

AYA

PROJECT REPORT

*Submitted in partial fulfillment of the requirements
For the award of the degree of*

BACHELOR OF COMPUTER SCIENCE

MAHATMA GANDHI UNIVERSITY

KOTTAYAM-686560

2018-2021



SUBMITTED BY

AJAY P S

REG NO: 180021029489

Under the guidance of

MS. CHINJU KURIAKOSE

DEPARTMENT OF COMPUTER SCIENCE

AL AMEEN COLLEGE, EDATHALA

AYA

PROJECT REPORT

*Submitted in partial fulfillment of the requirements
For the award of the degree of*

BACHELOR OF COMPUTER SCIENCE

MAHATMA GANDHI UNIVERSITY

KOTTAYAM-686560

2018-2021



SUBMITTED BY

AJAY P S

REG NO: 180021029489

Under the guidance of

MS. CHINJU KURIAKOSE

DEPARTMENT OF COMPUTER SCIENCE

AL AMEEN COLLEGE, EDATHALA



AL-AMEEN COLLEGE

EDATHALA, ALUVA

DEPARTMENT OF COMPUTER SCIENCE

CERTIFICATE

*This is to certify that the project entitled “AYA” is a bonafide record of the work done by **Ajay PS** with **Register Number 180021029489**, of Department of Computer Science, Al-Ameen College Edathala in partial fulfillment of the requirements for the award of the degree of Bachelor of Computer Science by **Mahatma Gandhi University, Kottayam** during the year 2018-2021.*

Project Guide

Head of the Department

Submitted for the university exam held on

Internal Examiner

External Examiner



An ISO 9001:2015 Certified Company



July 2, 2021

TO WHOM SO EVER IT MAY CONCERN

This is to certify that **AJAY .P.S** 6th semester **BSC COMPUTER SCIENCE** student of **AL AMEEN COLLEGE, EDATHALA** has successfully completed a project titled “**AYA USING ANDROID**” from our organization.

The duration of the project was for 3 months. The Project was incorporated in **ANDROID** and was implemented successfully.

Thanking you,

For **LCC Computer Education**

A handwritten signature in black ink, appearing to read "Ramaswamy", with a stylized flourish at the end.

Ramaswamy
Director



Alwaye Office : 2nd Floor, Gold Fort Buildings, Bypass Junction, Alwaye 683101 Phone Off : +91 484 2629996 E-mail: lccalwaye@gmail.com

ACKNOWLEDGEMENT

I am using this opportunity to express my gratitude to everyone who supported me throughout the course of this project. I am thankful for their aspiring guidance, invaluable constructive criticism and friendly advice during the project work. I am sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project.

I express my deep and sincere gratitude to **Dr. Cini Kurian**, Principal, Al-Ameen College, Edathala for the overwhelming support and direction.

I express my special thanks to **Dr. Leena Varghese**, IQAC coordinator for all the support and kind cooperation to complete the project.

I am deeply indebted to **Ms. Beenatries N Joy**, Head of the Department for her guidance, help and encouragement for the completion of my project work.

I express my warm thanks to **Ms. Chinju kuriakose**, project Guide for the support and guidance.

ABSTRACT

Aya” is an application. It is the automation of baby and mother caring provided by Asha workers under government of Kerala. This is usually managed under panchayat level and records are kept hand written. Also, to provide notification on health case the workers have to go each home. The main functionalities for baby and mother are baby and mother health, food, vaccination details. Vaccination details of baby will be sent to mobile as alert until vaccination is over. The Aya consist of four users, Government, Panchayat, Asha worker and Mother/pregnant women. The government and panchayat are using web based application and The Asha worker and users are using android application. In this application, Government can view all reports of Panchayat. The Government (Admin) add Panchayat and publish all schemes (e.g.: Maternity Benefit Programme) of government through this application and Asha workers and Users can view these schemes through mobile app .Panchayat are managed by admin and can view report by district wise. Panchayat add all programs conduct by panchayat and Asha worker and Users can view all these programs.

Kerala Government delivers raw foods for children's and pregnant women's, and conducting so many programs like vaccination and program like mothers meeting. Asha worker informs relevant user's availability of food, vaccination conducting details and program conducting details. Users and Asha worker can access these app at anywhere .Additionally this app give other information , which vaccinations you may or may not need during your pregnancy and acts as a child's healthcare provider if your child is up to date with all recommended vaccines. Asha worker can broadcast emergency messages like blood donation needed for a women during delivery. It is obvious that developing Aya project would help in disseminating information much easier between users and Asha worker, panchayat, government and Asha worker and panchayat, and government.

CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
1.	INTRODUCTION	10
	1.1. Introduction of Domain	11
1.	2.Project Description	13
	1.3. Problem Definition	14
	1.4.Objectives	14

2.	SYSTEM ANALYSIS	15
	2.1 .Existing System	16
	2.1.1 .Disadvantages	16
	2.2. Proposed System	17
	2.2.1.Advantages	17
	2.3. Feasibility Study	18
	2.3.1. Technical Feasibility	19
	2.3.2. Economical Feasibility	19
	2.3.3.Operational Feasibility	20
	2.3.4. Behavioral Feasibility	20
3.	SYSTEM REQUIREMENTS	21
	3.1.Software Requirements	21
	3.2.Hardware Requirements	22
		22
	3.3.Software Description	22
	3.3.1. Object Oriented Concepts	22
	3.3.2. Android studio IDE for Android	23
	3.3.3. Java programming language	24

4.	SYSTEM DESIGN	28
	4.1. Data flow diagrams	29
	4.2. Module Description	35
	4.2.1. Module 1	35
	4.2.2. Module 2	35
	4.2.3. Module 3	35
	4.4. ER diagram	36
5.	SYSTEM TESTING	38
	5.1. Testing objectives	39
	5.2. Testing strategies	40
	5.2.1. White box testing	40
	5.2.2. Black box testing	40
	5.3. Types of testing	41
	5.3.1. Unit testing	41
	5.3.2. Integration testing	41
	5.3.3. System testing	42
	5.3.4. Validation testing	42
	5.3.5. User acceptance testing	42
6.	IMPLIMENTATION	43

7.	TABLES	46
8.	FUTURE ENHANCEMENT	50
9.	CONCLISION	52
10.	ANNEXURE	54
11.	PSUEDO CODE	82
12.	BIBLIOGRAPHY	107

INTRODUCTION

INTRODUCTION

1.1 INTRODUCTION OF DOMAIN

Accredited social health activists (ASHAs) are community health workers instituted by the government of India's Ministry of Health and Family Welfare (MoHFW) as a part of the National Rural Health Mission (NRHM). The mission began in 2005; full implementation was targeted for 2012. Once fully implemented, there is to be "an ASHA in every village" in India, a target that translates into 250,000 ASHAs in 10 states. The grand total number of ASHAs in India was reported in July 2013 to be 870,089. There are 859,331 ASHAs in 32 states and union territories as per the data provided by the states in December 2014. This excludes data from the states of Himachal Pradesh, Goa, Puducherry, and Chandigarh, since the selection of ASHA is under way in these states.

ASHAs are local women trained to act as health educators and promoters in their communities. Their tasks include motivating women to give birth in hospitals, bringing children to immunization clinics, encouraging family planning (e.g., surgical sterilization), treating basic illness and injury with first aid, keeping demographic records, and improving village sanitation. ASHAs are also meant to serve as a key communication mechanism between the healthcare system and rural populations. ASHAs preference for selection is they must have qualified up to the tenth grade, preferably be between the ages of 25 and 45, and are selected by and accountable to the gram panchayat (local government). If there is no suitable literate candidate, these criteria may be relaxed.

A baseline survey is to be taken at the district level. It is for fixing decentralized monitoring goals and indicators. The community monitoring would be at the village level. The planning commission would be the eventual monitor of outcomes. External evaluation will be taken up at frequent intervals. There are many benefits from the Government of

Kerala for women and children. But most of them do not know the benefits , and they do not get it. Make contact between women and asha workers more and more and all Government schemes for women and children have come to them, we introduce an application “AYA”.

Each ward of an village will be atleast one Ashaworker but will be looking at the women and children who are present. Kerala Government delivers raw foods for childrens and pregnant womens, and conducting somany programs at ward level for women and children. On the basis of my local area survey, most of the people do not know, when the food arrives and delivers. AYAs an innovation to solve these requirements.

1.2 PROJECT DESCRIPTION

"AYA" is an application. It is the automation of baby and mother caring provided by Asha-workers under government of Kerala. This is usually managed under panchayath level and records are kept hand written. Also, to provide notification on health case the workers have to go each home. The main functionalities for baby and mother are baby and mother health, food, vaccination details. Vaccination details of baby will be sent to mobile as alert until vaccination is over. The AYA consist of four users, Government, Panchayath, Ashaworker and Mother/pregnant women. The government and panchayath are using web based application and the ashaworker and users are using android application. In this application, Government can view all reports of Panchayath. The Government(Admin) add Panchayath and publish all schemes(eg: Maternity Benefit Programme) of government through this application and Ashaworkers and Users can view these schemes through mobile app .Panchayath are managed by admin and can view report by district wise. Panchayath add all programs conduct by panchayath and Ashaworker and Users can view all these programs.

Kerala Government delivers raw foods for childrens and pregnant womens, and conducting somany programs like vaccination and program like mothers meeting. Ashaworker informs relevant users availability of food, vaccination conducting details and program conducting details. Users and Ashaworker can access these app at any where .Additionally this app give other information , which vaccinations you may or may not need during your pregnancy and acts as a child's healthcare provider if your child is up to date with all recommended vaccines. Ashaworker can broadcast emergency messages like blood donation needed for a women during delivery. It is obvious that developing AYA project would help in disseminating information much easier between users and ashaworker,panchayth, governement, and ashaworker and panchayath, and governement.

1.3 PROBLEM DEFINITION

The problems faced by manual work could be well resolved by the implementation of our AYA mobile application that brings an advanced means of passing day today details to pregnant women and childs in a much easier and efficient way. In this system the authorized users admin, panchaytah, and ashaworker can upload the information and details of vaccination, food, program conducting by corresponding user. Ashaworker can add users and pass information needed for those users.

1.4 OBJECTIVES

AYA is a mobile application which is engaged in providing up-to-date information and details needed for a pregnant women or/and mother and other information's for all the users associated with the Ashaworker. The project aims to mould a healthy generation, how the government can improve the efficiency and Getting your child vaccinated on time will help protect him or her against 15 vaccine preventable diseases when it comes to gaining the information from the ashaworker. AYA is one of the applications to improve day today contact between ashaworker and users by making it available online. This mobile application involves almost all the features of AYA, where the implementation helps the pregnant women and mothers to retrieve all the information and notification from asha and government directly through their cell phones. The AYA system will take care of the current information's detail at any point of time. All the updates like add, view is done by endusers and ashaworker, so that user will get the updated current information through mobile. To run the AYA as a program that can be viewed anywhere and anytime.

SYSTEM ANALYSIS

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

In the existing system normally there is no way to alert mothers about vaccination details or food details Eg:pulse polio vaccination.It is very important for child upto 5years.Also about the food details for childs and pregnant.The new Panchayath programs, health centre programs and Government schemes are unavailable to public.The asha worker not bother about new born childs and mothers.

2.1.1 Disadvantages

1. Ashaworker can't manage efficiently each pregnant women and child. Some of the information may not get all public.
2. There are many benefits from the Government of Kerala for women and children. But most of them do not know the benefits , and they do not get it. For example Pregnancy Aid Scheme financial benefit of INR 6000 to pregnant women, but the public don't know how to apply, how it sanctioned.
3. Each month health center conduct program like mothers meeting, most of people don't know about it. Panchayath programs like Gramasabha meeting may be forget.
4. Ashaworker can't continuous look each pregnant womens in their place, some of the pregnant women are working, they are not in home at working time. Then there may have communication gap between ashaworker and pregnant women and mothers.

2.2 PROPOSED SYSTEM

The AYA consist of four users, Government, Panchayath, Ashaworker and Mother/pregnant women. The government and panchayath are using web based application and The ashaworker and users are using android application. In this application, Government can view all reports of Panchayath. The Government(Admin) add Panchayath and publish all schemes(eg: Maternity Benefit Programme) of government through this application and Ashaworkers and Users can view these schemes through mobile app .Panchayath are managed by admin and can view report by district wise. Panchayath add all programs conduct by panchayath and Ashaworker and Users can view all these programs. The Ashaworker are kept updated each time with this mobile app based on their number of organizations through AYA.

Ashaworker informs relevant users availability of food, vaccination conducting details and program conducting details. Users and Ashaworker can access these app at any where .Additionally this app give other information , which vaccinations you may or may not need during your pregnancy and acts as a child's healthcare provider if your child is up to date with all recommended vaccines. Ashaworker can broadcast emergency messages like blood donation needed for a women during delivery. It is obvious that developing AYA project would help in disseminating information much easier between users and ashaworker,panchayth,government,and ashaworker and panchayath, and gouvernement.

2.2.1 Advantages

1. To eliminate wastage of time and energy: AYA will be able to save a lot of paper and time.
2. To bring system into real life: It would be dire need of all Ashaworkers having multiple pregnant women and child as its easy and shortcut method to inform details and guide them.

3. Each child get vaccines at correct time, then we can avoid vaccine diseases, make a healthy generation etc.
4. Enable mobility : Usability of mobile phones are increasing day by day so this app was very helpful in our daily life. A Multi window system was enabled through this app.
4. Easy to use: By the help of free AYA users can access the vaccination details, food details, program details. Asha worker can inform users with update details at any time.
5. Easy analyzing of data: AYA app will get quickly not only in the particular people.
6. Decision making: The information system can sent only selected users only, for example vaccine taken for child at age of 2, this message sent to corresponding users only.

2.3 FEASIBILITY STUDY

Feasibility study is concerned to select the best system that meets performance requirements. These entities are an identification description, an evaluation of candidate systems and the selection of the job.

Economical feasibility

Technical feasibility

Behavioral feasibility

Operational feasibility

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is ensuring that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. The assessment is based on an outline design of the system requirements in terms of input, Process, output, Fields, Programs, and procedures. This can be quantified in terms of volumes of data,

trends, frequency of updating, etc. Technological feasibility is carried out to determine whether the company has the capability, in terms of software, hardware, personnel and expertise, to handle the completion of the project.

2.3.1 Technical feasibility

Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources and being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system. This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Technical analysis center on the existing computer system (Hardware, Software etc) and to what extend it can support the proposed addition. This involves financial consideration to accommodate technical enhancement. If the budget is a serious constraint, then the project is judged not feasible.

2.3.2 Economical feasibility

Economic analysis is the most frequently used method for evaluating the effectiveness of the candidate system this study is carried to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditure must be justified. Thus, the development system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased. Before going further into the development of the proposed system the system analyst has to check the economic feasibility for the proposed system. The cost of developing the system and cost for running the system is compared with the cost benefit that can be achieve by implementing the system.

2.3.3 Operational feasibility

The new system is very much easier and user friendly than the existing system. It satisfies the requirements identified in the requirements analysis phase of system

development. It reduces the operational time considerably. Operational cost is very less. The maintenance and modification of the new system needs very less human effort. The new system is operationally feasible and makes the operations simpler and quite easier.

2.3.4 Behavioral feasibility

The level of acceptance by the users solely depends the methods that are employed to educate the user about the system and to make him familiar with it. User level of confidence must be raised so that the user is also able to make some constructive criticism, which is welcomed, as that user is the final user of the system. An estimate should be made of how strong a reaction the user staff likely to have toward the development of a computerized system. It is common knowledge the computer installations have something do understandable that the introduction of a candidate system requires special effort to educate, sell and train the staff on new ways of considering business. The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as necessity. Proposed system is beneficial only if they can be turned into information system that will meet the organization's operating requirements. People are inheritably resistant to change and computers have been known to facilitate change's estimate should be made to know how strong the reaction of a user administrator is likely to have towards the develop. It always delivers this information at a right place at a right time.

SYSTEM REQUIREMENTS

SYSTEM REQUIREMENTS

3.1 SOFTWARE REQUIREMENTS

Operating system : Windows 10
Database : MySQL
Platform : Android Studio, NetBeans

3.2 HARDWARE REQUIREMENTS

Processor : Intel® Core™ i3-5005U
Speed : 1.60GHz
Memory : 4GB Ram
Hard Disk : 1TB

3.3 SOFTWARE DESCRIPTION

3.3.1 OBJECT ORIENTED CONCEPTS OBJECTS

Objects are the basic runtime entities in the object-oriented system. They may represent a person, a place, a bank account, a table of data etc. They may represent user defined data such as vectors, time, and lists. Programming problem is analyzed in terms of objects and the nature of the communication between them. $\text{Object} = \text{Data} + \text{Process}$

Program objects should be chosen such that they may represent real time world objects. Objects take up space in the memory and have associated address. When a program is executed, the objects interact by sending messages to one another. Each object contains data and code to manipulate the data.

CLASS

Class is defined as an abstract data type characterized by a set of properties (attributes and functions) common to its objects. The class expresses the common features and constitutes a means of classification. The entire set of data encode of an object can be made a user defined data type with the help of a class. The objects are the variables of type class. Once the class has been defined, we can create any number of objects belonging to that class. Each object is associated with the data of type class with which they are created. A class is thus a collection of objects of similar type.

3.3.2 ANDROID STUDIO IDE FOR ANDROID

The project is developed in Android Studio environment. Android and PHP are the programming languages for validation. The database is created in MySQL. Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built in NetBeans's IntelliJ IDEA software and designed specifically for Android development. It is available for download on windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android development Tools (ADT) as primary IDE for native Android application development.

Android Studio was announced on May 16, 2013 at the Google I/O conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0. The current stable version is 3.2, which was released in September 2018.

Android is a Linux based operating system it is designed primarily for touch screen mobile devices such as smart phones and tablet computers. The operating system has developed a lot in last 15 years starting from black and white phones to recent smart phones or mini computers. One of the most widely used mobile OS these days is android. The android is software that was founded in Palo Alto of California in 2003.

The android is a powerful operating system and it supports large number of applications in Smartphones. These applications are more comfortable and advanced for the users. The hardware that supports android software is based on ARM architecture platform. The android is an open source operating system means that it's free and any one can use it. The android has got millions of apps available that can help you managing your life one or other way and it is available low cost in market at that reason's android is very popular.

3.3.3 JAVA PROGRAMING LANGUAGE

The official language for Android development is Java. Large parts of Android are written in Java and its APIs are designed to be called primarily from Java. That said, it is possible to develop C and C++ apps using the Android Native Development Kit (NDK), however it isn't something that Google promotes.

The most important characteristics of Java are that was design from the outset to be machine independent. Java programs can run unchanged on any operating system that supports Java. An application written in Java will only require a single set of source code regardless of the number of different computer platforms on which it is run. In any programming languages the application will frequently require the source code to be tailored to accommodate different environments particularly if there is an extensive GUI involved. Java offers substantial saving in time and resources in developing supporting and maintaining major applications on several different hardware platforms and operating system. The next most important characteristics of Java is that it is object oriented. Object oriented programs are easier to understand and less time consuming to maintain and extend then program that have been returned without the benefit of using object. Java is currently one of most programming languages in use, and is widely used from applications software to applications.

The Java programming language is a high-level language that can be characterized by all the following:

- Simple

- Object-Oriented
- Distributed
- Multi-threaded
- Dynamic
- Portable
- Architecture Neutral

In Java programming language all source code is written in plain text files ending with Java extension. Those source files then compiled into class files by the Javac compiler. A. class file does not contain code that is native to your processor: it instead contains byte codes- the machine language of Java Virtual Machine (JVM). The Java launched tool then runs on your application with an instance of JVM. Because the JVM is available on many different operating system, the same class files are capable of running on Microsoft Windows, Solaris OS.

Characteristics of Java

The target of Java is to write a program once and then run this program on multiple operating systems.

Java has the following properties:

- Platform independent: Java programs use the Java virtual machine as abstraction and do not access the operating system directly. This makes Java programs highly portable. A Java program (which is standard complaint and follows certain rules) can run unmodified on all supported platforms, e.g. Windows or Linux.
- Object-orientated programming language: Except the primitive data types, all elements in Java are objects.

- Strongly-typed programming language: Java is strongly-typed, e.g. the types of the used variables must be pre-defined and conversion to other objects is relatively strict, e.g. must be done in most cases by the programmer.
- Interpreted and compiled language: Java source code is transferred into the bytecode format which does not depend on the target platform. These bytecode instructions will be interpreted by the Java Virtual machine (JVM). The JVM contains a so-called Hotspot-Compiler which translates performance critical bytecode instructions into native code instructions.
- Automatic memory management: Java manages the memory allocation and de-allocation for creating new objects. The program does not have direct access to the memory. The so-called garbage collector deletes automatically objects to which no active pointer exists.

