R.M.K GROUP OF ENGINEERING INSTITUTIONS

R.M.K
GROUP OF INSTITUTIONS

R.M.K
GROUP OF
INSTITUTIONS

R.M.K GROUP OF INSTITUTIONS

Please read this disclaimer before proceeding:

This document is confidential and intended solely for the educational purpose of RMK Group of Educational Institutions. If you have received this document through email in error, please notify the system manager. This document contains proprietary information and is intended only to the respective group / learning community as intended. If you are not the addressee you should not disseminate, distribute or copy through e-mail. Please notify the sender immediately by e-mail if you have received this document by mistake and delete this document from your system. If you are not the intended recipient you are notified that disclosing, copying, distributing or taking any action in reliance on the contents of this information is strictly prohibited.

R.M.K GROUP OF INSTITUTIONS

22AI001

AI in Blockchain

Batch/Year: 2022-2026/IV

Created by:

Mr. K. Rajesh Kumar Assist Prof/CSE (CS)
RMKCET

R.M.K GROUP OF INSTITUTIONS

Table of Contents

R.M.K
GROUP OF
INSTITUTIONS

Course Objectives

R.M.K
GROUP OF
INSTITUTIONS

Course Objectives

- To acquire knowledge in Blockchain Technologies.
- To Understand how block chain and AI can be used to innovate.
- To explain Cryptocurrencies and AI.
- To develop applications using blockchain.
- To understand the limitations and future scope of AI in Blockchain.

R.M.K GROUP OF INSTITUTIONS

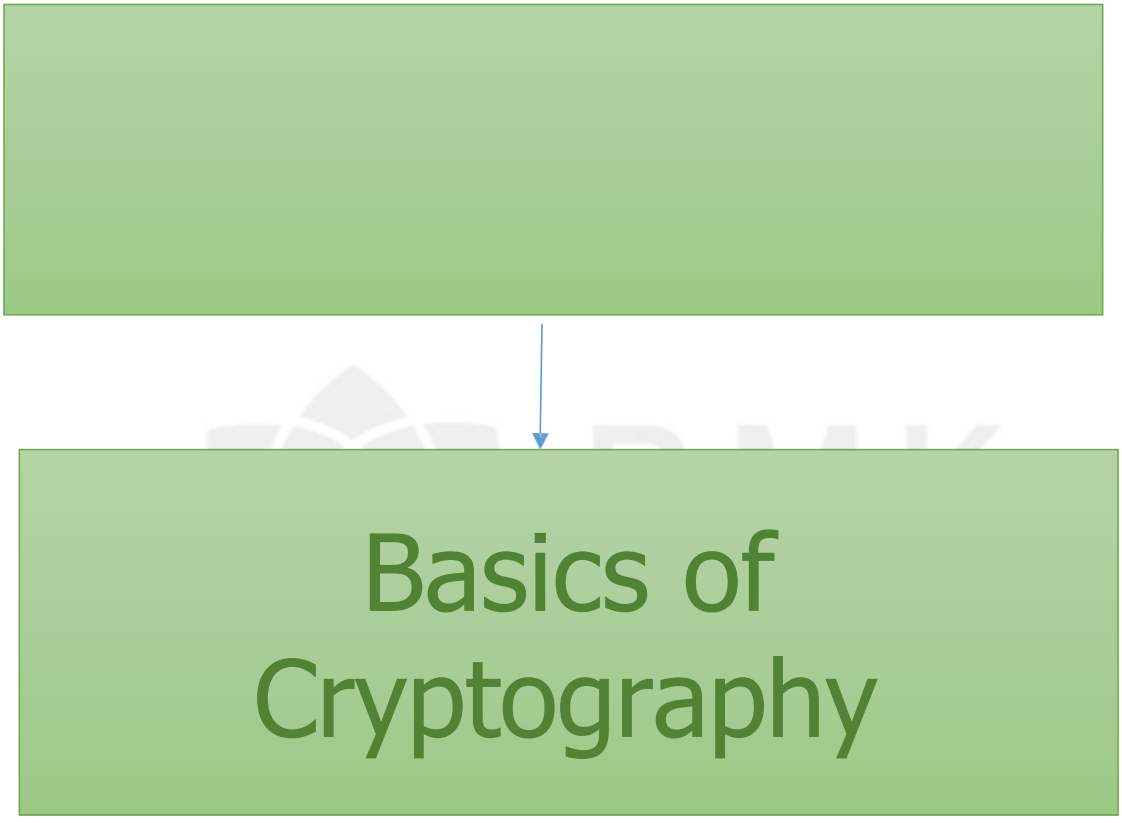
R.M.K GROUP OF INSTITUTIONS

R.M.K
GROUP OF
INSTITUTIONS

PRE REQUISITES

R.M.K
GROUP OF
INSTITUTIONS

Prerequisites



R.M.K
GROUP OF
INSTITUTIONS

R.M.K
GROUP OF
INSTITUTIONS

Syllabus

R.M.K
GROUP OF
INSTITUTIONS

OPEN ELECTIVE (Offered to Other Departments by ADS)

22AI001	AI in BLOCK CHAIN	L	T	P	C
		3	0	0	3

OBJECTIVES:

- To acquire knowledge in Blockchain Technologies.
- To understand how block chain and AI can be used to innovate.
- To elaborate Cryptocurrencies and AI.
- To develop applications using blockchain.
- To understand the limitations and future scope of AI in Blockchain.

UNIT I	INTRODUCTION TO BLOCKCHAIN	9

Overview – Blockchain vs Distributed Ledger Technology vs Distributed Databases – Public vs private vs permissioned blockchains – Privacy in blockchains – Blockchain platforms - Hyperledger – Hashgraph, Corda – IOTA - Consensus Algorithms – Building DApps with blockchain tools.

UNIT II	BLOCKCHAIN AND ARTIFICIAL INTELLIGENCE	9

Introduction to the AI landscape - AI and Blockchain driven Databases – Centralized vs Distributed data – Blockchain data – Big data for AI analysis – Global databases – Data Management in a DAO - Benefits of combining blockchain and AI – Aicumen Technologies -Combining blockchain and AI to humanize digital interactions.

UNIT III	CRYPTOCURRENCY AND AI	9

Bitcoins – Ethereum - Role of AI in cryptocurrency – cryptocurrency trading – Making price predictions with AI – Market making – future of cryptocurrencies.

UNIT IV	DEVELOPING BLOCKCHAIN PRODUCTS	9

Development Life Cycle of a DIApp – Designing a DIApp – Developing a DIApp – Testing – Deploying – Monitoring – Implementing DIApps.

UNIT V	LIMITATIONS AND FUTURE OF AI WITH BLOCKCHAIN	9

Technical Challenges – Business Model Challenges – Scandals and Public perception – Government Regulation – Privacy Challenges for Personal Records – Convergence of AI with Blockchain – Future – Enterprise.

TOTAL: 45 PERIODS

OUTCOMES:

At the end of this course, the students will be able to:

CO1: Acquire knowledge in Blockchain Technologies.
CO2: Understand how block chain and AI can be used to innovate.
CO3: Elaborate Cryptocurrencies and AI.
CO4: Develop applications using blockchain.
CO5: Understand the limitations and future scope of AI in Blockchain.
CO6: Elaborate the various applications of AI in Blockchain.

TEXT BOOKS:

1. Ganesh Prasad Kumble, Anantha Krishnan, "Practical Artificial Intelligence and Blockchain: A guide to converging blockchain and AI to build smart applications for new economies", Packt Publications, 2020.
2. Melanie Swan, "Block Chain: Blueprint for a New Economy", O'Reilly, 2015.

REFERENCES:

1. Daniel Drescher, "Block Chain Basics", Apress; 1st edition, 2017.

R.M.K
GROUP OF
INSTITUTIONS

Course Outcomes

R.M.K
GROUP OF
INSTITUTIONS

Course Outcomes

CO#	COs	K Level
CO1	Acquire knowledge in Blockchain Technologies	K1
CO2	Understand how block chain and AI can be used to innovate.	K2
CO3	Explain Cryptocurrencies and AI.	K2
CO4	Develop applications using blockchain.	K4
CO5	Understand the limitations and future scope of AI in Blockchain.	K4
CO6	Elaborate the various applications of AI in Blockchain.	K4

Knowledge Level	Description
K6	Evaluation
K5	Synthesis
K4	Analysis
K3	Application
K2	Comprehension
K1	Knowledge

R.M.K GROUP OF INSTITUTIONS

R.M.K
GROUP OF
INSTITUTIONS

CO – PO/PSO Mapping

R.M.K
GROUP OF
INSTITUTIONS

CO – PO /PSO Mapping Matrix

CO #	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PS0 3
CO1	3	3	2	1	1	-	-	-	2	-	-	2	3	2	-
CO2	3	2	2	2	2	-	-	-	2	-	-	2	3	2	-
CO3	3	3	2	2	2	-	-	-	2	-	-	2	3	2	-
CO4	3	2	2	2	2	-	-	-	2	-	-	2	3	3	-
CO5	3	2	2	2	2	-	-	-	2	-	-	2	3	2	-
CO6	2	2	1	1	1	-	-	-	2	-	-	2	3	2	-

R.M.K GROUP OF INSTITUTIONS

Lecture Plan

Unit IV

R.M.K
GROUP OF
INSTITUTIONS

Lecture Plan – Unit 4 –Developing Blockchain Products

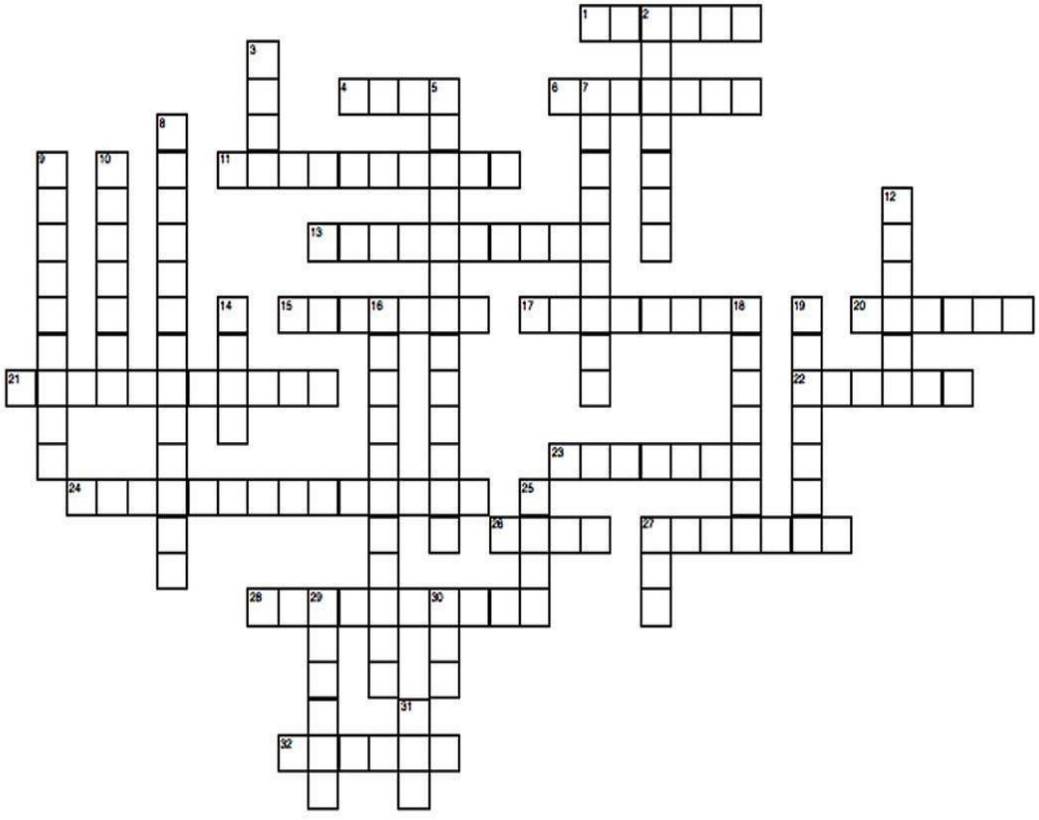
Sl. No.	Topic	Number of Periods	Proposed Date	Actual Lecture Date	CO	Taxonomy Level	Mode of Delivery
1	Development Life Cycle of a DIApp	1	08.09.2025	08.09.2025	CO4	K2	Chalk & Talk
2	Designing a DIApp	1	09.09.2025	09.09.2025	CO4	K2	Chalk & Talk
3	Designing a DIApp	1	09.09.2025	09.09.2025	CO4	K2	Chalk & Talk
4	Developing a DIApp	1	10.09.2025	10.09.2025	CO4	K2	Chalk & Talk
5	Developing a DIApp	1	12.09.2025	12.09.2025	CO4	K3	Chalk & Talk
6	Testing	1	16.09.2025	16.09.2025	CO4	K3	Chalk & Talk
7	Deploying	1	17.09.2025	17.09.2025	CO4	K2	Chalk & Talk
8	Monitoring	1	19.09.2025	19.09.2025	CO4	K3	Chalk & Talk
9	Implementing DIApps	1	20.09.2025	20.09.2025	CO4	K3	Chalk & Talk

R.M.K
GROUP OF
INSTITUTIONS

Activity Based Learning

R.M.K
GROUP OF
INSTITUTIONS

Activity Based Learning



R.M.K GROUP OF INSTITUTIONS

Across

1-. disguised writing (added hint - was used by Julius Caesar)
4-. distributed computer to computer file sharing protocol created by Juan Benet
6-. Dan ____? Blockchain developer of EOS, Bitshares & Steemit
11-. digital ledger aka
13-. another name for user & computers that share resources (added hint: napster was one)
15-. free falling out of the sky?
17-. Bitcoin followers?
20-. act of hashing?
21-. shorthand writing
22-. Nazi encryption machine
23-. annoying online anti-bot filter (added hint - coined by Luis Von Ahn)
24-. encrypted digital money
26-. Bitcoin enthusiasts worst nightmare? (added hint - its an acronym like 3Down)
27-. Leon Battista _____
28-. grandfathers of crypto? (added hint: Hal Finney was one)
32-. where you keep your bitcoin

Down

2-. public key and _____ key?
3-. bitcoiners don't sell, they _____?
5-. cryptocontract, another word for
7-. mathematical computer problem solving process
8-. hiding a message within a message
9-. converted data into ciphered computer code
10-. satoshi's creation
12-. Alan _____? "father of modern day computing"
14-. internet junk mail
16-. ability to use the same digital token more than once (added hint - think twice)
18-. spartan encryption tool
19-. _____.com "a blockchain based social media network"
25-. in the crypto-world it's not a spoon, it's a _____?
27-. 128 bit encryption
29-. original hackers (added hint - phone _____?)
30-. 1st freeware encryption program for the public? (added hint- think Hal again)
31-. 56 bit encryption

R.M.K GROUP OF INSTITUTIONS

R.M.K
GROUP OF
INSTITUTIONS

Lecture Notes – Unit 4

R.M.K
GROUP OF
INSTITUTIONS

UNIT IV

4.1 Applying SDLC practices in blockchains

With more than 10 years since the advent of blockchain technology, there is a need for newer emerging patterns that can apply blockchain technology and AI techniques to address the growing demands from the ever-expanding internet, and effectively manage the software development practices across various industry verticals.

