DEVOPS AND AUTOMATION FOR AI CA2

YOUR ONE-STOP AI VEGGIE PREDICTION WEBSITE :

VEGGIELENS

Name : Dario Prawara Teh Wei Rong

Class : DAAA / FT / 2B / 04

Admin No : 2201858

PROJECT OBJECTIVE

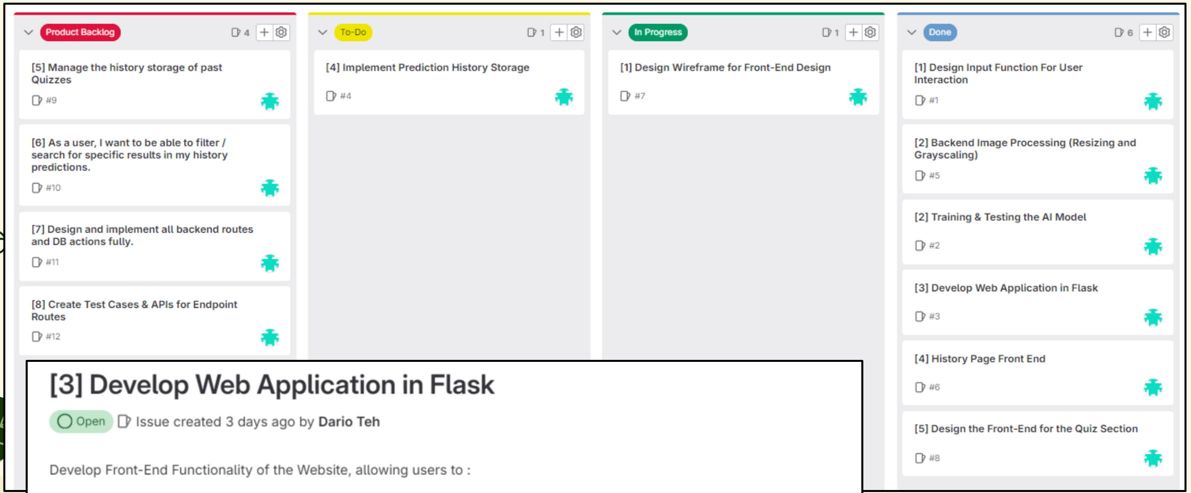
- Develop a Deep Learning web application, using GitLab and Flask Framework, incorporating the 3 DevOps best practices.

- To use an image dataset with **15 classes of vegetable images and two CNN models** to predict the class the images belong to.

GITLAB & SCRUM PROCESSES

- Labels for product backlog, to-do, in progress, and done to track web development.
- Created **user stories**, using user objectives to fulfil issues and project tasks along the way.

- Set up 6 branches for different **development** purposes.



Product Backlog 4

[5] Manage the history storage of past Quizzes
#9

[6] As a user, I want to be able to filter / search for specific results in my history predictions.
#10

[7] Design and implement all backend routes and DB actions fully.
#11

[8] Create Test Cases & APIs for Endpoint Routes
#12

To-Do 1

[4] Implement Prediction History Storage
#4

In Progress 1

[1] Design Wireframe for Front-End Design
#7

Done 6

[1] Design Input Function For User Interaction
#1

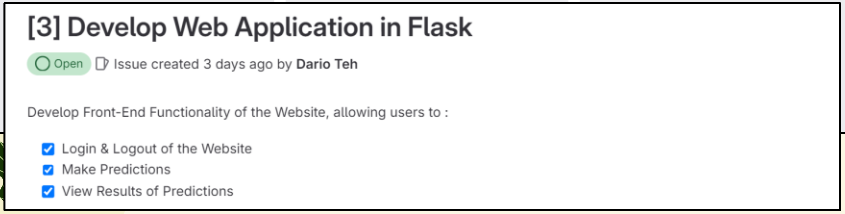
[2] Backend Image Processing (Resizing and Grayscaling)
#5

[2] Training & Testing the AI Model
#2

[3] Develop Web Application in Flask
#3

[4] History Page Front End
#6

[5] Design the Front-End for the Quiz Section
#8

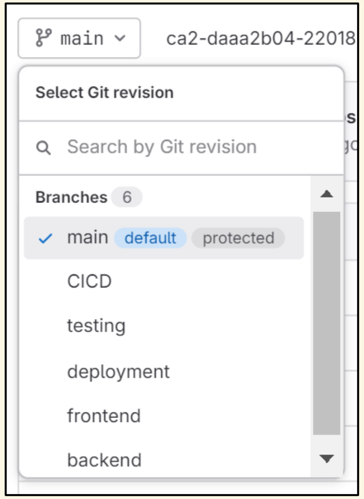


[3] Develop Web Application in Flask

Open Issue created 3 days ago by Dario Teh

Develop Front-End Functionality of the Website, allowing users to :

- [x] Login & Logout of the Website
- [x] Make Predictions
- [x] View Results of Predictions



main ca2-daaa2b04-22018

Select Git revision

Search by Git revision

Branches 6

✓ main default protected

CICD

testing

deployment

frontend

backend

GENERAL INFORMATION

ONLINE DEPLOYMENT

Website is deployed successfully on Render at : https://veggielens.onrender.com/

LOGIN AUTHENTICATION

Bypassing authentication by **typing in the direct link is addressed and prevented**.

The user can either log in or sign up for an account.

MULTIPLE DEVICE RESPONSIVENESS

Responsive web design using **Tailwind CSS** and **JavaScript**.

MODEL DEVELOPMENT

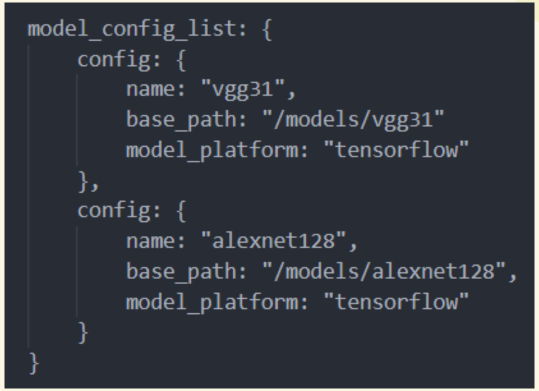
DEEP LEARNING MODELS USED

- 31 x 31 px Images : Custom VGG Model (vgg31)
- 128 x 128 px Images : Custom AlexNet Model (alexnet128)

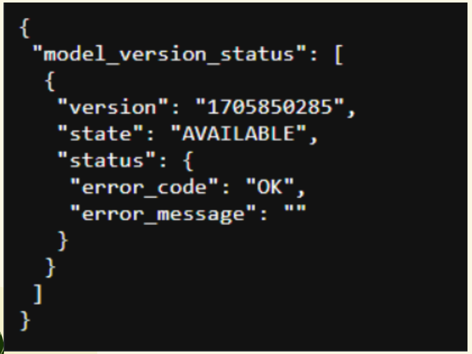
DEPLOYMENT PROCESS

- Both models were deployed together on the local DockerFile into the same Render URL.
- Checked that server is set up correctly by checking the model status from **/v1/models**.

