SW Engineering CSC648/848 Summer 2016

Application: Gatorslist

Milestone 2

July 15th, 2016

Rev 2.0

Team 1

Khalid Alrashed (alrashed@mail.sfsu.edu),
Eric Chen, Robert Chung,
Tai Nguyen, Guoyi Ruan.

Table of Contents

1. Use Cases

User-Friendly Home Page

John is an SFSU student currently in his sophomore year. John had a bad experience buying textbooks directly from the campus store because of how expensive they are, and so this time he decides to go to Gatorslist. On Gatorslist he is presented with a simple layout of a variety of items for sale and their titles underneath, sorted by how recently they have been added. He can then easily navigate through the list of items and look search for the textbooks or any other item he needs at an affordable and competitive price.

Search (Unregistered User)

Consuela has bought tickets for game 7 of the NBA Finals; however two days before the big game, she has an emergency situation at home and cannot attend. She has a friend who is interested in purchasing the tickets from her but the problem is, she doesn't know how much they are worth. She had only bought them for retail price but knows the demand for the tickets has caused the price to increase dramatically. She doesn't want to overcharge her friend but wants to sell it at a fair price for herself as well. Looking around for buy and sell websites, she discovers Gatorslist and successfully browses through the content on the website without registering for an account. She finds several listings for tickets with similar seating positions compared to her's and decides to write down the prices. Using the prices she obtained from Gatorslist, she was able to determine a fair price to sell the tickets to her friend for. Although she wasn't able to see the game in person, she was still able to watch it from home while her friend made full use of the tickets.

Register and Sell Products

Rebecca is graduating from SFSU this semester. When she first started attending SFSU, she became interested in collecting designer handbags and over the years at SFSU she has amassed a massive collection as a result. But now she's planning to move out, and doesn't have the space to take her handbags with her. Most of the bags are brand new and she could use the extra money living on her own so she decides to go to Gatorslist and sell them. On the website's homepage she is presented with an intuitive website layout which leads her to quickly understands how to sell the bags on the site. As she navigates to the product upload page, she is notified that she must register first. After completing the account registration, she returns to the same product upload page and is presented with several required forms to be completed before she can list her bag for sale. Fortunately, there are only few essential properties of the item that needs to be filled out. After submitting the item for sale, she realizes that she a huge mistake and forgot a 0 for the price of her bag. She scrambles to try to delete and remake the listing before anyone can contact her but she realizes she can edit the existing listing instead. After making the changes, she proceeds to upload the rest of her designer handbag collection on Gatorslist and makes a small profit.

Register and Buy Products

Marc is on the 2nd month of his second semester at SFSU, and realizes that he should've bought a new laptop for his major classes. The nearby electronic stores including the Apple store have all sold out of the new Macbook he has been eyeing; he is told it won't be in stock anywhere for the next few months. Marc cannot wait a few months because the semester will be over by then. Searching around the web, he finds Gatorslist. On the site, he searches for the Macbook he desires and by some miracle, there is listing from another student selling the Macbook for only a small amount above retail price. Driven by excitement, Marc quickly views the listing details to check if it's a scam. He sees the description and large images of the Macbook and surely enough, they are legitimate. He tries to contact the seller through the listing details page but is told he that feature is not available because he has not yet registered. Afraid someone might buy it before him, Marc rapidly fills in the simple information fields and creates a new account. He enters his preferred payment method as cash and sets up a meeting with the seller. The transaction is successful and with peace of mind knowing he was able to buy the Macbook he wanted without waiting several months, he confirms the transaction on Gatorslist.

Administrator

Max is a retired software developer from Silicon Valley. He has often found himself bored with retirement and so he has found a new job working for a buying and selling startup website called Gatorslist as an administrator. It isn't the most demanding work for Max but it keeps him occupied and using the tools provided to him, he is easily able to keep Gatorslist safe from any malicious content by managing content uploaded by registered users. Aside from approving product upload requests made by the users, he can edit and delete both user accounts and product listings that violate Gatorslist's terms of service.

2. Data Definition

2.1 Term Description

Term		**Description**
Unverified User	Guest	Browse the website. Search products. View products.
Verified User	Buyer	Have all rights of unverified users. Create, login and logout account. Order products on the website. Review seller.
	Seller	Have all rights of unverified users. Create, login and logout account. Upload and delete products. Set/Reset a price for products.

	Administrator	Have all rights of unverified users. Login and logout account. Manage all accounts. Manage all products.
Product		Information about a particular product includes: ● Product title. ● Product id number. ● Seller's id number. ● Price. ● Image. ● Post date. ● Condition. ● Status.
Search		Our site provides search: ● By category. ● By keyword.
Filter		Filter provides better control to achieve more effective and efficient searches. Our site provides filter: ● By price. ● By condition. ● By date posted.
Registration Information		Includes: ● First name ● Last name ● Username ● Password ● Email Address
Order Information		Information about a particular order includes: ● Order number. ● Product id number. ● Buyer's id number. ● Order Date. ● Detail. ● Price ● Status.

2.2 Users Permission

	Guest	Buyer	Seller	Admin

Browse website	+	+	+	+
Search products	+	+	+	+
View products	+	+	+	+
View user's profile	+	+	+	+
Reset their own password		+	+	+
Edit their own profile page		+	+	+
Order products		+		
Cancel order		+	+	
Rate seller		+		
View order history		+		
View saved search		+	+	+
Add products			+	
Edit products			+	
Delete products			+	+
Set/reset price			+	
Delete user account				+

'+': Permission approved.

3. Functional Requirements

Guests

Priority 1:

- Guests shall be able to search for and browse items best matched to the search criteria they provided. [1]
- Guests shall be able to filter through their search results based on a predetermined set of filtering options. [1]
- Guests shall be able to sort through search results based on a predetermined set of sorting options. [1]
- Guests shall be able to register for a new account using their SFSU student ID. [1]

Priority 3:

- Guests shall be able to sort through search results based on academic major. [3]

Registered Users

Priority 1

- Registered users shall be able to search for and browse items best matched to the search criteria they provided. [1]
- Registered users shall be able to filter through their search results via a predetermined set of filtering options. [1]
- Registered users shall be able to sort through search results based on a predetermined set of sorting options. [1]
- Registered users shall be able to upload their items for sale listings. [1]
- Registered users shall be able to edit their items for sale listings. [1]
- Registered users shall be able to send and receive messages from other registered users. [1]
- Registered users shall be able to edit personal information such as phone number onto their account profile. [1]

Priority 2

- Registered users shall be able to review other users they have completed transactions with. [2]

- Registered users shall be able to report any listings that violate the terms of service. [2]

Priority 3

- Registered users shall be able to upload their class schedule. [3]
- Registered users shall be able to sort through search results based on academic major. [3]

