Software Design Document

\

For

DAC APP

Version-1.0

Table of Contents

- **Introduction** **3**

Project 3

1. Problem Statement **3**

To Retrieve Digital Address Code(DAC) along with the building footprints on the Indian Maps. 3

Scope 3

Technologies Used 4

●.○.1 React Native (version 0.72.0) 4

●.○.2 Node.js (version v20.3.1) 4

●.○.3 JDK (version 8 or above) 5

●.○.4 Android Studio (version 2022.3.1) 5

●.○.5 PostgreSQL (version 13.4.2) 5

●.○.6 pgAdmin 5

●.○.7 PHP (version 8.0.9) 5

●.○.8 XAMPP Server (version 3.3.0) 6

●.○.9 phppgadmin (version 7.13.0) 6

- **Functional Requirement** **7**

Use Cases 7

Database Description 15

●.○.1 Table ‘users’ 15

●.○.2 Table ‘Register’ 15

●.○.3 Table ‘spatial_ref_sys’ 16

●.○.4 Table ‘jamalpur_dac’ 16

●.○.5 Table ‘Saved Polygon’ **17**

- **User Manual** **18**

Splash Screen 18

Sign In Screen 18

Register Screen 20

Main Screen 22

Drawer for Navigation 25

- **Definitions and Acronyms** **34**
- **References** **35**

● Introduction

Project

An app to get the Digital Address Code(DAC) of the current and marked location of the user on the Indian Map.

1.Problem Statement

To Retrieve Digital Address Code(DAC) along with the building footprints on the Indian Maps.

Scope

The DAC Finder app is designed to provide users with the Digital Address Code (DAC) of their current location and any marked location on the Indian map. Additionally, the app will also display building footprints on the map, making it a valuable tool, especially for postal workers in India. The scope of the app includes the following features and functionalities:

- Digital Address Code (DAC) Retrieval: The app will allow users to obtain the unique Digital Address Code (DAC) for any location in India. The DAC is a precise geocoding system that simplifies address identification, making it easier for postal services and other logistics operations.
- Current Location DAC: Users can quickly find the DAC of their current location by accessing their device's GPS functionality. This feature will enable individuals, including postmen, to determine the DAC for any point they are currently stationed.
- Mark Location and Retrieve DAC: Users can manually mark any location on the Indian map, and the app will provide the corresponding DAC for the marked point. This functionality will enable users to identify the DACs of various delivery destinations and efficiently plan their routes.
- Building Footprints: The app will display building footprints on the map, providing postal workers with a visual representation of the structures at specific locations. This

feature will help postmen identify buildings easily and deliver mail and packages more accurately.

- User Accounts: The app may incorporate optional user accounts to enable personalized features like saving favorite locations, history of marked locations, and syncing data across devices.
- User-Friendly Interface: The app will have an intuitive and user-friendly interface, making it accessible to a wide range of users, including those with minimal technical knowledge.
- Security and Privacy: The app will prioritize the security and privacy of user data. It will adhere to data protection regulations and ensure that sensitive location information is safeguarded.
- Cross-Platform Support: The app will be available for both Android and iOS devices, making it accessible to a broader user base.

Technologies Used

●.○.1 React Native (version 0.72.0)

React Native is a JavaScript framework used for developing a real, native mobile application for iOS and Android. It uses only JavaScript to build a mobile application. It is like React, which uses native components rather than using web components as building blocks. React Native apps are not web applications. They are running on a mobile device, and it does not load over the browser. React Native apps are the real native app, the JavaScript code stays as JavaScript, and they run in some extra thread by the compiled app. The user interface and everything is compiled to native code.

The entire DAC application is written in react native.

●.○.2 Node.js (version v20.3.1)

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. It is designed to build scalable network applications.
The app requires NPM for installing the dependencies.

●.○.3 JDK (version 8 or above)

JDK is required for developing apps using React Native CLI(version 2.0.1).

●.○.4 Android Studio (version 2022.3.1)

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, designed specifically for Android development.

It also provides an emulator (a virtual device) on which the app can be run, thus helping in debugging.

●.○.5 PostgreSQL (version 13.4.2)

PostgreSQL is a free and open-source relational database management system (RDBMS) emphasizing extensibility and SQL compliance.
This app uses PostgreSQL database to store user information, weather forecast & agricultural advisory based on location and date, user feedback for a bulletin, and the available locations.

●.○.6 pgAdmin

pgAdmin is a GUI for interacting with the PostgreSQL database.

●.○.7 PHP (version 8.0.9)

PHP is a general-purpose scripting language geared towards web development. PHP code is usually processed on a web server by a PHP interpreter implemented as a module, a daemon, or as a Common Gateway Interface (CGI) executable. On a web

server, the result of the interpreted and executed PHP code forms the whole or part of an HTTP response.

PHP is used for connecting the react-native frontend with the PostgreSQL database of the app.

●.○.8 XAMPP Server (version 3.3.0)

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible.

●.○.9 phppgadmin (version 7.13.0)

phpPgAdmin is a fully functional web-based administration utility for a PostgreSQL database server. It handles all the basic functionality as well as some advanced features such as triggers, views, and functions (stored procs). The administrator can fully interact with the database here.

● Functional Requirement

Use Cases

Use Case 0: REGISTER

★ Actors : User

★ Type : Primary and Essential

★ Pre-condition : None

★ Main Case Scenario :

- User enters personal information (name, mobile number, designation, department, address, state, pincode , etc) to create a new account.
- On submitting, the system creates a user account and stores the required information into the database.

● Exception Scenario :

- User inputs a mobile number containing less than 10 digits. And the pin code containing less than 6 digits. The user is requested to provide the correct information to create an account.
- User leaves either of the necessary fields - name, mobile number, state, pin code, or block - empty. The user is requested to provide all the details to create an account.
- With Ragex expression, the user can't enter the script which leads to app crash.
- The system is unable to connect to the database. 'Server Error' message is displayed, and the user is

requested to try again.

- By pressing the Register button, the registration is successful and then the user is directed to SignIn screen.

Use Case 1: LOG IN

★ Actors : User

★ Type : Primary and Essential

★ Pre-condition : User already has an account registered by their mobile number

★ Main Case Scenario :

- User enters the registered mobile number to log into the account.
- The system checks for an account with the mobile number provided in the database and fetches the account details.
- User successfully logs into their account.

● Exception Scenario :

- The mobile number entered contains less than 10 digits. The user is requested to enter the correct mobile number.
- No account exists with the mobile number entered. The user is requested to create a new account.
- The system is unable to connect to the database. 'Server Error' message is displayed, and the user is requested to try again.

Use Case 2: PROFILE

★ Actors : System

★ Type : Primary and Essential

★ Pre-condition : User has already logged into their account.

★ Main Case Scenario :

- System fetches user information from the database and displays it to the user.

● Exception Scenario :

- The system is unable to connect to the database. ‘Server Error’ message is displayed, and the user is requested to try again.

Use Case 3: Digital Address Code (DAC) Retrieval

★ Actors : Administrator, System

★ Type : Primary and Essential

★ Pre-condition : User has already logged into their account.

★ Main Case Scenario :

- User opens the DAC Finder app and logs into their account.
- The app determines the user's current location using the device's GPS functionality.
- The system fetches the Digital Address Code (DAC) for the user's current location from the database.
- The app displays the retrieved DAC on the map along with the building footprints of the area surrounding the user's location.
- The user can also choose to manually mark a different location on the map.
- If the user marks a different location, the system fetches the DAC for the marked location and displays it on the map with the corresponding building footprints.
- The system is unable to connect to the database. ‘Server Error’ message is displayed, and the user is

requested to reload (or try again).

- Exception Scenario :
  - If the app is unable to fetch the DAC from the database due to a server error or any other connectivity issues, an error message is displayed to the user, indicating the problem. The user is prompted to reload the app or try again later.
  - If the DAC for the co-ordinates is not available in the backend and database, then it returns NULL.

