FAKE CURRENCY DETECTION USING IMAGE PROCESSING

A PROJECT REPORT

Submitted by

SRIRAM KARTHICK K (311617104073)

VARUN KUMAR (311617104079)

in partial fulfilment for the award of

degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

MISRIMAL NAVAJEE MUNOTH JAIN ENGINEERING

COLLEGE THORAIPAKKAM, CHENNAI-600097

ANNA UNIVERSITY: CHENNAI 600025

APRIL 2023

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report

"FAKE CURRENCY DETECTION USING IMAGE PROCESSING

" is the bonafide work of

"SRIRAM KARTHICK K , VARUN KUMAR"

who carried out the project work under my supervision.

SIGNATURE

Dr. N.Savaranan B.E., ME., Ph.D

HEAD OF THE DEPARTMENT

Department of Computer

Science and Engineering

Misrimal Navajee Munoth

Jain Engineering College,

Thoraipakkam,

Chennai-600097

SIGNATURE

Mr. V.Vinoth Kumar B.E , M.E

SUPERVISOR

ASSISTANT PROFESSOR

Department of Computer

Science and Engineering

Misrimal Navajee Munoth

Jain Engineering College,

Thoraipakkam,

Chennai-600097

Submitted for the Project Viva-Voce Examination held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our sincere gratitude to our honorable **Secretary** (Administration) **Dr. Harish L Metha** and Secretary (Academic) **Shri L. Jaswant S Munoth** for providing the infrastructure facilities to do this project during our course period.

We thank our **Principal**, **Dr. C. C. Christopher** for his support and motivation for the development and completion of this project.

We express profound sense of gratitude and thanks to our **Head Of The Department**, **Dr.N. Saravanan** for his valuable suggestions and guidance for the development and completion of this project.

We thank our project coordinator, **Mrs. D. Yashwanthini, Associate Professor**, for her valuable suggestions and constant encouragement that led us to the successful completion of our project.

We thank our supervisor **Mr. V. Vinoth Kumar Assistant Professor** for his constant help and valuable suggestions.

We thank all the Teaching and Non-Teaching Staff members of our Department who helped us to complete our project.

Above all we thank the Almighty, and our parents and for their constant support and encouragement for completing this project

ABSTRACT

Fake currency detection using image processing and CNN involves using Machine learning techniques to differentiate between real and counterfeit banknotes. The process includes acquiring an image of the banknote and applying various image processing methods to extract relevant features that differentiate genuine from fake banknotes. The extracted features are then used to train a CNN model, which can accurately classify new banknotes as genuine or fake. A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data. The model learns to identify patterns and features in the banknote images to make predictions based on these features. The accuracy of the model depends on the quality of input images, the effectiveness of image processing techniques, and the complexity of the CNN architecture. This technique can help prevent financial crimes and maintain the monetary system's integrity. Name of the dataset is “BANKNOTES”, In the dataset we can collect the dataset of 100 and 50 notes of fake currency and Real currency notes around 8000. In the dataset we can collect the dataset of 100 and 50 notes of fake currency and Real currency notes around 8000. The accuracy which we have achieved is 85%

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

S.NO	ABBREVIATIONS	EXPANSION
1	**ML**	MACHINE LEARNING
2	**TP**	TRUE POSITIVE
3	**TN**	TRUE NEGATIVE
4	**FP**	FALSE POSITIVE
5	**FN**	FALSE NEGATIVE
6	**CNN**	CONVOLUTIONAL NEURAL NETWORK
8	**ReLU**	Rectified Linear Unit

CHAPTER 1
INTRODUCTION

- A fake currency detection using machine learning algorithms and image processing techniques. The objective of this study is to develop an efficient and accurate method to detect counterfeit banknotes using Convolutional Neural Network (CNN) and image processing. The proposed system was trained and tested on a dataset of 400 banknote images captured using mobile phones, with a resolution of 512*720 pixels.
- Fake currency is a significant problem worldwide, and it is essential to detect counterfeit banknotes as early as possible to prevent financial losses. With the advancements in machine learning and computer vision, automated detection of counterfeit currency has become feasible. In this paper, we explore the use of deep learning techniques to classify banknotes as genuine or fake.
- The proposed system utilizes a Convolutional Neural Network, which is a deep learning algorithm commonly used in image classification tasks. In addition, we employed various image processing techniques such as image pre-processing, feature extraction, and dimensionality reduction to enhance the performance of the system.
- The proposed method achieved an accuracy of 85%, which outperforms existing methods for fake currency detection. We also conducted a comparative analysis of different deep learning models and evaluated the performance of our system on a real-time dataset.
- The proposed method provides a promising solution for detecting counterfeit banknotes using machine learning algorithms and image processing techniques. This approach can be further enhanced with a larger and more diverse dataset, which can lead to improved accuracy and

robustness of the system.

- Deep learning is a subset of machine learning that involves artificial neural networks with multiple layers. It is inspired by the structure and function of the human brain, and uses a combination of input data and algorithms to learn and make predictions. Deep learning is used for a wide range of applications, including computer vision, natural language processing, speech recognition, and more. The training process involves feeding large amounts of data into the neural network, adjusting the weights and biases of the nodes in the network, and gradually improving the accuracy of predictions.
- Image processing is a field of study that involves using algorithms and mathematical techniques to manipulate digital images. Common techniques used in digital image processing include image enhancement, image restoration, image compression, and image segmentation. Image processing is widely used in fields such as medical imaging, satellite imaging, computer vision, and remote sensing. The goal of image processing is to extract useful information from images, which can then be used for analysis, interpretation, and decision-making.
- CNNs are a type of neural network commonly used in computer vision tasks such as image and video recognition. They are designed to process and analyze data with a grid-like structure, such as images, by performing convolution operations. The convolution operation involves sliding a filter over the input data and computing dot products to extract features.

CHAPTER 2
LITERATURE SURVEY

Various papers are available that contain information on Fake currency detection. Some referred papers are mentioned here.

