Project Report

1. INTRODUCTION:

1.1. Project Overview

The problem of inaccessible diet plans in the palm of your hand affects fitness enthusiasts, individuals who follow diets, and people who cannot go to the gym or follow a costly diet. If fixed, many more people need not rely on gyms for fitness and dieting. Even though diet plans can be constrained by culture and religion of different people in food products like meat, easy access to nutritional information would at-least spread awareness amongst people. Moreover, one of the major issues for people is the easy access to this information.

1.2. Purpose:

The implementation of this app would revolutionize the fitness and health industry and enable more people to be healthy and fit. Stakeholders such as investors and developers will receive recognition and proft for helping raise community awareness about health and users will become healthier.

2. LITERATURE SURVEY

2.1. Existing problem

Hossain et. al[1]. proposes a deep learning model for fruit classification as it is an important task in many industrial applications. A fruit classification system may be used to help a supermarket cashier identify the fruit species and prices. It may also be used to help people decide whether specific fruit species meet their dietary requirements. Hossain et. al. proposes an efficient framework for fruit classification using deep learning. The framework is based on two different deep learning architectures. The first is a proposed light model of six convolutional neural network layers, whereas the second is a fine-tuned visual geometry group-16 pre-trained deep learning model. Two color image datasets, one of which is publicly available, are used to evaluate the proposed framework. The first dataset (dataset 1) consists of clear fruit images, whereas the second dataset (dataset 2) contains fruit images that are challenging to classify. Classification accuracy of 99.49% and 99.75% were achieved on dataset 1 for the first and second models, respectively. On dataset 2, the first and second models obtained accuracy of 85.43% and 96.75%, respectively.

Chithra PL et al[2]. have proposed a new method for classifying fruits using image processing. The data set contains 70 apple images and 70 banana images for training and 25 images of apples and 25 images of bananas for testing. RGB images were first converted into HSI images. Then by using Otsu's thresholding method, the region of interest was segmented by taking into account only the HUE component image of the HSI image. Later, after background subtraction, a total of 36 statistical and texture features were extracted with the help of the coefficients obtained by applying wavelet transformation on the segmented image using Haar filter. The extracted features were given as inputs to a SVM classifier to classify the test images as apples and bananas. As KNN classification method did not give 100% accuracy while classification SVM classification method was used. 140 sample images of apples and bananas were used for training and 25 images of bananas and 25 images of apples were used for testing the proposed algorithm. The proposed algorithm gave a 100% accuracy rate.

Ormsbee et al[3]. tries to optimize pre-exercise nutritional strategies for endurance performance in his paper. They found out that timing and composition of the pre-exercise meal is a significant consideration for optimizing metabolism and subsequent endurance performance. Consuming a CHO-rich meal in the hours prior to endurance exercise had improved performance. Performance was also seen to be improved by ingesting CHO within 60 min of exercise. High fat meals may enhance fat oxidation during subsequent exercise, although the performance effects are unclear, most studies report that there is no benefit or decrement *versus* a CHO meal. Finally, caffeine and beetroot juice (dietary nitrates) appeared to enhance performance, although these effects may be modulated by genetic factors and/or training status.

Convolutional neural networks can automatically extract features by directly processing original images, which has attracted wide interest from researchers in fruit classification terms. However, it is difficult to obtain more accurate identification due to the complexity of class similarity. VGG16 has been used to recognize different types of fruit images. Next, the fruit data set which includes 6 classes also created for network model training and evaluation performance. Images of a group of fruits were collected and a deep convolutional neural network was built to identify six types of fruits. Indicating the feasibility of this model, the ratio reached 100%. Inclusive the approach to training real learning models on large, publicly available image data sets offers a clear path toward easy fruit classification. In this paper, Abu-Jamie et. al[4]. proposes a machine learning based approach for classifying and identifying 6 different fruits with a dataset that contains 2677 images

Mureşan et. al[5]. introduces a new, high-quality, dataset of images containing fruits. They also present the results of some numerical experiments for training a neural network to detect fruits. This concludes that we have to design an interface which can manage multiple classes of fruit data and identify nutrient values of each fruit respectively.

2.2.References

[1] Hossain, M. Shamim, Muneer Al-Hammadi, and Ghulam Muhammad. "Automatic fruit classification using deep learning for industrial applications." *IEEE transactions on industrial informatics* 15.2 (2018): 1027-1034.

[2] Chithra PL, Henila M. Fruits classification using image processing techniques. International Journal of Computer Sciences and Engineering. 2019 Mar;7(5).

[3] Ormsbee, Michael J., Christopher W. Bach, and Daniel A. Baur. "Pre-exercise nutrition: the role of macro nutrients, modified starches and supplements on metabolism and endurance performance." *Nutrients* 6.5 (2014): 1782-1808.

[4] Abu-Jamie, Tanseem N., et al. "Six Fruits Classification Using Deep Learning." (2022).

[5] Mureşan, Horea, and Mihai Oltean. "Fruit recognition from images using deep learning." *arXiv preprint arXiv:1712.00580* (2017).