Administrators

Priority 1

- Administrators shall be able to edit or delete any listings that violate the terms of service. [1]
- Administrators shall be able to edit or delete any students accounts that violate the terms of service. [1]

General

Priority 1

- System shall have a simple and user centric designed interface. [1]
- Users shall be able to navigate through the website with only basic knowledge of computing. [1]
- Account information and item listings shall be stored on a secure server. [1]
- All servers shall be stored using best industry practices to protect user security. [1]

4. Non-Functional Requirements

Application

- Application shall be developed using class provided LAMP stack.
- Application shall be developed using pre-approved set of SW development and collaborative tools provided in the class. Any other tools or frameworks have to be explicitly approved by Marc Sosnick on a case by case basis.
- Application shall be hosted and deployed on Amazon Web Services as specified in the class.
- Application shall be optimized for standard desktop/laptop browsers, and shall render correctly on the two latest versions of all major browsers: Mozilla, Safari, Chrome and IE. It shall degrade nicely for different sized windows using class approved programming technology and frameworks.
- Application shall be served from the team's account
- Application shall be very easy to use and intuitive. No prior training shall be required to use the website.

Data

- Data shall be stored in the database on the class server in the team's account
- Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.

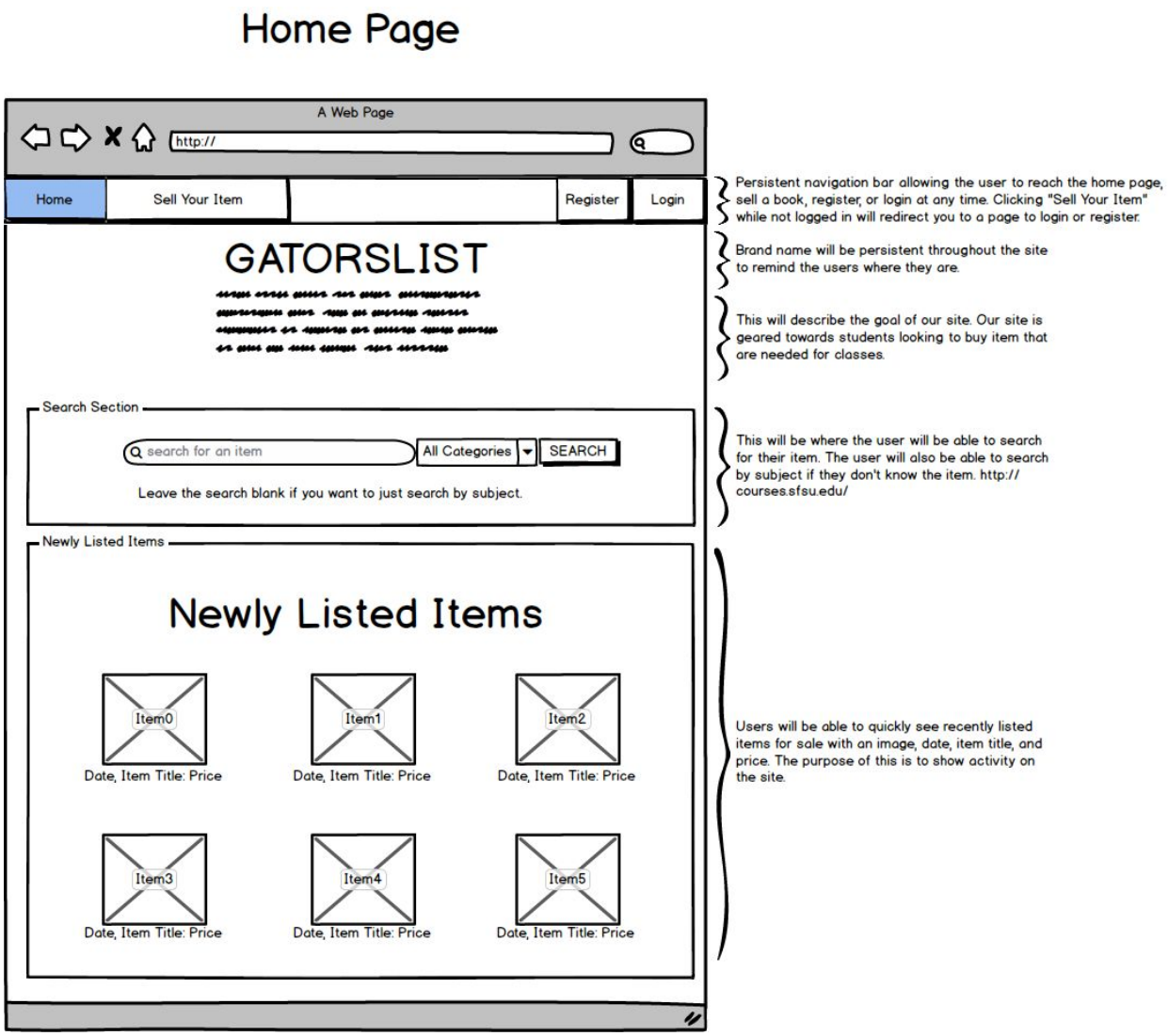
Website

- The language used shall be English.
- No more than 50 concurrent users shall be accessing the application at any time
- Google analytics shall be added for major site functions.
- Messaging between users shall be done only by class approved methods to avoid issues of security with e-mail services.
- Site security: basic best practices to be applied (as covered in the class)

