LocAdoc
Technical Manual

Version 1.0

Prepared by: Abhi Jay Krishnan (5025448)
Kim Hyeoncheol (5026052)
Rivaldo Erawan (5026374)
Durrah Afshan (5025916)

Table of Contents

1. Introduction

1.1 Project team structure

Name	Email	UOW ID	Role
Abhi Jay Krishnan	ajk126@uowmail.edu.au	5025448	Project Manager, Design Lead
Durrah Afshan	durrahafshan@gmail.com	5025916	Analyst Lead
Rivaldo Erawan	rivaldo.erawan97@gmail.com	5026374	Development Lead
Kim Hyeoncheol	effectmix@gmail.com	5026052	QA Lead

The above team structure is implemented to clearly distinguish the area each team member is to focus on and it does not mean he/she will be the only one involved in doing that task. This is to have a pair of eyes watching each aspect of solution development.
We have decided to use a private GitHub repository for version controlling and team collaboration. Each team member will have local repository which will then be merged with the central repository. All members will be working on different part of the project at a given point in time to prevent clashes when committing to central repository. All documents and source codes will be stored in this central repository.
Each project meeting will have a project dairy summarizing the content of the meeting and what actions must be taken.

1.2 Document Conventions

Main Section Titles

- Font : Times New Roman
- Face : Bold
- Size : 16

Sub Section Titles

- Font : Times New Roman
- Face : Bold
- Size : 13

Other Text Explanations

- Font : Times New Roman
- Face : Normal
- Size : 11

1.3 Intended Audience and Reading Suggestions

This document is intended for all individuals taking part in the locAdoc application development, such as developers, project managers, users, supervisor and testers.
Readers who are interested in the overview of the project should concentrate on Part 1 which will give an introduction of the project. Part 2 will describe the overall project requirement specification. Part 3 will describe the detailed database design. Part 4 will touch on the architecture adopted while developing the system. Part 5 will touch upon component level design. Part 6 shows the user interface design developed before the construction phase. Part 7 focuses system level testing report followed further enhancement and conclusion.

1.4 Market Survey

We have done a bit of a research on the currently available applications in the market before defining the project scope. Here are the apps, their features and somethings that they are missing out.

App	Platform	Description	Security	Features	Short-comings
Secure Safe [1]	iOS, Android, Desktop (Mac)	App for online file storage & password management. Unique service as it provides double encryption, triple data storage and zero knowledge architecture, which ensures very high level of data security and privacy protection.	Automatic logout upon exiting the program Encryption is done using AES-256 and RSA 2048 Files are decryptable only by the user itself Communication between client-server is secured using https Passwords are encrypted for maximum security Support 2-factor authentication (SMS token) for premium users	Login includes password and id (email) Save texts, images, documents (scans), and videos Scans (similar like pictures and convert to pdf) Email system Import/export data	Connection independent (slow connection leads to slow access to data) Need internet connection, offline mode can only access passwords Allow multiple login attempts with wrong password/id No password recovery
Passible [2]	iOS	A password management app which is redesigned for fast and simple experience while entering website's logins and credit cards.	Automatic logout after exit from program AES-256 encryption Does not allow multiple login attempts (timeout 5 minutes for 3 consecutive wrong attempts; after that each wrong attempt will get a 5-minute timeout) Encryption with key and random string	Login using 4-digit pins Support Touch-ID (fingerprint) Save account credentials and credit card details Analyzing password strength features Support offline mode Private web browser	Does not support import/export data No password recovery
File Locker [3]	Android	App to protect your content against unauthorized reading, playing, watching, etc. The application encodes the file and makes it unreadable.	Fast way to lock huge files (like movies) by hashing just both beginning and ending of file (optional). Tracks temporary unlocked files by notification, to keep in mind you left unprotected docs. Encodes the content and file names as well.	Involves Android Media Scanner automatically after change to make file visible by other applications. Smart looking through directories for documents, locked and unlocked files. Opens documents directly from app. Notifies about forgotten unlocked files, which were left by user.	No password recovery as password is not stored anywhere. Only one-way encryption.
File Locker	Android	App that keeps all files secure	Encrypts file and save in secret location in SD Card	Unlimited files can be locked.	Encryption and decryption is

– Lock any file [4]		and private with file locker. File Lock lets you password-protect your personal files (ex: photos, videos, documents, etc) in android phones.	so that file is completely secure. Import files from SD Card / Phone Memory Password protected app entry with a numeric code or Pattern lock.	Optimized for HD tablets. Fastest lock process with multi-select feature to import hundreds of files quickly. Intuitive interface for a great experience. Hide sensitive videos and pictures. Opens documents directly from app. Password recovery option is available	not location based.
Private Photo Vault [5]	iOS, Android	A photo safe that keeps all private pictures and videos hidden behind a password.	Automatic logout after exit from program. Login using 4-digit pin/pattern lock. Import/export data Pass recovery (email)	Save photo and videos Support offline mode Private browser	Allow multiple login attempts with wrong pins

1.4.1 Conclusion

The table above shows a comparison between 5 software applications available in the market. This survey shows the different types of platform the application can run on. It provides information on the features that have been used. The most common feature in all these app is that they use encrypted password for securing files such as images, videos, documents, etc. Major drawback noted from the table is the lack of password recovery and the encryption/decryption of files is not location based.

1.5 Project Scope

1.5.1 Project purpose

This project aim to provide user a way to store confidential documents in mobile devices and access it only in the area he/she find it is safe. By including two factor protections, one being password (what the user knows) and second being the location (where the user is currently), we will be able to provide a better solution compared to the applications currently in the market (based on market survey).
These are few ways a document in a mobile device may be compromised: -

- The documents stored in mobile device may end up in the wrong hands if the device itself is stolen.
- The user may lend the device to someone who intern may wish to gain access to these documents.
- The documents may be accessed remotely by penetrating device through network.

Our solution aims to provide a secure vault for document storage so the it does not get into wrong hands even if the device is compromised. The solution also provides a secure backup cloud storage with double layer encryption one by the app itself and one by Amazon server.

1.5.2 Target Users

There are several applications for this solution in the market, here are few of them: -

1. Employees working in defense industry may have to handle highly secretive documents that should not be taken of the secure premises.
2. Most of the large firms these days have documents that are their intellectual property and wish to keep them from getting into wrong hands.
3. General public may want to store their personal information and keep it within their safe zone such as their home.
4. Governmental authorities may wish to keep their confidential documents within the country or within the restricted area.
5. A hospital may wish to keep the patient's document within the campus but at the same time giving staff the freedom to view it while moving around.
6. A school may want to let the authorized staff to review an exam paper on the move while keeping the document within the restricted zone.

1.5.3 Proposed features

The proposed solution is an android based app with following features: -

1.5.3.1 Functional Features

1.5.3.1.1 PDF Viewer

A PDF viewer for user to view his documents in the vault. The pdf viewer will only be accessible after the user has been authenticated and if the user is within the radius of the location stored in the database. The pdf viewer will close when the user moves out of this zone. The file that the user wishes to see will be the only one that will be decrypted. The rest will remain as cipher text even when the user is in authorized area. This pdf viewer will help the user to be more productive by having the ability to access sensitive document while moving within the secure location.

1.5.3.1.2 Deleting files

The user has the option to delete the files that are not needed, and these files will also be deleted from the backup.

1.5.3.1.3 Setting preferred locational radius

Once the user adds a new file he can set the radius he wishes with small radius being more secure and larger radius being more convenient. The files will be grouped based on the location and the user can choose the area if there is an overlap.

1.5.3.1.4 Less clustered interface

The user will be only able to view the files that was saved to a location making file accessing, pleasant and less tedious.

1.5.3.1.5 Import files

The user will be able to import a new file from the local file directory and secure it through encryption. The original file can be deleted to prevent adversary from viewing it.

1.5.3.1.6 Secure cloud storage

The data will be safely stored in the central database with additional layer of encryption by the cloud infrastructure provider.

1.5.3.2 Non-Functional features

1.5.3.2.1 Changing Password

The user after logging into the app can change the password by providing the old password. The password will be saved to the central database. The password forms a part of the key used to secure files and tables in the database. Once the password is changed the database will be secured using new password when the user logout.

1.5.3.2.2 Password recovery

If the user forgets the password, he will receive a password recovery code through his email or notification. Upon entering the code, he will be redirected to the changing password procedure.

1.5.3.2.3 Dealing with stolen device

The user can migrate to a new device using the backup data and change password so that the data in the old phone can never be decrypted as the sever will never authenticate the user even if he knows the old password.

1.5.3.2.4 Dealing with missing folder

The user will be promoted to download the backup folder and will be able to access files as usual or the user can choose to create a new one.

1.5.3.2.5 Protection against GPS spoofing

The application checks if the mock GPS settings have been turned on every 30sec, If turned on it will logout to prevent GPS spoofing.

1.5.3.2.6 Secure cryptographic algorithms

Files are encrypted using secure cryptographic algorithm such as Advanced Encryption standard (AES128). The file will not have the same name as the original file after encryption, the original name will be stored in the database along with the new name. The password will be hashed using strong hash functions such as the SHA256. The central database will have an additional layer of encryption provided by the cloud service provider (Amazon Web Service).

1.5.3.2.7 Migration

The user may wish to move to a new device, and the app allows user to download the backup file from the cloud and continue using as usual. The user credentials are stored in a central database to ease migration.

1.5.3.2.8 Backup

The user can back up his data to cloud (AWS S3) to retrieve it later when the disaster strikes or for migration. The system will track the changes and only save the newly added file. The data stored in cloud is send over after encryption.

1.5.3.2.9 Speed

A temporary SQLite database will be created to store the user credentials locally to costs (time) that is incurred through constant communication with the central database. This database will be dropped when the user logout.
These are the key features that will be included in the application. Further enhancements such as support for more file types will be added if these basic requirements are met.

1.6 Programming development methodologies

1.6.1 Waterfall

This is a legacy model for software development projects. The development cycle is linear and not capable of supporting challenges faced by modern software development domain such as changing user requirements and adaptations. There is high amount of risk and uncertainty using [6].

Diagram of Waterfall-model:

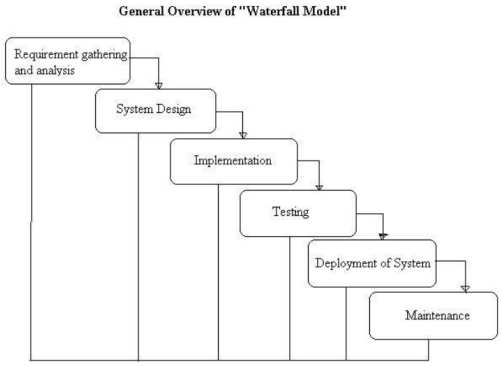


Figure 1: water fall model

1.6.2 Prototyping model

Under this model the requirements are gathered from user and a prototype is build based on it for evaluation before refining requirements. This iteration repeats until final design is confirmed before final construction of the product itself. This model may be inappropriate for this project as there are no rapid changes that may need iterative prototyping. Prototyping also require active involvement of end user which is not possible for this project [7].

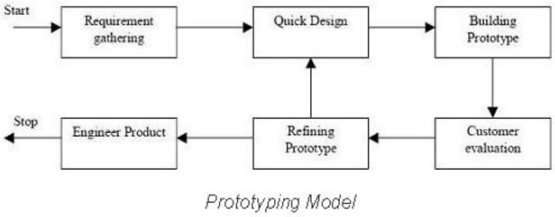


Figure 2:prototyping model

1.6.3 Agile model

Agile methodologies include Extreme Programming (XP), Scrum, Crystal, Dynamic Systems Development Method (DSDM), Lean Development, and Feature-Driven Development (FDD). They all are based on a common principle of iteration and continues feedback that it provides to successively refine and deliver a software system. This method also requires active involvement of end-users and may track off the user is not sure about the outcome they want. There is also lack of emphasis on necessary designing and documentation. Hence, agile model may not suitable for this project [8].

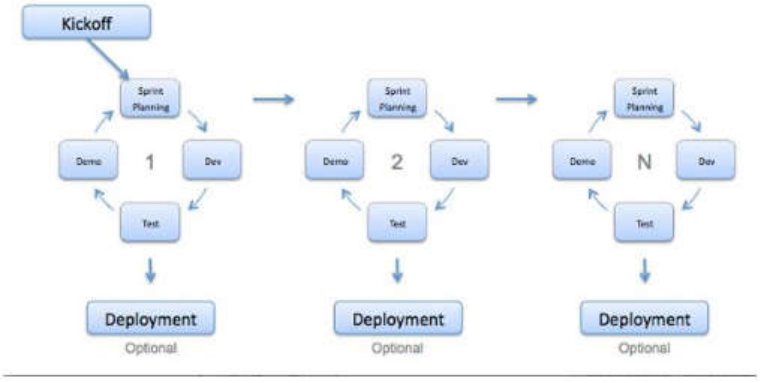


Figure 3:agile model

1.6.4 Rational Unified processing

Rational unified processing model provide a disciplined approach to assigning task and responsibilities within the development organization. The development is divided into 4 sections: -
Inception – The project scope is defined along with requirements gathering and Risk analysis.
Elaboration – Coming up with detailed design of the system.
Construction – The actual development of the system.
Transition – Testing and deployment of the system.
There may be more than one iterations for each of the above stages and the also provide a flexibility to make changes to requirement deep into the development cycle [9].

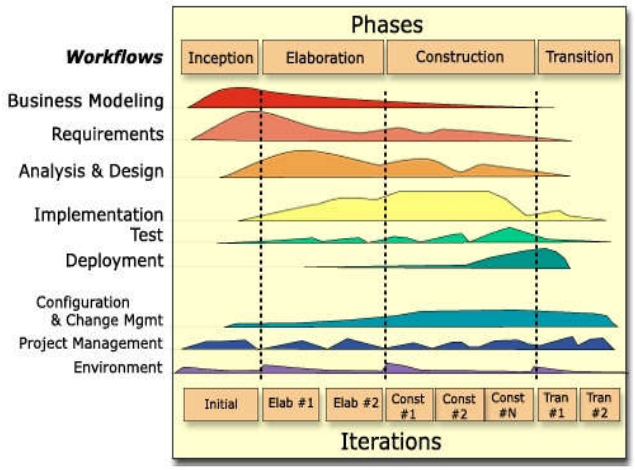


Figure 4: Rational Unified Processing

After considering above different models we have decided RUP is the most suitable for this project as it allow us to decide the business case, need less involvement of end user and gives flexibility to make changes during later part of the development process.

1.7 Table 1: Proposed application development language

Language	IDE	Developer	Description
Java	Android SDK, Eclipse + Android ADT	Android	Java is used to develop android native app and java is the official language for development android application using android studio.
C++	QT	The QT company	An android application can be developed in C++ using Qt libraries. It archives the same speed as natively developed app, but the app package size is significantly large. Using C++ will also require more development time.
Kotlin	Android SDK	JetBrains and opensource developers	Kotlin relatively new programming language that runs on java virtual machine.
HTML JavaScript	Apache Cordava	Adobe Systems	HTML and Java script can be used to develop web apps that run in android OS and cordava.js help in connecting with systems features such as camera, accelerometer and GPS. This method normally produces apps which are slower that natively developed apps.

There are other methods used to develop an android application, but none have the same flexibility and speed as the natively developed apps using java. Hence, we have considered in using java and android SDK to develop this solution. Android SDK also uses XML to define the user interface elements.

1.8 Table 2: Platform comparison

	Desktop platform	**Mobile devices**
Portability	A Desktop is restricted to a local area while the while the laptop is normally less mobile than a mobile phone.	Due the small form factor, mobile devices such as smart phones and tablets are more portable compared to desktop platforms.
Connectivity	A desktop environment normally has built in WIFI and Ethernet connectivity.	A mobile device comes with a WIFI connectivity but lacs Ethernet connectivity. One the positive side mobile devices comes with cellular data connectivity.
Operating System	Desktop environment may have large verity of hardware and there are compatibility issues when OS tries to interact with hardware.	Mobile OS is mostly Android, IOS and Windows Mobile normally have fixed hardware.
Cost	Desktops and laptops have larger capacity and better processing capabilities compared to mobile devises hence they are generally more expensive.	Mobile devices are generally less expensive.

From the above comparison, we can see that mobile devises have better portability and low-cost factor which makes it an ideal platform for user to reference document on the go. The user will also find mobile devices a cheaper option compared to desktop.

1.9 Table 3: Mobile OS Comparison

	Android	IOS	Windows
Downloadable Apps and Apps Market	- **Google Play** - Nowadays, Google Play have dominated in the application market due to convenience of App approval process - Google Play is well integrated with Google's Applications such as Gmail, **Google Maps**, Google Drive and so on.	- **App Store** - In the Initial Period of App Market, App Store dominated in the application market.	- **Window Store** - Most of popular Applications are already in the market. But the update speed is slower than Android and IOS. -It is available to unify the application between Desktop such as Windows 10 and Mobile Devices.
OS Platform's Features	- **Samsung Flow**: version is still Beta version - Hangout - Google Fit - Have Wide Range of available devices (Mobile	- Devices based on IOS can be synchronized between Apple Devices. - iMessage - Apple Health	**Continuum**: plug mobile device into the Monitor with Keyboard and Mouse then get the interface

	phone, tablets and wearable devices such as smart watches). Not only Samsung but also google and so on.	- IOS is only limited into Apple own devices such as MAC. - Most of the devices based on IOS have **expensive cost** among 3 OS.	such as Desktop. Thus use the phone like PC. - Skype - Microsoft Health
Virtual Assistance	- **Google Now** - Google Now opens its API to developers who can use it for operating or referencing other apps - **Picture Recognition**: It is available to offer information by input screenshot or picture	- **Siri** - Siri has **accurate understanding** compared as Google now - But the information area by Siri is limited For example, Playing music, setting timer or alarm and so on.	- **Cortana** - It is the latest virtual Assistance in Windows at 2015 - Bing Search, Music Recognition - Still need to be uploaded about features compared as the Android and Siri
Security	- Android is based on **Open Source Code** - Easy to submit the Application into Google Play with cheap submission fee compared as the App Store - it has reinforced security in Google Play store after stage fright attack in 2015 - Direct booting which allows application to begin with the lower layer in mobile devices - File Encryption which allows protecting the personal data in devices. - Because the wide range of devices is available in Android OS, compared as the other OS platforms, it is vulnerable from malicious attack and Not whole Android OS platform is updated at the same time. -	- requires only use Apple's own devices. - App Store requires signature and checking from Apple Before submit Application into App store. - using secure encrypted channel when upload/update apps - Like Android, IOS is one of the OS that a lot of users use in nowadays. Thus, There are probabilities attacked from malicious third party.	- Like IOS, Window store has strict app submission process - Device Encryption based on the local contents - Easy to integrate apps between PC and mobile devices - **Microsoft passport**: strong authentication process to access to resources. - **Device Guard**: protect data from malicious programs - **Microsoft Enterprise Mobility**: focus on Security Session with ATA (Advanced Threat Analytics) - One of the Problems in Windows store is small market compared as Android and IOS. Thus, it has a probability to being attacked in the future and also lack of features in the markets.
Biometric Security	- **Fingerprint sensor technology** after IOS - lyrics pattern recognition technology from Galaxy Note 7 in Samsung	- IOS is the **FIRST** OS which released **Fingerprint sensor technology** in the devices.	- **Windows Hello**: Authentication to sign in Windows 10 devices securely. (Surface Pro 4,

			Surface Book, most PCs) with fingerprint readers or Face recognition always work.

After considering the above comparison between different OS platforms we have decided to go ahead Android as it has the largest market share [10] and cross platform option does not achieve the same performance level as natively build apps [11].

1.10 Table 4: Comparison between Relational and Non-Relational Database

	Relational Database	Non-Relational Database
DB Schema [12]	➢ Structured Query Language ➢ Stores items in tables to minimize duplicate values ➢ Fixed schema ➢ Example: MySQL, Oracle	➢ NoSQL database ➢ Unstructured data ➢ No fixed schema ➢ Document-oriented ➢ Example: MongoDB, HBase
Scalability [13]	➢ The tables relationship makes scaling more resource-intensive	➢ Stores each item as single document for high scalability ➢ Includes sharing or partitioning ➢ Uses elastic scalability
Flexibility [14]	➢ Provides flexible structure to meet changing requirements and increasing amounts of data ➢ This model permits changes to a database structure to be implemented easily without impacting the data or the rest of the database ➢ In reality, growth and change are limited by the relational database management system and physical computing hardware	➢ Flexible data model ➢ Easy to store and combine data of any structure ➢ Defining types of data in advance is not required ➢ Allows dynamic modification of schema without performance impact
Cost [15]	➢ Rely on expensive proprietary servers and storage systems ➢ Licenses for this system can be quite expensive	➢ Uses clusters of cheap commodity servers ➢ Databases are open source and therefore free
Consistency [12]	➢ Tight consistency	➢ Eventual consistency rather than ACID property
Transaction [16]	➢ Transaction with ACID property (Atomicity, Consistency, Isolation & Durability)	➢ Does not support transactions
Performance [14]	➢ High performance speed	➢ High read and write performance ➢ Unlimited growth with higher throughput ➢ Lower latency than relational database ➢ Faster development life cycles for developers
Reliability [17]	➢ Changes committed in a transaction are stored and available in the database even if there is power failure or the database goes offline suddenly.	➢ Automatic back up of data in separate facilities (E.g. In DynamoDB)
Reporting tools [17]	➢ Wide array of reporting tools helps to prove application's validity	➢ Lack of reporting tools for analysis and performance testing

Security [18]	➢ By splitting data into tables, certain tables can be made confidential ➢ The system can then limit access only to those tables whose records they are authorized to view	➢ Has weak password storage ➢ Lack of encryption support for the data files ➢ Weak authentication both between client and the servers ➢ Vulnerability to SQL injection ➢ Denial of service attacks.

After considering the above factors we have decided to use non-rational database (NoSQL) as a central database as it is scalable and prevent a lot of administrative complications. DynamoDB by Amazon Web Service is such database and well established. AWS also provide a library to interact with the database through the Android application.

1.11 Table 6: Comparison between cloud service platforms

	Amazon Web Services [19]	**Microsoft Azure** [20]	**Google Cloud Platform** [21]
Years in operation	11 years	6 years	5 years
Managed By	Amazon	Microsoft	Google
Free Tier features	• AWS Lambda – 1 Million Request/Month • Amazon S3 – 5GB of data storage • Amazon RDS - 750 Hours per month of db. T2.micro database usage (applicable DB engines) 20 GB of General Purpose (SSD) database storage 20 GB of storage for database backups and DB Snapshots. • DynamoDB (NoSQL) storage : 25 GB AND more…	• 14 virtual machines for 1 month • 40 SQL database for a month • 8TB of storage for a month • $200 credit • Up to 10 web and mobile apps.	• 28 instance hours per day. • 5 GB Cloud Storage. • Shared mem cache. • 1000 search operations per day, 10 MB search indexing. • 100 emails per day.
DDOS protection	Yes	No	No
Mobile App Software Development Toolkit	Yes	Yes	No
Mobile App Testing Service	Yes	Yes	Yes
Mobile App Development Services	Yes	Yes	Yes

NoSQL database Systems	Yes	Yes	Yes

We choose to go ahead with Amazon Web Services as it provides a lot of features in free tier and its highly scalable. Amazon is also the is provide a mobile API along with backup for database. Security wise Amazon shield provide protection against DDOS attacks. Aws also has very large customer profile and many large successful firms trust their services [22].
After considering all the above comparisons in the next section we are going to talk about the proposed system architecture.

