Sprint 3 – A Gesture Based Tool for Sterile Browsing of Radiology Images

Team ID: PNT2022TMID53060

Team Members: A Anirudh, Karthik D, Nestor Ingarshal J, Pugalarasu K

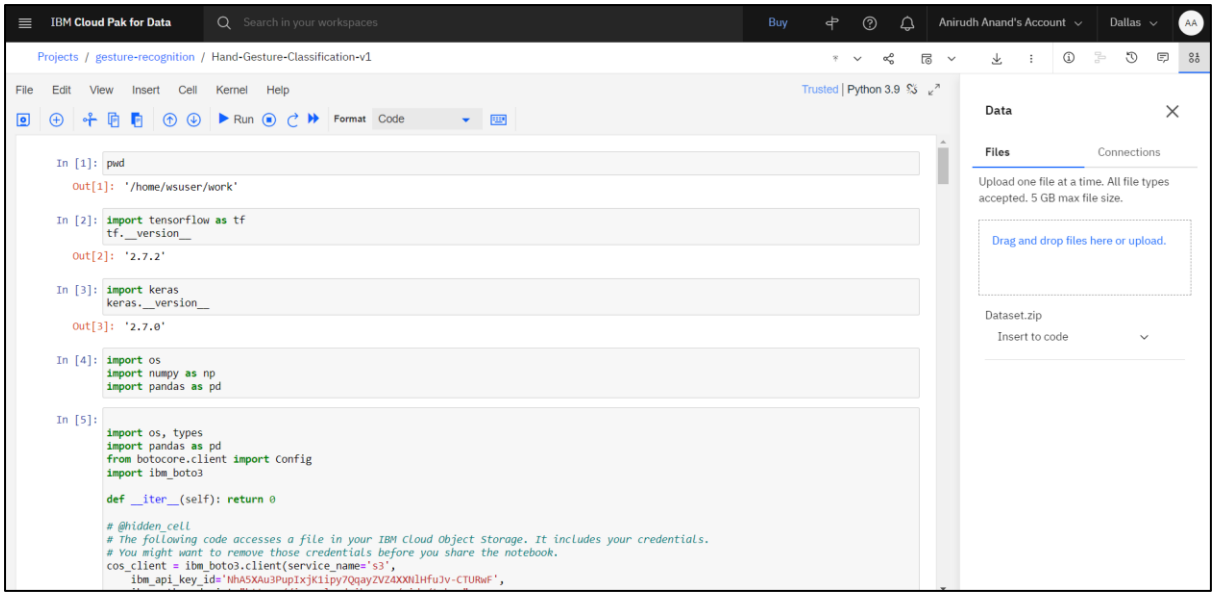
Code: Updated on GitHub in the Sprint-3 folder under Project Development Phase.

Description of USN and Screenshots:

USN-8:

As a user, I need the deep learning model to be accessible worldwide.

1. *Model Training on Watson Studio*



```
In [10]: train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
         test_datagen = ImageDataGenerator(rescale=1./255)

In [11]: x_train = train_datagen.flow_from_directory('/home/wsuser/work/Dataset/train/', target_size=(64, 64), batch_size=5, color_mode='grayscale'
         x_test = test_datagen.flow_from_directory('/home/wsuser/work/Dataset/test/', target_size=(64, 64), batch_size=5, color_mode='grayscale'

         Found 594 images belonging to 6 classes.
         Found 30 images belonging to 6 classes.

In [12]: x_train.class_indices

Out[12]: {'0': 0, '1': 1, '2': 2, '3': 3, '4': 4, '5': 5}

In [13]: classifier = Sequential()

In [14]: classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 1) ,activation='relu'))
         classifier.add(MaxPooling2D(pool_size=(2, 2)))
         classifier.add(Conv2D(32, (3, 3), activation='relu'))
         classifier.add(MaxPooling2D(pool_size=(2, 2)))
         classifier.add(Flatten())

In [15]: classifier.add(Dense(units=128, activation='relu'))
         classifier.add(Dense(units=6, activation='softmax'))

In [16]: classifier.summary()

         Model: "sequential"
```

