ONLINE MOBILE RECHARGE SYSTEM

FINAL PROJECT

[ASP.NET CORE 2.1 - WEB API /ANGULAR 10 / BOOTSTRAP/ MATERIAL DESIGNS/ JWT/ ENTITY FRAMEWORK / SQL SERVER]

Submitted By:

Naveeth Z A

Rajagurunathan Manikandan

Saravana Kumar B

November - 2020

PROJECT DESCRIPTION

Project Title : Online Mobile Recharge System

Abstract : This System provides prepaid recharge of many network operators as well as all kind of recharge are possible. This System will provide facility for customer to recharge mobile at anytime from anywhere. This System keeps the history of the past recharges. The user has to register himself.

Type of Project: Web API Application

Technology : ASP.Net Core 2.1– Web API, Angular 10, Bootstrap, Material Designs, JWT, Entity Framework

Tools : Visual Studio, Visual Studio code, Postman.

Database : MS SQL-Server

Our Web Application Name:

RPAY

A Payments App built for India, by Indians

DESCRIPTION:

The proposed project for recharging mobiles online is a web application developed to automate the mobile recharging process. It roots out the manual card system of recharge and introduces a new and genuine online recharge process. It can be used by costumer of any telecommunication company from anywhere.

The online mobile recharge system is beneficial to both the admins and users. Using the online application, the admin can add new tariff plans and update or modify the existing tariff plans. It helps the users in creating their account, and then recharging the mobiles phones at any time.

OBJECTIVE:

Online Mobile Recharge is a web-based application developed in .NET & Angular 10 to recharge mobile phones. The project focuses at providing an easy and reliable platform to recharge mobile of any telecommunication company through online without buying recharge card. The registered users of the system can recharge their prepaid mobile phones from anywhere at any time. The proposed project for recharging mobiles developed to automate the mobile recharging process. It roots out the manual card system of recharge and introduces a new and genuine online recharge process. The online mobile recharge system is beneficial to both the admins and users. Using the online application, the admin can add new operators, tariff plans, offers and update or modify the existing tariff plans. It helps the users in creating their account, and then recharging the mobiles phones at any time.

PROJECT DEMO

Type of Users:

1. User / Customer
2. Admin (Jolo API)

Project Flow:

- **User Functionalities:**

1. Signup: User will enter personal & mobile network details
2. Login: Authentication
3. User Home: Users can view recharge plans and various network operator details.
4. Recharge: Users can view recharge plans and perform recharge based on network operator and plan type.
5. My Wallet: Users can add money to their wallet using various payment modes through razor pay payment gateway with SMS authentication. The newly added money will get displayed in my wallet transaction history.
6. Operator Finder: Users can able to view mobile network operators name according to the entered mobile number.
7. My Profile: User can update their personal and mobile network details.

8. Customer Support (Chatbot service): Customer support is achieved through chatbot, which helps the user to solve their queries and book an appointment to contact our support team.

9. Logout: Logout from the application

- **Admin (Jolo API) Functionalities:**

1. Login: Authentication

2. Dashboard (Home): Admin can view transaction count, commission earnings in different visualization format such as pie chart, bar graph and generate reports in pdf format based on the dates filtered.

3. Transactions: Admin can view, update, delete both the success and failed transactions in same page. He can also filter those data based on date & store it in pdf format.

4. Statement: Admin can able to make a monthly statement on user's earnings based on the tarrif plans.

5. Audit: Admin can view, update, delete and search the commission amount details based on the tarrif plans.

6. Logout: Logout from the application

DATABASE DESIGN

Entity Relationship Diagram:

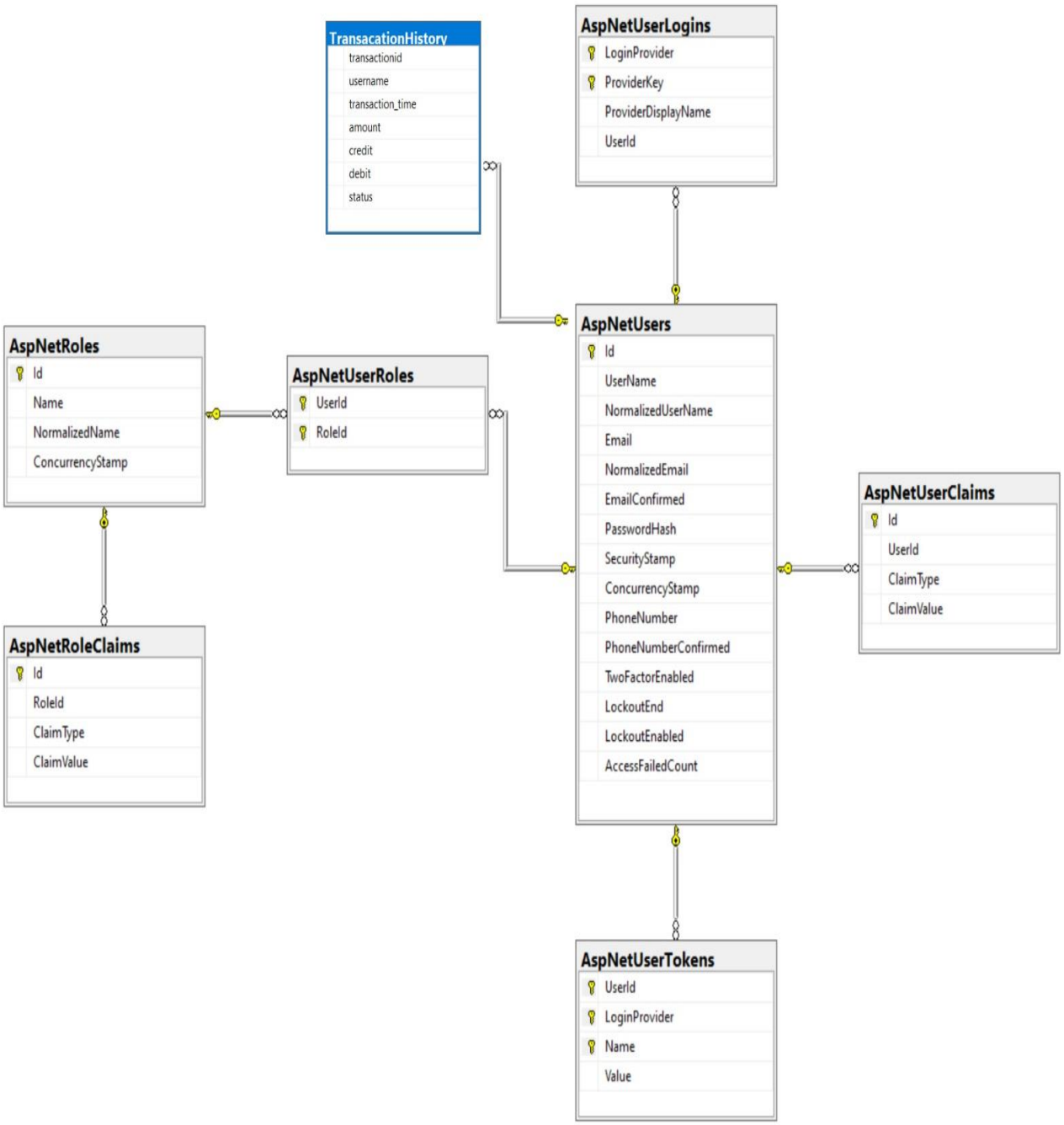


Table Schema Diagram:

TABLE 1: Users

Column Name	Data Type	Allow Nulls
Id	nvarchar(450)	☐
UserName	nvarchar(256)	☑
NormalizedUserName	nvarchar(256)	☑
Email	nvarchar(256)	☑
NormalizedEmail	nvarchar(256)	☑
EmailConfirmed	bit	☐
PasswordHash	nvarchar(MAX)	☑
SecurityStamp	nvarchar(MAX)	☑
ConcurrencyStamp	nvarchar(MAX)	☑
PhoneNumber	nvarchar(MAX)	☑
PhoneNumberConfirmed	bit	☐
TwoFactorEnabled	bit	☐
LockoutEnd	datetimeoffset(7)	☑
LockoutEnabled	bit	☐
AccessFailedCount	int	☐
Discriminator	nvarchar(MAX)	☐
FullName	nvarchar(150)	☑
balance	int	☑
		☐

TABLE 2: User Login

Column Name	Data Type	Allow Nulls
LoginProvider	nvarchar(128)	☐
ProviderKey	nvarchar(128)	☐
ProviderDisplayName	nvarchar(MAX)	☑
UserId	nvarchar(450)	☐
		☐

TABLE 3: User Tokens

Column Name	Data Type	Allow Nulls
UserId	nvarchar(450)	☐
LoginProvider	nvarchar(128)	☐
Name	nvarchar(128)	☐
Value	nvarchar(MAX)	☑
		☐

TABLE 4: User Roles

Column Name	Data Type	Allow Nulls
UserId	nvarchar(450)	☐
RoleId	nvarchar(450)	☐
		☐

TABLE 5: User Claims

Column Name	Data Type	Allow Nulls
Id	int	☐
UserId	nvarchar(450)	☐
ClaimType	nvarchar(MAX)	☑
ClaimValue	nvarchar(MAX)	☑
		☐

TABLE 6: Transactions

Column Name	Data Type	Allow Nulls
transactionid	varchar(500)	☐
username	varchar(250)	☑
transaction_time	datetime	☑
amount	int	☑
credit	varchar(50)	☑
debit	varchar(50)	☑
status	varchar(50)	☑
		☐

SOURCE CODE

ANGULAR 10 CODE

UserService.ts

```
import { Injectable } from '@angular/core';
import { FormBuilder, Validators, FormGroup } from '@angular/forms';
import { HttpClient } from "@angular/common/http";
import { Observable } from 'rxjs';
@Injectable({
  providedIn: 'root'
})
export class UserService {
  httpclient: any;

  constructor(private fb: FormBuilder, private http: HttpClient) { }

    readonly BaseURI ='https://rpay2point0.mrwhitehost.in/api';


  formModel = this.fb.group({
    UserName: ['', Validators.required],
    Email: ['', Validators.email],
    FullName: [''],
    Passwords: this.fb.group({
      Password: ['', [Validators.required, Validators.minLength(4)]],
      ConfirmPassword: ['', Validators.required]
    }, { validator: this.comparePasswords })

  });

  comparePasswords(fb: FormGroup) {
    let confirmPswrdCtrl = fb.get('ConfirmPassword');

    if (confirmPswrdCtrl.errors == null || 'passwordMismatch' in confirmPswrdCtrl.errors) {
      if (fb.get('Password').value != confirmPswrdCtrl.value)
        confirmPswrdCtrl.setErrors({ passwordMismatch: true });
      else
        confirmPswrdCtrl.setErrors(null);
    }
  }

  register() {
    var body = {
      UserName: this.formModel.value.UserName,
      Email: this.formModel.value.Email,
      FullName: this.formModel.value.FullName,
      Password: this.formModel.value.Passwords.Password
    };
    return this.http.post(this.BaseURI + '/ApplicationUser/Register', body);
  }

  login(formData) {
    return this.http.post(this.BaseURI + '/ApplicationUser/Login', formData);
  }

  getUserProfile() {
    return this.http.get(this.BaseURI + '/UserProfile');
  }

  GetAirtelPlans():Observable<any>
  {
    return this.http.get(this.BaseURI+'/Recharge/GetAirtelPlans');
  }
  GetJioPlans():Observable<any>
  {
    return this.http.get(this.BaseURI+'/Recharge/GetJioPlans');
  }
  GetBsnlPlans():Observable<any>
  {
    return this.http.get(this.BaseURI+'/Recharge/GetBsnlPlans');
  }
  GetVodafoneplans():Observable<any>
  {
    return this.http.get(this.BaseURI+'/Recharge/GetVodafonePlans');
  }
  Getuserbl():Observable<any>
  {
    return this.http.get(this.BaseURI+'/UserUpdate/GetMyBalance');
  }

  GetOperator():Observable<any>
  {
    return this.http.get(this.BaseURI+'/OperatorFinder/Operator');
  }

  GetWalletHistory():Observable<any>
  {
    return this.http.get(this.BaseURI+'/Wallet/WalletTranaction');
  }

  GetMyDetails():Observable<any>
  {
    return this.http.get(this.BaseURI+'/UserUpdate/GetMyDetails');
  }
}
```

Login.ts

```ts
import { ToastrService } from 'ngx-toastr';
import { UserService } from './../../shared/user.service';
import { Component, OnInit } from '@angular/core';
import { NgForm } from '@angular/forms';
import { Router } from '@angular/router';

@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls:['./login.component.css']
})
export class LoginComponent implements OnInit {
  formModel = {
    UserName: '',
    Password: ''
  }
  constructor(private service: UserService, private router: Router, private toastr: ToastrService) { }

  ngOnInit() {
    if (localStorage.getItem('token') != null)
      this.router.navigateByUrl('/home');
  }

  onSubmit(form: NgForm) {
    this.service.login(form.value).subscribe(
      (res: any) => {
        localStorage.setItem('token', res.token);
        this.router.navigateByUrl('/home');
      },
      err => {
        if (err.status == 400)
          this.toastr.error('Incorrect username or password.', 'Authentication failed.');
        else
          console.log(err);
      }
    );
  }
}
```

Auth.guard.ts

```ts
import { Injectable } from '@angular/core';
import { CanActivate, ActivatedRouteSnapshot, RouterStateSnapshot, Router } from '@angular/router';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class AuthGuard implements CanActivate {


  constructor(private router: Router) {
  }
  canActivate(
    next: ActivatedRouteSnapshot,
    state: RouterStateSnapshot): boolean {
    if (localStorage.getItem('token') != null)
      return true;
    else {
      this.router.navigate(['/user/login']);
      return false;
    }

  }
}
```

Interceptor.ts

```ts
import { HttpInterceptor, HttpRequest, HttpHandler, HttpEvent } from "@angular/common/http";
import { Injectable } from "@angular/core";
import { Observable } from "rxjs";
import { tap } from "rxjs/operators";
import { Router } from "@angular/router";

@Injectable()
export class AuthInterceptor implements HttpInterceptor {

    constructor(private router: Router) {

    }

    intercept(req: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {
        if (localStorage.getItem('token') != null) {
            const clonedReq = req.clone({
                headers: req.headers.set('Authorization', 'Bearer ' + localStorage.getItem('token'))
            });
            return next.handle(clonedReq).pipe(
                tap(
                    succ => { },
                    err => {
                        if (err.status == 401){
                            localStorage.removeItem('token');
                            this.router.navigateByUrl('/user/login');
                        }
                    }
                )
            )
        }
        else
            return next.handle(req.clone());
    }
}
```

