FoodZilla

This is a unique food delivery app , this consists of otp verification system , it also has regular updates which will be pushed to keep the app bugs free.This updates will be displayed on the profile page as a message.It has also has captcha system to verify its a human and reduce spamming. When the user tries changing the password from another device the device which is already having that details will get auto logged out from that device.
[Scopes and Expectation From App]

Group member Details :

	Name	Roll No.	Stream	Worked on
1	Jugal Rajeshbhai Shah	AU1940316	ME $2nd$ Yr	Food Zilla App
2	Deep Patel	AU1940092	ME $2nd$ Yr	Bug System App , ER Diagram , Relational Table .
3	DHAVALKUMAR HARSH…	AU1940180	CSE $2nd$ Yr	Backend part (Data Managing , Procedures, Triggers) and Table Design

Visit Latest Commit in Frontend Code: ShahJugalR/FoodZilla (github.com)

Click Here to Download Latest Build of FoodZilla app and Bug Reporter(Separately)

[FoodZilla v0.3*b*.zip]	[FoodZilla Bugs Report Section]

Technologies Used:-

Frontend	Backend
.NET 4.72	SQL Server Hosted on Azure Server

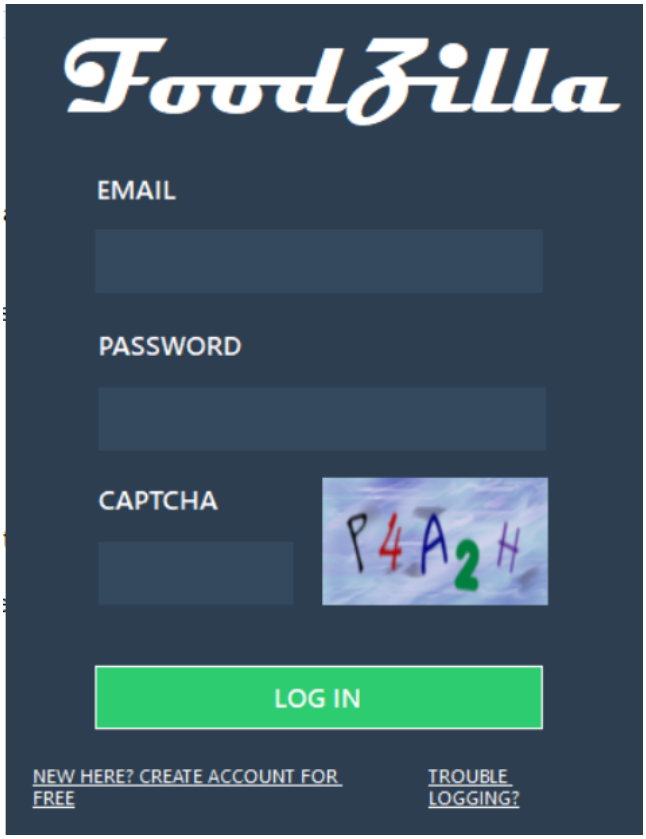
Minimum Specifications Requirement:-

Operating System	Windows 7 Ultimate
Disk Space	~200 MB of free space.
CPU	1GHz, 4MB Cache(Pentium Gold 6500Y, A6 Radeon or M1)
GPU	No Dedicated GPU Needed Integrated will work well.(Intel® UHD Graphics 615, Basic AMD Graphics)
Memory Resource(RAM)	~1 GB
Architecture	x64, x86 or ARM
Internet Connectivity	2 megabytes per sec connection
Pre-Installed Framework	.NET 4.72