MYSQL

MySQL is a database system used on the web. Basically, a MySQL database allows you to create a relational database structure on a web-server somewhere in order to store data or automate procedures. If you think of it in comparison to Microsoft Access, MySQL is what holds all of your tables, PHP acts as your queries (among other things), and your forms are basically web pages with fields in them. With all of this combined, you can create truly spectacular projects on the web. A database most often contains one or more tables. Each table is identified by a name (E.g. “Customer” or “Orders”). A table contains record (rows) with data. With SQL we can query a database and have a result set returned. SQL is the syntax for executing queries. But the SQL language also includes the syntax to insert and delete records. These query and update commands together form the Data Manipulation Language (DML) part of SQL. The Data Definition Language (DDL) part of SQL permits database tables to be created or detected. We can also define indexes (keys), specify links between tables and imposes constraints between databases.

DATABASE DESIGN

Database design is the process of producing a detailed data model of a database. This logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in Data Definition Language, which can then be used to create a database. A fully attributed data model contains detailed attribute for each entity. The term database design can be used to describe many different parts of the design of an overall system. In the relational model these are the tables and views. In an object database design, the entities and relationship map directly to object classes and named relationships.

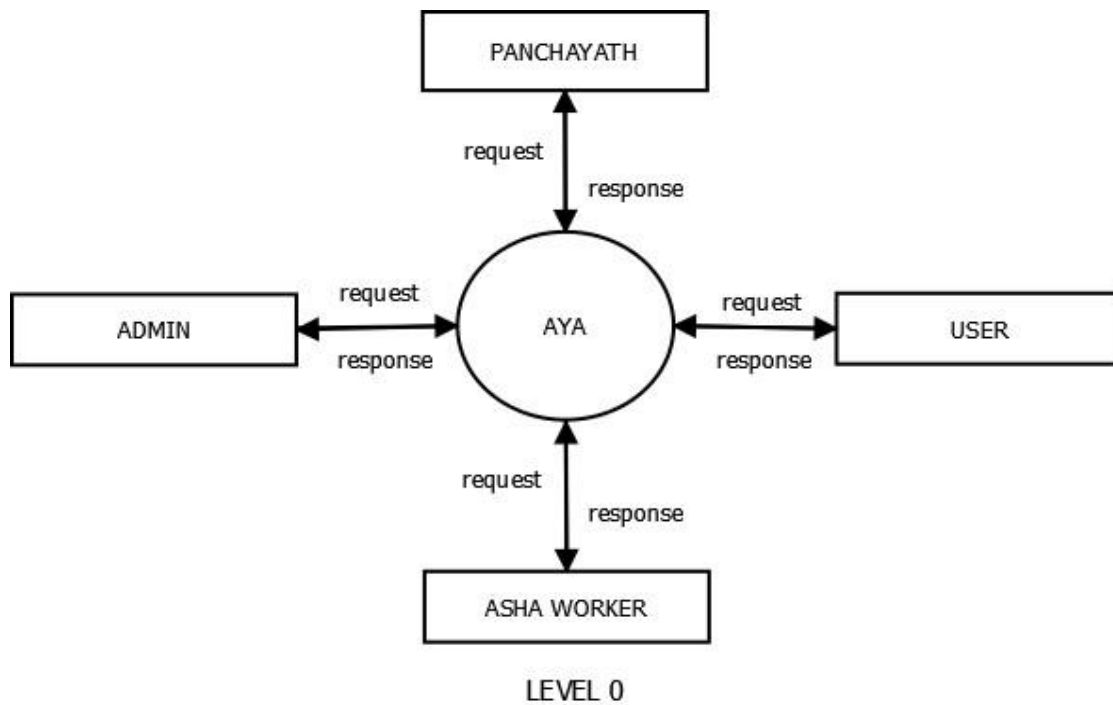
SYSTEM DESIGN

SYSTEM DESIGN

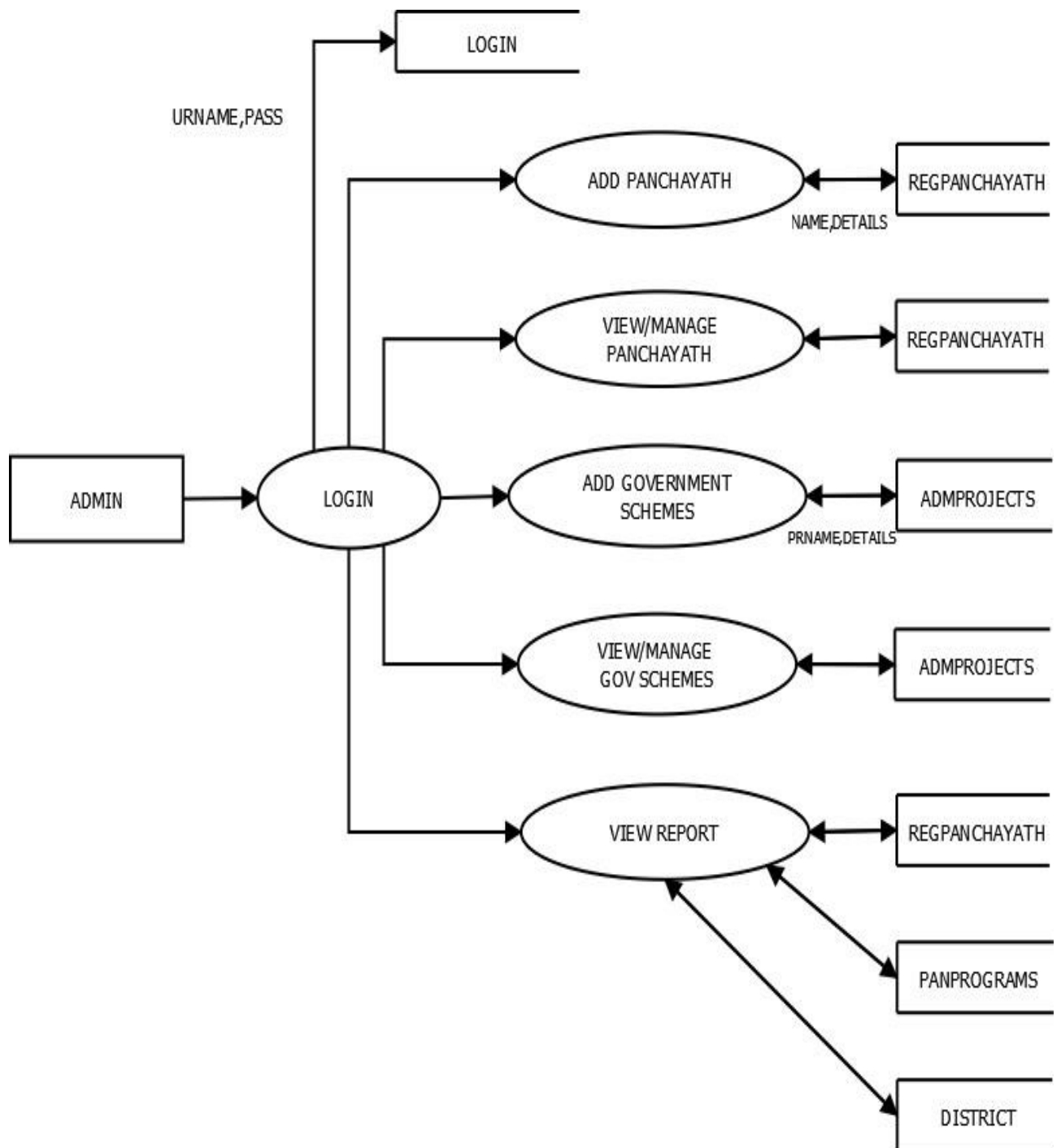
4.1 DATA FLOW DIAGRAMS

A data flow diagram (DFD) is a graphical representation of the “flow” of data through an information system, modeling its process aspects. Often, they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs also be used for the visualization of data processing (Structured design). Data flows are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it.

A DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of the processes, or information about whether processes will operate in sequence or in parallel. The first level DFD shows the main processes within the system. each of these processes can be broken into further processes until you reach pseudo code.

