

# Marieteam Web

## I. Vue d'ensemble du projet

- A. Les stacks
- B. Architecture du site
- C. Diagramme cas d'utilisation
- D. Schéma de la base de données
- E. Gestion du versionning
- F. Le déploiement du site

## II. Fonctionnalités

- A. Authentification
- B. Réservation
- C. Mon compte
- D. Administration

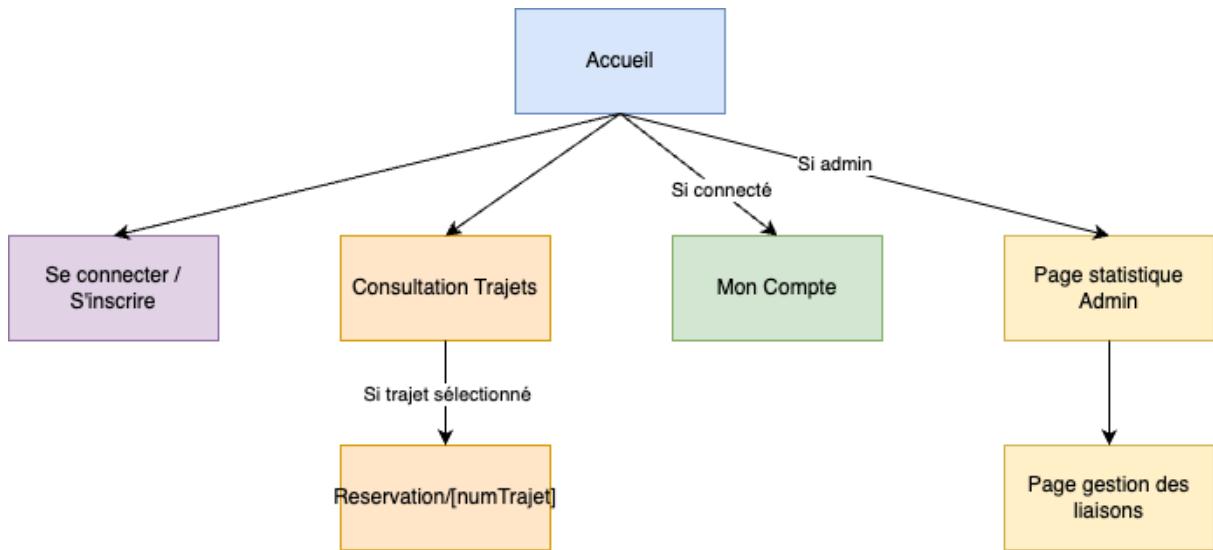
## I. Vue d'ensemble du projet

### A. Les stacks

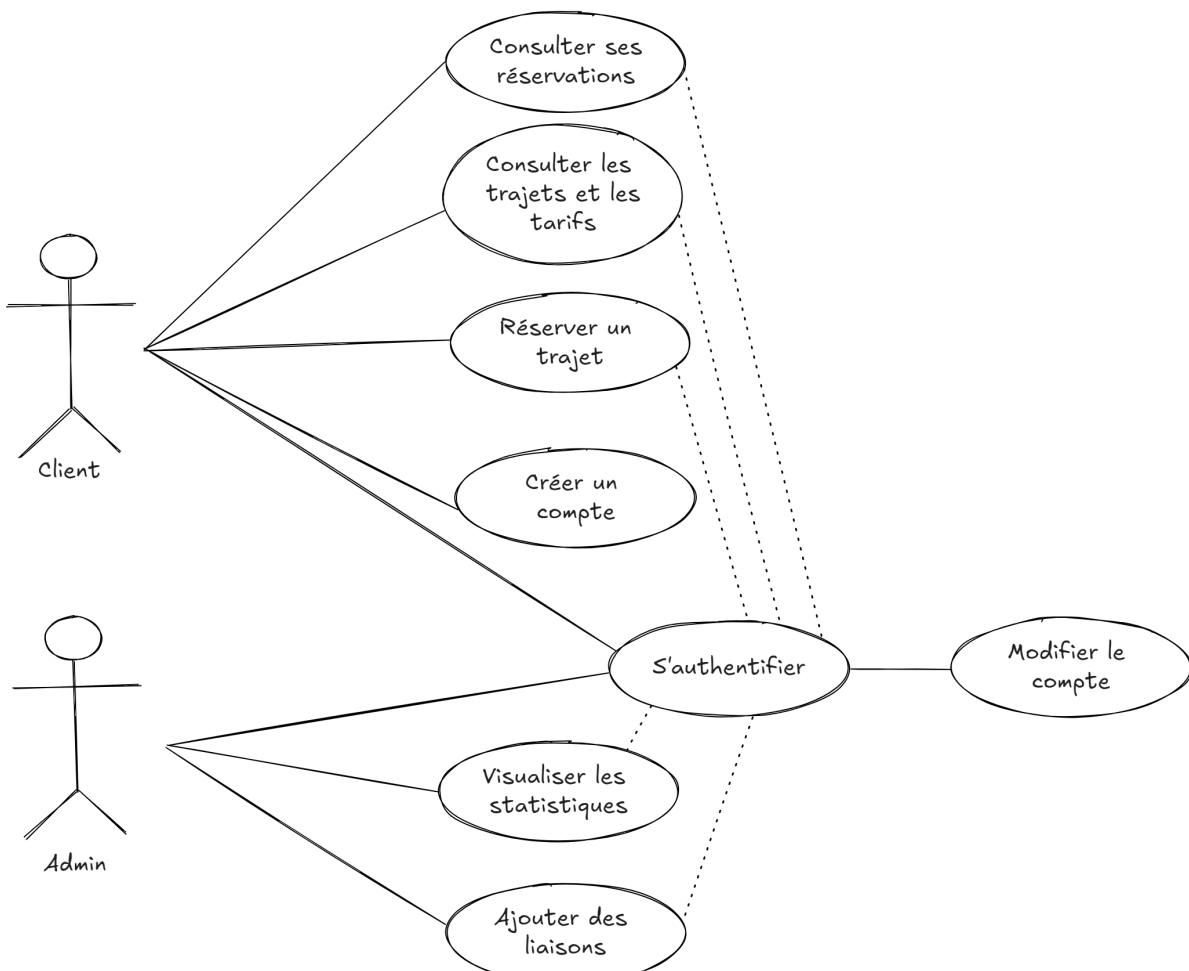
Marieteam Web est développé en JavaScript avec le framework Next.js. Le style est réalisé avec la bibliothèque Tailwind CSS.

