**Blockchain A Game Changer for Secure Transferring of Data**

**Abstract**: A consortium of organizations collaborates and exchanges information to create synergies in their operations. Centralized systems of file-sharing cannot provide distributed trust and transparency. Blockchain technology can be used to share files securely and transparently. This paper proposes a blockchain-based inter-organizational secure file-sharing system. It can be used by a consortium of organizations to securely exchange files in a distributed fashion. Hyperledger Fabric, an enterprise blockchain framework, is used for blockchain network setup and the development of smart contracts. The Inter Planetary File System (IPFS) is used for storing files in a distributed way. The paper provides the workflow for identity management and file-sharing processes. The proposed system allows a consortium of organizations to share files with confidentiality, integrity, and availability using blockchain.

**Existing System**

In order to preserve the privacy for traceable encryption in blockchain, previous works proposed a system in which authenticity and non-repudiation of digital content is guaranteed. The problem tackled by authors is the secret key of user, which when shared with other entities does not hold the specific information of user. In case the shared key is corrupted or abused, it makes difficult to analyse the source of secret key. Moreover, leakage of confidential information in access control is a bottleneck for existing systems. Therefore, previous works have integrated the privacy protection algorithm such as attribute based encryption (ABE) to secure the secret keys.

**Proposed System:**

The proposed system provides secure file-sharing across a consortium of organizations using blockchain. It provides confidentiality, integrity, and availability of shared files. It ensures end to end encryption of the files. The content ID of the shared file is stored on the blockchain in a tamper resistant way. The encrypted file and file metadata is stored in a distributed fashion on the distributed storage and blockchain ledger respectively.

**System Requirements**

**Hardware Requirements**

|  |  |
| --- | --- |
| RAM | 4 GB Minimum |
| Processor | i3 Minimum |
| Hard disk | 500 GB HDD Min |

**Software Requirements**

|  |  |
| --- | --- |
| Technology | Python 3.6 |
| Operating System | Windows Family |
| IDE | VS Code |
| Technology | Python, Django |
| Database Server | MySQL |
| Front Design Technology | HTML, CSS, JS |

**INTRODUCTION**

With the increasing need for collaboration and information exchange among organizations, there is a demand for secure and transparent file-sharing systems. Existing centralized systems of file-sharing cannot provide the necessary levels of trust and transparency, especially when it comes to sharing sensitive information (Huang et al., 2020). However, the use of blockchain technology can address these concerns by enabling secure and transparent file-sharing in a distributed fashion. By leveraging Hyperledger Fabric as the underlying blockchain framework and the Inter Planetary File System for decentralized storage, this proposed system provides a reliable and efficient solution for secure file-sharing among consortiums of organizations. The proposed system allows organizations within a consortium to securely exchange files by leveraging blockchain technology. Through the implementation of smart contracts on Hyperledger Fabric, identities and permissions can be managed efficiently and securely. This ensures that only authorized users can access and share files, providing an added layer of security. Furthermore, the use of multi-signatures for usage control enhances the overall protection of digital content.

**LITERATURE SURVEY**

**S. Nakamoto, “bitcoin: A peer-to-peer electronic cash system,” 2008.**

A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone. We have proposed a system for electronic transactions without relying on trust. We started with the usual framework of coins made from digital signatures, which provides strong control of ownership, but is incomplete without a way to prevent double-spending. To solve this, we proposed a peer-to-peer network using proof-of-work to record a public history of transactions that quickly becomes computationally impractical for an attacker to change if honest nodes control a majority of CPU power. The network is robust in its unstructured simplicity. Nodes work all at once with little coordination. They do not need to be identified, since messages are not routed to any particular place and only need to be delivered on a best effort basis. Nodes can leave and rejoin the network at will, accepting the proof-of-work chain as proof of what happened while they were gone. They vote with their CPU power, expressing their acceptance of valid blocks by working on extending them and rejecting invalid blocks by refusing to work on them. Any needed rules and incentives can be enforced with this consensus mechanism.

**Naz M, Al-zahrani FA, Khalid R, Javaid N, Qamar AM, Afzal MK, Shafiq M. A secure data sharing platform using blockchain and interplanetary file system. Sustainability. 2019 Dec 10;11(24):7054.**

In a research community, data sharing is an essential step to gain maximum knowledge

from the prior work. Existing data sharing platforms depend on trusted third party (TTP). Due to the involvement of TTP, such systems lack trust, transparency, security, and immutability. To overcome these issues, this paper proposed a blockchain-based secure data sharing platform by leveraging the benefits of interplanetary file system (IPFS). A meta data is uploaded to IPFS server by owner and then divided into n secret shares. The proposed scheme achieves security and access control by executing the access roles written in smart contract by owner. Users are first authenticated through RSA signatures and then submit the requested amount as a price of digital content. After the successful delivery of data, the user is encouraged to register the reviews about data. These reviews are validated through Watson analyzer to filter out the fake reviews. The customers registering valid reviews are given incentives. In this way, maximum reviews are submitted against every file. In this scenario, decentralized storage, Ethereum blockchain, encryption, and incentive mechanism are combined. To implement the proposed scenario, smart contracts are written in solidity and deployed on local Ethereum test network. The proposed scheme achieves transparency, security, access control, authenticity of owner, and quality of data. In simulation results, an analysis is performed on gas consumption and actual cost required in terms of USD, so that a good price estimate can be done while deploying the implemented scenario in real set-up. Moreover, computational time for different encryption schemes are plotted to represent the performance of implemented scheme, which is shamir secret sharing (SSS). Results show that SSS shows the least computational time as compared

to advanced encryption standard (AES) 128 and 256.

**Liu J, Li X, Ye L, Zhang H, Du X, Guizani M. BPDS: A blockchain based privacy-preserving data sharing for electronic medical records. In2018 IEEE Global Communications Conference (GLOBECOM) 2018 Dec 9 (pp. 1-6). IEEE.**

Electronic medical record (EMR) is a crucial form of healthcare data, currently drawing a lot of attention. Sharing health data is considered to be a critical approach to improve the quality of healthcare service and reduce medical costs. However, EMRs are fragmented across decentralized hospitals, which hinders data sharing and puts patients' privacy at risks. To address these issues, we propose a blockchain based privacy-preserving data sharing for EMRs, called BPDS. In BPDS, the original EMRs are stored securely in the cloud and the indexes are reserved in a tamper-proof consortium blockchain. By this means, the risk of the medical data leakage could be greatly reduced, and at the same time, the indexes in blockchain ensure that the EMRs can not be modified arbitrarily. Secure data sharing can be accomplished automatically according to the predefined access permissions of patients through the smart contracts of blockchain. Besides, the joint-design of the CP-ABE-based access control mechanism and the content extraction signature scheme provides strong privacy preservation in data sharing. Security analysis shows that BPDS is a secure and effective way to realize data sharing for EMRs.

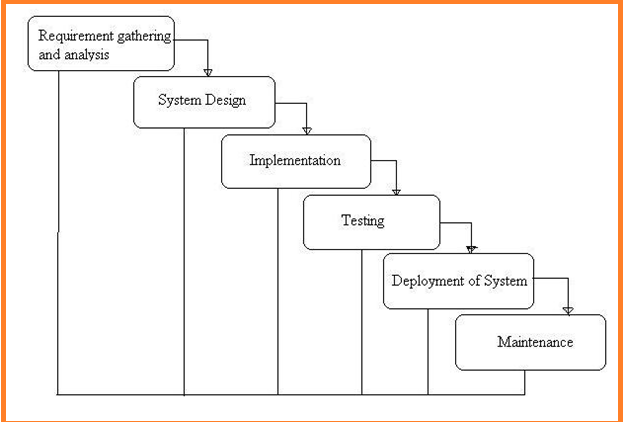
**Satapathy U, Mohanta BK, Panda SS, Sobhanayak S, Jena D. A secure framework for communication in internet of things application using hyperledger based blockchain. In2019 10th international conference on computing, communication and networking technologies (ICCCNT) 2019 Jul 6 (pp. 1-7). IEEE.**

In the era of the Internet of Things(IoT) smart devices are connected with wire or wireless way. The IoT devices are capable of sensing the environment and has the ability to transmit that information to the next level. The application area of IoT is Smart city, Smart transportation,

Healthcare sector, Agriculture, Monitoring environment. Each of these applications, lots of information are share or transmit among different devices. In the information sharing system

among devices, lots of security and privacy challenges exist like data leakage, data modification, device identity. In this paper, authors firstly identify the communication protocols used in IoT application and given their working principle. Secondly, challenges exist in IoT and corresponding Blockchain solution approach are explained, Lastly, the authors proposed a secure architecture based on open Blockchain which can solve some of the challenges in IoT applications. It is anticipated that Blockchain will make far-reaching changes in IoT applications in near future. We have addressed the vulnerabilities occur in secure communication in an IoT application by integrating it with Blockchain. In this paper, we proposed an architecture to communicate securely in IoT Applications using Hyderledger based Blockchain. In IoT application the data sent by various devices stored in a centralized database making it vulnerable for security breaches. Also the authenticity of the sender could not be verified properly making it open for security threats. In this paper we have proposed a secure architecture based on open Blockchain (Hyperledger) for IoT applications. Malicious actor detection will be easy as every node aware of all other node in the Hyperledger network. Concluding it guarantees the security measures for non-repudiation, privacy and scalablity in an IoT application.

