Mobile App Development

Project (SYBScIT)

Topic – MANDIPEDIA – Analysis App

Vikas Rana | Varun Kumar | Jeet Waral | Alan Cardoza

The Journey

1

While shopping for groceries through Zepto one day, we wondered—wouldn't it be convenient if there was a single platform that could compare the prices of fruits and vegetables across multiple apps? Instead of switching between apps to find the best deal, we envisioned an app that could do the comparison for us, helping users select the most cost-effective option to suit their needs.

2

After discussing our idea, we decided to build the app using Flutter. Its cross-platform capabilities, ease of development, and smooth performance made it the perfect choice for our project. We outlined key features and planned how the app would function, focusing on user-friendliness and accurate price comparisons. With a clear goal in mind, we started designing the interface, setting up the backend, and implementing essential features to bring Mandipedia to life.

3

As we delved deeper into the project, we recognized the importance of understanding how inflation has affected the prices of fruits and vegetables. What better way to showcase this data than through graphical representation? Thus, we incorporated a screen that provides users with visual insights into price trends over time.

4

To enhance user experience, we wanted to provide personalized recommendations based on past orders. By tracking user purchases, our app suggests frequently bought fruits and vegetables, helping users make quick and efficient decisions while shopping.

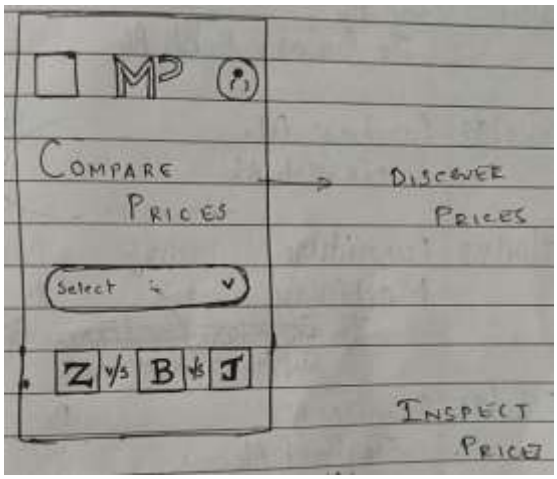
5

The integration of Naive Bayes Theorem brought another innovative aspect to our app. By applying this predictive model, we developed a feature that forecasts future prices of fruits and vegetables. This empowers users with insights into potential price fluctuations, allowing them to make informed purchasing decisions.

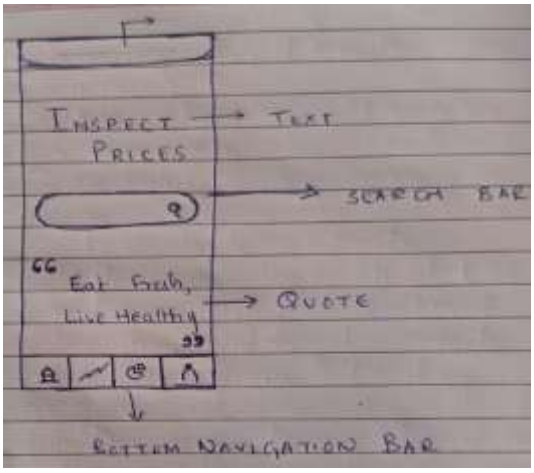
6

Inspired by the popular Akinator game, we wanted to introduce an interactive element to Mandipedia. Enter **FRUVENATOR**—a fun, engaging game where users think of a fruit or vegetable, and our app attempts to guess it by asking a series of targeted questions. To achieve this, we implemented **decision tree-based classification**, where each question helps narrow down the possible choices based on distinguishing features like colour, seasonality, or taste. This feature not only increases user engagement but also adds a playful aspect to our app by blending entertainment with machine learning.

