Chapter 1. Introduction

1.1 Introduction

This Chapter includes the basic overview of the project and its need in current world, the background and motivation behind this project is followed by the statement of problem and important project modules with their description.

History

The underlying concept of cloud computing dates back to 1950s, when large scale mainframe became available in academia and corporations, accessible via thin clients/terminal computers, often referred to as "dumb terminals", because they were used for communication but had no internal computation capacities. To make more efficient use of costly mainframes, a practice evolved that allowed multiple users to share both the physical access to the computer from multiple terminals as well as to share the CPU time. This eliminated periods of inactivity on the mainframe and allowed for a greater return on the investment. The practice of sharing CPU time on a mainframe become popularly known as time-sharing in the industry.

In the 1990s, telecommunications companies, who previously offered primarily dedicated point-to-point data circuits, began offering virtual private network (VPN) services with more comparable quality of service, but at lower cost. They began to use the cloud symbol to denote the demarcation point between what the providers was responsible for and what users were responsible for. Cloud Computing extends this boundary to cover servers as well as the network infrastructure.

As computers became more prevalent, scientists and technologies explored wats to make large scale computing power available to more users through time sharing experimenting with algorithms to provide the optimal use of the infrastructure, platform and applications with prioritized access to the CPU and efficiency for the end users.

The development of the internet from being document centric via semantic data towards more and more services was described as "Dynamic Web". This contribution focused in particular in the need for better meta-data able to describe not only implementation details but also conceptual details of **model- based** applications. The ubiquitous availability of high-capacity networks, low -cost computers storage devices as well as widespread adoption of hardware virtualization, service -oriented architecture, autonomic, and utility computing have led to tremendous growth in cloud computing.

As related to project REST framework is being used to create API 's that will transfer the data to the front end and Algorithms are developed on them to query results into the front end. The Django REST framework allows us to use methods such as 'GET' 'PUT' 'UPDATE' 'DELETE'. Based on this one can manipulate the main data frame easily.

1.2Problem Definition

– While implementing this project few obstacles are included in this section.

Typically known as Representational State Transfer (REST)- based API that enables machines to interact with cloud software in the same way that a traditional user interface (e.g., a computer desktop) facilitates interaction between humans and computers. REST framework contains parameters that needs to be covered while testing, a specific combination could expose errors, making the API vulnerable to security threats or not functioning at all. Such errors can cause the failure or service operations, which can use an organization to face financial or trust losses.

Another issue is Complex input types, RESTful APIs of today enable communication between hybrid networks, backends, and all types of devices. Sending data is possible anytime, and it can be anything from a single file or collection of files to basic integers and texts. In order to test RESTful APIs, the framework for creating tests must have the ability to infer test input values, which is especially challenging when dealing with complicated types.

Cloud computing that has been envisioned in this project is the key to solve many problems in IT enterprises. But Cloud computing facing following problems like latency issues, Availability, Data Security, Monitoring. To solve these issues, we overcome these problems by using Load Balancing algorithm and using Third Party Auditor, Key-pair exchange for trusted system.

Sr no.	**Problem in Cloud**	**Solution**
1.	Data security	Service Level Agreement (SLA), Encryption, Digital signature
2.	Monitoring	Accountability, Role based Security
3.	Latency issues	Load Balancing (Intra or Inter Cluster)
4.	Availability	Backup Server using Raid 1/Raid 0/ Raid 5

1.3 Proposed Methodology

1.3.1 Proposed approach

SDLC is a process that defines the various stages involved in the development of software for delivering a high-quality product. SDLC stages cover the complete life cycle of software i.e. from inception to retirement of the product. The purpose of SDLC is to deliver a high-quality product which is as per the customer's requirement.

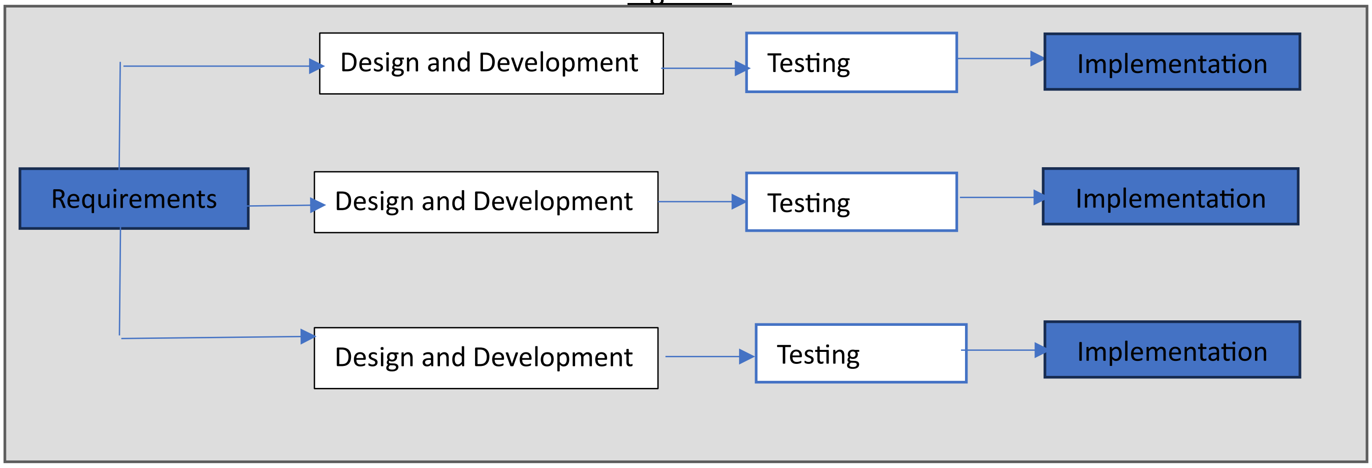
SDLC has defined its phases as Requirement gathering, Designing, Coding, Testing, and Maintenance. It is important to adhere to the phases to provide the Product in a systematic manner.

B. SDLC Model

A software life cycle model is a descriptive representation of the software development cycle. The software development model helps the developer to select a strategy to develop the software. A software development model has its own set of tools, methods and procedures, which are expressed clearly and defines the software development life cycle. This project has been developed using the Iterative model. In this life cycle model, a Project Control List (PCL) on the basis of current known requirements is developed. A PCL is a list containing the series of tasks/functionalities that are to be present in the given system. If at a certain phase of development, we come across any new requirement, we add it to our Project Control List.

For developing the project, a task is chosen from the given PCL and Planning, Analysis, Designing, Testing and Evaluation is performed as shown in Figure 1. When the specific functionality is added we remove it from the Project Control List. In a similar way, one task at a time from PCL is chosen, implemented and then removed from PCL. This process iterates

Figure 1



until the desired requirements of the product are not met. After each iteration, the management team can do work on risk management and prepare for the next iteration. Because a cycle includes a small portion of the whole software process, it is easier to manage the development process. In the Iterative model, the newer iterations are incrementally improved versions of previous iterations. Moreover, in the event that a new iteration fundamentally breaks a system in a catastrophic manner, a previous iteration can quickly and easily be implemented or "rolled back," with minimal losses, which is a boon for post-release maintenance. In the Iterative Model, the initial run-through of all stages may take some time, but each subsequent iteration will be faster and faster, allowing the life cycle of each new iteration to be trimmed down to a matter of days or even hours in some cases.

1.2.2 Proposed methodology

This project is developed on Django framework, the backend is built on python, no SQL database such as MongoDb. The frontend consists of HTML and CSS. The project developed is highly efficient, user-friendly and simple.

The Project is developed via multiple steps. The major steps are enlisted here:

1. Installing Python and adding it to the windows path.

2. Creation of Virtual Environment (Following commands are written in Command Prompt) - pip install virtualenvwrapper-win - mkvirtualenvenvironmentname (any name can be given) - workonenvironmentname