Aman Bhatia, Vansh Kedia, Anshul Shroff, Mayand Kumar, Bickey Kumar Shah, Aryan [1] In the past studies the data collected for the fake note detection was with professional cameras but in those data, accuracy seen was to be fair and good due to simple machine learning algorithms. K nearest neighbor algorithms were used traditionally for the detection of fake notes. Systems were getting slower when the data size became large. After that system came across to classify the precision and recognition rate with some enhancement in Machine learning
algorithms and deep learning concepts Due to high and large data sets, data sets were getting distorted, and the precision was not effective a lot though it was 98%. All of these detections were carried out earlier only with open cv and python but
time and again with modern deep learning techniques data were collected with the count of 100 images per denomination and then measured. Accuracy of
training and testing sets were measured. This brings the chain type efficiency that elongates to a larger value in comparison to other techniques.

Eshita Pilania, Bhavika Arora [4], as mentioned, no one can be 100 percent sure of the manual recognition and so the system was proposed to compare images of currency with the stored data and detect whether the currency is fake or genuine. This system used MATLAB to run and perform the operations of the

system. The feature extraction proc ess mostly focuses on HSV values of the currency where the image is divided into blocks and the operations are performed on the ROI.

P. Julia Grace, Ph.D., A. Sheema [5], the survey paper proposes a system to improve the currency detection system especially in commercial areas like banks, shopping malls, etc. Here some different pre-processing techniques were mentioned such as Radiometric corrections and Geometric corrections for correcting spectral errors or distortions due to sensor-Earth geometric variations etc. Different papers were compared and results were provided based on the 3accuracy rate obtained by using different methods.

Komal Vora, Ami Shah, Jay Mehta [6], a system is proposed to detect fake currency based on different features that can be extracted for comparison. Various methods are used at different stages histogram equalization, using feature vectors to stored extracted features, etc. The features that were used for currency detection were security thread, RBI micro-print and serial number detection.

R.Mirza and V.Nanda[7] The research on a currency verification system based on image processing and the extraction of features was carried out. This research was performed on Indian currency. Image Acquisition, Grayscale conversion, Edge Detection, Image Segmentation Characteristic Extraction, and Comparison are the six steps of the system. To perform a comparison between the actual and counterfeit notes, edge detection and picture segmentation were applied.

CHAPTER 3
SYSTEM OVERVIEW

3.1 EXISTING SYSTEM

3.1.1. OVERVIEW

- In the existing system, different traditional strategies and methods are available for fake currency identification based on the colors, width, and serial numbers mentioned.
- Processing these attributes by using digital technologies will give high false positive results leading to lower accuracy of the technology.
- In this system First, to classify the nationality to use certain predefined rules. Areas of significance, and then derive the denomination value.
- Using features such as scale, color, or script on a note, based on how distinctive the notes are in the same region.

3.1.2. DRAWBACK

- The main disadvantage of existing solutions for fake currency detection based on attributes **like colors, width, and serial numbers is their high false positive rates**, which can reduce the accuracy of the technology.
- The reliance on **predefined rules and handcrafted features** can also **limit their ability to detect subtle differences between genuine and counterfeit banknotes**, and they may not be effective in detecting sophisticated counterfeit banknotes that closely resemble genuine banknotes.

3.2. PROPOSED SYSTEM

3.2.1. OVERVIEW

- Our proposed solution/system is to use Convolutional Neural Network (CNN).
- A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data.
- Convolutional Neural Network is trained considering three train test ratio **80:20, 70:30** and **60:40** and measured their performance on the basis various quantitative analysis parameter like Precision, Accuracy, Recall, MCC, F1-Score and others.

3.2.2.ADVANTAGES

- **Local feature extraction**
  - CNNs is a model which is designed to perform local feature extraction by applying a set of filters or convolutional kernels on the input image.
  - This helps in capturing the spatial relationships between different pixels in an image and identifying local patterns or features that are important for detecting fake currency.
- **High accuracy**
  - CNNs have been shown to achieve high accuracy in various image classification tasks, including fake currency detection.
  - They can learn to differentiate between genuine and fake notes based on subtle differences in texture, color, and other features.

3.3. REQUIREMENT ANALYSIS

The requirement specification is a technical specification of requirements for the software products. The purpose of the software requirement specification is to provide a detailed overview of the software project, its parameter and goal. It describes the project target audience and its user interface, hardware and software requirements.

3.3.1 SOFTWARE REQUIREMENT

The software requirements give a detailed description of the system and all its features.

- Python
- Jupyter Notebook
- Visual Studio Code
- Nodejs
- Android Studio

3.3.2 HARDWARE REQUIREMENT

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete engineer as the starting point for the system design.

Ram	:	8GB Ram or more
Processor	:	Any Intel Processor
GPU	:	4GB or more
Hard Disk	:	10GB or more
Speed	:	1.4GHZ or more

3.4 TECHNOLOGIES USED

3.4.1 TENSORFLOW

- TensorFlow is an open-source platform for machine learning and artificial intelligence developed by Google. It is designed to help developers create and deploy machine learning models in a scalable and efficient way. One of the key technologies used in TensorFlow is the computational graph. This is a directed graph that represents the mathematical operations of a machine learning model. TensorFlow also uses automatic differentiation to optimize the parameters of the model during training. Other important technologies used in TensorFlow include:

- TensorBoard, a web-based visualization tool that helps developers monitor and debug their models
- TensorFlow Lite, a lightweight version of TensorFlow that is designed to run on mobile and embedded devices
- TensorFlow.js, a JavaScript library for training and deploying machine learning models in the browser or on Node.js
- TensorFlow Extended (TFX), a platform for building and deploying production-ready machine learning pipelines
- Keras, a high-level API for building neural networks that is integrated with TensorFlow
- Distributed TensorFlow, which allows developers to train models on multiple machines in parallel
- TensorFlow Probability, a library for building probabilistic machine learning models
- TensorFlow Federated, a framework for training machine learning models on decentralized data sources

- TensorFlow Hub, a repository of pre-trained machine learning models that can be used for transfer learning.

3.4.2. OPENCV

- OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.
- The first OpenCV version was 1.0. OpenCV is released under a BSD license and hence it's free for both **academic** and **commercial** use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. When OpenCV was designed the main focus was real-time applications for computational efficiency. All things are written in optimized C/C++ to take advantage of multi-core processing.
- OpenCV is written in the programming language C++, as is its primary interface, but it still retains a less comprehensive though extensive older C interface. All newer developments and algorithms appear in the C++ interface. There are language bindings in Python, Java, and MATLAB/Octave. The application programming interface (API) for these interfaces can be found in the online documentation. Wrapper libraries in several languages have been developed to encourage adoption by a wider audience. In version 3.4, JavaScript bindings for a selected subset of OpenCV functions were released as OpenCV.js, to be used for web platforms.

