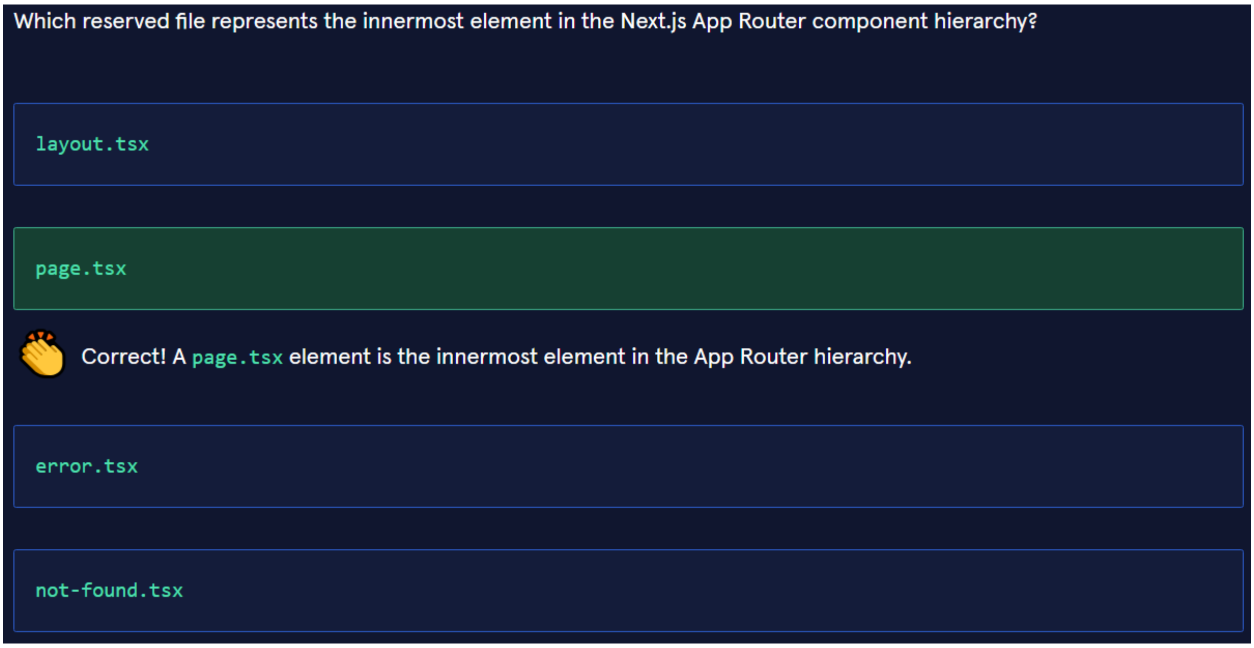
QUIZ



Which reserved file represents the innermost element in the Next.js App Router component hierarchy?

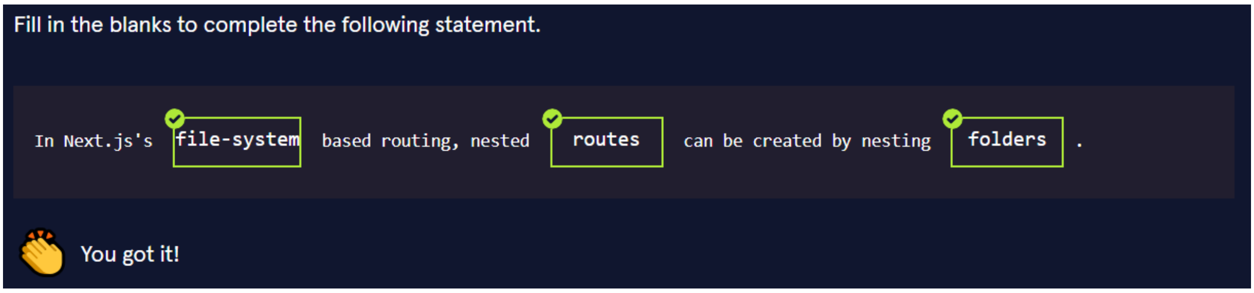
`layout.tsx`

`page.tsx`

Correct! A `page.tsx` element is the innermost element in the App Router hierarchy.

