Solution Design Document

Team A4 - Cashless Vending Machine

Version 1.2

Version	Date	Change List	Modified By
1.0	19/06/2020	Initial Draft	Team A4
1. 1	20/06/2020	Added sections Project Background, Exhaustive List of Requirements, Identified Scope of Minimum Viable Product (In Scope), Overall Block Diagram and Process Flow and Component Design	Team A4
1.2	21/07/2020	Integration of Components, Supporting POC's, Resources / Components Dependencies, MVP Implementation Cost and Time Estimation (Optional), References and Team Information	Team A4

Project Background

Cashless vending machine is a device that will provide you the product you need to buy without you paying cash to the vendor. This enables easy transactions and the user does not need to worry about the cash he/she has to carry everywhere. Cashless vending machines are available in the market but they are having some issues which need to be re-addressed:

1. Those machines don’t provide adequate feasibility for the user to use it fluently.
2. It needs physical intervention for the transaction to be completed and the purchase to be made by the user.
3. The complete process is not automated, which makes user think that directly providing cash is better than these Cashless Vending Machines.

These were only some of the issues faced by the pre-existing machines. This pre-existing system creates a lot of issues for the people using them on a regular basis. Usually these machines are to be seen in a commercial environment surrounded with people who work there; the small issue like you need to physically touch the machine to complete the purchase of the product is now going to have lot of trouble considering this Coronavirus situation. Also, these machines work on a singular unit basis i.e. it is a single machine having the program installed on it and provides all the data through that machine itself. This generates issues larger companies where a lot of machines are needed in that campus itself. These machines need to be based on an interconnected server-client system which will ensure centralized data is used throughout the range of machines used in that campus.

This will ensure that the company does not need to take note of each and every machine placed, separately. Instead they can just update the central database and the changes will be automatically reflected to all the machines under that network. Also getting to the point that user needs to physically intervene to the system for the process to complete is going to be changed. Users need to touch the small display of the machine to select what product they have to buy and also what quantity is needed by them to be bought from the vending machine. This will no longer be true because our project makes this process completely automated and you can stay away from machine by not going near or touching it to get your purchase done. We will create a secondary interface available to you so that interact with that 2nd interface to get your things done rather than touching the machine.

This Project will address the above issues and construct a reliable and more efficient Cashless Vending machine. Our team has also decided to add an extra functionality to the machine. This project will construct a vending machine which will be cashless as well as touch-less. Elaborating the “Touch-less” perspective of the project:

We all are aware of the Corona Virus or COVID-19 pandemic getting affected to more and more people not only in India but all the countries across the world. WHO (World Health Organization) has evaluated that this situation will need to be taken care of by all the people by staying away from each other - Social Distancing not only for now but also when the people are cured just so that they can prevent themselves from future risks. So, our team has planned to make a Vending Machine which will be completely Automated i.e. you will not need to touch the machine for any task while buying your product. This will be achieved by us using IoT in an effective way leading this project to another level. We will prepare a network of devices interconnected to each other to perform all the essential work in an orderly manner using server-based technology and IoT devices like Node MCU to make machine achieve the “automated functionality”.

This project will contain certain modules which will integrate together to make the whole system work in harmony. These modules are explained below:

1. **Vending Machine:** This is the main pillar of the system as it will be mainly responsible for all the tasks performed. This machine will contain a glass chamber containing all the products intended to sell. Those products will have unique ID and Price tags visible to the user when he/she visits to the machine. Also, a display will be available which will show essential transaction details during each purchase made on that machine.
2. **Processing Circuit:** A processing board of Arduino as it’s subcategory Node MCU ESP32 will be used as a controlling as well as communication device in the Vending machine. This will be used in the vending machine to provide the users an interface to interact with. By no means it will be touched by anyone but it will only display the essential information to the user like products available in the machine, QR code for payment, other acknowledgements and feeds user will need while buying stuff from the machine.
3. **Centralized Database:** This is a major part of our system as all the important data will be stored here. It will keep records of the inventory as well as the status of the machines connected to the Database making it easier for the management of the products from a single portal. Also, this database will have connections to all the resources like payment portals, displays on the vending machine and the software which is used by the user while ordering products at any given instance of time.
4. **Software (Mobile App):** This part of the project is meant to be user friendly and fluent enough to make the user feel safe as he buys the product from the vending machine. This Android Mobile Application will be connected to the central database which is connected to individual machines. This application will ensure that no user touches the machine. All the necessary inputs user needs to give to the machine to complete his order are given through his mobile. The interface which was earlier placed on the machine will now be in the hands of the user itself making the system feasible and reliable to certain aspects like security, safety, etc.

Complete system will be designed keeping in mind the User Experience in mind so that everyone in the society could use the product in a fluent manner without anything new to be learnt. The highlighted benefits of the system are:

- End-to-end connected system
- Smart IoT devices for Remote Connectivity
- Touchless Experience as precautionary measure in COVID19 Environment
- Network Connected System for remote Access
- Online Payments for Zero Contact Purchase
- Mobile Application Based Control for Users as well as Vendor.
- Sanitization of Machine as well as Product resulting into safe environment.

Exhaustive List of Requirements

The list and explanation of the requirements are as given below:

Vendor Based:

1. It should be profitable -

   The vendor should be comfortable with the costings of the vending machine as he shouldn't be facing loss in any kind while he sells his products through the machine.

2. Vending machine should be reliable –

   The vendor shouldn't be worrying about the mis-happenings to the machine and it should be reliable as well as dependable enough so that the vendor will be effectively using the machine with suitable timing for each purpose like vending time, refilling, checking of the collected data, etc. Also, the reliability extends towards the technical aspects of the system. The connectivity should be secure and encrypted to avoid data leaks and anonymous penetration into network.

3. Machine should have adequate security (physical / connectivity) –

   The security to this machine can be provided in two manners; in the physical aspect to avoid physical damage or stealing of products and the second one is the technical aspect mentioned above to prevent data leaks or anonymous penetration into the network. These security measures should be provided to maintain sustainability and qualitative measures from the side of manufactures or designers.

4. Machine should be fast –

The vending machine should perform the task in less time so that the user is not wasting time on vending some food or any product. It should be just as fast and easy as traditional face-to-face buying methodology. The process should initiate as soon as user interacts with the machine and each part of the process should be fast enough to make system feel snappy. Also, the transaction process should be performed in less time so that the next customer doesn't have to wait long.

5. Vending machine should be user friendly –

   As the scope of this machine is wide enough for the area of usage, the people using the machine will also be coming from a wide variety of backgrounds. To avoid confusions and to make the process of buying a product extremely easy, the machine should have a user-friendly interface so that each and every user, irrespective of his/her knowledge of technical things should easily be able to use the machine in an effective manner.

6. Easy to refill the products –

   The vendor will have to refill the products in regular time cycles and that process should not be tedious to him as that will result into ignorance to that activity. The vendor will feel better to do some other task rather than re-filling the machine. It is not appropriate. So, the machine should be designed in such a manner that the process of re-filling the content to the machine should be easy for the vendor to perform on a regular basis.

Conditional Requirements:

1.Vending machine should be Touchless –

   Considering the COVID 19 situation worldwide, the machine will be used in public environments, many people will buy stuff from the machine daily, so to prevent the contamination of products as well as the machine itself, the process of buying product from the machine should be completely touchless. This will affect into a more robust and cleaner future for this vending machine.

2. Transactions should be done in cashless manner –

   Taking into note, the Digital India Campaign, the transactions should be done digitally and also the COVID condition restricts people from touching un-necessary stuff. Money is the most circulating object throughout the country and this makes the currency more prone to containing the COVID-19 virus. Henceforth, we will adapt the system to cashless transaction process and this will enable less strain full and more effective way for payments to the vendor. This will ensure that not only the user is in advantage of not touching the machine completely but also it will ensure that the vendor too is in huge advantage as he does not need to collect money and keep records any more. This will bring out a huge change.

3. The machine should be Smart –

   Studying the pre-existing traditional vending machines, we observed that a new set of information can be obtained from the normal usage of the machine which will benefit the vendor in a drastic manner. Also, the mobile access to the machine makes use of the recent technologies to provide a stable standpoint in the technical aspects of the system. The mobile app will provide the users with the history of their previous purchases from the machine along with a statistical representation of different aspects like expenditure, history of products bought, weekly/monthly consumption of a certain snack or beverage, etc. This Smart nature of the machine will be a great upgrade to be aimed by a lot of vendors.

4. It should be well-sanitized periodically –

   The precautions to the pandemic should be taken not only by humans but also by the machines should lend a hand in making it happen. The vending machine will be designed in such a way that the sanitization of the products inside the machine as well as the machine itself will be done in a periodical manner. This sanitization will result into a safer and reliable option for the users out there.

5. It should collect the data for analysis purpose –

   If machines are connected to a centralized network system, it will be beneficial for the vendor to manage multiple units in a huge campus. And the benefits extend to the point that the data collected can also be statistically presented to observe certain aspects of the system in a month or week like performance, sales number, number of cancelled transactions, etc.; which will enable vendor to do the needed changes to increase the sales rate. This collected data will also help the vendor to focus on the marketing side of the market, as each person will perform the transaction, the phone number will be stored in the database. These phone numbers can be targeted on certain occasions when vendor will announce a Sale or Discount on products, it will first be notified to the gathered phone numbers. These applications are limitless, the collected data can be used in a huge variety of ways to profit vendor every time.

6. Vending machine should be attractive, luring people to use it –

   The physical appearance of the machine also holds a good consideration because people go to buy something only if it pleases them to do so. Hence, the looks of the vending machine play an important role in luring the people to buy stuff from it. The proposed idea by us is that we will make construction of the machine in Glass. The base of the machine will be metal to hold on the weight and the back plate of the machine will be metal for connectivity and ventilation purpose and the rest 4 sides will be made of Glass giving complete transparency to the mechanism inside the vending machine. This will not only look good and distinctive but also a huge amount of people will get attracted to the machine to get a glance of how their order is processed mechanically.

Functional Requirements:

1. Vending machine should have internet access –

   For the functionalities like the remote connectivity and cross platform access to the physical machine, the system should be continuously online and connected to the servers on the internet to parse the data across the connection.

2. Optional functionalities for offline working –

   The machine should provide the functionalities to make the working of the machines much easier and available in offline conditions also. The offline aspect of the machine should be designed in such a manner that the machine will perform all the transactions without connecting to the online database of the user. But it should be kept in mind that the internal working of the vending machine will remain online only, because all the connectivity of the servers is depending on the data provided by the payment portal as the transactions are completely cashless.

3. Fluent mechanism for product dispatchment –

   The most important functionality of the vending machine which will be commonly noticed by the end user is the speed or fluency of the mechanism when he orders something. Remember when you are at ATM and the machine starts making noises and you’re sure that money is coming out in a second or two. That reliability should be built up by using profound methods to build robust machinery which will be power efficient as well as fast and reliable.

Identified Scope of Minimum Viable Product (MVP)

In Scope

The minimum viable product (MVP) for the Cashless Vending Machine will be consisting of at least the complete process of transition of the product from the machine to the customer dispatched on demand. Considering the requirements stated above and the categorization of the requirements on basis of the scope of them in the system, we have stated 3 types of requirements, namely:

1. Vendor Based Requirements
2. Conditional Requirements
3. Functional Requirements

According to the definition *"A Minimum Viable Product (MVP) is a version of a product with just enough features to satisfy early customers and provide feedback for future product development."* Taking this into consideration, the MVP (Minimum Viable Product) for Cashless Vending Machine will be considering all the Conditional and Functional Requirements in this design itself. We have covered all the points in the Conditional and Functional Requirement for exception of some. The points in Scope in the MVP are:

1. Vending machine should be Touchless
2. Transactions should be done in cashless manner
3. The machine should be Smart
4. It should be well-sanitized periodically
5. Vending machine should have internet access
6. Fluent mechanism for product dispatchment

All these requirements are explained in detail in the section above. These elements are the foundational requirements of the machine. If any of these requirements are eliminated, the vending machine will not work as intended or be in a useless state. NOTE: Sanitization part of the requirement does not affect any working of the vending machine but it is the basic need of each and every outdoor object in COVID-19 situation. Hence it is included in the MVP listing.

