SWE3999 - Technical Answers to Real World Problems (TARP)

Project Report

Karbonless - A Personal Carbon Footprint Tracker

By

18MIS1029	Vaishnavi S
18MIS1081	Daiyaan Ahmed
18MIS1007	Harshitha Devineni
18MIS1078	Abhishek T
18MIS1084	Janarthanan K
18MIS1088	Subha Shri S

M.Tech Software Engineering
(5 Year Integrated)

Submitted to

Dr. Nithya Darisini P.S

School of Computer Science and Engineering

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

December 2021

<u>**DECLARATION**</u>

I hereby declare that the report titled "**Karbonless - A Personal Carbon Footprint Tracker"** submitted by me to VIT Chennai is a record of bona-fide work undertaken by me under the supervision of **Dr. Nithya Darisini P.S and Dr. Asnath Victy Phamila Y**, School of Computer Science and Engineering, Vellore Institute of Technology, Chennai.

Vaishnavi.S

Vaishnavi S
Reg. No. 18MIS1029

Daiyaan Ahmed

Daiyaan Ahmed
Reg. No. 18MIS1081

[illegible]

Harshitha Devineni
Reg. No. 18MIS1007

[illegible]

Abhishek T
Reg.No. 18MIS1078

K. Janarthanan

Janarthanan K
Reg.No. 18MIS1084

[illegible]

Subha Shri S
Reg.No. 18MIS1088

CERTIFICATE

Certified that this project report entitled "**Karbonless - A Personal Carbon Footprint Tracker"** is a bonafide work of **Vaishnavi S (18MIS1029), Daiyaan Ahemd (18MIS1081), Harshitha Devineni (18MIS1007), Abhishek T(18MIS1078), Janarthanan K(18MIS1084), Subha Shri S(18MIS1088)** and they carried out the Project work under my supervision and guidance for SWE3999 - Technical Answers to Real World Problems (TARP).

Dr. Nithya Darisini P.S

SCOPE, VIT Chennai

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guides Dr. Nithya Darisini P S and Dr. Asnath Victy Phamila Y of School of Computer Science and Engineering for their consistent encouragement and valuable guidance offered to us throughout the course of this project work.

Vaishnavi. S

Vaishnavi S
Reg. No. 18MIS1029

Daiyaan Ahmed

Daiyaan Ahmed
Reg. No. 18MIS1081

[illegible]

Harshitha Devineni
Reg. No. 18MIS1007

[illegible]

Abhishek T
Reg.No. 18MIS1078

K. Janarthanan

Janarthanan K
Reg.No. 18MIS1084

[illegible]

Subha Shri S
Reg.No. 18MIS1088

ABSTRACT

The idea behind the carbon footprint tracker is to develop a mobile application that gives the users incentives to reduce their carbon footprint. The application takes user input like food consumed, mode of travel, etc to calculate the carbon emitted by the user's activity. The users are given a weekly carbon emission quota that is calculated from India's Per Capita emission for a year and are rewarded with badges if they manage to stay within the limit.

CONTENTS

1. Introduction

1.1 **Objective and goal of the project**

The main objective of this project is to make sure that the users are aware of the carbon footprint generated by their respective actions and help them keep track of it. By tracking their carbon footprint generated and analyzing them, we can provide resourceful insight that will help them reduce their carbon footprint, thus conserving energy as well. This ultimately makes a positive impact on the environment by reducing global warming and climate change while contributing to the 17 sustainable goals of the UN. The app has the following functionalities:-

- Track Individual Carbon Footprint
- Set an emission quota
- Maintain Karbon Kredits
- Reward with Incentives

1.2 **Problem Statement**

Climate change is affecting every country on every continent. It is disrupting national economies and affecting lives. Weather patterns are changing, sea levels are rising, and weather events are becoming more extreme. 2019 was the second warmest year on record and the end of the warmest decade (2010- 2019) ever recorded. (Goal 13)
Carbon dioxide (CO2) levels and other greenhouse gases in the atmosphere rose to new records in 2019. (Goal 13)Although greenhouse gas emissions were projected to drop during the pandemic, the global economy has begun to recover from the pandemic and the emissions are expected to return to much higher levels. Our idea for this problem not only helps with climate change but also contributes to sustainable production and consumption of resources while making life on earth better

2. Literature Survey

Our project focuses on carbon emissions from 3 major sectors namely, Transportation, Food and Manufacturing of products. Hence this literature review will be subdivided into these 3 subdivisions. Apart from these 3 categories, we have also included studies related to how electricity contributes to emission since it is the driving force behind the machinery involved in manufacturing.

2.1 **Transportation**

As part of the sustainable urban plan program, the capacity of public transportation modes has been identified to reduce CO2 emissions and energy consumption and to improve roadway safety. Increasing the amount of public transportation ridership shares compared to drive-alone methods of transportation mode would therefore be a huge step toward more environment-friendly and stress-free cities, and so this study aims to produce possible public transportation policies to be adopted by the government, policymakers or urban planners. Although public transportation is one of the major subsections of urban planning, it has a wide variety of aspects that affect locality and the environment, so a system dynamics approach is used to model and enhance the most realistic and practical CO2 mitigation scenarios for U.S. cities by adopting public transportation policies for the upcoming years. Based on historical data and applicable model validation processes, the behavior of various U.S. commuters' transportation mode choice and the capacity of transit transportation to mitigate CO2 emissions are both forecasted for the upcoming years up to 2050 under several possible policy scenarios. The results indicate that, in order to decrease fuel consumption and CO2 emission trends, marginal and ambitious scenarios should be incorporated. For instance, increasing public transportation ridership by 9% has the potential to reduce CO2 emissions by 766,000 tonnes annually in 2050, whereas a 25% increase in ridership could potentially reduce total cumulative CO2 emissions by 61.3 million tonnes.

As of now, the atmospheric emissions coming from various industries and means of transportation are expanding at a very fast pace across the globe. A survey was applied within the academic community of an institution to obtain transportation data and their respective emission rates. The results obtained show that 90.67% of instructors travel to the university in their cars, which contributes to a high carbon footprint potential; that is, emission of 51.91 kilograms of carbon dioxide equivalent (kg CO2equiv.) per month per instructor. For students, the emission is 6.97 kg CO2equiv. into the atmosphere per month per individual for those students who use the bus, as 63.32% of the university's students use public transportation to get around. In this way, the study provides a lesson regarding the environmental impact of the academic community, and it generates knowledge and aims at a more sustainable future by reducing environmental impacts through the means of transport that people choose to use.