Use Case 4: Building Footprints

★ Actors : Administrator, System

★ Type : Primary and Essential

★ Pre-condition : User has already logged into their account.

★ Main Case Scenario :
- Based on the latitude and longitude of the marked and selected location on the Indian Map, it runs a query in the Postgres database and return the DAC along with Geom, having the vertices of the building footprints (closed polygon) to the frontend.
- If the selected location doesn't have the DAC saved in the backend it returns "NULL" and no building footprints for the respected marked co-ordinates on the Indian Map.
- It runs the query in the Backend in O(n) times complexity, making it quick and efficient to find the DAC in the large database and to be displayed on the app Screen.
- It places the building foot prints in the black outlined borders and blue as fill-up color inside the

boundary to provide contrast to the map Screen.

- Exception Scenario :
  - If the app is unable to fetch the DAC from the database due to a server error or any other connectivity issues, an error message is displayed to the user, indicating the problem. The user is prompted to reload the app or try again later.
  - The system is unable to connect to the database. 'Server Error' message is displayed, and the user is requested to reload (or try again).

Use Case 5: Drawing Building Footprints

★ Actors : User

★ Type : Primary and Essential

★ Pre-condition : User has already logged into their account.

★ Main Case Scenario :
- If the DAC returned is NULL of a particular marked location and no building footprints are retrieved from the backend, then it provides user the functionality to draw polygon around the marked building so the user can store the vertices of the edges of the drawn polygon in the separate database.
- It stores the vertices of the drawn polygon in the separate database in Postgres SQL for future references.

- Exception Scenario :
  - It provides the user with a maximum of 16 points/vertices of the polygon to mark on the

building footprints. If it excessed then it shows an alert and then restarts the drawing process again.

Use Case 6: USEFUL LINKS

- ★ Actors : System
- ★ Type : Primary and Essential
- ★ Pre-condition : User has already logged into their account.
- ★ Main Case Scenario :
  - The system displays some useful links related to Bhuvan map and NRSC official site in the About page of the DAC app.
- ● Exception Scenario : None

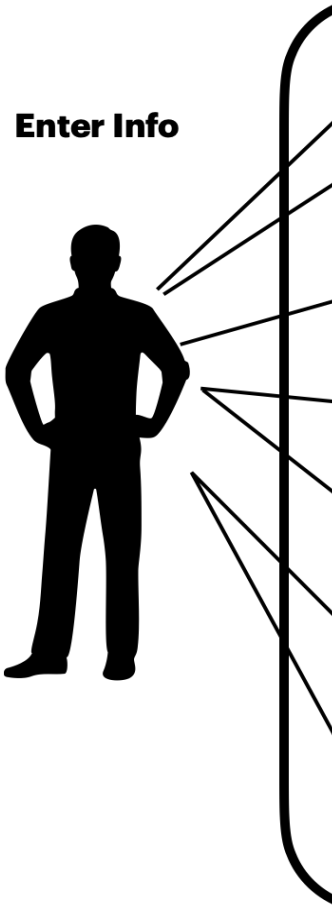
Use Case 7: LOG OUT

- ★ Actors : User
- ★ Type : Primary and Essential
- ★ Pre-condition : User has already logged into their account.
- ★ Main Case Scenario :
  - User logs out of their account.

- The system successfully returns to the welcome page.

- Exception Scenario : None

USE CASE DIAGRAM


Enter Info

Database Description

●.○.1 Table ‘users’

The table contains the information of the users registered in the app.

Field Name	Data Type	Description
id	Serial	Gives a serial number and is the primary key of the table.
user_name	VARCHAR(40)	Stores the name of the user.
user_phoneNumber	VARCHAR(10)	Stores the mobile number of the user.
mob_imei_no	VARCHAR(15)	Stores the IMEI number of the mobile of the user.
mob_model	VARCHAR(30)	Stores the model name of mobile.
mob_os	VARCHAR(20)	Stores the operating system name of the mobile.
os_version	VARCHAR(5)	Stores the version of the operating system of mobile.
registered_on	timestamp (without time zone)	Stores the registration time of the user.

●.○.2 Table ‘Register’

The table contains the information of different locations consists of – Name, Phone Number, Department, Designation, Address , State and PinCode.

Field Name	Data Type	Description

id	Serial	Gives a serial number and is the primary key of the table.
user_name	VARCHAR(40)	Stores name of the user.
user_phoneNumber	VARCHAR(10)	Stores the mobile number of the user.
designation	VARCHAR(30)	Stores designation of the user.
department	VARCHAR(50)	Stores department of the user.
address	VARCHAR(50)	Stores address of the user.
state	VARCHAR(30)	Stores name of the user's State.
district_code	Integer	Stores code of the district. This code is unique for each district of a particular state, but may repeat for districts of another state.

●.○.3 Table 'spatial_ref_sys'

The table contains the information of spatial reference system of India.

Field Name	**Data Type**	**Description**
srid	Integer	Gives a serial number and is the primary key of the table.
auth_name	Character Varying (256)	Stores location id of a block.
auth_srid	Integer	Stores the currently authorized srid of the location.
sr_text	Character Varying(2048)	Stores maximum srid of the location.

●.○.4 Table 'jamalpur_dac'

The table contains the information of dac at Jamalpur,Patna,India.

Field Name	**Data Type**	**Description**
id	Serial	Gives a serial number and is the primary key of the table.

dac	VARCHAR	Stores the location DAC of the location corresponding to the latitude and longitude.
dac_id	VARCHAR	Stores the dac_id with respect to the to the respective dac of the corresponding latitude and longitude.
geom	VARCHAR(1000)	Stores the contour/boundary point latitude and longitude of the physically mapped contours.

●.○.5 Table 'Saved Polygon'

Field Name	Data Type	Description
id	Serial	It gives a serial number and is the primary key of the table.
latitude	VARCHAR	Stores the latitude of the respective selected and marked location on the Indian map.
longitude	VARCHAR	Stores the longitude of the respective selected and marked location on the Indian map.
Polygon Number	VARCHAR	When the user finished marking the polygon on the Indian map manually. It assigns the polygon number to all the unfilled previous marker until the already marked polygon.