Data

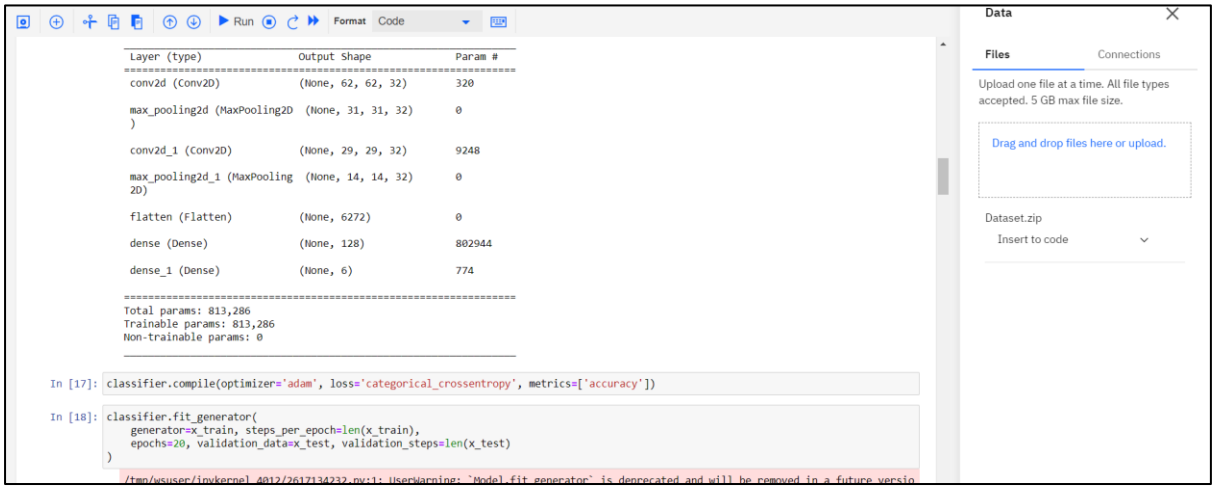
Files | Connections

Upload one file at a time. All file types accepted. 5 GB max file size.

Drag and drop files here or upload.

Dataset.zip

Insert to code



```
 Layer (type)                 Output Shape              Param #
=================================================================
 conv2d (Conv2D)              (None, 62, 62, 32)        320

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)        0
 )

 conv2d_1 (Conv2D)            (None, 29, 29, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)        0
 2D)

 flatten (Flatten)            (None, 6272)              0

 dense (Dense)                (None, 128)               802944

 dense_1 (Dense)              (None, 6)                 774

=================================================================
Total params: 813,286
Trainable params: 813,286
Non-trainable params: 0
_________________________________________________________________

In [17]: classifier.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

In [18]: classifier.fit_generator(
             generator=x_train, steps_per_epoch=len(x_train),
             epochs=20, validation_data=x_test, validation_steps=len(x_test)
         )

         /tmp/wsuser/ipykernel_4012/2617134232.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future versio
```

Data

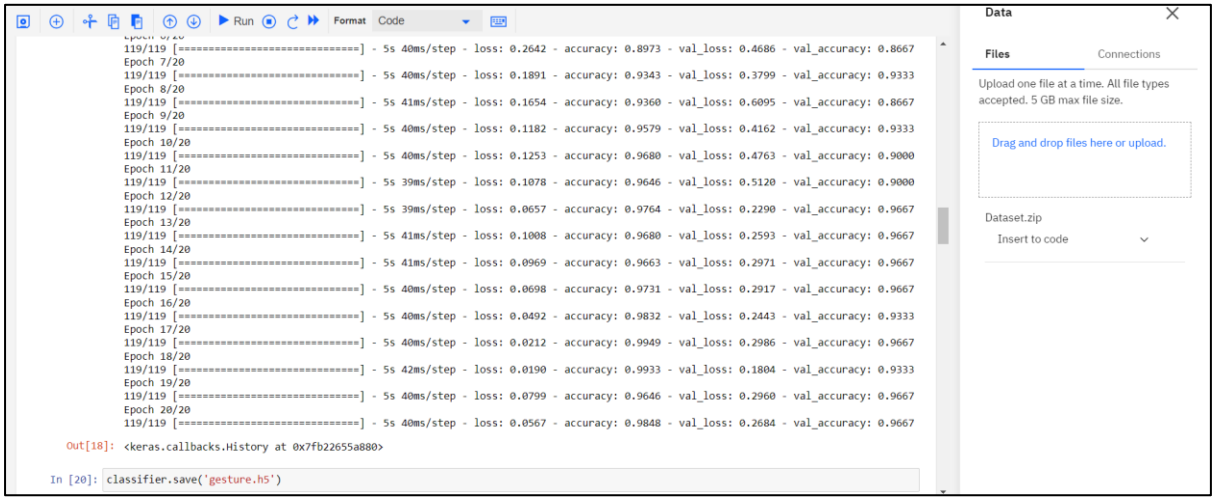
Files | Connections

Upload one file at a time. All file types accepted. 5 GB max file size.

