Charles Ryan Barrett, Timothy Davis, Jonathon Gower

EKU CSC 440  December 5th, 2020

Tony’s Pizza app

Contents

[**I.** Introduction 2](#_Toc58100183)

[**a.** Problem Statement 2](#_Toc58100184)

[**b.** Proposal 2](#_Toc58100185)

[**II.** System Description 2](#_Toc58100186)

[**II.** System Requirements 2](#_Toc58100187)

[A. Functional Requirements 2](#_Toc58100188)

[B. Non-functional Requirements 10](#_Toc58100189)

[**III.** The system shall store and keep track of user information to make ordering quicker (Save address & payment information)Use Case Diagram 11](#_Toc58100190)

[**IV.** Class Diagram 12](#_Toc58100191)

[**V.** Sequence Diagrams 13](#_Toc58100192)

[**VI.** Activity Diagrams 15](#_Toc58100193)

[**VII.** State Diagram 19](#_Toc58100194)

[**VIII.** Database Design 19](#_Toc58100195)

[**IX.** Conclusion 21](#_Toc58100196)

[**X.** Data Dictionary 21](#_Toc58100197)

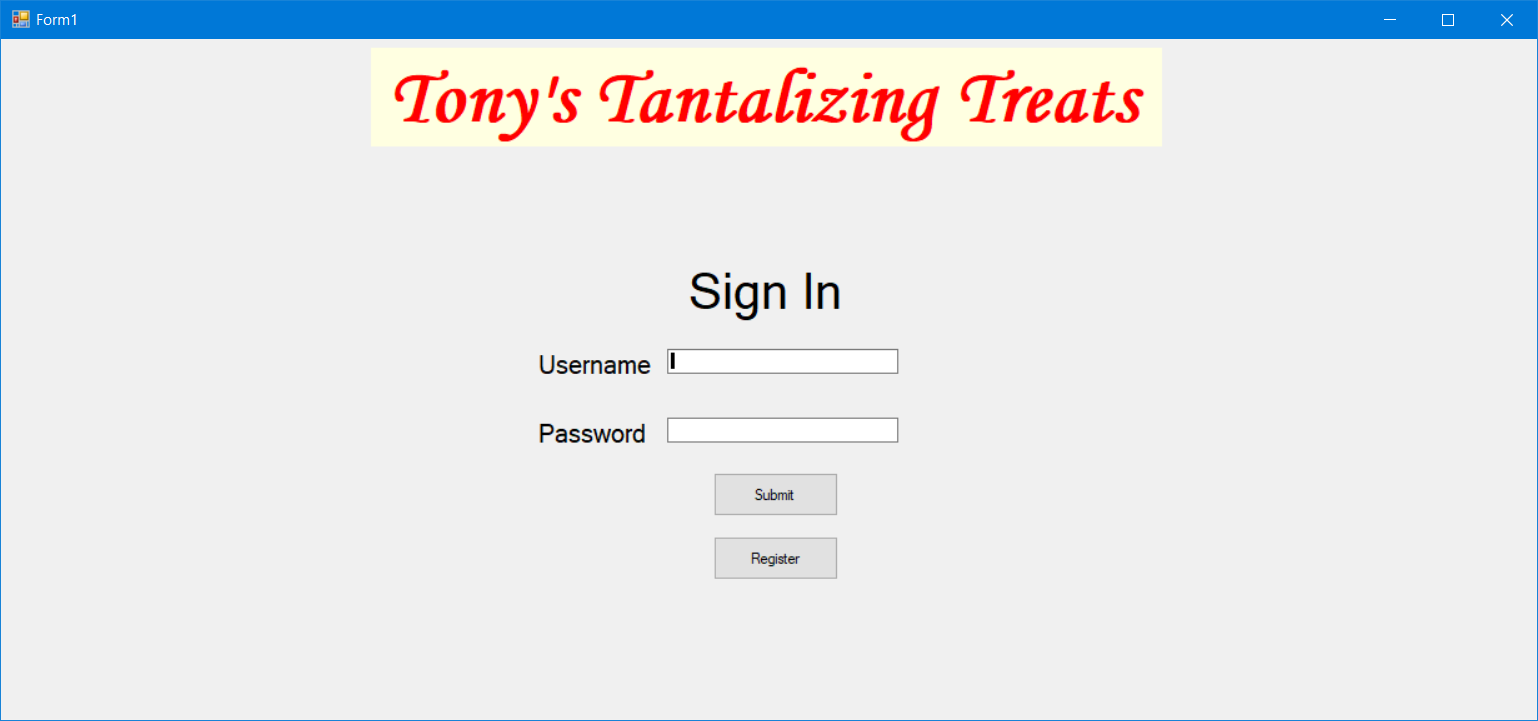
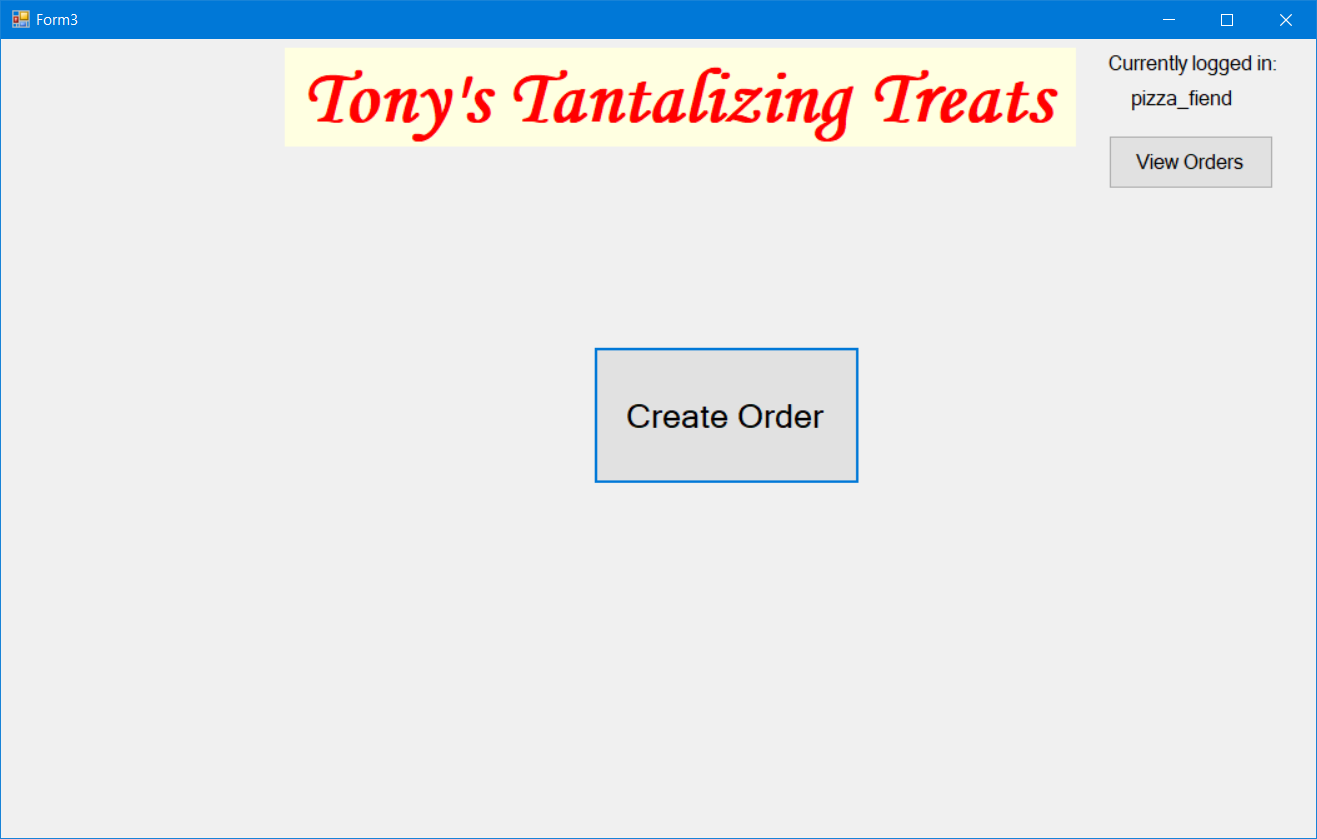
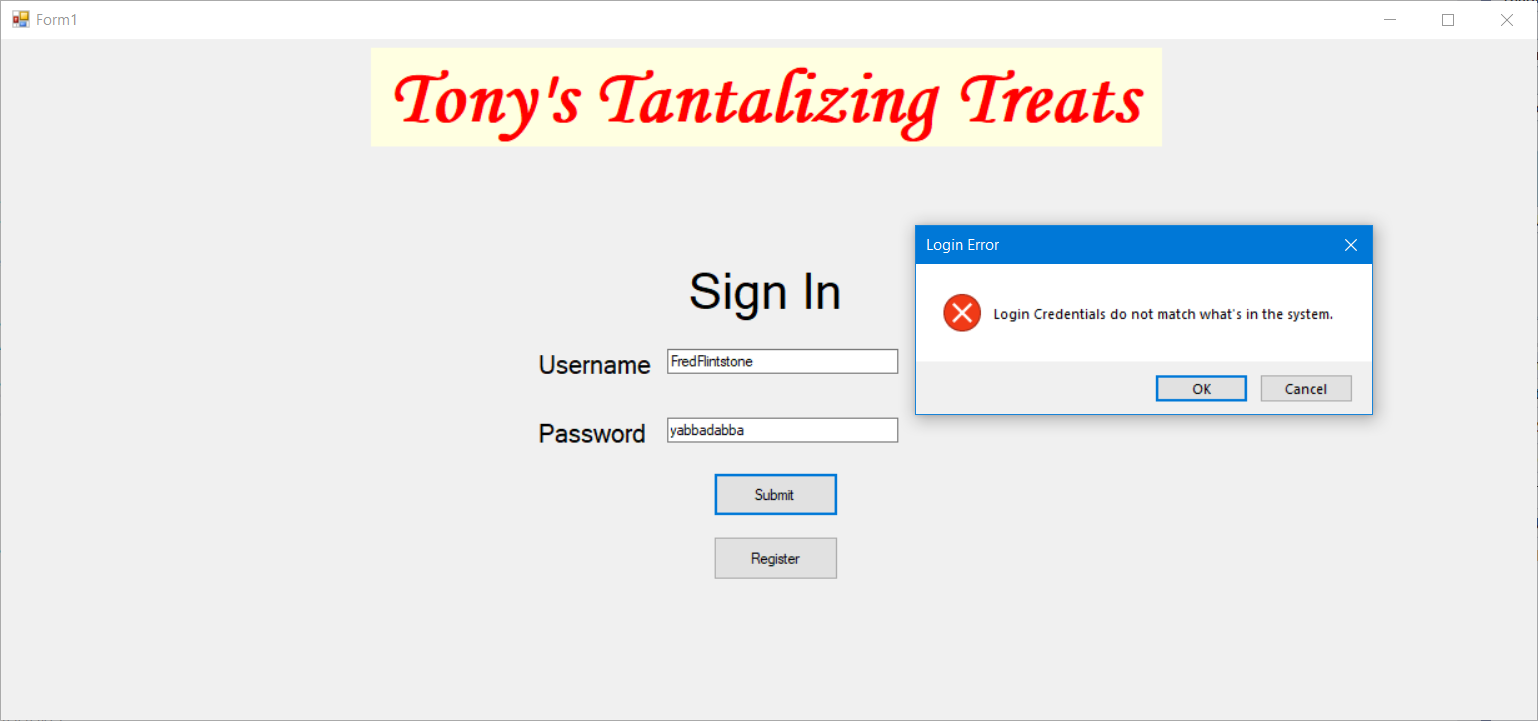
