Go Pasar SG

A marketplace where local produce sellers can list their products for sale.

The platform supports many seller to many buyers.

Objective

To understand the challenges of building and maintaining e- commerce platform.

Project Focus

User Authentication & Session Management

- Controlling access to the site

Product Discovery

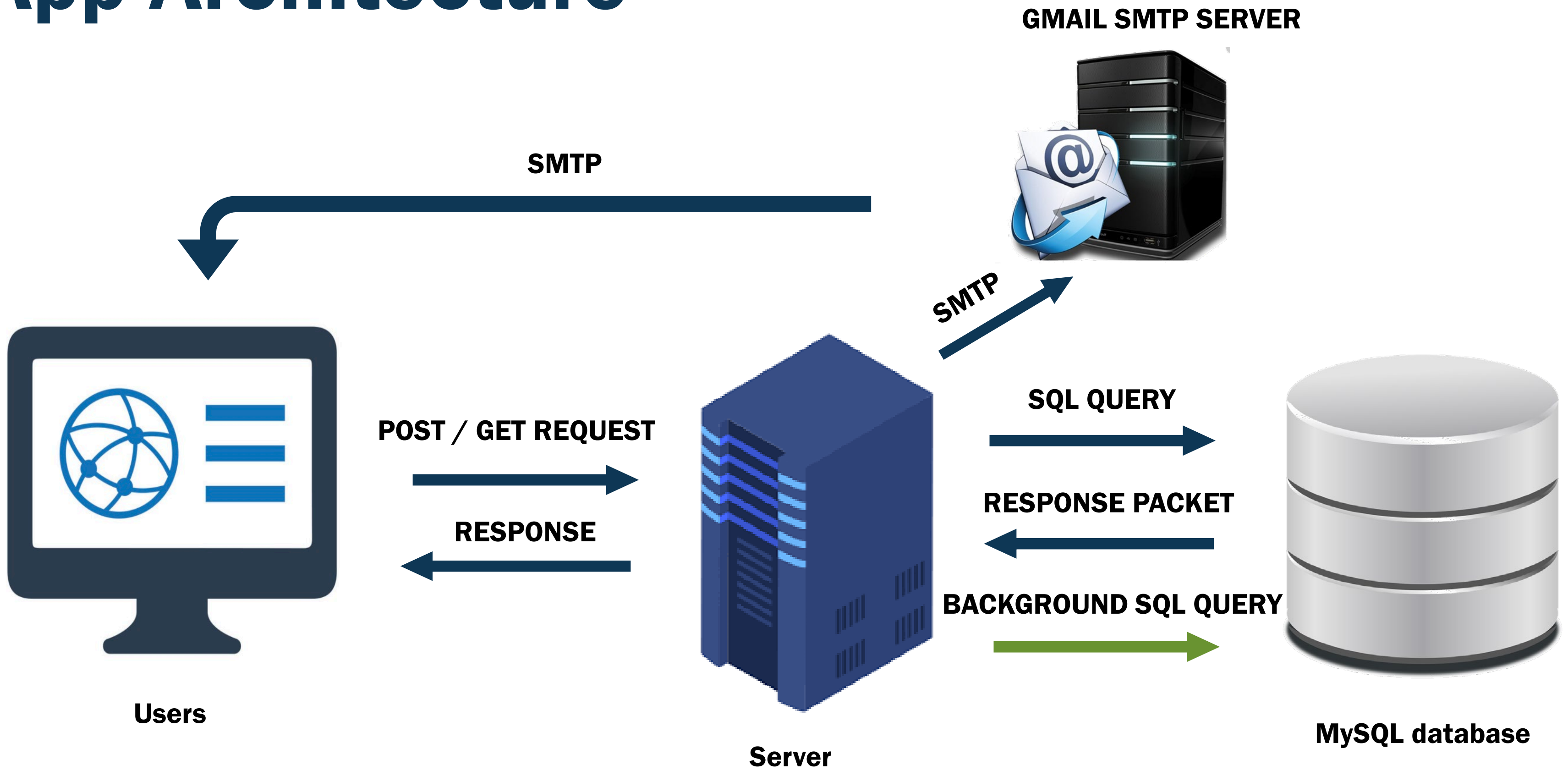
- Helping users find/discover what they want with the least effort

- ### Transaction Facilitation

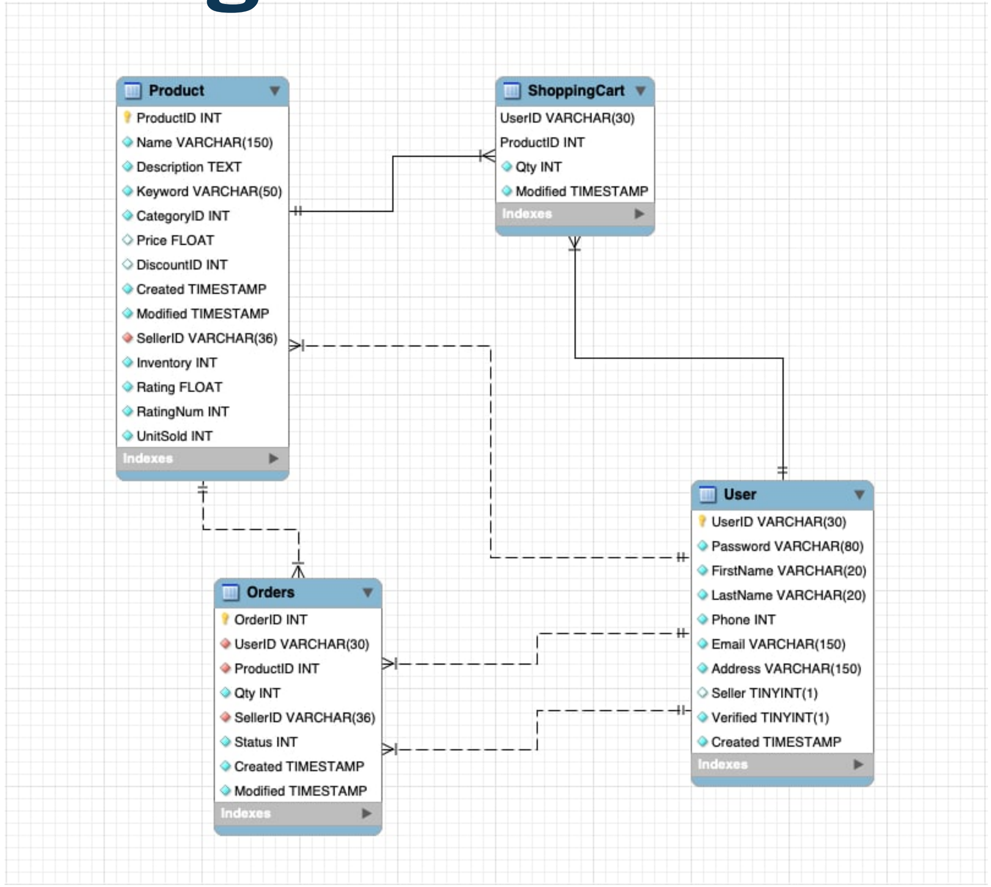
- Facilitating communication between sellers & buyers

Demo

App Architecture

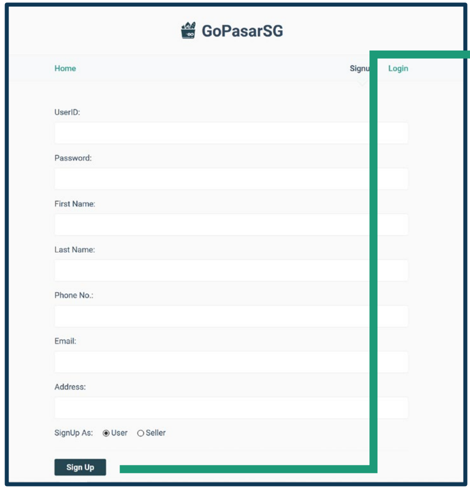


MySQL ER Diagram for GoPasar SG



User Authentication
Session Management

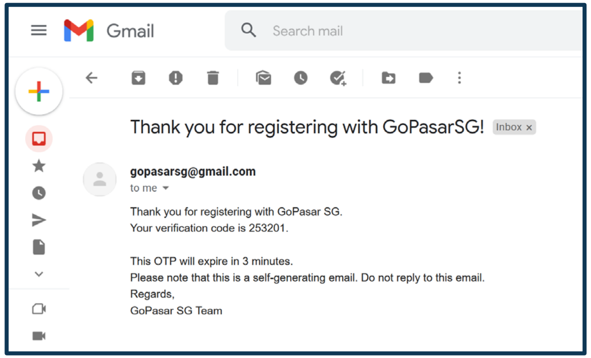
Client



HTTP response can be written instantly while OTP is being prepared and delivered.

Server

Goroutine 1



Thank you for registering with GoPasarSG!

gopasarsg@gmail.com
to me

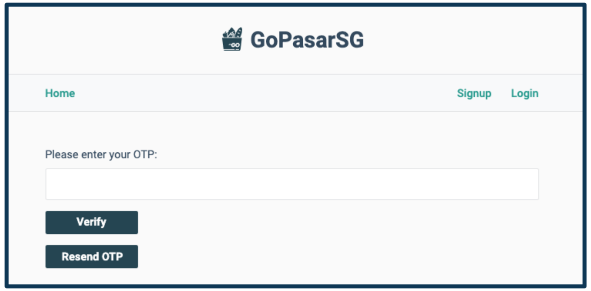
Thank you for registering with GoPasar SG.
Your verification code is 253201.

