San Francisco State University

SW Engineering CSC648/848 Spring 2020
"Gator Crawl"

Section 02 Team 01

Francis Cruz: Fcruz2@mail.sfsu.edu
Zain Khan
Jugal Bhatt
Jeffery Wan
George Freedland: gfreedland@mail.sfsu.edu
Huy Phan

M1 Date Submitted	M1 Date Revised
2/25/20	3/1/20

1. Functional Requirements - prioritized

1. Must Haves:

a. Users

   i. A easy user registration process

      1. Our website is targeted only for SFSU students and teachers.For users to access the website, they are required to have an account with us. The user will have to register with their school email. The user will need to set a username, password and user typer (student/teacher)

   ii. User has to login

      1. For a user to be able to view or sell a product, they would need to be logged in. Once The login credentials will be their username and password.

   iii. User can post Items to sell

      1. If a user wants to sell an item on our website. They would need to upload a picture of the item to the site, alongside a description and price.

   iv. User can search for Items

      1. Users can use the provided search bar to look up specific items by authors, classes, product types

   v. Users can view an item and description

      1. Once a user searches and selects an item, they will be able to view the picture and description of the item.

   vi. Users can add items to their cart

      1. If a user likes an item and wants to buy it, they can add it to the shopping cart to purchase once they are done browsing.

   vii. Users can buy and purchase items

      1. Once a user is done browsing, they can purchase the items in their carts through us. They can buy either in cash, credit/debit or paypal.

   viii. Users can bid on items for a cheaper price

      1. Sellers have an option to set their products as a bidding option. This can allow buyers to purchase a product at a lower price.

   ix. Users can build profiles

      1. Users can build a profile for their account. This can display their information about themselves. This will also show the items and products that the user is selling.

x. Users can filter there searches by categories /classes
   1. Filtering will help the user locate their item faster because you can narrow down the search results

b. Admin:
   i. Delete items if necessary
      1. Admin can delete posts if it conflicts with our terms and services. Admin will then message seller with reason to delete post
   ii. Admin will be able to view items before public
      1. Admin will be able to view each item before it reaches the public to ensure the item falls under Gator Crawl's terms and services
   iii. Admin will be able to accept items
      1. Admin will be able to accept items to be public
   iv. Admin will be able to reject items
      1. Admin will be able to reject items if they deem it to be violating our terms and services.

2. <u>Desired:</u>

a. User can message other users if they have question about a product
   i. Each item will have a "Message Seller" function which will allow a buyer to communicate with the seller. This will allow the buyer to ask questions about the product.

b. Users have an option to use exchange program
   i. For the buyers who are not able to afford a service or product, they have an option to pay the seller with another product or service provided by the buyer. Meaning you pay for an item with another item which is equal or higher value.
   ii. Both buyer and seller agree upon an exchange before the product is sold.

c. User can close item
   i. If a buyer and a seller agree on an exchange or different price, the seller can close the item for sale.

d. User has an inbox for messages from other buyers/sellers
   i. On each page you can go to your inbox on the top toolbar to check your messages. You will be notified when you receive a new message from someone else.

3. **Opportunistic**

a. Admin

  i. Admin can send a prewritten report for accepting/rejecting new items.

    1. In the admin dashboard when processing new items, If item is accepted, there will be a pre-written message sent to the selling saying that their item has been approved. If rejected, Admin will select from a dropdown of reasons why item was rejected and a pre written message sent to the user explaining why the item was rejected.

b. Users

  i. Users can enable a dark theme

    1. Users can enable a dark theme which can be less strain on the eyes at night. This will change the white background to black and darken the accents.