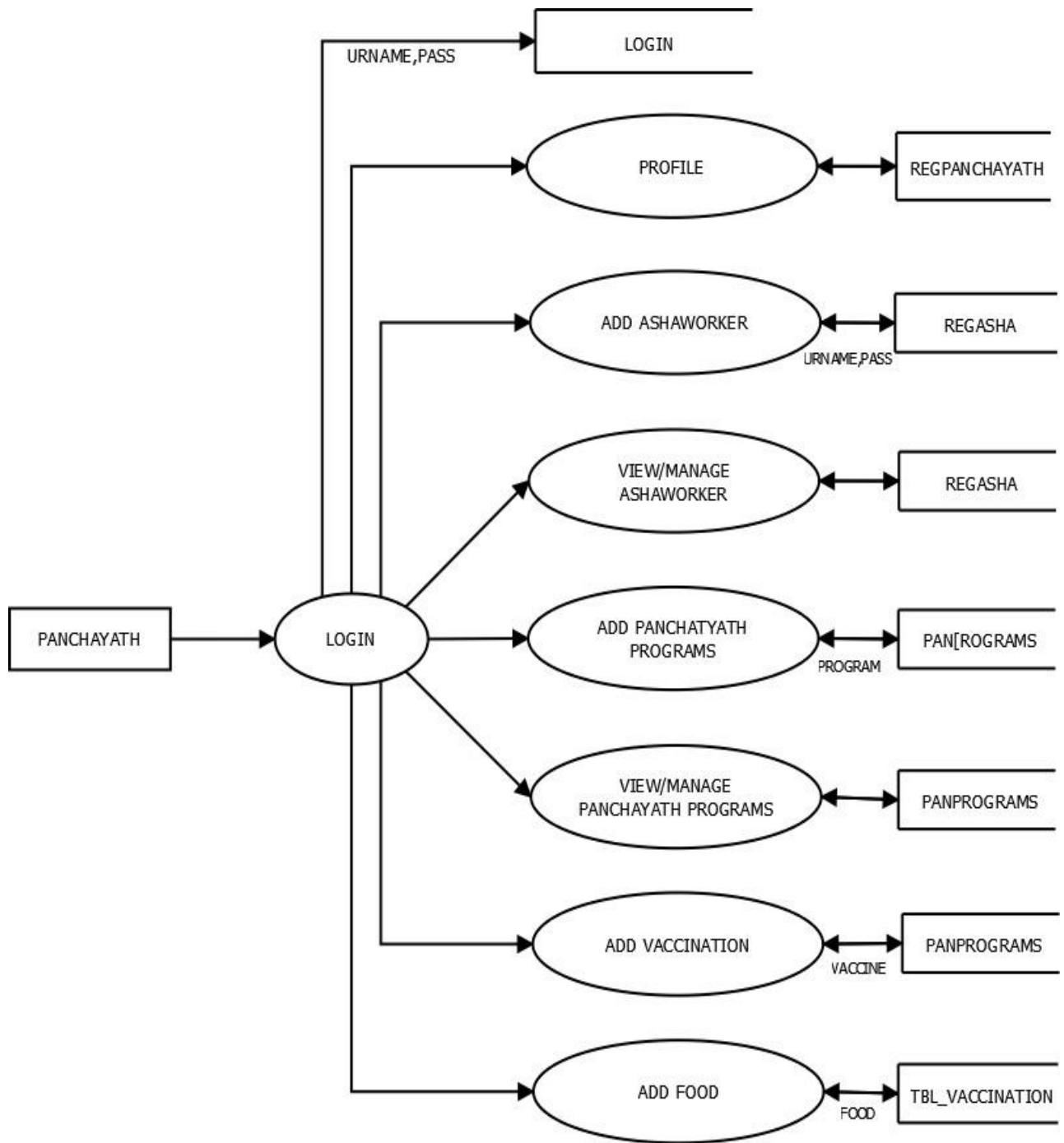


ADMIN-LEVEL 1

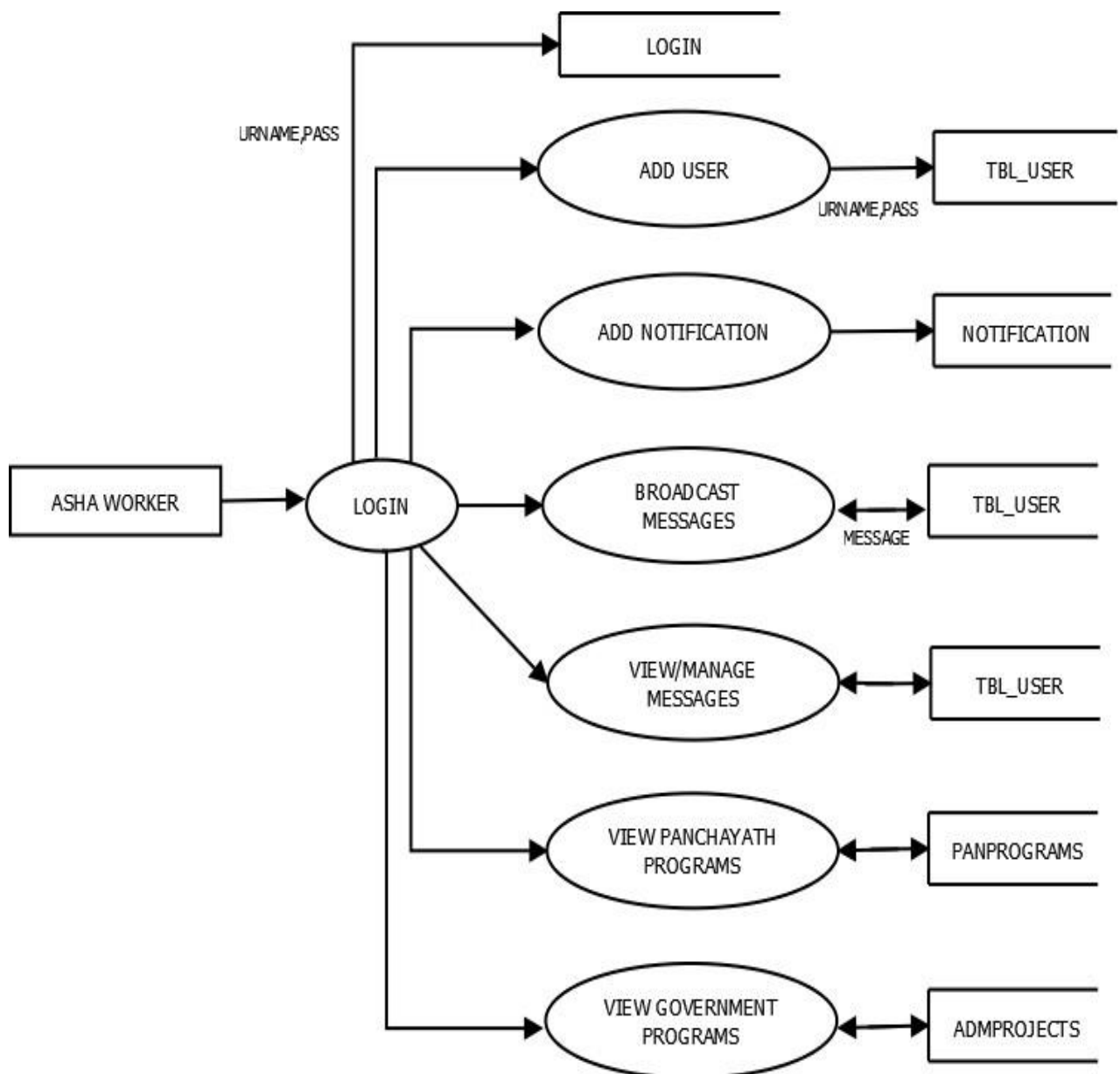


ADMIN-LEVEL 1

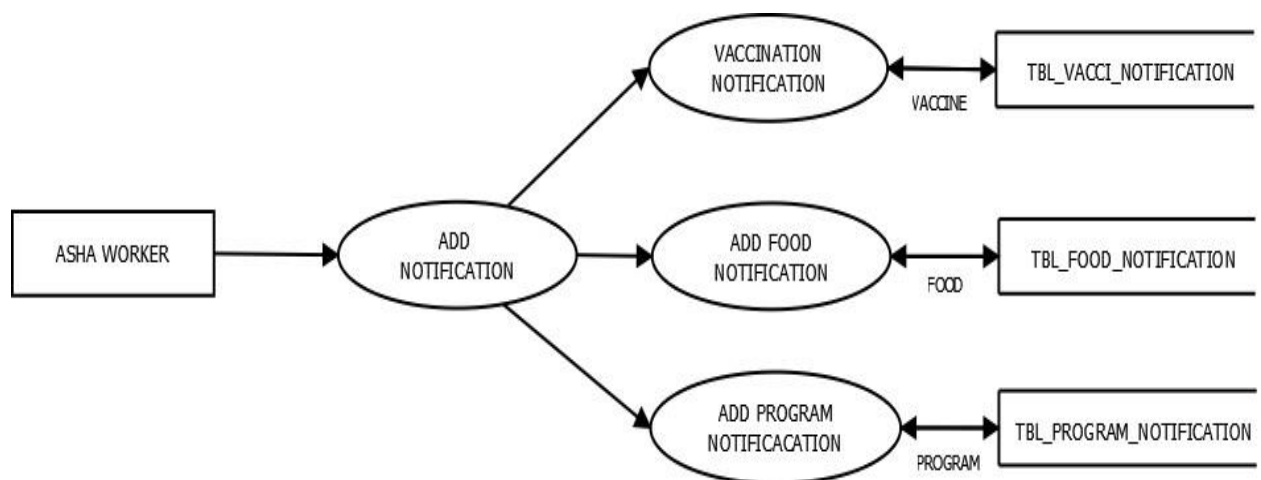
PANCHAYATH LEVEL 1

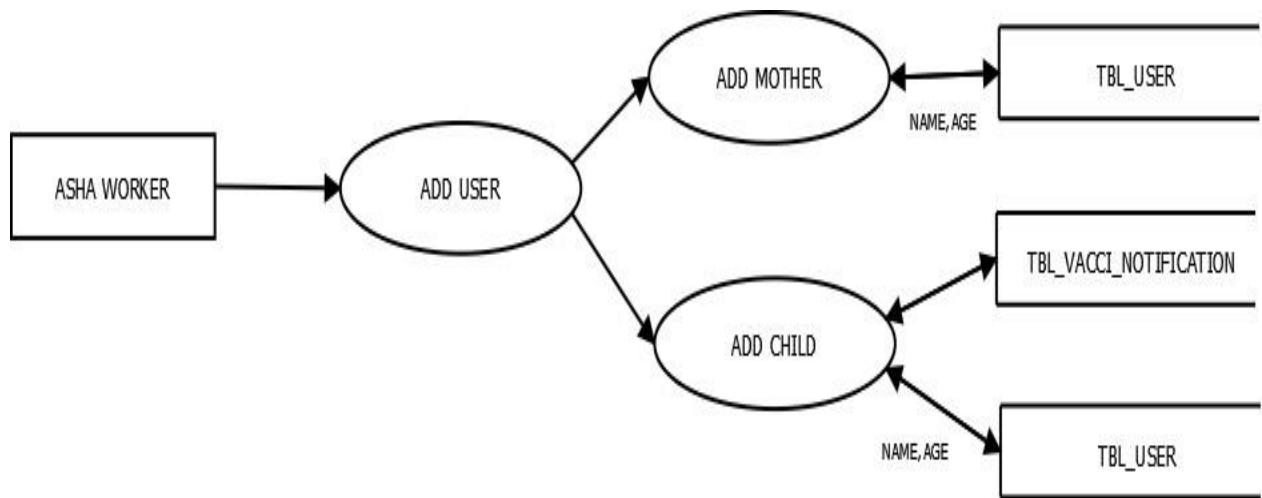


PANCHAYATH-LEVEL1

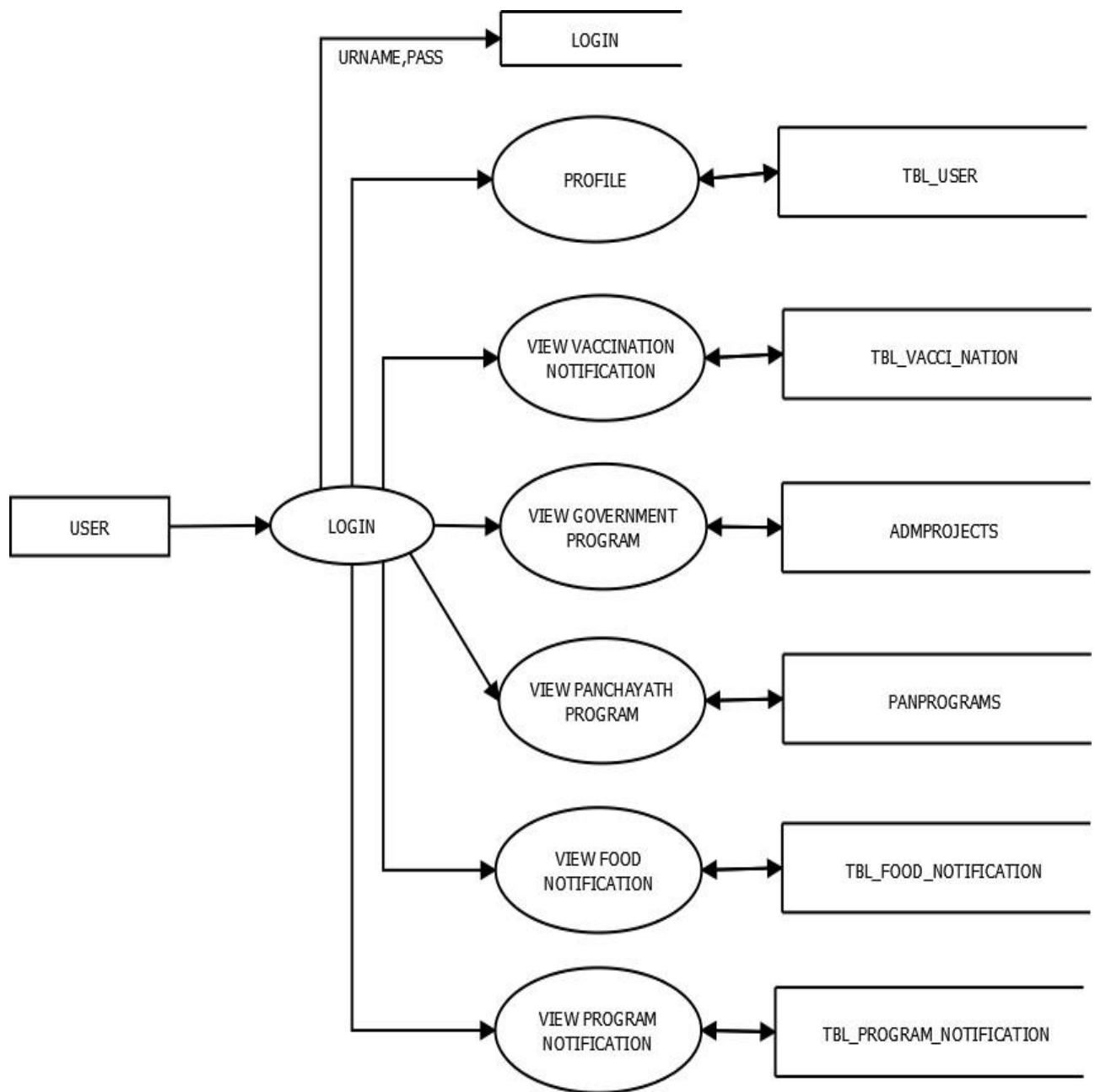
ASHAWORKER LEVEL 1

ASHA WORKER-LEVEL 1

ASHA WORKER LEVEL 2



USER LEVEL 1



USER-LEVEL1

4.2 MODULE DESCRIPTION

4.2.1 MODULE 1: ADMIN MODULE

The AYA word generally intends to act as a support system for the health center. Admin module is a web part of the AYA application. The Government act as a admin of

the system. After the action of the government , the whole process initiated. The responsibilities of admin are, add panchayath, manage panchayath, add government projects, manage projects, view reports of each panchayath.

4.2.2 MODULE 2:PANCHAYATH MODULE

Panchayath module is second module in the application, which is another web part of the application. In this system panchayath are add ashaworkers, and mange ashaworkers. Other responsibilities are edit profile of the panchayath, add food needed for child and mother and manage them, add and view vaccination details, add and manage panchayath programs.

4.2.3 MODULE 3:ASHAWORKER MODULE

Ashaworker is third module of the application, its mobile app.Ashaworker have previlage to add users, the user may be pregnant women and/or mothers. Asha worker can view all panchayath programs and government schemes. Asha worker add child on the basis of mother registration.Ashaworker can sent mobile message as a emergency message. Ashaworker can vie and mange the users of the ashaworker.

4.2.4 MODULE 4:USER MODULE

User module is second module in the mobile app. In this system users are Pregnant women and/or mother. User can view panchayath programs and government schemes. Users can view vaccines for women and child. Food needed in the pregnancy period and child. Get alerts from the mobile app as notification about vaccination. User can view their profile.

4.4 E-R DIAGRAM

Entity-Relationship (ER) Model is based on the notion of real-world entities and relationships among them. While formulating real-world scenario into the database model, the ER Model creates entity set, relationship set, general attributes and constraints. Entity Relationship Diagram

(ERD) illustrates the logical structure of databases.ER Model is best used for the conceptual design of a database.

ER Model is based on –

- Entities and their attributes.
- Relationships among entities.

The ER diagram illustrates the database schema for a TB management system. It includes the following entities and their attributes:

- tbluser**: name, uid, email, phone
- tblfood**: fid
- tblvaccination**: vid
- tblproject**: pid
- tblnotification**: fid, pid
- tblregasha**: aid
- tblregpanchayath**: panid
- tbladmpoints**: admid

The relationships between these entities are defined by the following constraints:

- tbluser** has **tbladmpoints** (1:1 relationship)
- tbladmpoints** has **tblvaccination** (1:1 relationship)
- tbladmpoints** has **tblproject** (1:1 relationship)
- tblproject** has **tblregasha** (1:1 relationship)
- tblproject** has **tblregpanchayath** (1:1 relationship)
- tblproject** has **tblnotification** (1:1 relationship)
- tblregasha** has **tblregpanchayath** (1:1 relationship)
- tblregpanchayath** has **tblfood** (1:1 relationship)
- tblnotification** has **tblfood** (1:1 relationship)
- tblnotification** has **tblproject** (1:1 relationship)
- tblnotification** has **tbladmpoints** (1:1 relationship)
- tblnotification** has **tblregasha** (1:1 relationship)
- tblnotification** has **tblregpanchayath** (1:1 relationship)
- tblnotification** has **tblfood** (1:1 relationship)

SYSTEM TESTING

SYSTEM TESTING

Software testing is critical element of software quality assurance and represent the ultimate review of the specification, design and coding. System testing makes a logical assumption that all the part of the system is correct; the goal will be successfully achieved. Testing is a set of activity that can be planned in advance and conducted. Systematically, this is aimed at ensuring that the system works accurately and efficiently before live operations commences.

- Testing is the process of correcting a program with intend of finding an error.
- A good test case is one that has high probability of finding a yet undiscovered error.
- A successful test is one that uncovers a yet undiscovered error.

5.1 TESTING OBJECTIVE

There are several rules that can serve as testing objectives

- Testing is a process of executing a program with the Intel of finding an error.
- A good test case is one that has high probability finding an undiscovered error.
- A successful test is one that uncovers and undiscovered error.

Testing is vital of the success of the system. System testing makes a logical assumption that if all parts of the system are subject to variety of tests on-line response, volume, stress, recovery and security and usability tests. A series of tests are performed before the system is ready for user acceptance testing.

5.2 TESTING STRATEGIES

- White Box Testing
- Black Box Testing

5.2.1 WHITE BOX TESTING

White box testing is also known as code testing. The code testing strategy check for the correctness of every statement in the program. To follow this strategy, there should be cases that result in execution of every instruction in the program or module, which is every path in

the program, is tested. The test cases should be guaranteed that independent paths within module are executed once. □ Exercise all logical decision on their true or false sides.

□ Execute all loops at their boundaries and within their operational bounds.

This testing strategy, on the face of it, sounds exhaustive. If every statement in the program is checked for its validity, there does not seem to be much scope of error.

5.2.2 BLACK BOX TESTING

Black box is also known as specification testing. To perform black box testing, the analyst examines the specification taking what the program or modules should do and how it should perform on the various conditions and submitted for processing. By examine the result, the analyst can examine whether the program performs according to the specified requirements.

5.3 TYPES OF TESTING

Different types of testing are,

- Unit testing
- Integration testing
- System testing
- User acceptance testing

5.3.1 UNIT TESTING

Unit testing enables a programmer to detect error in coding. A unit test focuses verification of the smallest unit of software design. This testing was carried out during the coding itself. In this testing step, each module is going to be work satisfactorily as the expected output from the module.

5.3.2 INTEGRATION TESTING

Through each program work individually, they should work after linking together. This is referred to as interfacing. Data may be lost across the interface; one module can have adverse effect on the other. Subroutines after linking may not do the desired function expected by the main routine. Integration testing is the systematic technique for constructing the program structure while at the same time conducting test to uncover errors associated with the interface. Using integrated test plan prepared in the design phase of the system development as a guide, the integration test was carried out. All the errors found in the system were corrected for the next testing step.

5.3.3 SYSTEM TESTING

After performing the integration testing, the next step is output testing of the proposed system. No system could be useful if it doesn't produce the required output in a specified format. The output generated are displayed by the system under consideration and then tested by comparing with the format require by the user. Here the output format is considered in to two ways, one in on-screen and other in printed format.

5.3.4 VALIDATION TESTING

The user has to work with the system and check whether the project meets his needs. In the validation checking, the user works with the beta version of the software.

5.3.5 USER ACCEPTANCE TESTING

User acceptance of a system is a key factor of the success of any system. The system under consideration was tested for user acceptance by running a prototype of the software.

IMPLEMENTATION

IMPLEMENTATION

The implementation phase of the software design consists of different tasks to be done sequentially for obtaining the desired results. Here we do not implement parallel; instead we first implement admin and then the user module. The different phases are:

1. Creating the database

A Database is created, in which all the tables are defined which are required to do the different operations such as storage and retrieval of information. Databases are designed in such a way it can handle the different database queries. User and admin can retrieve required details from the system on clicking on the links and buttons.

2. Creating the User Interface

Graphical User Interface is created in Android Studio 3.0 for a user-friendly interface. It is intended for two purposes. First is to create a user-friendly interface for the software. Having a good user interface makes it easier for the user to use and understand the different functionalities of the software. Secondly, the user interface hides the end users from the complexities in the working of the software. So, the user is made unaware of how a task is performed when he chooses to perform it.

3. Creating System Environment

For the intended project to work on, we need to implement its required hardware and software requirements. This system is build using Android Studio 3.0 using the Java language and based on Windows Operating System. Memory and Hard disk should confirm according to hardware requirements mentioned above.

4.Performing Functions

Initially we implement the admin module because the operation made in admin reflects in the user module. After implementing this, we check whether all the operations and reports are up to the specification. Only after ensuring this we go for user module. With the help of a user-friendly visual interface, the end users can perform the desired operations. Upon selecting any operation, the corresponding link commands will be passed to database and the response is also received. The responses are then retrieved and displayed in screen.

TABLES

**Table: admprojects****Columns (5)**

Calculate Optimal Datatypes

Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	pro_id	bigint(30) NOT NULL	
	pro_name	varchar(120) NULL	
	pro_user	varchar(100) NULL	
	pro_det	varchar(200) NULL	
	pro_date	date NULL	

**Table: district****Columns (2)**

Calculate Optimal Datatypes

Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	DId	bigint(20) NOT NULL	
	DName	varchar(80) NULL	

**Table: login****Columns (5)**

Calculate Optimal Datatypes

Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	logID	bigint(20) NOT NULL	
	UserId	bigint(20) NULL	
	UserName	varchar(70) NULL	
	Password	varchar(70) NULL	
	Type	varchar(70) NULL	

**Table: panprograms****Columns (7)**

Calculate Optimal Datatypes

Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	ProgID	bigint(20) NOT NULL	
	PID	varchar(20) NULL	
	ProgName	varchar(120) NULL	
	PanProLoc	varchar(150) NULL	
	PanProDate	date NULL	
	proptime	time NULL	
	ProgDet	varchar(150) NULL	

**Table: regasha****Columns (10)**

Calculate Optimal Datatypes

Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	AID	bigint(30) NOT NULL	
	PID	varchar(20) NULL	
	Name	varchar(100) NULL	
	DOB	date NULL	
	Gender	varchar(10) NULL	
	Address	varchar(200) NULL	
	WardNo	varchar(20) NULL	
	Qualification	varchar(150) NULL	
	Email	varchar(120) NULL	
	Phone	varchar(50) NULL	

**Table: register****Columns (6)**
[Calculate Optimal Datatypes](#)

 Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	rid	int(10) NOT NULL	
	name	varchar(40) NOT NULL	
	Gender	varchar(6) NULL	
	DOB	date NULL	
	Address	varchar(80) NULL	
	District	varchar(70) NULL	

**Table: regpanchayath****Columns (11)**
[Calculate Optimal Datatypes](#)

 Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	PID	bigint(20) NOT NULL	
	DId	bigint(20) NULL	
	PName	varchar(100) NULL	
	PAddress	varchar(250) NULL	
	PArea	varchar(100) NULL	
	President	varchar(80) NULL	
	empNos	varchar(30) NULL	
	Wards	varchar(30) NULL	
	houses	varchar(30) NULL	
	Email	varchar(120) NULL	
	Phone	varchar(100) NULL	

**Table: tbl_childregister****Columns (8)**

Calculate Optimal Datatypes

Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	cid	bigint(20) NOT NULL	
	aid	bigint(30) NULL	
	hnumber	varchar(20) NULL	
	mid	bigint(30) NULL	
	name	varchar(60) NULL	
	gender	varchar(60) NULL	
	dob	varchar(50) NULL	
	date	varchar(50) NULL	

**Table: tbl_emergmsg****Columns (4)**

Calculate Optimal Datatypes

Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	mid	bigint(40) NOT NULL	
	aid	bigint(30) NULL	
	subject	varchar(40) NULL	
	message	varchar(120) NULL	

**Table: tbl_program_notification****Columns (7)**

Calculate Optimal Datatypes

Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	progid	bigint(30) NOT NULL	
	aid	bigint(20) NULL	
	programe	varchar(50) NULL	
	pdate	varchar(50) NULL	
	plocation	varchar(60) NULL	
	ptime	varchar(20) NULL	
	description	varchar(80) NULL	

**Table: tbl_user****Columns (11)**

Calculate Optimal Datatypes

Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
	uid	bigint(50) NOT NULL	
	aid	bigint(30) NULL	
	hnumber	varchar(30) NULL	
	name	varchar(70) NULL	
	dob	varchar(30) NULL	
	carring_month	varchar(20) NULL	
	address	varchar(100) NULL	
	phone	varchar(60) NULL	
	email	varchar(30) NULL	
	photo	longblob NULL	
	date	varchar(40) NULL	

FUTURE ENHANCEMENT

The proposed system will be implemented, continued, reviewed and improved in a later work. The system had been developed in an attractive way in a limited platform as a standalone application. User with minimum knowledge about the phone can also operate the system easily. In future it can be used or implemented in more attractive ways

CONCLUSION

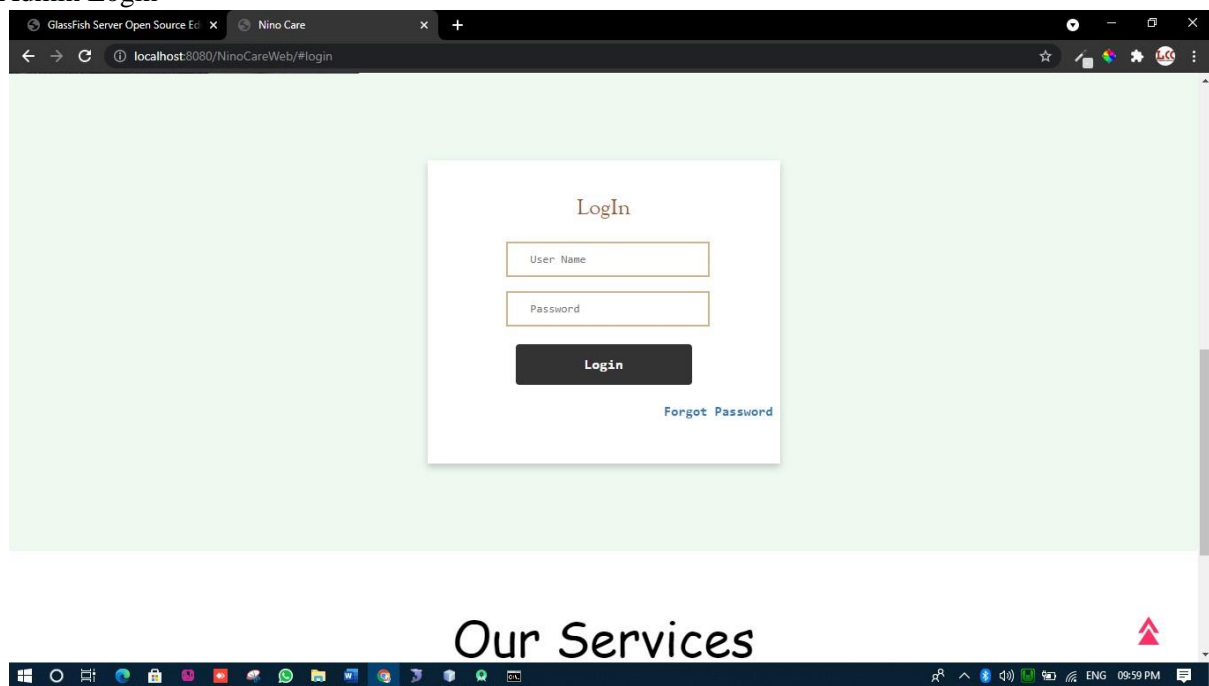
CONCLUSION

The AYA establishes a communication between government, panchayath, asha worker and users. The application is easy to track the status of applications at any level at any point of time. Can access required information and manage easily. The Nino Care is very much helpful for the pregnant womens and babies. Normally there is no way to alert mothers about vaccination details or food details. Eg:pulse polio vaccination.It is very important for child upto 5years.Also about the food details of new born childs. At the heart of the AYA Includes the following features ,Nino Care the application is very simple to use with efficient user friendly interface which allow the end users to easily retrieve the information. It avoids more manual hours that need to spend in record keeping and generating reports. This application keeps the data in a centralized way which is available to all the users simultaneously. Thus, we can conclude that this project is just a start, an idea to make use of mobile in business organization to a next level.

