AWS Mobile Blog

Amplify Framework simplifies configuration for OAuth flows, the hosted UI, and AR/VR scenes for mobile and web developers

by Nader Dabit | on 02 APR 2019 | Permalink | Comments | Share

Written by Kurt Kemple, Sr. Developer Advocate & Gerard Sans, Sr. Developer Advocate

The Amplify Framework is an open-source project for building cloud-enabled applications. Today, we're happy to announce new features in the authentication , XR, storage, and API categories. It's now possible to configure OAuth 2.0 authorization flows and enable the Amazon Cognito hosted UI from the Amplify command line interface (CLI) (part of the Amplify Framework). Previously, you had to go to the Amazon Cognito console to set this up and construct the proper application configurations manually in the web or mobile application.

For the Amplify XR category, the Amplify Framework provides a simplified configuration setup for augmented reality (AR) or virtual reality (VR) scenes that are powered by Amazon Sumerian in your mobile and web applications. The Amplify Framework enables you to automatically add authorization to your scenes from the Amplify CLI by using Amazon Cognito and IAM.

In addition, the Amplify Framework now supports setting up fine-grained CRUD-style permissions for the storage and API (REST) categories. This works for both authenticated and unauthenticated workflows.

Setting up the hosted UI and OAuth from the Amplify CLI

In this release of Amplify, you can now set up the Amazon Cognito hosted UI and OAuth from the Amplify CLI. This enables you to use Amazon Cognito user pools to set up federation between various identity providers (such as Amazon, Google, and Facebook) from the Amplify CLI. You use OAuth flows for the federation—either through the hosted UI or your application code, by using the endpoints directly.

Let’s look at setting up OAuth for Facebook in a React app. The first step is to create a new project.

Setting up the project

We will be using npx to create the React application. To learn more about npx, click here.

Use the following command to create a new React app project:

Bash

```
npx create-react-app my-project
```

Next you need to set up Amplify:

Bash

```
cd my-project
amplify init
npm i aws-amplify aws-amplify-react
```

Setting up OAuth with Facebook

After Amplify is set up, the next step is to create a Facebook app to enable authentication. For instructions on how to set up a Facebook app, see Adding Social Identity Providers to a User Pool.

Adding the authentication category with the Amplify CLI

Now that you have a Facebook app, it's time to add authentication to your app with the Amplify CLI. Make sure to enter the values in the snippet below.

Bash

```
amplify add auth

// Do you want to use the default authentication and security c
// How do you want end users to be able to sign up? Username
// What attributes are required for signing up? Email
// What domain name prefix you want us to create for you? <ente
// Enter your redirect signin URI: http://localhost:3000/
// Do you want to add another redirect signin URI? No
// Enter your redirect signout URI: http://localhost:3000/
// Do you want to add another redirect signout URI? No
// Select the identity providers you want to configure for your
// Enter your Facebook App ID: <facebook_app_id>
// Enter your Facebook App Secret: <facebook_app_secret>
```

After the command completes, you need to run `amplify push` to create your Amazon Cognito user pool and get your endpoints for your hosted UI. You get the domain URL from the output of the “amplify push” command.

You need the endpoints to finish configuring your Facebook app.

To finish setting up your Facebook app go back to the documentation linked in the “Setting Up OAuth With Facebook” section and follow the steps for adding your app domain. You get the domain URL from the output of the amplify push command.

Add OAuth to React

The next step is to configure Amplify and the authentication category in your app. Update src/App.js with the following:

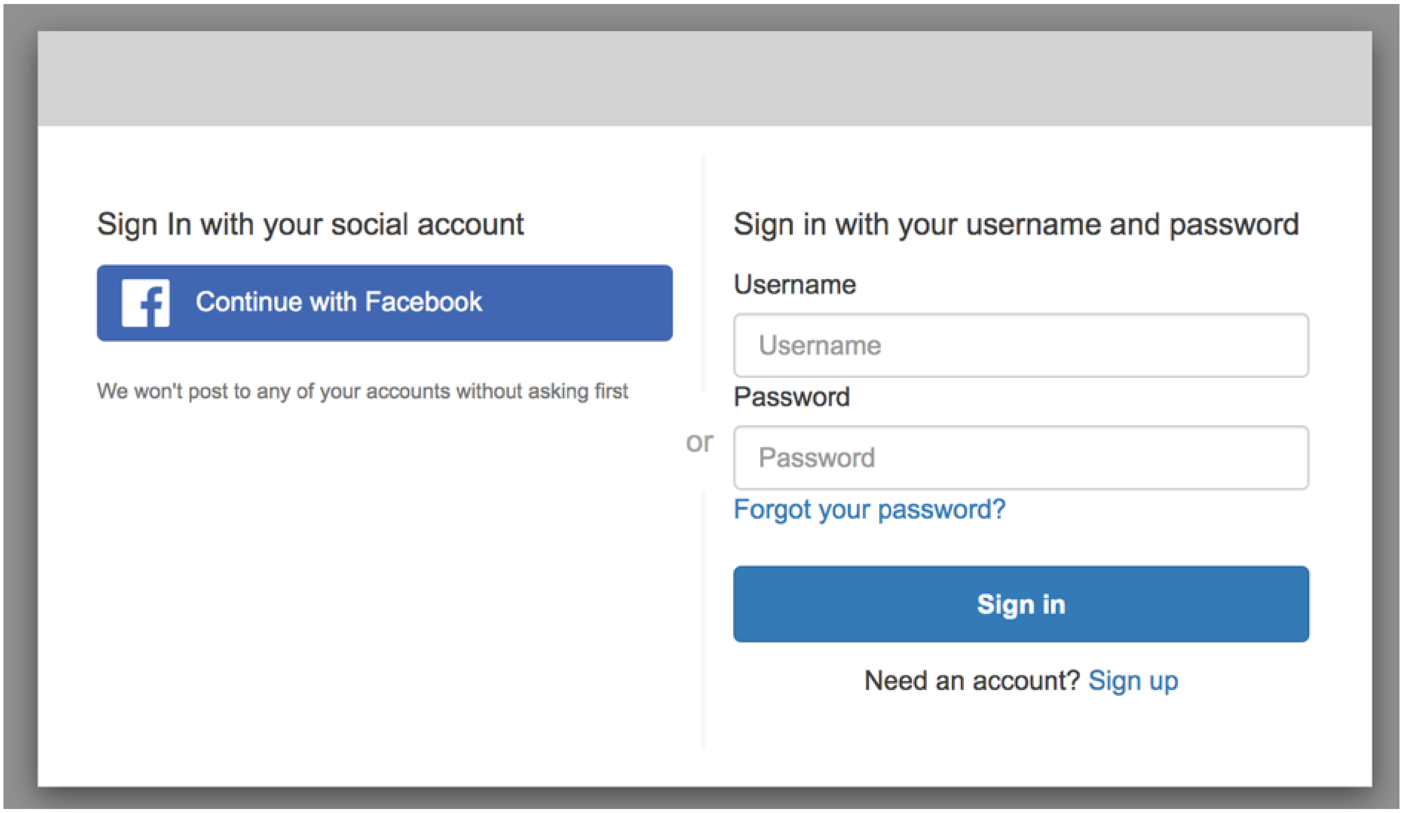
```js
import React, { Component } from "react";
import logo from "./logo.svg";
import "./App.css";
import Amplify from "aws-amplify";
import { withOAuth } from "aws-amplify-react";
import aws_exports from "./aws-exports";

Amplify.configure(aws_exports);

class App extends Component {
  render() {
    return (
      <div className="App">
        <header className="App-header">
```

```
            <img src={logo} className="App-logo" alt="logo" />
            <p>
              Edit <code>src/App.js</code> and save to reload.
            </p>
```

The withOAuth higher-order component (HOC) provides the property OAuthSignIn that takes the user to the hosted UI for authenticating. If you don't want to use the HOC and instead want to handle sending users to the hosted UI, see the instructions in Launching the Hosted UI.

Try it out!

After you've updated App.js, it's time to run the app and test the login. You should see a screen like the following when you choose the **Sign in with Facebook** button.



Sign In with your social account

We won't post to any of your accounts without asking first

Sign in with your username and password

Username: Username

Password: Password

Need an account? Sign up

Improving the Amazon Sumerian scene setup

The XR category within Amplify Framework enables applications to use AR and VR content, and provides built-in Amazon Sumerian support.

With this release, we've simplified adding scenes from Amazon Sumerian. The Amplify CLI now prompts a few more questions so it can set up authorization for a scene, and add all the necessary roles and policies for you.

Adding a new scene workflow

Let's look at the new workflow. We're using a React app to demonstrate it.

Bash

```
npx create-react-app xr-app
cd xr-app
npm i aws-amplify aws-amplify-react
```

To initialize the new project, run the following:

Bash

```
amplify init
```

Before proceeding, we need to publish the scene and download its configuration as before by using the Amazon Sumerian console.

1. In the Sumerian console, create or open the scene that you want to use, choose ***Publish*** dropdown from the top-right corner, and then choose ***Host privately***.
2. You're then prompted with a dialog box. Choose the **Publish** button.
3. Choose the **Download JSON configuration** button, and place the file into the ***src*** directory of ***xr-app***.
4. Rename the file to **'scene1_config.json'**.