Drag and drop files here or upload.

Dataset.zip

Insert to code



```
119/119 [==============================] - 5s 40ms/step - loss: 0.2642 - accuracy: 0.8973 - val_loss: 0.4686 - val_accuracy: 0.8667
Epoch 7/20
119/119 [==============================] - 5s 40ms/step - loss: 0.1891 - accuracy: 0.9343 - val_loss: 0.3799 - val_accuracy: 0.9333
Epoch 8/20
119/119 [==============================] - 5s 41ms/step - loss: 0.1654 - accuracy: 0.9360 - val_loss: 0.6095 - val_accuracy: 0.8667
Epoch 9/20
119/119 [==============================] - 5s 40ms/step - loss: 0.1182 - accuracy: 0.9579 - val_loss: 0.4162 - val_accuracy: 0.9333
Epoch 10/20
119/119 [==============================] - 5s 40ms/step - loss: 0.1253 - accuracy: 0.9680 - val_loss: 0.4763 - val_accuracy: 0.9000
Epoch 11/20
119/119 [==============================] - 5s 39ms/step - loss: 0.1078 - accuracy: 0.9646 - val_loss: 0.5120 - val_accuracy: 0.9000
Epoch 12/20
119/119 [==============================] - 5s 39ms/step - loss: 0.0657 - accuracy: 0.9764 - val_loss: 0.2290 - val_accuracy: 0.9667
Epoch 13/20
119/119 [==============================] - 5s 41ms/step - loss: 0.1008 - accuracy: 0.9680 - val_loss: 0.2593 - val_accuracy: 0.9667
Epoch 14/20
119/119 [==============================] - 5s 41ms/step - loss: 0.0969 - accuracy: 0.9663 - val_loss: 0.2971 - val_accuracy: 0.9667
Epoch 15/20
119/119 [==============================] - 5s 40ms/step - loss: 0.0698 - accuracy: 0.9731 - val_loss: 0.2917 - val_accuracy: 0.9667
Epoch 16/20
119/119 [==============================] - 5s 40ms/step - loss: 0.0492 - accuracy: 0.9832 - val_loss: 0.2443 - val_accuracy: 0.9333
Epoch 17/20
119/119 [==============================] - 5s 40ms/step - loss: 0.0212 - accuracy: 0.9949 - val_loss: 0.2986 - val_accuracy: 0.9667
Epoch 18/20
119/119 [==============================] - 5s 42ms/step - loss: 0.0190 - accuracy: 0.9933 - val_loss: 0.1804 - val_accuracy: 0.9333
Epoch 19/20
119/119 [==============================] - 5s 40ms/step - loss: 0.0799 - accuracy: 0.9646 - val_loss: 0.2960 - val_accuracy: 0.9667
Epoch 20/20
119/119 [==============================] - 5s 40ms/step - loss: 0.0567 - accuracy: 0.9848 - val_loss: 0.2684 - val_accuracy: 0.9667

Out[18]: <keras.callbacks.History at 0x7fb22655a880>

In [20]: classifier.save('gesture.h5')
```

Data

Files | Connections

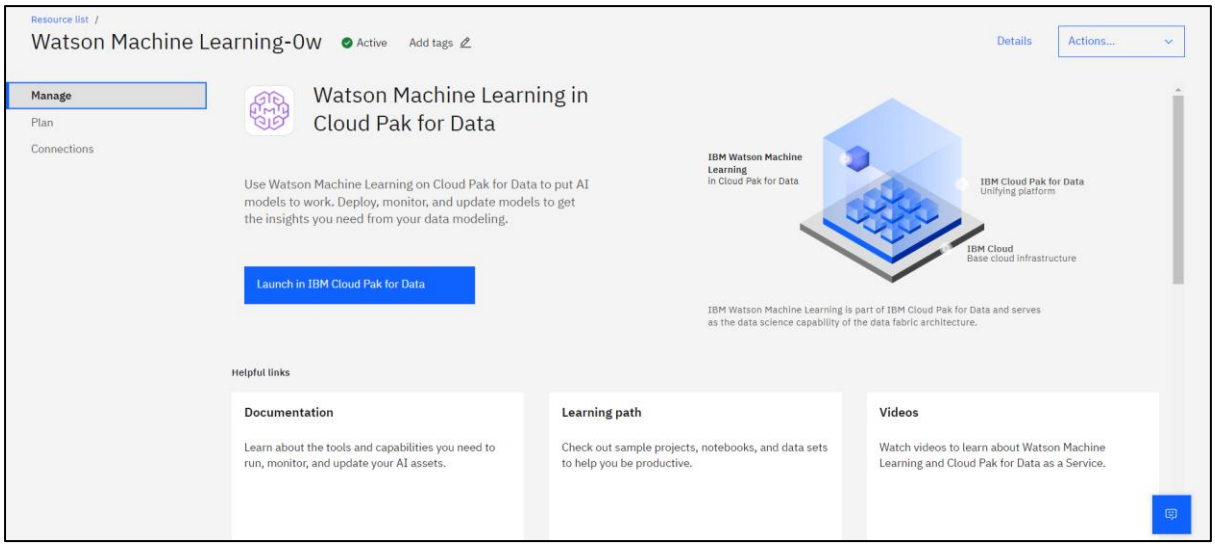
Upload one file at a time. All file types accepted. 5 GB max file size.

