Image Steganography with CNN Based Encoder-Decoder Model Abstract:

Image steganography is a significant area of research that aims at finding any hidden images or information within other digital images. In this paper, we propose a novel approach combining the Convolutional Neural Network (CNN) model and an encoder and decoder network to ensure accurate and efficient image steganalysis. The proposed method leverages the discriminative power of CNNs to extract features of importance from images and then uses the encoder-decoder networks to reconstruct the original image from the stego or hidden content. Our customized CNN model is designed to capture the features present in the steganographic images. The Encoder-Decoder network plays a crucial role in steganalysis by reconstructing the original image from the hidden content. By training the network on a diverse set of steganographic images, it learns to identify the distinctive artifacts introduced during the steganographic embedding process. The reconstructed image is then compared with the original image using appropriate similarity measures, allowing us to accurately detect the presence of hidden information. The combination of the customized CNN model and the Encoder-Decoder network enables efficient and robust image steganalysis, making it a valuable tool for digital forensics and security applications.

**INTRODUCTION**

Image Steganography is a method of hiding information within digital images to ensure secure communication. Leveraging advancements in deep learning, Convolutional Neural Networks (CNNs) are now used to create encoder-decoder models for embedding and retrieving secret messages. This approach enhances the robustness and efficiency of steganography compared to traditional methods. The CNN-based models not only allow precise encoding but also improve the quality of the decoded images.

**SYSTEM ANALYSIS**

**EXISTING SYSTEM**

Traditional image steganography systems rely on methods like Least Significant Bit (LSB) manipulation, Discrete Cosine Transform (DCT), and other signal processing techniques. These approaches have limitations in terms of payload capacity, robustness against attacks, and visual imperceptibility of the embedded image. Additionally, traditional systems often lack the adaptability required to handle the diverse and dynamic challenges in secure communication.

**DISADVANTAGES**:

1. Limited payload capacity, restricting the amount of data that can be embedded.
2. Susceptibility to attacks such as noise addition, compression, or cropping.
3. Degradation in image quality due to simplistic embedding methods.
4. Lack of scalability and adaptability for complex image datasets.

**PROPOSED SYSTEM**

The proposed system employs a CNN-based encoder-decoder model to perform image steganography. The encoder is responsible for embedding the secret message into the image, while the decoder retrieves it without compromising the quality of the image or the message. The system leverages the learning capabilities of CNNs to optimize the encoding and decoding processes, ensuring imperceptibility, robustness, and accuracy in various scenarios. By using deep learning, the proposed system adapts to varying image features and enhances security.

**ADVANTAGES**

1. High payload capacity for embedding larger messages.
2. Enhanced robustness against various attacks and distortions.
3. Maintains the visual quality of the carrier image.
4. Increased retrieval accuracy with minimal data loss.
5. Adaptive to diverse datasets and dynamic requirements.

**SYSTEM REQUIREMENTS:**

**HARDWARE REQUIREMENTS:**

* System : Pentium Dual Core.
* Hard Disk : 120 GB.
* Monitor : 15’’ LED
* Input Devices : Keyboard, Mouse
* Ram : 1 GB

**SOFTWARE REQUIREMENTS:**

* Operating system : Windows 10/11.
* Coding Language : python
* Tool : pycharm
* Database : MYSQL
* framework : django

**SYSTEM ARCHITECTURE**

**Fig:** Architecture

IMPLEMENTATION

MODULES

1. DATA CONSUMER
2. DATA PRODUCER
3. TRUSTED AUTHORITY
4. UAVCLOUDPROVIDER
5. BLOCKCHAIN

Module description

1. **Preprocessing Module**: Resizes and normalizes input images and messages.
2. **Encoding Module**: Uses CNN to embed the message into the image.
3. **Stegno Transmission Module**: Transfers the Stegno-image securely.
4. **Decode Module**: Extracts the hidden message from the stego-image using CNN
5. **Analysis Module**: Evaluates the quality of the stego-image and retrieval accuracy

**SYSTEM DESIGN**

**UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

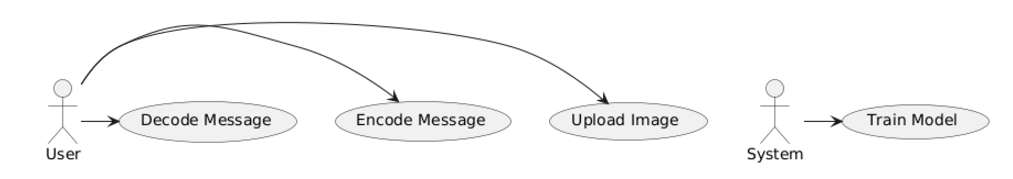
**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

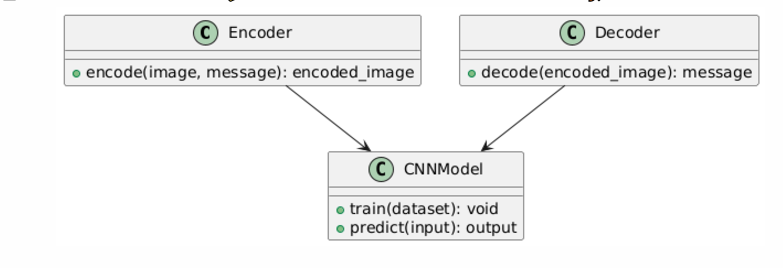
**USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



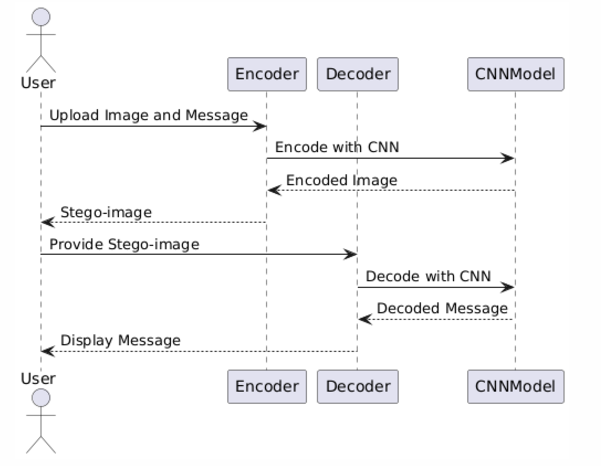
**CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

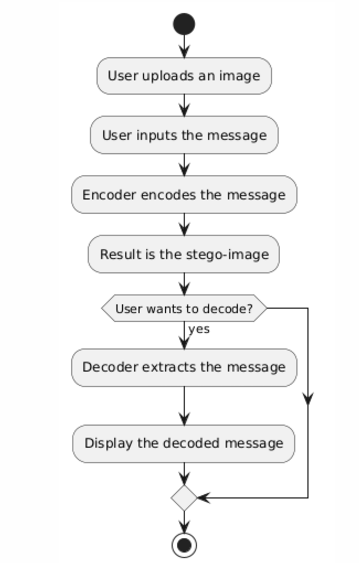
****

**SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

****

**ACTIVITY:** An activity diagram is a type of Unified Modeling Language (UML) flowchart that shows the flow from one activity to another in a system or process. It's used to describe the different dynamic aspects of a system and is referred to as a 'behavior diagram' because it describes what should happen in the modeled system.

