AGORA

SFSU Community Market Place

Software Engineering CSC 648
Spring 2025

Team 02

Team Members
Nathan Delos Reyes - Team Lead
Xiaoxuan Wang - Github Master
Jose Ramirez- Front-end Assistant
Ranbir Atkar - Front-end Lead
Jeshwanth - Back-end Lead

Milestone	Date Submitted
M1	March 12, 2025
Revised M1	March 14, 2025

Table of Contents

Executive Summary

Agora is a specialized buy-and-sell platform developed for the San Francisco State University (SFSU) community, including students, faculty, and staff. Recognizing the unique needs of our university community, Agora offers a secure, user-friendly marketplace designed to facilitate buying, selling, and trading university-related items such as textbooks, electronics, dorm necessities, and student services.

What makes Agora custom-built for SFSU is its focus on campus-specific needs, creating a community-driven platform tailored to support university life. Beyond buying and selling, Agora offers features such as skill sharing and listings for student-led services, including tutoring and tech help. The platform emphasizes ease of use, with powerful search and filtering functionalities that help students and faculty find what they need quickly and effortlessly, regardless of their technical skills.

Key functionalities include user messaging, enabling direct communication between buyers and sellers without requiring users to disclose their personal contact information. Product listings are thoughtfully designed, supporting multiple high-quality images and product descriptions to represent item conditions and features. Furthermore, a built-in rating system helps users establish and assess credibility, promoting accountability and transparency in every transaction.

Agora also prioritizes administrative efficiency by providing moderators with tools and analytics, empowering them to maintain platform integrity, manage disputes effectively, and address fraudulent or inappropriate activities. Automated flagging systems help identify suspicious listings, ensuring the marketplace remains secure, reliable, and compliant with community standards.

Our team is composed of SFSU Computer Science students committed to creating a meaningful and lasting solution that enhances campus life. As a student-led startup, we are connected to the real challenges and needs of our peers, and Agora reflects that firsthand experience. More than just a transactional platform, Agora is a solution designed to address and improve the buying and selling needs of the SFSU community. By investing in Agora, you are not only supporting a student-driven tech solution but also empowering students to build a stronger, safer, and more connected campus community.

Personae

Persona 1: Jacob - The Student Seller

About Jacob:

- A third-year business major at SFSU who wants to earn extra money by selling textbooks and used devices
- Busy schedule balancing classes, part-time job, and extracurricular activities
- Prefers a fast and easy way to list items and communicate with buyers
- Tech-savvy but doesn't want to spend too much time managing listings
- Worried about meeting with strangers and getting scammed when selling

Goals and Scenario:

- Wants to sell used textbooks instead of letting them sit and collect dust
- Needs an easy and quick way to list items with descriptions and photos
- Prefers to sell only to verified SFSU students
- Wants a secure and reliable way to coordinate meetups on and off campus
- Wants a rating system so he can see if a buyer is trustworthy

Persona 2: Molly - The Professor Buyer

About Molly:

- A part-time lecturer at SFSU who looks for affordable teaching materials and uses tech
- Not tech-savvy but can use simple websites
- Prefers verified sellers and clear product descriptions
- Wants a safe and hassle-free way to buy items from students or faculty
- Concerned about fraud or unusable products when purchasing online

Goals and Scenario:

- Looking to buy a second-hand tablet for a class presentation
- Prefers filters and search options to find specific items
- Wants an easy method to contact sellers without needing to share personal details
- Wants a policy that protects buyers or a way to report fraudulent sellers

Persona 3: Sophia - The Admin Moderator

About Sophia:

- A graduate student working as a moderator for Agora
- Responsible for removing inappropriate listings, and disputes, and ensuring Agora is safe
- Has strong technical skills and is comfortable navigating admin dashboards
- Concerned about potential scammers, policy violations, and maintaining user trust
- Wants efficient tools to moderate content

Goals and Scenario:

- Needs an automated flagging system for suspicious or inappropriate listings
- Wants a clear way to verify SFSU students
- Has to handle disputes between buyers and sellers
- Wants analytical tools to track Agora activity
- Wants a way to remove scammers

Persona 4: David - The first time buyer

About David:

- First year CS student at SFSU
- Tech Savvy but new to buying and selling platforms
- Cautions about online transactions and hesitant about second hand purchases
- Spends a lot of time online on schoolwork and personal interests
- Prefers local buying options to avoid shopping tax and delivery fees
- Concerned about legitimacy and safety of purchasing on an online marketplace

Goals and Scenario:

- Looking to buy textbooks and notes for the upcoming semester
- Wants a simple, easy to use platform for buying and selling
- Prefers local items to avoid additional costs
- Wants the ability to check the status of items before purchasing
- Needs a secure and straightforward way to communicate with seller before proceeding with the purchase

High-level Use Cases

1. Seller interacting with Agora to post details about products they are selling

- Actors: Jacob (User), Agora (Software)

- Assumptions
    - Jacob is tech-savvy but doesn't want to spend too much time managing listings
    - Jacob wants to make some extra cash
    - Jacob is worried about meeting with strangers and getting scammed when selling

Jacob is a third-year business student at SFSU. He has old textbooks that have been collecting dust in his closet and doesn't want to simply throw them away. Jacob wants to sell his old textbooks and he knows other students need his old textbooks. He finds Agora and makes an account so he can start selling. He posts several of his old textbooks. Each post has several pictures and a short description.

2. Buyer messaging seller about product details and meetup

- Actors: Molly (User), Agora (Software)

- Assumptions
    - Molly prefers verified sellers and clear product descriptions
    - Molly wants an easy method to contact sellers without needing to share personal details
    - Molly is not tech-savvy but can use simple websites

Molly is a part-time lecturer at SFSU who looks for affordable teaching materials and uses tech. Molly is able to use search filters to find the specific products she wants. However, the item she wants only has one picture, and the post is missing information, such as condition and meetup location. Molly can easily find the message button when she navigates to the post. With messaging, Molly can ask for more pictures and questions about the item from the seller. Their communication is smooth, so they meet on campus and have a successful transaction through Agora.

3. Admin interacting with Agora to moderate content

- Actors: Sophia (Admin), Agora (Software)

- Assumptions:
    - Sophia has strong technical skills and is comfortable navigating admin dashboards
    - Concerned about potential scammers, policy violations, and maintaining user trust
    - Wants efficient tools to moderate content
    - Has to handle disputes between buyers and sellers

Sophia moderates content on Agora by reviewing and addressing flagged listings that violate platform policies. After receiving a notification about a reported listing, Sophia reviews the product description, images, and seller profile. By cross-referencing the seller's history, past reports and customer reviews, she finds out that the seller accidentally listed a restricted item, as the seller has no prior violations and their other listings comply with the rule. In another case, she investigates another report and finds out the seller has repeatedly posted prohibited items despite previous warnings. Given the repeated violations, Sophia suspends the account entirely to maintain the integrity of the marketplace.

4. Skill sharing on Agora

- Maria (Learner/Buyer), Agora (Software)

- Assumptions:
    - Maria is a senior CS student at SFSU
    - Maria (Learner/Buyer), Agora (Software)
    - Maria is tech-savvy and comfortable navigating online platforms.

Maria, a senior Computer Science student at SFSU, has recently developed an interest in learning how to knit as a way to relax and take breaks from her studies. Wanting to learn, she decides to use Agora. Maria creates an account on Agora and navigates to the Skill Sharing section. Using Agora's search and filtering, she looks for knitting lessons or people offering one-on-one tutoring in knitting. After browsing several listings, reading descriptions, and reviewing ratings of available tutors, Maria finds a few students who offer knitting lessons. She reaches out to one of them through Agora's messaging feature to inquire about availability, lesson format, and pricing. Maria feels confident that she will be able to connect with the right person to help her learn knitting.