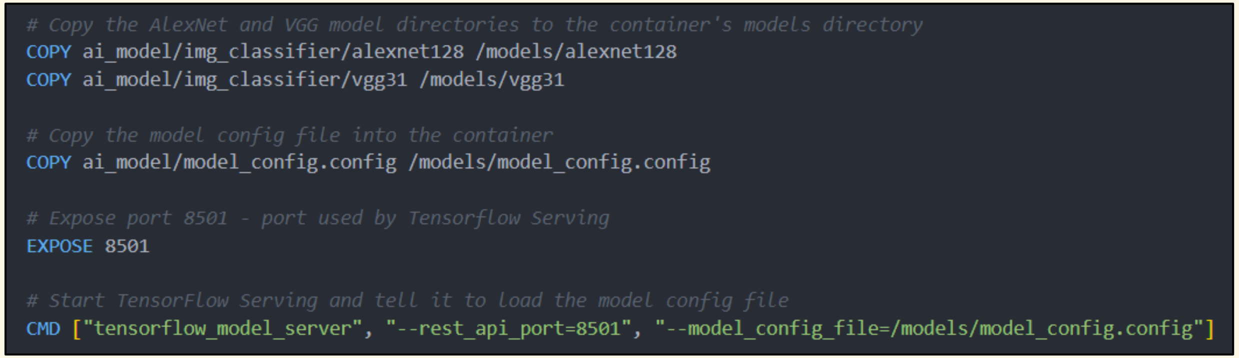
Model Config & Docker Files



```
model_config_list: {
    config: {
        name: "vgg31",
        base_path: "/models/vgg31"
        model_platform: "tensorflow"
    },
    config: {
        name: "alexnet128",
        base_path: "/models/alexnet128",
        model_platform: "tensorflow"
    }
}
```



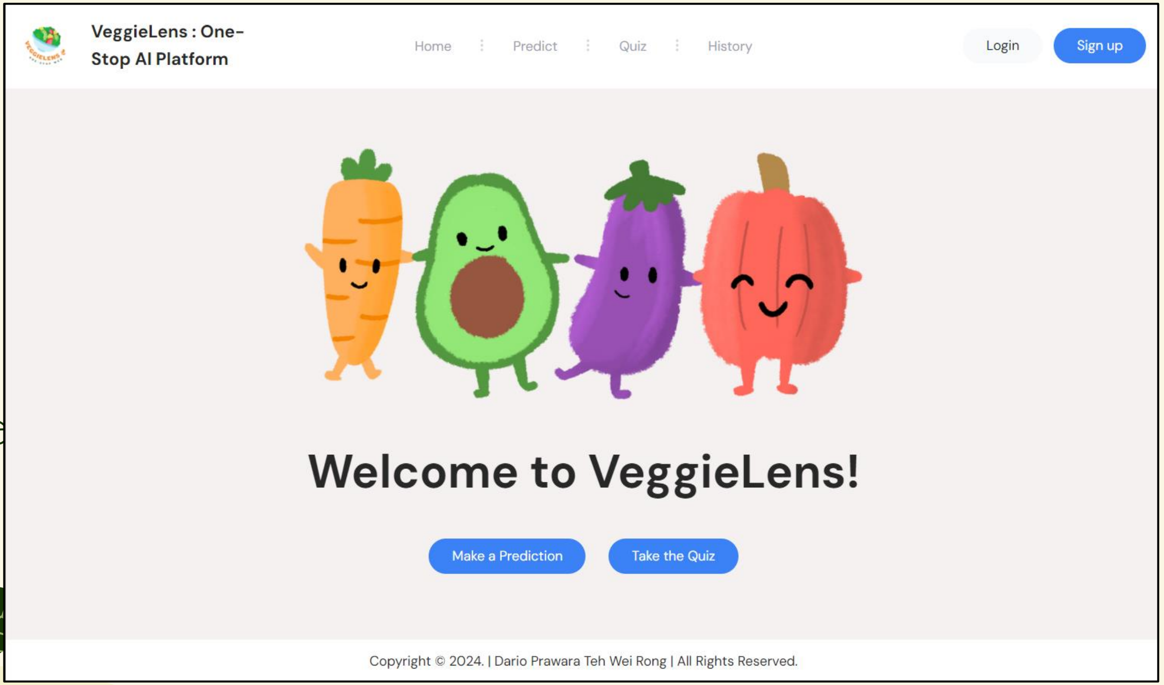
```
{
 "model_version_status": [
  {
   "version": "1705850285",
   "state": "AVAILABLE",
   "status": {
    "error_code": "OK",
    "error_message": ""
   }
  }
 ]
}
```



```
# Copy the AlexNet and VGG model directories to the container's models directory
COPY ai_model/img_classifier/alexnet128 /models/alexnet128
COPY ai_model/img_classifier/vgg31 /models/vgg31

# Copy the model config file into the container
COPY ai_model/model_config.config /models/model_config.config

# Expose port 8501 - port used by Tensorflow Serving
EXPOSE 8501

# Start TensorFlow Serving and tell it to load the model config file
CMD ["tensorflow_model_server", "--rest_api_port=8501", "--model_config_file=/models/model_config.config"]
```

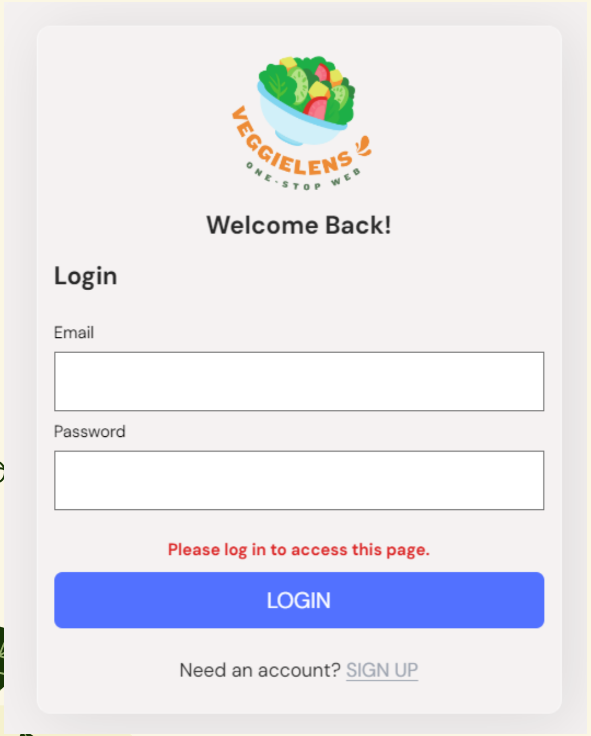
INTRODUCING…VEGGIELENS!

https://veggielens.onrender.com/


VeggieLens : One-Stop AI Platform
Home
Predict
Quiz
History
Login
Sign up
Welcome to VeggieLens!
Make a Prediction
Take the Quiz
Copyright © 2024. | Dario Prawara Teh Wei Rong | All Rights Reserved.

- VeggieLens is an AI web application where users can predict vegetable images & compete in a quiz with our AI in a mini competition.
- Users can also view history to review their mistakes to and improve for future quizzes.

VEGGIELENS – LOGIN / SIGNUP



Welcome Back!

Login

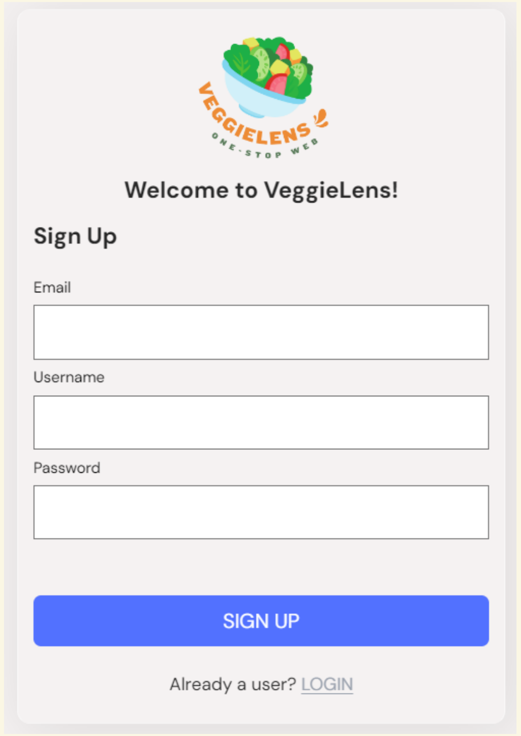
Email

Password

Please log in to access this page.

LOGIN

Need an account? SIGN UP



Welcome to VeggieLens!

Sign Up

Email

Username

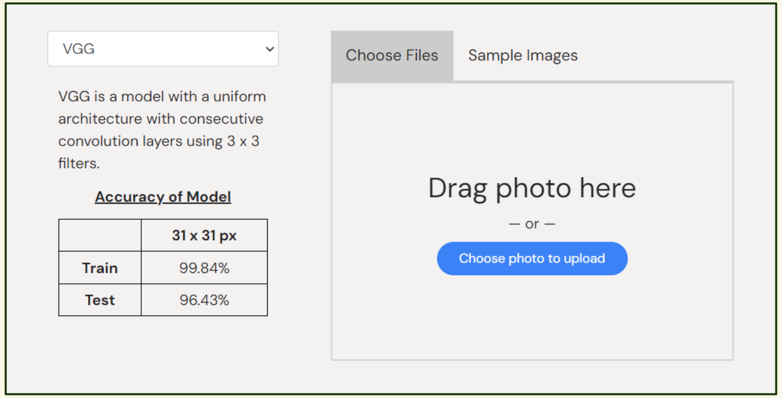
Password

SIGN UP

Already a user? LOGIN

- Users can either login or sign up (create a new account) to the web application.
- When the user signs up, it redirects them to the login page to enter credentials.