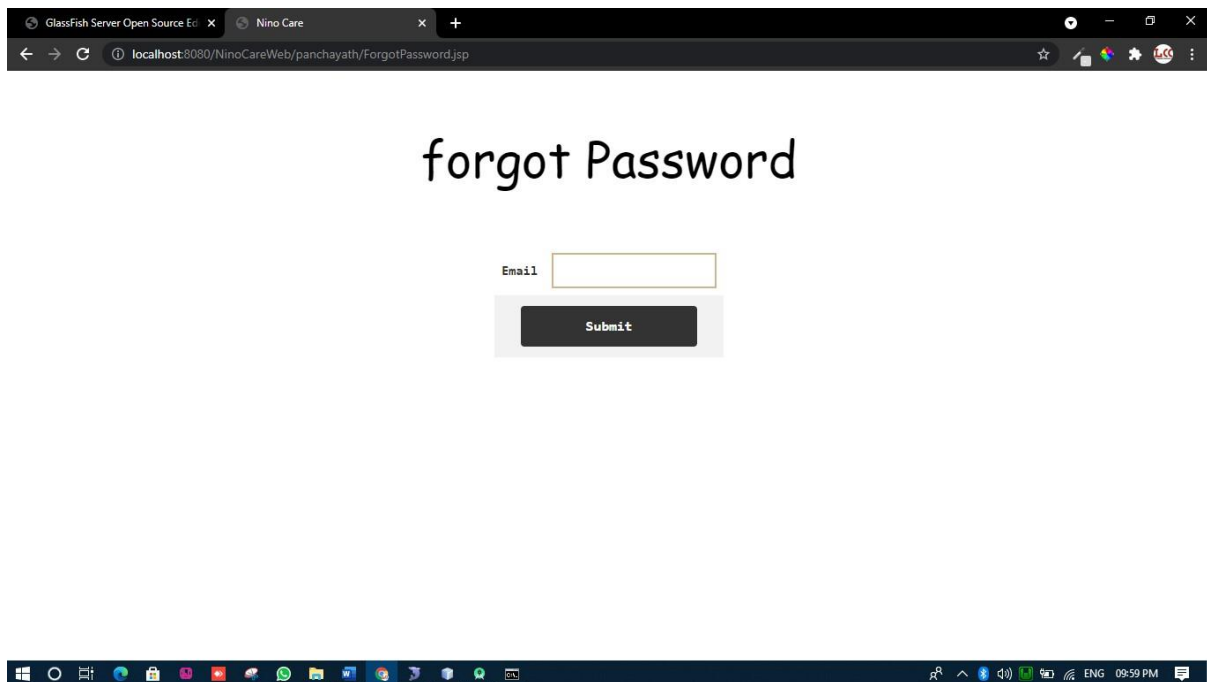
ANNEXURE

Screen Shots

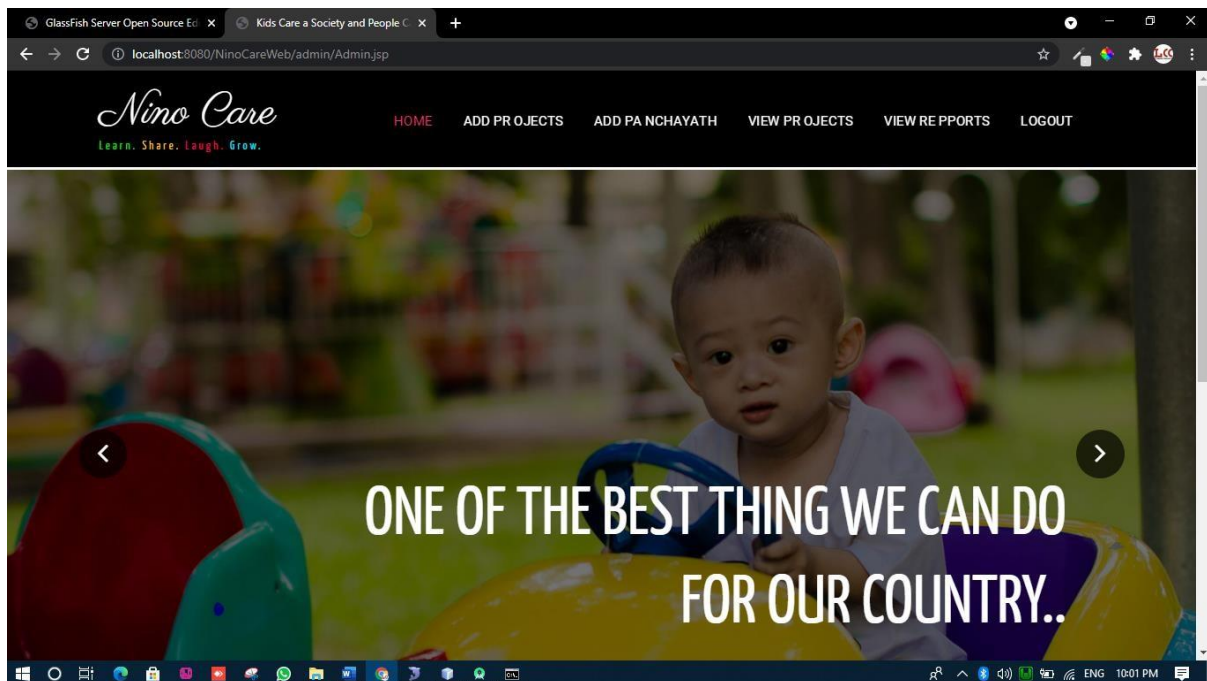
Admin Login



Forgot Password



Admin Home



Add Project

Name Of Project
 Project User
 Project Details
 Announced Date
 Add

Add Panchayath

District
 Name Of Panchayath
 Address
 Area
 President
 Total number of Employees
 Wards
 Houses
 Email
 Phone
 Username
 Password
 Register

View Project

Project Lists

Sl No	Project Name	Project User	Project Details	Announced Date	Edit/Delete
1	Foodcare	Elina	huhfrueh	2019-04-03	EDIT DELETE
2	foodcareprojec	Aleena	food for children	2019-04-11	EDIT DELETE
3	Baby	sowmya	Project is focusing for hw to care a baby	2019-05-04	EDIT DELETE

Report List

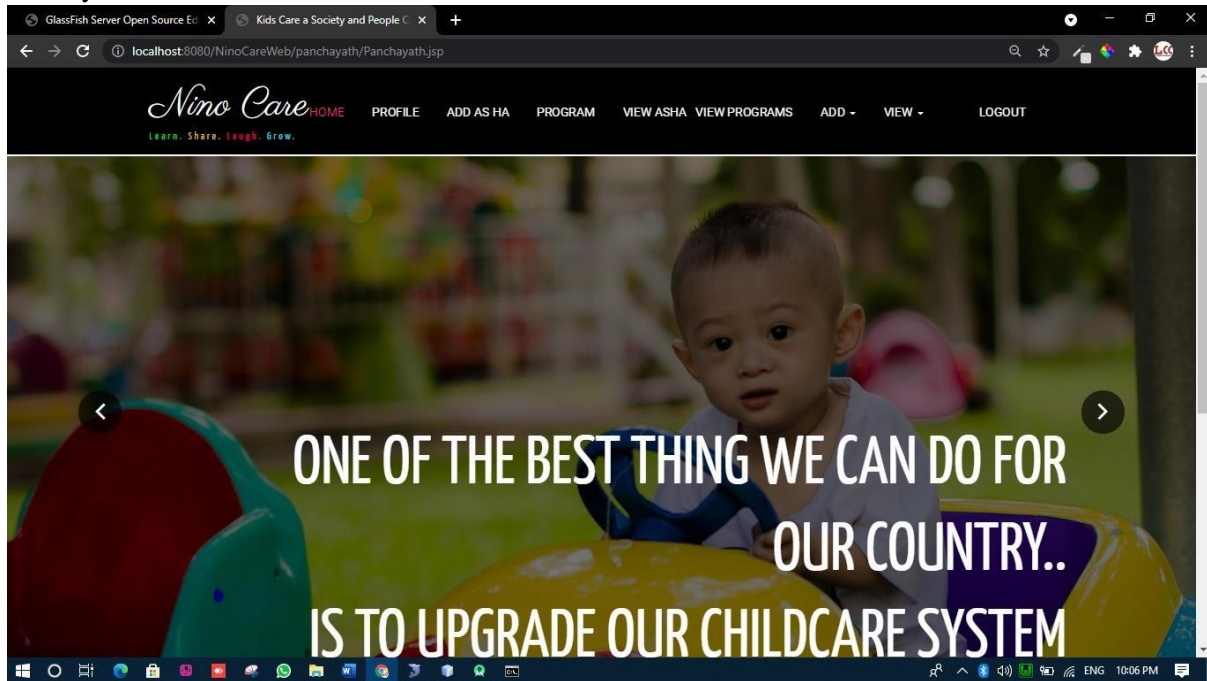
Report Lists

Select District:

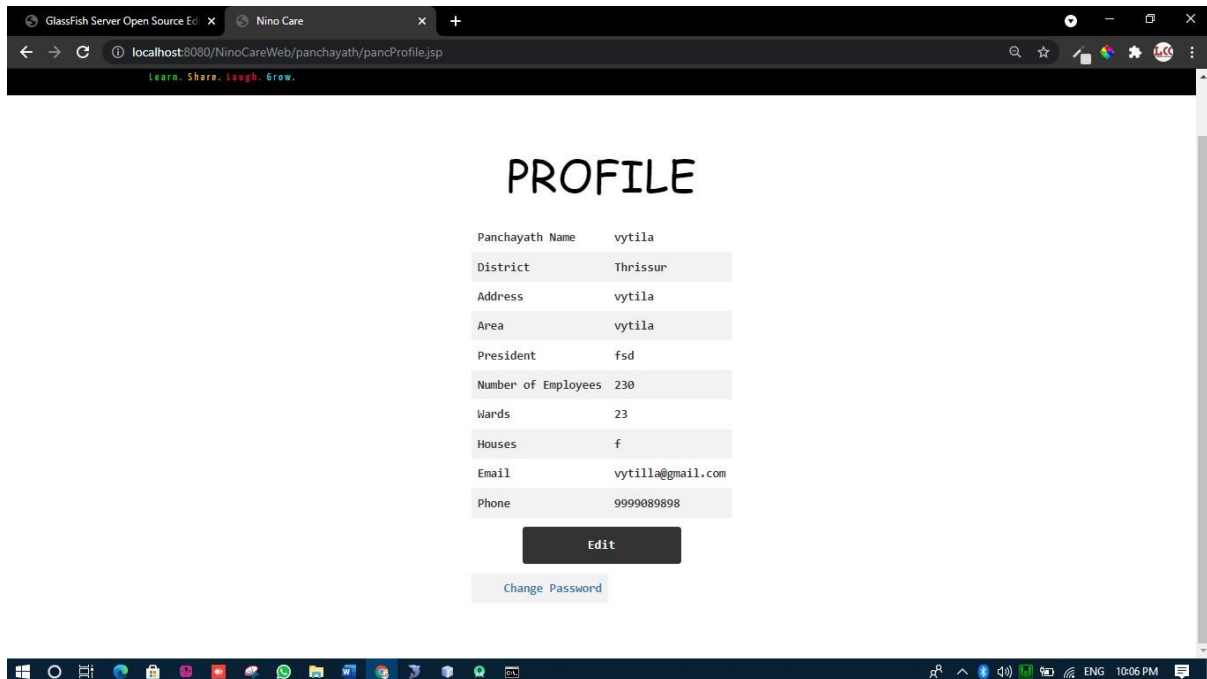
Select Panchayath:

Sl.No	Program Name	Location	Program Date	program time	Program Details
1	mothers meeting	kadaaa	2018-09-13	04:00:00	chvdudew
2	counseling	perumpilavu	2018-09-13	04:00:00	chvdudew

Panchayath Home



Profile



Edit Profile

The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/pancProfile.jsp`. The page has a dark header with the 'Nino Care' logo and navigation links: HOME, PROFILE, ADD AS HA, PROGRAM, VIEW ASHA, VIEW PROGRAMS, ADD -, VIEW -, and LOGOUT. The main content area is titled 'EDIT PROFILE' and contains a form with the following fields:

Panchayath Name	vytila
District	Thrissur
Address	vytila
Area	vytila
President	fsd
Number of Employees	230
Wards	23
Houses	f
Email	vytila@gmail.com

Add Asha

The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/regAsha.jsp`. The page has a dark header with the 'Nino Care' logo and navigation links: HOME, PROFILE, ADD AS HA, PROGRAM, VIEW ASHA, VIEW PROGRAMS, ADD -, VIEW -, and LOGOUT. The main content area is titled 'Add Asha Worker' and contains a form with the following fields:

Name	
DOB	dd-mm-yyyy
Gender	<input type="radio"/> Male <input type="radio"/> Female
Address	
Ward No	1
Qualification	
Email	
Phone	
username	
password	xlysg7

Below the form is a 'Register' button.

Add Programs

The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/addProgram.jsp`. The page has a dark header with the Nino Care logo and navigation links: HOME, PROFILE, ADD AS HA, PROGRAM, VIEW ASHA, VIEW PROGRAMS, ADD -, VIEW -, and LOGOUT. The main content area is titled 'Add Program' and contains a form with the following fields:

- Name Of Program:
- Program Location:
- Date: with a calendar icon
- Time: with a clock icon
- Program Details:
- Add:

The Windows taskbar at the bottom shows the time as 10:07 PM.

Asha Worker List

The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/viewAsha.jsp`. The page has a dark header with the Nino Care logo and navigation links: HOME, PROFILE, ADD ASHA, PROGRAM, VIEW ASHA, VIEW PROGRAMS, ADD -, VIEW -, and LOGOUT. The main content area is titled 'ASHA WORKERS LIST' and contains a table with the following data:

Sl.No	Name	Gender	WardNo	Email	Phone	Remove
1	ammu	Female	7	sikacr@gmail.com	222222222222	<input type="button" value="Delete"/>

The Windows taskbar at the bottom shows the time as 10:08 PM.

Program List

The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/viewPrograms.jsp`. The page features a navigation bar with the Nino Care logo and links for HOME, PROFILE, ADD AS HA, PROGRAM, VIEW ASHA, VIEW PROGRAMS, ADD -, VIEW -, and LOGOUT. The main content area is titled "PROGRAMS LIST" and contains a table with the following data:

Sl.No	Program Name	Location	Date	Time	Details	Edit/Delete
1	mothers meeting	kadaaa	2018-09-13	04:00:00	dwdwdew	EDIT DELETE
2	counselling	perumpilavu	2018-09-13	04:00:00	dwdwdew	EDIT DELETE

Add Food

The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/addFood.jsp`. The page features the same navigation bar as the previous screenshot. The main content area is titled "ADD FOOD" and contains a form with the following fields:

- Food Name:
- User Of the Food:
- Food Period:
- Food Details:

Below the form is an "Add" button.

Add Vaccines

The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/addVaccination.jsp`. The page has a dark header with the 'Nino Care' logo and navigation links: HOME, PROFILE, ADD AS HA, PROGRAM, VIEW ASHA, VIEW PROGRAMS, ADD -, VIEW -, and LOGOUT. The main content area is titled 'ADD VACCINATION' and contains a form with the following fields:

- Vaccination Name:
- User Type:
- Vaccination Age:
- Vaccination Details:

Below the form is a dark 'Add' button. The Windows taskbar at the bottom shows the time as 10:09 PM.

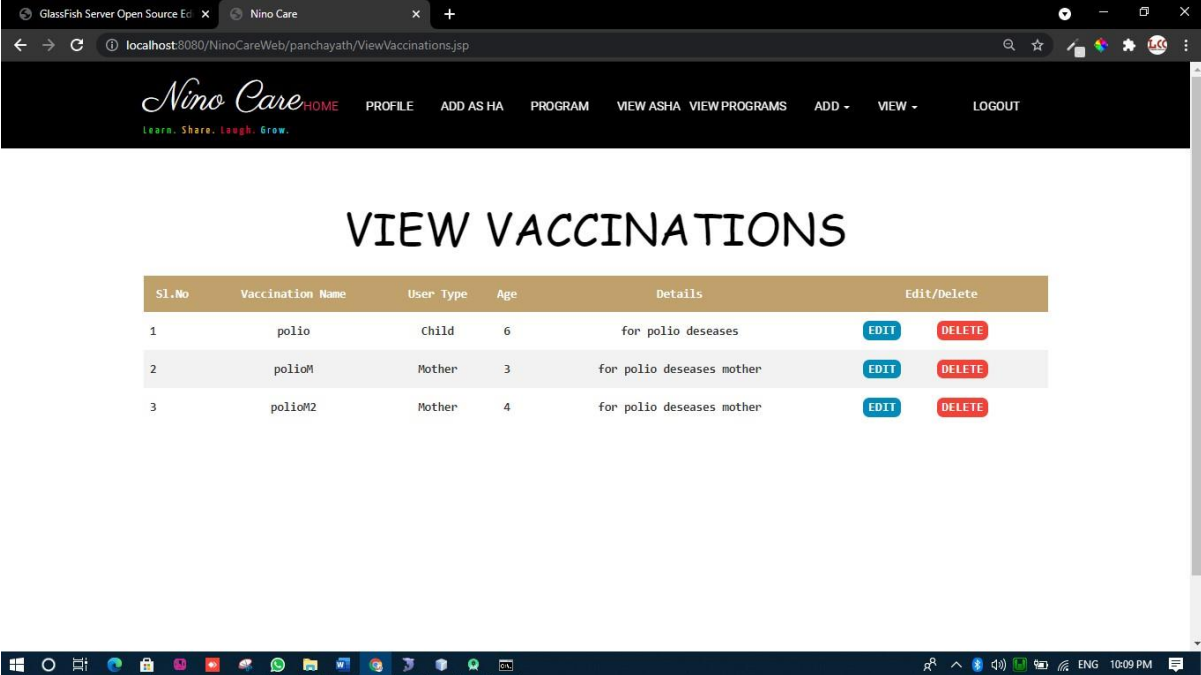
View Food

The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/ViewFood.jsp`. The page has the same dark header as the previous screenshot. The main content area is titled 'VIEW FOODS' and displays a table with the following data:

Sl.No	*Food Name	User Type	Period	Details	Edit/Delete
1	boiled vegetable salad	Mother	1	its good for health.high vitamins	EDIT DELETE

The Windows taskbar at the bottom shows the time as 10:09 PM.

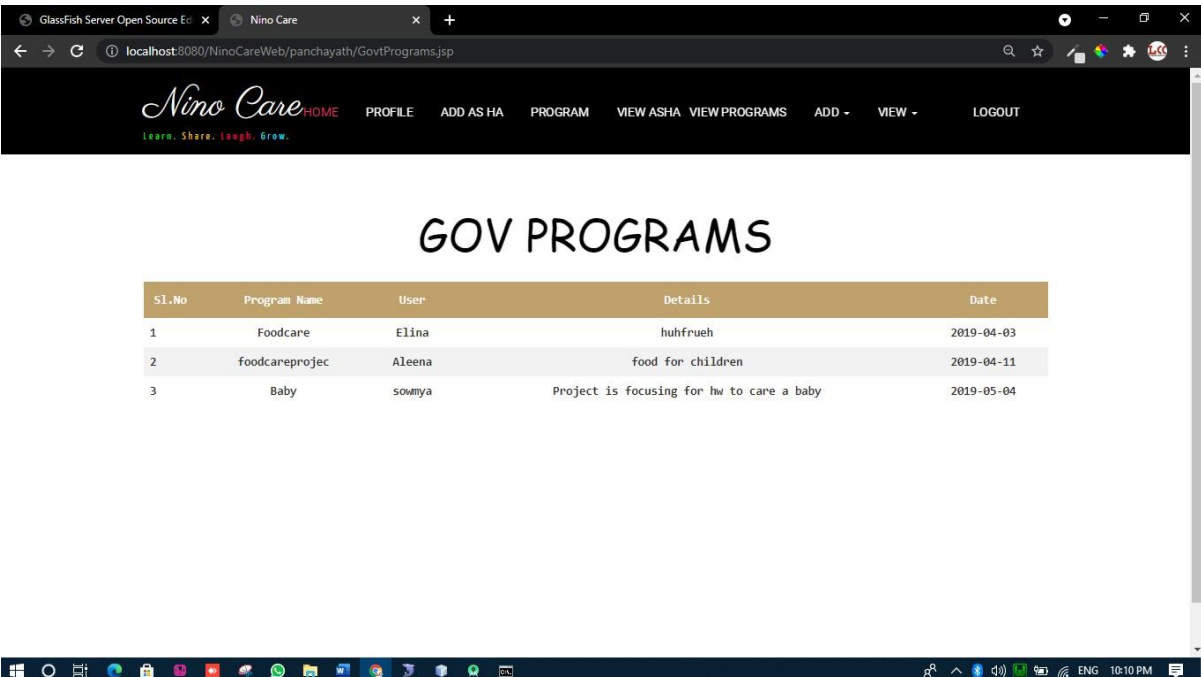
View Vaccines



The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/ViewVaccinations.jsp`. The page features a dark navigation bar with the Nino Care logo and links: HOME, PROFILE, ADD AS HA, PROGRAM, VIEW ASHA, VIEW PROGRAMS, ADD -, VIEW -, and LOGOUT. The main content area is titled 'VIEW VACCINATIONS' and contains a table with the following data:

Sl.No	Vaccination Name	User Type	Age	Details	Edit/Delete
1	polio	Child	6	for polio diseases	EDIT DELETE
2	polioM	Mother	3	for polio diseases mother	EDIT DELETE
3	polioM2	Mother	4	for polio diseases mother	EDIT DELETE