Pour la base de données, le choix a été fait de la réaliser en PostgreSQL avec la solution Supabase, qui permet de gérer plusieurs méthodes plus facilement comme l'authentification, etc.

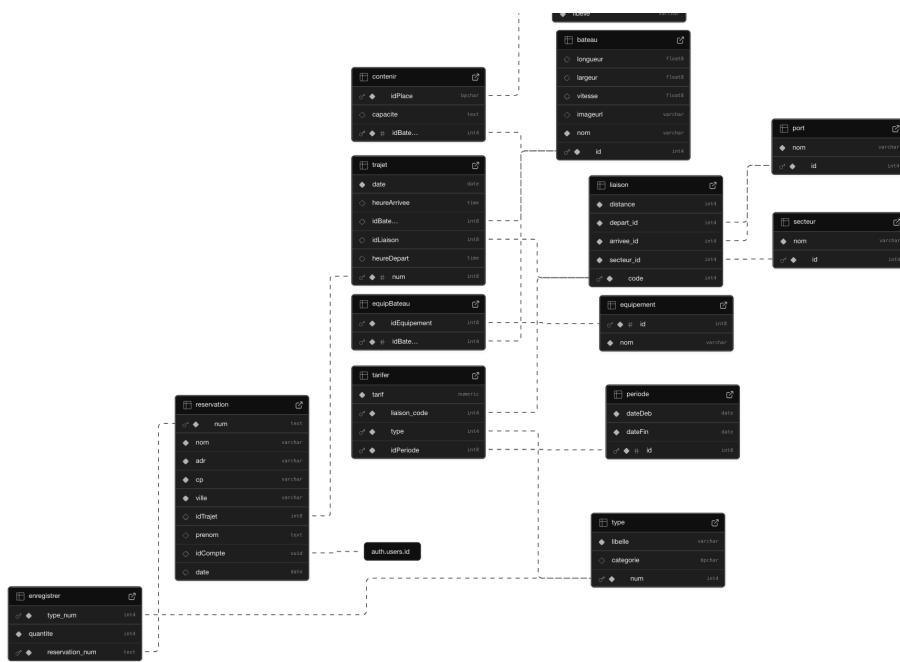
### B. Architecture du site



## C. Diagramme cas d'utilisation



## D. Schéma de la base de données

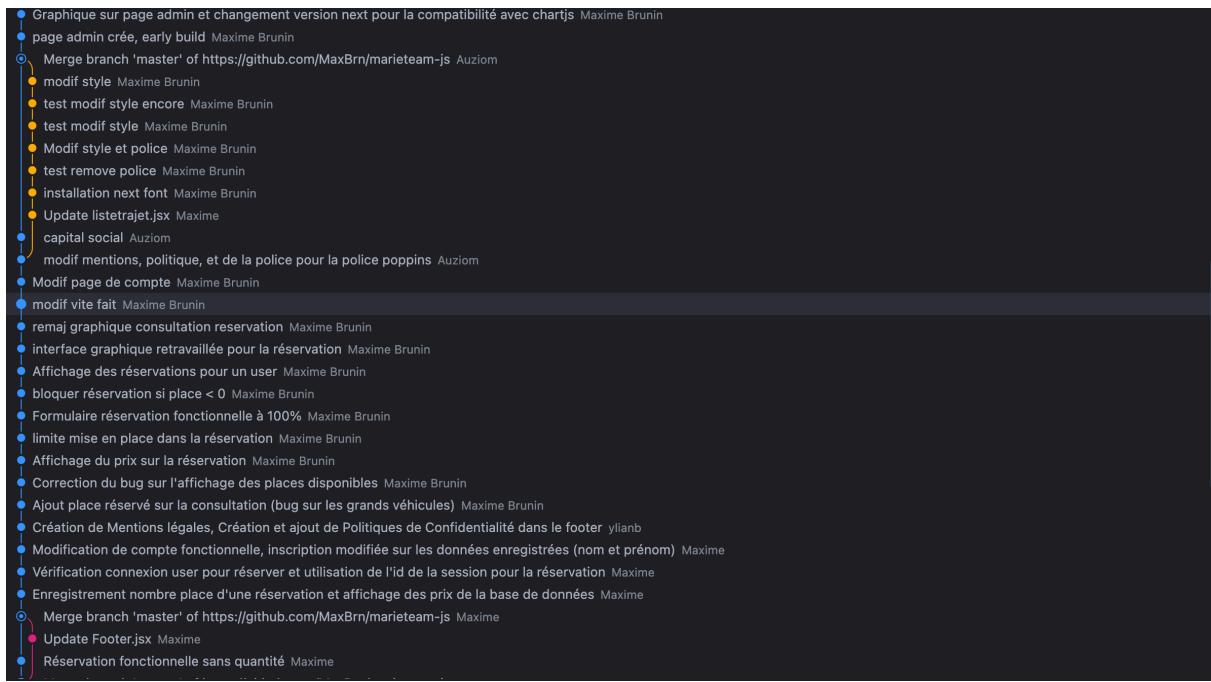


## E. Gestion du versionning

Le projet est enregistré sur un dépôt distant Git afin de mieux gérer le versionning et la collaboration entre les membres du groupe.

