



શ્રી સ્વામિનારાયણ ગુરુકુલ રાજકોટ સંસ્થાન

શાસ્ત્રી સ્વામી શ્રી ધર્મજીવનદાસજી

સાયન્સ & IT ગુરુકુલ કોલેજ

ગુરુકુલ કેમ્પસ, કોલેજ રોડ, જૂનાગઢ

Rapid Reels

Project Partners:

MR. VINEX SURELIYA, B.Sc.I.T-6TH

MR. DIPAK SOLANKI, B.Sc.I.T-6TH

:: submitted to::

BKNM University, Junagadh

:: GUIDED BY::

Mr. Ripal Pandya

Mr. MilindAnandpara



શ્રી સ્વામિનારાયણ ગુરુકુલ રાજકોટ સંસ્થાન

શાસ્ત્રી સ્વામી શ્રી ધર્મજીવનદાસજી



સાયન્સ & IT ગુરુકુલ કોલેજ

ગુરુકુલ કેમ્પસ, કોલેજ રોડ, જૂનાગઢ



(Affiliated to **Bhakta Kavi Narsinh Mehta University, Junagadh**)

Project Completion Certificate

This certificate is awarded to

Sureliya Vinex Dineshbhai

Bsc.I.T.6-2024

Solanki Dipak Rameshbhai

Bsc.I.T.6-2024

in completion of project work

Android

19 / 12 / 2023

Firebase

11 / 03 / 2024

Mr. Ripal V. Pandya

Mr. Milind V. Anadpara

Project Guide

Director

www.sssdiit.junagadhgurukul.org



A

PROJECT REPORT ON

Rapid Reels

Submitted in Fulfillment of Requirements

For Completion of Semester - 6 in

Bachelor of Science in Information Technology

Year 2024

To

SHASHTRI SWAMI SHREE DHARMAJIVANDASJI INSTITUTE OF
INFORMATION TECHNOLOGY

JUNAGADH

Guided By:

Mr. Ripal Pandya

Mr. Milind Anadpara

Prepared By:

Mr. Vinex D. Sureliya

Mr. Dipak R. Solanki

PREFACE

Welcome to the world of "Rapid Reels," a Reel app for android is a social media application that allows users to create shorts video. Users can upload, share these videos with their followers, explore content from other users, engage with likes, comments and share the videos.

Background:-

The reel app originated from the growing popularity of short-form video content on social media platforms. With the success of other reels app developers saw an opportunity to create a dedicated platform specifically for short video creation and sharing. The goal is to offer a platform where users can express themselves creatively, connect with others, and explore new trends and ideas in the form of short, entertaining videos.

The Purpose:-

This Android software solution is to provide users with a platform sharing, uploading and discovering short-form video content. It aims to facilitate self-expression, creativity, and connection among users by offering tools for sharing, liking, and commenting on users content. Additionally, reel app often serves as a space for users to explore trending topics, challenges and content trends. Overall the purpose is to entertain, inspire and empower users to express themselves through engaging short videos.

Key Features:-

- Social interactions: Allows users to engage with content by liking, commenting, sharing and following other users.
- Trending challenges: Features trending challenges or themes that users can participated in by creating videos related to the specific challenge.

- User profiles: provides users with customizable profiles where they can showcase their videos, followers count.
- User-Friendly Interface: Intuitive design for easy navigation and usability.

Getting Started:-

This documentation will guide you through the installation, setup, and utilization of "Rapid Reels." Whether you are a software developer, a content creator, or a concerned citizen, we encourage you to explore the software and contribute to our shared mission of entertainment for all.

Acknowledgments:-

We would like to express our gratitude to all the individuals, organizations, and communities who have inspired and supported the development of "Rapid Reels." Without your support and contributions, our reel app would not have been possible. Thank you for being a part of this journey.

Thank you for choosing "Rapid Reels."

Mr. Vinex Sureliya

Mr. Dipak Solanki

ACKNOWLEDGEMENT

We are very thankful to all whose have helped in preparing this project. We are feeling a great happiness to present this website project. First of all we would like to thank “BKNM University” who give me an opportunity to give a chance to prepare a project.

Before we get in to thick of the things we would to add a few heartfelt words for the people who were part of this project numerous ways, people who give unending support right from the stage project ideas was conceived. In particular we would like to thank Mr. Ripal Pandya, Mr. Milind Anandpara (Project Guide), who has always inspired us and has directed us towards the successful completion of our project. They have been the guided through the project and their encouragement has left me indebted to them.

We are very thankful to the Director ShadhuRushikeshdashjiSwami and the Asst. Director Mr. Rajesh Bharad of Shastri Swami Shree Dharmajivandasji Institute of Information Technology – Junagadh.

We are also thankful to our classmate and few other people who helped us directly or indirectly in solving problem and in making our app development project more efficient and attractive.

Thank you...

Date:

Mr. Vinex Sureliya

Place: Junagadh

Mr. Dipak Solanki

I N D E X

NO	Particulars	Page No
1	Project Profile	1
2	Use of System Development Life Cycle Model	2
3	Feasibility Study	6
3	Requirement Gathering	10
	Requirement Analysis	12
	1) Hardware and Software Requirement	
	2) Front - End Tools	
	3) Back - End Tools	
	4) Other Tools & Technology Used	
4	Project Abstracts (User Roles & Capabilities)	13
5	Proposed System	14
6	Advantages & Limitations of Proposed System	15
7	Evaluative Report Using Pert Chart and Gantt Chart	18
8	Data Flow Diagram	23
	1) Context level	25
	2) 1st Level	26
	3) 2nd Level	27
9	Use Case Diagram	28
10	Flow Chart	29
11	Cost Estimation	32
12	Data Dictionary	36
13	Screen Layouts	38
14	Special Utilities	49
15	Testing	50
16	Implementation	54
17	Bibliography	56

PROJECT PROFILE

Project Title	Rapid Reels
Project Description	It's a reel app that is a social media app. That allows users to upload, follow each other, share, like on the content that is uploaded by the other users.
Front End	Kotlin(Android)
Back End	Firebase
Other Tools	-
Guide	Ripal V. Pandya Milind V. Anadpara
Submitted To	S.S.S.D.I.I.T College

USE OF SYSTEM DEVELOPMENT LIFE CYCLE MODEL

A software life cycle model (also termed process model) is a pictorial and diagrammatic representation of the software life cycle. A life cycle model represents all the methods required to make a software product transit through its life cycle stages. It also captures the structure in which these methods are to be undertaken.

In other words, a life cycle model maps the various activities performed on a software product from its inception to retirement. Different life cycle models may plan the necessary development activities to phases in different ways. Thus, no element which life cycle model is followed, the essential activities are contained in all life cycle models though the action may be carried out in distinct orders in different life cycle models. During any life cycle stage, more than one activity may also be carried out.

Need of SDLC

The development team must determine a suitable life cycle model for a particular plan and then observe to it.

Without using an exact life cycle model, the development of a software product would not be in a systematic and disciplined manner. When a team is developing a software product, there must be a clear understanding among team representative about when and what to do. Otherwise, it would point to chaos and project failure. This problem can be defined by using an example. Suppose a software development issue is divided into various parts and the parts are assigned to the team members. From then on, suppose the team representative is allowed the

freedom to develop the roles assigned to them in whatever way they like. It is possible that one representative might start writing the code for his part, another might choose to prepare the test documents first, and some other engineer might begin with the design phase of the roles assigned to him. This would be one of the perfect methods for project failure.

A software life cycle model describes entry and exit criteria for each phase. A phase can begin only if its stage-entry criteria have been fulfilled. So without a software life cycle model, the entry and exit criteria for a stage cannot be recognized. Without software life cycle models, it becomes tough for software project managers to monitor the progress of the project.

Following are the different Life Cycle Model example.

- Waterfall model
- Iterative waterfall model
- Prototyping model
- Evolutionary model
- Spiral model
- R.A.D. model (Rapid Application Development)

Waterfall MODEL

- **Requirement Gathering and analysis :**

All Possible requirements of the system to be developed are captured in this phase and documented in a requirement specific doc.

- **System Design :**

The requirement specification from first phase is studied in this phase and system design is prepared, system Design helps in defining overall system architecture.

- **Implementation :**

With Inputs from system design the system is first developed in small programs called nits which are integrated in the next phase, each unit is developed and tested for its functionality which is referred to as unit testing.

- **Integration and Testing :**

All the Units Develops in the Implementation phase are integrated into a system after testing of each unit post integration the entire system is tested for any faults and failures.