****

**LITERATURAL SURVEY**

1. **Title: Image Steganography Using Deep Convolutional Neural Networks  
   Authors**: J. Zhu, R. Kaplan, J. Johnson, L. Fei-Fei

**Year: 2018**

**Abstract:** This paper presents a novel approach to image steganography using a deep convolutional neural network (DCNN). The model encodes a secret image into a cover image, preserving the visual quality of both. Results demonstrate improved robustness and imperceptibility compared to traditional methods.

1. **Title: Deep Learning-Based Steganography and Steganalysis  
   Authors:** H. Baluja

**Year:** 2019

**Abstract:** This work explores the use of deep learning for both steganography and steganalysis. A CNN-based encoder-decoder architecture is proposed for embedding and retrieving hidden messages in images, offering robustness against transformations and achieving high retrieval accuracy.

1. **Title: Secure Image Steganography with GANs and CNNs  
   Authors:** A. Kumar, S. Tripathi

**Year:** 2020

**Abstract:** This study integrates Generative Adversarial Networks (GANs) with CNNs to improve the robustness and security of image steganography. The proposed system enhances imperceptibility and achieves high embedding efficiency in various test conditions.

1. **Title:** A Deep Learning Framework for Image Steganography Using CNN **Authors:** K. Patel, V. Mishra

**Year:** 2021

**Abstract**: The paper proposes a CNN-based steganographic model that embeds textual and image data within digital images. The framework focuses on preserving image quality while maximizing the capacity and retrieval accuracy of the hidden data.

1. **Title:** Robust Image Steganography with Encoder-Decoder Networks **Authors:** M. Zhang, Y. Li, X. Zhou

**Year:** 2022

**Abstract:** This research develops an encoder-decoder network for secure image steganography. By learning spatial and frequency features, the model achieves imperceptible embedding and robust retrieval under various attackscenarios**.**

**SOFTWARE ENVIRONMENT**

**PYTHON**

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An [interpreted language](https://en.wikipedia.org/wiki/Interpreted_language), Python has a design philosophy that emphasizes code [readability](https://en.wikipedia.org/wiki/Readability) (notably using [whitespace](https://en.wikipedia.org/wiki/Whitespace_character) indentation to delimit [code blocks](https://en.wikipedia.org/wiki/Code_block) rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer [lines of code](https://en.wikipedia.org/wiki/Source_lines_of_code) than might be used in languages such as [C++](https://en.wikipedia.org/wiki/C%2B%2B)or [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many [operating systems](https://en.wikipedia.org/wiki/Operating_system). [CPython](https://en.wikipedia.org/wiki/CPython" \o "CPython), the [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) of Python, is [open source](https://en.wikipedia.org/wiki/Open_source) software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit [Python Software Foundation](https://en.wikipedia.org/wiki/Python_Software_Foundation). Python features a [dynamic type](https://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory management](https://en.wikipedia.org/wiki/Memory_management). It supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [functional](https://en.wikipedia.org/wiki/Functional_programming) and [procedural](https://en.wikipedia.org/wiki/Procedural_programming), and has a large and comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library)

**DJANGO**

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes [reusability](https://en.wikipedia.org/wiki/Reusability" \o "Reusability)and "pluggability" of components, rapid development, and the principle of [don't repeat yourself](https://en.wikipedia.org/wiki/Don%27t_repeat_yourself). Python is used throughout, even for settings files and data models.



Django also provides an optional administrative [create, read, update and delete](https://en.wikipedia.org/wiki/Create,_read,_update_and_delete) interface that is generated dynamically through [introspection](https://en.wikipedia.org/wiki/Introspection_(computer_science)) and configured via admin models



**INPUT AND OUTPUT DESIGN**

**INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**OBJECTIVES**

1.Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the
* Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.

**SYSTEM STUDY**

**FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

**Three key considerations involved in the feasibility analysis are,**

* **ECONOMICAL FEASIBILITY**
* **TECHNICAL FEASIBILITY**
* **SOCIAL FEASIBILITY**

**ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**SYSTEM TEST**

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, 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.

### TYPES OF TESTS

**Unit testing**

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.

**Integration testing**

Integration tests are designed to test integrated software 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.

**Functional test**

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 : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**Unit Testing**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

# Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

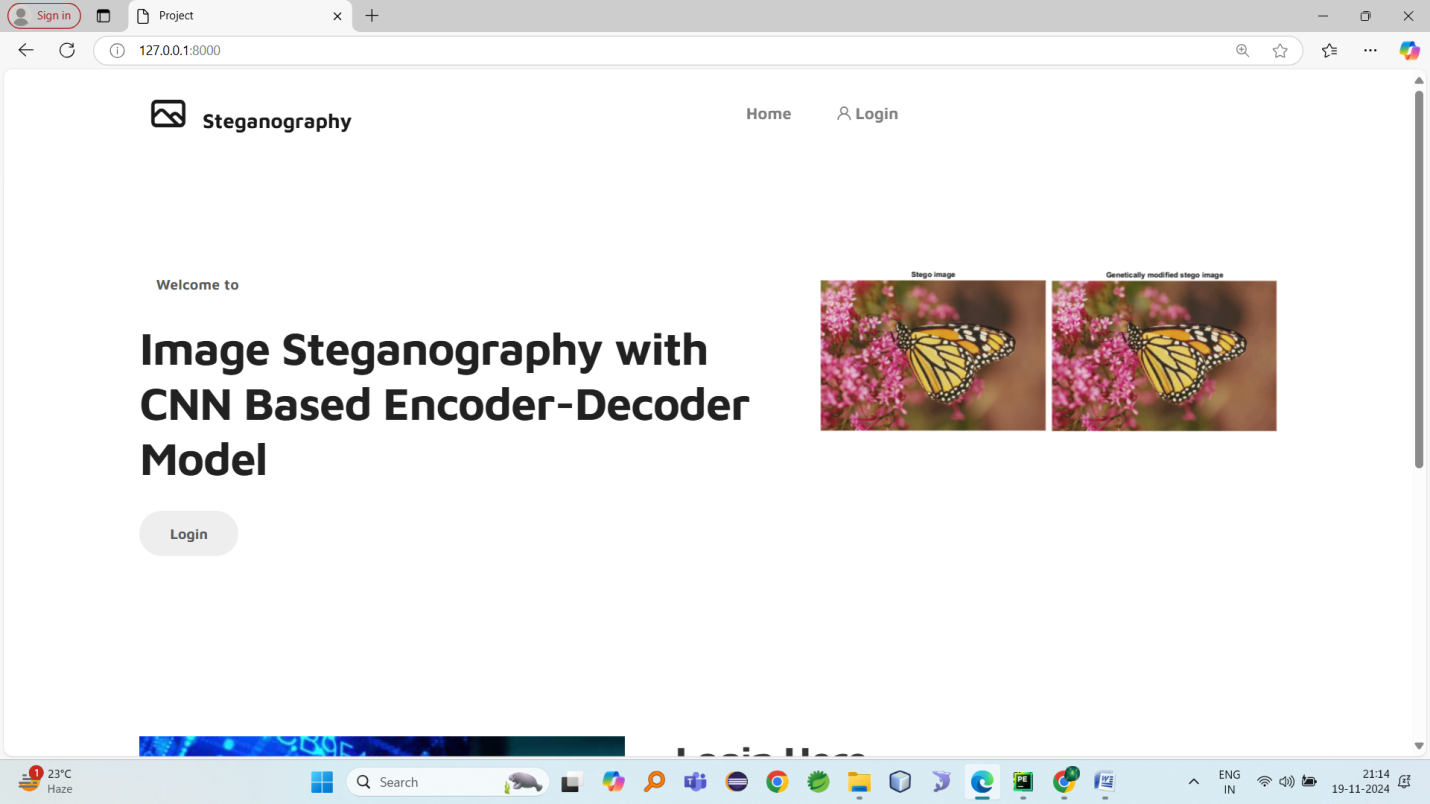
**Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

