



**KURUNJI VENKATRAMANA GOWDA
POLYTECHNIC SULLIA-574327
5TH SEMESTER AI/ML WEEK-1**

Week 1:

History:

Artificial Intelligence History

The term artificial intelligence was coined in 1956, but AI has become more popular today thanks to increased data volumes, advanced algorithms, and improvements in computing power and storage.

Early AI research in the 1950s explored topics like problem solving and symbolic methods. In the 1960s, the US Department of Defence took interest in this type of work and began training computers to mimic basic human reasoning. For example, the defence Advanced Research Projects Agency (DARPA) completed street mapping projects in the 1970s. And DARPA produced intelligent personal assistants in 2003, long before Siri, Alexa or Cortana were household names.

This early work paved the way for the automation and formal reasoning that we see in computers today, including decision support systems and smart search systems that can be designed to complement and augment human abilities.

While Hollywood movies and science fiction novels depict AI as human-like robots that take over the world, the current evolution of AI technologies isn't that scary – or quite that smart. Instead, AI has evolved to provide many specific benefits in every industry. Keep reading for modern examples of artificial intelligence in health care, retail and more.

1950s–1970s Neural Networks

Early work with neural networks stirs excitement for “thinking machines.”

1980s–2010s Machine Learning

Machine learning becomes popular.

Present Day Deep Learning

Deep learning breakthroughs drive AI boom.

Fundamentals of AI:

Artificial Intelligence/AI:

What is artificial intelligence (AI)?

Artificial intelligence is the simulation of human intelligence processes by machines, especially computer systems. Specific applications of AI include expert systems, natural language processing, speech recognition and machine vision.

How AI works?

AI works by combining large amounts of data with fast, iterative processing and intelligent algorithms, allowing the software to learn automatically from patterns or features in the data. AI is a broad field of study that includes many theories, methods and technologies, as well as the following major subfields:

- **Machine learning:** Automates analytical model building. It uses methods from neural networks, statistics, operations research and physics to find hidden insights in data without explicitly being programmed for where to look or what to conclude.
- **A neural network:** Is a type of machine learning that is made up of interconnected units (like neurons) that processes information by responding to external inputs, relaying information between each unit. The process requires multiple passes at the data to find connections and derive meaning from undefined data.
- **Deep learning:** Uses huge neural networks with many layers of processing units, taking advantage of advances in computing power and improved training techniques to learn complex patterns in large amounts of data. Common applications include image and speech recognition.
- **Computer vision:** Relies on pattern recognition and deep learning to recognize what's in a picture or video. When machines can process, analyse and understand images, they can capture images or videos in real time and interpret their surroundings.
- **Natural language processing (NLP):** It is the ability of computers to analyse, understand and generate human language, including speech. The next stage of NLP is natural language interaction, which allows humans to communicate with computers using normal, everyday language to perform tasks.

Additionally, several technologies enable and support AI:

- **Graphical processing units:** They are key to AI because they provide the heavy compute power that's required for iterative processing. Training neural networks requires big data plus compute power.

- **The Internet of Things (IOT):** Generates massive amounts of data from connected devices, most of it unanalysed. Automating models with AI will allow us to use more of it.
- **Advanced algorithms:** They are being developed and combined in new ways to analyse more data faster and at multiple levels. This intelligent processing is key to identifying and predicting rare events, understanding complex systems and optimizing unique scenarios.
- **APIs, or application programming interfaces:** They are portable packages of code that make it possible to add AI functionality to existing products and software packages. They can add image recognition capabilities to home security systems and Q&A capabilities that describe data, create captions and headlines, or call out interesting patterns and insights in data.

In summary, the goal of AI is to provide software that can reason on input and explain on output. AI will provide human-like interactions with software and offer decision support for specific tasks, but it's not a replacement for humans – and won't be anytime soon.

Purpose of AI:

There are some main purposes and features in which artificial intelligence is used in different fields or zones. These are the main purposes of artificial intelligence.

1. **Improves decision making**
2. **Singularity**
3. **Machine learning**
4. **Business process optimization**
5. **Creative work in technologies**
6. **Provides financial services**
7. **Health care**
8. **Automotive/Robotics**
9. **HR & Recruitment**

1 – Improves decision making:

The basic goal of artificial intelligence is to provide mechanism for decision making. This decision making is based on rare data as input data and will provide artificial intelligent result like human mind.

Artificial intelligence has the ability to make better decisions by automating the different physical and other tasks. These tasks can reduce the human labour and also saves the time. 2

– Singularity:

The ultimate objective of artificial intelligence is to overtake the work of human being. In near future, the growth of technology will become uncontrollable that will result into massive changes in human life style. Besides their side effects, these intelligent technologies will make the work simpler and efficient.

3 – Machine learning

The main difference between machine learning and artificial intelligence is that machine learning is mainly concerned with accuracy. Machine learning is the sub-field of artificial intelligence and it takes data to produce the output as it is more focused.

4 – Business process optimization

Business has the vital integrity in the economy of any country. The business process optimization is carried out by streamlining the work and removing the redundancies that ultimately results in improvement of the business.

The robotic process optimization is also used to minimize the daily routine work performed by the humans through different algorithms.

5 – Creative work in technologies

There are number of technologies that are used to simplify the workflow and are easy to integrate across the business. These technologies are very important and are playing an important role in different fields of life like Virtual Reality, Live streaming apps, Drones etc.

6 – Provide financial services

Artificial intelligence has played a huge role in financial services. It is used in fraud detection, risk management, asset management and insurance besides countless other subfields of financial services. There is almost no sub-field left without the use of artificial intelligence applications.

7 – Health care

It the most important sector where the artificial intelligence has revolutionized the sector. A large number of healthcare institutions are using the artificial intelligence machines for better and fast diagnoses the diseases in the patients.

8 – Automotive

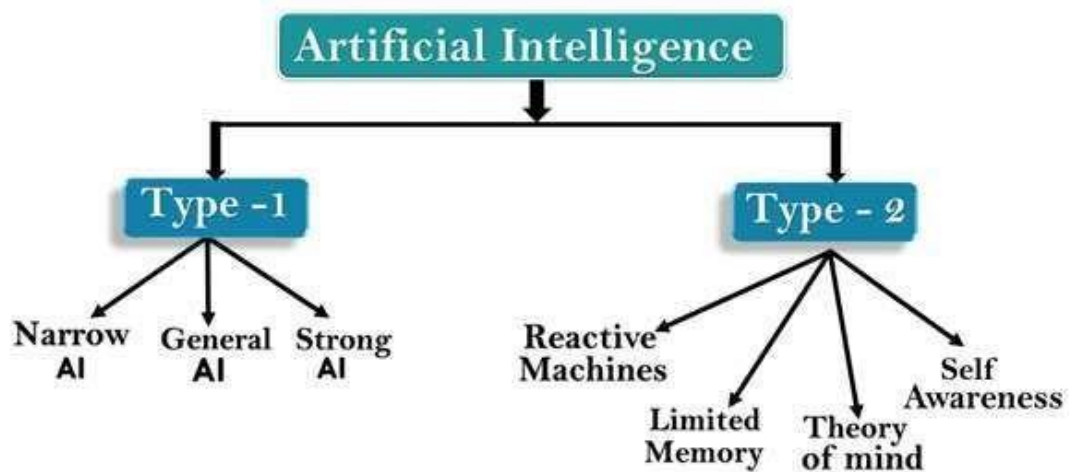
Artificial intelligence has a huge impact on automotive industry. You will find it everywhere from car manufacturing industry to driver monitoring and driver recognition. There is artificial intelligence software available for driver monitoring. The software can make seat adjustment, mirror adjustment and even temperature adjustment.

9 – HR & Recruitment

Artificial intelligence in HR & recruitment is to boost-up the speed and precision of decision making and make the selection more reliable and accurate.

KNUGP

Types of Artificial Intelligence:



First of all, we are going to discuss about the AI type-1 which is based on Capabilities:

AI type-1: Based on Capabilities

1. Weak AI or Narrow AI:

- † Narrow AI is a type of AI which is able to perform a dedicated task with intelligence. The most common and currently available AI is Narrow AI in the world of Artificial Intelligence.
- † Narrow AI cannot perform beyond its field or limitations, as it is only trained for one specific task. Hence it is also termed as weak AI. Narrow AI can fail in unpredictable ways if it goes beyond its limits.
- † Apple Siri is a good example of Narrow AI, but it operates with a limited pre-defined range of functions.
- † IBM's Watson supercomputer also comes under Narrow AI, as it uses an Expert system approach combined with Machine learning and natural language processing.
- † Some Examples of Narrow AI are playing chess, purchasing suggestions on ecommerce site, self-driving cars, speech recognition, and image recognition.

2. General AI:

- † General AI is a type of intelligence which could perform any intellectual task with efficiency like a human.

- ✦ The idea behind the general AI to make such a system which could be smarter and think like a human by its own.
- ✦ Currently, there is no such system exist which could come under general AI and can perform any task as perfect as a human.
- ✦ The worldwide researchers are now focused on developing machines with General AI.
- ✦ As systems with general AI are still under research, and it will take lots of efforts and time to develop such systems.

3. Super AI:

- ✦ Super AI is a level of Intelligence of Systems at which machines could surpass human intelligence, and can perform any task better than human with cognitive properties. It is an outcome of general AI.
- ✦ Some key characteristics of strong AI include capability include the ability to think, to reason, solve the puzzle, make judgments, plan, learn, and communicate by its own.
- ✦ Super AI is still a hypothetical concept of Artificial Intelligence. Development of such systems in real is still world changing task.

Now, the second type which is type-2 based on functionality:

Artificial Intelligence type-2: Based on functionality

1. Reactive Machines

- ✦ Purely reactive machines are the most basic types of Artificial Intelligence.
- ✦ Such AI systems do not store memories or past experiences for future actions.
- ✦ These machines only focus on current scenarios and react on it as per possible best action.
- ✦ IBM's Deep Blue system is an example of reactive machines. ✦ Google's AlphaGo is also an example of reactive machines.

2. Limited Memory

- ✦ Limited memory machines can store past experiences or some data for a short period of time.
- ✦ These machines can use stored data for a limited time period only.
- ✦ Self-driving cars are one of the best examples of Limited Memory systems. These cars can store recent speed of nearby cars, the distance of other cars, speed limit, and other information to navigate the road.

3. Theory of Mind

- ✦ Theory of Mind AI should understand the human emotions, people, beliefs, and be able to interact socially like humans.
- ✦ This type of AI machines are still not developed, but researchers are making lots of efforts and improvement for developing such AI machines.

4. Self-Awareness

- † Self-awareness AI is the future of Artificial Intelligence. These machines will be super intelligent, and will have their own consciousness, sentiments, and selfawareness.
- † These machines will be smarter than human mind.
- † Self-Awareness AI does not exist in reality still and it is a hypothetical concept.

Goals of Artificial Intelligence



- † **Logic, problem-solving**
- † **Knowledge representation**
- † **Planning**
- † **Learning**
- † **Social Intelligence**
- † **Creativity**
- † **General Intelligence**

1. Logic problem-solving:

Logic, problem-solving: Early researchers developed algorithms that simulate humans' step-by-step reasoning when solving puzzles or making logical deductions. By the late 1980s and 1990s, AI research had developed methods for dealing with uncertain or incomplete information, employing concepts from probability and economics.

2. Knowledge representation:

Knowledge representation and knowledge engineering are central to AI research. Many of the problems that machines are expected to solve will require extensive world knowledge.

3. Planning:

Intelligent agents must be able to set goals and achieve them. They need a way to envision the future - a representation of the state of the world and make predictions about

how their actions will change it - and be able to make choices that maximize the utility (or "value") of the options available.

4. Learning:

Machine learning, a fundamental concept of AI research since the field's inception, is the study of computer algorithms that automatically improve through experience.

Unsupervised learning is the ability to find patterns in a stream of input. Supervised learning includes both classification and numerical regression. After seeing several examples of things from several categories, classification is used to determine which category something falls into. Regression attempts to construct a function that describes the relationship between inputs and outputs and predicts how the outputs should change as the inputs change.

5. Social Intelligence:

Effective computing is the study and development of systems that can detect, interpret, process, and simulate human It is an interdisciplinary field spanning computer science, psychology, and cognitive science. While the origins of the field can be traced to early philosophical inquiries into emotion, the more modern branch of computer science originated from Rosalind Picard's 1995 paper on "effective computing".

6. Creativity:

A sub-field of AI addresses creativity theoretically (philosophical, psychological perspective) and practically (the specific implementation of systems that produce novel and useful outputs). Some related areas of computational research include artificial intuition and artificial thinking.

7. General Intelligence:

Many researchers think that their work will eventually result in a machine with artificial general intelligence, combining all the skills described above and exceeding human capacity in most or all of these areas. Some believe that such a project may require anthropomorphic features such as artificial consciousness or an artificial brain

Application of AI:

Artificial Intelligence has various applications in today's society. It is becoming essential for today's time because it can solve complex problems with an efficient way in multiple industries, such as Healthcare, entertainment, finance, education, etc. AI is making our daily life more comfortable and faster.

Following are some sectors which have the application of Artificial Intelligence:



1. AI in Astronomy

- ✦ Artificial Intelligence can be very useful to solve complex universe problems. AI technology can be helpful for understanding the universe such as how it works, origin, etc.

2. AI in Healthcare

- ✚ In the last, five to ten years, AI becoming more advantageous for the healthcare industry and going to have a significant impact on this industry.
- ✚ Healthcare Industries are applying AI to make a better and faster diagnosis than humans. AI can help doctors with diagnoses and can inform when patients are worsening so that medical help can reach to the patient before hospitalization.

3. AI in Gaming:

- ✚ AI can be used for gaming purpose. The AI machines can play strategic games like chess, where the machine needs to think of a large number of possible places.

4. AI in Finance:

- ✚ AI and finance industries are the best matches for each other. The finance industry is implementing automation, chatbot, adaptive intelligence, algorithm trading, and machine learning into financial processes.

5. AI in Data Security:

- ✚ The security of data is crucial for every company and cyber-attacks are growing very rapidly in the digital world. AI can be used to make your data more safe and secure. Some examples such as AEG bot, AI2 Platform, are used to determine software bug and cyber-attacks in a better way.

6. AI in Social Media:

- ✚ Social Media sites such as Facebook, Twitter, and Snapchat contain billions of user profiles, which need to be stored and managed in a very efficient way. AI can organize and manage massive amounts of data. AI can analyze lots of data to identify the latest trends, hashtag, and requirement of different users.

7. AI in Travel & Transport:

- ✚ AI is becoming highly demanding for travel industries. AI is capable of doing various travel related works such as from making travel arrangement to suggesting the hotels, flights, and best routes to the customers. Travel industries are using AI-powered chatbots which can make human-like interaction with customers for better and fast response.

8. AI in Automotive Industry:

- ✚ Some Automotive industries are using AI to provide virtual assistant to their user for better performance. Such as Tesla has introduced TeslaBot, an intelligent virtual assistant.
- ✚ Various Industries are currently working for developing self-driven cars which can make your journey more safe and secure.

9. AI in Robotics:

- ✚ Artificial Intelligence has a remarkable role in Robotics. Usually, general robots are programmed such that they can perform some repetitive task, but with the help of AI,

we can create intelligent robots which can perform tasks with their own experiences without pre-programmed.

- ✦ Humanoid Robots are best examples for AI in robotics, recently the intelligent Humanoid robot named as Erica and Sophia has been developed which can talk and behave like humans.

10. AI in Entertainment

- ✦ We are currently using some AI based applications in our daily life with some entertainment services such as Netflix or Amazon. With the help of ML/AI algorithms, these services show the recommendations for programs or shows.

11. AI in Agriculture

- ✦ Agriculture is an area which requires various resources, labor, money, and time for best result. Now a day's agriculture is becoming digital, and AI is emerging in this field. Agriculture is applying AI as agriculture robotics, solid and crop monitoring, predictive analysis. AI in agriculture can be very helpful for farmers.

12. AI in E-commerce

- ✦ AI is providing a competitive edge to the e-commerce industry, and it is becoming more demanding in the e-commerce business. AI is helping shoppers to discover associated products with recommended size, color, or even brand.

13. AI in education

- ✦ AI can automate grading so that the tutor can have more time to teach. AI chatbot can communicate with students as a teaching assistant.
- ✦ AI in the future can be work as a personal virtual tutor for students, which will be accessible easily at any time and any place.

Ethics in AI:

AI ethics is a set of guidelines that advise on the design and outcomes of artificial intelligence. Human beings come with all sorts of cognitive biases, such as recency and confirmation bias, and those inherent biases are exhibited in our behaviours and subsequently, our data.

1. **Respect for Persons:** This principle recognizes the autonomy of individuals and upholds an expectation for researchers to protect individuals with diminished autonomy, which could be due to a variety of circumstances such as illness, a mental disability, age restrictions.
2. **Beneficence:** This principle takes a page out of healthcare ethics, where doctors take an oath to “do no harm.”

3. **Justice:** This principle deals with issues, such as fairness and equality. Who should reap the benefits of experimentation and machine learning? The Belmont Report offers five ways to distribute burdens and benefits, which are by:

✚ Equal share	Societal contribution
✚ Individual need	Merit
✚ Individual effort	

Examples of AI in real world:

1. Self-Driving and Parking Vehicles:

Self-driving and parking cars use deep learning, a subset of AI, to recognize the space around a vehicle. Technology company Nvidia uses AI to give cars “the power to see, think, and learn, so they can navigate a nearly infinite range of possible driving scenarios,” Nvidia explains on its website. The company’s AI-powered technology is already in use in cars made by Toyota, Mercedes-Benz, Audi, Volvo, and Tesla, and is sure to revolutionize how people drive—and enable vehicles to drive themselves.

2. Digital Assistants:

Apple’s Siri, Google Now, Amazon’s Alexa, and Microsoft’s Cortana are digital assistants that help users perform various tasks, from checking their schedules and searching for something on the web, to sending commands to another app. AI is an important part of how these apps work because they learn from every single user interaction. This allows them to better recognize speech patterns and serve users results that are tailored to their preferences. Microsoft says that Cortana is “continually learns about its user” and that it will eventually anticipate user needs.

3. Vehicle Recognition Identification

Did you know that many of the traffic cameras around your city use AI to read license plates? Companies such as PlateSmart, IntelliVision, and Sighthound, among others, use computer vision—a form of AI that can see and understand images—along with deep learning to turn conventional surveillance into vehicle monitoring; this is a very important part of integrated traffic systems and also a big help to authorities as well, as surveillance videos are now searchable for specific plate numbers. That’ll make you think twice about blowing through that red light.

