Add Flow to Web Application

Table of Contents

1 Objective

The objective of this exercise is to take your flow and integrate into a web application. This will allow the application to invoke the flow and provide the CSS (styling and look and feel) for it along with potentially integrating other flows to provide a much richer user experience. The application you will use is simple in that the focus is on how to leverage the DaVinci widget in your application with minimal code changes.

In order to use this application you will need to have a Glitch (https://glitch.com/) free or paid account to host the application. It is easy to signup for Glitch all you need is an email address and you can even use a Magic link to login so that you don't need to provide more details other than your email or define yet another password. You will be creating your own version of the application by using the Glitch Remix option, more on that later.

> **Tip**
>
> All the number instruction steps in each section must be followed to successfully complete this exercise. Any bullet points or images are for example only unless called out in a numbered step.

2 Do the following task items

The basic outline of this exercise is to first define the application in DaVinci. Once this is completed you will then create your own version of the application by doing a *remix* and customizing this to point to your DaVinci application. Then finally will come the testing of this integration.

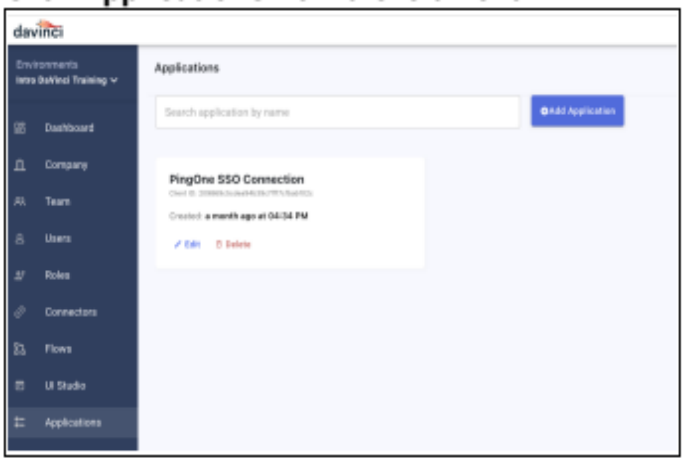
2.1 Configure application in DaVinci

Flows can be invoked by external entities such as web applications or native applications or even APIs by making a call to DaVinci. This call is made to an application that defines the credentials

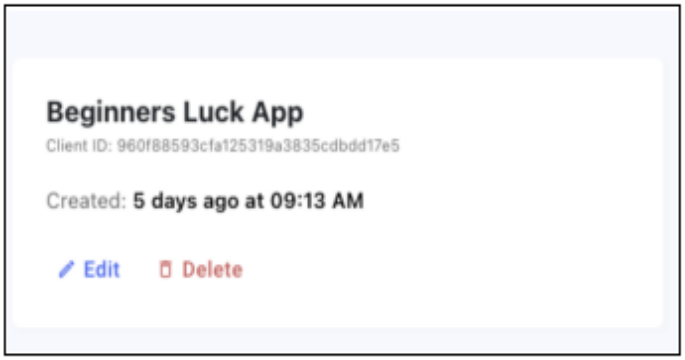
needed by the application and what flow to be invoked. The application can also be used to define a policy for A/B testing that lets you try two different flows for the same base functionality and gather information on which one is the most effective.

In this section you will define the application that will be used by your web application to invoke your registration flow. You will be copying certain information that is needed to configure the web application to a temporary text editor to facilitate copy and paste actions.

1 If you have not done so you can exist out of your flow, quickest way to do this is to use the left menu; of course save any changes first.
2 Click **Applications** from the left menu.



   - You will have a default application that is the integration point between PingOne platform and DaVinci. This application is used to log you into the administrative console for DaVinci.
3 Click **Add Application** button to create a new application.
4 For the application **Name** type
   - **Beginners Luck App**
5 Click **Create** button.



