**BACK END API SOURCE CODE**

package com.kitchenStory;

import java.util.Collections;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.Bean;

import springfox.documentation.builders.RequestHandlerSelectors;

import springfox.documentation.service.ApiInfo;

import springfox.documentation.spi.DocumentationType;

import springfox.documentation.spring.web.plugins.Docket;

@SpringBootApplication

public class KitchenStoryApiApplication {

public static void main(String[] args) {

SpringApplication.run(KitchenStoryApiApplication.class, args);

System.out.println("Kitchen Story App works!");

}

@Bean

public Docket swaggerConfiguration() {

return new Docket(DocumentationType.SWAGGER\_2)

.select()

.apis(RequestHandlerSelectors.basePackage("com.kitchenStory"))

.build()

.apiInfo(apiDetails());

}

private ApiInfo apiDetails() {

return new ApiInfo(

"KitchenStory API",

"Spring Boot REST API for kitchenStory.com",

"1.0",

"Free to use",

new springfox.documentation.service.Contact("Kalpana Rathod", "https://github.com/Kalp1992/", "kalpanarathod6@gmail.com"),

"API License",

"https://github.com/Kalp1992/",

Collections.emptyList()

);

}

}

ADMIN CONTROLLER

package com.kitchenStory.controllers.admin;

import java.util.NoSuchElementException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.models.Admin;

import com.kitchenStory.services.AdminAuthService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/admin")

public class AdminAuthController {

@Autowired

private AdminAuthService adminAuthService;

@GetMapping("/signIn")

public ResponseEntity<Admin> SignIn( String adminName, String password) {

try {

Admin admin = adminAuthService.SignInAdmin(adminName, password);

return new ResponseEntity<Admin>(admin, HttpStatus.OK);

} catch (NoSuchElementException e) {

return new ResponseEntity<Admin>(HttpStatus.NOT\_FOUND);

}

}

}

package com.kitchenStory.controllers.admin;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.models.Admin;

import com.kitchenStory.services.AdminService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/admin")

public class AdminController {

@Autowired

private AdminService adminService;

@PostMapping("/signUp")

public String create(Integer adminId, String Name,String adminName, String password) {

Admin admin = new Admin(adminId, Name, adminName, password);

try {

if (Boolean.TRUE.equals(this.adminService.AddAdmin(admin))) {

return "New Admin record added successfully";

}

} catch (SQLException ex) {

System.out.println("Exception occurred while inserting a new admin record!\n" + ex);

}

return "Sign up Failed";

}

}

package com.kitchenStory.controllers.admin;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.services.DeleteItemService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/item")

public class DeleteItemController {

@Autowired

private DeleteItemService deleteItemService;

@DeleteMapping("/delete/{id}")

public ResponseEntity<HttpStatus> deleteItem(@PathVariable("id") Integer id) {

try {

deleteItemService.DeleteItem(id);

return new ResponseEntity<>(HttpStatus.NO\_CONTENT);

} catch (Exception e) {

return new ResponseEntity<>(HttpStatus.INTERNAL\_SERVER\_ERROR);

}

}

}

package com.kitchenStory.controllers.admin;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.models.Admin;

import com.kitchenStory.services.GetAllAdminService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/admin")

public class GetAllAdminController {

@Autowired

private GetAllAdminService getAllAdminService;

@GetMapping("/all")

public List AdminDetail() {

List<Admin> allAdmin=getAllAdminService.GetallAdmin();

return allAdmin;

}

}

package com.kitchenStory.controllers.admin;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.models.Customer;

import com.kitchenStory.services.GetAllCustomerService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/customer")

public class GetAllCustomerController {

@Autowired

private GetAllCustomerService getAllCustomerService;

@GetMapping("/all")

public List CustomerDetail() {

List<Customer> allCustomer=getAllCustomerService.GetallCustomer();

return allCustomer;

}

}

package com.kitchenStory.controllers.admin;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.models.Item;

import com.kitchenStory.services.GetAllItemService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/admin")

public class GetAllItemController {

@Autowired

private GetAllItemService getAllItemService;

@GetMapping("/item/all")

public List ItemDetail() {

List<Item> allItem=getAllItemService.GetallProduct();

return allItem;

}

}

package com.kitchenStory.controllers.admin;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.models.Item;

import com.kitchenStory.services.ItemService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/item")

public class ItemController {

@Autowired

private ItemService itemService;

@PostMapping("/add")

public String create(Integer itemId, String itemName,String category, Double price,String itemVendor) {

Item item = new Item(itemId, itemName, category, price, itemVendor);

try {

if (Boolean.TRUE.equals(this.itemService.AddItem(item))) {

return "New Item record added successfully";