  ii. User can Add comment/review for current product/services

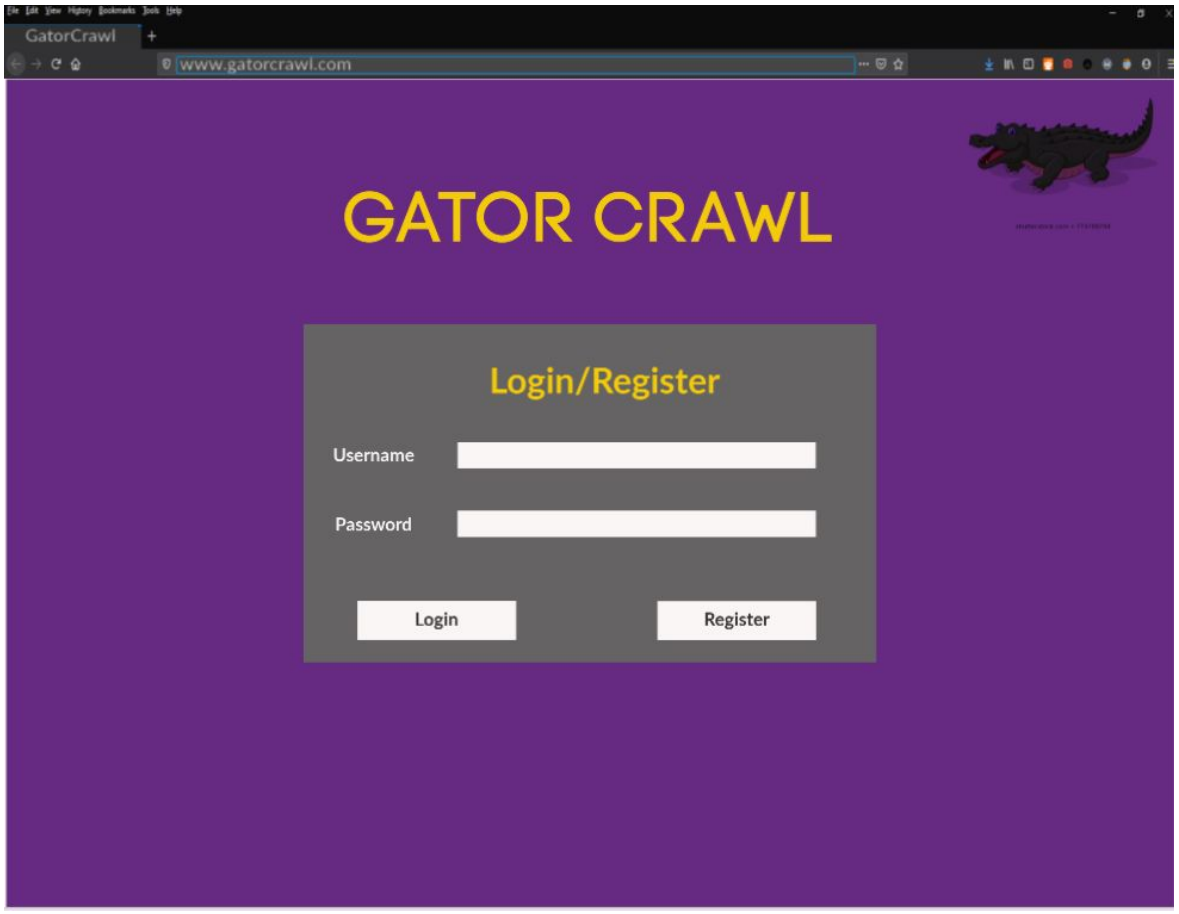
    1. Buyers can leave reviews on sellers profile rating the products and services that they received.

  iii. Users can save their ideal form of payment (credit/debit/ cash/etc)

