Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with 'A' Grade (CGPA 3.18)

Continuous Assessment for Laboratory / Assignment sessions

Academic Year 2023-24

Name: Amartya Mishra SAP ID: 60004210210

Course: Software Engineering Course Code: DJ19CEC601

Year: T.Y. B.Tech. Sem: VI Batch: C31

Department: Computer Engineering

Performance Indicators (Any no. of Indicators) (Maximum 5 marks per indicator)	1	2	3	4	5	6	7	8	9	10	Σ	Avg	A1	A2	Σ	Avg
Course Outcome	1	2	2	2	3	2	5	4	4	6						
1. Knowledge (Factual/Conceptual/Procedural/ Metacognitive)	2	2	3	3	3	3	2	2	2	3			2	2		2
2. Describe (Factual/Conceptual/Procedural/ Metacognitive)	2	-	3	3	3	2	3	3	3	3			2	2		2
3. Demonstration (Factual/Conceptual/Procedural/ Metacognitive)	-	-	3	-	3	-	-	3	-	3			2	2		2
4. Strategy (Analyse & / or Evaluate) (Factual/Conceptual/ Procedural/Metacognitive)	2	3	3	3	2	2	3	3	3	-			2	2		2
5. Interpret/ Develop (Factual/Conceptual/ Procedural/Metacognitive)	-	2	3	2	-	3	3	3	-	-			-	-		-
6. Attitude towards learning (receiving, attending, responding, valuing, organizing, characterization by value)	3	3	2	3	2	3	-	-	3	3			2	2		2
7. Non-verbal communication skills/ Behaviour or Behavioural skills (motor skills, hand-eye coordination, gross body movements, finely coordinated body movements speech behaviours)	3	3	-	-	-	-	3	-	3	3			-	-		-
Total	12	13	14	14	13	13	14	14	14	14			10	10		10
Signature of the faculty member																

Outstanding (5), Excellent (4), Good (3), Fair (2), Needs Improvement (1)

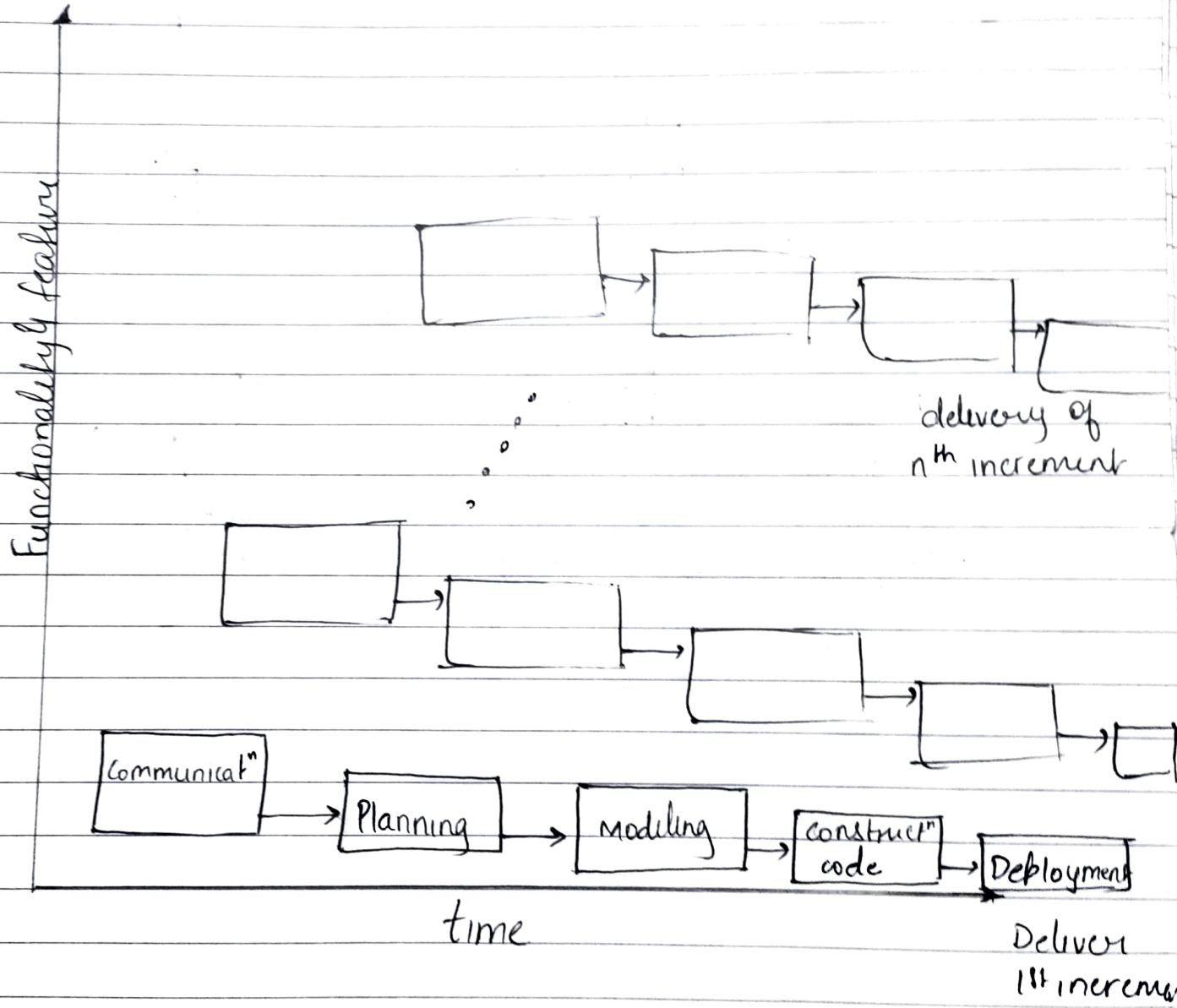
Laboratory marks Σ Avg. = 13.5	Assignment marks Σ Avg. = 10	Total Term-work (25) = 23.5
Laboratory Scaled to (15) =	Assignment Scaled to (10) =	Sign of the Student:

Signature of the Faculty member: 30/4/24
Name of the Faculty member:

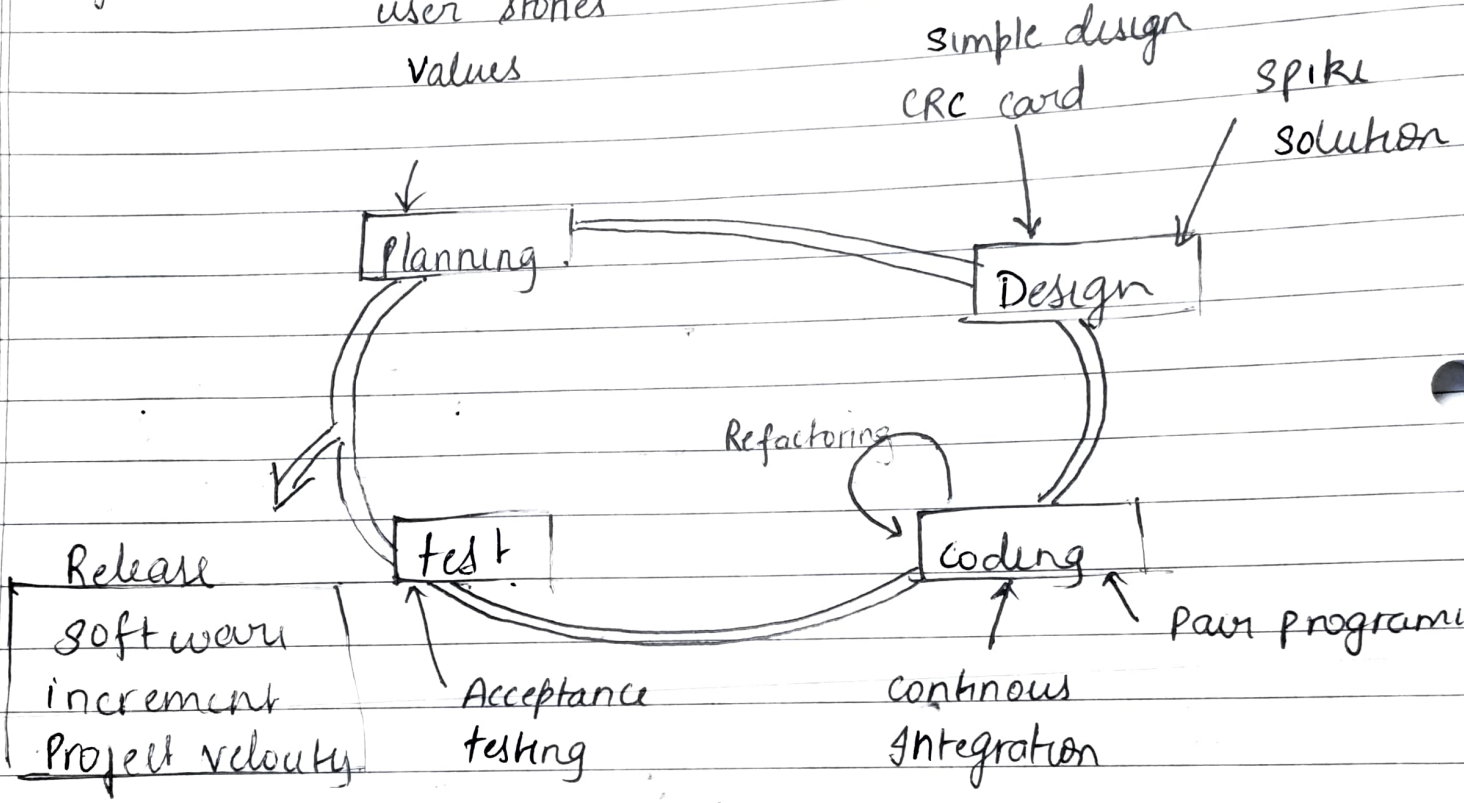
Signature of Head of the Department
Date:

60004210210
Amarkya Mishra
COMPS - C31
SE Lab

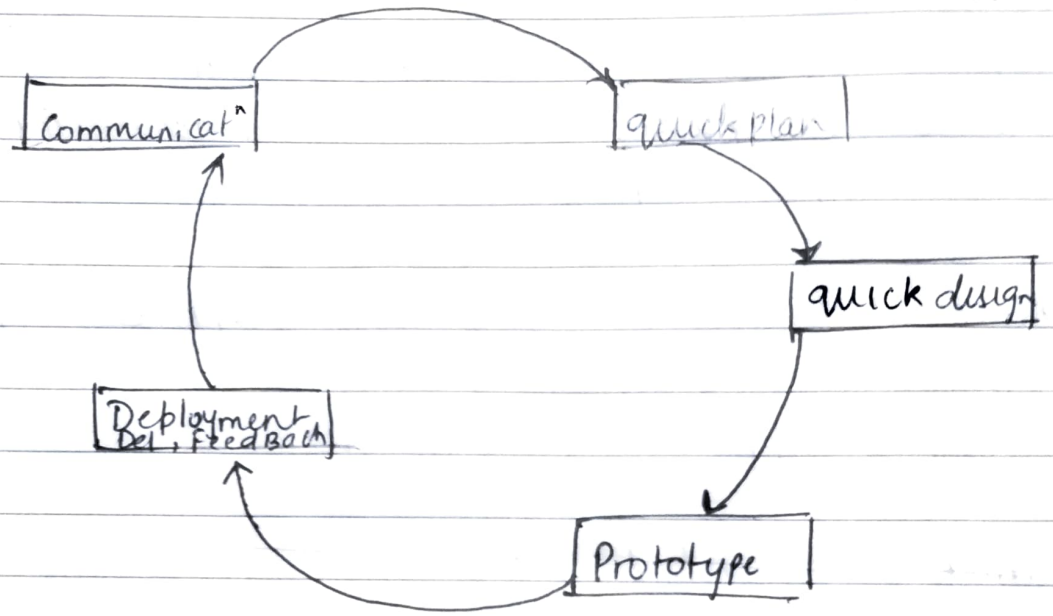
→ The Incremental Model



Agile Model → Extreme Programming



→ Prototyping



Justification :

In context of our application chosen life cycle model is the Incremental Model. It is well suited for projects where initially requirements are unclear It allows flexibility & adjustments throughout the development process

Incremental model offers an adaptive approach to accomodate changes & Enhancements as the circlify application progresses

Initial Increments : in our case study involves the development of core features in circlify such as seamless media sharing, user connections & basic collaborative functions

60004210210
Amartya Mishra
COMPS - C31

SE EXPERIMENT - 3

Aim: Identify Scenario & develop UML use case & class Diagram.

Theory:

Abstraction & Simplification:
class Diagrams abstract complex systems into managable components focusing on class & relationship.

Communication:
They provide a common language for Stakeholders to understand & discuss the system Structure & behavior.

Blueprint & implementation:
class diag. guide developers in writing code, defining class hierarchies & Establishing Relationships.

Analysis & design:
They are in early stage & analysis & design by identifying key classes & relationship facilitating iterative refining.

