

M6UF4PF

Ch1 - Getting Started

```
PS C:\Users\Jcarl\Documents\M06-UF4-PF> npm install -g pnpm
```

```
PS C:\Users\Jcarl\Documents\M06-UF4-PF> npx create-next-app@latest nextjs-dashboard --example "https://github.com/vercel/next-learn/tree/main/dashboard/starter-example" --use-pnpm
```

```
PS C:\Users\Jcarl\Documents\M06-UF4-PF> cd .\nextjs-dashboard\
```

```
PS C:\Users\Jcarl\Documents\M06-UF4-PF\nextjs-dashboard> pnpm i
```

```
PS C:\Users\Jcarl\Documents\M06-UF4-PF\nextjs-dashboard> pnpm i
Lockfile is up to date, resolution step is skipped
Already up to date
```

Warning

Ignored build scripts: bcrypt, sharp.
Run "pnpm approve-builds" to pick which dependencies should be allowed to run scripts.

```
PS C:\Users\Jcarl\Documents\M06-UF4-PF\nextjs-dashboard> pnpm dev
```

```
> @ dev C:\Users\Jcarl\Documents\M06-UF4-PF\nextjs-dashboard
> next dev --turbo
```

```
▲ Next.js 15.1.6 (Turbopack)
- Local:      http://localhost:3000
- Network:    http://192.168.56.1:3000
```

```
✓ Starting...
✓ Ready in 1951ms
○ Compiling / ...
✓ Compiled / in 2.7s
GET / 200 in 3235ms
```

localhost:3000

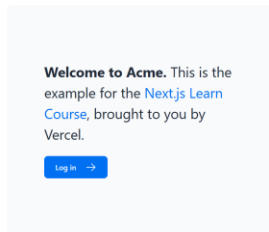
localhost:3000

Welcome to Acme. This is the example for the [Next.js Learn Course](https://github.com/vercel/next-learn/tree/main/dashboard/starter-example), brought to you by Vercel.

[Log in](#)

Ch2 - CSS Styling

```
layout.tsx M X
nextjs-dashboard > app > layout.tsx > RootLayout
1 | import '@app/ui/global.css';
```



```
# global.css 3 X
nextjs-dashboard > app > ui > # global.css
1 | @tailwind base;
2 | @tailwind components;
3 | @tailwind utilities;
```

```
page.tsx X
nextjs-dashboard > app > page.tsx > ...
1 | import AcmeLogo from '@app/ui/acme-logo';
2 | import { ArrowRightIcon } from '@heroicons/react/24/outline';
3 | import Link from 'next/link';
4 |
5 | export default function Page() {
6 |   return (
7 |     <main className="flex min-h-screen flex-col p-6">
8 |       <div className="flex h-20 shrink-0 items-end rounded-lg bg-blue-500 p-4 md:h-52">
9 |         </> <AcmeLogo /> </div>
10 |       </div>
11 |       <div className="mt-4 flex grow flex-col gap-4 md:flex-row">
12 |         <div className="flex flex-col justify-center gap-6 rounded-lg bg-gray-50 px-6 py-10 md:w-2/5 md:px-20">
13 |           <p className="text-xl text-gray-800 md:text-3xl md:leading-normal">
14 |             <strong>Welcome to Acme.</strong> This is the example for the(' ')
15 |             <a href="https://nextjs.org/learn/" className="text-blue-500">
16 |               Next.js Learn Course
17 |             </a>
18 |           , brought to you by Vercel.
19 |         </p>
20 |         <Link
21 |           href="/login"
22 |           className="flex items-center gap-5 self-start rounded-lg bg-blue-500 px-6 py-3 text-sm font-medium text-white transition-colors hover:bg-blue-400 md:text-base"
23 |         >
24 |           <span>Log in</span> <ArrowRightIcon className="w-5 md:w-6" />
25 |         </Link>
26 |       </div>
27 |       <div className="flex items-center justify-center p-6 md:w-3/5 md:px-28 md:py-12">
28 |         </> Add Hero Images Here </div>
29 |     </div>
30 |   </main>
31 | );
32 |
33 | }
```

What shape do you see when using the code snippet above?

C

A black triangle

✓ Correct

The border class names are used to create a triangle shape.

```
# home.module.css U X
nextjs-dashboard > app > ui > # home.module.css > .shape
1  .shape {
2    height: 0;
3    width: 0;
4    border-bottom: 30px solid black;
5    border-left: 20px solid transparent;
6    border-right: 20px solid transparent;
7  }
```

```
page.tsx M X
nextjs-dashboard > app > page.tsx > Page
1  import AcmeLogo from '@app/ui/acme-logo';
2  import { ArrowRightIcon } from '@heroicons/react/24/outline';
3  import Link from 'next/link';
4  import styles from '@app/ui/home.module.css';
5
6  export default function Page() {
7    return (
8      <main className="flex min-h-screen flex-col p-6">
9      <div className={styles.shape} />
```

What is one benefit of using CSS modules?

B

Provide a way to make CSS classes locally scoped to components by default, reducing the risk of styling conflicts.

✓ Correct

CSS Modules create unique class names for each component, so you don't have to worry about style collisions.

```
status.tsx X
nextjs-dashboard > app > ui > invoices > status.tsx > InvoiceStatus
1  import { CheckIcon, ClockIcon } from '@heroicons/react/24/outline';
2  import clsx from 'clsx';
3
4  export default function InvoiceStatus({ status }: { status: string }) {
5    return (
6      <span
7        className={clsx(
8          'inline-flex items-center rounded-full px-2 py-1 text-xs',
9          {
10           'bg-gray-100 text-gray-500': status === 'pending',
11           'bg-green-500 text-white': status === 'paid',
12         },
13       )}
14    >
```

Search for "clsx" in your code editor, what components use it to conditionally apply class names?

