CSE 1005

PROJECT

Project Title: Restaurant Management System

TEAM MEMBERS

Sai Varun-19BCE7092

Table of Contents:

Feasibility Study

Existing System

Restaurants are popular since the 18th century, with a variety of cuisines and dishes. Currently, in most restaurants, the menu and orders are still done manually by waiters. Customers must wait until they receive the menu to place an order. Using a digital menu instead of old ones has lots of advantages for a profitable business. Here are a few points showing digital menu is better than an old menu.

Pros of digital menu

- Dishes are available with their pictures
- Popular dishes are spotlighted
- Cuisine combos
- Quick placement of orders
- Know the real-time status of an order
- Great insights about the dish

Cons of an old menu

- All dishes are mentioned irrespective of the availability
- No quick review of the dish
- Dependent on the waiter to place an order
- Not so attractive

Problem Statement

Under this, the following are the various problems that are faced so often, we have to wait for longer times to place an order, which in return waste a lot of time and a greater level of patience. There could be many human errors while dealing with the information manually. There are various benefits on account of automation. This includes an increase in customer satisfaction, better decision-making timeliness of information, expediting activities, and making the work easy for management as well as customers.

Objective

The software which we are proposing now has many enhancing features of the 21st century, which not only saves time but is also user-friendly, which helps for better and more accurate alignment. The digital menu features, such as food items display, and direct order placement improves customer satisfaction. Not just photos of dishes, popular recommended foods based on previous customer reviews and a whole new combo of food from different origins that customers can explore.

1. PROBLEM ANALYSIS

1.1 Overview of the project:

Using a digital menu instead of old ones has lots of advantages for a profitable business. Customers can order their food directly through an electronic tablet, available at each table. The order directly goes to the chef, which helps the chef to get a clear idea of the next orders.

1.2 Identification of project scope:

Task involved:

The main scope of this project is to improve customer satisfaction with features such as digital menus where customers get pictures of dishes, their origin, and taste. And make the management easy for restaurants.

1.3 Objectives

The software which we are proposing now has many enhancing features of the 21st century, which not only saves time but is also user-friendly, which helps for better and more accurate alignment. The digital menu features, such as food items display, and direct order placement improves customer satisfaction. Not just photos of dishes, popular recommended foods based on previous customer reviews, and a whole new combo of food from different origins that customers can explore.

1.4 Infrastructure

Each developer in our team will work on a particular aspect of the project and push it on Github from time to time. Once the project is done, we will create a test environment wherein we will run tests on the application using the TEM tool Apwide Golive**.** Next, we will create a QA environment to test the existing functionality, log the bugs and retest the fixed bugs, and perform code reviews using Test I/O. Once this is done, Selenium S is used to test the user interface design. Also, we will create a pipeline for automatic build, test and deploy.

2. SOFTWARE REQUIREMENT ANALYSIS AND PLANNING

2.1 Description of individual phase/module:

Module	**Description**
Login	Phone number OTP login (for previous order analysis and feedback)
Menu	List of dishes with photos
Chef	Get info about orders to prepare

Status	Current order status
Waiter	Get info about next dish to serve along with table id
Payment	Payment through: UPI, Net Banking, Card, Cash

2.1.1 User characteristics

- Users should know how to operate a computerized system
- All the systems will have a proper connection
- Admin should beware of malware, virus and other aspects which can harm the computer system
- Users should be familiar with self-payment after the final bill

2.1.2 General constraints

- We required tablets with custom software installed
- All the tablets should be connected to an Admin PC using Wi-fi

2.1.3 Assumption:

- Tablets for all tables with software installed
- Power points at each table
- Large monitors in the kitchen and waiter's space
- Good Wi-Fi connection
- Backup in case of a powercut **Dependency:**
- Min Tablet specification:
  - Android 9+
  - RAM: 2GB
  - Storage: min 5GB
- Tablets should be compactable with the software, if not, the software or tablet should be changed
- Cloud service for database administration and data backup
- Power generators for consistent power supply

2.1.4 Functional requirements:

A functional requirement is a requirement that, when satisfied, will allow the user to perform some kind of function. The priority value is in the range from 1 to 3 where 1 is a high priority and 3 is a low priority.

Requirements	Priority
Organised display of active orders	1
View preferences and optional choices for every meal	1
Inform client; update order to ‘in-progress’	2
Inform waiter; update order to ‘ready’	2
Display elapsed time and progress2 of each order	3

2.2 Identify individual module deliverables

Login:

- Primary actor: client
- Purpose: To identify the client
- Function information: Personalized experience of the client

Menu:

- Primary actor: client
- Purpose: Choosing a dish
- Function information: Dishes with photos are available. List of orders with ORDER button.

Chef:

- Primary actor: chef
- Purpose: Get the list of orders pending
- Function information: As soon as the client orders, it appears on the chef module.

Status:

- Primary actor: waiter, chef
- Purpose: Know the status of orders
- Function information: As soon as the client orders a dish, it appears on the chef module. The chef should update the dish status before starting as well as in the end.

Waiter:

- Primary actor: waiter
- Purpose: For knowing the order status to serve the food

- Function information: After the status shows finished it comes to the waiter module as ready to serve. The waiter updates it as finished after serving it to the table.

Payment:

- Primary actor: client
- Purpose: Checkout bill and payment portal
- Function information: As soon as the client requests the bill, the payment portal opens up with various modes of payment. The client has to choose one to complete the transaction.

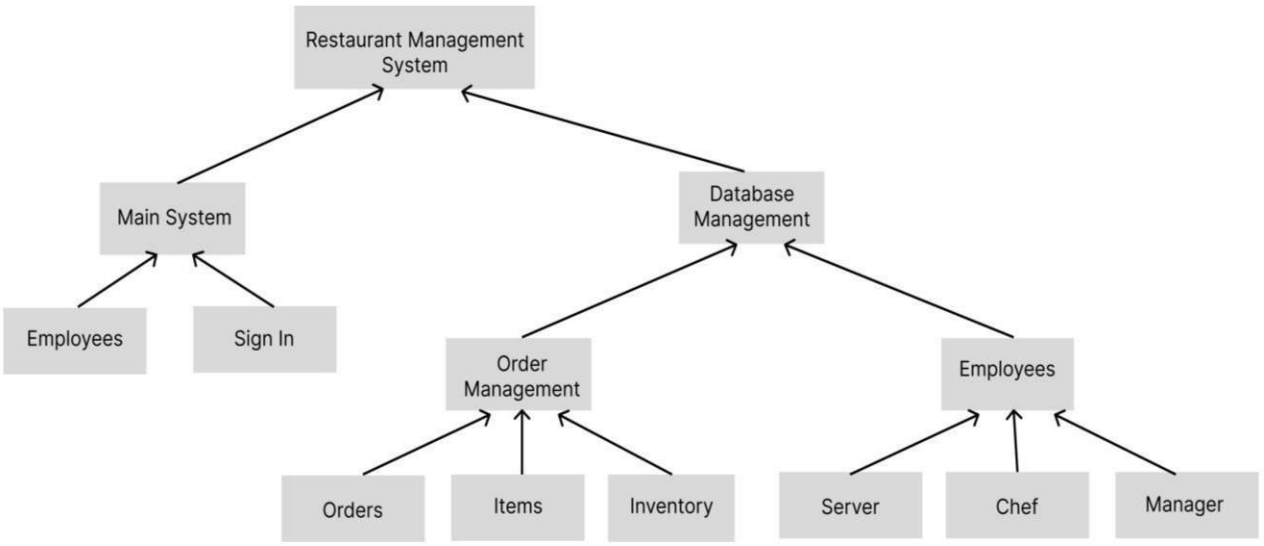
Database:

- Primary actor: Admin
- Purpose: Storing customer data
- Function information: Stores data of customer with phone number as the super key. Previous feedback from the client is used to improve current experience and satisfaction.

