File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\SlCapstoneFoodboxApplication.java

```java
package com.slbootcamp.foodbox;

import com.slbootcamp.foodbox.jdbc.FoodDao;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SlCapstoneFoodboxApplication implements CommandLineRunner {
    private Logger logger = LoggerFactory.getLogger(this.getClass());

    @Autowired
    FoodDao foodDao;
    public static void main(String[] args) {
        SpringApplication.run(SlCapstoneFoodboxApplication.class, args);
    }

    @Override
    public void run(String... args) throws Exception {
        logger.info(" - FindAll() ---> {}","x");
        logger.info(" - FindAll() ---> {}", foodDao.getAllFoods());
    }
}

```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\jdbc\FoodDao.java

```java
package com.slbootcamp.foodbox.jdbc;

import com.slbootcamp.foodbox.entity.Food;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.stereotype.Repository;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.List;

@Repository
public class FoodDao {
    @Autowired
    JdbcTemplate jdbcTemplate;

    private Logger logger = LoggerFactory.getLogger(this.getClass());
    final String INSERT_FOOD_ITEM = "INSERT INTO food " +
            " (CUISINE, DESCRIPTION, food_NAME, IMAGE_URL, IS_ENABLED, PRICE)" +
            " VALUES (?,?,?,?,?,?)";

    final String UPDATE_FOOD_ITEM = "UPDATE food SET cuisine = ?, description = ?, food_name = ?, image_url = ?, " +
            "is_enabled = ?, price = ? WHERE id = ?";

    final String SELECT_ALL_FOOD = "SELECT * FROM food order by ID DESC";

    final String SELECT_FOOD_ITEM = "SELECT * FROM food WHERE food.id = ?";

    final String DELETE_FOOD_ITEM = "DELETE FROM food WHERE id = ?";

    final String TOOGLE_FOOD_ITEM = "UPDATE food SET is_enabled = ? WHERE id = ?";

    public List<Food> getAllFoods() {

        return jdbcTemplate.query(SELECT_ALL_FOOD, new BeanPropertyRowMapper<>(Food.class));
    }


    public int addFoodItem(Food food) {
        ;
        return jdbcTemplate.update(INSERT_FOOD_ITEM, food.getCuisine(), food.getDescription(), food.getFoodName(),
                food.getImageUrl(), food.getIsEnabled(), food.getPrice());
    }

    public int updateFoodItem(Food food) {
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\jdbc\FoodDao.java

```
50          return jdbcTemplate.update(UPDATE_FOOD_ITEM, food.getCuisine(), food.getDescription(), food.getFoodName(),
51                  food.getImageUrl(), food.getIsEnabled(), food.getPrice(), food.getId());
52      }
53
54      public Food getFoodItem(int foodId) {
55          logger.info("Query for getFoodItem for SELECT_FOOD_ITEM: " + SELECT_FOOD_ITEM);
56          return jdbcTemplate.queryForObject(SELECT_FOOD_ITEM, new FoodMapper(), foodId);
57      }
58
59      public int deleteFoodItem(int foodId) {
60          return jdbcTemplate.update(DELETE_FOOD_ITEM, foodId);
61      }
62
63      public int toogleFoodItem(Food food) {
64          logger.info("toogleFoodItem: " + food);
65          return jdbcTemplate.update(TOOGLE_FOOD_ITEM, food.getIsEnabled(), food.getId());
66      }
67
68      private static final class FoodMapper implements RowMapper<Food> {
69          public Food mapRow(ResultSet rs, int rowNum) throws SQLException {
70              Food food = new Food();
71              food.setId(rs.getInt("ID"));
72              food.setCuisine(rs.getString("CUISINE"));
73              food.setDescription(rs.getString("DESCRIPTION"));
74              food.setFoodName(rs.getString("FOOD_NAME"));
75              food.setImageUrl(rs.getString("IMAGE_URL"));
76              food.setIsEnabled(rs.getString("IS_ENABLED"));
77              food.setPrice(rs.getLong("PRICE"));
78              return food;
79          }
80      }
81  }
82
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\jdbc\UserDao.java

```java
package com.slbootcamp.foodbox.jdbc;

import com.slbootcamp.foodbox.entity.User;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.dao.EmptyResultDataAccessException;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.stereotype.Repository;

import java.sql.ResultSet;
import java.sql.SQLException;

@Repository
public class UserDao {
    @Autowired
    JdbcTemplate jdbcTemplate;

    private final Logger logger = LoggerFactory.getLogger(this.getClass());

    private final String SELECT_USER = "SELECT * FROM user WHERE username=? AND password=? ";

    public User login(User user) {
            return jdbcTemplate.queryForObject(SELECT_USER, new UserMapper(), user.getUsername(), user.getPassword());

    }

    public static final class UserMapper implements RowMapper<User>{
        public User mapRow(ResultSet rs, int rowNum) throws SQLException {
            User user = new User();
            user.setId(rs.getLong("id"));
            user.setUsername(rs.getString("username"));
            user.setName(rs.getString("name"));
            user.setPassword(rs.getString("password"));
            user.setProfile(rs.getString("profile"));
            user.setContact(rs.getString("contact"));
            user.setCredit(rs.getString("credit"));

            return user;
        }
    }
}