This OTP will expire in 3 minutes.
Please note that this is a self-generating email. Do not reply to this email.
Regards,
GoPasar SG Team

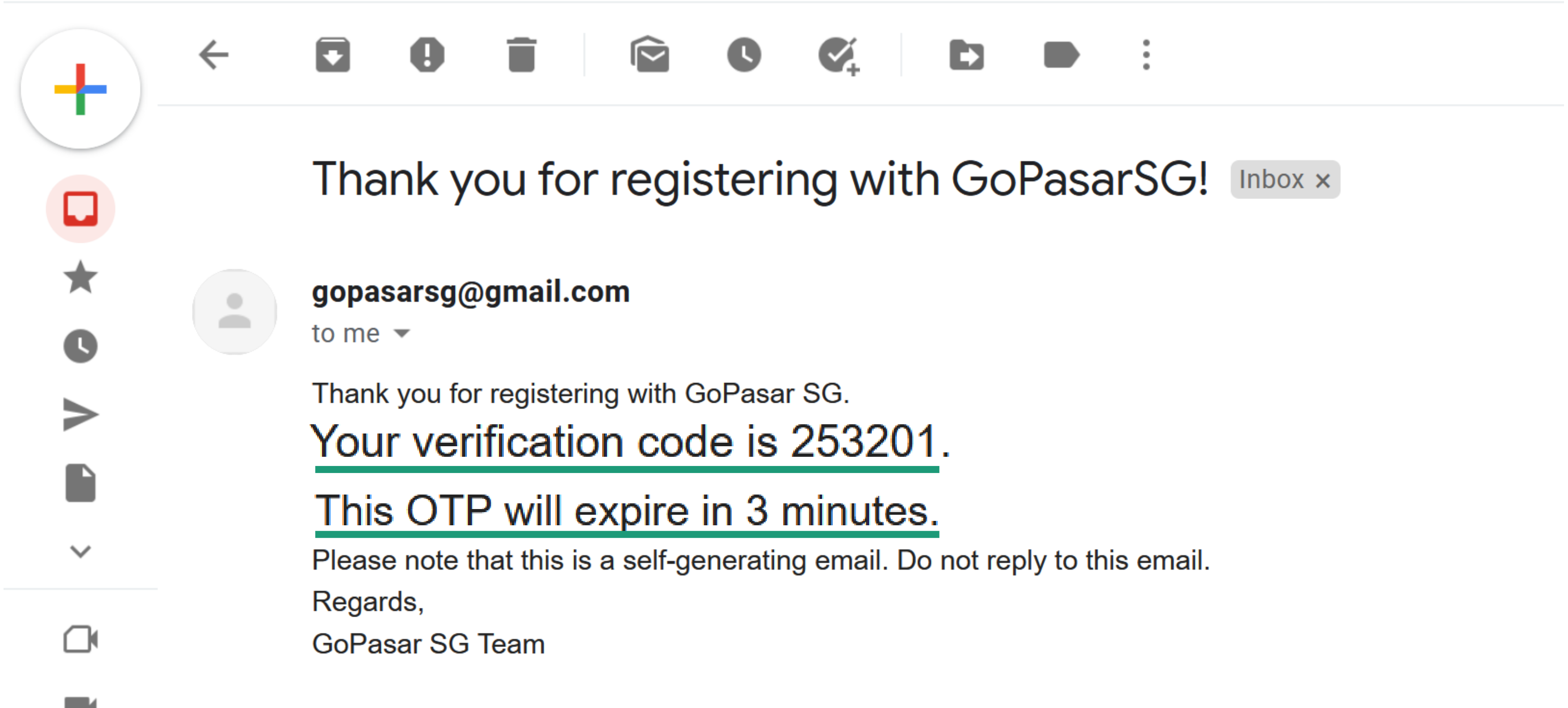
Store for verification

Delete after 3 min.

OTPs



User Verification via OTP



Thank you for registering with GoPasarSG!

gopasarsg@gmail.com

Thank you for registering with GoPasar SG.

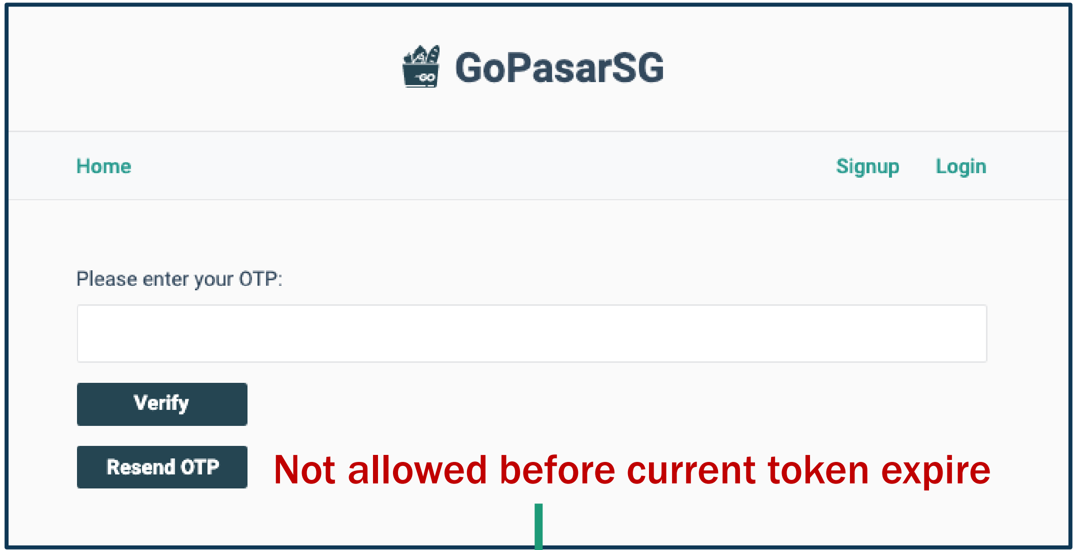
Your verification code is 253201.

This OTP will expire in 3 minutes.

Please note that this is a self-generating email. Do not reply to this email.

Regards,

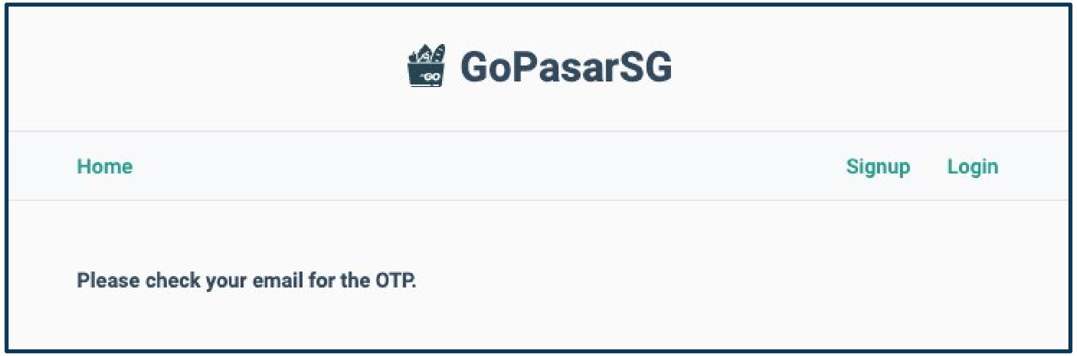
GoPasar SG Team


GoPasarSG

Home Signup Login

Please enter your OTP:

Not allowed before current token expire


GoPasarSG

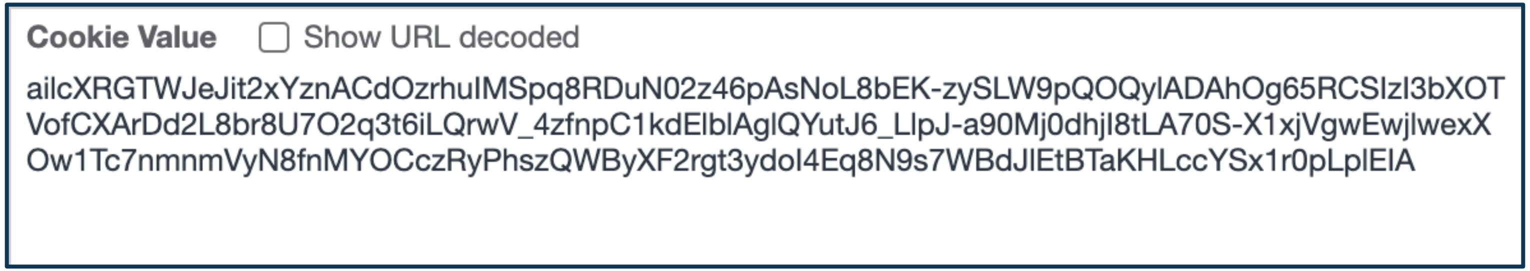
Home Signup Login

Please check your email for the OTP.

