==========================Infosys.QuickKart.BusinessLayer================================

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Infosys.QuickKart.DataAccessLayer;

using Infosys.QuickKart.Common.Models;

namespace Infosys.QuickKart.BusinessLayer

{

public class CustomerBL

{

public string ValidateUserCredentials(string emailId, string password)

{

string roleName = "";

try

{

var dal = new QuickKartRepository();

roleName = dal.ValidateLoginUsingLinq(emailId, password);

}

catch (Exception)

{

roleName = "Invalid credentials";

}

return roleName;

}

public bool RegisterUserUsingAPI(Common.Models.User user)

{

bool status;

try

{

DataAccessLayer.User userObj = new DataAccessLayer.User();

userObj.EmailId = user.EmailId;

userObj.UserPassword = user.UserPassword;

userObj.Gender = user.Gender;

userObj.DateOfBirth = user.DateOfBirth;

var dal = new QuickKartRepository();

status = dal.RegisterUserUsingLinq(userObj);

}

catch (Exception)

{

status = false;

}

return status;

}

public List<Common.Models.Product> GetProducts()

{

try

{

var dal = new QuickKartRepository();

var productList = dal.DisplayProductDetails();

var products = new List<Common.Models.Product>();

if (productList.Any())

{

foreach (var prod in productList)

{

var product = new Common.Models.Product();

product.ProductId = prod.ProductId;

product.ProductName = prod.ProductName;

product.CategoryId = prod.CategoryId;

product.Price = prod.Price;

product.QuantityAvailable = prod.QuantityAvailable;

products.Add(product);

}

}

return products;

}

catch (Exception)

{

return null;

}

}

public List<Common.Models.Category> GetCategories()

{

try

{

var dal = new QuickKartRepository();

var categoryList = dal.GetCategoriesUsingLinq();

var categories = new List<Common.Models.Category>();

if (categoryList.Any())

{

foreach (var cat in categoryList)

{

var category = new Common.Models.Category();

category.CategoryId = cat.CategoryId;

category.CategoryName = cat.CategoryName;

categories.Add(category);

}

}

return categories;

}

catch (Exception ex)

{

return null;

}

}

public int AddProductToCart(Common.Models.Cart cartObj)

{

int returnvalue = -1;

try

{

var dal = new QuickKartRepository();

returnvalue = dal.AddProductToCartUsingUSP(cartObj.ProductId, cartObj.EmailId);

}

catch (Exception)

{

returnvalue = -1;

}

return returnvalue;

}

public List<Common.Models.CartProductsDetails> GetCartProducts(string emailId)

{

try

{

var dal = new QuickKartRepository();

var cartProductList = dal.DisplayCartProductDetails(emailId);

Common.Models.CartProductsDetails product;

var cartProducts = new List<Common.Models.CartProductsDetails>();

if (cartProductList.Any())

{

foreach (var prod in cartProductList)

{

product = new Common.Models.CartProductsDetails();

product.ProductId = prod.ProductId;

product.ProductName = prod.ProductName;

product.Quantity = prod.Quantity;

product.QuantityAvailable = prod.QuantityAvailable;

product.Price = prod.Price;

cartProducts.Add(product);

}

}

return cartProducts;

}

catch (Exception)

{

return null;

}

}

public bool UpdateCartProducts(Common.Models.Cart cartObj)

{

bool status = false;

try

{

var dal = new QuickKartRepository();

status = dal.UpdateCartProductsLinq(cartObj.ProductId, cartObj.EmailId, cartObj.Quantity);

}

catch (Exception)

{

status = false;

}

return status;

}

public bool DeleteCartProduct(Common.Models.Cart cartObj)

{

var status = false;

try

{

var dal = new QuickKartRepository();

status = dal.DeleteCartProduct(cartObj.ProductId, cartObj.EmailId);

}

catch (Exception ex)

{

status = false;

}

return status;

}

public List<Common.Models.PurchaseDetail> GetPurchaseDetailsByEmailId(string emailId)

{

try

{

var dal = new QuickKartRepository();

var purchaseList = dal.DisplayPurchaseDetailsByCustomer(emailId);

var purchases = new List<Common.Models.PurchaseDetail>();

if (purchaseList.Any())

