Folio

by Shaheen Ghiassy

Web	Monthly Ad Campaigns	Pg. 3
UI/UX	Website Redesign	Pg. 4
3D	Winter Cottage	Pg. 5
3D	Behind the Scenes	Pg. 6
Print	Consumer Catalog	Pg. 7
Print	Commercial Catalog	Pg. 8
UX	Mobile Merchant UX	Pg. 9
Mobile	Mobile Merchant	Pg. 10
Web	Paintable Business Hours	Pg. 11
Web	Pre-purchase Availabilities	Pg. 12
Web	S3 Integration	Pg. 13
Mobile	Holiday Gift Shop	Pg. 14

Web Design

ROLE Design Director, Co-Designer

TECHNOLOGIES USED Photoshop, Illustrator

SUMMARY Directed and led monthly advertising campaigns with original artwork. Recruited, developed and trained design staff.

WALLBEDFACTORY.COM
AN OLD CREEK COMPANY

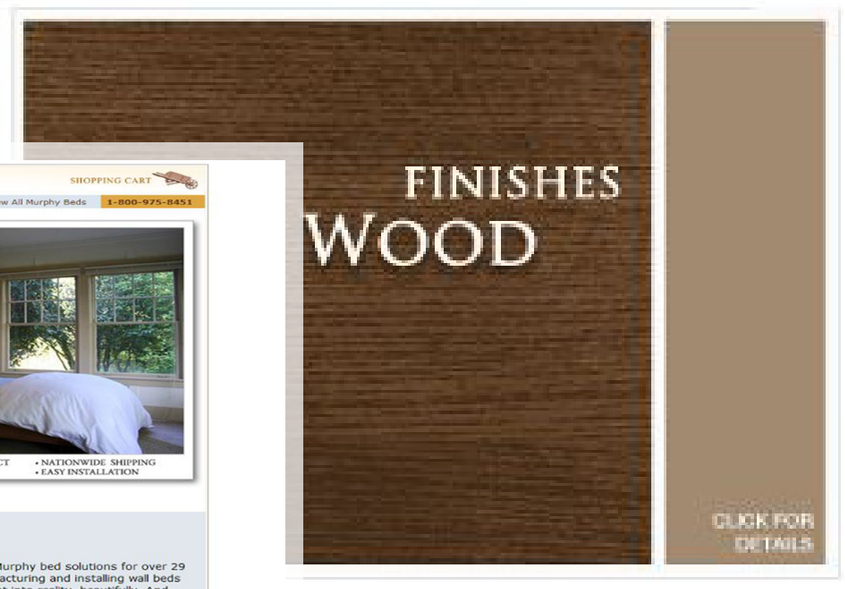
Catalog Request | Company Blog | Real-Time Tracking

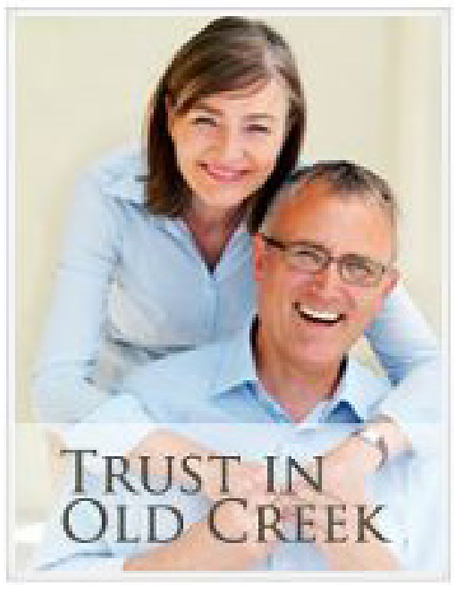
0 item(s) in cart | Checkout
My Account | My Wishlist | Login
SEARCH KEYWORD OR ITEM # GO

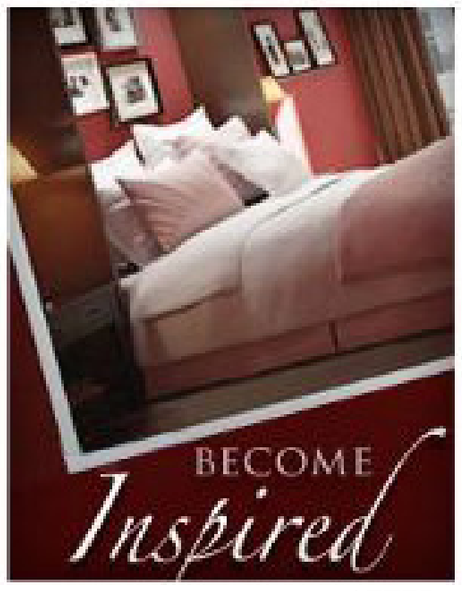
PRODUCTS | PLANNING | ORDERING | RESOURCE CENTER | COMMUNITY | OUR COMPANY

ORDER ANY MURPHY BED AND SAVE $150. ORDER OVER $2500 AND GET EVEN MORE. SUMMER TIME SALE LEARN MORE

BEFORE







AFTER

ROLE DESIGN Director, Engineering Director, Project Manager, Co-Designer

TECHNOLOGIES USED OmniGraffle, Photoshop, Illustrator, CrazyEgg, Google Analytics

SUMMARY Directed and led website redesign. Responsible for all technical and design aspects. Managed project from design inspiration, to technical development. Followed-through with SEO and conversion optimization.

