

❖ A. I. Project:

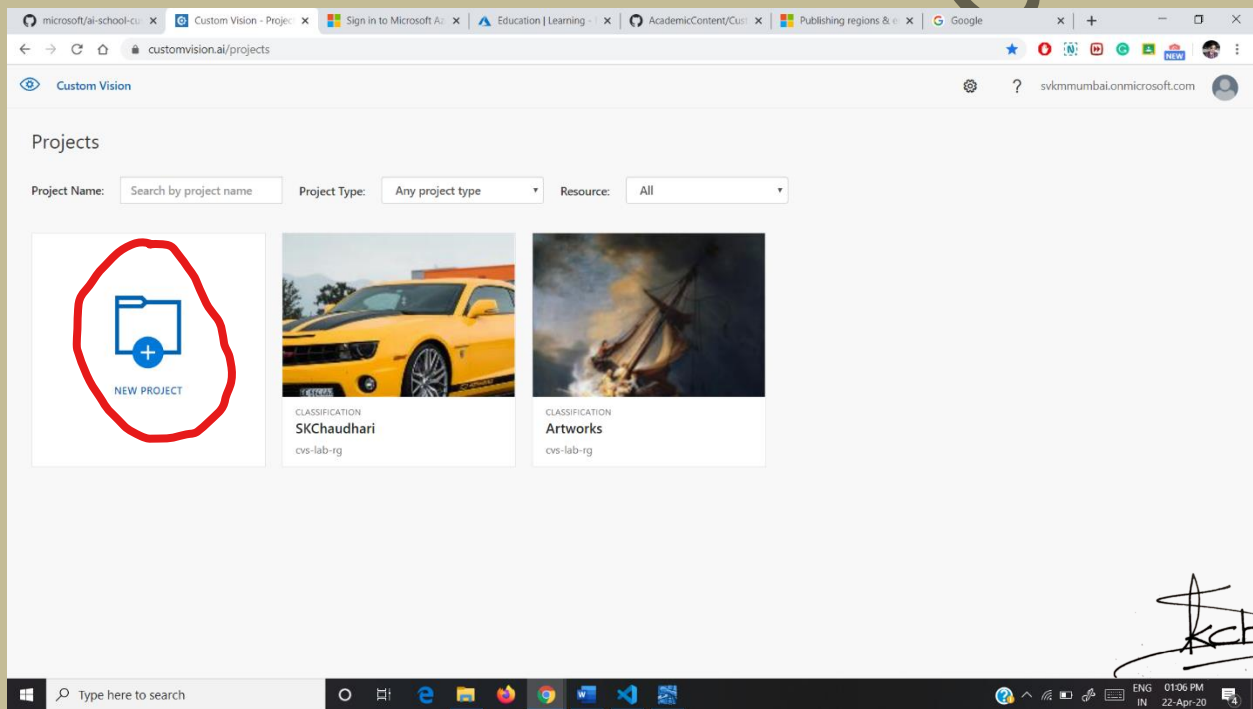
Project Name: Image Classification and Recognition App using Node.js and Azure API's.

Name: Pooja Suresh Chaudhari Roll No:07

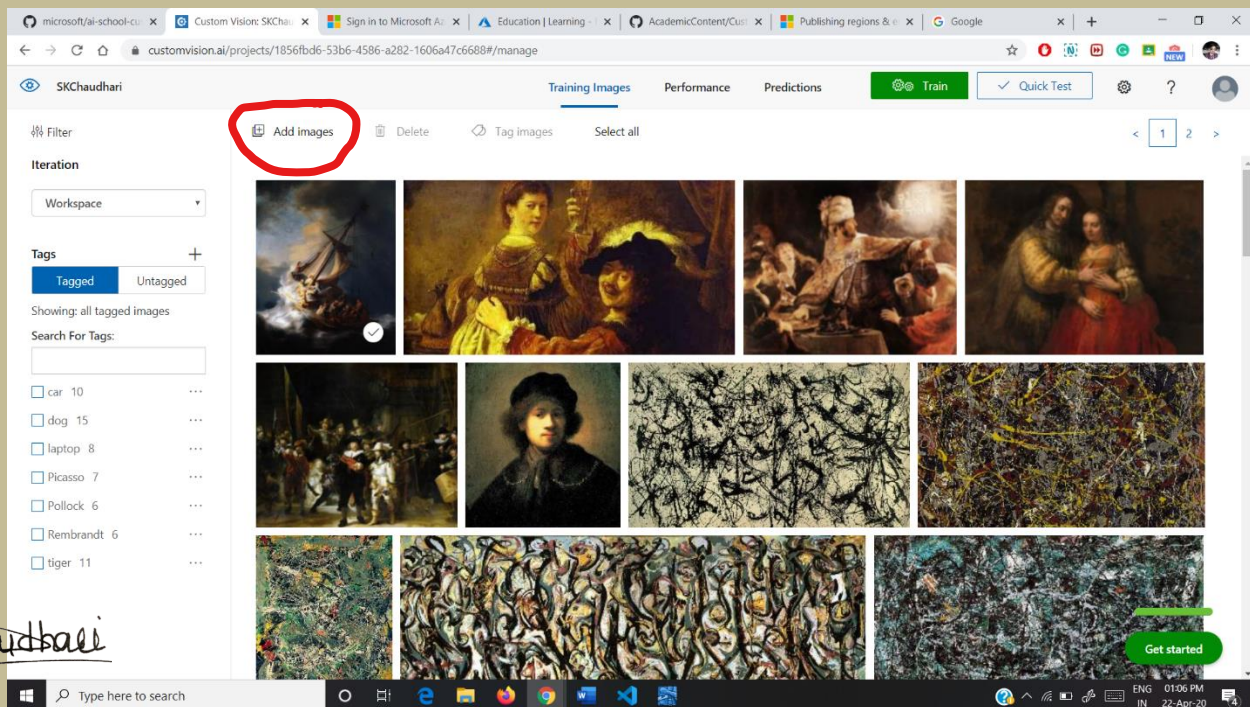
Name: Sanket Kishor Chaudhari Roll No: 50

Module 1) Create A.I. based model to generate API.