`error.tsx`

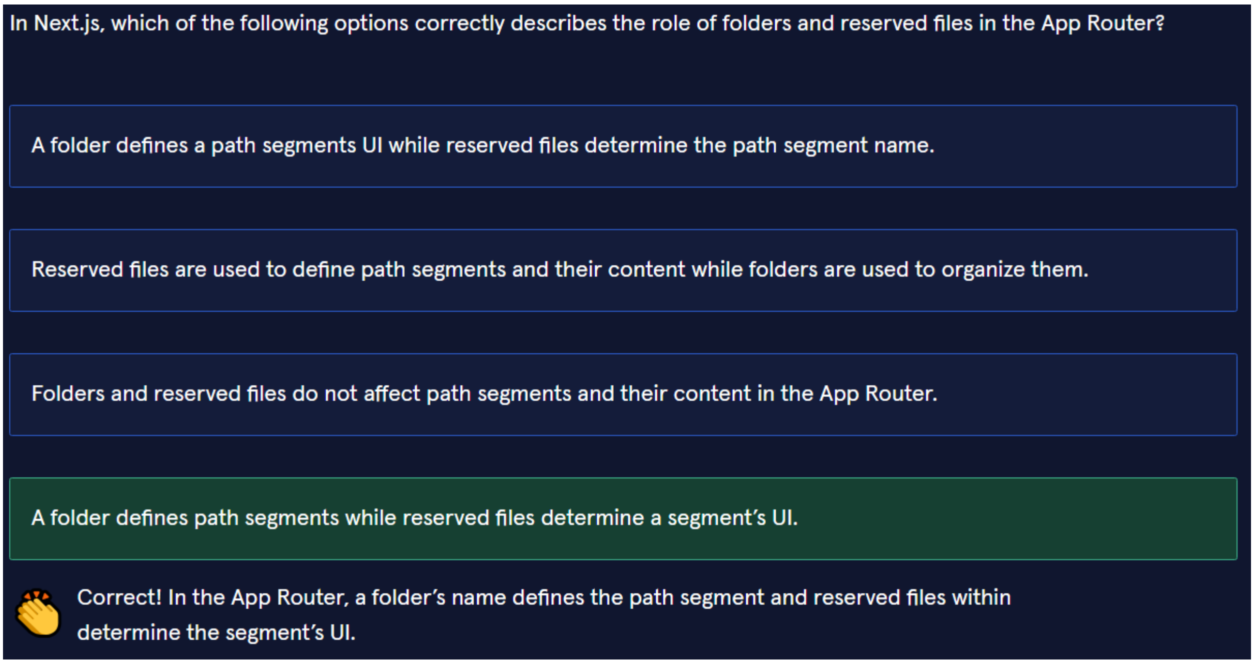
`not-found.tsx`



Fill in the blanks to complete the following statement.

```
In Next.js's file-system based routing, nested routes can be created by nesting folders .
```

You got it!



In Next.js, which of the following options correctly describes the role of folders and reserved files in the App Router?