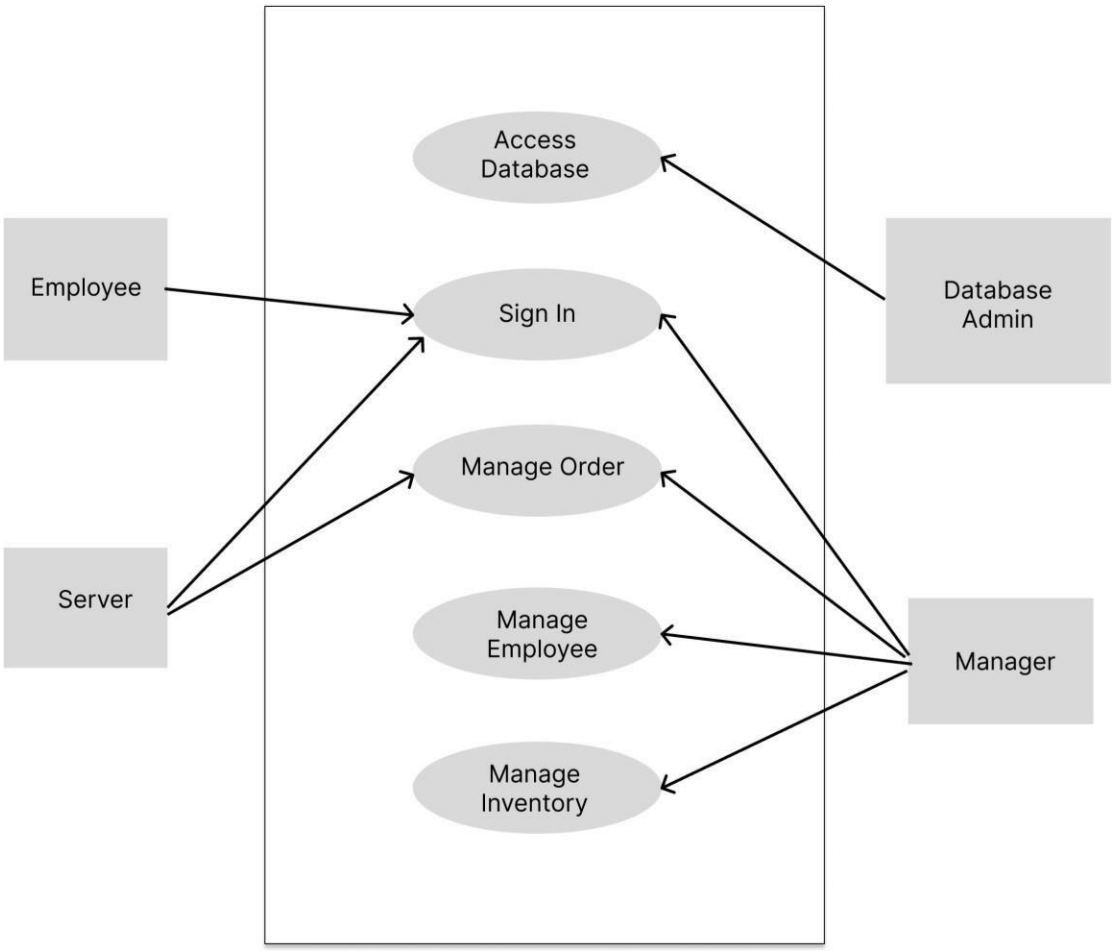
3. DATA MODELING

UML diagrams:

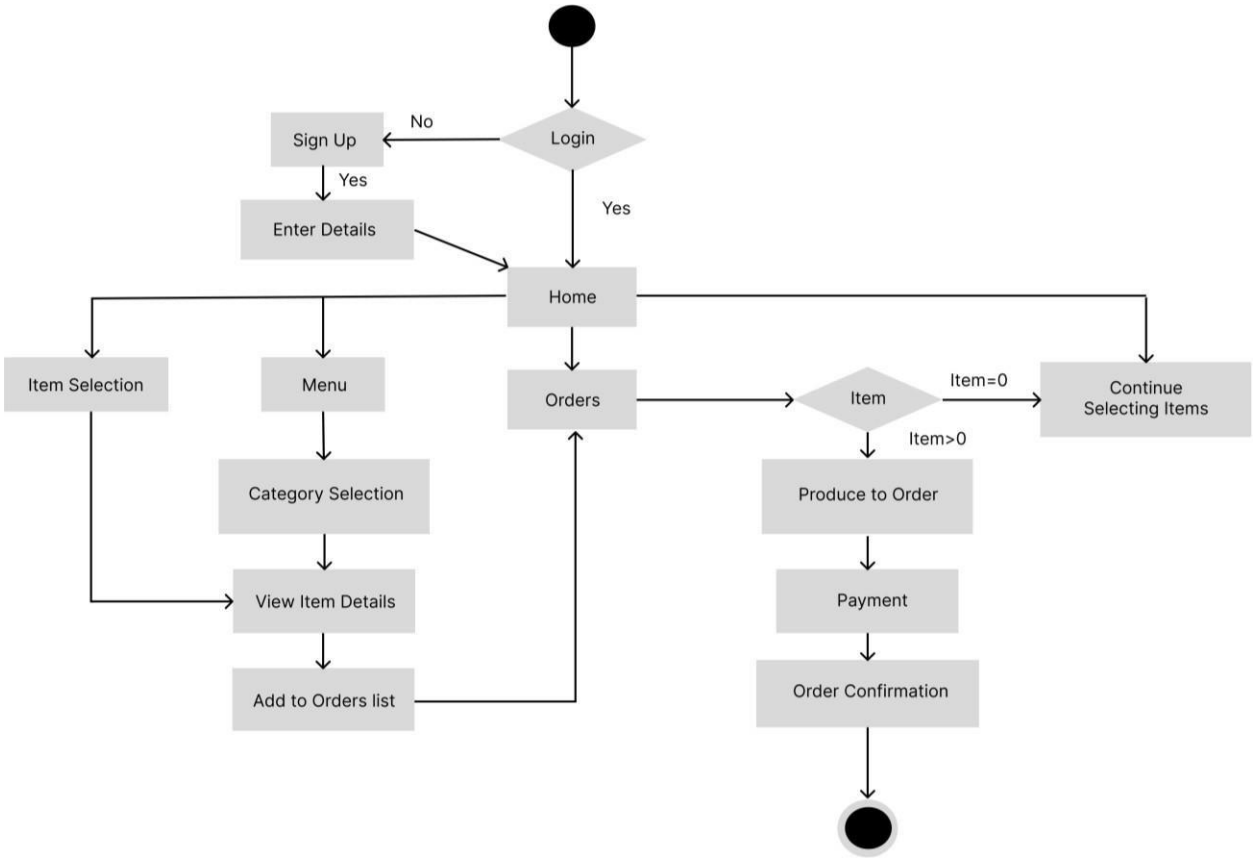
1. System Architecture Design



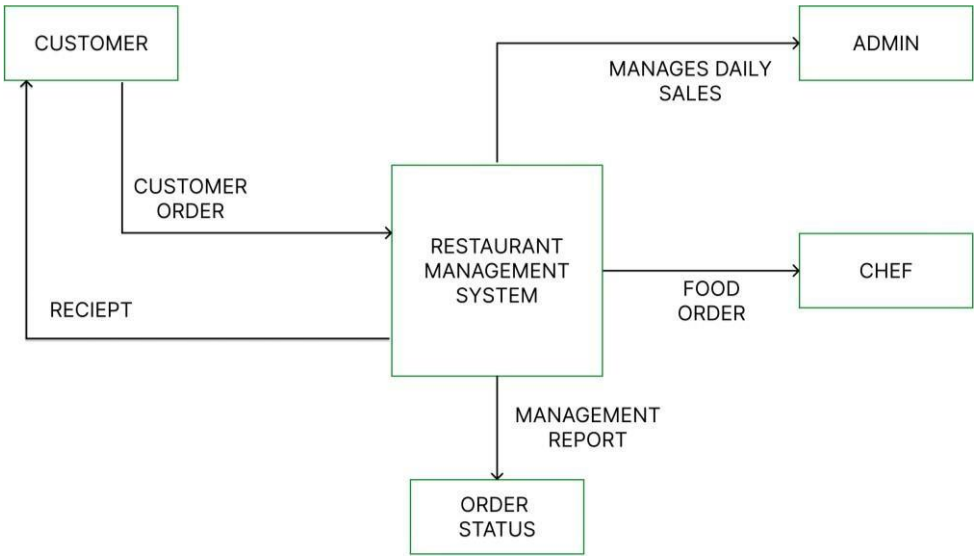
2. Use Case Diagram



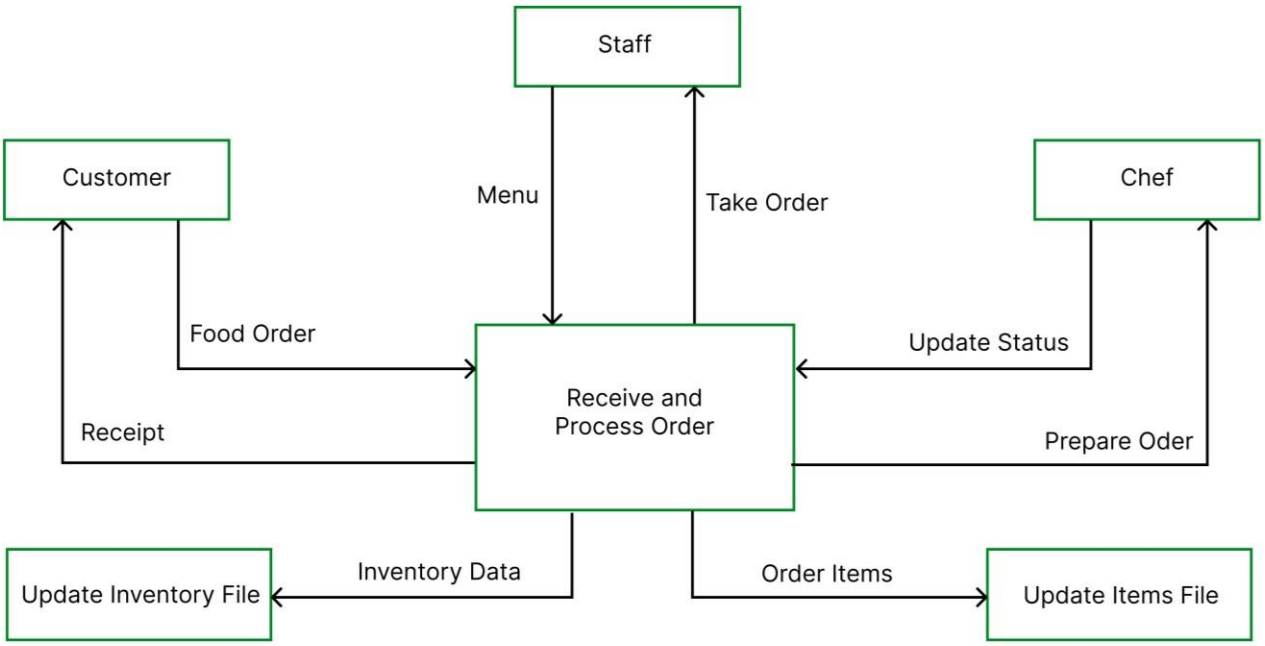
3. Activity Diagram

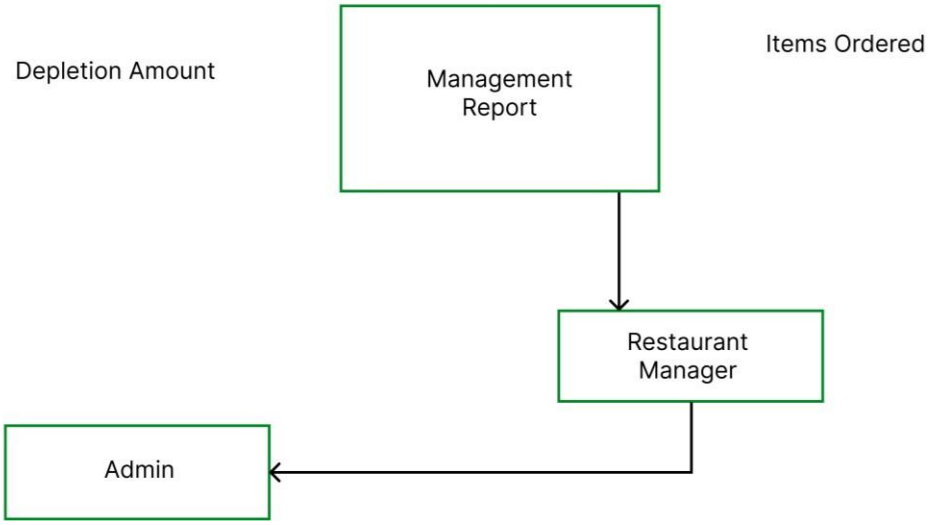


4. DFD Diagram



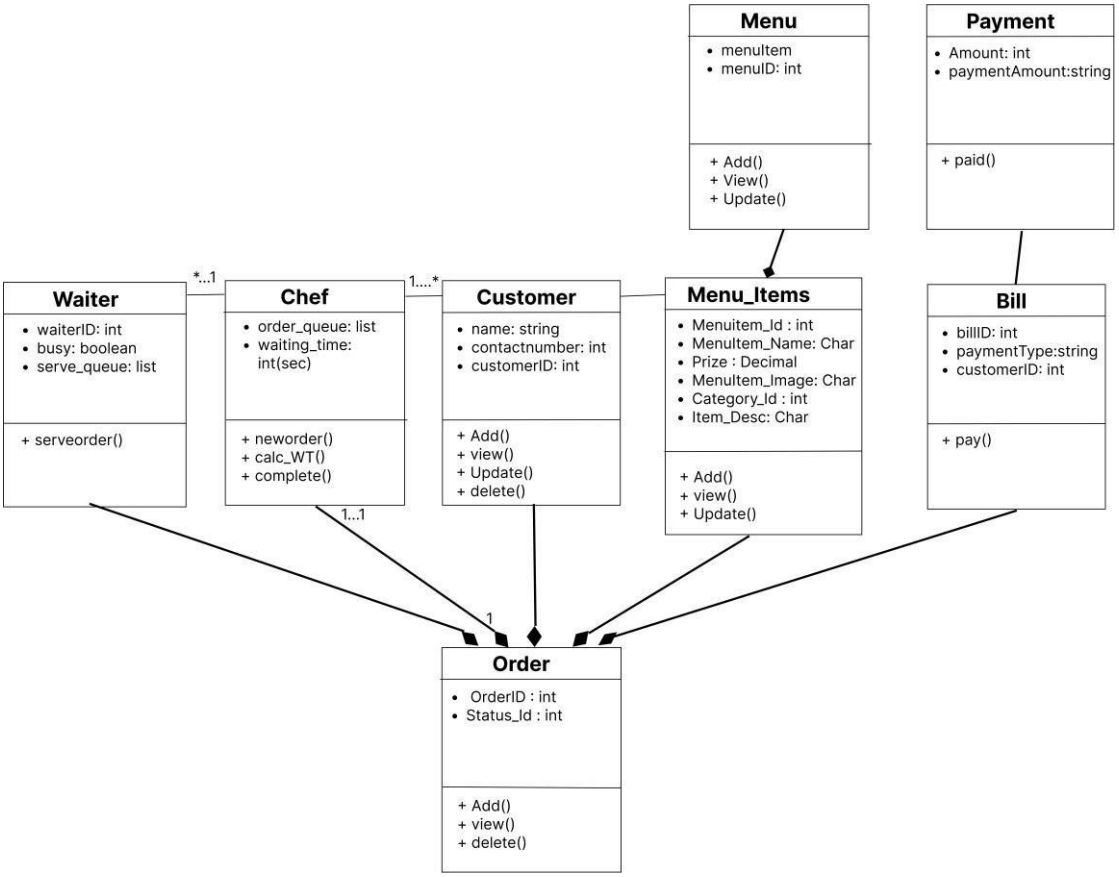
Level 1: Data Flow Diagram



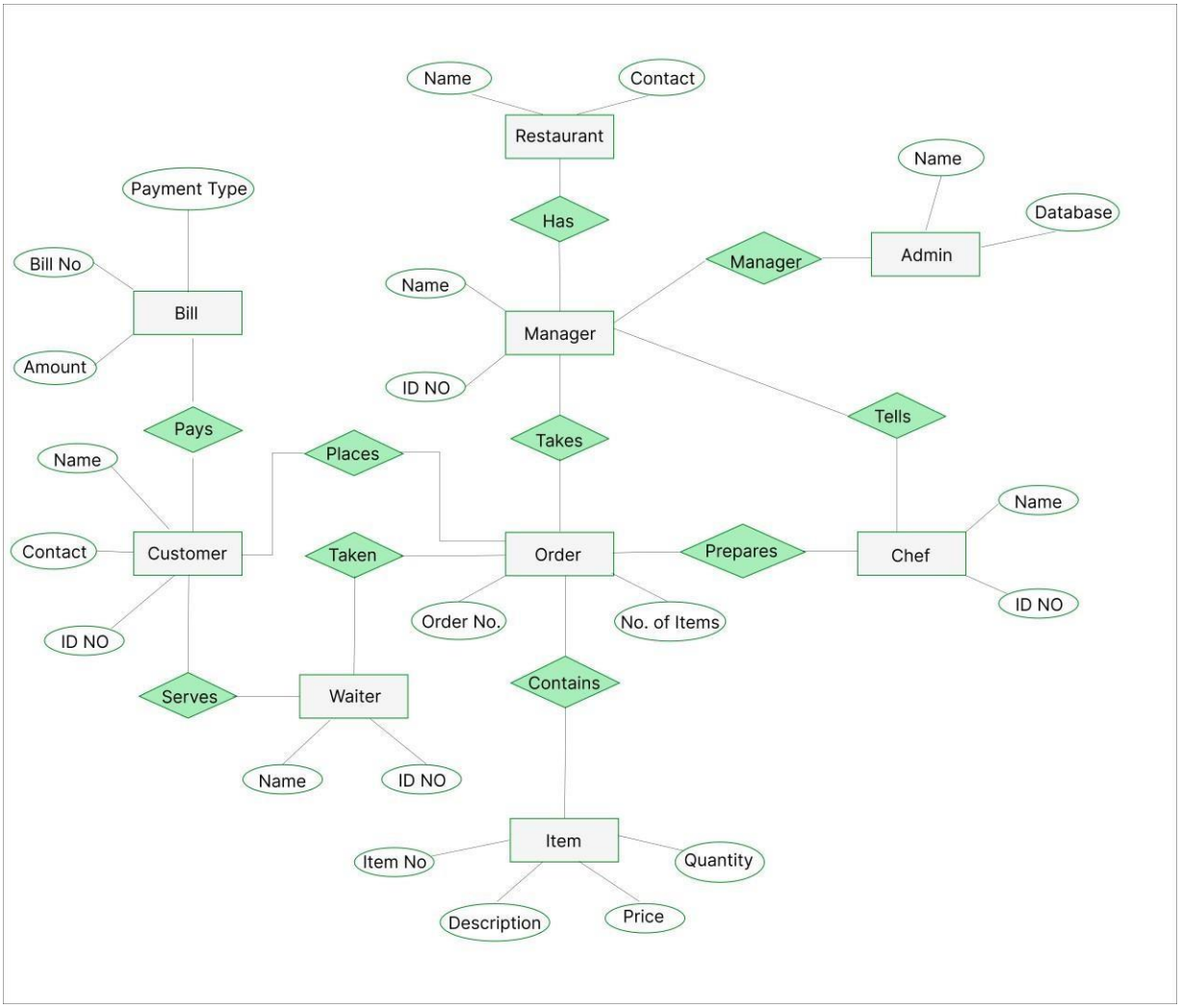


Level 2: Data Flow Diagram

5. Class Diagram



6. Database Diagram:



4. DEVELOPMENT

4.1 Coding and Implementation

Java Code files

<MainActivity.java>

```
5     package     com.example.easykitchen;     import

   androidx.appcompat.app.AppCompatActivity;

   import android.content.Intent;
   import android.os.Bundle; import
   android.view.View; import
   android.view.Window; import
   android.widget.EditText; import
   android.widget.Toast;
 public class MainActivity extends AppCompatActivity {
   EditText ph;


       @Override
       protected void onCreate(Bundle savedInstanceState) {
           super.onCreate(savedInstanceState);
           this.requestWindowFeature(Window.FEATURE_NO_TITLE);
       setContentView(R.layout.activity_main); }
 public void next(View view) { ph =
       findViewById(R.id.ph_no); String p =
       ph.getText().toString();
       if(p.length()==10){ startActivity(new
   Intent(MainActivity.this,MainScreen.class));
           } else{
               Toast t = Toast.makeText(getApplicationContext(),"Please
   enter a valid 10-digit mobile number",Toast.LENGTH_LONG);
               t.show();
           }
       }
   }
```

<MainScreen.java>

```
package com.example.easykitchen;
 import android.app.AlertDialog; import
android.content.DialogInterface;
import android.content.Intent; import
android.os.Bundle; import
android.provider.Settings; import
android.view.LayoutInflater; import
android.view.View; import
android.widget.ArrayAdapter; import
android.widget.ListView; import
android.widget.Toast;
 import
com.example.easykitchen.ui.home.menu_list;
```

```
import com.google.android.material.bottomnavigation.BottomNavigationView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.navigation.NavController; import
androidx.navigation.Navigation;
import       androidx.navigation.ui.AppBarConfiguration;       import

androidx.navigation.ui.NavigationUI;                              import

com.example.easykitchen.databinding.ActivityMainScreenBinding; import

java.util.ArrayList; public class MainScreen extends AppCompatActivity

{ private ActivityMainScreenBinding binding;