4. Robots

The Roomba 980 model vacuum (the one that cleans your floor on its own) uses AI to scan a living area’s size, look for objects that might be in the way, and remember the best route for cleaning the carpet. The vacuum bot can also identify how much cleaning it needs to

do based on the size of the room, repeating a cleaning cycle three times in smaller rooms or cleaning twice in a medium-sized room.

Artificial Intelligence and Human Machine Interface (AI & HMI)

Artificial Intelligence (AI) driven technologies, leveraged IoT, Advanced Embedded Systems, Cloud Computing, Big Data, Cognitive Systems, Virtual and Augmented Reality etc are ready to generate new paradigms and developments paths at any level of “physical” systems as automated industrial production systems.

Beside enabling effective management of complex systems and interconnected systems-of-systems, AI driven technologies can strongly impact, and **potentially revolutionize, mechatronics, robotics, automation and human-machine interaction.**

The partnership intends to support the adoption of Artificial Intelligence enhanced cyberphysical system, and more specifically AI driven HMI.

The partnership intends to take into account and address three main targets:

- 1. Competitiveness and revenues**
- 2. Social ecosystem**
- 3. People**

1. Competitiveness and revenues:

Introducing a new enhanced HMI AI driven will allow the **best use of sensors and monitoring systems based on IoT, big-data and cloud technologies.** Moreover, it will yield ever-increasing knowledge processes for **reviewing and improving design and manufacturing.** Shifting automatic machinery and production plants from current mechanics and electronics to AI enhanced cyber-physical systems will **increase performance while reducing material and energy consumptions.**

2. Social ecosystem:

Appropriate adoption of AI technologies will avoid or mitigate the negative impact of a growing extended automation on employment in production and management fields, by generating **new jobs**, strongly based on digital familiarity (typical of new generations), from workshop level to engineering and management ones.

3. People:

A significant **impact on workers** is envisaged in terms of:

- Easing the possible conflict between Artificial Intelligence and Human intelligence, by anticipating relevant risk, and challenges.
- Putting the operator at the center by considering the Person and its cultural environment.
- Improving the efficiency of HMI by a larger use of the virtual cyber and physical environment.
- Reducing workers accident and generally improve job quality.
- Substituting person in boring and alienating jobs.

AI Software Development Life Cycle:

- **Understanding “why” AI is needed:**

The fundamental step is to define and understand why AI is needed. For this, the inputs should ideally be sourced from the organization’s employees who directly deal with customers. The ultimate aim is to offer convenience to customers, so the frontline employees can better understand “why” [AI needs to be incorporated in business processes](#) and how it can transform and improve existing customer-centric operations.

- **Identifying “what” needs AI transformation:**

The next step is to identify the business processes where AI will be implemented. The target is to achieve maximum organizational efficiency and reap greater value through intelligent functionalities, but not every operation needs to undergo automation. Carefully identifying specific areas for AI transformation will ensure streamlined progress towards the desired target, prevent gaps and unnecessary overheads of finances and resources. target, prevent gaps and unnecessary overheads of finances and resources.

- **Selecting data sets for AI solution**

Implementing AI essentially means training applications to process data, gain experience and operate based on the learnings. So it is essential to choose which data sets are suitable to train the AI application. This is an extremely vital step since the final output quality will entirely depend on the quality of data used. And no matter how well the design is, low-quality data will never yield proper results. Data must be gathered from relevant

channels, and it should be high-quality, well-structured, and credible real data. Also, the higher the volume of data better will be the performance of the AI software.

- **Choosing “which” AI capabilities are required:**

Most AI software solutions are a combination of two or more AI capabilities. Studying and selecting the right capabilities based on the targeted objectives make all the difference here. AI capabilities to choose from include-

- [Machine Learning \(ML\)](#), which comprises deep learning, unsupervised and supervised algorithms
- Natural Language Processing (NLP), which includes extraction of content, classification, answering questions, machine translation, and generating text
- [Computer Vision](#), which includes machine vision and image recognition
- Speech, which constitutes capabilities for speech to text conversion and vice versa
- Planning, robotics, and expert systems.

- **Deciding on the right SDLC for the project**

Since AI software development projects are undertaken to achieve high-value targets, deciding the requirements and finalizing them upfront can save effort, time, and funds. Going by this ideology, the waterfall [SDLC method](#) is considered most suitable for AI projects. The waterfall model consists of the following phases-

1. **Requirement analysis phase**
2. **Design phase**
3. **Development phase**
4. **Testing or QA phase**
5. **Deployment phase**
6. **Maintenance phase**

AI software development VS Traditional Software development:

There are many aspects should be compared while comparing the traditional software development with AI software development like type of problem solving, infrastructure etc.

While developing the AI software we generally consider or implement the agile software development methodology which involves below phases:

Each iteration of agile SDLC consists of cross-functional teams working on various phases:

1. Requirement gathering and analysis
2. Design the requirements
3. Construction/ iteration
4. Testing
5. Deployment/installation and maintenance
6. Feedback

We also consider the below these aspects while developing the AI based softwares:

- ✚ Understanding “why” AI is needed: ✚ Identifying “what” needs AI transformation: ✚ Selecting data sets for AI solution
- ✚ Choosing “which” AI capabilities are required:
- ✚ Deciding on the right SDLC for the project

Traditional Software development/ waterfall model software development:

But, in traditional software development or waterfall model software development life cycle we usually develop simple software. Traditional Software Development: Traditional software development is **the software development process used to design and develop simple software**. It is used when the security and many other factors of the software are not much important. It is used by freshers to develop the software.

Artificial Intelligence for playing chess:

AlphaZero is an AI developed by google, The new system, which DeepMind is calling MuZero, is based in part on DeepMind's work with the [AlphaZero](#) AI, which taught itself to master rule-based games like chess and Go. But MuZero also adds a new twist that makes it substantially more flexible.

That twist is called "model-based reinforcement learning." In a system that uses this approach, the software uses what it can see of a game to build an internal model of the game state. Critically, that state isn't prestructured based on any understanding of the game—the AI is able to have a lot of flexibility regarding what information is or is not included in it. The reinforcement learning part of things refers to the training process, which allows the AI to

learn how to recognize when the model it's using is both accurate and contains the information it needs to make decisions.

Predictions:

The model it creates is used to make a number of predictions. These include the best possible move given the current state and the state of the game as a result of the move. Critically, the prediction it makes is based on its internal model of game states—not the actual visual representation of the game, such as the location of chess pieces. The prediction itself is made based on past experience, which is also subject to training.

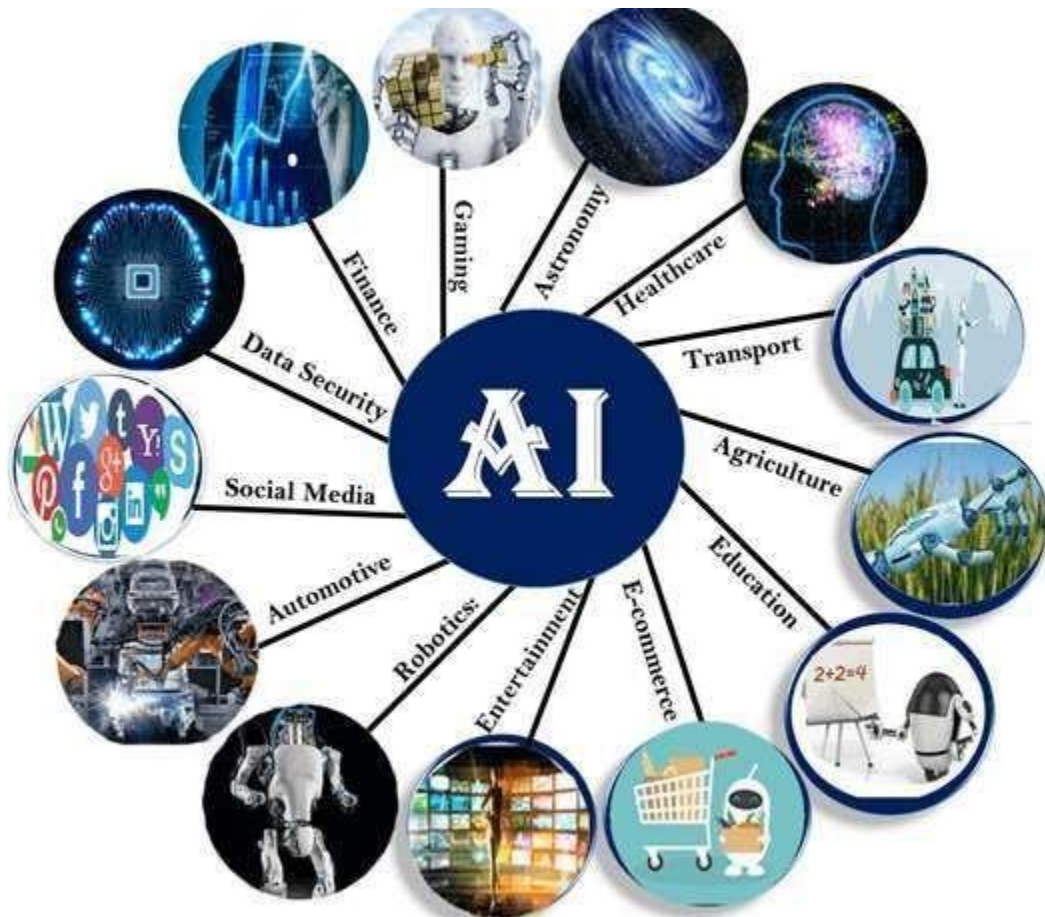
Finally, the value of the move is evaluated using the algorithms predictions of any immediate rewards gained from that move (the point value of a piece taken in chess, for example) and the final state of the game, such as the win or lose outcome of chess. These can involve the same searches down trees of potential game states done by earlier chess algorithms, but in this case, the trees consist of the AI's own internal game models.

If that's confusing, you can also think of it this way: MuZero runs three evaluations in parallel. One (the policy process) chooses the next move given the current model of the game state. A second predicts the new state that results, and any immediate rewards from the difference. And a third considers past experience to inform the policy decision. Each of these is the product of training, which focuses on minimizing the errors between these predictions and what actually happens in-game.

Applications of AI:

Artificial Intelligence has various applications in today's society. It is becoming essential for today's time because it can solve complex problems with an efficient way in multiple industries, such as Healthcare, entertainment, finance, education, etc. AI is making our daily life more comfortable and faster.

Following are some sectors which have the application of Artificial Intelligence:



1. AI in Astronomy:

- ✚ Artificial Intelligence can be very useful to solve complex universe problems. AI technology can be helpful for understanding the universe such as how it works, origin, etc.

2. AI in Healthcare:

- ✚ In the last, five to ten years, AI becoming more advantageous for the healthcare industry and going to have a significant impact on this industry.
- ✚ Healthcare Industries are applying AI to make a better and faster diagnosis than humans. AI can help doctors with diagnoses and can inform when patients are worsening so that medical help can reach to the patient before hospitalization.

3. AI in Gaming:

- ✚ AI can be used for gaming purpose. The AI machines can play strategic games like chess, where the machine needs to think of a large number of possible places.

4. AI in Finance:

- ✚ AI and finance industries are the best matches for each other. The finance industry is implementing automation, chatbot, adaptive intelligence, algorithm trading, and machine learning into financial processes.

5. AI in Data Security:

- ✦ The security of data is crucial for every company and cyber-attacks are growing very rapidly in the digital world. AI can be used to make your data more safe and secure. Some examples such as AEG bot, AI2 Platform, are used to determine software bug and cyber-attacks in a better way.

6. AI in Social Media:

- ✦ Social Media sites such as Facebook, Twitter, and Snapchat contain billions of user profiles, which need to be stored and managed in a very efficient way. AI can organize and manage massive amounts of data. AI can analyze lots of data to identify the latest trends, hashtag, and requirement of different users.

7. AI in Travel & Transport:

- ✦ AI is becoming highly demanding for travel industries. AI is capable of doing various travel related works such as from making travel arrangement to suggesting the hotels, flights, and best routes to the customers. Travel industries are using AI-powered chatbots which can make human-like interaction with customers for better and fast response.

8. AI in Automotive Industry:

- ✦ Some Automotive industries are using AI to provide virtual assistant to their user for better performance. Such as Tesla has introduced TeslaBot, an intelligent virtual assistant.
- ✦ Various Industries are currently working for developing self-driven cars which can make your journey more safe and secure.

9. AI in Robotics:

- ✦ Artificial Intelligence has a remarkable role in Robotics. Usually, general robots are programmed such that they can perform some repetitive task, but with the help of AI, we can create intelligent robots which can perform tasks with their own experiences without preprogrammed.
- ✦ Humanoid Robots are best examples for AI in robotics, recently the intelligent Humanoid robot named as Erica and Sophia has been developed which can talk and behave like humans.

10. AI in Entertainment:

- ✦ We are currently using some AI based applications in our daily life with some entertainment services such as Netflix or Amazon. With the help of ML/AI algorithms, these services show the recommendations for programs or shows.

11. AI in Agriculture:

- ✦ Agriculture is an area which requires various resources, labor, money, and time for best result. Now a day's agriculture is becoming digital, and AI is emerging in this field. Agriculture is applying AI as agriculture robotics, soil and crop monitoring, predictive analysis. AI in agriculture can be very helpful for farmers.

12. AI in E-commerce:

- ✦ AI is providing a competitive edge to the e-commerce industry, and it is becoming more demanding in the e-commerce business. AI is helping shoppers to discover associated products with recommended size, color, or even brand.

13. AI in education:

- ✦ AI can automate grading so that the tutor can have more time to teach. AI chatbot can communicate with students as a teaching assistant.
- ✦ AI in the future can be work as a personal virtual tutor for students, which will be accessible easily at any time and any place etc..

KNUGP

GIT AND GIT HUB:

Git:

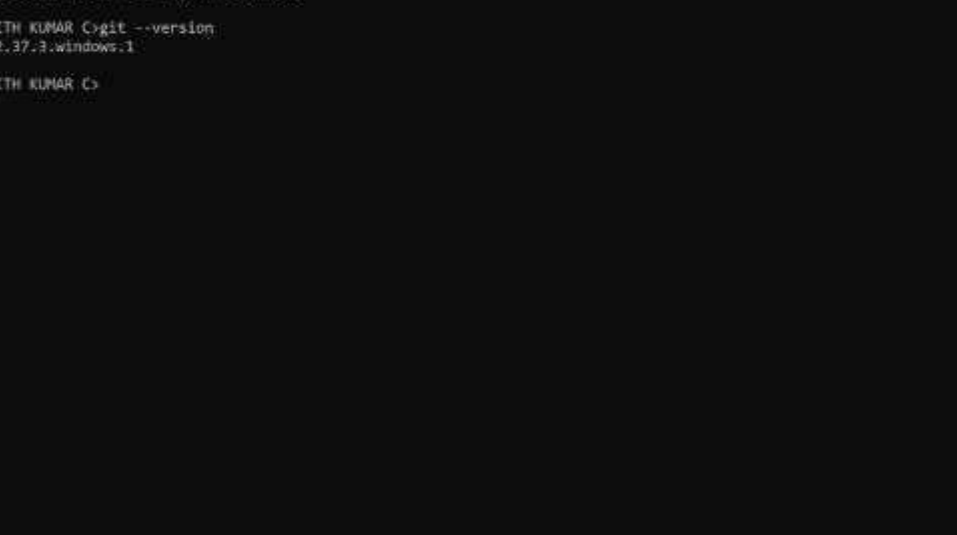
Git is a DevOps tool used for source code management. It is a free and open-source version control system used to handle small to very large projects efficiently. Git is used to tracking changes in the source code, enabling multiple developers to work together on nonlinear development.

Git Downloading:

You can Install git from the below link to your Operation System <https://git-scm.com/downloads>



- Once you download the git executable file from the official link, Open that file.
- After opening the file, in the Git Installation wizard click on “next” after you read the terms and Conditions
- Leave the default settings and click on “next” until the installation is completed.



The screenshot shows a Windows Command Prompt window with the following text:

```

Microsoft Windows [Version 10.0.22000.856]
(c) Microsoft Corporation. All rights reserved.

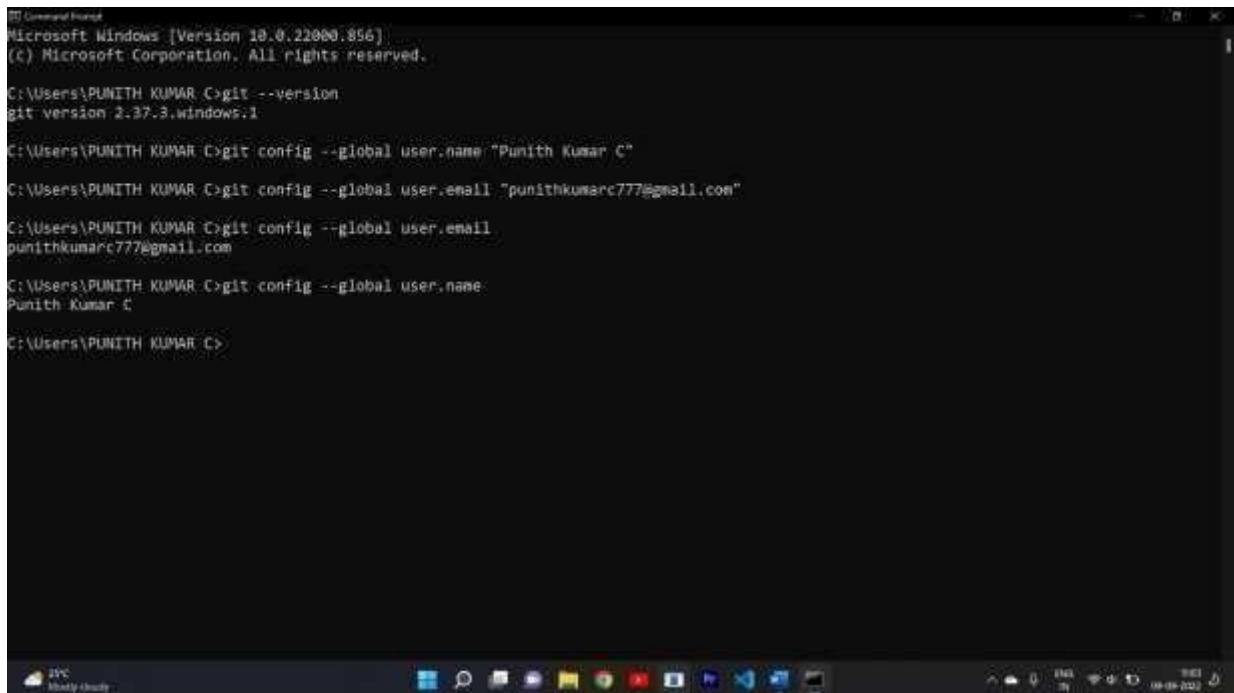
C:\Users\PUNITH KUMAR C>git --version
git version 2.37.3.windows.1

C:\Users\PUNITH KUMAR C>
  
```

The taskbar at the bottom of the screen shows the Start button, a search icon, and several pinned applications including File Explorer, Microsoft Edge, and the Visual Studio Code icon. The system tray on the right indicates the date and time as 08/18/2022, 11:18 AM.