UX

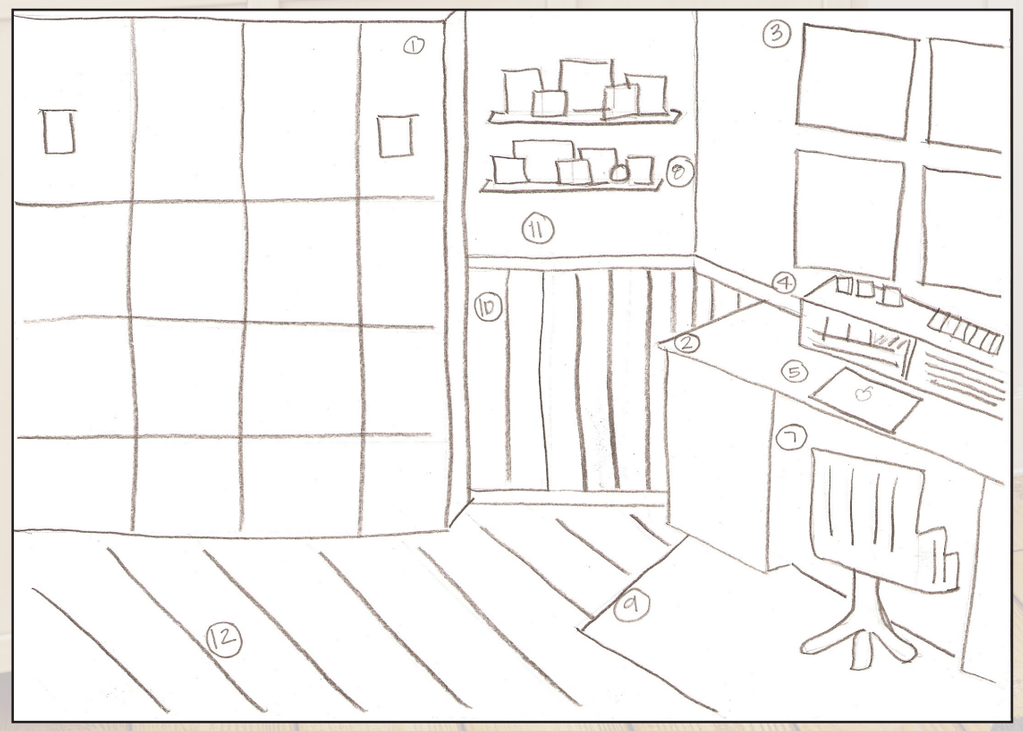
ROLE Design Director

TECHNOLOGIES USED AutoDesk 3DS

SUMMARY Led 3D design initiative. On a tight budget, found and recruited international team to develop 3D renderings. Led project from inspiration and coordination to final rendering.

3D

3D



ORIGINAL SKETCH

ROLE Design Director

TECHNOLOGIES USED AutoDesk 3DS

SUMMARY Led 3D design initiative. On a tight budget, found and recruited international team to develop 3D renderings. Led project from inspiration and coordination to final rendering.

Print

ROLE Design Director, Project Manager, Co-Designer

TECHNOLOGIES USED InDesign, Photoshop, Illustrator, Autodesk 3DS Max

SUMMARY Managed and directed consumer catalog creation from original inspiration, design curation and development to the printing presses.

PRINTER Bridgetown Printing

Print

ROLE Design Director, Project Manager, Co-Designer

TECHNOLOGIES USED InDesign, Photoshop, Illustrator, Autodesk 3DS Max

SUMMARY Managed and directed commercial catalog creation from original inspiration, design curation and development to the printing presses.

