Maverick Home Center

CSE-5325-004: Software Engineering: Management, Maintenance & Quality Assurance

Group #13 - Mavericks

Team Members

S. No.	Student Name	UTA ID
1.	Aditya Vikram Bhat	1002014494
2.	Akash Biswas	1002055500
3.	Amit Gupta	1002066302
4.	Ravi Rajpurohit	1002079916

Maverick Home Center

Maverick Home Center is an app that provides home décor, used furniture, and all other accessories that one needs/wants to study. This app is for students at the University of Texas at Arlington and living close to the campus. This would allow students to have cheap (possibly free) and affordable furniture and would save the time and resources it would take to buy them firsthand or from other sellers. Maverick Home Center is here to help you whether you want to sell things you no longer need or if you're seeking a certain item at a good price.

There are many online marketplaces for buying and selling used goods, including eBay, Facebook Marketplace, Etsy, and Ebid. However, the main issue with the worldwide market is that anyone can pose as a buyer or a seller, which has the potential to defraud other users. This would ultimately result in a loss of faith in the system. Additionally, the user interface and user experience of the current market are so overwhelming that any first-time user could run into problems while attempting to complete fundamental tasks on the application.

The main advantage of having this application is that it is made by students with the aim to provide a seamless and easy-to-access place where one can get all the items one needs to have a convenient and easy journey to studies and have an immersive college life experience. Maverick Home Center, which is only available to students at The University of Texas at Arlington, overcomes all the aforementioned problems by gathering all the products in one location. Here, a buyer and seller can interact to lay the groundwork for negotiation and to have a better understanding of the state of the commodity. There would be no chance of being duped because the application only allows students who have been validated by the university to register. Additionally, the user can sell or buy a thing with just a few clicks because of the interactive design's simple user interface.

Overall Design Approach

The application would be developed by following the Object Oriented Design. Object-oriented programming (OOP) is a programming methodology that emphasizes the use of objects and classes to represent real-world entities and concepts. Object-oriented design works around the entities and their characteristics instead of the functions involved in the software design. There would be two major entities involved in the project -

- Buyer
- Seller
- Platform Maintainer

The application will support certain features which will allow the user to do the following actions -

- Searching for the product
- Placing a buy request
- Communicating with the sellers

Also, the sellers will be able to do certain actions too -

- Upload product images
- Insert product descriptions
- Communicate with the buyer

The development would follow a Top-Down design approach by utilizing Agile Methodology. The entire application would be broken down into multiple sub-systems and components. Then, those sub-systems would be analyzed and worked upon until the entire application is built. The tentative sub-systems defined are mentioned following -

- The primary goal is to create a personalized onboarding and login system to collect initial custom data. This data will be utilized to develop the following sub-systems.
- This would be followed by the development of the landing or items screen subsystem.
- The final step is to develop the payment gateway so that the payment record is created and stored.
- At the end, the process of testing would take place to ensure quick deployment of the completed subsystem.

System Design

Design for Database:

The application will utilize Firebase for storing all the data. The data would be stored as JSON objects as a tree. For example:

```
{
        "users": {
                "1002012345": {
                        "firstName": "Adam",
                        "lastName": "Smith",
                        "mavId": 1002012345,
                        "phoneNo": 6821234321,
                        "dob": 929173819738,
                        "phoneExt": "+1"
                },
                "1002012346": {
                        ...
                }
        }
}
```

User Interface Aspects:

Inputs:

The system enables both buyers and sellers to interact with it in multiple ways. It allows users to upload media files, including product images, and record geographical data in terms of latitude and longitude. Additionally, buyers can fill out forms to input product specifications.

Outputs:

Users can search for specific products based on selected filters and search criteria. The application also sends push notifications to both buyers and sellers when a message is received. Buyers have the ability to track their product listings.