1.12 System Architecture

The architecture adopted by the application will be a new and sophisticated *cloud-client architecture*. This architecture works by making use of cloud computing resources to manage user's data and credentials as well as authenticating user. The proposed application will be using AWS (Amazon Web Services) as its cloud service provider and AWS Mobile SDK [23] for the development of the app. The main advantage this architecture has is that we don't need to manage the resources in the cloud including the security of it.

Overview of the architecture:

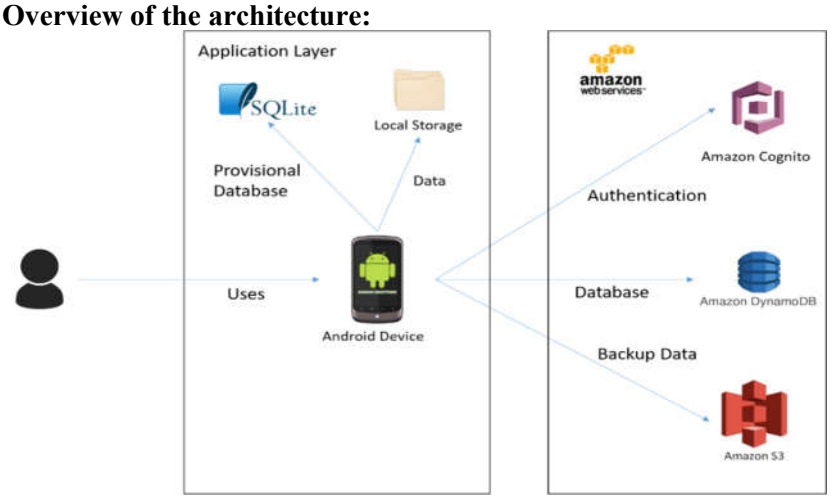


Figure 5: Client cloud architecture

Components of the architecture:

Amazon Cognito

Amazon cognito makes it easy to add user sign-up and sign-in and manage permissions for mobile apps. We can create our own user directory within Amazon Cognito, or we can authenticate users through social identity providers such as Facebook, Twitter, or Amazon; with SAML identity solutions; or by using our own identity system. With Amazon Cognito, we can focus on creating great app experiences instead of worrying about building, securing, and scaling a solution to handle user management and authentication [24].

Amazon DynamoDB

Amazon DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale. It is a fully managed cloud database and supports both document and key-value store models. Its flexible data model, reliable performance, and automatic scaling of throughput capacity, makes it a great fit for mobile, web, gaming, ad tech, IoT, and many other applications [25].

Amazon S3

Amazon Simple Storage Service (Amazon S3) makes it simple and practical to collect, store, and analyze data - regardless of format – all at massive scale. S3 is object storage built to store and retrieve any amount of data from anywhere – web sites and mobile apps, corporate applications, and data from

IoT sensors or devices. It is designed to deliver 99.999999999% durability, and has many customers each storing billion of object and exabytes of data. You can use it for media storage and distribution, as the "data lake" for big data analytics, as a backup target, and as the storage tier for server-less computing applications. It is ideal for capturing data like mobile device photos and videos, mobile and other device backups, machine backups, machine-generated log files, IoT sensor streams, and high-resolution images, and making it available for machine learning to other AWS services and third-party applications for analysis, trending, visualization, and other processing [26].

SQLite

SQLite is an embedded SQL database engine that can act as a database in android local system. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file. The database file format is cross-platform - you can freely copy a database between 32-bit and 64-bit systems or between big-endian and little-endian architectures. These features make SQLite a popular choice as an Application File Format.

Other architecture that have been considered are standalone, client-server, and web-based. Standalone is simpler and more robust, it mostly doesn't need any internet connection to use all the features in the app, but the number of feature it can have is limited. Client-server opens more possibilities for the application to have more interesting features, however there would be a problem of securing the communication, the cost to set up the server, and the maintenance of the server itself. All of this can be applied to web-based architecture too since it uses a server. Additionally, web-based is less desirable due to its performance issue because there is no local storage for the application, therefore the performance is very dependent on network connection to access the data.

Design principles of the application:

Modularity: separating the functionality of a program into independent, interchangeable modules, such that each contains everything necessary to execute only one aspect of the desired functionality.

High Cohesion: each module has functions and elements that are strongly related, only to fulfil one particular purpose or task

Low Coupling: modules are loosely coupled and independent so that a change in one module do not affect the other modules

Standardization: implementation will conform to standard that has been established and agreed by different parties, this is crucial for things like security

The proposed application will have the following quality:

Extensibility

The proposed application can add additional functionality without changing or damaging much of the current system. New data types can be added if it is supported by the android.

Maintainability

Following the design principles of high cohesion and low coupling, small modifications will not be a problem. Changing one module will not affect other modules significantly.

Performance

The response time will be in acceptable manner even with the huge amount of data that are processed. Efficient encryption algorithm is used as well as other processing algorithm

Usability

Adapting KISS (Keep It Simple Stupid) principle in designing interfaces will give user easier times in learning and figuring out the proposed application. It lets user to take less time to perform a certain task.

Compatibility

The proposed application will be able to run in various type of android devices as well as different version of android.

Security

Data are kept safe by encryption and login is required to have access. Security measures like protection against SQL injection or encryption algorithm will follow standard.

1.13 Table 7: Risk List

Risks	Actions
Sudden growth in requirement as the project progress.	Readjust project time line to accommodate the changes.
Team members unable to contribute to the project due to health or other valid reasons.	Distribute workload among remaining members. Making use of GitHub repository will help the team member who is not able to come for meeting to contribute remotely. Also using other means such as skype calls and TeamViewer.
Estimation and scheduling of development time is done on initial stage and there may be glitches along the way that will set back the project timeline.	Project plan can be revisited and adjusted to fit within the given deadline.
Usage of new and ever-changing products in the market will lead to bugs in software that is being developed.	Proper research and training is critical when using a new tools, techniques, protocol or systems.
System performance may be compromised when having substantial number of features.	Usage of good programming practices and threading.
Selecting an unsuitable design architecture	Doing a good research and developing a good picture of the end goal based on past experiences (market survey).
Loosing support for API (Application programming interface) used while making the solution.	Usage of good object oriented or modular programming architecture will help in transiting from one API to another with minimal work.
Amazon service outage	There may be a need to migrate to new service provider who provide similar services like Azure or Google cloud platform.

2. Requirement Specification

2.1 Overall Description

2.1.1 Product Perspective

LocAdoc is a new, self-contained application running on android platform written in Java. It is intended in providing the user a safe place to store his/her pdf files and view it at an area that they find secured. The system uses a client cloud architecture where the client is the application running on a mobile device and it makes use of cloud services provided by amazon web services.

2.1.2 Product Features

The features of this system are:

2.1.3 PDF Viewer

A PDF viewer for user to view his documents in the vault. The pdf viewer will only be accessible after the user has been authenticated and if the user is within the radius of the location stored in the database. The pdf viewer will close when the user moves out of this zone. The file that the user wishes to see will be the only one that will be decrypted. The rest will remain as cipher text even when the user is in authorised area. This pdf viewer will help the user to be more productive by having the ability to access sensitive document while moving within the secure location.

2.1.4 Deleting files

The user has the option to delete the files that are not needed, and these files will also be deleted from the backup.

2.1.5 Setting preferred locational radius

Once the user adds a new file he can set the radius he wishes with small radius being more secure and larger radius being more convenient. The files will be grouped based on the location and the user can choose the area if there is an overlap.

2.1.6 Less clustered interface

The user will be only able to view the files that was saved to a location making file accessing, pleasant and less tedious.

2.1.7 Import files

The user will be able to import a new file from the local file directory and secure it through encryption. The original file can be deleted to prevent adversary from viewing it.

2.1.8 Secure cloud storage

The data will be safely stored in the central database with additional layer of encryption by the cloud infrastructure provider.
These are the key features that will be included in the application. Further enhancements such as support for more file types will be added if these basic requirements are met.

2.2 User Classes and Characteristics

2.2.1 Physical Actors

Mobile users: The user who uses the system and make use of the services provided by the application.
System Actors:
DynamoDB: A NoSQL database service provided by Amazon web services(AWS) and this service will be used to create a central application database.

AWS Cognito: A user password authentication service provided by Amazon and will be used to authenticate user based on a central user pool.
AWS S3: A central file storage facility provided by the AWS will be used to backup user data.

2.3 Operating Environment

This system only operates in Android Operating System.

2.4 Design and Implementation Constraints

The main constraint of this program is the support for several file formats, as each file format will require dedicated viewer. For now, we are sticking to PDF format and we will consider other formats for future releases.

2.5 User Documentation

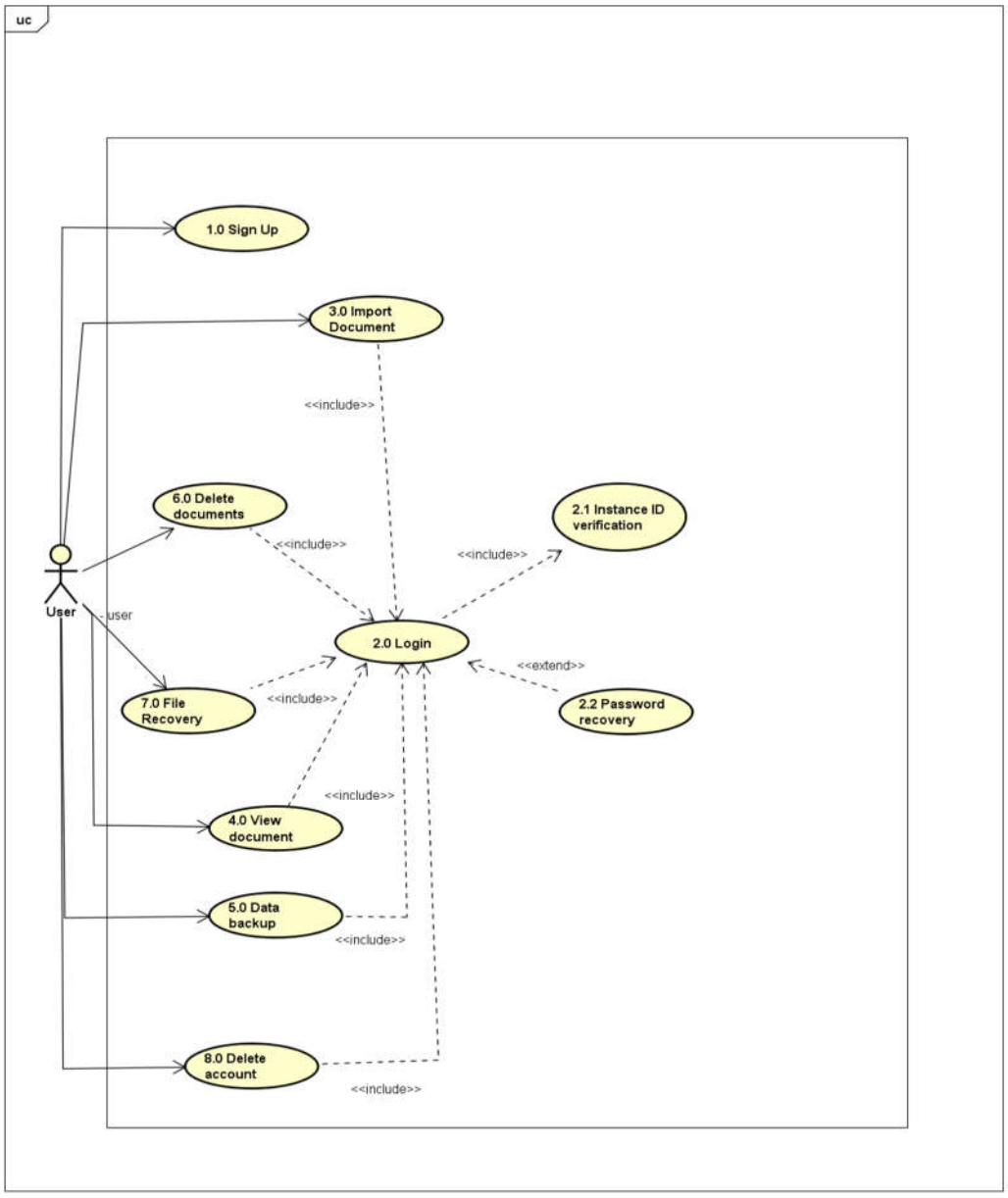
The user can use the help menu in the system to understand the interfaces more.

2.6 Assumptions and Dependencies

- The user is liable in keeping the copy documents outside the system boundary secured.
- The user is liable in keeping his login credentials secured

2.7 System Features

2.7.1 Main Use Case Diagram

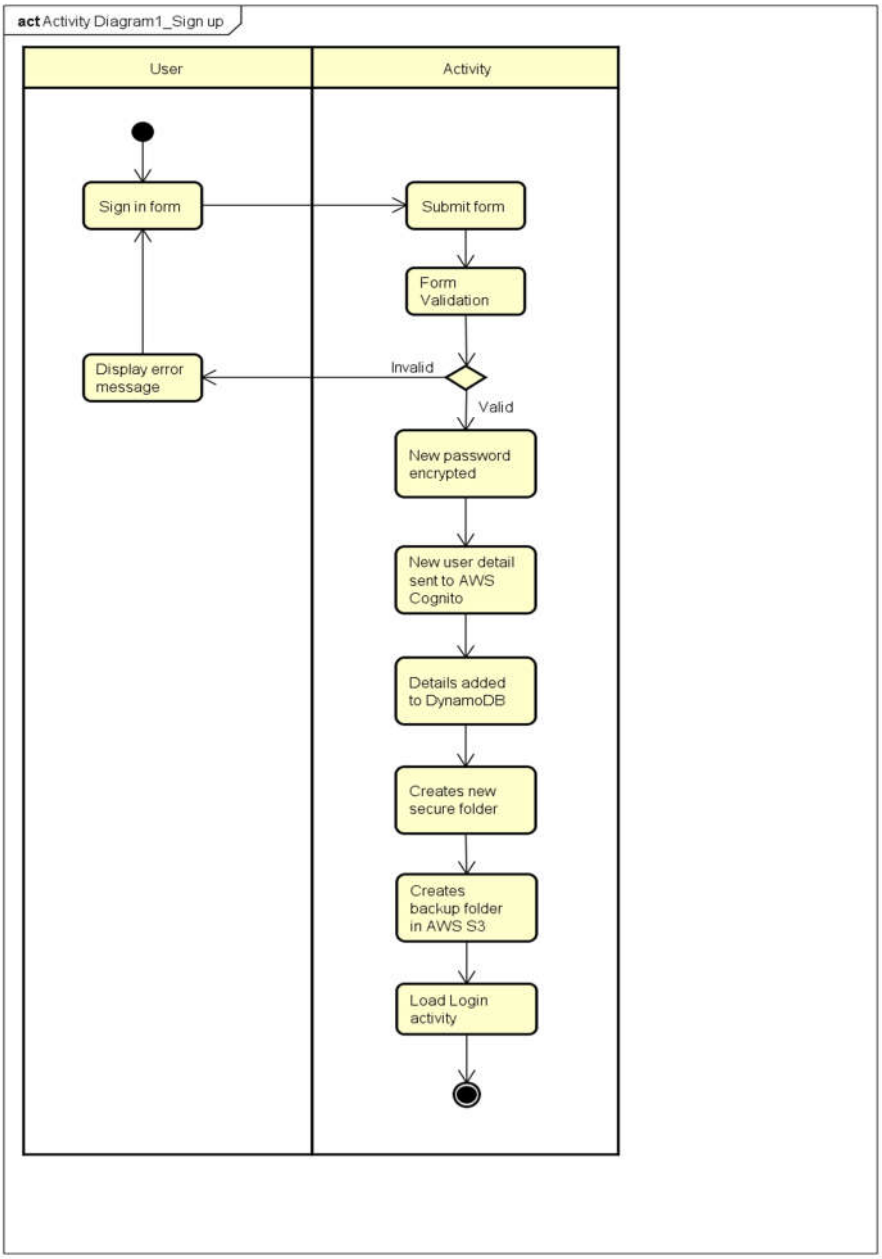


2.7.2 Use Case 1.0: Sign up

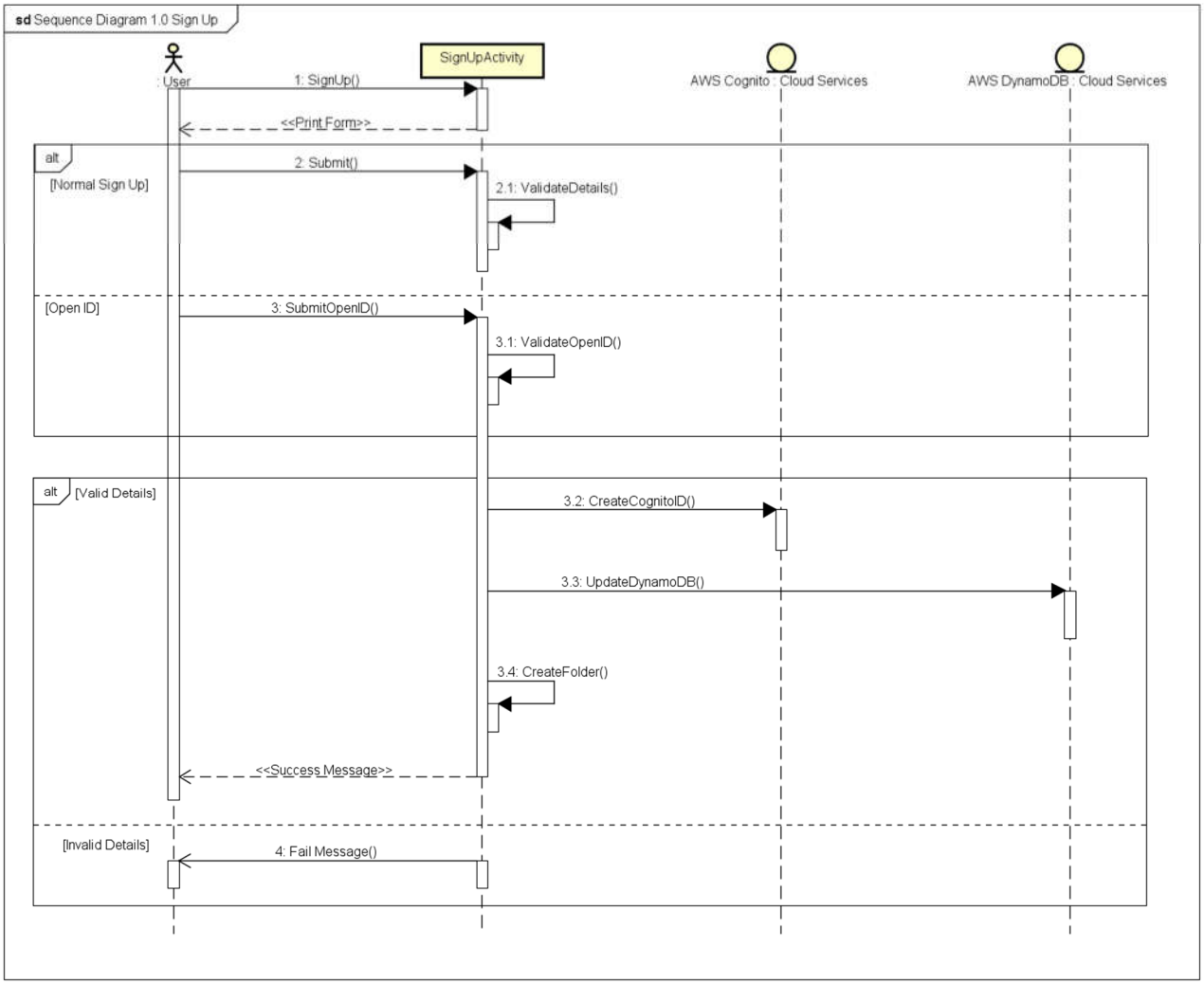
Use Case Textual Description	
UC-ID	1.0
Name	Sign up
Description	To allow new user to create a new account after he download the application for the first time.
Actor(s)	User who is interacting with the application, Signup activity, AWS Cognito, AWS DynamoDB, AWS S3.

Precondition	A user has installed the application for the first time and wish to create a new account.
Main Scenario	Step 1: System prints out sign up form with fields asking for user's particulars. Step 2: User submits the form and the system does form validation. Alternate Step 2: If the data is invalid, an error message is displayed and the user is sent back to the form to reenter the data correctly. Step 3: The user details are encrypted using the newly entered password. Step 4: The system updates the AWS Cognito server. Step 5: The system updates the AWS Dynamo DB. Step 6: A new secure folder is created Step 7: A backup is created in the AWS S3. Step 8: The user is directed to the login screen.

