



**RAJALAKSHMI  
ENGINEERING COLLEGE**  
An AUTONOMOUS Institution  
Affiliated to ANNA UNIVERSITY, Chennai

**CLOUD BIN  
FILE MANAGEMENT AND SHARING SYSTEM**

**A MINI PROJECT REPORT**

*Submitted by*

|                         |                  |
|-------------------------|------------------|
| <b>GIRIDHAR U</b>       | <b>231501047</b> |
| <b>EZHIL ADHITHYA P</b> | <b>231501045</b> |
| <b>ARAVIND S</b>        | <b>231501019</b> |
| <b>HARISH T</b>         | <b>231501060</b> |

*In partial fulfillment of the award of the degree of*

***BACHELOR OF TECHNOLOGY***

***IN***

***ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING***

**RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)**

**THANDALAM, CHENNAI-602105**

**2024-2025**



# **RAJALAKSHMI ENGINEERING COLLEGE**

**An AUTONOMOUS Institution  
Affiliated to ANNA UNIVERSITY, Chennai**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**CLOUD BIN - FILE MANAGEMENT AND SHARING SYSTEM**” is the Bonafide work of “**GIRIDHAR U (231501047), EZHIL ADHITHYA P (231501045), ARAVIND S (23150101019), HARISH T (231501060)**” who carried out the project under my supervision.

**Submitted for Practical Examination held on \_\_\_\_\_**

### **SIGNATURE**

**Mr. U. Kumaran,  
Assistant Professor(SS),  
Artificial Intelligence and Machine Learning,  
Rajalakshmi Engineering College (Autonomous)  
Thandalam, Chennai-602105**

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## **ABSTRACT**

CloudBin is a proposed file management and sharing system designed to improve the organization, storage and sharing of digital files for individuals. It focuses on user-friendliness through an intuitive interface that allows users to easily upload, delete and share files. Key features include a Rule-Based AI Chatbot that enhances user interactions by providing automated responses based on predefined guidelines, making navigation within the system simpler. Users can securely upload files, ensuring that their documents are stored safely and privately. Additionally, the system includes a secured user authentication page to enhance security and protect access to user accounts. It also generates QR codes for each file, simplifying the sharing process and enhancing accessibility while maintaining security; this feature allows users to share files quickly without the need for complex links or email attachments. These functionalities work together to support efficient document management and protect user privacy. CloudBin is compatible with various file types, including PDFs, images and audio files, making it a versatile tool that enhances productivity and collaboration in both personal and professional settings.

# TABLE OF CONTENTS

|   |           |
|---|-----------|
| <b>1. INTRODUCTION.....</b>                 | <b>1</b>  |
| 1.1. Importance of CloudBin.....            | 1         |
| 1.2. Objectives of CloudBin.....            | 1         |
| 1.3. Key Features of CloudBin.....          | 4         |
| 1.4. Benefits of implementing ClouBin.....  | 4         |
| <b>2. SURVEY OF TECHNOLOGIES.....</b>       | <b>5</b>  |
| 2.1. Front End Technologies.....            | 6         |
| 2.2. Backend Technologies.....              | 6         |
| 2.3. Database Management System.....        | 7         |
| 2.4. Additional Modules.....                | 7         |
| <b>3. REQUIREMENTS AND FOUNDATIONS.....</b> | <b>10</b> |
| 3.2. Hardware Requirements.....             | 11        |
| 3.3. Software Requirements.....             | 13        |
| 3.3. Architecture Diagram.....              | 16        |
| 3.4. Entity Relationship (ER) Diagram.....  | 17        |
| <b>4. PROGRAM CODE.....</b>                 | <b>20</b> |
| <b>5. OUTPUT.....</b>                       | <b>73</b> |
| <b>6. RESULTS AND DISCUSSION.....</b>       | <b>82</b> |
| <b>7. CONCLUSION.....</b>                   | <b>84</b> |
| <b>8. REFERENCES.....</b>                   | <b>85</b> |

## **1) INTRODUCTION**

In today's digital landscape, effective file management and sharing are essential for enhancing productivity and collaboration across various sectors. As organizations and individuals increasingly rely on diverse digital formats - such as PDFs, images and audio files—the need for a robust system that can seamlessly handle multiple file types has never been greater. The proposed file management and sharing system addresses these challenges by offering an intuitive platform that simplifies the organization, storage and sharing of digital assets. With integrated cloud storage, users can access their files from anywhere, ensuring flexibility and scalability to meet growing demands. Additionally, features like advanced search capabilities, version control and customizable access permissions enhance security and streamline workflows.

### **1.1. Importance of CloudBin**

CloudBin serves as the backbone of digital asset organization. They provide a structured approach to storing, retrieving and managing files in an efficient manner. In a world where data is generated at an unprecedented rate, the ability to quickly locate and manage files is crucial for both individual productivity and organizational efficiency. A well-designed FMS allows users to categorize files logically, making it easier to find necessary documents when needed. This not only saves time but also reduces frustration associated with disorganized data repositories.

### **1.2. Objectives of CloudBin**

The primary objectives of CloudBin are designed to enhance user experience, streamline file management and foster collaboration among users. These objectives are essential for meeting the evolving needs of individuals and organizations in a digital-first environment. Below are the key objectives:

### **(i) Seamless File Management**

- ***Organized Storage:*** CloudBin aims to provide an intuitive interface for users to organize their files efficiently. Users can categorize files into folders, making it easier to locate and manage documents across various formats
- ***Multi-format Support:*** The system is designed to handle a diverse range of file types, including PDFs, images, audio files and more. This flexibility ensures that users can store all their digital assets in one place without compatibility issues

### **(ii) Enhanced Accessibility**

- ***Cloud-based Access:*** By leveraging cloud technology, CloudBin allows users to access their files from any device with an internet connection. This objective supports remote work and ensures that users can retrieve important documents anytime, anywhere
- ***Cross-device Synchronization:*** Users can sync their files across multiple devices, ensuring that they always have access to the latest versions of their documents regardless of where they are working

### **(iii) Improved Collaboration**

- ***Real-time Collaboration Tools:*** CloudBin integrates features that enable multiple users to work on documents simultaneously. This fosters teamwork and enhances productivity by allowing team members to edit, comment and share feedback in real time
- ***Secure File Sharing:*** The system provides robust sharing options that allow users to send files securely to colleagues or clients. Customizable access permissions ensure that sensitive information is only shared with authorized individuals

#### (iv) Advanced Security Features

- **Data Protection:** Security is a top priority for CloudBin. The system implements advanced encryption methods for data at rest and in transit, protecting user information from unauthorized access and breaches
- **Access Control Mechanisms:** Users can set permissions for who can view or edit their files, providing an additional layer of security. Detailed activity logs track access and modifications, ensuring accountability and transparency

#### (v) Scalability and Flexibility

- **Scalable Infrastructure:** As user demands grow, CloudBin is built on a scalable infrastructure that allows organizations to increase their storage capacity without significant upfront investments in hardware
- **Cost-effective Solutions:** By adopting a pay-as-you-go model for cloud storage, CloudBin enables users to pay only for the resources they utilize. This flexibility is particularly beneficial for small businesses looking to manage costs while expanding their digital capabilities

#### (vi) Enhanced User Experience

- **Intuitive User Interface:** The design of CloudBin focuses on user-friendliness, ensuring that even those with minimal technical expertise can navigate the system effortlessly
- **Support and Training:** To facilitate smooth onboarding and ongoing usage, CloudBin offers comprehensive support resources and training materials for users

### 1.3. Key Features of CloudBin

An effective file management system encompasses several key features that enhance usability and functionality:

- **Cloud Integration:** By leveraging cloud technology, users can access their files from any device with internet connectivity. This ensures that critical documents are always available, regardless of location
- **Advanced Search Capabilities:** Users can quickly locate files using metadata tagging or keyword searches. This feature is particularly important as the volume of stored data grows
- **Version Control:** This allows users to track changes made to documents over time, ensuring that the most current version is always accessible while retaining previous iterations for reference
- **Customizable Access Permissions:** Organizations can control who has access to specific files or folders, enhancing security and ensuring compliance with data protection regulations
- **Collaboration Tools:** Integrated tools such as commenting systems or real-time editing capabilities facilitate teamwork by allowing multiple users to work on documents simultaneously

### 1.4. Benefits of Implementing a File Management System

The implementation of a robust file management system offers numerous benefits:

- **Increased Productivity:** By streamlining file organization and retrieval processes, employees can spend less time searching for information and more time focusing on their core tasks
- **Improved Collaboration:** With features that support real-time collaboration, teams can work together more effectively, regardless of geographical barriers



- **Enhanced Security:** Customizable access controls ensure that sensitive information is only accessible to authorized personnel, reducing the risk of data breaches
- **Cost Savings:** Transitioning from traditional paper-based systems to digital file management can significantly reduce costs associated with physical storage, printing and document handling
- **Compliance and Audit Readiness:** A well-organized FMS helps organizations maintain compliance with industry regulations by providing clear records of document access and modifications

## 2) SURVEY OF TECHNOLOGIES

In developing CloudBin, a robust file management and sharing system, we utilized a modern tech stack that combines powerful frontend and backend technologies with an efficient database management system. The frontend is built using HTML, CSS and JavaScript, providing a responsive and user-friendly interface that enhances user experience. On the backend, Node.js and Express.js work together to create a scalable and efficient server environment capable of handling multiple requests simultaneously. MongoDB serves as the database management system, offering flexibility in data storage and retrieval through its NoSQL architecture. This combination of technologies ensures that CloudBin is not only functional but also capable of meeting the demands of users in an increasingly digital world. The combination of these technologies in CloudBin not only enhances performance but also ensures that the application remains adaptable to future needs. By leveraging modern web development practices and tools, CloudBin aims to provide users with an intuitive platform for effective file management and sharing.

## 2.1. Frontend Technologies:

- **HTML (HyperText Markup Language)**

HTML is the foundational markup language used to structure content on the web. In CloudBin, HTML provides the basic layout for all pages, ensuring that content is organized logically and semantically. This structure is crucial for accessibility and SEO (Search Engine Optimization).

- **CSS (Cascading Style Sheets)**

CSS is utilized to style the HTML elements, enhancing the visual presentation of CloudBin. It allows developers to create visually appealing layouts, apply colors, fonts and spacing and ensure that the application is responsive across different devices. By leveraging CSS frameworks like Bootstrap or custom stylesheets, we can achieve a modern look and feel.

- **JavaScript**

JavaScript adds interactivity to the CloudBin application. It enables dynamic content updates without requiring page reloads, enhances user engagement through interactive elements such as buttons and forms and allows for real-time data handling. JavaScript libraries and frameworks, such as React or Vue.js, can be integrated to further streamline development and improve user experience.

## 2.2. Backend Technologies:

- **Node.js**

Node.js is a JavaScript runtime built on Chrome's V8 engine that allows developers to execute JavaScript code server-side. It is known for its non-blocking I/O model, which makes it lightweight and efficient for handling multiple requests simultaneously. In CloudBin, Node.js provides the foundation for building scalable network applications.

- **Express.js**

Express.js is a minimalistic web application framework for Node.js that simplifies the process of building web applications and APIs. It provides robust features for routing, middleware integration and request handling. In CloudBin, Express.js facilitates communication between the frontend and backend by managing API endpoints efficiently.

## **2.3. Database Management:**

- **MongoDB**

MongoDB is a NoSQL database that stores data in flexible JSON-like documents. This document-oriented approach allows for easy storage and retrieval of complex data structures without requiring a fixed schema. In CloudBin, MongoDB enables efficient management of user authentication data and uploaded files while supporting scalability through its ability to handle large volumes of data across distributed systems.

## **2.4. Additional Modules:**

- **Mongoose**

Mongoose is an Object Data Modeling (ODM) library designed to work with MongoDB and Node.js. It provides a schema-based solution to model application data, allowing developers to enforce a specific structure on their data collections. This is particularly beneficial in a schema-less database like MongoDB, where data can vary significantly. Mongoose facilitates:

- ***Schema Validation:*** Ensures that documents adhere to defined structures before being saved to the database
- ***Relationships Management:*** Allows for the establishment of relationships between different data models
- ***Middleware Support:*** Provides hooks for pre and post-processing of data operations (e.g., validation or transformation)

- **CRUD Operations:** Simplifies Create, Read, Update and Delete operations through an intuitive API

Mongoose enhances the developer experience by abstracting complex MongoDB queries into simpler JavaScript methods while maintaining performance and scalability.

- **Bcryptjs**

Bcryptjs is a library used for hashing passwords securely. It implements the bcrypt algorithm, which is designed to be computationally intensive to deter brute-force attacks on password databases. By using Bcryptjs, CloudBin ensures that user credentials are stored securely, minimizing the risk of unauthorized access. Key features include:

- **Password Hashing:** Converts plain-text passwords into hashed strings that are stored in the database
- **Salting:** Automatically generates a unique salt for each password hash, adding an additional layer of security
- **Verification:** Provides methods to compare hashed passwords against user input during authentication processes

- **Body-parser**

Body-parser is middleware used in Express applications to parse incoming request bodies before they reach the route handlers. It supports various content types such as JSON and URL-encoded data. Key functionalities include:

- **Parsing Request Bodies:** Makes it easy to access form data submitted by users through req.body
- **Middleware Integration:** Can be configured to handle different types of requests based on content type

This module simplifies data handling within CloudBin by ensuring that incoming data is processed correctly before being utilized in application logic

- **Cookie-parser**

Cookie-parser is middleware that parses cookies attached to client requests. It provides an easy way to access cookie values from incoming requests in Express applications. Key features include:

- ***Cookie Parsing:*** Converts cookie header strings into a JavaScript object accessible via req.cookies
- ***Session Management:*** Facilitates user session management by allowing the storage of session identifiers or tokens in cookies

In CloudBin, cookie-parser helps maintain user sessions securely while enabling personalized experiences across sessions.