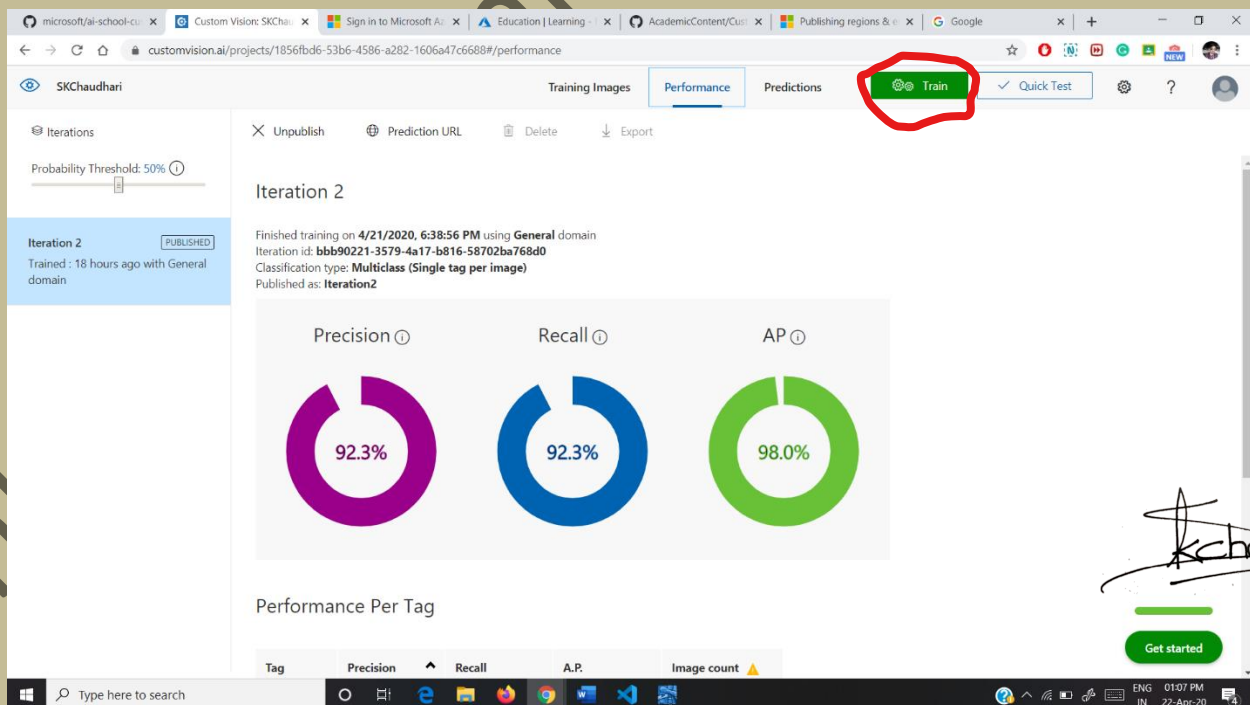
1) Click NEW PROJECT to create project.



2) Add images for training of model and attach Tags to them.



3) Now train the model.



SKChaudhari

Training Images **Performance** Predictions Train Quick Test

Iterations

Probability Threshold: 50%

Iteration 2 **PUBLISHED**
Trained: 18 hours ago with General domain

Unpublish Prediction URL Delete Export

Performance Per Tag

Tag	Precision	Recall	A.P.	Image count
Pollock	100.0%	100.0%	100.0%	6
Picasso	100.0%	100.0%	100.0%	7
laptop	100.0%	100.0%	100.0%	8
dog	100.0%	100.0%	100.0%	15
car	100.0%	100.0%	100.0%	10
tiger	66.7%	100.0%	100.0%	11
Rembrandt	0.0%	0.0%	100.0%	6

Get started

4) After training test the model.