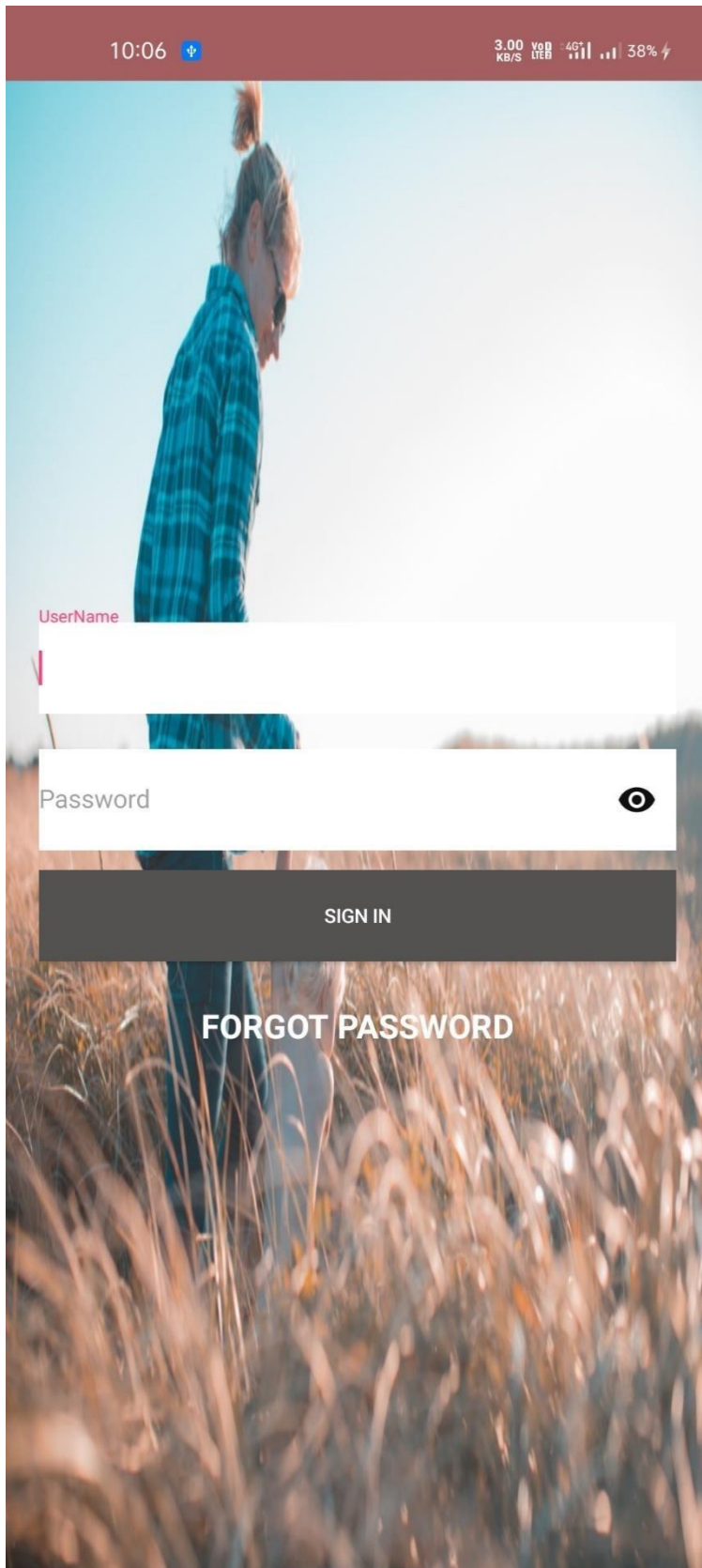
Govt Programs



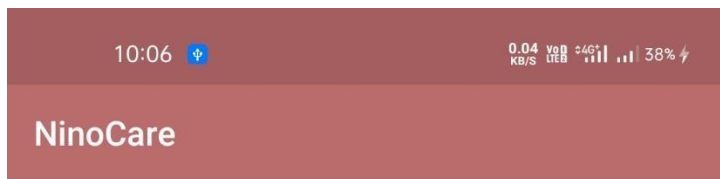
The screenshot shows a web browser window with the URL `localhost:8080/NinoCareWeb/panchayath/GovtPrograms.jsp`. The page features a dark navigation bar with the Nino Care logo and links: HOME, PROFILE, ADD AS HA, PROGRAM, VIEW ASHA, VIEW PROGRAMS, ADD -, VIEW -, and LOGOUT. The main content area is titled 'GOV PROGRAMS' and contains a table with the following data:

Sl.No	Program Name	User	Details	Date
1	Foodcare	Elina	huhfrueh	2019-04-03
2	foodcareprojec	Aleena	food for children	2019-04-11
3	Baby	sommya	Project is focusing for hw to care a baby	2019-05-04

Login



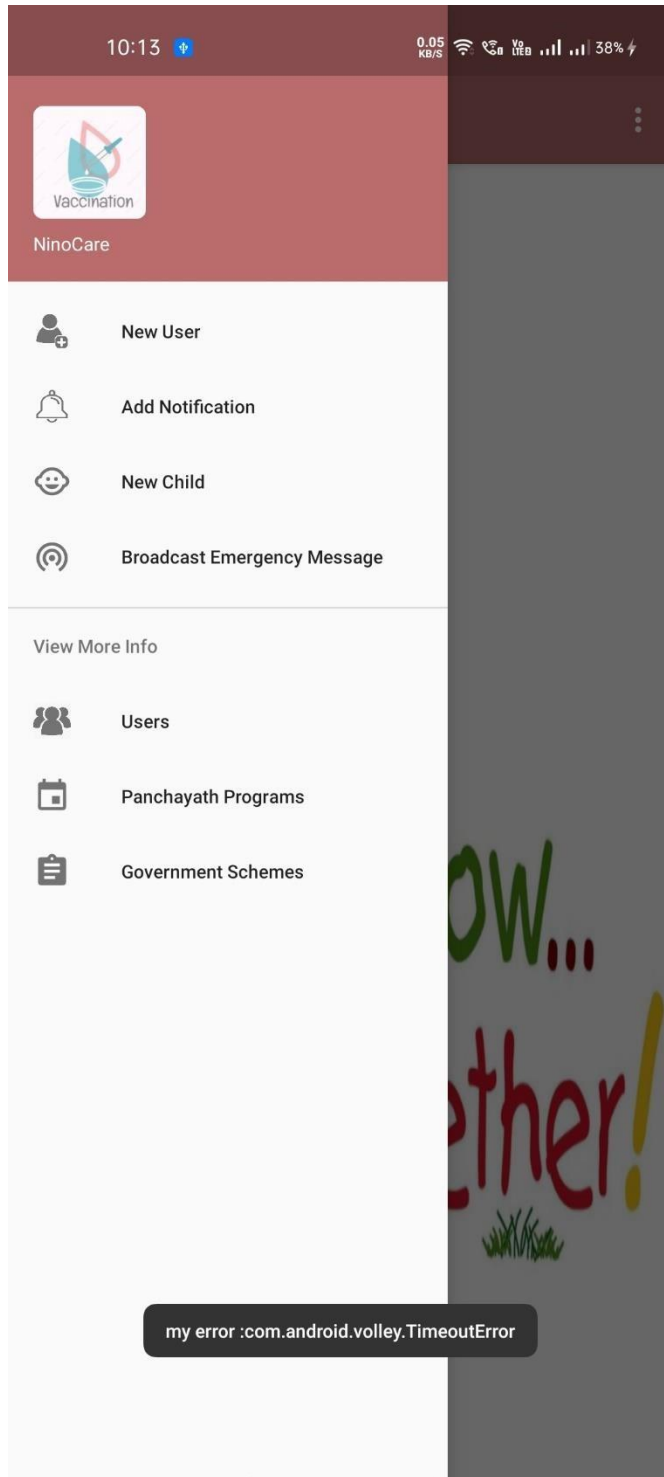
Forgot password



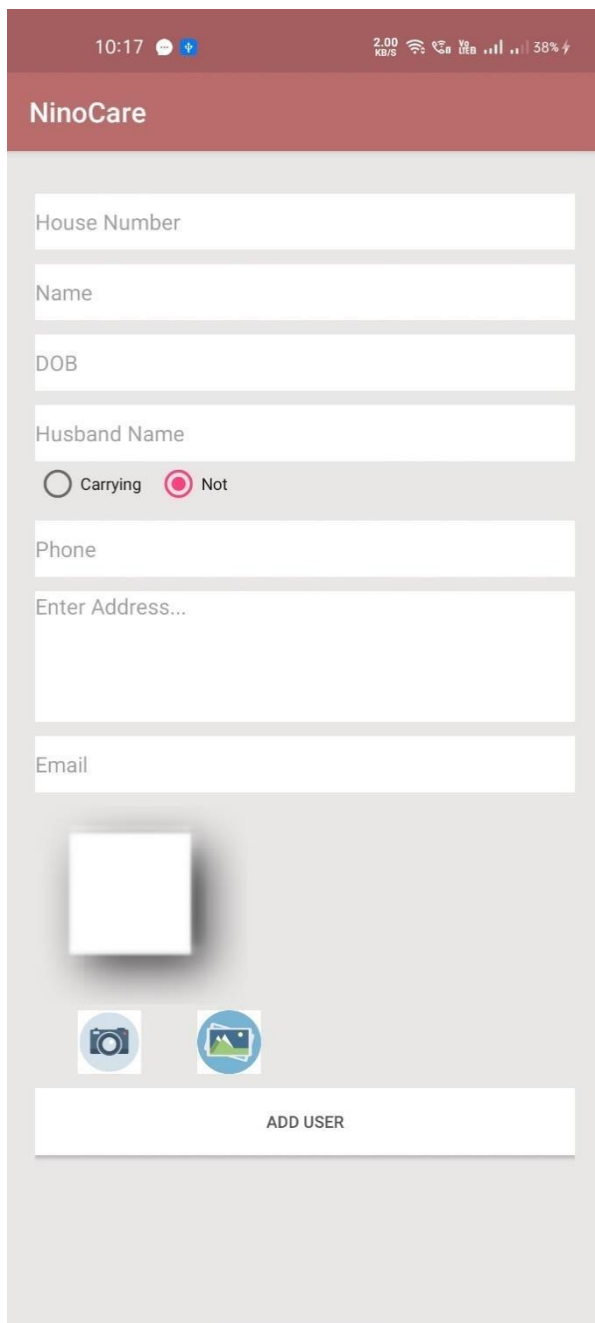
| Enter Email ID

SUBMIT

Asha Worker Home



Add Parent



The screenshot shows the 'Add Parent' form in the NinoCare app. The form is displayed on a mobile device screen with a status bar at the top showing the time 10:17, signal strength, and battery level at 38%. The app's header is red with the text 'NinoCare' in white. The form consists of several input fields: 'House Number', 'Name', 'DOB', 'Husband Name', 'Phone', 'Enter Address...', and 'Email'. Below the 'Husband Name' field, there are two radio buttons: 'Carrying' (unselected) and 'Not' (selected). Below the 'Enter Address...' field, there is a large white square placeholder for a profile picture. At the bottom of the form, there are two circular icons: a camera icon and a gallery icon. A white button labeled 'ADD USER' is positioned at the bottom of the form.

10:17 2.00 KB/s 38%

NinoCare

House Number

Name

DOB


Husband Name



☐ Carrying ☒ Not

Phone

Enter Address...

Email



ADD USER

Add Food

The screenshot displays the NinoCare mobile application interface. At the top, a status bar shows the time 10:17, signal strength, and battery level at 38%. Below the status bar is a red header with the text "NinoCare". The main content area is a light gray background with a white form. The form contains the following fields: "Select User Type", "Food Name", "Description", and "Food Receiving Date". Below these fields is a dark gray button labeled "SEND". At the bottom of the screen, there is a dark gray button labeled "Select User Type".

10:17 0.03 KB/S 38%

NinoCare

Select User Type

Food Name

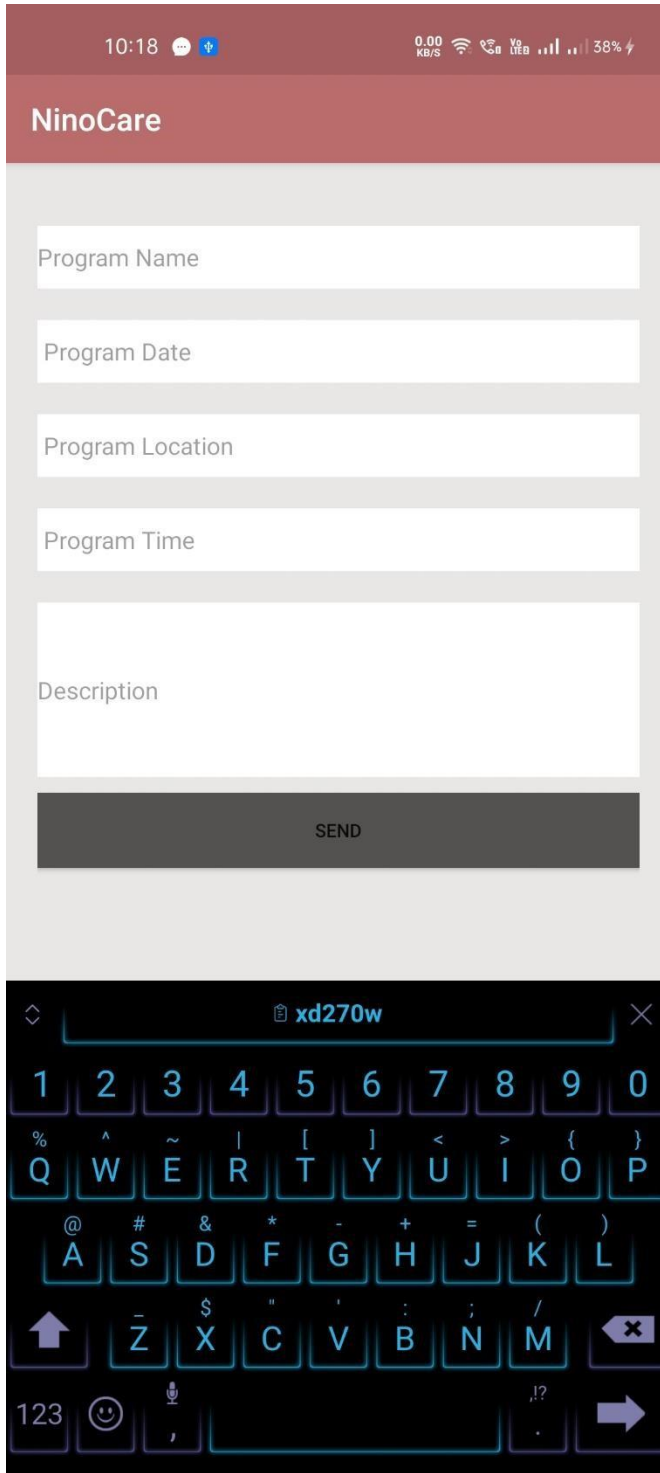
Description

Food Receiving Date

SEND

Select User Type

Add Programs



The screenshot displays the NinoCare mobile application interface. At the top, a red header bar contains the app name "NinoCare". Below this, a light gray background features five white input fields stacked vertically, labeled "Program Name", "Program Date", "Program Location", "Program Time", and "Description". A dark gray button labeled "SEND" is positioned below the "Description" field. At the bottom of the screen, a standard Android keyboard is visible, with a text suggestion "xd270w" appearing above the numeric row.

Add vaccination

The screenshot displays the NinoCare mobile application interface. At the top, the status bar shows the time 10:18, signal strength, and battery level at 38%. The app's header is a dark red bar with the text "NinoCare" in white. Below the header, there is a form with five input fields, each with a light gray border and a light gray background. The fields are labeled "Select User Type", "Vaccination name", "Vaccination Date", "Vaccination Location", and "Vaccination Time". Below these fields is a dark gray button with the text "SEND" in white. At the bottom of the screen, a virtual keyboard is visible, featuring a dark background with light blue and white keys. A small gray tooltip with the text "Please Select User" is positioned over the keyboard, specifically near the spacebar area.

10:18 0.15 KB/S 38%

NinoCare

Select User Type

Vaccination name

Vaccination Date

Vaccination Location

Vaccination Time

SEND

1 2 3 4 5 6 7 8 9 0

@ # \$ % & _ - () =

{&= " * Please Select User ? + ✕

abc , _ . ➔

Broad Cast Emergency Message

10:19 0.00 KB/S 38%

NinoCare

Subject

Message

SEND

Ben the I

1 2 3 4 5 6 7 8 9 0

% ^ ~ | [\ < > { }

q w e r t y u i o p

@ # & * - + = ()

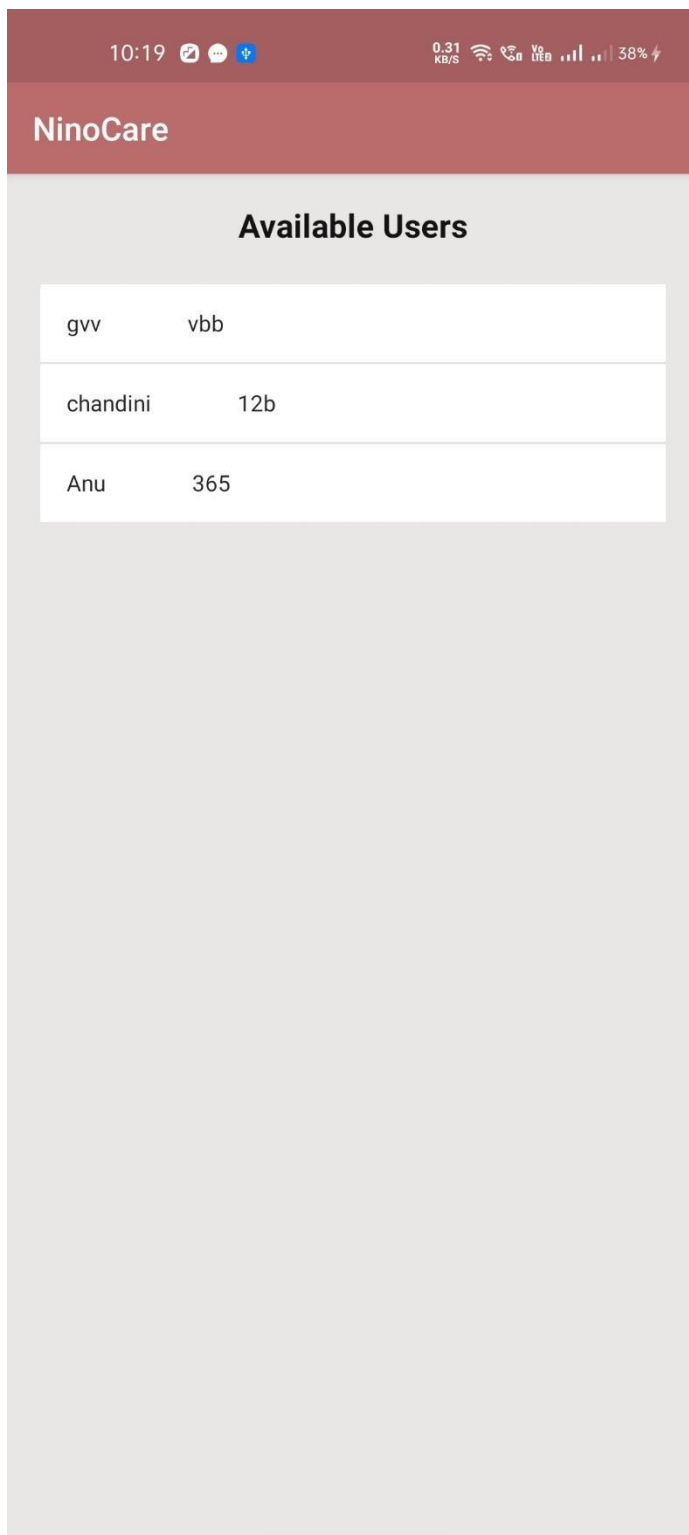
a s d f g h j k l

↑ _ \$ " ' : ; / ↵

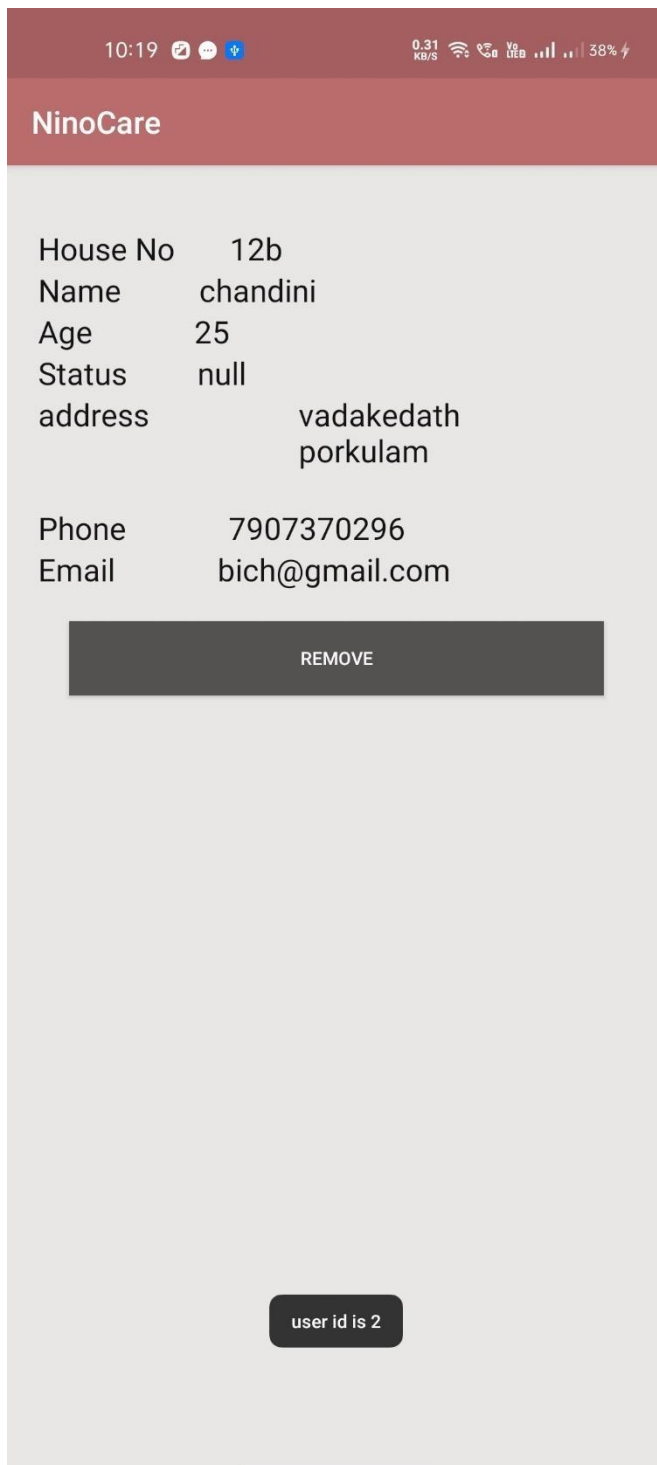
123 😊 , . ,!? ➡

failed !

