

Documentation du projet groupe hôtelier : Valery MBELE

A. Spécifications techniques

Server :

Ubuntu 20.14
LAMP
PHP7.4
Docker
Postgres
MySql

Frontend :

React
Material ui
Axios
React-hook-form
React-query
Js Cookie

Backend :

Symfony
Api-Platform
Express
Sequelize
Multer
Jwt
Nodemon

Utilitaires :

Figma
Gitmind
Trello

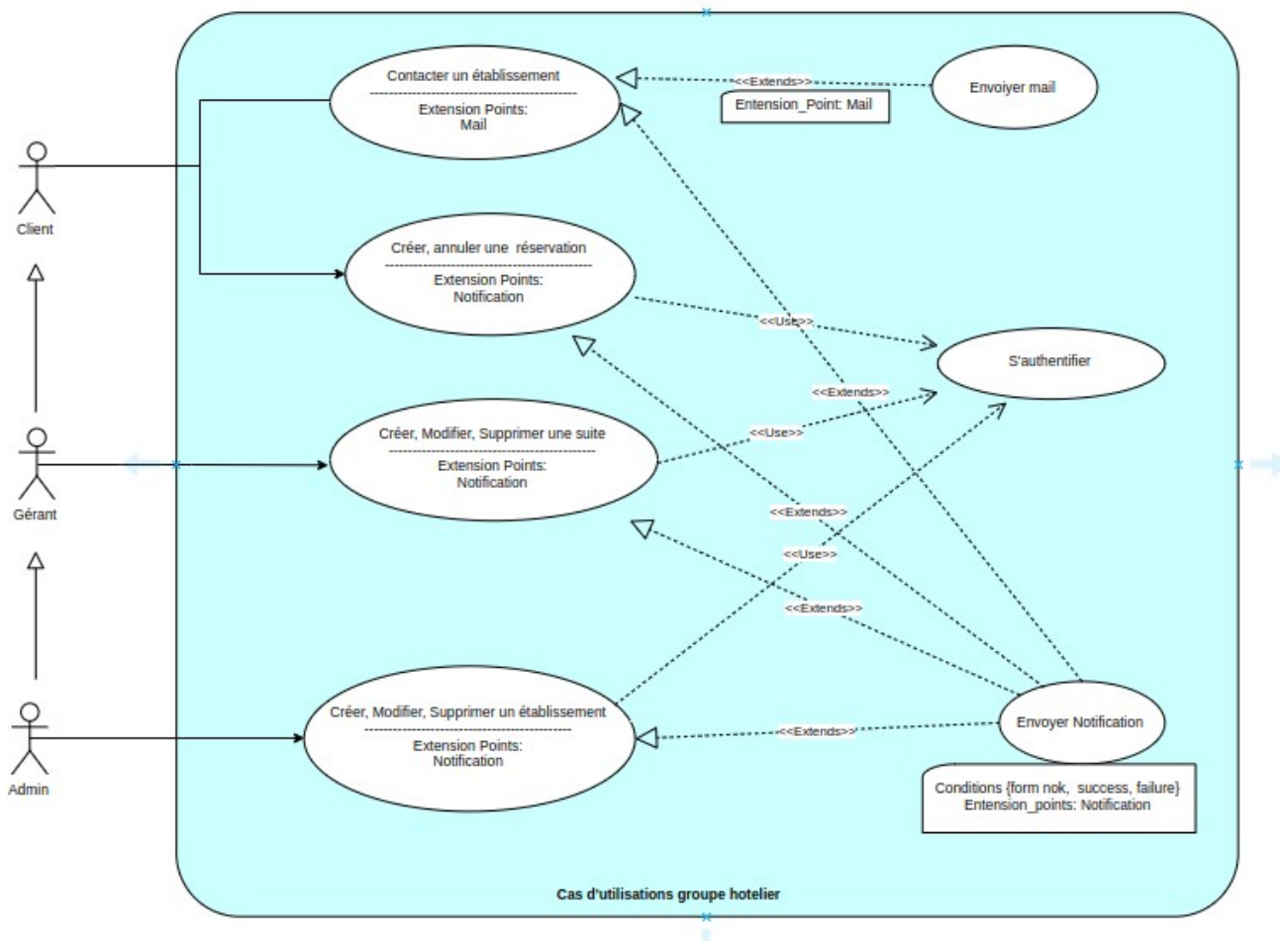
Code :

VS Code
Eslint
Prettier

CI :

github actions
github

B. Diagramme de cas d'utilisation

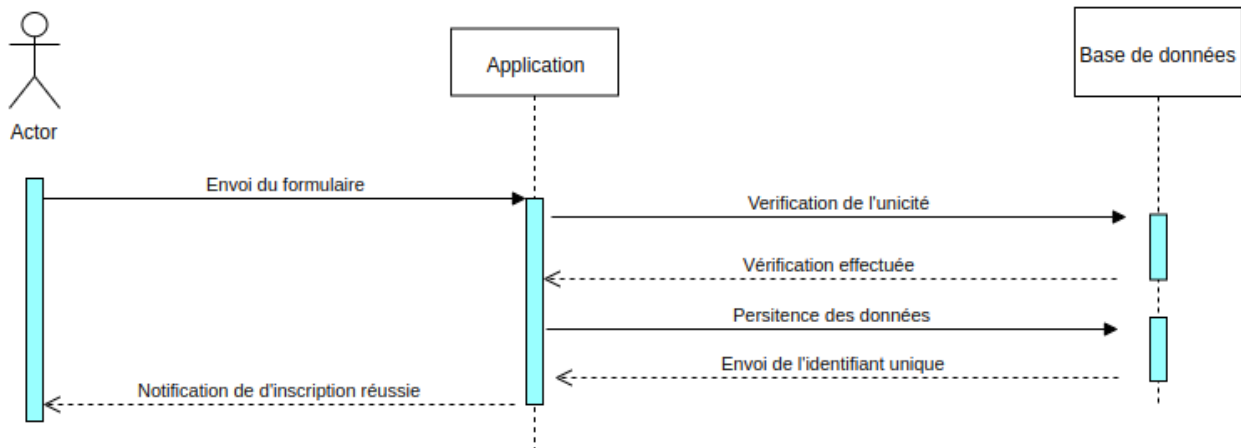


Dans ce diagramme , j'ai compris que l'authentification était primordiale. Mais aussi qu'il fallait avoir un système de notification fiable pour souvent informer l'utilisateur de ce qui se passe .