}

} catch (SQLException ex) {

System.out.println("Exception occurred while inserting a new item record!\n" + ex);

}

return "Item not added";

}

}

package com.kitchenStory.controllers.admin;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.PatchMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.services.UpdateAdminNameService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/admin")

public class UpdateAdminNameController {

@Autowired

private UpdateAdminNameService updateAdminNameService;

@PatchMapping("/{adminId}/update/adminName")

public String update(String adminName, Integer adminId) throws SQLException {

if (Boolean.TRUE.equals(this.updateAdminNameService.UpdateAdminName(adminName, adminId))) {

return String.format("AdminName of admin #%d updated successfully", adminId);

}

return String.format("Failure in updating the adminName of admin #%d!", adminId);

}

}

package com.kitchenStory.controllers.admin;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.PatchMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.services.UpdateAdminPasswordService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/admin")

public class UpdateAdminPasswordController {

@Autowired

private UpdateAdminPasswordService updatePasswordService ;

@PatchMapping("{adminid}/update/password")

public String update(Integer adminId,String password) throws SQLException {

if (Boolean.TRUE.equals(this.updatePasswordService.UpdateAdminPassword(password, adminId))) {

return String.format("Password of admin #%d updated successfully", adminId);

}

return String.format("Failure in updating the password of admin #%d!", adminId);

}

}

CUSTOMER CONTROLLER

package com.kitchenStory.controllers.customer;

import java.util.NoSuchElementException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.models.Customer;

import com.kitchenStory.services.CustomerAuthService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/customer")

public class CustomerAuthController {

@Autowired

private CustomerAuthService customerAuthService;

// private CustomerRepository customerRepository;

@PostMapping("/signIn")

public ResponseEntity<Customer> SignIn( String customerName, String password) {

try {

Customer customer = customerAuthService.SignInCustomer(customerName, password);

return new ResponseEntity<Customer>(customer, HttpStatus.OK);

} catch (NoSuchElementException e) {

return new ResponseEntity<Customer>(HttpStatus.NOT\_FOUND);

}

}

// public Status loginCustomer( @RequestBody Customer customer) {

// List<Customer> customers;

// try {

// customers = customerRepository.findAllCustomer();

// for (Customer other : customers) {

// if (other.equals(customer)) {

// // customer.setLoggedIn(true);

// // userRepository.save(user);

// return Status.SUCCESS;

// }

// }

// } catch (SQLException e) {

// // TODO Auto-generated catch block

// e.printStackTrace();

// }

//

//

// return Status.FAILURE;

// }

// public Map <String, String> signIn(@RequestBody MultiValueMap<String, String> credentials) {

// Map <String, String> signInResponse=new HashMap<>();

// Customer customerAuthenticated= customerauthService.SignInCustomer(credentials.get("customerName").get(0), credentials.get("password").get(0));

// if(customerAuthenticated != null) {

// signInResponse.put("status", "true");

// signInResponse.put("massege", "The customer has been authenticated successfully");

// }else {

// signInResponse.put("status", "false");

// signInResponse.put("massege", "Invalid credential!");

// }

// return signInResponse;

//

//

// }

}

package com.kitchenStory.controllers.customer;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.models.Customer;

import com.kitchenStory.services.CustomerService;

import io.swagger.v3.oas.models.parameters.RequestBody;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/customer")

public class CustomerController {

@Autowired

CustomerService customerService;

@PostMapping("/signUp")

public String create(Integer customerId, String Name,String customerName, String password) {

Customer customer = new Customer(customerId, Name, customerName, password);

try {

if (Boolean.TRUE.equals(this.customerService.AddCustomer(customer))) {

return "New Customer record added successfully";

}

} catch (SQLException ex) {

System.out.println("Exception occurred while inserting a new customer record!\n" + ex);

}

return "Sign up Failed";

}

}

// public Status signUp(@RequestBody Customer newCustomer) {

// List<Customer> customers1;

// try {

// customers1 = customerRepository.findAllCustomer();

// System.out.println("New Customer: " + customerRepository.toString());

// for (Customer customer : customers1) {

// System.out.println("Registered user: " + newCustomer.toString());

// if (customer.equals(newCustomer)) {

// System.out.println("User Already exists!");

// return Status.USER\_ALREADY\_EXISTS;

// }

// }

// customerRepository.createCustomer(newCustomer);

//

// } catch (SQLException e) {

// // TODO Auto-generated catch block

// e.printStackTrace();

// }

// return Status.SUCCESS;