6 Click on the **Beginners Luck App** application or the **Edit** link on the application to gather the information you need and finish configuring it.
7 **Open** a text editor with an empty page, this will be your buffer to save the temporary information you need to configure the web application in the next section.
8 In your browser tab with your DaVinci application open:
   8.1 Highligh the **Company ID** value in the **General** tab of the application.
      - Make sure that you highlight all of it.
   8.2 **Copy** the **Company ID** and **paste** into your text editor.
   8.3 In the **API Key** field use the **eye** icon to toggle the key visible.
   8.4 Highligh the **API Key** value in the **General** tab of the application.
      - Make sure that you highlight all of it, even what is not shown on the screen.
   8.5 **Copy** the **API Key** and **paste** into your text editor.
9 Click **Flow Policy** tab in your application.

10 Click **+ Add flow Policy** link to define a flow policy.

11 For the flow policy **Name** type
   - **Beginners Luck Registration Policy**
12 The **All Flow Policies** radio button should be selected, this will not be a PingOne Flow Policy.
13 Click **Progressive Registration** flow from the list.
14 Click **Latest Version** checkbox.

   - You want to always use the latest version for this training, but you could select a specific version for a production application. And then continue to edit the flow to enhance it. And when ready to roll that into production you could then change the policy.
   - The flow will move to the Flows Added list below the name, this allows you to have more than one flow used in a policy, for example if you wanted to do A/B testing.
   - More on A/B testing in a later exercise.
15 Click **Create Flow Policy** button in the bottom right of the dialog.
16 Click **Save Flow Policy** button, you are not going to setup any weight distribution as there is only one flow.
17 Highligh the **Policy ID** value in the flow policy you defined.
   - Make sure that you highlight all of it.
18 **Copy** the **Policy ID** and **paste** into your text editor.
19 You can close the application, by clicking **Applications** in the left menu.
20 Continue with the next section.

2.2 Remix the web application and configure it

The next step is to define your web application, as mentioned at the start of this exercise Glitch is being used to host the application. This simplifies the setup from a lab perspective, but technically you can use any web application. Details for integrating your application to a DaVinci flow can be found in the developer documentation.

You will be using the widget method but not to worry all the code is done, you just need to paste in the configuration from the previous section.

Important

After you created your application using the steps in this section please review the comment in your code about the endpoints depending on the region your tenant is running on.

You will find this comment in the files:

- `views/index.html`

Please follow the instructions in the comment, repeated below for reference. You will also find the details in the documentation.

```
IMPORTANT API ENDPOINT CONFIGURATION

Depending on what region your PingOne tenant is located you may need to
adjust the endpoints defined in this code to use the correct region endpoints.

The code on this page is by default using North American endpoints, if that is
your region then no changes are needed. Otherwise, look for the following
comment tag as to where you need to change the endpoints:

    TENANT ENDPOINT

Then change them as follows:
  APAC
    api.pingone.com is api.pingone.asia
    auth.pingone.com is auth.pingone.asia
    orchestrate-api.pingone.com is orchestrate-api.pingone.asia
    assets.pingone.com is assets.pingone.asia
  EU
    api.pingone.com is api.pingone.eu
    auth.pingone.com is auth.pingone.eu
    orchestrate-api.pingone.com is orchestrate-api.pingone.eu
    assets.pingone.com is assets.pingone.eu
  Canada
    api.pingone.com is api.pingone.ca
    auth.pingone.com is auth.pingone.ca
    orchestrate-api.pingone.com is orchestrate-api.pingone.ca
    assets.pingone.com is assets.pingone.ca
```

Note

You don't want to expose the API key and other system information in your HTML or JavaScript files that are loaded into the browser. In a quick demonstration you may do this to simplify your code. This application has been setup with server-side code to handle the API calls and use environment variables at run time. This will not expose the actual API key in your browser. This makes the application code a bit more complex, but no worries everything is provided for you in the exercises. Your welcome to review the code and even ask the instructor about it, but the focus of course is DaVinci so follow lab instructions in modifying the application carefully.