Out Scope

The non-considered elements of the MVP which are not implemented or designed are:

1. Optional functionalities for offline working
2. It should collect the data for analysis purpose

These two aspects are required for MVP but aren’t included in the design. The design and modelling of these two requirements was not possible due to the restrictions of time for the Internship Project. These two elements definitely are needed for MVP as they contribute a major impact on the overall working of the system as well as the general user interaction with the Vending Machine.

Overall Block Diagram and Process Flow

Overall Block Diagram of Cashless Vending Machine:

Fig 1. Master Block Diagram

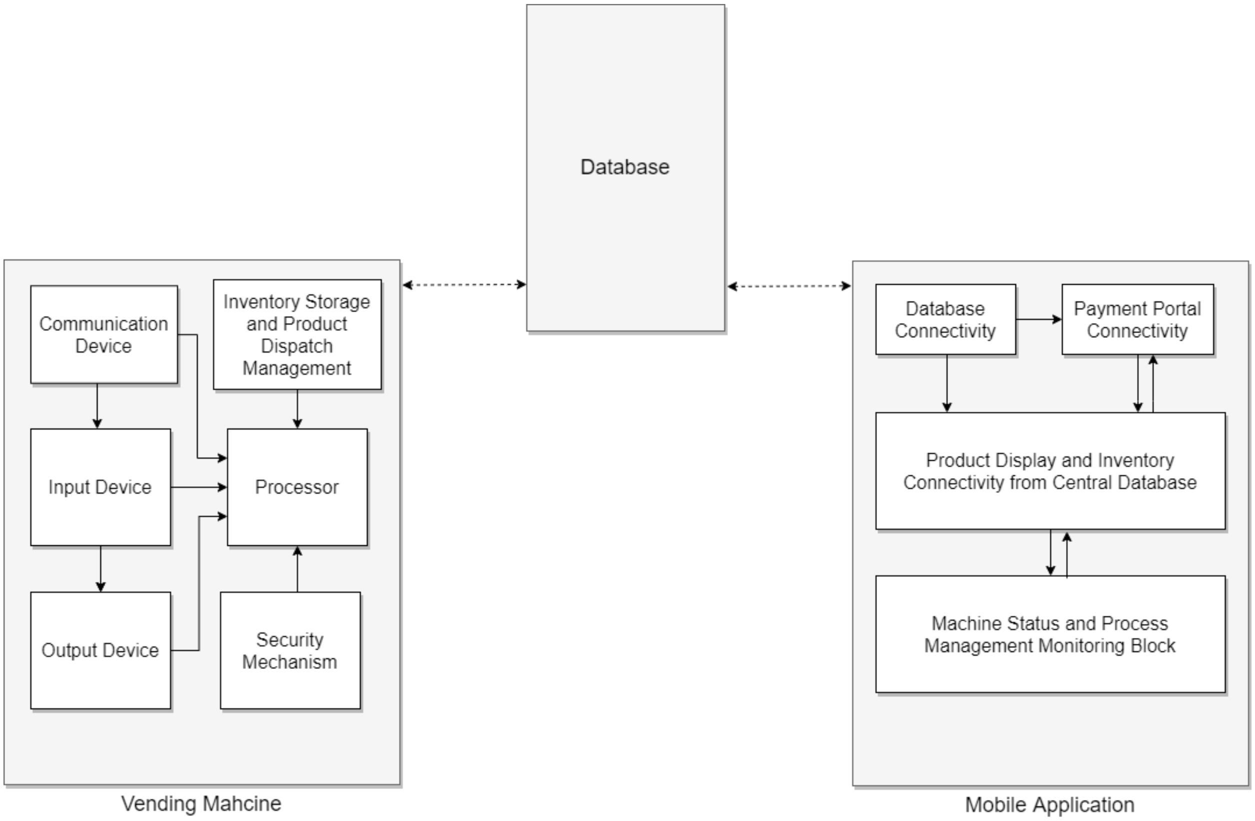

Database
Communication Device
Inventory Storage and Product Dispatch Management
Input Device
Processor
Output Device
Security Mechanism
Vending Mahcine
Database Connectivity
Payment Portal Connectivity
Product Display and Inventory Connectivity from Central Database
Machine Status and Process Management Monitoring Block
Mobile Application

Fig 2. Vending Machine Block

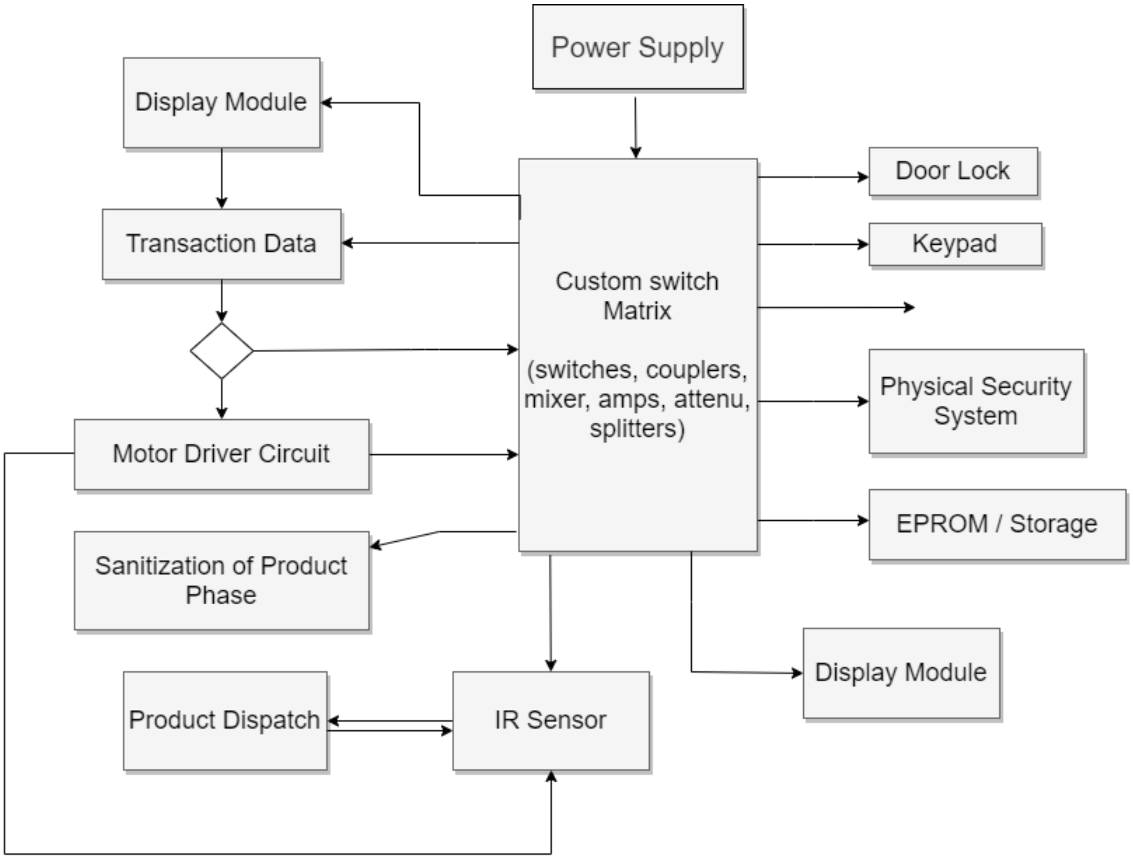


Fig 3. Security System Block

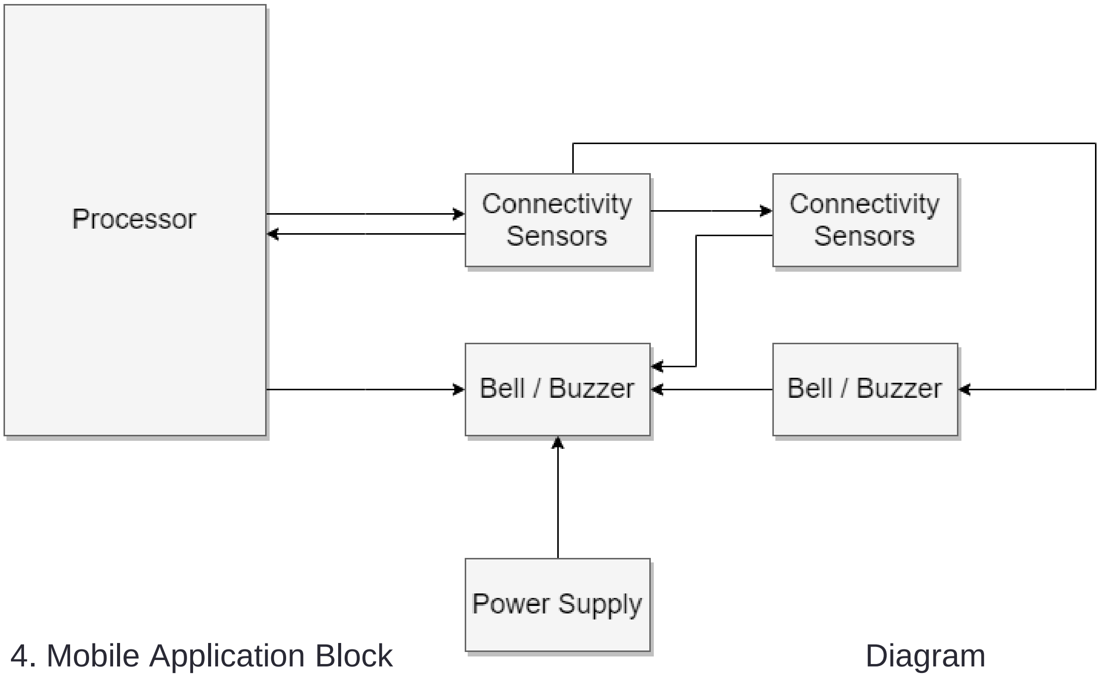
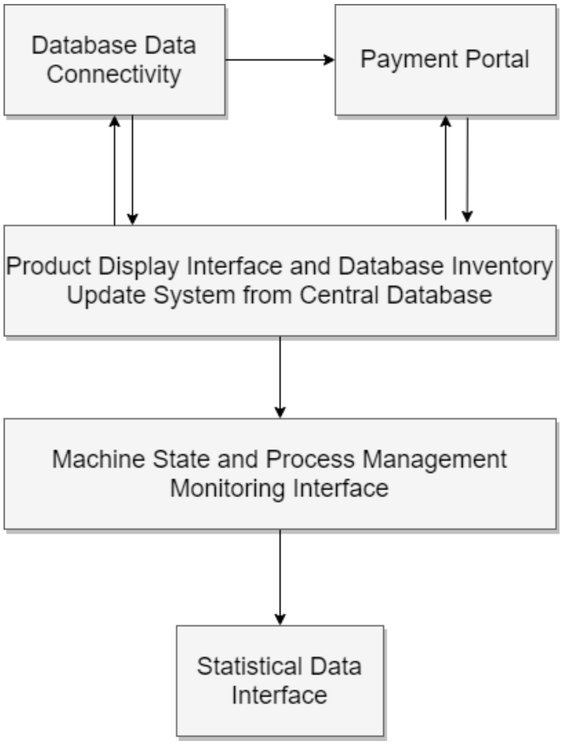
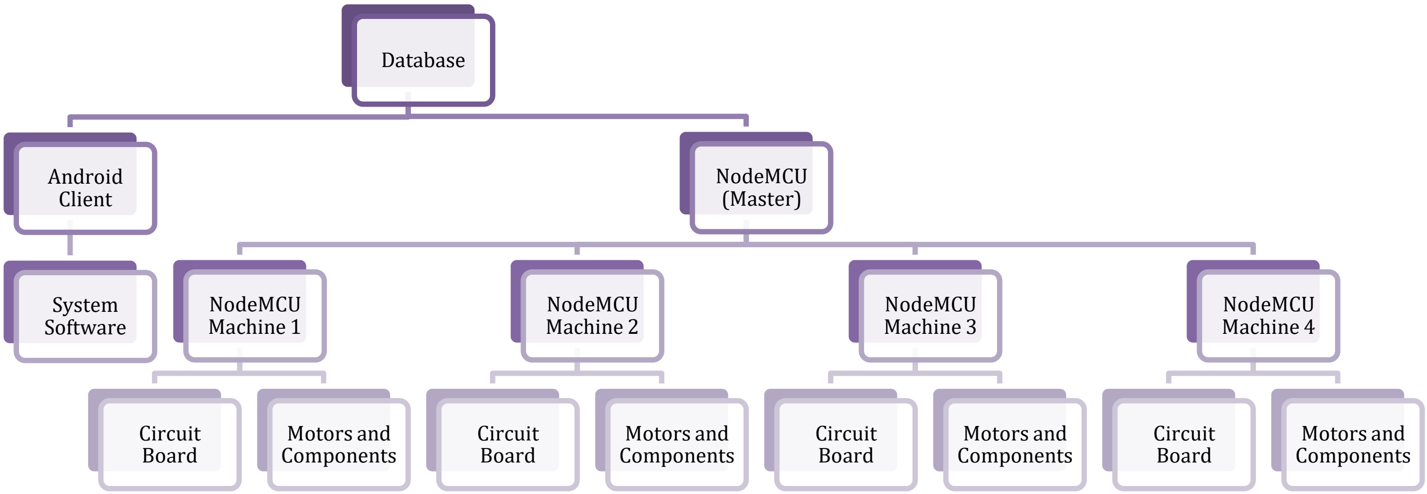