- **Multer**

Multer is middleware specifically designed for handling multipart/form-data, primarily used for file uploads. Its key features include:

- ***File Upload Handling:*** Simplifies the process of receiving files from client requests
- ***Storage Configuration:*** Allows developers to configure storage options (e.g., in-memory or disk storage) based on application needs
- ***File Filtering:*** Enables validation of file types before processing uploads

Multer is essential for CloudBin as it allows users to upload files seamlessly while ensuring that only valid files are accepted.

- **Path**

The Path module is a built-in Node.js module that provides utilities for working with file and directory paths. Its key functionalities include:

- ***Path Manipulation:*** Offers methods for resolving paths, joining paths together and extracting components from paths (e.g., file extensions)
- ***Cross-platform Compatibility:*** Ensures that path operations work consistently across different operating systems (Windows, macOS, Linux)

In CloudBin, the Path module aids in managing file locations effectively during upload and retrieval processes.

- **QRCode**

QRCode is a library used for generating QR codes within applications. This module enhances user experience by enabling easy sharing of files through scannable QR codes. Its primary features include:

- ***Dynamic QR Code Generation:*** Allows users to create QR codes that link directly to files or resources stored in CloudBin
- ***Customization Options:*** Provides options for customizing the appearance of QR codes (e.g., size and error correction levels)

### **3) REQUIREMENTS AND FOUNDATIONS**

To successfully develop and deploy CloudBin, specific hardware requirements must be met to ensure optimal performance and user experience. The following hardware specifications are designed to support the application's frontend, backend and database functionalities while accommodating potential growth in user demand.

## 3.1. HARDWARE REQUIREMENTS

### (i) Development Environment

For developers working on CloudBin, the following hardware specifications are recommended:

❖ **Processor (CPU):**

- Minimum: Dual-core processor (2.0 GHz or faster)
- Recommended: Quad-core processor (Intel i5 or equivalent)

❖ **Memory (RAM):**

- Minimum: 8 GB
- Recommended: 16 GB or more for smoother multitasking and handling of development tools

❖ **Storage:**

- Minimum: 256 GB SSD (Solid State Drive) for faster read/write speeds.
- Recommended: 512 GB SSD or larger to accommodate project files, dependencies and version control repositories

❖ **Graphics Card (GPU):**

- Integrated graphics are sufficient for development purposes; however, a dedicated GPU may be beneficial for tasks involving graphic design or video processing

❖ **Network:**

- Reliable internet connection with a minimum speed of 10 Mbps for efficient access to cloud services and collaboration tools

### (ii) Production Environment

For deploying CloudBin in a production environment, the following hardware specifications are recommended:

❖ ***Server Processor (CPU):***

- Minimum: Quad-core processor (2.5 GHz or faster)
- Recommended: Six-core processor (Intel Xeon or equivalent) to handle multiple concurrent requests efficiently

❖ ***Memory (RAM):***

- Minimum: 16 GB
- Recommended: 32 GB or more to support high traffic and multiple simultaneous users

❖ ***Storage:***

- Minimum: 512 GB SSD for fast access to files and databases.
- Recommended: 1 TB SSD or larger with RAID configuration for redundancy and improved data security

❖ ***Network:***

- High-speed internet connection with at least 100 Mbps bandwidth to ensure quick data transfer rates and support multiple users accessing the application simultaneously

❖ ***Backup Solutions:***

- External storage solutions or cloud backup systems to ensure data redundancy and recovery in case of hardware failure

### **(iii) Additional Considerations**

- ❖ ***Scalability:*** Ensure that the server infrastructure can be easily scaled up as user demand increases. This may involve using cloud services that allow for dynamic resource allocation
- ❖ ***Power Supply:*** A reliable power supply with backup options (such as UPS systems) is essential to prevent downtime during power outages
- ❖ ***Cooling Systems:*** Adequate cooling solutions should be in place to maintain optimal operating temperatures for servers, especially in data center environments



### 3.2) SOFTWARE REQUIREMENTS:

To ensure the successful development and deployment of CloudBin, specific software requirements must be met. The software stack includes an operating system, web server, database management system, development frameworks and various libraries that enhance functionality and security. This combination of software tools is essential for creating a seamless user experience while maintaining the integrity and performance of the application.

#### Operating System

- **Windows/Linux/Unix:** CloudBin can be developed and deployed on various operating systems. The choice of OS may depend on the server environment and developer preferences. Linux is often preferred for server environments due to its stability, security and performance capabilities

#### Web Server

- **Node.js:** As the runtime environment for executing JavaScript on the server side, Node.js is essential for CloudBin. It allows for building scalable network applications and handling multiple client requests simultaneously
- **Express.js:** This web application framework for Node.js simplifies the development of web applications and APIs. It provides robust features for routing, middleware integration and request handling

#### Database Management System

- **MongoDB:** A NoSQL database that stores data in flexible JSON-like documents. MongoDB is ideal for CloudBin due to its ability to handle unstructured data efficiently and its scalability features, which are crucial for managing large volumes of user-uploaded files

## Development Tools

- ***Integrated Development Environment (IDE)***: Tools like Visual Studio Code or WebStorm are recommended for writing code efficiently with features such as syntax highlighting, debugging support and version control integration
- ***Version Control System***: Git is essential for managing code changes and collaborating with other developers. Platforms like GitHub or GitLab can be used to host repositories

## Middleware Libraries

- ***Mongoose***: An ODM (Object Data Modeling) library that provides a schema-based solution to model application data in MongoDB. Mongoose simplifies interactions with the database by providing an intuitive API for CRUD operations
- ***Bcryptjs***: A library used for hashing passwords securely. It helps protect user credentials by converting plain-text passwords into hashed strings before storing them in the database
- ***Body-parser***: Middleware that parses incoming request bodies in Express applications, making it easier to handle form submissions and JSON data
- ***Cookie-parser***: Middleware that parses cookies attached to client requests, enabling session management and user tracking
- ***Multer***: Middleware designed to handle multipart/form-data, primarily used for file uploads in CloudBin

## Networking Software

- **APIs (Application Programming Interfaces):** APIs are crucial for integrating various services within CloudBin. They enable communication between the frontend and backend components of the application

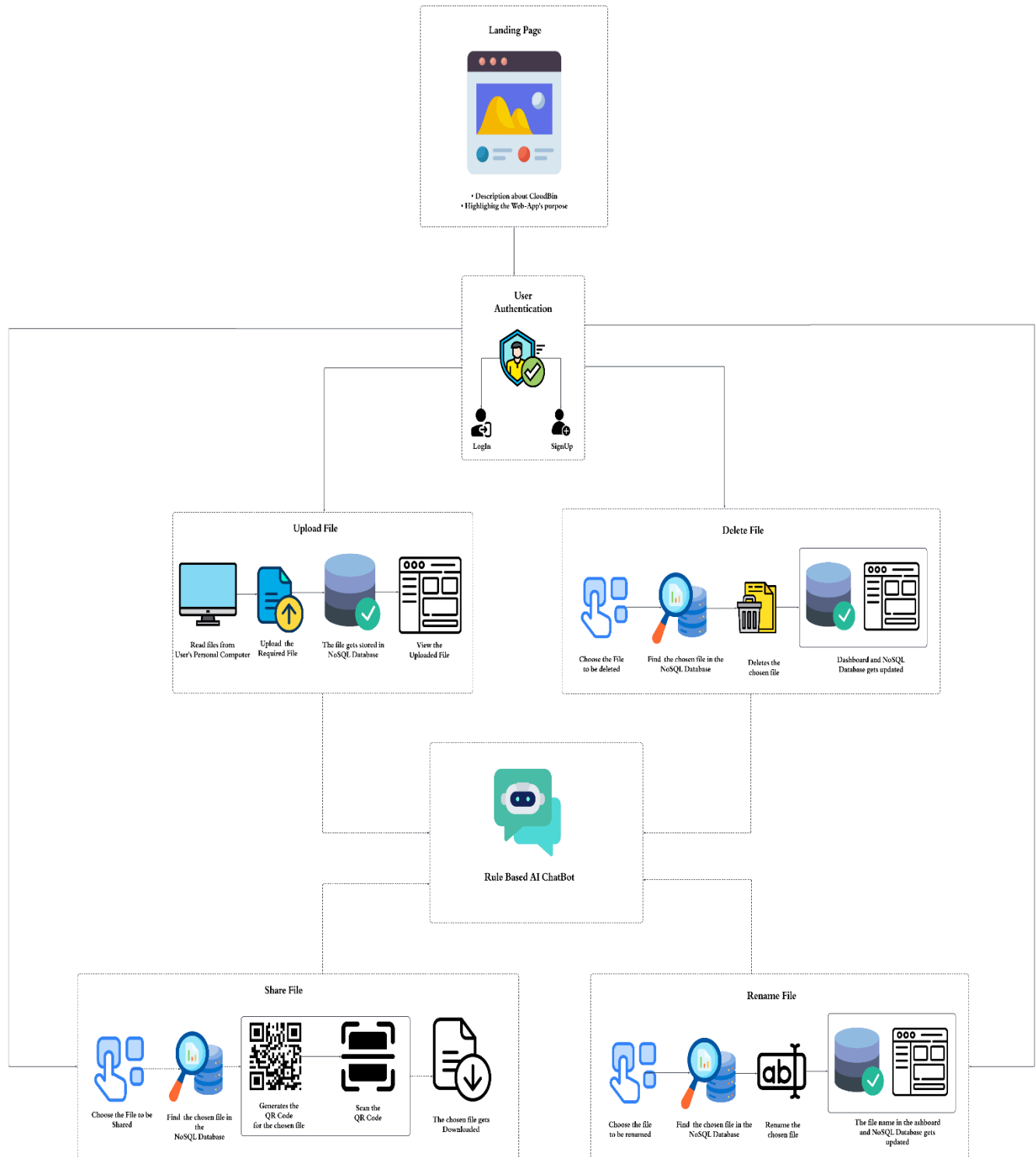
## Security Software

- **Firewalls and Antivirus Software:** Essential for protecting the application from unauthorized access and cyber threats. Regular updates and monitoring are necessary to maintain security

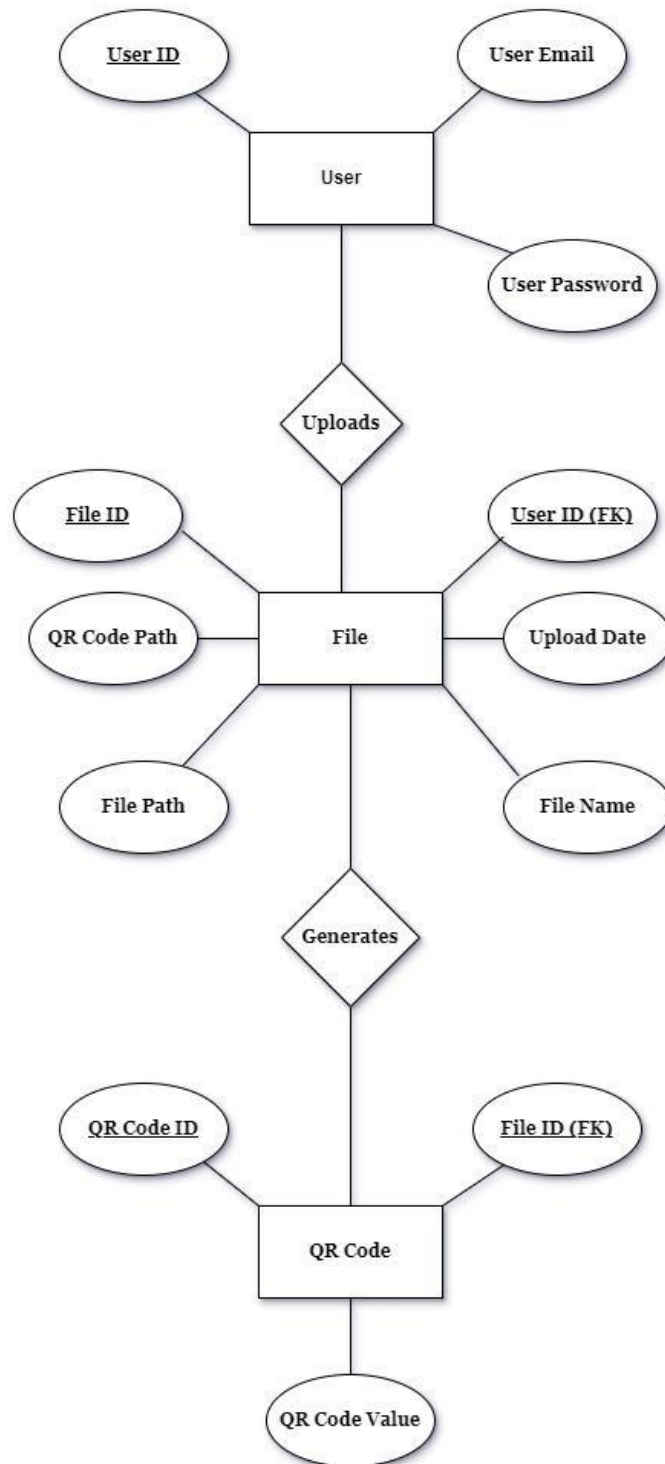
## Collaboration Tools

- **Communication Platforms:** Tools such as Slack or Microsoft Teams may be utilized for team collaboration during development, allowing developers to communicate effectively and share updates on project progress

### 3.3) ARCHITECTURE DIAGRAM:



### 3.4) ENTITY RELATIONSHIP (ER) DIAGRAM:



### 3.4.1. Overview of the Entity-Relationship (ER) Diagram for CloudBin

The Entity-Relationship (ER) diagram for CloudBin serves as a foundational blueprint for understanding the data structure and relationships within the file management and sharing system. This diagram outlines the key entities involved, their attributes, and the relationships that define how these entities interact with one another.

