AI on IBM Z

Fraud detection solution template

This solution template provides an example on how to deploy AI using an IBM Z environment, while making use of open source frameworks, Machine Learning for IBM z/OS (MLz), and more.

Within this solution template, there are various phases of the AI lifecycle included. Work through each of the following steps to deploy your own fraud detection solution on IBM Z.

Table of contents

Step 1

AI model training

We will build a fraud detection AI model by training with the provided rapid AI on IBM Z development Jupyter notebook. Simply point the Jupyter notebook to your dataset and run it to generate your AI model. This trained AI model can then be deployed with MLz.

All sample code for this section is within

`aionz-st-fraud-detection/zST-model-training-jupyter`

Prerequisites

- Must have Python (3.9 or 3.10) installed

Dataset guidance

Sample open source credit card transaction dataset can be found on Kaggle -

https://www.kaggle.com/datasets/ealtman2019/credit-card-transactions

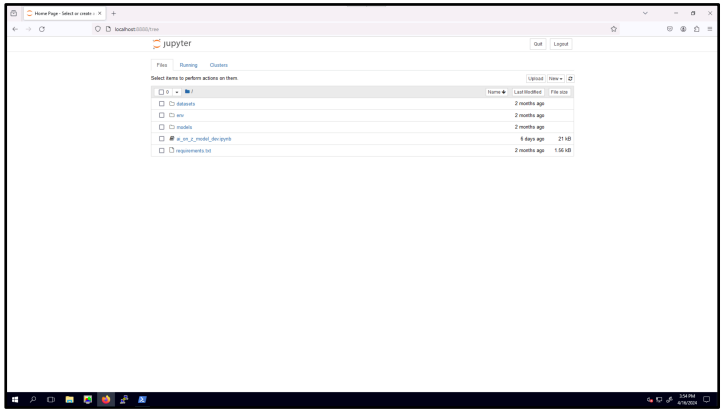
There are several files included within the download. You can use credit_card_transactions-ibm_v2.csv for training. Due to the size of the sample dataset, the provided Jupyter notebook takes a subset of the data to decrease the training time. Please modify the code in the “Fetch and process data” cell of the provided Jupyter notebook later to use more data during training.

Required features

- User (integer) – unique ID for user making transaction
- Card (integer) – unique ID for credit card
- Year (integer) – year of the transaction
- Month (integer) – month of the transaction
- Day (integer) – day of the month of the transaction
- Time (integer) - time of the transaction (HH:MM)
- Amount (float) – dollar amount of the transaction
- Use Chip (string) – the type of transaction
- Merchant Name (integer) – unique ID for merchant name
- Zip (integer) – zip code of the transaction

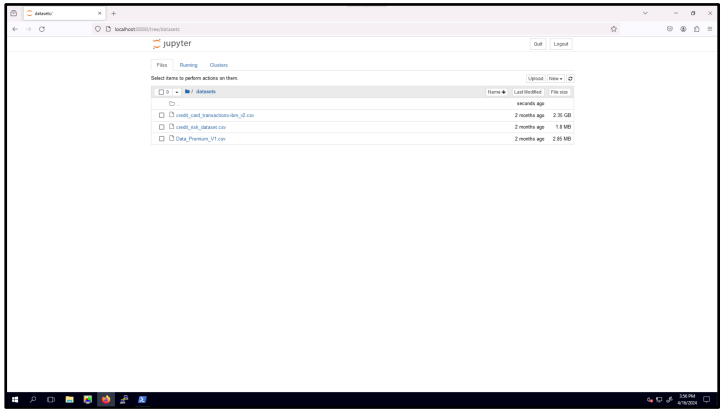
Access rapid AI on IBM Z development environment

1. Access sample code
   ```
   cd zST-model-training-jupyter
   ```
2. Create and activate Python virtual environment
   ```
   python -m venv env
   source env/bin/activate
   ```
3. Install required Python packages
   ```
   pip install -r requirements.txt
   ```
4. Run Jupyter
   ```
   jupyter notebook
   ```
5. View Jupyter interface
   a. Go to localhost:8888 in a web browser



Provide data

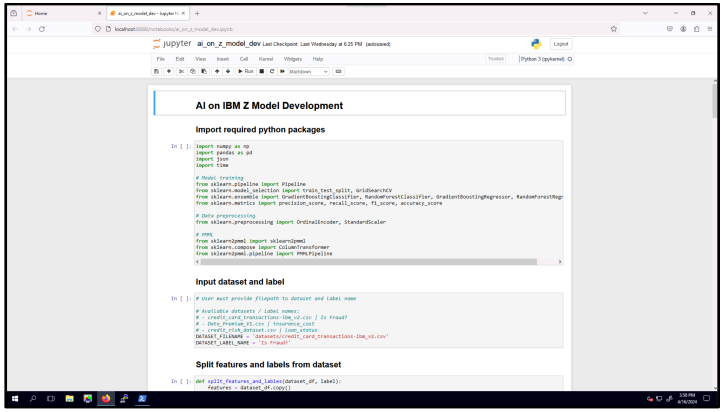
1. Your input dataset (csv) in `datasets/` directory
2. Add input data to Jupyter notebook (ai_on_z_model_dev.ipynb)
    a. Set `DATASET_FILENAME` to the path to your dataset
    b. Set `DATASET_LABEL_NAME` to the name of the column you're predicting from the dataset