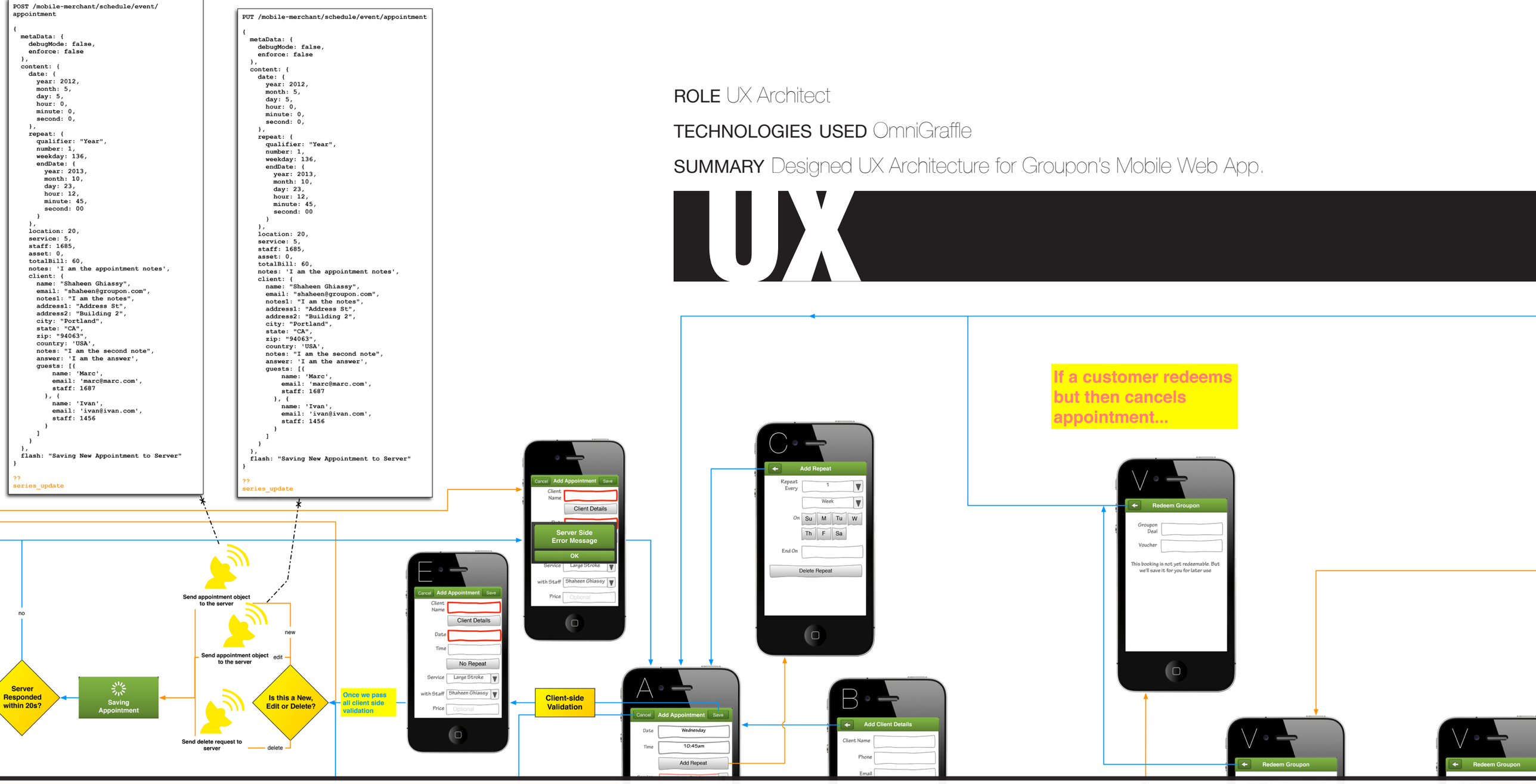
PRINTER B&B Printers

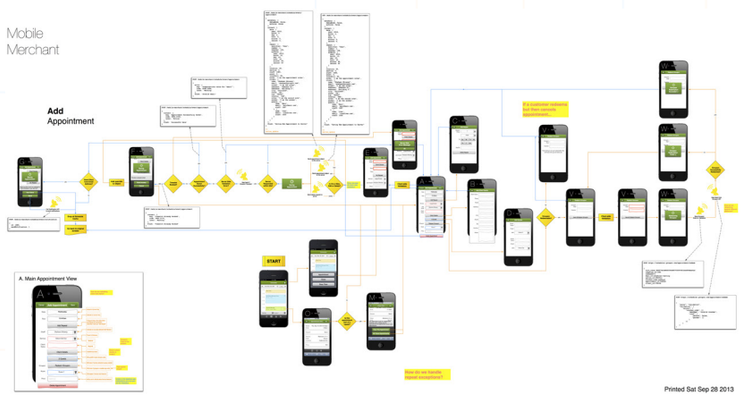
ROLE UX Architect

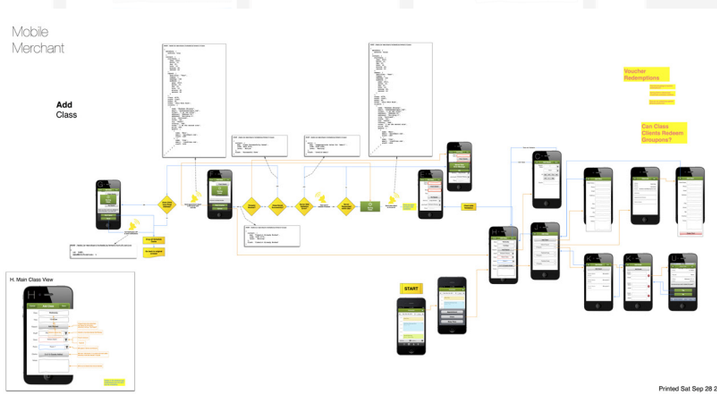
TECHNOLOGIES USED OmniGraffle

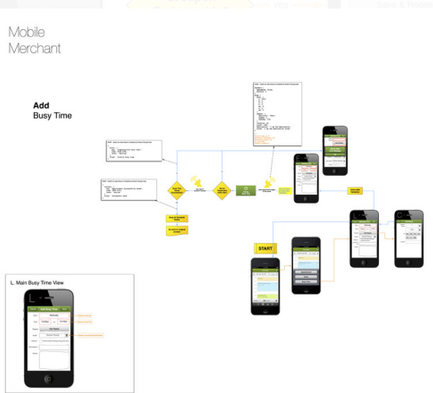
SUMMARY Designed UX Architecture for Groupon's Mobile Web App.

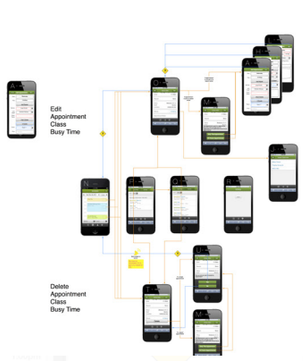
UX

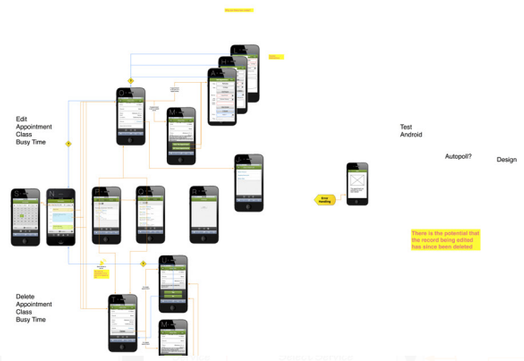


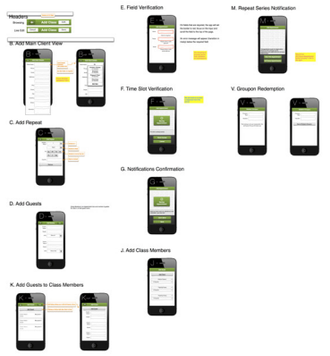




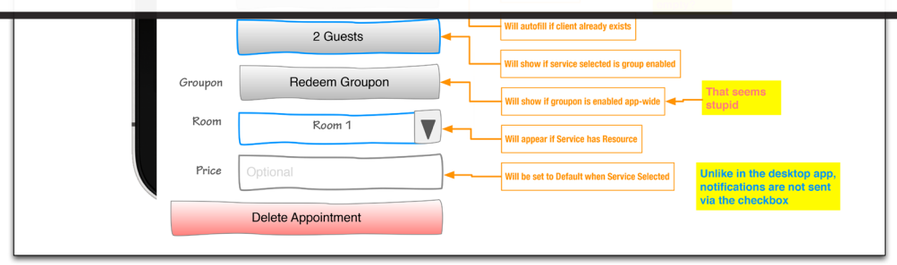








FULL VIEW



ROLE Front-end Web Engineer

TECHNOLOGIES USED HTML5, JavaScript, CSS, Ember.js

SUMMARY Programmed Mobile Web App for Groupon Scheduler Merchants. Solely programmed entire application using advanced client-side technologies.