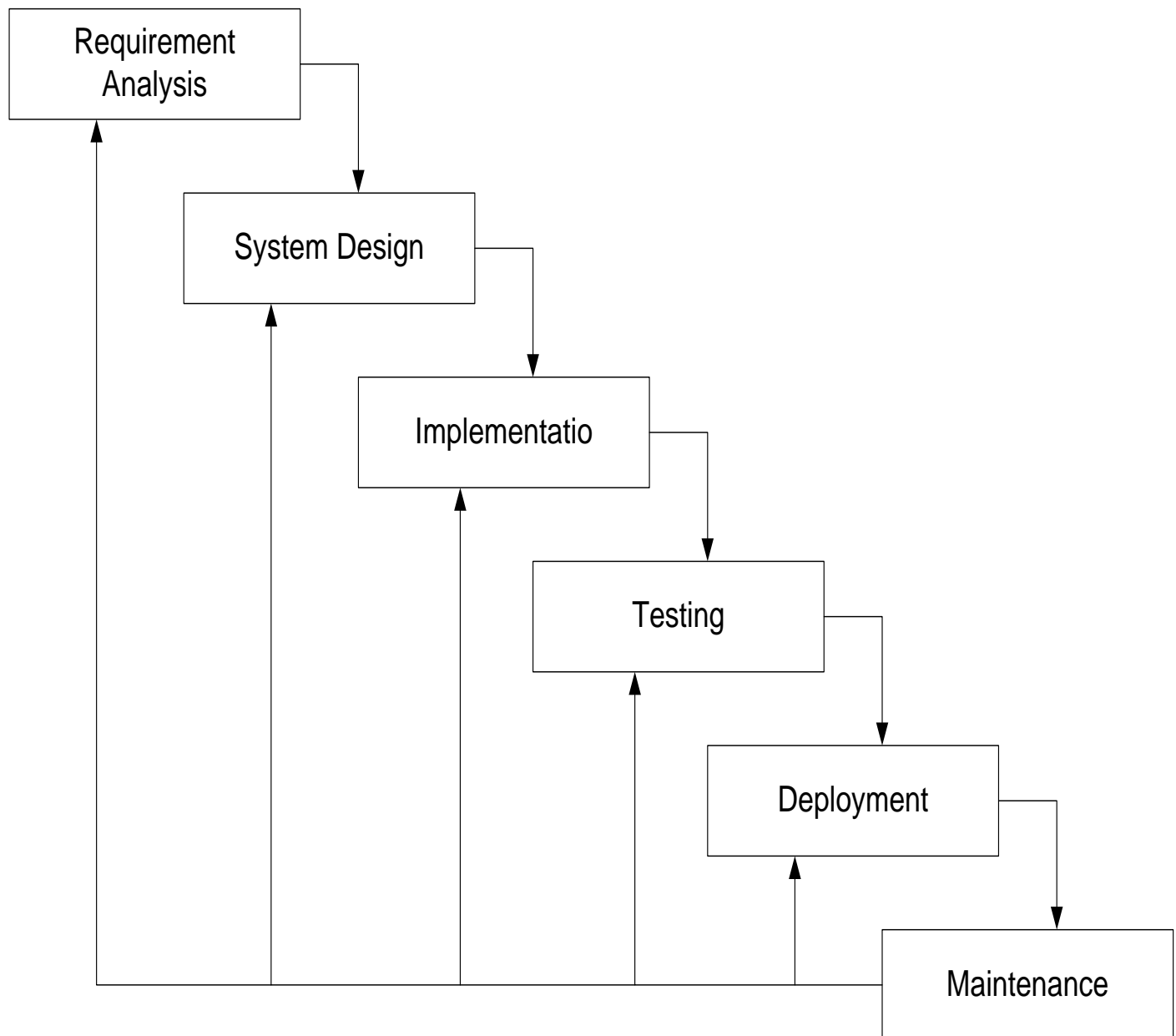
- **Deployment Of System :**

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

- **Maintenance :**

There are some issues which commune in the environment to fix those issue patches are released, also to enhance the product some better version are released.

Waterfall MODEL DIAGRAM



Feasibility Study

What is feasibility study?

A feasibility study is a critical early phase in the project planning process. It involves a systematic examination of a proposed software project to determine whether it is technically, economically, and operationally viable. The primary goal of a feasibility study is to assess whether the project is worth pursuing and whether it can be completed successfully. Feasibility studies help organizations make informed decisions about whether to proceed with a project, allocate resources, and set realistic expectations.

Types of feasibility study:-

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

1) Technical Feasibility:

Technical feasibility is an essential aspect of a feasibility study in various fields, including software engineering. It assesses whether a proposed project or initiative can be implemented from a technical standpoint. It evaluates the availability of the necessary technology, resources, and expertise to successfully complete the project. Here are some common types or aspects of technical feasibility:

➤ Hardware Requirement:-

Tools	Required
Processor	Intel i5 or higher
Hard disk	SSD 256 GB or higher
RAM	8 GB or higher
Other i/o device	Monitor/Printer

➤ **Software Requirement:-**

Software
Android Studio
Kotlin
Firebase

2) Operational Feasibility:-

Operational feasibility is a critical aspect of a feasibility study in project management and software engineering. It assesses whether a proposed project or system can be effectively integrated into an organization's existing operations and whether it can be operated and maintained efficiently once it is implemented.

- user can add videos to profile.
- Videos of different users can be seen.
- Videos can be liked, shared and commnected.
- User can see the follower count.
- User can follow each other.

3) Economic Feasibility:-

Economic feasibility, also known as cost-benefit analysis, is a crucial component of a feasibility study in project management and software engineering. It focuses on assessing whether a proposed project or system is financially viable and economically justifiable.

Our software total cost is ₹10000.

REQUIREMENT GATHERING

Requirements gathering are the process of identifying and defining what a software system needs to do to meet the needs of its users. It involves talking to stakeholders, analyzing user needs, and defining clear specifications that developers can use to build the system.

Here are the requirements of our client:-

Questionnaire:

Q.1 what type of content do you primarily want to create and share on the app?

- I'm interested in creating and sharing short videos showcasing my daily routine .

Q.2 Do you need a login page for accessing software?

- Yes

Q.3 How important is it for you to discover challenges and topics within app?

- Discovering trending challenges is crucial for me as it helps me to stay up-to-date with the latest trends and challenges.

Q.4 How important is it for you to interact with other users through likes and comments?

- Interacting with other users is essential for me as it helps me connect with fellow users and receive feedback on my content.

Q.5 How often do you anticipate using the app and uploading new content?

- I plan to use the app regularly and upload new content at least a few times a week.

Q.6 What user profile information should be included in the app?

- Usernames, profile pictures, follower count, posts and following count.

REQUIREMENT ANALYSIS

Requirement analysis is significant and essential activity after elicitation. We analyze, refine, and scrutinize the gathered requirements to make consistent and unambiguous requirements. This activity reviews all requirements and may provide a graphical view of the entire system. After the completion of the analysis, it is expected that the understandability of the project may improve significantly. Here, we may also use the interaction with the customer to clarify points of confusion and to understand which requirements are more important than others.

Types of fact finding

- Interviews
- Questionnaires
- Record reviews
- Observation

From the about option, we have used the **questionnaire** and **personal interview** methods of the fact finding techniques. I have adopted Questionnaires because I can properly understand their needed of desktop app. By using Personal Interview I have understood the smallest need of their application and some idea of layout and designing

- Admin can add, delete, user video.
- Admin can add, delete users.
- Users can add & delete videos.
- Users can like, share on video.
- User can see the post videos count, followers and follow each other.

PROJECT ABSTRACTS

User Group:

- Admin
- User

Admin:-

Admin have all rights of modify all data on whole software like, Insert-update-delete the user record, View user activity, challenges, add or delete the video of users etc...

User:-

User have rights of modify data of him on software like, post video. Post new content everyday, make challenges, liking, commenting on others post, follow each other, and share the videos etc...

PROPOSED SYSTEM

The software has been developed with Android Studio and the database is Firebase a no-SQL database.

- Android Studio (window)
- Kotlin (A programming language)
- Firebase

Using of all is acceptable factors for the real implementation of software. Because in three are much number of advantage and simplicity and also security in the point of view for this software development task. It's a mobile app that can able to manage users videos, content, with facilitate with the like, share and comment on the posted videos.

ADVANTAGES & LIMITATIONS OF PROPOSED SYSTEM

Advantages:-

➤ Conciseness:

Kotlin is more concise than java, which means developers can write the same functionality with fewer lines of code. This leads to increased productivity and easier maintenance.

➤ Null Safety:

Kotlin's type system helps eliminate null pointer exceptions by distinguishing between nullable and non-nullable types at compile time. This reduces the risk of crashes caused by null references.

➤ Interoperability:

Kotlin is fully interoperable with java, allowing developers to use kotlin and java code together seamlessly. This makes it easy to adopt kotlin gradually in existing java projects.

➤ Readability :

Kotlin's syntax is designed to be more readable and intuitive than java, making code easier to understand and maintain.

➤ Extension functions:

Kotlin allows developers to add new functions to existing classes without modifying their source code. This promotes clean and modular code architecture.

➤ **Coroutines :**

Kotlin provides first-class supports for coroutines, which are lightweight threads that simplify asynchronous programming. Coroutines make it easier to write asynchronous code that is both efficient and easy to understand.

➤ **Tooling support:**

Kotlin is well supported by popular IDEs like Android Studio, IntelliJ IDEA, and Eclipse. These tools provide features such as code completion, refactoring, and debugging support for Kotlin code.

➤ **Officially supported by Google:**

Kotlin is officially supported by Google as a first-class language for Android development. This means that Google actively contributes to its development and provides tools and resources for Android developers using Kotlin.

Limitation:-**➤ Runtime performance:**

While Kotlin aims to be fully interoperable with Java and typically has comparable performance, certain Kotlin features such as data classes and higher-order functions may incur a slight runtime performance overhead compared to equivalent Java code .