Model training

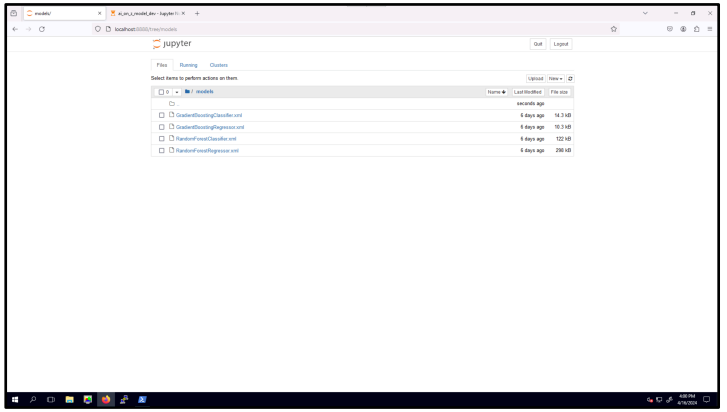
1. Step through and run all cells within Jupyter notebook (ai_on_z_model_dev.ipynb) within web browser

    Note: This may take several minutes



Access trained AI model

1. Once training is complete, you can find your AI models within the models/ directory (choose one for the following AI model deployment step)



1. AI model training. 2. AI model deployment 3. AI model analysis 4. AI model integration

AI model training complete

Prerequisites

- Must have MLz installed

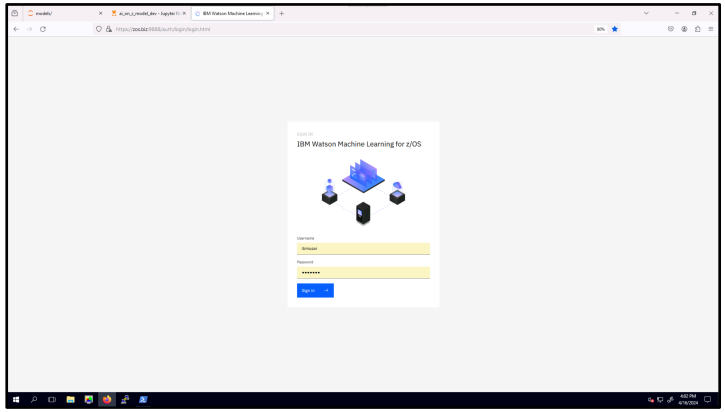
Step 2

AI model deployment

We will deploy our fraud detection AI model using MLz. We can utilize the model import functionality on the MLz UI. This deployed AI model can then be integrated into applications within the IBM Z environment.

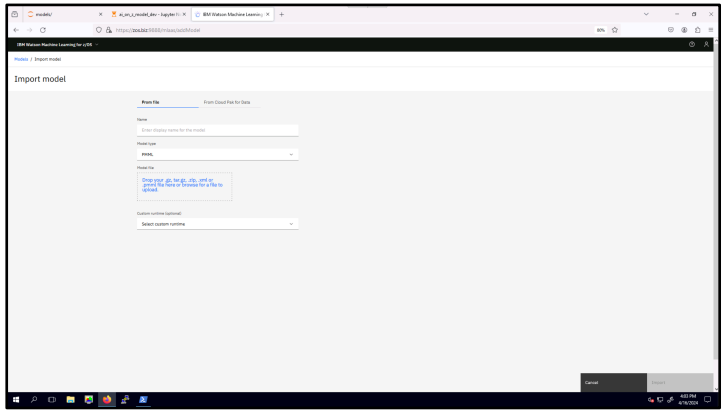
Go to MLz UI

1. Sign in with username/password

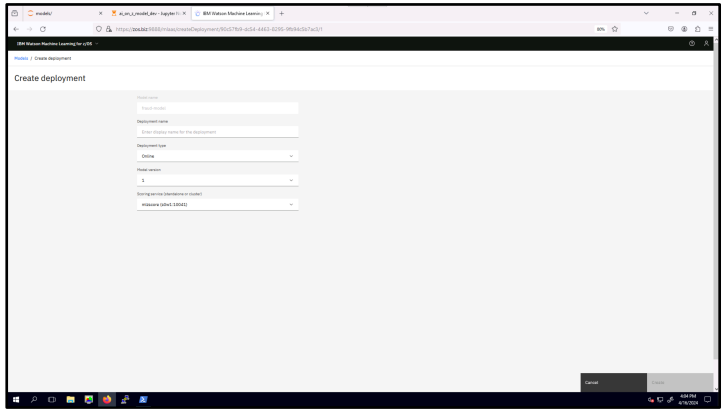


Import AI model

1. Go to models tab
2. Click import model
3. Enter model name
4. Choose model type
    a. Choose PMML if using your previously trained model
5. Drag and drop model file
   Use your previously trained model
6. Click import