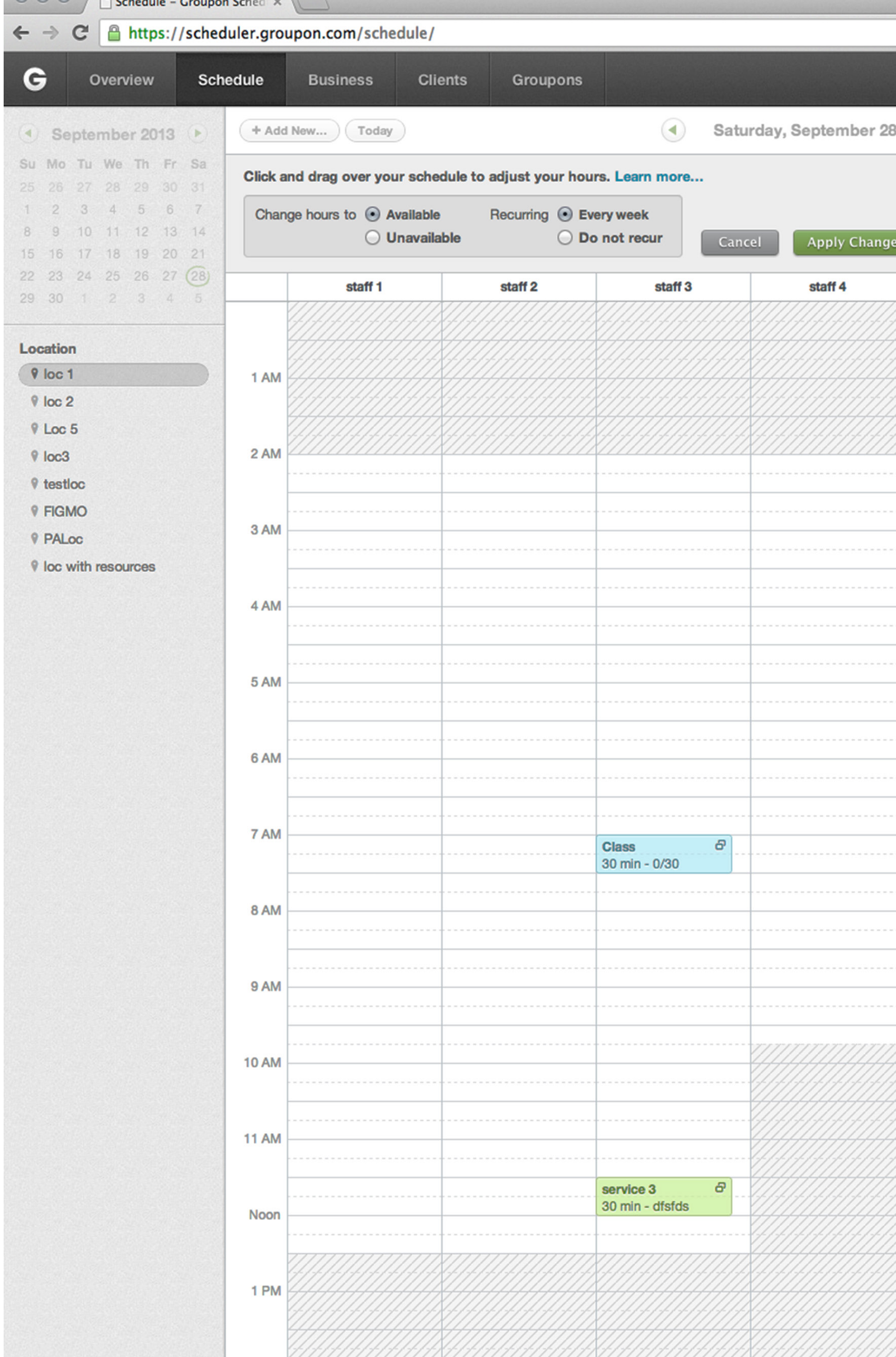
Mobile Web

Web

ROLE Full-stack Web Engineer

TECHNOLOGIES USED HTML5, JavaScript, CSS3, jQuery, PHP, CodeIgnitor, MySQL

SUMMARY Programmed advanced UI functionality into Groupon's Scheduler web application. Coded functionality across the entire web-stack, from HTML & CSS, to PHP & MySQL.



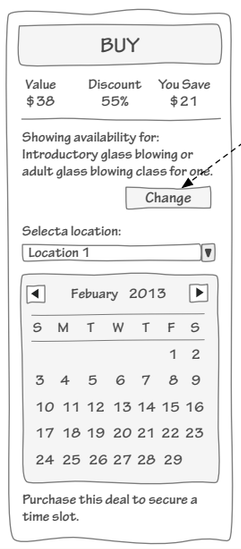
Pre-purchase Availabilities Preview

Pre-Purchase Availability

Scheduler-enabled deals will get a "Check Availability" button (on their Groupon.com deal page) that would allow the consumer to get a preview of availability before they decide to purchase the Groupon. The hypothesis is that this will improve the customer experience, and also help diminish refunds (~15% of refunds are because of scheduling issues).

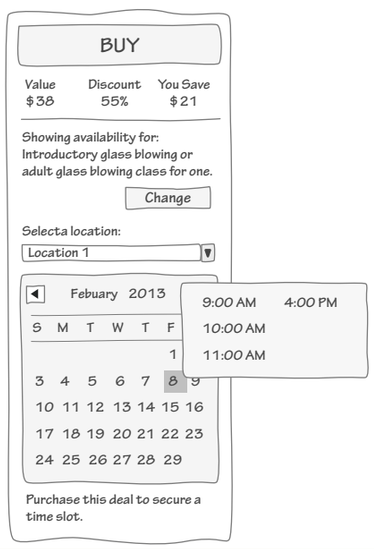
Clicking on the the Check Availability button should show all the options for the deal via a similar popover to the current deal option popover. If it is not a multi-option deal, then no popover would be shown.

Once you select an option, the date picker with available dates will fade in. Like the new booking steps timeslot step, dates with availability should be highlighted and dates without availability disabled.



Clicking this would bring up the option choice popover again.

When you hover over a date, a popover would appear with available timeslots.



