Watson Machine Learning + DevOps

Introduction

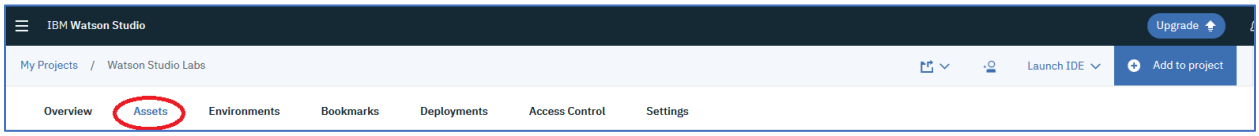
This lab introduces the Watson Machine Learning (WML) capability using the Female Human Trafficking datasets. It also covers the DevOps toolchain in the IBM Cloud. The lab consists of the following steps:

- Adding a data asset to the Watson Studio Labs project
- Creating a model to predict the trafficking risk for a traveler using the Model Builder interface. Saving the model to the model repository.
- Deploying and testing the model using the WML UI.
- Deploying a simple web front-end application and connecting it to the deployed model using an IBM Cloud toolchain.

Adding a Data Asset to the project

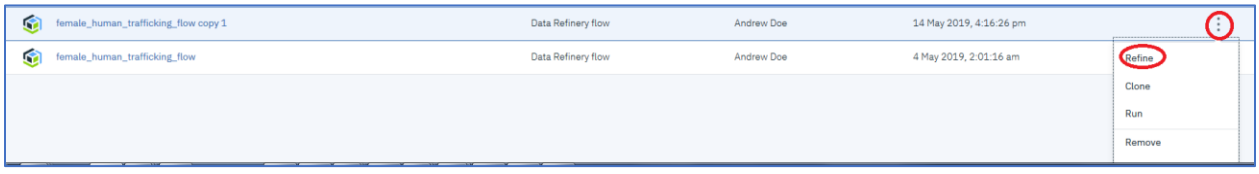
Our input into the model creation consists of the VETTING_LEVEL, AGE, Category, COUNTRIES_VISITED_COUNT, and the PASSPORT_COUNTRY fields. The VETTING_LEVEL is the target field. The others are input fields. We will modify the Data Refinery flow created in a previous lab to create this data asset and add it to the project.

1. Click on the **Assets** tab in the Project.



2. Click on the vertical ellipse on the right side of the **female_human_trafficking_flow** and click on **Clone**.

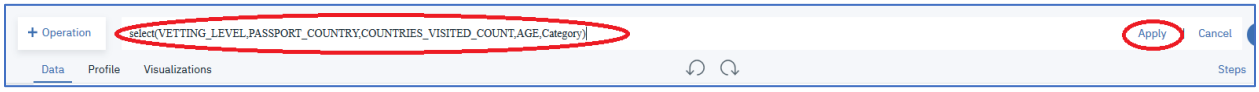
3. Click on the vertical ellipse on the right side of the **female_human_trafficking_flow copy 1** and click on **Refine**.



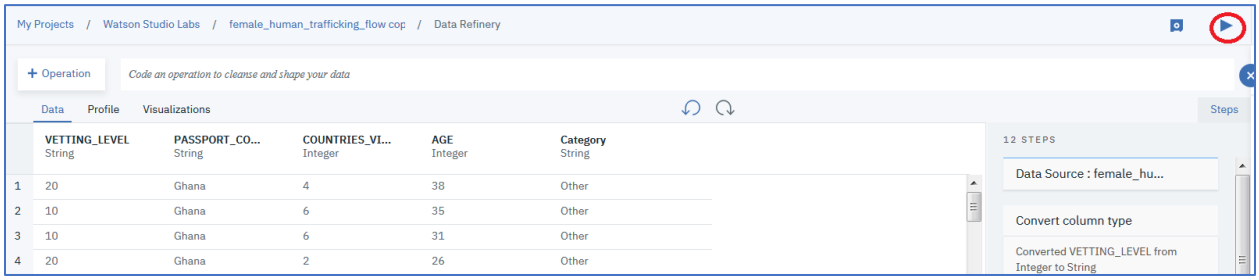
4. In the Operation entry field type the following:

   select(VETTING_LEVEL,
   PASSPORT_COUNTRY,COUNTRIES_VISITED_COUNT,AGE,Category)

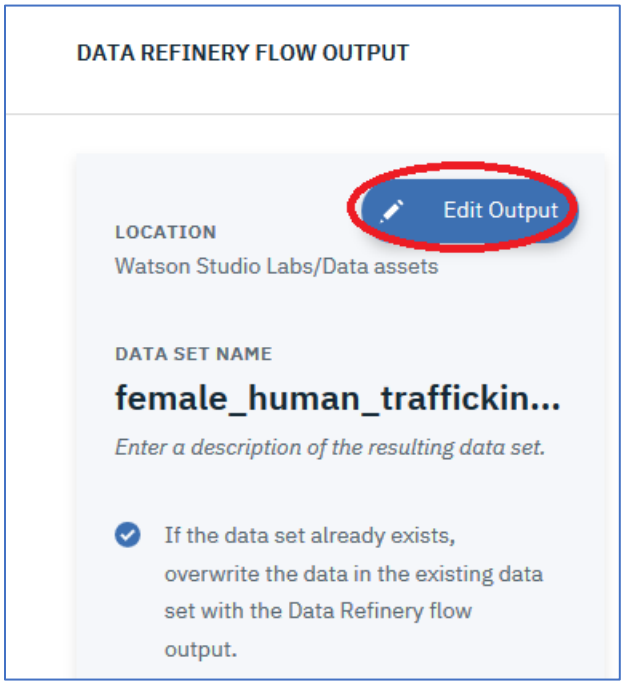
   then click **Apply**.



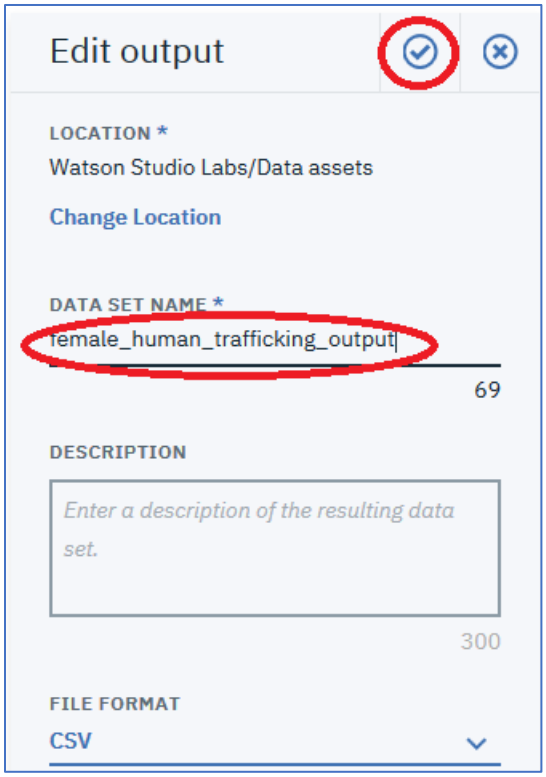
5. Click on the **Run** icon.



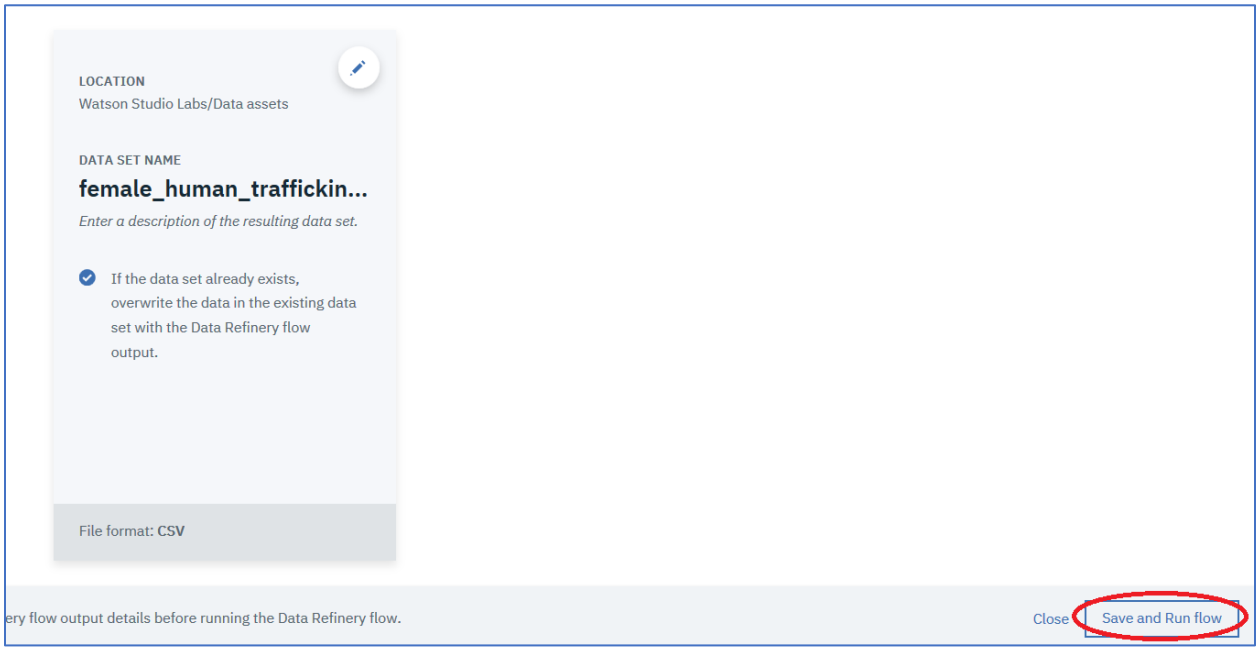
6. Click on the **pencil** icon  to modify the **DATA SET NAME**.



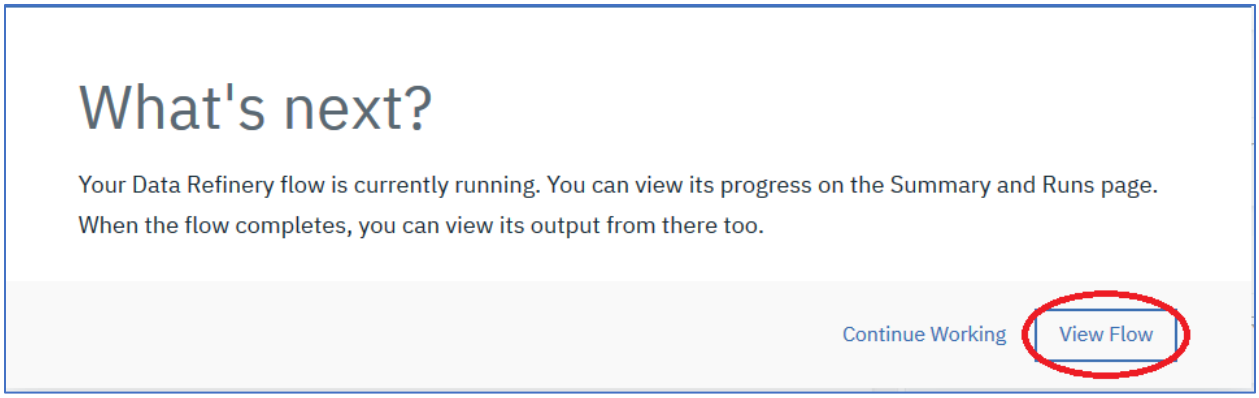
7. Change the DATA SET NAME to **female_human_trafficking_output** and click on the check icon 



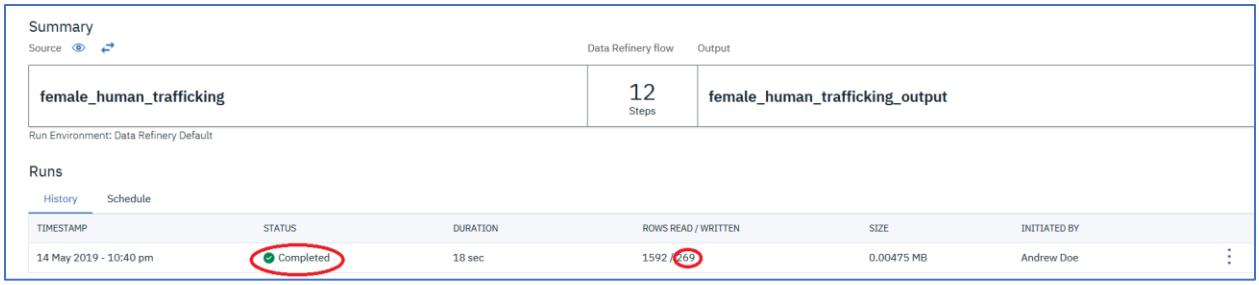
8. Click on **Save and Run flow**



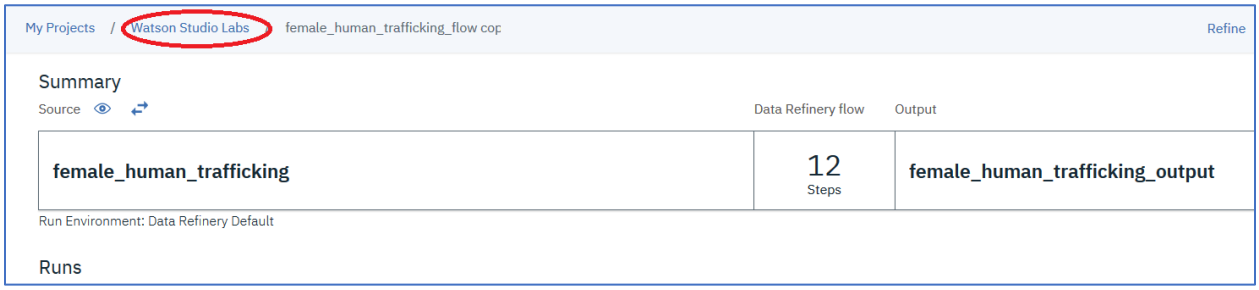
9. Click on **View Flow**.



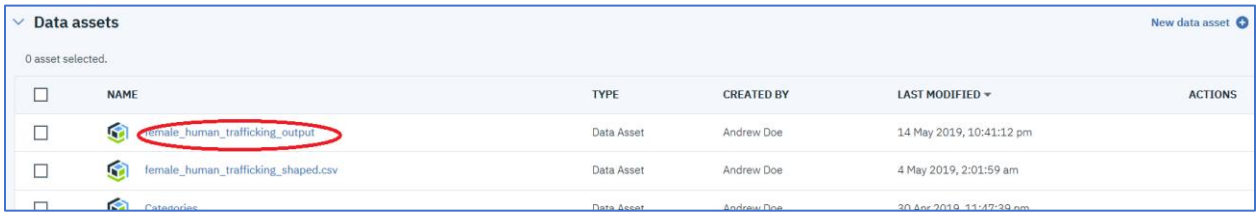
10. The flow has completed successfully, and 269 rows have been written to the female_human_trafficking_output data asset.