Session Management

Client-side session management

- Session information is stored as an encrypted string in the session cookie
- Encryption is performed using a 32 bytes secret key



Cookie Value ☐ Show URL decoded

```
ailcXRGTWJeJit2xYznACdOzrhuIMSpq8RDuN02z46pAsNoL8bEK-zySLW9pQOQylADAhOg65RCSIzI3bXOT
VofCXArDd2L8br8U7O2q3t6iLQrwV_4zfnpC1kdElblAglQYutJ6_LlpJ-a90Mj0dhjI8tLA70S-X1xjVgwEwjlwexX
Ow1Tc7nmnmVyN8fnMYOCczRyPhszQWByXF2rgt3ydol4Eq8N9s7WBdJlEtBTaKHLccYSx1r0pLplElA
```

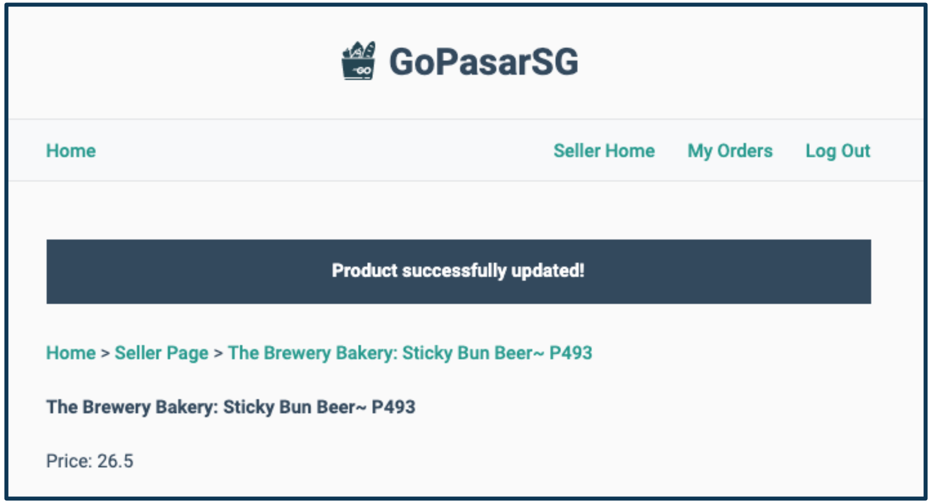
Cryptography should be left to experts:

- github.com/golangcollege/sessions

Session Management

Cool feature provided by this session package – we can store and retrieve messages in the cookie

- Display confirmation message for some actions users perform.
- However, there is a limit to the amount of information that can be stored.



Product successfully updated!

Home > Seller Page > The Brewery Bakery: Sticky Bun Beer~ P493

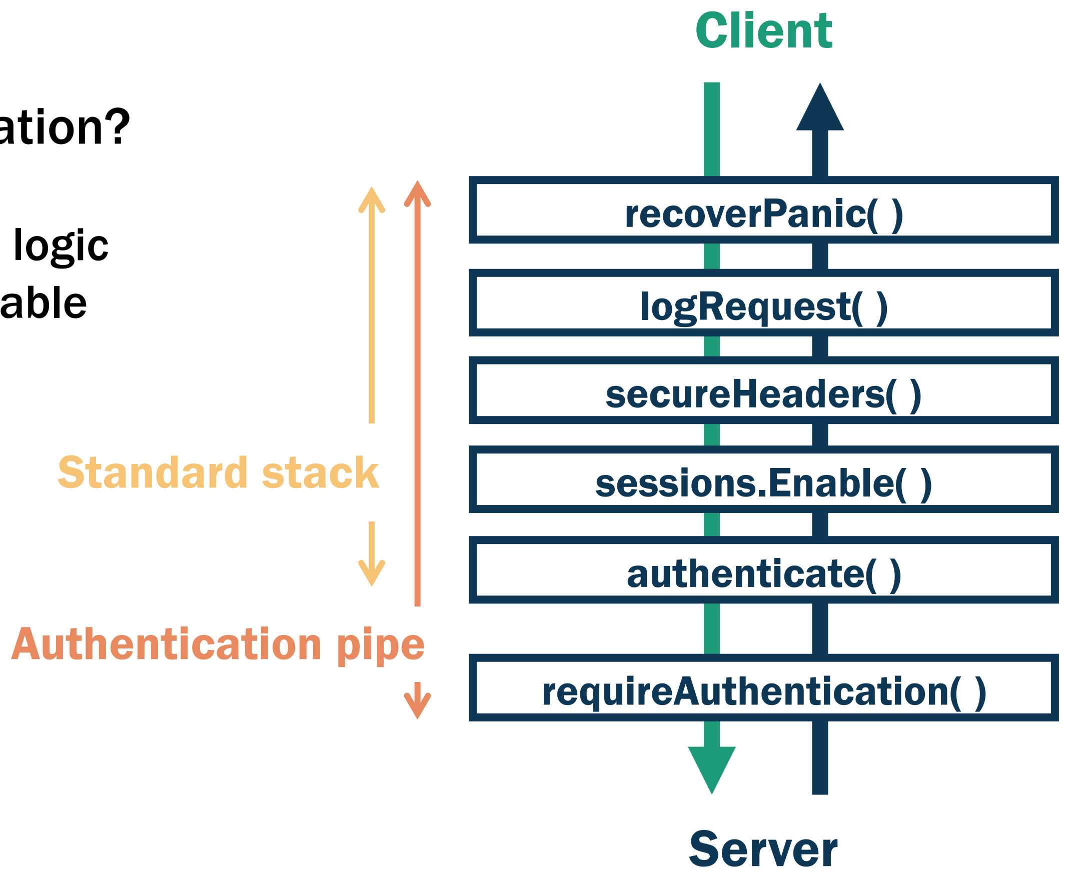
The Brewery Bakery: Sticky Bun Beer~ P493

Price: 26.5

Authentication Middleware

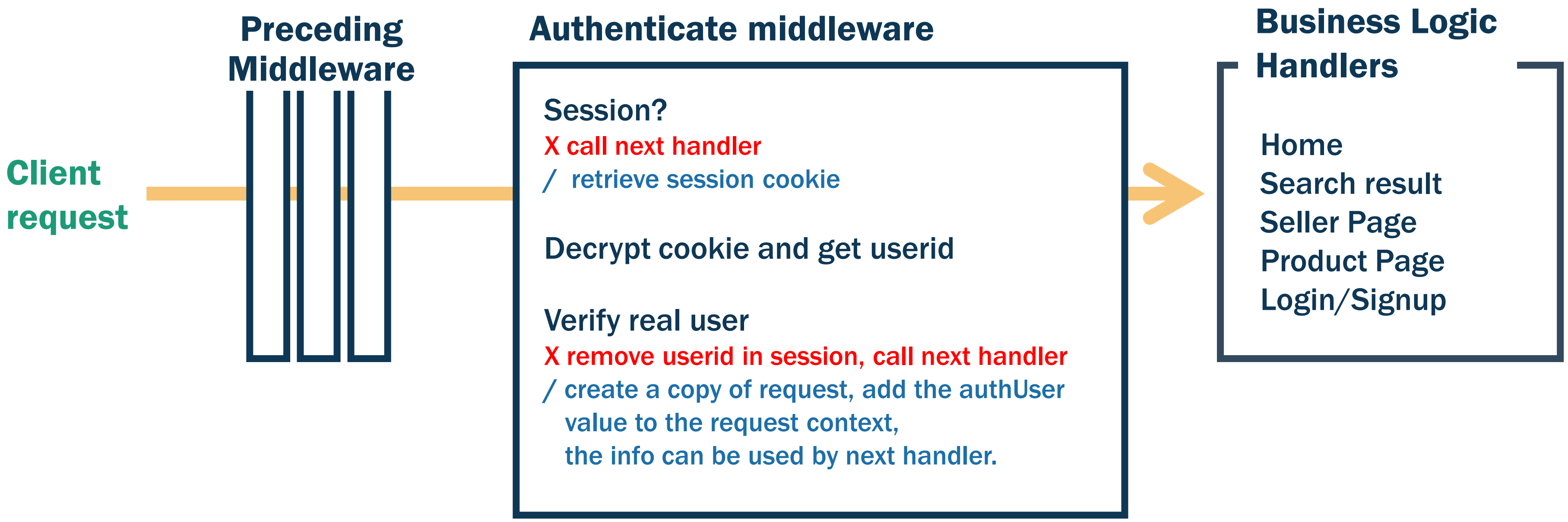
Why middleware for authentication?

- can be handled uniformly
- separated from main business logic
- middleware are reusable, sharable



Authentication Middleware

Standard middleware stack



Authenticate middleware

Session?
X call next handler
/ retrieve session cookie

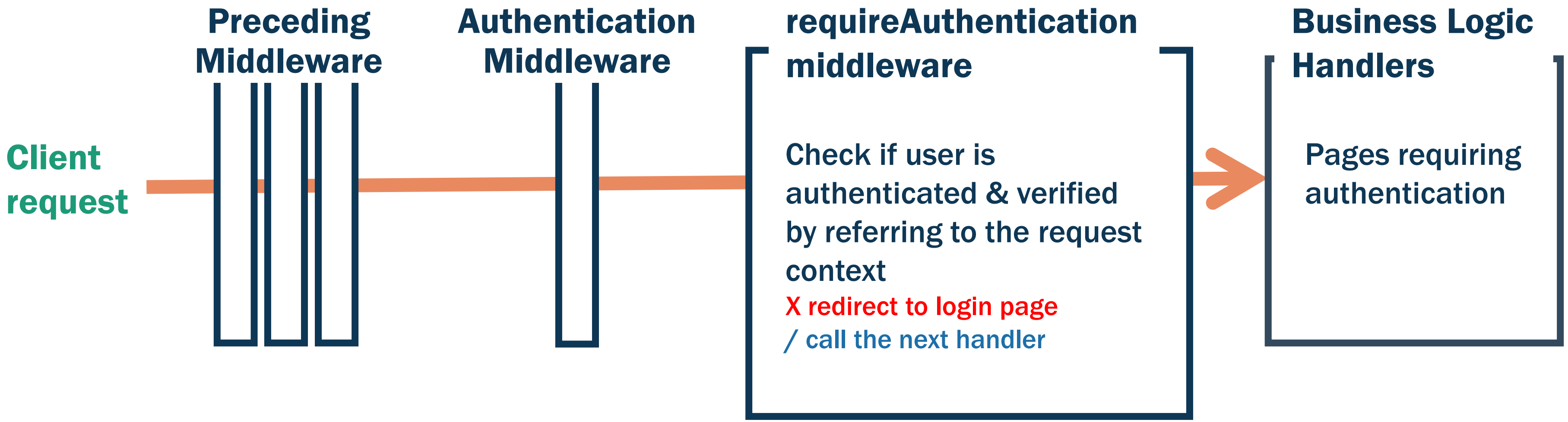
Decrypt cookie and get userid

Verify real user
X remove userid in session, call next handler
/ create a copy of request, add the authUser value to the request context, the info can be used by next handler.

