AUTODESK UNIVERSITY

Developing Web Apps with Forge Data Exchange and AWS Amplify

Jon Clark, Sr. Partner Solutions Architect, AWS
Enrico Chionna, BIM Technology Manager, AWS

September 2022

Who we are…

Enrico Chionna
BIM Technology Manager
AWS

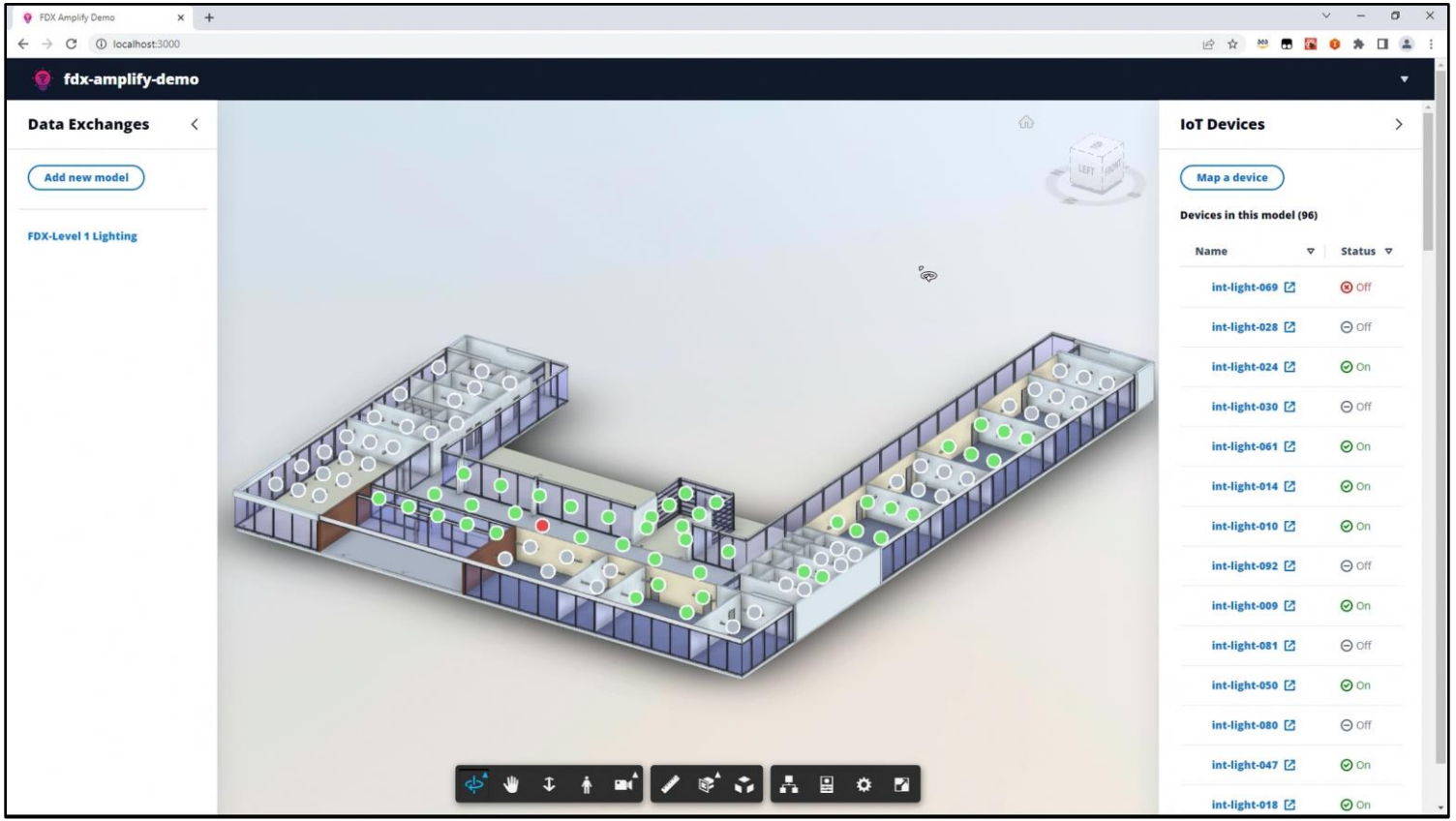
Jon Clark
Sr. Partner Solutions Architect
AWS

*"Developing Web Apps with **Forge Data Exchange** and **AWS Amplify**"*

Key learning objectives

1. Understand the use of Autodesk Forge Data Exchange
2. Integrate disparate data into a single application
3. Build an application using AWS and Autodesk Forge
4. Identify new use cases for Autodesk Forge Data Exchange

What we'll be demoing



What we'll be demoing

AUTODESK UNIVERSITY

Data Exchange Overview

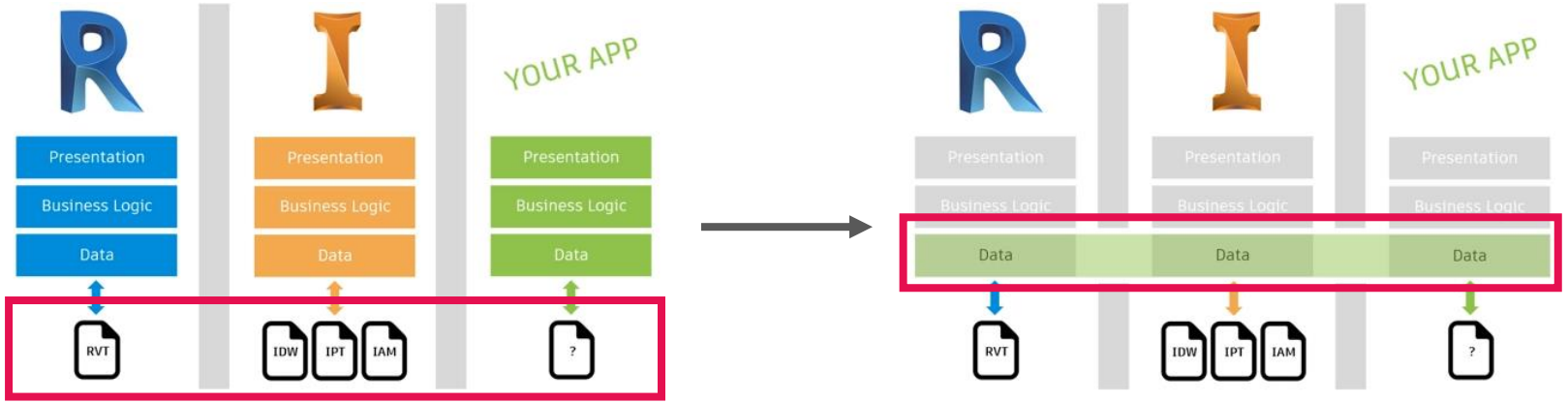
What is Autodesk Data Exchange?

How does Data Exchange work with Forge API?

aws

Forge API + Data Layer

Forge is a platform of web service APIs that enable you to embed Autodesk components into your web or mobile applications.



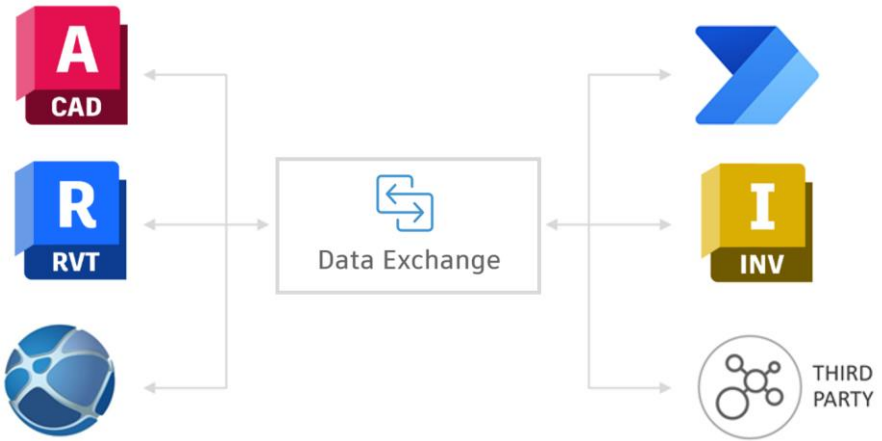
Monolithic binary files

Binary files with a ***Data Layer***

Forge Data Exchange

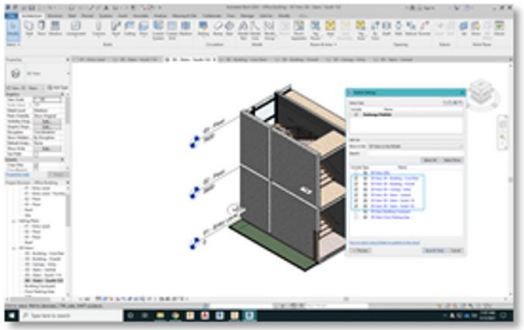
Share only the portions of your data that you need to share

- Works in Autodesk Docs, Revit 2023, and Inventor 2023
- Available in public beta on April 22, 2022
- Data Exchanges are built on the Forge Data platform



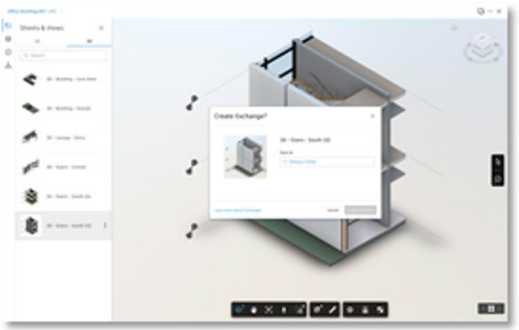
Forge Data Exchange

REVIT



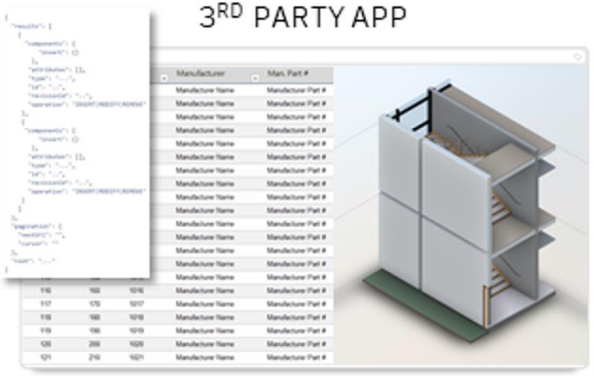
Revit user publishes a model that includes 3D views

AUTODESK DOCS



Autodesk Docs user selects a 3D view to create an exchange from

3RD PARTY APP



3rd party app uses Data Exchange Read APIs to load in property data from the exchange

RVT → AUTODESK Construction Cloud → AWS Amplify

AUTODESK UNIVERSITY

AWS Amplify

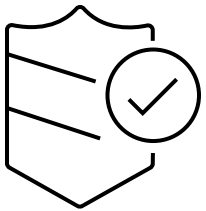
Build full-stack web and mobile apps.

aws

What is AWS?

The most flexible and secure cloud computing environment.

Security



Security is priority #1

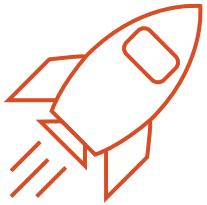
98 security standards and compliance certifications

Shared security model

Cost Optimized

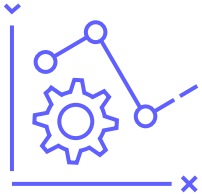
Pay only for what is consumed

Agility



Innovate more quickly and more frequently

Elasticity

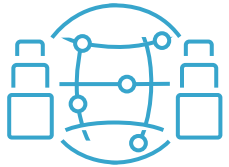


Automatically scale up or down with workload demands

Flexibility

200+ AWS services supporting any cloud workload

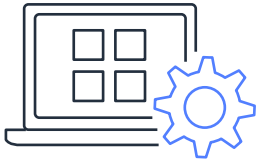
Go Global



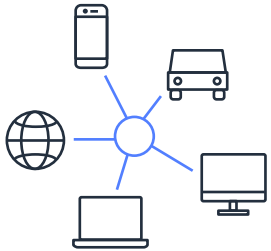
Deploy apps quickly across 26 geos and 84 availability zones

AWS Amplify provides tools to build, ship, and scale your app

Build



Configure application backend

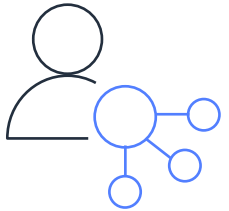


Build features and connect frontends to AWS Services

Ship

Deploy your app frontend and backend

Scale



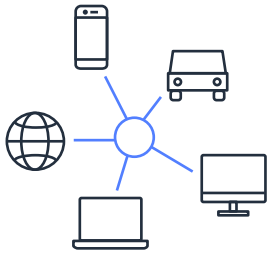
Customize and integrate with CDK

BUILD with AWS Amplify

Backend feature categories

Build

Configure AWS backends fast



Seamlessly connect frontends

CLI Studio Libraries

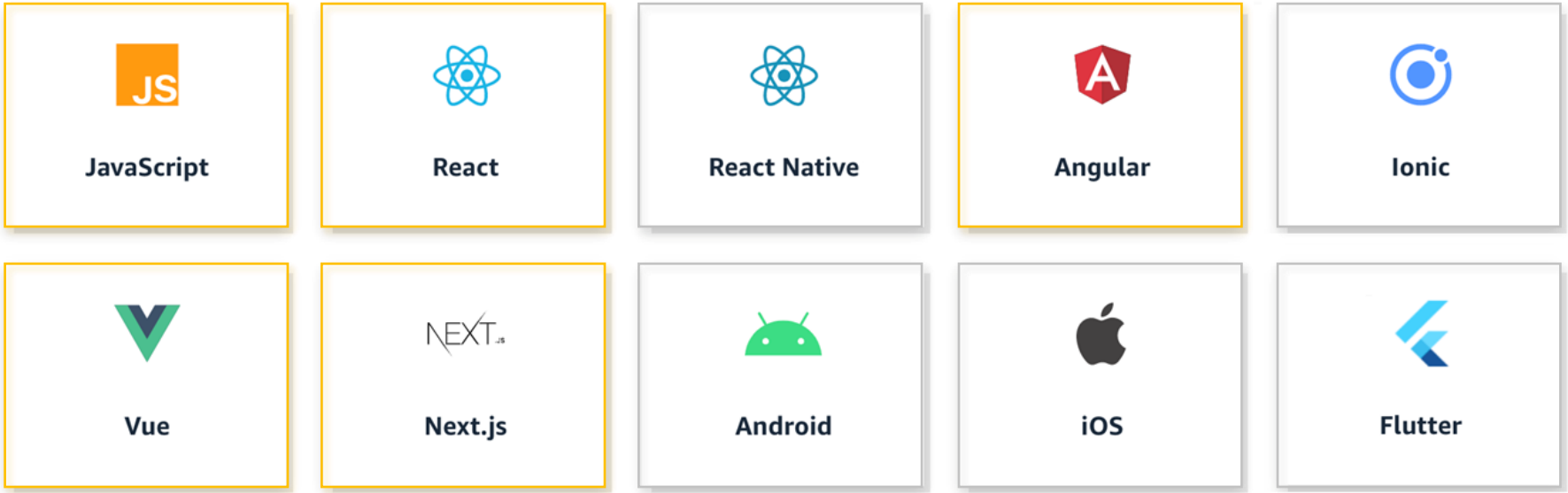
Feature categories

- **Authentication**
- DataStore
- Storage
- **API (GraphQL & REST)**
- Functions
- Geo
- Analytics
- **PubSub**
- Predictions
- Interactions
- Notifications
- Extensibility

Powered by AWS services such as
AWS AppSync (GraphQL API) and Amazon Cognito (Authentication)

BUILD – Amplify Libraries

Frontend libraries for Web, iOS, Android, React Native, and Flutter



AUTODESK UNIVERSITY

AWS Amplify
Backend development

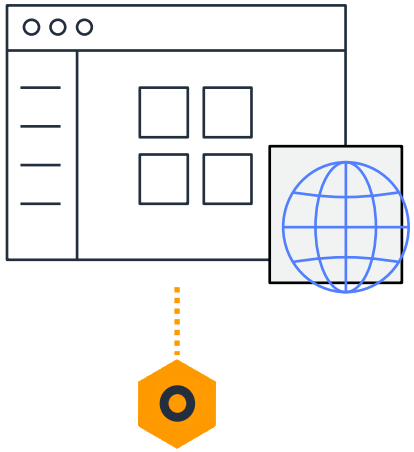
aws

BUILD – Environments

Toolchain to create, integrate, and manage the AWS cloud services for your app.