3. Installing Django - pip install Django

4. Go to Destination Place where you want the project to be kept, using cd command.

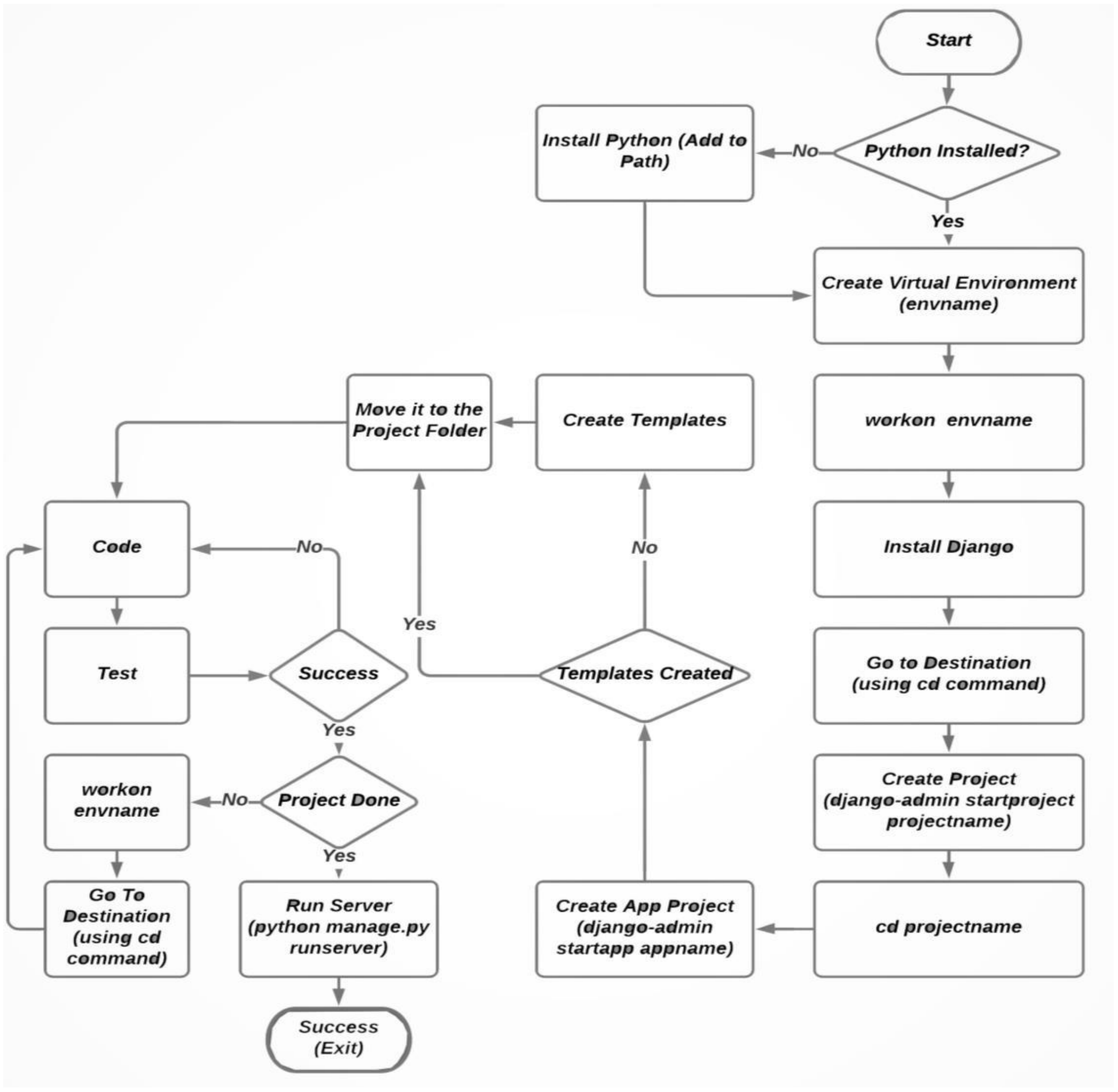
5. Create Project as follows - django-admin startprojectsomeprojectname - cd someprojectname

6. Create App of the project as - django-admin startappappname - python manage.py makemigrations - python manage.py migrate

7. Copy the Template Folder (if Front End Template is downloaded) to the project folder created.

8. Run Server (localhost:8000) - python manage.py runserver

Figure 2


Start
Python Installed?
No
Install Python (Add to Path)
Yes
Create Virtual Environment (envname)
workon envname
Install Django
Go to Destination (using cd command)
Create Project (django-admin startproject projectname)
cd projectname
Create App Project (django-admin startapp appname)
Templates Created
No
Create Templates
Move it to the Project Folder
Yes
Code
Test
Success
No
Yes
Project Done
No
workon envname
Go To Destination (using cd command)
Yes
Run Server (python manage.py runserver)
Success (Exit)

The Flow Chart (Figure 2) illustrates the steps that are required in order to install the prerequisites of the project and then the steps involved in the development of project. Commands written in bracket are to be run on Command Prompt. They are the steps that are required to install virtual environment and to run local server on the system project is to be developed on. The code of backend and frontend can be coded in any code editor (Sublime Text was used in this project). All the changes that were made in the project can be seen on the local server. The Data was stored on online cloud service Cloudinary.

1.4 Review of literature

This chapter includes the literature done on the project "eLearning platform" based on cloud-computing through various sources, in order to understand technical details. In this we are covering Cloud security, accountability and load balancing with a meaningful craft done on the data stored in cloud. Data can be accessed by multiple users. So here we are using role -based security to protect our data and also keep record usage of data. Another issue is cloud security and load balancing. Data Security is important because of two kinds of attack may be possible. One is passive attack and other is Active attack. Passive attack we can prevent using encryption, Active attack easily detected using digital signature. This shall form the theoretical framework of the thesis.

Various websites and research papers have been developed keeping the idea of educational resource sharing and its importance in mind. Few of them are listed here:

1. Used Books Factory: It is an online platform to sell old books of different categories.

2. Vioric-Torri, C. & Alexandrache, C. (2012): The study reflects how educational technology influences the learning styles of students and how to form and develop the competences of learning in the new generations.

3. TutorialsPoint: The website provides tutorials on different topics related to computer science and technology. Provides pdf notes for the same and also provides guidance for competitive exams.

4. The Physics Classroom: For PDF notes and tutorials related to the various fields of Physics.

5. Kelly, L., & Breault, K. (2006): The objective of the research project was to provide the Australian Museum with guidance on how to best develop a website that meets the needs of students and teachers in the primary and secondary levels across a range of curriculum areas. General objectives were to gain insights into how students and teachers are using the internet and what they are looking for when they access websites.

6. Aglasem: Online Portal that provides previous year question papers and answer keys related to different competitive exams and some universities‟ semester papers.

After a brief study of the related works, it has been observed that all these websites possess different functions of the project proposed but none of them have all of the features collectively. Also, there is no such website where sharing class notes can be done except for social media. The following project has been developed keeping all these disadvantages in mind.

Chapter 2 : Survey Of Technology

2.1 Domain: Software Development

In this project the main aim is to create a software that is unique and trending in current world. The domain is however is just a preface of what software developers do. To keep in mind the main objective behind the software development is to find its way to the current market needs. One can say that if a software will find its way if it follows a several key factors that are considered important. These include:

Usability: The software should be easy to use and understand for its intended audience.

Reliability: The software should work as expected and be free of bugs or errors.

Scalability: The software should be able to handle an increasing amount of users or data.

Security: The software should protect user data and be free of vulnerabilities.

Maintainability: The software should be easy to update and maintain over time.

Ultimately, the most important aspect of creating software will vary depending on the specific project and its goals.

Certainly! Let's delve deeper into each of these important factors when creating software:

Usability: Usability refers to how easily and efficiently users can interact with the software. A user-friendly interface, intuitive navigation, and clear instructions contribute to a positive user experience. Understanding the target audience and their needs is crucial for designing software that is intuitive and easy to use.

Reliability: Reliability is a fundamental aspect of software quality. It means that the software consistently performs its intended functions correctly, without unexpected errors or crashes. Quality assurance processes such as testing, debugging, and code reviews help identify and fix issues, ensuring that the software functions reliably.

Scalability: Scalability refers to the software's ability to handle increasing workloads or accommodate a growing number of users, without compromising performance. Designing software with scalability in mind allows it to adapt and handle increased demand. This may involve optimizing code, utilizing distributed systems, or employing cloud technologies.

Security: Software security involves protecting the software and its users' data from unauthorized access, data breaches, and malicious attacks. Implementing robust security measures such as encryption, authentication protocols, access controls, and regular security audits helps safeguard sensitive information and maintain user trust.

Maintainability: Maintainability focuses on how easily the software can be updated, enhanced, or fixed over time. Well-structured code, clear documentation, and adherence to coding best practices contribute to maintainability. This aspect is crucial for long-term software sustainability, as it allows developers to efficiently make changes and address issues as they arise.

While these factors are important individually, they are often interconnected. For example, an emphasis on usability can enhance user satisfaction and adoption, which in turn affects the software's reliability and success. Similarly, ensuring security and scalability contribute to user trust and confidence in the software.

Moreover, it's important to note that the specific priorities and emphasis on these factors may vary depending on the nature of the software being developed. For instance, a banking application would place a higher emphasis on security, while a social media platform may prioritize usability and scalability. Ultimately, successful software development involves striking a balance among these factors based on the specific requirements and goals of the project.

2.2 Domain : Cloud computing , information security and full stack development

Cloud computing along with information security offer a range of benefits that are related to each other in a such a way it makes highly useful in software development. Here are some key benefits of leveraging cloud computing in software development:

Scalability and Flexibility: Cloud platforms provide the ability to scale computing resources up or down based on demand. This allows developers to easily accommodate varying workloads and handle spikes in traffic without investing in and managing physical infrastructure. Scaling resources in the cloud helps ensure optimal performance and cost-efficiency.

Cost Savings: Cloud computing eliminates the need for upfront hardware and infrastructure investments. Instead, developers can leverage cloud services on a pay-as-you-go model, only paying for the resources they use. This significantly reduces upfront costs, especially for small teams or startups, and allows for better budget management.