**SYSTEM ANALYSIS**



**What is Waterfall Model?**

Waterfall Model is a sequential model that divides software development into different phases. Each phase is designed for performing specific activity during SDLC phase. It was introduced in 1970 by Winston Royce.

**Requirements:**

The first phase involves understanding what needs to design and what is its function, purpose, etc. Here, the specifications of the input and output or the final product are studied and marked.

**System Design:**

The requirement specifications from the first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The software code to be written in the next stage is created now.

**Implementation:**

With inputs from system design, the system is first developed in small programs called units, which are integrated into the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

**Integration and Testing:**

All the units developed in the implementation phase are integrated into a system after testing of each unit. The software designed, needs to go through constant software testing to find out if there are any flaws or errors. Testing is done so that the client does not face any problem during the installation of the software.

**Deployment of System:**

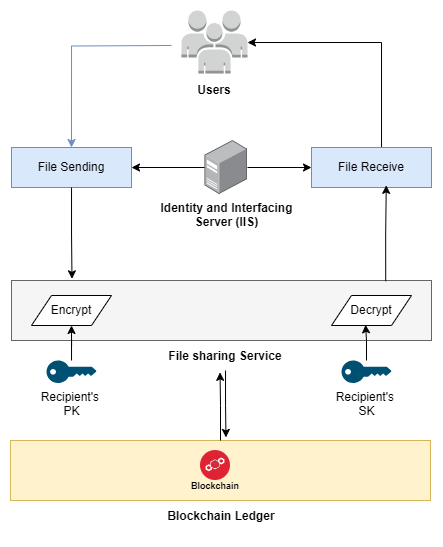
Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.

**Maintenance:**

This step occurs after installation, and involves making modifications to the system or an individual component to alter attributes or improve performance. These modifications arise either due to change requests initiated by the customer, or defects uncovered during live use of the system. The client is provided with regular maintenance and support for the developed software.

**IMPLEMENTATION**

In a consortium of organizations, a number of organizations can share data in the form files and synergies their operations. A blockchain network created among multiple organizations, each of the organizations will host Identity and Interfacing Server (IIS), Smart contract, and blockchain ledger. IIS maintains the identity details in identity database and is also the interfacing point with the smart contract. A smart contract is a program, which contains the business logic of the proposed file-sharing mechanism, is installed on each of the organizations. The blockchain ledger maintains transactions in the form of blocks. The following Figure illustrates the high-level view of the proposed system.



**Modules**

**Identity and Interfacing Server**

For sharing files among the users of the participant or organizations, the concerned users need to be registered with the blockchain through the smart contract. A key pair (pki, ski) is generated by the end user application. The private key ski is kept with the user machine and the public key along with user details like name, email, organization, password, etc is sent to the IIS. IIS admin verifies the user identity registration request. Upon successful verification the hash of the password is written to the identity database.   
**Blockchain Ledger**

IIS sends the user registration request to the smart contract with public key pki and user details. The smart contract generates the blockchain identifier BCIDi and inserts BCIDi, userdetails, pki into blockchain ledger. The blockchain update status(success/failure) and BCIDi is sent to IIS. IIS sends the user identity registration status and BCIDi to the end user application.

**File Sharing**

Once the user is registered with blockchain, he can securely share files with any other registered user. The user logs in using the user authentication process. The user selects the file to be shared and specifies the file receivers. The end user application generates the symmetric key, K and encrypts file, M to be shared using K. The encrypted file M is uploaded to the IPFS distributed storage. IPFS returns content ID of the uploaded file, Mcid to the end user application. The end user application generates file identifier, Mfid for the unique identification of the file. The end user application requests the public keys of (R1, R2, ...Rn) i.e., the receivers with whom the file is to be shared.

**Feasibility study**

A feasibility analysis evaluates the project’s potential for success; therefore, perceived objectivity is an essential factor in the credibility of the study for potential investors and lending institutions.

**Technical Feasibility**

Technical resources need for project Development

* Windows family Operating System
* Python 3.6 Technology
* Vs Code
* Mysql
* Sqlyog

**Economic Fesibility**

Cost/ benefits analysis of the project as over project is academic project we will not have only basic cost for learning of the technologies

**Operational Feasibility**

This assessment involves undertaking a study to analyze and determine whether and how well the organization’s needs can be met by completing the project. Operational feasibility studies also examine how a project plan satisfies the requirements identified in the requirements analysis phase of system development.

**Functional Requirements**

* Admin Login
* User Login
* User Signup
* View Block Chain

**NON- Functional Requirements**

* Serviceability requirement
* Manageability requirement
* Recoverability requirement
* Security requirement

**SOFTWARE OVER VIEW:**

**History of Python**

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

**Input as CSV File**

Reading data from CSV(comma separated values) is a fundamental necessity in Data Science. Often, we get data from various sources which can get exported to CSV format so that they can be used by other systems. The Panadas library provides features using which we can read the CSV file in full as well as in parts for only a selected group of columns and rows.

The CSV file is a text file in which the values in the columns are separated by a comma. Let's consider the following data present in the file named input.csv.You can create this file using windows notepad by copying and pasting this data. Save the file as input.csv using the save As All files(\*.\*) option in notepad.

import pandas as pd

data= pd.read\_csv('path/input.csv')

print(data)

**Operations using NumPy**

NumPy is a Python package which stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Using NumPy, a developer can perform the following operations −

* Mathematical and logical operations on arrays.
* Fourier transforms and routines for shape manipulation.
* Operations -related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

## Key Features of Pandas

* Fast and efficient DataFrame object with default and customized indexing.
* Tools for loading data into in-memory data objects from different file formats.
* Data alignment and integrated handling of missing data.
* Reshaping and pivoting of date sets.
* Label-based slicing, indexing and subsetting of large data sets.
* Columns from a data structure can be deleted or inserted.
* Group by data for aggregation and transformations.
* High performance merging and joining of data.
* Time Series functionality.

**Django Web framework for Python**

**Introduction to Django**

If you go to the Web site djangoproject.com using your Web browser or, depending on the decade in which you’re reading this destined-to-be-timeless literary work, using your cell phone, electronic notebook, shoe, or any Internet-superceding contraption you’ll find this explanation: “Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design.” That’s a mouthful or eyeful or pixelful, depending on whether this book is being recited, read on paper or projected to you on a Jumbotron, respectively. Let’s break it down. Django is a high-level Python Web framework… A high-level Web framework is software that eases the pain of building dynamic Web sites. It abstracts common problems of Web development and provides shortcuts for frequent programming tasks. For clarity, a dynamic Web site is one in which pages aren’t simply HTML documents sitting on a server’s filesystem somewhere. In a dynamic Web site, rather, each page is generated by a computer program a so-called “Web application” that you, the Web developer, create. A Web application may, for instance, retrieve records from a database or take some action based on user input. A good Web framework addresses these common concerns:

● It provides a method of mapping requested URLs to code that handles requests. In other words, it gives you a way of designating which code should execute for which URL. For instance, you could tell the framework, “For URLs that look like /users/joe/, execute code that displays the profile for the user with that username.”

● It makes it easy to display, validate and redisplay HTML forms. HTML forms are the primary way of getting input data from Web users, so a Web framework had better make it easy to display them and handle the tedious code of form display and redisplay (with errors highlighted). ● It converts user-submitted input into data structures that can be manipulated conveniently. For example, the framework could convert HTML form submissions into native data types of the programming language you’re using.

-● It helps separate content from presentation via a template system, so you can change your site’s look-and-feel without affecting your content, and vice-versa.

● It conveniently integrates with storage layers such as databases but doesn’t strictly require the use of a database.

● It lets you work more productively, at a higher level of abstraction, than if you were coding against, say, HTTP. But it doesn’t restrict you from going “down” one level of abstraction when needed.

● It gets out of your way, neglecting to leave dirty stains on your application such as URLs that contain “.aspx” or “.php”.

It is never too late to start **learning** and it would be a shame to miss an opportunity to learn a tutorial or course that can be so useful as **Django Web framework for Python** especially when it is free! You do not have to register for expensive classes and travel from one part of town to another to take classes. All you need to do is download the course and open the PDF file. This specific program is classified in the Web programming category where you can find some other similar courses.

