Restaurant Manager Assistant

Security Manual

**November 27, 2022**

**Slippery Rock University**

Bradley Smith **bms1040@sru.edu**

Ethan Pasman **eap1008@sru.edu**

# Overview

This document describes the security of the Restaurant Manager Assistant system. This will be an extensive look at what can go wrong and what we have done to prevent or solve these flaws.

# 2. Navigation

* To assist in navigating this document, Headers have been made for each major topic of the document.
  1. Ex. ‘Login Security’ header for the section about the security around the login feature.

# 3. Login Security

## The main security of the system revolves around user accounts. These accounts are the objects given roles, connecting to the database, and allowed access to a number of pages corresponding to their authorities.

# User Registration

## When not logged in, guest users are able to register a new account from a button on the home page. This button takes them to a form which when filled out, will add a new entry in our Customers repository.

* All fields in the form are required to be filled out except for the checkbox corresponding to signing up for customer rewards. Attempting to register with any of the text fields left blank will show a message saying “Please fill out this field” instead of continuing and registering a customer with any null attributes.
* The “Email” field is required to have an @ symbol with characters before and after the symbol, however it does not check with any DNS server to ensure the email entered is valid.

# Login Page

## Upon pressing the “Sign-In” element under the “Account” drop down menu, the user is redirected to a login page. This login page is the default login page generated by Spring Security. Details of this default login page can be found at the following link:

## <https://docs.spring.io/spring-security/site/docs/current/api/org/springframework/security/config/annotation/web/configurers/FormLoginConfigurer.html>

* The way this checks the credentials of a user is through our implementation of FakeApplicationUserDaoService. We have a list of ApplicationUsers with information from each repository belonging to customers, servers, managers, and administrators, as well as two hardcoded accounts for HQ manager and HQ administrator roles.
* This list of ApplicationUsers is used to find a user where getUsername() will be equal to the username parameter entered in the login page, and then check that account to see if the encoded password is the same as an encoded version of the password parameter entered in the login page.
* The PasswordEncoder we used is a BCryptPasswordEncoder with a strength value of 10.
* Upon authentication through a successful login, RestaurantController.authorityCheckForLoginRedirects() is called, which uses the current SecurityContext to get the user’s list of authorities, and redirects to the home page for the highest role the user has.

# Logging Out

## Upon pressing the “Sign-out” element under the “Account” drop down menu if the user is logged in, the user will be redirected to a confirmation page, asking “Are you sure you want to log out?” and a “Log Out” button.

* This logout confirmation page is the default logout page generated by Spring Security. Details of this default page can be found at the following link:
* <https://docs.spring.io/spring-security/site/docs/current/api/org/springframework/security/config/annotation/web/builders/HttpSecurity.html#logout()>
* Pressing this confirmation button will redirect the user back to the login page after clearing the authentication and invalidating the HttpSession associated with the user.

# 4. Roles

## User accounts are given roles based on the highest staff position. These roles are used for giving authorities and granting access to specific pages within the system. The use of roles within the system is done to ensure security and keep access to only people who need access. Attempting to visit a page the user does not have the role for will automatically redirect them to the login page.

# Guest

## A guest user that is not logged into any account has no roles or authorities, and can only visit the few pages accessible to everyone.

* Guests will be able to view the local menu, create a cart and place orders through said cart, view the contact page, and register a new account through the registration form.

# Customer

## A customer account is any user that has registered and can log into the system. They are able to do everything that a guest user can do along with a few more pages.

* Customers can view all of their account information, including order history and rewards, and edit any of their attributes except for their rewards status and order history.

# Server

## A restaurant server account is the lowest staff role within the employee hierarchy. Other than all authorities that a customer can do, its home page is the server home page, showing outgoing orders and the current menu including unavailable items.

* Servers can also view the basic customer information of anyone who has an outgoing order, as well as mark any order as completed.

# Manager

## A restaurant manager account can view the local restaurant’s action log, and see all the major changes made and logged within the location. Restaurant managers also have all of a server’s authorities, and can view the local server pages.

* Along with this, managers can view a list of all local customers with their location set to the same restaurant. They can edit or delete basic customer information for any customer. They can do the same with a list of all local servers, along with edit and delete pages for server information.
* Managers can view the local inventory for the restaurant, edit any inventory stock, view all current shipments for a restaurant, and request a new shipment.

# Warehouse Manager

## A warehouse manager account can view the local warehouse’s action log, and see all the major changes made and logged within the location.

* Warehouse managers can view a list of all local warehouse employees, and edit their basic information. They can also view and edit local inventory for the warehouse.
* Along with this, warehouse managers can view a list of shipments and either accept or deny incoming shipments, changing their status to either “In Progress” or “Denied”.

# Admin

## An admin account can view the local restaurant’s action log, and see all the major changes made and logged within the location. Admins also have all authorities of a server and manager, and can access their pages.

* Along with this, admins can view a list of all local customers. They can edit or delete all customer information for any customer. Unlike managers, admins can also add new customers. They can do the same with a list of all local managers, servers, and warehouse employees, along with add, edit and delete pages for information about each account.

# HQ Manager

## An HQ manager account can view the full action log for all locations, and see major changes made from any location. HQ managers also have all authorities of a server and manager, and can access their pages.

* HQ managers can view full lists of restaurant managers and warehouse managers, and can add new restaurant or warehouse manager accounts, edit existing account information, and delete a manager account.
* Along with this, HQ managers can view a list of all offices, restaurants, and warehouses and see their information.

# HQ Admin

## An HQ admin account can view the full action log for all locations, and see major changes made from any location. HQ admin acts as a superuser and can access all pages.

* HQ admins can view full lists of administrators, offices, restaurants, and warehouses, and can add new information for each, edit existing information, and delete information from the database.

# 5. Database security

## All information saved in the system is stored in a database.

# Current Database Implementation

## Our database repositories are all stored within separate tables in plaintext.

* All entities have a specific ID associated with them.

# 6. Future Plans

## If this project is expanded upon by others in the future, some security features can and should be improved within the system. The following will be a plan to improve the system’s current security features.

# Encrypting Database Connections

## Currently, all entities in our databases are stored in plaintext, including sensitive information such as emails, locations, and passwords.

* All information should be encrypted to protect user information.

# Restrictions for User Registration

## Currently, all users are able to register without any restrictions for valid email accounts or secure passwords.

* Restrictions should be added for both email and password fields, with email being checked for validity, and password needing restrictions on character count, and requiring it to be alphanumeric with special characters.
* Registration should fail if a user with the same email exists within the database.

# Create Custom Implementation of Login and Logout Pages

## Currently, this system uses the default login and logout pages generated by Spring Security. This means that people outside the system can tell what framework the website security is created with.

* The login and logout pages should be changed from the default pages to obscure how our security works.