➤ Learning Curve:

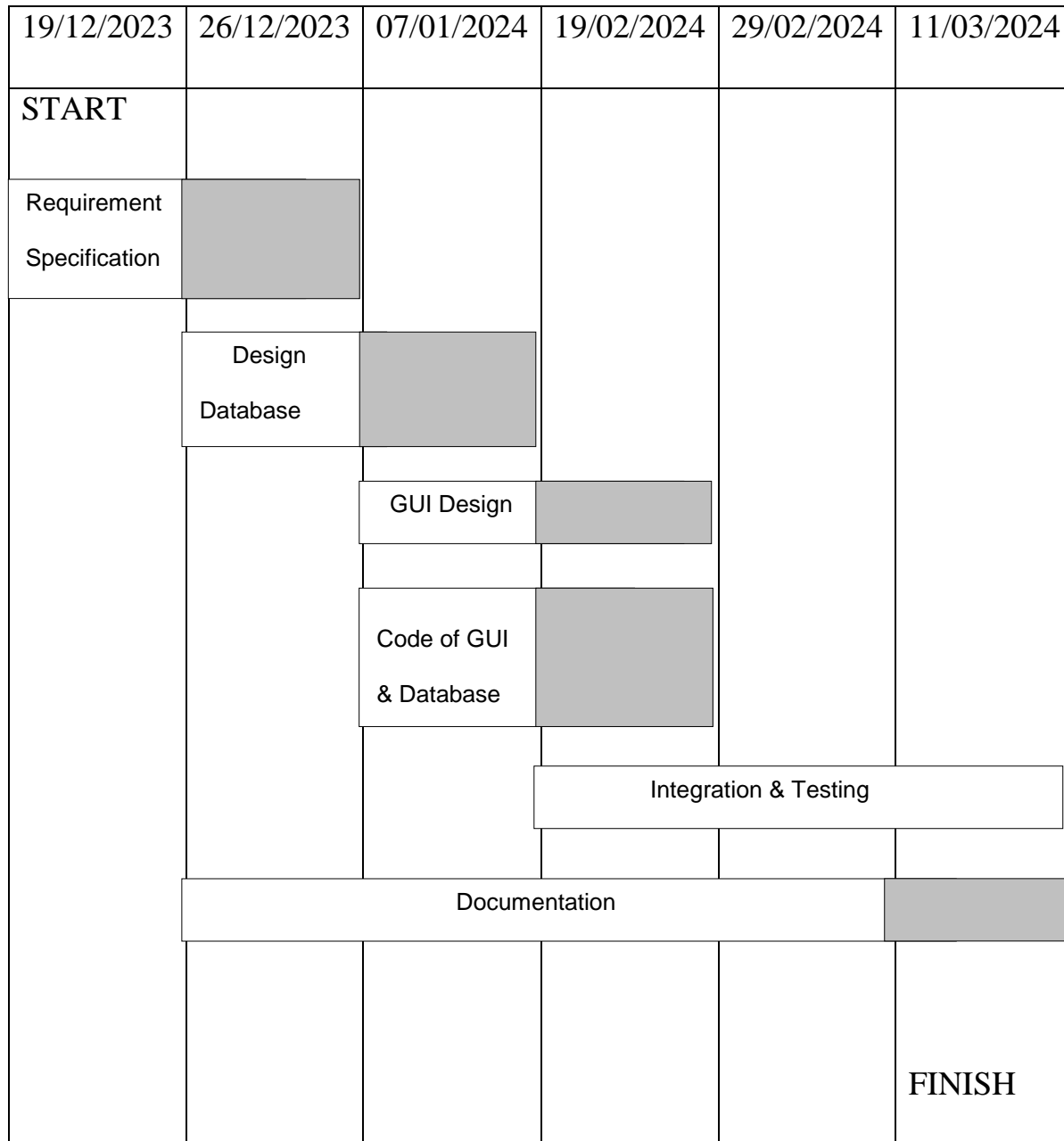
Although Kotlin's syntax is designed to be more concise and intuitive than Java, there is still a learning curve for developers who are new to the language, especially those with a background in Java.

PERT CHART AND GANTT CHART

Gantt chart:-

The first Gantt chart was devised in the mid-1890s by Karol Adamiecki, a Polish engineer who ran a steelworks in southern Poland and had become interested in management ideas and techniques.

Gantt charts are useful for planning and scheduling projects. They help you assess how long a project should take, determine the resources needed, and plan the order in which you'll complete tasks. They're also helpful for managing the dependencies between tasks.



Advantages:-

- It is a tool for representing the project schedule.
- It is easy to create.
- It is a visual tool.

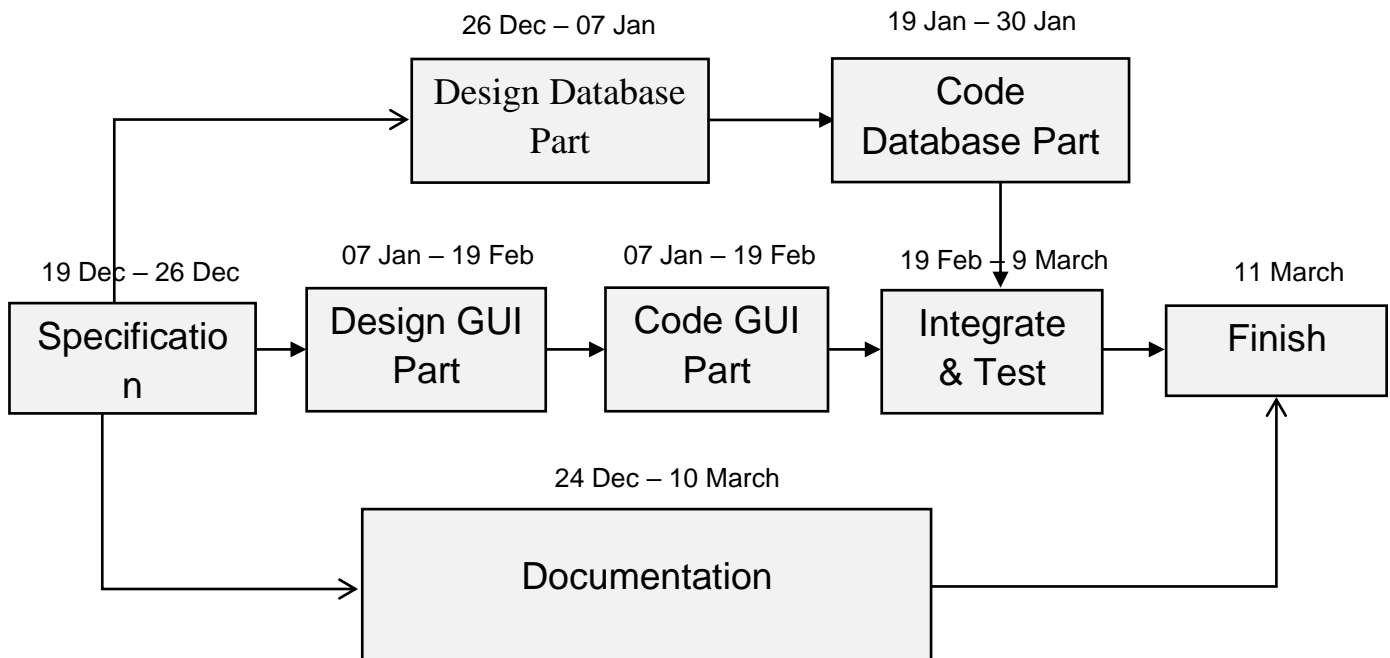
- It is easy for stakeholder's and project teams to understand the work schedule
- It enhances team productivity
- Different activity groups can be represented on the same schedule in one simple chart
- It improves communication and coordination among stakeholders in order to complete tasks as per the work schedule
- It supports procurement processes by showing delivery dates
- It helps to control the project status by comparing the planned dates shown on the chart
- It helps project and department managers to easily coordinate their teamwork
- It is a communication tool that improves stakeholder's decision-making.

Disadvantages:-

- Without software, it is hard to prepare and update the work schedule in a short period of time
- It is not easy to see all the activities in a complex project
- Inserting activities and establishing activity relationships may be time-consuming in large and complex projects
- Without software, it is not possible to assign resources to tasks
- Without software, it is not possible to make resource leveling
- It is difficult to see everything on one sheet of paper
- It is difficult to realign the activities from one WBS (Work Breakdown Structure) level to another

Pert chart:-

A program evaluation review technique (PERT) chart is a graphical representation of a project's timeline that displays all of the individual tasks necessary to complete the project.



Advantages:-

- A PERT chart allows a manager to evaluate the time and resources necessary to complete a project. It also allows the manager to track required assets during any stage of production in the course of the project.
- PERT analysis incorporates data and information supplied by a number of departments. This combining of information encourages department responsibility and identifies all responsible parties across the organization.
- Finally, PERT charts are useful input for what-if analyses. Understanding the possibilities concerning the flow of project resources and milestones allows management to achieve the most efficient and useful project path.

Disadvantages:-


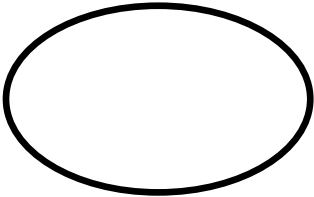
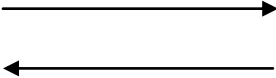
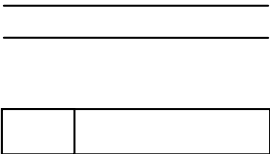
- The information that goes into a PERT chart can be highly subjective. They may include unreliable data or unreasonable estimates for cost or time.
- PERT charts are deadline-focused and might not fully communicate the financial positioning of a project.
- Creating a PERT chart is labor-intensive and maintaining and updating the information requires additional time and resources. Continual review of the information provided, as well as the prospective positioning of the project, is required for a PERT chart to be of value.