#### Key Entities

##### *1. User*

The **User** entity represents individuals who interact with the CloudBin application. Each user is identified by a unique `user_id`, which serves as the primary key. Additional attributes include:

- **Email:** The user's email address for communication and recovery purposes
- **Password:** A hashed string used for authentication

This entity is crucial as it forms the basis for user authentication and access control within the application.

##### *2. File*

The **File** entity encapsulates all uploaded files within CloudBin. Each file is identified by a unique `file_id`, which acts as its primary key. Key attributes include:

- **Filename:** The name of the uploaded file.
- **Upload Date:** The timestamp indicating when the file was uploaded.
- **User ID:** A foreign key linking back to the User entity, indicating which user uploaded the file.

This entity is vital for managing user-generated content and ensuring that files are associated with their respective owners.

### 3. *QRCode*

The **QRCode** entity represents generated QR codes associated with files in CloudBin. Each QR code is identified by a unique `qr_code_id`, serving as its primary key. Important attributes include:

- **File ID:** A foreign key linking to the File entity, indicating which file the QR code corresponds to.
- **QR Code Value:** The actual data encoded within the QR code, which may include a URL or other relevant information.

This entity enhances user experience by facilitating quick access to files through scannable codes, promoting efficient sharing and collaboration.

### 4. *Relationships*

The relationships depicted in the ER diagram illustrate how these entities interact:

- **User to File Relationship:** This is a one-to-many relationship, where one user can upload multiple files. This relationship is represented by the foreign key `user_id` in the File entity, establishing ownership of each file.
- **File to QRCode Relationship:** This is also a one-to-many relationship, where one file can generate multiple QR codes. This allows for dynamic sharing options, enabling users to create different QR codes for various purposes or versions of a file.

#### 4) **PROGRAM CODE**

- **server.js:**

```
const express = require('express');

const mongoose = require('mongoose');

const bcrypt = require('bcryptjs');

const bodyParser = require('body-parser');

const cookieParser = require('cookie-parser');

const multer = require('multer');

const path = require('path');

const fs = require('fs');

const QRCode = require('qrcode');

const app = express();

const PORT = 3000;

// Middleware

app.use(bodyParser.json());

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

app.use(cookieParser());

app.use(express.static(path.join(__dirname, 'public'))); // Serve static files
from public directory

app.use('/uploads', express.static(path.join(__dirname, 'uploads'))); //
Serve static files from uploads directory
```



```

// Ensure the uploads directory exists

const uploadDir = path.join(__dirname, 'uploads');

if (!fs.existsSync(uploadDir)) fs.mkdirSync(uploadDir);


// Configure multer for file uploads

const storage = multer.diskStorage({

  destination: (req, file, cb) => {

    cb(null, uploadDir);

  },

  filename: (req, file, cb) => {

    cb(null, `${Date.now()}-${file.originalname}`);

  },

});

const upload = multer({ storage });


// Connect to MongoDB

mongoose.connect('mongodb://127.0.0.1:27017/loginSystem', {

  useNewUrlParser: true,

  useUnifiedTopology: true,

})

.then(() => console.log('MongoDB Connected'))

.catch(err => {

```

```

        console.error('MongoDB connection error:', err);

        process.exit(1); // Exit process on failure
    });

    // User Schema

    const userSchema = new mongoose.Schema({

        email: { type: String, required: true, unique: true },

        password: { type: String, required: true },

    });

    const User = mongoose.model('User', userSchema);

    // File Schema

    const fileSchema = new mongoose.Schema({

        userId: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required:
true },

        filename: String,

        filepath: String,

        qrCodePath: String,

        uploadDate: { type: Date, default: Date.now },

    });

    const File = mongoose.model('File', fileSchema);

```

```

// Routes

// Upload Route

app.post('/upload', upload.single('file'), async (req, res) => {

  const { userId } = req.body;

  if (!req.file) return res.status(400).send('No file uploaded.');
```

```

  try {

    const filePath = `/uploads/${req.file.filename}`;

    const qrCodePath = path.join(uploadDir,
`${req.file.filename}.png`); // Path for QR code image

    // Generate QR code for the uploaded file URL

    await QRCode.toFile(qrCodePath,
`http://localhost:${PORT}${filePath}`); // Adjust for production URL

    const newFile = new File({

      userId,

      filename: req.file.originalname,

      filepath: filePath,

      qrCodePath: `/uploads/${req.file.filename}.png`, // Store relative
path of QR code

    });

```

```

        await newFile.save();

        res.send(`File uploaded successfully! <br> `); // Send back QR code
image

    } catch (error) {

        console.error('Error uploading file:', error);

        res.status(500).send('Error uploading file');

    }

});

// Signup Route

app.post('/signup', async (req, res) => {

    const { email, password } = req.body;

    try {

        const existingUser = await User.findOne({ email });

        if (existingUser) {

            return res.status(400).send('Email already in use.');
```

```

        }

        const hashedPassword = await bcrypt.hash(password, 10);

        const newUser = new User({ email, password: hashedPassword });

        await newUser.save();
    }
});

```

```

        res.redirect('/'); // Redirect to login after successful signup
    } catch (error) {

        console.error('Error creating user:', error);

        res.status(500).send('Error creating user');

    }

});

// Login Route

app.post('/login', async (req, res) => {

    const { email, password } = req.body;

    try {

        const user = await User.findOne({ email });

        if (!user) return res.status(400).send('Invalid email or password');

        const isMatch = await bcrypt.compare(password, user.password);

        if (!isMatch) return res.status(400).send('Invalid email or
password');

        res.redirect(`/upload.html?userId=${user._id}`); // Redirect to
upload page with user ID

    } catch (error) {

        console.error('Error during login:', error);

        res.status(500).send('Error during login');
    }
});

```

```

    }
  });

// File Upload Route

app.post('/upload', upload.single('file'), async (req, res) => {

  const { userId } = req.body;

  if (!req.file) return res.status(400).send('No file uploaded.');
```

```

  try {

    const newFile = new File({

      userId,

      filename: req.file.originalname,

      filepath: `/uploads/${req.file.filename}`,

    });

    await newFile.save();

    res.send('File uploaded successfully!');

  } catch (error) {

    console.error('Error uploading file:', error);

    res.status(500).send('Error uploading file');

  }

});

// Function to delete a file

function deleteFile(fileId) {
```

```

    fetch(`/files/${fileId}`, {
      method: 'DELETE',
    })

    .then(response => {
      if (!response.ok) throw new Error('Network response was not ok');
      return response.text();
    })

    .then(message => {
      alert(message); // Show success message
      location.reload(); // Reload the page to refresh the file list
    })

    .catch(error => console.error('Error deleting file:', error));
  }

  // Delete File Route
  app.delete('/files/:id', async (req, res) => {
    const { id } = req.params;

    try {
      // Find the file in the database
      const file = await File.findById(id);

      if (!file) return res.status(404).send('File not found');

      // Delete the file from the filesystem

```

```

        const filePath = path.join(__dirname, 'uploads',
path.basename(file.filepath));

        fs.unlink(filePath, async (err) => {

            if (err) {

                console.error('Error deleting file:', err);

                return res.status(500).send('Error deleting file');

            }

            // Remove the file record from the database

            await File.deleteOne({ _id: id });

            res.send('File deleted successfully');

        });

    } catch (error) {

        console.error('Error deleting file:', error);

        res.status(500).send('Error deleting file');

    }

});

// Retrieve User Files

app.get('/files', async (req, res) => {

    const { userId } = req.query;

    try {

        const files = await File.find({ userId });

        res.json(files);

    }

```



```

    } catch (error) {

        console.error('Error fetching files:', error);

        res.status(500).send('Error fetching files');

    }

});

// Serve HTML Files

app.get('/', (req, res) => {

    res.sendFile(path.join(__dirname, 'public', 'index.html'));

});

// Serve upload.html directly when requested

app.get('/upload.html', (req, res) => {

    res.sendFile(path.join(__dirname, 'public', 'upload.html'));

});

// Start Server

app.listen(PORT, () => console.log(`Server running on
http://localhost:${PORT}`));

```

### **first.html**

```

<!DOCTYPE html>

<html>

```

```
<head>

<!-- Basic -->

<meta charset="utf-8" />

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<!-- Mobile Metas -->

<meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no" />

<!-- Site Metas -->

<meta name="keywords" content="" />

<meta name="description" content="" />

<meta name="author" content="" />

<link rel="shortcut icon" href="s1.png" type="">


<title>CloudBin.</title>


<!-- bootstrap core css -->

<link rel="stylesheet" type="text/css" href="css/bootstrap.css" />


<!-- fonts style -->

<link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500;
700;900&display=swap" rel="stylesheet">
```

```
<!--owl slider stylesheet -->

<link rel="stylesheet" type="text/css"
href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.3.4/assets/o
wl.carousel.min.css" />


<!-- font awesome style -->

<link href="css/font-awesome.min.css" rel="stylesheet" />


<!-- Custom styles for this template -->

<link href="css/style.css" rel="stylesheet" />

<!-- responsive style -->

<link href="css/responsive.css" rel="stylesheet" />


</head>


<body>


<div class="hero_area">


<div class="hero_bg_box">

<div class="bg_img_box">



</div>

</div>
```

</div>

<!-- header section strats -->

<header class="header\_section">

<div class="container-fluid">

<nav class="navbar navbar-expand-lg custom\_nav-container ">

<a class="navbar-brand" href="#">

<span>

CloudBin.

</span>

</a>

<button class="navbar-toggler" type="button" data-  
toggle="collapse" data-target="#navbarSupportedContent" aria-  
controls="navbarSupportedContent" aria-expanded="false" aria-  
label="Toggle navigation">

<span class=""> </span>

</button>

<div class="collapse navbar-collapse"  
id="navbarSupportedContent">

<ul class="navbar-nav ">

<li class="nav-item active">

```

        <a class="nav-link" href="index.html">Home <span class="sr-
only">(current)</span></a>

    </li>

    <li class="nav-item">

        <a class="nav-link" href="index.html"> <i class="fa fa-user"
aria-hidden="true"></i> Login</a>

    </li>

</ul>

</div>

</nav>

</div>

</header>

<!-- end header section -->

<!-- slider section -->

<section class="slider_section ">

    <div id="customCarousel1" class="carousel slide" data-
ride="carousel">

        <div class="carousel-inner">

            <div class="carousel-item active">

                <div class="container ">

                    <div class="row">

                        <div class="col-md-6 ">

                            <div class="detail-box">

```

<h1>

CLOUDBin

</h1>

<p>

CloudBin is "made in India" storage platform, that allows users to upload, manage,

and share their files effortlessly while benefiting from low-latency access. With features like unique QR code sharing.

</p>

<div class="btn-box">

</div>

</div>

</div>

<div class="col-md-6">

<div class="img-box">



</div>

</div>

</div>

</div>

</div>

</div>

</div>

```

</section>

<!-- end slider section -->

</div>


<!-- service section -->

<section class="service_section layout_padding">

  <div class="service_container">

    <div class="container ">

      <div class="heading_container heading_center">

        <h2>

          Our <span>Services</span>

        </h2>

      </div>

      <div class="row">

        <div class="col-md-4 ">

          <div class="box ">

            <div class="img-box">

            </div>

            <div class="detail-box">

```

<h5>

QR Code Sharing

</h5>

<p>

Generate unique QR codes for each uploaded file, allowing you to share access quickly and efficiently with anyone, anywhere.

Recipients can scan the code to download or view the file instantly.

</p>

</div>

</div>

</div>

<div class="col-md-4 ">

<div class="box ">

<div class="img-box">



</div>

<div class="detail-box">

<h5>

Security Features

</h5>

<p>



Enjoy peace of mind with our advanced security measures that protect your files during upload and while stored in the cloud.

We prioritize your privacy and data integrity.

</p>

</div>

</div>

</div>

<div class="col-md-4 ">

<div class="box ">

<div class="img-box">



</div>

<div class="detail-box">

<h5>

User Support

</h5>

<p>

Our dedicated support team is available to assist you with any questions or issues you may encounter while using CloudBin.

We are committed to providing excellent customer service.

</p>

</div>

</div>

```

        </div>

    </div>

    <div class="btn-box">

    </div>

</div>

</div>

</section>

<!-- end service section -->

<!-- about section -->

<section class="about_section layout_padding">

    <div class="container ">

        <div class="heading_container heading_center">

            <h2>

                About <span>Us</span>

            </h2>

            <h3>

                We Are CloudBin.

            </h3>

            <p>

```

CloudBin is an innovative platform designed to simplify file management in the digital age. Our mission is to empower users by providing a seamless experience for uploading, sharing and managing their files online. With CloudBin, you can easily upload various types of files, from documents to images and share them effortlessly through unique QR codes. This not only enhances accessibility but also ensures that your files are securely stored in the cloud.

</p>

<p>

At CloudBin, we prioritize user control and privacy. Users have the flexibility to delete their files whenever they choose, ensuring that they maintain full ownership and management of their data. Our user-friendly interface and robust security measures make CloudBin the ideal solution for individuals and businesses alike who seek a reliable file-sharing platform.