SKChaudhari

Quick Test

Image URL

Enter Image URL

or

Browse local files

File formats accepted: .jpg, .png, .bmp
File size should not exceed: 4mb

Using model trained in

Iteration

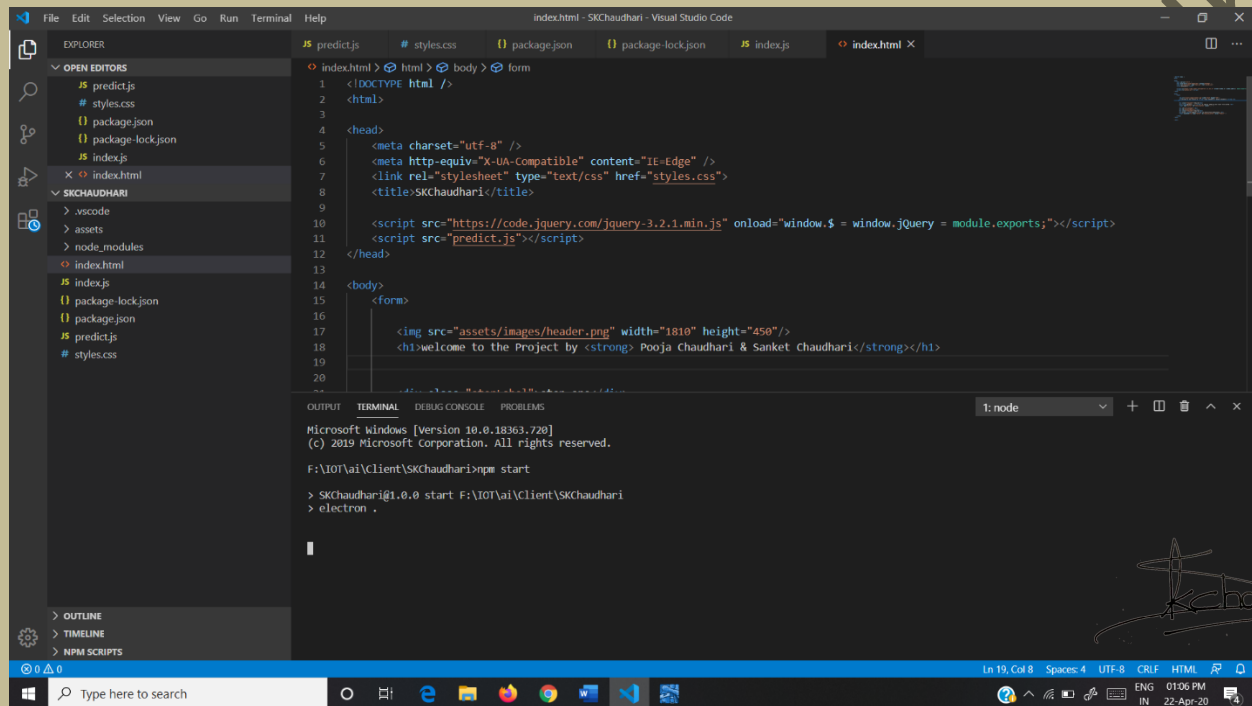
Iteration 2

Predictions

Tag	Probability
car	100%
Picasso	0%
Rembrandt	0%

Module 2) Create a Node.js app that uses the model for Classification and Recognition.

1) Use visual studio code for App building and calling API of Azure model.

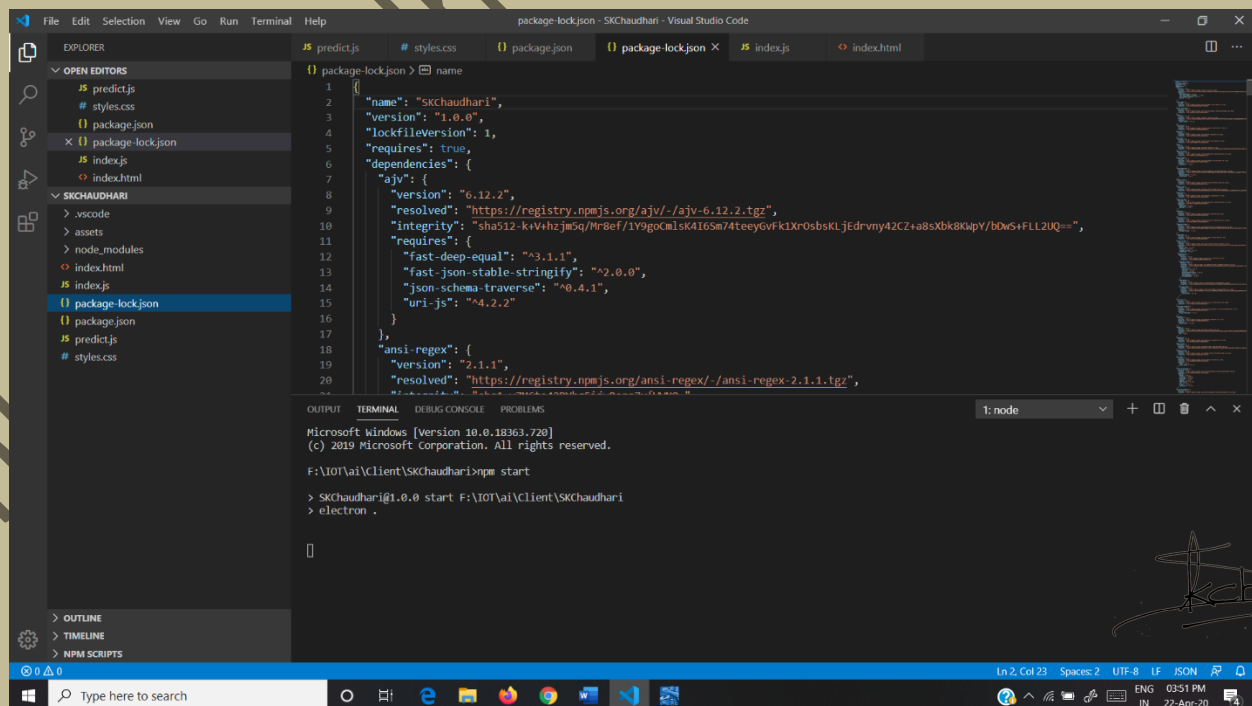


```
index.html - SKChaudhari - Visual Studio Code
1 <!DOCTYPE html />
2 <html>
3
4 <head>
5   <meta charset="utf-8" />
6   <meta http-equiv="X-UA-Compatible" content="IE=edge" />
7   <link rel="stylesheet" type="text/css" href="styles.css">
8   <title>SKChaudhari</title>
9
10  <script src="https://code.jquery.com/jquery-3.2.1.min.js" onload="window.$ = window.jQuery = module.exports;"></script>
11  <script src="predict.js"></script>
12 </head>
13
14 <body>
15   <form>
16
17     
18     <h1>welcome to the Project by <strong>Pooja Chaudhari & Sanket Chaudhari</strong></h1>
19
20  </body>
21 </html>
```

Microsoft Windows [version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

F:\IoT\ai\Client\SKChaudhari>npm start

> SKChaudhari@1.0.0 start F:\IoT\ai\Client\SKChaudhari
> electron .



