

Configuration



Les fichiers au format yaml sont le format par défaut pour la configuration de Symfony.

- jwt/ : la clef private et public pour le token
- package/ : se trouve les répertoires dev, prod et test. Ces répertoires permettent de facilement différencier une configuration pour un environnement de développement, pour le site en ligne, ou encore pour le contexte de test

- bundles.php :

```
return [  
    Symfony\Bundle\FrameworkBundle\FrameworkBundle::class => ['all' => true],  
    Sensio\Bundle\FrameworkExtraBundle\SensioFrameworkExtraBundle::class => ['all' => true],  
    Symfony\Bundle\TwigBundle\TwigBundle::class => ['all' => true],  
    Symfony\Bundle\MonologBundle\MonologBundle::class => ['all' => true],  
    Doctrine\Bundle\DoctrineBundle\DoctrineBundle::class => ['all' => true],  
    Doctrine\Bundle\MigrationsBundle\DoctrineMigrationsBundle::class => ['all' => true],  
    Symfony\Bundle\SecurityBundle\SecurityBundle::class => ['all' => true],  
    Twig\Extra\TwigExtraBundle\TwigExtraBundle::class => ['all' => true],  
    Nelmio\CorsBundle\NelmioCorsBundle::class => ['all' => true],  
    ApiPlatform\Core\Bridge\Symfony\Bundle\ApiPlatformBundle::class => ['all' => true],  
    Lexik\Bundle\JWTAuthenticationBundle\LexikJWTAuthenticationBundle::class => ['all' => true],  
    Gesdinet\JWTRefreshTokenBundle\GesdinetJWTRefreshTokenBundle::class => ['all' => true],  
    Vich\UploaderBundle\VichUploaderBundle::class => ['all' => true],  
    Doctrine\Bundle\FixturesBundle\DoctrineFixturesBundle::class => ['dev' => true, 'test' => true],  
    Symfony\Bundle\DebugBundle\DebugBundle::class => ['dev' => true],  
    Symfony\Bundle\MakerBundle\MakerBundle::class => ['dev' => true],  
    Symfony\Bundle\WebProfilerBundle\WebProfilerBundle::class => ['dev' => true, 'test' => true],  
];
```

Focus sur le bundle API Platform

API Platform est un framework écrit en PHP et basé sur Symfony qui permet de mettre en place simplement et rapidement une API Rest . Il se base pour cela sur le système de configuration qui va permettre de transformer les modèles de notre application en ressources d'API avec les points d'entrée correspondants. Il va aussi automatiquement générer une documentation OpenAPI qui permet de décrire le fonctionnement de l'API aux personnes qui souhaiteraient l'utiliser.

Puisqu'il utilise le framework Symfony il est aussi très facile d'étendre le fonctionnement de l'outil en créant de nouveaux services ou en utilisant le système d'événements intégré.

IV - Charte graphique

1. Couleurs

Nous avons pris le soin de choisir une palette de couleurs cohérente avec la fonction de notre application et les préférences des utilisateurs en 2022. Comme vous avez pu le deviner, le design épuré et rouge et noir fait référence à une célèbre plateforme de streaming : Netflix.