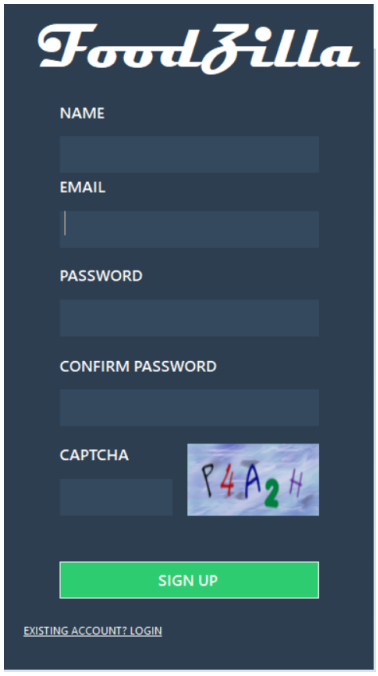
Application Snapshots and detail

Sign Up/Sign In

- If you are opening application for first time you will be required to Sign in

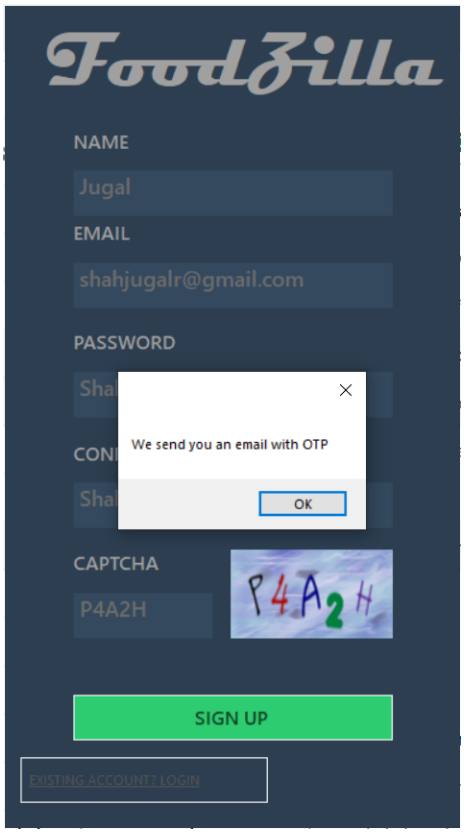


- You can Sign in also make account or proceed to recover forgotten credentials

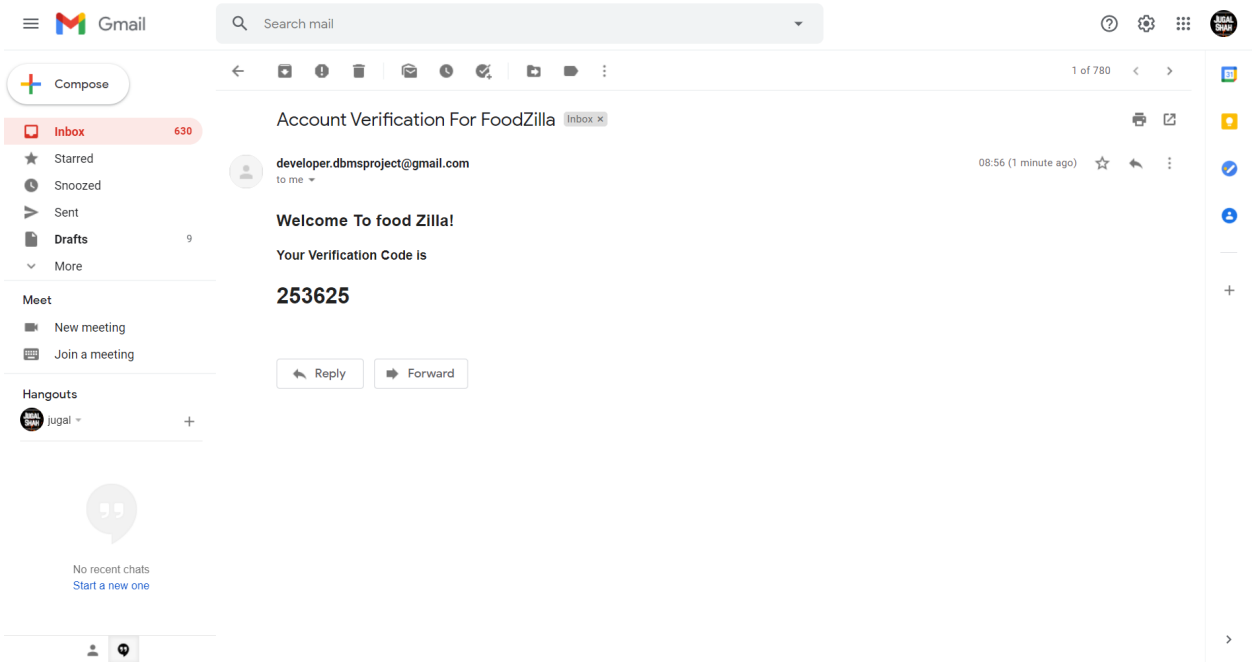


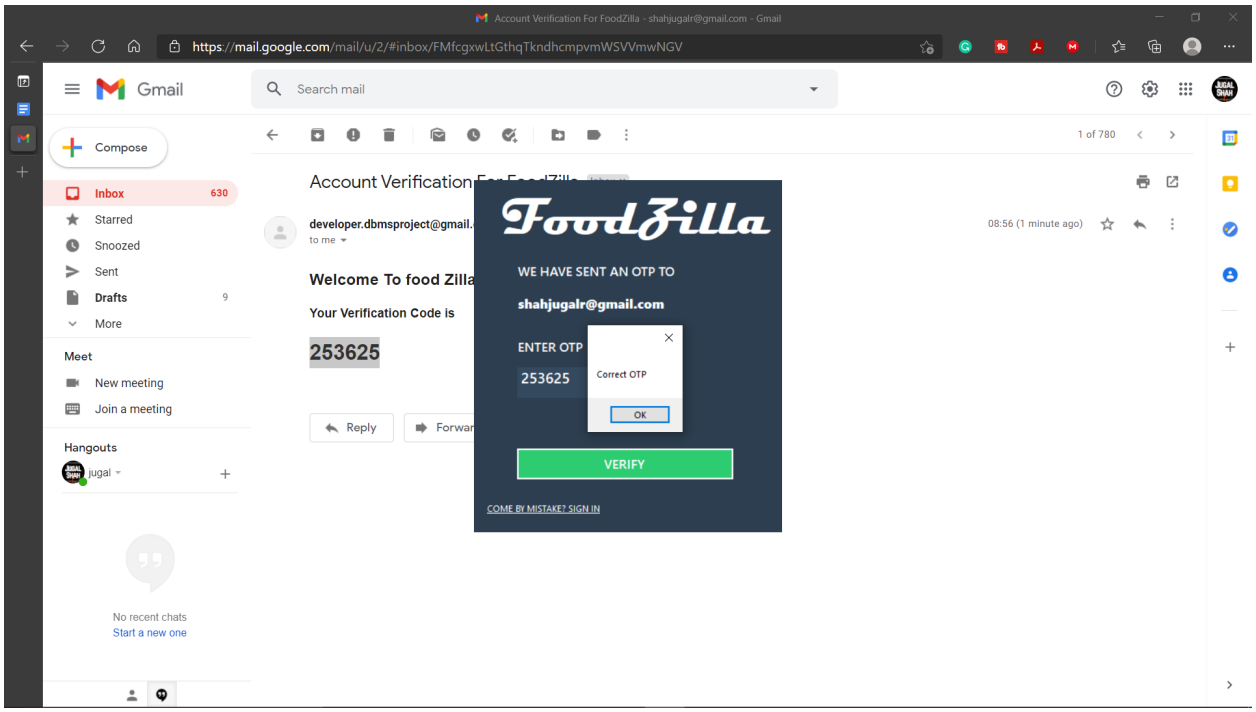
Here You can fill out details for your account

We added captcha to reduce spamming



Our frontend is capable of sending OTPs independently

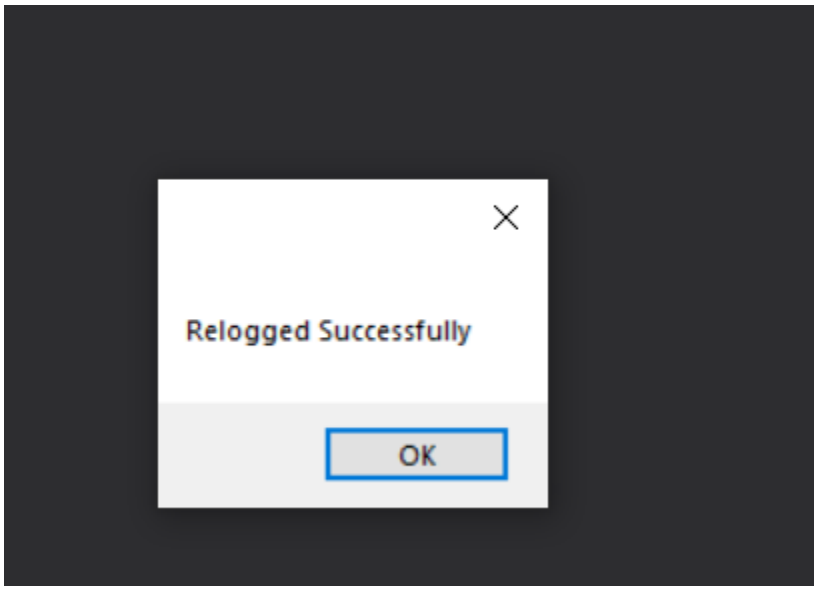




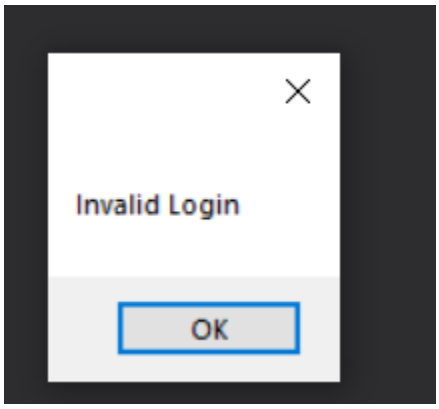
SAVED LOGIN FEATURE

Once User is Created He/She will be Signed in. And Next time when he/she will open app his/her credentials will be saved and let us suppose if she updates his/her password she will be auto logged out from this app.

