

Gator Market

SW Engineering CSC 648/848
Spring 2025

TEAM 4

Team Lead: Clark Batungbakal
cbatungbakal@sfsu.edu

Front-end Lead: Thanh Duong

Front-end member: Daniel Lee

Back-end Lead: Caleb Onuonga

GitHub Master: Trinity Godwin

Version	Date Submitted	Revised
Ver 1	03/12/2025	03/21/2025
Ver 2	03/25/2025	04/03/2025
Ver 3	04/25/2025	04/27/2025
Ver 4	05/16/2025	5/20/2025
Ver 5	05/20/2025	

TABLE OF CONTENTS

PRODUCT SUMMARY	3
MILESTONE DOCUMENTS	4
PRODUCT SCREENSHOTS	50
DATABASE ORGANIZATION	55
GITHUB ORGANIZATION	58
GOOGLE ANALYTICS STATS	60
PROJECT MANAGEMENT	60
TEAM MEMBER SELF-ASSESSMENT AND CONTRIBUTIONS	67

PRODUCT SUMMARY

Name of the product?

GatorMarket

Product Description

GatorMarket is a student-exclusive online marketplace built for the San Francisco State University (SFSU) community. It provides a safe and convenient platform where students can buy, sell, and trade items directly with one another. GatorMarket makes it easy to find or list what students need within a trusted network, from textbooks and electronics to furniture and school supplies. Unlike public platforms like Craigslist or Facebook Marketplace, GatorMarket verifies users through their @sfsu.edu email address, ensuring all members are part of the SFSU community. This adds a layer of trust and campus relevance that external platforms lack.

The application is designed with simplicity in mind. Students can post listings with images and descriptions, search by keyword or category, and contact sellers directly through a secure messaging system. GatorMarket reduces friction in student transactions by providing an all-in-one interface that is tailored to the university's needs, helping students save money, declutter, and connect locally.

Final Priority Features

These are the core functions the GatorMarket team is committed to fully delivering and testing for the final release:

- Users can register using a valid @sfsu.edu email address.
- Users must accept the terms of service during registration.
- Passwords are securely encrypted before storage in the database.
- Users can securely log in and log out.
- Logged-in users can create new listings with:
 - Title
 - Description
 - Image upload
 - Category
 - Price
- Listings are displayed in a browsable, responsive grid layout.
- Users can filter listings by category or keyword.
- Users can search listings using a search bar (up to 40 alphanumeric characters).
- Users can view detailed information for individual listings.
- The website is accessible and mobile-friendly.

Note: While our posting and messaging features are designed and partially implemented, they are not currently functioning end-to-end due to unresolved issues with data persistence and live communication. We've identified where the breakdown occurs and plan to continue debugging after the final demo.

Uniqueness & Custom Features for SFSU

What sets GatorMarket apart is its exclusive focus on the SFSU community. This platform is tailored to students' and professors' needs, providing a cost-effective, local, and safe way to trade. By limiting access to verified members of the university, GatorMarket creates a trusted environment for buying and selling within the campus community.

URL: <http://13.52.231.140:3001/>

MILESTONE DOCUMENTS

Milestone 1

EXECUTIVE SUMMARY

Why this product?

Students and professors at San Francisco State University (SFSU) frequently encounter difficulties when attempting to buy, sell, or trade various items, particularly during the transitions at the beginning and end of each semester. While alternative platforms such as Craigslist and OfferUp offer general marketplaces, they often come with significant drawbacks, including potential scams, high transaction fees, and unreliable buyers or sellers, making them less than ideal for students.

To address these challenges, GatorMarket provides a secure, student-exclusive marketplace designed specifically for the SFSU community. By ensuring that all users are verified SFSU students and professors, GatorMarket fosters a trusted environment where students can conveniently and safely engage in transactions without concerns about fraud or excessive fees.

Functions and services

GatorMarket is a web-based marketplace created exclusively for SFSU students and professors, offering a secure and scam-free environment by requiring SFSU email verification to restrict access to authorized users only. The platform enables users to buy, sell, and trade a wide range of items, including textbooks, electronics, furniture, clothing, and collectibles such as Pokémon cards. With an integrated real-time chat feature, buyers and sellers can negotiate prices, arrange

meetups, and finalize transactions without depending on third-party messaging apps. To further enhance safety, designated on-campus meetup locations provide a secure space for in-person exchanges, eliminating the risks of public transactions. Additionally, user profiles with ratings and reviews help build trust and credibility within the marketplace. GatorMarket is entirely free and it allows the users to retain 100% of their earnings. Unlike traditional platforms that impose listing or transaction fees. With these comprehensive features, GatorMarket offers a reliable, convenient, and cost-effective platform for the SFSU community to conduct trusted peer-to-peer transactions efficiently.

Uniqueness & Custom Features for SFSU

What sets GatorMarket apart is its exclusive focus on SFSU community. This platform is tailored to students and professor's needs, providing a cost-effective, local, and safe way to trade by integrating features like campus-specific meetup points. Having a safe and exchange points within the university eliminates the risks of meeting strangers in public places. To ensure the student has a secure and reliable experience, GatorMarket has a real-time chat for negotiation, and a rating system to build user trust.

About our team

Our team consists of dedicated and innovative computer science students at SFSU who are passionate about technology, problem-solving, and creating impactful solutions. As members of the SFSU community, we recognize the need for a secure and student-focused marketplace and are committed to developing an application that enhances convenience, safety, and accessibility for our peers. Our goal is to empower students and professors with a trusted platform for buying, selling, and trading within the university. We believe that GatorMarket has the potential to not only streamline campus transactions but also serve as a model for other universities looking to build their own exclusive community-driven marketplaces.

PERSONAE

James

Demographics	Psychographic
Male 47 yrs old Truck Driver Part Time	Value Education Veteran Enjoys Traveling
Pain Point	Goal

Doesn't trust internet people Skeptical of online deals	Not worry about doing trades Trusting environment
--	--

Scenario

James is a 47-year-old part time student in business. He has a wife 3 kids working as a truck driver. Having limited time, he doesn't trust online deals and wants things safe and quick

Kevin

Demographics	Psychographic
Male 21 years old Full time Student Physiology Major	Busy Focused on grades Collector of limited items Good with people
Pain Point	Goal
Very Busy No time to browse Knows what he wants so no nonsense	Finish degree with good marks to get into med school

Scenario

Kevin is 21 years old majoring in physiology aspiring to be a doctor after bachelors. He has a hobby of collecting all sorts of collectibles, whether it be cards or clothes. Leading a busy life with hard goals, he doesn't have time to browse.

Emily

Demographics	Psychographic
Female 22 years old Graduating senior International Student	In a rush Want to finish and go home Sell everything
Pain Points	Goals
Fast and quick No Bogus	Liquidate Everything Go Home

Scenario

Being away from home for 4 years, she is eager to go home. To do that, she has to get rid of everything, but she wants to make some extra money. Selling clothes and furniture at a convenience.

Daniel

Demographics	Psychographic
Male 18 years old Undecided freshman Committed week before start of semester	Rushing Need Furniture Quick Procrastinator
Pain Point	Goal
Does last minute things Need stuff quick	Buy clothes and furniture before school start

Scenario

Daniel did a late commit to SFSU with 1 week to spare, he needs clothes and furniture to move with good filters. He is shopping for essentials as fast as he can.

HIGH LEVEL USE CASES

Use Cases 1: Searching/Browsing

Kevin is a full-time physiology student with a busy schedule and little time to browse listings. He needs a fast and efficient way to find collectibles, clothes, or other essentials without wasting time scrolling through endless listings. To make his shopping experience easier, the platform offers an advanced search and filtering system. Kevin can quickly filter results by category, price range, item condition, and seller rating. He can also sort items by relevance, newest listings, or lowest price to find the best deals. With these streamlined search tools, Kevin can quickly locate what he needs without disrupting his studies, making his buying experience efficient and hassle-free.

Use Cases 2: Posting Items

Emily is a graduating international student who needs to sell all her belongings quickly before leaving for home. Since she has been away for four years, she has accumulated furniture, clothes, and other items she no longer needs. With little time left before her departure, she wants a fast and hassle-free way to list and sell everything. The platform allows her to create a listing in just a few steps. She can upload photos, add a description, set a price, and choose a category. With these features, Emily can liquidate her belongings quickly, avoid last-minute stress, and earn some extra money before heading home.

Use Cases 3: Wishlist Item

Daniel is a last-minute incoming freshman who committed to SFSU just a week before the semester started. He needs to quickly buy essentials like furniture, clothes, and school supplies to settle in before classes begin. However, some of the items he needs are not currently

available on the marketplace. To solve this, Daniel adds the missing items to his Wishlist, allowing him to track availability without constantly checking the site. Whenever a seller lists an item matching his Wishlist, he receives a notification, so he can act fast before someone else buys it. With the Wishlist and notification features, Daniel can efficiently find and purchase what he needs without wasting time, ensuring he is ready before the semester begins.

Use Cases 4: On-Campus Meetup Points

James is a truck driver and part-time student who does not trust online deals and prefers safe, in-person transactions. To address his concerns, the platform offers designated on-campus meetup points where buyers and sellers can safely exchange items. These locations are well-lit, monitored, and frequently visited areas on campus, reducing the risk of fraud or unsafe encounters. Before finalizing a purchase, James and the seller agree on a meetup location from a pre-approved list, ensuring both parties feel secure. With this feature, James can confidently trade without worrying about scams or dealing with strangers off-campus.

Use Cases 5: Creating an Account

James, Kevin, Emily, and Daniel want to start using the marketplace to buy, sell, and trade items within the SFSU community. To create an account, they simply fill out a registration form with their name, SFSU email, and password. The system automatically checks if their email ends with @sfsu.edu to ensure they are part of the university. Once they submit the form, their account is instantly created, allowing them to browse listings, post items for sale, and connect with other students without any additional verification steps. This process makes it fast and easy for SFSU students to join while keeping the marketplace exclusive to the university community.

DATA ITEMS AND ENTITIES

Users - Encapsulates all accounts, buyers and sellers

Buyer - An SFSU student or staff looking to buy items from other SFSU students or staff

Seller - An SFSU student or staff looking to sell items to other SFSU students or staff

Admins - Operators of the website who will help users sort out disputes and issues. They will be reachable through the in-site messaging service for any user

Backend Items:

Users Table- repository for information on user accounts such as username, their relevant SFSU email, rating, history, and other necessary metadata.

Listings Table- catalogue of all currently available items for sale, their price, who's selling them, and any other relevant data to the listing.

Messaging Table- holds information about messages sent.

REQUIREMENTS

HIGH LEVEL FUNCTIONAL REQUIREMENTS

Unregistered Users

1. System shall allow users to browse all available items without requiring login
2. System shall provide search functionality by item name, and type
3. System shall display detailed item information including images, description, and market price
4. System shall allow users to register for a new account
5. System shall provide information about trading policies and platform guidelines

Registered Users

6. System shall allow users to log in securely with username and password
7. System shall enable users to list their own item for sale or trade
8. System shall provide a way for users to set prices and trade preferences for their items
9. System shall facilitate direct messaging between users for trading negotiations
10. System shall allow users to purchase items using the platform's payment system
(platform does not have payment system)
11. System shall notify users of new listings matching their wishlist criteria
12. System shall allow users to rate and review other users after completed transactions
13. System shall provide transaction history for all past trades and purchases

Admin Users

15. System shall allow administrators to approve or reject item listings
16. System shall enable administrators to suspend or ban users who violate platform rules
17. System shall provide administrators with transaction monitoring capabilities
18. System shall allow administrators to resolve disputes between users
19. Admins will not be blockable like regular users

SFSU-Specific Functions

19. System shall facilitate on-campus meetups for item trading between SFSU students
20. System shall provide a special marketplace section for SFSU-exclusive item events
21. System shall verify SFSU student status using school email domains for special privileges

NON-FUNCTIONAL REQUIREMENTS

1. Application shall be developed, tested and deployed using tools and cloud servers approved by Class CTO and as-agreed in M0
2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest version of two major browsers
3. All or selected application functions shall render well on mobile devices (o native app to be developed)
4. Posting of sales information and messaging to sellers shall be limited only to SFSU students
5. Critical data shall be stored in the database on the team's deployment server.
6. No more than 50 concurrent users shall be accessing the application at any time
7. Privacy of users shall be protected
8. The language used shall be English (no localization needed)
9. Application shall be very easy to use and intuitive
10. Application shall follow established architecture patterns
11. Application code and its repository shall be easy to inspect and maintain
12. Google analytics shall be used
13. No e-mail client or chat services shall be allowed. Interested users can only message to sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application
14. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.
15. Site security: basic best practices shall be applied (as covered in the class) for main data items
16. Media formats shall be standard as used in the market today
17. Modern SE processes and tools shall be used in the market today
18. The application UI (WWW and mobile) shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Spring 2025. For Demonstration Only" at the top of the WWW page Nav bar. (Important to not confuse this with a real application). You must use this exact text without any editing.

COMPETITIVE ANALYSIS

Features	Craigslist	OfferUp	GatorMarket
On-Campus Safe Meetup	+	-	+
Real-Time Chat System	+	-	+
Trade Option	-	-	+
Reporting Option	-	-	++

+Feature Exist; ++Superior; -does not exist

Our site's value proposition comes from its community centered approach. It provides all essential features from Craigslist and OfferUp plus more. With a unique campus meetup and trade options exclusive only to SFSU students, safety is ensured. We pride ourselves in community and the safety of our users.

HIGH-LEVEL SYSTEM ARCHITECTURE AND TECHNOLOGIES

Main SW Components

Database - AWS RDS for MySQL ver 8.0

WWW Server: Apache 2.4

Node.js (20.x)

Deployment Cloud Server

AWS free tier - EC2 instance free for 12 months

Frontend Frameworks

Express.js 4.18+

React (Next.js) 19.0.0

Browsers We Support

Chrome (Version 135)

Firefox (Version 136)

GENAI USAGE

Tool:

ChatGPT 4o

Task where it was used:

Executive Summary

Useful rating High

How it was used and benefits:

ChatGPT was utilized to edit the Executive Summary that I had originally wrote down and organize it

Prompt Used:

“Organize and edit the paragraph to make it sound professional”

Tool:

Claude 3.7 Sonnet

High Level Functional Requirements

MEDIUM

How it was used and benefits:

We used gen ai to assist with organizing and formatting the high-level functional requirements into user categories.

Prompt Used:

“Can you organize these high level functionalities into user based categories? Examples - Unregistered Users, Registered Users, Admin Users”

TEAMS AND ROLES

Roles	Name	E-mail
Team Leader	Clark Batungbakal	cbatungbakal@sfsu.edu
Back-End Leader	Caleb Onuonga	jonuonga@sfsu.edu
Front-End Leader	Thanh Duong	tduong7@sfsu.edu
Front-End Member	Daniel Lee	dlee40@sfsu.edu
GitHub Master	Trinity Godwin	cgodwin@mail.sfsu.edu

TEAM LEAD CHECKLIST

ON TRACK	So far, all team members are fully engaged and attending team sessions when required
ON TRACK	Team found a time slot to meet outside of class
ON TRACK	Team ready and able to use the chosen back and front-end frameworks and those who need to learn are working on learning and practicing
DONE/OK	Team reviewed class slides on requirements and use cases before drafting Milestone 1
DONE/OK	Team reviewed non-functional requirements from “How to start...” document and developed Milestone 1 consistently
DONE/OK	Team lead checked Milestone 1 document for quality, completeness, formatting and compliance with instructions before submission
DONE/OK	Team lead ensured that all team members read the final M1 and agree/understand it before submission
ON TRACK	Team shared and discussed experience with GenAI tools among themselves
ON TRACK	GitHub organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.)