We're now ready to add the scene. Make sure that you have the name of your scene and the path to its configuration (for example: **src/scene1_config.json**). The Amplify CLI prompts all these settings, including a setting for if you want to allow unauthenticated users.

Bash

```
amplify add xr

Open the Amazon Sumerian console: https://console.aws.amazon.cc
Publish the scene you want to add.
Then download the JSON configuration to your local computer.
? Press Enter when ready.
? Provide a name for the scene: scene1
? Enter the path to the downloaded JSON configuration file for
Adding XR to your project requires the Auth category.
Successfully added auth resource locally.
? Allow unauthenticated users to access your XR scene (y/N): Y
Successfully added resource scene1 locally

Some next steps:
"amplify push" builds all of your local backend resources and p
"amplify publish" builds all of your local backend and front-er
```

We're ready to create all the resources in the cloud:

Bash

```
amplify push

Current Environment: dev

| Category | Resource name    | Operation | Provider plugin    |
| -------- | --------------- | --------- | ----------------- |
| Auth     | cognito         | Create    | awscloudformation |
| Xr       | scene1          | Create    | awscloudformation |
? Are you sure you want to continue? (Y/n) Y
⠋ Updating resources in the cloud. This may take a few minutes
✓ All resources are updated in the cloud
```

Let's add the scene to the new project. Open **'src/App.js'** and replace its content with the following:

```Js
import React, { Component } from 'react';
import Amplify from 'aws-amplify';
import { SumerianScene } from 'aws-amplify-react';
import aws_exports from './aws-exports';

Amplify.configure(aws_exports);

class App extends Component {
  render() {
    return (
      <div className="App">
        <SumerianScene sceneName={"scene1"} />
      </div>
    );
  }
}

export default App;
```

We need to make some adjustments to the styles so that our scene shows as centered for all devices and screen sizes. Open **src/index.css** and add the following styles at the top:

CSS

```css
html, body {
  height: 100%;
}
```

Open **src/App.css** and replace the create-react-app default styles with the following:

CSS

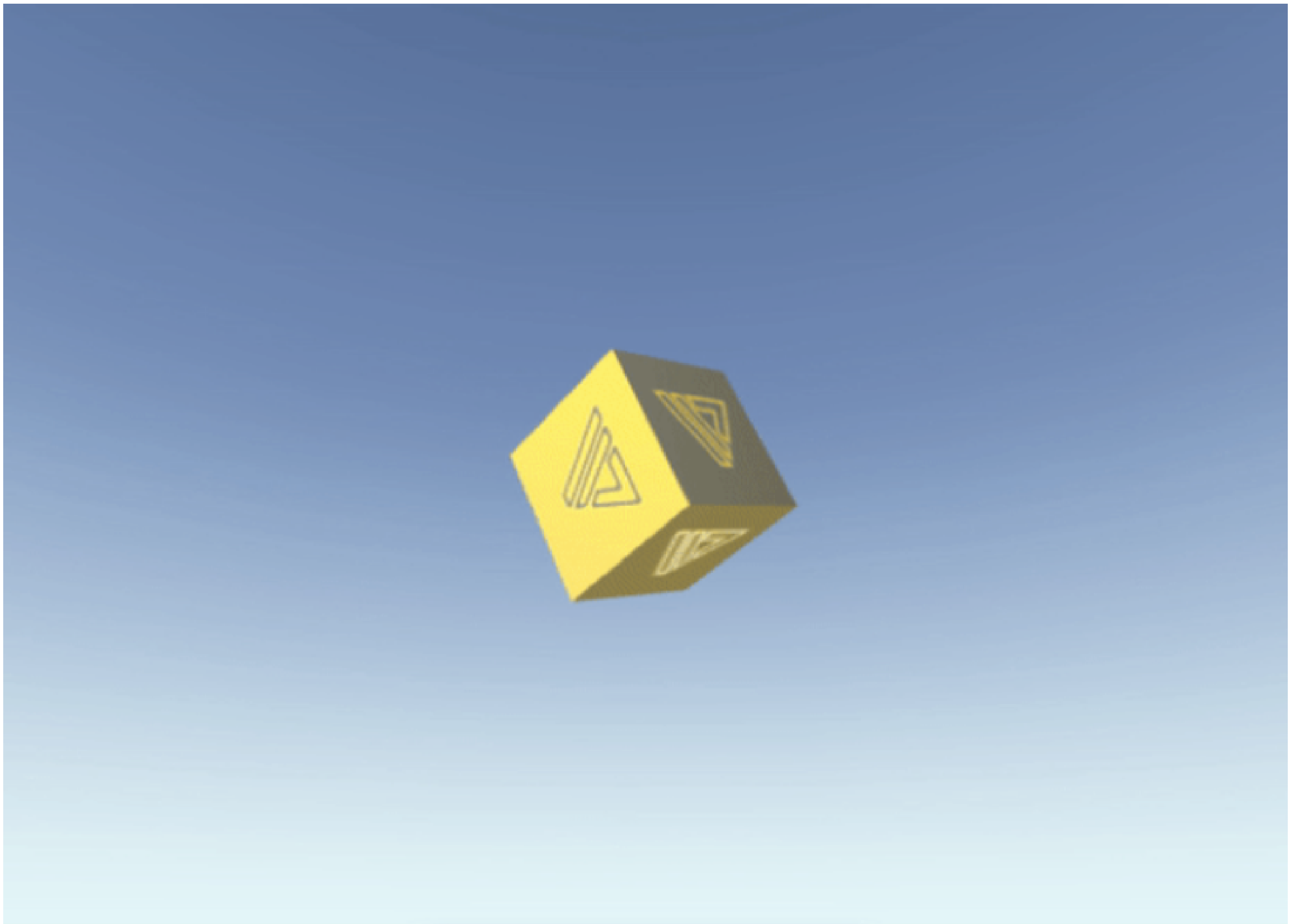
```css
.App {
  width: 100%;
  height: 100%;
  display: flex;
  justify-content: center;
  align-items: center;
}
```