User Interface Design:

Login/ Sign-Up Screen:

The login or Sign-up window will enable authentication for the application's user and also allow new users to onboard to the application.
Sign-up would require basic details such as First Name, Last Name, UTA Email, UTAID, Date of Birth, and phone number.
Once the user has signed up, they can always log in with their registered details

Dashboard Screen:

The initial dashboard view for both buyers and sellers will show a list of products available for sale in the local vicinity. Additionally, there will be a floating action button that enables users to access the product selling page.

Profile Screen:

The profile screen would contain all the basic actions such as view profile, edit profile, logout, and settings.

Product Feed Screen:

The purpose of this screen is to enable users to generate a product listing. It begins by allowing the user to upload an image of the product, followed by input fields for product details such as name, listing price, specifications, and pick-up location.

Chats Screen:

On this screen, the user can view a list of all active ongoing communications with other users. Each chat item includes information about the last message time and the number of unread messages. Clicking on an item will lead to the direct chat screen.

Notifications Screen:

This screen would display any push notifications and provide an option for the user to mark all notifications as read.

Requirements

Functional requirements:

- **FR1: Payment Gateway Flexibility**
  The system must support multiple payment gateways, including credit/debit cards, UPI payments, and financial technology companies such as Paypal, Zelle, and Venmo.

- **FR2: Cross-Device Functionality**
  The platform should be accessible on various devices, including mobiles, tablets, and PCs, as most users will likely attempt to use it from portable devices.

- **FR3: Device Usage Monitoring**
  The system should track the number of users accessing the platform from different devices to determine which device is most popular among users and prioritize its development.

- **FR4: Filterable Product Features**
  The site's homepage should include filterable product features, allowing users to sort products based on their intended use.

- **FR5: User Login**
  Users must create a login ID associated with their account to track orders. The system should include the option to create an account if the user does not have one.

- **FR6: Streamlined Checkout and Order Tracking**
  To facilitate easy purchasing, the system should provide a smooth checkout process and straightforward order tracking.

Non-Functional requirements:

- **NFR1: Social Media Integration**
  The platform should feature links to all relevant social media handles so that users can access important updates and announcements related to our e-commerce website.

- **NFR2: User Data Protection**
  To ensure user privacy, the system must safeguard personal data such as mobile numbers, addresses, payment details, and other sensitive information.

- **NFR3: Efficient Performance**
  The website must load quickly and remain responsive even during periods of high traffic or site congestion. We can establish performance benchmarks to measure the speed of functionality and integration loading.

- **NFR4: Scalability**
  After the initial release, the platform should be scalable to accommodate additional functionalities, integrations, features, servers, memory, and computations without compromising performance.

Interface requirements:

- **IR1: User-Friendly Interface**
  The platform must be easy to use for all users, regardless of their technical knowledge and experience with using websites.