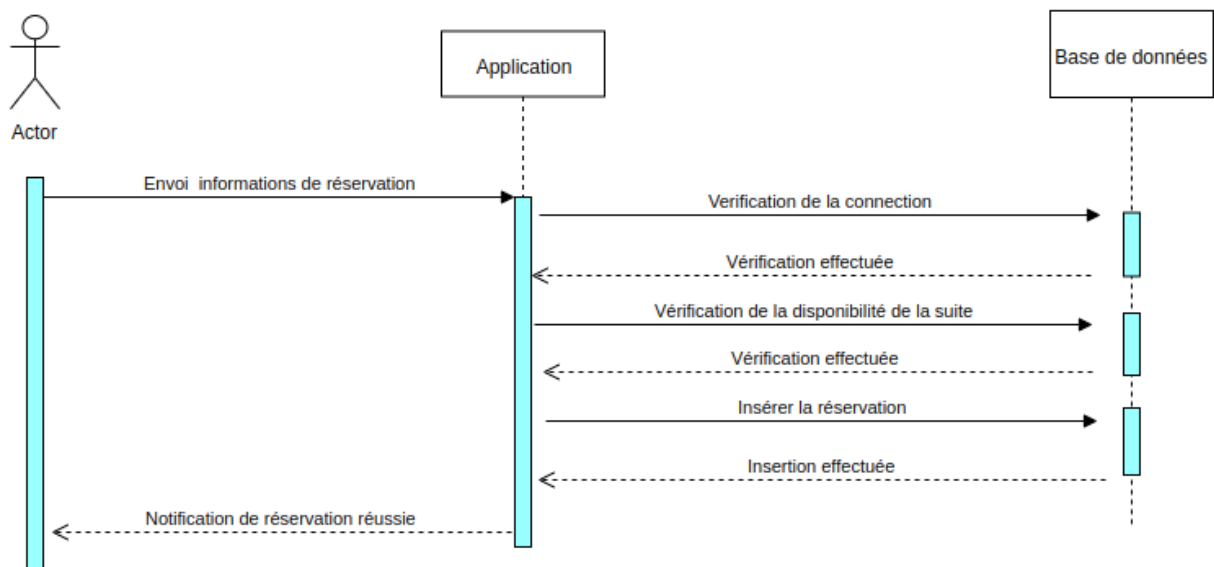
L'administrateur occupe un rôle prépondérant un rôle clé.

C. Diagrammes de séquence .

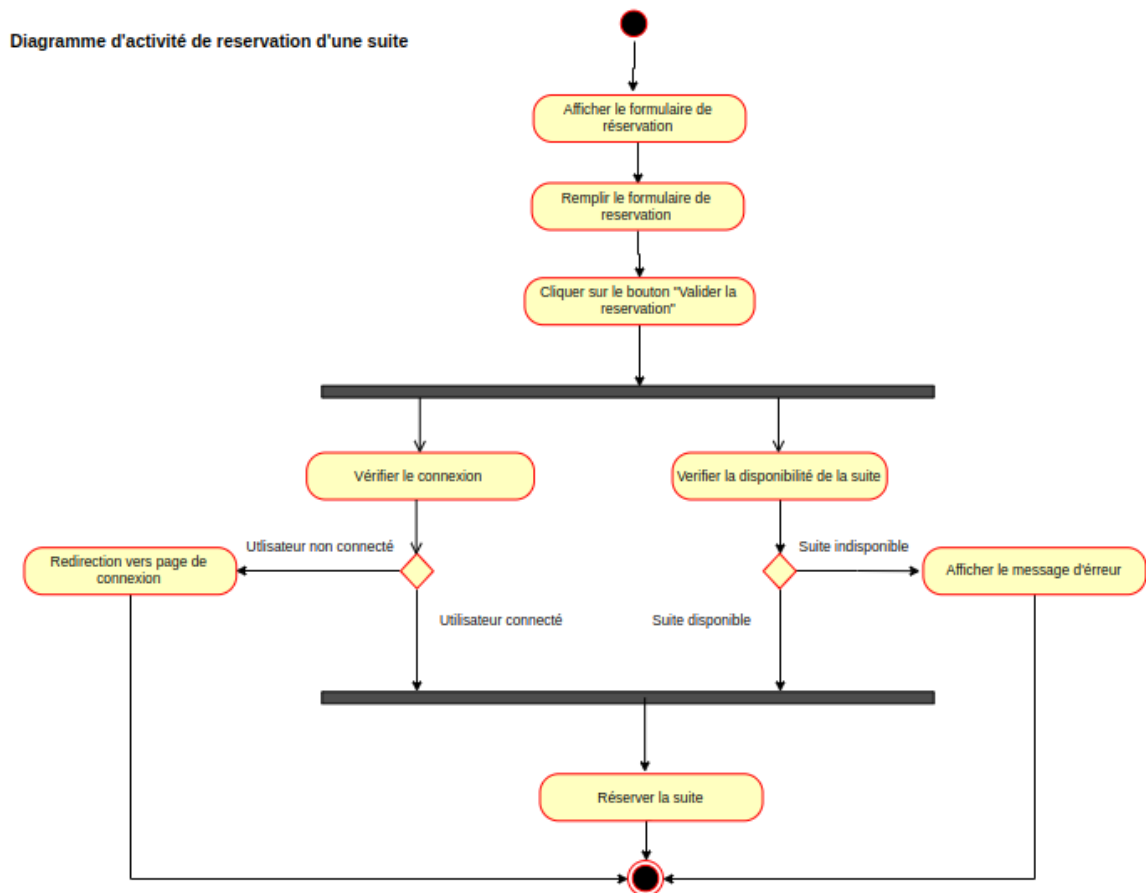
C1. Séquence authentification : cas réussi



