



# MINI PROJECT

## (2021-22)

**“Cryptocurrency Screener with Face Lock Security ”**

**Project Report**



**Institute of Engineering & Technology**  
**Submitted By-**

Name	University Roll no.	Section
Shaurya Omar	191500752	L
Shivam Tiwari	191500763	K
Prachi Singh	191500553	L
Ritika Singh	191500664	L
Divyanshu Rai	191500278	M

**Under the Supervision Of**

**Mr. Framanul Haque**

**Technical Trainer**

**Department of Computer Engineering & Applications**



Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,  
Chaumuha, Mathura – 281406 U.P (India)

### **Declaration**

I/we hereby declare that the work which is being presented in the Bachelor of technology. Project “**Cryptocurrency screener with Face Lock Security**”, in partial fulfillment of the requirements for the award of the *Bachelor of Technology* in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of **Mr. Framanul Haque, Technical Trainer, Dept. of CEA, GLA University.**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

**Name of Candidate:** Shivam Tiwari

**Name of Candidate:** Shaurya Omar

**University Roll No.:**191500763

**University Roll No.:**191500752

**Name of Candidate:** Prachi Singh

**Name of Candidate:** Divyanshu  
Rai

**University Roll No.:**191500553

**University Roll No.:**191500278

**Name of Candidate:** Ritika  
Agarwal



**Department of Computer Engineering and Applications**

**GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,**

**Chaumuha, Mathura – 281406 U.P (India)**

## **Certificate**

This is to certify that the project entitled “Cryptocurrency Screener with Face Lock Security”, carried out in Mini Project – I Lab, is a bonafide work by Shaurya Omar, Shivam Tiwari, Divyanshu Rai ,Prachi Singh ,Ritika Varshney and is submitted in partial fulfillment of the requirements for the award of the degree Bachelor of Technology (Computer Science & Engineering).

**Signature of Supervisor:**

**Name of Supervisor:** Mr. Farmanul Haque **Date:**



Department of Computer Engineering and Applications GLA University,  
17 km. Stone NH#2, Mathura-Delhi Road, Chaumuha, Mathura – 281406  
U.P (India)

## ACKNOWLEDGEMENT

Presenting the ascribed project paper report in this very simple and official form, we would like to place my deep gratitude to GLA University for providing us the instructor Mr Farmanul Haque, our technical trainer and supervisor.

He has been helping us since Day 1 in this project. He provided us with the roadmap, the basic guidelines explaining on how to work on the project. He has been conducting regular meeting to check the progress of the project and providing us with the resources related to the project. Without his help, we wouldn't have been able to complete this project.

And at last but not the least we would like to thank our dear parents for helping us to grab this opportunity to get trained and also my colleagues who helped me find resources during the training.

### Thanking You

**Name of Candidate:** Shaurya Omar

**University Roll No.:**191500752

**Name of Candidate:** Shivam Tiwari

**University Roll No.:**191500763

**Name of Candidate:** Divyanshu Rai

**University Roll No.:**191500278

**Name of Candidate:** Prachi Singh

**University Roll No.:**191500553

**Name of Candidate:** Ritika Varshney

## **ABSTRACT**

In this project, we are creating an application, A Cryptocurrency Screener with Face lock security. As we can see that crypto currency industry is now witnessing a boom in India, Indian lawmakers and regulators are now inclining their sentiments toward the adoption of crypto currency , there is an need of platform that can provide user to purchase and trade currencies.

Our application, Cryptoscreener, focuses on showing you all the latest cryptocurrency and altcoins and will show the frequent changes that keep happening when anyone around the world. Adding the face security in this program adds the touch of privacy everyone wants. It will recognize the desired people who are authorized to access the app and will let you in, or will inform you via notification that if there is an intruder trying to sneak in your personal transactions.