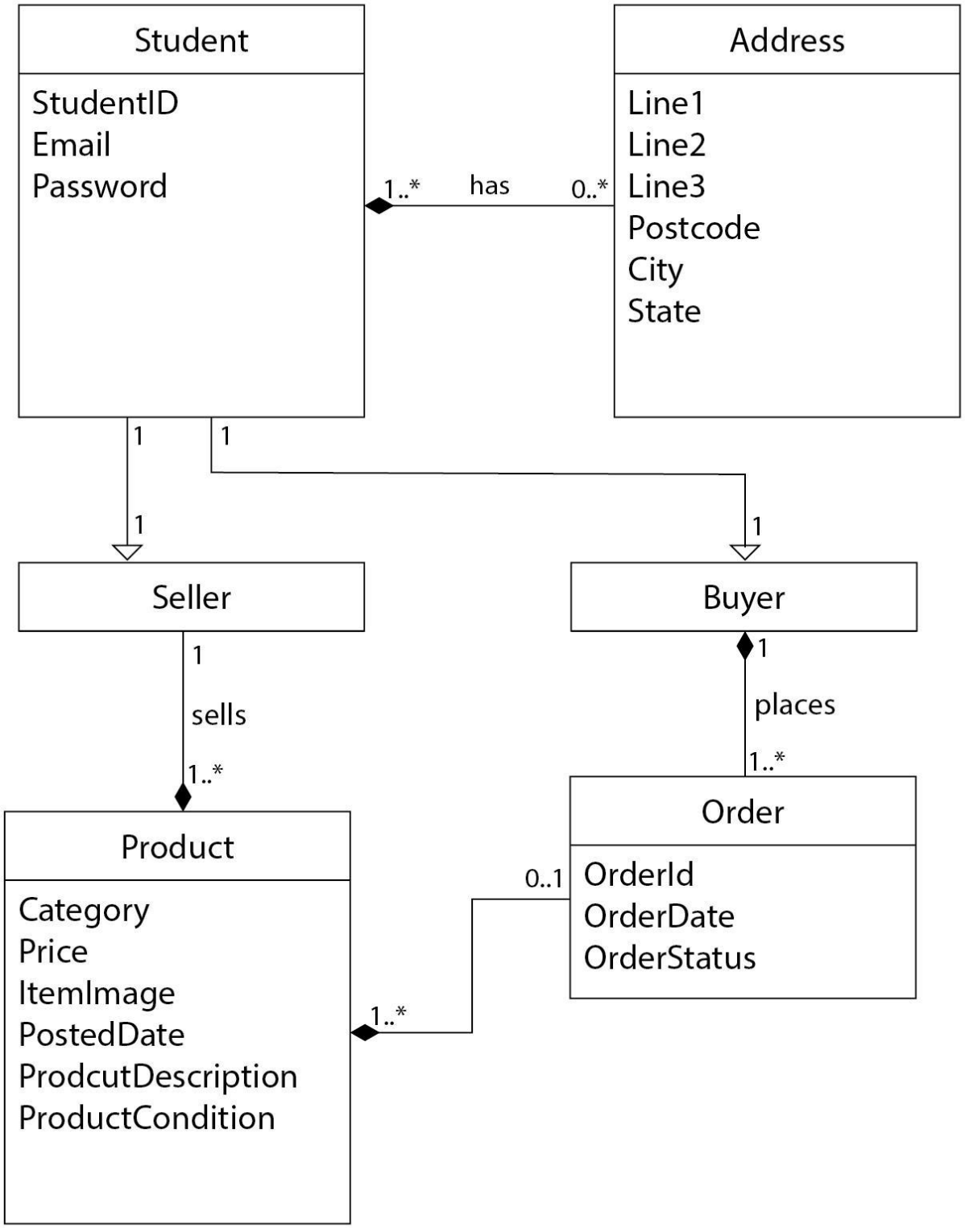
- **IR2: User Feedback Collection**
  To gauge usability, the system should offer a survey to users after they have spent some time on the website. The survey should focus on their ability to find what they were looking for, completion of tasks, and navigation for uncertain situations.

- **IR3: Product Image Requirements**
  Sellers must include at least three images of the product they intend to sell, showcasing its physical condition and appearance.

