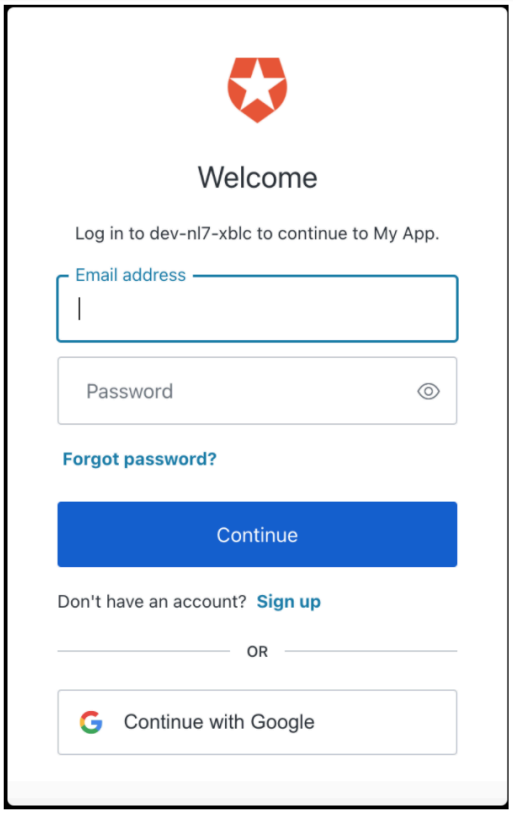
How to write a React + Auth0 App

井民全, Jing, mqjing@gmail.com

In the document, I'll show you how to write a React Application with Auth0 to enable the user authentication function. When user clicked the login button, the app will show an Auth0 Universal Login page for users to login or signup using a username and password or a social provider.

Here is the result.



Enjoy!

By Jing.

Table of contents

1. Configure

1.1. Get the App key

Page: https://manage.auth0.com/dashboard

[Applications] -> [Applications] -> Your app

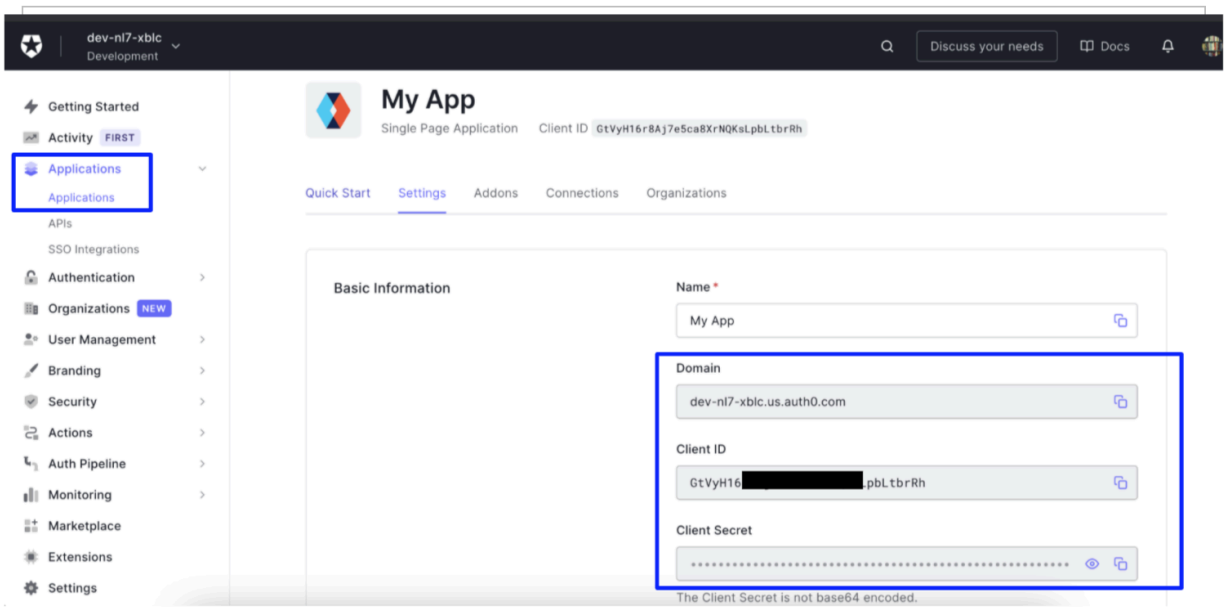


Fig. The App dashboard. (Edit)

Auth0 Dashboard: App Setup

Domain: Your-Domain Client ID: Your-Client-ID

1.2. Setup URL

1.2.1. Allowed Callback URLs

The URLs is the your app URL after user has authenticated.