# CONTENTS

Cover Page

Declaration

Certificate

Training Certificate

Acknowledgement

Abstract

Content

Introduction

- 1.1 Context
- 1.2 Motivation
- 1.3 Objective
- 1.4 Sources

Software Requirement Analysis

- 2.1 Hardware Requirement
- 2.2 Software Requirements

Chapter 4 Technology Used

- 4.1 Android
- 4.2 Version of Android
- 4.3 Artificial Intelligence
- 4.4 Machine Learning
- 4.5 Deep Learning
- 4.6 Flutter



- 4.3 Tools and Languages

- 4.4 Basic Terminology

Implementation and User Interface

Conclusion

References

# INTRODUCTION

## **1.1 CONTEXT**

This Application “Cryptocurrency Screener with Face Lock Security” has been submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering at GLA University, Mathura supervised by Mr.Farmanul Haque. This project has been completed approximately three months and has been executed in modules, meetings have been organised to check the progress of the work and for instructions and guidelines.

## **1.2 MOTIVATION**

The main motivation for the project is the upcoming youth who is interested in crypto currencies, this app will actually be very helpful for them as this actually tells you the realtime price of any coin or altcoin at any point in your day.

## **1.3 OBJECTIVE**

The main objective of this application is to create a Cryptocurrency Screener with face lock security” Our application, Cryptoscreener, focuses on showing you all the latest cryptocurrency and altcoins and will show the frequent changes that keep happening when anyone around the world. This also comes with a face recognition security using which the people can secure the data they have been looking for.

## **1.4 EXISTING SYSTEM**

Our app can run on two major operating systems- Windows and Android.  
Further devices compatibility will be added with updates in the future.

## **1.5 SOURCES**

The source of our project (including all the project work, documentations and presentations) will be available at the following link <https://github.com/>

# **Software Requirement Analysis**

## 2.1 HARDWARE REQUIREMENTS

- Processor : Intel i3 and above for Windows, Android 8 and above in Mobile
- Operating System : Windows 7 and above
- RAM : 4GB minimum
- Hardware Devices : N/A
- Hard disk : 32GB storage
- Display : N/A

## 2.2 SOFTWARE REQUIREMENTS

- Technology Implemented : AI, ML, Deep Learning, Flutter
- Language : Python, Dart, Used Shell Scripting
- Database : Multiple Images currency and Live Pricing of

- User : Using Flutter
  - User : Google Chrome
- Interface compatible  
Design • Web Browser

## **CHAPTER-4      TECHNOLOGY USED**

### **4.3 ANDROID**

Android is a linux-based operating system designed primarily for touch screen devices such as smart phone tablets and computers. Released in 2008, is now owned by Google. So android is an operating system like Windows, Ubuntu and Mac OS and a lot of devices use Android these days like mobile phones, watches, laptop and television. So we also created an android application “Cryptocscreeener”, a library of e-books. Play Store is a market place for all the Android Apps. So we need to know what basically an android app is. An Android app is software running on an Android Platform. So this can be concluded that like all the

software it is a combination of Backend and Frontend. Backend to design the logical parts of the app, for the functionality whereas Front End to develop the User Interface. And to implement the various parts of the android app, we require a number of tools and technologies which will come into picture. But first it would be great to see the three different type of Android Apps:-

- **Native Apps:** An executable program coded in the machine language of the hardware platform it is running in. **Native applications** are compiled into the machine language of that CPU. For example, **Windows** and Mac executable **apps** are in x86 machine language, while **mobile apps** are ARM based. Native apps are the most common. They're coded in a specific language like Swift for **iOS** or Java for Android. A popular example is WhatsApp.
- **Web Apps:** are accessed via the internet browser and will adapt to whichever device you're viewing them on. They are not native to a particular system, and don't need to be downloaded or installed. Due to their responsive nature, they do indeed look and function a lot like mobile apps — and this is where the confusion arises.
- **Hybrid Apps:** Hybrid apps are deployed in a native container that uses a mobile Web View object. When the app is used, this object displays web content thanks to the use of web technologies (CSS, JavaScript, HTML, HTML5). It is in fact displaying web pages from a desktop website that are adapted to a Web View display. The web content can either be displayed as soon as the app is opened or for

certain parts of the app only i.e. for the purchase funnel. In order to access a device's hardware features (accelerometer, camera, contacts...) for which the

native apps are installed, it is possible to include native elements of each platform's user interfaces (iOS, Android): native code will be used to access the specific features in order to create a seamless user experience. Hybrid apps can also rely on platforms that offer JavaScript APIs if those functionalities are called within a Web View.