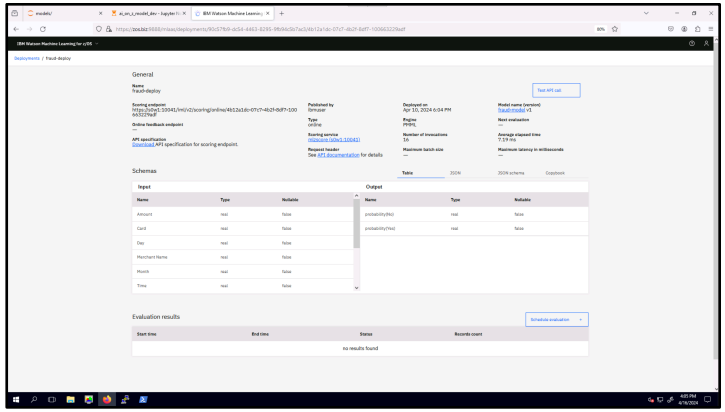


Deploy AI model



1. Go to models tab
2. Click action button for your model (on right side)
3. Click deploy
4. Enter deployment name
5. Choose deployment type
6. Choose model version
7. Choose scoring service
   Note: you should choose the correct scoring service based on your application (e.g. CICS or REST)
8. Click create

View deployed AI model



1. Go to deployments tab
2. Click on action button for your deployed model (on right side)
3. Click view details

☑ AI model deployment complete

Prerequisites

- Must have MLz installed
- Must have Python installed
- Must have Git installed
- Must have Docker or Podman installed

Step 3

AI model analysis

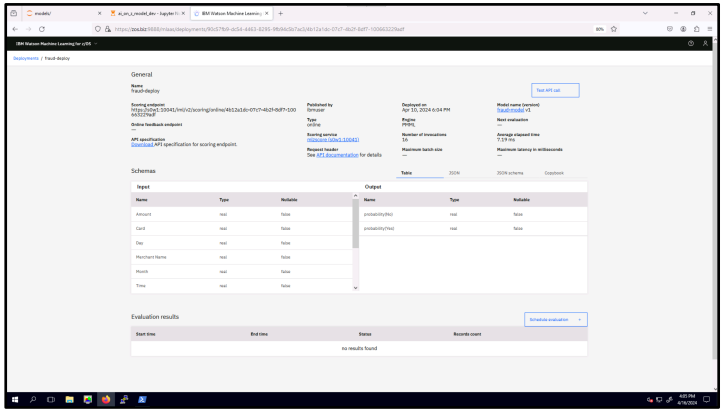
We will analyze our fraud detection AI model using a sample AI on IBM Z Fraud Detection Dashboard. We can invoke the API of this sample dashboard from another sample application to visualize the AI model inferencing.

All sample code for this section is within

`aionz-st-fraud-detection/zST-model-analysis`

Get model details for inferencing

1. Go to MLz UI
2. Go to deployments tab
3. Click on action button for your deployed model (on right side)
4. Click view details
5. Copy scoring endpoint

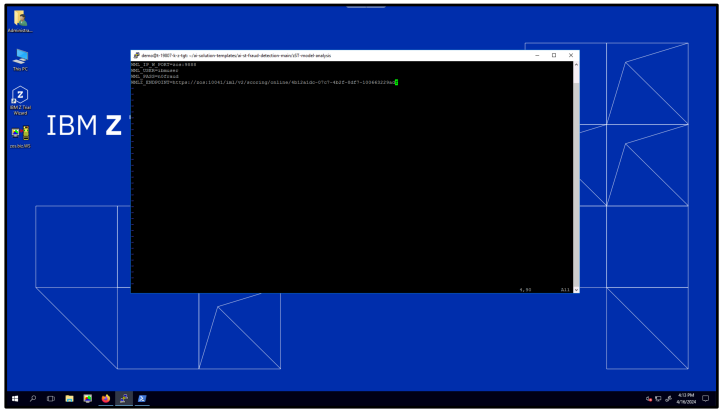


Configure sample application

1. Set the environment variables within

   `aionz-st-fraud-detection/zST-model-analysis/env.list`

   WML_IP_W_PORT (IP address of MLz with port)
   WML_USER (MLz username)
   WML_PASS (MLz password)
   WMLZ_ENDPOINT (scoring endpoint for deployed AI model)