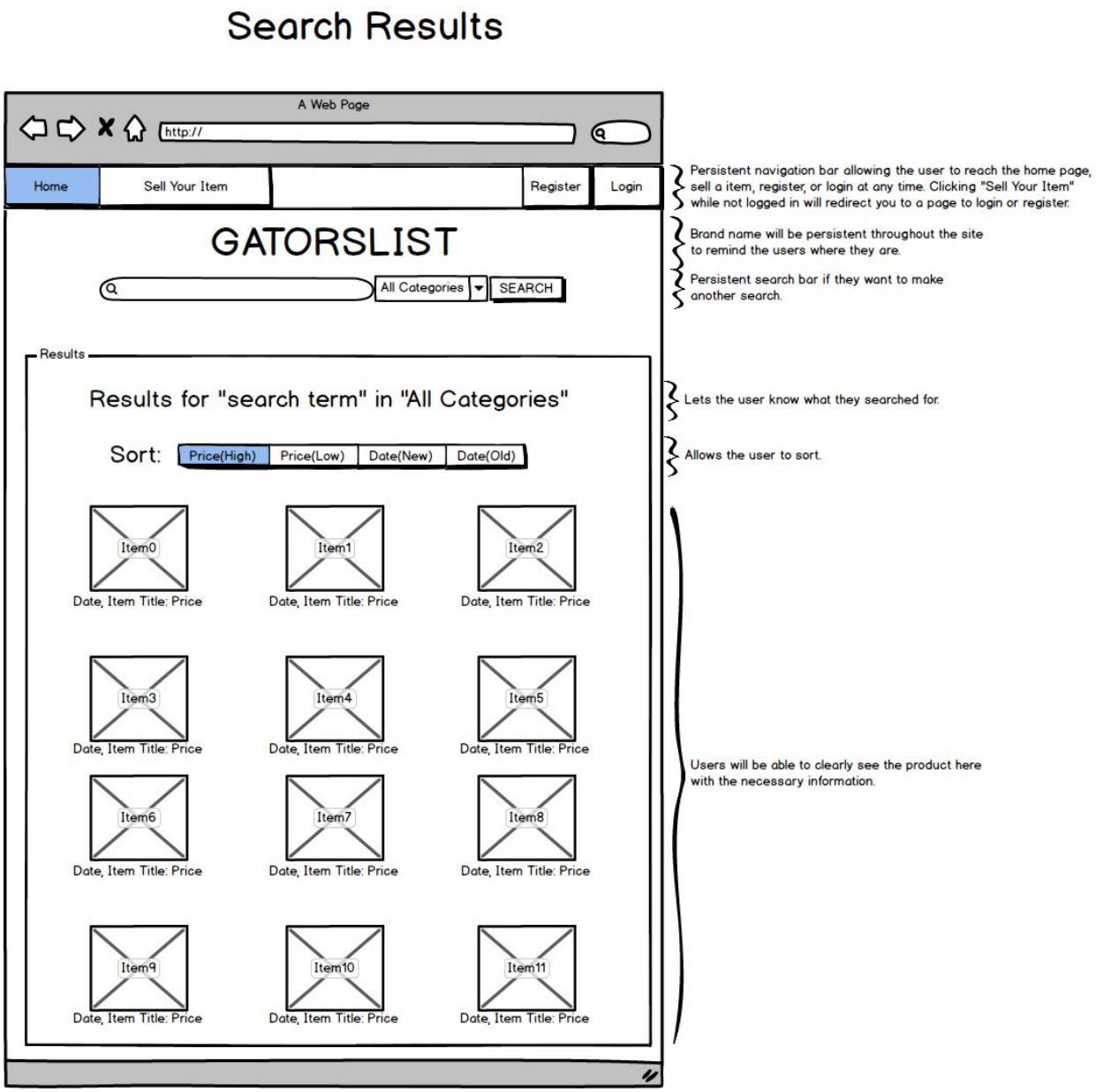
- Modern SE processes and practices must be used as specified in the class, including collaborative and continuous SW development, using the tools approved by instructors
- The website shall prominently display the following text on all pages "SFSU/FAU/Fulda Software Engineering Project, Summer 2016. For Demonstration Only". (Important so as to not confuse this with a real application).

5. UI Mockups and Storyboard

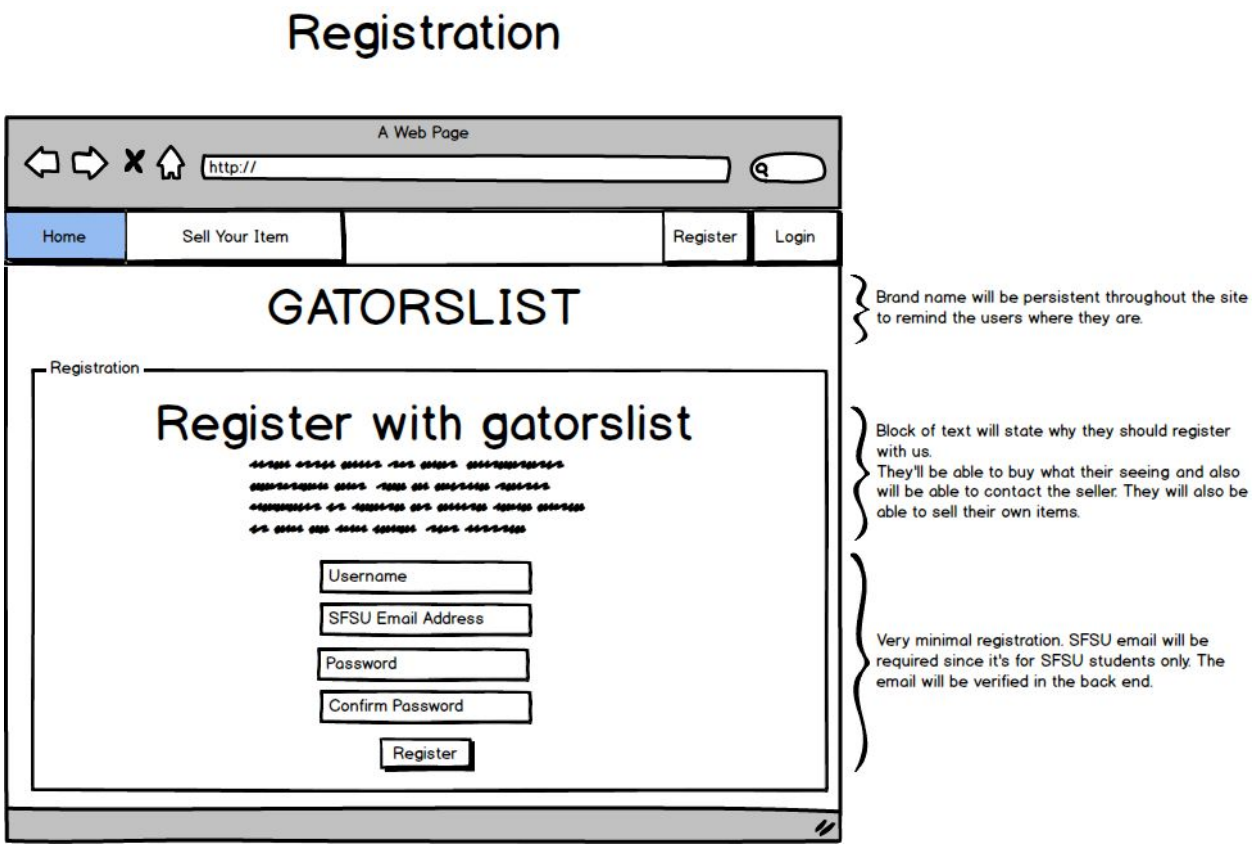
Home: The user is presented with a persistent toolbar with buttons that respectively lead to selling your items and registering/logging into the website. There is a search bar with a drop down menu for item categories and a section for newly listed items.