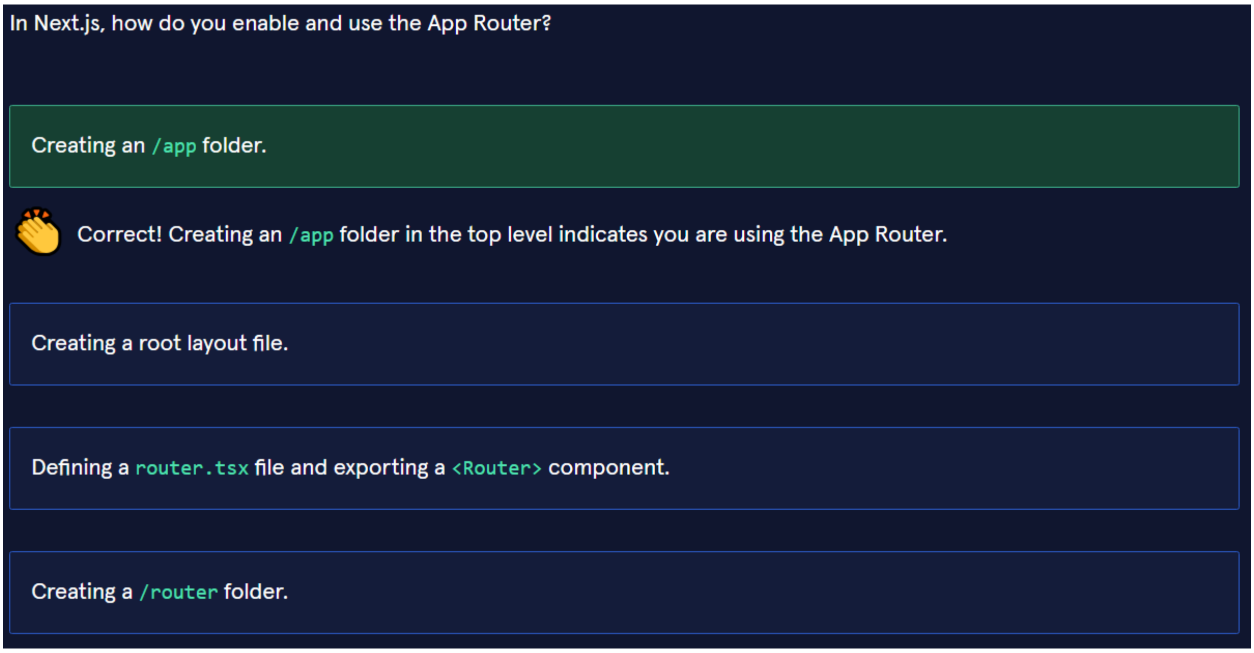
A folder defines a path segments UI while reserved files determine the path segment name.

Reserved files are used to define path segments and their content while folders are used to organize them.

Folders and reserved files do not affect path segments and their content in the App Router.

A folder defines path segments while reserved files determine a segment's UI.

Correct! In the App Router, a folder's name defines the path segment and reserved files within determine the segment's UI.



In Next.js, how do you enable and use the App Router?

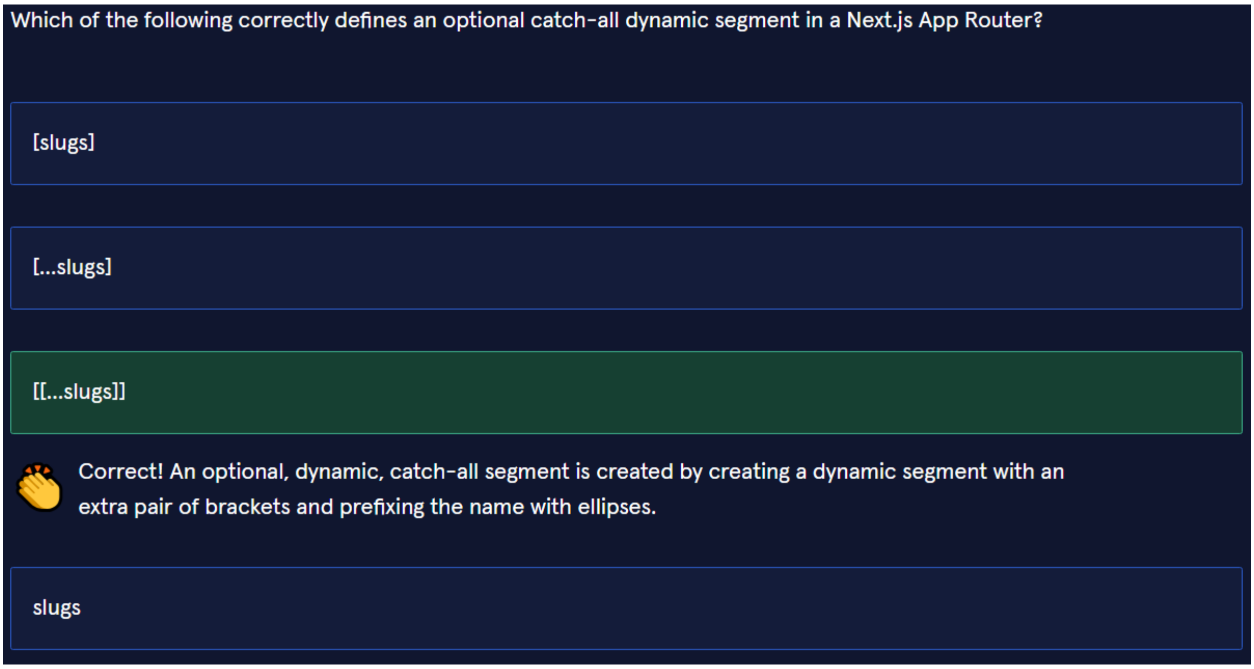
Creating an `/app` folder.