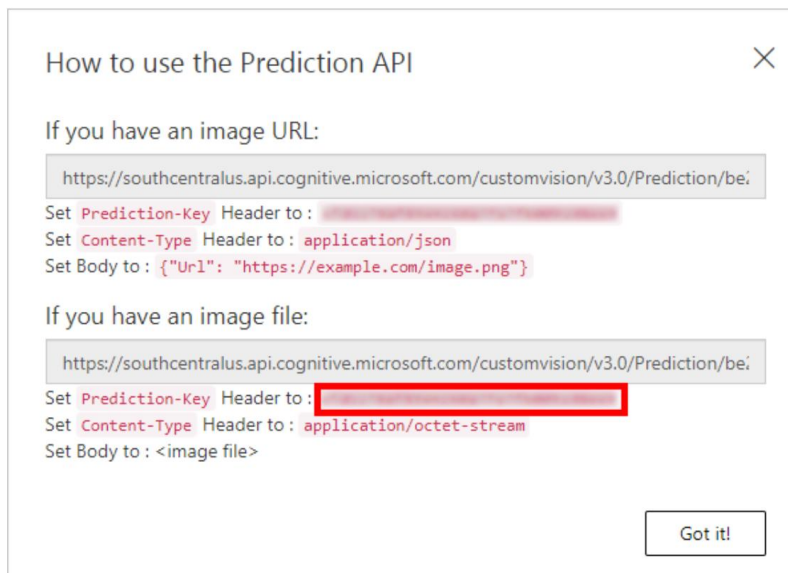
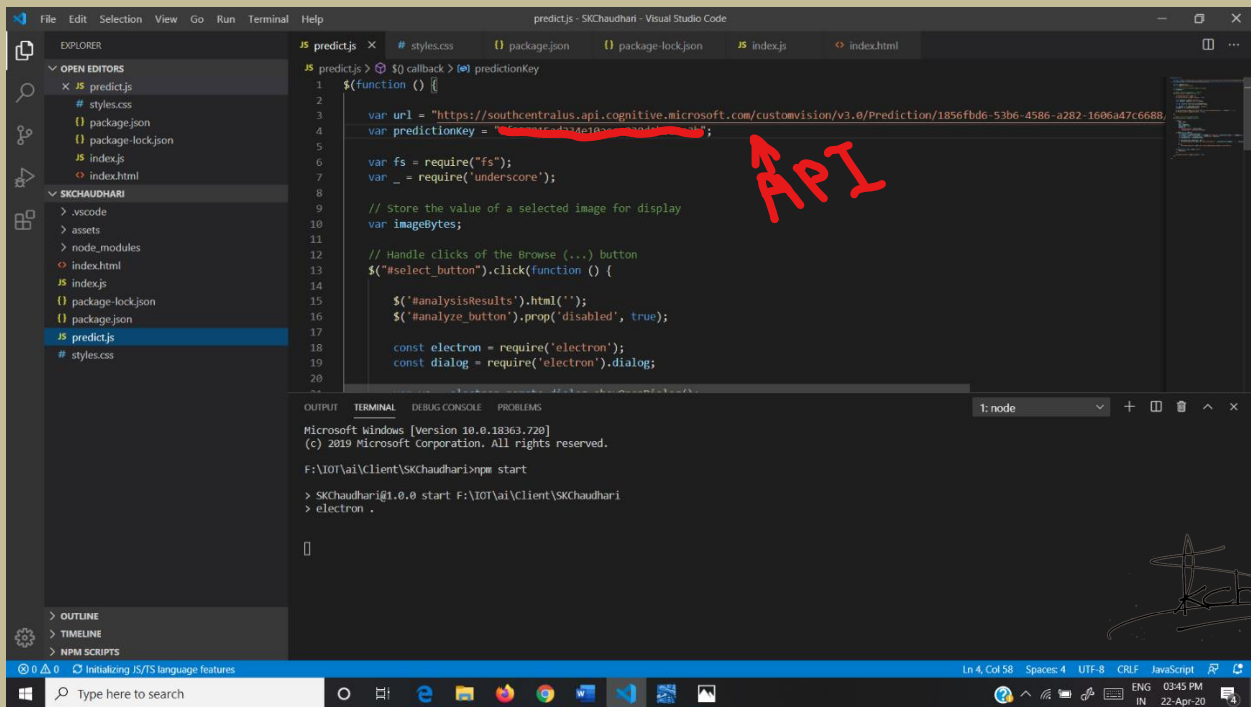
```
package-lock.json - SKChaudhari - Visual Studio Code
1 {
2   "name": "SKChaudhari",
3   "version": "1.0.0",
4   "lockfileVersion": 1,
5   "requires": true,
6   "dependencies": {
7     "ajv": {
8       "version": "6.12.2",
9       "resolved": "https://registry.npmjs.org/ajv/-/ajv-6.12.2.tgz",
10      "integrity": "sha512-kWVzIvR50fL6IwVrjA4ieYz/gj2NnbzXl0ik7XFk7Lj2h3o0kxNpZ19zuaGf7sUyJdINhUa9LQRgj/0jUQ==",
11      "requires": {
12        "fast-deep-equal": "^3.1.1",
13        "fast-json-stable-stringify": "^2.0.0",
14        "json-schema-traverse": "^0.4.1",
15        "uri-js": "^4.2.2"
16      }
17     },
18     "ansi-regex": {
19       "version": "2.1.1",
20       "resolved": "https://registry.npmjs.org/ansi-regex/-/ansi-regex-2.1.1.tgz",
21       "integrity": "sha1-w7M694L3704daztUwJ1eRPRgh2Y="
22     }
23   }
24 }
```