To try out the scene, we can run our project with the following:

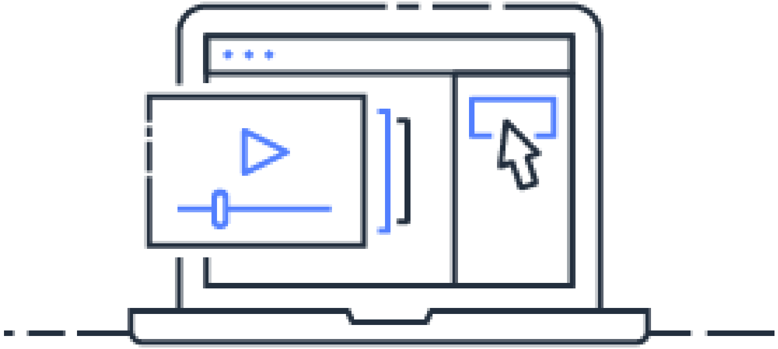
Bash

```bash
npm start
```

Check the final output at http://localhost:3000. The following image is only for demonstration purposes.



We hope you like these new features! As always, let us know how we’re doing, and submit any requests in the Amplify Framework GitHub Repository. You can read more about AWS Amplify on the AWS Amplify website.



AWS Training & Certifications

Free digital courses to help you develop your skills

Learn more »

AWS Podcast

Subscribe for weekly AWS news and interviews

Learn more »

AWS Partner Network

Find an APN member to support your cloud business needs

Learn more »

Resources

AWS AppSync

AWS Device Farm

Amazon Pinpoint

AWS Amplify

Follow

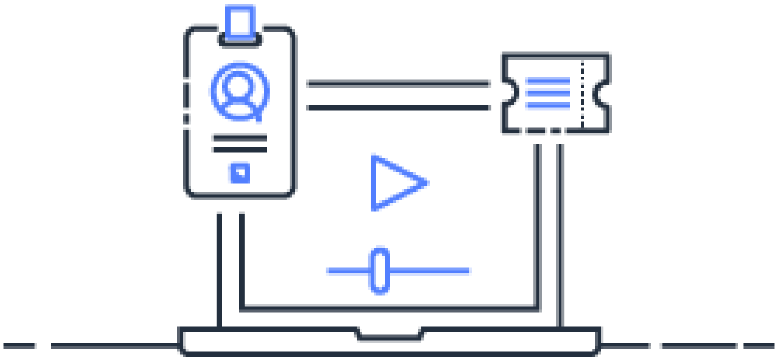
AWS for Mobile

Facebook

LinkedIn

Twitch

✉ Email Updates



AWS Events

Discover the latest AWS events in your region

Learn more »

Related Posts

Texas statewide collaboration with AWS Educate announced by Secretary of State Ruth Hughs to provide education-to-workforce opportunities

Simplify Your AWS SFTP Structure with Chroot and Logical Directories

Enabling workforce development and tech for good at the AWS Summit Bahrain: recap

Data Migration Best Practices with AWS Snowball Edge

Free AWS Toronto Summit l Oct 3, 2019 – Explore AWS Storage

[illegible]

Now Available – EC2 Instances (G4) with NVIDIA T4 Tensor Core GPUs

Integrating an Inferencing Pipeline with NVIDIA DeepStream and the G4 Instance Family

Sending Push Notifications to iOS 13 and watchOS 6 Devices