Modularity & reusability:
Organize classes that - coherent units promotes Modularity & reusability facilitating maintenance

· Documentation:

class diagram serves as a valuable document offering a visual representation of system's architecture for reference throughout the development life cycle.

Experiment 3

Darshit Sarda 60004210208

Amartya Mishra 60004210210

Gaurang Bhogle 60004210192

Shubham Mehta 60004210191

Aim: Identify scenarios & develop UML Use case and Class Diagram for the project

Theory:

A class diagram in software engineering serves as a foundational tool for visualizing the structure and relationships within a system. It is a graphical representation of the classes, attributes, operations, and relationships among objects that make up the software system. Here's a breakdown of its importance and roles:

1. Abstraction and Simplification: Class diagrams help in abstracting complex systems into manageable components. By representing classes and their relationships visually, developers can focus on high-level design decisions without getting bogged down in implementation details.

2. Communication: Class diagrams serve as a common language between stakeholders, including developers, designers, and clients. They provide a clear and concise way to communicate the system's structure and behavior, facilitating collaboration and understanding among team members.

3. Blueprint for Implementation: Class diagrams provide a blueprint for the implementation phase of software development. Developers use them as a guide to write code, define class hierarchies, and establish relationships between objects.

4. Analysis and Design: During the early stages of software development, class diagrams aid in analysis and design. They help identify the key classes and their relationships, allowing developers to model the system's structure before writing any code. This iterative process enables refining the design based on feedback and changing requirements.

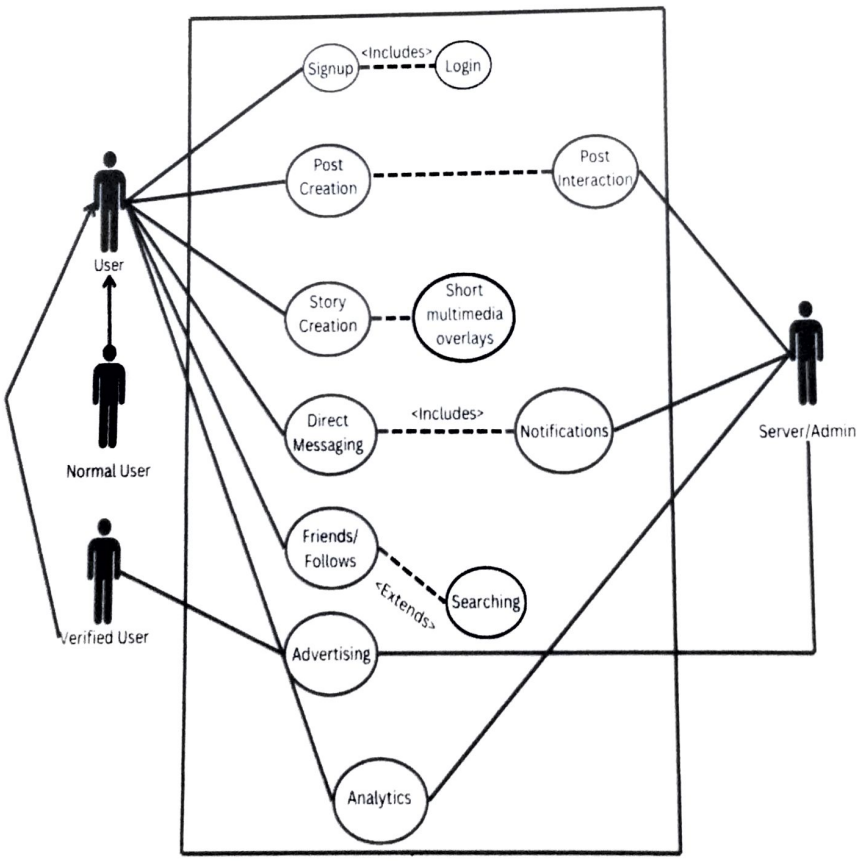
5. Modularity and Reusability: Class diagrams promote modularity and reusability by organizing classes into coherent units. Well-designed class hierarchies and relationships allow developers to encapsulate functionality within classes, making it easier to maintain and extend the system over time. Additionally, reusable classes can be leveraged across multiple projects, reducing development time and effort.

6. Documentation: Class diagrams serve as valuable documentation for the software system. They provide a visual representation of the system's architecture, which can be referenced by developers, testers, and other stakeholders throughout the software development

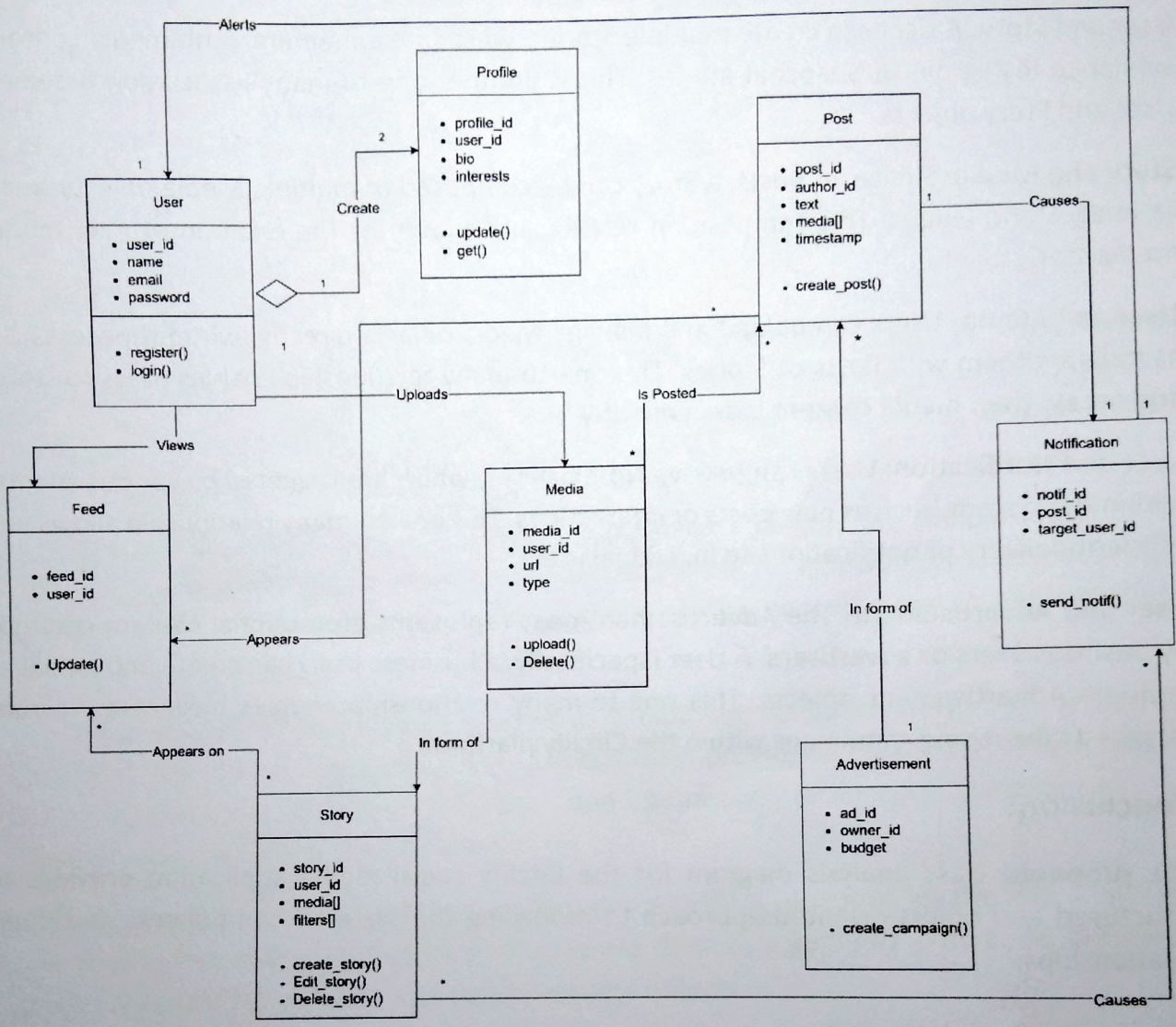
lifecycle. Additionally, class diagrams can be supplemented with additional documentation, such as class descriptions and usage guidelines, to further enhance understanding.

In essence, class diagrams play a crucial role in software engineering by facilitating communication, guiding implementation, promoting modularity and reusability, and serving as documentation for the system. They are indispensable tools for designing and understanding complex software systems, helping teams build robust and maintainable software products.

Use case Diagram:



Class Diagram:



User and Profile: The User class represents the core entity of the system, encompassing user accounts and authentication. Each User has an associated Profile, which stores additional information about the user, such as their bio and interests. This relationship is a composition, where a Profile object cannot exist without a User object.

User and Post: A User can create multiple Posts, which can contain text, media, and other content. This relationship is a one-to-many association, where one User can be associated with multiple Post objects.

Post and Media: A Post can be composed of one or more Media objects, which represent images, videos, or other multimedia content. This is a composition relationship, where Media objects are owned by and exist within the context of a Post.

Feed and Post: The Feed class is an aggregation of multiple Post objects. A Feed represents the collection of posts that a User will see in their personalized content feed. This relationship allows for efficient retrieval and organization of posts for each user.

User and Story: A User can create multiple Stories, which are ephemeral multimedia content similar to Instagram or Snapchat stories. This is another one-to-many association between User and Story objects.

Story and Media: Similar to Posts, a Story can be composed of multiple Media objects, such as images and videos. This composition relationship allows for the creation of rich, multi-media stories.

User and Media: Users can upload and manage Media objects directly, without necessarily associating them with Posts or Stories. This one-to-many relationship enables users to store and access their media content independently.

User and Notification: Users can receive Notifications, which are triggered by various events within the system, such as new posts or interactions. This one-to-many relationship allows for efficient delivery of notifications to individual users.

User and Advertisement: The Advertisement class represents promotional content created by business users or advertisers. A User (specifically, a business user) can create and manage multiple Advertisement objects. This one-to-many relationship enables businesses to run targeted advertising campaigns within the Circlify platform.

Conclusion:

The proposed class analysis diagram for the Circlify social media application provides a structured and object-oriented approach to modelling the system's components and their relationships.

60004210210
Amartya Mishra
COMPS - C31

SE EXPERIMENT - 4

Aim: To develop activity diagram & DFD (upto 2 levels) for our case study

Theory:

Steps in activity Diagram are as follows:

→ Start → the activity begins

→ Login or Signup : The user is prompted to either login with their existing account or signup if they are new to the app.

→ If New user : The user does not have pre-existing credentials, then they are directed to signup process They provide necessary information.

→ Validated information: The user input is validated by the system to check if given details are acceptable or not

→ Send verification mail : It is a part of security enhancing method as this step ensures only valid users get access to system functionalities.

→ Verify Identity : The user checks Emails & clicks on verification link to confirm their Identity.

→ Login : The user returns to app & continues with login by inputting credentials.

→ Login successfull : The user is provided with auth token & sucessfully can access the app.

→ Show loading page: The user is directed to landing page of circlify app where they can view their feed, post content, interact with other users post & etc.

⇒ Swimlane diagram:

- User Swimlane: It represents the actions & interactions performed by user such as login attempts etc.

- System Swimlane: It represents actions performed by circlify app. such as verifying credentials, storing information etc.

- Admin Swimlane: It represents actions performed by admin on the system. This includes monitoring the data on platform, deleting & eliminating fraudulent accounts with misleading content.

Conclusion: Thus we created & represented the various Elements of circlify using Activity, Swimlane & DFD diagrams.

