Alphagraphy

A PROJECT REPORT

Submitted By

Vanshita Gupta
(University Roll No- 2000290140130)
Nitin Goyal
(University Roll No- 2000290140079)
Anuj Goel
(University Roll No- 2000290140023)
Hritick Varshney
(University Roll No- 2000290140052)

Submitted in partial fulfilment of the Requirements for the Degree of

MASTER OF COMPUTER APPLICATIONS

Under the Supervision of
Naresh Chandra
Professor

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206

(JAN 2022)

CERTIFICATE

Certified that **Vanshita Gupta (Enrollment No-200029014005815), Anuj Goel (Enrollment No-200029014005708) Nitin Goyal (Enrollment No- 200029014005764) , Hritick Varshney (Enrollment No-200029014005737)** have carried out the project work having "**Title of Report Alphagraphy**" for Master of Computer Applications from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself /herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date: 13/01/2022

Vanshita Gupta (University Roll No- 2000290140130)
Anuj Goel (University Roll No- 2000290140023)
Nitin Goyal (University Roll No- 2000290140079)
Hritick Varshney (University Roll No- 2000290140052)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:
13/01/2022

Mr. Naresh Chandra
Professor
Department of Computer Applications
KIET Group of Institutions, Ghaziabad

Signature of Internal Examiner **Signature of Internal Examiner**

Dr. Ajay Shrivastava
Head, Department of Computer Applications
KIET Group of Institutions, Ghaziabad

ABSTRACT

Bitcoin and other prominent cryptocurrencies have gained much attention since the last several years. Globally known as digital coin and virtual currency, this cryptocurrency is gained and traded within the blockchain system. The blockchain technology adopted in using the cryptocurrency has raised the eyebrows within the banking sector, government, stakeholders and individual investors. The rise of the cryptocurrency within this decade since the inception of Bitcoin in 2009 has taken the market by storm. Cryptocurrency is anticipated as the future currency that might replace the current paper currency worldwide. Even though the interest has caught the attention of users, many are not aware of its opportunities, drawbacks and challenges for the future. Researches on cryptocurrencies are still lacking and still at its infancy stage. In providing substantial guide and view to the academic field and users, this project will discuss the opportunities in the cryptocurrency such as the security of its technology, low transaction cost and high investment return.

ACKNOWLEDGEMENTS

Success in life is never attained single handedly. My deepest gratitude goes to my project supervisor, **Naresh Chandra** for his guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Ajay Kumar Shrivastava, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help at various occasions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Vanshita Gupta

Anuj Goel

Nitin Goyal

Hritick Varshney

TABLE OF CONTENTS

1. Introduction

A crypto currency is a digital asset designed to work as a medium of exchange that uses cryptography to send its transactions . In the current Digital era, cryptocurrency is moving fast led by Bitcoin which was created in 2009 was the first decentralised crypto currency. Bitcoin is followed by Ether, Bitcoin etc are all taking the financial storm and influencing the public to invest and buy these currencies. Government of various countries are much concerned about crypto currency such as Bitcoin. The significant feature of these currencies are payments can be made without the involment of bank .customers can transfer the huge sum of money through the digital wallets. Central bank of various countries like Bank of England and Bank of isral are trying to launch their own digital currencies. this will help people using the official system which has the benefit of both traditional and crypto currencies. The traditional system for electronic payments and transfers and security checks on each transaction by banks consumes much time and cost. Whereas the crypto currencies help to process these transactions much faster and the transactions would be recorded instantly and need not be cleared by Banks instead technology known as Block chain is used.

1.1 Project description

The Project titled as ALPHAGRAPHY and is used to show various artefact of all the Digital Currency. Alphagraphy is the name of Application built for various O/S which helps the user to know indepth details of currency, it includes Charts , price fluctuation , Rates in different worldwide used currencies , News of exchanges , News related to Virtual Currency etc. Thus it help the people to get quick overview of market, helps them in better decision making , Moreover it help the people to grow their income.

1.2 Project Scope

The Project aims to build an application that provides people all the latest news of Digital Currency , exchanges news , live price of coins , fundamentals of different currencies , Charts of cryptos and in future will also be used in Paper Trading.

1.3 Project Definition

Definitions:

- **Developers:** Project developers can go through this document and understand the requirements presented

Processor	MT6762 (Helio P22) Processor or above
Storage	Between 50MB and 100 MB
Ram	Minimum 40 MB
Screen resolution	Automatic adjustable as per the device

- **Users:** Users, mainly people who are experiencing or going through with depression or anxiety, can refer this document to get a better idea of the functioning of the DIGIBUDD and determine whether the requirements listed here have been satisfactorily implemented.

- **Testers:** Testers are required to go through the requirements very carefully to design test cases and test the end product to ensure that the requirements presented by the client have been implemented and the software is working efficiently.

1.4 Hardware and software use in project

Hardware Specification

Software Specification

Operating System	**Windows 7 and above , IOS 5 and above**
IDE	**Visual Studio Code**
Backend	**PHP | MYSQL**
Frontend	**HTML, CSS, JS, BOOTSTRAP**

2. Feasibility Study

A feasibility analysis is used to determine the viability of an idea, such as ensuring a project is legally and technically feasible as well as economically justifiable. It tells us whether a project is worth the investment—in some cases, a project may not be doable. There can be many reasons for this, including requiring too many resources, which not only prevents those resources from performing other tasks but also may cost more than an organization would earn back by taking on a project that isn't profitable.

A well-designed study should offer a historical background of the business or project, such as a description of the product or service, accounting statements, details of operations and management, marketing research and policies, financial data, legal requirements, and tax obligations. Generally, such studies precede technical development and project implementation.

