Fall Progress Report

CS461 Senior Capstone

Devin Foulger, Bryan Liauw, and Hector Trujillo

Project Description

- EBay recently created multiple APIs that allow online retailers to start their stores or applications
- APIs will be released to the public soon which will allow third party developers to create their own eBay applications
- EBay would like us to test up to three of their APIs by creating an application that demonstrates their use. The application will consist of college merchandise
- It will allow users to search for and purchase merchandise related to any college and will target the new Android operating system, N OS
- The goal for eBay is that their APIs are tested and well documented for later use by other third party developers and to get more eBay related apps onto the Google Play store

Project Recap

- Week 2:
  - During week 2, our group had just been in contact with our client, Luther
  - Discussed what was expected of the project and what eBay wanted
  - Began work on our problem statement
  - Set up a GitHub repo to hold all our documents and code

Project Recap

- Week 3:
  - We completed our problem statement during week 3
- Week 4:
  - Made plans on what needed to be revised in the problem statement
  - The team had also created developer accounts on the eBay website and began looking at the APIs we would be using.
  - Created requirements for requirements document
  - Began the weekly meetings with our TA

Project Recap

- Week 5:
  - Worked towards completing our requirements document
  - Client had also provided us with a mock up of what our application could look like

- Week 6:
  - Met with our client in Portland. They helped us to better define the requirements that needed to be completed
  - Finished our requirements document

Project Recap

- Week 7:
  - Began work on the tech review
  - Split the requirements into three major parts: Searching and gathering data, UI and presenting data, and purchasing the items
  - Each team member focused on the parts that they had been assigned

- Week 8:
  - Created mock design of what the application would look like
  - Each member had also completed their part of the tech review
  - The review featured many different technologies that we would be using, discussing the benefits of each one

Project Recap

- Week 9:
  - Large portions of the design document were completed
  - The group created a UML to demonstrate what classes might be used during development
- Week 10:
  - Completed our design document
  - Began work on fall progress report

Devin's Functional Requirements

- Checkout/Purchase
- Single Item View
- Checkout Page

Hector's Functional Requirements

- Home Page
- Search Results List View
- Search Notifications

Bryan's Functional Requirements

- Search for items
  - Allow user to filter their search
- Save user's preference
- Fix user's search keyword

Technologies: Checkout, Checkout UI, Item View Page

- Checkout options:
  - Checkout as guest user
  - Checkout after signing in
  - Checkout with sign in and guest user
- Checkout UI:
  - Indigo Studio
  - Just In Mind
  - Android Layout Editor
- Getting data for single item view:
  - GSON Parser
  - JsonReader
  - Native Android JSON Parser

Technologies: Home Page, Search List View, Search Notifications

- Home Page
  - Simple Interface
  - Search Text Box with Submit Button
- Item List View
  - Present results in a list
  - Minimal details of each item
  - Item Filtering
- Search Notifications
  - No results based off search input
  - No Results based off filters used