VEGGIELENS – PREDICTION PAGE



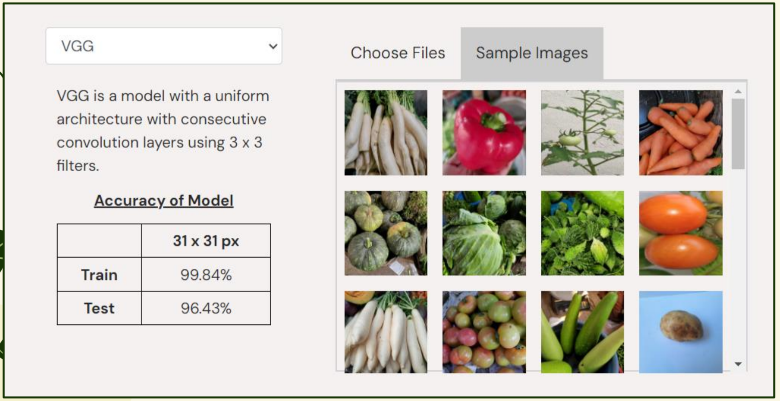
VGG is a model with a uniform architecture with consecutive convolution layers using 3 x 3 filters.

Accuracy of Model

	31 x 31 px
Train	99.84%
Test	96.43%

HOW DOES THE USER MAKE PREDICTIONS

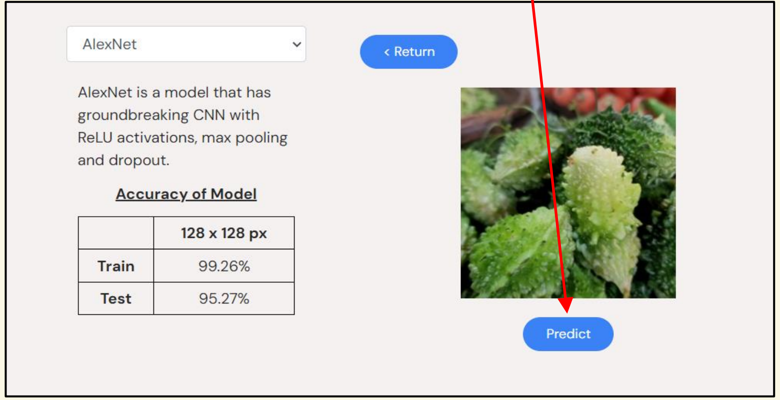
- Either **upload an image** or **choose from pre-loaded images**.
- Select model type **(VGG for 31px or AlexNet for 128px images)** with **information** on each model.
- After uploading the image, a 'Predict' button appears for the user to predict.



VGG is a model with a uniform architecture with consecutive convolution layers using 3 x 3 filters.

Accuracy of Model

	31 x 31 px
Train	99.84%
Test	96.43%



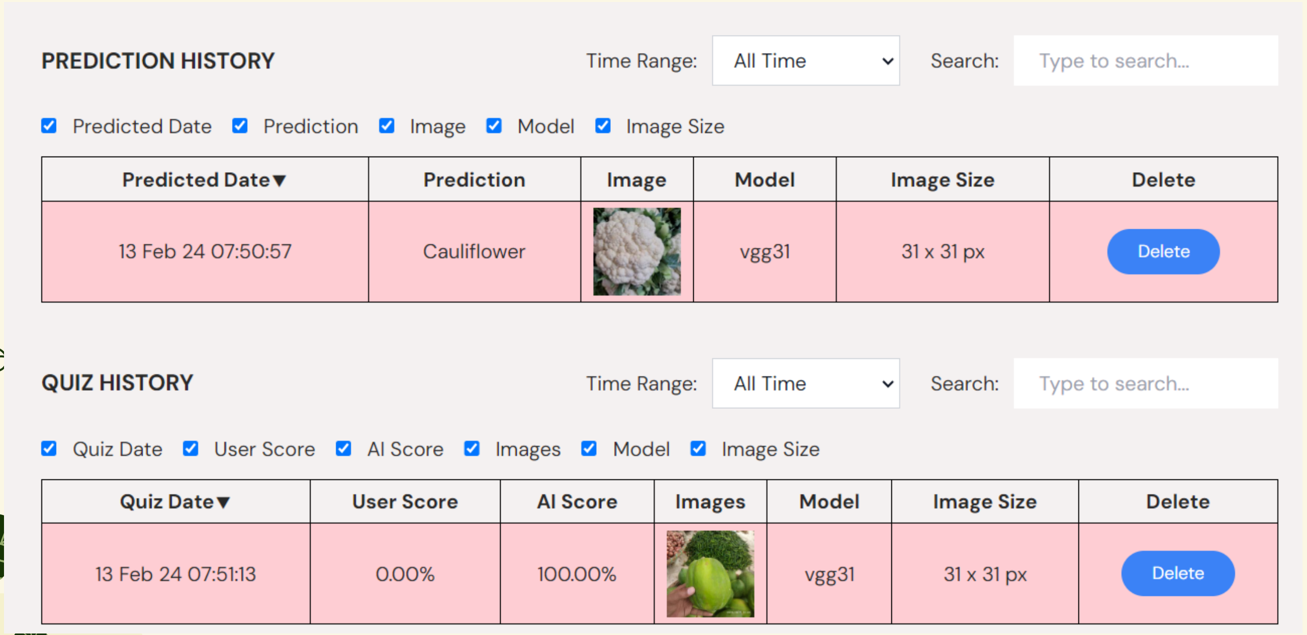
AlexNet is a model that has groundbreaking CNN with ReLU activations, max pooling and dropout.

Accuracy of Model

	128 x 128 px
Train	99.26%
Test	95.27%

VEGGIELENS – HISTORY PAGE

- Filter options : Time Range & Columns to Display
- Search & sort options : Search / Sort by any column (except Delete & Image(s))



PREDICTION HISTORY

Time Range: All Time

Search: Type to search...

☑ Predicted Date ☑ Prediction ☑ Image ☑ Model ☑ Image Size

Predicted Date▼	Prediction	Image	Model	Image Size	Delete
13 Feb 24 07:50:57	Cauliflower		vgg31	31 x 31 px	Delete

QUIZ HISTORY

Time Range: All Time

Search: Type to search...

☑ Quiz Date ☑ User Score ☑ AI Score ☑ Images ☑ Model ☑ Image Size

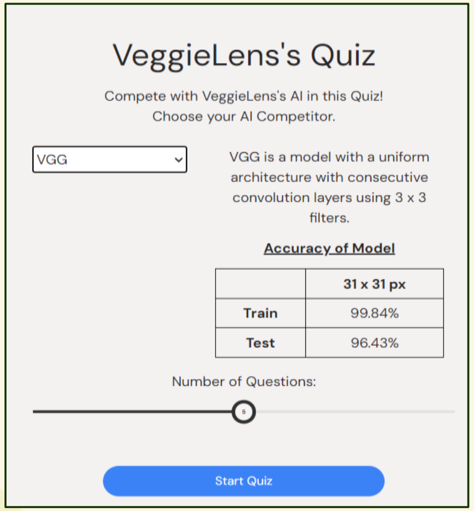
Quiz Date▼	User Score	AI Score	Images	Model	Image Size	Delete
13 Feb 24 07:51:13	0.00%	100.00%		vgg31	31 x 31 px	Delete

- Dynamic filtering & searching functionality (updates as it changes).
- Able to view past histories for predictions and quizzes.

VEGGIELENS – QUIZ

- The user can choose the AI model and number of questions for the quiz.
- A sample image and a dropdown allows users to select the class the image best belongs to.

- Results are shown to display both the user's score as well as the AI Model's score.