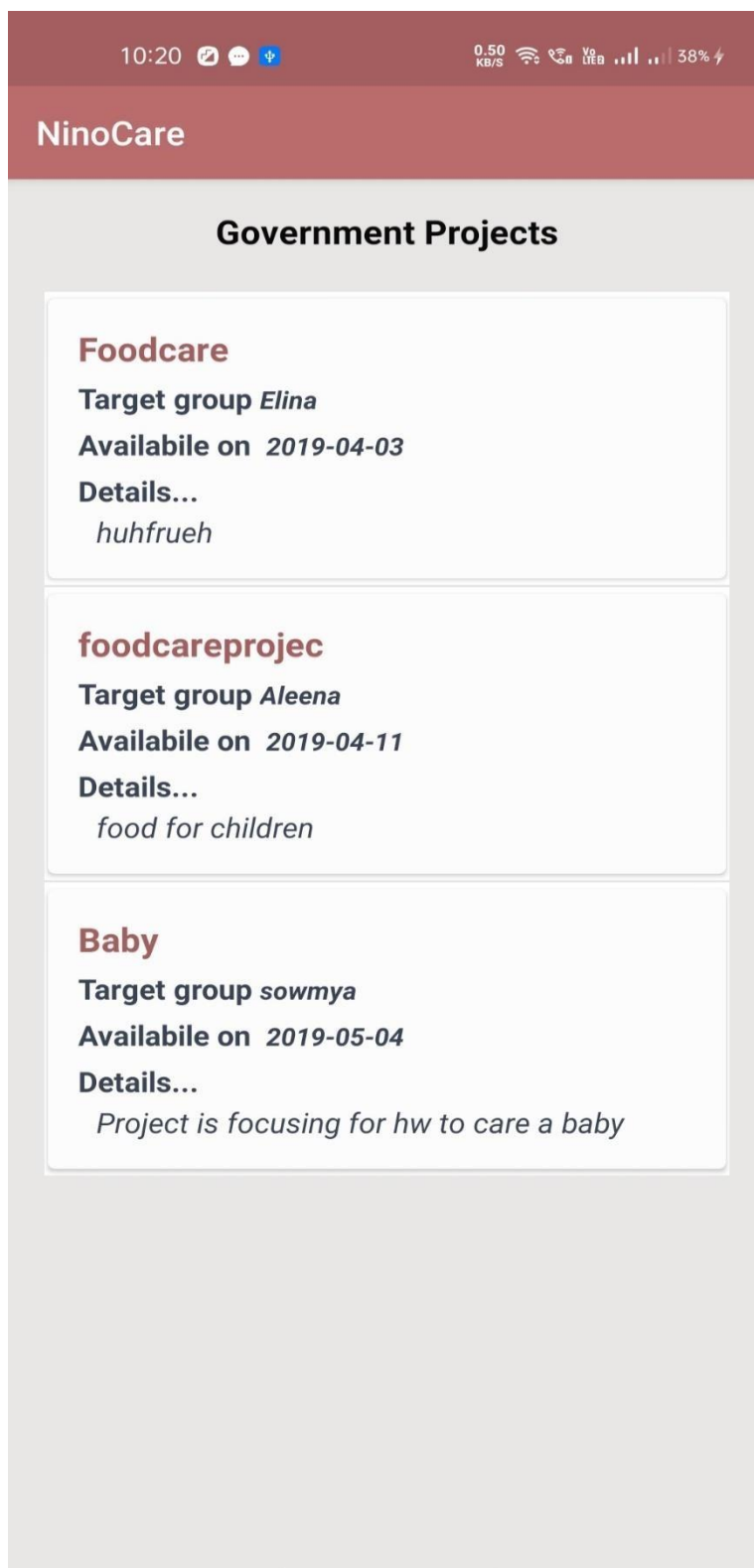
Available Users



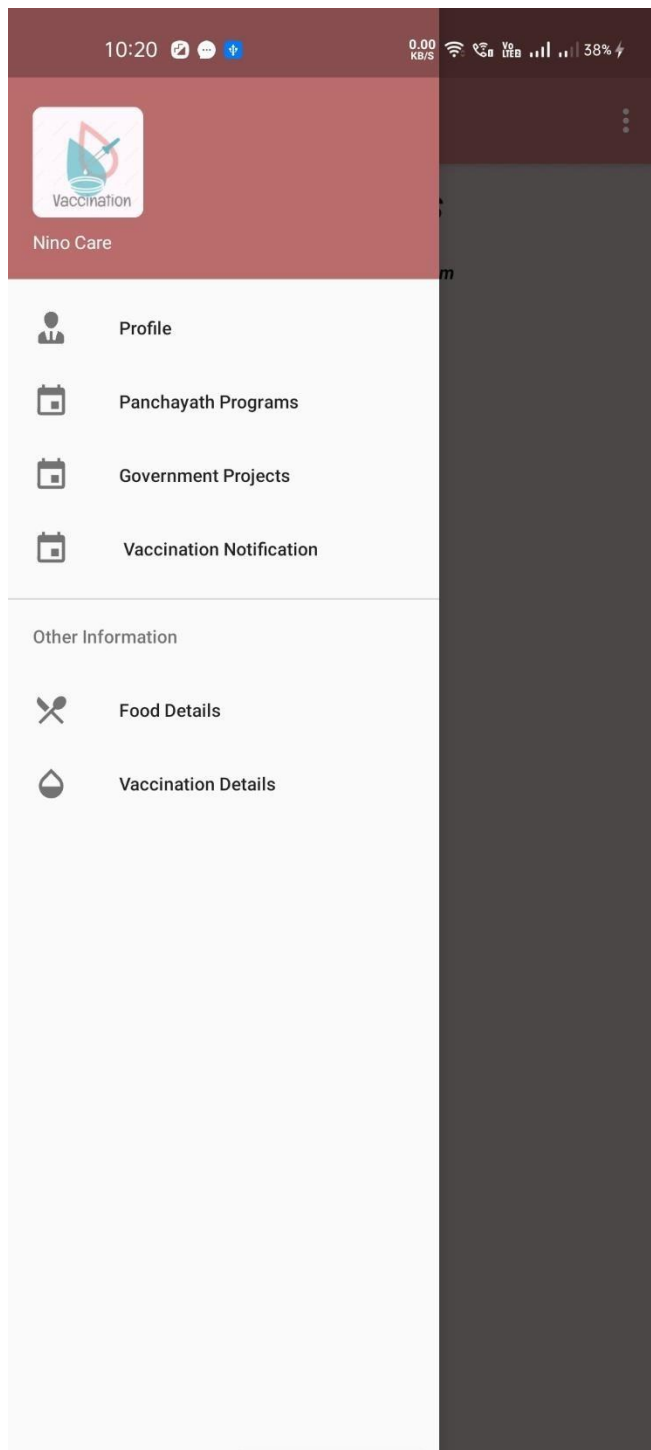
View User Location



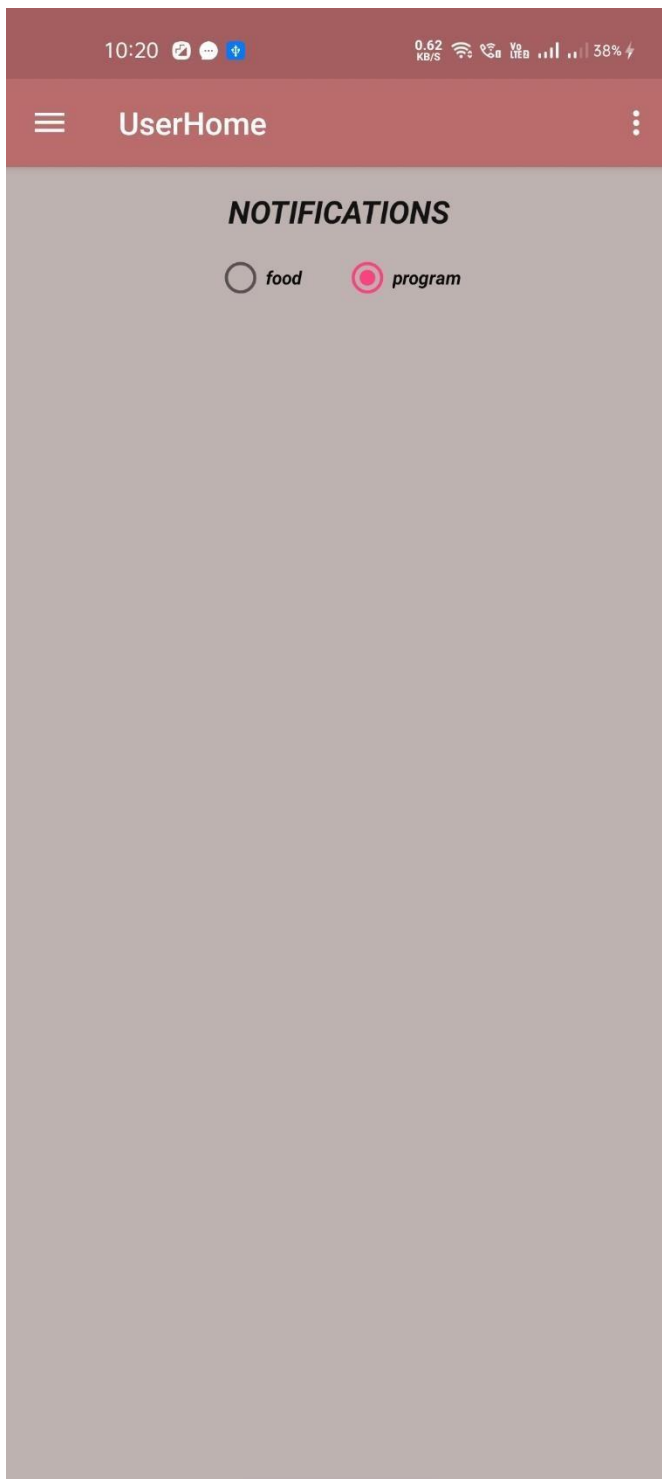
Govt Projects



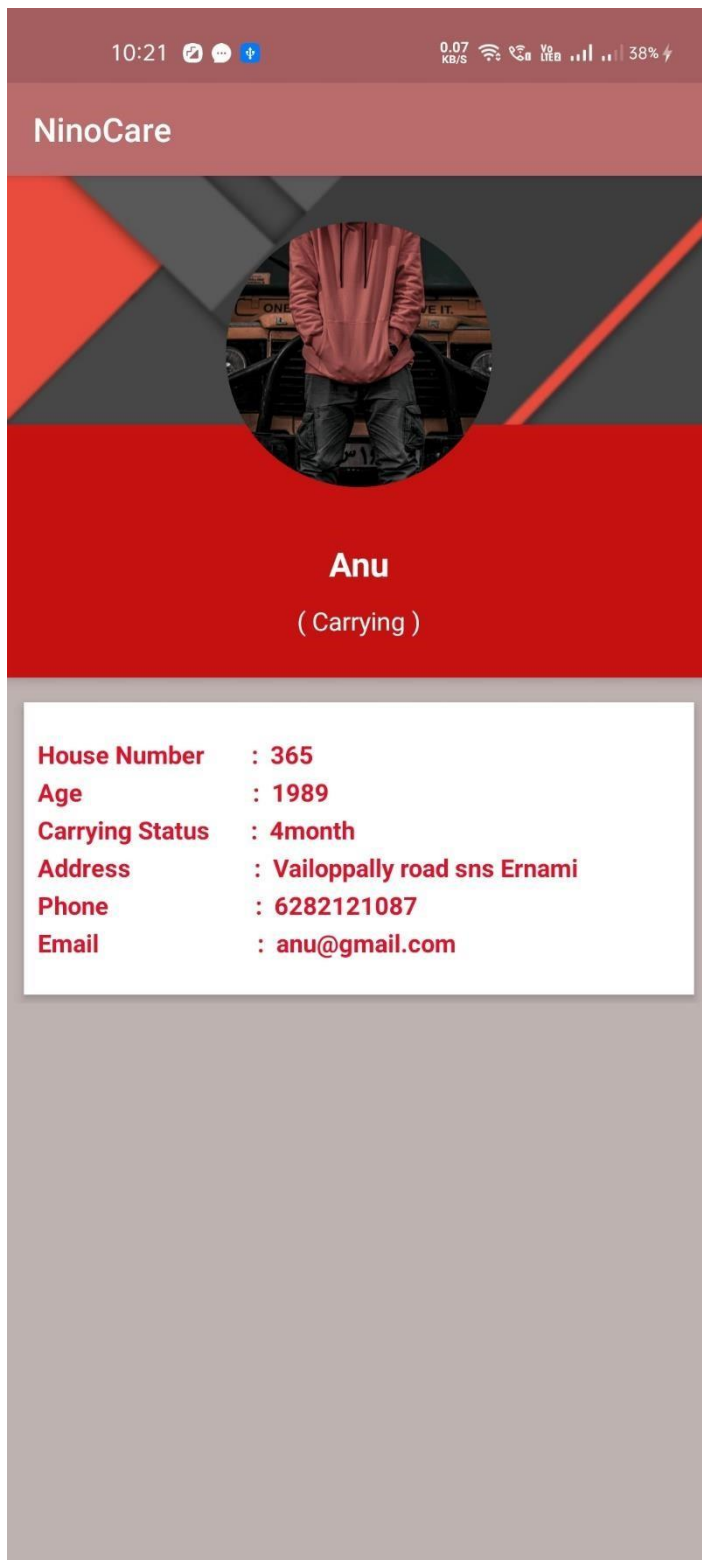
User Home



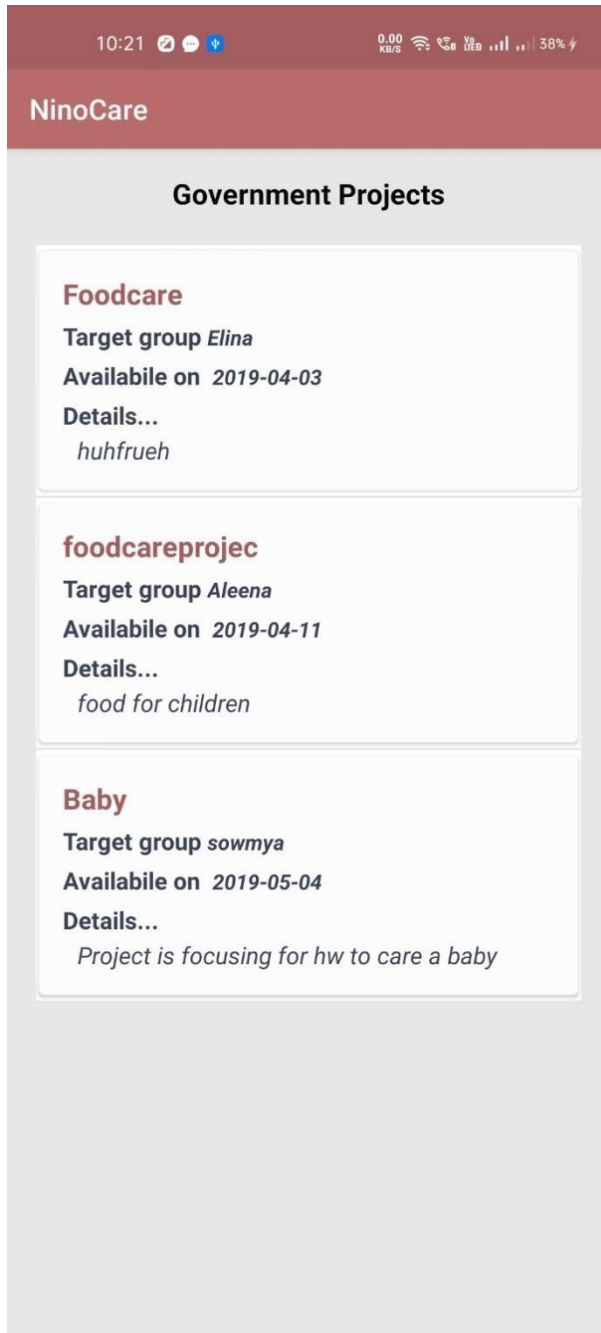
Notification



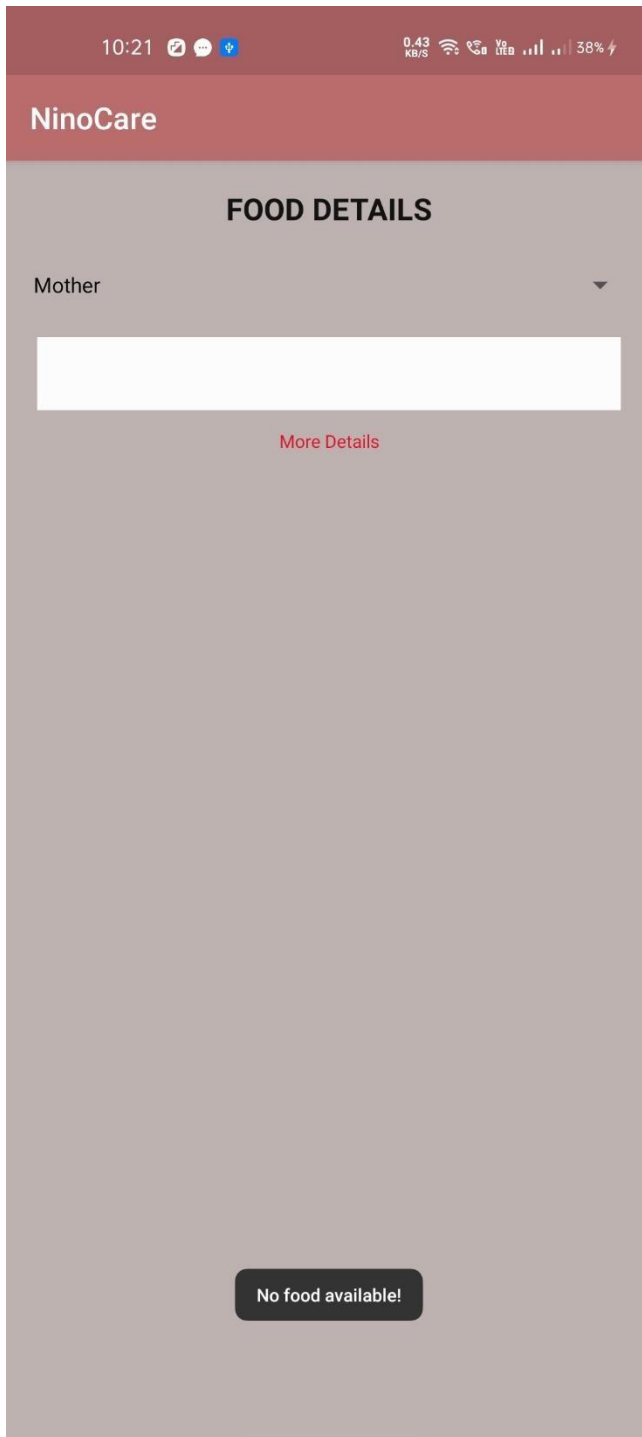
[View Profile](#)