{

foreach (var purchase in purchaseList)

{

var purchaseObj = new Common.Models.PurchaseDetail();

purchaseObj.PurchaseId = purchase.PurchaseId;

purchaseObj.EmailId = purchase.EmailId;

purchaseObj.ProductId = purchase.ProductId;

purchaseObj.ProductName = purchase.Product.ProductName;

purchaseObj.QuantityPurchased = purchase.QuantityPurchased;

purchaseObj.PurchaseDate = purchase.DateOfPurchase.ToShortDateString();

purchases.Add(purchaseObj);

}

}

return purchases;

}

catch (Exception ex)

{

return null;

}

}

public List<Common.Models.Rating> DisplayAllReviewDetailsByCustomer(string emailId)

{

try

{

var dal = new QuickKartRepository();

var ratingList = dal.DisplayAllReviewDetailsByCustomer(emailId);

Common.Models.Rating rating;

var ratings = new List<Common.Models.Rating>();

if (ratingList.Any())

{

foreach (var rate in ratingList)

{

rating = new Common.Models.Rating();

rating.EmailId = rate.EmailId;

rating.ProductId = rate.ProductId;

rating.ProductName = rate.Product.ProductName;

rating.ReviewComments = rate.ReviewComments;

rating.ReviewRating = rate.ReviewRating;

ratings.Add(rating);

}

}

return ratings;

}

catch (Exception ex)

{

return null;

}

}

public bool InsertRating(Common.Models.Rating rating)

{

bool status = false;

try

{

var dal = new QuickKartRepository();

status = dal.AddRatings(rating);

}

catch (Exception)

{

status = false;

}

return status;

}

//public Common.Models.Rating GetProductReviewByCustomer(string emailId, string productId)

//{

// var rating = new Common.Models.Rating();

// try

// {

// var dal = new QuickKartRepository();

// var rate = dal.GetProductReviewByCustomer(emailId, productId);

// rating.EmailId = rate.EmailId;

// rating.ProductId = rate.ProductId;

// rating.ReviewRating = rate.ReviewRating;

// rating.ReviewComments = rate.ReviewComments;

// }

// catch (Exception ex)

// {

// rating = null;

// }

// return rating;

//}

public bool UpdateReviewComments(Common.Models.Rating rating)

{

bool status = false;

try

{

var dal = new QuickKartRepository();

status = dal.UpdateReviewComments(rating.EmailId, rating.ProductId, rating.ReviewComments);

}

catch (Exception)

{

status = false;

}

return status;

}

public bool DeleteRating(Common.Models.Rating rating)

{

var status = false;

try

{

var dal = new QuickKartRepository();

status = dal.DeleteRating(rating.EmailId, rating.ProductId);

}

catch (Exception ex)

{

status = false;

}

return status;

}

}

}

==========================Infosys.QuickKart.Common.Models===============================

namespace Infosys.QuickKart.Common.Models

{

public class Cart

{

public string ProductId { get; set; }

public string EmailId { get; set; }

public short Quantity { get; set; }

}

public class CartProductsDetails

{

public string ProductId { get; set; }

public string ProductName { get; set; }

public decimal Price { get; set; }

public short Quantity { get; set; }

public int QuantityAvailable { get; set; }

}

public class Category

{

public byte CategoryId { get; set; }

public string CategoryName { get; set; }

}

public class Product

{

public string ProductId { get; set; }

public string ProductName { get; set; }

public Nullable<byte> CategoryId { get; set; }

public decimal Price { get; set; }

public int QuantityAvailable { get; set; }

}

public class PurchaseDetail

{

public long PurchaseId { get; set; }

public string ProductName { get; set; }

public string EmailId { get; set; }

public string ProductId { get; set; }

public short QuantityPurchased { get; set; }

public String PurchaseDate { get; set; }

}

public class Rating

{

public string EmailId { get; set; }

public string ProductId { get; set; }

public string ProductName { get; set; }

public byte ReviewRating { get; set; }

public string ReviewComments { get; set; }

}

public class Role

{

public byte RoleId { get; set; }

public string RoleName { get; set; }

}

public class User

{

public string EmailId { get; set; }

public string UserPassword { get; set; }

public Nullable<byte> RoleId { get; set; }

public string Gender { get; set; }

public System.DateTime DateOfBirth { get; set; }

public string Address { get; set; }

public virtual Common.Models.Role Role { get; set; }

}

}

================ Infosys.QuickKart.DataAccessLayer==================================

using System;

using System.Collections.Generic;

using System.Data.Entity.Core.Objects;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Infosys.QuickKart.Common.Models;

namespace Infosys.QuickKart.DataAccessLayer

{

public class QuickKartRepository

{

private QuickKartDBContext Context { get; set; }

public QuickKartRepository()