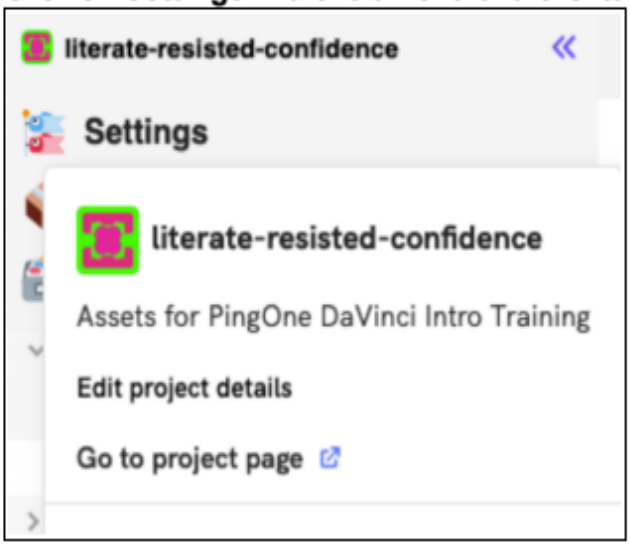
Tip

Going forward in future exercises you will be using Glitch a lot, especially editing the code and testing your flow. Before you start a lab or need to access Glitch, ensure that you are logged into your Glitch account and accessing the current project.

And follow the instructions in this section to rename your project to something more meaningful than the default name assigned to your project.

1 In your browser **login in** to Glitch with your account.
2 Open a new tab and **paste** the following URL to open the project.
   - https://glitch.com/edit/#!/ping-davinci-training-lab-v2?path=server.js%3A1%3A0

3 Click the **Remix to Edit** button in the top right of the page.
   - This will generate your own copy of the application, this is what you will edit going forward in this exercise.
   - It may take a bit for it to complete, therefore be patient.
4 Your application was given a random name when it was created by glitch, lets set this to a more meaningful name.
   4.1 Click on **Settings** in the left menu of the Glitch editor.



4.2 Click **Edit project details** menu.

4.3 Change the project name starting with **your initials** and appending **davinci-intro-training-lab06** for example:

- The name does need to be unique because it will become part of the url for your application.
- If the domain already exists, somebody has the same initials as you, just append a number to it to make it unique.

4.4 Click **Save** button.

5 Don't forget about the **important** note at the start of this section. This is a good time to apply those changes as you go through the following steps to update the configuration of your application. Read carefully through everything in the following steps.

6 Click **.env** file in your Glitch project.

- This contains the environment variables used in the code of this application.
- Glitch will protect this information by only making it visible to owners of the project or those

you gave access to the project.
- When the application is remixed the values are not copied from the original project.

7 You will set some of the values with what you saved in the earlier section and add URLs for the DaVinci endpoints.

7.1 For the URLs if you are not in North America then use the above important note to get the URLs for your region.
- The file in `views/index.html` does contain a URL for the DaVinci assets, you will also want to adjust this one; if you are not in North America.

8 In the **.env** update the variables as shown below:

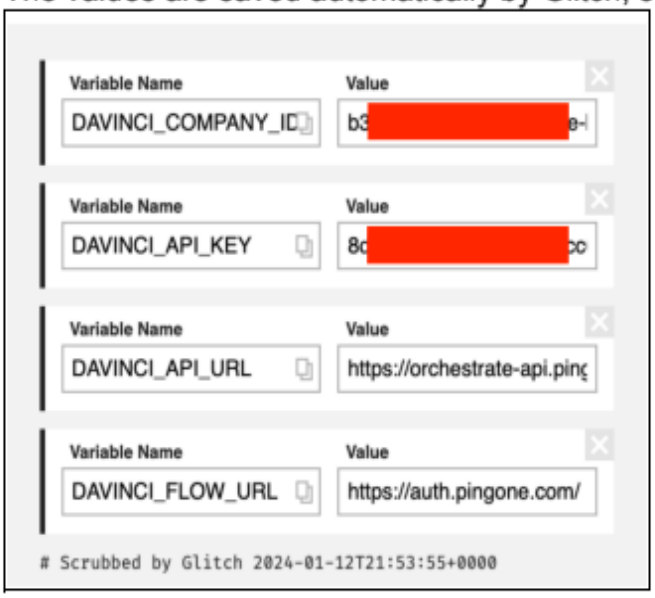
8.1 Variable **DAVINCI_COMPANY_ID** from your editor **copy/paste** the **company ID**.