3.4.3. EXPRESS

- Express.js is a small framework that works on top of Node.js web server functionality to simplify its APIs and add helpful new features. It makes it easier to organize your application's functionality with middleware and routing. It adds helpful utilities to Node.js HTTP objects and facilitates the rendering of dynamic HTTP objects.
- Express is a fast, assertive, essential and moderate web framework of Node.js. You can assume express as a layer built on the top of the Node.js that helps manage a server and routes. It provides a robust set of features to develop web and mobile applications.
- It can be used to design single-page, multi-page and hybrid web applications.
- It allows to setup middlewares to respond to HTTP Requests.
- It defines a routing table which is used to perform different actions based on HTTP method and URL.
- It allows to dynamically render HTML Pages based on passing arguments to template.

3.4.4. NUMPY

- NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python.
- In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster

than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

- NumPy arrays are stored at one continuous place in memory unlike lists, so processes can access and manipulate them very efficiently. This behavior is called locality of reference in computer science. This is the main reason why NumPy is faster than lists. Also it is optimized to work with latest CPU architectures. NumPy is a Python library and is written partially in Python, but most of the parts that require fast computation are written in C or C++.
- NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents.
- NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops, using NumPy.

- Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars.
- The core functionality of NumPy is its "ndarray", for n-dimensional array, data structure. These arrays are strided views on memory. In contrast to Python's built-in list data structure, these arrays are homogeneously typed: all elements of a single array must be of the same type.

3.4.5. ANDROID STUDIO

- Android Studio is a powerful and sophisticated development environment, designed with the specific purpose of developing, testing, and packaging Android applications. It can be downloaded, along with the Android SDK, as a single package. It is a collection of tools and components. Many such tools are installed and updated independently of each other.
- Android Studio is not the only way to develop Android apps; there are other IDEs, such as Eclipse and NetBeans, and it is even possible to develop a complete app using nothing more than Notepad and the command line.
- Built for a purpose, Android Studio has attracted a growing number of third-party plugins that provide a large array of valuable functions, not available directly via the IDE. These include plugins to speed up build times, debug a project over Wi-Fi, and many more.
- Every developer has deadlines to meet, and getting to grips with unfamiliar software can slow them down considerably at first. But Android studio is the official IDE for Android studio and every android app developer should be wary of the differences between the two so that they can figure out the similarities and the differences, and see what works for them.
- Code completion and refactoring: The way that Android Studio intelligently completes code as you type makes it a delight to use. It regularly anticipates what you are about to type, and often a whole line of code can be entered with no more than two or three keystrokes.
- Emulation: Studio comes equipped with a flexible virtual device editor,

allowing developers to create device emulators to model any number of real-world devices.

- Build tools: Android Studio employs the Gradle build system, which performs the same functions as the Apache Ant system that many Java developers will be familiar with.
- Android Studio also provides an amazing time-saving device in the form of Instant Run. This feature cleverly only builds the part of a project that has been edited, meaning that developers can test small changes to code without having to wait for a complete build to be performed for each test. This feature can bring waiting time down from minutes to almost zero.

CHAPTER 4
SYSTEM DESIGN

4.1. OVERVIEW OF ARCHITECTURE:

The diagram shows the overall architecture of the project which consists of the web pages and server block. Each block shows their structural and functional components of the project.

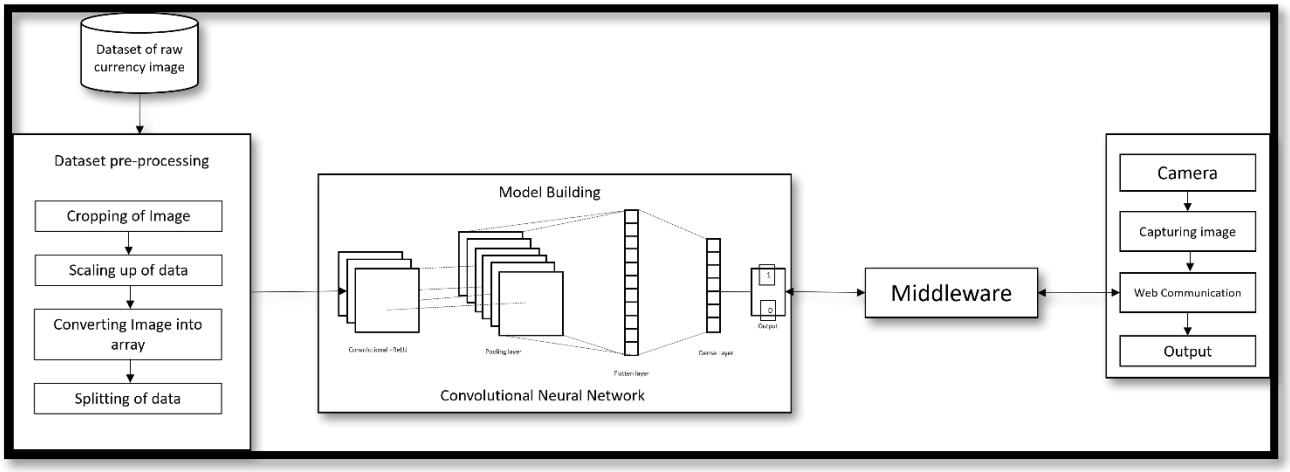


Figure 4.1. Overview of Architecture

4.2. USECASE DIAGRAM

This diagram shows the user such as actor, system and the role of developer in this project. This behaviour diagram models the functionality of the system using use cases.