Collaboration and Teamwork: Cloud platforms enable seamless collaboration among developers and teams, regardless of their physical location. Shared development environments, version control systems, and real-time collaboration tools simplify teamwork and foster efficient development processes. Developers can work simultaneously on the same project, easily share code, and collaborate on debugging and troubleshooting.

Reduced Time to Market: Cloud computing accelerates software development by providing readily available infrastructure, development tools, and services. Developers can leverage pre-built components, APIs, and frameworks offered by cloud providers, reducing development time and speeding up the overall software delivery process. This enables faster time to market, allowing businesses to stay competitive and respond to market demands quickly.

High Availability and Reliability: Cloud platforms offer robust infrastructure and redundancy mechanisms that ensure high availability and reliability of software applications. Service-level agreements (SLAs) provided by cloud providers guarantee uptime and availability, minimizing the risk of application downtime and improving user experience.

Easier Deployment and Management: Cloud platforms simplify the deployment and management of software applications. Developers can utilize automated deployment tools, continuous integration and delivery (CI/CD) pipelines, and infrastructure-as-code (IaC) practices to streamline the deployment process. Cloud-based management consoles and monitoring tools facilitate easy monitoring, logging, and troubleshooting of applications.

Global Reach and Scalable Architecture: Cloud platforms have a global presence with data centers located in various regions. This enables software developers to deploy their applications closer to their target audience, reducing latency and improving performance. Additionally, cloud architectures can be designed to span multiple regions, providing high availability and scalability across geographically distributed users.

These advantages make cloud computing an attractive choice for software development, enabling faster development cycles, cost savings, increased collaboration, and improved scalability and reliability. Cloud platforms provide a robust foundation for building, testing, deploying, and managing software applications, allowing developers to focus more on their core competencies rather than infrastructure management.

2.3 Hardware Requirements

The project developed satisfies all the functional and nonfunctional

requirements enlisted. The following specifications

are required for the project to run on any device.

1) System Specifications

Processor: Intel(R) Core (TM) ie-5005U CPU @ 2.00GHz

RAM: 2 GB

System Type: 32-bit/64-bit operating system, x32 or x64

based processor

Operating System: Windows 7/8/10.

2.4 Software Requirements

The minimum software requirements to run this project includes:

Web Browser: You need a web browser to access and view web applications. Common web browsers include Google Chrome, Mozilla Firefox, Microsoft Edge, Safari, and others. Many web applications are designed to work on various web browsers, so you have some flexibility in your choice.

Internet Connection: A stable internet connection is necessary to access web applications hosted online. The speed of your internet connection will affect the application's loading times and responsiveness.

Device: You can view web applications on a wide range of devices, including desktop computers, laptops, tablets, and smartphones. The display size and capabilities of your device may impact the user experience, especially for applications with complex layouts or multimedia content.

Operating System: Most web applications are platform-agnostic and can be accessed from various operating systems, including Windows, macOS, Linux, Android, and iOS.

JavaScript Enabled: Many modern web applications heavily rely on JavaScript for interactivity and dynamic content. Ensure that JavaScript is enabled in your web browser to experience the full functionality of the application.

Screen Resolution: Some web applications may require a minimum screen resolution to display their content properly. This requirement varies depending on the application's design and responsiveness.

These are the basic requirements to this project. However, for a seamless experience, you may want to keep your web browser up to date, have a faster internet connection for faster loading times, and make sure JavaScript is enabled, as many web applications use client-side scripting for interactivity.

Chapter 3 System design requirements and analysis

3.1 Feasibility Study

It is feasible to develop an e-learning system. In fact, e-learning systems have become increasingly popular and widely adopted in recent years. An e-learning system provides a digital platform for delivering educational content, conducting assessments, and facilitating interactions between instructors and learners. Here are some key considerations for developing an e-learning system:

Functionalities: The functionalities and features that   includes in your e-learning system may make it feasible. This may include course management, content delivery, assessments, progress tracking, discussion forums, multimedia support, and communication tools.

User Interface and Experience: Design an intuitive and user-friendly interface that allows learners to easily navigate through the system, access course materials, submit assignments, and engage with interactive content. Consider the user experience across different devices, such as desktops, tablets, and mobile devices.

Content Management: Develop a robust content management system (CMS) that enables instructors to create, organize, and manage course materials. This may involve supporting various content formats (text, video, audio, presentations), version control, and the ability to easily update and modify course content.

Assessment and Progress Tracking: Implement assessment features that allow instructors to create and manage quizzes, assignments, and exams. Provide learners with feedback and progress tracking mechanisms to monitor their performance and identify areas for improvement.

User Management: Develop user management features that enable learners and instructors to create and manage their profiles, enroll in courses, and track their progress. Implement authentication and authorization mechanisms to ensure secure access to the system.

Collaboration and Communication: Incorporate communication and collaboration tools to facilitate interaction between instructors and learners. This may include discussion forums, chat features, messaging systems, and virtual classrooms for live sessions and webinars.

Scalability and Performance: Consider the scalability requirements of the e-learning system, as it may need to handle a large number of concurrent users and course enrollments. Design the system architecture to be scalable and ensure that it can handle increased traffic and user demand.

Security and Privacy: Pay attention to the security of the e-learning system, as it may handle sensitive user data and educational content. Implement robust security measures, such as encryption, secure authentication, role-based access control, and regular security audits, to protect user information.

Integration and Interoperability: Consider integrating the e-learning system with other platforms or learning management systems (LMS) to leverage existing resources and tools. This may include integration with payment gateways, content delivery networks (CDNs), third-party applications, or learning analytics systems.

Testing and Quality Assurance: Conduct thorough testing throughout the development process to ensure the reliability, performance, and usability of the e-learning system. Implement automated testing, user acceptance testing, and regularly gather feedback from instructors and learners to improve the system.

Developing an e-learning system requires a multidisciplinary approach, involving software development, instructional design, and user experience considerations. It is important to conduct thorough planning, gather requirements, and engage with instructors and learners during the development process to create an effective and engaging e-learning platform.

3.2 Operational Feasibility

Operating an e-learning website is feasible and has proven to be successful for many organizations and individuals. E-learning websites provide a platform for delivering educational content, conducting courses, and facilitating learning interactions online. Here are some key considerations by which operating an e-learning website is feasible.

Content Management: Ensuring a robust content management system (CMS) in place to organize and manage your educational content. This includes creating and updating course materials, managing multimedia assets, and organizing content hierarchically for easy navigation.

Course Administration: Implementing features that allow you to manage courses efficiently. This may include creating and scheduling courses, enrolling students, tracking progress, managing assessments, and providing instructor support.

User Management: Set up user management functionalities to enable learners to create accounts, enroll in courses, and manage their profiles. Implement authentication and authorization mechanisms to ensure secure access to course resources.

Interactive Learning Tools: Offer a variety of interactive learning tools to engage learners. This can include discussion forums, chat features, quizzes, assignments, video lectures, virtual classrooms, and collaborative projects. Consider integrating tools like video conferencing, screen sharing, and whiteboarding for live sessions and real-time interactions.

Responsive Design: Ensure your e-learning website is responsive and accessible across various devices, including desktops, tablets, and mobile devices. Responsive design enables learners to access course materials and participate in learning activities from their preferred devices.

Scalability and Performance: As your e-learning platform grows, ensure that your infrastructure can handle increased traffic and user demand. Utilize cloud-based services and scalable hosting solutions to maintain optimal performance and accommodate growing numbers of learners.

Support and Communication: Provide effective support channels for learners and instructors. This can include a help center, FAQs, support tickets, and direct communication channels like email or live chat. Encourage instructors to be responsive to learner questions and provide timely feedback.

Analytics and Reporting: Implement analytics and reporting features to track learner progress, assess course effectiveness, and gather insights for continuous improvement. Analytics can help identify areas where learners may be struggling, optimize course content, and measure learner engagement.

Security and Privacy: Pay careful attention to security measures to protect user data, including personal information and course progress. Implement secure authentication, data encryption, and regular security audits to ensure the privacy and confidentiality of user information.

Operating an e-learning website requires ongoing monitoring, maintenance, and continuous improvement. Regularly update course content, fix bugs, and incorporate user feedback to enhance the learning experience. Engage with learners and instructors to understand their needs and provide a supportive and engaging online learning environment.