24 | Page

Syntax to edit the email and username: `git config --global --edit`



```
Microsoft Windows [Version 10.0.22000.856]
(c) Microsoft Corporation. All rights reserved.

C:\Users\PUNITH KUMAR C>git --version
git version 2.37.3.windows.1

C:\Users\PUNITH KUMAR C>git config --global user.name "Punith Kumar C"

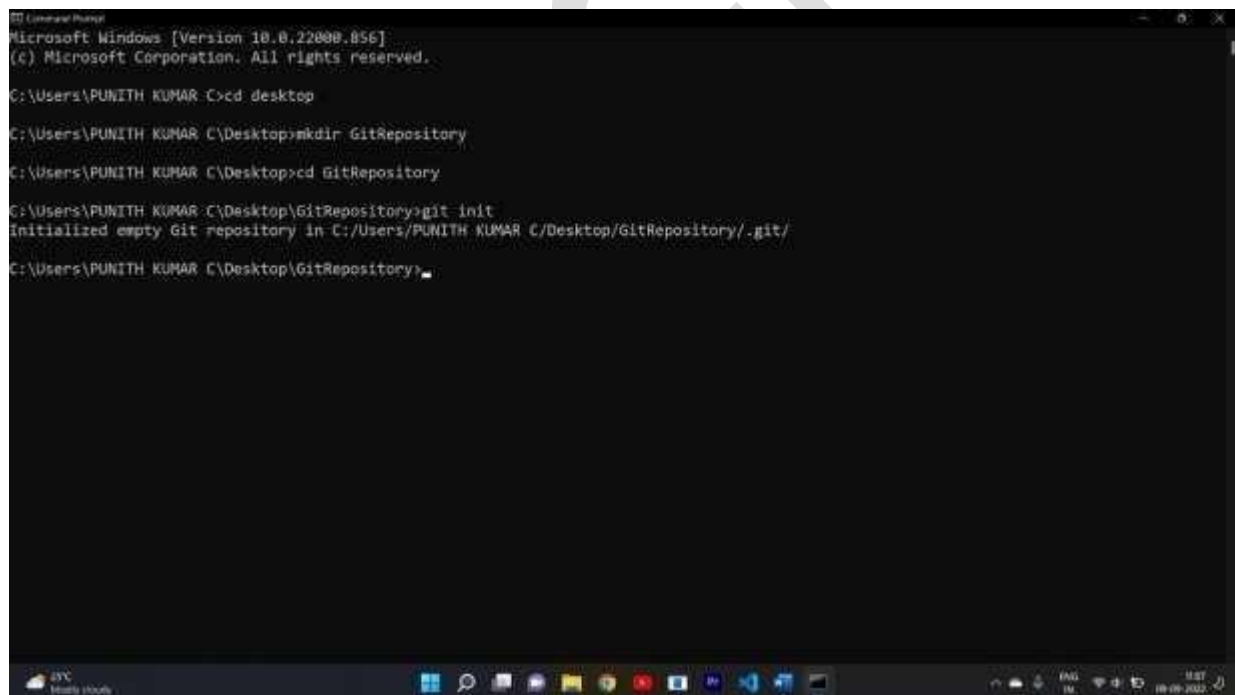
C:\Users\PUNITH KUMAR C>git config --global user.email "punithkumarc777@gmail.com"

C:\Users\PUNITH KUMAR C>git config --global user.email
punithkumarc777@gmail.com

C:\Users\PUNITH KUMAR C>git config --global user.name
Punith Kumar C

C:\Users\PUNITH KUMAR C>
```

Creating Repository in Git:



```
Microsoft Windows [Version 10.0.22000.856]
(c) Microsoft Corporation. All rights reserved.

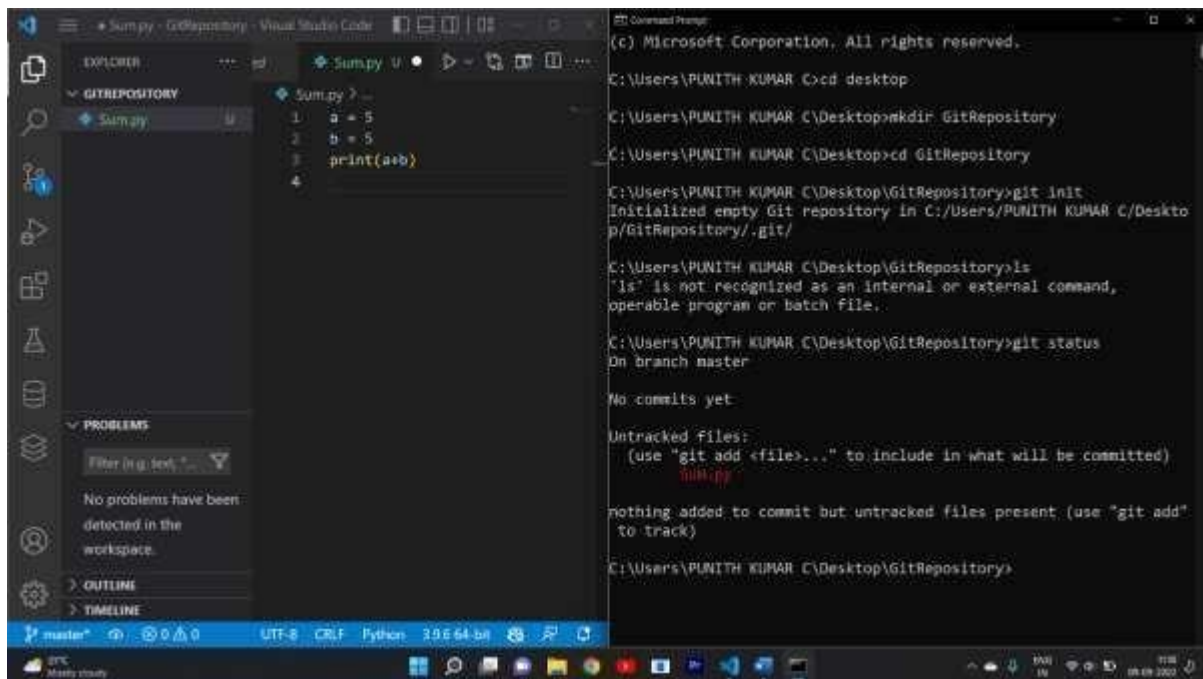
C:\Users\PUNITH KUMAR C>cd desktop

C:\Users\PUNITH KUMAR C\Desktop>mkdir GitRepository

C:\Users\PUNITH KUMAR C\Desktop>cd GitRepository

C:\Users\PUNITH KUMAR C\Desktop\GitRepository>git init
Initialized empty Git repository in C:/Users/PUNITH KUMAR C/Desktop/GitRepository/.git/

C:\Users\PUNITH KUMAR C\Desktop\GitRepository>
```



The screenshot shows the Visual Studio Code interface on the left and a Command Prompt on the right. In VS Code, a file named `Sum.py` is open, containing the following Python code:

```
1 a = 5
2 b = 5
3 print(a+b)
4
```

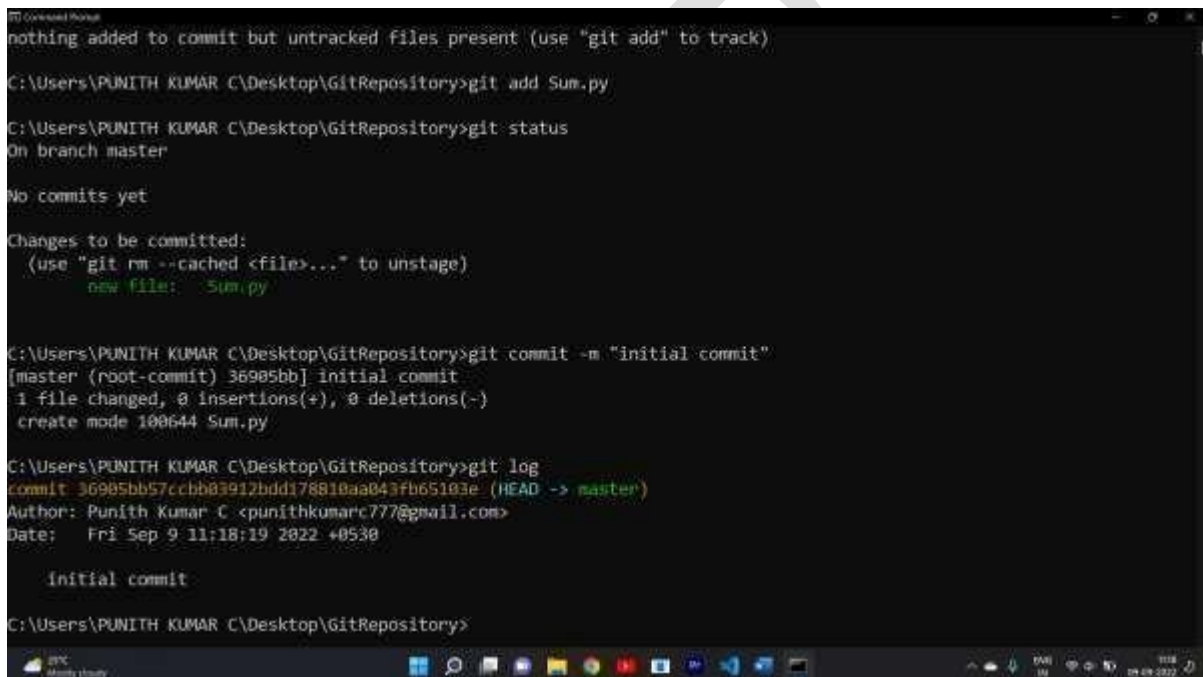
The Command Prompt shows the following commands and their outputs:

```
(C:) Microsoft Corporation. All rights reserved.
C:\Users\PUNITH KUMAR C>cd desktop
C:\Users\PUNITH KUMAR C\Desktop>mkdir GitRepository
C:\Users\PUNITH KUMAR C\Desktop>cd GitRepository
C:\Users\PUNITH KUMAR C\Desktop\GitRepository>git init
Initialized empty Git repository in C:/Users/PUNITH KUMAR C/Desktop/GitRepository/.git/
C:\Users\PUNITH KUMAR C\Desktop\GitRepository>ls
'ls' is not recognized as an internal or external command,
operable program or batch file.
C:\Users\PUNITH KUMAR C\Desktop\GitRepository>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Sum.py

nothing added to commit but untracked files present (use "git add"
to track)
C:\Users\PUNITH KUMAR C\Desktop\GitRepository>
```



The Command Prompt continues with the following commands and outputs:

```
nothing added to commit but untracked files present (use "git add" to track)
C:\Users\PUNITH KUMAR C\Desktop\GitRepository>git add Sum.py
C:\Users\PUNITH KUMAR C\Desktop\GitRepository>git status
On branch master

No commits yet

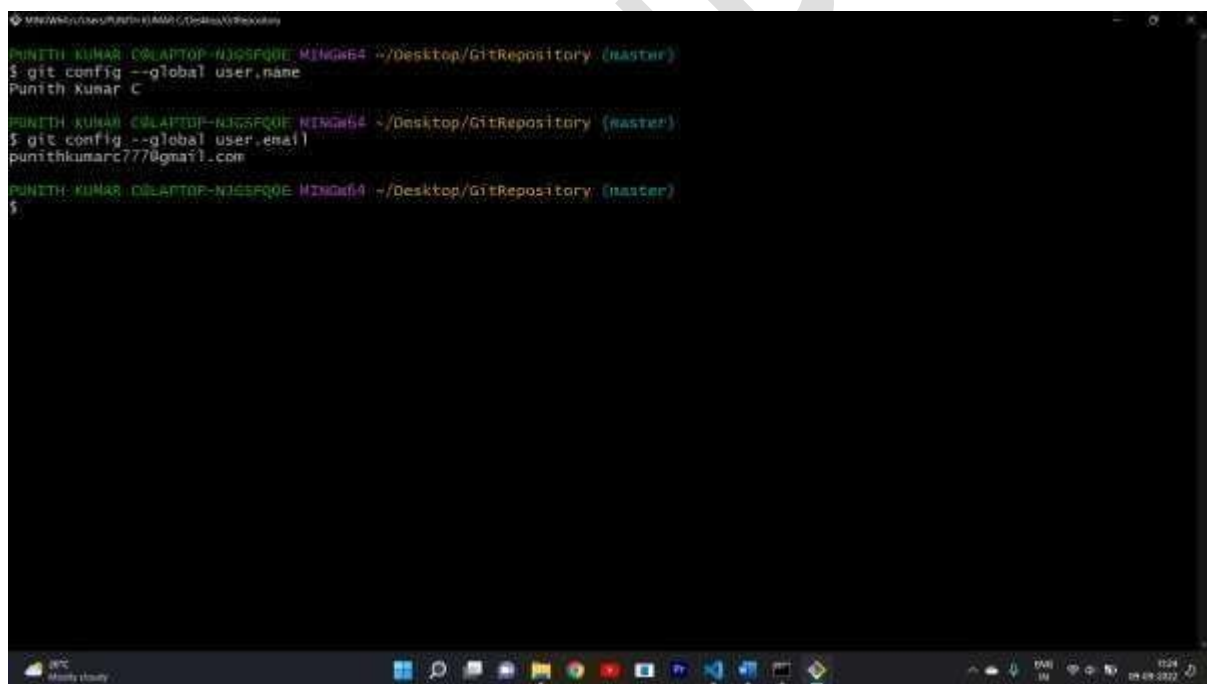
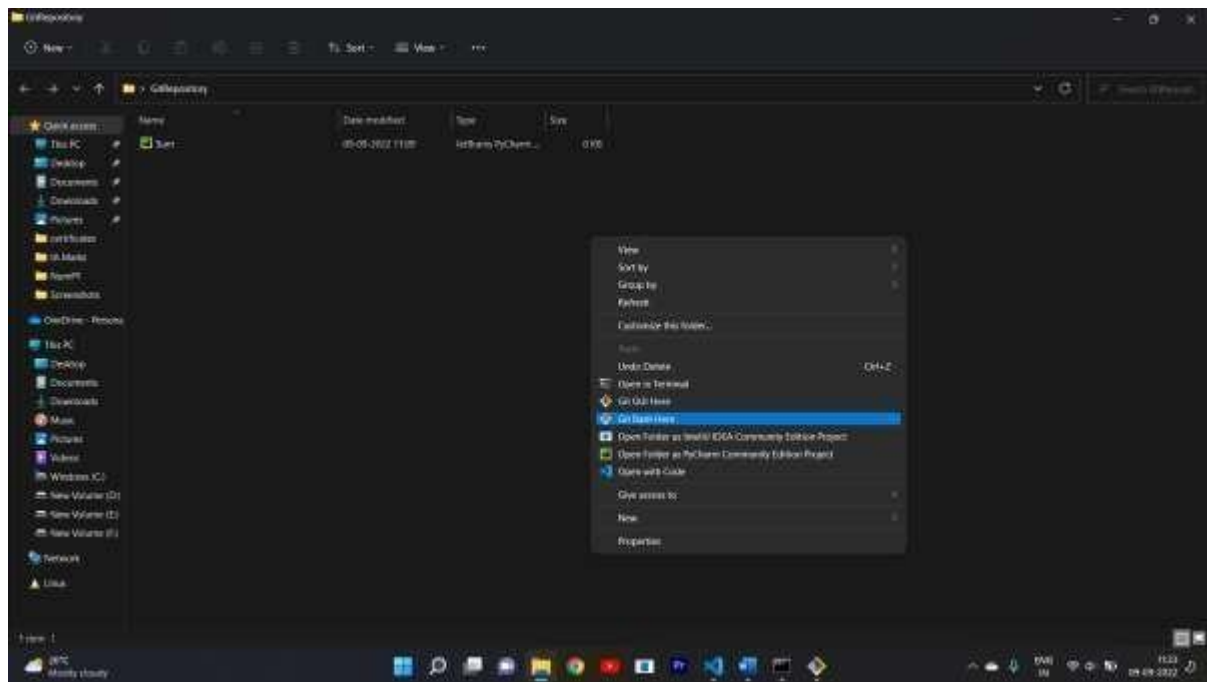
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   Sum.py

C:\Users\PUNITH KUMAR C\Desktop\GitRepository>git commit -m "initial commit"
[master (root-commit) 36905bb] initial commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Sum.py

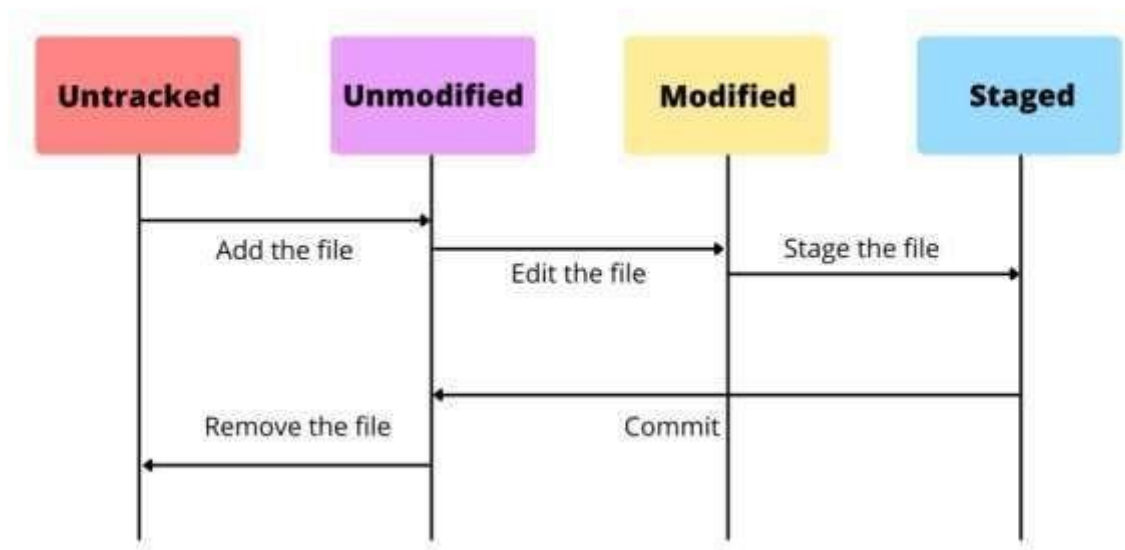
C:\Users\PUNITH KUMAR C\Desktop\GitRepository>git log
commit 36905bb57ccbb83912bdd178818aa043fb65103e (HEAD -> master)
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date:   Fri Sep 9 11:18:19 2022 +0530

    initial commit

C:\Users\PUNITH KUMAR C\Desktop\GitRepository>
```



File Life Cycle in Git



STEPS TO TRACK FILE IN GIT:-

Initial Commit part:

git commit: initial commit

Staging part:

git add -A or git add filename: to add the files to the git

Committing part:

git commit -m "Message": git commit -m "Committed filenames" **IMP:**

Committing all stage files:

Syntax: git commit -a -m "Commit message"

File has four states in Status lifecycle. that are

Untracked

Unmodified

Modified

Staged

1.Untracked state : Files are present in the local directory, but not added in the github repository index.