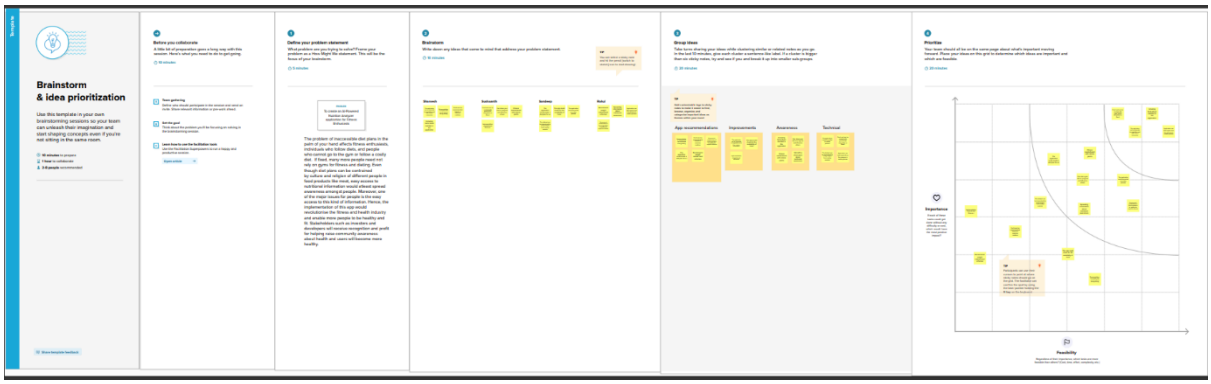
2.3.Problem Statement Definition

To create an AI-Powered Nutrition Analyzer application for Fitness Enthusiasts

3. IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas

3.2. Ideation & Brainstorming



3.3.Proposed Solution:

S.No	Parameter	Description
1	Problem Statement(Problem Solved)	To create an AI-Powered Nutrition Analyzer application for Fitness Enthusiasts
2	Idea/Solution description	The problem of inaccessible diet plans in the palm of your hand affects fitness enthusiasts, individuals who follow diets, and people who cannot go to the gym or follow a costly diet. If fixed, many more people need not rely on gyms for fitness and dieting. Even though diet plans can be constrained by the culture and religion of different people in food products like meat, easy access to nutritional information would at least spread awareness amongst people. Moreover, one of the major issues for people is the easy access to this these information. Hence, the implementation of this app would revolutionize the fitness and health industry and enable more people to be healthy and fit. Stakeholders such as investors and developers will receive recognition and profit for helping raise community awareness about health and users will become more healthy.
3	Novelty/Uniqueness	The web application will be able to classify the fruits instantly and inform the user about the nutritional information by using real time images from mobile phones.
4	Social Impact/Customer Satisfaction	The customer will have an idea of their daily calories

		intake thus improving the health of the user. There will be reduction in consumption in high calorie-containing fruits which might even affect its production and sale.
5	Business Model	The targeted customers for the product are those people who are health conscious , Fitness enthusiasts and they always want to eat healthy. Their main goal is to consume less calories by avoiding junk food and eating fruits and vegetables. This app will provide them a way to know the calories contained in the fruit. There will be high costs involved in advertisement and this is the most crucial step for the success of the product. To get things right the clear survey or data has to be studied and analyze what are the main smaller functionalities required by the target audience and add these functionalities in the newer versions of the app. The versions have to be regularly released to maintain customer engagement. This might take a high cost for maintenance and this may cause demand for more man power.
6	Scalability of the solution	The proposed solution can be extended to vegetables using the same logic for the fruits. This has fairly low scalability for common food items as by pictures we cannot know the ingredients used.

3.4. Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1.Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Instagram Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email
FR-3	Identification of Nutritional Value in a particular fruit	Identify nutritional value from given input picture and display information to the user
FR-4	Share diet plans with friends	Sharing diet plans by adding fruits taken every day that might engender interest in dieting

4.2.Non-functional Requirements:

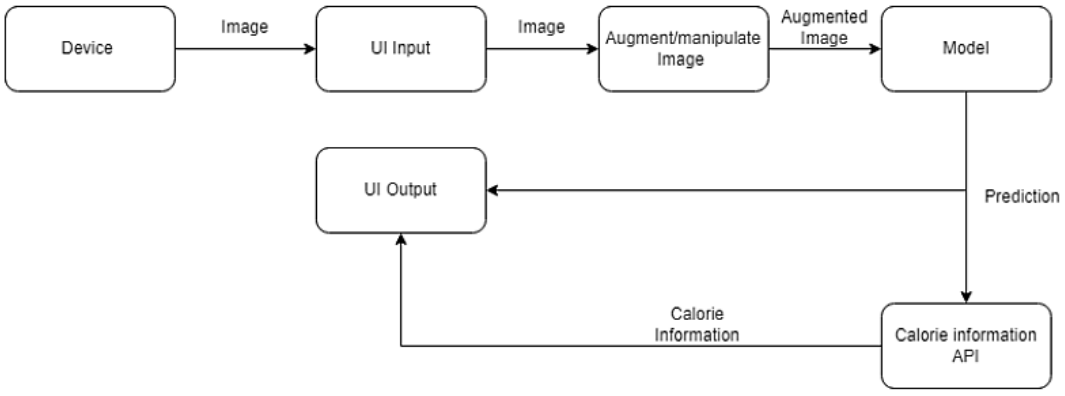
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Easy to use and control information given in the application.
NFR-2	**Security**	Preventing misuse of information is considered to be the topmost priority in the developed application .
NFR-3	**Reliability**	Highly reliable, prone to less failure due to the availability of the application in a cloud environment.
NFR-4	**Performance**	The application is developed in such a way that the classification of fruits and information regarding the nutritional values are concordant with actual observations.
NFR-5	**Availability**	The application is available as a web application which is the most common way in which people look up information on the internet.
NFR-6	**Scalability**	The application can be further scaled to create a mobile application and can further help in providing information on nutritional values of vegetables.

