**Introduction**

Many vets have difficulty accessing some of their repetitive tasks, particularly recording data, because they must rely on hard copies. A system which is often fragmented, unreliable and inefficiently operated. Lack of proper framework is the leading obstacle to meeting mobility needs of the people who need the services the most. The purpose of this deployment is to replicate and advance the success by providing an efficient filing system to leverage other activities inclusive of decision making. Goals are to use service technology integration to:

* Increase mobility and accessibility.
* Achieve more efficient use of service, that is do more with less.

**Constraints**

Timeline-the fixed timeline of ten weeks could lead to burn out and hamper with creativity, this will have an effect on the design process.

Device specific-The pet state application will only be available to devices running on android operating system with at least 2GB RAM and 16GB storage.

Cost-during design some of the design tools require premium membership so as to be applied on the project. Premium membership incurs additional cost.

**Architecture of the system**

Data Flow Diagram

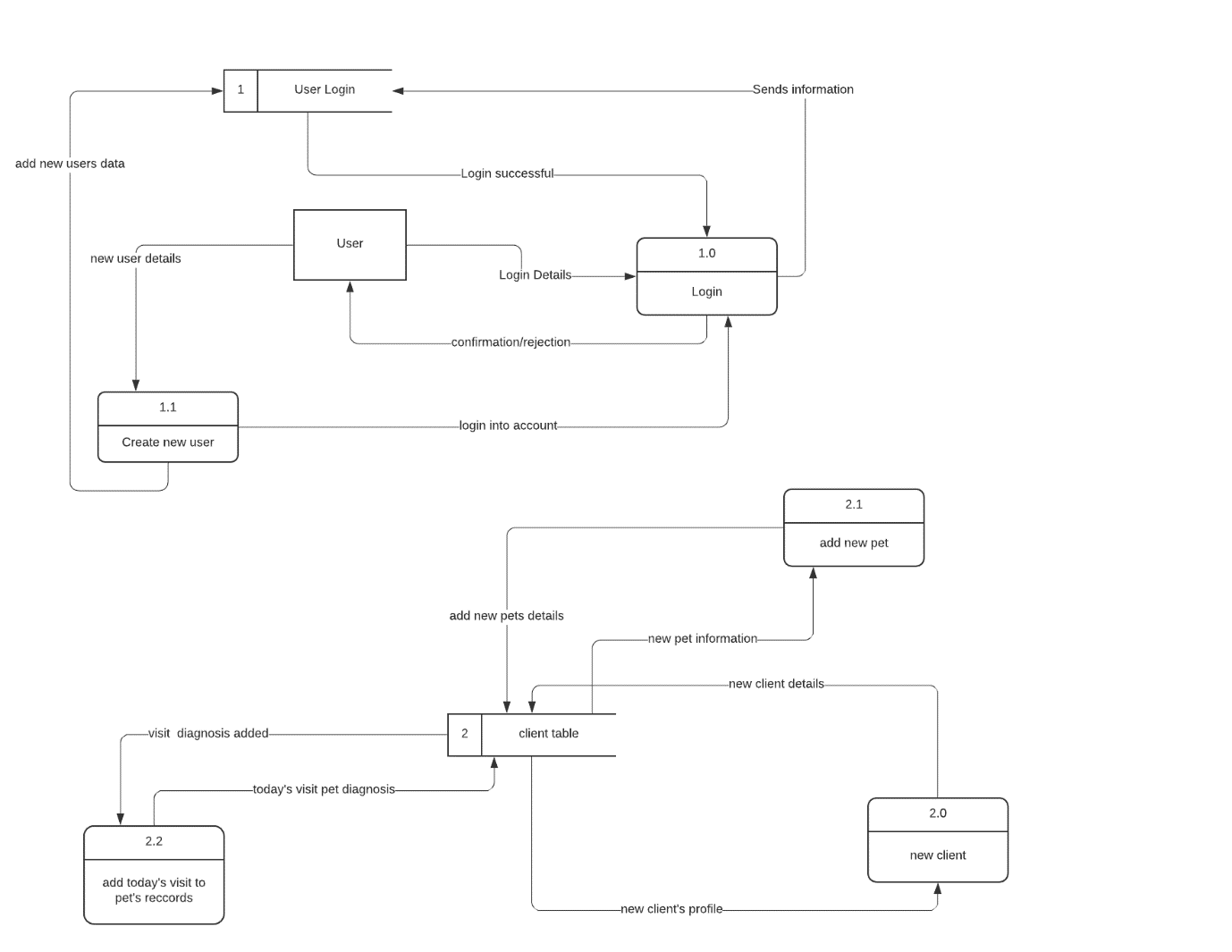


Figure 1(Data Flow Diagram for Petstate project)