Operator Finder.ts

```
import { Component, Injectable, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { Plans} from 'src/app/shared/plans';
import { UserService } from 'src/app/shared/user.service';
import {HttpClient} from '@angular/common/http'
import { data } from 'jquery';
import { OperatorFinder } from 'src/app/shared/operatorfinder';
import { stringify } from 'querystring';

@Component({
  selector: 'app-operatorfinder',
  templateUrl: './operatorfinder.component.html',
  styleUrls: ['./operatorfinder.component.css']
})
@Injectable()
export class OperatorfinderComponent implements OnInit {

  constructor(private router: Router, private service: UserService , private http:HttpClient) { }

  lstOperator:any=[];

  phonenumber:string;

  onSubmit(data)
  {
    this.phonenumber = data;
    this.http.get('https://rpay2point0.mrwhitehost.in/api/OperatorFinder/Operator?phonenumber='+
data.phonenumber)
    .subscribe(
      data=>
      {
        this.lstOperator=data;
      }
    );

    console.log(data);
  }


  ngOnInit(): void {

  }

}
```

Recharge.ts

```
import { UserService } from './../../shared/user.service';
import { Component, OnInit } from '@angular/core';
import { ToastrService } from 'ngx-toastr';

@Component({
  selector: 'app-registration',
  templateUrl: './registration.component.html',
  styleUrls: ['./registration.component.css']
})
export class RegistrationComponent implements OnInit {

  constructor(public service: UserService, private toastr: ToastrService) { }

  ngOnInit() {
    this.service.formModel.reset();
  }

  onSubmit() {
    this.service.register().subscribe(
      (res: any) => {
        if (res.succeeded) {
          this.service.formModel.reset();
          this.toastr.success('New user created!', 'Registration successful.');
        } else {
          res.errors.forEach(element => {
            switch (element.code) {
              case 'DuplicateUserName':
                this.toastr.error('Username is already taken','Registration failed.');
                break;

              default:
              this.toastr.error(element.description,'Registration failed.');
                break;
            }
          });
        }
      },
      err => {
        console.log(err);
      }
    );
  }

}
```

Mywallet.ts

```
import { HttpClient } from '@angular/common/http';
import { Component, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { UserService } from 'src/app/shared/user.service';

@Component({
  selector: 'app-mywallet',
  templateUrl: './mywallet.component.html',
  styleUrls: ['./mywallet.component.css']
})
export class MywalletComponent implements OnInit {

  userbalance='';
  constructor(private router: Router, private service: UserService , private http:HttpClient) { }
  ngOnInit(){
    this.service.Getuserbl()
    .subscribe(
      data=>
      {
        this.userbalance=data;

      console.log(this.userbalance);
      }
    );
  }

}
```

Wallet History.ts

```
import { Component, Injectable, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { Plans} from 'src/app/shared/plans';
import { UserService } from 'src/app/shared/user.service';
import {HttpClient} from '@angular/common/http'
import { data } from 'jquery';
import { OperatorFinder } from 'src/app/shared/operatorfinder';
import { stringify } from 'querystring';
import { WalletHistory} from 'src/app/shared/wallethistory';

@Component({
  selector: 'app-wallethistory',
  templateUrl: './wallethistory.component.html',
  styleUrls: ['./wallethistory.component.css']
})
@Injectable()
export class WallethistoryComponent implements OnInit {

  constructor(private router: Router, private service: UserService , private http:HttpClient) { }

  lsttransaction:any=[];

  ngOnInit(): void
  {
    this.service.GetWalletHistory()
    .subscribe(
      data=>
      {
        this.lsttransaction=data;
      }
    );
  }

}
```

UserUpdate.ts

```
import { Component, Injectable, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { Plans} from 'src/app/shared/plans';
import { UserService } from 'src/app/shared/user.service';
import {HttpClient} from '@angular/common/http'
import { data } from 'jquery';
import { OperatorFinder } from 'src/app/shared/operatorfinder';
import { stringify } from 'querystring';
import { Observable } from 'rxjs';
import { FormBuilder, FormGroup } from "@angular/forms";


@Component({
  selector: 'app-userupdate',
  templateUrl: './userupdate.component.html',
  styleUrls: ['./userupdate.component.css']
})


@Injectable()
export class UserupdateComponent implements OnInit {

  constructor(private router: Router, private service: UserService , private http:HttpClient) { }

  lstdetails:any=[];
  lstupdate:any=[];

  username='RajaDemo';

  update()
  {
    this.http.get('https://rpay2point0.mrwhitehost.in/api/UserUpdate/UpdateProfile?
fname='+this.name+'&mail='+this.email+'&phno='+this.phonenumber)
    .subscribe(
      data=>
      {
        this.lstupdate=data;
        console.log(data);
      }
    );
  }

  name:string;
  email:string;
  phonenumber:string;


  ngOnInit(): void
  {
    this.http.get('https://rpay2point0.mrwhitehost.in/api/UserUpdate/GetMyDetails')
    .subscribe(
      data=>
      {
        this.lstdetails=data;
        this.name=this.lstdetails[0].fullName;
        this.email=this.lstdetails[0].email;
        this.phonenumber=this.lstdetails[0].phoneNumber;

        console.log(data);

        console.log(this.lstdetails[0].fullName);
        console.log(this.lstdetails[0].email);
        console.log(this.lstdetails[0].phoneNumber);

        console.log(this.name);
        console.log(this.email);
        console.log(this.phonenumber);



      }
    );

  }

}
```

App Routing Module.ts

```ts
import { AuthGuard } from './auth/auth.guard';
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { UserComponent } from './user/user.component';
import { RegistrationComponent } from './user/registration/registration.component';
import { LoginComponent } from './user/login/login.component';
import { HomeComponent } from './home/home.component';
import { RechargeComponent } from './home/recharge/recharge.component';
import { MywalletComponent } from './home/mywallet/mywallet.component';
import { WallethistoryComponent } from './home/wallethistory/wallethistory.component';
import { OperatorfinderComponent } from './home/operatorfinder/operatorfinder.component';
import { UserupdateComponent } from './home/userupdate/userupdate.component';
import { FrontPageComponent } from './front-page/front-page.component';

const routes: Routes = [
  {path:'',redirectTo:'/Homepage',pathMatch:'full'},
  {
    path: 'user', component: UserComponent,
    children: [
      { path: 'registration', component: RegistrationComponent },
      { path: 'login', component: LoginComponent }
    ]
  },
  {path:'home',component:HomeComponent,canActivate:[AuthGuard]},
  {path:'recharge',component:RechargeComponent,canActivate:[AuthGuard]},
  {path:'mywallet',component:MywalletComponent,canActivate:[AuthGuard]},
  {path:'wallethistory',component:WallethistoryComponent,canActivate:[AuthGuard]},
  {path:'operatorfinder',component:OperatorfinderComponent,canActivate:[AuthGuard]},
  {path:'userupdate',component:UserupdateComponent,canActivate:[AuthGuard]},
  {path:'Homepage',component:FrontPageComponent}
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

.NET CORE 2.1 WEB API CODES:

Application User Controller

```
using System;
using System.Collections.Generic;
using System.IdentityModel.Tokens.Jwt;
using System.Linq;
using System.Security.Claims;
using System.Text;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Cors;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Options;
using Microsoft.IdentityModel.Tokens;
using WebAPI.Models;

namespace WebAPI.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    [EnableCors("AllowMyOrigin")]
    public class ApplicationUserController : ControllerBase
    {
        private UserManager<ApplicationUser> _userManager;
        private SignInManager<ApplicationUser> _singInManager;
        private readonly ApplicationSettings _appSettings;

        public ApplicationUserController(UserManager<ApplicationUser> userManager,
SignInManager<ApplicationUser> signInManager,IOptions<ApplicationSettings> appSettings)
        {
            _userManager = userManager;
            _singInManager = signInManager;
            _appSettings = appSettings.Value;
        }

        [HttpPost]
        [Route("Register")]
        [EnableCors("AllowMyOrigin")]
        //POST : /api/ApplicationUser/Register
        public async Task<Object> PostApplicationUser(ApplicationUserModel model)
        {
            var applicationUser = new ApplicationUser() {
                UserName = model.UserName,
                Email = model.Email,
                FullName = model.FullName
            };

            try
            {
                var result = await _userManager.CreateAsync(applicationUser, model.Password);
                return Ok(result);
            }
            catch (Exception ex)
            {

                throw ex;
            }
        }

        [HttpPost]
        [Route("Login")]
        [EnableCors("AllowMyOrigin")]
        //POST : /api/ApplicationUser/Login
        public async Task<IActionResult> Login(LoginModel model)
        {
            var user = await _userManager.FindByNameAsync(model.UserName);
            if (user != null && await _userManager.CheckPasswordAsync(user, model.Password))
            {
                var tokenDescriptor = new SecurityTokenDescriptor
                {
                    Subject = new ClaimsIdentity(new Claim[]
                    {
                        new Claim("UserID",user.Id.ToString())
                    }),
                    Expires = DateTime.UtcNow.AddDays(1),
                    SigningCredentials = new SigningCredentials(new
SymmetricSecurityKey(Encoding.UTF8.GetBytes(_appSettings.JWT_Secret)),
SecurityAlgorithms.HmacSha256Signature)

                };
                var tokenHandler = new JwtSecurityTokenHandler();
                var securityToken = tokenHandler.CreateToken(tokenDescriptor);
                var token = tokenHandler.WriteToken(securityToken);
                HttpContext.Session.Remove("Email");
                HttpContext.Session.Remove("FullName");
                HttpContext.Session.Remove("UserName");
                return Ok(new { token });
            }
            else
                return BadRequest(new { message = "Username or password is incorrect." });
        }
    }
}
```

Recharge Controller

```
        //Get : /api/Recharge/RechargeComplete
        [Authorize]
        [HttpPost]
        [EnableCors("AllowMyOrigin")]
        [Route("RechargeComplete")]
        public async Task<ActionResult> RechargeComplete([FromBody] RechargeModel obj)
        {
            string userId = User.Claims.First(c => c.Type == "UserID").Value;
            var user = await _userManager.FindByIdAsync(userId);
            var username = user.UserName;
            int amount = Convert.ToInt32(obj.Amount);
            string phonenumber = obj.PhoneNumber;
            string myoperator = obj.Operator;

            string q_balance = "select balance from aspnetusers where UserName = '" + username+"'";

            con.Open();

            SqlCommand cmd = new SqlCommand(q_balance, con);

            int balance = Convert.ToInt32(cmd.ExecuteScalar());

            if(balance < amount)
            {

            }
            else
            {
                Random rdm = new Random();
                string randomnumber = Convert.ToString(rdm.Next());


                string uri = "https://joloapi.com/api/v1/recharge.php?"+credentials1+"&operator=" +
myoperator + "&service=" + phonenumber + "&amount=" + amount + "&orderid=" + randomnumber;

                var client = new HttpClient();
                HttpResponseMessage response = await client.GetAsync(uri);
                response.EnsureSuccessStatusCode();
                var result = JsonConvert.DeserializeObject<dynamic>(await
response.Content.ReadAsStringAsync());


                DateTime dt = DateTime.Now;

                string q_transact = "insert into TransacationHistory
values('DBT"+randomnumber+"','"+username+"','"+dt+"',"+amount+",'false','true','success')";

                SqlCommand cmd2 = new SqlCommand(q_transact, con);

                cmd2.ExecuteNonQuery();

                string q_update_balance = "update aspnetusers set balance = balance - "+amount+" where
UserName = '"+username+"'";

                SqlCommand cmd3 = new SqlCommand(q_update_balance, con);

                cmd3.ExecuteNonQuery();

                con.Close();

                return Ok(result);


            }

            return Ok("Out of Recharge");

        }
```

Wallet Controller

```
using System;
using System.Collections.Generic;
using System.IdentityModel.Tokens.Jwt;
using System.Linq;
using System.Security.Claims;
using System.Text;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Options;
using Microsoft.IdentityModel.Protocols;
using Microsoft.IdentityModel.Tokens;
using WebAPI.Models;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System.Data.Common;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Cors;

namespace WebAPI.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    [EnableCors("AllowMyOrigin")]
    public class WalletController : ControllerBase
    {

        SqlConnection con = new SqlConnection("Server=mssql.mrwhitehost.in;Database=rpay;Persist
Security Info=True;User ID=rpaybase;Password=5$Z00gmb;MultipleActiveResultSets=True;");

        private UserManager<ApplicationUser> _userManager;
        public WalletController(UserManager<ApplicationUser> userManager)
        {
            _userManager = userManager;
        }
        [Authorize]
        [HttpGet]
        [EnableCors("AllowMyOrigin")]
        [Route("WalletTranaction")]
        //POST : /api/Wallet/WalletTranaction
        public async Task<IActionResult> WalletTranactionAsync()
        {
            con.Open();

            string userId = User.Claims.First(c => c.Type == "UserID").Value;
            var user = await _userManager.FindByIdAsync(userId);
            var username = user.UserName;

            DataTable dt = new DataTable();

            String q = "select  * from TransacationHistory where UserName = '"+ username + "'and credit
= 'true' order by transaction_time  desc";

            SqlDataAdapter ad = new SqlDataAdapter(q, con);


            ad.Fill(dt);


            return Ok(dt);


        }


    }
}
```

Operator Finder Controller

```
using System;
using System.Collections.Generic;
using System.IdentityModel.Tokens.Jwt;
using System.Linq;
using System.Security.Claims;
using System.Text;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Options;
using Microsoft.IdentityModel.Protocols;
using Microsoft.IdentityModel.Tokens;
using WebAPI.Models;
using System.Configuration;
using System.Data.SqlClient;
using System.Data;
using System.Data.Common;
using System.Net.Http;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Cors;
using Newtonsoft.Json;
using Newtonsoft.Json.Linq;

namespace WebAPI.Controllers
{
    [Authorize]
    [Route("api/[controller]")]
    [ApiController]
    [EnableCors("AllowMyOrigin")]
    public class OperatorFinderController : ControllerBase
    {
        private string credentials1 = "userid=userid&key=123456789";
        private string credentials = "userid=userid&key=123456789";
        private UserManager<ApplicationUser> _userManager;
        public OperatorFinderController(UserManager<ApplicationUser> userManager)
        {
            _userManager = userManager;
        }
        //Get : /api/OperatorFinder/Operator
        [HttpGet]
        [Route("Operator")]
        [EnableCors("AllowMyOrigin")]
        public async Task<ActionResult> OperatorAsync([FromQuery]string phonenumber)
        {
            string uri = "https://joloapi.com/api/v1/operatorfinder.php?"+credentials+"&mob=" +
phonenumber;
            var client = new HttpClient();
            HttpResponseMessage response = await client.GetAsync(uri);
            response.EnsureSuccessStatusCode();
            var result = JsonConvert.DeserializeObject<dynamic>(await
response.Content.ReadAsStringAsync());

            return Ok(result);


