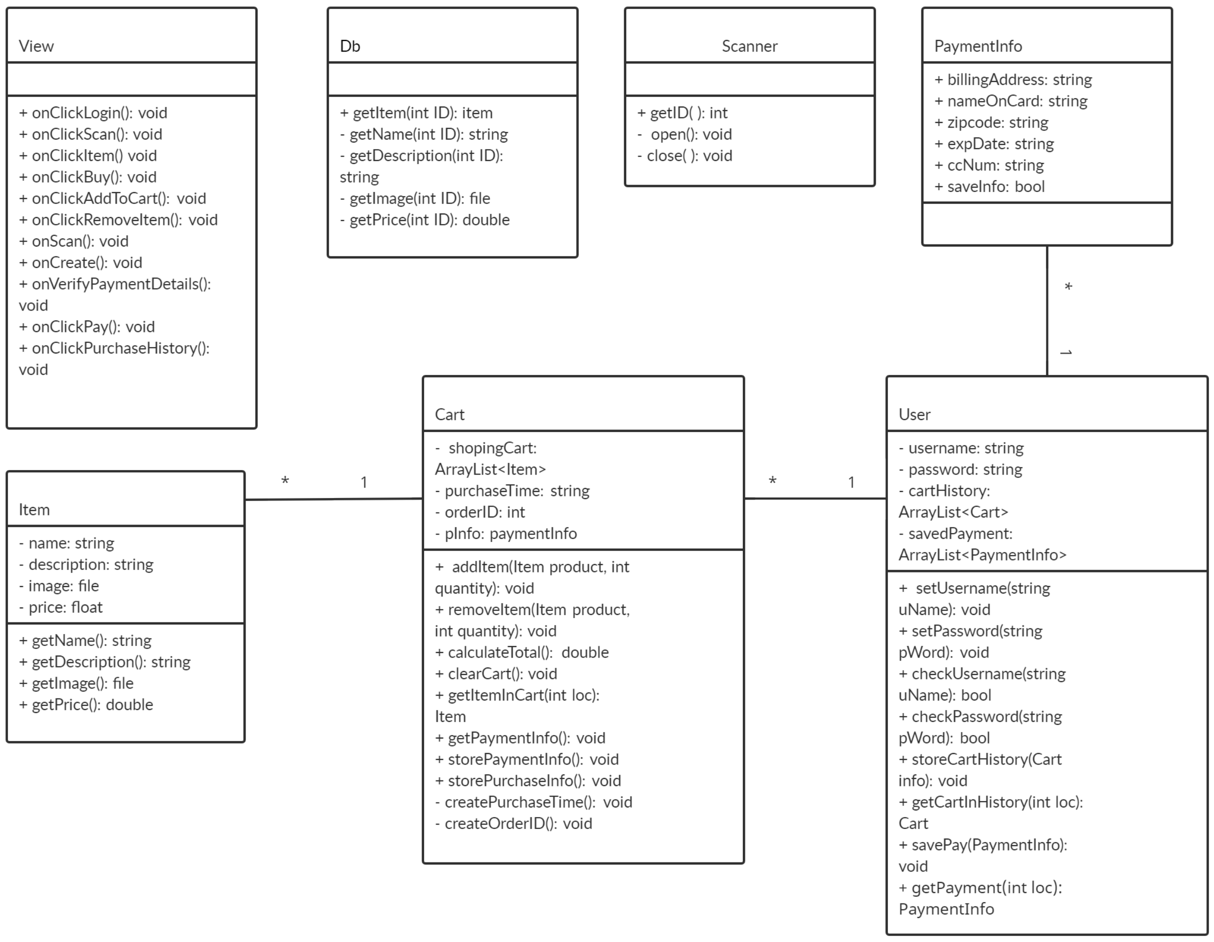
Group Name: Convenience Store Personal Scanner

Group Members: Andrew McKernan, Ian Hoole, Jack Gable, Josue Vargas, Logan Santos