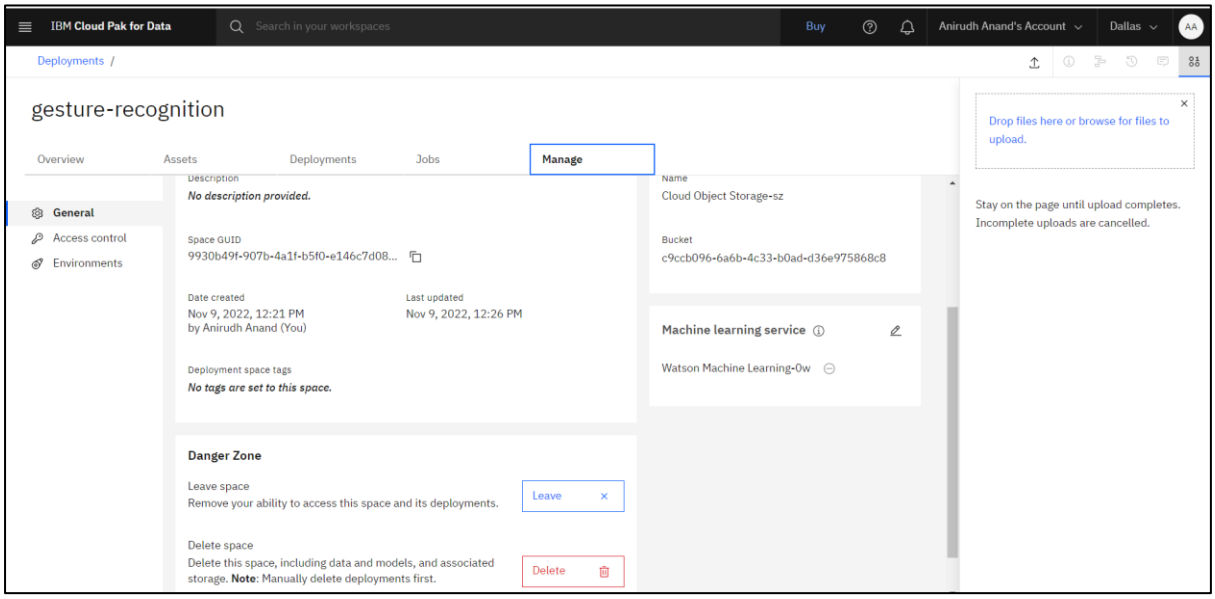
Drag and drop files here or upload.