Amplify Studio

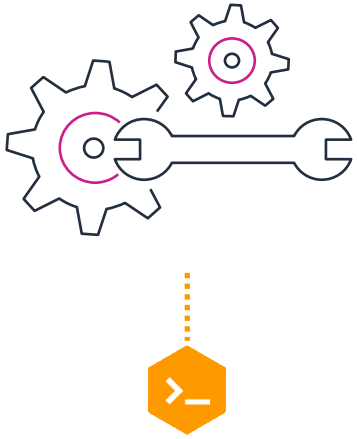
Visual development environment



Studio

Amplify CLI

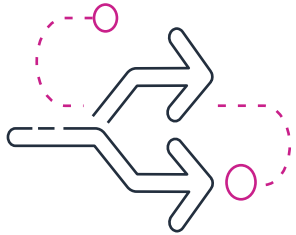
Full-stack CI/CD for developers



CLI

Amplify + CDK

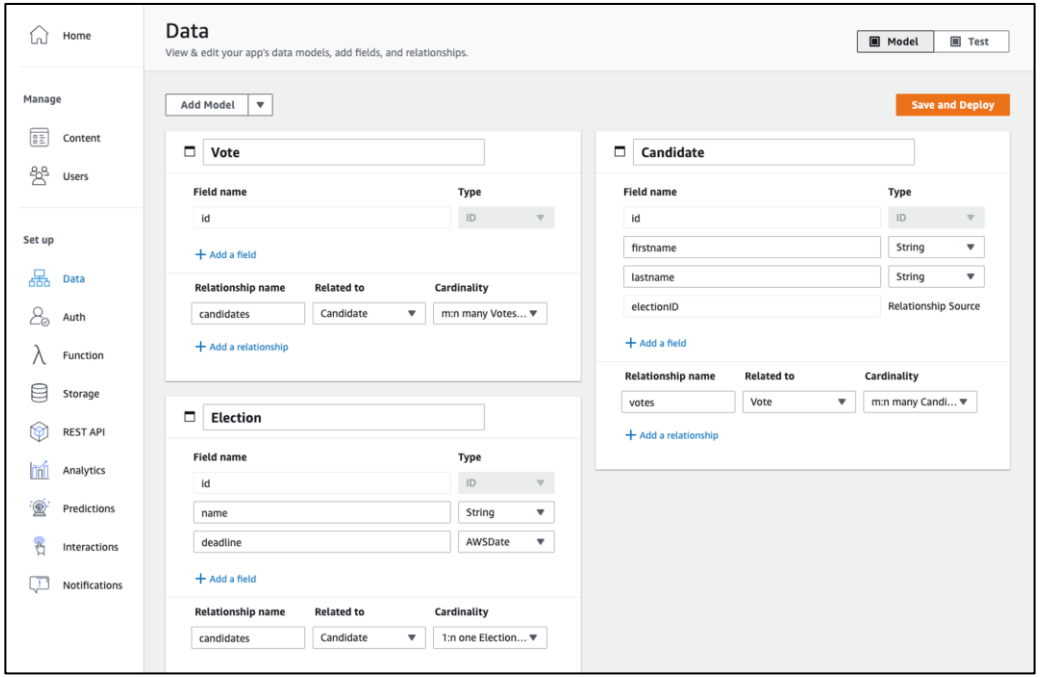
Access to 175+ AWS resources



BUILD – Amplify Studio

Configure your backend app visually

- Model data with GraphQL API and DynamoDB tables
- Manage users, groups, file storage
- Configure authentication
- Extend with CLI toolchain to add functions



BUILD – Amplify CLI

Quickly create backends for your web and mobile applications

- Full-stack CI/CD for developers
- Model data with GraphQL API and DynamoDB tables
- Manage users, groups, file storage
- Configure authentication
- Transform schema definitions to GraphQL API and DynamoDB tables
- Codegen for Swift (iOS), Java (Android), and JavaScript

```
type Blog @model {
  id: ID!
  name: String!
  posts: [Post] @connection(name: "BlogPosts")
}
type Post @model {
  id: ID!
  title: String!
  blog: Blog @connection(name: "BlogPosts")
  comments: [Comment] @connection(name: "PostComments")
}
type Comment @model {
  id: ID!
  content: String
  post: Post @connection(name: "PostComments")
}
```

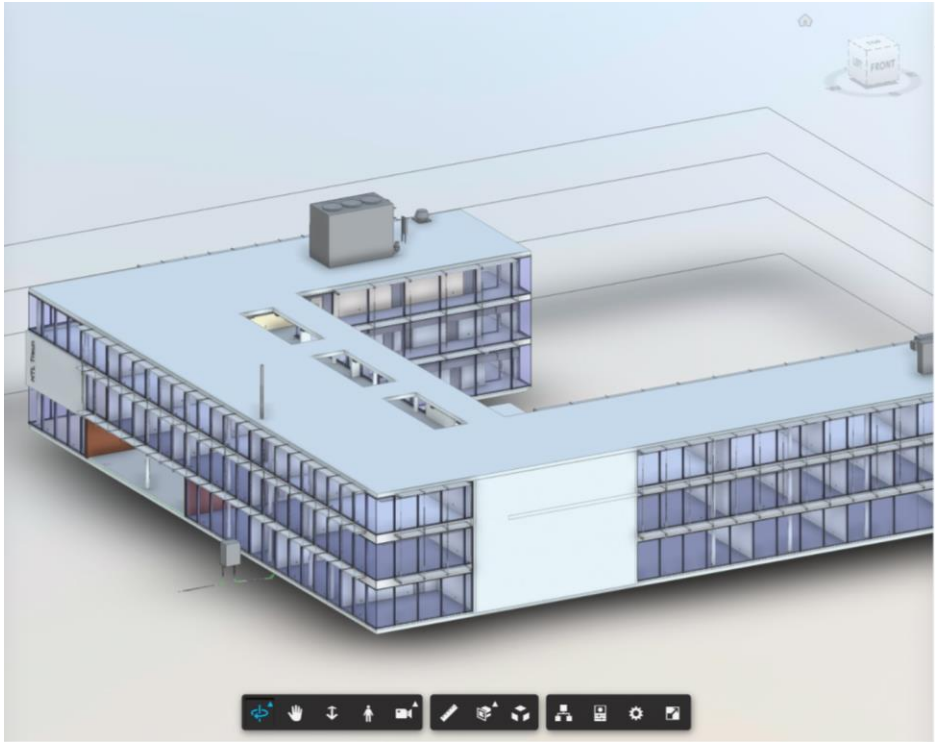
AUTODESK UNIVERSITY

AWS Amplify UI Development

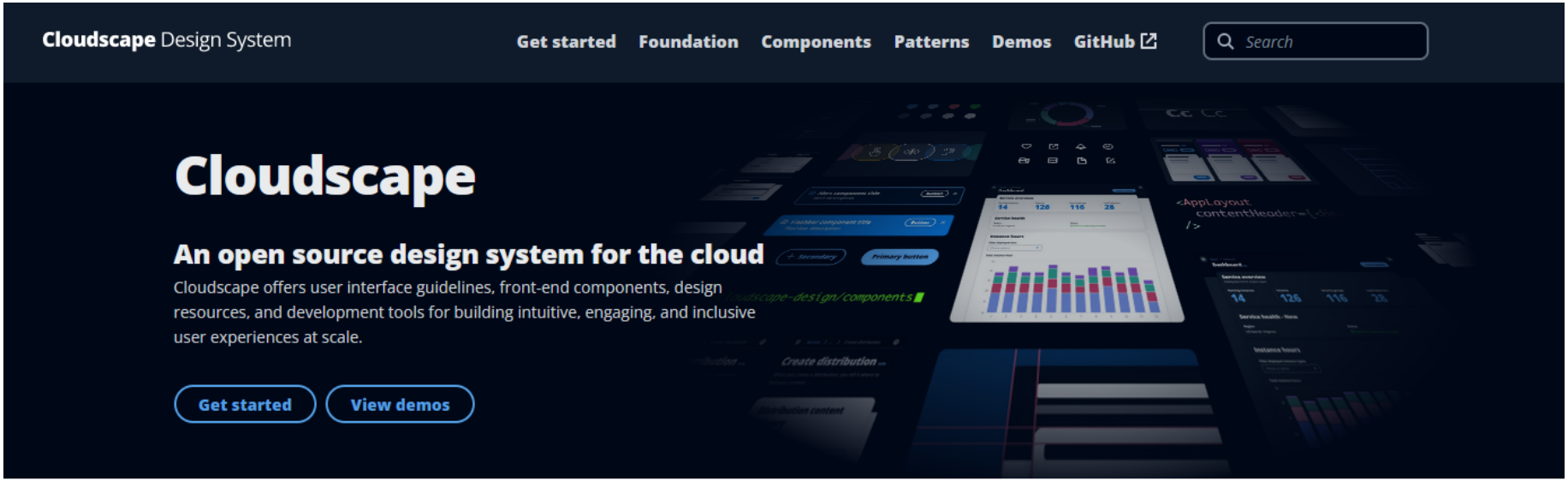
aws

Autodesk Forge Viewer

- A WebGL-based, JavaScript library for 3D and 2D model rendering.
- Model data may come from AutoCAD, Fusion 360, Revit, etc.
- Compatible with all modern browsers



Cloudscape - https://cloudscape.design



Foundation

Learn about the visual foundation of our design system, such as colors, spacing, and iconography.

Components

Discover our library of components. Built with React, they are tested, accessible, and responsive.

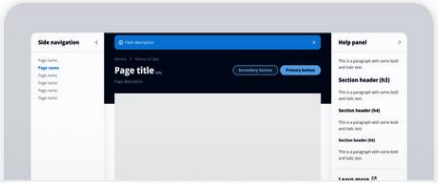
Patterns

Explore our guidelines and best practices for building user experiences.

Demos

View Cloudscape in action by browsing through our demos.

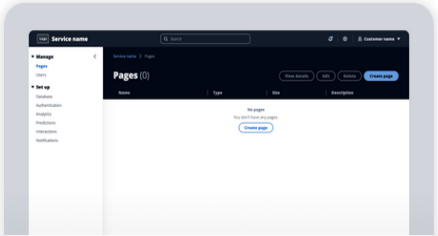
Cloudscape - https://cloudscape.design



App layout

Provides the basic layout for all types of pages, including collapsible side navigation, tools panel, and split panel.

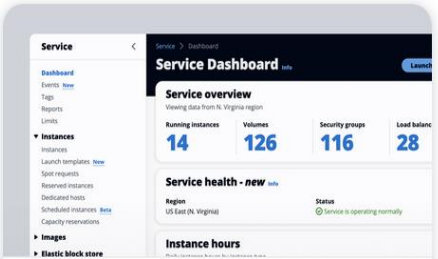
- Basic layout
- Collapsible panels



Top navigation

The top navigation, when paired with side navigation and breadcrumbs, forms a global, scalable navigation system

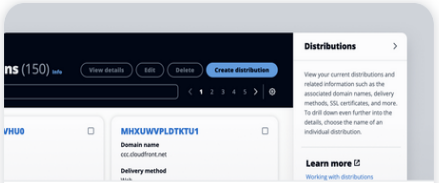
- Navigation
- Breadcrumbs



Side navigation

A list of navigational links that point to the pages within an application.

- Navigational links



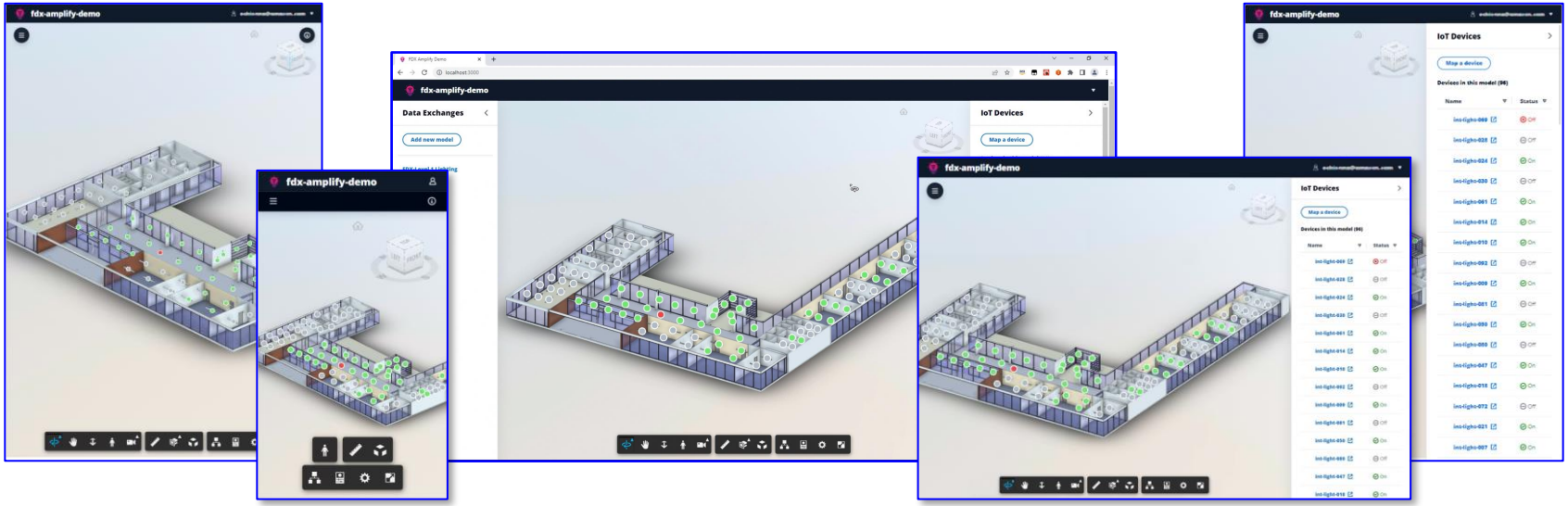
Help panel

The panel displays help content that relates to a concept, term, setting, option, or task within the main page content.