{

Context = new QuickKartDBContext();

}

//Validate user using LINQ

public string ValidateLoginUsingLinq(string emailId, string password)

{

string roleName = "";

try

{

var objUser = (from usr in Context.Users

where usr.EmailId == emailId && usr.UserPassword == password

select usr).FirstOrDefault<User>();

if (objUser != null)

{

roleName = objUser.Role.RoleName;

}

else

{

roleName = "Invalid credentials";

}

}

catch (Exception)

{

roleName = "Invalid credentials";

}

return roleName;

}

//Register customer using LINQ

public bool RegisterUserUsingLinq(User user)

{

bool status;

try

{

var role = (from rol in Context.Roles where rol.RoleName == "Customer" select rol).FirstOrDefault<Role>();

if (role != null)

{

user.Role = role;

}

else

{

status = false;

}

Context.Users.Add(user);

Context.SaveChanges();

status = true;

}

catch (Exception)

{

status = false;

}

return status;

}

//Get all categories using LINQ

public List<Category> GetCategoriesUsingLinq()

{

List<Category> lstCategories = null;

try

{

lstCategories = (from c in Context.Categories

orderby c.CategoryId

ascending

select c).ToList<Category>();

}

catch (Exception)

{

lstCategories = null;

}

return lstCategories;

}

//Display all the products

public List<Product> DisplayProductDetails()

{

List<Product> lstProducts = null;

try

{

lstProducts = (from c in Context.Products

orderby c.CategoryId

ascending

select c).ToList<Product>();

}

catch (Exception ex)

{

lstProducts = null;

}

return lstProducts;

}

//Display purchases of customer

public List<PurchaseDetail> DisplayPurchaseDetailsByCustomer(string emailId)

{

List<PurchaseDetail> lstPurchaseDetails = null;

try

{

lstPurchaseDetails = Context.PurchaseDetails.Where(x => x.EmailId == emailId).OrderByDescending(x => x.DateOfPurchase).ToList<PurchaseDetail>();

}

catch (Exception ex)

{

lstPurchaseDetails = null;

}

return lstPurchaseDetails;

}

//Display product based upon product name

public List<Product> DisplayProductDetailsByProductName(string subStr)

{

List<Product> lstProducts = null;

try

{

lstProducts = (from c in Context.Products

where c.ProductName.ToLower().Contains(subStr.ToLower())

orderby c.CategoryId ascending

select c).ToList<Product>();

}

catch (Exception ex)

{

lstProducts = null;

}

return lstProducts;

}

//

public int AddProductToCartUsingUSP(string productId, string emailId)

{

System.Nullable<int> returnvalue = -1;

try

{

returnvalue = Context.usp\_AddProductToCart(productId,emailId).SingleOrDefault();

}

catch (Exception)

{

returnvalue = -1;

}

return Convert.ToInt32(returnvalue);

}

public List<ufn\_FetchCartProductByEmailId\_Result> DisplayCartProductDetails(string emailId)

{

List<ufn\_FetchCartProductByEmailId\_Result> lstProducts = null;

try

{

lstProducts = Context.ufn\_FetchCartProductByEmailId(emailId).ToList<ufn\_FetchCartProductByEmailId\_Result>();

}

catch (Exception)

{

lstProducts = null;

}

return lstProducts;

}

public bool UpdateCartProductsLinq(string productId,string emailId, short quantity)

{

bool status = false;

Cart cartproduct = null;

try

{

cartproduct = (from cartProd in Context.Carts where cartProd.ProductId == productId && cartProd.EmailId==emailId select cartProd).FirstOrDefault<Cart>();

if (cartproduct != null)

