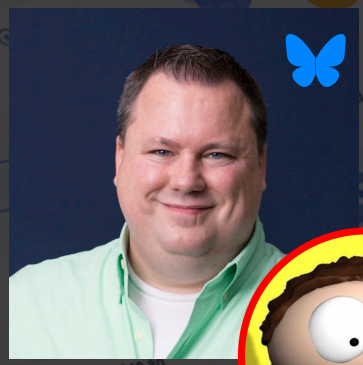
Front to Back

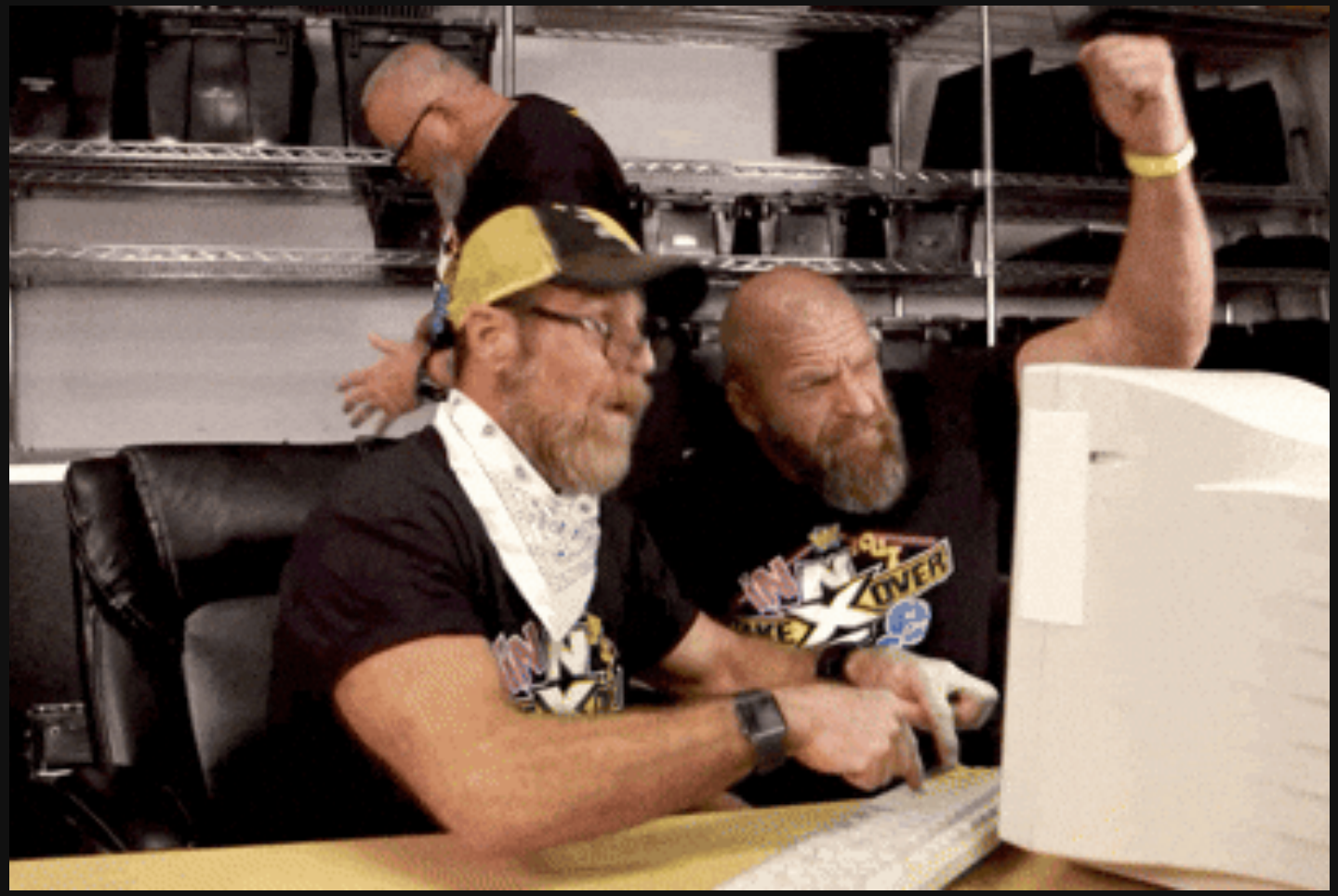
Never Back to Front



adamlbarrett.bsky.social

BigAB

Software Development



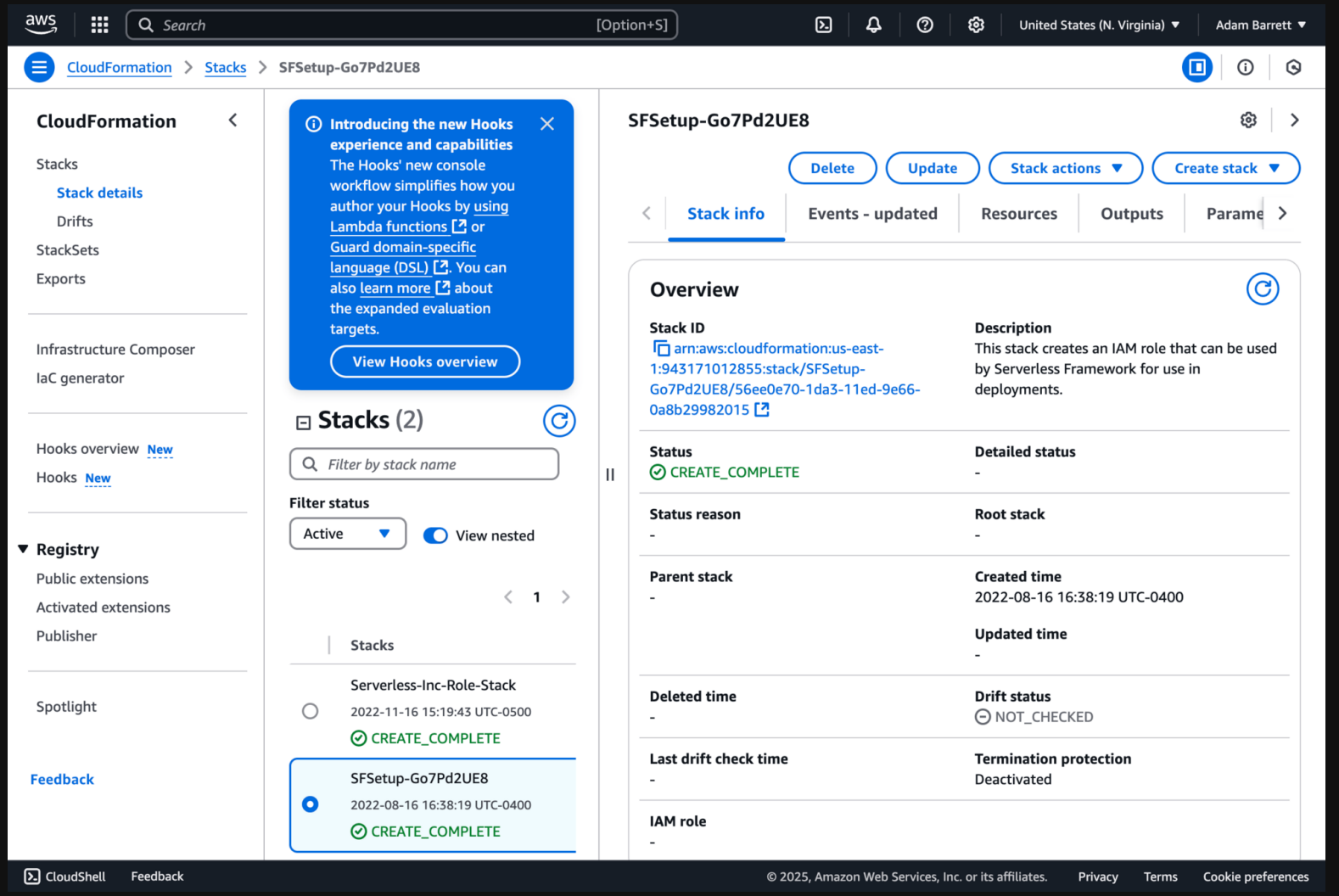
Big Companies and Small

PEBCAK


Front to Back
Never Back to Front

Never Back to Front

AWS Console


aws
Search
[Option+S]
United States (N. Virginia)
Adam Barrett
CloudFormation > Stacks > SFSetup-Go7Pd2UE8
CloudFormation
Stacks
Stack details
Drifts
StackSets
Exports
Infrastructure Composer
IaC generator
Hooks overview New
Hooks New
Registry
Public extensions
Activated extensions
Publisher
Spotlight
Feedback
Introducing the new Hooks experience and capabilities
The Hooks' new console workflow simplifies how you author your Hooks by using Lambda functions or Guard domain-specific language (DSL). You can also learn more about the expanded evaluation targets.
View Hooks overview
Stacks (2)
Filter by stack name
Filter status
Active
View nested
1
Stacks
Serverless-Inc-Role-Stack
2022-11-16 15:19:43 UTC-0500
CREATE_COMPLETE
SFSetup-Go7Pd2UE8
2022-08-16 16:38:19 UTC-0400
CREATE_COMPLETE
SFSetup-Go7Pd2UE8
Delete
Update
Stack actions
Create stack
Stack info
Events - updated
Resources
Outputs
Parame
Overview
Stack ID
arn:aws:cloudformation:us-east-1:943171012855:stack/SFSetup-Go7Pd2UE8/56ee0e70-1da3-11ed-9e66-0a8b29982015
Description
This stack creates an IAM role that can be used by Serverless Framework for use in deployments.
Status
CREATE_COMPLETE
Detailed status
-
Status reason
-
Root stack
-
Parent stack
-
Created time
2022-08-16 16:38:19 UTC-0400
Updated time
-
Deleted time
-
Drift status
NOT_CHECKED
Last drift check time
-
Termination protection
Deactivated
IAM role
-
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


It's happening.
It's happening.
peacock

Submit to whom?