VeggieLens's Quiz

Compete with VeggieLens's AI in this Quiz!
Choose your AI Competitor.

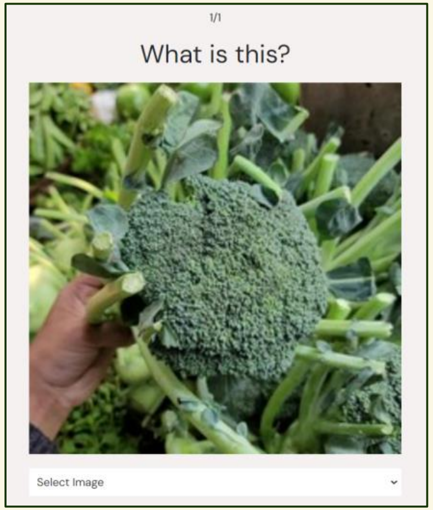
VGG

VGG is a model with a uniform architecture with consecutive convolution layers using 3 x 3 filters.

Accuracy of Model

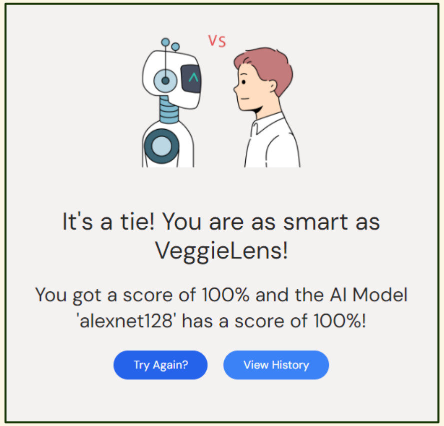
	31 x 31 px
Train	99.84%
Test	96.43%

Number of Questions:



1/1

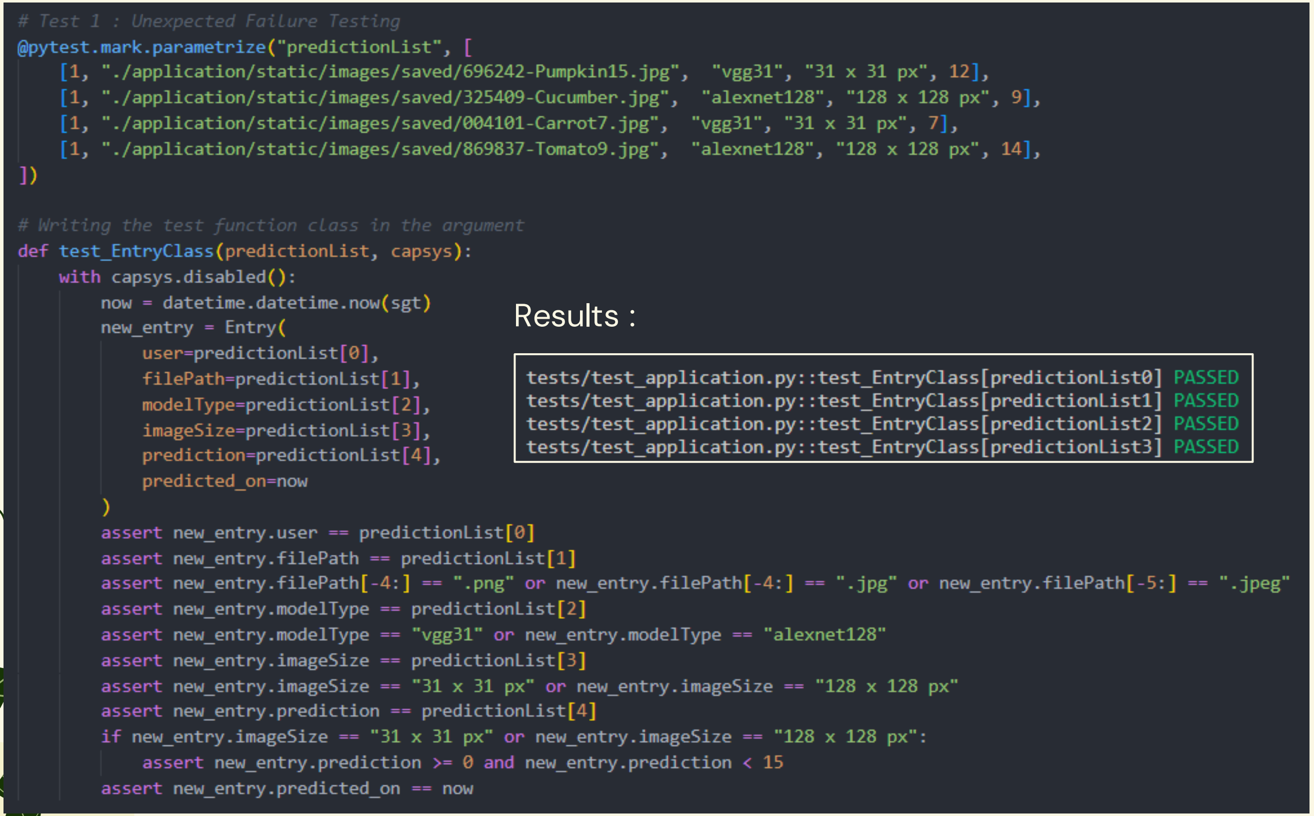
What is this?



It's a tie! You are as smart as VeggieLens!

You got a score of 100% and the AI Model 'alexnet128' has a score of 100%!

UNEXPECTED FAILURE TESTING



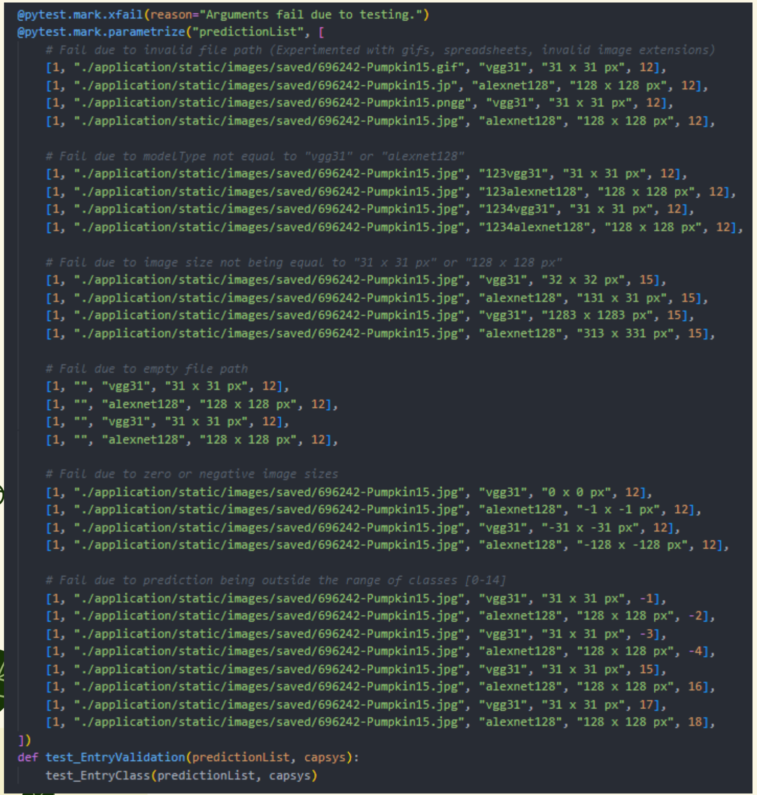
```
# Test 1 : Unexpected Failure Testing
@pytest.mark.parametrize("predictionList", [
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg",  "vgg31", "31 x 31 px", 12],
    [1, "./application/static/images/saved/325409-Cucumber.jpg",  "alexnet128", "128 x 128 px", 9],
    [1, "./application/static/images/saved/004101-Carrot7.jpg",  "vgg31", "31 x 31 px", 7],
    [1, "./application/static/images/saved/869837-Tomato9.jpg",  "alexnet128", "128 x 128 px", 14],
])

# Writing the test function class in the argument
def test_EntryClass(predictionList, capsys):
    with capsys.disabled():
        now = datetime.datetime.now(sgt)
        new_entry = Entry(
            user=predictionList[0],
            filePath=predictionList[1],
            modelType=predictionList[2],
            imageSize=predictionList[3],
            prediction=predictionList[4],
            predicted_on=now
        )
        assert new_entry.user == predictionList[0]
        assert new_entry.filePath == predictionList[1]
        assert new_entry.filePath[-4:] == ".png" or new_entry.filePath[-4:] == ".jpg" or new_entry.filePath[-5:] == ".jpeg"
        assert new_entry.modelType == predictionList[2]
        assert new_entry.modelType == "vgg31" or new_entry.modelType == "alexnet128"
        assert new_entry.imageSize == predictionList[3]
        assert new_entry.imageSize == "31 x 31 px" or new_entry.imageSize == "128 x 128 px"
        assert new_entry.prediction == predictionList[4]
        if new_entry.imageSize == "31 x 31 px" or new_entry.imageSize == "128 x 128 px":
            assert new_entry.prediction >= 0 and new_entry.prediction < 15
        assert new_entry.predicted_on == now
```