Emissions from the transportation sector depend mainly on the type of transport and fuel apart from the type of combustion engine, emission mitigation techniques, maintenance procedures, and vehicle age. The major pollutant emitted from transport are Carbon dioxide (CO2), Methane (CH4), Carbon monoxide (CO), Nitrogen oxides (NOx), Nitrous oxide (N2O), Sulphur dioxide (SO2), Non-methane volatile organic compounds (NMVOC), Particulate matter (PM) and Hydrocarbon (HC).[1] (Ramachandra, T.V & Shwetmala, 2009). The paper also lists the emission factors for road vehicles in grams per kilometer based on the type of transport which is compiled from various papers (Mittal and Sharma, 2003; EEA, 2001; CPCB, 2007; Kandlikar and Ramachandran, 2000; UNEP, 1999)

2.2 **Food Production and Consumption**

The fact sheet compiled by the University of Michigan lists down the sources of emission by food. Food accounts for 10-30% of a household’s carbon footprint, typically a higher portion in lower-income households. Production accounts for 68% of food emissions, while transportation accounts for 5% [2] (University of Michigan,2020). The houses below the poverty line and large festive cooking consume coal thereby emitting a good amount of greenhouse gases. The production of food contributes more than the transportation factor due to the agricultural practices, fertilizers, and modern machinery used these days. Food production emissions consist mainly of CO_2, N_2O, and CH_4, which result primarily from agricultural practices [2] (University of Michigan,2020).

2.3 **Manufacturing & Electricity**

All Cities are the major source of socio-economic activities and the hotspot of consumers. With the increase of e-commerce and online shopping, consumer activities have started to account for a greater deal of carbon emissions including manufacturing, packaging, supply chain, and delivery operations. Tracking urban carbon flows from production to consumption by sectors indicated that Manufacturing was one of the largest emission-consuming sector, accounting for 81.9% of the total urban carbon flow[3]

Our findings have found that the following important suggestions reported by many countries with highest records of carbon emissions;(i) are the so-called fossil fuel capitalism needs to be overtaken, and a major switch to low carbon, eco-friendly, energy mix content is immediately required, (ii) renewable energy sources must be prioritized, (iii) adoption of electric vehicles should be made an important initiative, not just as complements to internal combustion engine vehicles but as substitutes should be encouraged, (iv) levying of environmentally sensitive taxes and subsidies must be intensified, and (v) better participation in the global drive for decarbonization must be encouraged.[1] (Ramachandra, T.V & Shwetmala, 2009).Washing clothes in 'cold' reduces CO2 emissions by 1.2-14.9 pounds per laundry load, depending on washing machine type, hot water temperature, and electricity source. Refrigerators are one of the largest users of household appliance energy; in 2019, an average of 672 lbs CO_2e per household was due to refrigeration [2] (University of Michigan,2020).

3 Requirements Specification

3.1 Hardware Requirements

Karbonless is an app created using flutter. In order to use the application a smartphone with A10+ Bionic Snapdragon 600+ processor is required with minimum 1GB RAM and 4GB ROM. Basic camera and GPS modules are also required to use all the features of the application. Since flutter is used for development the application can be easily ported to other platforms. It can be used as a mobile application, web application or as a desktop application. Hardware is required for the frontend usage only. The backend of our application is developed as a standalone API and is made open source hence any device or service can easily use our backend to calculate the carbon emission.

3.2 Software Requirements

Android or iOS is required for running the application along with JavaScript and Google Services. The application is developed using flutter which is a single code base for different environments. The backend of the application uses MongoDB for storing user data and is developed using NodeJS and Python.

4 System Design

The Karbonless app focuses on carbon emissions from 3 major sectors namely, Transportation, Food, and Manufacturing of products. The user activities related to these 3 sectors are tracked and their personal carbon footprint is calculated. Overall the app has the following modules :

- User Authentication
- User Logging Activity
- Track Carbon Footprint
- Maintain Karbon Kredits
- Compute Carbon Quota

4.1 User Authentication

This module is to authenticate the user and allow them to access their data. Authentication is done by collecting user credentials from the user, namely username, password, and verifying them with the existing data stored in the database. Once the user credentials are authenticated, a JWT(JSON web token) is provided to the user client to access the application. This token acts as an authenticator for the rest of the process until the user logs out or the token expires. JSON Web Token is a proposed Internet standard for creating data with optional signature and/or optional encryption whose payload holds JSON that asserts some number of claims. The tokens are signed either using a private secret or a public/private key.

New users can register a new account by providing user credentials and some basic information.

4.2 User Activity Logging

The primary objective of this module is to log all the activities that are performed by the users. This module is used to maintain all the historical data about the carbon footprint generated by the users through various activities. Data gathered by this module can be used to attain resourceful insights about the carbon footprint generated by the users and analyze the performance of the user. These logs can be filtered, sorted, and categorized using different parameters and can be represented visually using charts and statistics, which helps users to better understand the impact they are creating on the environment.

4.3 Track Carbon Footprint

In this module, we keep track of the carbon emission generated by the users on a daily basis and maintain their carbon footprint within acceptable limits. We accept parameters related to carbon emission and provide an equivalent carbon footprint generated by that activity. This activity can be categorized into Travel, Food, and Products. Each category has different input parameters and maintains the total emission caused by the activity. All activities are logged using the “User Activity Logging” module to maintain the carbon footprint limit.

4.4 Maintain Karbon Kredits

Rewards are given to users who manage to stay within the given acceptable limit and perform activities that have a positive impact on the environment. These rewards include Karbon Kredits, Rank Badges, and other incentives. Rank badges are given to users who manage to keep their carbon footprint within acceptable amounts and maintain their streak for a long period of time. These badges are ranked from Novice to Expert level, which depends on the daily, weekly, and monthly performance of the users. Karbon Kredits are the currency/points/coins which are awarded to the users based on their activities and their emission rates. Karbon Kredits are also awarded at each Rank level. These currency/points/coins can be spent on other carbon-related transactions and unlocking certain features.

4.5 Compute Carbon Quota

In this module, we compute the daily, weekly, and monthly carbon quota and keep track of the carbon footprint generated by the users against it. This module maintains the user streak and rewards them to keep their carbon footprint within the computed quota/limit. Carbon Quota is computed according to the data gathered by the Global Carbon Project [8] the annual carbon emission of the year 2020 in India, which was 1.770 tonnes per capita or 1770 kilograms for a year. From this data, the carbon emission per person for a day can be estimated to be around 4.84 kilograms. We further approximated it to 4.5 kg and this computed quota is used to check if the users manage to stay within the acceptable limit.

