
Know UR Food

Introduction to Augmented and Virtual Reality (AT62001) Group Project Final Report

Group – 3

Team Members:

Suryam Arnav Kalra

Pramita Kastha

Abhishant Kumar

Shivam Lahoti

Harshith Shivaprakash

Why is there a need for our product?

There are several problems with the traditional restaurant menu, some of which are:

- ✚ Poor visualization: The menu lacks pictures of dishes, making it difficult for customers to visualize what they are ordering. This can lead to disappointment when the dish arrives and does not meet their expectations. Customers often rely on visual cues to make decisions about what to order, and without pictures, they may be less likely to try new dishes or make adventurous choices.
- ✚ Lack of dietary information: Health-conscious customers find it difficult to manage their diet in the absence of nutritional information on the menu. This can result in them avoiding the restaurant altogether or making less healthy choices. Providing detailed nutritional information can help customers make informed decisions about what to eat and can also demonstrate our commitment to promoting healthy eating habits.
- ✚ Lack of engagement: Traditional menus can be monotonous and make ordering food a chore. This can result in a lack of excitement and engagement from customers, leading to a less enjoyable dining experience. By incorporating interactive elements or creative design into our menu, we can create a more engaging experience for customers and encourage them to explore our offerings.
- ✚ Indecision and regret: Confusing menu descriptions may lead to customers making choices they later regret. This can result in dissatisfaction with their meal and a negative overall experience at our restaurant. Clear and concise menu descriptions can help customers make informed decisions and feel confident in their choices.

What problems does our app solve?

To address the problems with our menu, we have developed an app that provides customers with all the information they need to make informed decisions about what to order. The app includes the following features:

- ✚ Calorie counts: Customers can view the calorie count for each dish to help them make healthy choices. This information is clearly displayed and easy to understand, allowing customers to quickly compare the calorie content of different dishes and make informed decisions about what to eat. By providing this information, we are empowering our customers to take control of their dietary choices and make decisions that align with their health goals.
- ✚ Ingredients and components: The app provides detailed information about the ingredients and components of each dish, allowing customers to check for allergens or dietary

restrictions. This information is presented in a clear and concise manner, making it easy for customers to quickly identify any ingredients they need to avoid. By providing this information, we are helping our customers make safe and informed choices about what they eat.

- 🍴 **Plating:** Customers can visualize their food before it is served, helping to manage their expectations and avoid disappointment. The app includes high-quality images of each dish, allowing customers to see exactly what their food will look like when it arrives at the table. This feature adds an element of excitement and anticipation to the ordering process and helps ensure that our customers are satisfied with their choices.
- 🍴 **Fun facts and stories:** The app includes fun facts and stories about each dish, adding an element of engagement and entertainment to the ordering process. These stories provide interesting background information about the origins and history of each dish, helping customers feel more connected to the food they are eating. By providing this information, we are creating a more engaging and enjoyable dining experience for our customers.
- 🍴 **Nutritional value:** In addition to calorie counts, the app provides detailed nutritional information for each dish, helping health-conscious customers make informed decisions. This information includes not only calorie content but also other important nutritional data such as fat, protein, and carbohydrate content. By providing this information, we are helping our customers make informed choices about what they eat and supporting their efforts to maintain a healthy diet.

A brief market analysis

The Indian food service market is expected to register a CAGR of 10.51% over the forecast period. During the COVID-19 pandemic, India's food service market saw a sharp decline in revenues for street food vendors and food service companies due to strict lockdowns enforced by the government¹. According to the National Restaurant Association of India (NRAI), 25% of restaurants closed shop completely in FY21, resulting in over 2.3 million jobs lost¹. However, the Indian food service market is one of those vibrantly growing markets that has seen exceptional growth during the past decade and is expected to expand rapidly in the upcoming five years of the forecast period.

A high percentage of the young and working population is driving the Indian food service market. The availability of organized retail space aids the industry in encouraging the growth of local and international brands across different formats. Additionally, plant-based foods are undergoing a surge in growth around the country, owing to the increasing consumer demand for healthy foods that are also environmentally sustainable. Foodservice outlets are beginning to offer products that capitalize on this interest in vegan and vegetarian products in India, which in turn is set to boost the growth of the food service market in the country.