3 Engineers taking 6 Weeks

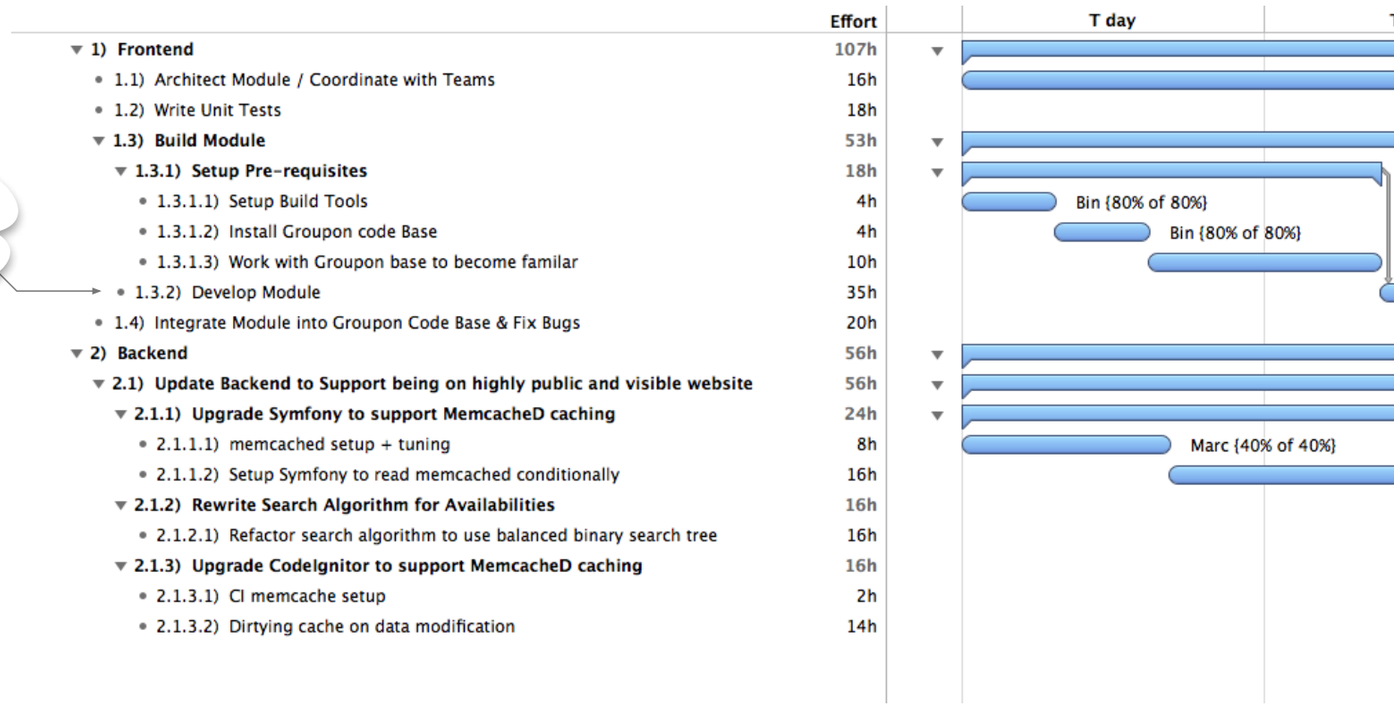
Total Hours: 163h

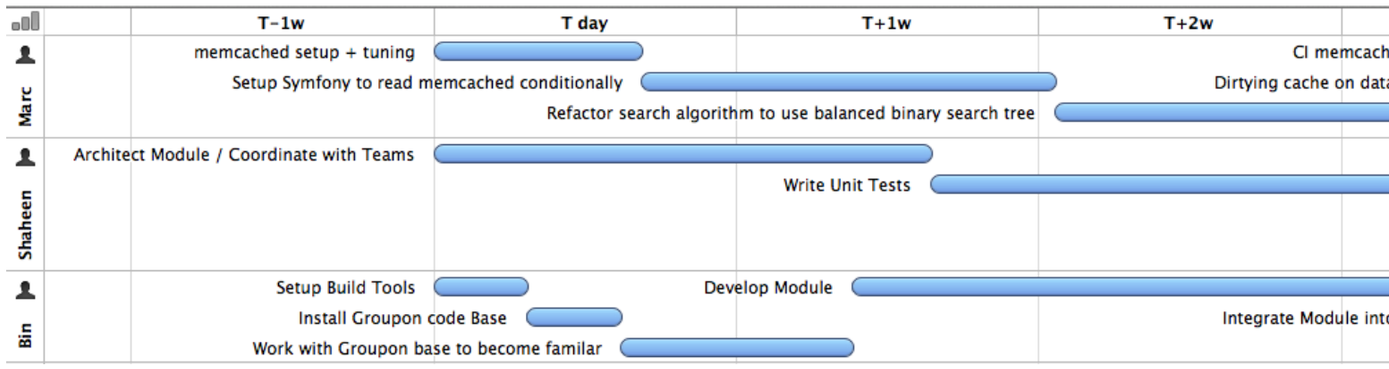
Front-end Hours: 107h

Back-end Hours: 56h

There is the possibility of reusing the code from Getaways. This is currently unknown.

	Effort
1) Frontend	107h
1.1) Architect Module / Coordinate with Teams	16h
1.2) Write Unit Tests	18h
1.3) Build Module	53h
1.3.1) Setup Pre-requisites	18h
1.3.1.1) Setup Build Tools	4h
1.3.1.2) Install Groupon code Base	4h
1.3.1.3) Work with Groupon base to become familar	10h
1.3.2) Develop Module	35h
1.4) Integrate Module into Groupon Code Base & Fix Bugs	20h
2) Backend	56h
2.1) Update Backend to Support being on highly public and visible website	56h
2.1.1) Upgrade Symfony to support MemcacheD caching	24h
2.1.1.1) memcached setup + tuning	8h
2.1.1.2) Setup Symfony to read memcached conditionally	16h
2.1.2) Rewrite Search Algorithm for Availabilities	16h
2.1.2.1) Refactor search algorithm to use balanced binary search tree	16h
2.1.3) Upgrade CodeIgnitor to support MemcacheD caching	16h
2.1.3.1) CI memcache setup	2h
2.1.3.2) Dirtying cache on data modification	14h





Research Notes

Why does this project take so long to build?