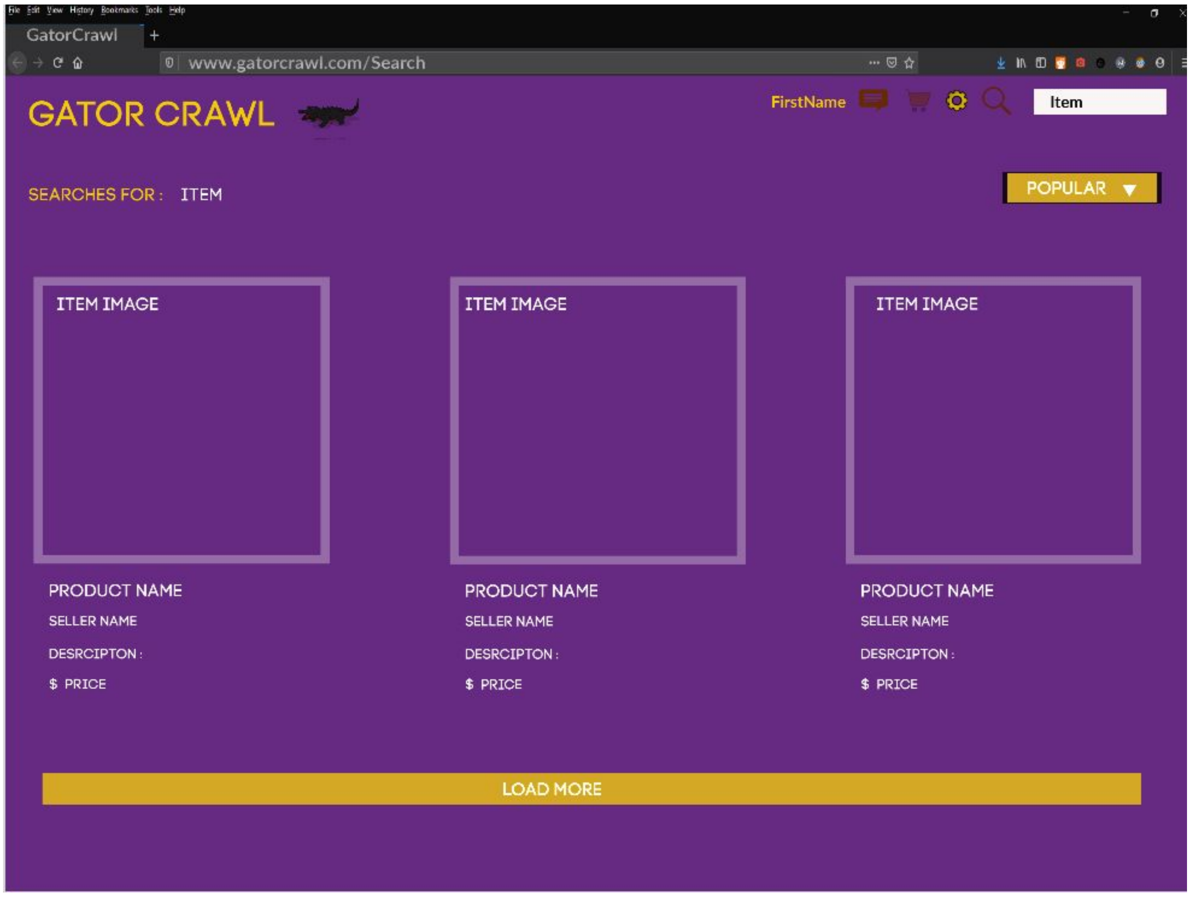
    1. Users can set up their ideal form of payment. They can upload a debit or credit card to be used for purchasing. Users also have an option to link their paypal to their account.