#000000



#FF1616



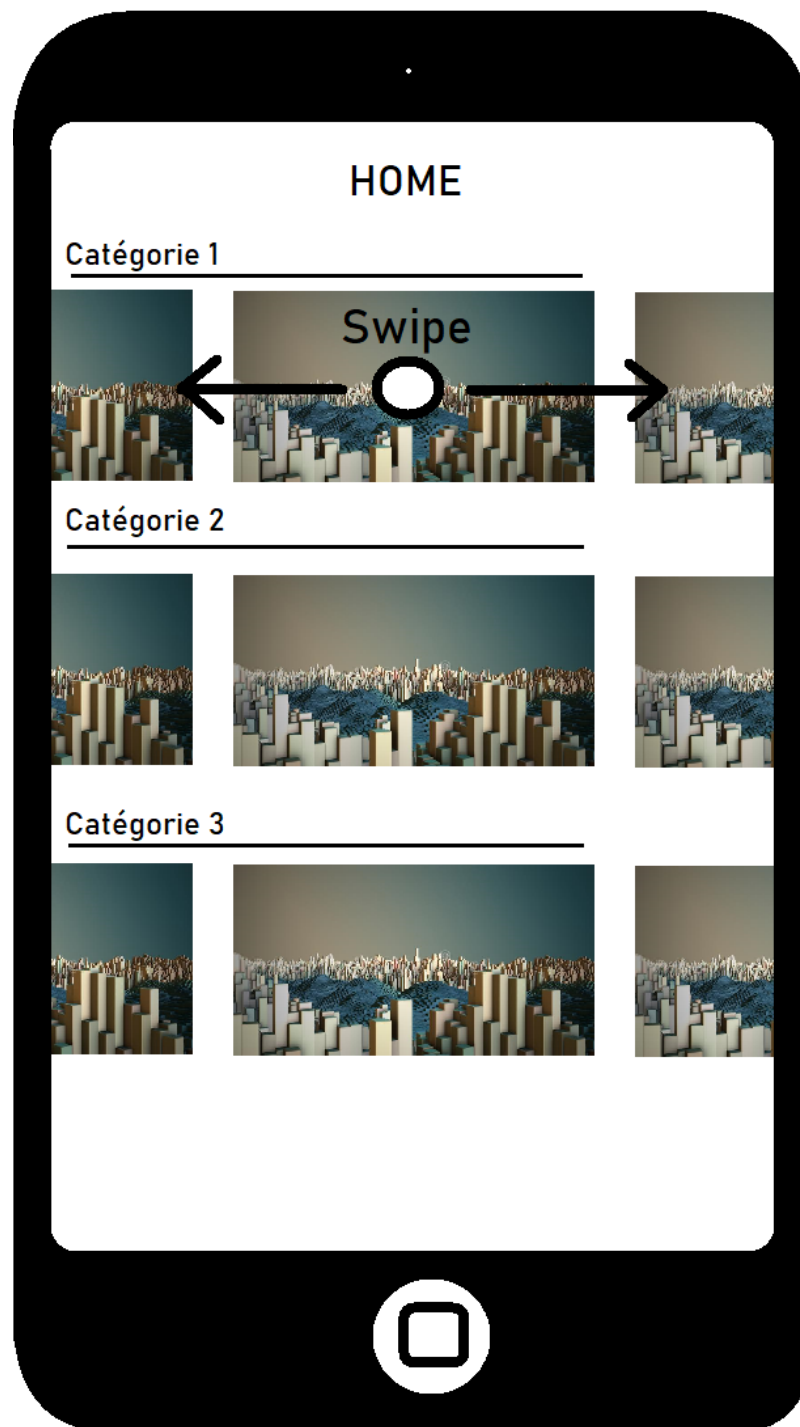
#FF5757



#FFFFFF

2. Maquette

Aux prémices de notre projet nous avons décidé d'exploiter un design défini sur un tableau blanc **[Voir Annexes]**. J'ai également conçu un wireframe simple pour bien comprendre la structure des pages de notre application



V – Cahier des charges

1. Présentation d'ensemble du projet

1.1. Les objectifs de l'application :

Comme j'ai pu le préciser dans la présentation d'ensemble, notre application a pour but de proposer un service de création d'événements pour les entreprises tierces mais elle permet aussi aux clients d'avoir un accès aux événements plus rapidement. NetEvent n'a pas seulement pour but de fluidifier cette relation entreprise/client mais elle permettra aussi d'alléger l'agenda des organisateurs et faire découvrir d'autres événements aux potentiels utilisateurs.

Nos ambitions ne s'arrêtent pas à une simple application d'événements ; Nous sommes déjà en train de penser à la conception d'une fonctionnalité qui permettra aux utilisateurs de créer des liens et fonder une communauté par la suite.

Tout projet se doit d'obtenir la confiance de ses utilisateurs. C'est pourquoi nous avons mis un poing d'honneur à ce que la sécurité soit la plus solide possible. Les données ne seront pas partagées. Elles seront seulement étudiées par nos algorithmes pour permettre aux clients de gagner du temps pendant la recherche d'événements.