Thanks to people (like you?) Who share their knowledge, you can discover the extent of our being selected to easily learn without spending a fortune! Django Web framework for Python. is available for free by its author. But also many other tutorials are accessible just as easily!

Computer PDF guide you and allow you to save on your studies.

**Django Web framework for Python**. help on the contact form if problems

Computer PDF is also courses for training in **html & html5, css & css3, javascript, php, asp, j2ee, ajax, jquery, node.js, angularjs** and many others IT.

You should come see our Web programming documents. You will find your happiness without trouble !

The latest news and especially the best tutorials on your favorite topics, that is why Computer PDF is number 1 for courses and tutorials for download in pdf files - Django Web framework for Python. and Web programming!

Download other tutorials for advice on Django Web framework for Python. you will see! We will do everything to help you!

And you dear surfers what you need? The best **course and tutorial, and how to learn and use Django Web framework for Python**. of course!

**About MySQL:**

**MySQL** is a relational database management system (RDBMS)[]](http://en.wikipedia.org/wiki/MySQL#cite_note-1) that runs as a server providing multi-user access to a number of databases.  The SQL phrase stands for Structured Query Language.Free-software-open source projects that require a full-featured database management system often use MySQL. For commercial use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: TYPO3, Joomla, WordPress, phpBB, Drupal and other software built on the LAMP software stack. MySQL is also used in many high-profile, large-scale World Wide Web products, including Wikipedia, Google  , Facebook, and Twitter.

MySQL is the world's most popular open source database software, with over 100 million copies of its software downloaded or distributed throughout it's history. With its superior speed, reliability, and ease of use, MySQL has become the preferred choice for Web, Web 2.0, SaaS, ISV, Telecom companies and forward-thinking corporate IT Managers because it eliminates the major problems associated with downtime, maintenance and administration for modern, online applications.

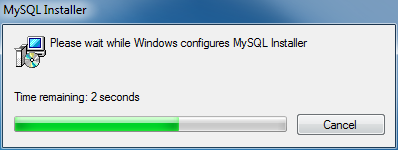
Many of the world's largest and fastest-growing organizations use MySQL to save time and money powering their high-volume Web sites, critical business systems, and packaged software including industry leaders such as Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, Wikipedia, and Booking.com.

The flagship MySQL offering is MySQL Enterprise, a comprehensive set of production-tested software, proactive monitoring tools, and premium support services available in an affordable annual subscription.

MySQL is a key part of LAMP (Linux, Apache, MySQL, PHP / Perl / Python), the fast-growing open source enterprise software stack. More and more companies are using LAMP as an alternative to expensive proprietary software stacks because of its lower cost and freedom from platform lock-in.

MySQL was originally founded and developed in Sweden by two Swedes and a Finn: David Axmark, Allan Larsson and Michael "Monty" Widenius, who had worked together since the 1980's. More historical information on MySQL is

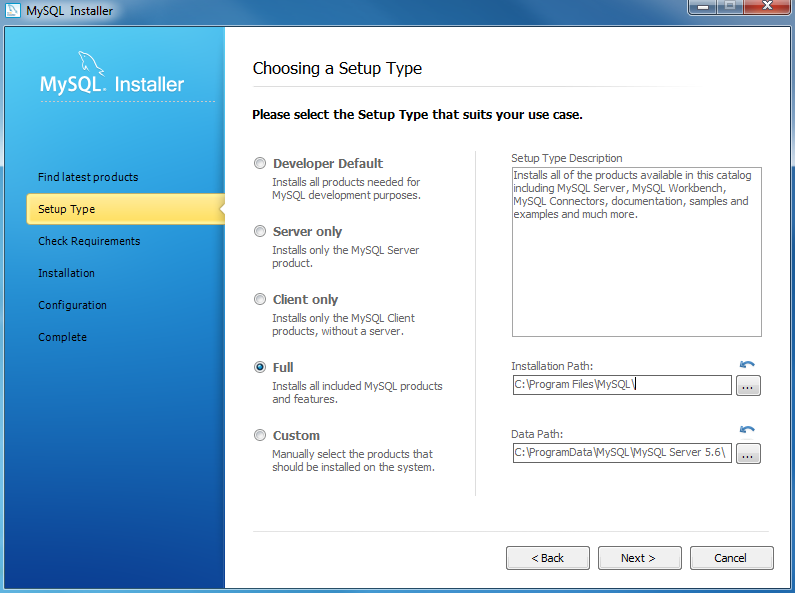
Download MySQL Installer from Office website <http://dev.mysql.com/downloads/installer/>.



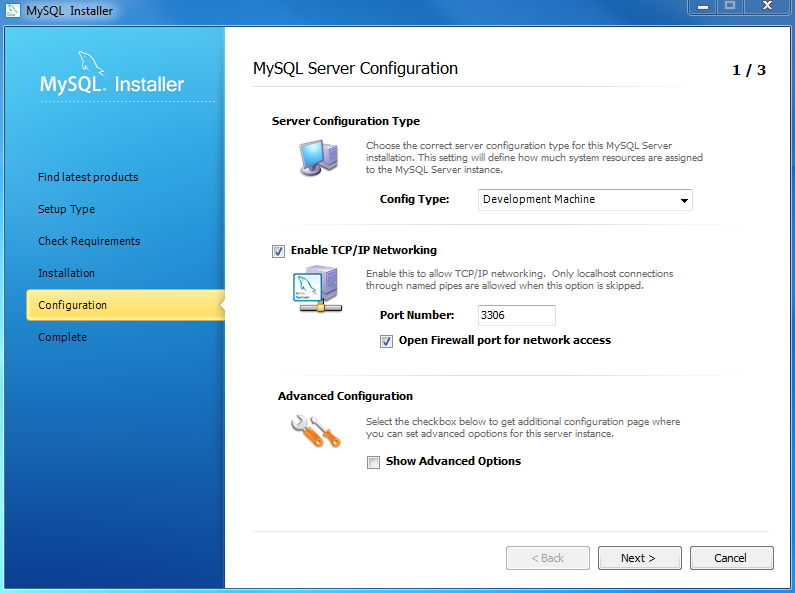
Install MySQL Step 1: Windows configures MySQL Installer



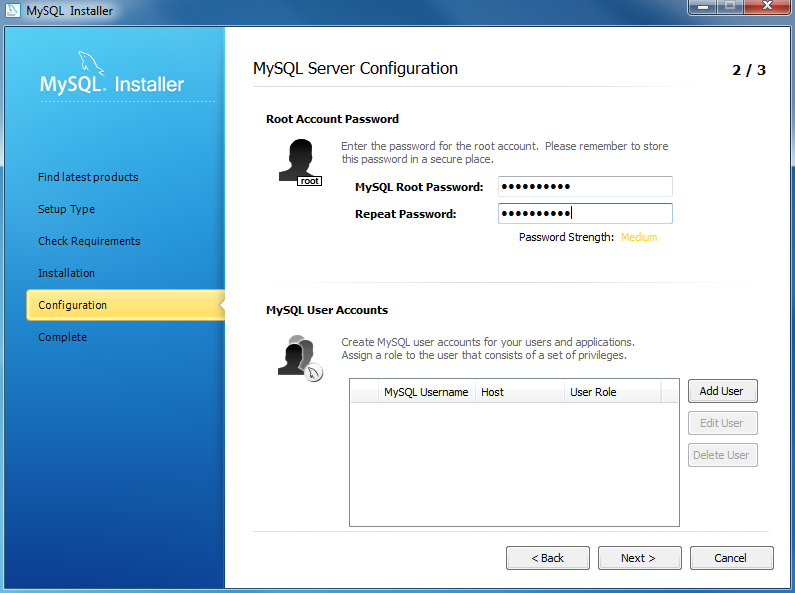
Select Install MySQL Products



Select Full version and disk space for installing

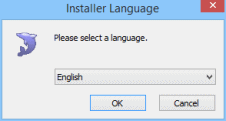


Set the connection port number “3306”