APPENDIX

Site Pages / Services:

User creation page - page where new users will be able to register/create their accounts.

Homepage - landing page for all users, logged in or otherwise, will serve as the primary page for looking at listings, finding listings, messaging Admins and other essential activities.

User page - All users will have a user page giving basic information about that user such as their username, access to message them, their rating, number of transactions, how long the account has existed and any other useful information. There will also be a block button. If the user is a Seller, it will also display the user's current listings.

Listings - This page will serve as the interaction portal for buyers to look at a particular seller's item. It will have an image of the card, a link to the sellers User page, a brief description, the seller's asking price, a chat box to send a message to the seller, quantity of available cards if there are multiple, any other relevant information about the card such as if it has been graded, and finally an 'offer' selection tab which will allow a buyer to send a proposed price, meetup location (selected from the predesignated on-campus locations) and a proposed meeting time.

Offers - These are special messages sent to the seller. The message will contain all relevant information including who is trying to buy, proposed location, price, quantity to buy, and meetup time. That seller will be given a certain amount of time to approve or deny. Replying to this message will allow the seller to send a counteroffer or negotiate with the buyer. Once an offer is accepted the listing will be taken off the website.

Blocking Service - Buyers and sellers can block users seamlessly without the blocked user being informed. All messages and offers made by a blocked User will not be shown to the blocker.

Blocked sellers will not have their listings appear when searching / on the Homepage.

Messaging Service - The messaging service will serve as the primary way buyers and sellers are able to communicate and negotiate. Users will be able to go to another user's User page and start a conversation there as well as at the Listing page. The Homepage will have a notification bar that will alert a user to new messages and allow them to respond. This messaging service will also be how users are able to get in touch with the admins.

Wishlist - Users will be able to 'save' or otherwise highlight listings of interest as well as wish list sought-after cards. When a card the user has wish listed has a new listing, the User will automatically be messaged about the new listing.

Location Selection - Users will be only able to meet up at predesignated locations on campus where Admins have evaluated it to be safe. AKA places with cameras and bystanders to help prevent the service being used to target users.

Milestone 2

EXECUTIVE SUMMARY

Why this product?

Students and professors at San Francisco State University (SFSU) frequently encounter difficulties when attempting to buy, sell, or trade various items, particularly during the transitions

at the beginning and end of each semester. While alternative platforms such as Craigslist and OfferUp offer general marketplaces, they often come with significant drawbacks, including potential scams, high transaction fees, and unreliable buyers or sellers, making them less than ideal for students.

To address these challenges, GatorMarket provides a secure, student-exclusive marketplace designed specifically for the SFSU community. By ensuring that all users are verified SFSU students and professors, GatorMarket fosters a trusted environment where students can conveniently and safely engage in transactions without concerns about fraud or excessive fees.

Functions and services

GatorMarket is a web-based marketplace created exclusively for SFSU students and professors, offering a secure and scam-free environment by requiring SFSU email verification to restrict access to authorized users only. The platform enables users to buy, sell, and trade a wide range of items, including textbooks, electronics, furniture, clothing, and collectibles such as Pokémon cards. With an integrated real-time chat feature, buyers and sellers can negotiate prices, arrange meetups, and finalize transactions without depending on third-party messaging apps. To further enhance safety, designated on-campus meetup locations provide a secure space for in-person exchanges, eliminating the risks of public transactions. Additionally, user profiles with ratings and reviews help build trust and credibility within the marketplace. GatorMarket is entirely free and it allows the users to retain 100% of their earnings. Unlike traditional platforms that impose listing or transaction fees. With these comprehensive features, GatorMarket offers a reliable, convenient, and cost-effective platform for the SFSU community to conduct trusted peer-to-peer transactions efficiently.

Uniqueness & Custom Features for SFSU

What sets GatorMarket apart is its exclusive focus on SFSU community. This platform is tailored to students and professor's needs, providing a cost-effective, local, and safe way to trade by integrating features like campus-specific meetup points. Having a safe and exchange points within the university eliminates the risks of meeting strangers in public places. To ensure the student has a secure and reliable experience, GatorMarket has a real-time chat for negotiation, and a rating system to build user trust.

About our team

Our team consists of dedicated and innovative computer science students at SFSU who are passionate about technology, problem-solving, and creating impactful solutions. As members of the SFSU community, we recognize the need for a secure and student-focused marketplace and are committed to developing an application that enhances convenience, safety, and accessibility for our peers. Our goal is to empower students and professors with a trusted platform for buying, selling, and trading within the university. We believe that GatorMarket has the potential to not

only streamline campus transactions but also serve as a model for other universities looking to build their own exclusive community-driven marketplaces.

DATA ITEMS AND ENTITIES

User Types

Users - Encapsulates all accounts, buyers and sellers

Buyer - An SFSU student or staff looking to buy items from other SFSU students or staff

Seller - An SFSU student or staff looking to sell items to other SFSU students or staff

Admins - Operators of the website who will help users sort out disputes and issues. They will be reachable through the in-site messaging service for any user

Product Categories

- **Books** - Academic and recreational reading materials
 - *Textbooks* (Mandatory: Title, Author, ISBN, Condition, Price; Optional: Course Code, Edition, Images)
 - *Fiction* (Mandatory: Title, Author, Genre, Condition, Price; Optional: Images, Description)
 - *Non-Fiction* (Mandatory: Title, Author, Category, Condition, Price; Optional: Images, Description)
 - *Course Materials* (Mandatory: Title, Course Code, Condition, Price; Optional: Images, Department)
- **Clothing** - Apparel for all demographics
 - *Men's* (Mandatory: Type, Size, Condition, Price; Optional: Brand, Color, Material, Images)
 - *Women's* (Mandatory: Type, Size, Condition, Price; Optional: Brand, Color, Material, Images)
 - *Kids* (Mandatory: Type, Size, Age Range, Condition, Price; Optional: Brand, Color, Material, Images)
 - *Accessories* (Mandatory: Type, Condition, Price; Optional: Brand, Size, Color, Images)
- **Collectible Items** - Specialized collector's merchandise
 - *Trading Cards* (Mandatory: Game/Series, Card Name, Condition, Price; Optional: Rarity, Set, Images)
 - *Action Figures* (Mandatory: Character, Series, Condition, Price; Optional: Manufacturer, Scale, Images)
 - *Plushies* (Mandatory: Character, Series, Condition, Price; Optional: Size, Manufacturer, Images)

Backend Items

- **Users Table**- repository for information on user accounts such as username, their relevant SFSU email, rating, history, and other necessary metadata.
 - *User Profile* (Mandatory: Username, SFSU Email, Password; Optional: Profile Picture, Phone Number, Bio)
- **Listings Table**- catalogue of all currently available items for sale, their price, who's selling them, and any other relevant data to the listing.
 - *Item Details* (Mandatory: Title, Category, Subcategory, Condition, Price, Seller ID; Optional: Description)
 - *Item Media* (Mandatory: At least one image; Optional: Additional Images, Video)
 - *Listing Metadata* (Mandatory: Post Date; Optional: View Count, Save Count)
- **Messaging Table**- holds information about messages sent.
 - *Message Content* (Mandatory: Sender ID, Recipient ID, Timestamp, Message Text; Optional: Related Listing)
 - *Message Status* (Optional: Read/Unread)
 - *Chat Threads* (Mandatory: Participants IDs; Optional: Last Activity Timestamp)
- **Search Index** - Facilitates item discovery through search functionality
 - *Searchable Fields* (Title, Description, Category, Subcategory, Price Range, Condition)
 - *Filter Options* (Price Range, Condition, Date Posted, Seller Rating)
 - *Sort Options* (Price: Low to High/High to Low, Date: Newest/Oldest, Relevance)
 - *Chat Threads* (Mandatory: Participants IDs; Optional: Last Activity Timestamp)
- **Blocked List Table**
 - *Account ID*

HIGH LEVEL FUNCTIONAL REQUIREMENTS

Unregistered Users

- 1.1 System shall allow users to browse all available items without requiring login
- 1.2 System shall provide search functionality by item name, and type
- 1.3 System shall display detailed item information including images, description, and market price
- 1.4 System shall allow users to register for a new account
- 1.5 System shall provide information about trading policies and platform guidelines

Registered Users

- 2.1 System shall allow users to log in securely with username and password
- 2.2 System shall enable users to list their own item for sale or trade
- 2.3 System shall provide a way for users to set prices and trade preferences for their items
- 2.4 System shall facilitate direct messaging between users for trading negotiations
- 2.5 System shall allow users to rate and review other users after completed transactions
- 2.6 System shall provide transaction history for all past trades and purchases
- 2.7 System shall notify users of new listings matching their wishlist criteria

Admin Users

- 3.1 System shall allow administrators to approve or reject item listings
- 3.2 System shall enable administrators to suspend or ban users who violate platform rules
- 3.3 System shall provide administrators with transaction monitoring capabilities
- 3.4 System shall allow administrators to resolve disputes between users
- 3.5 Admins will not be blockable like regular users

SFSU-Specific Functions

- 4.1 System shall facilitate on-campus meetups for item trading between SFSU students
- 4.2 System shall verify SFSU student status using school email domains for special privileges
- 4.3 System shall provide marketplace section for SFSU-exclusive item events

Priority 1: (Must have)

Unregistered Users

- 1.1 System shall allow users to browse all available items without requiring login
- 1.2 System shall provide search functionality by item name, and type
- 1.3 System shall display detailed item information including images, description, and market price
- 1.4 System shall allow users to register for a new account
- 1.5 System shall provide information about trading policies and platform guidelines

Registered Users

- 2.1 System shall allow user to log in securely with username and password
- 2.2 System shall enable users to list their own item for sale or trade

2.3 System shall provide a way for users to set prices and trade preferences for their items

2.4 System shall facilitate direct messaging between users for trading negotiations

2.5 System shall allow user to rate and review other users after completed transactions

2.6 System shall provide transaction history for all past trades and purchases

Admin Users

3.1 System shall allow user administrators to approve or reject item listings

3.2 System shall enable administrators to suspend or ban users who violate platform rules

SFSU-Specific functions

4.1 System shall facilitate on-campus meetups for item trading between SFSU students

4.2 System shall verify SFSU student status using school email domains for special privileges

4.3 System shall provide marketplace section for SFSU-exclusive item events

Priority 2: (Desired)

Registered Users

2.7 System shall notify user of new listings matching their wishlist criteria

Admin Users

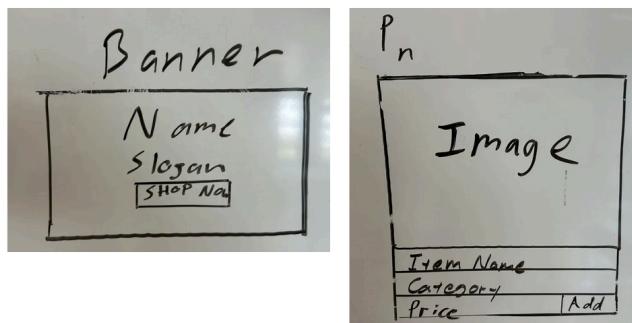
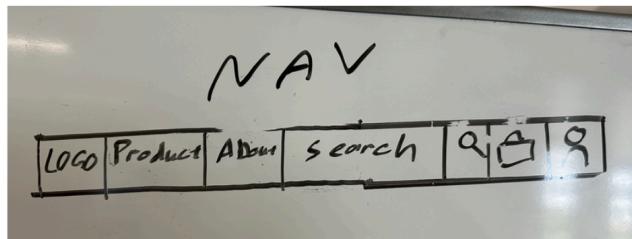
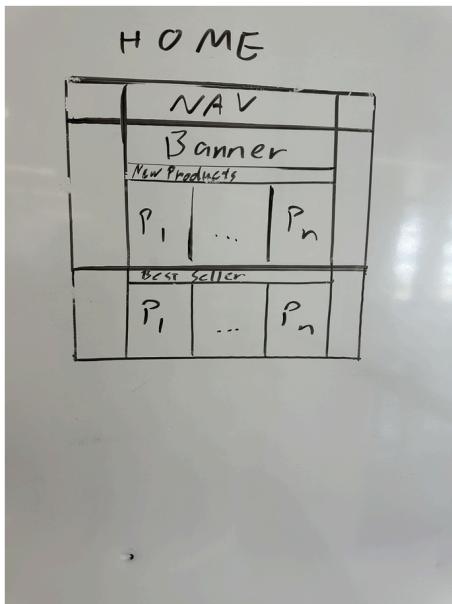
3.3 System shall provide administrators with transaction monitoring capabilities

3.4 System shall allow administrators to resolve disputes between users

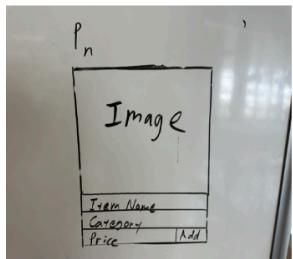
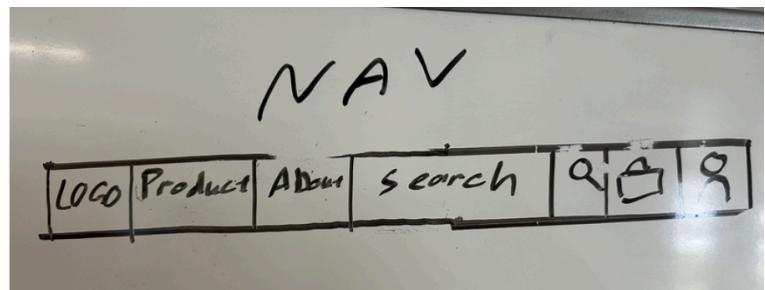
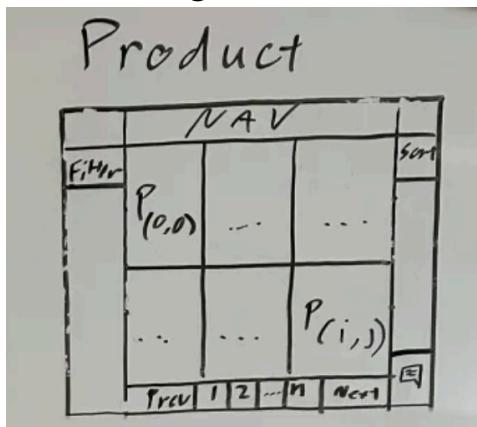
3.5 Admins will not be blockable like regular users

