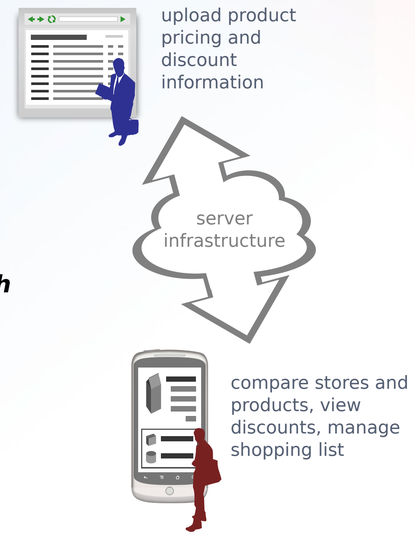
Store Selector

Group Dec1206
Blair Billings
Timothy Kalpin
Kurt Kohl
Chris Morgan
Kerrick Staley

Client
Google
Muthu Muthusrinivasan

Adviser
Manimaran Govindarasu



Concept Sketch

Abstract

• Online shopping has advanced the shopping experience, but brick-and-mortar shopping hasn't adopted the same improvements
• In particular, it's hard to compare prices and products across brick-and-mortar stores; Store Selector addresses this shortcoming
• Store Selector will allow customers to easily locate the best prices on the products they plan to purchase at such stores
• It gives the user an easy-to-manage account that responds to his/her shopping habits
• It presents the user with information on best price per item and best price per list of items
• It also presents the user with item suggestions based on his/her transaction history
• Store Selector comprises a database backend and web and mobile frontends
• Mobile was stressed because people frequently make purchases on-the-go, and the database is used to keep up-to-date, reliable pricing information
• Based on our testing, Store Selector makes it easier to find better deals and improves the overall shopping experience

Design Requirements

Functional Requirements

• Allow store managers to enter and maintain pricing data
• Allow customers to create and maintain item lists via Web Interface
• Allow customers to enter input via the voice interface on Android devices
• Allow customers to perform same functionality as the Web Interface on an Android device, plus GPS and voice

Non-Functional Requirements

• Be easy to use and aesthetically pleasing
• Ensure price data is accurate
• Be quick and responsive
• Provide clear feedback if not enough data is available to answer query
• The code in the final product shall be modular, readable, and well-documented

Operating Environment

• Host server (physical or virtual)
• Consistent availability (static IP or DNS entry)
• Mobile availability (server-side computation)
• Privacy / security of information

Intended Users and Uses

• Managers of physical and local stores
• Customers who shop at physical and local stores

Technical Details

Technologies

• Language: Groovy
• Web framework: Grails
• Development Environment: Groovy/Grails Tool Suite
• Mobile platform: Android
• Database: MySQL
• Web server: Apache Tomcat

Grails Server

• Generate pages for browser and Android interfaces
• Determine best store, other processing

MySQL Server

• Execute queries, manage data

Consumer Android App

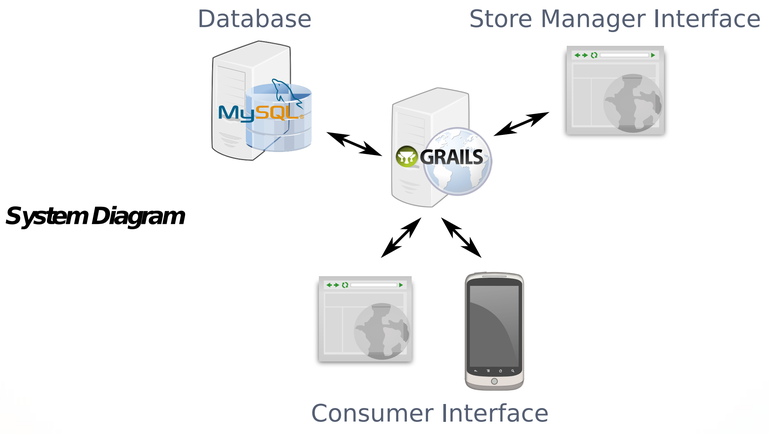
• Display pages from Grails server
• Interface to native functionality: location, microphone

Consumer Web App

• Alternative to Android app
• Display information in larger format
• Accessible from all devices

Store Manager Web App

• Allow management of store inventory
• Allow inventory import from e.g. spreadsheet



System Diagram

Testing

Front End

• Visual inspection
• Use of Firebug plugin to catch front end errors while developing

Back End

• Unit tests for Domain Objects
• Integration Tests
• Database Fuzzing
• Stress Testing

Usability Testing

• Give the application to 10 "test users"
• "test users" test mobile and desktop versions
• Gather feedback