PACKAGE DIAGRAM

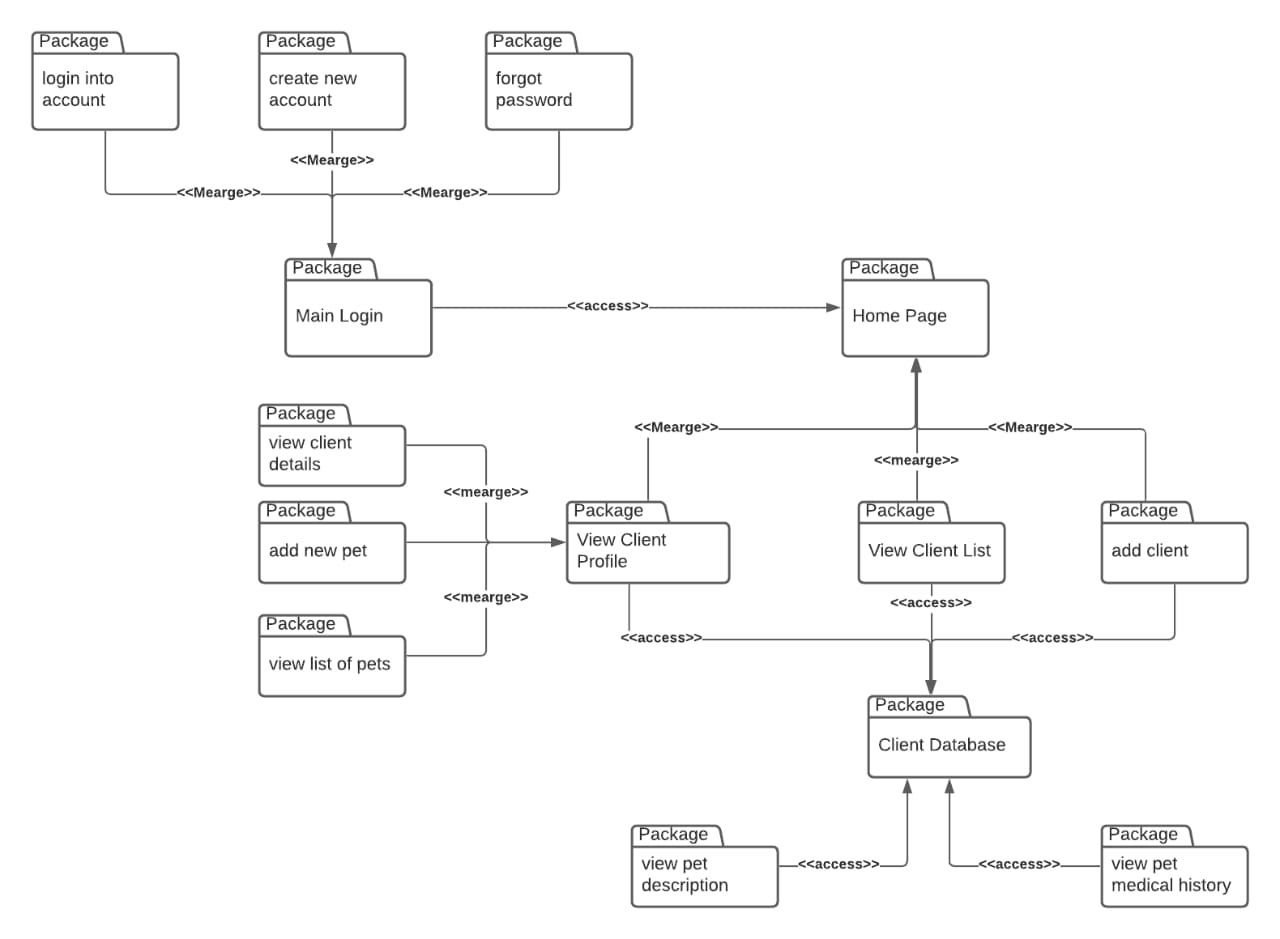


Figure 2(Package diagram for Petstate project)