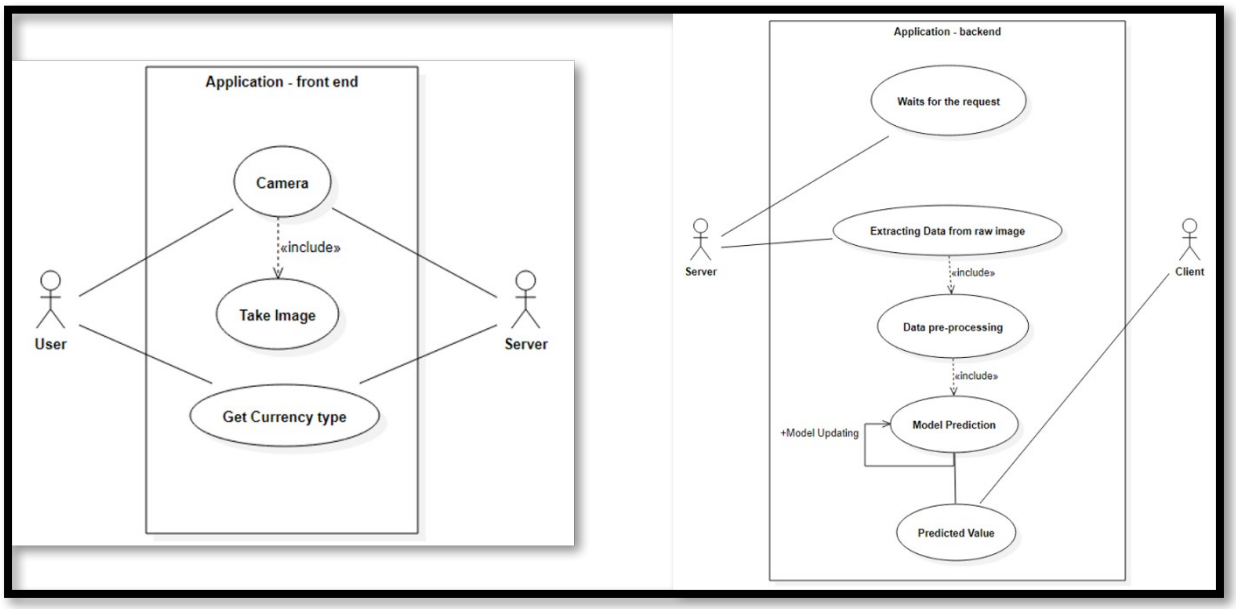


Figure 4.2. Use Case Diagram

4.3.CLASS DIAGRAM

This is the final step which is to de ploy the application in the web.

The structure of the application id described in the class diagram by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. Each class has their attributes and the operations defined with the relationship between the classes.

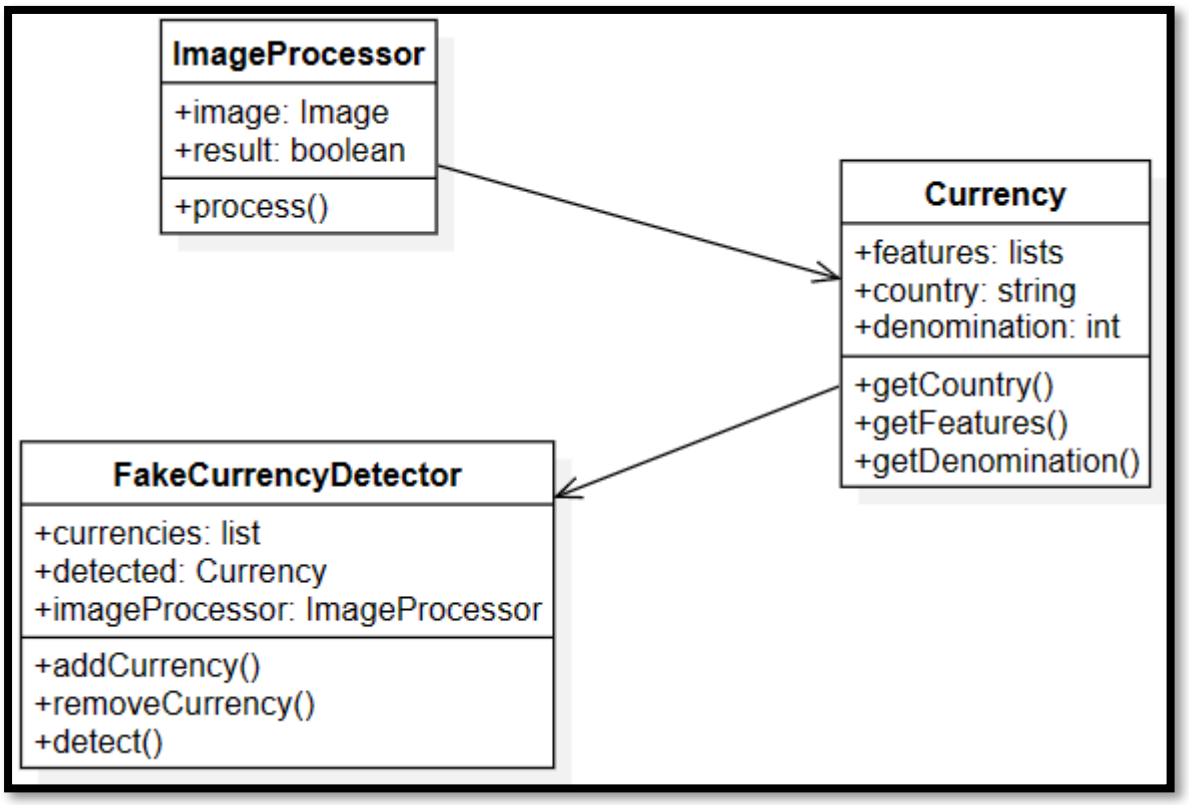


Figure 4.3 Class Diagram

4.4. ACTIVITY DIAGRAM

This module diagram represents the flow from one activity to another activity. The activity can be described as an operation of the system. Some activities are based on conditions satisfied by the actor/object.