4.2 Major aspects of SDLC

Ideation to productization

Many ideas for building new applications on top of blockchain exist. However, fewer ideas are converted into designs. Although efforts are made to design decentralized applications, only a few designs are practical enough for the real world. This is due to the mismatch between product expectations and the readiness of technology at the developer's disposal. In some cases, the technology and features may become available, but they may not be supported well enough due to the recent and unstable growth of these blockchains. Hence, it is a commonly perceived problem that not all ideas in blockchain can be converted into a product.

Apart from the limitations of the platforms themselves, there is also a tendency to blockchainify every existing solution, due to the **Fear of Missing Out (FOMO)** on opportunities. Some companies want to try it even when it may be unsuitable for their purpose, and this can lead to issues as well

Steps to resolve issues:

1. **Understand the business process:**

R.M.K GROUP OF INSTITUTIONS

In most of the blockchain-based use cases I have come across, a clear understanding of the business process is crucial since a majority of blockchain solutions will be affecting the operational as well as financial aspects of a business. Therefore, it is important to educate all stakeholders about the business process in greater detail before dangerous assumptions can be made on the intricate steps involved or the method required.

For example, if a new solution is being developed to digitally transform a dairy firm using blockchain and AI, everyone on the team must have a comfortable understanding of the end-to-end process of bringing milk from the cattle farm to the table.

2. Establish clear requirements:

It is a common observation in any emerging technology that most of the efforts are often hyped, thereby lacking the objective clarity needed to build apps. Before we can identify potential points of integration, we must have clarity on what pain points are being addressed by the application of blockchain or AI in the solution. All stakeholders must do their best here to clearly communicate the functional and non-functional requirements. This can be helpful in managing expectations in the future.

For example, the decision makers or owners of the organization may establish the need to achieve better transparency and accountability from the local cattle centers that produce raw unprocessed milk. The owners may also identify the need to analyze the current sales trends for all their dairy products in order to ensure that milk is made available for further processing to manufacture the necessary dairy products according to demand.

3. **Identify critical checkpoints in the business processes:**

Once the requirements have been identified and clarified, it is crucial to not jump straight into design in the case of blockchain. We must

R.M.K GROUP OF INSTITUTIONS

identify the current implementations and understand the critical business components and the checkpoints where technology must be applied and integrated.

For example, identifying each cow in the cattle herd could be one of the most basic and foundational checkpoints for the team. Also, it is crucial to understand whether the volumes of raw milk collected at the local producers are manually inputted into the system or automated through a digital weight scale. In the case of maintaining a certain amount of milk as a reserve for other dairy products, it is important to understand who makes the decision or approves the manufacture of dairy goods out of the reserve milk.

4. Check whether technology integration is feasible:

Once the checkpoints are identified, we can now identify various technical approaches to solve the problem and check whether the approach fits in the larger process. A lot of the time, unlike traditional solutions, blockchain developers are limited, either by the infrastructure and platforms, or by the lack of technical stability of the features supported by a given blockchain platform.

For example, let's assume that we want to identify each cow in the cattle herd by a Radio Frequency Identification (RFID) tag. Now, each cow tagged with an RFID tag must be virtually represented in the blockchain through a state variable. Here, it is imperative that the developers do not assume that the value of the RFID tag will be persistable on a blockchain. Most blockchain platforms have serious constraints on the type of data that can be stored. They also impose serious restrictions on the length and range of the data types that could be persisted in a blockchain through a smart contract, due to the blockchain's decentralized nature. In this case, we may want to identify the structure of an RFID tag and try to store it in a secondary storage network such as MóiBit to identify all cows in the herd.

R.M.K GROUP OF INSTITUTIONS

Similarly, in the case of measuring the volume of milk produced, it is imperative to identify whether a digital weight scale is capable of making a smart contract call in the blockchain. If the milk is collected from very remote areas, it is important to identify such operational impediments also.

Finally, it is also very crucial to identify how the application of AI models on the information gathered on the blockchain can be harnessed to address user requirements. That is, designers, architects, and developers must be aware of the blockchain's transactional information and its data structure. Initial efforts must be made to understand that the transactional information can be processed in such a way as to ensure that an AI model can be trained sufficiently to predict the sales trends and set aside necessary milk reserves for milk-based dairy products

5. Establish technical dependency across affected components:

Once we identify potential integration points, we should carefully establish the technical dependencies. These dependencies can be either internal dependencies or external dependencies. Technical dependencies are internal if the development of a solution depends on its design, architecture, user stories, or acceptance criteria. The poor design of a given solution or information flow can cause many issues. Incomplete architectural decisions can also lead to issues in implementation.

The following diagram summarizes the need for better clarity in all aspects to build the ideal solution using blockchain and AI:

R.M.K GROUP OF INSTITUTIONS

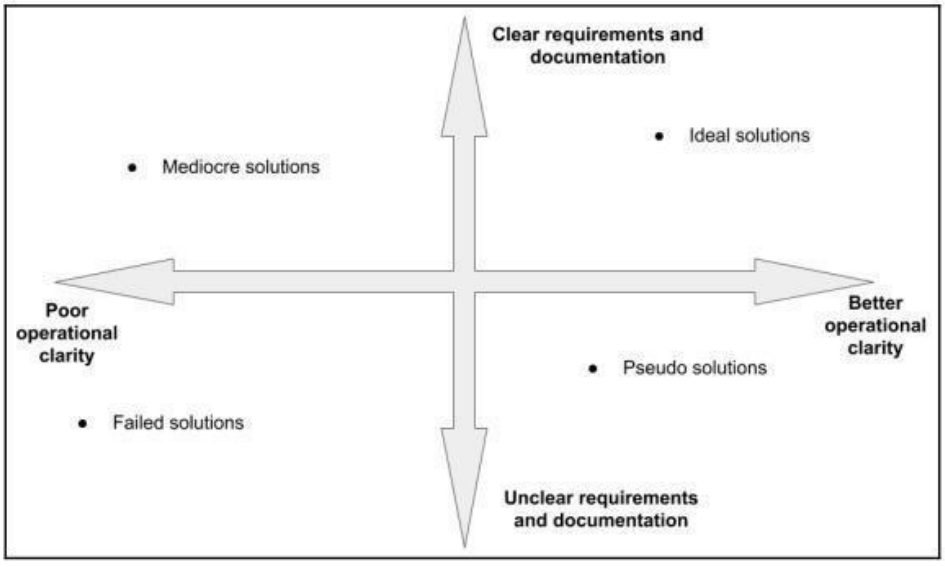


Fig 7.1: A diagram representing various outcomes and the quality of solutions

To summarize, as mentioned in the preceding diagram, ideal solutions can be achieved only when the requirements are very clear and specific, along with having the upper hand in technical capabilities. However, if the requirements are not fully documented, understood, nor communicated, the outcome is a partial solution that does not propose addressing the complete set of problems. The two other types of solutions in the left-side quadrants are poor and ineffective due to the lack of technical capabilities in establishing the dependencies with some room for ineffective documentation of requirements

4.3 Introduction to DIApps

Decentralized applications or dApps (also written as DApp or Dapp) are user applications that run on a platform hosted by a number of nodes in a distributed manner. DApps emerged as a solution architecture on top of blockchains wherein custom business logic could be programmed in a particular language of support. Once the logic is interpreted into code, we deploy this code on the respective blockchain

R.M.K GROUP OF INSTITUTIONS

platforms. Once the logic is deployed on the platform, we further integrate the logical program with frontend applications for user interactions.

In contrast to DApps, a Decentralized Intelligent Application (DIApp) is an enhanced pattern of a DApp that facilitates the application of AI wherever applicable, on top of a blockchain platform, in a much more robust manner that provides value to all stakeholders. Although the concept of slapping AI on top of solutions is not an unusual thing, DIApps are an understandable approach and a novel pattern that makes more sense for future generations of solutions built using both blockchain and AI.

A DIApp is an application offering both decentralized and intelligent capabilities. Since it is a pattern inherited from DApps, it is decentralized by default, thanks to the implementation currently followed by all the blockchain platforms. However, DIApps have the exclusive capability of being more intelligent. This is made possible by an intermediary off-chain database that gives DIApps the ability to store big data from an application or the users through the application running on the blockchain

The following diagram providers a general schematic of a DIApp as explained in this section:

R.M.K GROUP OF INSTITUTIONS

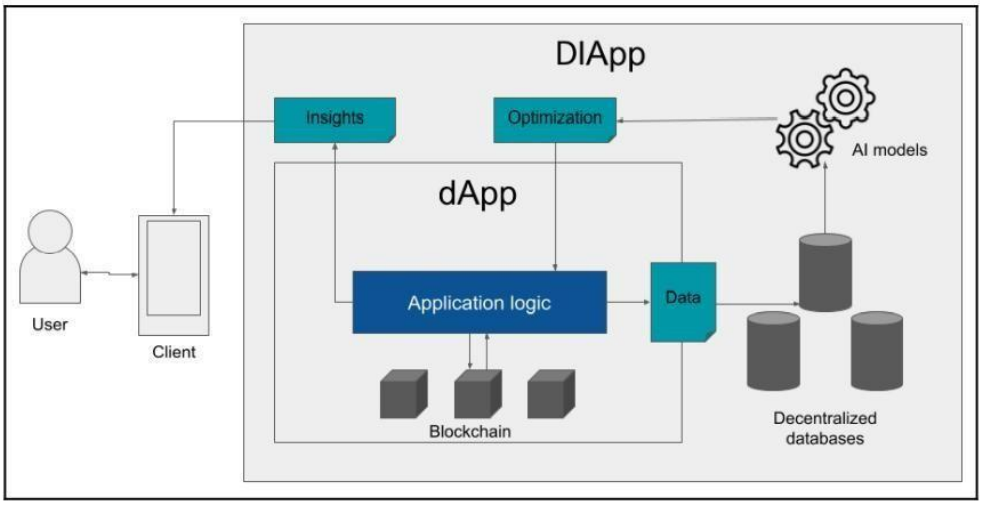


Fig 7.2: Composite view of the DIApp pattern

The preceding diagram depicts the composite view of a DApp within a DIApp pattern. The pattern is depicted in this fashion to help you understand the enhancements that are made to the existing pattern such that the application of AI can be facilitated.

Challenges in the enterprise

Although several blockchain platforms represented new waves of experimental approaches, most of these approaches radically focused on the full disintermediation of all the entities involved in a business process, thereby making the solutions purely peer-to-peer. The reason for the full disintermediation in most of the emerging patterns is due to the maximalist nature of blockchain to decentralize the world. In contrast to this ideology, enterprise solutions fundamentally rely on maintaining accountability among key stakeholders without full disintermediation. Hence, there is a need for an enhanced pattern that is fit for adoption by enterprises.

Solution architecture of a DApp

The solution architecture of a DApp primarily comprises the following key points:

The business logic is written in a Domain-Specific Language (DSL) in a smart contract.

The smart contract is deployed on a blockchain network, identified by an address.

The smart contract will acquire a new address if the business logic is to be updated in a majority of the blockchain platforms.

A web or mobile application is developed as a client to access and carry out operations on the business logic.

Almost all the critical data is stored on top of the blockchain, with little or no scope for analytical capabilities.

Solution architecture of a DIApp

The solution architecture of a DIApp primarily comprises adding technical capabilities to make the pattern more approachable by enterprises. Here are the key highlights:

Business logic is usually written in any high-level language such as C, Python, Java, or Golang, among others supported by the Low-Level Virtual Machine (LLVM) compiler. Also, normal smart contract languages can be used.

The majority of smart contracts are upgradeable, meaning that the code can be updated without having to change its address

Traditional client applications can also integrate with these types of smart contracts as there is less need for integrating it with newer middleware.

Critical business data is not only stored in blockchain networks but also in decentralized data storage systems to ensure better integrity of the data.