A

'status.tsx' and 'pagination.tsx'

✓ Correct

The 'status.tsx' and 'pagination.tsx' components use 'clsx' to conditionally apply class names.

Ch3 - Optimizing Fonts and Images

How does Next.js optimize fonts?

D

It hosts font files with other static assets so that there are no additional network requests.

✓ Correct

Next.js downloads font files at build time and hosts them with your other static assets. This means when a user visits your application, there are no additional network requests for fonts which would impact performance.

TS fonts.ts U

nextjs-dashboard > app > ui > TS fonts.ts > ...

```
1 import { Inter } from 'next/font/google';
2
3 export const inter = Inter({ subsets: ['latin'] });
```

layout.tsx

app > layout.tsx > RootLayout

```
1 import '@app/ui/global.css';
2 import { inter } from '@app/ui/fonts';
3
4 export default function RootLayout({
5   children,
6 }): {
7   children: React.ReactNode;
8 } {
9   return (
10    <html lang="en">
11      <body className={` ${inter.className} antialiased`}>{children}</body>
12    </html>
13  );
14 }
```

```

TS fonts.ts X
app > ui > TS fonts.ts > ...
1  import { Inter, Lusitana } from 'next/font/google';
2
3  export const inter = Inter({ subsets: ['latin'] });
4
5  export const lusitana = Lusitana({
6    weight: ['400', '700'],
7    subsets: ['latin'],
8  });

```

⚙️ page.tsx

```
import styles from '@app/ui/home.module.css';
```

```

<p
  className={` ${lusitana.className} text-xl text-gray-800 md:text-3xl md:leading-normal` }
></p>

```

```
<AcmeLogo />
```

```
import Image from 'next/image';
```

```

<div className="flex items-center justify-center p-6 md:w-3/5 md:px-28 md:py-12">
  { /* Add Hero Images Here */ }
  <Image
    src="/hero-desktop.png"
    width={1000}
    height={760}
    className="hidden md:block"
    alt="Screenshots of the dashboard project showing desktop version"
  />
  <Image
    src="/hero-mobile.png"
    width={560}
    height={620}
    className="block md:hidden"
    alt="Screenshot of the dashboard project showing mobile version"
  />

```



You've Completed Chapter 3

You've learned how to optimize fonts and images using Next.js.

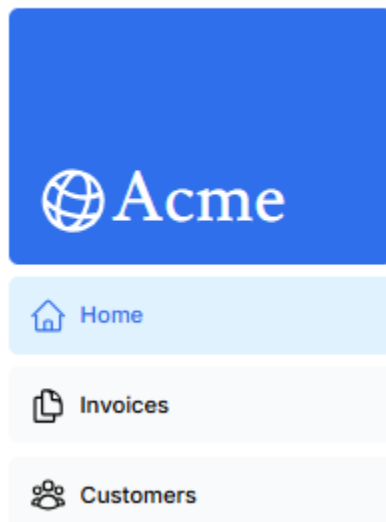
Ch4 - Creating Layouts and Pages

```
page.tsx X
app > dashboard > page.tsx > Page
1 export default function Page() {
2   return <p>Dashboard Page</p>;
3 }

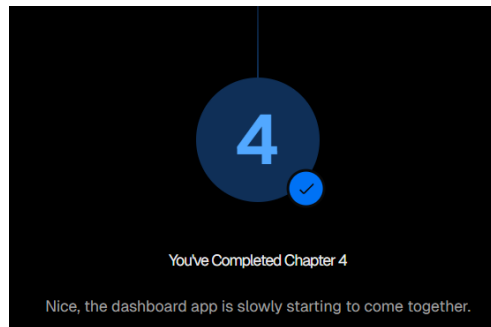
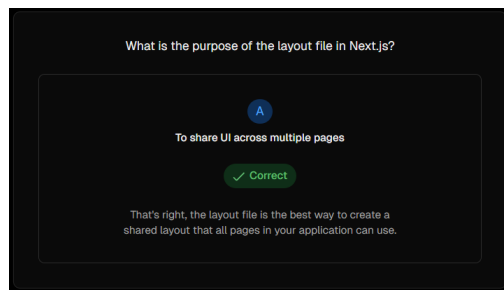
app
├── dashboard
│   ├── customers
│   │   └── page.tsx
│   ├── invoices
│   │   └── page.tsx
│   └── layout.tsx
└── page.tsx

layout.tsx X
app > dashboard > layout.tsx > Layout
1 import SideNav from '@app/ui/dashboard/sidenav';
2
3 export default function Layout({ children }: { children: React.ReactNode }) {
4   return (
5     <div className="flex h-screen flex-col md:flex-row md:overflow-hidden">
6       <div className="w-full flex-none md:w-64">
7         <SideNav />
8       </div>
9       <div className="flex-grow p-6 md:overflow-y-auto md:p-12">{children}</div>
10    </div>
11  );
12 }
```