Experiment 4

Darshit Sarda 60004210208

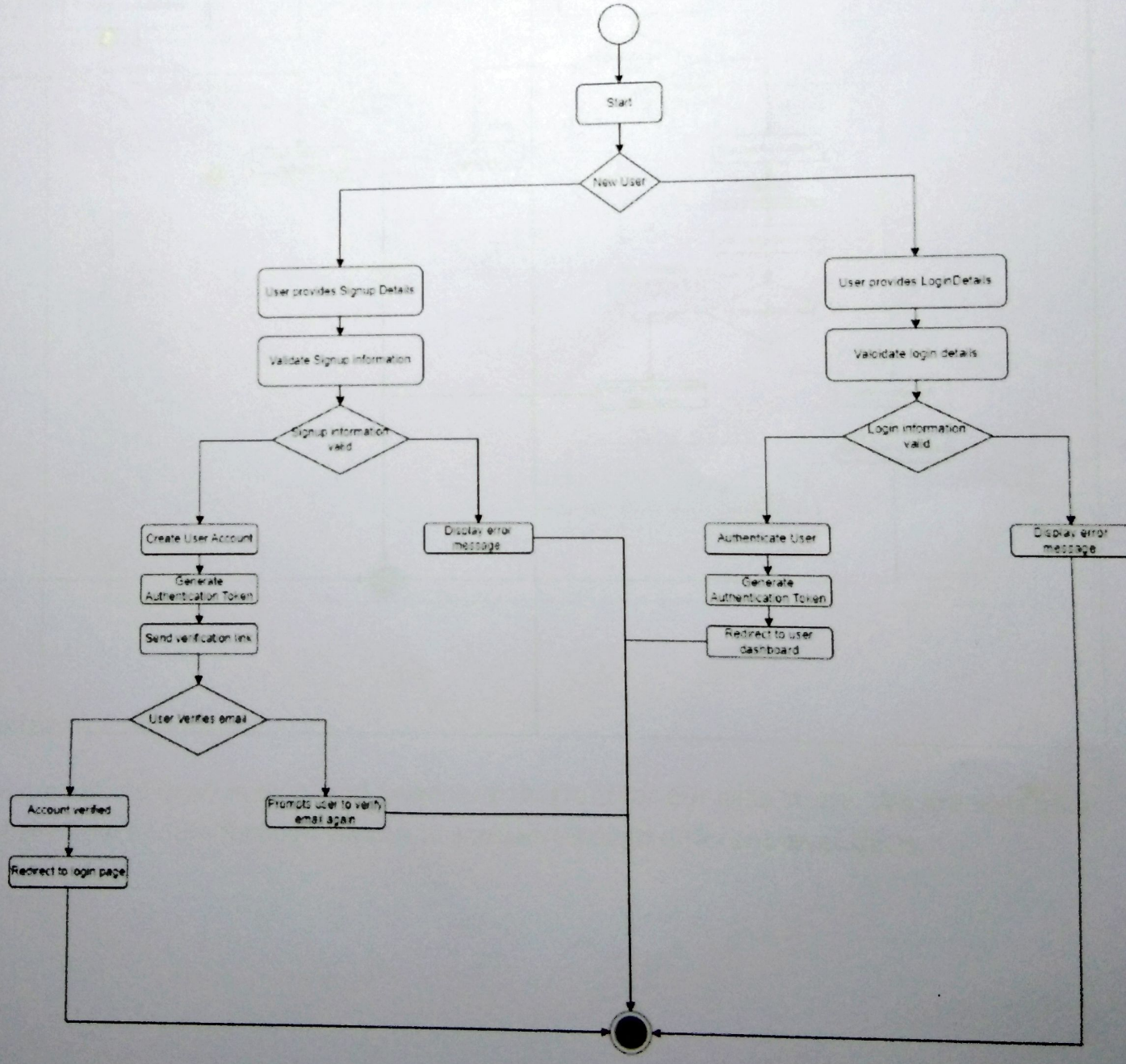
Amartya Mishra 60004210210

Gaurang Bhogle 60004210192

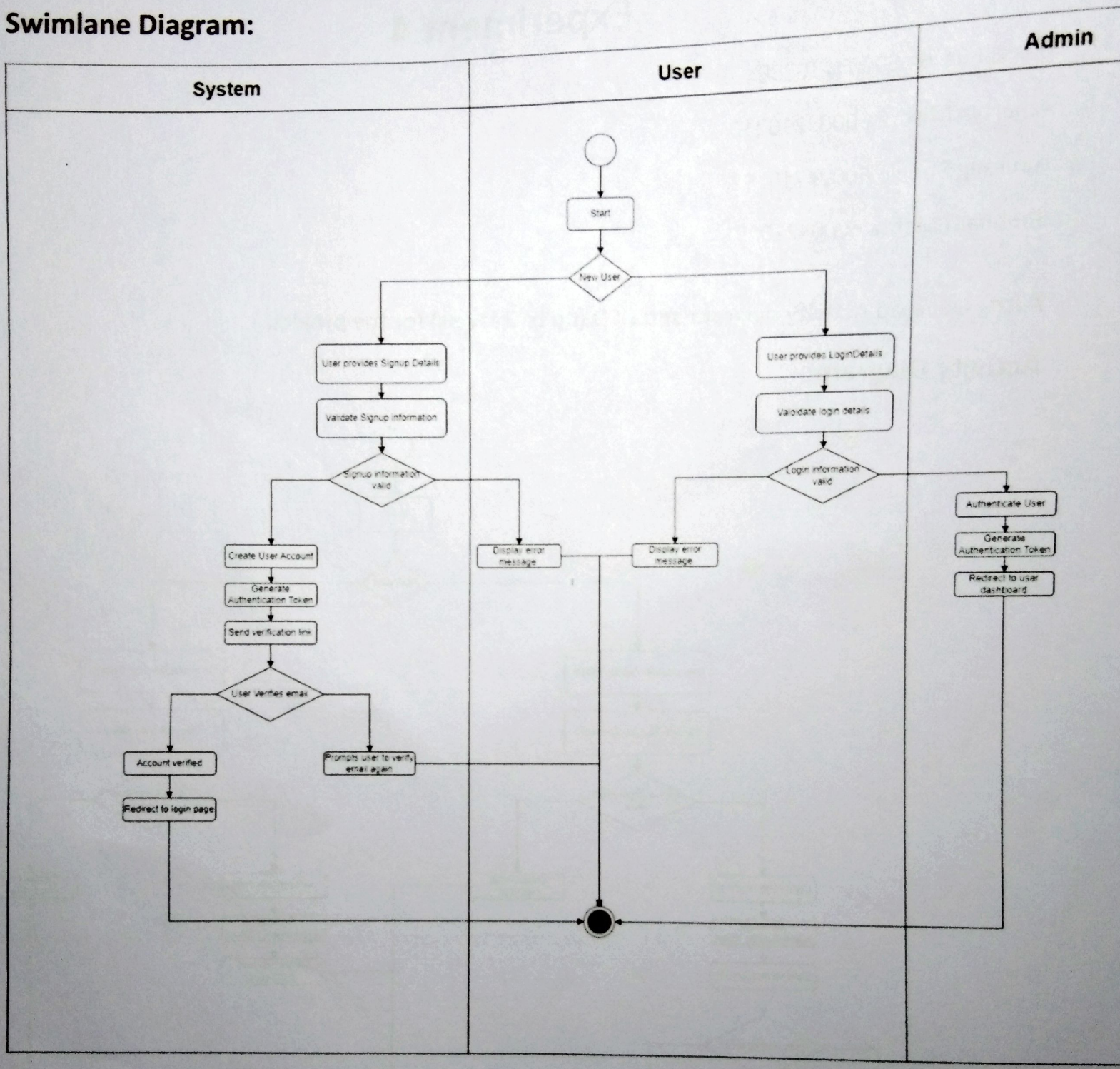
Shubham Mehta 60004210191

Aim: Develop Activity diagram and DFD (up to 2 levels) for the project.

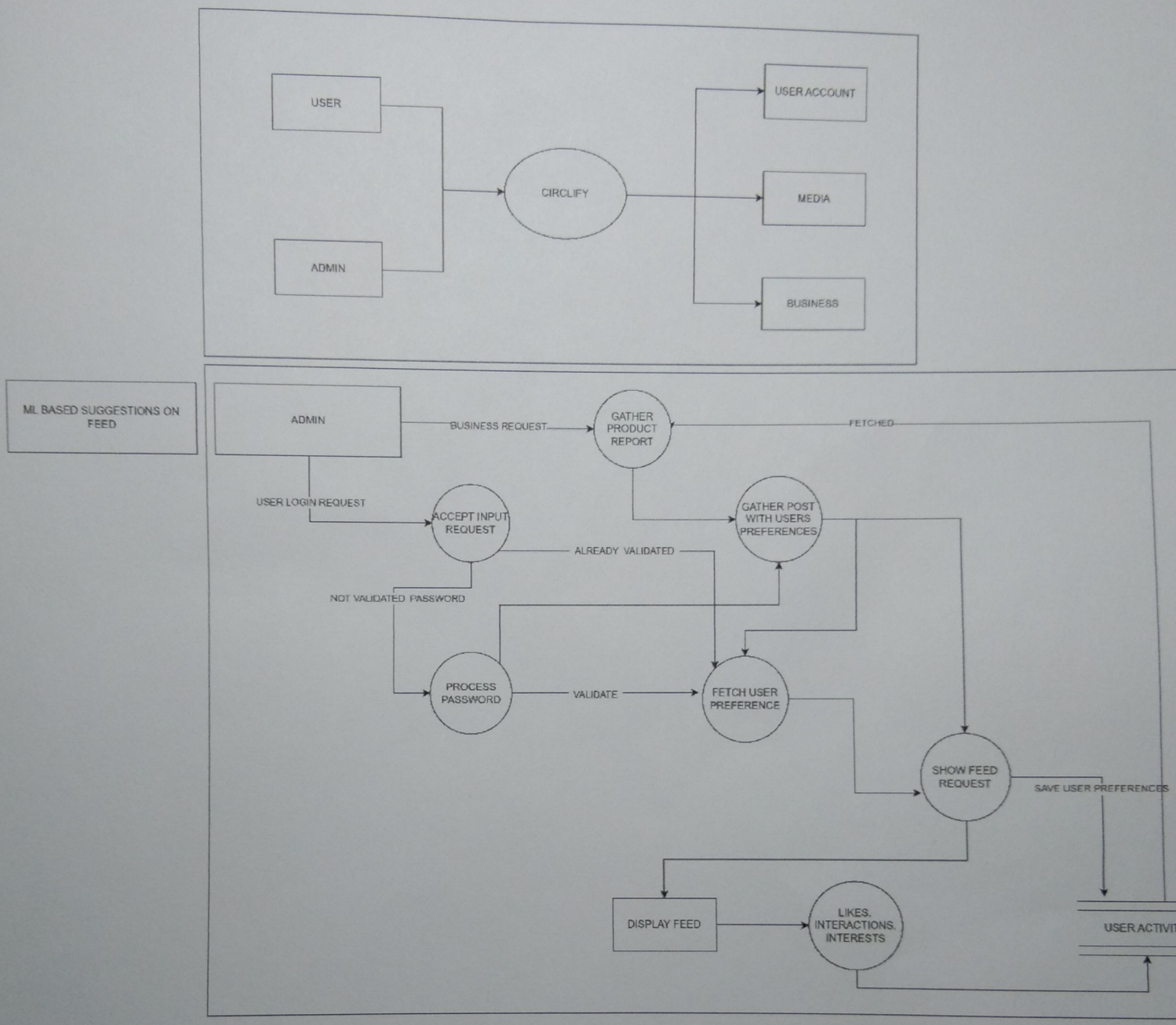
Activity Diagram:



Swimlane Diagram:



DFD:



Conclusion:

Thus, we are able to draw Activity and Swim lane diagram for our case study. We are also able to depict the flow of data through various processes through different level DFDs

Amartya Mishr
COMPS-C31

SE EXPERIMENT-5

Aim: Develop Sequence, Collaboration & State diagra for the project.

Theory:

Sequence diagram:

→ Screen listner: Allows user to interact with the app through Physical touch inputs.

→ feed: Display the recent post & content ~~to~~ most interacted with.

→ Post: content shared by user in form of ~~user~~ audio, vedio, ~~audio~~ etc. visible on apps ~~field~~ feed for others to view.

→ Media: Media is in form of audio, vedio or both, can be shared by user using various methods.

→ Profile: provides details of users personal & public information. for logged in users this page also has options to edit the profile.

State Diagram:

→ Signup login: Its the initial distination for new users where new creds are created.

→ Scrolling: It becomes accessible post credential Validation. allowing users to explore their feed

→ user can access feature of Direct message (DM) to share text & other medias in form of post or normal uploads from device itself.

→ Profile feature Enables user to view & edit their own profile details. User can make security changes, password changes etch here as well.

→ users can engage with media by sharing stories liking post & performing various preferences updat within the platform to access content the person likes more frequently.

Conclusion: Thus we are able to draw sequence collaboration & State diagram.