DATA FLOW DIAGRAM

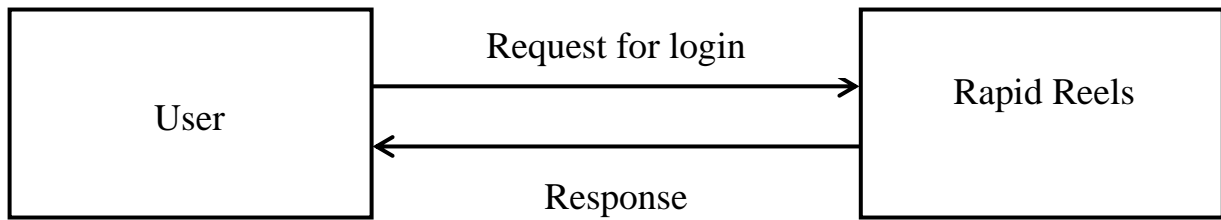
What is Data flow diagram:-

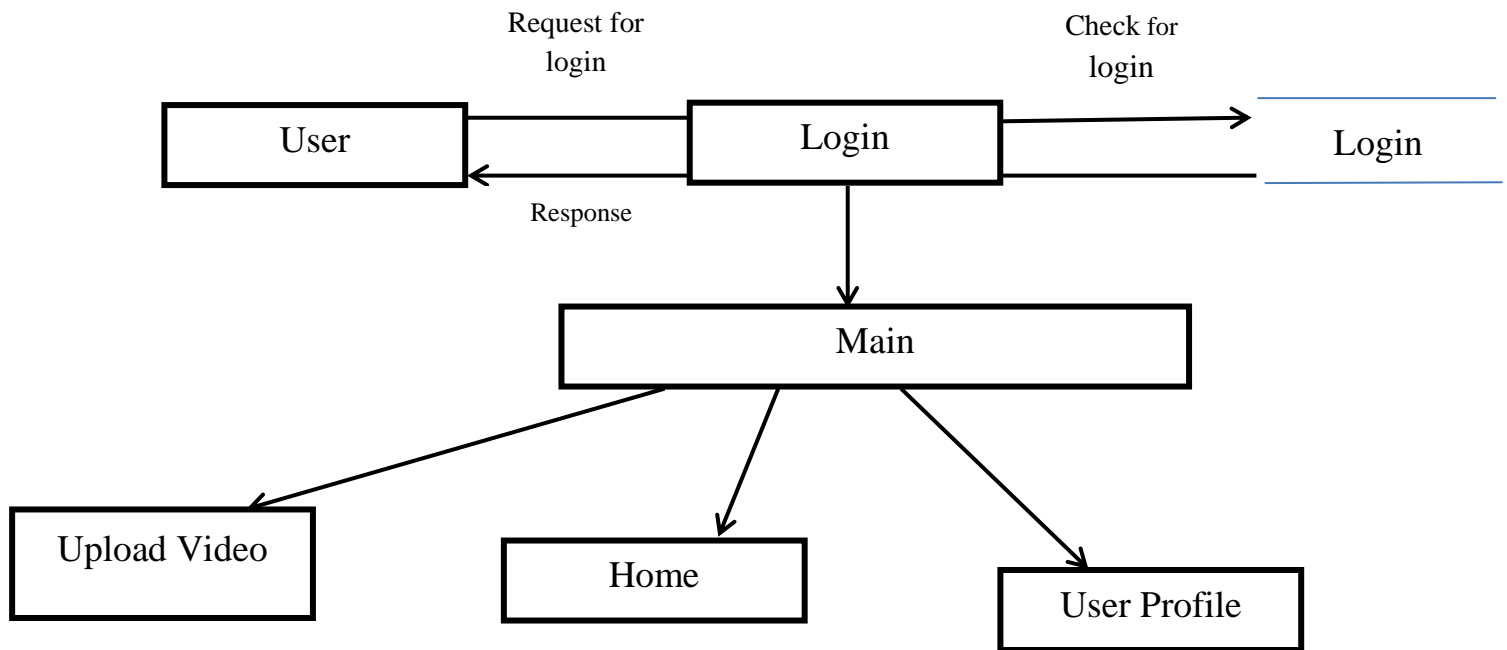
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

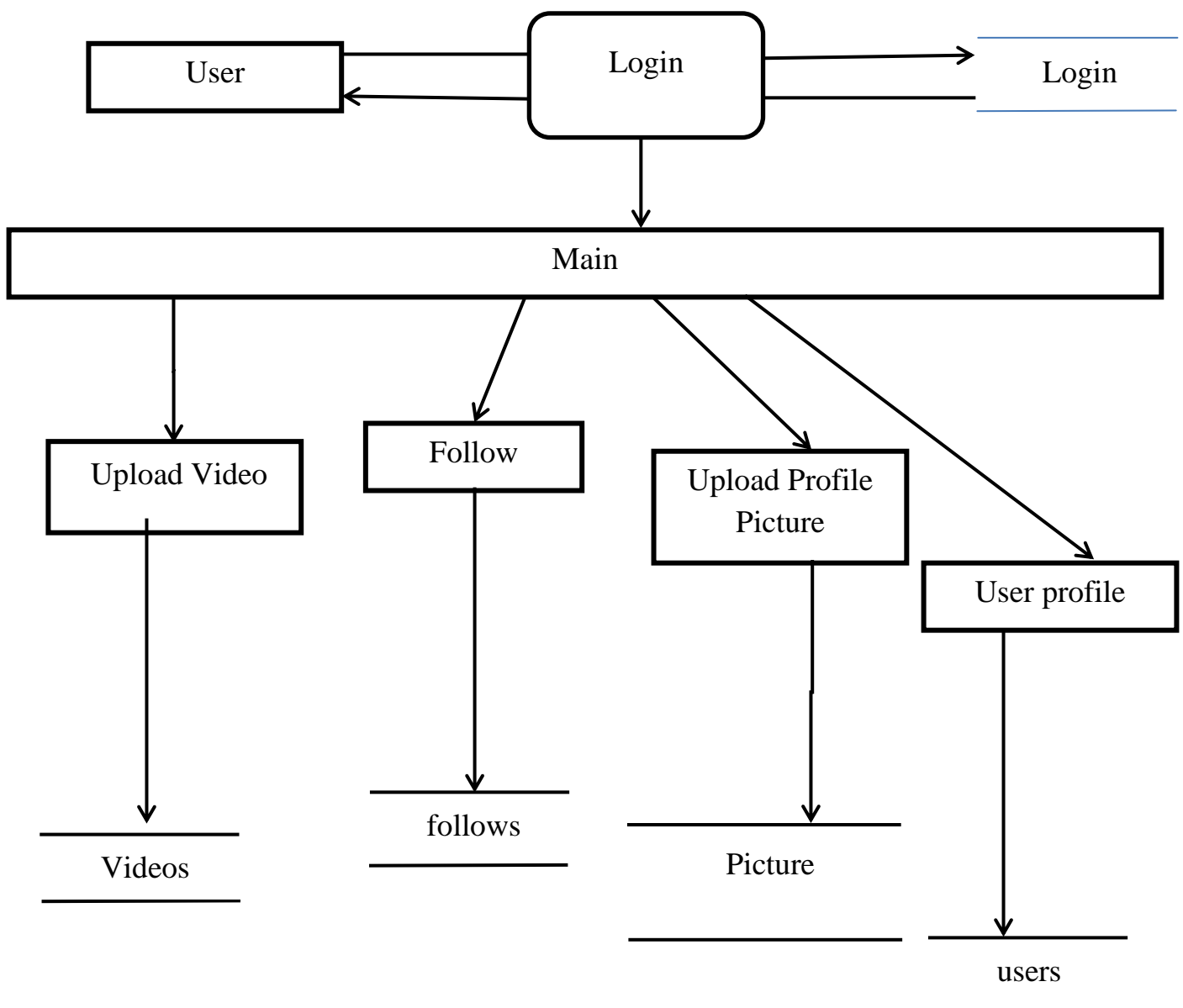
It shows how data enters and leaves the system, what changes the information, and where data is stored.

Symbol	Name	Use
	External Entity	Rectangle source and / sink destination data.
	Process / Function	Transformed, Store, or Distribute. Annotated with number and name of function.
	Data Flow	Direction of data flow single piece of data or logical collection of data.
	Data Store	Open Rectangle Parallel lines Data Structure, File, Table and Database.

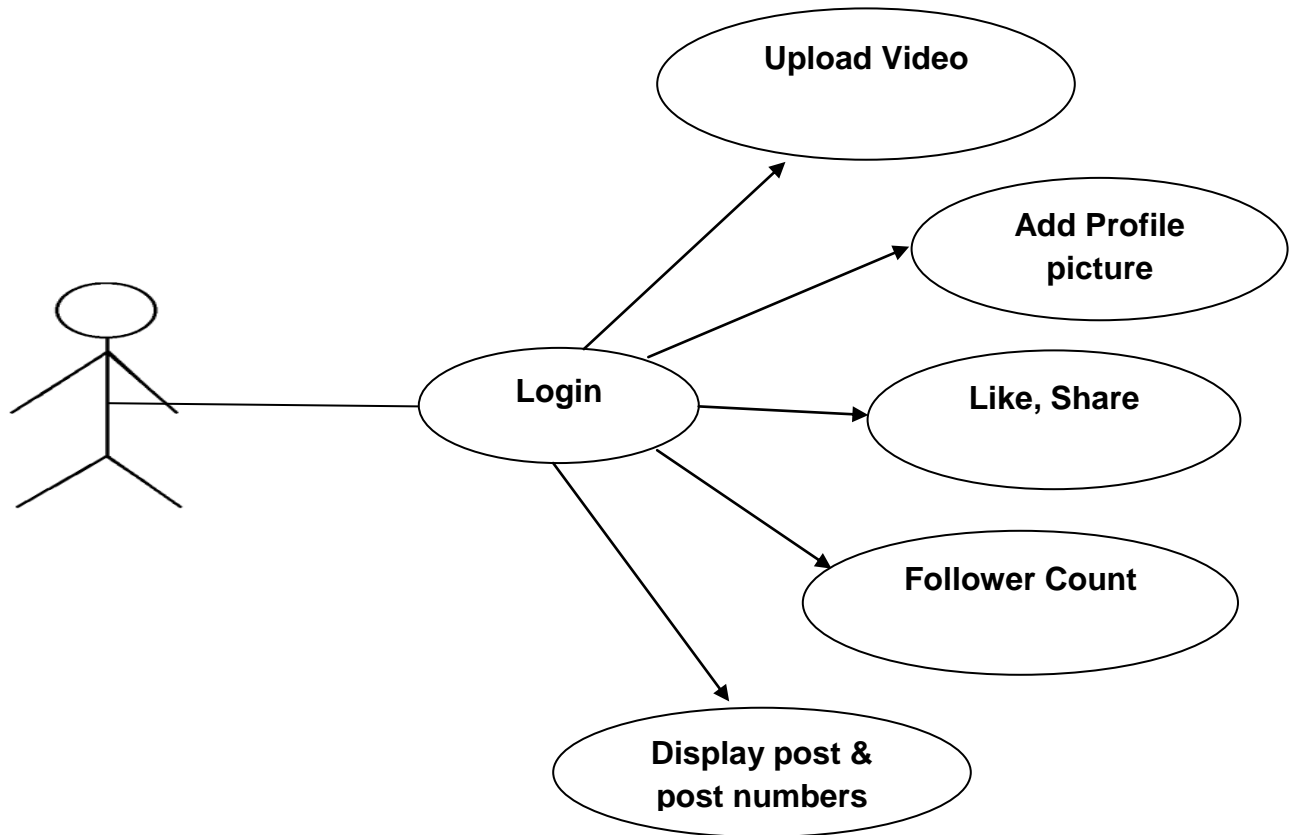
The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