// }

// }

// Map <String, String> signUpResponse=new HashMap<>();

// Boolean customerRegistered= customerService.AddCustomer(Integer.parseInt(newAdminDetails.get("customerId").get(0)),newAdminDetails.get("name").get(0),newAdminDetails.get("customerName").get(0), newAdminDetails.get("password").get(0));

// if(customerRegistered) {

// signUpResponse.put("status", "true");

// signUpResponse.put("massege", "The customer has been registered successfully");

// }else {

// signUpResponse.put("status", "false");

// signUpResponse.put("massege", "Invalid Info!");

// }

// return signUpResponse;

//

//

// }

//}

package com.kitchenStory.controllers.customer;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.models.Item;

import com.kitchenStory.services.FindFoodItemByNameService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/fooditem")

public class FindFoodItemByNameController {

@Autowired

private FindFoodItemByNameService findFoodItemByNameService;

@GetMapping("/GetbyName")

// public List FoodDetail(String itemName) {

// List<Food> allFood= findFoodByNameService.FindFoodByName(itemName);

// return allFood;

// }}

public ResponseEntity<List<Item>> findByItemName(String itemName) {

try {

List<Item> items = findFoodItemByNameService.FindFoodByName(itemName);

if (items.isEmpty()) {

return new ResponseEntity<>(HttpStatus.NO\_CONTENT);

}

return new ResponseEntity<>(items, HttpStatus.OK);

} catch (Exception e) {

return new ResponseEntity<>(HttpStatus.INTERNAL\_SERVER\_ERROR);

}

}}

**package** com.kitchenStory.controllers.customer;

**public** **enum** Status {

***SUCCESS***,

***USER\_ALREADY\_EXISTS***,

***FAILURE***

}

package com.kitchenStory.controllers.customer;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.PatchMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.kitchenStory.services.UpdateCustomerNameService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/customer")

public class UpdateCustomerNameController {

@Autowired

private UpdateCustomerNameService updateCustomerNameService;

@PatchMapping("/{customerId}/update/customerName")

public String update(String customerName, Integer customerId) throws SQLException {

if (Boolean.TRUE.equals(this.updateCustomerNameService.UpdateCustomerName(customerName, customerId))) {

return String.format("CustomerName of customer #%d updated successfully", customerId);

}

return String.format("Failure in updating the customerName of customer #%d!", customerId);

}

}

**package** com.kitchenStory.controllers.customer;

**import** java.sql.SQLException;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.web.bind.annotation.CrossOrigin;

**import** org.springframework.web.bind.annotation.PatchMapping;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RestController;

**import** com.kitchenStory.services.UpdateCustomerPasswordService;

@CrossOrigin(origins = "http://localhost:4200")

@RestController

@RequestMapping("/api/customer")

**public** **class** UpdateCustomerPasswordController {

@Autowired

**private** UpdateCustomerPasswordService updateCustomerPasswordService;

@PatchMapping("/{customerId}/update/password")

**public** String update(String password, Integer customerId) **throws** SQLException {

**if** (Boolean.***TRUE***.equals(**this**.updateCustomerPasswordService.UpdateCustomerPassword(password, customerId))) {

**return** String.*format*("Password of customer #%d updated successfully", customerId);

}

**return** String.*format*("Failure in updating the password of customer #%d!", customerId);

}

}

DAO

package com.kitchenStory.daos;

import java.sql.SQLException;

import java.util.ArrayList;

import com.kitchenStory.models.Admin;

public interface AdminDao {

Integer createAdmin(Admin admin) throws SQLException ;

// Read/Retrieve Operations

Admin findAdminByAdminNameandPassword(String adminName, String password) throws SQLException ;

Integer updateAdminName(Admin admin) throws SQLException;

Integer updateAdminPassword(Admin admin) throws SQLException;

Admin findByAdminId(Integer adminId) throws SQLException ;

ArrayList<Admin> findAllAdmin() throws SQLException ;

}

package com.kitchenStory.daos;

import java.sql.SQLException;

import java.util.ArrayList;

import com.kitchenStory.models.Customer;

public interface CustomerDao {

Integer createCustomer(Customer customer) throws SQLException ;

// Read/Retrieve Operations

Customer findCustomerByCustomerNameandPassword(String customerName, String password) throws SQLException ;

Integer updateCustomerName(Customer customer) throws SQLException;

Integer updateCustomerPassword(Customer customer) throws SQLException;

ArrayList<Customer> findAllCustomer() throws SQLException ;

Customer findByCustomerId(Integer customerId) throws SQLException ;

