Expresso

Project Plan

Team Number	34
Bill Adamowski	Client & Advisor
Lucas Collins	Communication Lead
Jonny Krysh	Webmaster
Jake Long	Technical Lead
Garret Meier	Team Lead
Derek Yu	Key Concept Holder

Last Revised: October 16th, 2016

Table of Contents

1 Introduction
1.1 Project statement
1.2 Problem Statement
1.3 Goals
1.4 Definitions

2 Deliverables
2.2 Overall Project Deliverables
2.3.1 Consumer and Provider Interface Solutions
2.3.2 Back End System Solutions
2.3.3 Shipment Method Solutions

3 Design
3.1 Previous work/literature
3.1.1 Bean Box
3.1.2 Blue Bottle Coffee
3.1.3 Counter Culture Coffee
3.1.4 Ritual Coffee Roasters
3.2 Proposed System Block diagram
3.3 Assessment of Proposed methods
3.4 Proposed Solution
3.5 Validation

4 Project Requirements/Specifications
4.1 functional
4.2 Non-functional
4.3 Acceptance Criteria
4.3.1 Inventory Acceptance Criteria
4.3.2 Delivery Acceptance Criteria
4.3.3 Communication Acceptance Criteria
4.3.4 Data Analysis Acceptance Criteria
4.3.5 Billing Acceptance Criteria

5 Challenges
5.1 Partnering with Local Providers
5.2 Logistics
5.3 Handling Sensitive Information
5.4 Cost Estimates

6 Timeline
6.1 First Semester
6.2 Second Semester

7 Conclusions

8 References

1 Introduction

1.1 Project statement

Our aim with Expresso is to provide services to local coffee roasters for expanding and optimizing their business by creating an efficient, automated platform for coffee roasters to promote and sell their unique coffee brands to a broad range of customers.

1.2 Problem Statement

Small, local coffee shops struggle to make decent profits on roasting and distributing their beans. The low margins in the coffee industry are due to many factors like price competition with international chains (Starbucks, Dunkin, etc.), high cost of beans or roasting, and inventory optimization. Conversely, local coffee businesses often impact coffee growers most, paying them more reasonably than large chains, as well as influencing communities by providing a unique atmosphere for work and leisure. Creating a system similar to Expresso is highly cost prohibitive for individual roasters. Adding in the cost of building and maintaining an online customer base, only a few, high-profile shops have entered the market (Counter Culture, Blue Bottle). By removing the necessity of shops to create their own e-commerce platform, we hope to provide the opportunity for local coffee shops to expand their businesses with minimal effort and cost. Coffee shops currently roast their beans in house to save money and create a unique tasting blend, but expanding that business takes significant technical and marketing effort. As a result, most local roasters don’t sell their coffee outside of their brick and mortar shop.

1.3 Goals

From a consumer perspective, our service will help people discover their favorite coffee brands and pay to get them shipped to their door. We want to provide a convenient, reliable, and easy way for people to have just the right amount of coffee that fits their taste delivered to their door. We plan to do this by creating a fully automated delivery service for roasters, as well as a platform for customers to discover, purchase, and review a variety of coffee roasted by small shops around the country. This will begin in Ames, eventually scaling to Iowa and (a lofty foresight) nationally.

1.4 Definitions

Term	**Definition**
Consumer	The external client who is viewing, purchasing, and receiving shipments; the coffee consumer.
Provider	An external entity which holds a type and amount of coffee; the coffee roaster.

Figure 1

2 Deliverables

2.2 Overall Project Deliverables

- Consumer can order and receive periodic coffee shipments
- Providers can view and edit an inventory of available goods
- Providers can receive consumer shipment information to fulfil orders
- Consumers can be alerted to incoming and delivered shipments
- Consumers are billed for purchased goods
- Providers are paid for fulfilled shipments

These deliverables will take the form of a software application comprised of an interface for Consumers, an interface for Provider, a back end software system for managing data collected from Consumers and Producers, and a method for shipments.