</p>

</div>

</section>

<!-- end about section -->

<!-- team section -->

<section class="team\_section layout\_padding">

<div class="container-fluid">

<div class="heading\_container heading\_center">

<h2 class="">

Our <span> Team</span>

</h2>

</div>

<div class="team\_container">

<div class="row">

<div class="col-lg-3 col-sm-6">

<div class="box ">

<div class="img-box">



</div>

<div class="detail-box">

<h5>

Giridhar

</h5>

<p>

Project Head /

Fullstack Developer

</p>

</div>

<div class="social\_box">

<a href="https://www.linkedin.com/in/giridhar-uppili-966358192/">

```

        <i class="fa fa-linkedin" aria-hidden="true"></i>

    </a>

    <a href="https://www.instagram.com/_giridharphotography/">

        <i class="fa fa-instagram" aria-hidden="true"></i>

    </a>

</div>

</div>

</div>

<div class="col-lg-3 col-sm-6">

    <div class="box ">

        <div class="img-box">

        </div>

        <div class="detail-box">

            <h5>

                Ezhil Adhithya

            </h5>

            <p>

                Backend engineer

            </p>

        </div>

        <div class="social_box">

            <a href="https://www.linkedin.com/in/ezhiladhithya/">

```

```

        <i class="fa fa-linkedin" aria-hidden="true"></i>

    </a>

    <a href="https://www.instagram.com/_hakkanoodles_">

        <i class="fa fa-instagram" aria-hidden="true"></i>

    </a>

</div>

</div>

</div>

<div class="col-lg-3 col-sm-6">

    <div class="box ">

        <div class="img-box">

        </div>

        <div class="detail-box">

            <h5>

                Harish

            </h5>

            <p>

                Database Administrator

            </p>

        </div>

        <div class="social_box">

            <a href="https://www.linkedin.com/in/harish-t-7b4428316/">

```

```

        <i class="fa fa-linkedin" aria-hidden="true"></i>

    </a>

    <a href="https://www.instagram.com/harixh.__.2510/">

        <i class="fa fa-instagram" aria-hidden="true"></i>

    </a>

</div>

</div>

</div>

<div class="col-lg-3 col-sm-6">

    <div class="box ">

        <div class="img-box">

        </div>

        <div class="detail-box">

            <h5>

                Aravind

            </h5>

            <p>

                Frontend Developer

            </p>

        </div>

        <div class="social_box">

            <a href="https://www.linkedin.com/in/aravind-s-2659102a2/">

```

```
<i class="fa fa-linkedin" aria-hidden="true"></i>

</a>

<a href="https://www.instagram.com/aravinds106">

  <i class="fa fa-instagram" aria-hidden="true"></i>

</a>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</section>

<!-- end team section -->


<!-- footer section -->

<section class="footer_section">

  <div class="container">

    <p>

      &copy; <span id="displayYear"></span> All Rights Reserved By

Cloudbin Pvt Ltd.

    </p>

  </div>

</section>
```



```
<!-- footer section -->
```

```
</body>
```

```
</html>
```

- **index.html**

```
<!DOCTYPE html>
```

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

```
<head>
```

```
<meta charset="UTF-8" />
```

```
<meta name="viewport" content="width=device-width, initial-  
scale=1.0" />
```

```
<link rel="stylesheet" href="styles.css" />
```

```
<title>Cloud Bin</title>
```

```
</head>
```

```
<body>
```

```
<div class="loading-screen">
```

```
<div class="loader"></div>
```

```
<div class="loading-text">CloudBin.</div>
```

```
</div>
```

```
<div class="wrapper">
```

```
<div class="title-text">
```

```
<div class="title login">Login Form</div>
```

```
<div class="title signup">Signup Form</div>

</div>
```

```
<div class="form-container">

  <div class="slide-controls">

    <input type="radio" name="slide" id="login" checked />

    <input type="radio" name="slide" id="signup" />

    <label for="login" class="slide login">Login</label>

    <label for="signup" class="slide signup">Signup</label>

    <div class="slider-tab"></div>

  </div>
```

```
<div class="form-inner">

  <!-- Login Form -->

  <form action="/login" method="POST" class="login">

    <div class="field">

      <input type="text" name="email" placeholder="Email Address"
required />

    </div>

    <div class="field">

      <input type="password" name="password"
placeholder="Password" required />

    </div>
```

```

<div class="pass-link"><a href="#">Forgot password?</a></div>

<div class="field btn">

  <div class="btn-layer"></div>

  <input type="submit" value="Login" />

</div>

<div class="signup-link">Not a member? <a href="">Signup
now</a></div>

</form>

<!-- Signup Form -->

<form action="/signup" method="POST" class="signup">

  <div class="field">

    <input type="text" name="email" placeholder="Email Address"
required />

  </div>

  <div class="field">

    <input type="password" name="password"
placeholder="Password" required />

  </div>

  <div class="field">

    <input

      type="password"

      name="confirmPassword"

```

```

        placeholder="Confirm password"

        required

    />

</div>

<div class="field btn">

    <div class="btn-layer"></div>

    <input type="submit" value="Signup" />

</div>

</form>

</div>

</div>

</div>

<script src="script.js"></script>

<script>

    // Hide loading screen after 3 seconds

    setTimeout(() => {

        document.querySelector('.loading-screen').style.display = 'none';

    }, 1000); // Adjust time as needed

</script>

</body>

</html>

```

- **style.css**

```
{  
    margin: 0;  
    padding: 0;  
    box-sizing: border-box;  
    font-family: 'Poppins', sans-serif;  
}  
  
html,body{  
    display: grid;  
    height: 100%;  
    width: 100%;  
    place-items: center;  
    background: -webkit-linear-gradient(left, #003366,#004080,#0059b3  
, #0073e6);  
}  
  
::selection{  
    background: #1a75ff;  
    color: #fff;  
}  
  
.loading-screen {  
    position: fixed;  
    top: 0;
```

```

left: 0;

width: 100%;

height: 100%;

background-color: rgba(0, 0, 0, 0.8); /* Semi-transparent background
*/

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

z-index: 999; /* Ensure it covers everything */

}

```

```

.loading-text {

font-size: 48px; /* Adjust size as needed */

font-weight: bold;

color: #fff; /* Text color */

margin-top: 20px; /* Space between loader and text */

}

```

```

/* Loader styles */

.loader {

border: 16px solid #f3f3f3; /* Light grey */

border-top: 16px solid #003366; /* Blue */

```

```

border-radius: 50%;

width: 120px; /* Size of the loader */

height: 120px; /* Size of the loader */

animation: spin 2s linear infinite; /* Animation for spinning effect */

}

/* Keyframes for spin animation */

@keyframes spin {

    0% { transform: rotate(0deg); }

    100% { transform: rotate(360deg); }

}

.wrapper{

    overflow: hidden;

    max-width: 390px;

    background: #fff;

    padding: 30px;

    border-radius: 15px;

    box-shadow: 0px 15px 20px rgba(0,0,0,0.1);

}

.wrapper .title-text{

    display: flex;

    width: 200%;

}

```

```
.wrapper .title{  
  
    width: 50%;  
  
    font-size: 35px;  
  
    font-weight: 600;  
  
    text-align: center;  
  
    transition: all 0.6s cubic-bezier(0.68,-0.55,0.265,1.55);  
  
}  
  
.wrapper .slide-controls{  
  
    position: relative;  
  
    display: flex;  
  
    height: 50px;  
  
    width: 100%;  
  
    overflow: hidden;  
  
    margin: 30px 0 10px 0;  
  
    justify-content: space-between;  
  
    border: 1px solid lightgrey;  
  
    border-radius: 15px;  
  
}  
  
.slide-controls .slide{  
  
    height: 100%;  
  
    width: 100%;  
  
    color: #fff;  
  
    font-size: 18px;
```



```

font-weight: 500;

text-align: center;

line-height: 48px;

cursor: pointer;

z-index: 1;

transition: all 0.6s ease;
}

.slide-controls label.signup{

color: #000;

}

.slide-controls .slider-tab{

position: absolute;

height: 100%;

width: 50%;

left: 0;

z-index: 0;

border-radius: 15px;

background: -webkit-linear-gradient(left,#003366,#004080,#0059b3
, #0073e6);

transition: all 0.6s cubic-bezier(0.68,-0.55,0.265,1.55);
}

input[type="radio"]{

display: none;

```

```

}

#signup:checked ~ .slider-tab{

    left: 50%;

}

#signup:checked ~ label.signup{

    color: #fff;

    cursor: default;

    user-select: none;

}

#signup:checked ~ label.login{

    color: #000;

}

#login:checked ~ label.signup{

    color: #000;

}

#login:checked ~ label.login{

    cursor: default;

    user-select: none;

}

.wrapper .form-container{

    width: 100%;

    overflow: hidden;

}

```

```
.form-container .form-inner{  
  
    display: flex;  
  
    width: 200%;  
  
}  
  
.form-container .form-inner form{  
  
    width: 50%;  
  
    transition: all 0.6s cubic-bezier(0.68,-0.55,0.265,1.55);  
  
}  
  
.form-inner form .field{  
  
    height: 50px;  
  
    width: 100%;  
  
    margin-top: 20px;  
  
}  
  
.form-inner form .field input{  
  
    height: 100%;  
  
    width: 100%;  
  
    outline: none;  
  
    padding-left: 15px;  
  
    border-radius: 15px;  
  
    border: 1px solid lightgrey;  
  
    border-bottom-width: 2px;  
  
    font-size: 17px;  
  
    transition: all 0.3s ease;
```

```

}

.form-inner form .field input:focus{

    border-color: #1a75ff;

    /* box-shadow: inset 0 0 3px #fb6aae; */

}

.form-inner form .field input::placeholder{

    color: #999;

    transition: all 0.3s ease;

}

form .field input:focus::placeholder{

    color: #1a75ff;

}

.form-inner form .pass-link{

    margin-top: 5px;

}

.form-inner form .signup-link{

    text-align: center;

    margin-top: 30px;

}

.form-inner form .pass-link a,

.form-inner form .signup-link a{

    color: #1a75ff;

    text-decoration: none;

```

```

}

.form-inner form .pass-link a:hover,

.form-inner form .signup-link a:hover{

    text-decoration: underline;

}

form .btn{

    height: 50px;

    width: 100%;

    border-radius: 15px;

    position: relative;

    overflow: hidden;

}

form .btn .btn-layer{

    height: 100%;

    width: 300%;

    position: absolute;

    left: -100%;

    background: -webkit-linear-gradient(right,#003366,#004080,#0059b3

, #0073e6);

    border-radius: 15px;

    transition: all 0.4s ease;;

}

form .btn:hover .btn-layer{

```

```
    left: 0;

}

form .btn input[type="submit"]{

    height: 100%;

    width: 100%;

    z-index: 1;

    position: relative;

    background: none;

    border: none;

    color: #fff;

    padding-left: 0;

    border-radius: 15px;

    font-size: 20px;

    font-weight: 500;

    cursor: pointer;

}
```

- **upload.html**

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
```

```

<title>Cloud Bin</title>

<link rel="stylesheet" href="styleless.css" />

</head>

<body>

  <div class="container">

    <header>

      <h1>CloudBin.</h1>

      <button id="logout-button">Logout</button>

    </header>

    <form id="upload-form" action="/upload" method="POST"
  enctype="multipart/form-data">

      <h2>Upload File</h2>

      <input type="file" name="file" required />

      <input type="hidden" name="userId" id="userId" />

      <button type="submit">Upload</button>

    </form>

    <div id="file-list">

      <h3>Your Files</h3>

      <div id="file-display"></div>

    </div>

  </div>

```

```

<!-- Chatbot Widget -->

<div id="chatbot">

    <div class="chat-header">Chat with us!</div>

    <div class="chat-body">

        <div id="chat-messages"></div>

        <input type="text" id="chat-input" placeholder="Type your
message..." />

    </div>

</div>

<script>

    // Logout functionality

    document.getElementById('logout-button').addEventListener('click',
() => {

        window.location.href = 'first.html';

    });

    // Extract userId from the query string

    const userId = new
URLSearchParams(window.location.search).get('userId');

    document.getElementById('userId').value = userId;

```



```

// Fetch and display files for the logged-in user

fetch(`/files?userId=${userId}`)

  .then(response => {

    if (!response.ok) throw new Error('Failed to fetch files');

    return response.json();

  })

  .then(files => {

    const fileDisplay = document.getElementById('file-display');

    fileDisplay.innerHTML = "";

    if (files.length === 0) {

      fileDisplay.innerHTML = '<p>No files uploaded yet.</p>';

    } else {

      files.forEach(file => {

        const card = document.createElement('div');

        card.className = 'file-card';

        const link = document.createElement('a');

        link.href = file.filepath;

        link.textContent = file.filename;

        link.target = '_blank';

        const qrButton = document.createElement('button');

```

```

qrButton.textContent = 'QR';

qrButton.onclick = () => window.open(file.qrCodePath,
'_blank'); // Open QR code image

```

```

const deleteButton = document.createElement('button');

deleteButton.textContent = 'Delete';

deleteButton.onclick = () => deleteFile(file._id);

card.appendChild(link);

card.appendChild(qrButton); // Add QR button

card.appendChild(deleteButton);

fileDisplay.appendChild(card);

});

}

})

.catch(error => {

    alert('Error fetching files. Please try again later.');
```

```

    console.error('Error fetching files:', error);

```

```

});

```

```

// Function to delete a file

```

```

function deleteFile(fileId) {

    fetch(`/files/${fileId}`, {

```

```

        method: 'DELETE',
    })

    .then(response => {

        if (!response.ok) throw new Error('Network response was not
ok');

        return response.text();

    })

    .then(message => {

        alert(message);

        location.reload();

    })

    .catch(error => console.error('Error deleting file:', error));

}

```

```

// Chatbot functionality

```

```

let faqData = [];

```

```

// Fetch FAQ data from JSON file

```

```

fetch('faq.json')

    .then(response => response.json())

    .then(data => { faqData = data; })

    .catch(error => console.error('Error loading FAQ data:', error));

```

```

const chatMessages = document.getElementById('chat-messages');

const chatInput = document.getElementById('chat-input');

chatInput.addEventListener('keypress', function(event) {

    if (event.key === 'Enter') {

        const userMessage = chatInput.value.trim();

        if (userMessage) {

            addMessage('You: ' + userMessage);

            chatInput.value = "";

            // Find matching FAQ answer by keyword

            const botResponse = getBotResponse(userMessage);

            addMessage('Bot: ' + botResponse);

        }

    }

});

function getBotResponse(userMessage) {

    const lowerCaseMessage = userMessage.toLowerCase();