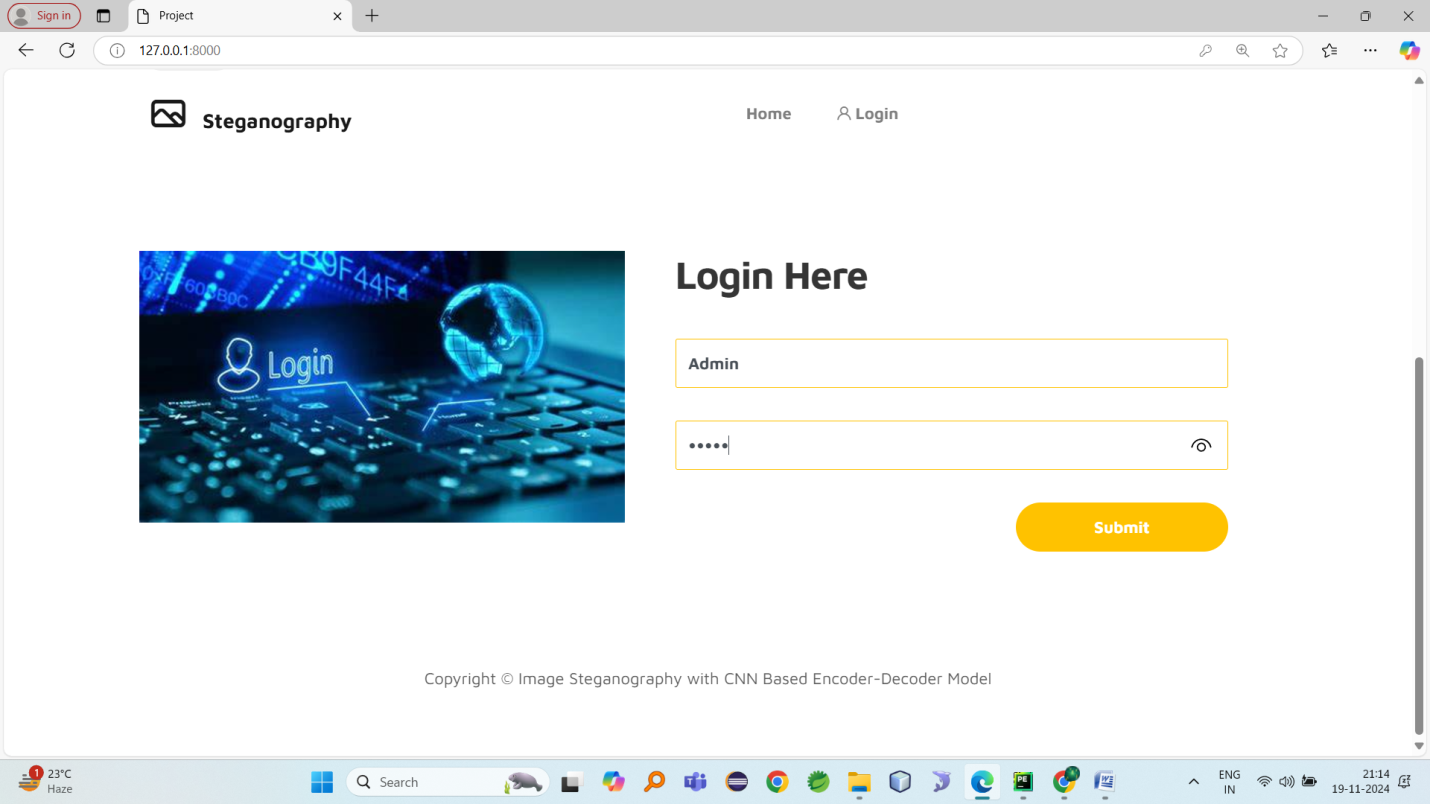
**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