Relogging success



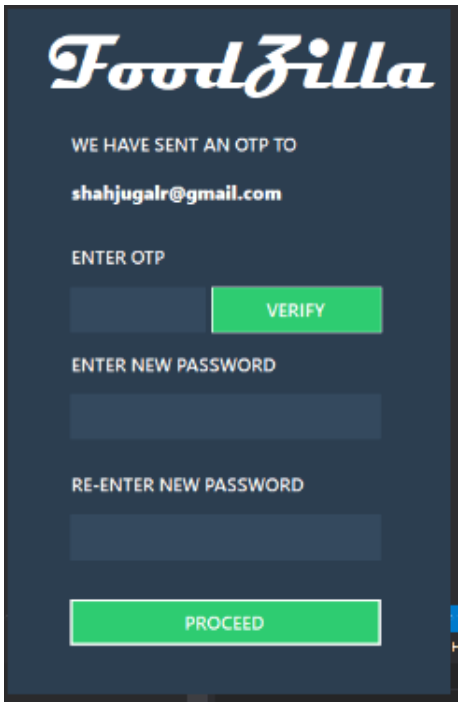
Else



And User will be logged out

FORGOTTEN CREDENTIALS


FoodZilla
EMAIL
REQUEST OTP
CAME BY MISTAKE? SIGN IN


FoodZilla
WE HAVE SENT AN OTP TO
shahjugalr@gmail.com
ENTER OTP
VERIFY
ENTER NEW PASSWORD
RE-ENTER NEW PASSWORD
PROCEED

Welcome To food Zilla!

Your Verification Code is

253625

---

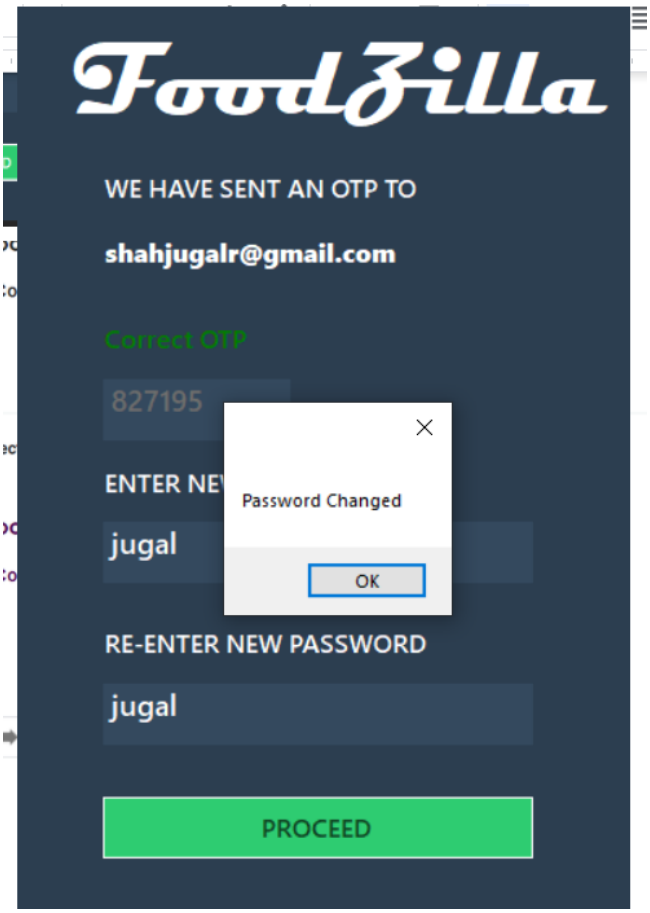
developer.dbmsproject@gmail.com
to me

Welcome To food Zilla!