    // Check for keywords in each FAQ entry

    for (const entry of faqData) {

```

```

        if (entry.keywords.some(keyword =>
lowerCaseMessage.includes(keyword))) {

            return entry.answer;

        }

    }

    return "I'm sorry, I don't understand that. Try asking something
else!";

}

function addMessage(message) {

    const messageDiv = document.createElement('div');

    messageDiv.textContent = message;

    chatMessages.appendChild(messageDiv);

    chatMessages.scrollTop = chatMessages.scrollHeight; // Auto-
scroll to latest message

}

</script>

<!-- Add styles for chatbot -->

<style>

    /* Your existing styles for chatbot here */

    #chatbot {

```

```
position: fixed;

bottom: 20px;

right: 20px;

width: 300px;

border: 1px solid #ccc;

border-radius: 8px;

background-color: #fff;

box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);

display: flex;

flex-direction: column;

}
```

```
.chat-header {

background-color: #1a73e8;

color: white;

padding: 10px;

text-align: center;

border-top-left-radius: 8px;

border-top-right-radius: 8px;

}
```

```
.chat-body {

padding: 10px;
```

```

        flex-grow: 1; /* Allow body to grow */

        overflow-y: auto; /* Enable scrolling */
    }

    #chat-messages {

        max-height: 200px; /* Limit height */
    }

    #chat-input {

        border: none;

        padding: 10px;

        width: calc(100% - 20px); /* Full width minus padding */
    }
</style>

</body>

</html>

```

- **styleless.css**

```

/* General styles */

html, body {

    display: grid;

    height: 100%;

```

```
width: 100%;

place-items: center;

background: -webkit-linear-gradient(left, #003366, #004080, #0059b3,
    #0073e6);

}
```

```
/* Container styles */
```

```
.container {

    max-width: 900px;

    padding: 20px;

    background: #ffffff; /* White background for container */

    border-radius: 8px;

    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);

}
```

```
/* Header styles */
```

```
header {

    display: flex;

    justify-content: space-between;

    align-items: center;

}
```

```
#logo {
```



```

    max-width: 200px; /* Adjust width as needed */

    height: auto; /* Maintain aspect ratio */

    margin-bottom: 20px; /* Space below logo */

}

/* Heading styles */

h1, h2, h3 {

    color: #202124; /* Darker text for better readability */

}

/* Button styles */

button {

    background-color: #1a73e8; /* Google Drive blue */

    color: #fff;

    border: none;

    border-radius: 4px;

    padding: 10px 15px;

    cursor: pointer;

    font-size: 14px;

    transition: background-color 0.3s ease;

}

button:hover {

```

```

        background-color: #155ab6; /* Darker blue on hover */
    }

    /* Form styles */

    form {

        margin-top: 20px;

    }

    /* File list styles */

    #file-list {

        margin-top: 20px;

    }

    #file-display {

        display: flex;

        flex-wrap: wrap; /* Allow cards to wrap */

    }

    .file-card {

        background-color: #f9f9f9; /* Light card background */

        border-radius: 4px;

        padding: 15px;

        margin: 10px;

```

```

    box-shadow: 0 1px 3px rgba(0, 0, 0, 0.2);

    width: calc(33% - 20px); /* Three cards per row with spacing */
}

.file-card a {

    display: block; /* Makes the link fill the card area */

    color: #1a73e8; /* Link color similar to Google Drive */
}

.file-card button {

    margin-top: 10px; /* Space between link and button */
}

/* Additional styles for QR button */

.file-card button:nth-child(2) { /* Targeting the QR button specifically */

    margin-left: 10px; /* Space between filename and QR button */
}

```

- **script.js**

```

const loginText = document.querySelector(".title-text .login");

const loginForm = document.querySelector("form.login");

const loginBtn = document.querySelector("label.login");

const signupBtn = document.querySelector("label.signup");

```

```
const signupLink = document.querySelector("form .signup-link a");

signupBtn.onclick = ()=>{

    loginForm.style.marginLeft = "-50%";

    loginText.style.marginLeft = "-50%";

});

loginBtn.onclick = ()=>{

    loginForm.style.marginLeft = "0%";

    loginText.style.marginLeft = "0%";

});

signupLink.onclick = ()=>{

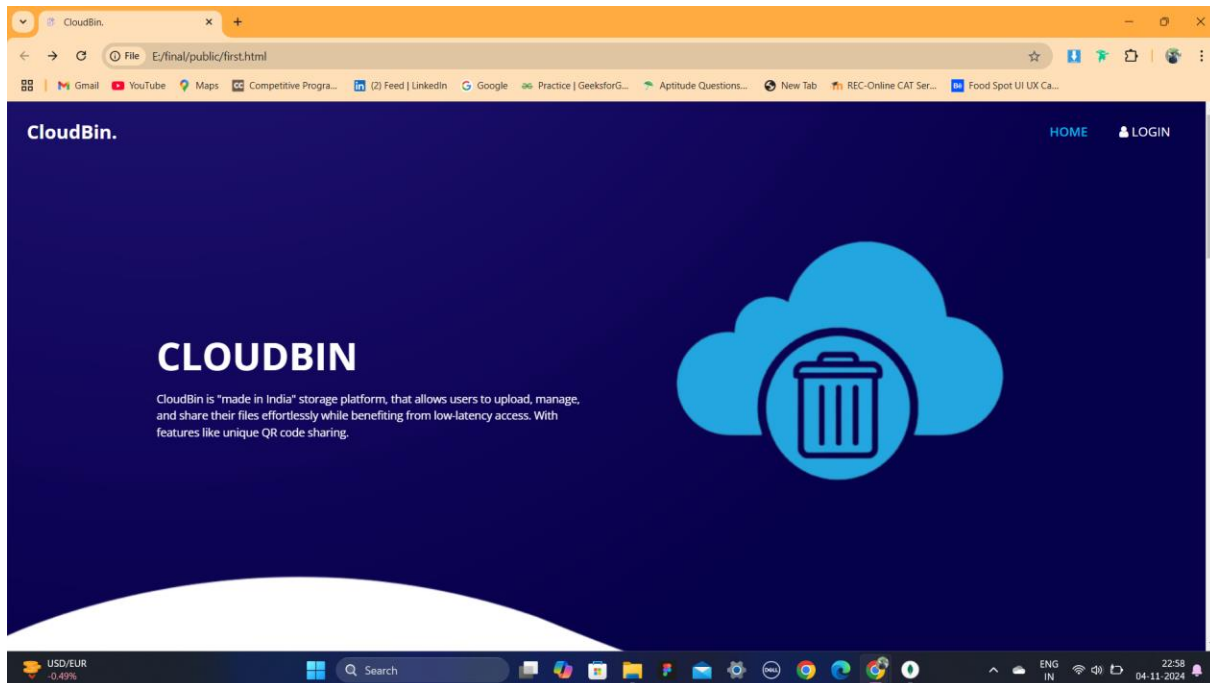
    signupBtn.click();

    return false;