        }

    }
}
```

User Update Controller

```
        }
        [Authorize]
        [HttpGet]
        [Route("UpdateProfile")]
        [EnableCors("AllowMyOrigin")]
        //POST : /api/UserUpdate/UpdateProfile
        public async Task<IActionResult> UpdateProfile([FromQuery] string fname, [FromQuery] string
mail, [FromQuery] string phno)
        {
            con.Open();

            DataTable dt = new DataTable();

            string userId = User.Claims.First(c => c.Type == "UserID").Value;
            var user = await _userManager.FindByIdAsync(userId);
            var uname = user.UserName;

            string fullname = fname;
            string email = mail;
            string phone = phno;

            string q_update = "update aspnetusers set FullName = '" + fullname + "' , email = '" +
email + "', PhoneNumber = '" + phone + "' where UserName = '" + uname + "'";

            SqlCommand cmd = new SqlCommand(q_update, con);
            cmd.ExecuteNonQuery();

            String q = "select * from aspnetusers where UserName = '" + uname + "'";

            SqlDataAdapter ad = new SqlDataAdapter(q, con);


            ad.Fill(dt);
            con.Close();

            return Ok(dt);
        }
        [Authorize]
        [HttpGet]
        [Route("GetMyDetails")]
        [EnableCors("AllowMyOrigin")]
        //POST : /api/UserUpdate/GetMyDetails
        public async Task<IActionResult> GetMyDetails()
        {
            con.Open();

            string userId = User.Claims.First(c => c.Type == "UserID").Value;
            var user = await _userManager.FindByIdAsync(userId);
            var uname = user.UserName;

            DataTable dt = new DataTable();
            String q = "select * from aspnetusers where UserName = '" + uname + "'";

            SqlDataAdapter ad = new SqlDataAdapter(q, con);


            ad.Fill(dt);
            con.Close();

            return Ok(dt);
        }

        //GET : /api/UserUpdate/GetMyBalance
        [Authorize]
        [HttpGet]
        [Route("GetMyBalance")]
        [EnableCors("AllowMyOrigin")]
        public async Task<IActionResult> GetMyBalanceAsync()
        {
            con.Open();

            string userId = User.Claims.First(c => c.Type == "UserID").Value;
            var user = await _userManager.FindByIdAsync(userId);
            var username = user.UserName;

            //create session for username

            string q_balance = "select balance from aspnetusers where UserName ='"+ username+"'";

            SqlCommand cmd = new SqlCommand(q_balance, con);

            int balance = Convert.ToInt32(cmd.ExecuteScalar());
            con.Close();
            return Ok(balance);
        }
    }
```

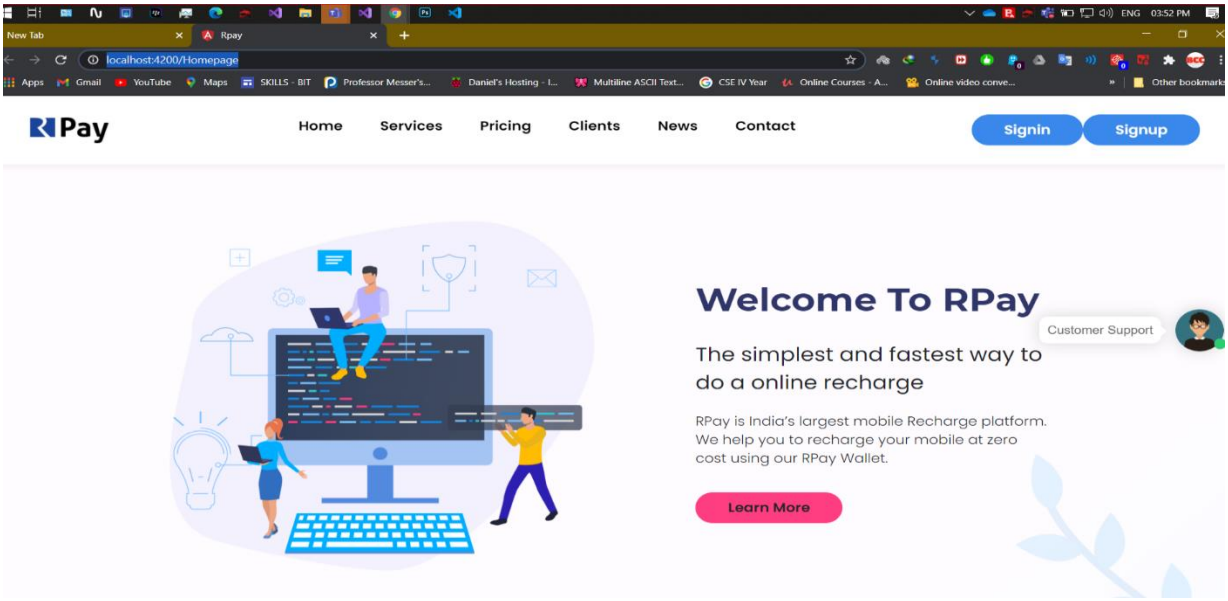
Startup.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Authentication.JwtBearer;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Session;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Logging;
using Microsoft.Extensions.Options;
using Microsoft.IdentityModel.Tokens;
using WebAPI.Models;

namespace WebAPI
{
    public class Startup
    {

        public Startup(IConfiguration configuration)
        {
            Configuration = configuration;
        }

        public IConfiguration Configuration { get; }


        public void ConfigureServices(IServiceCollection services)
        {

            services.Configure<ApplicationSettings>(Configuration.GetSection("ApplicationSettings"));
            services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Latest);
            services.AddDistributedMemoryCache();
            services.AddSession();
            services.AddDbContext<AuthenticationContext>(options =>
            options.UseSqlServer(Configuration.GetConnectionString("IdentityConnection")));
            services.AddDefaultIdentity<ApplicationUser>()
                .AddEntityFrameworkStores<AuthenticationContext>();
            services.Configure<IdentityOptions>(options =>
            {
                options.Password.RequireDigit = false;
                options.Password.RequireNonAlphanumeric = false;
                options.Password.RequireLowercase = false;
                options.Password.RequireUppercase = false;
                options.Password.RequiredLength = 4;
            }
            );
            services.AddCors(options =>
            {
                options.AddPolicy("AllowMyOrigin",
                    builder => builder.WithOrigins("http://localhost:4200",
"https://joloapi.com/").AllowAnyHeader().AllowAnyMethod().AllowCredentials().Build());
            });


            var key =
Encoding.UTF8.GetBytes(Configuration["ApplicationSettings:JWT_Secret"].ToString());

            services.AddAuthentication(x =>
            {
                x.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
                x.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
                x.DefaultScheme = JwtBearerDefaults.AuthenticationScheme;
            }).AddJwtBearer(x=> {
                x.RequireHttpsMetadata = false;
                x.SaveToken = false;
                x.TokenValidationParameters = new
Microsoft.IdentityModel.Tokens.TokenValidationParameters
                {
                    ValidateIssuerSigningKey = true,
                    IssuerSigningKey = new SymmetricSecurityKey(key),
                    ValidateIssuer = false,
                    ValidateAudience = false,
                    ClockSkew = TimeSpan.Zero
                };
            });
        }

        public void Configure(IApplicationBuilder app, IHostingEnvironment env)
        {
            app.UseCors("AllowMyOrigin");

            app.Use(async (ctx, next) =>
            {
                await next();
                if (ctx.Response.StatusCode == 204)
                {
                    ctx.Response.ContentLength = 0;
                }
            });

            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }
            app.UseAuthentication();
            app.UseSession();
            app.UseMvc();
        }
    }
}
```