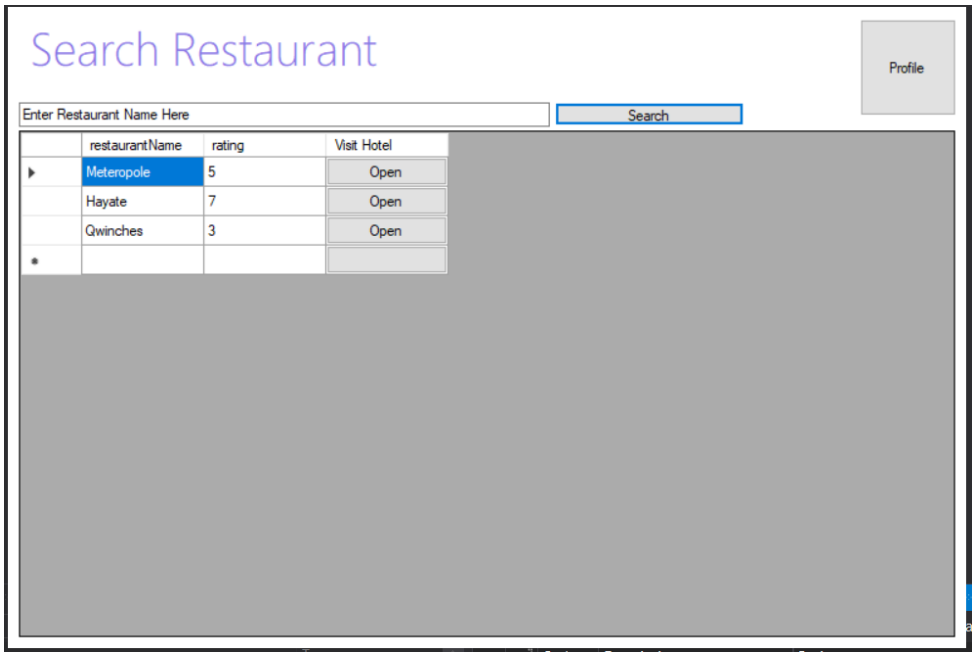
Your Verification Code is

827195

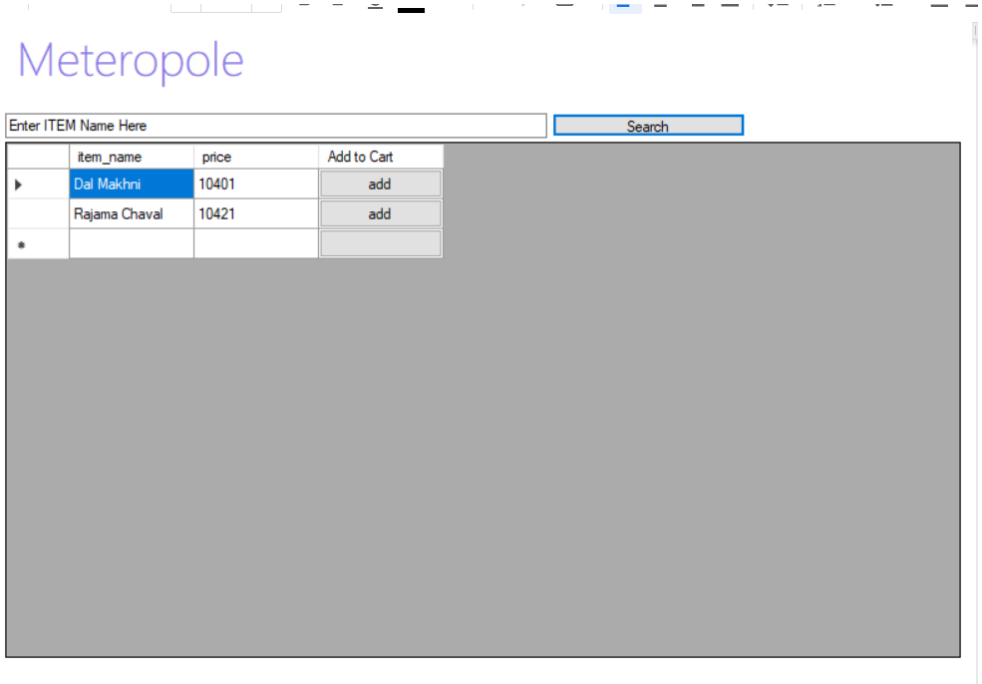
Reply Forward



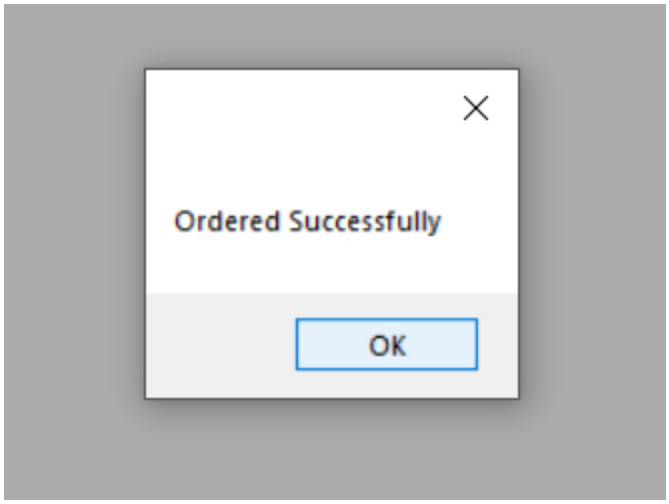
Now once you signed in the app u will be able to see our restaurant selection screen



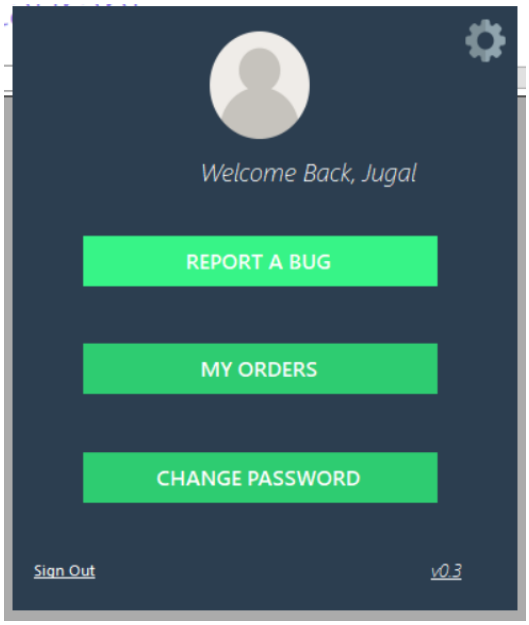
Now U can open any restaurant

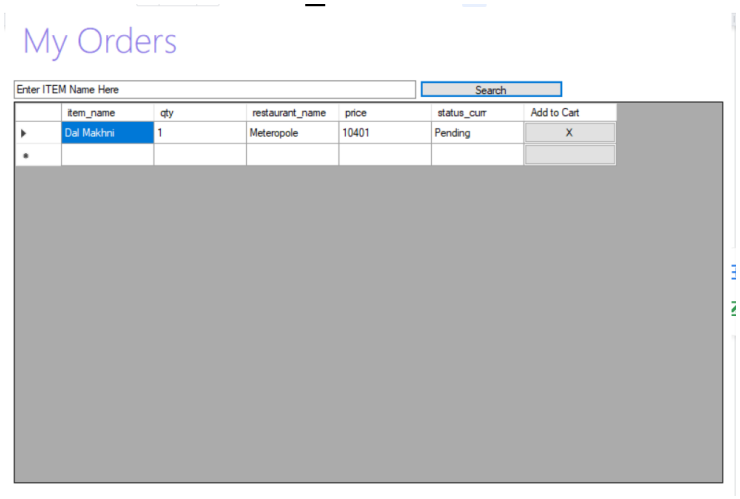


Now you can add to cart or direct order



Once u ordered u can track it on my order’s section homepage> my profile > my orders