< > ↺ 🌐 localhost:3000/dashboard



Dashboard Page

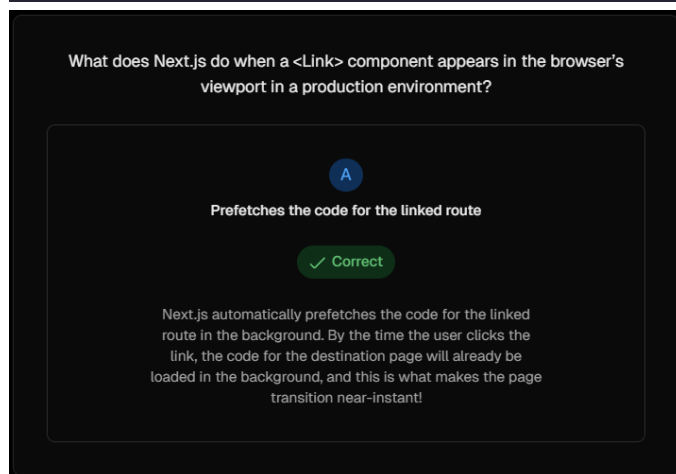


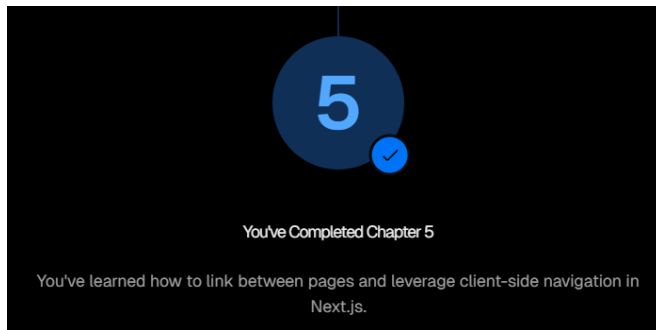
Ch5 - Navigating Between Pages

app > ui > dashboard > nav-links.tsx

```
'use client';
import {
  UserGroupIcon,
  HomeIcon,
  DocumentDuplicateIcon,
} from '@heroicons/react/24/outline';
import Link from 'next/link';
import { usePathname } from 'next/navigation';
import clsx from 'clsx';
```

```
<Link
  key={link.name}
  href={link.href}
  className={clsx(
    'flex h-[48px] grow items-center justify-center gap-2 rounded-md bg-gray-50 p-3 text-sm font-medium hover:bg-sky-100 hover:text-blue-600 md:flex-none md:justify-start md:p-2 md:px-3',
    {
      'bg-sky-100 text-blue-600': pathname === link.href,
    },
  )}
>
  <LinkIcon className="w-6" />
  <p className="hidden md:block">{link.name}</p>
</Link>
```





Ch6- Setting Up Your Database

New Project

Importing from GitHub
JCARLO/nextjs-dashboard ↗ master

Choose where you want to create the project and give it a name.

Vercel Team: JCARLO's projects Hobby ▾
Project Name: nextjs-dashboard

Framework Preset: Next.js ▾

Root Directory: ./ Edit

> Build and Output Settings

> Environment Variables

Deploy


vercel.com/jcarlo's-projects/nextjs-dashboard/7KtpooqafpXUQ36jN8k8pD4MMMD

JCARLO's projects Hobby / nextjs-dashboard / 7Ktpooqafp

Deployment Logs Resources Source Open Graph

Deployment Details

Share Visit ...



Created: JCARLO 7m ago
Status: Ready Latest
Time to Ready: 39% 6m ago
Environment: Production Current

Domains:
nextjs-dashboard-sand-ru-16.vercel.app →
nextjs-dashboard-gt-master-jcarlo's-projects.vercel.app
nextjs-dashboard-q8t9lqgr-jcarlo's-projects.vercel.app

Source:
master
6367881 updated bycrypt

> Deployment Configuration Fluid Compute Deployment Protection Skew Protection

> Build Logs 1m 11s ✓

> Deployment Summary Resources ✓

> Assigning Custom Domains 1s ✓

Runtime Logs View and debug runtime logs & errors

Observability Monitor app health & performance

Web Analytics Not Enabled Analyze visitors & traffic in real-time

Speed Insights Not Enabled Performance metrics from real users

▲

JCARLO's projects

hibby

nextjs-dashboard

Ship tickets →FeedbackChangelogHelpDocs

ProjectDeploymentsAnalyticsSpeed InsightsLogsObservabilityFirewallStorageFlagsAISettings

Storage

Read and write directly to databases and stores from your projects. [Learn more](#)

Connect to a Database

You can connect to existing Databases from this team, or create a new one and connect it to this project.

Q Search provider or database.

Q All

Create Database

Browse Storage

Create databases and stores that you can connect to your projects.

Create New

Select Existing

Edge Config

Ultra-low latency reads

Blob

Fast object storage

KV and Postgres are now available through the Marketplace.

Learn more

Marketplace Database Providers

Learn more

Neon

Serverless Postgres

Upstash

Serverless DB (Redis, Vector, Queue)

Supabase

Postgres backend

Redis

Serverless Redis

Nile

Postgres re-engineered for B2B

Cancel

Continue

Create Database

Set up your new Neon Database, powered by Neon

Database Name (*)

nextjs-dashboard

This is how we call your Database in the Vercel dashboard and in Neon.

Your Selection

Parameters

Region

Frankfurt, Germany (West)

fra1

Free

Always-available free tier, no credit card required.

Maximum projects

10

Maximum branches

10

Maximum time on non-primary branch

5.00 hours

Back

Create

Create Database

Set up your new Neon Database, powered by Neon

Region

Frankfurt, Germany (West)

Choose your database region

Installation Plans

Expand All

Free

Free

Always-available free tier, no credit card required.

Launch

\$19.00/month

The resources, features, and support you need to launch.

Scale

\$69.00/month

More capacity and functionality for scaling production workloads.

Business

\$700.00/month

For larger workloads, partners, and best compliance/security.