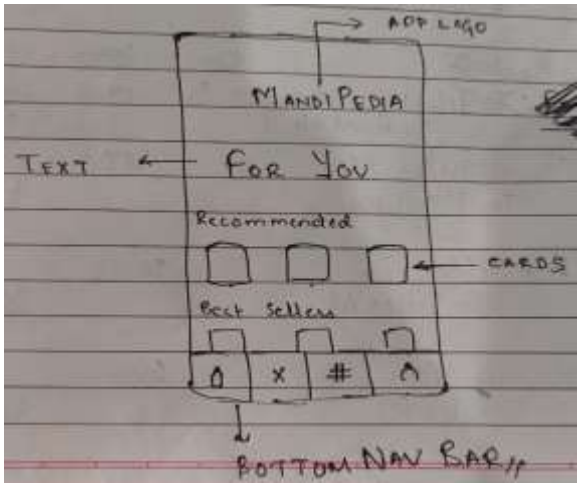
Initial Concept Design



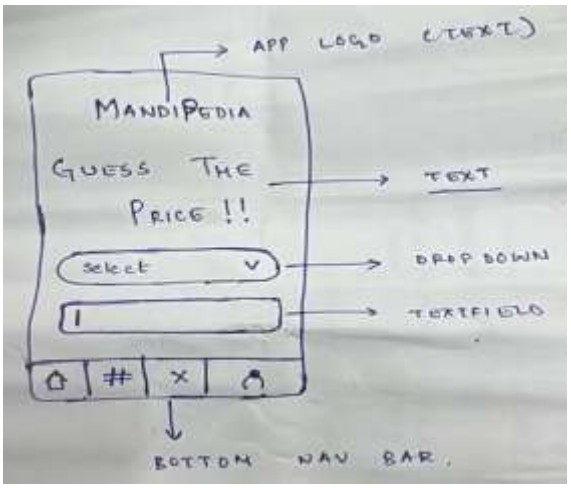
Screen 1: Compare Prices



Screen 2: Inspect Prices



Screen 3: For You



Screen 4: Guess the Price

For the 5th Screen (FruveNator), we directly adopted the Akinator-style theme without an initial hand-drawn design, as the concept was already well-defined and implemented accordingly.

During development, we made several refinements to our initial designs based on feasibility, UI/UX improvements, and technical constraints. While the core functionality remains intact, some visual and structural modifications were necessary. Below, we present the original concept designs alongside the final implemented screens to highlight the evolution of our project.

Individual Screens

1. Splash Screen

I) Definition

A splash screen is the first screen that appears when an app is launched. It usually displays the app's logo, name, or a brief animation while the app loads in the background. It enhances the user experience by providing a smooth transition into the main interface.

II) Importance

In Mandipedia, the splash screen serves multiple purposes:

a) **Brand Identity** – It introduces users to the app with a visually appealing logo and design.

b) **Smooth Loading Experience** – While data such as price comparisons and recommendations are being fetched, the splash screen keeps users engaged.

c) **First Impression** – A well-designed splash screen ensures a professional and polished look, making users feel confident about the app's reliability.

d) **Preloading Data** – It allows us to load essential data in the background before the user interacts with the main interface, improving app performance.

- **IMAGES**


MP
MandiPedia
Discover Prices, Explore Trends
MP
MandiPedia
Discover Prices, Explore Trends

2. Comparing Prices

I) UL/UX Design

For the price comparison feature, we designed an intuitive and user-friendly interface that allows users to view price differences at a glance. The UI focuses on clarity and efficiency, ensuring that users can easily identify the cheapest option without excessive navigation. A simple yet effective layout with a combination of lists, filters, and graphical representations ensures a seamless experience.

II) Web Scraping

To retrieve real-time price data from various grocery platforms, we implemented web scraping. This technique enables us to extract the latest pricing information, ensuring that users always have access to up-to-date comparisons. By continuously fetching and updating the data, our app maintains accuracy and relevance.

- **APIs and HTML Code Analysis**: Since some grocery platforms restrict scraping, we analysed the HTML structure of various grocery apps and integrated APIs where available.
- **Selenium in Python:** For the scraping process, we used Selenium in Python to automate browser interactions and fetch necessary data efficiently.

III) Analysis

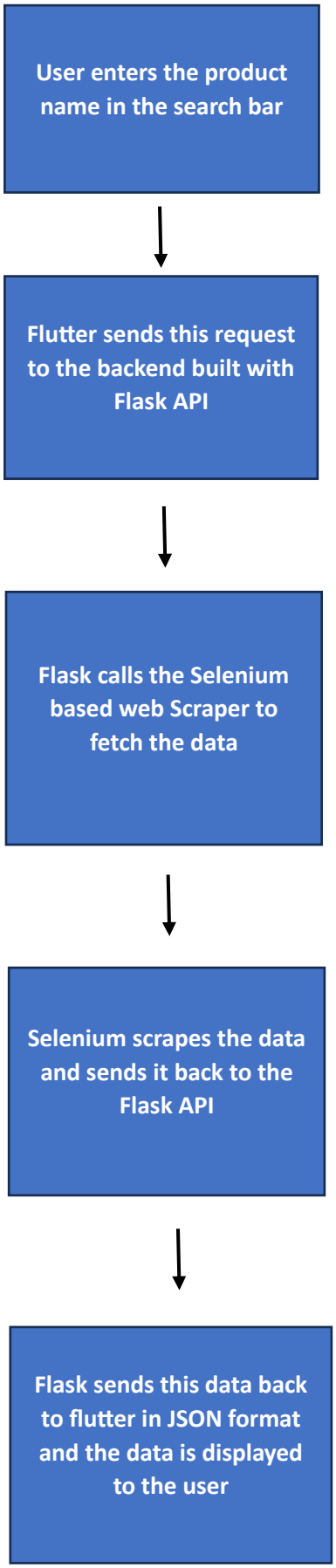
- **Comparing Prices Analysis** – We fetch data using **web scraping** and compare it against various sources to ensure accuracy. The comparison is done dynamically, ensuring that the best price across multiple grocery platforms is always presented to the user. This step involves filtering duplicate entries, normalizing price formats, and cross-checking values against historical trends for consistency.