2. UI Mockups and Storyboards (high level only)

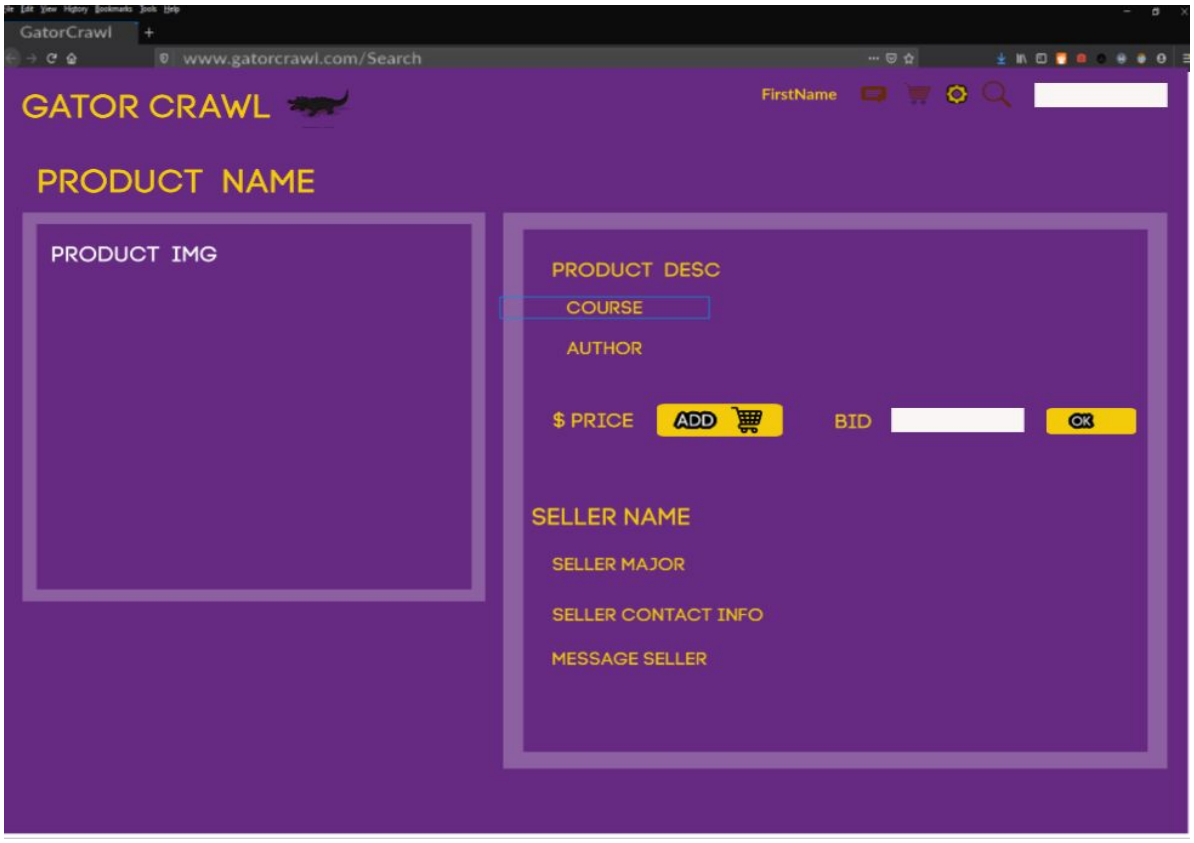
Landing Page



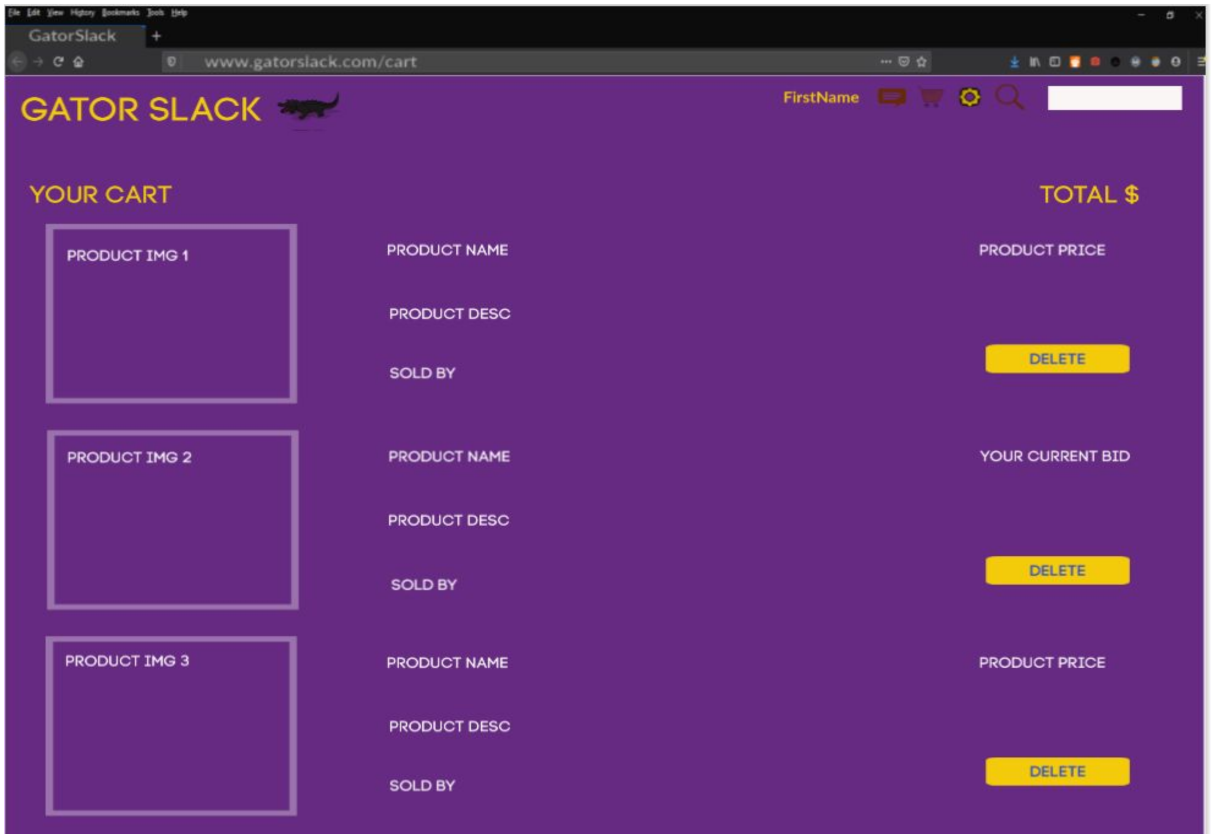