● User Manual

Splash Screen

Sign In Screen

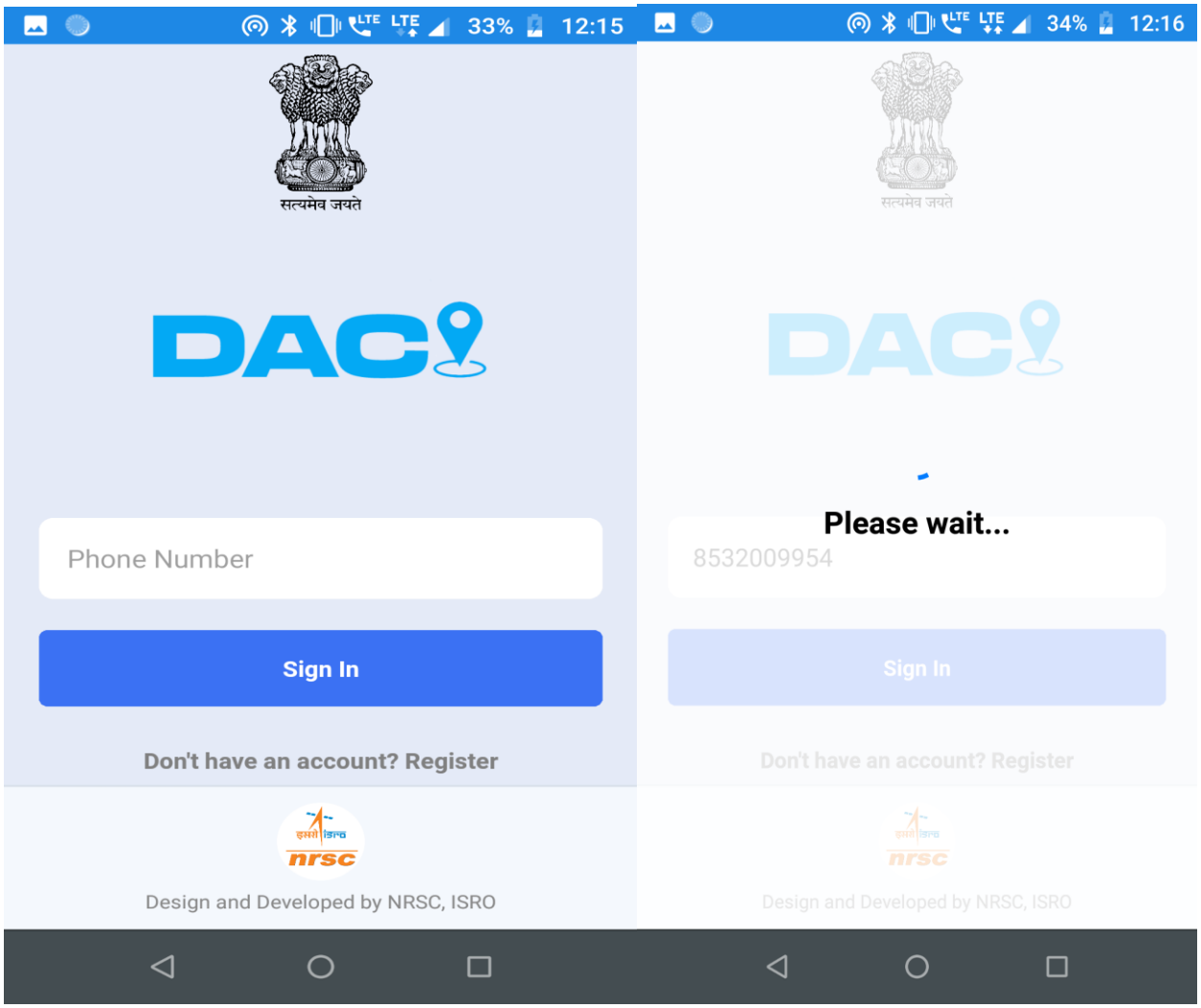
The welcome screen will vanish when the user presses the “Get Started” button and the Sign In Screen will appear respectively.

The user should enter their registered mobile number (containing 10 digits) to successfully log into their account.

If the mobile number is not registered, a warning message will be prompted and the user may register from the link given below the log in button.

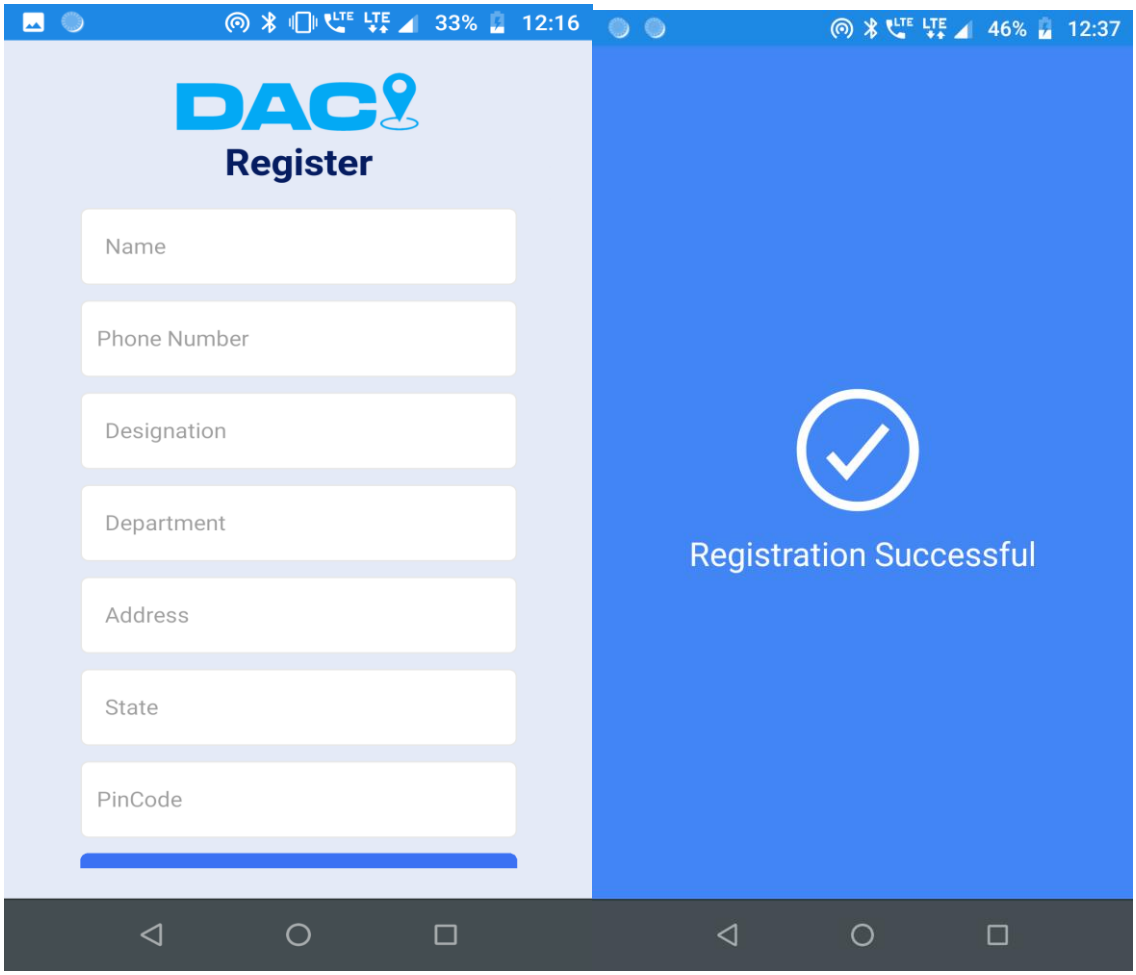
The screen contains two buttons:

1. Register (For new users) - New users may create their account
2. Log In (For registered users) – Existing users can log into their account
3. At the bottom of the screen, links to Terms and Conditions and Privacy Policy are given.



Register Screen

This contains the user registration form. New users may fill in their details – name, phone number, department, designation, address, state, district Pincode to create their account.



Please check the form again before submitting, as nothing can be done afterwards.