2.1 Technical feasibility

This assessment focuses on the technical resources available to the organization. It helps organizations determine whether the technical resources meet capacity and whether the technical team is capable of converting the ideas into working systems. Technical feasibility also involves the evaluation of the hardware, software, and other technical requirements of the proposed system. As an exaggerated example, an organization wouldn't want to try to put Star Trek's transporters in their building—currently, this project is not technically feasible.

2.2 Operational Feasibility

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

2.3 Economical Feasibility

In Economic Feasibility study cost and benefit of the project is analyzed. Means under this feasibility study a detail analysis is carried out what will be cost of the project for development which includes all required cost for final development like hardware and software resource required, design and development cost and operational cost and so on. After that it is analyzed whether project will be beneficial in terms of finance for organization or not.

2.4 Behavioral Feasibility

It evaluates and estimates the user attitude or behaviour towards the development of new system. It helps in determining if the system requires special effort to educate, retrain, transfer, and changes in employee's job status on new ways of conducting business.

3. Database Design

It is a collection of processes that facilitate the designing, development, implementation and maintenance of enterprise data management systems. Properly designed database are easy to maintain, improves data consistency and are cost effective in terms of disk storage space. The database designer decides how the data elements correlate and what data must be stored.

The main objectives of database design is to produce logical and physical designs models of the proposed database system.

- The logical model concentrates on the data requirements and the data to be stored independent of physical considerations. It does not concern itself with how the data will be stored or where it will be stored physically.

- The physical data design model involves translating the logical DB design of the database onto physical media using hardware resources and software systems such as database management systems (DBMS).

3.1 Use Case Diagram

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

The purpose of use case diagram is to capture the dynamic aspect of a system. However, this definition is too generic to describe the purpose, as other four diagrams (activity, sequence, collaboration, and State chart) also have the same purpose. We will look into some specific purpose, which will distinguish it from other four diagrams.

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

When the initial task is complete, use case diagrams are modelled to present the outside view.

In brief, the purposes of use case diagrams can be said to be as follows −

i. Used to gather the requirements of a system.

ii. Used to get an outside view of a system.

iii. Identify the external and internal factors influencing the system.

iv. Show the interaction among the requirements are actors.

How to draw a Use Case diagram?

It is essential to analyze the whole system before starting with drawing a use case diagram, and then the system's functionalities are found. And once every single functionality is identified, they are then transformed into the use cases to be used in the use case diagram. After that, we will enlist the actors that will interact with the system. The act ors are the person or a thing that invokes the functionality of a system. It may be a system or a private entity, such that it requires an entity to be pertinent to the functionalities of the system to which it is going to interact. Once both the actors and use cases are enlisted, the relation between the actor and use case/ system is inspected. It identifies the no of times an actor communicates with the system. Basically, an actor can interact multiple times with a use case or system at a particular instance of time. Following are some rules that must be followed while drawing a use case diagram:

i. A pertinent and meaningful name should be assigned to the actor or a use case of a system.
ii. The communication of an actor with a use case must be defined in an understandable way.
iii. Specified notations to be used as and when required.
iv. The most significant interactions should be represented among the multiple no of interactions between the use case and actors.

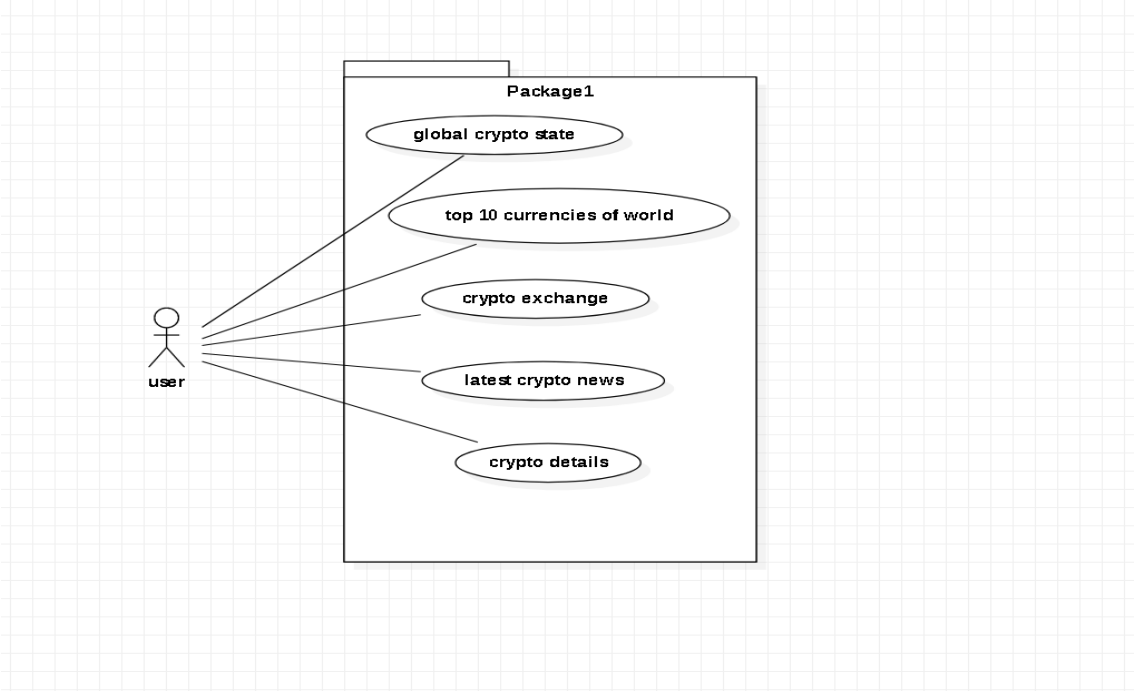

Fig 3.1.1: Use Case Diagram

3.2 Sequence Diagram

A sequence diagram or **s**ystem sequence diagram (SSD) shows object interactions arranged in time sequence in the field of software engineering. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

For a particular scenario of a use case, the diagrams show the events that external actors generate, their order, and possible inter-system events. All systems are treated as a black box; the diagram places emphasis on events that cross the system boundary from actors to systems. A system

sequence diagram should be done for the main success scenario of the use case, and frequent or complex alternative scenarios.

i. To model high-level interaction among active objects within a system.
ii. To model interaction among objects inside a collaboration realizing a use case.
iii. It either models generic interactions or some certain instances of interaction.