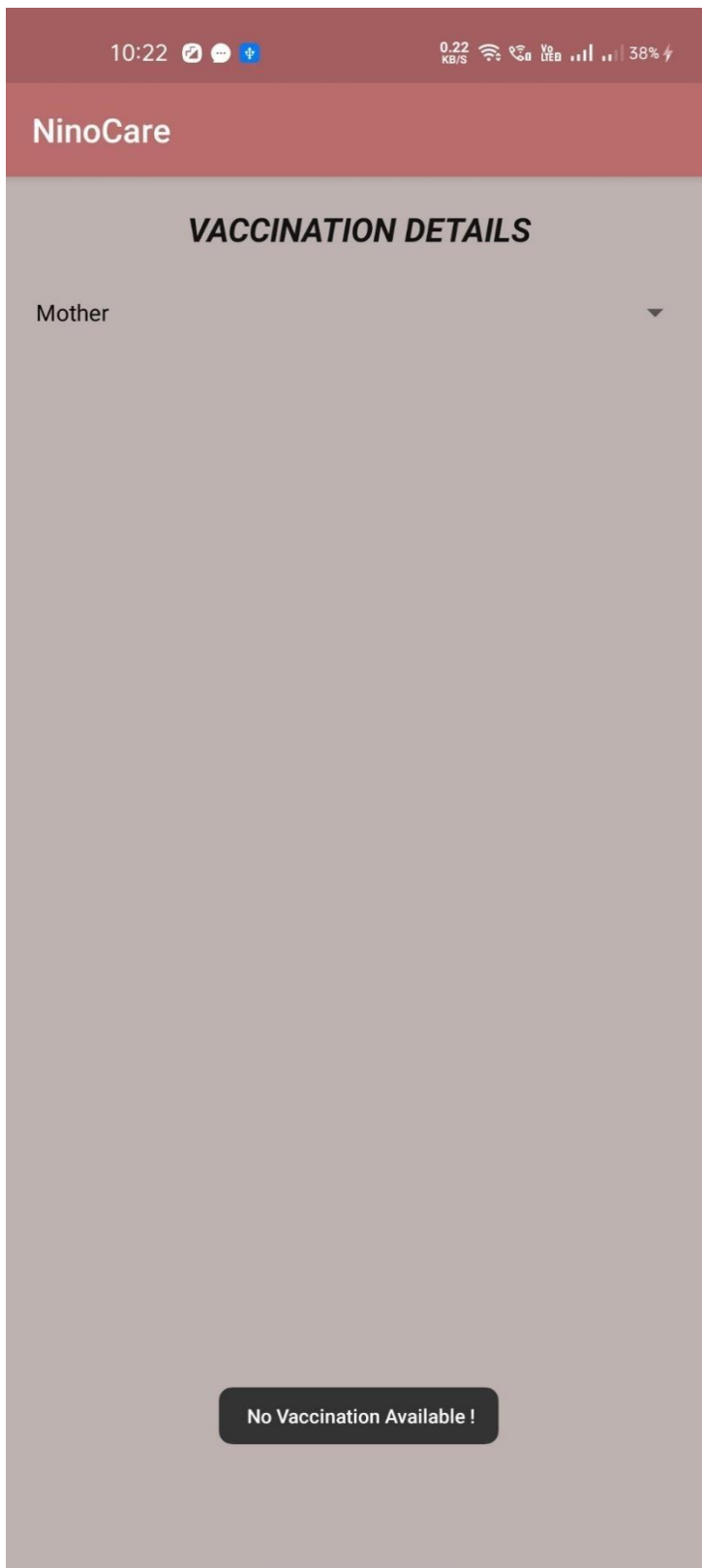
View Govt. Projects



Food Details



Vccination Details



PSUEDO CODE

Sample Core Code

Utility.java

```
package com.syntax.ninocare.Common;
```

```
import java.util.Random;
```

```
public class Common {
```

```
    //public static String url = "http://192.168.137.75:8080/NinoCareWeb/android/Ninoserver.jsp";  
    public static String url = "http://192.168.43.191:8080/NinoCareWeb/android/Ninoserver.jsp";
```

```
    public static char[] OTP(int len)
```

```
    {
```

```
        String numbers = "0123456789";
```

```
        Random rndm_method = new Random();
```

```
        char[] otp = new char[len];
```

```

    for (int i = 0; i < len; i++) {
        otp[i] = numbers.charAt(rndm_method.nextInt(numbers.length()));
    }
    return otp;
}
}

```

activity_login.xml

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/img1"
    android:gravity="center_vertical"    android:orientation="vertical"
    tools:context=".Login">

```

```

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"    android:layout_margin="20dp"
        android:orientation="vertical">

```

```

        <android.support.design.widget.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

```

```

            <android.support.design.widget.TextInputEditText
                android:id="@+id/login_edt_email"
                android:layout_width="match_parent"
                android:layout_height="55dp"
                android:layout_marginBottom="10dp"
                android:background="@color/white"    android:hint="UserName"
            />

```

```

        </android.support.design.widget.TextInputLayout>

```

```

            <android.support.design.widget.TextInputLayout
                android:layout_width="match_parent"    android:layout_height="wrap_content"
                app:passwordToggleDrawable="@drawable/ic_remove_red_eye_black_24dp"
                app:passwordToggleEnabled="true"
            />

```

```
app:passwordToggleTint="#111">
```

```
    <android.support.design.widget.TextInputEditText
        android:id="@+id/login_edt_pass"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/white"
        android:hint="Password"
        android:inputType="textPassword"
        android:paddingVertical="20dp" />
```

```
</android.support.design.widget.TextInputLayout>
```

```
    <Button
        android:id="@+id/btnlog"
        android:layout_width="match_parent"
        android:layout_height="55dp"
        android:layout_gravity="center_vertical|center_horizontal"
        android:layout_marginTop="10dp"
        android:layout_marginBottom="10dp"
        android:background="@color/signin"        android:text="
        SIGN IN "        android:textColor="@color/white" />
```

```
    <TextView
        android:id="@+id/login_forgotpassword"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="15dp"
        android:gravity="center"
        android:text="FORGOT PASSWORD"
        android:textColor="@color/white"
        android:textSize="20dp"
        android:textStyle="bold" />
```

```
</LinearLayout>
```

```
</LinearLayout>
```

```
Login.java package
com.syntax.ninocare;
```

```
import android.Manifest; import
android.content.Context; import
android.content.Intent; import
android.content.SharedPreferences; import
android.content.pm.PackageManager; import
android.os.Build; import
android.support.v4.app.ActivityCompat; import
android.support.v7.app.ActionBar; import
android.support.v7.app.AppCompatActivity;
import android.os.Bundle; import android.util.Log;
import android.view.View; import
android.widget.Button; import
android.widget.EditText; import
android.widget.TextView; import
android.widget.Toast;
```

```
import com.android.volley.AuthFailureError; import
com.android.volley.Request; import
com.android.volley.Response; import
com.android.volley.VolleyError; import
com.android.volley.toolbox.StringRequest; import
com.android.volley.toolbox.Volley; import
com.syntax.ninocare.AshaWorker.AshaHome; import
com.syntax.ninocare.Common.Common; import
com.syntax.ninocare.User.UserHome;
```

```
import java.util.HashMap;
import java.util.Map;
```

```
public class Login extends AppCompatActivity {
    EditText username,password;
    TextView tvreg,forgot;
    Button btnlog;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        int PERMISSION_ALL = 1;
        String[] PERMISSIONS = {
//            Manifest.permission.ACCESS_FINE_LOCATION,
//            Manifest.permission.ACCESS_COARSE_LOCATION,
            Manifest.permission.ACCESS_NETWORK_STATE,
            Manifest.permission.RECEIVE_SMS,
```

```

        Manifest.permission.READ_SMS,
        Manifest.permission.SEND_SMS,
        Manifest.permission.CAMERA,
        Manifest.permission.WRITE_EXTERNAL_STORAGE,
        Manifest.permission.READ_EXTERNAL_STORAGE,
        Manifest.permission.READ_PHONE_STATE,
    };

    if (!hasPermissions(this, PERMISSIONS)) {
        ActivityCompat.requestPermissions(this, PERMISSIONS, PERMISSION_ALL);
    }

    ActionBar actionBar=getSupportActionBar();
    actionBar.hide();
    username=(EditText)findViewById(R.id.login_edt_email);
    password=(EditText)findViewById(R.id.login_edt_pass);
    btnlog=(Button)findViewById(R.id.btnlog);    forgot=
    (TextView)findViewById(R.id.login_forgotpassword);
    btnlog.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            volley_call_login();
        }
    });

    forgot.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

            startActivity(new Intent(getApplicationContext(),Forgot.class));
        }
    });
}

public void volley_call_login()
{

    com.android.volley.RequestQueue queue = Volley.newRequestQueue(getApplicationContext());
    StringRequest request = new StringRequest(Request.Method.POST, Common.url, new
    Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            Log.d("*****",response);
            if(!response.trim().equals("failed"))

```

```

        {
            String data=response;
            String arr[]=data.trim().split(":");

//            Toast.makeText(getApplicationContext(), "Success="+id,
Toast.LENGTH_LONG).show();

            SharedPreferences.Editor editor = getSharedPreferences("SharedData",
MODE_PRIVATE).edit();
            editor.putString("loginid", arr[0]);
            editor.putString("type", arr[1]);
            editor.commit();

            if(arr[1].equals("user")) {
                Intent i = new Intent(Login.this, UserHome.class);
startActivity(i);
            }
            else if(arr[1].equals("asha")){
                Intent i = new Intent(Login.this, AshaHome.class);
startActivity(i);
            }
        }
    else
    {
        Toast.makeText(getApplicationContext(), "You entered an Invalid UserName or Password
!", Toast.LENGTH_LONG).show();
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {

        Toast.makeText(getApplicationContext(), "my error :" + error,
Toast.LENGTH_LONG).show();
        Log.i("My error", "" + error);
    }
}) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {

        Map<String, String> map = new HashMap<String, String>();
//        SharedPreferences sp=getSharedPreferences("booking_info", Context.MODE_PRIVATE);
map.put("key", "login");        map.put("uname",username.getText().toString().trim());
map.put("pass",password.getText().toString().trim());

```



```

        return map;
    }
};
queue.add(request);
}
public static boolean hasPermissions(Context context, String... permissions) {
    if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.M && context != null &&
permissions != null) {
        for (String permission : permissions) {
            if (ActivityCompat.checkSelfPermission(context, permission) !=
PackageManager.PERMISSION_GRANTED) {
                return false;
            }
        }
    }
    return true;
}
}

```

NinoServer.jsp

```

<% @page import="common.VacciBean"%> <% @page
import="java.time.Period"%>
<% @page import="java.time.LocalDate"%>
<% @page import="com.google.gson.Gson"%>
<% @page import="java.util.List"%>
<% @page import="java.util.ArrayList"%>
<% @page import="common.NotificationBean"%>
<% @page import="java.util.Date"%>
<% @page import="java.text.SimpleDateFormat"%>
<% @page import="java.util.Vector"%>
<% @page import="java.util.Iterator"%>
<%
    dbConnection.DbConnection con = new dbConnection.DbConnection();

    String key = request.getParameter("key").trim();
    System.out.println("key=" + key);

```

```
//login
if (key.equals("login")) {
String id = "";
    String qry = "select UserId,Type from login where UserName=" + request.getParameter("uname")
+ "and Password=" + request.getParameter("pass") + " ";
    String username = request.getParameter("uname");
    System.out.println("qry=" + qry);
    Iterator it = con.getData(qry).iterator();
    if (it.hasNext()) {
        Vector v = (Vector) it.next();
        id = "" + v.get(0) + ":" + v.get(1) + "";
        System.out.println("id" + id);
        out.print(id);
    } else {
        System.out.println("else id=" + id);
        out.print("failed");
    }
}
```

```
//getphoneForgot
if (key.equals("getphoneForgot")) {

    System.out.print("hihi"); //
    String id = "",qry1="";
    //    String qry = "select UserId,Type from login where UserName=" + request.getParameter("email")
+ " ";
    //    System.out.println("qry=" + qry);
    //    Iterator it = con.getData(qry).iterator();
    //    if (it.hasNext()) {
    //        Vector v = (Vector) it.next();
    //        id = v.get(0).toString();
    //
    //        if(id.equals("asha")){
    //            qry1="SELECT `Phone` FROM `regasha`WHERE `Email`=" +
request.getParameter("email") + " "; //            }else if(id.equals("user")){
//            qry1="SELECT `phone` FROM `tbl_user`WHERE `email`=" + request.getParameter("email")
+ " ";
//
//            }else{
//                out.print("failed");
//            }
//        }
//    }
}
```

```

//      }
//
//      System.out.println(qry1);
//
//      Iterator it1 = con.getData(qry1).iterator();
//      if (it1.hasNext()) {
//          Vector v1 = (Vector) it1.next();
//          String phone = v1.get(0).toString();
//
//          out.print(phone);
//      }
//
//
//      } else {
//          System.out.println("else id=" + id);
//          out.print("failed");
//      }
//  }
}

//add users
if (key.equals("AshaAddUser")) {

    String aid = request.getParameter("aid");
    String name = request.getParameter("name");
    String housenumber = request.getParameter("house");
    String husname = request.getParameter("husname");
    String month = request.getParameter("month");
    String phone = request.getParameter("phone");
    String email = request.getParameter("email");
    String dob = request.getParameter("dob");
    String address = request.getParameter("address");
    String image = request.getParameter("image");
    System.out.println("image" + image);
    String pass = new String(common.Utilities.OTP(6));

    Date date = new Date();
    SimpleDateFormat format = new SimpleDateFormat("YYYY-MM-dd");
    String strDate = format.format(date);

    String qry = "INSERT INTO `tbl_user`
(`aid`,`hnumber`,`name`,`dob`,`carring_month`,`address`,`phone`,`email`,`photo`,`date`) VALUES('" + aid +
"," + housenumber + "," + name + "," + dob + "," + month + "," + address + "," + phone + "," +
email + "," + image + "," + strDate + "') ";

```

```
String qry1 = "insert into login (UserId,username,password,type)values((select max(uid)from
tbl_user)," + email + "," + pass + "','user')";

    if (con.putData(qry) > 0) {
if (con.putData(qry1) > 0) {
    out.print(pass);
    }
    } else {
    out.print("failed");
    }
    }

//add childs
if (key.equals("AshaAddChildUser")) {

    String aid = request.getParameter("aid");
    String house = request.getParameter("house");
    String mid = request.getParameter("mid");
    String name = request.getParameter("name");
    String gender = request.getParameter("gender");
    String dob = request.getParameter("dob");

    Date date = new Date();
    SimpleDateFormat format = new SimpleDateFormat("YYYY-MM-dd");
    String strDate = format.format(date);

    String qry = "INSERT INTO `tbl_childregister` (aid,hnumber,mid,name,gender,dob,date)
VALUES('" + aid + "','" + house + "','" + mid + "','" + name + "','" + gender + "','" + dob + "','" + strDate
+ "') ";

    System.out.println(qry);

    if (con.putData(qry) > 0) {

        out.print("successful");

    } else {
        out.print("failed");
    }
    }

//ashaworker broadcasting emergency messages
if (key.equals("AshaBroadMsg")) {
    System.out.print("haii");
}
```

```

        String aid = request.getParameter("aid");
        String sub = request.getParameter("sub");
        String msg = request.getParameter("msg");
        String qry1 = "SELECT phone FROM `tbl_user` WHERE aid=" + aid;
        String phn = "";
        Iterator it1 = con.getData(qry1).iterator();
        while (it1.hasNext()) {
            Vector v = (Vector) it1.next();
            phn += v.get(0) + "#";
            System.out.println();
        }
        String qry = "INSERT INTO `tbl_emergmsg` (`aid`,`subject`,`message`) VALUES('" + aid + "','" +
        sub + "','" + msg + "') ";

        System.out.print(qry);

        if (con.putData(qry) > 0) {

            out.print(phn);
        } else {
            out.print("failed");
        }
    }

//to get registered mother names in one house
if (key.equals("AshaGetMother")) {
    System.out.print("haii");
    String aid = request.getParameter("aid");
    String hnumber = request.getParameter("houseNo");
    String qry1 = "SELECT uid,NAME FROM `tbl_user` WHERE hnumber=" + hnumber + " AND
aid=" + aid;
    System.out.println(qry1);
    String data = "", id = "", name = "";
    Iterator it1 = con.getData(qry1).iterator();
    if (it1.hasNext()) {
        while
        (it1.hasNext()) {
            Vector v = (Vector) it1.next();
            id += v.get(0) + ":";
            name += v.get(1) + ":";
        }
        data = id + "#" + name;
        out.print(data);
    } else {
        out.print("NotFound");
    }
}

```

```
}

//*****Notifications***** //Food*****Notification
if (key.equals("AshaFoodNotification")) {

    String aid = request.getParameter("aid");
    String usertype = request.getParameter("usertype");
    String foodname = request.getParameter("foodname");
    String descript = request.getParameter("descript");
    String fdate = request.getParameter("fdate");

    String qry = "INSERT INTO `tbl_food_notification` (aid,usertype,foodname,descript,date)
VALUES('" + aid + "','" + usertype + "','" + foodname + "','" + descript + "','" + fdate + "') ";
    System.out.println(qry);

    if (con.putData(qry) > 0) {

        out.print("successful");

    } else {
        out.print("failed");
    }
}
//
// //Program Notifications
if (key.equals("AshaProgramNotification")) {

    String aid = request.getParameter("aid");
    String progname = request.getParameter("progname");
    String pdate = request.getParameter("pdate"); String
    plocat = request.getParameter("plocat");
    String ptime = request.getParameter("ptime");
    String pdescript = request.getParameter("pdescript");

    String qry = "INSERT INTO `tbl_program_notification`
(aid,progname,pdate,plocation,ptime,description) VALUES('" + aid + "','" + progname + "','" + pdate +
',' + plocat + "','" + ptime + "','" + pdescript + "') ";

    System.out.println(qry);

    if (con.putData(qry) > 0) {

        out.print("successful");
```

```

    } else {
        out.print("failed");
    }
}

if (key.equals("AshaViewPanPrograms")) {

    String aid = request.getParameter("aid");
    List<VacciBean> allVacciInfos = new ArrayList<VacciBean>();

    String data = "";
    String info = "";

    String qry = "SELECT p.ProgName,p.PanProLoc,p.PanProDate,p.progtime,p.ProgDet FROM
panprograms p,regasha a WHERE p.PID=a.PID AND a.AID='" + aid + "'";
    System.out.println(qry);
    Iterator it1 = con.getData(qry).iterator();
    if (it1.hasNext()) {        while
(it1.hasNext()) {
        Vector v = (Vector) it1.next();
//        data = "Prgm Name : " + v.get(0) + "\nLocation      : " + v.get(1) + "\nDate          : " +
v.get(2) + "\nTime          : " + v.get(3) + "\nDetails      : " + v.get(4) + "\n";
//        System.out.println(data);
//        info += data + "#";
//    }
//    out.print(info);
//    } else {
//        out.print("failed");
//    }
//    }
//    }

    VacciBean vbean = new VacciBean();
    vbean.setProgName(v.get(0).toString());
    vbean.setPanProLoc(v.get(1).toString());
    vbean.setPanProDate(v.get(2).toString());
    vbean.setProgtime(v.get(3).toString());
    vbean.setProgDet(v.get(4).toString());        allVacciInfos.add(vbean);
    }
    Gson gson = new Gson();

    info = gson.toJson(allVacciInfos);

```

```

        System.out.println(info);
    if (info.isEmpty()) {
    info = "failed";
        }
        out.println(info);
    } else {
        System.out.println(info);
    out.println("failed");
    }
    }
    if (key.equals("UserViewPanPrograms")) {

        String uid = request.getParameter("uid");
        List<VacciBean> allVacciInfos = new ArrayList<VacciBean>();

        String data = "";
        String info = "";

        String qry = "SELECT p.ProgName,p.PanProLoc,p.PanProDate,p.progtime,p.ProgDet FROM
panprograms p,regasha a,tbl_user u WHERE u.`aid`=a.`AID` AND a.`PID`=p.`PID` AND u.`uid`=" +
uid;
        System.out.println(qry);
        Iterator it1 = con.getData(qry).iterator();
        if (it1.hasNext()) {            while
        (it1.hasNext()) {
            Vector v = (Vector) it1.next();
            //      data = "Prgm Name      : " + v.get(0) + "\nLocation      : " + v.get(1) + "\nDate
: " + v.get(2) + "\nTime          : " + v.get(3) + "\nDetails          : " + v.get(4) + "\n" + "#"; //
            System.out.println(data);
            //      info += data;
            //      }
            //      out.print(info);
            //      } else {
            //      out.print("failed");
            //
            //      }