Context level diagram:-

1st Level Diagram:

2nd level Diagram:

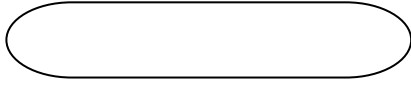
USE CASE DIAGRAM



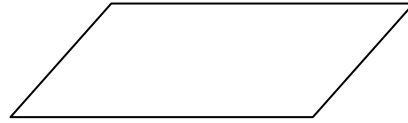
FLOW CHART

A flowchart is a diagram that depicts a process, system or computer algorithm. They are widely used in multiple fields to document, study, and plan, improve and communicate often complex processes in clear, easy-to-understand diagrams. Flowcharts, sometimes spelled as flow charts, use rectangles, ovals, diamonds and potentially numerous other shapes to define the type of step, along with connecting arrows to define flow and sequence. They can range from simple, hand-drawn charts to comprehensive computer-drawn diagrams depicting multiple steps and routes. If we consider all the various forms of flowcharts, they are one of the most common diagrams on the planet, used by both technical and non-technical people in numerous fields. Flowcharts are sometimes called by more specialized names such as Process Flowchart, Process Map, Functional Flowchart, Business Process Mapping, Business Process Modeling and Notation (BPMN), or Process Flow Diagram (PFD). They are related to other popular diagrams, such as Data Flow Diagrams (DFDs) and Unified Modeling Language (UML) Activity Diagrams

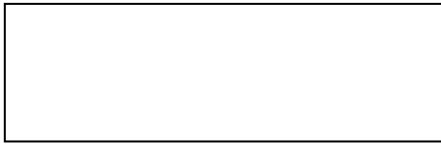
Basic Flowchart Symbols



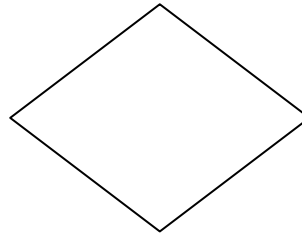
Terminal



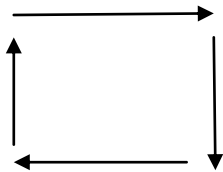
Input / Output



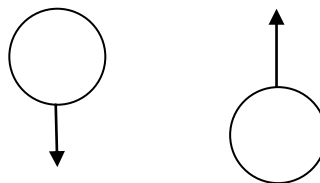
Processing



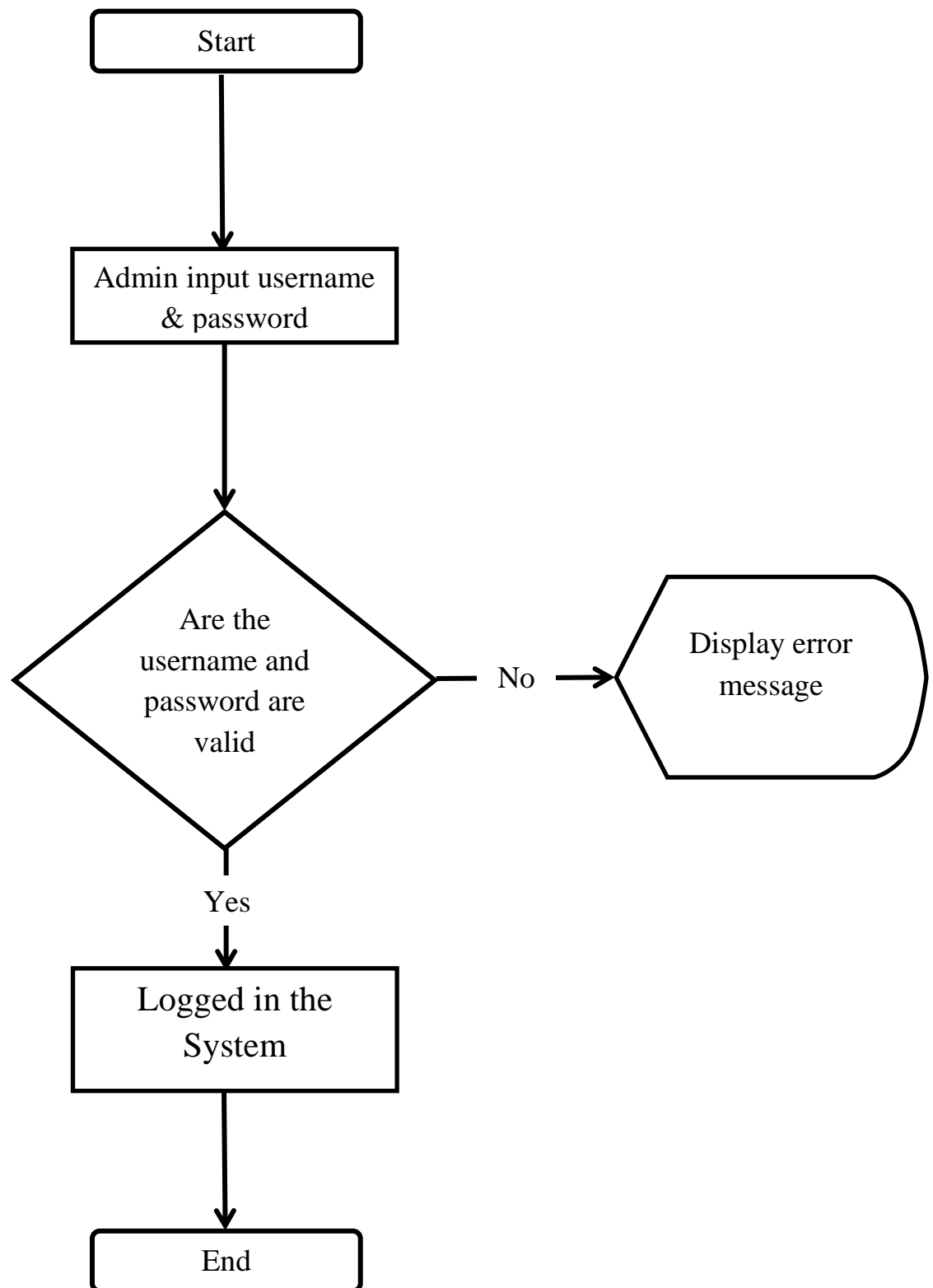
Decision



Flow Lines



Connectors

Flow chart for our program:-

COST ESTIMATION

For any new software project, it is necessary to know how much it will cost to develop and how much development time will it take. These estimates are needed before development is initiated, but how is this done? Several estimation procedures have been developed and are having the following attributes in common.

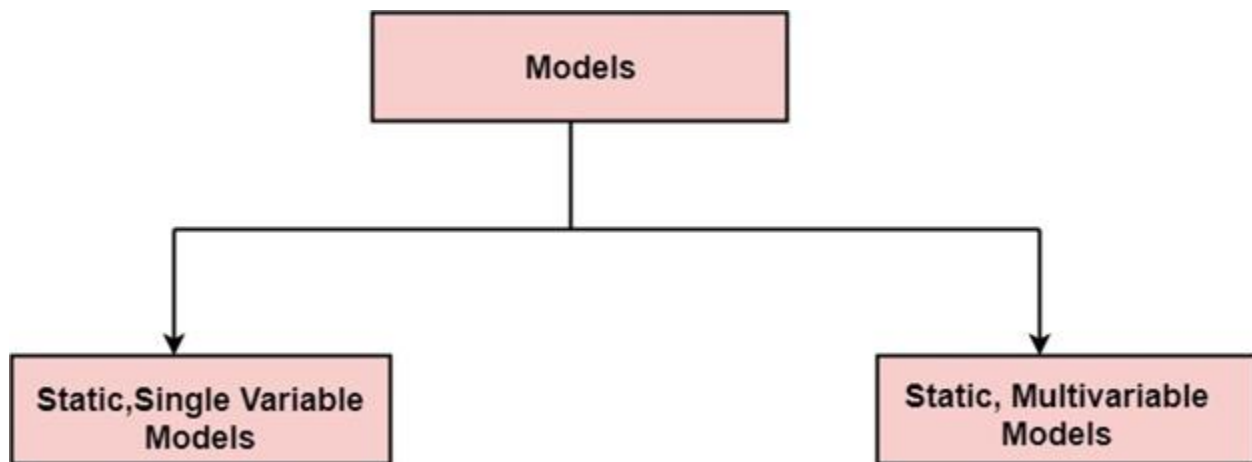
1. Project scope must be established in advanced.
2. Software metrics are used as a support from which evaluation is made.
3. The project is broken into small PCs which are estimated individually.
To achieve true cost & schedule estimate, several option arise.
4. Delay estimation
5. Used symbol decomposition techniques to generate project cost and schedule estimates.
6. Acquire one or more automated estimation tools.

Uses of Cost Estimation

1. During the planning stage, one needs to choose how many engineers are required for the project and to develop a schedule.
2. In monitoring the project's progress, one needs to access whether the project is progressing according to the procedure and takes corrective action, if necessary.