2.7.3 Activity Diagram 1.0 Sign Up



2.7.4 Sequence Diagram 1.0 Sign Up

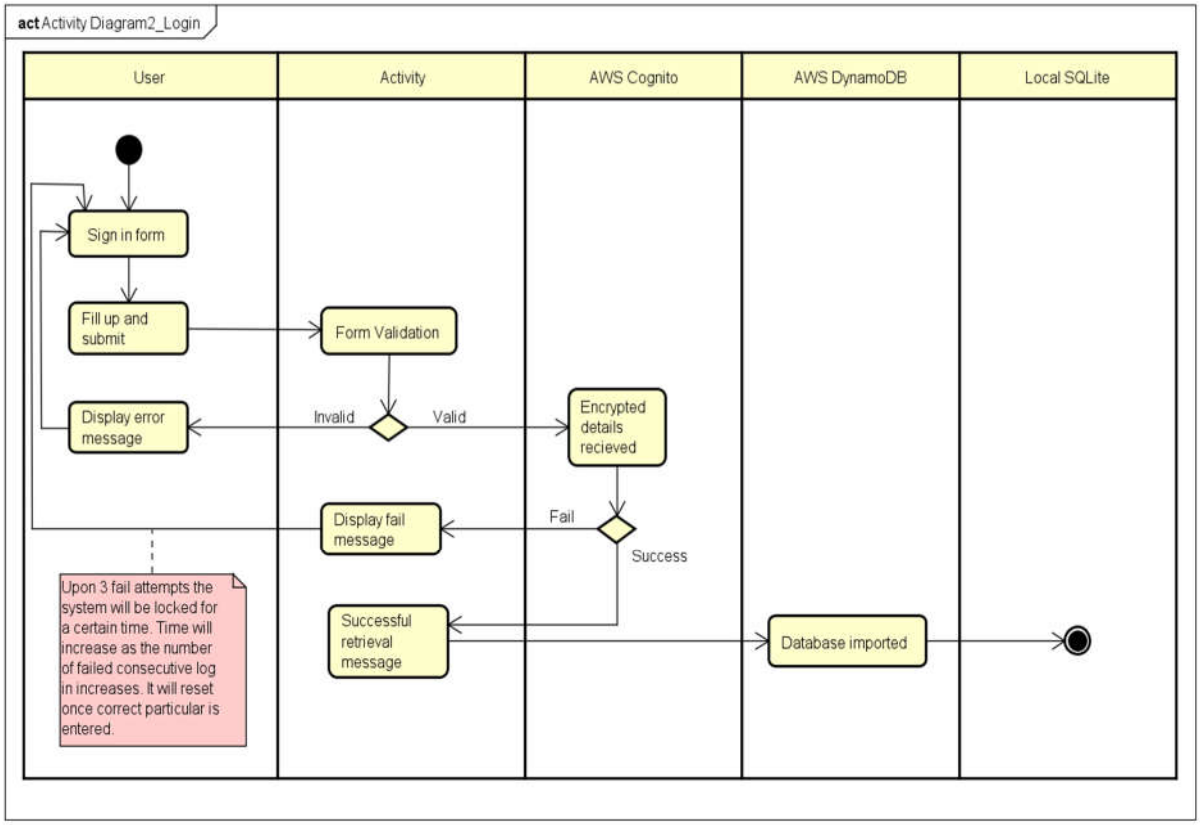


2.7.5 Use Case 2.0: Login

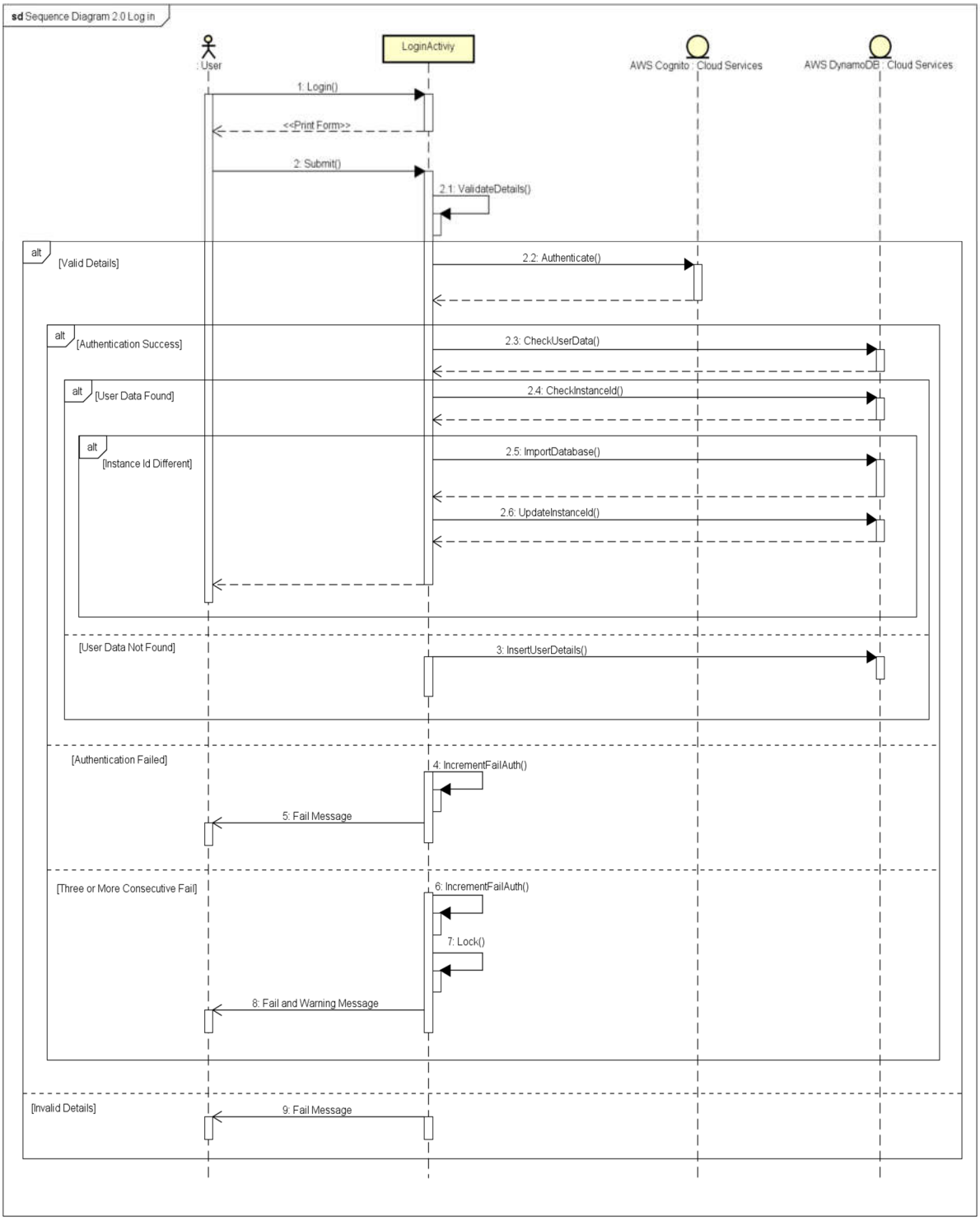
Use Case Textual Description	
UC-ID	2.0
Name	Login
Description	To allow existing user to enter and use the if the user have signed up or create and account in the system
Actor(s)	User who is interacting with the application, System, AWS Cognito, AWS Dynamo DB, SQLite
Precondition	A user has an active account in the system (sign up) and wish to use the system
	Step 1: Activity prints out sign in form with fields asking for user's account and password.

Main Scenario	Step 2: User fills in the form and submits it, then the activity does form validation. Step 3: If the data is valid the activity hands over the encrypted details to the AWS Cognito. Alternate Step 3: If the data is invalid, an error message is displayed and the user is sent back to the form to reenter the particular correctly. Step 4: The activity receives a success reply from AWS Cognito, the user will enter the system. Alternate 1 of Step 4: The activity receives a fail reply AWS Cognito and the user is sent back to the form to reenter the particular correctly. Alternate 2 of Step 4: The activity consecutively receives a fail reply from AWS Cognito 3 times, the system will be locked for a certain period of time. After that every consecutive fail will lock the system, and the time will increase as the number of failed consecutive log in increases. It resets when a correct particular is entered Step 5: Import user's data from AWS Dynamo DB to local SQLite database.

2.7.6 Activity Diagram 2.0 Login



2.7.7 Sequence Diagram 2.0 Log in

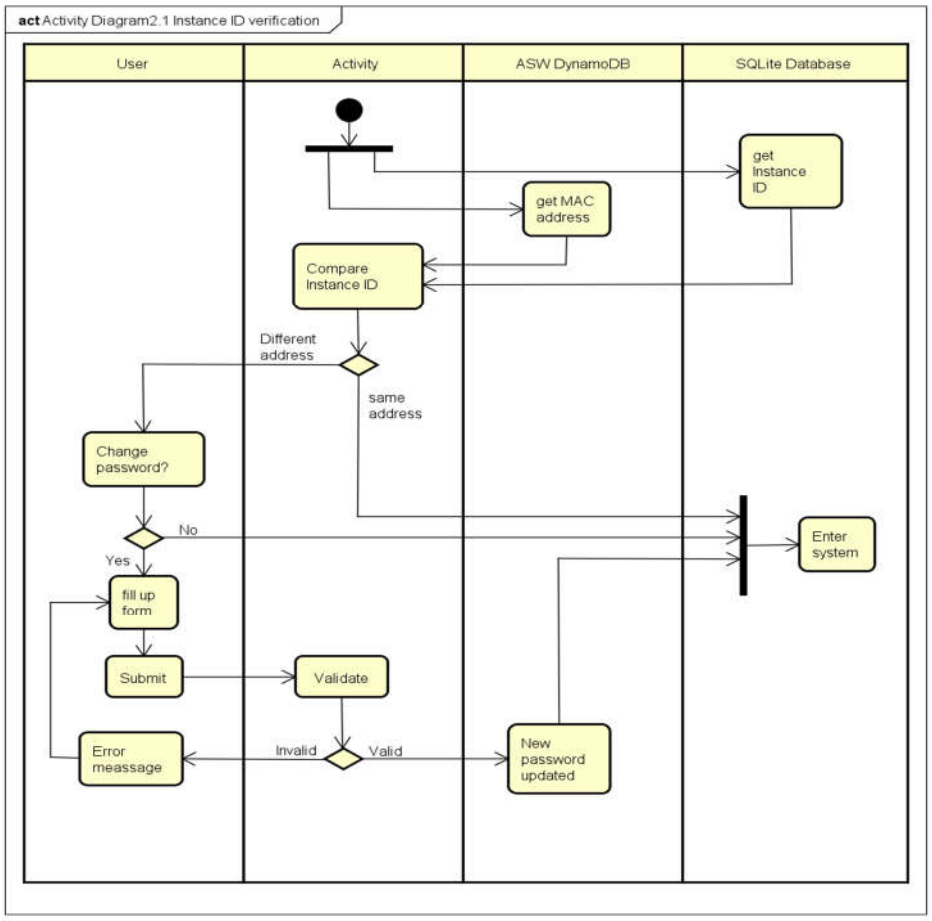


2.7.8 Use Case 2.1: Index ID Verification

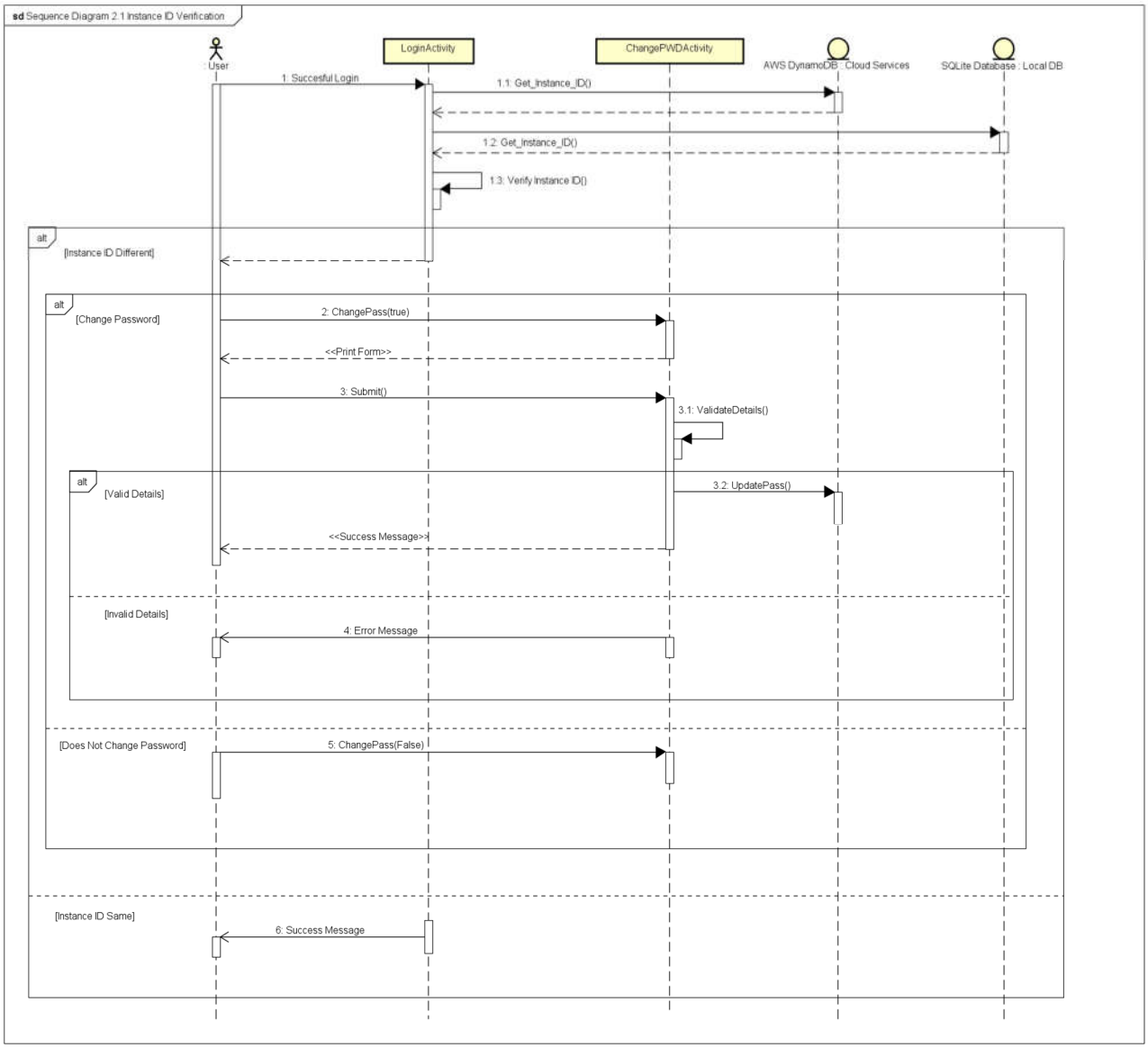
Use Case Textual Description	
UC-ID	2.1
Name	Index ID Verification

Description	Index ID is generated on installation is used to verify if the user is using another device and also used to makes sure that the all the devises that are not currently not used stays offline.
Actor(s)	User who is interacting with the application, Activity, AWS Dynamo DB, Device.
Precondition	An instance ID is generated and stored in the local database and AWS DynamoDB database on first successful login.
Main Scenario	Step 1: Activity compare the Instance ID in the AWS Dynamo DB with the Instance ID stored in the device. Step 2: If the Instance ID is different, the activity will display the notification message and give the user an option to change the password Alternate Step 2: If the Instance ID is same, the user will enter the system Step 3: If the user decides to change password, a form will be printed Alternate Step 3: If the user decides to not change the password, go to Step 6 Step 4: User fills in the form and submits it, then the activity does form validation. Step 5: If the data is valid the activity will update the new password to the AWS Dynamo DB Alternate Step 5: If the data is invalid, an error message is displayed and the user is sent back to the form to reenter the new password correctly. Step 6: Enter the system.

2.7.9 Activity Diagram 2.1 Instance ID Verification



2.7.10 Sequence Diagram 2.1 Instance ID Verification

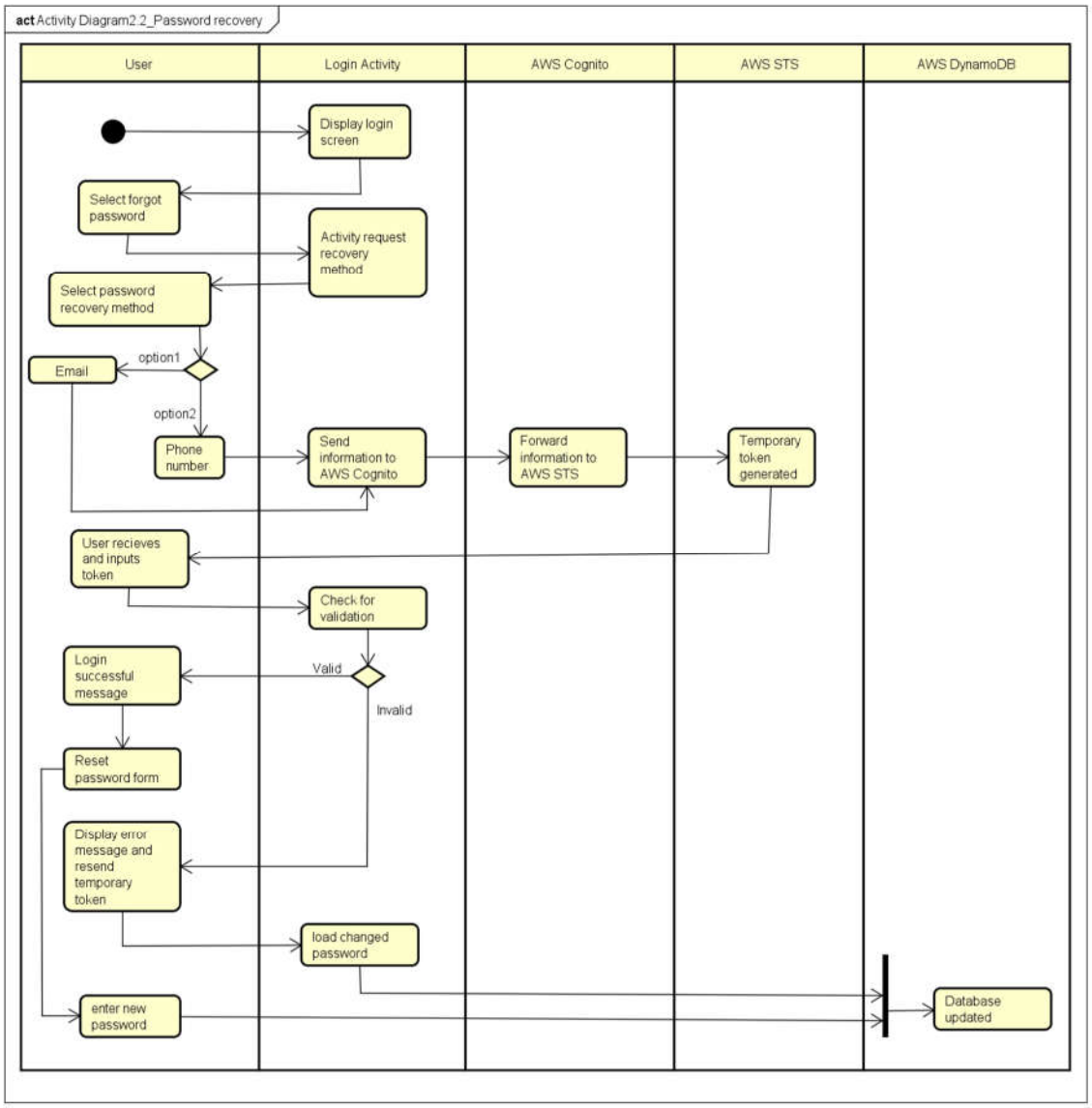


2.7.11 Use Case 2.2 Password Recovery

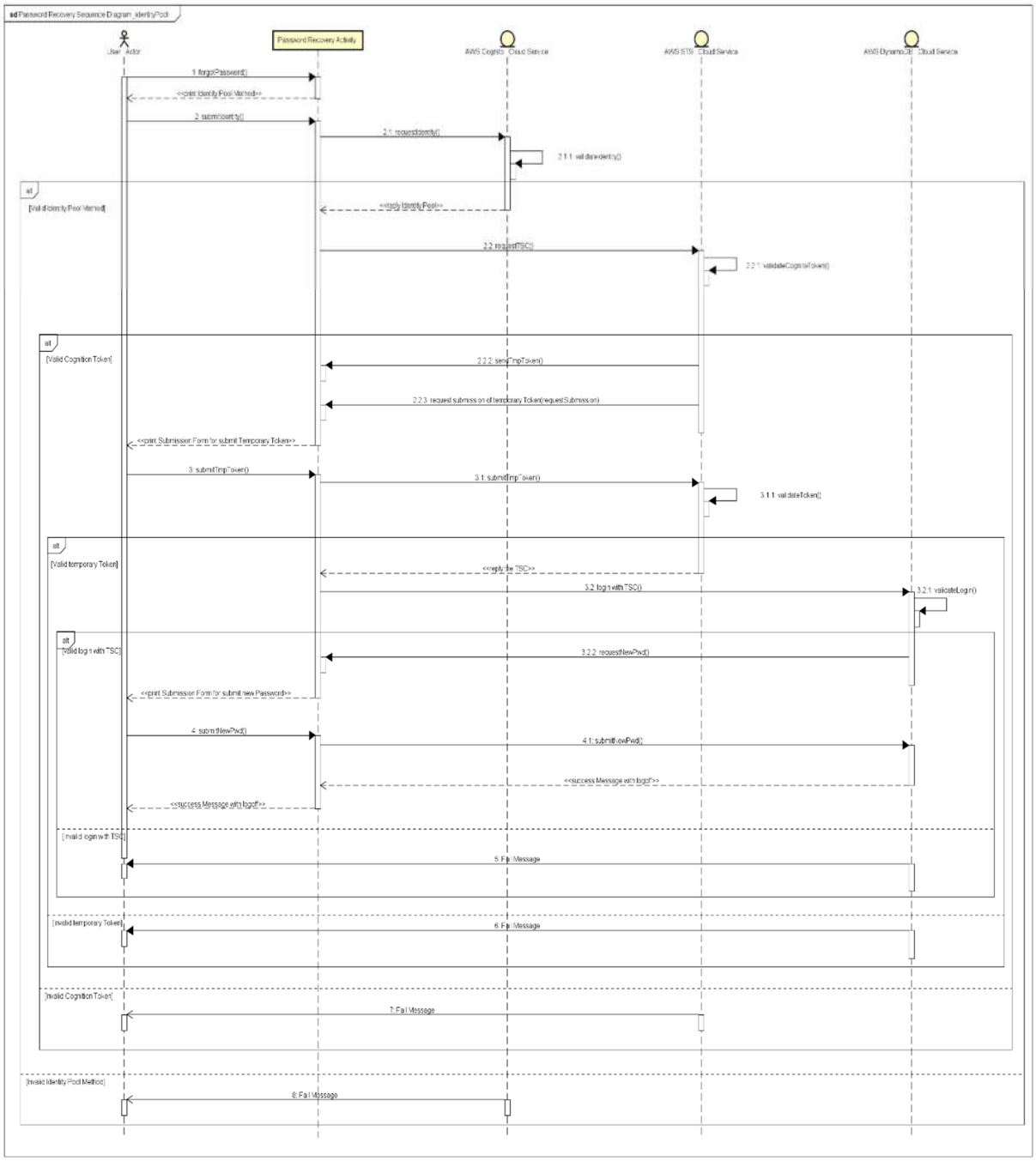
Use Case Textual Description	
UC-ID	2.2
Name	Password Recovery
Description	IF user forget their own password to login, the mobile app available to recover user's password by connecting with Amazon Identity Provider and Amazon Web Service Cognito.
Actor(s)	User who is interacting with the application, Amazon Identity Provider, Amazon Web Service Cognito, STS (Security Token Service) and DynamoDB and Login activity

Precondition	User has the mobile application in mobile device and account (User ID + password) in the Mobile App. And the account should have the verified either user Email or Contact Number to authentication in recovery step.
Main Scenario	Step 1: Activity display Login Screen Step 2: User select the Forgot Password in login Screen to recover user's password Step 3.1: The activity request user for password recovery method (Email Address or short Message Service) Step 4.1: user choose the recovery method which is verified. Step 5.1: The Activity exchange the Identity to Cognito Token by communicating with AWS Cognito. Step 6.1: After Exchange, AWS STS (Security Token Service) send temporary token to end user's email or SMS based on the user's choice in step 4. Step 7: IF user input valid token, THEN end user will receive secure credentials from STS which enable user to access to the AWS server. And then the app will display the reset password form to change let user to change to new password. ALTERNATE Step 7: IF user input invalid token, THEN the app will display fail message. Step 8: The Login activity load change password activity. Step 9: The user enters new password and the system will update both local and central AWS DynamoDB.