Search Results



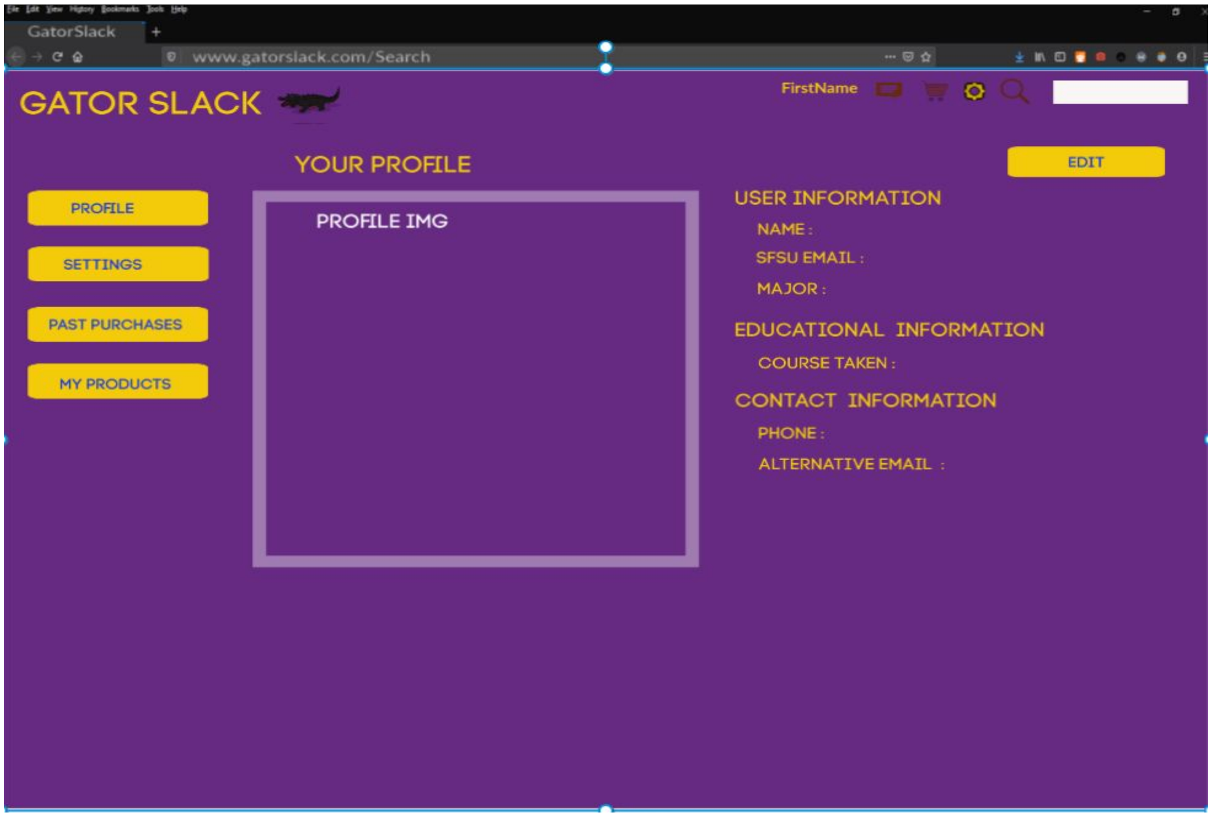
Product Page



Shopping Cart



User DashBoard



USER INFORMATION

NAME :