2.3 Possible Solutions

2.3.1 Consumer and Provider Interface Solutions

We will have unique interfaces for each developed as:

- Mobile Application
- Desktop Application
- Responsive Website

2.3.2 Back End System Solutions

- Microservice architecture
- Monolithic server architecture

2.3.3 Shipment Method Solutions

- Centralized distribution center containing all goods
- Decentralized shipments sent to Provider

3 Design

3.1 Previous work/literature

3.1.1 Bean Box

This coffee distributor, found at https://beanbox.co/, works with 21 of Seattle's roasters to distribute their coffee in consolidated boxes to Consumers monthly. This service uses the centralized shipping method for coffee, and offers subscriptions of one, 12oz bag for $20 monthly. They tout fresh (within 48 hours) roasts delivered with free shipping around the US. Bean Box has high marks in high quality roasts, but lacks in customizability and cost effectiveness. Consumers are tied to the roasting schedule and specific amounts monthly.

3.1.2 Blue Bottle Coffee

Formerly Tonx, Blue Bottle, at https://bluebottlecoffee.com, was one of the early entries into subscription-based coffee services. They run their entire vertical from sourcing beans through roasting and distribution. Blue Bottle gives among the highest quality beans and mid-level prices, but lacks in variety and customizability. All Consumers receive the same bags of coffee each month, and other options are few and far between.

3.1.3 Counter Culture Coffee

Counter Culture is one of the more fleshed out coffee subscription brands with a complete vertical from purchasing beans to shipping the roasted coffees. Found at https://counterculturecoffee.com/, they offer many, customizable roasts and subscription offerings. Their coffee isn't branded as high class as Blue Bottle or Ritual, but their technical features are quite sound. The primary drawback to the service is that it's central to Counter Culture roasts rather than offering a solution to local roasters.

3.1.4 Ritual Coffee Roasters

Ritual Coffee, a San Francisco based coffee roaster, found at https://www.ritualroasters.com, functions similarly to Blue Bottle, but uses an external service called Shopify (http://ritual.myshopify.com/) to handle their eCommerce functionality. Ritual has a few rotating coffees, priced higher than almost all other options. They offer reasonable customizability, but lack the personalization of Counter Culture of Blue Bottle, who host their own services. Their approach of using Shopify to host their eCommerce functionality, is one that undoubtedly has a high barrier of entry for Providers, but is worth looking into as we continue our implementation of Expresso.