IV) What we Achieved

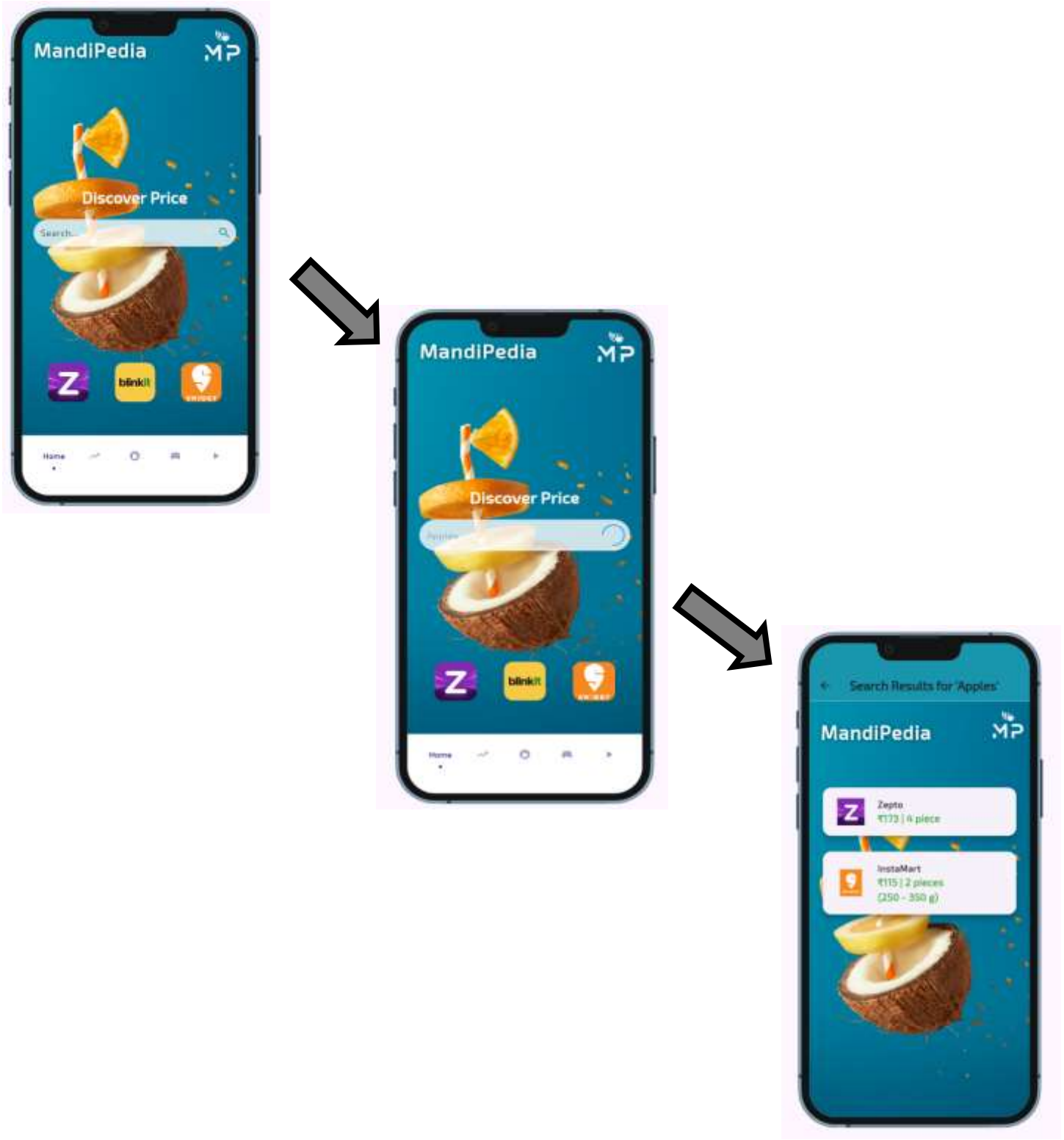
By implementing the price comparison feature, we successfully accomplished several key objectives:

- **Time and Cost Savings for Users**: Users no longer need to manually check multiple grocery apps for price comparisons. Our app provides a one-stop solution to find the best deals instantly
- **Real-Time Data Accuracy**: Through web scraping and API integration, we ensured that users always get the most updated and accurate price listings.
- **Enhanced User Experience**: The intuitive UI and well-organized price display make it easy for users to compare prices and make informed purchasing decisions effortlessly.
- **Improved Shopping Efficiency**: By filtering and sorting price data effectively, users can quickly find the most cost-effective options, leading to better shopping decisions.
- **Scalability**: Our approach allows for easy integration with additional grocery platforms, making it adaptable for future enhancements.

Flow Chart


User enters the product name in the search bar
Flutter sends this request to the backend built with Flask API
Flask calls the Selenium based web Scraper to fetch the data
Selenium scrapes the data and sends it back to the Flask API
Flask sends this data back to flutter in JSON format and the data is displayed to the user

- **Images**


MandiPedia
Discover Price
blinkit
MandiPedia
Discover Price
blinkit
Search Results for 'Apples'
MandiPedia
Zepto
InstaMart

3. Graphical Analysis

I) Collection of Data

To analyse price trends over time, we needed historical pricing data for fruits and vegetables. We obtained this data from Kaggle, a popular platform that provides datasets for machine learning and data analysis. The dataset contained the prices of various fruits and vegetables recorded over a period of one year, with monthly entries.