SFSU EMAIL :

MAJOR :

EDUCATIONAL INFORMATION

COURSE TAKEN :

CONTACT INFORMATION

PHONE :

ALTERNATIVE EMAIL :

3. High level Architecture, Database Organization

DB Organization

Users
SFSUID INT PRIMARY KEY,
FirstName VARCHAR(30) NOT NULL,
LastName VARCHAR(30) NOT NULL,
Major VARCHAR(30) NOT NULL,
Username VARCHAR(30) NOT NULL UNIQUE,
Email VARCHAR(100) NOT NULL UNIQUE,
EncryptedPassword VARCHAR(250) NOT NULL,
ProfilePhoto VARCHAR(100),

**Roles include 0:Admin, 1:Student, 2:Teacher

Roles
RoleID INT PRIMARY KEY,
SFSUID INT NOT NULL
RoleName VARCHAR(30) NOT NULL

Bids
BidID INT PRIMARY KEY,
BidderID INT NOT NULL
ProductID INT NOT NULL
Price DECIMAL(10,2) NOT NULL
FOREIGN KEY (BidderID) REFERENCES Users(SFSUID)
FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

Sales
SaleID INT PRIMARY KEY,
BuyerID INT NOT NULL
SellerID INT NOT NULL

ProductID INT NOT NULL
Price DECIMAL(10,2) NOT NULL
FOREIGN KEY (BuyerID) REFERENCES Users(SFSUID),
FOREIGN KEY (SellerID) REFERENCES Users(SFSUID),
FOREIGN KEY (ProductID) REFERENCES Products(ProductID)

Categories
CategoryID INT PRIMARY KEY,
CategoryName VARCHAR(20) NOT NULL

Messages
MessageID INT PRIMARY KEY,
MessagerID INT NOT NULL,
MessageeID INT NOT NULL,
Message TEXT NOT NULL,
FOREIGN KEY (MessagerID) REFERENCES Users(SFSUID),
FOREIGN KEY (MessageeID) REFERENCES Users(SFSUID)

Products
ProductID INT PRIMARY KEY,
ProfilePhoto VARCHAR(100) NOT NULL
CategoryID INT NOT NULL,
Description TEXT
ProductName VARCHAR(50) NOT NULL
SellerID INT NOT NULL
StartDate TIMESTAMP,
EndDate TIMESTAMP,
Bidding BOOL NOT NULL DEFAULT ‘f’
Price DECIMAL(10,2)
FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID),
FOREIGN KEY (SellerID) REFERENCES Users(UserID)

Media Storage

We will be using DB BLOBS for our file storage. We will use Amazon S3 buckets to store and retrieve product photos.

Search/Filter Architecture

We will primarily use the product name for our search queries. Using the sql LIKE operator and some helper functions provided by the ORM Sequelize, our default search will search our products table for a product name containing a substring of a given query with the regex %like%. In the search results page, the user will then be given options to narrow their search with dropdown menus for filters such as, categories and price range. Once satisfied with their filter options, the user will then click an apply filter button that will narrow the search. These filters will be added onto the previous query with the sql AND operator, as well as the BETWEEN operator for price search.