11. Click on **Watson Studio** to return to the **Assets** page.



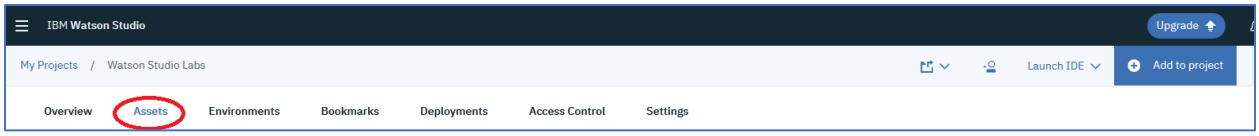
12. The female_human_trafficking_output asset has been added to the **Data Assets**.



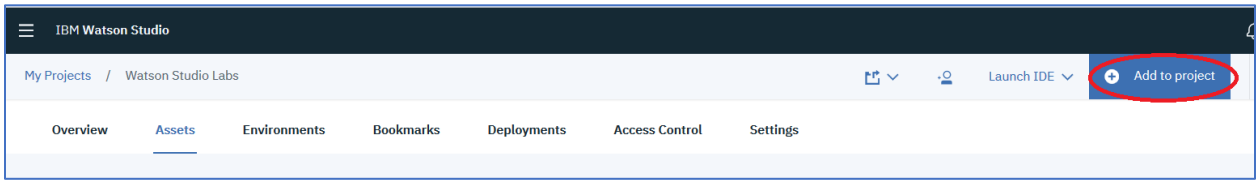
.

Create a Model to predict VETTING_LEVEL

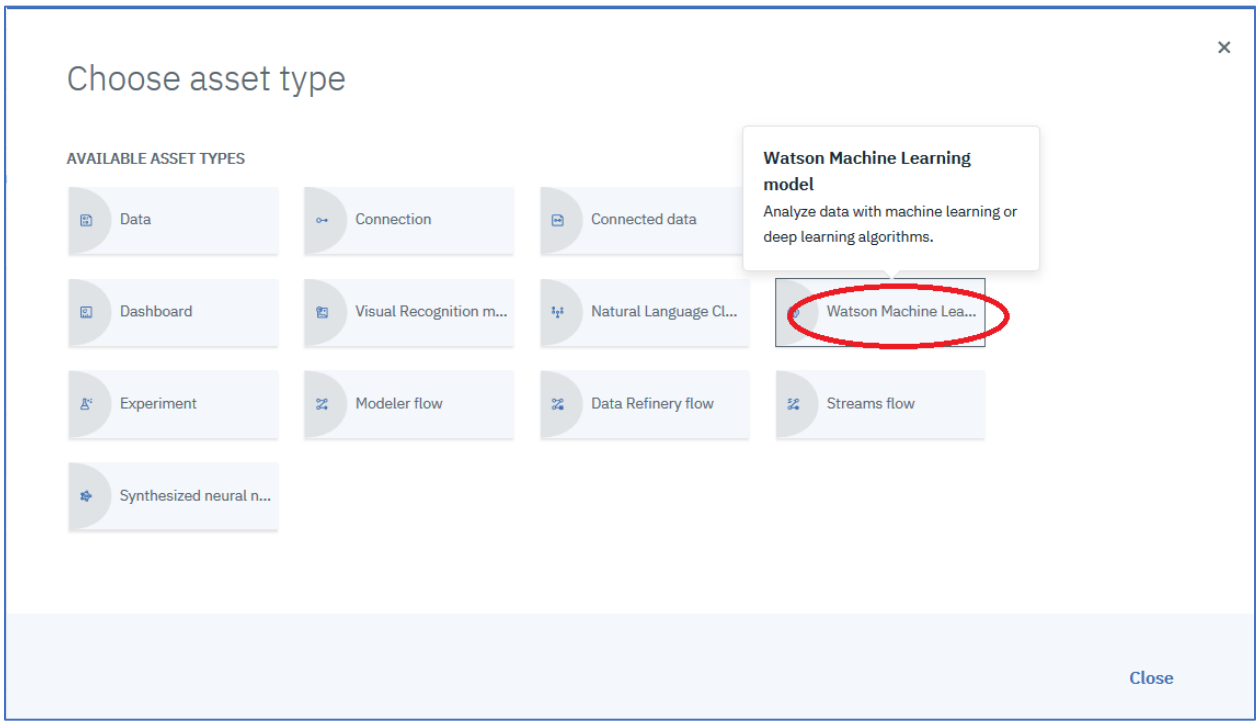
1. If not on the **Assets** page, click on the **Assets** Tab



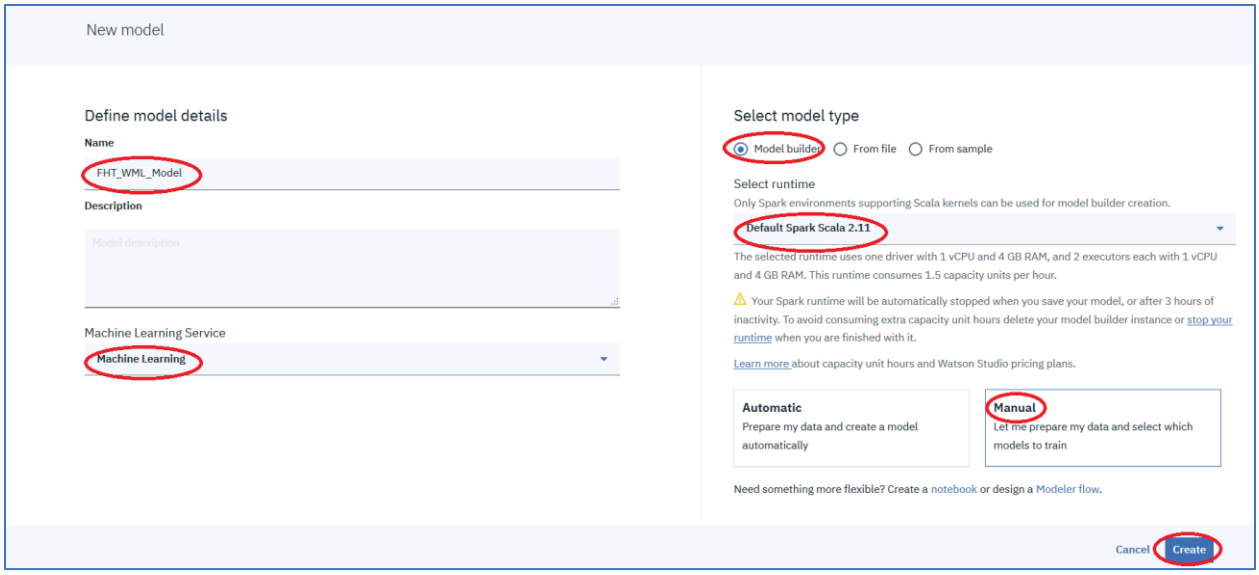
2. Click on **Add to project**.



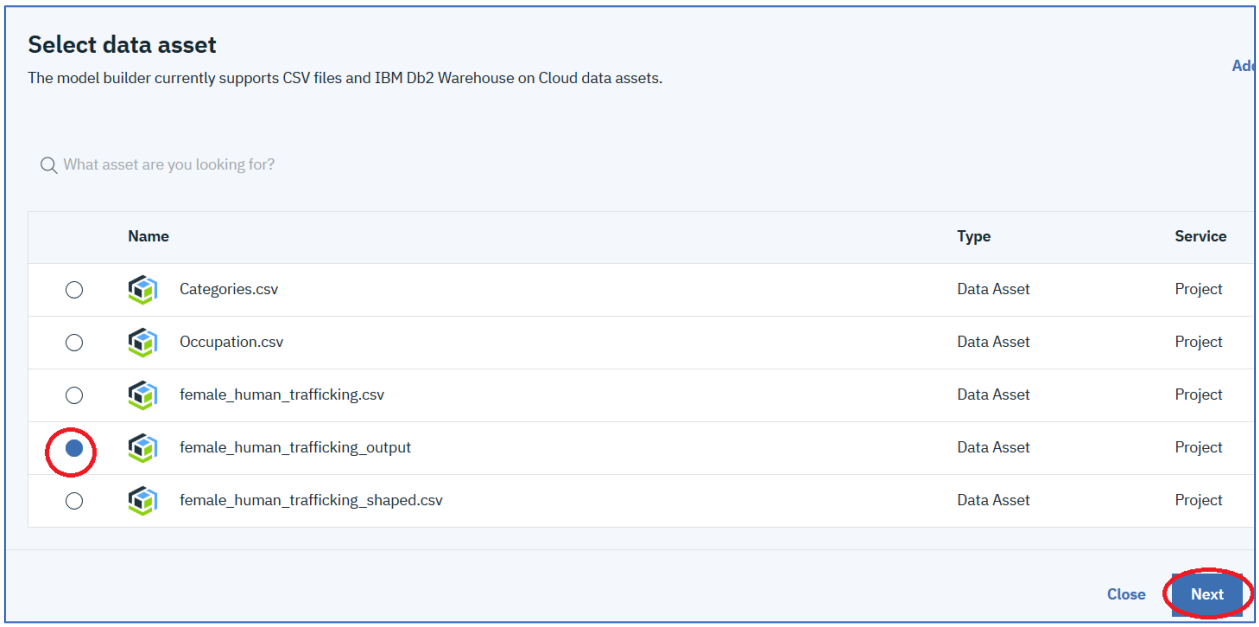
3. Click on **Watson Machine Learning**.



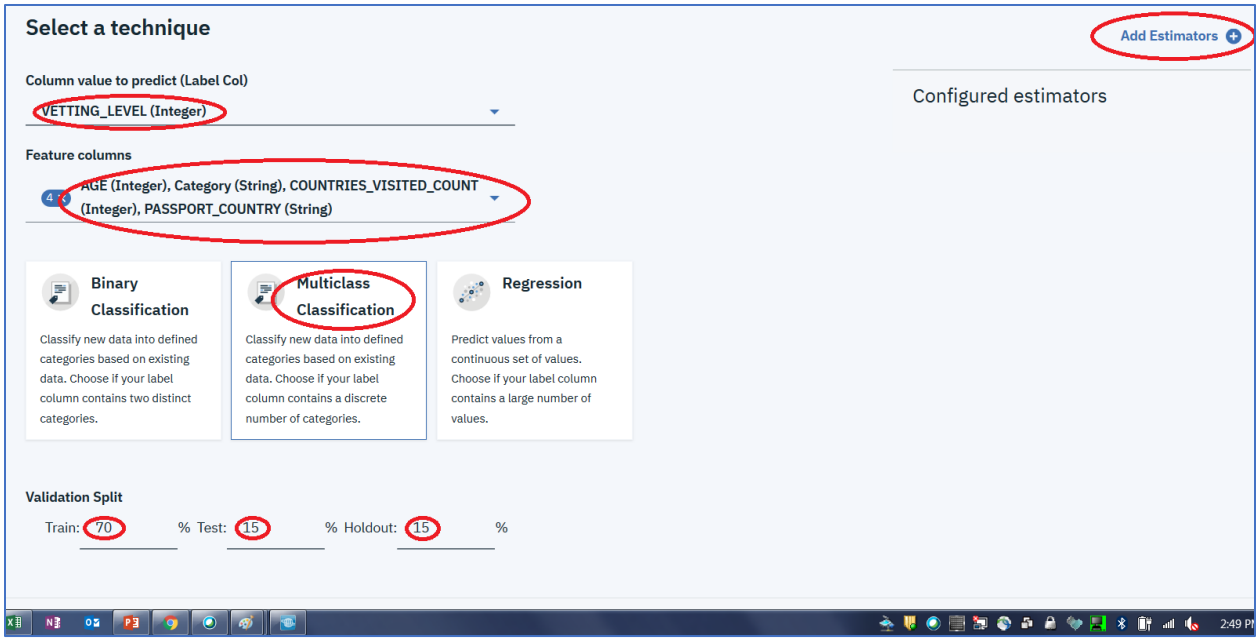
4. Enter a Model **Name** (eg FHT_WML_Model), optionally a **Description**, leave the default for the **Machine Learning Service** (should be the one created in the pre-reqs), leave **Model builder** selected, select the **Spark Scala Service**, select **Manual**, and click on **Create**.



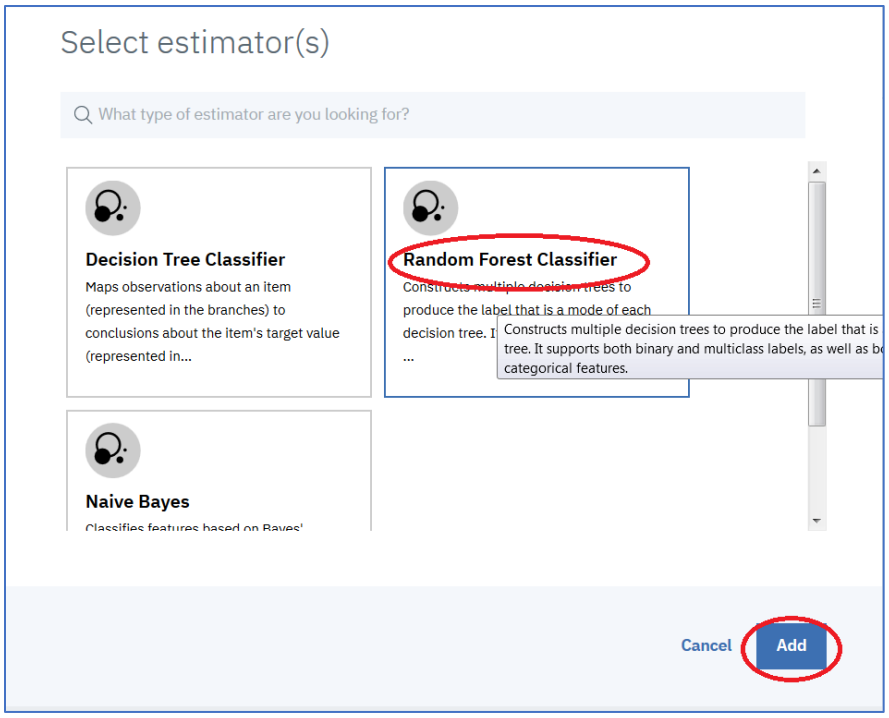
5. Click on the **female_human_trafficking_output** and click on **Next**



6. For **Column value to predict (Label Col)** select **VETTING_LEVEL**. For **Feature columns** select the following features (**AGE, Category, COUNTRIES_VISITED_COUNT, and PASSPORT_COUNTRY**). Click on the **Multiclass Classification** Box. Adjust the **Validation Split** to be 70%, 15%, 15%. Click on **Add Estimators** to add the specific models to use.

