Store Selector

Dec1206
Client: Google

Chris W. Morgan
Blair M. Billings
Kerrick A. Staley
Timothy R. Kalpin
Kurt D. Kohl

Problem Statement

It's very easy to compare prices across multiple sites on the internet, but this is far more difficult with physical stores. Store Selector will allow customers to easily locate the best prices on the products they plan to purchase at physical stores, and provide other features to optimize the shopping experience.

Market Survey

- Account Based
  - LivingSocial
  - Groupon
  - ScoutMob
- Coupon Apps
  - GeoQpons
  - The Coupon App

Our application brings together select functions of each site mentioned above and gives the user an easy-to-manage account that responds to your shopping habits, instead of giving you annoying, possibly irrelevant ads.

Group: Dec1206
Project: Store Selector
Client: Google

Concept

upload product pricing and discount information

compare stores and products, view discounts, manage shopping list

High-Level Description

- Store Manager Interface
  - Web application
  - Store managers will enter and maintain product data via this portion of our product
- Consumer Web Interface
  - Account based
  - Maintain shopping lists, purchased products
  - View discounts and compare store prices
- Consumer Android Application
  - Includes features of Consumer Web Interface in a convenient native Android Application

Functional Requirements

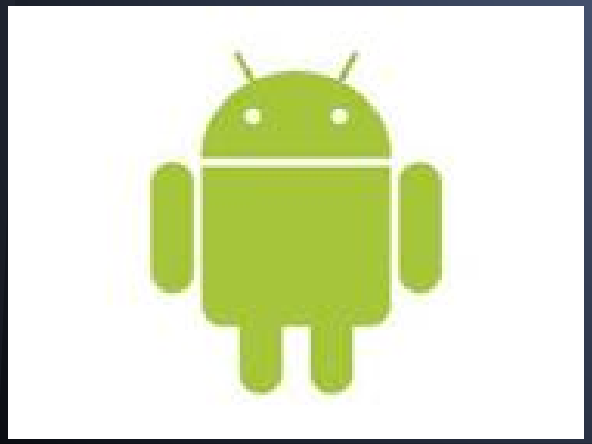
- Allow Store Managers to enter, maintain pricing data
- Allow consumers to create, maintain item lists via Web Interface
- Consumers shall be able to enter input via the voice interface on Android devices
- Allow customers to perform same functionality as the Web Interface on an Android device