- Contextual help content

Responsive

- Cloudscape React components are responsive
- Responsive web apps adapt to different screen sizes, resolutions, and orientations



Single-Page vs. Progressive vs. Mobile Apps

- **Single-Page Application** (SPA)
  - A website that loads content dynamically within a single page.
  - Runs within a single Document Object Model (DOM).
  - Requires network connection.

- **Progressive Web App** (PWA)
  - A web application designed to run as if it’s a native mobile app.
  - Installable
  - Uses background workers that load cached data even without a network.
- **Native Mobile App**
  - Built to run on a specific mobile OS (iOS or Android).
  - Network connection not required.

Single-Page vs. Progressive vs. Mobile Apps

	Single Page App	Progressive Web App	Native Mobile App
Platform	Any browser, any device	Any browser, any device	Device OS
Language	HTML, CSS, JavaScript	HTML, CSS, JavaScript	Mobile-specific
Building and updating	Faster	Faster	Slower
Hosting/distributing	http or https hosting	https hosting (required)	App Store
Search engine	None	SEO	ASO
Installable	No	Yes	Yes
Offline-able	No	Yes	Yes

AUTODESK UNIVERSITY

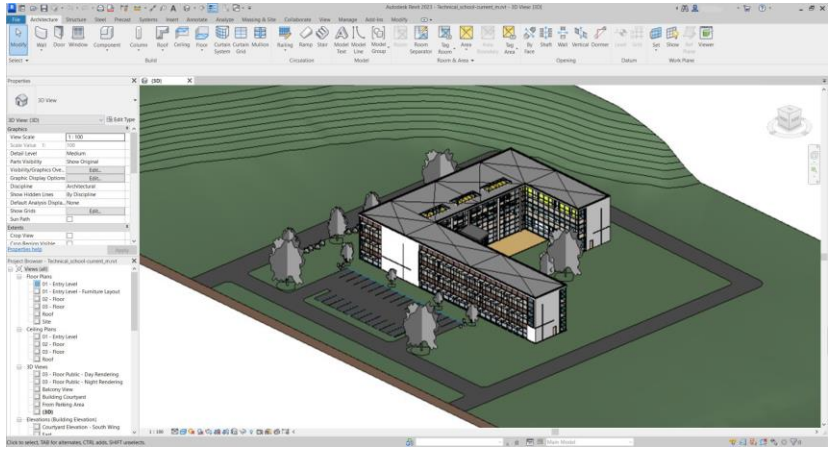
Use Case Spotlight

One example of a Data Exchange use case

aws

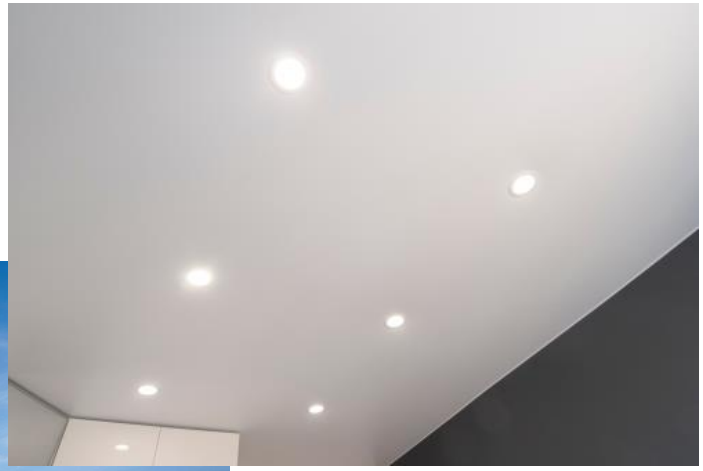
Use case - Architect

Architect



AUTODESK
Construction Cloud

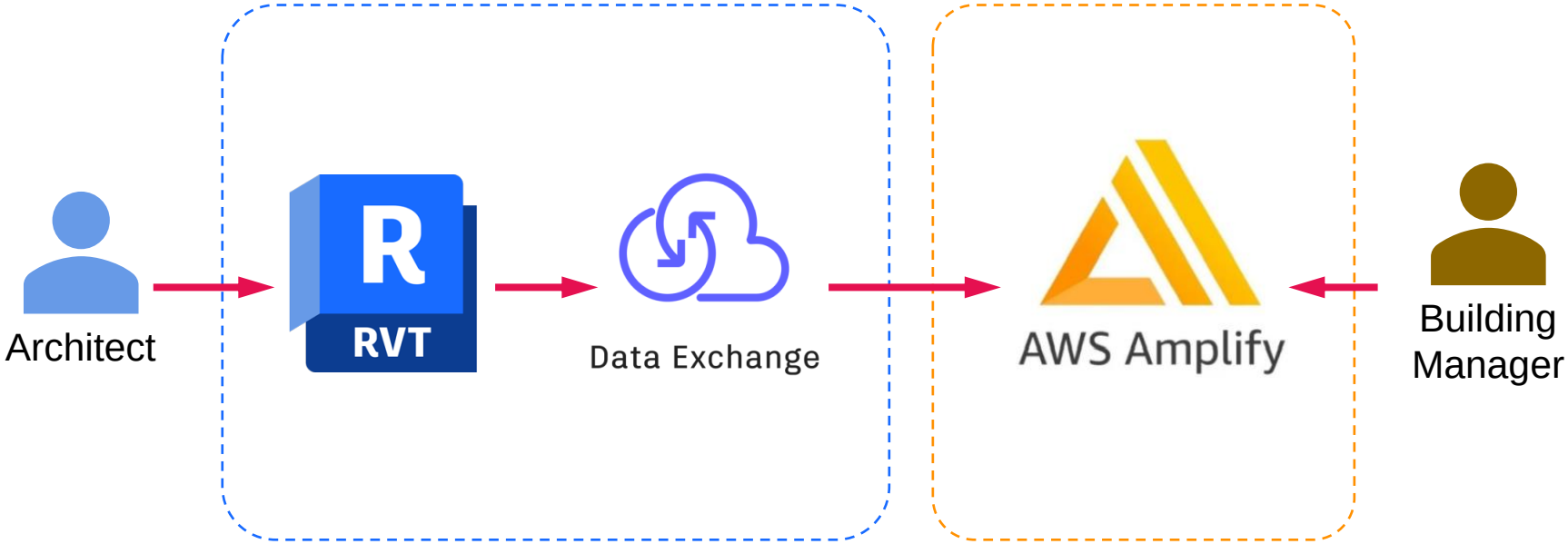
Use case – Building Manager



Building Manager

Use case – Data Exchange

Data Exchange – Amplify workflow



Share specific subsets of Revit data with third party applications

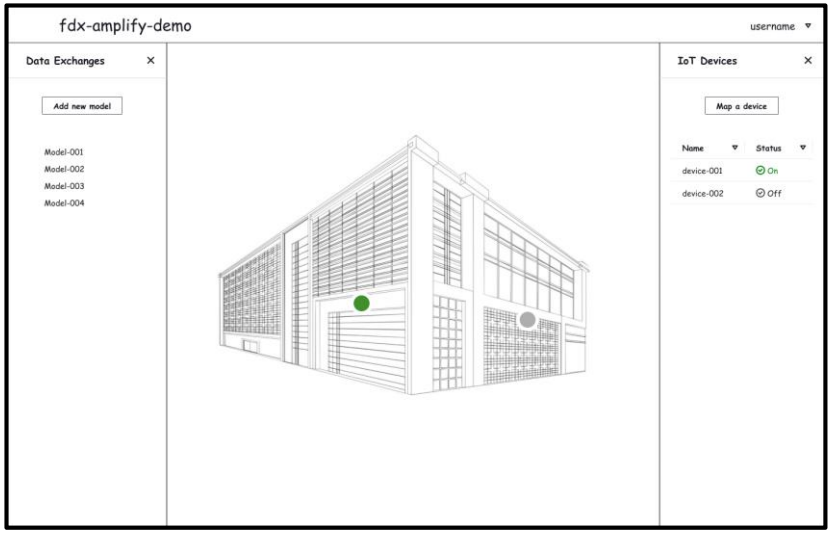
AUTODESK UNIVERSITY

App Development Workflow

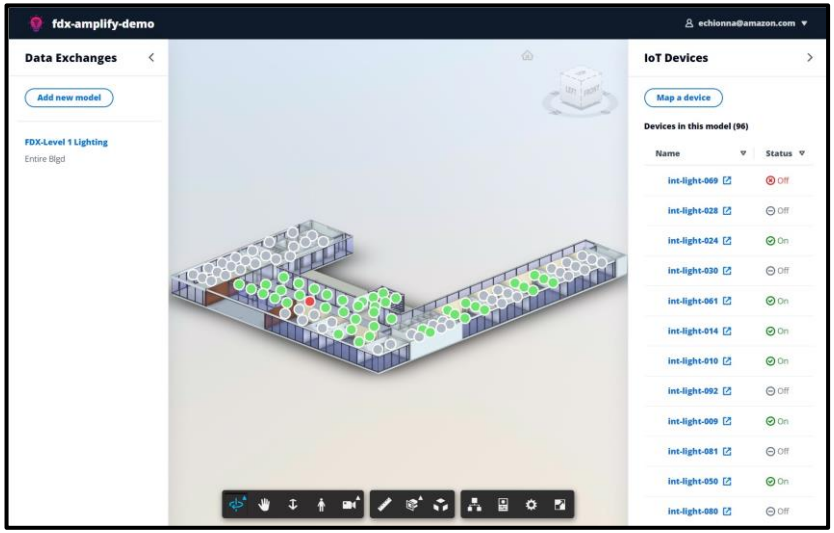
aws

App UI Design

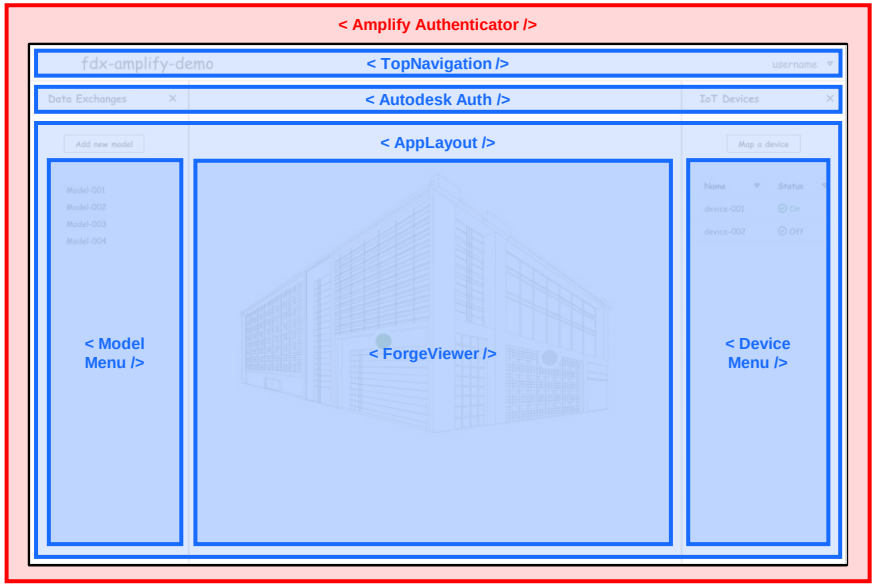
Wireframe

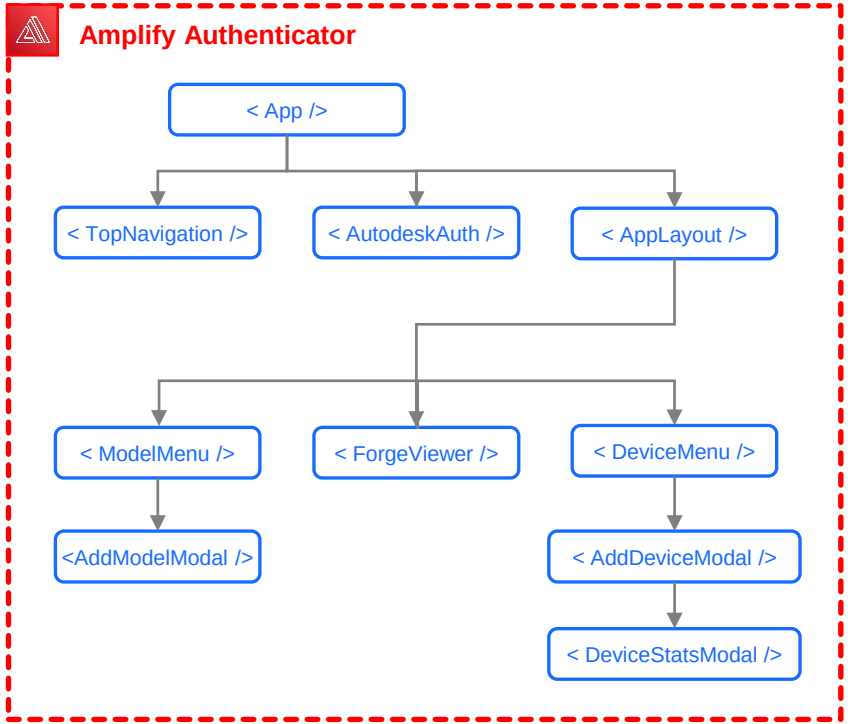


High-fidelity



React UI Components





Forge API Used

Authentication

- Get 3-legged token with implicit grant

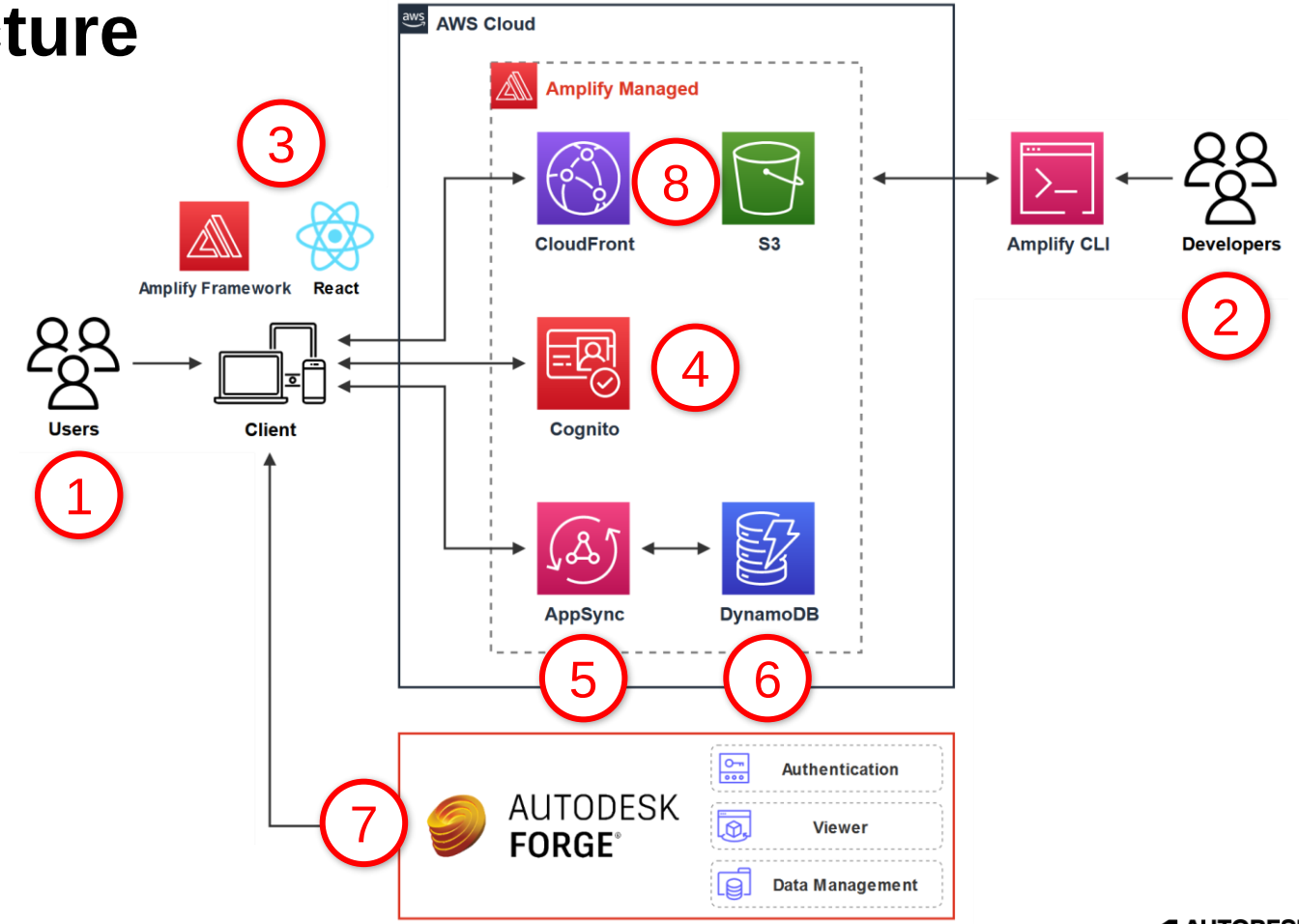
Viewer

- Load and display models

Data Management

- Get latest model version URN

Architecture



Prerequisites