👏 Correct! Creating an `/app` folder in the top level indicates you are using the App Router.

Creating a root layout file.

Defining a `router.tsx` file and exporting a `<Router>` component.

Creating a `/router` folder.



Which of the following correctly defines an optional catch-all dynamic segment in a Next.js App Router?

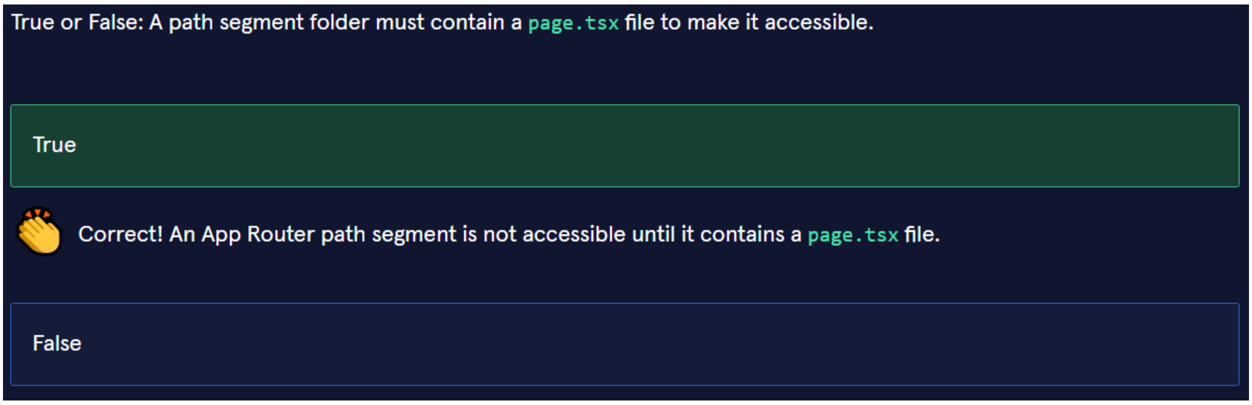
[slugs]

[...slugs]

[[...slugs]]

👏 Correct! An optional, dynamic, catch-all segment is created by creating a dynamic segment with an extra pair of brackets and prefixing the name with ellipses.