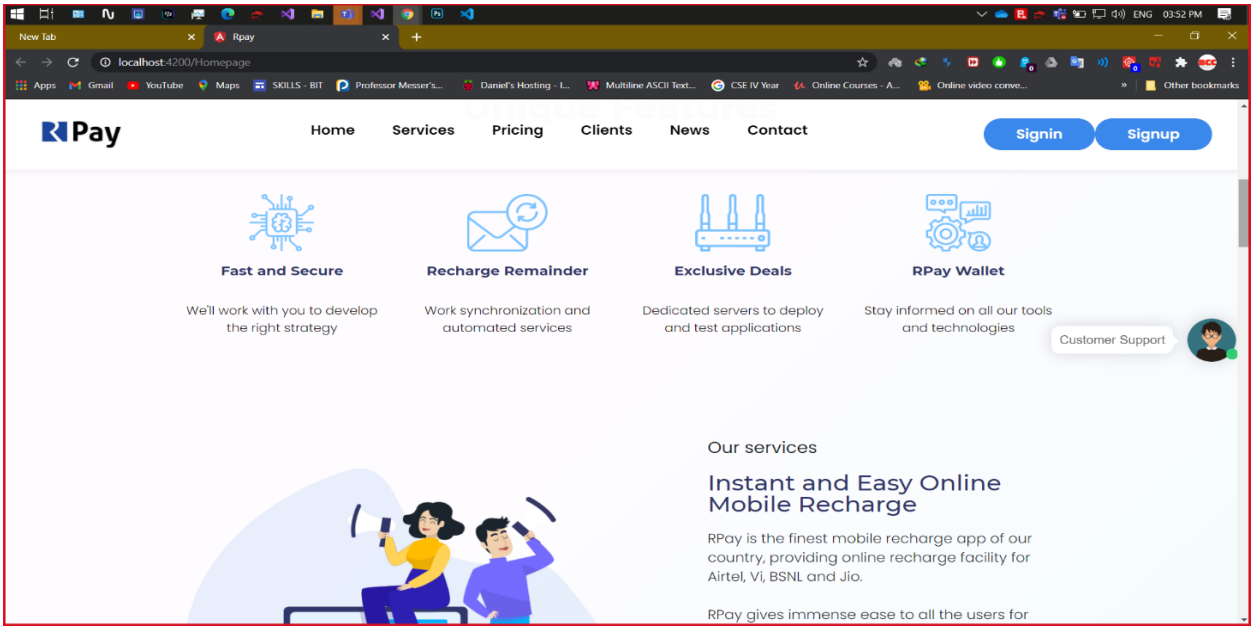
PROJECT OUTPUT SCREENSHOTS

HOME PAGE

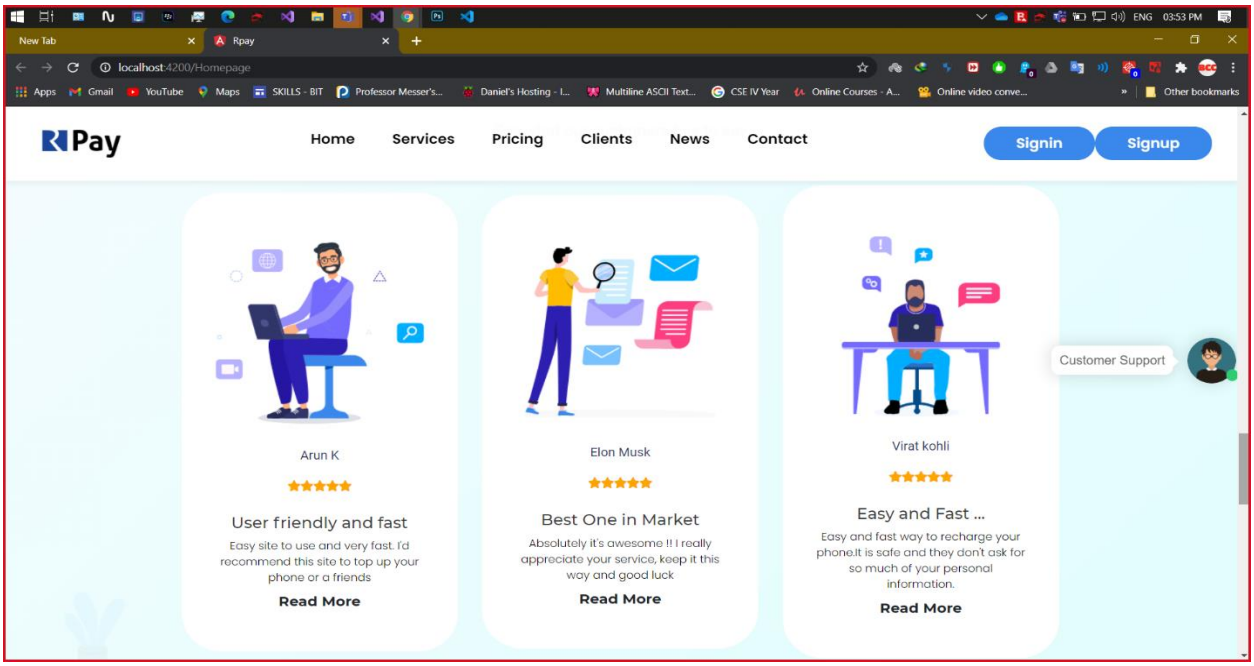
Screenshot 1.1 Welcome to Rpay



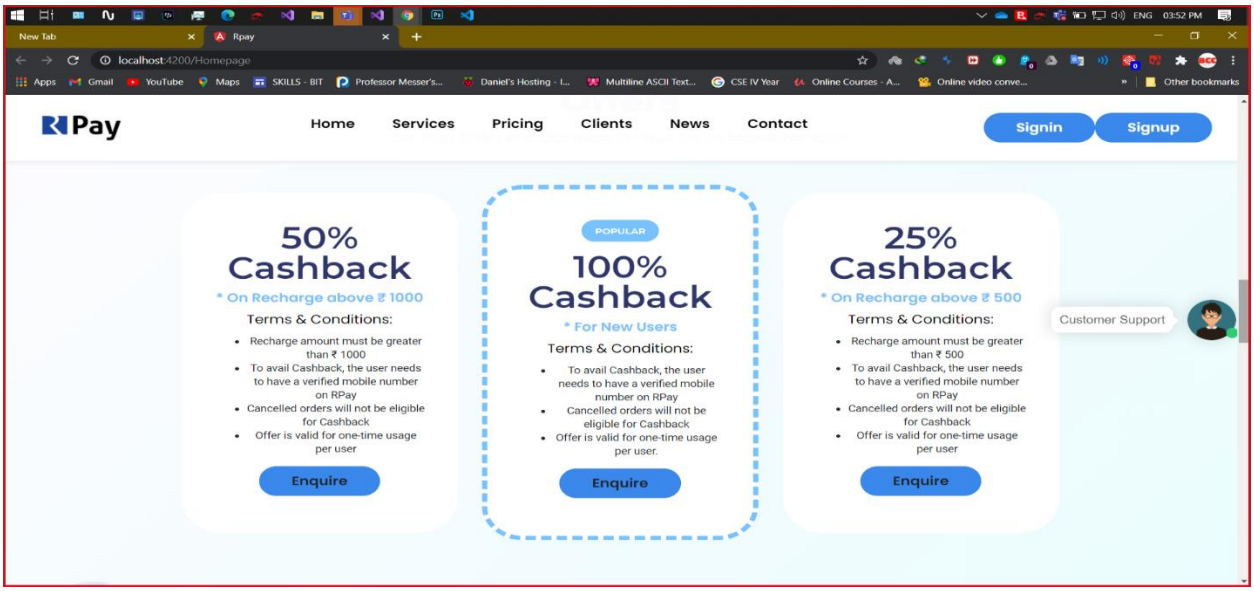
Screenshot 1.2 Our Salient Features



Screenshot 1.3 Customer Reviews

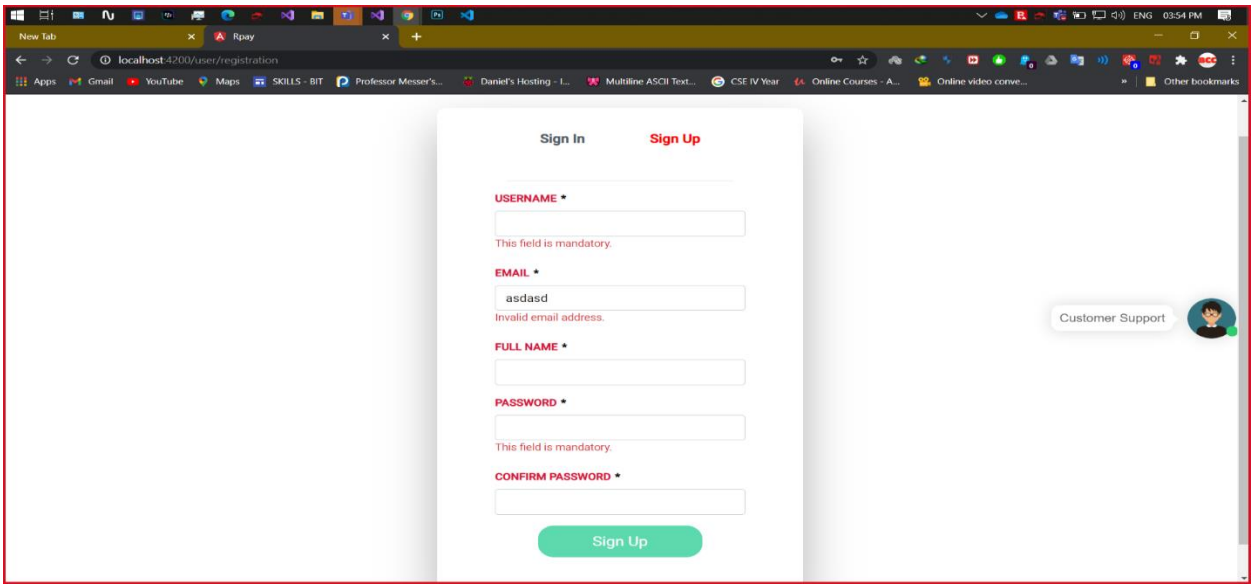


Screenshot 1.4 Cashback Offers



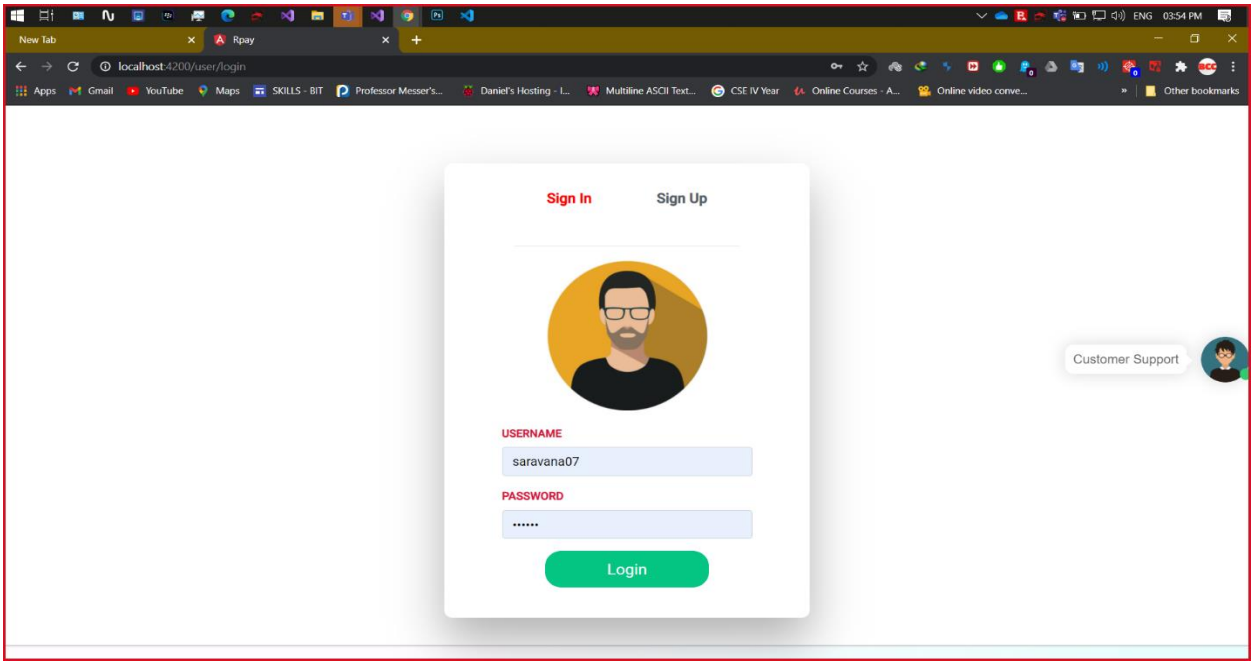
SIGN UP

Screenshot 2.1 SignUp

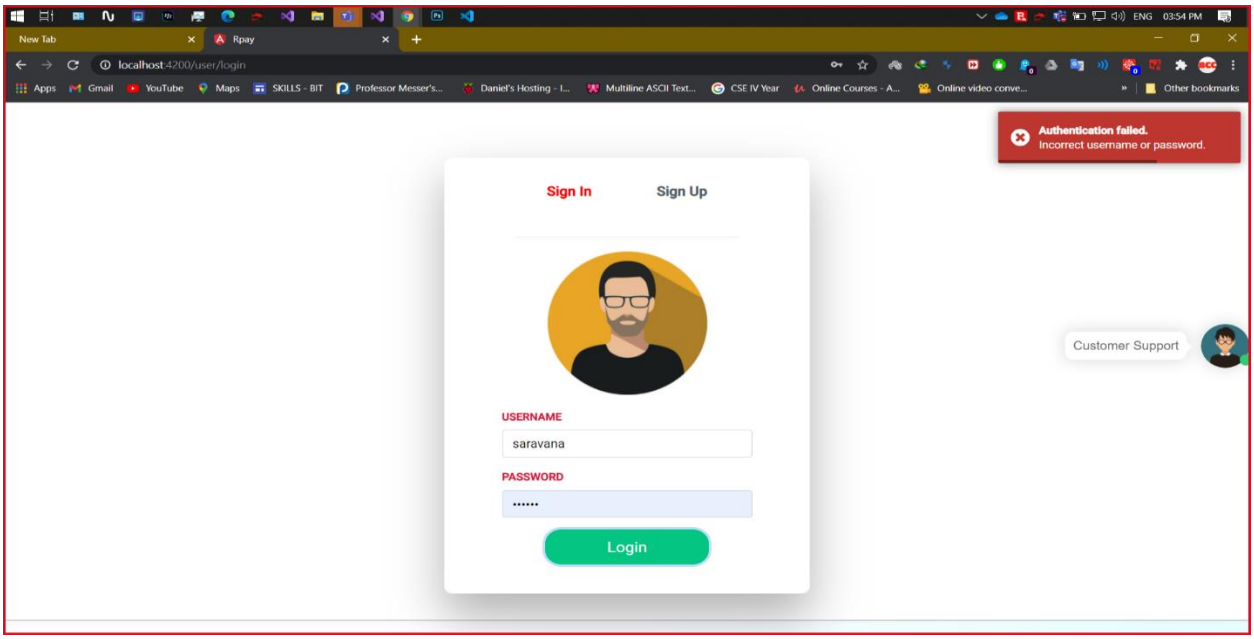


SIGN IN

Screenshot 3.1 Sign In

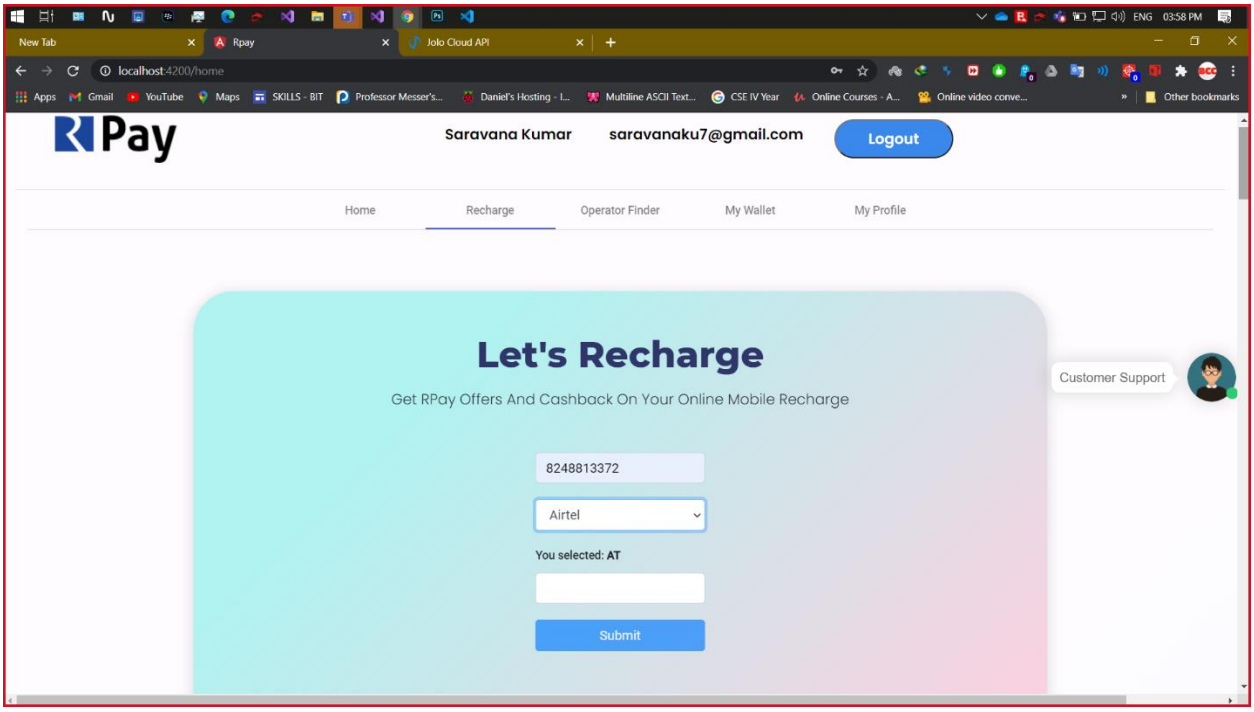


Screenshot 3.2 SignIn with Invalid credentials

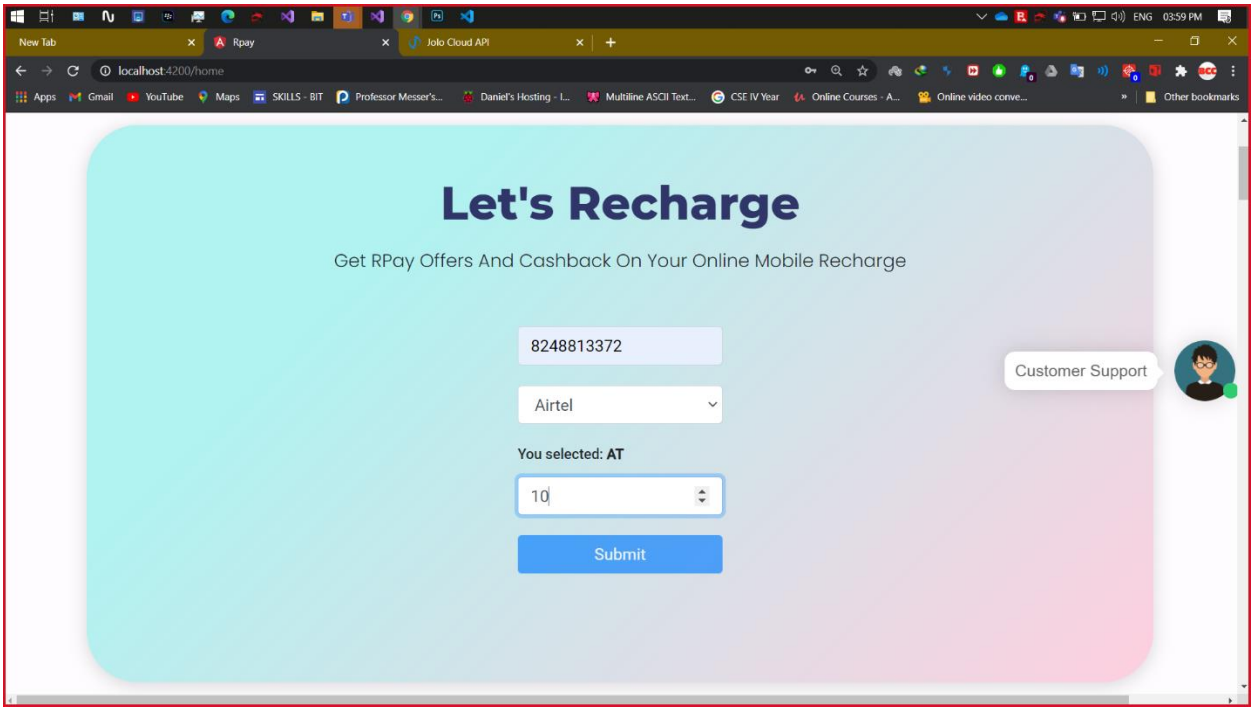


RECHARGE

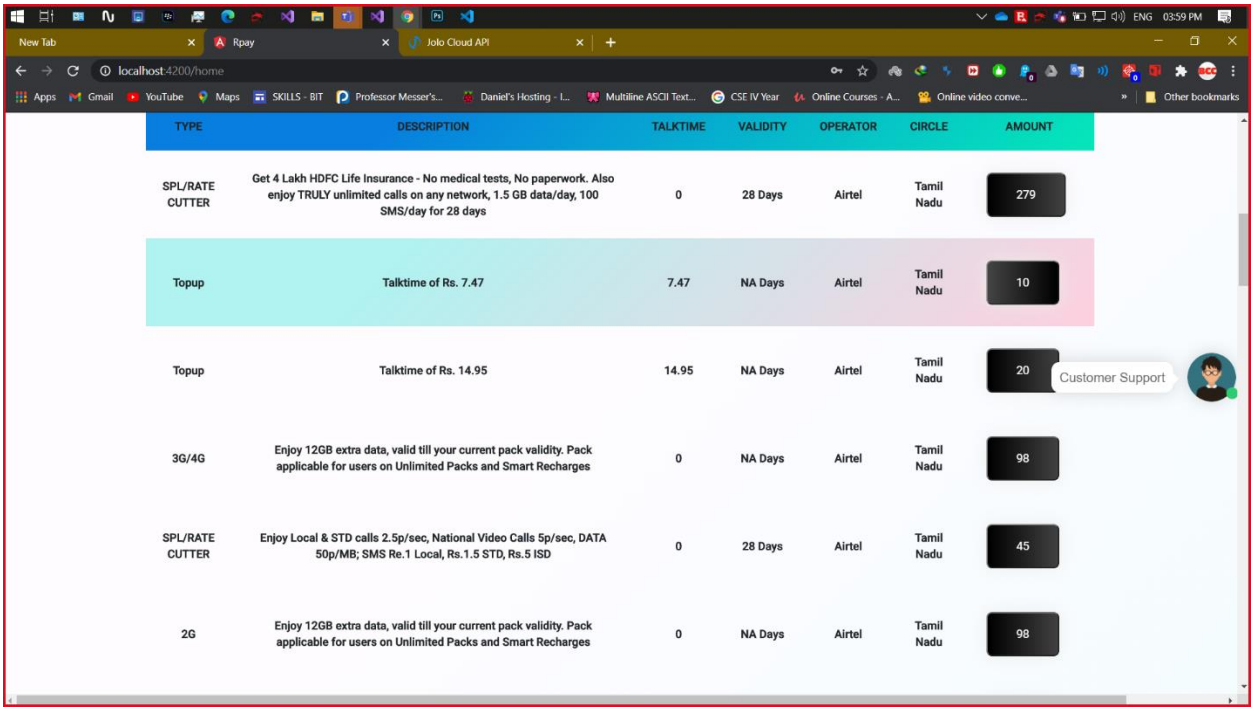
Screenshot 4.1 User Mobile Number & Operator



Screenshot 4.2 Plan amount

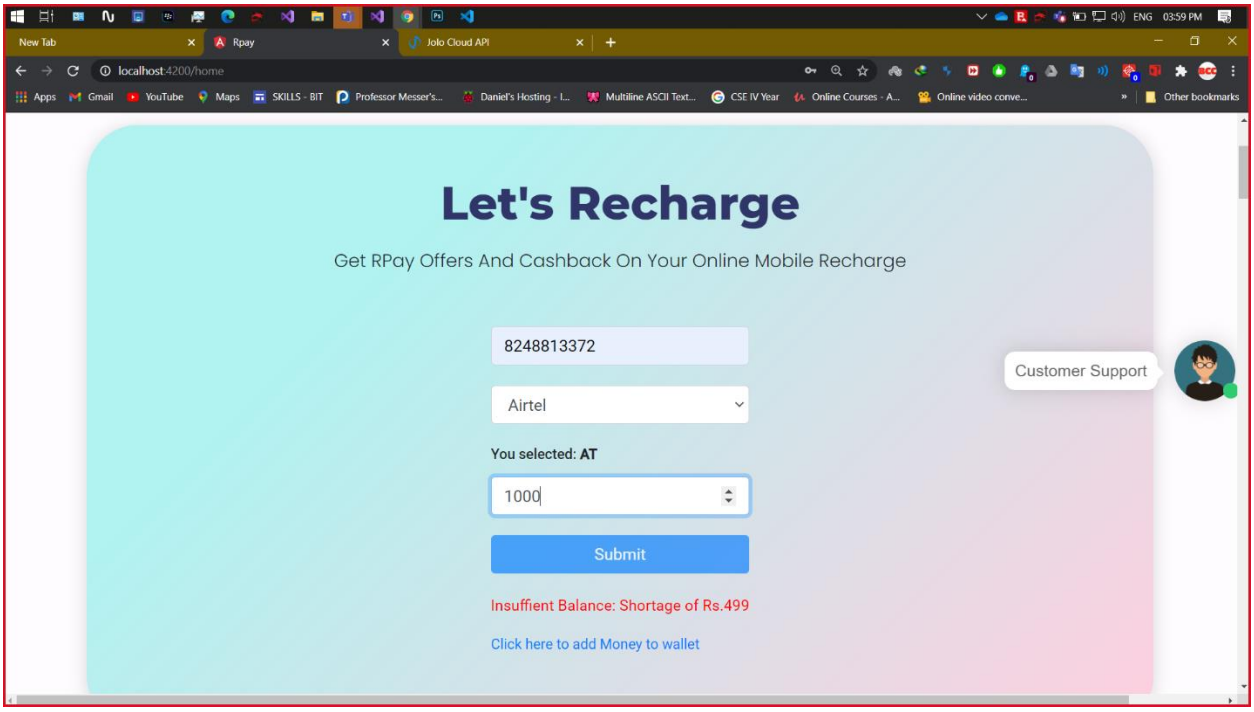


Screenshot 4.3 Tarrif Plans

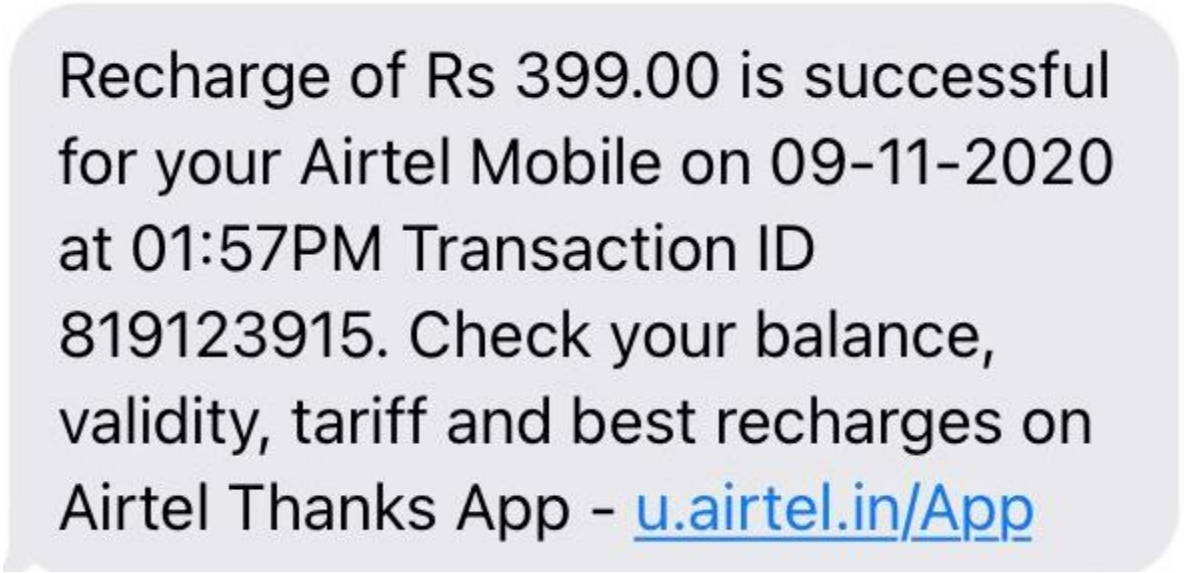