Build sample application

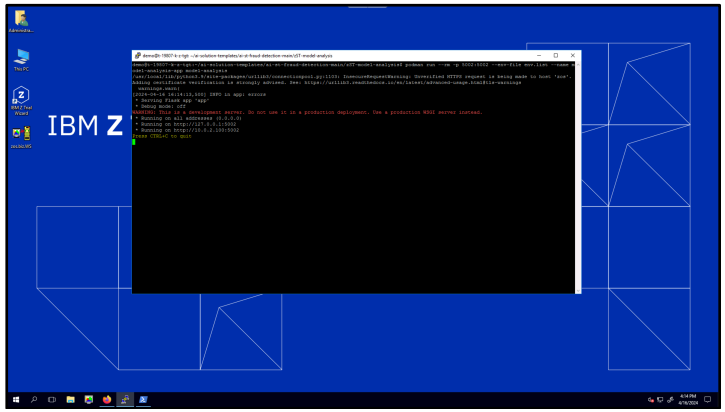
1. Run command in terminal

```
podman build -t model-analysis
```

Deploy sample application

1. Run command in terminal (e.g. port 5002)

```
podman run --rm -p 5002:5002 --env-file
env.list --name model-analysis-app
model-analysis
```



Access sample application

1. View the following URL in a web browser http://{IP address}:{port}
    a. IP address: IP where app is deployed
    b. Port: port you used with podman run

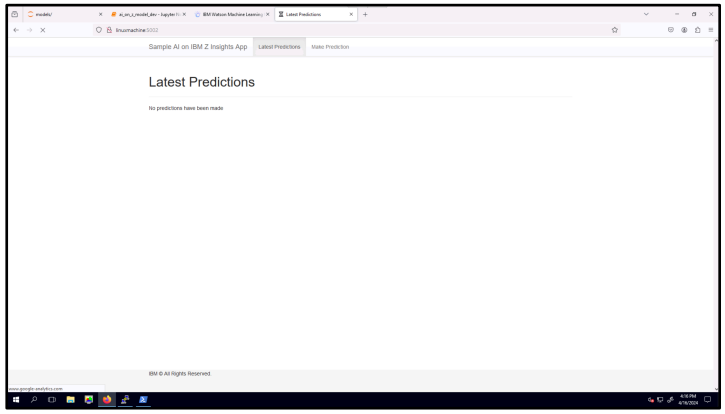
Analyze credit card events

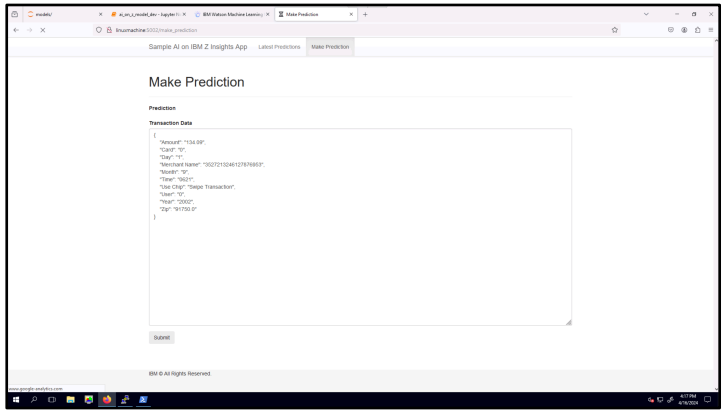
1. Go to latest predictions tab

Make predication

1. Go to latest make predication tab
2. Input json data in following format:
    a.
```
{"Amount": "134.09", "Card":
"0","Day": "1", "Merchant Name":
"3527213246127876953", "Month":
"9", "Time": "0621","Use Chip":
"Swipe Transaction", "User": "0",
"Year": "2002", "Zip": "91750.0"}
```

3. Click submit





☑ AI model analysis complete

Step 4

AI model integration

Choose One:

Ecommerce web application

CICS-COBOL application

We can use our deployed MLz fraud detection AI model and integrate it into different types of applications. The AI model can be analyzed and/or provide inferencing APIs using the sample AI on IBM Z Fraud Detection Dashboard.

Ecommerce web application

Prerequisites

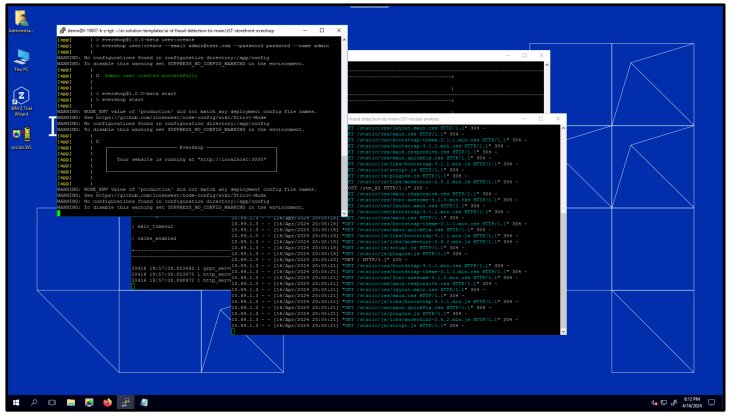
- Must have AI on IBM Z Sample Fraud Detection Dashboard deployed for inferencing and analysis
- Must have Podman or Docker installed

The sample e-commerce application is based on the open source EverShop Storefront and has been extended to integrate with the AI on IBM Z Solution Template. EverShop is a GraphQL Based and React ecommerce platform with essential commerce features. Built with React, modular and fully customizable.