This data serves as the foundation for our price trend analysis, helping users visualize how inflation and seasonal changes impact the cost of groceries. By leveraging Kaggle's structured datasets, we ensured accuracy and reliability in our analysis.

II) Conversion of Excel Data into JSON

The raw dataset from Kaggle was in Excel format (.xlsx), which is not directly usable in our Flutter application. To make the data more accessible and efficiently loadable, we converted it into a JSON format.

Steps involved in conversion:

1. **Loading the Excel file** – We used Pandas (Python library) to read the dataset.
2. **Cleaning the data** – Unnecessary columns and null values were removed to ensure a clean dataset.
3. **Converting to JSON** – The processed data was transformed into JSON format using Pandas' .to_json() method.
4. **Storing the JSON file** – The final JSON file was saved and integrated into our Flutter project for easy data retrieval.

This conversion allowed us to efficiently handle, parse, and display data within the app without performance issues. JSON format also made it easier to fetch and manipulate the data dynamically within Flutter.

III) Analysis

1. **Taking User Input** – The user selects a fruit or vegetable they want to analyse.
2. **Matching with the Database** – The input is matched against our pre-stored **JSON database**, which contains historical price data for various items.
3. **Analysing Price Trends** – Once the selected item is found, the system retrieves its historical price data and generates a **trend analysis**. This includes:
    - Identifying seasonal fluctuations in pricing.
    - Comparing past and present costs to understand inflation trends.

IV) Representation of the Analysed Data

Once the data was converted into JSON, the next step was to visually represent the analysed data within the Mandipedia app. To achieve this, we used flutter_riverpod, a state management package that simplifies data handling and UI updates.

Implementation details:

- **State Management** – We used flutter_riverpod to efficiently manage and update the price data dynamically.
- **Graphical Representation** – Price trends were displayed using line charts and bar graphs, helping users understand how prices fluctuated over time.

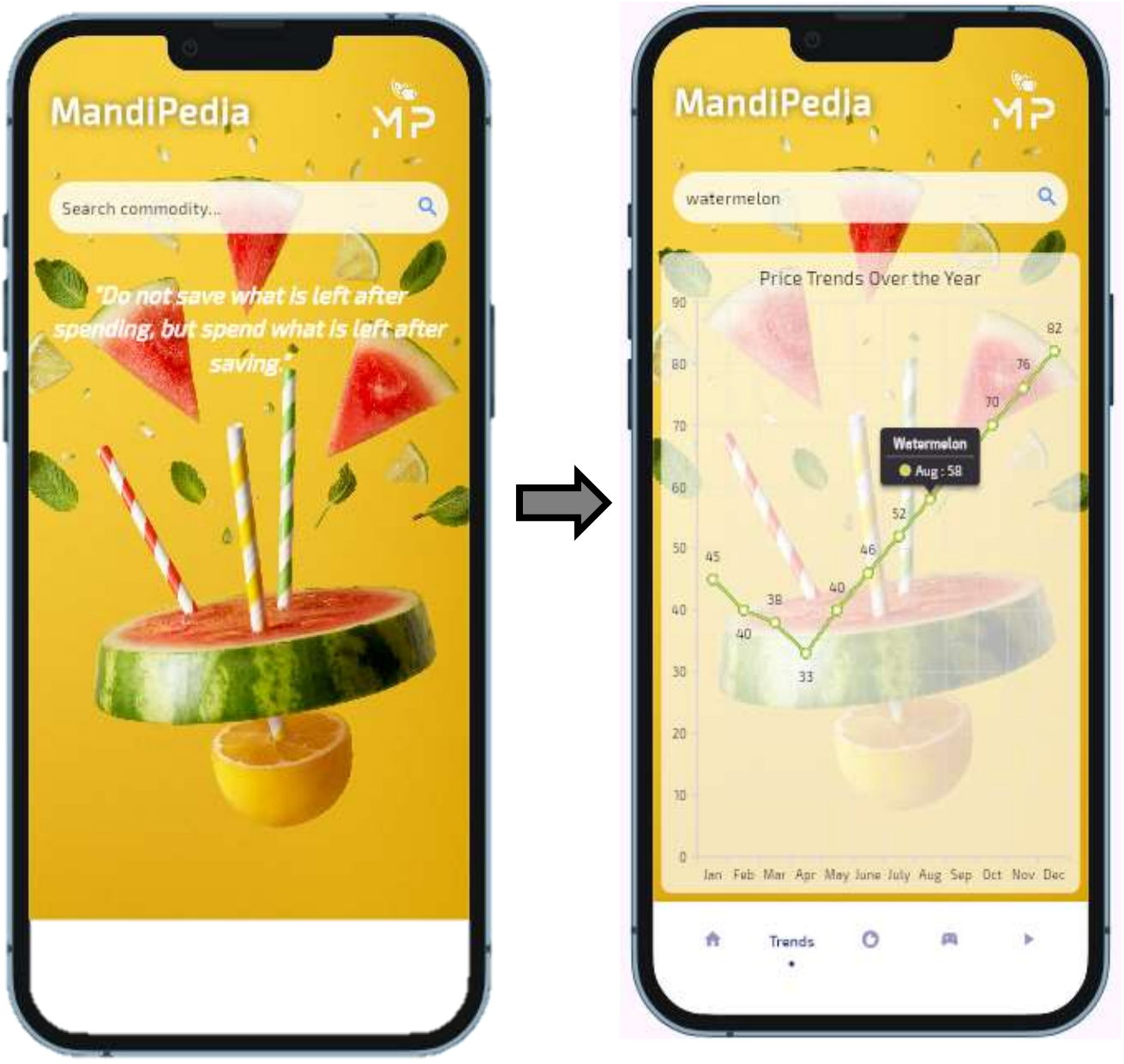
- **User Interaction** – The graphs allowed users to filter data based on time range, specific fruits/vegetables, and price comparisons across different months.
- **Performance Optimization** – flutter_riverpod ensured that data updates were efficient and real-time, improving user experience without unnecessary UI rebuilds.