2.7.12 Activity Diagram 2.2 Password recovery



2.7.13 Sequence Diagram 2.2 Password Recovery

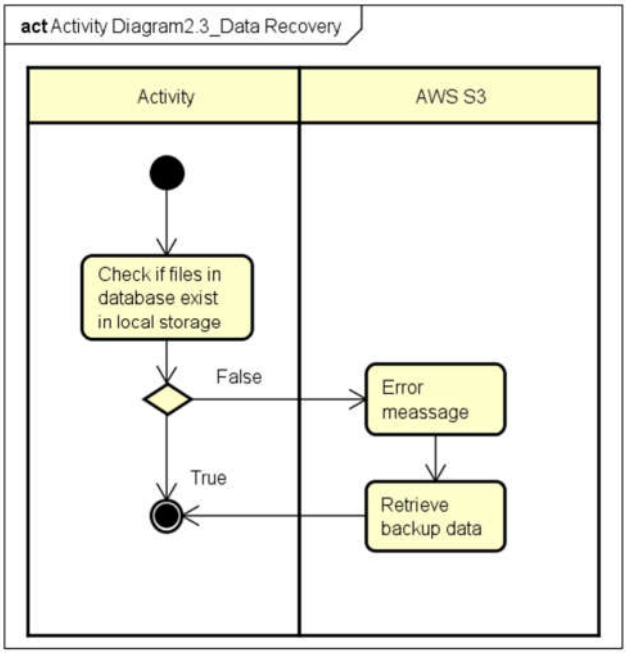


2.7.14 Use Case 2.3: Data Recovery

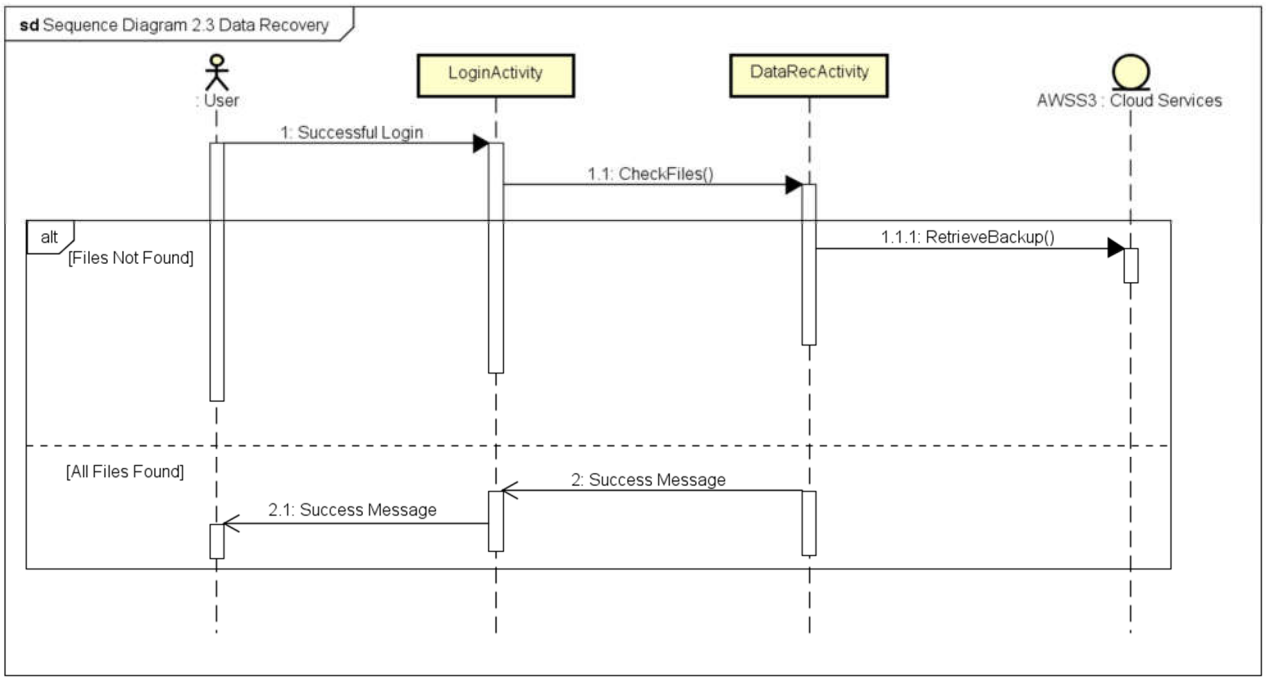
Use Case Textual Description	
UC-ID	2.3
Name	Data Recovery
Description	To allow user to retrieve backed up data from AWS S3 when he change device or removed the app

Actor(s)	User who is interacting with the application, System, AWS S3
Precondition	The user has successfully logged in to the system
Main Scenario	Step 1: Activity check if all the files inside database exist in the local storage Step 2: If it doesn't, the activity will retrieve back up data from ASW S3 and show the appropriate message Alternate Step 2: If it does, enter the system Step 3: Enter the system

2.7.15 Activity Diagram 2.3 Data recovery



2.7.16 Sequence Diagram 2.3 Data Recovery

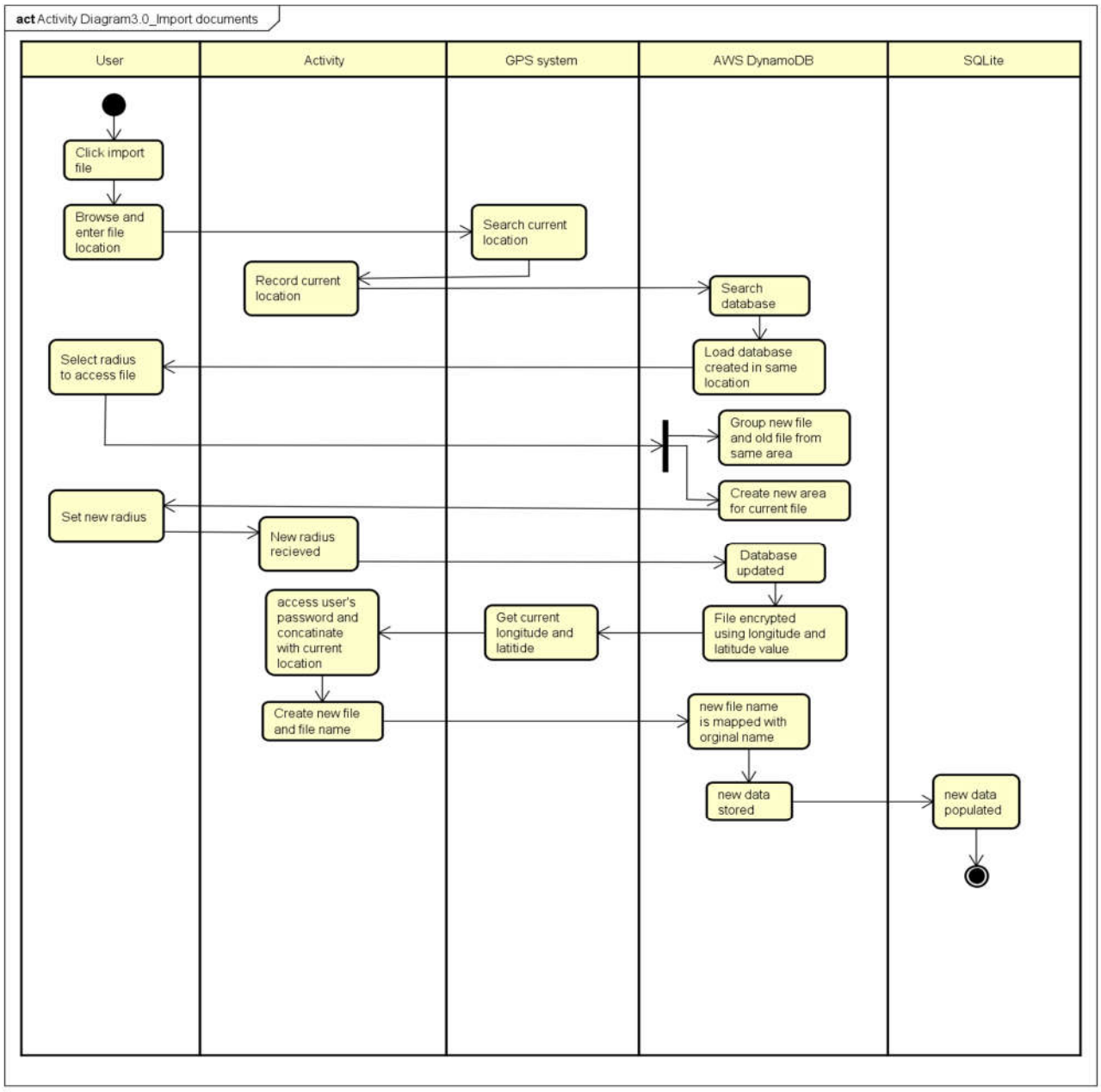


2.7.17 Use Case 3.0: Import documents

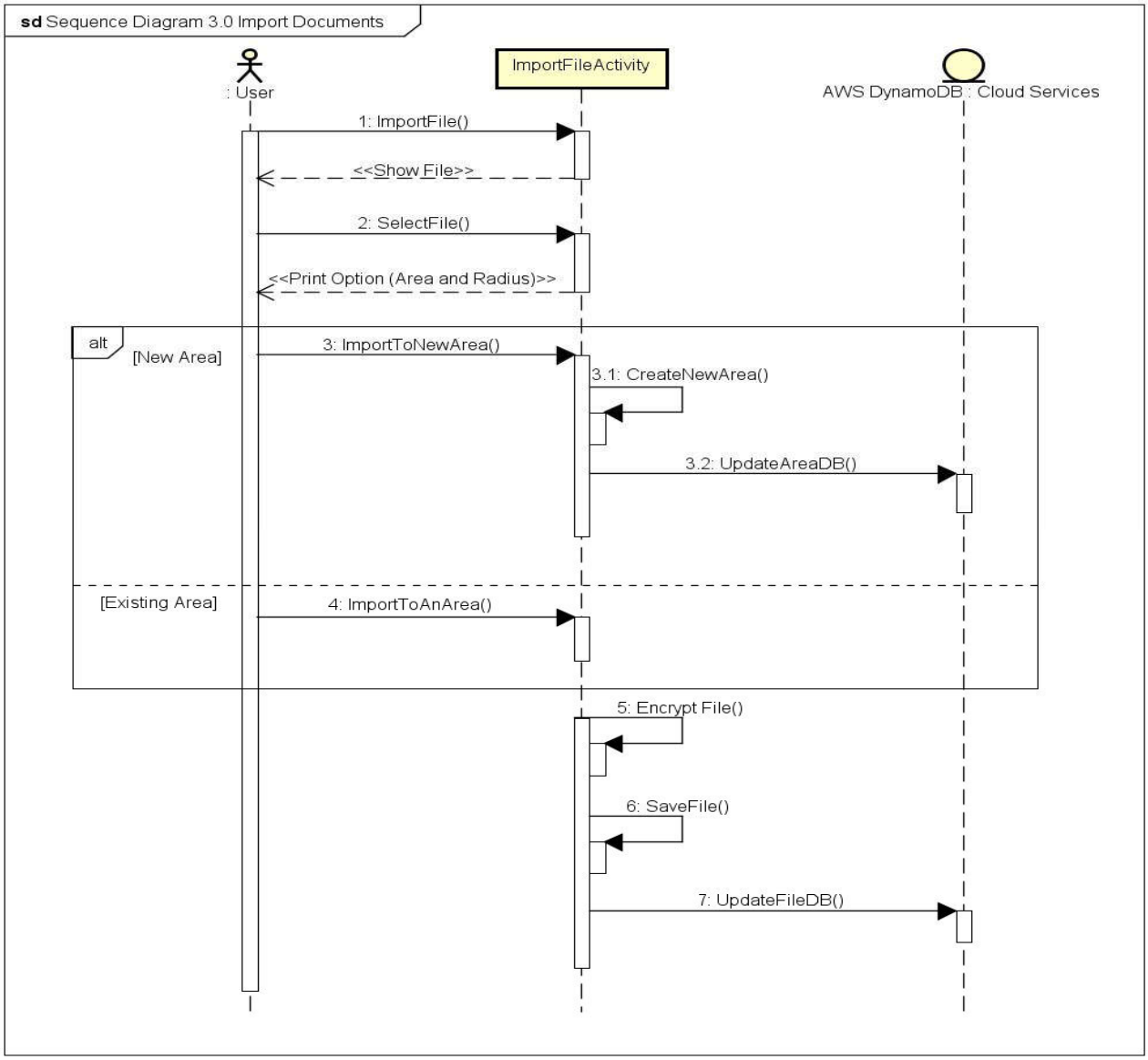
Use Case Textual Description	
UC-ID	3.0
Name	Import document
Description	This functionality makes sure that the data imported to the sever application is stored in a secured manner.
Actor(s)	User who is interacting with the application, Import Document Activity, GPS(System), AWS DynamoDB, Local SQLite database.
Precondition	The user wishes to secure a new document.
Main Scenario	Step 1: On clicking the import file option the user is prompted brows and enter the location of the file in the file system. Step 2: The activity records the current location of the user (from GPS system). Step 3: The activity searches the database and loads areas that where previous created in same location. The user is prompted to choose the radius where he wishes to access the file. The system groups the new file along with the order files in same area. Alternate to step 3: The user may choose to create a new area for the current file. If he wishes to create a new area he will be promoted to set the radius of his choice. Step 4: The activity will update the database and the file will be encrypted using the longitudinal and latitudinal value where the file was imported. Step 5: The activity accesses the user's password and concatenates it with the current longitude and latitude received from phones GPS (System).

	Step 6: A key is produced by hashing password digest produced in step 2 (Hash(pwd|Locationaldata)). Step 7: The activity creates a new file name for the file and this name is mapped with the original name in the database name. Step 8: All the above generated data is populated to local database (SQLite and the central database (DynamoDb) in the cloud. Step 9: The file is encrypted using AES 256 and the key generated in Step 3.

2.7.18 Activity Diagram 3.0 Import documents



2.7.19 Sequence Diagram 3.0 Import Documents

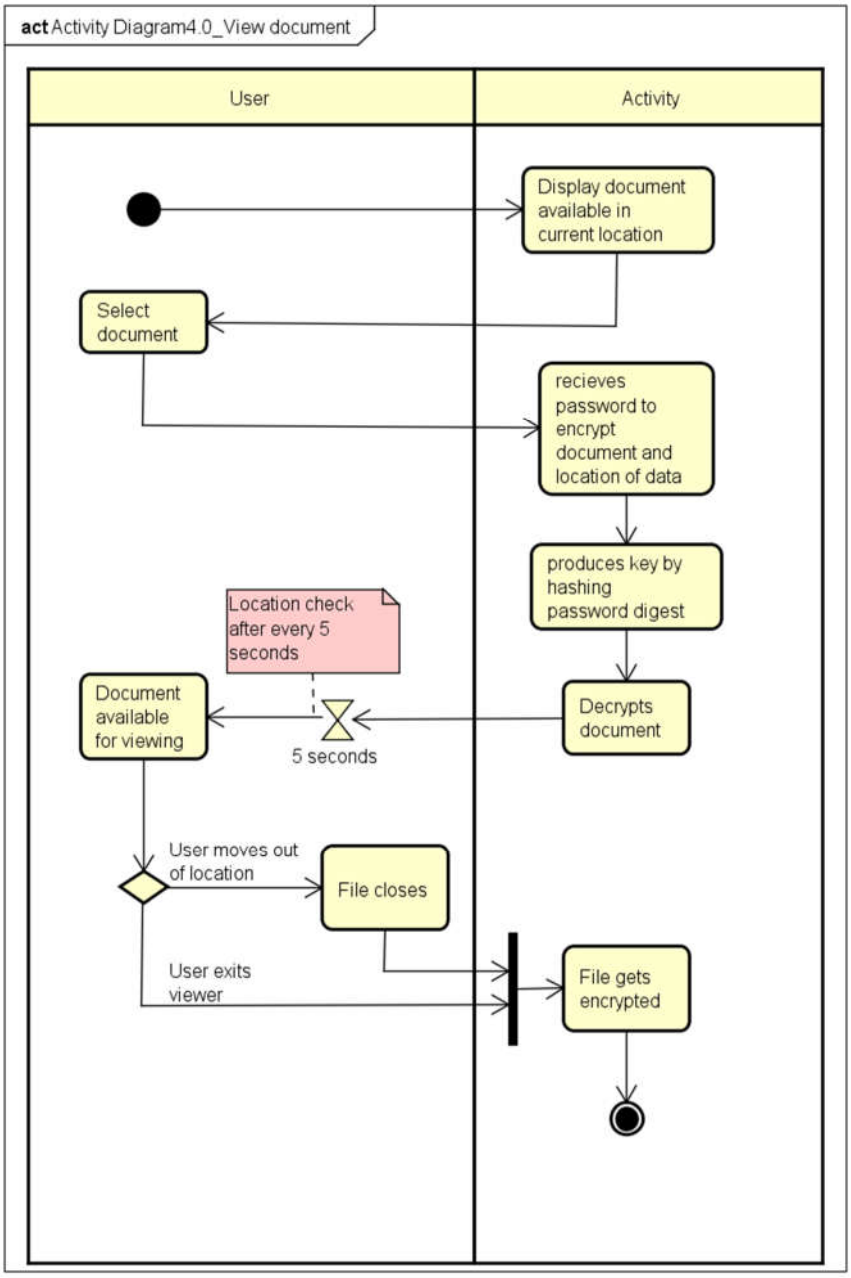


2.7.20 Use Case 4.0: View Document

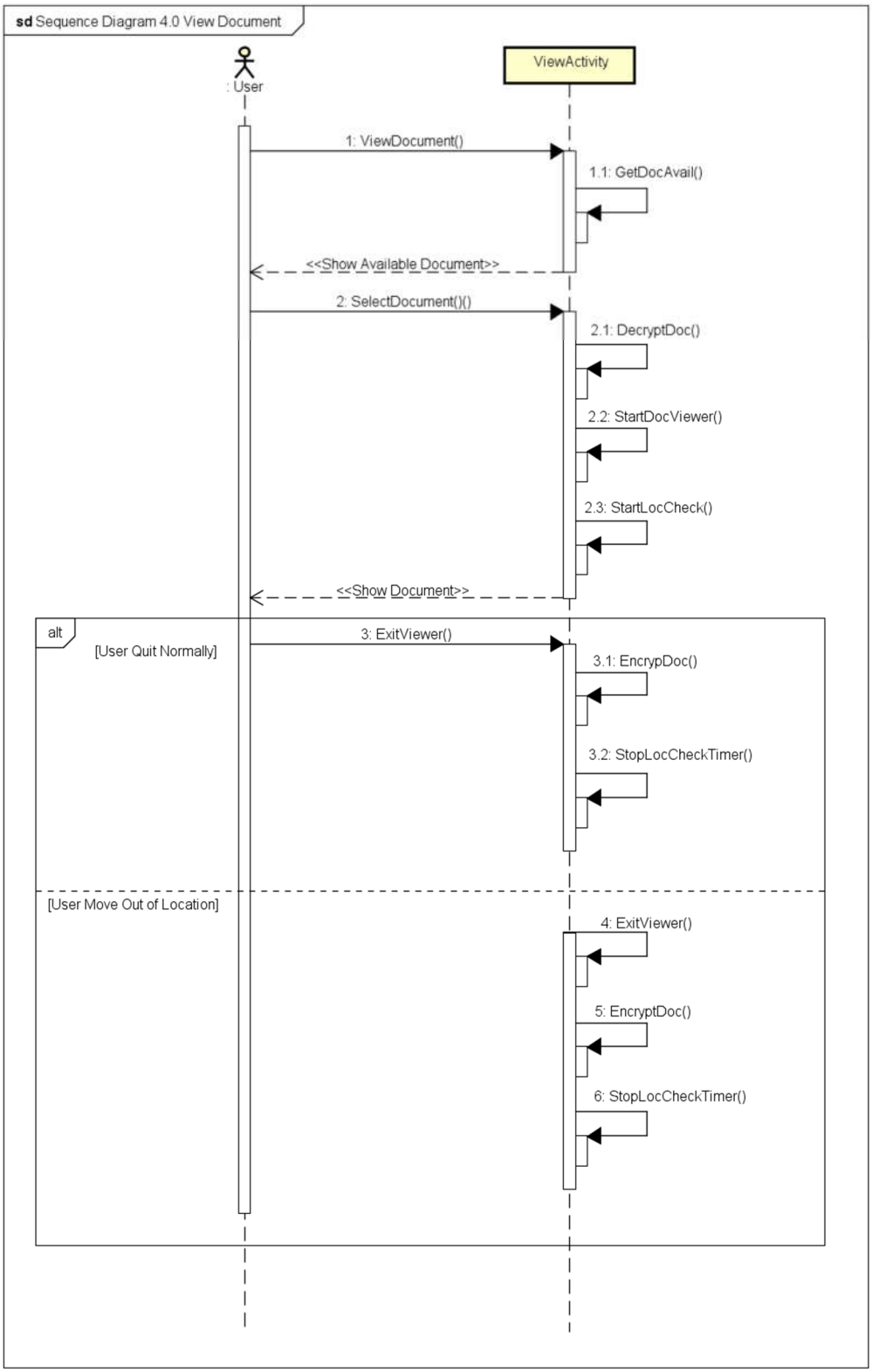
Use Case Textual Description	
UC-ID	4.0
Name	View Document
Description	To allow user to view its encrypted documents
Actor(s)	User who is interacting with the application, System, SQLite DB
Precondition	The user has successfully logged in to the system and is within the area of the document
	Step 1: Activity show the list of documents available in the user's current position

Main Scenario	Step 2: User select the desired document Step 3: The activity retrieves the password used to encrypt this document as well as the location of data from local SQLite DB Step 4: The activity produces the corresponding key by hashing password digest (Hash(pwd|Locationaldata)) Step 5: The activity decrypts the document with AES 256 using password digest computed in Step 4 as its key Step 6: The activity shows the document to the user Step 7: When the user is viewing a file, the activity checks his location every five seconds. Alternate Step 7: When the user moves out of the secure location the viewer will be closed and the file will be encrypted. A warning would be displayed if the user approaches the boundary. Step 8: When the user exits the viewer, the document is encrypted back with AES 256 using password digest computed in Step 4 as its key

2.7.21 Activity Diagram 4.0 View documents



2.7.22 Sequence Diagram 4.0 View Document

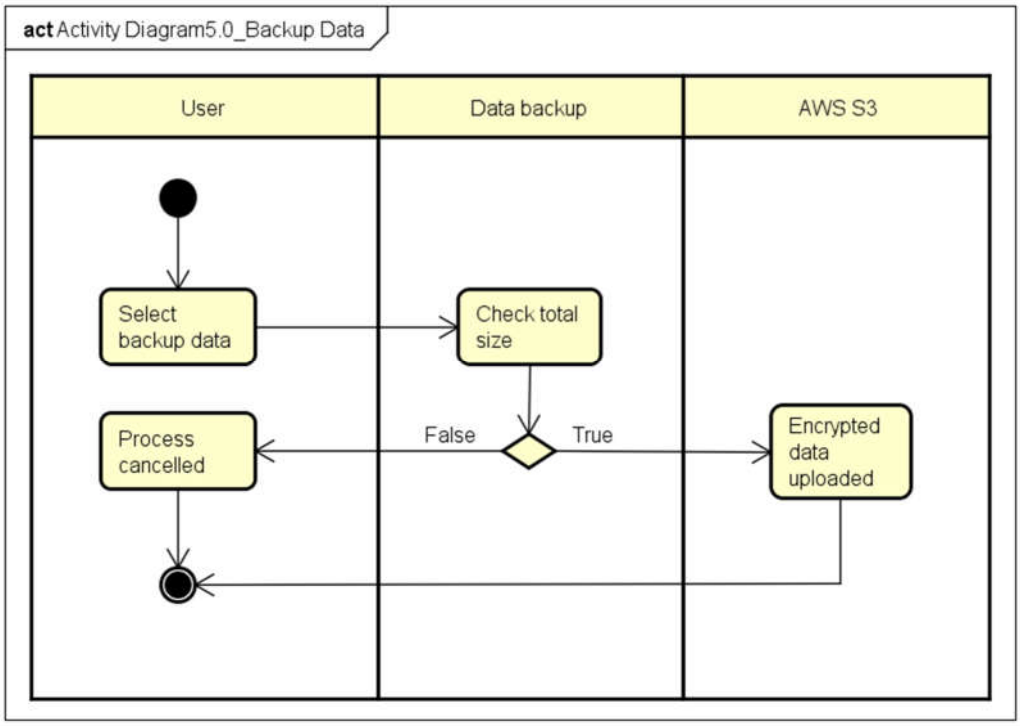


2.7.23 Use Case 5.0: Data Back up

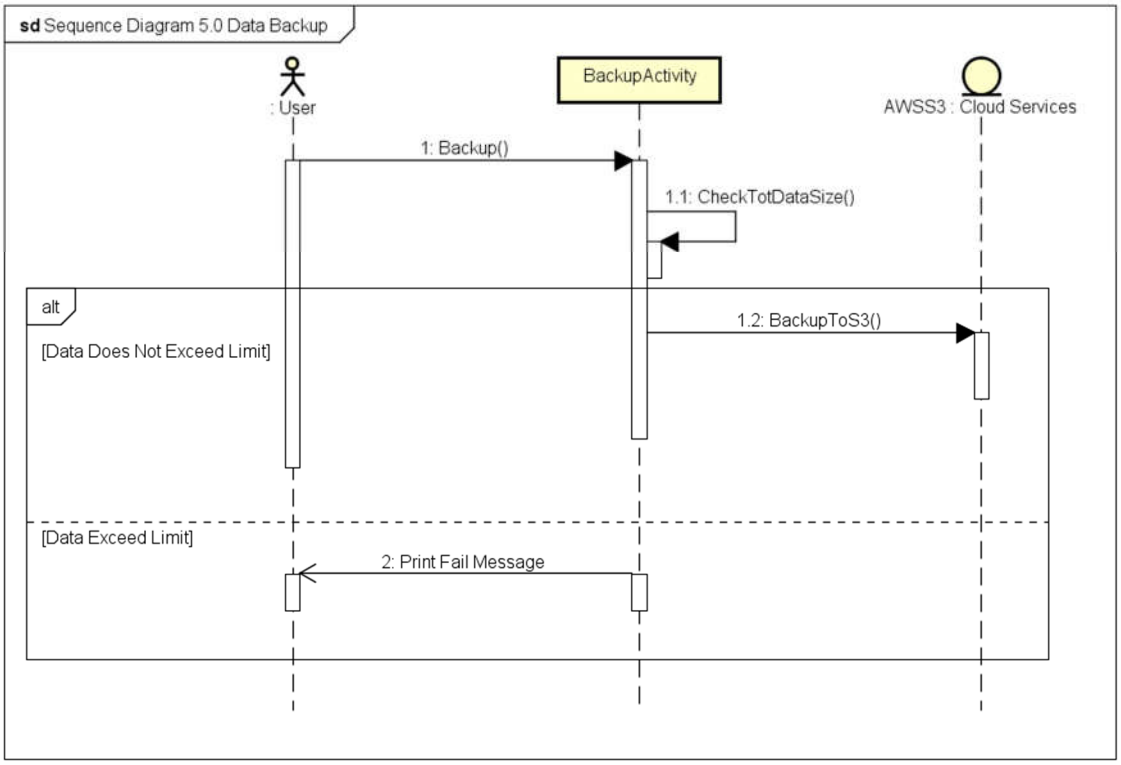
Use Case Textual Description	
UC-ID	5.0

Name	Data Back up
Description	To allow user to back up its encrypted documents to AWS S3
Actor(s)	User who is interacting with the application, Data backup activity, AWS S3
Precondition	The user has successfully logged in to the system
Main Scenario	Step 1: User select to back up its data Step 2: Activity check the total size of the backup data (Check limits, by default 500MB). Step 3: If it is within the limit, all the encrypted data is uploaded to AWS S3 Alternate Step 3: If it exceeds the limit, activity will cancel the process and show the appropriate message

