File Sharing System

E4 SEM2 PROJECT REPORT

Submitted by

DEVARA APSARABHANU
Roll No : R161815

SOMA CHANDANA
Roll No:R161375

In partial fulfilment of the project requirements for the degree of

BACHELOP OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Under the guidence of

Mr.N.Sathyanandaram

Lecturer,Dept.Of CSE,

RGUKT, RK Valley.

BONAFIDE CERTIFICATE

This is to certify that the project report titled “**FILE SHARING SYSTEM**’ submitted by DEVARA APSARABHANU ,SOMA CHANDANA bearing registration number R161815,R161375, respectively is a record of bonafide project work carried out under my supervision.

(Head of the Department)	(Project Guide)
Ms.CH Rathnakumari	Mr.N Satyanandaram
Asst.Professor in CSE	Lecturerin CSE
Department of CSE	Department of CSE
RGUKT, R.K Valley	RGUKT, R.K Valley.

Acknowledgement

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of our project. All that we have done is only due to such supervision and assistance and we would not forget to thank them.

We would like to express our sincere gratitude to **Mr.M Sathyanandaram** for valuable suggestions and keen interest throughout the progress of our course of research.

We are grateful to our **Ms.CH Ratna Kumari, HOD CSE**, for providing excellent computing facilities and a congenial atmosphere for progressing with our project.

At the outset, we would like to thank **Rajiv Gandhi University of Knowledge and Technologies, R.K Valley**, for providing all the necessary resources for the successful completion of our course work.

ABSTRACT

File sharing is one of the oldest applications of the internet. One way of sharing files online is for a user to upload files to a common space on the web and other users can download the files from the common web space. The objective of this project was to design an online file sharing website where industry can upload files and other industries can download them and also useful for general users. To attain this objective an AJAX enabled interactive user interface involving features like versioning control, extensive search capabilities was developed. To make the website more user friendly, users were given two space-constrained visualizations of their file system to view space occupied by the files and folders, and three AJAX based file management systems that works like browsing files on a desktop computer with drag and drop, context menu functionalities.

CONTENTS

1.INTRODUCTION

Online file sharing system is a document management system. Where a user can upload any format files such as computer programs, multimedia (audio, images and video), documents, or electronic books. etc. to the File Sharing System. The basic concept behind the system is to avoid redundant mail sending with attachments and easy to access mode.

1.1 AIM OF THE PROJECT

The aim of this project is to design and implement a file sharing application for Web based devices. This project will allow multiple users to share files to multiple devices. This project would provide a stable platform to enable collaboration through file sharing. To this end, files may be uploaded by one user and available to another by accessing through link, all simplified through an easy to use web application.Online file sharing system is a document management system. Where a user can upload any format files such as computer programs, multimedia (audio, images and video), documents, or electronic books. etc. to the File Sharing System. The basic concept behind the system is to avoid redundant mail sending with attachments and easy to access mode.

1.2 APPLICATION OF THE PROJECT

This project can be applied in several places or for several reasons. For a better understanding of its application, let’s take a look at the illustration below:

John is a lecturer at Lagos-state University. He wakes up late for a presentation he is giving in the department of computer science with his fellow lecturers. He rushes into the department, where his group is just about to go up to make a presentation. He doesn’t have any of his notes with him.

His team has updated their proposed ideas that morning, and so uploaded the notes into their shared web and sent the file link to John. John can open up the link and bring up the file on his device, so he has a set of hints to follow through for the presentation. It went well, and their Head of Department was satisfied

1.3 BENEFITS:

Using a File System Sharing offers many benefits for both the consumer and the producer.**1.**

1.Saves Time

Employees don’t have to worry about sending files via email, uploading, or downloading the files. Instead, they can quickly access the data they need when they need it.

2. Eliminates Multiple File Versions

Instead of having multiple versions of the same file with different information, all the information is current and is always the most up-to-date version of the file.

3. Saves Money

You don’t have to worry about users having sufficient storage space on their workstations to save files when you save data on a network server or cloud-based server.

4. Prevents Lost Data

In the event of a computer crash, data is not lost when it is saved on a server with nightly data backups.

5. File Permissions Limits Access

You can configure network file structures so that only those employees who need access to the data have it. Additionally, you can even control what they do with the files, such as read-only, update, save, etc.

1.4 SHORTCOMINGS:

- Lack of monitoring and security features
- Loss of full control of data
- They can get hacked, too

2.LITERATURE REVIEW

Amazon Web Services

I. INTRODUCTION

In 2006, **Amazon Web Services (AWS)** started to offer IT services to the market in the form of web services, which is nowadays known as **cloud computing**. With this cloud, we need not plan for servers and other IT infrastructure which takes up much of time in advance. Instead, these services can instantly spin up hundreds or thousands of servers in minutes and deliver results faster. We pay only for what we use with no up-front expenses and no long-term commitments, which makes AWS cost efficient.Today, AWS provides a highly reliable, scalable, low-cost infrastructure platform in the cloud that powers multitude of businesses in 190 countries around the world.

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers,storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

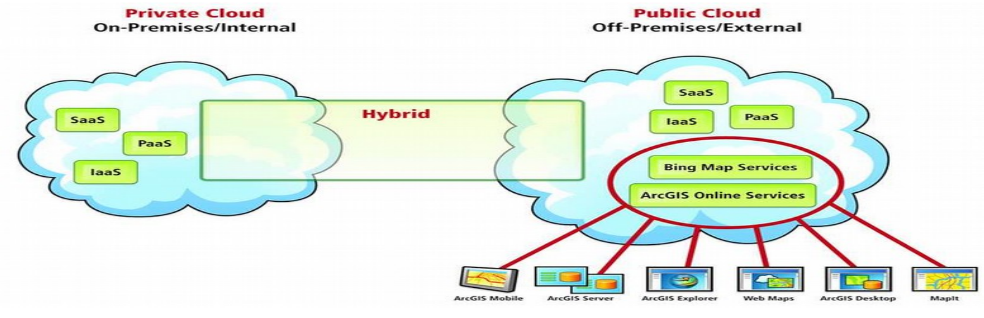
II. HISTORY OF COMPUTING

When the era of personal computers began in the 1970s, it was generally called micro-computers. It later became an affordable consumer goods when the microprocessors where developed. The workstation computers started in the 1990s. With various developments of hardware and processor components, the computers became much more easily available for cheaper costs.The applications that were running on the personal computers were platform dependent and there was no direct connectivity to share data from one toanother. Security was a major concern during this time.System administration and responsibility of data, system crash and virus infections became widespread.

In July 2002,*Amazon.com Web Services*, managed by Colin Bryar,launched its first web services opening up the Amazon.com platform to all developers.Over a hundred applications were built on top of it by 2004.This unexpected developer interest took Amazon by surprise and convinced them that developers were "hungry for more”.In late 2003, the concept for compute, which would later launch as EC2,In November 2004, the first AWSinfrastructure service launched for public usage:Simple Queue Service (SQS).

III. BASIC CLOUD COMPUTING

Cloud Deployment Models



Most cloud hubs have tens of thousands of servers and storage devices to enable fast loading. It is often possible to choose a geographic area to put the data “closer” to users. Thus, deployment models for cloud computing are categorized based on their location.

Public Cloud: Public cloud allows system and services to be easily accessible to general public. Public cloud could also be less secure attributable to its openness e.g. Email.

Benefits of Public Cloud

- **Minimal Investment**– As a pay-per-use service, there is no large upfront cost and is ideal for businesses who need quick access to resources
- **No Hardware Set-up**–The entire infrastructure is fully funded by the cloud service providers
- **No Infrastructure Management**– Does not require an in-house team to utilize the public cloud

Limitations of Public Cloud

- **Data Security and Privacy Concerns**– Since it is accessible to all, it does not fully protect against cyber-attacks and could lead to vulnerabilities
- **Reliability Issues**– Since the same server network is open to a wide range of users, it can lead to malfunction and outages
- **Service/License Limitation**– While there are many resources that you can exchange with tenants, there is a cap on usage.

Private Cloud: Private cloud allows systems and services to be accessible within an organisation.It offers accrued security attributable to its personal nature.

Benefits of Private Cloud

- **Data Privacy**– It is ideal for storing corporate data where only authorized personnel gets access
- **Security**– Segmentation of resources within the same infrastructure can help with better access and higher levels of security
- **Supports Legacy Systems**– This model supports legacy systems that cannot access the public cloud

Limitations of Private Cloud

- **Higher Cost**– With the benefits that you get, the investment will also be larger than the public cloud. Here, you will pay for software, hardware and resources for staff and training.
- **Fixed Scalability**– The hardware you choose will accordingly help you scale in a certain direction
- **High Maintenance**– Since it is managed in-house, the maintenance costs also increase

Community Cloud: Community cloud allows system and services to be accessible by group of organisation.

- **Smaller Investment**– A community cloud is much cheaper than the private & public cloud and provides great performance
- **Setup Benefits**– The protocols and configuration of a community cloud must align with industry standards. This allows customers to work much more efficiently

Limitations of Community Cloud

- **Shared Resources**– Due to restricted bandwidth and storage capacity, community resources often pose challenges
- **Not as Popular**– Since this is a recently introduced model, it is not that popular or available across industries

Hybrid Cloud: Hybrid cloud is a mixture of private and public cloud. However, the critical activities are performed using private cloud while non critical activities are performed using public cloud.

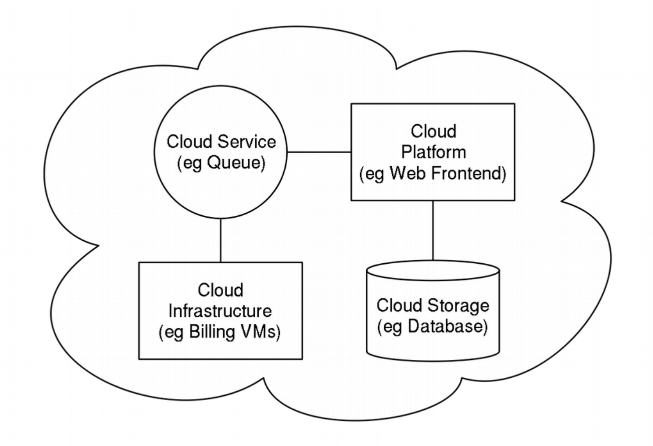
Benefits of Hybrid Cloud

- **Cost-Effectiveness**– The overall cost of a hybrid solution decreases since it majorly uses the public cloud to store data
- **Security**– Since data is properly segmented, the chances of data theft from attackers is significantly reduced
- **Flexibility**– With higher levels of flexibility, businesses can create custom solutions that fit their exact requirements

Limitations of Hybrid Cloud

- **Complexity**– It is complex setting up a hybrid cloud since it needs to integrate two or more cloud architectures
- **Specific Use Case**– This model makes more sense for organizations that have multiple use cases or need to separate critical and sensitive data

CLOUD ARCHITECTURE


Cloud Service (eg Queue)
Cloud Platform (eg Web Frontend)
Cloud Infrastructure (eg Billing VMs)
Cloud Storage (eg Database)

CLOUD SERVICE MODELS

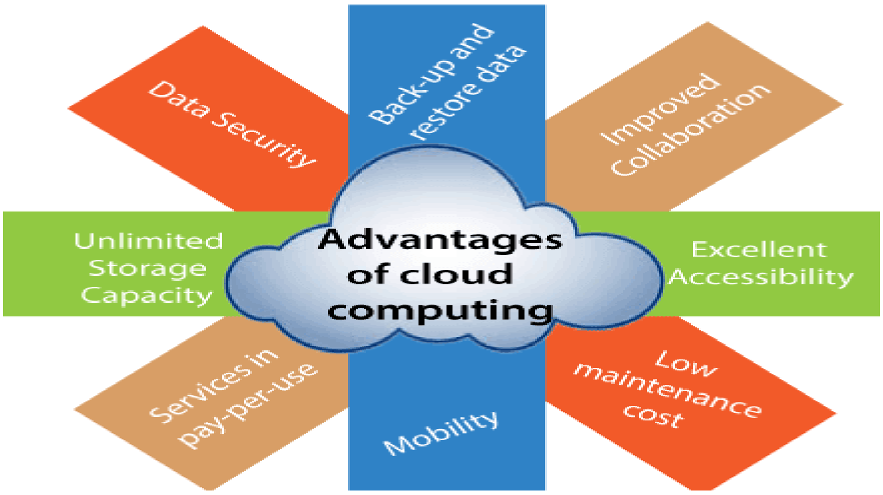
There are three types of service models in cloud − IaaS, PaaS, and SaaS.

IaaS: IaaS stands for **Infrastructure as a Service**. It provides users with the capability to provision processing, storage, and network connectivity on demand. Using this service model, the customers can develop their own applications on these resources.

PaaS: PaaS stands for **Platform as a Service**. Here, the service provider provides various services like databases, queues, workflow engines, e-mails, etc. to their customers. The customer can then use these components for building their own applications. The services, availability of resources and data backup are handled by the service provider that helps the customers to focus more on their application's functionality.

SaaS: SaaS stands for **Software as a Service**. As the name suggests, here the third-party providers provide end-user applications to their customers with some administrative capability at the application level, such as the ability to create and manage their users. Also some level of customizability is possible such as the customers can use their own corporate logos, colors, etc.

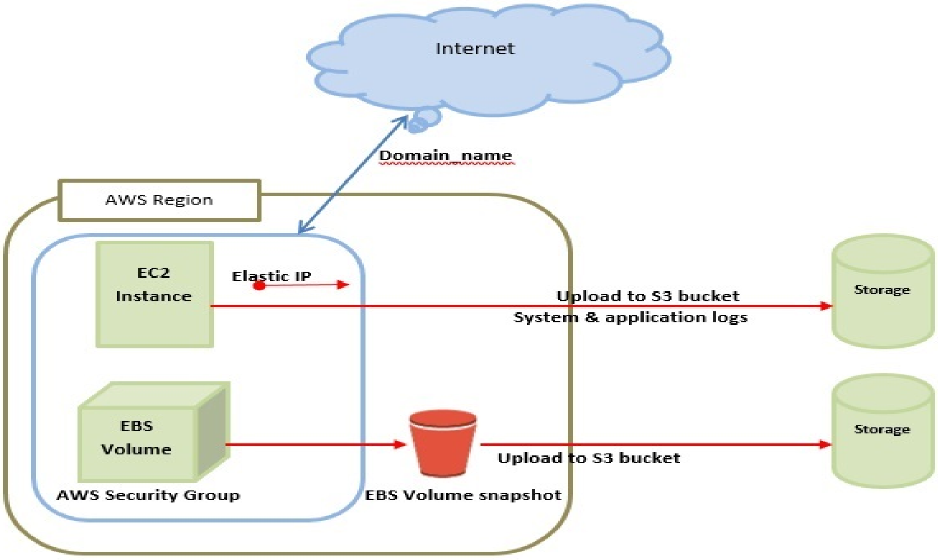
ADVANTAGES OF CLOUD COMPUTING



- **Back-up and restore data:**Once the data is stored in the cloud, it is easier to get back-up and restore that data using the cloud.
- **Improved collaboration:**Cloud applications improve collaboration by allowing groups of people to quickly and easily share information in the cloud via shared storage.
- **Excellent accessibility:**Cloud allows us to quickly and easily access store information anywhere, anytime in the whole world, using an internet connection. An internet cloud infrastructure increases organization productivity and efficiency by ensuring that our data is always accessible.
- **Low maintenance cost:**Cloud computing reduces both hardware and software maintenance costs for organizations.
- **Mobility:**Cloud computing allows us to easily access all cloud data via mobile.
- **Services in the pay-per-use model:**Cloud computing offers Application Programming Interfaces (APIs) to the users for access services on the cloud and pays the charges as per the usage of service.
- **Unlimited storage capacity:**Cloud offers us a huge amount of storing capacity for storing our important data such as documents, images, audio, video, etc. in one place.
- **Data security:**Data security is one of the biggest advantages of cloud computing. Cloud offers many advanced features related to security and ensures that data is securely stored and handled.