  By incorporating data visualization, we made it easier for users to make informed decisions about their grocery purchases based on historical price trends.

V) What We Achieved

- Successfully provided users with real-time price trends for fruits and vegetables.
- Enabled users to make data-driven decisions by offering insights into price fluctuations.
- Created a predictive model that helps users anticipate future price hikes or drops based on historical data.
- Enhanced user experience through visual representation of price trends, making complex data easily interpretable.

- **Images**


MandiPedia
Search commodity...
"Do not save what is left after spending, but spend what is left after saving."
MandiPedia
watermelon
Price Trends Over the Year
Watermelon
Aug : 58
Jan Feb Mar Apr May June July Aug Sep Oct Nov Dec
Trends

4. ‘For You’ Screen

I) Collection of Data

The "For You" screen is designed to offer personalized recommendations based on user activity and market trends. To achieve this, we analyse user search history, tracking how frequently a particular fruit or vegetable has been searched. Additionally, we consider sales data, identifying which items have been sold the most across different platforms. By combining these two approaches, we ensure that recommendations are both user-centric and market-driven.

II) Sorting and Conversion of Raw Data into JSON

Once the search history and sales data are collected, they need to be processed for efficient analysis and retrieval. Using Python, we clean and organize the raw data by:

- Filtering out irrelevant or infrequent searches.
- Structuring the search frequency and sales data in a meaningful way.
- Converting the refined data into a JSON file format, ensuring easy integration with our Flutter app.

This structured approach allows the app to efficiently access, update, and utilize user preferences dynamically.

III) Analysing the Top 4 Results in Both Categories

To provide users with the most relevant recommendations, we extract the top 4 most-searched items and the top 4 best-selling items based on the collected data. The recommendations are then categorized into:

1. **"Recommended for You"** – Based on the user's personal search patterns, showing fruits and vegetables they frequently look for.
2. **"Bestsellers"** – Highlighting the most popular items across all users, helping them discover trending or widely purchased products.

By implementing sorting algorithms and ranking techniques, we ensure that users see the most relevant and valuable recommendations at a glance.

IV) What Was Achieved

The "For You" screen significantly enhances the user experience by:

- **Providing Personalized Shopping Suggestions**: Users get recommendations tailored to their interests, reducing search time and making shopping more efficient.
- **Highlighting Market Trends**: The bestsellers section helps users stay updated on popular and cost-effective produce.
- **Improving Decision-Making**: With easy access to frequently searched and top-selling items, users can make informed choices without extra effort.
- **Enhancing Engagement**: Personalization encourages users to interact more with the app, leading to better user retention.

By combining search history analysis, sales data insights, and smart recommendations, the "For You" screen transforms Mandipedia into a more intuitive and user-friendly shopping assistant.

- **Images**

5. Guess the Price

I) Collection of Data from Excel Sheet

For the "Guess the Price" feature, we gathered historical pricing data for various fruits and vegetables over a period of one year. This data was sourced from Kaggle, ensuring accuracy and consistency. The dataset includes:

- **Monthly price trends** for each fruit and vegetable.
- **Seasonal variations** affecting price fluctuations.
- **Inflation-based price changes** over time.

This data serves as the foundation for predicting future prices and enabling users to engage with the price-guessing game.

II) Analysis of Data Using Naïve Bayes Theorem

To make accurate price predictions, we implemented Naïve Bayes, a machine learning algorithm that helps estimate the future price of a selected fruit or vegetable. The model considers:

- **Historical price trends** to understand past fluctuations.
- **Market conditions and seasonal impact** to account for predictable changes.
- **Inflation trends** to estimate price increases.

When a user enters the name of a fruit or vegetable, our model predicts its expected future price using the trained dataset. This prediction helps create a more engaging experience for users, allowing them to guess whether the price will rise or fall.

III) Increase in Points if the User Guesses Correctly

The game rewards users for accurate price predictions. The scoring system works as follows:

- If the user correctly guesses the approximate future price, they earn points.
- The closer their guess is to the predicted inflated price, the more points they receive.
- This gamified approach encourages user engagement and helps users develop a better understanding of price trends in the market.