Cost Estimation Models

A model may be static or dynamic. In a static model, a single variable is taken as a key element for calculating cost and time. In a dynamic model, all variable are interdependent, and there is no basic variable.



Static, Single Variable Models: When a model makes use of single variables to calculate desired values such as cost, time, efforts, etc. is said to be a single variable model. The most common equation is:

Formula:-

$$C=aL^b$$

The Software Engineering Laboratory established a model called SEL model, for estimating its software production. This model is an example of the static, single variable model.

$$E=1.4L^{0.93}$$

$$DOC=30.4L^{0.90}$$

$$D=4.6L^{0.26}$$

Where E= Efforts (Person per month)

DOC=Documentation (Number of pages)

D = Duration (D, in months)

L = Number of Lines per code

Static, Multivariable Models:

These models are based on method (1), they depend on several variables describing various aspects of the software development environment. In some model, several variables are needed to describe the software development process, and selected equation combined these variables to give the estimate of time & cost. These models are called multivariable models.

WALSTON and FELIX develop the models at IBM provide the following equation gives a relationship between lines of source code and effort:

$$E=5.2L^{0.91}$$

In the same manner duration of development is given by

$$D=4.1L^{0.36}$$

The productivity index uses 29 variables which are found to be highly correlated productivity as follows:

$$I = \sum_{i=1}^{29} W_i X_i$$

Where W_i is the weight factor for the i^{th} variable and $X_i = \{-1, 0, +1\}$ the estimator gives X_i one of the values **-1, 0 or +1** depending on the variable decreases, has no effect or increases the productivity.

Cost estimate of our software:-

Here, we are estimating the price of the software by the module wise. So the price of the software's one module is ₹2000 because the modules are made dynamically and the market prices of the dynamic modules are ₹2000.

So we have the total 5 modules available in the project. Then in total amount of the software is ₹10,000

1 dynamic module price = ₹2000

Module 5 * 2000 = 10000/-

DATA DICTIONARY& NORMALIZATION

Database Name: Rapid Reels

Table 1: Login

Field	Type	Null	Primary Key	Extra
username	varchar(200)	No	No	
password	varchar(200)	No	No	

Table 2: User

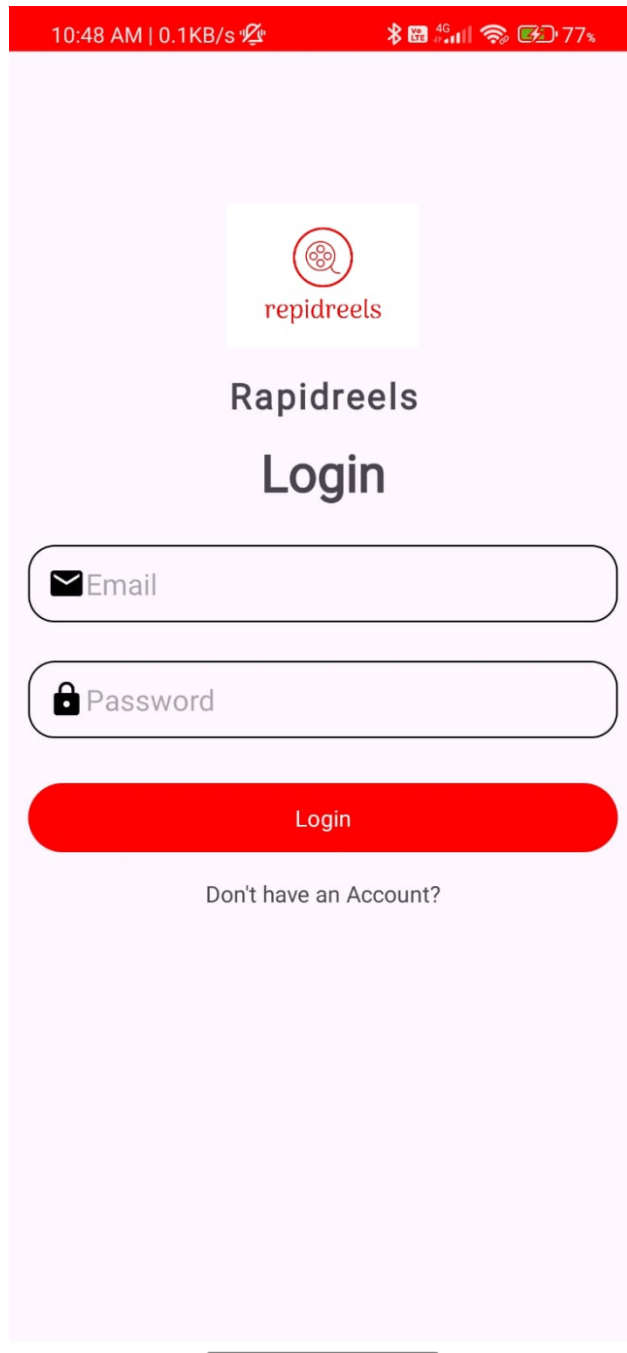
Field	Type	Null	Primary Key	Extra
Id	varchar(200)	No	No	
Follower List	Varchar(200)	No	No	
Following List	Int	No	No	
Email Id	varchar(20)	No	No	
Profile Pic	varchar(150)	No	No	
Username	varchar(100)	No	No	

Table 3: Video

Field	Type	Null	Primary Key	Extra
Title	varchar(100)	No	No	
Uploader_id	Varchar(100)	No	No	
Video Id	numeric(18,0)	No	No	
URL	numeric(18,0)	No	No	

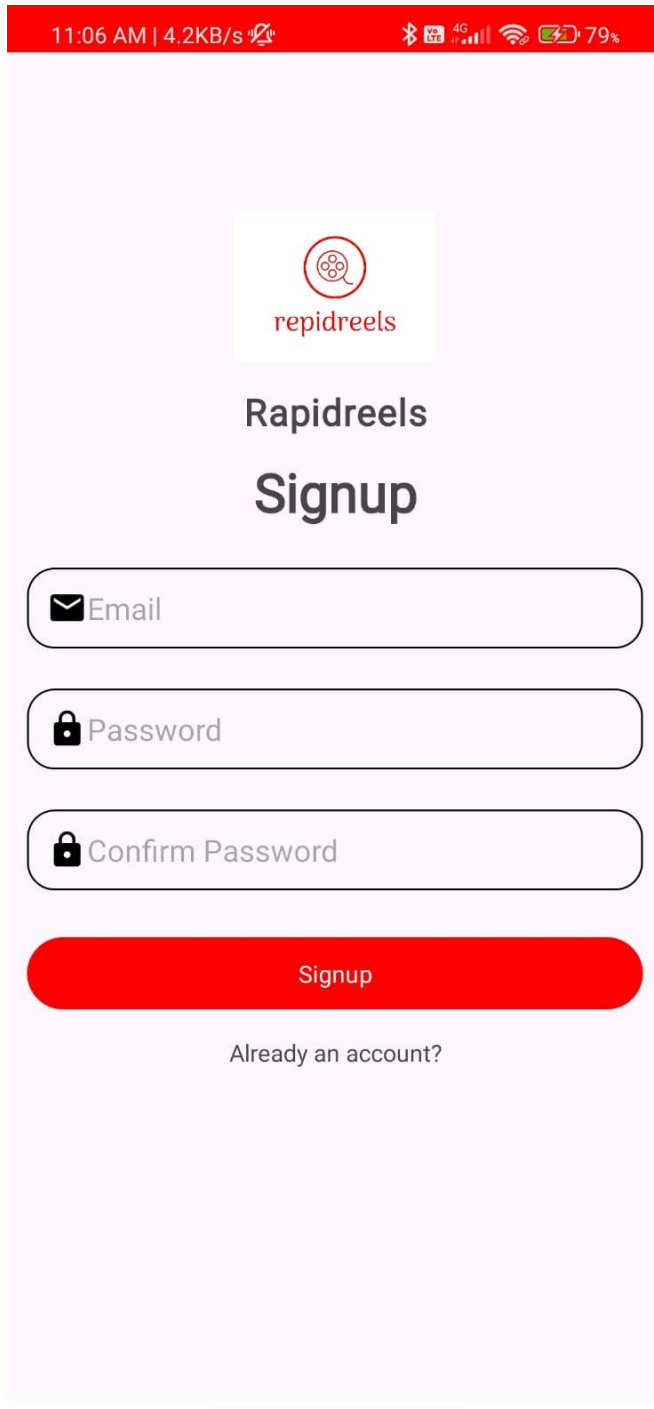
SCREEN LAYOUTS

Login Page



The image shows a mobile app login screen for 'Rapidreels'. At the top is a red status bar with the time '10:48 AM', data speed '0.1KB/s', and various connectivity icons. Below this is a light purple background. In the center, there is a white square containing a red circular logo with a film reel icon and the text 'repidreels' below it. Under the logo, the text 'Rapidreels' is displayed in a bold, dark grey font, followed by 'Login' in a larger, bold, dark grey font. Below the text are two rounded rectangular input fields. The first field has an envelope icon and the placeholder text 'Email'. The second field has a padlock icon and the placeholder text 'Password'. Below these fields is a large, rounded red button with the text 'Login' in white. At the bottom of the screen, there is a link that says 'Don't have an Account?' in a small, dark grey font. A thin black horizontal line is visible at the very bottom of the screen, likely representing the home indicator bar.

```
fun login() {  
    val email = binding.emailInput.text.toString()  
    val password = binding.passwordInput.text.toString()  
  
    if (!Patterns.EMAIL_ADDRESS.matcher(email).matches()) {  
        binding.emailInput.setError("Email not valid")  
        return;  
    }  
    if (password.length < 6) {  
        binding.passwordInput.setError("Minimum 6 character")  
        return;  
    }  
    loginWithFirebase(email, password)  
}  
  
fun loginWithFirebase(email: String, password: String) {  
    setInProgress(true)  
    FirebaseAuth.getInstance().signInWithEmailAndPassword(  
        email,  
        password  
    ).addOnSuccessListener {  
        UiUtil.showToast(this, "Login successfully...")  
        setInProgress(false)  
        startActivity(Intent(this, MainActivity::class.java))  
        finish()  
    }.addOnFailureListener {  
        UiUtil.showToast(this, "Login failed")  
        setInProgress(false)  
    }  
}
```

Signup page:-

The image shows a mobile app interface for the 'repidreels' app. At the top, there is a red status bar with the time '11:06 AM', data speed '4.2KB/s', and various connectivity icons. Below this, the app's logo (a red circle with a camera icon) and the name 'repidreels' are displayed. The main heading is 'Rapidreels Signup'. There are three input fields: 'Email' with an envelope icon, 'Password' with a lock icon, and 'Confirm Password' with a lock icon. Below these fields is a large red 'Signup' button. At the bottom, there is a link that says 'Already an account?'. The entire app interface is set against a light purple background.