Technical description, system architecture, codes

Mainmenu.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class Main_Menu : MonoBehaviour
{
    public void NextScene()
    {
        SceneManager.LoadScene(1);
    }
}
```

Firebase databasemanager.cs

```
using Firebase.Database;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class FirebaseDatabaseManager : MonoBehaviour
{
    DatabaseReference reference;

    void Start()
    {
        Firebase.AppOptions options = new Firebase.AppOptions();
        options.ApiKey = "AIzaSyDu7q0E6j729r_EXjx7s0IOu0K-stChIiQ";
        options.AppId = "1:516917387431:android:0e9342e8c165f6647d7b82";
        options.ProjectId = "fir-demo-c38ae";
        options.StorageBucket = "fir-demo-c38ae.appspot.com";
        options.DatabaseUrl = new System.Uri("https://fir-demo-c38ae-default-rtdb.firebaseio.com/");

        var app = Firebase.FirebaseApp.Create(options);
        reference = FirebaseDatabase.DefaultInstance.RootReference;
    }

    public void CreateNewUser(string userId, string value)
    {
        reference.Child("users").Child(userId).SetValueAsync(value);
        Debug.Log("New User Created");
    }
}
```

Inputscript.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;

public class Inputscript : MonoBehaviour
{
    // Start is called before the first frame update
    public TMP_InputField keyIF;
    public TMP_InputField value_IF;

    public FirebaseDatabaseManager firebaseDatabaseManagerScript;

    public void GetMessage()
    {
        firebaseDatabaseManagerScript.CreateNewUser(keyIF.text, value_IF.text);
    }
}
```

```
    }  
}
```

Nav_menu.cs

```
using System.Collections;  
using System.Collections.Generic;  
using UnityEngine;  
using UnityEngine.SceneManagement;  
  
public class Nav_Menu : MonoBehaviour  
{  
    public void Next_Scene()  
    {  
        int curr = SceneManager.GetActiveScene().buildIndex;  
        int next = curr + 1;  
        SceneManager.LoadScene(next);  
    }  
    public void Prev_Scene()  
    {  
        int curr = SceneManager.GetActiveScene().buildIndex;  
        int prev = curr - 1;  
        SceneManager.LoadScene(prev);  
    }  
}
```

Order.cs

```
using System.Collections;  
using System.Collections.Generic;  
using UnityEngine;  
using TMPro;  
  
public class Order : MonoBehaviour  
{  
    // Start is called before the first frame update  
    int burger = 0;  
    int frenchfries = 0;  
    int kashipan = 0;  
    int tofu = 0;  
    int sushi = 0;  
  
    public TMP_Text burgerText;  
    public TMP_Text FFText;  
    public TMP_Text kpText;  
    public TMP_Text tofuText;  
    public TMP_Text sushiText;  
  
    private void ChangeBurgerText() {  
        burgerText.text = "Quantity: " + burger.ToString();  
    }  
  
    private void ChangeFFText() {  
        FFText.text = "Quantity: " + frenchfries.ToString();  
    }  
  
    private void ChangeKPText() {  
        kpText.text = "Quantity: " + kashipan.ToString();  
    }  
  
    private void ChangeTofuText() {  
        tofuText.text = "Quantity: " + tofu.ToString();  
    }  
}
```

```

private void ChangeSushiText() {
    sushiText.text = "Quantity: " + sushi.ToString();
}

public void addBurger() {
    burger += 1;
    ChangeBurgerText();
    OrderDetails.burger = burger;
}

public void removeBurger() {
    burger -= 1;
    if(burger < 0) {
        burger = 0;
    }
    ChangeBurgerText();
    OrderDetails.burger = burger;
}

public void addFF() {
    frenchfries += 1;
    ChangeFFText();
    OrderDetails.ff = frenchfries;
}

public void removeFF() {
    frenchfries -= 1;
    if(frenchfries < 0) {
        frenchfries = 0;
    }
    ChangeFFText();
    OrderDetails.ff = frenchfries;
}

public void addKashipan() {
    kashipan += 1;
    ChangeKPText();
    OrderDetails.kp = kashipan;
}

public void removeKashipan() {
    kashipan -= 1;
    if(kashipan < 0) {
        kashipan = 0;
    }
    ChangeKPText();
    OrderDetails.kp = kashipan;
}

public void addtofu() {
    tofu += 1;
    ChangeTofuText();
    OrderDetails.tofu = tofu;
}

public void removetofu() {
    tofu -= 1;
    if(tofu < 0) {
        tofu = 0;
    }
    ChangeTofuText();
    OrderDetails.tofu = tofu;
}

public void addsushi() {
    sushi += 1;
    ChangeSushiText();
    OrderDetails.sushi = sushi;
}

public void removesushi() {
    sushi -= 1;
}