UML Diagram


View
+ onClickLogin(): void
+ onClickScan(): void
+ onClickItem() void
+ onClickBuy(): void
+ onClickAddToCart(): void
+ onClickRemoveItem(): void
+ onScan(): void
+ onCreate(): void
+ onVerifyPaymentDetails(): void
+ onClickPay(): void
+ onClickPurchaseHistory(): void
Db
+ getItem(int ID): item
- getName(int ID): string
- getDescription(int ID): string
- getImage(int ID): file
- getPrice(int ID): double
Scanner
+ getID( ): int
- open(): void
- close( ): void
PaymentInfo
+ billingAddress: string
+ nameOnCard: string
+ zipcode: string
+ expDate: string
+ ccNum: string
+ saveInfo: bool
*
1
Item
- name: string
- description: string
- image: file
- price: float
+ getName(): string
+ getDescription(): string
+ getImage(): file
+ getPrice(): double
*
1
Cart
- shopingCart: ArrayList<Item>
- purchaseTime: string
- orderID: int
- pInfo: paymentInfo
+ addItem(Item product, int quantity): void
+ removeItem(Item product, int quantity): void
+ calculateTotal(): double
+ clearCart(): void
+ getItemInCart(int loc): Item
+ getPaymentInfo(): void
+ storePaymentInfo(): void
+ storePurchaseInfo(): void
- createPurchaseTime(): void
- createOrderID(): void
*
1
User
- username: string
- password: string
- cartHistory: ArrayList<Cart>
- savedPayment: ArrayList<PaymentInfo>
+ setUsername(string uName): void
+ setPassword(string pWord): void
+ checkUsername(string uName): bool
+ checkPassword(string pWord): bool
+ storeCartHistory(Cart info): void
+ getCartInHistory(int loc): Cart
+ savePay(PaymentInfo): void
+ getPayment(int loc): PaymentInfo

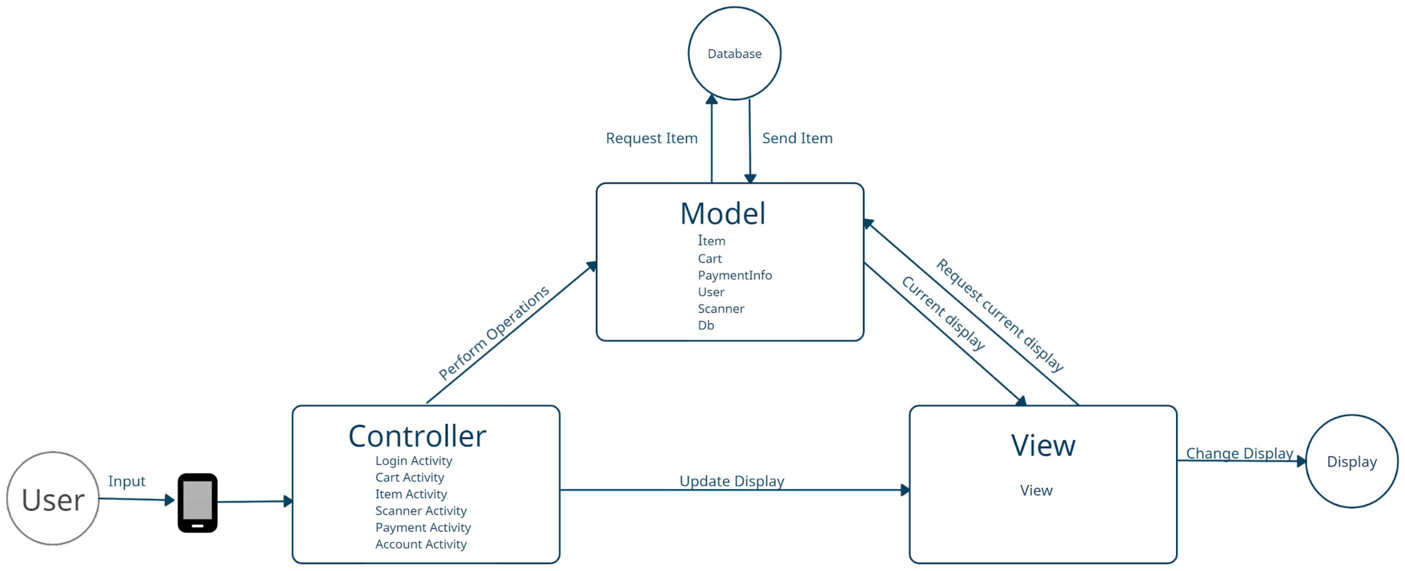
Demonstration of class inheritance, method inheritance, and static variables

Class inheritance: No class inheritance within our application.