The button with the text "submit", though a known design anti-pattern, is so common that people have no problem with it, but it is still a good example of how thinking of the implementation works its way forward to the User Experience

Username:

Email:

Password:

Submit

When Backend Developer
work as Frontend Developer

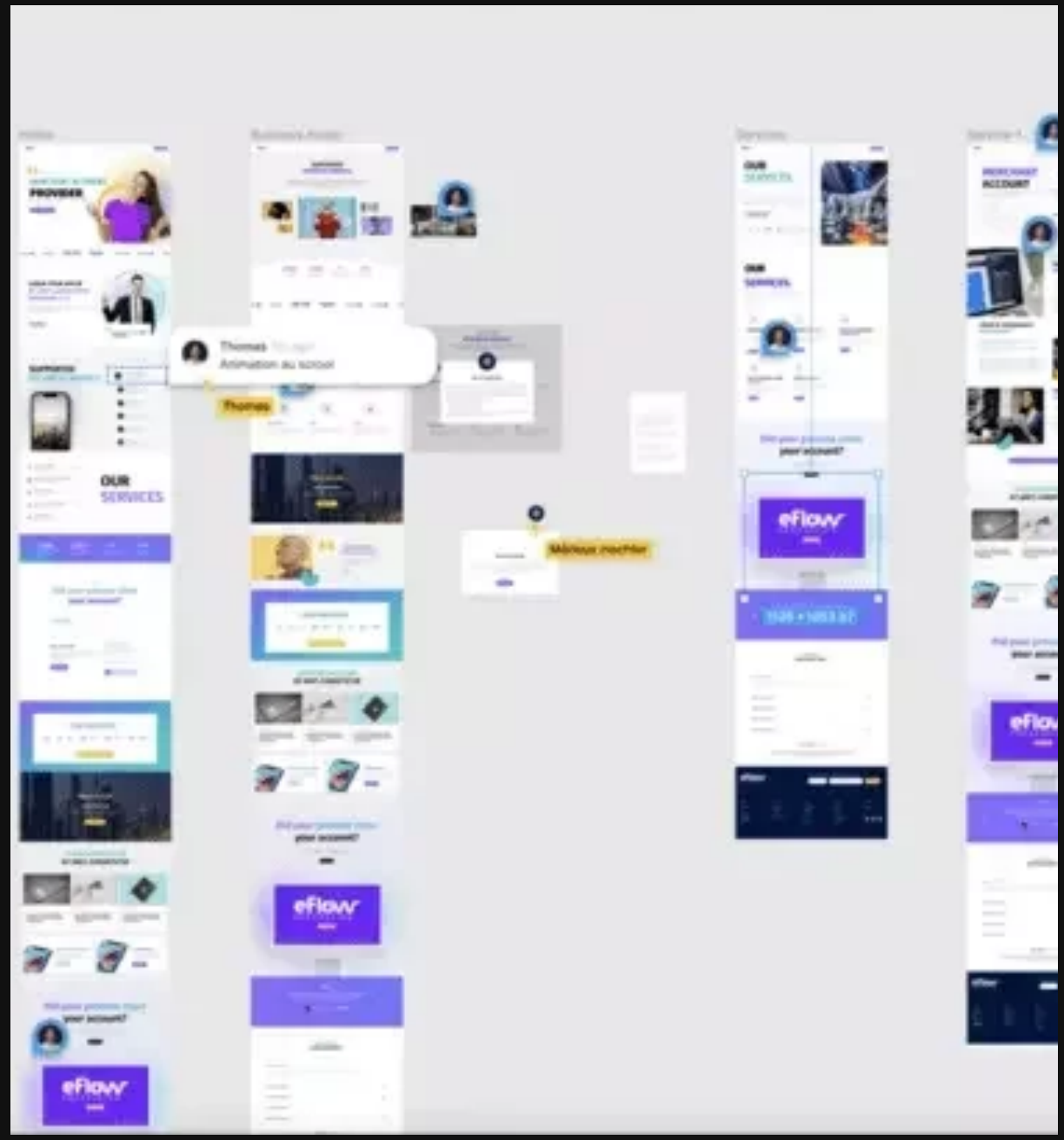
😅😅

So how do we do this?

(the whole front-to-back thing)

Ask yourself...

- What can the user see?
- What can the user do?



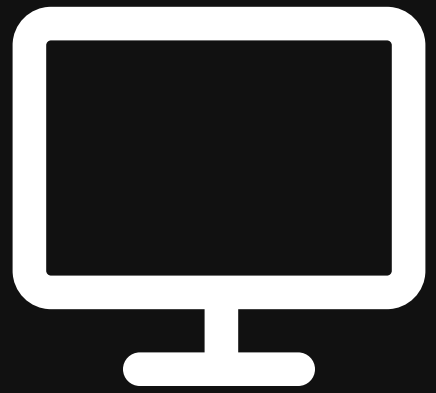
What can the user see?

Where can I get the info I need?

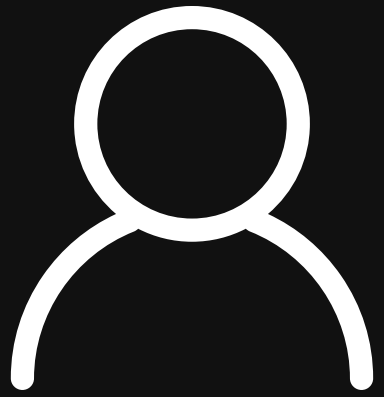
What can the user do?