Microsoft Windows [version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

F:\IoT\ai\Client\SKChaudhari>npm start

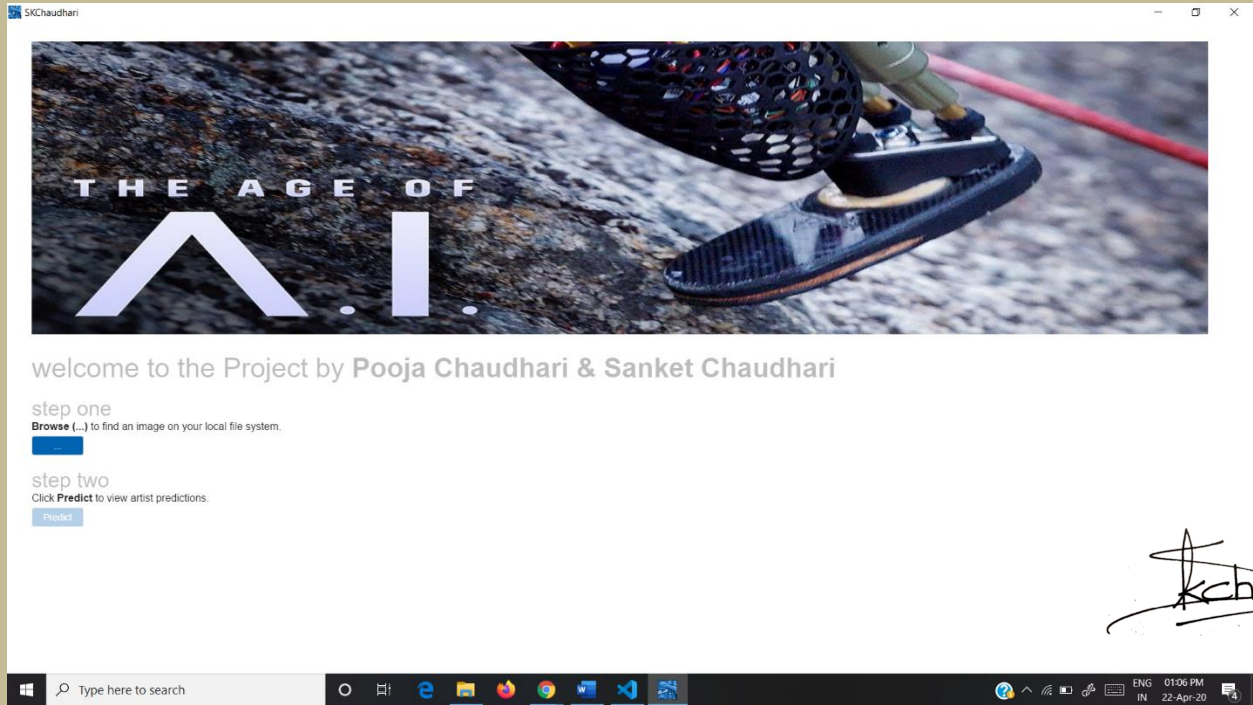
> SKChaudhari@1.0.0 start F:\IoT\ai\Client\SKChaudhari
> electron .

