

Wilf’s App

Group 4



December 5, 2016

Mason Cooper (140328200), Neil Ong (140733080)

**Project Timeline:**

* Tuesday Oct. 25: Meeting to discuss roles and division of work, on the requirements phase.
* Sunday Oct. 30: Collaboration of work on requirements phase, assembled by phase leader – Neil.
* Monday Oct. 31: Submission of requirements phase.
* Tuesday Nov. 1: Meeting to discuss roles and division of work, on the analysis phase.
* Sunday Nov. 6: Collaboration of work on analysis phase, assembled by phase leader – Mason.
* Monday Nov. 7: Submission of analysis Phase.
* Monday Nov. 14: Meeting to discuss roles and division of work, on the design phase.
* Sunday Nov. 20: Collaboration of work on design phase, assembled by phase leader – Neil.
* Monday Nov. 21: Submission of design Phase.
* Tuesday Nov. 22: Meeting to discuss roles and division of work for final report.
* Thursday Nov. 24: Meeting to discuss roles and division of work, on the implementation phase.
* Sunday Dec. 4: Collaboration of work on final report, assembled by phase leader – Mason.
* Monday Dec. 5: Submission of final report and collaboration of work of the implementation phase.
* Tuesday Dec. 6: Submission of implementation phase.

**Skills Matrix:**

|  |  |  |
| --- | --- | --- |
| **Skills** | A | B |
| User Interface Design | S,I | I |
| Database Design |  | S,I |
| Middleware Design | I |  |

**OO Requirements**

**Actors:**

* Consumer
* Wilf’s employee

**Scenarios:**

1. User can use the app to view:
   1. Menu items
   2. Prices
2. User can use the app to order food and know when their food is ready for pick-up.

**Use Cases:**

Name: *ItemsForOrder*

Participating actor: Initiated by Consumer

Entry Condition: Consumer opens menu

Flow of events:

* User selects category
  + i.e. special, pasta, wings, pizza, drinks
* App displays list of items in the category
* User selects an item
* App opens item page displaying price and description
* User presses ‘add to order’ button
* App adds it to their order
* User Clicks on order tab
* App displays current order of the consumer and the total price
* User Presses checkout button
* App asks for email
* App sends consumer email confirmation

Exit Condition: Consumer receives order confirmation containing their order and order number.

**Non-functional requirements**

* App should provide email confirmation (order, price, order number) within 30 seconds of user entering their email.

<<include>>

Customer

Server

Email Send

Order Confirmed

View Order

Add to order

View Items

View Menu

**OO Analysis**

**Use Case:** *ItemsForOrder*

|  |  |  |
| --- | --- | --- |
| Entity | Boundary | Control |
| * Consumer * Employee | * Screen * Order Tab * Menu Tab * Order Button |  |

**Class Responsibility Collaboration**

|  |
| --- |
| Order() |
| * Contains items objects * Can output what’s in the order * Delete or add items in order object * Verifies if order is correct with user * Creates notification object once the order is confirmed |
| * Notification * Items * Menu |

|  |
| --- |
| Notification() |
| * Sends email to user with order number and total order. * Sends notification to user when food is ready. |
| * Order * User |

|  |
| --- |
| Menu() |
| * Uses arrays of objects to store item objects. Each an array is a category. * The menu can display itself to the user and what’s in its categories. * Can add or remove item objects from categories. * Confirms if the user is logged in or not. |
| * Order * User * Items |

|  |
| --- |
| User() |
| * The user class contains the unique information for each account. * This unique information is name, email, and password. * The user class can also create a brand-new user object. * It is called by the menu class and is used to confirm the user. |
| * Notification * Menu |

|  |
| --- |
| Items() |
| * Item objects are used to hold all the information of an object. * It contains key information for the specific object like description, price, and ingredients. * Its responsibility is to be used by the menu object and placed into its categories * It also can be added or removed from the order object. |
| * Order * Menu |

**State Diagram:**

Items()

Added

Start

Available In the order

Removed

Notification()

Sent

Notification sent

Sending

Resend

Failed

Failed to send

**Sequence Diagram:**

Order

Create

Add to Order

Order Tab

Items

Categories

Menu

Login

Email()

back ()

send\_order ()

send\_email ()

Confirm\_info() ()

Confirm\_order ()

Show\_order ()

Add\_to\_order ()

Dp\_item\_info ()

Dp\_items ()

Dp\_categories ()

password()

**OO Design**

**System Diagram**

Item

User

OrderInterface

Notification

LoginInterface

MenuInterface

*LoginInterface* subsystem implements the user interface of the login screen.