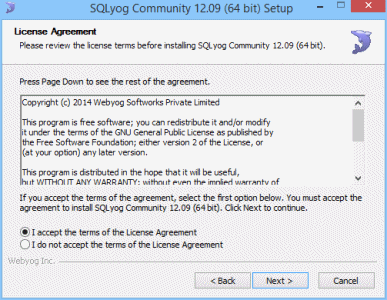


Set Password & Conform password

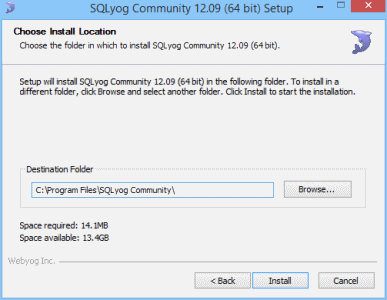
**Install SQLyog community edition**



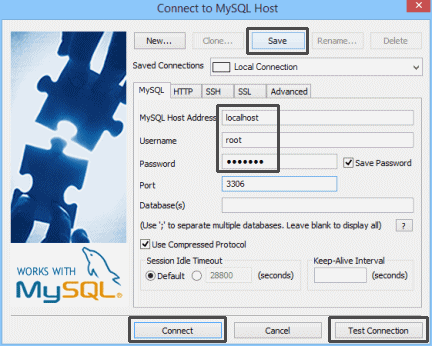
Select the Language



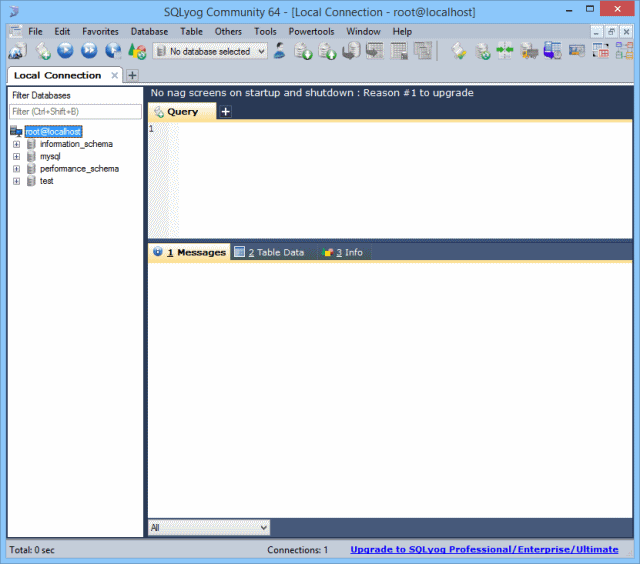
Accept terms of the software



Selecting the installing location in our system

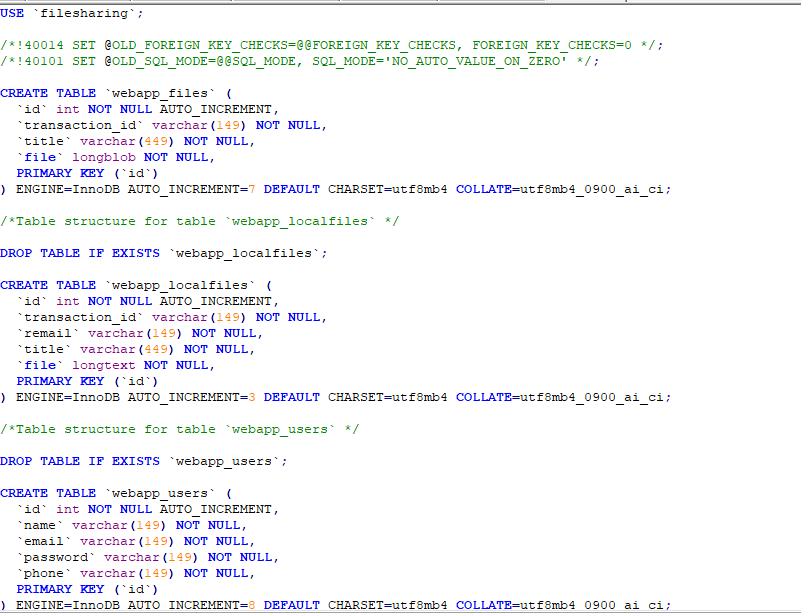


Create a New MYSQL Connection



Create any New Database

Project database tables

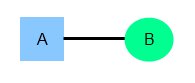


**Entity-relationship model**

A relationship is how the data is shared between entities. There are three types of relationships between entities:

**One-to-One**

One instance of an entity (A) is associated with one other instance of another entity (B). For example, in a database of employees, each employee name (A) is associated with only one social security number (B).

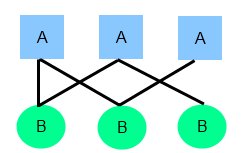


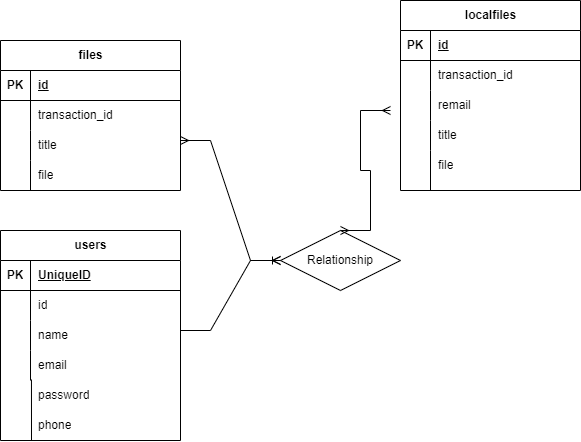
**2. One-to-Many**

One instance of an entity (A) is associated with zero, one or many instances of another entity (B), but for one instance of entity B there is only one instance of entity A. For example, for a company with all employees working in one building, the building name (A) is associated with many different employees (B), but those employees all share the same singular association with entity A.

**3. Many-to-Many**

One instance of an entity (A) is associated with one, zero or many instances of another entity (B), and one instance of entity B is associated with one, zero or many instances of entity A. For example, for a company in which all of its employees work on multiple projects, each instance of an employee (A) is associated with many instances of a project (B), and at the same time, each instance of a project (B) has multiple employees (A) associated with it.



****

**SYSTEM DESIGN**

**UML DIAGRAMS**

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

**Global Use Case Diagrams:**

Identification of actors:

**Actor:** Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

Graphical representation:



<<Actor name>>

An actor is someone or something that:

Interacts with or uses the system.

* Provides input to and receives information from the system.
* Is external to the system and has no control over the use cases.

Actors are discovered by examining:

* Who directly uses the system?
* Who is responsible for maintaining the system?
* External hardware used by the system.
* Other systems that need to interact with the system.

Questions to identify actors:

* + Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
  + Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
  + Which external hardware or systems (if any) use the system to perform tasks?
  + What problems does this application solve (that is, for whom)?
  + And, finally, how do users use the system (use case)? What are they doing with the system?

The actors identified in this system are:

1. **System Administrator**
2. **Customer**
3. **Customer Care**

Identification of usecases:

**Usecase:** A use case can be described as a specific way of using the system from a user’s (actor’s) perspective.

**Graphical representation:**



A more detailed description might characterize a use case as:

* Pattern of behavior the system exhibits
* A sequence of related transactions performed by an actor and the system
* Delivering something of value to the actor

Use cases provide a means to:

* capture system requirements
* communicate with the end users and domain experts
* test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Guide lines for identifying use cases:

* For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal
* Name the use cases.
* Describe the use cases briefly by applying terms with which the user is familiar.

This makes the description less ambiguous

Questions to identify use cases:

* What are the tasks of each actor?
* Will any actor create, store, change, remove or read information in the system?
* What use case will store, change, remove or read this information?
* Will any actor need to inform the system about sudden external changes?
* Does any actor need to inform about certain occurrences in the system?
* What usecases will support and maintains the system?

**Flow of Events**

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

* When and how the use case starts and ends
* Use case/actor interactions
* Data needed by the use case
* Normal sequence of events for the use case
* Alternate or exceptional flows

Construction of Usecase diagrams:

Use-case diagrams graphically depict system behavior (use cases). These diagrams present a high level view of how the system is used as viewed from an outsider’s (actor’s) perspective. A use-case diagram may depict all or some of the use cases of a system.

A use-case diagram can contain:

* actors ("things" outside the system)
* use cases (system boundaries identifying what the system should do)
* Interactions or relationships between actors and use cases in the system including the associations, dependencies, and generalizations.

Relationships in use cases:

**1. Communication:**

The communication relationship of an actor in a usecase is shown by connecting the actor symbol to the usecase symbol with a solid path. The actor is said to communicate with the usecase.

**2. Uses:**

A Uses relationship between the usecases is shown by generalization arrow from the usecase.

**3. Extends:**

The extend relationship is used when we have one usecase that is similar to another usecase but does a bit more. In essence it is like subclass.

**SEQUENCE DIAGRAMS**

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence what happens first, what happens next. Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.

There are two main differences between sequence and collaboration diagrams: sequence diagrams show time-based object interaction while collaboration diagrams show how objects associate with each other. A sequence diagram has two dimensions: typically, vertical placement represents time and horizontal placement represents different objects.

**Object:**

An object has state, behavior, and identity. The structure and behavior of similar objects are defined in their common class. Each object in a diagram indicates some instance of a class. An object that is not named is referred to as a class instance.

The object icon is similar to a class icon except that the name is underlined:

An object's concurrency is defined by the concurrency of its class.