COMPONENT DIAGRAM

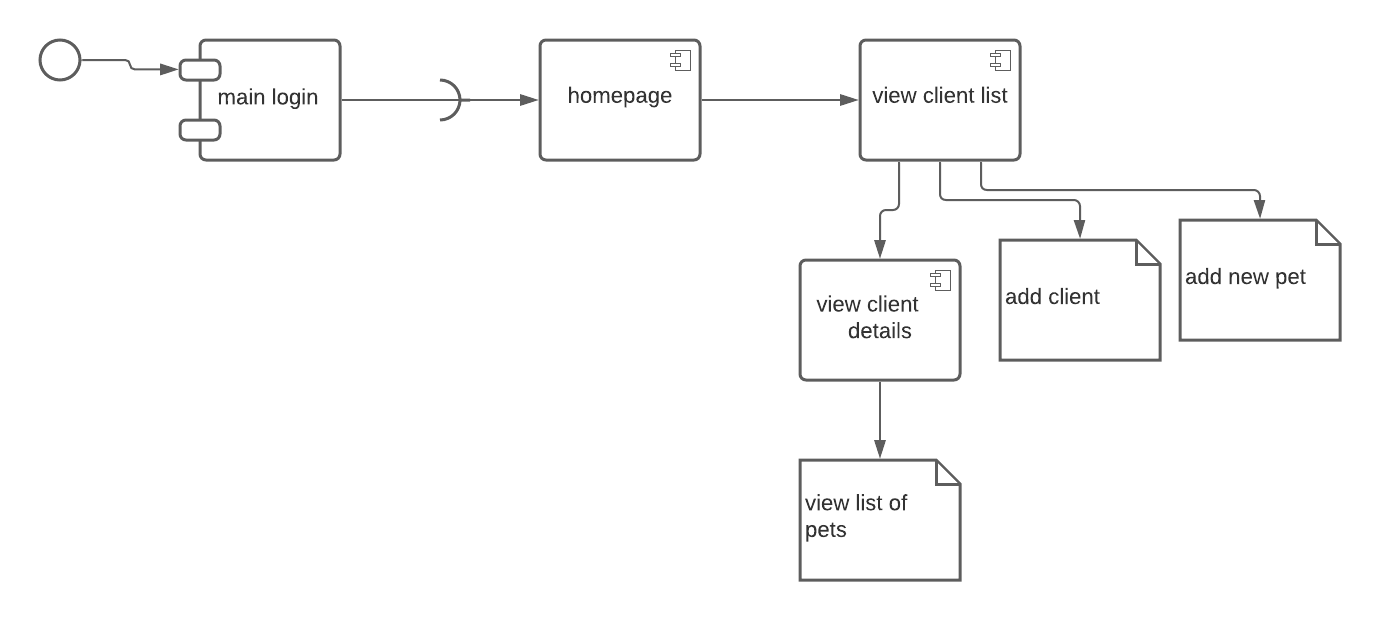


Figure 3(Component Diagram for Petstate project)