C2. Séquence réservation réussie

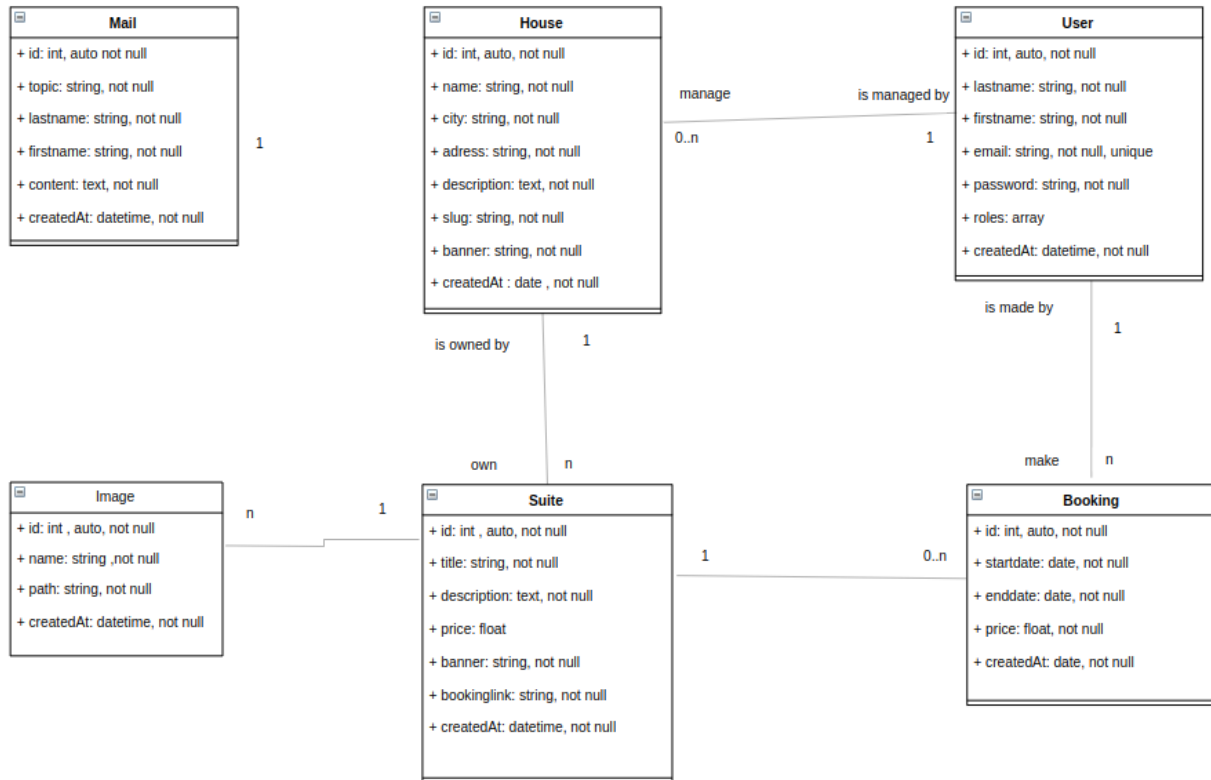


D. Diagramme d'activité de réservation :



Ce schéma est assez clair , et ne laisse pas la place à l'improvisation pendant la phase de développement.

E. Diagramme de classe



Les trois problématiques qui se sont posées sur le modèle de données étaient :

- Les gérants doivent t-ils être une entité à part ? J'ai opté pour le non , car ils ont exactement les mêmes attributs qu'un user, la différence se joue sur le fait qu'ils soient liés ou non à l'établissement
- Faut-il faire une relation n to n entre la Suite et le User ? J'ai opté pour le non . Car dans la pratique , le n to n passe souvent par une table intermédiaire
- Faut-il gérer les images directement comme un attribut des entités qui les utilisent ? J'ai encore dit non , car je ne suis pas sur que ce soient une chose simple que de faire des recherches sql dans un champs qui est un tableau.

E. Reformulation des users stories

US1. Gérer les établissements

US1.1 Gérer les établissements

Titre : En tant qu'administrateur , je veux créer, modifier ou supprimer un établissement pour mettre à jour les informations

La page ne doit pas se recharger

Critères d'acceptation :

scénario 1 : Tentative de modification avec un compte valide

Étant donné que je possède un compte utilisateur (Robert)

Lorsque j'accède page de gestion des établissements

Et que je clique sur créer un établissement ,

Et que je remplis le formulaire

Et que je clique sur le bouton créer un formulaire

Alors j'ai une notification que l'établissement a été crée,

Et je suis redirigé vers la liste des établissement

US1.2 Gérer les gérants

Titre : En tant qu'administrateur , je veux accéder au profil d'un utilisateur pour pouvoir le rendre ou non gérant d'un établissement

US1.3 En tant qu'administrateur, je veux accéder au profil d'un gérant d'établissement afin de modifier ses informations

US2. Gérer les suites

US2.1

Titre : En tant que gérant , je dois accéder à la gestion de l'établissement pour créer , modifier , supprimer une suite

Règles métier :

Le même formulaire sera utilisé pour la création et la modification

La validation de la suppression se fait par une modale et ne recharge pas la page

Le gérant ne peut pas accéder à la page de gestion des autres établissements

US3. Découvrir le catalogue des établissements

US31 Découvrir la liste des établissements

Titre : En tant que Visiteur , je souhaite consulter la liste des établissement ou rechercher un établissement afin de faire mon choix

US32. Découvrir les suites d'un établissements

Titre : En tant que Visiteur , je souhaite consulter la page d'un établissement afin de découvrir ce qu'elle offre : suites , photos , description.

Règle : La page est présentée sous forme de liste

US4. Réserver une suite

Titre : En tant que Visiteur ou Client , je souhaite réserver une suite pour mon séjour

Règle : La réservation se fait sur une page dédiée

On doit pouvoir savoir si la suite est disponible sans rechargement de page

US5. Voir les réservations

US51. Voir la liste des réservations

Titre : En tant que Client , je dois pouvoir accéder à ma liste de réservation afin de m'organiser

US52. Annuler une réservation

Titre : En tant que Client , je souhaite annuler une réservation

Règle : annulation uniquement 3 jours avant au plus tard

US6. Accélérer la réservation d'une suite

US61. En tant que Visiteur ou Client , je dois accéder à la page de réservation depuis la page d'accueil d'un établissement .

Règle : Le remplissage des informations de la suite dans la page de réservation est automatique

US7. Contacter le groupe hôtelier

Titre : En tant que Visiteur ou Client , je veux contacter l'administrateur par mail via un formulaire de contact afin de commander un service non inclus pour le jour J

Règles :

Le nom , prénom , email du client se saisisent automatiquement

Le mail sera envoyé à l'administrateur

F. Afin d'être au plus près du besoin de nos utilisateurs , j'ai défini 4 personas qui pourraient être assez représentatifs de notre cas .