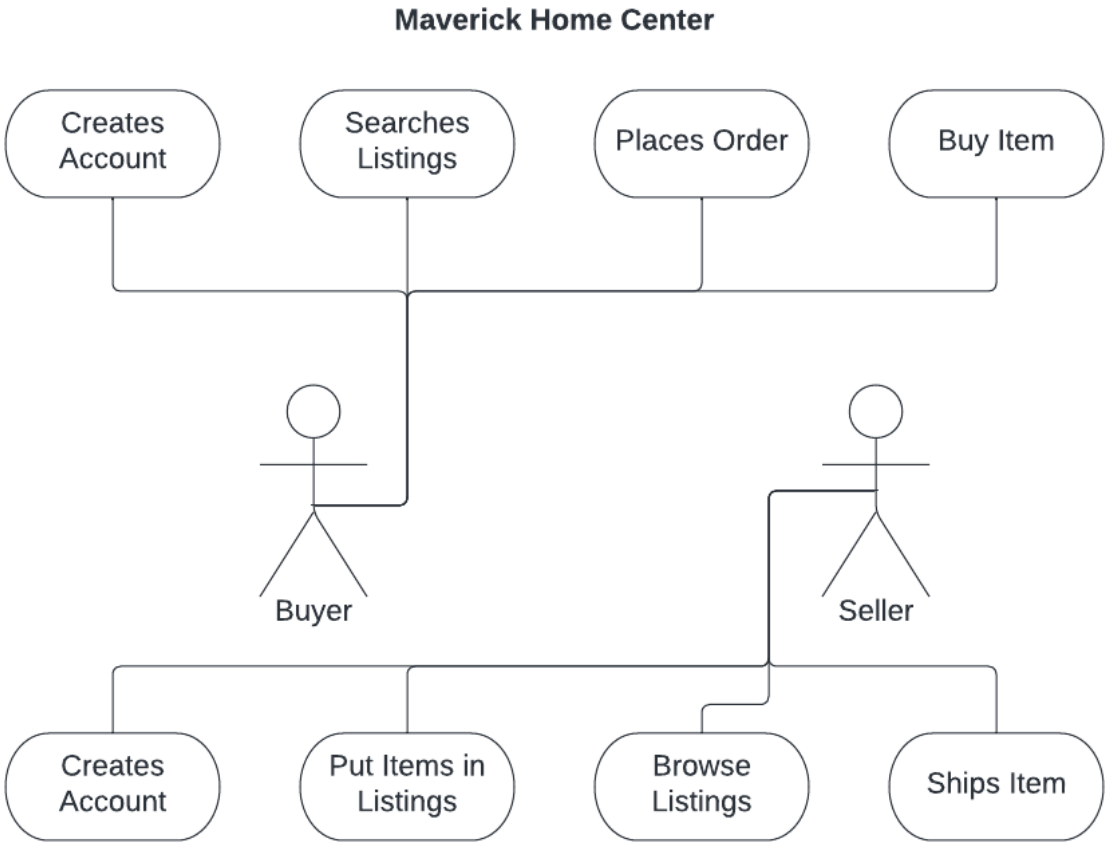
- **IR4: Demo Video Request**
  The platform can also request that sellers provide a demo video of the product they wish to sell, demonstrating its functionality, physical condition, and any internal damage or other issues.