#### **4.4 VERSION OF ANDROID.**

#### **4.3ARTIFICIAL INTELLIGENCE :-**

Artificial intelligence (AI) is intelligence demonstrated by machines, as opposed to natural intelligence displayed by animals including humans. Artificial intelligence was founded as an academic discipline in 1956, and in the years since has experienced several waves of optimism, followed by disappointment and the loss of funding (known as an "AI winter"), followed by new approaches, success and renewed funding. AI research has tried and discarded many different approaches since its founding, including simulating the brain, modeling human problem solving, formal logic, large databases of knowledge and imitating animal behavior. AI applications include advanced web

search engines (e.g., Google), recommendation systems (used by YouTube, Amazon and Netflix), understanding human speech (such as Siri and Alexa), self-driving cars (e.g., Tesla), automated decision-making and competing at the highest level in strategic game systems (such as chess and Go).

#### **4.4 MACHINE LEARNING :-**

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, uncovering key insights within data mining projects.

#### **4.5 DEEP LEARNING :-**

Deep learning is a type of machine learning and artificial intelligence (AI) that imitates the way humans gain certain types of knowledge. Deep learning is an important element of data science, which includes statistics and predictive modeling. At its simplest, deep learning can be thought of as a way to automate predictive analytics. While traditional machine learning algorithms are linear, deep learning algorithms are stacked in a hierarchy of increasing complexity and abstraction



## 4.6 FLUTTER

Flutter is a free and open-source mobile UI framework created by Google and released in May 2017. In a few words, it allows you to create a native mobile application with only one codebase. This means that you can use one programming language and one codebase to create two different apps (for iOS and Android).

Flutter consists of two important parts:

**An SDK (Software Development Kit):** A collection of tools that are going to help you develop your applications. This includes tools to compile your code into native machine code (code for iOS and Android).

**A Framework (UI Library based on widgets):** A collection of reusable UI elements (buttons, text inputs, sliders, and so on) that you can personalize for your own needs.

## 4.7 TOOLS AND LANGUAGES

Tools used to build the Android App are:-

Languages used in building an Application are:

- **Dart:** Dart is a programming language designed for client development, such as for the web and [mobile apps](#). It is developed by [Google](#) and can also be used to build server and desktop

applications. Dart is an object-oriented, class-based, garbagecollected language with C-style syntax.[11] Dart can compile to either native code or JavaScript. It supports interfaces, mixins, abstract classes, reified generics, and type inference.

- **Python:** Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

- **Shell Scripting:** A shell script is a computer program designed to be run by the Unix/Linux shell which could be one of the following:
  - 1.The Bourne Shell
  - 2.The C Shell
  - 3.The Korn Shell
  - 4.The GNU Bourne-Again Shell

