ABSTRACT

Crypto Currency, nowadays is the most demanding and many people are investing in these currencies to get high returns. CRYPTOFYI is an accessible, comprehensive platform designed as an application to simplify and demystify the world of cryptocurrency for both novice and intermediate users. In this app, the home page we will be displaying the current rates of different cryptocurrencies present in the crypto market along with search functionality. Top gain's and top losses of present crypto tokens are represented through the graphical representation along with currency rate. Another feature is favorite list by selecting a particular token we can add that into a list then it shows the complete information about that token.

Contents

LIST OF FIGURES

List of Tables

1. INTRODUCTION

CRYPTOFYI is based on the concept of the online currency.

Cryptocurrency started in India around 2009 with the introduction of Bitcoin. The first commercial transaction using cryptocurrency took place in 2010, when someone traded 10,000 Bitcoin for two pizzas. The first cryptocurrency exchange in india opened in 2013.

There are around 100 types of different cryptocurrencies, and millions of people who have invested in these currencies.

Cryptocurrency does not exist in physical form and is typically not issued by a central authority. Cryptocurrency typically use decentralized control as opposed to a central bank digital currency (CBDC). When cryptocurrency is created prior to issuance or issued by a single issuer, it is generally considered as centralized. When implemented with decentralized control, each cryptocurrency works through distributed ledger technology, typically a block chain , that serves as a public financial transaction database.

They are systems that allow for secure payments online which are denominated in terms of virtual "tokens", which are represented by ledger entries internal to the system. "Crypto" refers to the various encryption algorithms and cryptographic techniques that safeguard these entries, such as elliptical curve encryption, public-private key pairs , and hashing functions.

2. LITERATURE SURVEY

The **CRYPTOFYI** has been a standard measure for assessing cryptocurrency market trends and mining metrics since its development. However, its application in financial assessments has evolved, highlighting both its utility and limitations.

Historical Evolution of CRYPTOFYI

CRYPTOFYI was introduced as a means to study cryptocurrency market trends and dynamics. It gained prominence as a simple method to categorize and analyze data based on key metrics like profitability and market capitalization. Over time, CRYPTOFYI became a standard tool in financial analysis and public reporting for identifying trends in the cryptocurrency industry.

Technological Advancements in CRYPTOFYI Calculation

Advancements in technology have led to more precise methods of assessing cryptocurrency market data. Techniques such as block-chain analysis, artificial intelligence, and machine learning provide detailed insights into mining profitability, transaction patterns, and user behavior, offering a more comprehensive understanding of the industry's health and trends.

User Experience and Interface Design

The simplicity of CRYPTOFYI calculations—based on metrics like mining difficulty, hash rates, and market prices—has contributed to its widespread use. However, the increasing availability of digital tools has led to the development of user-friendly applications that incorporate CRYPTOFYI metrics alongside other financial data, enhancing user engagement and providing a more holistic view of the cryptocurrency market.

Educational Components of CRYPTOFYI Tools

Modern CRYPTOFYI tools often include educational resources to help users interpret their results. These resources may offer guidance on mining profitability, market trends, and the limitations of CRYPTOFYI as a sole indicator, promoting informed decision-making in cryptocurrency investments.

Global Inclusive and Unit Preferences

CRYPTOFYI calculations are standardized, but preferences vary globally depending on local currencies and mining regulations. This has led to the development of tools that accommodate both global and local metrics, ensuring accessibility and inclusive for a diverse user base.

Conclusion

CRYPTOFYI remains a valuable screening tool for identifying trends in cryptocurrency markets and mining. However, its limitations in assessing the complete financial landscape have prompted the development of more advanced methods and tools. Ongoing research and technological advancements continue to refine our understanding of cryptocurrency market dynamics, underscoring the importance of using CRYPTOFYI in conjunction with other assessments for a comprehensive evaluation.