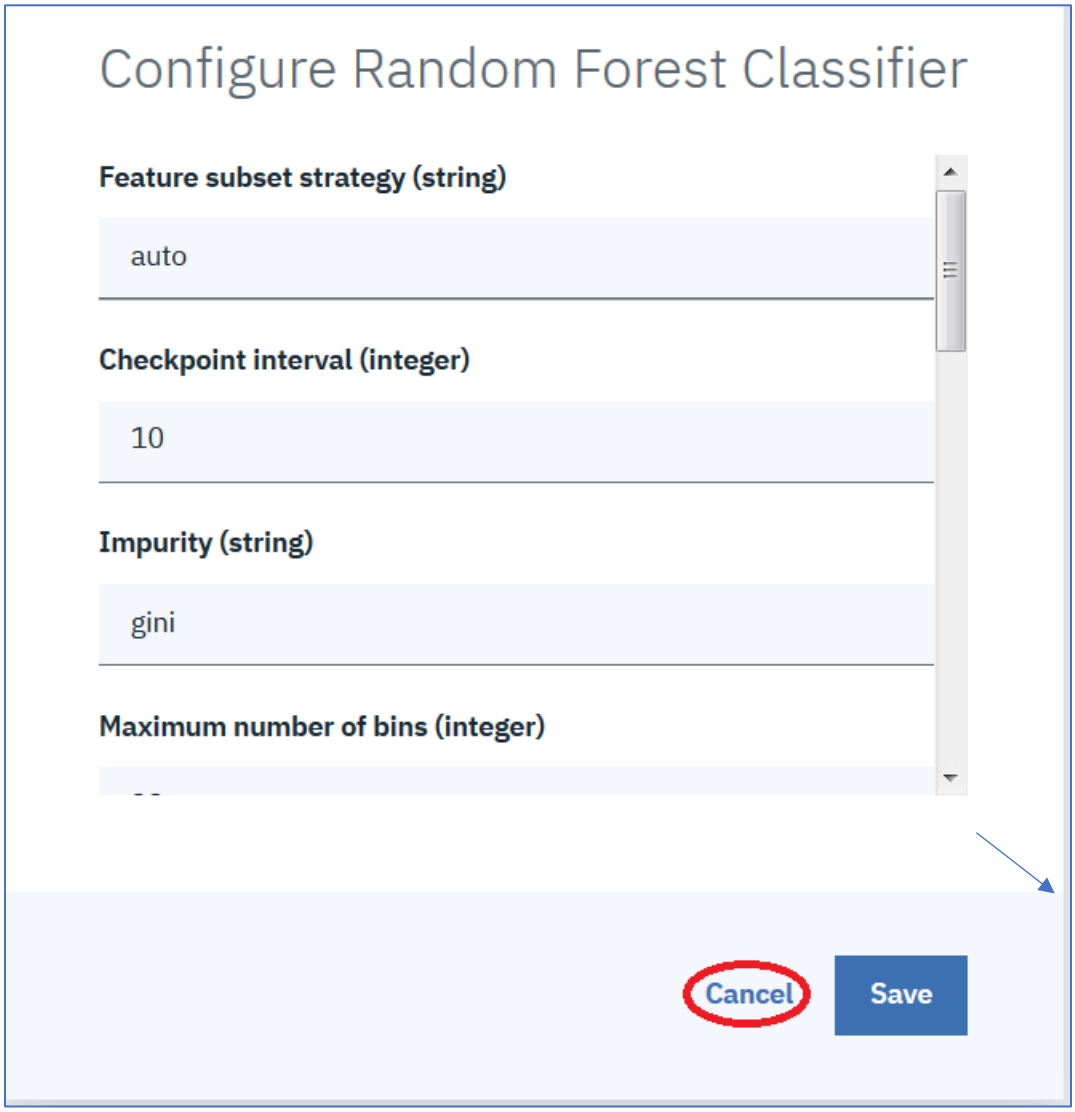


7. Select **Random Forest**. You can select more if you wish to see the results of multiple models. Select **Add**.

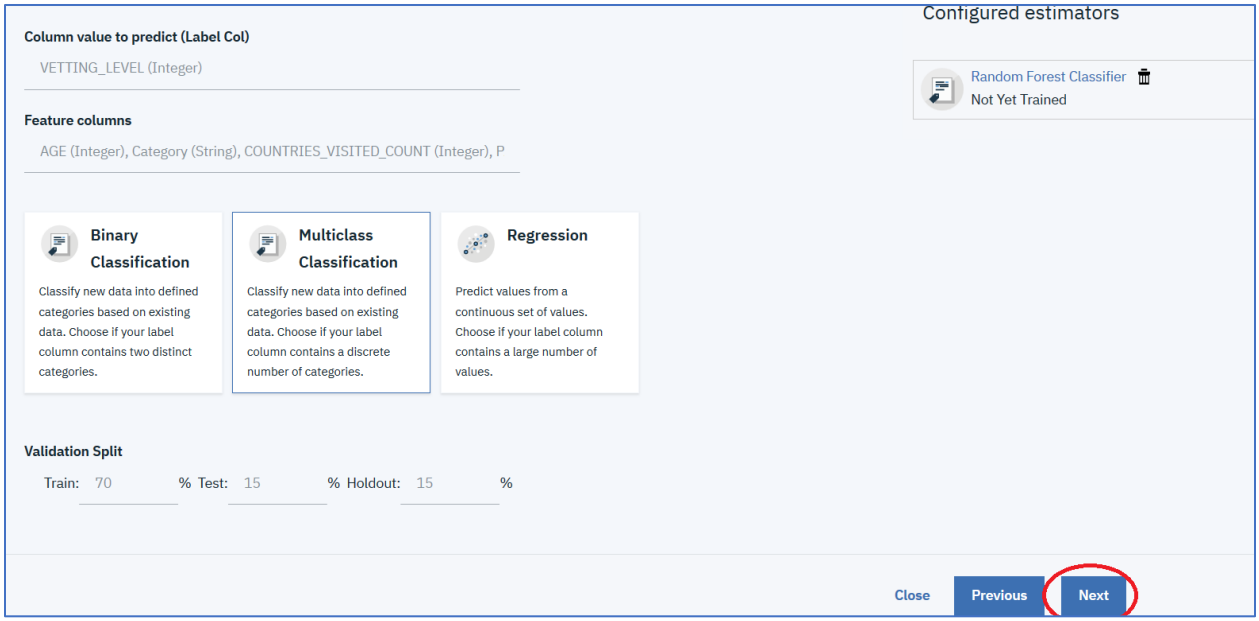


8. Note you can adjust the algorithm's hyperparameters by clicking on **Random Forest**.

9. For now, we will leave the hyperparameters unchanged, so click **Cancel**.

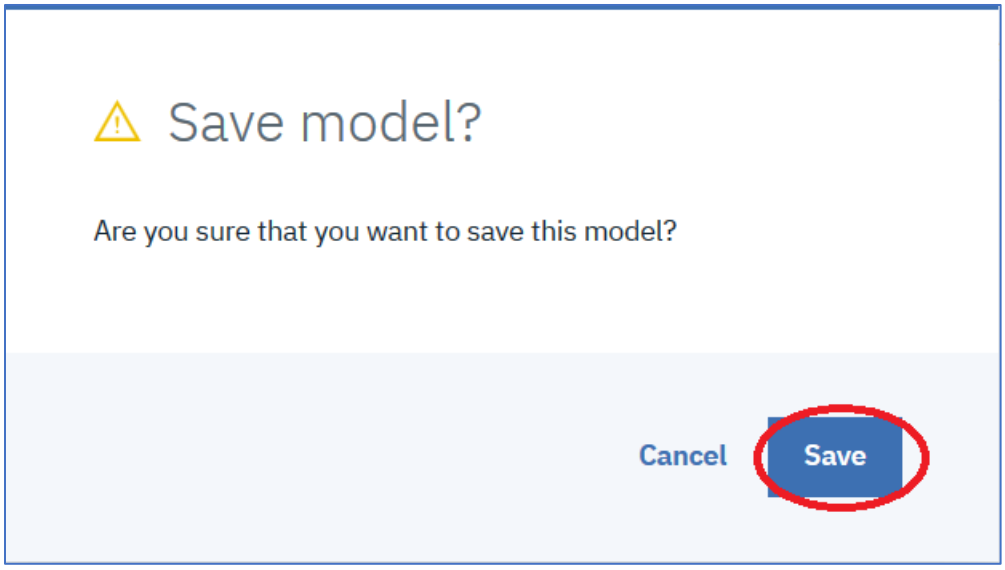


10. Select the **Next** button.

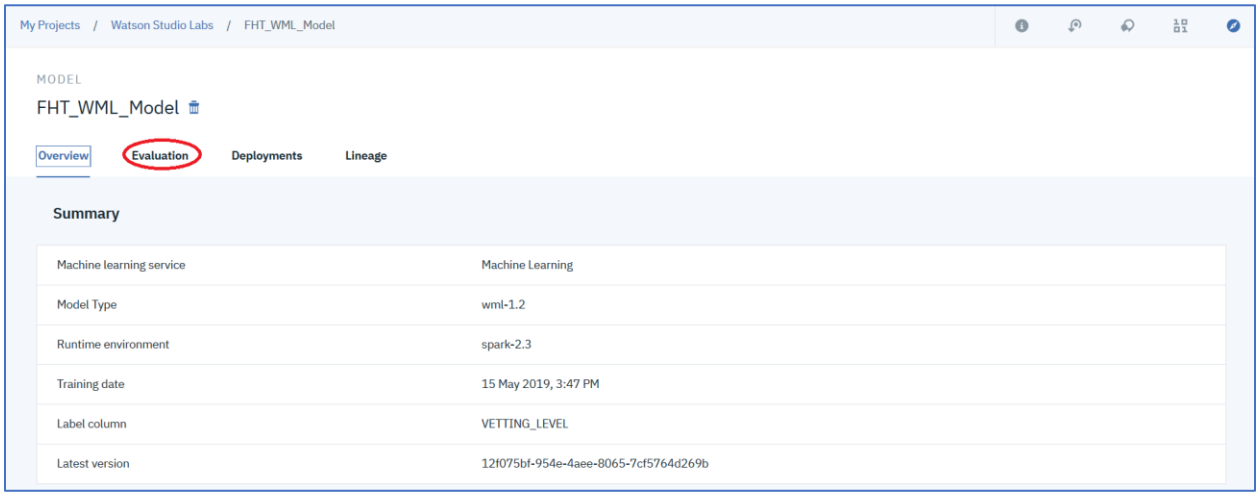


11. The system trains and evaluates each model. If more than one model was selected, the models would be listed in descending order of quality with the best result at the top. Note: if a model fails to run (rare, but happens), select Previous, delete the estimator, and re-add it. Then run again. Click on **Random Forest** (if it is the best) and then click **Save**.

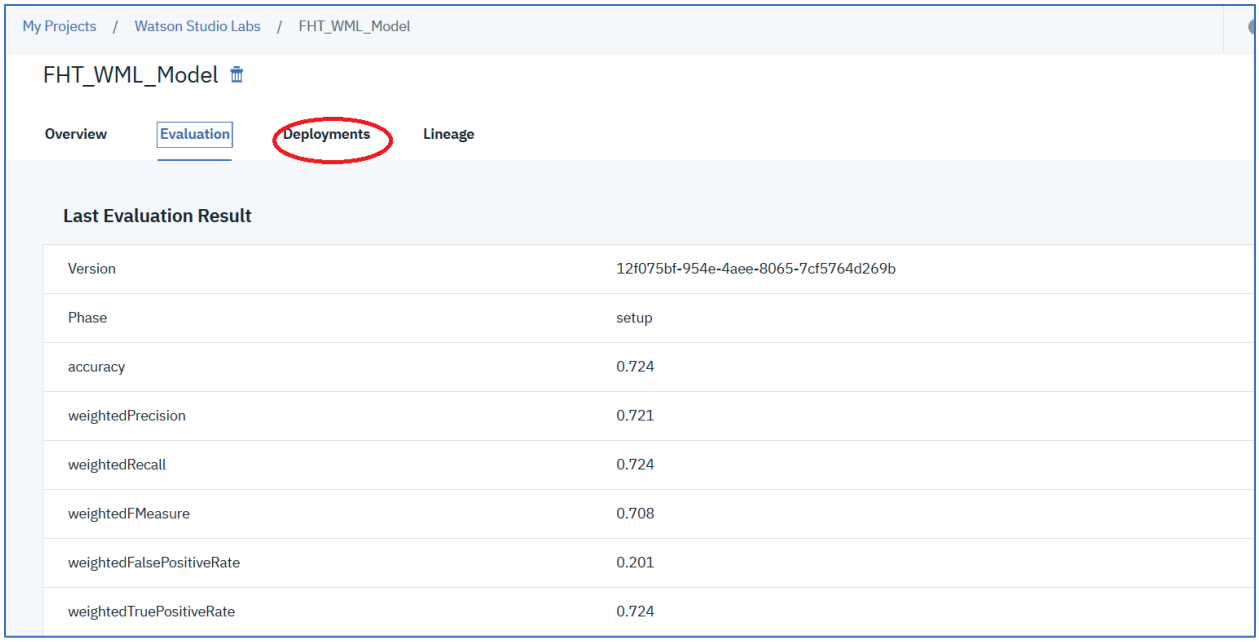
12. Click **Save** again on the next screen. The model is saved in the Model Repository.



13. The system displays the model training summary. Click on **Evaluation.**



14. The system displays the recorded evaluation statistics for the run. You can also set up Continuous Learning (Performance Monitoring) on this screen. We will not do this now. Select **Deployment**.