UI DIAGRAMS

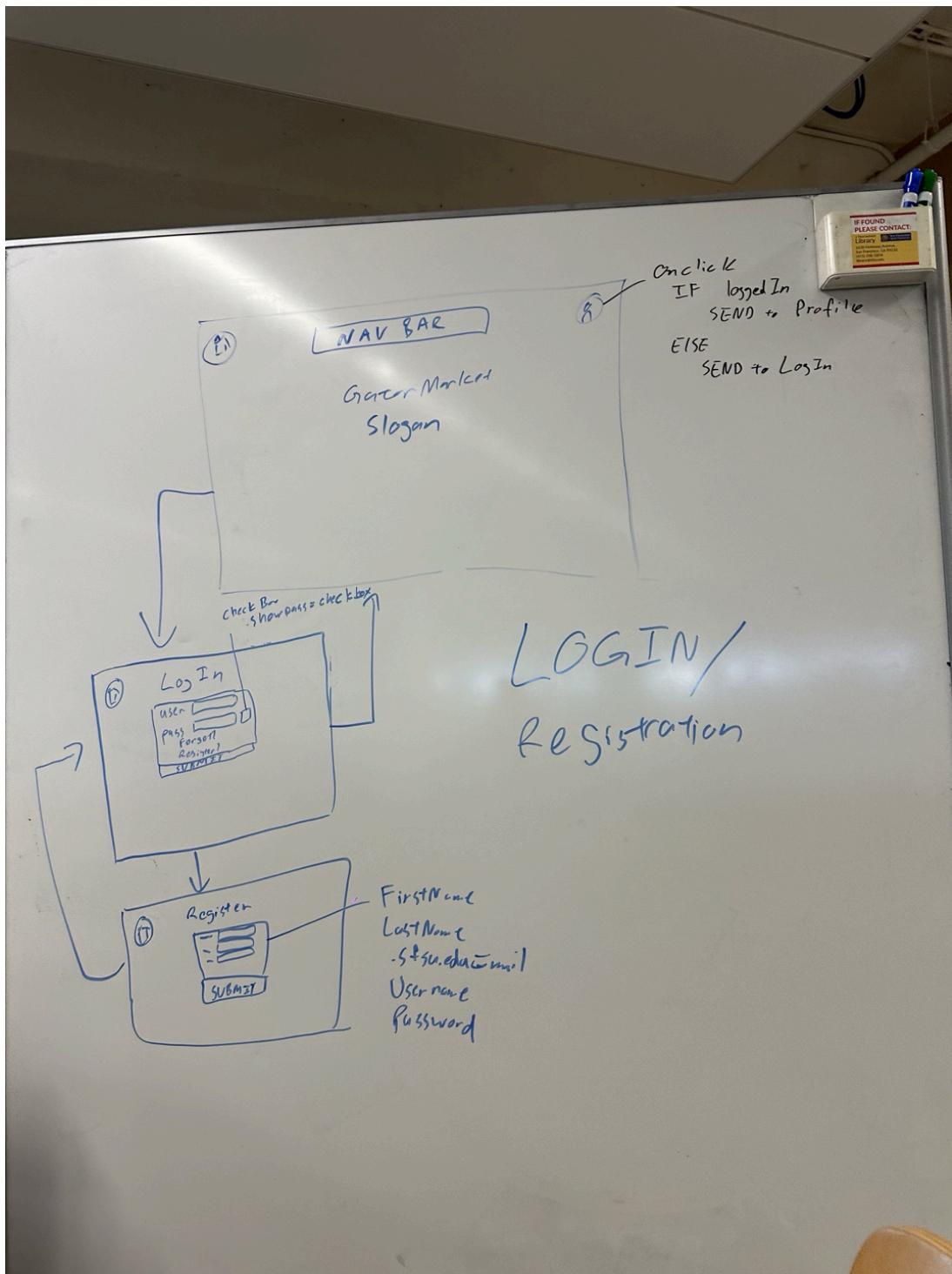
Home Page



Product Page



Create Account



Registration

① Register

Name
DOB
SPSU Email
Password abc
Confirm Password
User Name

DO NG!

② View password

③ Submit registration

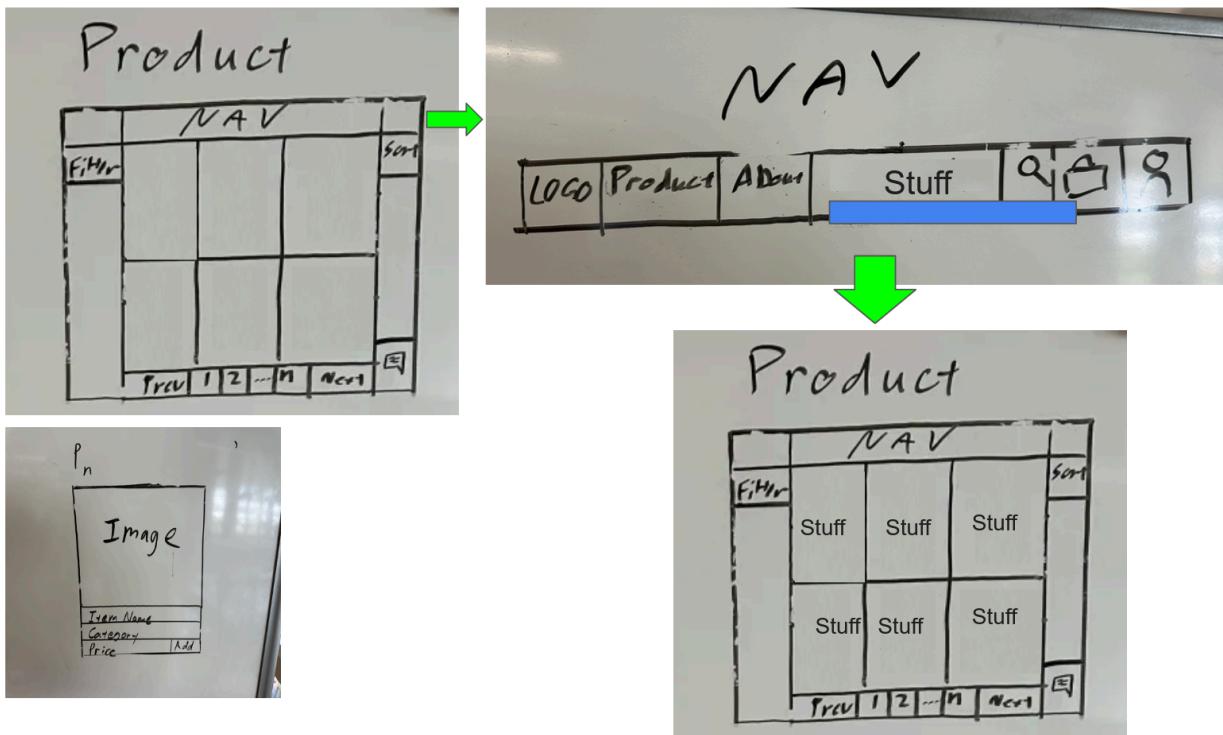
Login

① Log In

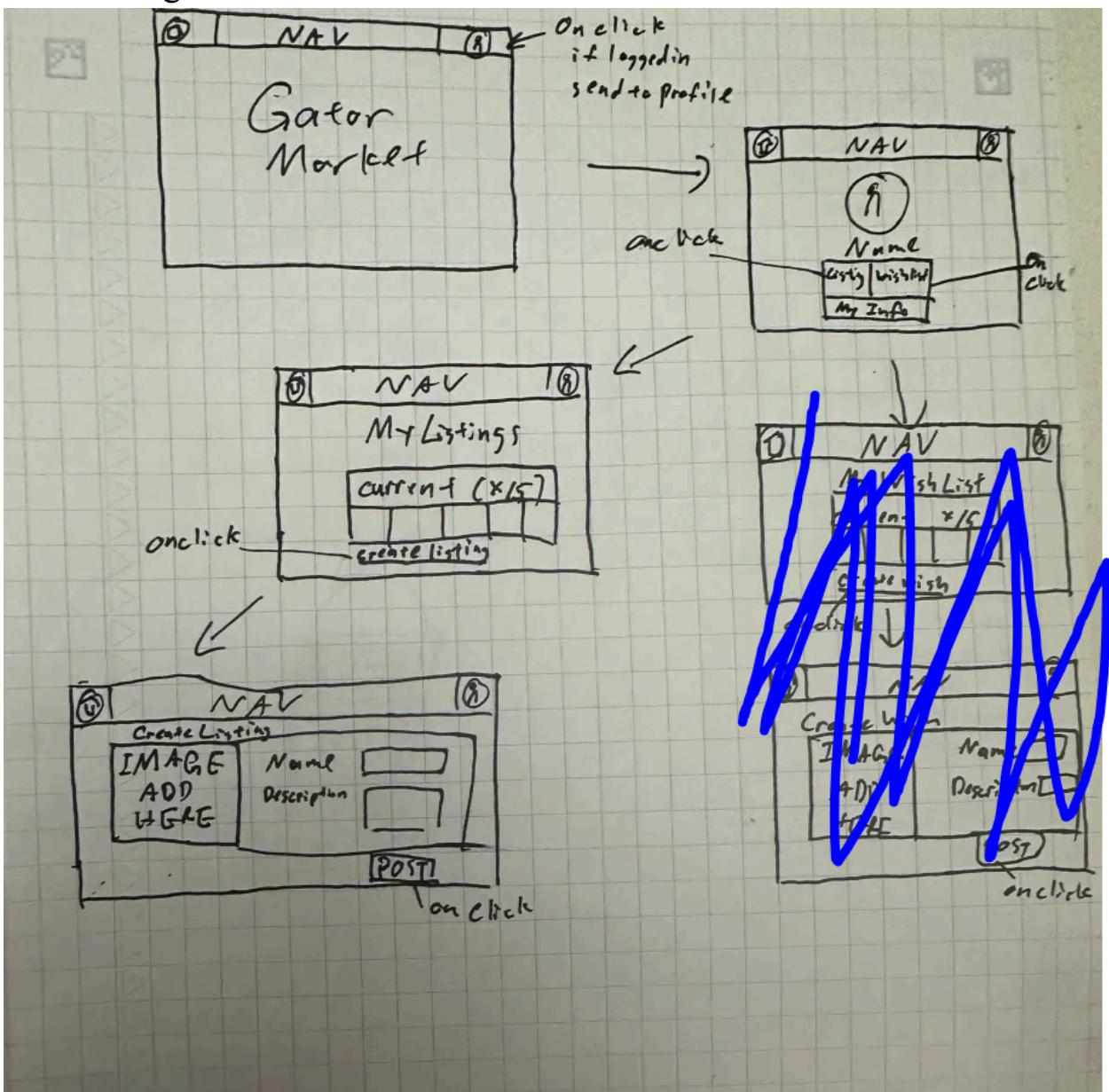
User
Pass
Forgot?

② Forget Password
Maybe not needed

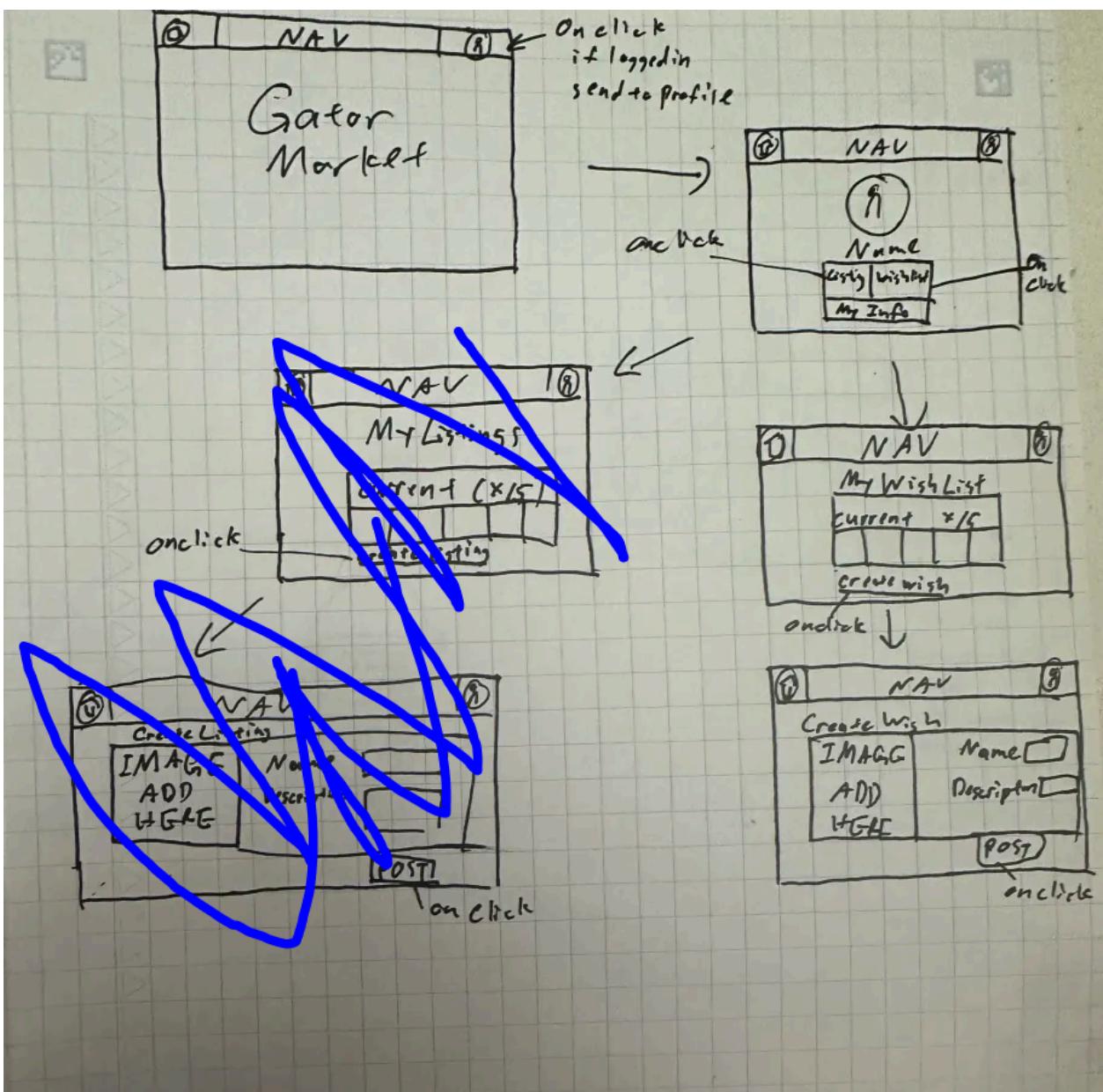
Search Item



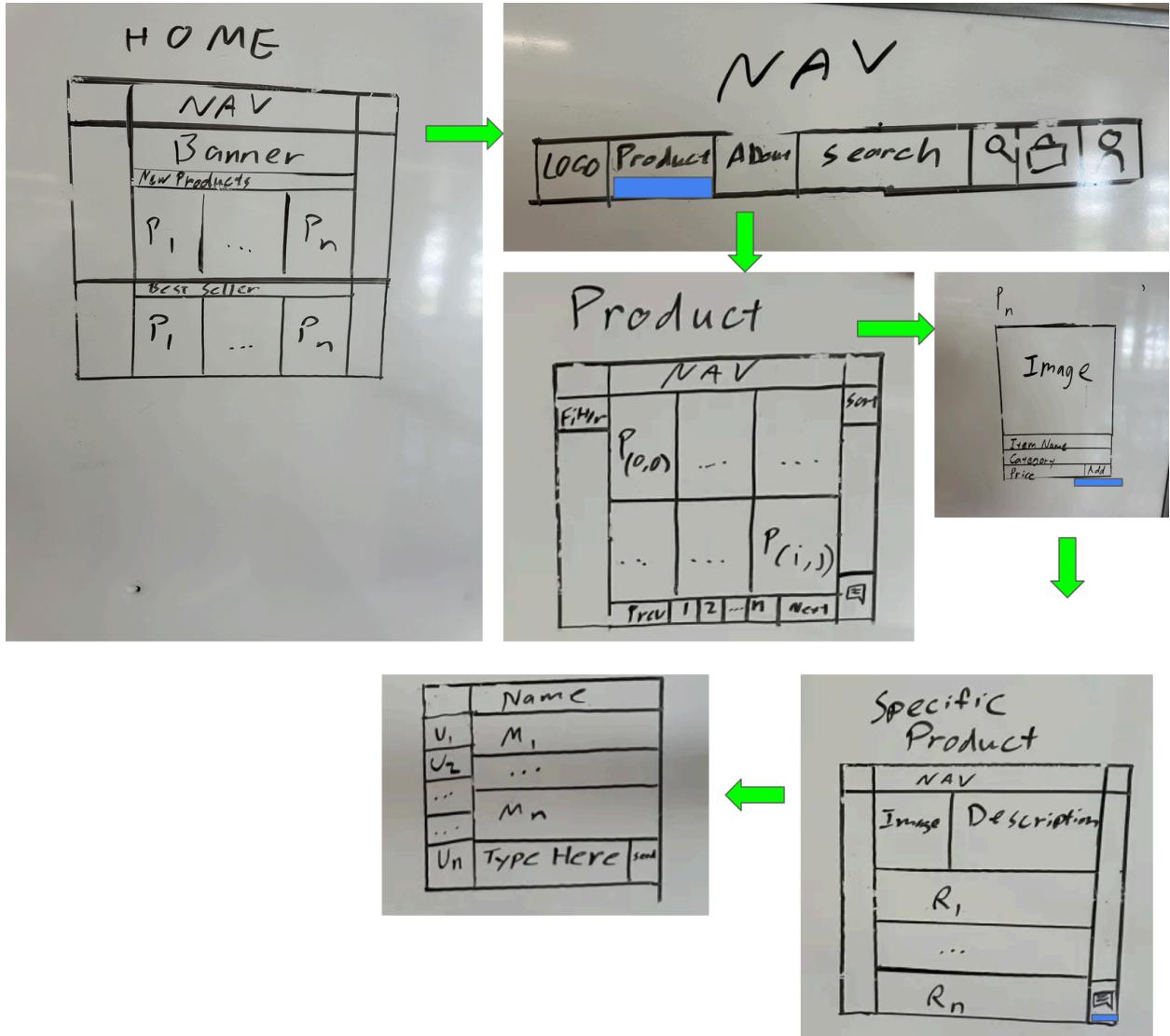
Post Listing



Make Wishlist



On Campus Meetup



HIGH LEVEL SYSTEM ARCHITECTURE AND TECHNOLOGIES

Main SW Components

Database - AWS RDS for MySQL ver 8.0

WWW Server: Apache 2.4

Node.js (20.x)