1.2. La cible adressée par l'application :

Comme j'ai pu le préciser précédemment, l'application s'adresse non seulement aux particuliers mais aussi aux entreprises. Les particuliers pourront participer à des événements proposés par les entreprises partenaires mais aussi créer leurs propres événements, sous réserve de notre approbation. En effet nous ne voulons pas permettre de problèmes « d'événements fantômes » comme nous aimons les appeler.

1.3. Objectifs quantitatifs :

Le volume attendu devrait se situer autour des 500 utilisateurs journaliers dans un premier lieu. C'est un but certes plutôt ambitieux mais réaliste pour un projet de cette envergure.

La future communauté dépendra essentiellement du volume d'utilisateurs présent sur notre application. C'est pourquoi nous mettrons tout en œuvre pour proposer des fonctionnalités innovantes et transformer NetEvent en un outil quotidien.

1.4. Périmètre du projet :

Questions les plus courantes :

Votre application doit-elle être multilingue ?

Notre application devra être multilingue, pour atteindre nos objectifs nous aurons besoin d'un appui des potentiels touristes qui parcourent le territoire.

L'application devra être compatible avec quels supports ?

L'application sera compatible avec tous les supports mobiles et tablettes.

L'application doit-elle être compatible avec Apple, Android, Windows... ?

NetEvent sera disponible sur Android et Apple. Par la suite l'application pourra être compatible avec d'autres systèmes.

Quelles sont les solutions de paiement à intégrer (PayPal, Carte bancaire ...) ?

L'application sera totalement gratuite. La vie de NetEvent dépendra de nos associés et partenaires. Nous portons tous nos engagements sur une future communauté.

Y a-t-il déjà une précédente version de l'application mobile ?

Non, c'est pourquoi nous essayons au maximum de prévenir tous les potentiels problèmes qui peuvent contraindre l'utilisation de notre application.

2. Description graphique et ergonomique

2.1. Charte graphique :

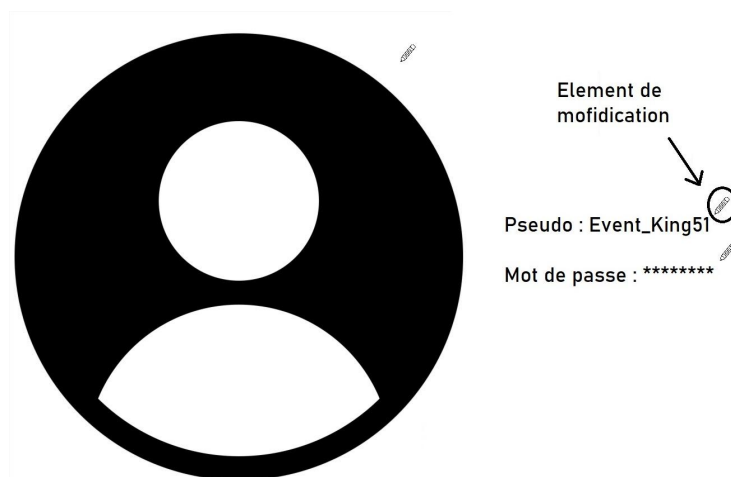
L'application se compose essentiellement de rouge et de noir. L'interface utilisateur sera constituée de formes simples pour simplifier l'utilisation au maximum. La page d'accueil sera constituée de « swipers » (carrousels) **[Voir le Document n°1]**. La page Profil sera simple également ; Néanmoins le profil est personnalisable (photo de profil, description courte, informations de l'entreprise ou du particulier etc...) **[Voir le Document n°2]**.

Notre logo comporte plusieurs déclinaisons de couleur pour habiller nos futurs documents et nos réseaux. L'UX sera régit par des swipers de façon à intégrer le plus d'événements possible sans pour autant inonder le visuel.