    public ListView cust_view,serve_view,chef_view;
    public ArrayList<item>
    curr_menu,chef_list,serve_list;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityMainScreenBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        BottomNavigationView navView = findViewById(R.id.nav_view);
        // Passing each menu ID as a set of Ids because each
// menu should be considered as top level destinations.
AppBarConfiguration appBarConfiguration = new
AppBarConfiguration.Builder(
                R.id.navigation_home, R.id.navigation_dashboard,
R.id.navigation_notifications)
                .build();
        NavController navController = Navigation.findNavController(this,
R.id.nav_host_fragment_activity_main_screen);

        NavigationUI.setupWithNavController(binding.navView,
navController);

        cust_view = findViewById(R.id.cust_list); chef_view
        = (ListView) findViewById(R.id.chef_list); curr_menu
        = menu_list.it_arr;



        ListAdapter listAdapter = new ListAdapter(this,curr_menu);
        cust_view.setAdapter(listAdapter);

    }

    public    void    add_open(View    add)    {    startActivity(new
        Intent(MainScreen.this,menu_list.class));
    }

    public void generate_bill(View view) { int
        Total = 0;
        for (item i : menu_list.it_arr) {
            Total+=i.getCost()*i.getQuant();
        }
        AlertDialog altr = new AlertDialog.Builder(this).create();
```

```
        altr.setTitle("Toatal Bill"); altr.setMessage("₹
        "+Total);
        altr.setButton("PAY", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                System.exit(0);

            } });
        altr.show();

    }



}
```

<item.java>

```
package com.example.easykitchen; public

class item {

    String dish_name,status;
    int    cost,quant;    int
    imgID;
 public item(String dish_name,int cost, int quant,String status){
    this.dish_name = dish_name; this.cost = cost; this.quant =
    quant; this.imgID = imgID; this.status = status;
    }
 public String getDish_name(){return dish_name;}
 public void setDish_name(String t_dish){this.dish_name = t_dish;}

    public int getImgID(){return imgID;} public void setImgID(int

    t_imgID){this.imgID = t_imgID;} public int getCost(){return

    cost;} public void setCost(int t_cost){this.cost = t_cost;}

    public int getQuant(){return quant;} public void setQuant(int

    t_quant){this.quant = t_quant;} public String getStatus(){return

    status;}

    public void setStatus(String t_status){this.status = t_status;} }
```