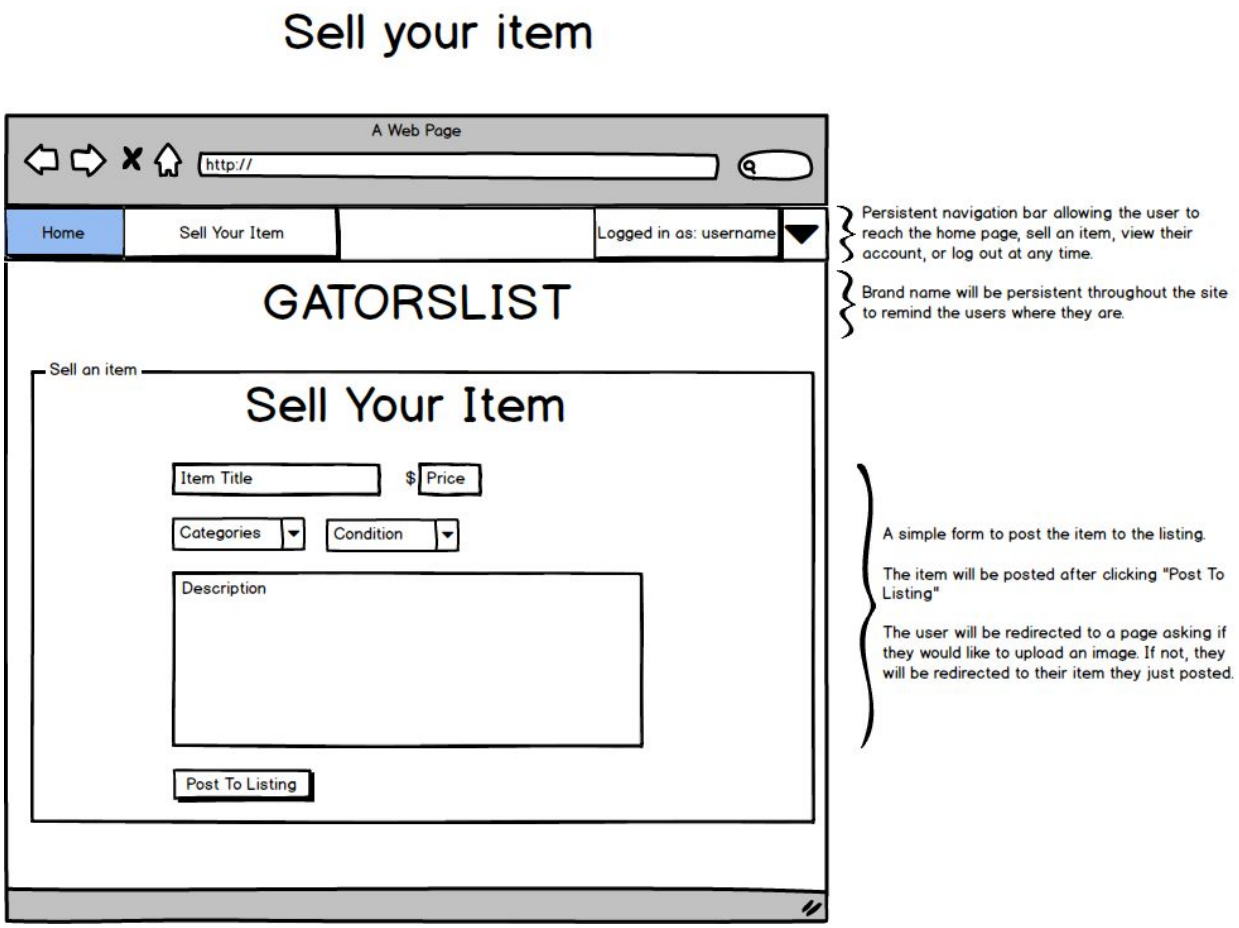
Searching and Search Results: The user is able to search through listings based on a specific category. From these listings, the users are able to sort and filter through them.



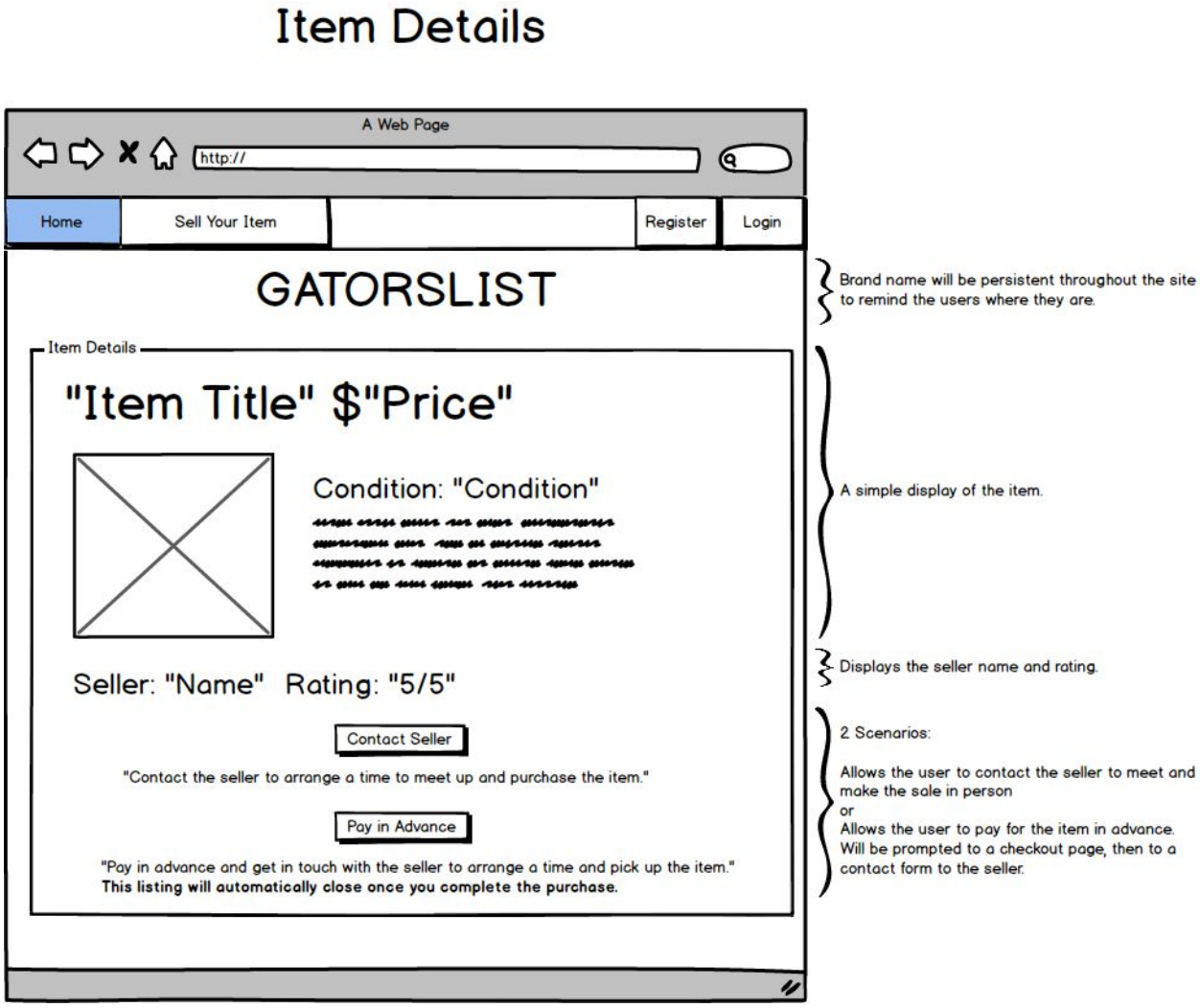
Registration: The user is linked to a registration page if he/she had no association with the system before. The successful registration will let them to be granted an account to starting buying or selling item.