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

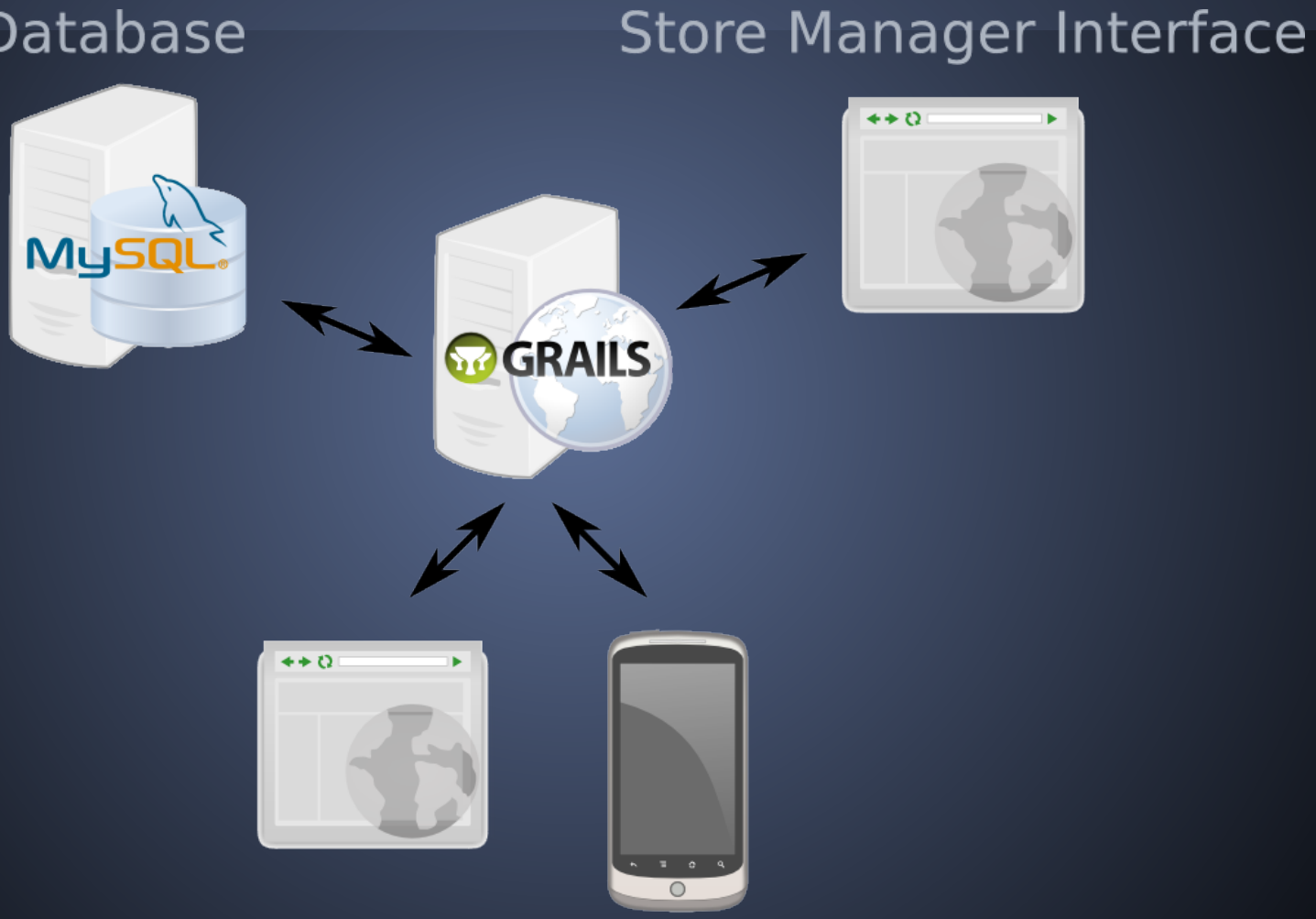
Constraints and Considerations

- Time constraints
  - Complete all iterations and implementations by next December
- User's mode of use
  - Android smartphone
  - Desktop computer webpage



Group: Dec1206
Project: Store Selector
Client: Google

System Diagram



Platforms Used

- Framework: Groovy and Grails
- Mobile Platform: Android OS
- Language: Java
- Database: MySQL

Technology Considerations

- Web Framework
  - Choice: Groovy and Grails
    - Groovy is an enhanced version of Java, and can utilize the vast Java ecosystem
    - Two group members have used Groovy and Grails; all have used Java
    - Robust SpringSource IDE
  - Alternative: Django
    - One group member has experience
- Mobile Platform
  - Android OS required

Technology Considerations

- Database
  - MySQL is free, well-established, and easily integrated with Grails
- App Development Framework
  - Choice: Android Framework (Java)
    - Allows access to phone features including microphone
    - Easier to ensure compatibility across all Android devices
  - Alternative: Web app in WebView wrapper
    - Allows rapid development with diverse, well-established technologies