Module 3) connect app to the model using API

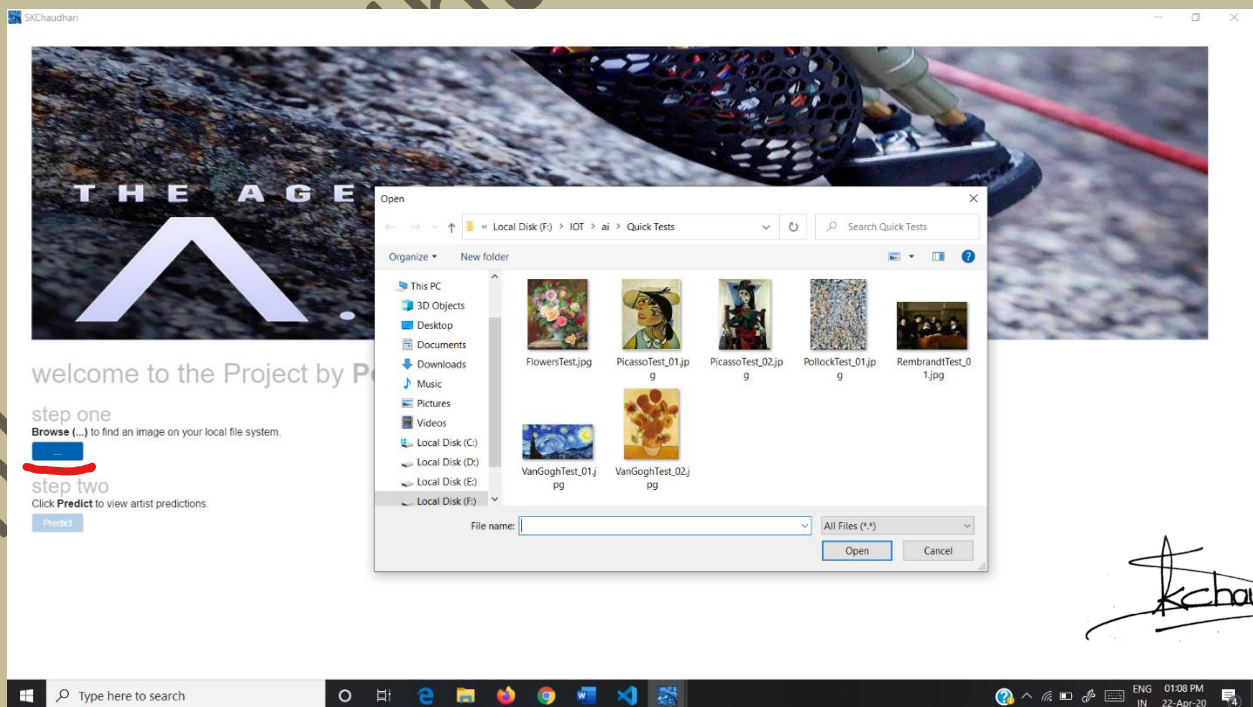


Module 4) How to operate App.

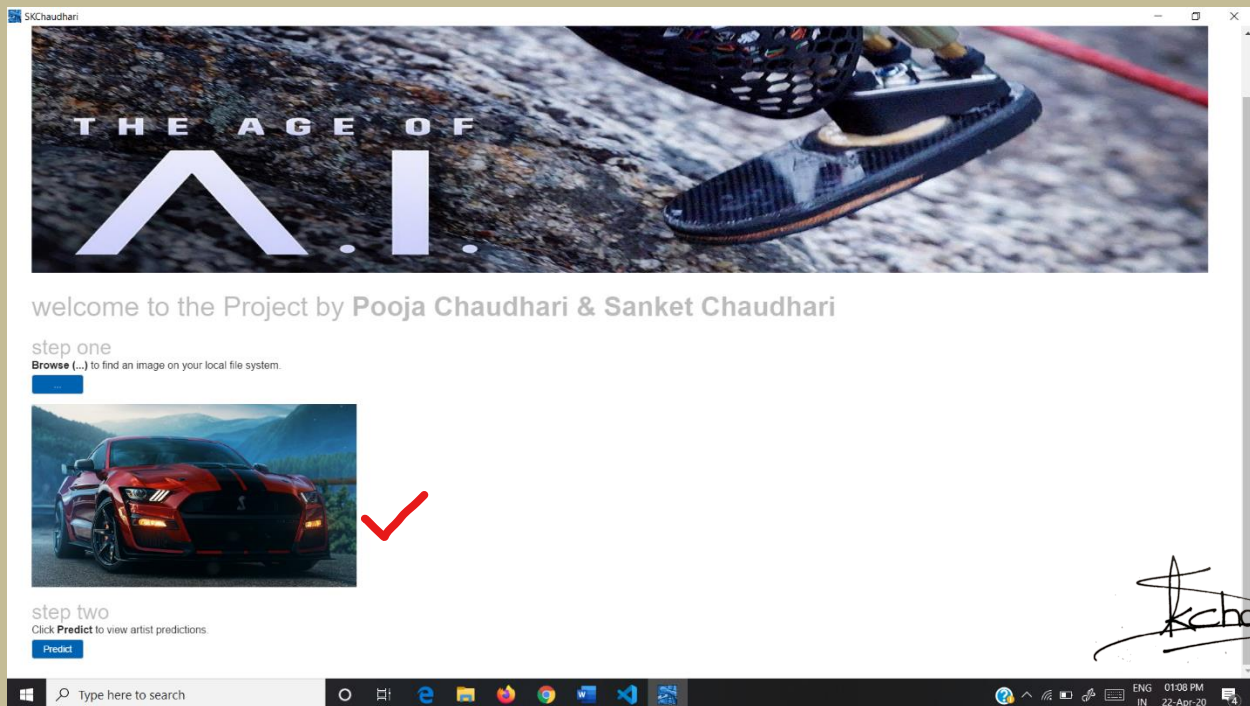
1) Main App.



2) First button is used to upload images from local disk.

