# All practice is done in the 18. Wild Oasis project

Supabase

# What is Supabse?

* A service to create a complete back-end with a Postgres database
* Supabase will automatically create the **database** and the **API**
  + We will interact only with the API
* More than just an **API. Also includes**
  + User Authentication
  + File Storage

# Creating a new DB

* Create Account on supabase.com
* New Project

## Model the Application State

* To figure out what tables to we need, we need to think about the app STATE
* We are thinking at **Application Feature Level** for state
* We will store **GLOBAL REMOTE STATE** on Supabse
  + Bookings State
  + Cabins State
  + Guests State
  + Settings State
  + Users State
* These states will be managed in the front end using **React Query**
* For each of these **States** will be a **Table in Supabase**

## The Bookings Table

* Is the table that will manage the connection between **GUESTS** and **CABINS**
* Will store the
  + **guestId**
  + **cabinId**
* These will be connected through the **Foreign Key**

### Table Editor

* Create a new Table
* Define the Columns
* Save 🡺 Good to go

# Accessing the data through the API

* Supabase offers an API
* API Docs
* Install
  + npm install @supabase/supabase-js
* Create Client
  + const SUPABASE\_KEY = SupaBase\_Key();
  + const SUPABASE\_URL = SupaBase\_Url();
  + const supabase = createClient(SUPABASE\_URL, SUPABASE\_KEY);
* We must **create POLICIES**

## RLS - Row level Security

* Prevents everyone who has the KEY to do whatever they want in the DB
* To create a new Policy
  + Authentication 🡪 Policies 🡪 New policy
    - Enable Read access for all users – TEMPLATE

## For now, we create Access policies for all users

## Will update these later only for logged in users

# Storage Buckets

* We can use the **Storage Buckets** to upload large files into Supabase

### Setup

* Storage 🡪 New Bucket
* Drag and Drop
* After upload 🡪 Click on image 🡪 Get URL
  + We can use the URL to render images in our app
* Update the **Cabins Table** with the URL