Joseph Cavanaugh 5/20/2020

README

1. The user browses the menu of food items the Cafeteria offers as a whole list or by category.
   1. This requirement is satisfied – The button labeled “Menu” takes the user to the entire list of items available
2. The user places orders for any number of items and different quantities
   1. This requirement is satisfied – After going to the menu, clicking on any item in the menu will take the user to an area where they can select the quantity they want (up to 10, but not less than 1). The user can add the item to the cart from here, it will not take them to the shopping cart at this point. The back button will take the user back to the menu.
3. The user can view any order he/she placed using the order Id and *make changes*
   1. This requirement is partially satisfied – The user can view the order id, the total cost, and when it was ordered, but cannot make any changes to the order. They can see this order by hitting the button labeled “Orders”
4. *The user can rate and comment on the service after each order is fulfilled/delivered*
   1. This requirement is not satisfied –
5. The user is able to browse the menu and make orders any time with or without Internet connection. In other words, when there is no cell service and/or data service, or no WIFI, the user can still use the app as if it were online. The app will send the order to the cafeteria automatically once the network is available.
   1. This requirement is satisfied(?) – The user can order and it is added to the local database while both online and offline, however there is nothing to send the order to. (Very confused about this requirement, its probably not done if the order needed to be sent to a repository)
6. The innovative owner of the cafeteria is famous for adding new food items. The app should make those food items available to the user as soon as possible.
   1. This requirement is fulfilled – The user(owner) will be able to add a new menu item. This is done by hitting the “Add New Item” button. The user can then make a name, a type, a description and price for the item. This item will instantly be put into the repository, but will not be updated to the local until the entire menu updates.
7. The app uses WorkManager to update menu items from the Lunchilicious web service. The updates should be performed once every 30 minutes.
   1. This requirement is satisfied – The UpdateWorker updates the old menu to a new menu once every 30 minutes.
8. The user can order food items by selecting a menu item from any of the four lists and specifying the quantity, which should never be less than 1. it is recommended that you use Fragments to implement this function.
   1. This requirement is partially satisfied – This is the same as 2, but is not selected from any of the four lifts. A way to fully satisfy this would be to have buttons that pass in a value that only designates for that type to be selected and sort by that.
9. The selected item should be added to a "shopping cart" and the user can view the contents of the shopping cart any time. You can decide where the shopping cart is stored, e.g., in memory or in database.
   1. This requirement is satisfied - After adding an item to the shopping cart the user can view the shopping cart by going to the main menu and hitting the button labeled “Cart”. From here the user can look at their current cart and place the order. Unfortunately, there is no way for the user to currently alter their cart. Emptying the cart without ordering isn’t possible at the moment.
10. When the shopping cart is being displayed, the user can place the order. The order is saved in the local database
    1. This requirement is satisfied – In the shopping cart, the user can place their order which will empty the cart. The order is moved to the “Orders” part of the app and are displayed in order.
    2. The id is based on the year, day, hour, minute, second. The date is year, day, hour. The total cost of the order is displayed alongside these
11. The user is able to view an order by the order ID. The user enters the order ID and the app displays the order in a user-friendly manner.  For example, it should display at least the order ID, order date and the total cost. Ideally, it would display all the items and their quantity and unit cost, and the total cost. You may assume there is no sales tax.
    1. This requirement is not satisfied – The user cannot currently able to order the orders by their Id. This could probably be implemented with the SQLite and ordering by date. Alongside this displaying all the items could probably be handled similarly to the menu. By tapping on an order you could view all the items that were in the cart at the moment of the order.

Additional

Nothing