Key differences

R.M.K GROUP OF INSTITUTIONS

DApp	DIApp
An application deployed on a blockchain platform with all the core logical elements.	Core logical elements are deployed on a blockchain platform, but are also powered by AI-driven insights in parallel.
Data is mostly sitting on the blockchain platform, making it costly in terms of expense and time for retrieval.	Critical provenance data resides on the blockchain, but the remaining data is persisted on a cheaper off-chain storage system.
Not all DApps are upgradeable, hence a new address is required for every update to the logic. This could break systems.	Most applications can easily be upgraded without making changes to the address, hence nothing breaks in the system.
Data is rarely analyzed due to the cost of read-writes in a blockchain platform, and also due to the lack of structured data required to train AI models.	Data can be easily structured in an off-chain storage system and tightly coupled back to the blockchain platform as well as the AI models.

4.4 Designing a DIApp

The design aspects of DIApps have been often considered to be somewhat challenging and dictated by the technical complications introduced in the constant waves of change seen in almost all blockchain platforms. It is also a common perception that the solution space lacks a common structure to define key components, resulting in an inconsistent design strategy for applications

For example, the tools required by the user of an Ethereum-based DIApp are very distinct from that of Hyperledger Fabric, and subsequently, that of EOS. This is due to the distinct design of the respective blockchain's UI/UX framework, which is deeply dependent on its own design paradigms. Hence, it is important to identify the design constraints of the application before commencing the future steps.

Research

Before choosing which blockchain technology or platform you will use, it is critical to understand the user requirements, analyze them, and perform better research at an early stage of design. Instead of asking

R.M.K GROUP OF INSTITUTIONS

which blockchain technology can help address the requirements, consider these scenarios:

1. Does the solution demand full decentralization over a public network for peer-to-peer interactions?

2. Does the solution need to be implemented in a private network due to the extreme sensitivity of the data and minimal exposure of the business logic?

3. Can the solution be implemented on a public network with all the sensitive data encrypted on a public decentralized storage system?

4. Can the same solution be implemented on a public network with all the sensitive data stored in a private virtual decentralized storage service?

5. Does the solution require a permissioned network of authorized nodes across multiple stakeholders of a consortium?

Based on the preceding pointers, we can undertake the following analysis. For scenario 5 in the preceding list, you are better off choosing a blockchain such as Hyperledger Fabric. However, in the cases of scenarios 3 and 4, you will need the InterPlanetary File System (IPFS) or an IPFS-based service provider who will encrypt the data and store it in a safe medium, offering security and redundancy with a secondary network. In scenario 2, you may be better off again with Hyperledger Fabric. Finally, in scenario 1, which might be the majority of cases, Ethereum might best fit your solution requirements

Conceptualization

Formulating the Proof of Concept is an essential step in the development of solutions using emerging technologies. The development of proof of concepts for decentralized applications is also important, as they ensure alignment between the requirements and

R.M.K GROUP OF INSTITUTIONS

delivery of the solution. This helps maintain functional and design conformity right from the early days of development.

Also, it is notable that the proof of concept specifications may change due to the waves of design and architectural change in the blockchain landscape. Hence, proof of concepts should focus more on the functional viability of compatible patterns available to the developers.

Product-market fit

Developing the proof of concepts and prototyping the critical aspects of an application form a recursive practice until we are able to see a product-market fit. The definition and constraints of the product-market fit for DIApps cannot be general since each DIApp may target a specific problem in a specific domain. However, a few common attributes could help assert the product-market fit of a DIApp. They are as follows:

Does the DIApp solve a unique problem faced by various stakeholders in the industry?

Does the DIApp disintermediate a current entity that is inefficient in the current process, or bring more order to the process?

Can the benefits of DIApps be achieved only by utilizing the combination of blockchain and AI?

Does the DIApp facilitate users to derive insights by using the DIApp with the help of AI models?

By answering the preceding questions, we may be able to identify whether the DIApp has achieved a suitable product-market fit. It is also important for you to identify other key indicators apart from the preceding general attributes to identify the product-market fit for your DIApp.

R.M.K GROUP OF INSTITUTIONS

4.5 Developing a DIApp

The development of a DIApp can be tricky. Since AI and blockchain are yet to see the limelight in development, some practices in the industry are not yet visible to all. As blockchain and AI are on the bleeding edge of innovation, it is an open truth that many organizations are still in the process of building combined expertise in the respective technologies. Having said that, it is also essential to set up a team of members with complementary skill sets in smart contract development, web or mobile application development, and finally AI or data science modeling. The following section provides an overview of an ideal DIApp team.

Team formation

It is assumed that the team members are enabled with regular technical and solution expertise:

Two smart contract developers with good hands-on knowledge of the Solidity, Rust, and Golang programming languages are essential. Knowledge of Haskell is also preferred, since a few blockchains offer smart contracts based on functional programming languages.

One smart contract developer may be able to focus on feature development and another smart contract developer could work on bug fixes, internal auditing, code quality reviews, and so on.

One full-stack web developer with basic knowledge of blockchain to develop the frontend web application. If the target audience is on mobile, you might choose a mobile application developer accordingly.

One SMACK-stack developer may be required to set up the analytics platform needed for building AI capabilities in the solution.

One machine learning or deep learning engineer may be required to build the required models. If an application requires deep learning or a neural network, you should make your choice accordingly.

Finally, you may need one DevOps engineer to orchestrate the infrastructure and deploy all the necessary components across vendors

R.M.K GROUP OF INSTITUTIONS

or cloud platforms through a well-defined Continuous Integration/Continuous Delivery (CI/CD) pipeline.

An all-star team of six members with complementary skill sets can help develop the proof of concept and establish the needed technical features. The two smart contract developers can be replaced by one senior smart contract developer with end-to-end experience.

Agile development

Once the team is set up and the requirements are neatly documented, establish a clear release plan for the proof of concept and the subsequent builds as much as possible. As mentioned earlier in the Designing a DIApp section, the technical aspects of blockchain technologies change very frequently. In order to match the pace of change, it is important to manage client expectations. Otherwise, it could lead to another level of complication on top of technical issues.

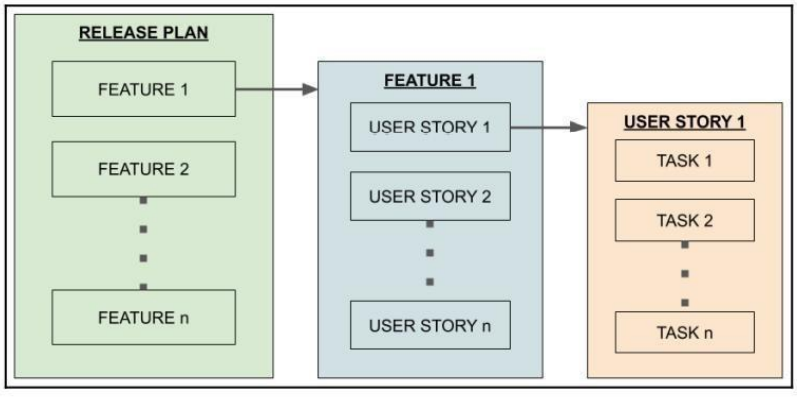


Fig 7.3: Schematic of breaking down the release plan into workable tasks for the team

As shown in the preceding diagram, the requirements need to be documented very specifically and broken down into items across three levels. Managers and product owners both work on features in a release plan, whereas the product owner alone specifies the requirements for

each aspect of the proof of concept or any release in the form of a user story. Once the user story is documented with an expected output, the team can collectively break it down into many work items or tasks. You can also break down the task into multiple sub-tasks for a large project beyond the proof of concept release for a better and more granular documentation of efforts. The tasks can be assigned to individuals and can be either blockchain-related or AI-related.

If the requirements are documented clearly, you can organize sprint planning. As per agile development, you can organize a sprint for at least 2 to 3 weeks. A proof of concept release can be planned across one or many sprints. At the end of each sprint, the team must organize a retrospective meeting to review the progress and reflect on the current practices.

Compared to retrospectives of traditional apps, you could also identify the risks or issues created by the components. Since the ecosystem is still under development, the observed deficiencies can be converted into issues. The issues can be segregated and filed at the repositories on the respective dependencies. Most of the dependencies reside on GitHub. So, the developers must have sufficient awareness of the platform to navigate it, collaborate with others, and resolve the issues.

The process repeats across many sprints until all the features, requirements, and bug fixes have been addressed in alignment with the release plan.

4.6 Testing a DIApp

Since these applications will most likely manage high-value assets and sensitive information, testing DIApps is a crucial step in the process before and after the deployment of all the modules in the network. Ensuring the correctness and lasting service of the application is critical for businesses and hence, it is a very important aspect of the

R.M.K GROUP OF INSTITUTIONS

development life cycle to bring back rigorous testing practices into the process.

Authoring the test cases

Before the software can be deployed, we can try to ensure the correctness of the software and also confirm the fitness of the runtime environment by running a few crucial tests. Usually, these tests are implemented within the source code, in the form of unit test files containing dummy input values being passed to the functions, later checked through assertion to ensure the correctness of the software in the relatively new environment. Each test case represents a scenario for the logic to perform and provide a predetermined output scenario. Test cases are executed across two levels of testing: unit testing and integration testing.

Unit testing

Each component in the DIApp pattern delivers an exclusive value to the solution. Hence, it is important to ensure the correctness of each component in the solution. This can be made possible by performing unit testing. Under unit testing, we can test the core functionalities of each module before integrating it with the other modules. Unit testing helps in formally verifying the correctness of these modules by comparing the real output to the desired output. This process repeats for each change made to the module, thereby preserving the correctness of the module each time a change has been made.

Once the modules are unit tested and integrated, we can perform integration testing, as explained in the following section.

Integration testing

As explained in the previous sections, a DIApp is a hybrid composition of many technologies. Hence, it is important to test the behavior of each heterogeneous component in order to ensure that they

R.M.K GROUP OF INSTITUTIONS

provide accurate results. This is achieved by performing integration testing. Once the components are integrated, we can run a few test cases that can formally verify the correctness of a few critical components that rely upon one or more heterogeneous components to provide output

Testing AI models

Although traditional software can be tested by unit tests and integrations tests, we need different measures and methods to test AI systems. Testing AI models and capabilities can be split into two phases. One phase of the testing is before the model reaches production. The other phase of the testing is applied post-production. Before bringing the AI model to production, testing can be performed by verifying the correctness and completeness of the training data. Similarly, once the AI models are deployed, we can test them frequently for accuracy and availability. Turing tests can also be performed to understand whether the AI model has sufficiently been able to replace a human response to the task.

4.7 Deploying a DIApp

Unlike other emerging technologies, blockchain platforms demand a relatively larger stretch of time to set up the network and make the whole ecosystem functional. Accordingly, several blockchain platforms have understood the need for DevOps as an integral part of developing those platforms. Hence, it is also important to note that DevOps knowledge is also essential in using these platforms for developing applications and deploying solutions.

Scaling the application for production

Deployment of the DIApp is the final crucial step in the life cycle. Apart from correctness, it is imperative that the application is designed, architected, and developed in a manner supportive for the deployment of the application in a scalable manner. The scale is not only measured

R.M.K GROUP OF INSTITUTIONS

in terms of the number of users, but also depends on the cost, form factors, and other economic attributes that may directly affect the operations.

Several tools such as Docker, Kubernetes, Ansible, Terraform, and Mesos are available for deploying DIApps and their dependencies.

Docker

Docker is a platform for packaging, deploying, and running applications. Docker applications run in containers that can be used on any system: a developer's laptop, systems on premises, or in the cloud.

Container

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.

A Docker container

A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings. Container images become containers at runtime and in the case of Docker containers – images become containers when they run on Docker Engine.

It is available for both Linux and Windows-based applications, containerized software will always run the same, regardless of the infrastructure.

Containers isolate software from its environment and ensure that it works uniformly despite differences for instance between development and staging.

Virtual Machine

A virtual machine is a software that allows us to install and use other operating systems (Windows, Linux, and Debian) simultaneously on our machine. The operating system in which virtual machine runs are called virtualized operating systems. These virtualized operating systems can run programs and preforms tasks that we perform in a real operating system.

Containers	Virtual Machine
Integration in a container is faster and cheap.	Integration in virtual is slow and costly.
No wastage of memory.	Wastage of memory.
It uses the same kernel, but different distribution.	It uses multiple independent operating systems

Docker is designed to benefit both the Developer and System Administrator. There are the following reasons to use Docker -

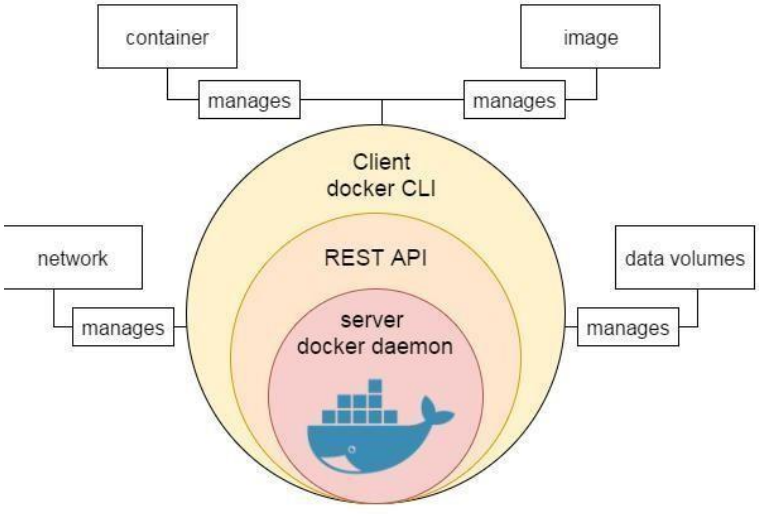
- Docker allows us to easily install and run software without worrying about setup or dependencies.
- Developers use Docker to eliminate machine problems, i.e. "**but code is worked on my laptop**." when working on code together with co-workers.
- Operators use Docker to run and manage apps in isolated containers for better compute density.