}

package com.kitchenStory.daos;

import java.sql.SQLException;

import java.util.ArrayList;

import com.kitchenStory.models.Item;

public interface ItemDao {

Integer createItem(Item item) throws SQLException ;

//Item findProductByItemId(Integer itemId) throws SQLException ;

ArrayList<Item> findAllItem() throws SQLException;

Integer deleteItemByItemId(Integer itemId) throws SQLException;

Item findByItemId(Integer itemId) throws SQLException ;

ArrayList<Item> findItemByName(String itemName) throws SQLException ;

}

MODEL

package com.kitchenStory.models;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

@ApiModel(description="Details about the admin")

public class Admin {

@ApiModelProperty(notes="The unique adminId of the admin")

Integer adminId;

@ApiModelProperty(notes="The name of the admin")

String name;

@ApiModelProperty(notes="The admin name of the admin")

String adminName;

@ApiModelProperty(notes="The password of admin")

String password;

public Admin() {

super();

}

public Admin(Integer adminId) {

super();

this.adminId = adminId;

}

public Admin(String adminName, String password) {

super();

this.adminName = adminName;

this.password = password;

}

public Admin(Integer adminId, String name, String adminName, String password) {

super();

this.adminId = adminId;

this.name = name;

this.adminName = adminName;

this.password = password;

}

public Integer getAdminId() {

return adminId;

}

public void setAdminId(Integer adminId) {

this.adminId = adminId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getAdminName() {

return adminName;

}

public void setAdminName(String adminName) {

this.adminName = adminName;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

@Override

public String toString() {

return "Admin [adminId=" + adminId + ", name=" + name + ", adminName=" + adminName + ", password=" + password

+ "]";

}

}

package com.kitchenStory.models;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

@ApiModel(description="Details about the customer")

public class Customer {

@ApiModelProperty(notes="The unique customerId of the customer")

Integer customerId;

@ApiModelProperty(notes="The name of the customer")

String name;

@ApiModelProperty(notes="The customer name of customer")

String customerName;

@ApiModelProperty(notes="The password of the customer")

String password;

public Customer() {

super();

}

public Customer(Integer customerId) {

super();

this.customerId = customerId;

}

public Customer(String customerName, String password) {

super();

this.customerName = customerName;

this.password = password;

}

public Customer(Integer customerId, String name, String customerName, String password) {

super();

this.customerId = customerId;

this.name = name;

this.customerName = customerName;

this.password = password;

}

public Integer getCustomerId() {

return customerId;

}

public void setCustomerId(Integer customerId) {

this.customerId = customerId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCustomerName() {

return customerName;

}

public void setCustomerName(String customerName) {

this.customerName = customerName;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

@Override

public String toString() {

return "Customer [customerId=" + customerId + ", name=" + name + ", customerName=" + customerName + ", password="

+ password + "]";

}

}

package com.kitchenStory.models;

import io.swagger.annotations.ApiModel;

import io.swagger.annotations.ApiModelProperty;

@ApiModel(description="Details about the customer")

public class Item {

@ApiModelProperty(notes="The unique itemId of the food item")

Integer itemId;

@ApiModelProperty(notes="The name of the food item")

String itemName;

@ApiModelProperty(notes="The category of the food item")

String category;

@ApiModelProperty(notes="The price of the food item")

Double price;

@ApiModelProperty(notes="The vendor of the food item")

String itemVendor;

public Item() {

super();

}

public Item(Integer itemId) {

super();

this.itemId = itemId;

}

public Item(String itemName) {

super();

this.itemName = itemName;

}

public Item(Integer itemId, String itemName, String category, Double price, String itemVendor) {

super();

this.itemId = itemId;

this.itemName = itemName;

this.category = category;

this.price = price;

this.itemVendor = itemVendor;

}

public Integer getItemId() {

return itemId;

}

public void setItemId(Integer itemId) {

this.itemId = itemId;

}

public String getItemName() {

return itemName;

}

public void setItemName(String itemName) {

this.itemName = itemName;

}

public String getCategory() {

return category;

}

public void setCategory(String category) {

this.category = category;

}

public Double getPrice() {

return price;

}

public void setPrice(Double price) {

this.price = price;

}

public String getItemVendor() {

return itemVendor;

}

public void setItemVendor(String itemVendor) {

this.itemVendor = itemVendor;

}

@Override

public String toString() {

return "Item [itemId=" + itemId + ", itemName=" + itemName + ", category=" + category + ", price=" + price

+ ", itemVendor=" + itemVendor + "]";

}

}

REPOSITORY

**package** com.kitchenStory.repositories;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.ArrayList;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Repository;