Elle adore voyager et découvrir des mondes qu'elle n'a pas eu le temps d'explorer quand elle était active. Car sa situation financière le permet.

Elle est très attachée aux valeurs écologiques .

Elle adore aller sur les réseaux sociaux via son téléphone quand elle se repose dans un cadre naturel et paradisiaque.

Valery , 50 ans , Visiteur.

C'est un haut cadre de la finance qui adore le luxe . Il a 3 enfants avec lesquels il adore partager des destinations de rêve. Il est néanmoins très exigeant , mais se laisse facilement tenter quand ses copains lui parlent de suite luxueuse et écologiques.

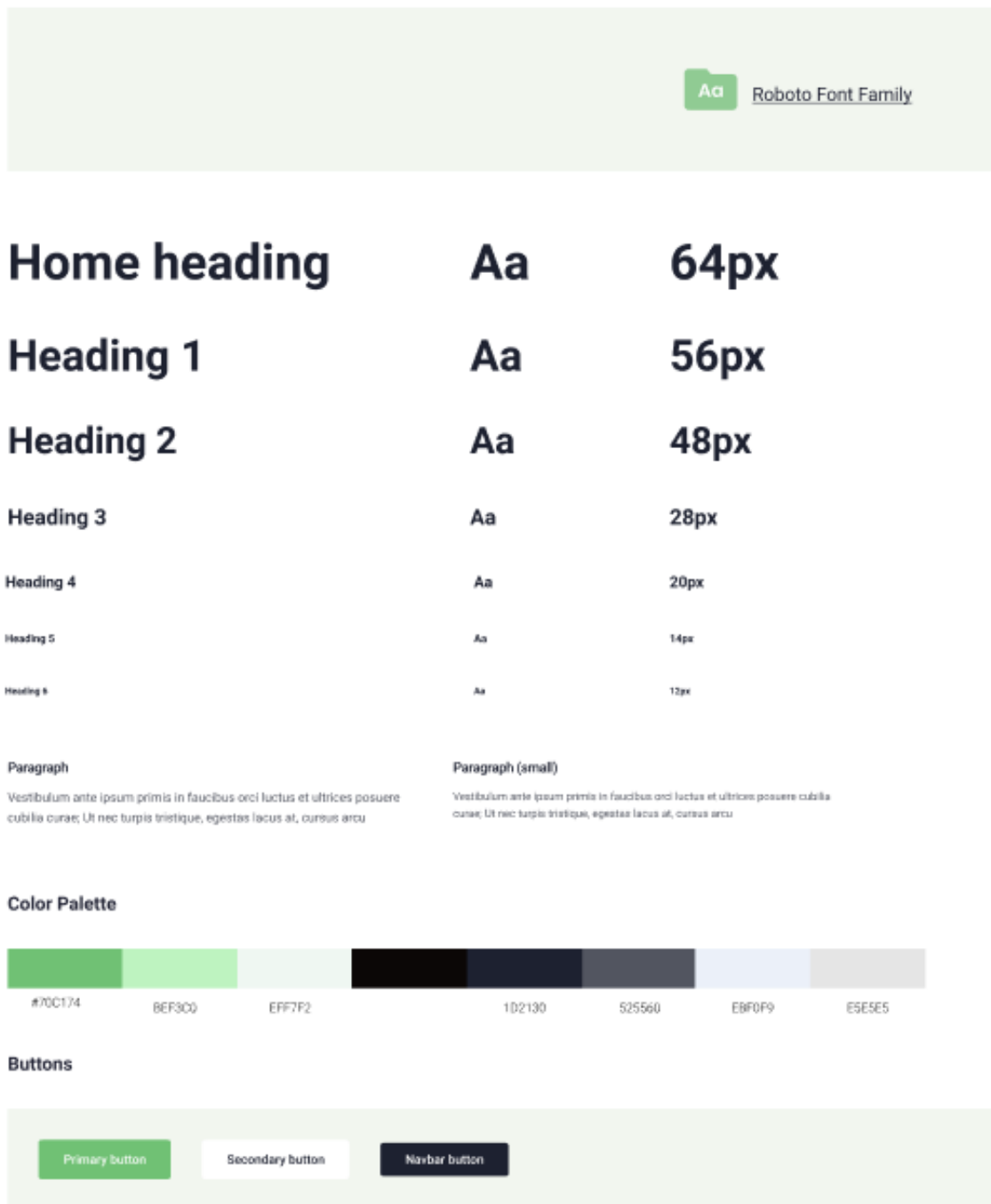
Daniel, 43 ans , gérant d'hôtel.

Daniel n'aime pas du tout passer son temps au téléphone. Ce qu'il préfère avant tout , c'est le contact humain et la vie réelle . Il est tout le temps pressé car les clients n'attendent pas.