Selling an item(product): The user is presented with several blank fill in forms necessary to list an item for sale. Included are the price, title, condition, item description and images of the item.

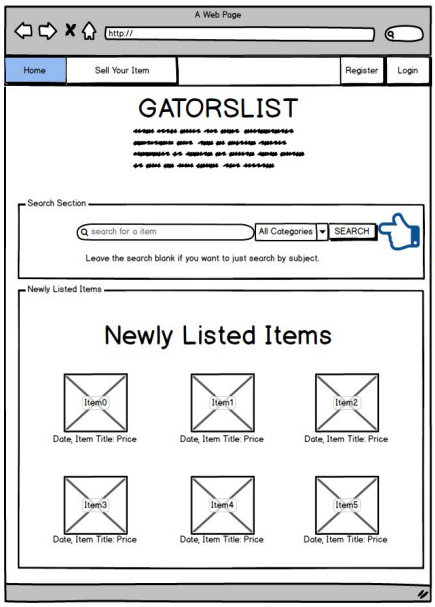


Sample item listing: The item listing page displays the essential information that was entered by the seller. At the top will show images of the item and next to it will be the title, price and condition of the item. Underneath these sections, there is a large area reserved for the description of the item. At the very bottom is a button to allow for users to contact the seller.



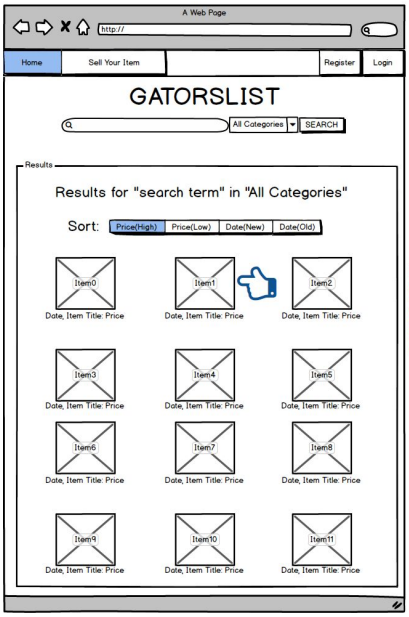
Buyer StoryBoard: (From left to right, top-down)

Home Page



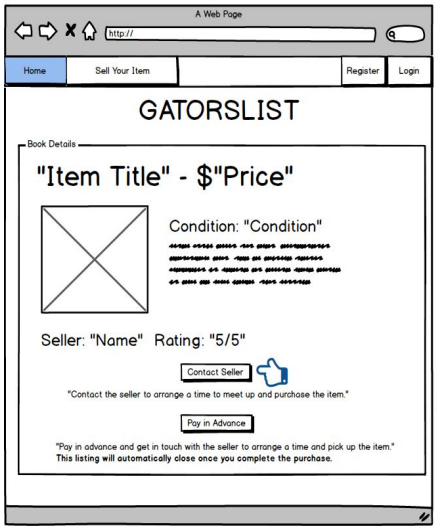
After John access to the homepage address. He starts searching his item by entering its name on the search field.

Search Results



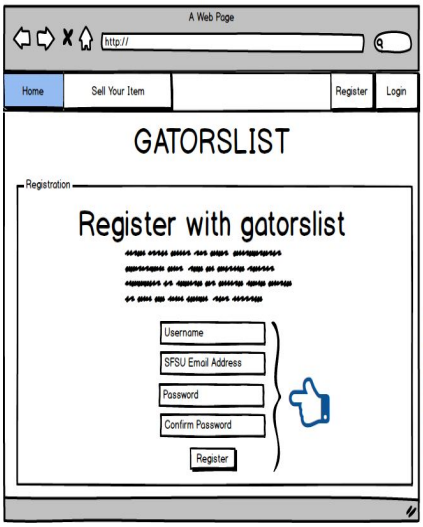
If his item was found, the result page would display the all available items that matched with his search. John then can choose one of them to view its detail in which he can see its name, price.

Item Details



After John had clicked to one of the items displayed in search page, he now is able to view its detail. He found that he likes the item and would like to contact the seller to purchase it. In order to do this, he clicked the contact seller button.

Registration



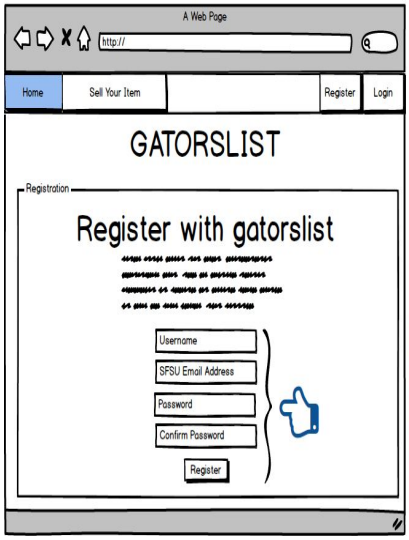
Then, John linked to the page where he must register in order to continue to see the seller's contact. After filing all of his legal information and clicked registration, he was granted an account to officially view the seller's contact.

Seller_StoryBoard: (From left to right, top-down)

Home Page

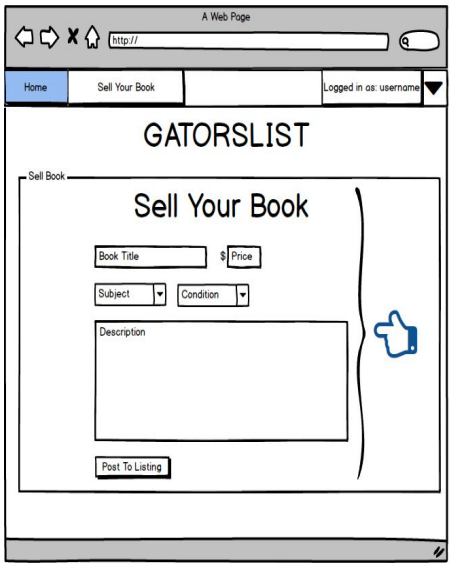
Adam would like to sell his book which he was remainning from previous semesters. He comes to our website address and start selling one by clicking the "Sell your Item" button.

