REVA UNIVERSITY
Bengaluru, India

SCHOOL OF ELECTRONICS AND COMMUNICATION ENGINEERING

A MINI PROJECT REPORT
ON

" FOOD WASTE MANAGEMENT USING MOBILE APPLICATION"

Submitted in fulfillment of the requirements for the award of the Degree of

BACHELOR OF TECHNOLOGY
IN
ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

VARSHA C S	(R18EC334)
VARSHINI V	(R18EC335)
VASI HEMANTH	(R18EC337)
JUPALLI BALAJI	(R18EC121)

Under the guidance of
Prof. NITYA S
Asst. Prof., School of ECE
REVA University

May 2021
Rukmini Knowledge Park, Kattigenahalli, Yelahanka, Bengaluru-560064
www.reva.edu.in

DECLARATION

I, Mr. / Ms. **VARSHA C S**(R18EC334) student of B.Tech, belongs to School of Electronics and Communication Engineering, REVA University, declare that this Mini Project Report / Dissertation entitled "**FOOD WASTE MANAGEMENT USING MOBILE APPLICATION**" is the result the of project / dissertation work done by me under the supervision of Prof. NITYA S, Asst. Prof., School of ECE REVA University, Bengaluru.

I am submitting this Project Report / Dissertation in partial fulfillment of the requirements for the award of the completion in 6th semester of Bachelor of Technology in Electronics and Communication Engineering by the REVA University, Bengaluru during the academic year 2020-2021.

I further declare that this project / dissertation report or any part of it has not been submitted for award of any other Degree / Diploma of this University or any other University/ Institution.

Varsha C S
(Signature of the Student)

Signed by me on 05th May 2021.

Certified that this project work submitted by Ms. Varsha C S (R18EC334) has been carried out under my / our guidance and the declaration made by the candidate is true to the best of my knowledge.

Signature of Guide
Date: 05thMay2021

Signature of Director
Date: 05th May2021
Official Seal of the School

REVA UNIVERSITY
Bengaluru, India

SCHOOL OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

Certified that the mini project work entitled " **FOOD WASTE MANAGEMENT USING MOBILE APPLICATION**" carried out under my / our guidance by ***Ms. Varsha C S (R18EC334)*** a bonafide student of REVA University during the academic year 2020-2021, is submitting the mini project report in partial fulfillment for the completion in 6Th semester of Bachelor **of Technology in Electronics and Communication Engineering** during the academic year **2020-2021**. The mini project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Signature with date	**Signature with date**
Prof. NITYA S	**Dr. R.C. Biradar**
Guide	**Director**

Name of the Examiner with affiliation	**Signature with Date**
1.	
2.	

ACKNOWLEDGEMENTS

We like to share our sincere gratitude to all those who help us in completion of this project. During the work we faced many challenges due to our lack of knowledge and experience but these people help us to get over from all the difficulties and in final compilation of our idea to a shaped sculpture.

We would like to thank Prof Nitya S mam for his governance and guidance, because of which our whole team was able to learn the minute aspects of a project work.

We would also like to show our gratitude to our Project Coordinators Prof Dilip Chandra E and Prof Chaitra Nayak J Ankit for their continuous help and monitoring during the project work.

In the last we would like to thank the management of School of ECE of Reva University for providing us such an opportunity to learn from these experiences.

Thank you All

Varsha C S (R18EC334)
Varshini V (R18EC335)
Vasi Hemanth (R18EC337)
Jupalli Balaji (R18EC121)

TABLE OF CONTENTS

This page is intentionally left blank

LIST OF FIGURES

This page is intentionally left blank

ABSTRACT

Majority of people stop thinking about it, when something is thrown away. Yet, food is a common item which is wasted daily. It is a growing evidence that a significant share of global food is thrown away every single day. Wasting food has started happening decades ago, the contribution to stop and manage this problem is in everyone's hands. Reducing Food waste is a key sustainability challenge for every one globally .Despite of the issue, the link between innovation practices and food waste management has grabbed a little attention in the past few years. With the advancement in technology, the amount of food wasted globally can be reduced to a huge percent. Additionally hunger is also a major problem that is related to food waste. This paper explains about a vast range of food and hunger management initiatives, methods to solve this problem globally.

Chapter 1

INTRODUCTION

1.1Main Section

Mobile phone applications have seen wide use in recent years. It's known that Android is the most popular platform for mobile, right now android is used on over 190 countries in the world on millions of mobile devices. Android is the most installed platform for mobile, and the number is increasing rapidly since almost 1 million users every day purchase new Android devices and use it immediately to get digital content such as games, application, and many other services. Due to that, we developed an android application *“Food Waste Management App”* using React Native which can also be used to build for ios platform too. The purpose of this development is to limit the wastage of food in the entire world. Many restaurants tend to throw the leftover food at the end of the day even though the food is perfectly fine to be eaten, which means that huge amounts of food are wasted. While all that food is being wasted, some families can barely afford proper meals with their limited money. They don't get enough nutrition due to the lack of having three meals in a day. Therefore, we decided to create our application to link the Vendors (people who are willing to donate food) with the clients (people who are willing to take food), so instead of throwing the food, the clients will be able to pick it up from the vendors at the end of the day. The application allows the both Vendors and clients to log in, and upload an image of the meals they have as leftovers along with a description of that meal, and the location where to pick it up. The users then, can log in and choose the meal suitable to their choice and can pick it up once they send a request to the application.

Chapter 2

LITERATURE SURVEY

[1] Food talks back: Exploring the role of mobile applications in reducing domestic food wastage Conference: In Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: the Future of Design At: University of Technology, Sydney, Australia Authors: *Geremy Farr-Wharton 5.6The Commonwealth Scientific and Industrial Research OrganisationJaz Hee-Jeong ChoiMarcus Fo*

[2] A Methodology for Sustainable Management of Food Waste Guillermo Garcia-Garcia,Elliot Woolley, Shahin Rahimifard, James Colwill, Rod White & Louise Needham Waste and Biomass Valorization volume 8, pages2209–2227(2017) Open Access Publication date: 25 November 2016.

[3] Waste Management in Restaurants May 2014 Authority:

[4] Food Waste ManagementSolving the Wicked Problem Editors: Närvänen, E., Mesiranta, N., Mattila, M., Heikkinen, A. (Eds.).

[5] Food waste management innovations in the foodservice industry September 2018 *Waste Management*79:196 DOI:*10.1016/j.wasman.2018.07.033*Authors:*Carlos Martin-Rios 22.79Christine Demen-Meier 4.49*.

[6] Other *reference* Projects *Sustainable Innovations in Food Service*, *Sustainable Innovations in Food Service*.

[7] *Android the world's most popular mobile platform |Android Developers*, 2017, [online] Available: https://developer. android.com/about/index.html.

Show Context *Google Scholar*

Chapter 3

PROPOSED WORK

With the advancement in technology, evolution of the human race has become quite simpler and easier. Smart phones were a remarkable achievement, which helps billions of people nowadays to manage everything by using them. With the invention of smart phones communication has established in all forms, eventually they became a daily necessity for everyone.