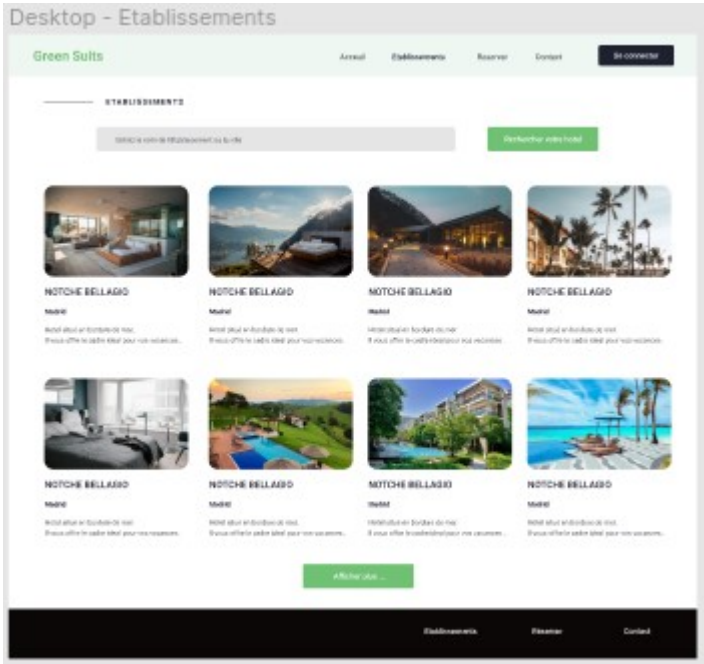
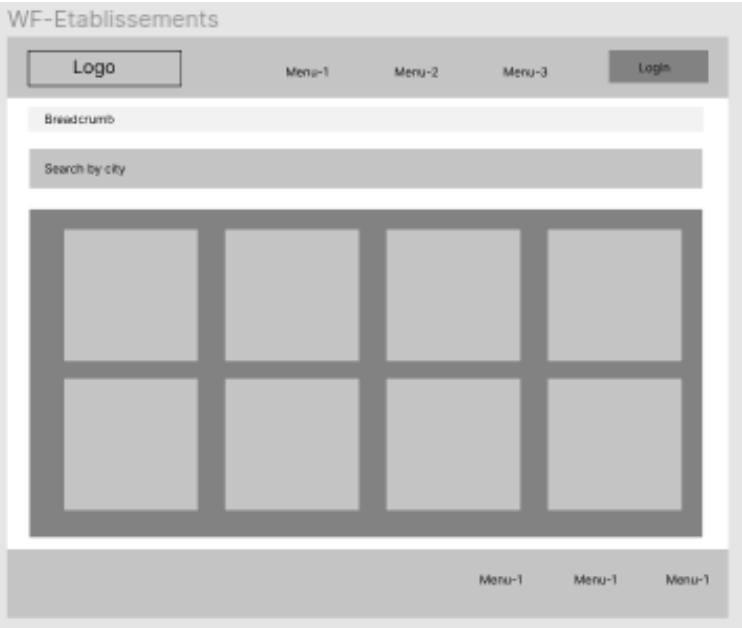
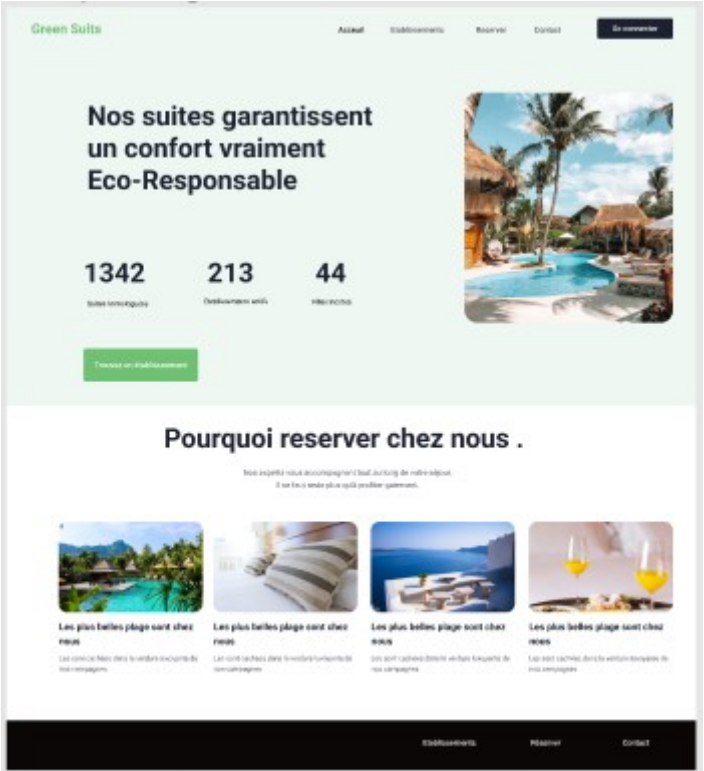
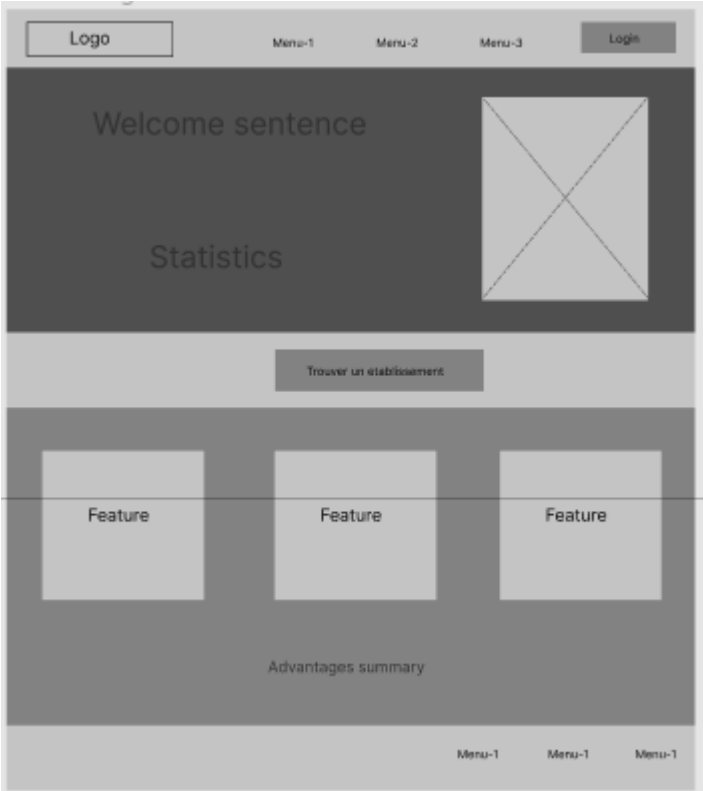
Anne , 36 ans est une mère de famille posée. Elle est déjà administratrice dans son groupe hôtelier , et a attendu avec hâte que la numérisation arrive . Elle a le sens du détail.

L'application va donc s'approcher d'un thème un peu écologique

G .Charte graphique du site :



Les couleurs restent assez épurées et légère.