- Enterprises use Docker to securely built agile software delivery pipelines to ship new application features faster and more securely.
- Since docker is not only used for the deployment, but it is also a great platform for development, that's why we can efficiently increase our customer's satisfaction.

DockerEnigne

It is a client server application that contains the following major components.

- A server which is a type of long-running program called a daemon process.
- The REST API is used to specify interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface client.



1. Docker Client

Docker client uses **commands** and **REST APIs** to communicate with the Docker Daemon (Server). When a client runs any docker command

on the docker client terminal, the client terminal sends these docker commands to the Docker daemon. Docker daemon receives these commands from the docker client in the form of command and REST API's request.

Docker Client uses Command Line Interface (CLI) to run the following commands -

docker build

docker pull

docker run

2. Docker Host

Docker Host is used to provide an environment to execute and run applications. It contains the docker daemon, images, containers, networks, and storage.

3. Docker Registry

Docker Registry manages and stores the Docker images.

There are two types of registries in the Docker -

Pubic Registry - Public Registry is also called as **Docker hub**.

Private Registry - It is used to share images within the enterprise.

DockerOjebcts

There are the following Docker Objects -

Docker Images

Docker images are the **read-only binary templates** used to create Docker Containers. It uses a private container registry to share container images within the enterprise and also uses public container registry to share container images within the whole world. Metadata is also used by docket images to describe the container's abilities.

Docker Containers

Containers are the structural units of Docker, which is used to hold the entire package that is needed to run the application. The advantage of containers is that it requires very less resources.

Docker Networking

Using Docker Networking, an isolated package can be communicated. Docker contains the following network drivers -

- **Bridge -** Bridge is a default network driver for the container. It is used when multiple docker communicates with the same docker host.
- **Host -** It is used when we don't need for network isolation between the container and the host.
- **None -** It disables all the networking.
- **Overlay -** Overlay offers Swarm services to communicate with each other. It enables containers to run on the different docker host.
- **Macvlan -** Macvlan is used when we want to assign MAC addresses to the containers.

Docker Storage

Docker Storage is used to store data on the container. Docker offers the following options for the Storage -

- **Data Volume -** Data Volume provides the ability to create persistence storage. It also allows us to name volumes, list volumes, and containers associates with the volumes.
- **Directory Mounts -** It is one of the best options for docker storage. It mounts a host's directory into a container.
- **Storage Plugins -** It provides an ability to connect to external storage platforms.

4.8 Monitoring a DIApp

Explorers

Most of the decentralized solutions are deployed on a public blockchain network or digital ledger. In most of the public networks, blockchain explorers are available to look up information concerning a transaction or block. However, if the DIApp solution is implemented in a private or permissioned environment, these public blockchain explorers may not be able to provide information on transactions belonging to a private network or private ledgers. Hence, we must be able to deploy existing blockchain explorers and plug them into the endpoints of a private service. This is the only way to facilitate users' monitoring of their transactions in a private environment. Several open source implementations of blockchain explorers are available, which could be downloaded and connected to private services.

R.M.K GROUP OF INSTITUTIONS

Some examples of public blockchain explorers include Etherscan, EthStats, and BlockScout. BlockScout is an open source explorer that can be used to create a specific explorer for private Ethereum network.

What is Blockchain Explorer?

An online tool, Blockchain Explorer is designed and devised to view all the information associated with blocks, addresses, and past and current transactions on a Blockchain. These online tools generally provide useful information such as network hash rate and transaction growth indexes.

Practical applications of blockchain explorer

1) Explore any wallet address's transaction history: In order to audit any wallet address and increase blockchain transparency, you can use a blockchain explorer.

2) Look into changing and receiving addresses: In contrast to the transaction receiving address, you can also view the change address, which is an output that gives crypto to the spender in order to prevent an excessive amount of the input value from going to transaction fees. This enhances the transactions' transparency as well.

3) Study the most significant transaction of the day: This is available by various explorers.

4) Analyze Mempool status: This enables us to analyze the specifics of unconfirmed blockchain transactions.

5) Investigate double-spend occurrences: Some explorers are in favor of finding out how many double-spend transactions are occurring in a blockchain.

6) Examine orphaned and stale blocks: These are blocks that, although having been mined, are not connected to the longest blockchain and whose parent blockchain is unidentified. Blocks that have known parents but aren't yet joined the longest known chain are stale

blocks. Thanks to some explorers, we can see how many of these blocks were actually realized in a blockchain.

7) Discover the mining pool or individual who discovered or mined a specific block: In any specified blockchain, mining pools (groups that pool their computing power to mine cryptocurrency) compete to discover blocks, and explorers let us discover who successfully discovered a provided block defined by its height.

8) Investigate genesis blocks: You can discover who mined the very first block on a specific chain as well as additional information about that block.

9) Provides users with access to data such as hash rate, transaction fees, and blockchain difficulty.

Components of Explorer

1. **Price**

The price section in the explorer portal represents an aggregated USD price feed throughout different markets. In majority of cases, the price factor is directly related to the feed's provider without any indications of spot prices.

2. **Transactions**

Transactions in the blockchain explorer portal represent the different unique transactions carried out in the course of the last 24 hours. A transaction should be in a validated block for confirmation.

3. **Estimated Hash Rate**

The blockchain explorers could also provide a visual representation of the predictions for computing resources implemented by miners for blockchain. The estimated hash rate is generally perceived as the proxy for security in a Proof of Work (PoW) blockchain.

4. **Mempool Size**

The mempool size can help in tracking the aggregate size of transactions, generally in bytes, awaiting inclusion in a specific block. The mempool size basically serves as a proxy for the volume of activity on the blockchain. Most important of all, it can also showcase the fees needed for faster confirmation of transactions.

5. **Transaction Volume**

Transaction volume represents the overall value of outputs that have been confirmed on the blockchain in the recent 24 hours. The total transaction volume also adds up unspent outputs that are reverted back to the 'spending' wallet in the form of change.

6. **Estimated Transaction Volume**

Estimated transaction volume refers to the estimate of actual transaction volume shifted among unique wallets. The estimated transaction volume is the difference between transaction value and estimate of outputs returning to 'spending' wallet as change.

7. **Latest Blocks**

One of the most functional factors in the working of a blockchain explorer refers to latest blocks. The latest blocks functionality in block explorers could help in outlining the confirmed blocks, beginning from newest to oldest. The functionality can provide information about the blocks, such as timestamp, block size, and block height.

These are the common transaction statuses shown in a blockchain explorer

- **Pending**: The network is still processing the transaction.
- **Confirmed**: The transaction goes through different confirmation stages, and "confirmed" tells you that your transaction has been completely processed.

- **Complete**: At this point, the transaction is irreversible because the process is already complete.
- **Failed**: This shows that the transaction encountered some problems and cannot be continued.

How does a blockchain explorer work?

From a technical point of view, here's what it appears like:

- Application programming interfaces (APIs), rational databases, and SQL databases are used by blockchain explorers combined with a blockchain node to extract data from a network.
- This data is organized into a database by the software program, which then presents the data in a searchable manner.
- A straightforward user interface (think: search engine) that enables individuals to conduct searches can then be used by the explorer to do searches across a structured table in response to user demands.
- Through the creation of a web page, the explorer server can communicate with users.
- The explorer can communicate with other computers, all because of the API.
- The backend server receives search requests and answers to the user interface.
- In the end, in order to make the results easy to understand, the user interface and API deliver web pages in HTML format to the user's browser.

ETHERSCAN

Etherscan is one such effective block explorer that abstracts the complexity of retrieving data on the blockchain by providing a simple interface for Ethereum blockchain.

What is Etherscan?

Etherscan serves as a blockchain explorer that allows users to look through the transactions, smart contracts, wallet addresses, blocks, and other on-chain data related to Ethereum-based trades and assets.

Why Do You use Etherscan?

Etherscan allows users to search and explore the Ethereum blockchain for transactions, tokens, addresses, and other activities. Ether, an Ethereum token, is the world's second-largest cryptocurrency for market capitalization. Etherscan provides Token Tracker for this widely used token by providing information, such as price, holders, total supply, and transfers.

It allows to monitor high-profile NFT holders' portfolios. Moreover, you can track what a certain NFT project or company is doing with its funds.

Etherscan can be used to search for and validate Ethereum transactions. The use cases may involve:

- Searching and viewing wallets and transactions
- Tracking gas fees
- Interacting with smart contracts
- Revoking or reviewing your token approvals
- Using Token ignore list feature to hide your tokens
- Staying up-to-date with the Ethereum ecosystem

Etherscan combines different technologies, such as blockchain explorers, decentralized applications (Dapps), and Web3 libraries, and displays the information in an easy-to-access format.

The working of Etherscan is based on three parts:

- Data retrieval
- Data storage
- Data production

Data Retrieval:

Etherscan uses an Application Programming Interface (API) to access data. APIs allow one software to interact with another. In the case of Etherscan, the API connects block explorers to the Ethereum blockchain. Etherscan uses JSON-RPC 2.0 specification, which means it utilizes JSON (JavaScript Object Notation) to communicate with Ethereum. These APIs provide users access to the transactions and blocks on Ethereum.

Data Storage:

After retrieving data from the Ethereum blockchain, Etherscan stores the data in the form of relational tables in the "SQL Database". The relational format means the data in one table is related to the data in other tables. This relationship helps explorers to display information easily.

Data Production:

In this final phase, the data retrieved from the Ethereum blockchain and displayed in the Database is converted to a human-understandable language. Here, Etherscan takes the help of different programming languages, such as HTML, CSS, and JavaScript. HTML structures the data on block explorers, CSS gives styles to the content, while JavaScript helps data to interact with the user.

Ethstats is often used by cryptocurrency traders and investors to track the health of the Ethereum network and make informed decisions about buying or selling Ether (ETH), the cryptocurrency that powers the network. It can also be useful for developers who want to monitor the performance of their Ethereum-based applications.

Ethstats is a service that displays real time and historical statistics about individual nodes connected to a network and about the network itself. Individual node statistics include the last received block, block time, propagation time, connected peers, latency etc. Network metrics include the number of nodes, average block times, node geolocation, transaction counts etc.

These statistics are presented to the user in the form of a dashboard served to a web browser. This can be configured using the public Ethstats server for Ethereum mainnet or some public testnets, or using a local copy of Ethstats for private networks.

Ethstats

Ethstats has three components:

a server that consumes data sent to it by each individual node on a network and serves statistics generated from that data.

a client that queries a node and sends its data to the server

a dashboard that displays the statistics generated by the server

Through EthStats, the network monitoring dashboard, you can check the usage, performance, and overall state of whichever Ethereum-based network you are connected to in real time. Additionally, for each connected node you can analyze statistics or use the History functionality to replay node behavior. This feature allows you to start with a certain block and observe the order in which nodes recognized it as the highest in the network's chain.

R.M.K GROUP OF INSTITUTIONS

Here is a general overview of how Ethstats works:

Ethstats collects data from Ethereum nodes. Nodes are computers that run the Ethereum software and store a copy of the blockchain. Ethstats aggregates the data received from the nodes and displays it in an easy-to-understand format on its web-based dashboard. The dashboard provides information about the Ethereum network's overall health, including the number of nodes currently online, the number of transactions per second, and the average block time. Ethstats also provides users with the ability to view detailed information about individual transactions, blocks, and addresses. Ethstats uses a peer-to-peer network to share data between nodes. This helps to ensure that the data is accurate and up-to-date.

Steps to use the Ethstats website:

Open your web browser and navigate to the Ethstats website at https://ethstats.net.

Once on the website, you will see a dashboard that displays various statistics about the Ethereum network. The dashboard is divided into several sections, including Block Time, Gas Usage, Difficulty, and Network Hash Rate.

To view more detailed information about a specific statistic, click on the corresponding section of the dashboard. For example, to view more information about the Block Time, click on the Block Time section of the dashboard.

The website will display a chart or graph that shows the historical data for the selected statistic. You can adjust the time range of the chart using the dropdown menu at the top of the page.

To view information about a specific Ethereum node or miner, click on the "Nodes" tab at the top of the page. This will display a list of all the

R.M.K GROUP OF INSTITUTIONS

nodes and miners currently connected to the Ethereum network, along with their IP addresses, location, and other details.

You can filter the list of nodes and miners by selecting specific criteria from the dropdown menus at the top of the page. For example, you can filter the list to show only nodes running a specific version of the Ethereum software.

To view more detailed information about a specific node or miner, click on its name in the list. This will display a detailed page that shows information about the node's hardware and software configuration, its current status, and other details.

Finally, you can use the search bar at the top of the page to search for specific nodes, miners, or other information on the Ethstats website.