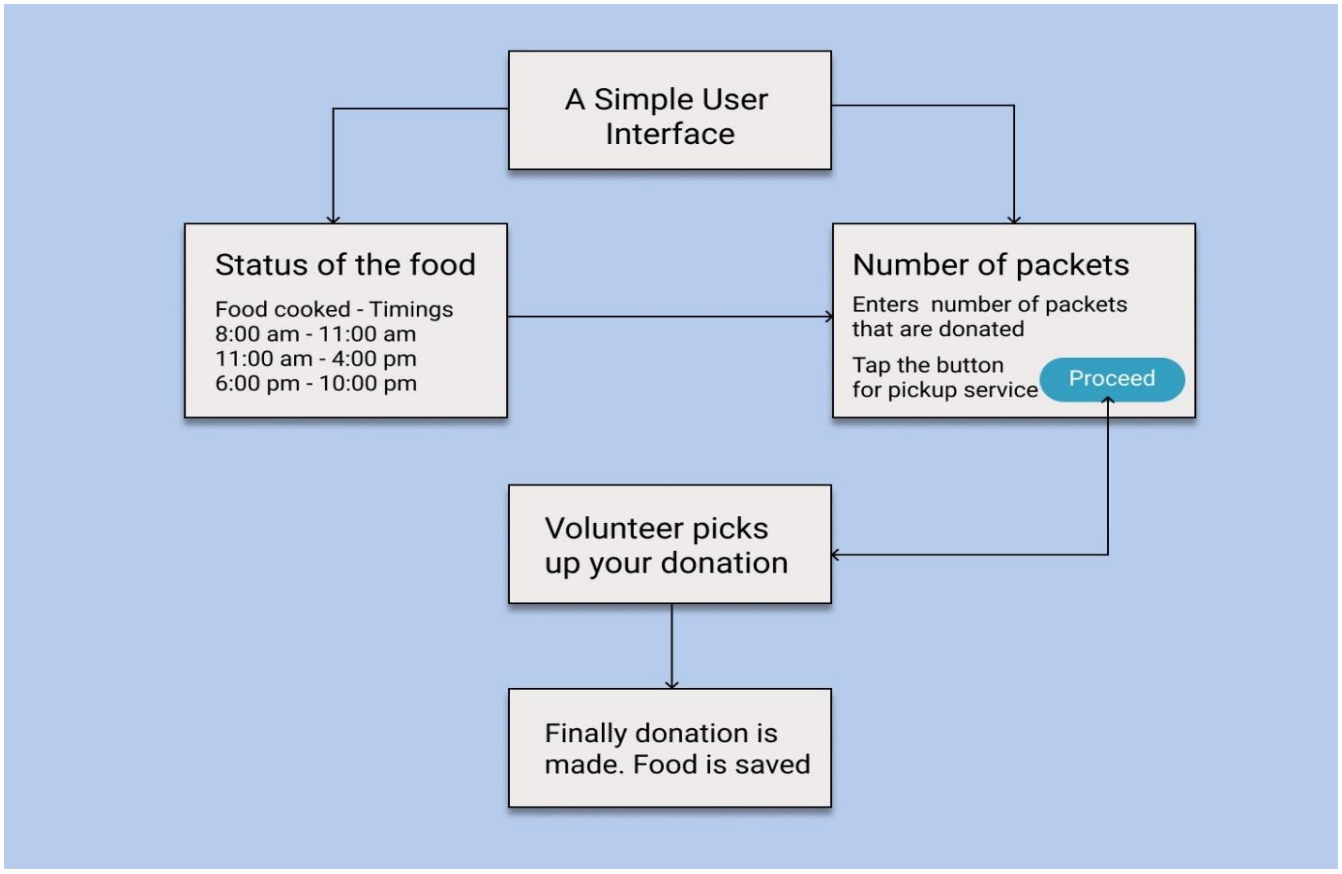

A Simple User Interface
Status of the food
Food cooked - Timings
8:00 am - 11:00 am
11:00 am - 4:00 pm
6:00 pm - 10:00 pm
Number of packets
Enters number of packets that are donated
Tap the button for pickup service
Proceed
Volunteer picks up your donation
Finally donation is made. Food is saved

Figure 1: Food Waste Management using mobile Application (overview)

Smart phones also provide us with real time application data like date, time and location which help us to act accordingly. The growth in tech will help us further in every day to day life to solve problems. Nowadays it has become very easy to find the location of any smart phone user through GPS (*Global Positioning System*). With this lead, solving many big problems with a simple smart phone application becomes the easiest way. The major problem in the current world's to manage food that is leftover and food that is not properly used in time which can be governed with our actions. Developing a smart phone application to reduce food waste helps many people to find food and reduce the waste percentage of food. A simple smart phone app with many applications is designed which helps users (*Client*) to pick up food from other users (*Vendors*) who are willing to donate food rather than wasting food. Our proposed method uses end to end encryption which can be accessed authentically, that helps the users to share food. This mobile application is built with React Native, Firebase SDK (*software development kit*) and requires Api and services (location service, monitoring service, analytic service) that render data to process the application according to the user's (*Vendors and Clients*) needs.

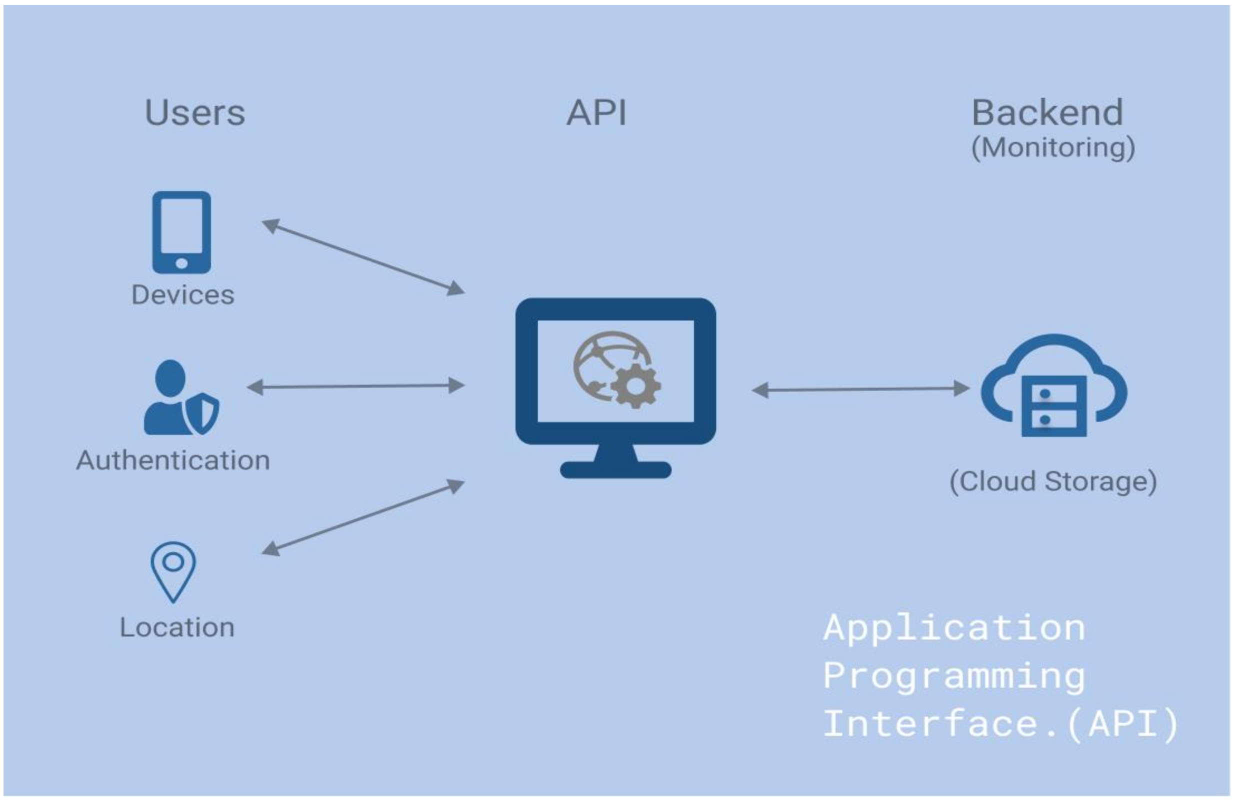


Figure 2: API (Application Programming Interface)

An **API** is a set of programming code that enables data transmission between one software product and another. It also contains the terms of this data exchange.
Application programming interfaces consist of two components:

- Technical specification describing the data exchange options between solutions with the specification done in the form of a request for processing and data delivery protocols.
- Software interface written to the specification that represents it.

The software that needs to access information (i.e., X hotel room rates for certain dates) or functionality (i.e., a route from point A to point B on a map based on a user's location) from another software, calls its API while specifying the requirements of how data/functionality must be provided. The other software returns data/functionality requested by the former application

.And the interface by which these two applications communicate is what the API specifies.

3.2 Location or Geo-location API : The Geo-location API which is used in our app to return a location and accuracy radius based on information about cell towers and Wi-Fi nodes that the mobile client can detect. This document describes the protocol used to send this data to the server and to return a response to the client.

Communication is done over HTTPS using POST. Both request and response are formatted as JSON, and the content type of both is application/json.

