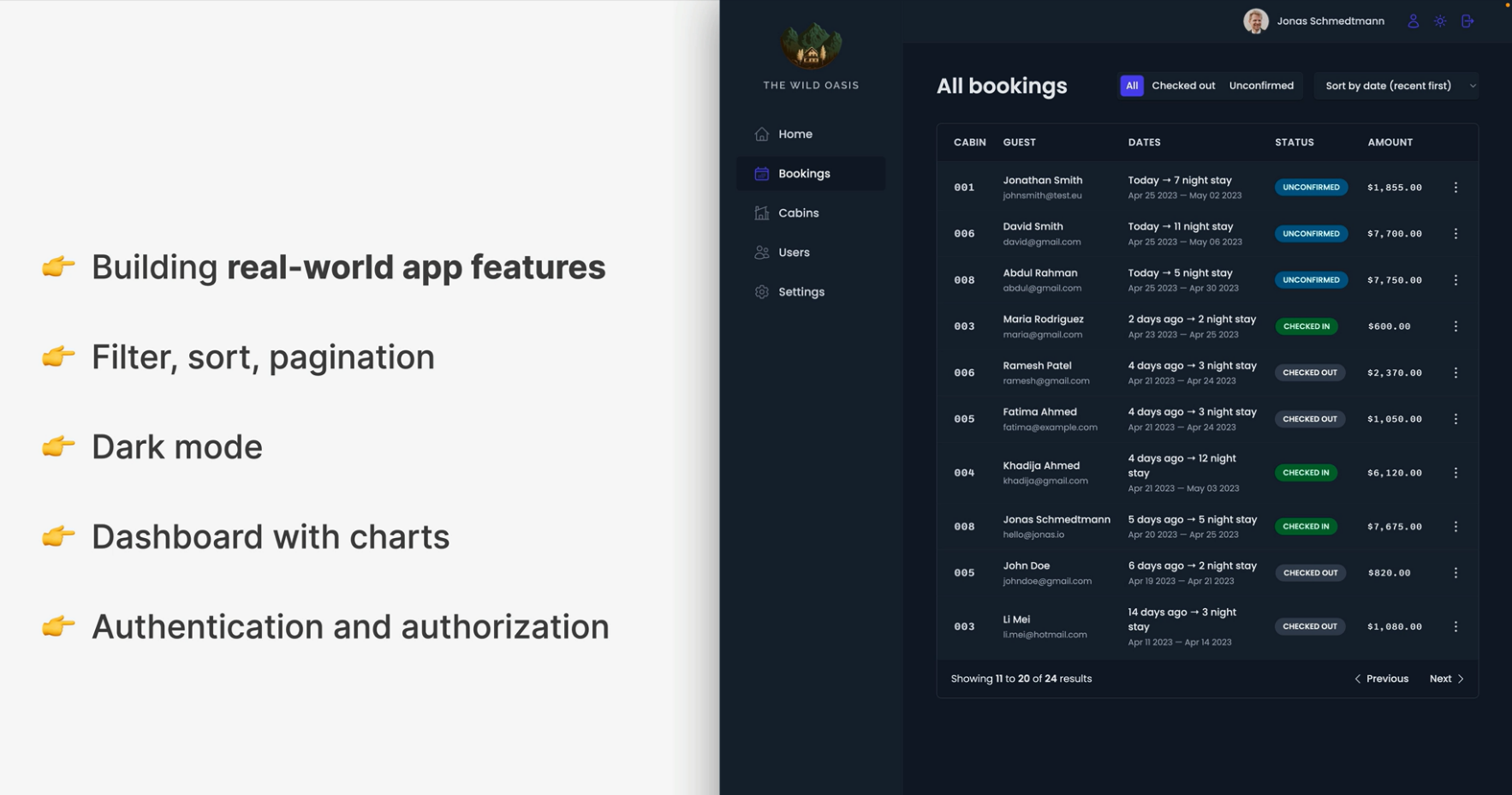
بسم الله الرحمن الرحيم والصلاة والسلام علي اشرف المرسلين سيدنا محمد

Section-Overview



A screenshot of a document

Description automatically generatedRequirements

375- Client-Side Filtering: Filtering Cabins

هنشتغل دلوقتي علي اننا نحط features جديدة وأولهم هو اننا نعمل filter reusable هنستخدمه مع الـ Cabin  
وفي التطبيق كله عموما في المستقبل في أي مكان هيبقي فيه filtration

الـ filtration دا هعمله في الـ URL بحيث يكون قابل للمشاركه مع اي حد وقابل انه يبقي بوك مارك

الـ filtration بتاعي هيكون مكون من زراير واحد All بيعرض الكل والتاني with-discount بيعرض بس اللي فيهم discount والتاني no-discount بيعرض اللي مفيش فيهم discount يلا نبدأ

import { useSearchParams } from "react-router-dom";

import styled, { css } from "styled-components";

const StyledFilter = styled.div`

  border: 1px solid var(--color-grey-100);

  background-color: var(--color-grey-0);

  box-shadow: var(--shadow-sm);

  border-radius: var(--border-radius-sm);

  padding: 0.4rem;

  display: flex;

  gap: 0.4rem;

`;

const FilterButton = styled.button`

  background-color: var(--color-grey-0);

  border: none;