Domain Model and Actor System Interaction Model

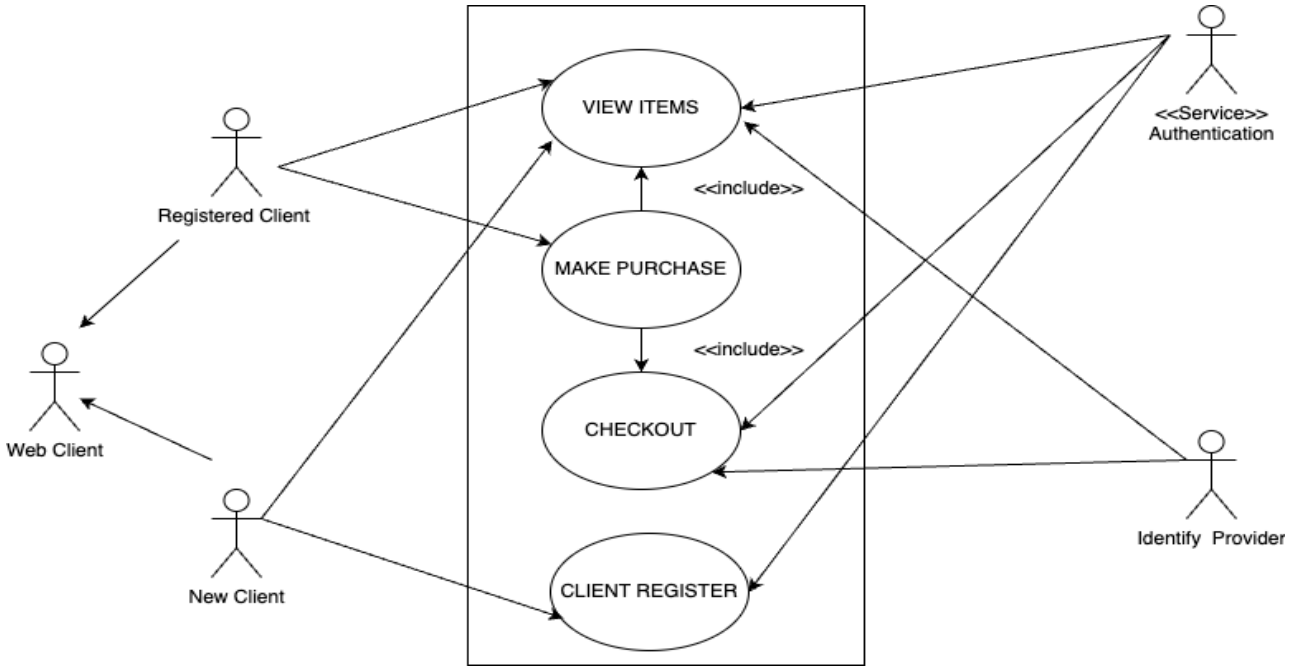
- **Domain Model:**



- **Actor System Interaction Model:**



Use Cases



Following are some of the use cases for this application -

UC1:

View Items - One of the first use cases is that both the kind of users - buyer or seller, can view the product items.

UC2:

Make Purchase - Once the user has found out the product of their interest, they can go ahead and follow the steps to purchase it. From the user's perspective, this is the most important use case.

UC3:

Check Out - After finalizing the product of interest, the user will follow the steps to checkout the product for delivery. This includes providing address information and payment for the product.

UC4:

Register - The user can register online and save their information to keep track of the products they like and buy them later. This helps in making the checkout process faster and keep the record of purchase history.

RUCTM

Requirement No.	Priority Weight	UC1	UC2	UC3	UC4
FR1	5		-	-	
FR2	5	-			
FR3	5		-		-
FR4	5				
FR5	4				
FR6	4	-			
NFR1	4				
NFR2	3				
NFR3	3			-	-
NFR4	3				
IR1	2				
IR2	2				
IR3	1				
IR4	1				
UC Priority		9	10	8	8

Note: Priority weight: 1 denotes low, 5 denotes highest priority

Increment Matrix

Use Case	Priority	Effort (Person-week)	Depends On	ITR 1 3wks	ITR 2 3wks	ITR 3 3wks	ITR 4 3wks
UC1	9	3	-		12		
UC2	10	3	-			12	
UC3	8	3	-				12
UC4	8	3	-	12			
Total Effort		12		12	12	12	12

Team Size = 4 People

Configuration Management

Configuration management is the process of identifying, organizing, controlling, and tracking the project's software and documentation. It involves managing the changes made to the software and ensuring that the software is delivered in a consistent and controlled manner. The following are the configuration management items, plans, and workflow for the project:

Configuration Management Items:

- Source code
- Build scripts
- Documentation
- Test scripts and plans
- Release notes
- Project management plans and reports

Configuration Management Plan:

This plan defines the configuration management processes and procedures to be followed throughout the project. It outlines the roles and responsibilities of the team members and defines the tools and techniques that will be used for configuration management.

The configuration management processes and procedures to be followed throughout a project are: configuration identification, control, status accounting, auditing, verification and validation, and reporting. These processes ensure that software and documentation are delivered in a controlled and consistent manner, reducing errors and improving quality while making it easier to manage changes.