3.3 SOFTWARE DESCRIPTION

3.3.1 React Native

React Native is a JavaScript framework for writing real, natively rendering mobile applications for iOS and Android. It's based on React, Facebook's JavaScript library for building user interfaces, but instead of targeting the browser, it targets mobile platforms. In other words: web developers can now write mobile applications that look and feel truly "native," all from the comfort of a JavaScript library that we already know and love. Plus, because most of the code you write can be shared between platforms, React Native makes it easy to simultaneously develop for both Android and iOS.

Similar to React for the Web, React Native applications are written using a mixture of JavaScript and XML-esque markup, known as JSX. Then, under the hood, the React Native "bridge" invokes the native rendering APIs in Objective-C (for iOS) or Java (for Android). Thus, your application will render using real mobile UI components, *not* webviews, and will look and feel like any other mobile application. React Native also exposes JavaScript interfaces for platform APIs, so your React Native apps can access platform features like the phone camera, or the user's location.

React Native currently supports both iOS and Android, and has the potential to expand to future platforms as well. The vast majority of the code we write will be cross-platform. And yes: you can really use React Native to build production-ready mobile applications! Some anecdota: Facebook, Palantir, and TaskRabbit are al-ready using it in production for user-facing applications.

The fact that React Native actually renders using its host platform's standard rendering APIs enables it to stand out from most existing methods of cross-platform application development, like Cordova or Ionic. Ex-isting methods of writing mobile applications using combinations of JavaScript, HTML, and CSS typically render using webviews. While this approach can work, it also comes with drawbacks, especially around per-formance. Additionally, they do not usually have access to the host platform's set of native UI elements. When these frameworks do try to mimic native UI elements, the results usually "feel" just a little off; reverse-engineering all the fine details of things like animations takes an enormous amount of effort, and they can quickly become out of date.

In contrast, React Native actually translates your markup to real, native UI elements, leveraging existing means of rendering views on whatever platform you are working with. Additionally, React works separately from the main UI thread, so your application can maintain high performance without sacrificing capability. The update cycle in React Native is the same as in React: when props or state change, React Native re-renders the views. The major difference between React Native and React in the browser is that React Native does this by leveraging the UI libraries of its host platform, rather than using HTML and CSS markup. This means you can write mobile apps with the performance and look and feel of a native application, while using familiar tools. React Native also represents an improvement over normal mobile development in two other areas: the developer experience and cross-platform development potential.

Create native apps for Android and iOS using React

- React Native combines the best parts of native development with React, a best-in-class JavaScript library for building user interfaces.
- Using React Native today in your existing Android and iOS projects or you can create a whole new app from scratch.

Written in JavaScript—rendered with native code

- React primitives render to native platform UI, meaning your app uses the same native platform APIs other apps do.
- Many platforms, one React. Create platform-specific versions of components so a single codebase can share code across platforms. With React Native, one team can maintain two platforms and share a common technology—React.

Native Development For Everyone

- React Native lets you create truly native apps and doesn't compromise your users' experiences. It provides a core set of platform agnostic native components like View, Text, and Image that map directly to the platform’s native UI building blocks.

Seamless Cross-Platform

- React components wrap existing native code and interact with native APIs via React’s declarative UI paradigm and JavaScript. This enables native app development for whole new teams of developers, and can let existing native teams work much faster.

Fast Refresh

- See your changes as soon as you save. With the power of JavaScript, React Native lets you iterate at lightning speed. No more waiting for native builds to finish. Save, see, repeat.

IMPLEMENTAION

The working principles of React Native are virtually identical to React except that React Native does not manipulate the DOM via the Virtual DOM. It runs in a background process (which interprets the JavaScript written by the developers) directly on the end-device and communicates with the native platform via serialized data over an asynchronous and batched bridge.

React components wrap existing native code and interact with native APIs via React's declarative UI paradigm and JavaScript. This enables native app development for whole new teams of developers, and can let existing native teams work much faster.[18]

While React Native styling has a similar syntax to CSS, it does not use HTML or CSS.[19] Instead, messages from the JavaScript thread are used to manipulate native views. React Native also allows developers to write native code in languages such as Java or Kotlin for Android and Objective-C or Swift for iOS, which makes it even more flexible.

Setting up the development environment

The easiest way to get started is with Expo CLI. Expo is a set of tools built around React Native and, while it has many features, the most relevant feature for us right now is that it can get you writing a React Native app within minutes. You will only need a recent version of Node.js and a phone or emulator. If you'd like to try out React Native directly in your web browser before installing any tools, you can try out Snack.

Prerequisites

- Node js version (Node 12 LTS or greater installed)
- npm to install the Expo CLI

To install expo-cli for a quickstart

```
npm install -g expo-cli
```

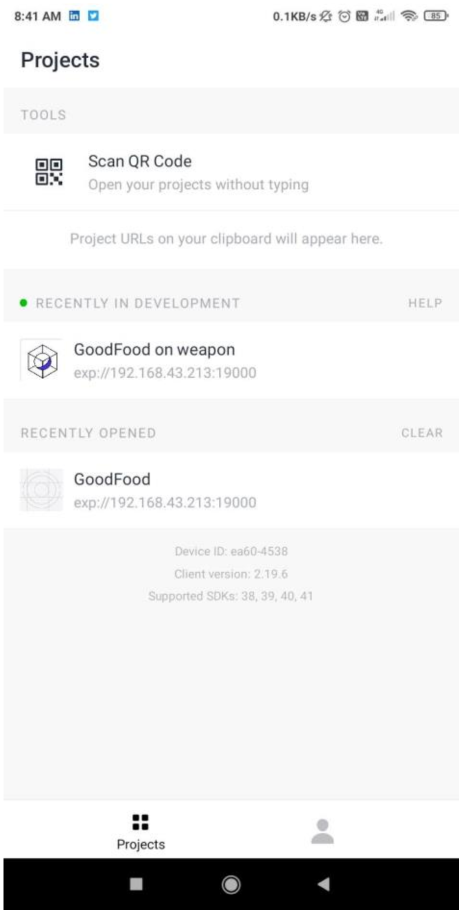
To initialize a new React native expo project and start it's development server

```
expo init AwesomeProject
```

```
cd AwesomeProject
npm start # you can also use: expo start
```

Running React Native application

Install the Expo client app on your iOS or Android phone and connect to the same wireless network as your computer. On Android, use the Expo app to scan the QR code from your terminal or metro bundler to open your project. On iOS, use the built-in QR code scanner of the Camera app.



Projects

TOOLS

Scan QR Code
Open your projects without typing

Project URLs on your clipboard will appear here.