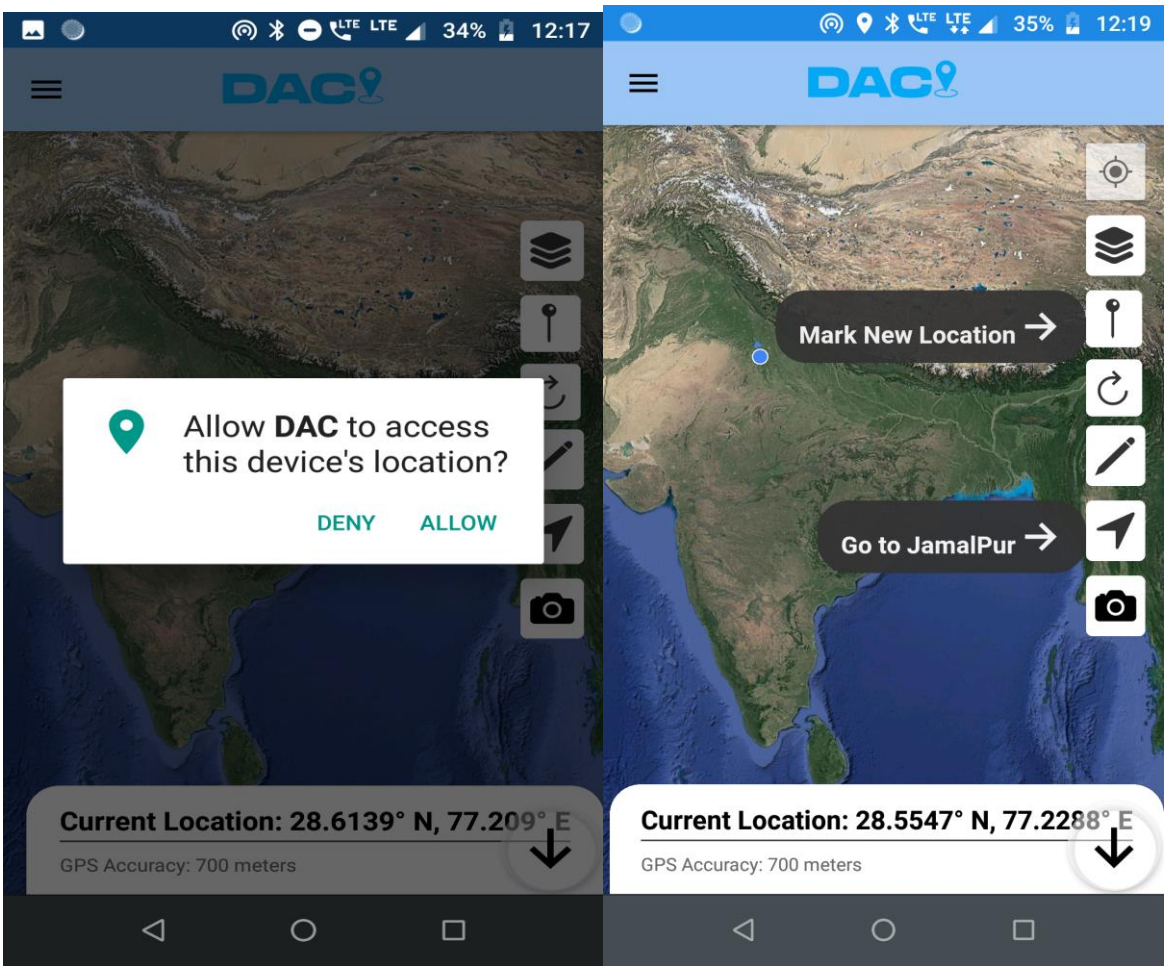
However, if a user enters incorrect data (like mobile number having less than 10 digits) or any of the fields is left blank, the same will be notified and registration will be aborted.

Successful registration will display the message “Successfully registered” and user will be logged into their account.

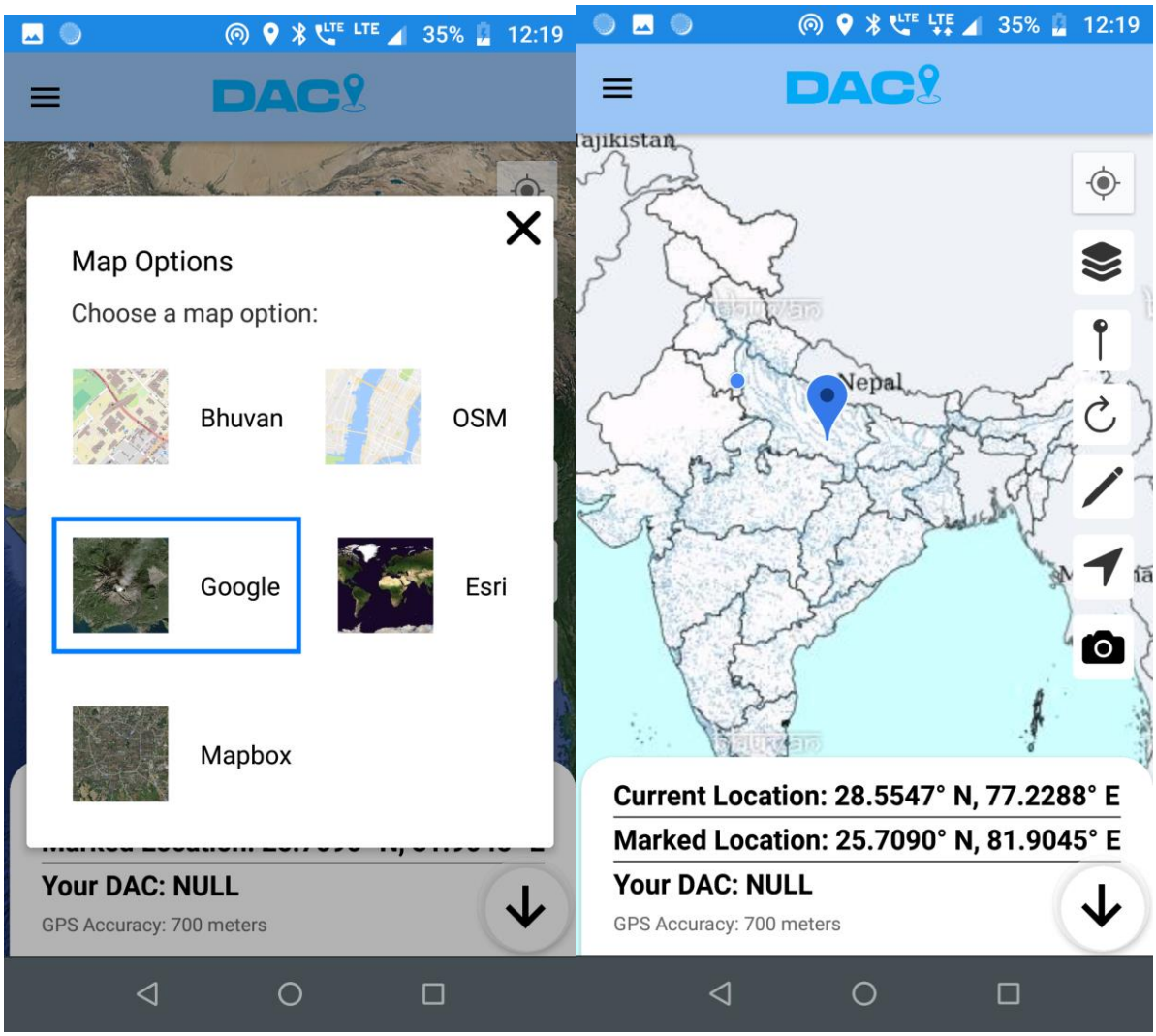
If user has an existing account and they try to register, a warning message will be displayed and registration will be aborted. They may go to the Log In page from the link at the bottom of the screen.

Main Screen

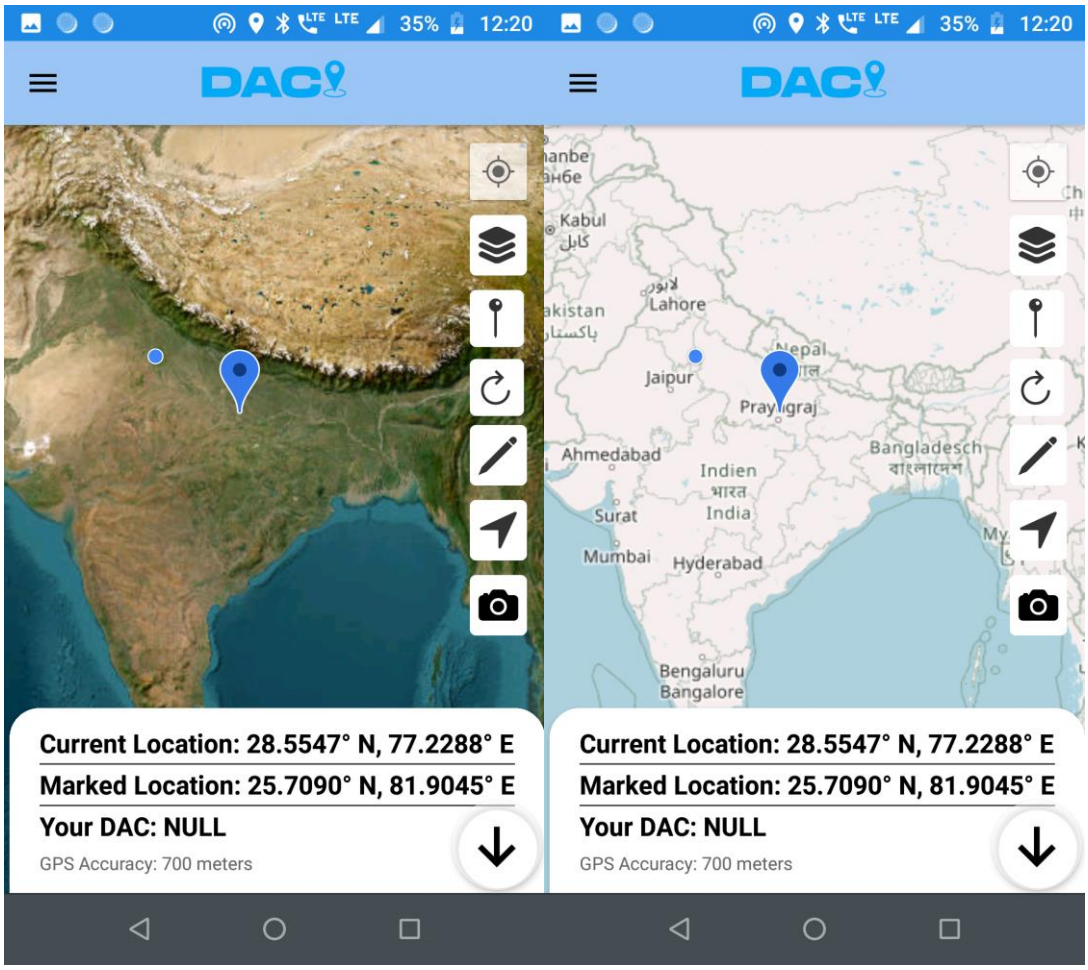
After successful registration or log in, the user will be directed to the main screen. Main screen asks for user permission to access location via GPS. It displays the current location of the user at the footer tab.