IV. BASIC AWS ARCHITECTURE

AWS



SECURITY

Cloud security at AWS is the highest priority. As an AWS customer, you will benefit from a data center and network architecture built to meet the requirements of the most security-sensitive organizations. Security in the cloud is much like security in your on-premises data centers—only without the costs of maintaining facilities and hardware. In the cloud, you don't have to manage physical servers or storage devices. Instead, you use software-based security tools to monitor and protect the flow of information into and out of your cloud resources.

The AWS Cloud enables a shared responsibility model. While AWS manages security **of** the cloud, you are responsible for security **in** the cloud. This means that you retain control of the security you choose to implement to protect your own content, platform, applications, systems, and networks no differently than you would in an on-site data center.

You get access to hundreds of tools and features to help you to meet your security objectives. AWS provides security-specific tools and features across network security, configuration management, access control, and data encryption.

Benefits of AWS Security

- **Keep Your Data Safe**: The AWS infrastructure puts strong safeguards in place to help protect your privacy. All data is stored in highly secure AWS data centers.
- **Meet Compliance Requirements**: AWS manages dozens of compliance programs in its infrastructure. This means that segments of your compliance have already been completed.
- **Save Money**: Cut costs by using AWS data centers. Maintain the highest standard of security without having to manage your own facility
- **Scale Quickly**: Security scales with your AWS Cloud usage. No matter the size of your business, the AWS infrastructure is designed to keep your data safe.

AWS COMPLIANCE

Compliance is nothing but “the action of complying with a command,” or “the state of meeting rules or standards”.AWS Cloud Compliance enables you to understand the robust controls in place at AWS to maintain security and data protection in the cloud. As systems are built on top of AWS Cloud infrastructure, compliance responsibilities will be shared. By tying together governance-focused, audit-friendly service features with applicable compliance or audit standards, AWS Compliance enablers build on traditional programs. This helps customers to establish and operate in an AWS security control environment.

AWS - Management Console::

- ➔ AWS Management Console is a web application for managing Amazon Web Services. AWS Management Console consists of list of various services to choose from. It also provides all information related to our account like billing.This console provides an inbce to perform AWS tasks like working with Amazon S3 buckets, launching and connecting to Amazon EC2 instances, setting Amazon CloudWatch alarms, etc.

AWS – Console Mobile App::

- ➔ The AWS Console mobile app, provided by Amazon Web Services, allows its users to view resources for select services and also supports a limited set of management functions for select resource types.
- ➔ ***Features of AWS Mobile App*** : To have access to the AWS Mobile App, we must have an existing AWS account. Simply create an identity using the account credentials and select the region in the menu. This app allows us to stay signed in to multiple identities at the same time.For security reasons, it is recommended to secure the device with a passcode and to use an IAM user's credentials to log in to the app. In case the device is lost, then the IAM user can be deactivated to prevent unauthorized access.Root accounts cannot be deactivated via mobile console. While using AWS Multi-Factor Authentication (MFA), it is recommended to use either a hardware MFA device or a virtual MFA on a separate mobile device for account security reasons.The latest version is 1.14. There is a feedback link in the App's menu to share our experiences and for any queries.
- ➔ Following are the various services and supported functions that can be accessed using the mobile app.

- **EC2 (Elastic Compute Cloud)**

  •Browse, filter and search instances.
  •View configuration details.
  •Check status of CloudWatch metrics and alarms.
  •Perform operations over instances like start, stop, reboot, termination.
  •Manage security group rules.
  •Manage Elastic IP Addresses.
  •View block devices.

- **Elastic Load Balancing**
  - Browse, filter and search load balancers.
  - View configuration details of attached instances.
  - Add and remove instances from load balancers.
- **S3**
  - Browse buckets and view their properties.
  - View properties of objects.

- **Route 53**

    - Browse and view hosted zones.
    - Browse and view details of record sets.
- **RDS (Relational Database Service)**
    - Browse, filter, search and reboot instances.
    - View configuration details, security and network settings.
- **Auto Scaling**
    - View group details, policies, metrics and alarms.
    - Manage the number of instances as per the situation.
- **Elastic Beanstalk**
    - View applications and events.
    - View environment configuration and swap environment CNAMEs.
    - Restart app servers.
- **DynamoDB**
    - View tables and their details like metrics, index, alarms, etc.
- **CloudFormation**
    - View stack status, tags, parameters, output, events, and resources.
- **OpsWorks**
    - View configuration details of stack, layers, instances and applications.
    - View instances, its logs, and reboot them.
- **CloudWatch**
    - View CloudWatch graphs of resources.
    - List CloudWatch alarms by status and time.
    - Action configurations for alarms.
- **Services Dashboard**
    - Provides information of available services and their status.
    - All information related to the billing of the user.
    - Switch the users to see the resources in multiple accounts.

V. AMAZON FEATURED SERVICES

Amazon Computer Services

1. Amazon EC2 (Elastic Compute Cloud):

It is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers.These are just the virtual machines in the cloud on which you have the OS level control. You can run whatever you want in them.Amazon EC2 offers the broadest and deepest compute platform with choice of processor, storage, networking, operating system, and purchase model.It offer the fastest processors in the cloud and it is the only cloud with 400 Gbps ethernet networking. EC2 has the most powerful GPU instances for machine learning training and graphics workloads, as well as the lowest cost-per-inference instances in the cloud. More SAP, HPC, Machine Learning, and Windows workloads run on AWS than any other cloud.

What is an Instance?

An instance is a virtual server for running applications on Amazon's EC2. It can also be understood like a tiny part of a larger computer, a tiny part which has its own Hard drive, network connection, OS etc. But it is actually all virtual. You can have multiple "tiny" computers on a single physical machine, and all these tiny machines are called Instances.

Why AWS EC2 ?

Suppose you are a developer, and since you want to work independently you buy some servers, you estimated the correct capacity, and the computing power is enough. Now, you have to look after the updating of security patches every day, you have to troubleshoot any problem which might occur at a back end level in the servers and so on.

But if you buy an EC2 instance, you don't have to worry about any of these things as it will all be managed by Amazon; you just have to focus on your application.

Types of EC2 Computing Instances

Computing is a very broad term, the nature of your task decides what kind of computing you need. Therefore,AWS EC2 offers 5 types of instances which are as follows:

1. General Instances

For applications that require a balance of performance and cost.

E.g email responding systems, where you need a prompt response as well as the it should be cost effective, since it doesn't require much processing.

2. Compute Instances

For applications that require a lot of processing from the CPU.

E.g analysis of data from a stream of data, like Twitter stream

3. Memory Instances

For applications that are heavy in nature, therefore, require a lot of RAM.

E.g when your system needs a lot of applications running in the background i.e multitasking.

4. Storage Instances

For applications that are huge in size or have a data set that occupies a lot ofspace.

E.g When your application is of huge size.

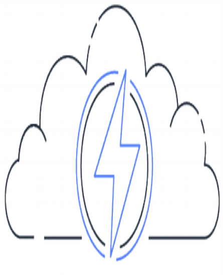
5. GPU Instances

For applications that require some heavy graphics rendering.

E.g 3D modeling etc.

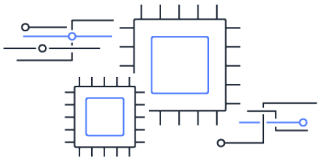
Building Blocks

Faster innovation and increased security with AWS Nitro System



The AWS Nitro System is the underlying platform for our next generation of EC2 instances that offloads many of the traditional virtualization functions to dedicated hardware and software to deliver high performance, high availability, and high security while also reducing virtualization overhead. It giving us the flexibility to design and rapidly deliver new EC2 instance types with an ever-broadening selection of compute, storage, memory, and networking options.

Choice of Processors

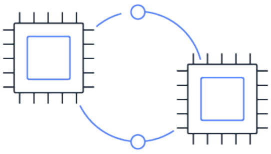


A choice of latest generation Intel Xeon, AMD EPYC, and AWS Graviton CPUs enables you to find the best balance of performance and price for your workloads.

High Performance Storage

Amazon Elastic Block Store (EBS) provides easy to use, high performance block storage for use with Amazon EC2.

Enhanced networking



Enhanced networking enables you to get significantly higher packet per second (PPS), lower network jitter, and lower latency. For high performance computing (HPC) applications, Elastic Fabric Adapter is a network interface for Amazon EC2 instances that offers low-latency, high-bandwidth interconnect between compute nodes to help scale applications to thousands of cores.

Choice of purchasing model

Amazon EC2 offer a choice of multiple purchasing models with On-Demand, Spot Instances, and Savings Plan.

2.Amazon Lightsail

Lightsail is an easy-to-use virtual private server (VPS) that offers you everything needed to build an application or website, plus a cost-effective, monthly plan.

If you don’t have any prior experience with AWS this is for you. It automatically deploys and manages compute, storage and networking capabilities required to run your applications.

Simple web applications

With pre-configured development stacks like LAMP, Nginx, MEAN, and Node.js., we make it easy to get your web application online.

Websites
Quickly create website,we can create and customize our blog,website with lightsails pre-configured applications.

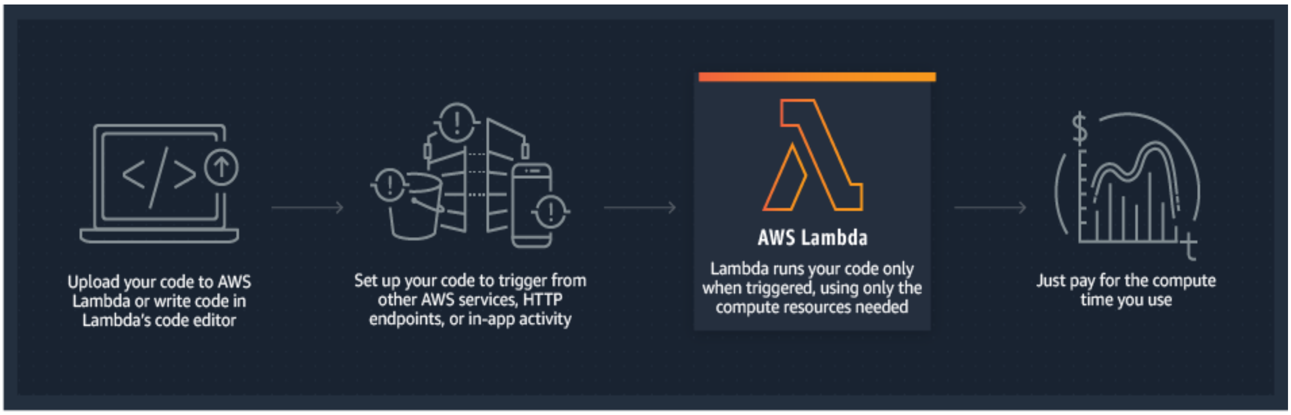
Business Software
Lightsail helps you quickly launch your line-of-business software, like file storage and sharing, backups, financial and accounting software, and much more.

Dev/Test Environment
Spin up a developer or test environment in seconds.

3.AWS Lambda

Run code without thinking about servers or clusters. Only pay for what you use. With Lambda, you can run code for virtually any type of application or backend service - all with zero administration. Just upload your code as a ZIP file or container image, and Lambda automatically and precisely allocates compute execution power and runs your code based on the incoming request or event, for any scale of traffic. You can set up your code to automatically trigger from over 200 AWS services and SaaS applications or call it directly from any web or mobile app.

How it Works


Upload your code to AWS Lambda or write code in Lambda's code editor
Set up your code to trigger from other AWS services, HTTP endpoints, or in-app activity
AWS Lambda
Lambda runs your code only when triggered, using only the compute resources needed
Just pay for the compute time you use

Benefits

- No servers to manage
- Continuous scaling
- Cost optimized with millisecond metering
- Consistent performance at any scale

Amazon Networking and content delivery

1.Amazon Virtual Private Cloud

It is simply a data center in the cloud in which you deploy all your resources. It allows you to better isolate your resources and secure them.You have complete control over your virtual networking environment, including selection of your own IP address range, creation of subnets, and configuration of route tables and network gateways. You can use both IPv4 and IPv6 for most resources in your virtual private cloud, helping to ensure secure and easy access to resources and applications.

For **example**, you can create a public-facing subnet for your web servers that has access to the Internet, and place your backend systems, such as databases or application servers, in a private-facing subnet with no Internet access. You can leverage multiple layers of security (including security groups and network access control lists) to help control access to EC2 instances in each subnet.

2.Amazon CloudFront

Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, high transfer speeds, all within a developer-friendly environment.

It is AWS’s Content Delivery Network (CDN) that consists of Edge locations that cache resources.

3.Amazon Route 53

A reliable and cost-effective way to route end users to Internet applications.

It is AWS’s highly available DNS (Domain Name System) service. You can register domain names through it.Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. It is designed to give developers and businesses an extremely reliable and cost effective way to route end users to Internet applications by translating names like www.example.com into the numeric IP addresses like 192.0.2.1 that computers use to connect to each other. Amazon Route 53 is fully compliant with IPv6 as well.

Benefits

- Highly available and reliable
- Flexible
- Designed for use with Amazon Web Services
- Simple
- Fast
- Cost-effective
- Secure
- Scalable
- Simplify the Hybrid cloud

4.Amazon Direct Connect

Using it you can connect your data center to an Availability zone using a high speed dedicated line.AWS Direct Connect permits to create a private network connection from our network to AWS location. It uses 802.1q VLANs, which can be partitioned into multiple virtual interfaces to access public resources using the same connection. This results in reduced network cost and increased bandwidth. Virtual interfaces can be reconfigured at any time as per the requirement.

5.API Gateway

Allows you to create, store and manage APIs at scale.

AWS Storage

AWS offers a complete range of services for you to store, access, govern, and analyze your data to reduce costs, increase agility, and accelerate innovation. Select from object storage, file storage, and block storage services, backup, and data migration options to build the foundation of your cloud IT environment.

1.S3(Simple Storage Service)

Storage service of AWS in which we can store objects like files, folders, images, documents, songs, etc. It cannot be used to install software, games or Operating System.

S3 (Simple Storage Service) is a scalable, high-speed, low-cost web-based service designed for online backup and archiving of data and application programs. It allows to upload, store, and download any type of files up to 5 TB in size. This service allows the subscribers to access the same systems that Amazon uses to run its own web sites. The subscriber has control over the accessibility of data, i.e. privately/publicly accessible.

We can add your own code to process data retrieved from S3 before returning it to an application

2.Amazon Elastic Block Store

Amazon Elastic Block Store (EBS) is an easy to use, high-performance, block-storage service designed for use with Amazon Elastic Compute Cloud (EC2) for both throughput and transaction intensive workloads at any scale. A broad range of workloads, such as relational and non-relational databases, enterprise applications, containerized applications, big data analytics engines, file systems, and media workflows are widely deployed on Amazon EBS.

AWS Database

1.AMAZON RDS

Allows you to run relational databases like MySQL, MariaDB, PostgreSQL, Oracle or SQL Server. These databases are fully managed by AWS like installing antivirus and patches.It is a fully-managed SQL database cloud service that allows to create and operate relational databases. Using RDS you can access your files and database anywhere in a cost-effective and highly scalable way.

Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups. It frees you to focus on your applications so you can give them the fast performance, high availability, security and compatibility they need.

2.Amazon DynamoDB

It is a fully managed NoSQL database service that allows to create database tables that can store and retrieve any amount of data. It automatically manages the data traffic of tables over multiple servers and maintains performance. It also relieves the customers from the burden of operating and scaling a distributed database. Hence, hardware provisioning, setup, configuration, replication, software patching, cluster scaling, etc. is managed by Amazon.

It is a highly scalable, high performance NoSQL database. It provides single-digit millisecond latency at any scale.

GitHub

I.INTRODUCTION

What is GitHub

GitHub is a Git repository hosting service.It is a web-based service.It is a file or code-sharing service to collaborate with different people. GitHub is a highly used software that is typically used for version control. It is helpful when more than just one person is working on a project.

Example a software developer team wants to build a website and everyone has to update their codes simultaneously while working on the project. In this case, Github helps them to build a centralized repository where everyone can upload, edit, and manage the code files.

Why is Github so popular?

GitHub has various advantages but many people often have a doubt as to why not use dropbox or any cloud based system? Let, Say more than two software developers are working on the same file and they want to update it simultaneously. Unfortunately, the

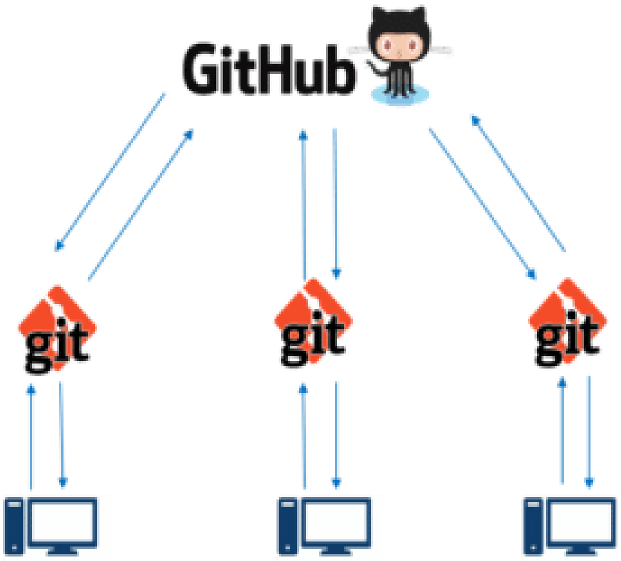
person who save the file first will get precedence over the others.While in Github, this is not the case. Github document the changes and reflect them in an organized manner to avoid any chaos between any of the files uploaded.
Therefore using GitHub centralized repository, it avoids all the confusion and working on the same code becomes very easy.

What is Git

Git is an open-source distributed version control system. It is designed to handle minor to major projects with high speed and efficiency. It is developed to co-ordinate the work among the developers. The version control allows us to track and work together with our team members at the same workspace.
Git is foundation of many services like GitHub and GitLab, but we can use Git without using any other Git services. Git can be used privately and publicly.

Git was created byLinus Torvalds in 2005 to develop Linux Kernel. It is also used as an important distributed version-control tool for the DevOps.

Git Vs GitHub



GitHub is a central repository and Git is a tool which allows you to create a local repository. Now people usually get confused between git and GitHub but its actually very different. Git is a version control tool that will allow you to perform all kinds of operations to fetch data from the central server or push data to it whereas GitHub is a core hosting platform for version control collaboration. GitHub is a company that allows you to host a central repository in a remote server.

Git	GitHub
Git is a distributed version control tool that can manage a programmer's source code history.	GitHub is a cloud-based tool developed around the Git tool.
A developer installs Git tool locally.	GitHub is an online service to store code and push from the computer running the Git tool.
Git focused on version control and code sharing.	GitHub focused on centralized source code hosting.
It is a command-line tool.	It is administered through the web.
It facilitates with a desktop interface called Git Gui.	It also facilitates with a desktop interface called GitHub Gui.
Git does not provide any user management feature.	GitHub has a built-in user management feature.
It has minimal tool configuration feature.	It has a market place for tool configuration.

Features of GitHub

GitHub is a place where programmers and designers work together. They collaborate, contribute, and fix bugs together. It hosts plenty of open source projects and codes of various programming languages.

Some of its significant features are as follows.

- Collaboration
- Integrated issue and bug tracking
- Graphical representation of branches
- Git repositories hosting
- Project management
- Team management
- Code hosting
- Track and assign tasks
- Conversations
- Wikisc

Benefits of GitHub

GitHub can be separated as the Git and the Hub. GitHub service includes access controls as well as collaboration features like task management, repository hosting, and team management.

The key benefits of GitHub are as follows.

- ➢ It is easy to contribute to open source projects via GitHub.
- ➢ It helps to create an excellent document.
- ➢ You can attract recruiter by showing off your work. If you have a profile on GitHub, you will have a higher chance of being recruited.
- ➢ It allows your work to get out there in front of the public.
- ➢ You can track changes in your code across versions.

➔GitHub provides you a beautiful visual interface which helps you to track or manage your version controlled projects locally.
➔Once you register on GitHub, you can connect with social network and build a strong profile.

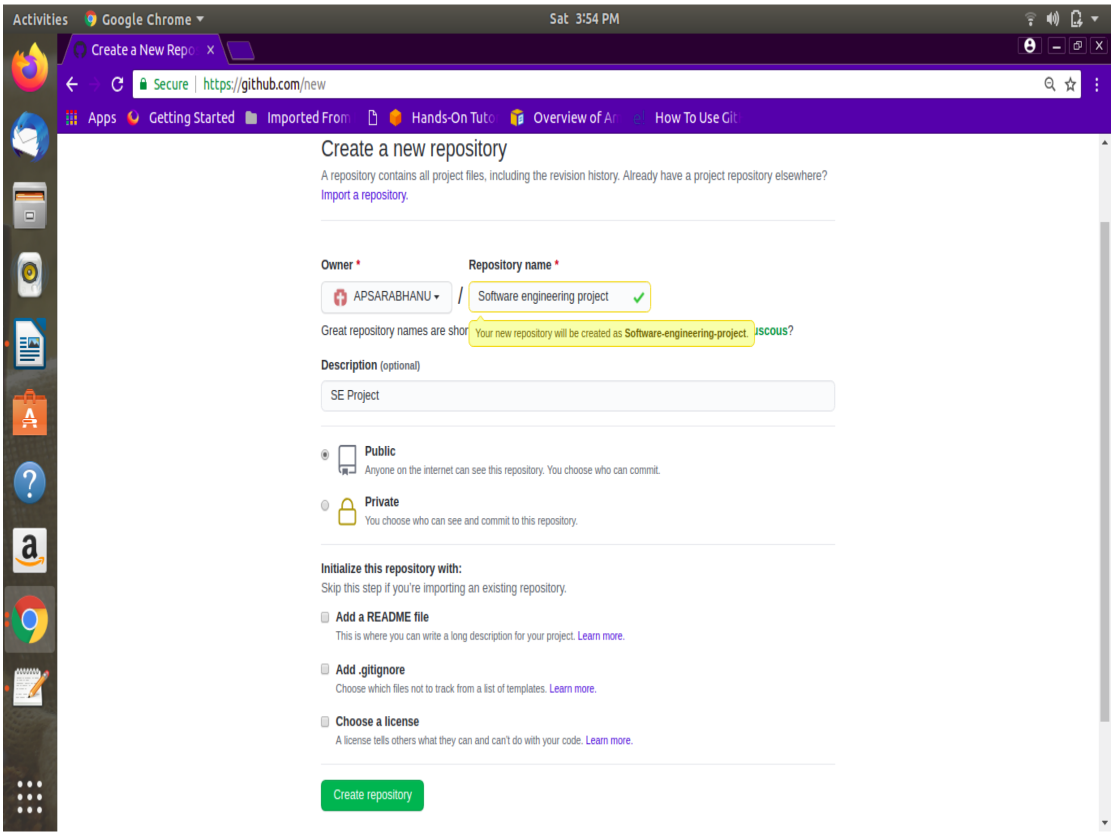
GitHub Repository

A repository is a storage space where your project lives. It can be local to a folder on your computer, or it can be a storage space on GitHub or another online host. You can keep code files, text files, images or any kind of a file in a repository. You need a GitHub repository when you have done some changes and are ready to be uploaded. This GitHub repository acts as your remote repository.

By default a GitHub repository is public which means that anyone can view the contents of this repository whereas in a private repository, you can choose who can view the content. Also, private repository is a paid version.Initialize the repository with a README file. This file contains the description of the file and once you check the box, that will be the first file inside your repository.

Create GitHub account and GitHub Repository

- Go to the link: **https://github.com/** .Fill the sign up form and click on “Sign up for Github”.
- Click on “Start a new project”.
- Click on “Create repository”.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * APSARABHANU / Repository name * Software engineering project ✓

Great repository names are shor Your new repository will be created as **Software-engineering-project**. uscous?

Description (optional)

SE Project

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

□ Add a README file
This is where you can write a long description for your project. Learn more.

□ Add .gitignore
Choose which files not to track from a list of templates. Learn more.

□ Choose a license
A license tells others what they can and can't do with your code. Learn more.

central repository has been sucessfully created! Once this is done, you are ready to commit, pull, push and perform all the other operations.

Create a Branch

Branching is the way to work on different versions of a repository at one time. By default your repository has one branch named main which is considered to be the definitive branch. We use branches to experiment and make edits before committing them to main.

When you create a branch off the main branch, you’re making a copy, or snapshot, of main as it was at that point in time. If someone else made changes to the main branch while you were working on your branch, you could pull in those updates.

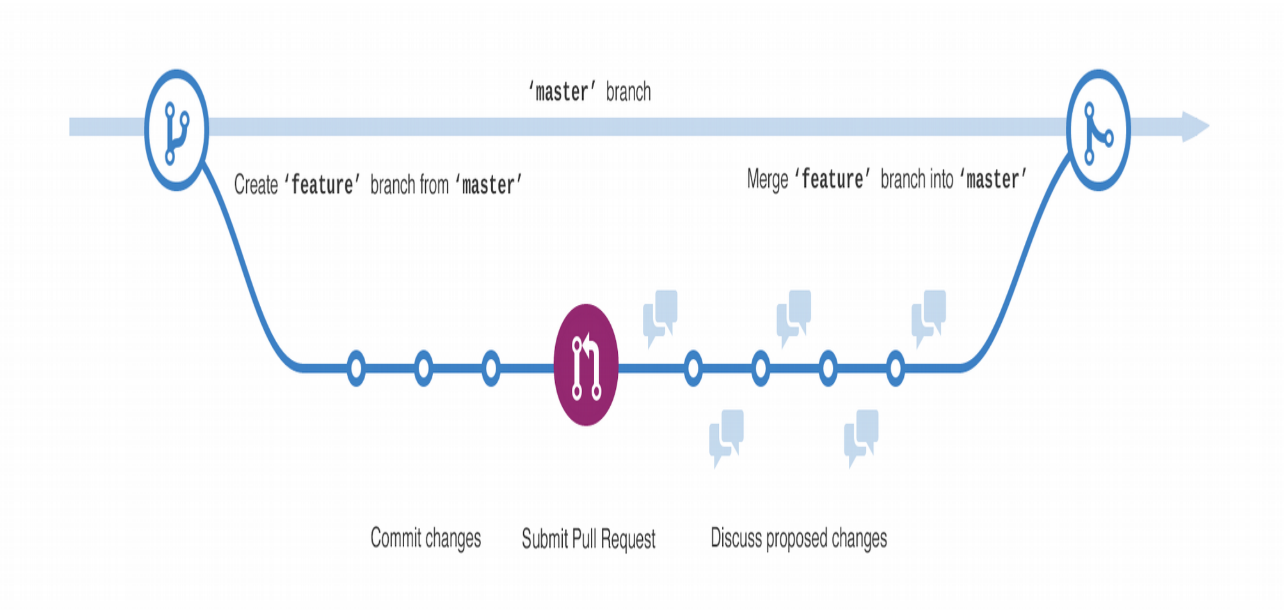
Let’s say you want to add a new feature (which is in the development phase), and you are afraid at the same time whether to make changes to your main project or not. This is where git branching comes to rescue. Branches allow you to move back and forth between the different states/versions of a project. In the above scenario, you can create a new branch and test the new feature without affecting the main branch. Once you are done with it, you can merge the changes from new branch to the main branch.

This diagram shows:

•The main branch

•A new branch called feature (because we’re doing ‘feature work’ on this branch)

•The journey that feature takes before it’s merged into main.


‘master’ branch
Create ‘feature’ branch from ‘master’
Merge ‘feature’ branch into ‘master’
Commit changes
Submit Pull Request
Discuss proposed changes

To create a new branch

- Go to your new repository Software engineering project.
- Click the drop down at the top of the file list that says branch: main.
- Type a branch name,readme-edits, into the new branch text box.
- Select the blue Create branch box or hit “Enter” on your keyboard.

Now you have two branches, main and readme-edits. They look exactly the same, but not for long!

GitHub: Operations

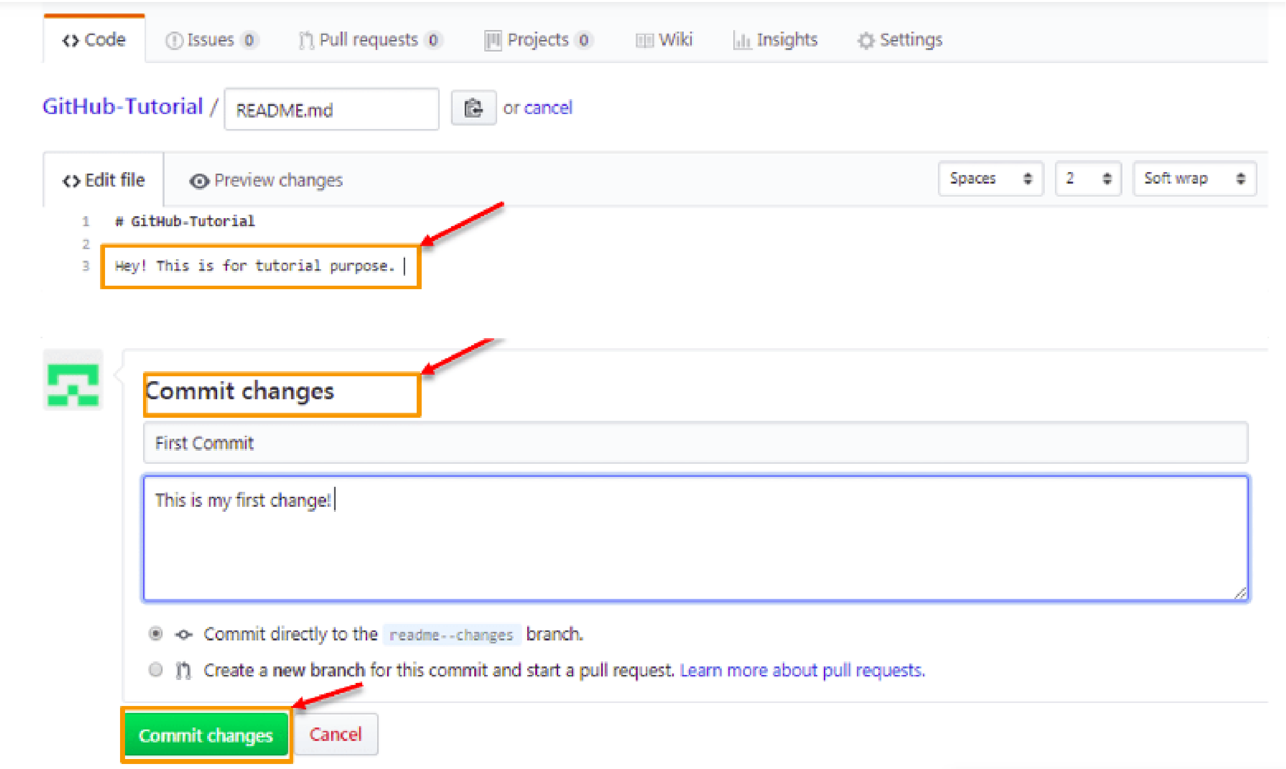
Commit Command:

On GitHub, saved changes are called commits. Each commit has an associated commit message, which is a description explaining why a particular change was made. Commit messages capture the history of your changes, so other contributors can understand what you’ve done and why.