RECENTLY IN DEVELOPMENT HELP

GoodFood on weapon
exp://192.168.43.213:19000

RECENTLY OPENED CLEAR

GoodFood
exp://192.168.43.213:19000

Device ID: ea60-4538
Client version: 2.19.6
Supported SDKs: 38, 39, 40, 41

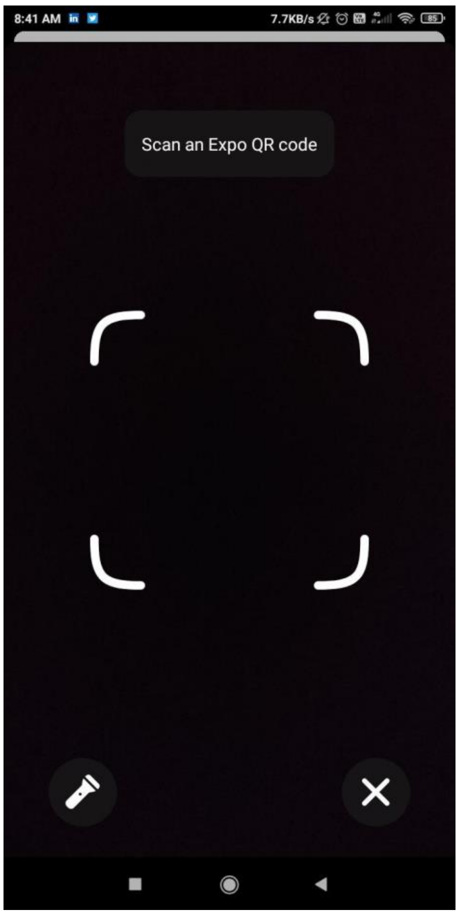
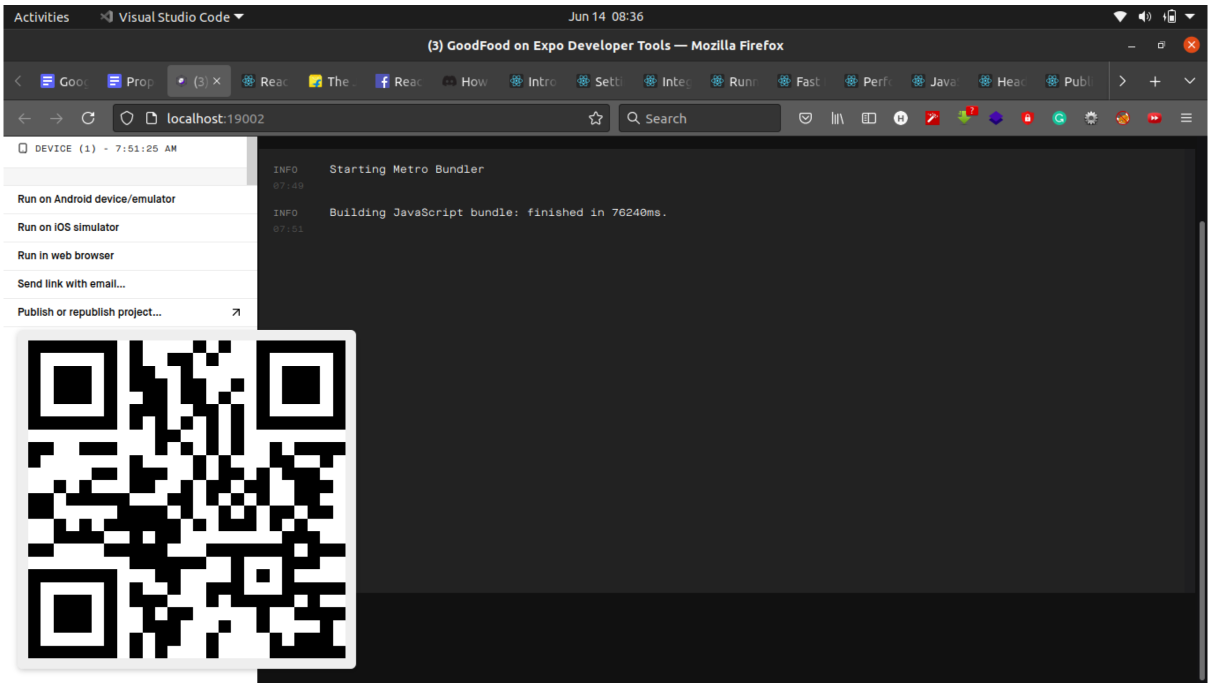


Figure 3: Expo Go **Figure 4. Scan QR code**



```
INFO   Starting Metro Bundler
07:49

INFO   Building JavaScript bundle: finished in 76240ms.
07:51
```

Figure 5: Metro Bundler on localhost:190002 (Development server)

Running your app on a simulator or virtual device

Expo CLI allows you to run your React Native app on a physical device without setting up a development environment. If you want to run your app on the iOS Simulator or an Android Virtual Device, please refer to the instructions for "React Native CLI Quickstart" to learn how to install Xcode or set up your Android development environment.

Once you've set these up, you can launch your app on an Android Virtual Device by running npm run android, or on the iOS Simulator by running npm run ios (macOS only).

Running your app on Android devices

Development OS

1. Enable Debugging over USB

Most Android devices can only install and run apps downloaded from Google Play, by default. You will need to enable USB Debugging on your device in order to install your app during development.

To enable USB debugging on your device, you will first need to enable the "Developer options" menu by going to **Settings** → **About phone** → **Software information** and then tapping the Build number row at the bottom seven times. You can then go back to **Settings** → **Developer options** to enable "USB debugging".

2. Plug in your device via USB

Let's now set up an Android device to run our React Native projects. Go ahead and plug in your device via USB to your development machine.

Next, check the manufacturer code by using lsusb (on mac, you must first install lsusb). lsusb should output something like this:

```
$ lsusb

Bus 002 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub

Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

Bus 001 Device 003: ID 22b8:2e76 Motorola PCS

Bus 001 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub

Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub

Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

These lines represent the USB devices currently connected to your machine.

You want the line that represents your phone. If you're in doubt, try unplugging your phone and running the command again:

```
$ lsusb

Bus 002 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub

Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub

Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

You'll see that after removing the phone, the line which has the phone model ("Motorola PCS" in this case) disappeared from the list. This is the line that we care about.

Bus 001 Device 003: ID 22b8:2e76 Motorola PCS

From the above line, you want to grab the first four digits from the device ID:

22b8:2e76

In this case, it's 22b8. That's the identifier for Motorola.

You'll need to input this into your udev rules in order to get up and running:

```
echo 'SUBSYSTEM=="usb", ATTR{idVendor}=="22b8", MODE="0666", GROUP="plugdev"' | sudo tee
/etc/udev/rules.d/51-android-usb.rules
```

Make sure that you replace 22b8 with the identifier you get in the above command.

Now check that your device is properly connecting to ADB, the Android Debug Bridge, by running adb devices.

```
$ adb devices

List of devices attached

emulator-5554 offline # Google emulator

14ed2fcc device # Physical device
```

Seeing device in the right column means the device is connected. You must have **only one device connected** at a time.

3. Run your app

Type the following in your command prompt to install and launch your app on the device:

```
$ npx react-native run-android
```

If you get a "bridge configuration isn't available" error, see Using adb reverse.

Hint: You can also use the React Native CLI to generate and run a Release build (e.g. npx react-native run-android --variant=release).

Connecting to the development server

You can also iterate quickly on a device by connecting to the development server running on your development machine. There are several ways of accomplishing this, depending on whether you have access to a USB cable or a Wi-Fi network.

Method 1: Using adb reverse (recommended)

You can use this method if your device is running Android 5.0 (Lollipop) or newer, it has USB debugging enabled, and it is connected via USB to your development machine.

Run the following in a command prompt:

```
$ adb -s <device name> reverse tcp:8081 tcp:8081
```

To find the device name, run the following adb command:

```
$ adb devices
```

You can now enable Live reloading from the Developer menu. Your app will reload whenever your JavaScript code has changed.

Method 2: Connect via Wi-Fi

You can also connect to the development server over Wi-Fi. You'll first need to install the app on your device using a USB cable, but once that has been done you can debug wirelessly by following these instructions. You'll need your development machine's current IP address before proceeding.

Open a terminal and type /sbin/ifconfig to find your machine's IP address.

1. Make sure your laptop and your phone are on the **same** Wi-Fi network.
2. Open your React Native app on your device.
3. You'll see a red screen with an error. This is OK. The following steps will fix that.
4. Open the in-app Developer menu.
5. Go to **Dev Settings → Debug server host & port for device**.
6. Type in your machine's IP address and the port of the local dev server (e.g. 10.0.1.1:8081).
7. Go back to the **Developer menu** and select **Reload JS**.

You can now enable Live reloading from the Developer menu. Your app will reload whenever your JavaS-cript code has changed.

Building your app for production#

You have built a great app using React Native, and you are now itching to release it in the Play Store. The process is the same as any other native Android app, with some additional considerations to take into account.

Performance Overview

A compelling reason for using React Native instead of WebView-based tools is to achieve 60 frames per sec-ond and a native look and feel to your apps. Where possible, we would like for React Native to do the right thing and help us to focus on your app instead of performance optimization.