Deployment Cloud Server

AWS free tier - EC2 instance free for 12 months

Frontend Frameworks

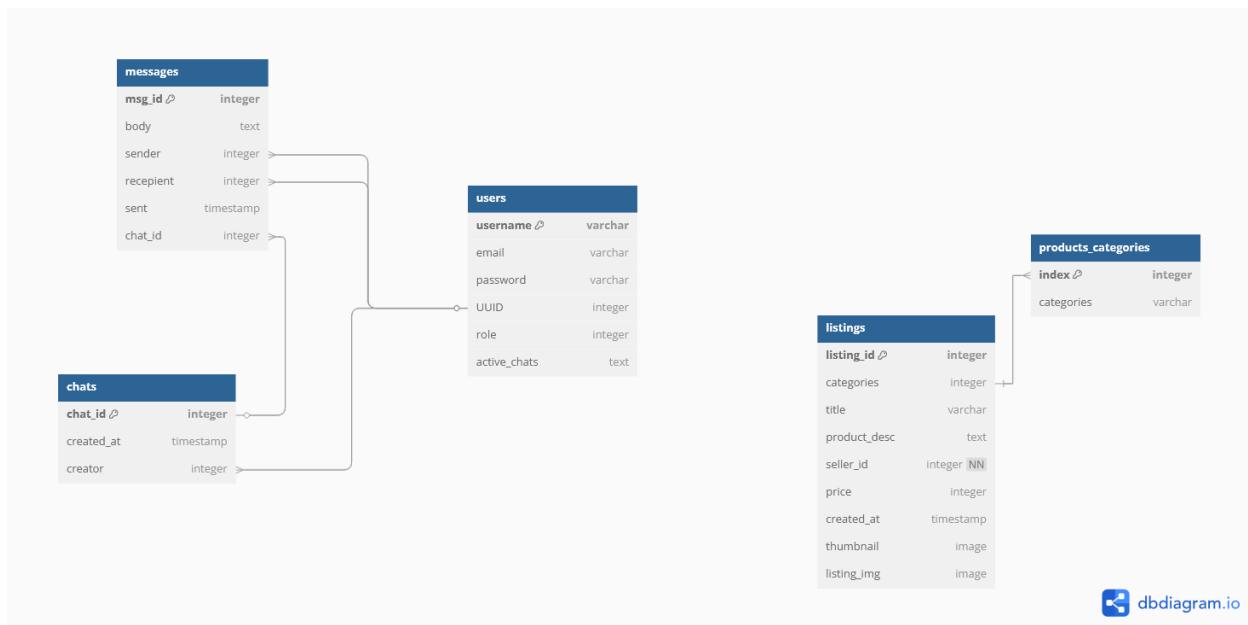
Express.js 4.18+

React (Next.js) 19.0.0

Browsers We Support

Chrome (Version 135)

Firefox (Version 136)



Media Storage: We will be using file systems to be storing images and other large media.

Search

Filter: We will be using SQL and selecting using product categories, price, and possibly product description to find stuff for the users

We're aiming to minimize the rating system on any non-trivial algorithms or processing.

KEY RISK

Skill Risks:

Some team members need to refresh their web development knowledge. Since it's been a while since they last worked with certain technologies, there may be a temporary dip in efficiency. To address this, we are revisiting key concepts through resources like YouTube tutorials. These help reinforce skills we previously learned in class and get us back up to speed.

Schedule Risks:

Balancing project work with other classes and part-time jobs may affect our productivity and ability to meet deadlines. School breaks can also disrupt team communication and workflow. To mitigate this, we identified key tasks and organized them based on priority. By completing important work early, we aim to maintain a flexible and manageable schedule.

Technical Risks:

Several team members have limited experience with backend technologies like AWS, Express.js, and SQL. This could slow down our progress on server-side development and integration with the application. To overcome this, team members are actively studying these technologies using YouTube tutorials and other free online resources. Continuous learning and collaboration will help us close knowledge gaps and keep development on track.

Teamwork Risks:

Communication gaps during spring break or busy school weeks could hinder team coordination. Limited availability during these times may lead to delays in progress or misunderstandings. To address this, the team scheduled additional meetings during the break. We also emphasized proactive communication through Discord and regular check-ins tied to project deadlines.

PROJECT MANAGEMENT

As the team lead for Team 04, I carefully evaluate the requirements for each milestone and anticipate upcoming tasks. Based on my assessment, I assign roles according to each team member's strengths and capabilities. To ensure equal contribution, I include everyone's name when distributing tasks. After assigning responsibilities, I clearly communicate each person's role and set deadlines well ahead of the milestone due date. This approach allows me to identify if someone is falling behind and gives them a few extra days to catch up before the final deadline.

For future tasks, I plan to continue refining this system by regularly checking in with each member, adjusting roles as needed based on progress and feedback, and breaking down larger assignments into smaller, more manageable parts to maintain steady progress and accountability.

GENAI USAGE

Tool:

ChatGPT 4o

Task where it was used:

Executive Summary

Useful rating High

How it was used and benefits:

ChatGPT was utilized to edit the Executive Summary that I had originally wrote down and organize it

Prompt Used:

“Organize and edit the paragraph to make it sound professional”

Tool:

Claude 3.7 Sonnet

High Level Functional Requirements

MEDIUM

How it was used and benefits:

We used gen ai to assist with organizing and formatting the high-level functional requirements into user categories.

Prompt Used:

“Can you organize these high-level functionalities into user based categories? Examples - Unregistered Users, Registered Users, Admin Users”

Tool: Claude 3.7 Sonnet

Section: High Level Architecture and DB Organization

Useful Rating: MEDIUM

How it was used and benefits:

We asked Claude for simple examples of database designs and architectures

Prompt Used:

“Give me an example of a simple DB Design structure for an ecommerce site.”

TEAM LEAD CHECKLIST

ON TRACK	So far, all team members are fully engaged and attending team sessions when required
ON TRACK	Team ready and able to use the chosen back and front-end frameworks and those who need to learn are working on learning and practicing
DONE/OK	Team reviewed class slides on requirements and use cases before drafting Milestone 2
DONE/OK	Team lead checked Milestone 2 document for quality, completeness, formatting and compliance with instructions before submission
DONE/OK	Team lead ensured that all team members read the final Milestone 2 document and agree/understand it before submission
ON TRACK	Team shared and discussed experience with GenAI tools among themselves

Milestone 3

REVIEW AND PLANS FOR FURTHER DEVELOPMENT

Summary of Feedback on UI

- Improve contrast on various UI elements.
- Shrink the green notification/info box to reduce wasted space.
- Revise hover text color for better readability.
- Add asterisks or indicators for required fields in forms.
- Listing page layout needs reorganization: add image and seller details; adjust message box size.
- Placement of contact information should be more intuitive.
- Fix alignment and structure of the search results page.

Summary of Feedback on Code and Architecture

- Backend image serving needs to be restructured.
- Use custom middleware instead of Express.static for image handling.

- Codebase organization should support cleaner modularization of backend image functions.

Summary of Feedback on GitHub Usage

- Mention any GitHub practices noted, e.g., need for better commit messages or branch organization.

Summary of Feedback on Database

- Add if any issues or feedback related to DB schema, image paths, or required fields were discussed.

Summary of Feedback on Teamwork and Risk Management

- Need for clearer task prioritization.
- Emphasis on timely completion of action items before next Monday to avoid delays.
- No major teamwork conflicts reported, but better coordination on backend/frontend integration was advised.

Architecture Review Status

- Team has reviewed architecture for adherence to MVC pattern.
- Identified need to modularize image serving logic.
- Minor coding style or documentation improvements may be needed.

P1 Feature List for Final Delivery

- Functional listing page with:
 - Display of image
 - Seller details
 - Messaging component
- Custom middleware to serve images dynamically
- Proper indication of required fields in all forms
- Revised UI components for search results and listings
- Branding update to replace trademarked SFSU logo with compliant logo

Other Comments or Issues

- All tasks to be completed and integrated before the next meeting.
- Action items delegated based on feature ownership.

Milestone 4

PRODUCT SUMMARY

Name of the product?

GatorMarket

Product Description

GatorMarket is a student-exclusive online marketplace built for the San Francisco State University (SFSU) community. It provides a safe and convenient platform where students can buy, sell, and trade items directly with one another. GatorMarket makes it easy to find or list what students need within a trusted network, from textbooks and electronics to furniture and school supplies. Unlike public platforms like Craigslist or Facebook Marketplace, GatorMarket verifies users through their @sfsu.edu email address, ensuring all members are part of the SFSU community. This adds a layer of trust and campus relevance that external platforms lack.

The application is designed with simplicity in mind. Students can post listings with images and descriptions, search by keyword or category, and contact sellers directly through a secure messaging system. GatorMarket reduces friction in student transactions by providing an all-in-one interface that is tailored to the university's needs, helping students save money, declutter, and connect locally.

Final Priority Features

These are the core functions the GatorMarket team is committed to fully delivering and testing for the final release:

- Users can register using a valid @sfsu.edu email address.
- Users must accept the terms of service during registration.
- Passwords are securely encrypted before storage in the database.
- Users can securely log in and log out.
- Logged-in users can create new listings with:
 - Title
 - Description
 - Image upload
 - Category

- Price
- Listings are displayed in a browsable, responsive grid layout.
- Users can filter listings by category or keyword.
- Users can search listings using a search bar (up to 40 alphanumeric characters).
- Users can view detailed information for individual listings.
- The website is accessible and mobile-friendly.

Note: While our posting and messaging features are designed and partially implemented, they are not currently functioning end-to-end due to unresolved issues with data persistence and live communication. We've identified where the breakdown occurs and plan to continue debugging after the final demo.

USABILITY TEST PLAN - SEARCH FUNCTION

Test Objectives

The search function is a major part of our site. Every user, buyer, and seller will use it. The first thing the user will see is the home page with a preview of our products. If a user is looking for something specific, they can use the search function to find it quickly. We have a categorical search and a string search. Categorical for a broad search and string for anything specific.

Categorical:

- Electronics
- Books
- Fashion
- Home and Kitchen
- Computers and Accessories

Test Background and Setup

System Setup: Any device with an internet connection and a modern browser.

Starting Point: <http://13.52.231.140:3001/> (Homepage of GatorMarket)

HW Tester Needs: Device with Chrome, Firefox, Safari, etc.

Intended Users: SFSU students and staff

Test Environment: Conducted on campus, in person, without prior training. Monitor starts off; time begins when monitor is turned on.

URL: <http://13.52.231.140:3001/>

Usability Task Description

Task: Search for a book using either categorical or string-based search.

Starting Condition: User begins on homepage.

Success Criteria: User finds the book using either method.

Benchmark: Task completed in under 15 seconds.

Effectiveness Evaluation

The search function is evaluated by how easily a user can locate a specific item (in this case, a book) using either method. Our internal testing showed that it takes under 5 seconds for a familiar user. To account for new users, we used Carnegie Mellon's Eberly Center guidance, allowing ~15 seconds for task completion. Observers will note whether the user prefers categorical or string search, and if any confusion arises during the process.

Efficiency Evaluation

Efficiency will be measured by how long it takes a user to complete the task from a cold start (monitor off). Once the monitor is turned on and the homepage loads, the timer begins. Time will be recorded once the correct result is found, and the method used (categorical or string) will be documented.

User Satisfaction Evaluation (Likert Scale Evaluation)

Based on a scale of: Strongly Agree, Agree, Neutral, Disagree, Strongly Disagree

- a) The search bar was easy to find
- b) The categorical search was useful
- c) The string search was useful

GENAI Usage

Tool Used: ChatGPT (GPT-4)

How We Used It:

ChatGPT was used to review and improve clarity in our task instructions, Likert scale questions, and evaluation criteria. It also helped us rephrase technical terms to better suit non-technical users and align our test logic with usability testing best practices.

Prompts Used:

- “Can you review this usability test description for clarity?”
- “Is this Likert scale well-formatted for a usability test?”
- “How can I better define efficiency and effectiveness in a test plan?”

Key Suggestions from GenAI:

- Recommended separating categorical and string examples more clearly
- Improved the phrasing of success criteria
- Suggested reordering instructions for better flow

Utility Ranking: HIGH

QUALITY ASSURANCE TEST PLAN AND TESTING

Test Objectives

Evaluating the functionality, accuracy, and reliability of the search feature, ensuring it returns correct and relevant results based on user input. The QA process also validates how the search behaves with both valid and invalid inputs.

Hardware and Software Setup

Hardware: Desktop, laptop, tablet, or smartphone with stable internet connection.

Software: Chrome v118 and Firefox v117 browsers

Test Environment: Postman for API testing

URL: <http://13.52.231.140:3001>

Feature to be Tested

The search functionality allows users to:

1. Search for listings by entering text in the search bar (matches against product titles and descriptions)
2. Filter listings by category using a dropdown menu
3. Combine text search with category filtering
4. View matching results with relevant product information

QA Test Plan

Test ID	Test Title	Test Description (What is being tested)	Test Input	Expected Correct Output	Chrome v118	Firefox v117
S-001	Basic Text Search	Test if the system returns correct results when searching by keyword	Navigate to homepage, enter “table” in search bar, click search button	Status 200, JSON response with listings containing “table” in title or description, count > 0	PASS	PASS
S-002	Category Filter	Test if the system filter results correctly by category	Navigate to homepage, select “Books” (category 1) from dropdown, click search button	Status 200, JSON response with listings only from Books category, count > 0	PASS	PASS
S-003	Combined Search	Test if the system correctly combines text search with category filtering	Navigate to homepage, select “Furniture” (category 3) from dropdown, enter “table” in search bar, click search button	Status 200, JSON response with listings containing “table” in title/description AND from Furniture category only	PASS	PASS

Test Execution Results

All tests were successfully executed on both Chrome v118 and Firefox v117 browsers. The search functionality performed as expected, with proper filtering based on text queries and category selection. Response times were consistently fast (under 500ms), and the user interface correctly displayed the filtered results.

No significant differences were observed between the two browsers. The search functionality handled all test cases appropriately and returned the expected results in both environments.