DATABASE DESIGN

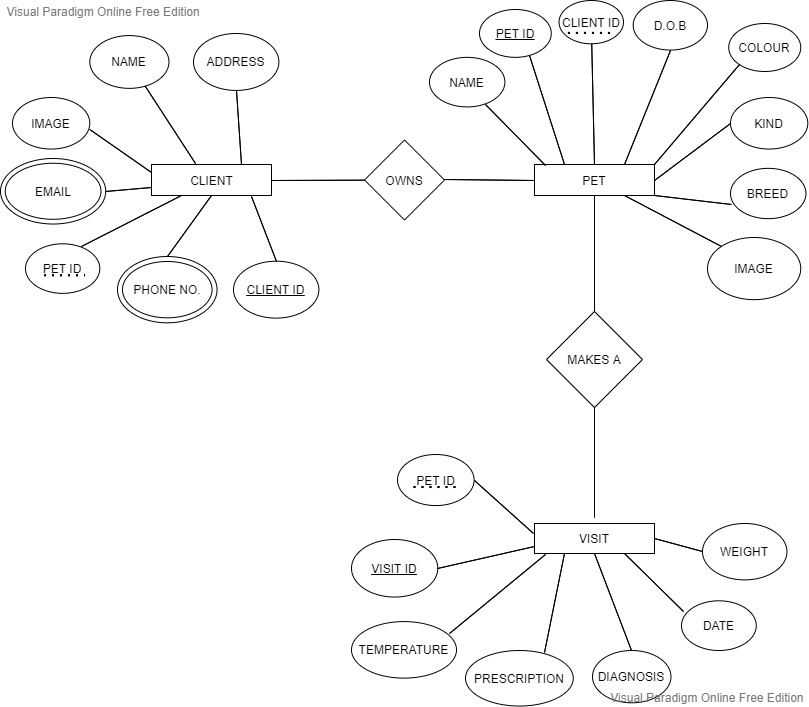


Figure 4(Chen-Type ERD for Petstate project)