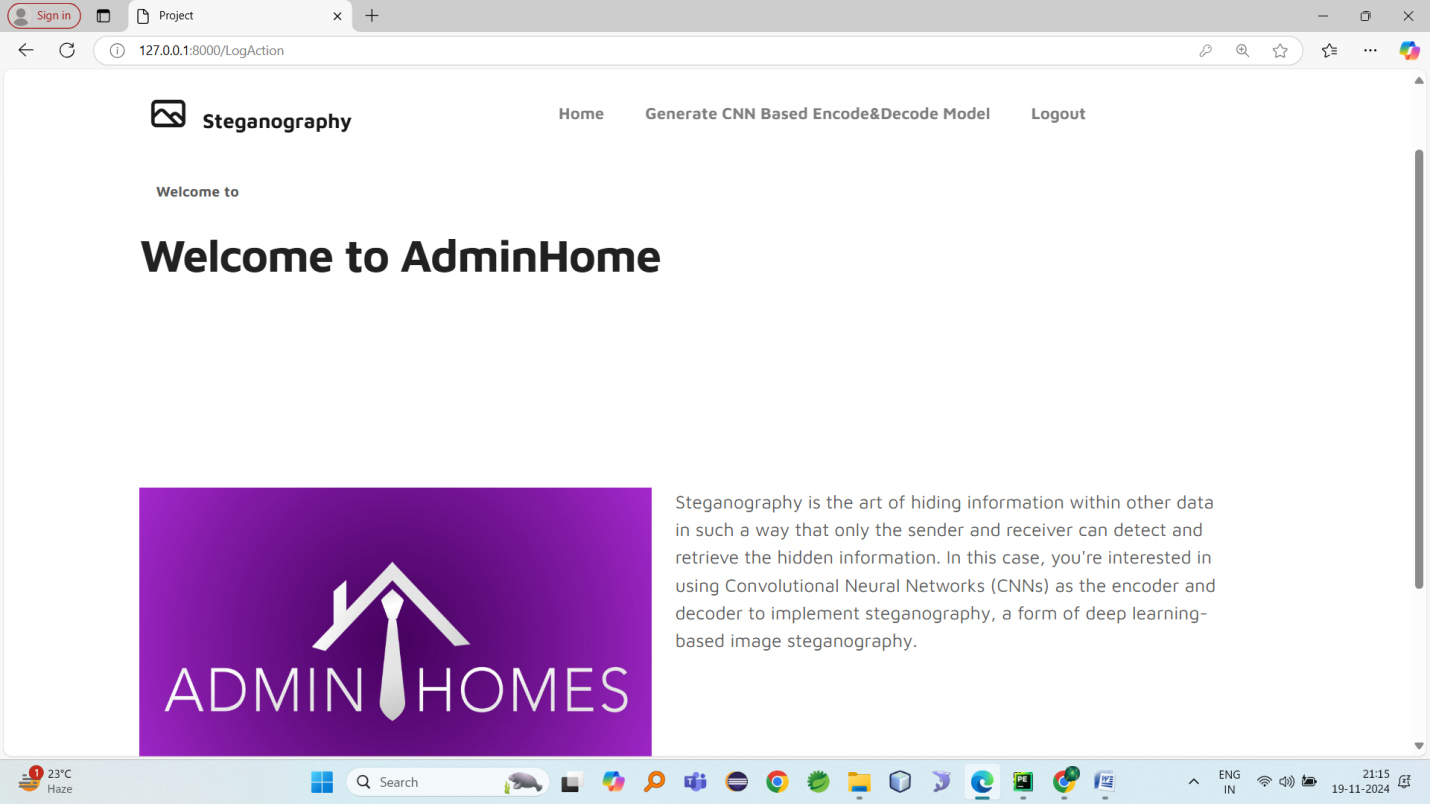
**SCREEN SHOTS**

Index page

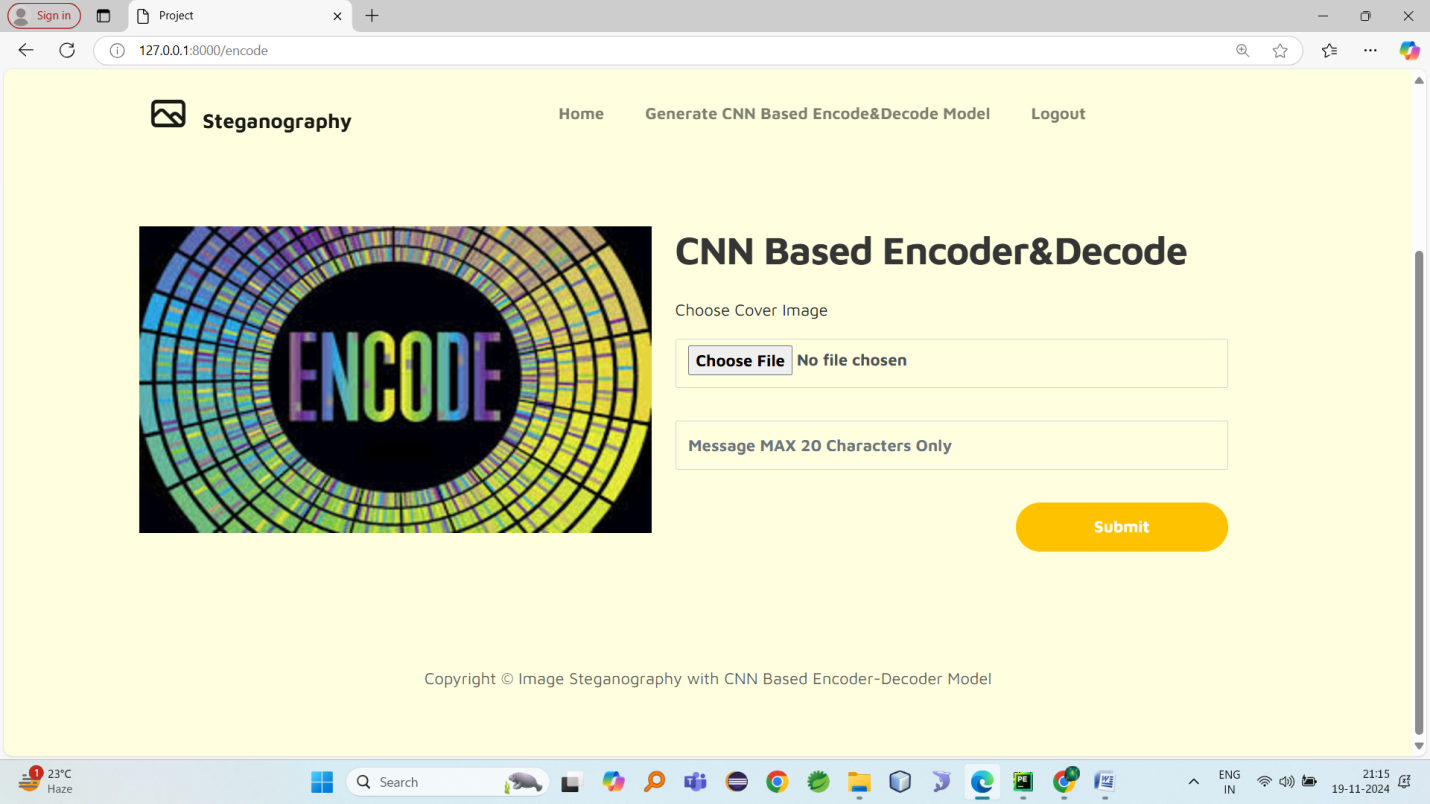


Admin login

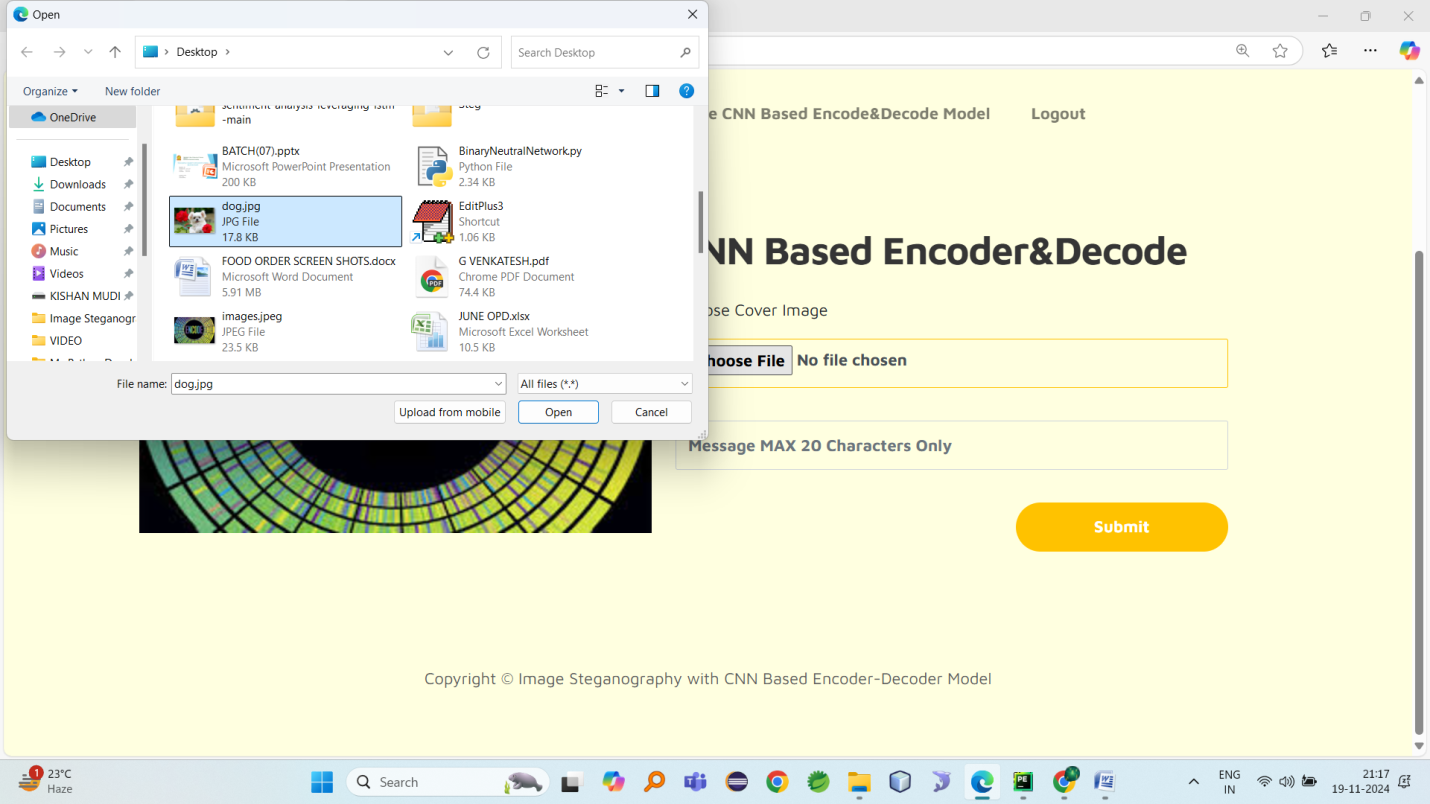
  
Admin home page



Cnn Based Encode And Decode

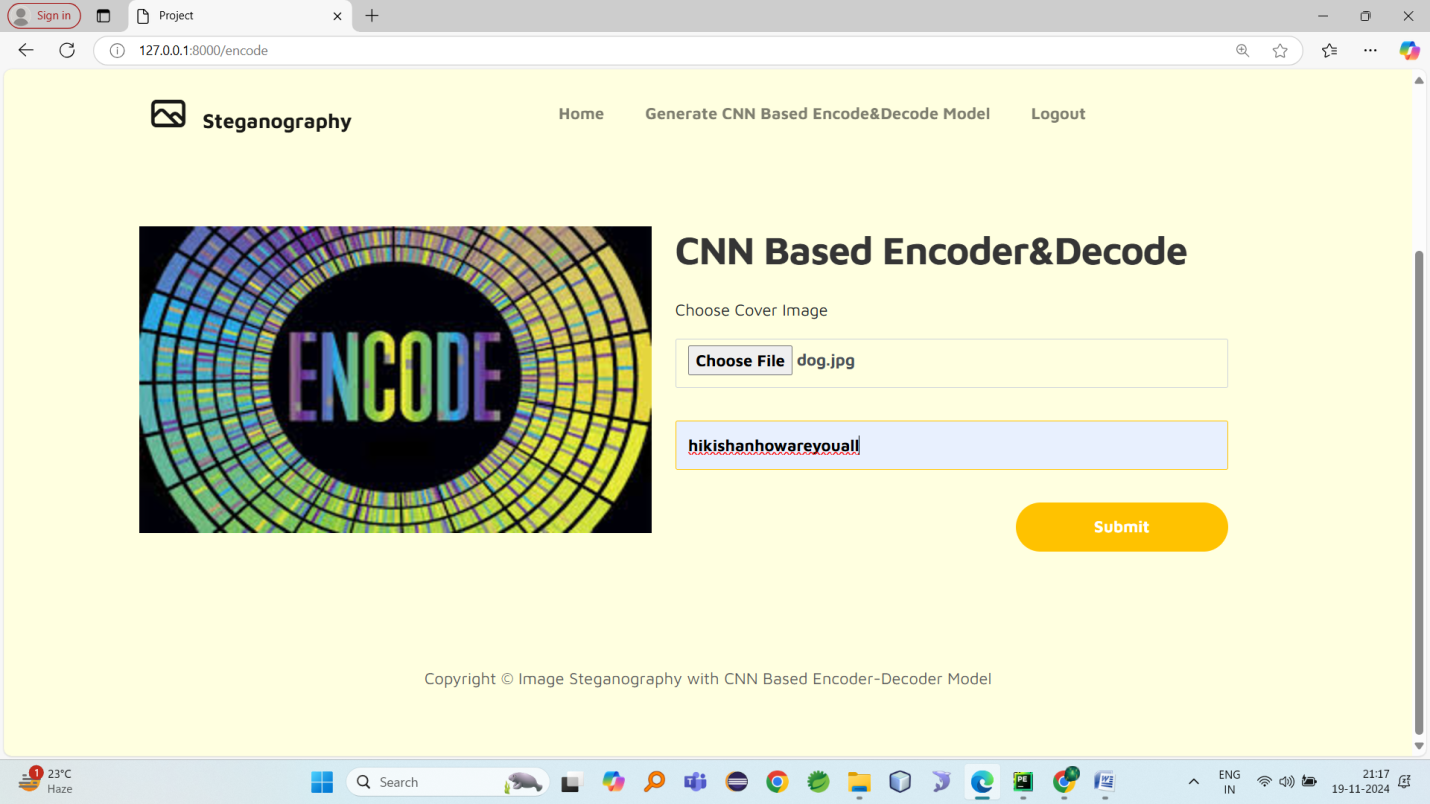


Uploading dog image as cover image

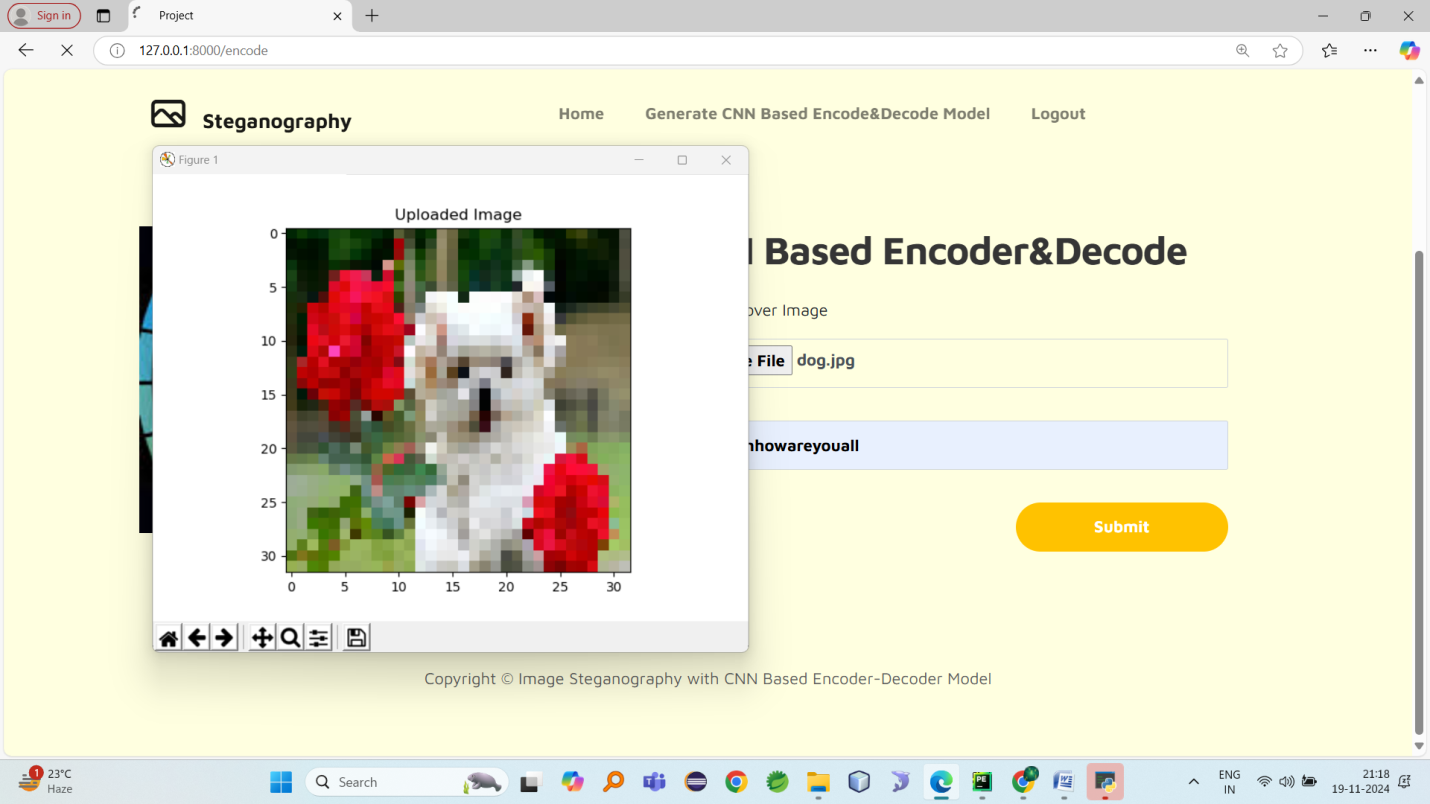


And we are trying to add data in to dog image

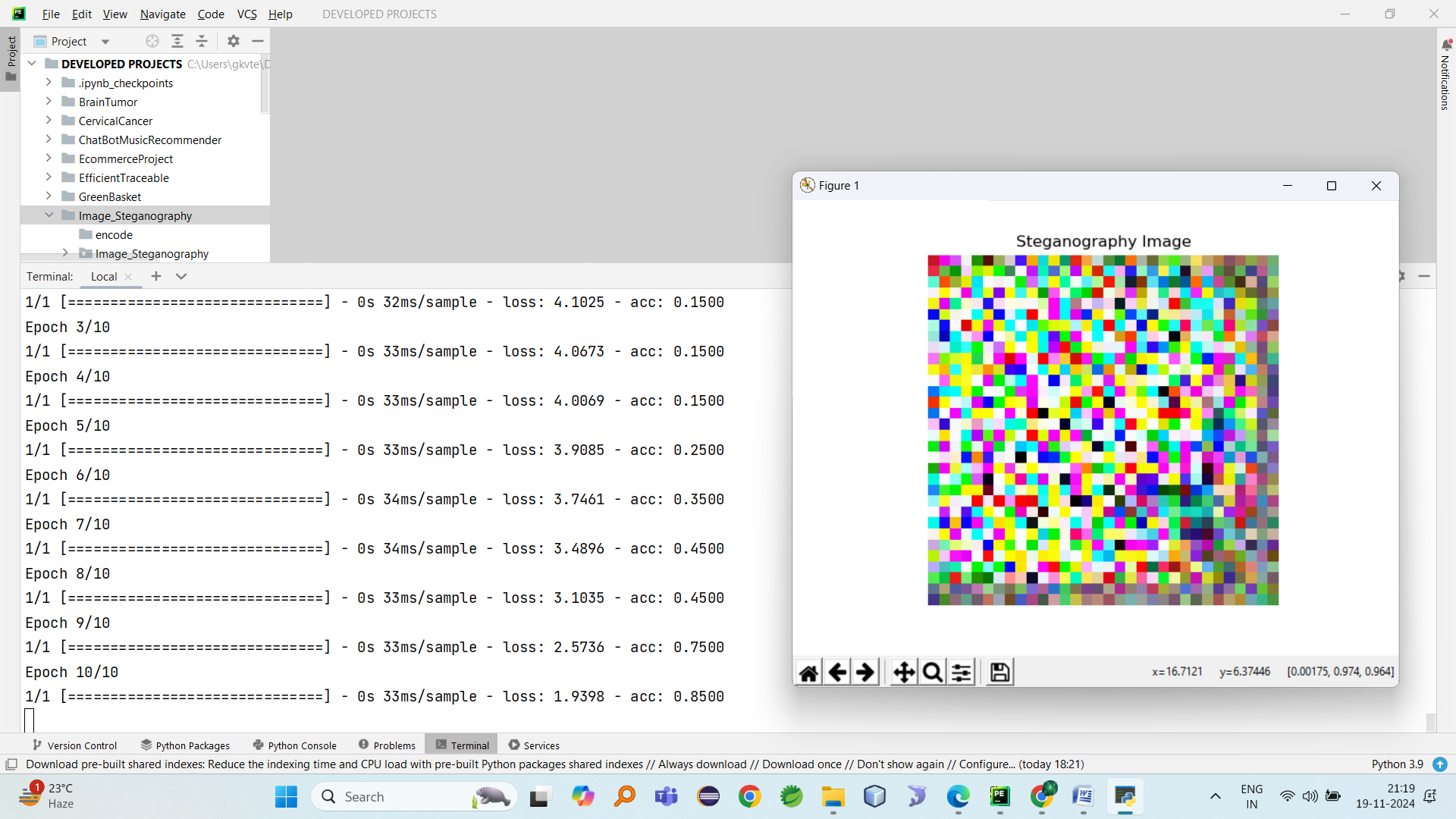
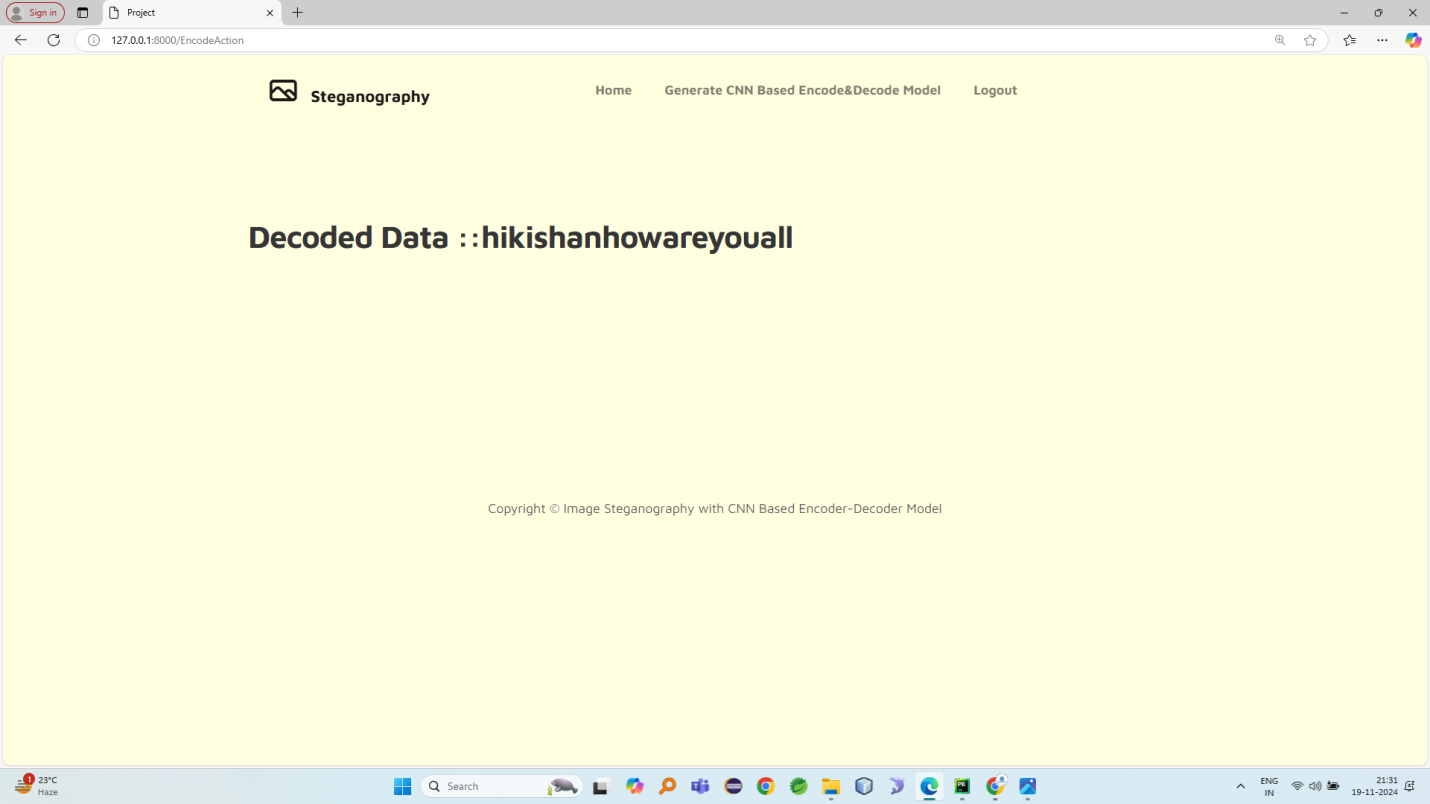
Note: must give 20 character without spaces



See dog image after pre-process into 32X32 pixels



Cnn based encoded model generated and steganography image generated

