# Exercise 10: Protected Routes

We will run and verify our app after the implementation of Protected Routes.

WE'LL COVER THE FOLLOWING

- Project
- Exercise

## Project #

Environment Variables

| Key: | Value: |
| --- | --- |
| REACT_APP_API_KEY | Not Specified... |
| REACT_APP_AUTH_... | Not Specified... |
| REACT_APP_DATAB... | Not Specified... |
| REACT_APP_PROJEC... | Not Specified... |
| REACT_APP_STORA... | Not Specified... |
| REACT_APP_MESSA... | Not Specified... |

```
import React from 'react';

import { AuthUserContext } from '../Session';
import { PasswordForgetForm } from '../PasswordForget';
import PasswordChangeForm from '../PasswordChange';
import { withAuthorization } from '../Session';

const AccountPage = () => (
  <AuthUserContext.Consumer>
    {authUser => (
      <div>
        <h1>Account: {authUser.email}</h1>
        <PasswordForgetForm />
        <PasswordChangeForm />
      </div>
    )}
  </AuthUserContext.Consumer>
);

const authCondition = authUser => !!authUser;
```

```
export default withAuthorization(authCondition)(AccountPage);
```

Now, all the user-specific routes are protected. Without signing in, we cannot access a protected page such as `Home` or `Landing`.

If we try replacing the `/signup` keyword with `/home` in the app's URL, it redirects us to the sign-up page.

## Exercise #

1. Confirm your source code for this section.
2. Research how a *role-based* or *permission-based* authorization can be implemented.

---

The next section will deal with how our application interacts with the Firebase database in order to perform operations like retrievals.