5. Buyer interacting with search features and sellers

- David (Buyer), Agora (Software)
- Assumptions:
    - David is tech-savvy but new to online marketplaces.
    - David prefers local purchases to avoid extra costs like shipping and taxes.
    - David is cautious about legitimacy and safety, verifying sellers before buying.
    - David is looking for affordable second-hand textbooks and notes.
    - David prefers a simple and secure buying process, including clear listings and easy communication.

David, a first-year CS student at SFSU, is looking for affordable textbooks and class notes for the upcoming semester. New to online marketplaces, he's cautious about second-hand purchases and prefers local transactions to avoid extra costs. He searches on Agora, using filters to narrow results by price, condition, and location. While browsing, he finds a set of notes for a challenging class and a used textbook at a reasonable price. Wanting to ensure legitimacy, he checks seller ratings, reviews, and listing details before reaching out. Through Agora's messaging system, he asks the seller about the notes' completeness and arranges to meet on campus to inspect the textbook before finalizing the purchase. By

using Agora's search, filtering, and communication tools, David feels more confident making a safe and budget-friendly purchase.

Data Glossary

Data Glossary

1. **Agora**
    - **Meaning:** The name for the buy-and-sell platform.
    - **Usage:** This is the name of the website, and it will be used across the application, the database, and UI elements (e.g., the logo, footer). "Agora" represents the marketplace where transactions between buyers and sellers take place.
2. **Buyer**
    - **Meaning:** A registered user who purchases goods listed by sellers.
    - **Usage:** The Buyer has the ability to browse, search, and purchase listed items. Buyers interact with the platform's product listings and can add items to their cart for purchase. This role can also rate or provide feedback for the seller after completing a transaction.
3. **Seller**
    - **Meaning:** A registered user who lists items for sale on the platform.
    - **Usage:** Sellers are responsible for creating listings by adding a product description, price, and images. They can manage their items, track sales, and edit or delete their listings. Sellers are also responsible for interacting with buyers regarding inquiries or sales.
4. **Unregistered User**
    - **Meaning:** A user who has not signed up or logged into the platform.
    - **Usage:** Unregistered users can browse the platform and view available listings, but they cannot perform transactions such as purchasing or listing items. This role is for site visitors who are not registered.
5. **Registered User**
    - **Meaning:** A user who has signed up for an account using an email and password.
    - **Usage:** Registered users can log in and perform actions like purchasing items (as Buyers) or listing items for sale (as Sellers). They have access to their account details, transaction history, and can track both purchase and sales activities.
6. **List**
    - **Meaning:** The act of a Seller posting an item for sale on the platform.
    - **Usage:** Sellers create a listing by adding a product description, price, and images to the platform. A listing is stored in the database and made available for Buyers to view. It includes attributes such as the item name, description, price, and photos.
7. **Item**
    - **Meaning:** A good or product available for sale or trade on the platform.
    - **Usage:** Items represent the core products in the platform's marketplace. These include attributes like title, description, price, images, and current status (available, sold, etc.). Items are listed by Sellers and available for Buyers to purchase.
8. **Transaction**

- **Meaning:** A completed purchase or sale event between a Buyer and a Seller.
- **Usage:** Transaction records track interactions where a Buyer purchases an item listed by a Seller. It includes item details, purchase price, and transaction status (completed, pending). This record is crucial for tracking user purchases and sales.

9. **Admin**
    - **Meaning:** A user with administrative privileges for managing platform operations.
    - **Usage:** Admins have the ability to view and manage user accounts, moderate content (approve/reject listings), and resolve disputes. They are responsible for maintaining platform integrity, ensuring compliance with terms of service, and making backend changes.
10. **Skill**
    - **Meaning:** A specific technique to do something well
    - **Usage:** Skills are a unique feature of Agora. They can be listed and buyers are given access to tutorials or lessons from the seller to learn their desired skill. Skills can range from getting tutoring for a subject or how to code.
11. **Tech-savvy**
    - **Meaning:** A term used to describe a user who is proficient and comfortable with using technology, including websites, mobile apps, and online platforms.
    - **Usage:** This term is used to characterize users, like students or faculty, who can easily navigate Agora's interface and utilize its features without requiring additional assistance. "Tech-Savvy" users are expected to perform tasks such as searching for products, posting listings, and messaging other users independently.