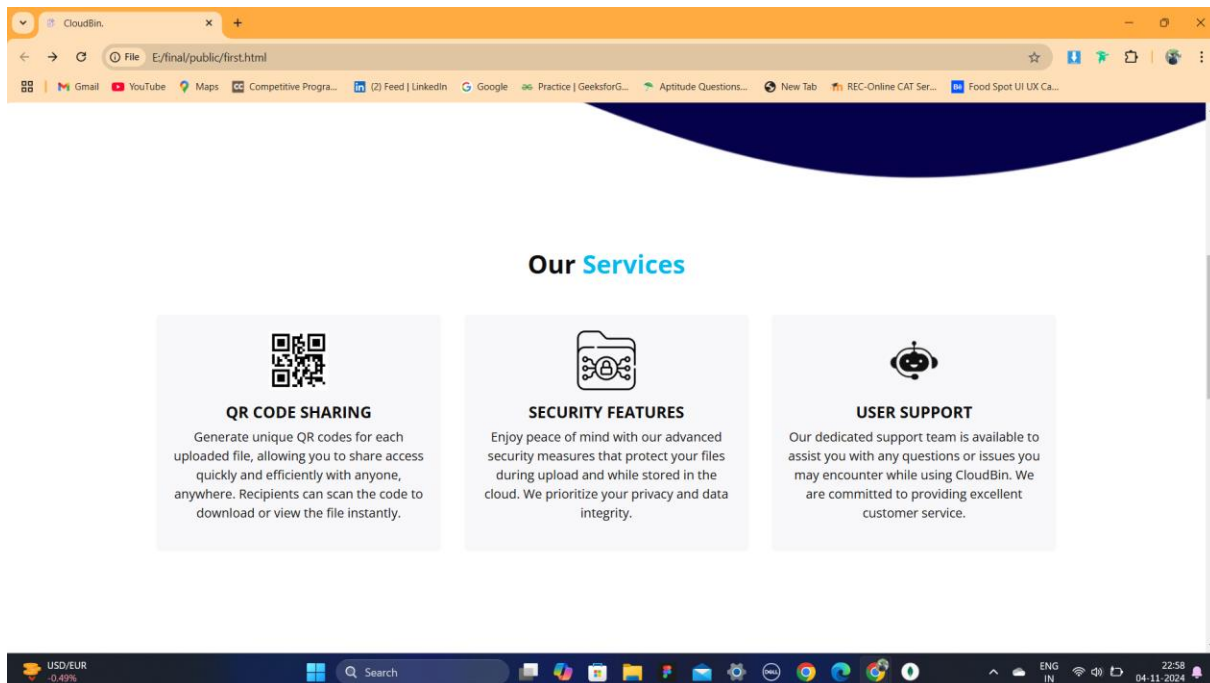
});
```

## 5) OUTPUT

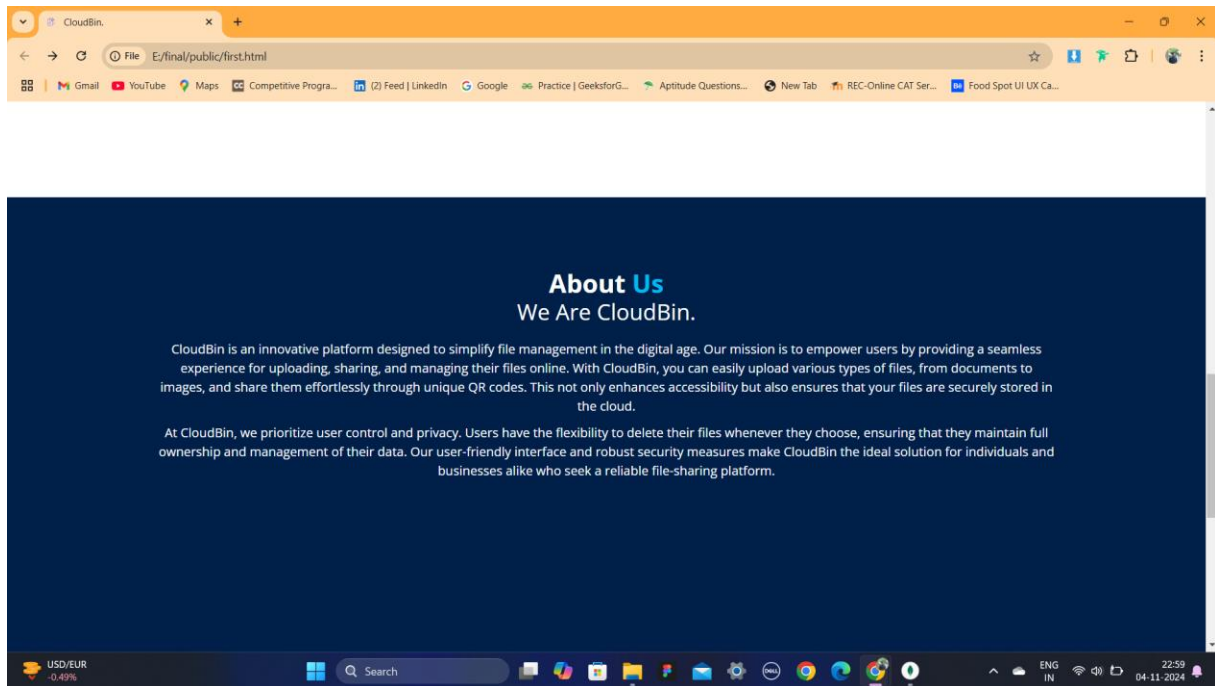
### *HOME PAGE*



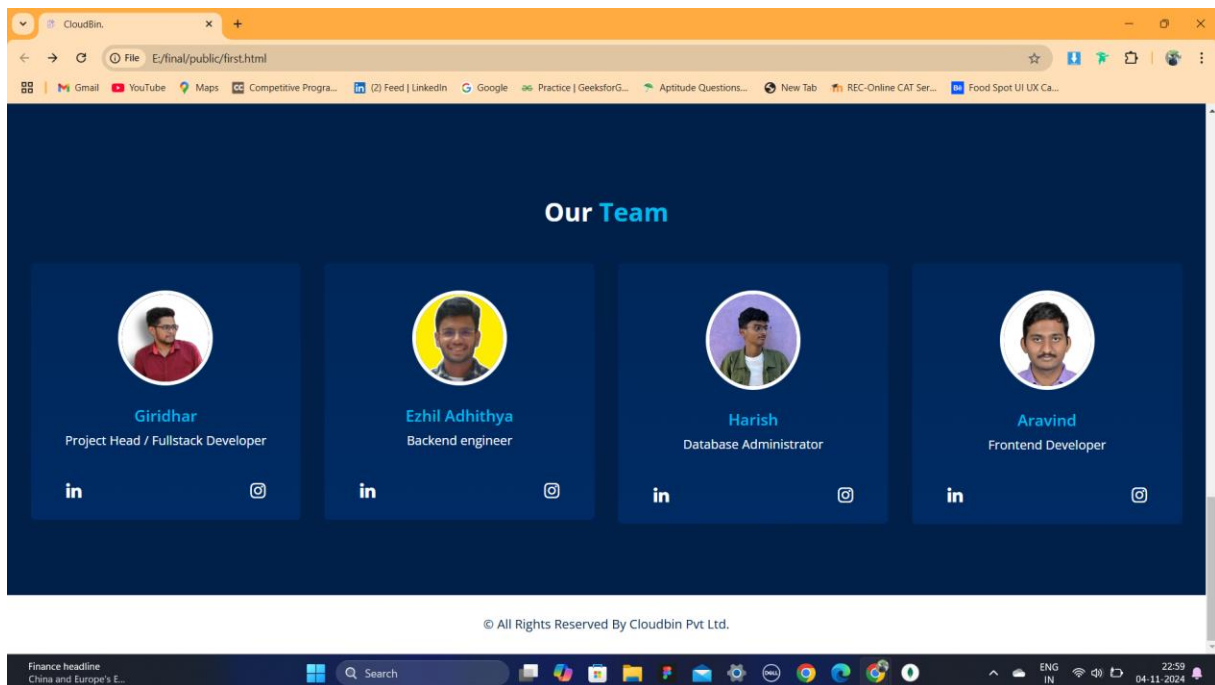
### *HOME PAGE (OUR SERVICES)*



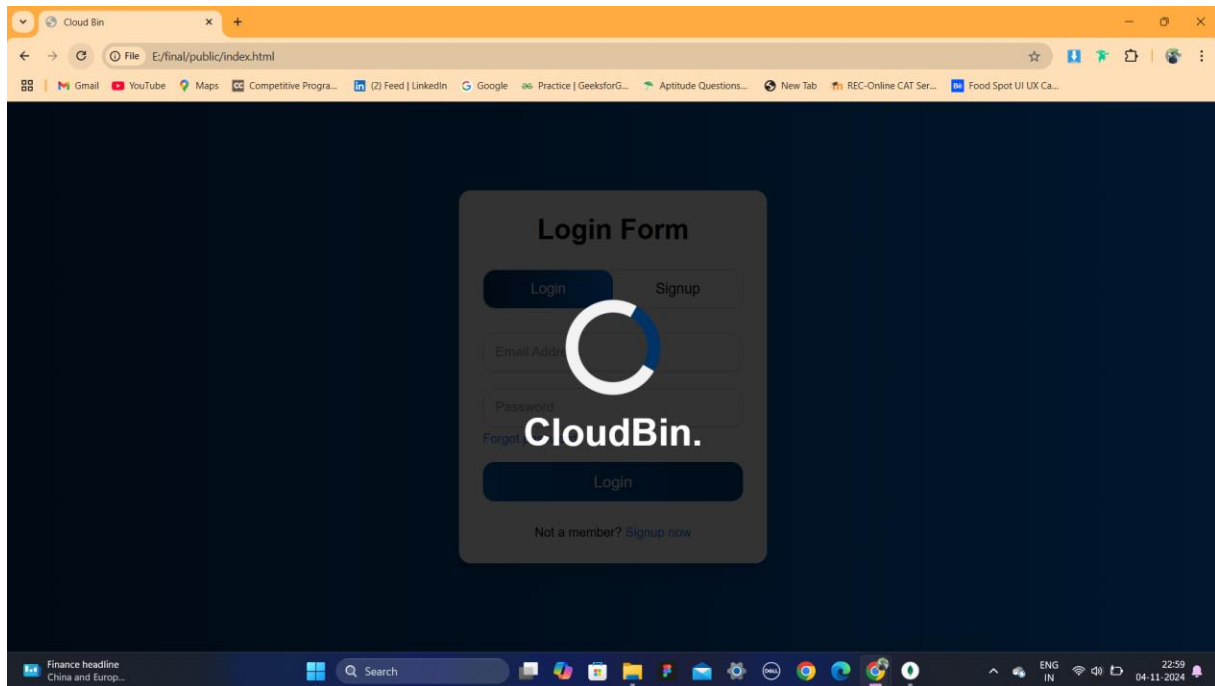
## HOME PAGE (ABOUT US)



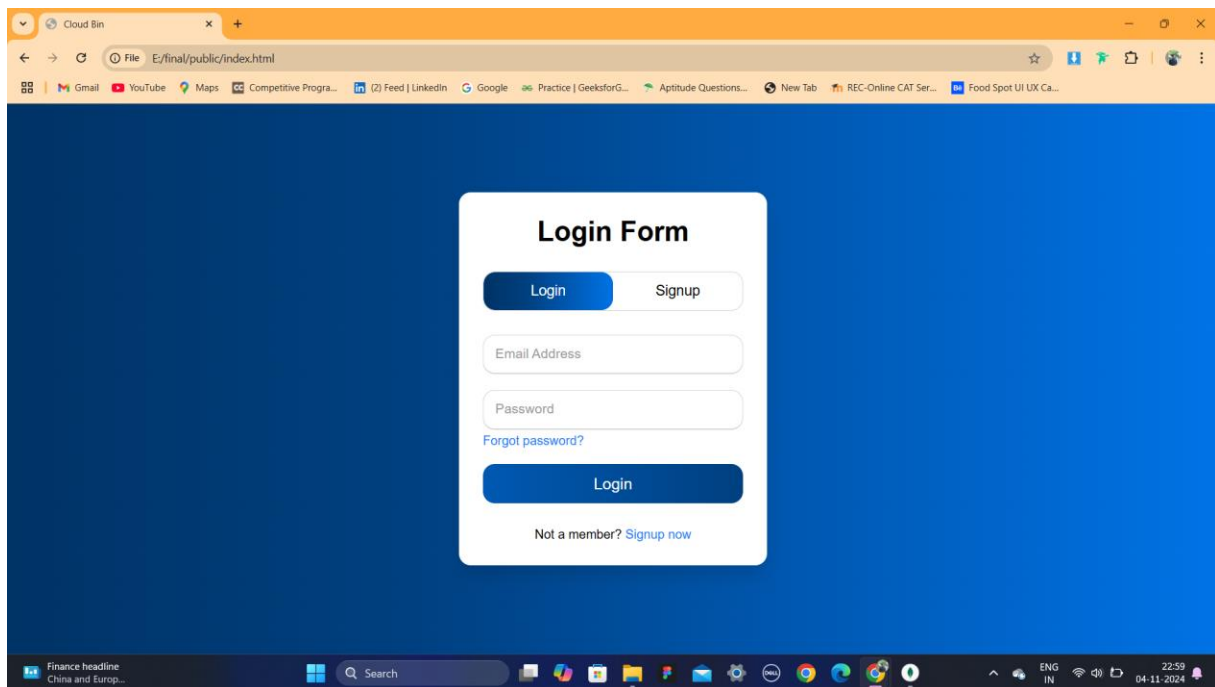
## HOME PAGE (OUR TEAM)



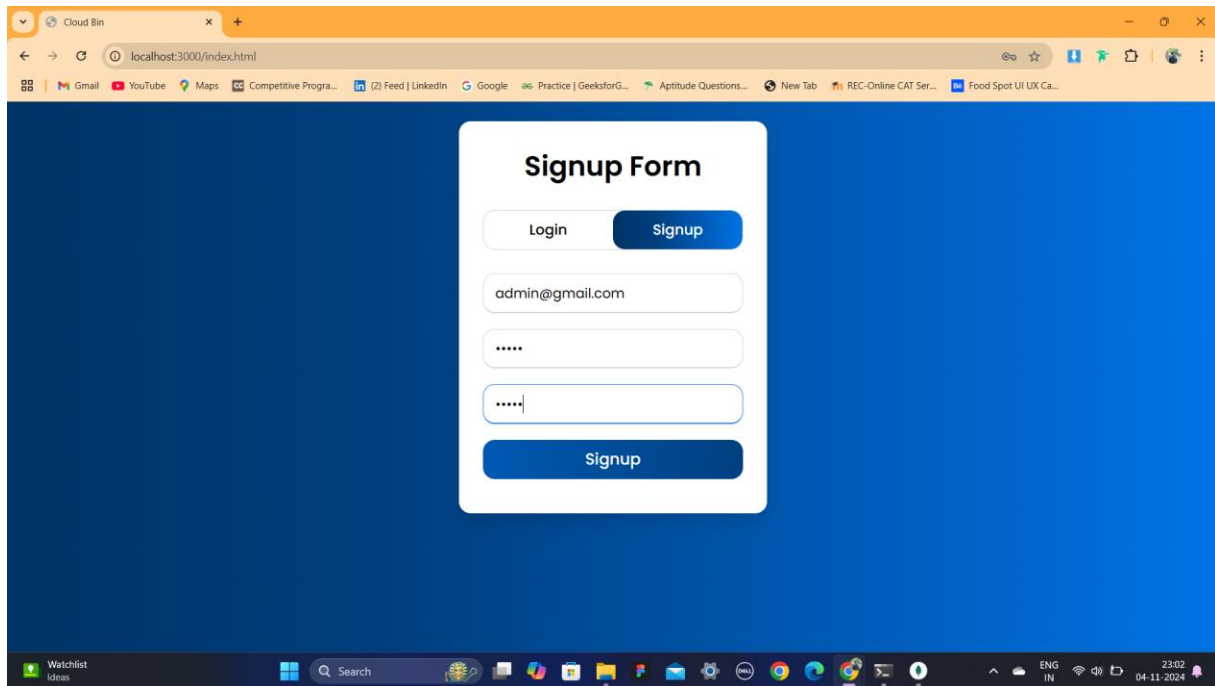
## LOGIN/SIGNUP PAGE



## LOGIN FORM



## ***SIGNUP FORM***



Cloud Bin

localhost:3000/index.html

Signup Form

Login Signup

admin@gmail.com

\*\*\*\*\*

\*\*\*\*\*

Signup

Watchlist Ideas

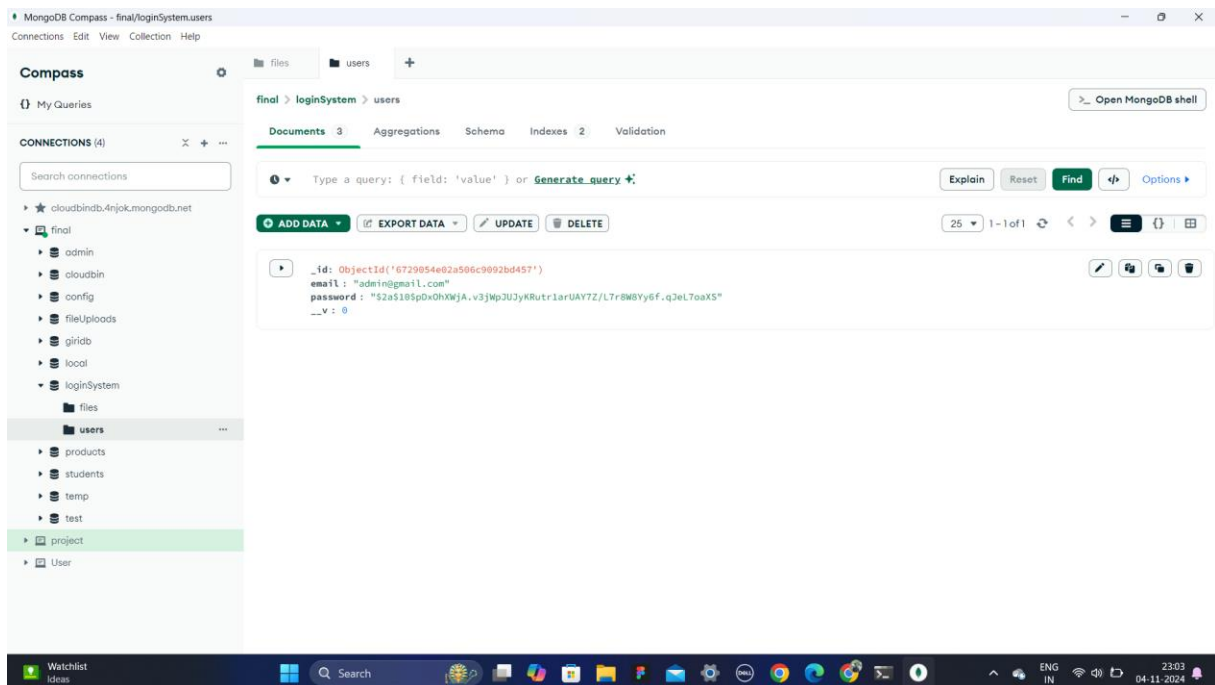
Search

ENG IN

23:02

04-11-2024

## ***USERS COLLECTION (NoSQL DATABASE ~ MongoDB)***



MongoDB Compass - final/loginSystem.users

Connections Edit View Collection Help

Compass

My Queries

CONNECTIONS (4)

Search connections

cloudbindb.4njok.mongodb.net

final

admin

cloudbin

config

fileUploads

giridb

local

loginSystem

files

users

products

students

temp

test

project

User

final > loginSystem > users

Documents 3 Aggregations Schema Indexes 2 Validation

Type a query: { field: 'value' } or [Generate query](#)

Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

25 1-1 of 1

`{ "_id": ObjectId("6720054e02a506c9092bd457"), "email": "admin@gmail.com", "password": "52a5185pDx0hXkJA.v3JMp3U3yK8Rutr1arUAY7Z/L7r8W8Yy6f.q3eL7oaxS", "__v": 0 }`

Watchlist Ideas

Search

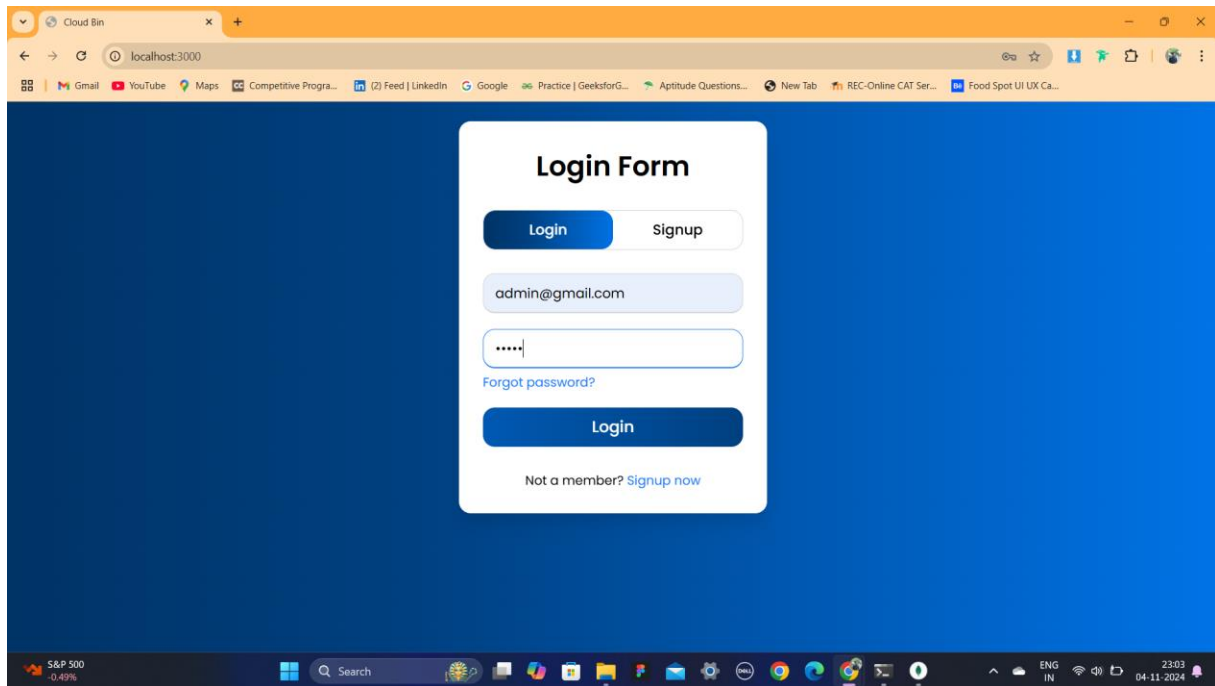
ENG IN

23:02

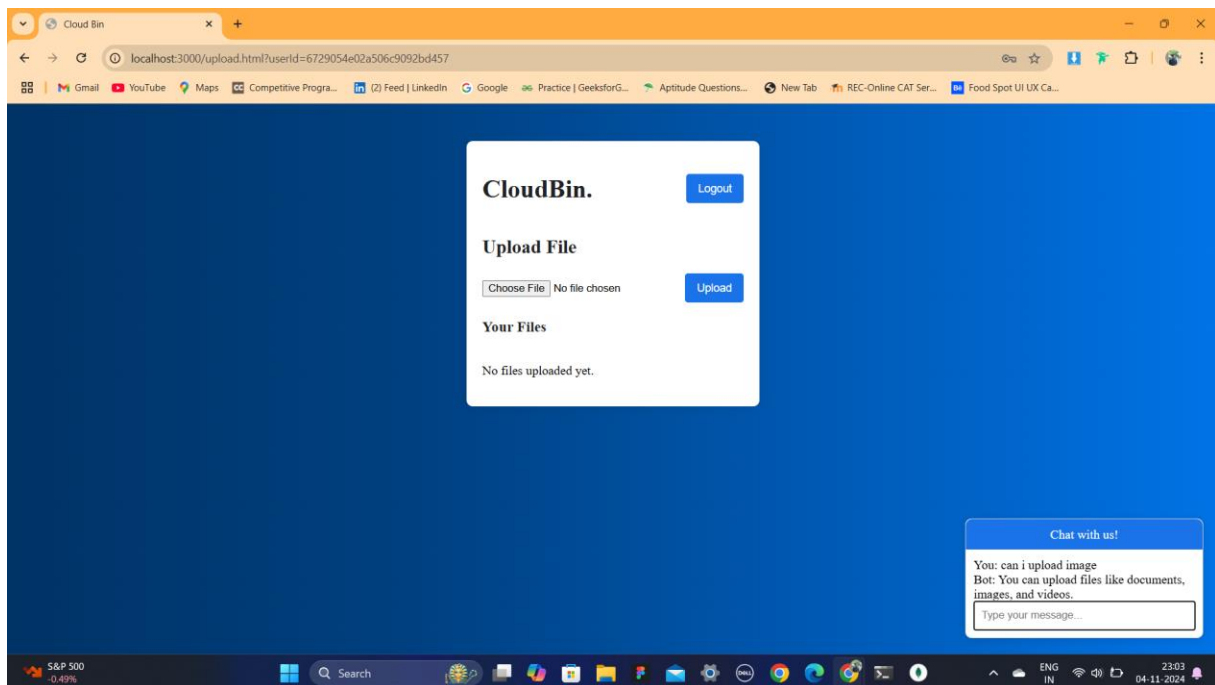
04-11-2024



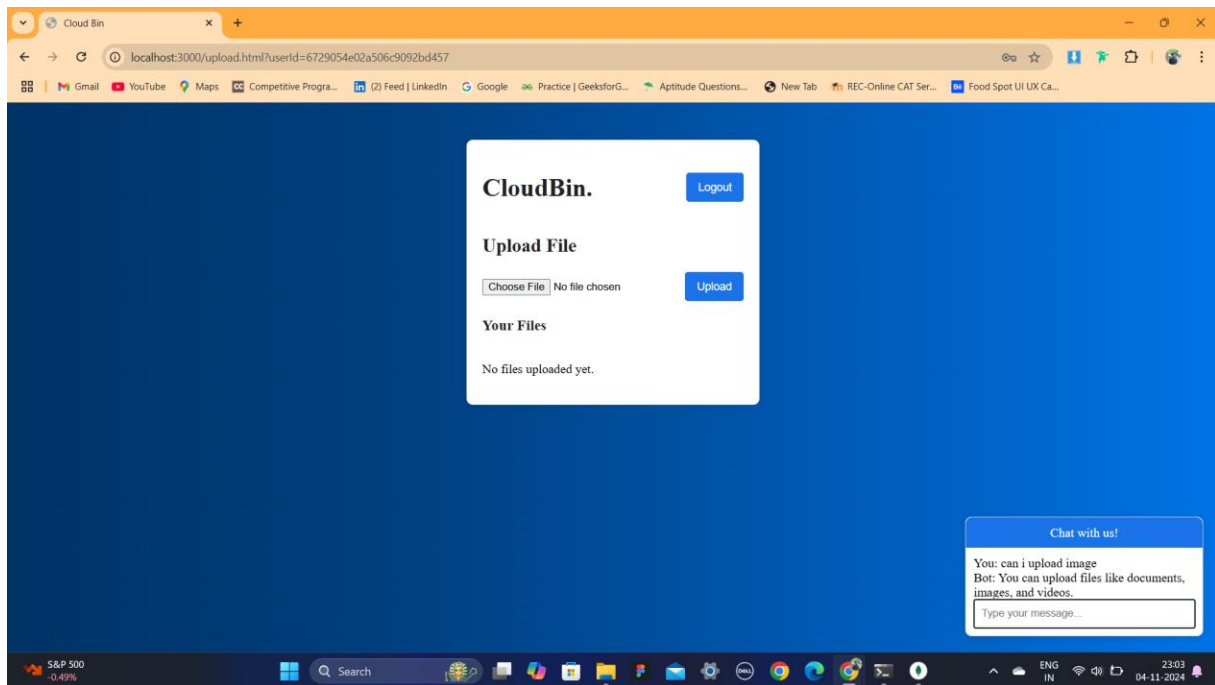
## EXISTING USER LOGIN



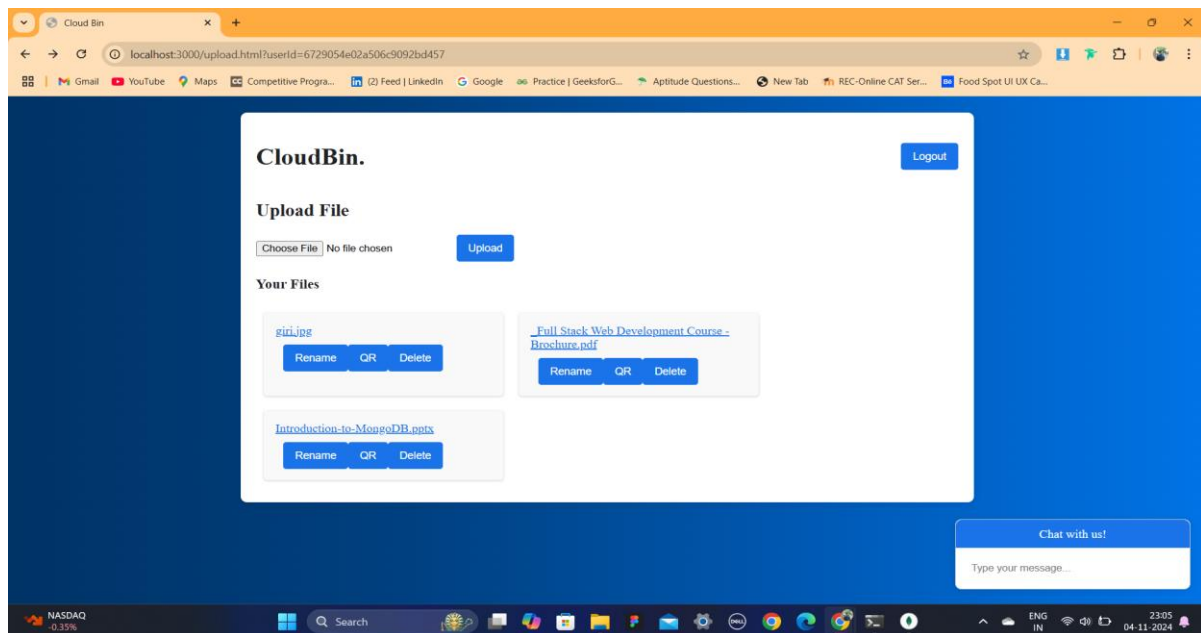
## UPLOAD PAGE



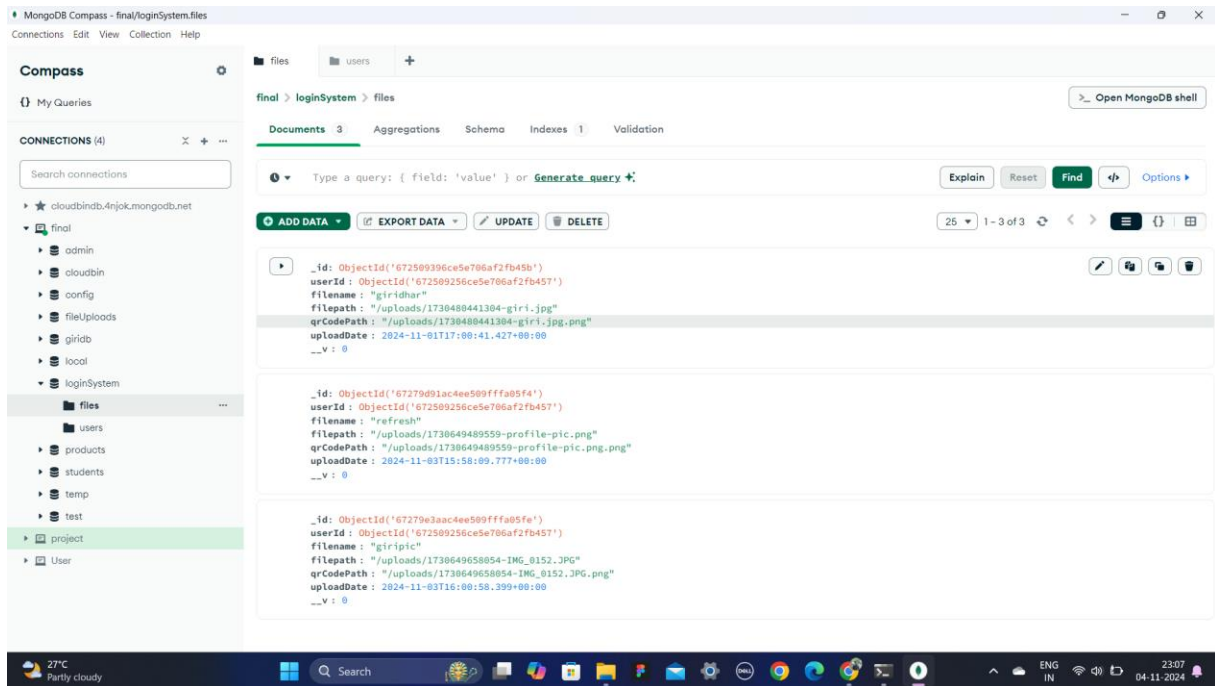
## ***UPLOAD PAGE (ALONG WITH RULE BASED AI CHATBOT)***



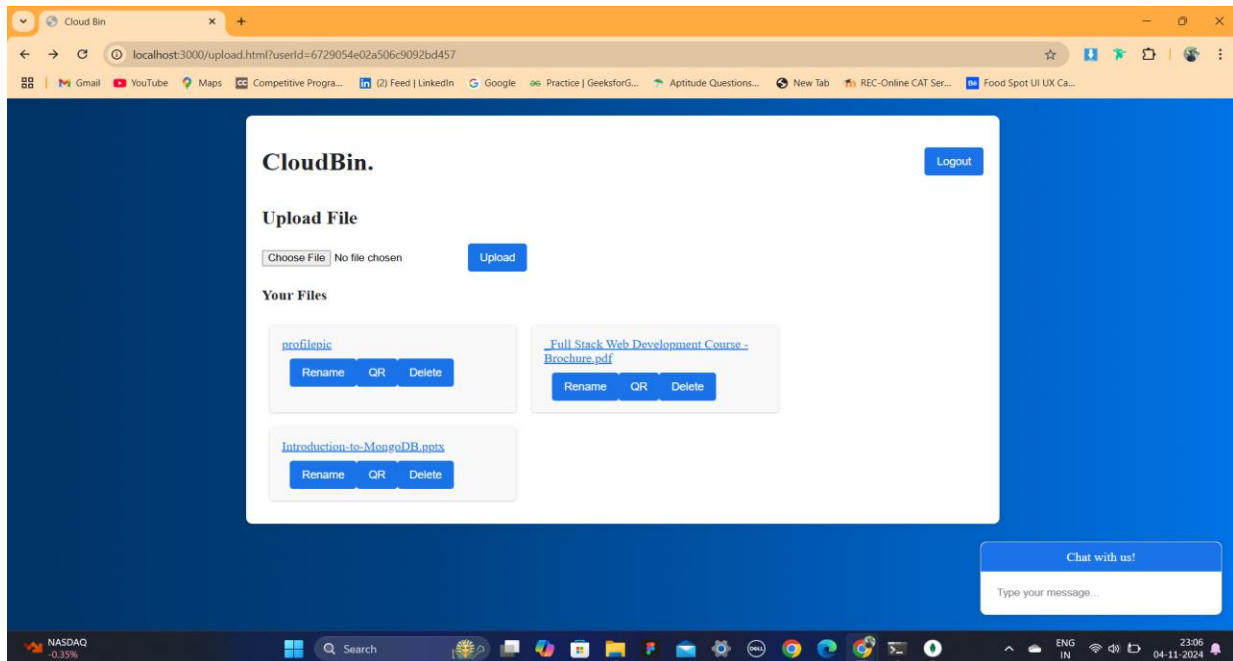
## ***EXISTING FILES IN THE USER'S ACCOUNT***



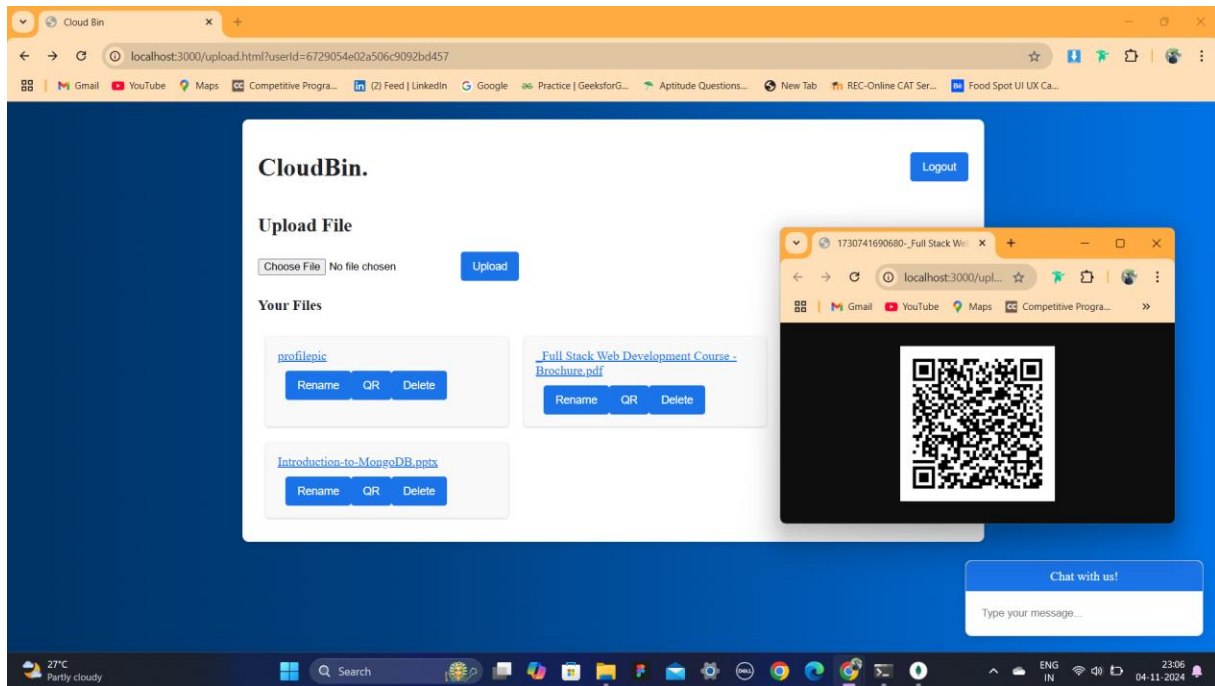
## EXISTING DOCUMENTS IN THE FILES COLLECTION



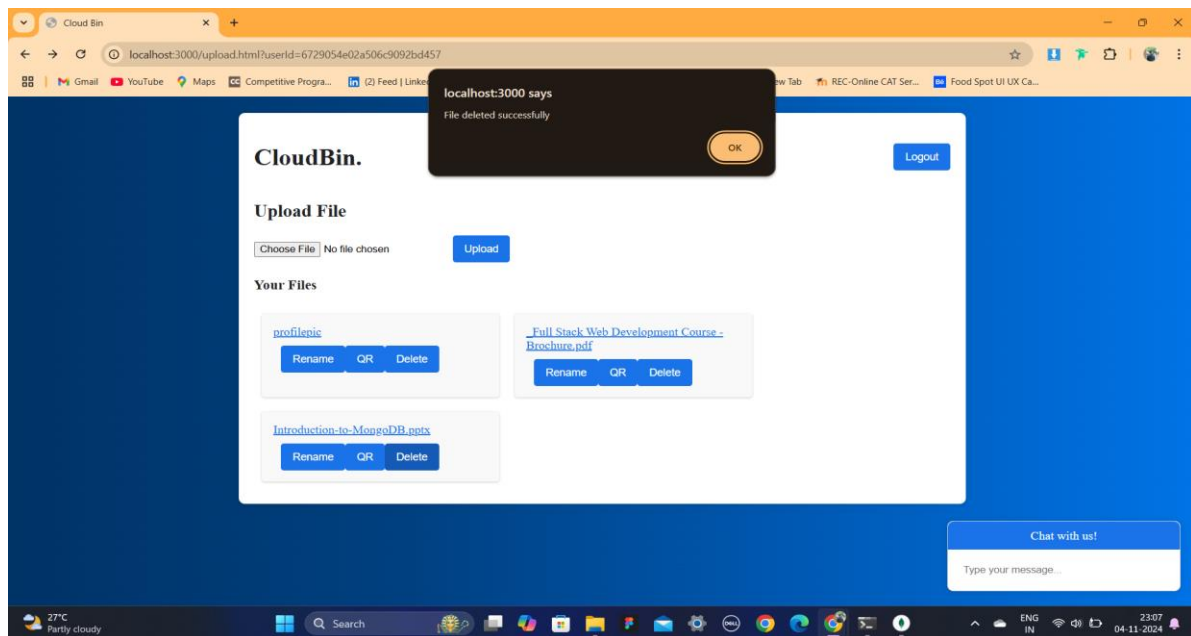
## RENAMING THE FILE



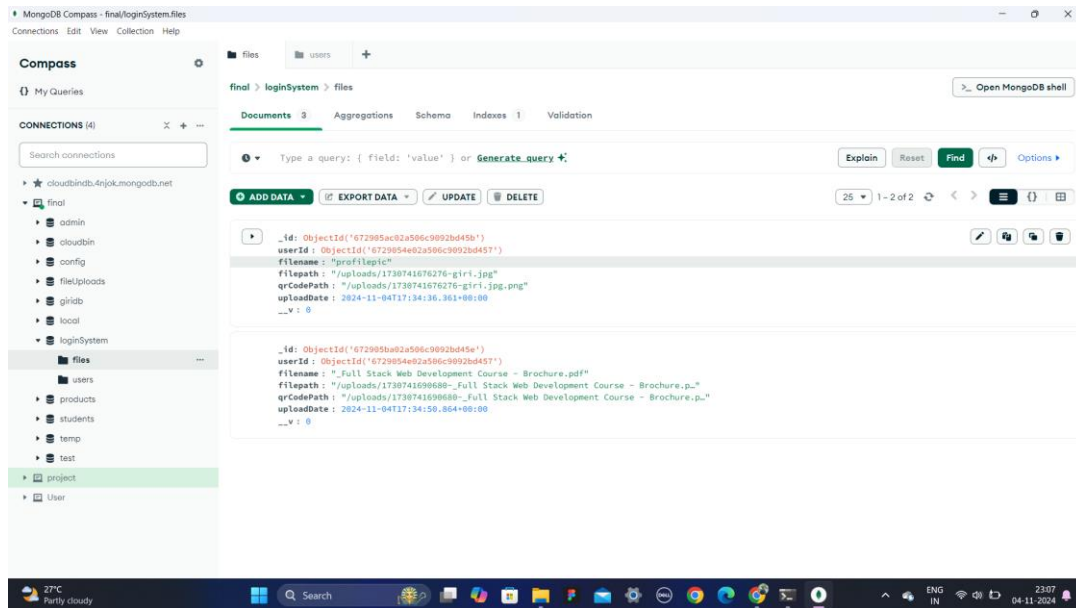
## ***QR CODE FOR SHARING FILE***



## ***FILE DELETED BY THE USER***



## FILES COLLECTION (UPDATED)



## **6) RESULTS AND DISCUSSION**

The implementation of CloudBin as a file management and sharing system has yielded significant results that demonstrate its effectiveness in enhancing user experience, streamlining workflows and ensuring secure file handling. This section presents the key outputs from the application, accompanied by discussions that highlight their implications and relevance to users.