Home
Search result
Seller Page
Product Page
Login/Signup

Authentication Middleware

Authentication Pipe middleware chain

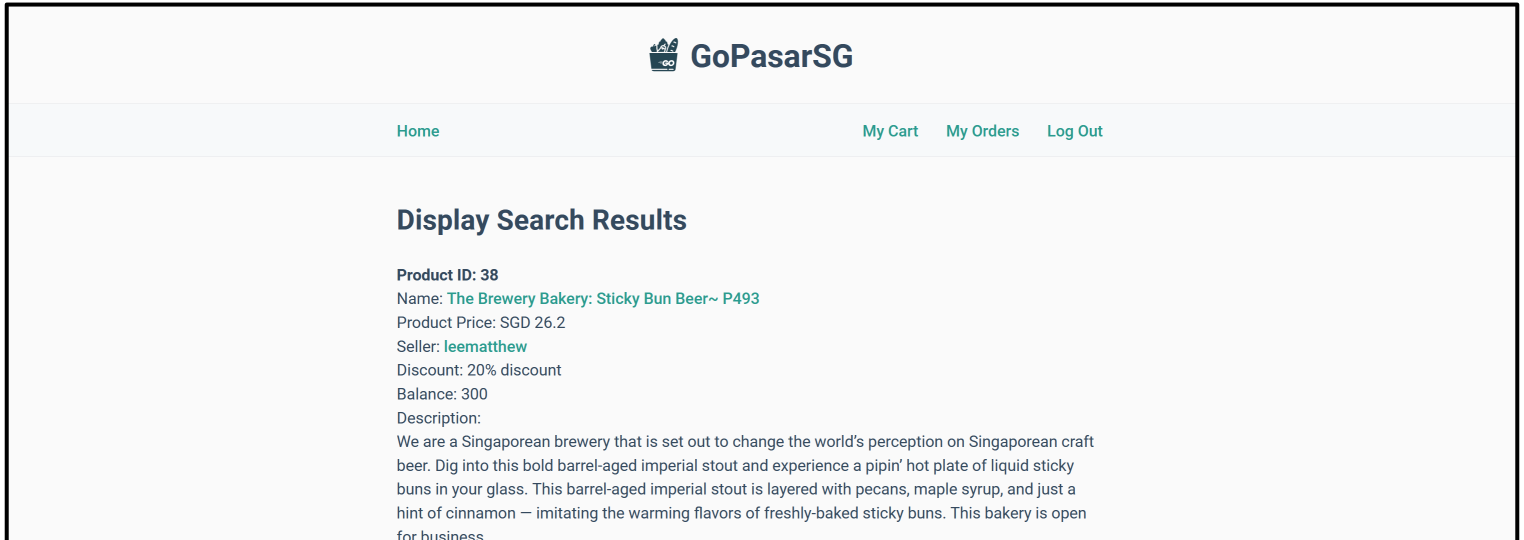

Client request
Preceding Middleware
Authentication Middleware
requireAuthentication middleware
Check if user is authenticated & verified by referring to the request context
X redirect to login page
/ call the next handler
Business Logic Handlers
Pages requiring authentication

Product Discovery

Searching

Based on inverted index (Go Map Type) for product name, description and keywords

- Stored in memory & refreshed periodically for up to date search results
- Complexity O(1) search for a single token search.



GoPasarSG

Home · My Cart · My Orders · Log Out

Display Search Results

Product ID: 38
Name: The Brewery Bakery: Sticky Bun Beer~ P493
Product Price: SGD 26.2
Seller: leematthew
Discount: 20% discount
Balance: 300
Description:
We are a Singaporean brewery that is set out to change the world’s perception on Singaporean craft beer. Dig into this bold barrel-aged imperial stout and experience a pipin’ hot plate of liquid sticky buns in your glass. This barrel-aged imperial stout is layered with pecans, maple syrup, and just a hint of cinnamon — imitating the warming flavors of freshly-baked sticky buns. This bakery is open for business

Input

Slice of structs with ID, Name, Desc, Keyword fields

	ID	Name	Desc	Keyword		
{	51	Singapore ice cold beer	ice cold beer	beer	}	,
{	52	Malaysia fresh cold beer	ice cold beer	beer	}	,
{	53	Belgian IPA beer	cold beer	beer	}	,
{	54	French ice cold beer	brewed in France	beer	}	,
{	55	USA ice cold beer	ice beer	alcohol	}	

Output

Maps mapping word to IDs at which word is found

Map for Name

beer	51 52 53 54 55
belgian	53
cold	51 52 54
french	54
fresh	52

...

Map for Desc

beer	51 52 53 55
brew	54
cold	51 52 53
franc	54
ice	51 52 55

Map for Keyword

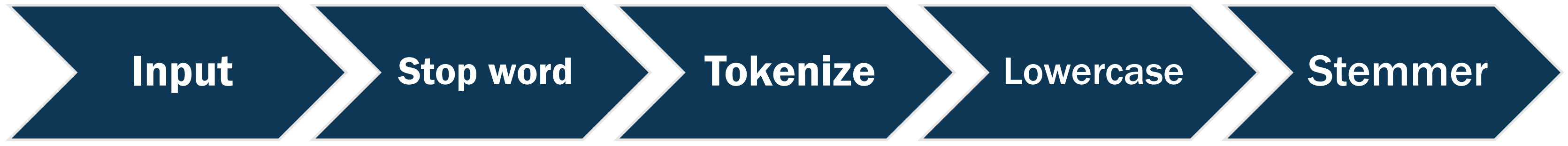
alcohol	55
beer	51 52 53 54

Preparation: Text analysis

```
{"fish", "fish"},
{"fished", "fish"},
{"fisherman", "fisherman"},
{"fishermen", "fishermen"},
{"fishes", "fish"},
{"fishing", "fish"},
```

Around 500 stop words recognized

Around 29k words recognized



“~~An~~ Apple ~~A~~ Day ~~Keep~~ ~~The~~ Doctor ~~Away~~”

“Apple Day Doctor”

[“Apple”, “Day”, “Doctor”]

[“apple”, “day”, “doctor”]

[“apple”, “day”, “doctor”]

Search

Search terms

ice
cold
beer

Map for Name

beer	51 52 53 54 55
belgian	53
cold	51 52 54
french	54
fresh	52
ice	51 54 55
ipa	53
malaysia	52
singapor	51
usa	55

Map for Desc

beer	51 52 53 55
brew	54
cold	51 52 53
franc	54
ice	51 52 55

Map for Keyword

alcohol	55
beer	51 52 53 54

Match weightage:

- Keyword x 4
- Name x2
- Description x 1

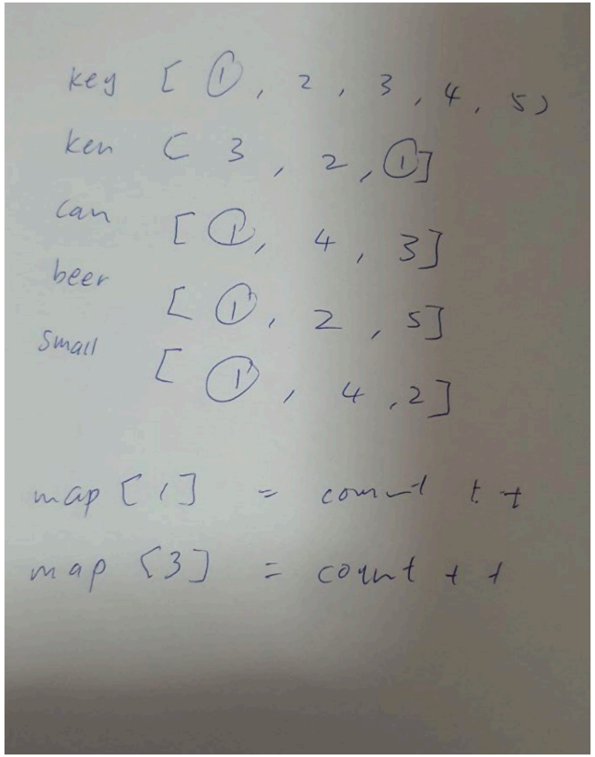
E.g. “ice” →

Map Score

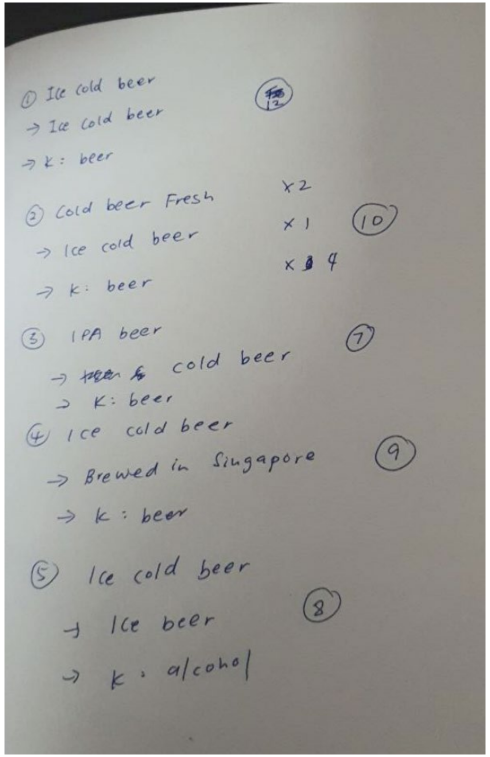
map[51]+=2, map[54]+=2, map[55]+=2,

map[51]++ , map[52]++ , map[55]++

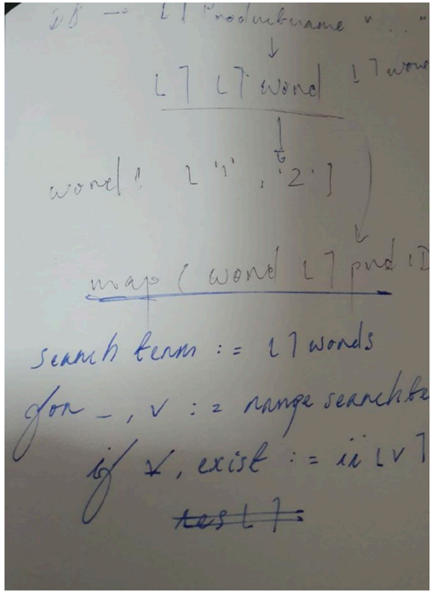
/* keyword map does not contain the word “ice” */