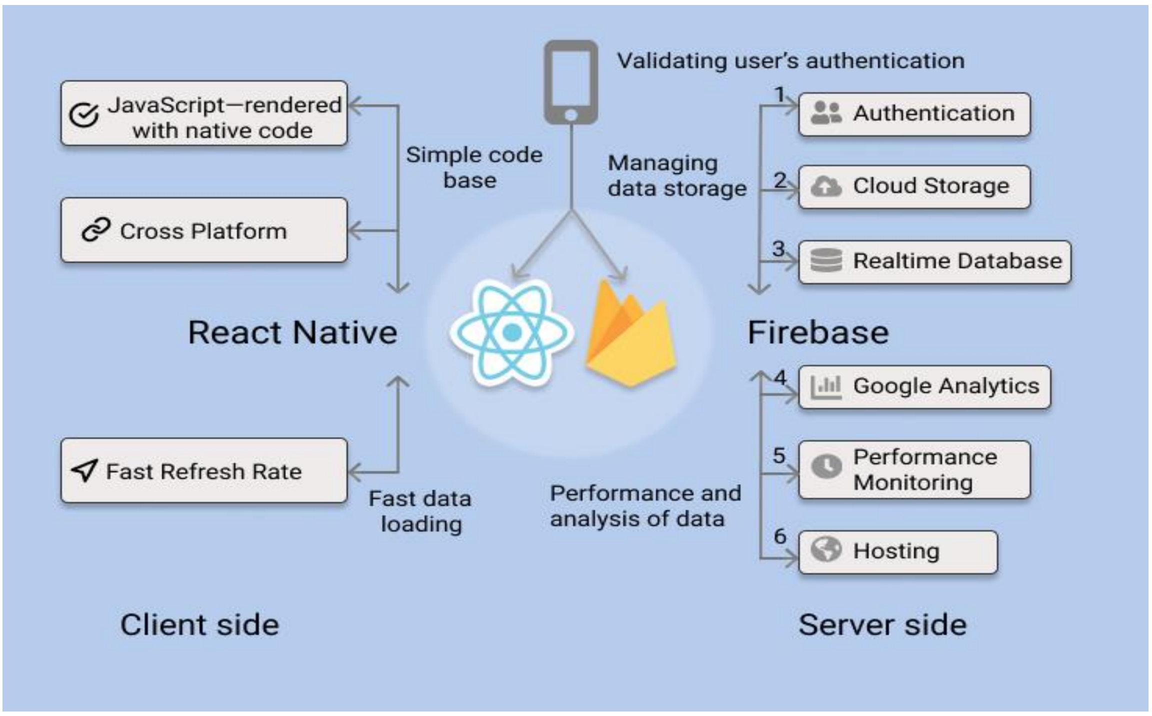


Figure 6: React Native & Firebase

3.3.2 Firebase

Firebase is Google's mobile application development platform that helps you build, improve, and grow your app.Firebase also gives you functionality like analytics, databases, messaging and crash reporting so you can move quickly and focus on your users. Firebase is built on Google infrastructure and scales automatically, for even the largest apps.

To use Firebase in the Expo Go app with the managed workflow, it is recommend using the Firebase JS SDK. It supports Authentication, Firestore & Realtime databases, Storage, and Functions on React Native. Other modules like Analytics are not supported through the Firebase JS SDK, but you can use expo-firebase-analytics for that. If you'd like access to the full suite of native firebase tools, we recommend using the bare workflow and react-native-firebase, because we cannot support this in the Expo Go app currently.

Luckily, the Firebase JavaScript SDK starting from version 3.1+ has almost full support for React Native, so adding it to our Expo app is super easy. The one caveat covered later in this guide is that the user login components typically provided by the Firebase SDKs will **not** work for React Native, and thus we will have to work around it.

Prerequisites

Before getting started, the documentation assumes you are able to create a project with React Native and that you have an active Firebase project. If you do not meet these prerequisites, follow the links below:

- Getting started with React Native
- Create a new Firebase projectFig[redacted]

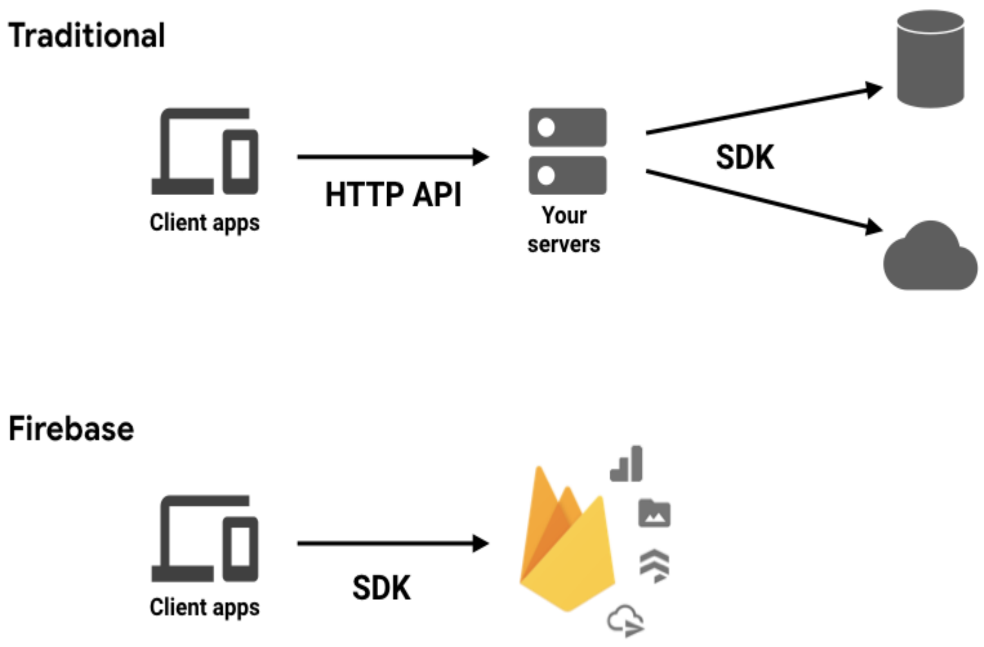


Figure 7: Firebase SDK (software development kit)

1. Install via NPM

Install the React Native Firebase "app" module to the root of your React Native project with NPM or Yarn:

```
# Using npm

npm install --save @react-native-firebase/app
```

```
# Using Yarn

yarn add @react-native-firebase/app
```

The @react-native-firebase/app module must be installed before using any other Firebase service.

2. Installing using Expo

First we need to set up a Firebase Account and create a new project. We will be using the JavaScript SDK provided by Firebase, so pull it into your Expo project.

```
expo install firebase
```

Firebase Console provides you with an API key, and other identifiers for your project needed for initialization. firebase-web-start has a detailed description of what each field means and where to find them in your console.

```
import firebase from 'firebase/app'
// Optionally import the services that you want to use
//import "firebase/auth";
//import "firebase/database";
//import "firebase/firestore";
//import "firebase/functions";
//import "firebase/storage";
// Initialize Firebase
const firebaseConfig = {
  apiKey: 'api-key',
  authDomain: 'project-id.firebaseapp.com',
  databaseURL: 'https://project-id.firebaseio.com',
  projectId: 'project-id',
  storageBucket: 'project-id.appspot.com',
 messagingSenderId: 'sender-id',
  appId: 'app-id',
measurementId: 'G-measurement-id',
};
firebase.initializeApp(firebaseConfig);
```

User Authentication

Firebase Authentication takes care of getting users logged in and identified. This product is essential to getting some of the other products configured properly, especially if you need to restrict access to per-user data (which nearly every app will want to do).

What's special about Firebase Authentication is that it makes easy to perform secure logins, which is incredibly difficult to implement correctly on your own.Firebase SDKs provide authentication methods for developers, so they don't have to reimplement common login systems such as Google or Facebook login.

This includes UI elements in the Web, Android, and iOS SDK versions for Firebase, however, these UI components do not work with React Native and **should not** be called. Thankfully, Firebase gives us ways to authenticate our data access given that we provide user authentication ourselves.