3. DESIGN ANALYSIS

3.1 System Architecture:

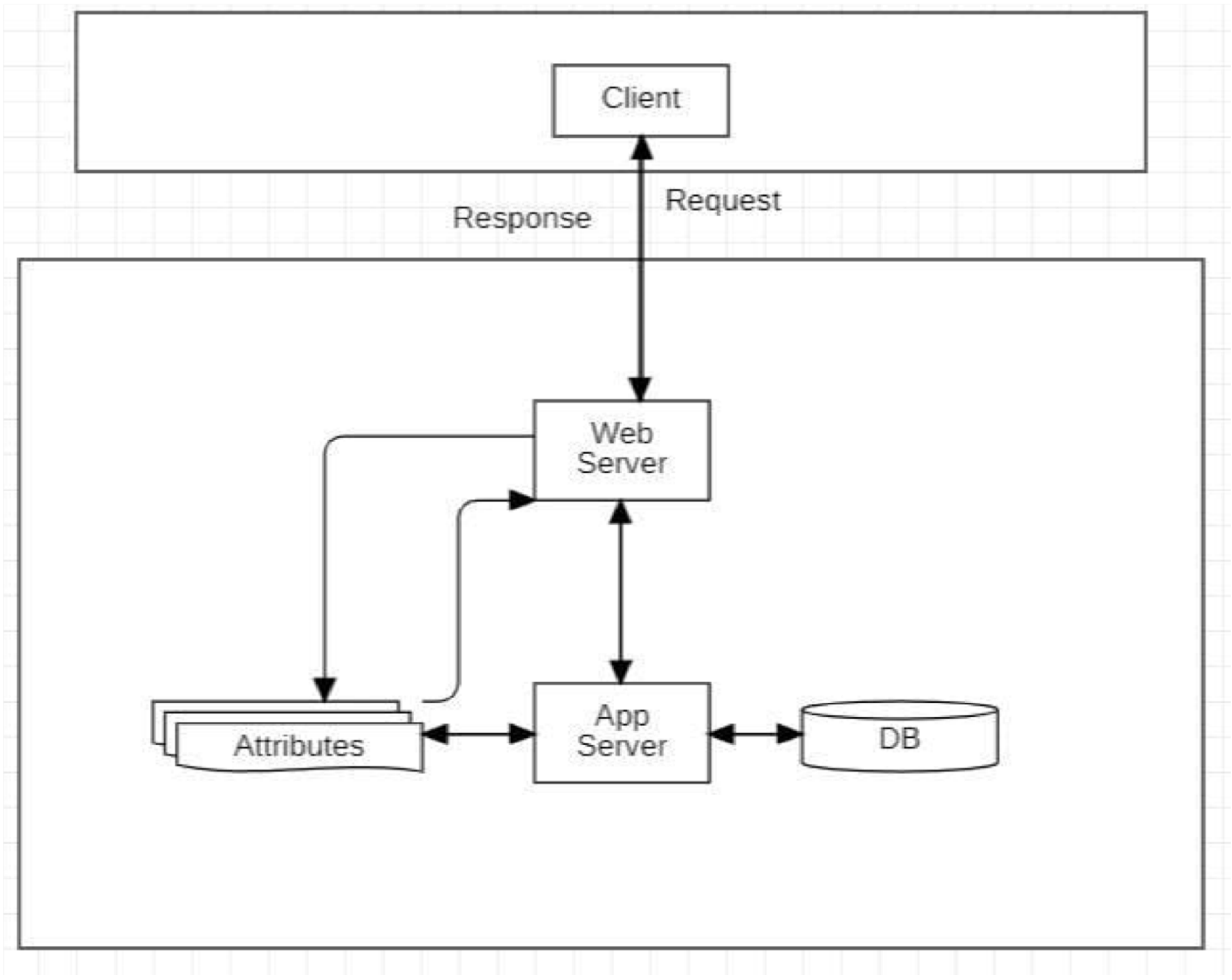
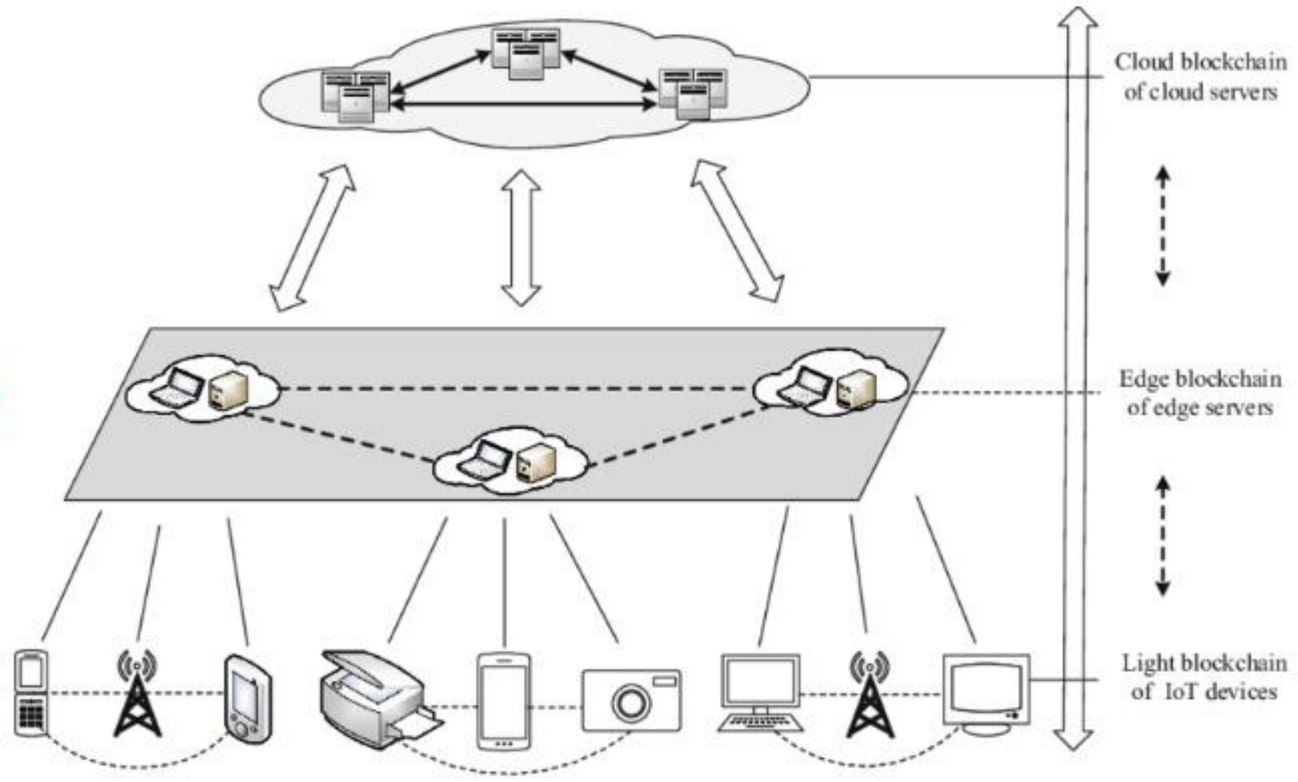


Fig 3.1 System Architecture


Cloud blockchain of cloud servers
Edge blockchain of edge servers
Light blockchain of IoT devices

The architecture of the edge computing integrated with blockchain.