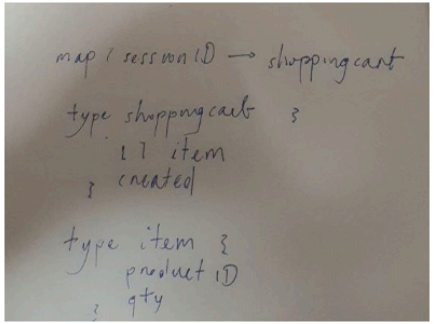
```
key    [ ①, 2 , 3 , 4 , 5 )
ken    ( 3 , 2 , ① ]
can    [ ①, 4 , 3 ]
beer   [ ①, 2 , 5 ]
small  [ ①, 4 , 2 ]

map [ 1 ] = count ++
map [ 3 ] = count ++
```



```
① Ice cold beer                    ⑫
 → Ice cold beer
 → k: beer

② Cold beer Fresh          x2
 → Ice cold beer           x1       ⑩
 → k: beer                 x 4

③ IPA beer                          ⑦
 → ~~[illegible]~~ cold beer
 → K: beer
④ Ice cold beer
 → Brewed in Singapore              ⑨
 → k : beer

⑤ Ice cold beer
 → Ice beer                         ⑧
 → k : alcohol
```



```
DB → [ ] "productname " . . "
          ↓
   [ ] [ ] word   [ ] word
          ↓
word ! [ '1' , '2' ]

   map ( word [ ] prod ID

search term := [ ] words
for _, v := range searchte
    if v, exist := ii [v]
       ~~tes [ ]~~
```



```
map ( session ID → shoppingcart

type shoppingcart {
      [ ] item
      created
}

type item {
      product ID
      qty
}
```

Input

Slice of structs with ID, Name, Desc, Keyword fields

	ID	Name	Desc	Keyword		
{	51	Singapore ice cold beer	ice cold beer	beer	}	,
{	52	Malaysia fresh cold beer	ice cold beer	beer	}	,
{	53	Belgian IPA beer	cold beer	beer	}	,
{	54	French ice cold beer	brewed in France	beer	}	,
{	55	USA ice cold beer	ice beer	alcohol	}	

Output

Ranked result for search text = "ice cold beer"

Score for product ID 51 → (2X3) + (1X3) + (4X1) = 13

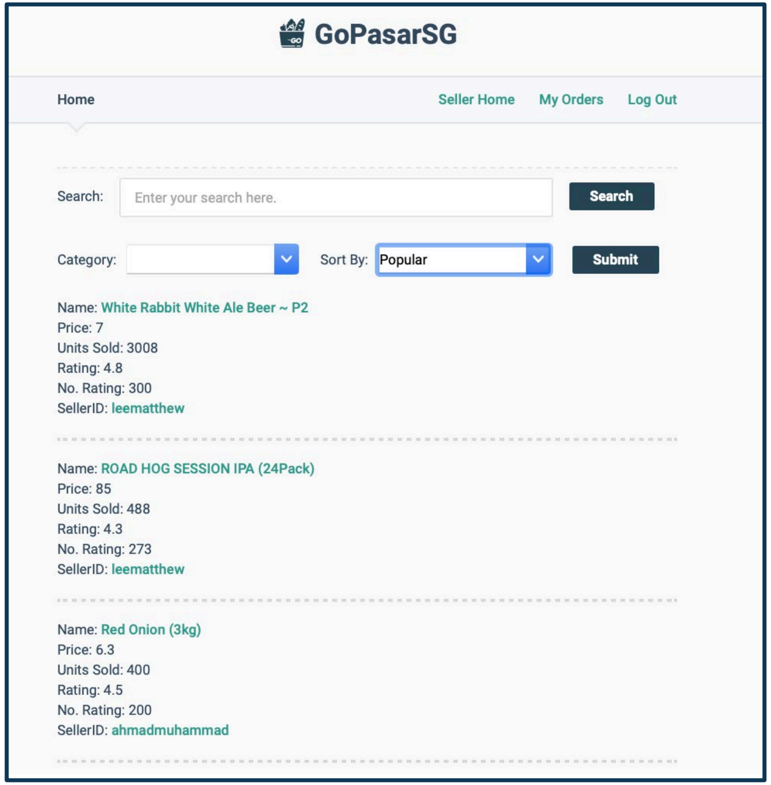
Score for product ID 52 → (2X2) + (1X3) + (4X1) = 11

Score for product ID 53 → (2X1) + (1X2) + (4X1) = 8

Score for product ID 54 → (2X3) + (1X0) + (4X1) = 10

Score for product ID 55 → (2X3) + (1X2) + (4X0) = 8

Sorting


GoPasarSG

Home

Seller Home My Orders Log Out

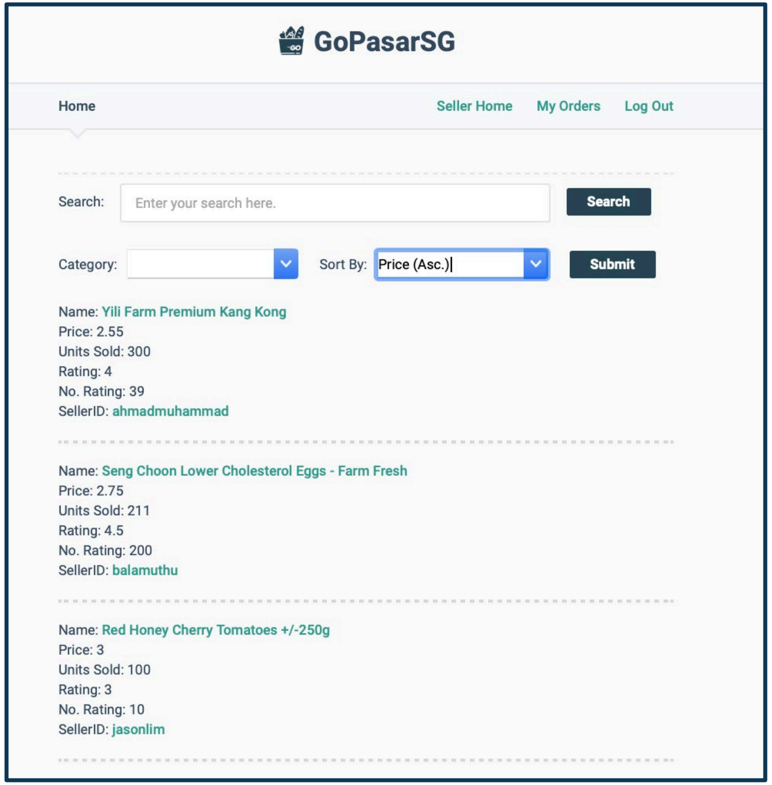
Search: Enter your search here. Search

Category: Sort By: Popular Submit

Name: White Rabbit White Ale Beer ~ P2
Price: 7
Units Sold: 3008
Rating: 4.8
No. Rating: 300
SellerID: leematthew

Name: ROAD HOG SESSION IPA (24Pack)
Price: 85
Units Sold: 488
Rating: 4.3
No. Rating: 273
SellerID: leematthew

Name: Red Onion (3kg)
Price: 6.3
Units Sold: 400
Rating: 4.5
No. Rating: 200
SellerID: ahmadmuhammad