- **Node.js** v14.x or later
- **AWS Account**
- **Npm** v6.14.4 or later
- **Autodesk Account**
- **Git** v2.14.1 or later
- **Forge Account**

Steps to Create an Amplify Project

1. Install Amplify CLI
2. Configure Amplify
3. Create a New React App
4. Initialize a New Amplify Project
5. Connect Amplify to the Frontend

Install & Configure Amplify CLI

1

```
npm install -g @aws-amplify/cli
```

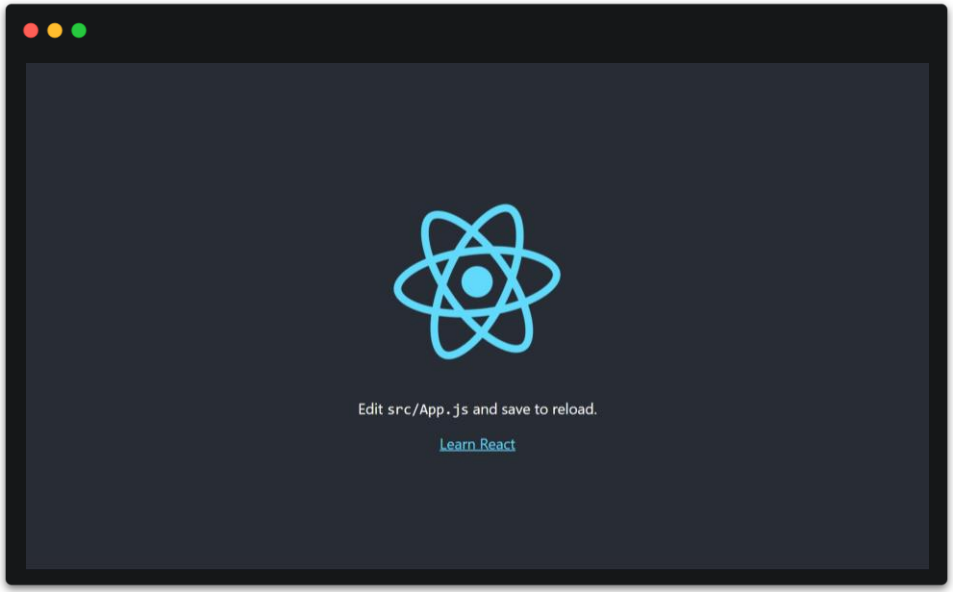
2

```
amplify configure

...

Specify the AWS Region
? region:  # Your preferred region
Specify the username of the new IAM user:
? user name:  # User name for Amplify IAM user
Complete the user creation using the AWS console

...

Enter the access key of the newly created user:
? accessKeyId:  # YOUR_ACCESS_KEY_ID
? secretAccessKey:  # YOUR_SECRET_ACCESS_KEY
This would update/create the AWS Profile in your local machine
? Profile Name:  # (default)

Successfully set up the new user.
```

Create a New React App

3

```
npx create-react-app fdx-amplify-demo
cd fdx-amplify-demo
npm start
```



Initialize a New Amplify Project

4

```
amplify init
```

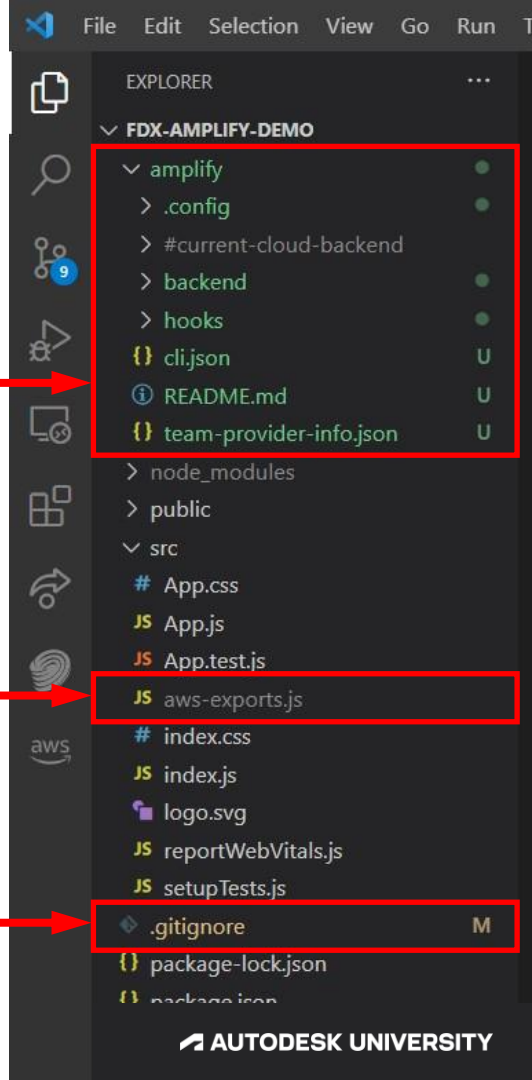
```
Enter a name for the project (amplified) # Your project name

The following configuration will be applied:

Project information
| Name: amplified
| Environment: dev
| Default editor: Visual Studio Code
| App type: javascript
| Javascript framework: react
| Source Directory Path: src
| Distribution Directory Path: build
| Build Command: npm run-script build
| Start Command: npm run-script start

? Initialize the project with the above configuration? Yes

? Select the authentication method you want to use:
❯ AWS profile
  AWS access keys

? Please choose the profile you want to use:
❯ YOUR_PROFILE # profile created earlier with `amplify configure`
  OTHER_PROFILE
```

Amplify Generated Files

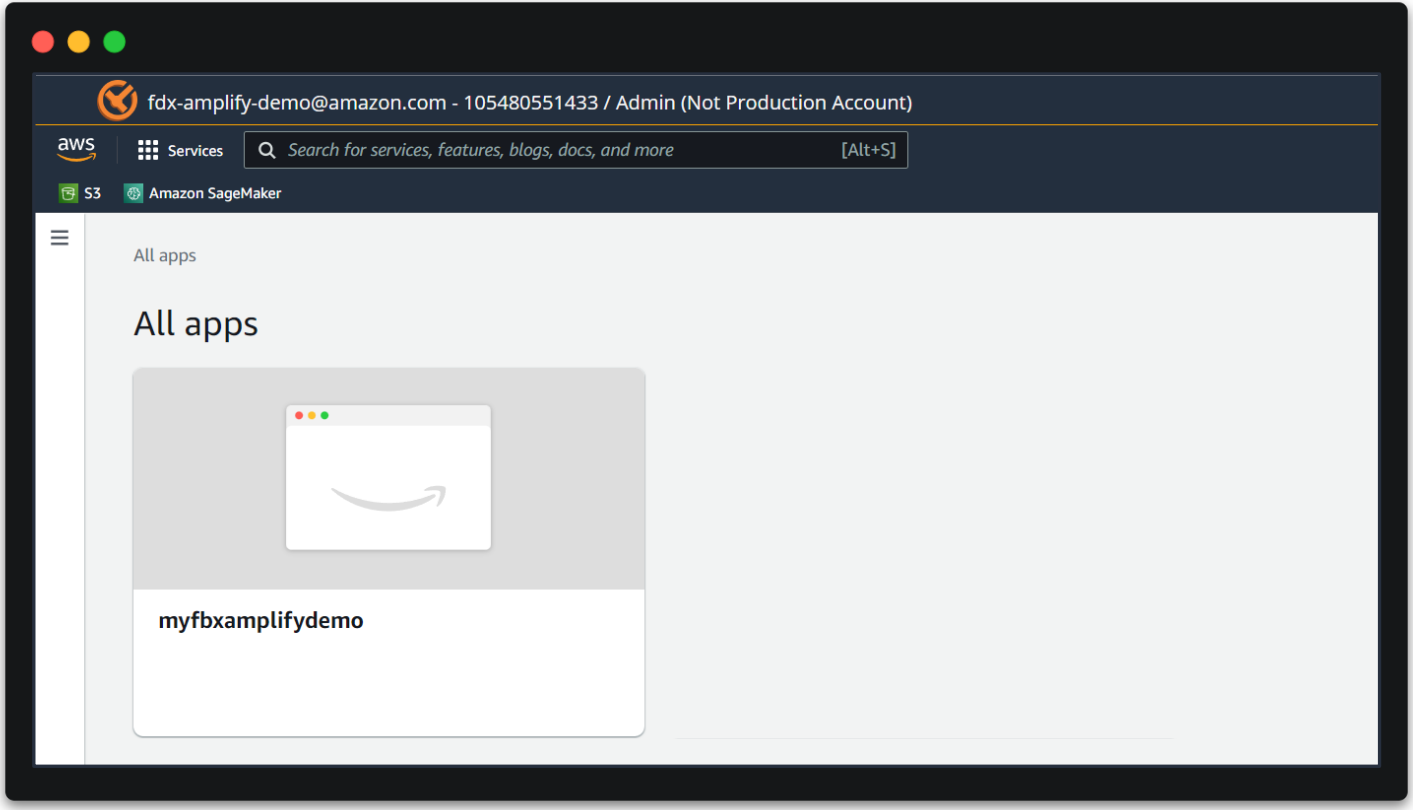
creates top level directory called `amplify` with backend definition

`aws-export.js` holds configuration for the backend services

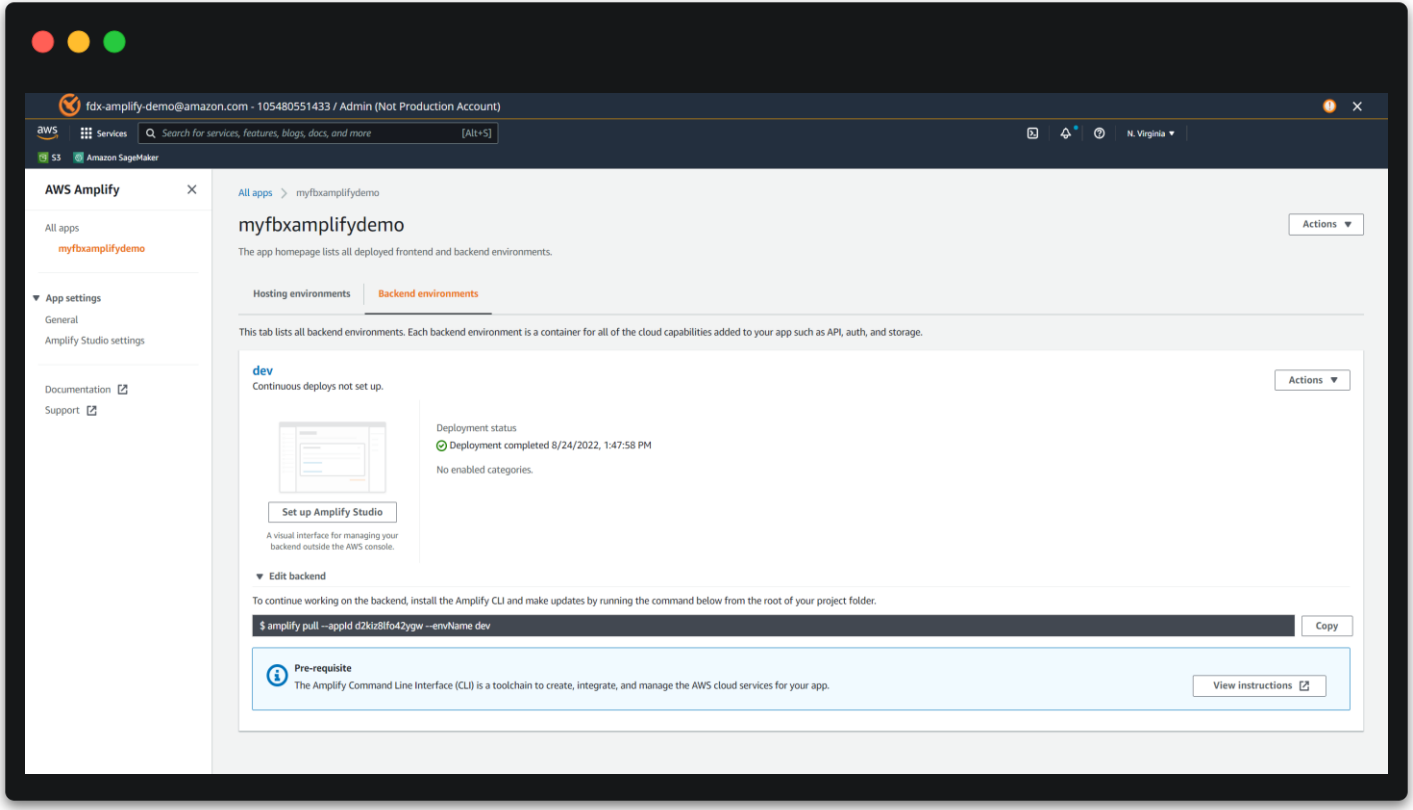
`.gitignore` adds some Amplify generated files to the ignore list