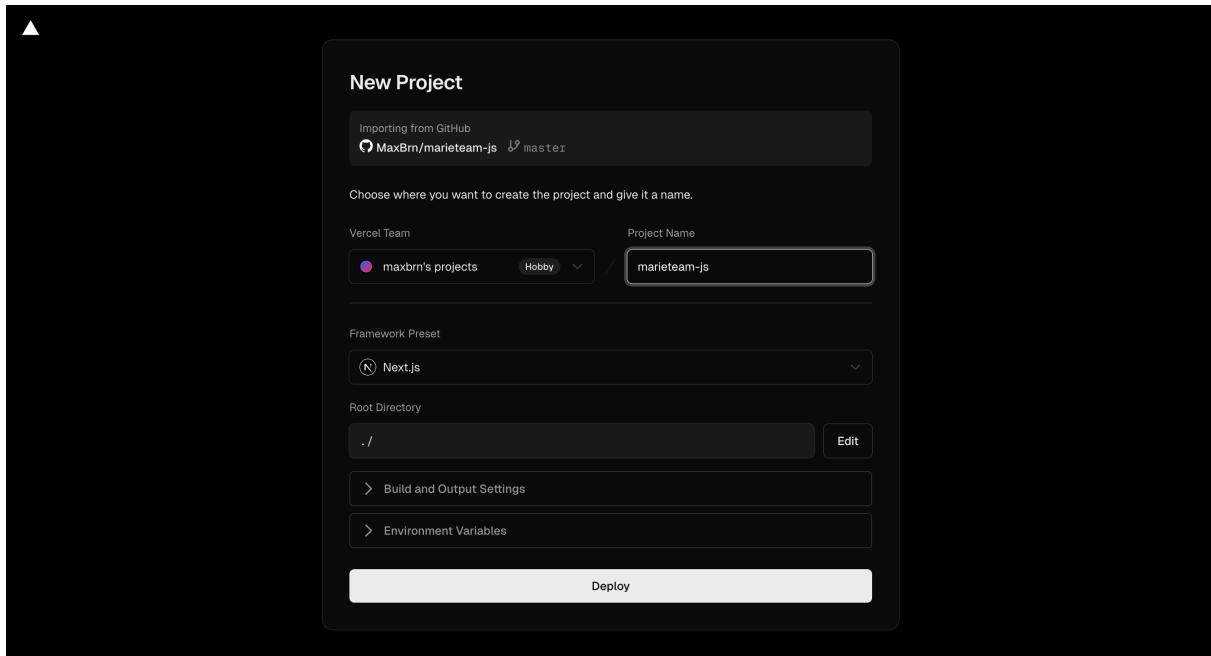
The screenshot shows the GitHub repository page for 'marieteam-js'. Key details include:

- Repository Information:** Private, master branch, 1 branch, 0 tags.
- Commits:** 90 commits, latest commit by MaxBrn on d080188, 3 weeks ago.
- Files:** app, components, lib, pages, public, .eslintrc.json, .gitignore, README.md, components.json, jsconfig.json, next.config.mjs, package-lock.json, package.json, postcss.config.mjs, tailwind.config.js.
- Activity:** 1 watching, 0 forks.
- Releases:** No releases published, Create a new release.
- Packages:** No packages published, Publish your first package.
- Contributors:** MaxBrn, Auziom, Ylianb.
- Deployments:** 77, Production 3 weeks ago.



## F. Le déploiement du site

Pour déployer le site, nous avons choisi d'utiliser Vercel. Pour se faire on a juste à renseigner la branche Git sur laquelle se trouve le projet et le déployer:



Une fois déployé, on a accès au dashboard du projet qui permet de gérer plusieurs choses, comme l'ajout d'outils analytiques, de voir l'url du projet, etc.

The screenshot shows the Vercel Project dashboard for the 'marieteam-js' project. At the top, there's a navigation bar with links to Project, Deployments, Analytics, Speed Insights, Logs, Observability, Firewall, Storage, Flags, AI, and Settings. Below the navigation is the project name 'marieteam-js'. To the right of the project name are buttons for Repository, Usage, Domains, Visit, Build Logs, Runtime Logs, and Instant Rollback. The main content area is titled 'Production Deployment' and features a preview of the website. The preview shows a landing page for 'Marieteam' with a boat image and a cruise ship image. To the right of the preview, there's a 'Deployment' card for 'marieteam-jm0xepy6w-maxbrns-projects.vercel.app'. The deployment status is 'Ready' (green). Below the deployment card are sections for 'Source' (master branch) and 'Analytics' (Track visitors and page views). At the bottom of the dashboard, there are three circular cards: 'Firewall 24h' (blue), 'Observability 6h' (blue), and 'Analytics' (grey).

Le site est accessible à l'url suivant: [marieteam.vercel.app](https://marieteam.vercel.app).