**Message:**

A message is the communication carried between two objects that trigger an event. A message carries information from the source focus of control to the destination focus of control. The synchronization of a message can be modified through the message specification. Synchronization means a message where the sending object pauses to wait for results.

**Link:**

A link should exist between two objects, including class utilities, only if there is a relationship between their corresponding classes. The existence of a relationship between two classes symbolizes a path of communication between instances of the classes: one object may send messages to another. The link is depicted as a straight line between objects or objects and class instances in a collaboration diagram. If an object links to itself, use the loop version of the icon.

**CLASS DIAGRAM:**

Identification of analysis classes:

A class is a set of objects that share a common structure and common behavior (the same attributes, operations, relationships and semantics). A class is an abstraction of real-world items.

There are 4 approaches for identifying classes:

1. Noun phrase approach:
2. Common class pattern approach.
3. Use case Driven Sequence or Collaboration approach.
4. Classes , Responsibilities and collaborators Approach
5. **Noun Phrase Approach:**

The guidelines for identifying the classes:

* + Look for nouns and noun phrases in the usecases.
  + Some classes are implicit or taken from general knowledge.
  + All classes must make sense in the application domain; Avoid computer

implementation classes – defer them to the design stage.

* + Carefully choose and define the class names After identifying the classes we have to eliminate the following types of classes:
* Adjective classes.

1. **Common class pattern approach:**

The following are the patterns for finding the candidate classes:

* + Concept class.
  + Events class.
  + Organization class
  + Peoples class
  + Places class
  + Tangible things and devices class.

1. **Use case driven approach:**

We have to draw the sequence diagram or collaboration diagram. If there is need for some classes to represent some functionality then add new classes which perform those functionalities.

1. **CRC approach:**

The process consists of the following steps:

* + Identify classes’ responsibilities ( and identify the classes )
  + Assign the responsibilities
  + Identify the collaborators.

Identification of responsibilities of each class:

The questions that should be answered to identify the attributes and methods of a class respectively are:

1. What information about an object should we keep track of?
2. What services must a class provide?

Identification of relationships among the classes:

Three types of relationships among the objects are:

Association: How objects are associated?

Super-sub structure: How are objects organized into super classes and sub classes?

Aggregation: What is the composition of the complex classes?

Association:

The **questions** that will help us to identify the associations are:

1. Is the class capable of fulfilling the required task by itself?
2. If not, what does it need?
3. From what other classes can it acquire what it needs?

Guidelines for identifying the tentative associations:

* A dependency between two or more classes may be an association. Association often corresponds to a verb or prepositional phrase.
* A reference from one class to another is an association. Some associations are implicit or taken from general knowledge.

Some common association patterns are:

Location association like part of, next to, contained in…..

Communication association like talk to, order to ……

We have to eliminate the unnecessary association like implementation associations, ternary or n-ary associations and derived associations.

Super-sub class relationships:

Super-sub class hierarchy is a relationship between classes where one class is the parent class of another class (derived class).This is based on inheritance.

Guidelines for identifying the super-sub relationship, a generalization are

1***.* Top-down*:***

Look for noun phrases composed of various adjectives in a class name. Avoid excessive refinement. Specialize only when the sub classes have significant behavior.

2.**Bottom-up*:***

Look for classes with similar attributes or methods. Group them by moving the common attributes and methods to an abstract class. You may have to alter the definitions a bit.

3.**Reusability*:***

Move the attributes and methods as high as possible in the hierarchy.

4. **Multiple inheritances*:***

Avoid excessive use of multiple inheritances. One way of getting benefits of multiple inheritances is to inherit from the most appropriate class and add an object of another class as an attribute.

**Aggregation or a-part-of relationship:**

It represents the situation where a class consists of several component classes. A class that is composed of other classes doesn’t behave like its parts. It behaves very difficultly. The major properties of this relationship are transitivity and anti symmetry.

The **questions** whose answers will determine the distinction between the part and whole relationships are:

* Does the part class belong to the problem domain?
* Is the part class within the system’s responsibilities?
* Does the part class capture more than a single value?( If not then simply include it as an attribute of the whole class)
* Does it provide a useful abstraction in dealing with the problem domain?

There are three types of aggregation relationships. They are:

**Assembly:**

It is constructed from its parts and an assembly-partsituation physically exists.

**Container:**

A physical whole encompasses but is not constructed from physical parts.

**Collection member:**

A conceptual whole encompasses parts that may be physical or conceptual. The container and collection are represented by hollow diamonds but composition is represented by solid diamond.

**Use Case Diagram:**

****

**Sequence Diagram:**

****

****

**Collaboration:**

****

****

**StateChart Diagram:**

****

**Component Diagram:**

****

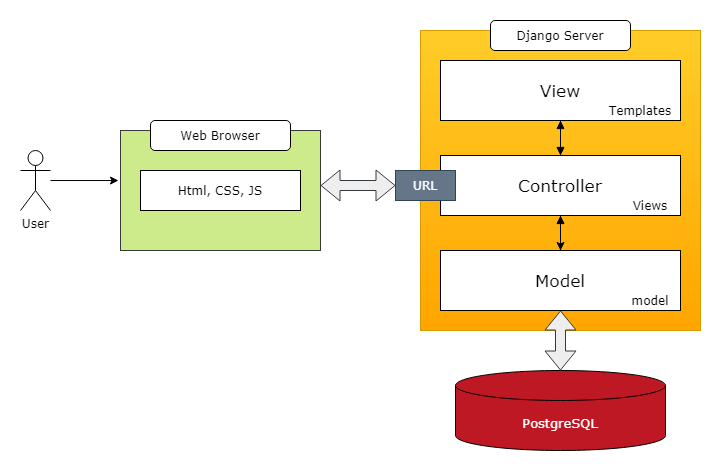
**Activity**

****

**Class**

****

**Deployment:**



**SOFTWARE TESTING**

Software testing is one of the main stages of project development life cycle to provide our cessation utilizer with information about the quality of the application and ours, in our Project we have under gone some stages of testing like unit testing where it’s done in development stage of the project when we are in implementation of the application after the Project is yare we have done manual testing with different Case of all the different modules in the application we have even done browser compatibility testing in different web browsers in market, even we have done Client side validation testing on our application

**Unit testing**

The unit testing is done in the stage of implementation of the project only the error are solved in development stage some of the error we come across in development are given below

**TESTING**Testing is the debugging program is one of the most critical aspects of the computer programming triggers, without programming that works, the system would never produce an output of which it was designed. Testing is best performed when user development is asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not quantity but quality of the data used the matters of testing. Testing is aimed at ensuring that the system was accurately an efficiently before live operation commands.

**Testing objectives:**

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

A successful test is one that uncovers an as yet undiscovered error.

A good test case is one that has probability of finding an error, if it exists.

The test is inadequate to detect possibly present errors.

The software more or less confirms to the quality and reliable standards.3999**Levels of Testing:**

In order to uncover present in different phases we have the concept of levels of testing.

**The basic levels of Testing:**

Client needs acceptance testing

Requirements system testing

Design integration testing

Code unit testing

Figure: Levels of Testing

**Code testing:**

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

**Specification Testing:**

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

**Unit testing:**

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

Each Module can be tested using the following two Strategies:

1. Black Box Testing
2. White Box Testing

**BLACK BOX TESTING**

**What is Black Box Testing?**

Black box testing is a software testing techniques in which **functionality of the software under test (SUT) is tested without looking at the internal code structure**, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

**In Black Box Testing we just focus on inputs and output of the software system** without bothering about internal knowledge of the software program.



The above Black Box can be any software system you want to test. For example : an operating system like Windows, a website like Google ,a database like Oracle or even your own custom application. Under Black Box Testing, you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

**Black box testing - Steps**

Here are the generic steps followed to carry out any type of Black Box Testing.

* Initially requirements and specifications of the system are examined.
* Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
* Tester determines expected outputs for all those inputs.
* Software tester constructs test cases with the selected inputs.
* The test cases are executed.
* Software tester compares the actual outputs with the expected outputs.
* Defects if any are fixed and re-tested.

**Types of Black Box Testing**

There are many types of Black Box Testing but following are the prominent ones -

* **Functional testing** – This black box testing type is related to functional requirements of a system; it is done by software testers.
* **Non-functional testing** – This type of black box testing is not related to testing of a specific functionality, but non-functional requirements  such as performance, scalability, usability.
* **Regression testing** – Regression testing is done  after code fixes , upgrades or any other system maintenance to check the new code has not affected the existing code.

**WHITE BOX TESTING**

White Box Testing is the testing of a software solution's internal coding and infrastructure.It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability.White box testing is also known as **clear, open, structural, and glass box testing**.