Configuration Management Workflow:

- **Identification:** This involves identifying the configuration items, their versions, and their interdependencies. The team has identified all the items that are to be controlled, including source code, documentation, and build scripts.
- **Version control:** This involves creating a baseline for each configuration item and controlling changes to the baseline. We will be using git as a version control system to manage changes and ensure that each version is clearly identified. We plan to use GitHub as the git service provider.
- **Change control:** This involves reviewing and approving changes to configuration items. The team has a formal process for reviewing and approving changes. We have assumed a change control board (CCB) amongst ourselves that is responsible for reviewing and approving changes.
- **Build and release:** This involves building and releasing the software to stakeholders. We will be releasing the application for our use and include testing and quality assurance methods.

- **Auditing and reporting:** This involves auditing the configuration management processes and producing reports. We will be regularly auditing the configuration processes to keep track of the system while in progress and to ensure that they are being followed correctly.

By defining the configuration management items, plans, and workflow, the team can ensure that the software and documentation are delivered in a controlled and consistent manner. This will help to reduce errors, improve quality, and make it easier to manage changes to the software throughout the project.

People Management

Agile development methodology emphasizes the importance of collaboration, flexibility, and iterative development. To manage a team following agile, the team should be organized into cross-functional groups, with each group responsible for a particular aspect of the project. The team should be small enough to be easily managed, but large enough to provide the necessary expertise and skills. For this project, our team is of perfect size. The task division is done in a way that makes each team member responsible for a component of the project.

The team is self-organizing, meaning that the team members are responsible for determining how best to accomplish the project's goals. The team plans to work in short cycles or sprints, with each sprint typically lasting 1-3 weeks.

During each sprint, the team focuses on delivering a small set of working features, and then reviews and adjusts the approach for the next sprint. The team holds weekly stand-up meetings to keep everyone informed of progress and any issues that need to be resolved. There is constant communication on social media platforms as well to keep everyone in the loop about the project's progress. Additionally, the team plans to hold retrospective meetings at the end of each sprint to discuss what went well, what didn't, and how they can improve their processes for the next sprint.

The team encourages collaboration, provides guidance to each other, and helps one another to remove any obstacles that may hinder progress.

Overall, we are following the agile development methodology which makes our working model to be highly collaborative, flexible, and focused on delivering working features in short cycles. By following these principles, our team can more efficiently deliver high-quality software that meets the needs of their stakeholders.

Managing Software Quality

Since we are using the agile project development methodology, our high-level plan includes the following steps:

- **Define the quality standards and metrics:** Before the start of the project, we defined the quality standards and metrics that the team will use to assess the quality of the application. These standards and metrics are aligned with the project goals, stakeholder requirements, and industry best practices.
- **Conduct regular code reviews:** To ensure that the code meets the quality standards, we conduct regular code reviews throughout the development process. This helps identify and fix any coding errors, potential performance issues, and adherence to coding standards.
- **Implement testing:** We plan to test the app functionalities to catch any issues before they are released to users. This includes unit tests, integration tests, and end-to-end tests. We discuss and decide on test cases as part of each sprint, and the results would be used to guide future development.
- **Prioritize quality:** The team prioritizes the quality as part of the acceptance criteria. We have defined the non-functional requirements such as performance, security, and scalability. This would ensure that the application is built with quality in mind from the start.
- **Regular usability testing:** To ensure that the application meets the user's needs, we will conduct regular usability testing throughout the development process. This helps in identifying any issues with the user interface, user experience, and usability.
- **Consistently improve quality:** Quality is not a one-time activity, but a continuous process. Therefore, the team continuously monitors and improves the quality of the application. This includes regularly assessing the quality metrics, identifying any areas for improvement, and taking corrective actions as necessary.

Overall, our plan is to focus on building quality into the application from the start, ensuring that quality is a priority throughout the development process, and continuously monitoring and improving the quality of the application. By following these steps, the team can ensure that the application meets the needs of the stakeholders, meets the required quality standards, and provides a great user experience.