# Introduction

De nos jours, il devient de plus en plus facile pour une personne de réserver une table dans un restaurant, mais toutes ses applications que nous utilisons ne sont pas optimisées entièrement pour les restaurateurs. C’est pourquoi, avec l’aide de M. Garcia et de Mme. Perdrizat (gérante du restaurant « l’Atelier » à Genève), nous avons décidé de revoir entièrement le fonctionnement d’une application de gestion de réservation mais en concentrant nos efforts sur le restaurateur.

Cette application permettra donc facilement au restaurateur de gérer ses réservations, mais surtout son établissement.

# Mise en place

## GitHub

Afin d’avoir un suivi constant de mon projet, j’ai décider de créer un GitHub. Dans ce github j’ai donc régulièrement mis à jour le code et la documentation.

Le github est structuré de la manière suivante :

Travail\_Diplome\_ES\_2020

├── Documentation

├── Tests

├── RESA

├── README.md

├── logbook.md

Le dossier RESA contient tout le code source de l’application.

## Trello

Trello est un système de gestion du temps qui permet de facilement créer, déplacer et terminer des tâches.

J’ai créé 5 colonnes :

1. A faire
2. En cours
3. En validation
4. Terminés
5. En continu

La colonne 3 « En validation » ce sont les tâches terminées qui demande une validation de la part de M. Garcia afin de pouvoir classer la tâche dans la colonne terminée. La colonne 5 « En continu », représente la colonne des tâches que je dois suivre en continu (ex. le journal de bord).

# Recherches post-travail

Afin de pouvoir réaliser au mieux mon travail, j’ai dû rechercher les langages de programmation et des librairies qui vont m’aider le mieux que possible à réaliser le travail qui m’est demandé dans le cahier des charges.

La bibliothèque JavaScript qui m’a semblée le plus adaptée à mes besoins est celle de « React ». En effet, React est une bibliothèque Javascript pensée pour la création d’interfaces utilisateurs.

« React est une bibliothèque JavaScript déclarative, efficace et flexible pour construire des interfaces utilisateurs (UI). Elle vous permet de composer des UI complexes à partir de petits morceaux de code isolés appelés « composants ». » - React

React fonctionne à base de « **composants** » qui peuvent prendre de propriétés nommées « **props** ». Ce composant renvoie une arborescence de vues à afficher via la méthode « **render** »

## Installation

Tout d’abord je dois disposer d’une mise à jour récente de Node.js. Afin de créer une application de test, je dois entrer la commande suivante dans le dossier ou je souhaite créer l’application :

npx create-react-app my-app

“my-app” représente le nom de l’application

Une fois l’application crée, on obtient un dossier contenant l’architecture suivante

my-app

├── README.md

├── node\_modules

├── package.json

├── .gitignore

├── public

│ ├── favicon.ico

│ ├── index.html

│ └── manifest.json

└── src

├── App.css

├── App.js

├── App.test.js

├── index.css

├── index.js

├── logo.svg

└── serviceWorker.js

Le dossier « src » contient tout le code de l’application en tant que tel, c’est-à-dire les pages html, js, etc.

# Analyse organique

## Base de données

Afin de pouvoir stocker les données, j’ai créé une base de données nommée « resa ».

### Privilèges

Pour accéder à la base de données, il faut utiliser les privilèges suivants :

* Username : **resa\_tech\_es**
* Password : **WhutMerYmZeR6EHb**

### Structure

Ma base de données est structurée de la manière suivante :

Resa

├── client

│ ├── id

│ ├── first\_name

│ ├── last\_name

│ ├── phone

│ ├── email

├── dish