```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\jdbc\OrdenDao.java

```java
package com.slbootcamp.foodbox.jdbc;

import com.slbootcamp.foodbox.entity.DisplayOrden;
import com.slbootcamp.foodbox.entity.Food;
import com.slbootcamp.foodbox.entity.Orden;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.dao.DataAccessException;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.stereotype.Repository;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

@Repository
public class OrdenDao {

    @Autowired
    JdbcTemplate jdbcTemplate;

    String INSERT_FOOD_ITEM = "INSERT INTO box (ID, FOOD_ID, QUANTITY) VALUES (?,?,?)";
    final String INSERT_ORDEN = "INSERT INTO orden (BOX_ID, USER_ID, STATUS) VALUES (?,?,'PENDING')";

    final String SELECT_ORDEN = "SELECT user.username, orden.id, CONCAT(food.food_name, \" (\",box.quantity,\")\") as food_name, food.price, orden.status " +
            " FROM food, user, box, orden " +
            " WHERE orden.box_id = box.id " +
            " AND orden.user_id = user.id " +
            " AND box.food_id = food.id " +
            " AND user.id = ? " +
            " ORDER BY orden.id DESC";

    final String UPDATE_CLIENT = "UPDATE user SET NAME = ?, CONTACT=?, CREDIT = ? WHERE ID = ? ";


    private Logger logger = LoggerFactory.getLogger(this.getClass());

    public List<DisplayOrden> getDisplayOrden(int ordenId) {
        return jdbcTemplate.query(SELECT_ORDEN, new OrdenDao.DisplayOrdenMapper(), ordenId);
    }

    public int saveOrden(Orden orden) {
        String pkBox = generatePK();
        //Save cart items
```

```
49
50          List<Food> cart = orden.getCart();
51          Map<Integer, Integer> foodMap = new <Integer, Integer>HashMap();
52
53          for (Food food : cart) {
54              Integer cantidad = 0;
55              if(foodMap.containsKey(food.getId())){
56                  logger.info(" ---------------- containsKey: " + food.getId() + " ----------------");
57                  cantidad = foodMap.get(food.getId());
58                  logger.info(" ---------------- cantidad: " + cantidad + " ----------------");
59              }
60
61              foodMap.put(food.getId(), ++cantidad);
62
63
64          }
65
66          // INSERT
67          for (Map.Entry<Integer, Integer> foodEntry : foodMap.entrySet()) {
68              try {
69                  jdbcTemplate.update(INSERT_FOOD_ITEM, pkBox, String.valueOf(foodEntry.getKey()), String.valueOf(foodEntry.getValue()));
70
71              } catch (DataAccessException e) {
72                  logger.error("Could not save food");
73
74              }
75          }
76
77
78          //save orden using user_Id and status="PENDING"
79          jdbcTemplate.update(INSERT_ORDEN, pkBox, orden.getUser().getId());
80
81          jdbcTemplate.update(UPDATE_CLIENT, orden.getUser().getName(), orden.getUser().getContact(), orden.getUser().getCredit(), orden.
    getUser().getId());
82          return 1;
83      }
84
85      private static String generatePK() {
86          return String.valueOf(System.currentTimeMillis());
87      }
88
89      private static final class DisplayOrdenMapper implements RowMapper<DisplayOrden> {
90          public DisplayOrden mapRow(ResultSet rs, int rowNum) throws SQLException {
91              DisplayOrden displayOrden = new DisplayOrden();
92              displayOrden.setId(rs.getInt("ID"));
93              displayOrden.setUsername(rs.getString("USERNAME"));
94              displayOrden.setFoodName(rs.getString("FOOD_NAME"));
95              displayOrden.setPrice(rs.getInt("PRICE"));
96              displayOrden.setStatus(rs.getString("STATUS"));
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\jdbc\OrdenDao.java