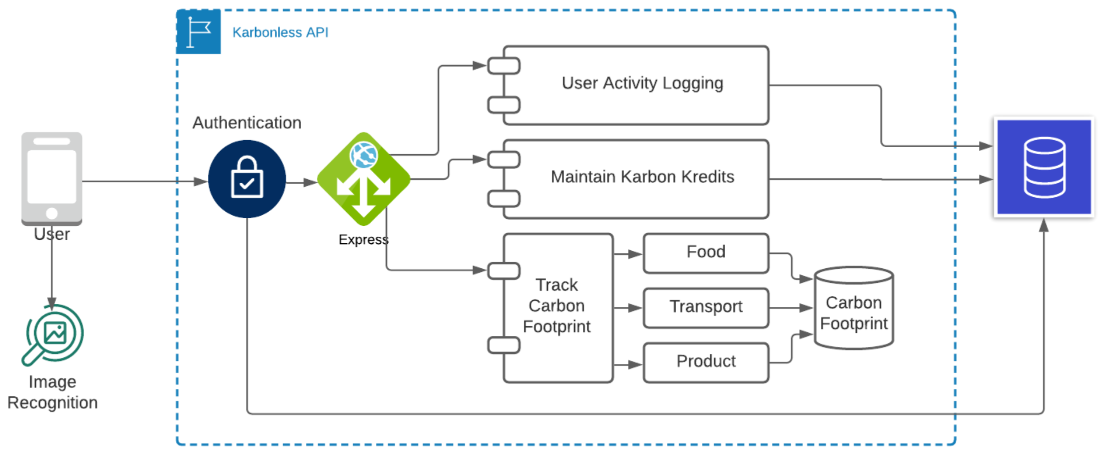


Fig 4.1 Architecture Diagram

5 Implementation of System

5.1 Calculating emissions from travel

The transport domain accounts for a 6.4 percent share of total India's Gross domestic product(GDP). Amidst the various transport sectors, road transport alone contributes to a 5.4 percent share of India's GDP. This mode of transport utilizes 78% of the energy share used for transport, while rail transport and air transport each utilize 11% of the energy share.[7]As part of sustainable urban planning, the potential of public transportation modes has been studied to decrease CO2 emissions and energy consumption and to increase roadway safety. Improving the number of public transportation ridership shares compared to drive-alone transportation mode would therefore be a huge step toward more environmentally friendly and stress-free cities, and so this study aims to produce possible public transportation policies to be adopted by policymakers or urban planners.

The two major fuels used by the transport sector are gasoline and diesel. These fuels contain carbon and mainly contain 80-85% of carbon by weight. On combustion of these, these fuels release greenhouse gases along with other pollutants thus contributing to the GHG effect.[7]

S.no	Category	kg CO_2/km
1	Petrol	0.1135
2	Diesel	0.1322
3	CNG	0.10768

Fig 5.1 Carbon emission of fuels

India is the 4th largest GHG contributor in the world. With the 2nd largest population in the world, the per capita emissions of the country are very low as compared to other developed and developing countries. Road transportation, being the dominant mode of transportation, emits 87% of the total CO2 equivalent emissions from the transport sector.[2]The major pollutants that are emitted from transport are Carbon dioxide (CO2), Methane (CH4), Carbon monoxide (CO), Nitrogen oxides (NOx), Nitrous oxide (N2O), Sulphur dioxide (SO2), Non-methane volatile organic compounds (NMVOC), Particulate matter (PM) and Hydrocarbon (HC).[1] (Ramachandra, T.V & Shwetmala, 2009).

The emissions from a vehicle depends on many other factors including ,but not restricted to ,engine efficiency,type of fuel,driving habit,vehicle maintenance,traffic,adulterated fuel,aging fleet,routes,passenger load,cargo load,management,urban planning,weather conditions,etc. The various results show that 90.67% of instructors travel to the university in their cars, which contributes to a high carbon footprint potential; that is, emission of 51.91 kilograms of carbon dioxide equivalent (kg CO2 Equiv.) per month per instructor. For university students, the emission is 6.97 kg CO2 Equiv. into the atmosphere per month per individual for those students who use the bus, as 63.32% of the university's students use public transportation to get around.[1] In this way, the study referred to provides a lesson regarding the environmental impact of the academic community, and it generates knowledge and aims at a more suitable and sustainable future by reducing environmental impacts through the means of transport that people choose to use.

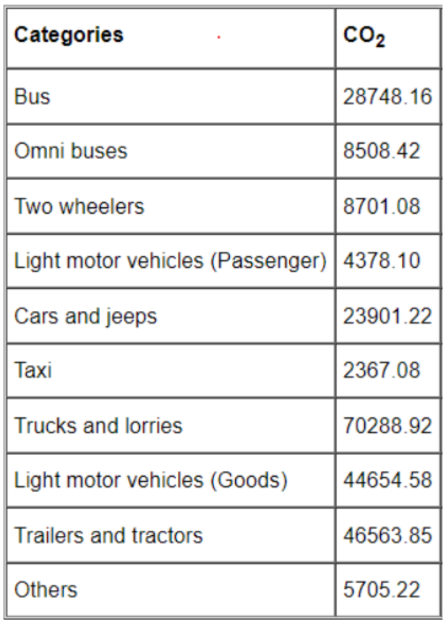
Inorder to calculate emissions factors,the basic data used are:

- Fuel type(gasoline,diesel,natural gas)
- Vehicle type(passenger cars,3 wheelers,2 Wheelers)
- Consideration of operating conditions (Speed,road,driving cycles,etc)
- Activity data(passenger-km,ton-km)
- Calorific value of fuel and emission factors fuel
- Fuel emission standard (g CO_2/Km,where applicable)

The formula used by the GHG organization to calculate the emission factor and the carbon emission for a few types of vehicles are listed below.

$$\text{GHG EF} = \frac{\text{Fuel consumption (TJ)} \times \text{Fuel Emission Factor (Tones } CO_2 \text{ eq/TJ)}}{\text{Passenger-km or Tones-km}}$$

Fig 5.2 Formula for calculating Greenhouse gas emission factor



Categories	CO_2
Bus	28748.16
Omni buses	8508.42
Two wheelers	8701.08
Light motor vehicles (Passenger)	4378.10
Cars and jeeps	23901.22
Taxi	2367.08
Trucks and lorries	70288.92
Light motor vehicles (Goods)	44654.58
Trailers and tractors	46563.85
Others	5705.22

Fig 5.3 Emission factor of vehicles

5.2 Calculating emissions from food consumption

The paper titled Carbon footprints of Indian food items[5] shows the emission data of greenhouse gases in various stages of their life cycle like Production, Processing, Transportation, and Preparation. This data was generated through a series of field experiments by the Indian Agricultural research institute. The average emissions among all these stages are calculated and stored in the backend. The data is available only for 24 food items and the user has to select food items available in the dropdown. According to the user input, the corresponding value is fetched from the database and the carbon expenditure is added. If a food item is not available from the list provided the user has the option to enter the name of the food item. This item will then be saved to the database and updated at a later stage.

5.3 Calculating emissions from products

Calculating emissions from the production and consumption of everyday products is important to assess the overall carbon footprint of a person's lifestyle. The data for quantifying CF of a minimum of ten regular products has been collected from various sources [9] [10] [11]. Bottled water, smartphones, plastic cutlery and other regular products that users can purchase on a daily basis have a carbon footprint that can heavily impact the environment. The user can opt to click a picture of the product they have purchased through the Karbonless application. This image will then be processed through a machine learning algorithm powered which will detect the object in the image and correspondingly fetch its factor of carbon footprint from our collected data. This will then be added to the users overall carbon footprint score and be used to quantify their daily emission

5.4 Karbon Kredits

According to the data gathered by the Global Carbon Project [8] the annual carbon emission of the year 2020 in India was 1.770 tonnes per capita which is 1770 kilograms for a year. From this data, the carbon emission per person for a day can be estimated to be around 4.84 kilograms. The users have to log their activities and ensure they stay under their limit of 4.84 kg/per day. If they manage to stay within their limit, they receive a point. The users have to keep scoring points in a similar way and for each 7 points (after a 1-week streak) they receive badges that they can share on their social media.