Logo

Menu-1

Menu-2

Menu-3

Logos

Breadcrumb

Hotel Relais de la reine

114 Place de la Reine Marie, Marseille, France

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

Revenir vers sur booking.com

Choisissez votre suite

Suite de Luxe vue sur Mer



The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

340 €

Reserver

Suite de Luxe vue sur Mer



The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

340 €

Reserver

Menu-1

Menu-1

Menu-1

Desktop - Creation etablissement

Green Suits

Accueil

Etablissements

Revenir

Contact

Mon Compte

Se deconnecter

NON COMPTE / CREER UN ETABLISSEMENT

Création / Modif établissement

Nom

Ville

Description

Type your Message

Valider

Etablissements

Revenir

Contact

Desktop - Etablissement

Green Suits

Accueil

Etablissements

Revenir

Contact

Se connecter

ETABLISSEMENTS / HOTEL REALAIS DE LA REINE

HOTEL RELAIS DE LA REINE

114 place de la Reine Marie

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

The standard chunk of Lorem Ipsum used since the 1500s is reproduced below for those interested. Sections 1.10.32 and 1.10.33 from "de Finibus Bonorum et Malorum" by Cicero are also reproduced in their exact original form, accompanied by English versions from the 1914 translation by H. Rackham.

Choisissez votre suite

Sagittarius A



There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.

246 €

Reserver maintenant

Sagittarius A



There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.

246 €

Reserver maintenant

Sagittarius A



There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.

There are many variations of passages of Lorem Ipsum available, but the majority have suffered alteration in some form, by injected humour, or randomised words which don't look even slightly believable.

246 €

Reserver maintenant

Etablissements

Revenir

Contact

MON COMPTE / MES RESERVATIONS

Toutes mes réservations



Camping Paradis

Nancy
Suite la Fayette
Jeu 21 Mai - Sam 23 Mai 2020
250 €

Annuler



Camping Paradis

Nancy
Suite la Fayette
Jeu 21 Mai - Sam 23 Mai 2020
250 €

Annuler



Camping Paradis

Nancy
Suite la Fayette
Jeu 21 Mai - Sam 23 Mai 2020
250 €

Annuler



Camping Paradis

Nancy
Suite la Fayette
Jeu 21 Mai - Sam 23 Mai 2020
250 €

Annuler



Camping Paradis

Nancy
Suite la Fayette
Jeu 21 Mai - Sam 23 Mai 2020
250 €

Annuler



Camping Paradis

Nancy
Suite la Fayette
Jeu 21 Mai - Sam 23 Mai 2020
250 €

Annuler

WF-MonCompte-MesReservation

Logo

Menu-1

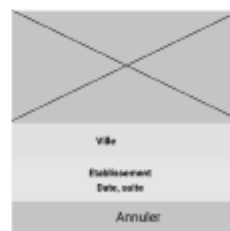
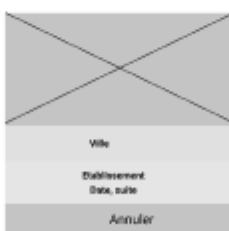
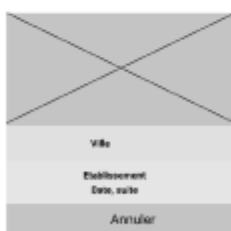
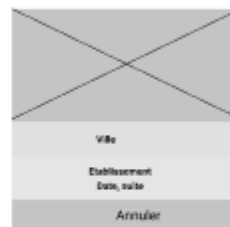
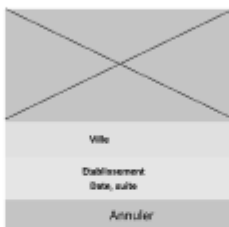
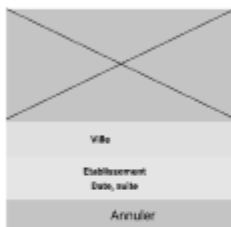
Menu-2

Menu-3

Login

Breadcrumb

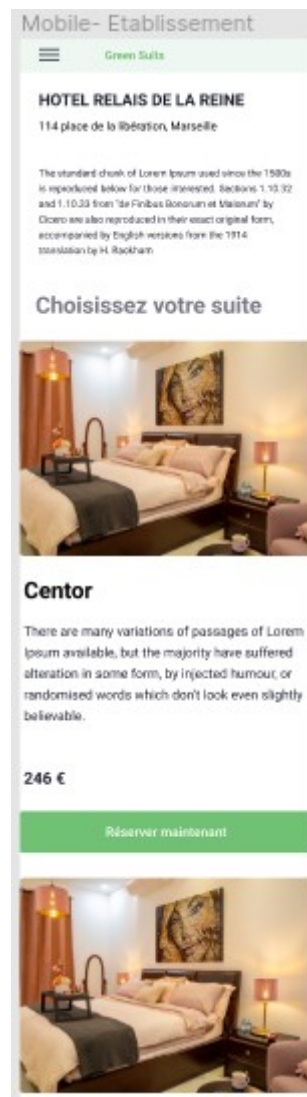
Title



Menu-1

Menu-1

Menu-1



Synthèse :

Pour un projet de ce genre , j'ai trouvé que le couplage Symfony Réact était adapté avec de pour donner du dynamisme à ces couleurs coté front , et de la puissance en backend . C'est la raison pour laquelle j'ai opté pour ApiPlatform/Reactjs.

De plus utiliser docker dans une application symfony facilite grandement la tâche . Le gain de temps est phénoménal .

En ce qui concerne postgres et mysql , la différence était faible dans ce cadre. Notamment aussi parce que , si besoin , les recettes sql peuvent se faire directement sur symfony comme sur Express.

Conclusion :

Le fait d'avoir terminé avec Express js en backend m'a permis de réellement avoir sur un même projet une comparaison de deux frameworks , de deux langages . Le dénominateur commun reste le langage de base de données.

Je trouve aussi que c'est vraiment difficile de faire un projet complet sans mettre en place des tests unitaires et fonctionnel , car les régressions de code sont vraiment source de beaucoup de perte de temps. Sur la partie symfony, je les ai mis en place pour la partie Entités.

Liens utiles :

repository projet sur symfony : <https://github.com/zoutigo/hotel-group.git>

repository projet sur express : <https://github.com/zoutigo/hotels.git>

Trello : <https://trello.com/b/F6N7qDcN/ecf>

GitMing : <https://gitmind.com/app/diagrams/dir/dc689ccaba4774a621b229a0fa700aa0>

Extraits de code importants

1. Backend

Le backend sera fait avec Express, Sequelize , Postgres sous Nodejs

1.1 Sequelize

1.1.1 Setup

Pour installer la dépendance sequelize avec le driver postgres, je lance la commandes suivantes :

```
npm install --save sequelize pg pg-hstore
```

Ensuite , puisque je vais utiliser le sytème de migration de sequelize en CLI (command line interface) en développement, j'installe la dépendance avec la commande

```
npm install sequelize-cli --save-dev
```

Les dépendances étant prêtes , je peux désormais initialiser mon projet.
A la racine de mon projet , je joue la commande

```
npx sequelize-cli init
```

Cette commande va créer les dossiers migrations, models , seeders , ainsi qu'un fichier config pour la base de donnée.