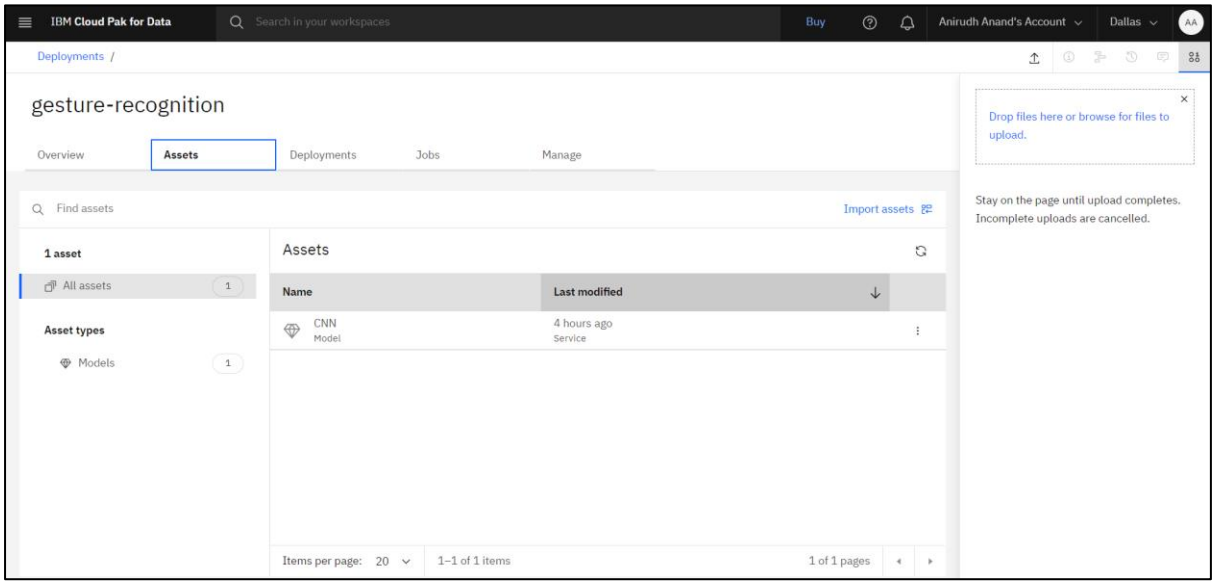
Dataset.zip

Insert to code

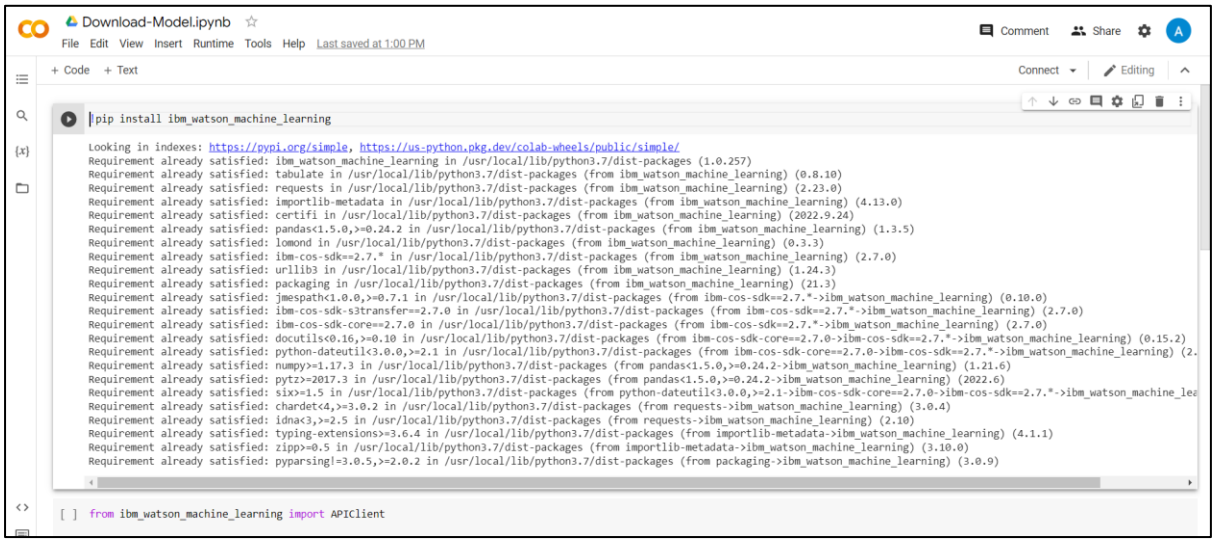
2. *Setting up Deployment Space using IBM's Machine Learning Service*







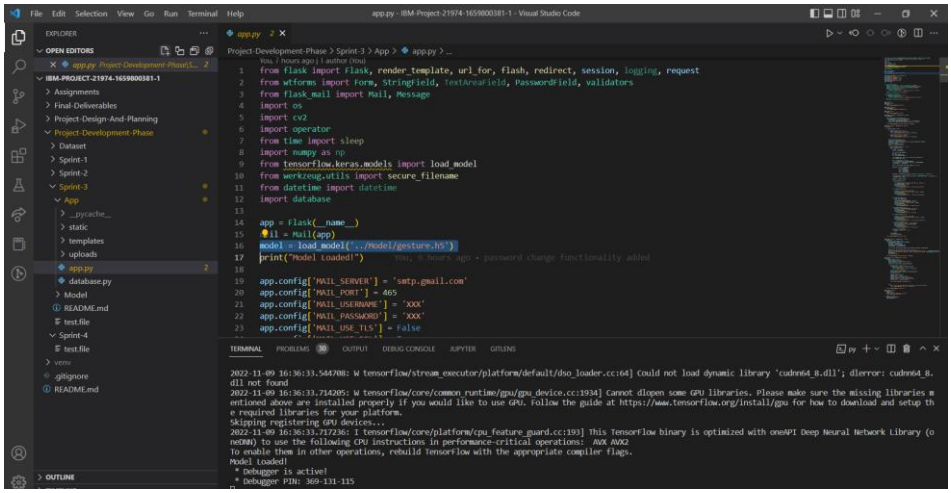
3. *Downloading the model to a local system*