Document n°1

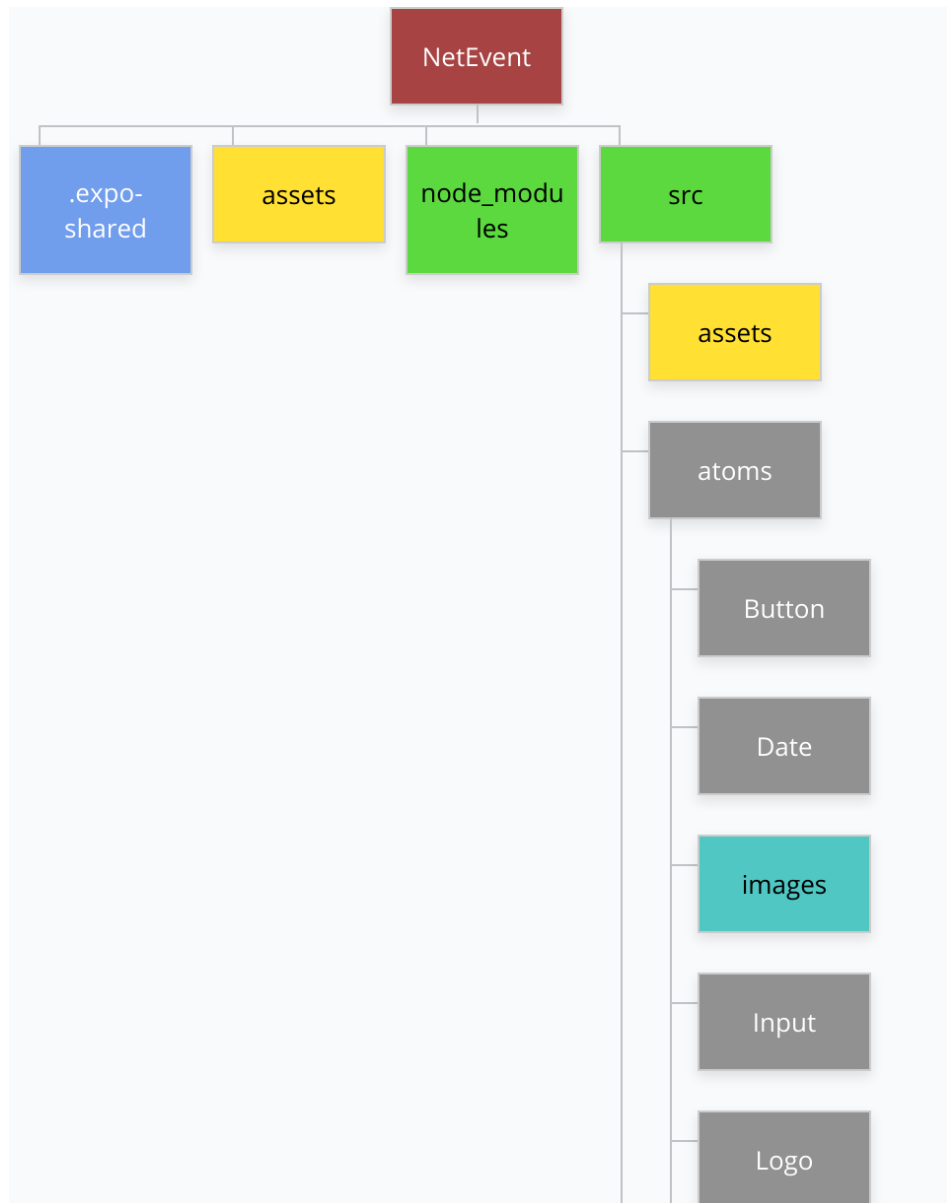


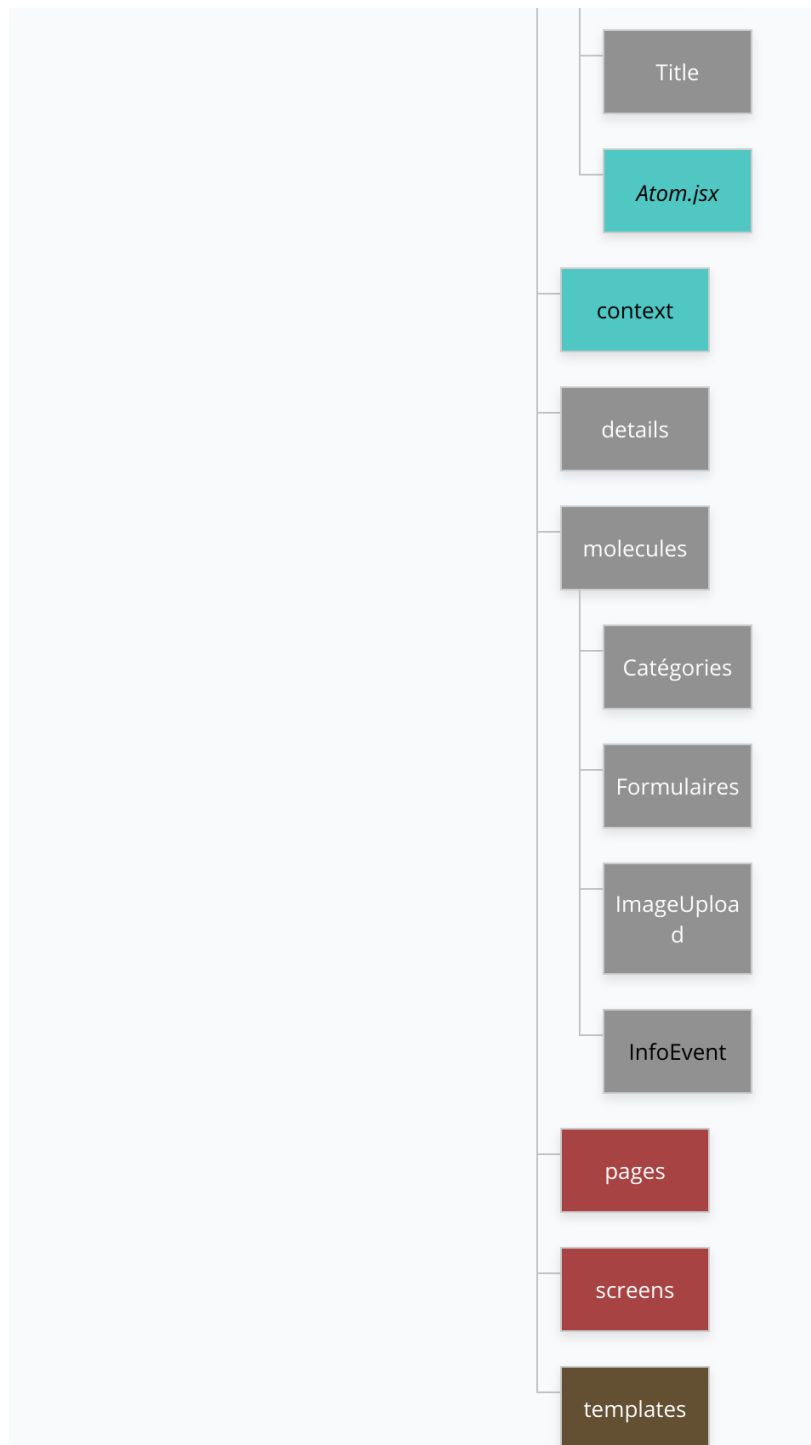
Document n°2



3. Description fonctionnelle et technique

3.1 Arborescence du de l'application :





3.2. Contraintes techniques :

- Hébergement
- Assurer la maintenance
- Intégrer des services tiers : site web, CRM...
- Quelles seront les solutions utilisées avec votre application (Frameworks, templates, etc...)

VI – Sources et inspirations

1. Recherche d'informations

Un bon site web se doit de respecter les normes W3C.



Le W3C ou World Wide Web Consortium désigne un organisme international à but non lucratif. Son rôle est de définir les standards techniques liés au web. Surtout, ils déterminent les mêmes règles pour tous les développeurs du monde. L'idée est également de favoriser l'accès au web, au plus grand nombre.

Un site web doit également respecter un design graphique moderne. Il faut donc se référer à plusieurs préférences des utilisateurs.

graphiste.com est un site qui classe des design de site web selon les préférences des utilisateurs.