Fig 4. Mobile Application Block Diagram


Database Data Connectivity
Payment Portal
Product Display Interface and Database Inventory Update System from Central Database
Machine State and Process Management Monitoring Interface
Statistical Data Interface

These are the basic block diagrams for the system representing the structure of the system in a broad perspective. The functional arrangement of the components in their own objective position is shown in the diagram shown below:



Process Workflow:

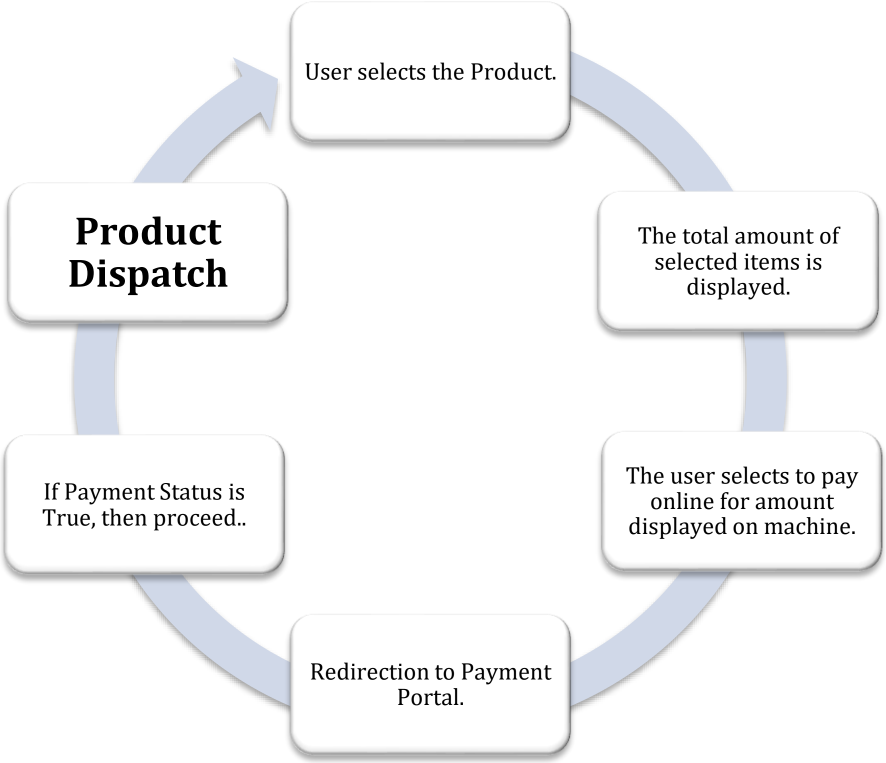

User selects the Product.
The total amount of selected items is displayed.
The user selects to pay online for amount displayed on machine.
Redirection to Payment Portal.
If Payment Status is True, then proceed..
Product Dispatch

Fig 5. Process Workflow

1. Product Selection:

   This is the first interaction of user with the machine. Here, the end user selects the product he is willing to buy from the vending machine from the mobile itself by scanning the QR code displayed on the machine. He will only be shown the list of products that are available in the machine he is standing in front of.

2. Display Acknowledgement:

   After the user selects the product he needs to buy, he / she is shown the amount the user has to pay for purchasing the items selected. The “To be Paid” amount will be reflected on the vending machine as well as the mobile application used by the user.

3. Payment Interface:

   When the user will select to Pay, he will be re-directed to the payment gateway connected to the backend of the system.

4. Payment Process:

When the payment portal is successfully opened in the mobile device used by the user, the user will then input the required data for the payment like phone number and OTP provided by the Banking Authorities. Doing this will complete the payment.

5. Post-payment Status:

When the payment is successful, the payment gateway API will submit a unique key for the acknowledgement of completion of a transaction triggered by the user. Once the key is received by the server, payment is recorded in our system.

6. Product Dispatch:

Once system records the successful payment of an order, the machine triggers the dispatch mechanism which will move the necessary components of the vending machine in order to dispatch the product ordered by the user.

Yay!! The ordered product is received by the user…

Components

The components in this system are interconnected to each other for the purpose of completion of the designed modules. Each module is interconnected to each other in such a way that data which is generated in a single module can be passed to other components with minimum latency so that machine is fast and fluent when a transaction is triggered by the user.

The components listed below are fitted into a vending machine chassis designed specifically for them to fit in. Each component and module are connected to internet which is provided by them via WIFI or Ethernet (client-based environment setup).

Component 1 (Hardware)

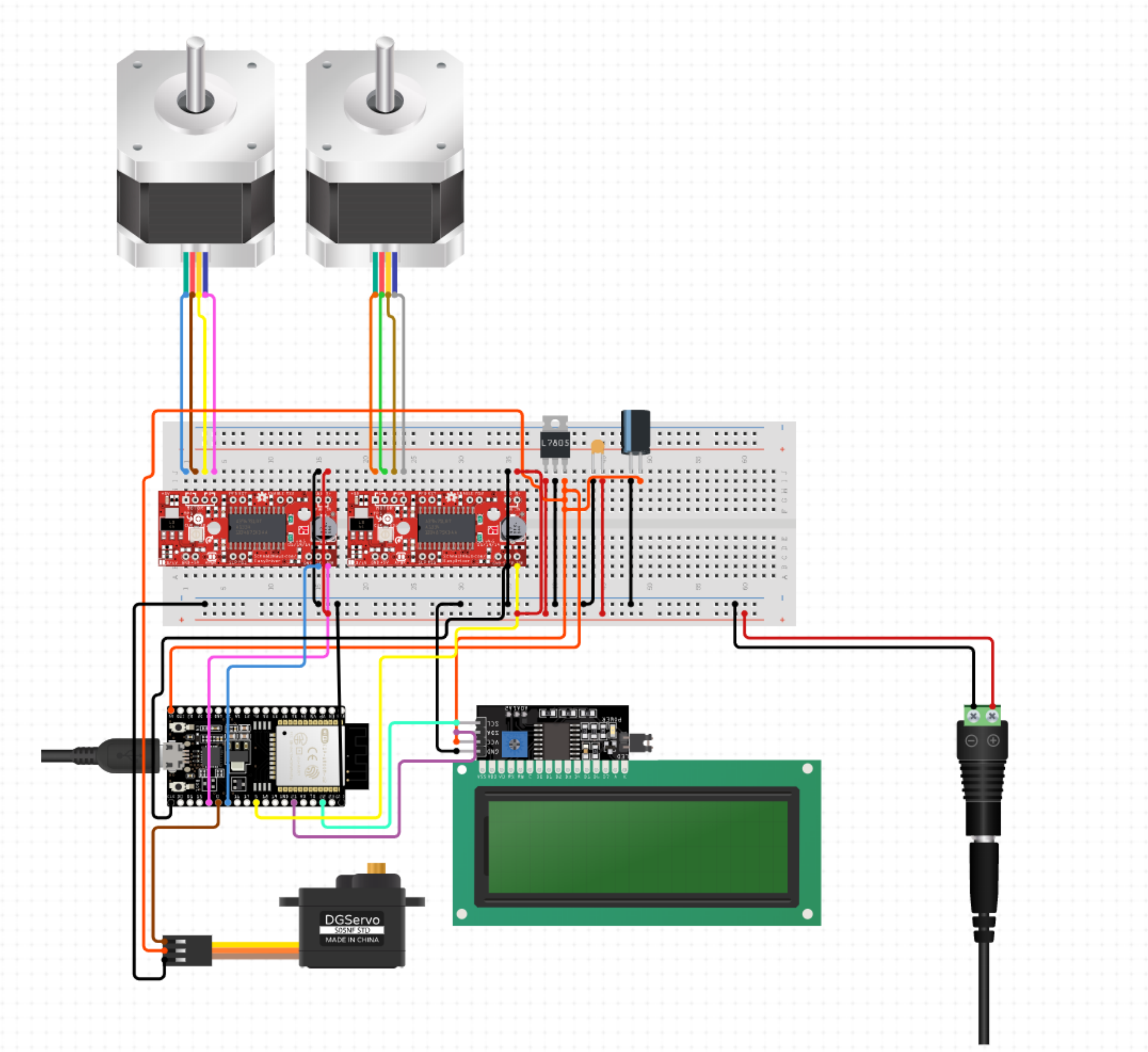
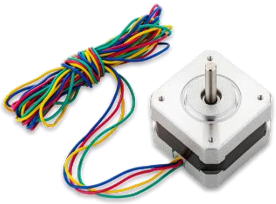


Fig 6. Component Circuit Diagram

1. Node MCU ESP32:

This is the basic building block of the vending machine. ESP32 will work as the brain of the individual vending machine. ESP32 has capability to connect to Bluetooth and WIFI and can be easily connected to the internet. This will help to establish connection of vending machine and the server. This needs to be bought from any e-commerce site or physical store available nearby. This component is available around ₹ 400.

2. Stepper Motor:



This stepper motor is used to rotate the spring mechanism of the vending machine. This needs to be rotated on the instruction of the ESP32. This motor needs to be bought from 3rd party sources as it will prove easier for production. The motor is available on the internet as well as physical stores around ₹ 700.

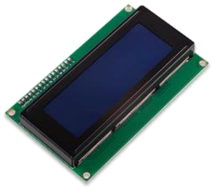
Stepper Motor Driver:

This motor also requires an additional circuit board for the integration purpose with the Node MCU ESP32. This motor cannot be directly connected to the circuit. This is model A4988 and is available on the internet for ₹ 150.

3. Servo Motor:

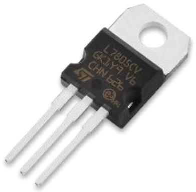
Servo motor is used for the automated control of the maintenance door of the vending machine. This motor will be controlled by the vendor from mobile itself. This motor needs to accept instructions from ESP32 in order to function on demand. This costs ₹ 1499.