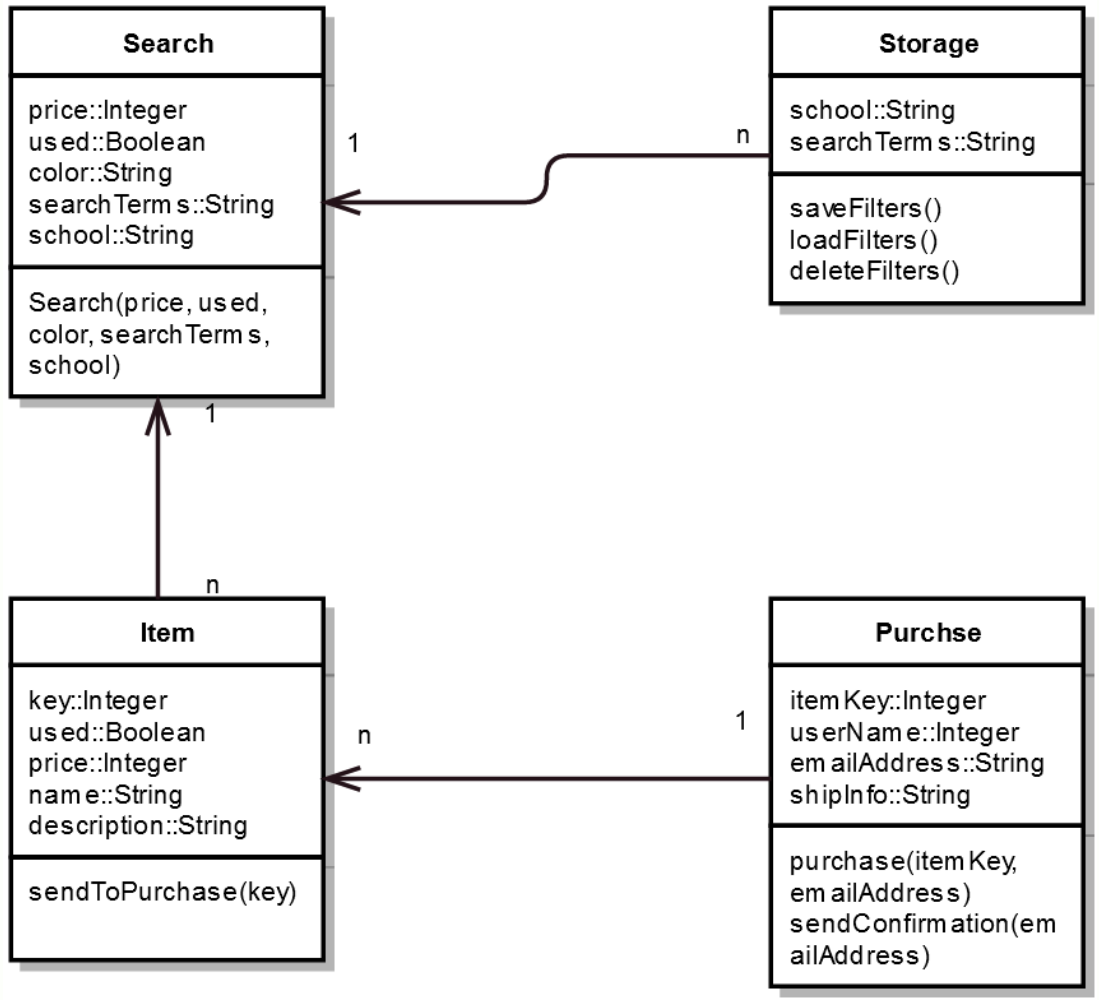
Technologies: Search Items, Save Preferences, Fix Keywords

- Search Items:
  - Use Ebay's API
  - Use Binary Search
  - Use Linear Search
- Save Preferences:
  - Save on internal storage
  - Save on external memory card
  - Save on a server
- Fix Keywords:
  - Levenshtein Distance Algorithm
  - Metaphone
  - Hamming Distance Algorithm

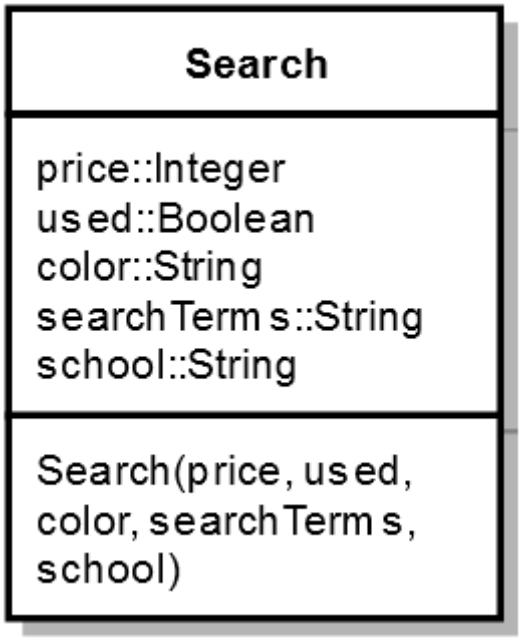
Project Context

- The context of the project is from the viewpoint of the user
- Two main functions the user will perform: search and purchase