  ${(props) =>

    props.active &&

    css`

      background-color: var(--color-brand-600);

      color: var(--color-brand-50);

    `}

  border-radius: var(--border-radius-sm);

  font-weight: 500;

  font-size: 1.4rem;

  /\* To give the same height as select \*/

  padding: 0.44rem 0.8rem;

  transition: all 0.3s;

  &:hover:not(:disabled) {

    background-color: var(--color-brand-600);

    color: var(--color-brand-50);

  }

`;

const Filter = () => {

  const [searchParams, setSearchParams] = useSearchParams();

  function handleClick(value) {

    searchParams.set("discount", value);

    setSearchParams(searchParams);

  }

  return (

    <StyledFilter>

      <FilterButton onClick={() => handleClick("all")}>All</FilterButton>

      <FilterButton onClick={() => handleClick("no-discount")}>

        No Discount

      </FilterButton>

      <FilterButton onClick={() => handleClick("with-discount")}>

        With Discount

      </FilterButton>

    </StyledFilter>

  );

};

export default Filter;

هنا انا كل زرار حطيت عليه الـ onClick وباصيت الـ Value للـ Function

بعدين استخدمت الـ useSearchParams  
ومنه استخدمت الـ searcParams عشان اعمل set لحاجة اسمها discount وكمان الـ value اللي انا باصيتها فوق للـ URL

بعد كدا باسستخدام الـ setSearchPararms استخدمت الـ searcParams

ثم انتقلت للـ CabinTable.jsx

  const [searchParams] = useSearchParams();

استخدمت تاني الـ searchParams

 const filteredValue = searchParams.get("discount") || "all";

  let filteredCabins;

  if (filteredValue === "all") filteredCabins = cabins;

  if (filteredValue === "no-discount") {

    filteredCabins = cabins.filter((cabin) => cabin.discount === 0);

  }

  if (filteredValue === "with-discount") {

    filteredCabins = cabins.filter((cabin) => cabin.discount > 0);

  }

بعدين عملت ثابت اسمه filteredValue خزنت فيه الـ اللي هجيبه من الـ URL فوق

ثم عملت متغير اسمه filteredCabins وبدأت اقارن

اول if قولتله لو الـ filteredValue اللي انا جبتها فوق بتساوي ‘all’ يبقي ساعتها خلي الـ filteredCabins = cabins

تاني if لو الـ filteredValue بتساوي ‘no-discount’ ساعتها خلي الـ filteredCabins بتساوي الـ cabins بعد ما اعمل عليها filter والشرط بتاعها ان cabin.discount بتساوي 0 مافيش discount

تاني if لو الـ filteredValue بتساوي ‘with-discount’ من الـ URL ساعتها filteredCabins بتساوي cabins بعد ما اعمل عليها filter والشرط ان الـ cabin.discount يكون اكبر من الـ 0

طيب الـ cabins اللي معلم عليها فوق دي جت منين ؟!

جت من الـ useCabins hook

  const { isLoading, cabins } = useCabins();

واللي ببساطة بفيتش الداتا بالـ query

import { useQuery } from "@tanstack/react-query";

import { getCabins } from "../../services/apiCabins";

export const useCabins = () => {

  const { isLoading, data: cabins } = useQuery({

    queryKey: ["cabins"],

    queryFn: getCabins,

  });

  return { isLoading, cabins };

};