Technology Considerations

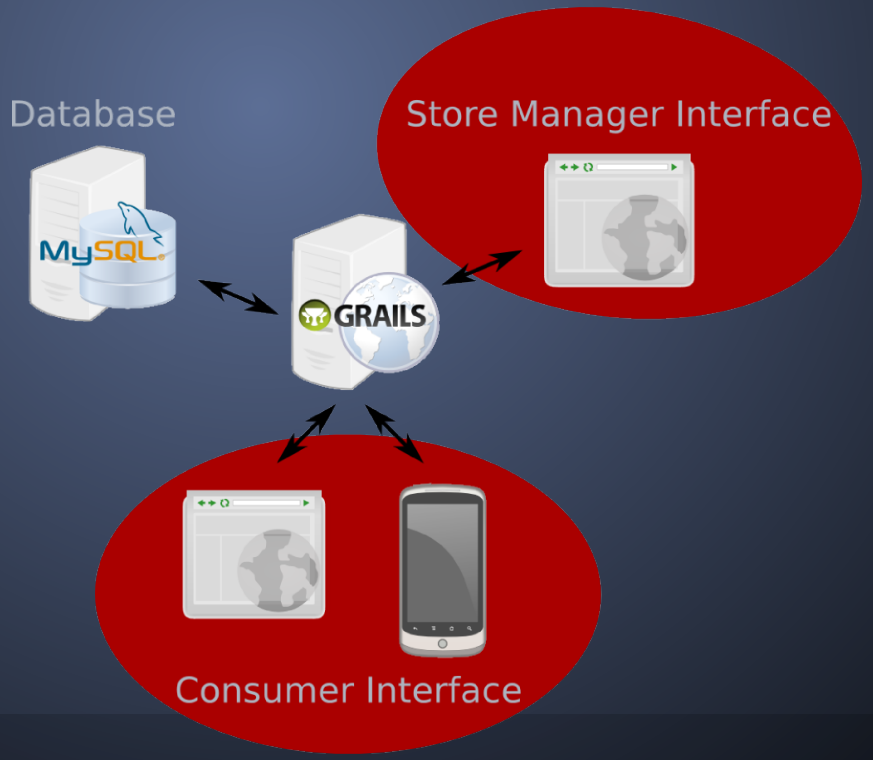
- App - Server Communication Format
  - Choice: JSON
    - Supported by Grails framework
    - Android has built-in support
  - Alternative: XML
    - Also supported by Grails framework
    - Android has built-in support, but parsing XML is harder than JSON
  - Alternative: Serialized Java Objects
    - Not actually possible due to different Java implementations in Grails vs. Android

Technology Considerations

- Computation of Price Comparisons
  - Choice: Mostly server-side (i.e. in Grails)
    - Information is more readily available on the server, so less data needs to be transferred to the client
    - Reduces processing load on processor-limited mobile device
  - Alternative: Mostly client-side (i.e. on Android)
    - Reduces processing load on server

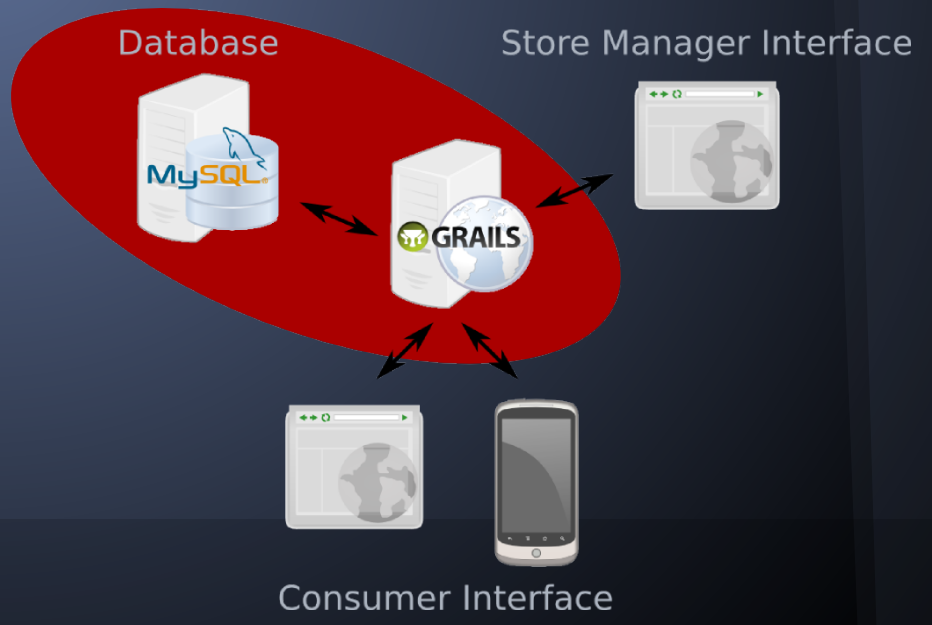
Functional Decomposition - Frontend

- Frontend
  - Android app for mobile interface
  - HTML/CSS/Javascript web pages for browser interface



