|  |
| --- |
| Regis University MSSE 682 Enterprise C# Software Development |
| Order Entry Application |
| Design Document |

|  |
| --- |
| Stephen Schwenck  3-9-2014 |

Table of Contents

[Revision Table 2](#_Toc382168492)

[Overview 3](#_Toc382168493)

[Use Case Diagram 3](#_Toc382168494)

[Use Cases 4](#_Toc382168495)

[Fully Dressed Use Case – Create Order 4](#_Toc382168496)

[Casual Use Case – Add Product 5](#_Toc382168497)

[Brief Use Case – Accept Tender 5](#_Toc382168498)

[References 6](#_Toc382168499)

|  |  |  |
| --- | --- | --- |
| **Revision** | **Description** | **Date** |
| 1 | Initial project commit/setup | 03/09/14 |
| 2 | Added .sln file | 03/14/14 |
| 3 | Adding .Test project | 03/14/14 |
| 4 | Forgot to add .Debug and .Release config files | 03/14/14 |
| 5 | Adding unintentionally excluded files | 03/14/14 |
| 6 | Removing useless tests | 03/14/14 |
| 7 | UI changes | 03/16/14 |
| 8 | Week 2 assignment commit | 03/17/14 |
| 9 | DesignDoc wasn’t included | 03/17/14 |
| 10 | Updating models | 03/23/14 |
| 11 | Design doc update | 03/23/14 |
| 12 | Duplicate model entries | 03/23/14 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

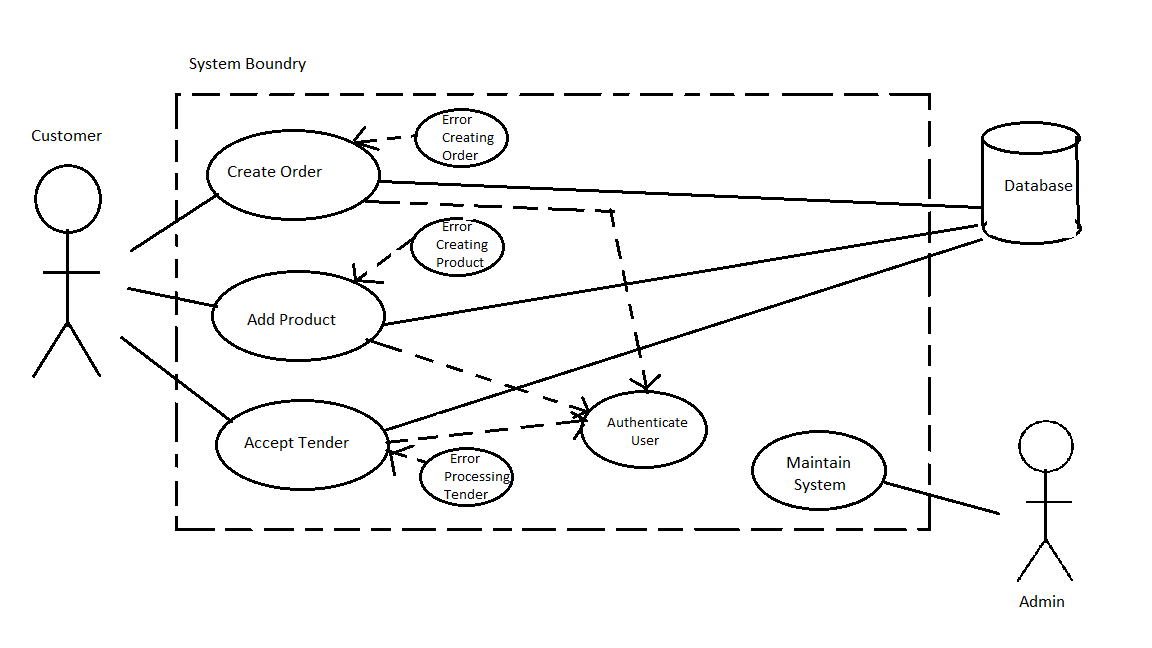
# Revision Table

# Overview

Order Entry will be an easy to use application that allows users to enter orders from a web application. The need for this application coms from real life demand for an application that allows user to do simple order entry without entering into an enterprise scale application, which requires a stable and authenticated connection and a very large footprint. By creating an application that aims to satisfy only order entry and not everything else involved in an enterprise distribution solution, we can reduce the footprint of the application and hopefully make it much more accessible for users.

The application will be based on enterprise level order entry, which on its own would be too large to implement for this course. Instead, this app will focus on allowing users to enter regular stock or counter-sale orders, add stock products, and accept tendering. These three main use cases will build the basis for my Order Entry app. Additional functionality may include reviewing and editing taxing information, and order addons and possibly editing created orders.

# Use Case Diagram



# Use Cases

* Create Order
* Add Product
* Accept Tender
* Product Validation
* Error Creating Order
* Error Adding/Validating Product
* Error Processing Tender
* Validate User
* Login

## Fully Dressed Use Case – Create Order

Primary Actor – Order Entry App User

Stakeholders and Interests

* User: wants fast, easy way to enter orders
* Company: wants reliable way to allow users to enter orders without using enterprise app

Preconditions – User has logged in and been authenticated

Success Guarantee – an order has been created and the user is put into “Add Products” mode

Main Success Scenario

1. User logs in and is authenticated.
2. User is prompted with initial “Create Order” screen.
3. User enters required information for creating order (customer and warehouse).
4. The application validates that the customer/warehouse combination (and any other information entered) is valid.
5. The order is created and the user is put into the “Add Products” screen.

Extensions

Invalid Order Information

1. Steps 1-3 of Main Success Scenario.
2. Application validates order information and an error is returned and displayed to the user.
3. Fields in error are highlighted and the user is prompted to check entered data and try again.

Special Requirements

* Application must be accessible via the web.
* Must be able to get to “Add Products” screen in than 3 mouse clicks

Technology and Data Variation List –

Frequency of Occurrence – Potentially 1 every 2-3 minutes.

Open Issues – N/A

## Casual Use Case – Add Product

#### Main Success

The user is already authenticated and engaged in an active order. Once an order is created the user is automatically put into the “Add Products” section of the application. User enters a product number (or searches for one). Once a product is selected it is Validated and basic line information is brought back. User can then edit product information (quantity, price etc.) and submit the product to be added to the order. Product s re-validated and added to the order.

#### Alternate Scenarios

Product Cannot Be Found– The user enters an invalid product and attempts to validate. The user is presented with a warning message that the product entered is not valid or cannot be found and is prompted to check information and try again.

Product Not Validated – User enters invalid information for validated product and attempts to validate/add. One of the changed fields is not valid (i.e. quantity of 0) and the error is returned and presented to the user. The invalid field is highlighted and the user is prompted to correct data and attempt revalidation/addition of product.

## Brief Use Case – Accept Tender

The user is already authenticated and engaged in an active order. The user navigates to the “Payment” section and user is presented with options for payment type. User selects payment (from credit card, check and cash), and is asked to enter further payment information (if required) and amount. User submits payment information for authentication (if required) and after data is authenticated, data is saved to the payment record attached to the order.

# References

**There are no sources in the current document.**