slugs

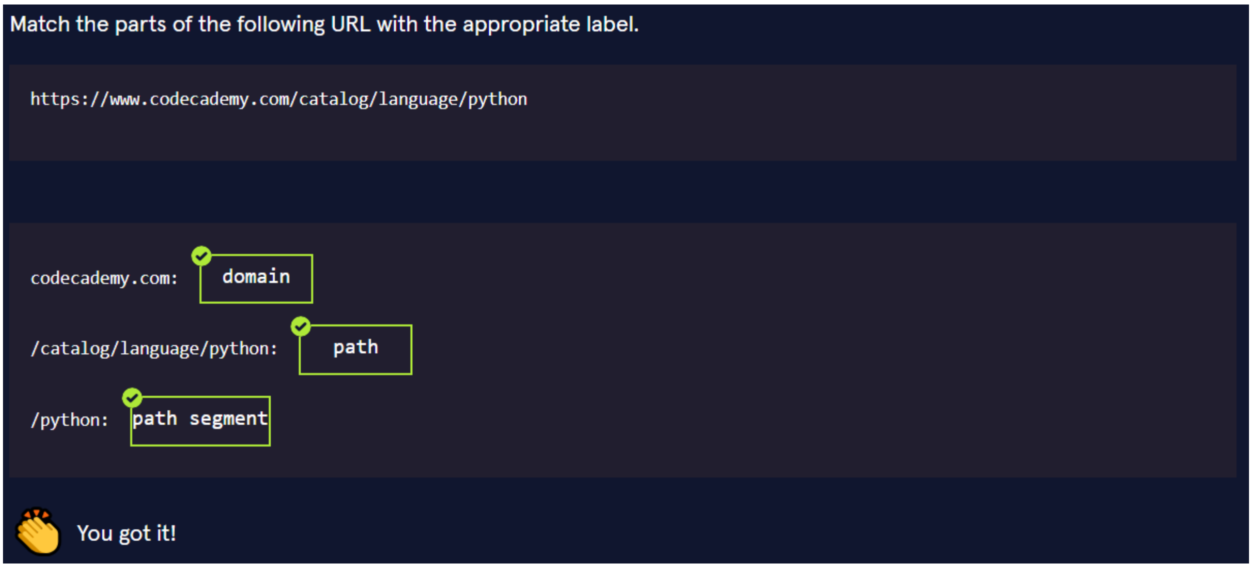


True or False: A path segment folder must contain a `page.tsx` file to make it accessible.

True

👏 Correct! An App Router path segment is not accessible until it contains a `page.tsx` file.

False

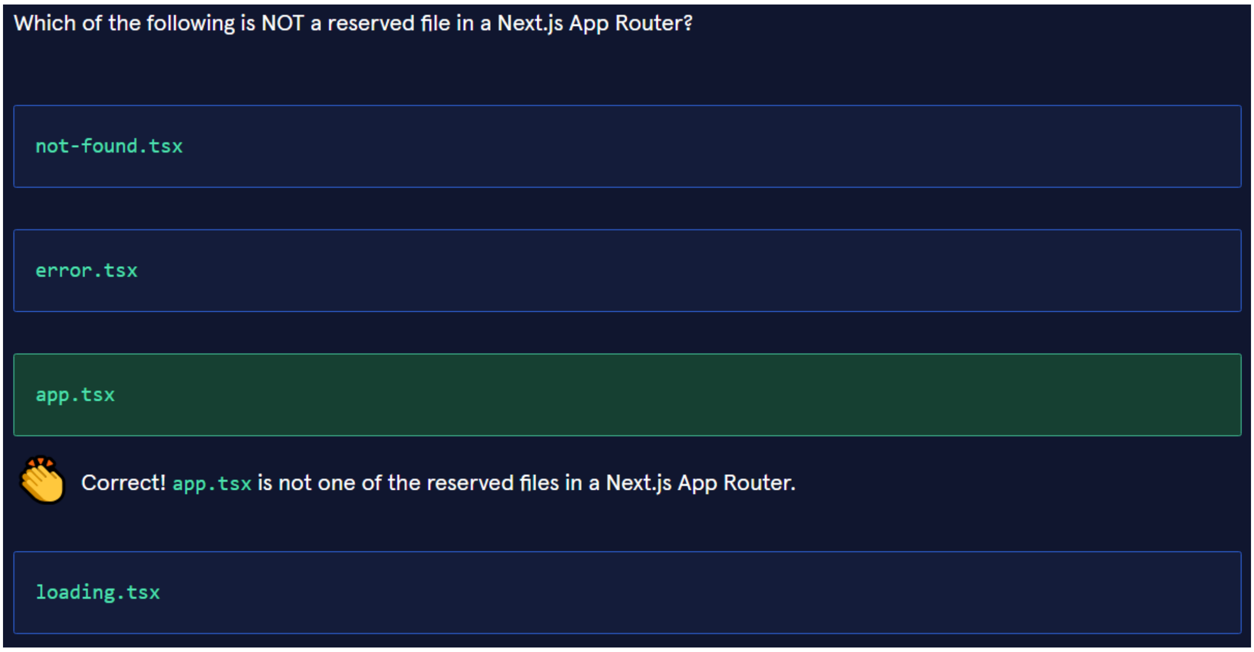


Match the parts of the following URL with the appropriate label.

```
https://www.codecademy.com/catalog/language/python
```

```
codecademy.com: domain
/catalog/language/python: path
/python: path segment
```

👏 You got it!