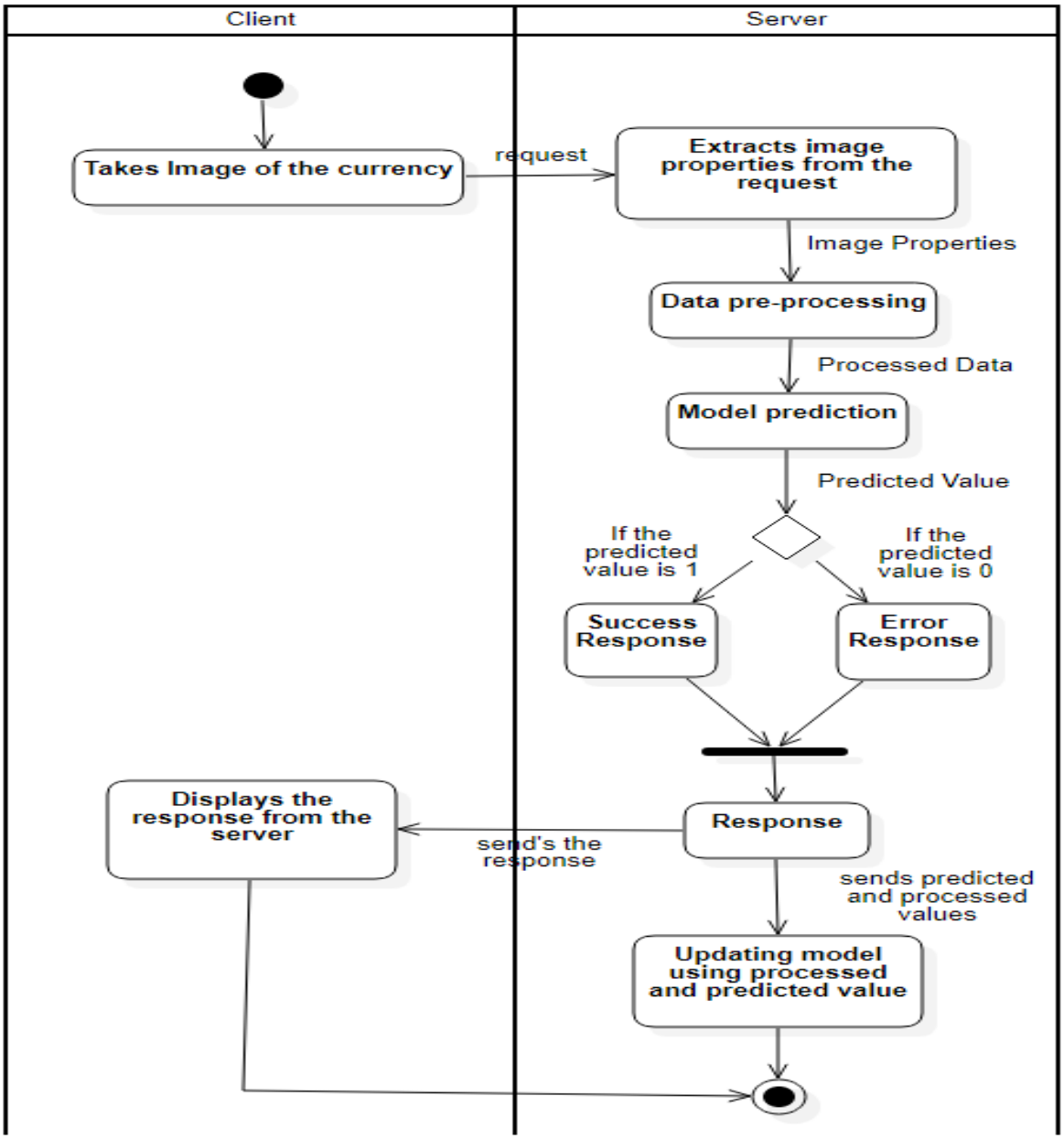


Figure 4.4 Activity Diagram

CHAPTER 5
IMPLEMENTATION

5.1. MODULES

- DATA COLLECTION
- DATA PREPROCESSING
- MODEL BUILDING
- TRAINING AND TESTING
- APP BUIDING
- DEPLOYMENT THE APP

5.1.1 DATA COLLECTION

- Name of the dataset is “BANKNOTES”, In the dataset we can collect the dataset of 100 and 50 notes of fake currency and Real currency notes around 8000.
- This is the first step Ensuring an unbiased model requires the first step of creating a quality dataset, where the quantity alone is not sufficient. Inducing bias can occur easily during data collection, which highlights the importance of selecting relevant features from the raw data. Using irrelevant or redundant features can lead to poor machine learning model performance, further emphasizing the significance of feature selection.
- There are various techniques available for feature selection, including correlation analysis, mutual information, and recursive feature elimination. Correlation analysis can help identify the relationship between features, where those with high correlation can be removed. Mutual information measures the amount of information that one feature provides about another, allowing for the selection of features that provide the most information.
- Techniques such as correlation analysis, mutual information, and recursive feature elimination can be used for feature selection.

- Recursive feature elimination involves ranking features based on their importance, where the least important features are removed iteratively. This technique can help identify features that may not be obvious from correlation analysis or mutual information.

5.1.2 DATA PREPROCESSING

- Data preprocessing is a crucial step that involves cleaning, normalizing, and transforming raw data to make it suitable for analysis. This process aims to remove redundancy, fill in missing values, and address inconsistencies and outliers.
- The main techniques used in data preprocessing include data cleaning, data transformation, feature engineering, cropping, scaling, converting, and splitting the data.
- Cropping the image to a specific size, such as 512x720 pixels, can help reduce the size of the image and improve processing efficiency.
- Scaling the data ensures that all features are on the same scale and helps prevent some features from being weighted more than others.
- Converting images into arrays is essential for machine learning models to understand them. The pixels of the images are converted into numerical values that can be analysed by the model.
- Splitting the data into training, validation, and testing sets is crucial to evaluate the performance of the model accurately. The training set is used to train the model, the validation set is used to optimize the model, and the testing set is used to assess the model's accuracy on unseen data.
- Overall, data pre-processing is essential to ensure the accuracy and reliability of the machine learning model. Techniques such as cropping, scaling, converting, and splitting the data are critical components that contribute to successful model training and evaluation.

5.1.3 MODEL BUILDING

- In this module the main part is choosing the appropriate algorithm for finding the fake currency from the properties of the image which is given.
- For our project we have concluded to use Convolutional Neural Network detect the fake currency, as CNN algorithm keep adding new data to the dataset, the prediction is adjusted without having to retrain a new model. So It is easy to keep the model up to date.
- A CNN consists of multiple layers, with each layer performing a specific type of computation on the input data. The basic building blocks of a CNN are convolutional layers, which use filters to scan the input image and detect features, such as edges and textures.
- During the convolution operation, the filter slides over the input image, and a dot product is computed between the filter and the pixels in the image. This produces a feature map, which highlights the presence of the feature in the input image. Multiple filters can be applied to the same input image, producing multiple feature maps that capture different types of features.
- After the convolutional layer, the output is typically passed through an activation function, such as ReLU, which helps to introduce non-linearity into the model. The activation function takes the output of the convolutional layer and applies a threshold, setting all negative values to zero and passing through positive values.
- Pooling layers are another important component of CNNs, which help to reduce the spatial dimensions of the feature maps while retaining the most salient features. Pooling is achieved by applying a pooling operation, such as max pooling or average pooling, which down samples the input image by taking the maximum or average value of the pixels in a certain region.