بعدها خلي الـ data بتاعتي تكون بدل ما هي الـ cabins لا تكون الـ filteredCabins

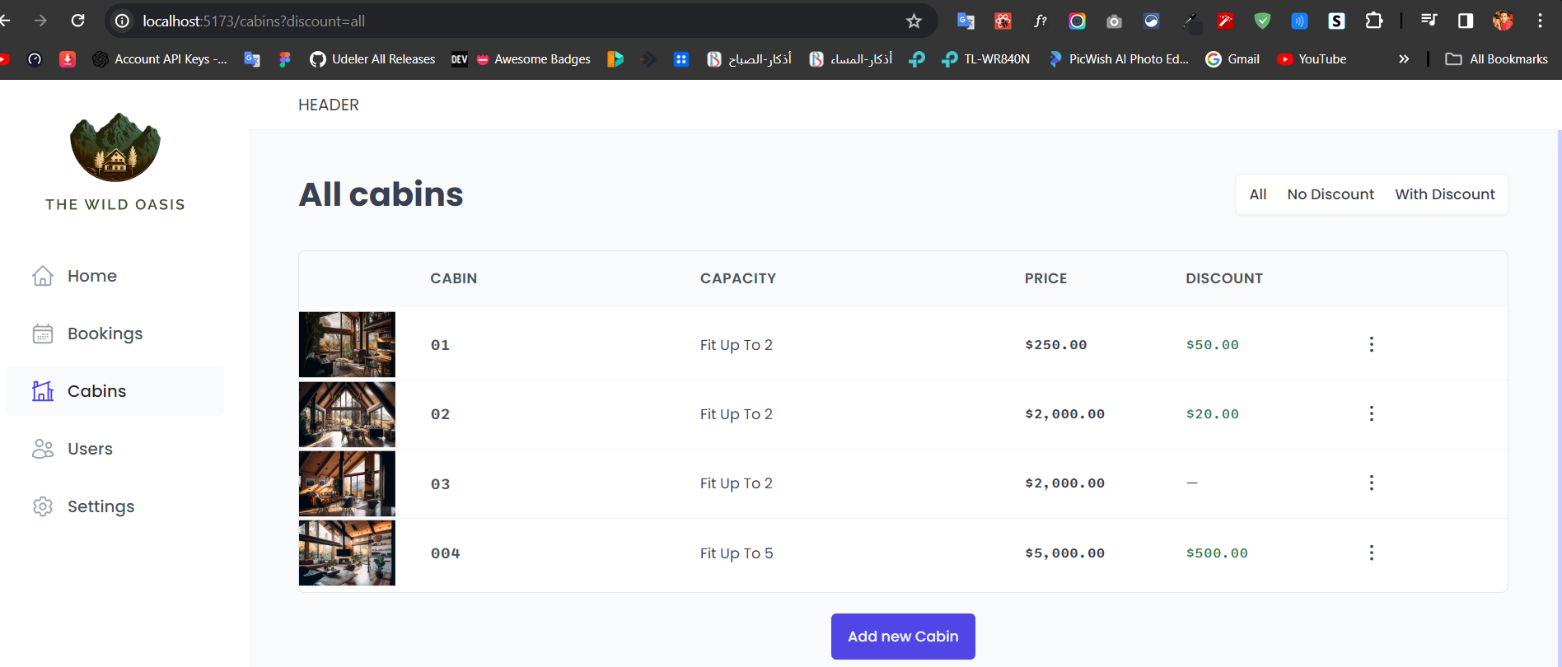
Table.Body

          // data={cabins}

          data={filteredCabins}

          render={(cabin) => <CabinRow cabin={cabin} key={cabin.id} />}

        />



A screenshot of a computer

Description automatically generated

const Filter = ({ filterField, options }) => {

  const [searchParams, setSearchParams] = useSearchParams();

  function handleClick(value) {

    searchParams.set(filterField, value);

    setSearchParams(searchParams);

  }

  return (

    <StyledFilter>

      {options.map((option) => (

        <FilterButton

          key={option.value}

          onClick={() => handleClick(option.value)}

        >

          {option.label}

        </FilterButton>

      ))}

    </StyledFilter>

  );

};

اما بقا الـ Filter.jsx دا عشان يكون more reusable بعتلها حاجتين الـ filterField و الـ options من   
الـ CabinTableOperations.jsx

const CabinTableOperations = () => {

  const options = [

    { value: "all", label: "All" },

    { value: "no-discount", label: "No discount" },

    { value: "with-discount", label: "With discount" },

  ];

  return (

    <TableOperations>

      <Filter filterField="discount" options={options} />

    </TableOperations>

  );

};

وبناءا ع الـ active بقا دي هختار الـ value اللي في الـ url اطابقها بالـ value اللي في option لو true يبقي الزرار اللي تم الضغط عليه بقا active وبالتالي يتعلم عليه دونًا عن الباقي

const FilterButton = styled.button`

  background-color: var(--color-grey-0);

  border: none;

  ${(props) =>

    props.active &&

    css`

      background-color: var(--color-brand-600);

      color: var(--color-brand-50);

    `}

 const currentFilter = searchParams.get(filterField) || options.at(0).value;

 <FilterButton

          key={option.value}

          onClick={() => handleClick(option.value)}

          active={option.value === currentFilter}

        >

376- Client-Side Sorting: Sorting Cabins

 const sortOptions = [

    { value: "name-asc", label: "Sort by name (A-Z) " },

    { value: "name-desc", label: "Sort by name (Z-A) " },

    { value: "regularPrice-asc", label: "Sort by price (low first) " },

    { value: "regularPrice-desc", label: "Sort by price (high price) " },

    { value: "maxCapacity-asc", label: "Sort by capacity (low first) " },

    { value: "maxCapacity-desc", label: "Sort by capacity (high first) " },

  ];

    <TableOperations>

      {/\* <Filter options={filterOptions} filterField="discount" /> \*/}

  <SortBy options={sortOptions} />

    </TableOperations>

import { useSearchParams } from "react-router-dom";

const useUrl = (value) => {

  const [searchParams, setSearchParams] = useSearchParams();

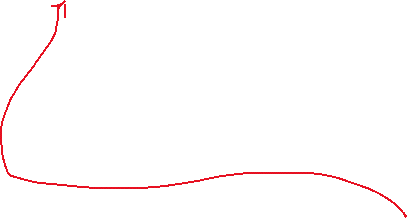
  function handleSearchParams(e) {

    searchParams.set(value, e.target.value);

    setSearchParams(searchParams);

  }

  return { searchParams, handleSearchParams };



};

export default useUrl;

const SortBy = ({ options }) => {

  const { searchParams, handleSearchParams } = useUrl("sortBy");



  const currentSelectedValue = searchParams.get("sortBy") || "";

  return (

    <Select

      value={currentSelectedValue}

      onChange={handleSearchParams}

      options={options}

      type="white"

    />

  );

};

export default SortBy;

الـ currentSelectedValue دي بتيجي لما بسرش في الـ url علي اول param اللي هو الـ sortBy بيجيب النتيجة اللي مرتبطة بيه بقا

import styled from "styled-components";

const StyledSelect = styled.select`

  font-size: 1.4rem;

  padding: 0.8rem 1.2rem;

  border: 1px solid

    ${(props) =>

      props.type === "white"