It offers User to select between five multiple Map types to be displayed on the map before or after marking the location on the Indian Map.



Bhuvan

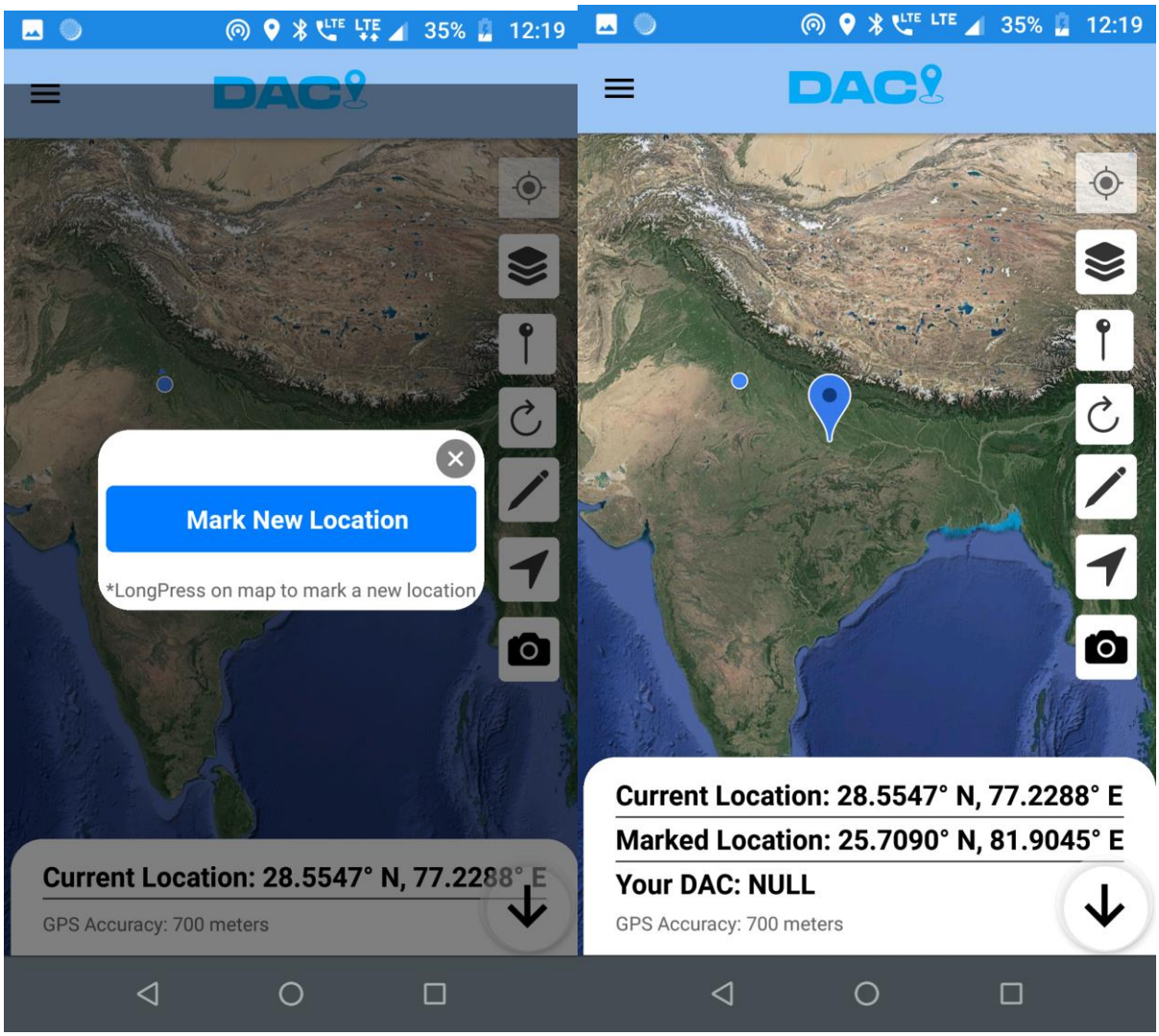


ESRI Satellite Map OSM

The application provides users with the capability to designate a specific point on the map by employing a prolonged press gesture. This functionality enables users to select or mark a location of interest on the map by holding down the touch input.

Additionally, users have the flexibility to update the selected location by performing a prolonged press on a different area of the map. Consequently, the revised current location will be promptly displayed

in the footer, positioned below the user's active and initial current location.