2.7.24 Activity Diagram 5.0 Backup data



2.7.25 Sequence Diagram 5.0 Data Backup

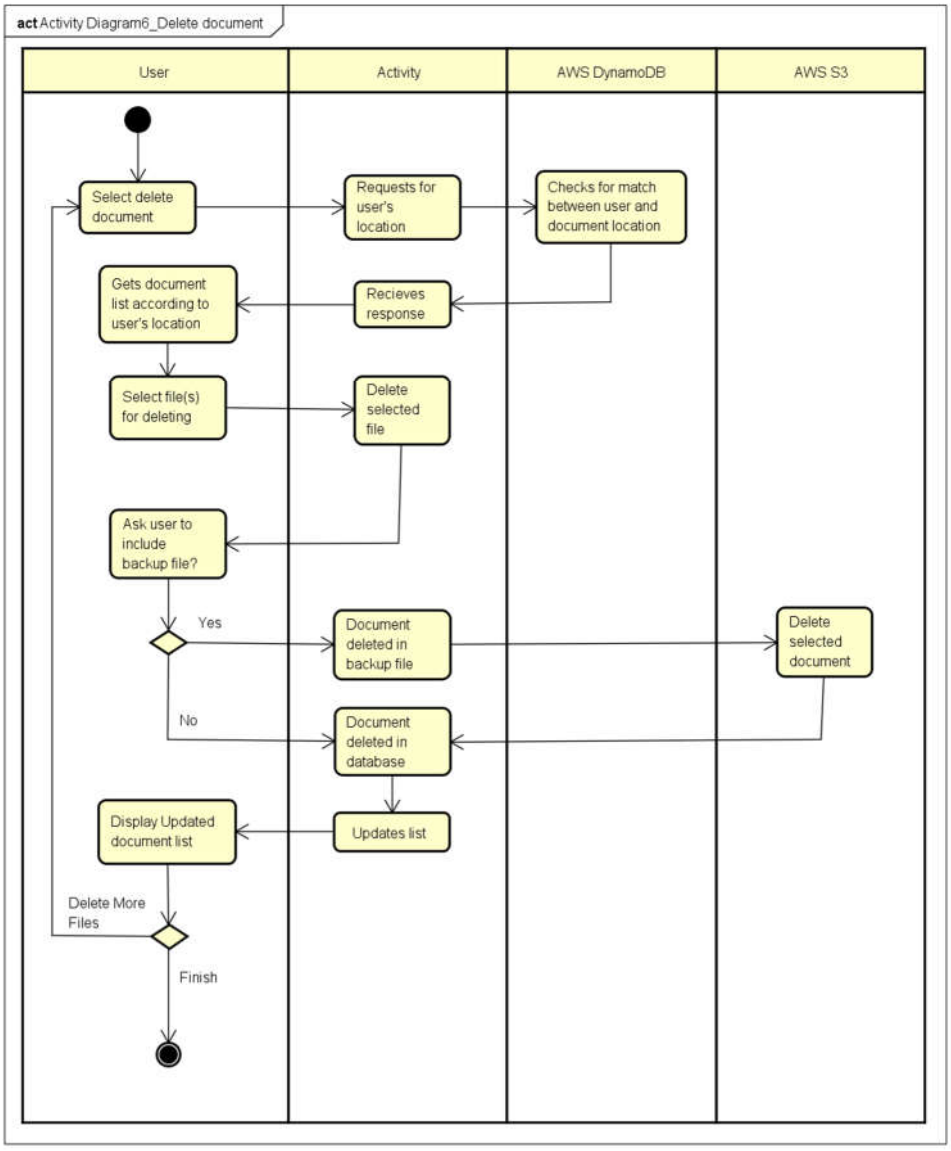


2.7.26 Use Case 6.0 Delete Document

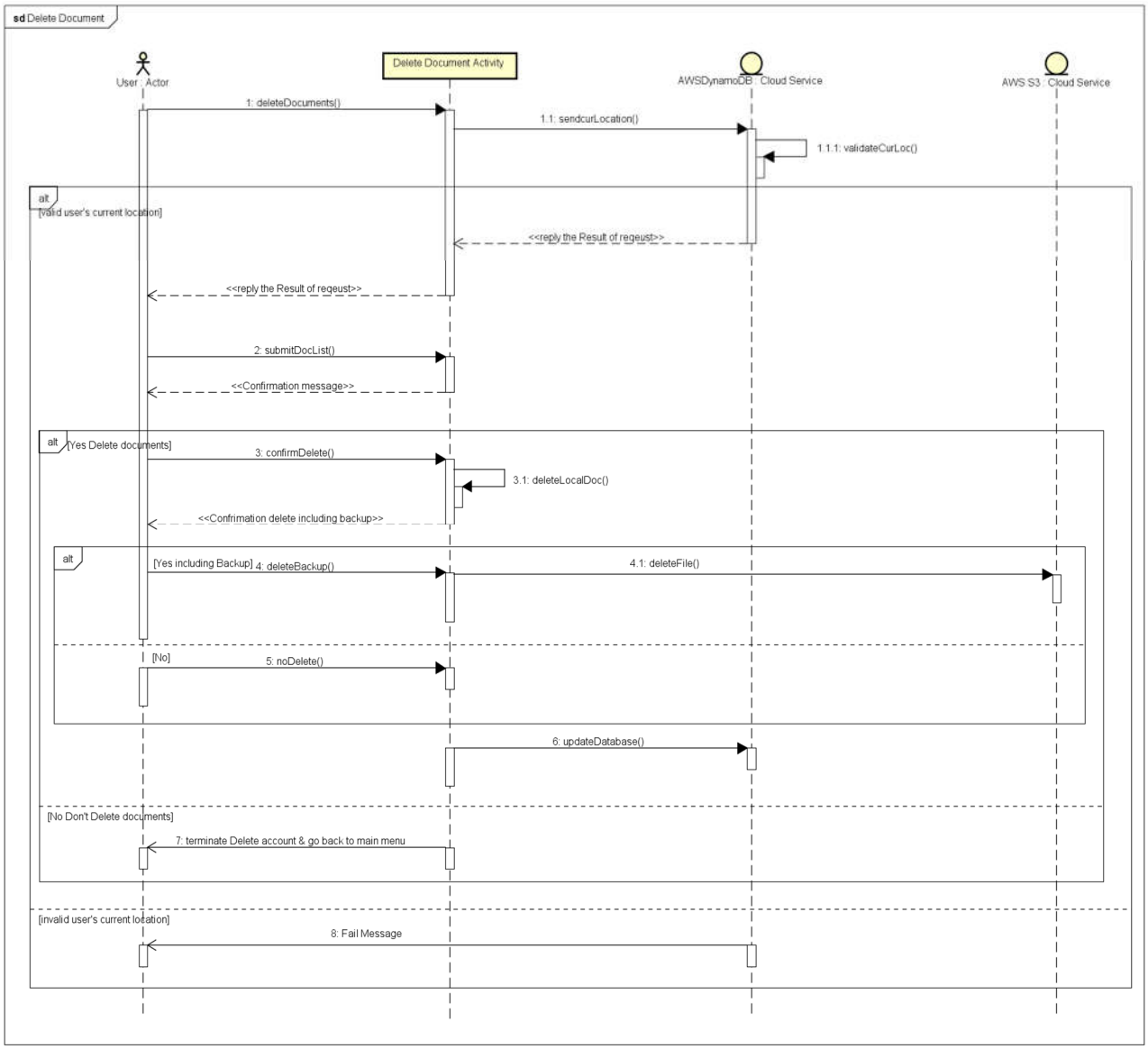
Use Case Textual Description	
UC-ID	6.0
Name	Delete Document
Description	This Function allow user to delete the document file in device and database in Server. Deleting file in Data Backup File is optional.
Actor(s)	User who is interacting with the application, AWS DynamoDB
Precondition	A user has installed the application. Also, user has an account in the app and some files in device, database and backup file. Before execute this function, user must already login to the app.
Main Scenario	Step 1: User select the Delete document option in Menu Display Step 2: The Activity request documents list, which is matching with the user's current location, by sending user's current location value to AWS DynamoDB. Step 3: AWS server validates the user's location value. Step 4: IF the value is valid, THEN the server sends the request result. ALTERNATE Step 4: IF the value is invalid, THEN the server rejects and send Fail Message Step 5: user select files for delete to Activity Step 6: Activity reconfirm to user whether they want to delete Step 7: IF user say yes, THEN send message to activity to remove file ALTERNATE Step7: IF user say no, THEN terminate the Delete Documents Step 8: The documents are removed in local database

	Step 9: Activity ask user whether they want to delete including backup in AWS S3. Step 10: IF User's answer is yes, THEN delete files in AWS S3 (Back up) ALTERNATE Step 7: IF User's answer is No, THEN terminate the process

2.7.27 Activity Diagram 6.0 Delete document



2.7.28 Sequence Diagram 6.0 Delete Document

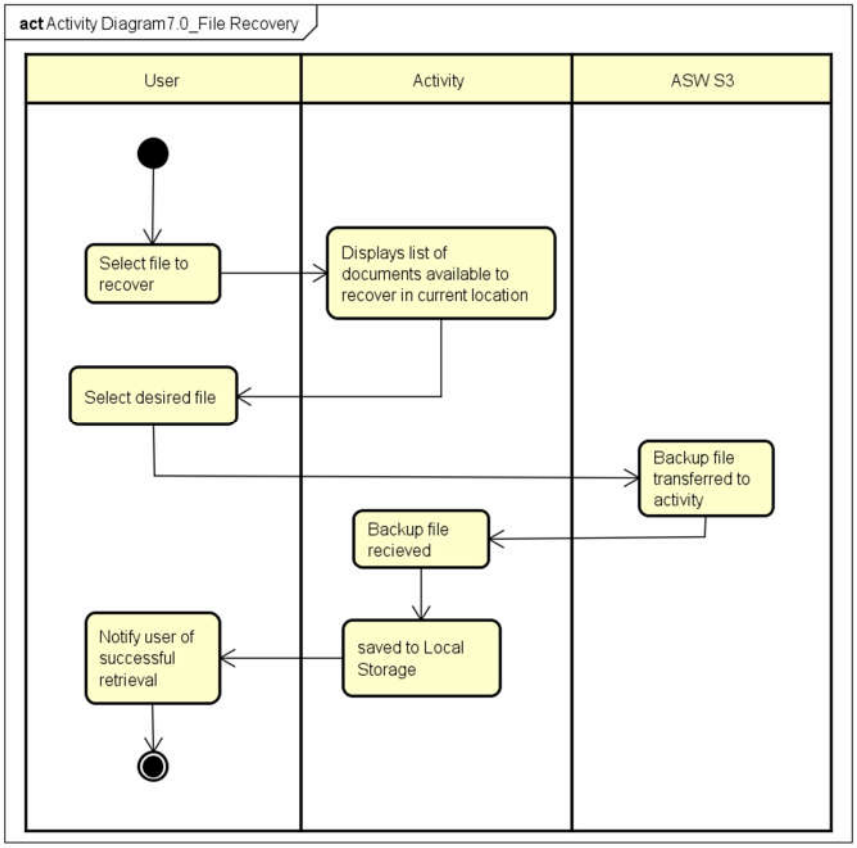


2.7.29 Use Case 7.0: File Recovery

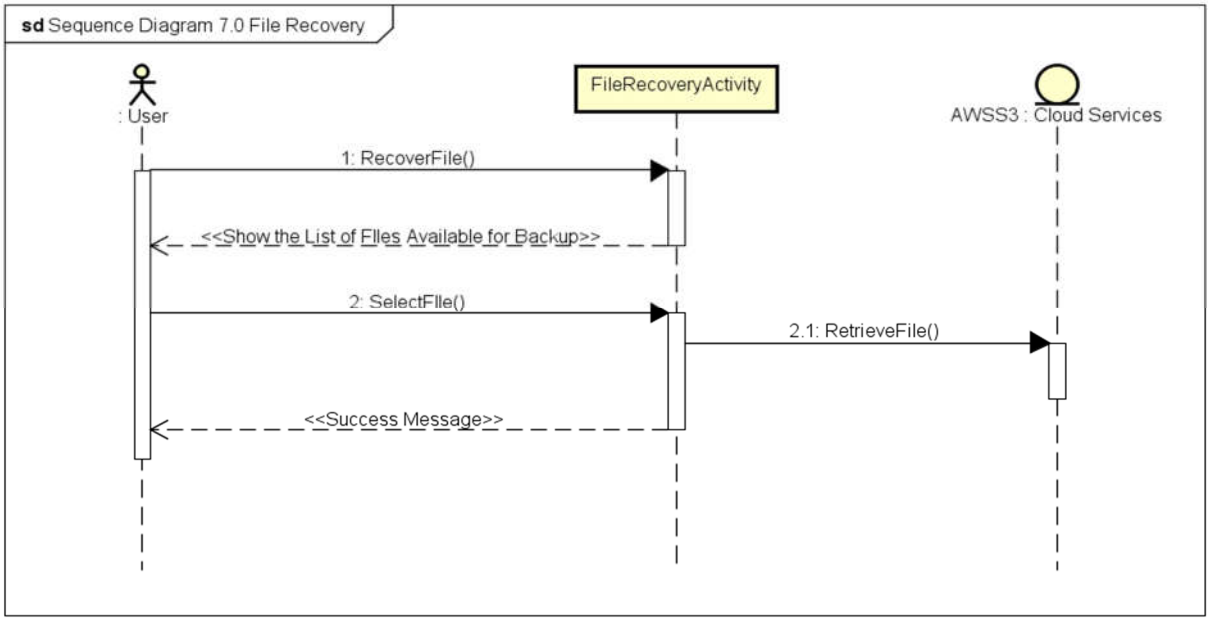
Use Case Textual Description	
UC-ID	7.0
Name	File Recovery
Description	To allow user to retrieve backed up data from AWS S3 when the user accidentally removed a file
Actor(s)	User who is interacting with the application, System, AWS S3
Precondition	The user has successfully logged in to the system
	Step 1: User select to recover a file. Step 2: Activity show the list of documents that are available to recover in the user's current position Step 3: User select the desired file

Main Scenario	Step 4: Activity retrieve backed up file from AWS S3 and saves it to the local storage. Step 5: The user is notified.

2.7.30 Activity Diagram 7.0 File recovery



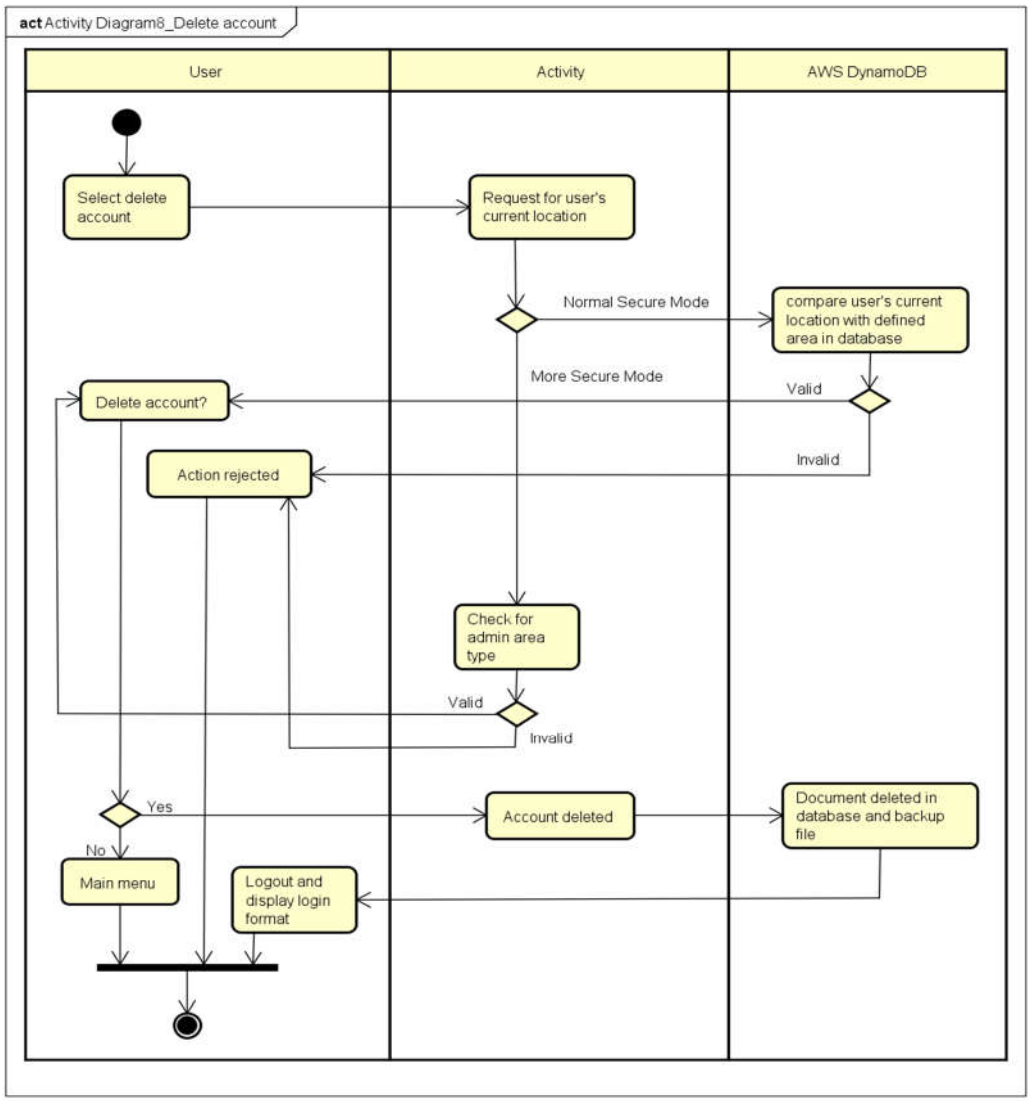
2.7.31 Sequence Diagram 7.0 File Recovery



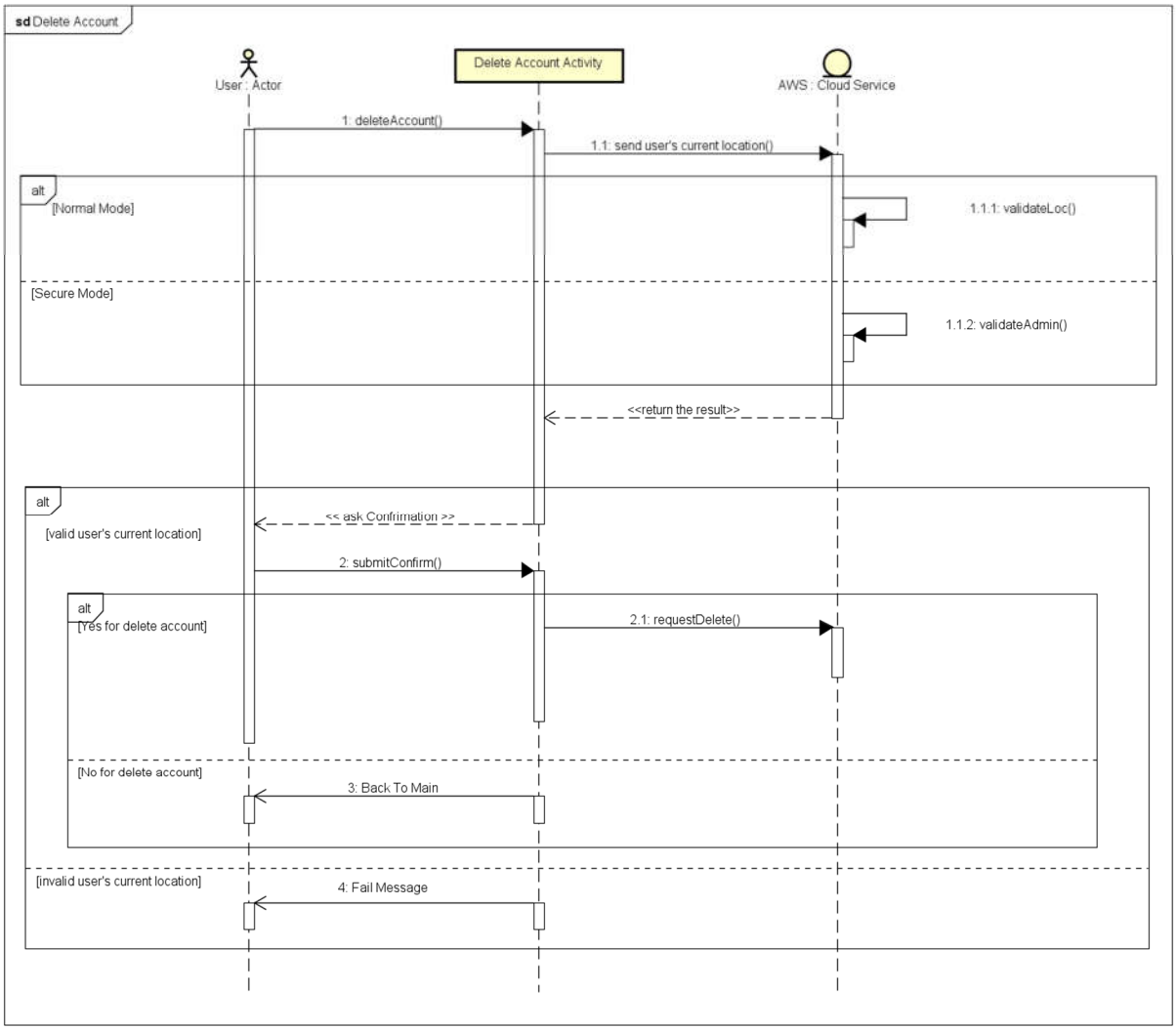
2.7.32 Use Case 8.0 Delete account

Use Case Textual Description	
UC-ID	8.0
Name	Delete account
Description	This Function allow user to delete user's account in the Application Server.
Actor(s)	User who is interacting with the application, AWS DynamoDB
Precondition	A user has installed the application. Also, user has an account in the application. user locates in valid area to delete account. Before execute this function, user must already login to the app. The Location Type will be divided as Normal Type or Admin Type. Default Mode will be Normal Type in location, which allow user to access and modify account data in whole area. (Whole areas are set as Normal Area) If user selects the more secure mode in the app, then user can access and modify the user account only in Admin Type of location. (There are some admin area and normal area.)
Main Scenario	Step 1: User select the Delete account option in Menu Display Step 2: The Mobile app request to AWS server with sending the user's current location. **METHOD1.** Normal Secure Mode (Only Normal Area Type) Step 3.1: Then AWS server compares user's current location area with user defined area in database Step 4.1: IF user locate in valid Normal Area, then the AWS server send message to App whether user will delete the account ALTERNATE Step 4.1: If it is not valid area, then the server sends reject message to app and app display reject message. **METHOD2**. More Secure Mode (Using Both Normal Area Type and Admin Area Type) Step 3.2: Then AWS server check user's current location area whether it is Admin Area Type or not. Step 4.2: If it is Valid Admin Area Type, then the AWS server send message to App whether user will delete the account, ALTERNATE Step 4.2: If it is not valid area or not Admin Area but Normal Area Type, then the server sends reject message to app and app display reject message. Step 5: IF user Select yes to delete account and AWS server delete the account with its document in database and backup file. ALTERNATE Step5: IF user select no, then terminate the process and go back to main menu. Step 6: After Deleting Account, The app will logout and display login format.