R.M.K GROUP OF INSTITUTIONS

Implementing DIApps

Evolution of decentralized applications

The following diagram shows how applications have evolved over the past three decades:

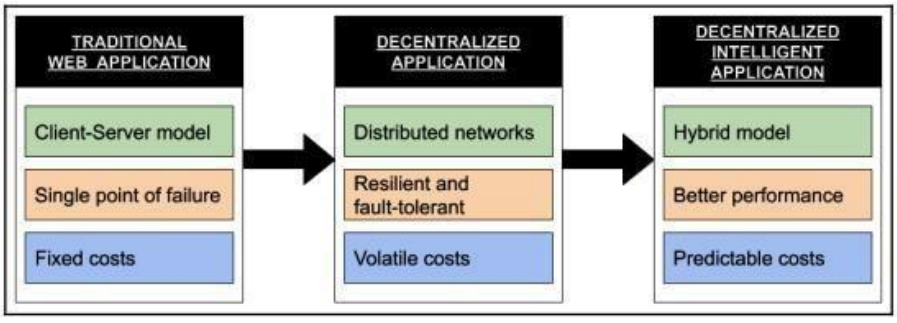


Fig 8.1: Evolution of applications since the dawn of the internet

Traditional web applications

Although prominent work for the foundation of the internet was laid out in the 1970s and the 1980s, it was the World Wide Web that made a significant leap in how information can be shared easily among nodes in a network. Traditional web applications such as blogs, chatrooms, and e-commerce websites emerged since the advent of the public internet in the mid-1990s. The majority of internet traffic is still driven by such applications, offering information and services to internet users. Most of these applications started with a simple web server in the backend. Servers were hosted to accept limited client connections across the world but easily crashed after reaching the desired threshold. Now, the majority of these applications have moved toward an n-tier architecture. The currently used n-tier architecture, also called multi-tier architecture or multi-layer architecture, is used to manage large-scale web applications such as e-commerce websites, social media sites, chatrooms, blogging platforms, and so on.

R.M.K GROUP OF INSTITUTIONS

Dedicated infrastructures are managed on the cloud in order for companies to facilitate transactions performed by users. Compared to the primitive version of the client-server model, the n-tier model separates the functions into many layers (hence the name). This also means that the n-tier model is serving as a better version of the client-server model in terms of handling failures and managing the liveness of traditional web applications. Companies manage their own infrastructure, such as servers, which is needed to provide services to users. Most of the infrastructure used by companies can be purchased or rented from cloud providers for a short period of time. This means the costs of operating traditional web applications are almost fixed.

Decentralized applications

The foremost characteristic of a decentralized application is the fact that it is not controlled by a single entity. Another important characteristic is that it facilitates a common bar of entry to use the app. By textbook definition, Napster and BitTorrent were some of the earliest ecentralized applications that could be used to host and share various files of sorts in a peer-to-peer format. Comparing decentralized applications with traditional web applications, we can see that decentralized applications run on more than one server or computer. Also, such computers are not necessarily owned and operated by a single entity or individual. They usually consist of interested parties who are willing to run the software for a benefit or an incentive. Hence, it is safe to conclude that most of the decentralized applications run on a peer-to peer network. Unlike traditional web applications, these applications do not experience a single point of failure. These peer-to-peer networks are defined by a protocol and are usually fault-tolerant to protect the users from many attack vectors. Compared to a client-server model, it is very difficult to compromise a peer-to-peer network due to its distributed topology.

R.M.K GROUP OF INSTITUTIONS

Incentivization and penalties in a blockchain protocol make it more difficult to compromise the network as the cost of an attack is much higher than the returns gained from such an attack. Although decentralized applications address the two aforementioned problems, the cost of running business and mission-critical applications is challenging due to the volatility associated with token prices. At the time of writing, efforts are being made to reduce this volatility in terms of signing the transaction on behalf of the user or paying the user in advance for the transactional costs. Decentralized intelligent applications Now that business models are leaning toward transparency and efficiency, there is a need for a new design pattern that favors a combination of traceability, decentralization, and predictability. Decentralized intelligent applications can use reliable infrastructure in a peer-to-peer network. This network is composed of nodes operated by more than one entity that may have vested interests in serving users and growing their businesses. Also, transactions on this kind of network are usually confirmed in a very short period of time at significantly lower fees. Similar to large public networks, it would be costly to attempt to compromise the network due to the deficit between the benefit of the attack and the cost of performing an attack. A key differentiator between this pattern and others is the closer integration of AI models with the help of decentralized databases. This makes it possible to build mission-critical and business-centric applications with traceability and insights as first-class features.

Contrast and analysis

In the case of traditional web applications, we may observe better performance when using a centralized and dedicated infrastructure. The transactions will achieve near-immediate finality at a reasonable and fixed cost for the company developing the app. However, this design

R.M.K GROUP OF INSTITUTIONS

pattern may lack some security and traceability features offered by other design patterns.

In the case of decentralized applications, we may observe reasonable performance and traceability at the expense of variable transaction costs. Compared to traditional web applications, the speed of transactions may be impacted. In the case of decentralized intelligent applications, we may observe predictable costs and near-immediate finality for most of the transactions. Apart from better cost, speed, security, and performance, the pattern also provides decentralized storage for building privacy preserving applications that can be used to derive actionable insights with ethical usage of AI models on user data.

	Traditional web application (App)	Decentralized application (DApp, dApp, or Dapp)	Decentralized intelligent application (DIApp)
Network	The client-server model is used with n-tier architectures.	A distributed network topology is used to allow anyone to join the network.	A distributed network topology is used to allow anyone to join the network.
Security	Single points of failure are very likely. Data can be hacked or leaked due to weak encryption or centralized control over data.	The user is the owner of the data. All the data and operations are secured by a unique key pair. Limited data can be stored.	Users can store and operate on larger sizes of data with the same key pair security in order to own their own data.
Cost	Fixed costs in managing a dedicated infrastructure throughout the year.	The costs of transactions are volatile since the costs depend on the price of the native token in the blockchain.	Prices are relatively stable in smaller yet decentralized groups of nodes.
Transparency	Apps and data operations are not transparent to the user or other stakeholders.	The logic of the app and most of the operations are transparent. Private transactions are optionally allowed.	Logic and operations are transparent, with options to use privacy without harming security.
Performance	A large number of transactions can be processed with immediate finality	The throughput is low due to the large distribution of nodes. Finality is slower.	Transaction throughputs are higher. Finality is also achieved quickly.
Privacy	Complete privacy and the anonymity of user data is seldom practiced by companies while harnessing insights as data is hosted on a centralized or distributed database controlled by organizations.	Anonymity is practiced through wallets, but data management on blockchain networks is a costly affair. The privacy of the user depends on the application's policies. AI models are not used regularly.	DIApps aim to provide complete privacy and anonymity to users while providing meaningful and actionable insights using AI in a fair manner, without hampering the anonymity of the user.

Building a sample DIApp

Problem statement

A novel coronavirus that goes by the name of Severe Acute Respiratory Syndrome coronavirus 2 (SARS-CoV-2) has created a new pandemic outbreak called Coronavirus Disease 2019 (COVID-19). At the time of writing this chapter, the virus has infected over 11 million people globally through various modes of transmission, tragically taking the lives of more than 500,000 people: a sad page in the history books of humanity. Although efforts have been made by local governments to reduce these infections, some virus carriers appear to be asymptomatic. This means that a person may be carrying the virus without knowing it. Sometimes, the prescribed checkups could also fail to recognize the virus in its early stage of incubating inside the patient's body.

Current challenges

This virus has introduced humanity to new challenges. Let's take a look at the two major challenges that are, at the time of writing, being faced. Detecting the virus in asymptomatic patients: As discussed previously, the SARS-CoV-2 virus poses a new challenge to medical professionals in terms of identifying infections in a human body that fails to show any symptoms. Such people may be allowed to continue their daily life, thus risking the wellness of the entire community they belong to. This leaves a gaping hole in security checks, which may allow asymptomatic people to access public services or interact with people who may potentially contract the virus from the patient.

Tracing the transmission of the virus: Although it is difficult to detect the virus in its early stages for a few people who are asymptomatic, symptoms do surface over time. Once they do, and the person tests positive, it is important to trace back all the actions of the diagnosed patient in order to contain the infection. This

R.M.K GROUP OF INSTITUTIONS

is difficult to achieve without an accurate history of the patient's activity over the past couple of weeks. Any efforts to jot it down will need time and will remain inaccurate due to human errors. To contain such infections, medical professionals resort to contact tracing.

Tracing the transmission of the virus: Although it is difficult to detect the virus in its early stages for a few people who are asymptomatic, symptoms do surface over time. Once they do, and the person tests positive, it is important to trace back all the actions of the diagnosed patient in order to contain the infection. This is difficult to achieve without an accurate history of the patient's activity over the past couple of weeks. Any efforts to jot it down will need time and will remain inaccurate due to human errors. To contain such infections, medical professionals resort to contact tracing.

Contact tracing

Contact tracing is a process of identifying all the people involved in the patient's activities over the past few days or weeks since the diagnosis of the infection. This is a process carried out by health department officials in coordination with law enforcement agencies.

A generic workflow of contact tracing is as follows:

1. The doctor has diagnosed the patient as positive for SARS-CoV-2.
2. A contact tracer is assigned to the case.
3. The contact tracer interacts with the patient to identify the activity of the patient.
4. Depending on the jurisdiction/country, the contact tracer is responsible for collecting accurate information about the patient's whereabouts for a designated number of days or weeks. For example, a patient that tests positive in India might be asked to share their activity for the past 14 calendar days.
5. Based on the input provided by the patient, the contact tracers may verify some

R.M.K GROUP OF INSTITUTIONS

information.

6. If the information shared by the patient is convincing, more contact tracers are hired to identify the first-degree people infected by the patient.

7. Once this manual search is over, the people who have been contacted are tested for the virus.

8. Depending on the jurisdiction, the people who have been contacted may be placed under mandatory quarantine for up to 14 days to check whether symptoms develop.

9. The quarantined people are tested for the virus periodically.

10. If there are no signs of the virus, the suspected people are released. However, if they test positive, the same process repeats.

Although you may find this process to be tightly planned and sophisticated, most of the steps mentioned here are manually carried out by many countries. Although some countries have opted to automate contact tracing with the help of digital technologies, they do not consider all the agents of infection.

Issues with contact tracing

it is very difficult to track infections from non-human sources. Hence, it is imperative to consider monitoring animals and non-living objects for infections of the SARS-CoV-2 virus. As such, there is a need for a digital contact tracing algorithm that can address the possibility of infections among these two agents in the ecosystem. With the need for digital contact tracing for animals and objects clearly established, let's try to formulate a solution approach.

Solution approach

Since we are developing a sample application to track potential infections from animals and non-living objects, name of the solution Decentralized Intelligent Contact Tracing for Animals and Objects (DICTAO). As discussed in the preceding section, there

R.M.K GROUP OF INSTITUTIONS

is a need to track animals and objects. We must be able to track down the infection status at
a granular level for the sake of transparency. A public blockchain with a decentralized and
open ledger can offer this feature. Similarly, we must understand that the global supply
chain is a busy world of its own. Manually tracking down all potential contacts is virtually
impossible. Similarly, it is very difficult to identify potential contacts between animals.
Hence, there is a need for an automated but intelligent way of identifying the potential
infections and separating them from the noise. Thus comes the need for blockchain and AI in tracing animals and objects.

Choosing the blockchain technology

As discussed previously, blockchains are essential in maintaining the transparency of the status of animals and objects. The Ethereum network has many testnets for developers to deploy and test their applications in a sandboxed test environment. One of the most famous test networks is called the Kovan testnet. Kovan is a Proof of Authority (PoA)-based Ethereum blockchain network. It is maintained by the Ethereum developer community. The Kovan testnet is known for its speed of execution, reliability, and free access to test ethers made available through a faucet. You can read more about the Kovan testnet here: https://kovan testnet.github.io/website/

Faucet is a piece of software used by smart contract developers and users to acquire testnet tokens for free, without the need for mining them on their local PCs. Most of the blockchain testnets have their own respective faucets.

R.M.K GROUP OF INSTITUTIONS

Choosing a decentralized database

Digital contact tracing often involves collecting data. To enable accuracy and provide high quality predictions, more data could be collected at a high frequency. Over time, data can become very large. Such data should not be stored on a blockchain directly since it incurs a high cost. Also, storing huge amounts of data can often lead to bottleneck issues and may hamper the performance of a blockchain. Hence, there is a need to store the activity of the animals or objects in a decentralized database. The solution approach uses MóiBit REST API as the decentralized database for this sample DIApp.

MóiBit offers a developer-friendly API environment, which makes it an easy choice. Since MóiBit is an IPFS-based decentralized storage service, each change to a new file or an existing file generates a new hash. Similar to blockchains, each new hash represents a successful state change. However, updating the data on MóiBit will be cheaper and faster compared to blockchains. Since it is driven by hashes, the file's integrity is also secured and easily verifiable.

Choosing an AI technique

Tracking innumerable objects and animals across an area – even a small one – will lead to a humongous quantity of data points. It is virtually impossible to perform contact tracing manually on these data points. Due to some low-quality data points, the efforts of a manual contact tracer could easily go to waste. As the number of positive cases increases, more pressure mounts on the manual contact tracers to close a case and move on to the next one. This could also lead to inaccurate identification of infections. In order to reduce errors and automate the process of cleaning, ordering, grouping, and predicting the infections, we can leverage AI techniques. As discussed previously, contract tracing usually involves the process of analyzing the activity of the infected person/animal/object. It is safe to assume that snapshotting the location data, along with a timestamp of when it occurred, will provide enough

R.M.K GROUP OF INSTITUTIONS

insights into potential contact cases. Therefore, we will be analyzing geolocation data.

We'll use geospatial analysis to identify some anomalies in the data points. Specifically, we will be using the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm to perform geospatial analysis to identify potential infections among animals and objects. It is a data clustering algorithm used to effectively group data points in a range under a cluster and drop the outliers that cannot be reached by other data points. To learn more about the DBSCAN algorithm, consult the following

Wikipedia article: https://en.wikipedia.org/wiki/DBSCAN.

Now that we have decided what technologies and techniques will be used in the sample DIApp, let's try to formalize it in the form of a reference technical architecture by borrowing it from the DIApp design pattern.