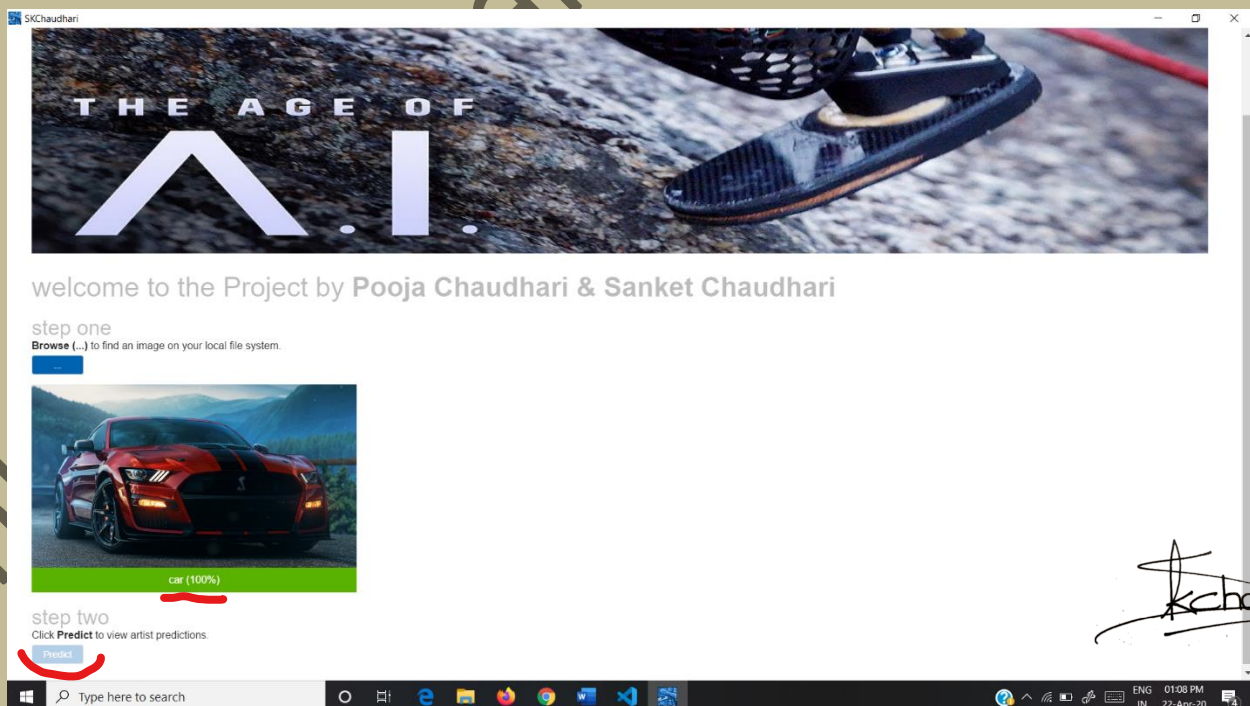


3) After uploading image it will appear on app window.

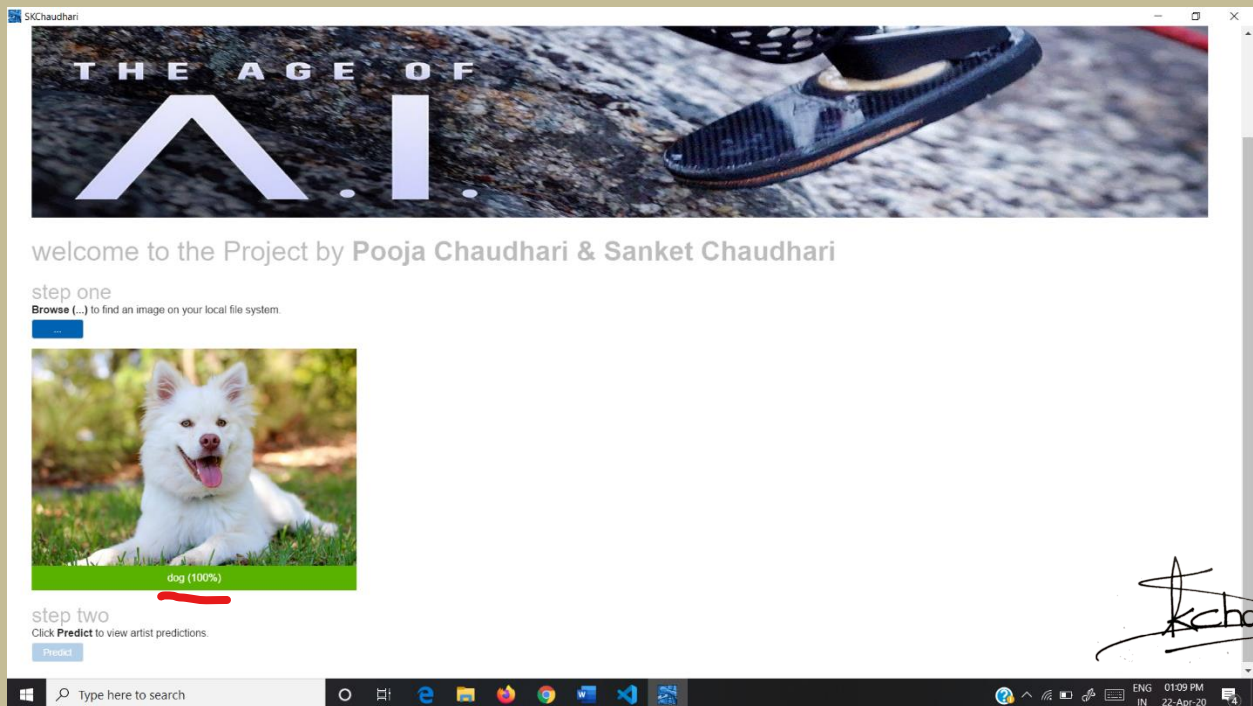


4) After clicking Predict button API will trigger and image is recognized by the model, after completion of process model gives answer.