It is one of two parts of the **"box testing" approach** of software testing. Its counter-part, blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "whitebox" was used because of the see-through box concept. The clear box or whitebox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested

## What do you verify in White Box Testing ?

White box testing involves the testing of the software code for the following:

* Internal security holes
* Broken or poorly structured paths in the coding processes
* The flow of specific inputs through the code
* Expected output
* The functionality of conditional loops
* Testing of each statement, object and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

**How do you perform White Box Testing?**

To give you a simplified explanation of white box testing, we have divided it into **two basic steps**. This is what testers do when testing an application using the white box testing technique:

**STEP 1) UNDERSTAND THE SOURCE CODE**

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

**Step 2) CREATE TEST CASES AND EXECUTE**

The second basic step to white box testing involves testing the application’s source code for proper flow and structure. One way is by writing more code to test the application’s source code. The tester will develop little tests for each process or series of processes in the application. This  method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include manual testing, trial and error testing and the use of testing tools as we will explain further on in this article.

**System testing:**

Once the individual module testing is completed, modules are assembled and integrated to perform as a system. The top down testing, which began from upper level to lower level module, was carried out to check whether the entire system is performing satisfactorily.

There are three main kinds of System testing:

1. Alpha Testing
2. Beta Testing
3. Acceptance Testing

**Alpha Testing:**

This refers to the system testing that is carried out by the test team with the Organization.

**Beta Testing**:

This refers to the system testing that is performed by a selected group of friendly customers

**Acceptance Testing:**

This refers to the system testing that is performed by the customer to determine whether or not to accept the delivery of the system.

**Integration Testing:**

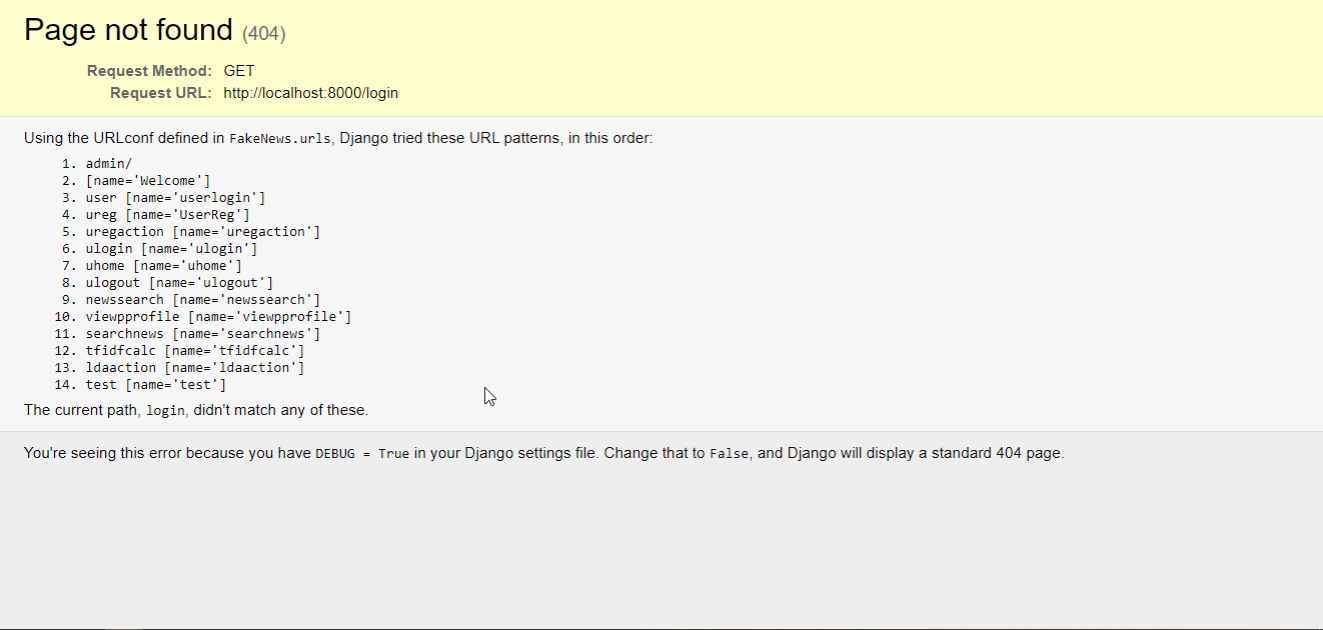
Data can be lost across an interface, one module can have an adverse effort on the other sub functions, when combined, may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. The testing was done with sample data. The developed system has run successfully for this sample data. The need for integrated test is to find the overall system performance.

**Output testing:**After performance of the validation testing, the next step is output testing. The output displayed or generated by the system under consideration is tested by asking the user about the format required by system.

**Unit testing**

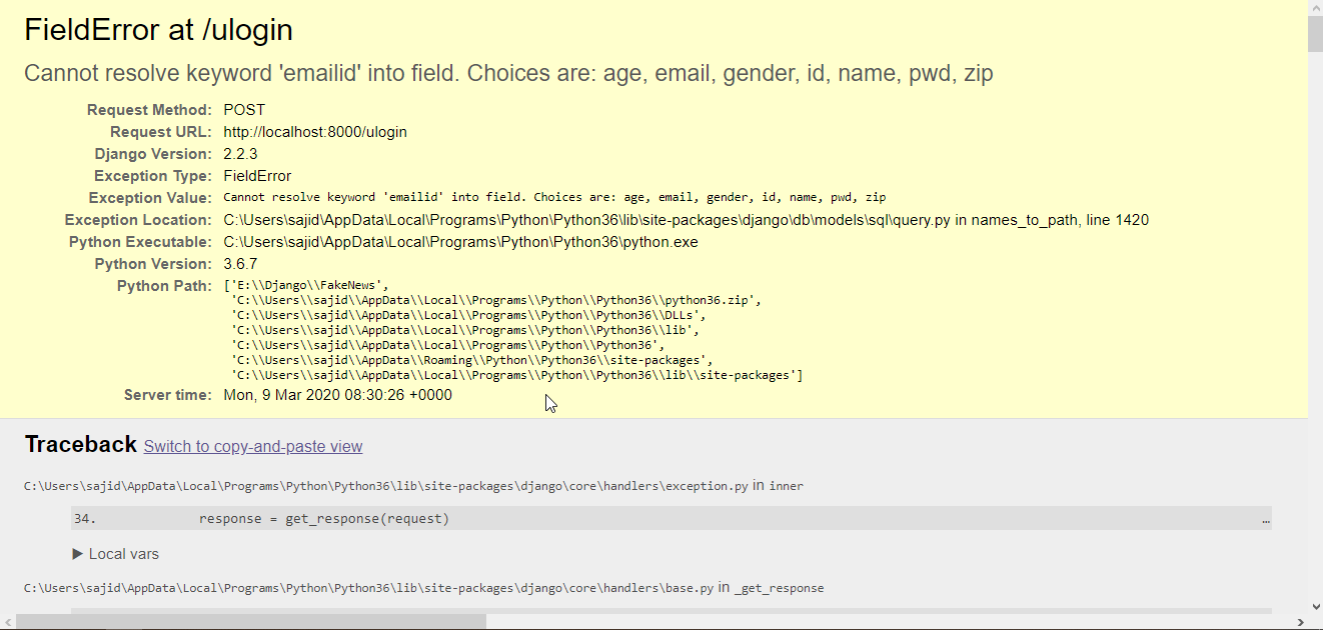
**URL Mismatch Error**

When we give URL like localhost:8000/login and if it’s not match in urls.py files, we can get this error.