The technical architecture of the sample DIApp

Based on the decisions made in the preceding sections, I have compiled all the solution components into one diagram, as follows:

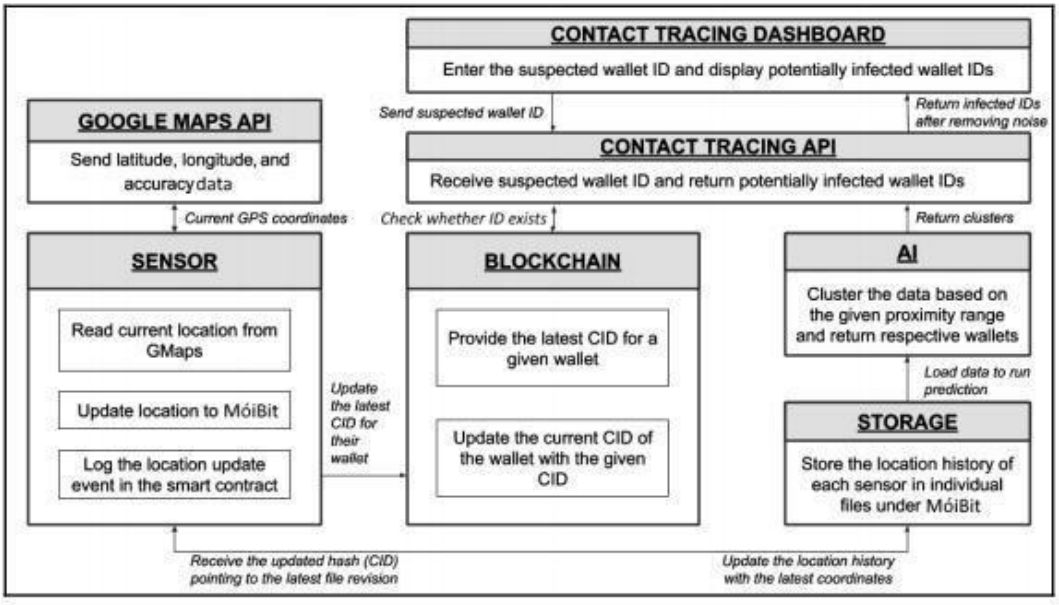


Fig 8.2: DIApp reference architecture of our sample application

In the preceding diagram, we have represented all the solution components as per the DIApp reference architecture proposed in the DIApp design pattern. Now, let's examine each component:

Sensor:

A sensor is a hardware device attached to the animal or object that needs to be tracked in case of infection. Each sensor is identified by a unique wallet address that is recognizable on the Ethereum blockchain. Also, it is important to note that each sensor gets a dedicated file on MóiBit where the location history of the corresponding sensor can be stored.An application will run inside the sensor. This application is expected to automatically read its location and update the location to its dedicated file on MóiBit. The current location of the sensor can be accessed by making a call to a geolocation API. In our sample DIApp example, we are using the Google Maps API to retrieve the current location of the sensor. When called, the Google Maps Geolocation API returns the latitude, longitude, and accuracy of the coordinates back to the sensor. For the sake of simplicity, we will be ignoring the accuracy value returned by the Google Maps API and uploading the rest of the data to MóiBit.

Since the example DIApp is trying to demonstrate the convergence of AI and blockchain technologies, optimizations are not given priority in this book. Now, once the sensor receives the response from the Google Maps Geolocation API, the data is restructured and uploaded to the dedicated location history file on MóiBit.

Smart contract in the blockchain:

The significance of the smart contract in this sample DIApp is to maintain proof for each location update performed by every sensor. Every time a sensor updates its location history on MóiBit, a new function call is made to the smart contract to update the Content Identifier (CID) value for the corresponding wallet

R.M.K GROUP OF INSTITUTIONS

designated for the sensor.

Contact tracing dashboard: The contact tracing dashboard is a simple Express-based Node.js web application that spawns a local server to host a single HTML file. Through the dashboard web page rendered by the Node.js application, users can enter the suspected wallet ID. The web application performs basic form validation and returns a list of potentially infected wallet IDs, if any.

Contact tracing API:

The user input to track a suspected wallet ID is received by the contact tracing API. This is a backend API that also predicts the infected IDs in order to return them to the dashboard.

Developing the smart contract

we will discuss the design and structure of the smart contract used to record the proof each time the location history is updated.

It is important that you understand the semantics of the Solidity programming language to understand this section and learn the benefits of a smart contract in the proposed DIApp solution.

We know that the solution aims to trace infections that originate from animals and objects. Hence, we understand that a tracking device or sensor is necessary. Let's assume that each sensor can be identified by a unique Ethereum wallet address. Apart from sensors, we also have end users such as medical rofessionals and enforcement teams who may want to identify the potentially infected actors from a given animal or object:

1. Each time the sensor has a new location to update on MóiBit, it makes the

necessary update and receives a hash back from MóiBit. We are using that hash

R.M.K GROUP OF INSTITUTIONS

and the wallet address from the calling sensor to maintain a mapping on the
smart contract. Each time a new location coordinate is updated by a sensor, the
corresponding value of the sensor's wallet address needs to be updated. Hence,
we need to define a mapping to record any changes that are made to the location
history of each sensor. We can do this with the following mapping:

```
mapping(address => string) private deviceIDToLatestCID;
```

From the preceding mapping, we can see that the address serves as a unique key
to an updated string value, which is basically the CID received from MóiBit when
the location of the corresponding sensor is updated.

2. Apart from the preceding mapping, we will also need another mapping to track
all the sensors by their addresses to confirm whether they have ever updated
their location. We can achieve this by persisting a list of all the wallet addresses
in the following mapping:

```
mapping(address => uint256) private deviceIDExists;
```


From the preceding mapping, we can see that every sensor that has ever updated
its history will be recorded in this mapping. We will learn more about the reason
behind this soon.

3. Finally, we will record the address of each and every sensor that has ever
updated its history to this smart contract with the following array of wallet
addresses:

```
address[] private deviceIDs;
```

From the preceding array declaration, designated users can access the list of all
the addresses that have ever updated their location history in the smart contract.
Do not confuse the deviceIDs array with the deviceIDExists mapping. The
array is used to directly access each and every sensor's wallet address, whereas
the mapping is used to check whether a sensor has already updated the location
history before.
Before we move on to the functional aspects of this smart contract, we have two
more declarations: a modifier and an event.
4. A modifier is a conditional instruction that must be fulfilled before executing a
function. If the conditions of a modifier for a function are not satisfied, that
function call will not happen. In our case, we use a modifier to control who can
update the location history mapping: deviceIDToLatestCID. As you may have
observed, this mapping needs to be updated with the CIDs pointing to the latest

R.M.K GROUP OF INSTITUTIONS

location history of a sensor. But we also need to make sure that only the sensor
can update its own value. Other sensors, users, or developers should not be
allowed to update the location history of an unknown sensor. This important
design decision will prevent other rogue actors from corrupting the reputation of
a good standing sensor on the blockchain.
To achieve this, we can use the following modifier:

```
modifier onlyBy(address _account) {
require(
msg.sender == _account,
"Sender not authorized to update this mapping!"
);
_; // The "_;"! will be replaced by the actual function body
when the modifier is used.
}
```

Implementing DIApps
Chapter 8

From the preceding modifier declaration, we can observe that a function with the
preceding modifier is executed only if the caller is the same as the address whose
location history is being updated in the blockchain. We will understand the
implication of this onlyBy mapping when we understand how the location is
updated by the setLatestCID function. But before that, let's quickly go through
one last declaration of an event.

5. Events are very helpful when working on complex use cases. As a user who may
be depositing some ethers or other tokens to a wallet, we wait for the transaction
receipt as confirmation. For confirmation of some logical execution, we can't just
wait for transaction receipts. There may be other sub-components that may have
to be triggered, depending on the successful execution of the logic. Events come
to our rescue here. Events in Solidity are a logging feature that helps non
blockchain applications take a cue point and continue with their execution.

We'll declare one event in our smart contract, as follows:

```
event MappingUpdated(address deviceID, string latestCID);
```

From the preceding event declaration, we can see that an event can be emitted with the wallet address of a sensor, along with its latest CID. This event is emitted every time the mapping is successfully updated with the new CID pointing to the latest location history of a sensor on MóiBit.

6. We will learn more about the application of this MappingUpdated event by looking in the setLatestCID function body, as shown here:

```
function setLatestCID(address deviceID, string memory latestCID)
public
onlyBy(deviceID)
{
deviceIDToLatestCID[deviceID] = latestCID;
if (deviceIDExists[deviceID] == 0) {
deviceIDs.push(deviceID);
deviceIDExists[deviceID] = 1;
```

R.M.K GROUP OF INSTITUTIONS

```
}
emit MappingUpdated(deviceID, latestCID);
}
```

As you can see from the preceding function declaration, setLatestCID is a setter function that allows each and every sensor to update its own location history by passing its wallet address, along with the CID pointing to the latest history on MóiBit. The address and string types are used to define the deviceID and latestCID input parameters. deviceID is the wallet address of the sensor, which is calling the function, while latestCID is the hash pointing to the latest history of the corresponding sensor on MóiBit. The public keyword defines that the function can be called by anyone globally. Thereafter, we see the onlyBy modifier being used to validate the function call. It takes the same input argument, deviceID, and checks whether the caller intending to update the location history is the sensor itself. If the modifier conditions are validated to be true, the remaining function body is executed. Otherwise, the transaction will be reverted.

Inside the function's body, we can observe that the latestCID value is assigned to deviceID immediately. Once the mapping is updated, it checks whether the sensor had previously updated its location. This is made possible by checking the status bit for the corresponding sensor's wallet address in the deviceIDExists mapping. If no entry exists for the given wallet address, it is added to the deviceIDExists mapping and the corresponding status bit value is set to 1.

Simultaneously, we can also observe that we are appending the deviceIDs array to the new array. Updating this array under this condition ensures that the wallet address is not added to the array again as a duplicate. This means that the setLatestCID function only appends the wallet address when a new sensor is onboarded to the smart contract. Finally, once the location mapping is updated and status bits are managed, the MappingUpdated event is emitted by the function.

R.M.K GROUP OF INSTITUTIONS

You can see that the input parameters have been supplied in the parentheses to log the event for corresponding values. This summarizes the details of the setLatestCID setter function.

7. Once we set the location history mapping with a new CID for a given sensor, we may have to read the mapping in case we need the details of a sensor. Hence, we will define a getter function to read the latest CID of the sensor from the mapping, as follows:

```
function getLatestCID(address deviceID)
public
view
returns (string memory latestCID)
{
return deviceIDToLatestCID[deviceID];
}
```

As you can see, the getter function, getLatestCID, reads one input parameter. The deviceID input parameter represents the wallet address of a sensor, with the Solidity type address. Since anyone should be able to read the proof that a sensor is updating its location from time to time, we have to make this getter function accessible globally. This is possible by using the public keyword. Also, since this is a function that only fetches the data from the blockchain and does not intend to make changes, it is also required to use the view keyword. This ensures that the getLatestCID function has read-only powers. Since we want anybody to be able to call the function, we do not have modifiers for this function. In the function's body, we can only see one line of instructions, and that is to return the CID value of the corresponding sensor from the deviceIDToLatestCID mapping. Since the returned value is a string that has been defined in the mapping, the function's header also defines the same. This summarizes the getter function, getLatestCID.

8. Now, let's look at the peripheral functions required by the backend scripts. We'll continue with one more getter function, as defined here:

R.M.K GROUP OF INSTITUTIONS

```
function getDeviceIDsLength() public view returns (uint256) {
return deviceIDs.length;
}
```

As you can see, getDeviceIDsLength is a getter function that does not take any input, but simply returns the current length of the deviceIDs array. Since we need to call this from a backend program, we have set the visibility of this function to public too. Similar to our previous function, this function is also a read-only function returning an unsigned integer value. Hence, view and uint256 are used in the function's header. This summarizes the getter function, getDeviceIDsLength.

9. Now, let's take a look at the last function in the contract:

```
function getIDByIndex(uint256 index) public view returns
(address) {
return deviceIDs[index];
}
```

As you can see, getIDByIndex is a getter function that returns the wallet address by the index value from the deviceIDs array.

Training the model

The steps to building our AI-based contact tracing algorithm are as follows:

1. Preparing the training dataset: The location history of each sensor is stored in a separate file under MóiBit. Each file serves as a subset of the main DataFrame and will be used to identify potential infections. In order to understand the location history and its applicability in DBSCAN, we generated a training dataset with preset random IDs, a timestamp, and lat-long values. It is not easy or safe to assign random values by ourselves. Hence, we used the JSON Generator tool. JSON Generator allows users to generate JSON documents with random values in a

customizable manner. This is made possible by programming the JSON generator to use specific values for a given field.Analyzing the training dataset: Now that we have created a training dataset with random alues and loaded it into Jupyter Notebook, we shall analyze it a bit further to understand the dataset. This process is called analyzing the training dataset. This step helps us understand more about the nature of the data points and how they are distributed.

First, we begin the analysis by describing how to get topline information about the dataset. We do this by calling the info function:

The output of the info function call also lists all the columns in the DataFrame, including their types. It also checks whether there are any null values. Since our schema for JSON Generator was very specific, we do not have any null values in any rows of the DataFrame.

Feature engineering: Feature engineering usually involves identifying critical data points, transforming the data points, and grooming them for a better analysis. Since there are no missing values or NaN values in our dataset, we will not be performing any featuring engineering on our dataset.

Exploratory data analysis: Since we are dealing with geographic data, it is better to understand the data points by plotting them against a real map. We are going to be using the Plotly library to plot the lat-long coordinates on a real map

Splitting the training dataset: Most of the time, we split the training dataset into two parts. One part is used to train the model, while the other part is used to predict the values and compare the predicted values with the actual values in the training dataset. Since we are clustering the data and not using regression-based models to actually predict a value, there is no need to split our training dataset.