User Roles and Permissions

1. **Buyer (Role)**

   Permissions:

   - View and browse items listed for sale.
   - Purchase items.
   - View their purchase history.
   - Provide ratings and reviews for sellers after transactions.

2. **Seller (Role)**

   Permissions:

   - List items for sale and manage their listings.
   - View their sales history.
   - Modify or delete their listings.
   - Respond to buyer inquiries.

3. **Unregistered User (Role)**

   Permissions:

   - Browse items on the platform.
   - Cannot purchase or list items without registration.

4. **Admin (Role)**

   Permissions:

   - Full access to all user accounts and platform data.
   - Moderate content (approve/reject listings).
   - Manage disputes and handle platform-wide settings.

Data Structures/Entities

1. **Product Listing**
   - **Attributes**: Title, description, price, seller_id (foreign key), status (available, sold), images, created_at, updated_at.
   - **Usage**: Stores product information added by sellers, including descriptions, pricing, and images. Each listing is linked to a Seller via the seller_id foreign key.
2. **Transaction**
   - **Attributes**: Buyer_id (foreign key), seller_id (foreign key), item_id (foreign key), transaction_date, total_amount, status (completed, pending).
   - **Usage**: Captures details of completed or ongoing transactions. Includes references to the Buyer, Seller, and Item involved in the transaction.
3. **User**
   - **Attributes**: User_id, email, password_hash, role (buyer, seller, admin), created_at, updated_at.
   - **Usage**: Stores user credentials, role information, and timestamps for user account creation and updates. This table tracks all platform users and their associated roles.
4. **Review/Feedback**
   - **Attributes**: Review_id, transaction_id (foreign key), rating, comment, created_at.
   - **Usage**: Stores reviews and ratings provided by Buyers after completing a purchase. This helps to build the credibility of Sellers and provides feedback for improvement.

High-level Functional Requirements

Unregistered Users

1. Unregistered Users shall register with email and password.
2. Unregistered User shall become a registered user by creating an account.
3. Unregistered Users shall view posts.
4. Unregistered Users shall scroll through listings and view photos and descriptions.
5. Unregistered Users shall use keywords to search for products.
6. Unregistered Users shall view skills and skill-sharing post descriptions.

Registered Users

7. Registered Users shall view posts.
8. Registered Users shall scroll through listings and view photos and descriptions.
9. Registered Users shall view skills and skill-sharing post descriptions.
10. Registered Users shall buy products.
11. Registered User shall sell products.
12. Registered User shall sell services or skills.
13. Registered Users shall offer tutoring or skill tutorials.
14. Registered Users shall bundle multiple products into one purchase.
15. Registered Users shall upload images for listings.
16. Registered User shall describe products for sale.
17. Registered User shall report fraudulent or inappropriate products.
18. Registered Users shall message sellers.
19. Registered Buyer shall decide payment methods.
20. Registered Buyer shall set up meeting points for transactions.
21. Registered Buyer shall rate sellers after purchases.
22. Registered Users shall use keywords to search for products.

Admin

23. Admin shall track total sales.
24. Admin shall view registered users.
25. Admin shall remove flagged posts.
26. Admin shall remove flagged registered users.

Non-functional Requirements

1. The application shall be developed, tested, and deployed using tools and cloud servers approved by Class CTO and as agreed in M0
2. The application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers
3. All or selected application functions shall render well on mobile devices (no 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. The privacy of users shall be protected.
8. The language used shall be English (no localization needed)
9. The application shall be very easy to use and intuitive
10. Application shall follow established architectural patterns
11. Application code and its repository shall be easy to inspect and maintain
12. Google Analytics shall be used
13. No e-mail clients 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 as specified in the class, including collaborative and continuous SW development and GenAI tools
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).