Results :

```
tests/test_application.py::test_EntryClass[predictionList0] PASSED
tests/test_application.py::test_EntryClass[predictionList1] PASSED
tests/test_application.py::test_EntryClass[predictionList2] PASSED
tests/test_application.py::test_EntryClass[predictionList3] PASSED
```

- Created a test for unexpected failure testing for data sent to the SQLite Database.
- All results **PASSED** as no failure was expected from the parameters inputted.

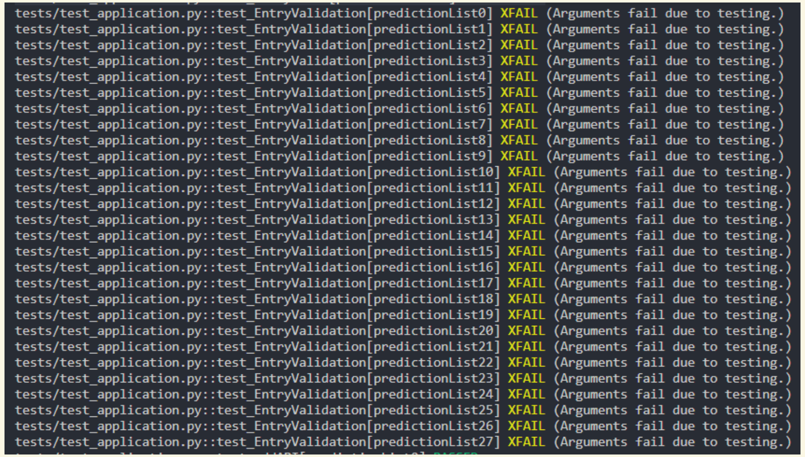
EXPECTED FAILURE TESTING



```python
@pytest.mark.xfail(reason="Arguments fail due to testing.")
@pytest.mark.parametrize("predictionList", [
    # Fail due to invalid file path (Experimented with gifs, spreadsheets, invalid image extensions)
    [1, "./application/static/images/saved/696242-Pumpkin15.gif", "vgg31", "31 x 31 px", 12],
    [1, "./application/static/images/saved/696242-Pumpkin15.jp", "alexnet128", "128 x 128 px", 12],
    [1, "./application/static/images/saved/696242-Pumpkin15.pngg", "vgg31", "31 x 31 px", 12],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "alexnet128", "128 x 128 px", 12],

    # Fail due to modelType not equal to "vgg31" or "alexnet128"
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "123vgg31", "31 x 31 px", 12],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "123alexnet128", "128 x 128 px", 12],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "1234vgg31", "31 x 31 px", 12],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "1234alexnet128", "128 x 128 px", 12],

    # Fail due to image size not being equal to "31 x 31 px" or "128 x 128 px"
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "vgg31", "32 x 32 px", 15],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "alexnet128", "131 x 31 px", 15],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "vgg31", "1283 x 1283 px", 15],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "alexnet128", "313 x 331 px", 15],

    # Fail due to empty file path
    [1, "", "vgg31", "31 x 31 px", 12],
    [1, "", "alexnet128", "128 x 128 px", 12],
    [1, "", "vgg31", "31 x 31 px", 12],
    [1, "", "alexnet128", "128 x 128 px", 12],

    # Fail due to zero or negative image sizes
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "vgg31", "0 x 0 px", 12],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "alexnet128", "-1 x -1 px", 12],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "vgg31", "-31 x -31 px", 12],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "alexnet128", "-128 x -128 px", 12],

    # Fail due to prediction being outside the range of classes [0-14]
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "vgg31", "31 x 31 px", -1],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "alexnet128", "128 x 128 px", -2],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "vgg31", "31 x 31 px", -3],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "alexnet128", "128 x 128 px", -4],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "vgg31", "31 x 31 px", 15],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "alexnet128", "128 x 128 px", 16],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "vgg31", "31 x 31 px", 17],
    [1, "./application/static/images/saved/696242-Pumpkin15.jpg", "alexnet128", "128 x 128 px", 18],
])
def test_EntryValidation(predictionList, capsys):
    test_EntryClass(predictionList, capsys)
```

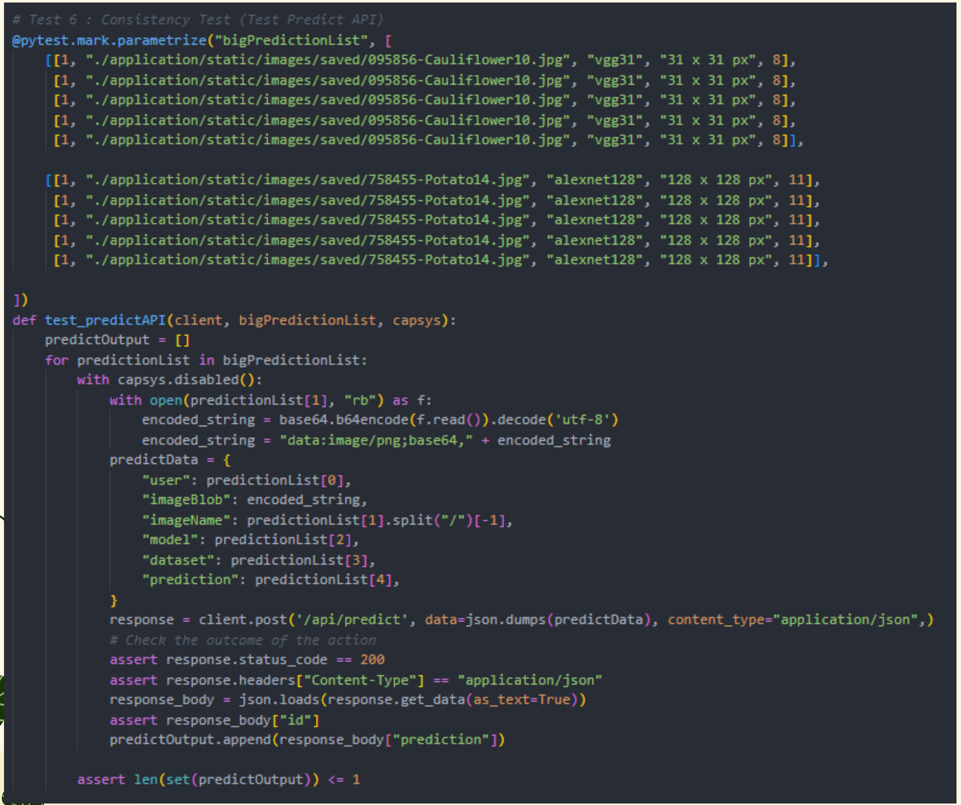
For expected testing, the following were checked :

- Invalid file path
- Invalid model type (not VGG or AlexNet)
- Invalid image size processed
- Empty file paths
- Zero or negative image sizes
- Prediction out of the class range – 0 to 14



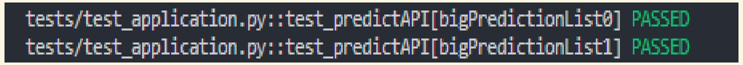
```
tests/test_application.py::test_EntryValidation[predictionList0] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList1] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList2] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList3] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList4] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList5] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList6] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList7] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList8] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList9] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList10] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList11] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList12] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList13] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList14] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList15] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList16] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList17] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList18] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList19] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList20] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList21] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList22] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList23] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList24] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList25] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList26] XFAIL (Arguments fail due to testing.)
tests/test_application.py::test_EntryValidation[predictionList27] XFAIL (Arguments fail due to testing.)
```