{

cartproduct.Quantity = quantity;

Context.SaveChanges();

status = true;

}

else

{

status = false;

}

}

catch (Exception ex)

{

status = false;

}

return status;

}

public bool DeleteCartProduct(string productId, string emailId)

{

bool status = false;

try

{

var product = (from cart in Context.Carts

where cart.ProductId == productId && cart.EmailId==emailId

select cart).FirstOrDefault<Cart>();

Context.Carts.Remove(product);

Context.SaveChanges();

status = true;

}

catch (Exception ex)

{

status = false;

}

return status;

}

public List<Rating> DisplayAllReviewDetailsByCustomer(string emailId)

{

List<Rating> lstReviewDetails = null;

try

{

lstReviewDetails = Context.Ratings.Where(x => x.EmailId == emailId).ToList<Rating>();

}

catch (Exception ex)

{

lstReviewDetails = null;

}

return lstReviewDetails;

}

public bool AddRatings(Common.Models.Rating rating)

{

bool status = false;

try

{

Rating ratingObj = new Rating();

ratingObj.EmailId = rating.EmailId;

ratingObj.ProductId = rating.ProductId;

ratingObj.ReviewComments = rating.ReviewComments;

ratingObj.ReviewRating = rating.ReviewRating;

Context.Ratings.Add(ratingObj);

Context.SaveChanges();

status = true;

}

catch (Exception ex)

{

status = false;

}

return status;

}

//public Rating GetProductReviewByCustomer(string emailId, string productId)

//{

// Rating rating;

// try

// {

// rating = Context.Ratings.Where(r => r.EmailId == emailId && r.ProductId == productId).Select(r => r).SingleOrDefault();

// }

// catch (Exception ex)

// {

// rating = null;

// }

// return rating;

//}

public bool UpdateReviewComments(string emailId, string productId, string newComment)

{

bool status = false;

try

{

Rating rating = Context.Ratings.Where(r => (r.EmailId == emailId && r.ProductId == productId)).Select(r => r).FirstOrDefault<Rating>();

rating.ReviewComments = newComment;

Context.SaveChanges();

status = true;

}

catch (Exception ex)

{

status = false;

}

return status;

}

public bool DeleteRating(string emailId, string productId)

{

bool status = false;

try

{

Rating rating = Context.Ratings.Where(r => (r.EmailId == emailId && r.ProductId == productId)).Select(r => r).FirstOrDefault<Rating>();

Context.Ratings.Remove(rating);

Context.SaveChanges();

status = true;

}

catch (Exception ex)

{

status = false;

}

return status;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using System.Web.Http.Results;

using Infosys.QuickKart.DataAccessLayer;

using Infosys.QuickKart.Services.Repository;

using Infosys.QuickKart.BusinessLayer;

using Infosys.QuickKart.Common.Models;

namespace Infosys.QuickKart.Services.Controllers

{

public class CategoryController : ApiController

{

[HttpGet]

public JsonResult<List<Common.Models.Category>> GetCategories()

{

try

{

CustomerBL customerBLObj = new CustomerBL();

var categoryList = customerBLObj.GetCategories();

return Json<List<Common.Models.Category>>(categoryList);

}

catch (Exception ex)

{

return null;

}

}

}

public class ProductController : ApiController

{

[HttpGet]

public JsonResult<List<Common.Models.Product>> GetProducts()

{

try

{

CustomerBL customerBLObj = new CustomerBL();

var productList = customerBLObj.GetProducts();

return Json<List<Common.Models.Product>>(productList);

}

catch (Exception ex)

{

return null;

}

}

}

public class PurchaseController : ApiController

{

[HttpGet]

public JsonResult<List<Common.Models.PurchaseDetail>> GetPurchaseDetailsByEmailId(string emailId)

{

try

{

CustomerBL customerBLObj = new CustomerBL();

var purchaseList = customerBLObj.GetPurchaseDetailsByEmailId(emailId);

return Json<List<Common.Models.PurchaseDetail>>(purchaseList);

}

catch (Exception ex)

{

return null;

}

}

}

………….

public class RatingController : ApiController

{

[HttpGet]

public JsonResult<List<Common.Models.Rating>> DisplayAllReviewDetailsByEmailId(string emailId)