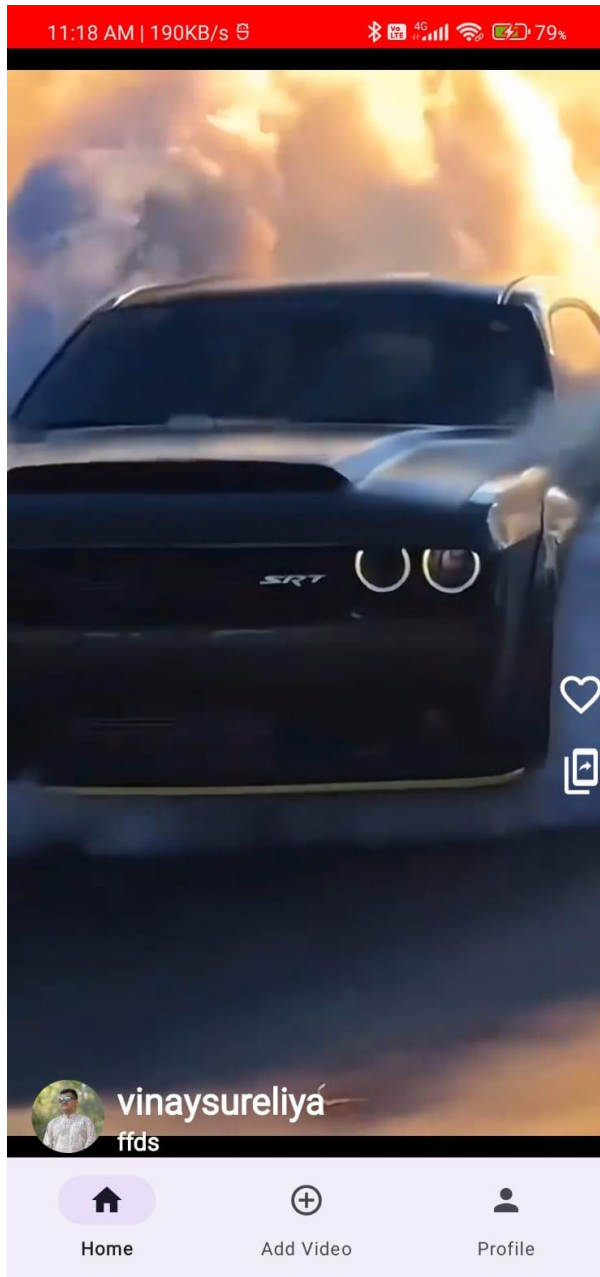
Code:-

```
fun singup() {
    val email = binding.emailInput.text.toString()
    val password = binding.passwordInput.text.toString()
    val confirmPassword = binding.confirmPasswordInput.text.toString()

    if (!Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
        binding.emailInput.setError("Email not valid")
        return;
    }
    if (password.length < 6) {
        binding.passwordInput.setError("Minimum 6 character")
        return;
    }
    if (password != confirmPassword) {
        binding.confirmPasswordInput.setError("Password not matched")
        return;
    }
    singupWithFirebase(email, password)
}

fun singupWithFirebase(email: String, password: String) {
    setInProgress(true)
    FirebaseAuth.getInstance().createUserWithEmailAndPassword(
        email, password
    ).addOnSuccessListener {
        it.user?.let { user ->
            val userModel = UserModel( user.uid, email,
            email.substringBefore("@"))
            Firebase.firestore.collection("users")
                .document(user.uid)
                .set(userModel).addOnSuccessListener {
                    UiUtil.showToast(applicationContext, "Account
created successfully...")
                    setInProgress(false)
                    startActivity(Intent(this,
MainActivity::class.java))
                    finish()
                }
            }
        }.addOnFailureListener {
            UiUtil.showToast(applicationContext, "Somthing went wrong")
            setInProgress(false)
        }
    }
}
```

Home Page:-



Code:-

```
super.onCreate(savedInstanceState)

binding = ActivityMainBinding.inflate(layoutInflater)
setContentView(binding.root)

binding.bottomNavBar.setOnItemSelectedListener {menuItem->
    when (menuItem.itemId) {
        R.id.bottom_menu_home->{
            //Goto home activity
            startActivity(Intent(this,MainActivity::class.java))
        }

        R.id.bottom_menu_add_video->{
            //Goto upload video activity
            startActivity(Intent(this,VideoUploadActivity::class.java))
        }

        R.id.bottom_menu_profile->{
            //Goto profileActivity
            val intent = Intent(this,ProfileActivity::class.java)
            intent.putExtra("profile user id",
                FirebaseAuth.getInstance().currentUser?.uid)
            startActivity(intent)
        }
        false
    }

    binding.shareBtn.setOnClickListener {
        val videoUri =
            "https://firebasestorage.googleapis.com/v0/b/rapidreels-7e588.appspot.com/o/videos%2F351140648?alt=media&token=62b62656-621c-4576-ba70-2c094b06e2e3"
        val intent = Intent(Intent.ACTION_SEND).apply {
            type = "video/mp4"
            putExtra(Intent.EXTRA_STREAM, Uri.parse(videoUri))
        }
        startActivity(Intent.createChooser(intent,"Share Via"))
    }
    setupViewPager()
}

private fun setupViewPager() {
    val options = FirestoreRecyclerOptions.Builder<VideoModel>()
        .setQuery(
            Firebase.firestore.collection("videos"),
            VideoModel::class.java
        )
}
```

```
        ).build()
        adpder = VideoListAdapter(options)
        binding.viewPager.adapter = adpder
    }

    override fun onStart() {
        super.onStart()
        adpder.startListening()
    }

    override fun onStop() {
        super.onStop()
        adpder.startListening()
    }
}
```

Add Video:-

11:20 AM | 0.1KB/s

 80%

Write your caption



CANCEL

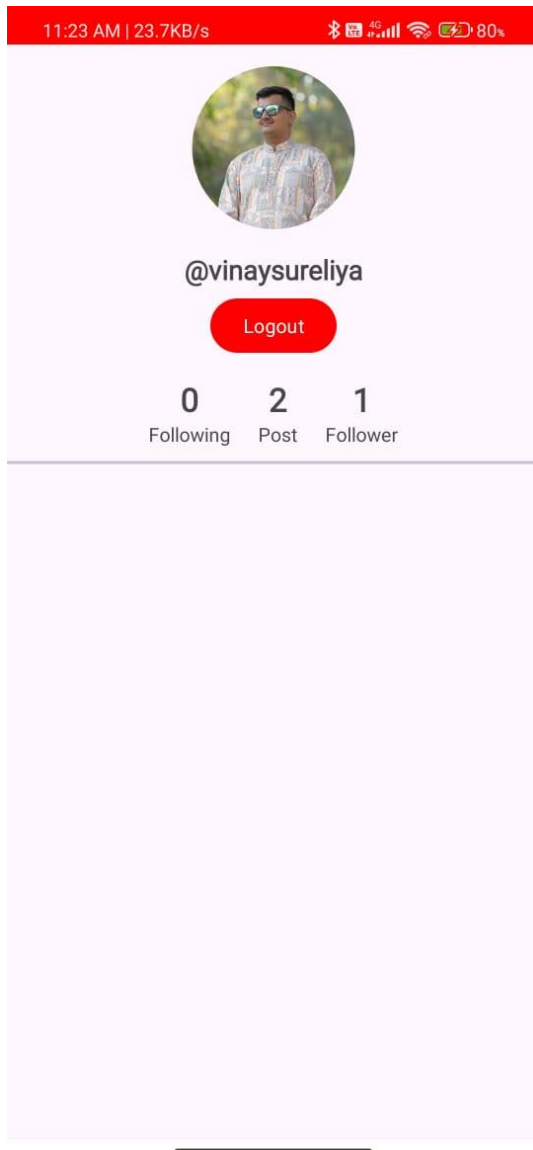
Post

Code:-

```
private fun postVideo() {
    if (binding.postCaptionInput.text.toString().isEmpty()) {
        binding.postCaptionInput.setError("Write something")
        return;
    }
    setInProgress(true)
    selectedVideoUri?.apply {
        val videoRef = FirebaseStorage.getInstance().
            reference.
            child("videos/" + this.lastPathSegment)
        videoRef.putFile(this)
            .addOnSuccessListener {
                videoRef.downloadUrl.addOnSuccessListener {
downloadUrl->
                    postToFirestore(downloadUrl.toString())
                }
            }
    }
}

private fun postToFirestore(url:String){
    val videoModel = VideoModel(
        FirebaseAuth.getInstance().currentUser?.uid!! +
        "_" + Timestamp.now().toString(),
        binding.postCaptionInput.text.toString(),
        url,
        FirebaseAuth.getInstance().currentUser?.uid!!,
        Timestamp.now()
    )
    Firebase.firestore.collection("videos")
        .document(videoModel.videoId)
        .set(videoModel)
        .addOnSuccessListener {
            setInProgress(false);
            UiUtil.showToast(applicationContext,"Video Uploaded")
            finish()
        }.addOnFailureListener {
            setInProgress(false);
            UiUtil.showToast(applicationContext,"Video failed to
upload")
        }
}
```