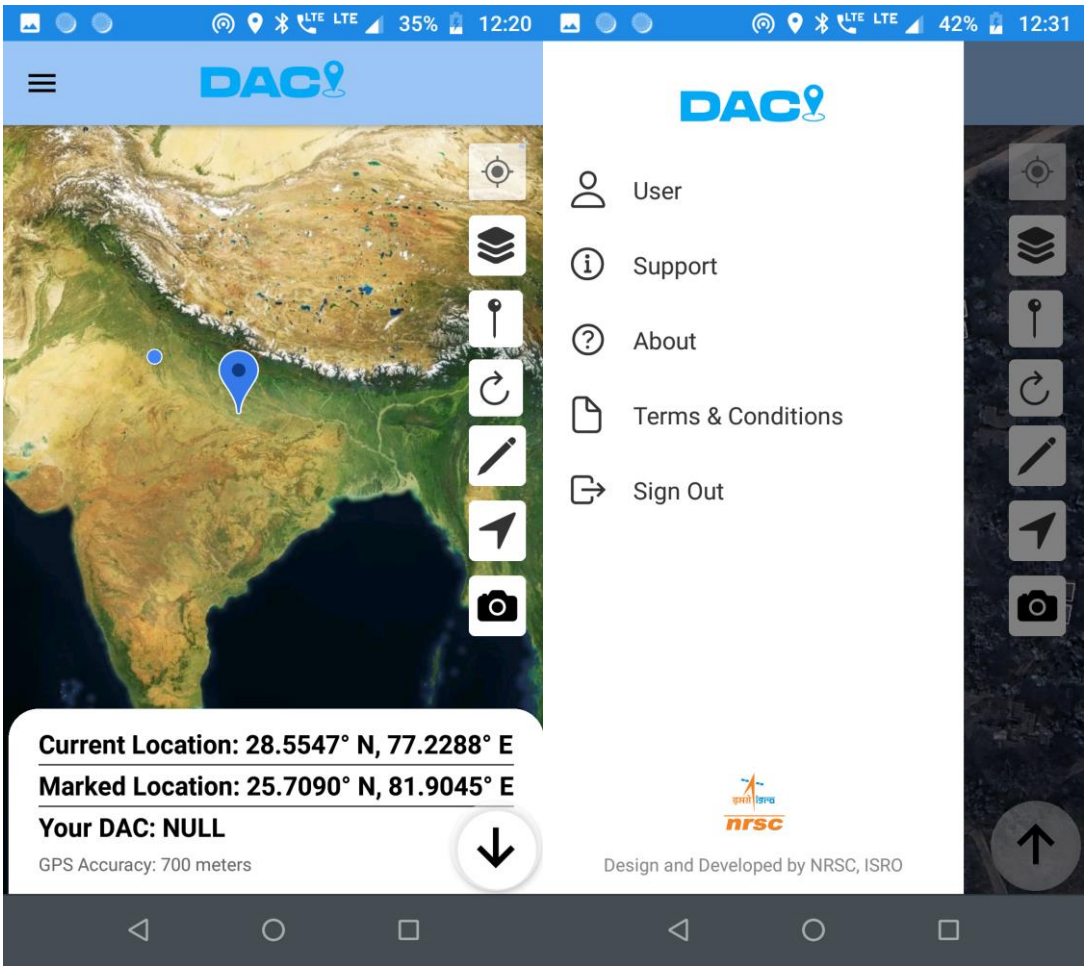


Drawer for Navigation

Tabs are present at the bottom of the screen for navigating between the four screens:

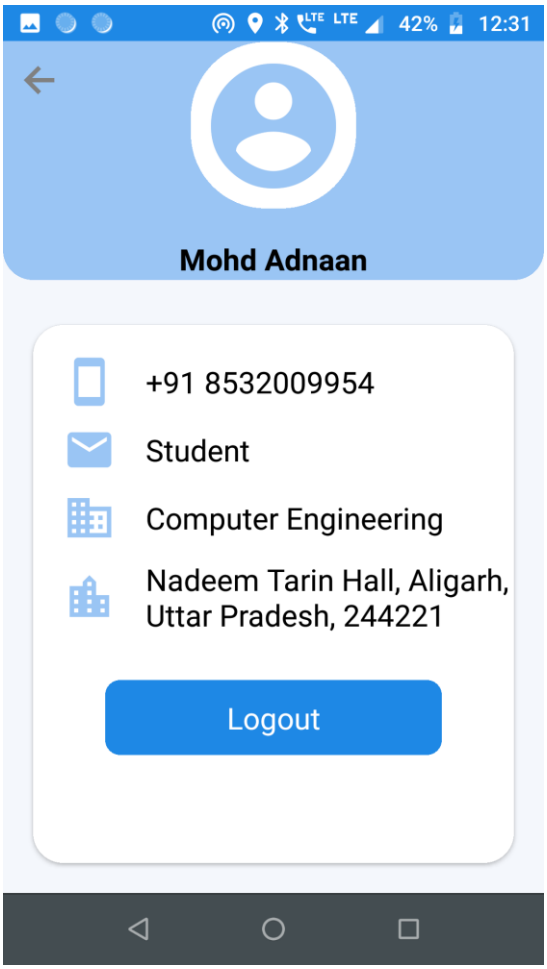
- Main Screen

  This is the main screen after the successful login/registration.



- User Screen

  This can be accessed from the first tab of the drawer. It contains information about the user.



- Support Screen

  This can be accessed from the second tab of the drawer. It contains the support page.

- About the DAC Application

  This can be accessed from the third tab of the drawer. It contains useful information about the DAC application.

- Save and Share

Main Functionality

The DAC Finder app is designed to provide users with the Digital Address Code (DAC) of their current location and any marked location on the Indian map. Additionally, the app will also display building footprints on the map, making it a valuable tool, especially for postal workers in India. The scope of the app includes the following features and functionalities:

→ ***Digital Address Code (DAC) Retrieval:*** Geolocation Identification Service: The application will grant users access to retrieve the exclusive Digital Address Code (DAC) for any location within India. The DAC serves as a highly accurate geocoding system, streamlining address identification and enhancing efficiency for postal services and logistics operations.

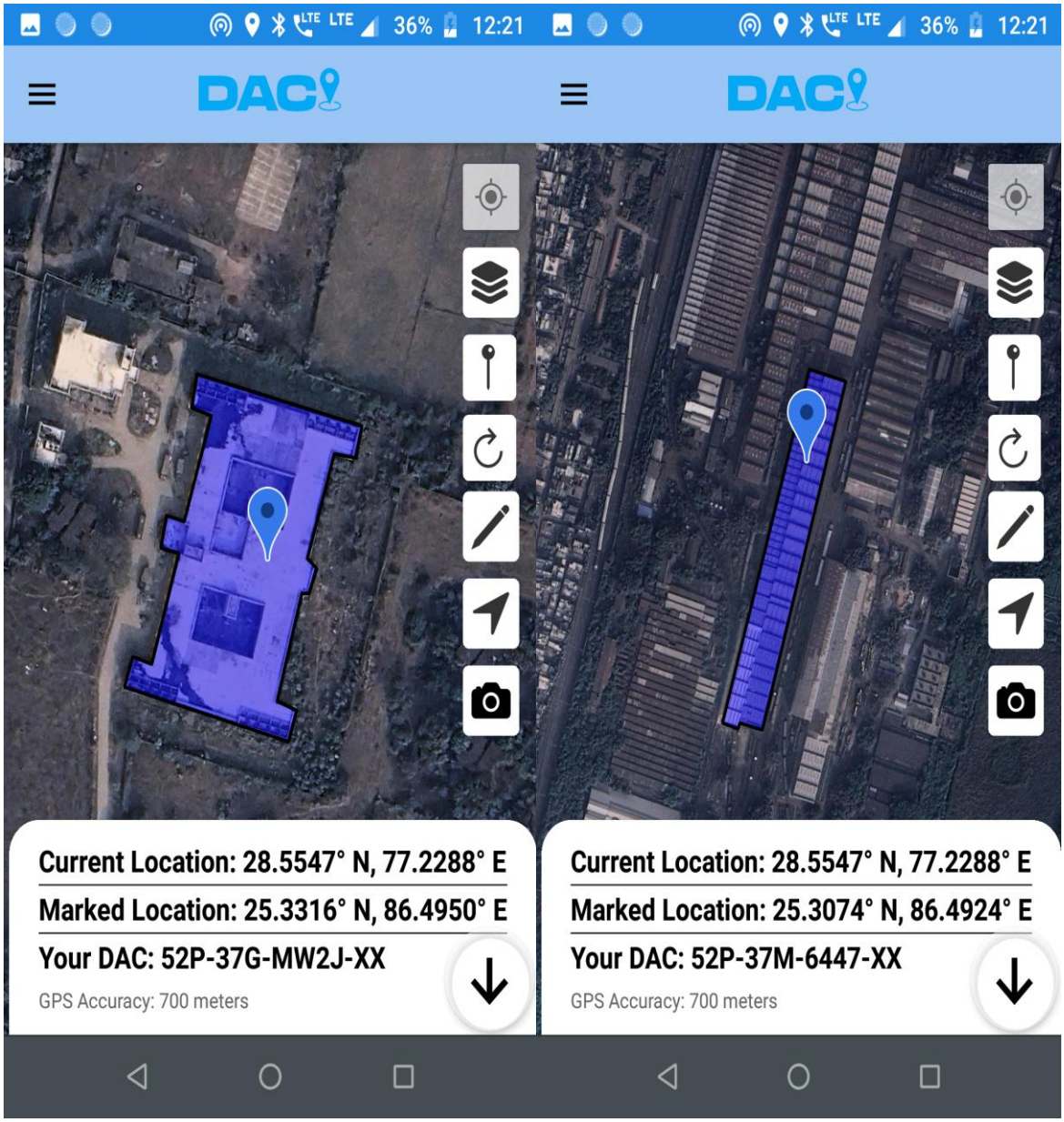
→ ***Mark Location and Retrieve DAC:*** Users can manually mark any location on the Indian map, and the app will provide the corresponding DAC for the marked point. This functionality will enable users to identify the DACs of various delivery destinations and efficiently plan their routes.