Inherited methods: An example of an inherited method would be the getID() in the scanner class. The item class inherits this scanner method in order to populate the item that was just scanned.

Static variables: We have a static scanner variable called scn. This variable is located within the controller class and would be utilized by the view class.

Subsystems



Dataflow between Subsystems

The data that will be shared in our system starts with the user opening the application on their smartphone and accessing the controller. This Controller will serve as an entry point in our application and request updates to the display. As well as perform the requested operations from the Model, whether that be an item, cart, user, scanner, or the DB the model will request the item from the systems Database and the Database will send the needed information. The View will request and receive information about our current state from the Model, and accordingly modify the display.

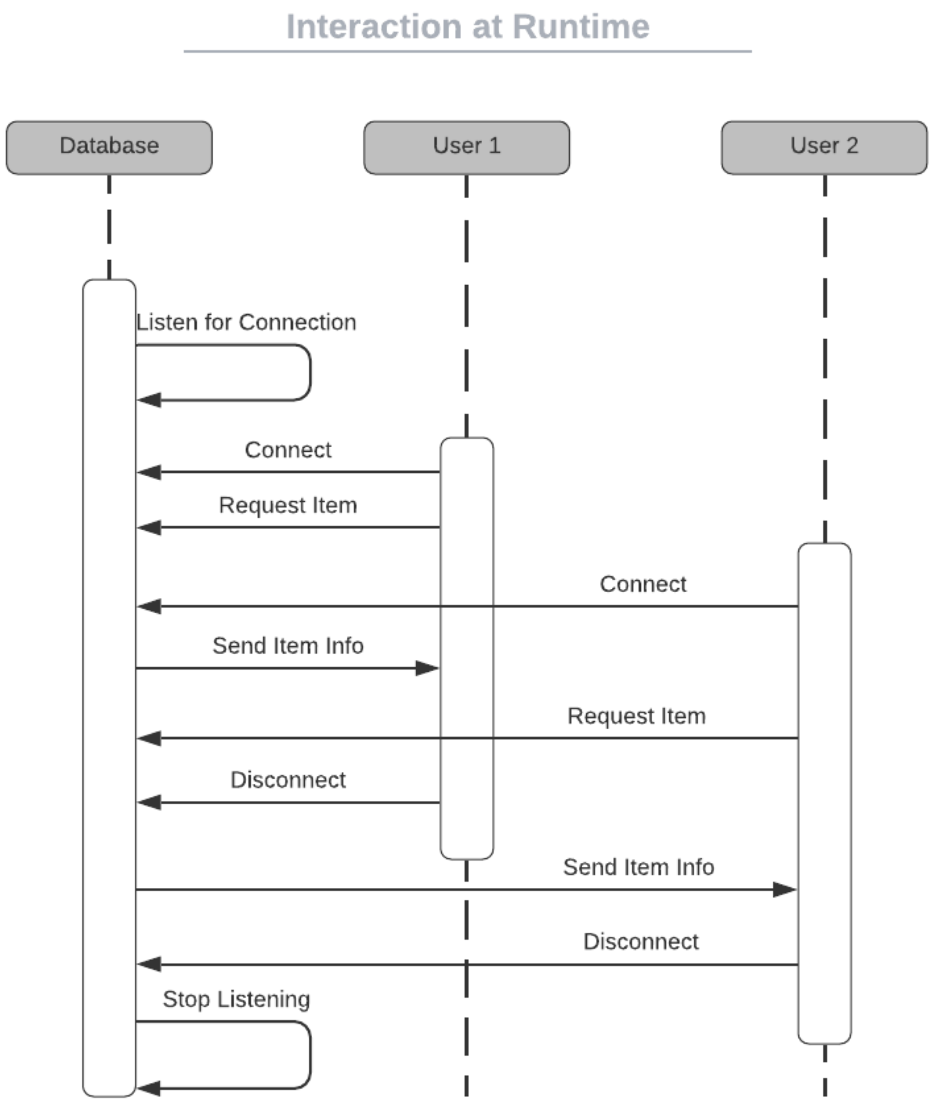
Ex. Scanning a product

1. User is in the Scanner Activity and scans a barcode
2. Scanner sends the barcode data to the DB class, which in turn looks up the barcode in the database.
3. The database returns information on the item to the DB class, which returns an

item object to the scanner.

4. The scanner presents an Item Activity with information on the item to the user along with the option of adding it to cart.
5. After the user adds to cart, the scanner sends the item object to the cart object and adds it to the vector of items.
6. The user is then presented an updated cart Activity with the item they scanned in it.

Interaction at Runtime



MVC on Android

Views / Controllers AKA Activities:
What we call a view and controller Android groups together and calls an activity. It is composed of an XML file that describes the on-screen layout and a class that controls the “views” inside of it.

- Login Activity
    - The Login Activity will be a simple login screen controlled by the Login controller. Once credentials are confirmed, the user is allowed access to the apps functionality.
    - List of methods
        - onClickLogin()
            - Takes the login credentials and verifies user
- Cart Activity
    - The cart Activity shows whatever is in the users cart at the moment. Will have a ListView along with the total price at the bottom of the screen. This will be controlled by the Cart Controller.
    - List of methods
        - OnClickItem()
            - See item description
        - OnClickBuy()
            - Takes user to purchase Activity
        - OnClickScan()
            - Takes user to scanner Activity
- Item Activity
    - The item activity shows the content of the item object, which includes item name, description, price, and a picture of the item.
    - List of methods
        - OnClickAddToCart()
            - Adds item to the cart.

- Scanner Activity