3.2 Proposed System Block diagram

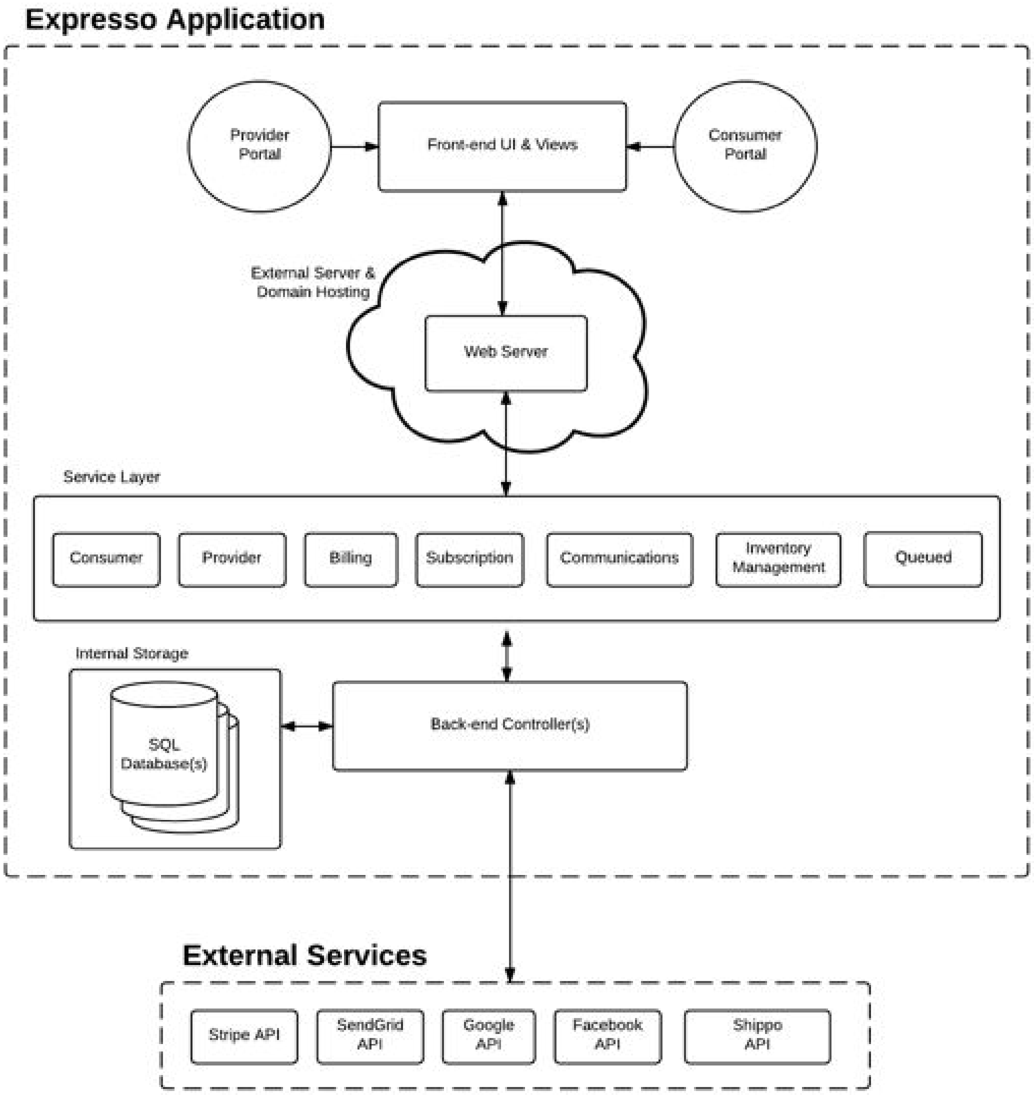

Figure 2

Our front-end will be written using React/Redux for views. Using these paradigms promotes seamless data flow across the application. Our web server (hosted externally) will be written as a Node.js server that has handlers for each view and calls API's to subsequent services. Both the front-end and server will use Javascript. We will break every service down into it's own cluster in order to provide opportunities for scaling and fault tolerance. In addition, this makes it easier for us to test services since we can mock out responses from our back-end API calls. Our back-end controllers will be written in Go and take advantage of services similar to RabbitMQ to handle load balancing, queued services, messaging and server faults.

The providers and consumers register through the front-end and are authenticated to either the provider portal or the consumer portal. Following that, any actions will be sent to the web server as (authenticated) HTTPS requests.   The web server sends all communication through the service layer, written in Go, to the back-end controller, which will handle communication through all external services, datastores, as well as internal storage in SQL databases.

3.3 Assessment of Proposed methods

Centralized distribution center containing all goods		**Decentralized shipments sent to Provider**	
Advantages	Disadvantages	Advantages	Disadvantages
Easier to ensure successful delivery of beans	Challenging to find suitable storage for coffee beans	Shipped coffee beans will be fresher	Harder to ensure successful delivery of beans
Easier to maintain an up-to-date inventory count for all beans	Increased manual labor	Saves time and money by not needing to ship to some middle-man warehouse	Coffees shops will need to take the time to box up packages and take to a shipping center

Microservice architecture		**Monolithic architecture**	
Advantages	Disadvantages	Advantages	Disadvantages
Easily Scalable.	More moving parts to manage and upkeep (from a developer standpoint).	(Generally) one programming language and small number of moving parts.	Does not scale well at all.
Easier to deploy (since each microservice can be individually deployed in some way)	If there are not very many microservices, deploying each individual microservice may add unnecessary complexity and required time to test.	May be less work involved (compared to microservice) if the code is not intended to be too big to manage/maintain come the future.	Code will be harder to maintain.