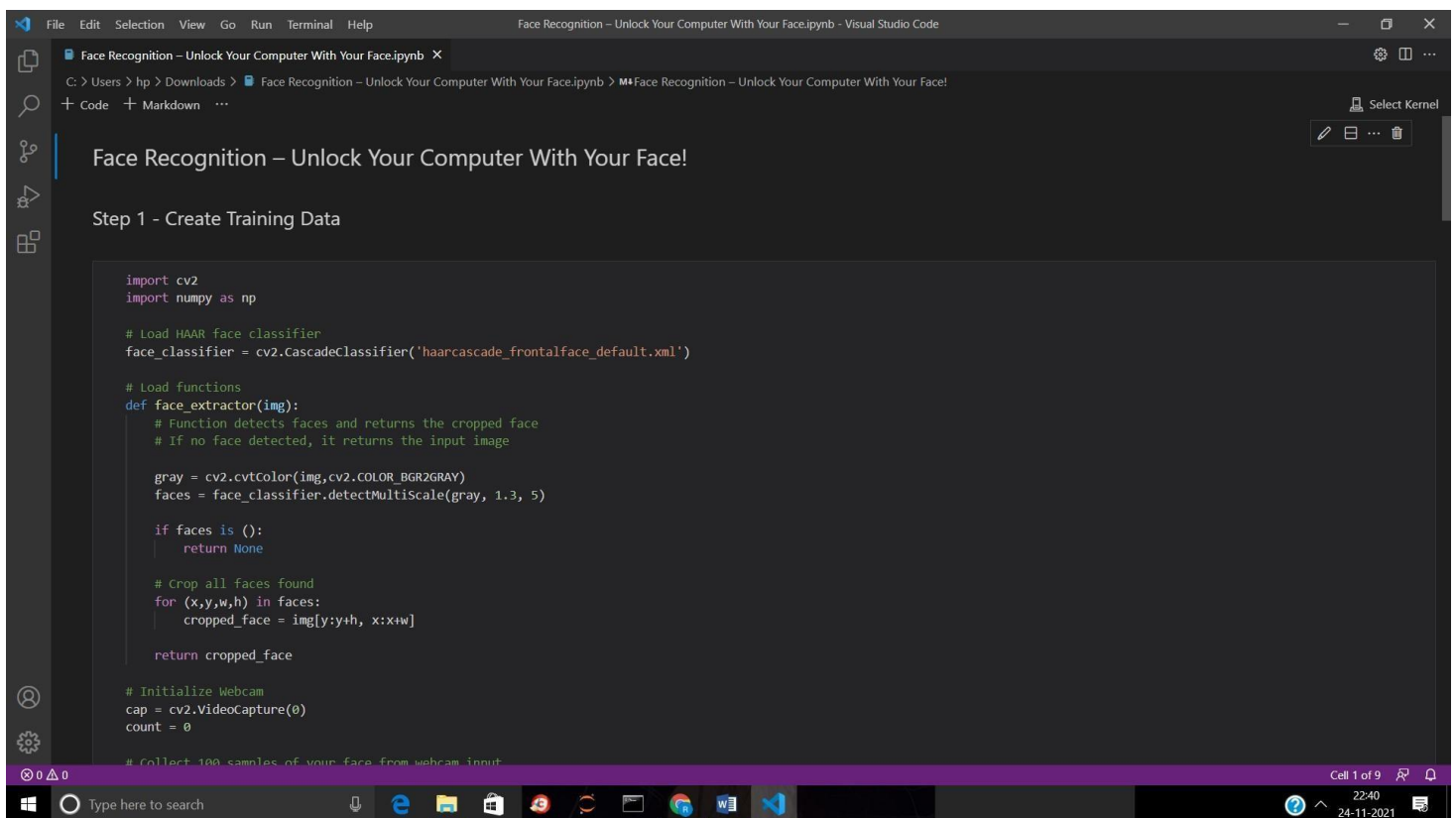
A shell is a command-line interpreter and typical operations performed by shell scripts include file manipulation, program execution, and printing text.