- CNNs often include fully connected layers at the end of the network, which perform classification or regression on the feature maps produced by the convolutional and pooling layers. The fully connected layers take the flattened feature maps and pass them through a series of densely connected layers, with each layer performing a linear transformation followed by an activation function.
- A Flatten layer in a CNN (Convolutional Neural Network) model is a layer that takes the output of the previous layer, which is typically a 3D tensor representing the feature maps, and flattens it into a 1D vector. This vector is then passed as input to a dense layer, which is a fully connected neural network layer that performs the classification task.
- The Flatten layer is necessary because the output of the previous layer, which is a tensor of feature maps, is not suitable for feeding into a dense layer, which expects a 1D vector as input. The Flatten layer simply reshapes the tensor into a vector without changing its contents.
- After the Flatten layer, one or more Dense layers can be added to the model. A Dense layer is a fully connected neural network layer where each neuron is connected to every neuron in the previous layer. The output of the Dense layer is typically passed through an activation function, such as the Rectified Linear Unit (ReLU), to introduce non-linearity into the model.
- The number of neurons in the Dense layer and the number of Dense layers in the model can vary depending on the complexity of the problem being solved. In classification problems, the number of neurons in the final Dense layer should be equal to the number of classes in the dataset. The output of the final Dense layer is passed through a softmax activation function to obtain the probability distribution over the classes.

- The Flatten layer in a CNN model is used to convert the output of the previous layer into a 1D vector that can be fed into a Dense layer. The Dense layer is a fully connected neural network layer that performs the classification task.

5.1.4 TRAINING AND TESTING

- Model training is an iterative process that involves repeatedly training the model with the preprocessed and cleaned data until it stabilizes and produces accurate results. The accuracy of the model is evaluated using testing, which is the process of assessing the model's performance on a separate dataset that was not used during training. This helps to obtain an unbiased estimate of the model's performance.
- The testing process should be performed on a dataset that is representative of the real-world data that the model will encounter during deployment. This can be achieved by using a validation set during training to fine-tune the model's hyperparameters and evaluate its performance on a separate dataset. Cross-validation techniques can also be used to obtain a more robust estimate of the model's performance.
- Model training and testing are iterative processes that involve cleaning and preprocessing the data, training the model, and evaluating its performance on a separate dataset. It is important to normalize the data properly and carefully examine each step in the pipeline to avoid introducing bias. The testing process should be performed on a representative dataset, and cross-validation techniques can be used to obtain a more robust estimate of the model's performance.

5.1.5 APP BUILDING

- In the deployment stage, the model which has been built is integrated with a mobile application. This is achieved by developing a backend server using Node.js which serves as an intermediary between the mobile application and the machine learning model. The backend server exposes a RESTful API that allows the mobile application to communicate with the model.
- The mobile application is developed using Android Studio, which is an Integrated Development Environment (IDE) for Android app development. The application is tested for any errors that may occur due to
- communication failure between the mobile application and the backend server.
- Once the mobile application is deployed, users can interact with the machine learning model by making requests through the mobile application. The backend server processes these requests, sends them to the machine learning model, and returns the results to the mobile application. The mobile application can then display the results to the user.
- It is important to test the mobile application thoroughly to ensure that it is reliable and performs well under different conditions. This includes testing for various scenarios such as poor network connectivity, high user load, and unexpected errors.

5.1.6 DEPLOY THE APPLICATION

- This is the final step which is to deploy the application in the web.
- The deployment stage involves integrating the machine learning model with a mobile application through a backend server developed using Node.js. The mobile application is developed using Android Studio and is thoroughly tested for reliability and performance.

CHAPTER 6
RESULT AND ANALYSIS

6.1. RESULT:

FACI – an app which identifies fake currencies is basically designed to identify the currencies which are fake . Fake currency is the major problem these days and to provide solution of that , FACI was developed. Based on the previous analysis which was developed using KNN machine learning model, the accuracy obtained was 67% over 1000 datasets. Though fair accuracy was achieved, to improve its accuracy over large datasets was the aim and so CNN model is used in this project. By implementing CNN model , the accuracy achieved is 85% and it was done with 20K datasets. The test accuracy is fairly good and the loss is minimum. The given figure shows the graph of accuracy and loss.