Allowed Callback URLs

http://localhost:3000

After the user authenticates we will only call back to any of these URLs. You can specify multiple valid URLs by comma-separating them (typically to handle different environments like QA or testing). Make sure to specify the protocol (`https://`) otherwise the callback may fail in some cases. With the exception of custom URI schemes for native clients, all callbacks should use protocol `https://` . You can use Organization URL parameters in these URLs.

1.2.2. Allowed Web Origins

The URLs listed should be allowed for use with Cross-Origin.

Allowed Web Origins

http://localhost:3000

Comma-separated list of allowed origins for use with Cross-Origin Authentication, Device Flow, and web message response mode, in the form of `<scheme> "://" <host> [ ":" <port> ]` , such as `https://login.mydomain.com` or `http://localhost:3000` . You can use wildcards at the subdomain level (e.g.: `https://*.contoso.com`). Query strings and hash information are not taken into account when validating these URLs.

1.2.3. Allowed Logout URL

Allowed Logout URLs

http://localhost:3000

A set of URLs that are valid to redirect to after logout from Auth0. After a user logs out from Auth0 you can redirect them with the `returnTo` query parameter. The URL that you use in `returnTo` must be listed here. You can specify multiple valid URLs by comma-separating them. You can use the star symbol as a wildcard for subdomains (`*.google.com`). Query strings and hash information are not taken into account when validating these URLs. Read more about this at https://auth0.com/docs/logout

2. Coding

2.1. Install dependency

bash

```
# react template
npx create-react-app my-app --template typescript
cd my-app

# auth0
yarn add @auth0/auth0-react
```

2.2. Step 1: Configure the Auth0Provider component

Integrate the Auth0 React SDK by wrapping your root component with the Auth0Provider from @auth/auth0-react. For further information about how Auth0 manage the user authentication state, check React Context .

Fill the "**YOUR-DOMAIN**" and "**YOUR-CLIENT-ID**" from the Auth0 dashboard.

File: src/index.ts

```
import React from 'react';
```

```
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import * as serviceWorker from './serviceWorker';
import { Auth0Provider } from "@auth0/auth0-react";

ReactDOM.render(
  <Auth0Provider
    domain="YOUR-DOMAIN"
    clientId="YOUR-CLIENT-ID"
    redirectUri={window.location.origin}
  >
    <React.StrictMode>
    <App />
    </React.StrictMode>

  </Auth0Provider>,
  document.getElementById('root')
);

// If you want your app to work offline and load faster, you can change
// unregister() to register() below. Note this comes with some pitfalls.
// Learn more about service workers: https://bit.ly/CRA-PWA
serviceWorker.unregister();
```

2.3. Step 2: Create Login/Logout buttons

2.3.1. Login Button

Create a login button using the **loginWithRedirect**() from the **useAuth0**() hook. When user clicked, it will redirect user to the Auth0 Universal Login page for login. Check the detail at here.

File: src/features/LoginButton/LoginButton.tsx

```
import React from "react";
import { useAuth0 } from "@auth0/auth0-react";

const LoginButton = () => {
  const { loginWithRedirect } = useAuth0();

  return <button onClick={() => loginWithRedirect()}>Log In</button>;
```

```
};

export default LoginButton;
```

2.3.2. Logout Button

```
import React from "react";
import { useAuth0 } from "@auth0/auth0-react";

const LogoutButton = () => {
  const { logout } = useAuth0();

  return (
    <button onClick={() => logout({ returnTo: window.location.origin })}>
      Log Out
    </button>
  );
};

export default LogoutButton;
```