When you commit a file, you should always provide the message, just to keep in the mind the changes done by you. Though this message is not compulsory but it is always recommended so that it can differentiate the various versions or commits you have done so far to your repository. These commit messages maintain the history of changes which in turn help other contributors to understand the file better.

Make and Commit changes

1. Click the README.md file.
2. Click the pencil icon in the upper right corner of the file view to edit.
3. In the editor, write a bit about yourself.
4. Write a commit message that describes your changes.
5. Click Commit changes button.



GitHub-Tutorial / README.md

```
# GitHub-Tutorial

Hey! This is for tutorial purpose.
```

Commit changes

First Commit

This is my first change!

● Commit directly to the `readme--changes` branch.

○ Create a new branch for this commit and start a pull request. Learn more about pull requests.

These changes will be made to just the README file on your readme-edits branch, so now this branch contains content that’s different from main.

Pull Command:

Pull Requests are the heart of collaboration on GitHub. When you open a pull request, you’re proposing your changes and requesting that someone review and pull in your contribution and merge them into their branch. Pull requests show differences, of the content from both branches. The changes, additions, and subtractions are shown in green and red.

As soon as you make a commit, you can open a pull request and start a discussion, even before the code is finished.

Pull command is the most important command in GitHub. It tell the changes done in the file and request other contributors to view it as well as merge it with the master branch. Once the commit is done, anyone can pull the file and can start a discussion over it. Once its all done, you can merge the file. Pull command compares the changes which are done in the file and if there are any conflicts, you can manually resolve it.

STEPS:

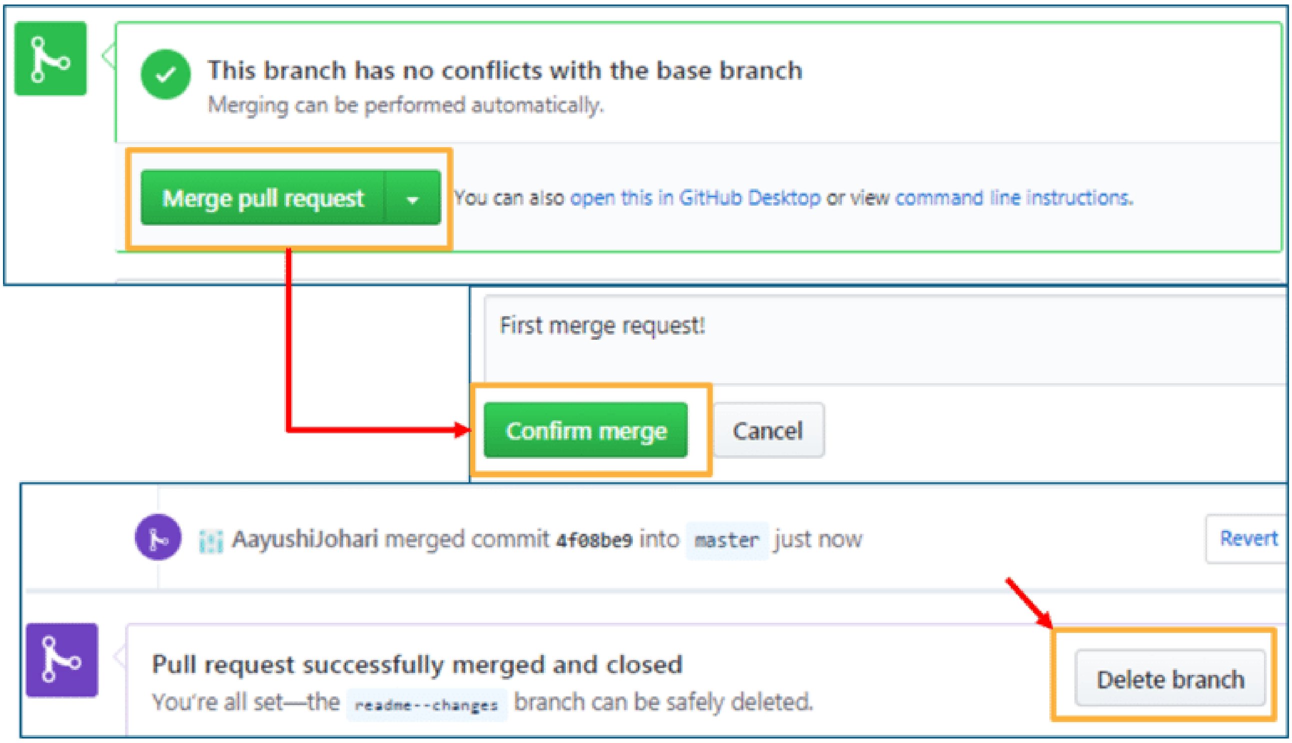
1. Click the ‘Pull requests’ tab.
2. Click ‘New pull request’.
3. Once you click on pull request, select the branch and click ‘readme- changes’ file to view changes between the two files present in our repository.
4. Click “Create pull request”.
5. Enter any title, description to your changes and click on “Create pull request”.

Merge Command:

The last command which merge the changes into the main master branch.

STEPS:

1.Click on “Merge pull request” to merge the changes into master branch.
2.Click “Confirm merge”.
3.You can delete the branch once all the changes have been incorporated and if there are no conflicts.



This branch has no conflicts with the base branch

Merging can be performed automatically.

You can also open this in GitHub Desktop or view command line instructions.

First merge request!

AayushiJohari merged commit 4f08be9 into master just now

Pull request successfully merged and closed

You're all set—the readme--changes branch can be safely deleted.

CLONING REPOSITORY

why do we need to clone a repository?

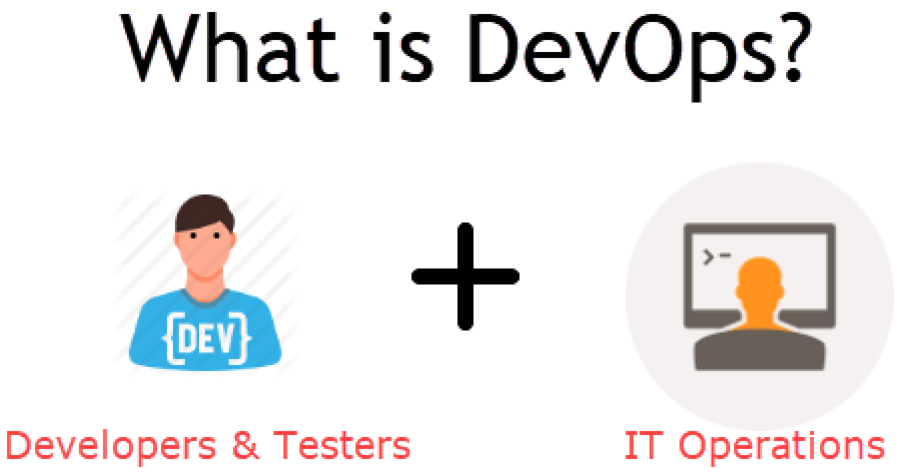
Suppose you want to use some code which is present in a public repository, you can directly copy the contents by cloning or downloading.

DevOps

DevOps is a set of practices that combines software development and IT operations. It aims to shorten the systems development life cycle and provide continuous delivery with high software quality. DevOps is complementary with Agile software development; several DevOps aspects came from the Agile methodology.

I. What is DevOps

DevOps is a collaboration between Development and IT Operations to make software production and deployment in an automated & repeatable way. DevOps helps to increase the organization's speed to deliver software applications and services. The word 'DevOps' is a combination of two words, 'Development' and 'Operations.'



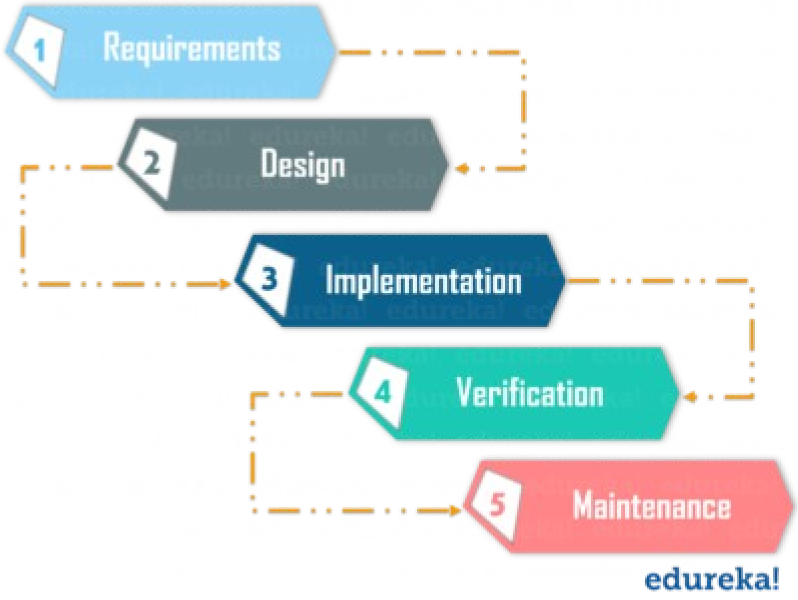
- DevOps is a practice that allows a single team to manage the entire application development life cycle, that is, development, testing, deployment, and monitoring.
- The ultimate goal of DevOps is to decrease the duration of the system’s development life cycle while delivering features, fixes, and updates frequently in close synchronization with business objectives.
- DevOps is a software development approach with the help of which you can develop superior quality software quickly and with more reliability. It consists of various stages such as continuous development, continuous integration, continuous testing, continuous deployment, and continuous monitoring.

II. History of DevOps

Waterfall model

The waterfall model is a software development model that is pretty straight forward and linear. This model follows a top-down approach.

•This model has various starting with **Requirements gathering and analysis**. This is the phase where you get the requirements from the client for developing an application. After this, you try to analyze these requirements.



•The next phase is the **Design** phase where you prepare a blueprint of the software.
•Once the design is ready, you move further with the **Implementation** phase where you begin with the coding for the application. The team of developers works together on various components of the application.
•Once you complete the application development, you test it in the **Verification** phase. There are various tests conducted on the application such as unit testing, integration testing, performance testing, etc.
•After all the tests on the application are completed, it is deployed onto the production servers.
•At last, comes the **Maintenance** phase. In this phase, the application is monitored for performance. Any issues related to the performance of the application are resolved in this phase.

Advantages of the Waterfall Model:

•Simple to understand and use
•Allows for easy testing and analysis
•Saves a significant amount of time and money
•Good for small projects if all requirements are clearly defined
•Allows for departmentalization & managerial control

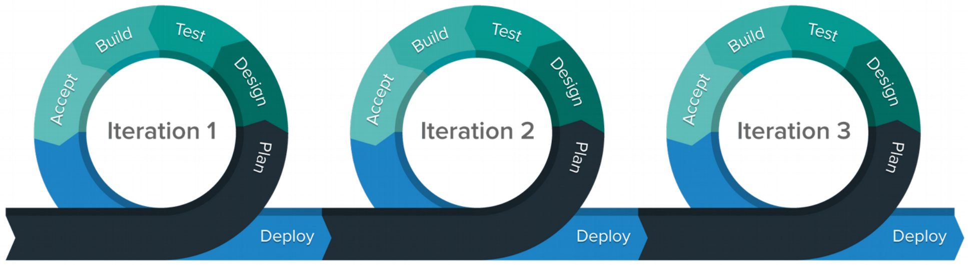
Disadvantages of Waterfall Model:

•Risky and uncertain
•Lack of visibility of the current progress
•Not suitable when the requirements keep changing
•Difficult to make changes to the product when it is in the testing phase

•The end product is available only at the end of the cycle
•Not suitable for large and complex projects

Agile Methodology

Agile Methodology is an iterative based software development approach where the software project is broken down into various iterations or sprints. Each iteration has phases like the waterfall model such as Requirements Gathering, Design, Development, Testing, and Maintenance. The duration of each iteration is generally 2-8 weeks.



Agile Process

•In Agile, a company releases the application with some high priority features in the first iteration.
•After its release, the end-users or the customers give you feedback about the performance of the application.
•Then you make the necessary changes into the application along with some new features and the application is again released which is the second iteration.
•You repeat this entire procedure until you achieve the desired software quality.

Advantages of Agile Model

•It adaptively responds to requirement changes favorably
•Fixing errors early in the development process makes this process more cost-effective
•Improves the quality of the product and makes it highly error-free
•Allows for direct communication between people involved in software project
•Highly suitable for large & long-term projects
•Minimum resource requirements & very easy to manage

Disadvantages of Agile Model

•Highly dependent on clear customer requirements
•Quite Difficult to predict time and effort for larger projects
•Not suitable for complex projects
•Lacks documentation efficiency
•Increased maintainability risks

Why is DevOps is Needed?

•Before DevOps, the development and operation team worked in complete isolation.

•Testing and Deployment were isolated activities done after design-build. Hence they consumed more time than actual build cycles.

•Without using DevOps, team members are spending a large amount of their time in testing, deploying, and designing instead of building the project.

•Manual code deployment leads to human errors in production

•Coding & operation teams have their separate timelines and are not in synch causing further delays.

How is DevOps different from traditional IT

let's compare traditional software waterfall model with DevOps to understand the changes DevOps bring.

Old Process	DevOps
After placing an order for new servers, the Development team works on testing. The Operations team works on extensive paperwork as required in enterprises to deploy the infrastructure.	After placing an order for new servers Development and Operations team work together on the paperwork to set-up the new servers. This results in better visibility of infrastructure requirement.
Projection about failover, redundancy, data center locations, and storage requirements are skewed as no inputs are available from developers who have deep knowledge of the application.	Projection about failover, redundancy, disaster recovery, data center locations, and storage requirements are pretty accurate due to the inputs from the developers.
Operations team has no clue on the progress of the Development team. Operations team develop a monitoring plan as per their understanding.	In DevOps, the Operations team is completely aware of the progress the developers are making. Operations team interact with developers and jointly develop a monitoring plan that caters to the IT and business needs. They also use advance Application Performance Monitoring (APM) Tools
Before go-live, the load testing crashes the application. The release is delayed.	Before go-live, the load testing makes the application a bit slow. The development team quickly fixes the bottlenecks. The application is released on time.