GenAI Usage

GenAI Tool Used: Claude 3.7 Sonnet

How the Tool Was Used: The tool was used to analyze our search functionality code and generate

a structured QA test plan that focuses on the key functionality. It helped identify critical test cases

and expected behaviors.

Key Examples and Prompts: We provided the codebase for our search functionality API and asked for a QA test plan with 3 essential test cases focusing on text search, category filtering, and

combined search. The prompt specified the need for browser compatibility testing. Utility

Ranking: HIGH - The GenAI tool efficiently analyzed our code and produced a comprehensive test plan aligned with software engineering best practices, saving significant time in test planning while ensuring thorough coverage of the functionality.

PEER CODE REVIEW

Code Review E-mail

Code Review

TG Trinity Liliana Godwin
To: Ⓛ Thanh Cong Duong
Cc: Ⓛ Clark Eullo Batungbakal; Ⓛ Caleb Onuonga; Ⓛ Daniel Lee

Hi Thanh,
Hope you're doing well.

Can you review and test our backend feature of searching. Any feedback regarding security, reliability, or any other concern would be appreciated.
You will find the feature with the route signature app.get("/api/search")

You can find the full code at the link below:
<https://github.com/CSC-648-SFSU/csc648-fa25-0104-team04/blob/milestone3/application/backend/app.js>

Any suggestions and criticism would be greatly appreciated.

Completed. No issues. I will check it out.

Reply Reply all Forward

Wed 5/7/2025 4:12 PM

TG Thanh Cong Duong
To: Ⓛ Trinity Liliana Godwin

Hi Trinity,

I also hope you are doing great!

I have reviewed the backend feature for searching, as requested, and I have already pushed my comments regarding security, reliability, and other concerns.

You can find the feedback in the same link you provided: <https://github.com/CSC-648-SFSU/csc648-fa25-0104-team04/blob/milestone3/application/backend/app.js>

Please feel free to reach out if you need any further clarification or have additional questions.

Best regards,
Thanh Duong.

...

Reply Forward

Fri 5/9/2025 6:48 AM

Code Comment Review

```
270 // API endpoint for search
271 app.get("/api/search", (req, res) => {
272   // Extract category and query parameters from the request
273   let { category, query } = req.query;
274
275   // Base SQL query
276   // Combining title and description for search
277   let sql = `
278     SELECT l.*, pc.categories as category_name
279     FROM listings l
280     JOIN products_categories pc ON l.categories = pc.index_id
281     WHERE 1=1
282   `;
283   /*
284     It is good to use parameterized queries to prevent SQL injection.
285     However, I recommend validating inputs like category and query to ensure
286     they are the expected types and avoid errors or inefficient queries.
287   */
288
289   const params = [];
290
291   // Add category filter if provided
292   if (category && category !== "default") {
293     // I would recommend validating that category is a valid integer before using it in the query.
294     sql += ` AND l.categories = ?`;
295     params.push(parseInt(category));
296   }
```

```
297 // Add text search filter if provided, searching title and description
298 if (query) {
299   // Searching for term in both title and description
300
301   /* I suggest trimming the query string and limiting its length
302    * to avoid overly broad searches that can slow down the database.
303   */
304   let sql: string
305   sql += ` AND (l.title LIKE ? OR l.product_desc LIKE ?`;
306   params.push(`%${query}%`); // Title Search
307   params.push(`%${query}%`); // Description Search
308 }
309
310 // You can consider to add pagination, which is LIMIT and OFFSET to avoid large result sets.
311
312 // Execute query with parameterized values
313 pool.query(sql, params, (err, results) => {
314   // I think it will be good to avoid showing raw database errors to the client.
315   // Instead, I log the errors on the server and send a simple error message to the client.
316   if (err) {
317     console.error("Error executing search query:", err);
318     return res.status(500).json({ error: "Database error" });
319   }
320
321   // Return results with count
322   return res.status(200).json({
323     count: results.length,
324     items: results,
325   });
326 });
327 });
328 */
329 The search functionality is working as intended and reliably returns accurate results.
330 The /api/search endpoint was tested using various combinations of title keywords and category filters.
331 The function successfully returned the expected results, accurately filtering listings based on
332 both the provided query, which matches the title and the category_name. The SQL query is parameterized,
333 minimizing the risk of SQL injection, and the response includes both the item count and relevant listing data.
334 */
335
```

Please review the code of the search function for the website of the marketplace. Any feedback regarding security, reliability, or any other concern backend feature of searching would be appreciated.

```
// API endpoint for search
app.get("/api/search", (req, res) => {
  let { category, query } = req.query;

  // Base SQL query
  // Combining title and description for search
  let sql =
    `SELECT l.*, pc.categories as category_name
     FROM listings l
    JOIN products_categories pc ON l.categories = pc.index_id
    WHERE 1=1
    ;

  const params = [];

  // Add category filter if provided
  if (category && category !== "default") {
    sql += ` AND l.categories = ?;
    params.push(parseInt(category));
  }

  // Add text search filter if provided, searching title and description
  if (query) {
    // Searching for term in both title and description
    sql += ` AND (l.title LIKE ? OR l.product_desc LIKE ?);
    params.push(` + encodeURIComponent(query) + `); // Title Search
    params.push(` + encodeURIComponent(query) + `); // Description Search
  }

  // Execute query with parameterized values
  pool.query(sql, params, (err, results) => {
    if (err) {
      console.error("Error executing search query:", err);
      return res.status(500).json({ error: "Database error" });
    }

    // Return results with count
    return res.status(200).json({
      count: results.length,
      items: results,
    });
  });
});
```

Output of GenAI for code review

Your search endpoint is generally functional and uses parameterized queries (good for security), but there are several **key improvements** and **concerns** around **security, reliability, scalability, and user experience** you should address:

What's Good:

- **Parameterized Queries:** Prevents SQL injection — this is critical.
- **Flexible Search:** Handles optional filters like `category` and `query`.
- **Clear Response Format:** Returns both `count` and `items`, which is useful for frontend rendering.

⚠️ Security Concerns:

1. No Input Validation/Sanitization:

- You're trusting `req.query` directly. While SQL injection is mitigated, malformed or unexpected inputs may cause issues.
- **Fix:** Validate types and add default values:

```
js                                     ⚡ Copy ⚡ Edit

const category = Number.isInteger(parseInt(req.query.category)) ? parseInt(req.query.category) : 0;
const query = typeof req.query.query === 'string' ? req.query.query.trim() : '';
```

2. No Rate Limiting / Abuse Protection:

- Without protection, this endpoint is vulnerable to abuse via brute-force or scraping.
- **Fix:** Use middleware like `express-rate-limit` or add caching with `Redis` for frequent queries.

3. Error Leaking:

- Avoid logging raw SQL errors in production; they might leak internal database structure.
- **Fix:** Log detailed errors server-side but return only generic messages:

```
js                                     ⚡ Copy ⚡ Edit

console.error(err); // Keep detailed logs
res.status(500).json({ error: "Internal server error" });
```

🛠 Reliability / Functionality Enhancements:

1. Pagination is Missing:

- Returning all matching items without pagination can cause performance issues.
- Fix: Add `LIMIT` and `OFFSET` support:

```
js

const page = parseInt(req.query.page) || 1;
const limit = parseInt(req.query.limit) || 10;
const offset = (page - 1) * limit;

sql += ` LIMIT ? OFFSET ?`;
params.push(limit, offset);
```

2. Order Results for Better UX:

- Right now, results are unordered.
- Fix: Add something like `ORDER BY l.created_at DESC` or relevant sort logic.

3. Support for Full-Text Search (optional):

- MySQL's `LIKE` becomes slow on large datasets.
- Fix (optional): Use `FULLTEXT` indexes and `MATCH ... AGAINST` for better performance and relevance.

Minor Code Quality Suggestions:

- **Avoid SQL string concatenation** (e.g., `sql += ...`) where possible. Consider building an array and joining.
- **Refactor for readability:** Break the query building into clear, named helper functions if it grows larger.

Summary of Recommended Fixes:

Category	Issue	Fix Suggestion
Security	No validation	Type-check and sanitize inputs
Security	No rate limiting	Use <code>express-rate-limit</code>
Security	Raw error logging	Return generic messages to client
Reliability	No pagination	Add <code>LIMIT</code> and <code>OFFSET</code> support
Performance	LIKE queries may scale poorly	Consider <code>FULLTEXT</code> or external search engine
UX	Results not ordered	Add <code>ORDER BY</code> clause

Would you like a refactored version of the endpoint with all improvements applied?



GenAI Usage

What genAI tool and version you used: The ChatGPT-4 model by OpenAI

Explain briefly how you used the tool and what benefit it offered:

The way I used the tool was that I told the AI about what I was looking for. I provided the code for the search function and asked for feedback. The tool offered valuable insights by reviewing the code for potential issues. It also suggested improvements to help me ensure the code is secure, efficient, and scalable.

Provide brief examples of key concepts and prompts:

- “Please review the code of the search function for the website of the marketplace. Any feedback regarding security, reliability, or any other concerns regarding the backend feature of searching would be appreciated.”
- “Can you help me ensure that my backend API doesn't expose raw database errors to the client in production?”
- “Can you review the way I handle user inputs in my backend search function, and suggest improvements for validating and sanitizing inputs?”
- “How can I optimize the performance of this search function if my database becomes larger and larger?”

Utility Ranking: Medium

BEST PRACTICES FOR SECURITY

This section outlines the major assets protected within GatorMarket, identifies potential security threats to each, and describes our strategies to mitigate those threats. We also confirm that best practices such as password encryption and input validation are implemented across the application.

Security Risk Table

Asset to be Protected	Types of Expected Attacks	Consequence of Security Breach	Strategy to Mitigate/Protect
User Passwords	Brute force, credential stuffing, SQL injection	Unauthorized access to user accounts	Passwords hashed using bcrypt before storage
User Email Addresses	Data scraping, phishing	Exposure of personal information, targeted attacks	Restricted email visibility; validation on input
Product Listings	Spam submissions, script injection (XSS)	Corruption of UI, loss of trust	Input validation and sanitization
User Sessions	Session hijacking, token theft	Impersonation of users	Session tokens with HTTP-only and secure flags
Messages	Spam or malicious links	User harassment, phishing	Only authenticated users can message; input validation

Search Input	Injection attacks, abuse	System crash, UI bugs	Limit input to 40 alphanumeric characters
Database	SQL injection	Data theft or loss	Use of parameterized queries and ORM protection

Confirmation of Security Practices

- Password encryption: All passwords are securely hashed using bcrypt before being stored in the database.
- Input Validation Enforced:
 - Search input: Limited to 40 alphanumeric characters
 - Registration email: Must end with 'sfsu.edu'
 - Terms of Services: The checkbox must be accepted before registration can be completed

ADHERENCE TO ORIGINAL NON-FUNCTIONAL REQUIREMENTS

Non-functional Requirement	Status	Notes (if ISSUE)
Application shall be developed, tested and deployed using tools and cloud servers approved by Class CTO and as-agreed in M0	DONE	
Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest version of two major browsers	DONE	
All or selected application functions shall render well on mobile devices (no native app to be developed)	DONE	
Posting of sales information and messaging to sellers shall be limited only to SFSU students	DONE	

Critical data shall be stored in the database on the team's deployment server	DONE	
No more than 50 concurrent users shall be accessing the application at any time	ON TRACK	System supports 10+ users; more concurrency testing pending
Privacy of users shall be protected	DONE	Data is encrypted and access-controlled
The language used shall be English (no localization needed)	DONE	
Application shall be very easy to use and intuitive	ON TRACK	Usability testing in progress
Application shall follow established architecture patterns	DONE	
Application code and its repository shall be easy to inspect and maintain	ON TRACK	Code is structured, commented, and in GitHub
Google Analytics shall be used	ISSUE	Not implemented yet
No e-mail client or chat services shall be allowed. Messaging is limited to one-way in-site messages	ON TRACK	Only seller messaging is implemented
Pay functionality, if any, shall not be implemented nor simulated in UI	DONE	
Site security: basic best practices shall be applied	DONE	See Section 5 for details
Media formats shall be standard as used in the market today	DONE	Image upload supports JPG, PNG
Modern SE processes and tools shall be used	DONE	Using GitHub, issue tracking, and VSCode
Application UI shall display: 'SFSU Software Engineering Project CSC 648-848, Spring 2025. For Demonstration Only' at the top of each page	ON TRACK	Banner added to some pages; rollout to all pages in progress

PRODUCT SCREENSHOTS

Home page

The screenshot shows the homepage of the San Francisco State University Market. At the top, there is a dark header bar with a small circular profile picture, a search bar, and some navigation icons. Below the header is a large green banner with the text "San Francisco State University Market" and a subtitle "Your trusted online shopping destination for quality products and excellent service." Underneath the banner, there is a section titled "All Products" featuring four product cards:

- Calculus Textbook**: An image of the book "CALCULUS" by James Stewart, showing a blue cover with mathematical diagrams.
- Java Programming Book**: An image of the book "JAVA Programming" showing a dark cover with a network of glowing nodes.
- World History Encyclopedia**: An image of the book "WORLD HISTORY ENCYCLOPEDIA" featuring illustrations of historical figures and artifacts.
- Organic Chemistry Lab Manual**: An image of the book "LABORATORY MANUAL FOR ORGANIC CHEMISTRY" showing a green cover with molecular structures.

A small green circular icon with a white phone symbol is located in the bottom right corner of the page.

All categories

SFSU Software Engineering Project CSC 648-848, Spring 2025. For Demonstration Only

The screenshot shows a dark-themed web application interface. At the top, there's a navigation bar with a user icon, a search input field, and a green search button. Below the header, the text "Search results for 'All'" is displayed. The main content area contains four product cards in a grid:

- Calculus Textbook** by James Stewart, Eighth Edition. Price: \$100.00.
- Java Programming Book** by Java, Tenth Edition. Price: \$60.00.
- World History Encyclopedia**. Price: \$85.00.
- Laboratory Manual for Organic Chemistry** by Kenneth F. Cerny, Marietta H. Schwartz, Christopher E. Katz, Fourth Edition. Price: \$70.00.

Below these, there are three partially visible items: "THE SCRIBNER ANTHOLOGY OF CONTEMPORARY SHORT FICTION", a handheld electronic device, and a black curved object.