*MenuInterface* subsystem implements the user interface of the menu.

*OrderInterface* subsystem implements the user interface of the user’s order.

*User* subsystem implements the storage of the user’s login information (i.e. username, password, etc.)

*Notification* subsystem implements the communication between the user and Wilf’s

*Item* subsystem implements the storage of menu items.

**Rationale**

The System design is a layered format. This is because the application stems off the main menu system. From there the actor, can access the login, select items, view their order and checkout. This format was chosen because the actor will be allowed to view items on the menu without being logged in. However, when the user attempts to select items for order, they will be prompt to login. After the user logs in the will have access to their order and will be able to complete orders.

**Object Design**

**Object Description**

User:

The user object lets the user log in to their unique account. It also allows them to create a new account if they do not have one. For a user object to be created it needs to be passed both an email and a password. The user object then holds onto both email and password.

Once the user object is created from within the menu class. A method is called to ask if the user is a new or an existing user. If the user is an existing one a method is called for the user to sign in. The password and email is checked until the user enters valid information, then the user object is set to logged in. If the user is new, then a method is called to ask for an email and password and that is set up for the new account and is now considered logged in.

Menu:

The menu object is used to hold all the categories and items. Categories can be made by passing a string with the category name. Each category is an array of item objects stored within the Menu class. This class also displays itself to the user.

The menu class first calls upon the user class to get key information for the order class and notification class. Once the user object status is updated to logged in the menu is displayed to the user. A method is called to display the main menu page. From the main menu page the user can click on different categories, which will take them to various category pages. Once on a category page items are displayed. The item can be then clicked to open the item object.

Items:

Each item object is an item on the menu. An item was a description, a price, and allergies. This is all stored within the item class. It is put into a menu category. It also has the ability to add itself to the order.

Once the user clicks on an item and opens its page, a method is called to display its description and price. At the very bottom a button is placed with a method that can add the item to the order object.

Order:

The order class holds all the items that the user wants to order. It has an add and remove function for items. It can display the items and price it contains to the user. It must also confirm before an order is sent. Once confirmation is accepted it uses the notification object.

Once the user opens the order page, a method is called to display all the items and prices added to the order. The user can also user order remove method to remove item objects that they decided they do not want anymore. Once the user is ready confirm the order, they press a button which calls a method within the already initialized notification object that sends a confirmation email the user.

Notification:

The notification object is created and passed the user email from the user object and the order from the order object. Once both has been passed it sends an order email to the user. Also, once the order is ready a WILF’s employee is able to send a notification to the user email that the food is ready.

The notification object is created once the user object is set to “logged in”. The user object passes it key information like name and email. Once the order object calls the notification method to send the order confirmation email it creates one using the information from user and sends it to the email. It can also be used to send a notification to the user once the food is ready.

|  |
| --- |
| Notification() |
| * Sends email to user with order number and total order. * Sends notification to user when food is ready. |
| * Order * User |

|  |
| --- |
| Items() |
| * Item objects are used to hold all the information of an object. * It contains key information for the specific object like description, price, and ingredients. * Its responsibility is to be used by the menu object and placed into its categories * It also can be added or removed from the order object. |
| * Order * Menu |

|  |
| --- |
| User() |
| * The user class contains the unique information for each account. * This unique information is name, email, and password. * The user class can also create a brand new user object. * It is called by the menu class and is used to confirm the user. |
| * Notification * Menu |

**WILF’s App Class Diagram**

send()

login()

|  |
| --- |
| Menu() |
| * Uses arrays of objects to store item objects. Each an array is a category. * The menu can display itself to the user and what’s in its categories. * Can add or remove item objects from categories. * Confirms if the user is logged in or not. |
| * Order * User * Items |

|  |
| --- |
| Order() |
| * Contains items objects * Can output what’s in the order * Delete or add items in order object * Verifies if order is correct with user * Creates notification object once the order is confirmed |
| * Notification * Items * Menu |

add() / remove()

display() / go\_back()

**Data Flow Diagram:**

Verified

Database

Login

Item Information

Add to Order

Notification

Order

Items

Menu

Email/Password