USN-9:

As a user, I need the application to be accessible. The application should be able to plug-in different models with minimal modifications.

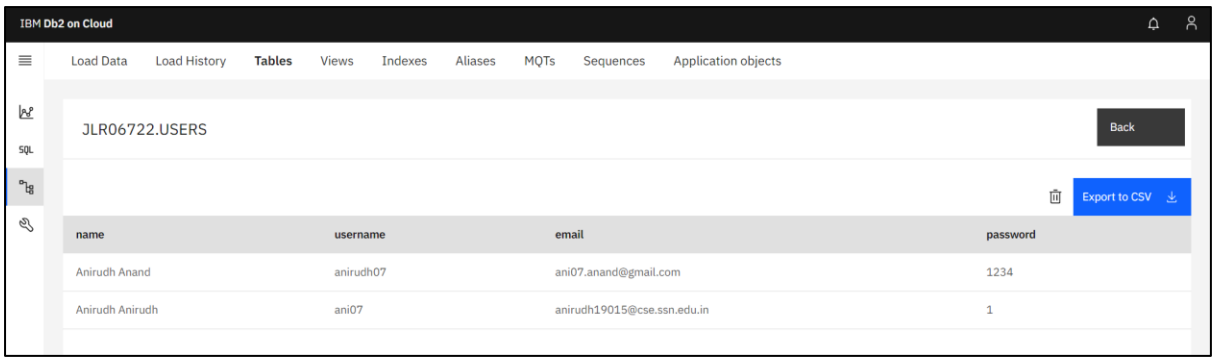
- The model trained on IBM Watson Studio is stored on the deployment space using IBM's Machine Learning Service.
- From here, the model can be downloaded (as a zip file) using the ibm_watson_machine_learning python package.
- The downloaded model can be extracted and loaded using Kera's load_model() function.



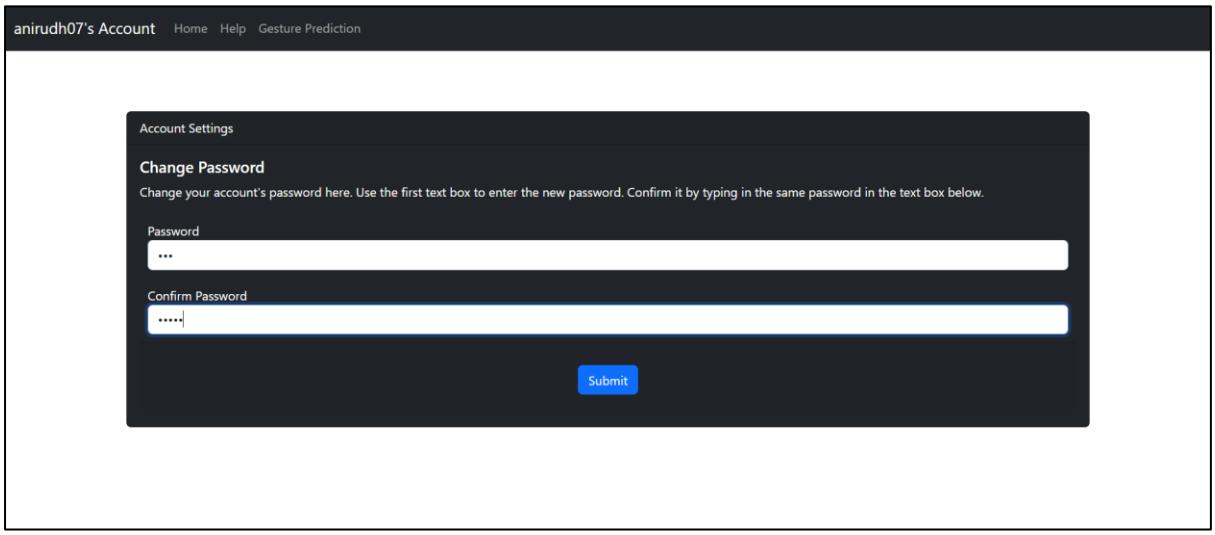
USN-10:

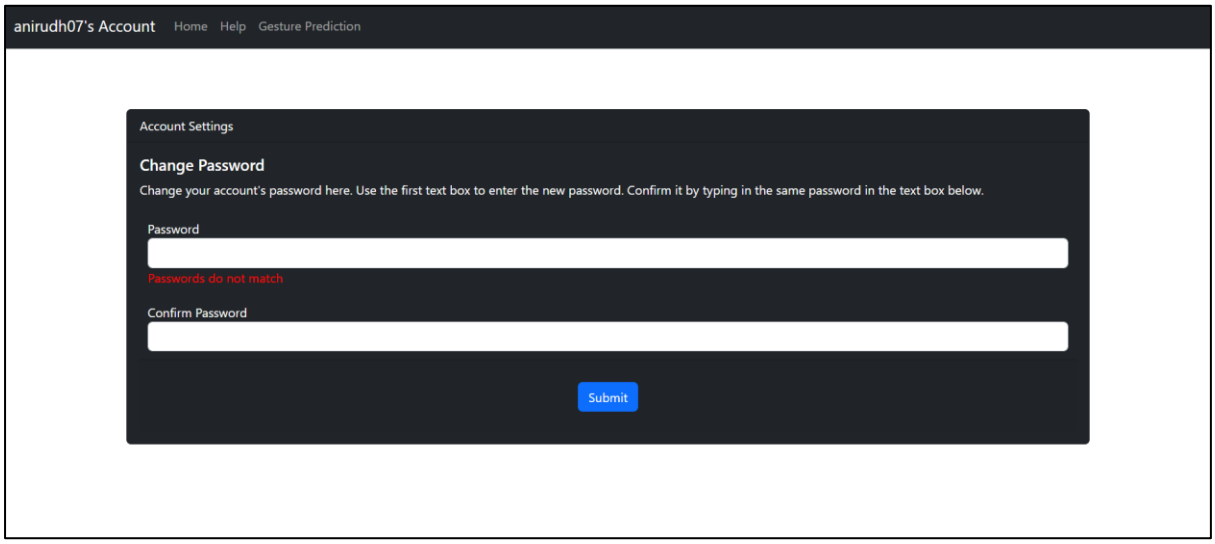
As a user, I should be able to access my dashboard to change my password.