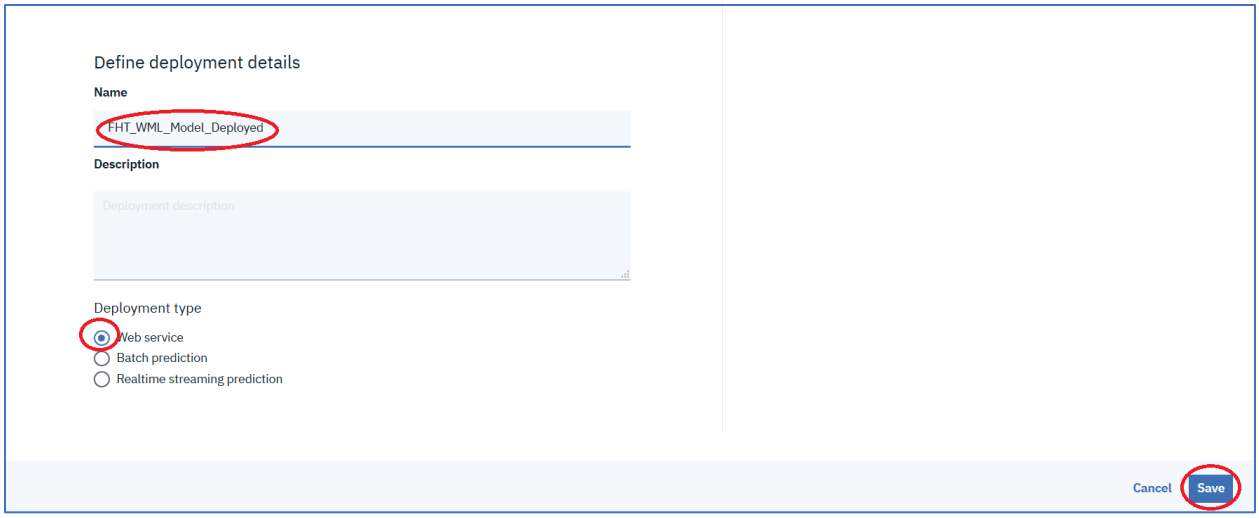


Deploying the Model

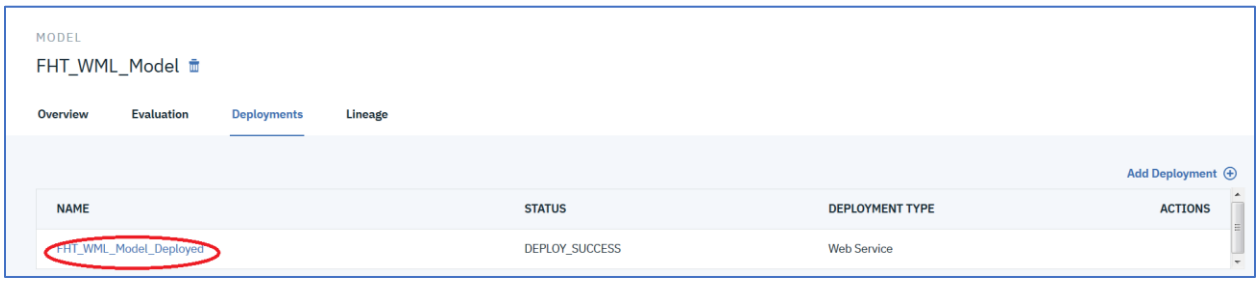
We can deploy the model from the model repository to enable applications to invoke it via an API call. This is a called a Web Service deployment or Online deployment.

1. Click on **Add Deployment**

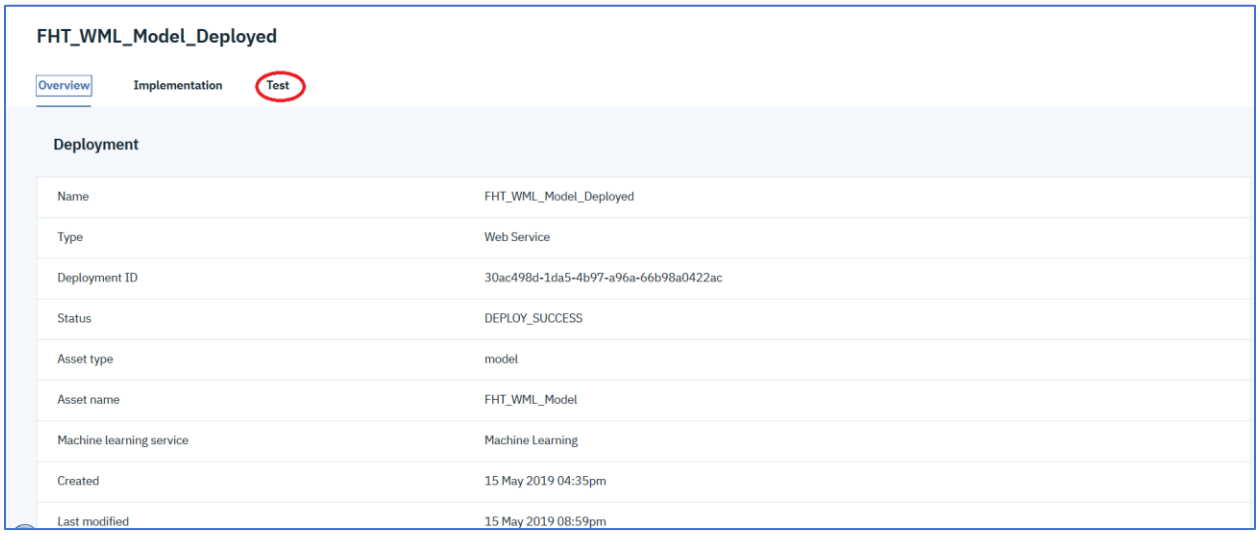
2. Enter **FHT_WML_Model_Deployed** for **Name**, optionally a **Description**, select the **Web Service** radio button, and click on **Save**.



3. The system responds with an acknowledgement that the model was successfully deployed. Click on **FHT_WML_Model_Deployed Titanic_Deployment** to test the deployed API.



4. The system displays information about the deployed service. Click on **Test** to test out the API.



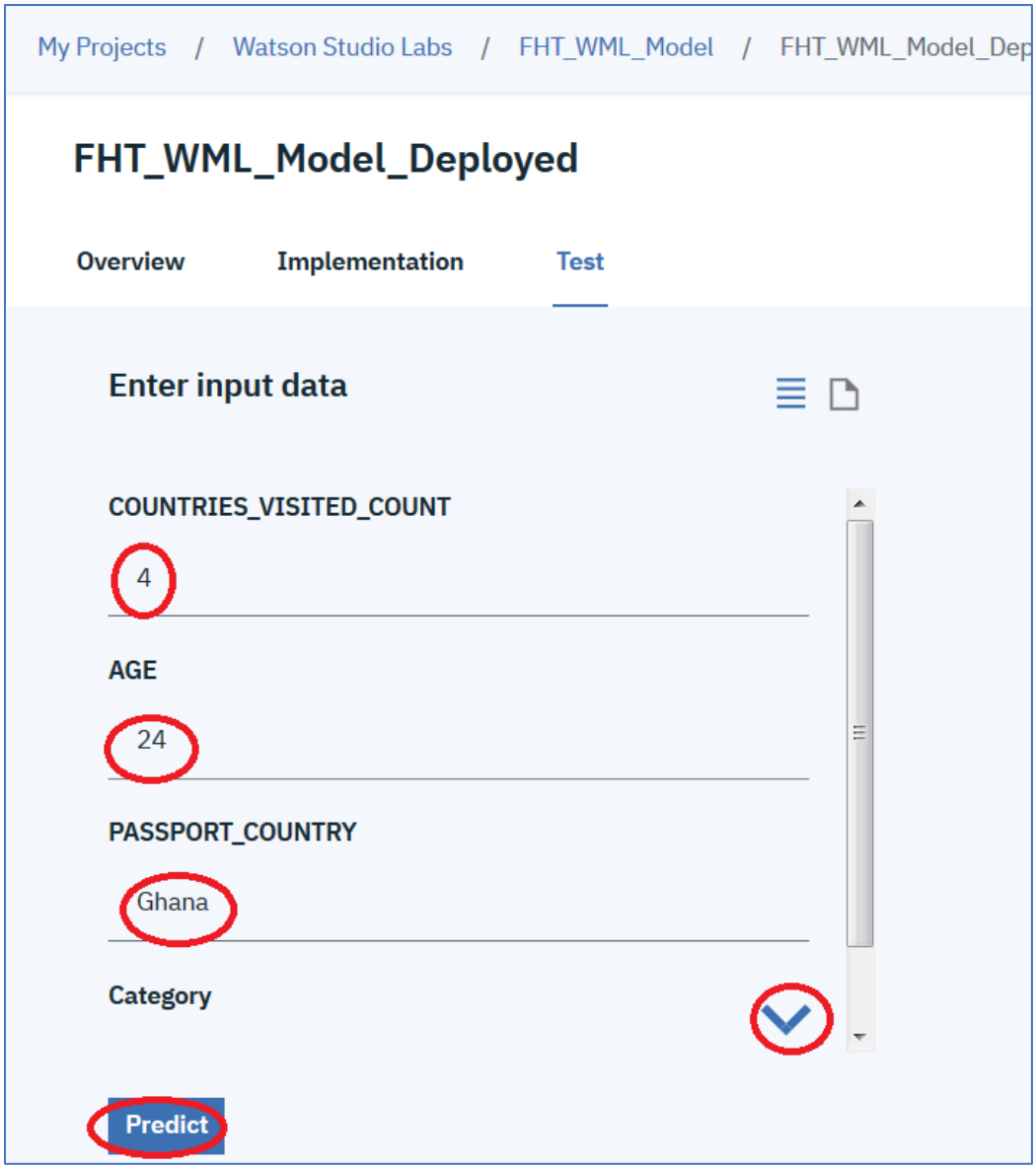
5. Enter values for the following fields:
COUNTRIES_VISITED_COUNT: 4

AGE: 24
PASSPORT_COUNTRY: Ghana

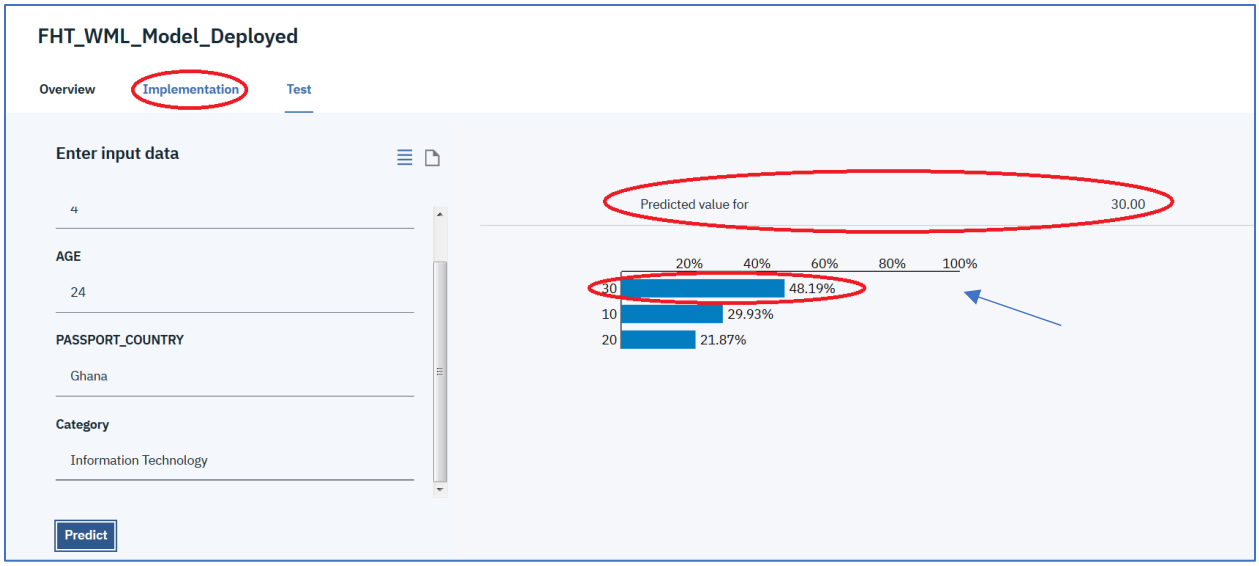
Scroll down

Category: Information Technology

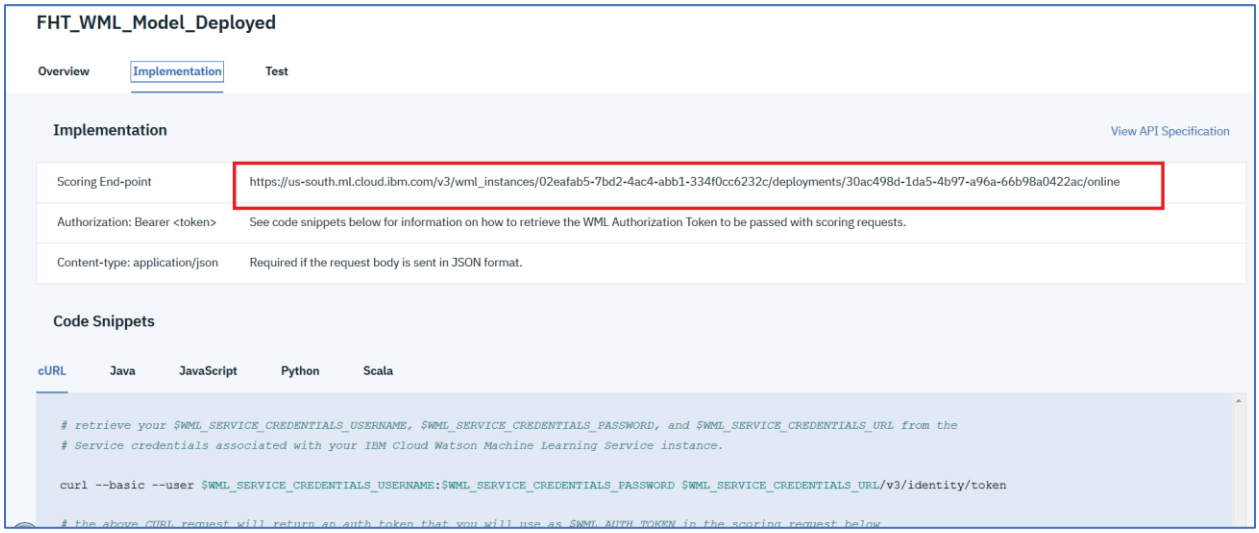
and then click on **Predict**.



7. The predicted result should be returned which indicates that the model has been successfully deployed. Now click on the **Implementation** tab.



8. The Implementation panel provides information for the application developers to invoke the deployed model. It includes sample code in various programming languages and the scoring endpoint to be used when invoking the web service. Open Windows Notepad to copy and paste the scoring endpoint (just leave this panel available to cut and paste the scoring endpoint). We will need the scoring endpoint in the next section.



Deploy a simple web front-end to invoke the Watson Machine Learning service

This section provides an example of a simple Python Flask web front-end application that invokes the Female Human Trafficking risk prediction deployed model, demonstrating embedding machine learning in a web app. You will click on a link below that will deploy the

sample Python web application into your IBM Cloud account. A toolchain will be set up for continuous delivery of the application. The application code will be cloned from a public Git repository into a private Git repo in your account that will be set up as part of the toolchain. Each time you commit changes to the repo, the app will be built and deployed.