Experiment 5

Darshit Sarda 60004210208

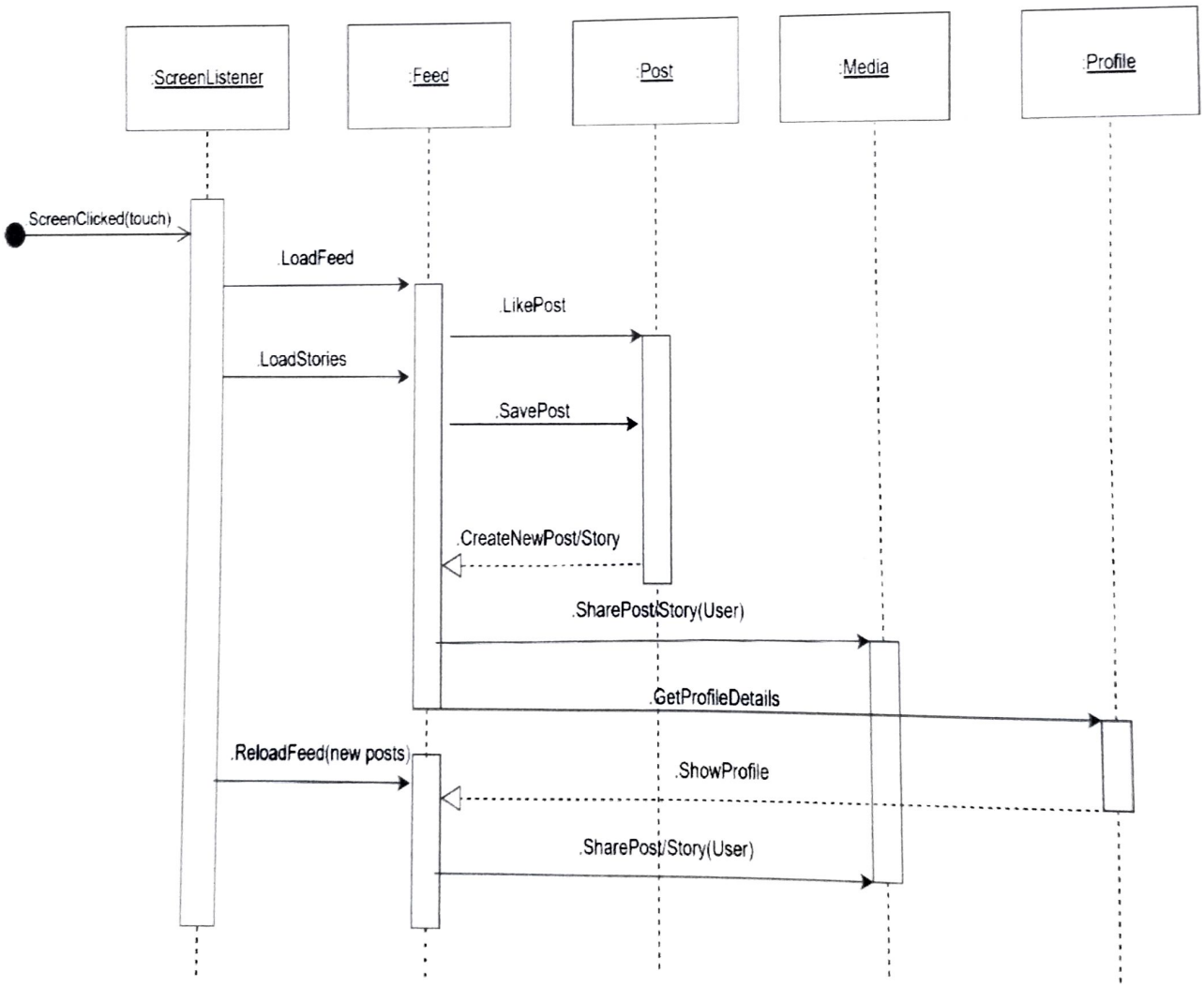
Amartya Mishra 60004210210

Gaurang Bhogle 60004210192

Shubham Mehta 60004210191

Aim: Develop Sequence, Collaboration and State diagram for the project.

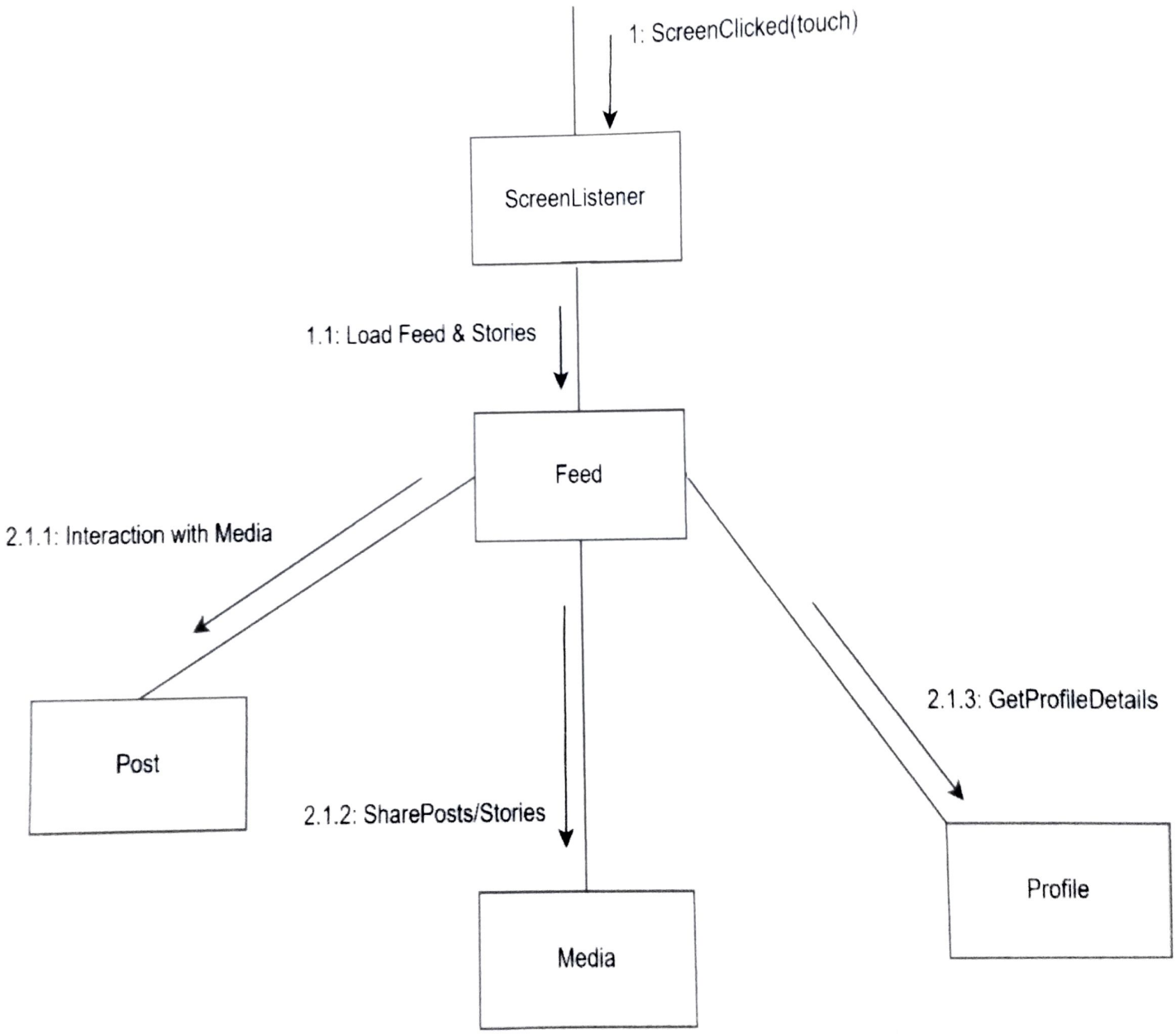
Sequence Diagram:



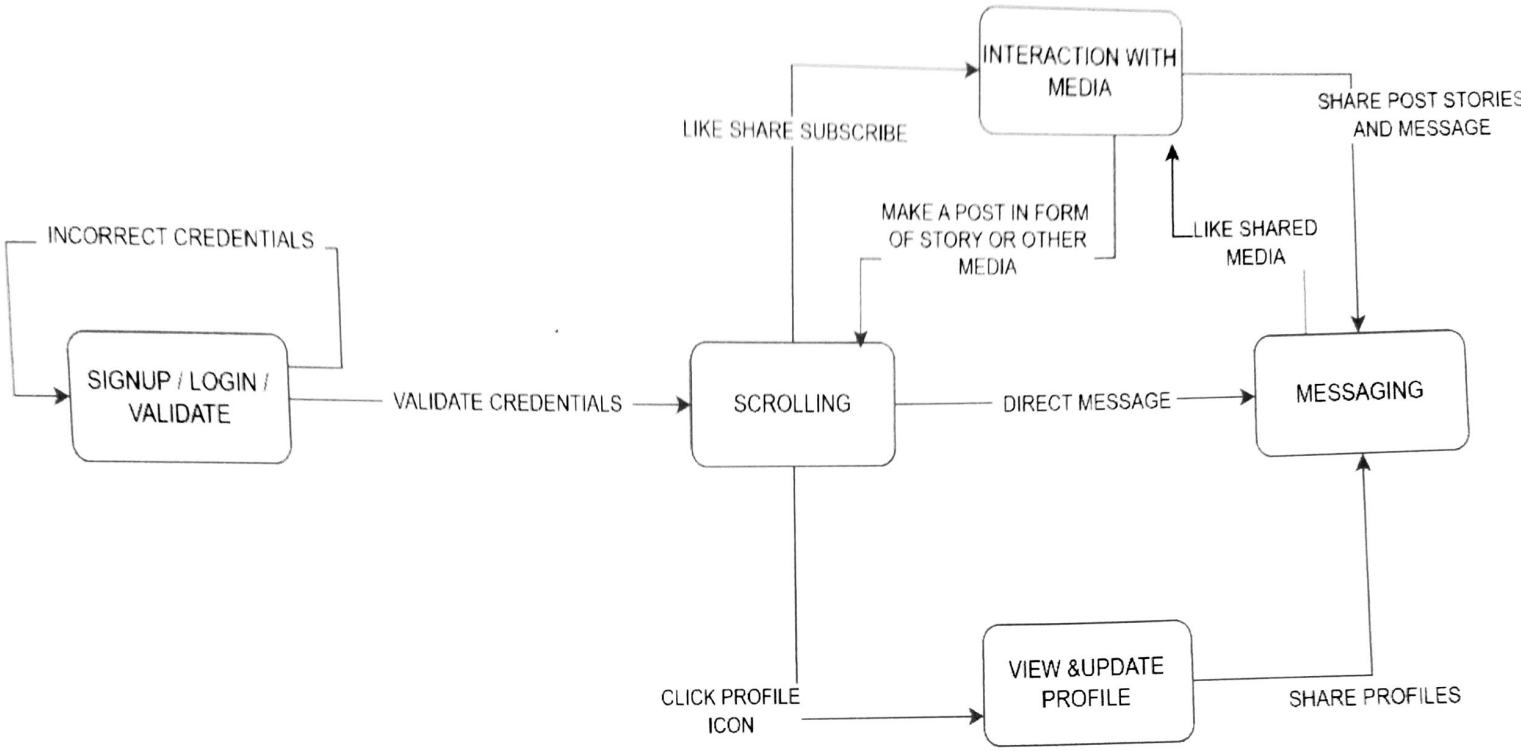
- ScreenListener: Touchscreen for interacting with the app by physical touch
- Feed: Loading the recent posts of the accounts you follow
- Post: Photos/videos posted by user that shows up on the feed

- Media: Sharing posts/stories (Media) to different users
- Profile: Details about every individual's profile on Circlify

Collaboration (Communication) Diagram:



State Diagram:



- Signup/login/validation page is the landing page for new users
- Scrolling can start after user has validated credentials and follows different accounts, consequently having posts load up on their screen
- User can share posts to other users via DM (direct message)
- User can click on profile to view and update their profiles
- User can also interact with media: sharing stories, liking posts etc.

Conclusion:

Thus, we are able to draw Sequence, Collaboration and State diagram for a functionality of our case study.

SE - Experiment 6

60004210210
Amartya Mishra
COMPS - C31

Aim: Estimate effort & cost required for project using FP/ COCOMO. Create WBS & Gaant chart for the same. use PM tool to depict a plan.

Theory:
FP calculation:

Info Domains	Value Cnt	Weight factor Simple	Avg	complex
EI's	3	3	4	6
EO's	2	4	5	7
EQ's	1	3	4	6
ILF's	4	7	10	15
EIF's	1	5	7	10

count

3 → External Inputs (EI's) : User login request, business request , Accept input request.

2 → External outputs (EO's): Show feed request, Save user preferance.

1 → External Enquiries : Gather Post with user preferenc

4 → Internal logic files: User Account, Activity, Media, Bussnu

1 → External Interface files: ML Based Suggestions on feed.

Cost drivers:

1) Does system require Reliable Backup & recovery?
2) Are specialized Data communication required
3) Are there distribute processing functons?
4) Is performance critical?
5) Will system run in heavily utilized Environment?
6) Does the system require Online Data Entry?
7) is Online Data Entry happening via Multiple screens & funct
8) Are IFL's updated online?
9) Are input output file inquiries complex?
10) is the internal processing complex?
11) Is the code Reusable?

$\Sigma Fi = 59$

FP Estimated = count - total $\times [0.65 + 0.01 \times (\Sigma Fi)]$

count total = 53

FP = $53 \times [0.65 + 0.01 \times 59]$

= 66

Proj Dural^r: 3 months = 22 person months

No. of Individuals = 4 $\therefore$ 22/4 = 5.5 PM/P

Conclusion: Thus we are able to calculate the efforts required for our project & prepare a Gaant chart.

Experiment 6

Darshit Sarda 60004210208

Amartya Mishra 60004210210

Gaurang Bhogle 60004210192

Shubham Mehta 60004210191

Aim: Estimate effort and cost required using FP/COCOMO for the project. Create WBS and Gantt Chart for the same. Use PM Tool to depict a project plan.