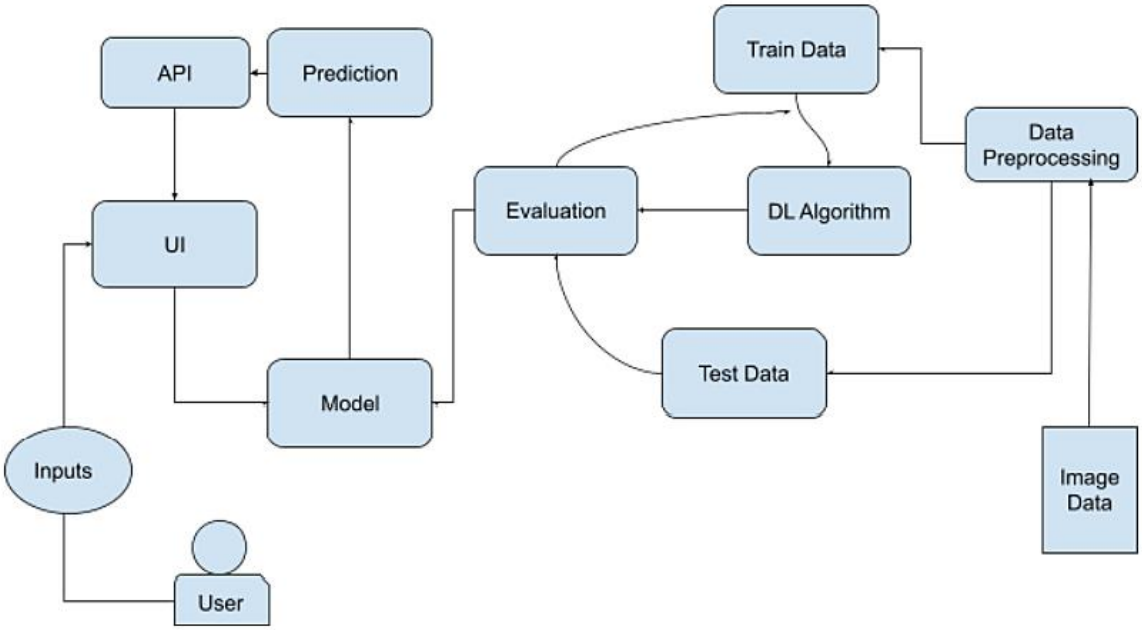
5. PROJECT DESIGN

5.1. Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2.Solution & Technical Architecture



5.3.User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can successfully create an account using email.	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1

		USN-3	As a user, I can register for the application through a mobile number.	I can successfully create an account using my mobile number.	Medium	Sprint-1
		USN-4	As a user, I will receive confirmation by sms once I have registered for the application	I can receive confirmation SMS.	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can access my account / dashboard	High	Sprint-1
	Dashboard	USN-6	As a user, I can navigate through the dashboard and be able to enter the input image via camera or the gallery.	I can navigate and enter the input.	Medium	Sprint-1
	Model	USN-7	As a user, I can classify fruits using the real time images.	I can classify fruits based on images using the application.	High	Sprint-2
	API	USN-8	As a user, I can get the nutritional information of the fruit.	I can obtain nutritional information using the application.	High	Sprint-3
Administrator		USN-9	As an administrator, I can get application reviews from the customers.	I can collect user reviews..	Low	Sprint-4

Testing team		USN-10	As the testing team, I can test the user interface features.	All the UI features work properly.	Mediu m	Sprint-1
		USN-11	As the testing team, I can test the model used for classification.	The model is accurate	High	Sprint-2
		USN-12	As the testing team, I can test the API which contains the nutritional data.	Valid facts are given by the API	HIgh	Sprint-3
		USN-13	As the testing team, I can test the integration of the UI, model and the API.	All the integrated features work together properly.	High	Sprint-4
Maintenance Team	Update the application	USN-14	As a member of the maintenance team, I can update the model and resolve any technical glitches.	The glitches are resolved	Low	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1.Product Backlog, Sprint Schedule, and Estimation

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	I can register for the application by entering my email, password, and	3	High	Sushaanth

			confirming my password.			
Sprint-1		USN-2	I will receive confirmation email once I have registered for the application	2	High	Sandeep
Sprint-1		USN-3	I can register for the application through a mobile number.	2	Medium	Rahul
Sprint-1		USN-4	I will receive confirmation by sms once I have registered for the application	2	Medium	Sharvesh
Sprint-2	Login	USN-5	I can log into the application by entering email & password	1	High	Rahul
Sprint-2	Dashboard	USN-6	I can navigate through the dashboard and be able to enter the input image via camera or the gallery.	2	Medium	Sushaanth, Sharvesh
Sprint-2	Model	USN-7	I can classify fruits using real time images.	5	High	Sandeep, Sushaanth, Rahul, Sharvesh
Sprint-3	API	USN-8	I can get the nutritional information of the fruit.	4	High	Rahul, Sandeep, Sushaanth
Sprint-4		USN-9	I can get application reviews from the customers.	1	Low	Sushaanth
Sprint-1		USN-10	I can test the user interface features.	2	Medium	Sandeep
Sprint-2		USN-11	I can test the model used for classification.	3	High	Sushaanth

Sprint-3		USN-12	I can test the API which contains the nutritional data.	2	HIgh	Sharvesh
Sprint-4		USN-13	I can test the integration of the UI, model and the API.	3	High	Rahul
Sprint-4	Update the application	USN-14	I can update the model and resolve any technical glitches.	2	Low	Sandeep, Sharvesh

6.2.Project Tracker, Velocity & Burndown Chart:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	11	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	11	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	6	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	6	19 Nov 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

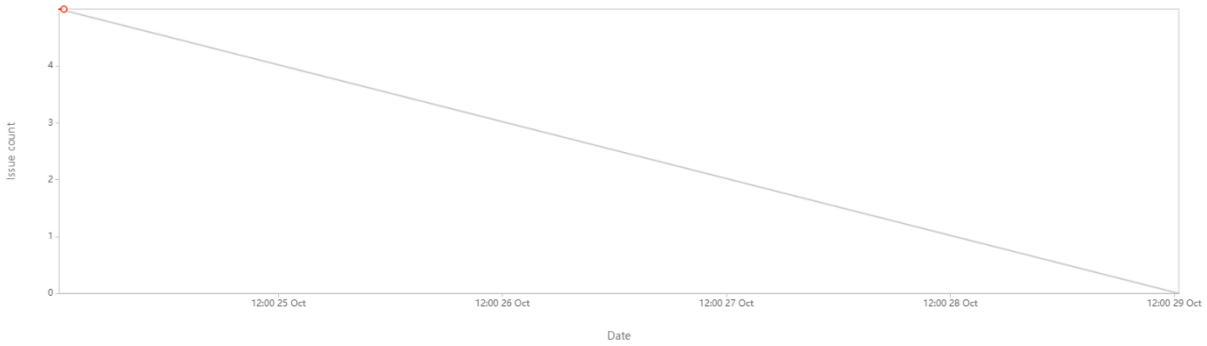
Sprint	Average Velocity
Sprint-1	1.8
Sprint-2	1.8
Sprint-3	1
Sprint-4	1