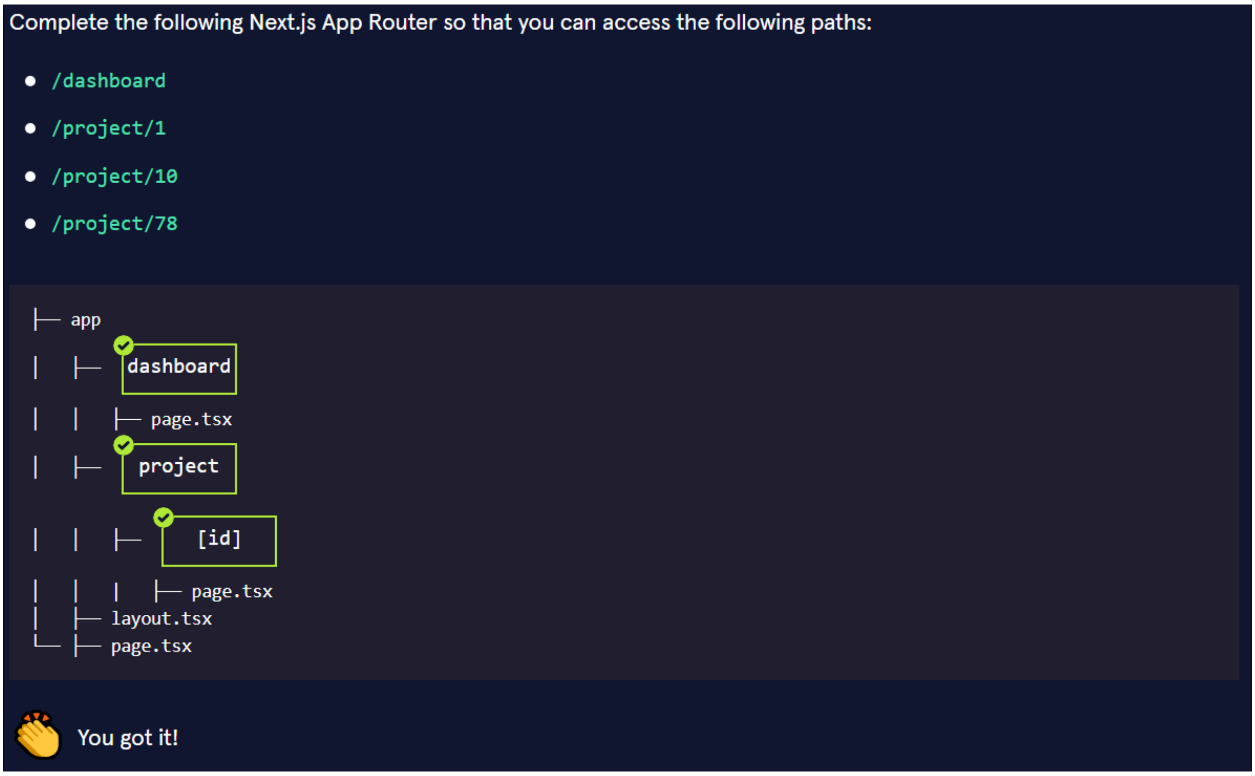


Which of the following is NOT a reserved file in a Next.js App Router?

- `not-found.tsx`
- `error.tsx`
- `app.tsx`

👏 Correct! `app.tsx` is not one of the reserved files in a Next.js App Router.