```
 97              return displayOrden;
 98          }
 99      }
100
101
102 }
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\entity\Cart.java

```java
package com.slbootcamp.foodbox.entity;

import lombok.Data;

@Data
public class Cart {
    private Long id;
    private Food foods;
}

```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\entity\Food.java

```java
package com.slbootcamp.foodbox.entity;

import lombok.Data;

@Data
public class Food {
    private int id;
    private String foodName;
    private Long price;
    private String cuisine;
    private String description;
    private String isEnabled;
    private String imageUrl;
}

```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\entity\User.java

```java
package com.slbootcamp.foodbox.entity;

import lombok.Data;

@Data
public class User {
    private Long id;
    private String username;
    private String name;
    private String password;
    private String contact;
    private String credit;
    private String profile;
}

```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\entity\Orden.java

```java
package com.slbootcamp.foodbox.entity;

import lombok.Data;

import java.util.List;

@Data
public class Orden {

    private String status;
    private List<Food> cart;
    private User user;

}

```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\entity\DisplayOrden.java

```java
package com.slbootcamp.foodbox.entity;

import lombok.Data;

@Data
public class DisplayOrden {
    private int id;
    private String username;
    private String foodName;
    private int price;
    private String status;
}

```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\controller\FoodController.java

```java
package com.slbootcamp.foodbox.controller;

import com.slbootcamp.foodbox.entity.Food;
import com.slbootcamp.foodbox.jdbc.FoodDao;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;

/*
 * find about the ResponseTransfer
 */

@CrossOrigin(origins ="*", allowedHeaders = "*")
@RestController
public class FoodController {
    private Logger logger = LoggerFactory.getLogger(this.getClass());

    public FoodController() {
    }


    @Autowired
    FoodDao foodDao;

    @GetMapping("/")
    public String getWelcome() {
        return "Welcome to the Spring Boot FoodBox!";
    }

    @GetMapping("/food/getAllFoods")
    public List<Food> getAllFoods() {
        return foodDao.getAllFoods();
    }



    @GetMapping("/food/getFoodItem/{foodId}")
    public Food getFoodItem(@PathVariable("foodId") int foodId) {

        return foodDao.getFoodItem(foodId);
    }

    @PostMapping("/food/updateFoodItem") //-0
    public int updateFoodItem(@RequestBody Food food){
        logger.info("---------------> UpdateFoodItem: " + food);
        return foodDao.updateFoodItem(food);
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\controller\FoodController.java

```
50      }
51
52
53      @PostMapping("/food/addFoodItem")
54      public int addFoodItem(@RequestBody Food food) {
55          logger.info("----------------> food " + food);
56          int ret = foodDao.addFoodItem(food);
57          logger.info("----------------> " + String.valueOf(ret));
58          return ret;
59      }
60      //    DELETE -> "/deleteFoodItem"(FoodItem) -> remove the food item
61      @DeleteMapping("/food/deleteFoodItem/{foodId}")
62      public int deleteFoodItem(@PathVariable("foodId") int foodId) {
63          logger.info("----------------> food " + foodId);
64          int ret = foodDao.deleteFoodItem(foodId);
65          return ret;
66      }
67
68      //    PUT -> toogleFoodItemAvailablity ->"/enableOrDisable"(FoodItem, toggleValue) -> change the availability of the food
69      @PutMapping("/food/toogleFoodItem")
70      public int toogleFoodItem(@RequestBody Food food) {
71          logger.info("----------------> food " + food);
72          int ret = foodDao.toogleFoodItem(food);
73          return ret;
74      }
75  }
76
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\controller\UserController.java

```java
package com.slbootcamp.foodbox.controller;

import com.slbootcamp.foodbox.entity.User;
import com.slbootcamp.foodbox.jdbc.UserDao;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
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.RequestBody;
import org.springframework.web.bind.annotation.RestController;

@CrossOrigin(origins = "*", allowedHeaders = "*")
@RestController
public class UserController {
    private Logger logger = LoggerFactory.getLogger(this.getClass());

    @Autowired
    UserDao userDao;

    //    POST -> "/login/
    @PostMapping("/login")
    public User login(@RequestBody User user) {
        return userDao.login(user);
    }
}