```

```

        if(sushi < 0) {
            sushi = 0;
        }
        ChangeSushiText();
        OrderDetails.sushi = sushi;
    }

    public int getBurger() {
        return burger;
    }

    public int getFrenchFries() {
        return frenchfries;
    }

    public int getKashipan() {
        return kashipan;
    }

    public int getTofu() {
        return tofu;
    }

    public int getSushi() {
        return sushi;
    }
}
Orderdetails.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class OrderDetails : MonoBehaviour
{
    public static int burger;
    public static int ff;
    public static int kp;
    public static int sushi;
    public static int tofu;
    public static int userid;

    void Start() {
        userid = 0;
    }
}
PanelToggler.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PanelToggler : MonoBehaviour
{
    bool isOn = false;

    [SerializeField] GameObject InfoPanel;
    [SerializeField] GameObject AddToOrderBtn;
    [SerializeField] GameObject RemoveFromOrderBtn;
    [SerializeField] GameObject Quantity;

    public void togglePanel()
    {
        if(isOn)
        {

```

```

        InfoPanel.SetActive(false);
        AddToOrderBtn.SetActive(true);
        RemoveFromOrderBtn.SetActive(true);
        Quantity.SetActive(true);
        isOn = false;
    }
    else
    {
        InfoPanel.SetActive(true);
        AddToOrderBtn.SetActive(false);
        RemoveFromOrderBtn.SetActive(false);
        Quantity.SetActive(false);
        isOn = true;
    }
}
}
RotateModel.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PanelToggler : MonoBehaviour
{
    bool isOn = false;

    [SerializeField] GameObject InfoPanel;
    [SerializeField] GameObject AddToOrderBtn;
    [SerializeField] GameObject RemoveFromOrderBtn;
    [SerializeField] GameObject Quantity;

    public void togglePanel()
    {
        if(isOn)
        {
            InfoPanel.SetActive(false);
            AddToOrderBtn.SetActive(true);
            RemoveFromOrderBtn.SetActive(true);
            Quantity.SetActive(true);
            isOn = false;
        }
        else
        {
            InfoPanel.SetActive(true);
            AddToOrderBtn.SetActive(false);
            RemoveFromOrderBtn.SetActive(false);
            Quantity.SetActive(false);
            isOn = true;
        }
    }
}
}
SceneManager.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class SceneManager : MonoBehaviour
{
    public void MainMenu()
    {
        SceneManager.LoadScene("MainMenu");
    }
}

```



```

public void ARScene()
{
    OrderDetails.burger = 0;
    OrderDetails.ff = 0;
    OrderDetails.kp = 0;
    OrderDetails.sushi = 0;
    OrderDetails.tofu = 0;
    SceneManager.LoadScene("ARScene");
}

public void FinalScene() {
    SceneManager.LoadScene("FinalScene");
}

public void Quit()
{
    Application.Quit();
}
}
SliderControl.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class sliderControl : MonoBehaviour
{
    public Slider scaleSlider;
    public Slider rotateSliderX;
    public Slider rotateSliderY;
    public Slider rotateSliderZ;

    public float rotMinValueX;
    public float rotMaxValueX;
    public float rotMinValueY;
    public float rotMaxValueY;
    public float rotMinValueZ;
    public float rotMaxValueZ;
    public float scaleMinValue;
    public float scaleMaxValue;
    // Start is called before the first frame update
    void Start()
    {
        //scale slider
        scaleSlider.minValue = scaleMinValue;
        scaleSlider.maxValue = scaleMaxValue;

        scaleSlider.onValueChanged.AddListener(ScaleSliderUpdate);

        //rotate axis wise slider
        rotateSliderX.minValue = rotMinValueX;
        rotateSliderX.maxValue = rotMaxValueX;

        rotateSliderX.onValueChanged.AddListener(RotateSliderUpdateX);

        rotateSliderY.minValue = rotMinValueY;
        rotateSliderY.maxValue = rotMaxValueY;

        rotateSliderY.onValueChanged.AddListener(RotateSliderUpdateY);
    }
}