What are all the different interactions?

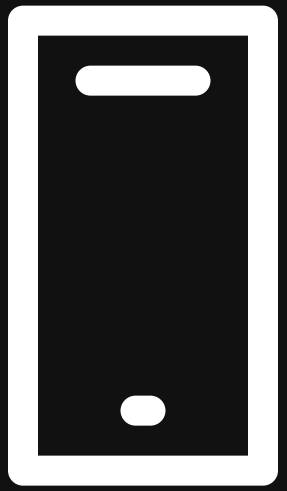
UI Events



User Interactions



System Actions



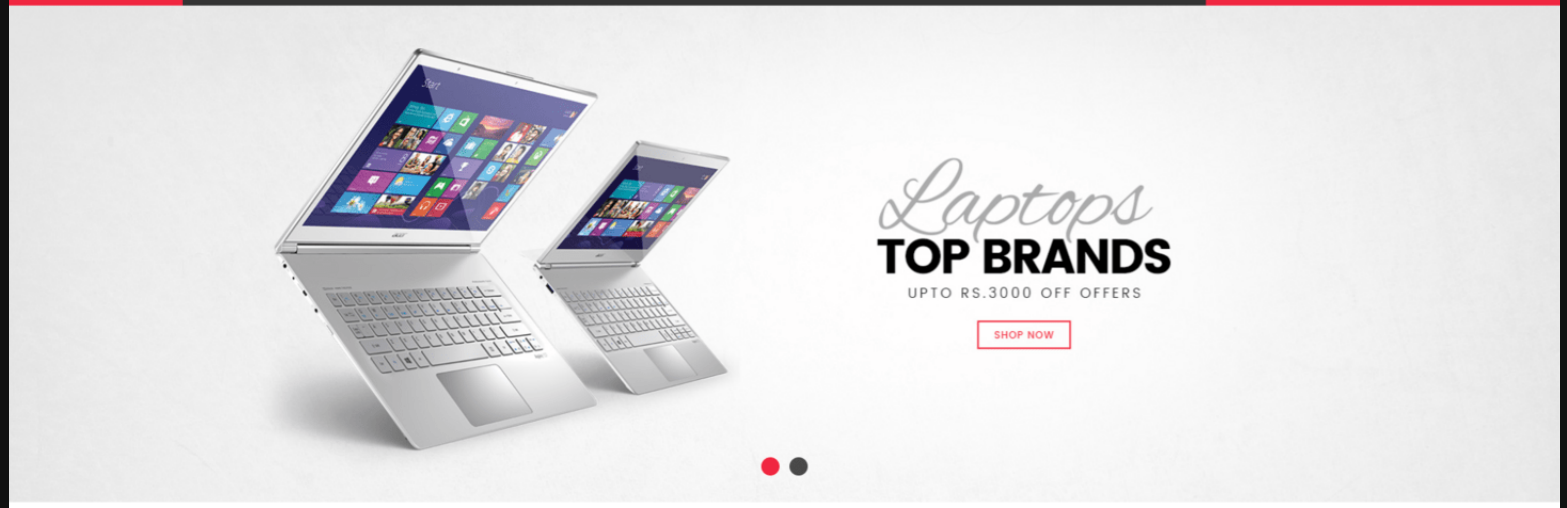
WELCOME TO SHOPTOWN ONLINE SHOPPING STORE. LANGUAGE | $ CURRENCY | MY ACCOUNT

SHOP Town

Categories | SEARCH PRODUCTS HERE

FREE CALL US
(+91) 800 000 0000

MENU (0)