2.7.33 Activity Diagram 8.0 Delete account



2.7.34 Sequence Diagram 8.0 Delete Account



2.8 External Interface Requirements

2.8.1 Hardware Interfaces

The Hardware mainly consist of the mobile phone with android operating system and a central Amazon server. The Amazon server provides all the necessary infrastructure along with maintenance to do deploy the cloud storage service and a central database. All the mobile devises will make connection with this central server.

2.8.2 Software Interfaces

The client-side software will be installed from the android store and which then will retrieve all the necessary user information from Amazon DynamoDB. The Amazon service will scale as the number of users using the service increase.

2.8.3 Communications Interfaces

The communication interface is handled by AWS mobile API. All information passed to the API will be encrypted by the application. The communication is conducted over internet.

2.9 Non-functional Requirements

2.9.1 Performance Requirements

A temporary SQLite database will be created to store the user credentials locally to costs (time) that is incurred through constant communication with the central database. This database will be dropped when the user logout. This will boost the performance as the encryption and decryption processes will be heavily depend on this data. The performance is key requirement as mobile devises are small and don't have computational power of personal computers.

2.9.2 Safety Requirements

One of the key non-functional requirement is to keep the user data safe. Following are some of the identified Safety requirements: -

2.9.2.1 Dealing with stolen device

The user can migrate to a new device using the backup data and change password so that the data in the old phone can never be decrypted as the sever will never authenticate the user even if he knows the old password.

2.9.2.2 Dealing with missing folder

The user will be promoted to download the backup folder and will be able to access files as usual or the user can choose to create a new one.

2.9.2.3 Migration

The user may wish to move to a new device, and the app allows user to download the backup file from the cloud and continue using as usual. The user credentials are stored in a central database to ease migration.

2.9.2.4 Backup

The user can back up his data to cloud (AWS S3) to retrieve it later when the disaster strikes or for migration. The system will track the changes and only save the newly added file. The data stored in cloud is send over after encryption.

2.9.2.5 Changing Password

The user after logging into the app can change the password by providing the old password. The

password will be saved to the central database. The password forms a part of the key used to secure files and tables in the database. Once the password is changed the database will be secured using new password when the user logout.

2.9.2.6 Password recovery

If the user forgets the password, he will receive a password recovery code through his email or notification. Upon entering the code, he will be redirected to the changing password procedure.

2.9.2.7 Protection against GPS spoofing

The application checks if the mock GPS settings have been turned on every 30sec, If turned on it will logout to prevent GPS spoofing.

2.10 Security Requirements

Since the main aspect of this application is to keep files secure, this requirement is to non-trivial. Files should be encrypted using secure cryptographic algorithm such as Advanced Encryption standard (AES256). The file should not have the same name as the original file after encryption, the original name will be stored in the database along with the new name. The password should be hashed using strong hash functions such as the SHA256. The central database will have an additional layer of encryption provided by the cloud service provider (Amazon Web Service).
As for SQL injection, the query is performed by AWS Mobile API hence the adversary will not be able to inject malicious code to the server. All data that is sent over is not executed and will be stored in encrypted format.

2.11 Software Quality Attributes

Availability: Checking that the system always has something to function and always pop up error messages in case of component failures. In that case the error messages appear when something goes wrong to prevail availability problems.
Usability: Checking that the system is easy to handle and navigates in the most expected way with no delays. In that case the system program reacts accordingly and transverses quickly between its states.
Functionality: Checking that the system provides the right tools to perform task mentioned in the functional requirement section. Also testing these functionalities run smoothly and providing simple learnable interface.

3. Database Design

This section aims to meet following objectives:

- Describe the design of a DynamoDB and SQLite database, that is, a collection of related data stored in one or more computerized files in a manner that can be accessed by users or computer programs via a database management system (DBMS). It can also describe the software units used to access or manipulate the data.
- To serve as the basis for implementing the database. It provides the acquirer visibility into the design and provides information needed for software support.

3.1 DynamoDB design (NoSQL database)

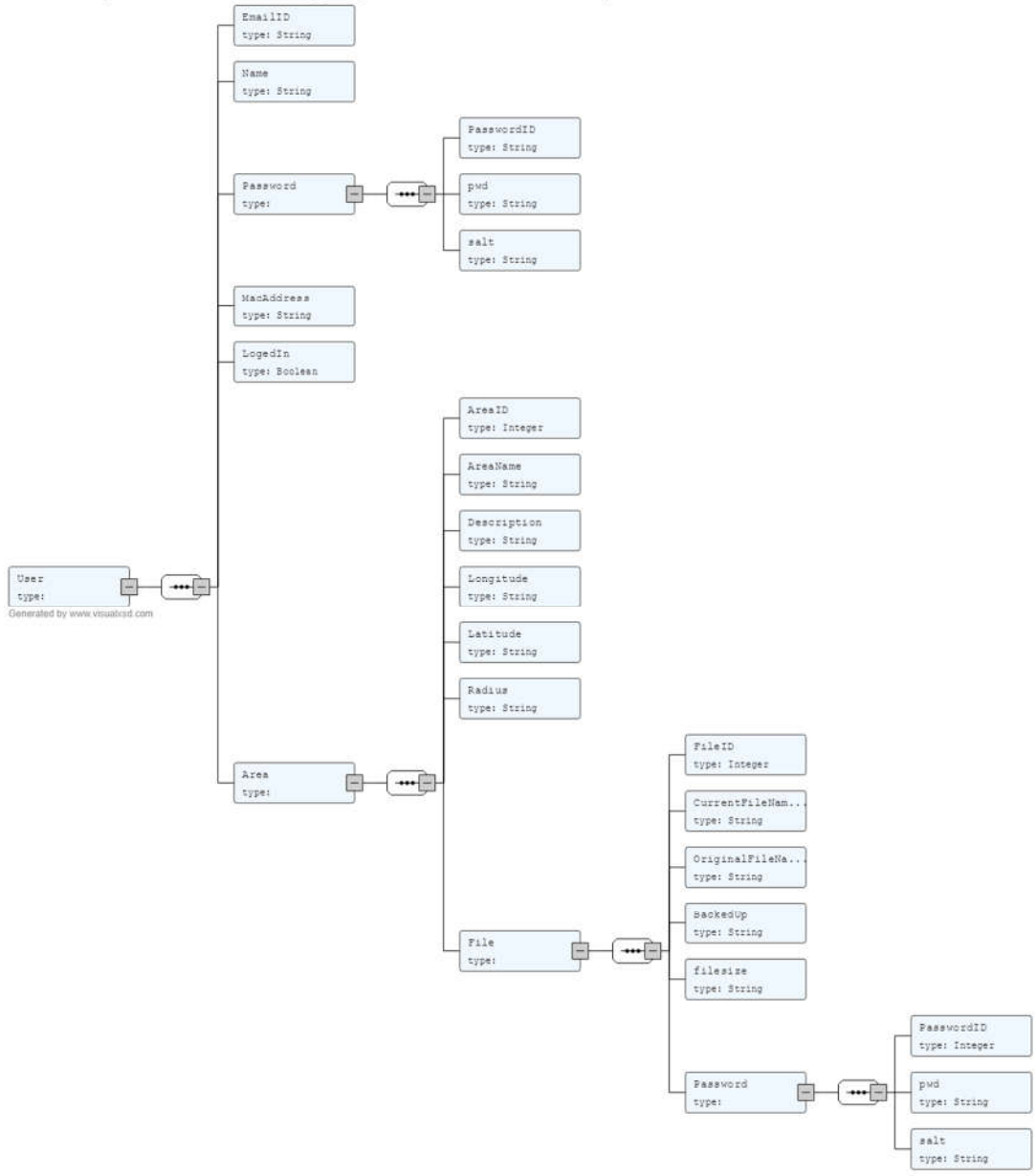


Figure 6: Database design

The diagram given above visualize the NoSQL database. This design was developed after creating a XML schema (Appendix 1) and using an online converter. [27]
The user element will hold all the user details which be mapped to there to his password and files. He will also have an option to set a user administration area.

3.1.1 Object Diagram

Here is an object diagram to show a given instance of the database.

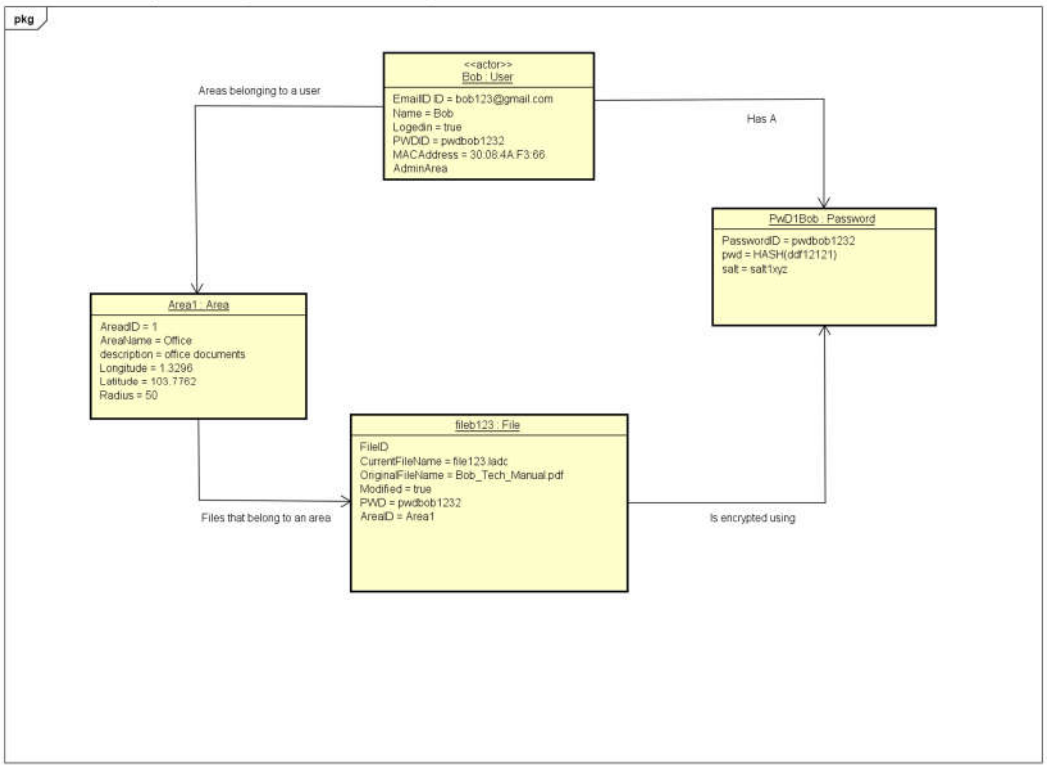


Figure 7: object diagram

This object diagram shows a sample data set and their relationships with other objects. For the propose of the object please refer to the section 3.2.2.

3.1.2 Data dictionary

3.1.2.1 Data dictionary for Element: User

Name	Data Type	Constrain	Description
Email ID (primary key)	string	Min :1, Max:1	Email ID of the user
Name	String		Name of the user
Password (Foreign Key)	Integer	Min :1, Max:1	The password of the user
LogedIn	String		Used to flag if the person is currently logged in a devise. So the second login can be detected.
InstanceID	String		Stores the application installation instance.
AdminArea	Integer		User has the freedom to set the admin area.

3.1.2.2 Data dictionary for Element: Password

Name	Data Type	Constrain	Description
Password ID (primary key)	Integer	Min :1, Max:1	ID to identify the password
Password	String		Hashed Password
Salt	String		Salt to prevent repeated keys being generated for

			encryption due to similar password.

3.1.2.3 Data dictionary for Element: File

Name	Data Type	Constrain	Description
FileID (primary key)	Integer		
CurrentFileName	string	Min :1, Max:1	A new name for the file assigned by the application
OriginalFileName	String		The original file name assigned by user.
BackedUP	Boolean		This a variable to make sure if the data has been backed up or new.
Password	Integer		Password that was used to encrypt the file (password ID)
Area	Integer		The area where the file has been grouped in (Area ID)

3.1.2.4 Data dictionary for Element: Area

Name	Data Type	Constrain	Description
Area ID (primary key)	string	Min :1, Max:1	ID to identify the area
AreaName	string		Stores the name of the area given by the user.
Description	string		Stores the description created by the user for the area.
Longitude	Decimal		The Longitude of the first file that was created in this area
Latitude	Decimal		The Latitude of the first file that was created in this area.
Radius	Decimal		The radius around the point where the first file was created.

3.2 SQLite database design (Relational database)

3.2.1 Conceptual diagram

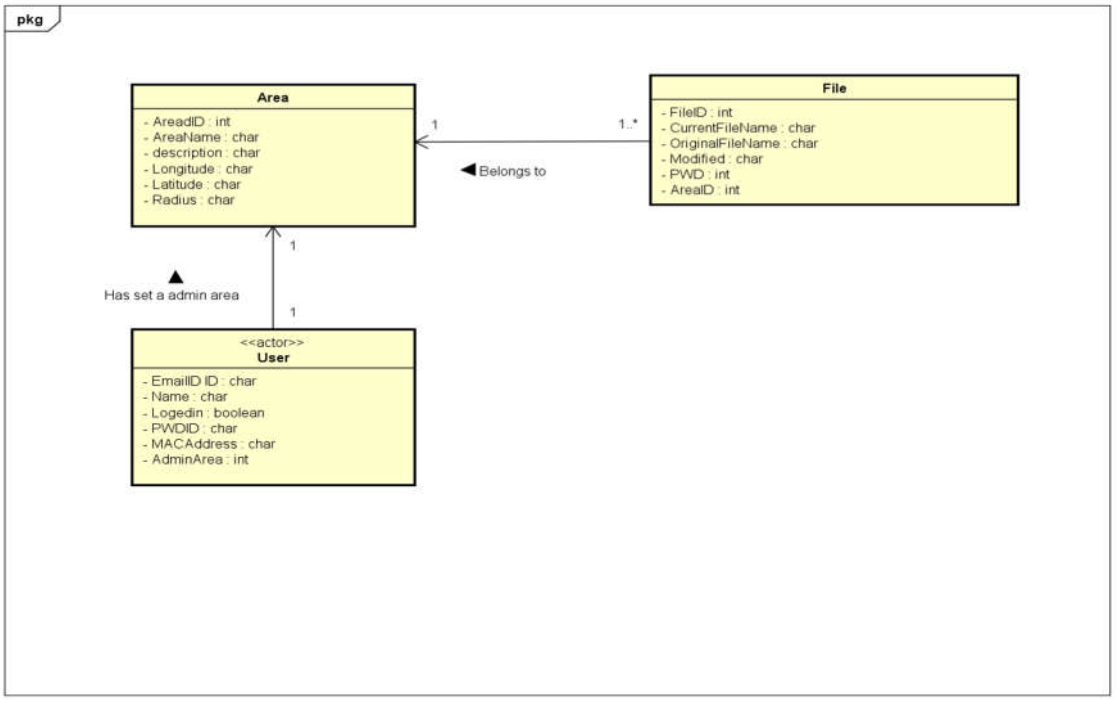


Figure 8: conceptual diagram

3.2.2 Description

This diagram displays the conceptual model of the SQLite database. This database will be created after the it has imported the user's data. The user session will have the user's details who is currently logged in. The user will have a password and an admin area where he can make changes to his account such as changing password (optional up to user to set it up). Each user will have zero or more files. The local database will only hold the Area and file information. All the data other than the primary and foreign keys will be stored after encryption using user's password. The database contents will be decrypted when user makes request.

3.2.3 Purpose of Tables

3.2.3.1 Purpose of Area Table

This table stores the information regarding the Area a file was created. The radius is the area around that point where the files grouped in that area can be accessed. The longitudinal and latitudinal value is used to encrypt the file in that area.

3.2.3.2 Purpose of File Table

This table stores all the information regarding a file used by the user. When the file is imported into the application, a new file name is generated, and it is mapped with the actual table. The longitude and the latitude of the location where the file was created is also stored. The file will also will have a password which was used to encrypt the file.

3.2.3.3 Purpose of User Table

The user stores the details of the user currently logged into the system. The user will have a password and an admin area where he can make changes to the account. The primary key of the table will be the email ID. This table will only one record as the database only belongs to one user.

3.2.4 Table 10: Relations

From Table	To Table	Relation
User	Area	A user may set an admin area.
Files	Area	A file belongs to an area.
User	File	A user may save more than one file.
File	Password	A file will be encrypted using a password.
User	Password	A user has a password.

4. Architecture Design

4.1 Architecture overview

The system works in mobile environment. The user may carry his mobile device along with him where ever he may wish to go. Hence the system is developed taking this into consideration. The user may be anyone who wish to use secure his or her files.
The architecture takes these functionalities into consideration.

- User authentication
- Signup
- Password recovery
- Importing new files
- User account management
- File backup
- Local file storage
- Deleting files
- File recovery and data (all files) recovery
- Location based file locking
- Grouping of files in same area.
- User account management

The application will use a SQLite database internally and will require continues internet access and GPS connectivity.
The system makes use of the external interface that is the AWS services such as Cognito, DynamoDB and S3.

4.2 Architecture Design Patter

The application will be implemented using *Model View Presenter* (MVP) architectural pattern.

4.2.1 Why MVP pattern?

MVP pattern developed from MVC (Model View Controller) architectural pattern which has been gaining importance over time.
In MVC we have a problem arising from the fact that Android activities are closely coupled to both interface and data access mechanisms. This leads to following issues: -

- **Testability** - The controller is tied so tightly to the Android APIs that it is difficult to unit test.
- **Modularity & Flexibility** - The controllers are tightly coupled to the views. It might as well be an extension of the view. If we change the view, we have to go back and change the controller.
- **Maintenance** - Over time, particularly in applications with anemic models, more and more code start getting transferred into the controllers, making them bloated and brittle.

MVP pattern solve this by combining the activity with the View in MVC (XML UI definition) and creating a new layer called the precentor to update the view.

4.3 Logical architecture overview

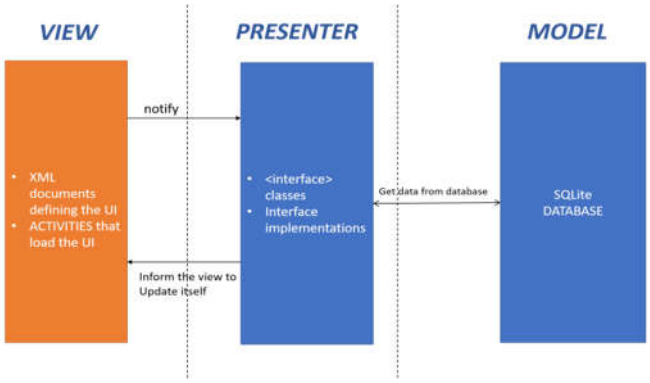


Figure 9: MVP design

The application logical architecture is developed based on the MVP pattern described above.

4.3.1 Layer diagram

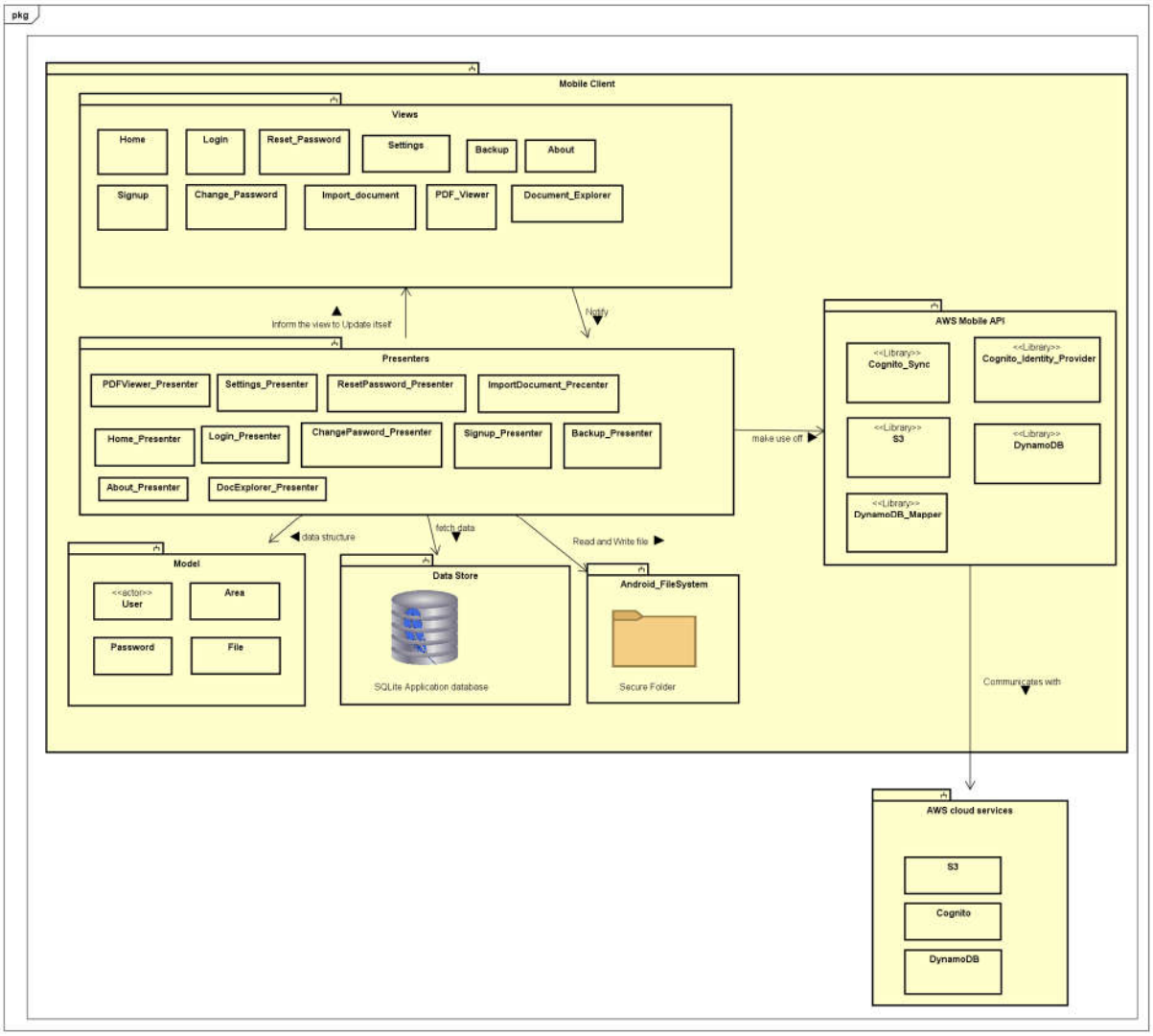


Figure 10: System Architecture

4.3.1.1 View

This layer contains activity classes and xml files that define the structure of the android application user interface. Each view will also have an interface to link with the Presenter. The interface will define the different sections that can be modified in the UI.
There are 11 separate activates the layout of these document will be defined in the user interface design document.