R.M.K GROUP OF INSTITUTIONS

6. Selecting the model: When it comes to performing digital contact tracing for animals and objects, one approach is to use clustering algorithms that can provide customizations as per the new-found medical data and approaches. Although there are many clustering approaches, such as K-means, hierarchical clustering, and density-based clustering, we chose density-based clustering in this sample application as it is simple to understand and also offers some customizations that can be applied for practical use cases.

Training and fitting: Now that we have created the training dataset, analyzed it, and visualized it, we need to use the training dataset to train our model using the DBSCAN algorithm to cluster the data points and identify them distinctly. As per the medical norms accepted by many practitioners globally, it is a popular opinion that we can contract coronavirus if people are not maintaining the minimum safety distance of at least 6 feet. So, we are assuming the same accepted metric for physical distancing and creating our model so that it clusters data points that are connected to each other whose distance is less than or equal to 6 feet.

8. Evaluating the model: Usually, prediction models use logistic regressors or classifiers. But in our sample application, we are not predicting any new values; we're simply using machine learning to split the data into smaller, customized clusters.

Tuning the parameters: There are only three main input parameters that are required by the DBSCAN algorithm. They are epsilon, min_sample, and the haversine metric function. Apart from these parameters, you can also add some more parameters when you are trying this solution on your system.

R.M.K GROUP OF INSTITUTIONS

Predicting infections: The model is now ready to take live data from MóiBit and predict the potential infections by clustering the data points using the preceding parameters. Of course, the model will be prone to some false positives for neighboring data points of up to 10 meters as it is not fully tuned.

Developing the backend

The backend API is responsible for performing contact tracing by reading a wallet address and returning the wallet addresses of any other potentially infected sensors. we start by noting down all the import statements that are necessary for proceeding with the API's development:

```
import sys
import flask
from flask import request, jsonify
from web3 import Web3
import web3
import json
import http
from flask_cors import CORS
import datetime as dt
import pandas as pd
from sklearn.cluster import DBSCAN
```

We initiate the API by defining the flask web app, as follows:

```
app = flask.Flask(__name__)
CORS(app)
app.config["DEBUG"] = True
```

The entry point to this API script is made by using the following instruction in the script:

```
app.run()
```

When the script is run, it procedurally falls back to the home function defined, as follows:

```
@app.route('/', methods=['GET'])
def home():
return "<h1>DICTAO - Decentralized Intelligent Contact Tracing of Animals and Objects</h1><p>This is a simple demonstration of applying
blockchain, decentralized storage and AI to solve the COVID-19 crisis.</p>"
```

the home function is a handler function used to respond to the GET based API requests made to the root of the API. In this case, I am returning simple HTML content. To make sure that we send a proper response for any illicit request or invalid requests, we have defined the page_not_found handler function, as follows:

```
@app.errorhandler(404)
def page_not_found(e):
return "The given ID could not be found", 404
```

As shown in the preceding code block, this function returns a string response along with an HTTP response code 404 back to the client, which means file/resource not found. Apart from illicit or invalid client requests, we also need to cover some of the internal errors that may occur. This can be achieved by defining the internal_server_error function, as follows:

```
@app.errorhandler(500)
def internal_server_error(e):
return e, 500
```

R.M.K GROUP OF INSTITUTIONS

For each
wallet address registered in the smart contract, the latest location
history CID of each wallet
is retrieved from the blockchain. Furthermore, each CID is used to
retrieve the location
history data of each registered sensor from MóiBit.
The corresponding code can be seen in the following code block:

```
def getLatestCID(id):
w3 = Web3(Web3.HTTPProvider(blockchain_url))
contract = w3.eth.contract(
os.environ['PROOF_SMART_CONTRACT_ADDRESS'], abi=abi)
cid = ""
try:
cid = contract.functions.getLatestCID(id).call()
except web3.exceptions.ValidationError:
print("ID does not exist!")
return ""
except:
print("Some other error occured!")
return ""
else:
print(cid)
return cid
```

From the preceding code block, you can observe that the getLatestCID function is used to fetch the latest CIDs of the respective wallet addresses of the sensors, once the get_infections function retrieves each and every wallet address. The CID value read from the mapping in the

smart contract is returned to the caller function, get_infections. Now that the get_infections handle function contains the CID hashes of the corresponding wallet addresses of each and every registered sensor, it is used to retrieve the location history data from MóiBit, as follows:

```
def getJsonDataFromMoiBit(cid):
pre_payload = {"hash": cid}
payload = json.dumps(pre_payload)
conn.request("POST", moibit_url+"readfilebyhash",
payload, moibit_header_obj)
res = conn.getresponse()
if res.status == 200:
responseObject = json.loads(res.read())
print(
"updateLocationHistory(): Appending the captured data to
historic data.")
return responseObject
```

From the preceding code block, you can see that the retrieved cid from the getLatestCID
function is passed along to the getJsonDataFromMoiBit function. This CID is used to
retrieve the latest location history data of the corresponding sensors. Now that the data is available for analysis, the AI-based contact tracing algorithm we
designed in the previous section comes into the picture.

The AI model is incorporated in the following function:

```
def get_infected_ids(input_id):
basePath = os.path.dirname(os.path.abspath('live_dataset.json'))
dflive = pd.read_json(basePath + '/' + 'live_dataset.json')
epsilon = 0.0018288 # a radial distance of 6 feet, which is medically
presribed
min_sample = 2
```

R.M.K GROUP OF INSTITUTIONS

```
model = DBSCAN(eps=epsilon, min_samples=2,
metric='haversine').fit(dflive[['latitude', 'longitude']])
dflive['cluster'] = model.labels_.tolist()
input_id_clusters = []
for i in range(len(dflive)):
if dflive['id'][i] == input_id:
if dflive['cluster'][i] in input_id_clusters:
pass
else:
input_id_clusters.append(dflive['cluster'][i])
infected_ids = []
for cluster in input_id_clusters:
if cluster != -1:
ids_in_cluster = dflive.loc[dflive['cluster'] == cluster, 'id']
for i in range(len(ids_in_cluster)):
member_id = ids_in_cluster.iloc[i]
if (member_id not in infected_ids) and (member_id !=
input_id):
infected_ids.append(member_id)
else:
pass
return infected_ids
```

Developing the frontend

The purpose of the dashboard is to help us identify all the potentially infected IDs by entering the ID or wallet address of the suspected sensor that may be attached to an animal or an object. The dashboard application is simply composed of two components: an Express server that hosts the static files and an index.html HTML file that reads the input from a user, calls the contact tracing API, and prints all IDs

R.M.K GROUP OF INSTITUTIONS

returned by the backend API. The dashboard web server code is as follows:

```
const express = require('express')
const app = express()
const port = 3000
app.use(express.static('public'));
app.get('/', (req, res) => res.send('Welcome to DICTAO: Contact tracing web app!'))
app.listen(port, () => console.log(`DICTAO: Contact tracing web app listening at http://localhost:${port}`))
```

Testing the sample DIApp

Testing smart contracts: Truffle is one of the most renowned toolchains for Solidity smart contract development.

Testing sensor implementation: The sensor application is implemented using basic Python programming skills. You may have already observed that the script interacts with the Google Maps Geolocation API, Ethereum, and MóiBit.

Testing the AI model for accuracy:

Testing the AI models with Mean Absolute Error (MAE) is pretty simple and straightforward.

Testing the contact tracing backend API:

Visit the official testing documentation of Flask for more information on testing Flask web applications:

https://flask.palletsprojects.com/en/1.1.x/testing/

Testing the web dashboard frontend app:

Finally, the frontend web application is a simple piece of implementation. Test the inline JavaScript function in index.html to get better

R.M.K GROUP OF INSTITUTIONS

validation, pagination, and other edge cases that can make the UX better while you're presenting it

R.M.K GROUP OF INSTITUTIONS

Deploying the sample DIApp

Signing up for the Google Maps API

As you know, we use the Google Maps Geolocation API to get the current lat-long coordinates of the sensor. Hence, please follow the instructions in the following documentation and get yourself an API key: https://developers.google.com/maps/premium/apikey/geolocation-apikey.

Signing up for MóiBit

As you know, we use the MóiBit decentralized file storage API to store the location history data of each sensor. Hence, you are required to sign up for the MóiBit API. The signup process for MóiBit is very straightforward. You can sign up for MóiBit at the following link: https://account.moibit.io/#/signup. Once you've verified your email address and your password, a new API key will be generated for you.Implementing DIApps

Using these credentials, you are expected to create a new folder under the root folder. Please create a new folder there and name it dictao, as it is hardcoded into our current implementation. This makes sure that all the files will be persisted in a dedicated folder. This will also help you use MóiBit for other applications without any hassle or clutter. Again, make sure that your API key is not visible or accessible to the public.

Signing up for Infura

We use Infura to connect to the Ethereum Kovan blockchain. You need to create a new Infura account and create a new project. Once you've created a new project, you will need to copy the credentials for the project and use them to get dedicated access to the blockchain using Infura's infrastructure. The registration process for Infura is also pretty

R.M.K GROUP OF INSTITUTIONS

straightforward. You can sign up for an Infura account here: https://infura.io/register.

Updating your local justfile

manage an isolated file on the host that can privately share these credentials to the respective processes. To achieve this, we will be using the just command. You can install the just command by following the instruction available on GitHub:https://github.com/casey/just#installation

Please follow the installation instructions that fit your system the best, and make sure that you create a justfile, which is untracked by the git protocol. This is possible by adding the name justfile to the .gitignore file.

Fill in the necessary fields by replacing the question marks with the appropriate credentials for the services you have now signed up for:

```
export GMAPS_API_KEY := "?"
export MOIBIT_API_KEY := "?"
export MOIBIT_API_SECRET := "?"
export WEB3_INFURA_PROJECT_ID := "?"
export PROOF_SMART_CONTRACT_ADDRESS := "?"
export WALLET_PRIVATE_KEY := "?"
export WALLET_ADDRESS := "?"
run-client:
python iot-client-code/python/main.py
run-web:
cd frontend-tracking-dashboard && node index.js Implementing
DIApps  run-server:
python backend-contact-tracing/server.py
install-dependencies:
pip install --user -r requirements.txt
```

R.M.K GROUP OF INSTITUTIONS

cd frontend-tracking-dashboard && npm install

Depending on where you write the source code, you may need to change the relative paths of the source code files as well. Now, your justfile is ready to launch the necessary applications, along with your credentials

Deploying client code into sensors

You can deploy the sensor application by running the just command, as follows:

just run-client

If the credentials entered by you are valid and under the service quota, your client application will run. Also, make sure that the relative path to the Python script is updated in the just file.

Deploying the backend API

You can deploy the contact tracing backend API by running the just command, as follows:

just run-server

If the credentials entered by you are valid and under the service quota, your backend API will run. Also, make sure that the relative path to the Python script is updated in the just file.

Deploying the web dashboard

You can deploy the web dashboard application by running the just command, as follows:

just run-web

R.M.K GROUP OF INSTITUTIONS

If the credentials entered by you are valid and under the service quota, your dashboard application will run. Also, make sure that the relative path to the Node.js script is updated in the justfile.

Merits of the sample DIApp

Here are some of the merits of our proposed sample DIApp solution:

It covers other agents of infection, apart from humans.

It helps in restoring the global economy and normalcy.

It allows insurance companies and organizations to assess supply chain risks.

Limitations of the sample DIApp

Here are some of the limitations in our proposed sample DIApp solution:

The AI algorithm can be prone to some false positives. Optimization will be needed.

Due to a lack of hardware precision, software accuracy, and a better approach to computational complexity, the current implementation of the DIApp cannot be used in production.

The DIApp is unable to trace infections indoors as GPS is unable to identify the floor that the sensor is currently placed in. Other alternatives such as Wi-Fi, Bluetooth, a manual check-in register, and CCTV image analysis can be considered to boost the accuracy of the model.

R.M.K GROUP OF INSTITUTIONS

Assignments

Toppers

Analyze real-world examples of successful decentralized applications in various domains, such as gaming, supply chain management, and social networking. Evaluate the impact of decentralized apps on industries and society as a whole.

Above Average

Explain different governance models used in decentralized applications, such as on-chain governance and decentralized autonomous organizations (DAOs).

Evaluate the strengths and weaknesses of each governance model in ensuring transparency and accountability.

Average

Identify common security vulnerabilities in decentralized applications.

Discuss best practices for auditing and securing DApps to prevent vulnerabilities and exploits.

Below Average

Explore different development frameworks for building decentralized applications.

Compare popular frameworks such as Truffle, Embark, and Remix, highlighting their features and use cases.

Slow Learners

Define smart contracts and explain their role in decentralized applications.

Describe the Ethereum platform and its significance in enabling the development of DApps.

R.M.K GROUP OF INSTITUTIONS

PART - A

1. Write the steps to resolve issues:

1. Understand the business process:
2. Establish clear requirements:
3. Identify critical checkpoints in the business processes:
4. Check whether technology integration is feasible:
5. Establish technical dependency across affected components

2. Define internal technical dependency

Technical dependencies are internal if the development of a solution depends on its design, architecture, user stories, or acceptance criteria. The poor design of a given solution or information flow can cause many issues. Incomplete architectural decisions can also lead to issues in implementation.

3. What is DIApp?

Decentralized Intelligent Application (DIApp) is an enhanced pattern of a DApp that facilitates the application of AI wherever applicable, on top of a blockchain platform, in a much more robust manner that provides value to all stakeholders.