On first glance there is quite a discrepancy between the tangible size of the feature and the amount of time / resources it takes to complete the project. This dichotomy of expectations revolves primarily around two axis. Organizational Communication Requirements vs. Living in a Box & Engineering vs. Hacking.

In the first case, two of the three engineers being tasked to this project have management level responsibilities and various relationships / commitments that must be honored throughout the week. This coordination with other teams and the other extraneous responsibilities, reduces the amount of focus they can apply to this project. The project can be sped up by sticking them in a box, but will come at the expense of other organizational initiatives falling short of their needs.

Secondly, engineering projects vs. hacking projects are quite different. The core amount of work is only 55 hours and is tasked to a single developer. If we want to take the hacker approach (a reasonable decision under certain conditions), the work time can be reduced drastically. Higher-level engineering requirements such as, unit tests, code reviews, performance optimization, team member development can be dropped in order to achieve maximum output.

What is the timeline if we stick everyone in a box?

3 Engineers taking 4 Weeks

What is the timeline if we ignore Engineering level requirements and hack it?

1 Engineer taking 3 Weeks

Hacking a project together is perfectly reasonable under certain conditions. Engineering level requirements can be overly expensive at times. It is recommended that we do not hack this project together though. This code will live on the public facing portion of the Groupon.com website. Reductions in load time, stability and usability will be highly visible and could cause consumer frustration. This frustration could decrease sales. Saving several weeks worth of resources, at the cost of reducing the sales conversion rate, does not seem like the appropriate trade off.

Web

ROLE Software Architect

TECHNOLOGIES USED OmniGraffle, OmniPlan

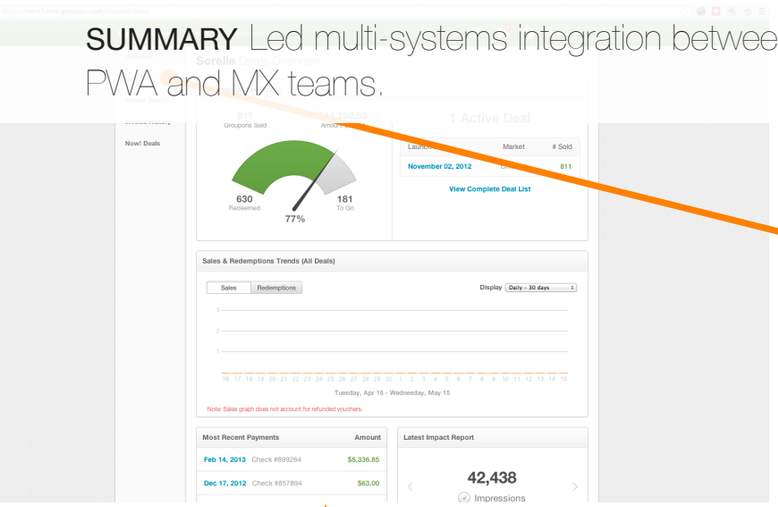
SUMMARY Created engineering architecture and project plans for Groupon's roll-out of advanced booking functionality.

Web

ROLE Engineering Manager

TECHNOLOGIES USED OmniGraffle

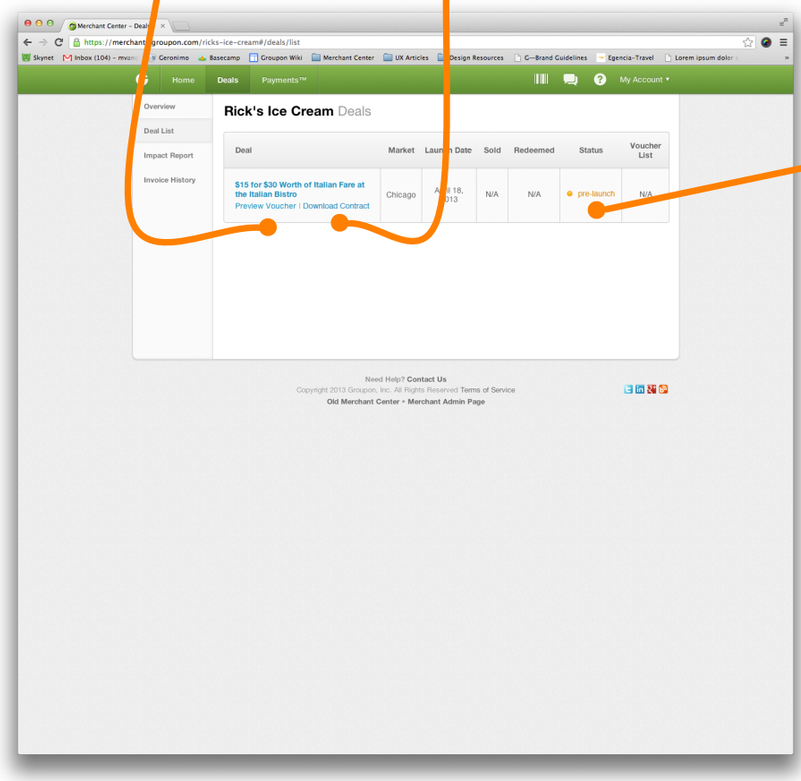
SUMMARY Led multi-systems integration between S3, PWA and MX teams.



Where?

Where?

Where?