Amplify Generated Application in AWS Console



Amplify Generated Backend in AWS Console

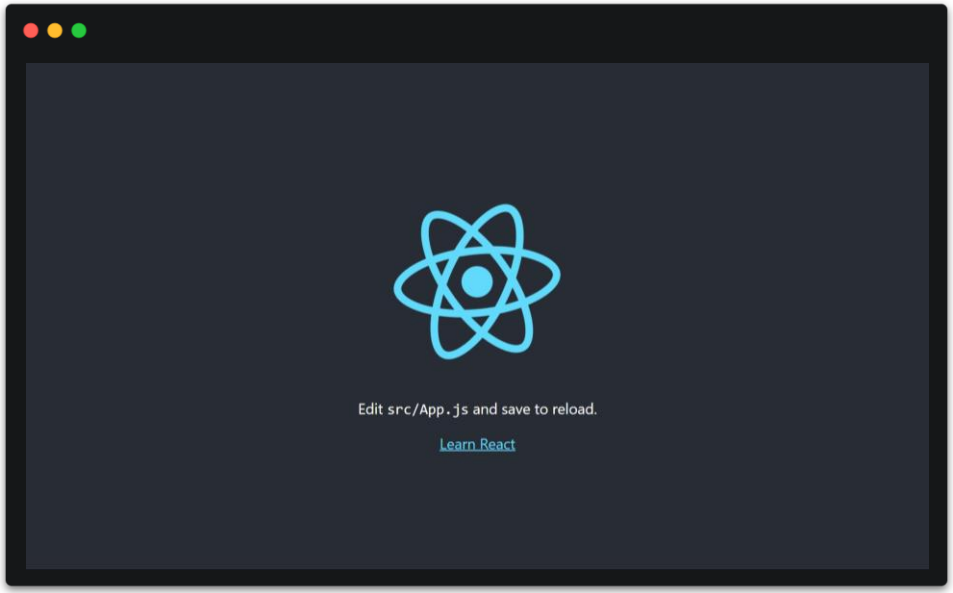


Connect Amplify to the Frontend

src/index.js

5

```
import { Amplify } from 'aws-amplify';
import awsExports from './aws-exports';
Amplify.configure(awsExports);
```



AUTODESK UNIVERSITY

App Development

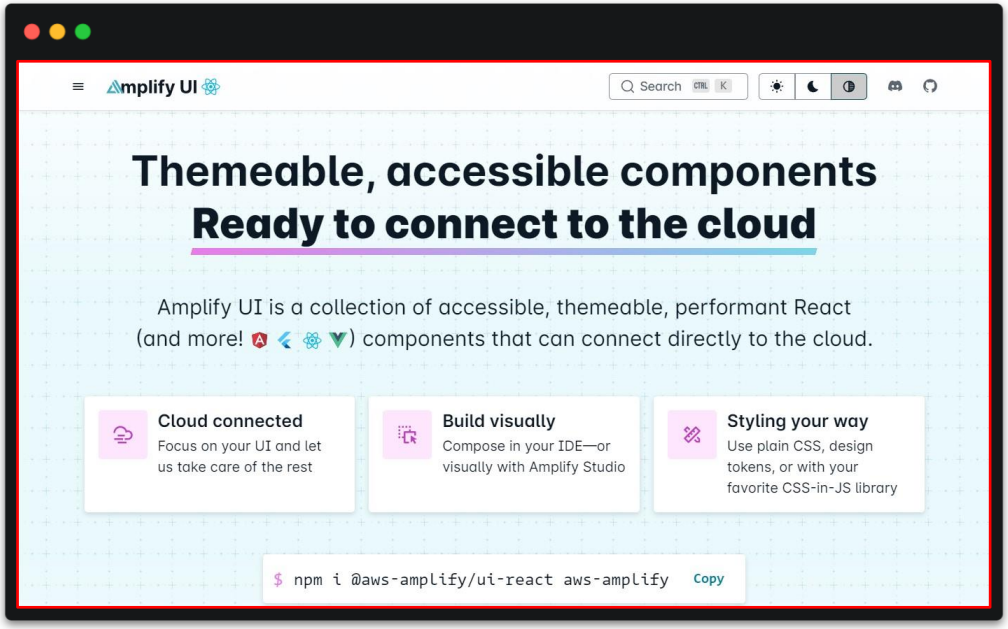
Adding Services: Authentication, API, Forge Viewer

aws

Install Amplify UI Package

https://ui.docs.amplify.aws/

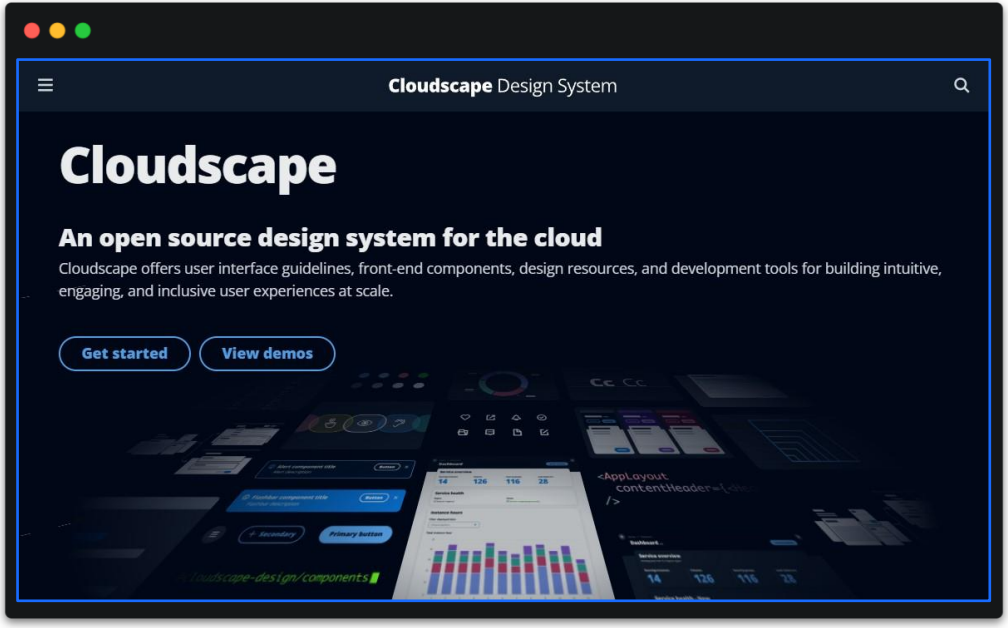
```
npm install aws-amplify @aws-amplify/ui-react
```



Install Cloudscape Packages

https://cloudscape.design/

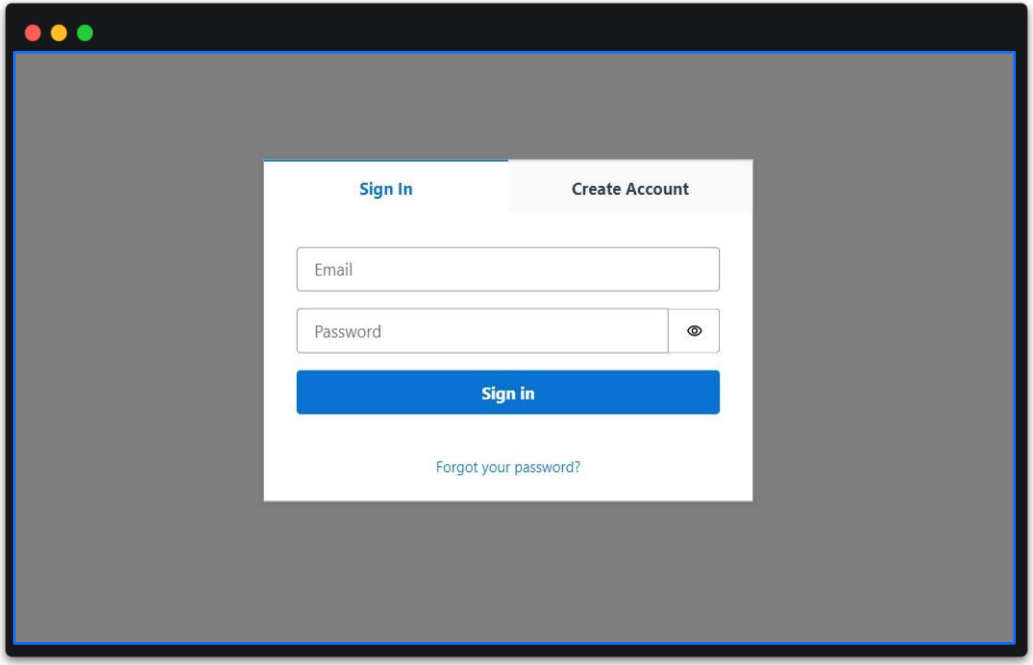
```
npm install @cloudscape-design/global-styles
npm install @cloudscape-design/components
npm install @cloudscape-design/design-tokens
```



Add Authentication

- **Sign in**
  - Sign in with email and password
- **Sign up**
  - Sign up with email and password
  - Verify email with verification code
- **Recover password**
  - Reset forgotten password
  - Verify email with verification code
- **Custom styling**

Add Authentication



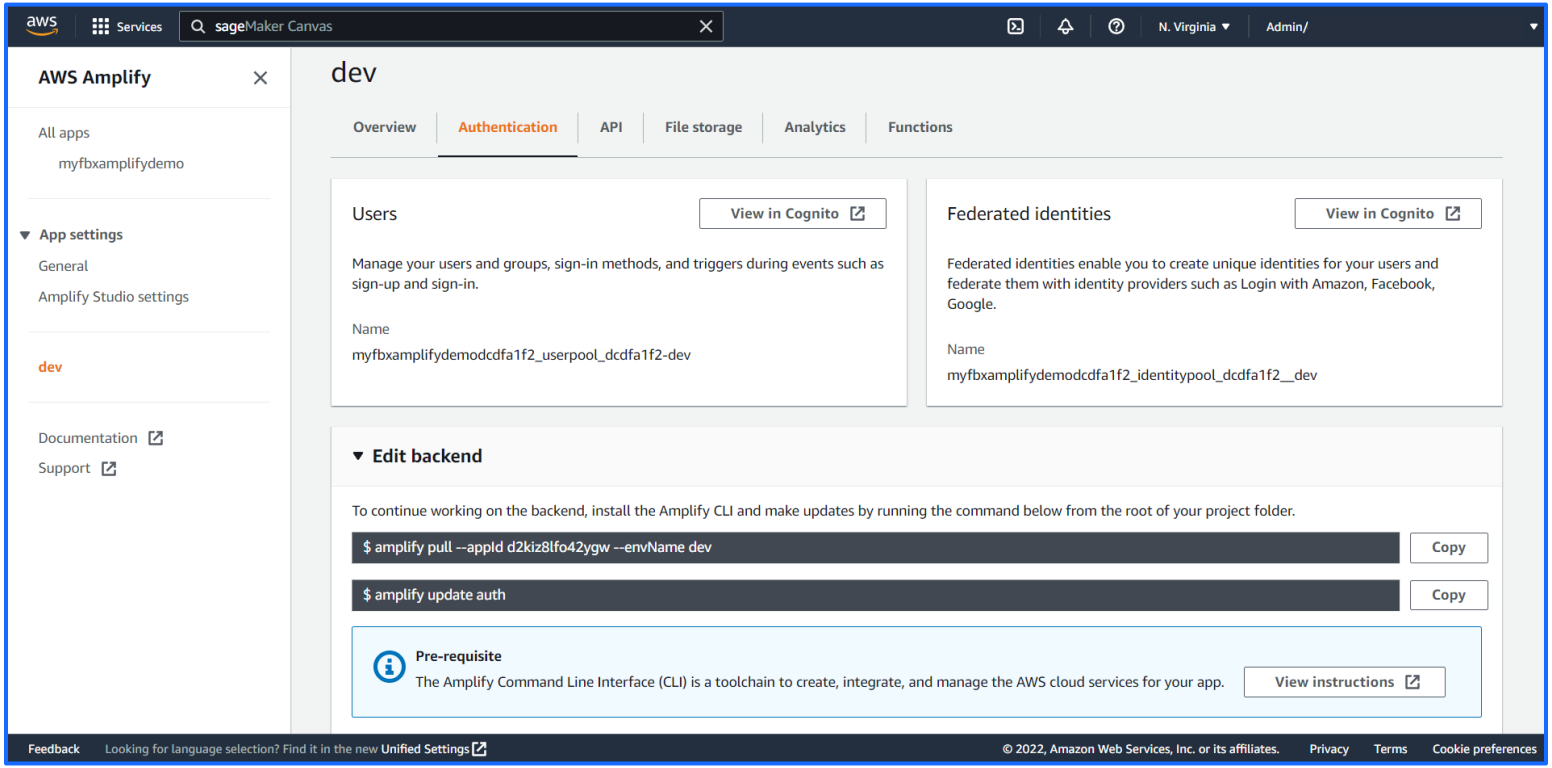
Create Authentication Service

```
amplify add auth

? Do you want to use the default authentication and security
configuration? Default configuration
? How do you want users to be able to sign in? Email
? Do you want to configure advanced settings?  No, I am done.
```