```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\java\com\slbootcamp\foodbox\controller\OrdenController.java

```java
package com.slbootcamp.foodbox.controller;

import com.slbootcamp.foodbox.entity.DisplayOrden;
import com.slbootcamp.foodbox.entity.Orden;
import com.slbootcamp.foodbox.entity.User;
import com.slbootcamp.foodbox.jdbc.OrdenDao;
import com.slbootcamp.foodbox.jdbc.UserDao;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@CrossOrigin(origins ="*", allowedHeaders = "*")
@RestController
public class OrdenController {

    private Logger logger = LoggerFactory.getLogger(this.getClass());

    @Autowired
    OrdenDao ordenDao;

    @PostMapping("/cart/postPurchase")
    public int placeOrden(@RequestBody Orden orden) {
//    public int placeOrden(@ModelAttribute Orden orden) {
//        logger.info("---------------> /cart/postPurchase: ");
      logger.info("----- --- ---- ---> Orden: " + orden);


        return ordenDao.saveOrden(orden);
//        int ret =  ordenDao.placeOrden(orden);
//        logger.info(" ret --------> " + ret);
//        return ret;
    }

    @GetMapping("/cart/getDisplayOrden/{id}")
    public List<DisplayOrden> getDisplayOrden(@PathVariable("id") int id) {
        logger.info("---------------> getDisplayOrden: " + id);
        return ordenDao.getDisplayOrden(id);
    }
}