Competitive Analysis

3-4 competitive products: Facebook Marketplace, eBay, Craigslist, OfferUp

*(1 star): Very poor or missing; the feature is either unavailable or significantly flawed.
**(2 stars): Below average; minimal functionally, limited usability, or poorly implemented.
***(3 stars): Average; provides basic functionality but lacks ease of use, advanced options, or thorough integration.
****(4 stars): Good; effective and user-friendly, with minor improvements possible.
*****(5 stars): Excellent; exceptional usability, comprehensive functionality, and outstanding implementation with little to no room for improvement.

Feature	Facebook Marketplace	OfferUp	Craigslist	Agora
User Account	*****	****	**	***
Messaging	****	*****	*	****
Posting	*****	****	***	***
Categories & Filters	****	*****	**	***
Searching	*****	****	***	**
Rating users/stars	***	*****	*	***
Special feature Skill Sharing and gig postings	*	**	***	****

Agora stands out by specifically serving the SFSU community, unlike general marketplaces such as Facebook Marketplace or OfferUp. Users must have valid campus credentials to register, ensuring that all participants are directly affiliated with SFSU. This creates a reliable and secure environment from the outset. Additionally, Agora's features, such as messaging and posting, are simplified to align with campus activities, making transactions straightforward. Users can conveniently arrange meetings at familiar campus locations rather than unfamiliar areas, reducing uncertainty compared to more open platforms.

Because Agora focuses on university-specific items, listings are more relevant. Categories and filters allow users to quickly find textbooks, dorm supplies, or club merchandise without sifting through unrelated listings. Additionally, Agora supports skill sharing and gig posting, allowing registered users to offer services such as tutoring and other skill-based activities. Furthermore, the built-in rating system and connection to SFSU accounts enhance accountability, unlike platforms that allow anonymous interactions. Overall, Agora provides a focused, secure, and community-oriented marketplace tailored specifically to the needs of the SFSU community.

High-Level System Architecture

1. Frontend:

- **HTML** provides the structure of the webpage, ensuring semantic and accessible content.
- **CSS 3** Styles the webpage with modern features like Flexbox, Grid, and animations for responsive layouts.
- **Javascript** enables interactivity and dynamic behavior on the client side.
- **Svelte**, a lightweight frontend framework
- **Responsive Design Principles** ensure the app is optimized across various devices using flexible grids and media queries.

2. Backend:

- **Python** for its simplicity and readability, allowing us to quickly develop and scale the backend.
- **FastAPI** is a modern framework ensuring high performance.
- **Gunicorn**, a robust WSGI server, handles the incoming traffic and distributes it across multiple processes.
- **Unicorn** is a super fast ASGI server that works seamlessly with FastAPI.

3. Database:

- **MySQL**, hosted on **AWS RDS**, for storing and managing user data and items.
- We use **SQLAlchemy ORM** to interact with the database in a more Pythonic way, instead of writing raw SQL queries.
- Connection pooling and secure credentials help us ensure smooth performance and data security.

4. Infrastructure:

- Running everything on an **Ubuntu Linux Server** for a stable environment.
- **Nginx** acts as a reverse proxy, handling incoming requests and forwarding them to FastAPI, as well as terminating SSL connections.
- We've set up **Let's Encrypt** for easy and free **SSL/HTTPS** certificates, ensuring secure communication.
- **AWS EC2** hosts our application and database in the cloud, making it scalable and accessible.

5. Deployment and Version Control

- **Git** is used for Version control, ensuring we can track changes and collaborate efficiently.
- **GitHub** is used to manage our project and allows for easy collaboration.
- **GitHub Actions/ Terraform and Jenkins** will be integrated to automate the deployment and testing processes.
- **Docker** is used for containerizing the application, making it easier to deploy across different environments.