All sample code for this section is within

`aionz-st-fraud-detection/zST-storefront-evershop`

Install and deploy sample ecommerce application



1. Set app_url_w_port variable in CheckoutForm.jsx to your server IP and port (ip:port)

   ```
   ai-st-fraud-detection-tis/zST-storefronteversh
   op/packages/evershop/src/components/frontStor
   e/stripe/checkout/CheckoutForm.jsx
   ```

2. Run command in terminal (password: demo)

   ```
   podman-compose up
   ```

Configure sample ecommerce application

Access admin panel from web browser

1. Enter URL in web browser using app url (e.g. localhost)

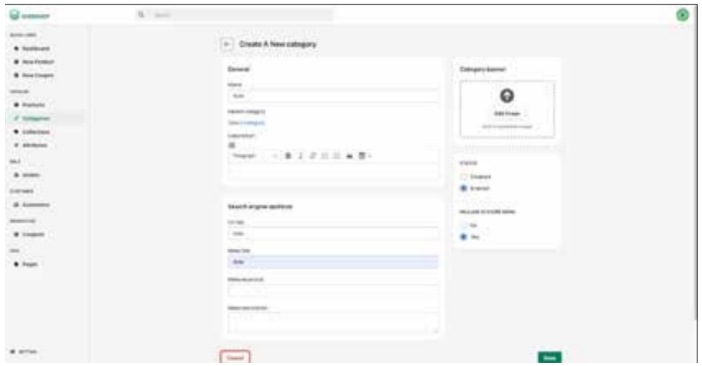
   http://localhost:3000/admin

2. Login with default admin credentials

   Email: admin@test.com
   Password: password

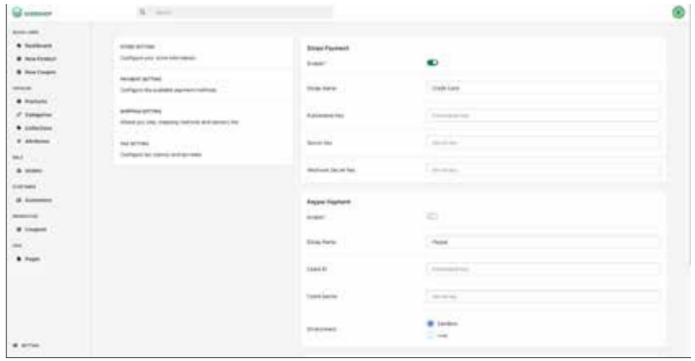
Add products

1. Create categories
    a. Click categories from the catalog section
    b. Click new category
    c. Add category details
        i. Add name (e.g. Kids)
        ii. Add URL key
        iii. Add meta title
        iv. Change status to enable
        v. Change include in store menu to yes
    d. Click save
2. Add products
    a. Click products from the catalog section
    b. Click new product
    c. Add category details
        i. Make sure to change add category
        ii. Make sure to change status to enable
        iii. Make sure to change visibility to visible