**import** com.kitchenStory.daos.AdminDao;

**import** com.kitchenStory.models.Admin;

**import** com.kitchenStory.utils.DatabaseConnection;

@Repository

**public** **class** AdminRepository **implements** AdminDao {

@Autowired

**private** DatabaseConnection connection;

@Override

**public** Integer createAdmin(Admin admin) **throws** SQLException {

String insertAdminFormat = "INSERT INTO admins (ADMIN\_ID, NAME, ADMIN\_NAME, PASSWORD) VALUES ( ?, ?, ?, ? )";

**try** (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(insertAdminFormat);) {

preparedStatement.setInt(1, admin.getAdminId());

preparedStatement.setString(2, admin.getName());

preparedStatement.setString(3, admin.getAdminName());

preparedStatement.setString(4, admin.getPassword());

**return** preparedStatement.executeUpdate();

}

}

@Override

**public** Admin findAdminByAdminNameandPassword(String adminName, String password) **throws** SQLException {

String getAdminFormat = "SELECT \*FROM admins WHERE ADMIN\_NAME = ? and PASSWORD=?";

**try** (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(getAdminFormat);) {

preparedStatement.setString(1, adminName);

preparedStatement.setString(2, password);

ResultSet rs = preparedStatement.executeQuery();

**if** (!rs.isBeforeFirst()) {

**return** **null**;

}

Admin admin = **new** Admin();

**while** (rs.next()) {

admin.setAdminId(rs.getInt("ADMIN\_ID"));

admin.setName(rs.getString("NAME"));

admin.setAdminName(rs.getString("ADMIN\_NAME"));

admin.setPassword(rs.getString("PASSWORD"));

}

**return** admin;

}

}

@Override

**public** Integer updateAdminName(Admin admin) **throws** SQLException {

String updateAdminNameFormat = "UPDATE admins SET ADMIN\_NAME = ? WHERE ADMIN\_ID = ?";

**try** (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(updateAdminNameFormat);) {

preparedStatement.setString(1, admin.getAdminName());

preparedStatement.setInt(2, admin.getAdminId());

**return** preparedStatement.executeUpdate();

}

}

@Override

**public** Integer updateAdminPassword(Admin admin) **throws** SQLException {

String updateAdminPasswodFormat = "UPDATE admins SET PASSWORD = ? WHERE ADMIN\_ID = ?";

**try** (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(updateAdminPasswodFormat);) {

preparedStatement.setString(1, admin.getPassword());

preparedStatement.setInt(2, admin.getAdminId());

**return** preparedStatement.executeUpdate();

}

}

@Override

**public** Admin findByAdminId(Integer adminId) **throws** SQLException {

String getAdminFormat = "SELECT \*FROM admins WHERE ADMIN\_ID = ? ";

**try** (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(getAdminFormat);) {

preparedStatement.setInt(1, adminId);

ResultSet rs = preparedStatement.executeQuery();

**if** (!rs.isBeforeFirst()) {

**return** **null**;

}

Admin admin = **new** Admin();

**while** (rs.next()) {

admin.setAdminId(rs.getInt("ADMIN\_ID"));

admin.setName(rs.getString("NAME"));

admin.setAdminName(rs.getString("ADMIN\_NAME"));

admin.setPassword(rs.getString("PASSWORD"));

}

**return** admin;

}

}

@Override

**public** ArrayList<Admin> findAllAdmin() **throws** SQLException {

String getAdminFormat = "SELECT \*FROM admins";

**try** (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(getAdminFormat);) {

ResultSet rs = preparedStatement.executeQuery();

ArrayList<Admin> admins = **new** ArrayList<>();

**if** (!rs.isBeforeFirst()) {

**return** admins;

}

**while** (rs.next()) {

Admin admin = **new** Admin();

admin.setAdminId(rs.getInt("ADMIN\_ID"));

admin.setName(rs.getString("NAME"));

admin.setAdminName(rs.getString("ADMIN\_NAME"));

admin.setPassword(rs.getString("PASSWORD"));

admins.add(admin);

}

**return** admins;

}

}

}

package com.kitchenStory.repositories;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Repository;

import com.kitchenStory.daos.CustomerDao;

import com.kitchenStory.models.Customer;

import com.kitchenStory.utils.DatabaseConnection;

@Repository

public class CustomerRepository implements CustomerDao {

@Autowired

private DatabaseConnection connection;