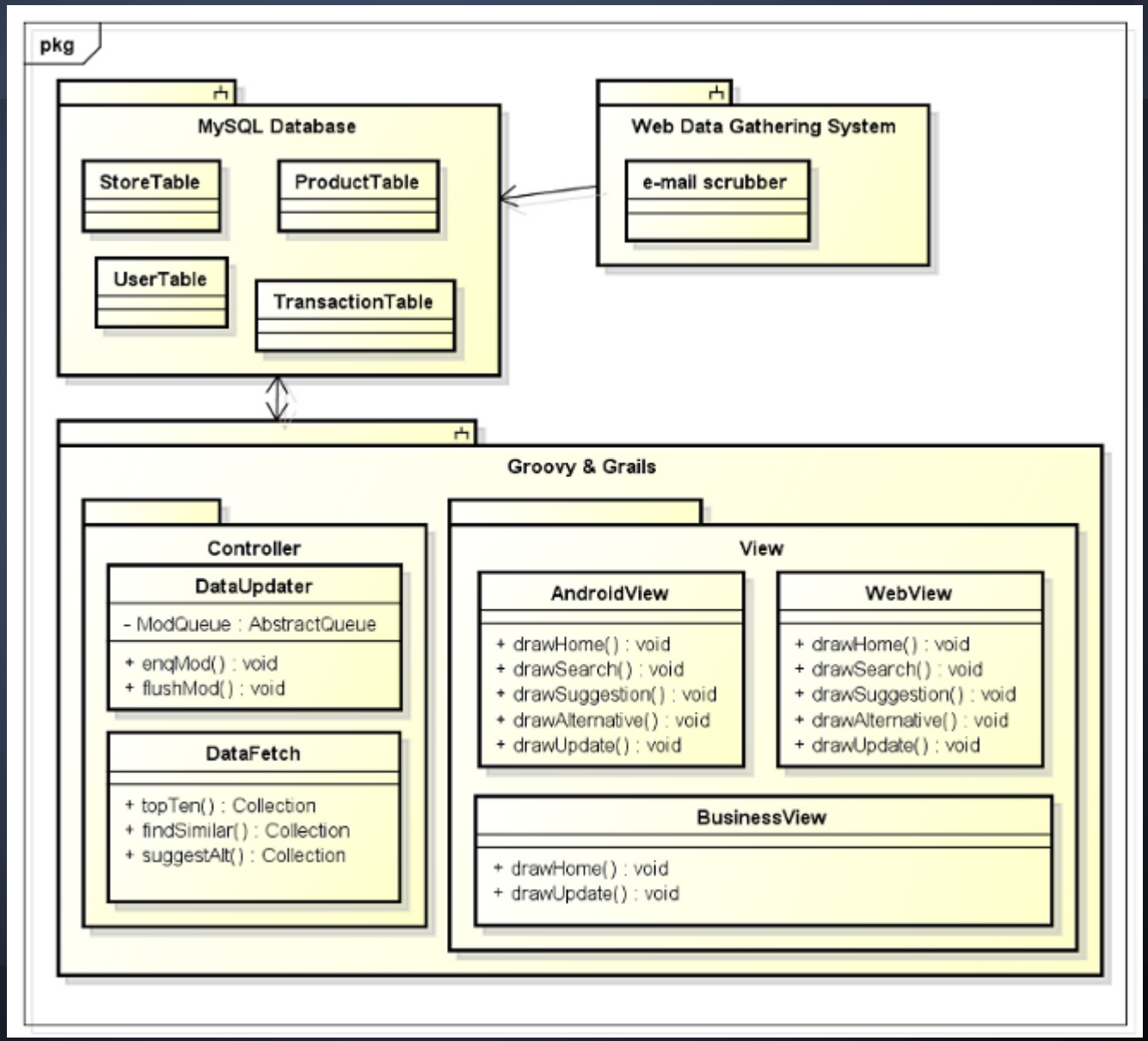
Functional Decomposition - Backend

- Backend
  - Parsers for consumers' product lists, store managers' inventories, and websites
  - Groovy Server Pages for dynamically generated/loaded web pages
  - Grails views producing JSON in response to queries from the mobile app
  - Database