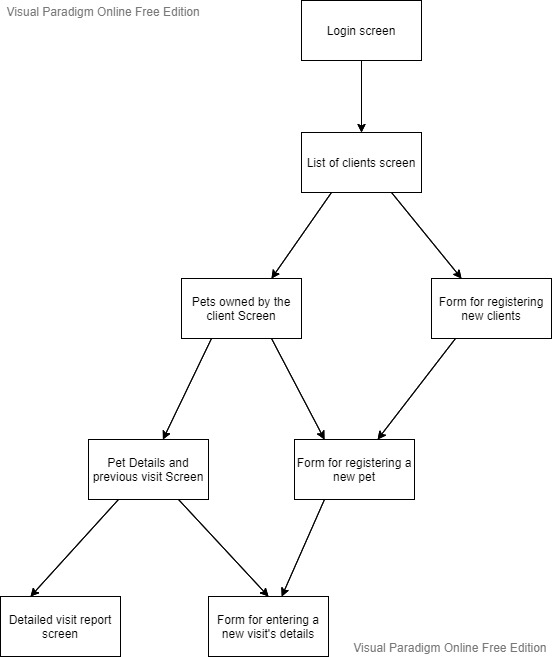
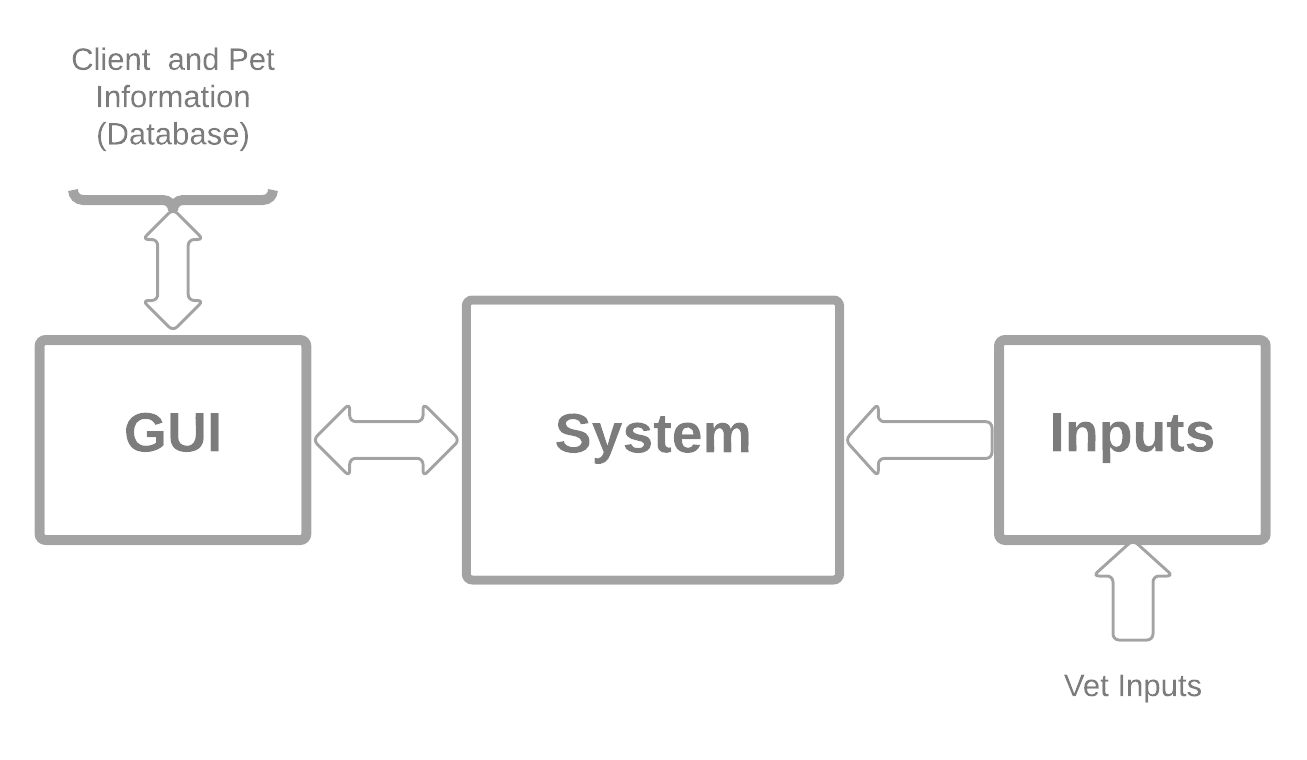


Figure 5(Human-Machine Interface Design for Petstate)

**Human Machine Interface**

1. **Inputs** – The system accepts data inputs entered by the Vet. The inputs include:
2. Client Information – an avatar to identify the client, full name of the client, email address and phone number and their physical address
3. Pet Information – an avatar to identify the pet, the name of the pet, breed, colour, kind of the pet and its date of birth.
4. Visit information about a pet, which includes, the measured weight of the pet during visitation, its temperature at the time of diagnosis as well as diagnosis details and prescription.
5. **Outputs** – the system generates reports on the pet information stored in the system database. The reports are generated on the last day of each month. The reports include:
6. List of pets that have been checked each month
7. Number of pets from a specific geographical region
8. Graphs comparing the kinds of pets seen by the vet in a period of one month



**Detailed design**

Kotlin is the primary language that will be used in designing since:

1. It is a modern statically typed programming language used by android developers that helps boost productivity, developer satisfaction and code safety.
2. The IDE (android studio) fully supports Kotlin
3. Kotlin is a cross-platform, general-purpose programming language with type inference. It is designed to interoperate fully with Java, and the JVM version of Kotlin's standard library depends on the Java Class Library, but type inference allows its syntax to be more concise.

Minimum API level 19 (kitkat 4.4) and above allows a 99.7% audience. The target is API level 29 along with build tools, revision 28.0.3.

The build tool component is updated by Android Gradle Plugin which is integrated into the IDE.

Gradle is the main building tool that compiles the code since it automatically resolves external dependencies.

**Software Detailed Design**

Among the exiting login providers we will be using Google’s firebase which is a cloud platform.