4. LCD Display:



This display will be used to display the acknowledgement messages to the user on the vending machine. Also, the prior QR code will be displayed on this display itself. This is a 20x4 I2C display and supports monochrome color. It costs around ₹ 450.

5. Voltage Regulator:



This is a simple voltage regulator IC 7805 used for stabilization of the power flowing through the circuit. This is just a precautionary measure in the complete circuit. It costs a whopping ₹ 10.

6. UVC Lamp:

This UV light will be used for sanitization of the products using UV rays. This UV light will be placed in a vertical manner so that each item on each rack gets adequate light to kill germs. This costs from ₹ 550 to ₹ 7000.

These are some major components in our system. Considering the smaller and intricate components, the detailed list is given below:

i. **Processing Board:** ESP32-DEVKITC with WIFI 802.11bn and Bluetooth 4.2
ii. **LCD Display:** DFR0154 I2C 20X4 Arduino LCD Display Mod
iii. **Servo Motor:** SERVOMOTOR RC 6V GENERIC METAL -- Positional Rotation DC Motor Servomotor, RC 6VDC
iv. **Stepper Motor:** ROB-09238 **Easy Driver Board** – A4988
v. Wall Adapter **Power Supply** - 12VDC 2A
vi. Female **DC Power adapter** - 2.1mm jack to screw terminal block
vii. USB **micro-B Cable** - 6 Foot
viii. Voltage Regulator 5v

ix. Capacitor Ceramic 100nF
x. Electrolytic Capacitor - 1uF/50V
xi. Bread Board
xii. Heat Sink TO-220
xiii. Jumper Wires Pack - M/M
xiv. Jumper Wires Pack - M/F
xv. Male Headers Pack- Break-Away

Component 2 (Software)

The software will be used to control the Vending machine in a remote manner. The application needs to be designed as per the structure given in Fig 4 stated above. The design of the software is primarily designed in React Native.

“React Native is an open-source mobile application framework created by Facebook. It is used to develop applications for Android, iOS, Web Apps and UWP (Universal Windows Platform) by enabling developers to use React along with native platform capabilities.”

This will help the system to be available cross platform without writing any redundant code for multiple platforms. We have used the latest version of React Native to support maximum number of devices out there. The system is designed keeping in mind that all the Transactions will be fluently performed by the user without any difficulties. The payment will be done with the help of Google Pay API. The Google Pay API enables fast, simple checkout within your app, and gives you convenient access to hundreds of millions of cards saved to Google Accounts worldwide.

When a user taps the Google Pay payment button, they see a payment sheet that displays the payment methods saved to their Google Account, as well as optional fields such as a shipping address field. Users can quickly select a payment method, add an optional shipping address, or add new information.

Payment Flow:

The payment flow for the Vending Machine purchase is as follows:

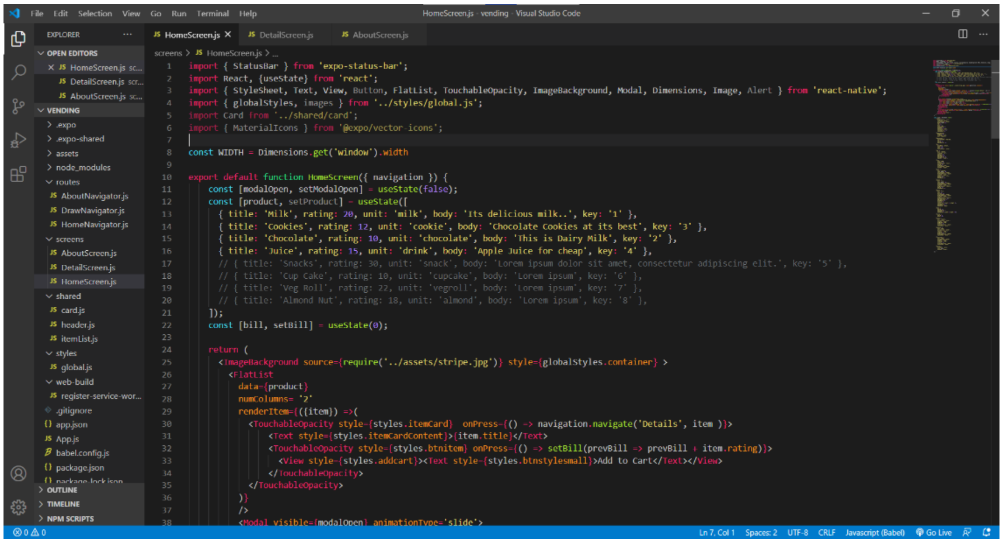
1. The user taps the Google Pay payment button and sees a payment sheet with a list of supported payment methods.
2. The user selects a payment method and Google Pay securely returns a payment token for that method to the app.
3. The application submits the payment token, along with details about the purchase, to its backend.
4. To execute the payment, the backend processes the purchase and sends the payment token to the payment service provider.

The backend i.e. the database server is having the needed components in its primary schema to support the API library.

Coming to the Application Design part, the application is constructed in React Native and the design layouts are as follows:

1. Home Screen

This page of the application is the basic and the most important page of the mobile application. This page contains information of the products which are available to buy from the Vending Machine.



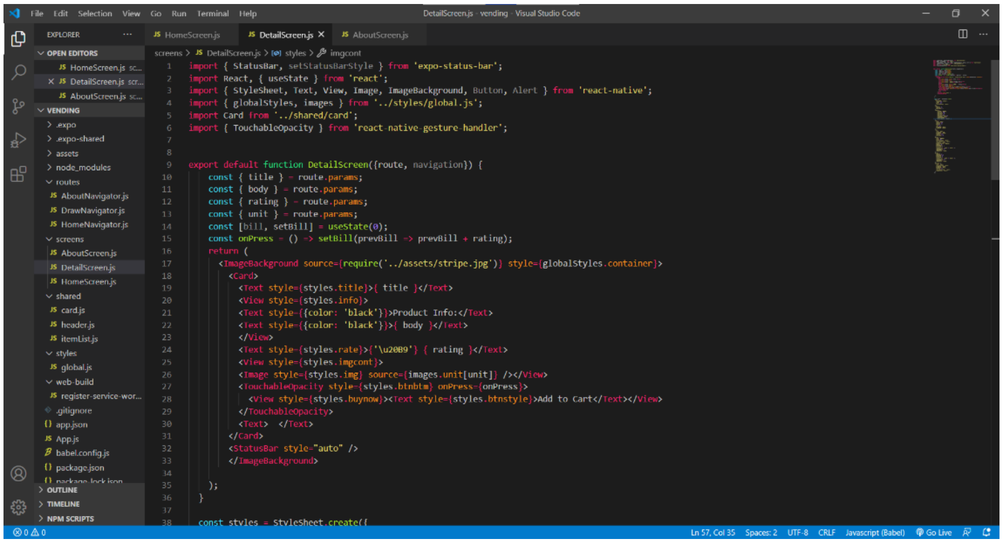
```
import { StatusBar } from 'expo-status-bar';
import React, {useState} from 'react';
import { StyleSheet, Text, View, Button, FlatList, TouchableOpacity, ImageBackground, Modal, Dimensions, Image, Alert } from 'react-native';
import { globalStyles, images } from '../styles/global.js';
import Card from '../shared/card';
import { MaterialIcons } from '@expo/vector-icons';

const WIDTH = Dimensions.get('window').width

export default function HomeScreen({ navigation }) {
    const [modalOpen, setModalOpen] = useState(false);
    const [product, setProduct] = useState([
      { title: 'Milk', rating: 20, unit: 'milk', body: 'Its delicious milk..', key: '1' },
      { title: 'Cookies', rating: 12, unit: 'cookie', body: 'Chocolate Cookies at its best', key: '3' },
      { title: 'Chocolate', rating: 10, unit: 'chocolate', body: 'This is Dairy Milk', key: '2' },
      { title: 'Juice', rating: 15, unit: 'drink', body: 'Apple Juice for cheap', key: '4' },
      // { title: 'Snacks', rating: 30, unit: 'snack', body: 'Lorem ipsum dolor sit amet, consectetur adipiscing elit.', key: '5' },
      // { title: 'Cup Cake', rating: 10, unit: 'cupcake', body: 'Lorem ipsum', key: '6' },
      // { title: 'Veg Roll', rating: 22, unit: 'vegroll', body: 'Lorem ipsum', key: '7' },
      // { title: 'Almond Nut', rating: 18, unit: 'almond', body: 'Lorem ipsum', key: '8' },
    ]);
    const [bill, setBill] = useState(0);

    return (
      <ImageBackground source={require('../assets/stripe.jpg')} style={globalStyles.container} >
        <FlatList
          data={product}
          numColumns= '2'
          renderItem={({item}) =>(
            <TouchableOpacity style={styles.itemCard}  onPress={() => navigation.navigate('Details', item )}>
              <Text style={styles.itemCardContent}>{item.title}</Text>
              <TouchableOpacity style={styles.btnitem} onPress={() => setBill(prevBill => prevBill + item.rating)}>
                <View style={styles.addcart}><Text style={styles.btnstylesmall}>Add to Cart</Text></View>
              </TouchableOpacity>
            </TouchableOpacity>
          )}
          />
          <Modal visible={modalOpen} animationType='slide'>
```

Fig 7. Home Screen Code Snap

This code contains the basic home layout structure in a React component named as HomeScreen. This component is then imported in the DrawerNavigation and StackNavigation screens.

2. Details Screen

This page appears when user clicks on a product to view the details of the product such as the information of the product and the price of the product as well as the “Add to Cart” button to add that product to the cart for buying.



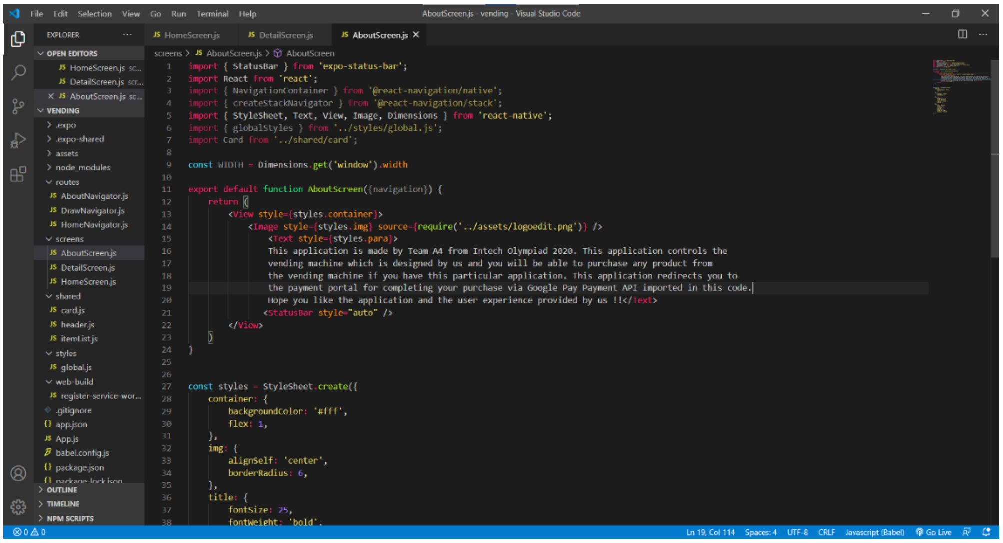
```
import { StatusBar, setStatusBarStyle } from 'expo-status-bar';
import React, { useState } from 'react';
import { StyleSheet, Text, View, Image, ImageBackground, Button, Alert } from 'react-native';
import { globalStyles, images } from '../styles/global.js';
import Card from '../shared/card';
import { TouchableOpacity } from 'react-native-gesture-handler';


export default function DetailScreen({route, navigation}) {
    const { title } = route.params;
    const { body } = route.params;
    const { rating } = route.params;
    const { unit } = route.params;
    const [bill, setBill] = useState(0);
    const onPress = () => setBill(prevBill => prevBill + rating);
    return (
      <ImageBackground source={require('../assets/stripe.jpg')} style={globalStyles.container}>
        <Card>
          <Text style={styles.title}>{ title }</Text>
          <View style={styles.info}>
          <Text style={{color: 'black'}}>Product Info:</Text>
          <Text style={{color: 'black'}}>{ body }</Text>
          </View>
          <Text style={styles.rate}>{'\u20B9'} { rating }</Text>
          <View style={styles.imgcont}>
          <Image style={styles.img} source={images.unit[unit]} /></View>
          <TouchableOpacity style={styles.btnbtm} onPress={onPress}>
            <View style={styles.buynow}><Text style={styles.btnstyle}>Add to Cart</Text></View>
          </TouchableOpacity>
          <Text>  </Text>
        </Card>
        <StatusBar style="auto" />
        </ImageBackground>

    );
  }

  const styles = StyleSheet.create({
```

Fig 8. Details Screen Code Snap

- This is the second code snippet which contains all the detailed information of the product user has selected to be viewed. The data of the product is imported by using props from HomeScreen component. Each attribute of item is stored in an array in the HomeScreen component, this array is then imported here and the elements of the array known as the attributes of the product are displayed on this DetailScreen component.
- This component is then imported in the DrawerNavigation and StackNavigation screens. The DrawerNavigation and the StackNavigation are components of React Native which help the application to generate structurally predefined navigation menus which we don’t have to hard-code every time we need them. The packages of React StackNavigator and React DrawerNavigator needs to be installed before running these codes.