## II. Fonctionnalités

### A. Authentification

Supabase gère l'authentification avec des fonctions prédéfinies

```
const { data, error } = await supabase.auth.signInWithEmailAndPassword({
  email: mail,
  password: mdp,
});

if (error) {
  setError(error.message || "Erreur de connexion.");
  setLoading(false);
  return;
}
```

```
// Si la connexion est réussie, récupérer le token JWT de Supabase
const token = await data.session.access_token;
```

```
// Stocker le token dans un cookie
Cookies.set("token", token, { expires: 1, path: "/" });

showNotification("success", "Connexion réussie");
```

Ici, Supabase vérifie les informations obtenues et si tout est bon on récupère un token pour pouvoir vérifier à chaque étape si le token est valide.

Par exemple avoir ce token de session permet de récupérer l'utilisateur connecté et de ce fait récupérer ses rôles et ajuster ses accès en fonction de ces derniers.

```
const fetchUserData = async () => {
  if (typeof window !== "undefined") {
    const tokenFromCookie = Cookie.get("token");
    setToken(tokenFromCookie || null);

    if (tokenFromCookie) {
      try {
        const {
          data: { user },
        } = await supabase.auth.getUser();
        const userRole = user?.user_metadata?.role;
        setRole(userRole);
      } catch (error) {
        console.error("Erreur lors de la récupération du rôle:", error);
        setRole(null);
      }
    } else {
      setRole(null);
    }
  };
};
```

Quand l'user se déconnecte, de ce fait on détruit le cookie et la session est terminée

```

const handleLogout = async () => {
  try {
    await supabase.auth.signOut();
    Cookie.remove("token");
    setToken(null);
    setRole(null);
    showNotification("success", "Déconnexion réussie");
    router.push("/");
  } catch (error) {
    console.error("Erreur lors de la déconnexion:", error);
    showNotification("error", "Erreur lors de la déconnexion");
  }
};

```

Il faut aussi noter que Supabase demande à l'utilisateur de vérifier son compte via un mail envoyé qui contient un lien d'activation. Sans quoi l'utilisateur ne peut pas accéder au site web.

Lors de l'inscription, le mot de passe doit être assez fort pour être accepté, des directives sont indiquées à l'utilisateur. Sans un mot de passe valide le compte ne peut pas être créé.

**Inscription**

Nom  
Brunin

Prénom  
Maxime

Email  
maxime.brunin@gastonberger.fr

Mot de passe

Confirmer le mot de passe

Un mot de passe valide doit contenir :

- 13 caractères minimum
- Une majuscule minimum
- Une minuscule minimum
- Un caractère spécial minimum

S'inscrire

Vous avez déjà un compte ? [Connectez-vous](#)

The screenshot shows a confirmation message for a successful registration. At the top right, there is a "Réserver un trajet" button with a car icon. On the left, the "Marieteam" logo is visible. The main message is displayed in a green bar: "Inscription réussie" (Registration successful) with a small circular icon. Below this, a note reads: "Encore une dernière chose ! Vous allez recevoir sous peu un mail contenant un lien permettant de vérifier votre compte. Sans ça vous ne pourrez pas accéder au site Marieteam !" (One last thing! You will soon receive an email containing a link to verify your account. Without it, you won't be able to access the Marieteam website!). A blue "Compris" button is at the bottom.

Marieteam

Réserver un trajet

Inscription réussie

Encore une dernière chose !

Vous allez recevoir sous peu un mail contenant un lien permettant de vérifier votre compte. Sans ça vous ne pourrez pas accéder au site Marieteam !

Compris

Copyright @ 2024 Marieteam

Politiques de Confidentialité

Mentions légales

Voici le mail que reçoit l'utilisateur à son inscription pour valider:

## Confirmer votre inscription

☀️ ☺️ ⏪ ⏴ ⏵

SA

○ Supabase Auth <noreply@mail.app.supabase.i...

Aujourd'hui à 14:13

À : ☎ Maxime BRUNIN

## Valider votre compte Marieteam

Pour utiliser nos services vous devez confirmer votre inscription via le lien suivant:

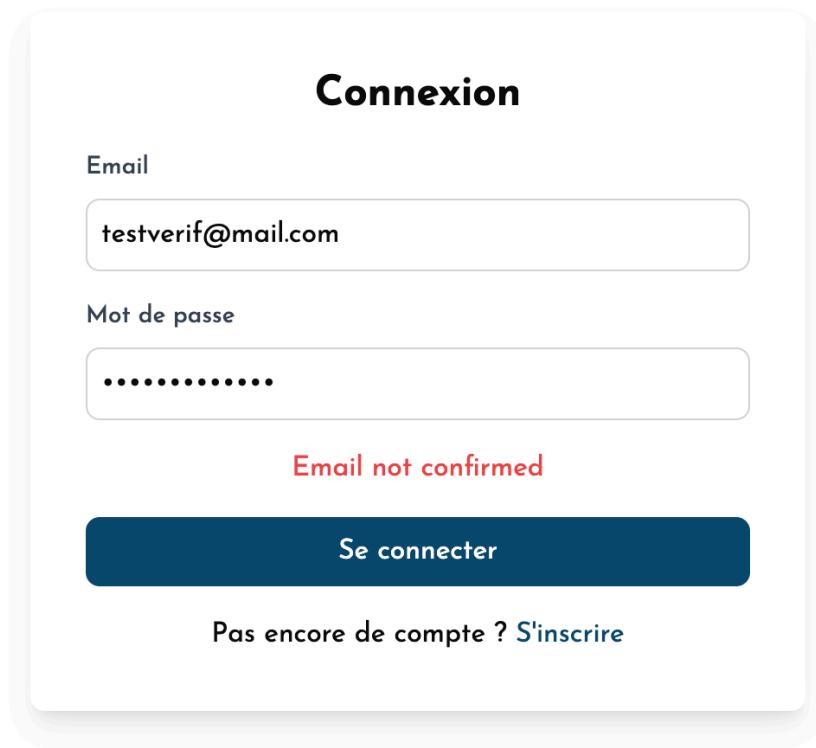
[Confirmer le mail](#)