6.3.Reports from JIRA

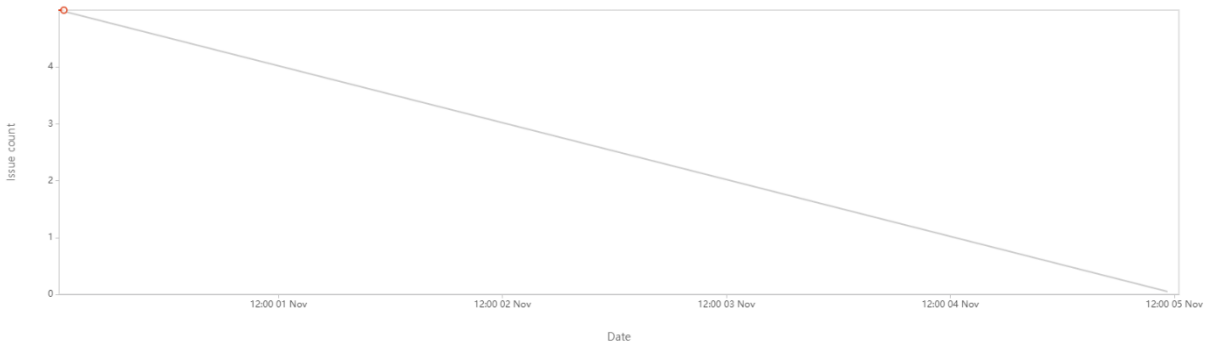
Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

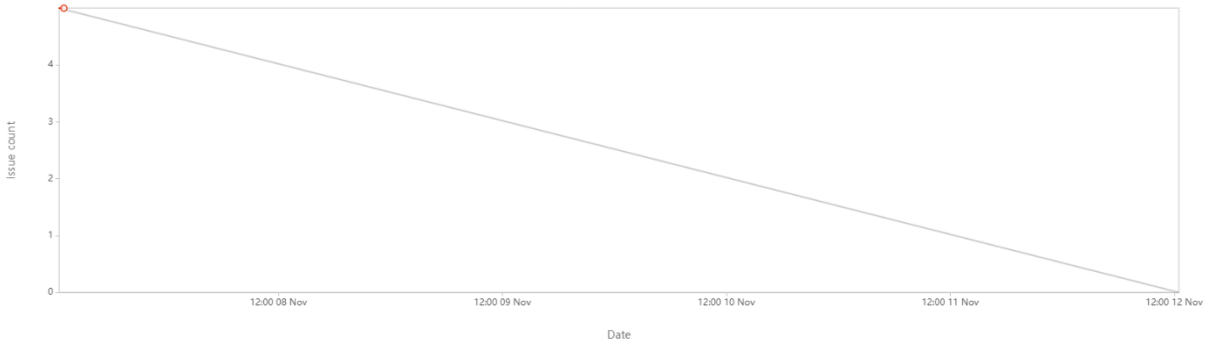
Burndown Chart - Sprint 1



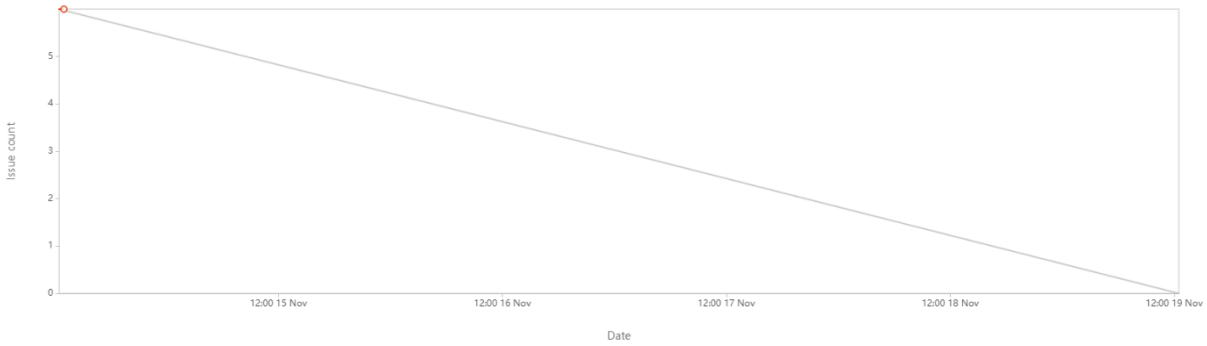
Burndown Chart - Sprint 2



Burndown Chart - Sprint 3



Burndown Chart - Sprint 4



7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1. Image classification model and Flask integration

An image classification model which can classify images of fruits and return the class value of the objects in the images, has been implemented. This model is integrated with the front end using Flask.

Code:

```
# Flask model code
img=image.load_img("some_image.png",target_size=(64,64))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)

pred=np.argmax(model.predict(x), axis=1)
print("prediction",pred)
index = ['Apple','Banana','Orange','Pineapple','Watermelon']

result = str(index[pred[0]])

print(result)
```

7.2. React webpage

A user interface is created using react to provide a way to take inputs from the user and display the results.