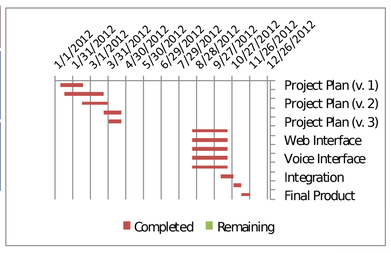
Environment

• Separate instance of application deployed to "test" environment
• Separate database instance as well
• Unit tests, Integration test run locally
• Database tests run on server

Budget and Resources

Resource	Cost
Web Server	$0
Android OS	$0
Total Cost	$0

Schedule



Summary

• We met our initial goals
• Completed:
  - Desktop component
  - Mobile component
  - Populated database
  - Working product

Risk Mitigation

We were able to mitigate work assignment distributions risks by assigning a different developer to each module, while meeting weekly to discuss status reports and module integration.
In a working installment of our application, store managers will be able to submit their own pricing information directly to the application; but as we didn't have a dedicated following while in the development stage, scraping the ISU bookstore information proved invaluable for testing module integration and user interface usability.

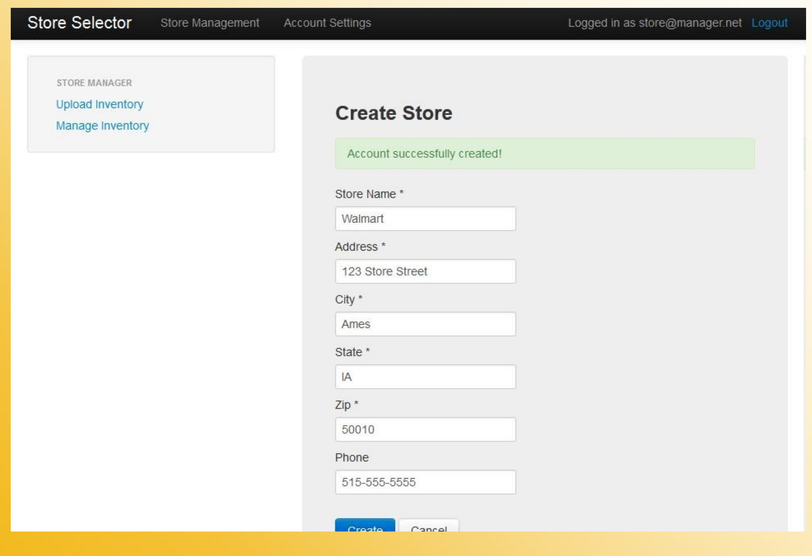
Development process

While the mobile application had several elements that were particularly native to android (voice input interface, client side views, etc.), the majority of the mobile development could not be done before the equivalent web page was completed. To mitigate this issue we started web development long before mobile development, and finished in plenty of time for the mobile

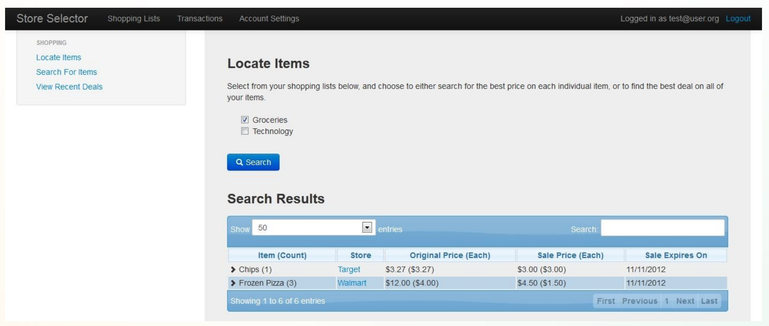
In Closing

The application code and documentation will be bundled up for a later senior design team to take on and implement features that we were unable to complete (user comments, flagging deals, deal moderators). We completed the goals that we had for this semester, and ultimately, the project was a success.

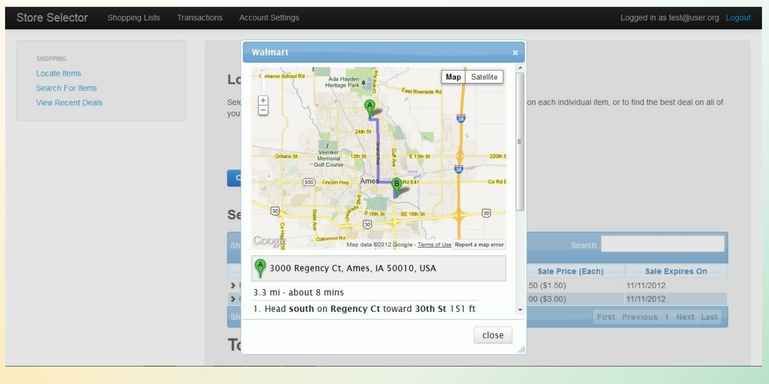
Manager Interface: Create a Store



Customer Interface: Locate Items



Customer Interface: Directions to Store



Customer Interface: Shopping Lists