2.Unmodified state : Files are already present in directory or added using *\$git add* command. If some changes did, then it not get tracked. Also after committing the changes file status become unmodified.

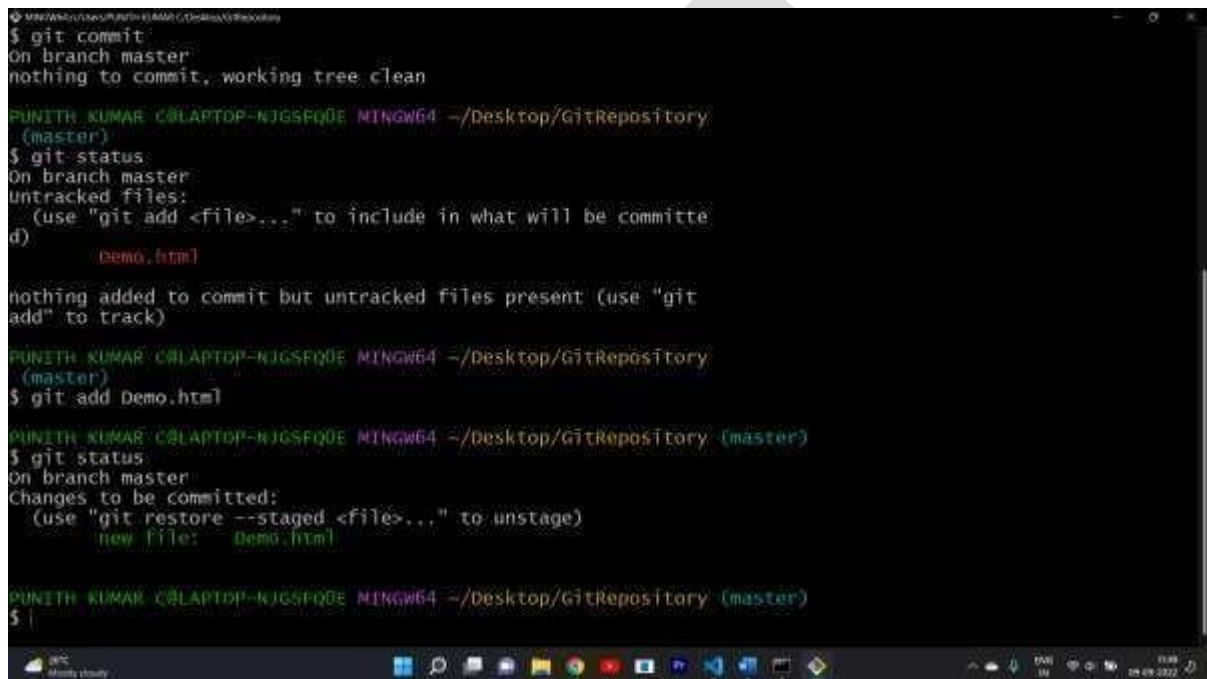
3.Modified state : When previously tracked file is edited, but not commit the changes.

4.Staged state : When files committed and ready to push in git repository, then they have staged status.

We must do initial Commit to track the file in git*****

Git add Demo.html

Git status



```
MINI-WIN64-USER@PUNITH-KUMAR-G:~/Desktop/GitRepository
$ git commit
On branch master
nothing to commit, working tree clean

PUNITH KUMAR C@LAPTOP-NJGSPQOE MINGW64 ~/Desktop/GitRepository
(master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        Demo.html

nothing added to commit but untracked files present (use "git add" to track)

PUNITH KUMAR C@LAPTOP-NJGSPQOE MINGW64 ~/Desktop/GitRepository
(master)
$ git add Demo.html

PUNITH KUMAR C@LAPTOP-NJGSPQOE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   Demo.html

PUNITH KUMAR C@LAPTOP-NJGSPQOE MINGW64 ~/Desktop/GitRepository (master)
$
```

Initial Commit: using “git commit”

In the new bash terminal Press “I” then type “Initial Commit” (This is the message)

```
Initial Commit
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
#
# Changes to be committed:
#   new file:   Demo.html
#

```

.git/COMMIT_EDITMSG [unix] (11:51 09/09/2022) 1,14 A |

Once the commit is done: git status will change to nothing to commit, working tree clean

```
PUNITH KUMAR C@LAPTOP-NJGSEFQ0E MINGW64 ~/Desktop/GitRepository
(master)
$ git status
On branch master
nothing to commit, working tree clean

PUNITH KUMAR C@LAPTOP-NJGSEFQ0E MINGW64 ~/Desktop/GitRepository
(master)
$ git commit
On branch master
nothing to commit, working tree clean

PUNITH KUMAR C@LAPTOP-NJGSEFQ0E MINGW64 ~/Desktop/GitRepository
(master)
$ git status
On branch master
Untracked files:
  (use "git add" to track)
    Demo.html
[master f4128b9] Initial Commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Demo.html

PUNITH KUMAR C@LAPTOP-NJGSEFQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
nothing to commit, working tree clean

PUNITH KUMAR C@LAPTOP-NJGSEFQ0E MINGW64 ~/Desktop/GitRepository (master)
$
```

Adding all files to staging area using “git add -A”:

```
PUNITH KUMAR C@LAPTOP-NJG5FQOE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    about.html
    contact.html
    monuments.html

nothing added to commit but untracked files present (use "git add" to track)

PUNITH KUMAR C@LAPTOP-NJG5FQOE MINGW64 ~/Desktop/GitRepository (master)
$ git add -A

PUNITH KUMAR C@LAPTOP-NJG5FQOE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   about.html
    new file:   contact.html
    new file:   monuments.html

PUNITH KUMAR C@LAPTOP-NJG5FQOE MINGW64 ~/Desktop/GitRepository (master)
$
```

Once the staging is done the files are ready to commit.

If you modify any files, it will show “modified: filename” message: Once you modified the files, you have to again add the files to the staging area by “git add filename” or “git add -A” commands

```
PUNITH KUMAR C@LAPTOP-NJG5FQOE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   about.html
    new file:   contact.html
    new file:   monuments.html

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   about.html
    modified:   contact.html

PUNITH KUMAR C@LAPTOP-NJG5FQOE MINGW64 ~/Desktop/GitRepository (master)
$ git add -A

PUNITH KUMAR C@LAPTOP-NJG5FQOE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   about.html
    new file:   contact.html
    new file:   monuments.html

PUNITH KUMAR C@LAPTOP-NJG5FQOE MINGW64 ~/Desktop/GitRepository (master)
$
```

Here in the above screenshot the modified files are again added to the staging area

Adding files to Commit:

Command: git commit -m "Message"

```
modified: about.html
modified: contact.html

PUNITH KUMAR C:\LAPTOP-NJG5FQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git add -A

PUNITH KUMAR C:\LAPTOP-NJG5FQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   about.html
        new file:   contact.html
        new file:   monuments.html

PUNITH KUMAR C:\LAPTOP-NJG5FQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git commit -m "Added more htmls"
[master e7aa450] Added more htmls
 3 files changed, 24 insertions(+),
 create mode 100644 about.html
 create mode 100644 contact.html
 create mode 100644 monuments.html

PUNITH KUMAR C:\LAPTOP-NJG5FQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
nothing to commit, working tree clean

PUNITH KUMAR C:\LAPTOP-NJG5FQ0E MINGW64 ~/Desktop/GitRepository (master)
$
```

Checkout Command in Git:

“git checkout filename”

Checkout command will reenable or rematches the old commit (If someone changes your files, then you can get that old file contents by “git checkout filename” command)

```
PUNITH KUMAR C:\LAPTOP-NJG5FQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   monuments.html

no changes added to commit (use "git add" and/or "git commit -a"):

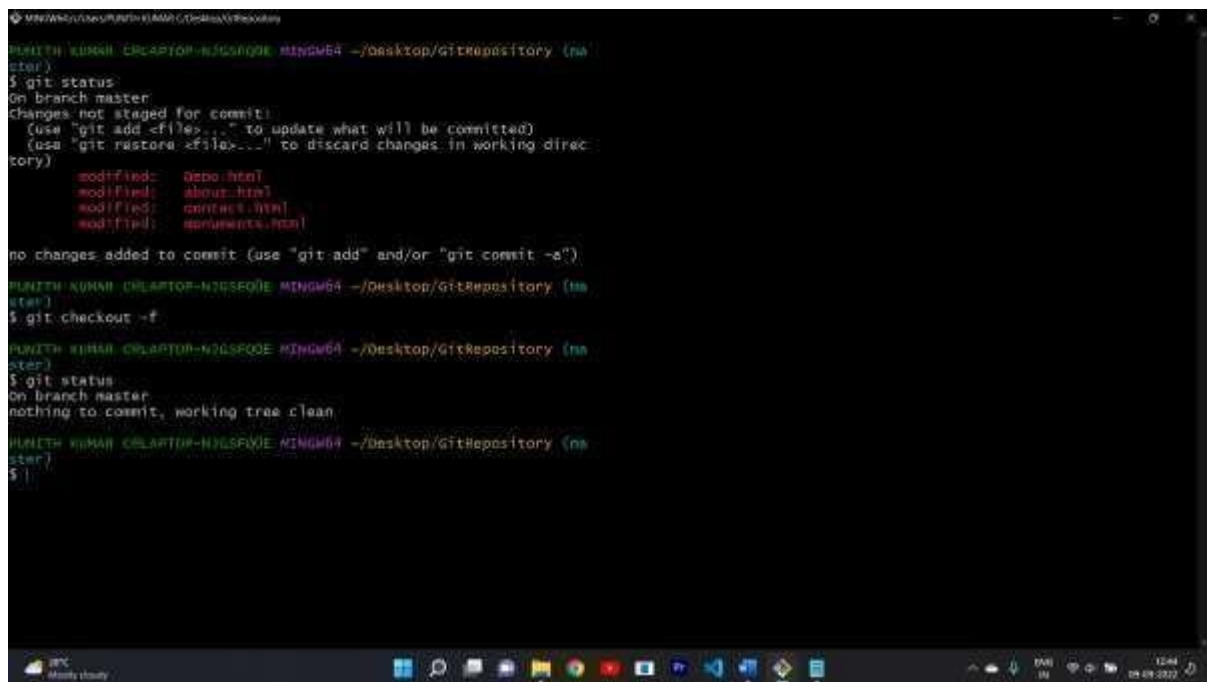
PUNITH KUMAR C:\LAPTOP-NJG5FQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git checkout monuments.html
updated 1 path from the index

PUNITH KUMAR C:\LAPTOP-NJG5FQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
nothing to commit, working tree clean

PUNITH KUMAR C:\LAPTOP-NJG5FQ0E MINGW64 ~/Desktop/GitRepository (master)
$
```

git checkout -f command:

This command is used to retain all the modified file contents:



```
PLNITH KUMAR @LAPTOP-NJGSPQJE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   demo.html
        modified:   about.html
        modified:   contact.html
        modified:   nonuser.html

no changes added to commit (use "git add" and/or "git commit -a")

PLNITH KUMAR @LAPTOP-NJGSPQJE MINGW64 ~/Desktop/GitRepository (master)
$ git checkout -f

PLNITH KUMAR @LAPTOP-NJGSPQJE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
nothing to commit, working tree clean

PLNITH KUMAR @LAPTOP-NJGSPQJE MINGW64 ~/Desktop/GitRepository (master)
$
```

Git log command:

Syntax: “git log”

This command will show all commits of the files.

```
PNITH KUNAR C@LAPTOP-NJG5FQOE: ~/Desktop/GitRepository (master)
$ git log
commit 11ef2b89c11a211d111191884e4eddca12fb41 (HEAD -> master)
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date: Fri Sep 9 12:47:52 2022 +0530

    Changed Demo

commit b16cb756dc477edac0a1ff156ed7ca42a74e154b
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date: Fri Sep 9 12:31:58 2022 +0530

    Commit after deleting of Sum.py

commit 2c747577eeb9fd95791c75499b5d8078e951248b
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date: Fri Sep 9 12:22:06 2022 +0530

    Committed monuments.html

commit e7aa4105fb8490911ecb046e8e409c297546fc5d
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date: Fri Sep 9 12:08:51 2022 +0530

    Added more htmls

commit f4175b9b0b14d751c457b0c81332dc62ac418a72
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date: Fri Sep 9 11:51:15 2022 +0530

    Initial Commit

commit 36905b657ccbb03912b8d178810aa043fb65103a
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date: Fri Sep 9 11:18:19 2022 +0530

    initial commit

PNITH KUNAR C@LAPTOP-NJG5FQOE: ~/Desktop/GitRepository (master)
$
```

Filtering in git log:

We can also filter using:

Syntax: `git log -p -NumberOfCommits`

Example: git log -p -2

```
PUNITH KUMAR @LAPTOP-HJG5FQDE MINGW64 ~/Desktop/GitRepository (master)
$ git log -p -2
commit 11af589c11a211d1119188fe4eddc12f41 (HEAD -> master)
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date: Fri Sep 9 12:47:52 2022 +0530

    Changed Demo

diff --git a/Demo.html b/Demo.html
index a69de29..8d17db6 100644
--- a/Demo.html
+++ b/Demo.html
@@ -0,0 +1 @@
+addfgr
\ No newline at end of file

commit b16cb756dc1f7adac0a1ff158ed7ca47a74e1b4b
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date: Fri Sep 9 12:31:38 2022 +0530

    Commit after deleting of Sum.py

diff --git a/Sum.py b/Sum.py
deleted file mode 100644
index a69de29..0000000

```

Git diff Command:

Syntax: “git diff”: This command is used to compares the working tree to staging area and shows if there any differences.

```
PUNITH KUMAR @LAPTOP-HJG5FQDE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")

PUNITH KUMAR @LAPTOP-HJG5FQDE MINGW64 ~/Desktop/GitRepository (master)
$ git diff
diff --git a/index.html b/index.html
index 92806db..85202d3 100644
--- a/index.html
+++ b/index.html
@@ -1,3 +0,3 @@
-<head>
-  <body>
-    <h1>this is html</h1>
-  </body>
-</html>
+<body>
+  <body body2>
+</body>
\ No newline at end of file

```

Git diff --staged:

This command will compare the staging area to the last commit

Syntax: “git diff --staged”

```
MINIWIN/C:/Users/PUNITH KUMAR/Desktop/GitRepository
PUNITH KUMAR C:\WINDOWS\system32\cmd /c cd /d C:\Desktop\GitRepository (master)
$ git add -A

PUNITH KUMAR C:\WINDOWS\system32\cmd /c cd /d C:\Desktop\GitRepository (master)
$ git diff --staged
diff --git a/index.html b/index.html
index 92806db..8520263 100644
--- a/index.html
+++ b/index.html
@@ -5,3,45,1 98
</head>
<body>
  <h1>this is html</h1>
</body>
+</body> body2>
</html>
\ No newline at end of file

PUNITH KUMAR C:\WINDOWS\system32\cmd /c cd /d C:\Desktop\GitRepository (master)
$
```

Committing files without Staging:

Syntax: “git commit -a -m “Message to commit” Example:

```
MINIWIN/C:/Users/PUNITH KUMAR/Desktop/GitRepository
PUNITH KUMAR C:\WINDOWS\system32\cmd /c cd /d C:\Desktop\GitRepository (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   about.html

no changes added to commit (use "git add" and/or "git commit -a")

PUNITH KUMAR C:\WINDOWS\system32\cmd /c cd /d C:\Desktop\GitRepository (master)
$ git commit -a -m "skipped staging area and fixed <"
[master 2de11f7] skipped staging area and fixed <
1 file changed, 1 insertion(+), 1 deletion(-)

PUNITH KUMAR C:\WINDOWS\system32\cmd /c cd /d C:\Desktop\GitRepository (master)
$ git log
commit 2de11f7fac7c67990c532de2d5ad7f047e88 (HEAD -> master)
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date:   Fri Sep 9 13:22:35 2022 +0530

    skipped staging area and fixed <

commit 739fac733c1d18793cddd6a6b18af474d7bfa
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date:   Fri Sep 9 13:02:04 2022 +0530

    Committed Index.html

commit 8b9eb41453767ef4678ca1469d83c4f007eb96d3
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date:   Fri Sep 9 12:58:26 2022 +0530

    Committed Index.html

commit 31af2b69c11a211d11101886e4eddc12fb41
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date:   Fri Sep 9 12:47:52 2022 +0530

    changed Desc
```

Git rm filename command:

This command is used to remove the file from both working directory and git staging area.