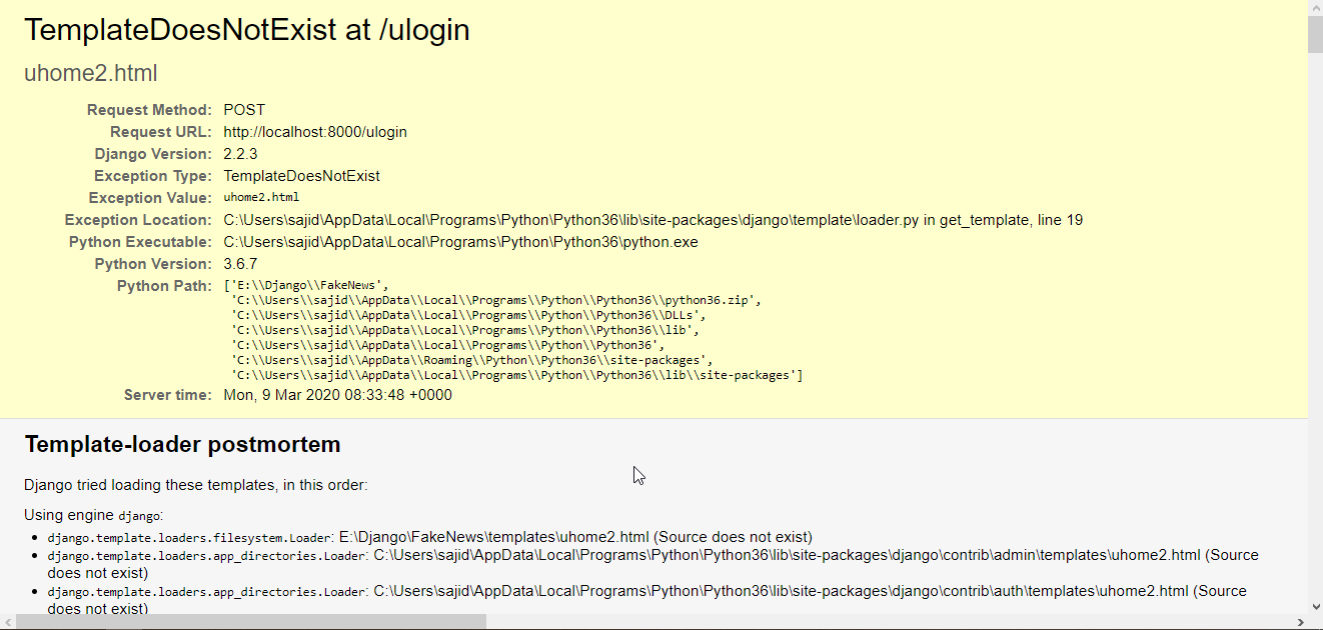
**Field Error**

Database field mismatch from model. Given keyword 'emailid' into field. expected: age, email, gender, id, name, pwd, zip.



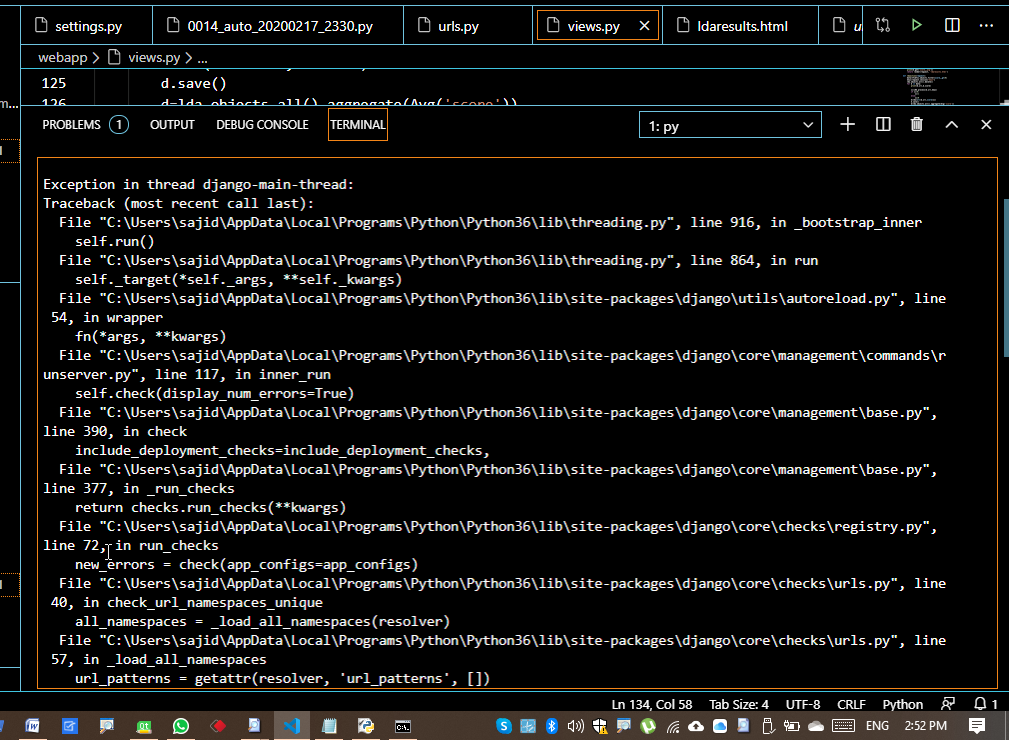
**Template Does Not Exist**

We get this error when URL redirecting and template file (html) not found, or mismatch



**Exception while running server**

If any syntactically mistakes in views.py we get this exception.



**Manual testing on project application**

**TEST CASES**

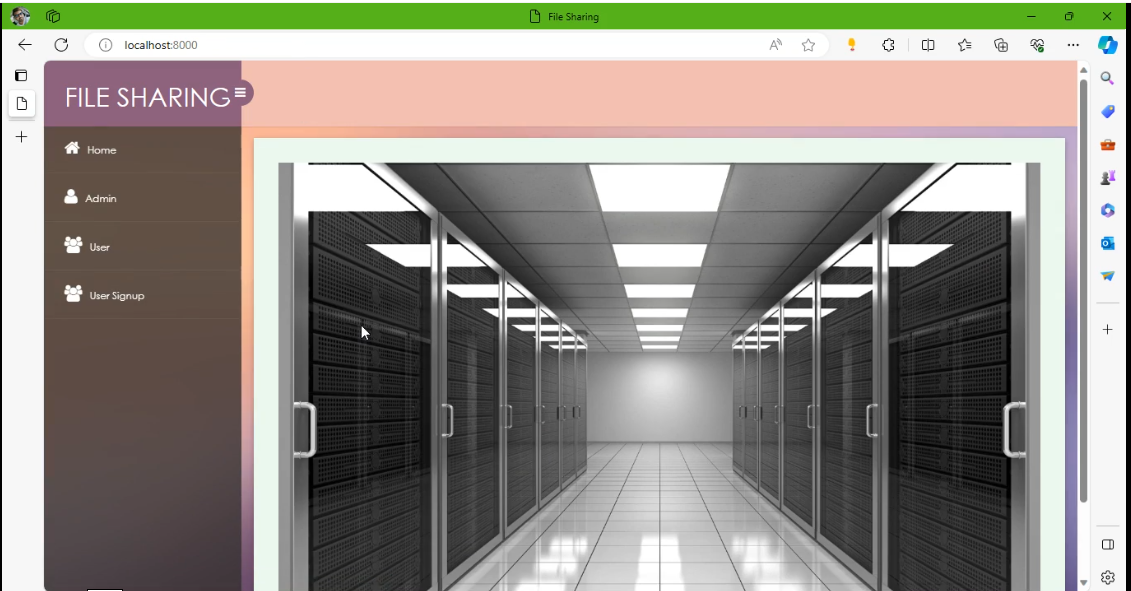
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** #1 | | **Test Case Description** - Validations in Registration Form | | | |
| **S#** | **Prerequisites** | | **S#** | **Test Data Requirement** | |
| 1 | User should be Registered | | 1 | Data should be valid | |
| **Test Condition** | | | | | |
| Entering data in registration form | | | | | |
| **Step #** | **Step Details** | **Expected Results** | | **Actual Results** | **Pass/Fail/Not Executed/Suspended** |
| 1 | User gives First and Last Name | Pop showing email verification message | | Enter valid email/password | Fail |
| 2 | Submitting the form without entering any details | Pop showing email verification message | | Enter email /password | Fail |
| 3 | User enters invalid format of email id | Pop showing email verification message | | Enter valid email id | Fail |
| 4 | User enters a phone number with < 10 digits | Pop showing email verification message | | Enter valid phone number | Fail |
| 5 | Entering valid username and password | Pop showing email verification message | | Pop showing email verification message | Pass |