Own APIs

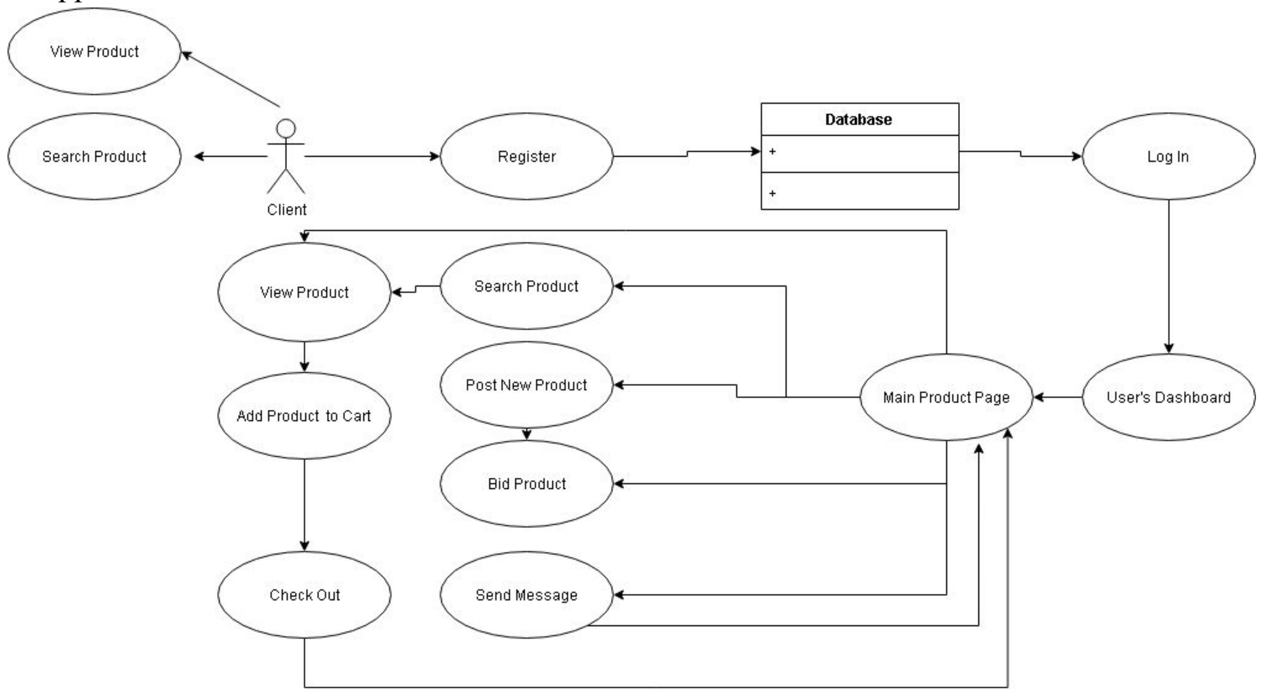
Our main API will be used to search and retrieve products, conduct sales, and send messages from user to user, as well as hidden functions used only for admin. These admin functions include deleting and approving products for sale.