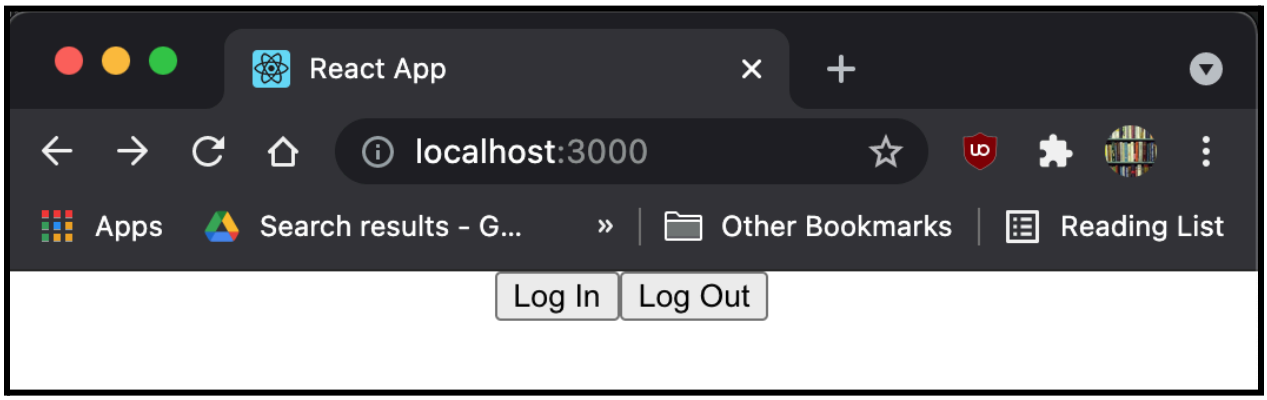
2.4. Step 3: Add the Login/Logout button

```
import React from 'react';
import './App.css';

import LoginButton from './features/LoginButton/LoginButton';
import LogoutButton from './features/LogoutButton/LogoutButton';

function App() {
  return (
    <div className="App">
      <LoginButton/>
      <LogoutButton/>
    </div>
  );
```