8.2 Variable **DAVINCI_API_KEY** from your editor **copy/paste** the **API key**.

8.3 Variable **DAVINCI_API_URL** you will **paste** the URL:
- **https://orchestrate-api.pingone.com/**
- Reminder this is for the North America region, if not in that region the the important note above.

8.4 Variable **DAVINCI_FLOW_URL** you will **paste** the URL:
- **https://auth.pingone.com/**
- Reminder this is for the North America region, if not in that region the the important note above.

8.5 The values are saved automatically by Glitch, so no need for a save button.



9 Open the **views** folder in your project.
- This contains the html pages that will be rendered in your application.
- The public folder contains the files that are publicly exposed by your application.

10 Select the **index.html** find the code `const policyId = "";` this will be at the top of the files in the second script tag.

10.1 From your editor **copy** the **Policy ID** value and **paste** in the **quotes** in the code block

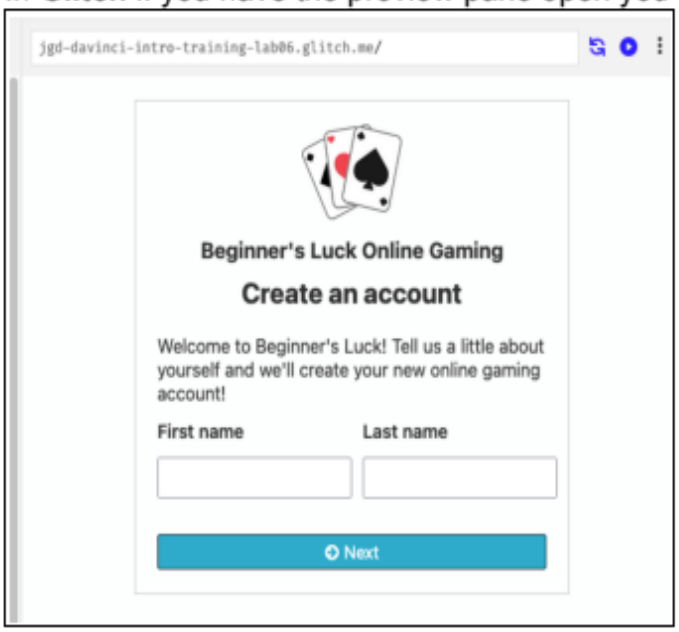
- You will note in the code an expression in **<% %>** for the *companyID* and *flowURL*

constant values.
- These are variable that will be replaced when the template is rendered.
- A little bit further down you will see the code as shown in the screenshot below:

```
/*** Retrieve DaVinci Token ***/
fetch("/fetchDaVinciToken", requestOptions)
  .then((response) => response.json())
  .then((responseData) => {
    // setup properties to call widget to render flow
    var props = {
      config: {
```

  - This is calling a function in the *server.js* file in your application.
  - This is the server side code that handles getting the code, more on this later.

11 In **Glitch** if you have the preview pane open you will see your flow rendered.



- This shows that you are successfully connected to your application and the flow is being invoked.
- You will more fully test this later in this exercise.

12 Continue with the next section.

2.2.1 Let's review code in application

In this section you will review various code blocks in the application you just created in the above section. This will provide a brief overview of the integration being used in the application for DaVinci. If you need more detail you should review Integrating Flows into Applications documentation. The goal is mainly so you know where the various parts of the code are and how they come together in this application. There is code in this application that will not come into play until later exercises in this training.