Responsive Web application		**Mobile App + Web Application**	
Advantages	Disadvantages	Advantages	Disadvantages
Enforces the use of our product on one platform which provides benefits from a developer standpoint (e.g., removes necessity for mobile-first design patterns)	Providers and consumers cannot use the application when away from their computers.	Provider can count and track inventory while using our application on a tablet or phone.	App store provides a pain point in that consumers and provider may never reach our application.
			More development required for MVP.
			Knowledge of more development tools is required.

3.4 Proposed Solution

Given the assessed advantages and disadvantages above, we decided on a solution with responsive web applications for both Consumers and Producers and a microservice back end system powering the interactions with a distributed shipping method.

By using a web application as the primary UI for Expresso, we can maximize usability across devices while condensing our development workflow into a single platform. While mobile applications provide easy access on phones and tablets, developing applications can be time intensive with limited abstraction between applications. Developing mobile solutions which function at a high level on both Android and iOS platforms would require significant development effort which could be instead applied to building the core product. An additional consideration in our decision to build the UI as a web application was the primary use cases of Expresso. From a user perspective, the ideal experience is ordering a subscription and continuously receiving high quality product within expected timeframes without visiting the website again. The value of Expresso isn't in an interface users spend a great deal of time using, but in reliability of our back end systems. Once again, this highlights that the best use of our development time is in building reliable systems for handling the logistics of shipping coffee subscriptions.

When considering whether to use a monolithic or microservices architecture, we weighed the technical as well as development process impacts of both options and decided on the microservice approach for a few reasons. One, microservices offer increased flexibility and autonomy for developers. Rather than performing rapid development on a code base with dependencies spread throughout, we can work on smaller services with mocked responses from areas which haven't been implemented. Scaling is an additional point in favor of using microservices which allow us to individually increase capacity for services which are facing a heady load rather than scaling the entire application. This would let us react rapidly to high demand situations.

Finally, we chose to work with the distributed model for shipping the Producer's product to our Consumers. This model allows Expresso to ensure fresh coffee being shipped along the shortest path from Producer to Consumer. The distributed model increases our logistical workload, with having to manage packaging and shipping information, but ensuring freshly delivered beans took priority. Additionally, this method removes a barrier to expanding Expresso to additional Producers by letting the Producers manage their own inventory and not sending their beans to a central distribution center.

3.5 Validation

Requirement	**Test**
Consumer can order and receive periodic coffee shipments.	Have someone sign up for a coffee subscription and then see if they receive their coffee beans periodically.
Providers can view and edit an inventory of available goods.	Have a coffee provider use the application. Make sure they can view and edit inventory in an acceptable way.
Providers can receive consumer shipment information to fulfil orders.	Have someone order coffee. Have a coffee provider use the application to see if they have been notified of a new order and if they can access necessary information to ship the product to the consumer.
Consumers should be alerted to incoming and delivered shipments.	Have someone order coffee and check to see if they receive text/email messages detailing the status of their shipment.
Consumers are billed for purchased goods.	Have someone order coffee and check to see if they are billed correctly. Do this by checking the consumer's credit card/bank account to see a corresponding charge.
Providers are paid for fulfilled shipments.	Have someone order coffee. Check that the money is routed through our project/system's account. Then check that the money is then routed to the correct providers. Check with the provider's to make sure their account's reflect the appropriate deposit amount.