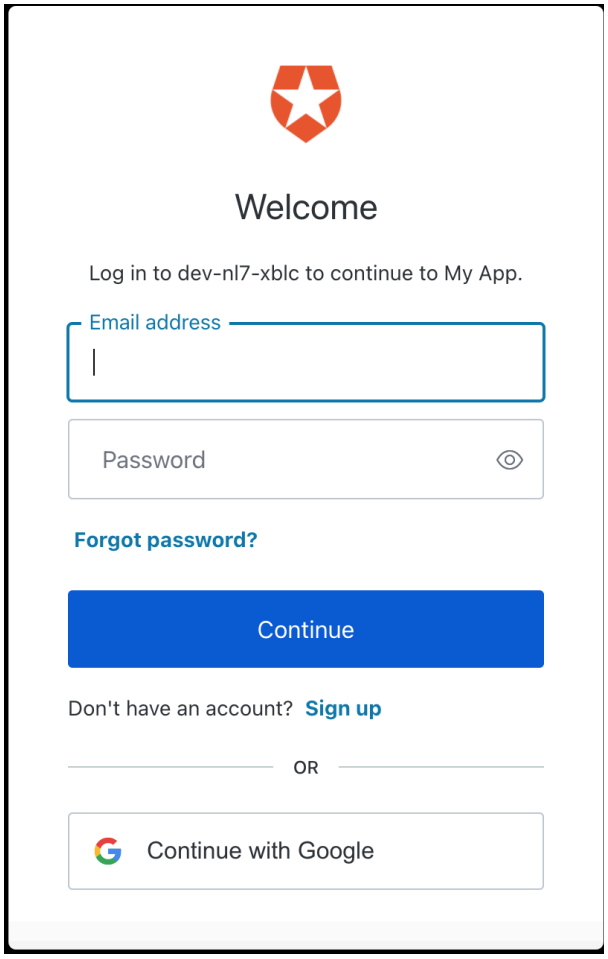
```
}

export default App;
```

3. Build && Run

yarn start



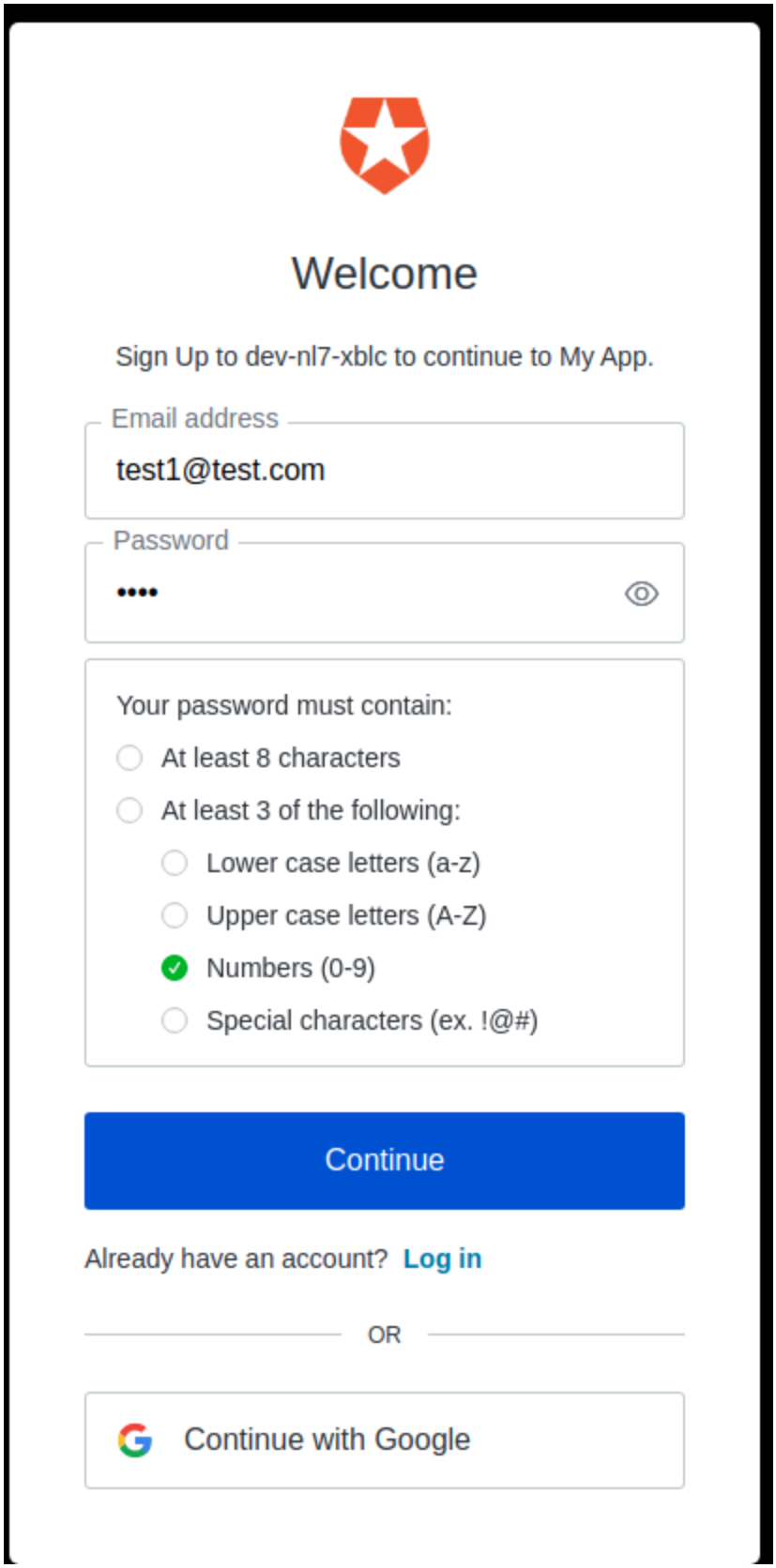
Click [Log in]