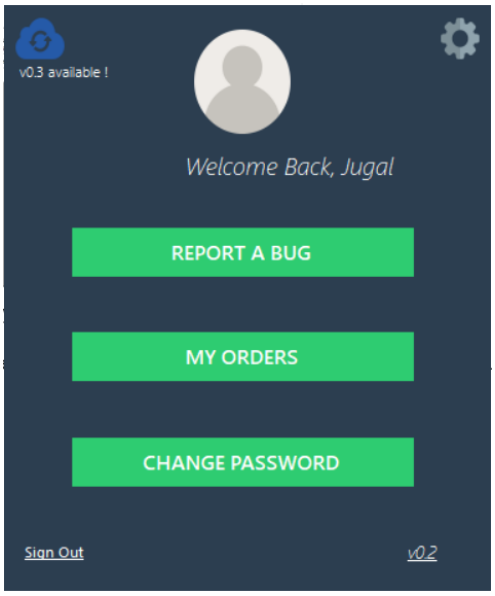




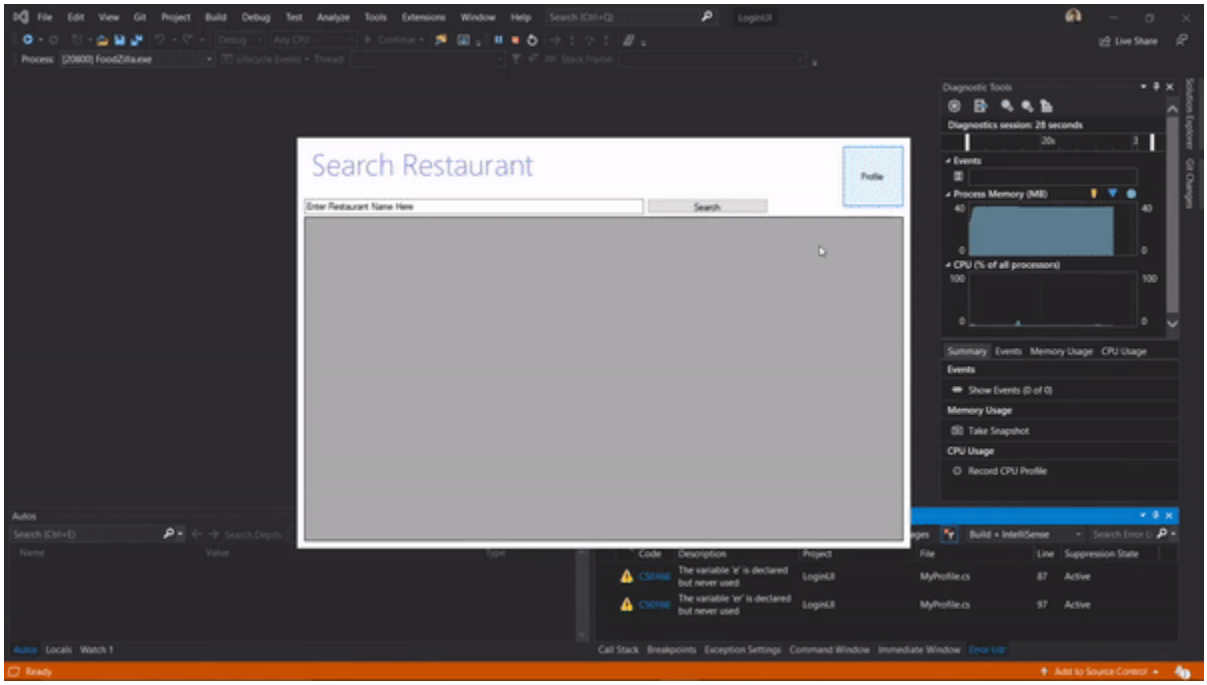
You can also cancel your order before its arrival itself.Also u can track status of your order As :
0 Order Pending
1 Order Confirmed
2 Accepted by restaurant
3 Preparing food
4 Packing
5 On the way
6 Delivered
7 Cancelled

App Update System

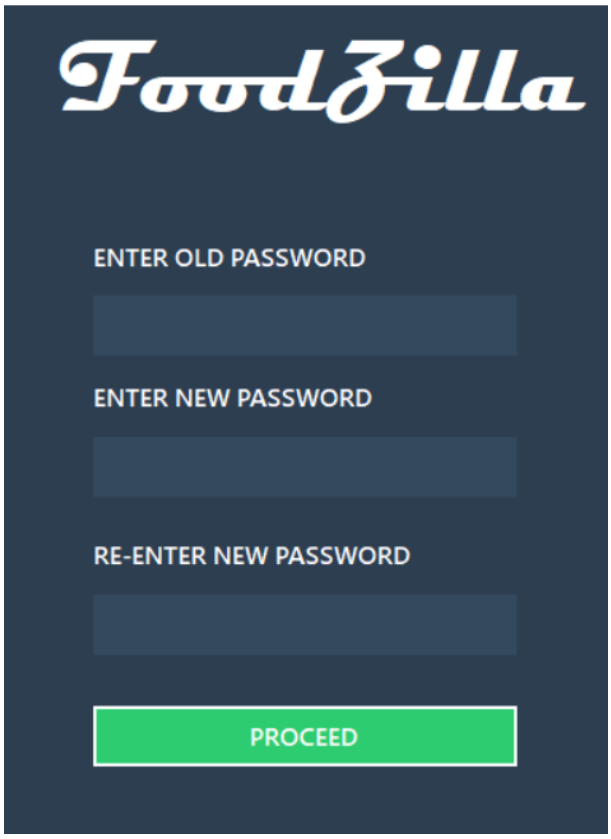
Also Our app can check for updates whenever we push update it will show popup and when u will click on update it will take to gdrive link where we hosted update