TYPE	DESCRIPTION	TALKTIME	VALIDITY	OPERATOR	CIRCLE	AMOUNT
SPL/RATE CUTTER	Get 4 Lakh HDFC Life Insurance - No medical tests, No paperwork. Also enjoy TRULY unlimited calls on any network, 1.5 GB data/day, 100 SMS/day for 28 days	0	28 Days	Airtel	Tamil Nadu	279
Topup	Talktime of Rs. 7.47	7.47	NA Days	Airtel	Tamil Nadu	10
Topup	Talktime of Rs. 14.95	14.95	NA Days	Airtel	Tamil Nadu	20
3G/4G	Enjoy 12GB extra data, valid till your current pack validity. Pack applicable for users on Unlimited Packs and Smart Recharges	0	NA Days	Airtel	Tamil Nadu	98
SPL/RATE CUTTER	Enjoy Local & STD calls 2.5p/sec, National Video Calls 5p/sec, DATA 50p/MB; SMS Re.1 Local, Rs.1.5 STD, Rs.5 ISD	0	28 Days	Airtel	Tamil Nadu	45
2G	Enjoy 12GB extra data, valid till your current pack validity. Pack applicable for users on Unlimited Packs and Smart Recharges	0	NA Days	Airtel	Tamil Nadu	98