Reasons as to why we will be using firebase include:

1. It is easily integrated with android, IOS, unity and c++
2. It is easy to monitor performance and stability
3. Has a fully managed backend infrastructure

**Application activities**

1. Login Activity
2. Sign up Activity
3. Forgot password Activity
4. Main Activity
5. Add Client Activity
6. View Client’s Pets Activity
7. Add Pet Activity
8. View Pet’s visits Activity
9. Add Visit Activity
10. View Visit Activity
11. Update Visit Activity
12. **Login activity**

The user logs in after the credentials are authenticated. Authentication data is handled by firebase backend infrastructure.

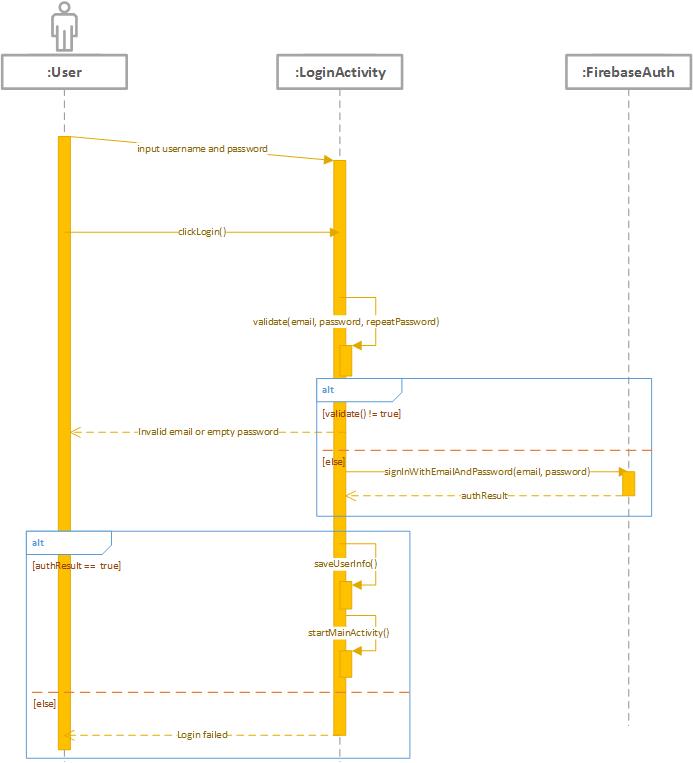


Figure (login sequence diagram)