L'équipe Marieteam

You're receiving this email because you signed up for an application powered by **Supabase** ⚡

[Opt out of these emails](#)

Une fois le compte validé, il peut se connecter, s'il n'est pas vérifié supabase refusera la connexion en retournant une erreur:



## B. Réservation

D'abord pour réserver, l'utilisateur doit être connecté. S'il ne l'est pas il sera redirigé:

```
const handleReservationClick = (e, trajetNum) => {
  const tokenFromCookie = Cookies.get("token");

  // Vérifie si le token est présent
  if (!tokenFromCookie) {
    e.preventDefault(); // Empêche le lien de fonctionner immédiatement

    // Enregistre l'ID du trajet dans les cookies
    Cookies.set("resTrajet", trajetNum, { expires: 1, path: "/" });

    // Redirige l'utilisateur vers la page de connexion
    router.push("/connexion");
}
```

```
    }  
};
```

Ici on vérifie si le token de session est présent pour être sûr que l'utilisateur est log, sinon il est redirigé vers la page de connexion. Aussi, le numéro de trajet est pris en cookie, de cette manière à sa connexion l'utilisateur sera redirigé directement sur le trajet qu'il voulait réserver.

On effectue cette vérification dans la page Connexion:

```
const reserveTrajet = Cookie.get("resTrajet");  
if (reserveTrajet) {  
    router.push(`/reservation/${reserveTrajet}`);  
    Cookie.remove("resTrajet");  
} else {  
    router.push("/");  
}
```

Sur la page, l'utilisateur peut rechercher un trajet via les champs, ces derniers sont non remplissables manuellement et sont des champs à sélections qui prennent en base de données les données disponibles. D'abord il choisit un secteur, le site retourne les liaisons disponibles dans ce secteur et affiche le champ pour en sélectionner une et l'utilisateur choisit une date. A noter que le champ de date ne peut sélectionner de date antérieure à celle du jour :



Une fois les champs remplis, l'utilisateur clique sur rechercher et choisit le trajet qu'il souhaite, il peut voir les places disponibles et choisir de réserver ce trajet.

En cliquant sur réserver ce trajet, le code va récupérer l'id du trajet et le transmettre à la page suivante, depuis cette page s'effectueront toutes les récupérations de données associées au trajet, comme les tarifs, les places disponibles, etc.

```

<Link
  href={`/reservation/${selectedTrajet.num}`}
  onClick={(e) => handleReservationClick(e, selectedTrajet.num)}
  className="p-3 bg-sky-900 rounded-xl text-white w-[80%] m-auto block t
>
  Réserver ce trajet
</Link>;

```

Ce bouton permet la redirection, ici handleReservationClick permet de vérifier si l'user est log, sinon la redirection se fait vers la page /reservation/[trajetNum]. Ou trajetNum est un argument passé en paramètre afin d'être récupéré de l'autre côté. Ici on donne celui du trajet que l'user a sélectionné.

Une fois sur la page réservation, toute la récupération des données s'effectue, voilà par exemple comment on retourne toutes les informations du trajet sélectionné à l'affichage:

```

export async function getServerSideProps(context) {
  const { trajetNum } = context.params;

  console.log(trajetNum);
  const { data: trajet, error } = await supabase
    .from('trajet')
    .select('num, heureDepart, heureArrivee, idBateau, idLiaison, date')
    .eq('num', trajetNum)
    .single();

  if (trajet) {
    console.log('trajet trouvé');
  } else {
    console.log('trajet non trouvé');
  }

  const { data: bateau } = await supabase
    .from('bateau')
    .select('nom')

```

```

.eq('id', trajet.idBateau)
.single();

const { data: liaison } = await supabase
  .from('liaison')
  .select('code, depart_id, arrivee_id')
  .eq('code', trajet.idLiaison)
  .single();

const { data: portDepart } = await supabase
  .from('port')
  .select('nom')
  .eq('id', liaison.depart_id)
  .single();

const { data: portArrivee } = await supabase
  .from('port')
  .select('nom')
  .eq('id', liaison.arrivee_id)
  .single();

const {data: reservation, error:errorReservation} = await supabase
  .from('reservation')
  .select('num')
  .eq('idTrajet',trajet.num);

let placePassagerReserv = 0;
let placePetitVehReserv = 0;
let placeGrandVehReserv = 0;

await Promise.all(
  reservation.map(async (res) => {
    const reservationNum = res.num;

    const { data: passagerReserv, error: errorPassagerReserv } = await supabase
      .from('enregistrer')
      .select('quantite')
      .eq('reservation_num', reservationNum)
  })
);