En programmation, il faut apprendre à apprendre. Pour gagner du temps nous devons savoir comment rechercher des informations et où les trouver.

Personnellement je procède comme tel :



2. Recherche sur des sites anglophones

MDN Web Docs



"MDN Web Docs is an open-source, collaborative project documenting Web platform technologies, including CSS, HTML, JavaScript, and Web APIs. We also provide an extensive set of learning resources for beginning developers and students"

MDN est un répertoire qui réunit de nombreux documents pour apprendre la programmation web.

W3Schools



"We create simplified and interactive learning experiences.

Learning web development should be easy to understand and available for everyone, everywhere !

W3Schools is a school for web developers, covering all the aspects of web development"

W3Schools est un site pour les développeurs en herbe comme pour les plus expérimentés. Il réunit une quantité gargantuesque de documentation pour parfaire nos sites web.

VII – Conclusion

1. Conclusion personnelle

Grâce à ce projet j'ai pu affiner mes compétences en javascript et en React Native. J'ai également pu découvrir d'autres dispositifs mis en place pour organiser notre projet.

Au cours de la conception j'ai pu me confronter à plusieurs problématiques qui m'ont amenées à adopter des méthodes de résolutions différentes ; tout en organisant le travail de façon à répartir les tâches efficacement dans toute l'équipe.

2. Conclusion de projet

Les débuts ont été compliqués, la mise en place d'un projet pour une équipe qui n'a jamais eu affaire à ce genre de conception nous a donné du fil à retordre.

La clé pour gagner du temps pour les parties les plus importantes sont :

- Une organisation pointilleuse
- Des réunions régulières avec toute l'équipe
- Une bonne méthode de recherche d'information

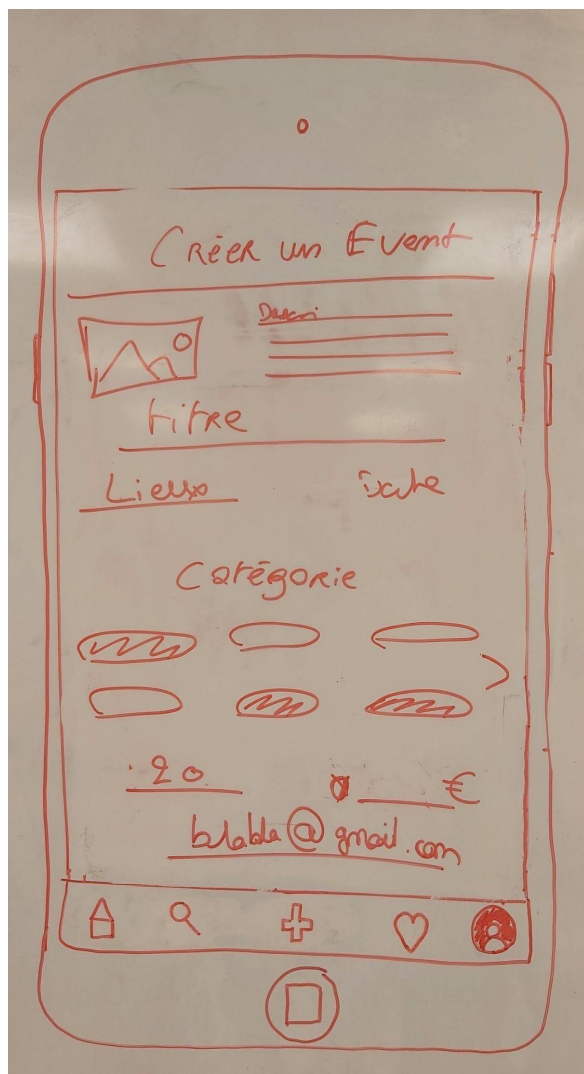
Finalement, la réussite d'un projet comme celui-ci réside dans la cohésion, une organisation solide et une base de connaissance suffisante.

Annexes

Wireframe

Vous retrouverez ici les maquettes que nous avons réalisées sur tableau blanc.

Page de création d'événement



Page d'accueil