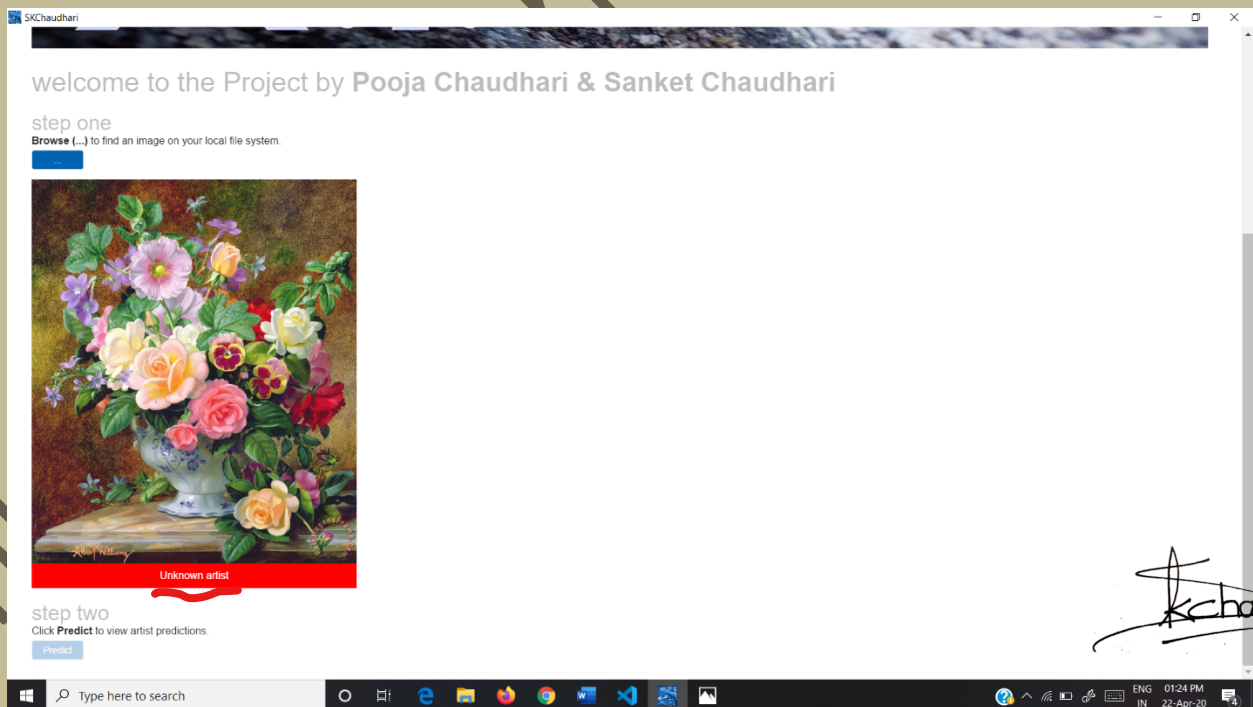
5) In below diagram model said it is a (car 100%).



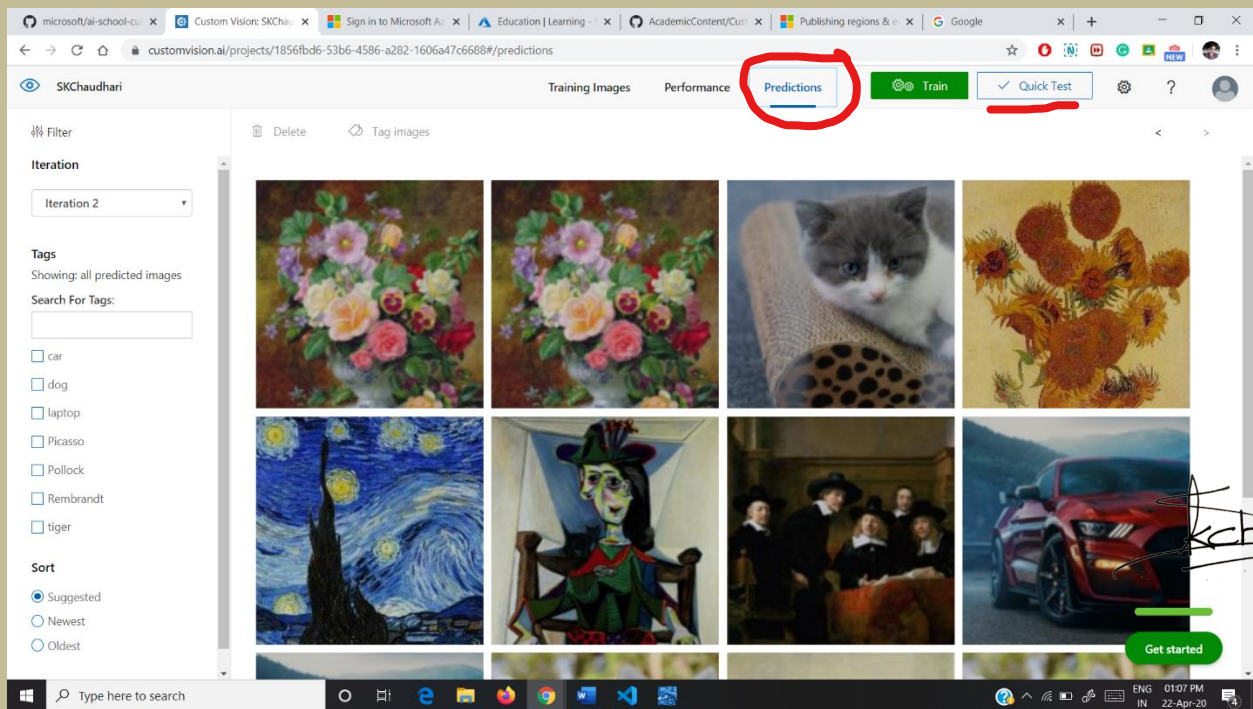
6) Model said it is a (dog 100%).



7) If model is not trained for particular image it will show the particular description like below image.



8) We can see all our prediction in our model.



Sanket Chaudhari & Pooja Chaudhari