        ? "var(--color-grey-100)"

        : "var(--color-grey-300)"};

  border-radius: var(--border-radius-sm);

  background-color: var(--color-grey-0);

  font-weight: 500;

  box-shadow: var(--shadow-sm);

`;

const Select = ({ options, onChange, value, ...props }) => {

  return (

    <StyledSelect onChange={onChange} value={value} {...props}>

      {options.map((option) => (

        <option key={option.value} value={option.value}>

          {option.label}

        </option>

      ))}

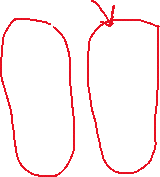
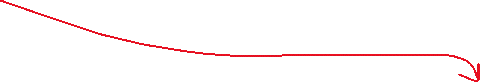
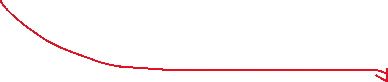
    </StyledSelect>

  );

};

A screenshot of a computer

Description automatically generatedexport default Select;



377 - Building the Bookings Table

379- API-Side Filtering: Filtering Bookings

في أول مرة عملت Filter في الـ Cabins كان من الـ Client side لكن الـ filter بتاع الـ Bookings المرة دي عايز اعمله من الـ API Side

function Bookings() {

  return (

    <>

      <Row type="horizontal">

        <Heading as="h1">All bookings</Heading>

        <BookingTableOperations />

      </Row>

      <BookingTable />

    </>

  );

}

export default Bookings;

BookingTableOperations.jsx

import SortBy from "../../ui/SortBy";

import Filter from "../../ui/Filter";

import TableOperations from "../../ui/TableOperations";

function BookingTableOperations() {

  return (

    <TableOperations>

      <Filter

        filterField="status"

        options={[

          { value: "all", label: "All" },

          { value: "checked-out", label: "Checked out" },

          { value: "checked-in", label: "Checked in" },

          { value: "unconfirmed", label: "Unconfirmed" },

        ]}

      />

      <SortBy

        options={[

          { value: "startDate-desc", label: "Sort by date (recent first)" },

          { value: "startDate-asc", label: "Sort by date (earlier first)" },

          {

            value: "totalPrice-desc",

            label: "Sort by amount (high first)",

          },

          { value: "totalPrice-asc", label: "Sort by amount (low first)" },

        ]}

      />

    </TableOperations>

  );

}

export default BookingTableOperations;

هبدأ في الـ custom Hook بتاعي أعمل Fetch للداتا

import { useQuery } from "@tanstack/react-query";

import { getBookings } from "../../services/apiBookings";

export const useBooking = () => {

  const {

    isLoading,

    error,

    data: bookings,

  } = useQuery({

    queryKey: ["bookings"],

    queryFn: getBookings,

  });

  return { isLoading, error, bookings };

};

بعد كدا استخدمت الـ custom Hook بتاعي اهو في الـ BookingTable.jsx

function BookingTable() {

  const { bookings, isLoading } = useBooking();

  if (isLoading) return <Spinner />;

  if (!bookings.length) return <Empty resourceName="bookings" />;

  return (

    <Menus>

      <Table columns="0.6fr 2fr 2.4fr 1.4fr 1fr 3.2rem">

        <Table.Header>

          <div>Cabin</div>

          <div>Guest</div>

          <div>Dates</div>

          <div>Status</div>

          <div>Amount</div>

          <div></div>

        </Table.Header>

        <Table.Body

          data={bookings}

          render={(booking) => (

            <BookingRow key={booking.id} booking={booking} />

          )}

        />

      </Table>

    </Menus>

  );

}

export default BookingTable;

ثم بعد كدا بعت الـ Data دي للـ BookingRow.jsx

const Cabin = styled.div`

  font-size: 1.6rem;

  font-weight: 600;

  color: var(--color-grey-600);

  font-family: "Sono";

`;

const Stacked = styled.div`

  display: flex;

  flex-direction: column;

  gap: 0.2rem;

  & span:first-child {

    font-weight: 500;

  }

  & span:last-child {

    color: var(--color-grey-500);

    font-size: 1.2rem;

  }

`;

const Amount = styled.div`

  font-family: "Sono";

  font-weight: 500;

`;

function BookingRow({ booking }) {

  const {

    id: bookingId,

    created\_at,

    startDate,

    endDate,

    numNights,

    numGuests,

    totalPrice,

    status,

    guests: { fullName: guestName, email },

    cabins: { name: cabinName },

  } = booking;

  const statusToTagName = {

    unconfirmed: "blue",

    "checked-in": "green",

    "checked-out": "silver",

  };

  return (

    <Table.Row>

      <Cabin>{cabinName}</Cabin>

      <Stacked>

        <span>{guestName}</span>

        <span>{email}</span>

      </Stacked>

      <Stacked>

        <span>

          {isToday(new Date(startDate))

            ? "Today"

            : formatDistanceFromNow(startDate)}{" "}

          &rarr; {numNights} night stay

        </span>

        <span>

          {format(new Date(startDate), "MMM dd yyyy")} &mdash;{" "}

          {format(new Date(endDate), "MMM dd yyyy")}

        </span>

      </Stacked>

      <Tag type={statusToTagName[status]}>{status.replace("-", " ")}</Tag>

      <Amount>{formatCurrency(totalPrice)}</Amount>

    </Table.Row>

  );

}

export default BookingRow;

Tag.jsx

import styled from "styled-components";

const Tag = styled.span`

  width: fit-content;

  text-transform: uppercase;

  font-size: 1.1rem;

  font-weight: 600;

  padding: 0.4rem 1.2rem;

  border-radius: 100px;

  /\* Make these dynamic, based on the received prop \*/

  color: var(--color-${(props) => props.type}-700);

  background-color: var(--color-${(props) => props.type}-100);

`;

export default Tag;

نرجع لموضوعنا الاساسي وهي عملة الـ Filter بقا

دلوقتي عشان اعمل filter من خلال الـ api فانا عندي method اسمها eq من خلالها اقدر افلتر بناءا علي اسم   
الـ Filter اللي هو status وبناءا ع الـ value بتاعته

عشان أوصل للـ status دي وأوصل للـ value بتاعته محتاج استخدم الـ useSearcParams hook

وافضل مكان لاستخدامه هو في الـ custom hook بتاعي الـ useBookings.js

لسببين

السبب الاول اني مقدرش اسشتخدم الـ useSearcParams دا في الـ apiBookings لانه regular functions

السبب التاني انه الـ useBookings.js هيكون reusable اقدر استخدمه في اي مكان بالتالي لما يكون جواه   
الـ useSearcParams يبقي هقدر استخدمها هي كمان من خلاله في أي مكان

السبب التالت انه الـ regular function اللي عايز استخدم فيها الـ useSearcParams بس مش هقدر اعمل دا جواها بشكل مباشر هي اصلا موجود في الـ useBookings.js بالتالي انا ممكن اباصي الـ values ليها من الـ custom hook دا واستخدمهم هناك بعد ما اباصيهم طيب تعالي نعمل implementation

export const useBooking = () => {

  const [searchParams] = useSearchParams();

  //1) FILTER

  const filteredValue = searchParams.get("status");

  const filter =

    !filteredValue || filteredValue === "all"

      ? null

      : { field: "status", value: filteredValue };

  console.log(filter);

  const {

    isLoading,

    error,

    data: bookings,

  } = useQuery({

    queryKey: ["bookings"],

    queryFn: () => getBookings({ filter }),

  });

  return { isLoading, error, bookings };

};

الـ filteredValue دي قيمتها بقت قيمة الـ Status يعني لو هو unconfirmed أو all يبقي دي قيمته متخزنة في   
الـ filteredValue دي

تاني حاجة عملت ثابت اسمه filter قولتله فيه والله لو الـ filteredValue! مش موجودة أو بتساوي all لو دا true ساعتها رجعلي null لو غير كدا اعملي object مكون من field قيمته بتساوي ‘status’ و value قيمتها بتساوي قيمة الـ filteredValue دي

   queryFn: () => getBookings({ filter }),

وبعدين باصي للـ getBookings api function الـ filter const دا

تعالي بقا للـ getBookings func بتاعتي دي

export async function getBookings({ filter }) {

  console.log(filter);

  const query = supabase

    .from("bookings")

    .select(

      "id, created\_at, startDate, endDate, numNights, numGuests, status, totalPrice, cabins(name), guests(fullName, email)"

    );

  if (filter) query = query.eq(filter.field, filter.value);

  let { data, error } = await query;

  if (error) {

    console.error(error);

    throw new Error("Bookings couldn't be loaded");

  }

  return data;

}

انا باعتلها object المفروض اللي هو دا

 { field: "status", value: filteredValue }

بس دا في حالة انه مفيش filtered Value وكمان ان القيمة دي مش بتساوي all

عملتله destructure

وقلتله لو الـ filter object مش بـ null اللي هو الشرط اللي هناك فوق يبقي دا معناه ان الـ filter قيمته هي   
الـ object اللي فوق دا

اذا بناءا عليه استخدم الـ eq method دي واللي اول value فيها هتكون بالـ filter.field اللي هو ‘status’ تاني

قيمة فيه هتكون الـ filter.value اللي هي ممكن تبقي unconfirmed او check-in أو أو وهكذا

if (filter) query = query.eq(filter.field, filter.value);

لكن بالنسبة للجزء دا بقا

const query = supabase

    .from("bookings")

    .select(

      "id, created\_at, startDate, endDate, numNights, numGuests, status, totalPrice, cabins(name), guests(fullName, email)"

    );

فانا عملت المتغير query دا عشان اخد النتيجة اللي بعده دي كلها اكمل عليها بالـ eq method

وعشان لو معملتش كدا هيبقي صعب احط الـشرط بتاعي دا

if (filter) query = query.eq(filter.field, filter.value);

اما الجزء دا عشان يعمل Refatcing automatic بناءا علي الـ filteredValue حاجة كدا من ريحة الـ useEffect

  const { isLoading, error, data: bookings } = useQuery({

    queryKey: ["bookings", filteredValue],

    queryFn: () => getBookings({ filter }),

  });

انا كمان عشان اخلي الموضوع more flexible مش هكتب الـ eq دي Hard coded كدا لا

  const filter =

    !filteredValue || filteredValue === "all"

      ? null

      : { field: "status", value: filteredValue, method: "eq" };

انا هبعت حاجة اسمها method في الـ filter برضو

if (filter !== null) query[filter.method || 'eq'](filter.field, filter.value);

وهستلمها بالشكل دا كدا

380- API-Side Sorting: Sorting Bookings

function BookingTableOperations() {

  return (

    <TableOperations>

      <Filter

        filterField="status"

        options={[

          { value: "all", label: "All" },

          { value: "checked-out", label: "Checked out" },

          { value: "checked-in", label: "Checked in" },

          { value: "unconfirmed", label: "Unconfirmed" },

        ]}

      />

      <SortBy

        options={[

          { value: "startDate-desc", label: "Sort by date (recent first)" },

          { value: "startDate-asc", label: "Sort by date (earlier first)" },

          {

            value: "totalPrice-desc",

            label: "Sort by amount (high first)",

          },

          { value: "totalPrice-asc", label: "Sort by amount (low first)" },

        ]}

      />

    </TableOperations>

  );

}

const SortBy = ({ options }) => {

  const { searchParams, handleSearchParams } = useUrl("sortBy");

  const currentSelectedValue = searchParams.get("sortBy") || "";

  return (

    <Select

      value={currentSelectedValue}

      onChange={handleSearchParams}

      options={options}

      type="white"

    />

  );

};

export default SortBy;

زي ما قولنا الدرس اللي فات انه انا بعمل الـ Filter من الـ API SIDE والمرادي عايز اعمل الـ Sort كذلك من الـ API

فاللي فوق دول الـ Components المبدأية

وزي ما قولت عشان مقدرش استخدم الـ useSearchParams في الـ regular function وان افضل مكان حاليا هو استخدامها في الـ custom hook بالتالي انا هطبق عملية الـ Sort في الـ custom hook دا وابعته للـ function  
اللي فيها الـ API

useBookings.js

  //1) SORT

  const sortByRow = searchParams.get("sortBy") || "startDate-desc";

  const [field, direction] = sortByRow.split("-");

  const sortBy = { field, direction };

اي اللي حصل ؟

اولا الـ SortByRow دا قيمته هتبقي حاجة من الاربعة التابعين للـ sortBy Param دا واللي هما

startDate-desc - startDate-asc - totalPrice-desc - totalPrice-asc

وعشان كل واحد من دول مكون من مقطعين بيفصل بينهم dash –

فانا هعمل متغيرين اول واحد وهو الـ field و التاني direction كل واحد فيهم هيحمل قيمة معينة  
فهستخدم الـ split علي الـ SortByRow دي عشان اول النتيجة لـ array من قمتين الفاصل بينهم Dash -   
اول قيمة هتروح للـ field اللي هي مثلا startData و التانية هتكون للـ direction واللي هي مثلا desc



  console.log(sortByRow.split("-"));

A blue text on a black background

Description automatically generated

بعد كدا السطر دا

  const sortBy = { field, direction };

عملت متغير اسمه sortBy حولت الـ array الي فوق دي لـ object يبقي قيمة الـ SortBy دي بقت object  
شايل key و value

const {

    isLoading,

    error,

    data: bookings,

  } = useQuery({

    queryKey: ["bookings", filteredValue, sortBy],

    queryFn: () => getBookings({ filter, sortBy }),

  });

  return { isLoading, error, bookings };

وبعدين بعت الـ SortBy دا للـ function بتاعت الـ getBookings دي زيها زي الـ filter بالظبط

export async function getBookings({ filter, sortBy }) {

  let query = supabase

    .from("bookings")

    .select(

      "id, created\_at, startDate, endDate, numNights, numGuests, status, totalPrice, cabins(name), guests(fullName, email)"

    );

  // FILTER

  if (filter) query = query[filter.method || "eq"](filter.field, filter.value);

  // SORT

  if (sortBy)

    query = query.order(sortBy.field, {

      ascending: sortBy.direction === "asc",

    });

  let { data, error } = await query;

  if (error) {

    console.error(error);

    throw new Error("Bookings couldn't be loaded");

  }

  return data;

}

381- Building a Reusable Pagination Component

        <Table.Footer>

          <Pagination count={50} />

        </Table.Footer>

const Pagination = ({ count }) => {

  function next() {}

  function prev() {}

  return (

    <StyledPagination>

      <P>

        Showing <span>1</span> to <span>10</span> of <span>{count}</span>

      </P>

      <Buttons>

        <PaginationButton>

          <HiChevronLeft />

          <span>Previous</span>

        </PaginationButton>

        <PaginationButton>

          <span>Next</span>

          <HiChevronRight />

        </PaginationButton>

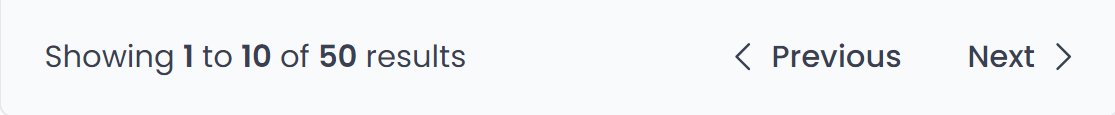
      </Buttons>

    </StyledPagination>

  );

};

export default Pagination;



طيب عشان بشكل منطقي أجيب الـ next page او ارجع للـ previous page لازم اكون عارف الـ current page

وعشان اعرف الـ current page هستخدم الـ useSearchParams

  //1) Get Current Page

  /\* calculating the next page or the previous page

  will, of course, always depend on the current page.

  next lines of code means if there is no current page this means i'm already in first page but if there is current page I should guarantee it's a number \*/

  const currentPage = !searchParams.get("page")

    ? 1

    : Number(searchParams.get("page"));

انا هنا بقوله المتغير currentPage دا لو الـ page param مش موجود اعمله بـ 1 ولو موجود خليه رقم  
بمعني اوضح بقوله لـ page مش موجودة يبقي دي أول صفحة يبقي الـ current page بتساوي 1   
لكن لو موجودة انا عايزها ديما تكون رقم وتكون دي قيمة الـ currentPage

بعد ما جبت الـ currently page محتاج اعرف عندي اصلا كام page

// Displaying how many results in every page

const PAGE\_SIZE = 10;

  //2) Knowing Page Count

  /\* it's depending on how many results you have

  if u have 50 results and every page fit to 10 results only

  this means u need 5 pages to display these results\*/

  const pageCount = Math.ceil(count / PAGE\_SIZE);

يعني لو عندي 50 نتيجة مقسومين علي عدد النتايج في الصفحة الواحدة يطلعلي نتيجة عدد الصفحات الاجمالي يعني 50 نتيجة وكل صفحة 10 نتايج يبقي انا محتاج 5 صفح لعرض الـ 50 نتيجة دول   
وطبعا استخدمت الـ Math.cile هنا عشان لو الـ Resutls او النتايج عندي 55 نتيجة مقسومين علي 10 نتايج في كل صفحة فدا معناه اني اعندي 5.5 صفحة طيب هو فيه 5.5 صفحة ؟ لا طبعا الطبيعي تبقي 6 صفحات

  function next() {

    /\* if current page 5 and i have only 5 pages (on other word we already in the last page)

     => so make us in this page without moving, In contrast move us to next page \*/

    const next = currentPage === pageCount ? currentPage : currentPage + 1;

    searchParams.set("page", next);

    setSearchParams(searchParams);

  }

,

<PaginationButton disabled={currentPage === pageCount} onClick={nextPage}>

   <span>Next</span>

  <HiChevronRight />

 </PaginationButton>

طيب معني الكود دا اني بقوله المتغير next هو اللي هيبقي شايل رقم الصفحة الحالي

وبقوله لو الـ currentPage ==== pageCount فدا معناه اني انا في اخر صفحة خلاص   
بالتالي لو دا حصل يبقي ديما خليني في أخر صفحة خلي الـ next دا قيمته هي الـ currently page بدون تغير   
ولكن لو مش في أخر صفحة فدا معناه اني عايز ازود علي الـ current page دي 1

بعد كدا قولتلها set في الـ query فوق حاجة اسمها page وقميته معتمدة ع الـ next number اللي هو معتمد   
علي الـ pageCount و الـ currentPage في نفس الوقت

بعدين set الحوار دا كله بالـ setSearchParams

  function prevPage() {

    const prev = currentPage === 1 ? currentPage : currentPage - 1;

    searchParams.set("page", prev);

    setSearchParams(searchParams);

  }

<PaginationButton disabled={currentPage === 1} onClick={prevPage}>

  <HiChevronLeft />

  <span>Previous</span>

</PaginationButton>

نفس الكلام انا بقوله لو الـ currentPage === 1 فدا معناه اني في أخر صفحة ساعتها خليني فيها   
لو العكس يبقي نقصلي من الـ currentPage دي 1 ونفس الحوارات الباقية

      <P>

        Showing <span>{(currentPage - 1) \* PAGE\_SIZE + 1}</span> to{" "}

        <span>

          {currentPage === pageCount ? count : currentPage \* PAGE\_SIZE}

        </span>{" "}

        of <span>{count}</span> results

      </P>

وهنا عشان اكمل اللوجيك صح الصح

قولتله ان يعرض الـ currentPage – 1 دا معناه انها لو 1 تبقي 0 بعدين الـ PAGE\_SIZE اللي هو 10 + 1  
فتبقي 0 x 10 بتساوي 0 + 1 تساوي 1

ولما تبقي الـ currentPage بتساوي 2 فتبقي 2 – 1 بتساوي 1 \* 10 + 1 بتساوي 11

اما تاني جزء بقوله لو الـ currentPage === pageCount اي اننا في أخر صفحة خلاص   
ساعتها رجعلي الـ count ايا كان رقمه الحالي لنفترض انه 45   
لو احنا مش في أخر صفحة يبقي الـ currentPage اللي هو مثلا 3 مضروب في PAGE\_SIZE اللي هو 10 يبقي بيساوي 30

382 - API-Side Pagination: Paginating Bookings

constants.js

export const PAGE\_SIZE = 10;

apiBookings.js

  let query = supabase

    .from("bookings")

    .select(

      "id, created\_at, startDate, endDate, numNights, numGuests, status, totalPrice, cabins(name), guests(fullName, email)",

      { count: "exact" }

    );

  let { data, error, count } = await query;

 return { data, count };

useBookings.js

  // PAGINATION

  const page = !searchParams.get("page") ? 1 : Number(searchParams.get("page"));

  const {

    isLoading,

    error,

    data: { data: bookings, count } = {},

  } = useQuery({

    queryKey: ["bookings", filteredValue, sortBy, page],

    queryFn: () => getBookings({ filter, sortBy, page }),

  });

  return { isLoading, error, bookings, count };

BookingTable.jsx

  const { bookings, isLoading, count } = useBooking();

  <Pagination count={count} />

A screen shot of a computer

Description automatically generated

apiBookings.js

  // PAGINATION

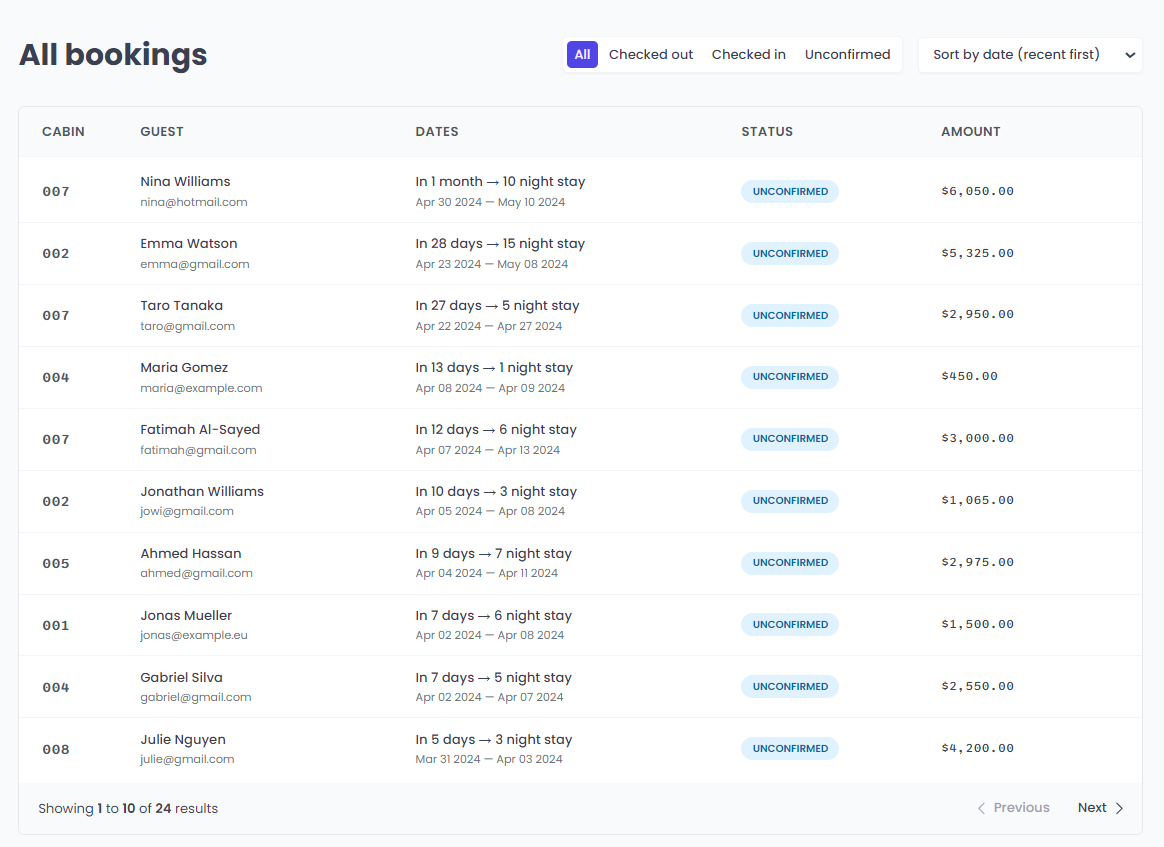
  if (page) {

    const from = (page - 1) \* PAGE\_SIZE; //0 index

    const to = from + PAGE\_SIZE - 1; //9 index

    query = query.range(from, to);

  }



query.range(from(0 index), to (9 index)) means display 10 results.

count

383- Prefetching With React Query

what actually is prefetching?

**Well, prefetching is all about fetching some data that we know might become necessary before we actually need that data to render it on the user interface. And in the context of pagination, usually that means that we fetch the next page before it is actually displayed.**

الـ prefetching عملية بقدر من خلالها اني استخدم داتا انا عارف انها هتبقي ضرورية مستقبلا لذلك هجهزها لحين عرضها

import { PAGE\_SIZE } from "../../utils/constants"; //(10 results)

  const queryCLint = useQueryClient();

  // PRE-FETCHING

  const pageCount = Math.ceil(count / PAGE\_SIZE);

  if (page < pageCount)

    queryCLint.prefetchQuery({

      queryKey: ["bookings", filteredValue, sortBy, page + 1],

      queryFn: () => getBookings({ filter, sortBy, page: page + 1 }),

    });

  if (page > 1)

    queryCLint.prefetchQuery({

      queryKey: ["bookings", filteredValue, sortBy, page - 1],

      queryFn: () => getBookings({ filter, sortBy, page: page - 1 }),

    });