Free World Delivery
Order Over $100

Win $100 To Shop
Enter Now

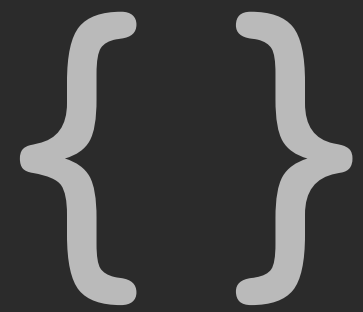
Best Online Support
Hours : 8am - 11pm

Money Back Guarantee
With A 30 Day

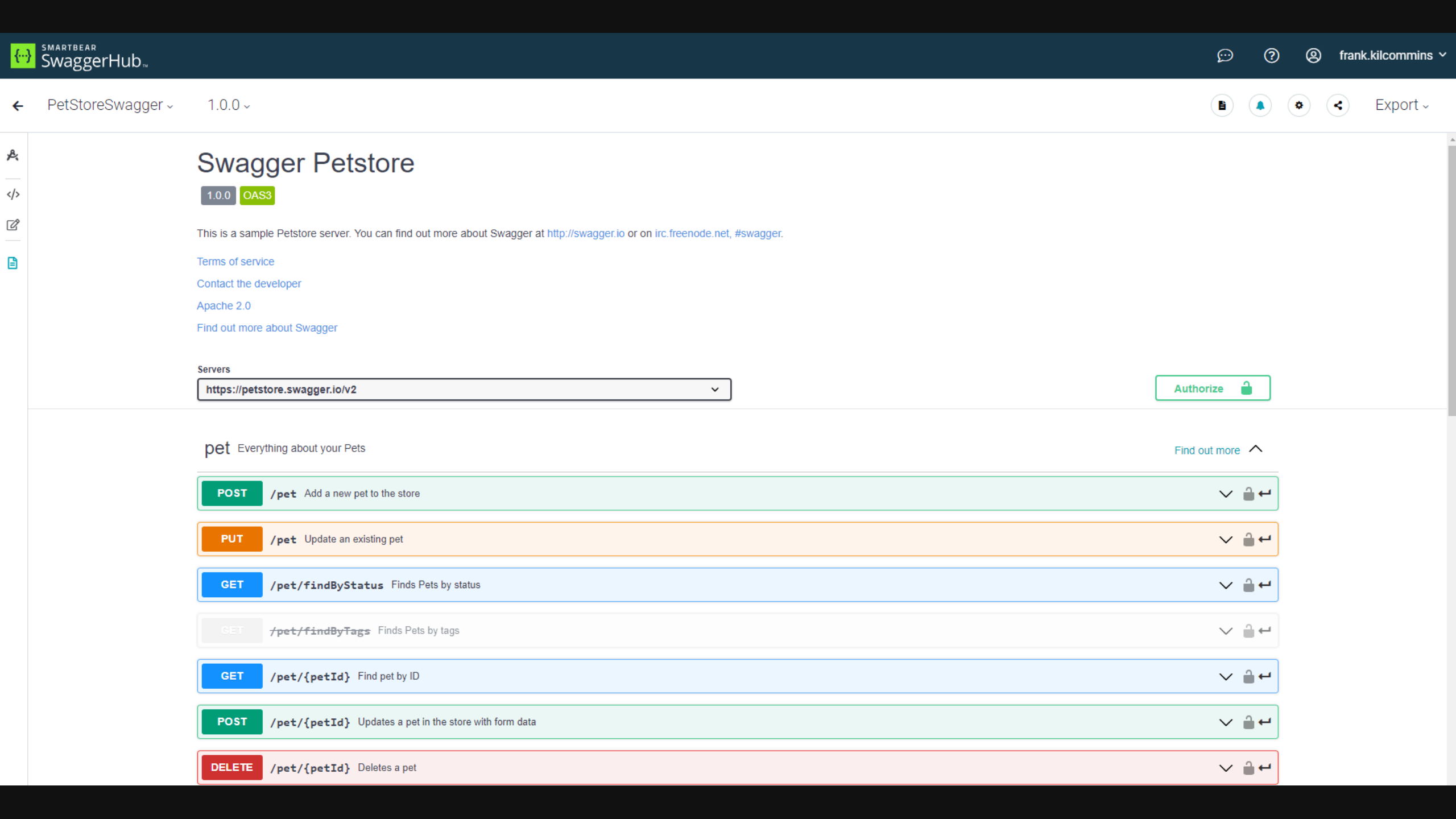
TOP PRODUCTS

LATEST | SPECIAL | BESTSELLER

CRUD APIS SUCK!



```
class Cart {
  addProduct(product, quantity = 1) {/*...*/}

  removeProduct(product) {/*...*/}

  changeQuantity(product, quantity) {/*...*/}

  addPromotionCode(code) {/*...*/}

  subscribe(callback) {/*...*/}

  get data() {/*...*/}
}
```


SMARTBEAR
SwaggerHub
frank.kilcommins
PetStoreSwagger
1.0.0
Export
Swagger Petstore
1.0.0
OAS3
This is a sample Petstore server. You can find out more about Swagger at http://swagger.io or on irc.freenode.net, #swagger.
Terms of service
Contact the developer
Apache 2.0
Find out more about Swagger
Servers
https://petstore.swagger.io/v2
Authorize
pet
Everything about your Pets
Find out more
POST /pet Add a new pet to the store
PUT /pet Update an existing pet
GET /pet/findByStatus Finds Pets by status
GET /pet/findByTags Finds Pets by tags
GET /pet/{petId} Find pet by ID
POST /pet/{petId} Updates a pet in the store with form data
DELETE /pet/{petId} Deletes a pet


SHOP
Town

Schemes

HTTPS

Authorize

Cart Everything about your Carts

Find out more

- POST /cart/add-product
- POST /cart/remove-product
- GET /cart Get this users stored cart
- POST /cart/add-promotion-code Promotion codes generally lead to discounts
- DELETE /cart/{cartId} Deletes a Cart

store Access to Petstore orders

- GET /store/inventory Returns pet inventories by status
- POST /store/order Place an order for a pet
- GET /store/order/{orderId} Find purchase order by ID

What can the user see?

What can the user do?

What data do I need?

What actions can I take?

What data do they need?

What actions can they take?