Go Back

Continue

```

.env
1 # Recommended for most uses
2 DATABASE_URL=postgres://neondb_owner:npg_cr2Y2ypIT91f@ep-orange-voice-a27hgqmk-pooler.eu-central-1.aws.neon.tech/neondb?sslmode=require
3
4 # For uses requiring a connection without pgbouncer
5 DATABASE_URL_UNPOOLED=postgres://neondb_owner:npg_cr2Y2ypIT91f@ep-orange-voice-a27hgqmk.eu-central-1.aws.neon.tech/neondb?sslmode=require
6
7 # Parameters for constructing your own connection string
8 PGHOST=ep-orange-voice-a27hgqmk-pooler.eu-central-1.aws.neon.tech
9 PGHOST_UNPOOLED=ep-orange-voice-a27hgqmk.eu-central-1.aws.neon.tech
10 PGUSER=neondb_owner
11 PGDATABASE=neondb
12 PGPASSWORD=npg_cr2Y2ypIT91f
13
14 # Parameters for Vercel Postgres Templates
15 POSTGRES_URL=postgres://neondb_owner:npg_cr2Y2ypIT91f@ep-orange-voice-a27hgqmk-pooler.eu-central-1.aws.neon.tech/neondb?sslmode=require
16 POSTGRES_URL_NON_POOLING=postgres://neondb_owner:npg_cr2Y2ypIT91f@ep-orange-voice-a27hgqmk.eu-central-1.aws.neon.tech/neondb?sslmode=require
17 POSTGRES_USER=neondb_owner
18 POSTGRES_HOST=ep-orange-voice-a27hgqmk-pooler.eu-central-1.aws.neon.tech
19 POSTGRES_PASSWORD=npg_cr2Y2ypIT91f
20 POSTGRES_DATABASE=neondb
21 POSTGRES_URL_NO_SSL=postgres://neondb_owner:npg_cr2Y2ypIT91f@ep-orange-voice-a27hgqmk-pooler.eu-central-1.aws.neon.tech/neondb
22 POSTGRES_PRISMA_URL=postgres://neondb_owner:npg_cr2Y2ypIT91f@ep-orange-voice-a27hgqmk-pooler.eu-central-1.aws.neon.tech/neondb?connect_timeout=15&sslmode=require

```

What is 'seeding' in the context of databases?

A

Populating the database with an initial set of data

✓ Correct

That's right! Seeding is useful when you want to have some data to work with as you build your application.

< > ↺ 🌐 localhost:3000/seed

Pretty-print ☐

```
{ "message": "Database seeded successfully" }
```

Which customer does this invoice belong to?

B

Evil Rabbit

✓ Correct

That's right!

6

You've Completed Chapter 6

With your database now set up and integrated, you can continue building your application.

Ch7 - Fetching Data

In which of these scenarios should you not query your database directly?

A

When you're fetching data on the client

✓ Correct

That's right, you should not query your database directly when fetching data on the client as this would expose your database secrets.

What's one advantage of using React Server Components to fetch data?

B

They allow you to query the database directly from the server without an additional API layer.

✓ Correct

Server components allow you fetch data directly from your database.

What does SQL allow you to do in terms of fetching data?

B

Fetch and manipulate specific data

✓ Correct

SQL allows you to write targeted queries to fetch and manipulate specific data

localhost:3000/dashboard

Acme

Home

Invoices

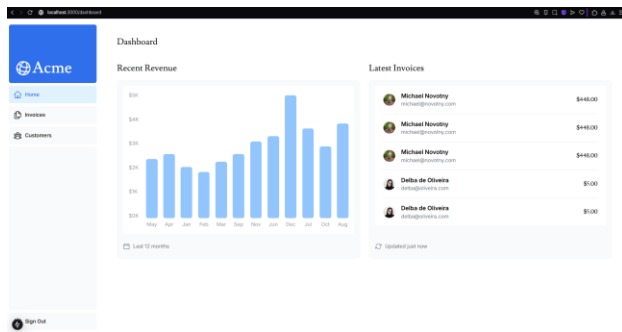
Customers

Dashboard

Recent Revenue



Last 12 months



```
app > dashboard > page.tsx > Page
1 import { Card } from '@app/ui/dashboard/cards';
2 import RevenueChart from '@app/ui/dashboard/revenue-chart';
3 import LatestInvoices from '@app/ui/dashboard/latest-invoices';
4 import { lusitana } from '@app/ui/fonts';
5 import {
6   fetchRevenue,
7   fetchLatestInvoices,
8   fetchCardData,
9 } from '@app/lib/data';
10
11 export default async function Page() {
12   const revenue = await fetchRevenue();
13   const latestInvoices = await fetchLatestInvoices();
14   const {
15     numberOfInvoices,
16     numberOfCustomers,
17     totalPaidInvoices,
18     totalPendingInvoices,
19   } = await fetchCardData();
20
21   return (
22     <main>
23       <h1 className={` ${lusitana.className} mb-4 text-xl md:text-2xl`}>
24         Dashboard
25       </h1>
26       <div className="grid gap-6 sm:grid-cols-2 lg:grid-cols-4">
27         <Card title="collected" value={totalPaidInvoices} type="collected" />
28         <Card title="Pending" value={totalPendingInvoices} type="pending" />
29         <Card title="Total Invoices" value={numberOfInvoices} type="invoices" />
30         <Card
31           title="Total Customers"
32           value={numberOfCustomers}
33           type="customers"
34         />
35       </div>
36       <div className="mt-6 grid grid-cols-1 gap-6 md:grid-cols-4 lg:grid-cols-8">
37         <RevenueChart revenue={revenue} />
38         <LatestInvoices latestInvoices={latestInvoices} />
39       </div>
40     </main>
41   );
42 }
```

When might you want to use a waterfall pattern?