3. About Screen

This page shows the basic information of the team just as an introduction to team and the Vending Machine itself. This page is just for contextual reference for the user about the product he / she is using.



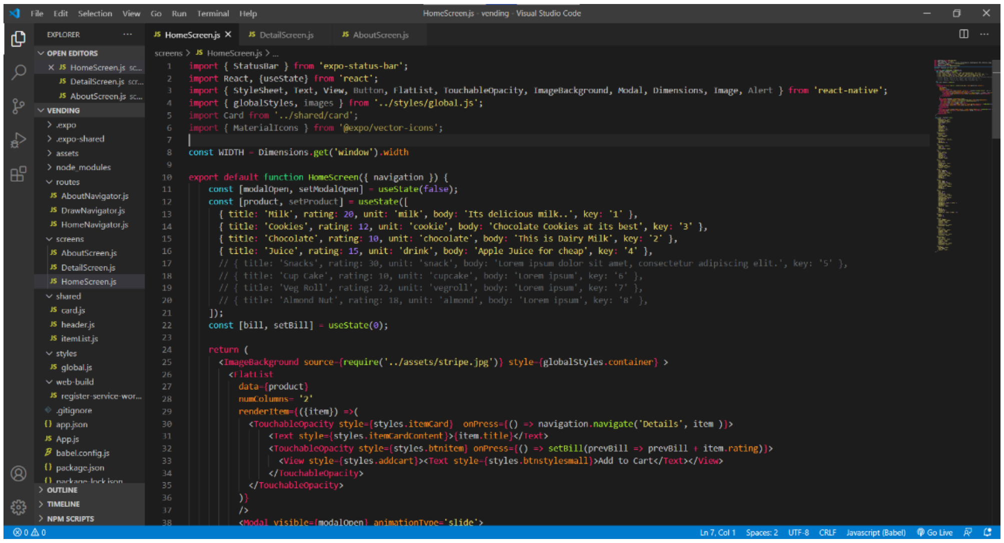
```
import { StatusBar } from 'expo-status-bar';
import React from 'react';
import { NavigationContainer } from '@react-navigation/native';
import { createStackNavigator } from '@react-navigation/stack';
import { StyleSheet, Text, View, Image, Dimensions } from 'react-native';
import { globalStyles } from '../styles/global.js';
import Card from '../shared/card';

const WIDTH = Dimensions.get('window').width

export default function AboutScreen({navigation}) {
    return (
        <View style={styles.container}>
            <Image style={styles.img} source={require('../assets/logoedit.png')} />
                <Text style={styles.para}>
                This application is made by Team A4 from Intech Olympiad 2020. This application controls the
                vending machine which is designed by us and you will be able to purchase any product from
                the vending machine if you have this particular application. This application redirects you to
                the payment portal for completing your purchase via Google Pay Payment API imported in this code.
                Hope you like the application and the user experience provided by us !!</Text>
               <StatusBar style="auto" />
        </View>
    )
}


const styles = StyleSheet.create({
    container: {
        backgroundColor: '#fff',
        flex: 1,
    },
    img: {
        alignSelf: 'center',
        borderRadius: 6,
    },
    title: {
        fontSize: 25,
        fontWeight: 'bold',
```

Fig 9. About Screen Code Snap

- This is the simplest component in the whole code, this AboutScreen component contains only text information and the team logo in an Image container. This component is also imported in the DrawerNavigation screen for fluent access from the HomeScreen itself.

4. Payment Screen

This page appears when the user clicks the Buy Now button on the Home Page itself. This page leads the user to the Payment Integration API connected to the Application. The further payment process will be carried out by the API.



HomeScreen.js - vending - Visual Studio Code
HomeScreen.js
DetailScreen.js
AboutScreen.js
Ln 7, Col 1 Spaces: 2 UTF-8 CRLF Javascript (Babel) Go Live

```
import { StatusBar } from 'expo-status-bar';
import React, {useState} from 'react';
import { StyleSheet, Text, View, Button, FlatList, TouchableOpacity, ImageBackground, Modal, Dimensions, Image, Alert } from 'react-native';
import { globalStyles, images } from '../styles/global.js';
import Card from '../shared/card';
import { MaterialIcons } from '@expo/vector-icons';

const WIDTH = Dimensions.get('window').width

export default function HomeScreen({ navigation }) {
  const [modalOpen, setModalOpen] = useState(false);
  const [product, setProduct] = useState([
    { title: 'Milk', rating: 20, unit: 'milk', body: 'Its delicious milk..', key: '1' },
    { title: 'Cookies', rating: 12, unit: 'cookie', body: 'Chocolate Cookies at its best', key: '3' },
    { title: 'Chocolate', rating: 10, unit: 'chocolate', body: 'This is Dairy Milk', key: '2' },
    { title: 'Juice', rating: 15, unit: 'drink', body: 'Apple Juice for cheap', key: '4' },
    // { title: 'Snacks', rating: 30, unit: 'snack', body: 'Lorem ipsum dolor sit amet, consectetur adipiscing elit.', key: '5' },
    // { title: 'Cup Cake', rating: 10, unit: 'cupcake', body: 'Lorem ipsum', key: '6' },
    // { title: 'Veg Roll', rating: 22, unit: 'vegroll', body: 'Lorem ipsum', key: '7' },
    // { title: 'Almond Nut', rating: 18, unit: 'almond', body: 'Lorem ipsum', key: '8' },
  ]);
  const [bill, setBill] = useState(0);

  return (
    <ImageBackground source={require('../assets/stripe.jpg')} style={globalStyles.container} >
      <FlatList
        data={product}
        numColumns= '2'
        renderItem={({item}) =>(
          <TouchableOpacity style={styles.itemCard}  onPress={() => navigation.navigate('Details', item )}>
              <Text style={styles.itemCardContent}>{item.title}</Text>
              <TouchableOpacity style={styles.btnitem} onPress={() => setBill(prevBill => prevBill + item.rating)}>
                <View style={styles.addcart}><Text style={styles.btnstylesmall}>Add to Cart</Text></View>
              </TouchableOpacity>
          </TouchableOpacity>
        )}
        />
        <Modal visible={modalOpen} animationType='slide'>
```

Fig 10. Payment Screen Code Snap

- This part of code is included on the HomeScreen itself and does not have a new JavaScript code file. The Payment is added as a Modal in React Native. A Modal is a small pop-up code which is triggered on the click of a button or on changebyElement triggers.
- The payment portal is attached to this part of the code and the amount to be paid is fetched from the array called bill which is generated when the user clicks Add to Cart Button. And hence the payment is further handled by the payment API.

The code for all the screens and the overall Android Application is attached in the References Section. Also, the snapshots of the Layouts of the screens are attached in the POC section of this report document.

Component 3 (Database)

This component is the mucilage of the complete system as depicted in Fig 2. This database will be installed on the Server on the cloud. This database server will interact with the IoT devices mounted in the Vending Machine as well as the mobile application which the user has running on the smartphone. The server will be Google Firebase which needs to be purchased for development purpose. This server has important information such as Admin Credentials, Vending machine states and the information of the products that are available in the vending machine at a certain point of time. The schema of the database is:

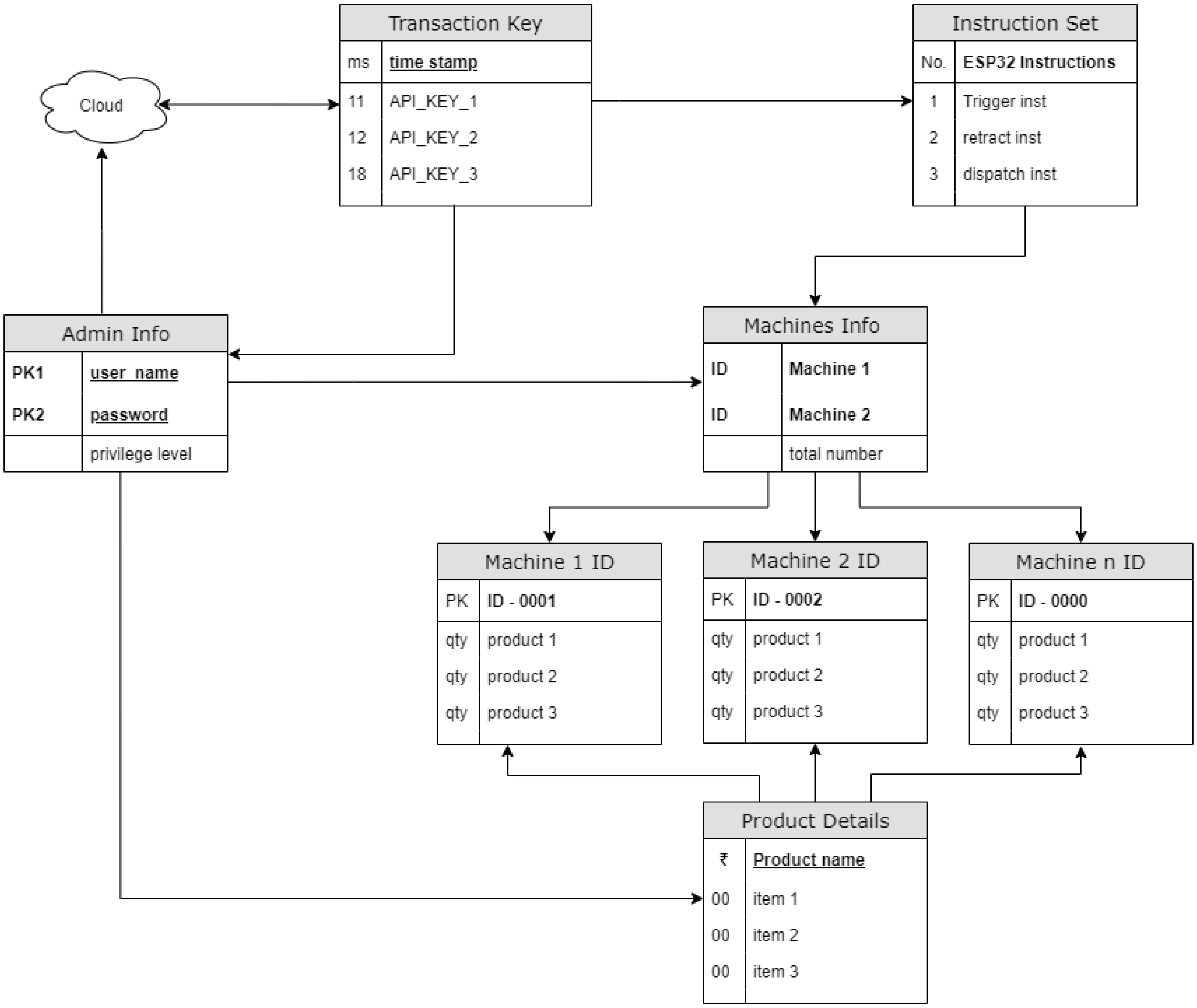


Fig 10. Database Schema

Integration of Components

Standard Vending Mechanism
IoT Integration with Remote Access
Revolutionary Product

Fig 11. Integration of Components

The main 3 components of the system are:

i. Hardware
ii. Software
iii. Database

These 3 components are interconnected to each other using internet. The hardware component contains smart IoT circuit like Node MCU ESP32 which has functionality of Bluetooth as well as WIFI which enables the hardware to communicate through the internet. The central database will remain the most important mucilage of the system as the system would not be able to obtain any instruction without a command from Cloud Database. The database is then receiving commands from the mobile interface used by the user who has started this instance. This structure can be referred in Fig 2. Let's have a look at an example:

Integration in Action:

1. A user goes to the Vending Machine and wants to buy a product. He has the Vending Machine application in his mobile and the end-users is able to see a QR code on the vending machine screen.