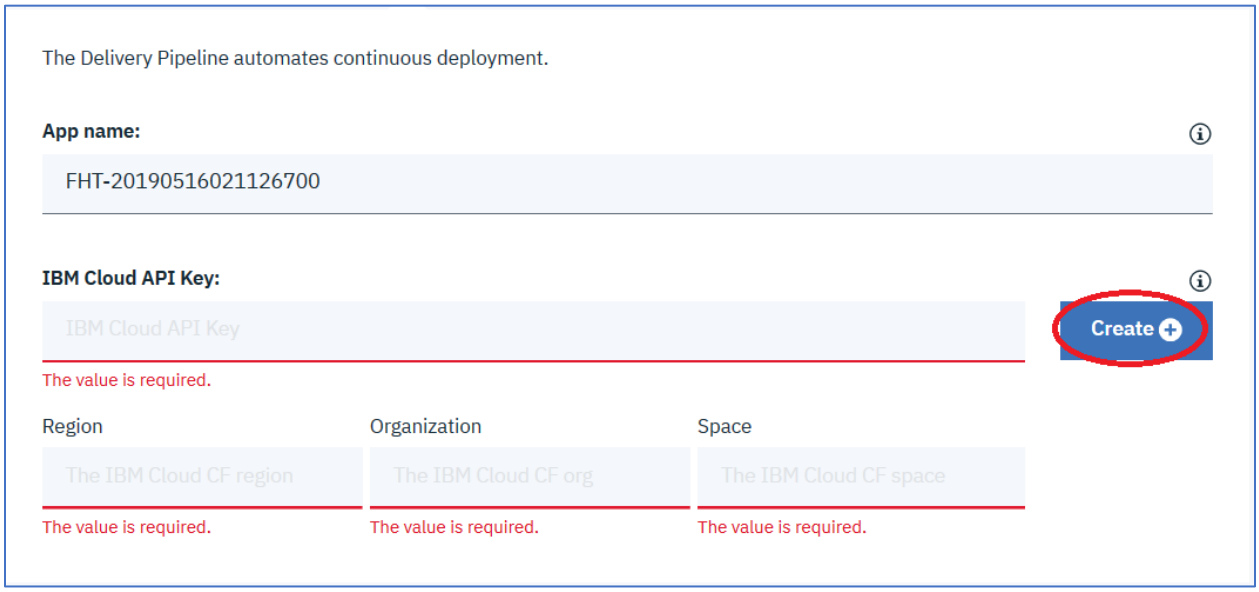
The toolchain uses tools that are part of the Continuous Delivery service. If an instance of that service isn't already in your account, when you click **Deploy**, it is automatically added with the free Lite plan selected.

The steps below guide you in configuring the application to connect to your Watson Machine Learning service, and to update the application with the deployed model's scoring endpoint.

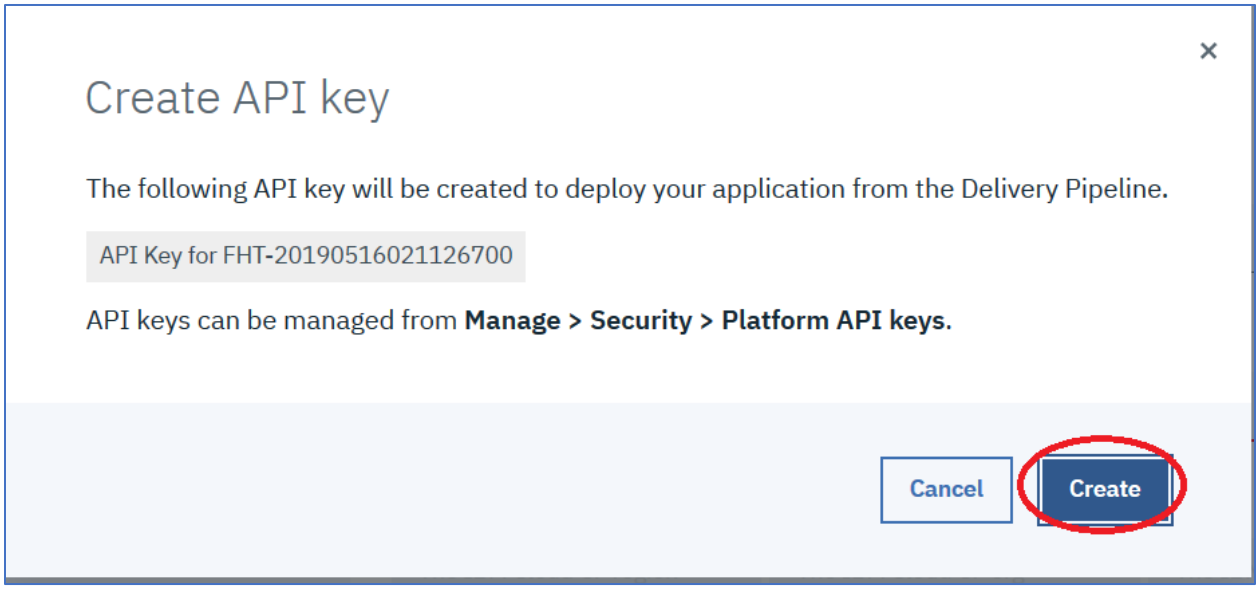
1. Click on the **Deploy to IBM Cloud** link below to deploy a sample Python Flash web application into your IBM Cloud account. Note you may get this message – "*An IBM Cloud account is required. To get started, click Log In or Sign Up at the top of this page*". If you get this message, click on **Log In.**

   Deploy to IBM Cloud

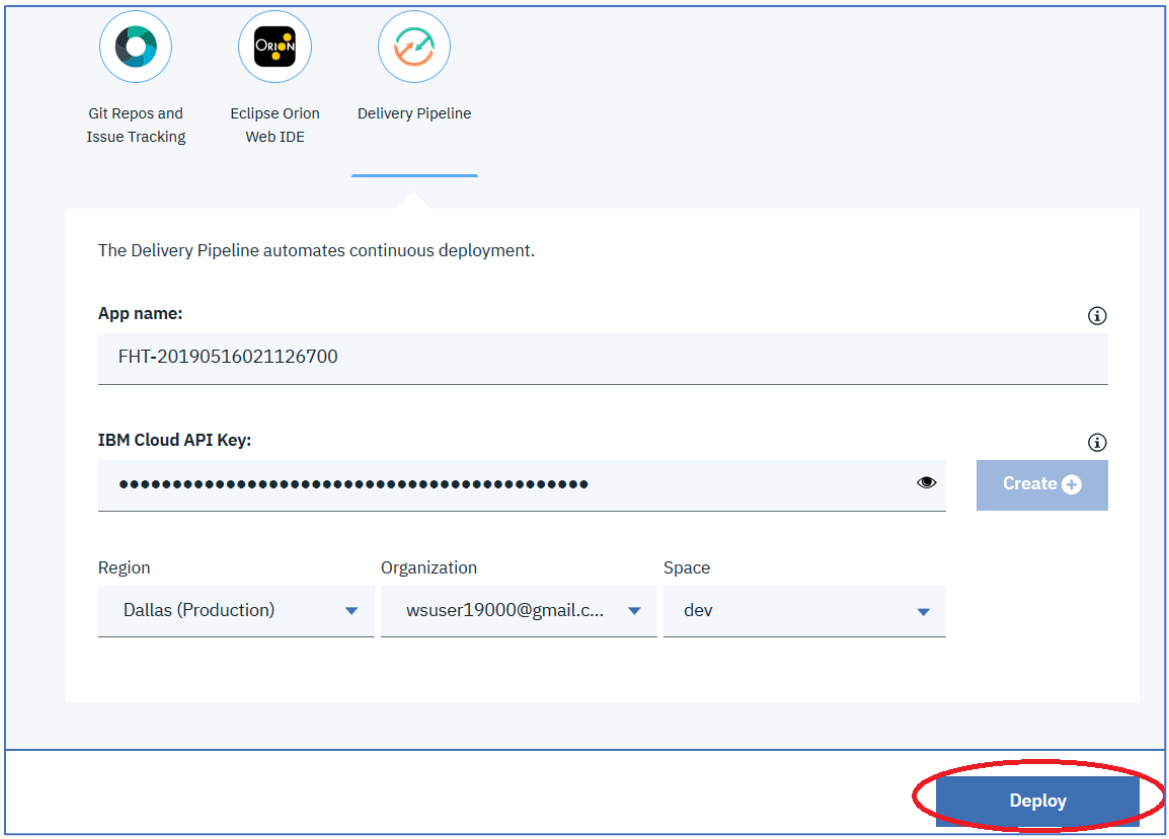
2. Scroll down to the bottom. Click on the **Create+** button to create an IBM Cloud API key.



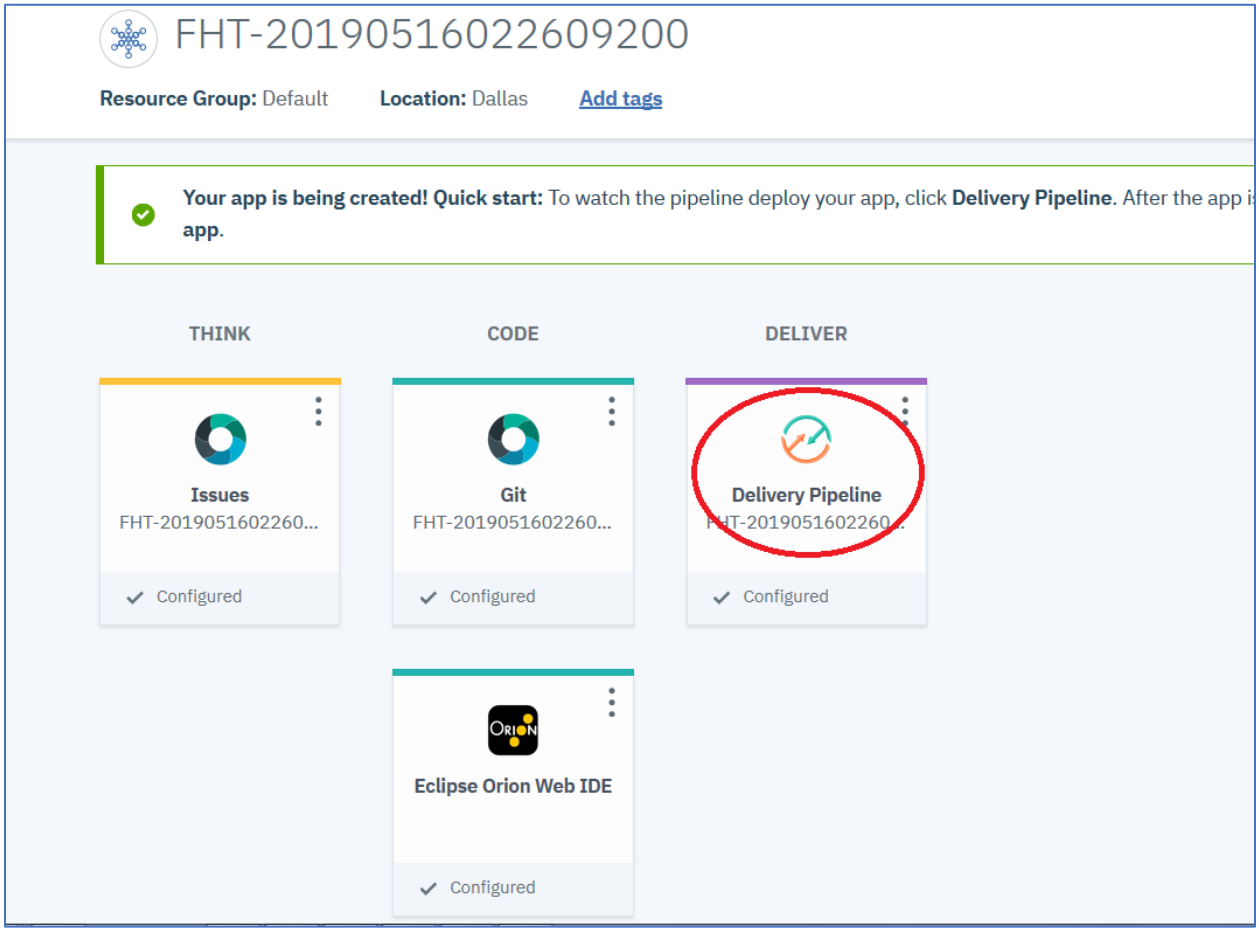
3. Click on the **Create** button.



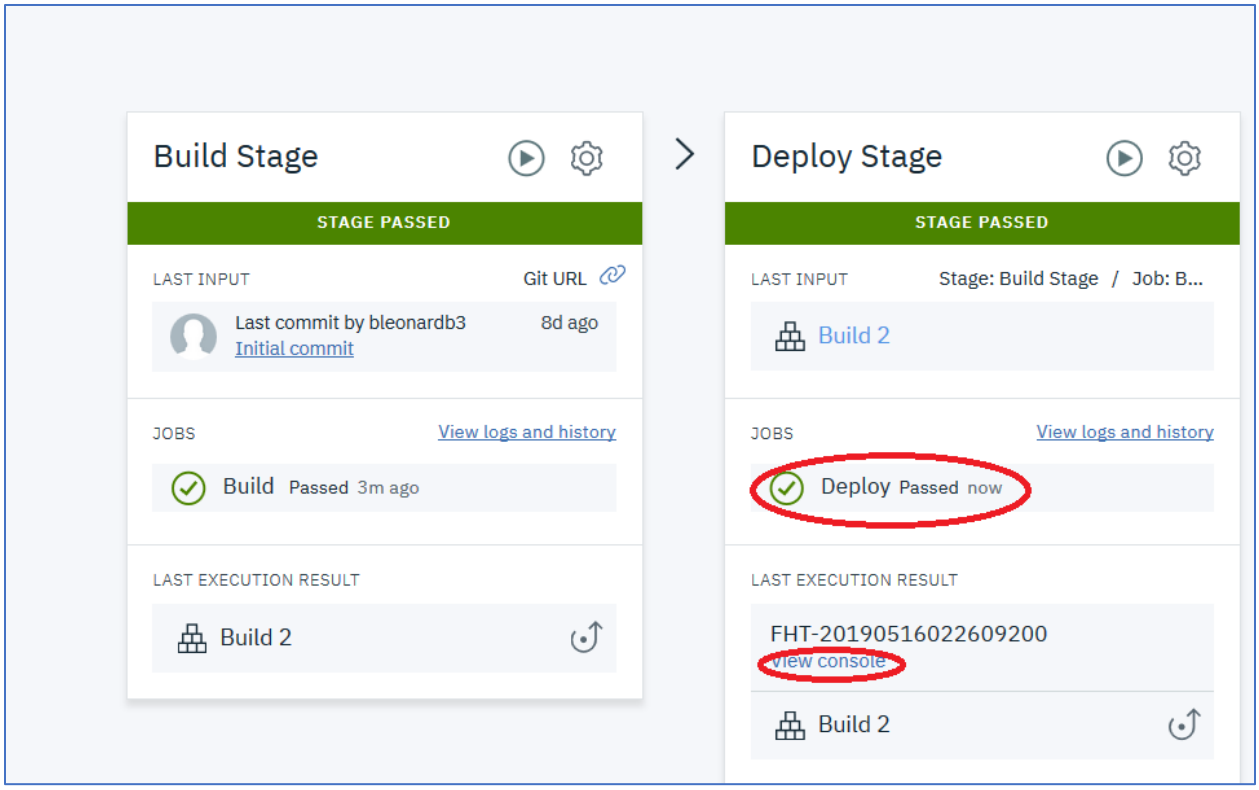
4. Please wait until the Region, Organization, and Space are filled in. **Note that these must match the corresponding Region, Organization, and Space where the FHT_WML_Model_Deployed is deployed.** Switch the **Region** to Dallas (Production), the Organization should display the userid, and the space should be filled in as well. If this is not the case, try a different **Region**. Click on **Deploy**.