Sequence Diagram Notations –

i. **Actors –** An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram
ii. **Lifelines –** A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram.
iii. **Messages –** Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.
iv. **Guards –** To model conditions we use guards in UML. They are used when we need to restrict the flow of messages on the pretext of a condition being met. Guards play an important role in letting software developers know the constraints attached to a system or a particular process.

Uses of sequence diagrams –

i. Used to model and visualise the logic behind a sophisticated function, operation or procedure.
ii. They are also used to show details of UML use case diagrams.
iii. Used to understand the detailed functionality of current or future systems.
iv. Visualize how messages and tasks move between objects or components in a system.

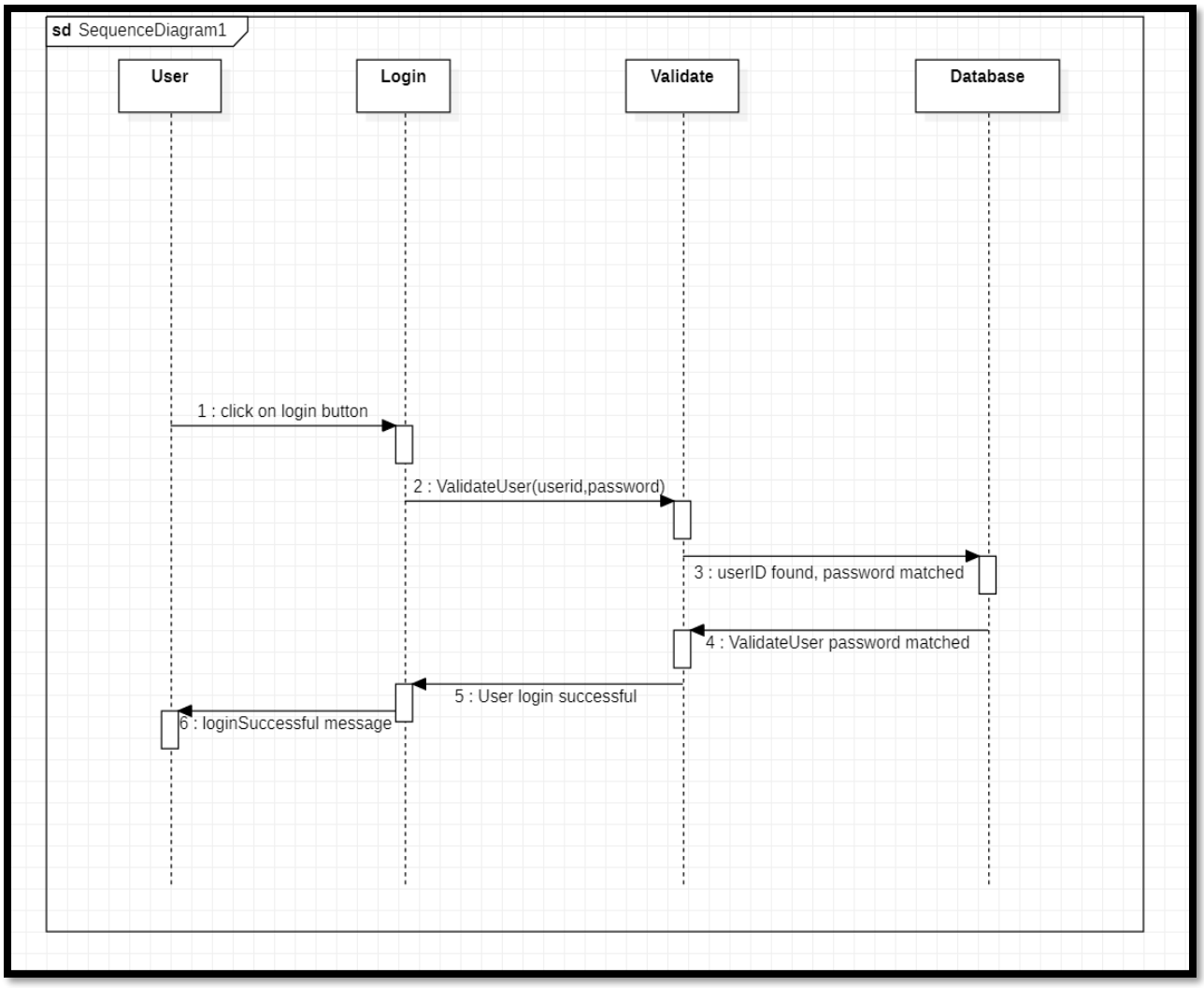

Fig 3.2.1: Sequence Diagram for login

3.3 Activity Diagram

The activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models the concurrent and sequential activities.

The activity diagram helps in envisioning the workflow from one activity to another. It put emphasis on the condition of flow and the order in which it occurs. The flow can be sequential, branched, or concurrent, and to deal with such kinds of flows, the activity diagram has come up with a fork, join, etc.

It is also termed as an object-oriented flowchart. It encompasses activities composed of a set of actions or operations that are applied to model the behavioral diagram

Why use Activity Diagram?