FP Calculation:

Information Domain Value	Count	Weighting Factor		
		Simple	Average	Complex
External Inputs (EIs)	3	3	4	6
External Outputs (EOs)	2	4	5	7
External Inquiries (EQs)	1	3	4	6
Internal Interface Files (ILFs)	4	7	10	15
External Interface Files (EIFs)	1	5	7	10

The counts for each component in the Function Point Analysis table were derived from the information provided in the data flow diagram.

External Inputs (EI):

Count = 3

- User Login Request
- Business Request
- Accept Input Request

External Outputs (EO):

Count = 2

- Show Feed Request
- Save User Preferences

External Inquiries (EQ):

Count = 1

- Gather Post with Users Preferences

Internal Logical Files (ILF):

Count = 4

- User Account
- Media
- Business
- User Activity

External Interface Files (EIF):

Count = 1

- ML Based Suggestions on Feed

The count represents the number of unique instances or occurrences of each component type identified from the data flow diagram. For example, there are three distinct External Inputs shown in the diagram, so the count for External Inputs is 3.

It's important to note that the accuracy of these counts depends on the level of detail and completeness of the information provided in the data flow diagram. As the project progresses and more detailed requirements are gathered, these counts may need to be adjusted accordingly.

Considering weighting factor as simple
FP estimated = count-total X [0.65+0.01*(∑Fi)]

count-total = 3*3+2*4+1*3+4*7+1*5

= 53

14 questions:

1. Will the application use data communications?
    - Yes (High Precedentedness) - 4
2. Are data or functions distributed?
    - No (Low Required Reusability) – 2

3. Are there specific performance objectives that must be met?
    - Yes (High Architecture/Risk Resolution) - 5
4. Will the application run on a heavily used configuration requiring special design considerations?
    - Yes (High Architecture/Risk Resolution) - 5
5. Will the transaction rate of the application be high?
    - Yes (High Product Complexity) - 5
6. Will there be on-line data entry?
    - Yes (High Product Complexity) - 5
7. Will the application be designed for end-user efficiency?
    - Yes (High Product Complexity) - 5
8. Will there be on-line updates?
    - Yes (High Product Complexity) - 5
9. Is complex processing logic involved?
    - Yes (High Product Complexity) - 5
10. Is there an intent to provide usability for other applications?
    - Yes (High Product Complexity) - 5
11. How important are installation ease and conversion?
    - Moderate (No specific information provided) - 3
12. How important is operational ease?
    - High (No specific information provided) - 4
13. Will the application be accessed from multiple sites?
    - No (Single Site development) - 1
14. Is there an intent with the design to facilitate change?
    - Yes (High Development Flexibility) - 5

$\sum Fi = 59$

FP estimated = 53 * [0.65+0.01*59]

= 66

Our project is having FP of 66

To calculate the productivity for each person based on the given information, we can follow these steps:

1. Calculate the total person-months required for the project using the given Function Points and project duration.
2. Divide the total person-months by the number of individuals working on the project to determine the average productivity per person.

Calculate total person-months:

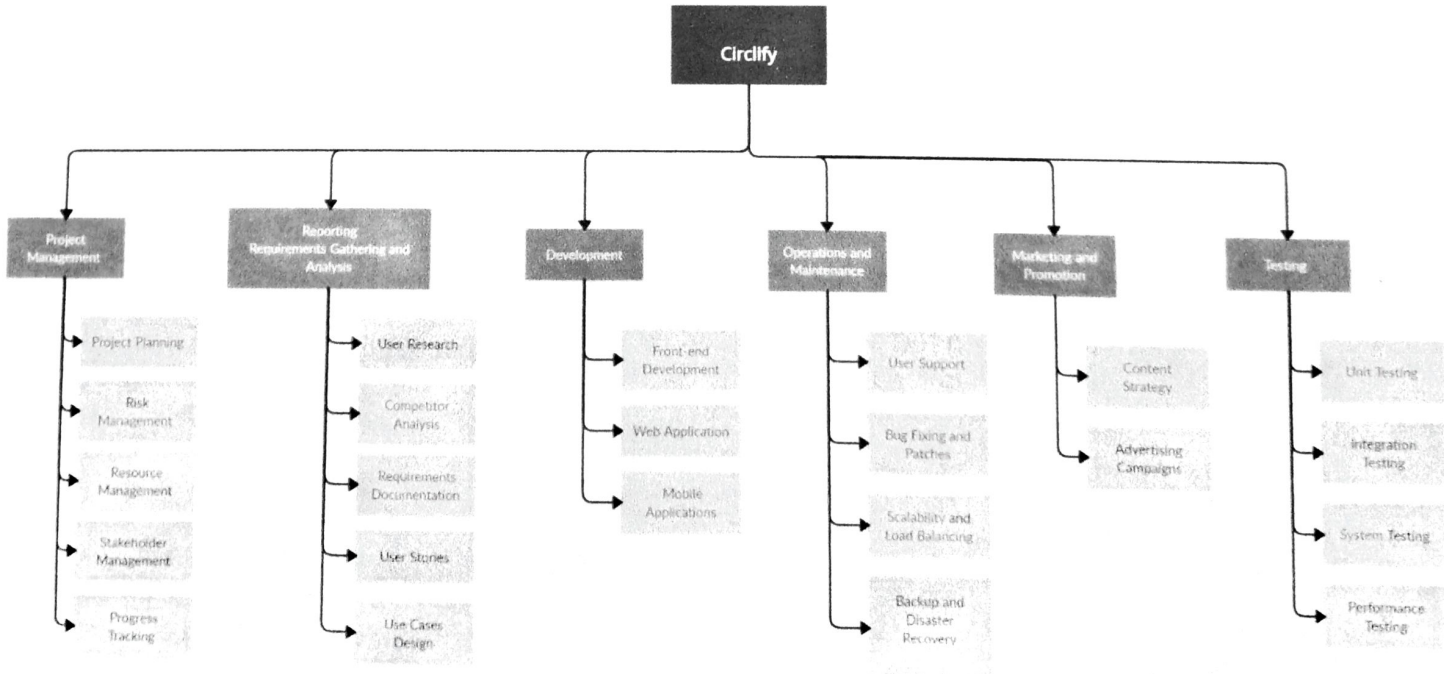
- Total Function Points (FP) = 66
- Project Duration = 3 months
- Total Person-Months = Total Function Points / Project Duration = 66 FP / 3 months ≈ 22 person-months

Determine average productivity per person:

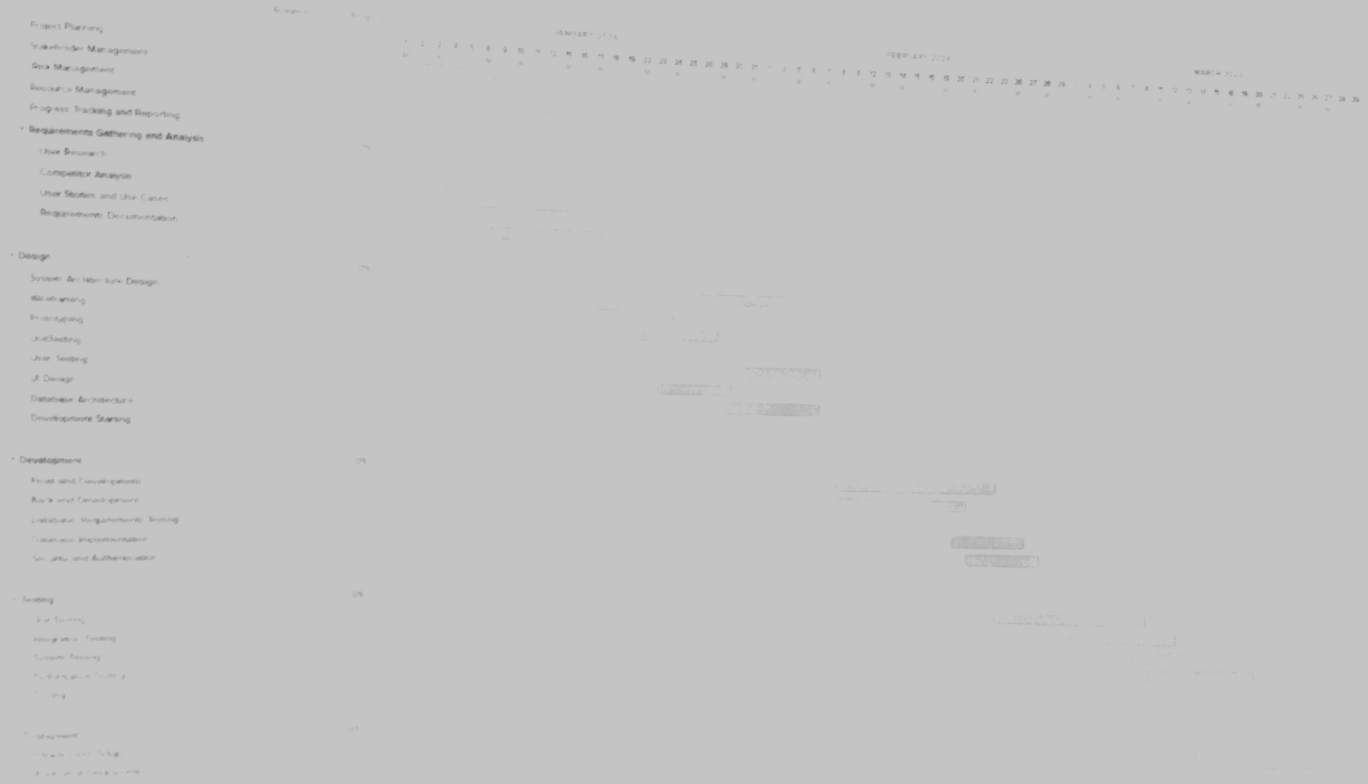
- Number of Individuals = 4
- Productivity per Person = Total Person-Months / Number of Individuals = 22 person-months / 4 ≈ 5.5 person-months per person

So, based on the given information, the average productivity per person for the project is approximately 5.5 person-months.

Work Breakdown Structure:



Gantt Chart:



Conclusion: Thus, we are able to estimate effort required for our project and also create Gantt Chart.

Circlify social media platform

Circlify SRS document

Amartya

4-Feb-14

10-Feb-14

Test scenario description	Impor-tance	No. of test cases
login signup for existing users	High	3
post media	Medium	
share media in form of audio videos and text	Medium	3
send follow request and accept request	Medium	3
ad reports (test wether ads have caused any change in number of purchases of the product)	Low	3

	Circlify social media platform
	circlify SRS document
	Amartya
	1-Jan-24
	29-Mar-24

Test case ID	Test Objective	Precondition	Steps:	Test data	Expected result	Post-condition
TC_1.1	To verify if a user with valid credentials can successfully login.	1. A valid User account exist and is registered 2. Good internet connection	1. In the login Pane, enter the username	"A valid username" Enter the actual data in your real time situation	The user is logged in successfully. There is only one expected result for the entire test case.	For first time users personal informatio is displayed. Note: This info is only additional. Just a a pointer to the tester
TC_1.2	password valid	1. Correct username inserted in the field	1. In the login Panel, enter the password	"A valid password format with specified length"	An Error message is displayed and the user is not logged in to the Orange HRM portal. "<Exact Error Message>"	As you can see, post condition can be left empty when there is nothing else add
TC_1.3	To verify if a user can sign up for a new account.	1. AUser name to login to be available for a first time user of the site. 2.site is launched on a compatible browser/ device	1. In the login Panel, enter the username	"A valid username"		
			2. Enter the Password for the User account in the password field	"A valid Password"		
			3. Click "Signup" button		The user is Registered successfully and the personal information page is displayed for further details	User account is created and the user registered in the Database
TC_2.1	Ability to upload media in form of image vedio and audio	1. User is logged in 2. Required permissions are granted to the application	1. Ask for permission to upload media from device		Permission granted thus access device data	
			2. Select the media from the device menus	sample data to upload in audion vedio and image format		
			3. Click on "create post"		The first name field needs to now show the new value entered	
TC_2.2	Check the upload of image	Permissions granted to access device storage and media	select photo in a valid Jpeg format	Test images in valid format	The edtiting image field is visible	
			Choose a image fie of type "JPG" that is less than 1 MB	Name of the image Location-path on the machine		
			Click on upload		The file gets uploaded	