```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\resources\data.sql

```sql


INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(5, 'ITALIAN', 'Delicious rice with fried shimp to perfection','http://tinyurl.com/shrimpFriedRiceImage','Y', 'Shrimp Fried Rice', 50);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(2, 'MEXICAN', 'Borrego asado en horno de hoyo ','http://tinyurl.com/barbacoaImage','Y', 'Tacos de Barbacoa', 20);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(3, 'MEXICAN', 'Receta de la abuelita de mole verde', 'http://tinyurl.com/moleVerdeImage','Y', 'Exquisite Mexican Mole Verde', 30);


INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(13, 'ASIAN', 'Muu haeng is thinly sliced pork, typically shoulder, while siin haeng is thinly sliced beef, typically a tough cut with fat, like top round.','https://tinyurl.com/haengImage','Y', 'Sun-Dried Beef', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(24, 'ASIAN', 'The meat is mixed with lemongrass, galangal, ginger, and garlic, then stuffed into natural casings.','https://tinyurl.com/herbalPorkImage','Y', 'Sai Oua (Herbal Pork Sausage)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(34, 'ASIAN', 'The mango flesh, bright yellow even before ripening, stays crunchy and provides a fresh sour note absent in green papaya.',        'https://tinyurl.com/mangoSaladImage','Y', 'Tam Muk Muang (Green-Mango Salad)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(28, 'ASIAN', 'Naem khao is a mixture of salt-cured ground pork, pig skin, steamed and dried white rice, and dried shredded coconut',        'https://tinyurl.com/coconutRiceImage','Y', 'Naem Khao (Crispy Coconut Rice)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(43, 'ASIAN', 'Salad made from ground meat and herbs, laced with fish sauce and lime juice and topped with a powder made from dry ground rice','https://tinyurl.com/duckSaladImage','Y', 'Laab Ped (Minced Duck Salad)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(14, 'ASIAN', 'Served with chicken thigh and leg meat and garnished with green onion, dried chile, Thai basil, lemongrass, and hon shimeji mushrooms.','https://tinyurl.com/AsianChickenSoupImage','Y', 'Gaeng Som (Chicken Soup With Fish Sauce and Tamarind)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(33, 'ASIAN', 'Chicken-broth soup with a red curry base that includes coconut milk','https://tinyurl.com/noodleSoupImage','Y', 'Khao Poon(Soup With Fermented Noodles)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(42, 'ASIAN', 'Charred in a wok with dark soy, ginger, garlic, onion, bell pepper, and mushrooms','https://tinyurl.com/SourPorkImage','Y',        'Muu Som (Rice-Fermented Sour Pork)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(6, 'ASIAN', 'Catfish fillets are mixed with rice flour and fried until crispy','https://tinyurl.com/crispyCatfishImage','Y', 'Paa Tod(Crispy catfish)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(12, 'ASIAN', 'Butter chicken is like pizza, hot dogs, and grilled cheese sandwiches: Even when it’s bad, it’s good.','https://tinyurl.com/butterChickenImage','Y', 'Butter Chicken', 10);

INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(23, 'AFRICAN', 'The sauce in this dish gets its kick from berbere, an Ethiopian chile powder fragrant with cardamom, fenugreek, and clove',        'https://tinyurl.com/BefTibsImage','Y', 'Beef Tibs ', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(7, 'AFRICAN', 'Mandazi are coconut doughnuts that are traditionally served alongside bharazi, or pigeon peas stewed in coconut
```

```
cream.',        'https://tinyurl.com/coconutDoughnutesImage','Y', 'Mandazi (East African Coconut Doughnuts)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(16, 'AFRICAN', 'Carrot soup with harissa and ginger.','https://tinyurl.com/spicySoupImage','Y', 'Spicy Carrot and Ginger Soup With Harissa',       10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(29, 'AFRICAN', 'Lamb meatball tagine, made with tender, big balls of ground lamb, flavored with carrots, shallots, tomatoes, golden raisins, cilantro, mint','https://tinyurl.com/LambMeatballmage','Y', 'Moroccan Lamb Meatball Tagine', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(41, 'AFRICAN', 'Well-spiced nuggets of beef or lamb are briefly fried in olive oil before being simmered in a simple, fragrant sauce of broth, onions, and tomato paste','https://tinyurl.com/tunisianMeatballsImage','Y', 'Tunisian Meatballs ', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(26, 'AFRICAN', 'Exciting contrast between sweet and spicy, harissa infused salmon with garlic and coriander',       'https://tinyurl.com/harissaSalmonImage','Y', 'Harissa-Honey Glazed Roasted Salmon', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(11, 'AFRICAN', 'Panko-dusted scallops, accented with lemon zest and cilantro. ','https://tinyurl.com/bakedScallopsImage','Y', 'Moroccan Baked Scallops', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(27, 'AFRICAN', 'Vegetarian shakshuka is a one-pan egg and tomato dish perfect for breakfast, lunch, or dinner.','https://tinyurl.com/poachedEggsImage','Y', 'Shakshuka (North African Style Poached Eggs in Spicy Tomato Sauce) ', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(15, 'AFRICAN', 'The soup is accented by a sautéed onion, a bit of ground cumin, minced garlic, and a few squeezes of lemon juice. ',            'https://tinyurl.com/chickpeaSpupImage','Y', 'Lablabi (Tunisian Chickpea Soup)', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(18, 'AFRICAN', 'Individual grains of rice that are cooked through but not mushy interspersed with tender, well-seasoned vegetables that hold their shape.','https://tinyurl.com/nigerianRiceImage','Y', 'Nigerian Fried Rice', 10);

INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(53, 'ITALIAN', 'Its bright flavors come from garlic, basil, and cherry tomatoes, which we sauté until they burst.','https://tinyurl.com/cherryTomatoImage','Y', 'Pasta With Blistered Cherry Tomato Sauce', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(35, 'ITALIAN', 'Dressed up with browned butter, fresh sage, and tender chunks of butternut squash','https://tinyurl.com/SquashPastaImage',         'Y', 'Pasta With Butternut Squash and Sage Brown Butter', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(44, 'ITALIAN', 'Meat-free, egg-free, dairy-free, and still absolutely delicious.','https://tinyurl.com/carbonaraPastaImage','Y', 'Vegan Carbonara Pasta', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(8, 'ITALIAN', 'Arrabbiata is Italian for "angry," and embrace that fieriness by adding lots of red pepper flakes.','https://tinyurl.com/arrabbiataSauceImage','Y', 'Penne With Hot-As-You-Dare Arrabbiata Sauce', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(40, 'ITALIAN', 'Traditional Genovese preparation of pasta with pesto, green beans.','https://tinyurl.com/pestoPastaImage','Y', 'Pesto Pasta With Potatoes and Green Beans', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(45, 'ITALIAN', 'We incorporate just a little cream, turning instead to pasta water, cornstarch, and an egg to achieve a brighter, but still thick, sauce.','https://tinyurl.com/fettuccineAlfredoImage','Y', 'Lighter Fettuccine Alfredo', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(30, 'ITALIAN', 'This simple vegetarian dish of whole wheat fusilli uses both leaves and stems of Swiss chard.','https://tinyurl.com/citrusPastaImage','Y', 'Vegetarian Citrus Pasta With Swiss Chard', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(36, 'ITALIAN', 'Celebrate the coming of warmer days by pairing fresh asparagus with heavy cream, Parmigiano Reggiano, crispy
```

```
prosciutto, and bright lemon. ','https://tinyurl.com/gnocchiProscuiuttoImage','Y', 'Ricotta Gnocchi With Asparagus and Prosciutto',
10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(10, 'ITALIAN', 'Browned mushrooms; white wine, shrimp and spinach. ',       'https://tinyurl.com/orecchietteShrimpImage','Y', '
One-Skillet Orecchiette With Shrimp, Spinach, and Mushrooms', 10);

INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(39, 'CARIBEAN', 'Our version of ropa vieja (Cuban shredded stewed beef) improves efficiency and flavor.','https://tinyurl.com/
ropaViejaImage','Y', 'Ropa Vieja Recipe', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(1, 'CARIBEAN', 'Three meats are pressed with cheese, pickles, and mustard until toasty and warm.','https://tinyurl.com/
cubanSandwichImage','Y', 'Cuban Sandwiches', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(46, 'CARIBEAN', 'Classic Cuban sides for all kinds of meals.','https://tinyurl.com/blackBeansImage','Y', 'Cuban Black Beans and
 Rice Recipe',       10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(17, 'CARIBEAN', 'Tender stewed chicken in a rich and silky gravy.','https://tinyurl.com/stewChickenImage','Y', 'Jamaican Brown
 Stew Chicken', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(50, 'CARIBEAN', 'Marinated chicken, smoked low and slow on the grill','https://tinyurl.com/SpicyGrilledChickenImage','Y'
,       'Spicy Grilled Jerk Chicken', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(9, 'CARIBEAN', 'Spicy Jamaican chicken curry.','https://tinyurl.com/chickenCurryImage','Y', 'Jamaican Chicken Curry Recipe ',
10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(31, 'CARIBEAN', 'When you combine tacos and Jamaican beef patties, something magical happens.','https://tinyurl.com/
jamaicanBeefTacosImage',       'Y', 'Jamaican Beef Tacos With Tropical Slaw', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(51, 'CARIBEAN', 'Jerk chicken has it all: sweetness, spice and come-hither appeal.','https://tinyurl.com/chickenRummyImage','Y
', 'Hellfire Jerk Chicken With Rummy Grilled Pineapple', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(20, 'CARIBEAN', 'Curry-laced dough is filled with a heady, spicy beef mixture and baked in the oven until golden brown.','https
://tinyurl.com/jamaicanPattiesImage','Y', 'Curried Jamaican Beef Patties', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(47, 'CARIBEAN', 'Onions, peppers, garlic, and fresh herbs are combined in the food processor for a powerfully flavorful Puerto
Rican sofrito.','https://tinyurl.com/SofritoImage','Y', 'Puerto Rican Sofrito From Scratch', 10);

INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(22, 'MEXICAN', 'Beef chuck and also oxtails, and then layers in even more flavor with multiple chilies and a splash of fish
sauce.',              'https://tinyurl.com/beefiestBarbacoaImage','Y', 'Beefiest Barbacoa', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(48, 'MEXICAN', 'Traditional carnitas are made by slowly cooking pork in a pool of lard.','https://tinyurl.com/carnitasImage','Y
', 'Killer Carnitas', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(38, 'MEXICAN', 'Mexican-style chile verde made with Hatch chilies','https://tinyurl.com/chileVerdeImage','Y', 'Chile Verde
Virtuosity', 10);
INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
values(52, 'MEXICAN', 'Made with pork shoulder and a handful of surprise characters, like raisins, orange juice from concentrate, and
fish sauce.',
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\resources\data.sql