CONSISTENCY TESTING



```python
# Test 6 : Consistency Test (Test Predict API)
@pytest.mark.parametrize("bigPredictionList", [
    [[1, "./application/static/images/saved/095856-Cauliflower10.jpg", "vgg31", "31 x 31 px", 8],
     [1, "./application/static/images/saved/095856-Cauliflower10.jpg", "vgg31", "31 x 31 px", 8],
     [1, "./application/static/images/saved/095856-Cauliflower10.jpg", "vgg31", "31 x 31 px", 8],
     [1, "./application/static/images/saved/095856-Cauliflower10.jpg", "vgg31", "31 x 31 px", 8],
     [1, "./application/static/images/saved/095856-Cauliflower10.jpg", "vgg31", "31 x 31 px", 8]],

    [[1, "./application/static/images/saved/758455-Potato14.jpg", "alexnet128", "128 x 128 px", 11],
     [1, "./application/static/images/saved/758455-Potato14.jpg", "alexnet128", "128 x 128 px", 11],
     [1, "./application/static/images/saved/758455-Potato14.jpg", "alexnet128", "128 x 128 px", 11],
     [1, "./application/static/images/saved/758455-Potato14.jpg", "alexnet128", "128 x 128 px", 11],
     [1, "./application/static/images/saved/758455-Potato14.jpg", "alexnet128", "128 x 128 px", 11]],

])
def test_predictAPI(client, bigPredictionList, capsys):
    predictOutput = []
    for predictionList in bigPredictionList:
        with capsys.disabled():
            with open(predictionList[1], "rb") as f:
                encoded_string = base64.b64encode(f.read()).decode('utf-8')
                encoded_string = "data:image/png;base64," + encoded_string
            predictData = {
                "user": predictionList[0],
                "imageBlob": encoded_string,
                "imageName": predictionList[1].split("/")[-1],
                "model": predictionList[2],
                "dataset": predictionList[3],
                "prediction": predictionList[4],
            }
            response = client.post('/api/predict', data=json.dumps(predictData), content_type="application/json",)
            # Check the outcome of the action
            assert response.status_code == 200
            assert response.headers["Content-Type"] == "application/json"
            response_body = json.loads(response.get_data(as_text=True))
            assert response_body["id"]
            predictOutput.append(response_body["prediction"])

        assert len(set(predictOutput)) <= 1
```

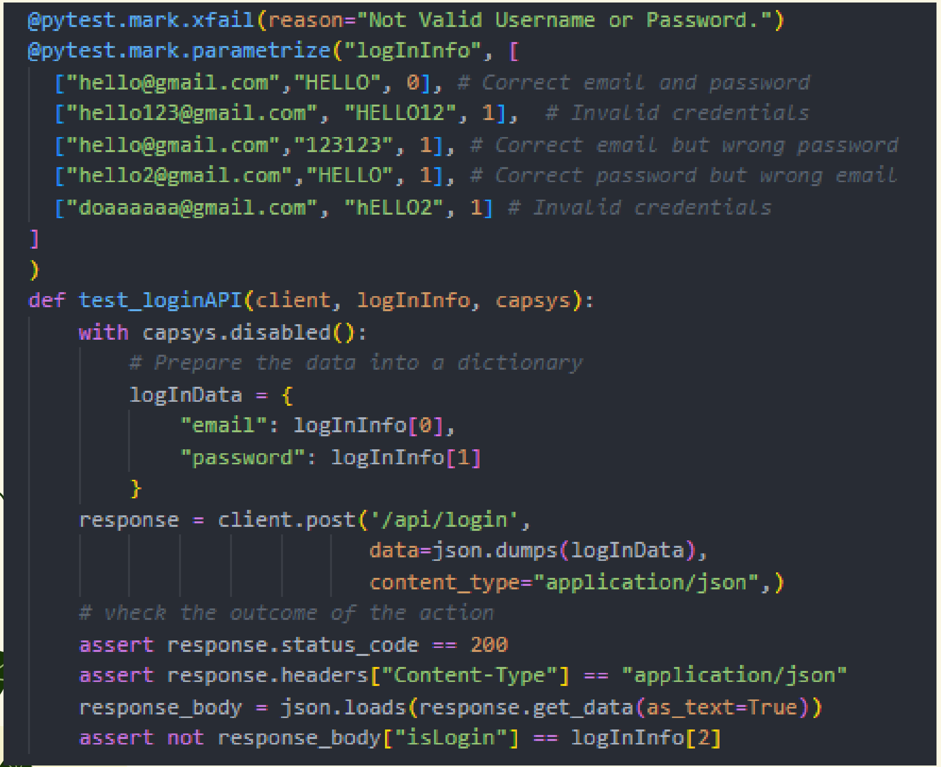
Results :



```
tests/test_application.py::test_predictAPI[bigPredictionList0] PASSED
tests/test_application.py::test_predictAPI[bigPredictionList1] PASSED
```

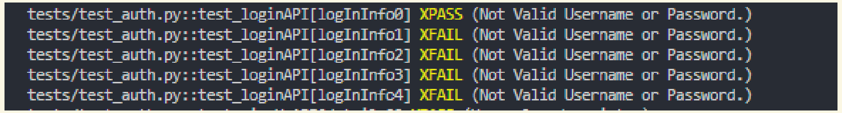
- Given identical inputs, it checks if the predictions are the same as each other.
- Grouped 2 arrays with random indexing, for Cauliflower and Potato, to test if the output predictions are consistent.

VALIDITY TESTING



```python
@pytest.mark.xfail(reason="Not Valid Username or Password.")
@pytest.mark.parametrize("logInInfo", [
    ["hello@gmail.com","HELLO", 0], # Correct email and password
    ["hello123@gmail.com", "HELLO12", 1],  # Invalid credentials
    ["hello@gmail.com","123123", 1], # Correct email but wrong password
    ["hello2@gmail.com","HELLO", 1], # Correct password but wrong email
    ["doaaaaaa@gmail.com", "hELLO2", 1] # Invalid credentials
]
)
def test_loginAPI(client, logInInfo, capsys):
    with capsys.disabled():
        # Prepare the data into a dictionary
        logInData = {
            "email": logInInfo[0],
            "password": logInInfo[1]
        }
    response = client.post('/api/login',
                           data=json.dumps(logInData),
                           content_type="application/json",)
    # vheck the outcome of the action
    assert response.status_code == 200
    assert response.headers["Content-Type"] == "application/json"
    response_body = json.loads(response.get_data(as_text=True))
    assert not response_body["isLogin"] == logInInfo[2]
```

Results :



```
tests/test_auth.py::test_loginAPI[logInInfo0] XPASS (Not Valid Username or Password.)
tests/test_auth.py::test_loginAPI[logInInfo1] XFAIL (Not Valid Username or Password.)
tests/test_auth.py::test_loginAPI[logInInfo2] XFAIL (Not Valid Username or Password.)
tests/test_auth.py::test_loginAPI[logInInfo3] XFAIL (Not Valid Username or Password.)
tests/test_auth.py::test_loginAPI[logInInfo4] XFAIL (Not Valid Username or Password.)
```

- Tests on valid data to determine if ordinary data can be used. [Expected & Valid Working Data]
- Since the first parameter is a **valid email and password**, it returns XPASS, and the rest returns XFAIL as they are all invalid.

ENDPOINT API TESTING

POST
/API/LOGIN

API used to perform authentication of users based on email & password credentials.

POST
/API/SIGNUP

API used to manage user registration and account creation within the system.

DELETE
/API/DELETE/{id}

Given the ID of the history, delete the record for the user.

GET
/API/GET/{predictionList[0]}

API used to retrieve information of a specific result given the ID.

POST
/API/ADD

API used to add a new prediction record into the database.