### **➤ User Interface Output**

The user interface of CloudBin is designed to be intuitive and user-friendly. Users can easily navigate through different sections, including file uploads, document views and settings. The clean layout enhances usability, allowing users to manage their files efficiently without a steep learning curve.

### **➤ File Upload Functionality**

Users can drag and drop files or select them from their device. The system supports multiple file types, ensuring versatility in file management. The progress bar indicates upload status, providing users with real-time feedback on their actions.

### **➤ QR Code Generation**

CloudBin generates unique QR codes for each uploaded file. This feature simplifies the sharing process, allowing users to easily distribute access to their files via scannable codes. The integration of QR codes enhances collaboration by enabling quick access to documents without the need for email attachments or links.

## ➤ **Security Features**

The use of Bcryptjs for password protection ensures that user credentials are securely stored, while customizable access permissions allow users to control who can view or edit their files. These measures significantly enhance data security and user trust in the application.

## 7) CONCLUSION

In conclusion, CloudBin stands as a comprehensive solution for modern file management and sharing needs, addressing the challenges faced by individuals and organizations in today's digital landscape. The implementation of this system has demonstrated significant benefits, including enhanced productivity, improved collaboration and robust security measures. The user-friendly interface of CloudBin simplifies the organization and retrieval of files, allowing users to manage their digital assets efficiently. With features such as real-time file uploads, QR code generation for easy sharing and customizable access permissions, CloudBin not only streamlines workflows but also fosters teamwork among users.

This is particularly important in environments where collaboration across different locations and time zones is essential. Security remains a top priority in CloudBin's design. By employing advanced encryption methods and secure password hashing through libraries like Bcryptjs, the application ensures that sensitive user data is protected from unauthorized access. Additionally, customizable access controls allow organizations to manage permissions effectively, minimizing the risk of data breaches.