decoded text (90-100% accurate decodes) 

SAMPLE CODE

**views.py**

from django.shortcuts import render

from django.core.files.storage import FileSystemStorage

import numpy as np

import tensorflow as tf

from tensorflow.keras.preprocessing import image

import matplotlib.pyplot as plt

import os

import cv2

from tensorflow.keras import layers, Model

# Create your views here.

def index(request):

return render(request,'index.html')

def LogAction(request):

uname=request.POST['username']

pwd=request.POST['password']

if uname == 'Admin' and pwd == 'Admin':

return render(request,'AdminHome.html')

else:

context={'msg':'Login Failed...!!'}

return render(request,'AdminApp/index.html',context)

def home(request):

return render(request,'AdminHome.html')

def encode(request):

return render(request,'Encode.html')

# Define the character set and mappings for one-hot encoding

char\_set = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789'

char\_to\_index = {char: idx for idx, char in enumerate(char\_set)}

index\_to\_char = {idx: char for idx, char in enumerate(char\_set)}

# Function to convert text to one-hot encoding

def text\_to\_one\_hot(text, char\_to\_index, text\_length):

one\_hot\_encoded = np.zeros((text\_length, len(char\_to\_index))) # Shape: (text\_length, vocab\_size)

for i, char in enumerate(text):

if i < text\_length:

index = char\_to\_index.get(char, -1) # Get index of the character

if index != -1:

one\_hot\_encoded[i, index] = 1 # Set the corresponding index to 1

return one\_hot\_encoded

# Encoder Network: Embeds the secret text into the image

def build\_encoder(input\_image\_shape, text\_length, vocab\_size):

image\_input = layers.Input(shape=input\_image\_shape)

text\_input = layers.Input(shape=(text\_length, vocab\_size))