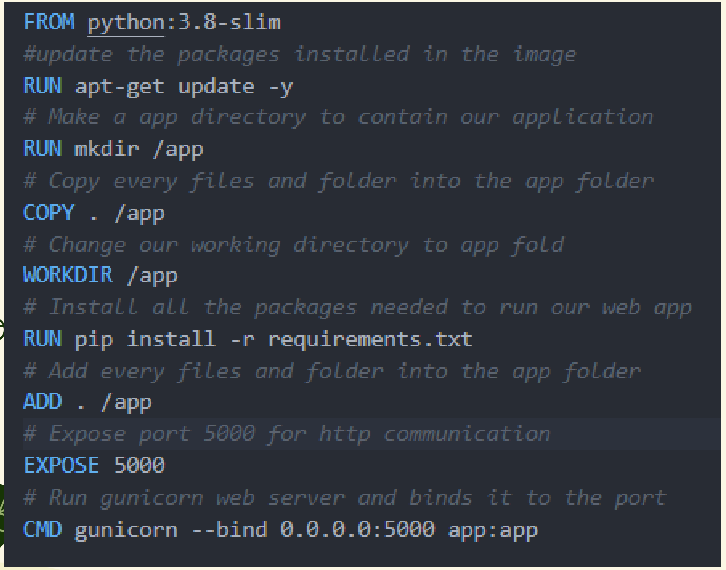
POST
/API/PREDICT

API used to perform prediction and retrieve result of class predicted.

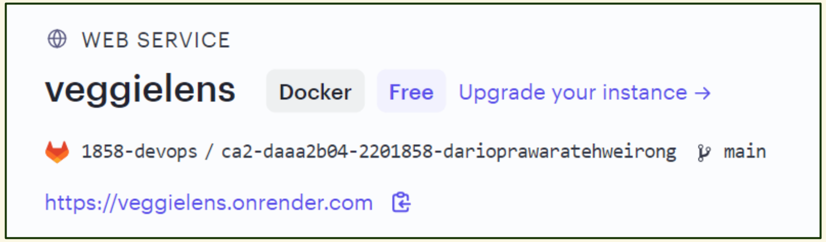
MLOPS – DEPLOYMENT

- Use Render Web Service to deploy the model & website to the internet.
- Render runs the Dockerfile that uses gunicorn as the WSL Server to host the website online.

Dockerfile



```
FROM python:3.8-slim
#update the packages installed in the image
RUN apt-get update -y
# Make a app directory to contain our application
RUN mkdir /app
# Copy every files and folder into the app folder
COPY . /app
# Change our working directory to app fold
WORKDIR /app
# Install all the packages needed to run our web app
RUN pip install -r requirements.txt
# Add every files and folder into the app folder
ADD . /app
# Expose port 5000 for http communication
EXPOSE 5000
# Run gunicorn web server and binds it to the port
CMD gunicorn --bind 0.0.0.0:5000 app:app
```



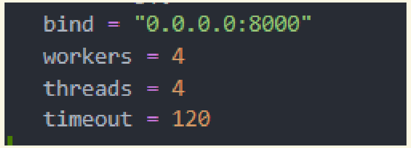
WEB SERVICE

veggielens Docker Free Upgrade your instance →

1858-devops / ca2-daaa2b04-2201858-darioprawaratehweirong main

https://veggielens.onrender.com

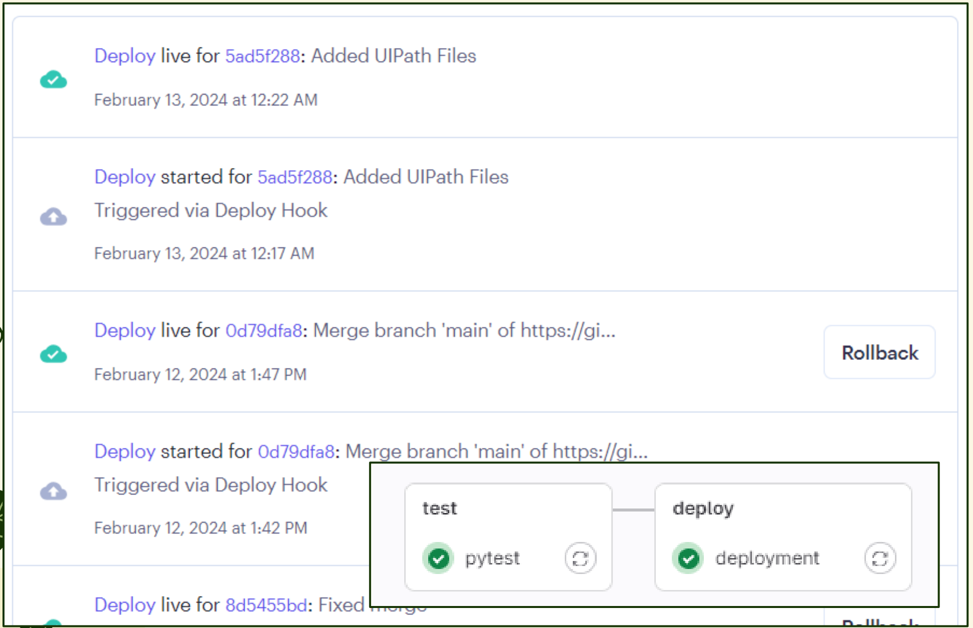
Gunicorn File



```
bind = "0.0.0.0:8000"
workers = 4
threads = 4
timeout = 120
```

MLOPS – CI / CD

- Linked to GitLab’s main branch
- Changes are **automatically deployed on Render**



Deploy live for 5ad5f288: Added UIPath Files
February 13, 2024 at 12:22 AM

Deploy started for 5ad5f288: Added UIPath Files
Triggered via Deploy Hook
February 13, 2024 at 12:17 AM

Deploy live for 0d79dfa8: Merge branch 'main' of https://gi...
February 12, 2024 at 1:47 PM

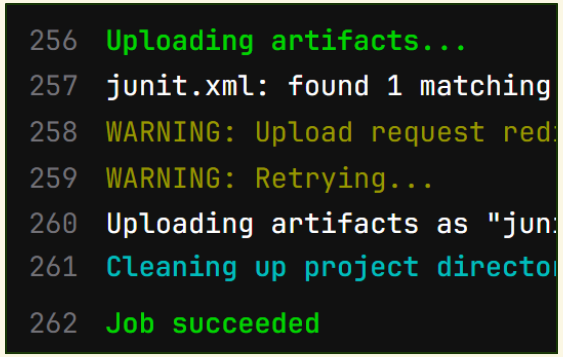
Rollback

Deploy started for 0d79dfa8: Merge branch 'main' of https://gi...
Triggered via Deploy Hook
February 12, 2024 at 1:42 PM

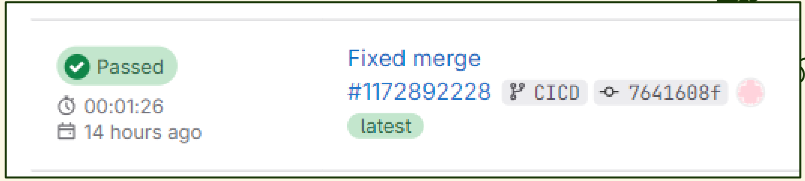
Deploy live for 8d5455bd: Fixed merge

test	deploy
✓ pytest	✓ deployment

- Pipeline jobs are successful – all passed
- Successfully integrating features to the main branch in GitLab



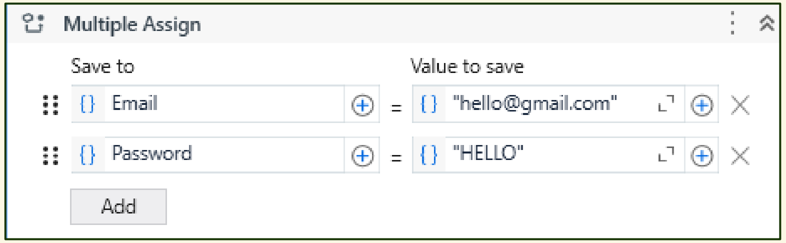
```
Uploading artifacts...
junit.xml: found 1 matching
WARNING: Upload request red
WARNING: Retrying...
Uploading artifacts as "jun
Cleaning up project directo

Job succeeded
```