<menu_list.java>

```
package com.example.easykitchen.ui.home; import

androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle; import
android.view.View; import
android.widget.AdapterView; import
android.widget.EditText; import
android.widget.ListView; import
android.widget.TextView; import
android.widget.Toast;

import com.example.easykitchen.ListAdapter;
import com.example.easykitchen.MainScreen; import
com.example.easykitchen.R;
import com.example.easykitchen.databinding.ActivityMainBinding;
import com.example.easykitchen.databinding.ActivityMainScreenBinding;
import com.example.easykitchen.databinding.ActivityMenuListBinding;
import com.example.easykitchen.item; import java.util.ArrayList;
public class menu_list extends AppCompatActivity {

    ActivityMenuListBinding binding;
    TextView dn,dc; EditText
    et;

    int[] dish_name={R.id.dish_name1,
            R.id.dish_name2, R.id.dish_name3,
            R.id.dish_name4, R.id.dish_name5,
            R.id.dish_name6, R.id.dish_name7,
            R.id.dish_name8, R.id.dish_name9,
            R.id.dish_name10
    };
    int[] dish_cost={R.id.dish_cost1,
            R.id.dish_cost2, R.id.dish_cost3,
            R.id.dish_cost4, R.id.dish_cost5,
            R.id.dish_cost6, R.id.dish_cost7,
            R.id.dish_cost8, R.id.dish_cost9,
            R.id.dish_cost10
    };
    int[] dish_quants={R.id.quant_field1,
            R.id.quant_field2, R.id.quant_field3,
            R.id.quant_field4, R.id.quant_field5,
            R.id.quant_field6, R.id.quant_field7,
            R.id.quant_field8,
```

```
            R.id.quant_field9,
            R.id.quant_field10
    };
    public static ArrayList<item> it_arr = new ArrayList<item>();

    String[] dishes = {"Tomato Soup","Panner Manchurian","Veg
    Biryani","Veg
Fried Rice","South Indian Thali","Panner Butter Masala","Butter
Naan","Chocolate Cake","Vanilla Ice-cream","Butter Milk"}; int[]
    cost = {129,269,275,229,199,299,50,599,169,40};



    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        binding = ActivityMenuListBinding.inflate(getLayoutInflater());
        setContentView(R.layout.activity_menu_list);

 for(int   i=0;i<dishes.length;i++){   dn   =
        findViewById(dish_name[i]);   dc   =
        findViewById(dish_cost[i]);
        dn.setText(dishes[i]); dc.setText("₹
        "+cost[i]);
        }


    }

 public void menu_close(View view) {
    this.finish();
    }



    public void order_these(View view) {
        for(int i=0;i<dish_quants.length;i++){
        et = findViewById(dish_quants[i]);
        String s =et.getText().toString();
        if(s.length()>0){ item it = new
item(dishes[i],cost[i],Integer.parseInt(s),"Food is Processing");
                it_arr.add(it);
            }
        }
        this.finish();
        startActivity(new Intent(menu_list.this,MainScreen.class));
    }
}
```

<ListAdapter.java>

```
package com.example.easykitchen;
 import android.content.Context;
import
android.view.LayoutInflater;
import android.view.View; import
android.view.ViewGroup; import
```

```
android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;
```

```
import  androidx.annotation.NonNull;  import
androidx.annotation.Nullable;
 import
org.w3c.dom.Text;
 import
java.util.ArrayList; import
java.util.List;

public class ListAdapter extends ArrayAdapter<item> {
 public ListAdapter(Context context, ArrayList<item> menuList){
    super(context,0,menuList); }

    @NonNull @Override
     public View getView(int position, @Nullable View convertView, @NonNull
ViewGroup parent) {

        item it = getItem(position);

        if(convertView == null){ convertView
            =
LayoutInflater.from(getContext()).inflate(R.layout.cust_orders,parent,false
);


        }
        TextView dish_name = convertView.findViewById(R.id.curr_dish_name);
        TextView dish_quant =
convertView.findViewById(R.id.curr_dish_quant);
        TextView dish_status =
convertView.findViewById(R.id.curr_dish_status);
 dish_name.setText(it.getDish_name());
        dish_quant.setText(String.valueOf(it.getQuant()));
        dish_status.setText(it.getStatus());

//   return   super.getView(position,   convertView,   parent);   return
        convertView;
    } }
```

<CustomerFragment.java>

```
package com.example.easykitchen.ui.home;
 import android.app.AlertDialog; import
android.content.DialogInterface; import
android.content.Intent;          import
android.os.Bundle;               import
android.view.LayoutInflater;     import
android.view.View;               import
android.view.ViewGroup;          import
android.widget.Button;           import
android.widget.ListView;         import
android.widget.TextView;
 import   androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
```

```
import androidx.lifecycle.ViewModelProvider;
 import com.example.easykitchen.ListAdapter;
import
com.example.easykitchen.MainActivity;
import com.example.easykitchen.MainScreen;
import com.example.easykitchen.R;
import com.example.easykitchen.databinding.FragmentHomeBinding;
import com.example.easykitchen.item;
 import  java.util.ArrayList;  public  class
HomeFragment extends Fragment {

    ListView cust_view;
    ArrayList<item>     curr_menu;     private
    FragmentHomeBinding    binding;    private
    AlertDialog.Builder dialogBuilder;
    private AlertDialog dialog; Button
    ad,cl;
 public View onCreateView(@NonNull LayoutInflater inflater,
                          ViewGroup container, Bundle
savedInstanceState) {
        HomeViewModel          homeViewModel          =          new
                ViewModelProvider(this).get(HomeViewModel.class);
 binding  =  FragmentHomeBinding.inflate(inflater,  container,  false);  View
        root = binding.getRoot();
 final TextView textView = binding.textHome;
        homeViewModel.getText().observe(getViewLifecycleOwner(),
textView::setText);

 cust_view = root.findViewById(R.id.cust_list);
        curr_menu = menu_list.it_arr;

        ListAdapter listAdapter = new ListAdapter(getContext(),curr_menu);
        cust_view.setAdapter(listAdapter);


 return root;
    }

    @Override
    public void onDestroyView() {
        super.onDestroyView();
        binding = null;
    }



}
```

<WaiterFragment.java>

```
package com.example.easykitchen.ui.dashboard;
 import android.app.AlertDialog; import
android.content.DialogInterface;
```

```
import android.os.Bundle; import
android.view.LayoutInflater; import
android.view.View; import
android.view.ViewGroup; import
android.widget.AdapterView; import
android.widget.ListView; import
android.widget.TextView;

import  androidx.annotation.NonNull;  import
androidx.fragment.app.Fragment;       import
androidx.lifecycle.ViewModelProvider;

import com.example.easykitchen.ListAdapter; import
com.example.easykitchen.R;
import com.example.easykitchen.databinding.FragmentDashboardBinding;
import com.example.easykitchen.item;
import com.example.easykitchen.ui.home.menu_list; import

java.util.ArrayList;

public class DashboardFragment extends Fragment {
    ListView serve_view;
    public ArrayList<item> serve_list,curr_menu;
    private FragmentDashboardBinding binding;

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle
savedInstanceState) {
        DashboardViewModel         dashboardViewModel         =         new
                ViewModelProvider(this).get(DashboardViewModel.class);

        binding = FragmentDashboardBinding.inflate(inflater, container,
false);
        View  root  =  binding.getRoot();  serve_view  =

        root.findViewById(R.id.serve_list);

        curr_menu = menu_list.it_arr; serve_list = new
        ArrayList<>(); for(int i =0;i<curr_menu.size();i++) { if
        (curr_menu.get(i).getStatus().charAt(0) == 'R') {
        serve_list.add(curr_menu.get(i));
            }
        }


        ListAdapter chefadapter = new
ListAdapter(requireContext(),serve_list);
        serve_view.setAdapter(chefadapter);
        serve_view.setClickable(true);



        serve_view.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int
position, long id) { final int which_item =
                position;
                new AlertDialog.Builder(requireContext())
                        .setIcon(R.drawable.ic_baseline_chef_24)
```

```
                        .setTitle("Served?")
                       .setMessage("Change status to Finished")
                       .setPositiveButton("YES", new
DialogInterface.OnClickListener() {
                           @Override
                             public void onClick(DialogInterface dialog, int
which) { change_curr(serve_list.get(which_item));
                               serve_list.remove(which_item);
                               chefadapter.notifyDataSetChanged();

                           }
                       })
                        .setNegativeButton("No",null)
                       .show();
            } });

 return root;
    }

    @Override
    public void onDestroyView() {
        super.onDestroyView(); binding =
        null;
    }
 public void change_curr(item it){ for(int
    i=0;i<menu_list.it_arr.size();i++){
    if(menu_list.it_arr.get(i)==it){ int temp =
    menu_list.it_arr.indexOf(it);
                menu_list.it_arr.get(temp).setStatus("Completed");
            }
        }

    } }
```