```

```

        rotateSliderZ.minValue = rotMinValueZ;
        rotateSliderZ.maxValue = rotMaxValueZ;

        rotateSliderZ.onValueChanged.AddListener(RotateSliderUpdateZ);
    }

    void ScaleSliderUpdate(float value)
    {
        transform.localScale = new Vector3(value, value, value);
    }
    void RotateSliderUpdateX(float value)
    {
        transform.localEulerAngles = new Vector3(value, transform.rotation.y,
transform.rotation.z);
    }
    void RotateSliderUpdateY(float value)
    {
        transform.localEulerAngles = new Vector3(transform.rotation.x, value,
transform.rotation.z);
    }

    void RotateSliderUpdateZ(float value)
    {
        transform.localEulerAngles = new Vector3(transform.rotation.x,
transform.rotation.y, value);
    }

    // Update is called once per frame
    void Update()
    {

    }
}
viewAndplaceOrders.cs
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPPro;
using Firebase.Database;

public class ViewAndPlaceOrder : MonoBehaviour
{
    public TMP_Text orderDetailsText;
    public TMP_Text placeOrderText;
    int userid = 0;

    string userId;
    DatabaseReference reference;

    void Start()
    {
        Firebase.AppOptions options = new Firebase.AppOptions();
        options.ApiKey = "AIzaSyDu7q0E6j729r_EXjx7s0IOu0K-stChIiQ";
        options.AppId = "1:516917387431:android:0e9342e8c165f6647d7b82";
        options.ProjectId = "fir-demo-c38ae";
        options.StorageBucket = "fir-demo-c38ae.appspot.com";
        options.DatabaseUrl = new System.Uri("https://fir-demo-c38ae-default-
rtbd.firebaseio.com/");

        var app = Firebase.FirebaseApp.Create(options);
        reference = FirebaseDatabase.DefaultInstance.RootReference;
    }
}

```

```

}

public void ViewOrder() {
    int burger = OrderDetails.burger;
    int ff = OrderDetails.ff;
    int kp = OrderDetails.kp;
    int sushi = OrderDetails.sushi;
    int tofu = OrderDetails.tofu;

    orderDetailsText.text = "Order Details\n";
    if(burger > 0) {
        orderDetailsText.text += "\nBurger : " + burger.ToString();
    }
    if(ff > 0) {
        orderDetailsText.text += "\nFrench Fries : " + ff.ToString();
    }

    if(kp > 0) {
        orderDetailsText.text += "\nKashipan : " + kp.ToString();
    }

    if(sushi > 0) {
        orderDetailsText.text += "\nSushi : " + sushi.ToString();
    }

    if(tofu > 0) {
        orderDetailsText.text += "\nTofu : " + tofu.ToString();
    }

    int price = burger*80 + ff*120 + kp*285 + sushi*375 + tofu*225;
    Debug.Log("price:" + price);
    orderDetailsText.text += "\nPrice : " + price.ToString();
}

public void PlaceOrder() {
    placeOrderText.text = "Thanks! Your order is placed";
    placeOrderText.color = Color.blue;
    userid = OrderDetails.userid;
    userId = userid.ToString();
    userid += 1;
    OrderDetails.userid += 1;
    Debug.Log("order in placeorder function");
    addOrderToDatabase();
}

public void addOrderToDatabase() {
    int burger = OrderDetails.burger;
    int ff = OrderDetails.ff;
    int kp = OrderDetails.kp;
    int sushi = OrderDetails.sushi;
    int tofu = OrderDetails.tofu;
    Debug.Log(ff);
    Debug.Log(burger);
    reference.Child("orders")
        .Child(userId)
        .Child("Burger")
        .SetValueAsync(burger);
}

```

```

        reference.Child("orders")
            .Child(userId)
            .Child("French Fries")
            .SetValueAsync(ff);

        reference.Child("orders")
            .Child(userId)
            .Child("Kashipan")
            .SetValueAsync(kp);

        reference.Child("orders")
            .Child(userId)
            .Child("Sushi")
            .SetValueAsync(sushi);


        reference.Child("orders")
            .Child(userId)
            .Child("Tofu")
            .SetValueAsync(tofu);


        Debug.Log("order in adding to db");
    }
}

```

Who is your target audience, and in what domain?