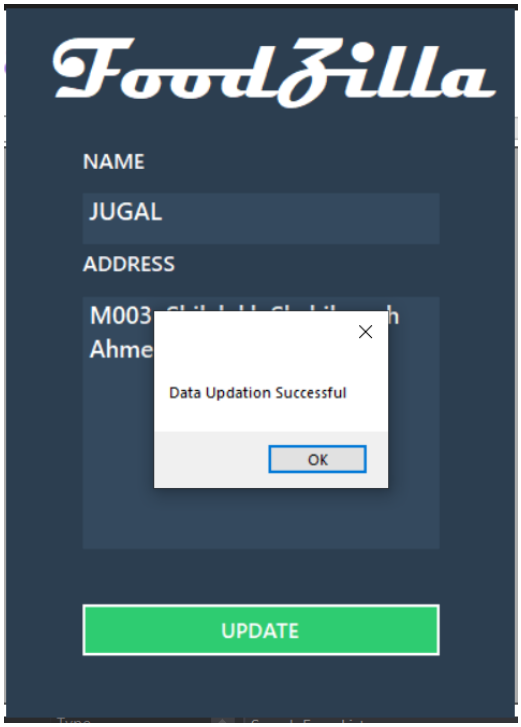
Working shown through GIF



Change Password system



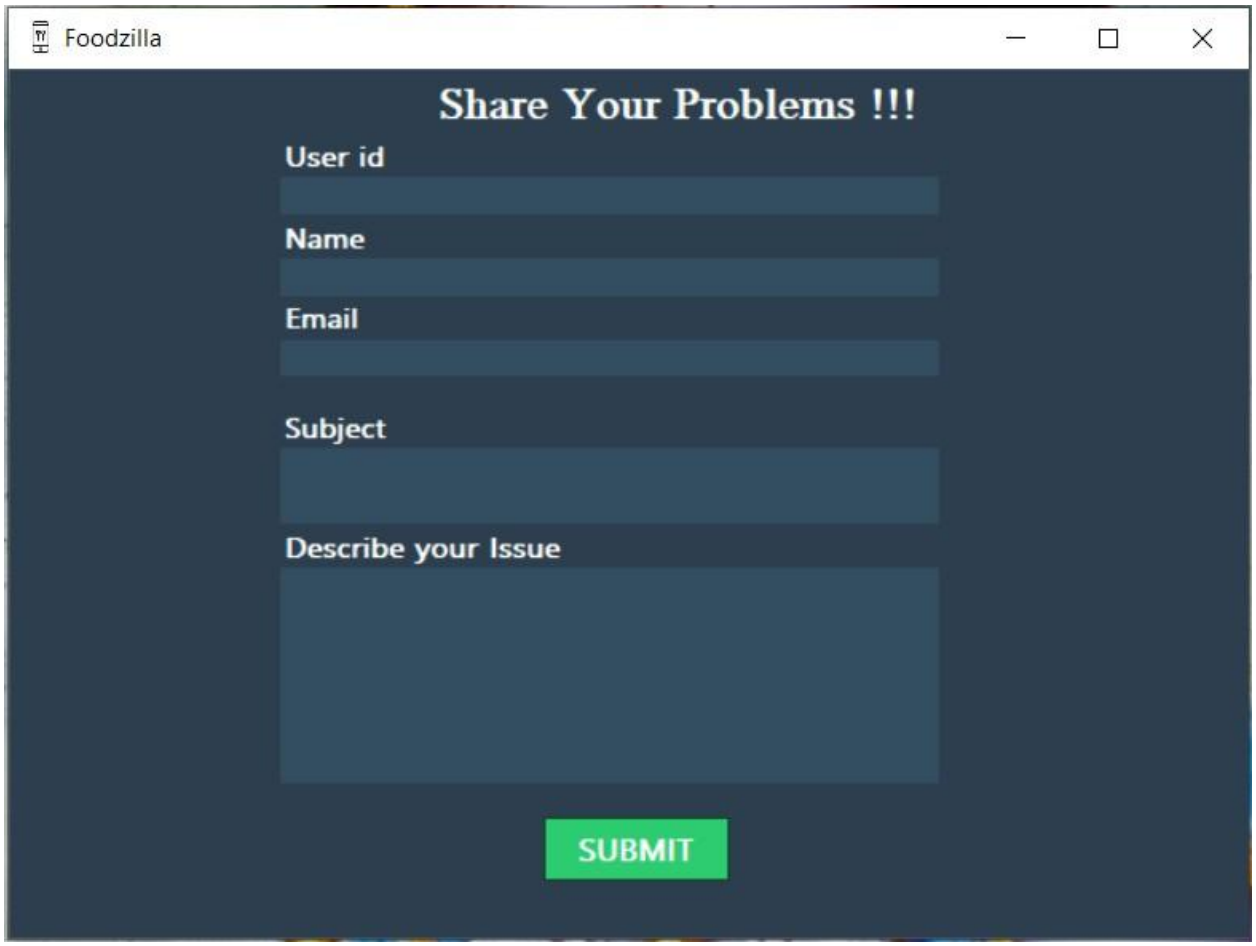
User Details Update



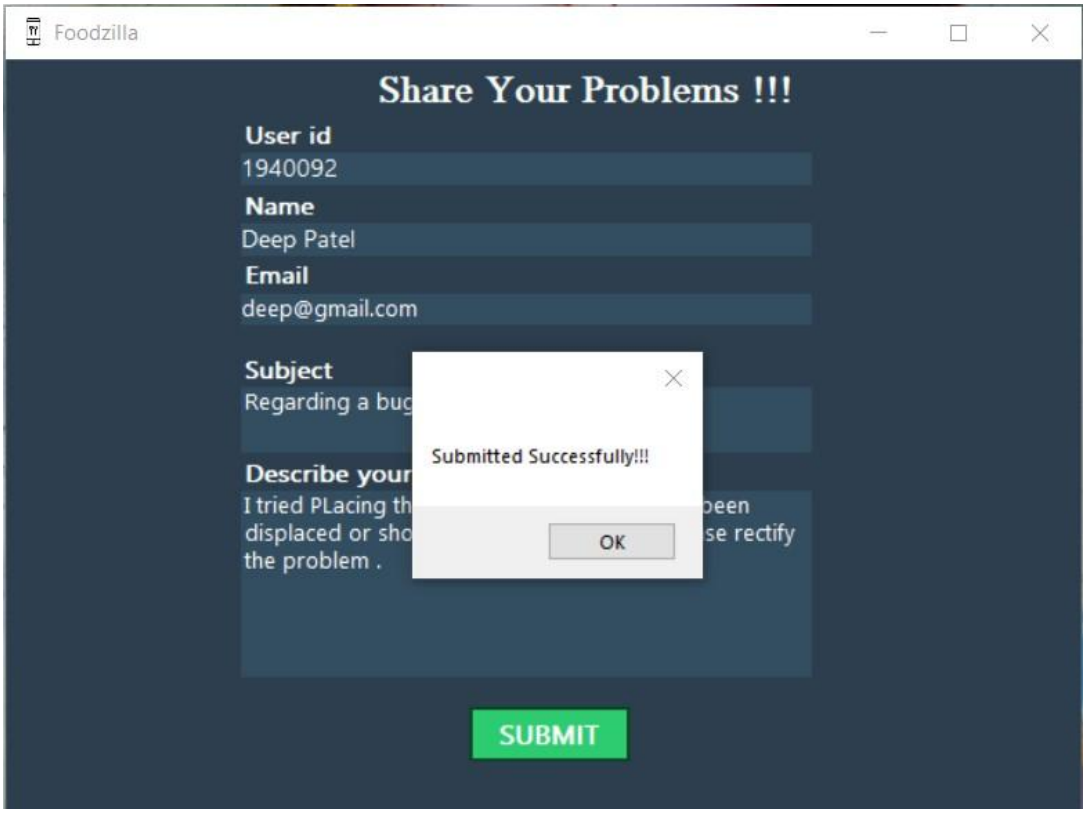
BUG Report system

Sending report to the developers database

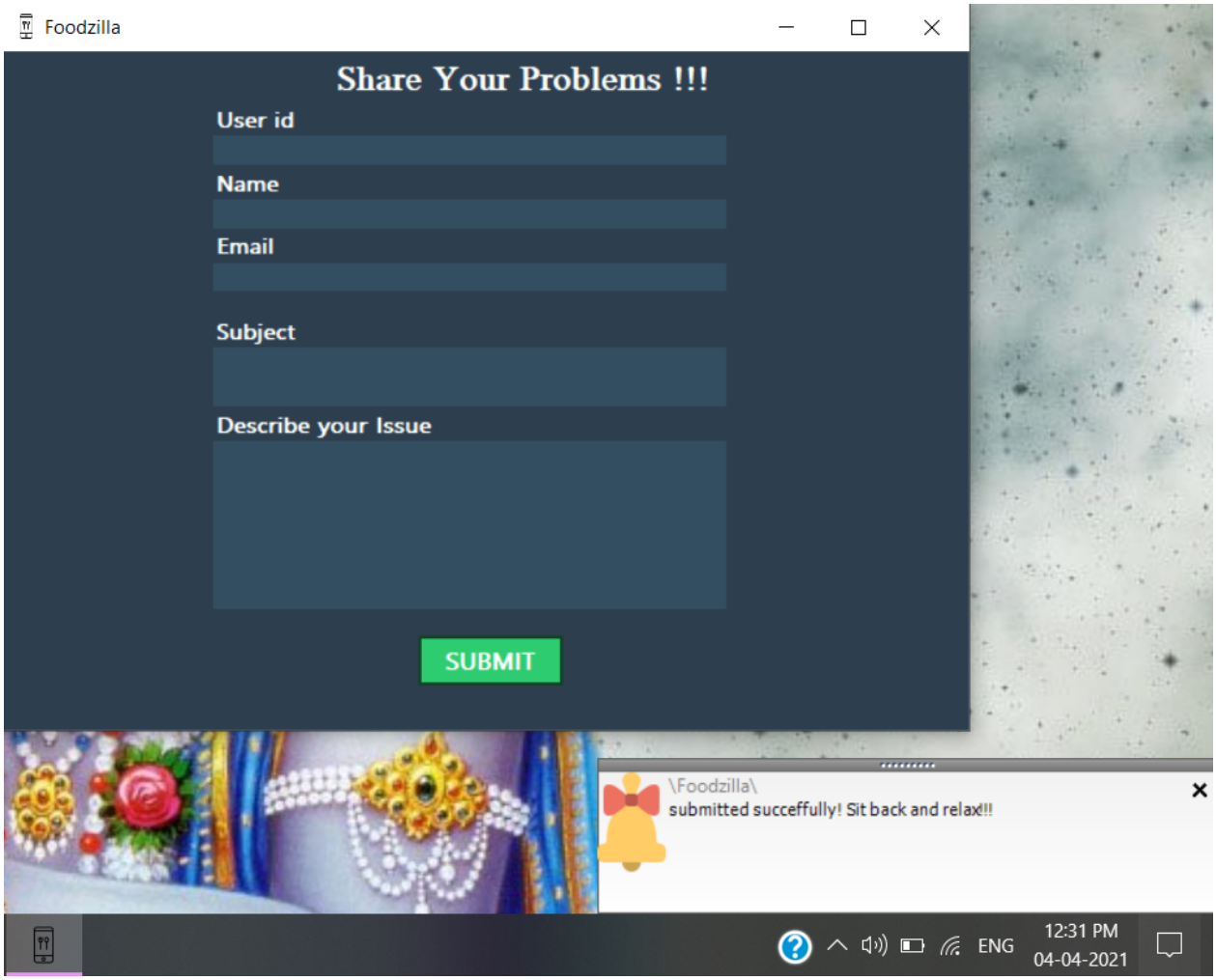
- **This is the UI for the bugs report section**



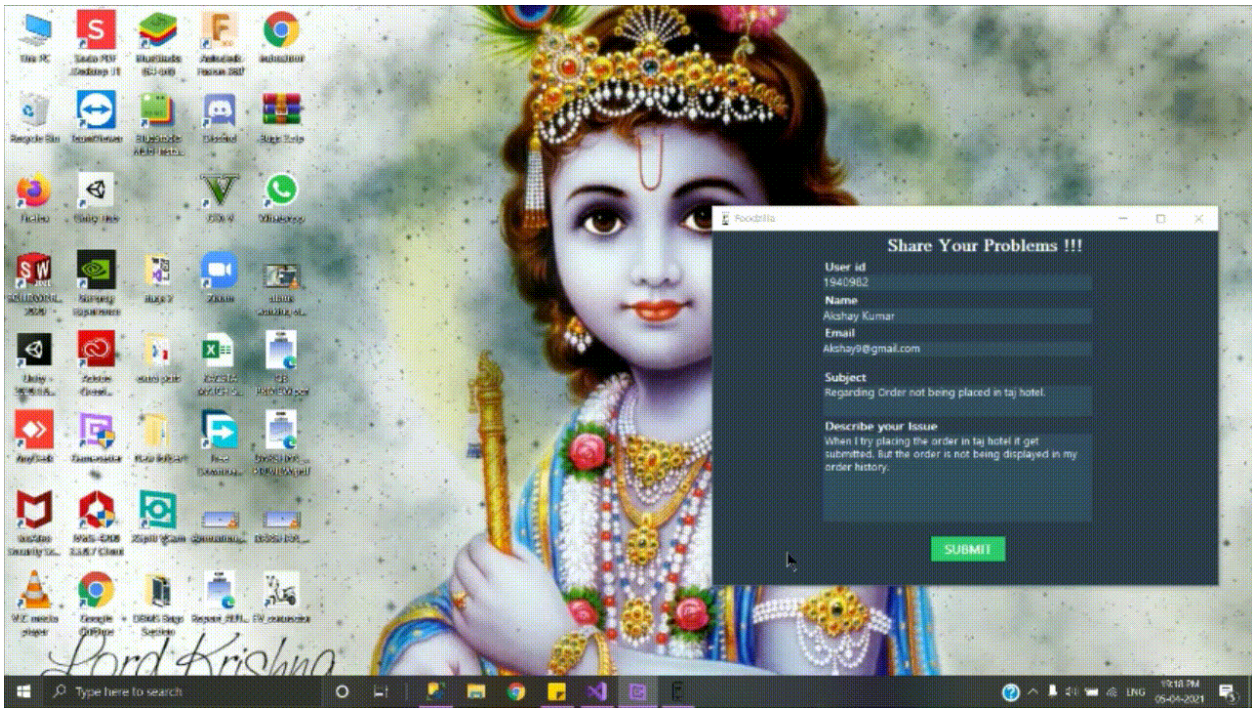
User needs to enter the details and after he presses the Submit Button he/she will receive a message box as shown in the below image.



Later, he/she will also receive an desktop notification about the data being received by the developer as shown below :-



The bug report submitted by the user gets stored into the developers database . Below is the GIF representing how it look like :-



ER Diagram :-

Relational Model :-

Table Design :-

Restaurant				
Field Name	**Datatype**	**Field Size**	**Description**	**Constraints**
r_id	Integer	10	Auto generated	Identityt(1,1), Primary key
r_name	Varchar	150	Name of the restaurant	-
r_address	Varchar	100	Address of the restaurant	-
r_ratings	Integer	2	Rating of a restaurant	Check
r_image	Varchar	150	Image of a restaurant	-
r_type	Varchar	20	Veg or non-veg type	-

Customer				
Field Name	**Datatype**	**Field Size**	**Description**	**Constraints**
c_id	Integer	10	Auto generated	Identityt(1,1), Primary key
c_name	Varchar	50	User name	Not Null
c_email	Varchar	100	User email id	Not Null , Unique
c_address	Varchar	150	User email address	Not Null
c_password	Varchar	20	Password of a customer	Not Null
c_isVerified	Integer	5	Verification of a user	Not Null , Default