3.2 UML Diagrams:

UML stands for unified modeling language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standardized is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or association with UML. It is important to distinguish between the UML model and the set of diagrams of a system. A diagram is a partial graphic representation of a system's model. The set of diagrams need not completely cover the model and deleting a diagram does not change the model. The model may also contain documentation that drives the model elements and diagrams. In 1997 UML was adopted as a standard by the Object Management Group (OMG), and has been managed by this organization ever since. In 2005 UML was also published by the International Organization for Standardization

(ISO) as an approved ISO standard. Since then it has been periodically revised to cover the latest revision of UML.

UML is not a development method by itself; however, it was designed to be compatible with the leading object-oriented software development methods of its time, for example OMT, Booch method, Objectory and especially RUP that it was originally intended to be used with when work began at Rational Software. UML (Unified Modeling Language) is a standard notation for the modeling of real-world objects as a first step in developing an object-oriented design methodology. The major perspectives of a UML are Design, Implementation, Process and Deployment. The center is the Use Case view which connects all these four.

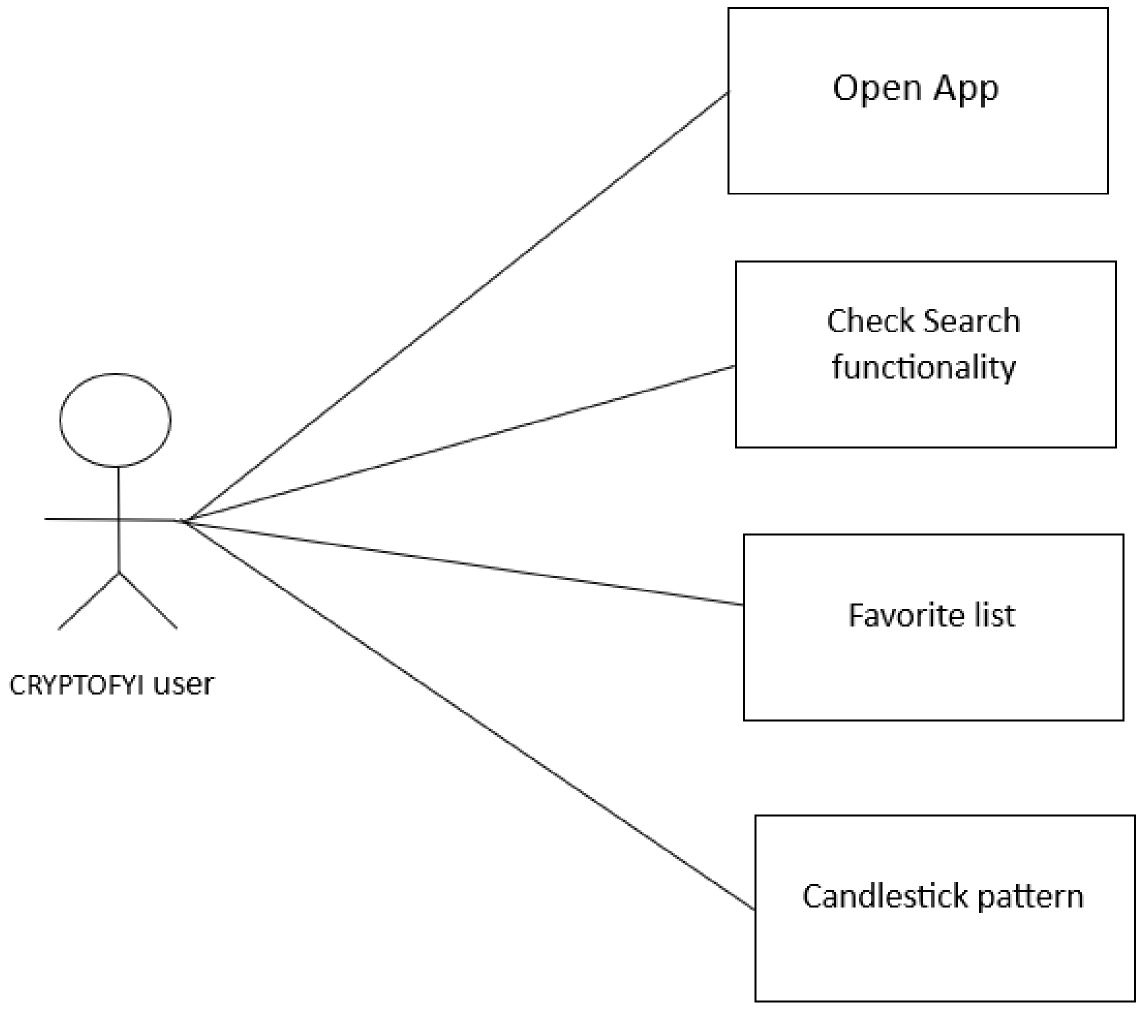
- Use Case Diagram
- Sequence Diagram
- Activity Diagram
- State Chart Diagram

3.2.1 Use Case Diagram

- A Use case represents the functionality of the system.
- Use Case diagrams in the UML is type of behaviour diagrams defined by and created from a

Use-case

.