Furthermore, the scalability of CloudBin allows it to grow alongside user demands. As more individuals and teams adopt cloud-based solutions for file management, CloudBin provides a flexible infrastructure that can accommodate increasing storage needs without compromising performance. The integration of modern technologies such as Node.js, Express.js and MongoDB enhances the application's capabilities while ensuring efficient data handling and retrieval. These technologies not only improve operational efficiency but also contribute to a smoother user experience.

Overall, CloudBin represents a significant advancement in file management and sharing systems, offering a reliable platform that meets the evolving needs of users. As businesses continue to embrace digital transformation, solutions like CloudBin will play an integral role in facilitating effective document management and fostering collaboration across diverse teams.



## **8) REFERENCES**

- S. Jin, Y. Wang, and H. Zhang, “Research on Electronic Document Management System Based on Cloud Computing,” *Computers, Materials & Continua*, vol. 66, no. 3, pp. 2645-2654, 2021. doi: 10.32604/cmc.2021.014371.
- 2 J. Crouzier and O. Bergman, “A Comprehensive Investigation of Researchers' Shared File Management Practices in Cloud Storage,” *Journal of Information Science*, vol. 50, no. 2, pp. 123-145, 2024. doi: 10.1080/07370024.2024.2310551.
- 3 V. Jawale, V. Jundre, and R. Bathe, “Secure Cloud Based Document Management System,” *International Journal of Engineering Research and Technology (IJERT)*, vol. 2, no. 3, pp. 1-5, 2013. [Online]. Available: <https://www.ijert.org/research/secure-cloud-based-document-management-system-IJERTV2IS3039.pdf>.
- 4 A.I. Regla and P.S. Marquez, “Workplace Document Management System Employing Cloud Computing and Social Technology,” in *Computational Science and Technology*, R. Alfred et al., Eds., vol. 603, pp. 123-134, Springer Nature Singapore Pte Ltd., 2020.
- 5 H. Straub and O. Breitenstein, “Estimation of heat loss in thermal wave experiments,” *J. Appl. Phys.*, vol. 109, no. 3, Art no. 064515, Mar. 2011, doi: 10.1063/1.3549734.

## **GitHub link**

- <https://github.com/GIRIDHAR-U-47/MiniProject>
- <https://github.com/Ezhil-Adhithya-P/MiniProject>
- <https://github.com/Harish-clg/MiniProject>
- <https://github.com/Aravind262005/MiniProject>