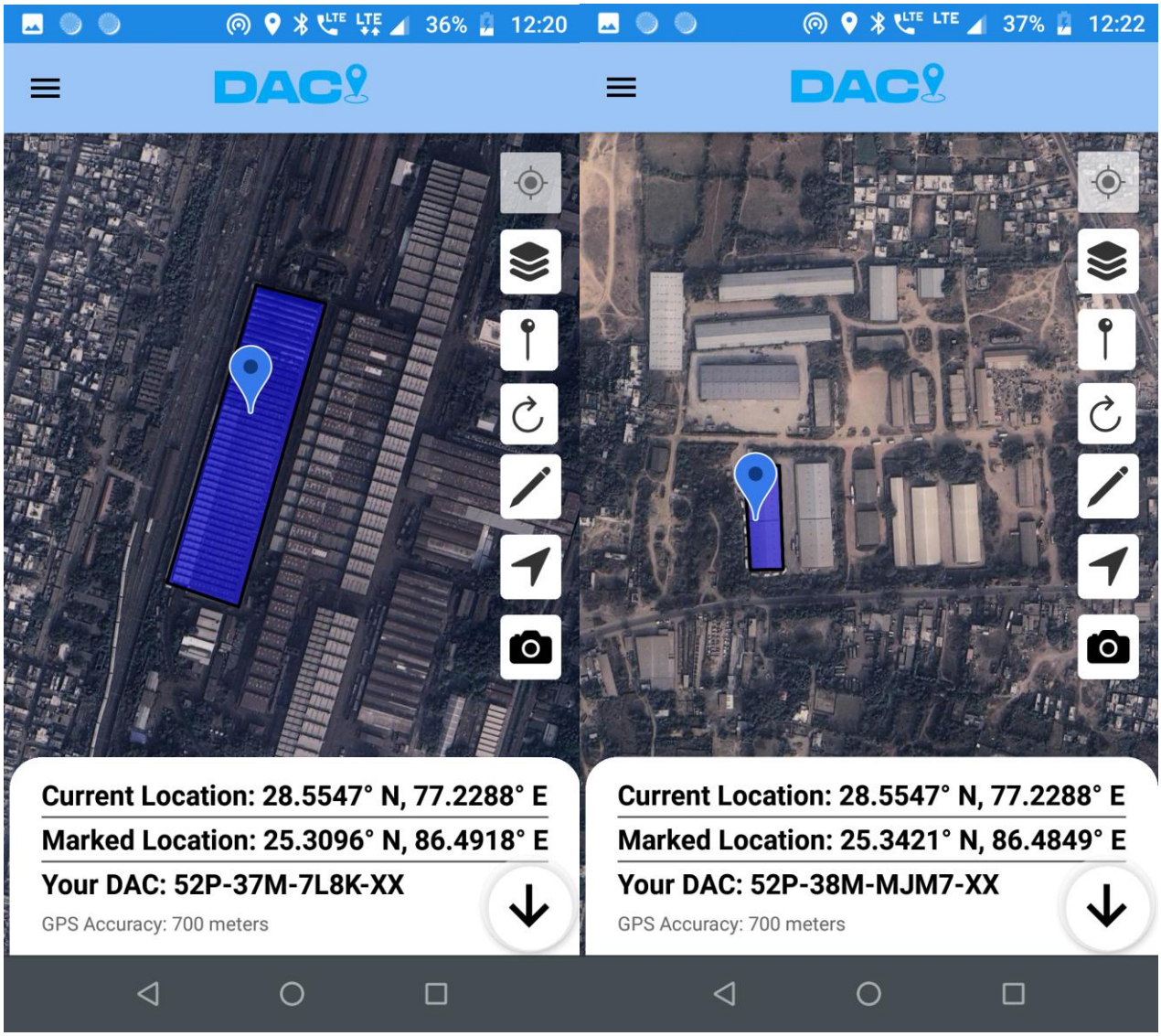
→ ***Building Footprints:*** The app will display building footprints on the map, providing postal workers with a visual representation of the structures at specific locations. This feature will help postmen identify buildings easily and deliver mail and packages more accurately.

→ ***Drawing Contour/Boundary on Building:*** The application offers a feature to draw contours or boundaries on buildings or coordinates for which the Digital Address Code (DAC) information is not available. This functionality empowers users to manually delineate contours on such buildings or coordinates using interactive tools. Once the contours are drawn, the user can proceed to store or

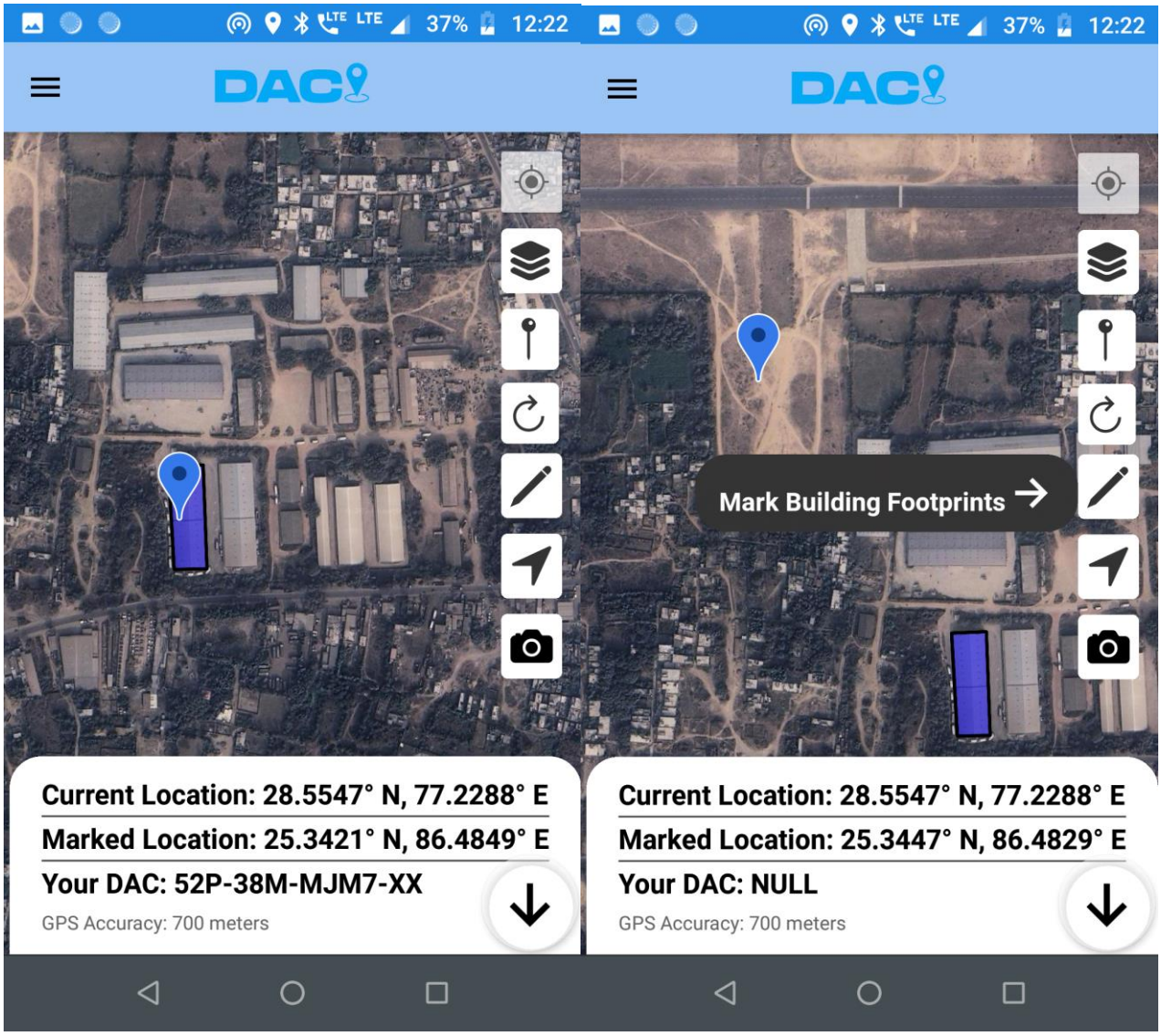
update this custom-generated data in the backend, ensuring that the information is accurately recorded and accessible for future reference and analysis.