Configure store settings

1. Enter URL in web browser using app URL (e.g. localhost)
2. Click settings on the bottom left
3. Click store setting

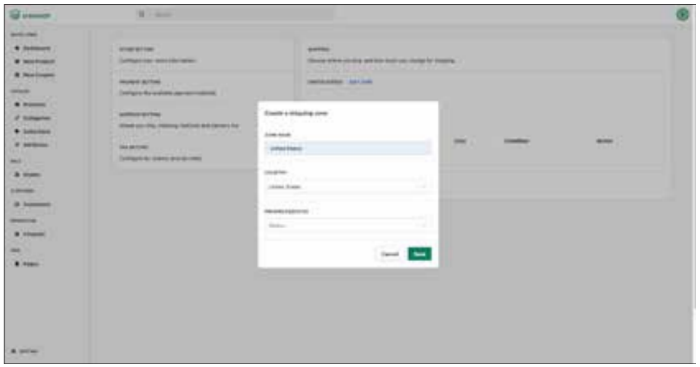


Configure payment settings

1. Click payment setting
2. Enable stripe payment
3. Click save

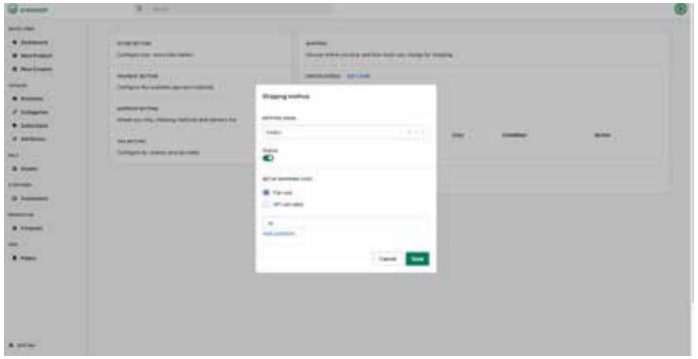
Configure shipping settings

1. Click shipping setting
2. Add shipping zone
    a. Click create new shipping zone
    b. Add shipping details
3. Click save



Add payment methods

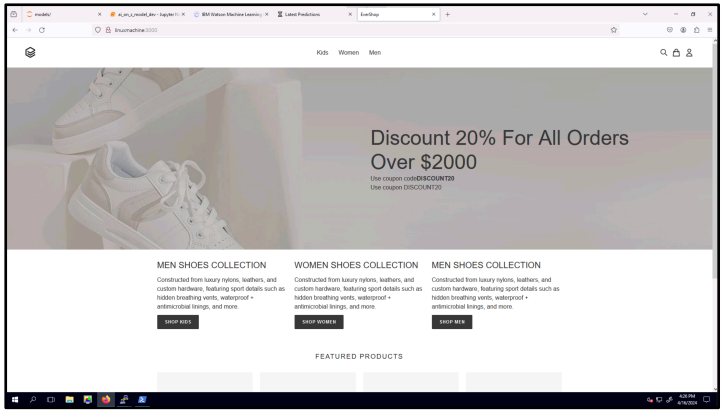
1. Click add method
2. Add method name (e.g. FedEx)
3. Enable status
4. Add flat rate (e.g. 10)
5. Click save



Access sample ecommerce application

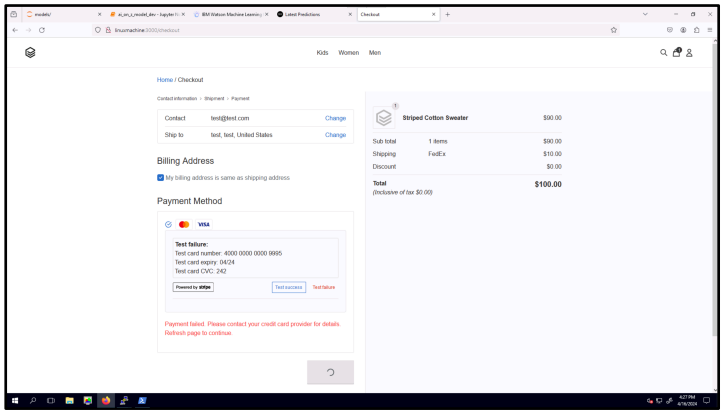
1. Enter URL in web browser using app URL (e.g. localhost)

   http://localhost:3000



Use fraud detection AI model with EverShop Storefront

1. Make sure you have AI on Linux on Z Sample Fraud Detection Dashboard deployed for inferencing and analysis on the same local system
2. AI on Linux on Z Sample Fraud Detection Dashboard is configured to invoke MLz AI model
3. Add items to cart
4. Place order
   a. Choose test failure as payment method for fraud transaction example
      Choose test success as payment method for non-fraud transaction example



☑ AI model integration complete (web)

CICS-COBOL application

We can use our deployed MLz fraud detection AI model and integrate it into different types of applications. Guidance on integrating the AI model into a sample CICS-COBOL application is below.

All sample code for this section is within

`aionz-st-fraud-detection/zST-model-integration-CICS`

Prerequisites

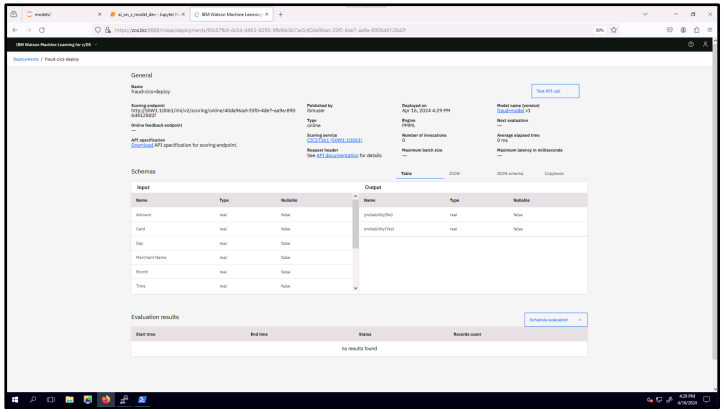
- Must have access to z/OS CICS environment
- Must have MLz installed
- Must have model deployed with the CICS-scoring server as scoring service
- Must have the JVM server and bundle setup and running. To verify the same execute the below commands in the CICS