TC_2.3	Check the upload of video	1. User logged in 2. A valid video to upload that	select a valid video from device		The edit vedio options is displayed	
			Click on upload		The file gets uploaded and the older image is replaced	
TC_3.1	Test ifuser is able to send post	1.Valid user login 2. User1 sending the post follows User2 receiving the post	Click on send post	A sample post	Share media screen shows up with user names of account User1 can send post to	
TC_3.2	Test if user able to send stories	1.Valid user login 2. User1 sending the story follows User2 receiving the post	Click on share stor	A sample story	Share media screen shows up with user names of account User1 can send post to	
TC_3.3	user able to direct message some one	1.Valid user login 2. User1 sending the post follows User2 receiving the message	Click on send Direct message icon	A sample account to receive and send text	The Direct messaging page shows up with all the account whome User1 has recently contacted	A messaging thred is created for User1 and User2
TC_4.1	Test if user is able to send follow request to User2	1.Valid user login 2. User1 sending the post follows User2 receiving the post 3. User2 has a account on the platform	1.Search for user account 2. On selecting the account click on send request	user account to send request to	the page updates the button of send request to sent	A requst thread is created which show User1 is sending request to User2
TC_4.2	Test If user able to accept follow request	1.Valid user login 2. User1 sending the post follows User2 receiving the request.	1.Click Accept request 2. Show option of follow back if User2 not already following the User1.		The User1 is added as a follower to person accepting the result.	
TC_4.3	Test If user able to remove followers and unfollow	1.Valid user login 2. User1 sending the post follows User2 receiving the request. 3. User1 following User2 and viceversa .	1.Select User2 account 2. Show option of unfollow User2. 3.Click on the unfollow button		Unfollow leads to deletion of relation of User1 User2 follower following instance	
TC_5.1	track user activity	1. User agrees to sharing information to app 2. User has some activity on the app	1.User Post and story interactions are recorded 2.User likes and dislikes are recorded 3.Report based on what user app usage patterns are is generated.	Sample User activity data	User activity is recorded and report is generated	
TC_5.2	test if ads worked	1. User has interacted with ads 2. business individuals have provided the data of customers aquired through Circlify	1.User ad interactions are recorded 2.User likes and dislikes are recorded 3.User buying patterns patter algorithms discovered by the algorithm	Sample User activity data	User activity is recorded and report is generated With detailed analysis of ads impacting user buying patterns	
TC_5.3	push ads based on activity	1. User has interacted with ads 2. business individuals have provided the data of customers aquired through Circlify	1. Algorithms filter the relevent ads from the various business stake holders 2.Based on filtered ads select ads which are best suited for the user .	sample ads and user activities matching the ad categories	Track the ads displayed and await the report for these ad performaces to submit to business stake holders	

SE - Experiment - 8

60004210210
Amartya Mishra
COMPS - C31

Aim: To create a RMMM plan. Create Risk assessment template for a case study.

Theory:

Risks:

- Compatibility issues accross various devices & OS.
- Poor quality Documentation.
- Under Estimating Data Size of user Data & interaction
- absence of clear monetization strategy.
- lack of structured change control process
- Change in requirements
- Lack of Development Experience
- deviation from Software standards.
- late delivery
- Significant Evolution of customer requirement while development

Risk Table:

Risk	Category	Prob	Impact
Compatibility issue	TI	75	1
Poor documentation	BU	75	1
under Estimate Data Size	PS	70	2
cutoff			
unclear Monitizat^ Strategy	BU	45	2
change in Requirement	PS	35	2
Lack of dev Exp	TI	20	3
late delivery	BU	20	4
change of Business Req	CR	15	4

Rsk for RMMM: under Estimating Data size for user Data & unter action.

RE (Rusk Exposure) : Prob of Rusk x impact of Rusk

RE = 0.7 x $50,000
RE = $35,000

Conclusion:
We success fully created A RMMM Plan.

Risk Id : 10001010DBS	Date: 15/02/2024	Probability : 80%	Impact: High

Description: Underestimating the necessary database size could lead to potential performance issues, data loss, or system crashes due to insufficient storage capacity.

Refinement/ Context:

1. Subcondition1: Initial user base and the reach of the app was underestimated with incorrect estimation of user base size.
2. Subcondition2: The design of system was calibrated to cater to 100000 request per second which might be less than the current request rate.
3. Subcondition3: The amount of media uploaded is unrestricted leading to overflowing memory and loss of uploaded media and affecting the application features.

Mitigation / Monitoring:

1. A team of few members to actively monitor the growth rate of the database.
2. Develop an algorithm to predict the time before capacity overflow of database
3. Implement data limits on users to limit the amount of data being uploaded by the user

Management/ Contingency plan / trigger:

1. The team cost approximates INR 1, 20, 000 given a team of size 3 and hiring new developers who can develop the algorithm to monitor and predict the data overflow date.
2. For maintaining Cloud services to altogether avoid the given situation the total cost incurred will be INR 4, 00, 000.

Current Status:
Mitigation steps initiated.

Originator: Darshit , Gaurang

Assigned: Amartya , Shubham

SE Expt. 9

Darshit Sarda -60004210208

Amartya Mishra-60004210210

Gaurang Bhogle-60004210192

Shubham Mehta-60004210191

Aim: Study of Configuration Management using GitHub.

Theory:

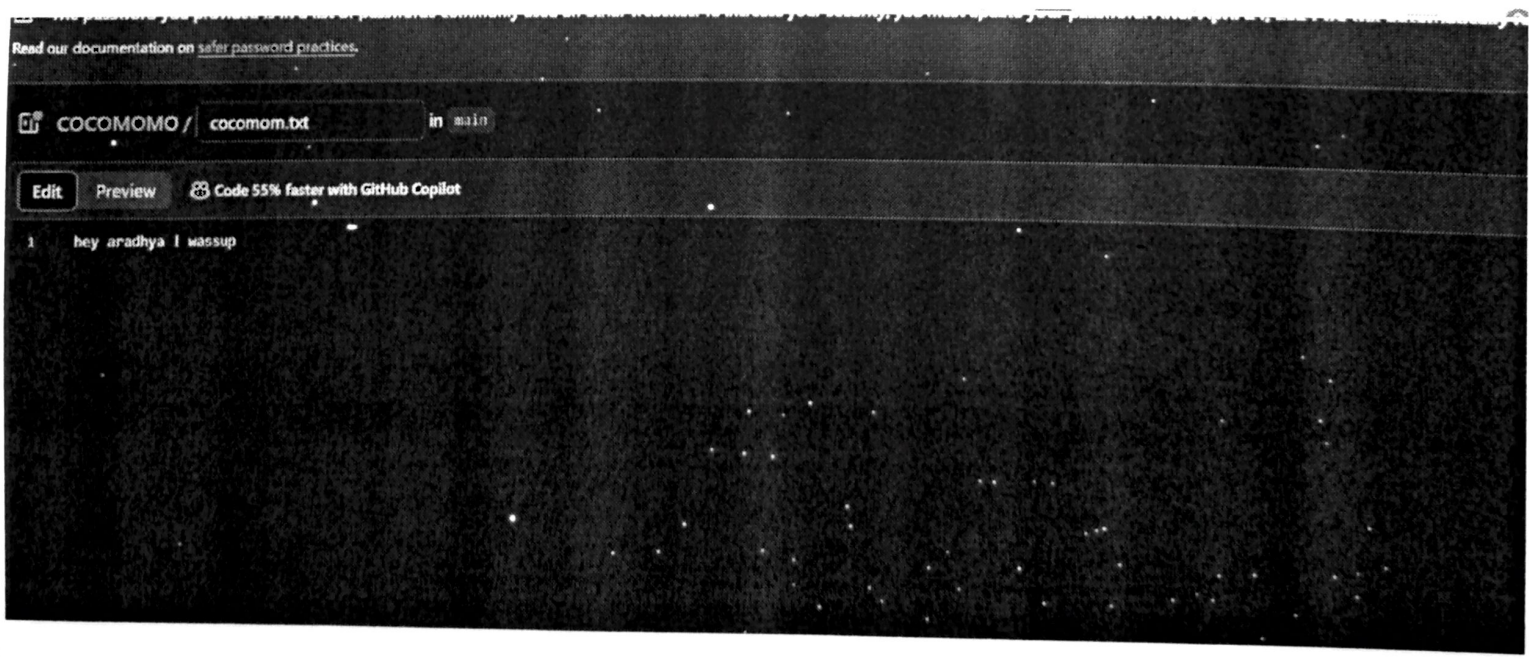
- To demonstrate and resolve a conflict in a GitHub repository with diagrams and an SRS uploaded, you can follow these steps:
- Create a Conflict: Make changes to a file in your repository from two different branches. For example, you can edit the SRS file in one branch and update a diagram in another branch.
- Commit Changes: Commit the changes to each branch separately. Ensure that the changes conflict with each other, meaning they modify the same lines of code or content in the files.
- Merge or Pull Request: Merge the branches or create a pull request to merge one branch into another. This action will trigger a conflict if changes from both branches affect the same lines in a file.

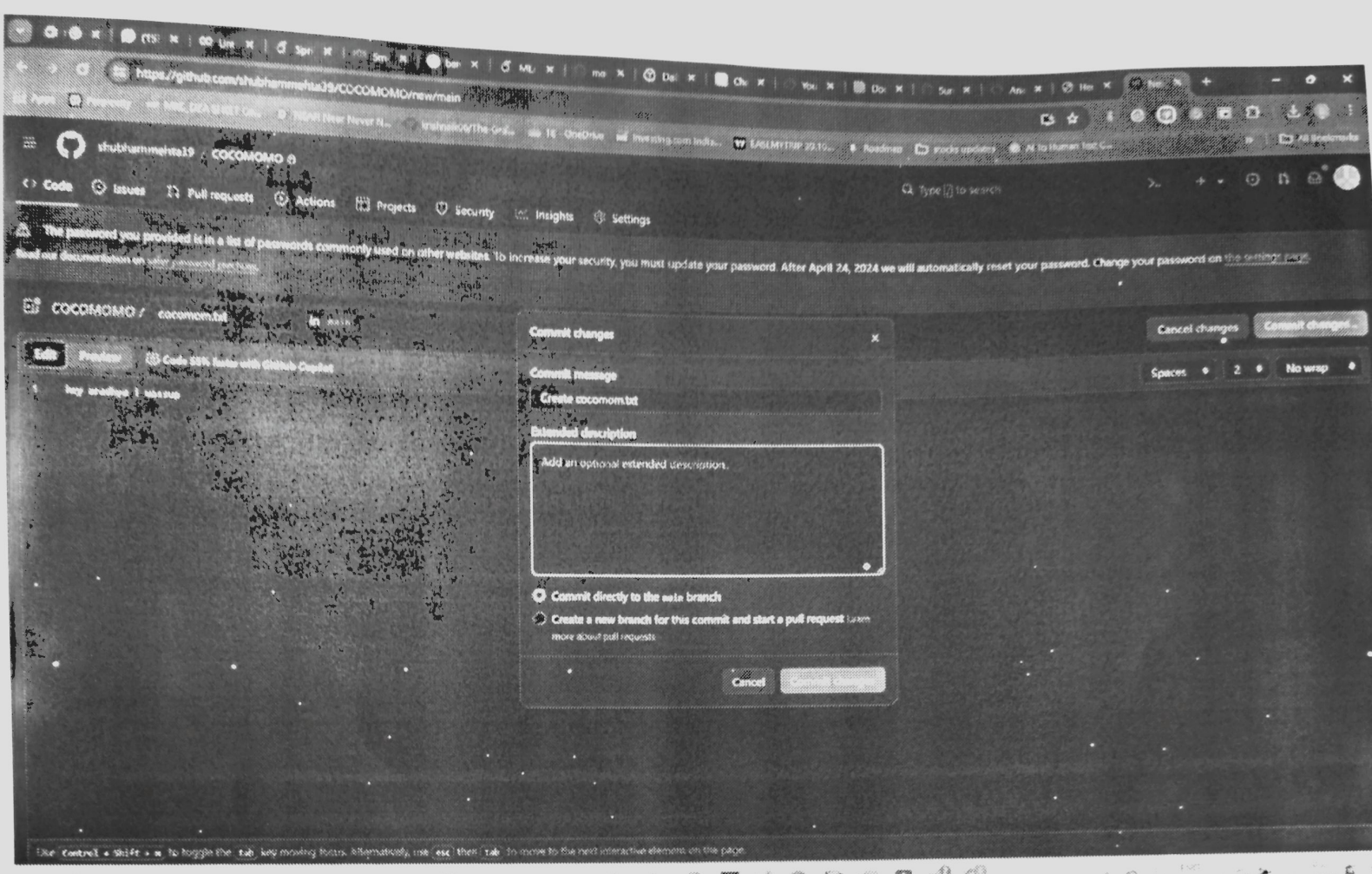
Resolve Conflict:

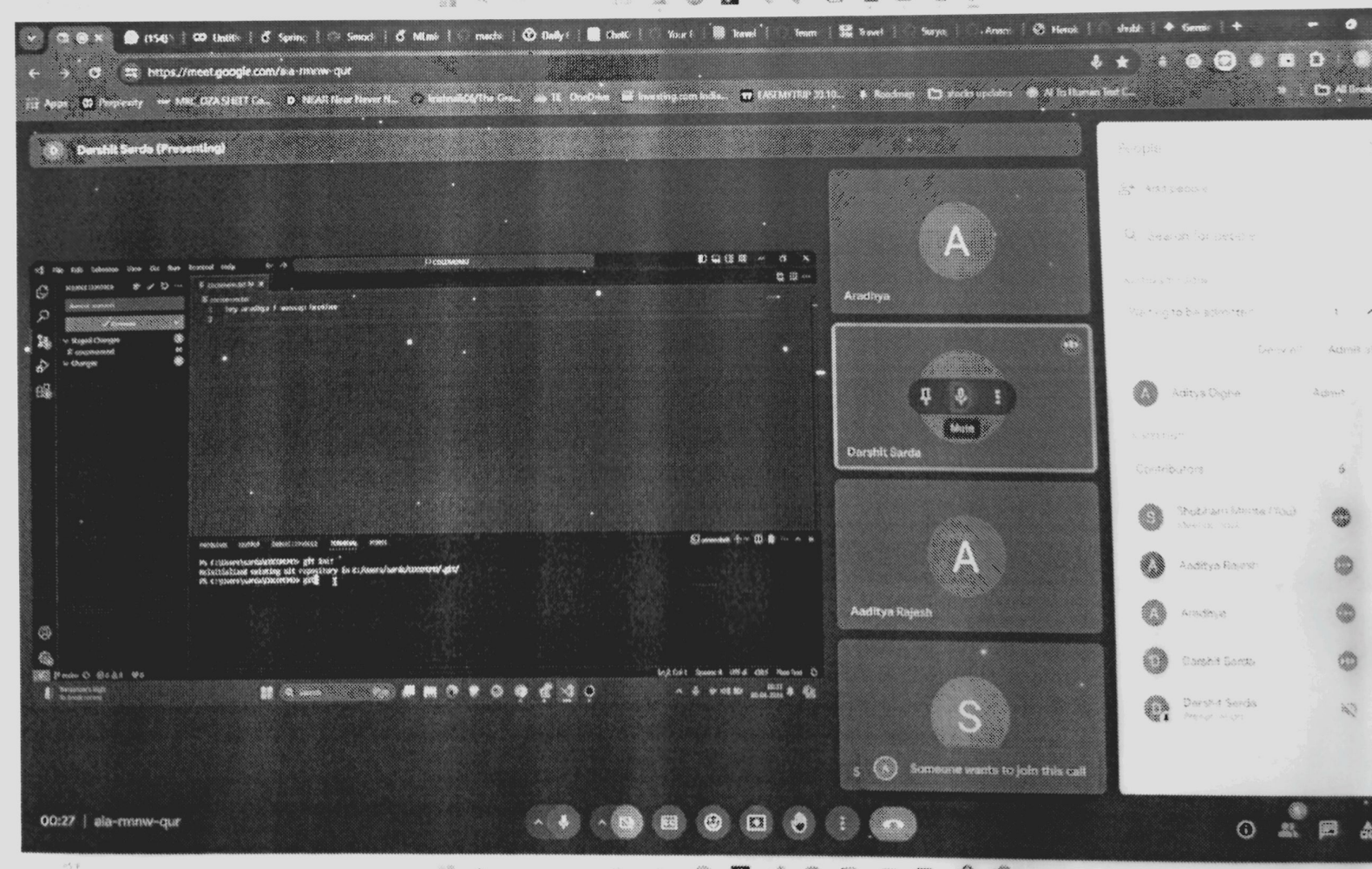
  - Open the affected file(s) in your preferred text editor or integrated development environment (IDE).
  - Locate the conflicting lines marked by Git. These lines will typically have both versions of the changes from each branch, surrounded by conflict markers (<<<<<<<, =======, >>>>>>>).
  - Edit the conflicting lines to resolve the conflict, keeping the changes that you want to keep or combining them as needed.
  - Remove the conflict markers (<<<<<<<, =======, >>>>>>>) once you have resolved the conflict.
  - Save the changes to the file(s).

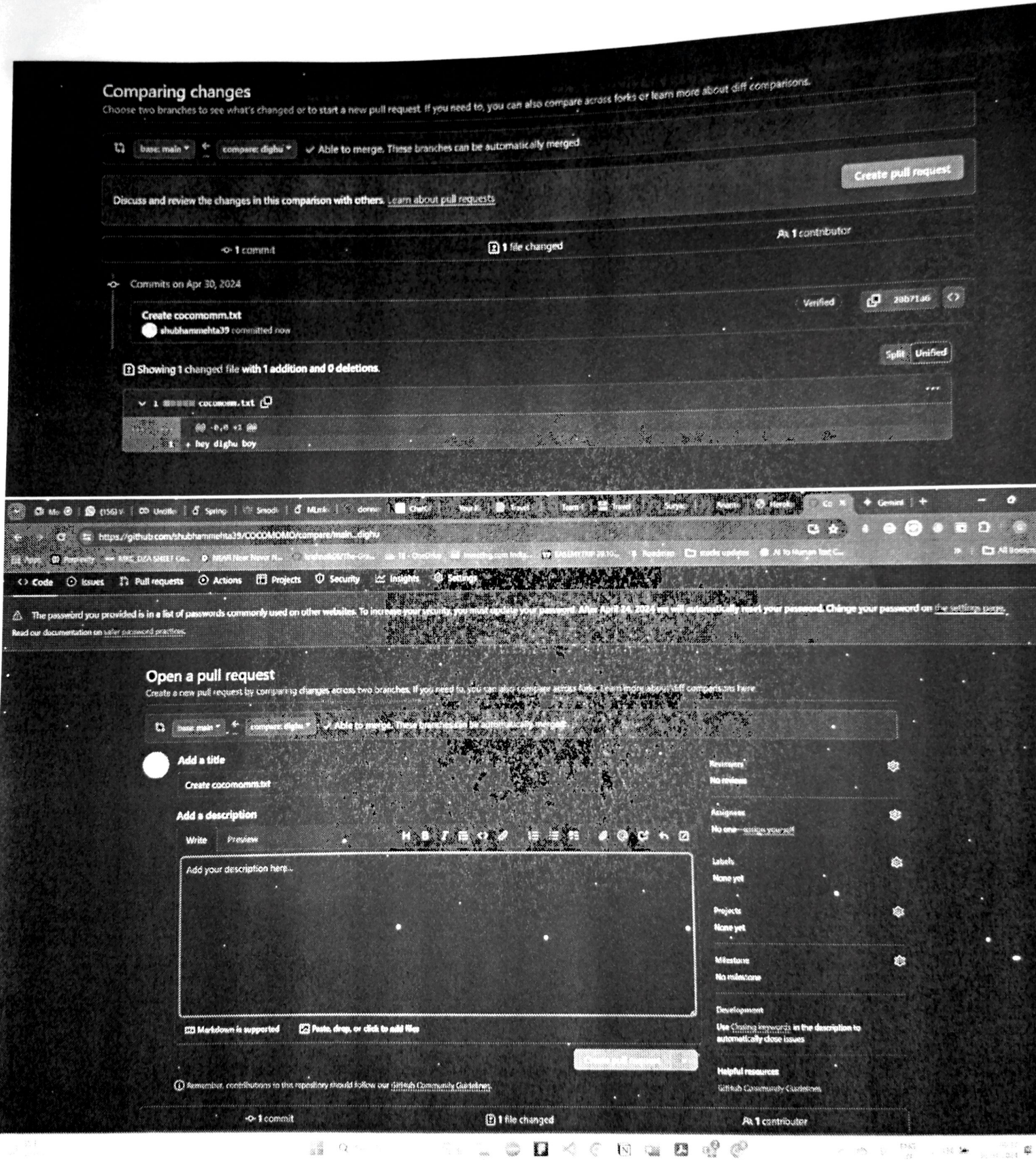
- Commit Changes: After resolving the conflict, commit the changes to complete the merge or pull request process.
- Push Changes: Push the merged changes to your GitHub repository to update it with the resolved conflict.
- Verify: Verify that the conflict has been resolved correctly by reviewing the files and ensuring that they contain the intended changes from both branches.

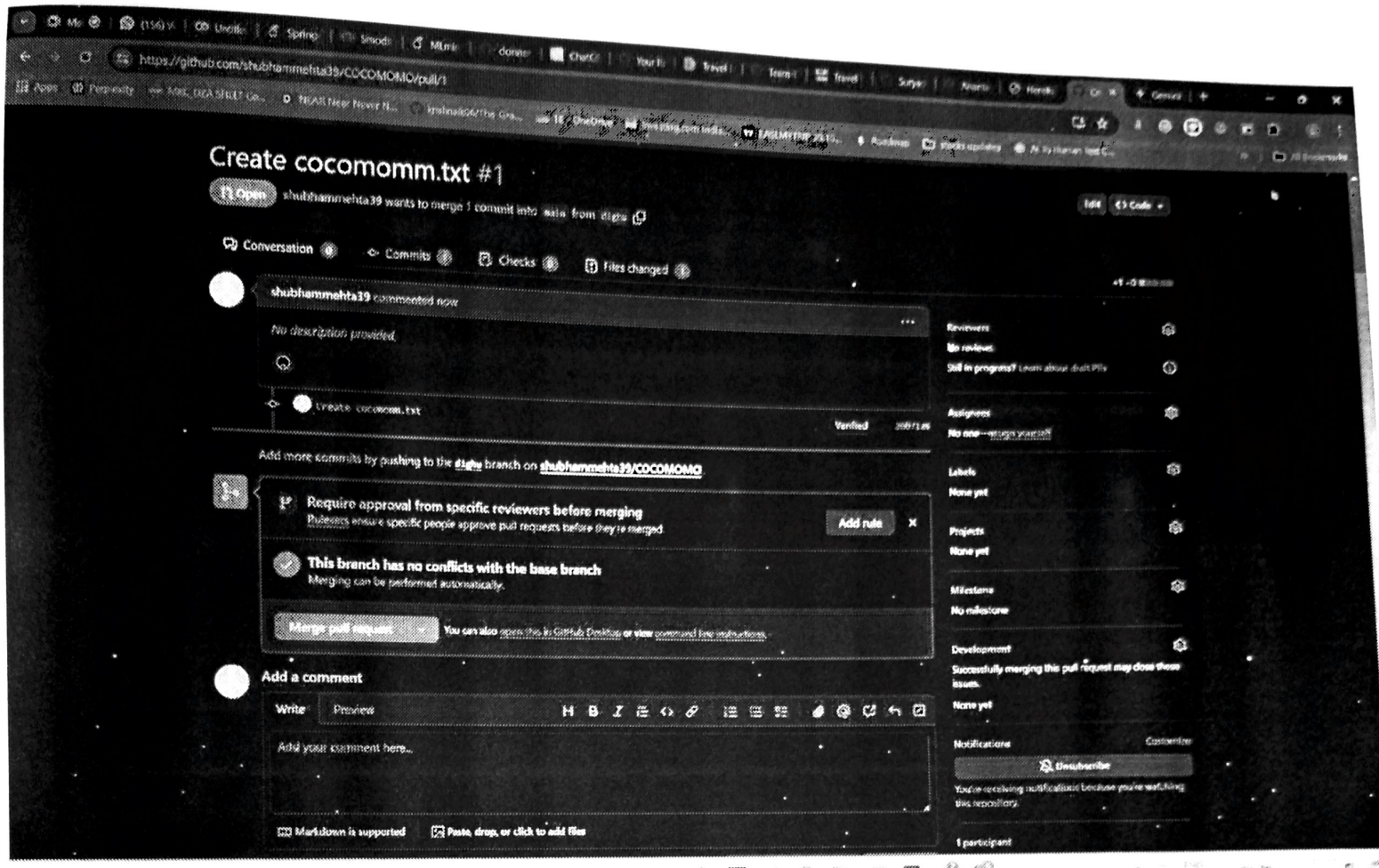

COCOMOMO
Private
Unwatch 1
Go to file
Add file
main
1 Branch
0 Tags
52a31b3 · 7 minutes ago
2 Commits
shubhammehta39 Add files via upload
9 minutes ago
README.md
Initial commit
7 minutes ago
SE_Exp09_Circlify.pdf
Add files via upload
7 minutes ago
SE_Exp10_Circlify.pdf
Add files via upload
7 minutes ago
SE_Exp1_Circlify.pdf
Add files via upload
7 minutes ago
SE_Exp2 SRS_Circlify.pdf
Add files via upload
7 minutes ago
SE_Exp3_Circlify.pdf
Add files via upload
7 minutes ago
SE_Exp4_Circlify.pdf
Add files via upload
7 minutes ago
SE_Exp5_Circlify.pdf
Add files via upload
7 minutes ago
SE_Exp6_Circlify.pdf
Add files via upload
7 minutes ago
SE_Exp7_SampleTest-Cases-Circlify.xlsx
Add files via upload
7 minutes ago
SE_Exp7_SampleTest-scenariosCirclify.xlsx
Add files via upload
7 minutes ago
SE_Exp8_Circlify.pdf
Add files via upload
7 minutes ago