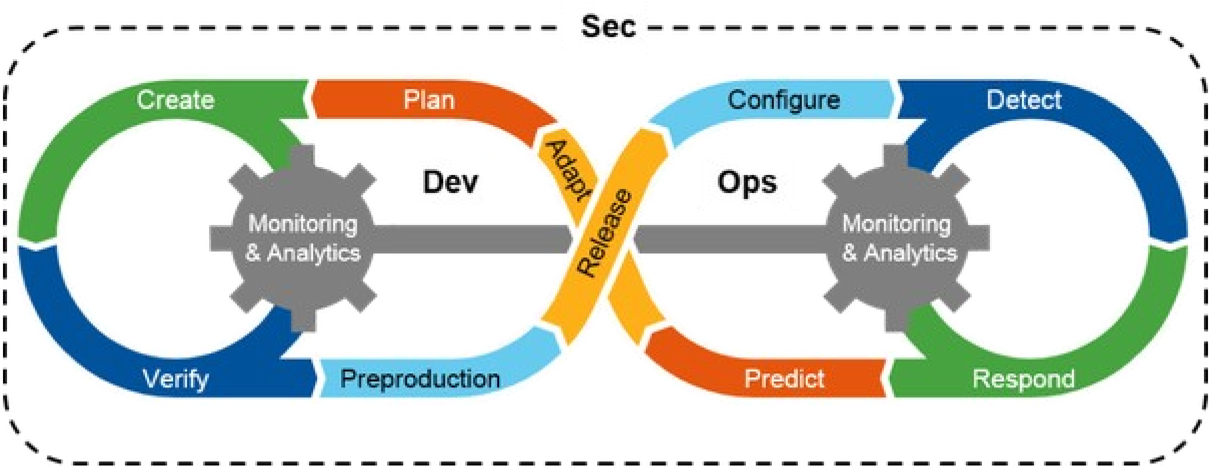
Why is DevOps used?

DevOps allows Agile Development Teams to implement Continuous Integration and Continuous Delivery. This helps them to launch products faster into the market.

When to adopt DevOps?

DevOps should be used for large distributed applications such as eCommerce sites or applications hosted on a cloud platform.

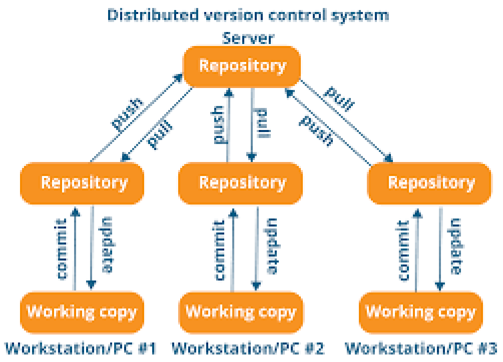
DevOps Lifecycle



Stage – 1: Continuous Development

Tools Used: Git, SVN, Mercurial, CVS

Process Flow:

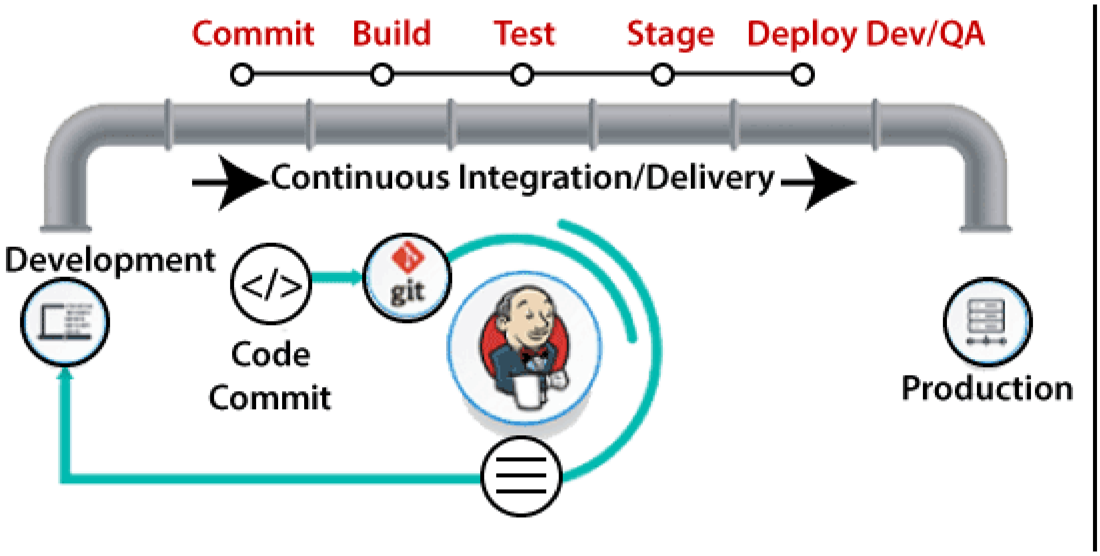


- This is the phase that involves ‘planning‘ and ‘coding‘ of the software. You decide the project vision during the planning phase and the developers begin developing the code for the application.
- There are no **DevOps tools** that are required for planning, but there are a number of tools for maintaining the code.
- The code can be in any language, but you maintain it by using Version Control tools. This process of maintaining the code is known as Source Code Management.

Stage – 2: Continuous Integration

Tools: Jenkins, TeamCity, Travis
Process Flow:

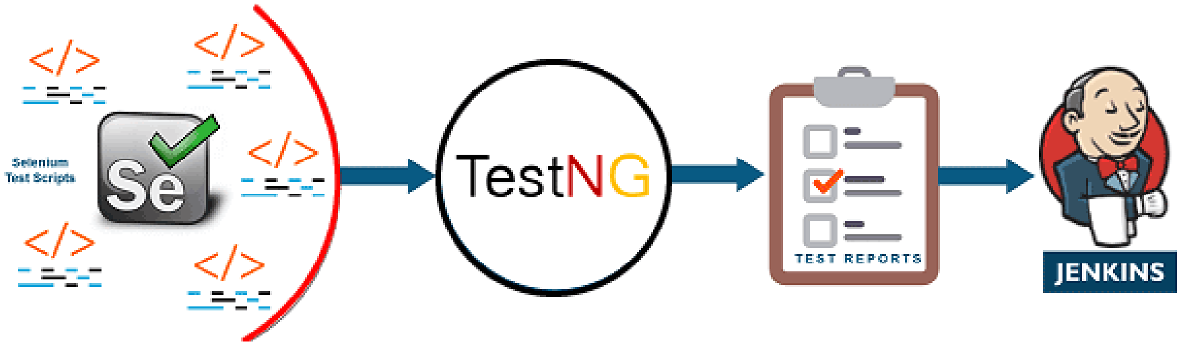
•This stage is the core of the entire DevOps life cycle. It is a practice in which the developers require to commit changes to the source code more frequently. This may be either on a daily or weekly basis.
•You then build every commit and this allows early detection of problems if they are present. Building code not only involves compilation but it also includes code review, unit testing, integration testing, and packaging.



•The code supporting new functionality is **continuously integrated** with the existing code. Since there is a continuous development of software, you need to integrate the updated code continuously as well as smoothly with the systems to reflect changes to the end-users.
•In this stage, you use the tools for building/ packaging the code into an executable file so that you can forward it to the next phases.
•Jenkins is a popular tool used in this phase. Whenever there is a change in the Git repository, then Jenkins fetches the updated code and prepares a build of that code, which is an executable file in the form of war or jar. Then this build is forwarded to the test server or the production server.

Stage – 3: Continuous Testing

Tools: Jenkins, Selenium TestNG, JUnit
Process Flow:

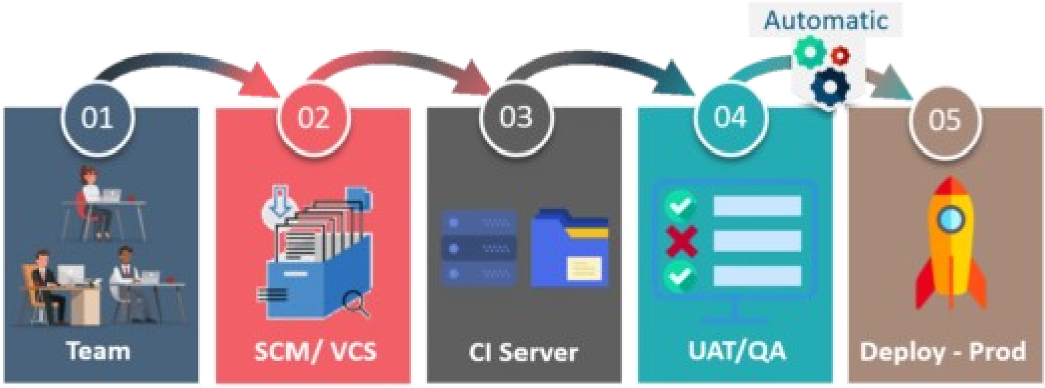


- **This is the stage where you test the developed software continuously for bugs using automation testing tools. These tools allow QAs to test multiple code-bases thoroughly in parallel to ensure that there are no flaws in the functionality. In this phase, you can use Docker Containers for simulating the test environment.**
- **Selenium** is used for automation testing, and the reports are generated by **TestNG**. You can automate this entire testing phase with the help of a Continuous Integration tool called Jenkins.
- Suppose you have written a selenium code in Java to test your application. Now you can build this code using ant or maven. Once you build the code, you then test it for User Acceptance Testing (UAT). This entire process can be automated using **Jenkins**.

Stage – 4: Continuous Deployment

Tools Used:Configuration Management – Chef, Puppet, Ansible
Containerization – Docker, Vagrant
Process Flow:

- This is the stage where you deploy the code on the production servers. It is also important to ensure that you correctly deploy the code on all the servers. Before moving on, let us try to understand a few things about Configuration management and **Containerization tools**. These set of tools here help in achieving Continuous Deployment (CD).



•**Configuration Management** is the act of establishing and maintaining consistency in an application's functional requirements and performance. Let me put this in easier words, it is the act of releasing deployments to servers, scheduling updates on all servers and most importantly keeping the configurations consistent across all the servers.

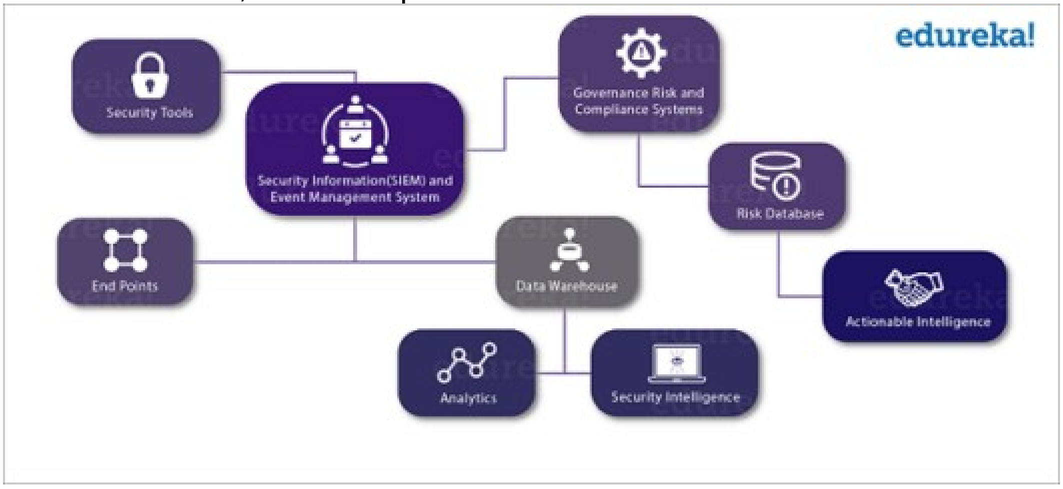
•Containerization tools also play an equally crucial role in the deployment stage. The containerization tools help produce consistency across Development, Test, Staging as well as Production environments. Besides this, they also help in scaling-up and scaling-down of instances swiftly.

Stage – 5: Continuous Monitoring

Tools Used: Splunk, ELK Stack, Nagios, New Relic

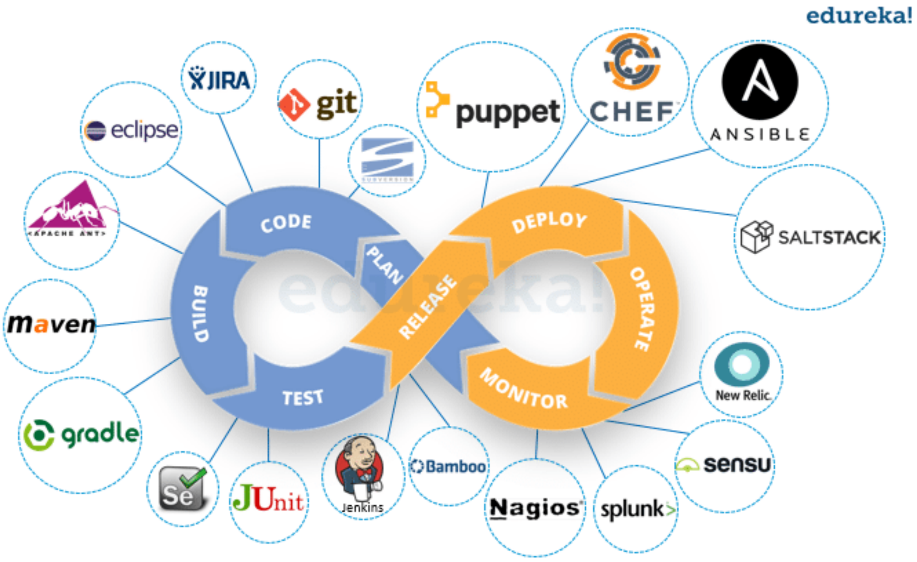
Process Flow:

•This is a very critical stage of the DevOps life cycle where you continuously monitor the performance of your application. Here you record vital information about the use of the software. You then process this information to check the proper functionality of the application. You resolve system errors such as low memory, server not reachable, etc in this phase.



•This practice involves the participation of the Operations team who will monitor the user activity for bugs or any improper behavior of the system.The Continuous Monitoring tools help you monitor the application's performance and the servers closely and also enable you to check the health of the system proactively.

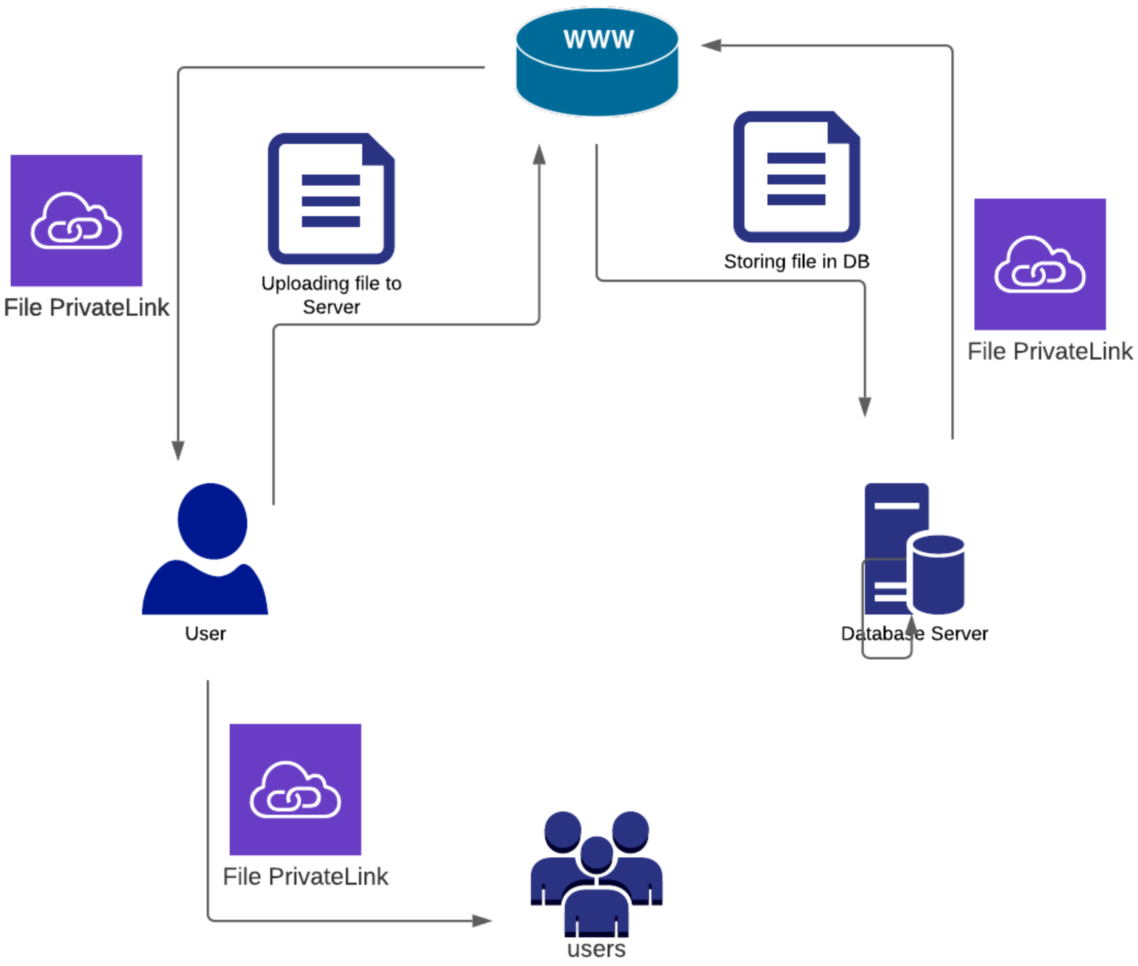
DevOps Tools



Git

Git is an open-source distributed version control system that is freely available for everyone. It is designed to handle minor to major projects with speed and efficiency. It is developed to co-ordinate the work among programmers. The version control allows you to track and work together with your team members at the same workspace. It is used as a critical distributed version-control for the DevOps tool.