  JMVSERVER
  `CEMT INQUIRE JVMSERVER(ALNSCSER)`

  BUNDLE
  `CEMT INQUIRE BUNDLE(ALNSCBDL)`

Get model details for inferencing

1. Go to MLz UI
2. Go to deployments tab
3. Click on action button for your deployed model (on right side)
4. Click view details
   Copy scoring endpoint



Generate helper classes

1. Review the input and output copybooks from the MLz UI
2. Access z/OS terminal
3. Run the

   `$IML_INSTALL_ENT_DIR/cics-scoring/bin/gen_helper_class.sh`

   script to generate helper classes within z/OS terminal

   a. SERVER_NAME is the name of the scoring service the model is deployed to.

   b. DEPLOYMENT_ID is the deployment ID of the model.

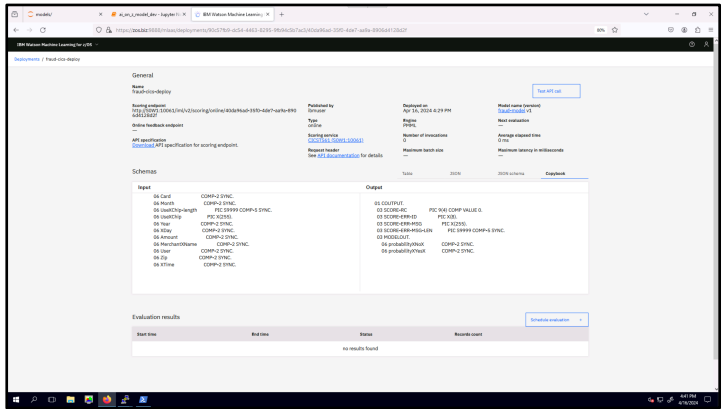
   c. JCLASS_PREFIX is the prefix to be used for the generated class.

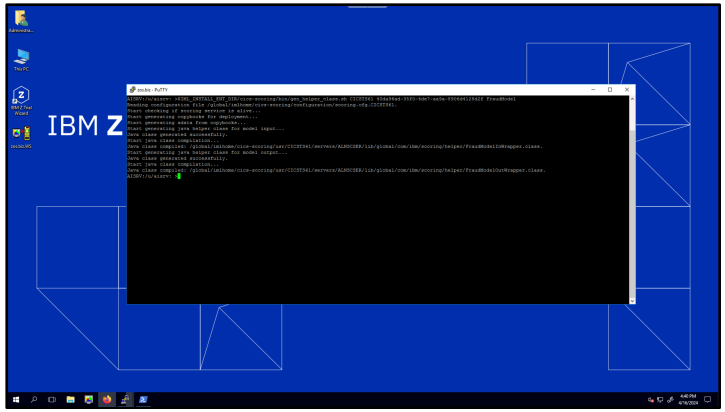
   d. The generated input class will be named as: <JCLASS_PREFIX>InWrapper.class

   e. The generated output class will be named as: <JCLASS_PREFIX>OutWrapper.class

   Example:

   ```
   $IML_INSTALL_ENT_DIR/cics-scoring/bin/gen_helper_class.sh CICSTS61 40da96ad-35f0-4de7-aa9a-8906d4128d2f FraudModel
   ```