A

To satisfy a condition before making the next request

✓ Correct

For example, you might want to fetch a user's ID and profile information first. Once you have the ID, you might then proceed to fetch their list of friends.

7

You've Completed Chapter 7

You've learned about some of the different ways to fetch data in Next.js.

Ch8 - Static and Dynamic Rendering

Why might static rendering not be a good fit for a dashboard app?

C

Because the application will not reflect the latest data changes

✓ Correct

When your data updates, you want to show the latest changes in your dashboard. Static Rendering is not a good fit for this use case.

What kind of information is typically only known at request time?

C

Cookies and URL search params

✓ Correct

Cookies and URL search params

app > lib > TS data.ts

```
export async function fetchRevenue() {
  try {
    // Artificially delay a response for demo purposes.
    // Don't do this in production :)

    console.log('Fetching revenue data...');
    await new Promise((resolve) => setTimeout(resolve, 3000));

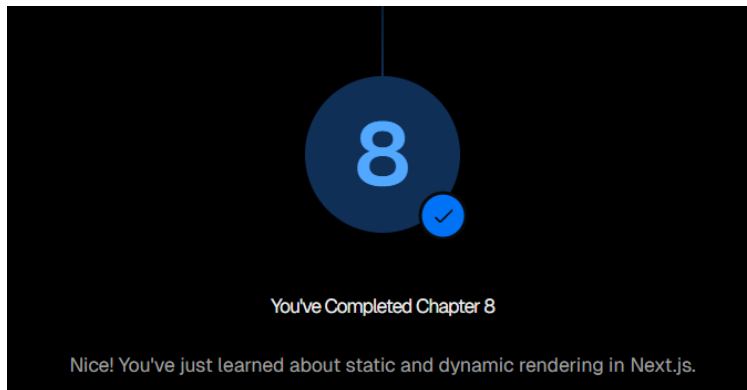
    const data = await sql<Revenue[]>`SELECT * FROM revenue`;

    console.log('Data fetch completed after 3 seconds.');
```

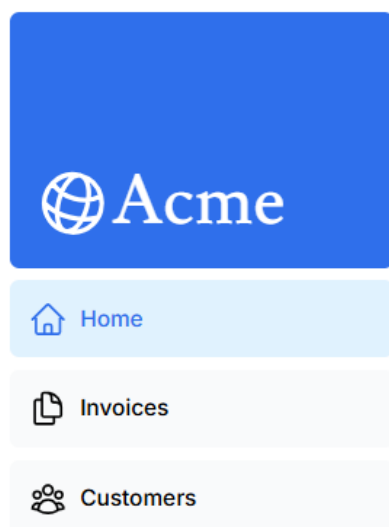
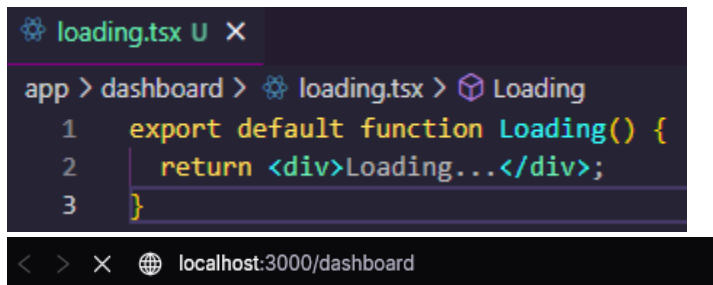
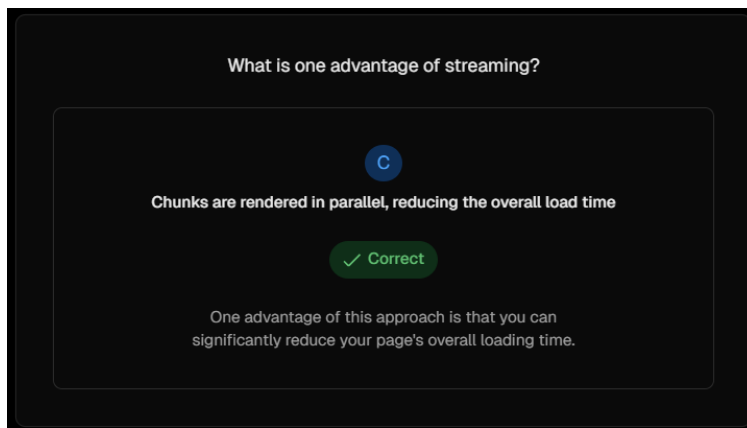
```
    return data;
  } catch (error) {
    console.error('Database Error:', error);
    throw new Error('Failed to fetch revenue data.');
```

```
  }
}
```

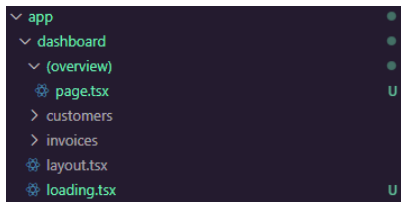
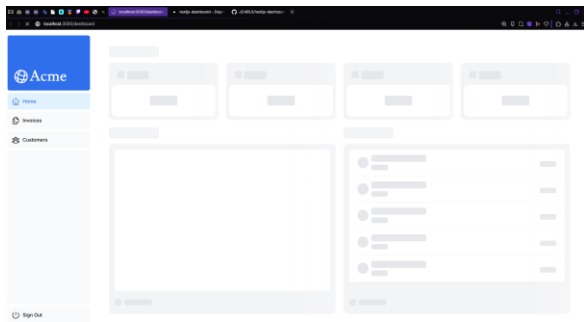
```
GET /dashboard 200 in 5605ms
Data fetch completed after 3 seconds.
Fetching revenue data...
```