3 . PROJECT ARCHITECTURE



User uploads a file to the website, and the offscreen website stores that file in the db and sends back a file url. User can share that url to others to avail that file.

4. PROJECT FUNCTIONALITIES & REQUIREMENTS

Drag & Drop files: Users can upload files either drag & drop or by simply clicking on the upload option.

Send an email to the receiver: Users can share the file link by sending an email.

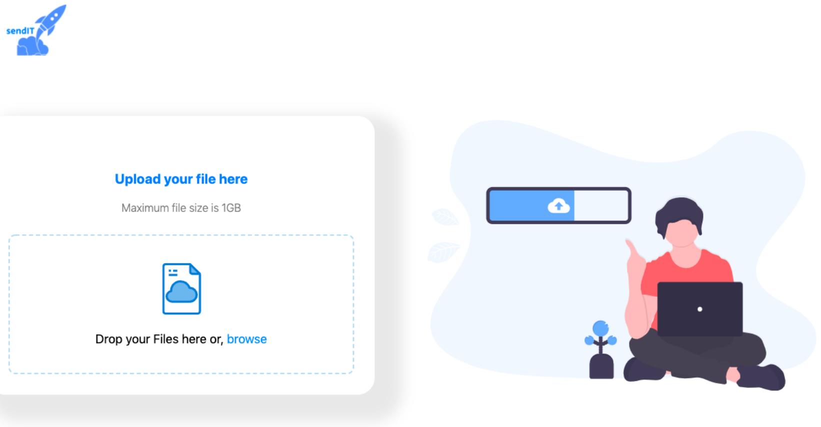
Link Sharing: Once a file gets uploaded, It shows a link. Users can easily share that link across all platforms.

24 hrs File Storage: Uploaded video will be available for next 24 hours.

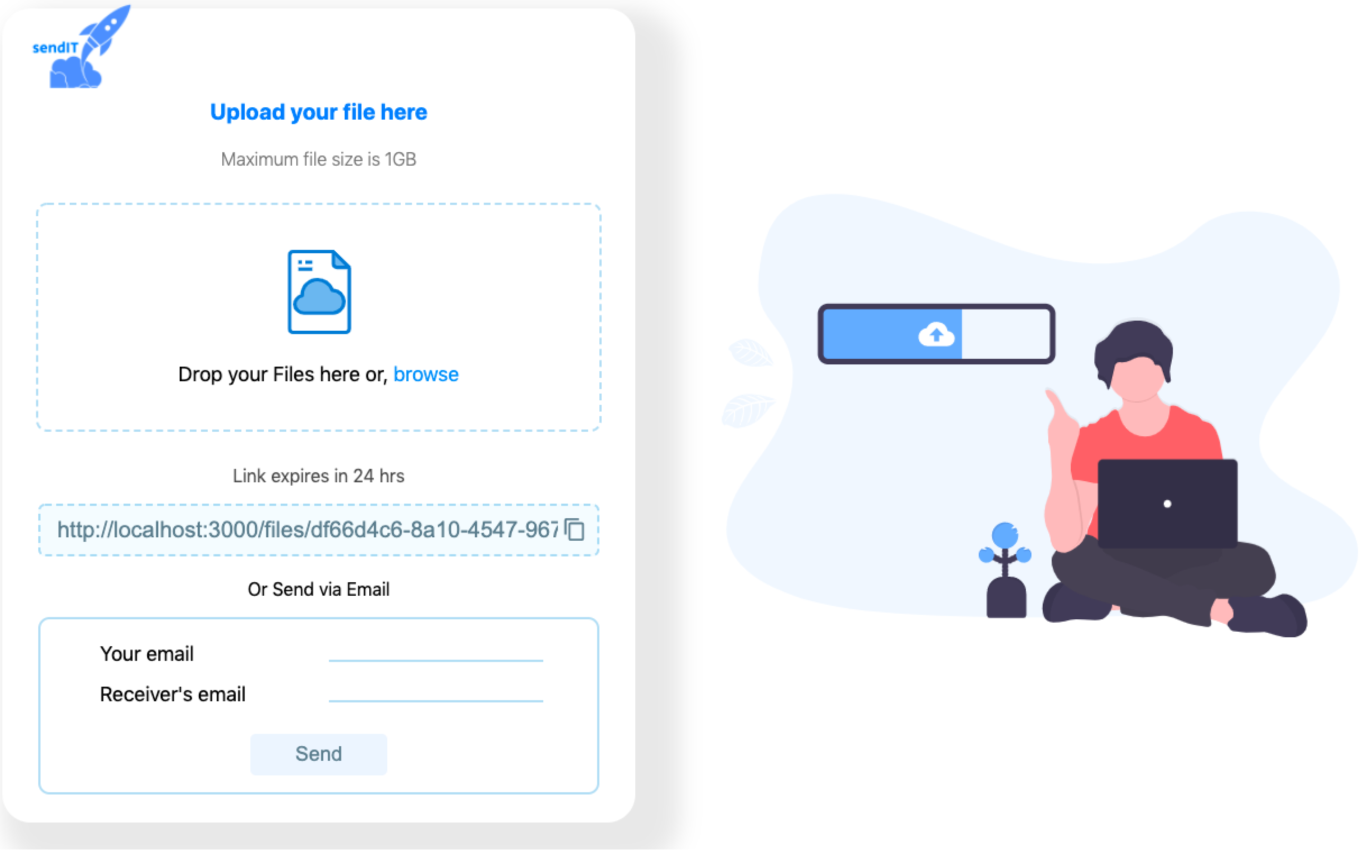
5. UI DESIGNS

USER INTERFACE:

5.1 Home Page:



5.2 AFTER FILE BEING UPLOADED:



Upload your file here

Maximum file size is 1GB

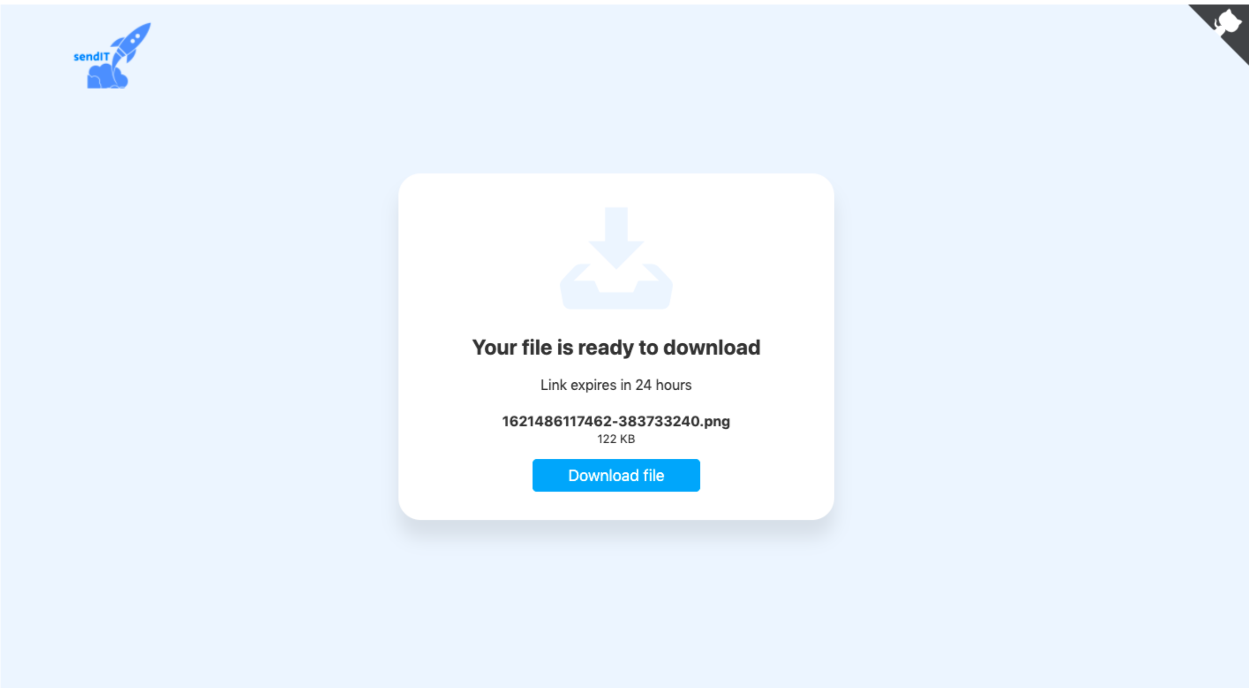
Drop your Files here or, browse

Link expires in 24 hrs

http://localhost:3000/files/df66d4c6-8a10-4547-967

Or Send via Email

5.3 DOWNLOAD PAGE:



Your file is ready to download

Link expires in 24 hours

1621486117462-383733240.png
122 KB

6. TECHNOLOGIES

To start any product some technologies must be needed. So currently trending technologies are NodeJS, HTML and MongoDB.

6.1 HTML & CSS:

The HyperText Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. .HTMLprovides a means to create structured

documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.

CSS stands for Cascading Style Sheets. **CSS** describes how HTML elements are to be displayed on screen, paper, or in other media. **CSS** saves a lot of work. It can control the layout of multiple web pages all at once. External stylesheets are stored in **CSS** files.

JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. Where HTML and CSS are languages that give structure and style to web pages,JavaScript gives web pages interactive elements that engage a use

6.2 NodeJS & ExpressJS

Node.js is a cross-platform runtime environment and library for running JavaScript applications outside the browser. It is used for creating server-side and networking web applications. It is open source and free to use. Node.js also provides a rich library of various JavaScript modules to simplify the development of web applications.

Express.js, or simply called Express. Express framework is able to layer in built-in structure and functions needed to actually build a site. It’s a pretty lightweight framework that’s great for giving developers extra, built-in web application features and the Express API without overriding the already robust, feature-packed Node.js platform. In short, Express and Node are changing the way developers build websites.

6.3 MongoDB

MongoDB is an open source database that uses a document-oriented data model. MongoDB is one of several database types to arise in the mid-2000s under the NoSQL banner. Instead of using tables and rows as in relational databases, MongoDB is built on an architecture of collections and documents. Documents comprise sets of key-value pairs and are the basic unit of data in MongoDB. Collections contain sets of documents and function as the equivalent of relational database tables.

MongoDB provides high performance data persistence. In particular, Support for embedded data models reduces I/O activity on database system. Indexes support faster queries and can include keys from embedded documents and arrays.

Advantages of MongoDB over RDBMS

- **Schema less** − MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.
- Structure of a single object is clear.
- No complex joins.
- Deep query-ability. MongoDB supports dynamic queries on documents using a document-based query language that's nearly as powerful as SQL.
- Tuning.
- **Ease of scale-out** − MongoDB is easy to scale.
- Conversion/mapping of application objects to database objects not needed.
- Uses internal memory for storing the (windowed) working set, enabling faster access of data.

6.4 RESTful API

REST stands for **RE**presentational **S**tate **T**ransfer. REST is a web standards-based architecture and uses HTTP Protocol for data communication. It revolves around resources where every component is a resource and resources are accessed by a common interface using HTTP standard methods.

In REST architecture, a REST Server simply provides access to resources and the REST client accesses and presents the resources. Here each resource is identified by URIs/ Global IDs. REST uses various representations to represent a resource like Text, JSON and XML.

6.5 HTTP Methods

The following HTTP methods are most commonly used in a REST based architecture.

- **GET** − Provides a read only access to a resource.

- **PUT** − Used to create a new resource.

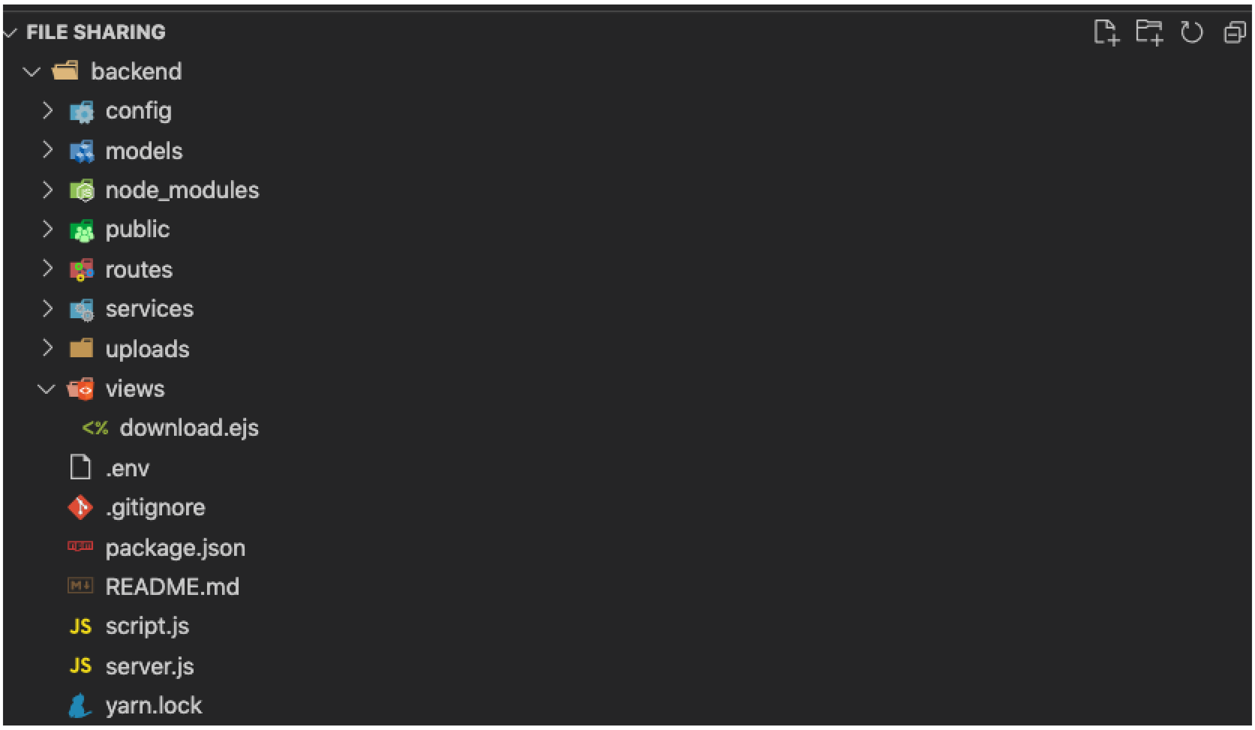
- **DELETE** − Used to remove a resource.