Update /deals to support S3

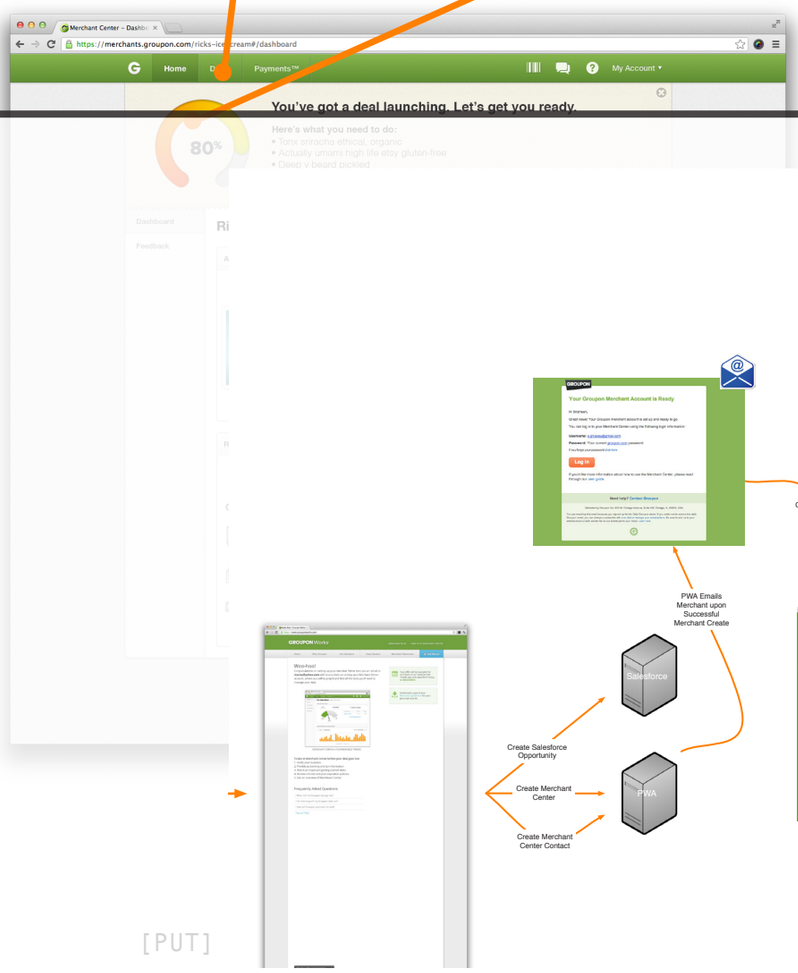
```
GET /v0/merchant/<:salesforceId>

[{
  claimNumber:
  mainLocation:
}]
```

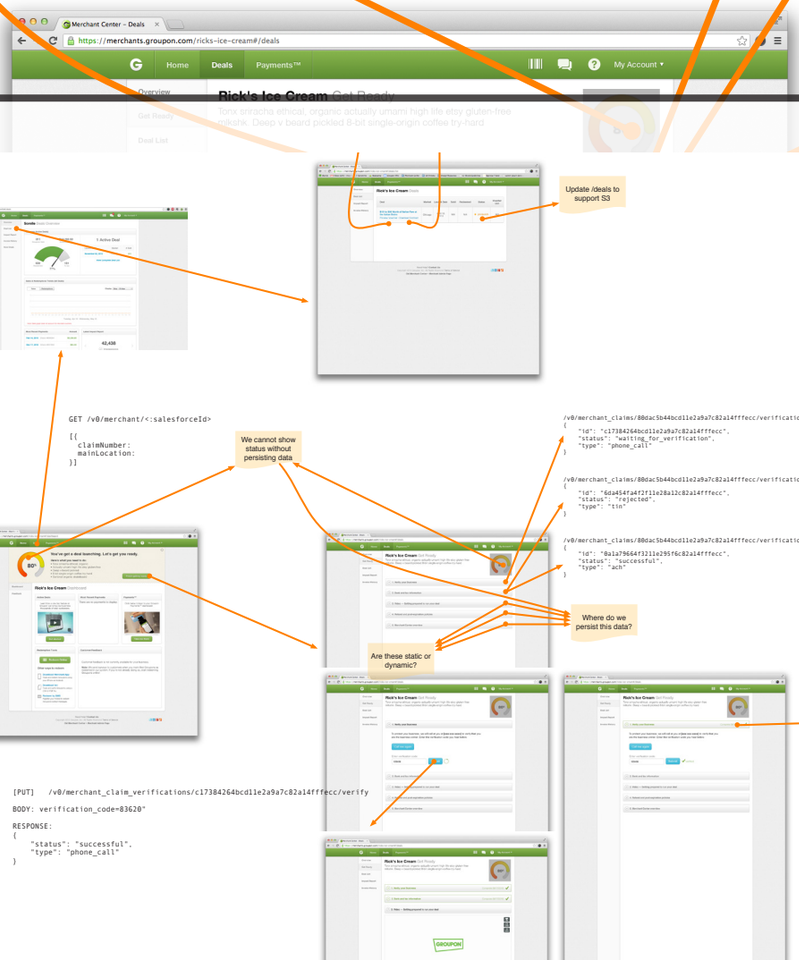
We cannot show status without persisting data

```
/v0/merchant_claims/80dac5b44bcd11e2a9
{
    "id": "c17384264bcd11e2a9a7c82a14f
    "status": "waiting_for_verificatio
    "type": "phone_call"
}
```

```
/v0/merchant_claims/80dac5b44bcd11e2a9
{
    "id": "6da454fa4f2f11e28a12c82a14f
    "status": "rejected",
    "type": "tin"
}
```

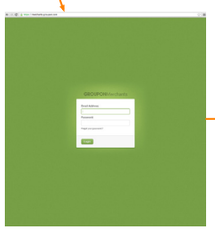
```
/v0/merchant_claims/80dac5b44bcd11e2a9
{
    "id": "0a1a79664f3211e295f6c82a14f
    "status": "successful",
    "type": "ach"
```





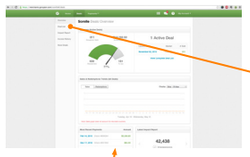
Customer Clicks Log-in Button

PWA Emails Merchant upon Successful Merchant Create



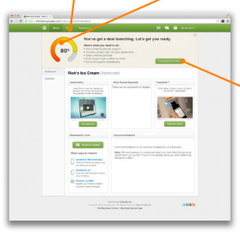
How do we know is an account is for an S3 Merchant?

```
GET /v2/merchants/<merchantPermalink>

{
merchant:
  {
    isSelfServiceMerchant: true
  }
}
```

Update /deals to support S3

```
GET /v0/merchant/<:salesforceId>

[{
  claimNumber:
  mainLocation:
}]
```