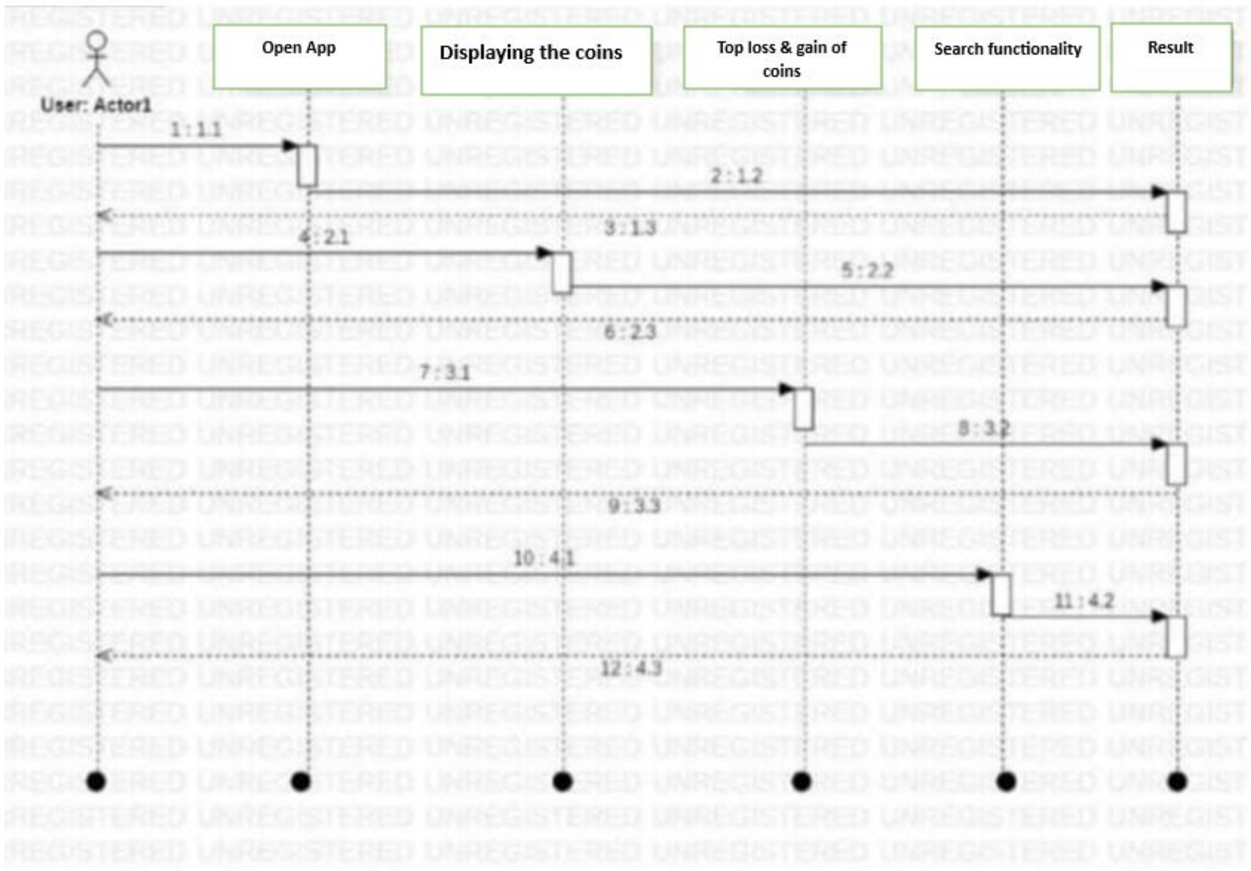
- It is used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).

- The main purpose of a Use case diagram is to show what system functions are performed for which actor.
- Use case diagrams are drawn to capture the functional requirements of a system.

Fig 3.2.1 Use Case Diagram

3.2.2 Sequence Diagram:

- A sequence diagram shows object interactions arranged in time sequence.
- It depicts objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.
- Sometimes called as event diagrams, event scenarios, timing diagram.
- A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order.


User: Actor1
Open App
Displaying the coins
Top loss & gain of coins
Search functionality
Result
1 : 1.1
2 : 1.2
3 : 1.3
4 : 2.1
5 : 2.2
6 : 2.3
7 : 3.1
8 : 3.2
9 : 3.3
10 : 4.1
11 : 4.2
12 : 4.3

•

3.2.3 Activity Diagram

Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. It can be used to describe the business and operational step-by-step workflows of components in the system.

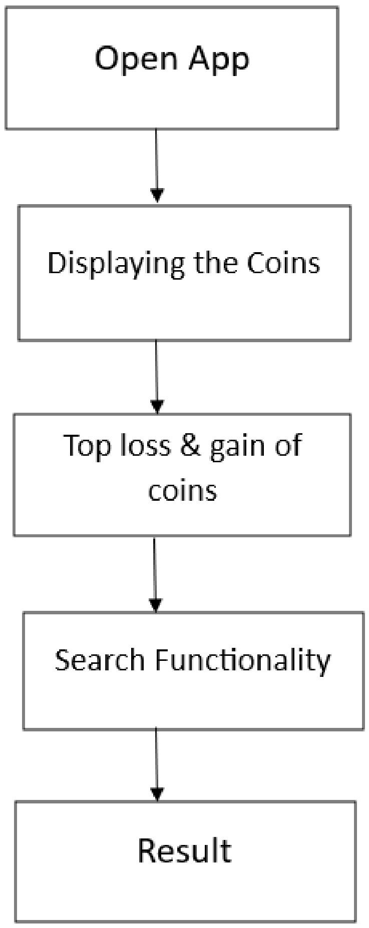


Fig: Activity Diagram

3.2.4 State Chart Diagram

A State chart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events.

State chart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of State chart diagram is to model lifetime of an object from creation to termination.

State chart diagrams are also used for forward and reverse engineering of a system. However, the main purpose is to model the reactive system.