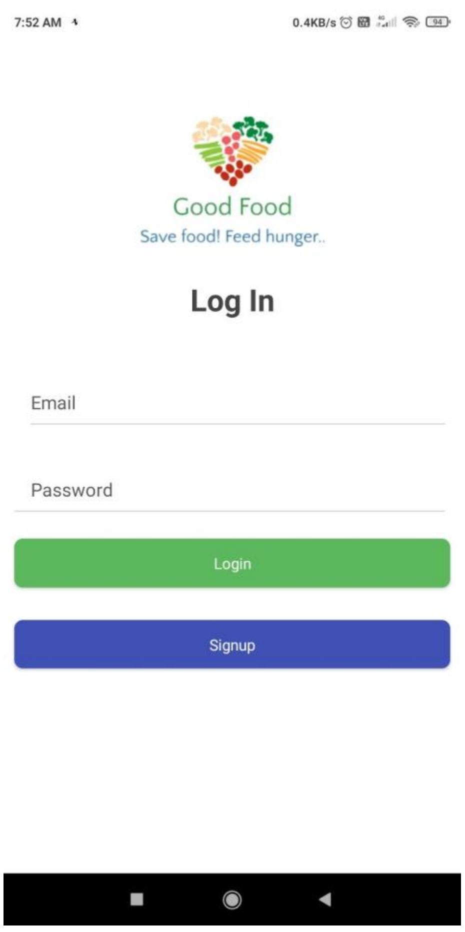


Figure 8: Firebase Authentication

Login Methods

We can choose different login methods that make sense to our application. The login method choice is orthogonal to the Firebase RTDB access, however, we do need to let Firebase know how we have set up our login system so that it can correctly assign authentication tokens that match our user accounts for data access control. You can use anything you want, roll your own custom login system, or even forego it altogether if all your users can have unrestricted access - though unrestricted access is strongly discouraged and instead Firebase recommends using their *Anonymous* authentication provider.

Firebase Realtime database Cloud firestore

Firebase Realtime Database and **Cloud Firestore** provide database services. I listed them both as “realtime, cloud hosted, NoSQL databases”. They have individual strengths and weaknesses, and you may need to do some research to figure out which one is better for your needs. Hint: start with Cloud Firestore, as it likely addresses more of your needs (and it’s also massively scalable). You can use either one, or both together, if that suits your app.

It’s worth noting that Firestore is technically a Google Cloud product, not a Firebase product. Why is it listed with Firebase? Firebase adds SDKs to use in your mobile app to make direct data access possible, removing the need for that pesky middleware component. There are other products listed here with a similar relationship with Google Cloud, which I’ll also note.

What’s really special about these databases is that they give you “realtime” updates to data as it changes in the database. You use the client SDK to set up a “listener” at the location of the data your app wants to use, and the listener gets invoked with that data repeatedly, every time a change is observed. This lets you keep your app’s display fresh, without having to poll the data of interest.

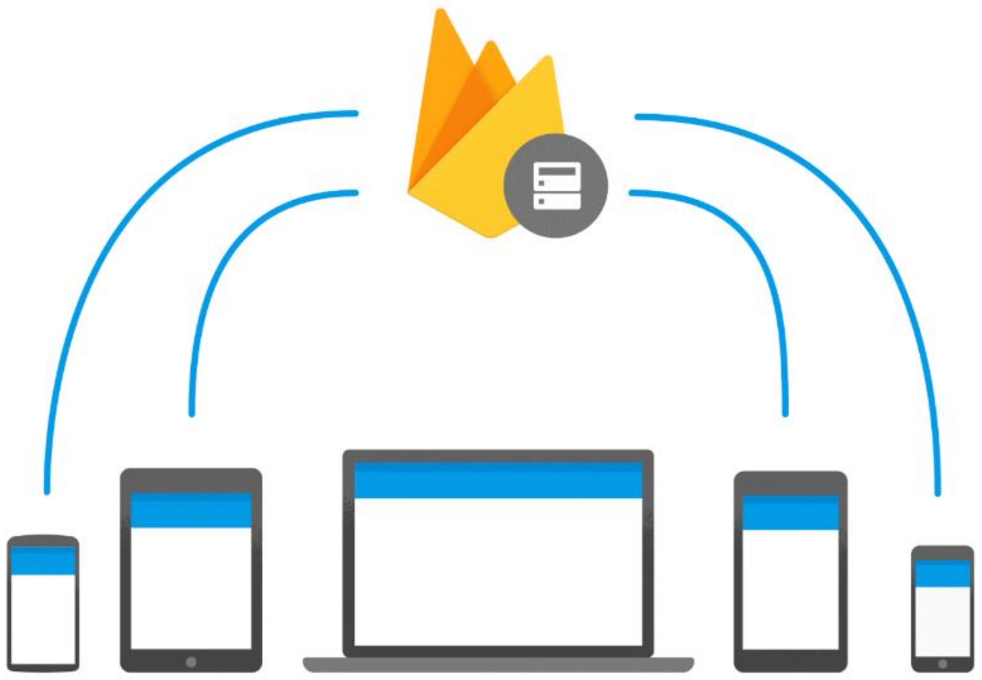


Figure 9: Firestore (All device access ability)

Installation

This module requires that the @react-native-firebase/app module is already set up and installed. To install the "app" module, view the Getting Started documentation.

```
# Install & setup the app module
yarn add @react-native-firebase/app
# Install the firestore module
yarn add @react-native-firebase/firestore
```

Using Expo with Firestore

Firestore a second database service in Firebase, the other being Realtime Database. Realtime Database can be thought of as a "JSON tree in the cloud" where your app can listen to and modify different portions of the tree. On the other hand, Firestore is a "document store" database. Your application will store and retrieve entire "documents" at a time, where a "document" is essentially a JavaScript object

```
import * as firebase from 'firebase'
import 'firebase/firestore';
const firebaseConfig = { ... }  // apiKey, authDomain, etc. (see above)
firebase.initializeApp(firebaseConfig);
const dbh = firebase.firestore();
dbh.collection("characters").doc("mario").set({
  employment: "plumber",
  outfitColor: "red",
  specialAttack: "fireball"
})
```

Cloud storage

Cloud Storage provides massively scalable file storage. It’s also technically a Google Cloud product, not a Firebase product. With Cloud Storage *for Firebase*, you get client SDKs to use in your app that enable you to upload and download files directly to and from your Cloud Storage “bucket”.

Authentication works extremely well with these three products with the use of **security rules** (for Realtime Database, Firestore, and Cloud Storage) that you can use to set access control to your data at the source. This ensures that clients can access that data *only* in the ways you allow, avoiding the tragic situation with the lolrus above. Users signed into an app with Authentication will automatically provide an identification token that you can use in your rules to protect who can read and write which items of data. So, if you store personal data about your users, *definitely* use Firebase Authentication with security rules to limit access appropriately. You may even get a gentle reminder from Firebase if your rules appear overly permissive.

Firebase Console

Firebase console allows the administrators to manage the work in the console window. All the backend management process of the firebase is completely controlled and maintained here. (Authentication, Realtime Database, Firestore, and Cloud Storage)