## CHAPTER -5 IMPLEMENTATION AND USER INTERFACE

### 5.2 Implementation of the Cryptoscreener:

#### 5.2.1 Step to be followed to develop the app

#### 5.2.2 Step to be followed by the user.



The screenshot displays a Jupyter Notebook titled "Face Recognition – Unlock Your Computer With Your Face.ipynb" within the Visual Studio Code editor. The notebook is at "Step 1 - Create Training Data". The code in the cell includes imports for cv2 and numpy, loading a Haar cascade classifier, defining a face\_extractor function, and initializing a webcam. The function face\_extractor takes an image, converts it to grayscale, uses the classifier to detect faces, and returns the cropped face if one is found. The notebook interface shows the file explorer on the left, the code editor in the center, and the Jupyter Notebook toolbar on the right. The Windows taskbar is visible at the bottom.

```
import cv2
import numpy as np

# Load HAAR face classifier
face_classifier = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# Load functions
def face_extractor(img):
    # Function detects faces and returns the cropped face
    # If no face detected, it returns the input image

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray, 1.3, 5)

    if faces is ():
        return None

    # Crop all faces found
    for (x,y,w,h) in faces:
        cropped_face = img[y:y+h, x:x+w]

    return cropped_face

# Initialize Webcam
cap = cv2.VideoCapture(0)
count = 0

# Collect 100 samples of your face from webcam input
```

```
File Edit Selection View Go Run Terminal Help Face Recognition - Unlock Your Computer With Your Face.ipynb - Visual Studio Code

Face Recognition - Unlock Your Computer With Your Face.ipynb X
C: > Users > hp > Downloads > Face Recognition - Unlock Your Computer With Your Face.ipynb > M Face Recognition - Unlock Your Computer With Your Face!
+ Code + Markdown ... Select Kernel

# Collect 100 samples of your face from webcam input
while True:

    ret, frame = cap.read()
    if face_extractor(frame) is not None:
        count += 1
        face = cv2.resize(face_extractor(frame), (200, 200))
        face = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)

        # Save file in specified directory with unique name
        file_name_path = 'C:\Users\shaur\OneDrive\Desktop\Kamehameha' + str(count) + '.jpg'
        cv2.imwrite(file_name_path, face)

        # Put count on images and display live count
        cv2.putText(face, str(count), (50, 50), cv2.FONT_HERSHEY_COMPLEX, 1, (0,255,0), 2)
        cv2.imshow('Face Cropper', face)

    else:
        print("Face not found")
        pass

    if cv2.waitKey(1) == 13 or count == 100: #13 is the Enter Key
        break

cap.release()
cv2.destroyAllWindows()
print("Collecting Samples complete")

... File "<ipython-input-1-8a56f248543c>", line 38
file_name_path = 'C:\Users\Shaurya Omar\OneDrive\Desktop\Kamehameha\Faces' + str(count) + '.jpg'
^
SyntaxError: (unicode error) 'unicodeescape' codec can't decode bytes in position 2-3: truncated \UXXXXXXXX escape

Cell 1 of 9 22:42 24-11-2021
```

```
File Edit Selection View Go Run Terminal Help Face Recognition - Unlock Your Computer With Your Face.ipynb - Visual Studio Code

Face Recognition - Unlock Your Computer With Your Face.ipynb X
C: > Users > hp > Downloads > Face Recognition - Unlock Your Computer With Your Face.ipynb > M Face Recognition - Unlock Your Computer With Your Face!
+ Code + Markdown ... Select Kernel

Step 2 - Train Model

cap.release()

Python

import cv2
import numpy as np
from os import listdir
from os.path import isfile, join

# Get the training data we previously made
data_path = 'C:\Users\Shaurya Omar\OneDrive\Desktop\Task pics\Faces'
onlyfiles = [f for f in listdir(data_path) if isfile(join(data_path, f))]

# Create arrays for training data and labels
Training_Data, Labels = [], []

# Open training images in our datapath
# Create a numpy array for training data
for i, files in enumerate(onlyfiles):
    image_path = data_path + onlyfiles[i]
    images = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    Training_Data.append(np.asarray(images, dtype=np.uint8))
    Labels.append(i)

# Create a numpy array for both training data and labels
Labels = np.asarray(Labels, dtype=np.int32)

# Initialize facial recognizer
# model = cv2.face.createLBPHFaceRecognizer()
# NOTE: For OpenCV 3.0 use cv2.face.createLBPHFaceRecognizer()
# pip install opencv-contrib-python
# model = cv2.face.LBPHFaceRecognizer_create()
```

```
File Edit Selection View Go Run Terminal Help
Face Recognition - Unlock Your Computer With Your Face.ipynb - Visual Studio Code

Face Recognition - Unlock Your Computer With Your Face.ipynb X
C: > Users > hp > Downloads > Face Recognition - Unlock Your Computer With Your Face.ipynb > M*Face Recognition - Unlock Your Computer With Your Face!

+ Code + Markdown ...
Select Kernel

# Initialize facial recognizer
# model = cv2.face.createLBPHFaceRecognizer()
# NOTE: For OpenCV 3.0 use cv2.face.createLBPHFaceRecognizer()
# pip install opencv-contrib-python
# model = cv2.createLBPHFaceRecognizer()

vimal_model = cv2.face_LBPHFaceRecognizer.create()
# Let's train our model
vimal_model.train(np.asarray(Training_Data), np.asarray(Labels))
print("Model trained sucessefully")

... Model trained sucessefully

Step 3 - Run Our Facial Recognition
+ Code + Markdown

import cv2
import numpy as np
import os

face_classifier = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

def face_detector(img, size=0.5):

    # Convert image to grayscale
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray, 1.3, 5)
    if faces is ():
        return img, []

for (x,y,w,h) in faces:
    cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,255),2)
    roi = img[y:y+h, x:x+w]
    roi = cv2.resize(roi, (200, 200))
    return img, roi

# Open Webcam
cap = cv2.VideoCapture(0)

while True:

    ret, frame = cap.read()

    image, face = face_detector(frame)

    try:
        face = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)

        # Pass face to prediction model
        # "results" comprises of a tuple containing the label and the confidence value
        results = vimal_model.predict(face)
        # harry_model.predict(face)

        if results[1] < 500:
            confidence = int( 100 * (1 - (results[1])/400) )
            display_string = str(confidence) + '% confident it is User'

            cv2.putText(image, display_string, (100, 120), cv2.FONT_HERSHEY_COMPLEX, 1, (255,120,150), 2)

        if confidence > 90:
            cv2.putText(image, "Hey Vimal", (250, 450), cv2.FONT_HERSHEY_COMPLEX, 1, (0,255,0), 2)
            cv2.imshow("Face Recognition", image )
            # os.system("chrome https://www.google.com/search?q=vimal+daga")
            os.system("wmplayer c:\\w.mp3")

    except:
        pass
```

```
File Edit Selection View Go Run Terminal Help
Face Recognition - Unlock Your Computer With Your Face.ipynb - Visual Studio Code

Face Recognition - Unlock Your Computer With Your Face.ipynb X
C: > Users > hp > Downloads > Face Recognition - Unlock Your Computer With Your Face.ipynb > M*Face Recognition - Unlock Your Computer With Your Face!

+ Code + Markdown ...
Select Kernel

for (x,y,w,h) in faces:
    cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,255),2)
    roi = img[y:y+h, x:x+w]
    roi = cv2.resize(roi, (200, 200))
    return img, roi

# Open Webcam
cap = cv2.VideoCapture(0)

while True:

    ret, frame = cap.read()

    image, face = face_detector(frame)

    try:
        face = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)

        # Pass face to prediction model
        # "results" comprises of a tuple containing the label and the confidence value
        results = vimal_model.predict(face)
        # harry_model.predict(face)

        if results[1] < 500:
            confidence = int( 100 * (1 - (results[1])/400) )
            display_string = str(confidence) + '% confident it is User'

            cv2.putText(image, display_string, (100, 120), cv2.FONT_HERSHEY_COMPLEX, 1, (255,120,150), 2)

        if confidence > 90:
            cv2.putText(image, "Hey Vimal", (250, 450), cv2.FONT_HERSHEY_COMPLEX, 1, (0,255,0), 2)
            cv2.imshow("Face Recognition", image )
            # os.system("chrome https://www.google.com/search?q=vimal+daga")
            os.system("wmplayer c:\\w.mp3")

    except:
        pass
```