4. Differentiate DApp and DIApp

R.M.K GROUP OF INSTITUTIONS

DApp	DIApp
An application deployed on a blockchain platform with all the core logical elements.	Core logical elements are deployed on a blockchain platform, but are also powered by AI-driven insights in parallel.
Data is mostly sitting on the blockchain platform, making it costly in terms of expense and time for retrieval.	Critical provenance data resides on the blockchain, but the remaining data is persisted on a cheaper off-chain storage system.
Not all DApps are upgradeable, hence a new address is required for every update to the logic. This could break systems.	Most applications can easily be upgraded without making changes to the address, hence nothing breaks in the system.
Data is rarely analyzed due to the cost of read-writes in a blockchain platform, and also due to the lack of structured data required to train AI models.	Data can be easily structured in an off-chain storage system and tightly coupled back to the blockchain platform as well as the AI models.

5. Mention some tools for deploying DIApps.

Several tools such as Docker, Kubernetes, Ansible, Terraform, and Mesos are available for deploying DIApps and their dependencies

6. Write about two phases of testing AI models.

Testing AI models and capabilities can be split into two phases. One phase of the testing is before the model reaches production. The other phase of the testing is applied post-production. Before

7. What is the need for turing test in AI models?

Turing tests can also be performed to understand whether the AI model has sufficiently been able to replace a human response to the task.

8. Why Integration testing is needed fir DIApps?

a DIApp is a hybrid composition of many technologies. Hence, it is important to test the behavior of each heterogeneous component in order to ensure that they provide accurate results. This is achieved by performing integration testing. Once the components are integrated, we can run a few test cases that can formally verify the correctness of a

few critical components that rely upon one or more heterogeneous components to provide output

9. Define unit testing

Each component in the DIApp pattern delivers an exclusive value to the solution. Hence, it is important to ensure the correctness of each component in the solution. This can be made possible by performing unit testing.

10. What are the benefits of unit testing

we can test the core functionalities of each module before integrating it with the other modules. Unit testing helps in formally verifying the correctness of these modules by comparing the real output to the desired output. This process repeats for each change made to the module, thereby preserving the correctness of the module each time a change has been made.

11. Define Agile development

Agile software development refers to software development methodologies centered around the idea of iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams.

12. Define test case

The test case is defined as a group of conditions under which a tester determines whether a software application is working as per the customer's requirements

13. What is proof of concept?

Proof of concept, also known as POC or proof of principle, is the realization or demonstration of a certain idea in order to demonstrate its feasibility.

R.M.K GROUP OF INSTITUTIONS

14. Define DApps

Decentralized applications or dApps (also written as DApp or Dapp) are user applications that run on a platform hosted by a number of nodes in a distributed manner. DApps emerged as a solution architecture on top of blockchains wherein custom business logic could be programmed in a particular language of support.

15. What are the key points in solution architecture of a DApp

The solution architecture of a DApp primarily comprises the following key points:

The business logic is written in a Domain-Specific Language (DSL) in a smart contract.

The smart contract is deployed on a blockchain network, identified by an address.

The smart contract will acquire a new address if the business logic is to be updated in a majority of the blockchain platforms.

A web or mobile application is developed as a client to access and carry out operations on the business logic.

16. What are the key points in solution architecture of a DIApp

The solution architecture of a DIApp primarily comprises adding technical capabilities to make the pattern more approachable by enterprises. Here are the key highlights:

Business logic is usually written in any high-level language such as C, Python, Java, or Golang, among others supported by the Low-Level Virtual Machine (LLVM) compiler.

The majority of smart contracts are upgradeable, meaning that the code can be updated without having to change its address

Traditional client applications can also integrate with these types of smart contracts as there is less need for integrating it with newer middleware.

Critical business data is not only stored in blockchain networks but also in decentralized data storage systems to ensure better integrity of the data.

17. List few common attributes could help assert the product-market fit of a DIApp.

Does the DIApp solve a unique problem faced by various stakeholders in the industry?

Does the DIApp disintermediate a current entity that is inefficient in the current process, or bring more order to the process?

Can the benefits of DIApps be achieved only by utilizing the combination of blockchain and AI?

Does the DIApp facilitate users to derive insights by using the DIApp with the help of AI models?

18. Mention the technical and solution expertise required in the team for developing DIApps.

Two smart contract developers, One full-stack web developer, One SMACK-stack developer, One machine learning or deep learning engineer and one DevOps engineer

19. Mention some Blockchain platforms for smart contract development for DIApps

- Ethereum
- TRON
- EOS

20. **Give the steps in Smart Contract Development Process**

- **Define the business logic and requirements**

R.M.K GROUP OF INSTITUTIONS

The parties involved define the specific business conditions of the smart contract and communicate them to the developer.

- **Design the smart contract's architecture**

  Design the architecture of the contract to represent the business logic. This serves as a blueprint during the development process.

- **Develop the smart contract**

  This stage involves writing the smart contract code for the blockchain platform of choice

- **Internal audit**

  An internal audit via a local blockchain and then a testnet is performed. The audit checks if the contract functions as intended, and meets all security standards.

- **Deployment on the blockchain**

  The contract is deployed on the marketplace on which it will operate.

21. Define SMACK stack

The SMACK stack is a collection of technologies composed to build a resilient and distributed data processing architecture to enable real-time data-analysis and fast deployment. The acronym SMACK stands for the Spark engine, the Mesos manager, the Akka toolkit and runtime, the Cassandra database and the Kafka message broker.

22. what is a spark engine

Spark is an engine for large-scale data processing and makes it easy to build parallel applications or batch jobs. An advantage over Hadoop MapReduce is a superior performance. Users have the ability to query structured data using Spark SQL.

23. What is Mesos

R.M.K GROUP OF INSTITUTIONS

Mesos is a scheduling framework to manage clusters. It provides resources for applications, services, and jobs and abstracts the underlying hardware. The workload is distributed across the cluster. The distributed operating system DC/OS, built on top of Mesos, simplifies deployment and scaling of containerized applications.

24. Define Cassandra

Cassandra is a NoSQL database. It is a wide column store and can be seen as a two-dimensional key-value store. Cassandra's query operations are limited in order to guarantee performance and linear, horizontal scaling. There is no single point of failure and it's possible to span a Cassandra cluster across data centers.

25. What is Kafka

Kafka is a distributed messaging system which is well-known for low latency and high availability and throughput. Kafka is using a shared commit log internally. It can replay the log which is its unique selling point. Producers publish messages to publish-subscribe message queues. Kafka partitions the data within a topic. Partitions can be distributed to cluster nodes and consumers receive the requested messages.

R.M.K GROUP OF INSTITUTIONS

Part-B Questions

Q. No.	Questions	CO Level	K Level
1	Write the development lifecycle of DIApps	CO2	K2
2	Differentiate between Dapp and DIApp	CO2	K2
3	Explain testing DIAPP in detail	CO2	K2
4	Explain Designing DIApp and Developing DAApp in detail	CO2	K2
5	Explain Mionitoring and implementing DIApp in detail	CO2	K2

R.M.K GROUP OF INSTITUTIONS

Supportive online Certification courses (NPTEL, Swayam, Coursera, Udemy, etc.,)

R.M.K GROUP OF INSTITUTIONS

Supportive Online Certification Courses

Sl. No.	Courses	Platform
1	Blockchain Basics	Coursera
2	DeFi Decentralized Finance	Coursera
3	Blockchain	NPTEL
4	Blockchain and its Applications	NPTEL

R.M.K GROUP OF INSTITUTIONS

Real time Applications in day to day life and to Industry

R.M.K GROUP OF INSTITUTIONS

Uniswap

Launched in 2018, Uniswap is a US-based DApp on the Ethereum blockchain. The platform primarily allows users to swap and trade ERC-20 tokens. It is the most popular decentralized exchange and, overall, the fourth largest crypto exchange on the internet. The platform hosts more than 150k monthly users, with a $2.5 billion daily trading volume on average.

Perhaps the most special feature of Uniswap is that it does not rely on buyers and sellers to create liquidity. Anyone can access the platform by simply connecting a crypto wallet. For this purpose, the most common choice among Uniswap users is the MetaMask wallet. The exchange charges a flat 0.3% fee per trade, plus the Ethereum gas fee, which varies from time to time.

Aave

Aave is one of the most popular DApps in the DeFi world. It is an open-sourced liquidity protocol, providing users with complete transparency. It allows users to lend, borrow, stake, and earn interest on deposits. Thanks to the decentralized nature of the platform, both lenders and borrowers can enjoy complete anonymity.

Probably the most important use of Aave is to carry out flash loans. These loans are conducted within a few seconds and are necessary for the DeFi space to optimize the overall financial structure. AAVE charges a 0.09% fee on flash loans.

R.M.K GROUP OF INSTITUTIONS

R.M.K
GROUP OF
INSTITUTIONS

Content Beyond Syllabus

R.M.K
GROUP OF
INSTITUTIONS

Dapps in Healthcare Industry

Digital decentralization has opened doors to new possibilities in the health sector. Moving to a decentralized crypto-enforced data system has the potential to revolutionize the healthcare sector in tremendous ways.

1. Data exchange

The real-time information sharing among users is one of the best-selling points of decentralized applications in healthcare. Take prescribed medication as an example. While two patients may exhibit similar symptoms, different practitioners could use different treatments on them to achieve the same result. One prescription may be more effective than the other, but because both practitioners work without interaction, there's little way for one to learn from the other.

so that the exchange of such information can happen in real-time, from one end to the next. If a hospital makes a prescription, the data would be updated automatically for everyone using the Dapp to see it. The same goes for patients visiting hospitals having undergone treatment elsewhere. The medical practitioner would be able to find out exactly what was administered to the patient before, and make an informed, accurate decision.

2. Data security

Although Blockchain and decentralization present their own security and privacy concerns, they offer a much safer paradigm for storing and distributing digital information; one that is a lot more robust against risks like hacker attacks than traditional data systems.

- Cybercriminals successfully target hospitals in part because they store most of their data in discrete and isolated central servers, which can easily be compromised. They exploit the existence of a centralized data management system to lock a hospital out of its own server.
- In a decentralized world, all information is distributed and shared across many points on the network, which means there's no single point of failure. Hacker attacks would therefore become a lot more difficult.

3. Public health

- Having a decentralized network that connects numerous health practitioners and organizations which can prove very be very beneficial when dealing with epidemic cases. A shared, immutable and trustworthy stream of information about ongoing diagnostics could help to keep everyone on the same page at all times.
- Dapps could also facilitate the sharing of research, clinical trials, advanced directives and safety analyses, and therefore enhance collaboration.

4. Hospital administration

- Decentralized applications are poised to significantly streamline the communication among staff in health organizations. If all authorised workers in a hospital have direct access to data, they'll be able to work under much less supervision. Keeping everyone in the loop regarding hospital operations will make administration, along with managing daily processes like patient verification and insurance claims much easier.

5. Managing patient data

- Perhaps the biggest potential of decentralized applications in healthcare is that they could empower patients to gather, own and manage their data, rather than having it stored in an EHR (Electronic Health Record) system that is out of their reach. Patients could use personal health devices like fitness trackers and IoT devices to record data and share it with medical practitioners in real time.

R.M.K GROUP OF INSTITUTIONS

Assessment Schedule (Proposed Date & Actual Date)

R.M.K GROUP OF INSTITUTIONS

Assessment Schedule

Assessment Tool	Proposed Date	Actual Date	Course Outcome	Program Outcome (Filled Gap)
Assessment I	14.08.2025	14.08.2025	CO1, CO2	
Assessment II	23.09.2025	23.09.2025	CO3, CO4	
Model	28.10.2025	28.10.2025	CO1, CO2, CO3, CO4, CO5	
End Semester Examination	20.11.2025	20.11.2025	CO1, CO2, CO3, CO4, CO5	

R.M.K GROUP OF INSTITUTIONS

R.M.K GROUP OF INSTITUTIONS

Prescribed Text Books & Reference

R.M.K GROUP OF INSTITUTIONS

Prescribed Text & Reference Books

Sl. No.	Book Name & Author	Book
1	Ganesh Prasad Kumble, Anantha Krishnan, "Practical Artificial Intelligence and Blockchain: A guide to converging blockchain and AI to build smart applications for new economies", Packt Publications, 2020. (unit 1-5)	Text Book
2	Melanie Swan, "Block Chain: Blueprint for a New Economy", O'Reilly, 2015. (unit 5)	Text Book
3	Daniel Drescher, "Block Chain Basics", Apress; 1st edition, 2017	Reference Book
4	Salman Baset, Luc Desrosiers, Nitin Gaur, Petr Novotny, Anthony O'Dowd, Venkatraman Ramakrishna, "Hands-On Block Chain with Hyperledger: Building Decentralized Applications with Hyperledger Fabric and Composer", Import, 2018	Reference Book

R.M.K
GROUP OF
INSTITUTIONS

Mini Project Suggestions

R.M.K
GROUP OF
INSTITUTIONS

Toppers:

Develop an app that assists users in financial planning, budgeting, and investment decisions.

Above Average

Create an app to assist travelers with trip planning, destination information, and navigation.

Average

Create an app to facilitate language learning through interactive lessons and resources.

Below Average

Develop an app to help users with career planning, job search, and skill development.

Slow Learners

Develop an app that provides recipes, cooking tips, and assistance for home cooks.

R.M.K GROUP OF INSTITUTIONS

Thank you

Disclaimer:

This document is confidential and intended solely for the educational purpose of RMK Group of Educational Institutions. If you have received this document through email in error, please notify the system manager. This document contains proprietary information and is intended only to the respective group / learning community as intended. If you are not the addressee you should not disseminate, distribute or copy through e-mail. Please notify the sender immediately by e-mail if you have received this document by mistake and delete this document from your system. If you are not the intended recipient you are notified that disclosing, copying, distributing or taking any action in reliance on the contents of this information is strictly prohibited

R.M.K GROUP OF INSTITUTIONS