Ch9 – Streaming

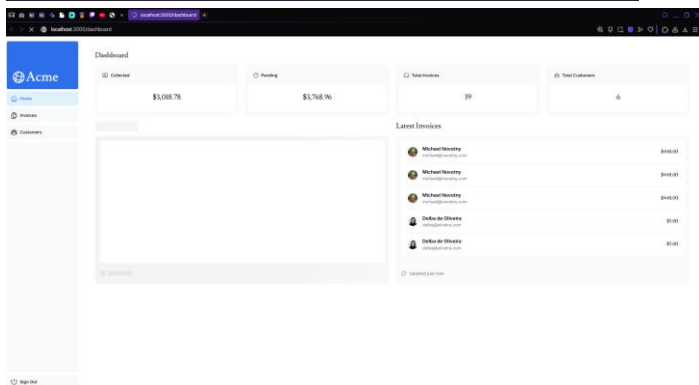


Loading...



```
page.tsx U X
app > dashboard > (overview) > page.tsx > Page

1  import { Card } from '@app/ui/dashboard/cards';
2  import RevenueChart from '@app/ui/dashboard/revenue-chart';
3  import LatestInvoices from '@app/ui/dashboard/latest-invoices';
4  import { lusitana } from '@app/ui/fonts';
5  import { fetchLatestInvoices, fetchCardData } from '@app/lib/data';
6  import { Suspense } from 'react';
7  import { RevenueChartSkeleton } from '@app/ui/skeletons';
8
9  export default async function Page() {
10   const latestInvoices = await fetchLatestInvoices();
11   const {
12     numberOfInvoices,
13     numberOfCustomers,
14     totalPaidInvoices,
15     totalPendingInvoices,
16   } = await fetchCardData();
17
18   return (
19     <main>
20       <h1 className={` ${lusitana.className} mb-4 text-xl md:text-2xl`} >
21         Dashboard
22       </h1>
23       <div className="grid gap-6 sm:grid-cols-2 lg:grid-cols-4">
24         <Card title="Collected" value={totalPaidInvoices} type="collected" />
25         <Card title="Pending" value={totalPendingInvoices} type="pending" />
26         <Card title="Total Invoices" value={numberOfInvoices} type="invoices" />
27         <Card
28           title="Total Customers"
29           value={numberOfCustomers}
30           type="customers"
31         />
32       </div>
33       <div className="mt-6 grid grid-cols-1 gap-6 md:grid-cols-4 lg:grid-cols-8">
34         <Suspense fallback=<RevenueChartSkeleton />>
35           <RevenueChart />
36         </Suspense>
37         <LatestInvoices latestInvoices={latestInvoices} />
38       </div>
39     </main>
40   );
41 }
```




```

latest-invoices.tsx M X
app > ui > dashboard > latest-invoices.tsx > LatestInvoices > latestInvoices.map() callback

1 import { ArrowPathIcon } from '@heroicons/react/24/outline';
2 import clsx from 'clsx';
3 import Image from 'next/image';
4 import { lusitana } from '@app/ui/fonts';
5 import { fetchLatestInvoices } from '@app/lib/data';
6
7
8 export default async function LatestInvoices() { // Remove props
9   const latestInvoices = await fetchLatestInvoices();

```

In general, what is considered good practice when working with Suspense and data fetching?

C

Move data fetches down to the components that need it

✓ Correct

By moving data fetching down to the components that need it, you can create more granular Suspense boundaries. This allows you to stream specific components and prevent the UI from blocking.

9



You've Completed Chapter 9

You've learned how to stream components with Suspense and loading skeletons.

Ch10 - Partial Prerendering

```
PS C:\Users\Jcarl\Documents\W06-UF4-PF\nextjs-dashboard> pnpm install next@canary
```

What are the holes in the context of Partial Prerendering?

B

Locations where dynamic content will load asynchronously

✓ Correct

That's right! Holes are locations where dynamic content will load asynchronously at request time.

TS next.config.ts

TS next.config.ts > ...

```
1 import type { NextConfig } from 'next';
2
3 const nextConfig: NextConfig = {
4   experimental: {
5     ppr: 'incremental'
6   }
7 };
8
9 export default nextConfig;
```

app > dashboard > layout.tsx

export const experimental_ppr = true;

Ch11 - Adding Search and Pagination

```
app > dashboard > invoices > @page > @Page
1 import Pagination from '@app/ui/invoices/pagination';
2 import Search from '@app/ui/search';
3 import Table from '@app/ui/invoices/table';
4 import { CreateInvoice } from '@app/ui/invoices/buttons';
5 import { lusitana } from '@app/ui/fonts';
6 import { Suspense } from 'react';
7 import { InvoiceTableSkeleton } from '@app/ui/skeletons';
8 import { fetchInvoicesPages } from '@app/lib/data';
9
10 export default async function Page(props: {
11   searchParams?: Promise<
12     query?: string;
13     page?: string;
14   >;
15 }) {
16   const searchParams = await props.searchParams;
17   const query = searchParams?.query || '';
18   const currentPage = Number(searchParams?.page) || 1;
19   const totalPages = await fetchInvoicesPages(query);
20
21   return (
22     <div className="w-full">
23       <div className="flex w-full items-center justify-between">
24         <h1 className={`${lusitana.className} text-2xl`}>Invoices</h1>
25       </div>
26       <div className="mt-4 flex items-center justify-between gap-2 md:mt-8">
27         <search placeholder="Search Invoices..." />
28         <CreateInvoice />
29       </div>
30       <Suspense key={query + currentPage} fallback={<InvoiceTableSkeleton />>
31         <Table query={query} currentPage={currentPage} />
32       </Suspense>
33       <div className="mt-5 flex w-full justify-center">
34         <Pagination totalPages={totalPages} />
35       </div>
36     </div>
37   );
38 }
```