An event is created as an activity diagram encompassing a group of nodes associated with edges. To model the behavior of activities, they can be attached to any modeling element. It can model use cases, classes, interfaces, components, and collaborations.

It mainly models processes and workflows. It envisions the dynamic behavior of the system as well as constructs a runnable system that incorporates forward and reverse engineering. It does not include the message part, which means message flow is not represented in an activity diagram.

It is the same as that of a flowchart but not exactly a flowchart itself. It is used to depict the flow between several activities.

How to draw an Activity Diagram?

An activity diagram is a flowchart of activities, as it represents the workflow among various activities. They are identical to the flowcharts, but they themselves are not exactly the flowchart. In other words, it can be said that an activity diagram is an enhancement of the flowchart, which encompasses several unique skills.

Since it incorporates swim lanes, branching, parallel flows, join nodes, control nodes, and forks, it supports exception handling. A system must be explored as a whole before drawing an activity diagram to provide a clearer view of the user. All of the activities are explored after they are properly analyzed for finding out the constraints applied to the activities. Each and every activity, condition, and association must be recognized.

After gathering all the essential information, an abstract or a prototype is built, which is then transformed into the actual diagram.

Following are the rules that are to be followed for drawing an activity diagram:

i. A meaningful name should be given to each and every activity.
ii. Identify all of the constraints.

iii. Acknowledge the activity associations

b.

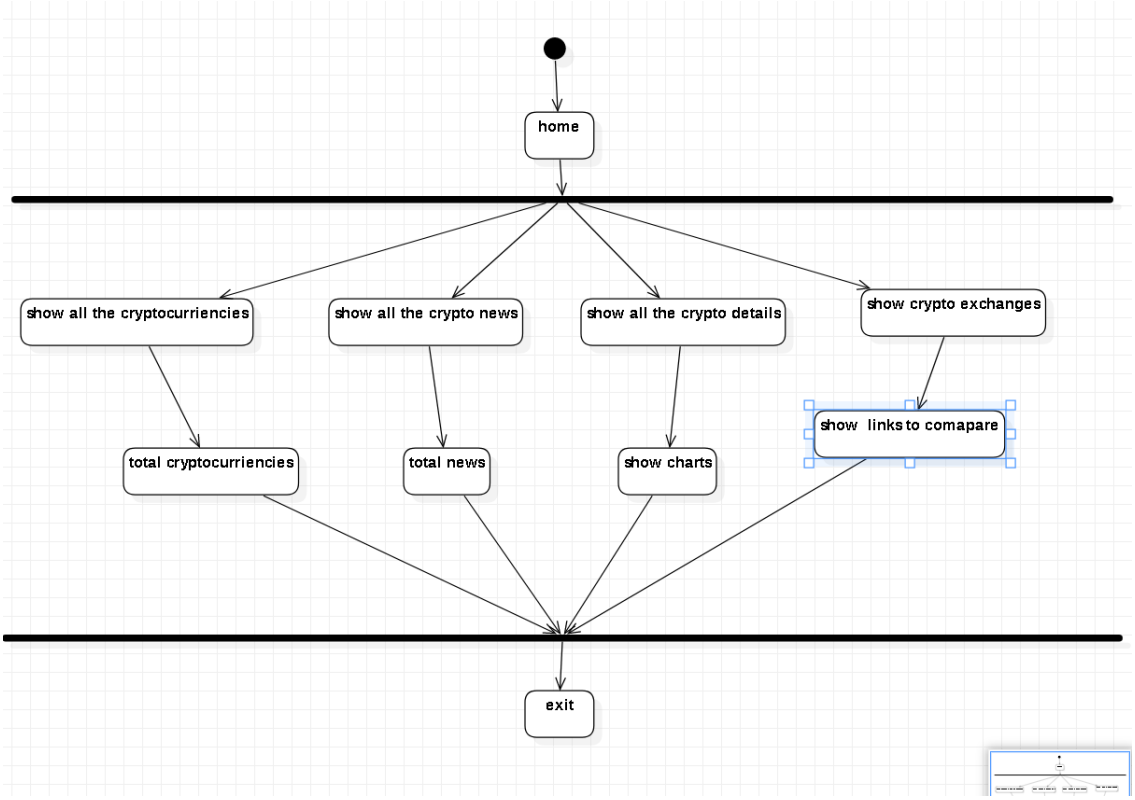

Fig 3.3.1: Activity Diagram

3.4 Class Diagram

The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them. A class consists of its objects, and also it may inherit from other classes. A class diagram is used to visualize, describe, document various different aspects of the system, and also construct executable software code.

It shows the attributes, classes, functions, and relationships to give an overview of the software system. It constitutes class names, attributes, and functions in a separate compartment that helps in software development. Since it is a collection of classes, interfaces, associations, collaborations, and constraints, it is termed as a structural diagram.

Purpose of Class Diagrams

The main purpose of class diagrams is to build a static view of an application. It is the only diagram that is widely used for construction, and it can be mapped with object-oriented languages.

It is one of the most popular UML diagrams. Following are the purpose of class diagrams given below:

i. It analyses and designs a static view of an application.
ii. It describes the major responsibilities of a system.
iii. It is a base for component and deployment diagrams.
iv. It incorporates forward and reverse engineering

How to draw a Class Diagram?

The class diagram is used most widely to construct software applications. It not only represents a static view of the system but also all the major aspects of an application. A collection of class diagrams as a whole represents a system.