- **POST** − Used to update an existing resource or create a new resource.

- **OPTIONS** − Used to get the supported operations on a resource.

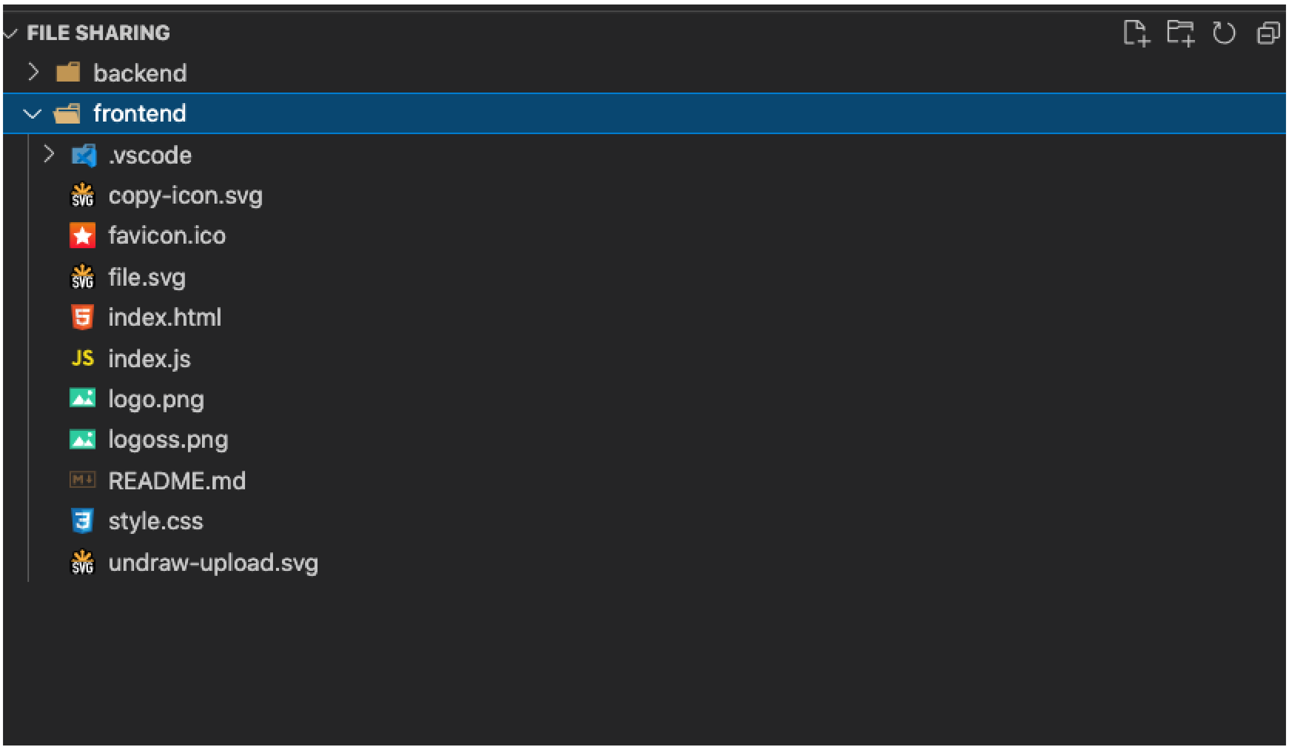
7. PROJECT STRUCTURE

7.1 BACKEND FILE STRUCTURE:



```
FILE SHARING
  backend
    config
    models
    node_modules
    public
    routes
    services
    uploads
    views
      download.ejs
    .env
    .gitignore
    package.json
    README.md
    script.js
    server.js
    yarn.lock
```

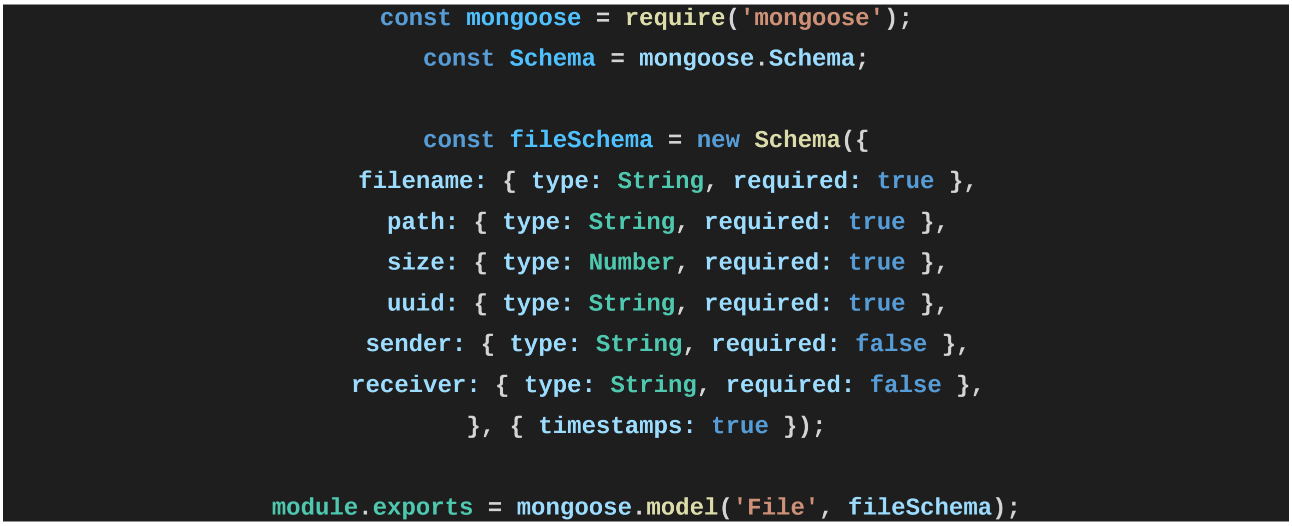
7.2 FRONTEND FILE STRUCTURE



```
FILE SHARING
  > backend
  v frontend
    > .vscode
      copy-icon.svg
      favicon.ico
      file.svg
      index.html
      index.js
      logo.png
      logoss.png
      README.md
      style.css
      undraw-upload.svg
```

8. DATABASE SCHEMAS

8.1 File Schema:



```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;

const fileSchema = new Schema({
filename: { type: String, required: true },
path: { type: String, required: true },
size: { type: Number, required: true },
uuid: { type: String, required: true },
sender: { type: String, required: false },
receiver: { type: String, required: false },
}, { timestamps: true });

module.exports = mongoose.model('File', fileSchema);
```