```

```

.or('type_num.eq.1,type_num.eq.2,type_num.eq.3');

if (!errorPassagerReserv) {
    placePassagerReserv += passengerReserv.reduce((sum, row) => sum + ro
}

const { data: petitVehiculeReserv, error: errorPetitVehiculeReserv } = await
    .from('enregistrer')
    .select('quantite')
    .eq('reservation_num', reservationNum)
    .or('type_num.eq.4,type_num.eq.5');

if (!errorPetitVehiculeReserv) {
    placePetitVehReserv += petitVehiculeReserv.reduce((sum, row) => sum + ro
}

const { data: grandVehiculeReserv, error: errorGrandVehiculeReserv } = await
    .from('enregistrer')
    .select('quantite')
    .eq('reservation_num', reservationNum)
    .or('type_num.eq.6,type_num.eq.7');

if (!errorGrandVehiculeReserv) {
    placeGrandVehReserv += grandVehiculeReserv.reduce((sum, row) => sum + ro
}
};

const { data: placePassager } = await supabase
    .from('contenir')
    .select('capacite')
    .eq('idBateau', trajet.idBateau)
    .eq('idPlace', 'A')
    .single();

const { data: placePetitVehicule } = await supabase
    .from('contenir')
    .select('capacite')

```

```

.eq('idBateau', trajet.idBateau)
.eq('idPlace', 'B')
.single();

const { data: placeGrandVehicule } = await supabase
  .from('contenir')
  .select('capacite')
  .eq('idBateau', trajet.idBateau)
  .eq('idPlace', 'C')
  .single();

const {data: prix } = await supabase
  .from('tarifer')
  .select('tarif')
  .eq('liaison_code',liaison.code)
  .order('type', {ascending: true});

const heureDepartFormat = trajet.heureDepart.substring(0,2) + 'h' + trajet.heureDepart.substring(3,5);
const heureArriveeFormat = trajet.heureArrivee.substring(0,2) + 'h' + trajet.heureArrivee.substring(3,5);
const dateFormat = trajet.date.substring(8,10) + '/' + trajet.date.substring(5,7) + '-' + trajet.date.substring(0,4);

const { hours, minutes } = () => {
  const [heureD, minuteD] = trajet.heureDepart.split(':');
  const [heureA, minuteA] = trajet.heureArrivee.split(':');

  const depart = new Date();
  depart.setHours(heureD, minuteD, 0, 0);

  const arrivee = new Date();
  arrivee.setHours(heureA, minuteA, 0, 0);

  const differenceInMillis = arrivee - depart;
  const differenceInMinutes = Math.floor(differenceInMillis / 60000);

  return {
    hours: Math.floor(differenceInMinutes / 60),
    minutes: differenceInMinutes % 60,
  };
}

```

```

})();

return {
  props: {
    trajet: {
      ...trajet,
      num: trajetNum,
      nomBateau: bateau.nom,
      tempsTrajet: `${hours}h ${minutes}m`,
      portDepart: portDepart.nom,
      portArrivee: portArrivee.nom,
      placePassager: placePassager.capacite - placePassagerReserv,
      placePetitVehicule: placePetitVehicule.capacite - placePetitVehReserv,
      placeGrandVehicule: placeGrandVehicule.capacite - placeGrandVehReserv,
      heureDepartFormat: heureDepartFormat,
      heureArriveeFormat: heureArriveeFormat,
      dateFormat: dateFormat,
      prix: prix
    },
  },
};

```

Une fois les champs remplis, l'utilisateur valide en cliquant sur réserver et aussi en validant une deuxième fois via une boîte de dialogue:

**Marseille - Nice**

le 30/04/2025 de 13h32 à 15h32

Nom Prénom Adresse Code Postal Ville Adulte 50 € Junior 40 € Enfant 30 € Voiture 70 € Camionnette 80 € CampingCar 90 € Camion 100 € 

Total à payer: 240 €

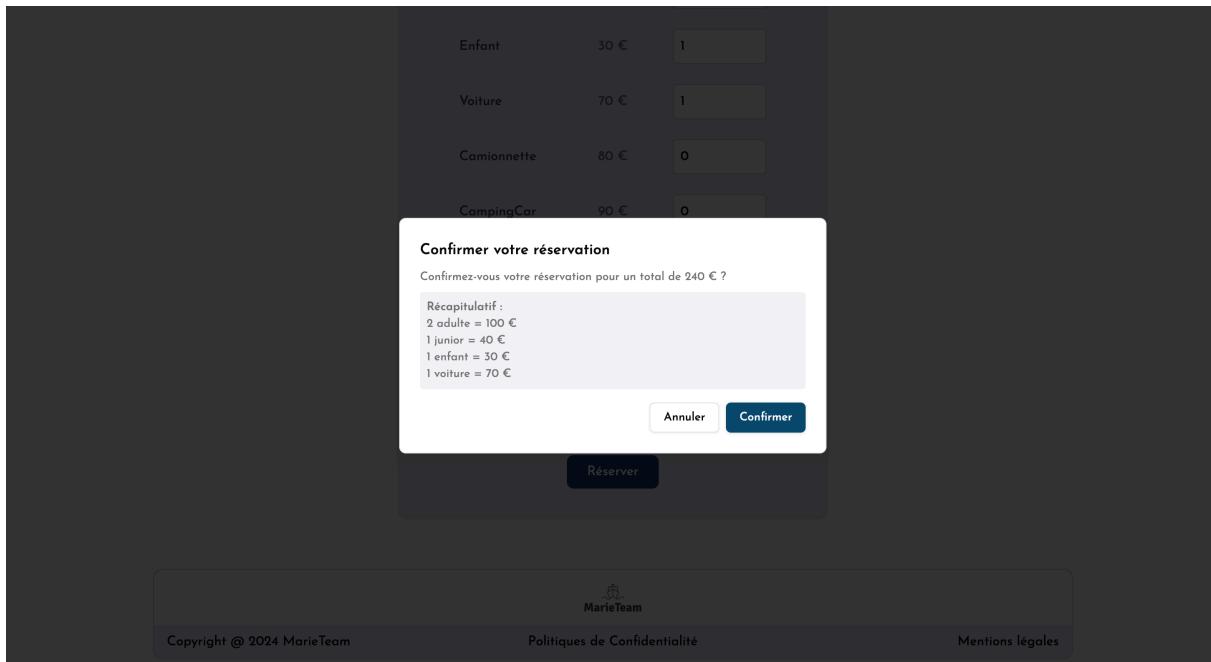
Places disponibles:

Place passager: 270

Place véhicule inférieur à 2m: 15

Place véhicule supérieur à 2m: 0

**Réserver**



S'il valide et que les vérifications faites du côté du code, pour les saisies et la validité des champs sont bonnes, alors le code prend en compte les informations et va enregistrer la réservation en base de données. D'abord il va créer un identifiant unique composé du timestamp du moment de réservation et l'id du compte, ainsi on peut s'assurer que chaque réservation a un identifiant unique.

```
const generateUniqueReservationNum = (idCompte) => {
  const timestamp = Date.now();
  return `${idCompte}-${timestamp}`;
};
```

Si tout est bon alors on va créer la réservation avec confirmReservation:

```
const confirmReservation = async () => {
  setIsSubmitting(true);
  setLoading(true);
  setShowConfirmDialog(false);

  try {
    const { data } = await axios.post('http://localhost:3001/reservations', {
      idCompte: idCompte,
      timestamp: timestamp,
      ...selectedItems
    });
  } catch (error) {
    console.error(error);
  }
};
```

```

data: { user },
} = await supabase.auth.getUser();
const idUser = user.id;
const numReservation = generateUniqueReservationNum(idUser);
const today = new Date().toISOString().split("T")[0];

// Insertion de la réservation principale
const { error } = await supabase.from("reservation").insert([
{
    num: numReservation,
    idTrajet: trajet.num,
    idCompte: idUser,
    nom: formData.nom,
    prenom: formData.prenom,
    adr: formData.adresse,
    cp: formData.codePostal,
    ville: formData.ville,
    date: today,
},
]);
if (error) throw error;

// Insertion des places réservées
const insertPromises = [];
if (formData.adulte > 0) {
    insertPromises.push(
        supabase.from("enregistrer").insert([
            {
                type_num: 1,
                reservation_num: numReservation,
                quantite: formData.adulte,
            },
        ],
    )
);
}
if (formData.junior > 0) {
    insertPromises.push(

```

```
supabase.from("enregistrer").insert([
  {
    type_num: 2,
    reservation_num: numReservation,
    quantite: formData.junior,
  },
])
);
}

if(formData.enfant > 0) {
  insertPromises.push(
    supabase.from("enregistrer").insert([
      {
        type_num: 3,
        reservation_num: numReservation,
        quantite: formData.enfant,
      },
    ])
  );
}

if(formData.voiture > 0) {
  insertPromises.push(
    supabase.from("enregistrer").insert([
      {
        type_num: 4,
        reservation_num: numReservation,
        quantite: formData.voiture,
      },
    ])
  );
}

if(formData.camionnette > 0) {
  insertPromises.push(
    supabase.from("enregistrer").insert([
      {
        type_num: 5,
        reservation_num: numReservation,
        quantite: formData.camionnette,
      },
    ])
  );
}
```

```

        },
    ])
);
}
}

if (formData.campingCar > 0) {
    insertPromises.push(
        supabase.from("enregistrer").insert([
            {
                type_num: 6,
                reservation_num: numReservation,
                quantite: formData.campingCar,
            },
        ])
);
}

if (formData.camion > 0) {
    insertPromises.push(
        supabase.from("enregistrer").insert([
            {
                type_num: 7,
                reservation_num: numReservation,
                quantite: formData.camion,
            },
        ])
);
}

await Promise.all(insertPromises);

// Récupération des détails de la réservation
const reservation = await recupRes(numReservation);
setSelectedReservation(reservation);
setDone(true);
showNotification("success", "Réservation effectuée avec succès !");
} catch (error) {
    console.error(error);
    showNotification(
        "error",

```

```

    "Une erreur est survenue lors de la réservation."
);
} finally {
    setIsSubmitting(false);
    setLoading(false);
}
};

```

Une fois créée, la réservation est récupérée et un récapitulatif est affiché à l'utilisateur.

The screenshot shows a confirmation message: "Merci pour votre réservation ! Votre réservation a bien été prise en compte. Nous vous remercions de nous avoir choisi, et avons hâte de vous retrouver sur les flots !" from "L'équipe MarieTeam". Below this, it displays the reservation details: "Réservation e4a32db3-9955-4a33-a72e-4281260e7177-1746012913419" for a trip from Marseille to Nice on 30/04/2025 between 13h32 and 15h32. A table shows reserved places: Adulte (2 places, 50 € / place), Junior 8 à 18 ans (1 places, 40 € / place), Enfant 0 à 7 ans (1 places, 30 € / place), and Voiture (1 places, 70 € / place). The total price is 240.00 €. A "Retourner à l'accueil" button is at the bottom.

## C. Mon compte

La page mon compte permet à l'utilisateur de visualiser ses données et ses réservations. Il peut modifier les informations de son compte et afficher les détails de toutes ses réservations en cliquant sur le ticket présent à côté de son numéro de réservation.