app > ui > search.tsx

```
search.tsx
1 'use client';
2
3 import { MagnifyingGlassIcon } from '@heroicons/react/24/outline';
4 import { usePathname, useRouter, useSearchParams } from 'next/navigation';
5 import { useDebounceCallback } from 'use-debounce';
6
7 export default function Search({ placeholder }: { placeholder: string }) {
8   const searchParams = useSearchParams();
9   const pathname = usePathname();
10   const { replace } = useRouter();
11
12   const handleSearch = useDebounceCallback((term: string) => {
13     const params = new URLSearchParams(searchParams);
14     params.set('page', '1'); // Reset to page 1 on new search
15     if (term) {
16       params.set('query', term);
17     } else {
18       params.delete('query');
19     }
20
21     replace(`${pathname}?${params.toString()}`);
22   }, 300); // Debounce with 300ms delay
23
24   return (
25     <div className="relative flex flex-1 flex-shrink-0">
26       <label htmlFor="search" className="sr-only">
27         Search
28       </label>
29       <input
30         id="search"
31         className="peer block w-full rounded-md border border-gray-200 py-[9px] pl-[18px] text-sm outline-2 placeholder:text-gray-500"
32         placeholder={placeholder}
33         onChange={e => handleSearch(e.target.value)}
34         defaultValue={searchParams.get('query')?.toString() || ''}
35       />
36       <MagnifyingGlassIcon className="absolute left-3 top-1/2 h-[18px] w-[18px] -translate-y-1/2 text-gray-500 peer-focus:text-gray-900" />
37     </div>
38   );
39 }
```

```

1  'use client';
2
3  import { ArrowLeftIcon, ArrowRightIcon } from '@heroicons/react/24/outline';
4  import clsx from 'clsx';
5  import Link from 'next/link';
6  import { generatePagination } from '@app/lib/utils';
7  import { usePathname, useSearchParams } from 'next/navigation';
8
9  export default function Pagination({ totalPages }: { totalPages: number }) {
10   const pathname = usePathname();
11   const searchParams = useSearchParams();
12   const currentPage = Number(searchParams.get('page')) || 1;
13
14   const createPageURL = (pageNumber: number | string) => {
15     const params = new URLSearchParams(searchParams);
16     params.set('page', pageNumber.toString());
17     return `${pathname}?${params.toString()}`;
18   };
19
20   // ...
21 }

```

PS C:\Users\Jcarl\Documents\M06-UF4-PF\nextjs-dashboard> pnpm i use-debounce

GET /dashboard/invoices?page=1&query=delba 200 in 294ms

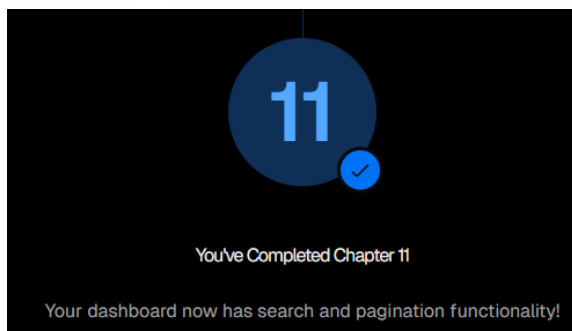
localhost:3000/dashboard/invoices

- Home
- Invoices
- Customers

Invoices

Q delba Create Invoice +

Customer	Email	Amount	Date	Status	
Delba de Oliveira	delba@oliveira.com	\$5.00	Aug 19, 2023	Paid	
Delba de Oliveira	delba@oliveira.com	\$5.00	Aug 19, 2023	Paid	
Delba de Oliveira	delba@oliveira.com	\$5.00	Aug 19, 2023	Paid	
Delba de Oliveira	delba@oliveira.com	\$203.48	Nov 14, 2022	Pending	
Delba de Oliveira	delba@oliveira.com	\$203.48	Nov 14, 2022	Pending	
Delba de Oliveira	delba@oliveira.com	\$203.48	Nov 14, 2022	Pending	



Ch12 - Mutating Data

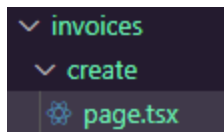
What's one benefit of using a Server Actions?

B

Progressive Enhancement.

✓ Correct

That's right! This allows users to interact with the form and submit data even if the JavaScript for the form hasn't been loaded yet or if it fails to load.



```
page.tsx U x
app > dashboard > invoices > create > page.tsx > Page
1  import Form from '@app/ui/invoices/create-form';
2  import Breadcrumbs from '@app/ui/invoices/breadcrumbs';
3  import { fetchCustomers } from '@app/lib/data';
4
5  export default async function Page() {
6    const customers = await fetchCustomers();
7
8    return (
9      <main>
10        <Breadcrumbs
11          breadcrumbs=[
12            { label: 'Invoices', href: '/dashboard/invoices' },
13            {
14              label: 'Create Invoice',
15              href: '/dashboard/invoices/create',
16              active: true,
17            },
18          ]
19        />
20        <Form customers={customers} />
21      </main>
22    );
23  }
```