5.5 Algorithms and Techniques

The quantified carbon footprint values through the proposed methodology will be converted to Karbon Kredits with an algorithm that will allot each credit based on inverse proportionality to the carbon footprint emitted by the person on a daily basis.
A separate algorithm will be devised to consolidate all computed carbon data and lay weightage and perform some fine-tuning before calculating carbon credits. This combined carbon score will be open for the user to view if they want to gain a deeper understanding of their carbon footprint and how it has been calculated.
The detection for purchased products by the user which will help in identifying the product and fetch its emission data will be performed by a pre-trained machine learning model called YoLo powered by openCV. This algorithm is fast as compared to other classification algorithms. In real time our algorithm processes 45 frames per second.This algorithm makes localization errors but predicts less false positives in the background. This algorithm is used for predicting the accurate bounding boxes from the image. The image divides into S xS grids by predicting the bounding boxes for each grid and class probabilities. Both image classification and object localization

techniques are applied for each grid of the image and each grid is assigned with a label. Then the algorithm checks each grid separately and marks the label which has an object in it and also marks its bounding boxes. The labels of the grid without an object are marked as zero.

The algorithm can predict with great accuracy a number of 80 objects from the COCOMO dataset, a database that contains annotated models of 80 regular objects. This algorithm frames object detection as a regression problem to spatially separated bounding boxes and associated class probabilities, which are predicted with a single neural network from full images in one evaluation. Besides, This algorithm selects GoogLeNet but not VGG-16 as base network, as their rates of the forwarding transport computation versus precision are the former is far faster than VGG. So, YOLO is fast by design and indeed real-time while keeping good accuracy.

6. Results and Discussion

With the increasing global warming, awareness is not adequate. Karbonless can be used to provide incentives to the users in order to use the application and change into a healthier and environment-friendly lifestyle. Karbonless application can be used to provide user with alternative transport options from source to destination along with their respective carbon emissions so they could make a environment-friendly choice. Karbonless can be used to suggest product(scope) choices with less carbon footprint so as to maintain a sustainable lifestyle. Karbonless can be promoted in offices and other social groups to incorporate a healthy green lifestyle

7. Conclusion and Future Work

The results of this application are to incentivize people around the globe to practice and incorporate an environment-friendly and sustainable lifestyle by limiting their carbon emission levels. This application demonstrates and works as it is built around the principles of behavioral psychology in the form of an ongoing awarding of the small number of reward points (Kredits) instantly/simultaneously following the completion of various tasks, activities, and goals, to be able to drive significantly higher engagement levels than those demonstrated in previous literature were exploring the intersection of awareness and descriptive incentives. Studies have demonstrated the presence of incentives such as rewards and points, matters to user engagement. A reward-based system encourages the user to maintain the lifestyle. Carbon labeling which is the practice of labeling the carbon emission of a product on its packaging is now being adopted by certain companies in the USA. According to a report by Independent [11], Unilever is set to use carbon labeling on 75,000 of its products and is about to test the labeling scheme by the end of 2021. This would introduce us to a new set of data and also make it easy to calculate the carbon footprint which can just be done by scanning the label.

8. REFERENCES

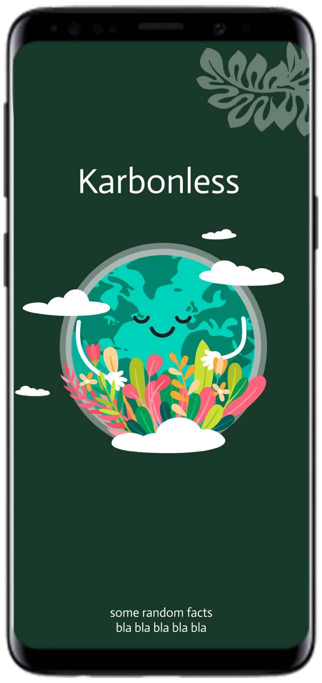
1. Ramachandra, T.V., Shwetmala, Emissions from India's transport sector: State Wise synthesis, Atmospheric Environment (2009), doi:10.1016/j.atmosenv.2009.07.015.
2. Center for Sustainable Systems, University of Michigan. 2020. "Carbon Footprint Factsheet." Pub. No. CSS09-05
3. Tracking urban carbon footprints from production and consumption perspectives Jianyi Lin4,1,2, Yuanchao Hu4,1,2, Shenghui Cui 5,1,2, Jiefeng Kang1,2 and Anu Ramaswami4,3 Published 30 April 2015 • © 2015 IOP Publishing Ltd.
4. John Lynch, Availability of disaggregated greenhouse gas emissions from beef cattle production: A systematic review, Environmental Impact Assessment Review,Volume 76,2019,Pages 69-78,ISSN 0195-9255,https://doi.org/10.1016/j.eiar.2019.02.003
5. Pathak, Himanshu et al. "Carbon footprints of Indian food items." Agriculture, Ecosystems & Environment 139 (2010): 66-73.
6. Seyi Saint Akadiri, Andrew Adewale Alola, Godwin Olasehinde-Williams, Mfonobong Udom Etokakpan, The role of electricity consumption,

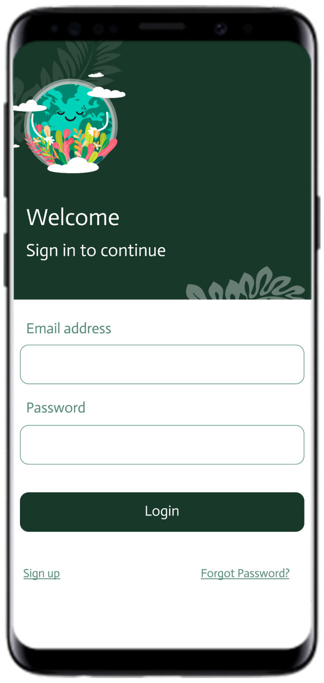
globalization and economic growth in carbon dioxide emissions and its implications for environmental sustainability targets,