5. Your app is being created! To watch the pipeline deploy your app, click **Delivery Pipeline**.

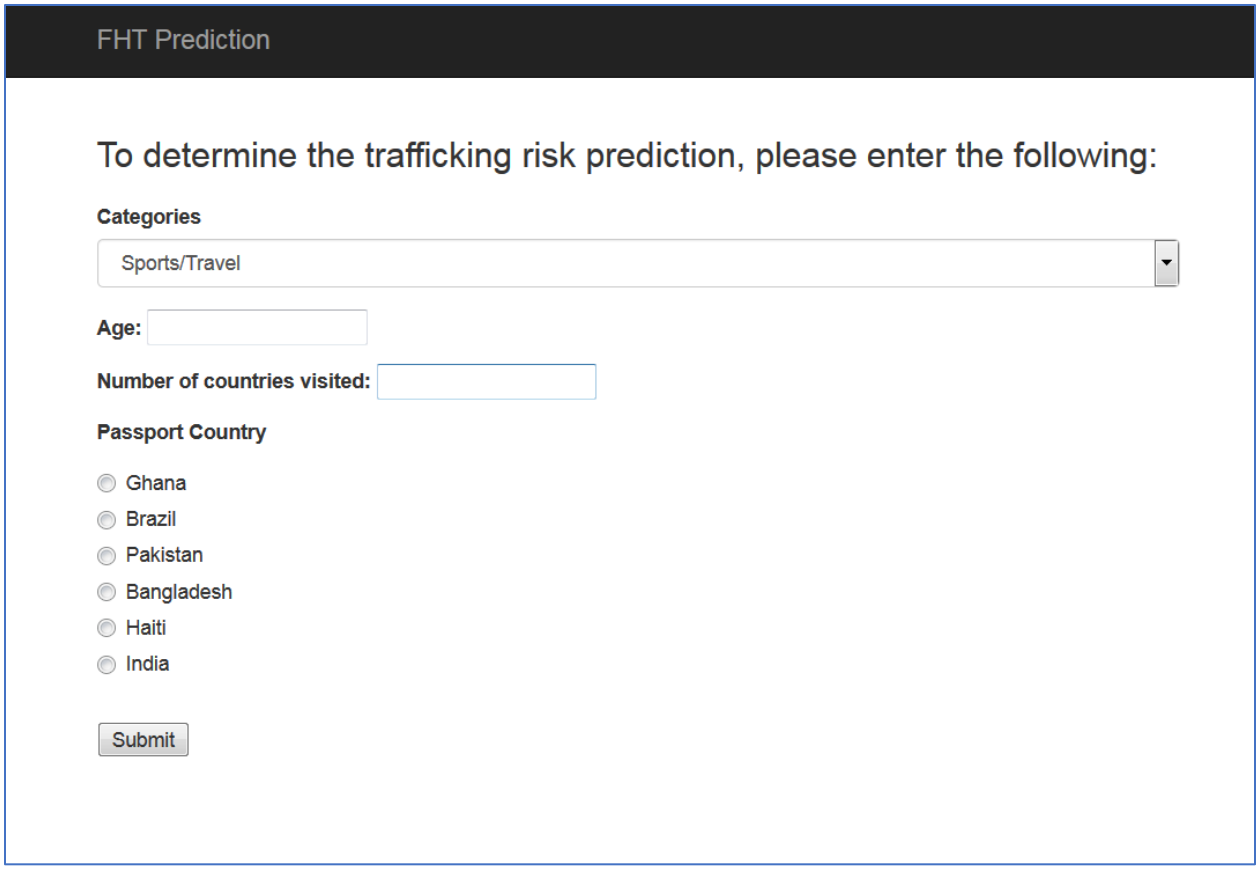


6. After the app is deployed successfully (should say Deploy Passed in the Deploy stage- may take about 2 minutes), view the running app by clicking on **View Console**

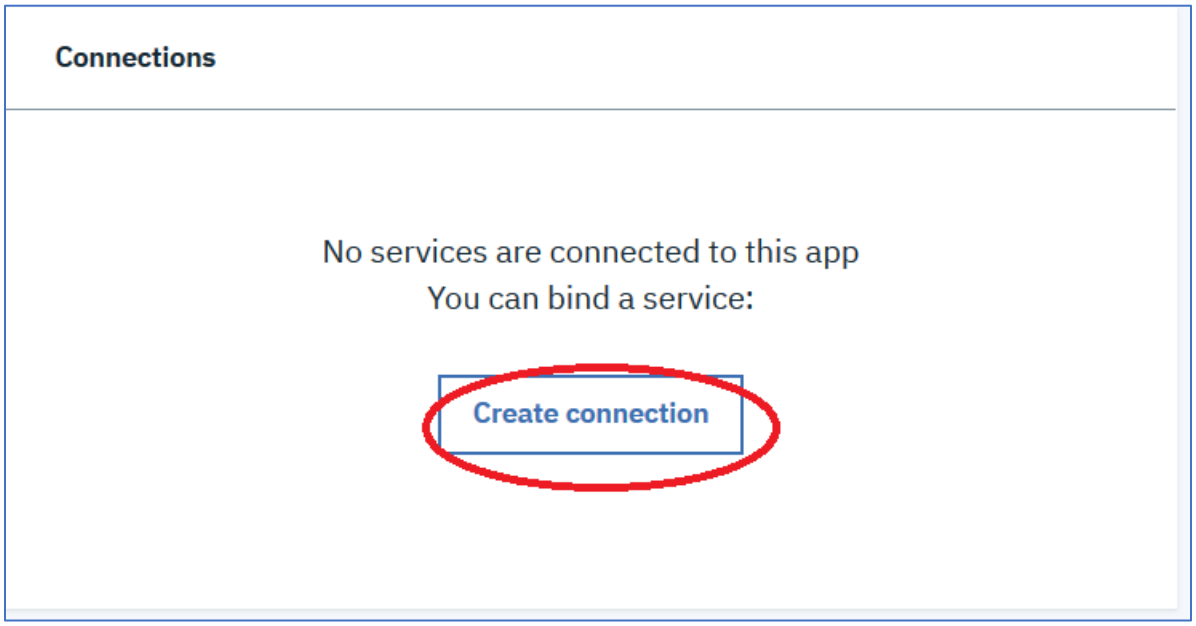


7. Click on **Visit App URL**

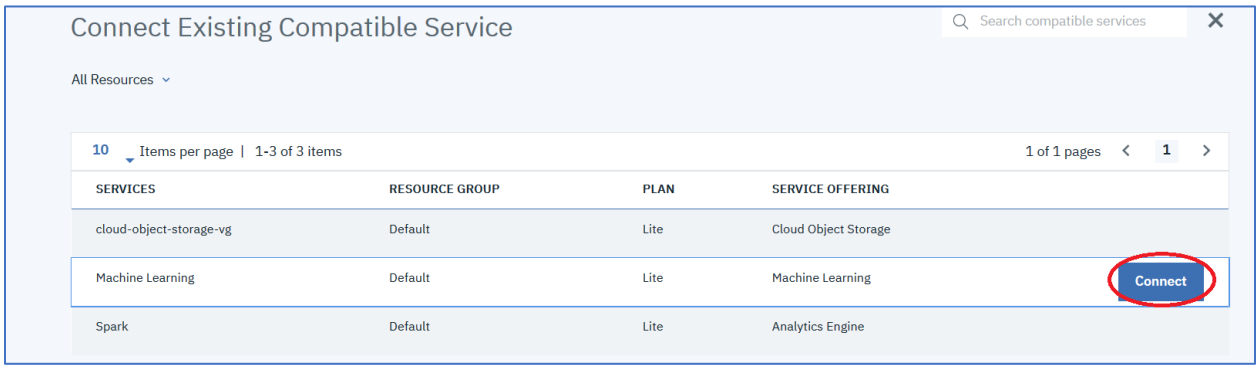
8. The web form collecting the FHT data should appear. Note that the application is not functional until we connect it to the Watson Machine Learning service so if you Submit you will get an error! Close the FHT Prediction browser tab.



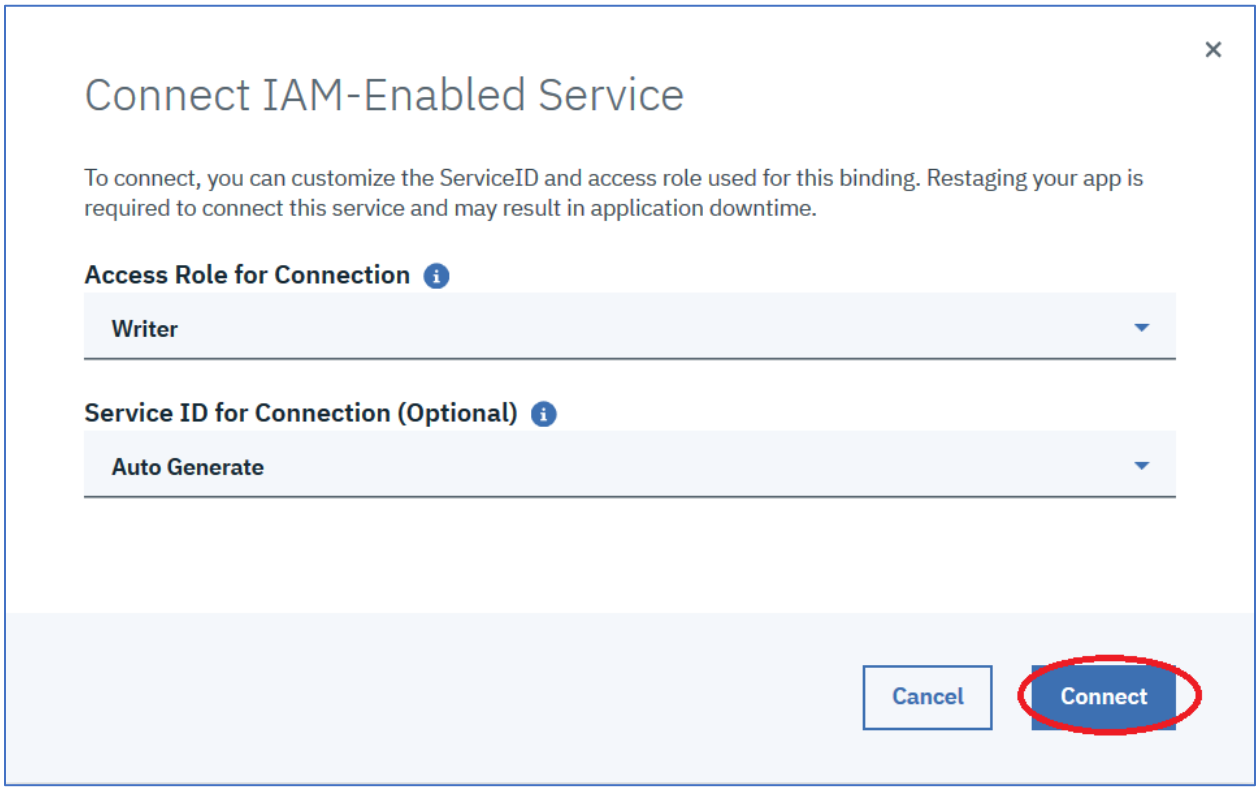
9. We are now going to connect the application to the Watson Machine Learning service that was created earlier. Scroll down until you see the Connections panel. Click on **Create Connection**.



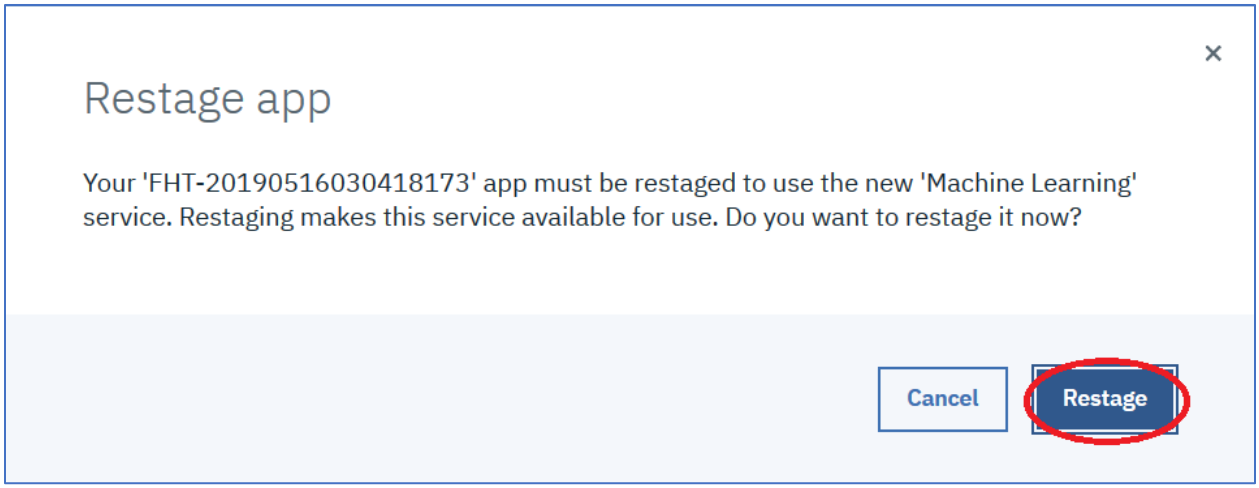
10. You should see at least 3 services listed, a Cloud Object Storage service, a Spark service, and a Watson Machine Learning service. Point the cursor on the **Machine Learning** service for your application, and then click on **Connect**.