GoPasarSG

Home

Seller Home My Orders Log Out

Search: Enter your search here. Search

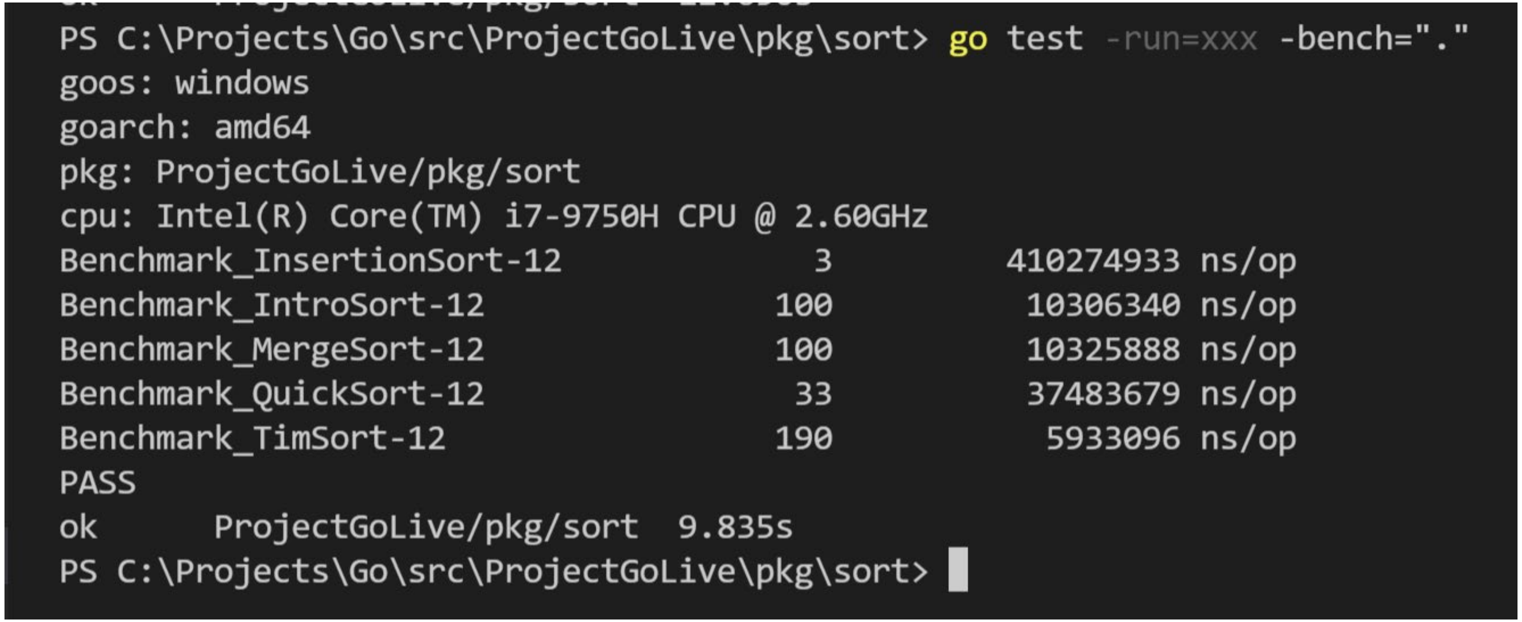
Category: Sort By: Price (Asc.) Submit

Name: Yili Farm Premium Kang Kong
Price: 2.55
Units Sold: 300
Rating: 4
No. Rating: 39
SellerID: ahmadmuhammad

Name: Seng Choon Lower Cholesterol Eggs - Farm Fresh
Price: 2.75
Units Sold: 211
Rating: 4.5
No. Rating: 200
SellerID: balamuthu

Name: Red Honey Cherry Tomatoes +/-250g
Price: 3
Units Sold: 100
Rating: 3
No. Rating: 10
SellerID: jasonlim

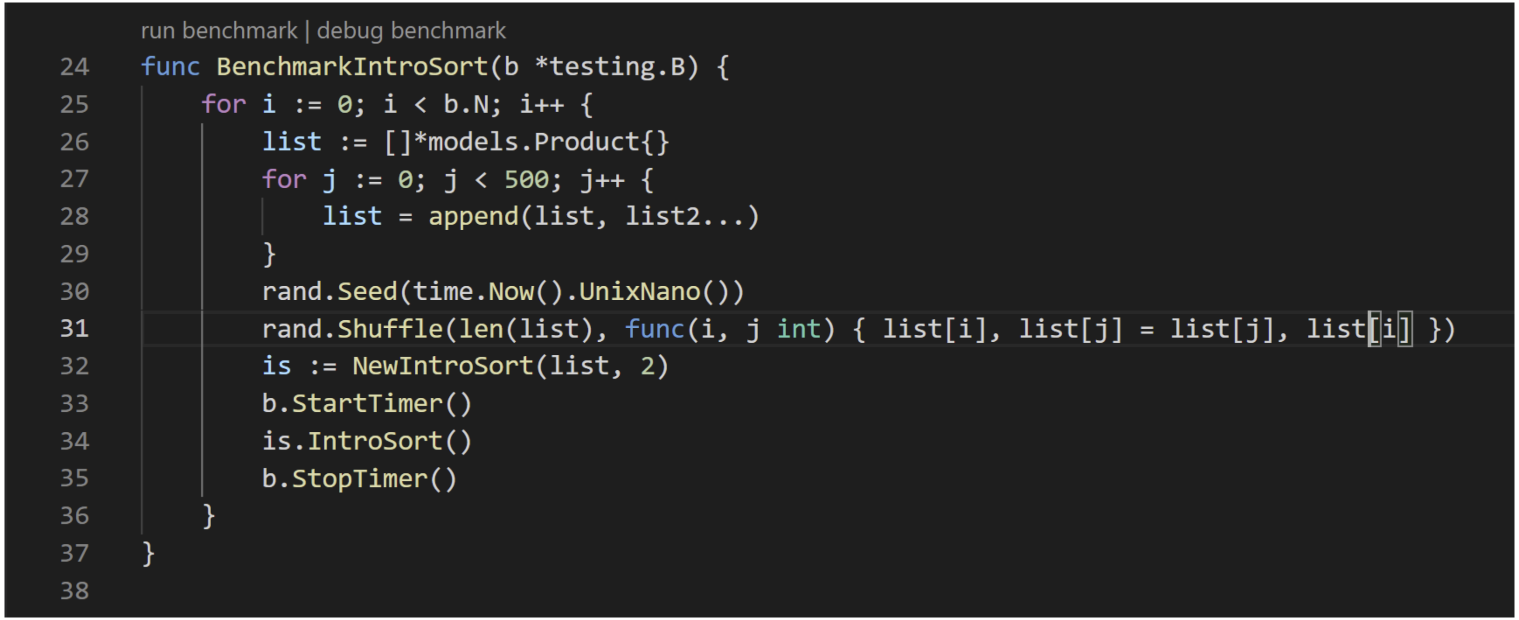
Sorting: Benchmark Testing



```
PS C:\Projects\Go\src\ProjectGoLive\pkg\sort> go test -run=xxx -bench="."
goos: windows
goarch: amd64
pkg: ProjectGoLive/pkg/sort
cpu: Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz
Benchmark_InsertionSort-12            3         410274933 ns/op
Benchmark_IntroSort-12              100          10306340 ns/op
Benchmark_MergeSort-12              100          10325888 ns/op
Benchmark_QuickSort-12               33          37483679 ns/op
Benchmark_TimSort-12                190           5933096 ns/op
PASS
ok      ProjectGoLive/pkg/sort  9.835s
PS C:\Projects\Go\src\ProjectGoLive\pkg\sort>
```

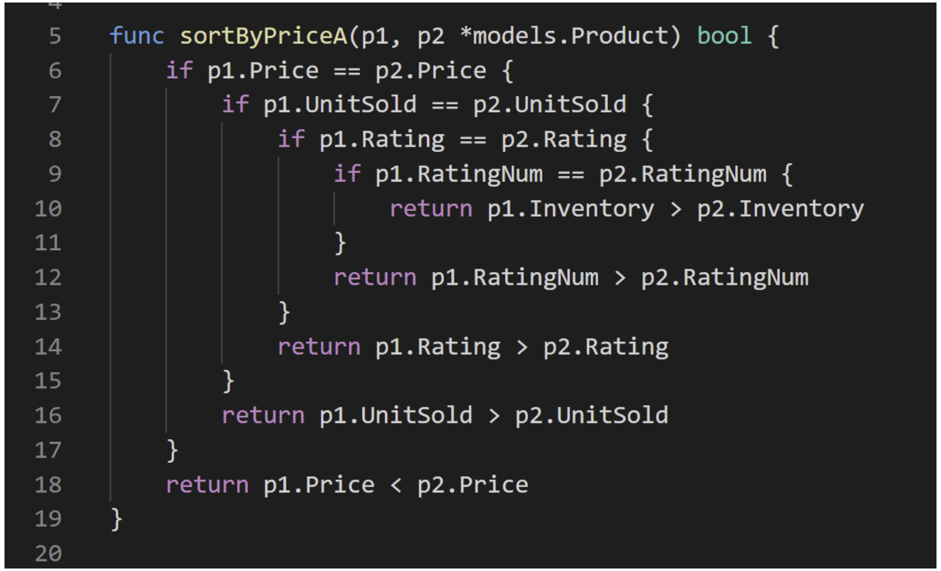
Sorting: Benchmark Testing

Anomaly in the results was caused by the way the test was set up

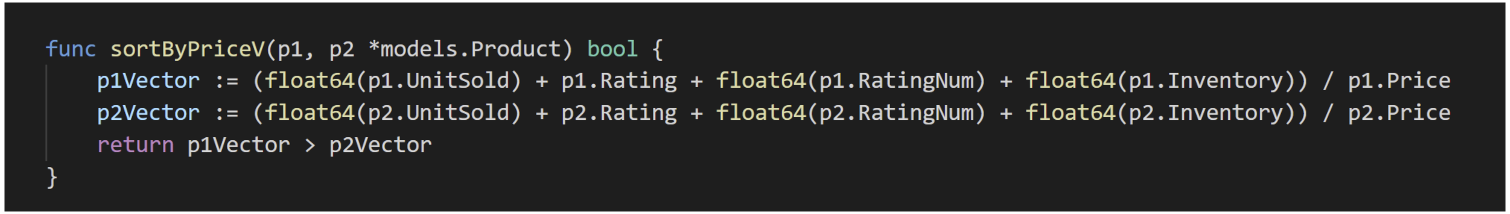


```go
func BenchmarkIntroSort(b *testing.B) {
    for i := 0; i < b.N; i++ {
        list := []*models.Product{}
        for j := 0; j < 500; j++ {
            list = append(list, list2...)
        }
        rand.Seed(time.Now().UnixNano())
        rand.Shuffle(len(list), func(i, j int) { list[i], list[j] = list[j], list[i] })
        is := NewIntroSort(list, 2)
        b.StartTimer()
        is.IntroSort()
        b.StopTimer()
    }
}
```

Sorting: Further Improvements?