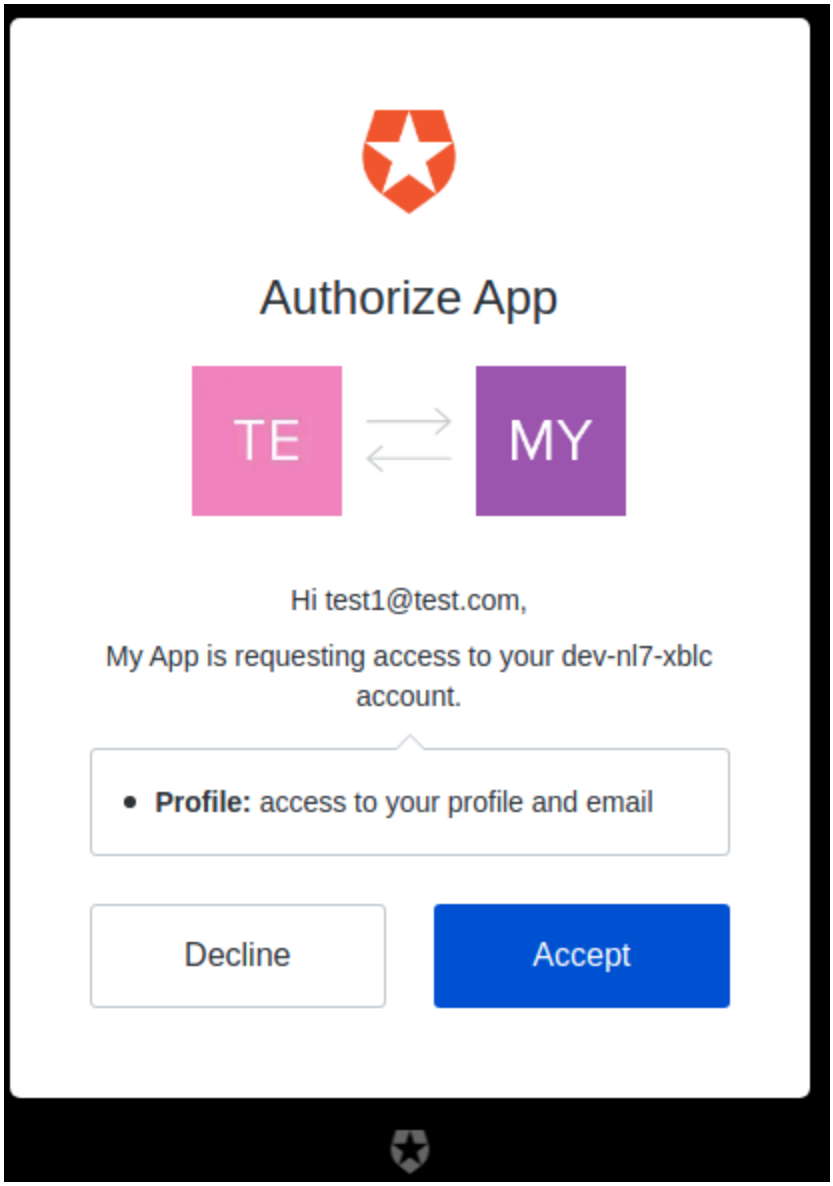


4. Misc

4.1. the Sign up flow

Step 1: Input [Email address] & [Password]


Welcome
Sign Up to dev-nl7-xblc to continue to My App.
Email address
test1@test.com
Password
••••
Your password must contain:
At least 8 characters
At least 3 of the following:
Lower case letters (a-z)
Upper case letters (A-Z)
Numbers (0-9)
Special characters (ex. !@#)
Continue
Already have an account? Log in
OR
Continue with Google



4.1.1. Check

Auth0 dashboard

[User management] -> [Users]

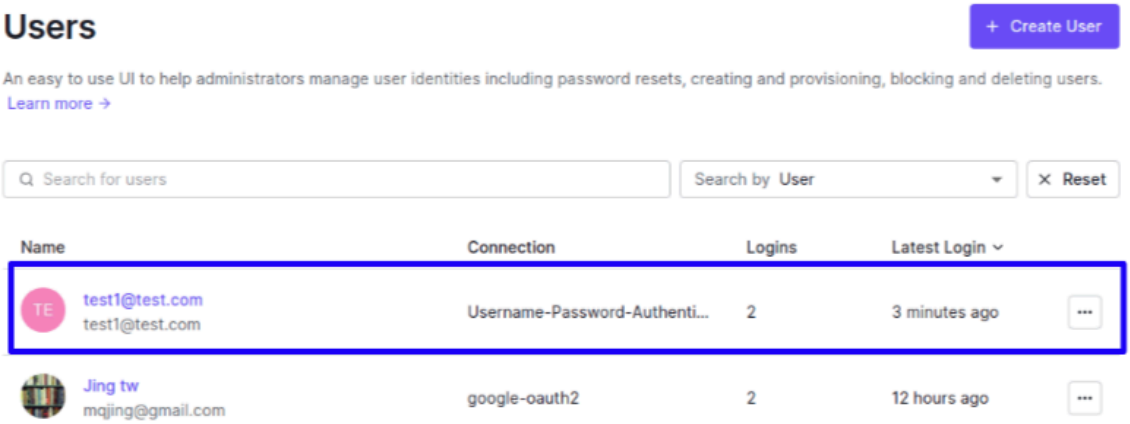


Fig. New user test1@test.com on the dashboard. (Edit)

:

5. References

1. https://auth0.com/docs/quickstart/spa/react/01-login
2. https://developer.mozilla.org/en-US/docs/Web/HTTP/CORS
3. https://auth0.com/blog/complete-guide-to-react-user-authentication/