Syntax: “git rm filename”

```
PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ touch demo2.html

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ ls
demo.html  about.html  contact.html  demo2.html  index.html  monuments.html

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ git add -A

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ git commit -a -m "Adding demo2.html"
[master 3e7659e] Adding demo2.html
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 demo2.html

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ git rm demo2.html
rm 'demo2.html'

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ ls
demo.html  about.html  contact.html  index.html  monuments.html

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    demo2.html

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$
```

Command removes file from only staging area and present in hard disk/directory:

Syntax: “git rm --cached filename” Example:

```
PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ ls
demo.html  contact.html  index.html
about.html  demo1.html  monuments.html

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
nothing to commit, working tree clean

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ git rm --cached demo1.html
rm 'demo1.html'

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    demo1.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        demo1.html

PUNITH KUMAR C:\A\TOP-N\GSP\QOE MINGW64 ~/Desktop/GitRepository (master)
$
```

Checking the modification of a file:

Syntax: “git status -s”

The green coloured M denotes that the file is modified in the staging area. (In the below image)

The Red coloured M denotes that the file is modified in the working tree. (In the below image)

```
rename from demo2.html
rename to demo1.html

RUPNITH KUMAR CHAKRABORTY-NUGSPQDE MINGW64 ~/Desktop/GitRepository
(master)
$ git status
On branch master
changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   demo.html
        modified:   about.html
        modified:   contact.html
        modified:   index.html
        modified:   monuments.html

no changes added to commit (use "git add" and/or "git commit -a")

RUPNITH KUMAR CHAKRABORTY-NUGSPQDE MINGW64 ~/Desktop/GitRepository (master)
$ git status -s
M Demo.html
M about.html
M contact.html
M index.html
M monuments.html

RUPNITH KUMAR CHAKRABORTY-NUGSPQDE MINGW64 ~/Desktop/GitRepository (master)
$ git add Demo.html

RUPNITH KUMAR CHAKRABORTY-NUGSPQDE MINGW64 ~/Desktop/GitRepository (master)
$ git status -s
M Demo.html
M about.html
M contact.html
M index.html
M monuments.html
}
```

Git Ignore file:

This file is used to ignore the unwanted files like log files that are present in the same folder or another folder etc.

Steps to create .gitignore file:

- **touch .gitignore**
- Once the file is created, insert the file names that you want to ignore and save it.
Example:

```
updated 1 path from the index

PUNITH KUMAR CHALAPATI-NIGESFODE MINGW64 ~/Desktop/GitRepository (master)
$ git checkout index.html
updated 1 path from the index

PUNITH KUMAR CHALAPATI-NIGESFODE MINGW64 ~/Desktop/GitRepository (master)
$ git checkout monuments.html
updated 1 path from the index

PUNITH KUMAR CHALAPATI-NIGESFODE MINGW64 ~/Desktop/GitRepository (master)
$ git checkout Demo.html
updated 0 paths from the index

PUNITH KUMAR CHALAPATI-NIGESFODE MINGW64 ~/Desktop/GitRepository (master)
$ touch mylog.log

PUNITH KUMAR CHALAPATI-NIGESFODE MINGW64 ~/Desktop/GitRepository (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   Demo.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        loggy
        mylog.log

PUNITH KUMAR CHALAPATI-NIGESFODE MINGW64 ~/Desktop/GitRepository (master)
$
```

If we want to ignore the unwanted files only where the .gitignore file is present:

- Create a .gitignore file using touch command
- Then in the .gitignore file type /filename (/To represent that to ignore the files only present in the .gitignore file's folder.

To ignore the directory:

In the .gitignore file type foldername/.

Git branches:

In Git, branches are a part of your everyday development process. Git branches are effectively a pointer to a snapshot of your changes. When you want to add a new feature or fix a bug—no matter how big or how small—you spawn a new branch to encapsulate your changes.

We basically use branches to modify the project or add new features if the master branch doesn't want the changes, then we can simply commit to the master branch by committing to master branch. **By "git commit master" then we can retrieve the original file content present in the master branch.**

Creating New Branches:

Syntax: "git branch branchName" Example:


```
PS C:\Users\Kunal\Documents> cd Desktop\GitRepo1
copy (master)
$ git branch
* master
  feature2

PS C:\Users\Kunal\Documents> cd Desktop\GitRepo1
copy (master)
$ git checkout feature2
Switched to branch 'feature2'

PS C:\Users\Kunal\Documents> cd Desktop\GitRepo1
copy (feature2)
$ git status
On branch feature2
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html
no changes added to commit (use "git add" and/or "git commit -a")

PS C:\Users\Kunal\Documents> cd Desktop\GitRepo1
copy (feature2)
$ git add -A

PS C:\Users\Kunal\Documents> cd Desktop\GitRepo1
copy (feature2)
$ git commit -m "Added < to index.html by feature2 branch"
[feature2 11c76a1] Added < to index.html by feature2 branch
1 file changed, 1 insertion(+), 1 deletion(-)

PS C:\Users\Kunal\Documents> cd Desktop\GitRepo1
copy (feature2)
$ git status
On branch feature2
nothing to commit, working tree clean

PS C:\Users\Kunal\Documents> cd Desktop\GitRepo1
copy (feature2)
$ git checkout master
Switched to branch 'master'
```

Here in the above image, we created a new branch named feature2.

Then we made some changes to the index.html file (modified) then added the file to the staging part, then we committed the same file.

Now if the master branch doesn't want that changes, we simply changed to master branch, by **"git checkout master"** then the changes are removed from the file, (here index.html)

Once the master branch is agreed with the changes by branch feature1 then we can merge the feature1 branch changes in the master branch, using git merge branch command.

Example:


```
MINGW64:/c/Users/PUNITH KUMAR C/Desktop/GitRepository
PUNITH KUMAR C@LAPTOP-NJGSFQ0E MINGW64 ~/Desktop/GitRepository (feature1)
$ git checkout master
Switched to branch 'master'

PUNITH KUMAR C@LAPTOP-NJGSFQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git log -p -2
commit 21fc6fa201bc6b69bac294cda7fec0d416d1919b (HEAD -> master)
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date:   Fri Sep 9 16:41:20 2022 +0530

    this change is done by master branch

diff --git a/index.html b/index.html
index 92806db..bcd6ee0 100644
--- a/index.html
+++ b/index.html
@@ -9,4 +9,4 @@
<body>
  <h1>this is html</h1>
</body>
-</html>
\ No newline at end of file
+</html>>
\ No newline at end of file

commit a17095557c58fbda4dc79680eda4f791141485
Author: Punith Kumar C <punithkumarc777@gmail.com>
Date:   Fri Sep 9 16:16:27 2022 +0530

    added logs folder to the gitignore file

diff --git a/.gitignore b/.gitignore
index e69de29..536a9bc 100644
--- a/.gitignore
+++ b/.gitignore
@@ -0,0 +1,2 @@
+mylogs.log
+logs/
\ No newline at end of file

PUNITH KUMAR C@LAPTOP-NJGSFQ0E MINGW64 ~/Desktop/GitRepository (master)
$ git merge feature1
Auto-merging index.html
Merge made by the 'ort' strategy.
 index.html | 1 +
 1 file changed, 1 insertion(+)

PUNITH KUMAR C@LAPTOP-NJGSFQ0E MINGW64 ~/Desktop/GitRepository (master)
$
```

Here in this image, we first checkout as **master** then we gave the command:

“git merge feature1” once the merge is done, the changes of the branch will added to the master branch.

If we create a new branch and add a new file in the Repository, then stage the file and commit it, once we checkout to the master branch, then the file of the new branch will get deleted, and once you again checkout to the new branch, the file of the new branch will retain.

Example:

```
MINOW64/~/Users/PUNITH KUMAR C/Desktop/GitRepository
PUNITH KUMAR C:\LAPTOP-NJG5FQDE\MINOW64 ~\Desktop/GitRepository (flaskIntegration)
$ git status
On branch flaskIntegration
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   __init__.py

PUNITH KUMAR C:\LAPTOP-NJG5FQDE\MINOW64 ~\Desktop/GitRepository (flaskIntegration)
$ git commit -a -m "This is done by flaskIntegration branch"
[flaskIntegration 196a7bf] This is done by flaskIntegration branch
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 __init__.py

PUNITH KUMAR C:\LAPTOP-NJG5FQDE\MINOW64 ~\Desktop/GitRepository (flaskIntegration)
$ git status
On branch flaskIntegration
nothing to commit, working tree clean

PUNITH KUMAR C:\LAPTOP-NJG5FQDE\MINOW64 ~\Desktop/GitRepository (flaskIntegration)
$ git checkout master
Switched to branch 'master'

PUNITH KUMAR C:\LAPTOP-NJG5FQDE\MINOW64 ~\Desktop/GitRepository (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   about.py

no changes added to commit (use "git add" and/or "git commit -a")

PUNITH KUMAR C:\LAPTOP-NJG5FQDE\MINOW64 ~\Desktop/GitRepository (master)
$ git commit -a -m "Modified about"
[master 982936d] Modified about
 1 file changed, 1 insertion(+)

PUNITH KUMAR C:\LAPTOP-NJG5FQDE\MINOW64 ~\Desktop/GitRepository (master)
$ git checkout flaskIntegration
Switched to branch 'flaskIntegration'

PUNITH KUMAR C:\LAPTOP-NJG5FQDE\MINOW64 ~\Desktop/GitRepository (flaskIntegration)
$
```

Here in the image we created the “**flaskIntegration**” branch, then we created a file and then staged the file and then we committed the file. But once we checkout to the master branch, the file of the “**flaskIntegration**” branch will be deleted, once again we checkout to the “**flaskIntegration**” branch, the file will be retained.

By this we can differentiate between the files and branches and not mess up with the master branch files.

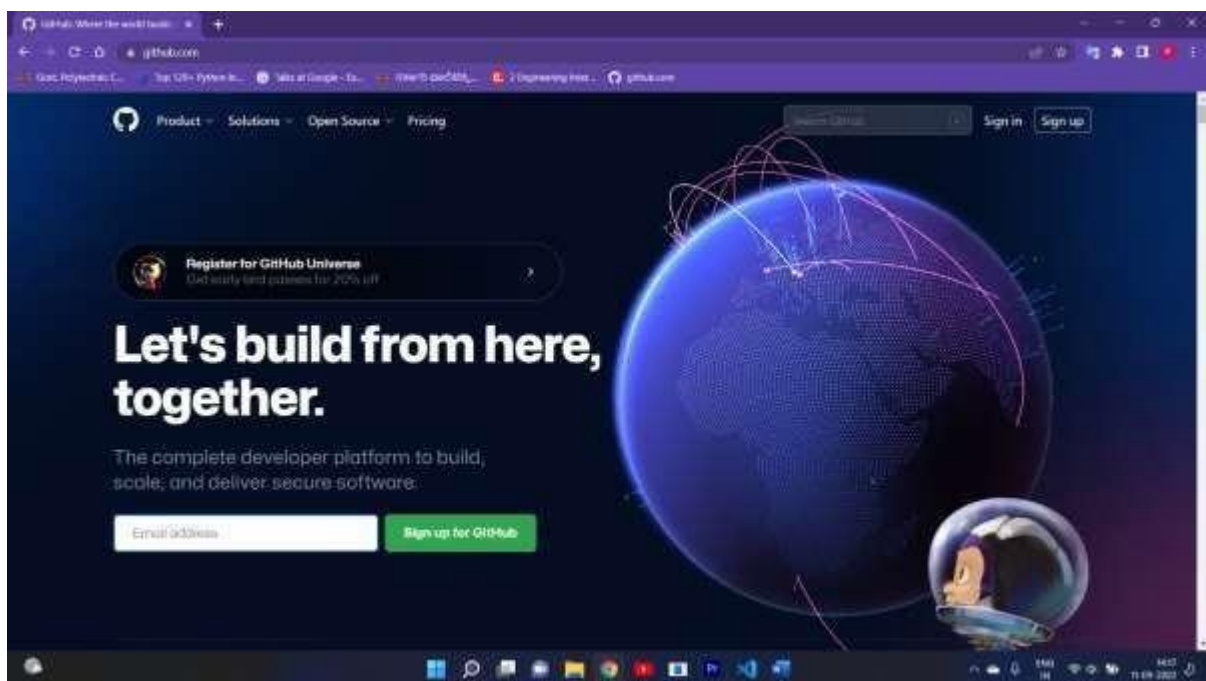
GitHub:

GitHub is a code hosting platform for **version control and collaboration like Git**. It lets you and others work together on projects from anywhere. This tutorial teaches you GitHub essentials like repositories, branches, commits, and pull requests.

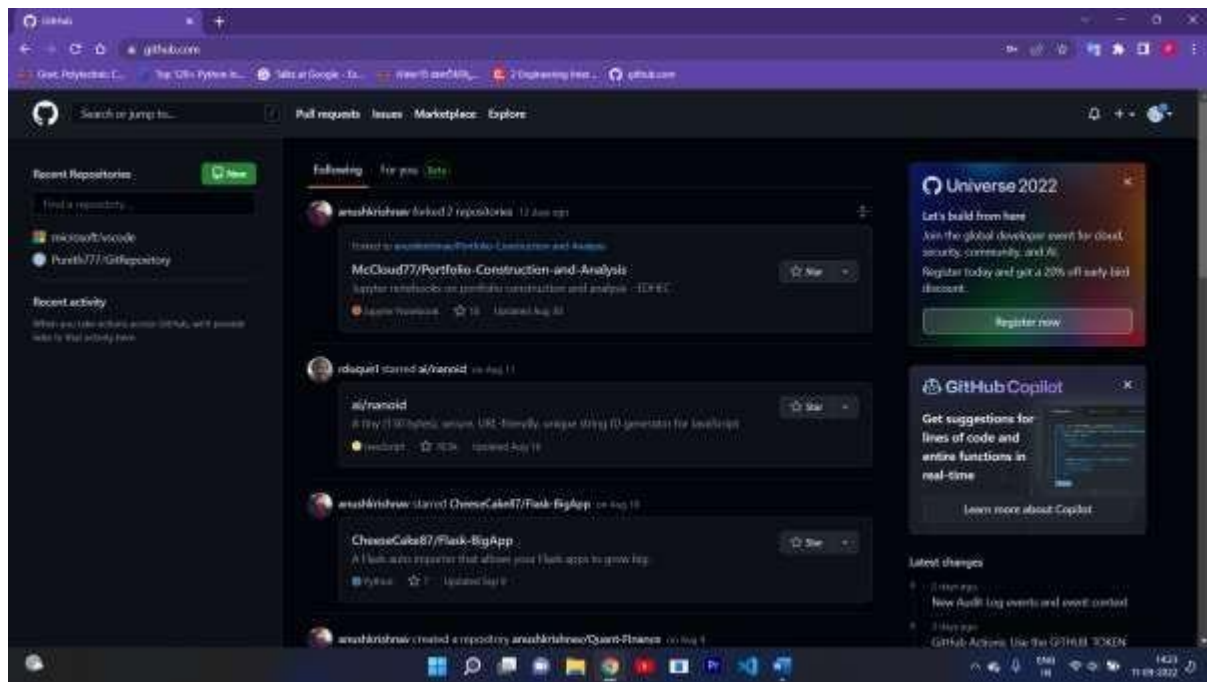
Creating a new account in Git:

Steps:

- Open any browser and open the link: <https://github.com>
- Enter your email address, then click on “Sign up for GitHub”
- Once you enter the new page, if you have an account click on sign in, else, Click on Create New Account and fill up the required details, create a new account.
- You have created a new GitHub Account.

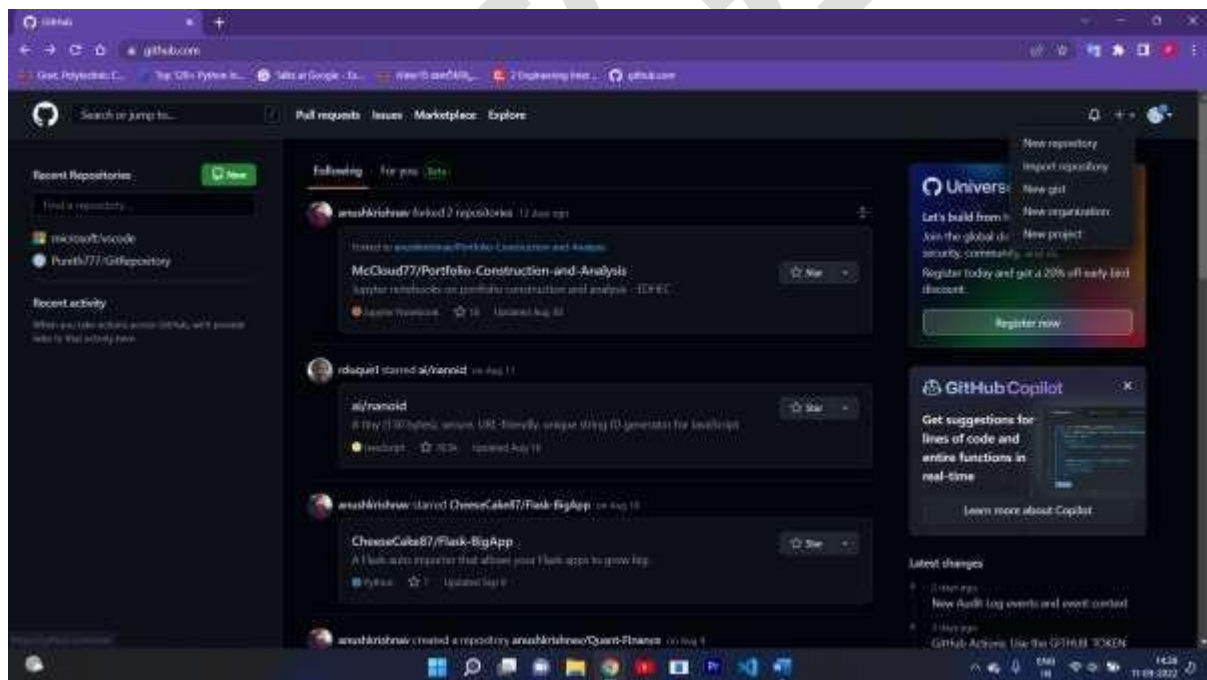


Once you successfully created an account, the page will look like this:

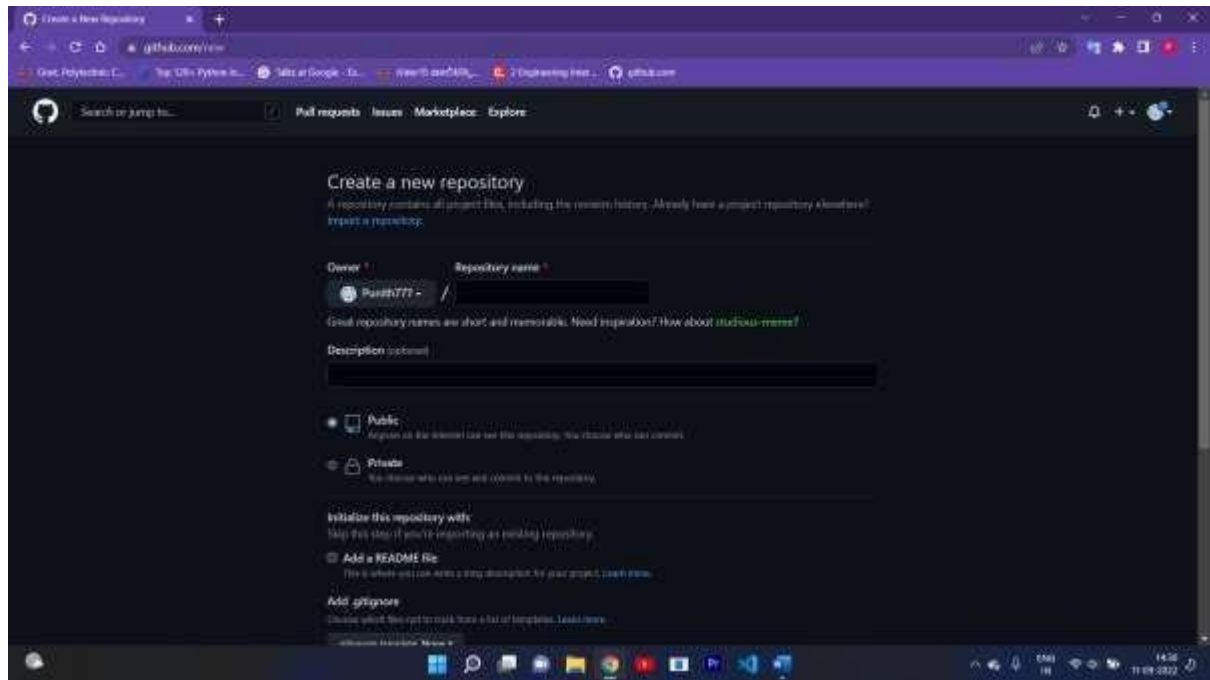


1. Creating a new Repository:

Click on “+” icon and the click on “New repository”



Enter the details, select the repository type, then click on “Create Repository”



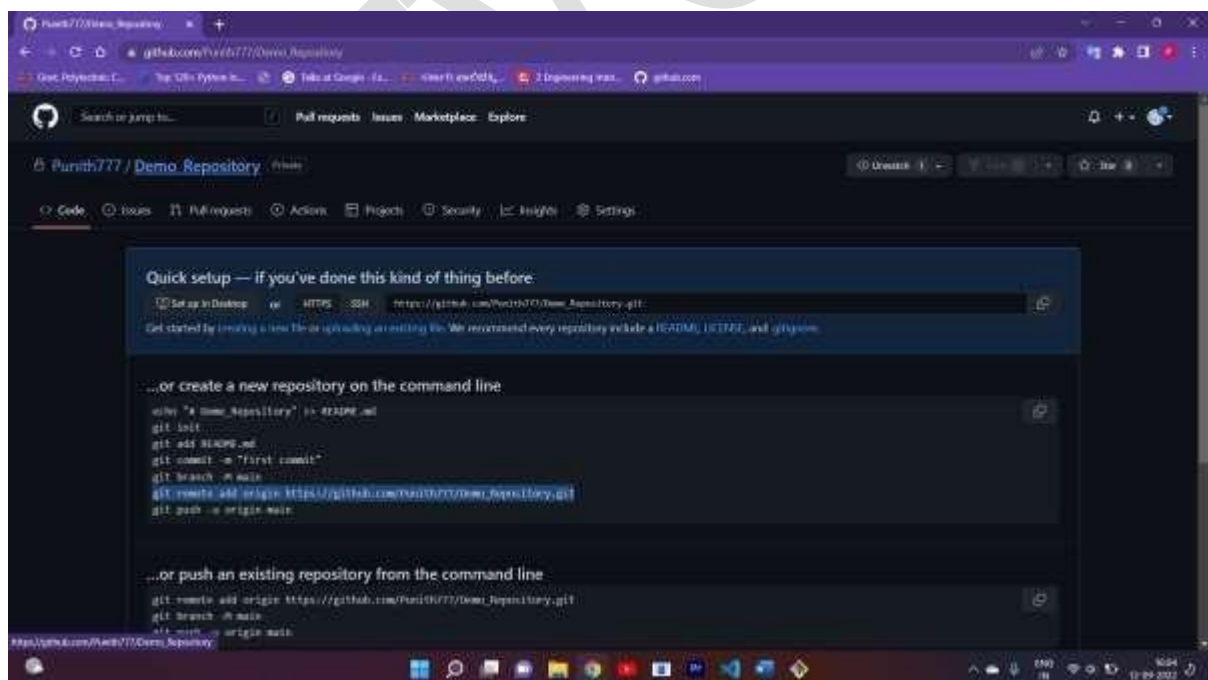
Now using above steps, we have created a remote repository using GitHub.

Now let's push the local Repository to the remote repository:

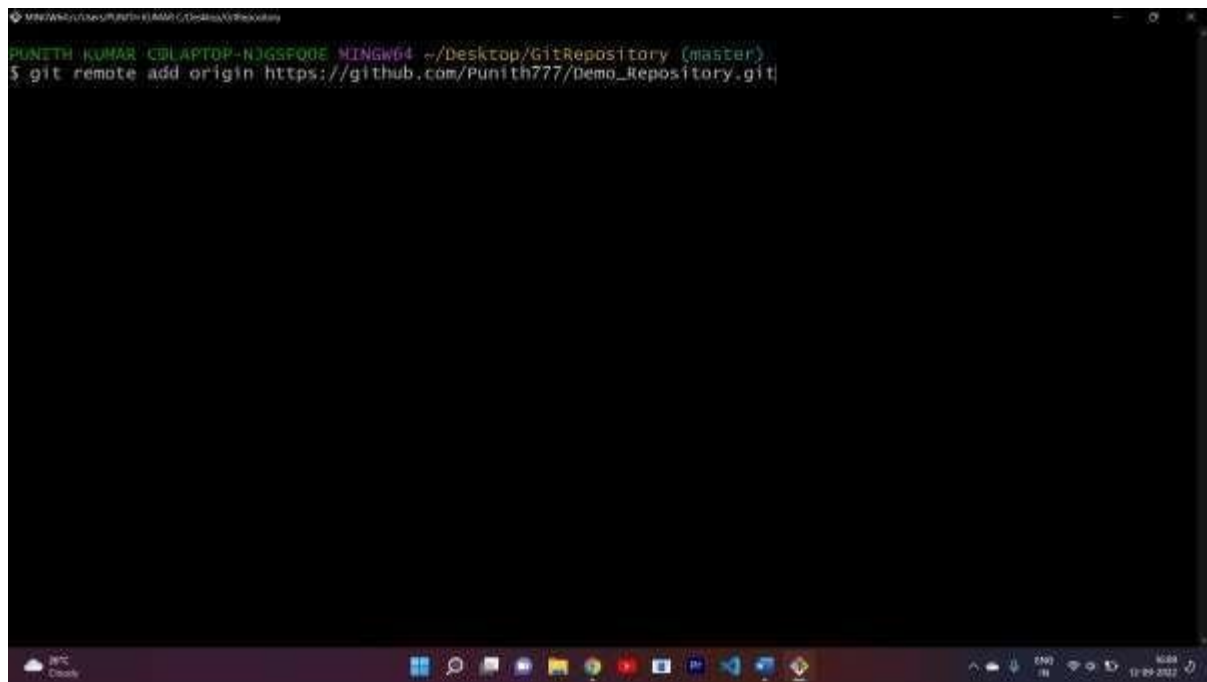
Step-1:

Copy the git remote add origin link of that specific repository created using GitHub.

Here the repository name is DemoRepository.



Paste that link on the git bash terminal opened in the path of that specific folder.

A screenshot of a Windows command prompt window. The title bar reads 'MINGW64: C:\Users\PUNITH KUMAR\Desktop\GitRepository'. The prompt shows the user is in the directory 'C:\Desktop\GitRepository' on the 'master' branch. The command entered is '\$ git remote add origin https://github.com/Punith777/Demo_Repository.git'. The command prompt is running on a Windows 10 desktop, with the taskbar visible at the bottom showing various application icons and the system clock at 11:59 AM on 11-09-2022.

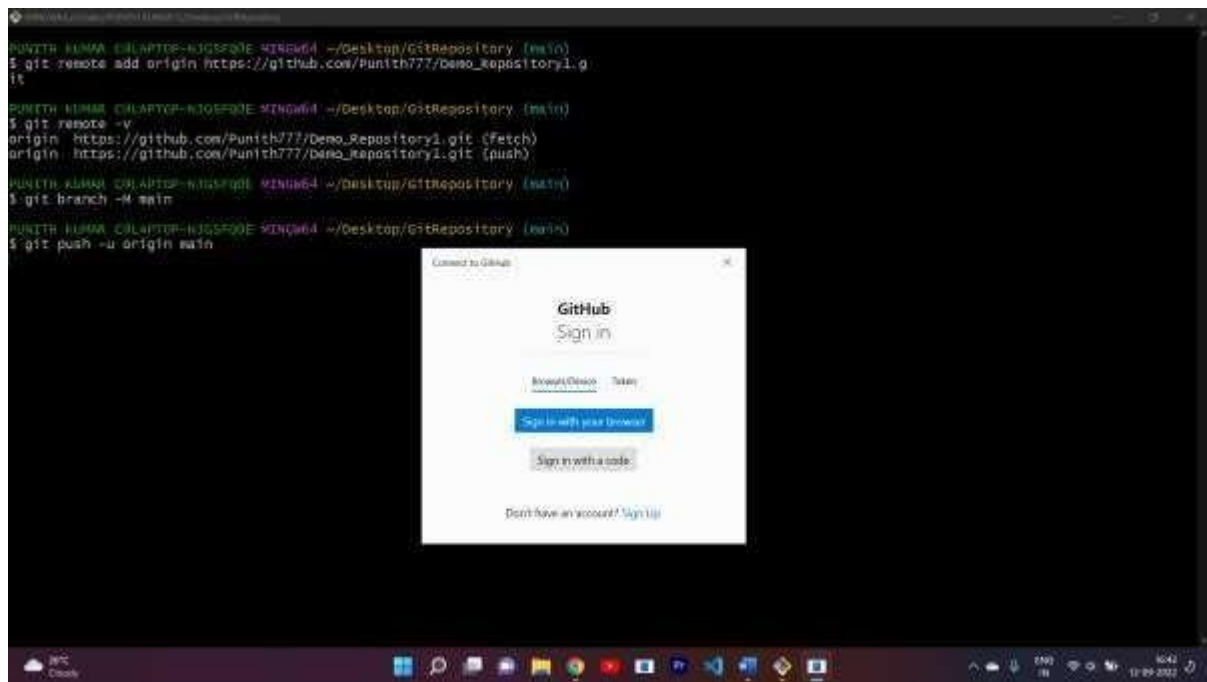
```
MINGW64: C:\Users\PUNITH KUMAR\Desktop\GitRepository
PUNITH KUMAR C:\LAPTOP-NJG5F00E\MINGW64 ~/\Desktop/GitRepository (master)
$ git remote add origin https://github.com/Punith777/Demo_Repository.git
```

- git remote command to check the remote repositories.

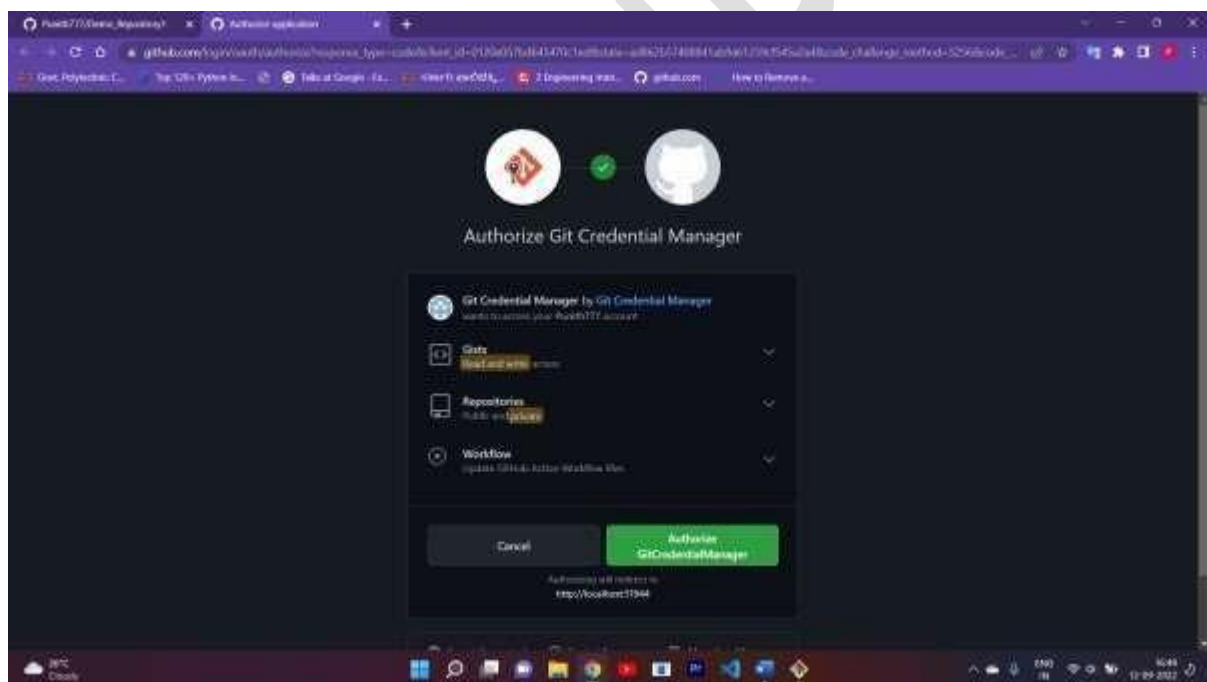
Follow the below steps to push the existing repository from the git bash terminal or command line. The first line of the below code will change from repository to repository, this code will pop up once you create a repository.

- git remote add origin https://github.com/Punith777/Demo_Repository.git
- git branch -M main
- git push -u origin main

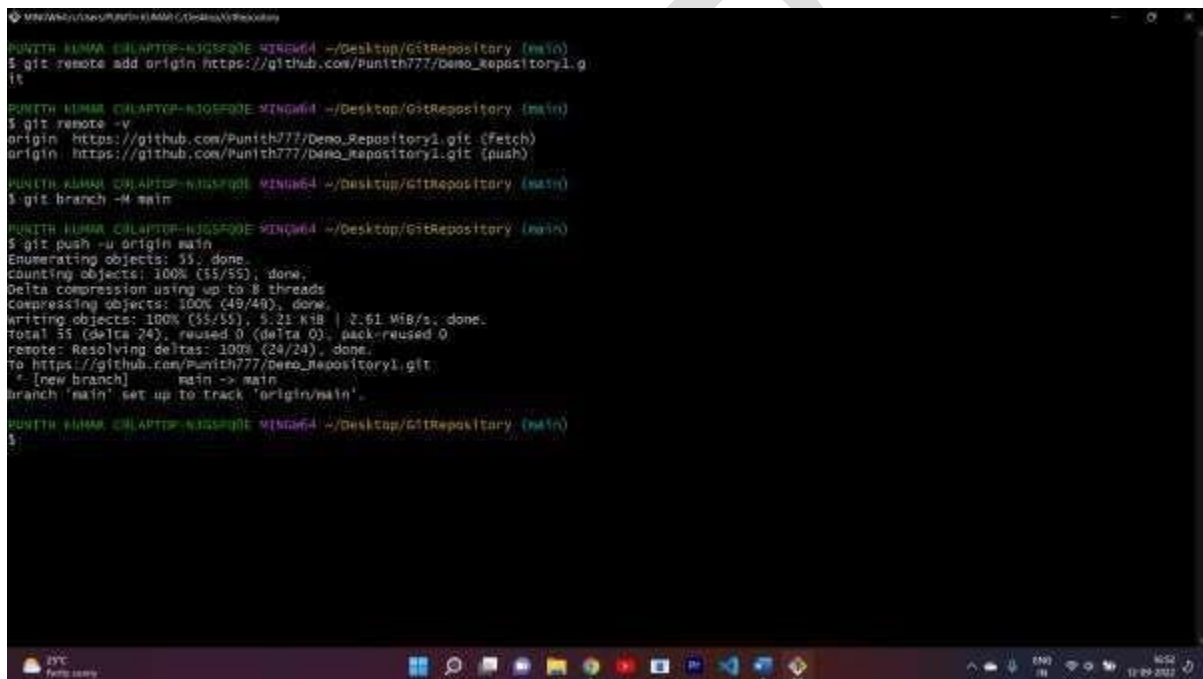
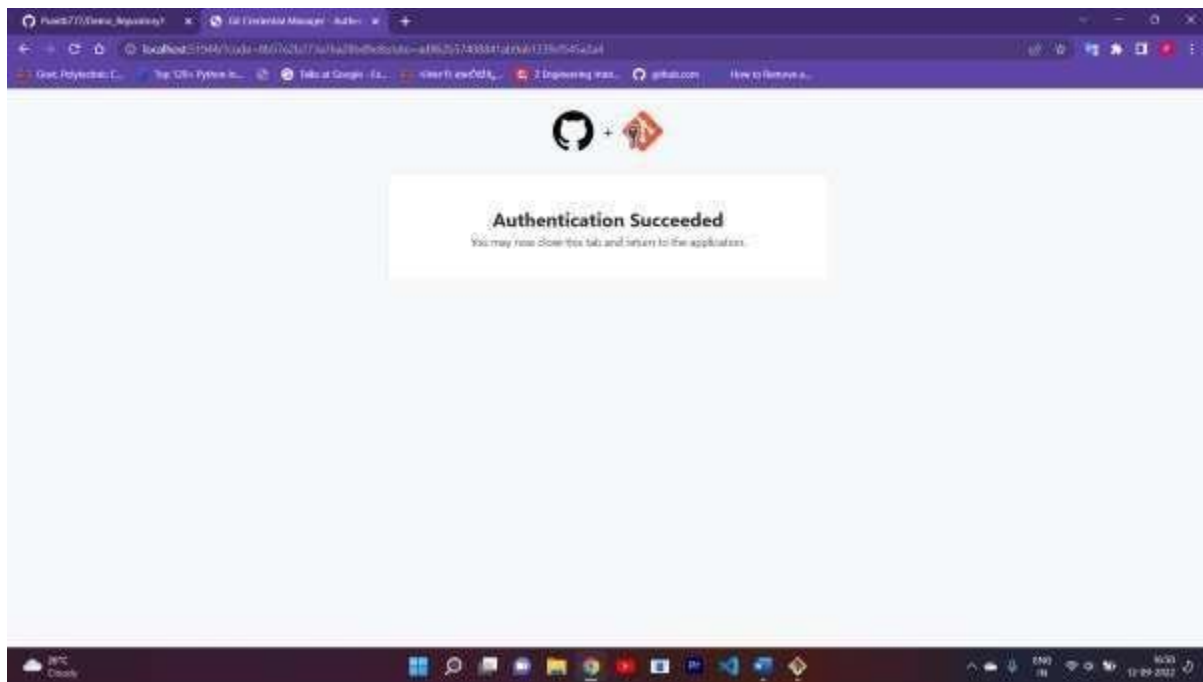
Once you run these commands in Git terminal, a Connect to GitHub window will pop up, then sign in with your id and password, then the repository will be uploaded.