            VacciBean vbean = new VacciBean();
            vbean.setProgName(v.get(0).toString());
            vbean.setPanProLoc(v.get(1).toString());
            vbean.setPanProDate(v.get(2).toString());
            vbean.setProgtime(v.get(3).toString());
            vbean.setProgDet(v.get(4).toString());          allVacciInfos.add(vbean);
            }
            Gson gson = new Gson();

```



```
        info = gson.toJson(allVacciInfos);

        System.out.println(info);
        if (info.isEmpty()) {
            info = "failed";
        }
        out.println(info);
    } else {
        System.out.println(info);
        out.println("failed");
    }
}

if (key.equals("AshaViewgovPrograms")) {

    List<VacciBean> allVacciInfos = new ArrayList<VacciBean>();

    String data = "";
    String info = "";

    String qry = "SELECT pro_name,pro_user,pro_det,pro_date FROM admprojects ";
    System.out.println(qry);
    Iterator it1 = con.getData(qry).iterator();
    if (it1.hasNext()) {        while
    (it1.hasNext()) {
        Vector v = (Vector) it1.next();
        //      data = "Project Name : " + v.get(0) + "\nProject User  " + v.get(1) + "\nProject Details " +
        v.get(2) + "\nProject Date " + v.get(3) + "\n";
        // System.out.println(data);
        //      info += data + "#";
    //bean class
        VacciBean vbean = new VacciBean();
        vbean.setPro_name(v.get(0).toString());
        vbean.setPro_user(v.get(1).toString());
        vbean.setPro_date(v.get(3).toString());
        vbean.setPro_det(v.get(2).toString());

        allVacciInfos.add(vbean);
    }
    Gson gson = new Gson();

    info = gson.toJson(allVacciInfos);
```

```

        System.out.println(info);
    if (info.isEmpty()) {
    info = "failed";
        }
        out.println(info);
    } else {
        System.out.println(info);
    out.println("failed");
    }
    }

    if (key.equals("UserViewVacciDetails")) {

        String uid = request.getParameter("uid");
        String usertype = request.getParameter("usertype");
        List<VacciBean> allVacciInfos = new ArrayList<VacciBean>();

        String data = "";
        String info = "";
        String qry = "SELECT v.vid,v.vacci_name,v.vacci_age,v.details FROM tbl_vaccination v,regasha
a,tbl_user u WHERE u.aid=a.AID AND a.PID=v.PID AND u.uid=" + uid + " AND v.usertype=" +
usertype + """;
        System.out.println(qry);
        Iterator it1 = con.getData(qry).iterator();
        String vname = " ", age = " ", det = " ";
        String testdata = "";
        if (it1.hasNext()) {

            while (it1.hasNext()) {
                Vector v = (Vector) it1.next();
                //      vname += "Vacci Name  :" + v.get(0);
                //      age += "Age      :" + v.get(1);
                //      det += "Details   :" + v.get(2);
                //      System.out.println(vname);
                //      System.out.println(age); //
                System.out.println(det);
                //      testdata = testdata + "Vaccin      " + v.get(0) + "\nAge      " + v.get(1) + "\nDetails
" + v.get(2) + "#";
                VacciBean vbean = new VacciBean();
                vbean.setVacci_id(v.get(0).toString());
                vbean.setVacci_name(v.get(1).toString());
                vbean.setVacci_details(v.get(3).toString());
                allVacciInfos.add(vbean);
            }
            //      data += vname + "\n" + age + "\n" + det + "\n" + "#";

```

```

//      out.print(testdata);
//      System.out.println(testdata);
//  } else {
//      out.print("failed");
//
//  }
    Gson gson = new Gson();

    info = gson.toJson(allVacciInfos);

    System.out.println(info);
    if (info.isEmpty()) {
    info = "failed";
        }
        out.println(info);
    } else {
        System.out.println(info);
    out.println("failed");
    }
//    Sy
}
    if (key.equals("UserViewFoodDetails")) {
String uid = request.getParameter("uid");
    String usertype = request.getParameter("usertype");
    List<VacciBean> allVacciInfos = new ArrayList<VacciBean>();

    String data = "";
    String info = "";
    String fname = "", period = "", det = "";
    String qry = "SELECT f.FoodName,f.FoodPeriod,f.FoodDetails FROM tbl_food f,regasha
a,tbl_user u WHERE u.aid=a.AID AND a.PID=f.PID AND u.uid=" + uid + " AND f.FoodUserType="
+ usertype + "";
    System.out.println(qry);
    Iterator it1 = con.getData(qry).iterator();

    if (it1.hasNext()) {
while (it1.hasNext()) {
        Vector v = (Vector) it1.next();
//        fname += "Food Name :" + v.get(0); //
period += "Period      :" + v.get(1);
//        det += "Details      :" + v.get(2);
//        System.out.println(fname);
//        System.out.println(period);
//        System.out.println(det);

```

```

VacciBean vbean = new VacciBean();
vbean.setFoodName(v.get(0).toString());
vbean.setFoodPeriod(v.get(1).toString());
vbean.setFoodDetails(v.get(2).toString());
allVacciInfos.add(vbean);
    }
//    data += vname + "\n" + age + "\n" + det + "\n" + "#";
//    out.print(testdata);
//    System.out.println(testdata);
//    } else {
//        out.print("failed");
//    }
//    }
    Gson gson = new Gson();

    info = gson.toJson(allVacciInfos);

    System.out.println(info);
    if (info.isEmpty()) {
        info = "failed";
    }
    out.println(info);
    } else {
        System.out.println(info);
        out.println("failed");
    }
//    }
//    data += fname + "\n" + period + "\n" + det + "\n" + "#";
//    out.print(data);
//    } else {
//        out.print("failed");
//    }
}
//Asha requests users details      if
(key.equals("AshaViewUsersDetails")) {
String aid = request.getParameter("aid");
String usertype = request.getParameter("usertype");
String data = "";
String info = "";
String id = "", users = "", hnumber = "";
String qry = "SELECT uid,name, hnumber FROM tbl_user WHERE aid='" + aid + "'";
System.out.println(qry);
Iterator it1 = con.getData(qry).iterator();
if (it1.hasNext()) {      data = "Name
HouseNo";      while (it1.hasNext()) {

```

```

        Vector v = (Vector) it1.next();
id += v.get(0) + ":";
        users += "Name  : " + v.get(1) + "\n House Number  : " + v.get(2) + ":";
        //hnumber+=v.get(2) + "";
        System.out.println(id);
        System.out.println(users);
        System.out.println(hnumber);
    }
    data = id + "#" + users + "\n\n";
System.out.print(data.trim());
out.print(data.trim());
    } else {
        out.print("failed");
    }
}
//getUserDetails start

//Asha requests users details      if
(key.equals("getUserDetails")) {      String
uid = request.getParameter("uid");
    String users = "";
    String qry = "SELECT * FROM `tbl_user` WHERE uid=" + uid;
    System.out.println(qry);
    Iterator it1 = con.getData(qry).iterator();
if (it1.hasNext()) {      while
(it1.hasNext()) {
    Vector v = (Vector) it1.next();
    users += v.get(0) + ":" + v.get(2) + ":" + v.get(3) + ":" + v.get(4) + ":" + v.get(5) + ":" +
v.get(6) + ":" + v.get(7) + ":" + v.get(8) + ":" + v.get(9);
    System.out.println(users);
    }
    System.out.print(users);
out.print(users);
    } else {
        out.print("failed");
    }
}
//getUserDetails start

//start ashaDeleteUser
if (key.equals("ashaDeleteUser")) {
    String uid = request.getParameter("userid");
    String qry = "Delete FROM tbl_user WHERE uid=" + uid;
    System.out.println(qry);

```

```

        if (con.putData(qry) > 0) {

            System.out.println("Deleted");
        } else {
            out.print("failed");
        }
    }
//end ashaDeleteUser
//getFoodNotification start

    if (key.equals("getFoodNotification")) {
        System.out.println("haii");
        List<NotificationBean> allInfos = new ArrayList<NotificationBean>();

        String station = request.getParameter("uid");
        String aid="";
        String qry=" SELECT `aid` FROM `tbl_user` WHERE `uid`='"+station+"' ";
        Iterator it3 = con.getData(qry).iterator();
        if(it3.hasNext()){
            Vector v3 = (Vector) it3.next();
            aid=v3.get(0).toString();
        }

        String str = "SELECT `foodid`,`foodname`,`descript`,`date` FROM `tbl_food_notification` where
`aid`='"+aid+"'";
        //    String str = "SELECT `foodid`,`foodname`,`descript`,`date` FROM `tbl_food_notification` ";

        System.out.println("qry=" + str);
        String data = "";
        String infoall = "";
        Iterator it = con.getData(str).iterator();
        if (it.hasNext()) {            while
        (it.hasNext()) {
            Vector vv = (Vector) it.next();
            LocalDate today = LocalDate.now();
            String datee = vv.get(3).toString();
            LocalDate date = LocalDate.parse(datee);
            Period period = Period.between(today, date);
            if ((period.getDays() >= 0) && (period.getMonths() >= 0) && (period.getYears() >= 0)) {
                NotificationBean bean = new NotificationBean();
                bean.setFoodnot_id(vv.get(0).toString());
                bean.setFood_name(vv.get(1).toString());
                bean.setFood_date(vv.get(3).toString());
                bean.setFood_decription(vv.get(2).toString());                allInfos.add(bean);
            }
        }
    }

```

```

    }
    Gson gson = new Gson();

    infoall = gson.toJson(allInfos);

    System.out.println(infoall);
    if (infoall.isEmpty()) {
        infoall = "failed";
    }
    out.println(infoall);
    } else {
        System.out.println(infoall);
        out.println("failed");
    }
    // System.out.println(infoall);
    }

//getFoodNotification end //getProgramNotification
start
    if (key.equals("getProgramNotification")) {
        System.out.println("haii");
        List<NotificationBean> allInfos = new ArrayList<NotificationBean>();

        String station = request.getParameter("uid");
        String aid="";
        String qry=" SELECT `aid` FROM `tbl_user` WHERE `uid`='"+station+"' ";
        Iterator it3 = con.getData(qry).iterator();
        if(it3.hasNext()){
            Vector v3 = (Vector) it3.next();
            aid=v3.get(0).toString();
        }

        String str = "SELECT `progid`,`prognose`,`pdate`,`plocation`,`ptime`,`description` FROM
        `tbl_program_notification` where `aid`='"+aid+"'";
        //String str = "SELECT `progid`,`prognose`,`pdate`,`plocation`,`ptime`,`description` FROM
        `tbl_program_notification`;
        System.out.println("qry=" + str);

        String data = "";
        String infoall = "";
        Iterator it = con.getData(str).iterator();
        if (it.hasNext()) {
            while
            (it.hasNext()) {
                Vector vv = (Vector) it.next();
                LocalDate today = LocalDate.now();

```

```

        String datee = vv.get(2).toString();
        LocalDate date = LocalDate.parse(datee);
        Period period = Period.between(today, date);
        if ((period.getDays() >= 0) && (period.getMonths() >= 0) && (period.getYears() >= 0)) {
            NotificationBean bean = new NotificationBean();
            bean.setProgram_id(vv.get(0).toString());
            bean.setProgram_name(vv.get(1).toString());
            bean.setProgram_date(vv.get(2).toString());
            bean.setProgram_location(vv.get(3).toString());
            bean.setProgram_time(vv.get(4).toString());
            bean.setProgram_description(vv.get(5).toString());
            allInfos.add(bean);
        }
    }
    Gson gson = new Gson();

    infoall = gson.toJson(allInfos);

    System.out.println(infoall);
    if (infoall.isEmpty()) {
        infoall = "failed";
    }
    out.println(infoall);
    } else {
        System.out.println(infoall);
        out.println("failed");
    }
    //    System.out.println(infoall);
    }

//getProgramNotification end //NotificationDetails
start
    if (key.equals("NotificationDetails")) {

        String uid = request.getParameter("uid");
        String data = "";
        String info = "";
        String id = "", dd = "";
        String carry = "initvalue", aid, child_status = "mother";

        //...get user month and aid start...
        String str = "SELECT `carring_month`, `aid` FROM `tbl_user` WHERE `uid`=" + uid + "";
        Iterator it = con.getData(str).iterator();
        if (it.hasNext()) {
            Vector v1 = (Vector) it.next();
            carry = v1.get(0).toString();

```



```

        aid = v1.get(1).toString();
        System.out.println("carry,aid" + carry + aid);
    }
    //...get user month and aid end...

    //...check mother child start...
    String str2 = "SELECT `dob` FROM `tbl_childregister` WHERE `mid`=" + uid + """;
    Iterator it2 = con.getData(str2).iterator();
    if (it2.hasNext()) {
        Vector v2 = (Vector) it2.next();
        child_status = v2.get(0).toString();
        System.out.println("child_status" + child_status);
    }
    //...check mother child end...
    String qry = "SELECT
v.`vacci_name`,v.`usertype`,v.`vacci_age`,v.`details`,vn.`date`,vn.`location`,vn.`time`,vn.not_id FROM
`tbl_vacci_notification` vn,`tbl_vaccination` v WHERE vn.`vid`=v.vid ";
    //System.out.println(qry);

    Iterator it1 = con.getData(qry).iterator();
    if (it1.hasNext()) {
        while
        (it1.hasNext()) {
            Vector v = (Vector) it1.next(); //
            Checking viewed status or not
            String qry4check = "SELECT st_id FROM `tbl_notification_status` WHERE notf_id =" +
v.get(7);
            Iterator itstatus = con.getData(qry4check).iterator();
            if (itstatus.hasNext()) {
                } else { // .....
                    LocalDate today = LocalDate.now();
                    String datee = v.get(4).toString();
                    LocalDate date = LocalDate.parse(datee);
                    Period period = Period.between(today, date);
                    // .....
                    if ((period.getDays() >= 0) && (period.getMonths() >= 0) && (period.getYears() >= 0)) {

                        if (v.get(1).equals("Child") && !child_status.equals("mother")) {

                            LocalDate childdob = LocalDate.parse(child_status);
                            Period agediff = Period.between(childdob, today);
                            System.out.println("age diff" + agediff.getYears());
                            int vacciage = Integer.parseInt(v.get(2).toString());
                            if (agediff.getYears() == vacciage) {
                                id = id +
v.get(5) + ":";
                                data = "\nPrgm Name : " + v.get(0) + "\nLocation : " + v.get(1) + "\nDate

```

```

: " + v.get(2) + "\nTime          : " + v.get(3) + "\nDetails          : " + v.get(4) + "\n";
    dd += data + "#";
    }
    }
    if (v.get(1).equals("Mother")) {
        System.out.println("inside ifmother");

        LocalDate carrymonth = LocalDate.parse(carry);
        System.out.println("carrymonth" + carrymonth + " today" + today);
        Period monthdiff = Period.between(carrymonth, today);

        int vaccimonth = Integer.parseInt(v.get(2).toString());
        System.out.println("mo diff=" + monthdiff.getMonths() + "vaccimonth" +
vaccimonth);
        if (monthdiff.getMonths() == (vaccimonth)) {
id = id + v.get(5) + ":";
            data = "\nPrgm Name : " + v.get(0) + "\nLocation      : " + v.get(1) + "\nDate
: " + v.get(2) + "\nTime          : " + v.get(3) + "\nDetails          : " + v.get(4) + "\n";
            //System.out.println(data);
            dd += data + "#";
        }
    }
    }
    }
    info = id + "%" + dd;
    System.out.println(info);    if
    (info.equals("%")) {
        info = "failed";
    }
    out.print(info);

    } else {
        out.print("failed");
    }

}

}

//NotificationDetails end //getProfile
start
    if (key.equals("getProfile")) {

        String uid = request.getParameter("uid");

```

```

String data = "";
String qry = "SELECT * FROM `tbl_user` WHERE `uid`='" + uid + "'";
//System.out.println(qry);
String monthcarry;
Iterator it1 = con.getData(qry).iterator();
if (it1.hasNext()) {

    Vector v = (Vector) it1.next();
    monthcarry = v.get(5).toString();    if
    (!v.get(5).equals("null")) {
        LocalDate today = LocalDate.now();
        LocalDate carrymonth = LocalDate.parse(v.get(5).toString());
        System.out.println("carrymonth" + carrymonth + " today" + today);
        Period monthdiff = Period.between(carrymonth, today);
        int carrymon = monthdiff.getMonths();    monthcarry = ""
        + carrymon;
    }

    data = v.get(0) + ":" + v.get(2) + ":" + v.get(3) + ":" + v.get(4) + ":" + monthcarry + ":" +
    v.get(6) + ":" + v.get(7) + ":" + v.get(8) + ":" + v.get(9);

    System.out.println(data);

    out.print(data);
} else {
    out.print("failed");

}
}

//getProfile end //getVacc_Details
start
if (key.equals("getVacc_Details")) {
    System.out.print("haii");
    String aid = request.getParameter("aid");
    String user_type = request.getParameter("userType");
    String qry1 = "SELECT v.`vid`,v.`vacci_name` FROM `tbl_vaccination` v,`regasha` a WHERE
v.`usertype`='" + user_type + "' AND a.`PID`=v.`PID` AND a.`AID`='" + aid + "'";
    System.out.println(qry1);
    System.out.print(qry1);
    String data = "", id = "", name = "";
    Iterator it1 = con.getData(qry1).iterator();
    if (it1.hasNext()) {        while
    (it1.hasNext()) {

```

```

        Vector v = (Vector) it1.next();
id += v.get(0) + " ";      name +=
v.get(1) + " ";
    }
    data = id + "#" + name;
out.print(data);
    } else {
        out.print("failed");
    }

}

//getVacc_Detailsend

//addVaccNotifiacion start
if (key.equals("addVaccNotifiacion")) {

    String aid = request.getParameter("aid");
    String vid = request.getParameter("vid");
    String date = request.getParameter("date");
    String location = request.getParameter("location");
    String time = request.getParameter("time");

    String qry = "INSERT INTO `tbl_vacci_notification` (`vid`,`aid`,`date`,`location`,`time`)
VALUES('"+ vid + "','"+ aid + "','"+ date + "','"+ location + "','"+ time + "')";

    System.out.println(qry);

    if (con.putData(qry) > 0) {

        out.print("successful");

    } else {
        out.print("failed");
    }
}

//addVaccNotifiacion end //
Start UpdateNotiStatus
if (key.equals("UpdateNotiStatus")) {

    String uid = request.getParameter("uid");
    String vid = request.getParameter("vid");
    String qry = "INSERT INTO `tbl_notification_status` (notf_id,uid,user_type) VALUES ('"+ uid +
"', '"+ vid + "','user')";

```

```
        System.out.println(qry);

        if (con.putData(qry) > 0) {

            out.print("successful");

        } else {
            out.print("failed");
        }
    }

//end UpdateNotiStatus

//FetchVacinaions

    if (key.equals("FetchVacinaions")) {

        System.out.println("hhhhhhh");

        String uid = request.getParameter("uid");
        String list = "",aid="";

        String qry = "SELECT * FROM `tbl_user` WHERE uid=" + uid;
        System.out.println(qry);
        Iterator it1 = con.getData(qry).iterator();
        if (it1.hasNext()) {

            Vector v = (Vector) it1.next();

            aid=v.get(1).toString();

            String qry2="SELECT * FROM `tbl_vacci_notification` WHERE `aid`='"+aid+"'";
            System.out.println(qry2);

            Iterator it2 = con.getData(qry2).iterator();
            if (it2.hasNext()) {
                while (it2.hasNext()) {
                    list += "Date    : " + v.get(3) + "\nLnoction : " + v.get(4) + "\nTime : " + v.get(5) + "#";
                }

                System.out.print(list);
            }
            out.print(list);
        }
    }
```

```
    } else {  
        out.print("failed");  
    }  
}  
  
}  
  
%>
```

BIBLIOGRAPHY

BIBLIOGRAPHY

- [1] Pradeep Kothari, “Android Application Development”, 2007
- [2] Luke Welling, “Beginning Android Programming with Android Studio”, **2003**.
- [3] Rasmus Lerdorf, “Programming PHP”, 2008.
- [4] David Powers, “PHP Object Oriented Solutions,” *2008*.
- [5] [https:// www.w3layouts.com/results? search_query=java+tutorial](https://www.w3layouts.com/results?search_query=java+tutorial)
- [6] <https://www.developrs.android.com>
- [7] <https://github.com>