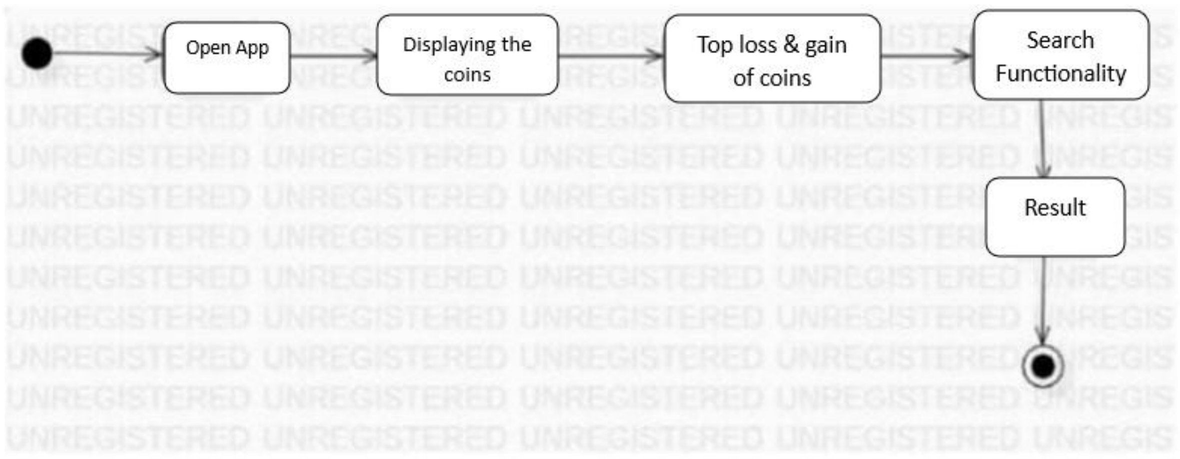


Fig: State Chart Diagram

4. Implementation

4.1 MODULES

This project consists of the following module:

- User module

4.2 MODULE DESCSRIPTION

4.2.1 User Module

User should open the website then the Entry page will be opened after that we need to fill the
The sequence to fill the details are:

- Enter your register credentials
- Select Coin
- Check if it is added to favourite list
- Data Presented

5.System Study

5.1 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Operational Feasibility
- Economic Feasibility
- Technical Feasibility

5.1.1 Operational Feasibility

Operational Feasibility deals with the study of prospects of the system to be developed. This system operationally eliminates all the tensions of the Admin and helps him in effectively tracking the project progress. This kind of automation will surely reduce the time and energy, which previously consumed in manual work. Based on the study, the system is proved to be operationally feasible.

5.1.2 Economic Feasibility

Economic Feasibility or Cost-benefit is an assessment of the economic justification for a computer based project. As hardware was installed from the beginning & for lots of purposes thus the cost on project of hardware is low. Since the system is a network based, any number of employees connected to the LAN within that organization can use this tool from at anytime. The Virtual Private Network is to be developed using the existing resources of the organization. So the project is economically feasible.

5.1.3 Technical Feasibility

According to Roger S. Pressman, Technical Feasibility is the assessment of the technical resources of the organization. The organization needs IBM compatible machines with a graphical web browser connected to the Internet and Intranet. The system is developed for platform Independent environment. Java Server Pages, JavaScript, HTML, SQL server and Web Logic Server are used to develop the system. The technical feasibility has been carried out. The system is technically feasible for development and can be and can be developed with existing facility.

6. GRAPHICAL USER INTERFACE

6.1 Input Design

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.

Validations are required for each data entered. Whenever a user enters an erroneous data, error message is displayed and the user can move on to the subsequent pages after completing all the entries in the current page.

6.2 Output Design

The cryptocurrency management system is designed to establish efficient communication and seamless management between the system administrator and users, such as investors, miners, and traders. It provides robust functionality for user account management, secure transactions, and resource allocation. The administrator oversees critical tasks like user validation, role assignment, and wallet management. Users can independently register, but validation and access to specific functionalities remain under the administrator's control.

The platform facilitates wallet and mining project assignments, monitors performance, and ensures that resources are reallocated efficiently after a project's completion. It implements stringent access control measures, restricting resource access based on user roles and enforcing secure login mechanisms like two-factor authentication to safeguard user data and transaction integrity.

The system operates efficiently once the server is initialized, allowing users to access the platform through a browser or dedicated client application. Designed to work seamlessly over a network, the server acts as the central hub, while other connected systems serve as client nodes.

With its user-friendly interface, even first-time users can navigate the platform with ease. The system ensures encrypted communication channels for secure data transfer and offers dynamic task reassignment to optimize user engagement and resource utilization. This cryptocurrency management system enhances operational efficiency, security, and scalability, meeting the evolving needs of decentralized financial ecosystems.

7. TESTING

7.1 System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.2 Testing Methodologies

The following are the Testing Methodologies:

- Unit Testing.
- Integration Testing.
- User Acceptance Testing.
- Output Testing.
- Validation Testing.

7.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing. During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

7.2.2 Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The

main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.
The following are the types of Integration Testing:

1. Top Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

2. Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub- function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.