Table 1 Registration test case

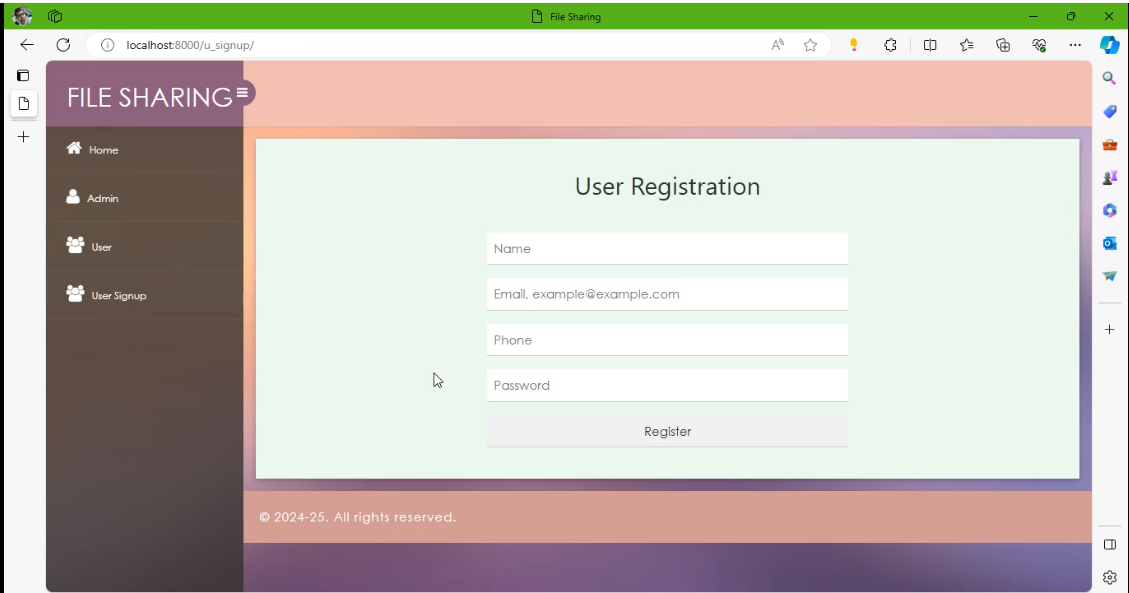
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** #2 | | **Test Case Description** - Validations in Login Form | | | |
| **S#** | **Prerequisites** | | **S#** | **Test Data Requirement** | |
| 1 | User should have an email id | | 1 | Data should be valid | |
| **Test Condition** | | | | | |
| Entering data in login form | | | | | |
| **Step #** | **Step Details** | **Expected Results** | | **Actual Results** | **Pass/Fail/Not Executed/Suspended** |
| 1 | User gives aemail or password of <6 characters | User logged in | | Enter valid email/password | Fail |
| 2 | Submitting the form without entering any details | User logged in | | Enter email /password | Fail |
| 3 | User enters wrong Email and (or) password | User logged in | | Enter correct email /password | Fail |

Table 2 Login test case

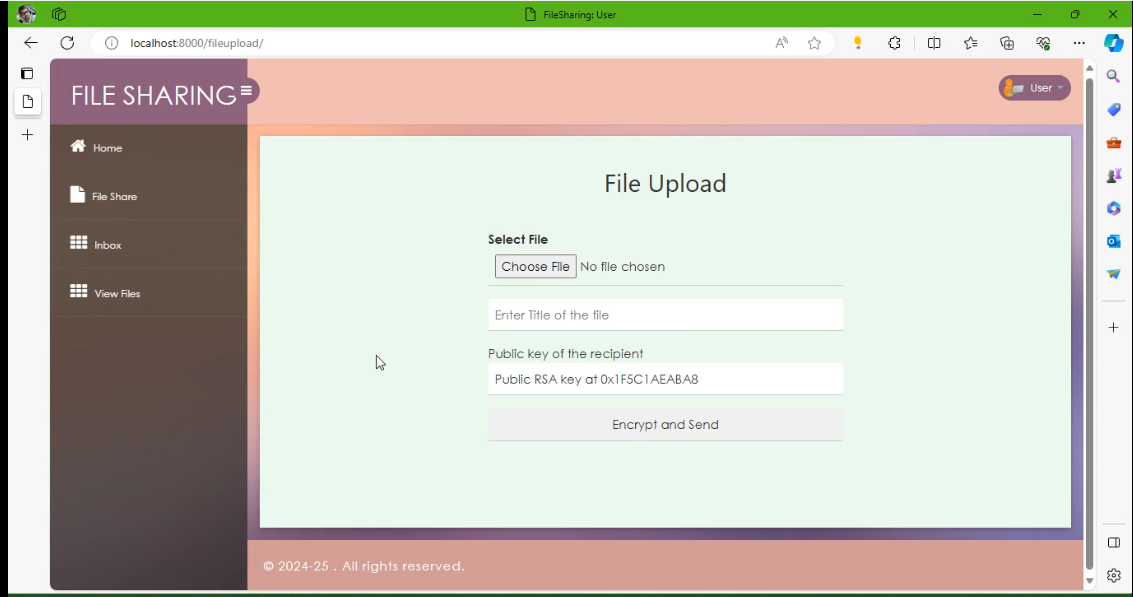
**SCREEN SHORTS**

****

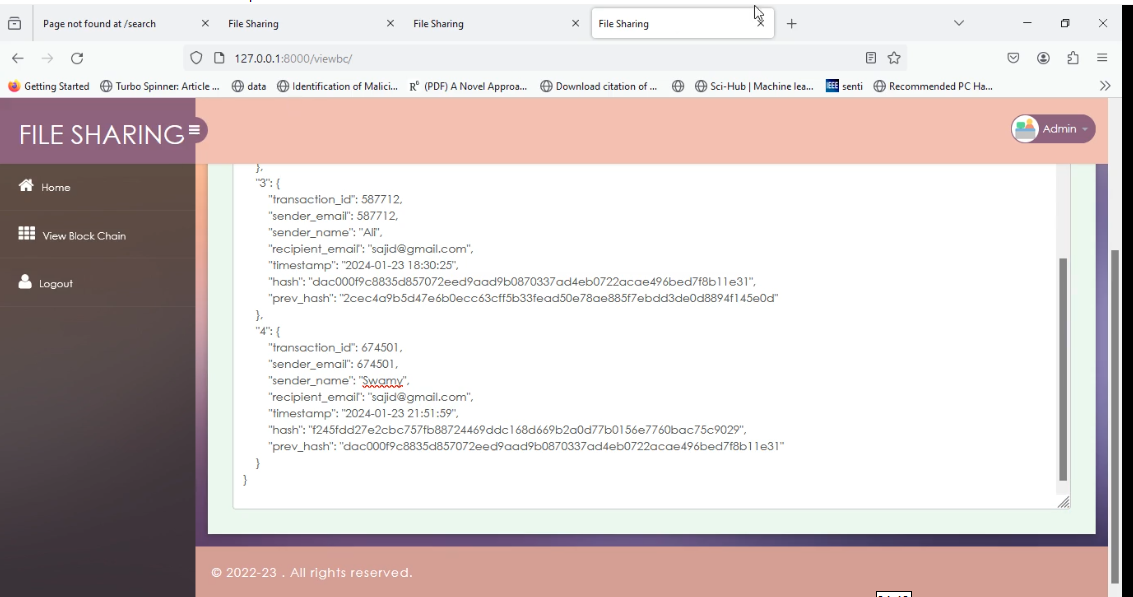
**Home Screen**

****

**User Registration**

****

**Key generation**

****

**View Block Chain**

**CONCLUSION**

The proposed system provides secure file-sharing across a consortium of organizations using blockchain. It provides confidentiality, integrity, and availability of shared files. It ensures end to end encryption of the files. The content ID of the shared file is stored on the blockchain in a tamper resistant way. The encrypted file and file metadata is stored in a distributed fashion on the distributed IPFS storage and blockchain ledger respectively. The system is realized using open source blockchain framework Hyperledger Fabric and tested using Hyperledger Caliper tool.

**REFERENCES**

[1] S. Nakamoto, “bitcoin: A peer-to-peer electronic cash system,” 2008.

[2] ”Hyperledger Fabric Documentation, Release main”, ttps://readthedocs.org/projects/hlf/downloads/pdf/latest/, accessed on Dec 2022

[3] Naz M, Al-zahrani FA, Khalid R, Javaid N, Qamar AM, Afzal MK, Shafiq M. A secure data sharing platform using blockchain and interplanetary file system. Sustainability. 2019 Dec 10;11(24):7054.

[4] Liu J, Li X, Ye L, Zhang H, Du X, Guizani M. BPDS: A blockchain based privacy-preserving data sharing for electronic medical records. In2018 IEEE Global Communications Conference (GLOBECOM) 2018 Dec 9 (pp. 1-6). IEEE.

[5] Satapathy U, Mohanta BK, Panda SS, Sobhanayak S, Jena D. A secure framework for communication in internet of things application using hyperledger based blockchain. In2019 10th international conference on computing, communication and networking technologies (ICCCNT) 2019 Jul 6 (pp. 1-7). IEEE.

[6] YSari L, Sipos M. FileTribe: blockchain-based secure file-sharing on IPFS. InEuropean Wireless 2019; 25th European Wireless Conference 2019 May 2 (pp. 1-6). VDE.

[7] Benet J. Ipfs-content addressed, versioned, p2p file system. arXiv preprint arXiv:1407.3561. 2014 Jul 14.

[8] ”Hyperledger Caliper Documentation, Getting Started”, ”https://hyperledger.github.io/caliper/v0.5.0/getting-started/”, accessed on Dec 2022

[9] ”Apache Couch Database Documentation: Release 3.3.0”, ”https://docs.couchdb.org/ /downloads/en/latest/pdf/”, accessed on Dec 2022

[10] Buterin, Vitalik. ”A next-generation smart contract and decentralized application platform.” white paper 3, no. 37 (2014): 2-1.