1 In the **index.html** file note the **loadwidget** function.
  1.1 Loaded during page load in your browser.

```
  <body onload="loadwidget()">
    <div id="widget" class="skWidget widget">
      If you're seeing this message, the DaVinci flow hasn't loaded yet.
    </div>
  </body>
</html>
```

1.2 The start of the function.

```
function loadwidget() {
  var requestOptions = {
    method: "GET",
  };
```

2 This function is what invokes the DaVinci widget to render the flow you created by using the policy you specified.

2.1 After using the **fetch** function call to retrieve a DaVinci token, by invoking an endpoint at the server side of your application.

```
/*** Retrieve DaVinci Token ***/
fetch("/fetchDaVinciToken", requestOptions)
  .then((response) => response.json())
  .then((responseData) => {
    // setup properties to call widget to render flow
    var props = {
      config: {
```

2.2 It then uses the DaVinci widget to render your flow

```
    // console.log(responseData.access_token);

    console.log(props);
    davinci.skRenderScreen(document.getElementById("widget"), props);
  })
  .catch((error) => console.log("error", error));
```

3 Open the **server.js** file in the Glitch editor.

- This is all server side code it does not execute in your browser.

3.1 Scroll down until you see **/fetchDaVinciToken** text. (Our search for it.)

3.2 This block of code is the endpoint that the loadwidget function invoked to get the DaVinci token:

```
// Fetch the DaVinci token used by the loadwidget process
app.get("/fetchDaVinciToken", function (request, response) {
  const apiURL =
    process.env.DAVINCI_API_URL +
    "v1/company/" +
    process.env.DAVINCI_COMPANY_ID +
    "/sdktoken";

  // const skGetTokenUrl = tokenURL + "/v1/company/" + companyId + "/sdktoken";

  console.log("Flow URL: " + apiURL);

  var requestOptions = {
    method: "GET",
    headers: {
      "X-SK-API-KEY": process.env.DAVINCI_API_KEY,
    },
    redirect: "follow",
  };

  //*** Call DaVinci API to Get Token flow to access token ***/
  fetch(apiURL, requestOptions)
    .then((response) => response.json())
    .then((result) => {
      console.log(result);
      response.send(result);
    })
    .catch((error) => console.log("error", error));

  //   response.send(resp);
});
```

3.2.1 You will see references to your variables, for example:

- process.env.DAVINCI_API_URL +
- "X-SK-API-KEY": process.env.DAVINCI_API_KEY,

3.3 Scroll up until you see **Starting point for application** text. (Our search for it.)

3.4 This code block is what renders the index.html replacing the variable expressions discussed in previous section.

```
// http://expressjs.com/en/starter/basic-routing.html
// Starting point for application
app.get("/", function (request, response) {
  var companyId = process.env.DAVINCI_COMPANY_ID;
  var flowURL = process.env.DAVINCI_FLOW_URL;
  console.log(companyId);

  response.render("index.html", {
    companyId: companyId,
    flowURL: flowURL,
  });
});
```

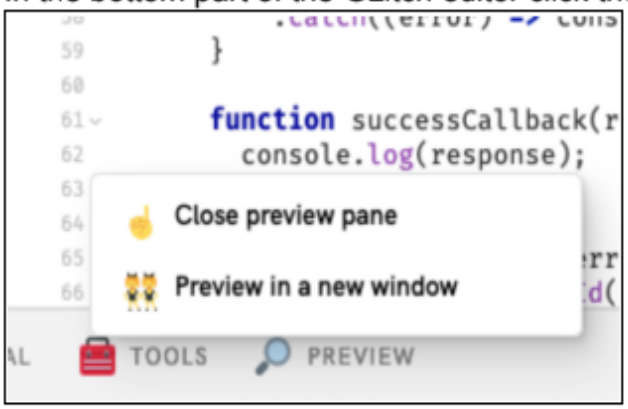
3.4.1 The key piece of information that is not exposed in your browser is of course the API key.