Screenshot 4.4 Recharge Success

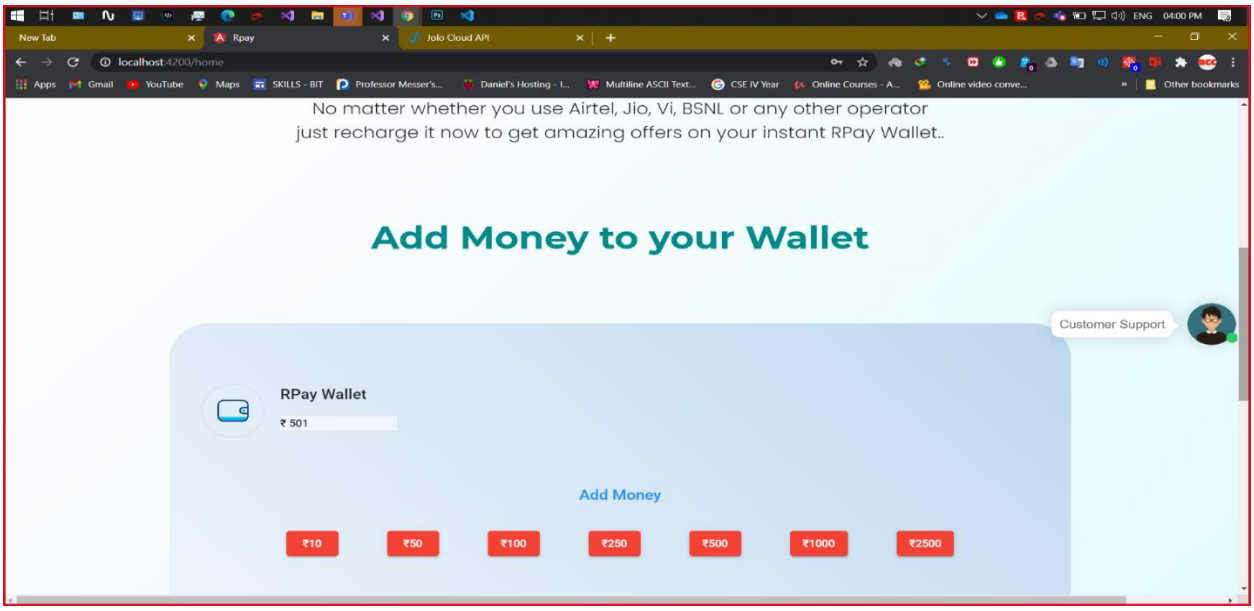


Screenshot 4.5 Recharge Success SMS

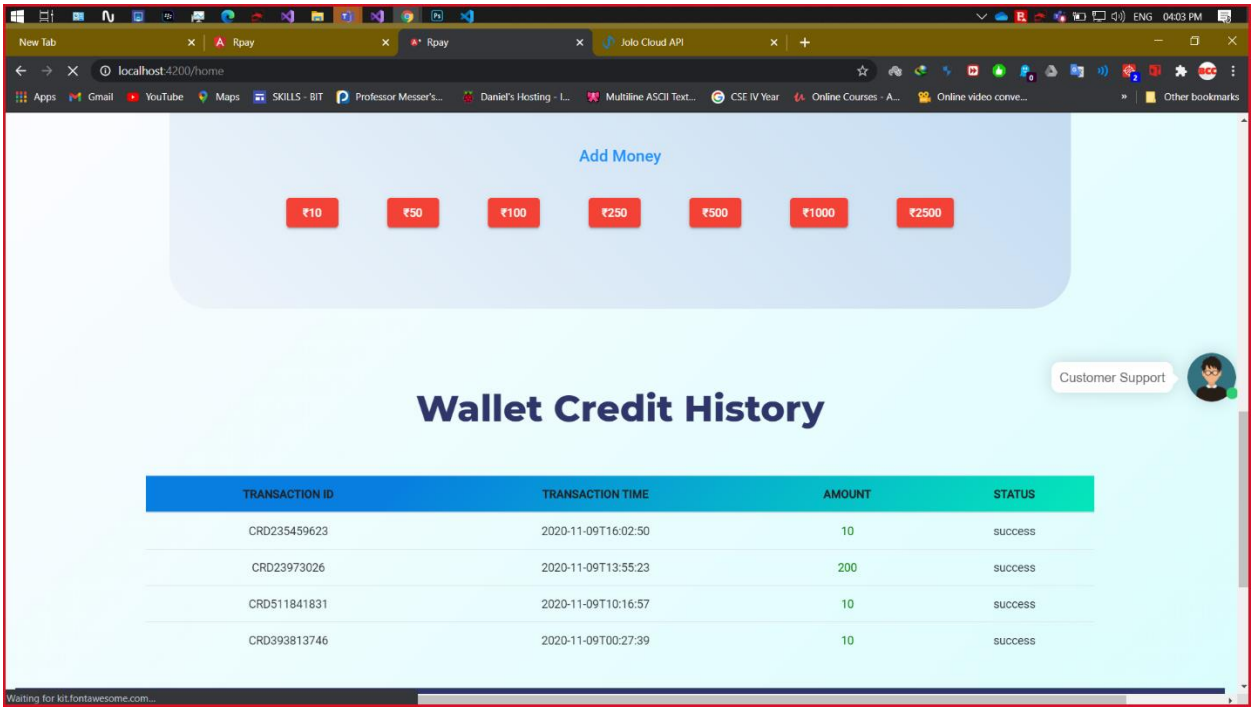


ADD MONEY TO WALLET

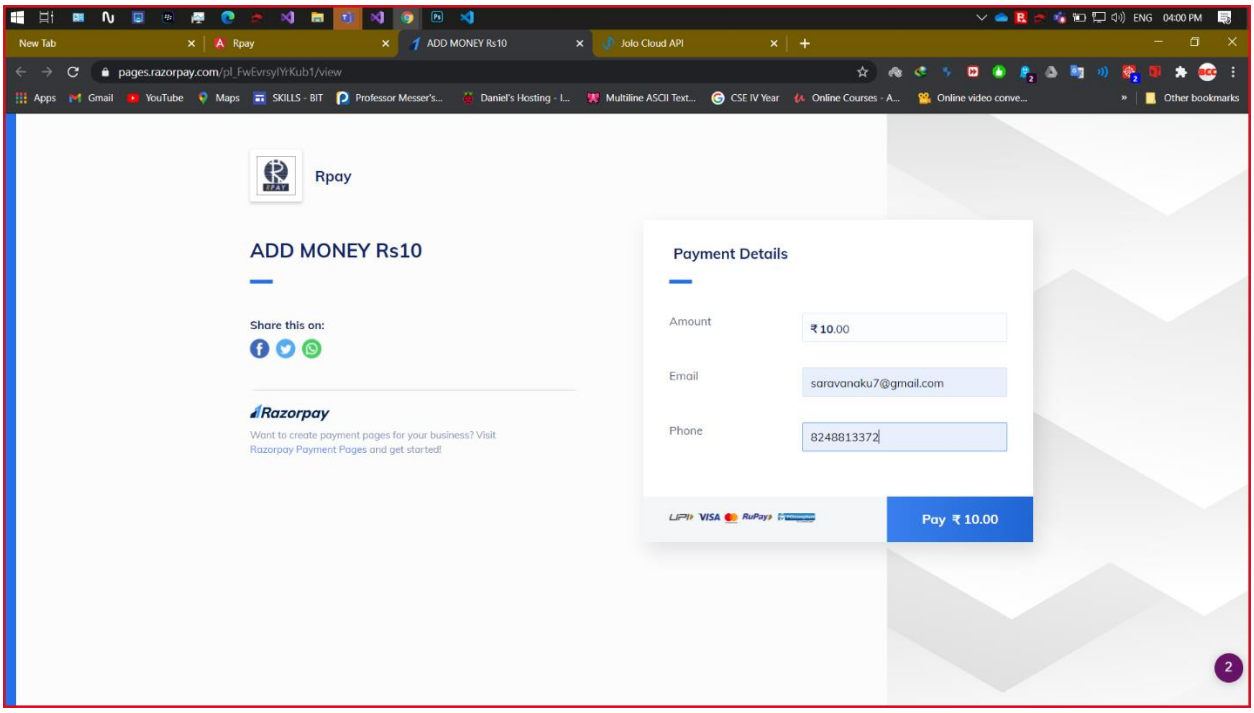
Screenshot 5.1 Current Wallet balance



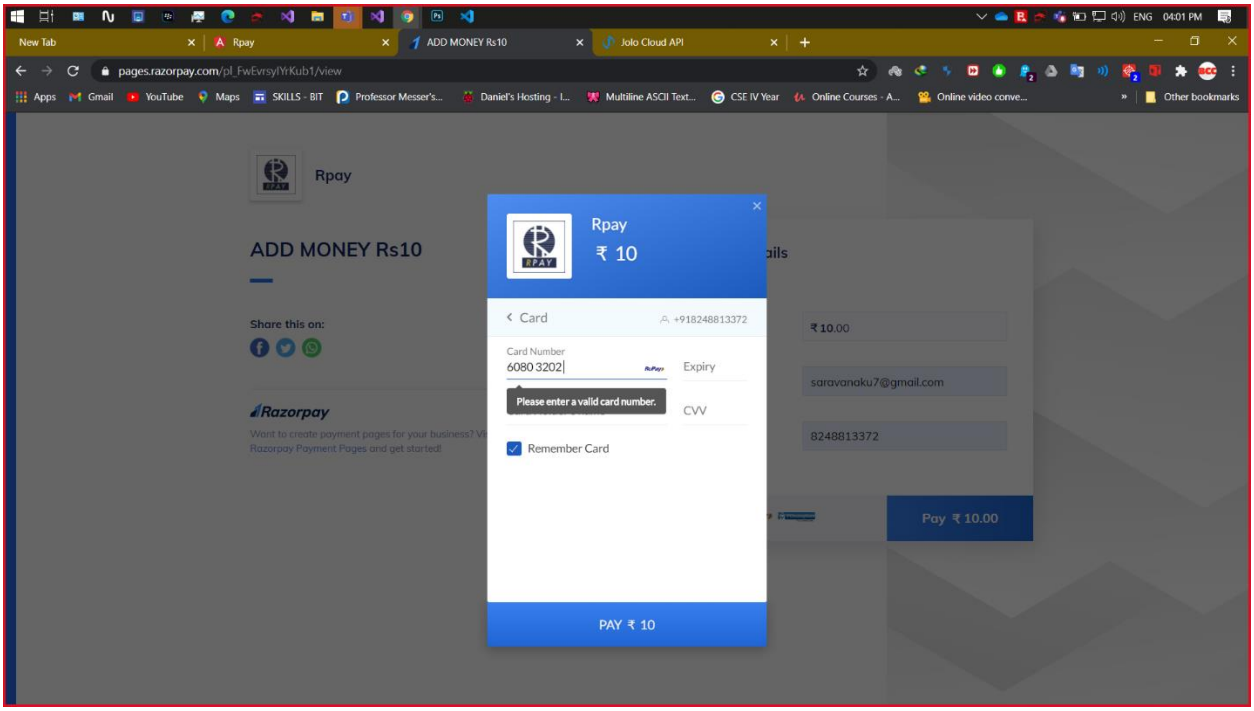
Screenshot 5.2 Wallet History



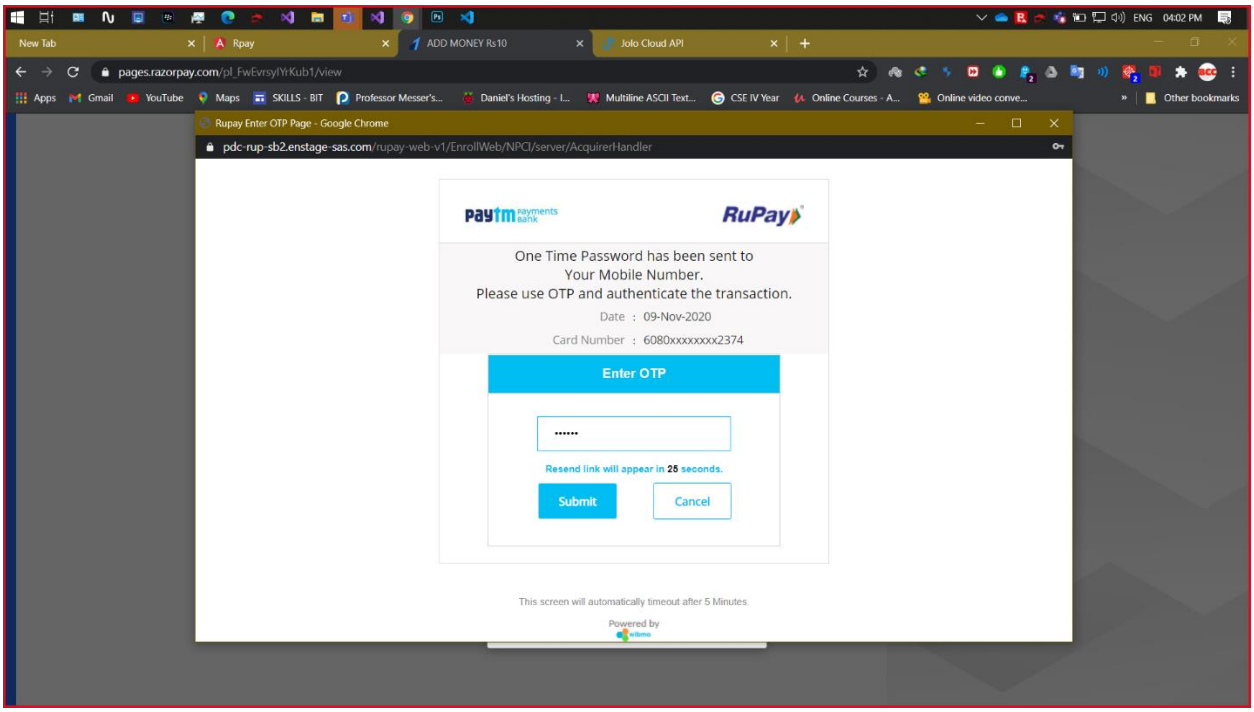
Screenshot 5.3 Razor Pay Payment Gateway



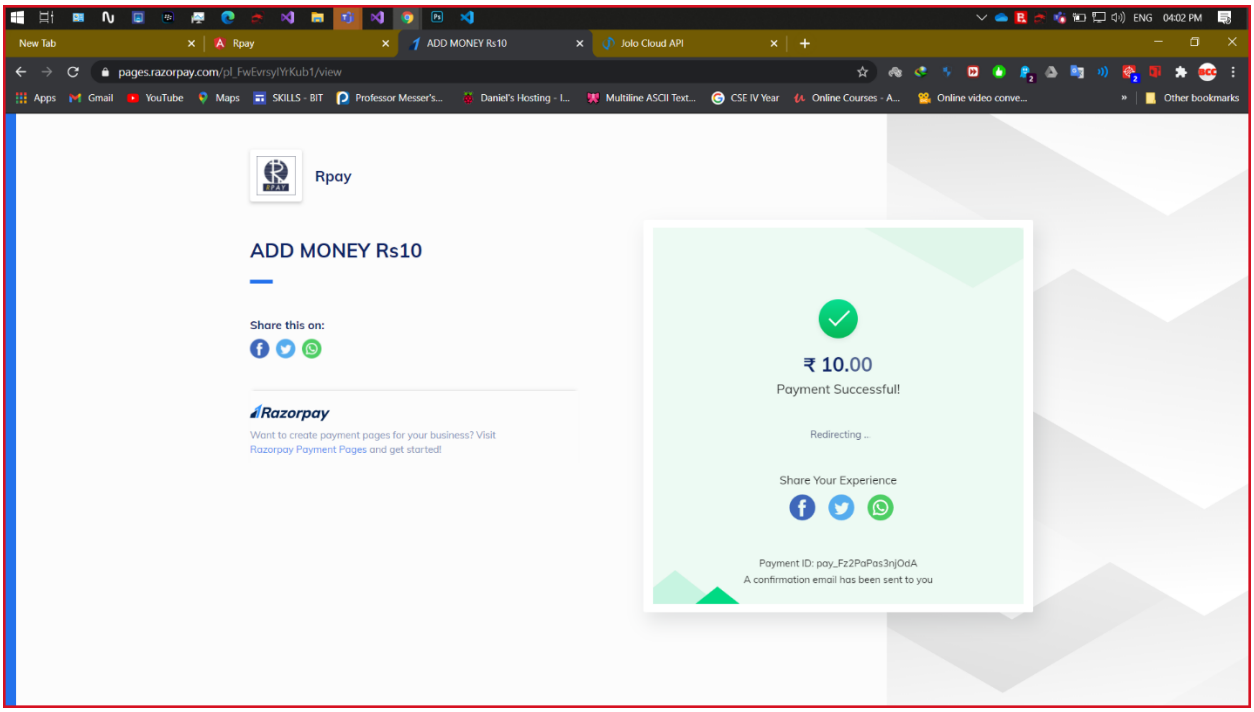
Screenshot 5.4 Add Money – Bank details Verification