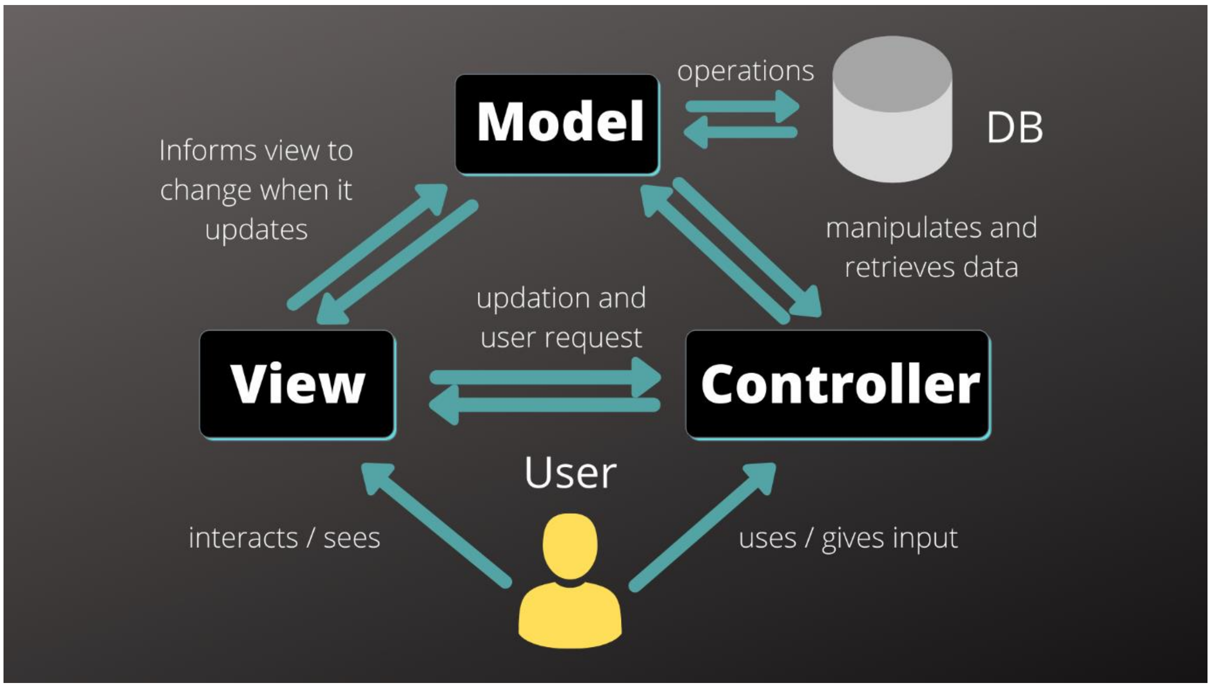
While operating an e-learning website requires effort and resources, it can be a rewarding venture, enabling education to reach a wider audience and providing learners with flexible and accessible learning opportunities.

3.3 Technical Feasibility

One should Keep in mind that the technical feasibility of designing an e-learning platform depends on factors such as the complexity of the desired features, the available development resources, and the expertise of the development team. However, with proper planning, utilizing appropriate technologies, and following best practices, it is technically feasible to design and develop a successful e-learning website. In this project we have proposed aspects that are technically feasible and easy to learn.

1. Front end development: Designing UI and UX by using HTML, CSS and popular JAVA framework REACT. Although we are not using REACT, one can consider it as good option for frontend developers
2. Back-end development: Implementing server -side logic on Python based framework such as Django proves that it is feasible to design the logic -based model view controller.

Feasible study of Django based application.


operations
Model
DB
Informs view to change when it updates
manipulates and retrieves data
updation and user request
View
Controller
User
interacts / sees
uses / gives input

Allowing to create Models separately it is easy to implement models and update them when needed. View sections handles the graphical part of application. As mentioned in methodology, Django is a widely used and highly feasible web framework for building web applications, including eLearning platforms. Here are some reasons why Django is a feasible choice:

Robust and Mature: Django is a mature and well-established framework that has been used in countless web applications, including eLearning platforms. It benefits from years of development and refinement.

Rapid Development: Django follows the "batteries-included" philosophy, providing a wide range of built-in features and libraries that make development faster and more efficient.

Scalability: Django can handle the development of both small and large applications. With proper architecture and optimization, it can scale to accommodate a growing user base.

Security: Django has built-in security features, such as protection against common web vulnerabilities like cross-site scripting (XSS), cross-site request forgery (CSRF), and SQL injection. It also provides a user authentication system.

Community and Documentation: Django has a large and active community of developers and a wealth of documentation, tutorials, and third-party packages available, making it easier to find solutions and support.

Modularity: Django's modularity and flexibility allow you to choose the components you need and integrate with third-party libraries when necessary.

Database Support: Django supports various databases, including PostgreSQL, MySQL, SQLite, and Oracle, giving you the flexibility to choose the database that best suits your needs.

RESTful APIs: Building RESTful APIs for mobile apps or third-party integrations is straightforward with Django, thanks to the Django REST framework.

Internationalization and Localization: Django provides tools for building multilingual and localized web applications, which is important for eLearning platforms serving a global audience.

Admin Interface: Django includes an admin interface that can save time and effort in managing application data and content.

Testing Support: Django includes a testing framework that makes it easier to write and run tests to ensure the reliability and stability of your application.

Cost-Effective: Using an open-source framework like Django can be cost-effective, as it eliminates the need for expensive licensing fees associated with proprietary software.

Cross-Platform: Django can be deployed on various operating systems and cloud platforms, making it versatile and accessible.

3.4 Economic Feasibility

Running an eLearning web application on the cloud can be economically feasible for many organizations, but the feasibility depends on various factors, including the specific requirements of your eLearning application, budget, and how you manage your cloud resources. Here are some considerations:

Scalability: Cloud services are known for their scalability. You can easily scale up or down based on demand. This can be more cost-effective than investing in on-premises infrastructure that might be underutilized or not handle spikes in usage.

Pay-as-You-Go Pricing: Cloud providers typically offer a pay-as-you-go pricing model, which means you only pay for the resources you actually use. This can be more cost-effective than upfront capital investments in hardware and infrastructure.

Resource Optimization: Efficient resource management and optimization are crucial for cost savings. You need to monitor your cloud usage, right-size instances, and leverage auto-scaling features to minimize costs.

Total Cost of Ownership (TCO): Consider the total cost of ownership, which includes not only infrastructure costs but also licensing, development, maintenance, and support. Compare this with the ongoing operational costs of running the eLearning application in the cloud.

Data Transfer Costs: Be mindful of data transfer costs, especially if your eLearning application involves a lot of multimedia content. Data transfer can add up, depending on your cloud provider's pricing structure.

Resource Efficiency: Optimize the use of cloud resources by stopping or de-provisioning instances when they are not in use. Many cloud providers offer tools for cost optimization.

Vendor Selection: Choose a cloud provider that aligns with your budget and requirements. Different providers may offer different pricing models, and their cost structures can vary.

Cost Management Tools: Make use of cost management and monitoring tools provided by cloud vendors to track and control expenses.

Long-term Commitments: Some cloud providers offer discounts for long-term commitments, like one- or three-year reserved instances. Consider whether such commitments are suitable for your application.

Hybrid or Multi-Cloud Strategies: In some cases, a hybrid or multi-cloud strategy may provide cost advantages. You can use cloud for scalability and on-premises infrastructure for stable workloads.

Content Delivery Networks (CDNs): To reduce data transfer costs and improve the performance of your eLearning application, consider using CDNs to cache and serve content from edge servers.

Licensing Costs: Licensing costs associated with the software and services you use on the cloud. These costs can add up.

In summary, the economic feasibility of running an eLearning web application on the cloud depends on how you manage your resources, your usage patterns, and the cloud provider you choose. With proper planning and cost management, many organizations find that cloud hosting offers the flexibility and cost-efficiency they need for eLearning applications.

3.5 Legal Feasibility

The legal feasibility of running an eLearning platform is a complex topic that involves various legal considerations and compliance requirements. The specific legal requirements and challenges you may face can vary by jurisdiction and the nature of your eLearning platform. Here are some key legal aspects to consider:

Intellectual Property Rights: It ensures the legal rights to use all content on eLearning platform, whether it's created by authority, licensed from others, or contributed by users. Respect copyright, trademark, and other intellectual property laws.

Privacy and Data Protection: Compliance with data protection laws, such as the General Data Protection Regulation (GDPR) in the European Union or the Children's Online Privacy Protection Act (COPPA) in the United States, is essential. You must handle personal data responsibly, obtain user consent, and provide clear privacy policies.

Accessibility: Many countries have laws and regulations mandating that websites and online platforms be accessible to individuals with disabilities. Ensuring your eLearning content is accessible can help you avoid legal issues related to discrimination.

Contractual Agreements: When dealing with users, instructors, or partners, it's important to have clear and enforceable terms and conditions, privacy policies, and user agreements. These documents should outline the rights and responsibilities of all parties involved.

Copyright and Licensing: If you use third-party content, ensure you have the proper licenses. Create a clear policy regarding how user-generated content is licensed and used on your platform.

Payment Processing: If you charge for your eLearning courses, you must comply with payment processing regulations and secure sensitive payment information in accordance with Payment Card Industry Data Security Standard (PCI DSS) requirements.

Cybersecurity and Data Breach Preparedness: Implement security measures to protect user data and have a plan in place in case of a data breach. Many jurisdictions require prompt disclosure of data breaches.