Some key points that are needed to keep in mind while drawing a class diagram are given below:

1. To describe a complete aspect of the system, it is suggested to give a meaningful name to the class diagram.
2. The objects and their relationships should be acknowledged in advance.
3. The attributes and methods (responsibilities) of each class must be known.
4. A minimum number of desired properties should be specified as a greater number of the unwanted property will lead to a complex diagram.
5. Notes can be used as and when required by the developer to describe the aspects of a diagram.
6. The diagrams should be redrawn and reworked as many times to make it correct before producing its final version.

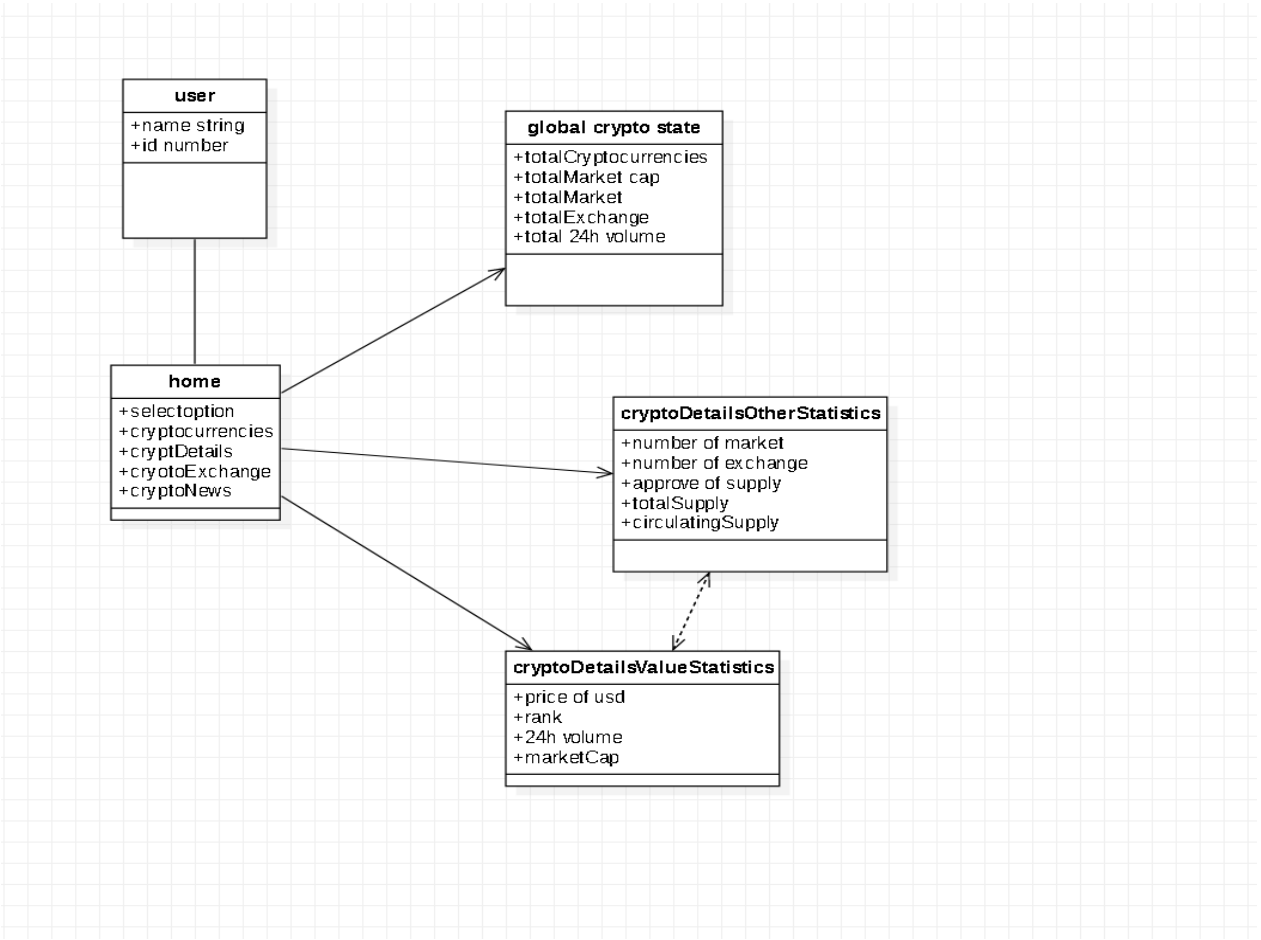

Fig 3.4.1: Class Diagram

3.5 State Chart Diagram

The name of the diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system. The states are specific to a component/object of a system.

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.

Activity diagram explained in the next chapter, is a special kind of a State Chart diagram. As State Chart diagram defines the states, it is used to model the lifetime of an object.

Purpose of State Chart Diagrams

State Chart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. State Chart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to 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.

Following are the main purposes of using State Chart diagrams −

- To model the dynamic aspect of a system.
- To model the life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model the states of an object.

How to draw a State Chart Diagram?

State Chart diagram is used to describe the states of different objects in its life cycle. Emphasis is placed on the state changes upon some internal or external events. These states of objects are important to analyse and implement them accurately.

State Chart diagrams are very important for describing the states. States can be identified as the condition of objects when a particular event occurs.

Before drawing a State Chart diagram, we should clarify the following points −

- Identify the important objects to be analysed.
- Identify the states.
- Identify the events.

Following is an example of a State Chart diagram where the state of Order object is analyzed

The first state is an idle state from where the process starts. The next states are arrived for events like send request, confirm request, and dispatch order. These events are responsible for the state changes of order object.

During the life cycle of an object (here order object) it goes through the following states and there may be some abnormal exits. This abnormal exit may occur due to some problem in the system. When the

entire life cycle is complete, it is considered as a complete transaction as shown in the following figure. The initial and final state of an object is also shown in the following figure.