FileEditSelectionViewGoRunTerminalHelp

Face Recognition - Unlock Your Computer With Your Face.ipynb - Visual Studio Code

Face Recognition - Unlock Your Computer With Your Face.ipynb X

C:\> Users > hp > Downloads > Face Recognition - Unlock Your Computer With Your Face.ipynb > Face Recognition - Unlock Your Computer With Your Face!

+ Code + Markdown ...

Select Kernel

```
if confidence > 90:
    cv2.putText(image, "Hey Vimal", (250, 450), cv2.FONT_HERSHEY_COMPLEX, 1, (0,255,0), 2)
    cv2.imshow('Face Recognition', image )
    # os.system("chrome https://www.google.com/search?q=vimal+daga")
    os.system("wmplayer c:\\lw.mp3")
    break

else:

    cv2.putText(image, "I dont know, how r u", (250, 450), cv2.FONT_HERSHEY_COMPLEX, 1, (0,0,255), 2)
    cv2.imshow('Face Recognition', image )

except:
    cv2.putText(image, "No Face Found", (220, 120) , cv2.FONT_HERSHEY_COMPLEX, 1, (0,0,255), 2)
    cv2.putText(image, "looking for face", (250, 450), cv2.FONT_HERSHEY_COMPLEX, 1, (0,0,255), 2)
    cv2.imshow('Face Recognition', image )
    pass

if cv2.waitKey(1) == 13: #13 is the Enter Key
    break

cap.release()
cv2.destroyAllWindows()
```

Python

```
... <>:13: SyntaxWarning: "is" with a literal. Did you mean "=="?
<>:13: SyntaxWarning: "is" with a literal. Did you mean "=="?
<ipython-input-9-837c4a3d27d4>:13: SyntaxWarning: "is" with a literal. Did you mean "=="?
if faces is ():
```