Digital Address Code (DAC) Retrieval: The app will allow users to obtain the unique Digital Address Code (DAC) for any location in India. The DAC is a precise geocoding system that simplifies address identification, making it easier for postal services and other logistics operations.

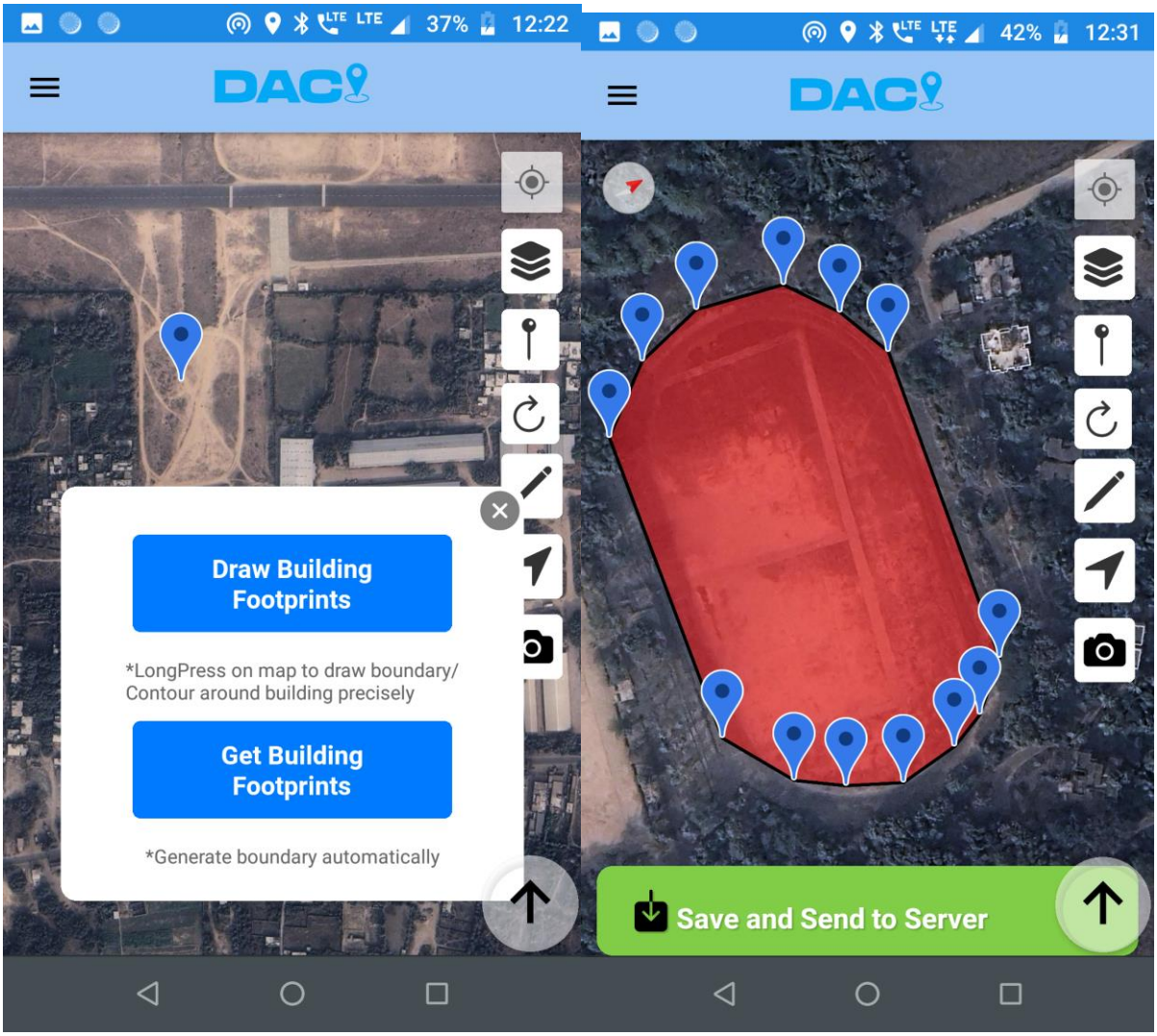


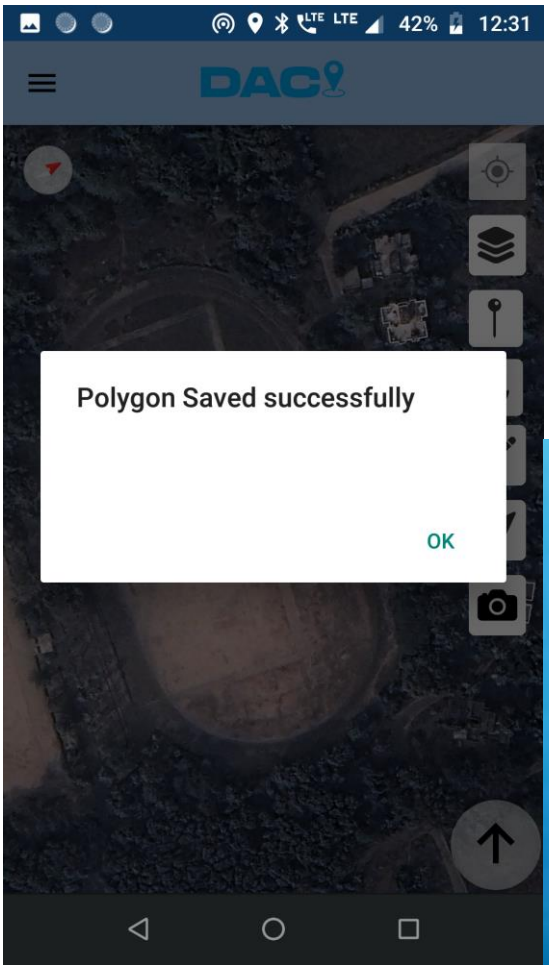


Geographical Point Marking and DAC Retrieval: Within the application, users possess the capability to manually designate any location on the Indian map, and subsequently, the app will furnish the corresponding Digital Address Code (DAC) for the marked point. This functionality empowers users to efficiently identify the DACs associated with diverse delivery destinations, facilitating optimal route planning and logistics operations.



The application allows users to draw contours on buildings or coordinates lacking **Digital Address Code (DAC)** data, empowering manual delineation using interactive tools. After drawing, users can store or update the custom-generated data in the backend for accurate recording, future reference, and analysis.


37% 12:22
DAC
Draw Building Footprints
*LongPress on map to draw boundary/ Contour around building precisely
Get Building Footprints
*Generate boundary automatically
42% 12:31
DAC
Save and Send to Server



● Definitions and Acronyms

Abbreviation	Description
SDD	Software Design Document
DFD	Data Flow Diagram
DAC	Digital Address Code
NPM	Node Package Manager
JDK	Java Development Kit
CLI	Command Line Interface
SQL	Structured Query Language
PHP	PHP Hypertext Preprocessor
HTTP	Hypertext Transfer Protocol
XAMPP	XAMPP Apache + MariaDB + PHP + Perl
IDE	Integrated Development Environment
CGI	Common Gateway Interface
RDBMS	Relational Database Management System
GUI	Graphical User Interface

OSM	Open street Map
NA	Not Applicable

• References

[1] React Native official website

[2] Node.js official webpage

[3] JDK official webpage

[4] Android Studio official webpage

[5] PostgreSQL official webpage

[6] PHP official webpage

[7] XAMPP official webpage

[8] phppgAdmin github repo