We cannot show status without persisting data


```
/v0/merchant_claims/80dac5b44bcd11e2a9a7c82a14fffecc/verifications?method=phone_call&merchant_location_id=56f024fb839eb06a9298013a914e749e&phone_number=6
{
  "id": "c17384264bcd11e2a9a7c82a14fffecc",
  "status": "waiting_for_verification",
  "type": "phone_call"
}
```

phone

```
/v0/merchant_claims/80dac5b44bcd11e2a9a7c82a14fffecc/verifications?method=tin&merchant_location_id=56f024fb839eb06a9298013a914e749e&tin=555555555
{
  "id": "6da454fa4f2f11e28a12c82a14fffecc",
  "status": "rejected",
  "type": "tin"
}
```

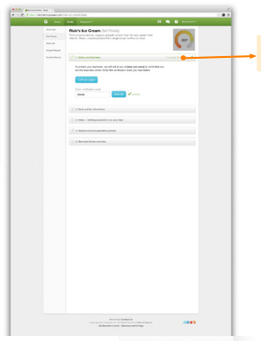
tin

```
/v0/merchant_claims/80dac5b44bcd11e2a9a7c82a14fffecc/verifications?method=ach&bank_name=Wells Fargo&bank_routing_number=053101273&bank_account_number=555
{
  "id": "8a1a79664f3211e295f6c82a14fffecc",
  "status": "successful",
  "type": "ach"
}
```

bank_name | bank_routing_number | bank_acc

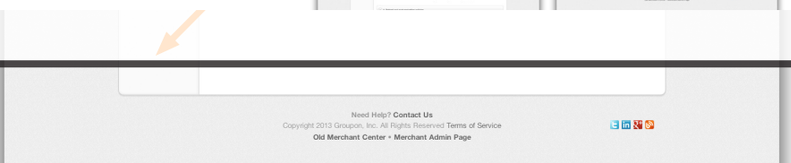
Where do we persist this data?

Are these static or dynamic?



Need to add completed data on GET

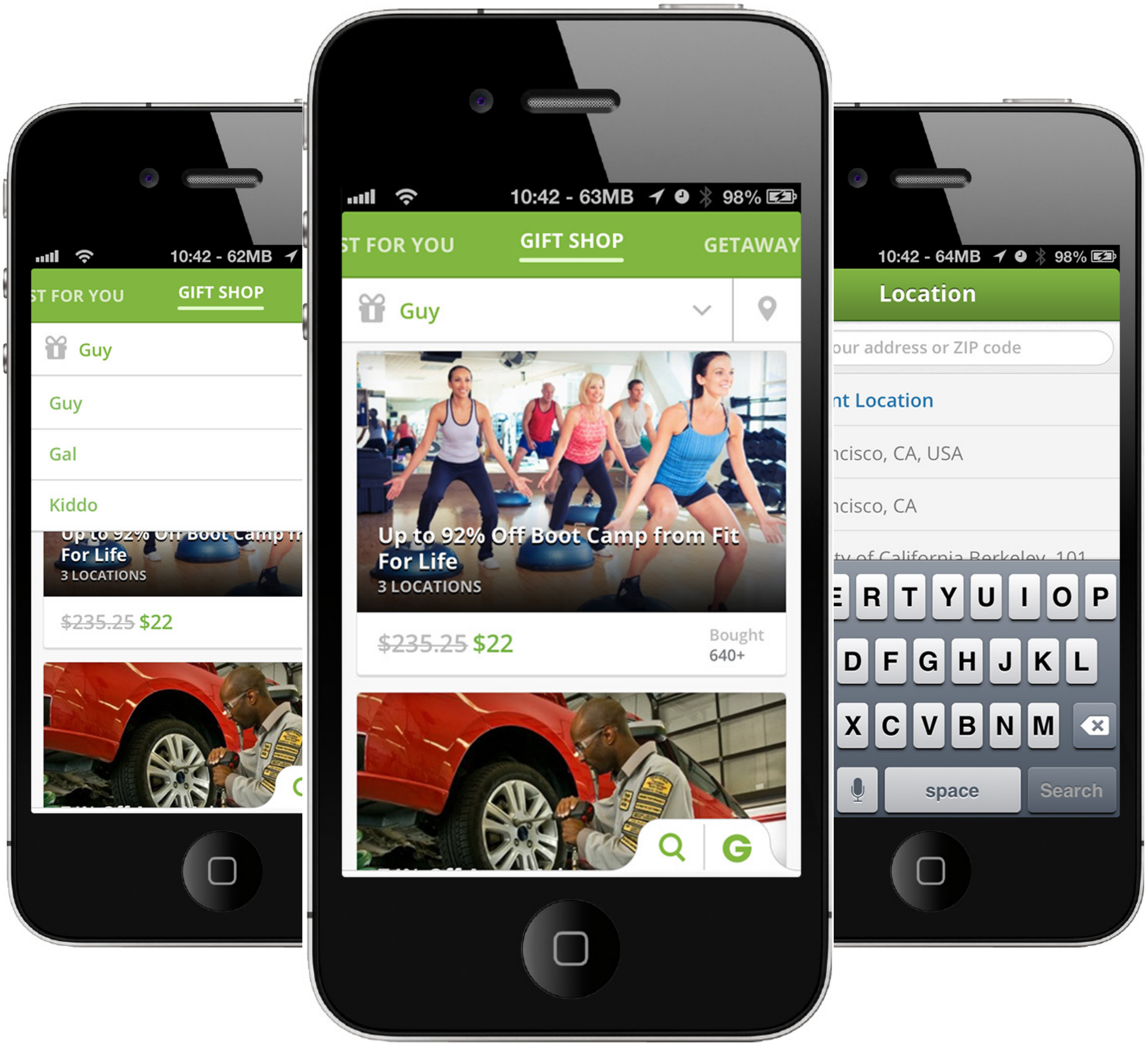
```
[PUT]   /v0/merchant_claim_verifications/c17384264bcd11e2a9a7c82a14fffecc/verify

BODY: verification_code=83620"

RESPONSE:
{
    "status": "successful",
    "type": "phone_call"
}
```




```
[PUT]

BODY:

RESPONSE:
{
    "status": "successful",
    "type": "phone_call"
```

Mobile

ROLE iOS Mobile Engineer

TECHNOLOGIES USED Xcode, iOS SDK, Objective-C

SUMMARY Programmed Groupon's Christmas holidays feature into iPhone app.