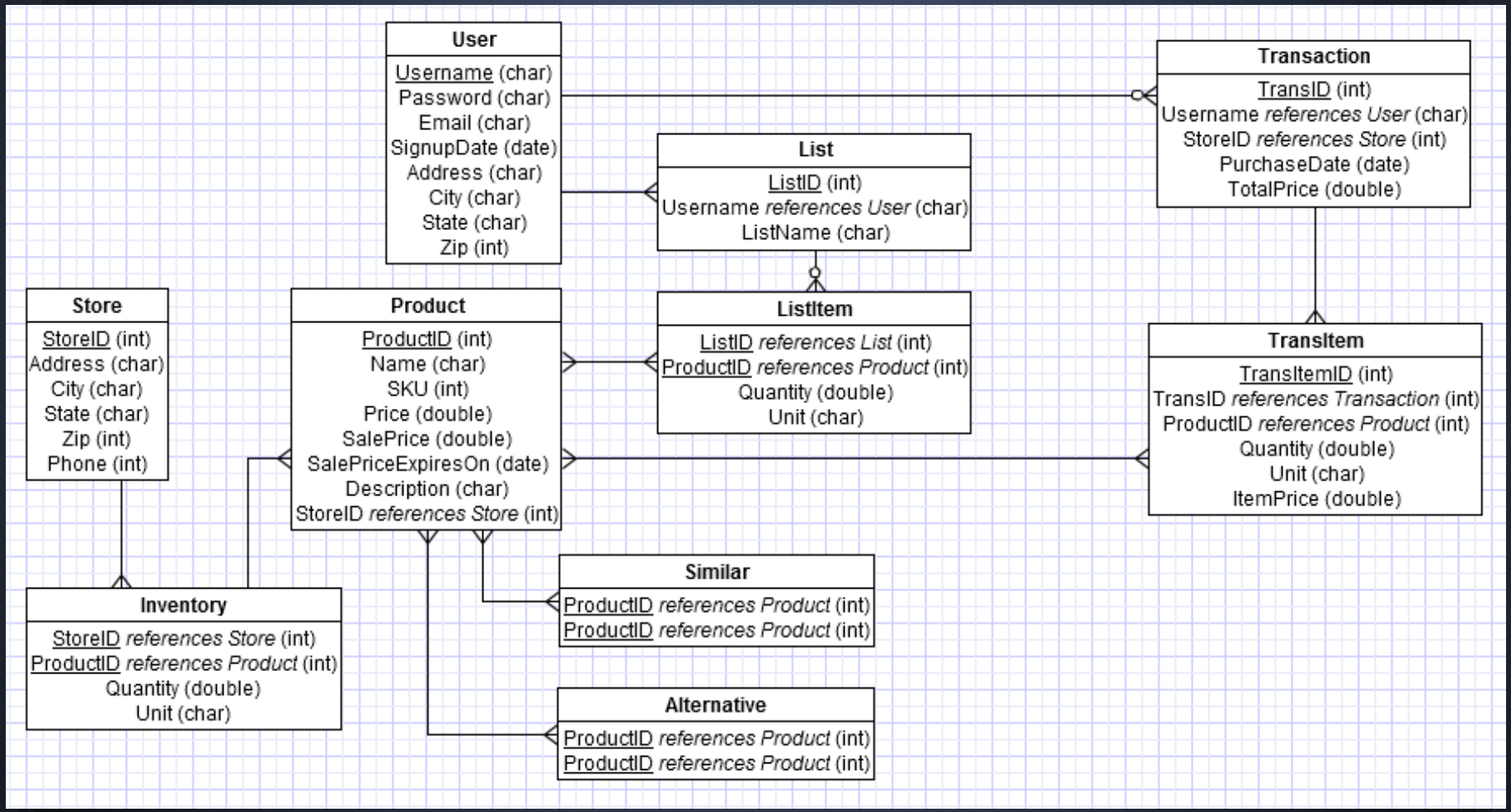
Detailed Design

Group: Dec1206
Project: Store Selector
Client: Google



Group: Dec1206
Project: Store Selector
Client: Google

Database Schema



Test Plan

- Front End Testing
  - Usability testing
  - Manual testing
  - Integration testing between controllers and views
- Back End Testing
  - Stress testing
  - Fuzzing (random data)

Estimations

Resource	Cost
Web Server [Provided by ISU]	$0
Android OS	$0
Total Cost	**$0**

Risks & Mitigation

- We've not yet been able to formalize our specification
  - Use AGILE development methodology to accommodate quickly changing customer demands.
- Unclear pricing availability on the web.
  - Allow users to flag incorrect data for modification or deletion.
  - Encourage and allow companies and businesses to submit their own information.