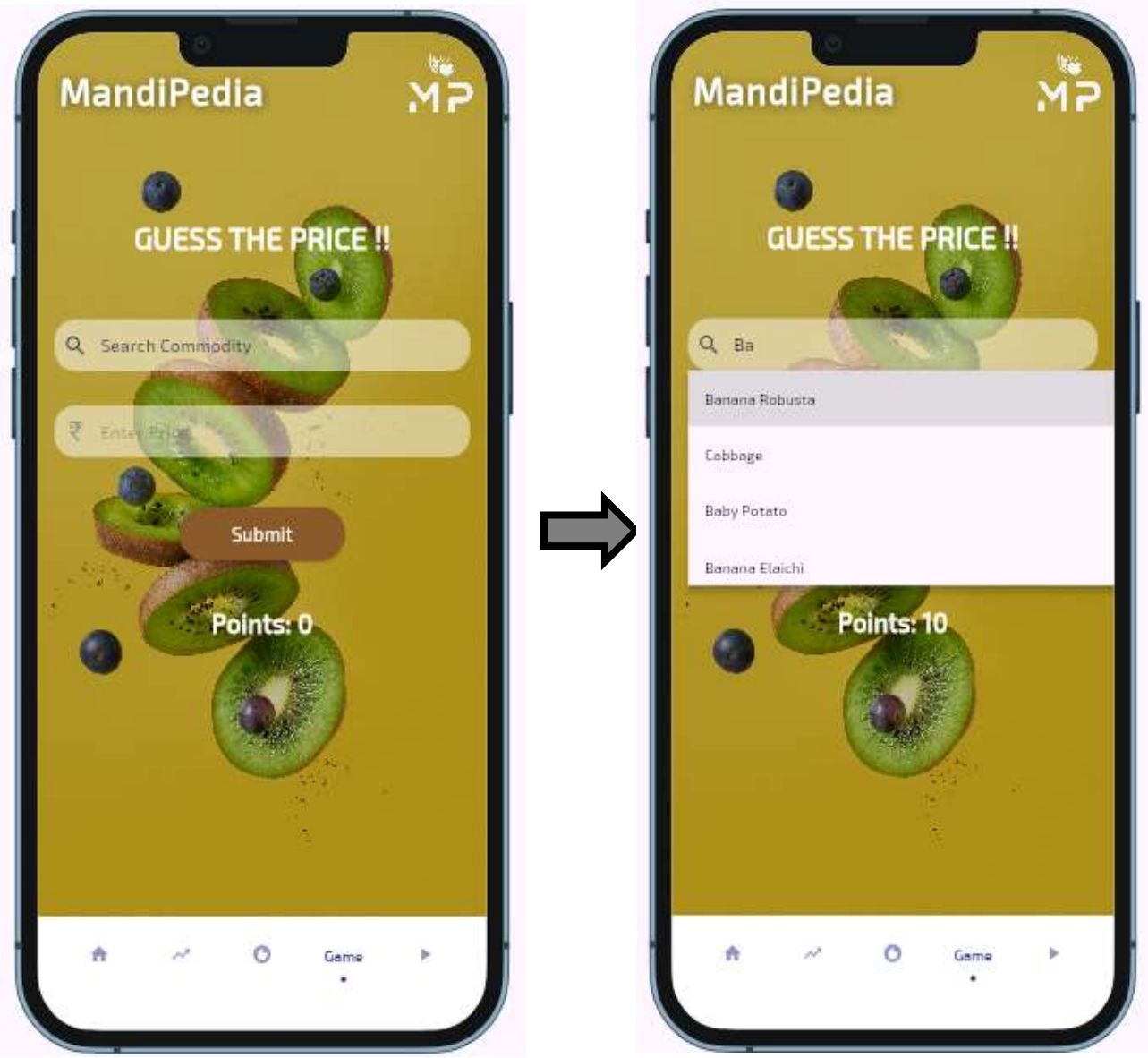
By integrating inflation-based price estimation, the feature makes learning about market trends both fun and educational.

IV) What Was Achieved

The "Guess the Price" game provides several key benefits:

- **Enhanced User Engagement**: Users interact more with the app through an interactive price prediction challenge.
- **Market Awareness**: Users gain better insights into price trends, helping them make informed buying decisions.
- **Gamified Experience**: The points system adds an element of competition, encouraging users to return and play again.
- **Practical Learning**: By predicting price movements, users understand inflation and market patterns, making them more conscious buyers.

- **Images**


MandiPedia
GUESS THE PRICE !!
Search Commodity
Submit
Points: 0
Game
MandiPedia
GUESS THE PRICE !!
Ba
Banana Robusta
Cabbage
Baby Potato
Banana Elaichi
Points: 10
Game

6. FRUVENATOR – A Fun User Interaction Feature

I) Introduction

Inspired by the popular Akinator game, we wanted to introduce an interactive element to Mandipedia. Enter FRUVENATOR—a game where users think of a fruit or vegetable, and the app attempts to guess it by asking a series of targeted questions.

To achieve this, we implemented decision tree-based classification, where each question helps narrow down the possible choices based on distinguishing features like colour, seasonality, taste, and texture. Additionally, reinforcement learning enables the app to improve its accuracy over time by learning from user responses.