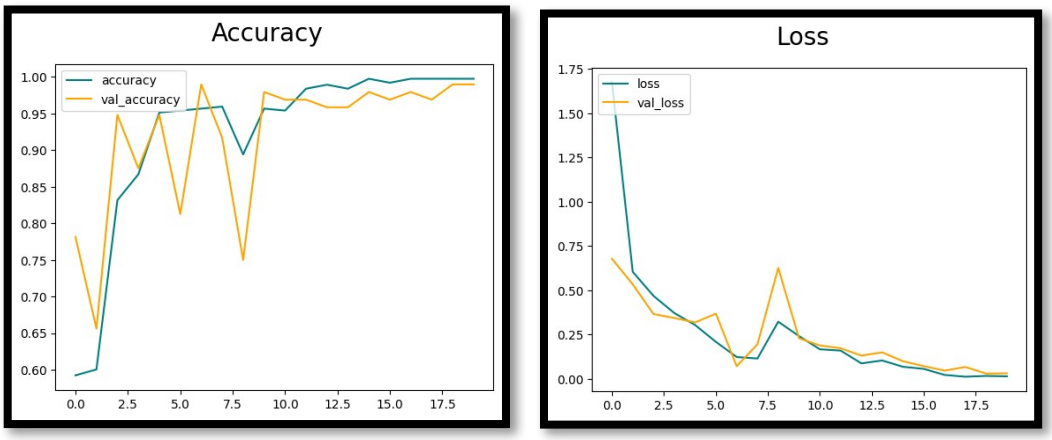


Figure 6.1 Accuracy and loss graph of the model

CHAPTER 7
SYSTEM TESTING

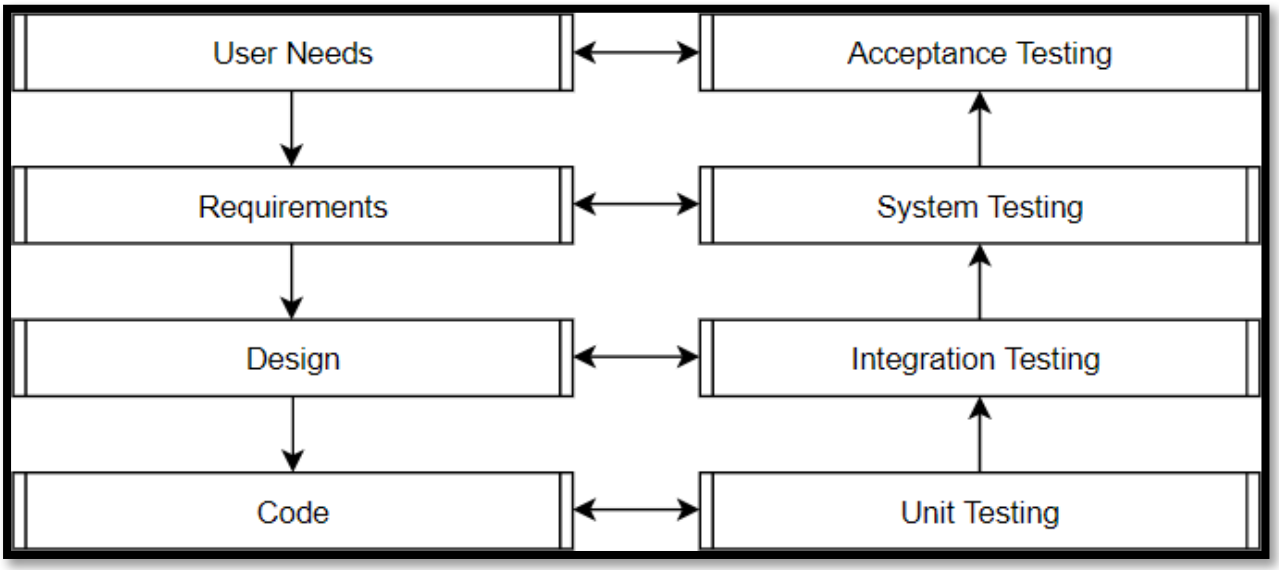
7.1. TESTING OBJECTIVES

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.2. TYPES OF TESTS

In order to uncover the errors, present in different phases we have the concept of levels of testing. The basic levels of testing are

7.2.1 UNIT TEST CASES



Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program
inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application.

It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

In this project the unit testing validates the program logic of all modules in range of machine learning.

The testing takes place as:

Input **:** Click the image.

Output **:** Show which module is selected and show instructions to work with that module.

7.2.2 FUNCTIONAL TEST CASE

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : Image, Serial Number,

Functions **:** Inputs must be processed based on the constraints and Deep Learning algorithm must be executed.

Output **:** Probability of data is carried out.

7.2.3. INTEGRATED TEST CASES

Integration tests are designed to test integrated software and hardware components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing. The combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.
In this project the integration takes place in hardware components and software testing is verified.

Input : Click the image and cropping the images.
Output : System working well, instantaneous output.
Front End : Python.
Back End : Nodejs, Android Studio.

CHAPTER 8
CONCLUSION AND FUTURE ENHANCEMENT

8.1. CONCLUSION:

- Image processing and machine learning techniques, particularly using convolutional neural networks (CNN), can be effective in detecting fake currency. By training a CNN model with a dataset of genuine and fake currency images, the model can learn to identify patterns and features that distinguish real from counterfeit bills.
- This can be achieved by extracting relevant features from the currency images and feeding them into the CNN model, which can then learn to classify the bills as genuine or fake. The effectiveness of the model can be further enhanced by incorporating additional features such as texture, watermark, and micro-printing.
- Overall, utilizing image processing and machine learning techniques can provide a reliable and efficient solution for detecting fake currency, which can help prevent economic fraud and ensure the integrity of financial transactions.
- The accuracy which can have achieved 85%.

8.2. FUTURE ENHANCEMENT

- Integration with blockchain technology to track the entire life cycle of banknotes and prevent counterfeit banknotes from entering the system.
- Use of deep learning algorithms, such as GANs (Generative Adversarial Networks), to generate synthetic data for training models and improve their accuracy in detecting sophisticated counterfeit banknotes.
- Development of mobile apps that allow individuals and businesses to easily scan and verify the authenticity of banknotes.
- Incorporation of additional features, such as holograms, unique patterns, and watermarks, into banknotes to increase their complexity and make them harder to counterfeit.
- Integration with existing security systems, such as surveillance cameras and facial recognition, to detect and prevent financial crimes in real-time.

APPENDICES

APPENDIX 1

SAMPLE CODING

Server.js

```
const express=require("express")
const app=express()
const multer = require('multer');
require('dotenv').config()
const bodyParser = require('body-parser');
const uuid=require("uuid")
const fs=require("fs")
const { spawn } = require('child_process');

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));

const upload = multer({ dest: './uploads/' }); // Set the destination folder for uploaded files
app.get("/",function(req,res){
  res.send("hello world")
})
app.route("/find-currency")
  .get((req,res)=>{
    console.log("from get root");
  })
  .post(upload.single('image'),async (req,res)=>{
    try{
```

```
      let file=req.file;
      let default_extension=".jpg"
       fs.rename(file.destination+file.filename,file.destination+file.filename+defau
lt_extension ,async function(err) {
        if (err) {
          res.status(200).json({error:true,result:"Server Error."})
        } else {
          findResult(file.destination+file.filename+default_extension)
          .then(async (result)=>{
            res.status(200).json({success:true,result:result})
            let removed=false
            while(!removed){
                                                                removed=await
removeFile(file.destination+file.filename+default_extension)
            }
          })
          .catch((error)=>{
            res.status(200).json({success:true,result:error})
          })
          // res.status(200).json({success:true,result:"result"})
        }
      });
    }catch(error){
      res.status(200).json({error:true,result:"Server Error."})
    }
   // res.status(200).json({success:true,result:"The given currency is Fake"})
  })

//function to calculate the answer
```

```
async function findResult(path){
  return new Promise((res,rej)=>{
    try{
      const req=JSON.stringify({path,method:"imageProcess"})
      const script = spawn('python', ["image.py",req]);
      // Read the output from the Python script
      script.stdout.on('data', function(data) {
        let result = data.toString().trim();
        res(result)
      });

      // Handle errors
      script.stderr.on('data', function(data) {
        let error=data.toString()
        res(error)
      });

      script.on('exit', function(code) {

      });
    }catch(error){
      res({error:true,result:"Server Error."})
    }
  })
}

//function to remove the particular file
async function removeFile(imagePath){
  try{
```

```
    fs.rmSync(imagePath)
  }catch(error){
    return false
  }
  return true
}

app.listen(process.env.PORT,()=>{
  console.log(`Running in ${process.env.PORT}`);
})
```