7.2.3 User Acceptance Testing

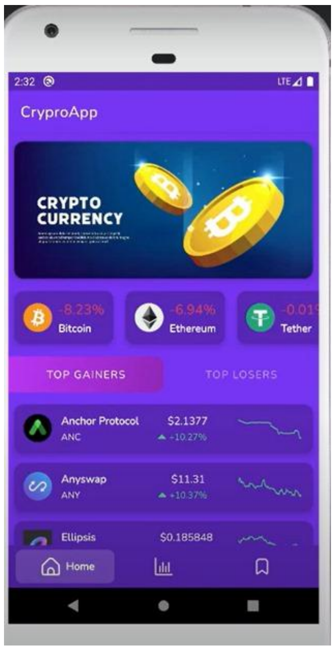
User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

7.2.4 Test Cases

Test Case : Cryptocurrency Price Tracking

Field	**Details**	**Case Description**	**Status**
Case 1	Step-1	Navigate to the market section	
	Step-2	Displaying the List of Crypto Tokens	Coins displayed
	Step-3	Select a Crypto Token to view its price trends	Pass
	Step-4	Provide Search functionality to search the specific token	Pass
	Step-5	Candle Stick pattern of coin	Pass

Screenshots:


2:32 LTE

CryproApp

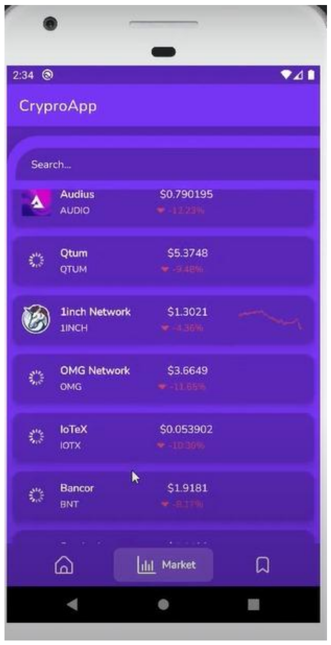
CRYPTO CURRENCY

-8.23% Bitcoin
-6.94% Ethereum
-0.01 Tether

TOP GAINERS TOP LOSERS

Anchor Protocol ANC	$2.1377	+10.27%
Anyswap ANY	$11.31	+10.37%
Ellipsis	$0.185848	

Home

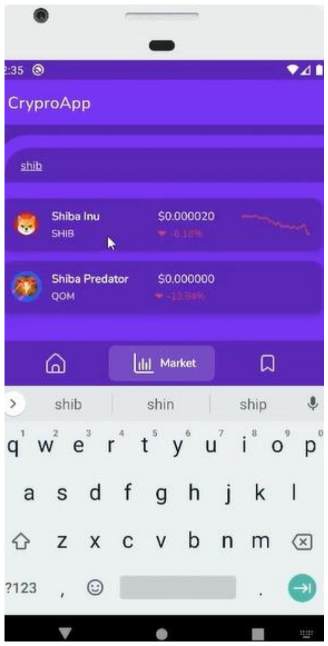

2:34

CryproApp

Search...

Audius AUDIO	$0.790195	-12.23%
Qtum QTUM	$5.3748	-9.48%
1inch Network 1INCH	$1.3021	-4.36%
OMG Network OMG	$3.6649	-11.65%
IoTeX IOTX	$0.053902	-10.36%
Bancor BNT	$1.9181	-8.17%

Market


2:35

CryproApp

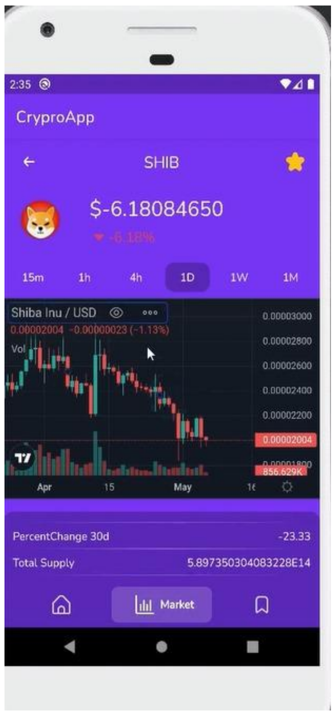
shib

Shiba Inu SHIB	$0.000020	-6.18%
Shiba Predator QOM	$0.000000	-13.94%

Market

shib shin ship

q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123 , .


2:35

CryproApp

SHIB

$-6.18084650

-6.18%

15m 1h 4h 1D 1W 1M

Shiba Inu / USD
0.00002004 -0.00000023 (-1.13%)

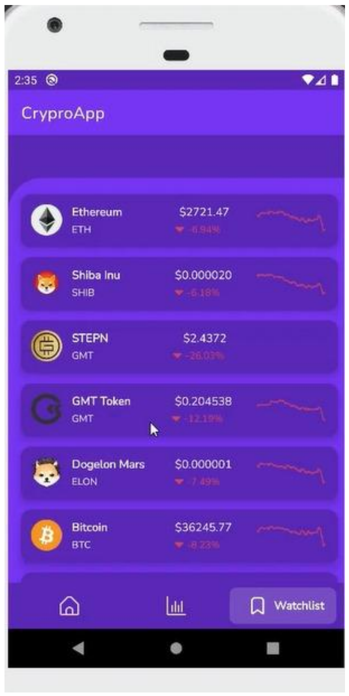
Vol

0.00003000
0.00002800
0.00002600
0.00002400
0.00002200
0.00002004
0.00001800
856.629K

Apr 15 May 16

PercentChange 30d	-23.33
Total Supply	5.897350304083228E14

Market


2:35

CryproApp

Ethereum ETH	$2721.47	-6.94%
Shiba Inu SHIB	$0.000020	-6.18%
STEPN GMT	$2.4372	-26.03%
GMT Token GMT	$0.204538	-12.19%
Dogelon Mars ELON	$0.000001	-7.49%
Bitcoin BTC	$36245.77	-8.23%

Watchlist

CONCLUSION

CRYPTOFYI is designed to serve as an accessible, comprehensive platform for users at both novice and intermediate levels, aiming to simplify the complex world of cryptocurrency. With the rapid growth and increasing popularity of digital currencies, many individuals are keen to invest in these assets. However, navigating the intricacies of the crypto market can be overwhelming for beginners and even experienced users. CRYPTOFYI addresses this challenge by providing a user-friendly interface that offers real-time cryptocurrency data, clear market insights, and valuable features to help users make informed decisions. The home page, displaying current rates of various cryptocurrencies, offers a streamlined experience that allows users to stay updated with minimal effort. Furthermore, the graphical representation of top gainers and losers, along with detailed market information, enables users to track trends, market shifts, and price movements effectively.