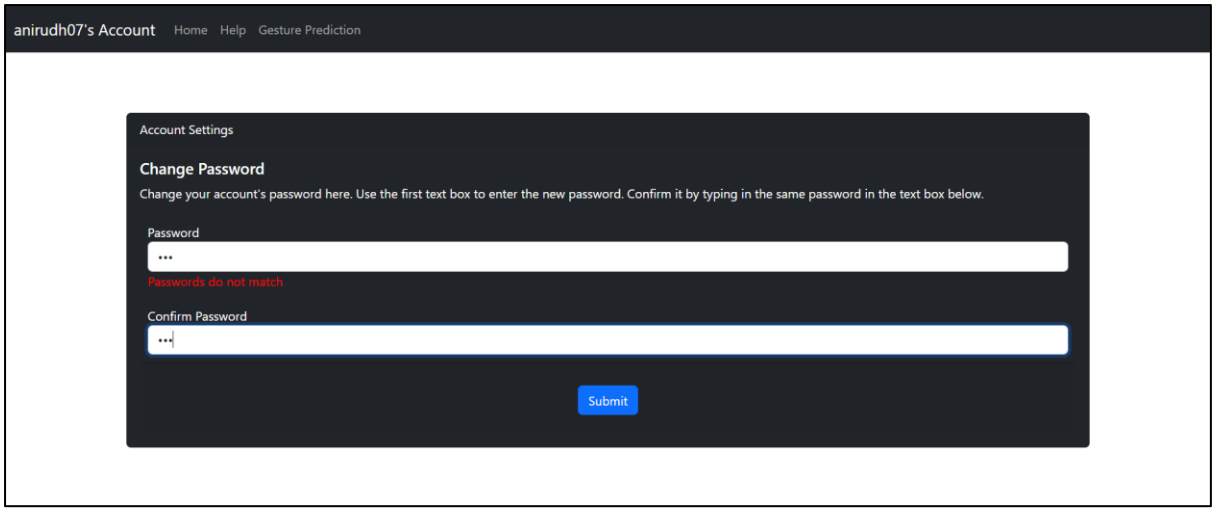
DB2 Database before password update

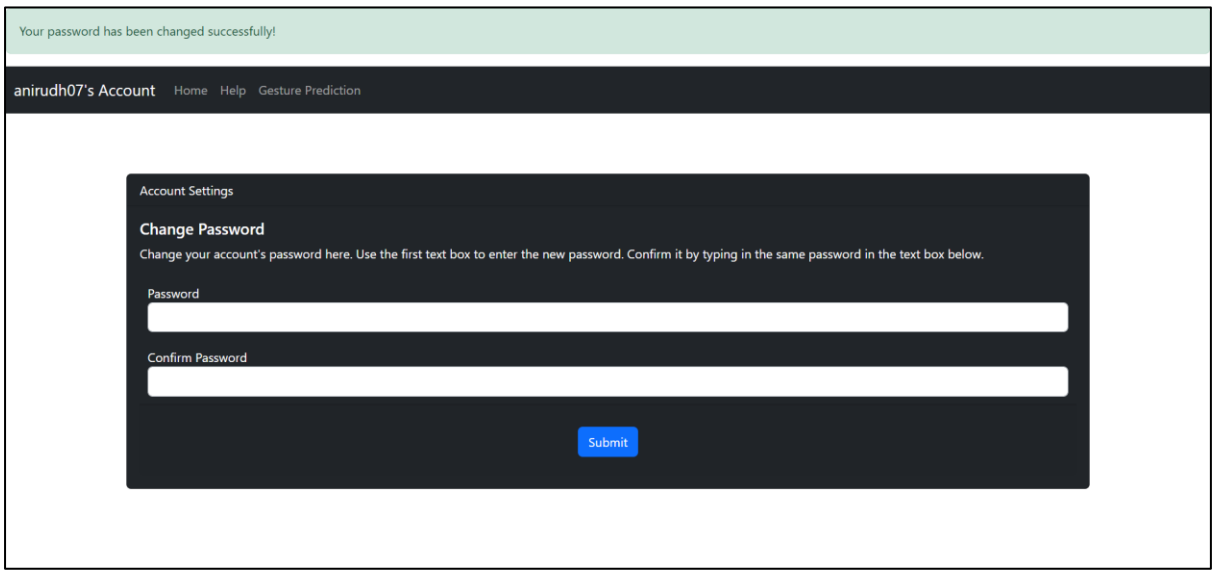


Password Update

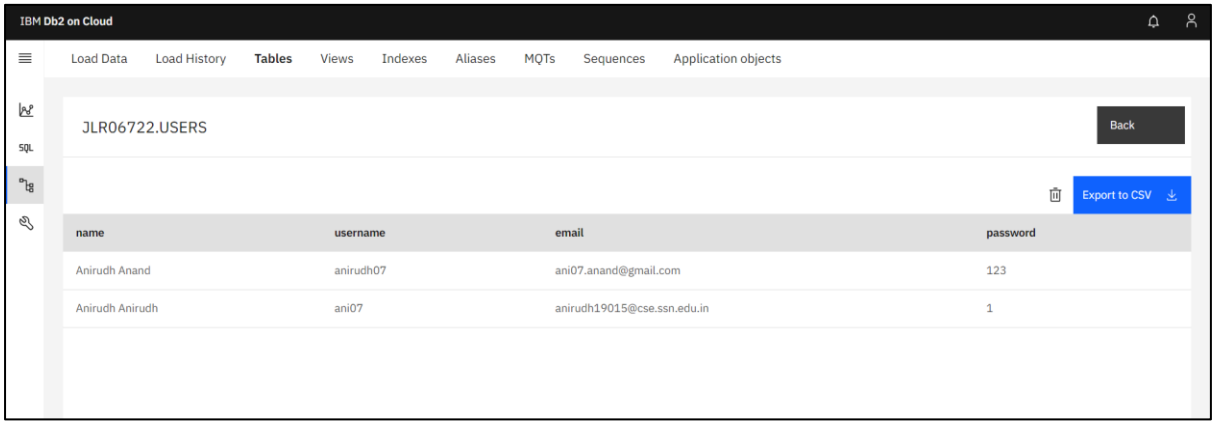








DB2 Database after password update



USN-11:

As a user, I should be alerted in case:

1. The application can’t access my webcam
2. The image I took/uploaded is corrupted