Responsibilities

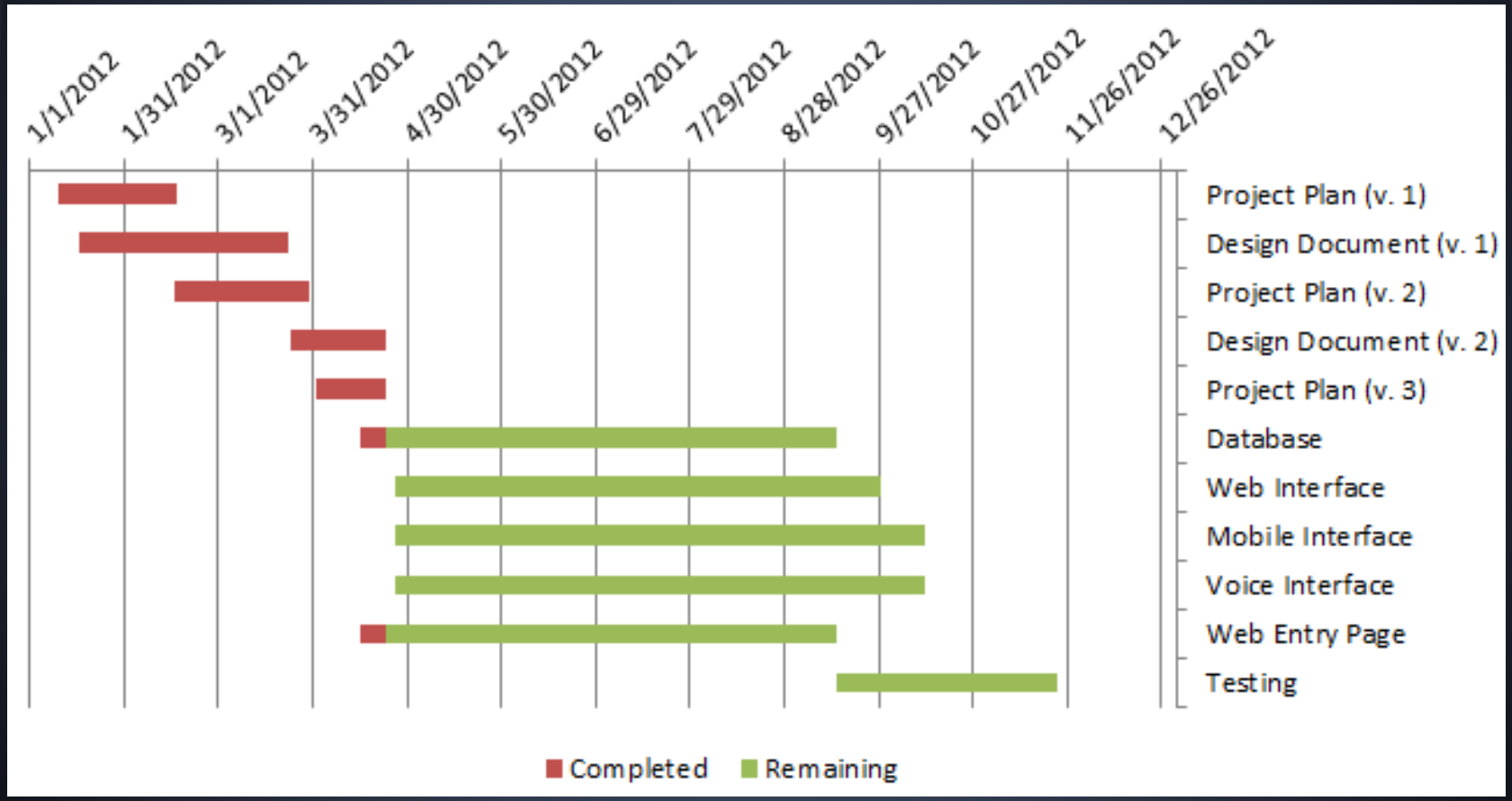
- Database - Chris
- Web Interface - Blair
- Mobile Interface - Kurt
- Voice Interface - Timothy
- Store Manager Interface and Input Parsers - Kerrick

Status

- Completed Design Document Final Draft
- Completed Project Plan Final Draft
- Decided on an AGILE development model
- Confirmed Groovy and Grails framework
- Began using mapping data objects in Grails framework

Group: Dec1206
Project: Store Selector
Client: Google

Schedule



Group: Dec1206
Project: Store Selector
Client: Google

Questions ??