Another key feature of CRYPTOFYI is the ability to create a personalized favorites list, which enhances the user experience by allowing individuals to track their preferred tokens. Once a cryptocurrency is added to the favorites list, users can access comprehensive information about that token, including its current price, market capitalization, and historical performance. This feature provides a more tailored approach to monitoring the market and allows users to focus on assets they are most interested in. As cryptocurrency continues to grow as an investment vehicle, having a platform like CRYPTOFYI can help users navigate the market with ease, make timely investment decisions, and potentially capitalize on price movements. In essence, CRYPTOFYI simplifies the process of learning about and investing in cryptocurrency while providing the necessary tools for users to stay informed and engaged in the digital currency market.

In conclusion, our CRYPTOFYI app for Android provides a secure, user-friendly, and feature-rich platform for users to manage their digital assess. It provides an Intuitive interface and advanced features to suit various needs and It provides real-time market data.

With the growing interest in digital currencies, a well-designed and secure cryptocurrency app can offer significant value to users looking to manage their crypto assets efficiently.

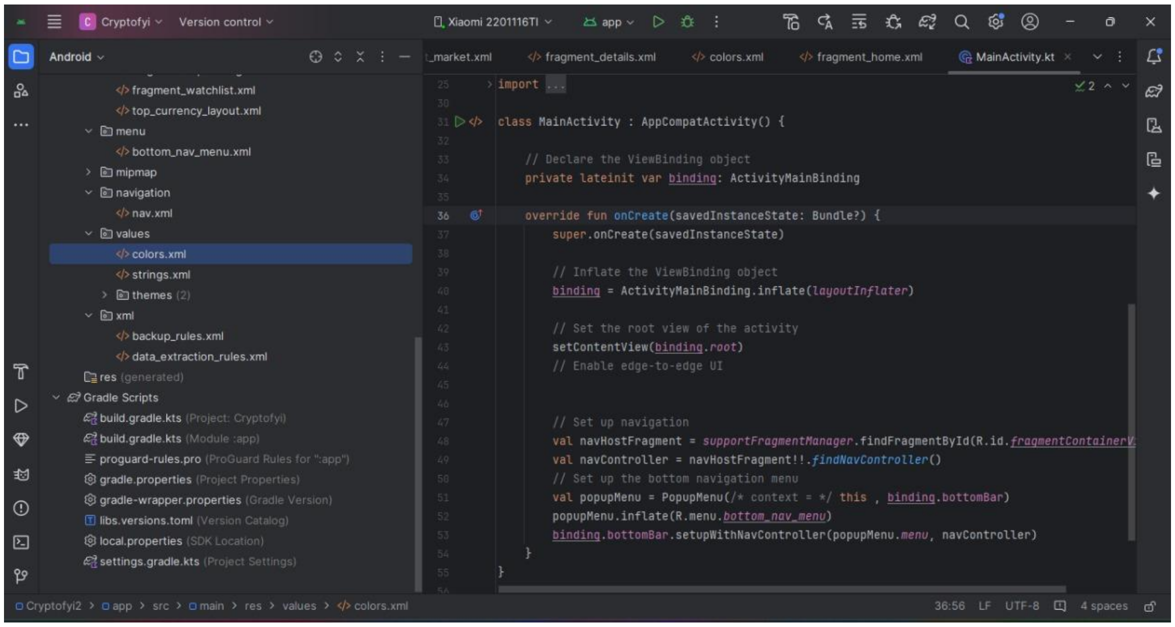
REFERENCES

- https://kotlinlang.org/docs/releases.html"
- Chatgpt
- Youtube

`

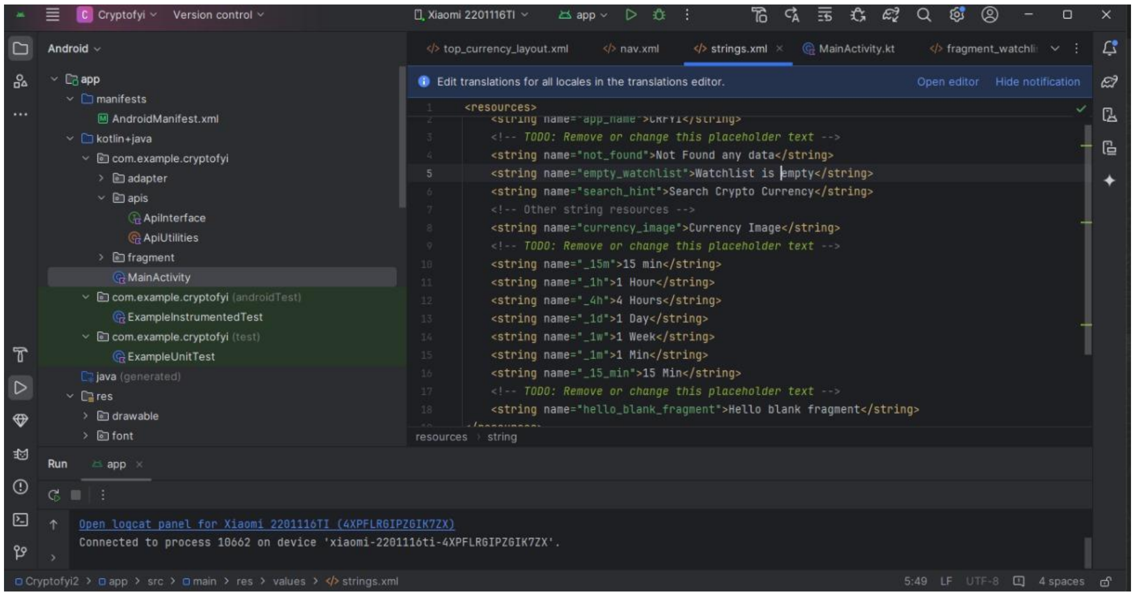
APPENDIX – A

Sample Code



```kotlin
import ...

class MainActivity : AppCompatActivity() {

    // Declare the ViewBinding object
    private lateinit var binding: ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        // Inflate the ViewBinding object
        binding = ActivityMainBinding.inflate(layoutInflater)

        // Set the root view of the activity
        setContentView(binding.root)
        // Enable edge-to-edge UI


        // Set up navigation
        val navHostFragment = supportFragmentManager.findFragmentById(R.id.fragmentContainerV
        val navController = navHostFragment!!.findNavController()
        // Set up the bottom navigation menu
        val popupMenu = PopupMenu(/* context = */ this , binding.bottomBar)
        popupMenu.inflate(R.menu.bottom_nav_menu)
        binding.bottomBar.setupWithNavController(popupMenu.menu, navController)
    }
}
```

Main Activity



Edit translations for all locales in the translations editor.

```xml
<resources>
    <string name="app_name">CRFYI</string>
    <!-- TODO: Remove or change this placeholder text -->
    <string name="not_found">Not Found any data</string>
    <string name="empty_watchlist">Watchlist is empty</string>
    <string name="search_hint">Search Crypto Currency</string>
    <!-- Other string resources -->
    <string name="currency_image">Currency Image</string>
    <!-- TODO: Remove or change this placeholder text -->
    <string name="_15m">15 min</string>
    <string name="_1h">1 Hour</string>
    <string name="_4h">4 Hours</string>
    <string name="_1d">1 Day</string>
    <string name="_1w">1 Week</string>
    <string name="_1m">1 Min</string>
    <string name="_15_min">15 Min</string>
    <!-- TODO: Remove or change this placeholder text -->
    <string name="hello_blank_fragment">Hello blank fragment</string>
```

```
Open logcat panel for Xiaomi 2201116TI (4XPFLRGIPZGIK7ZX)
Connected to process 10662 on device 'xiaomi-2201116ti-4XPFLRGIPZGIK7ZX'.