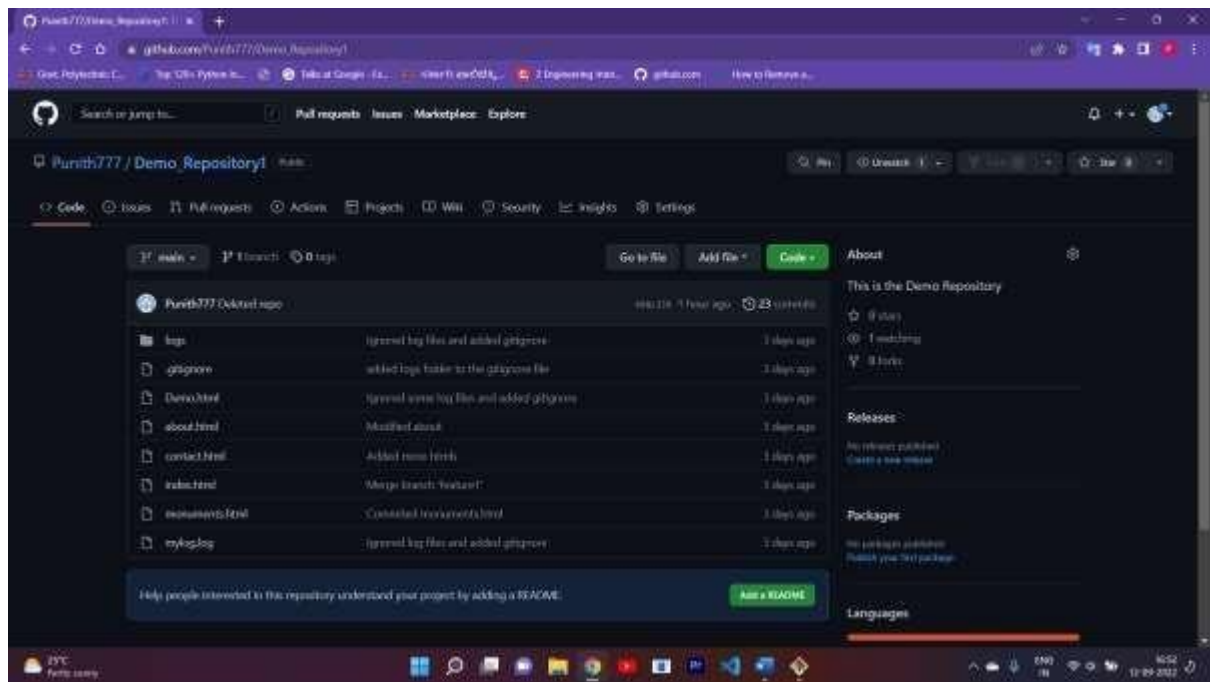
Click on Authorize Git Credential Manager and enter your password



Once you enter the password, the file will be uploaded to the repository.

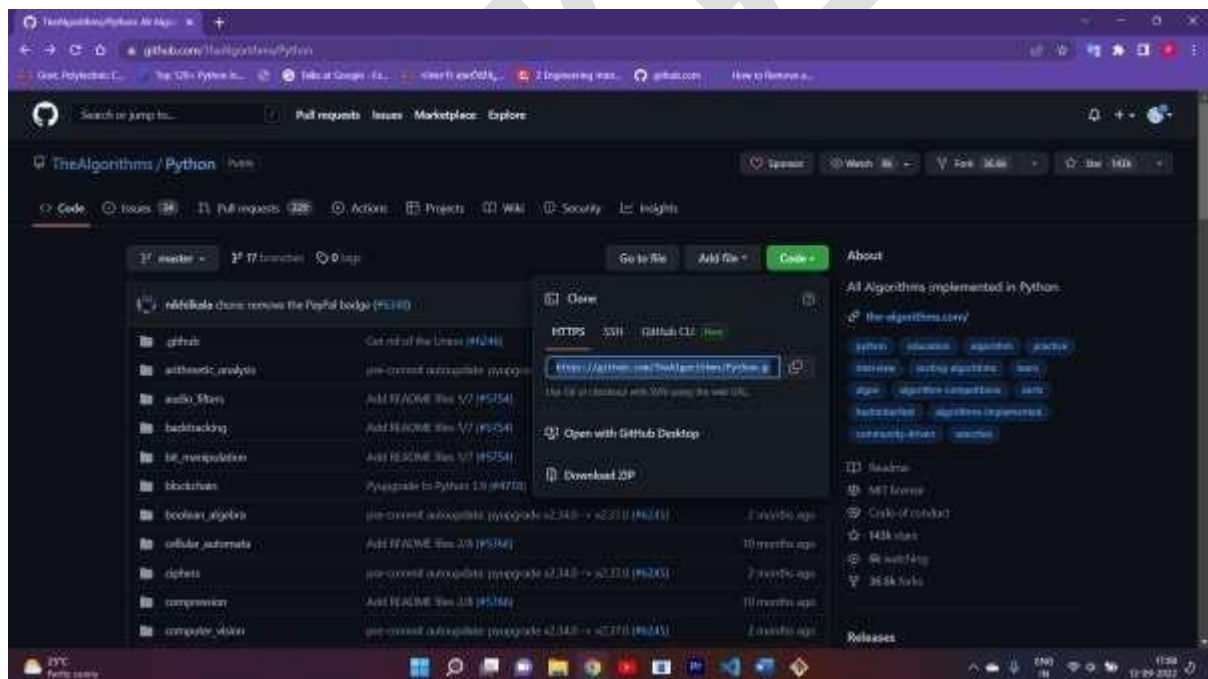


Once you refresh the GitHub website, the files are uploaded.



Cloning a Repository:

Copy the Repository URL form the Code menu:



Then open the git bash terminal in the desired folder path and type “git clone <repo url>

```
MINI KUNAR C:\Users\KUNAR\Desktop\Clone_GitHub_Repo
$ cd Clone_GitHub_Repo

MINI KUNAR C:\Users\KUNAR\Desktop\Clone_GitHub_Repo
$ git clone https://github.com/TheAlgorithm/Python.git
```

Once the clone is done you can view the all files of that repository in the folder.

```
MINI KUNAR C:\Users\KUNAR\Desktop\Clone_GitHub_Repo
$ cd Clone_GitHub_Repo

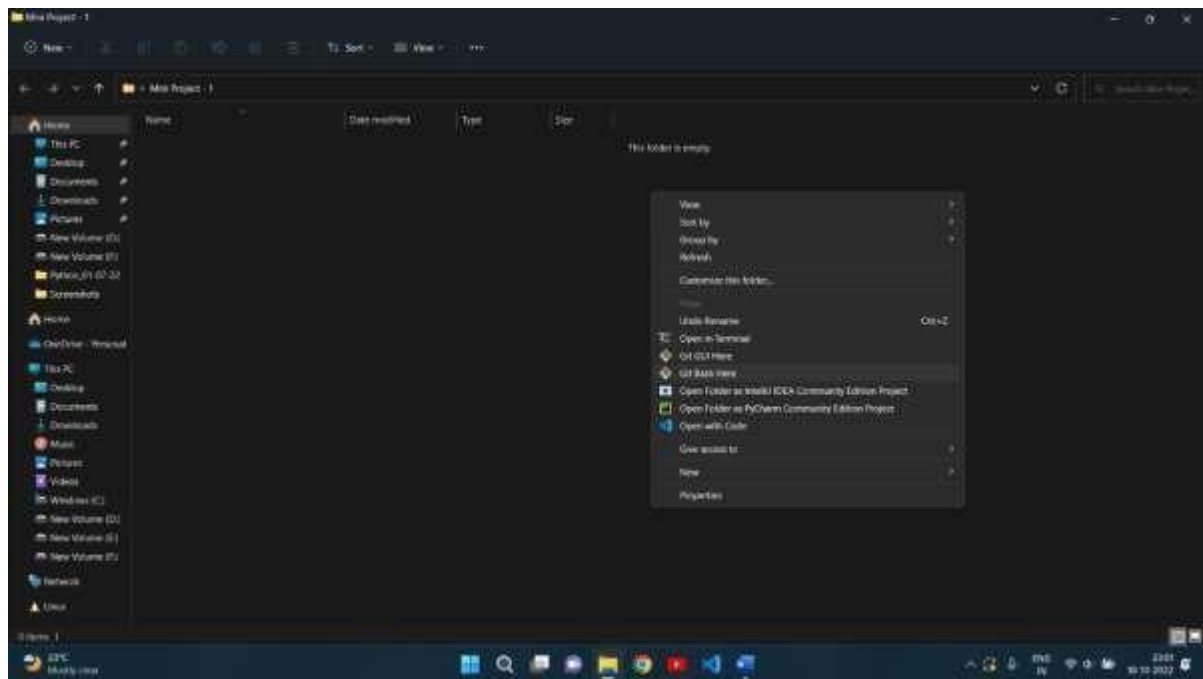
MINI KUNAR C:\Users\KUNAR\Desktop\Clone_GitHub_Repo
$ git clone https://github.com/TheAlgorithm/Python.git
Cloning into 'Python'...
remote: Enumerating objects: 13892, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (5/5), done.
Receiving objects: 19% (2760/13892), 3.40 MiB | 474.00 KiB/s
Receiving objects: 19% (2760/13892), 3.92 MiB | 504.00
Receiving objects: 19% (2760/13892), 4.14 MiB | 454.00
Receiving objects: 19% (2760/13892), 4.56 MiB | 423.00
Receiving objects: 19% (2760/13892), 4.57 MiB | 300.00
Receiving objects: 19% (2760/13892), 4.75 MiB | 237.00
Receiving objects: 19% (2760/13892), 5.11 MiB | 226.00
Receiving objects: 19% (2760/13892), 5.20 MiB | 192.00
Receiving objects: 19% (2760/13892), 5.21 MiB | 177.00
Receiving objects: 19% (2760/13892), 5.23 MiB | 136.00
Receiving objects: 19% (2760/13892), 5.25 MiB | 93.00
remote: Total 13892 (delta 0), reused 2 (delta 0), pack-reused 13887
Receiving objects: 100% (13892/13892), 12.35 MiB | 224.00 KiB/s, done.
Resolving deltas: 100% (7881/7881), done.

MINI KUNAR C:\Users\KUNAR\Desktop\Clone_GitHub_Repo
$
```

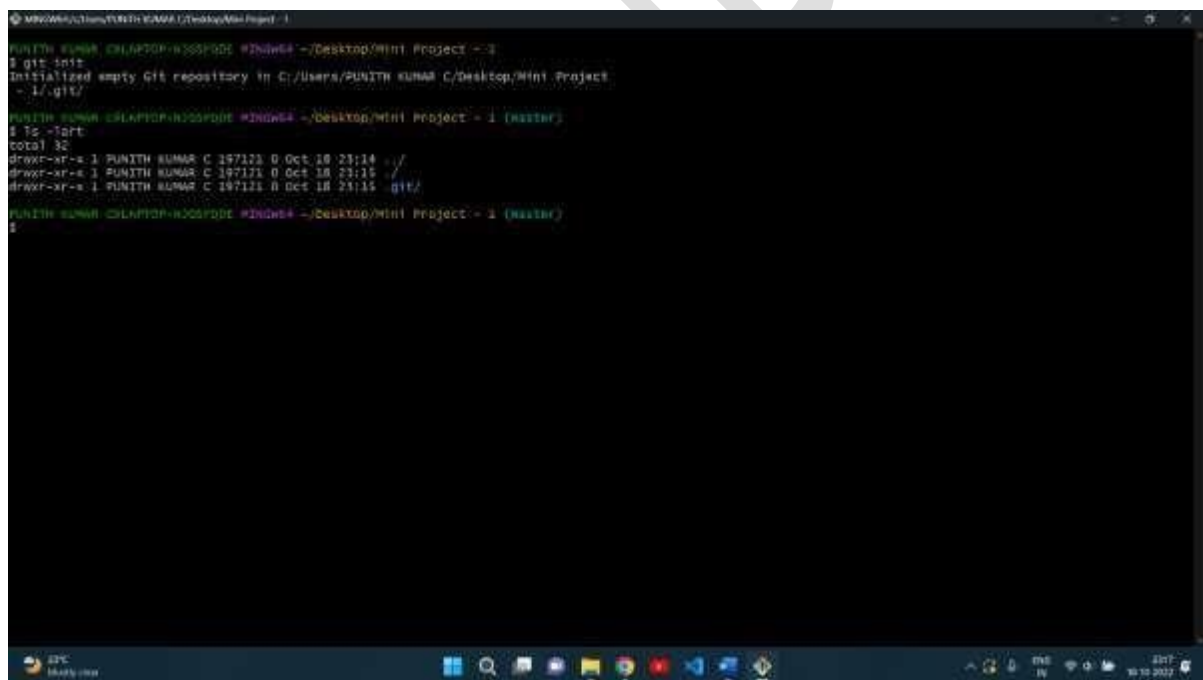
Create repository – named Mini project -1 Push the same to Git Hub:

Git in the Mini Project – 1 folder:

Open the Mini Project – 1 folder and right click and select Git Bash Here



Making the Mini Project – 1 project as git repository by “git init” command



Checking the status of the git repository:

```
MINGW64/C:/Users/PUNITH KUMAR/Desktop/Mini Project - 1
PUNITH KUMAR C:\LAPTOP-NJGSFQOE MINGW64 ~/Desktop/Mini Project - 1
$ git init
Initialized empty Git repository in C:/Users/PUNITH KUMAR/Desktop/Mini Project - 1/.git/

PUNITH KUMAR C:\LAPTOP-NJGSFQOE MINGW64 ~/Desktop/Mini Project - 1 (master)
$ ls -la
total 32
drwxr-xr-x 1 PUNITH KUMAR C 197121 0 Oct 18 23:14 ./
drwxr-xr-x 1 PUNITH KUMAR C 197121 0 Oct 18 23:15 ../
drwxr-xr-x 1 PUNITH KUMAR C 197121 0 Oct 18 23:15 .git/

PUNITH KUMAR C:\LAPTOP-NJGSFQOE MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Demo.py

nothing added to commit but untracked files present (use "git add" to track)
PUNITH KUMAR C:\LAPTOP-NJGSFQOE MINGW64 ~/Desktop/Mini Project - 1 (master)
$
```

Adding all files to staging area using “git add -A”

```
drwxr-xr-x 1 PUNITH KUMAR C 197121 0 Oct 18 23:15 .git/

PUNITH KUMAR C:\LAPTOP-NJGSFQOE MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Demo.py

nothing added to commit but untracked files present (use "git add" to track)
PUNITH KUMAR C:\LAPTOP-NJGSFQOE MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git add -A
PUNITH KUMAR C:\LAPTOP-NJGSFQOE MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   Demo.py

PUNITH KUMAR C:\LAPTOP-NJGSFQOE MINGW64 ~/Desktop/Mini Project - 1 (master)
$
```

Once the staging is done the files are ready to commit.

Using “git commit -m “message”

```
MINGW64~/Desktop/Mini Project - 1
(use "git add <file>..." to include in what will be committed)
Demo.py

nothing added to commit but untracked files present (use "git add" to track)
PUNITH KUMAR C:\LAPTOP-NJGSFQ0E\MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git add -A
PUNITH KUMAR C:\LAPTOP-NJGSFQ0E\MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   Demo.py

PUNITH KUMAR C:\LAPTOP-NJGSFQ0E\MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git commit -m "Committed Demo.py"
[master (root-commit) 1f646d6] Committed Demo.py
 1 file changed, 1 insertion(+)
 create mode 100644 Demo.py

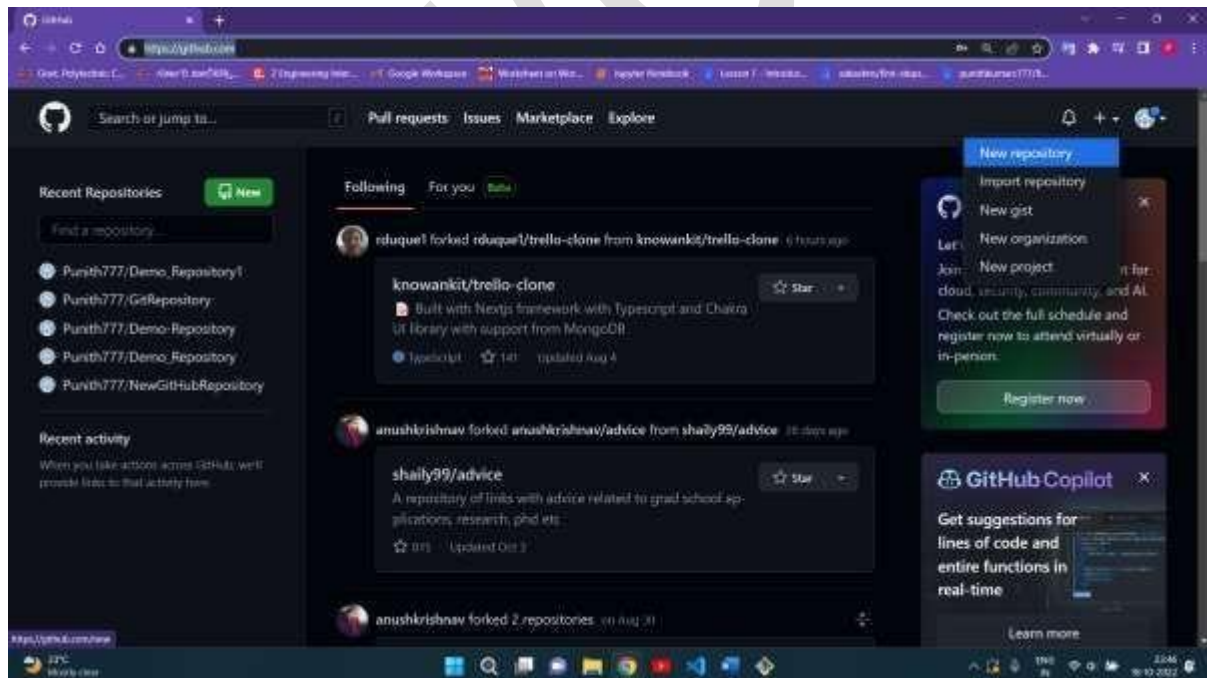
PUNITH KUMAR C:\LAPTOP-NJGSFQ0E\MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git status
On branch master
nothing to commit, working tree clean

PUNITH KUMAR C:\LAPTOP-NJGSFQ0E\MINGW64 ~/Desktop/Mini Project - 1 (master)
$ |
```