2. The user opens the application on his / her mobile device and scans the QR code visible on the machine's screen. (Once the application is connected to the database, it fetches the information of the machine user is standing in front of and the details which are to be shown to the user).

3. The user can now see the Home Screen which displays the products which are available in the vending machine in front of him / her.(when the user selects the products and adds them to cart, an array is generated in the code which stores all the items which are selected by the user to buy).

4. The user selects a product and then clicks on the “Buy Now” button. This takes the user to the payment portal and the user completes the payment process by entering credentials, OTP, etc. The payment API, after the completion of Successful Payment will send a unique TOKEN to the database which will indicate that the payment is successful.

5. After the Token is received, the database will then trigger the Node MCU present in the Vending machine (all the details such as item purchased and number of items to dispatch will be contained in the instance the user created when he / she scanned the QR code) resulting into the rotation of the spring mechanism and eventually the Dispatch of the requested product by the user.

Supporting POCs

POC 1

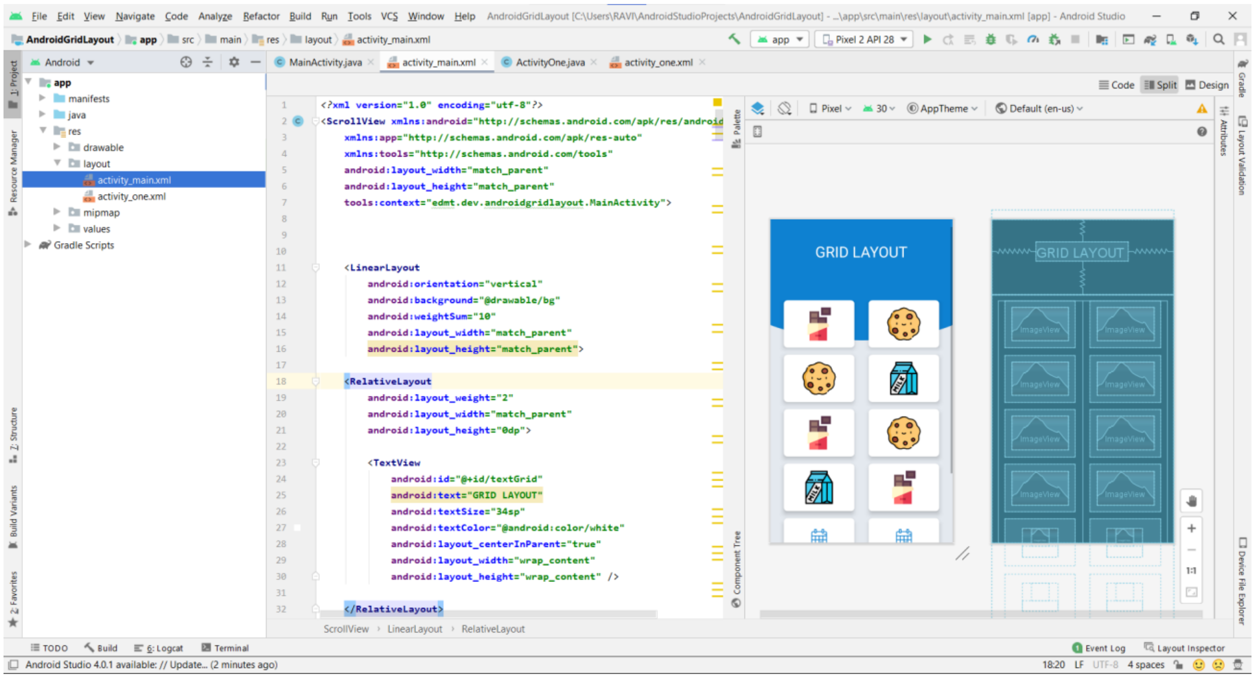
The POC 1 was the first iteration of the project designing phases. In this iteration the major part was the research of the pre-existing system and the differential factors which could be added to the system to make the vending machine more efficient and more useful than before. The team worked on the Research and Development (R&D) for many aspects of the vending machine and the first prototype was generated. The research is as follows:

Vending Machine Mechanism: Thorough research was done on the functional machinery of the vending machine. It highlighted that the vending machines available in the market are individual units and do not integrate with any other remote servers or such for general communication. Also, the vending machines available on date are not using any IoT components embedded resulting that the machines were not able to store or record the data in any sort. Team now focused on how to integrate the traditional vending machine with IoT devices and make it more productive and efficient. So, technologies were discovered to interconnect the vending machine to the internet.

Cashless Transactions in Vending Machine: As the team did research on the pre-existing vending machines, it was observed that many of the vending machines were using older payment methods such as coin system, token system, prepaid subscriptions, etc. It was also observed that some vending machines used RFID (Radio Frequency Identification) cards to pay for the product. This was quite well done but it still is away from the ultimate goal of cashless payments. Following the Digital India or promotion of Cashless transactions by the Government of India, we did a thorough research on how we can make a cashless transaction possible. The RFID option was still available but it was a bit riskier in this COVID19 situation. Hence, we went ahead with complete cashless Online Transactions using payment API’s by the global providers available in the market.

Sanitization of the Product & Touchless Perspective: Considering the COVID19 situation in our surroundings, it should be kept in mind that even when we all go out freely, we still need to follow the Physical Distancing Protocols. So, if the vending machine is produced and gets out there in the open, it should be considered that the machine as well as the products in the machine need to be sanitized from time to time for a healthier experience with the buying of the products. Also, it is now common now-a-days that nothing out in the open should be touched unnecessarily. The vending machine is a public device which can be used by anyone without any discrimination. Hence it would be extremely dangerous if each user / buyer touched the machine for purchasing anything from the machine. So, our team came up with a great solution to avoid any contact of any person with the machine whilst buying from the machine.

Mobile Application: For making the complete user experience touchless and automated. We need to create an android application which remotely controls the vending machine you are standing in front of. For this purpose, a small research was done on mobile application development and cross platform connection and control of IoT devices. Accordingly, we started the work on the Android Application on the most common platforms such as Android Studio and Java Programming Language. After a certain point in time, we observed that we were falling short of hardware resources to run the needed software and emulate the application in our computers. The Layout was generated and primary connection was integrated with the local database. A snapshot of the project environment is attached below. The primary prototype was successfully obtained which is as follows:



```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="edmt.dev.androidgridlayout.MainActivity">



    <LinearLayout
        android:orientation="vertical"
        android:background="@drawable/bg"
        android:weightSum="10"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    <RelativeLayout
        android:layout_weight="2"
        android:layout_width="match_parent"
        android:layout_height="0dp">

        <TextView
            android:id="@+id/textGrid"
            android:text="GRID LAYOUT"
            android:textSize="34sp"
            android:textColor="@android:color/white"
            android:layout_centerInParent="true"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

    </RelativeLayout>
```

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
AndroidGridLayout [C:\Users\RAVI\AndroidStudioProjects\AndroidGridLayout] - ...\app\src\main\res\layout\activity_main.xml [app] - Android Studio
AndroidGridLayout › app › src › main › res › layout › activity_main.xml
app
Pixel 2 API 28
Android
MainActivity.java
activity_main.xml
ActivityOne.java
activity_one.xml
app
manifests
java
res
drawable
layout
activity_main.xml
activity_one.xml
mipmap
values
Gradle Scripts
Code
Split
Design
Pixel
30
AppTheme
Default (en-us)
GRID LAYOUT
ImageView
ScrollView › LinearLayout › RelativeLayout
TODO
Build
6: Logcat
Terminal
Event Log
Layout Inspector
Android Studio 4.0.1 available: // Update... (2 minutes ago)
18:20 LF UTF-8 4 spaces

Fig 12. Android Studio Development Environment

POC 2

In this phase of development, the work done before was taken to another level. The functionality of the vending machine was incremented by reaching some milestone decisions and team started on the planned proposal immediately. Some highlights of the POC 2 are:

IoT implementation: We introduced IoT in Vending machines which will affect the overall working of vending machine. IoT will not only store data in a more secured manner but also will provide daily statistics and report of the happenings at the machine to the vendor. This will keep vendor well acquainted with the system all the time. Also, IoT will be implemented for using physical triggers which will be activated whenever any person tries to harm the machine and the vendor will get immediate alert and footage of the event. Usage of IoT devices will ensure that the maintenance will be done in shortest period of time for any physical damage of parts as they are available locally. IoT sensors will be installed throughout the device to keep checking internal system of the vending machine, if any internal part misbehaves or stops working, machine itself will send appropriate information to the vendor so that it could be repaired with minimum time loss.

Hardware Implementation: The Vending machine also got physically designed in this phase. The circuit diagram of the processing circuit was developed by the team and the benchmarking of various alternative components was done. Many aspects of the system like the main processor board was selected based on the Benchmarked results. At the end, Arduino – Noe MCU ESP32 was selected based on its advantages over Raspberry Pi and Arduino Uno and also the price of the component. Similarly, other components listed in the Component 1 section above were chosen after a benchmarking of a bunch of criteria on each component chosen for this system. The circuit diagram shown in Fig 6 was designed in this phase.