Code to send file to the backend:

```
async function sendFileToBackend() {
        console.log("TEST...");
        setFruit("Loading...");

        var imgData = "";
        var bodyFormData = new FormData();
```

```
    try {
        imgData = await toBase64(selectedImage);
    } catch (error) {
        console.error(error);
    }
    console.log("Img data");
    console.log(imgData);

    // console.log("img data type: ");
    // console.log(typeof imgData);

    bodyFormData.append("imageString", imgData);

    const headers = {
        "Content-Type": "multipart/form-data",
        "Access-Control-Allow-Origin": "*",
    };

    axios
        .post("http://localhost:5000/api/classify", bodyFormData, {
            headers: headers,
        })
        .then((response) => {
            console.log(response);
            setFruit(response.data.fruit);
            // console.log(response.data.listItems.items);
            console.log("Response data list items");
            var temp = JSON.parse(response.data.listItems).items[0];
            setAns(temp);
        })
        .catch((error) => {
            console.log(error);
        });
}
```

8. **TESTING**

Test Cases and User Acceptance Testing

TEST CASE ID	TEST SCENARIO	TESTING STEPS	TEST DATA	EXPECTED OUTCOME	ACTUAL OUTCOME	PASS OR FAIL
T01	Predict the fruit using preloaded image	Load the fruit image on	Banana Image	Banana prediction with its nutritional value	Banana prediction with its nutritional value	Pass

T02	Predict the fruit using preloaded image	Load the fruit image on	Apple Image	Apple prediction with its nutritional value	Apple prediction with its nutritional value	Pass
T03	Predict the fruit using preloaded image	Load the fruit image on	Pineapple Image	Pineapple prediction with its nutritional value	Pineapple prediction with its nutritional value	Pass
T04	Predict the fruit using preloaded image	Load the fruit image on	Watermelon Image	Watermelon prediction with its nutritional value	Watermelon prediction with its nutritional value	Pass
T05	Predict the fruit using preloaded image	Load the fruit image on	Orange Image	Orange prediction with its nutritional value	Orange prediction with its nutritional value	Pass
T06	Predict the any other object using pre-loaded image	Load the image	Other Object Image	Nearest matching fruit like is mapped	Nearest matching fruit is mapped	Pass
T07	Login	Enter the details	Correct input	Logged in to the web app	Logged in to the web app	Pass
T08	Login	Enter the details	Incorrect input	Login denied	Login denied	Pass
T09	Integrating API into flask app	Rapid calorie Ninga API is integrated into flask App	None	None	None	Pass
T10	Getting nutritional value using API	Enter the fruit name to the API to get the nutritional value	Fruit name	API gives nutritional value	API gives nutritional value	Pass

RESULTS

The created application was able to retrieve the nutritional information of the fruits in the given images. The model was able to achieve an accuracy of 44.6% in the testing data set.

ADVANTAGES & DISADVANTAGES

- The application is able to retrieve nutritional data for a given food in an image.
- The application provides an easy to user interface using React.js
- The application is modular in nature which make future updation of the application easier.
- The application accuracy for real time images is not high.
- The information retrieval could be slow if the internet speed is low.

CONCLUSION

The web-app successfully identifies the fruit and gives its nutritional value which would be helpful to all its user to achieve health goals.

FUTURE SCOPE

The web-app can be improved by using better classification model which is very light which may exist in future. The application could be extended to vegetables. The web-app can be remodeled to be more user friendly also android version of the application can be made.

APPENDIX

GitHub & Project Demo Link

GitHub Link:
https://github.com/IBM-EPBL/IBM-Project-14249-1659547104

Video Link:
https://drive.google.com/file/d/1Yf0M1frvJAL5c3FU7B0_UzIi6U0jtOP5/view?usp=sharing

Source Code

React Component:

_app.js

```
import '../styles/globals.css'

function MyApp({ Component, pageProps }) {
  return <Component {...pageProps} />
}

export default MyApp
```

index.js

```
import Head from "next/head";
```

```
import { ChakraProvider, Flex } from "@chakra-ui/react";

import { useRouter } from "next/router";

import Nav from "../components/Nav";

const navItems = [
 {
        label: "Home",
        href: "/",
 },
 {
        label: "Classify",
        href: "/classify",
 },
 {
        label: "About",
        children: [
                {
                        label: "About the WebApp",
                        // subLabel: "Find your dream design job",
                        href: "/about-webapp",
                },
                {
                        label: "About the Team",
                        // subLabel: "An exclusive list for contract work",
                        href: "/about-team",
                },
        ],
 },
];

export default function Home() {
 const { asPath } = useRouter();

 return (
        <ChakraProvider>
                <div>
                        <Head>
                                <title>APNA - AI-Powered Nutrition Analyzer</title>
                                <link rel="icon" href="/favicon.ico" />
                        </Head>
                        <Nav NAV_ITEMS={navItems} />
                        <Flex                height="30vh"                alignItems="center"
justifyContent="center"></Flex>
```

```
                        <Flex height="60vh" justifyContent="center">
                                {asPath}
                        </Flex>
                    </div>
            </ChakraProvider>
    );
}

// import Head from 'next/head'
// import Image from 'next/image'
// import styles from '../styles/Home.module.css'

// export default function Home() {
//   return (
//     <div className={styles.container}>
//       <Head>
//         <title>Create Next App</title>
//         <meta name="description" content="Generated by create next app" />
//         <link rel="icon" href="/favicon.ico" />
//       </Head>

//       <main className={styles.main}>
//         <h1 className={styles.title}>
//           Welcome to <a href="https://nextjs.org">Next.js!</a>
//         </h1>

//         <p className={styles.description}>
//           Get started by editing{' '}
//           <code className={styles.code}>pages/index.js</code>
//         </p>

//         <div className={styles.grid}>
//           <a href="https://nextjs.org/docs" className={styles.card}>
//             <h2>Documentation &rarr;</h2>
//             <p>Find in-depth information about Next.js features and API.</p>
//           </a>

//           <a href="https://nextjs.org/learn" className={styles.card}>
//             <h2>Learn &rarr;</h2>
//             <p>Learn about Next.js in an interactive course with quizzes!</p>
//           </a>

//           <a
//             href="https://github.com/vercel/next.js/tree/canary/examples"
//             className={styles.card}
//           >
```

```
//            <h2>Examples &rarr;</h2>
//            <p>Discover and deploy boilerplate example Next.js projects.</p>
//          </a>

//          <a
//            href="https://vercel.com/new?utm_source=create-next-app&utm_medium=default-
template&utm_campaign=create-next-app"
//            target="_blank"
//            rel="noopener noreferrer"
//            className={styles.card}
//          >
//            <h2>Deploy &rarr;</h2>
//            <p>
//              Instantly deploy your Next.js site to a public URL with Vercel.
//            </p>
//          </a>
//        </div>
//      </main>

//      <footer className={styles.footer}>
//        <a
//                    href="https://vercel.com?utm_source=create-next-app&utm_medium=default-
template&utm_campaign=create-next-app"
//          target="_blank"
//          rel="noopener noreferrer"
//        >
//          Powered by{' '}
//          <span className={styles.logo}>
//            <Image src="/vercel.svg" alt="Vercel Logo" width={72} height={16} />
//          </span>
//        </a>
//      </footer>
//    </div>
//  )
// }
```