Registration



Then, John linked to the page where he must register in order to continue to see the seller's contact. After filing all of his legal information and clicked registration, he was granted an account to officially view the seller's contact.

Sell your book



The successful registration would lead Adam to a page where he can put information of his book. After filing all legal data of his book and clicked submit button, his book now is available for people who would like to seek and buy it.

6. High Level Architecture, Database Organization

The following list below describes our main software components, products, APIs, tools and systems we will be using to create our final project:

LAMP stack

- **Linux**
  Our application will use Linux platform, and it will be hosted using Amazon Web Services (AWS), which is a subsidiary of Amazon.com and offers a suite of cloud computing services that make up an on-demand computing platform.

- **Apache**
  Our application will be served through the Apache HTTP web server, which is the world's most used web server software. This will allow users to access our application using the web.

- **MySQL**
  The database will be managed using MySQL to store information such as users, invoices, items for sale and tutors. The database will be useful to store and retrieve meaningful data using SQL statements.

- **PHP**
  The use of PHP scripting language will allow us a create a dynamic website and it will nicely work with MySQL and Apache server. These technologies are free to license.

Frameworks and APIs

- **MINI**
  MINI is an extremely simple and easy to understand skeleton PHP application, reduced to the max. It simplifies the MVC pattern needed for the web app.

- **Bootstrap**
  Bootstrap is the most popular HTML, CSS, and JavaScript framework for developing responsive web apps. Using this framework will allow us to use templates, themes, and other components.

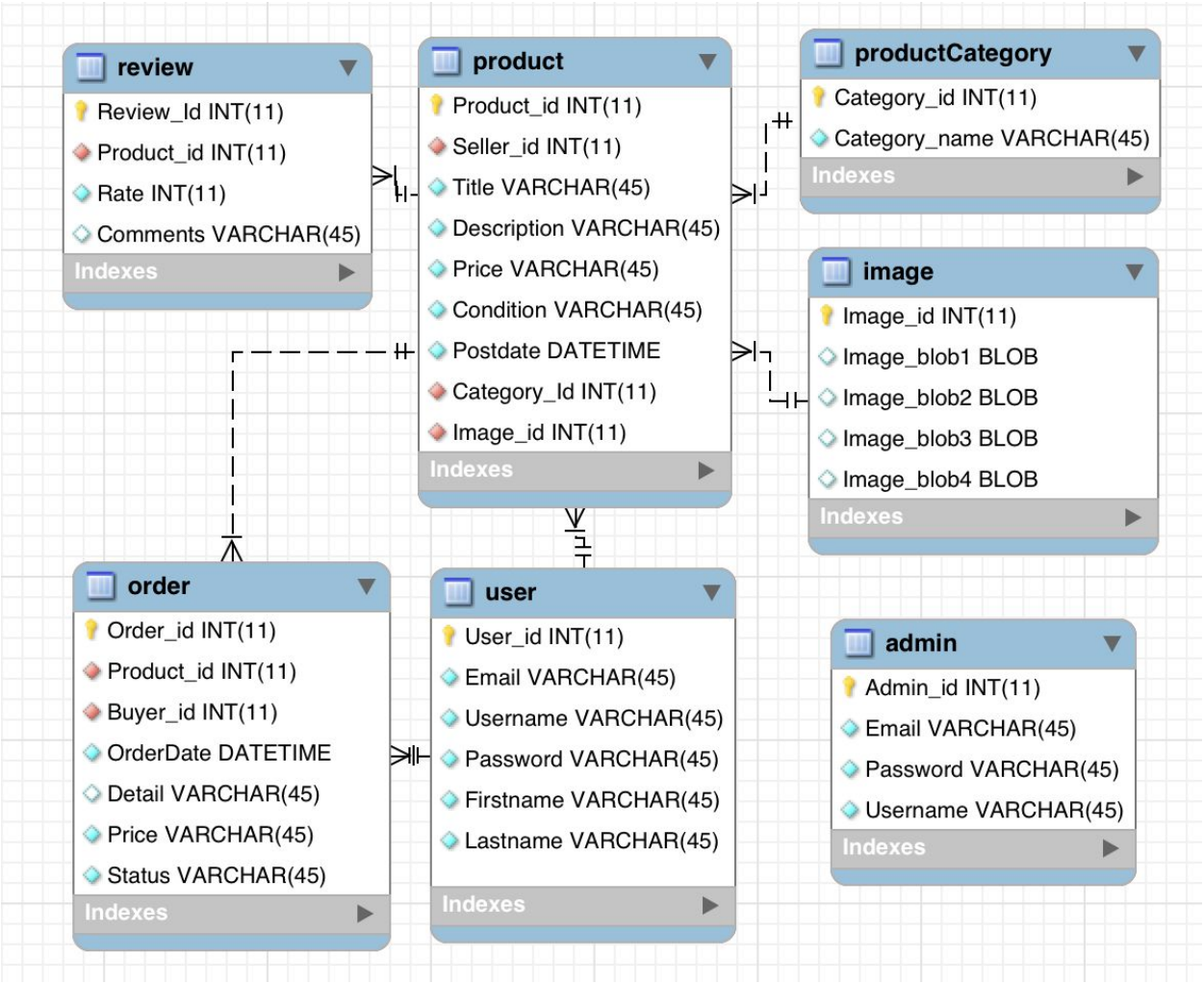
- **jQuery**

A cross-platform JavaScript library designed to simplify the client-side scripting of HTML. It uses its own API and provides wrappers to existing functions. This allow us to write less and cleaner JavaScript code.

- **Google Analytics API**
  Use to collect, configure, and report on user-interactions with the online content.

- **Supported Browsers**
  Our application will be optimized to run in a standard/laptop browsers and it will support the latest two versions of the following browsers:
  1. Chrome
  2. Safari
  3. Firefox
  4. IE

Database Organization



User Table:

Users	
User_id - int	PK
Username – varchar	
Firstname - varchar	
Lastname - varchar	
Email - varchar	
Password - varchar	

Admin Table:

Admin	
Admin_id - int	PK
Username – varchar	
Email - varchar	
Password - varchar	

Product Table:

Product	
Product_id - int	PK
Seller_id – int	FK
Category_id – int	FK
Image_id – int	FK
Title - varchar	
Description- varchar	
Price- decimal	
Condition -varchar	
Postdate - datetime	

FK: Seller_id refer to user table;
Image_id refer to image table;
Category_id refer to productCategory table;

ProductCategory Table:

ProductCategory

Category_id - int	PK
Category_name - varchar	

Order Table:

Order	
Order_id - int	PK
Product_id – int	FK
Buyer_id – int	FK
OrderDate - datetime	
Detail- varchar	
Price- decimal	
Status - varchar	

FK: Buyer_id refer to user table;
Product_id refer to product table;

Review Table:

Review	
Review_id - int	PK
Product_id – int	FK
Rate – int	
Comments - varchar	

FK: Product_id refer to product table;

Images:

Image	
Image_id – int	PK
Product_id – int	FK
Image_blob1 - blob	
Image_blob2 - blob	
Image_blob3 - blob	
Image_blob4 - blob	

FK: Product_id refer to product table;

Confirm Table:

Confirm

Comfirm_id - int	PK
User_id – int	FK
Key– varchar	
Email – varchar	

FK: User_id refer to user table;

Image:

All images shall be stored and retrieved in the image table as BLOB's. It's easier to keep all database entries valid with foreign keys, and easier for maintenance.