```
102         'https://tinyurl.com/carneAdobadaImage','Y', 'Carne Adovada', 10);
103 INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
104 values(49, 'MEXICAN', 'A combination of poblano, cubanelle and jalapeno peppers.','https://tinyurl.com/chileVerde2Image','Y', 'Chile
    Verde With Pork', 10);
105 INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
106 values(19, 'MEXICAN', 'The best way to make tacos for a crowd is to make tacos de canasta.','https://tinyurl.com/tacosCanastaImage','Y
    ', 'Tacos de Canasta (Basket Tacos for a Party or Potluck)', 10);
107 INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
108 values(55, 'MEXICAN', 'Chorizo Tacos combined with garlic, cumin, some warm spices, oregano','https://tinyurl.com/chorizoTacosImage'
    ,       'Y', 'Mexican Chorizo Tacos', 10);
109 INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
110 values(37, 'MEXICAN', 'Slow braised pork, shredded and then crisped up before serving.','https://tinyurl.com/shreddedPorkImage','Y',
    'Shredded Pork in Ancho-Orange Sauce', 10);
111 INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
112 values(21, 'MEXICAN', 'A blend of chiles and spices provide this classic Texas chili con carne with maximum depth of flavor.','https
    ://tinyurl.com/chileCarneImage','Y', 'Original Texas Chili Con Carne', 10);
113 INSERT INTO food (id, cuisine, description, image_url, is_enabled, food_name, price)
114 values(4, 'MEXICAN', 'Grilled skirt steak, well salted, and deeply browned on the edges, with beefy brawn shining through underneath'
    ,       'https://tinyurl.com/carneAsadaTortaImage','Y', 'Carne Asada Torta', 10);
115
116
117 insert into user(id,contact,email,password, profile, username, credit, name) values (1,"(214) 000-0001","hungrymike@gmail.com", "
    chicago2022", "C",  "mike", "", "Mike Rice");
118 insert into user(id,contact,email,password, profile, username, credit, name) values (2,"(214) 000-0002","admin@foodbox.com", "chicago2022
    ","A", "admin", "", "Jazmine the Admin");
119
120
121  insert into box (id, food_id, quantity)values(1,1,1);
122  insert into box (id, food_id, quantity)values(1,2,1);
123  insert into box (id, food_id, quantity)values(1,3,1);
124  insert into box (id, food_id, quantity)values(1,4,1);
125
126  insert into box (id, food_id, quantity)values(2,1,1);
127  insert into box (id, food_id, quantity)values(2,2,1);
128
129
130  insert into orden(id,status,box_id,user_id)values(1,"READY", 1,1);
131  insert into orden(id,status,box_id,user_id)values(2,"PENDING", 2,1);
132 insert into orden(id,status,box_id,user_id)values(3,"PENDING", 2,3);
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\resources\schema.sql

```sql
drop table if exists orden;
drop table if exists user;
drop table if exists box;
drop table if exists food;

create table food
(
    id          integer not null auto_increment,
    cuisine     varchar(255),
    description varchar(255),
    food_name   varchar(255),
    image_url   varchar(500),
    is_enabled  varchar(255),
    price       bigint,
    primary key (id)
);


create table user
(
    id       bigint not null auto_increment,
    contact  varchar(255),
    email    varchar(255),
    password varchar(255),
    name varchar(255),
    username varchar(255),
    profile varchar(255),
    credit varchar(255),
    primary key (id)
);


create table box
(
    id      bigint not null,
    food_id integer,
    quantity integer,
    primary key (id, food_id)
);



create table orden
(
    id      bigint not null auto_increment,
    box_id  bigint not null,
    user_id bigint not null,
    status  varchar(255),
    primary key (id)
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\resources\schema.sql