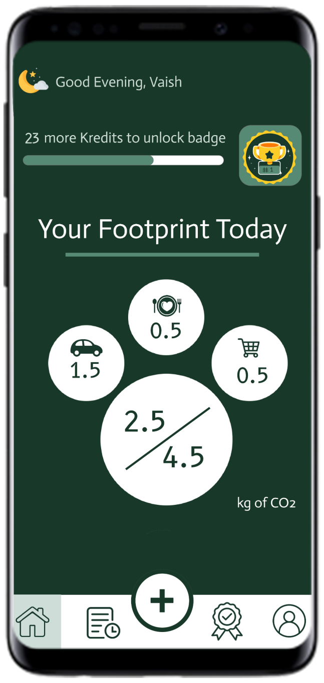
7. Science of The Total Environment, Volume 708, 2020, 134653, ISSN 0048-9697, https://doi.org/10.1016/j.scitotenv.2019.134653.
8. Gajjar, Chirag & Sheikh, Atik & Program, India. (2015). India Specific Road Transport Emission Factors. 10.13140/RG.2.2.28564.32646.
9. Global Carbon Project. (2021). Supplemental data of Global Carbon Project 2021 (1.0) [Data set]. Global Carbon Project. https://doi.org/10.18160/gcp-2021. https://ourworldindata.org/co2-emissions
10. Pacific Institute:Carbon Footprint of Bottled Water - https://iopscience.iop.org/article/10.1088/1748-9326/4/1/014009/pdf
11. Ercan, Mine & Malmodin, Jens & Bergmark, Pernilla & Kimfalk, Emma & Nilsson, Ellinor. (2016). Life Cycle Assessment of a Smartphone. 10.2991/ict4s-16.2016.15 https://www.researchgate.net/publication/308986891_Life_Cycle_Assessment_of_a_Smartphone/citation/download
12. Sheehan, Bill. (2017). Greenhouse Gas Impacts of Disposable vs Reusable Foodservice Products. 10.13140/RG.2.2.31993.93289. https://www.researchgate.net/publication/327076532_Greenhouse_Gas_Impacts_of_Disposable_vs_Reusable_Foodservice_Products/citation/download
13. https://www.independent.co.uk/climate-change/news/unilever-carbon-footprint-labels-food-b1882697.html