This feature increases user engagement while incorporating machine learning-driven interactions into the app.

II) How FRUVENATOR Works

1. **User Thinks of a Fruit or Vegetable**
    - The user selects a mystery fruit or vegetable in their mind**.**
2. **App Asks a Series of Questions**
    - Example questions:
        - Is it typically red in colour?
        - Is it available year-round?
        - Does it have a sour taste?

3. **Decision Tree-Based Guessing**
    - Each response helps eliminate incorrect options and narrow down possibilities.
4. **Final Guess**
    - After a few questions, the app attempts to guess the correct fruit or vegetable**.**
5. **User Feedback for Learning**
    - If the guess is incorrect, the app asks the user to enter the correct answer, improving its accuracy for future predictions.

III) Machine Learning Concepts Used

1. **Decision Tree Classification:**
    - The game uses a decision tree algorithm to determine the most likely fruit or vegetable.
    - Each question acts as a node, splitting possible choices based on yes/no responses.
    - The decision tree keeps refining until it reaches the most probable guess.

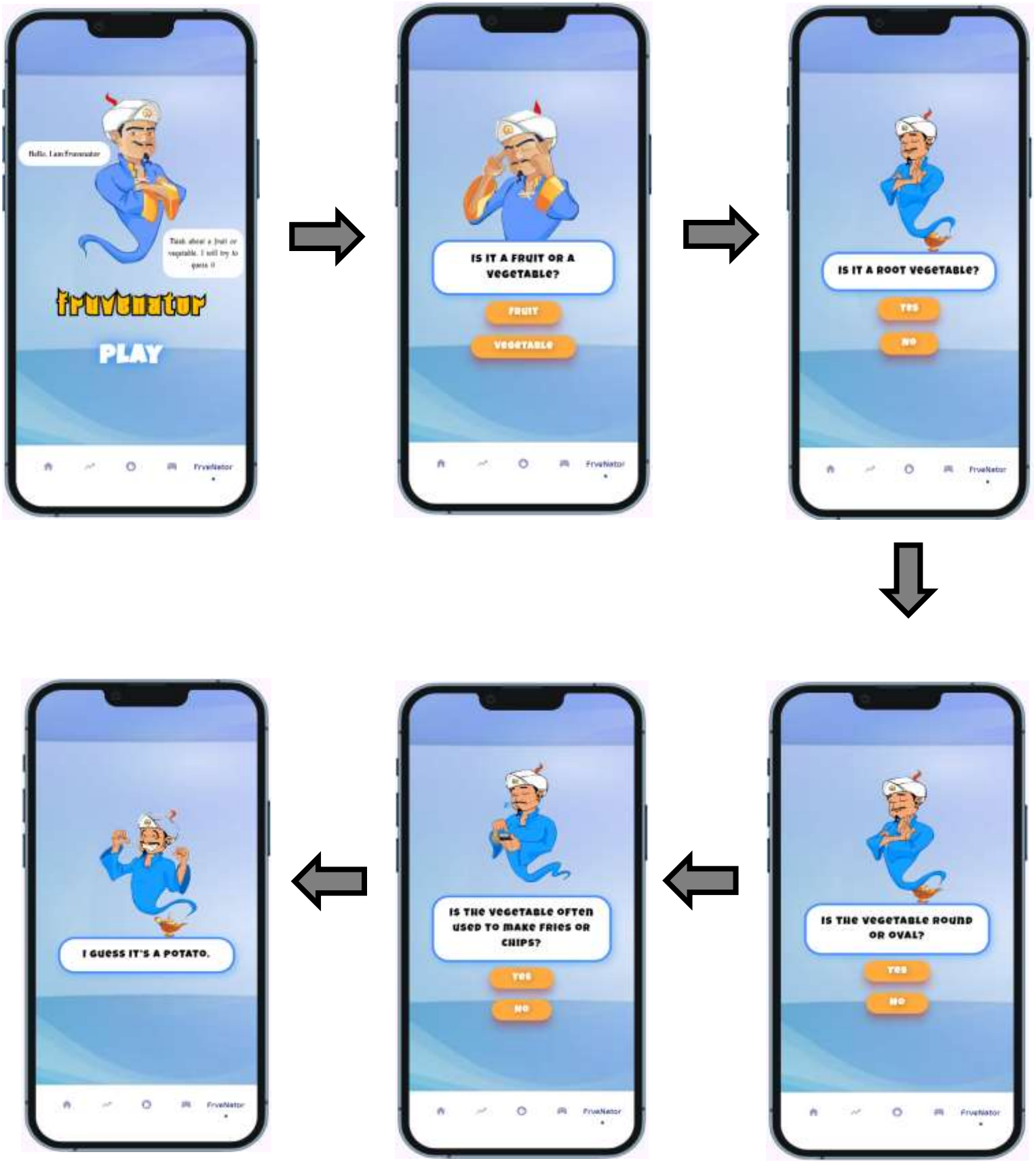
IV) How Data is Stored

1. **Predefined Dataset**: Initially, the app contains a pre-trained database with common fruits and vegetables, along with their characteristics.
2. **User-Generated Inputs**: If the app fails to guess correctly, the user can enter new data, which is stored in a Firebase database or local storage.
3. **Incremental Learning**: New entries refine the decision tree, improving the accuracy of future guesses.