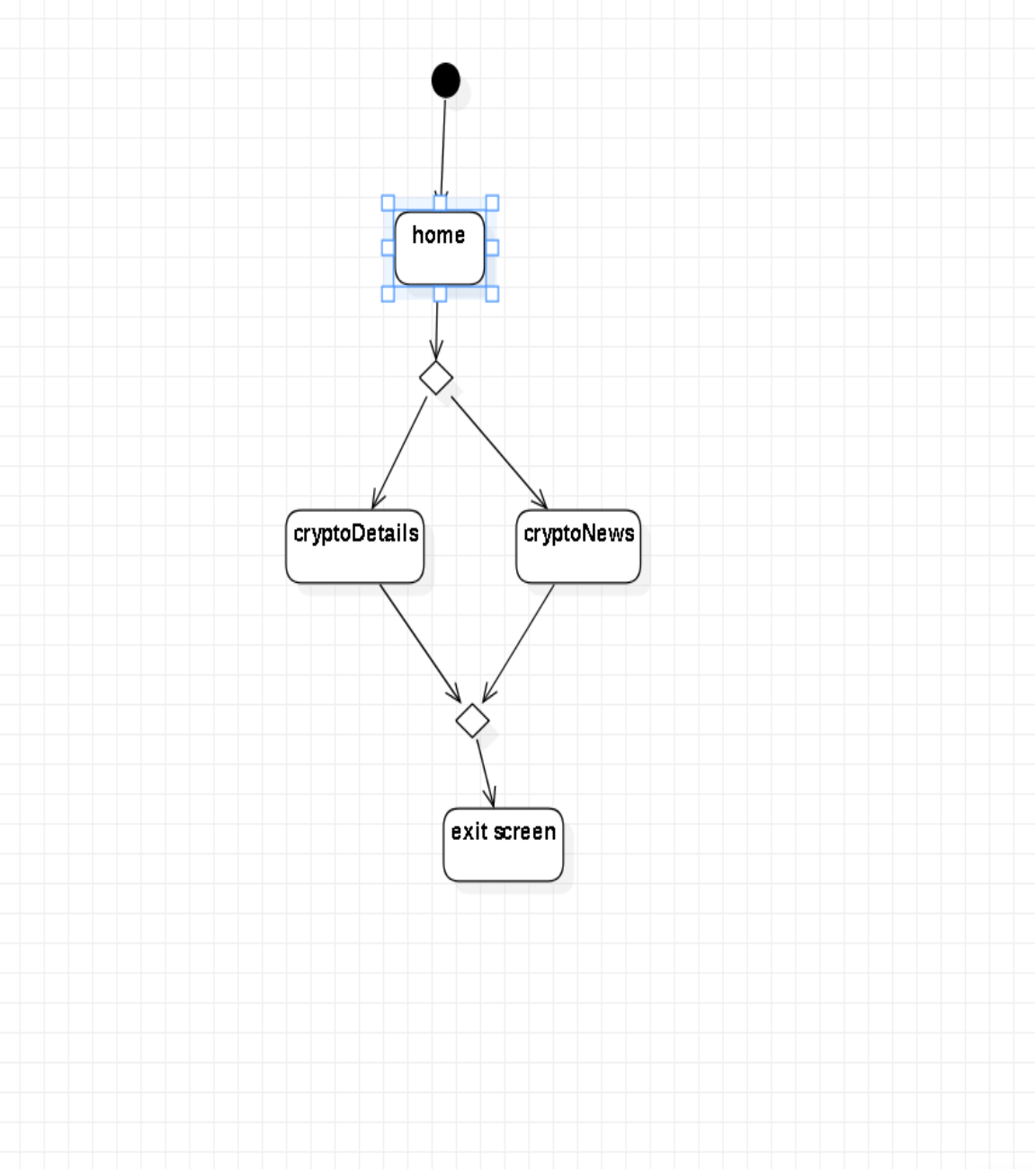

Fig 3.5.1: State Chart Diagram

3.6. Component Diagram:

Component diagrams are different in terms of nature and behavior. Component diagrams are used to model the physical aspects of a system. Now the question is, what are these physical aspects? Physical aspects are the elements such as executables, libraries, files, documents, etc. which reside in a node.

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems

Purpose of Component Diagrams

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities.

Thus, from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.

Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment.

A single component diagram cannot represent the entire system but a collection of diagrams is used to represent the whole.

The purpose of the component diagram can be summarized as −

- Visualize the components of a system.
- Construct executables by using forward and reverse engineering.
- Describe the organization and relationships of the components.

How to draw Component Diagram?

Component diagrams are used to describe the physical artifacts of a system. This artifact includes files, executables, libraries, etc

The purpose of this diagram is different. Component diagrams are used during the implementation phase of an application. However, it is prepared well in advance to visualize the implementation details.

Initially, the system is designed using different UML diagrams and then when the artifacts are ready, component diagrams are used to get an idea of the implementation.

This diagram is very important as without it the application cannot be implemented efficiently. A well-prepared component diagram is also important for other aspects such as application performance, maintenance, etc.

Before drawing a component diagram, the following artifacts are to be identified clearly −

- Files used in the system.
- Libraries and other artifacts relevant to the application.
- Relationships among the artifacts.

After identifying the artifacts, the following points need to be kept in mind.

- Use a meaningful name to identify the component for which the diagram is to be drawn.
- Prepare a mental layout before producing the using tools.
- Use notes for clarifying important points.

Following is a component diagram for order management system. Here, the artifacts are files. The diagram shows the files in the application and their relationships. In actual, the component diagram also contains dlls, libraries, folders, etc.