When designing modules try and remember the 80/20 rule of design. Cover the most common actions with direct functions or methods, but possible allow for more complicated actions too

```
1  export function addCourseToSchedule {
2    /*..*/
3  }
4
5  export function beginCourse {
6    /*..*/
7  }
8
9
10 export function abandonCourse {
11   /*..*/
12 }
13
14
15 export function singUpForWaitingList
16   /*..*/
17 }
```

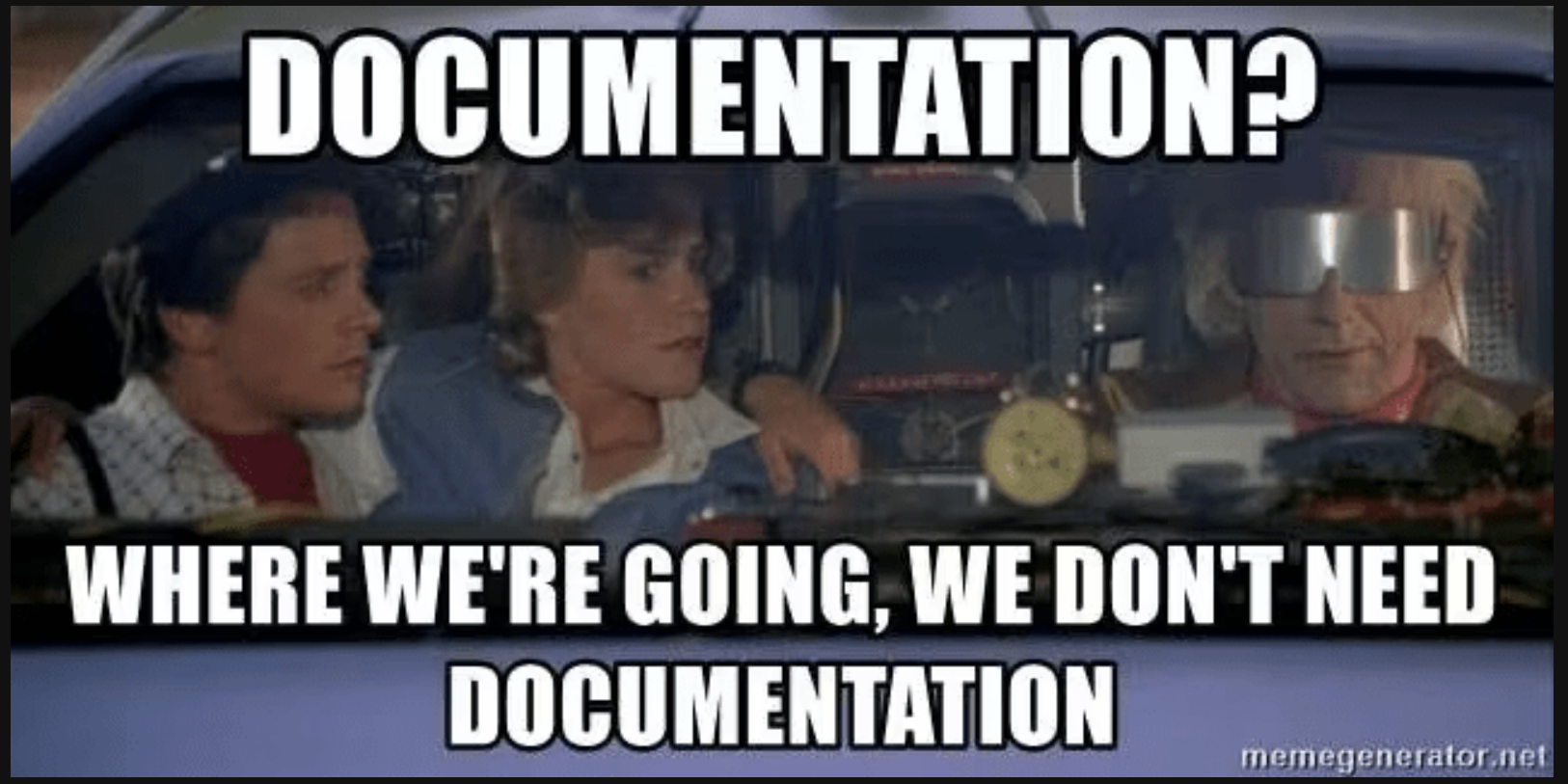

1
RED
Write a test
that fails
TDD
2
GREEN
Make the
code work
3
REFACTOR
Eliminate
redundancy

Behaviour Driven Development

(BDD)

Documentation Driven Development

(DDD)



What data do they need?

- List of Houses
- Available Filters
- Agent info

What actions can they take?

