**Final Project: Dispensewith.me**

**Class**

**Team Members**

* Angel Zambrano

Note: try using this for the card number 79927398713

**Program Goals**

* Our platform links buyers to a digital storefront. Users can order the best products anytime, day or night. Our goal is to become more than just an online delivery service, to create a relationship between your brand and your customers.
* Delivery: Allow the user to create a custom delivery date
* Card verification: our proprietary algorithm checks if a card is valid.

**Functionality**

* The game asks the user to input their username and password.
  + Create
* The user can also create an account
* The majority of the code is very compartmentalized.
* User can add products to their cart
* User can check out
* User can set date of delivery
  + There is validation here
  + Must be up to a year
  + If the date is past the current date, it is also not valid
* Card verification
  + Checks if the user entered a valid card using luhn algorithm
* Prints receipt

**Targeted Audience(s)**

* Younger Millennials
* Online shoppers/people prefer contactless shopping
* Parents

**Strengths**

* The program is very robust.
* There is portability and a lot of encapsulation

**Weaknesses**

* I didn't get to connect it to some API
* No data saving
* Inputting address
* I could have used a better implementation for my date
  + A calendar system

**Pseudocode**

**LogginController**

Display the log in meny

Get decision

If decision is 1

Call log in with database

If decision is 2

Call createAccount()

createAccout()

Display title

While true

Get username

Get password

If user does not exist, return

logginIn(self, database)

While true

Try

Get username

Get password

Create pending username

If (user does not exists)

Return usr

Else

Raise RunTimeError

Except

Continue

**StoreController**

Initializer

Create a store object

Create view object

While true

Get user input

Check if input is valid

If it is valid continue

Check if user input is -1

Break

If there is enough inventory

Update the view

Add to cart

**StoreController**

Initializer

Create a store object

Create view object

While true

Get user input

Check if input is valid

If it is valid continue

Check if user input is -1

Break

If there is enough inventory

Update the view

Add to cart

**Product**

Name

Price

Inventory

Def hasInventory()

If there is enough inventory return true

Return false

Def getPrice

Return price

Def decreaseInventory

Decrease inventory minus 1.

**Cart**

Product = []

Initializer

Total = 0

addProduct()

Self.total + price of product

get TotalProductInCart

Return size of product

**User**

Initializer

Set the user name to surname

Set the password to password

Create a new cart object

addProduct()

Self.total + price of product

get TotalProductInCart

Return size of product

**Checkout Controller**

Initializer

Get authorization from the user

Get card number by calling checkLuhn

Get valid date

Write receipt with date

For i in to size of caert

Write each item of the cart

Write cart total

Checkauthorization

While true

Get authorization while input is not N

GetValidDate

While true

Read the date in a dd/mm/yy format

If the format is not corrent

Continue

If the date isnt not in the correct range

Continue

Return the date as a string

**Main**

Create a database with fake data

Call the loggin controller

Get the current user

Call the storecontrolller

Call Checkout