@Override

public Integer createCustomer(Customer customer) throws SQLException {

String insertCustomerFormat = "INSERT INTO customers (CUSTOMER\_ID, NAME, CUSTOMER\_NAME, PASSWORD) VALUES ( ?, ?, ?, ? )";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(insertCustomerFormat);) {

preparedStatement.setInt(1, customer.getCustomerId());

preparedStatement.setString(2, customer.getName());

preparedStatement.setString(3, customer.getCustomerName());

preparedStatement.setString(4, customer.getPassword());

return preparedStatement.executeUpdate();

}

}

@Override

public Integer updateCustomerPassword(Customer customer) throws SQLException {

String updateCustomerPasswodFormat = "UPDATE customers SET PASSWORD = ? WHERE CUSTOMER\_ID = ?";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(updateCustomerPasswodFormat);) {

preparedStatement.setString(1, customer.getPassword());

preparedStatement.setInt(2, customer.getCustomerId());

return preparedStatement.executeUpdate();

}

}

@Override

public Customer findCustomerByCustomerNameandPassword(String customerName, String password) throws SQLException {

String getCustomerFormat = "SELECT \*FROM customers WHERE CUSTOMER\_NAME = ? and PASSWORD=?";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(getCustomerFormat);) {

preparedStatement.setString(1, customerName);

preparedStatement.setString(2, password);

ResultSet rs = preparedStatement.executeQuery();

if (!rs.isBeforeFirst()) {

return null;

}

Customer customer = new Customer();

while (rs.next()) {

customer.setCustomerId(rs.getInt("CUSTOMER\_ID"));

customer.setName(rs.getString("NAME"));

customer.setCustomerName(rs.getString("CUSTOMER\_NAME"));

customer.setPassword(rs.getString("PASSWORD"));

}

return customer;

}

}

@Override

public Integer updateCustomerName(Customer customer) throws SQLException {

String updateCustomerNameFormat = "UPDATE customers SET CUSTOMER\_NAME = ? WHERE CUSTOMER\_ID = ?";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(updateCustomerNameFormat);) {

preparedStatement.setString(1, customer.getCustomerName());

preparedStatement.setInt(2, customer.getCustomerId());

return preparedStatement.executeUpdate();

}

}

@Override

public ArrayList<Customer> findAllCustomer() throws SQLException {

String getCustomersFormat = "SELECT \*FROM customers";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(getCustomersFormat);) {

ResultSet rs = preparedStatement.executeQuery();

ArrayList<Customer> customers = new ArrayList<>();

if (!rs.isBeforeFirst()) {

return customers;

}

while (rs.next()) {

Customer customer = new Customer();

customer.setCustomerId(rs.getInt("CUSTOMER\_ID"));

customer.setName(rs.getString("NAME"));

customer.setCustomerName(rs.getString("CUSTOMER\_NAME"));

customer.setPassword(rs.getString("PASSWORD"));

customers.add(customer);

}

return customers;

}

}

@Override

public Customer findByCustomerId(Integer customerId) throws SQLException {

String getCustomerFormat = "SELECT \*FROM customers WHERE CUSTOMER\_ID = ? ";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(getCustomerFormat);) {

preparedStatement.setInt(1, customerId);

ResultSet rs = preparedStatement.executeQuery();

if (!rs.isBeforeFirst()) {

return null;

}

Customer customer = new Customer();

while (rs.next()) {

customer.setCustomerId(rs.getInt("CUSTOMER\_ID"));

customer.setName(rs.getString("NAME"));

customer.setCustomerName(rs.getString("CUSTOMER\_NAME"));

customer.setPassword(rs.getString("PASSWORD"));

}

return customer;

}

}

}

package com.kitchenStory.repositories;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Repository;

import com.kitchenStory.daos.ItemDao;

import com.kitchenStory.models.Item;

import com.kitchenStory.utils.DatabaseConnection;

@Repository

public class ItemRepository implements ItemDao {

@Autowired

private DatabaseConnection connection;

@Override

public Integer createItem(Item item) throws SQLException {

String insertItemFormat = "INSERT INTO items (ITEM\_ID, ITEM\_NAME,CATEGORY, PRICE, ITEM\_VENDOR) VALUES ( ?, ?, ?, ?, ? )";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(insertItemFormat);) {

preparedStatement.setInt(1, item.getItemId());

preparedStatement.setString(2, item.getItemName());

preparedStatement.setString(3, item.getCategory());

preparedStatement.setDouble(4, item.getPrice());

preparedStatement.setString(5, item.getItemVendor());

return preparedStatement.executeUpdate();

}

}

@Override

public ArrayList<Item> findAllItem() throws SQLException {

String getItemsFormat = "SELECT \*FROM items";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(getItemsFormat);) {

ResultSet rs = preparedStatement.executeQuery();