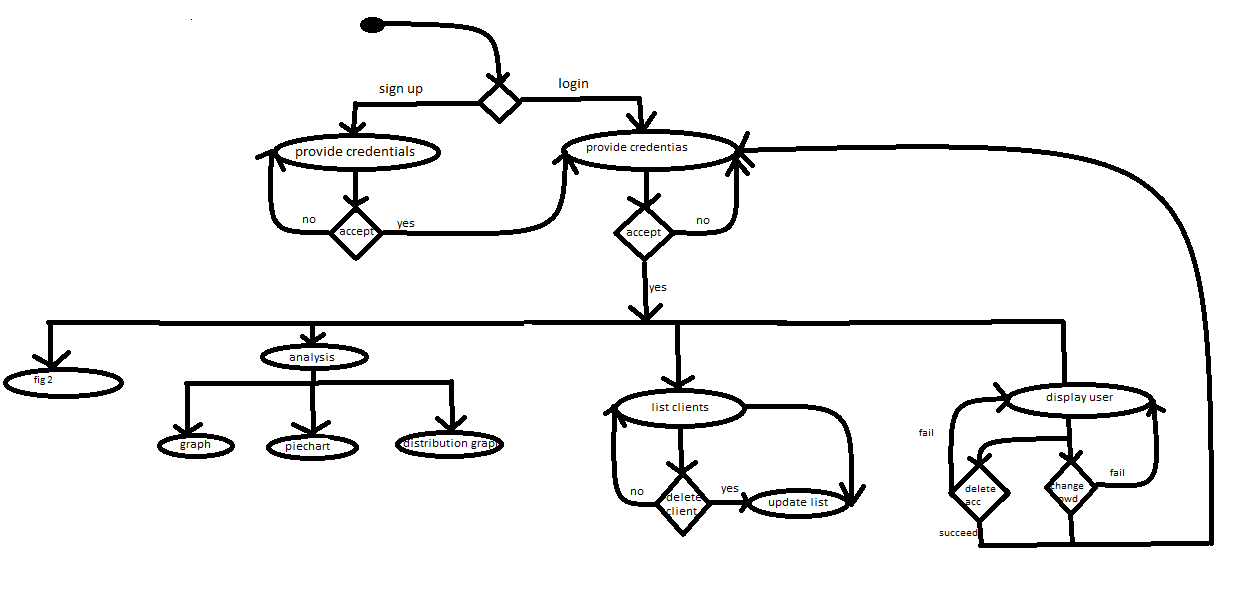
1. **Signup activity**

This happens when a user does not have an account. Users provide their signup credentials which are stored by firebase. The signup sequence is similar to the login sequence except that the credentials are stored as the first time data and therefore no wrong credentials exception.

1. **Main activity**

The main activity displays a list of existing clients showing their name, profile picture and their address. There also exists a Floating Action Button that when clicked displays three others which lead to Add New Client Activity, Analysis Activity and an Edits Activity.

Activity diagram



1. **Add Client Activity**

This activity enables the vet to save the personal details of the pet owner for future communication and pet identification. The details saved are: client name, phone number, email address and physical address.

1. **View Client’s Pets Activity**

This activity shows the fine details of the client (their name, profile picture, email address, phone number) as well as a list of pets they own. There is also a button to add a new pet for the same client.

1. **Add Pet Activity**

This activity enables the vet to enter the details of an individual pet owned by the client such as: the pet’s name, date of birth, kind, breed, colour and gender.

1. **View Pet’s Visits Activity**

Here, the fine details of an individual pet are shown, that is, their description and a list of visits they have attended in the past. There is also a button to add a new visit.