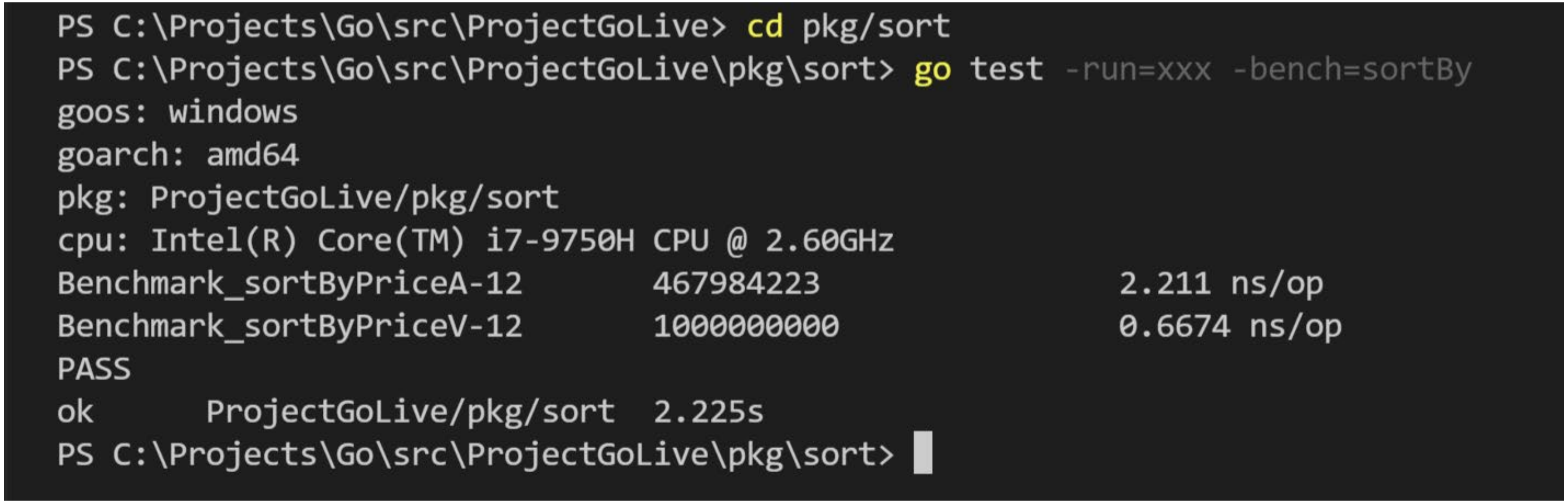


```go
func sortByPriceA(p1, p2 *models.Product) bool {
    if p1.Price == p2.Price {
        if p1.UnitSold == p2.UnitSold {
            if p1.Rating == p2.Rating {
                if p1.RatingNum == p2.RatingNum {
                    return p1.Inventory > p2.Inventory
                }
                return p1.RatingNum > p2.RatingNum
            }
            return p1.Rating > p2.Rating
        }
        return p1.UnitSold > p2.UnitSold
    }
    return p1.Price < p2.Price
}
```



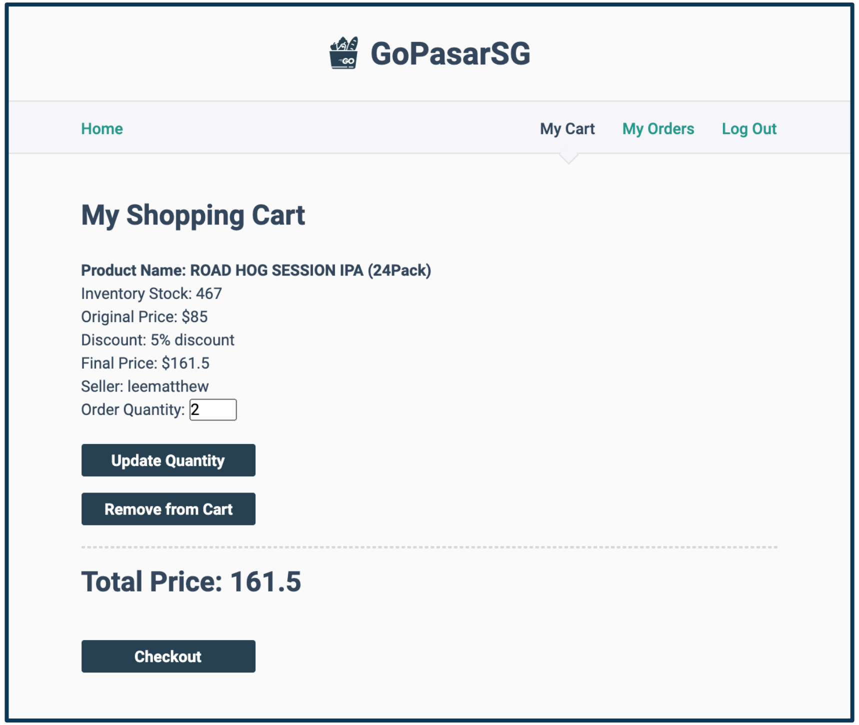
```go
func sortByPriceV(p1, p2 *models.Product) bool {
    p1Vector := (float64(p1.UnitSold) + p1.Rating + float64(p1.RatingNum) + float64(p1.Inventory)) / p1.Price
    p2Vector := (float64(p2.UnitSold) + p2.Rating + float64(p2.RatingNum) + float64(p2.Inventory)) / p2.Price
    return p1Vector > p2Vector
}
```

Sorting: Further Improvements?


```
PS C:\Projects\Go\src\ProjectGoLive> cd pkg/sort
PS C:\Projects\Go\src\ProjectGoLive\pkg\sort> go test -run=xxx -bench=sortBy
goos: windows
goarch: amd64
pkg: ProjectGoLive/pkg/sort
cpu: Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz
Benchmark_sortByPriceA-12       467984223                2.211 ns/op
Benchmark_sortByPriceV-12       1000000000               0.6674 ns/op
PASS
ok      ProjectGoLive/pkg/sort  2.225s
PS C:\Projects\Go\src\ProjectGoLive\pkg\sort>
```