```

xml

APPENDIX B

Software and Hardware Requirements

Software Requirements

Operating System	: Android Studio
Technology	: Kotlin+java
Kotlin Version	: 17

Hardware Requirements

RAM	: 1GB
Hard Disk	: 40GB

APPENDIX - C

Technology Used

Kotlin

Kotlin, a statically-typed programming language, has emerged as a significant player in the software development landscape since its introduction by JetBrains. Designed to be fully interoperable with Java, it has garnered widespread adoption, particularly in Android app development, where it is now an official language supported by Google.

At its core, Kotlin aims to enhance developer productivity by offering a concise and expressive syntax. The language is lauded for its ability to eliminate boilerplate code, making it more readable and reducing the likelihood of errors. Kotlin's concise syntax allows developers to express ideas more succinctly compared to Java, fostering a more enjoyable and efficient coding experience.

Interoperability with Java is a standout feature of Kotlin. This compatibility is a strategic move that allows developers to seamlessly integrate Kotlin into existing Java projects. It means that Kotlin can leverage Java libraries, frameworks, and tools, easing the transition for teams and projects that are already invested in Java. This interoperability has been a key factor in Kotlin's adoption, as it mitigates the risks and challenges associated with introducing a new language into an established codebase.

Addressing one of the common pain points in programming, Kotlin incorporates robust null safety features. In Kotlin, variables are non-nullable by default, reducing the possibility of null-related runtime errors. This design choice promotes safer and more predictable code, contributing to overall software reliability.

Kotlin introduces extension functions, allowing developers to augment existing classes with new functionalities without modifying their source code. This feature promotes a more modular and readable design, enabling developers to extend the capabilities of classes in a clean and concise manner. The ability to enhance classes without resorting to inheritance or modification of existing code aligns with modern software design principles.

A notable addition to Kotlin is its coroutine support, which simplifies asynchronous programming. Coroutines provide a more readable and concise alternative to traditional callback-based approaches, making it easier for developers to manage asynchronous tasks without the complexities of nested callbacks or the need for extensive thread management.

Beyond Android development, Kotlin's versatility has led to its adoption in various domains. It is increasingly used for server-side development, with frameworks like Ktor gaining popularity.

Additionally, Kotlin/Native enables the compilation of Kotlin code to native binaries, expanding its reach to platforms beyond the Java Virtual Machine (JVM).

The language's growing ecosystem, backed by strong community support and endorsement from major tech companies, positions Kotlin as a language with a promising future. Its modern features, seamless integration with Java, and ability to address common pain points make it an appealing choice for developers aiming to build robust and maintainable software across diverse domains. As the programming landscape continues to evolve, Kotlin's influence is expected to persist and shape the way developers approach software development.