1. **Add New Visit Activity**

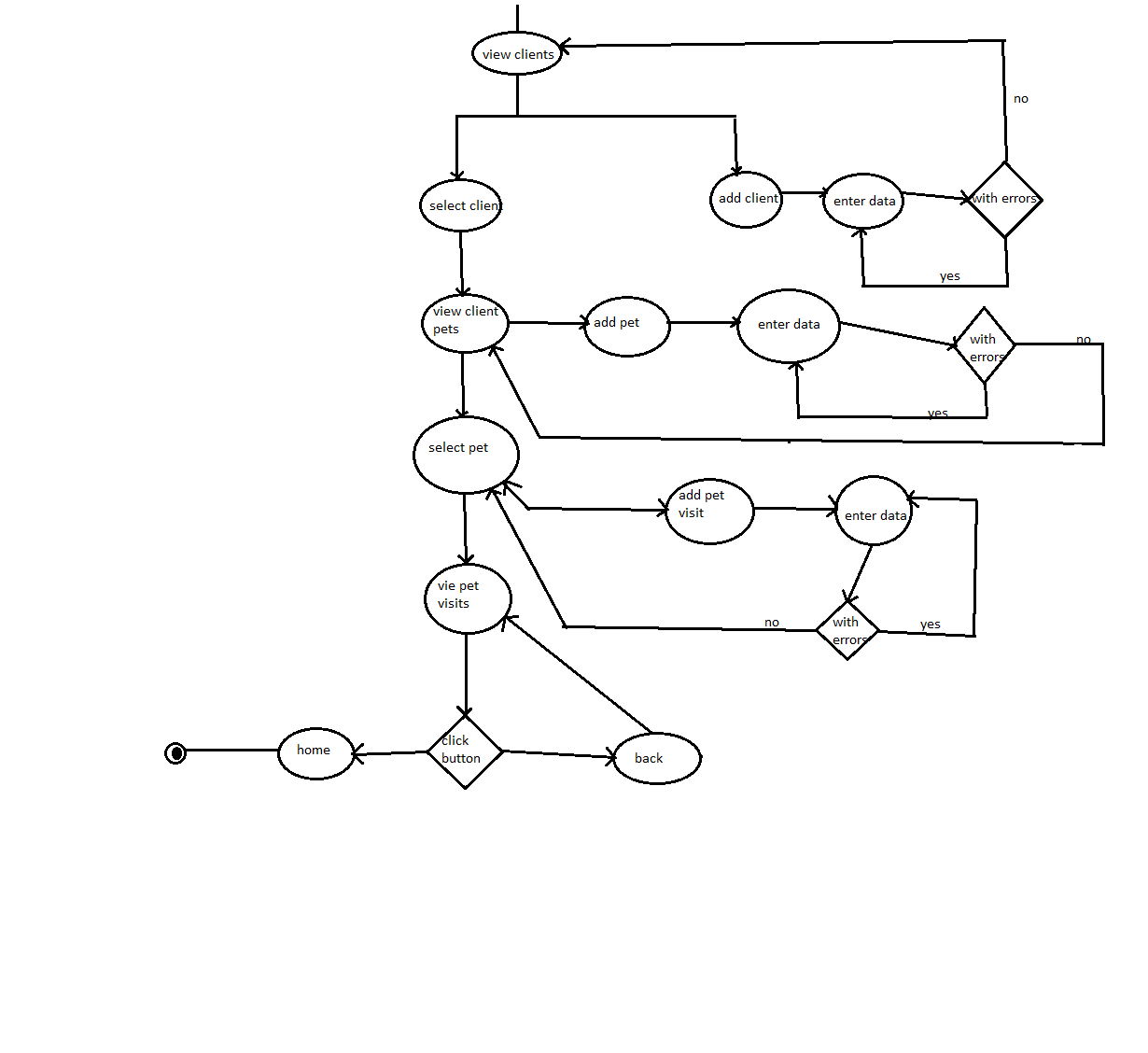
In this activity, a vet is able to insert the details of a checkup, such as the temperature, weight, diagnosis and prescription for the pet.

1. **View Visit Activity**

In this activity, the details of a past check-up are shown, including the weight, temperature, diagnosis and prescription. There is also an update button to change these details.

1. **Update Visit Activity**

Here, the vet is able to change the details of a particular visit in case of any necessary changes.

**Fig 2**

**Security Detailed Design**

Security is achieved through caching of user credentials in the mobile secure key store.

Authentication by firebase framework is achieved through secure communication to the server over an encrypted channel. The application is signed with key where signature is in SHA-1 format.

Once the communication channel is established and identity of the application is verified, the data transmission is initiated.

Authorization is maintained during account creation. Each account is only allowed to read and write every database in their domain.

Execution privileges are only allowed for the developer.

Auditing is supported by verbosely logging the trail to storage/emulated/[user ID].

Intrusion Detection System is powered by firebases server against various attacks. Once an attacker tries and is detected the fingerprint is blacklisted and cannot connect again.

**Performance**

The heap size is limited due to the little dependencies and a small file size. The application utilizes a RAM capacity of approximately 80 mbs.

Though a storage as a service is implemented, availability is guaranteed since the connection is not mandatory and data will be synchronized only when connection is available else cached locally in internal storage.

In the application manifest by setting allowbackup=TRUE allows back up of the application state alongside system backup.

In the event of a re-install the data is restored and user can continue working with previous clients easily.

**Communication**

For internal I/O functions an interface class exists that spawns across the entire Application domain.

By inheriting from the class R (a superclass) data from every interface can be easily accessed.

UI components before compilation are coded/structured under the XML format. The resource file tells the system where and how the window manager positions the user interface on the screen with varying display sizes.

The device modem provides a channel for communication between server and the application.

Therefore a catastrophic failure would occur if the device modem would be faulty or unavailable. A connection is required to provide authentication and fetch data in-case of a re-installation of the application.