Product Page

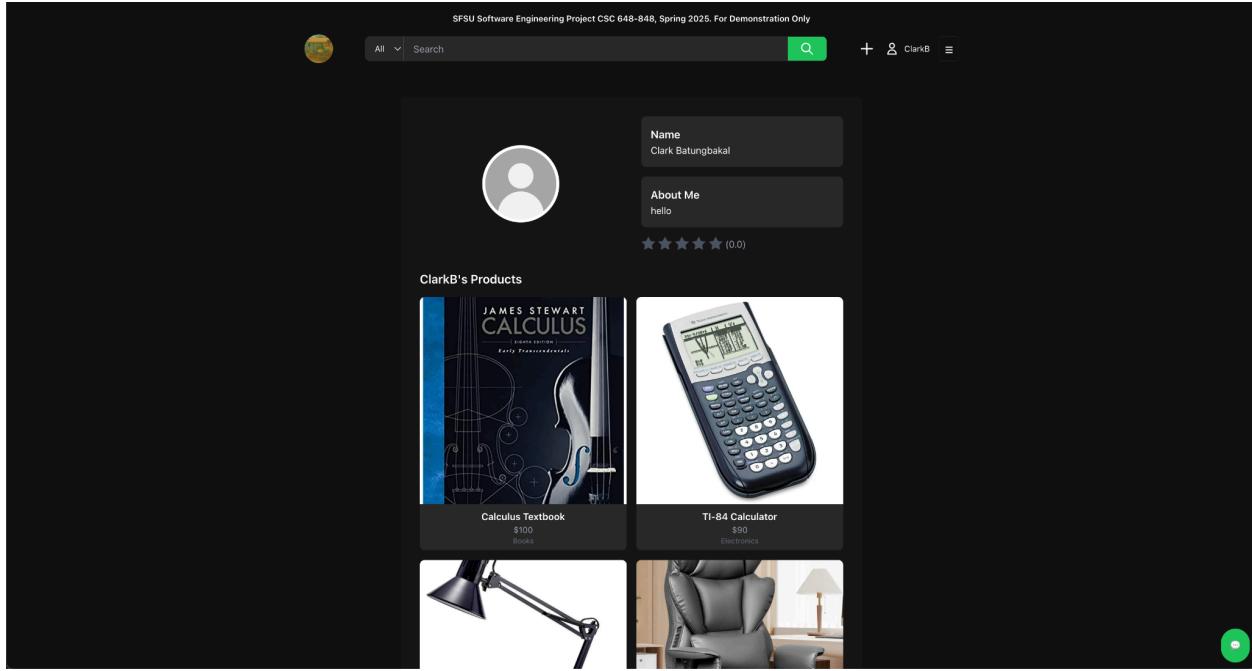
SFSU Software Engineering Project CSC 648-848, Spring 2025. For Demonstration Only

The screenshot shows a detailed product page for the "Calculus Textbook". The page includes the following information:

- Product Title:** Calculus Textbook
- Image:** Cover of the book "JAMES STEWART CALCULUS, Eighth Edition".
- Price:** \$100
- Category:** Books
- Description:** University level calculus textbook
- Contact Seller:** A green button.
- Seller Information:** Seller is Clark Batungbakal, with a profile picture and a 0-star rating.

At the bottom of the page, there are links for "Back to top" and copyright information: "© 2025, natormarket.com, CSC 648-848 Team 04".

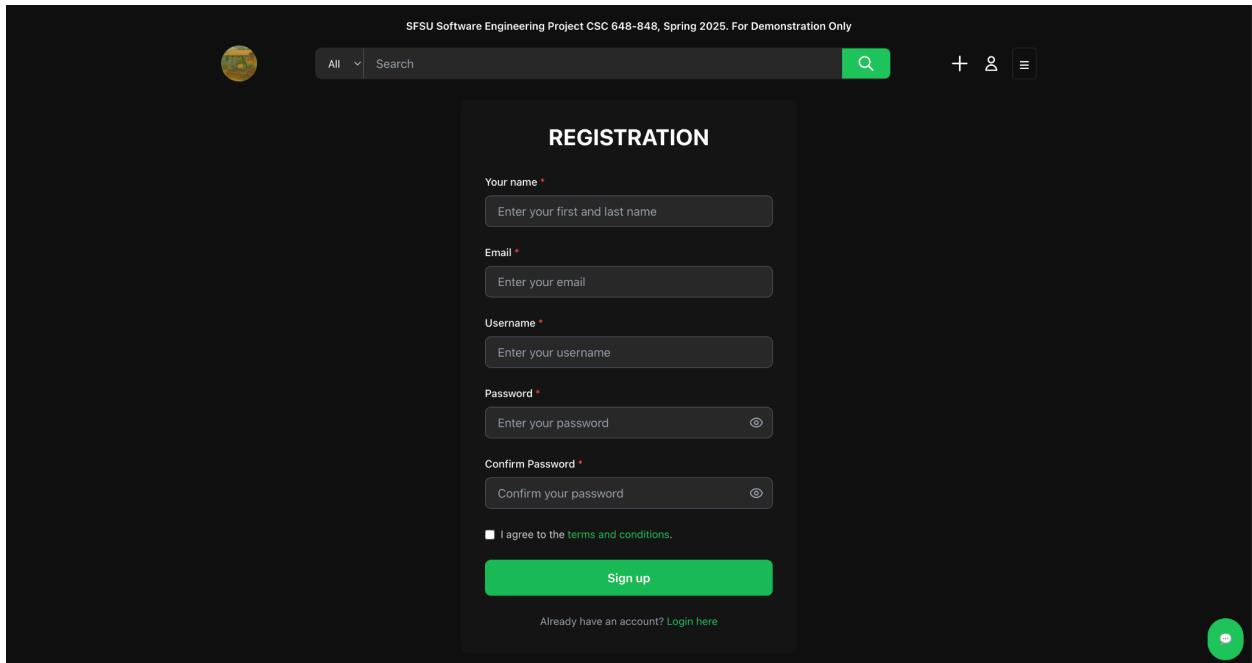
Seller Detail Page



The screenshot shows a dark-themed seller detail page. At the top, there's a navigation bar with a search bar and a user profile icon. Below the header, the seller's profile information is displayed: Name (Clark Batungbakal), About Me (hello), and a rating of 0.0 stars. A large circular placeholder image for the seller's profile picture is shown. Below this, a section titled "ClarkB's Products" displays four items:

- Calculus Textbook** by James Stewart, \$100, Books. An image of the book cover is shown.
- TI-84 Calculator**, \$90, Electronics. An image of the calculator is shown.
- Lamp**. An image of a desk lamp is shown.
- Office Chair**. An image of an office chair is shown.

Registration Page

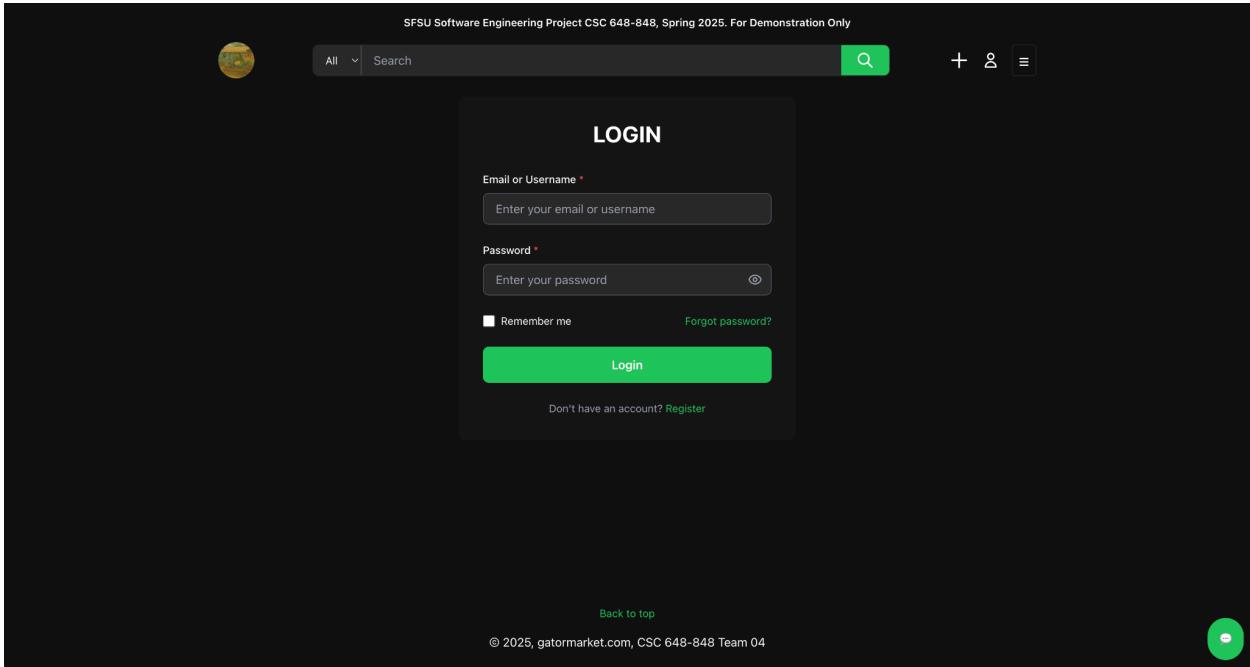


The screenshot shows a registration form on a dark-themed page. The title "REGISTRATION" is centered at the top. The form consists of several input fields with validation requirements indicated by asterisks (*):

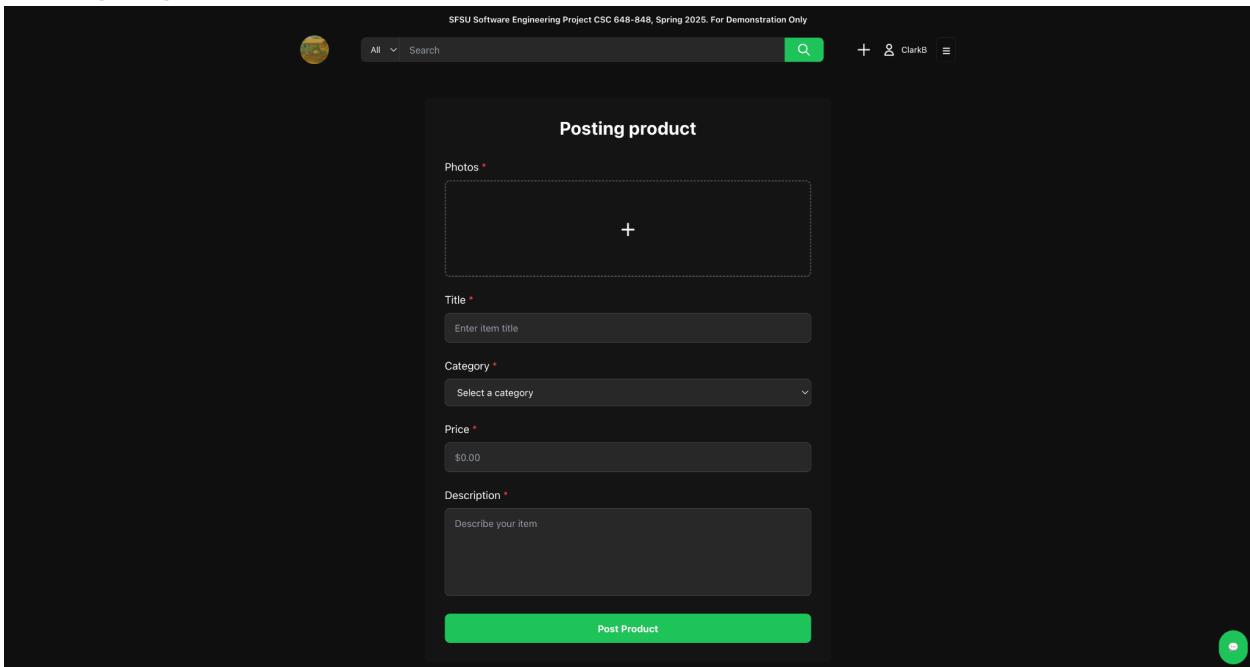
- Your name ***: A text input field with placeholder text "Enter your first and last name".
- Email ***: A text input field with placeholder text "Enter your email".
- Username ***: A text input field with placeholder text "Enter your username".
- Password ***: A password input field with placeholder text "Enter your password" and a visibility toggle icon (eye).
- Confirm Password ***: A password input field with placeholder text "Confirm your password" and a visibility toggle icon (eye).

Below the input fields is a checkbox labeled "I agree to the [terms and conditions](#)". At the bottom of the form is a green "Sign up" button. A small note at the very bottom says "Already have an account? [Login here](#)".

Login Page



Posting Page



Profile Page

SFSU Software Engineering Project CSC 648-848, Spring 2025. For Demonstration Only

All Search + ClarkB ⌂

Profile

Clark Batungbakal
@ClarkB

Email	Full Name
cbatungbakal@sfsu.edu	Clark Batungbakal
Username	Account Status
ClarkB	active

Contact Us Page

SFSU Software Engineering Project CSC 648-848, Spring 2025. For Demonstration Only

All Search + ClarkB ⌂

Contact Us

Name
Your name

Email
your.email@example.com

Subject
How can we help?

Message
Your message...

Send Message

Back to top

© 2025, gatormarket.com, CSC 648-848 Team 04

About us page

SFSU Software Engineering Project CSC 648-848, Spring 2025. For Demonstration Only

All Search

+ Clark8

About Us

"We are Team 4"

Our Team

Clark Batungbakal
Team Leader

I am a Marine Corps veteran and a Computer Science student at San Francisco State University, set to graduate in December 2025. My interests lie in cloud application development. I have a deep appreciation for Korean history and culture, which I have explored through coursework and personal experiences. I also enjoy traveling, especially to South Korea, where I have gained a new perspective on food, language, and traditions. Feel free to connect with me—I am always open to new opportunities and collaborations! Passionate about creating innovative solutions for our customers.

Thanh Cong Duong
Front-End Leader

I am a Computer Science student at San Francisco State University. I will graduate on Spring 2025. I am interested in traveling around the world to know more about many different cultures, especially food.

Caleb Onuonga
Back-End Leader

Hi everyone my name is Caleb Onuonga. I am team 4's backend lead. In my free time I enjoy reading, playing video games, watching anime, hiking, playing basketball and hanging with my friends.

Trinity Godwin
GitHub Master

Hello my name is Trinity Godwin. I am acting as Github master for team 04. I am excited to be working with my teammates and happy to be creating such a cool project.

Daniel Lee
Front-End Member

Ello, I am a double major in computer science and physiology. I like to try new things and the way I live is to go where the wind takes me. Currently I am looking into pharmacy school

DATABASE ORGANIZATION

```

mysql> SHOW TABLES;
+-----+
| Tables_in_team4db |
+-----+
| listings          |
| messaging         |
| products_categories |
| users             |
+-----+
4 rows in set (0.010 sec)