- `loading.tsx`

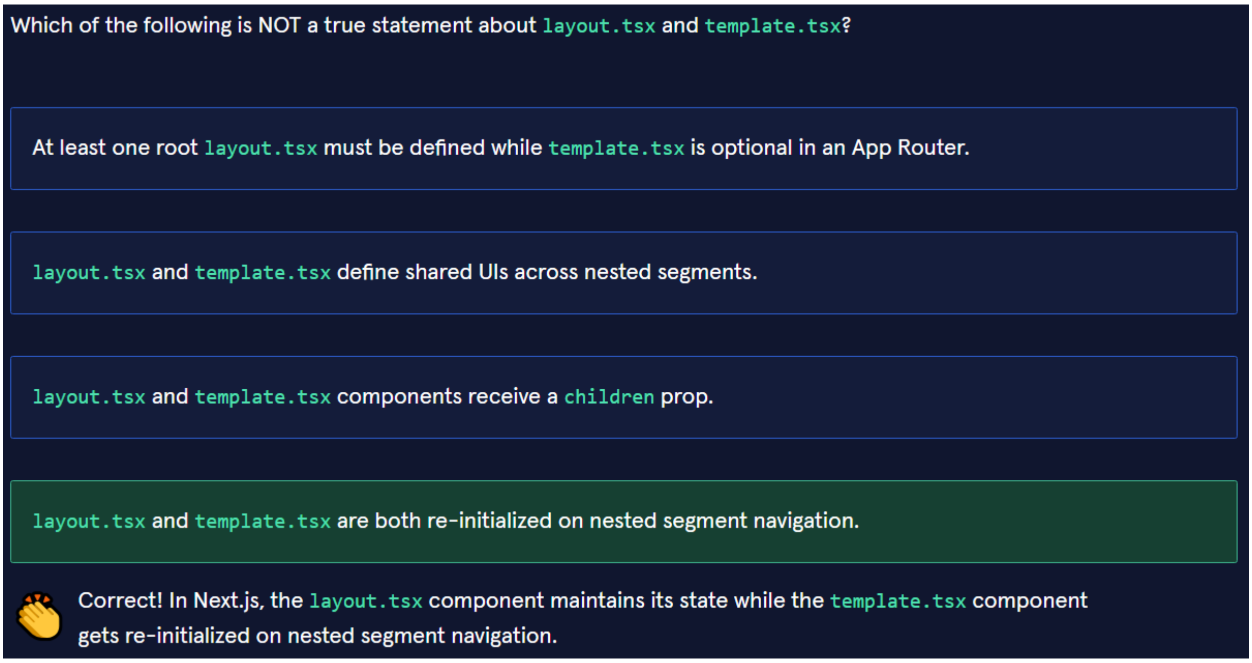


Complete the following Next.js App Router so that you can access the following paths:

- `/dashboard`
- `/project/1`
- `/project/10`
- `/project/78`

```
├─ app
|   ├─ ✓dashboard
|   |   ├─ page.tsx
|   ├─ ✓project
|   |   ├─ ✓[id]
|   |   |   ├─ page.tsx
|   ├─ layout.tsx
└─  ├─ page.tsx
```

👏 You got it!



Which of the following is NOT a true statement about `layout.tsx` and `template.tsx`?