Screenshot 5.5 Add Money – OTP Verification

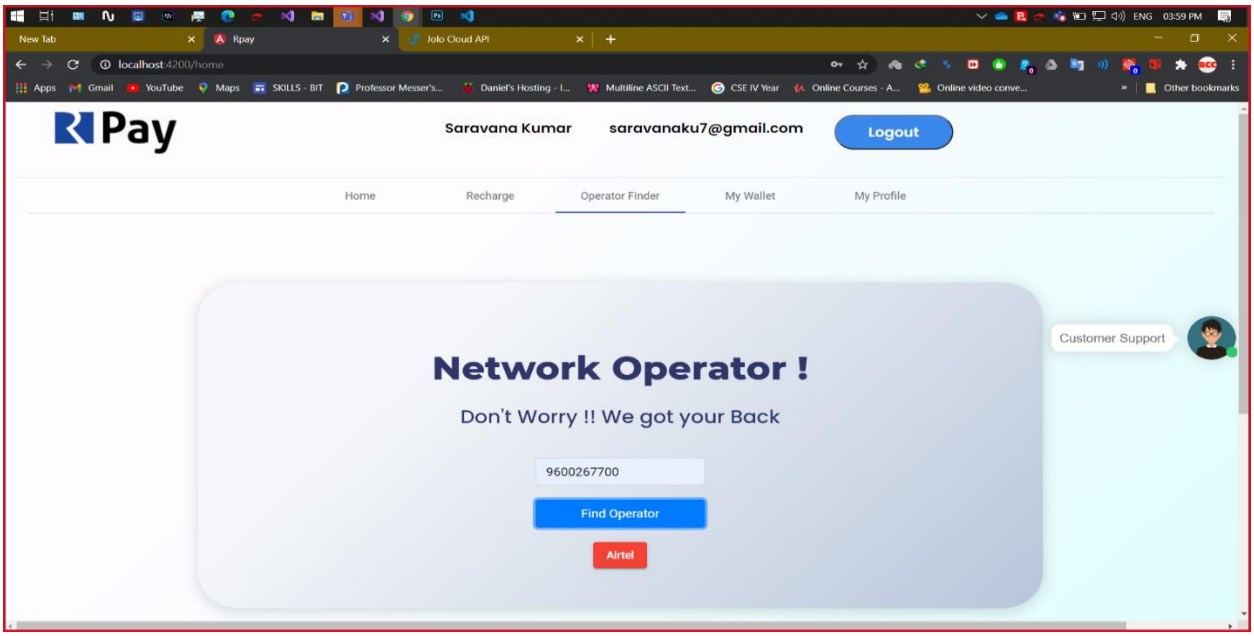


Screenshot 5.6 Receipt for Added Money



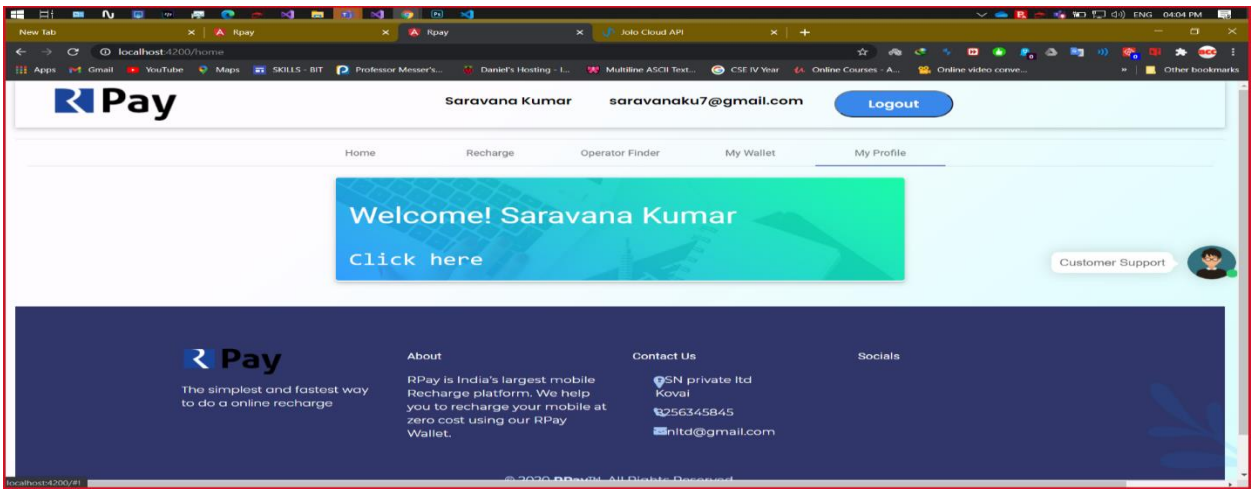
OPERATOR FINDER

Screenshot 6.1 Operator name

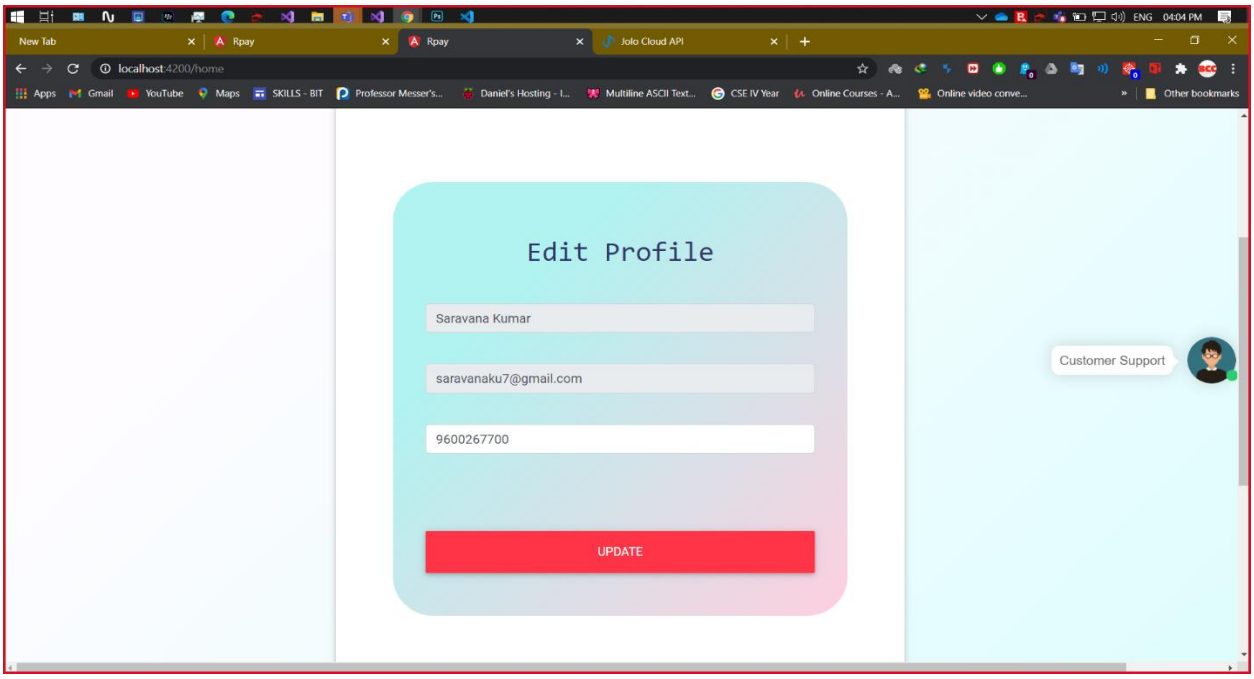


MY PROFILE

Screenshot 7.1 Current Profile details

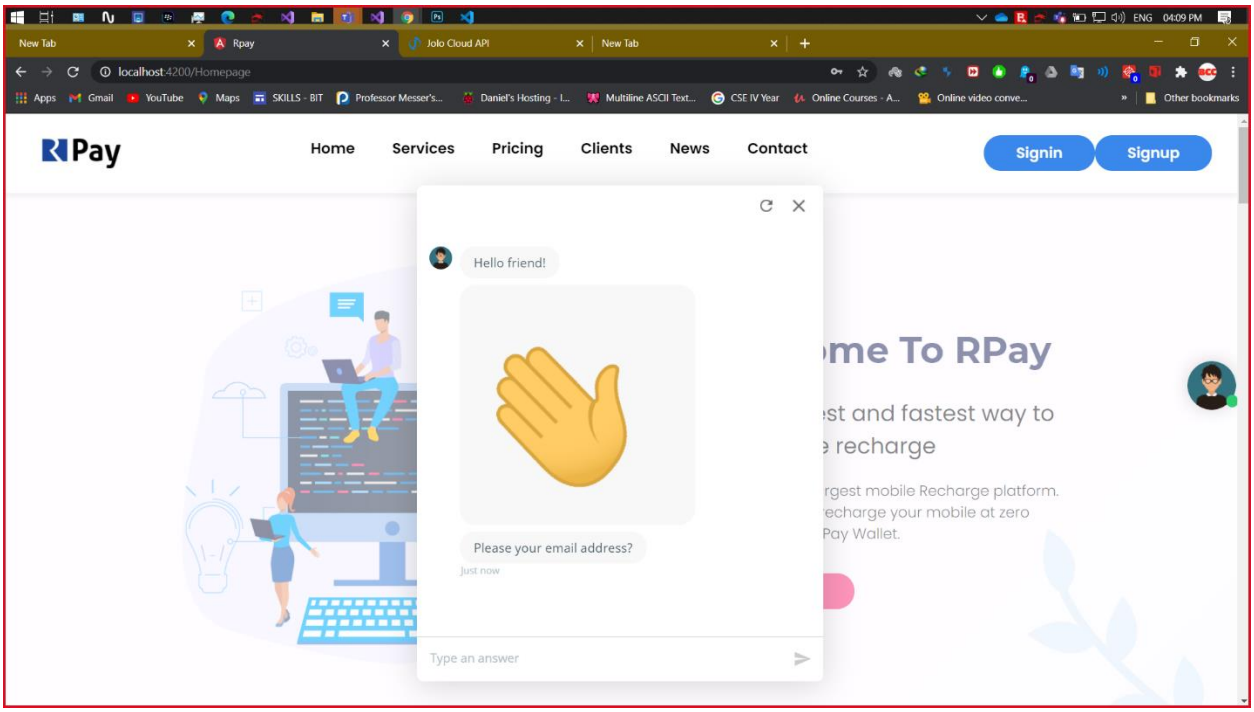


Screenshot 7.2 Update Profile

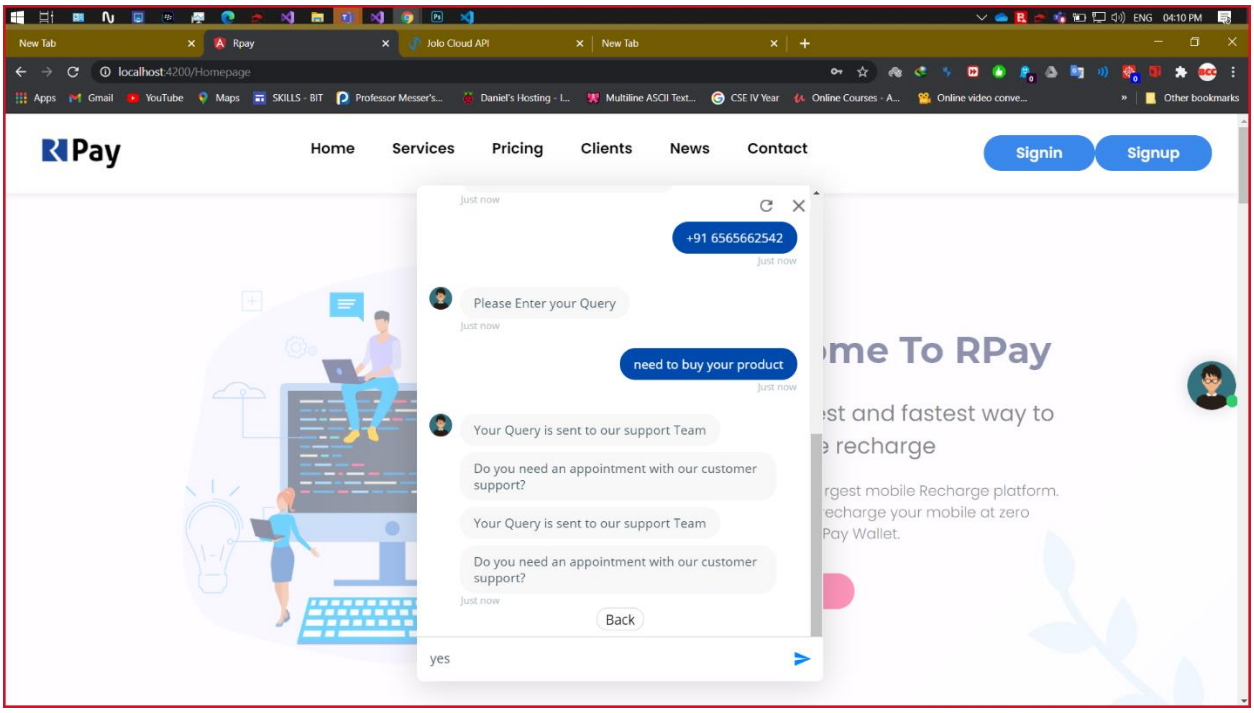


CHAT BOT

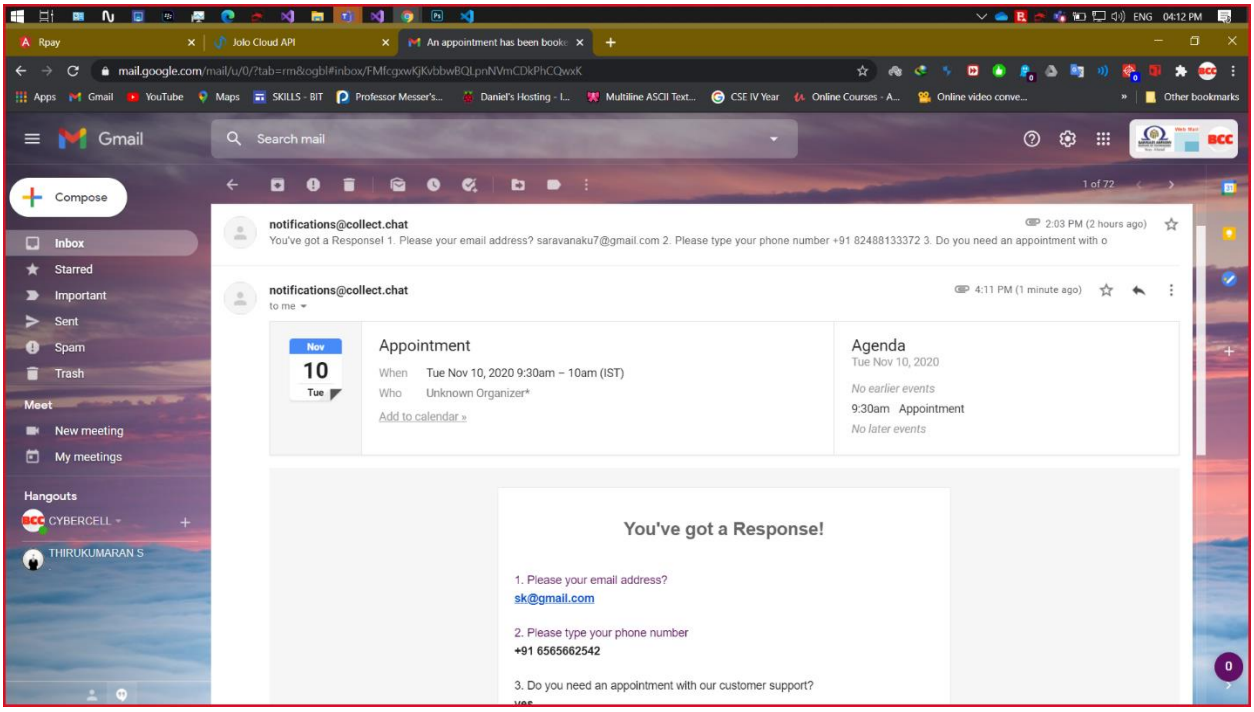
Screenshot 8.1 Chatbot



Screenshot 8.2 Chatbot Query & Response

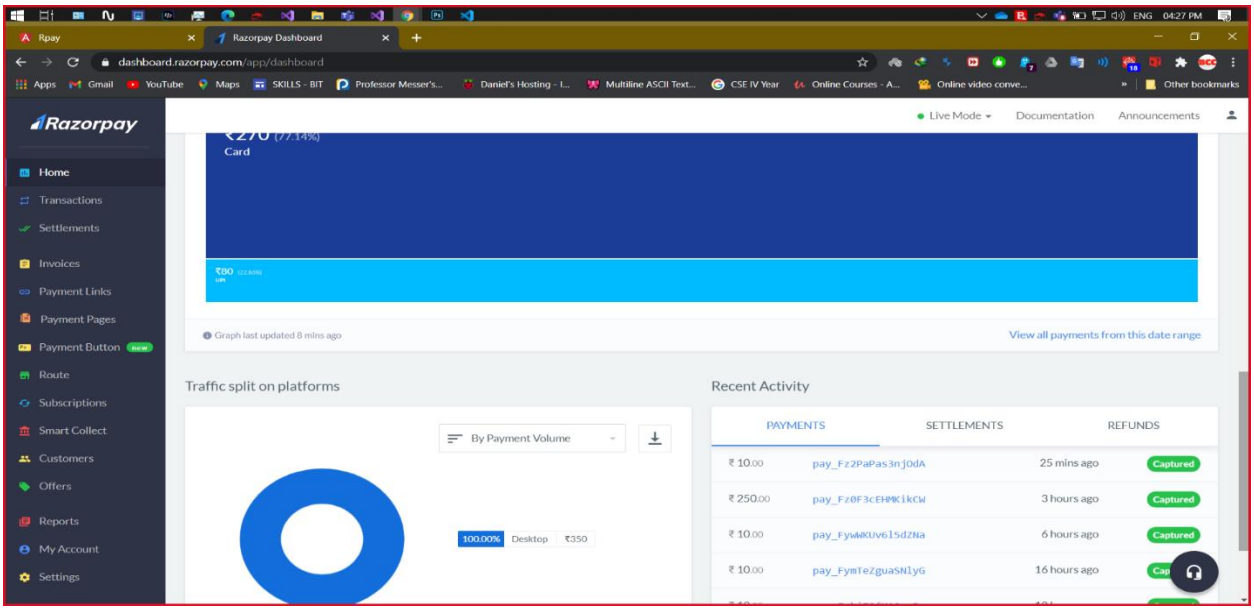


Screenshot 8.3 Chatbot Mail Response

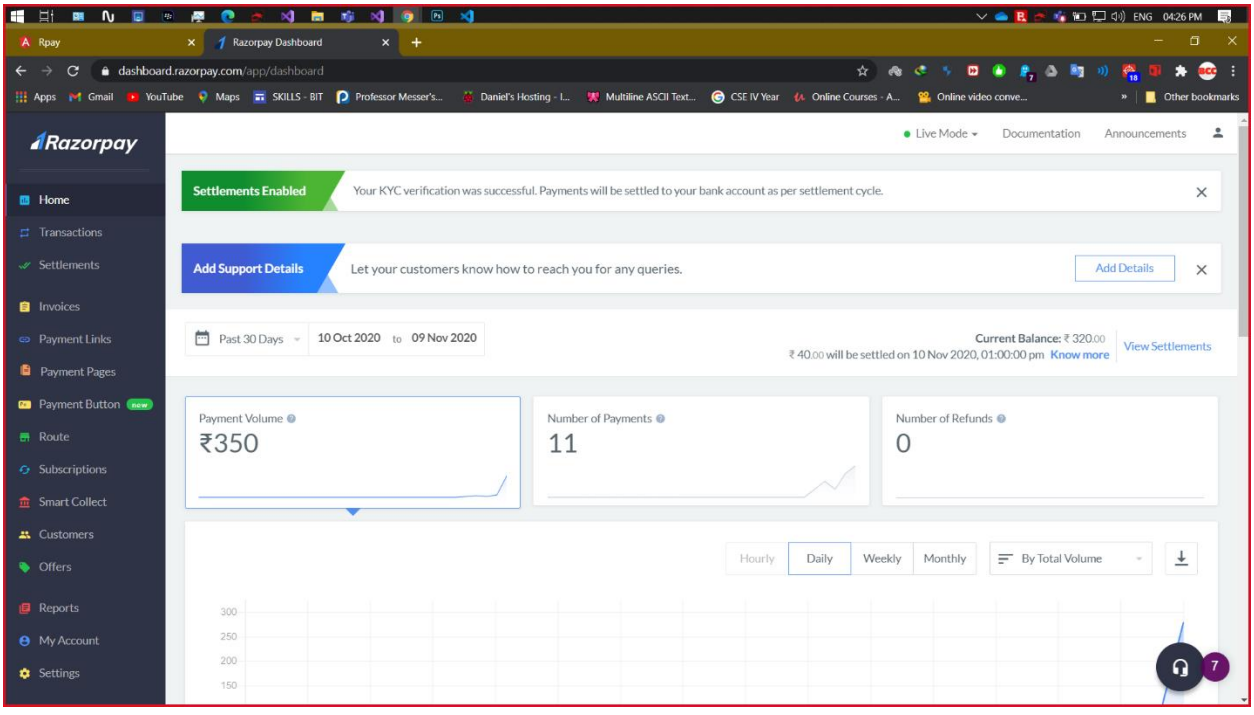


RAZOR PAY DASHBOARD

Screenshot 9.1 Razor pay Home

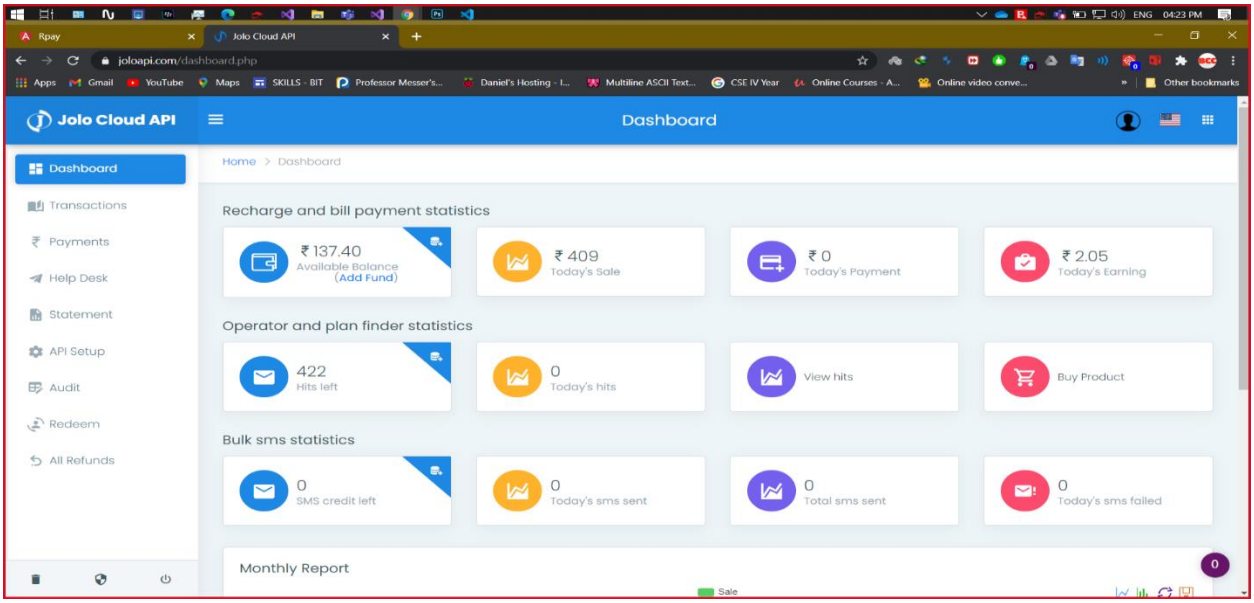


Screenshot 9.2 Payment Monitoring section

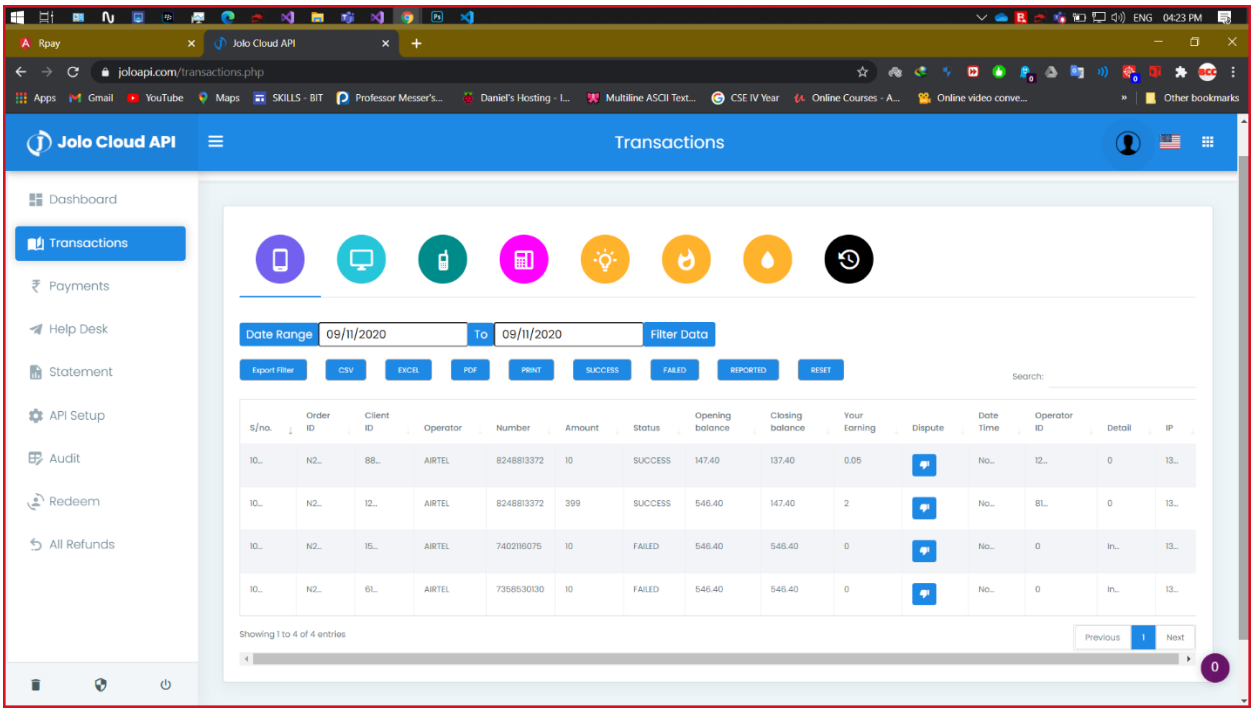


JOLO API DASHBOARD

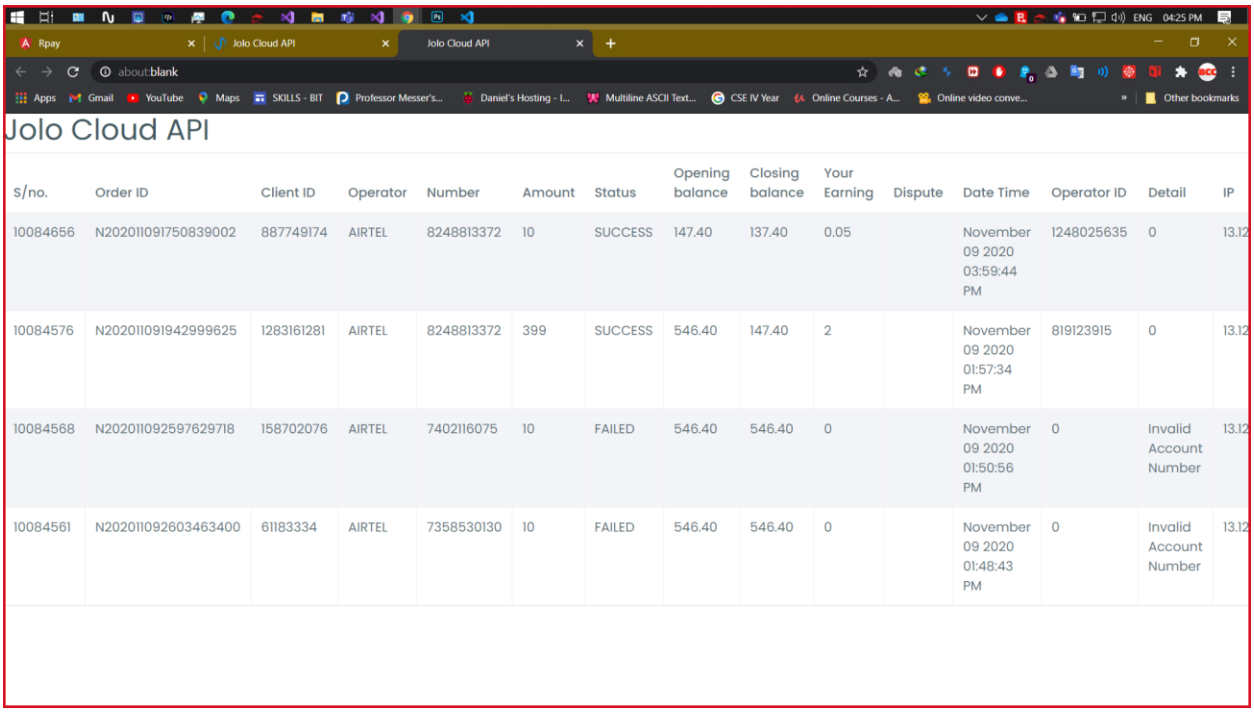
Screenshot 10.1 Jolo API Home



Screenshot 10.2 Transaction Monitoring Section



Screenshot 10.3 Transaction AutoGenerated PDF



Jolo Cloud API

S/no.	Order ID	Client ID	Operator	Number	Amount	Status	Opening balance	Closing balance	Your Earning	Dispute	Date Time	Operator ID	Detail	IP
10084656	N202011091750839002	887749174	AIRTEL	8248813372	10	SUCCESS	147.40	137.40	0.05		November 09 2020 03:59:44 PM	1248025635	0	13.12
10084576	N202011091942999625	1283161281	AIRTEL	8248813372	399	SUCCESS	546.40	147.40	2		November 09 2020 01:57:34 PM	819123915	0	13.12
10084568	N202011092597629718	158702076	AIRTEL	7402116075	10	FAILED	546.40	546.40	0		November 09 2020 01:50:56 PM	0	Invalid Account Number	13.12
10084561	N202011092603463400	61183334	AIRTEL	7358530130	10	FAILED	546.40	546.40	0		November 09 2020 01:48:43 PM	0	Invalid Account Number	13.12