about-webapp.js:

```
import Head from "next/head";

import { ChakraProvider, Flex, Heading, Text, VStack } from "@chakra-ui/react";

import { useRouter } from "next/router";

import Nav from "../components/Nav";
```

```
const navItems = [
 {
        label: "Home",
        href: "/",
 },
 {
        label: "Classify",
        href: "/classify",
 },
 {
        label: "About",
        children: [
                {
                        label: "About the WebApp",
                        // subLabel: "Find your dream design job",
                        href: "/about-webapp",
                },
                {
                        label: "About the Team",
                        // subLabel: "An exclusive list for contract work",
                        href: "/about-team",
                },
        ],
 },
];

export default function Home() {
 const { asPath } = useRouter();

 return (
        <ChakraProvider>
                <div>
                        <Head>
                                <title>APNA - About Webapp</title>
                                <link rel="icon" href="/favicon.ico" />
                        </Head>
                        <Nav NAV_ITEMS={navItems} />
                        <Flex height="30vh" alignItems="center" justifyContent="center">
                                <Heading as="h1" size="4xl" noOfLines={1}
textColor="green.400">
                                        About
                                </Heading>
                        </Flex>
                        <Flex height="60vh" justifyContent="center">
                                <VStack spacing={4}>
                                        <Text fontSize="xl">
```

```
                This application allows you to upload images of
fruits and
                vegetables. <br />
                It identifies the fruit/vegetable and also provides
nutritional
                statistics in order to help you plan out your diet.
            </Text>

            <Text fontSize="xl">
                The problem of inaccessible diet plans in the
palm of your hand
                affects fitness enthusiasts, individuals who
follow diets, and
                <br />
                people who cannot go to the gym or follow a
costly diet. If fixed,
                many more people need not rely on gyms for
fitness and dieting.
                Even
                <br />
                though diet plans can be contrained by culture
and religion of
                different people in food products like meat, easy
access to
                <br />
                nutritional information would atleast spread
awareness amongst
                people. Moreover, one of the major issues for
people is the easy
                <br />
                access to this kind of information. Hence, the
implementation of
                this app would revolutionise the fitness and
health industry and
                <br />
                enable more people to be healthy and fit.
Stakeholders such as
                investors and developers will receive recognition
and profit for
                <br />
                helping raise community awareness about health
and users will
                become more healthy.
                <br />
            </Text>
        </VStack>
```

```
                    </Flex>
                </div>
            </ChakraProvider>
    );
}
```

about-team.js

```
import Head from "next/head";

import { ChakraProvider, Flex } from "@chakra-ui/react";

import { useRouter } from "next/router";

import Nav from "../components/Nav";

const navItems = [
  {
        label: "Home",
        href: "/",
  },
  {
        label: "Classify",
        href: "/classify",
  },
  {
        label: "About",
        children: [
            {
                label: "About the WebApp",
                // subLabel: "Find your dream design job",
                href: "/about-webapp",
            },
            {
                label: "About the Team",
                // subLabel: "An exclusive list for contract work",
                href: "/about-team",
            },
        ],
  },
];

export default function Home() {
  const { asPath } = useRouter();

  return (
```

```
        <ChakraProvider>
                <div>
                        <Head>
                                <title>APNA - About Team</title>
                                <link rel="icon" href="/favicon.ico" />
                        </Head>
                        <Nav NAV_ITEMS={navItems} />
                        <Flex height="30vh" alignItems="center" justifyContent="center">
                                ABOUT TEAM
                        </Flex>
                        <Flex height="60vh" justifyContent="center">
                                Sharvesh Shankar, Sushaanth Srinivasan, Sandeep Sekhar,
Rahul Kumar
                        </Flex>
                </div>
        </ChakraProvider>
 );
}
```