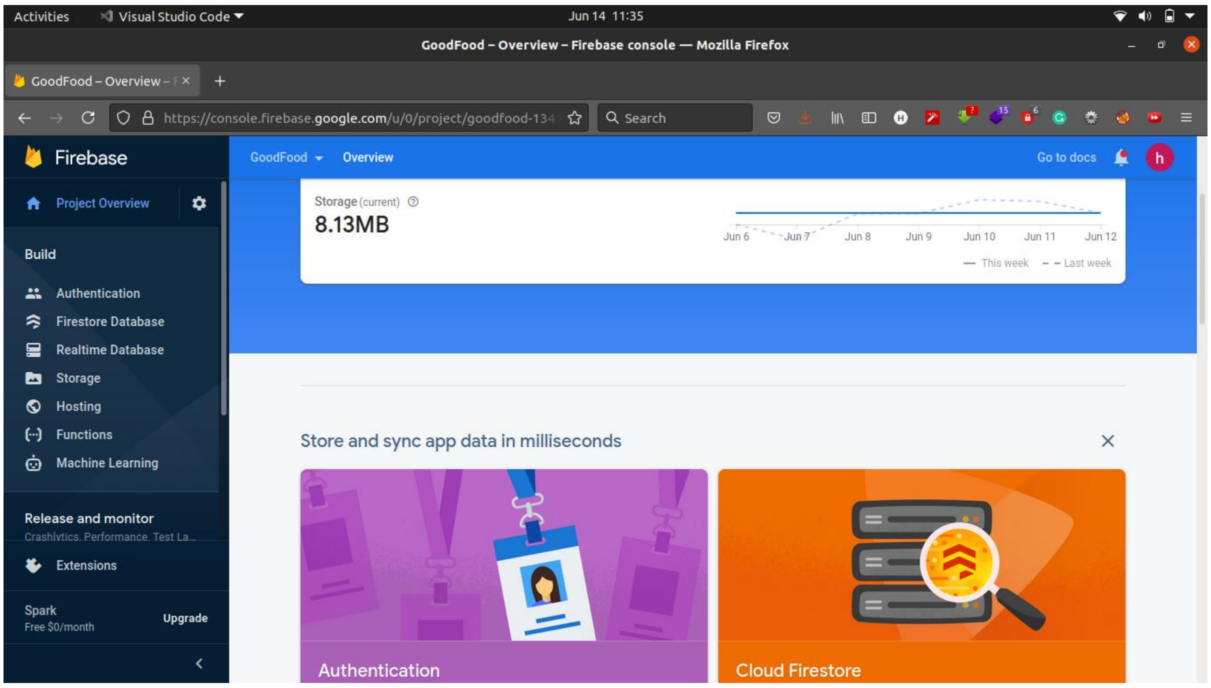


Figure 10: Firebase console

Chapter 4

Result Analysis

React Native which allows developers to develop apps in native style with a fast refresh rate and uses the same code base for both platforms makes it easier to build mobile apps. React Native smoothly interacts with firebase which performs all backend services required for the application. On the client side users provide the data as per the instructions in the applications which also serves to process data in the backend by firebase. Firebase validates the user authentication and manages user's data by storing in a realtime database. Firebase which works on the server side provides real time analytics and monitors performance of all the users.

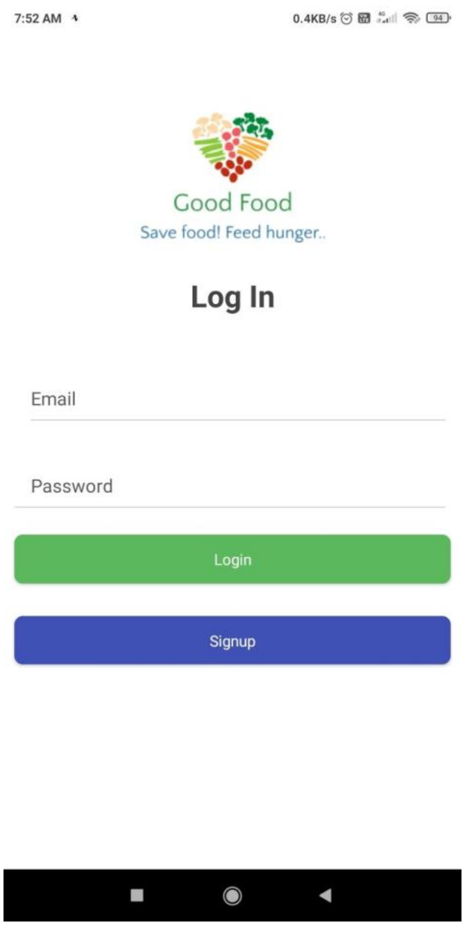
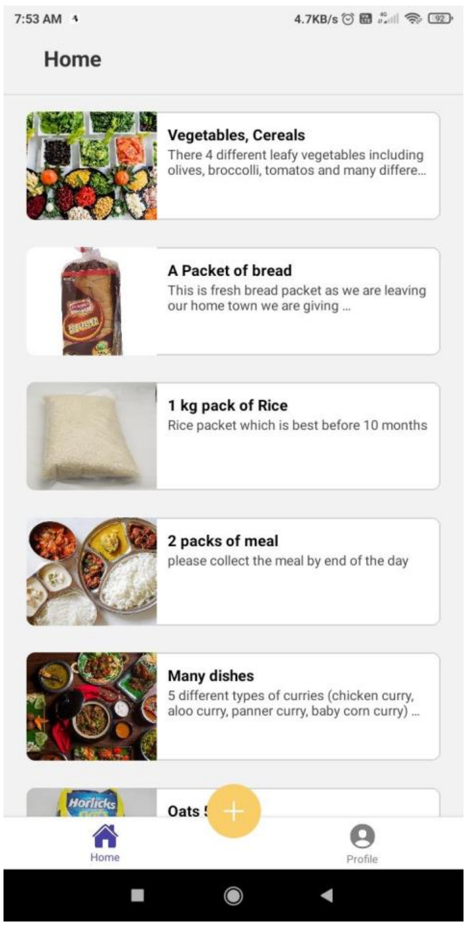


Figure 11: Login Screen



7:53 AM

Home

Vegetables, Cereals
There 4 different leafy vegetables including olives, broccolli, tomatos and many differe...

A Packet of bread
This is fresh bread packet as we are leaving our home town we are giving ...

1 kg pack of Rice
Rice packet which is best before 10 months

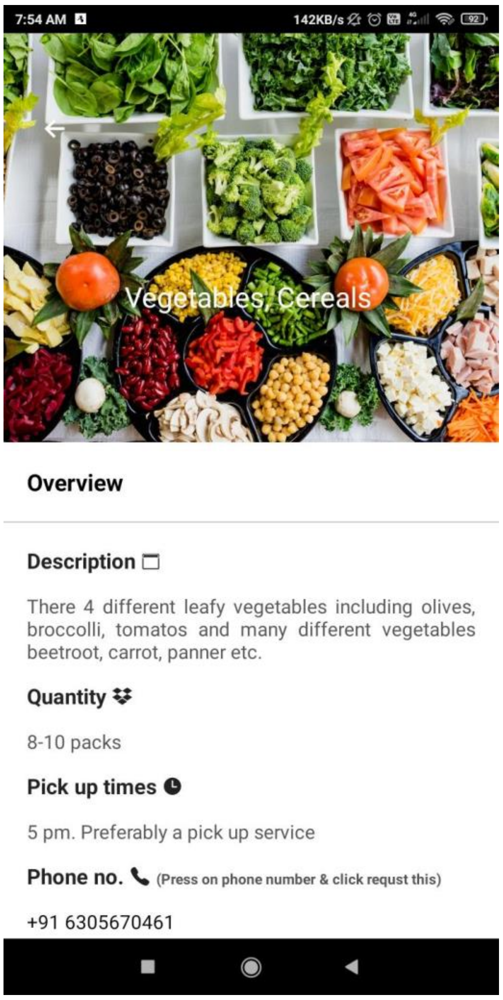
2 packs of meal
please collect the meal by end of the day

Many dishes
5 different types of curries (chicken curry, aloo curry, panner curry, baby corn curry) ...

Oats

Home Profile

Figure 12: Home Screen



Overview

Description

There 4 different leafy vegetables including olives, broccolli, tomatos and many different vegetables beetroot, carrot, panner etc.

Quantity

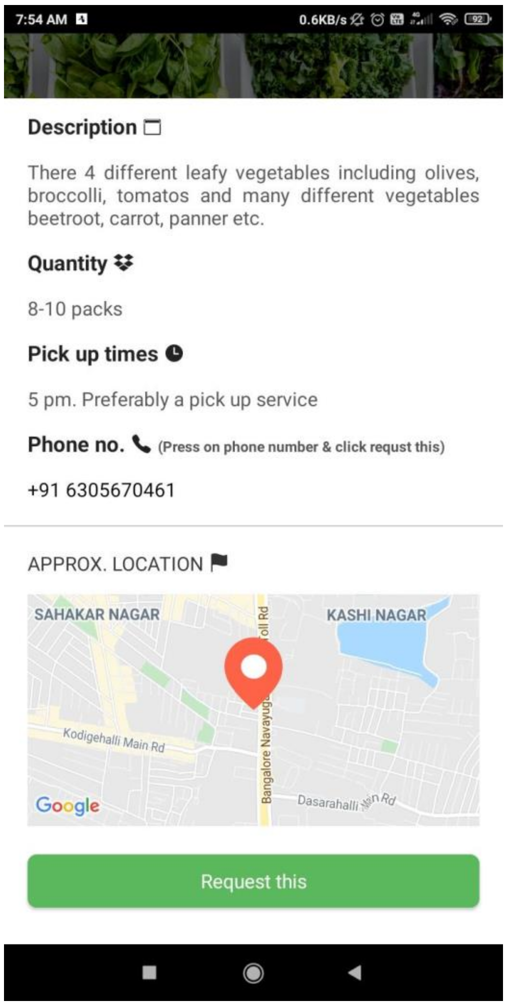
8-10 packs

Pick up times

5 pm. Preferably a pick up service

Phone no. (Press on phone number & click requst this)

+91 6305670461



Description

There 4 different leafy vegetables including olives, broccolli, tomatos and many different vegetables beetroot, carrot, panner etc.