## Mon Compte



Prenom  
Maxime

Nom  
Brunin

Email  
maxime\_brn@outlook.fr

## Mes Réservations

Trier par date ↓

Marseille - Nice le 30/04/2025 de 13h32 à 15h32



Marseille - Nice le 30/04/2025 de 13h32 à 15h32



Marseille - Nice le 30/04/2025 de 13h32 à 15h32



Marseille - Nice le 30/04/2025 de 13h32 à 15h32



Marseille - Nice le 25/04/2025 de 09h12 à 13h12



← 1 →

## Mon Compte



Prenom  
Maxime

Nom  
Brunin

Email  
maxime\_brn@

## Mes Réserv

Marseille - Nice

Marseille - Nice

Marseille - Nice

Marseille - Nice

Réservation e4a32db3-9955-4a33-a72e-4281260e7177-  
1746012913412

Marseille - Nice le 30/04/2025 de 13h32 à 15h32

Trier par date ↓

### Places réservées

Adulte	2 places, 50 € / place
Junior 8 à 18 ans	1 places, 40 € / place
Enfant 0 à 7 ans	1 places, 30 € / place
Voiture	1 places, 70 € / place

Prix total : 240.00 €

← 1 →

## D. Administration

Si un utilisateur est un admin alors il verra dans la barre de navigation l'option Dashboard s'afficher.

Pour procéder à la vérification des droits on vérifie les droits via ceux dans Supabase. A savoir qu'un utilisateur crée via le site n'est que User et pas Admin, il faut passer par l'interface Supabase pour donner le rôle admin à un utilisateur.

```
const checkAdminRole = async () => {
  try {
    const {
      data: { user },
    } = await supabase.auth.getUser();
    const UserRole = user?.user_metadata?.role;

    if (UserRole !== "admin") {
      router.push("/");
      return;
    }

    setIsAuthorized(true);
  } catch (error) {
    console.error("Erreur lors de la vérification du rôle:", error);
    router.push("/");
  } finally {
    setIsChecking(false);
  }
};
```

En cliquant dessus il sera redirigé vers le dashboard admin qui permet de visualiser les données sur une période donnée, aussi d'accéder à la partie gestion des liaisons.

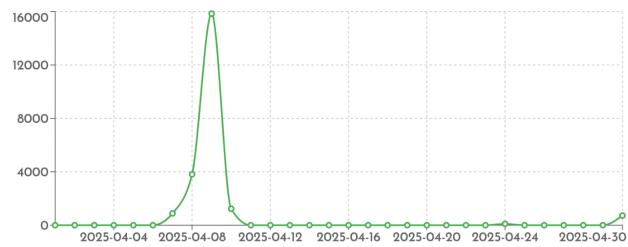
## Tableau de Bord

Date de début 01-04-2025 Date de fin 30-04-2025 Gestion des liaisons

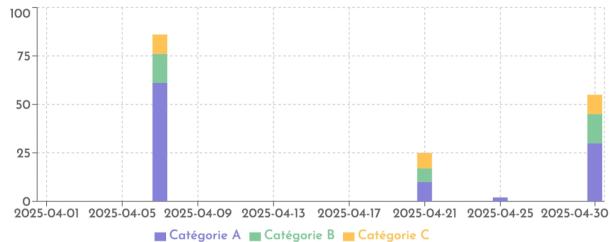
\$ Revenus  
**22580 €**

Passager  
168 passagers  
Cat A: 103  
Cat B: 37  
Cat C: 28

Revenus par date



Passagers par catégorie



Sur cette page on peut modifier la période souhaitée, consulter les revenus et le voir sous forme de graphique, consulter les passages transportés sur la période ainsi que les catégories concernées.

En cliquant sur Gestion des liaison on accède à cette page:

Réserver un trajet Dashboard 🌐

MarieTeam

## Gestion des Liaisons

+ Ajouter une liaison

Liaison 102 - Méditerranée - Marseille → Ajaccio (300 km)

Liaison 101 - Méditerranée - Marseille → Nice (300 km)

Copyright © 2024 MarieTeam

Politiques de Confidentialité

Mentions légales

MarieTeam

Ici on peut ajouter, modifier ou supprimer des liaisons via les différents boutons. Voici l'interface d'ajout:

Réserver un trajet Dashboard 🌐

MarieTeam

## Gestion des Liaisons

Secteur Distance

Départ Arrivée

Ajouter

Annuler

Liaison 102 - Méditerranée - Marseille → Ajaccio (300 km)

Liaison 101 - Méditerranée - Marseille → Nice (300 km)

Copyright © 2024 MarieTeam

Politiques de Confidentialité

Mentions légales

MarieTeam

Il y a une double confirmation pour la suppression afin d'éviter les erreurs:

The screenshot shows a dark-themed web interface for 'Marieteam'. At the top, there's a navigation bar with the logo 'Marieteam' and links for 'Réserver un trajet', 'Dashboard', and a user icon. Below the header, the title 'Gestion des Liaisons' is displayed, along with a green button '+ Ajouter une liaison'. A central modal window is open, titled 'Confirmer la suppression'. It contains the message: 'Êtes-vous sûr de vouloir supprimer la liaison 102 entre Marseille et Ajaccio ?' followed by the note 'Cette action est irréversible.' At the bottom of the modal are two buttons: 'Annuler' (white background) and 'Supprimer' (red background). The footer of the page includes the 'Marieteam' logo, copyright information 'Copyright © 2024 Marieteam', links for 'Politiques de Confidentialité' and 'Mentions légales', and a small 'Ajouter une liaison' button.