Home

Invoices

Customers

Invoices / Create Invoice

Choose customer

Select a customer

Choose an amount

Enter USD amount

Set the invoice status

☐ Pending

☒ Paid

Cancel

Create Invoice

Evil Rabbit

evil@rabbit.com

\$11,111.00

May 17, 2025

Paid

```
TS actions U
app > lib > TS actions.ts > createInvoice
1  'use server';
2
3  import postgres from 'postgres';
4  import { z } from 'zod';
5  import { revalidatePath } from 'next/cache';
6  import { redirect } from 'next/navigation';
7
8  const sql = postgres(process.env.POSTGRES_URL!, { ssl: 'require' });
9
10 const FormSchema = z.object({
11   id: z.string(),
12   customerId: z.string(),
13   amount: z.coerce.number(),
14   status: z.enum(['pending', 'paid']),
15   date: z.string(),
16 });
17
18 const CreateInvoice = FormSchema.omit({ id: true, date: true });
19
20 export async function createInvoice(formData: FormData) {
21   const { customerId, amount, status } = CreateInvoice.parse({
22     customerId: formData.get('customerId'),
23     amount: formData.get('amount'),
24     status: formData.get('status'),
25   });
26   const amountInCents = amount * 100;
27   const date = new Date().toISOString().split('T')[0];
28
29   await sql`
30     INSERT INTO invoices (customer_id, amount, status, date)
31     VALUES (${customerId}, ${amountInCents}, ${status}, ${date})
32   `;
33
34   revalidatePath('/dashboard/invoices');
35   redirect('/dashboard/invoices');
36 }
```

app > ui > invoices > buttons.tsx

```
export function UpdateInvoice({ id }: { id: string }) {
  return (
    <Link
      href={`/${dashboard}/invoices/${id}/edit`}
      className="rounded-md border p-2 hover:bg-gray-100"
    >
      <PencilIcon className="w-5" />
    </Link>
  );
}
```

```
pagetx U X
app > dashboard > invoices > [id] > edit > page.tsx > ...

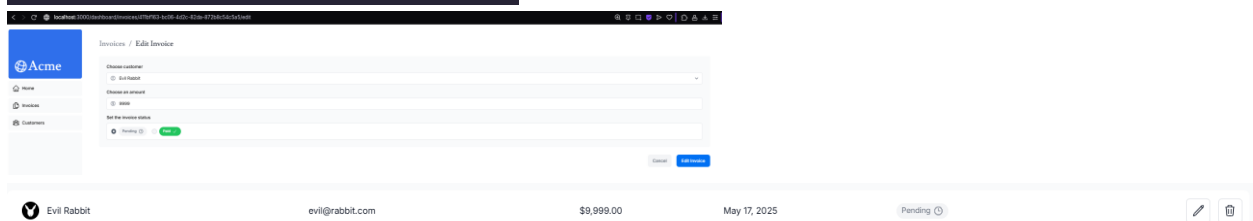
1   export const dynamic = "force-dynamic";
2
3   import Form from '@app/ui/invoices/edit-form';
4   import Breadcrumbs from '@app/ui/invoices/breadcrumbs';
5   import { fetchInvoiceById, fetchCustomers } from '@app/lib/data';
6   import { notFound } from 'next/navigation';
7
8   export default async function Page(props: { params: Promise<{ id: string }> }) {
9     const params = await props.params;
10    const id = params.id;
11    const [invoice, customers] = await Promise.all([
12      fetchInvoiceById(id),
13      fetchCustomers(),
14    ]);
15
16    if (!invoice) {
17      notFound();
18    }
19
20    return (
21      <main>
22        <Breadcrumbs
23          breadcrumbs={[
24            { label: 'Invoices', href: '/dashboard/invoices' },
25            {
26              label: 'Edit Invoice',
27              href: `/${dashboard}/invoices/${id}/edit`,
28              active: true,
29            },
30          ]}
31        />
32        <Form invoice={invoice} customers={customers} />
33      </main>
34    );
35  }
```

```
export async function updateInvoice(id: string, formData: FormData) {
  const { customerId, amount, status } = UpdateInvoice.parse({
    customerId: formData.get('customerId'),
    amount: formData.get('amount'),
    status: formData.get('status'),
  });

  const amountInCents = amount * 100;

  await sql`
    UPDATE invoices
    SET customer_id = ${customerId}, amount = ${amountInCents}, status = ${status}
    WHERE id = ${id}
  `;

  revalidatePath(`/${dashboard}/invoices`);
  redirect(`/${dashboard}/invoices`);
}
```



```
app > ui > invoices > buttons.tsx
```


```
import { deleteInvoice } from '@app/lib/actions';

export function DeleteInvoice({ id }: { id: string }) {
  const deleteInvoiceWithId = deleteInvoice.bind(null, id);

  return (
    <form action={deleteInvoiceWithId}>
      <button type="submit" className="rounded-md border p-2 hover:bg-gray-100">
        <span className="sr-only">Delete</span>
        <TrashIcon className="w-4" />
      </button>
    </form>
  );
}
```

```
app > lib > TS actions.ts
```

```
export async function deleteInvoice(id: string) {  
  await sql`DELETE FROM invoices WHERE id = ${id}`;  
  revalidatePath('/dashboard/invoices');  
}
```



Home

















Invoices

Customers

Invoices

Search Invoices...

Create Invoice +

Customer	Email	Amount	Date	Status		
 Michael Novotny	michael@novotny.com	\$448.00	Sep 10, 2023	Paid ✓		
 Michael Novotny	michael@novotny.com	\$448.00	Sep 10, 2023	Paid ✓		
 Michael Novotny	michael@novotny.com	\$448.00	Sep 10, 2023	Paid ✓		
 Delba de Oliveira	delba@oliveira.com	\$5.00	Aug 19, 2023	Paid ✓		
 Delba de Oliveira	delba@oliveira.com	\$5.00	Aug 19, 2023	Paid ✓		
 Delba de Oliveira	delba@oliveira.com	\$5.00	Aug 19, 2023	Paid ✓	