4 Project Requirements/Specifications

4.1 Functional

Providers shall:

- Post their different coffee bean types and prices
- View orders placed for their coffee beans
- Receive payments for orders
- Browse available coffee beans from various shop owners, and be able to place orders on these beans

Customers shall:

- Subscribe to periodical deliveries of recommended bean
- Send payments for orders they place
- Set preferences for their preferred coffee types
- View orders they have placed and track their orders

Both shall:

- Log in using an email and password — without both of these the user should not be allowed access to the service

4.2 Non-functional

- All parts of the system should be secure and protected against common attacks like:
    - XSS
    - SQL Injection
- Code should be easy to maintain and understand
- User interface should be easy to use

4.3 Acceptance Criteria

4.3.1 Inventory Acceptance Criteria

- Tracks available roasts and anticipated stock
- Collects and uses past data to inform ordering and distribution decisions
- Manages the supply chain from getting coffee from providers to consumers
- Predicts future demand to alert the providers when they'll need to increase production to keep up with more consumers

4.3.2 Delivery Acceptance Criteria

- An interface for providers to enter our onboarding pipeline which gives our systems the information they need to add new available roasts with minimal human oversight
- Gather consumer's taste and consumption profile information to make decisions about which coffees to recommend or deliver periodically

- Handle consumer transactions from the moment the first box is purchased through its packaging, shipping, and tracking

4.3.3 Communication Acceptance Criteria

- Inform providers about their past, current, and future orders which shows the value they've gained using our service
- Notify consumers of the current status of their delivery and potential promotions or new coffee additions
- Adjust taste profile for each coffee shipment based on changing consumer preferences

4.3.4 Data Analysis Acceptance Criteria

- Systems will gather and analyze data - varieties and amounts of beans - in order to personalize a consumer's experience the longer they subscribe

4.3.5 Billing Acceptance Criteria

- Use Stripe API to handle billing and personal information in a secure and scalable manner

5 Challenges

5.1 Partnering with Local Providers

We must discover providers who will be interested in partnering with us. Outreach and communication will be key, and we will need an effective pitch to generate interest for our system, while also taking into consideration a provider’s needs and requirements. If we cannot successfully complete this, then Expresso will have no providers, rendering out application unserviceable.

5.2 Logistics

We will need to develop an efficient inventory system and delivery system to ship coffee beans to consumers. These systems will need to accurately manage the supply chain - multiple provider’s inventories, and deliver beans to consumers. It will be challenging to create these complex systems and produce a smooth, automated Expresso experience for consumers: from ordering beans to receiving these beans at their doorstep, while also taking care of logistic details in the background.

5.3 Handling Sensitive Information

Expresso is a subscription based system, so we will need to handle sensitive information for both consumers and providers in a secure manner. We need to be careful any time sensitive information is managed, whether it be billing information or the payments.

5.4 Cost Estimates

We will take advantage of Stripe for payments and Shippo for shipments, both which offer pay as you go plans. Stripe charges 2.9% + 30¢ per processed payment, while Shippo charges shipping item costs + 5¢ per shipped item. We will determine the prices for each category of coffee based on market research, keeping prices both feasible and profitable. These prices are still yet to be determined.