User Profile:-



Code:-

```
//Upload Photo
fun openPhotoPicker() {
    var intent = Intent(Intent.ACTION_PICK,
        MediaStore.Video.Media.EXTERNAL_CONTENT_URI)
    intent.type = "image/*"
    photoLauncher.launch(intent)
}

//Logout
fun logout() {
    FirebaseAuth.getInstance().signOut()
    val intent = Intent(this, LoginActivity::class.java)
    intent.flags = Intent.FLAG_ACTIVITY_NEW_TASK or
        Intent.FLAG_ACTIVITY_CLEAR_TASK
    startActivity(intent)
}

//Follow or Unfollow
fun followUnfollowUser() {
    Firebase.firestore.collection("users")
        .document(currentUserId)
        .get()
        .addOnSuccessListener {
            val currentUserModel = it.toObject(UserModel::class.java)!!

            if (profileUserModel.followerList.contains(currentUserId)) {
                //Unfollow user
                profileUserModel.followerList.remove(currentUserId)
                currentUserModel.followingList.remove(profileUserId)
                binding.profileBtn.text = "Follow"
            } else {
                //Follow user
                profileUserModel.followerList.add(currentUserId)
                currentUserModel.followingList.add(profileUserId)
                binding.profileBtn.text = "Unfollow"
            }
            updateUserData(profileUserModel)
            updateUserData(currentUserModel)
        }
}

//Post Count
Firebase.firestore.collection("videos")
    .whereEqualTo("uploaderId", profileUserId)
    .get().addOnSuccessListener {
        binding.postCount.text = it.size().toString()
    }
```


SPECIAL UTILITIES

- User can like videos.
- User can Share videos.

TESTING

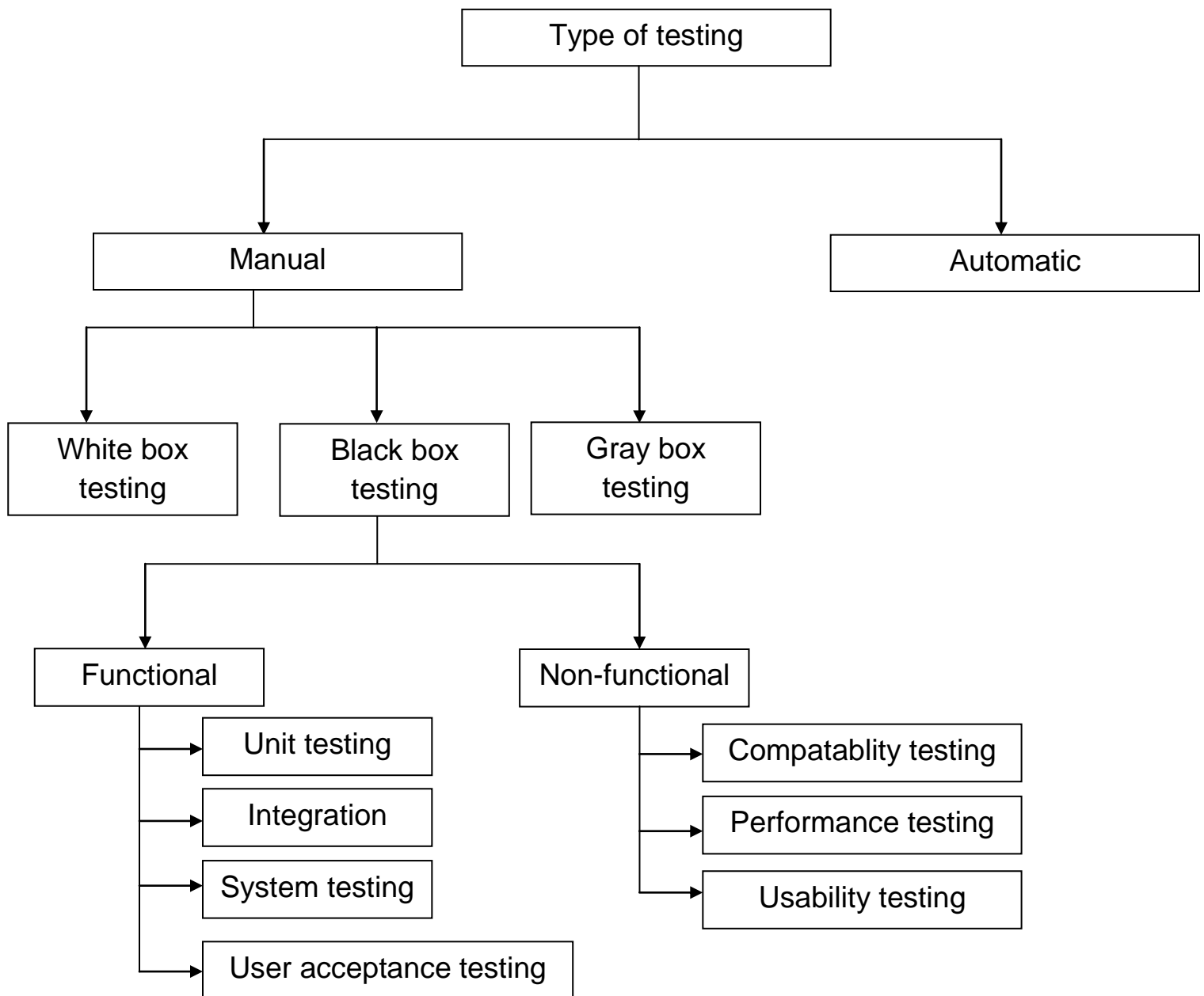
What is Software Testing

Software testing is a process of identifying the correctness of software by considering its all attributes (Reliability, Scalability, Portability, Re-usability, Usability) and evaluating the execution of software components to find the software bugs or errors or defects.

Software testing provides an independent view and objective of the software and gives surety of fitness of the software. It involves testing of all components under the required services to confirm that whether it is satisfying the specified requirements or not. The process is also providing the client with information about the quality of the software.

Types of Software testing

We have various types of testing available in the market, which are used to test the application or the software. The types of testing are below:-



1. Manual testing

The process of checking the functionality of an application as per the customer needs without taking any help of automation tools is known as manual testing. While performing the manual testing on any application, we do not need any specific knowledge of any testing tool, rather than have a proper understanding of the product so we can easily prepare the test document.

Manual testing can be further divided into three types of testing, which are as follows:

- **White box testing**
- **Black box testing**
- **Gray box testing**

➤ **White-box testing**

The white box testing is done by Developer, where they check every line of a code before giving it to the Test Engineer. Since the code is visible for the Developer during the testing, that's why it is also known as White box testing.

➤ **Black box testing**

The black box testing is done by the Test Engineer, where they can check the functionality of an application or the software according to the customer /client's needs. In this, the code is not visible while performing the testing; that's why it is known as black-box testing.

➤ Gray Box testing

Gray box testing is a combination of white box and Black box testing. It can be performed by a person who knew both coding and testing. And if the single person performs white box, as well as black-box testing for the application, is known as Gray box testing.

2. Automation testing

Automation testing is a process of converting any manual test cases into the test scripts with the help of automation tools, or any programming language is known as automation testing. With the help of automation testing, we can enhance the speed of our test execution because here, we do not require any human efforts. We need to write a test script and execute those scripts.

Test case

- If you can enter number in name then it can't allow.
- If you can enter alphabet in mobile no then it can't allow.

IMPLEMENTATION

In software engineering, implementation refers to the process of translating a software design or specification into a working and executable program or system. It is one of the crucial phases in the software development life cycle (SDLC) and typically follows the design phase. Implementation involves writing, coding, and testing the actual software components based on the design and requirements.

Here are some key aspects of implementation in software engineering:

1) Coding:

- During implementation, software developers write the source code for the software. This involves translating the high-level design and algorithms into a programming language that the computer can understand and execute.

2) Testing:

- Testing is an integral part of implementation. Developers perform unit testing to ensure that individual components or modules of the software work correctly. They also conduct integration testing to verify that these modules can work together seamlessly.

3) Debugging:

- Debugging is the process of identifying and fixing errors or bugs in the code. It's a critical step in ensuring the software functions as intended and doesn't produce unexpected or incorrect results.

4) Documentation:

- Developers create documentation during implementation to describe the code, its purpose, usage, and any dependencies. Proper

documentation is essential for future maintenance and collaboration among team members.

5) Version Control:

- Version control systems, such as Git, are often used during implementation to track changes to the codebase, collaborate with team members, and ensure that different versions of the software are well-managed.

6) Optimization:

- Developers may also optimize the code for performance, memory usage, and other factors to ensure the software meets its performance requirements.

7) Adherence to Coding Standards:

- Many development teams have coding standards or style guides that developers must follow to ensure consistency and maintainability of the codebase.

8) Security Considerations:

- Implementers should be mindful of security best practices to minimize vulnerabilities and protect the software from potential threats.

9) Reusability:

- In some cases, code modules or components may be designed for reuse in future projects or within the same project to improve efficiency and maintainability.

10) Peer Review:

- Code reviews are often conducted by other team members to assess the quality of the implementation, identify potential issues, and ensure compliance with coding standards.

BIBLIOGRAPHY

Website:-

- <https://developer.android.com>
- <https://chat.openai.com>

Books:-

- Android Programming: The Big Nerd Ranch Guide
- Kotlin for Android Developers: Learn Kotlin the Easy Way While Developing an Android App