Android Application: After a tough brainstorming and a mutual research on latest technology, our team decided to stop working on the Android Programming using Java Programming Language and team came up with a more elaborate solution on the same. According to the research done, we found out that the latest and more viable programming platform for the application was React Native. This enabled us to write code in live running environment and avoided the hardware restrictions put on us by Android Studio for emulating the device. This enabled us to write a single code and then compile it on different platforms to get Native Application for Android as well as Apple ecosystem in the easiest manner. We started to learn the React Native programming Framework and created a demo application for the Vending Machine. These are some snapshots of the demo application:

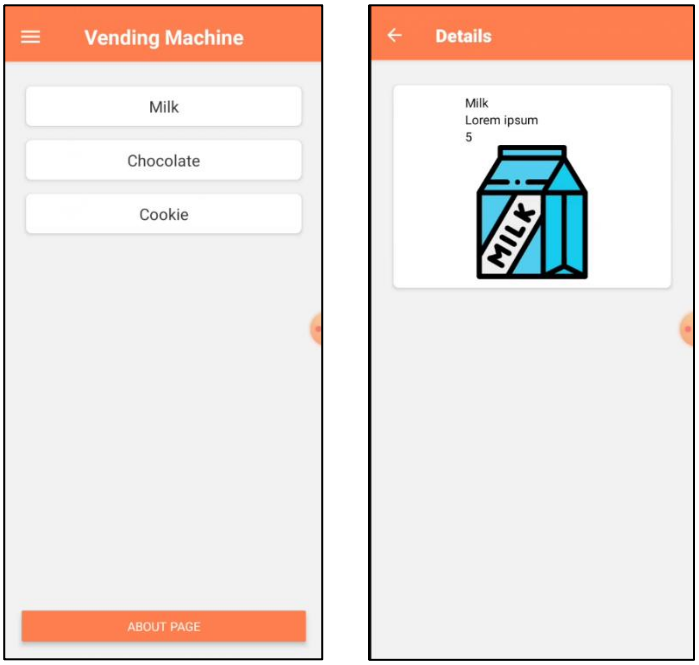


Fig 13. Snapshots of Demo Application - 1

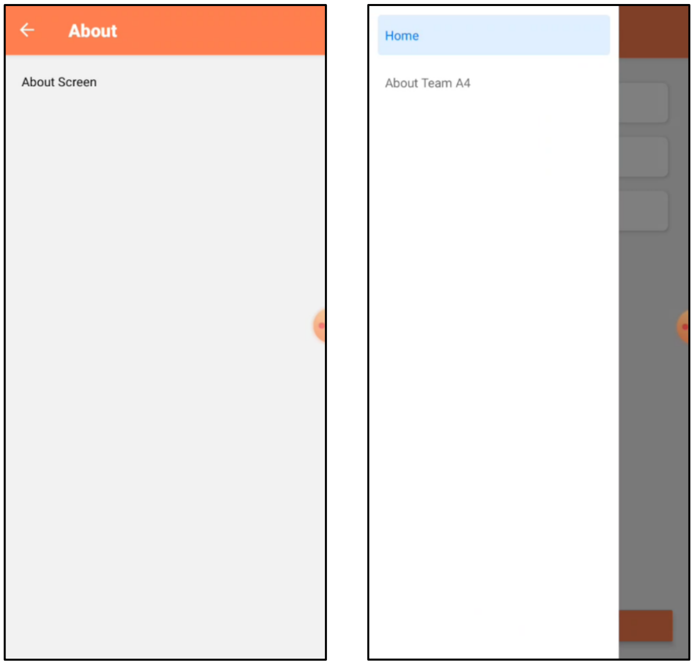


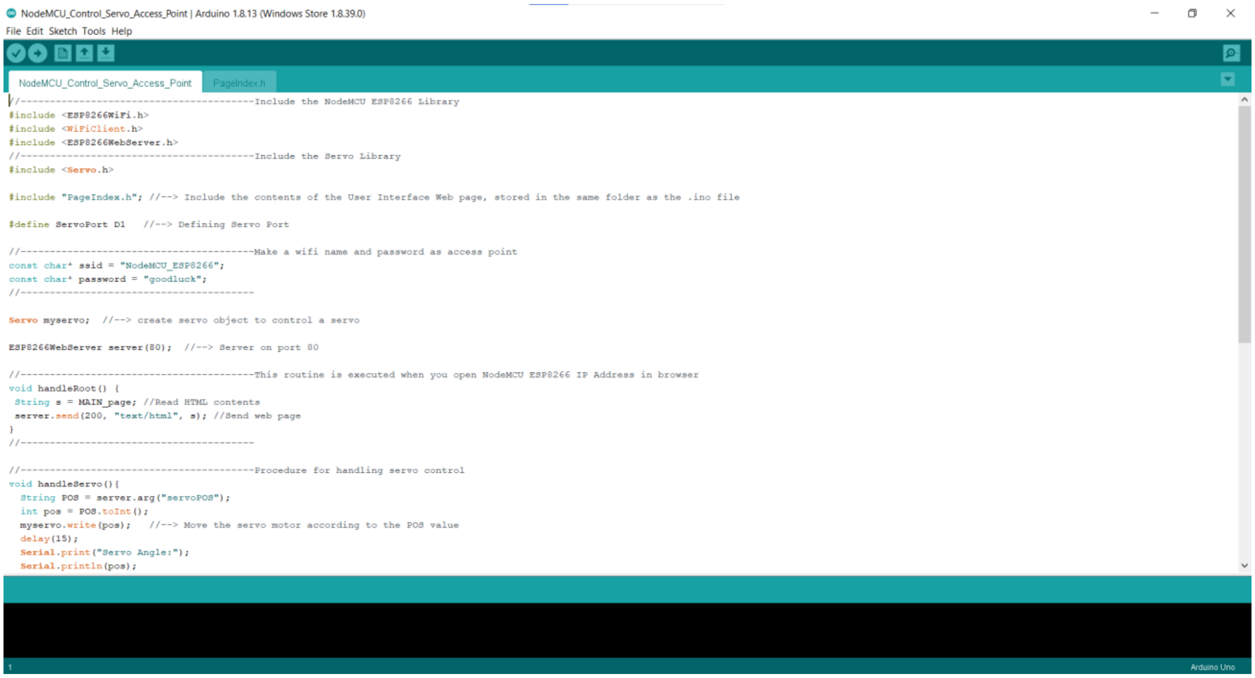
Fig 14. Snapshots of demo Application - 2

POC 3

After passing through the two POC's it now became quite more interesting for working on this project. Now each and every team member was a very huge part of the system built till date. The further development in this phase is as follows:

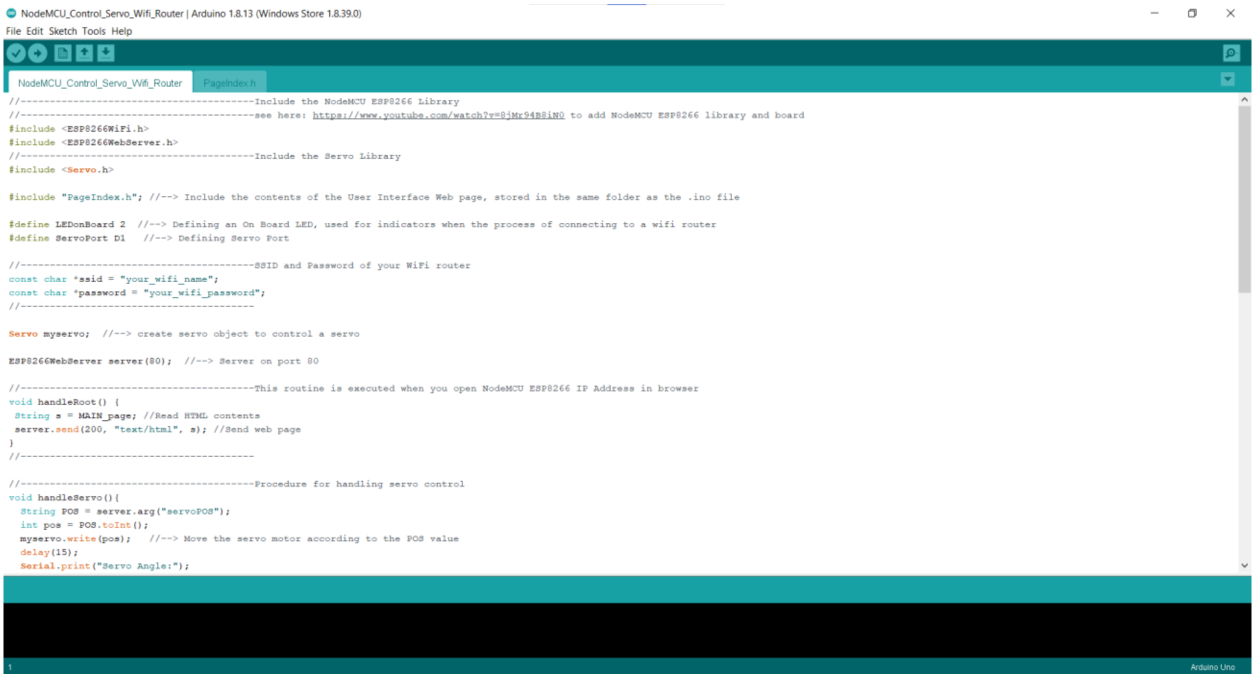
Android Application: The team now learnt the React Native Programming Framework and was now able to fluently create awesome designs for the application to be coded. Then we started integrating each and every component of the system into one single block of functionality. The layouts of all the pages was designed and are as follows:

Mechanical Machine: The concurrent development of the physical machine was also considered. The design of the mechanical components of the vending machine was also polished and finalized. During this process, the code for the IoT components like the servo motor, the stepper motor and other components was done. This code will be attached in the References section of this report.


NodeMCU_Control_Servo_Access_Point | Arduino 1.8.13 (Windows Store 1.8.39.0)
File Edit Sketch Tools Help
NodeMCU_Control_Servo_Access_Point
PageIndex.h
1
Arduino Uno

```
//----------------------------------------Include the NodeMCU ESP8266 Library
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ESP8266WebServer.h>
//----------------------------------------Include the Servo Library
#include <Servo.h>

#include "PageIndex.h"; //--> Include the contents of the User Interface Web page, stored in the same folder as the .ino file

#define ServoPort D1   //--> Defining Servo Port

//----------------------------------------Make a wifi name and password as access point
const char* ssid = "NodeMCU_ESP8266";
const char* password = "goodluck";
//----------------------------------------

Servo myservo;  //--> create servo object to control a servo

ESP8266WebServer server(80);  //--> Server on port 80

//----------------------------------------This routine is executed when you open NodeMCU ESP8266 IP Address in browser
void handleRoot() {
 String s = MAIN_page; //Read HTML contents
 server.send(200, "text/html", s); //Send web page
}
//----------------------------------------

//----------------------------------------Procedure for handling servo control
void handleServo(){
  String POS = server.arg("servoPOS");
  int pos = POS.toInt();
  myservo.write(pos);   //--> Move the servo motor according to the POS value
  delay(15);
  Serial.print("Servo Angle:");
  Serial.println(pos);
```

Fig 15. Servo Controller Access Point Code


NodeMCU_Control_Servo_Wifi_Router | Arduino 1.8.13 (Windows Store 1.8.39.0)
File Edit Sketch Tools Help
NodeMCU_Control_Servo_Wifi_Router
PageIndex.h
1
Arduino Uno

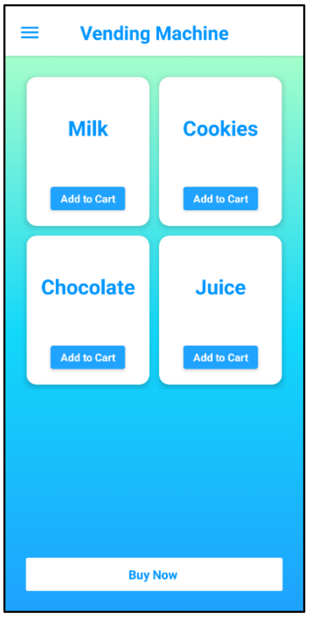
```
//----------------------------------------Include the NodeMCU ESP8266 Library
//----------------------------------------see here: https://www.youtube.com/watch?v=8jMr94B8iN0 to add NodeMCU ESP8266 library and board
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
//----------------------------------------Include the Servo Library
#include <Servo.h>

#include "PageIndex.h"; //--> Include the contents of the User Interface Web page, stored in the same folder as the .ino file

#define LEDonBoard 2  //--> Defining an On Board LED, used for indicators when the process of connecting to a wifi router
#define ServoPort D1   //--> Defining Servo Port

//----------------------------------------SSID and Password of your WiFi router
const char *ssid = "your_wifi_name";
const char *password = "your_wifi_password";
//----------------------------------------

Servo myservo;  //--> create servo object to control a servo

ESP8266WebServer server(80);  //--> Server on port 80

//----------------------------------------This routine is executed when you open NodeMCU ESP8266 IP Address in browser
void handleRoot() {
 String s = MAIN_page; //Read HTML contents
 server.send(200, "text/html", s); //Send web page
}
//----------------------------------------

//----------------------------------------Procedure for handling servo control
void handleServo(){
  String POS = server.arg("servoPOS");
  int pos = POS.toInt();
  myservo.write(pos);   //--> Move the servo motor according to the POS value
  delay(15);
  Serial.print("Servo Angle:");
```

Fig 16. Servo Controller WIFI router Code

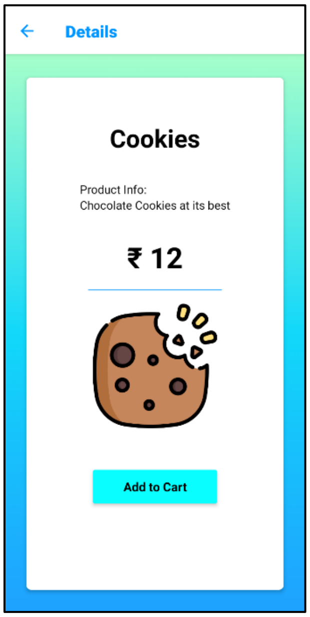
Database Schema: After scraping the Android Studio Project, the database generated for the local host was useless. So, a brand-new Database schema was designed now keeping in mind all the additional functionalities provided by React Native Framework.