Quantity

8-10 packs

Pick up times

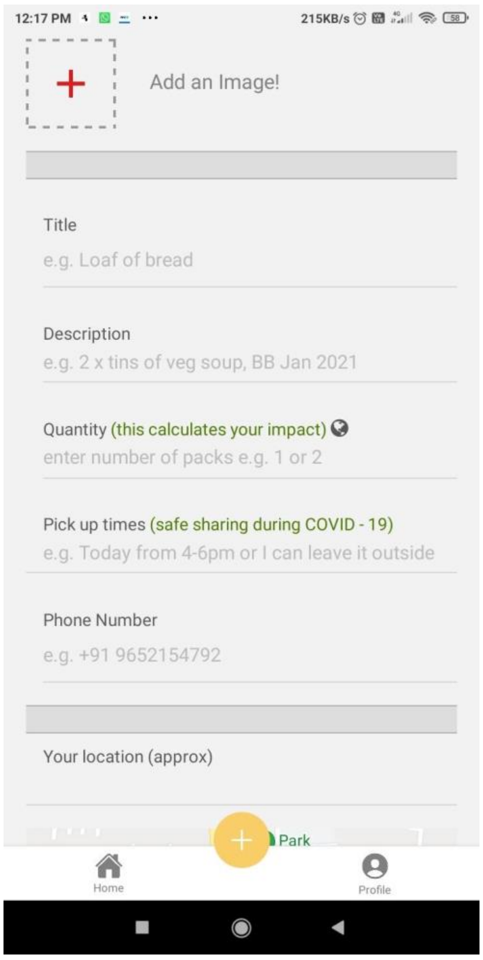
5 pm. Preferably a pick up service

Phone no. (Press on phone number & click requst this)

+91 6305670461

APPROX. LOCATION

Figure 13 & 14 : Overview and Description Screens


12:17 PM 215KB/s

Add an Image!

Title

e.g. Loaf of bread

Description

e.g. 2 x tins of veg soup, BB Jan 2021

Quantity (this calculates your impact)

enter number of packs e.g. 1 or 2

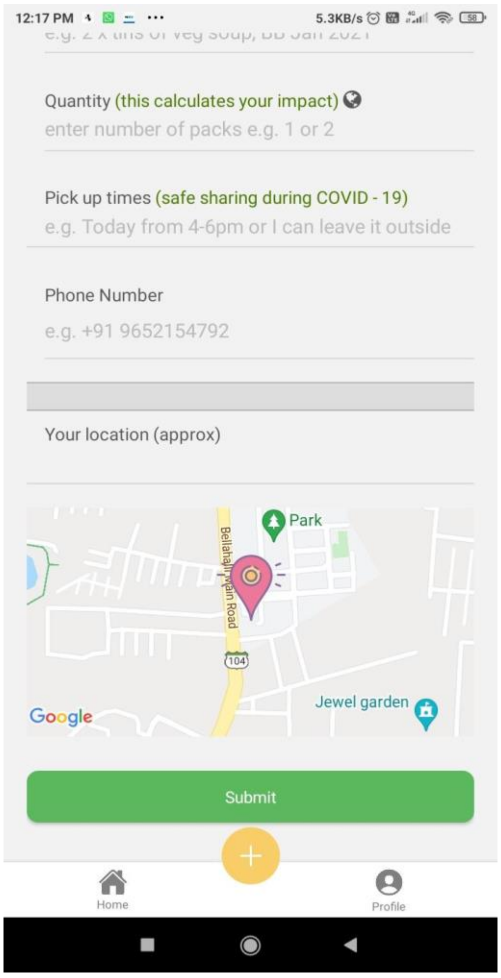
Pick up times (safe sharing during COVID - 19)

e.g. Today from 4-6pm or I can leave it outside

Phone Number

e.g. +91 9652154792

Your location (approx)


12:17 PM 5.3KB/s

Quantity (this calculates your impact)

enter number of packs e.g. 1 or 2

Pick up times (safe sharing during COVID - 19)

e.g. Today from 4-6pm or I can leave it outside

Phone Number

e.g. +91 9652154792

Your location (approx)

Figure 15 & 16: Post screen

The final result of our mobile application which allows users to manage food waste using our app. Here User logins into our app with their respective user Id and password. Then users find a list of items which are being donated by some user so everyone who signed up in our app can see them or even get them by opening and viewing the overview of the respective item they need. Users who want food have access to the photo description   phone number and respective location and pick up time of the vendor food from vendors which allow the clients to contact them and get the food that is required by them. Post screen allows the users to post an image of the food item that they are willing to donate and enter some details like title, description, quantity, pick up time, phone number and their location so other users who are in need can get the food from you.

Chapter 5

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION

The complex reasons behind why nearly one-third of all food produced for human consumption is wasted are evident throughout the food supply chain, from production to consumption. While there are many practical strategies that have been discussed to reduce food loss and waste (i.e., improving storage facilities, starting food waste awareness campaigns), these do not solve the underlying causes of why loss and waste still exist to such a large extent in today's word. So we would like to contribute a mobile application which saves food. Food Waste Management using a mobile application makes it easier to save excess food from wastage by simply donating it to the other users who have signed up for this mobile application. This proposed system ensures all the user integrity by setting up an Authentication system while logging in to the app which verifies the users with a real-time unique authentication token. This app also takes an account for saving user's location which helps the other users to find food in a specific location that is nearby to them. Minimizing food waste and denoting the food at the same time has become quite easier with this mobile application.

5.2 FUTURE ENHANCEMENT

A major problem faced by developers for applications like this is how to effectively manage data between the mobile devices and the server (Firebase).

- Rendering data from firebase (server) to mobiles can be made faster by effective data analysis.
- Understanding the working of Geolocation API helps in providing accurate user location.
- Enabling various kinds of contact methods (user-to-user messaging system) will enhance the communication between the users in a better way.
- Implementing a standard food information system on food packages that gives the user the information of both the name of the food, as well as its expiry date with OCR tools. However, the level of ease of using this option is only slightly greater than using the manual option of filling the food informtion.

Chapter 6

REFERENCES

[1] Food talks back: Exploring the role of mobile applications in reducing domestic food wastage Conference: In Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: the Future of Design At: University of Technology, Sydney, Australia Authors: *Geremy Farr-Wharton 5.6The Commonwealth Scientific and Industrial Research OrganisationJaz Hee-Jeong ChoiMarcus Fo*

[2] A Methodology for Sustainable Management of Food Waste Guillermo Garcia-Garcia,Elliot Woolley, Shahin Rahimifard, James Colwill, Rod White & Louise Needham Waste and Biomass Valorization volume 8, pages2209–2227(2017) Open Access Publication date: 25 November 2016.

[3] Waste Management in Restaurants May 2014 Authority:

[4] Food Waste ManagementSolving the Wicked Problem Editors: Närvänen, E., Mesiranta, N., Mattila, M., Heikkinen, A. (Eds.).

[5] Food waste management innovations in the foodservice industry September 2018 *Waste Management*79:196 DOI:*10.1016/j.wasman.2018.07.033*Authors:*Carlos Martin-Rios 22.79Christine Demen-Meier 4.49*.

[6] Other *reference* Projects *Sustainable Innovations in Food Service*, *Sustainable Innovations in Food Service*.

[7] *Android the world's most popular mobile platform |Android Developers*, 2017, [online] Available: https://developer. android.com/about/index.html.

Show Context *Google Scholar.*

Project Published details

Filled and published patent on 30/04/2021 and 11/06/2021 in Intellectual Property India | Patents

Patent Published link	: https://ipindiaservices.gov.in/PublicSearch/
Application number	: 202141019819
Title of Invention	: Food Waste Management Using Mobile Application
Field of Invetion	: Food
Publication Date	: 11/06/2021