Anti-Discrimination Laws: Be aware of and comply with anti-discrimination laws, such as the Americans with Disabilities Act (ADA) in the U.S., which may apply to online education services.

Content Review and Moderation: Implement content review and moderation policies to prevent the dissemination of illegal or inappropriate content on your platform.

Export Controls: eLearning platform which involves the international transfer of technology, to be aware of export control regulations that restrict certain types of content or interactions can solve further problems.

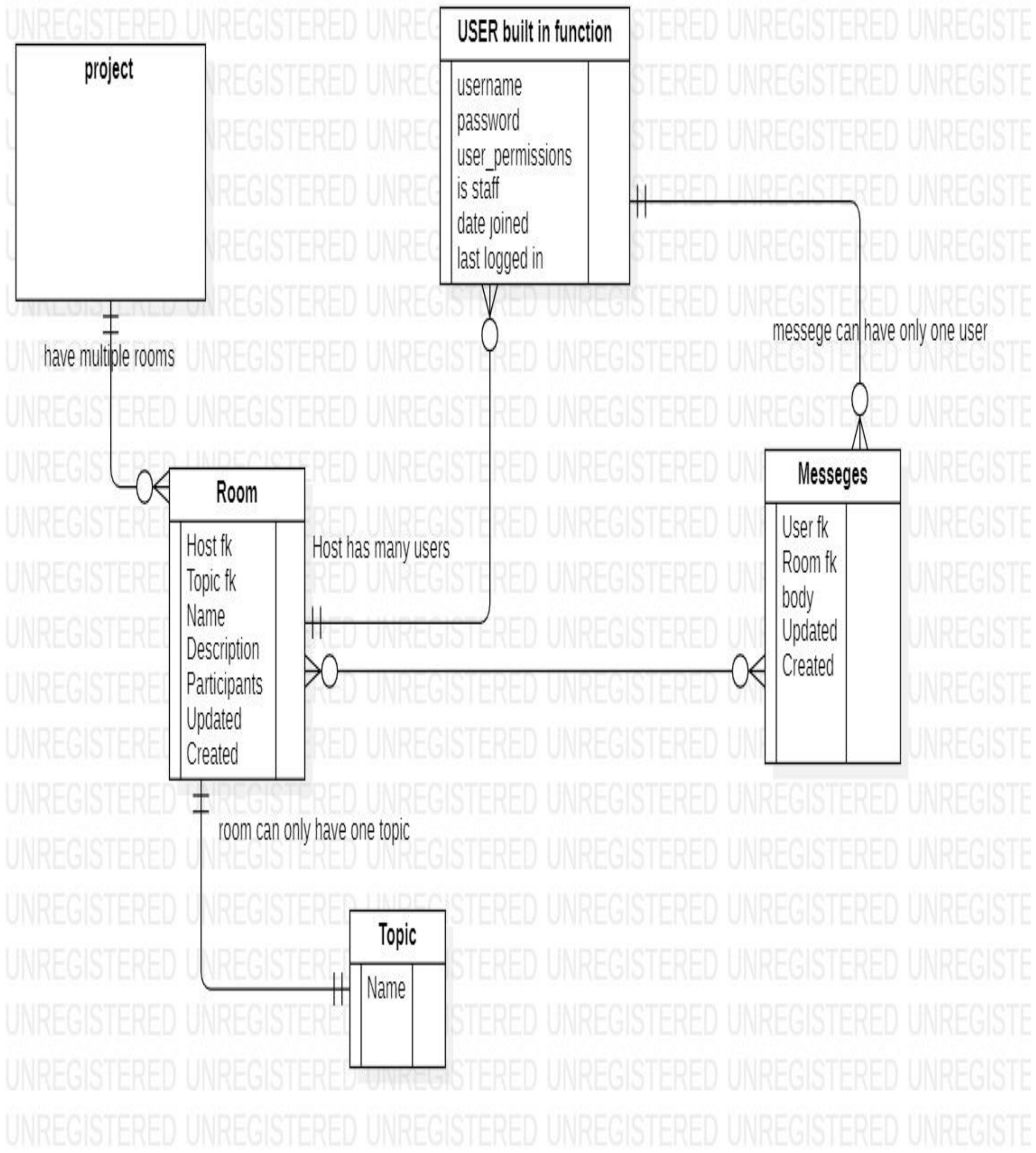
Online Learning Regulations: Some countries have specific regulations governing online learning institutions and services.

International Jurisdiction: eLearning platform serves users in multiple countries and there is a need to address the complexities of international jurisdiction and cross-border legal issues.

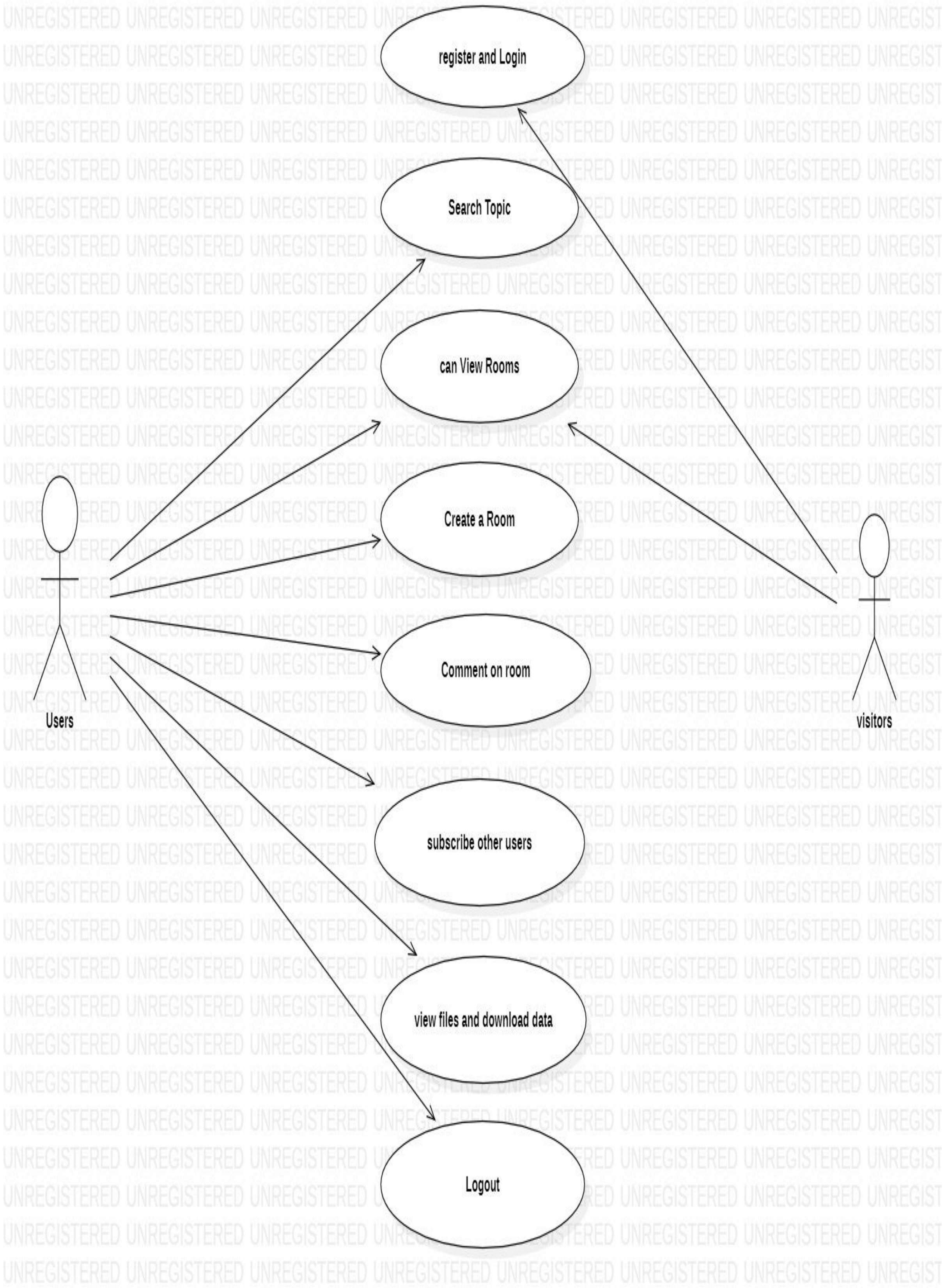
It's essential to consult with legal experts who are knowledgeable about the legal requirements in specific jurisdiction and who can help create policies and procedures that ensure legal compliance. Establishing a clear legal framework and maintaining a strong focus on compliance is critical to the long-term success and sustainability of eLearning platform.