Classify.js

```
import Head from "next/head";
import axios from "axios";

import {
 ChakraProvider,
 Heading,
 Flex,
 IconButton,
 HStack,
 VStack,
 StackDivider,
 TableContainer,
 Table,
 TableCaption,
 Thead,
 Tr,
 Th,
 Tbody,
 Td,
 Text,
} from "@chakra-ui/react";

import Nav from "../components/Nav";
import UploadAndDisplayImage from "../components/UploadAndDisplayImage";
```

```
import { useRef, useState } from "react";

const navItems = [
 {
        label: "Home",
        href: "/",
 },
 {
        label: "Classify",
        href: "/classify",
 },
 {
        label: "About",
        children: [
                {
                        label: "About the WebApp",
                        // subLabel: "Find your dream design job",
                        href: "/about-webapp",
                },
                {
                        label: "About the Team",
                        // subLabel: "An exclusive list for contract work",
                        href: "/about-team",
                },
        ],
 },
];

export default function Home() {
 const [selectedImage, setSelectedImage] = useState(null);
 const [ans, setAns] = useState("");
 const [fruit, setFruit] = useState("");

 const sendDataToParent = (image) => {
        // console.log(image);
        setSelectedImage(image);
 };

 const toBase64 = (file) =>
        new Promise((resolve, reject) => {
                const reader = new FileReader();
                reader.readAsDataURL(file);
                reader.onload = () => resolve(reader.result);
                reader.onerror = (error) => reject(error);
        });
```

```
async function sendFileToBackend() {
	console.log("TEST...");
	setFruit("Loading...");

	var imgData = "";
	var bodyFormData = new FormData();

	try {
		imgData = await toBase64(selectedImage);
	} catch (error) {
		console.error(error);
	}
	console.log("Img data");
	console.log(imgData);

	// console.log("img data type: ");
	// console.log(typeof imgData);

	bodyFormData.append("imageString", imgData);

	const headers = {
		"Content-Type": "multipart/form-data",
		"Access-Control-Allow-Origin": "*",
	};

	axios
		.post("http://localhost:5000/api/classify", bodyFormData, {
			headers: headers,
		})
		.then((response) => {
			console.log(response);
			setFruit(response.data.fruit);
			// console.log(response.data.listItems.items);
			console.log("Response data list items");
			var temp = JSON.parse(response.data.listItems).items[0];
			setAns(temp);
		})
		.catch((error) => {
			console.log(error);
		});
}

return (
	<ChakraProvider>
		<div>
			<Head>
```

```
                        <title>APNA - Classify</title>
                        <link rel="icon" href="/favicon.ico" />
                    </Head>
                    <Nav NAV_ITEMS={navItems} />
                    <Flex height="30vh" alignItems="center" justifyContent="center">
                        <Heading as="h1" size="4xl" noOfLines={1}
textColor="green.400">
                            Classify
                        </Heading>
                    </Flex>
                    <HStack spacing={8}>
                        <Flex height="50vh" width="100em" justifyContent="center">
                            <UploadAndDisplayImage
                                sendDataToParent={sendDataToParent}
                                sendFileToBackend={sendFileToBackend}
                            />
                        </Flex>
                        <Flex height="50vh" width="100em" justifyContent="center">
                            <Flex height="100vh" justifyContent="center">
                                <TableContainer>
                                    <Table variant="striped"
colorScheme="green">
                                        <TableCaption>Nutrient
Statistics</TableCaption>
                                        <Thead>
                                            <Tr>

 <Th>Nutrient</Th>

 <Th>Quantity</Th>
                                            </Tr>
                                        </Thead>
                                        <Tbody>
                                            <Tr>
                                                <Td>Fruit</Td>
                                                <Td>{fruit}</Td>
                                            </Tr>
                                            <Tr>
                                                <Td>Sugar
(g)</Td>

 <Td>{ans["sugar_g"]}</Td>
                                            </Tr>
                                            <Tr>
                                                <Td>Fiber
(g)</Td>
```

```
 <Td>{ans["fiber_g"]}</Td>
                                                    </Tr>
                                                    <Tr>
                                                        <Td>Serving  Size
(g)</Td>

 <Td>{ans["serving_size_g"]}</Td>
                                                    </Tr>
                                                    <Tr>
                                                        <Td>Sodium
(mg)</Td>

 <Td>{ans["sodium_mg"]}</Td>
                                                    </Tr>
                                                    <Tr>
                                                        <Td>Potassium
(mg)</Td>

 <Td>{ans["potassium_mg"]}</Td>
                                                    </Tr>
                                                    <Tr>
                                                        <Td>Fat Saturated
(mg)</Td>

 <Td>{ans["fat_saturated_g"]}</Td>
                                                    </Tr>
                                                    <Tr>
                                                        <Td>Fat       Total
(g)</Td>

 <Td>{ans["fat_total_g"]}</Td>
                                                    </Tr>
                                                    <Tr>

 <Td>Calories</Td>

 <Td>{ans["calories"]}</Td>
                                                    </Tr>
                                                    <Tr>
                                                        <Td>Cholesterol
(mg)</Td>

 <Td>{ans["cholesterol_mg"]}</Td>
                                                    </Tr>
                                                    <Tr>
```

```
                                                                    <Td>Protein
(g)</Td>

 <Td>{ans["protein_g"]}</Td>
                                                                        </Tr>
                                                                        <Tr>
                                                                            <Td>Total
Carbohydrates (g)</Td>

 <Td>{ans["carbohydrates_total_g"]}</Td>
                                                                        </Tr>
                                                                    </Tbody>
                                                                </Table>
                                                            </TableContainer>
                                                        </Flex>
                                                    </Flex>
                                                </HStack>
                                            </div>
                                        </ChakraProvider>
 );
}
```