ArrayList<Item> items = new ArrayList<>();

if (!rs.isBeforeFirst()) {

return items;

}

while (rs.next()) {

Item item = new Item();

item.setItemId(rs.getInt("ITEM\_ID"));

item.setItemName(rs.getString("ITEM\_NAME"));

item.setCategory(rs.getString("CATEGORY"));

item.setPrice(rs.getDouble("PRICE"));

item.setItemVendor(rs.getString("ITEM\_VENDOR"));

items.add(item);

}

return items;

}

}

@Override

public Integer deleteItemByItemId(Integer itemId) throws SQLException {

String deleteItemFormat = " DELETE FROM items WHERE ITEM\_ID = ?";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(deleteItemFormat);) {

preparedStatement.setInt(1, itemId);

return preparedStatement.executeUpdate();

}

}

@Override

public Item findByItemId(Integer itemId) throws SQLException {

String getItemFormat = "SELECT \*FROM items WHERE ADMIN\_ID = ? ";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(getItemFormat);) {

preparedStatement.setInt(1, itemId);

ResultSet rs = preparedStatement.executeQuery();

if (!rs.isBeforeFirst()) {

return null;

}

Item item = new Item();

while (rs.next()) {

item.setItemId(rs.getInt("ITEM\_ID"));

item.setItemName(rs.getString("ITEM\_NAME"));

item.setCategory(rs.getString("CATEGORY"));

item.setPrice(rs.getDouble("PRICE"));

item.setItemVendor(rs.getString("ITEM\_VENDOR"));

}

return item;

}

}

@Override

public ArrayList<Item> findItemByName(String itemName) throws SQLException {

String getItemFormat = "SELECT \*FROM items WHERE ITEM\_NAME = ? ";

try (PreparedStatement preparedStatement = connection.getConnection().prepareStatement(getItemFormat);) {

preparedStatement.setString(1, itemName);

ResultSet rs = preparedStatement.executeQuery();

ArrayList<Item> items = new ArrayList<>();

if (!rs.isBeforeFirst()) {

return items;

}

Item item = new Item();

while (rs.next()) {

item.setItemId(rs.getInt("ITEM\_ID"));

item.setItemName(rs.getString("ITEM\_NAME"));

item.setCategory(rs.getString("CATEGORY"));

item.setPrice(rs.getDouble("PRICE"));

item.setItemVendor(rs.getString("ITEM\_VENDOR"));

items.add(item);

}

return items;

}

}

}

SERVICES

package com.kitchenStory.services;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Admin;

import com.kitchenStory.repositories.AdminRepository;

@Service

public class AdminAuthService {

@Autowired

private AdminRepository adminRepository;

public Admin SignInAdmin(String adminName,String password) {

Admin admin= new Admin(adminName, password);

try {

return adminRepository.findAdminByAdminNameandPassword(adminName, password);

} catch (SQLException e) {

e.printStackTrace();

}

return null;

}

}

**package** com.kitchenStory.services;

**import** java.sql.SQLException;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Service;

**import** com.kitchenStory.models.Admin;

**import** com.kitchenStory.repositories.AdminRepository;

@Service

**public** **class** AdminService {

@Autowired

**private** AdminRepository adminRepository;

**public** Boolean AddAdmin(Admin admin) **throws** SQLException {

Integer recordsInserted = **this**.adminRepository.createAdmin(admin);

**return** recordsInserted > 0;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Customer;

import com.kitchenStory.repositories.CustomerRepository;

@Service

public class CustomerAuthService {

@Autowired

private CustomerRepository customerRepository;

public Customer SignInCustomer(String customerName,String password) {

Customer customer= new Customer(customerName, password);

try {

return customerRepository.findCustomerByCustomerNameandPassword(customerName, password);

} catch (SQLException e) {

e.printStackTrace();

}

return null;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Customer;

import com.kitchenStory.repositories.CustomerRepository;

@Service

public class CustomerService {

@Autowired

private CustomerRepository customerRepository;

public Boolean AddCustomer(Customer customer) throws SQLException {

Integer recordsInserted = this.customerRepository.createCustomer(customer);

return recordsInserted > 0;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Item;

import com.kitchenStory.repositories.ItemRepository;

@Service

public class DeleteItemService {

@Autowired

private ItemRepository itemRepository;

public Boolean DeleteItem(Integer itemId) {

Item item= new Item(itemId);

try {

return itemRepository.deleteItemByItemId(itemId)>0;

} catch (SQLException e) {

e.printStackTrace();

}

return false;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import java.util.ArrayList;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Item;

import com.kitchenStory.repositories.ItemRepository;