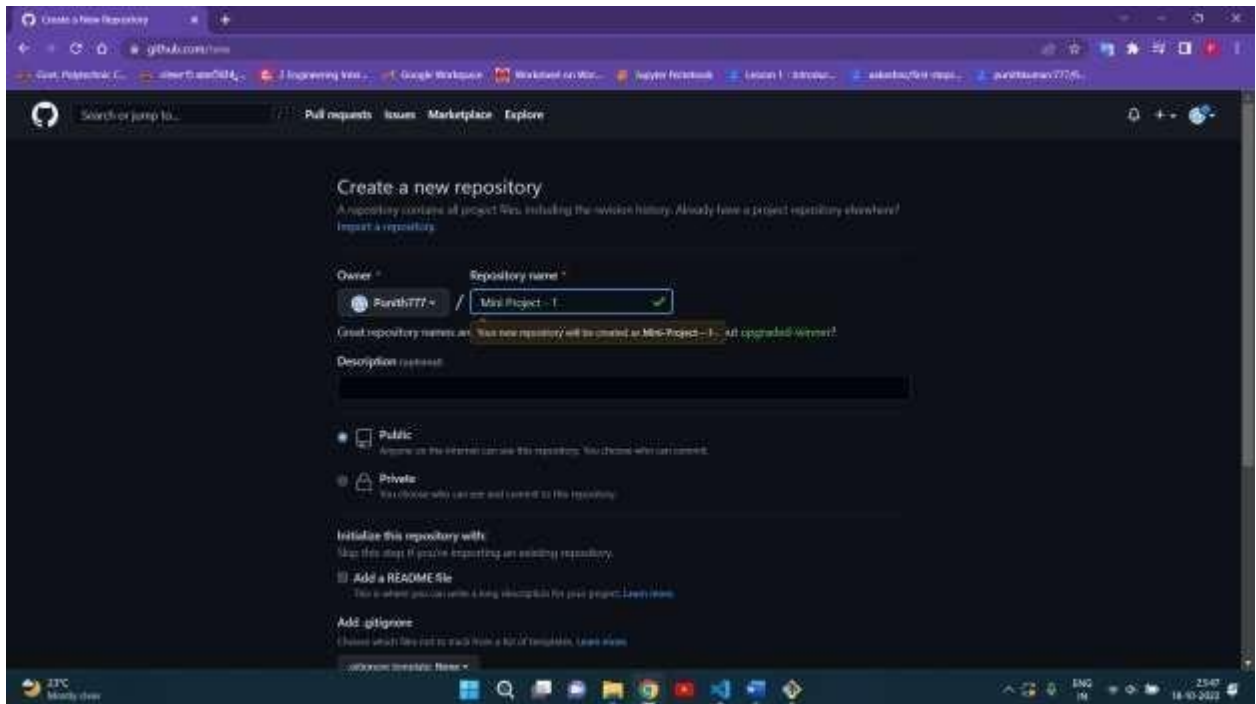
Uploading the Mini Project – 1 repository to the git hub

Step 1: Open any browser and open <https://github.com> and sign in with email and pwd

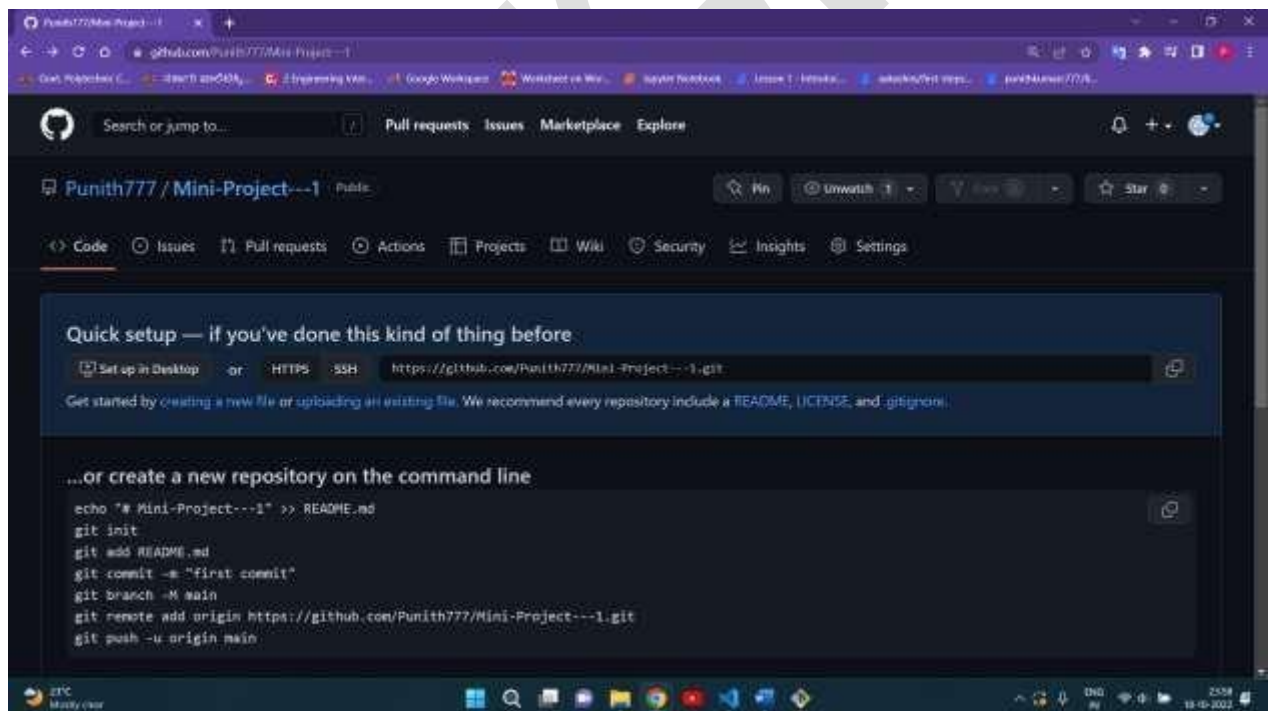
Step 2: Click on New Repository



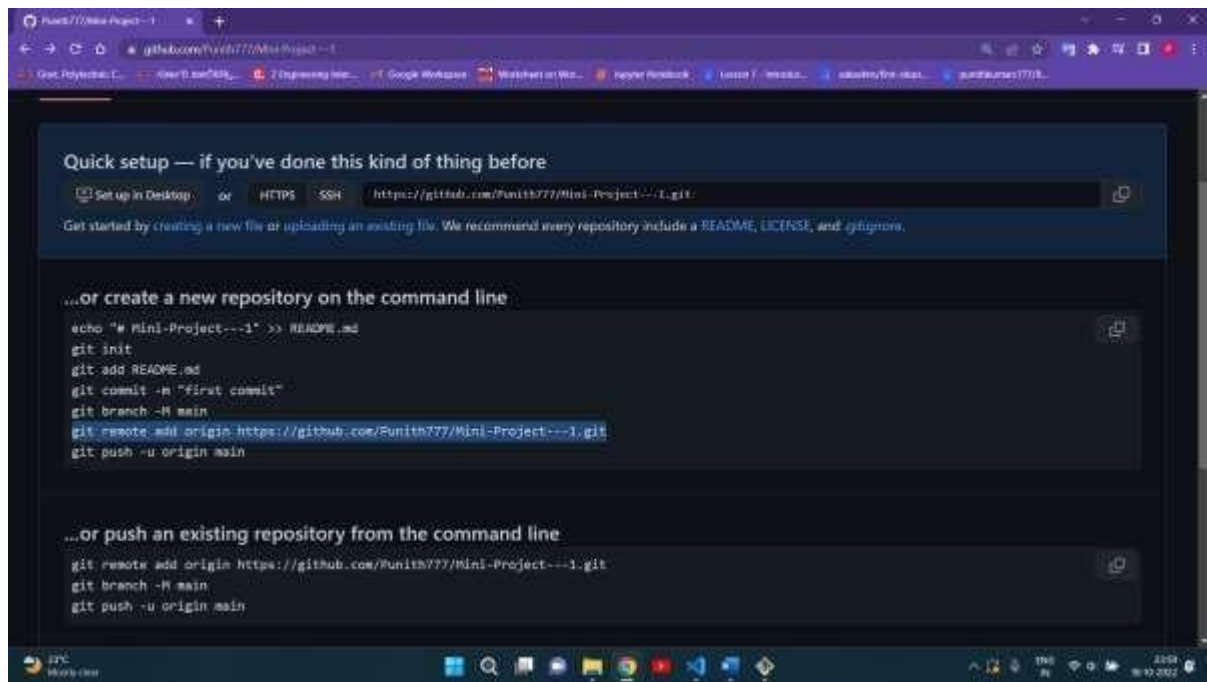
Name the repository name as “Mini Project -1”



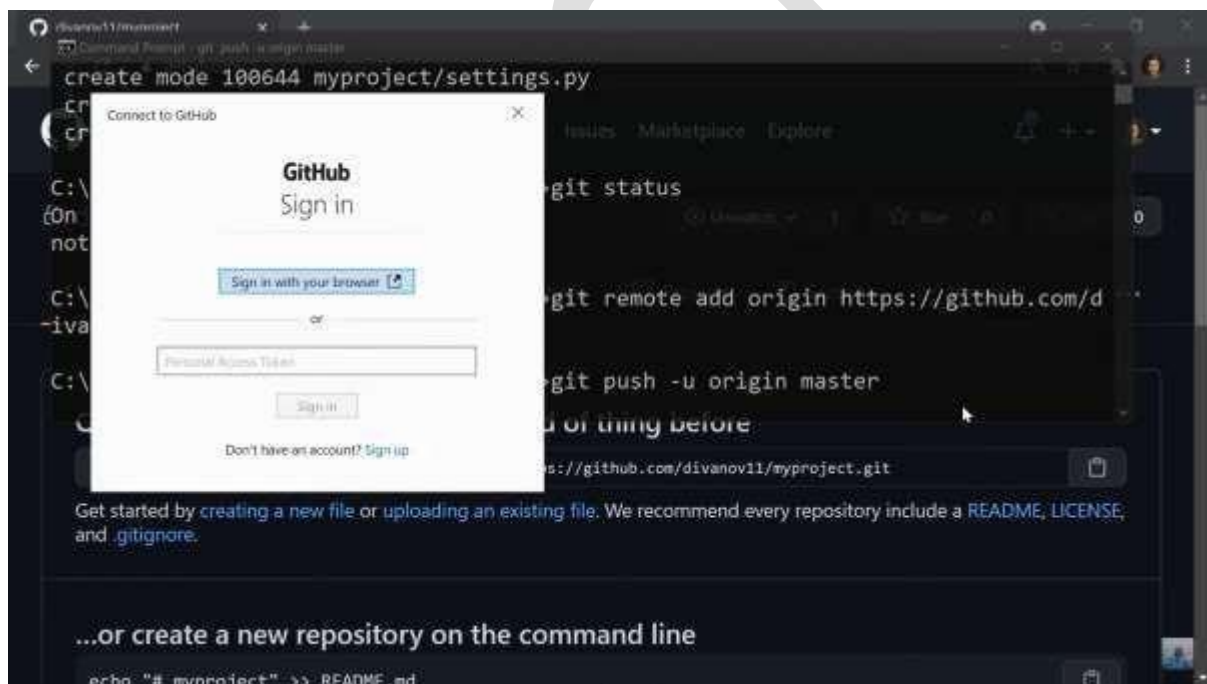
Once you click on “Create new repository”, the page will look like this



Copy the origin and paste it in git bash



Then give the “git push -u origin master” command



Then click on sign in with your browser and sign in with user name and pwd

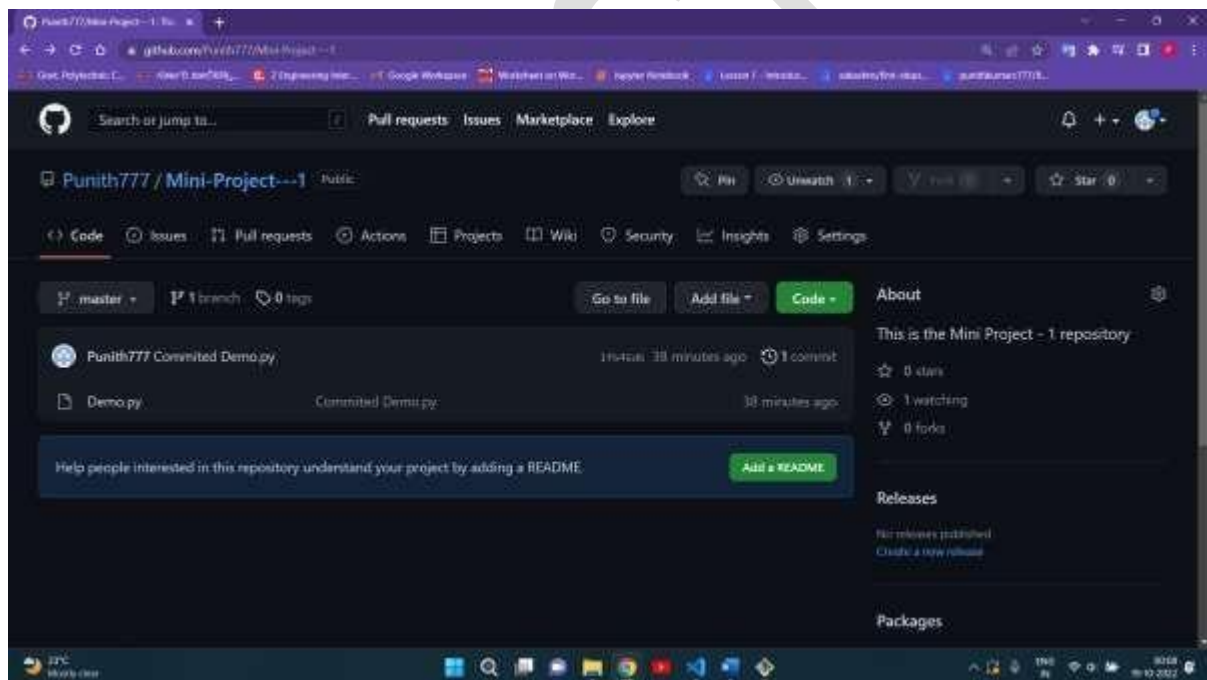
Then the files are added to the Mini Project – 1 repository in Git Hub

```
MINGW64/~/Desktop/Mini Project - 1
PUNITH KUMAR C@LAPTOP-NJGSEFQIE MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git remote add origin https://github.com/Punith777/Mini-Project---1.git

PUNITH KUMAR C@LAPTOP-NJGSEFQIE MINGW64 ~/Desktop/Mini Project - 1 (master)
$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 250 bytes | 250.00 kib/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Punith777/Mini-Project---1.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

PUNITH KUMAR C@LAPTOP-NJGSEFQIE MINGW64 ~/Desktop/Mini Project - 1 (master)
$
```

Once you refresh the same page, the file is added which is Demo.py



Explore and prepare a report on all popular AI cloud services (ML/DL) offered by Vendors.

1. Amazon Web Services (AWS):



The clear leader in cloud computing, AWS offers both consumer and business-oriented artificial intelligence (AI) products and services. Many of its professional AI services build on the AI services available in consumer products.

Its Alexa - embedded into the Amazon Echo - introduces AI to the home with its intelligent voice server. For AWS, the company's primary AI services include: Lex, for building conversational interfaces into any application; Polly, which turns text to speech; and Recognition, an image recognition service.

Amazon Machine Learning provides visualisation tools that guide customers through the process of creating machine learning (ML) models without having to learn complex ML algorithms and technology.

2. Google Cloud



Google Cloud

Leaders in AI and data analytics, Google has acquired a number of companies and startups to improve its AI capabilities for customers.

Google Cloud sells several AI and machine learning services to businesses, with an industry-leading software project in TensorFlow, as well as its own Tensor AI chip project. The Cloud Vision API can identify objects, logos, and landmarks within images, text within an image, can find similar images on the Web, or detect faces and read expressions.

The company also offers a Cloud ML service where developers can train high-quality machine learning models - such as customer service tech - using Google's existing APIs. For more experienced ML developers, Google offers ML Engine for bringing machine learning models to production, using TensorFlow models that need to be trained for various scenarios.

3. IBM Cloud



As a leader in AI for a number of years, it comes as no surprise that [IBM](#) has made this list.

With a number of cloud and AI acquisitions under its belt, [IBM](#) has a whole host of AI offerings available. In fact, under the Watson brand for AI services, IBM has no less than 16 services, and its Cloud AI services start with Watson Studio for building and training AI models, preparing data, and performing analysis on the data. IBM Watson Services for Core ML allows enterprises to build AI-powered apps that securely connect to their data and run either on-premises, offline or in the cloud.

4. Microsoft Azure



[Microsoft Azure](#) has a collection of AI solutions that can be split under three categories: AI Services, AI Tools and Frameworks, and AI Infrastructure.

AI Services is anything from pre-built capabilities - such as Azure Cognitive Services - to custom AI development with Azure Machine Learning (AML). With AI Tools and Frameworks, customers can utilise a number of Microsoft's AI services such as Azure Notebooks and Visual Studio Tools for AI. AI Infrastructure includes different services such as Azure Data Services and Azure Kubernetes Services.

5. Salesforce

With its AI platform, Einstein AI, [Salesforce](#) offers AI solutions that are fully integrated with other Salesforce cloud offerings. In doing so, the company enables its customers to build apps

using ML and predictive analytics as well as utilising their Salesforce data. With this, customers can build apps such as chatbots and sales prediction.

By providing deep sights from its customers' data, Salesforce empowers customers to use these insights to strengthen relationships, prioritise leads, cases, and campaigns to drive the business forward.

KNVP

REFERENCES

Sl. No.	Description
1.	Infosys Spring board
2.	https://www.sas.com
3.	https://databasetown.com
4.	https://www.javatpoint.com
5.	https://ziniosedge.com
6.	https://youtu.be/gwWKnnCMQ5c
7.	https://www.youtube.com/c/CodeWithHarry
8.	https://databasetown.com
9.	https://youtu.be/qMck70tLDuo

KVGP

KVGP