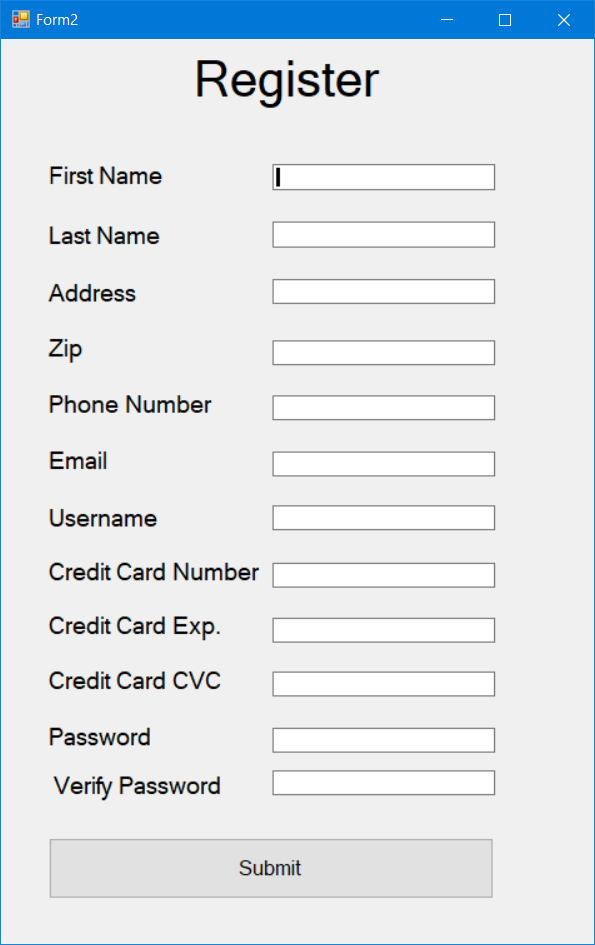
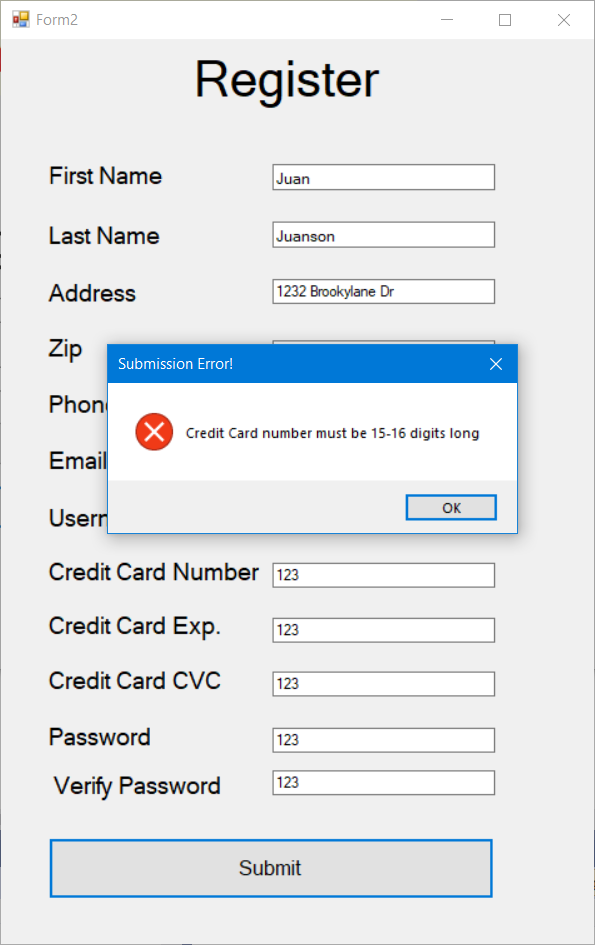
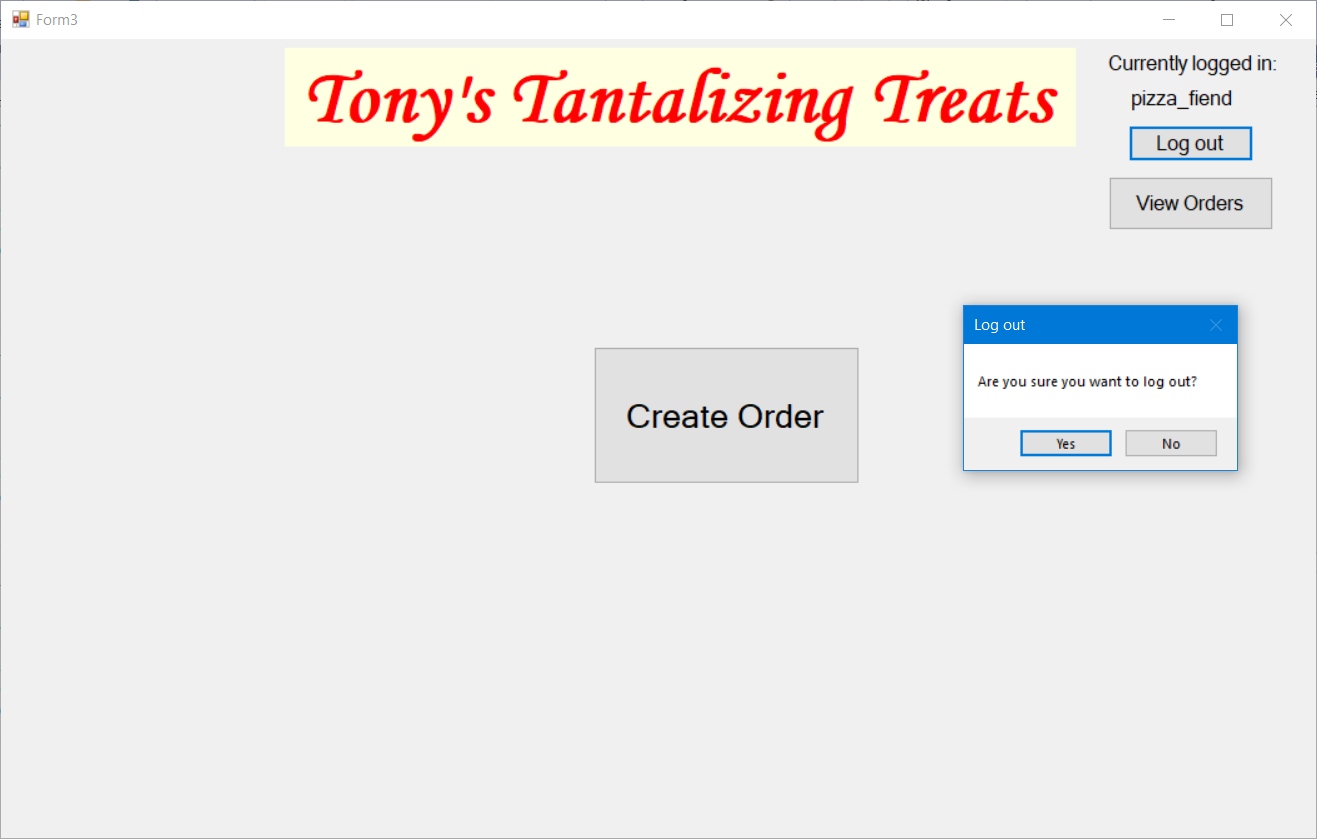
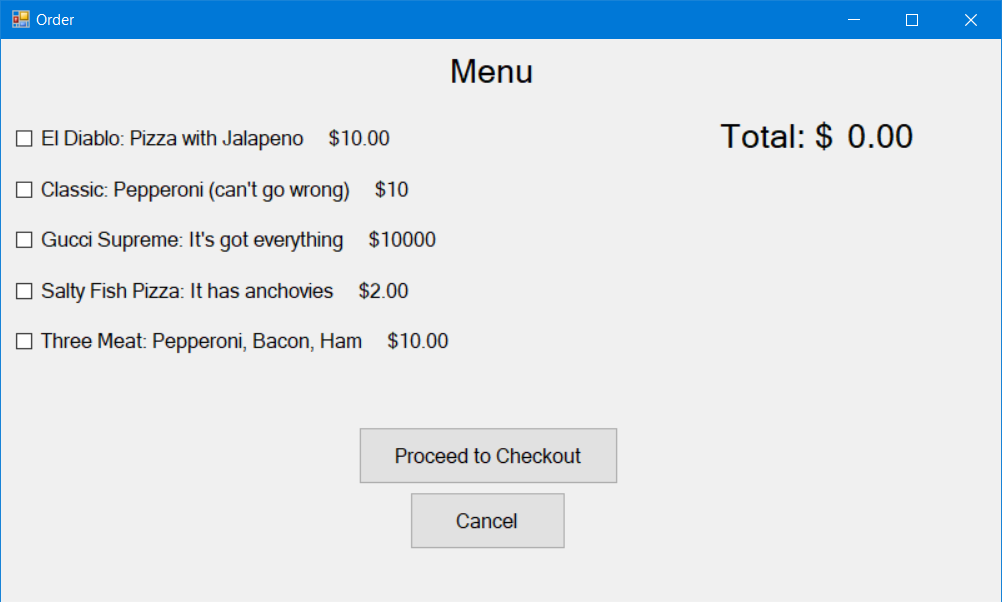