4 Continue with the next section.

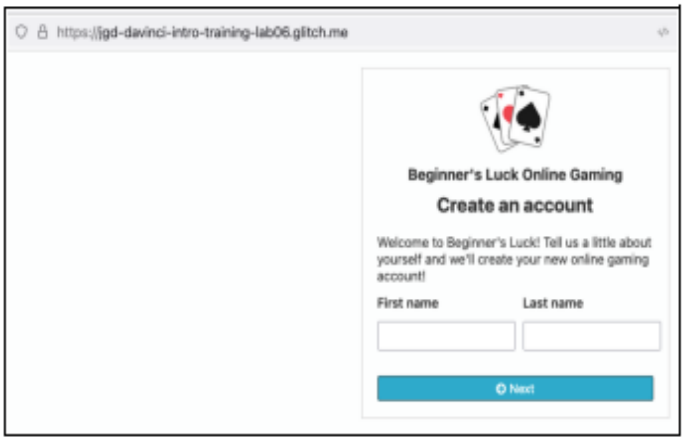
2.3 Test your web application integration to DaVinci

Lets now verify that the application will go through its paces executing the flow you created.

1 In the bottom part of the GLitch editor click the **Preview** button.



2 Click **Preview in a new window** this will launch the application in a new tab in your browser:



3 Enter the information through the various pages in the flow until you get to a successful result. Feel free to take the different paths such as not legal age and robot test.

4 Continue with the next section.

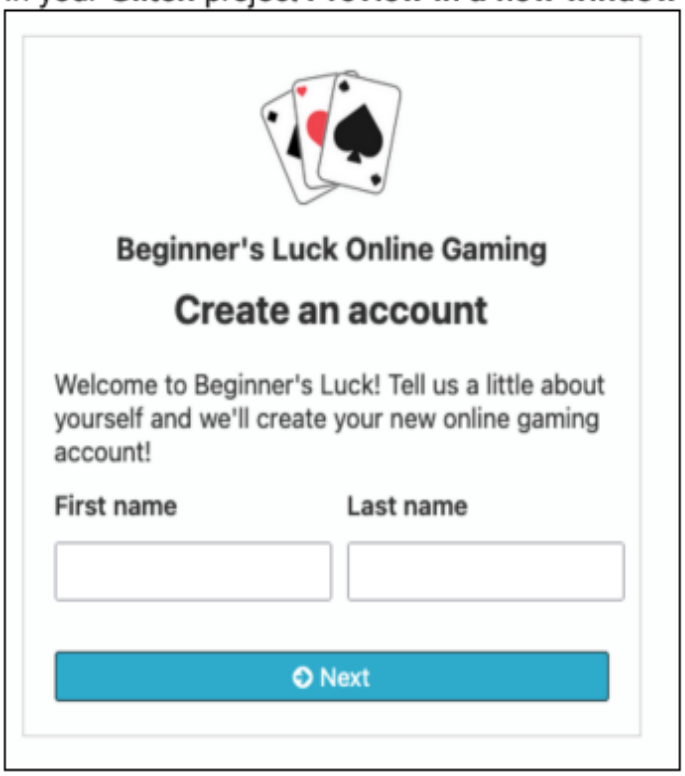
2.3.1 Turn off CSS reference in flow to use app CSS

At this time your flow is using the CSS you defined for it in its configuration. But in reality you want it to use the CSS presented by the application. It does happen to be the same CSS in this case but it is best practice to minimize the CSS in your flows and let the application handle the presentation.

1 In DaVinci click **Flows** in the left menu.
2 Click **Progressive Registration** flow to edit it.
3 Click **Flow Settings** for your flow.
4 Click **Customizations** tab.
5 Click **Use Custom CSS** toggle to turn it off.
6 Click **Save** button to save your changes then **Cancel** button to close the dialog.
7 Deploy and try your flow.

- No more custom CSS so it does not look as pretty.

8 In your **Glitch** project **Preview in a new window**



- Now only CSS from application is being used.
- Technically only CSS from application was being used, more on this in a later exercise.
- By turning it off in the flow it reminds you that the presentation of the UI will be controlled by an application that invokes the flow.

9 You have completed this exercise.