<ChefFragment.java>

```
package com.example.easykitchen.ui.notifications;
 import android.app.AlertDialog; import
android.content.DialogInterface;
import android.os.Bundle; import
android.view.LayoutInflater; import
android.view.View; import
android.view.ViewGroup; import
android.widget.AdapterView; import
android.widget.ListView; import
android.widget.TextView;
 import  androidx.annotation.NonNull;  import
androidx.fragment.app.Fragment;       import
androidx.lifecycle.ViewModelProvider;
 import
com.example.easykitchen.ListAdapter;
import com.example.easykitchen.MainScreen;
import com.example.easykitchen.R;
```

```
import com.example.easykitchen.databinding.FragmentNotificationsBinding;
import com.example.easykitchen.item;
import com.example.easykitchen.ui.home.menu_list;

import java.util.ArrayList;
import java.util.List;

public class NotificationsFragment extends Fragment {
    public ArrayList<item> chef_list,curr_menu;
    ListView chef_view;
    View view;
    private FragmentNotificationsBinding binding;

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle
savedInstanceState) {
        NotificationsViewModel notificationsViewModel = new
ViewModelProvider(this).get(NotificationsViewModel.class);

        binding = FragmentNotificationsBinding.inflate(inflater, container,
false);
        View root = binding.getRoot();

        final TextView textView = binding.textNotifications;
        notificationsViewModel.getText().observe(getViewLifecycleOwner(),
textView::setText);



        chef_view = root.findViewById(R.id.chef_list);

        curr_menu = menu_list.it_arr; chef_list = new
        ArrayList<>(); for(int i =0;i<curr_menu.size();i++) { if
        (curr_menu.get(i).getStatus().charAt(0) == 'F') {
        chef_list.add(curr_menu.get(i));
            }
        }


        ListAdapter chefadapter = new
ListAdapter(requireContext(),chef_list);
        chef_view.setAdapter(chefadapter);
        chef_view.setClickable(true);



        chef_view.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int
position, long id) { final int which_item =
                position;
                new AlertDialog.Builder(requireContext())
                        .setIcon(R.drawable.ic_baseline_chef_24)
                        .setTitle("Finished?")
                        .setMessage("Change status to serve")
                        .setPositiveButton("YES", new
DialogInterface.OnClickListener() {
                            @Override
```

```
                                public void onClick(DialogInterface dialog, int
which) { change_curr(chef_list.get(which_item));
                                  chef_list.remove(which_item);
                                  chefadapter.notifyDataSetChanged();

                              }
                          })
                           .setNegativeButton("No",null)
                          .show();
                }
        });
 return root;
    }



    @Override
    public void onDestroyView() {
        super.onDestroyView(); binding
        = null;
    }
 public void change_curr(item it){ for(int
    i=0;i<menu_list.it_arr.size();i++){
    if(menu_list.it_arr.get(i)==it){ int temp =
    menu_list.it_arr.indexOf(it);
                menu_list.it_arr.get(temp).setStatus("Ready to Serve");
            }
        }

    } }
```

5. SOFTWARE TESTING

5.1 Test plan 5.2 Test results and debugging

In computer hardware and software development, testing is used at key checkpoints in the overall process to determine whether objectives are being met.

Types of Testing:

- Unit Testing.
- Integrated Testing • Functional Testing.
- System Testing.
- Performance Testing.

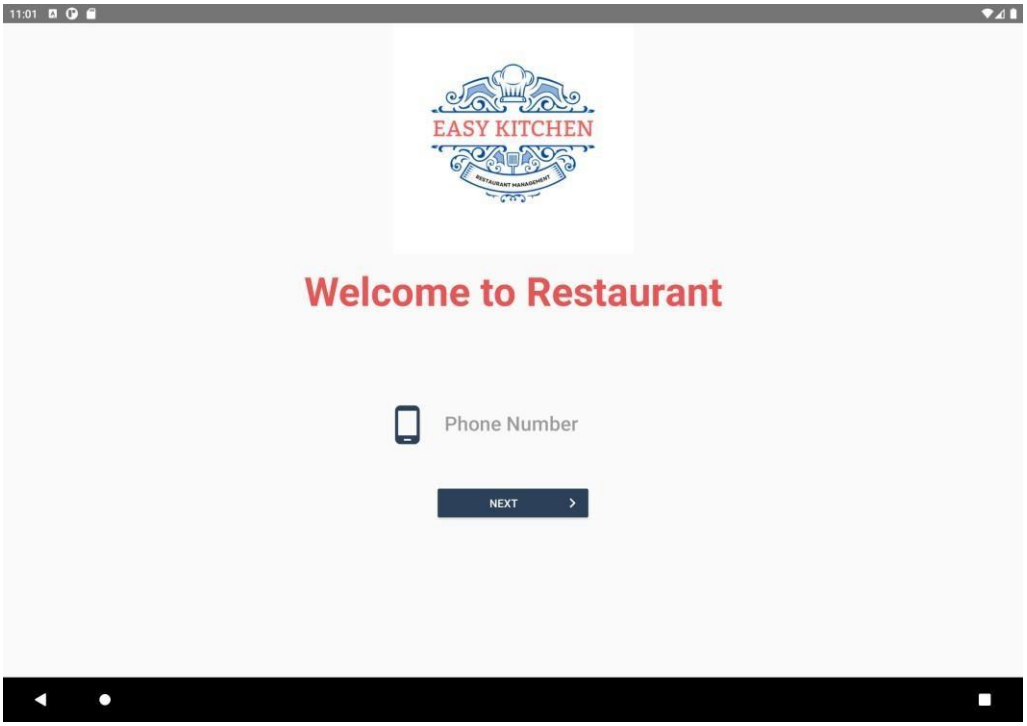
- Usability Testing.
- Beta Testing.

Test Case	Test Scenario	Test Steps	Test Plan	Expected Result	Actual Result	Pass / Fail
TC01	Check login	1. Enter various phone numbers 2. Check if it takes to main page	Only 10-digit phone number login is allowed	Use login to the application	As expected	Pass
TC02	Add Dishes	1. Add some dishes 2. check if its showing in respective tabs	Status should be updated with respective actions.	List and status of dishes appeared	As expected	Pass