# CNN layers to process the image

x = layers.Conv2D(32, (3, 3), activation='relu')(image\_input)

x = layers.MaxPooling2D()(x)

x = layers.Conv2D(64, (3, 3), activation='relu')(x)

x = layers.MaxPooling2D()(x)

x = layers.Conv2D(128, (3, 3), activation='relu')(x)

x = layers.MaxPooling2D()(x)

x = layers.Flatten()(x)

# Concatenate the text input with the image features

x = layers.Concatenate()([x, layers.Flatten()(text\_input)])

# Fully connected layer to produce the final encoded image

encoded\_image = layers.Dense(np.prod(input\_image\_shape), activation='sigmoid')(x)

encoded\_image = layers.Reshape(input\_image\_shape)(encoded\_image)

encoder = Model(inputs=[image\_input, text\_input], outputs=encoded\_image)

return encoder

# Decoder Network: Extracts the secret text from the image

def build\_decoder(input\_image\_shape, text\_length, vocab\_size):

image\_input = layers.Input(shape=input\_image\_shape)

# CNN layers to process the encoded image and extract text features

x = layers.Conv2D(128, (3, 3), activation='relu')(image\_input)

x = layers.MaxPooling2D()(x)

x = layers.Conv2D(64, (3, 3), activation='relu')(x)

x = layers.MaxPooling2D()(x)

x = layers.Conv2D(32, (3, 3), activation='relu')(x)

x = layers.MaxPooling2D()(x)

x = layers.Flatten()(x)

decoded\_text = layers.Dense(text\_length \* vocab\_size, activation='softmax')(x)

decoded\_text = layers.Reshape((text\_length, vocab\_size))(decoded\_text)

decoder = Model(inputs=image\_input, outputs=decoded\_text)

return decoder

global filename, uploaded\_file\_url

def EncodeAction(request):

global filename, uploaded\_file\_url

decoded\_text='null'

if request.method == 'POST' and request.FILES['file']:

myfile = request.FILES['file']

message = request.POST['message']

if len(message) == 20:

fs = FileSystemStorage()

location = myfile.name

filename = fs.save(myfile.name, myfile)

uploaded\_file\_url = fs.url(filename)

BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

imagedisplay = cv2.imread(BASE\_DIR + "/" + uploaded\_file\_url)

# Load an image from your system (ensure the path is correct)

#image\_path = myfile # Replace with the actual path to your image

img = image.load\_img(BASE\_DIR + "/" + uploaded\_file\_url, target\_size=(32, 32)) # Resize to 128x128

img\_array = image.img\_to\_array(img) # Convert image to numpy array

# Normalize the image to [0, 1]

img\_array = img\_array / 255.0

# Display the image

plt.imshow(img\_array)

plt.title('Uploaded Image')

plt.show()

plt.close()

print("Image Shape:", img\_array.shape)

# Example secret text and text length

secret\_text = message

text\_length = 20 # Maximum length of the secret text

encoded\_text = text\_to\_one\_hot(secret\_text, char\_to\_index, text\_length)

print("Encoded Text Shape:", encoded\_text.shape)

# Define input image shape

input\_image\_shape = (32, 32, 3)

# Build encoder and decoder

encoder = build\_encoder(input\_image\_shape, text\_length, len(char\_to\_index))

decoder = build\_decoder(input\_image\_shape, text\_length, len(char\_to\_index))

# Define the full model (encoder + decoder)

encoded\_image = encoder.output

decoded\_text = decoder(encoded\_image)

full\_model = Model(inputs=[encoder.input[0], encoder.input[1]], outputs=decoded\_text)

full\_model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

full\_model.summary()

# Prepare the image and text for training

train\_images = np.expand\_dims(img\_array, axis=0) # Add batch dimension

train\_texts = np.expand\_dims(encoded\_text, axis=0) # Add batch dimension

# Train the full model

full\_model.fit([train\_images, train\_texts], train\_texts, epochs=25, batch\_size=1)

# Save the model for future use

full\_model.save('encoder\_decoder\_model.h5')

# Test the model with a new image and secret text

test\_image = np.expand\_dims(img\_array, axis=0) # Use the same test image

test\_text = np.expand\_dims(encoded\_text, axis=0) # Use the same secret text

# Encode the secret text into the image

encoded\_test\_image = encoder.predict([test\_image, test\_text])

# Remove the batch dimension by squeezing it (assuming only one image in the batch)

img\_single = encoded\_test\_image[0] # Or use img\_array.squeeze() if you want to squeeze all dimensions

# Display the image

plt.imshow(img\_single)# img\_single will have shape (32, 32, 3)

plt.title('Steganography Image')

plt.axis('off') # Hide axis for clarity

plt.savefig('Stegano\_image.png', bbox\_inches='tight', pad\_inches=0)

plt.show()

plt.close()

# Decode the text from the encoded image

decoded\_text\_from\_image = decoder.predict(encoded\_test\_image)

# Convert the decoded output (one-hot encoding) back to text

decoded\_text\_from\_image = np.argmax(decoded\_text\_from\_image, axis=-1) # Get the indices

decoded\_text = ''.join([index\_to\_char[idx] for idx in decoded\_text\_from\_image[0]])

print("Decoded Text:", decoded\_text)

context = {'data': decoded\_text}

return render(request,'Decode.html', context)

else:

context = {'data': "Message Length Must Be 20 Characters"}

return render(request,'Encode.html', context)

**CONCLUSION**

The CNN-based encoder-decoder model for image steganography offers a robust and efficient solution for secure data hiding. It outperforms traditional methods in terms of imperceptibility, payload capacity, and resistance to distortions. This system demonstrates the potential of deep learning in enhancing steganographic techniques.

**REFERENCES**

1. Zhu, J., Kaplan, R., Johnson, J., &Fei-Fei, L. (2018). "Image Steganography Using Deep Convolutional Neural Networks."
2. Baluja, H. (2019). "Deep Learning-Based Steganography and Steganalysis."
3. Kumar, A., &Tripathi, S. (2020). "Secure Image Steganography with GANs and CNNs."
4. Patel, K., & Mishra, V. (2021). "A Deep Learning Framework for Image Steganography Using CNN."
5. Zhang, M., Li, Y., & Zhou, X. (2022). "Robust Image Steganography with Encoder-Decoder Networks."