11. A **Connect IAM-enabled service** pop up will appear. Click **Connect**.

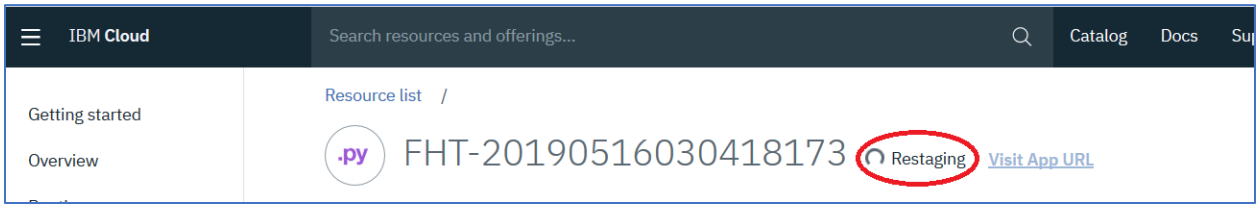


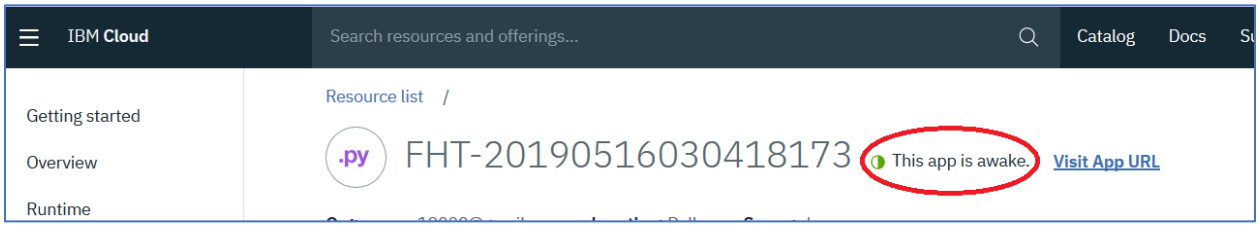
12. A **Restage app** pop up will appear. Click on **Restage**.



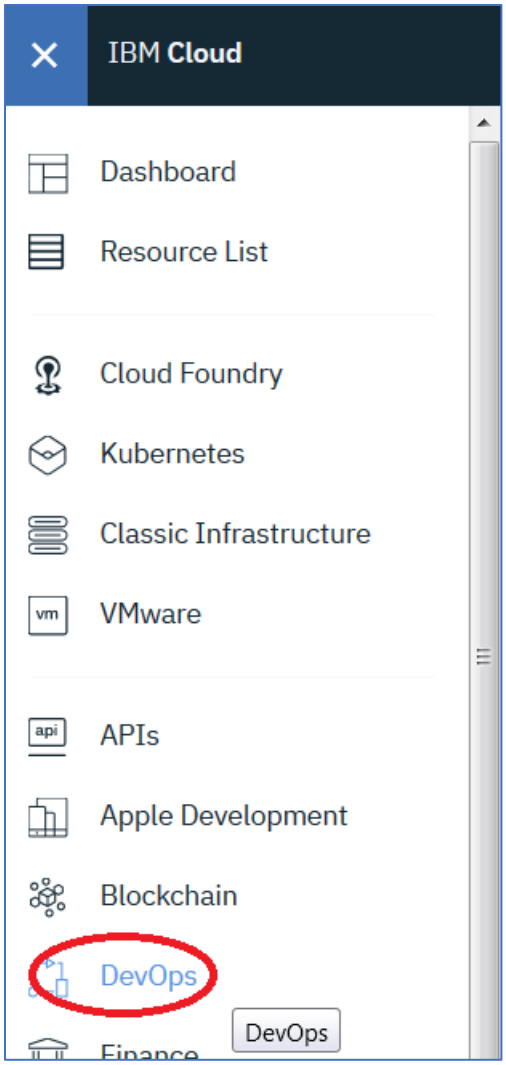
13. Wait for the application status to change from **Restaging** to 

, or something similar.

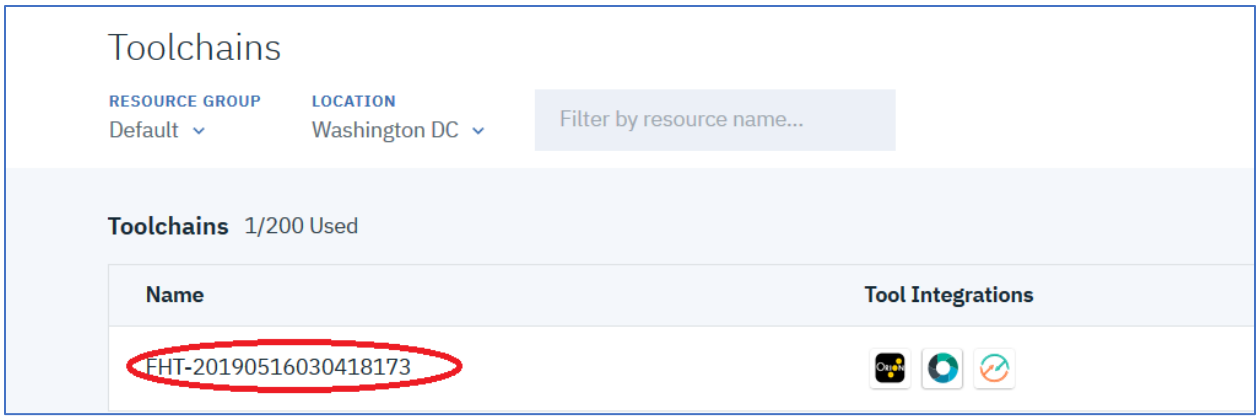




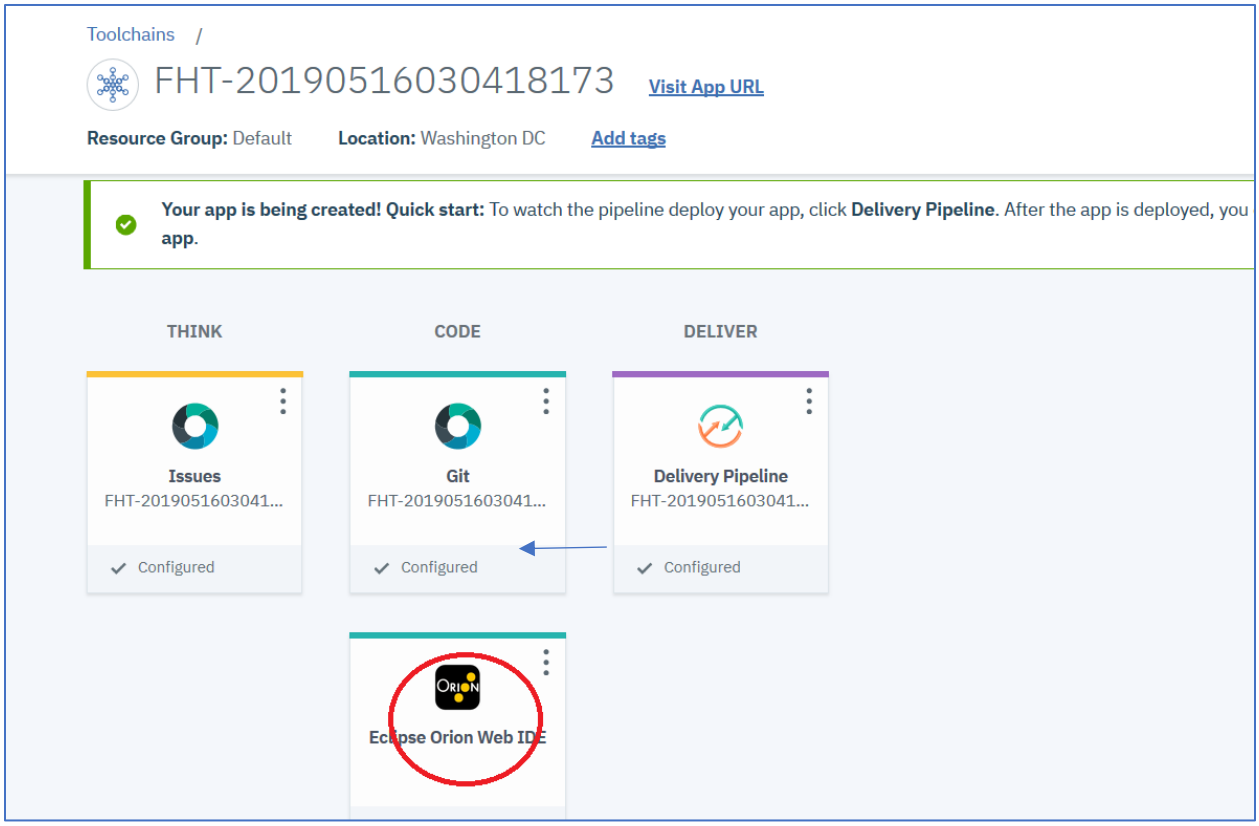
14. We now have tied the web application to the Watson Machine Learning service. Note that the Watson Machine Learning service could have more than one deployed model available to select and then embed in the web application. We now need to copy the scoring endpoint, which we previously copied and pasted into Notepad, and paste it in the web application code. Click on the ☰ icon and click on DevOps in the pulldown to navigate to the Toolchain.



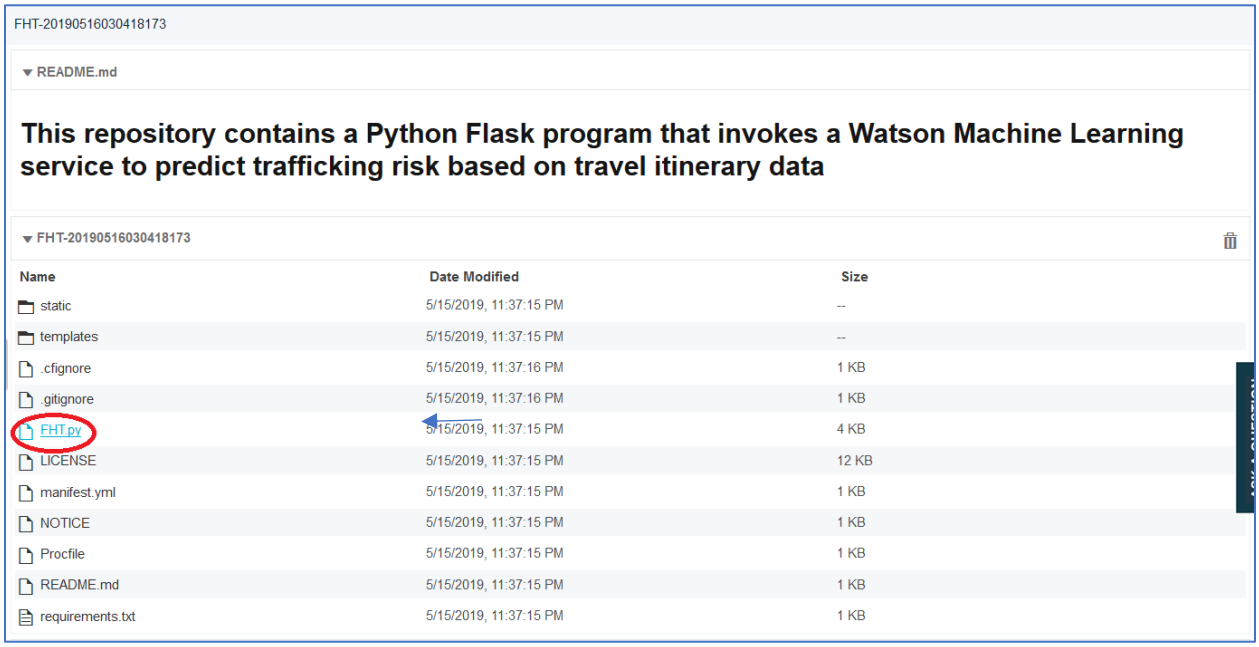
15. We are now going to paste the scoring endpoint into the application code. Click on the Toolchain (FHT-2019xxxxx below).



16. Click on the Eclipse Orion Web IDE. The IDE will enable the editing of the source code to update the scoring endpoint url.

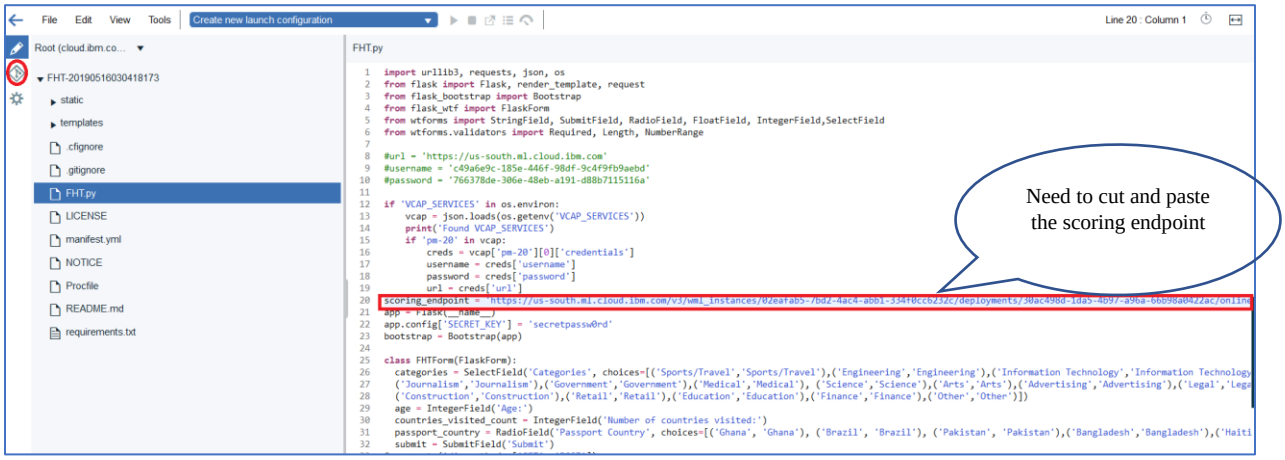


17. Click on the FHT.py file. This is a python source file.

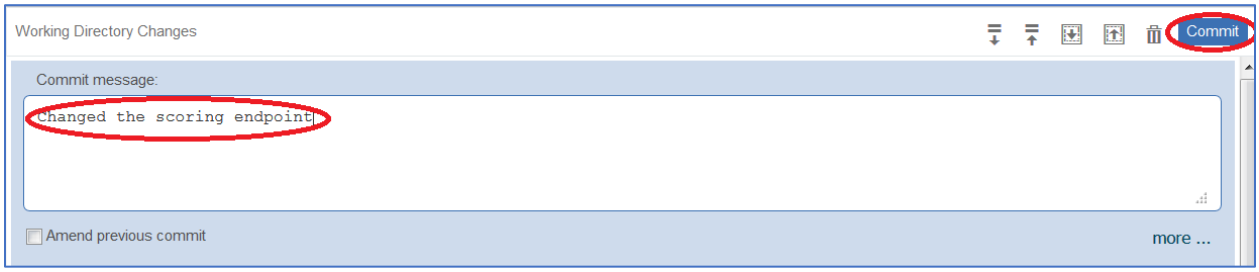


18. Go back to the Notepad file and copy the scoring endpoint to the clipboard. Look around line 20 in the FHT.py file for the "scoring endpoint =". Select the scoring endpoint value in line 20 (starting with https:// may want to use Shift-End to get to the end of the