Food Item				
Field Name	**Datatype**	**Field Size**	**Description**	**Constraints**
i_id	Integer	10	Auto generated	Identityt(1,1), Primary key
i_name	Varchar	50	Name of item	-
i_price	Float	10	Price of a item	-
i_image	Varchar	150	Image of an item	-
i_type	Varchar	50	Veg or non-veg	-

Order				
Field Name	**Datatype**	**Field Size**	**Description**	**Constraints**
order_id	Integer	10	Auto generated	Identityt(1,1), Primary key
i_price	Float	10	Price of a item that have been ordered	Not Null
i_qty	Integer	5	Quantity of item ordered	Not Null
total_price	Float	20	Total price of the order	Not Null
o_status	Integer	2	Current Status of the order	Not Null
r_id	Integer	10	Restaurant id as a foreign key	Foreign key
c_id	Integer	10	Customer id as a foreign key	Foreign key
i_id	Integer	10	Item id as a foreign key	Foreign key

Restaurant Menu				
Field Name	**Datatype**	**Field Size**	**Description**	**Constraints**
r_id	Integer	10	Restaurant id as a foreign key	Foreign key
i_id	Integer	10	Item id as a foreign key	Foreign key

Service				
Field Name	**Datatype**	**Field Size**	**Description**	**Constraints**
order_id	Integer	10	order id as a foreign key	Foreign key
i_qty	Integer	5	Quantity of item ordered	Not Null
i_price	Float	10	Price of a item ordered	Not Null
i_id	Integer	10	Item id as a foreign key	Foreign key

Backend

Procedures

```
/**PROCEDURE-----1*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'change_password'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.change_password
GO

CREATE PROCEDURE dbo.change_password
 @email AS NVARCHAR(200),
 @newPass AS  NVARCHAR(200)
AS
BEGIN
 UPDATE customer SET c_password = @newPass WHERE c_email =
@email
END
GO
```

```
/**PROCEDURE-----2*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'customer_update_details'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.customer_update_details
GO

CREATE PROCEDURE dbo.customer_update_details
 @email AS NVARCHAR(200),
 @c_name AS NVARCHAR(200),
 @c_address AS NVARCHAR(200)
AS
BEGIN
  UPDATE customer SET c_name = @c_name , c_address = @c_address
WHERE c_email = @email
END
GO
```

```
/**PROCEDURE-----3*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'get_User'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.get_User
GO

CREATE PROCEDURE dbo.get_User
 @email AS NVARCHAR(200)
AS
BEGIN
SELECT c_email AS "username", c_name AS "name", c_password AS
"password",c_id AS "u_id" FROM customer WHERE c_email = @email
 END
GO
```

```
/**PROCEDURE-----4*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'Create_User'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.Create_User
GO

CREATE PROCEDURE dbo.Create_User
 @email AS NVARCHAR(200),
 @name AS NVARCHAR(150),
 @password AS NVARCHAR(150)

AS
BEGIN
  INSERT INTO customer(c_email ,c_name,c_password)
VALUES(@email,@name,@password)
END
GO
```

```
/**PROCEDURE-----5*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'Verification'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.Verification
GO

CREATE PROCEDURE dbo.Verification
 @email AS NVARCHAR(200)
AS
BEGIN
 UPDATE customer SET c_isVerified = 1 WHERE c_email = @email
END
GO
```

```
/**PROCEDURE-----6*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'Create_order'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.Create_order
GO

CREATE PROCEDURE dbo.Create_order
 @i_id AS INT,
 @r_id AS INT,
 @i_price AS INT,
 @i_qty AS INT,
 @tprice AS FLOAT,
 @c_id AS INT
AS
BEGIN
  INSERT INTO [order] (r_id, i_id , i_price , i_qty,
total_price, c_id) VALUES (@r_id, @i_id , @i_price , @i_qty,
@tprice, @c_id)
END
GO
```

```
/**PROCEDURE-----7*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'My_order'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.My_order
GO

CREATE PROCEDURE dbo.My_order
 @c_id AS INT
AS
BEGIN
   SELECT * FROM [order]  WHERE c_id = @c_id
END
GO
```

```
/**PROCEDURE-----8*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'Edit_order'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.Edit_order
GO

CREATE PROCEDURE dbo.Edit_order
 @i_id AS INT,
 @r_id AS INT,
 @i_price AS INT,
 @i_qty AS INT,
 @tprice AS FLOAT,
 @o_id AS INT
AS
BEGIN
      UPDATE [order] SET i_id = @i_id, r_id = @r_id, i_price =
@i_price ,i_qty = @i_qty , total_price = @tprice WHERE order_id
= @o_id
 END
GO
```

```sql
/**PROCEDURE-----9*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'cancel_order'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.cancel_order
GO

CREATE PROCEDURE dbo.cancel_order
 @o_id AS NVARCHAR(200)
AS
BEGIN
UPDATE [order] SET o_status = 1 WHERE order_id = @o_id

END
GO
```

```
/**PROCEDURE-----10*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'show_restaurant_list'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.show_restaurant_list
GO

CREATE PROCEDURE dbo.show_restaurant_list

AS
BEGIN
     SELECT * FROM restaurant
   END
GO
```

```
/**PROCEDURE-----11*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'show_restaurant_bySearch'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.show_restaurant_bySearch
GO

CREATE PROCEDURE dbo.show_restaurant_bySearch
@rName AS NVARCHAR(200)
AS
BEGIN
     SELECT * FROM restaurant WHERE r_name = @rName OR r_name
LIKE '%@rName%'
   END
GO
```

```
/**PROCEDURE-----12*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'show_items_of_restaurnat'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.show_items_of_restaurnat
GO

CREATE PROCEDURE dbo.show_items_of_restaurnat
@r_id AS INT
AS
BEGIN
     SELECT * FROM food_items WHERE i_id = (SELECT * FROM
restaurant_menu WHERE r_id = @r_id)
   END
GO
```

```
/**PROCEDURE-----13*/
IF EXISTS (
SELECT *
  FROM INFORMATION_SCHEMA.ROUTINES
WHERE SPECIFIC_SCHEMA = N'dbo'
  AND SPECIFIC_NAME = N'show_items_of_restaurnat_by_search'
  AND ROUTINE_TYPE = N'PROCEDURE'
)
DROP PROCEDURE dbo.show_items_of_restaurnat_by_search
GO

CREATE PROCEDURE dbo.show_items_of_restaurnat_by_search
@r_id AS INT,
@iName AS NVARCHAR(200)
AS
BEGIN
  SELECT * FROM food_items WHERE  i_name = @iName OR i_name LIKE
'%@iName%'  (SELECT * FROM restaurant WHERE r_id = @r_id  )
   END
GO
```

Trigger Related Errors

[Frontend is Ready for that already]