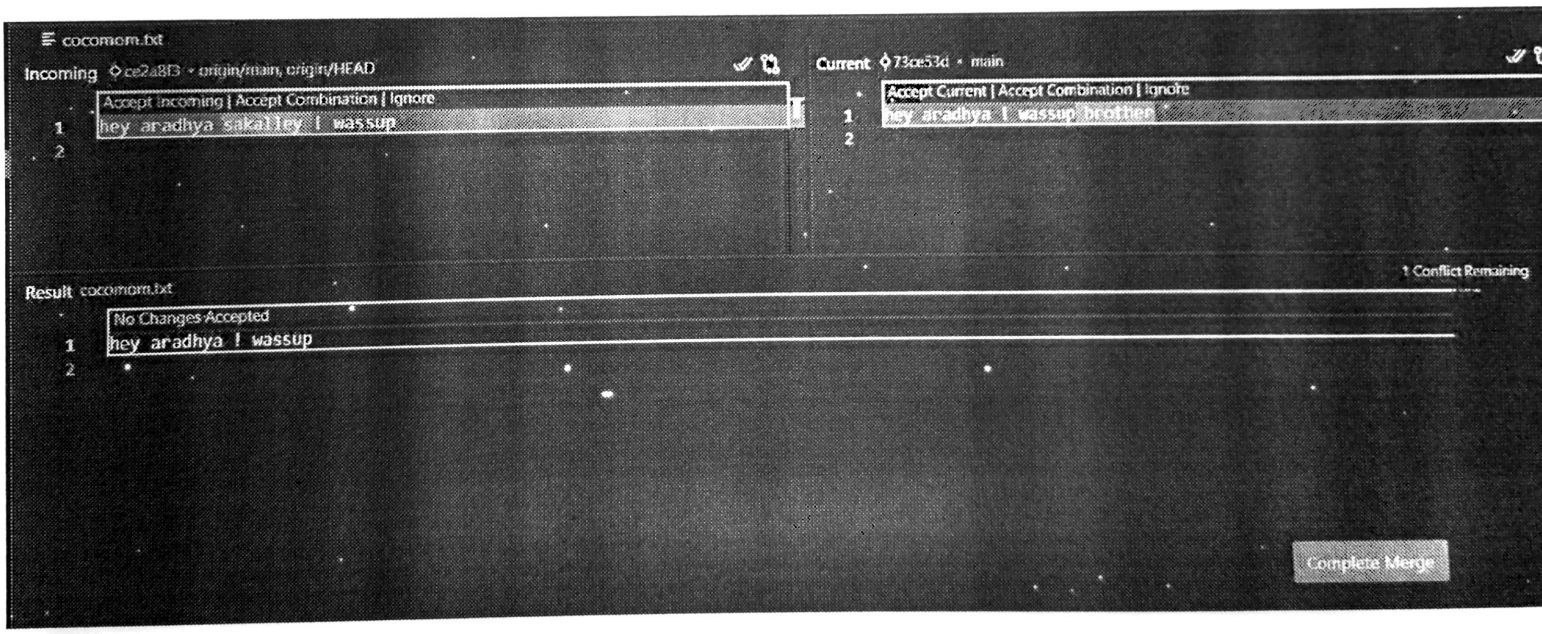

Read our documentation on safer password practices.
COCOMOMO / cocomom.txt in main
Edit
Preview
Code 55% faster with GitHub Copilot
1 hey aradhya I wassup


https://github.com/shubhammehta19/COCOMOMO/new/main
shubhammehta19 / COCOMOMO
Code
Issues
Pull requests
Actions
Projects
Security
Insights
Settings
COCOMOMO / cocomom.txt
Commit changes
Commit message
Create cocomom.txt
Extended description
Add an optional extended description...
Commit directly to the main branch
Create a new branch for this commit and start a pull request
Cancel changes
Cancel


https://meet.google.com/aia-rmnw-qur
Darshit Sarda (Presenting)
Aradhya
Darshit Sarda
Aaditya Rajesh
Someone wants to join this call
00:27 | aia-rmnw-qur


Comparing changes
Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks or learn more about diff comparisons.
base: main
compare: dighu
Able to merge. These branches can be automatically merged.
Create pull request
Discuss and review the changes in this comparison with others. Learn about pull requests
1 commit
1 file changed
1 contributor
Commits on Apr 30, 2024
Create cocomomm.txt
shubhammehta39 committed now
Verified
28b714e
Showing 1 changed file with 1 addition and 0 deletions.
Split
Unified
cocomomm.txt
@@ -0,0 +1 @@
+ hey dighu boy
Code
Issues
Pull requests
Actions
Projects
Security
Insights
Settings
Open a pull request
Create a new pull request by comparing changes across two branches.
Add a title
Create cocomomm.txt
Add a description
Write
Preview
Add your description here...
Markdown is supported
Paste, drop, or click to add files
Reviewers
No reviews
Assignees
No one—assign yourself
Labels
None yet
Projects
None yet
Milestone
No milestone
Development
Use Closing keywords in the description to automatically close issues
Helpful resources
GitHub Community Guidelines
Remember, contributions to this repository should follow our GitHub Community Guidelines.
1 commit
1 file changed
1 contributor





Con;usion

Hence we successfully studied and implemented configuration management using GitHub.

SE - Exp 10

60004210210
Amarsya Mishra
COMPS-C31

Aim. Study of Dev-Ops

Theory:

- A no. of online Devop tooles are available which are Open ~~Sour~~ source & free to use
- select set of tools for the various stakeges of SW Dev makes a ~~&~~ Dev Ops tool chain
- Diff. tools are used in different stages.

→ Collaboration: when a team working across diffrent part of project is able to create & manage the project at once, Its done using collaborative tools The Tool used is Docker / skype / slack

→ Planning: used to plan the development of product across various phases (ASANA, clarizen)

→ Source control: The Base version is fixed, changes are made on new Branches & merged (Git / Svn Version)

→ Issue tracking: Tracking issues & Point of failure possible using Dev-Ops (Jira, Zendesk)

→ config management: auto config tools avail (Ansible / Puppet / chef)

→ Continuous Integration - (Jenkins / team city)

→ Binary Repositories (RT factory / Nexus / mavan)

→ Monitoring (Big Panda & Sensu)

→ Automating all development & testing - ~~Ansible~~ Selenium
→ Database Integration. (Razor SQL, Team Desk)
→ Development (Docker & IBM)

Conclusion : Thus we listed all the phases & tools used in the Devops tool chain.

SE - Assignment 1.

6000 42102 10
Amartya Mishra
COMPS - C31

Q1 Elaborate Task set for creating component level design on OO project.

⇒ The Task set for creating ~~OO~~ component level design on OO projects

1) Identify all design corresponding to problem Domain
2) Identify all design classes that correspond to Infrastr. Dom.
3) Elaborate all design classes that are not acquired as reusable component.
   (i) Specify message Details when classes / components collaborate
   (ii) Identify appropriate interfaces for each component
   (iii) Elaborate attributes & define data types & Data-Structure required to implement them.
   (iv) Describe processing flow within each operation in detail.
4) Describe present DS & identify the classes required to manage them.
5) Develop & Elaborate Behavioral Representation for a class or component.
6) Elaborate Development diagrams to provide additional implementation Details.
7) factor every component level design Representation & always consider alternatives.

2) The golden Rule of user Interface are:

i) visibility
ii) ~~Feed Back~~ Feed Back
iii) Consistency
iv) Flexibility.
v) Simplicity

Methodology

a) Place user in control
b) Reduce users Memory load
c) Make interface Concistent

Define a Interaction modes in a way that does not force a user into unnecessary action.
Provide for flexibility in action
Streamline Interaction with increasing skill level & allow customized interaction.
Hide technical internals from casual users
Design for direct interaction with objects that appear on screen.

d) Reduce users Memory load

Reduce demand on short term memory
Establish meaning full defaults.
Define short-cut Intiutives
Disclose information in progressive fashion

e) Make interface concistent

Allow user to put current task into a meaning ful Content.

(3) Transform mapping is a process used in Software Engineering to map data flow from source to destination

It is a set of design step that allow a DFD with transform flow characteristics to be mapped into a specified architecture style

To map DFD into a software Architecture, we would initiate the following steps:

1) Review the fundamental system model.
2) Review & Refine Data flow diagram for software
3) Determine wether DFD has transform transaction flow characteristics.
4) Isolate the transform center by specifying incoming & outgoing flow boundries.
5) perform ~~furs~~ first level factoring.
6) Perform second level factoring.
7) Refine first iteration Architecture using design heuristic for improved ~~&~~ software

(4) Transaction mapping ~~is~~ involves mapping the flow of transaction within a system

It helps ensure that all necessary steps are identified & Executed correctly

Step:

→(1) Identify logical Transactions
→(2) Each logical transaction is mapped to a series of DB Operation
→(3) Transaction mapping Ensures automatic Transaction

(14) It involves optimizing performance of transaction

SE - Assignment 2

60004210210
Amartya Mishra
COMPS - C81

(1) • Key Devops fundamentals Revolve around the concept of continuous integration, delivery, automation & collaboration.

• Devops is more practice than technology thus many tools are applicable to different stages of developmen

• A Number of open source Devops tools are present clubbing ones of ourned makes a Dev-ops tool chain Thus product delivery more efficient.

• Different stages of sw dev:

1) Collaboration
2) Planning
3) Source control
4) Issue tracking
5) configeration management
6) continuous integration
7) Binary Repository
8) Monitoring
9) Automated testing
10) Development
11) Database.

• Companies require skilled individuals familiar with such Dev Ops tools.

• Faster Development: since, using these tools most of process is automated

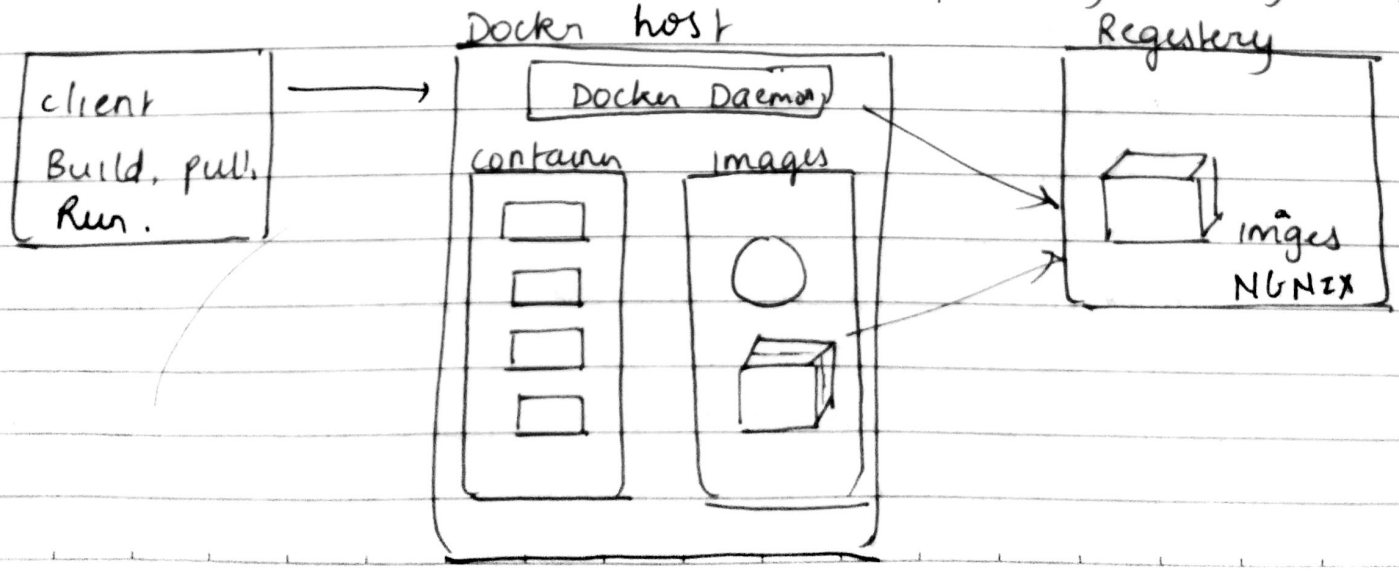
(2) Docker Architecture:

- Docker uses client server Architecture
- Docker client talks with Docker Daemon, Daemon Builds, Runs & distributes Docker containers
- Docker client & Daemon connect via REST API'S over UNIX socket or network interface.

1) Docker client:
When usn Runs any command on Docker clien CLI a Rest Request forwards it to the daemon
The following commands can be Run on CLI:
Docker Build
Docker pull
docker Run

2) Docker host: It provids an environment to ruen the applications, It contains docker daemon images containers, Networks & storage

3) Docker Registery: It contains images. Images may be Publick (visible to all) or private (for only one org acces)



(3) Containerization : is a method of packing, Running & distributing applications in containerized Environment / contained Environment. called a container. It allows isolation of application & its dependencies. Ensuring consistency & portability across different computing Environments.

- Docker is a popular platform for containerization.
- It provides tools & API's for creating deploying & managing containers. Docker containers package an application & its dependencies into standardized unit makit it easy to deploy & run applications on any systems that supports docker
- Docker uses layered file system where each layer represents the change to file system.
- Docker also provides Dockerfiles making it easy to manage container