Search Algorithm:

Search algorithm will be enabled via SQL query based on entries in the product table.
We provide search by keyword:

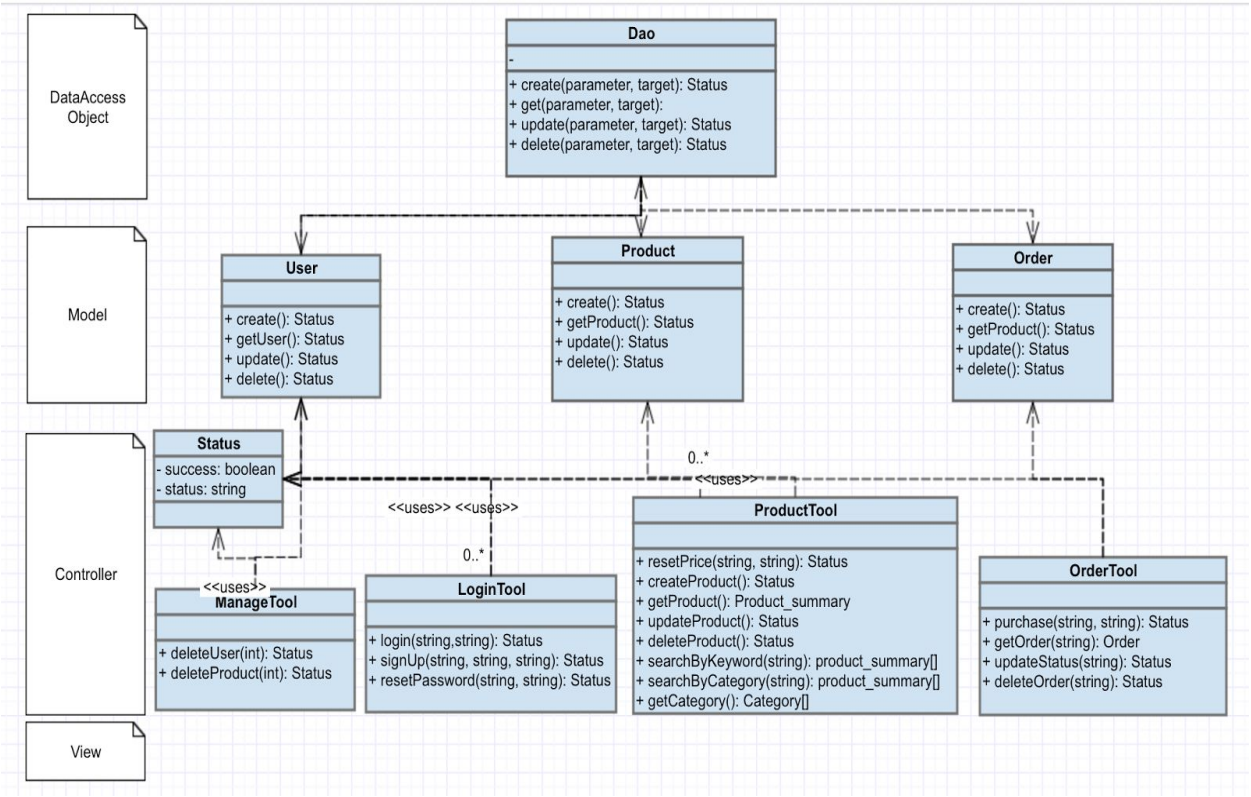
```
SELECT * FROM `product` WHERE description LIKE '%{$keyword}%';
```

And search by category:

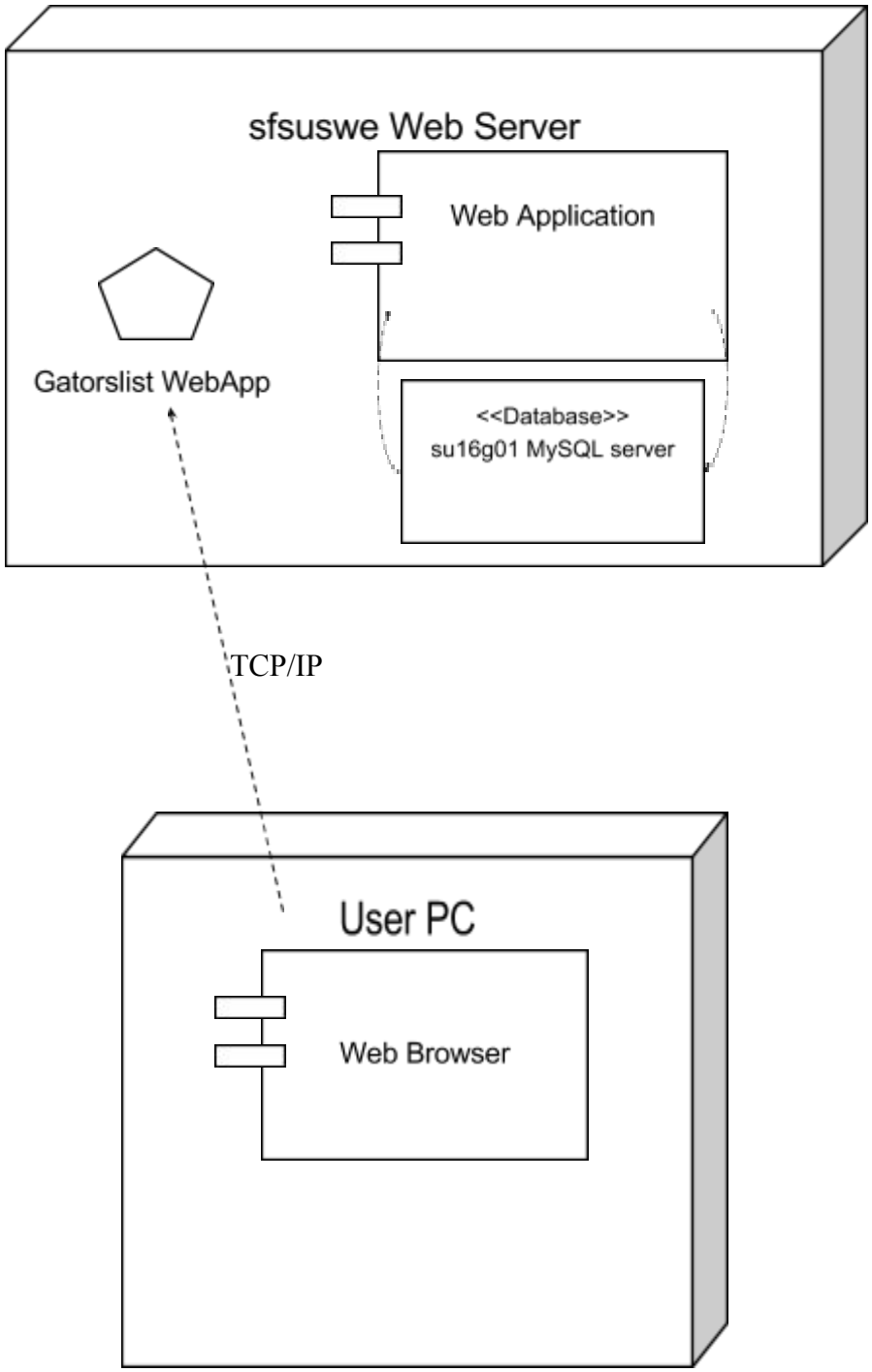
```
SELECT * FROM `product` WHERE category_id = $category_id AND  description LIKE '%{$keyword}%';
```