@Service

public class FindFoodItemByNameService {

@Autowired

private ItemRepository itemRepository;

public ArrayList<Item> FindFoodByName(String itemName) {

Item item= new Item(itemName);

try {

return itemRepository.findItemByName(itemName);

} catch (SQLException e) {

e.printStackTrace();

}

return null;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import java.util.ArrayList;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Admin;

import com.kitchenStory.repositories.AdminRepository;

@Service

public class GetAllAdminService {

@Autowired

private AdminRepository adminRepository;

public ArrayList<Admin> GetallAdmin() {

Admin admin= new Admin();

try {

return adminRepository.findAllAdmin();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return null;

}

}

**package** com.kitchenStory.services;

**import** java.sql.SQLException;

**import** java.util.ArrayList;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Service;

**import** com.kitchenStory.models.Customer;

**import** com.kitchenStory.repositories.CustomerRepository;

@Service

**public** **class** GetAllCustomerService {

@Autowired

**private** CustomerRepository customerRepository;

**public** ArrayList<Customer> GetallCustomer() {

Customer customer= **new** Customer();

**try** {

**return** customerRepository.findAllCustomer();

} **catch** (SQLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**return** **null**;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import java.util.ArrayList;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Item;

import com.kitchenStory.repositories.ItemRepository;

@Service

public class GetAllItemService {

@Autowired

private ItemRepository itemRepository;

public ArrayList<Item> GetallProduct() {

Item item= new Item();

try {

return itemRepository.findAllItem();

} catch (SQLException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return null;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Item;

import com.kitchenStory.repositories.ItemRepository;

@Service

public class ItemService {

@Autowired

private ItemRepository itemRepository;

public Boolean AddItem(Item item ) throws SQLException {

Integer recordsInserted = this.itemRepository.createItem(item);

return recordsInserted > 0;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Admin;

import com.kitchenStory.repositories.AdminRepository;

@Service

public class UpdateAdminNameService {

@Autowired

private AdminRepository adminRepository;

public Boolean UpdateAdminName(String adminName, Integer adminId) throws SQLException {

Admin admin = this.adminRepository.findByAdminId(adminId);

admin.setAdminName(adminName);

Integer recordsUpdated = this.adminRepository.updateAdminName(admin);

return recordsUpdated > 0;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Admin;

import com.kitchenStory.repositories.AdminRepository;

@Service

public class UpdateAdminPasswordService {

@Autowired

private AdminRepository adminRepository;

public Boolean UpdateAdminPassword(String password, Integer adminId) throws SQLException {

Admin admin = this.adminRepository.findByAdminId(adminId);

admin.setPassword(password);

Integer recordsUpdated = this.adminRepository.updateAdminPassword(admin);

return recordsUpdated > 0;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Customer;

import com.kitchenStory.repositories.CustomerRepository;

@Service

public class UpdateCustomerNameService {

@Autowired

private CustomerRepository customerRepository;

public Boolean UpdateCustomerName(String customerName,Integer customerId) throws SQLException {

Customer customer = this.customerRepository.findByCustomerId(customerId);

customer.setCustomerName(customerName);

Integer recordsUpdated = this.customerRepository.updateCustomerName(customer);

return recordsUpdated > 0;

}

}

package com.kitchenStory.services;

import java.sql.SQLException;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.kitchenStory.models.Customer;

import com.kitchenStory.repositories.CustomerRepository;

@Service

public class UpdateCustomerPasswordService {

@Autowired

private CustomerRepository customerRepository;

public Boolean UpdateCustomerPassword(String password, Integer customerId) throws SQLException {

Customer customer = this.customerRepository.findByCustomerId(customerId);

customer.setPassword(password);

Integer recordsUpdated = this.customerRepository.updateCustomerPassword(customer);

return recordsUpdated > 0;

}

}

UTILS

package com.kitchenStory.utils;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import javax.annotation.PostConstruct;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Component;

@Component

public class DatabaseConnection {

private Connection connection = null;

DatabaseConnection(

@Value("${db.url}")

String url,

@Value("${db.username}")

String username,

@Value("${db.password}")

String password) throws SQLException {

super();

this.connection = DriverManager.getConnection(url, username, password);

}

@PostConstruct

public void init() throws SQLException {

printConnectionDetails();

}

@SuppressWarnings("unused")

private void printConnectionDetails() throws SQLException {

System.out.println("Connected to database server "

+ this.connection.getMetaData().getDatabaseProductName()

+ " version: "

+ this.connection.getMetaData().getDatabaseProductVersion()

+ "\n");

}

public Connection getConnection() {

return connection;

}

}