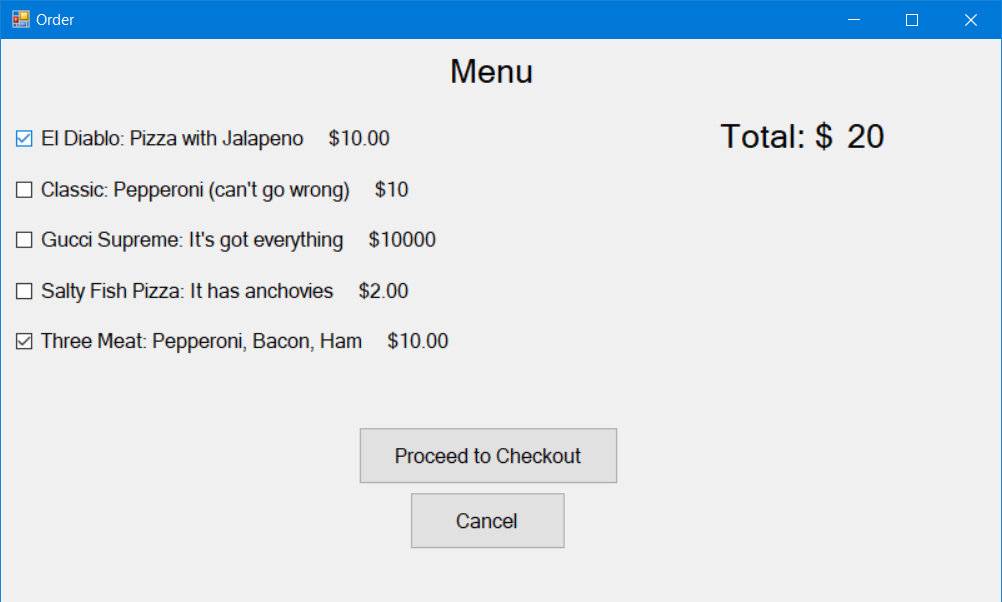
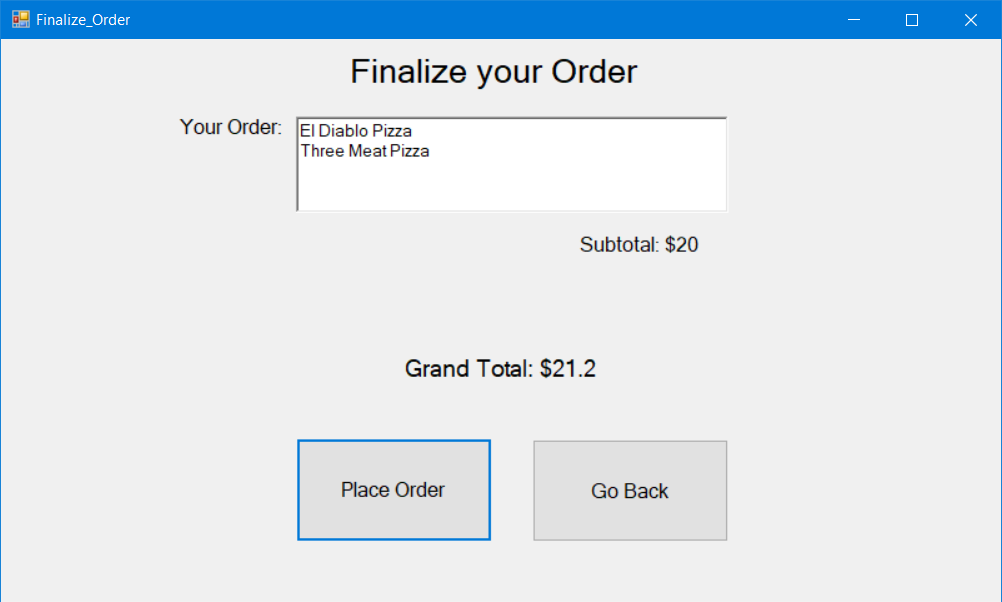
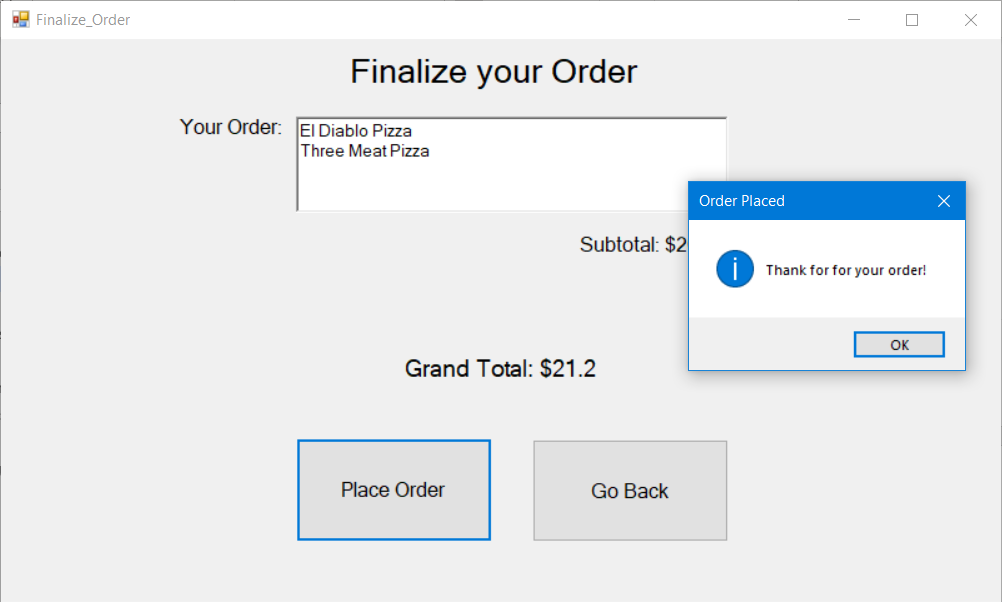
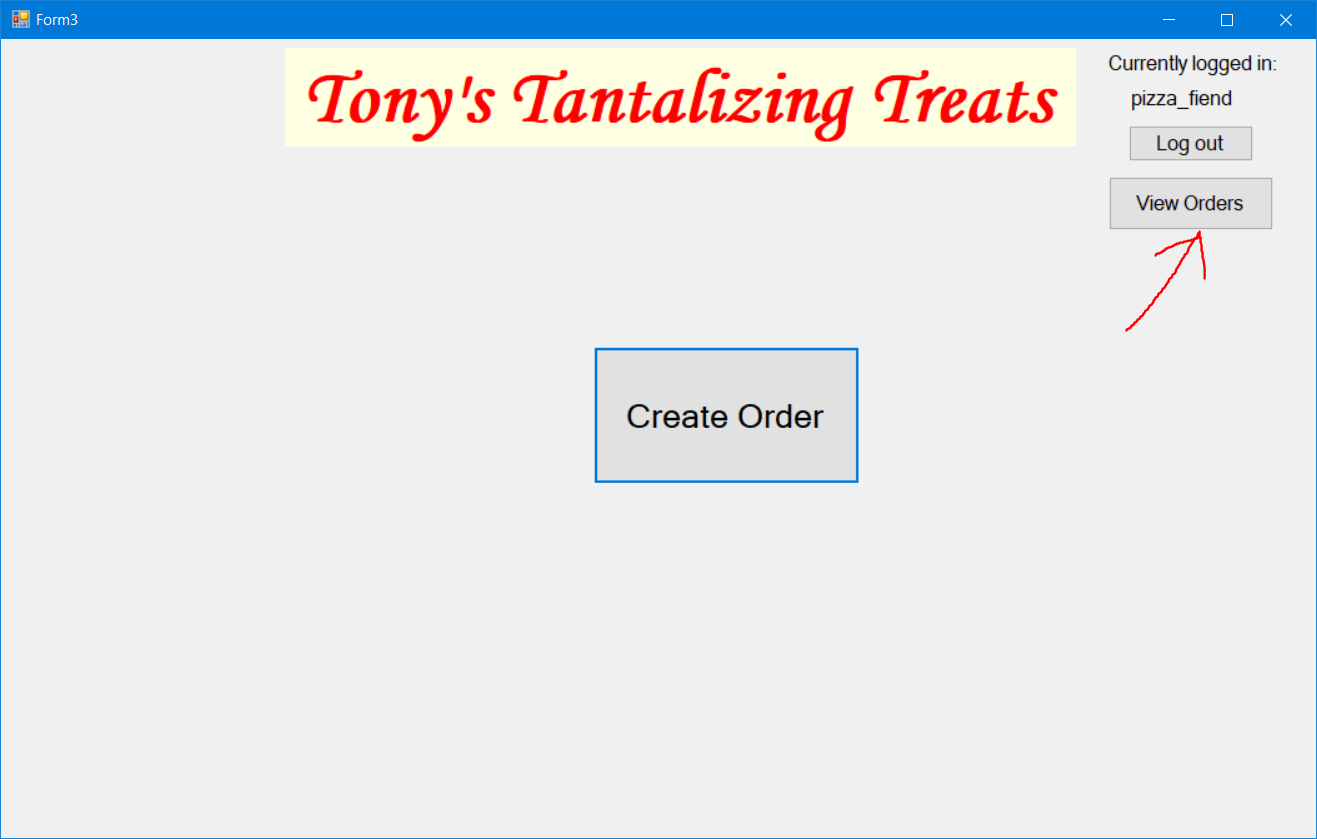
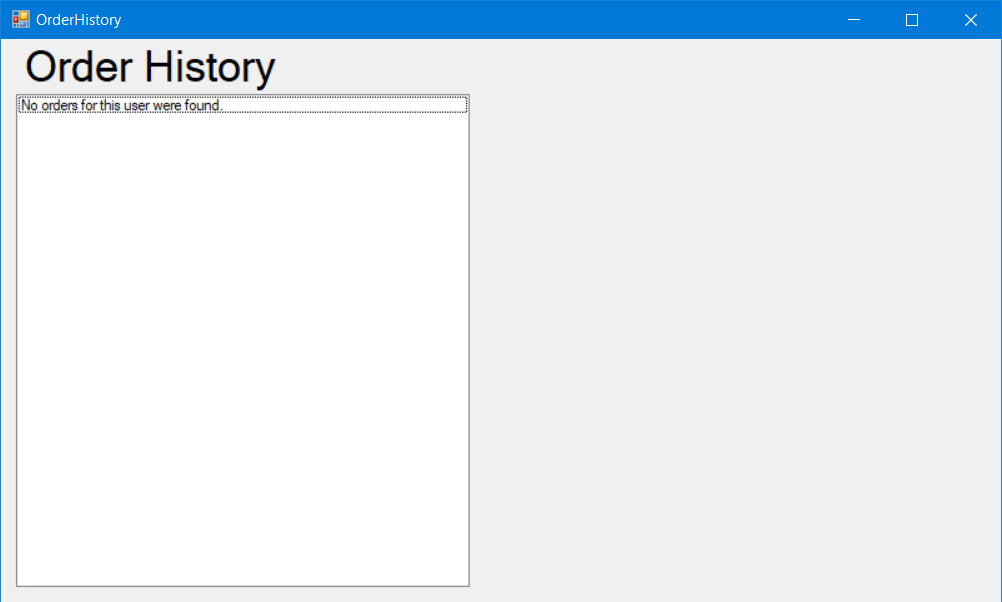
# Introduction