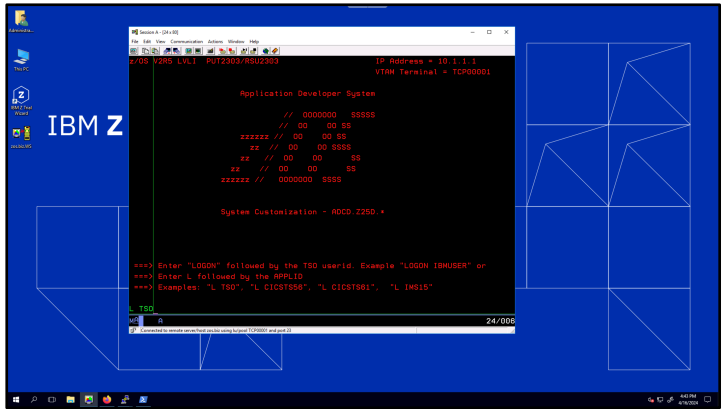


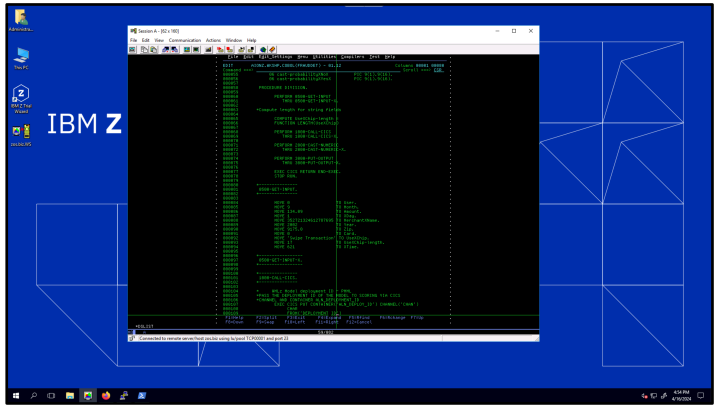
Create COBOL program

1. Access TSO
2. Create ALNJCGEN JCL
3. Copy provided sample COBOL program

   `aionz-st-fraud-detection/zST-modelintegration-CICS/FRAUDDET.cob`

4. Paste sample COBOL program into FRAUDDET
5. Copy deployment ID from MLz UI
6. Replace DEPLOYMENT_ID in sample COBOL program with your deployment ID
7. Update the names of the input and output wrapper classes created in the previous



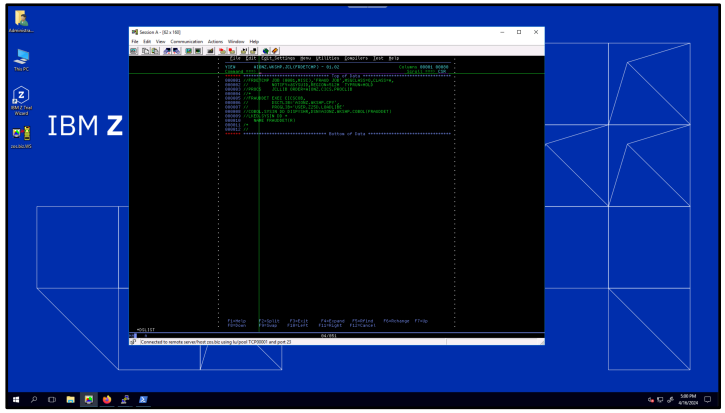


Compile and link-edit CICS-COBOL program

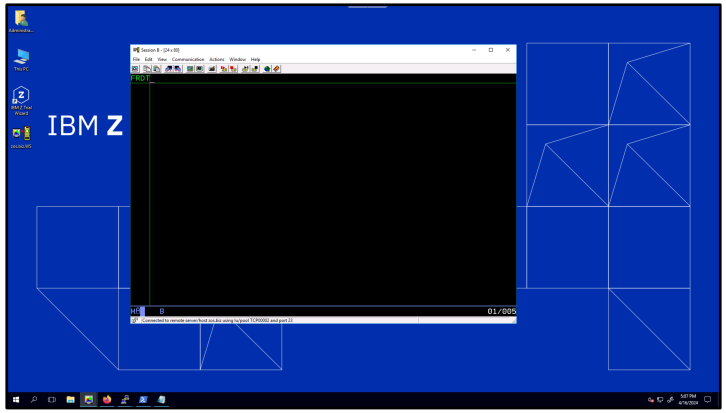
1. Access TSO
2. Create COMPILE file
3. Copy provided sample COMPILE file

   aionz-st-fraud-detection/zST-modelintegration-CICS/COMPILE.jcl

4. Paste sample COMPILE file into COMPILE file
5. Submit job
6. Verify job ran successfully (MAXCC = 0)



Run CICS transaction with AI inferencing



1. Access CICS
2. Run command to define transaction

```
CEDA DEFINE TRANS(FRDT) GROUP(ALNSCGRP)
PROGRAM(FRAUDDET) DESCRIPTION(TRAN to
execute FRAUDDET)
```

3. Run command to define program

```
CEDA DEFINE PROGRAM(FRAUDDET)
GROUP(ALNSCGRP) LANGUAGE(COBOL)
DESCRIPTION(FRAUD DETECTION)
```

4. Run command to install transaction

```
CEDA INS TRANS(FRDT) GROUP(ALNSCGRP)
```

5. Run command to install program

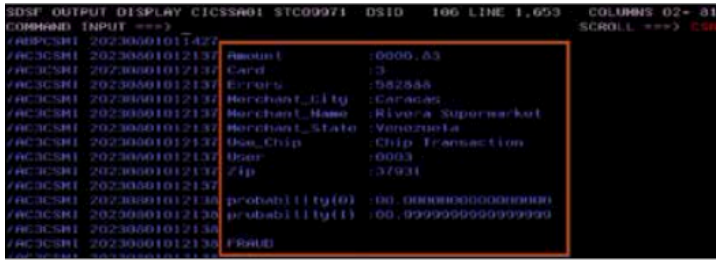
```
CEDA INS PROGRAM(FRAUDDET) GROUP
(ALNSCGRP)
```

6. Run transaction

```
FRDT
```

View output results

1. Go to the Spool of the started task for CICS (e.g. CICSTS61)
2. Navigate to the CEEMSG dataset
3. View displays from the module in this dataset
4. Sample output is shown as below



☑ AI model integration complete (CICS)