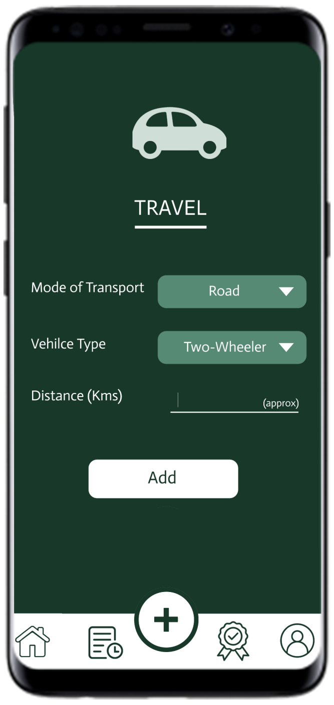
APPENDIX

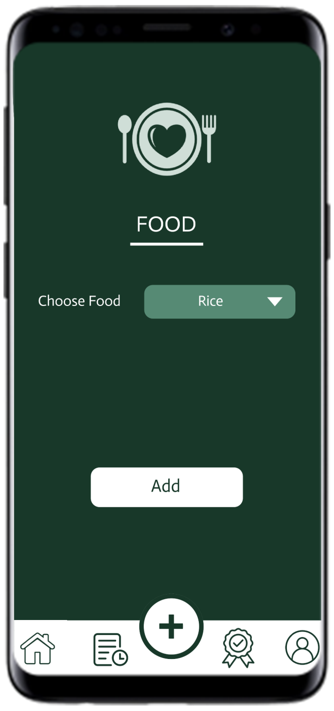
App Screen Mockups

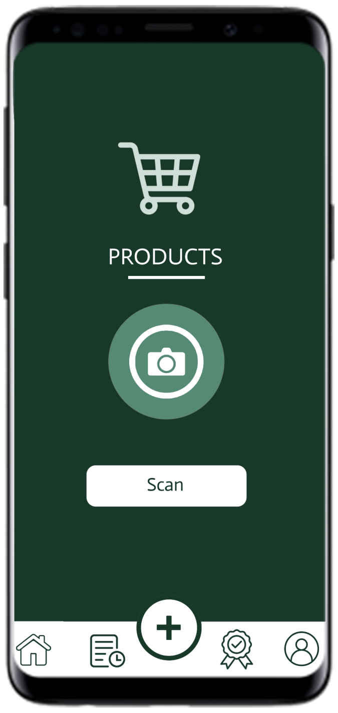


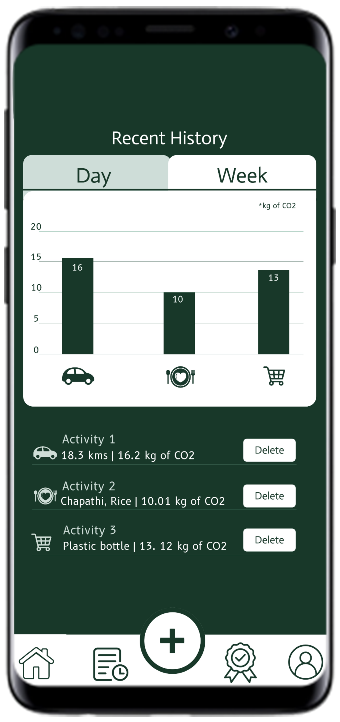








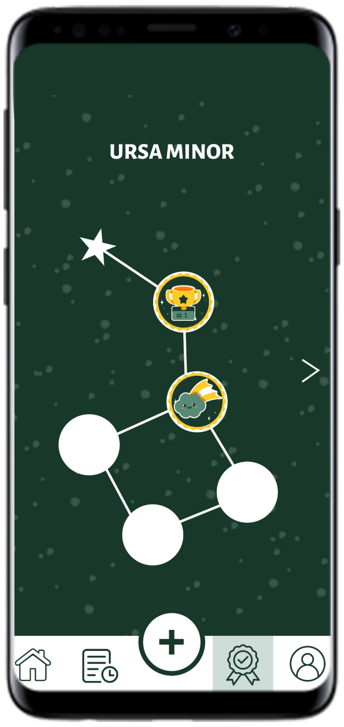




Activity 1
18.3 kms | 16.2 kg of CO2

Activity 2
Chapathi, Rice | 10.01 kg of CO2

Activity 3
Plastic bottle | 13. 12 kg of CO2



Project Code - 3 modules

Notification and State Management:

```
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:flutter_auth/store/travel_footprint.dart';
import 'package:flutter_local_notifications/flutter_local_notifications.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'package:velocity_x/velocity_x.dart';
import 'package:timezone/data/latest_all.dart' as tz;
import 'package:timezone/timezone.dart' as tz;
import 'food_footprint.dart';
class MyStore extends VxStore {
 PageController pageController;
 bool fabVisibility = true;
 Future<TravelFootprint> travelFootprint;
 int no_of_badges;
 int oldBadgeCount;
 final travelValues = InsertTravelValues();
 final foodValues = InsertFoodValues();
 TextEditingController distanceController;
 String userId;
 TextEditingController quantityController;
 String productValue;
 TimeOfDay breakfastTime;
 TimeOfDay lunchTime;
 TimeOfDay dinnerTime;
 FlutterLocalNotificationsPlugin plugin;
 String currentToken;
 String currentUser;
 void insertTravel() async {
 travelValues.distance = int.parse(distanceController.value.text);
 }
 void insertFood() {
 foodValues.quantity = int.parse(quantityController.value.text);
 }
 void setNotificationTimes() async {
 tz.initializeTimeZones();
 tz.setLocalLocation(tz.getLocation('Asia/Calcutta'));
 await plugin.zonedSchedule(
 0,
 'Time for Breakfast',
 'Add your breakfast to calculate carboon footprint',
 _nextInstanceOfTime(breakfastTime.hour, breakfastTime.minute),
 const NotificationDetails(
 android: AndroidNotificationDetails('1', 'Karbonize',
 channelDescription: 'daily notification description'),
 ),
 androidAllowWhileIdle: true,
 uiLocalNotificationDateInterpretation:
 UILocalNotificationDateInterpretation.absoluteTime,
 matchDateTimeComponents: DateTimeComponents.time,
 payload: 'Food');
 await plugin.zonedSchedule(
 0,
 'Time for Lunch',
 'Add your Lunch to calculate carboon footprint',
 _nextInstanceOfTime(lunchTime.hour, lunchTime.minute),
 const NotificationDetails(
 android: AndroidNotificationDetails('1', 'Karbonize',
 channelDescription: 'daily notification description'),
 ),
 androidAllowWhileIdle: true,
 uiLocalNotificationDateInterpretation:
 UILocalNotificationDateInterpretation.absoluteTime,
 matchDateTimeComponents: DateTimeComponents.time,
 payload: 'Food');
 await plugin.zonedSchedule(
 0,
 'Time for Dinner',
 'Add your Dinner to calculate carboon footprint',
 _nextInstanceOfTime(dinnerTime.hour, dinnerTime.minute),
 const NotificationDetails(
 android: AndroidNotificationDetails('1', 'Karbonize',
 channelDescription: 'daily notification description'),
 ),
 androidAllowWhileIdle: true,
 uiLocalNotificationDateInterpretation:
 UILocalNotificationDateInterpretation.absoluteTime,
 matchDateTimeComponents: DateTimeComponents.time,
 payload: 'Food');
 }
 }
```

```
tz.TZDateTime _nextInstanceOfTime(int hour, int minute) {
 final tz.TZDateTime now = tz.TZDateTime.now(tz.local);
 tz.TZDateTime scheduledDate =
 tz.TZDateTime(tz.local, now.year, now.month, now.day, hour, minute);
 if (scheduledDate.isBefore(now)) {
 scheduledDate = scheduledDate.add(const Duration(days: 1));
 }
 return scheduledDate;
}
```