4.3.1.2 Presenter

This layer defines the classes which hold the core business logic. The Presenters form a link between the UI and the model. The interface notifies the Presenter based on the user activity and the Presenter will do necessary logical computation and instruct the user interface to update itself. Each activity will have a corresponding Presenter, but the Presenter may resort to other helper classes to do some additional computations.
When there is a need to retrieve data from the database the Presenter will make connection with the SQLite database and view and same when it comes to updating database. Presenter also read and write to files in android files system. Presenter will also make use of AWS Mobile SDK to access AWS services.

4.3.1.3 Model

This layer forms a data structure layer which does all the necessary backend operations related to the business logic.

4.3.1.4 AWS Mobile API

This is the library created by AWS and forms a interface between the application and the AWS cloud services. The main sub libraries required for this application will be Cognito related libraries (User authentication) [28], S3 related libraries (Data storage) [29] and DinamoDB (Database) [30]. AWS IAM (Identity and Access Management) provide a secure access protocol to interact with amazon services [31].

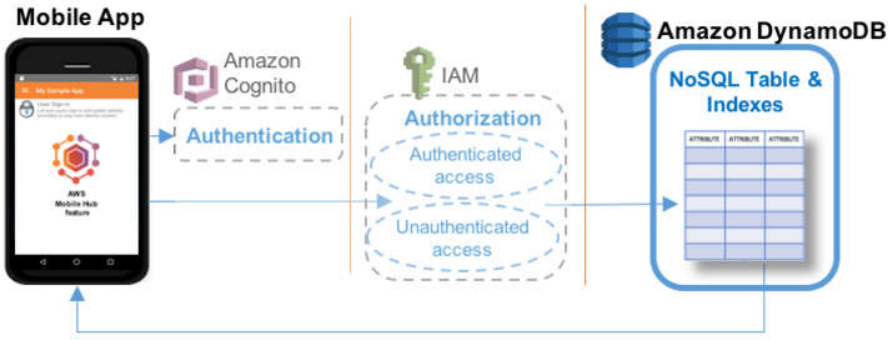


Figure 11: AWS DynamoDB Access

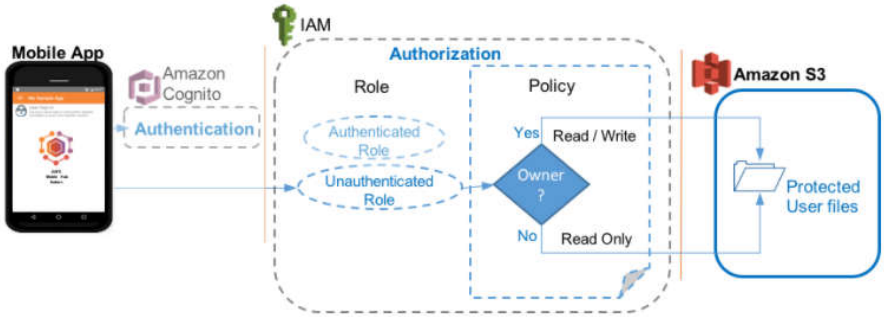


Figure 12: AWS S3 Access

4.3.2 Logical design qualities

The system intends to achieve following qualities through the above implementation
Design principles of the application.

4.3.2.1 Modularity

Separating the functionality of a program into independent, interchangeable modules, such that each contains everything necessary to execute only one aspect of the desired functionality.

4.3.2.2 High Cohesion

Each module has functions and elements that are strongly related, only to fulfil one purpose or task.

4.3.2.3 Low Coupling

Modules are loosely coupled and independent so that a change in one module do not affect the other modules.

4.3.2.4 Standardization

Implementation will conform to standard that has been established and agreed by different parties, this is crucial for things like security.

4.4 Physical Architecture

4.4.1 Physical Architecture Overview

The architecture adopted by the application will be a new and sophisticated *cloud-client architecture*. This architecture works by making use of cloud computing resources to manage user's data and credentials as well as authenticating user. The proposed application will be using AWS (Amazon Web Services) as its cloud service provider and AWS Mobile SDK [23] for the development of the app. The main advantage this architecture has is that we don't need to manage the resources in the cloud including the security of it.

4.4.2 Deployment Diagram

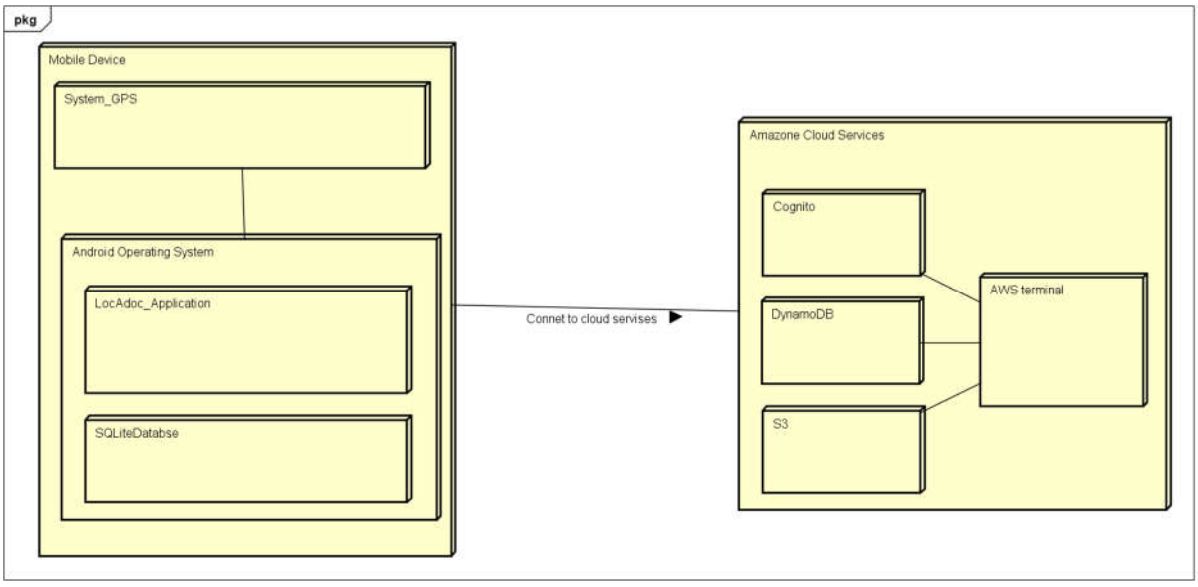


Figure 13: System deployment diagram

The above diagram shows the physical architecture of the system. The execution environment of the system is a mobile devise running android operating system. The application requires access to the phones GPS device to get locational details of the user. The application will also have local SQLite database.
Amazon Cloud Service will be the external interface providing centralized user authentication, database and cloud storage services.

5. System Architecture Qualities

The proposed application will have the following quality:

5.1.1 Extensibility

The proposed application can add additional functionality without changing or damaging much of the current system. New data types can be added if it is supported by the android.

5.1.2 Maintainability

Following the design principles of high cohesion and low coupling, small modifications will not be a problem. Changing one module will not affect other modules significantly.

5.1.3 Performance

The response time will be in acceptable manner even with the huge amount of data that are processed. Efficient encryption algorithm is used as well as other processing algorithm.

5.1.4 Usability

Adapting KISS (Keep It Simple Stupid) principle in designing interfaces will give user easier times in learning and figuring out the proposed application. It lets user to take less time to perform a certain task.

5.1.5 Compatibility

The proposed application will be able to run in various type of android devices as well as different version of android.

5.1.6 Security

Data are kept safe by encryption and login is required to have access. Security measures like protection against SQL injection or encryption algorithm will follow standard.

6. Component Design

6.1 Component Diagram

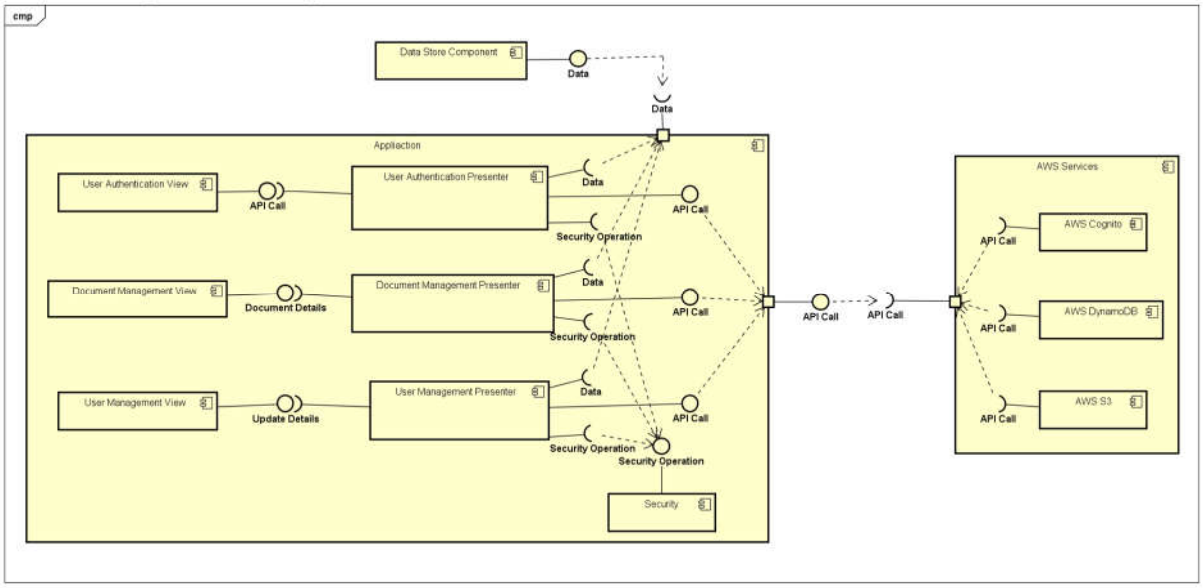


Figure 14: Component diagram

6.2 Component Description

6.2.1 User Authentication View Component

This component consists of classes that will show the UI (User Interface) for authenticating user and call its presenter when the user interact with it. The classes are as follow:

- **Home View**
  The UI (User Interface) when user first open the application. It gives user the option to create new account (sign up) or log in with existing account.
- **Log in View**
  The UI (User Interface) for user to log in with existing account. All details are entered and a log in button will call its presenter
- **Reset Password View**
  The UI (User Interface) for user to reset password if the user forget its current password
- **Sign up View**
  The UI (User Interface) for user to create a new account. All details are entered and a sign up button will call its presenter

6.2.2 File Management View Component

This component consists of classes that will show the UI (User Interface) for creating, viewing, or deleting a file, and then call its presenter when the user interact with it. The classes are as follow:

- **Document Explorer View**
  The UI (User Interface) when user want to explore directory and select a particular file
- **PDF Viewer View**
  Open and display a pdf file. This view is controlled by its presenter
- **Import Document View**
  The UI (User Interface) for user to select a file in local storage and import it to the application

6.2.3 User Management View Component

This component consist of classes that will show the UI (User Interface) for managing user's details, and will also include backup. The classes are as follow:

- **Settings View**

The UI (User Interface) when user want to view its details and/or change its particulars as well as to backing up its data.

- **Change Password View**
  The UI (User Interface) for user to enter its new password and submit the form. Submitting the form will call its presenter
- **Backup View**
  The UI (User Interface) for user to confirm to back up its data. Confirming will call its presenter

6.2.4 User Authentication Presenter Component

This component consists of classes that will handle all the user authentication, and make API call to AWS (Amazon Web Services) to do the operation. It makes use of Security Component and will update the Data Store Component. The classes are as follow:

- **Home Presenter**
  It will decide whether to bring user to Log in View or Sign up View based on the input
- **Log in Presenter**
  All logics related to authenticating user and decide on which view next to show, this includes API call to AWS Cognito
- **Reset Password Presenter**
  All logics related to resetting user's password and decide on which view next to show, this includes API call to AWS Cognito
- **Sign up Presenter**
  All logics related to creating new user and decide on which view next to show, this includes API call to AWS Cognito and AWS DynamoDB

6.2.5 File Management Presenter Component

This component consists of classes that will handle creating, viewing, and deleting a file; and make API call to AWS (Amazon Web Services) to do the operation. It makes use of Security Component to encrypt/decrypt the files and will update the Data Store Component. The classes are as follow:

- **Document Explorer Presenter**
  All logics related to exploring file and decide on what operation done to this file based on the user input (view or delete). This might include API call to AWS DynamoDB and AWS S3. After that it decides on which view to go next.
- **PDF Viewer Presenter**
  Decrypt selected file using Security Component and display it on screen. After user close the PDF Viewer View, encrypt back the file and display back Document Explorer View. This will include logic to constantly check user's location
- **Import Document Presenter**
  All logics related to selecting a file in the local storage and save it to the application, Security Component will be used here to encrypt the file. This will include API call to AWS Dynamo DB. After that it decides on which view to go next.

6.2.6 User Management View Component

This component consists of classes that will handle updating user's details and backup. API call to AWS (Amazon Web Services) is done to do the operation. It makes use of Security Component and will update the Data Store Component. The classes are as follow:

- **Settings Presenter**
  All logics related in making changes to user's particulars and backing up data. This might include API call to AWS Cognito, AWS Dynamo DB, and AWS S3. After that it decides on which view to go next.
- **Change Password Presenter**
  All logics related in changing the user's password, this includes API call to AWS Cognito and AWS Dynamo DB. After that it will display back Settings View.

- **Backup Presenter**
  All logics related to back up user's data, this include API call to AWS Dynamo DB and AWS S3. After that it will display back Settings View.

6.2.7 Data Store View Component

This component consists of classes that are used to save information about an item/entity. These classes are mainly used for AWS Dynamo DB and SQLite. Local database SQLite is also included in this component. The classes are as follow:

- **User**
  Keeping all the user's information that are listed in the DDD (Database Design Document)
- **Area**
  Keeping all the area's information that are listed in the DDD (Database Design Document)
- **Password**
  Keeping all the password's information that are listed in the DDD (Database Design Document)
- **File**
  Keeping all the file's information that are listed in the DDD (Database Design Document)

6.2.8 Security Component

This component consists of all security operation needed for the system. All of this operation will be: encryption algorithm, decryption algorithm, file encryption algorithm, file decryption algorithm, encoding, decoding, and hash.

6.2.9 AWS Cognito Component

AWS Cognito is a part of AWS that will be used by the system for managing the users of the system

6.2.10 AWS Dynamo DB

AWS Dynamo DB is a part of AWS that will be used by the system to act as a database for keeping all information about

6.2.11 AWS S3

AWS S3 is a part of AWS that will be used by the system to keep user's file for back up so that it can be retrieved later

7. User Interface Design

The purpose of this section is to provide a complete description of the user interface that would-be part of the completed LocAdoc application.
The main scope of this document is to: -

- Get a clear understanding on how user will transit between different activities.
- Get an overview of each interface design.

To readers may wish to refer to software requirement specification sheet for further details on system functionalities.

7.1 User flow design

7.1.1 User flow diagram

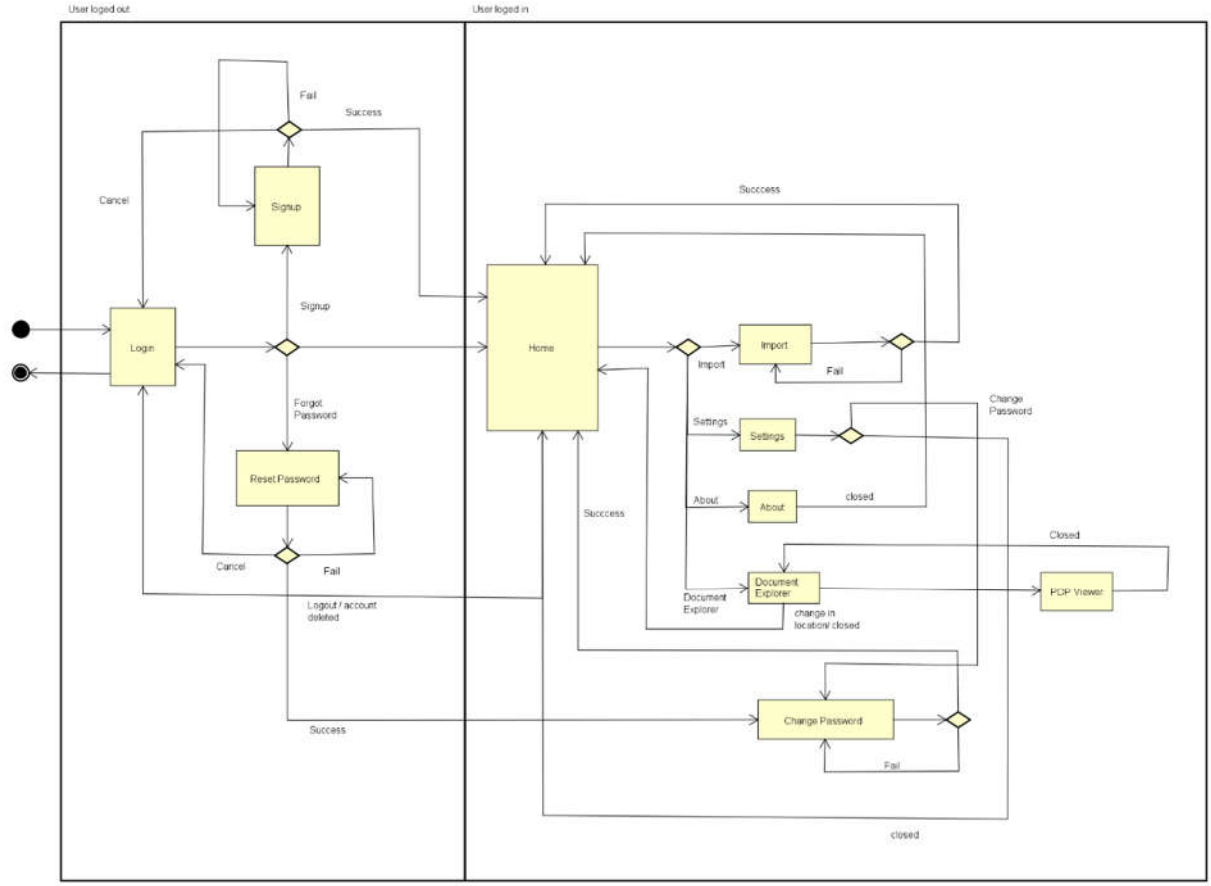


Figure 25: User flow diagram

7.1.2 User flow description

User start from login screen, which has 3 functionalities namely login, forget password (reset password) and signup. The user returns to login screen if they logout from application or cancel signup or rest process.
Once the login is successful the moves on to the home screen which is the central location that links to all system functionalities such as location map, brows file based on area, import document and menu with settings, help and about functions.
Setting have options the change password and to change user details. The user may wish to delete account from settings.
Document explorer is based on the area the user is currently in and the user may wish to view the currently available files or delete them.

7.2 UI designs

7.2.1 Login

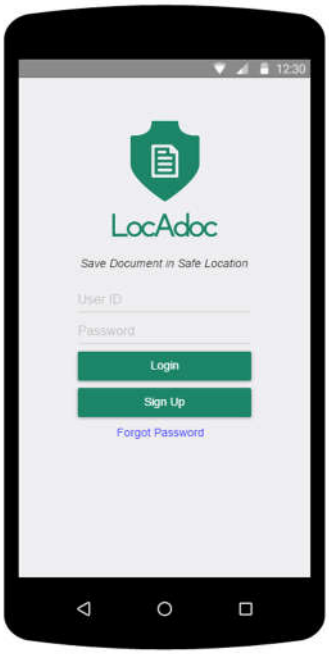


Figure 16: Login screen

Login screen will have user has the option to login, signup. He may also wish to create his own account using username and password. The login screen also leads to reset password feature.

7.2.2 Signup

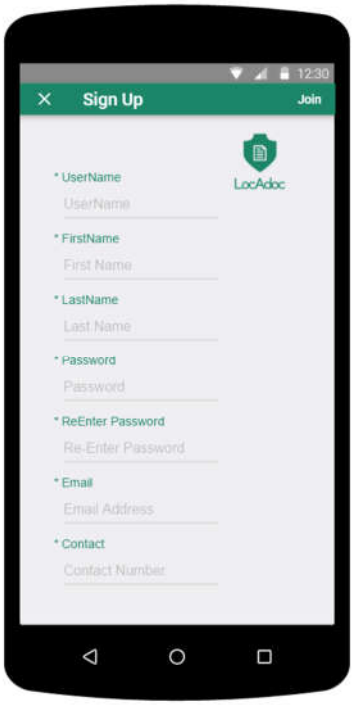


Figure 17: Signup screen

This is the signup form where the user can fill the above information and the system will signup the user for the services.

7.2.3 Reset password

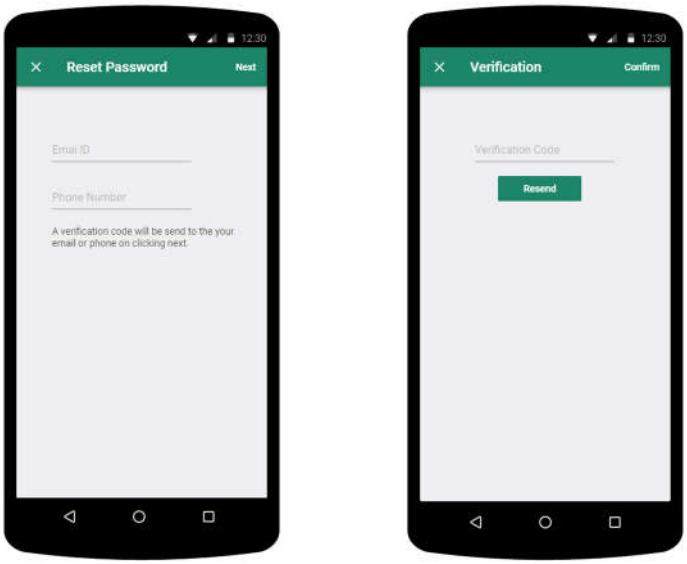


Figure 18: Reset password screens

The user will fill up the password reset form and on clicking next a verification code will be send to the email or phone. The second screen the user need to enter the verification code.