Python

Cell 1 of 9 22:47 24-11-2021

Type here to search

## CRYTYSREENER

**Enjin Coin**

enj

**4.62****+0.675463****+17.1374%****Monero**

xmr

**239.15****+4.88****+2.08356%**

The Graph

grt

**0.86684****-0.047123580702****-5.15596%****Helium**

hnt

**42.63****-0.200302071604****-0.46762%****Tezos**

xtz

**4.83****-0.340697428113****-6.58954%**



## CRYTYSREENER

**Fantom**

ftm

**2.43****+0.053769****+2.26171%****OKB**

okb

**23.37****-1.024101936715****-4.19734%****Hedera**

hbar

**0.346759****-0.020027656474****-5.4603%****Near**

near

**9.2****-0.493522737725****-5.09067%****cDAI**

cdai

**0.02173365****-0.000029968386****-0.1377%**







## CRYTYSREENER



FTX Token  
ftt

48.71

**-2.268714694523****-4.45002%**

Theta Network  
theta

6.62

**-0.000333747559****-0.00504%**

Lido Staked Ether  
steth

4203.87

**-101.567373309899****-2.35905%**

Filecoin  
fil

49.15

**-2.14210025932****-4.17619%**

Ethereum Classic  
etc

47.71

**-1.881444600452****-3.79365%**



## CRYTYSREENER



**Bitcoin Cash**  
bch

**586.85**  
**+22.58**  
**+4.00081%**



**Algorand**  
algo

**1.73**  
**-0.045817005707**  
**-2.57837%**



**Uniswap**  
uni

**20.97**  
**-0.972826465207**  
**-4.43276%**



**Axie Infinity**  
axs

**141.73**  
**+6.73**  
**+4.98884%**



**Elrond**  
egld

**460.09**  
**-26.172058945402**  
**-5.38235%**

## CRYTYSREENER



**Decentraland**  
mana

**5.46**  
**+1.5**  
**+37.90904%**



**TerraUSD**  
ust

**1.0**  
**-0.000880038066**  
**-0.08779%**



**TRON**  
trx

**0.09758**  
**-0.005513795206**  
**-5.34833%**



**cETH**  
ceth

**84.93**  
**-1.944585882274**  
**-2.23834%**



**The Sandbox**  
sand

**7.69**  
**+2.41**  
**+45.62066%**



## CRYTYSREENER



**Litecoin**  
ltc

**208.0**

**-4.814564731673**  
**-2.26238%**

Wrapped Bitcoin

**wbt**  
c

**56666.0**

**-806.290569901699**  
**-1.40292%**



**Binance USD**  
busd

**1.0**

**+0.00072364**  
**+0.07239%**



**Chainlink**  
link

**25.67**

**-1.670426781412**  
**-6.10895%**



**Polygon**  
matic

**1.7**

**+0.00404606**  
**+0.23923%**



## CRYTYSREENER



Dogecoin  
doge

**0.216104**

**-0.013045431887**  
**-5.69298%**



Avalanche  
AVA  
X

**117.38**

**-11.895119786765**  
**-9.20126%**



Crypto.com Coin  
cro

**0.950032**

**+0.113557**  
**+13.57568%**



Shiba Inu  
shib

**0.00003672**

**-0.000004947121**  
**-11.8737%**



Terra  
luna

**38.26**

**-4.056661810439**  
**-9.58559%**





## CRYTYSCREENER



Bitcoin  
btc

**56651.0**

**-844.761191174795**  
**-1.46927%**



Ethereum  
eth

**4238.15**

**-101.348342310181**  
**-2.33549%**



Binance Coin  
bnb

**586.68**

**-12.952861530109**  
**-2.16013%**



**Tether**  
usdt

**1.0**

**-0.000079417637**  
**-0.00794%**



Solana  
sol

**207.14**

**-15.135292142003**  
**-6.8093%**

## **CHAPTER -7**

### **CONCLUSION**

This application has been constructed for the betterment of the people interested in the latest day crypto currencies. This project if given proper resources, can actually expand and then can allow crypto transactions within the application as well. Concluding this report, the application has a lot of scope to perform very well in the future.

## REFERENCES

[www.github.com](https://www.github.com)

[www.wikipedia.com](https://www.wikipedia.com)

[www.analyticsvidhya.com](https://www.analyticsvidhya.com)

<https://github.com/Shivam-77/Crytyscreener>