6 Timeline

6.1 First Semester

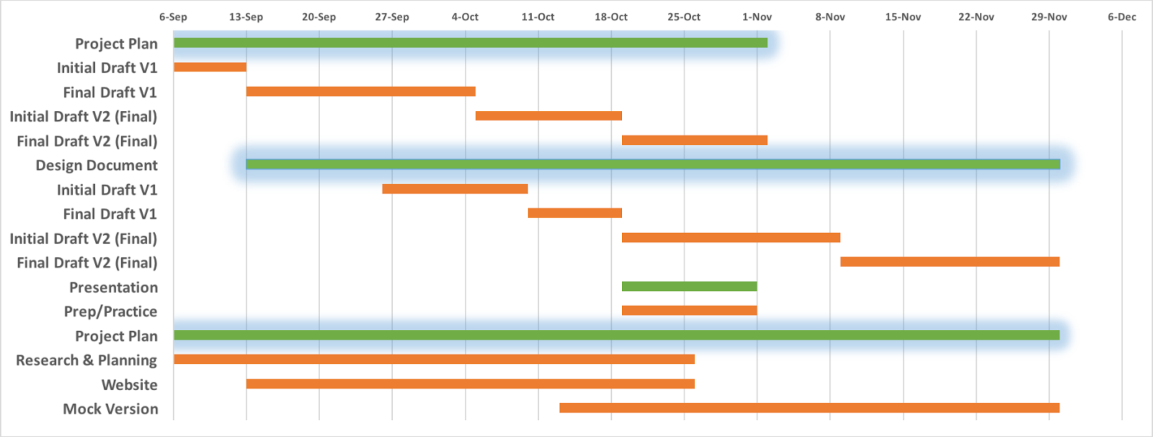

Figure 3

The Fall semester consists mostly of planning and writing up documents relating to planning and design. Throughout the semester we will all collectively be working on the first couple of versions of the Project Plan and Design documents. Throughout the first half of the semester we will be planning the implementation of the project. In the last half of the semester we will begin to create a base model (mock version) that will carry over into the Spring semester. All programming work for this base model will be split up between the five of us in a way that is yet to be decided.

6.2 Second Semester

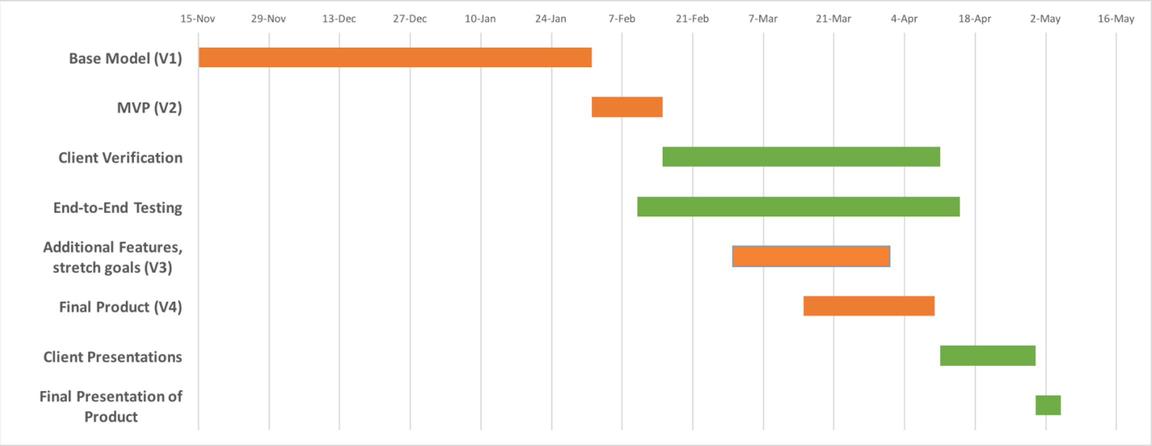

Figure 4

In the Spring semester we will spend most of our time programming the project. The first half of the semester will be spent working on a base model which will be cleaned up and made into an MVP. From there we will work with coffee shops (clients) to see if any improvements can be made that they would like to see (verification and user testing). From there we will do further testing and add any extra features that we feel should be included in the final product. The last section of the semester will be spent finishing up the final product and presenting to clients and the class. All of the above will be split up amongst the team in some way that is yet to be determined.

7 Conclusions

Our goal with Expresso is to give local roasters a higher profit margin by connecting them with a larger audience of consumers. In addition to making them more profitable, this will also allow the roasters to be more competitive with international chains. Expresso is also targeted at consumers that like to make their own coffee, allowing them to receive a monthly delivery of their favorite roasts, or to try completely new roasts from local roasters they would not have known about without Expresso.

We will design our product to be easily usable for both consumers and providers, with an intuitive user interface and easily accessible services. We will also design the application to be scalable to allow for easy future growth.

8 References

1. http://www.scaa.org/chronicle/2014/09/15/the-cost-of-a-cup-of-coffee-where-does-the-money-go-2/
2. https://www.quora.com/Whats-the-margin-on-coffee-beans-from-roaster-to-distributor
3. http://www.investopedia.com/articles/personal-finance/010816/economics-owning-coffee-shop.asp
4. https://www.fundera.com/blog/2015/02/24/the-economics-of-your-local-coffee-shop