In the following diagram, four files are identified and their relationships are produced. Component diagram cannot be matched directly with other UML diagrams discussed so far as it is drawn for completely different purpose.

The following component diagram has been drawn considering all the points mentioned above.

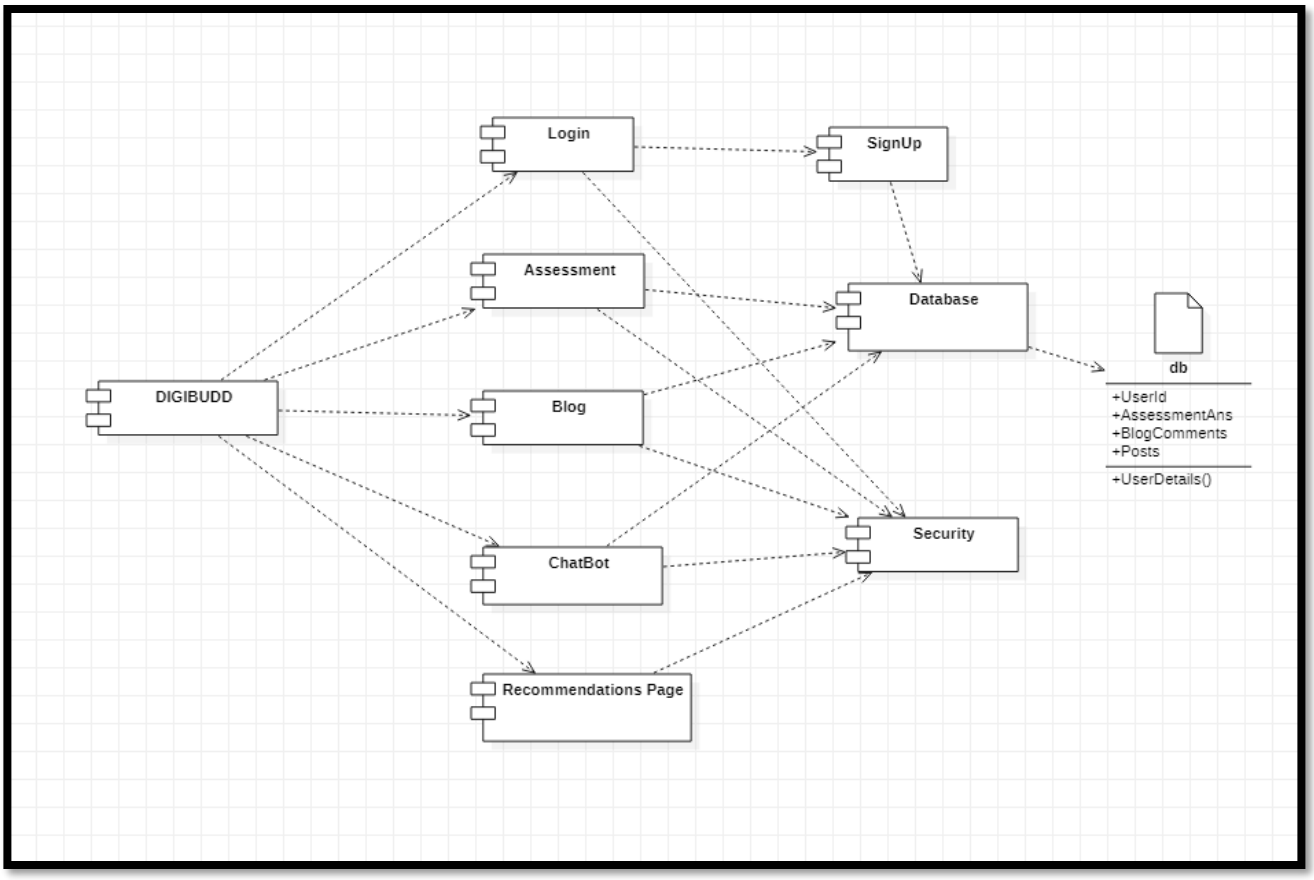

Fig 3.6.1: Component Diagram

3.7 Communication Diagram:

Component diagrams are different in terms of nature and behavior. Component diagrams are used to model the physical aspects of a system. Now the question is, what are these physical aspects? Physical aspects are the elements such as executables, libraries, files, documents, etc. which reside in a node.

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

Purpose of Communication Diagrams

- Model message passing between objects or roles that deliver the functionalities of use cases and operations
- Model mechanisms within the architectural design of the system
- Capture interactions that show the passed messages between objects and roles within the collaboration scenario

- Model alternative scenarios within use cases or operations that involve the collaboration of different objects and interactions
- Support the identification of objects (hence classes), and their attributes (parameters of message) and operations (messages) that participate in use cases

System Features

3.2 Functional Requirements

3.2.1 Login/Signup Page

Users will need to login by entering username and password. In case user forgot the password there will be an option of "Forgot password" where the user will be given an option to send a temporary password on their registered id, through which they can login and further change their password.

3.2.1.1 Description and Priority

- Provides the user with a page to login or register for the DIGIBUDD
- Priority = 10
- Stimulus: User clicks on Login Link.
- Response: Login Page is displayed
- Stimulus: User Enters Username and Password
- Response: Username and Password are validated from MySQL Database.
- Stimulus: User Clicks on Login Button
- Response: Home Page is displayed if Username and Password is correct else Error Message is displayed and the user is asked to enter username and password again.
- Stimulus: User clicks on register Link.
- Response: Register(signup) page is displayed
- Stimulus: User enters his/her details and clicks on register button
- Response: User's data gets saved; account is created and home page is displayed

3.2.1.2 Functional Requirements

- REQ-1: The user should be able to view and click on login link
- REQ-2: The user should be able to enter and validate the username and Password.
- REQ-3: There should be only one account per email else and error pop- up should appear.