```

mysql>

```

mysql> SELECT * FROM listings;
+-----+
| listing_id | categories | title           | product_desc          | seller_id | price | created_at      | thumbnail          | listing_img        |
+-----+
| 1 | 1 | Calculus Textbook | University level calculus textbook | 1 | 100 | 2025-04-04 01:49:30 | /uploads/image-1747826882408-557530012.png | /uploads/image-1747826882408-557530012.png |
| 2 | 2 | TI-84 Calculator | Graphing calculator for math classes | 1 | 98 | 2025-04-04 01:49:30 | /uploads/image-174787986918-983118613.png | /uploads/image-174787986918-983118613.png |
| 3 | 3 | Desk Lamp | LED desk lamp with adjustable brightness | 1 | 88 | 2025-04-04 01:49:30 | /uploads/image-174787915866-253758386.png | /uploads/image-174787915866-253758386.png |
| 4 | 4 | Office Chair | Ergonomic office chair for home or dorm | 1 | 70 | 2025-04-04 01:49:30 | /uploads/image-17478791577-864597780.png | /uploads/image-17478791577-864597780.png |
| 5 | 1 | Java Programming Book | Learn Java programming from scratch | 5 | 68 | 2025-04-04 01:49:30 | /uploads/image-174787915352-919737466.png | /uploads/image-174787915352-919737466.png |
| 6 | 1 | World History Encyclopedia | Comprehensive history reference book | 1 | 85 | 2025-04-11 02:16:36 | /uploads/image-174787915509-853803940.png | /uploads/image-174787915509-853803940.png |
| 7 | 1 | Organic Chemistry Lab Manual | University-level chemistry lab guide | 1 | 70 | 2025-04-11 02:16:36 | /uploads/image-174787915688-272664816.png | /uploads/image-174787915688-272664816.png |
| 8 | 1 | Modern Fiction Anthology | Collection of contemporary short stories | 1 | 55 | 2025-04-11 02:16:36 | /uploads/image-174787952945-788539831.png | /uploads/image-174787952945-788539831.png |
| 9 | 2 | Wireless Bluetooth Headphones | Noise-cancelling over-ear headphones | 1 | 128 | 2025-04-11 02:16:36 | /uploads/image-174787953079-327400103.png | /uploads/image-174787953079-327400103.png |
| 10 | 2 | Portable External Hard Drive | 2TB storage capacity | 1 | 98 | 2025-04-11 02:16:36 | /uploads/image-174787953188-518958998.png | /uploads/image-174787953188-518958998.png |
| 11 | 2 | Smart Home Hub | Voice-controlled home automation device | 1 | 75 | 2025-04-11 02:16:36 | /uploads/image-174787953281-899723789.png | /uploads/image-174787953281-899723789.png |
| 12 | 2 | Digital Camera | 24MP point-and-shoot camera | 1 | 150 | 2025-04-11 02:16:36 | /uploads/image-174787953424-593930166.png | /uploads/image-174787953424-593930166.png |
| 13 | 3 | Bookshelf | 5-tier wooden bookshelf for home storage | 1 | 128 | 2025-04-11 02:16:36 | /uploads/image-174787953561-111182591.png | /uploads/image-174787953561-111182591.png |
| 14 | 3 | Coffee Table | Modern glass and wood coffee table | 1 | 148 | 2025-04-11 02:16:36 | /uploads/image-1747879553745-73215635.png | /uploads/image-1747879553745-73215635.png |
| 15 | 3 | Bedside Table | Compact nightstand with drawer | 1 | 70 | 2025-04-11 02:16:36 | /uploads/image-1747879553989-38638858.png | /uploads/image-1747879553989-38638858.png |
| 16 | 3 | Computer Desk | Ergonomic workspace with cable management | 1 | 160 | 2025-04-11 02:16:36 | /uploads/image-1747879554096-348198501.png | /uploads/image-1747879554096-348198501.png |
| 17 | 4 | Blender Set | Professional-grade kitchen blender | 1 | 85 | 2025-04-11 02:16:36 | /uploads/image-1747879554241-622245868.png | /uploads/image-1747879554241-622245868.png |
| 18 | 4 | Cookware Set | Non-stick pots and pans collection | 1 | 118 | 2025-04-11 02:16:36 | /uploads/image-1747879554372-291282941.png | /uploads/image-1747879554372-291282941.png |
| 19 | 4 | Knife Block | Chef's knife set with wooden block | 1 | 95 | 2025-04-11 02:16:36 | /uploads/image-1747879554532-969286233.png | /uploads/image-1747879554532-969286233.png |
| 20 | 4 | Coffee Maker | Programmable drip coffee machine | 1 | 65 | 2025-04-11 02:16:36 | /uploads/image-1747879554727-283783134.png | /uploads/image-1747879554727-283783134.png |
| 21 | 5 | Mechanical Keyboard | RGB gaming keyboard with tactile switches | 1 | 95 | 2025-04-11 02:16:36 | /uploads/image-1747879554867-826373787.png | /uploads/image-1747879554867-826373787.png |
| 22 | 5 | Wireless Mouse | Ergonomic rechargeable computer mouse | 1 | 48 | 2025-04-11 02:16:36 | /uploads/image-1747879554998-931244942.png | /uploads/image-1747879554998-931244942.png |
| 23 | 5 | Laptop Cooling Pad | USB-powered cooling stand | 1 | 38 | 2025-04-11 02:16:36 | /uploads/image-1747879555106-591946582.png | /uploads/image-1747879555106-591946582.png |
| 24 | 5 | 24-inch Monitor | Full HD display with adjustable stand | 1 | 180 | 2025-04-11 02:16:36 | /uploads/image-1747879555266-293308160.png | /uploads/image-1747879555266-293308160.png |
| 25 | 5 | USB-C Hub | Multi-port adapter for laptops | 1 | 55 | 2025-04-11 02:16:36 | /uploads/image-1747879555401-171267665.png | /uploads/image-1747879555401-171267665.png |
| 26 | 6 | Leather Wallet | Genuine leather bifold wallet | 1 | 45 | 2025-04-11 02:16:36 | /uploads/image-1747879555511-532991223.png | /uploads/image-1747879555511-532991223.png |
| 27 | 6 | Wristwatch | Waterproof analog/digital watch | 1 | 128 | 2025-04-11 02:16:36 | /uploads/image-1747879555673-390566803.png | /uploads/image-1747879555673-390566803.png |
| 28 | 6 | Sunglasses | UV-protective polarized sunglasses | 1 | 75 | 2025-04-11 02:16:36 | /uploads/image-1747879555868-48397198.png | /uploads/image-1747879555868-48397198.png |
| 29 | 6 | Backpack | Water-resistant laptop backpack | 1 | 65 | 2025-04-11 02:16:36 | /uploads/image-1747879555978-664209156.png | /uploads/image-1747879555978-664209156.png |
| 30 | 6 | Running Shoes | Cushioned athletic footwear | 1 | 98 | 2025-04-11 02:16:36 | /uploads/image-1747879556112-54641789.png | /uploads/image-1747879556112-54641789.png |
+-----+
30 rows in set (0.011 sec)

mysql>

```

```
[mysql]> SELECT * FROM messaging;
+-----+-----+-----+-----+
| sender | receiver | sender_text | receiver_text |
+-----+-----+-----+
|     1   |      2 | Hello Receiver|Hello Receiver|Hello Receiver|Hello Receiver | Hello Sender|Hello Sender|Hello Sender|Hello Sender|
|     1   |      3 | Hello Receiver! |           |           |           | Hello Sender! helloworld|
|     6   |      1 | Hello Receiver! |           |           |           | Hello Sender!
|     7   |      1 | Hello Receiver! |           |           |           | Hello Sender!
+-----+-----+-----+
4 rows in set (0.008 sec)

mysql> █
```

```
[mysql]> SELECT * FROM products_categories;
+-----+-----+
| index_id | categories |
+-----+-----+
|       1   | Books      |
|       2   | Electronics |
|       3   | Furniture   |
|       4   | Home & Kitchen |
|       5   | Computers & Accessories |
|       6   | Fashion    |
+-----+-----+
6 rows in set (0.009 sec)

mysql> █
```

```

mysql> SELECT * FROM users;
+----+-----+-----+-----+-----+-----+-----+-----+
| uid | username | password | role | email | full_name | about_me | rating |
+----+-----+-----+-----+-----+-----+-----+-----+
| 1 | ClarkB | $2b$10$kyw44w9fH7DDXsf2cyIWez81CgqkJfoagydMGU2d6MnWp6a71Q/S | 0 | cbatungbakal@sfsu.edu | Clark Batungbakal | hello | 2 |
| 2 | clark123 | $2b$10$irzZec/M.AwhUjOU8M0cpuOBpw8/Ztgt21FxuK7yyqlpb816J6 | 0 | clarke@mail.sfsu.edu | Clark Batungbakal | NULL | NULL |
| 3 | calebttest123 | $2b$10$5b/7gfDsghNrzTBGdiBn2e1lPhqNX5cpW7hIIL0nceF0R1ZXKkbn2 | 0 | caleb123@user.com | caleb | NULL | NULL |
| 4 | lebron | $2b$10$mUD8wSw1K7GwybfH86/PhfURna8XR1HMLkiuOpOHM14tf54EQ92 | 0 | lebron@gmail.com | LeBron James | NULL | NULL |
| 5 | Daniel | $2b$10$hf.OsNdf4KebIdryiLi3eJYfdmFPV13Y1T4K0dwmuahlhVzeK6 | 0 | tduong7@sfsu.edu | Thanh Duong | hello | 4 |
| 6 | Alligator | $2b$10$sc0twIcd8K1MyP9vOxpYZ.hyt9u9R80s3UostPwT5Hftsi.z.tQMm | 0 | Alligator123@sfsu.edu | Alli Gator | NULL | NULL |
| 7 | JM | $2b$10$QB7ze7NCBmkP2m2NVFLKs.UmZY5dvpMP35K2XQ31Bv6nMiyaB1ePK | 0 | jmama@sfsu.edu | Joe Mama | NULL | NULL |
| 8 | MasterChief | $2b$10$Ehela9Wh3PDYIPr4R455LDf072BSx7RIu8KWX2USy3kgG2Ho.31I.Ai | 0 | mchief@sfsu.edu | Master Chief | NULL | NULL |
| 9 | 르박 | $2b$10$F4YDs.LUghUFNFT1HcHPLOb.qzyxv/TD/cbyZXOj/XVQGJrDmT1q | 0 | keullag@sfsu.edu | 르박 | NULL | NULL |
| 10 | Golden30 | $2b$10$08Q2M4KrDksUBmART4KSMoZijNm7xCVXabeniTwuvutg3o20u3HHi | 0 | curry@sfsu.edu | Steph Curry | NULL | NULL |
| 11 | dlee40 | $2b$10$08Q2M4KrDksUBmART4KSMoZijNm7xCVXabeniTwuvutg3o20u3HHi | 0 | dlee40@sfsu.edu | Daniel | NULL | NULL |
| 12 | thang | $2b$10$bt05oo70icAtgQ3jrl79Seig7.qzePISe91sfsL2Vki749AUJBDRT | 0 | thang1@sfsu.edu | thang1 | NULL | NULL |
| 13 | yeet | $2b$10$BA7K6WpTKpAdrP0v3gjmTu1Kdx0GupsNh40khzQ.LHc1ha6HH/S | 0 | kye@sfsu.edu | Kan Ye | NULL | NULL |
| 14 | a | $2b$10$BA7K6WpTKpAdrP0v3gjmTu1Kdx0GupsNh40khzQ.LHc1ha6HH/S | 0 | asd@sfsu.edu | ads | NULL | NULL |
| 15 | thang1111 | $2b$10$28yrGiJ2h9SLV7TwE31UU/vhz2hW2QudQzZDfxxy0BmV01f.zYY2 | 0 | thang@sfsu.edu | tahng | NULL | NULL |
+----+-----+-----+-----+-----+-----+-----+-----+
15 rows in set (0.014 sec)

mysql> ■

```

GITHUB ORGANIZATION

List of main branches:

- Main
- Hotfix_connecting_api
- Milestone2_dev
- Milestone2
- Milestone3_img_dev
- Milestone3
- Milestone4_tentative

Master Branch access:

All team members had access to the main branch. However, we practiced caution by primarily working on feature branches and using pull requests for merging.

 **csc648-fa25-0104-team04** (Private)

forked from [CSC-648-SFSU/csc-648-sfsu-2025-fall-souza-01-csc648-base-repo](#)

[Watch 0](#) [Fork 0](#) [Star 0](#)

[7 Branches](#) [0 Tags](#) [Go to file](#) [Add file](#) [Code](#)

This branch is 190 commits ahead of [main](#).

[Contribute](#) [Sync fork](#)

C1ark22 Milestone 3 pdf upload · 24a2ddc · 45 minutes ago · 191 Commits

Milestones	Milestone 3 pdf upload	45 minutes ago
application	Message page fixing	20 hours ago
credentials	Updating Sever IP.	2 months ago
.gitignore	Starting and Creating Backend Stuff.	3 months ago
LICENSE	Update everything	yesterday
README.md	Update everything	yesterday
package-lock.json	Update everything	yesterday
package.json	Update everything	yesterday

[Readme](#) [MIT license](#) [Activity](#) [Custom properties](#) [0 stars](#) [0 watching](#) [0 forks](#)

About

csc-648-sfsu-2025-fall-souza-01-csc648-base-repo created by GitHub Classroom

Releases

No releases published [Create a new release](#)

Packages

No packages published [Publish your first package](#)

Deployments 1

[github-pages](#) 3 months ago

Languages

- JavaScript 99.3%
- Other 0.7%

csc648 Repository

Please when ready add your teams application URL or IP to the repository description. This will help with grading. Teams are expected to keep this value up to date.

Please do the following steps before completing Milestone 0.

1. Change the name of the repository. All that needs to change is the NN to your respective team number. Team numbers whose value is less than 10, please pad with a 0. Ex team 1 is Team01 team 11 is Team11. Please make sure to also remove the username from the repository as well. Teams with incorrectly name repository will have points deducted from their milestone 0 grades.
 - o Please follow the naming convention assigned by your instructor.
2. PLEASE REMOVE THE USERNAME FROM THE REPOSITORY NAME!!!
3. Add ALL members of your team to this repository. For it to count, they must ACCEPT the invite.
4. Fill out the table below

Student Name	School Email	GitHub Username
Clark Batungbakal	cbatungbakal@sfsu.edu	C1ark22
Thanh Duong	tduong7@sfsu.edu	DanielDoubleDx
Trinity Godwin	cgodwin@mail.sfsu.edu	CommNs
Daniel Lee	dlee40@sfsu.edu	danleepro108
Caleb Onuonga	jonuonga@sfsu.edu	calebsfsu

NO code needs to be stored in the root of your repository. You may rename the application folder if you like to your team's application name. But all source code related to your team's application should be stored inside the application folder.

GOOGLE ANALYTICS STATS

We were unable to download and present the Google Analytics summary report for our website. Although the analytics script was installed, there was not enough lead time for meaningful data collection before the final demo. This was due to my failure as the team lead to account for and prioritize the setup and testing of Google Analytics earlier in the project timeline.

I take full responsibility for this oversight and recognize the importance of analytics in demonstrating real user engagement and performance metrics. This is a lesson I will carry forward into future projects.

PROJECT MANAGEMENT

Our team used a shared **Google Docs** file for project management and coordination during the early stages of the project. This document was accessible to all team members and was used to:

- Take meeting notes
- Assign responsibilities
- Track weekly progress
- Record design and development decisions

We consistently used Google Docs until the beginning of April. After that, our workflow shifted to using **Discord** more heavily — not just for communication but also for informal documentation. We began sharing updates, tracking progress, and discussing technical issues directly within dedicated Discord channels, which became our primary coordination tool.

While this approach worked for faster collaboration, it resulted in less structured documentation compared to Google Docs. We recognize that balancing real-time communication with proper documentation is important for long-term clarity and accountability.

Feb 13, 2025 8:00 PM

Team Meeting for Milestone 0

Attendees Clark Batungbakal, Trinity Godwin, Thanh Duong, Caleb Onuonga, Daniel Lee
Attachments File

Agenda

Topic	Time	File	Team member
Selecting Server/Platform Provider	5 min	<input type="button"/> File	Everyone
Software Stack for deploying <u>team</u> web app	40 min	<input type="button"/> File	Everyone

Summary or key decisions

- Summarize the most important meeting outcomes
- Align as a group on the takeaway
- As a team we have decided what we will use for

Action items

- Ask permissions CTO for approval by 2/14/25 by 1700

Details

Raw notes

- Select a Server / Platform provider
 - AWS
- Select a Software Stack
 - Server Host
 - AWS free tier - EC2 instance free for 12 months
 - OS
 - Ubuntu 22.04.2 (ver. based on ur pc)
 - Web Server
 - Apache 2.4
 - Database
 - AWS RDS for MySQL ver 8.0
 - Server-side language
 - Framework
 - Express.js 4.18+
 - Front end
 - React (Next.js)
 - Back end
 - JavaScript (Node.js) 20.x
 - Other important tech for software stack
 -

Ideas for later

- Use this area to add an idea, topic, or question for future consideration

Feb 20, 2025 8:00 PM

Milestone 0 Task 5

Attendees Clark Batungbakal, Trinity Godwin, Thanh Duong, Caleb Onuonga, Daniel Lee

Attachments File

Agenda

Topic	Time	File	Team member
Show an example of how to run the website	15 min	<input type="button"/> File	Everyone
Front-end <u>start</u> on the webpage	40 min	<input type="button"/> File	Everyone

Summary or key decisions

- Summarize the most important meeting outcomes
- Align as a group on the takeaway
- As a team we have decided what we will use for

Action items

- Front end make the template and structure for each team member. Add a little bit of CSS

Details

Raw notes

■

Ideas for later

- Use this area to add an idea, topic, or question for future consideration

M1 task

Cadence Bi-weekly

Event March 6, 2025

Participants All team members present

Agenda

Topic	Time	File
Progress on M1?	30 min	File
	10 min	File
	8 min	File

Date

Project debrief

Project updates

Challenges and dependencies

- Executive Summary Feedback
- Personae
- Promote seller to Trusted sellers, buyer to Trusted buyers
- Back end – no specifics for data structures, what information needs to be stored and architecture?
 - More of a dictionary, using SQL for database, what stack for certain card, High level name of pokemon
 - Database for Pokemon and DB for users
 - Higher level implementation
- Add a filter for pokemon cards

Next steps

Action items

- Start thinking about the next task for the final project Person

Topics for future discussions

- Add topic Date
- Add topic Date

Project debrief

Vertical Software Prototype (VP)

- Build a minimal vertical prototype
 - It would be good to have a separate branch as a throwaway test
 - Requirements:
 - UI modeled after Amazon search
 - One pull down menu referring to **CATEGORIES**, it must have atleast 3 entries here (electronics, books, furniture, and default) this component of search is exercised using simple SQL filter (in the architecture slides Part II)
 - One **TEXT ENTRY** field (this text is then used in SQL LIKE search on text string obtained by concatenating item title and description a values from the DB)
 - Search arguments (selection of menu and typed text) must remain persistent after search is executed
 - In upper left corner, below search UI, it must show the number of items found
 - If user doesn't input anything on the search bar it must show all the items within the DB
 - The VP results page should show a list of search results, where each item includes a title, price, description, and image. To make it easy to read, group all the details for each item together—such as by using a border, box, or horizontal line to separate each one.
 - Create the VP test home page and VP results page using only your chosen front-end framework. Avoid hardcoding any data—this will help you practice and test how the front end communicates with the back end using the tools you plan to use in full development.