7. High Level UML Diagrams

- High-level UML class diagram



- UML component and deployment diagram



8. High level APIs

Status:

- Boolean: success
- String: status

Login Tool:

- Function *login* takes the information from both the email and password forms that the user fills in. If the email is in the database and the supplied password is correctly corresponding to that email in the database, it will return a status of whether the login was successful.

  + login(email STRING, password STRING)
    Return: Status

- Function *signUp* takes the string username, email and checkpassword. Usually user fills their password two times. The two passwords are checked whether they are equal and if they are, the username, email and password is newly passed into the database as a new user account with the appropriate data fields inserted. It will return a status depending whether account creation has been successful.

  + signUp(username STRING, email STRING, checkpassword STRING)
    Return: Status

- Function *resetUserPassword* takes the username, old password and new password from the user and if the old password is correctly linked to the username in the database, the new password will replace it in the data field. It will return a status depending on if the password has been updated.

  + resetUserPassword(username STRING, oldpwd STRING, newpwd STRING)
    Return: Status

Product Tool:

- Function *getProduct* obtains the product id and uses it to search the database and display all of the other corresponding data fields from the same table. These data fields are used to populate the product page with all of the related information.

+ getProduct(product_id INT)
    Return: Null - if the product ID is not in the database
        Associated array includes
            String: Seller's name
            String: Title
            Double: Price
            String: Condition
            Datetime: Postdate
            Image

- Function *getResultsByKeyword* takes in the keyboard the user has input into the search bar and searches through the database for any listings that include that keyword. If it has found any listings, it will return all related product id's and if there are no available listings, null.

    + getResultsByKeyword(keyword STRING)
        Return: Null – if no products contains keyword
            Associated array includes
                String: Seller's name
                String: Title
                Double: Price
                String: Condition
                Datetime: Postdate
                Image

- Function *getResultsByCategory* takes a category id as an argument. Using this category id, it will search through the database and if there are any listings with that value of category id, it will return them. Otherwise, it will return null if there are no items with that category id available.

    + getResultsByCategory(Category_id INT, keyword STRING)
        Return: Null – if no category ID in the database
            Associated array includes
                String: Seller's name
                String: Title
                Double: Price
                String: Condition
                Datetime: Postdate
                Image

- Function *resetPrice* will take the product id and the new price that the buyer wants to set their product for. It finds the row containing that product id and replaces the old value of the price data field with the new desired price.

    + resetPrice(product_id INT, newprice DOUBLE)
        Return: Status

- Function *deleteProduct* accepts the product id as an argument and it will use it to delete the corresponding row from the database table containing products.

    + deleteProduct(product_id INT)
        Return: Status

Order Tool:

- Function *getOrder* accepts an order id as an argument and using it, the function searches the database for any orders with that particular order id and if it is found, it will return the rest of the row from the database table like the price and the order date.

    + getOrder(order_id INT)
        Return:Null – if order ID is not in the database
            Associated array includes
                String: Product name
                String: Buyer's name
                Double: Total
                Datetime: OrderDate
                String: Status
                String: Location
                String: Detail

- Function *purchase* takes both product id and the buyer's user id as parameters; it will return whether the purchase has been successful depending whether the buyer has sent the payment and the payment has properly been receiver by the seller.

    +purchase(product_id INT, user_id INT)
        Return: Status

Manage Tool:

- Function *deleteUser* will take the user id of the user's account that is selected to be deleted. The entire row corresponding to that user id will then be deleted from the database and return whether it has been successfully deleted.

    +deleteUser(user_id INT)
        Return: Status

- Function *deleteProduct* accepts the product id as an argument and it will use it to delete the corresponding row from the database table containing products.

    +deleteProduct(product_id INT)
        Return: Status

9. Key Risks

Skill Risks

- The group is inexperienced with the technologies used for this project
- Resolution: Everyone will have to dive right in and learn the basics of the technologies regardless of role.

Schedule Risks

- There are some scheduling conflicts where we are unable to spend time in person as a group for more than an hour. All members are taking additional classes, the group, tech, and another member have part time jobs.
- Resolution: We will have to schedule skype meetings to make up for the lack of face time.

Technical Risks

- There are risks with the use of git most of us have not used git hands on yet. We have never used MySQL workbench or worked with a database before.
- Resolution: We will be teaching each other about git and MySQL as we progress.

Teamwork Risks

- Tasks may be not completed in a timely manner.
- Resolution: We have set up a task sheet to keep track of what everyone's doing and to make sure that we're on track and not doing unnecessary tasks.

Legal/content Risk

- There may be risks of unauthorized usage of content/SW.
- Resolution: We will be reading the terms of service carefully as we use them.

10. Team Organization

Name	Role
Khalid Alrashed	Team Lead: Will be looking over the back-end for any issues and also finalizes and formats all documentation, on top of overall project management.
Eric Chen	Tech Lead/Frontend: Will be mainly working on the front-end, the workflow of the group, quality assurance, and user experience.
Robert Chung	Backend: Developer for the back-end features of Gatorslist.
Tai Nguyen	Frontend: Will be working on the front-end with analyzing the logic of website flows based on documentation.
Guoyi Ruan	Backend: In charge of Database and back-end development.