3.3 Assessment Page

After visiting the landing page, users will be suggested to take up an assessment which will be prepared with the help of psychologists in order to determine the mental status of the users.

There will be multiple situation-based questions which will put the user on the decision-making place and based on the responses, mental health of the user would be determined.

4.2.1 Description and Priority

- Provides the user with a questionnaire in order to determine the mental status of the users.
- Priority=10

4.2.2 Stimulus/Response Sequences

- Stimulus: User clicks on Assessment Page. Response: Assessment Page is displayed Stimulus: User clicks on take Assessment
- Response: Assessment page and its description is displayed.
- Stimulus: User clicks on start assessment
- Response: User gets Multiple Choice Question's (MCQ) so that the digital assistant assesses the user's mental state which is then displayed.

4.2.3 Functional Requirements

- REQ-1: The user should be able to view and click Assessment Page link.
- REQ-2: The user should attempt all the MCQ's.

4.3 Recommendations Page

After visiting the assessment page, users will be suggested some recommendations or suggestions based on the responses given by user in the assessment, user will be recommended customized books, activities, yoga, exercises, videos, movies etc.

4.3.1 Description and Priority

- Recommends some photos, videos, blogs, etc. to the users based on their calculated mental health.
- Priority = 5

4.3.2 Stimulus/Response Sequences

- Stimulus: User clicks on Recommendations Page
- Response: Recommendations Page is displayed.
- Stimulus: User clicks on Recommendations
- Response: Recommendation's page and its description is displayed.
- Stimulus: User clicks on Recommendations or Suggestions
- Response: Users will be suggested some books, activities, yoga, exercises, videos, movies etc.

4.3.3 Functional Requirements

- REQ-1: The user should be able to view and click Recommendations Page link.

4.4 View User Profile

4.4.1 Description and Priority

- Provides the user with a page to view his/her profile Priority = 9

4.4.2 Stimulus/Response Sequences

- Stimulus: User clicks on Profile Menu -> My Profile
- Response: User profile is displayed with user details.

4.4.3 Functional Requirements

- REQ-1: The user should be able to click and view his/her profile.

4.5 Logout Page

4.5.1 Description and Priority

- Provides the user with a logout option. Priority = 10

4.5.2 Stimulus/Response Sequences

- Stimulus: User clicks on Logout tab/button
- Response: User is logged out and home page is displayed.

4.5.3 Functional Requirements

- REQ-1: The user should be able to logout from the system.

4.6 External Interface Requirements

4.7 User Interfaces

4.7.1 Landing Page

There will be a landing page which will be the homepage of the DIGIBUDD website where the user will get welcomed and will get an overview about DIGIBUDD.

4.7.2 Assessment Page

After visiting the landing page, users will be suggested to take up an assessment which will be prepared with the help of psychologists in order to determine the mental status of the users. There will be multiple situation-based questions which will put the user on the decision-making place and based on the responses, mental health of the user would be determined. The responses will be Multiple Choice Question's (MCQ) but there will be an option of adding any particular answer for a given question, if the options are not suitable for the user.

4.7.3 Registration Page

There will be a registration page, where users will enter their details as asked by the form and if the responses are valid they will be registered or else an error message will be displayed.

4.7.4 Login Page

Users will need to login by entering username and password. In case user forgot the password there will be an option of "Forgot password" where the user will be given an option to send a temporary password on their registered id, through which they can login and further change their password.

4.7.5 Recommendations Page

Based on the responses given by user in the assessment, user will be recommended customized books, activities, yoga, exercises, videos, movies etc.

.

4.7 Hardware Interfaces

- Laptop, Personal Computer, Mobile phones, Tablet.

4.8 Software Interfaces

Alphagraphy is a multi-user environment, it uses HTML/CSS and JavaScript for the web pages and react as the backend application tool.

4.9 Communication Interfaces

Students can communicate with the admin via email provided on the website and hence uses simple mail transfer protocol (SMTP).

4.10 Other Non-functional Requirements:

4.11.1 Performance Requirements

- PE-1: Responses to activities should not take very long time onto the screen (approx.3-6s, depending upon user's internet speed).
- PE-2: The system should display confirmation message to the user very quickly after the user submits information to the system (depending upon user's internet speed).
- PE-3: DIGIBUDD should work fine even with a number of users using concurrently using it.

4.11.2 Safety Requirements

- SR-1: Consistency: Checking the fact that all clients must be attached to one server, so there is an appropriate control of the information.
- SR-2: Users' data should be kept confidential and secured.

4.12 Software Quality Attributes

- SQA-1: Alphagraphy should be available to the users all the time.
- SQA-2: The code should be neat and it should contain all the necessary comments for the ease of maintenance and better understanding.

5 Technologies Used

5.1 FRONT-END

5.1.1 HTML:

HTML is a *markup language* that defines the structure of your content. HTML consists of a series of **elements**, which you use to enclose, or wrap, different parts of the content to make it appear a certain way, or act a certain way. The enclosing tags can make a word or

image hyperlink to somewhere else, can italicize words, can make the font bigger or smaller, and so on.

5.1.2 CSS:

Cascading Style Sheets (**CSS**) is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS describes how elements should be rendered on screen, on paper, in speech, or on other media.

5.1.3 JAVASCRIPT(JS):

JavaScript is a scripting or programming language that allows you to implement complex features on web pages — every time a web page does more than just sit there and display static information for you to look at — displaying timely content updates, interactive maps, animated 2D/3D graphics, scrolling video jukeboxes, etc. — you can bet that JavaScript is probably involved.

8. Bibliography

- IEEE Software Requirement Specification standard format