7.2.4 Home Screen

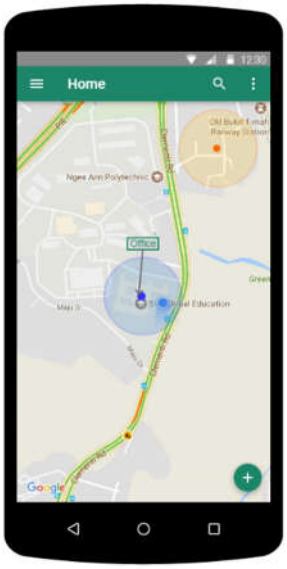


Figure 19: home screen

On logging in successfully the user will enter this screen where the user gets to see a map with his current location and the areas near him. The top bar has a menu icon on the left and a search bar.

He may click on the blue dot that represent the area to open up all the files in that location. The plus symbol below leads to adding a new document to the current location.

7.2.5 Home screen (searching area)

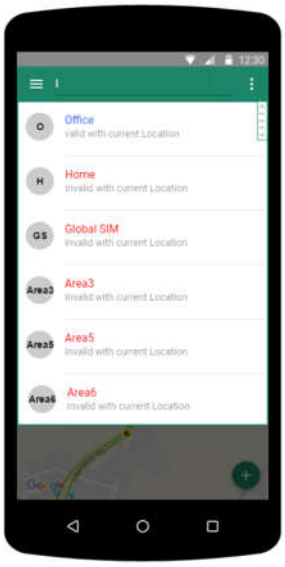


Figure 20: Search area

The user can search all the area from the search bar but can only access the files under that area if he/she is in that location.

7.2.6 Home screen (browser file)

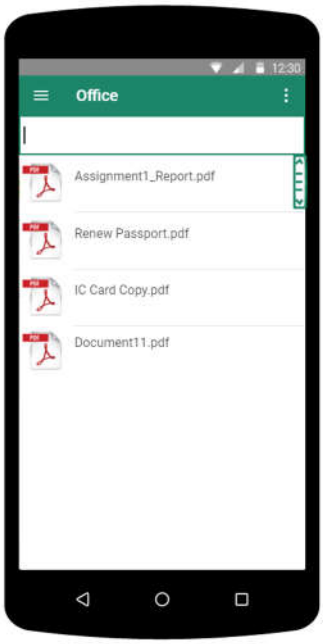


Figure 21: File browser

The app will list out all the files that is available in the current area. This screen can be approached by tapping the location pointer on the home screen map.

7.2.7 Home screen (file operations)

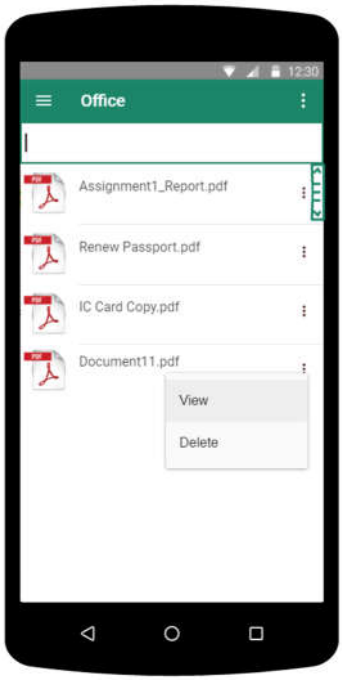


Figure 22:File browser file operation

On clicking the 3 dots the a menu will drop down asking if they wish to view the file or delete the file.

7.2.8 PDF viewer

Figure 23: PDF viewer screen

On clicking view in the former screen the application renders the PDF file and loads up on the screen.

7.2.9 Main Menu

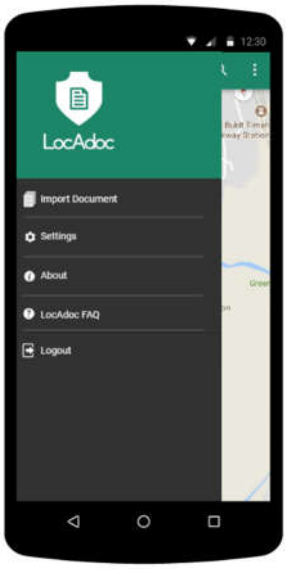


Figure 24: Main menu screen

Main menu loads up when we click the icon on the main activity. It has options to add a new document, settings, about, a FAQ link and a logout option.

7.2.10 Settings

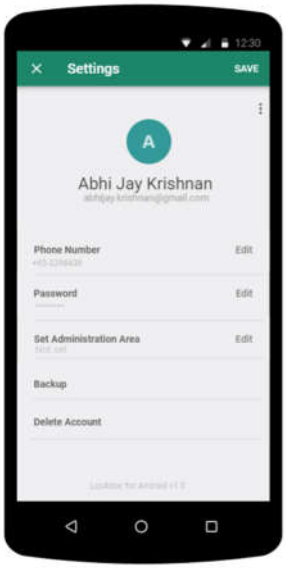


Figure 25: Settings screen

On this page, the user gets to edit his name and phone number. He may choose to back up his secured folder to AWS S3 by clicking on backup. The user may also wish to delete his account on which it will send the user back to login page. On clicking edit next to password will open a new activity.

7.2.11 Change password

Figure 26: Change password screen

The user can change password by filling up the above form and clicking confirm on the top.

7.2.12 About

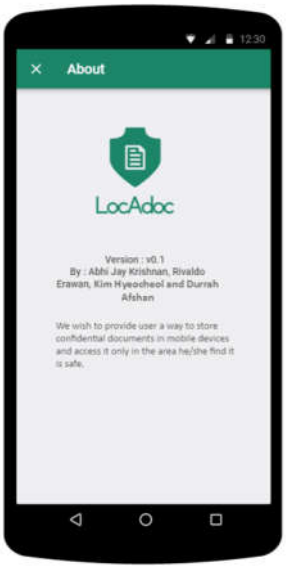


Figure 27: About screen

This activity can be reached from main menu and it describes the version of the application and credits.

8. System Testing

This test plan describes the testing approach and overall framework that will drive the testing of the LocAdoc system.
It describes:

- The distinctive features to be tested
- The test objective
- The result after testing
- The test environment

8.1 Test Plan Overview

This test plan will outline and define the strategy and approach taken to perform formal system qualification tests on LocAdoc app.

8.2 Objective

The objective of the test is to verify that the functionality of LocAdoc system works according to the specifications.

8.3 Approach

The test members will use Project Proposal and System Architecture Document to prepare the necessary test scripts and reports.
The testing phase is divided into 19 suites:

1. Sign up
2. Login
3. Instance ID Verification
4. Home page
5. PDF viewer
6. Import file
7. Add empty area
8. Area operations
9. File operation
10. Password recovery
11. Change password
12. Change name
13. Cloud storage
14. Change user
15. Delete user
16. File and data synchronization
17. GPS spoofing
18. Area explorer
19. File explorer

8.4 Black box testing

Black box testing is a software testing technique in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.
The advantages of black box testing are:

- Tests are done from a user's point of view and will help in exposing discrepancies in the specifications.
- The tester can be non-technical
- Test cases can be designed as soon as the functional specifications are complete.

- Tests can be conducted by a body independent from the developers, allowing for an objective perspective and the avoidance of developer-bias.

8.5 Features to be tested

This plan will execute specific test that exists to exercise the features provided and specified in the System Requirements Document of LocAdoc application.

8.5.1 Sign up

Test Objective	Test plan for Sign up form
Technique	Create tests for each input field to verify if the Signup function is working.
Completion Criteria	Sign up must be successful only upon entering valid values in all the fields.
Special Considerations	NIL

Table 11: Sign Up Testing

8.5.2 Login

Test Objective	Test plan for login form
Technique	Perform a test on input validation and authentication. Check added delay timer on 3 invalid tries.
Completion Criteria	Login must be successfully completed, and delay timer should slow down any adversary from brute forcing the password.
Special Considerations	NIL

Table 12: Login Testing

8.5.3 Instance ID Verification

Test Objective	Test if the Instance ID is verified and the system logout on change.
Technique	Login to two devices to check if the instance ID is working.
Completion Criteria	The first device should logout when logged into second device.
Special Considerations	NIL

Table 13: Instance ID Verification Testing

8.5.4 Home page

Test Objective	To test if all the user interface components are responsive.
Technique	Try out various features (menu, search bar, floating action button) of homepage one by one.
Completion Criteria	If all homepage are features are responsive.
Special Considerations	NIL

Table 14: Home page Testing

8.5.5 PDF Viewer

Test Objective	To test if the PDF viewer renders the PDF file and close on moving out of the location.
Technique	The test was conducted by importing a pdf file and moving out of the current area
Completion Criteria	If the pdf file is properly rendered and close on moving out of the designated area.
Special Considerations	NIL

Table 15: PDF Viewer Testing

8.5.6 Import File

Test Objective	To test if the files are imported successfully
Technique	Try out various scenarios of importing file that the user may end up performing.
Completion Criteria	If the files are imported and can be opened successfully. Check if files are destroyed if the user choose to empty them. The app should block invalid inputs.
Special Considerations	NIL

Table 16: Import File Testing

8.5.7 Add empty area

Test Objective	To test if an empty area can be created.
Technique	By creating empty using deferent configuration and see if the area is created
Completion Criteria	If the area can be created successfully and if the app blocks the invalid inputs.
Special Considerations	NIL

Table 17: Add empty area Testing

8.5.8 Area operations

Test Objective	To check if area related operations work.
Technique	By changing radius and deleting 2 areas one with no file and one with file.
Completion Criteria	The radius should change dynamically. The app should not allow deletion of file with a file and should delete an area with no file successfully. If the user is not currently in within the radius of the area he cannot perform area related operations.
Special Considerations	NIL

Table 18: Area operations Testing

8.5.9 File operations

Test Objective	To test if the file related operations function as per the requirement.
Technique	Each file loaded into one are moved or copied to another area. Finally, the files are deleted.
Completion Criteria	If all file operation works according to requirement.
Special Considerations	NIL

Table 19: File operations Testing

8.5.10 Password recovery

Test Objective	To test if the password is reset success fully
Technique	By clicking forget password and trying out various invalid and valid inputs.
Completion Criteria	Invalid inputs should be rejected, and valid inputs should lead to successful recovery.
Special Considerations	NIL

Table 20: Password recovery Testing

8.5.11 Change password

Test Objective	To test if the password can be changed successfully
Technique	Try various valid and invalid inputs and try change password.
Completion Criteria	If the user is blocked on invalid ties and valid tries lead to successful login.
Special Considerations	NIL

Table 21: Change password Testing

8.5.12 Change name

Test Objective	To test if the user name can be changed successfully
Technique	Try input valid and invalid inputs.
Completion Criteria	The application should block invalid inputs and allow valid onces
Special Considerations	NIL

Table 22: Change name Testing

8.5.13 Cloud storage

Test Objective	To test if the cloud storage operations
Technique	Added, delete, copy files in S3. Try upload files above storage limit.
Completion Criteria	All operations should work as per the requirement and the application should block the user from uploading files once the storage limit is hit.
Special Considerations	NIL

Table 23: Cloud storage limit Testing

8.5.14 Change user

Test Objective	To test if another user can login.
Technique	Clicking change user and try login using a new user account.
Completion Criteria	All data and area should be set asper the new account.
Special Considerations	NIL

Table 24: Change user Testing

8.5.15 Delete user Testing

Test Objective	To test if the user can delete his account
Technique	The delete user functionality is executed on login and try to re-login using same deleted account.
Completion Criteria	If the re-Login fails and if all the user record is deleted from both Cognito, DynamoDB and S3.
Special Considerations	NIL

Table 25: Delete user Testing

8.5.16 File and data synchronization Testing

Test Objective	To test if the files and data are synchronized over multiple devices.
Technique	Login to one device and add a file, then login to second devise using same account and see if the file can be viewed.
Completion Criteria	If the files and user data are consistent over multiple devices.
Special Considerations	NIL

Table 26: File and data synchronization Testing

8.5.17 GPS spoofing Testing

Test Objective	To test if the GPS spoofing can be detected.
Technique	Installing a GPS spoofing application in the phone and setting mock location. Then try login to the application.
Completion Criteria	The app should logout if the
Special Considerations	NIL

Table 27: GPS spoofing Testing

8.5.18 Area explorer

Test Objective	To check if the area based access control mechanism works as per the requirement
Technique	Add new area and see if the area is visible by moving in and out of the radius.
Completion Criteria	If the access control mechanism works based on the requirement.
Special Considerations	NIL

Table 28: Area explorer testing

8.5.19 File explorer

Test Objective	To test if the files are accessible based on location
Technique	Try open file with the designated area and outside designated area.
Completion Criteria	The file should not be accessible outside the designated area.
Special Considerations	NIL

Table 29: File explorer testing

8.6 Item Pass/Fail Criteria

This section specifies the Pass/Fail criteria for the tests covered in this plan. The test items detailed above act as the targets of this plan, which will be tested for the LocAdoc application.
The system will be deemed to have passed testing if:

- All tests defined have been executed, and
- The number of tests executed without any defects is more than 95% of the total, and
- Any defects detected have a severity classification of Low.

The system will be deemed to have failed testing if:

- The number of test executed with defects is more than 5% of the total, and
- There are defects with a severity classification of High.

8.7 Test Deliverables

The following documents will be generated by the test member and will be created after test completion.

S. No.	Deliverable Name	Author	Reviewer
1.	Test Plan	Testing team	
3.	Test Summary Report	Testing team	

Table 30: Deliverables

8.8 Test Environment

We tested this app on 4 devises all running different hardware made within past five years (2013 to 2017). The test was conducted by keeping following things in mind: -

- Processing power.
- Android version.
- Screen size.
- Network and GSP connectivity hardware.

The phones used are :-

- **Moto G (1st Generation)** – Released in 2013 with 1Gb of RAM and Quad-core 1.2 GHz Cortex-A7 processor. It runs android version 5.1.1(Lollipop) and considered to be the lowest configuration required to run this app.
- ***Sony Xperia Z5*** – Released in 2015 3GB of RAM and Octa-core (4x1.5 GHz Cortex-A53 & 4x2.0 GHz Cortex-A57, Qualcomm MSM8994 Snapdragon 810 chipset) processor. It runs android version 7.1.1 (Nougat).
- **Samsung galaxy Note 5** - Released in 2015 with 4GB of RAM and Octa-core (4x2.1 GHz Cortex-A57 & 4x1.5 GHz Cortex-A53, Exynos 7420 Octa chipset) processor. It runs android version 7.1.1 (Nougat).
- **Samsung galaxy S8** - Released in 2015 with 4GB of RAM and Octa-core (4x2.3 GHz & 4x1.7 GHz, Exynos 8895 Octa). It runs android version 7.1.1 (Nougat).

8.9 Test Summary Report

In Total 128 cases tested and all passed

8.9.1 Conclusion

After conducting 129 tests there is only one issue that is when is network connection is unstable during the login process or when changing password. This scenario may lead to app crashing. 128 (99.2%) testcases have generated a PASS.

8.9.2 Problems faced

The main difficulty faced is when there is poor network connection or GPS connection. The tester was setting small radius and see if the pdf file closes when he walks out of the specified radius, but the

location update moves is not consistent with the testers current location. It requires him to wait few seconds before the location update catch up with him.

8.9.3 Improve Test Assets.

The purpose of this activity is to maintain and improve test assets. In our case, test cases that can be re-used are:

- Login
- Signup
- Settings

8.9.4 Achievements

We have achieved a 99.2% pass in the test case phase. The one test case that failed is due to week network connection or hard ware level issues which cannot be controlled or simulated. All the errors faced have been rectified during the construction phase and test confirmed that all functionalities are working according to the requirement. We have also managed to do small tweaks to improve performance.

9. Marketing Plan

LocAdoc is an application which provides user a way to store confidential documents in mobile devices and access it only in the area he/she find it is safe.

Segmenting the audience will help us focus toward those people who most need the file locking app. Groups listed in the primary audiences are those firms who will need this app for confidentiality of their documents.

Primary:

1) Employees working in defense industry may have to handle highly secretive documents that should not be taken of the secure premises.
2) Governmental authorities may wish to keep their confidential documents within the country or within the restricted area.

Secondary:

1) General public may want to store their personal information and keep it within their safe zone such as their home.

To market the application, we can use Strategies like:

1) Develop information pieces that appeal to users and share the content on various social networking sites like Twitter, Facebook, Google+, LinkedIn, etc., several times in a week.
2) Providing information that will help current users promote the application to non-users. For eg: Tell a Friend Campaign.
3) Communicate the difference that our app makes in the community.
4) Run a contest. For example, encouraging people to tweet and share content on their social networks promoting the app. A random person every week can win a promo code to download the app for free.
5) Consider offering a promotional price, consider a promotional pricing of $0.99 during the launch time. This price point encourages impulse purchases.
6) Always be collecting emails. Gather potential customer emails through Facebook, Twitter or your website. An email list of people who have opted in is a powerful marketing tool.
7) Get speaking opportunities. Research events that touch on the topic of the app addresses and make an appearance. Start small and get recognized as an authority in the space.

We aim to make our product more affordable. The estimated price for 1GB storage and access documents in multiple device is S$12 per year.

10. Future Enhancements

Currently the app is based on individual usage, we have come up with an idea of converting the app to enterprise level. The enterprise edition will have another front end for the administrator who can designate files and location for substantial number of users. For example, A admin at the military camp may a lot some confidential documents that is to be used with the camp premises and the military personals will have the freedom to move around with many' documents accessible through a small a compact device.

Secondly, we can incorporate more file types and add more viewers within the application. This will improve the user experience.

Lastly there can be a payment system to increase the storage space. This will help the user to scale as per the demand.

11. Conclusion

Overall LocAdoc was developed around the problem "Read me where you left me". The application has covered all the functionalities stated in the Software requirement sheet. There are more ideas that could have been implemented (refer to section 12) but the current functionalities provide a complete solution to the customer.

The application does exactly what is stated in the problem description and gives additional level of security using Cognito Oath flow. It also provides additional features such as secure cloud storage and synchronization of documents over multiple devises.

12. Bibliography

[1] dSwizz, "SecureSafe," dSwizz, [Online]. Available: https://www.securesafe.com/fr/. [Accessed 2017 July 23].

[2] Passible, "Home page," Passible, 10 Oct 2016. [Online]. Available: http://www.passible.com/#security. [Accessed 23 July 2017].

[3] M. Wojas, "File Locker," Marcin Wojas, 6 May 2014. [Online]. Available: https://play.google.com/store/apps/details?id=com.mwgo.filelocker. [Accessed 23 July 2017].

[4] Innorriors, "File locker - Lock any File," Innorriors, 15 June 2017. [Online]. Available: https://play.google.com/store/apps/details?id=inno.filelocker. [Accessed 23 July 2017].

[5] Legendary Software Labs LLC, "Private Photo Vault," Legendary Software Labs LLC, 11 July 2017. [Online]. Available: https://play.google.com/store/apps/details?id=com.enchantedcloud.photovault. [Accessed 23 July 2017].

[6] "What is Waterfall model- advantages, disadvantages and when to use it?," ISTQB EXAM CERTIFICATION, [Online]. Available: http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/.

[7] ISTQB EXAM CERTIFICATION, "What is Prototype model- advantages, disadvantages and when to use it?," ISTQB, [Online]. Available: http://istqbexamcertification.com/what-is-prototype-model-advantages-disadvantages-and-when-to-use-it/.

[8] I. E. CERTIFICATION, "What is Agile model – advantages, disadvantages and when to use it?," ISQTB, [Online]. Available: http://istqbexamcertification.com/what-is-agile-model-advantages-disadvantages-and-when-to-use-it/.

[9] P. Christensson, "RUP," TechTerms, 2006. [Online]. Available: https://techterms.com/definition/rup. [Accessed 25 July 2017].

[10] J. Vincent, "99.6 percent of new smartphones run Android or iOS," 16 Feb 2016. [Online]. Available: https://www.theverge.com/2017/2/16/14634656/android-ios-market-share-blackberry-2016.

[11] P. Khatri, "Native vs Cross-Platform App Development: Pros and Cons of PhoneGap, Titanium, and Xamarin," Mobile Zone, 08 March 2017. [Online]. Available: https://dzone.com/articles/native-vs-cross-platform-app-development-pros-and.

[12] J. Serra, "Relational databases vs Non-relational databases," James Serra's Blog, 27 August 2015. [Online]. Available: http://www.jamesserra.com/archive/2015/08/relational-databases-vs-non-relational-databases/. [Accessed 30 July 2017].

[13] R. babu, "NoSql vs Relational database," stackoverflow, [Online]. Available: https://stackoverflow.com/questions/4160732/nosql-vs-relational-database. [Accessed 30 July 2017].

[14] D. L. Soltesz, "What are the Advantages of a Relational Database Model?," Techwalla, [Online]. Available: https://www.techwalla.com/articles/what-are-the-advantages-of-a-relational-database-model. [Accessed 30 July 2017].

[15] A. Gajani, "The key differences between SQL and NoSQL DBs.," monitis, [Online]. Available: http://www.monitis.com/blog/cc-in-review-the-key-differences-between-sql-and-nosql-dbs/. [Accessed 30 July 2017].

[16] "Relational Vs Non Relational Database," mongoDB, [Online]. Available: https://www.mongodb.com/scale/relational-vs-non-relational-database. [Accessed 30 July 2017].

[17] M. Ramachandran, "Relational Vs Non-Relational databases – Part 1," BIG DATA MADE SIMPLE, 23 April 2014. [Online]. Available: http://bigdata-madesimple.com/relational-vs-non-relational-databases-part-1/. [Accessed 30 July 2017].

[18] M. Obijaju, "NoSQL NoSecurity – Security issues with NoSQL Database," Perficient, 22 June 2015. [Online]. Available: http://blogs.perficient.com/dataanalytics/2015/06/22/nosql-nosecuity-security-issues-with-nosql-database/. [Accessed 30 July 2017].

[19] "AWS Free Tier," Amazone Web servises, [Online]. Available: https://aws.amazon.com/?nc2=h_lg.

[20] "Azure free account," Microsoft Azure, [Online]. Available: https://azure.microsoft.com/en-us/free/?v=17.15.

[21] "Google Cloud Platform Free Tier," Google Cloud Platform , [Online]. Available: https://cloud.google.com/free/.

[22] "All Customer Success Stories," Amazon Web Sevices, [Online]. Available: https://aws.amazon.com/solutions/case-studies/all/.

[23] Amazon, "AWS Mobile SDK," Amazon, [Online]. Available: https://aws.amazon.com/mobile/sdk/.

[24] Amazon, "What is Amazon Cognito?," Amazon, [Online]. Available: http://docs.aws.amazon.com/cognito/latest/developerguide/what-is-amazon-cognito.html.

[25] Amazon, "Amazon DynamoDB," Amazon, [Online]. Available: https://aws.amazon.com/dynamodb/.

[26] Amazon, "Amazon S3," Amazon, [Online]. Available: https://aws.amazon.com/s3/?sc_channel=PS&sc_campaign=acquisition_SG&sc_publisher=google&sc_medium=s3_b&sc_content=s3_e&sc_detail=amazon%20s3&sc_category=s3&sc_segment=82556503449&sc_matchtype=e&sc_country=SG&s_kwcid=AL!4422!3!82556503449!e!!g!!amazon%20s3.

[27] "Visual BSD," [Online]. Available: http://visualxsd.com/.

[28] Amazone, "User Sign-in," Amazone Web Services , [Online]. Available: https://docs.aws.amazon.com/mobile-hub/latest/developerguide/user-sign-in.html.

[29] Amazone, "User Data Storage," Amazone Web Services, [Online]. Available: http://docs.aws.amazon.com/mobile-hub/latest/developerguide/user-data-storage.html.

[30] Amazon, "NoSQL Database," Amazon Web Services, [Online]. Available: http://docs.aws.amazon.com/mobile-hub/latest/developerguide/nosqldb.html.

[31] Amazon, "What Is IAM?," Amazon Web Service, [Online]. Available: http://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html.