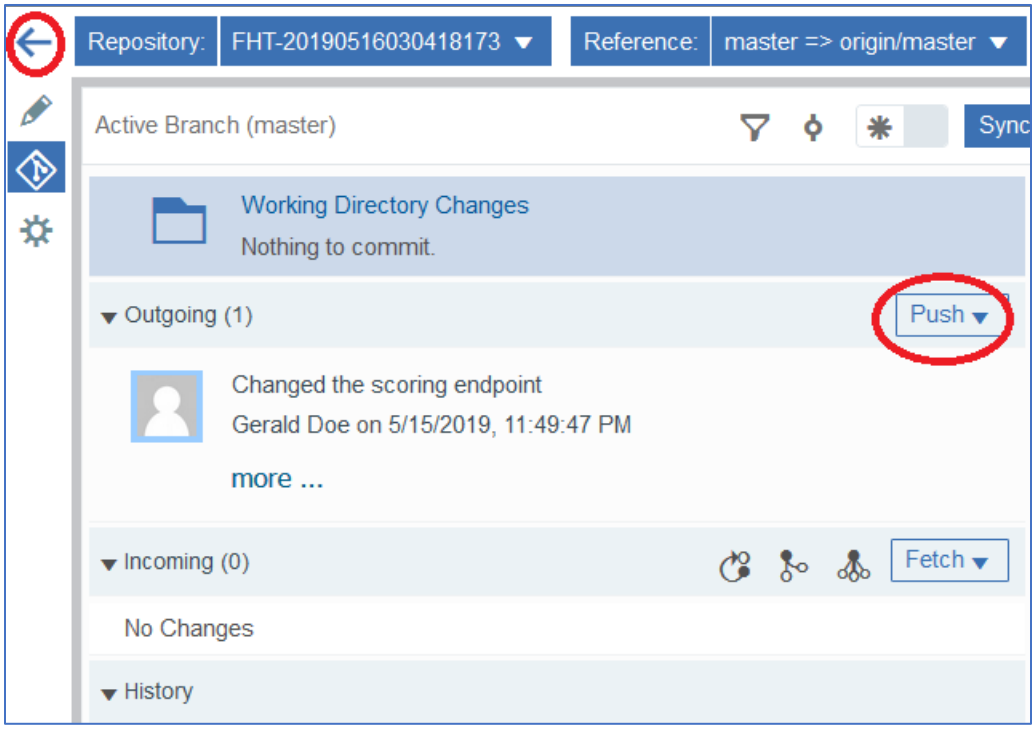
line, and then back up one space to not select the endpoint quote – if you do just make sure to put it back in). Enter Ctrl-V to paste the new scoring endpoint from your Notepad file. Enter Ctrl-S or File > Save to save the file. Then click on the  icon on the top left.



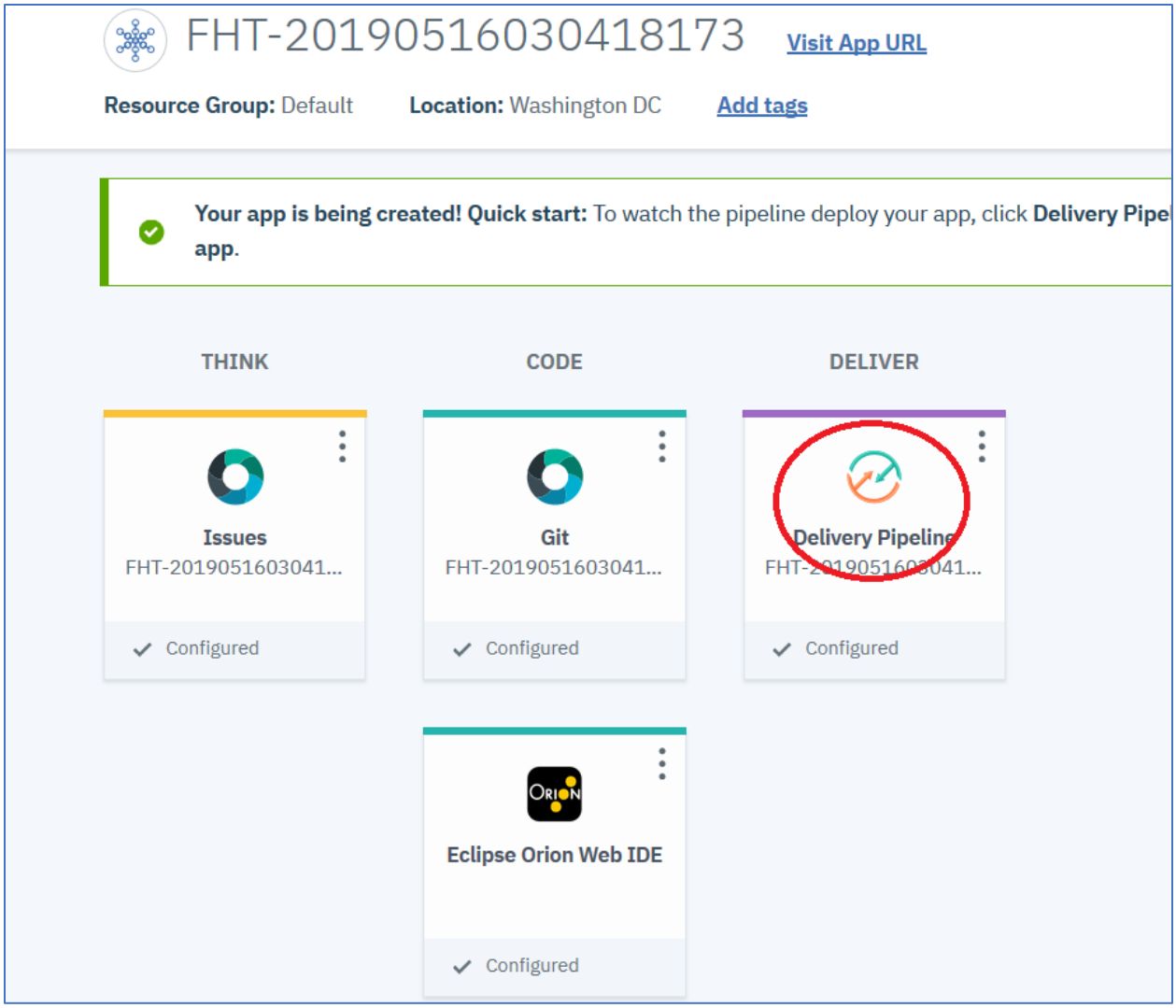
19. The next step is to commit the change to the git repository. Enter "Changed the Scoring Endpoint" in the Enter Commit Message field, and then click on **Commit**.



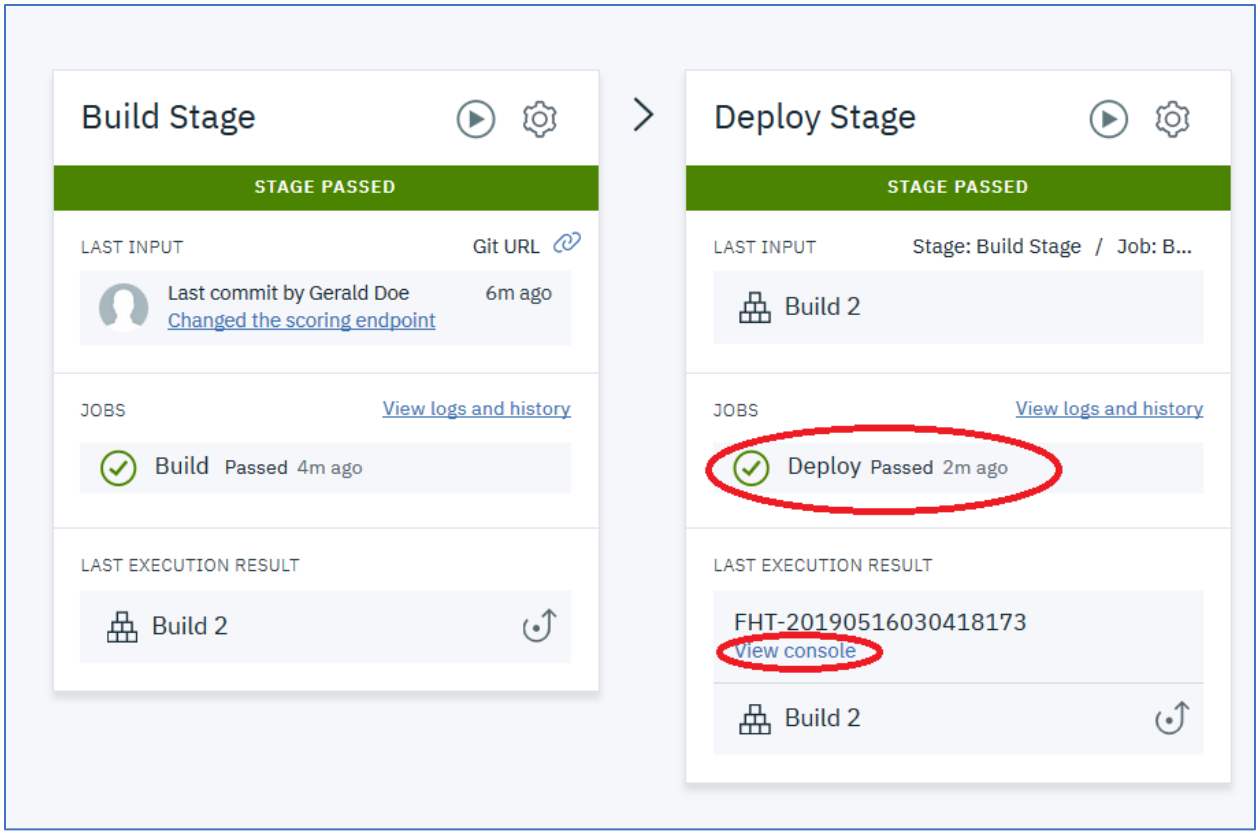
20. Then click on **Push** to push the changes to the central Git repo which will start the build and deploy of the application. Click on the left arrow to return to the Toolchain.



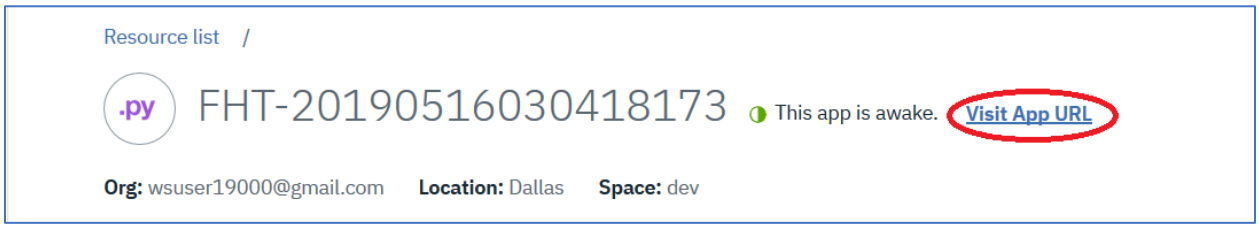
21. Click on the **Delivery Pipeline** to view status of the deployment as before.



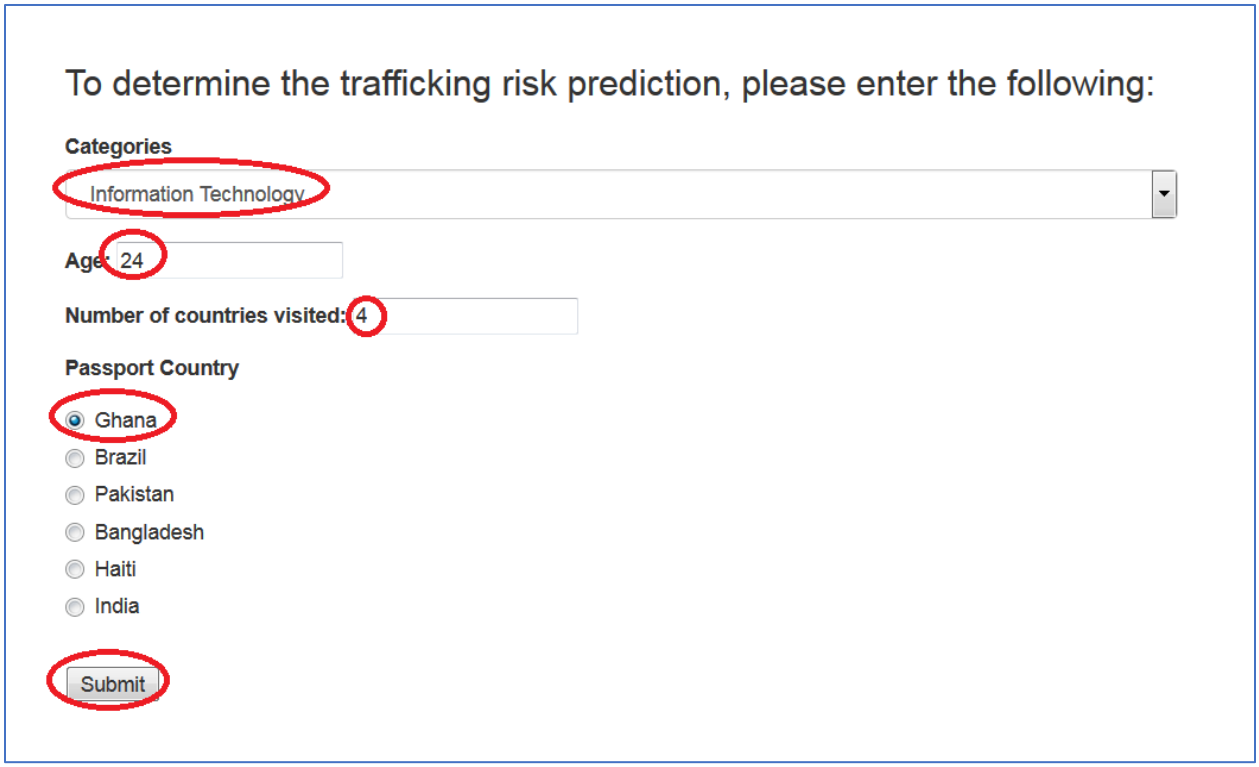
22. Once the Deployment status shows **Deploy passed now** it shouldn't take longer than 2 minutes (reload the browser in case the UI didn't update after 2 minutes). Click on **View Console.**



23. Click on **Visit App URL**.



24. The web form should appear. Enter data in all the fields and click on the **Submit** button.



25. You should see something similar to the following depending on the values of the input fields that you entered. Click on the **Try Again!**, if you want to experiment with different inputs.