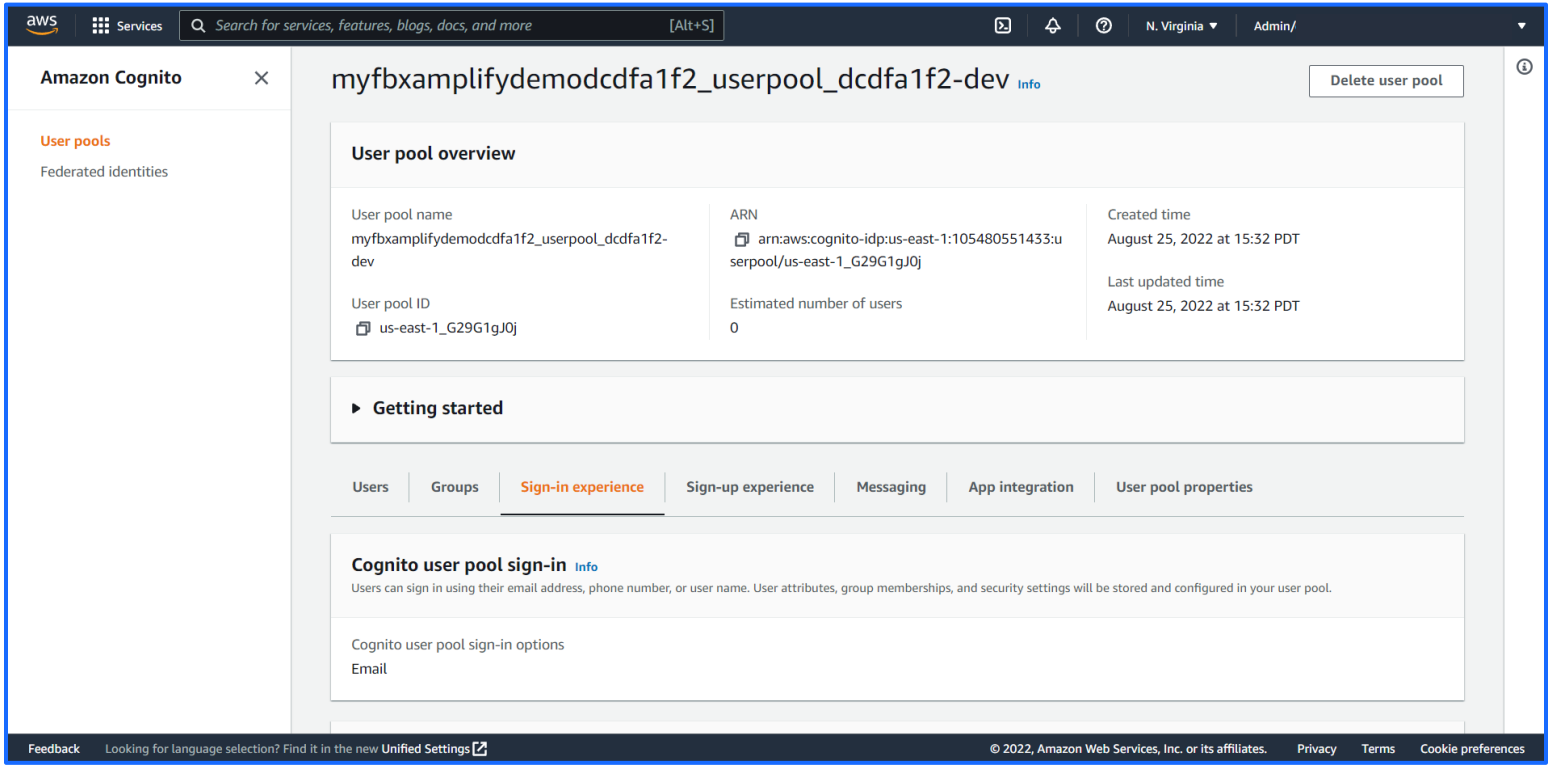
```
amplify push
```

```
amplify console
```

Create Authentication Service



Create Authentication Service



Create Login UI

src/app.js (before)

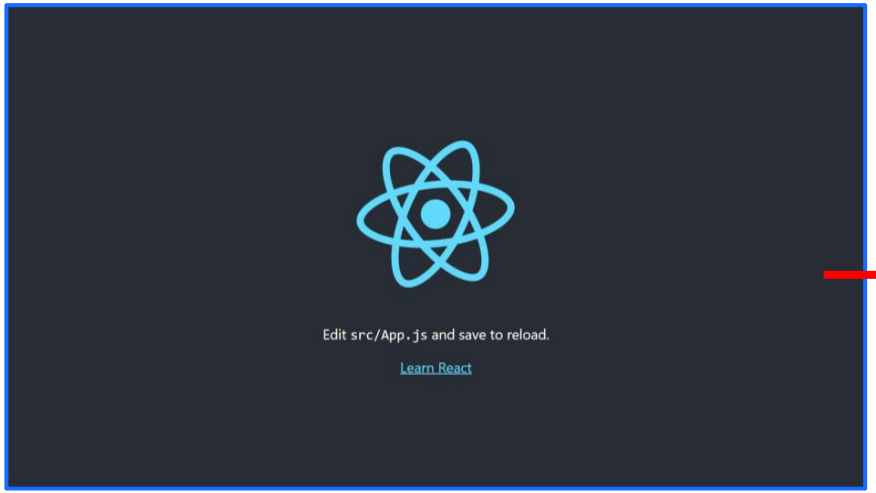
```
import logo from './logo.svg';
import './App.css';

function App() {
  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>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default App;
```

src/app.js (after)

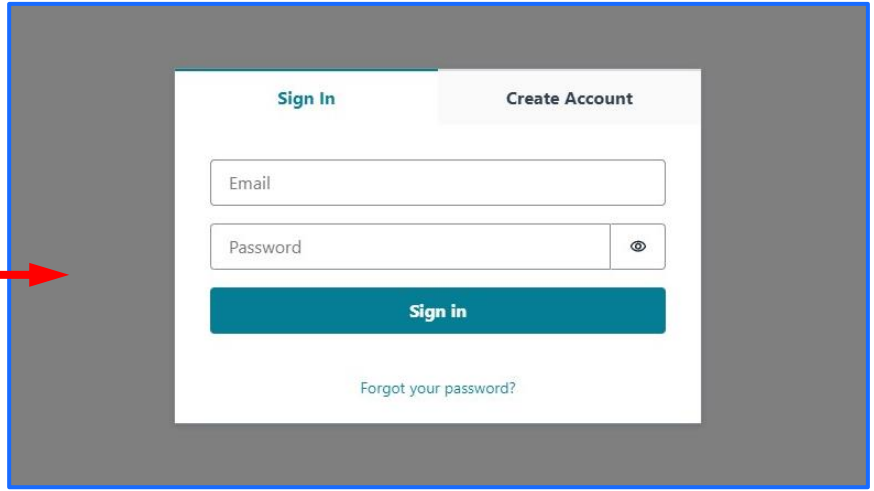
```
import logo from "./logo.svg";
import "./App.css";
import { withAuthenticator } from "@aws-amplify/ui-react";
import "@aws-amplify/ui-react/styles.css";

function App({ signOut, user }) {
  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>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default withAuthenticator(App);
```

Create Login UI

before



after



Login UI Custom Theme

https://ui.docs.amplify.aws/react/theming

src/index.js (before)

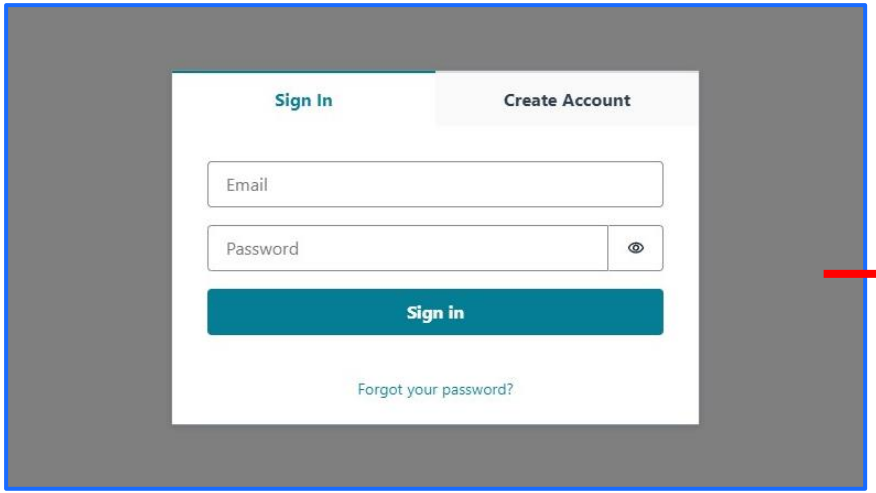
```
...
import { Amplify } from "aws-amplify";
import awsExports from "./aws-exports";
Amplify.configure(awsExports);

const root =
ReactDOM.createRoot(document.getElementById("root"));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

src/index.js (after)

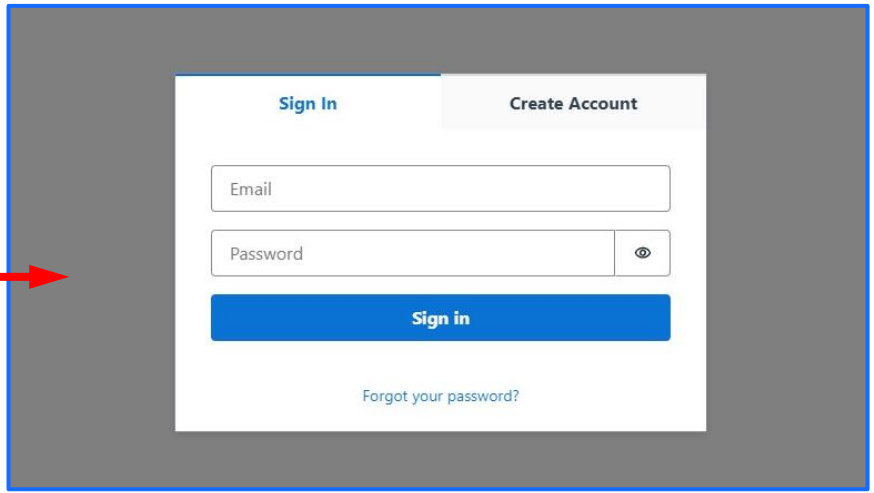
```
...
import { ThemeProvider } from "@aws-amplify/ui-react";
import * as awsui from "@cloudscape-design/design-tokens";
...

const theme = {
  name: "cloudscape-design",
  tokens: {
    colors: {
      brand: {
        primary: {
          80: { value: awsui.colorBackgroundButtonPrimaryDefault },
        },
      },
    },
  },
};

const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(
  <React.StrictMode>
    <ThemeProvider theme={theme}>
      <App />
    </ThemeProvider>
  </React.StrictMode>
);
```

Login UI Custom Theme

before



after



Add GraphQL API Using Amplify

- **Queries**
  - Get list of models
  - Get list of devices
- **Mutations**
  - Create a new model
  - Create a new device
- **Subscriptions**
  - Get real-time notifications when a device is updated

Steps to Add an API

1. Create GraphQL API and Database
2. Edit GraphQL Schema
3. Deploy the API
4. Connect Frontend to API

Create GraphQL API and Database

1

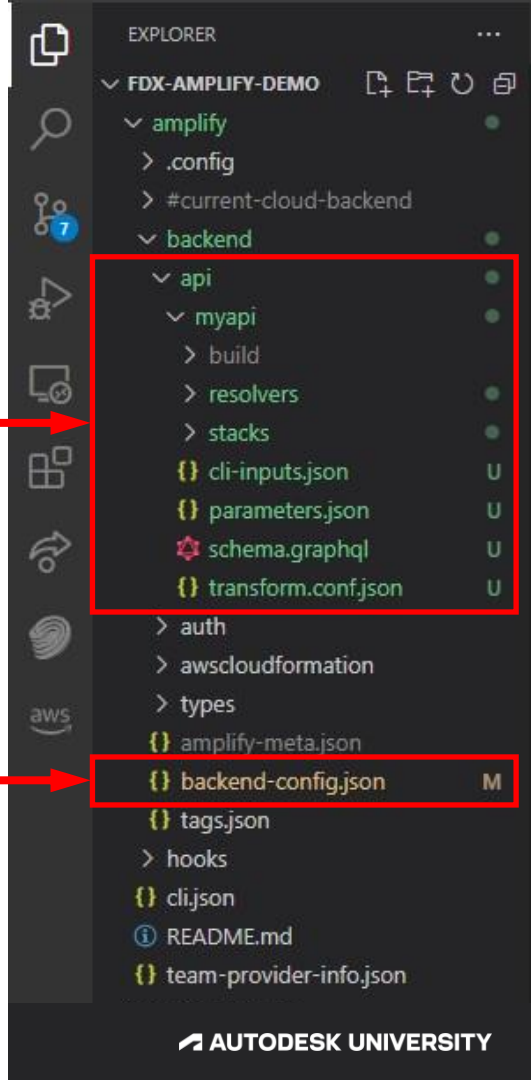
```
amplify add api
```

```
? Select from one of the below mentioned services:
# GraphQL
? Provide API name:
# myapi
? Choose the default authorization type for the API:
# API Key
? Enter a description for the API key:
# demo
? After how many days from now the API key should expire:
# 7 (or your preferred expiration)
? Do you want to configure advanced settings for the GraphQL
API:
# No
? Do you have an annotated GraphQL schema?
# No
? Choose a schema template:
# One-to-many relationship (e.g., “Blogs” with “Posts” and
“Comments”)
? Do you want to edit the schema now?
# Yes
```

Amplify Generated Files

Created backend api folder with `schema.graphql`

Added new api settings to `backend-config.json`



Edit GraphQL Schema

2

schema.graphql (before)

```
input AMPLIFY { globalAuthRule: AuthRule = { allow: public } }

type Blog @model {
  id: ID!
  name: String!
  posts: [Post] @hasMany
}

type Post @model {
  id: ID!
  title: String!
  blog: Blog @belongsTo
  comments: [Comment] @hasMany
}

type Comment @model {
  id: ID!
  post: Post @belongsTo
  content: String!
}
```

schema.graphql (after)

```
type Model @model @auth(rules: [{ allow: owner }]) {
  id: ID!
  name: String!
  accProjectId: String!
  accEntityId: String!
  urn: String
  devices: [Device] @hasMany
}

type Device @model @auth(rules: [{ allow: owner }]) {
  id: ID!
  name: String!
  state: String @default(value: "On")
  dbId: Int!
  point: AWSJSON!
  model: Model @belongsTo
  modelDevicesId: ID!
}

type Subscription {
  onDeviceUpdateByModelId(modelDevicesId: ID!): Device
    @aws_subscribe(mutations: ["updateDevice"])
}
```

Deploy the API

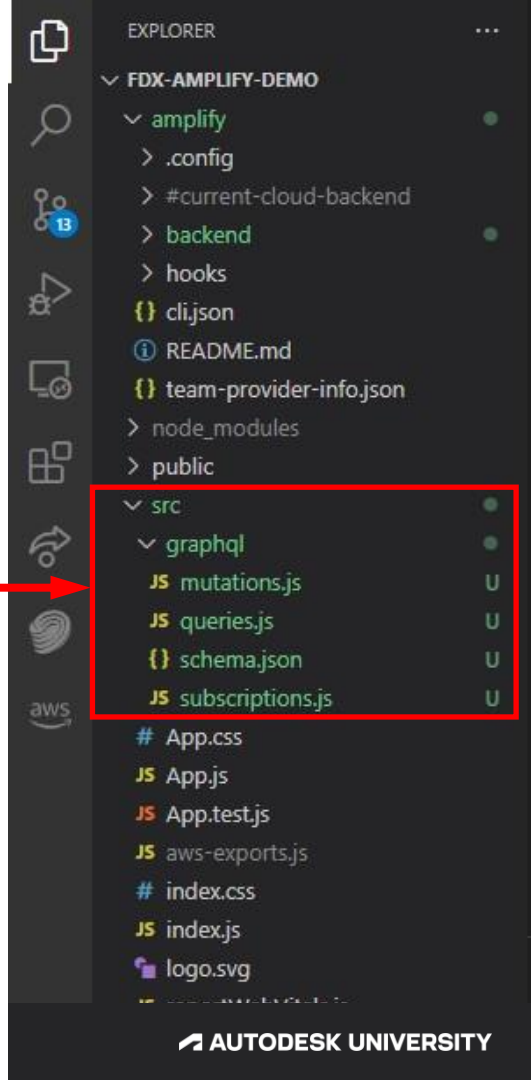
3