9.API's

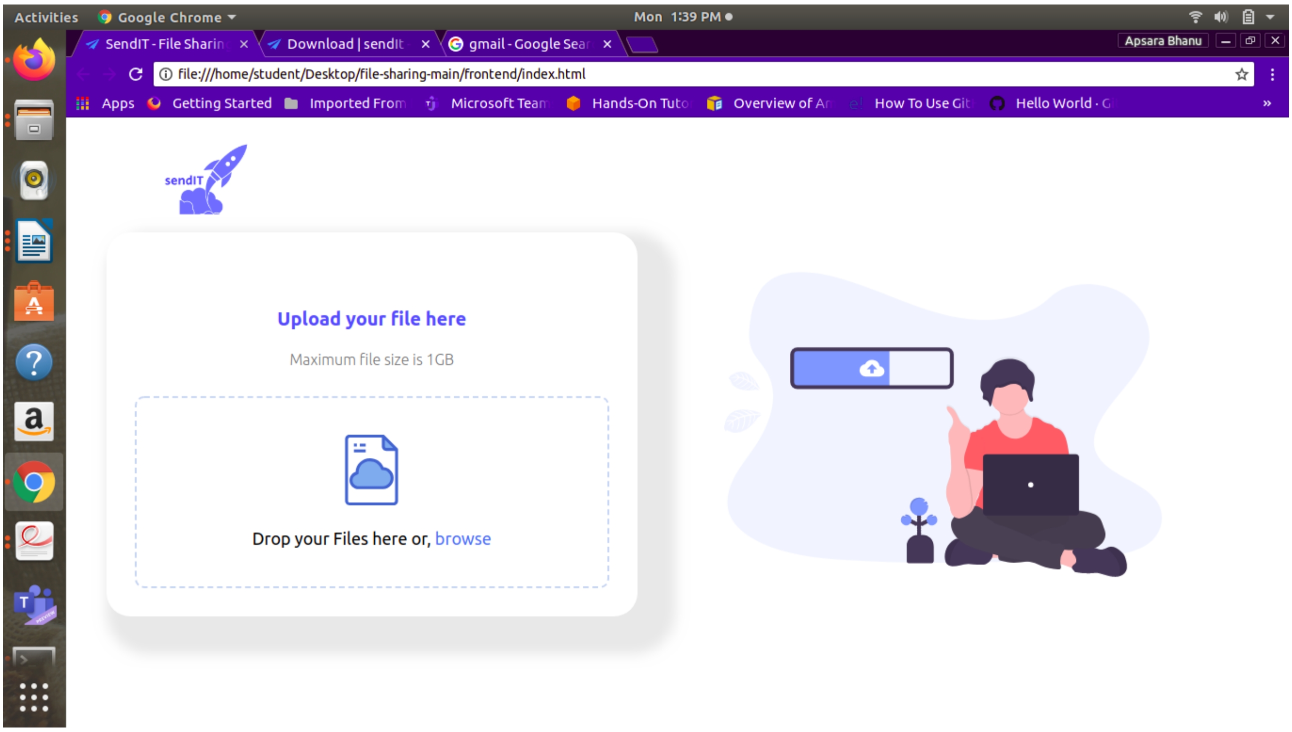
UPLOAD FILES: /api/files

SEND FILE LINK TO EMAIL: /api/files/send

SHOW FILE: files/:uuid

DOWNLOAD FILE: files/download/:uuid

10.OUTPUT SCREENS

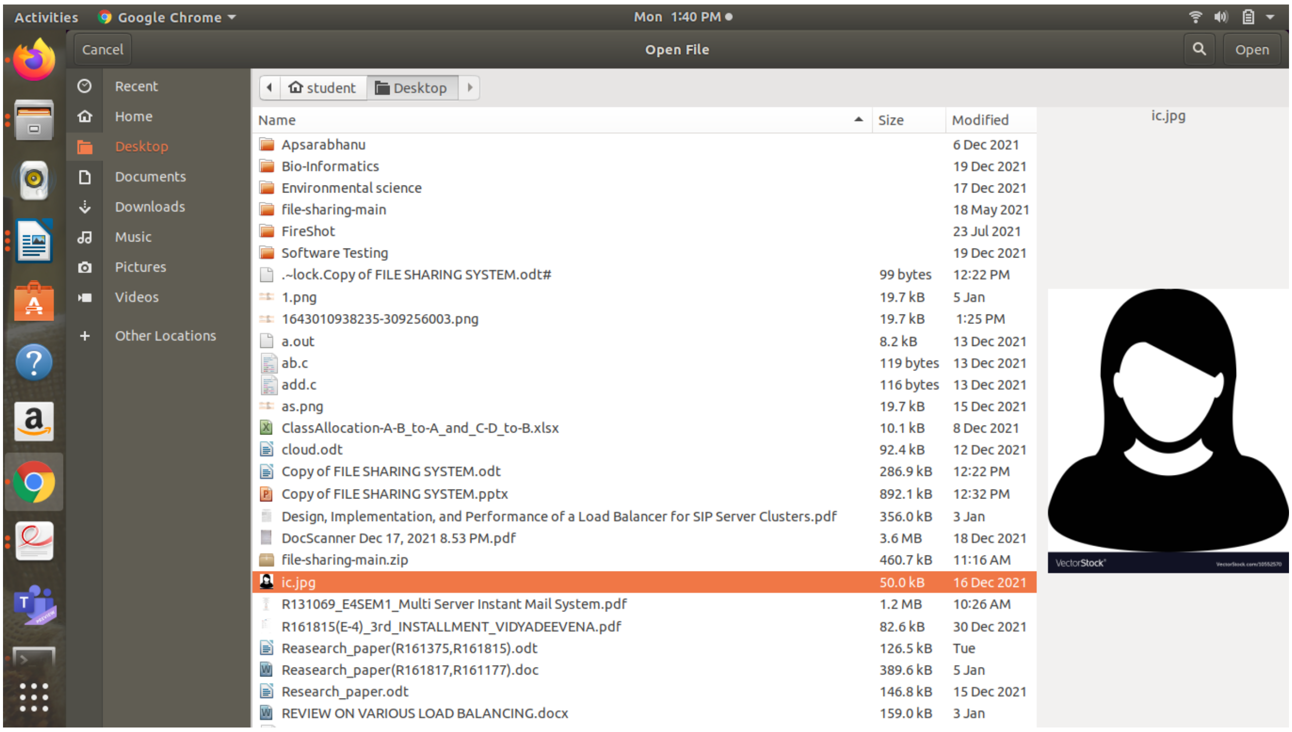


file:///home/student/Desktop/file-sharing-main/frontend/index.html

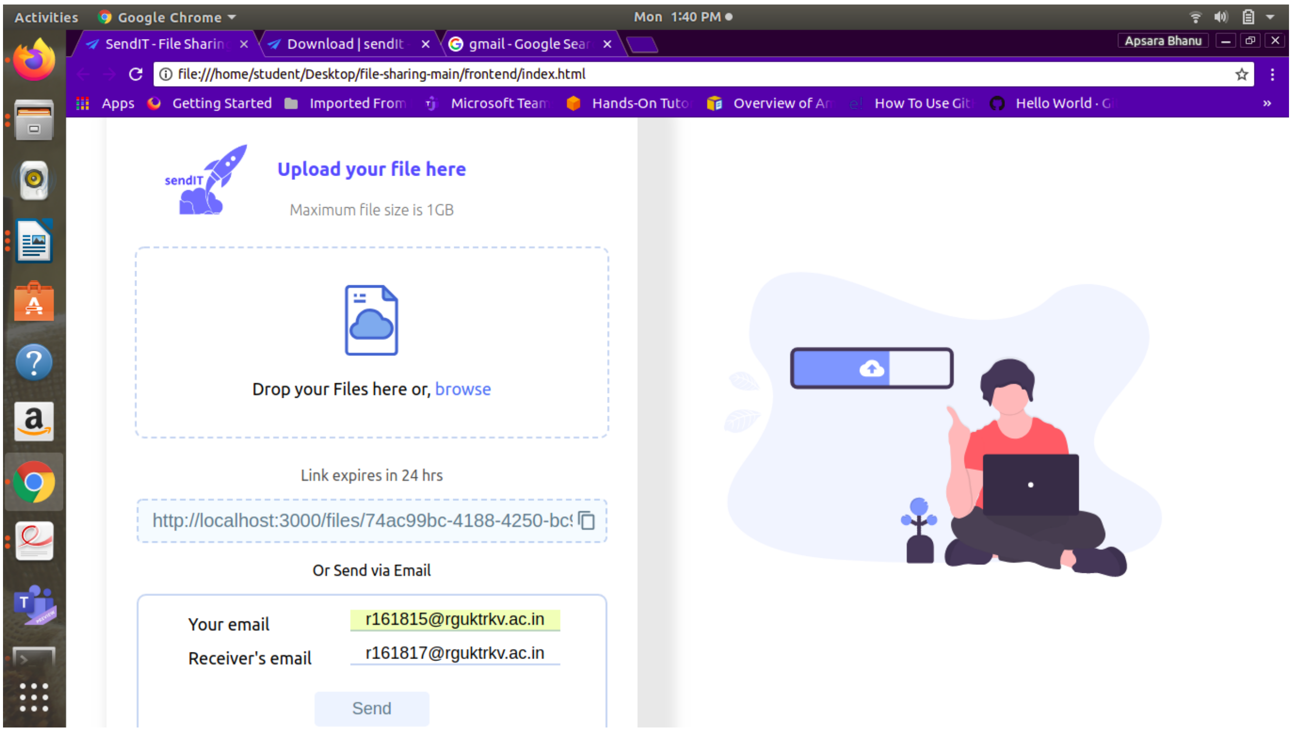
Upload your file here

Maximum file size is 1GB

Drop your Files here or, browse



Name	Size	Modified
Apsarabhanu		6 Dec 2021
Bio-Informatics		19 Dec 2021
Environmental science		17 Dec 2021
file-sharing-main		18 May 2021
FireShot		23 Jul 2021
Software Testing		19 Dec 2021
.~lock.Copy of FILE SHARING SYSTEM.odt#	99 bytes	12:22 PM
1.png	19.7 kB	5 Jan
1643010938235-309256003.png	19.7 kB	1:25 PM
a.out	8.2 kB	13 Dec 2021
ab.c	119 bytes	13 Dec 2021
add.c	116 bytes	13 Dec 2021
as.png	19.7 kB	15 Dec 2021
ClassAllocation-A-B_to-A_and_C-D_to-B.xlsx	10.1 kB	8 Dec 2021
cloud.odt	92.4 kB	12 Dec 2021
Copy of FILE SHARING SYSTEM.odt	286.9 kB	12:22 PM
Copy of FILE SHARING SYSTEM.pptx	892.1 kB	12:32 PM
Design, Implementation, and Performance of a Load Balancer for SIP Server Clusters.pdf	356.0 kB	3 Jan
DocScanner Dec 17, 2021 8.53 PM.pdf	3.6 MB	18 Dec 2021
file-sharing-main.zip	460.7 kB	11:16 AM
ic.jpg	50.0 kB	16 Dec 2021
R131069_E4SEM1_Multi Server Instant Mail System.pdf	1.2 MB	10:26 AM
R161815(E-4)_3rd_INSTALLMENT_VIDYADEEVENA.pdf	82.6 kB	30 Dec 2021
Reasearch_paper(R161375,R161815).odt	126.5 kB	Tue
Reasearch_paper(R161817,R161177).doc	389.6 kB	5 Jan
Research_paper.odt	146.8 kB	15 Dec 2021
REVIEW ON VARIOUS LOAD BALANCING.docx	159.0 kB	3 Jan



Upload your file here

Maximum file size is 1GB

Drop your Files here or, browse

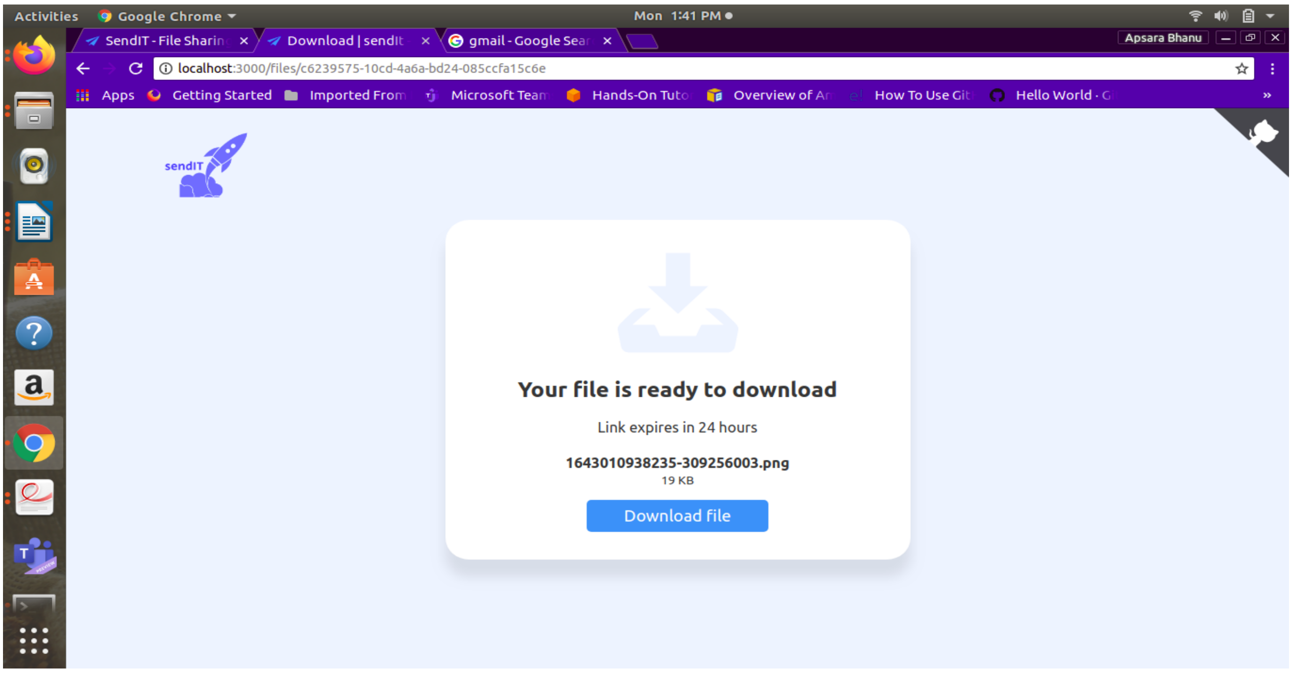
Link expires in 24 hrs

http://localhost:3000/files/74ac99bc-4188-4250-bc!

Or Send via Email

Your email r161815@rguktrkv.ac.in

Receiver's email r161817@rguktrkv.ac.in



Your file is ready to download

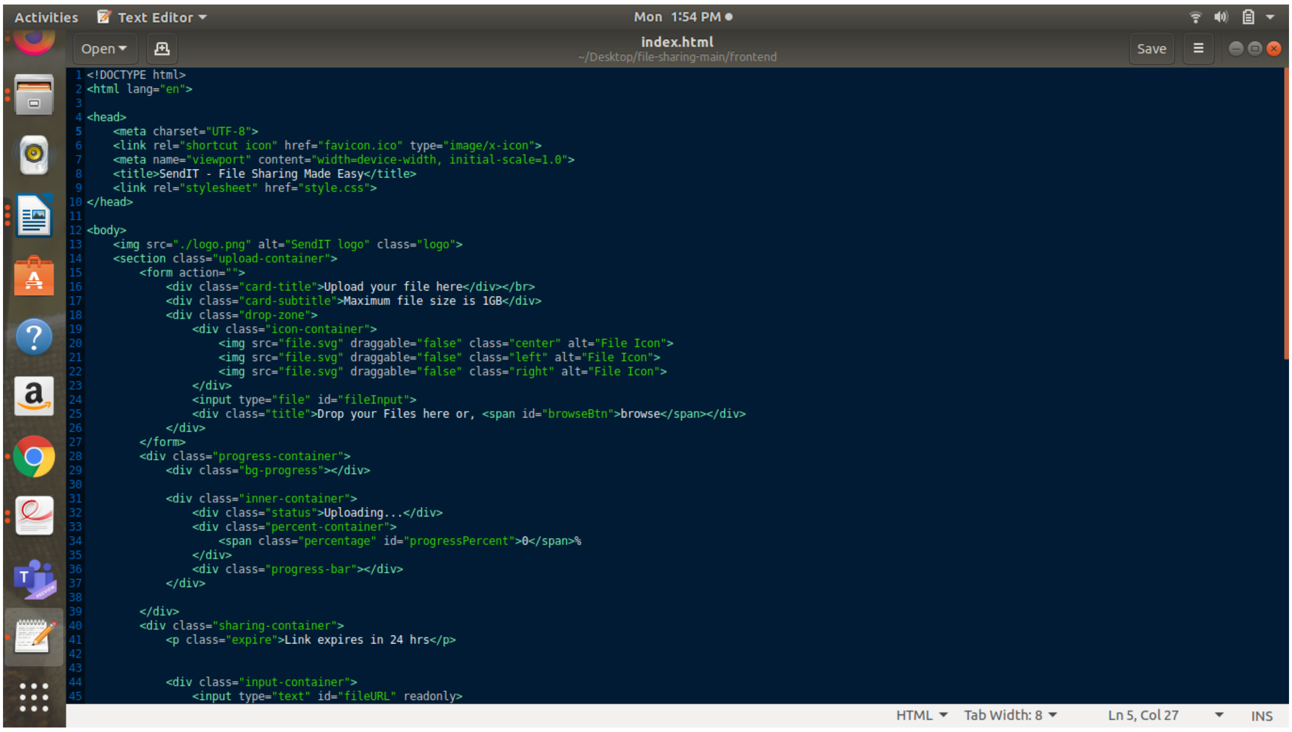
Link expires in 24 hours

1643010938235-309256003.png

19 KB

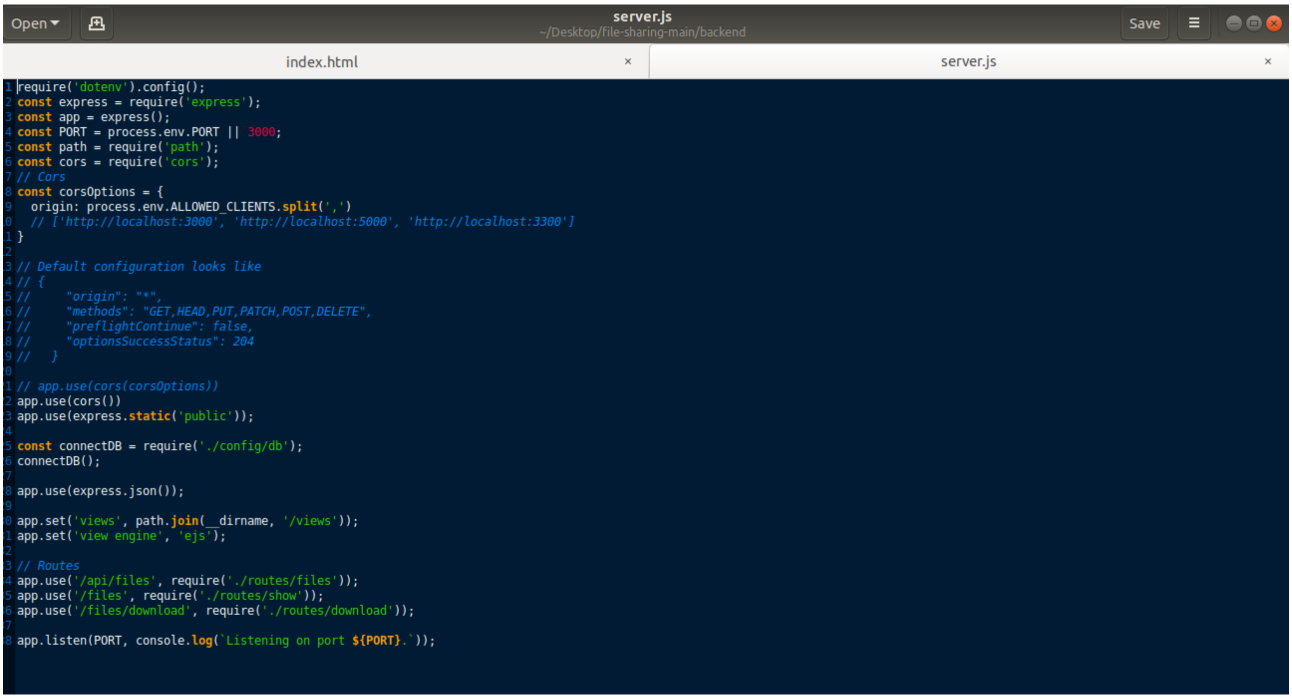
11.SAMPLE CODE:

front end:



```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>SendIT - File Sharing Made Easy</title>
    <link rel="stylesheet" href="style.css">
</head>

<body>
    <img src="./logo.png" alt="SendIT logo" class="logo">
    <section class="upload-container">
        <form action="">
            <div class="card-title">Upload your file here</div></br>
            <div class="card-subtitle">Maximum file size is 1GB</div>
            <div class="drop-zone">
                <div class="icon-container">
                    <img src="file.svg" draggable="false" class="center" alt="File Icon">
                    <img src="file.svg" draggable="false" class="left" alt="File Icon">
                    <img src="file.svg" draggable="false" class="right" alt="File Icon">
                </div>
                <input type="file" id="fileInput">
                <div class="title">Drop your Files here or, <span id="browseBtn">browse</span></div>
            </div>
        </form>
        <div class="progress-container">
            <div class="bg-progress"></div>

            <div class="inner-container">
                <div class="status">Uploading...</div>
                <div class="percent-container">
                    <span class="percentage" id="progressPercent">0</span>%
                </div>
                <div class="progress-bar"></div>
            </div>

        </div>
        <div class="sharing-container">
            <p class="expire">Link expires in 24 hrs</p>


            <div class="input-container">
                <input type="text" id="fileURL" readonly>
```

Backend:



```javascript
require('dotenv').config();
const express = require('express');
const app = express();
const PORT = process.env.PORT || 3000;
const path = require('path');
const cors = require('cors');
// Cors
const corsOptions = {
  origin: process.env.ALLOWED_CLIENTS.split(',')
  // ['http://localhost:3000', 'http://localhost:5000', 'http://localhost:3300']
}

// Default configuration looks like
// {
//     "origin": "*",
//     "methods": "GET,HEAD,PUT,PATCH,POST,DELETE",
//     "preflightContinue": false,
//     "optionsSuccessStatus": 204
//   }

// app.use(cors(corsOptions))
app.use(cors())
app.use(express.static('public'));

const connectDB = require('./config/db');
connectDB();

app.use(express.json());

app.set('views', path.join(__dirname, '/views'));
app.set('view engine', 'ejs');

// Routes
app.use('/api/files', require('./routes/files'));
app.use('/files', require('./routes/show'));
app.use('/files/download', require('./routes/download'));

app.listen(PORT, console.log(`Listening on port ${PORT}.`));
```

12.CONCLUSION

In this report we discuss how file sharing works and what are the technologies being used to develop it. which is the solution for emailing big files which are not allowed by providers. We have also noted their shortcomings.An interactive file sharing website with very few page refreshes is implemented using NodeJS and MongoDB as the back end. Users were given multiple views for viewing their file system. An analysis was done on where to store the uploaded files – database or file system.

13.BIBILOGRAPHY

https://www.edureka.co/blog/what-is-devops

https://www.javatpoint.com/devops-vs-agile

https://www.guru99.com/devops-tutorial.html#4

https://en.wikipedia.org/wiki/DevOps

www.google.com

https://github.com

https://guides.github.com/activities/hello-world7. http://www.telusko.com

https://aws.amazon.com

https://www.postman.com/

www.nodejs.org

www.mongodb.com

https://mongoosejs.com/