```
);
--
--
-- alter table box
--     add constraint UK_rb6npe7305gao860lxnuj2g7t unique (box_id);
--
--
-- alter table box
--     add constraint FKry8rla6994n8hd9579ia6a7k8
--         foreign key (food_id)
--             references food (id);
--
-- alter table orden
--     add constraint UK_rb6npe7305gao860lxnuj2g7t unique (box_id);
--
-- alter table orden
--     add constraint FKev4ou7yydtyartj534k0v82gx
--         foreign key (box_id)
--             references box (id);
--
--
-- alter table orden
--     add constraint FKdkqdxtdujrqslotkp947lgvqy
--         foreign key (client_id)
--             references client (id);
```

File - C:\Users\giann\Documents\CODE\sl-capstone-foodbox\src\main\resources\application.properties

```
1
2  spring.sql.init.mode=always
3  spring.sql.init.platform=mysql
4
5
6  spring.datasource.url=jdbc:mysql://foodboxdbinstance.c8danfwnugys.us-west-2.rds.amazonaws.com:3306/foodboxdb
7  spring.datasource.username=root
8  spring.datasource.password=fbitga01
9  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
10
11 spring.jpa.hibernate.ddl-auto=none
12 spring.jpa.database-platform=org.hibernate.dialect.MySQL5InnoDBDialect
13
14 server.port=5000
```