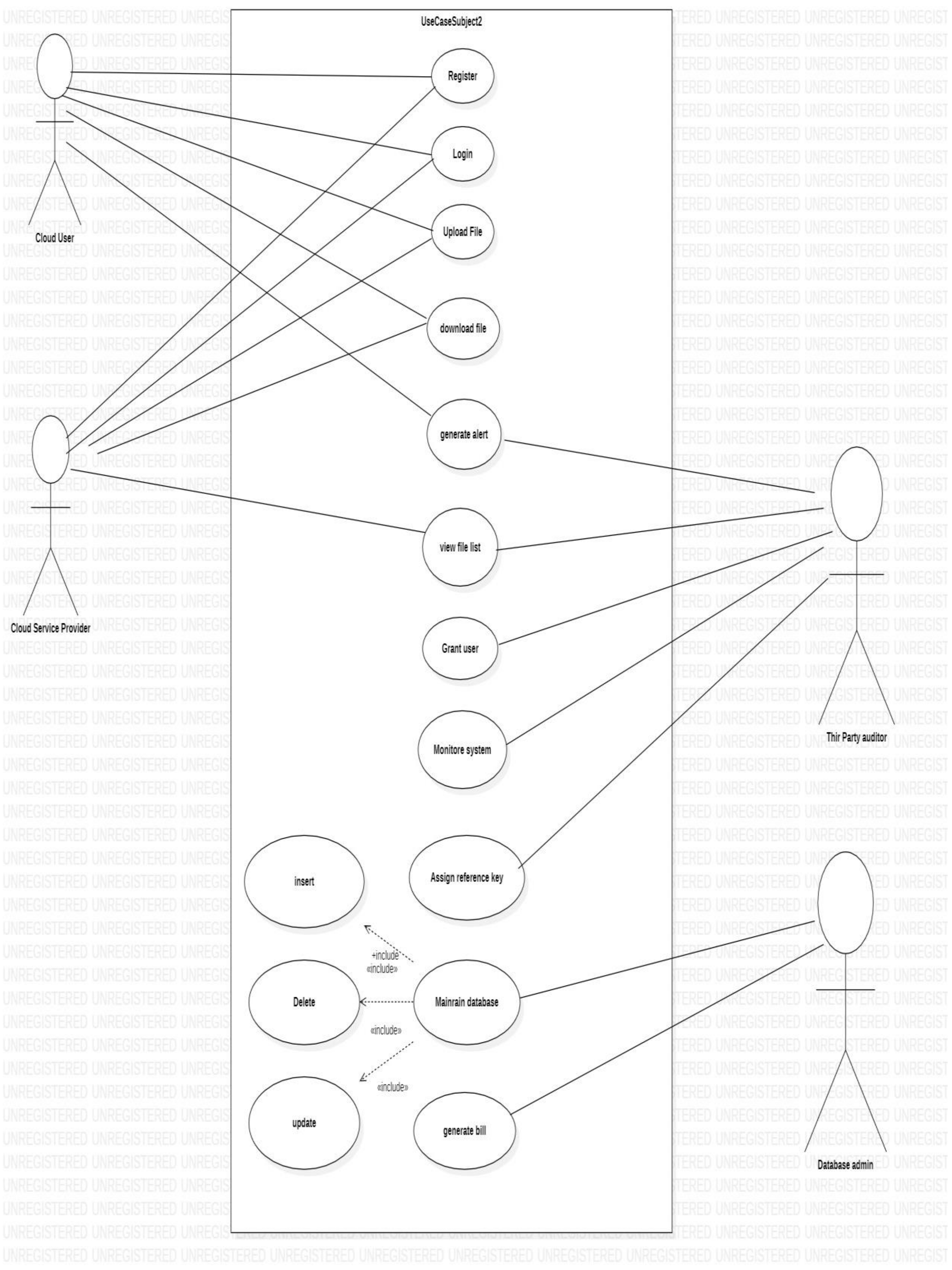
Chapter 4 System Design

4.1 ER -Diagram

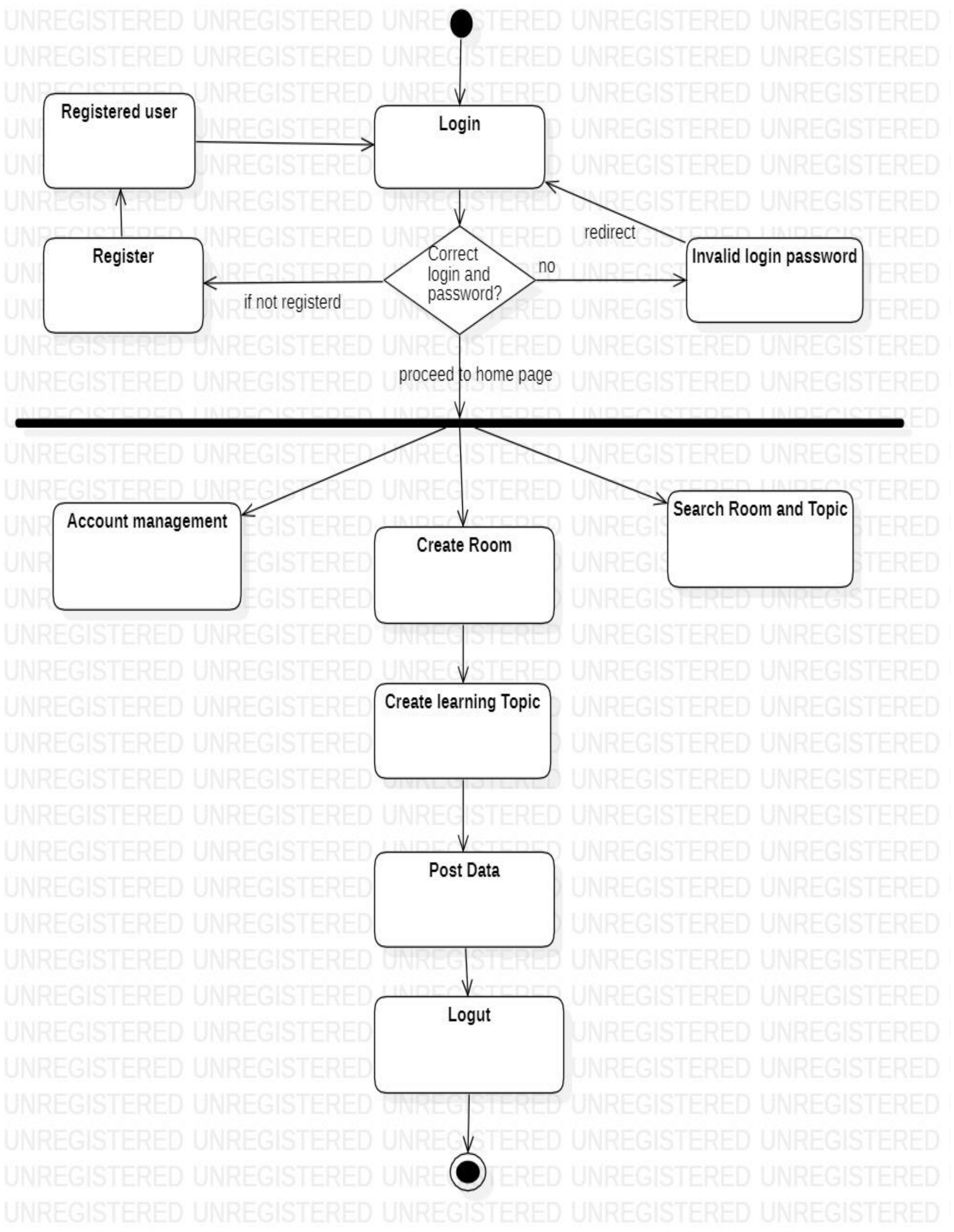


4.2 Use Case Diagram

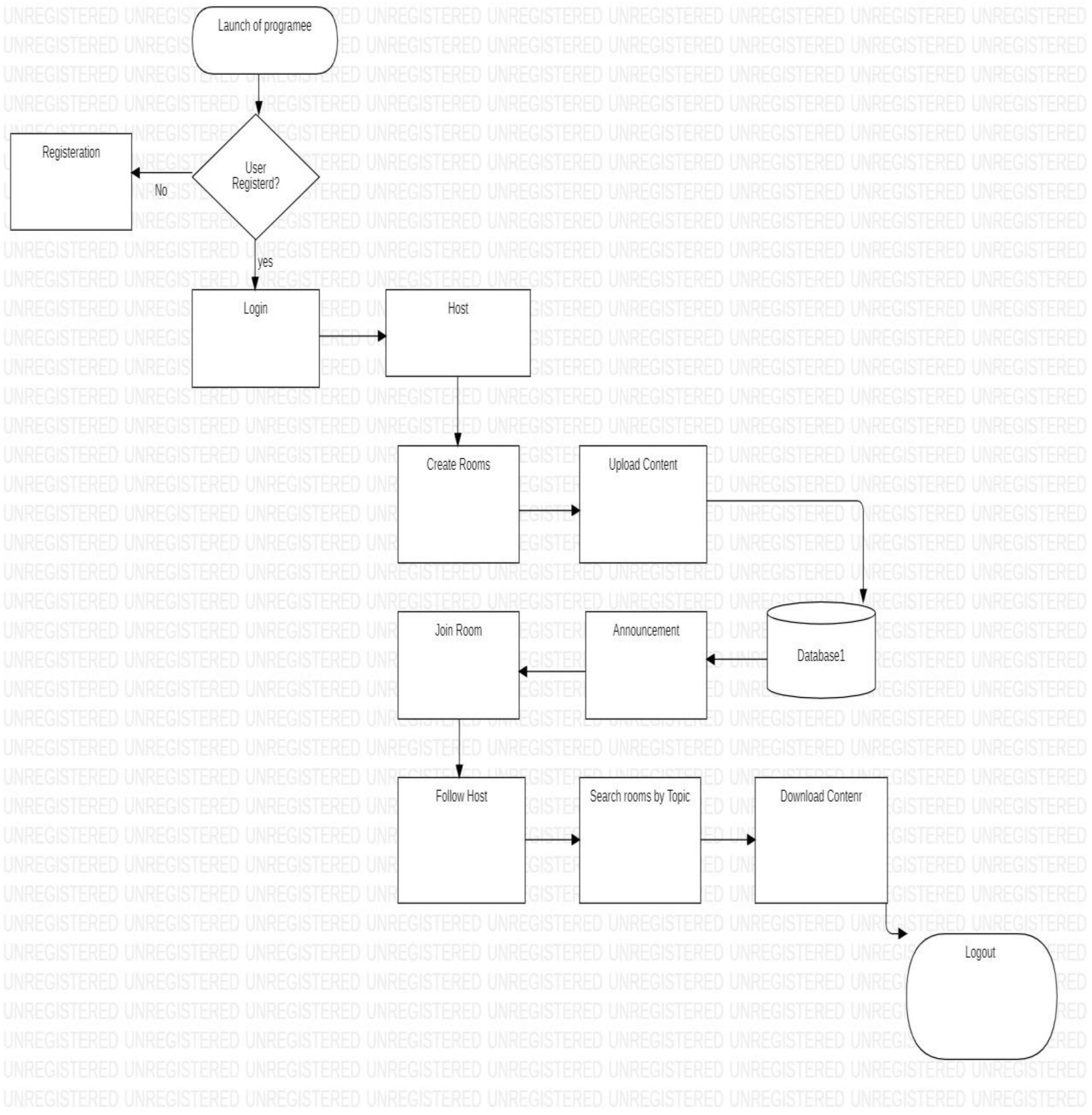


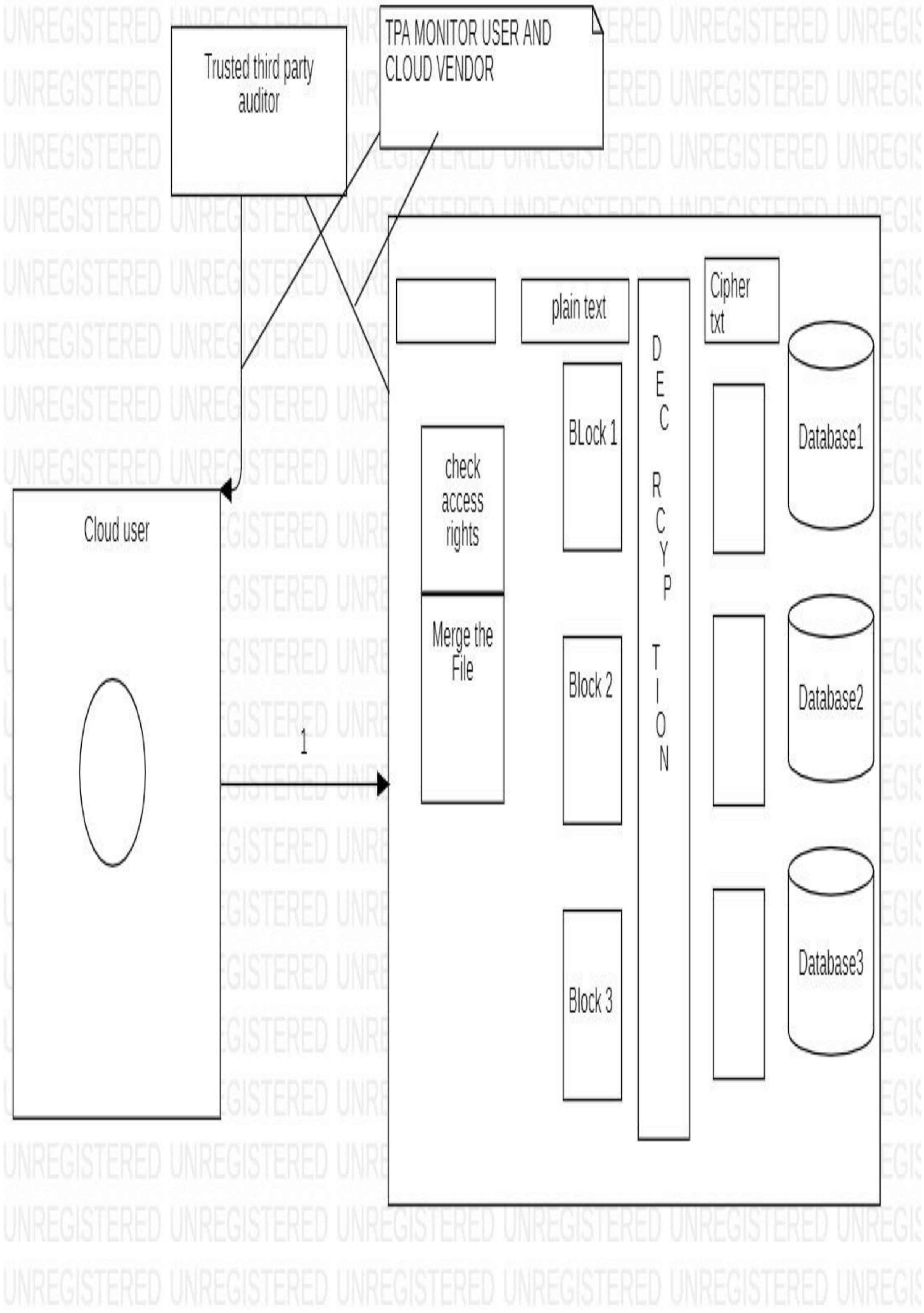

UseCaseSubject2
Register
Login
Upload File
download file
generate alert
view file list
Grant user
Monitore system
Assign reference key
insert
Delete
update
Mainrain database
generate bill
+include
«include»
«include»
«include»
Cloud User
Cloud Service Provider
Thir Party auditor
Database admin