Afin de mieux structurer mon projet , je déplace tous ces éléments dans un dossier nommé `backend` à la racine de mon projet et je le structure. Puis je crée un fichier .sequelizerc à la racine de mon projet .



Le fichier de configuration config.js se trouve en annexe.

Je m'assure que le service postgresql est démarré sur ubuntu avec la commande

```
sudo systemctl start postgresql
```

Je crée ma base de données de développement avec la commande

```
npx sequelize-cli db:create
```

Je peux vérifier que ma base est bien créée , sur un terminal , je me connecte avec mon user `postgres` , je rentre les commandes suivantes :


```
> sudo i -u postgres
> psql
> \l
```

hotels_devs-# \l

List of databases					
Name	Owner	Encoding	Collate	Ctype	Access privileges
augustin	postgres	UTF8	fr_FR.UTF-8	fr_FR.UTF-8	
augustin-dev-db	postgres	UTF8	fr_FR.UTF-8	fr_FR.UTF-8	
dvdrental	postgres	UTF8	fr_FR.UTF-8	fr_FR.UTF-8	
hotels_dev	postgres	UTF8	fr_FR.UTF-8	fr_FR.UTF-8	
hotels_devs	postgres	UTF8	fr_FR.UTF-8	fr_FR.UTF-8	
postgres	postgres	UTF8	fr_FR.UTF-8	fr_FR.UTF-8	
sequelize-tutorial	postgres	UTF8	fr_FR.UTF-8	fr_FR.UTF-8	
template0	postgres	UTF8	fr_FR.UTF-8	fr_FR.UTF-8	=c/postgres + postgres=Ctc/postgres
template1	postgres	UTF8	fr_FR.UTF-8	fr_FR.UTF-8	=c/postgres + postgres=Ctc/postgres

(9 rows)

Je vois donc que ma base de données `hotels_dev` a bien été créée

1.1.2 Création des modèles de données

Pour créer mon modèle

```
npx sequelize-cli model:generate --name Example --attributes
firstName:string,lastName:string,email:string
```

Cette commande va générer deux fichiers que je vais personnaliser : le modèle , et la migration.

1.1.2.1 Migration

1.1.2.1.a Migration simple

```
module.exports = {
  async up(queryInterface, Sequelize) {
    const transaction = await queryInterface.sequelize.transaction()
    try {
      await queryInterface.sequelize.query(
        'CREATE EXTENSION IF NOT EXISTS "uuid-osspp";'
      )
      await queryInterface.createTable(
        'users',
        {
          id: {
            allowNull: false,
            autoIncrement: true,
            primaryKey: true,
            type: Sequelize.INTEGER,
          },
          uuid: {
            type: Sequelize.UUID,
            defaultValue: Sequelize.literal('uuid_generate_v4()'),
          },
          lastname: {
            type: Sequelize.STRING(30),
            allowNull: false,
            validate: {
              notNull: {
                msg: 'le prénom est obligatoire',
              },
              len: {
                args: [2, 30],
                msg: 'le prénom doit avoir entre 2 et 30 caractères',
              },
            },
          },
          firstname: {
            type: Sequelize.STRING(30),
            allowNull: false,
            validate: {
              notNull: {
                msg: 'le nom est obligatoire',
              },
              len: {
                args: [2, 30],
                msg: 'le nom doit avoir entre 5 et 30 caractères',
              },
            },
          },
        },
      ),
    } catch (error) {
      await transaction.rollback()
      throw error
    }
    await transaction.commit()
  },
  async down(queryInterface, Sequelize) {
    await queryInterface.dropTable('users')
  }
}
```

Dans ce fichier qui crée un transaction sql j'ai,

- rajouté une requête qui met en place l'uuid, car c'est avec uuid que je vais souvent traiter .
- défini ma clé primaire sur l'id et lui avoir mis l'auto incrément , ainsi que le type INTEGER de postgres,
- défini le champ uuid
- en enfin pour tous les autres champs , j'ai mis les validations de données que je souhaitais.

Ce fichier est le reflet de ce qui va se passer dans ma base de donnée dorénavant.

Je peux donc jouer ma migration avec la commande :

```
npx sequelize-cli db:migrate
```

Ceci va exécuter la migration et créer un table dans ma base de donnée.

Je procède donc de la même manière pour toutes les autres tables .

1.1.2.1.b Migration relation

Pour une relation entre User et House , je crée la migration suivante :

```
module.exports = {
  async up(queryInterface, Sequelize) {
    const transaction = await queryInterface.sequelize.transaction()
    try {
      await queryInterface.addColumn(
        'houses',
        'userId',
        {
          type: Sequelize.DataTypes.INTEGER,
          allowNull: true,
          references: {
            model: 'users',
            key: 'id',
            onUpdate: 'CASCADE',
            onDelete: 'SET NULL',
          },
        },
        { transaction }
      )
      await transaction.commit()
    } catch (error) {
      await transaction.rollback()
      throw error
    }
  },

  async down(queryInterface, Sequelize) {
    const transaction = await queryInterface.sequelize.transaction()
    try {
      await queryInterface.removeColumn('houses', 'userId', { transaction })
      await transaction.commit()
    } catch (error) {
      await transaction.rollback()
      throw error
    }
  },
}
```

Dans ce fichier ,je fais une transaction qui crée une table supplémentaire dans la table `houses` , en lui ajoutant le champ userId qui sera non null.

Je procède ainsi pour toutes les autres relations de mon modèle de données.

Pour vérifier toutes ces manipulations , je vais sur mon terminal

```
> \c hotels_dev
> \dt
```

Je constate donc que toutes mes tables ont bien été créées .

hotels_devs-# \dt			
List of relations			
Schema	Name	Type	Owner
public	SequelizeMeta	table	postgres
public	bookings	table	postgres
public	houses	table	postgres
public	images	table	postgres
public	mails	table	postgres
public	suites	table	postgres
public	users	table	postgres
(7 rows)			

1.1.2.2 Model

Le modèle n'a pas d'incidence directe sur la base de donnée , mais c'est lui que je vais manipuler dans mon application Express. Je dois donc lui définir les mêmes caractéristiques que mes migrations , pour assurer l'intégrité des données , à savoir les validations, les contraintes et les relations.