Let us make clear our target audience with the help of some such user stories:

-  Priya Patel is a health-conscious individual who is always careful about what she eats. She needs detailed nutritional information about her meals and is selective about the ingredients used in her dishes. With our app, Priya can easily access all the information she needs to make informed decisions about what to eat. She can view calorie counts, ingredients, and nutritional value for each dish on the menu, helping her make healthy choices that align with her dietary goals. By providing this information, our app empowers Priya to take control of her dietary choices and make decisions that support her health and wellbeing.

-  Rohit Sharma loves trying new cuisines and visiting exotic restaurants. However, he often gets confused by the fancy names of dishes and is not sure what to order. With our app, Rohit can easily access clear and concise descriptions of each dish on the menu, along with pictures and other visual aids to help him make informed decisions about what to order. He can also learn fun facts and stories about each dish, adding an element of engagement and entertainment to his dining experience. By providing this information, our app helps Rohit navigate the sometimes-confusing world of exotic cuisine and make choices that he will enjoy.

Design principles/features involved in the developed AR/VR app.

- ✚ One of the most exciting features of the app is the inclusion of 3D models for each food item on the menu. This allows users to see what their meal will look like before they order, adding an extra level of interactivity and engagement. The 3D models are highly detailed and realistic, giving users a true-to-life representation of the food they are considering.
- ✚ Each menu item has a QR code that can be scanned. This allows users to quickly and easily access details about the food they are interested in, without having to navigate through multiple screens or menus. The QR codes are conveniently located and easy to scan, making it simple for users to get the information they need.
- ✚ Another great feature of the app is the inclusion of sliders and rotators to adjust the size and angle of the overlaid food image. This gives users more control over their experience and allows them to customize the way they view their food. The sliders and rotators are intuitive and easy to use, making it simple for users to make adjustments on the fly.
- ✚ The app also includes quantity buttons to make it easy to adjust the quantity of an order. This feature is particularly useful for group orders or for users who want to order multiple items at once. The quantity buttons are clearly labeled and easy to use, ensuring that users can quickly and easily make changes to their order.
- ✚ To improve user experience and reduce clutter, the app has multiple screens. This helps to keep the app organized and easy to navigate, even as more features and options are added. The screens are well-designed and logically arranged, making it easy for users to find what they are looking for.
- ✚ One of the most convenient features of the app is the ability for users to view and place orders directly through the app. This streamlines the ordering process and makes it more convenient for users to get the food they want. The ordering process is simple and straightforward, with clear instructions and prompts to guide users every step of the way.
- ✚ Finally, the app is integrated with a database, so orders are stored and persisted in the firebase database. This ensures that user data is securely stored and can be easily accessed for future orders. The database integration also allows for advanced features such as order history and personalized recommendations.

Conduct user-experience research on your app.

On opening the app, there is one screen with the start button in the middle of it, on clicking the start button we go to the next screen, which uses the camera on our phone to scan for the QR codes for each dish which is given in the restaurant menu. The 3d model of the dish gets overlayed over the QR code inside our phones, sliders for scaling and rotation of the model appears on the top right side of the screen, meanwhile options for adding/removing the dish and setting the quantity of order appears as buttons on the bottom right. On the bottom left there is an info button for displaying the nutritional information, on clicking it the info appears near the bottom right of the screen. On the top left there are 2 arrow keys, one for going back to the previous (start screen) and one for going to the next screen. In the next, i.e the final screen there are two buttons (view order and place order), near the middle, and the order summary and the total price is also displayed near the middle. On the top left there is a back arrow key for going back to the previous screen.