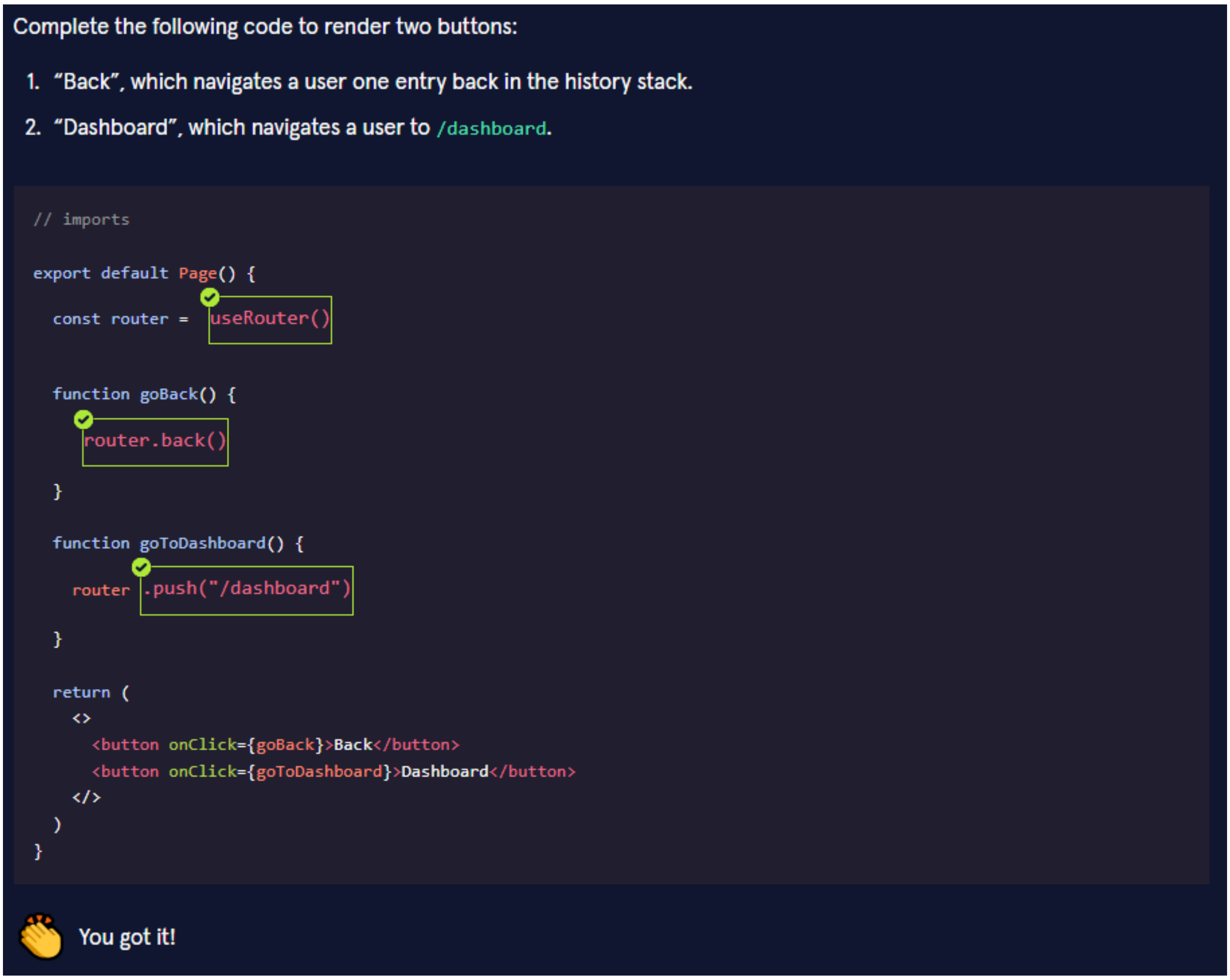
At least one root `layout.tsx` must be defined while `template.tsx` is optional in an App Router.

`layout.tsx` and `template.tsx` define shared UIs across nested segments.

`layout.tsx` and `template.tsx` components receive a `children` prop.

`layout.tsx` and `template.tsx` are both re-initialized on nested segment navigation.

👏 Correct! In Next.js, the `layout.tsx` component maintains its state while the `template.tsx` component gets re-initialized on nested segment navigation.



Complete the following code to render two buttons:

1. “Back”, which navigates a user one entry back in the history stack.
2. “Dashboard”, which navigates a user to `/dashboard`.

```
// imports

export default Page() {
  const router = ✓ useRouter()

  function goBack() {
    ✓ router.back()
  }

  function goToDashboard() {
    router ✓ .push("/dashboard")
  }

  return (
    <>
      <button onClick={goBack}>Back</button>
      <button onClick={goToDashboard}>Dashboard</button>
    </>
  )
}
```

👏 You got it!