How Everything Interacts:

- **Frontend**: Users interact with the web application built using **HTML**, **CSS3**, **JavaScript**, and **Svelte**. The frontend sends HTTPS requests to the backend API.
- **Nginx**: Receives user requests, manages SSL termination via **Let's Encrypt** certificates, and forwards requests to the **FastAPI** backend.
- **FastAPI** & **Gunicorn**/**Uvicorn**: FastAPI receives requests from Nginx and processes them using application logic. Gunicorn manages multiple Uvicorn ASGI workers to handle concurrent traffic efficiently.
- **SQLAlchemy** & **MySQL** (**AWS RDS**): FastAPI interacts with MySQL via SQLAlchemy ORM for database queries, updates, and transactions.
- **Response Cycle**: Data retrieved from the database is sent back through FastAPI to Nginx, returning responses securely to the user's front end.
- **Deployment**: Git manages code versions; **GitHub Actions** automates deployment and testing, and Docker ensures consistent environments.

Infrastructure Components

- **Server Host:** AWS vCPU (1 core)
- **Memory:** 1 GiB RAM
- **Operating System:** Ubuntu 22.04.4 LTS
- **Database:** MySQL 8.0+
- **Web Server:** NGINX 1.27.4
- **Server-Side Language:** Python 3.11+

Additional Technologies

- **Web Framework:** FastAPI 0.109.1 (Stable)
- **IDE:** VSCode 1.88.0 (Stable)
- **Application Server:** Gunicorn 21.2.0 (Stable)
- **ASGI Server:** Uvicorn 0.26.0 (Stable)
- **SSL Certificate:** Let's Encrypt (Latest)
- **Containerization:** Docker 25.0.3 (Stable)
- **ORM:** SQLAlchemy 2.0.27 (Stable)

Deployment Requirements

- **Container Orchestration:** Docker Compose 2.24.1 (optional for development)
- **Version Control:** Git 2.43.0
- **CI/CD:** GitHub Actions (Latest)

Browser Compatibility

- **Google Chrome:** 120+
- **Mozilla Firefox:** 115+
- **Apple Safari:** 17+
- **Microsoft Edge:** 120+

AI Disclosure

1. **ChatGPT(OpenAI):**
    - Helped generate code snippets for routine tasks, reducing the need to type everything from scratch.
    - Useful for troubleshooting and figuring out complex server configurations (e.g., Nginx and Gunicorn setups).
    - Assisted with writing documentation and explaining technical concepts clearly.
    - Even helped debug some tricky errors during the development process.

How Useful Were These Tools?

- **ChatGPT: MEDIUM**
- **What's Great:** ChatGPT was incredibly helpful for clarifying doubts, brainstorming ideas, and offering explanations for complicated concepts. It helped us speed up the design and documentation process.
- **Room for Improvement:** Sometimes, ChatGPT offered solutions that were a bit too complicated for the task at hand, or it didn't always understand the full project context. So, while it was great for helping us think through problems, it still needed human verification.

Team and Roles

Name	Email	Role
Nathan Delos Reyes	ndelosreyes@sfsu.edu	Team Lead
Ranbir Atkar	ratkar1@sfsu.edu	Front-End Lead
Jose Ramirez	jramierz6@sfsu	Front-End assistant
Jeshwanth Ravindra Singh	jsingh28@sfsu.edu	Back-End Lead
Xiaoxuan Wang	xwang35@sfsu.edu	GitHub Maintainer

Team Lead Checklist

So far, all team members are fully engaged and attending team sessions when required: ON TRACK
The team found a time slot to meet outside of the 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: ON TRACK

The team reviewed class slides on requirements and use cases before drafting Milestone: DONE

The team reviewed non-functional requirements from the “How to start...” document and developed Milestone 1 consistently: DONE

The team lead checked the Milestone 1 document for quality, completeness, formatting and compliance with instructions before the submission: DONE

The team lead ensured that all team members read the final M1 and agree/understand it before submission: DONE

The team shared and discussed the experience with GenAI tools among themselves: DONE

GitHub is organized as discussed in class (e.g. master branch, the development branch, the folder for milestone documents, etc.): DONEx