Image.js

```
import sys
import json
import cv2
import tensorflow as tf
from tensorflow.keras.models import load_model
import numpy as np
from matplotlib import pyplot as plt
import warnings
import os
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
warnings.filterwarnings("ignore")

def imageProcess(path):
    #loading model to the env
    model = load_model('models/imageclassifier.h5')
    IMAGE_SIZE=(512,720)
    #loading image to the env
```

```
    img = cv2.imread(path)

    img=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
    resize = tf.image.resize(img, IMAGE_SIZE)
    yhat = model.predict(np.expand_dims(resize/255, 0),verbose = 0)
    # plt.imshow(resize.numpy().astype(int))
    # plt.show()
    yhat=yhat[0][0]
    if yhat > 0.5:
                print("Predicted value is Original and the accuracy is {:.2f}
%".format(yhat*100))
    else:
                 print("Predicted value is Fake and the accuracy is {:.2f}
%".format(yhat*100))

functionMapper={
    "imageProcess":imageProcess
}
if(__name__=="__main__"):
    req=json.loads(sys.argv[1])
    functionMapper[req["method"]](req["path"])
```

APPENDIX 2

SCREENSHOTS

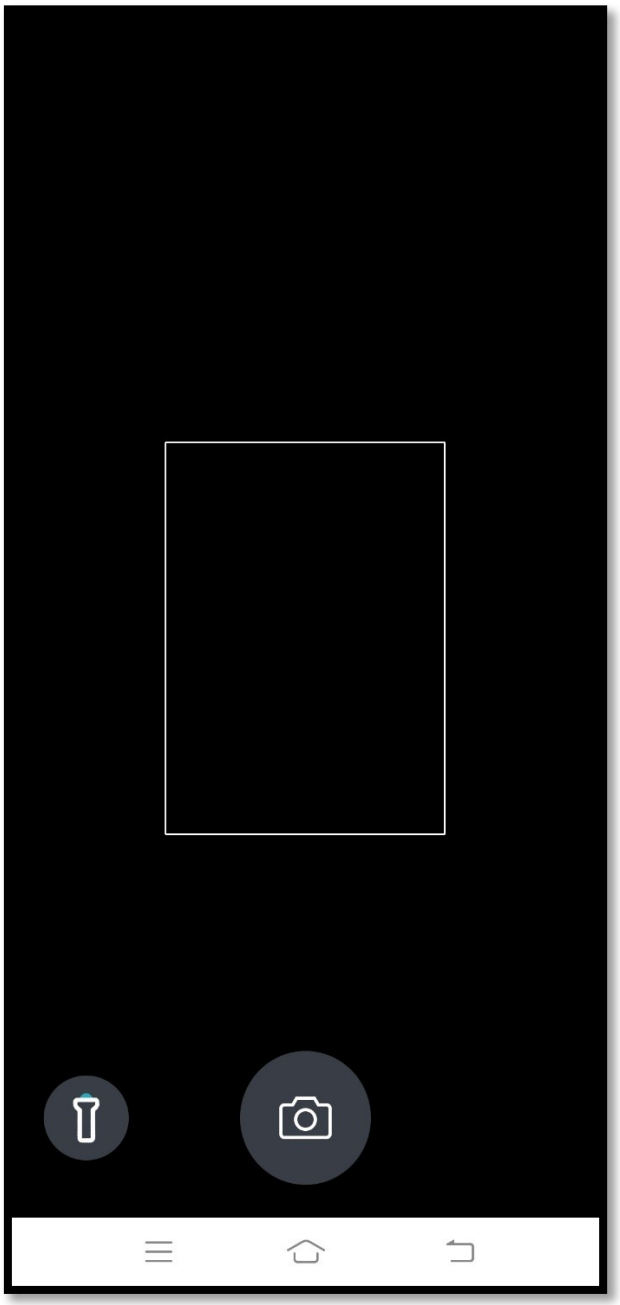


Figure 8.4.1 User Interface

Figure 8.4.2 Click the image and predict result

REFERENCES

1. [1] Aman Bhatia, Vansh Kedia, Anshul Shroff, Mayand Kumar, Bickey Kumar Shah, Aryan, "Fake Currency Detection with Machine Learning Algorithm and Image Processing" DOI:10.1109/ICICCS51141.2021.9432274.

2. [2] Chinmay Bhurke, Meghana Sirdeshmukh, Prof. Mrs. M.S.Kanitkar, ―Currency Recognition Using Image Processing International Journal of Innovative Research in Computer and Communication Engineering, Vol. 3, Issue 5, May 2015.

3. [3] Eshita Pilania, Bhavika Arora, ―Recognition of Fake Currency Based on Security Thread Feature of Currency‖ International Journal Of Engineering And Computer Science, ISSN: 2319-7242.

4. [4] Komal Vora, Ami Shah, Jay Mehta, ―A Review Paper on Currency Recognition System‖ International Journal of Computer Applications (0975 – 8887), Volume 115 – No. 20, April 2022.

5. [5] P. Julia Grace, Ph.D., A. Sheema, ―A survey on Fake Indian Paper Currency Identification System‖ Grace et al., International Journal of Advanced Research in Computer Science and Software Engineering 6(7), July- 2016, pp. 340-345 ISSN: 2277 128X.

6. [6] S. Arya and M. Sasikumar, "Fake CurrencyDetection," 2019 Int. Conf. Recent Adv. Energy Efficient Comput. Commun. ICRAECC 2019, pp.2019–2022, 2019, doi: 10.1109 /ICRAECC43874.2019.8994968.