* 1. Problem StatementTony, owner of the rising pizzeria restaurant “Tony’s Tantalizing Treats”, needs an app for people to order from his restaurant. The app needs to be user friendly and straightforward. It also needs to maintain databases that holds user information to make ordering easy and menu items.
  2. ProposalWe can create the app that Tony needs.

1. System DescriptionThe program we make shall be both user friendly and easy to use. Along with efficiently using, reading from, and writing to, databases for menu items, orders, and user information. The system shall be quick in operation and accurate in end product. We shall implement it as a forms app in visual studio using the programming language C#. The app design shall be pleasing to look at and efficient in operation.

# System Requirements

## Functional Requirements

1. The System Shall allow a customer to log in
   1. The User shall start the program and come to the log in page  
      
   2. The user shall enter their information and press the “Submit” button
      1. If the User’s information matches an account in the database, the system shall bring the user to the main menu  
         
      2. If the User’s information was incorrect, the system shall display an error message  
         
2. The System shall allow the user to register an account
   1. The user shall press the “Register” button on the login page (R1 1.1)
   2. The System shall display a form for the user to enter their information  
      
   3. After the user has entered their information the user shall press the “Submit” button
      * 1. If there are no issues (invalid zip code, CC info, etc) the account will be written to the database and the user will be registered.
        2. If there was an issue the system shall display an error message  
           
3. The System shall allow a user to log out if they are already logged in
   1. The user shall press the “Log out” button on the main menu
      1. The system shall ask if they are sure  
         
         1. If the user presses “Yes” They are logged out and returned to the log in page
         2. If the user presses “No” they shall remain logged in
4. **The System Shall allow a user to create an Order**
   1. The user shall select the “Order” button from the main menu
   2. The System shall display all menu items, item descriptions, and prices  
      
   3. The User shall select the check mark next to the items they want
   4. The System shall keep a tally of the sub total for the current order  
      
   5. Once the user has finished making selections, they shall press the “proceed to checkout” button
   6. The System shall display a menu showing the grand total (including delivery and taxes) along with all the order items  
      
      1. The user may click the “Go Back” button to return to the menu and make any edits
5. The System Shall allow the user to place an order
   1. Once the user has created an order as they did in R4 the user shall press the “Place Order” button
      1. If the User is logged in the system shall write the order to the database and display a confirmation and thank you message  
         
      2. If the user is not currently logged in, they will either need to login (R1) or register for an account (R2)
6. The System shall allow a user to view their previous orders
   1. The user shall log in to their account (R1)
      1. If the user does not have an account, they may create one (R2)
   2. The user shall press the “View Orders” Button below the logout button  
      
   3. The system shall write a query to the database for all orders made by the current user
   4. The system shall display all the previous orders to the user  
      

## Non-functional Requirements

* 1. The system shall not take longer than half a second to read from or write to the database
  2. The system shall not allow incorrect payment information to be used
  3. The system shall not allow incorrect address (zip code < 5 digits) to be used
  4. The system shall only allow users to log in if they enter the correct information matching an existing account
  5. The layout shall be concise and easy to use and understand
  6. The system shall store and keep track of user information to make ordering quicker (Save address & payment information)

# Use Case Diagram



# Class Diagram



This diagram shows the overview of the system’s User and Order classes. Each class has a set of functions that are responsible for the class’ operations.

# Sequence Diagrams



Login – This diagram shows the interactions in the system that occur when a user enters their login credentials to the login page.



Logout – This diagram shows the interactions in the system that occur when a user selects to logout of the system



Register – This diagram shows the interactions that take place for a user gets registered for the system.



Check order – This diagram shows the interactions that take place for a user that checks his previous orders that have been placed



Create Order – This diagram shows the interactions that take place for a user to create an order.



Place Order – This diagram shows the interactions that take place for a user to finalize an order and submit it to the database

# Activity Diagrams



Place Order – This diagram shows the activity flow of the system when a user finalizes an order.



Login – This diagram shows the activity flow of the system when a user enters their credentials into the login page.



Logout - These are the activities when a user logs out of their account.



Register - This diagram shows the activity flow of the system when a user registers for a new account.



Add item - This diagram shows the activity flow of the system when a user adds items to an order.



Order History - This diagram shows the activity flow of the system when a user wants to look up a previous order they have made.

# State Diagram



# Database Design



User

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| userID | userName | email | password | Phone | CreditSecCode | Credit exp | CreditCardNum | address |

Menu Item

|  |  |  |  |
| --- | --- | --- | --- |
| ItemID | Description | ItemName | Price |

Order

|  |  |  |
| --- | --- | --- |
| orderID | userID | ItemID |

# Conclusion

We have given the specifications and design plan for a system that can create food orders and keep a record of previous and current orders. Customers will be able to log into their account, log out of it, or register for one should they not already be signed up for an account. The system shall be robust to handle any incorrect inputs and let the user know what to properly enter. It is intuitive and easy to use, while taking up little space and being low impact and low cost to run and maintain.

# Data Dictionary