Voici le modèle User

```
module.exports = (sequelize, DataTypes) => {
  You, 5 minutes ago | 2 authors (zoutigo and others)
  class User extends Model {
    You, 1 second ago + Uncommitted changes
    static associate({ house, booking }) {
      house.belongsTo(this, {
        onDelete: 'SET NULL',
        onUpdate: 'CASCADE',
      })
      this.hasOne(house, {
        foreignKey: 'userId',
      })
      this.hasMany(booking, {
        foreignKey: 'userId',
        onDelete: 'SET NULL',
        onUpdate: 'CASCADE',
      })
      booking.belongsTo(this)
    }
    toJSON() {
      return { ...this.get(), id: undefined }
    }
  }
}
```

```
User.init({
  uid: {
    type: DataTypes.UUID,
    defaultValue: Sequelize.UUIDV4,
  },
  lastname: {
    type: DataTypes.STRING(30),
    allowNull: false,
    validate: {
      notNull: {
        msg: 'le prénom est obligatoire',
      },
      len: {
        args: [2, 30],
        msg: 'le prénom doit avoir entre 2 et 30 caractères',
      },
    },
  },
  firstname: {
    type: DataTypes.STRING(30),
    allowNull: false,
    validate: {
      notNull: {
        msg: 'le nom est obligatoire',
      },
      len: {
        args: [2, 30],
        msg: 'le nom doit avoir entre 5 et 30 caractères',
      },
    },
  },
  email: {
    type: DataTypes.STRING(50),
    unique: true,
    validate: {
      isEmail: {
        msg: 'ce format mail n'est pas valide',
      },
    },
  },
})
```

Dans la première partie de la classe , je définis mes associations

Dans la seconde , je mets les critères qui étaient déjà dans la migration.

Je reproduis donc la même opération pour toutes mes entités.

1.1.2.2 Seed ou fixtures

Contrairement au modèle et à la migration , le seed se crée avec une autre commande .

```
npx sequelize-cli seed:generate --name demo-user
```

Ceci a pour effet de créer un fichier dans le répertoire seeder que je dois améliorer.

Je définis l'admin et son mot de pass , puis les managers et quelques clients

```

module.exports = {
  async up(queryInterface, Sequelize) {
    // password hash
    const transaction = await queryInterface.sequelize.transaction()
    try {
      const password = await hashPassword('password')
      devAdmin.password = password
      const clients = devClients.map((client) => ({
        ...client,
        password: password,
      }))
      const managers = devManagers.map((manager) => ({
        ...manager,
        password: password,
      }))
      await queryInterface.bulkInsert('users', [devAdmin, ...managers], {
        transaction,
      })
      await queryInterface.bulkInsert('users', clients, {
        transaction,
      })
      await transaction.commit()
    } catch (err) {
      await transaction.rollback()
      throw err
    }
  },
  async down(queryInterface, Sequelize) {
    const transaction = await queryInterface.sequelize.transaction()
    try {
      await queryInterface.bulkDelete('users', null, { transaction })
      await transaction.commit()
    } catch (error) {
      await transaction.rollback()
      throw error
    }
  },
}

```

```

const devManagers = []
for (let a = 0; a < 10; a++) {
  const devManager = {
    lastname: faker.name.lastName(),
    firstname: faker.name.firstName(),
    email: `managers${a}@test.com`,
    password: 'password',
    roles: ['manager'],
    createdAt: new Date(),
    updatedAt: new Date(),
  }
  devManagers.push(devManager)
}

const devAdmin = {
  lastname: faker.name.lastName(),
  firstname: faker.name.firstName(),
  email: 'admin@test.com',
  password: 'password',
  roles: ['admin'],
  createdAt: new Date(),
  updatedAt: new Date(),
}

const devClients = []
for (let i = 0; i < 10; i++) {
  const client = {
    lastname: faker.name.lastName(),
    firstname: faker.name.firstName(),
    email: faker.internet.email(),
    password: 'password',
    roles: ['client'],
    createdAt: new Date(),
    updatedAt: new Date(),
  }
  devClients.push(client)
}

```

Je peux désormais jouer la fixture grace à la commande

`npx sequelize-cli db:seed:all`

Toutes ces données vont donc être créés dans ma base de donnée. Je le vérifie sur psql

```

hotels_devs=# SELECT email FROM users ;
 email
-----
admin@test.com
manager0@test.com
manager1@test.com
manager2@test.com
manager3@test.com
manager4@test.com
manager5@test.com
manager6@test.com
manager7@test.com
manager8@test.com
manager9@test.com
Maximo.Dietrich@gmail.com
Van90@yahoo.com
Marlen.Runolfsson24@yahoo.com
Ward_Gleason@gmail.com
Ken.Jaskolski@yahoo.com
Shawn.Waelchi72@hotmail.com
Christelle62@yahoo.com
Danielle_Cronin39@hotmail.com
Madyson_Zulauf17@yahoo.com
Bethel11@hotmail.com
(21 rows)

```

Je peux donc me consacrer à la création du backend avec express.

1.2 Express

2. React

Annexes.

1. config js de sequelize

```
require('dotenv').config()
module.exports = {
  development: {
    username: process.env.POSTGRES_USER,
    password: process.env.POSTGRES_PASSWORD,
    database: process.env.POSTGRES_DB,
    host: '127.0.0.1',
    port: 5432,
    dialect: 'postgres',
    dialectOptions: {
      bigNumberStrings: true,
    },
  },
  test: {
    username: process.env.POSTGRES_USER,
    password: process.env.POSTGRES_PASSWORD,
    database: process.env.POSTGRES_DB,
    host: '127.0.0.1',
    port: 3306,
    dialect: 'postgres',
    dialectOptions: {
      bigNumberStrings: true,
    },
  },
  production: {
    username: process.env.PROD_DB_USERNAME,
    password: process.env.PROD_DB_PASSWORD,
    database: process.env.PROD_DB_NAME,
    host: process.env.PROD_DB_HOSTNAME,
    port: process.env.PROD_DB_PORT,
    dialect: 'postgres',
    dialectOptions: {
      bigNumberStrings: true,
    },
  },
}
```