✓ Passed
00:01:26
14 hours ago

Fixed merge
#1172892228 CICD 7641608f
latest

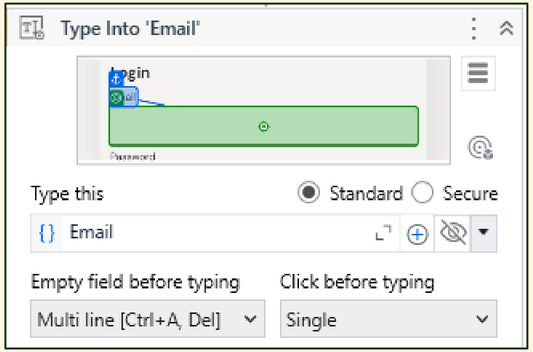
UIPATH – LOGIN RPA



Multiple Assign

Save to		Value to save
Email	=	"hello@gmail.com"
Password	=	"HELLO"

Add



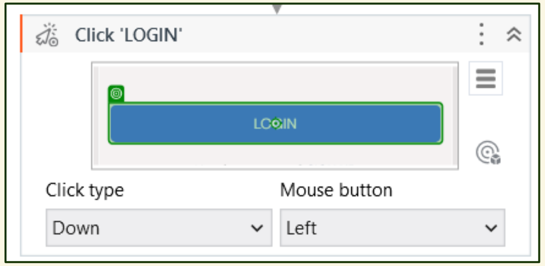
Type Into 'Email'

Type this: [✓ Standard] Secure

Email

Empty field before typing: Multi line [Ctrl+A, Del]

Click before typing: Single



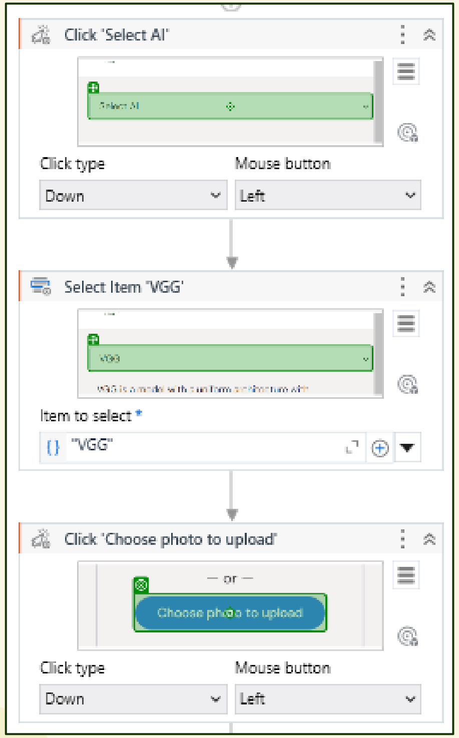
Click 'LOGIN'

Click type: Down

Mouse button: Left

- Assign and provide multiple assignments to the bot so that it knows what to type in the input boxes.
- Use “Type Into” activities to type into the input boxes. Allow the bot to click on the login button using the “Click” activity.

UIPATH – PREDICT RPA


Click 'Select AI'

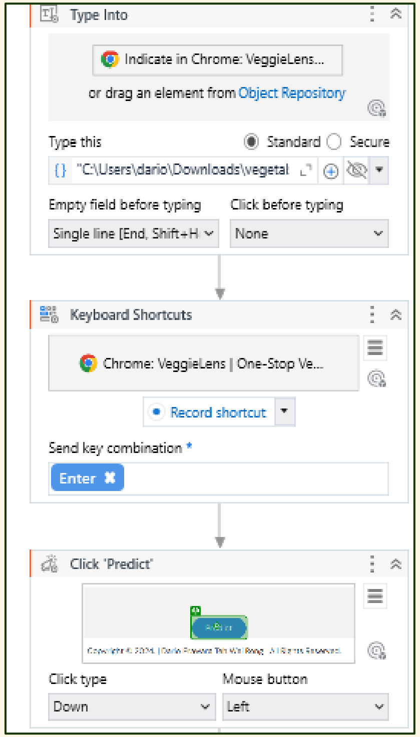
Click type: Down
Mouse button: Left

Select Item 'VGG'

Item to select *: "VGG"

Click 'Choose photo to upload'

Click type: Down
Mouse button: Left


Type Into

Indicate in Chrome: VeggieLens...
or drag an element from Object Repository

Type this: [✓ Standard] Secure
"C:\Users\dario\Downloads\vegetal

Empty field before typing: Single line [End, Shift+H
Click before typing: None

Keyboard Shortcuts

Chrome: VeggieLens | One-Stop Ve...

Send key combination *: Enter

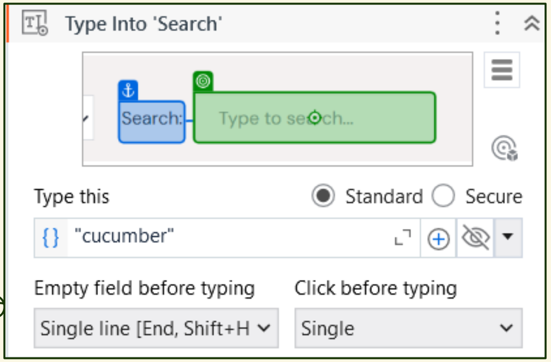
Click 'Predict'

Click type: Down
Mouse button: Left

- Use VGG model for demonstration. Apply click type and item select for the bot to select the model.
- Use the "Click" activity to upload photo, and "Type Into" to input the file path to the image.
- Apply the "Enter" key combination and another "Click" activity to predict the image.

UIPATH – SEARCH & DELETE RPA

SEARCH



Type Into 'Search'

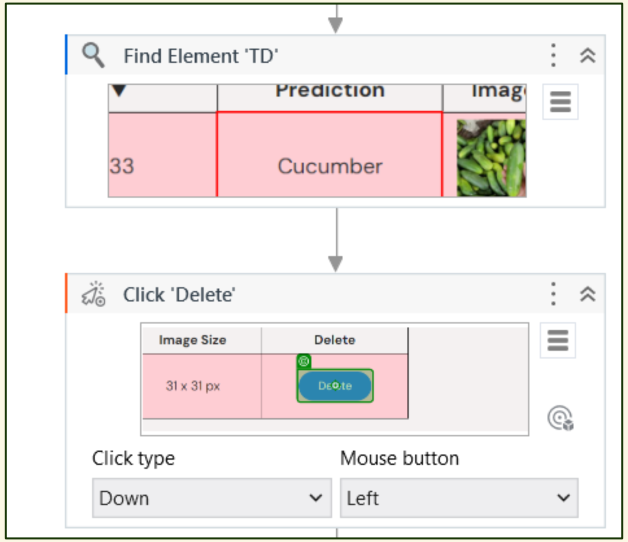
Search: Type to search...

Type this [✓ Standard] Secure

"cucumber"

Empty field before typing	Click before typing
Single line [End, Shift+H	Single

DELETE



Find Element 'TD'

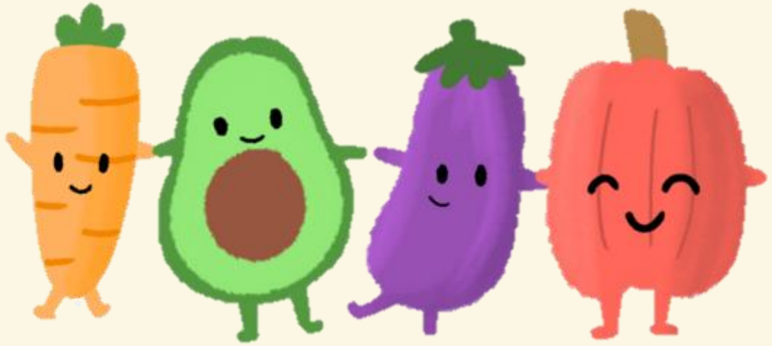
	Prediction	Imag
33	Cucumber	

Click 'Delete'

Image Size	Delete
31 x 31 px	Delete

Click type	Mouse button
Down	Left

- Use a combination of "Type Into" and "Type this" activities to enter (cucumber) into the search bar.
- Use the "Find Element" activity to locate the element in the page and delete the relevant entry using "Click" activity.



THANK YOU!