- The database should contain just a few sample items (about 2–3 per category). Each item must follow the complete schema as defined in the current M2 document. Make sure to initialize and populate the database accordingly.
- You don't need to develop the login and registration features for the VP right now—those will be handled later. For now, focus on implementing the core VP functions described above.

Task due March 31

- Create item table schema & sample data in DB, ensure that AWS deployment server is working **BACKEND TEAM**
 - Notes: add items for each categories mentioned above manually in workbench, ensure that server is publicly accessible
- make API endpoint contact (/api/search) **FRONTEND TEAM**
 - Decide on the JSON structure for request & response, if needed reach out to backend team/read the tutorials provided by the professor (can be found in Milestone 2 document)
- Build a basic backend API that accepts query params: category, text entries. Queries DB using LIKE. Test the backend locally with sample queries **BACKEND TEAM**
 - Run the test in the browser to ensure that its working properly when being called
- Create frontend search page and connect to backend API **FRONTEND TEAM**
 - Include a dropdown, text field, and search button similar to Amazon, use FETCH to call backend and display the results
- Use of GENAI notes **EVERYONE**

March 3, 2025



Clark 3/3/25, 10:17 PM

Milestone 1 assigned task:

1. Executive Summary - @Clark
2. Personas - @Daniel Lee
3. High level use cases - @thanh
4. Data Glossary - @Trinity Godwin
5. High Level functional requirements - @Caleb Onuonga & @Clark
6. Non functional requirements - @Clark (DONE)
7. Competitive analysis - @Clark & @Daniel Lee
8. High level system architecture & tech - @Caleb Onuonga

Deadline: Friday March 7, 2025

Please let me know if you can't do the task assigned to you, or if you have any question about it (edited)



3



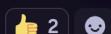
March 18, 2025



Clark 3/18/25, 11:40 PM

Milestone 2 Assigned task:

1. Executive Summary - @Clark
2. Data description & entities - @Caleb Onuonga @Trinity Godwin
3. Functional Requirements (Priority) - @Caleb Onuonga @Trinity Godwin @Clark
4. UI storyboards (WireFrames) - @Daniel Lee @thanh
5. High-level architecture & DB design - @Caleb Onuonga @Trinity Godwin
6. Risk Assessment - @everyone
7. Project Management Approach - @Clark
8. Use of GenAI tools - @everyone
9. Final Checklist & submission - @Clark



2



@everyone revisions should be done by 5 pm tomorrow

March 17, 2025



thanh ❤️ VIE 3/17/25, 5:53 PM

personae -> high level use cases:

1. searching/browsing
2. posting items
3. wishlist item
4. on-campus meetup points



Clark 3/17/25, 6:22 PM

5. Creating an account

P1: Creating an account, Posting Items

P2: searching/browsing

P3: On-campus meetup points, Wishlist items

March 27, 2025

Clark 3/27/25, 5:12AM
@everyone before today's meeting I'm expecting you to read Milestone 2 Part 2 and the document that I have attached here. I summarize what needs to be done for this project and thought of the things that is required for this part of the assignment. We will talk more later during the meeting

 Milestone 2 Part 2 project debrief.docx
17.16 KB

 1 

TEAM MEMBER SELF-ASSESSMENT AND CONTRIBUTIONS

Back End Lead - Caleb Onuonga

Backend Lead Self-Assessment & Contribution

JO Joel Caleb Onuonga
To: Clark Eullo Batungbakal; Trinity Liliana Godwin; Daniel Lee; Thanh Cong Duong
Tue 5/20/2025 1:41 AM

Dear Team,

In my role as Backend Lead, I was responsible for several key infrastructure components:

- I set up the AWS RDS server with managed instances to provide our database foundation
- I configured and deployed a managed SQL database to support our application needs
- Working collaboratively with the team, I developed backend APIs for various functionalities including image handling, posting, login, registration, and endpoint testing

Throughout the project, I contributed 29 commits to our codebase.

The main technical challenges I encountered involved getting the backend to work seamlessly with our frontend. Specifically, images would load correctly when testing locally but failed to display properly when deployed on our server. Additionally, my computer ran extremely slow throughout the project, which significantly impacted my development efficiency and workflow. However, I was able to overcome this challenge and get most of my work done despite my machine being slow.

I found generative AI tools particularly valuable during this project. They assisted with debugging issues, creating documentation, and providing reference materials for the various technologies we implemented as a team.

For future projects, I would improve in several areas:

- More thorough planning of database and backend application file structure to minimize bugs
- Scheduling additional coordination meetings with teammates
- Better time management strategies

The most significant learning outcomes from this project were enhanced time management skills, teamwork capabilities, and technical proficiency with AWS, SQL, Node, Express, Apache, and Nginx.

Best,
Caleb Onuonga

GitHub Master/Back End Member - Trinity Godwin

Team Member self assessment.

 Trinity Liliana Godwin
To: Ⓜ Clark Eullo Batungbakal
Cc: Ⓜ Thanh Cong Duong; Ⓜ Joel Caleb Onuonga; Ⓜ Daniel Lee

Tue 5/20/2025 12:00 PM

I acted as GitHub master and backend dev. In the beginning I mostly functioned to try and maintain proper branch / commit history and became more incorporated with the backend work toward the middle of the project. By the end most of my work then became backend. I would write the initial post route for images though that would take weeks of debugging to get right. I was responsible for the milestone4 rewrite of the backend, but development had progressed to much for that to be used so another rewrite was needed. Helped on some deploys as the system for doing so was not well understood in the beginning. Toward the end of the assignment, I would help write additional paths to get information to the frontend and post information to the backend. I also acted to bug fix and test backend functionality. I did database design planning and admin as well as redesigning core parts, especially messaging. Wrote the messaging backend path though that isn't full working from the front end as of me typing this. I would say I was very new to many technologies in terms of the depth needed for this project which led to many issues in the beginning but toward the later part of the later part of the project I felt comfortable in the backend tech stack.

[Reply](#) [Reply all](#) [Forward](#)

Front End Lead - Thanh Duong

Front-end Lead Self-Assessment & Contribution

 Thanh Cong Duong
To: Ⓜ Clark Eullo Batungbakal; Ⓜ Joel Caleb Onuonga; Ⓜ Trinity Liliana Godwin; Ⓜ Daniel Lee

Tue 5/20/2025 12:11 PM

Hello team,

As the front-end lead, I was responsible for designing and implementing the user interface of our application. I developed and connected the frontend to the backend API for three core components: the header, footer, and search bar. In addition, I built and integrated ten main pages, including Home, Posting, Login, Logout, and Register. I also organized the folder structures for both the frontend and backend to ensure better code maintainability and clarity for the team. On the backend, I created the product.js for the API listings endpoint to support the posting page functionality on the front end.

I made a total of 78 commits and consistently pushed my changes to the GitHub team's Dev branch.

One of the initial challenges I faced was deciding how to properly organize the folder structure for both the frontend and backend in a way that was clean and scalable for the team. However, the main technical issue I encountered was connecting the API for the Posting page. Specifically, I struggled with handling the category data, which I attempted to either send the category name directly or convert it to an index ID before passing it to the database, but the backend consistently declined the input and failed to store the data properly.

I used gen AI tools, specifically ChatGPT, to get feedback on my code, and they were quite effective. The AI provided useful suggestions and ideas that helped me improve the quality and structure of my pages. Through this process, I learned a lot about better practices in front-end development and how to refine my UI to make it more user-friendly and efficient.

This project taught me a lot about working in a team environment and building full-stack applications. I learned how important communication and structure are when collaborating on a shared codebase. I also gained deeper experience in handling API integration, database interactions, and solving real-world development issues.

Best,
Thanh Cong Duong.

[Reply](#) [Reply all](#) [Forward](#)

Front End Member - Daniel Lee

Team member self assessment and contribution



Daniel Lee

To: Clark Eullo Batungbakal



Reply

Reply all

Forward



...

Tue 5/20/2025 4:00 AM

i You forwarded this message on Tue 5/20/2025 11:25 AM

As a member of the front end team, my main job was the front end of the messaging. I added a few features here and there but mainly messaging. At the beginning I was doing the UI design and doc stuff; we did some of that vs live code too and it was pretty cool. The main struggle that I had was being able to test code. Throughout the project, I've made snippets of code but never integrated with the project. The most recent struggle was the integration of messaging with the backend. There is the cors error that I got stuck on. I guess the other thing was the cohesion of front and backend communication. Someone is always ahead of the other and it was tough to bring it together. Gen AI that I used was ChatGPT mainly for reading and interpreting code segments and looking for tiny errors. It came in handy when I was getting rid of axios. It converted the code just like that. Improvements for next time would be more communication so less toes were stepped on and smaller incremental changes for smoother transitions changing structure of code.

Get [Outlook for iOS](#)



Clark Eullo Batungbakal

To: Joel Caleb Onuonga; Thanh Cong Duong; Trinity Liliana Godwin; Daniel Lee



Reply

Reply all

Forward



...

Tue 5/20/2025 11:26 AM

Forwarding this message to all respective team members not directly to team leaders.

...

[Reply](#)

[Reply all](#)

[Forward](#)

Team Leader - Clark Batungbakal

CSC 648 Team Leader Self Assessment



Clark Eullo Batungbakal

To: Ⓜ Joel Caleb Onuonga; Ⓜ Trinity Liliana Godwin; Ⓜ Thanh Cong Duong; Ⓜ Daniel Lee

😊 ⏪ Reply ⏴ Reply all ⏵ Forward ⚡ 📎 ...

Tue 5/20/2025 8:04 PM

Good Evening Team 4,

As the team lead, I managed the team by assigning tasks, keeping track of progress, and ensuring that we met each milestone. I also stepped in as a fall back whenever additional work was needed such as handling tasks that weren't picked up or finished by others to keep the project on track.

I made

24 commits to the GitHub dev branch. I believe the number is lower than expected because a significant portion of my time was spent managing the team, resolving integration issues, and troubleshooting blockers instead of directly coding features.

The biggest challenge for me was managing the team and making sure everyone was aligned and contributing effectively to the project. Balancing coordination duties with hands-on development was tough. There were times when I wanted to step in and fix issues myself, especially when progress slowed or bugs lingered. But I knew that as a team lead, part of my responsibility was to trust my teammates and give them the space to learn and resolve problems on their own. It was important to support them without taking over their tasks, even when it tested my patience or instinct to jump in.

GenAI helped me a lot with interpreting code written by other team members, especially when I had to step in and assist or debug functions I wasn't originally assigned to. It saved me time and allowed me to quickly understand what was going on in different parts of the project.

Next time, I would schedule at least two recurring weekly meetings: one planning session to align tasks and priorities, and one dedicated code review session. This would help us improve communication and catch code issues earlier. I also feel I should have contributed more directly to the codebase to help the team deliver a more complete and functional final product.

Although my GitHub commit count is lower compared to others, I spent a considerable amount of time ensuring task delegation, resolving team issues, checking milestone status, and keeping communication active, especially when team productivity dipped or when blockers arose.

Respectfully,
Clark Batungbakal

⤪ Reply ⏴ Reply all ⏵ Forward

Overall feedback from team lead - Clark Batungbakal

Throughout this project, we experienced both individual growth and collective challenges as a team. Each of us contributed in different ways, and together we navigated a wide range of technical hurdles and coordination issues. It wasn't always easy, but we learned a lot about how to work together and adapt as the project evolved.

One of our biggest challenges was staying aligned and making sure everyone was contributing effectively. Balancing team coordination with individual development work was tough. There were moments when things slowed down or bugs persisted longer than expected. In those situations, it was tempting to jump in and fix things immediately, but we also understood the importance of trusting each other, giving each teammate the space to learn, troubleshoot, and grow from the experience.

On the technical side, we faced several roadblocks. At the beginning, we had a hard time settling on a folder structure that was both clean and scalable across the frontend and backend. One specific issue was integrating the Posting page with the backend, especially around category data—whether we sent the name or index, the backend just wouldn't accept it as expected.

We also ran into problems syncing frontend and backend behavior. Image uploads worked in local environments but often broke once deployed. One of our teammates had to work with a slow computer, which made development more challenging, but they still pushed through. Messaging integration proved difficult too, with CORS errors and mismatches between frontend and backend logic slowing us down.

Everyone contributed in different ways. One of our teammates started out managing GitHub branches and commits, and later became a key backend developer, rewriting routes, debugging issues, supporting frontend integration, and redesigning features like messaging and the database schema. Even though they were new to some of the technologies, they made consistent progress and became a reliable part of our backend workflow.

Looking back, one clear area where we could improve is communication. Having more structured and consistent meetings would've helped us stay more in sync. For future projects, we'd benefit from having at least two scheduled meetings each week—one for planning and one for code review—to better track progress and catch issues earlier.

While some of us focused more on leadership and coordination, we've all recognized the importance of balancing those roles with direct technical contributions. Moving forward, we plan to strengthen that balance to better support our teams both strategically and through hands-on development.

As a team, we found GenAI tools—especially ChatGPT—to be extremely useful throughout the development process. Different team members used it to interpret unfamiliar code, debug errors, generate documentation, and even rewrite API or Axios calls when switching libraries. It served as a quick feedback tool when we were stuck and helped us move past blockers without having to wait on someone else. While we didn't rely on it for everything, it sped up our learning and enhanced our problem-solving ability. For future projects, we'd continue using GenAI as a coding assistant, especially for troubleshooting and refining technical implementation.

This course taught us that software engineering is much more than just writing code. We learned the value of planning, documentation, and setting up workflows that allow for smooth collaboration. Processes like task delegation, regular team updates, version control discipline, and clear API documentation helped us manage complexity as the project grew.

We also learned about the importance of usability, security, and quality assurance in real-world software development. Concepts such as milestone-based planning, feature freeze periods, and structured testing helped reinforce what it means to deliver a reliable and maintainable product, not just a functional one.

Most importantly, we saw firsthand how crucial communication, time management, and adaptability are to the success of a software project. While tools and technology matter, it's the coordination and decision-making process behind them that makes or breaks a project. This class gave us a strong foundation in managing software development in a team-based environment, and we'll take those lessons with us into future academic and professional work.