Travel Screen:

```
import 'package:flutter/material.dart';
import 'package:flutter_auth/Screens/Dashboard/travel_page/travel_details.dart';
import 'package:flutter_auth/Screens/History/history.dart';
import 'package:flutter_auth/constants.dart';
import 'package:flutter_auth/store/travel_footprint.dart';
import 'package:flutter_auth/store/vxstore.dart';
import 'package:velocity_x/velocity_x.dart';
import 'package:http/http.dart' as http;
class TravelPage extends StatefulWidget {
 const TravelPage({Key key}) : super(key: key);
 @override
 _TravelPageState createState() => _TravelPageState();
}
class _TravelPageState extends State<TravelPage> {
 @override
 void initState() {
 MyStore store = VxState.store;
 store.fabVisibility = false;
 super.initState();
 }
 @override
 Widget build(BuildContext context) {
```

```
MyStore store = VxState.store;
Size size = MediaQuery.of(context).size;
return WillPopScope(
onWillPop: () async {
store.fabVisibility = true;
Navigator.pop(context);
return false;
},
child: Scaffold(
resizeToAvoidBottomInset: false,
// bottomNavigationBar: BottomNavBar(),
floatingActionButton: Container(),
body: SafeArea(
child: Container(
child: Column(children: [
Padding(padding: EdgeInsets.all(size.height / 35)),
Center(
child: Image.asset(
'assets/icons/car.png',
height: size.width / 4,
width: size.width / 3,
),
),
Padding(padding: EdgeInsets.all(size.height / 35)),
const Text(
'TRAVEL',
textAlign: TextAlign.center,
style: const TextStyle(
fontSize: 30,
fontWeight: FontWeight.bold,
color: Colors.white,
),
),
Padding(
padding: EdgeInsets.only(
left: size.width / 20,
right: size.width / 20,
top: size.width / 5),
child: Row(
mainAxisAlignment: MainAxisAlignment.spaceBetween,
children: [
const Text(
'Mode of Transport',
style: TextStyle(
fontSize: 24,
color: Colors.white,
),
),
// Padding(padding: EdgeInsets.only(right: size.width / 8)),
DropDown1(),
],
),
),
Padding(padding: EdgeInsets.all(size.width / 25)),
Padding(
padding: EdgeInsets.only(
left: size.width / 20, right: size.width / 20),
child: Row(
mainAxisAlignment: MainAxisAlignment.spaceBetween,
children: const [
Text(
'Vehicle',
style: TextStyle(
fontSize: 24,
color: Colors.white,
),
),
DropDown2(),
],
),
),
Padding(padding: EdgeInsets.all(size.width / 25)),
Padding(
padding: EdgeInsets.only(
left: size.width / 20, right: size.width / 20),
child: Row(
mainAxisAlignment: MainAxisAlignment.spaceBetween,
children: const [
Text(
'Distance (Kms)',
style: TextStyle(
fontSize: 24,
color: Colors.white,
),
),
DistanceTextField(),
],
),
),
Padding(padding: EdgeInsets.only(top: size.height / 8)),
Container(
width: size.width / 4,
child: TextButton(
child: const Text(
"Add",
style: const TextStyle(color: darkGreen, fontSize: 30),
),
style: ButtonStyle(
backgroundColor:
MaterialStateProperty.all<Color>(Colors.white),
shape: MaterialStateProperty.all<RoundedRectangleBorder>(
RoundedRectangleBorder(
borderRadius: BorderRadius.circular(10.0),
))),
onPressed: () async {
setState(() {
store.insertTravel();
showDialog(
context: context,
builder: (context) {
return Center(
child: Card(
color: darkGreen,
child: FutureBuilder<TravelFootprint>(
future: fetchEmission(store.travelValues),
builder: (context, snapshot) {
if (snapshot.connectionState ==
ConnectionState.waiting) {
return const CircularProgressIndicator
.adaptive();
} else {
return Column(children: [
Padding(
padding:
EdgeInsets.all(size.width / 8)),
const Icon(
Icons.done_all_rounded,
color: lightGreen,
 size: 80,
 ),
 Padding(
 padding: EdgeInsets.all(
 size.width / 15)),
 Padding(
 padding: const EdgeInsets.all(8.0),
 child: Center(
 child: Text(
 'The carbon emission for your recent travel is $
{roundDouble(snapshot.data.totalEmission, 4)} units and has been sucessfully added
to
your expenditure',
 textAlign: TextAlign.center,
 style: const TextStyle(
 fontSize: 25,
 // fontWeight: FontWeight.bold,
 color: Colors.white,
 ),
 ),
 ),
 ),
 Padding(
 padding: EdgeInsets.all(
 size.width / 15)),
 TextButton(
 onPressed: () {
 store.fabVisibility = true;
 Navigator.pop(context);
 Navigator.pop(context);
 },
 child: const Text(
 "Done",
 style: const TextStyle(
 color: darkGreen, fontSize: 30),
 ),
 style: ButtonStyle(
 backgroundColor:
 MaterialStateProperty.all<
 Color>(Colors.white),
 shape: MaterialStateProperty.all<
 RoundedRectangleBorder>(
 RoundedRectangleBorder(
 borderRadius:
 BorderRadius.circular(10.0),
 ))),
 )
 ]);
 }
 }),
 ),
 );
 });
 });
 },
 ),
 )
 ])))),
 );
 }
}
Future<List<String>> fetchModes() async {
 MyStore store = VxState.store;
 var response = await http.get(
 Uri.parse('https://karbonless-api.herokuapp.com/footprint/travel/all'),
 headers: <String, String>{'Authorization': 'Bearer ${store.currentToken}'},
 );
 List<String> modes = List<String>();
 final travelDetails = travelDetailsFromJson(response.body);
 print('Length: ${travelDetails.length}');
 for (int i = 0; i < travelDetails.length; i++) {
 modes.add(travelDetails[i].mode);
 }
 modes = modes.toSet().toList();
 print(modes);
 return modes;
}
Future<List<String>> fetchTypes() async {
 MyStore store = VxState.store;
 var response = await http.get(
 Uri.parse('https://karbonless-api.herokuapp.com/footprint/travel/all'),
 headers: <String, String>{'Authorization': 'Bearer ${store.currentToken}'},
 );
 List<String> types = List<String>();
 final travelDetails = travelDetailsFromJson(response.body);
 for (int i = 0; i < travelDetails.length; i++) {
 types.add(travelDetails[i].type);
 }
 types = types.toSet().toList();
 print(types);
 return types;
}
class DropDown1 extends StatefulWidget {
 DropDown1({
 Key key,
 }) : super(key: key);
 @override
 State<DropDown1> createState() => _DropDown1State();
}
class _DropDown1State extends State<DropDown1> {
 String value;
 @override
 Widget build(BuildContext context) {
 MyStore store = VxState.store;
 return Container(
 width: 150,
 height: 60,
 child: FutureBuilder<List<String>>(
 future: fetchModes(),
 builder: (context, snapshot) {
 return DropdownButtonFormField<String>(
 isDense: false,
 autovalidateMode: AutovalidateMode.always,
 decoration: InputDecoration(
 border: const OutlineInputBorder(
 borderRadius: BorderRadius.all(
 Radius.circular(10.0),
 ),
 ),
 filled: true,
 fillColor: lightGreen),
 // underline: Container(),
 style: const TextStyle(
 fontSize: 20,
 color: Colors.white,
 ),
 iconSize: 20,
 iconDisabledColor: Colors.white,
 iconEnabledColor: Colors.white,
 dropdownColor: lightGreen,
 value: value,
 items: !snapshot.hasData
 ? <String>['...'].map((String value) {
 return DropdownMenuItem<String>(
 value: value,
 child: Text(value),
 );
 }).toList()
 : snapshot.data.map((String value) {
 return DropdownMenuItem<String>(
 value: value,
 child: Text(value),
 );
 }).toList(),
 onChanged: (newValue) {
 setState(() {
 store.travelValues.mode = newValue;
 value = newValue;
 });
 });
 }),
 );
 }
}
class DropDown2 extends StatefulWidget {
 const DropDown2({Key key}) : super(key: key);
 @override
 _DropDown2State createState() => _DropDown2State();
}
class _DropDown2State extends State<DropDown2> {
```

```
String value;
@override
Widget build(BuildContext context) {
MyStore store = VxState.store;
return Container(
width: 150,
height: 60,
child: FutureBuilder<List<String>>(
future: fetchTypes(),
builder: (context, snapshot) {
return DropdownButtonFormField<String>(
isDense: false,
isExpanded: true,
decoration: InputDecoration(
border: const OutlineInputBorder(
borderRadius: BorderRadius.all(
Radius.circular(10.0),
),
),
filled: true,
fillColor: lightGreen),
style: const TextStyle(
fontSize: 20,
color: Colors.white,
),
iconSize: 20,
dropdownColor: lightGreen,
iconDisabledColor: Colors.white,
iconEnabledColor: Colors.white,
value: value,
items: !snapshot.hasData
? <String>['...'].map((String value) {
 return DropdownMenuItem<String>(
 value: value,
 child: Text(value),
 );
 }).toList()
 : snapshot.data.map((String value) {
 return DropdownMenuItem<String>(
 value: value,
 child: Text(value),
 );
 }).toList(),
 onChanged: (newValue) {
 setState(() {
 store.travelValues.vehicleType = newValue;
 value = newValue;
 });
 });
 }));
 }
}
class DistanceTextField extends StatefulWidget {
 const DistanceTextField({Key key}) : super(key: key);
 @override
 _DistanceTextFieldState createState() => _DistanceTextFieldState();
}
class _DistanceTextFieldState extends State<DistanceTextField> {
 TextEditingController controller = TextEditingController();
 @override
 Widget build(BuildContext context) {
 MyStore store = VxState.store;
 store.distanceController = controller;
 return Container(
 height: 60,
 width: 150,
 child: TextField(
 decoration: const InputDecoration(
 border: OutlineInputBorder(
 borderRadius: BorderRadius.all(
 Radius.circular(10.0),
 ),
 ),
 filled: true,
 fillColor: lightGreen),
 textAlign: TextAlign.end,
 keyboardType: TextInputType.number,
 controller: controller,
 style: const TextStyle(
 fontSize: 24,
 color: Colors.white,
 ),
 ));
 }
}
Future<TravelFootprint> fetchEmission(InsertTravelValues values) async {
 MyStore store = VxState.store;
 String vehicleType = values.vehicleType;
 String mode = values.mode;
 String distance = values.distance.toString();
 var response = await http.get(
 Uri.parse(
 'https://karbonless-api.herokuapp.com/footprint/travel?
type=$vehicleType&mode=$mode&distance=$distance'),
 headers: <String, String>{'Authorization': 'Bearer ${store.currentToken}'},
 );
 final footprint = travelFootprintFromJson(response.body);
 return footprint;}
```