The database schema includes all the components of the system which are designed till now. The schema is attached to the Component 3 section of this report. Fig 10.

Fig 17. Snapshots of Final Application

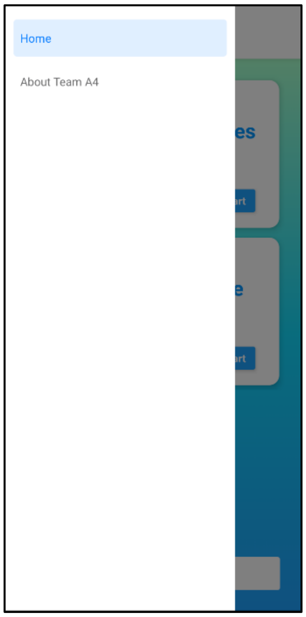


i.Home Screen

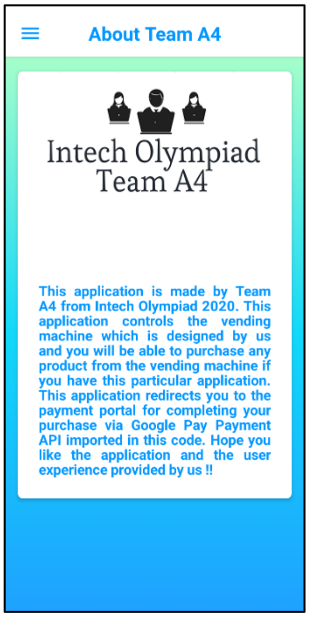


Product Info:
Chocolate Cookies at its best

ii Details Screen

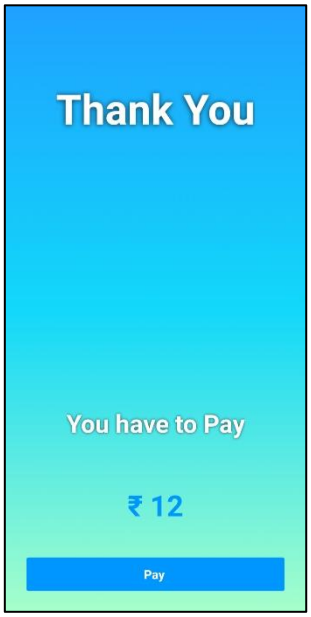


iii Navigation Drawer



This application is made by Team A4 from Intech Olympiad 2020. This application controls the vending machine which is designed by us and you will be able to purchase any product from the vending machine if you have this particular application. This application redirects you to the payment portal for completing your purchase via Google Pay Payment API imported in this code. Hope you like the application and the user experience provided by us !!

iv About Us Page



V Payment Confirmation Page

Quality Assurance Plan

Quality of the output will be monitored by the developers as well as alpha testers to ensure the process is delivering the prominent features which are mapped to be provided to the users of the system. All the phases of the system will also be checked with a wide variety of test cases to ensure there is no flaw in the system throughout. This project will be done in a complete sense of taking technology a step ahead towards helping people to fight against the pandemic and be safe while they are in social environment.

React Native Testing Environment:

Unit tests cover the smallest parts of code, like individual functions or classes. When the object being tested has any dependencies, you'll often need to mock them out, as described:

The great thing about unit tests is that they are quick to write and run. Therefore, as you work, you get fast feedback about whether your tests are passing. Jest even has an option to continuously run tests that are related to code.

Sometimes, when tested objects have external dependencies, you'll want to "mock them out." "Mocking" is when we replace some dependency of your code with your own implementation. Generally, using real objects in your tests is better than using mocks but there are situations where this is not possible.

For example:

Imagine you're writing an app that shows the current weather in your city and you're using some external service or other dependency that provides you with the weather information. If the service tells you that it's raining, you want to show an image with a rainy cloud. You don't want to call that service in your tests, because:

- It could make the tests slow and unstable (because of the network requests involved)
- The service may return different data every time you run the test
- Third party services can go offline when you really need to run tests!

Therefore, you can provide a mock implementation of the service, effectively replacing thousands of lines of code and some internet-connected thermometers!

Payment Test Environment:

Our test environment doesn't return live chargeable tokens in the PaymentData response, but the test environment allows us to test elements of your purchase:

- Confirmation pages
- Receipts
- Billing address (optional)
- Billing phone number (optional)
- Shipping address (optional)
- Email address (optional)

An Unrecognized App error appears on the Google Pay payment sheet until your app has production access. The value of the environment parameter from WalletOptions indicates whether the server runs in a production or in a test environment. The environment value can be one of the following:

WalletConstants.ENVIRONMENT_PRODUCTION
WalletConstants.ENVIRONMENT_TEST
For testing and development, use WalletConstants.ENVIRONMENT_TEST.

The assurance of the qualitative frontier of the system needs pillars to be strengthened on, some of the pillars or milestones we have covered are:

- ✓ The machine will be durable enough. It would not easily wear over time and would not demand frequent maintenance and elements replacement.
- ✓ The product will be colourfast and will not fade even after multiple times of washings, although it will soften if faced by harsh conditions.
- ✓ This product is becoming a must for the domestic and international market due to the COVID19 situation and will get a huge rise in demand.
- ✓ Payment system: In local currency or international currency through Credit card /online pay.
- ✓ Machine states (Material level alert) will be provided by the data collected by IoT devices on board.
- ✓ Data checking (sales volume), Dispensing controlling, Machine settings, Intelligent operation, Function to send mobile phone alert, etc. will be extremely helpful to the vending authorities making a more immersive experience.
- ✓ The machine inside can have more functions added in time being like addition of weather sensors or water detection mechanism on the exterior of the machine making it more intelligent and adds condition-based functionalities.
- ✓ Special function that makes sanitizing process automatic and easy (refer Component 1 section of the document).
- ✓ No need to disassemble the machine to clean the throat block unit because of motored access to the component door via the mobile itself.

Resources / Components Dependencies

The skills required for this solution to be implemented are:

1. Programming skills
    a. React Native
    b. Arduino Programming
    c. Embedded C Programming
2. Hardware Connectivity Mounting Skills

The above-mentioned skills are required for implementing this solution. The number of skill-set required is very less because the Hardware components in the system are not produced by us, they are bought from dealers itself. The components are complicated and costly to manufacture ourselves and hence are needed to be ordered from 3rd party provider. Only programming skills need to be polished to complete this solution as lot code needs to be done for the client-server architecture to be up and running.

Post Production Dependencies:

Once the vending machine is produced and established in real life environment, there are some factors which need to be considered as pre-establishment dependencies which will be required for the system to be up and running. The dependencies are:

- WIFI or Ethernet Connectivity in the surrounding providing the vending machine access to the internet.
- The user as well as the vendor should have the Vending Machine Mobile Application installed on their smartphones on their mobile devices to access the vending machine (to buy or to control the parameters of the machine).
- The mobile device used by the vendor or end-user to access the vending machine should have an active internet connection for the instructions to be fetched by server and the process to initiate.
- A centralized database needs to be set-up at the destined location which will store all the data collected by the vending machines connected to that network.

By providing the above-mentioned conditions, the solution project will be good to go and won’t have any issues till any parameter like internet connectivity fluctuates which will cause interruptions in the workflow.

MVP Implementation Cost and Time Estimation (Optional)

According to the research done by us on prior vending machines available in any form or other, the cost ranges from 1 Lakh to 4.5 Lakh Indian Rupees for the commercial vending machine which is used for general packet product dispatchment from the machine.
Current Cost of Vending Machine: ₹ 1,00,000 to ₹ 4,50,000.

Estimated Cost of our Project: ₹ 40,000 to ₹ 50,000.
Seems a lot less, doesn't it? Let's explore in detail…

The estimated cost of the project seems less due to the selection of components and the integration of the system in a way that each and every component in the system, even if it is bought from a 3rd party source, will result into drastically less overall cost of the final product. The mass production of the solution, if planned, will also deduct the price even more further resulting into more margins for the manufacturing source. The estimation of the costs are as follows:

Module or Component of System	Cost
The machine body with primary spring mechanism	10 - 12,000 ₹
Technical Components like display, circuitry, etc	5 – 7,000 ₹
Mechanical Components for any kind of protection, prevention, safety and security of machine	~10,000 ₹
Setup of Cloud server / online database services	5,000 – 7,000 ₹
Total Primary Cost	36,000 ₹
+ manufacturing, labour, software, etc	~10,000 ₹

Listing in detail for Technical components (circuitry):

Name of the Component	Cost
Node MCU ESP32-DEVKITC with WIFI 802.11bn	800 ₹
LCD Display DRF0154 I2C 20x4	1,763 ₹
Servo Motor Generic Metal Gear	819 ₹
Stepper Motor: ROB-09238 Driver Board - ROB-12779	2,250 ₹
Other minor components like wires, heatsink and plugs	200 - 250 ₹

NOTE: The cost of each and every component stated above is just an estimate and prices of each product can fluctuate time to time resulting into fluctuation of the estimated cost as well.

Team Member Contribution Details

Name of Team Member	Role	Activities
Ravivanshikumar Sangpal	Leader, Researcher/POC implementer/Documenting	Programming of Mobile Application, Research on architectural inter-connection, Selection of components, Distribution of work amongst team, Documentation, Overall co-ordination.
Prasad Gojare	Programmer, Documenting	Programming of Mobile Application, Selection of mechanical components, research on Android and documentation.
Yash Panse	Researcher, Documenting	Research on Machine Design, component research and

		documentation of various components.
Kshitija Pachpute	Researcher, Documenting	Research on React Native, logical components of the system and layout designer for Mobile Application.
Sunil Mane	Researcher, POC Implementor	Research and study of various electrical circuitry, overall vending machine study and materialization decisions.

References

<Please mention here list of all references that you have used while working in the project- If possible with the internet link to the same.>

React Native:
https://reactnative.dev/docs/getting-started
https://reactnavigation.org/docs/getting-started#!
https://reactnative.dev/docs/components-and-apis

All the coding files, folders and documents including patent files are present on this Google Drive Link:

Google Drive Folder Link

Intellectual Property Note

We Team A4 hereby declare that we are not violating any third-party intellectual property rights while producing this document or while producing any document supporting this document. Also, we declare that no third-party copyright was violated while producing this document or while producing any documents supporting this document.

We would like to agree that any further implementation of this Project will be happily be done by us. We have done this design in very serious and immense manner and we promise to do so if any other chance is provided by the authorities to work on this project any further.