Transaction Facilitation

Shopping Cart

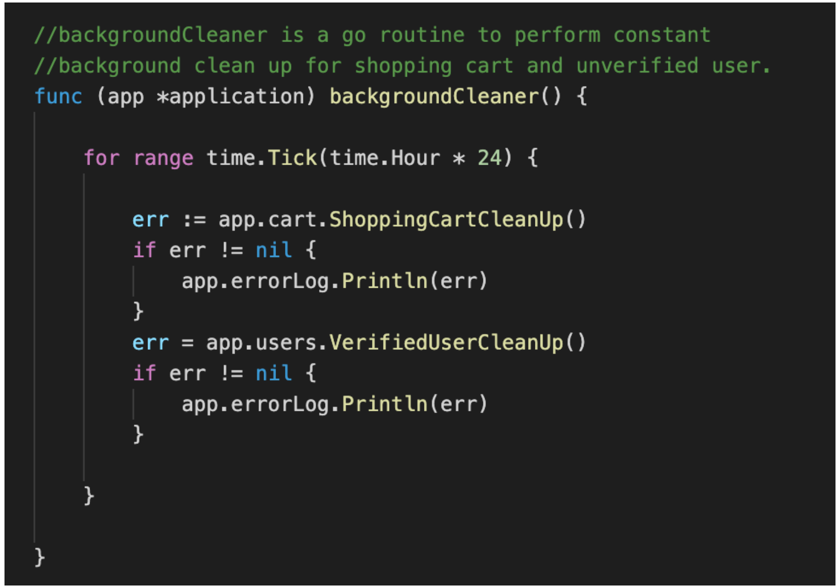


GoPasarSG

My Shopping Cart

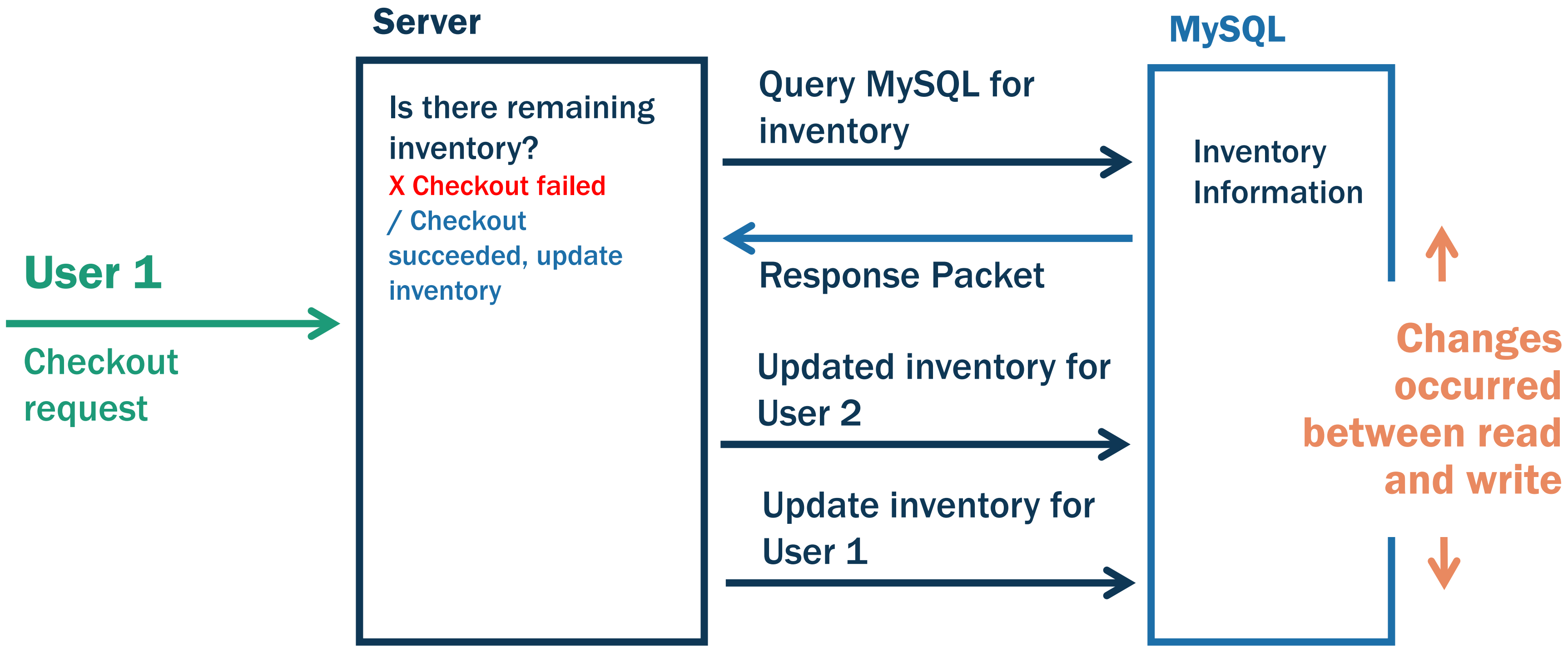
Product Name: ROAD HOG SESSION IPA (24Pack)
Inventory Stock: 467
Original Price: $85
Discount: 5% discount
Final Price: $161.5
Seller: leematthew
Order Quantity: 2

Total Price: 161.5

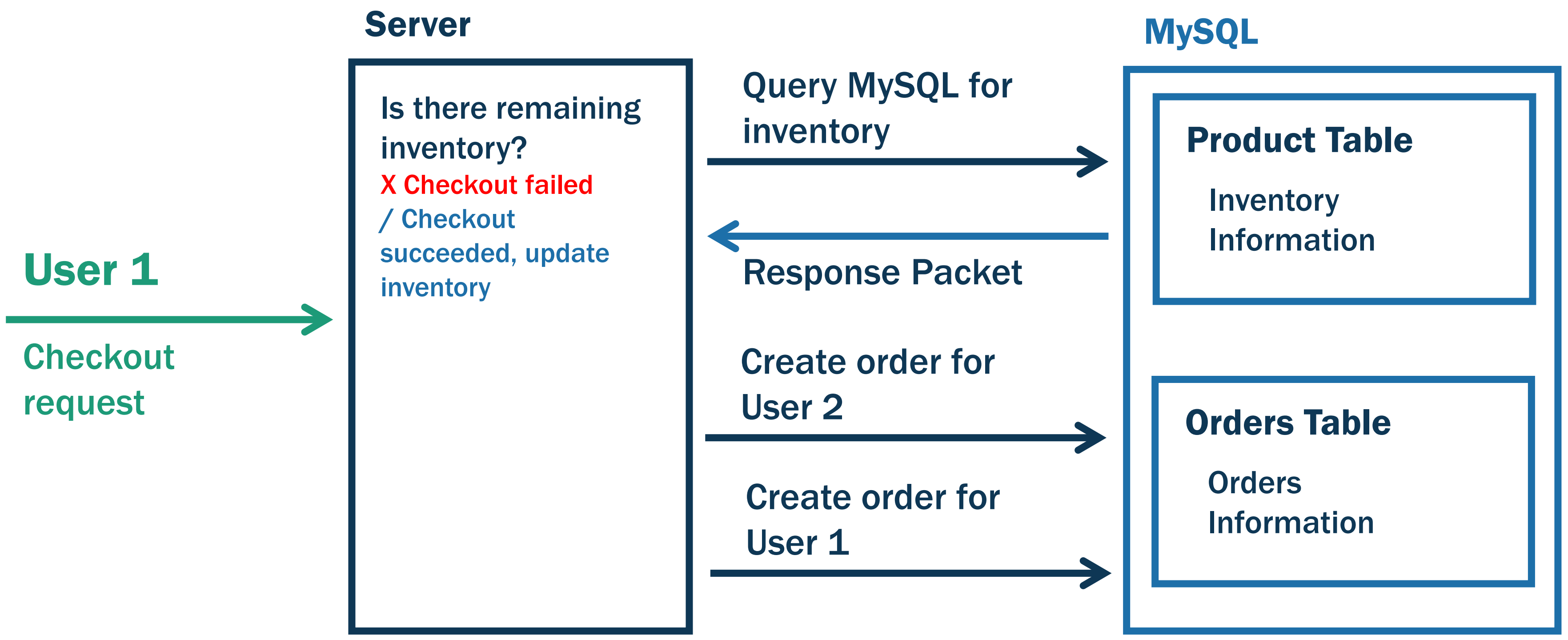


```go
//backgroundCleaner is a go routine to perform constant
//background clean up for shopping cart and unverified user.
func (app *application) backgroundCleaner() {

    for range time.Tick(time.Hour * 24) {

        err := app.cart.ShoppingCartCleanUp()
        if err != nil {
            app.errorLog.Println(err)
        }
        err = app.users.VerifiedUserCleanUp()
        if err != nil {
            app.errorLog.Println(err)
        }

    }

}
```

Checkout = Make Purchase

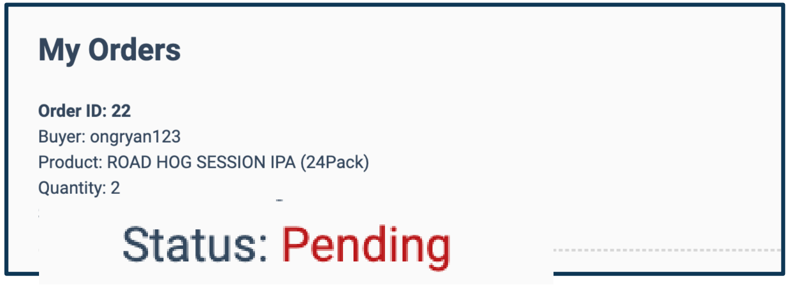

Server
MySQL
Is there remaining inventory?
X Checkout failed
/ Checkout succeeded, update inventory
Query MySQL for inventory
Inventory Information
Response Packet
User 1
Checkout request
Updated inventory for User 2
Update inventory for User 1
Changes occurred between read and write

Checkout = Order Created



Checkout + Seller Confirmation

Buyer

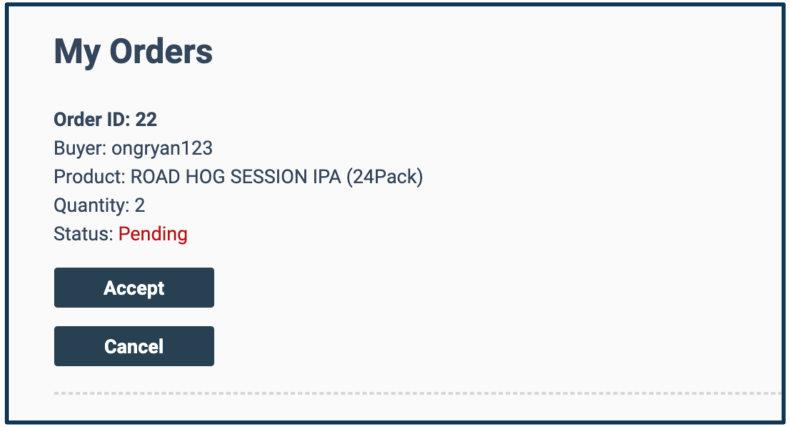


My Orders

Order ID: 22
Buyer: ongryan123
Product: ROAD HOG SESSION IPA (24Pack)
Quantity: 2

Status: Pending

Seller



My Orders

Order ID: 22
Buyer: ongryan123
Product: ROAD HOG SESSION IPA (24Pack)
Quantity: 2
Status: Pending

Limitations

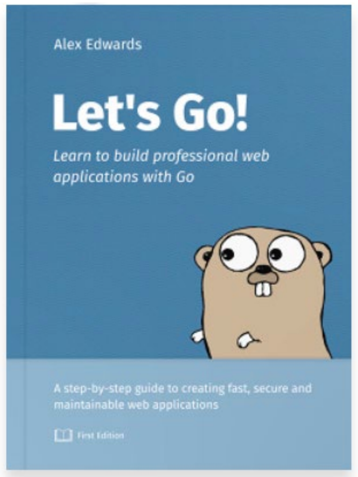
No. of read and write to the MySQL

Unpleasant user experience if order is cancelled

Possible Improvement

Use database cache such as Redis as buffer layer

Technology & References



Alex Edwards

Let's Go!

Learn to build professional web applications with Go

A step-by-step guide to creating fast, secure and maintainable web applications

First Edition

- github.com/bbalet/stopwords v1.0.0
- github.com/go-sql-driver/mysql v1.6.0
- github.com/golangcollege/sessions v1.2.0
- github.com/gorilla/mux v1.8.0
- github.com/joho/godotenv v1.3.0
- github.com/justinas/alice v1.2.0
- github.com/kljensen/snowball v0.6.0
- golang.org/x/crypto v0.0.0-20200317142112-1b76d66859c6

Thank You!