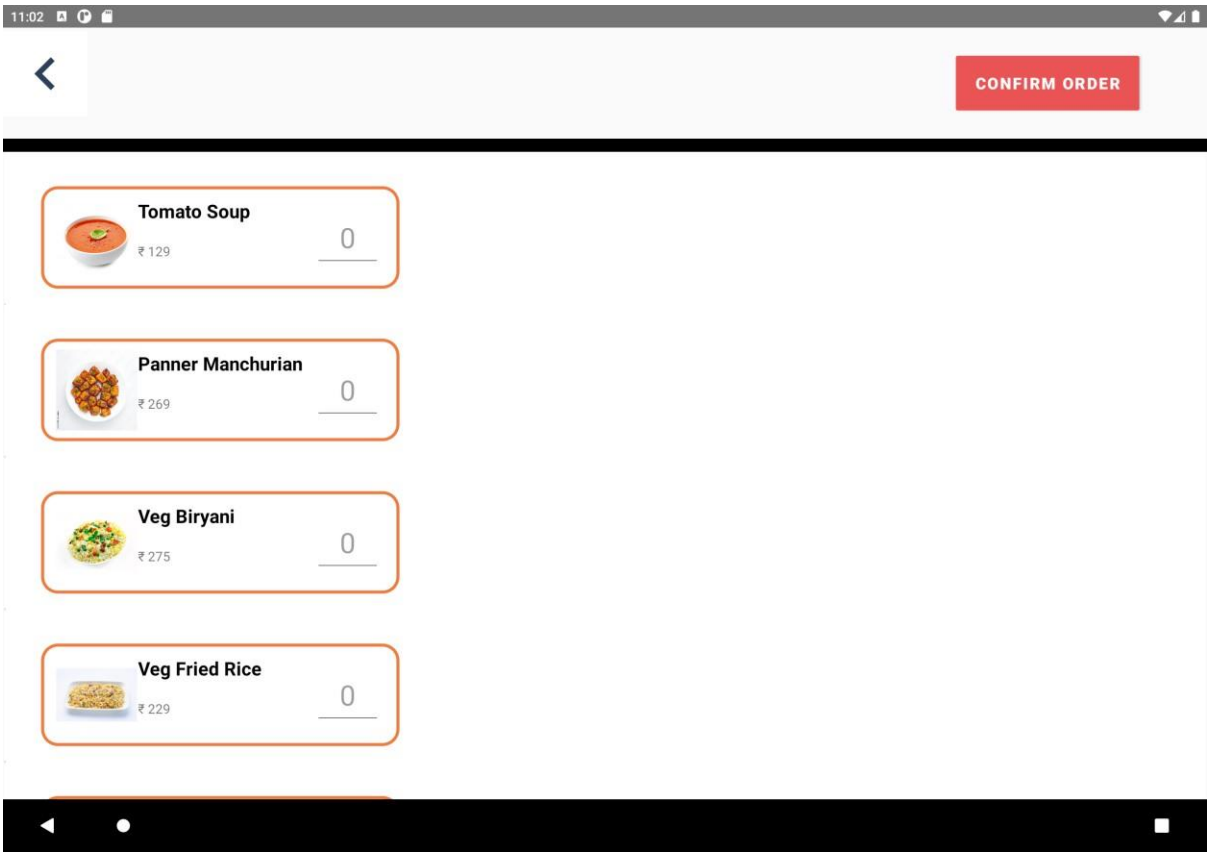
6. PROJECT DEMO:

6.1 Screen shots

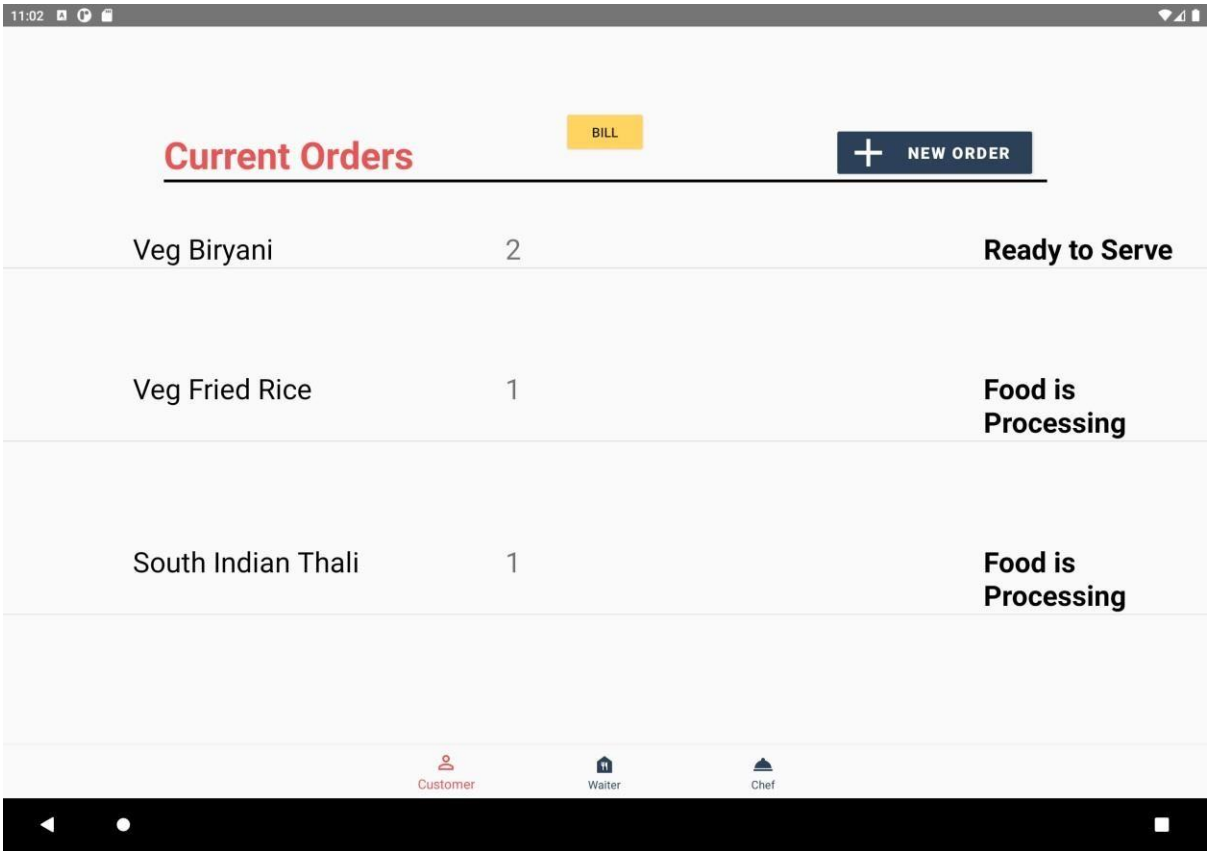
Login page



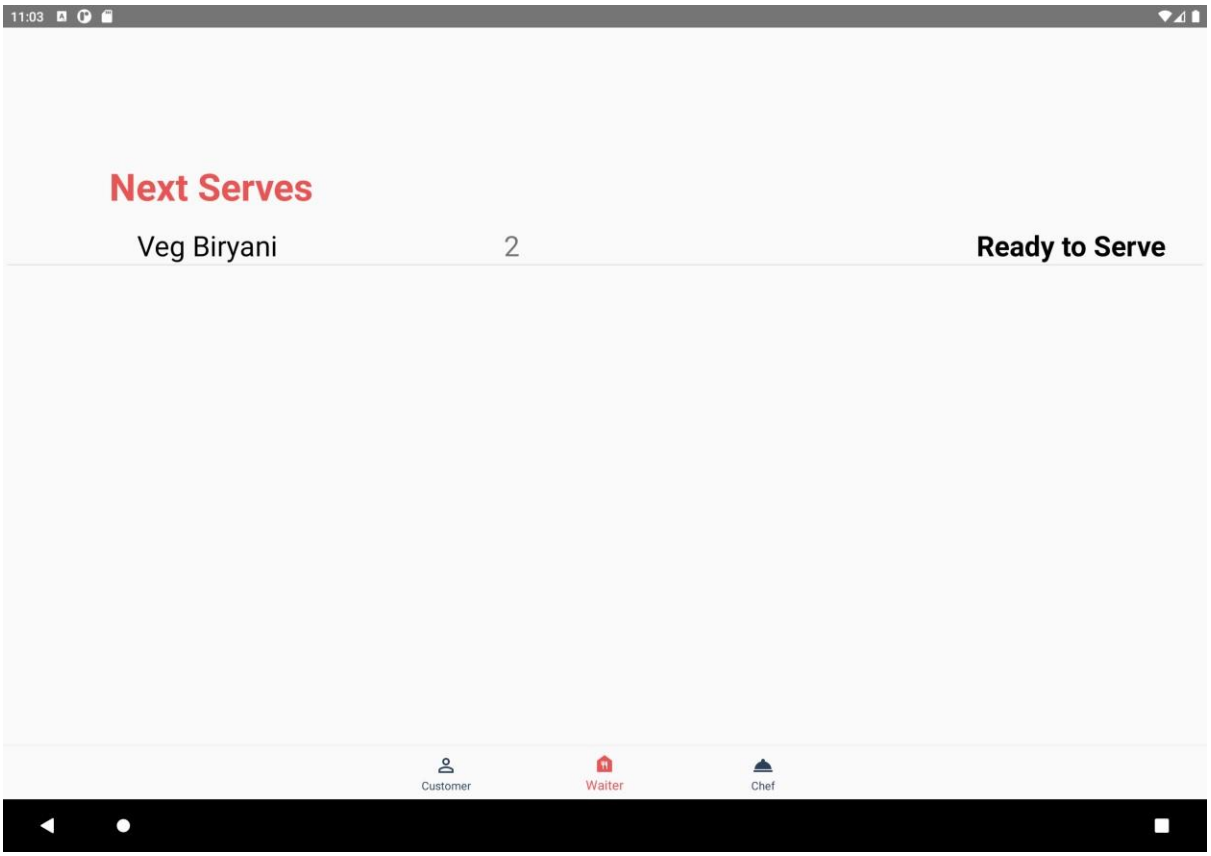
Menu



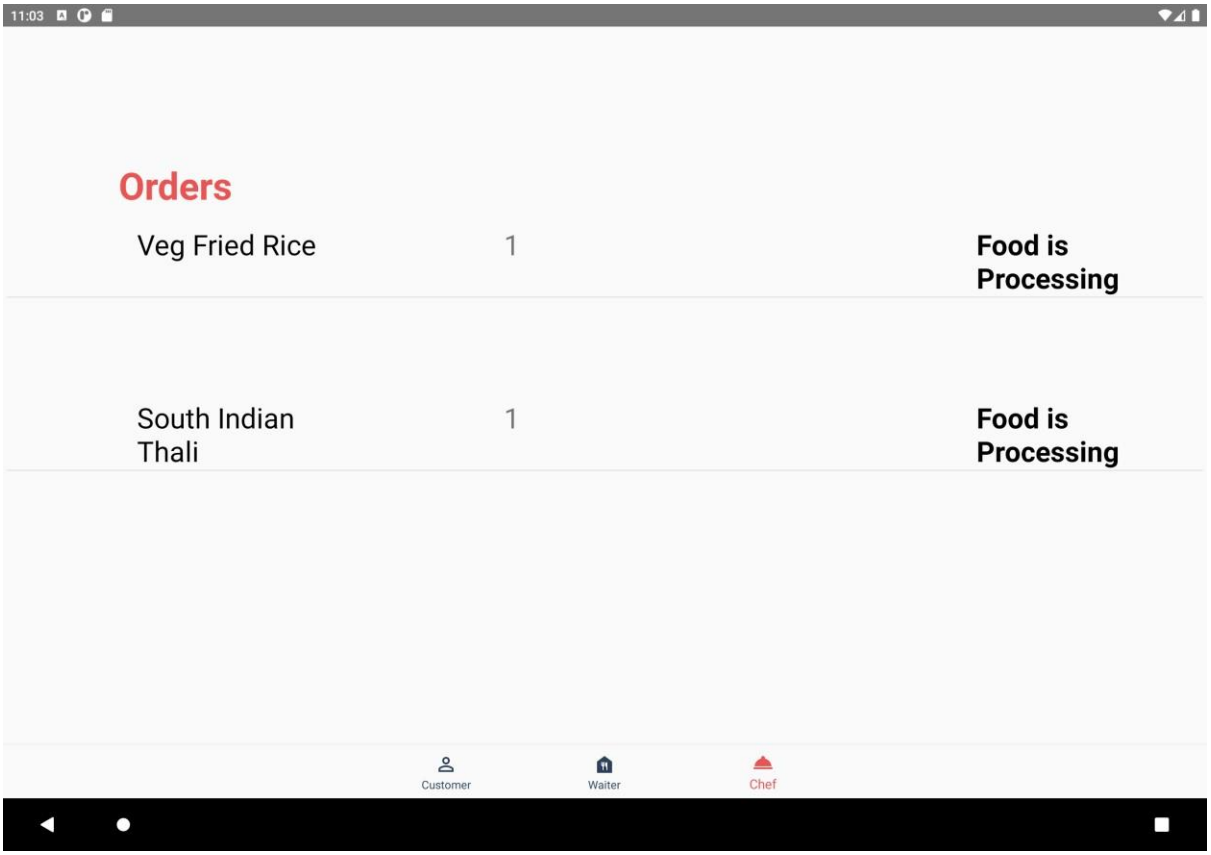
Customer tab



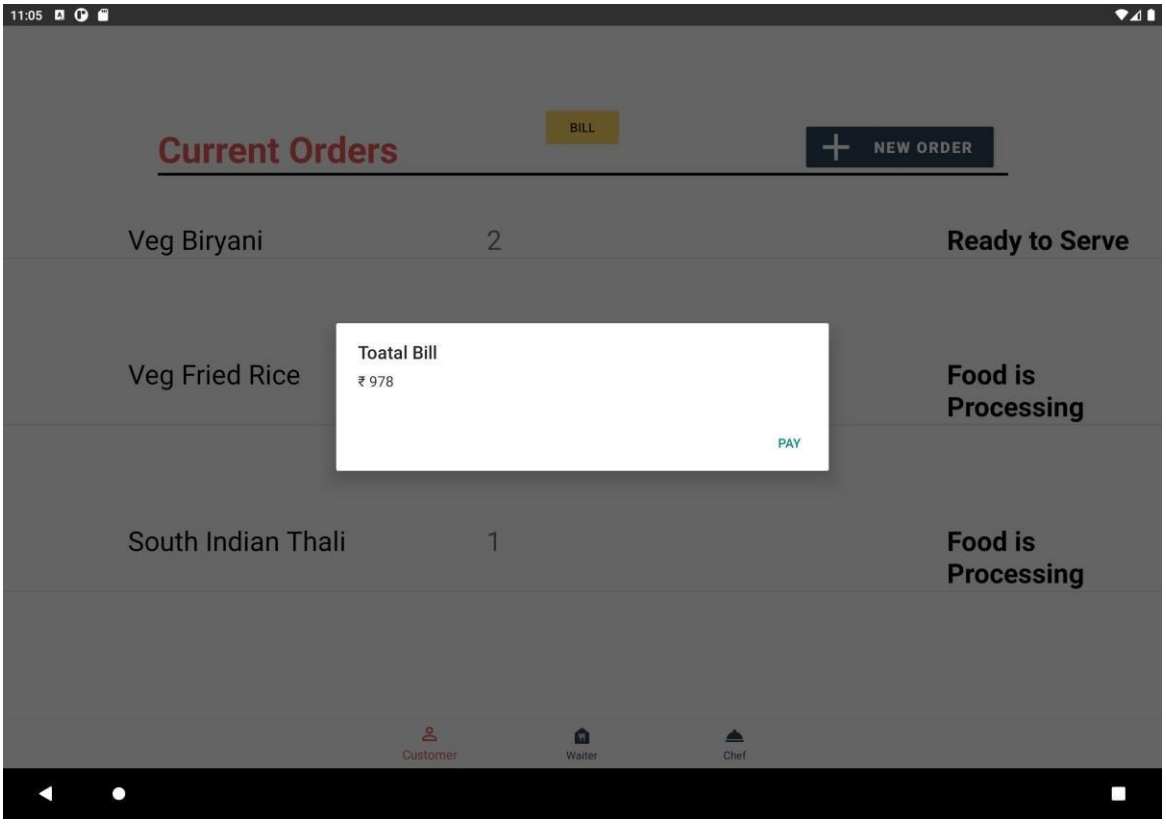
Waiter tab



Chef tab



Total Bill



Demo Video: https://youtu.be/0VXZxxcnxUI