Badges Screen:

```
import 'package:animated_background/animated_background.dart';
import 'package:awesome_dialog/awesome_dialog.dart';
import 'package:flutter/material.dart';
import 'package:flutter_auth/constants.dart';
import 'package:flutter_auth/store/iconHierarchy.dart';
import 'package:flutter_auth/store/vxstore.dart';
import 'package:shared_preferences/shared_preferences.dart';
import 'package:velocity_x/velocity_x.dart';
class Badges extends StatefulWidget {
 const Badges({Key key}) : super(key: key);
 @override
 _BadgesState createState() => _BadgesState();
}
class _BadgesState extends State<Badges> with TickerProviderStateMixin {
 @override
 void initState() {
 super.initState();
 WidgetsBinding.instance.addPostFrameCallback((timeStamp) async {
 MyStore store = VxState.store;
 Size size = MediaQuery.of(context).size;
 if (store.oldBadgeCount < store.no_of_badges) {
 await AwesomeDialog(
 btnOkColor: Colors.white,
 buttonsTextStyle: const TextStyle(color: darkGreen),
 dialogBackgroundColor: lightGreen,
 borderSide: const BorderSide(
 width: 7, color: Color.fromRGBO(251, 205, 38, 1)),
 customHeader: IconHierarchy().badges[store.no_of_badges],
 context: context,
 animType: AnimType.SCALE,
 dialogType: DialogType.INFO,
 body: Center(
 child: Column(
 children: const [
 Text(
 'CONGRATULATIONS!',
 style: TextStyle(
 fontWeight: FontWeight.bold,
 color: Colors.white,
 fontSize: 20),
 ),
 Padding(padding: EdgeInsets.only(top: 8)),
 Text(
 'You have unlocked a new badge!',
 textAlign: TextAlign.center,
 style: TextStyle(fontSize: 22, color: Colors.white),
 ),
 Padding(padding: EdgeInsets.only(top: 8)),
 Text(
 'Keep earning badges to unlock the constellation',
 textAlign: TextAlign.center,
 style: TextStyle(color: Colors.white, fontSize: 16),
 )
 ],
 ),
 ),
 title: 'CONGRATULATIONS',
 desc: 'You have unlocked a new badge!',
 btnOkOnPress: () async {
 SharedPreferences preferences =
 await SharedPreferences.getInstance();
 preferences.setInt('badgeCount', store.no_of_badges);
 },
 ).show();
 }
 });
 }
 @override
 Widget build(BuildContext context) {
 return AnimatedBackground(
 child: PageView(
 scrollDirection: Axis.vertical,
 children: const [Constellation1(), Constellation2()]),
 vsync: this,
 behaviour: RandomParticleBehaviour(
 options: const ParticleOptions(
 spawnMaxSpeed: 100,
 spawnMinSpeed: 10,
 baseColor: Colors.white)));
 }
 }
 class Constellation1 extends StatefulWidget {
 const Constellation1({Key key}) : super(key: key);
 @override
 _Constellation1State createState() => _Constellation1State();
 }
 class _Constellation1State extends State<Constellation1> {
 @override
 Widget build(BuildContext context) {
 MyStore store = VxState.store;
 var max = store.no_of_badges;
 if (store.no_of_badges > 5) max = 5;
```

```
 Size size = MediaQuery.of(context).size;
 return Column(children: [
 Padding(padding: EdgeInsets.all(size.height / 20)),
 const Text('URSA MINOR',
 textAlign: TextAlign.center,
 style: TextStyle(
 color: Colors.white, fontSize: 40, fontWeight: FontWeight.bold)),
 Padding(padding: EdgeInsets.all(size.height / 24)),
 max != 0
 ? Image.asset(
 'assets/badges/b$max.png',
 height: size.height / 2,
 )
 : Image.asset(
 'assets/badges/shape1.png',
 height: size.height / 2,
 ),
 const Padding(
 padding: EdgeInsets.only(top: 16.0),
 child: Icon(
 Icons.arrow_drop_down_sharp,
 color: Colors.white,
 size: 50,
 ),
 ),
 // Stack(alignment: Alignment.topLeft, children: createC1Children(size))
 ]);
 }
}
class Constellation2 extends StatefulWidget {
 const Constellation2({Key key}) : super(key: key);
 @override
 _Constellation2State createState() => _Constellation2State();
}
class _Constellation2State extends State<Constellation2> {
 @override
 Widget build(BuildContext context) {
 MyStore store = VxState.store;
 var max = store.no_of_badges;
 if (store.no_of_badges > 10) max = 10;
 Size size = MediaQuery.of(context).size;
 return Column(
 children: [
 const Padding(
 padding: EdgeInsets.only(top: 8.0),
 child: Icon(
 Icons.arrow_drop_up_sharp,
 color: Colors.white,
 size: 50,
 ),
 ),
 Padding(padding: EdgeInsets.all(size.height / 38)),
 const Text('CASSIOPEIA',
 textAlign: TextAlign.center,
 style: TextStyle(
 color: Colors.white,
 fontSize: 40,
 fontWeight: FontWeight.bold)),
 Padding(padding: EdgeInsets.all(size.height / 24)),
 max > 5
 ? Image.asset(
 'assets/badges/b$max.png',
 height: size.height / 2,
 )
 : Image.asset('assets/badges/shape2.png', height: size.height / 2)
 ],
 );
 }
}
```
Some API Calls:

Products:
```
 Future<List<ProductDetails>> fetchProducts() async {
 var request = http.MultipartRequest(
 'POST',
 Uri.parse(
 'https://karbonless-api.herokuapp.com/recognition/image'));
 request.files.add(await http.MultipartFile.fromPath(
 'input', image.path));
 var response = await request.send();
 var responseBody = await response.stream.bytesToString();
 debugPrint(responseBody);
 final productDetails =
 productDetailsFromJson(responseBody);
 var presponse = await http.get(
 Uri.parse(
 'https://karbonless-api.herokuapp.com/footprint/product/all'),
 headers: <String, String>{
 'Authorization': 'Bearer ${store.currentToken}'
 },
 );
 final products = allProductsFromJson(presponse.body);
 for (int i = 0; i < products.length; i++) {
 types.add(products[i].type);
 }
 for (int i = 0; i < productDetails.length; i++) {
 prods.add(productDetails[i].name);
 }
 return productDetails;
 }
Food:
Future<List<String>> fetchFood() async {
 MyStore store = VxState.store;
 var response = await http.get(
 Uri.parse('https://karbonless-api.herokuapp.com/footprint/food/all'),
 headers: <String, String>{'Authorization': 'Bearer ${store.currentToken}'},
 );
 List<String> types = List<String>();
 final foodDetails = foodDetailsFromJson(response.body);
 for (int i = 0; i < foodDetails.length; i++) {
 types.add(foodDetails[i].type);
 }
 return types;
}
User Information:
Future<int> readKredits() async {
 MyStore store = VxState.store;
 var response = await http.get(
 Uri.parse('https://karbonless-api.herokuapp.com/users/me'),
 headers: <String, String>{'Authorization': 'Bearer ${store.currentToken}'},
 );
 final user = userReadFromJson(response.body);
 store.userId = user.id;
 store.no_of_badges = user.kredit ~/ 100;
 setCount();
 print(user.kredit);
 print(user.kredit % 100);
 return user.kredit % 100;
}
```