V) What we Achieved

- **Increased User Engagement:** An interactive experience that encourages users to return to the app.
- **Machine Learning Integration:** An AI-powered guessing game that improves over time.
- **Blending Education with Entertainment:** Users learn about different fruits and vegetables in an engaging way.
- **Personalized Experience:** The app adapts based on user responses, making it more accurate over time.

- **Images**


Hello, I am Fruvenator
Think about a fruit or vegetable. I will try to guess it
fruvenator
PLAY
IS IT A FRUIT OR A VEGETABLE?
FRUIT
VEGETABLE
IS IT A ROOT VEGETABLE?
YES
NO
IS THE VEGETABLE ROUND OR OVAL?
YES
NO
IS THE VEGETABLE OFTEN USED TO MAKE FRIES OR CHIPS?
YES
NO
I GUESS IT'S A POTATO.

Errors Encountered

1. Graphical Analysis

- **Error:** Provider Scope Error for Graphical Analysis Screen

```
Another exception was thrown: Bad state: No ProviderScope found
Another exception was thrown: Bad state: No ProviderScope found
```

- **Debug:** By passing ProviderScope in main.dart.

2. Web Scraping Errors

- **Error:** NoSuchElementException (Scraped data was missing due to changing HTML tags).

  Debug: Used inspect element to find updated tag structure.

  Debug: Adjusted XPath/CSS selectors in the script.

- **Error:** requests.exceptions.ConnectionError (Bot protection blocked request).

  Debug: Added headers to mimic a browser request.

- **Error:** requests.exceptions.Timeout (Website took too long to respond).

  Debug: Increased timeout.

- **Error:** selenium.common.exceptions.TimeoutException (JavaScript content did not load in time).

  Debug: Used WebDriverWait to wait for elements to appear.

3. **Flutter Issues**

- **Error:** Version conflicts caused flutter pub get to fail.
  Debug: Ran flutter upgrade to update dependencies.
  Debug: Manually adjusted pubspec.yaml versions.

- **Error:** flutter build failed after Git merge due to dependency mismatch.
  Debug: Ran flutter clean and flutter pub get to refresh dependencies.

4. **Git & GitHub Issues**

- **Error:** Merge conflicts in lib/main.dart, pubspec.yaml, and pubspec.lock.
  Debug: Manually resolved conflicts and committed the merge.

Learnings

Through this project, we gained hands-on experience in full-stack development, including data collection, web scraping, backend integration, and game logic implementation. We learned how to efficiently fetch and process data from multiple sources while handling real-time requests. Additionally, we improved our UI/UX design skills, ensuring a seamless user experience. Debugging challenges and optimizing web scraping techniques helped us develop problem-solving skills. Collaboration played a key role, as we divided tasks effectively and coordinated efforts to complete the project successfully.

References

(1) **Dataset –**

https://www.kaggle.com/

(2) **Best Sellers and Recommended Products -**

https://www.sellerapp.com/blog/find-best-selling-amazon-products/

(3) **Line Graph in Flutter -**

https://youtu.be/zhcxdh4-Jt8?si=XO2gAoppaXeyFxFI

(4) **Web Scraping –**

https://selenium-python.readthedocs.io/

https://www.selenium.dev/documentation/webdriver/

https://devpost.com/software/quick-commerce-data-scraping-for-zepto-and-blinkit-platforms

https://youtu.be/XI5_nsClCYI?si=PCOwEk-Nl6BYzV9K

https://youtu.be/4TUsq4GizXk?si=T2cL9hyZYu0yhvBf

(5) **Fruvenator –**

https://en.akinator.com/

https://www.geeksforgeeks.org/machine-learning-algorithms/

https://www.simplilearn.com/10-algorithms-machine-learning-engineers-need-to-know-article

Individual Contribution Report

1. Vikas Rana (Project Leader) (2305033)

- Designed the Splash Screen and UI for Compare Prices (1st Screen).
- Co-developed Recommended Screen (3rd Screen) UI and functionality.
- Co-developed FruveNator Game (5th Screen), working on UI, logic and designing the decision tree for questions.
- Managed project coordination and overall decision-making.

2. Varun Kumaresan (2305012)

- Developed the backend functionality using Flask API, implemented web scraping and backend-to-frontend integration for Compare Prices (1st Screen).
- Developed UI and logic for Results Screen which is the part of 1st Screen (1.1 Screen).
- Co-developed FruveNator Game (5th Screen), working on UI, logic and designing the decision tree for questions.

3. Jeet Waral (2305070)

- Designed UI and developed the working of Inspect Prices (2nd Screen).
- Created UI and implemented logic for Guess the Price (4th Screen).

4. Alan Cardoza (2305011)

- Collected and prepared data for Inspect Prices (2nd Screen).
- Designed UI and implemented functionality for Recommended Screen (3rd Screen) along with Vikas.
- Assisted in creating the documentation