    - The scanner Activity will allow the user to scan barcodes using the phones camera system. Will have a camera viewfinder along with a reticle so the user can point to the barcode that they want scanned. This will be controlled with the scanner Controller.
    - List of methods
        - OnScan()
            - Shows the user the item they have scanned through an Item activity.
        - OnCreate()
            - Open viewfinder and begin looking for barcodes.
- Payment Activity
    - The payment activity allows the user to pay for what is in the cart. It is controlled by the purchase controller.
        - verifyPaymentDetails()
            - Checks that the payment details are valid.
        - OnClickPay()
            - Purchases items with payment entered by the user.
- Account Activity
    - The account Activity will let the user see and manage their account. It will display basic account information such as username, email, and past purchases. This will be controlled with the Account Controller.
        - OnClickPurchaseHistory()
            - Grab and display users purchase history.

Design Choices & Important Classes

One of the more important design choices we made was the best way to store product information in a hierarchy of classes. We have three classes that are primarily a way of storing and manipulating product information. We chose a simple item class that stores all the information needed for a single product, this item is then stored within an array in a cart class. Once the transaction is complete, we then store the cart class in an array the user class, this is useful for constructing a purchase history for the user. It may seem un-intuitive that the association between carts and user is many to one,

however this is because the cart class is used to store purchase history from past checkouts. This design choice allowed us to save processing time and memory by transferring data between different classes.
An additional significant design choice we made was the implementation of a username and password. We chose to store the username and password locally, meaning a user can not sign in on other devices. We chose this primarily due to time constraints, however we deemed a username and password as mandatory due to the application storing payment information. Verifying the username and password locally insures that only the intended user is able to access their payment info, however we lose the functionality of being able to sign in across multiple devices.

Item, Cart, PaymentInfo, DB, and User are all used primarily as a way to store and interpret data. These are necessary classes needed to have a good control and storage of data. We believe the View and Scanner classes will be the most significant in development when it comes to creating an enjoyable user experience. With the view class we wish to focus on an intuitive and interactive interface. With the scanner we wish to create a speedy and easy to use scan function.