{

try

{

CustomerBL customerBLObj = new CustomerBL();

var reviews = customerBLObj.DisplayAllReviewDetailsByCustomer(emailId);

return Json<List<Common.Models.Rating>>(reviews);

}

catch (Exception)

{

return null;

}

}

[HttpPost]

public bool InsertRating(Common.Models.Rating rating)

{

bool status = false;

try

{

CustomerBL customerBLObj = new CustomerBL();

status = customerBLObj.InsertRating(rating);

}

catch (Exception)

{

status = false;

}

return status;

}

[HttpGet]

//public Common.Models.Rating GetProductReviewByCustomer(string emailId, string productId)

//{

// var rating = new Common.Models.Rating();

// try

// {

// CustomerBL customerBLObj = new CustomerBL();

// rating = customerBLObj.GetProductReviewByCustomer(emailId, productId);

// }

// catch (Exception ex)

// {

// rating = null;

// }

// return rating;

//}

[HttpPut]

public bool UpdateReviewComments(Common.Models.Rating rating)

{

bool status = false;

try

{

CustomerBL customerBLObj = new CustomerBL();

status = customerBLObj.UpdateReviewComments(rating);

}

catch (Exception)

{

status = false;

}

return status;

}

[HttpDelete]

public bool DeleteRating(Common.Models.Rating rating)

{

var status = false;

try

{

CustomerBL customerBLObj = new CustomerBL();

status = customerBLObj.DeleteRating(rating);

}

catch (Exception ex)

{

status = false;

}

return status;

}

}

……………

public class UserController : ApiController

{

[HttpPost]

public string ValidateUserCredentials(Common.Models.User userObj)

{

string userRole = string.Empty;

try

{

CustomerBL customerBLObj = new CustomerBL();

userRole = customerBLObj.ValidateUserCredentials(userObj.EmailId, userObj.UserPassword);

}

catch (Exception)

{

userRole="Invalid credentials";

}

return userRole;

}

[HttpPost]

public bool InsertUserDetails(Common.Models.User user)

{

var status = false;

try

{

CustomerBL customerBLObj = new CustomerBL();

status = customerBLObj.RegisterUserUsingAPI(user);

}

catch (Exception ex)

{

status = false;

}

return status;

}

[HttpPost]

public bool AddProductToCart(Common.Models.Cart cartObj)

{

var status = false;

try

{

CustomerBL customerBLObj = new CustomerBL();

var returnValue = customerBLObj.AddProductToCart(cartObj);

if (returnValue > 0)

{

status = true;

}

}

catch (Exception ex)

{

status = false;

}

return status;

}

[HttpGet]

public JsonResult<List<Common.Models.CartProductsDetails>> GetCartProducts(string emailId)

{

try

{

CustomerBL customerBLObj = new CustomerBL();

var cardProducts = customerBLObj.GetCartProducts(emailId);

return Json<List<Common.Models.CartProductsDetails>>(cardProducts);

}

catch (Exception)

{

return null;

}

}

[HttpPut]

public bool UpdateCartProducts(Common.Models.Cart cartObj)

{

bool status = false;

try

{

CustomerBL customerBLObj = new CustomerBL();

status = customerBLObj.UpdateCartProducts(cartObj);

}

catch (Exception)

{

status = false;

}

return status;

}

[HttpDelete]

public bool DeleteCartProduct(Common.Models.Cart cartObj)

{

var status = false;

try

{

CustomerBL customerBLObj = new CustomerBL();

status = customerBLObj.DeleteCartProduct(cartObj);

}

catch (Exception ex)

{

status = false;

}

return status;

}

}

}

======================== Infosys.QuickKart.Services.Repository==================

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using AutoMapper;

using Infosys.QuickKart.DataAccessLayer;

namespace Infosys.QuickKart.Services.Repository

{

public class QuickKartMapper<Source, Destination>

where Source : class

where Destination : class

{

public QuickKartMapper()

{

Mapper.CreateMap<Source, Destination>();

Mapper.CreateMap<Destination, Source>();

}

public Destination Translate(Source obj)

{

return Mapper.Map<Destination>(obj);

}

}

}