```
amplify push
```

```
? Are you sure you want to continue? Y

# You will be walked through the following questions for GraphQL
code generation
? Do you want to generate code for your newly created GraphQL
API? Y
? Choose the code generation language target: javascript
? Enter the file name pattern of graphql queries, mutations and
subscriptions: src/graphql/**/*.js
? Do you want to generate/update all possible GraphQL operations
- queries, mutations and subscriptions? Y
? Enter maximum statement depth [increase from default if your
schema is deeply nested]: 2
```

Amplify Generated Files

Generated code for mutations, queries, and subscriptions



A Look Inside the Generated Code

```
...
export const listModels = /* GraphQL */ `
  query ListModels(
    $filter: ModelModelFilterInput
    $limit: Int
    $nextToken: String
  ) {
    listModels(filter: $filter, limit:
$limit, nextToken: $nextToken) {
      items {
        id
        name
        accProjectId
        accEntityId
        urn
        devices {
          nextToken
        }
        createdAt
        updatedAt
        owner
      }
      nextToken
    }
  }
`;
...
```

Queries

```
...
export const createModel = /* GraphQL */ `
  mutation CreateModel(
    $input: CreateModelInput!
    $condition: ModelModelConditionInput
  ) {
    createModel(input: $input, condition:
$condition) {
      id
      name
      accProjectId
      accEntityId
      urn
      devices {
        items {
          id
          name
          state
          dbId
          point
          modelDevicesId
          createdAt
          updatedAt
          owner
        }
        nextToken
      }
      createdAt
      updatedAt
      owner
    }
  }
`;
...
```

Mutations

```
...
export const onDeviceUpdateByModelId = /*
GraphQL */ `
  subscription
OnDeviceUpdateByModelId($modelDevicesId:
ID!) {
    onDeviceUpdateByModelId(modelDevicesId:
$modelDevicesId) {
      id
      name
      state
      dbId
      point
      model {
        id
        name
        accProjectId
        accEntityId
        urn
        devices {
          nextToken
        }
        createdAt
        updatedAt
        owner
      }
      modelDevicesId
      createdAt
      updatedAt
      owner
    }
  }
`;}
`;
...
```

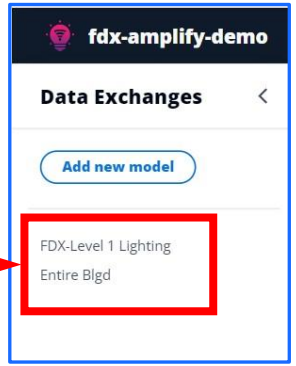
Subscriptions

Connect Frontend to API - Queries

4

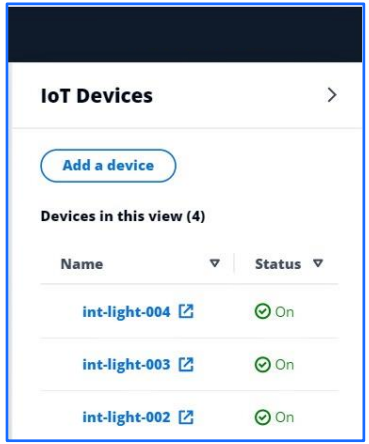
Get a list of models when the user signs in

```
import { API, graphqlOperation } from "aws-amplify";
import { listModels } from "../graphql/queries";

const modelsData = await API.graphql(graphqlOperation(listModels));
```



Get a list of devices by model ID after the user selects a model

```
import { API, graphqlOperation } from "aws-amplify";
import { listDevices } from "../graphql/queries";

const devicesData = await API.graphql(
      graphqlOperation(listDevices, {filter: { modelDevicesId: { eq: modelId } } })
    );
```

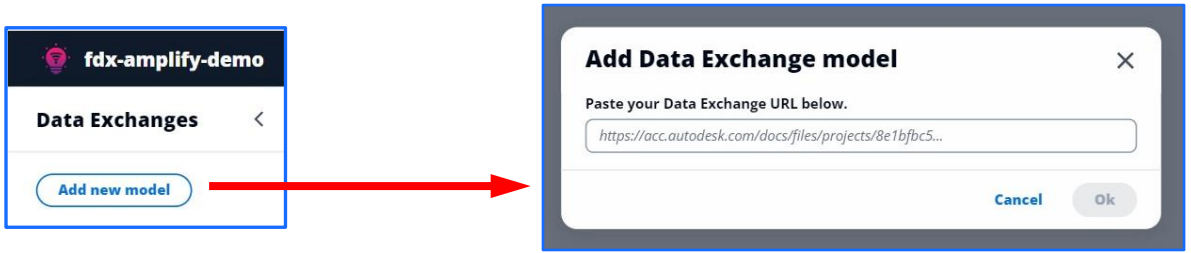


Connect Frontend to API - Mutations

Create new model with only 3 lines of code

4

```
import { API, graphqlOperation } from "aws-amplify";
import { createModel } from "../graphql/mutations";

const addModel = await API.graphql( graphqlOperation(createModel, { input: newModel }));
```



Connect Frontend to API - Mutations

4

Create new device with only 3 lines of code

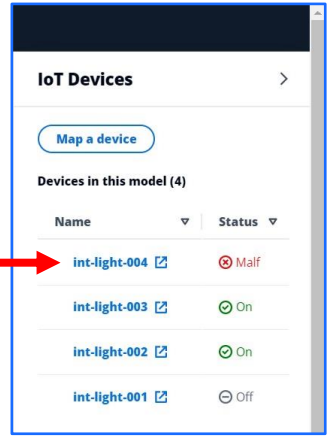
```
import { API, graphqlOperation } from "aws-amplify";
import { createDevice } from "../graphql/mutations";

const addDevice = await API.graphql(graphqlOperation(createDevice, {input: device }));
```

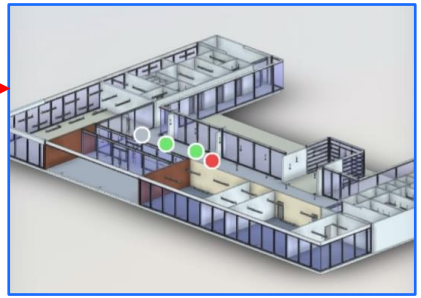
Connect Frontend to API - Subscriptions

4

Subscribe to update of Devices by ModelID

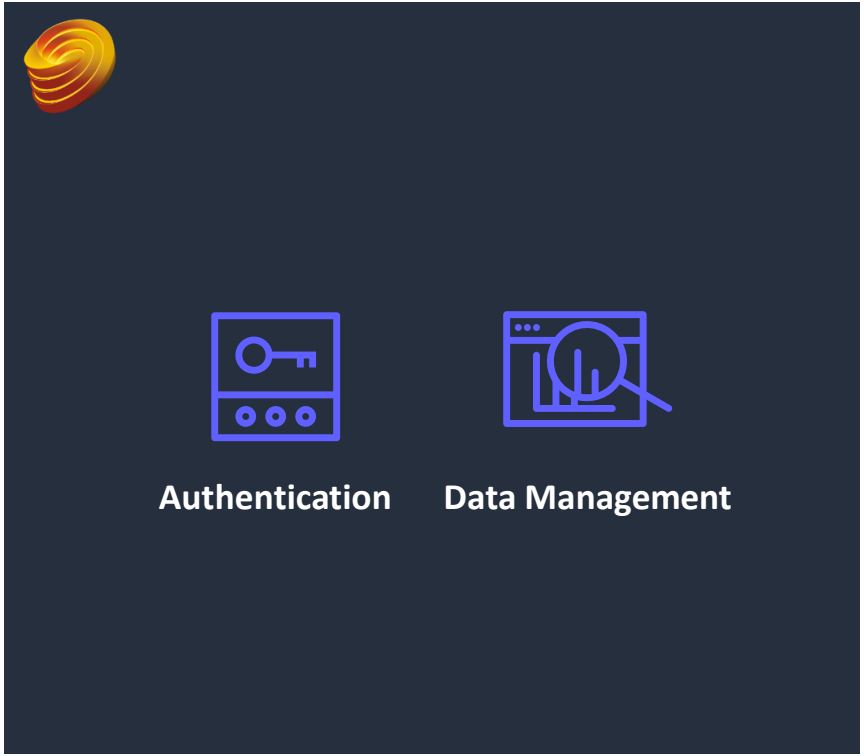
```
import { API } from "aws-amplify";
import { onDeviceUpdateByModelId } from "../graphql/subscriptions";

const subscription = modelId => {
  subscriptionOnDeviceUpdate = API.graphql({
    query: onDeviceUpdateByModelId,
    variables: {
      modelDevicesId: modelId,
    },
  }).subscribe({
    next: ({ provider, value }) => console.log({ provider, value }),
    error: (error) => console.warn(error)
  });
};
```





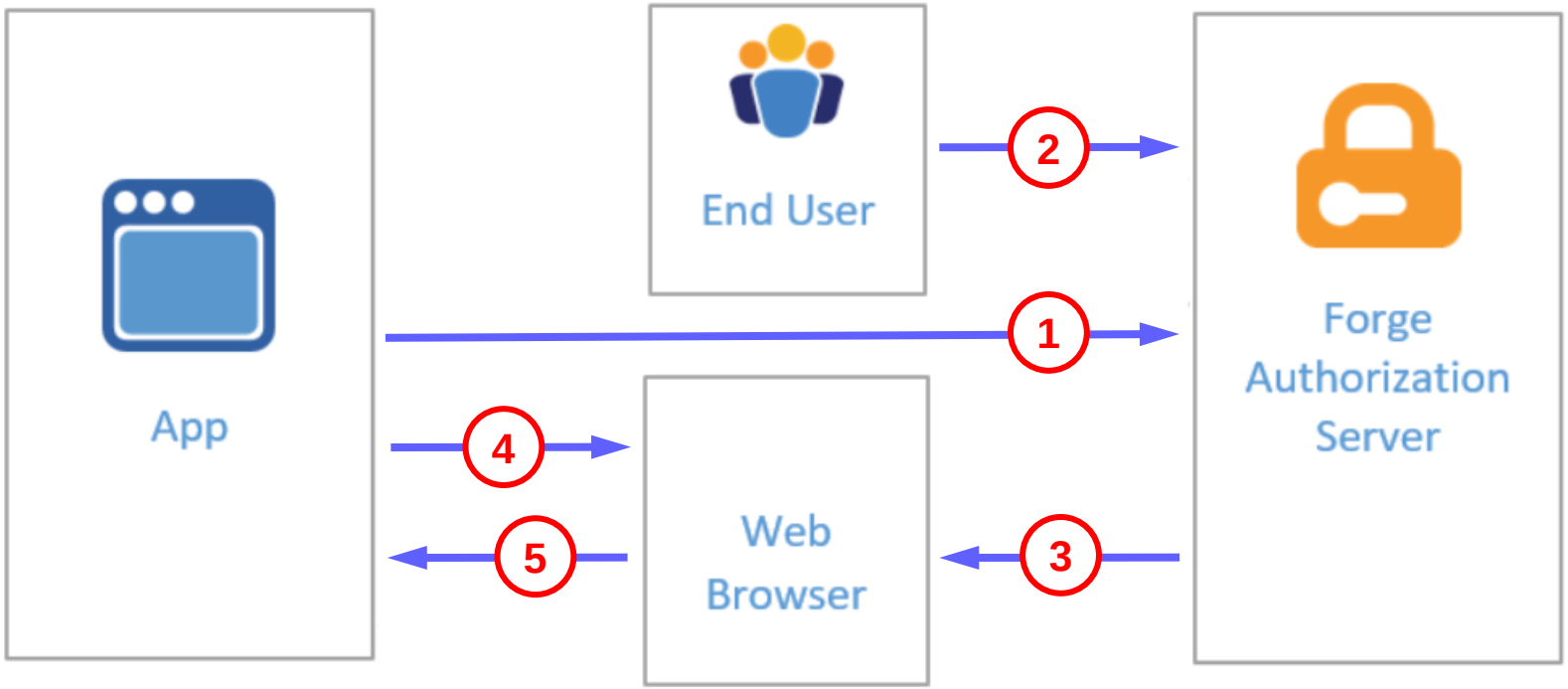
Add Forge API

- **Authentication API**
  - Get 3-legged token with implicit grant
- **Data Management API**
  - Get latest Data Exchange model version
  - Get Data Exchange model name



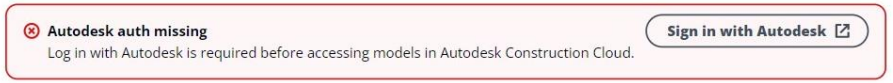
Get 3-Legged Token

https://forge.autodesk.com/en/docs/oauth/v2/tutorials/get-3-legged-token-implicit/



Get 3-Legged Token

https://forge.autodesk.com/en/docs/oauth/v2/tutorials/get-3-legged-token-implicit/



1. Redirect to the Authorization Web Flow

```
<a href="https://developer.api.autodesk.com/authentication/v1/authorize?
response_type=token&client_id=obQDn8P0GanGFQha4ngKKVWcxwyvFAGE&redirect_uri=http%3A%2F%2Fsampleapp.c
om%2Foauth2%2Fcallback&scope=data:read">Sign in with Autodesk</a>
```

2. Extract the Access Token from callback URL