- change filters
- see house details
- contact agent

Tournament Planning

Team Editor

Bacon United

Dora Paulsen
Keeper

Ashtyn Sneed
striker

Randi Erickson
center back

Areli Schwartz
left-back

Kidneypool

Tori Riggins
Keeper

Hillary Tabor
defensive midfielder

Odalys Hickey
center back

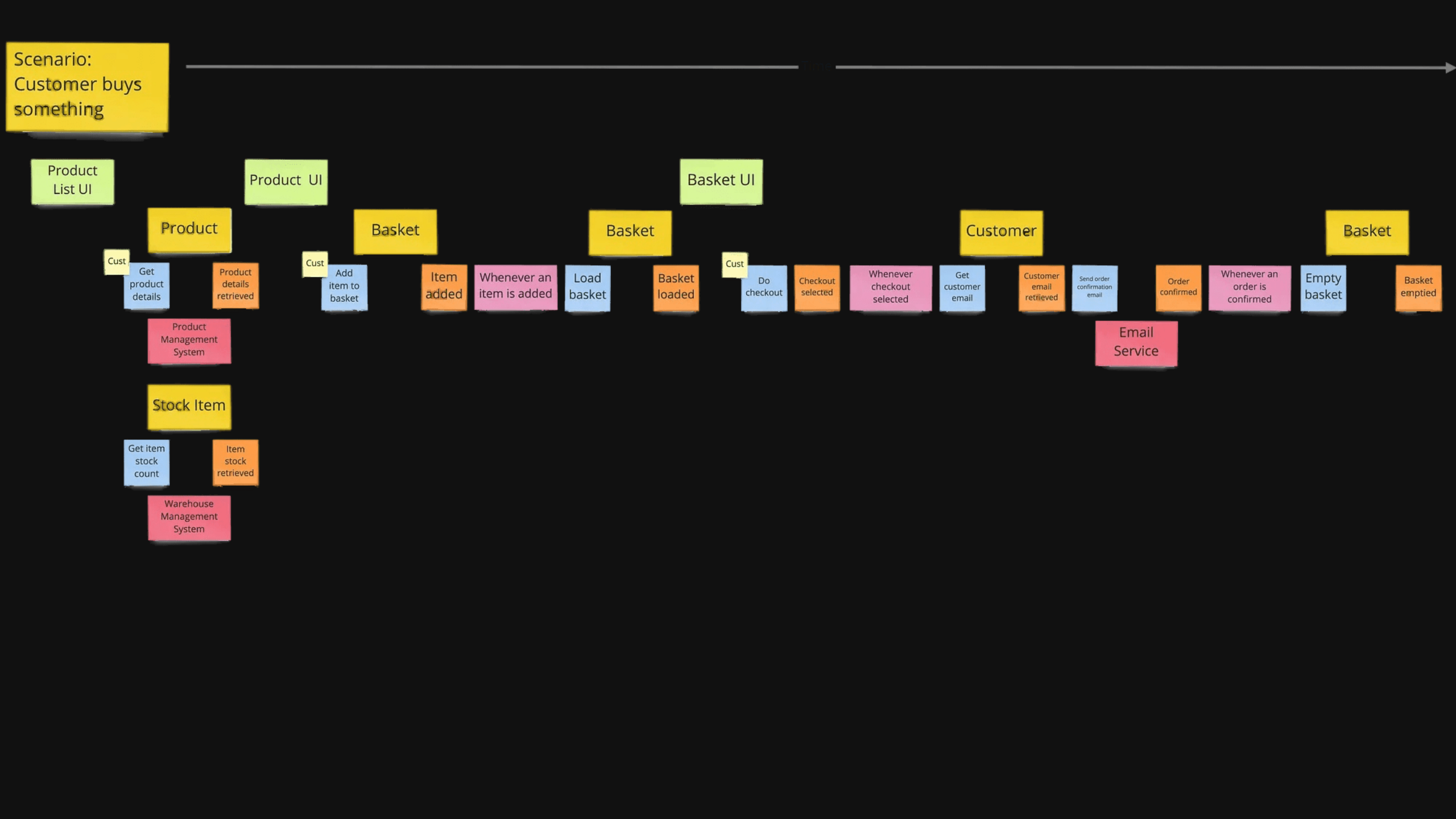
Margaret Carver
striker

Northampton

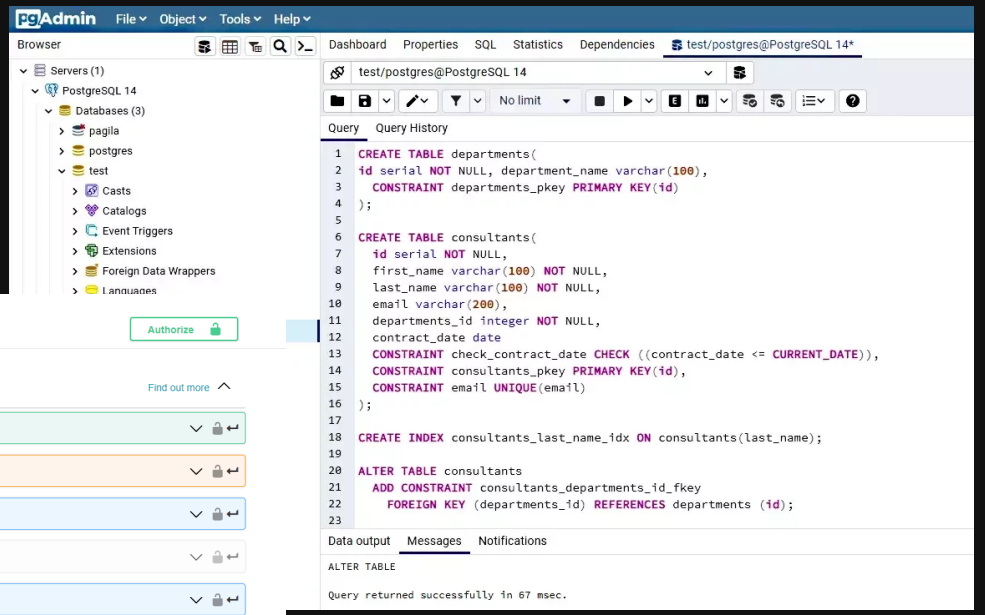
Eryn Bateman
Keeper

Gretchen Fortner
center back

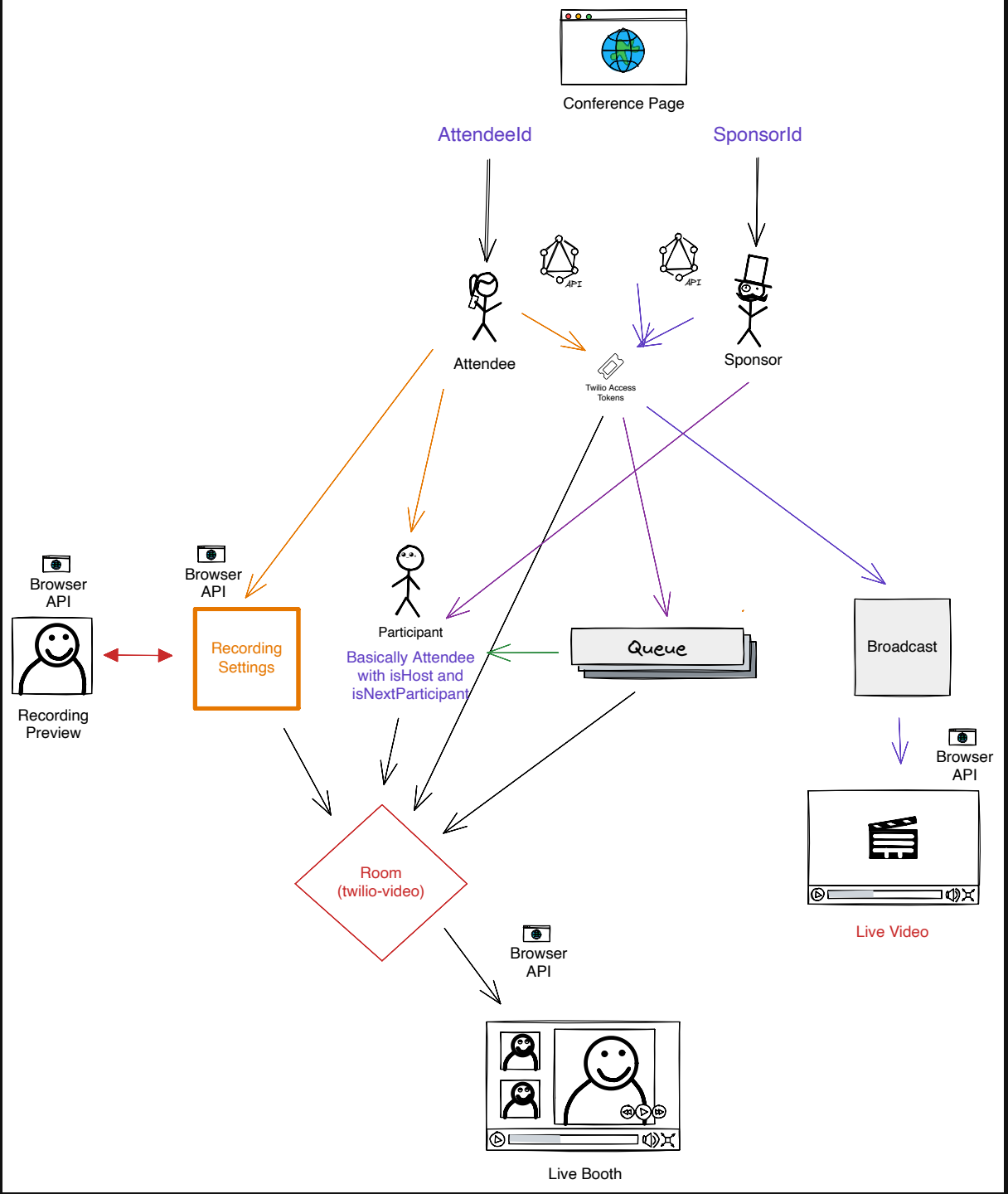
Isla Barrett
midfielders


Scenario: Customer buys something
Time
Product List UI
Product UI
Basket UI
Product
Basket
Basket
Customer
Basket
Cust
Get product details
Product details retrieved
Product Management System
Cust
Add item to basket
Item added
Whenever an item is added
Load basket
Basket loaded
Cust
Do checkout
Checkout selected
Whenever checkout selected
Get customer email
Customer email retrieved
Send order confirmation email
Email Service
Order confirmed
Whenever an order is confirmed
Empty basket
Basket emptied
Stock Item
Get item stock count
Item stock retrieved
Warehouse Management System

Your REST APIs don't have to mirror your DB tables....



Think before coding...



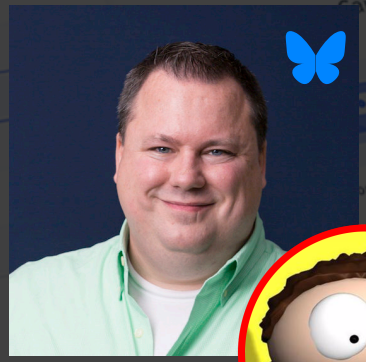
Front-To-Back

- Consumer First Thinking
- What data do you need?
- What actions can you take?

Front to Back

Never Back to Front

slides.com/bigab



adamlbarrett.bsky.social

BigAB