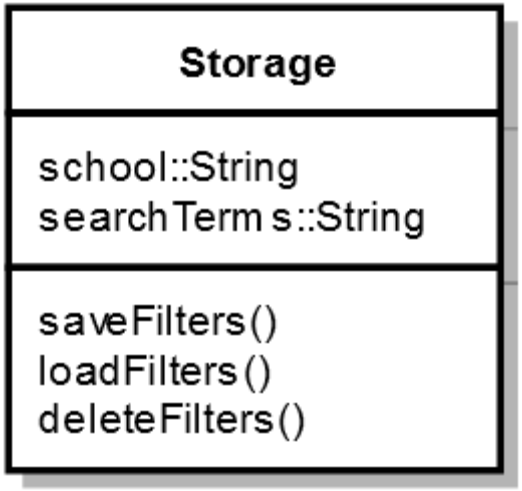
Project UML



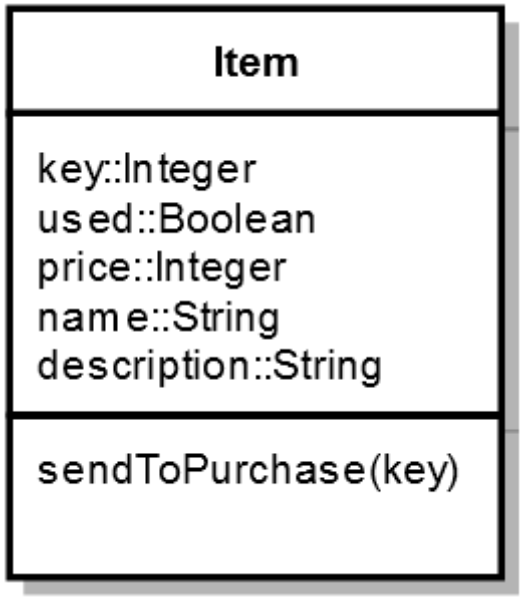
The Search Class



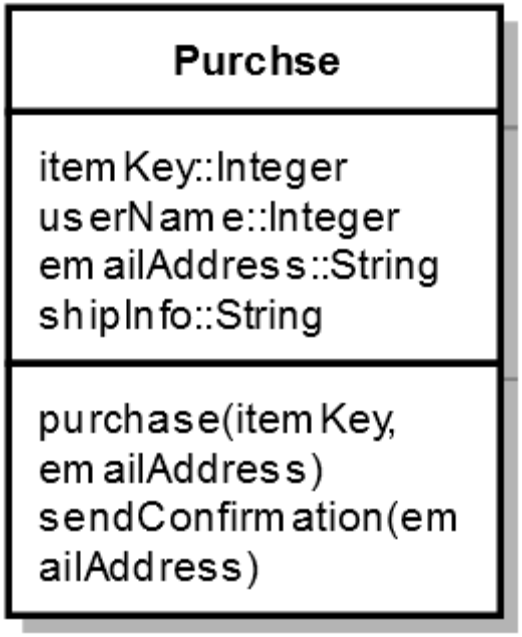
The Storage Class



The Item Class



The Purchase Class



Interaction Viewpoint

- Home Page
  - Type in search
  - Submit search
- Item List View
  - Display list of search results
  - Present minimal but important details
  - Filter search results
- Search Notifications
  - Bad input
  - Too many filters

Algorithm viewpoints (Levenshtein Distance)

- Will run regardless of errors
  - Fix will only happen if the minimum distance is not 0 and reaches a certain threshold
- Concerns regarding complexity
  - Might run slow since we are comparing keywords on a database
  - Can be rectified by having a sorted database
    - Stop once a minimum threshold is reached
- Algorithm involves creating 2D matrix with length of string 1's length times string 2 length

Algorithm pseudocode

- Given a string s, target string t and matrix m, the algorithm will fill each spot in the matrix using the minimum from this rule:
  - If s[n] == t[n] where n is != max(t.stringlength,s.stringlength), m[n][n] = 0, else if s[n] != t[n], m[n][n] = 1
  - m[a][b] where a < s.stringlength and b < t.stringlength = m[a-1][b]+1 or m[a][b-1]+1 whichever is smaller
  - m[a][b] = m[a][b] where a < s.stringlength and b < t.stringlength = m[a-1][b-1]+1
- The Distance output will be m[s.stringlength][t.stringlength]

In the end…

- We learned a lot from this class and became better software developers because of it
- Next term, we will begin development of our application
- We look forward to completing the beta of our project

~ FIN ~