Image Classification Model Component:

```
import numpy as np

import os

import time

import cv2

import pandas as pd

import matplotlib.pyplot as plt

from math import *

import tensorflow as tf

import keras

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.preprocessing.image import img_to_array

from tensorflow.keras.preprocessing.image import load_img


import argparse
```

```
mainPath="F:\IBM\TRAIN_SET"
destPath="F:\IBM\DEST"


im_size = 224
rangeLower=0
rangeUpper=100
imgsInFile=30
startFileCount=0
augmentCount=5
apples=995
b=1374
o=1019
pa=275
wm=475

classId=["APPLES","BANANA","ORANGE","PINEAPPLE","WATERMELON"]

from tensorflow.keras import layers
from tensorflow.keras.applications import EfficientNetV2S
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import LabelEncoder , OneHotEncoder

inputs = layers.Input(shape=(im_size, im_size, 3))
size = (im_size, im_size)

noOfClasses = 5
imgCategory=np.array([i for i in range(noOfClasses)])
inputImg=[]
```

```
imgLabels=[]



for i in range(len(classId)):
    dirPath=mainPath+"\\"+str(classId[i])
    cnt=0
    for j in os.listdir(dirPath):
        ImgPath=mainPath+"\\"+str(classId[i])+"\\"+j
        image = cv2.imread(ImgPath)
        if(image is not None):
            #BGR->RGB
            img2 = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
            img3 = cv2.resize(img2, (im_size, im_size))
            img3 = img3.astype('float32') / 255.0
            inputImg.append(img3)
            imgLabels=imgLabels+[i]
        cnt+=1



#data=np.array(inputImg[:60]+inputImg[apples:apples+60]+inputImg[apples+b:apples+b+60]+inputI
mg[apples+b+o:apples+b+o+60]+inputImg[apples+b+o+pa:apples+b+o+pa+60])

#labels=np.array(imgLabels[:60]+imgLabels[apples:apples+60]+imgLabels[apples+b:apples+b+60]+i
mgLabels[apples+b+o:apples+b+o+60]+imgLabels[apples+b+o+pa:apples+b+o+pa+60])

data=np.array(inputImg)

labels=np.array(imgLabels)

df = pd.DataFrame(labels,columns=["A"])


print(df)

print(type(df))

one_hot_encoded_data = pd.get_dummies(df,columns=["A"]).to_numpy()

print(one_hot_encoded_data)
```

```
print(data.shape)

print(labels.shape)

print(data.size)

print(labels.size)

print(len(inputImg))

print(len(imgLabels))

print(one_hot_encoded_data.shape)

outputs = tf.keras.applications.mobilenet_v2.MobileNetV2(include_top=True, weights=None,
classes=noOfClasses)(inputs)

model = tf.keras.Model(inputs, outputs)

model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"] )

model.summary()

model.fit(data, one_hot_encoded_data,shuffle=True, epochs=5, batch_size=30)

if not(os.path.exists('./WeightsDir2')):

   os.makedirs('./WeightsDir2')

model.save_weights('./WeightsDir2/weights')


testApples=266

testBanana=415

testOrange=248

os.listdir("F:\IBM\TEST_SET")

testPath="F:\IBM\TEST_SET"

inputImgT=[]

imgLabelsT=[]


tClassId=["APPLES","BANANA","ORANGE"]


for i in range(3):

   dirPath=testPath+"\\"+str(tClassId[i])

   #print(dirPath)

   cnt=0
```

```
    #F:\IBM\TEST_SET\APPLES
    for j in os.listdir(dirPath):
        ImgPath=dirPath+"\\"+j
        image = cv2.imread(ImgPath)
        #print(ImgPath)
        if(image is not None):
            #BGR->RGB
            img2 = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
            img3 = cv2.resize(img2, (im_size, im_size))
            img3 = img3.astype('float32') / 255.0
            inputImgT.append(img3)
            imgLabelsT=imgLabelsT+[i]
        cnt+=1
    #print(cnt)
print(len(inputImgT))
print(len(imgLabelsT))
dataT=np.array(inputImgT)
labelsT=np.array(imgLabelsT+[3,4])
print(dataT.shape)
print(labelsT.shape)
dfT = pd.DataFrame(labelsT,columns=["A"])
print(dfT)
print(type(dfT))
one_hot_encoded_dataT = pd.get_dummies(dfT,columns=["A"]).to_numpy()
one_hot_encoded_dataT = one_hot_encoded_dataT[:-2]
print(one_hot_encoded_dataT)
pred=model.predict(dataT)
model.evaluate(dataT,one_hot_encoded_dataT)
```

Flask Component:

```
from flask import Flask, jsonify, request
import tensorflow as tf
import numpy as np
import base64
from flask_cors import CORS
import requests
import os
import cv2
import json
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model

app = Flask(__name__)
CORS(app)

API_KEY = os.environ.get("API_KEY")
# label_file = "static/labels.json"
# modelname = "static/models/nutrition.h5"

model = load_model('.')

# sign_classifier = tf.keras.models.load_model('static/models/nutrition.h5')
res = None

# def predict(data):
#     res = np.array([data])
#     res = res.reshape(res.shape[0], res.shape[1], -1)
#     preds = sign_classifier.predict(res)
#     result = labels[np.argmax(preds[0])]
```

```
#     return result

def load_json(filename):
    with open(filename, 'r') as json_file:
        data = json.load(json_file)
    return data

@app.route('/')
def index():
    return "<h1>Hello World</h1>"

@app.route('/api/classify', methods=['GET', 'POST'])
def classify():
    if request.method == 'POST':
        img_string = request.form.get("imageString")

        metadata = img_string[:22]

        index1 = metadata.find('data:image/') + 11
        index2 = metadata.find(';base64')

        img_string = img_string[22:]

        with open("Output.txt", "w") as text_file:
            text_file.write(img_string)

        imgData = base64.b64decode(img_string)

        filename = 'some_image.png'
        with open(filename, 'wb') as f:
            f.write(imgData)
```

```
# model code
img=image.load_img("some_image.png",target_size=(64,64))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)

pred=np.argmax(model.predict(x), axis=1)
print("prediction",pred)
index = ['Apple','Banana','Orange','Pineapple','Watermelon']

result = str(index[pred[0]])

print(result)

# result = "apple"

# calorie ninja api hit
url = "https://calorieninjas.p.rapidapi.com/v1/nutrition"
querystring = {"query": result}
headers = {
   "X-RapidAPI-Key": API_KEY,
   "X-RapidAPI-Host": "calorieninjas.p.rapidapi.com"
}

response = requests.request("GET", url, headers=headers, params=querystring)
print("Response: ")
print(response.text)

responseText = response.text

return jsonify(status = 200, fruit=result, listItems = responseText)
```

```
    return "<h1>Invalid Request</h1>"


if __name__ == '__main__':
    app.run(debug=True)
```