ROLES AND RESPONSIBILITIES

SCREENSHOT WISE:

Screen shot No	Page Name	Frontend	Backend
1	Home Page	Naveeth	Rajagurunathan
2	Sign Up	Saravanakumar	Rajagurunathan
3	Sign In	Rajagurunathan	Saravanakumar
4	Recharge	Naveeth	Saravanakumar, Rajagurunathan
5	Add Money to wallet	Naveeth	Saravanakumar, Rajagurunathan
6	Operator Finder	Naveeth	Rajagurunathan, Saravanakumar
7	My Profile	Naveeth	Naveeth
8	Chat Bot Integration	Naveeth, Rajagurunathan, Saravanakumar	
9	Razor Pay Integration	Naveeth, Rajagurunathan, Saravanakumar	
10	Jolo API Integration	Naveeth, Rajagurunathan, Saravanakumar	
11	Database Management	Rajagurunathan	
12	Database Hosting	Saravanakumar	
13	Web Api Hosting	Saravanakumar	

MODULE WISE:

1) Member Name: Naveeth Z A

- Home Page - Front End
- Recharge - Front End
- Operator Finder - Front End
- My Profile - Front End & Back End
- Add Money to wallet - Front End
- Razor Pay Integration
- Chat Bot Integration
- Jolo API Integration
- Dry Run
- Integration

2) Member Name: Rajagurunathan Manikandan (Team Leader)

- Home Page –Back End
- Recharge - Back End
- Operator Finder – Back End
- Sign In – Front End
- Sign Up – Back End
- Add Money to wallet - Back End
- Razor Pay Integration
- Chat Bot Integration
- Jolo API Integration
- Database and Query Management
- Integration

3) Member Name: Saravanakumar Balasubramaniyam

- Recharge - Back End
- Operator Finder – Back End
- Sign In – Back End
- Sign Up – Front End
- Add Money to wallet - Back End
- Razor Pay Integration
- Chat Bot Integration
- Jolo API Integration
- Integration
- State Management using JWT
- Online Database Hosting
- Web Hosting