However, the app could also benefit from some improvements, such as:

- ✚ Adding more options for customization, such as choosing different sauces, toppings, or sides for the dishes. This would allow users to personalize their orders and cater to their preferences and dietary needs. For example, users could choose between spicy or mild sauce, cheese or no cheese, fries or salad, etc.
- ✚ Adding more feedback or confirmation messages, such as showing a pop-up or a sound when a dish is added or removed from the order, or when the order is placed successfully. This would provide users with a sense of progress and assurance that their actions are registered and executed correctly. For example, users could see a message like “Added to order” or “Order placed” along with a check mark or a ding sound.
- ✚ Adding more features or incentives to encourage user engagement, such as ratings, reviews, rewards, or recommendations for the dishes or the restaurants. This would motivate users to use the app more frequently and share their opinions and experiences with other users and the restaurants. For example, users could rate and review the dishes they ordered, earn points or coupons for using the app, or get suggestions for other dishes or restaurants they might like based on their previous orders.

SWOT analysis.

The SWOT analysis of our AR food app reveals the following:

- ✚ **Strengths:** Our app has a unique and innovative concept that allows users to see how their food will look like before ordering. This feature gives our app a competitive edge over other online food delivery platforms that only show pictures or descriptions of the food. It also helps users to make better and faster decisions, as well as to avoid disappointment or dissatisfaction with their orders. Our app also has a user-friendly interface that makes it easy to navigate and order. Our app has a simple and intuitive design that guides users through the process of selecting, viewing, and ordering their food. Our app also has a low cost of development and maintenance, as it uses existing AR technology and cloud computing services. This helps us to save money and resources, as well as to offer affordable prices to our customers. Our app also caters to a high demand for online food delivery services, especially in the post-pandemic era. Our app provides convenience, safety, and variety to our customers, who can order food from anywhere and anytime, without having to worry about health risks or limited options.

Weaknesses: Our app faces some technical challenges, such as glitches, bugs, and compatibility issues with some devices. These problems could affect the quality and reliability of our app, as well as the satisfaction and loyalty of our customers. We need to constantly test, update, and improve our app to ensure its smooth and optimal functioning. We also need to ensure that our app is compatible with different types of devices, operating systems, and browsers. Our app also has limited features compared to some of our competitors, such as customization options, loyalty programs, and social media integration. These features could enhance the value and attractiveness of our app, as well as the engagement and retention of our customers. We need to research and develop more features and functionalities for our app that could meet the needs and preferences of our target market. Our app also faces a high level of competition from other online food delivery platforms, such as Zomato, Swiggy, Uber Eats, etc. These platforms have established brands, large customer bases, wide networks of restaurants, and diverse offerings. We need to differentiate our app from these competitors by highlighting our unique value proposition and competitive advantages. We also need to monitor and analyze the market trends and customer feedback to stay ahead of the competition. Furthermore, our app also faces potential legal issues regarding the use of AR technology and customer data for online food delivery services. These issues could arise from the regulations or policies of different governments or organizations that could limit or ban the use of AR technology or customer data for commercial purposes. We need to comply with the laws and regulations of each country or region where we operate or plan to operate. We also need to ensure that we protect the privacy and security of our customers and their personal information.

Opportunities: Our app has a lot of potential to grow and expand to new markets, both locally and globally. We can collaborate with more restaurants and cuisines to offer a wider variety of choices to our users. We can also target new segments of customers who are interested in online food delivery services or AR technology. We can also explore new regions or countries where online food delivery services or AR technology are in high demand or have low penetration rates. We can also add more features and functionalities to our app, such as ratings, reviews, recommendations, coupons, and gamification. These features could increase the value and attractiveness of our app, as well as the engagement and retention of our customers. They could also help us to collect more data and feedback from our customers that could help us to improve our app and service quality. We can also improve the user experience by enhancing the quality and realism of the AR images, as well as the speed and accuracy of the delivery service. These improvements could increase the satisfaction and loyalty of our customers, as well as the reputation and trustworthiness of our app. Furthermore, we can leverage marketing and promotions strategies to increase our brand awareness and customer base. We can use various channels and platforms such as social media, blogs, podcasts, videos, etc., to showcase our app and its features. We can also use various methods such as advertisements, contests, referrals, etc., to attract more users to our app. We can also partner with influencers and celebrities who have large followings or credibility in the online food delivery or AR technology domains. They could endorse our app and influence more people to try it out.

Threats: Our app faces some external risks that could affect its performance and profitability. These include regulatory changes that could limit or ban the use of AR technology or

customer data for online food delivery services. These changes could occur due to various reasons such as privacy concerns, security