```
http://sampleapp.com/oauth/callback#access_token=eyJhbGciOiJIUzI1NiIsImtpZCI6Imp3dF9zeW1tZXRyaWNfa2V
5In0.eyJ1c2VyaWQiOiJIU0pHTTdUSFJIVUIiLCJleHAiOjE1MDE4OTI5MjgsInNjb3BlIjpbXSwiY2xpZW50X2lkIjoibk9oYnJ
PZW9HeHE4R2JBeFVvR2NIcXpEWmVqQWxxSUsiLCJhdWQiOiJodHRwczovL2F1dG9kZXNrLmNvbS9hdWQvand0ZXhwMTQ0MCIsImp
0aSI6Im12bmwyZ2tKOEU4Tkd2S2JEVk00S3BHaTRCYkZtRndyUmVrd2NjT3B3RU1OTlVTdnZrNnljNllWSGo3d29WWjMifQ.Niy8
dwBQVuhcaCTClZqttJleuKIoQtnS8yoT1ZJWgNg&token_type=Bearer&expires_in=86399
```

Get Data Exchange Info

forge.autodesk.com/en/docs/data/v2/reference/http/projects-project_id-items-item_id-tip-GET/

ACC URL:
https://acc.autodesk.com/docs/files/projects/8e1bfbc5-7807-4190-90ba-429cb434cd9c?folderUrn=urn%3Aadsk.wipprod%3Afs.folder%3Aco.cFlLi3uETZWniC1FJUibvg&entityId=urn%3Aadsk.wipprod%3Adm.lineage%3AZE5IZ5jYTuevlNQMmJ5igw&viewModel=detail&moduleId=folders

Get the latest ("tip") version of a given item (Data Exchange)

```
const axios = require("axios").default;

const getItemTipVersion = async (accessToken, projectId, itemId) => {
  const url = `https://developer.api.autodesk.com/data/v1/projects/${projectId}/items/${encodeURIComponent(itemId)}/tip`;
  const res = await axios.get(url, {
    headers: {
      "Content-Type": "application/json",
      Authorization: "Bearer " + accessToken,
    },
  });
  return res?.data?.data;
};
```

Get Data Exchange Info – Sample Response

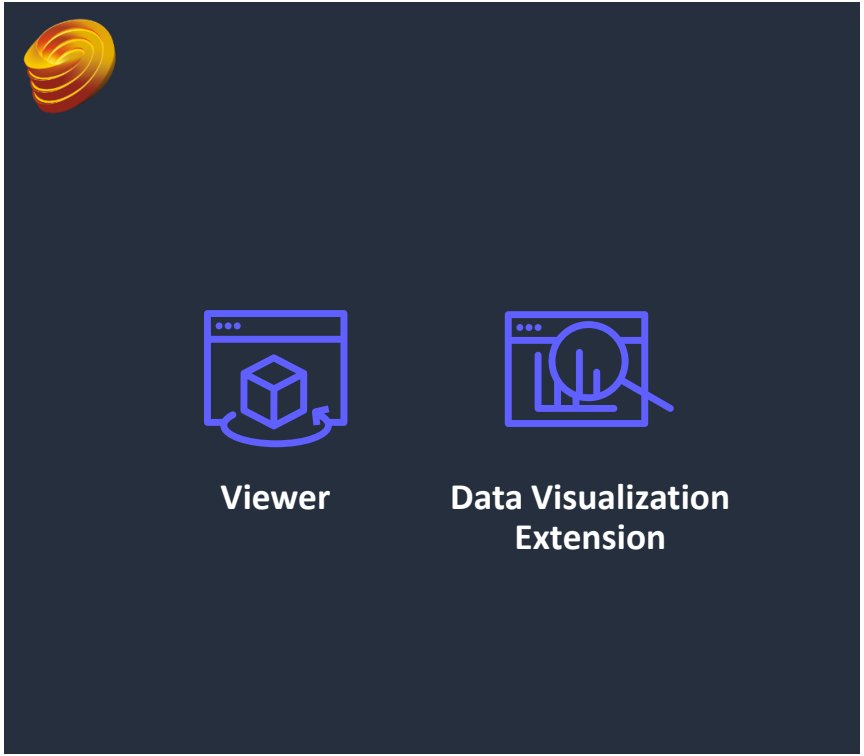
```
const sampleResponse_GetItemTipVersion = {
  data: {
    type: "versions",
    id: "urn:adsk.wipprod:fs.file:vf.FXwwyfYbSCWx1Ol7cpgk9w?version=1",
    attributes: {
      name: "FDX-Level 1 Lighting",
      displayName: "FDX-Level 1 Lighting",
      createTime: "2022-08-06T04:16:07.0000000Z",
      createUserName: "Enrico Chionna",
      ...
    },
    ...
    relationships: {
      ...
      derivatives: {
        data: {
          type: "derivatives",
          id: "dXJuOmFkc2sud2lwcHJvZDpmcy5maWxlOnZmLkZYd3d5ZlliU0NXeDFPbDdjcGdrOXc_dmVyc2lvbj0x",
        },
        ...
      },
      ...
    },
  },
};
```

Data Exchange Name

Data Exchange URN

Add Forge Viewer & Sprites

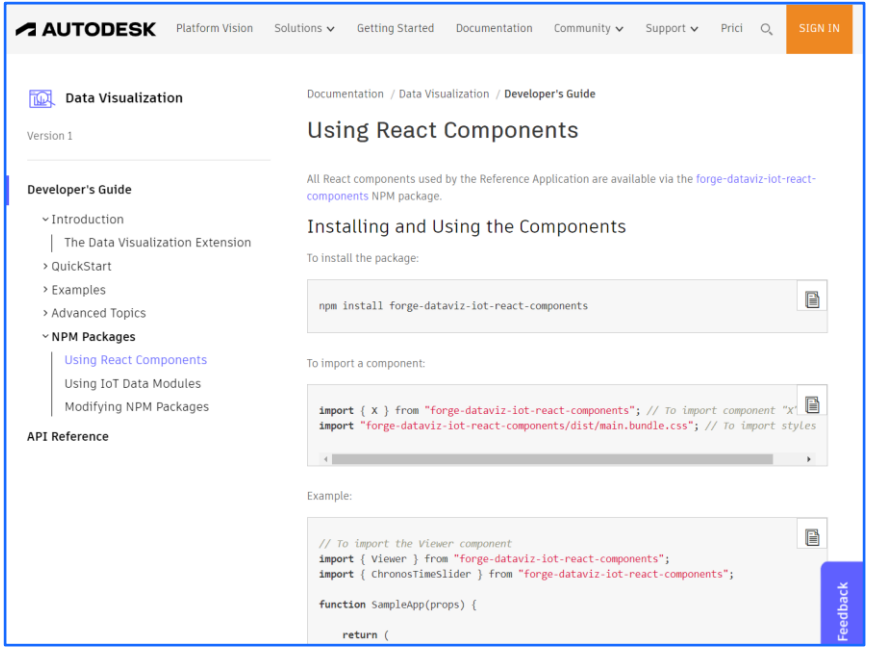
- **Viewer**
  - Load models in web viewer
- **Data Visualization Extension**
  - Add sprites representing IoT devices



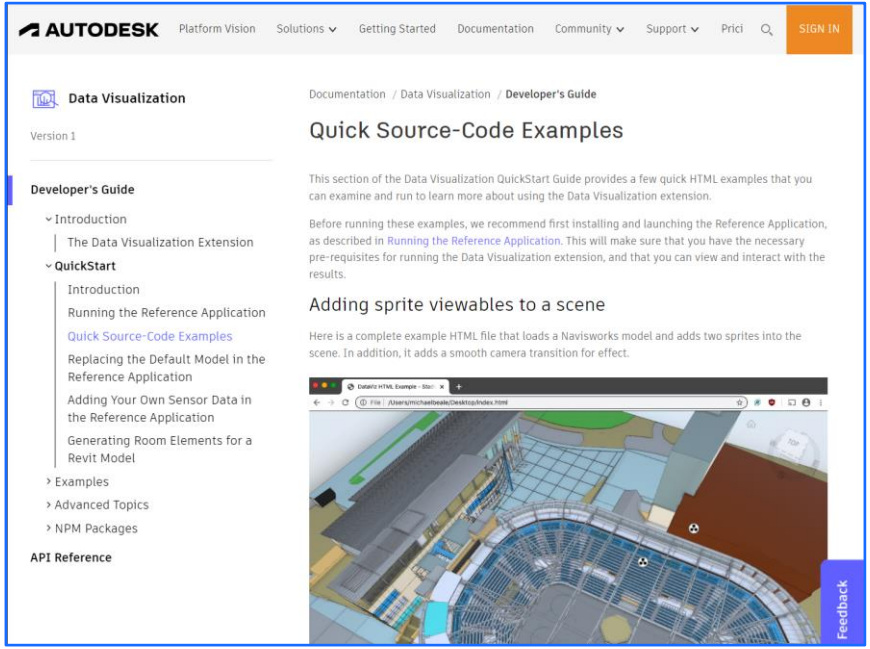
Add Forge Viewer & Sprites

https://forge.autodesk.com/en/docs/dataviz/v1/developers_guide/introduction/

Viewer



Data Visualization Extension (Sprites)

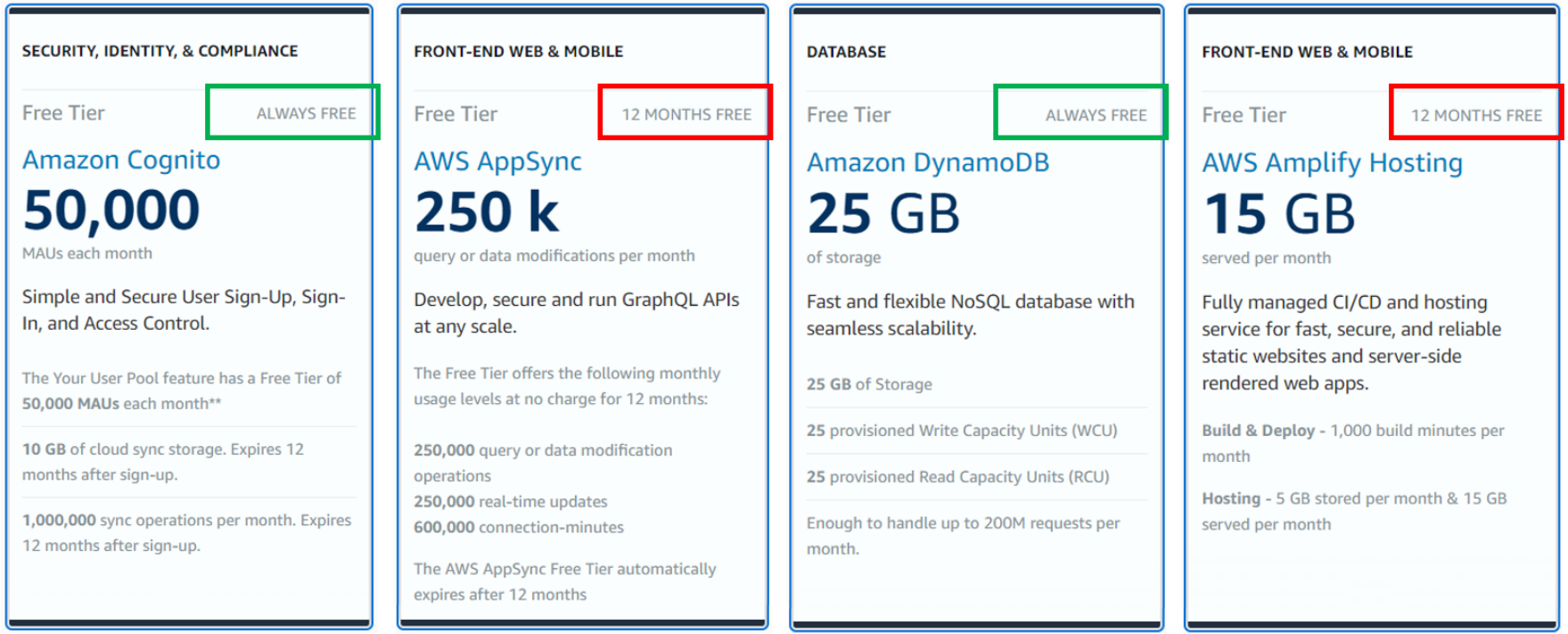


Costs - Autodesk

Revit		ACC / Docs		Forge	
Architect:	PAID	**Architect:**	PAID	**Architect:**	FREE
Vendor:	FREE	**Vendor:**	FREE	**Vendor:**	FREE

Costs – AWS

https://aws.amazon.com/free/



SECURITY, IDENTITY, & COMPLIANCE

Free Tier — ALWAYS FREE

Amazon Cognito

50,000

MAUs each month

Simple and Secure User Sign-Up, Sign-In, and Access Control.

The Your User Pool feature has a Free Tier of **50,000 MAUs** each month**

10 GB of cloud sync storage. Expires 12 months after sign-up.

1,000,000 sync operations per month. Expires 12 months after sign-up.

FRONT-END WEB & MOBILE

Free Tier — 12 MONTHS FREE

AWS AppSync

250 k

query or data modifications per month

Develop, secure and run GraphQL APIs at any scale.

The Free Tier offers the following monthly usage levels at no charge for 12 months:

250,000 query or data modification operations
250,000 real-time updates
600,000 connection-minutes

The AWS AppSync Free Tier automatically expires after 12 months

DATABASE

Free Tier — ALWAYS FREE

Amazon DynamoDB

25 GB

of storage

Fast and flexible NoSQL database with seamless scalability.

25 GB of Storage

25 provisioned Write Capacity Units (WCU)

25 provisioned Read Capacity Units (RCU)

Enough to handle up to 200M requests per month.

FRONT-END WEB & MOBILE

Free Tier — 12 MONTHS FREE

AWS Amplify Hosting

15 GB

served per month

Fully managed CI/CD and hosting service for fast, secure, and reliable static websites and server-side rendered web apps.

Build & Deploy - 1,000 build minutes per month

Hosting - 5 GB stored per month & 15 GB served per month

Resources and References

- **Autodesk Forge**:
  - https://forge.autodesk.com
- **Autodesk Data Exchange**:
  - https://forge.autodesk.com/blog/data-exchange-released-forge-data-exchange-apis-now-available-public-beta
  - https://forge.autodesk.com/en/docs/fdx/v1/developers_guide/fd_overview
  - https://forge.autodesk.com/en/docs/data/v2/reference/http/projects-project_id-items-item_id-tip-GET
- **Autodesk Authentication**:
  - https://forge.autodesk.com/en/docs/oauth/v2/tutorials/get-3-legged-token-implicit
- **Autodesk Forge Viewer**:
  - https://forge.autodesk.com/en/docs/viewer/v7/developers_guide/overview/
  - https://forge.autodesk.com/en/docs/dataviz/v1/developers_guide/introduction
- **Amplify Framework Documentation**:
  - https://ui.docs.amplify.aws
- **Amplify UI**:
  - https://ui.docs.amplify.aws
- **Figma**:
  - https://www.figma.com
- **React**:
  - https://reactjs.org
- **GraphQL**:
  - https://graphql.org
  - https://aws.amazon.com/graphql
  - https://aws.amazon.com/graphql/graphql-dynamodb-data-modeling
- **Cloudscape**:
  - https://cloudscape.design
- **AWS Free Tier**:
  - https://aws.amazon.com/free