Non-trivial algorithm or processes

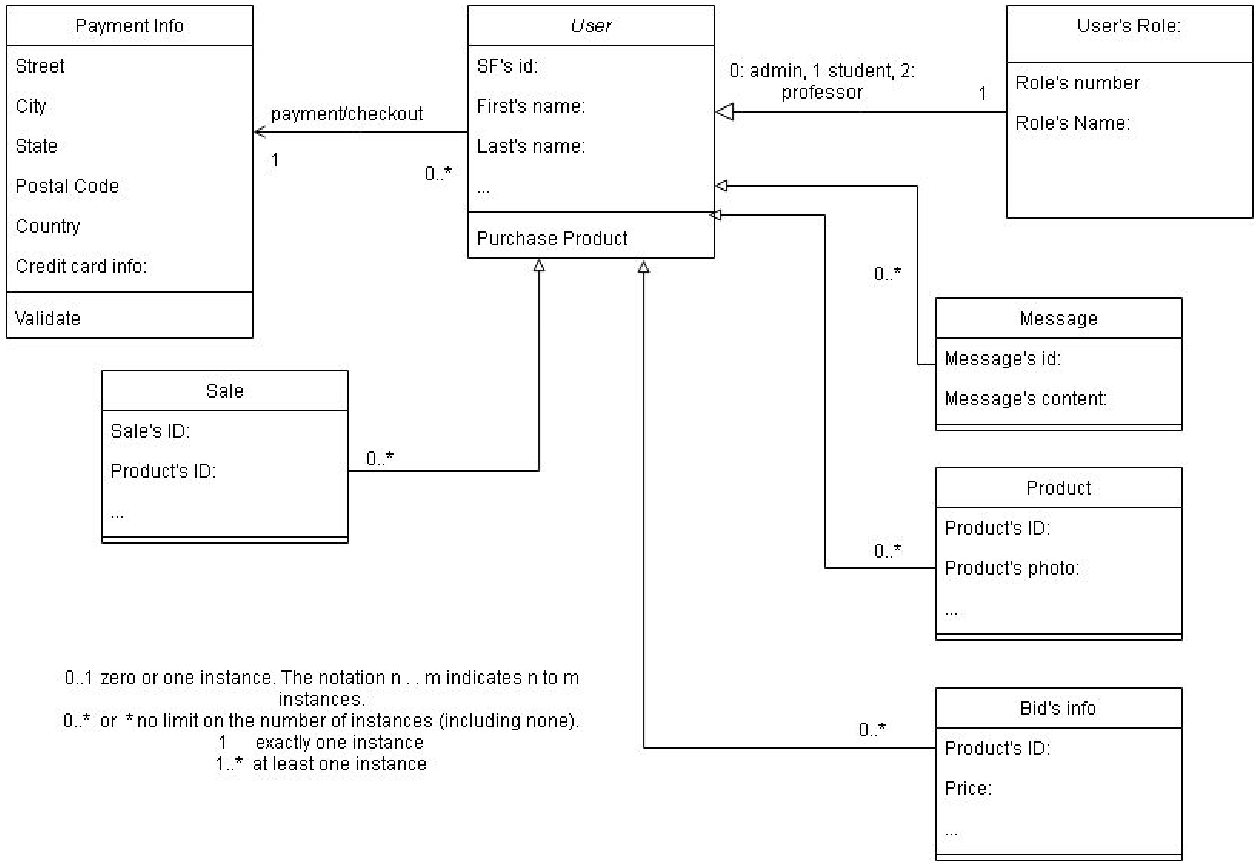
N/A

4. High Level UML Diagrams

Main web app:



b) *UML Component and deployment diagrams*



0..1 zero or one instance. The notation n . . m indicates n to m instances.
0..* or * no limit on the number of instances (including none).
1 exactly one instance
1..* at least one instance

5 .Identify *actual* key risks for your project at this time

Skill risks:

There isn't too much risk for our team because we have a lot of resources on our Notion page like tutorials for our exact fullstack framework, like a JS React + Redux Tutorial and a Server with NodeJS tutorial. Everybody can be on the same page just by looking through our team resources.

As a solution, If someone doesn't know a certain framework/language, a team lead will be more than happy to have a Skype session and help understand the code to team members by sharing screens.

Schedule risks:

Getting our commits in on time is not a big issue but is always a possible risk for any team.

For this, as a solution, we use Jira to keep everybody on track with their tasks. Having a good Slack Channel is also very important to keep everybody up to date. We also constantly update our GitHub so more and more documentation is available and it's easy to see when an assignment is done and how many commits have been made.

Technical risks:

We need to have our front-end implementation with the back-end server and our mySQL Database running all together on the cloud, AWS EC2. As a team we are making necessary adjustments for connections between front-end and back-end frameworks. The front-end team needs to get information from the back-end team and vice versa in order to establish endpoints and perfectly functionally working data transfers.

By using Slack it is much easier to collaborate with the whole team on our project. We can send endpoints and have a system with Swagger.io to document our endpoints so that any front-end developer will be able to easily send requests from the website.

Teamwork risks:

A problem may occur when a team member isn't doing their task. This could be stressful when a deadline is coming up.

As a solution we use Slack and have our notifications on so that we are always in the loop. We can always meet in-person, being SFSU students, and discuss how they feel about their task and if they would be happier doing a different part of the project, or if they just need help. If this liability is still happening, we can bring it up to the team lead and the CTO for further inspection.

Legal/Content risks:

There shouldn't be any legal/content risks at this point because all the media and coding environments we use are open source and anything we create using their software is creative commons. However a problem may arise if a user uploads a copyrighted image for their product.

Luckily we have a solution for that. We have an administrator that can verify the account and media uploaded to make sure it complies with our Terms and Services Agreement and have the ability to take the post down before legal problems arise.

6. Project management

We will be using Jira to assign and keep track of tasks. We use the slack integration to have all issues and sprints in one workplace.