4.3 Activity Diagram



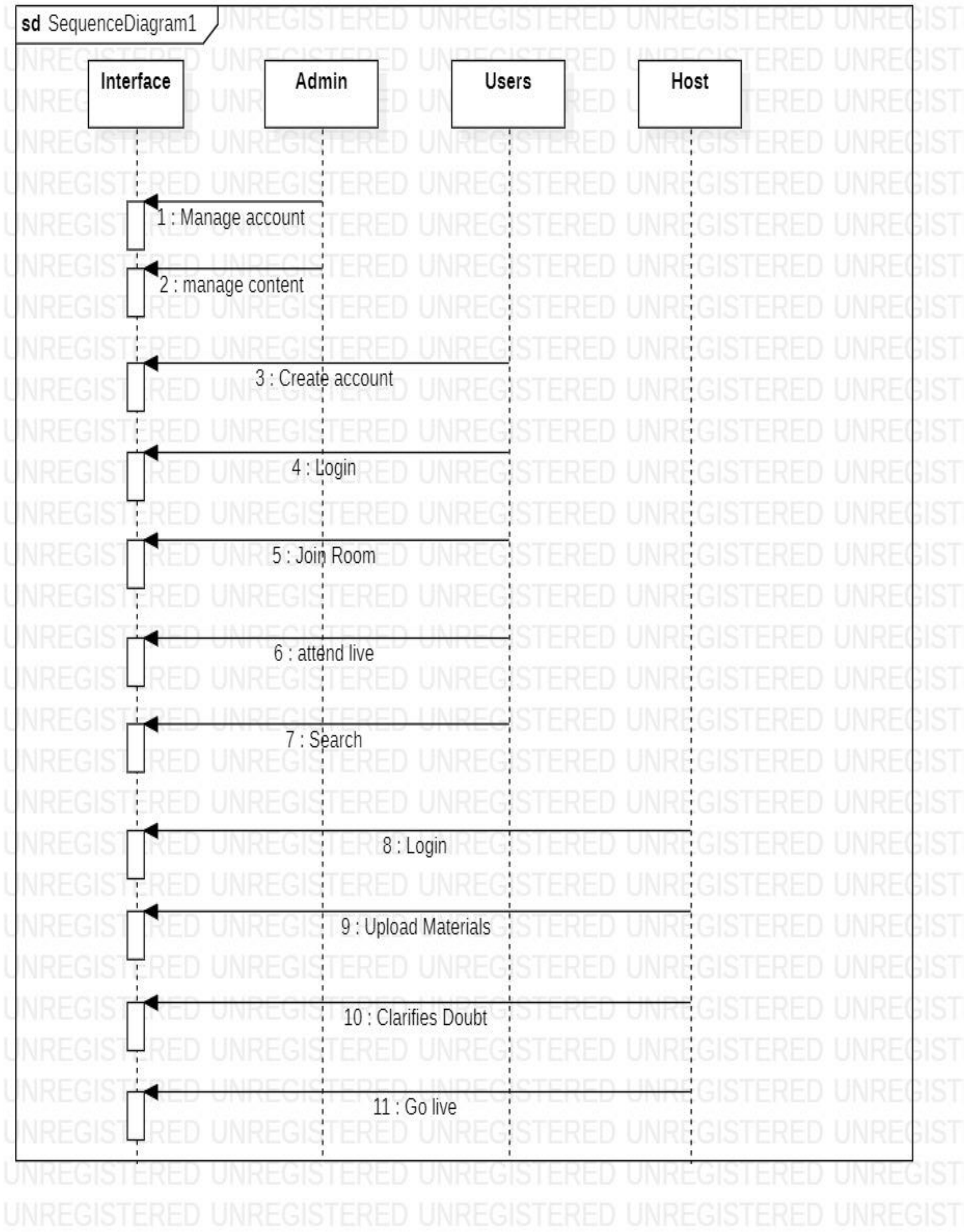
4.4 System Flowchart Diagram



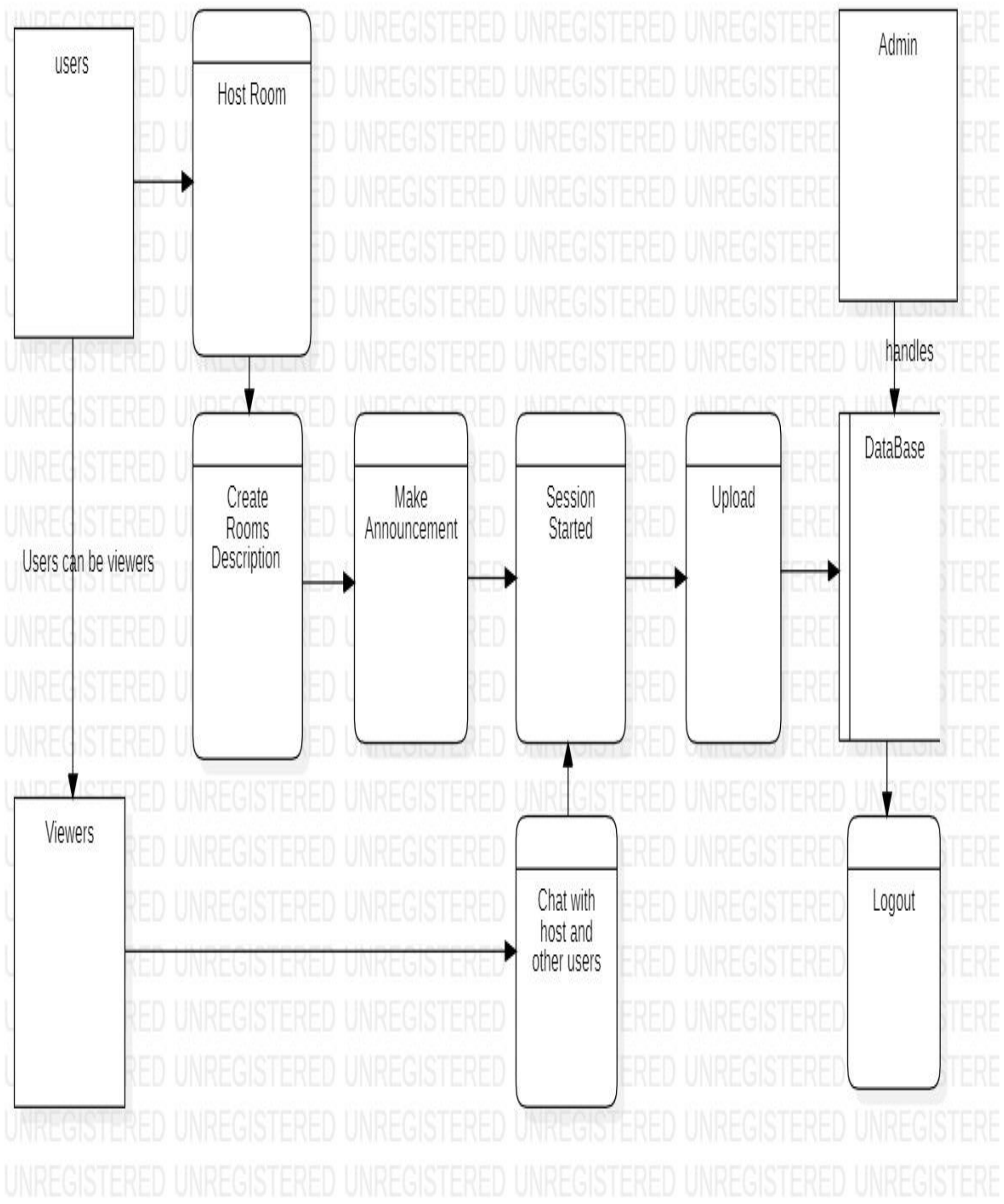


Trusted third party auditor
TPA MONITOR USER AND CLOUD VENDOR
Cloud user
1
check access rights
Merge the File
plain text
BLock 1
Block 2
Block 3
DEC RCYP TION
Cipher txt
Database1
Database2
Database3

4.5 Sequence diagram



4.6 Data Flow Diagram



Chapter 5. Cost and benefit analysis

5.1 Developing a cost benefit analysis

Since this is a demo project the upfront cost will be determined if the project varies. The initial cost of project is null and hence depends on whether or not the application has achieved significant amount of data and requires further establishment, however, the cost of the project depends on various factors. Those are included in this section:

The cost of setting up a Django-based web server on a cloud platform can vary widely depending on several factors. Here are some of the key factors that will influence the cost:

Cloud Provider: The cost of cloud hosting can vary significantly depending on the cloud provider you choose. Some popular cloud providers include Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), and others. Each provider has its own pricing structure.

Server Type: The type of server you choose will impact the cost. You can opt for virtual machines (VMs), containers, serverless functions, or other options, each with its own pricing.

Server Size: The amount of CPU, RAM, and storage you need will also affect the cost. Larger and more powerful servers typically cost more.

Storage: A storage for your database, static files and other assets. The cost of storage will depend on the amount of data needed to store.

Data Transfer: Cloud providers often charge for data transfer in and out of their networks. If your website has high traffic, this can be a significant cost.

Database: If you're using a database, the type of database and the amount of data you store will affect the cost.

Bandwidth: The amount of data transferred to and from the server will impact your bandwidth costs.

Geographical Location: The location of your server data center can also affect costs. Some regions may be more expensive than others.

Additional Services: Additional services like load balancing, content delivery networks (CDNs), or domain registration, which can add to the cost.

Traffic and Usage: Your website's traffic and usage patterns will directly affect the cost. More visitors and resource-intensive operations can increase costs.

Security and Compliance: If you need to implement additional security measures or comply with specific regulations, it might add to the cost.

Scaling: If you need to scale your web server to handle increased traffic, the cost will vary depending on the scalability options you choose.

To get an accurate estimate, the cost of this project should be verified at the end of the development.

Setting up a website with a low cost is possible by making strategic choices and utilizing affordable resources. Some key factors for lowering budget cost includes:

Select the Right Domain Name: Choose a simple and memorable domain name for your website. Many domain registrars offer domains at affordable prices. Look for promotional deals or discounts.

Choose a Cost-Effective Hosting Provider: There are hosting providers that offer budget-friendly hosting plans. Shared hosting is usually the most affordable option. Some providers even offer free hosting,

Optimize for SEO: Optimize your site for search engines to drive organic traffic. This will reduce the need for costly advertising.

Basic Security: Use free security plugins or features provided by your hosting provider to secure your site from common threats.

Responsive Design: Ensure your website is mobile-friendly, as it's essential for user experience and SEO. Most modern themes and website builders come with responsive designs.

Limit Plugins and Features: Avoid overloading your website with unnecessary plugins or features. This can reduce hosting costs and keep your site simple.

Content Delivery: Utilize Content Delivery Networks (CDNs) to reduce hosting costs and improve site speed. Some CDNs offer free plans.

Promote via Social Media: Instead of paid advertising, use social media and other free marketing channels to promote your website.

Monetization: consider monetizing it through affiliate marketing, ads, or selling digital products.

Regularly Review and Cut Costs: Periodically review the website's expenses and see if you can reduce costs further.

To opt for Free or Low-Cost Email Services: Instead of a custom email domain, you can use free or low-cost email services like Gmail for business communication.

5.2 Cost Evaluation

The initial cost evaluated is 0 Rs Upfront with AWS Educate.

Chapter 6 Conclusion and Future enhancement: