Contents - Milestone #1

Software Architectures - Team 13

1. **Arman Khondker**
2. **Ryan Lock**
3. **Alan Penichet-Paul**
4. **Varun Prabhu**
5. **Yulei Xu**

1. The Domain Problem and Stakeholders

1.1. Domain Description

The goal of this project is to design the architecture for an e-commerce website whose purpose is to facilitate the changing of hands of old business items such as outdated hardware from businesses to individual consumers or other small businesses.

The past decade has seen an era of relatively steady economic growth. In addition, in that period of time, technology has progressed rapidly. For example, a single Intel Xeon workstation CPU in 2010 would have had a maximum of 6 processing cores, while in 2019, a top of the line Xeon would have 28 processing cores.

Due to this rapid technological advancement, businesses often find themselves with obsolete equipment within a few years of purchase. More creative-oriented businesses that must stay on the forefront of technology to stay relevant may find themselves with a surplus of now-useless equipment. However, that old equipment may only be useless for that specific business or industry. There are other businesses, some in other fields, that could benefit greatly from such equipment, even if it's no longer cutting edge.

The e-commerce architecture planned for this project would serve as a platform for larger, more technologically or creatively oriented businesses to offload their old equipment quickly and conveniently to other businesses or individual consumers that would benefit from them. It serves as a platform to connect sellers of old business hardware with buyers, whether through automatically connecting a buyer with a seller after surveying the buyer's interests, or facilitating a buyer manually searching for a specific seller.

1.2. Envisioned System

Our system allows for users to be both a buyer and seller of surplus items from businesses and organizations or even individual sellers. This website is different from competitors because it serves a niche role in the market of reselling used items. Similar websites are very broad in the items allowed to be listed for sale but focusing our functionalities specifically on this audience of surplus items allows us to improve their experience. Buying used items in bulk often makes it so a business can save money without sacrificing much performance. For example, when a seller creates a listing, they can enable the ability for bulk sale and specify a minimum quantity needed to buy an order. If the

seller desires, they can set a discount for bulk purchases. Similarly, if the listing allows it, a seller can buy one individual item or a set of items in bulk.

The vision of our system is to connect sellers who would like to profit off their old hardware that is currently not being utilized with individuals or organizations who are in the market for used equipment. Users who would like to sell items can create listings with a description and pictures. Meanwhile, users who would like to make a purchase can browse the marketplace by searching and then refining the results by sorting and filtering. After a transaction has been made, features such as tracking shipment status and issuing refunds are built in to the website.

1.3. Stakeholders for the Architecture

1.3.1 Consumers

1. **Business Group**
   Expectations: The application should provide an ecommerce platform that provides large business with an aggregated platform to sell to their outdated business hardware. The final product should help businesses sell items at mass scale while also allowing them to sell in small quantities. Additionally, there should be support for smaller businesses to buy outdated hardware at scale.
   Role with respect to development project: Business expert, primary end user
   Role with respect to domain: Senior Management
   Part of organization they work: Management
   Impact: Unfinished Product Unscalable application, application with low ceiling of potential

2. **Individual Buyers:**
   Expectations: The application should provide an ecommerce platform that allows individual buyers to purchase outdated hardware from large businesses. The application should support account creation, transaction history, watchlists, and item recommendations. The buyers will want to
   Role with respect to development project: Customer, End User
   Role with respect to domain: Interested Buyers
   Part of organization they work: N/a
   Impact: Application that does not support purchasing, application with no viable end user

3. **Individual Sellers**
   Expectations: The application should provide individual sellers with an experience that allows them to effectively list their items and compete with the large corporations who exist on the platform. Additionally, individual sellers should not be at any sort of disadvantage when compared to large corporations to promote competitive pricing and capitalism.
   Role with respect to development project: Seller of items
   Role with respect to domain: Individual Seller
   Part of organization they work: N/a
   Impact: Application that is nonfunctional, application that cannot support the buyers needs

1.3.2 Producers

1. **Business Analyst:**

   Vision: The application meets the needs of all possible end users and sufficiently addresses all features required.

   Contribution: Meet with the Business Group, individual buyers, and individual sellers to discuss desired functionality and features.

   Impact if their vision is not met: The final application will not meet the requirements for our consumers.

2. **Software Architect:**

   Vision: The application needs to be supported by the software architecture designed by the software architects and meet the requirements that the business analyst sets forth. The architecture should set forth the standards that the application strictly abides by in order to preserve order and maintainability.

   Contribution: Software architecture for application

   Impact if their vision is not met: The software system will be unorganized and unmaintainable. Additionally, the functionality of

the system will not be well thought out. This will lead to disaster if the system gets implemented without their vision being met.

3. **Software Developers:**

   Vision: To turn the software architecture and user requirements into a quality software implementation. They want to develop a system that contains all required features and provides users with a seamless experience.

   Contribution: Develop the actual software implementation of the entire system.

   Impact if their vision is not met: Software will not be implemented properly and not be able to support the needs of end users. Additionally, the end users will not have access to all required functionality.

4. **Software Testers:**

   Vision: To ensure the system has no bugs and is reliable enough for release.

   Contribution: The software testers will extensively test the software with unit testing, regression testing, component testing, and load testing. They will identify features that have bugs and notify developers of their existence. Additionally, software testers will alert developers of potential risks that could occur with their software implementations.

   Impact if their vision is not met: The system will have bugs and not be able to support the end users.

1.4. Functional Requirements

1.4.1. Function Specifications

1. **Authenticate and Authorize:** This function allows a user to login to their seller/buyer account. After a user enters their username and password this functions checks the user database and then allows the user to log in to the website and access all website functionality.
   Input:
   Username (External),
   Password (External),
   Action Type [Login] (External)
   Event ‘User Requested Login function’ (External)
   Output:
   Logged into the website (all other functions will call this event)
   Performers: Businesses, Individual Buyers, Individual Sellers
   Resources: User database, Internet, Computer/Laptop, Website running
   Performance Location: Anywhere with internet, office of users
   Precondition: EventHasOccurred (User requested login function) And DataReceived (Username) And DataReceived (Password) And ActionType (Login)
   Post-condition: EventProduced (User logged into ecommerce platform)

2. **List an item:** This function allows a user to post an item for sale. On completion, the item will show up on buyers feeds and in the search feed. Additionally, this function adds the listed item to the sellers for sale list on their seller profile page.
   Input:
   Pictures of Item(External),
   Description of Item (External)
   Cost of Item (External),
   Item Condition (External),
   Action Type (List item),
   Event ‘User Requested List an Item Function’ (External)
   Output:
   Item listing made (External),
   Seller profile page updated (External)

Performers: Businesses, Individual Sellers
Resources: Items database, Seller Profile, Internet, Computer/Laptop, Website running,
Performance Location: Anywhere with internet, office locations of the seller
Precondition: EventHasOccurred (User requested login function) And DataReceived (Item Description) And DataReceived (Item Price) and DataReceived(Pictures of Item) And DataReceived (Item Condition) And ActionType (List Item)
Post-condition: EventProduced (User listed an item for sale)

3. **Buy an item:** This function allows a user to purchase an item. On completion, the buyer will be promoted to pay for the item. Additionally, the buyer will receive confirmation of the item order once payment has been received. The item will show up in the buyer's purchase history and we will recommend future purchases based on this purchase.
   Input:
   Item to be bought (List an Item),
   Action Type (Buy Item),
   Event 'User Requested to Buy an Item' (External)
   Output:
   Updated buyer purchase history (Internal),
   Future purchase Recommendations (Internal),
   Prompt user to pay for item (Pay for item)
   Performers: Businesses, Individual Buyers
   Resources: Items database, Internet, Computer/Laptop, Website running, Buyer Profile
   Performance Location: Anywhere with internet, office locations of the buyer
   Precondition: EventHasOccurred (User requested login function) And EventHasOccured(Seller listed this item) And DataReceived (Item to purchase) And ActionType (Buy item)
   Post-condition: EventProduced (User has purchased the item)

4. **Edit an item Listing:** This function allows a seller to change the details of an item that has previously been posted for sale. On completion, the item details will update and will show up again on all buyers feeds and in the search feed. Additionally, this function

updates the listed item on the sellers for sale list on their seller profile. It also updates the watchlists for buyers who already are watching this item.
Input:
Pictures of Item(External),
Cost of Item (External),
Item Condition (External),
Action Type (Edit item),
Event 'User Requested Edit an Item Listing Function (External)
Output:
Item details updated (External),
Update watchlist of buyers with item (Internal),
Performers: Businesses, Individual Sellers
Resources: Items database, Internet, Computer/Laptop, Website running, Seller Profile, Buyer Profile
Performance Location: Anywhere with internet, office locations of the seller
Precondition: EventHasOccurred (User requested login function) And/Or DataReceived (Item Description) And/Or DataReceived (Item Price) And/Or DataReceived(Pictures of Item) And/Or DataReceived (Item) And ActionType (Edit an Item)
Post-condition: EventProduced (Seller Edited an Item Listing)

5. **Check Recently Viewed Items:** This function allows a buyer or seller to go to their recently viewed items chart.
   Input:
   Action Type [Check Recently Viewed Items] (External)
   Event 'User Requested to View Recently Viewed Items' (External)
   Output:
   User will see a list of items that they have recently viewed (External)
   Performers: Businesses, Individual Buyers, Individual Sellers
   Resources: Items for sale database, Internet, Computer/Laptop, Website running, Recently Viewed List
   Performance Location: Anywhere with internet, office of users
   Precondition: EventHasOccurred (User requested login function) And ActionType (Check recently Viewed Items)
   Post-condition: EventProduced (User can view)

6. **Add an Item to watchlist:** This function allows a buyer to add an item that has been previously listed to their watchlist. On completion, the buyer's watchlist will be updated and show all items that they have added.
   Input:
   Item that has been previously Listed (List an Item)
   Action Type [Add Item to Watchlist] (External)
   Event 'User Requested to add Item to Watch List function' (External)
   Output:
   Buyers watchlist has been updated (Internal)
   Performers: Businesses, Individual Buyers
   Resources: Item database, Buyer Watchlist, Internet, Computer/Laptop, Website running
   Performance Location: Anywhere with internet, office of users
   Precondition: EventHasOccured (User requested login function) And EventHasOccurred (Seller Listed this Item) And ActionType (Add to watchlist)
   Post-condition: EventProduced (User has added an item to their watchlist)

7. **Add a description to the item:** This function allows a seller to add a section to their listing describing the item
   Input: Listing from database and description text from external source (user input)
   Output: Item listing updated
   Performers: Sellers(Business Groups/Individual Sellers)
   Resources: Listing database, Internet, Computer, Website running
   Performance Location: Anywhere with internet
   Precondition: EventHasOccurred (User requested login function) And EventHasOccured (Initial listing created)
   Post-condition: DataUpdated (description field in listing is updated)

8. **Attach images to the listing:** This function allows a seller to attach pictures of the item to their listing
   Input: Listing from database and images from external source (user input)

Output: Item listing updated
Performers: Sellers (Business Groups/Individual Sellers)
Resources: Listing database, Internet, Computer, Website running
Performance Location: Anywhere with internet
Precondition: EventHasOccurred (User requested login function) And EventHasOccured (Initial listing created)
Post-condition: DataUpdated (images in listing are updated)

9. **View shipping status:** This function allows a buyer to view the status of their order once purchased.
   Input: Order ID assigned by shipping service that is held in the order object
   Output: Status of order and estimated time of arrival
   Performers: Potential Buyers (Business Groups/Individual Buyers)
   Resources: Listing database, Internet, Shipping service
   Performance Location: Anywhere with internet
   Precondition: EventHasOccurred (User requested login function) And EventHasOccured (Transaction occurred)
   Post-condition: DataAvailable (Shipment status of order is viewable)

10. **Search for items:** This function allows a buyer to search for an item and find listing results that are relevant to their query
    Input: External user inputs a string that will then be used to query the items database
    Output: Returns a list of relevant item listings
    Performers: Buyers (Business Groups/Individual Buyers)
    Resources: Items for sale database, Internet, Computer/Laptop, Website running
    Performance Location: Anywhere with internet
    Precondition: DataReceived (Search keywords) And ActionType (Search for item)
    Post-condition: DataAvailable (All active listings that relate to the search keywords show up with basic information and can be clicked on to see individual details)

11. **Filter search:** This function allows a buyer to filter their search by brand, type of item, and price range
    Input: Search results of output from function 10 (Search for items)

Output: Returns a filtered list of items
Performers: Potential Buyers (Business Groups/Individual Buyers)
Resources: Items for sale database, Internet, Computer/Laptop, Website running
Performance Location: Anywhere with internet
Precondition: EventHasOccurred (Search for items has occurred) And ActionType (User chooses filters)
Post-condition: DataAvailable (All active listings that correspond to the filters show up with basic information and can be clicked on to see individual details)

12. **Sort search:** This function allows a buyer to sort their search results by price (ascending and descending)
    Input: Search results of output from function 10 (Search for items)
    Output: Returns a sorted list of items
    Performers: Potential Buyers (Business Groups/Individual Buyers)
    Resources: Items for sale database, Internet, Computer/Laptop, Website running
    Performance Location:
    Precondition: EventHasOccurred (Search for items has occurred) And ActionType (User chooses sort method)
    Post-condition: DataAvailable (All active listings show up in order of desired sort with basic information and can be clicked on to see individual details)

13. **Create shipping label:** This function allows a seller to print out a shipping label to attach to a packaged item once it has been purchased.
    Input:
    OrderNumber (Generate invoice),
    Output:
    Shipping Label (View Order Details - Seller)
    Performers: Businesses, Individual Sellers
    Resources: User database, Internet, Shipping service
    Performance Location: Backend application server
    Precondition: EventHasOccurred (User requested login function) And EventHasOccured (ProcessPayment) And DataReceived (OrderNumber)

Post-condition: EventProduced (Shipping label generated and displayed on ‘Order Details’ page)

14. **Remove item listing:** This function allows a seller to remove an item that they have listed before it has been purchased for any reason.
    Input:
    ItemNumber (Edit Item Listing),
    Output:
    ItemDeleted (View all of my listings)
    Performers: Businesses, Individual Sellers
    Resources: User database, Internet
    Performance Location: Backend application server after request submitted by seller on website.
    Precondition: EventHasOccurred (User requested login function) And EventHasOccured (List an Item) And (ItemNumber == Item to delete)
    Post-condition: EventProduced (Listing removed from seller’s store)

15. **Submit cancellation request:** This function allows a buyer to submit a cancellation request to the seller after they have purchased an item.
    Input:
    OrderNumber (View order details),
    Output:
    Cancellation Request Sent (View all of my listings)
    Performers: Businesses, Individual Buyers
    Resources: User database, Internet
    Performance Location: On website order details page by buyer/business.
    Precondition: EventHasOccurred (User requested login function) And EventHasOccured (Order processed) And (OrderNumber == Order to cancel) And EventHasNotOccured(Order has been shipped)
    Post-condition: EventProduced (Seller is notified that the order has been cancelled)

16. **View all orders:** This function creates allows a buyer to see a page with all of their past and current orders. They can click on an order to view more details and execute any additional functions on that specific item.
    Input:
    Username (Authorize and authenticate)
    Output:
    Can select any order on this page (View Order Details)
    Performers: Businesses, Individual Buyers
    Resources: User database, Internet
    Performance Location: On website home page by buyer/business.
    Precondition: EventHasOccurred (User requested login function)
    Post-condition: DataAvailable (All orders show up with basic information and can be clicked on to see individual details).

17. **View all of my listings:** This function allows a seller to view a list of all items that they currently have listed on the website. From here, the user can click on any listing to edit it.
    Input:
    Username (Authorize and authenticate)
    Output:
    Can edit any listing on this page (Edit a listing)
    Performers: Businesses, Individual Sellers
    Resources: User database, Internet
    Performance Location: On website home page by seller/business.
    Precondition: EventHasOccurred (User requested login function)
    Post-condition: DataAvailable (All active listings show up with basic information and can be clicked on to see individual details).

18. **View Order Details:** This function allows a buyer to see the status of their order and do things like request cancellations or check item details. Additionally, if a seller uses this function, they see the seller view of the order, including information about if the buyer has paid and viewing the shipping label.
    Input:
    OrderNumber(View all orders)
    Output:

Can interact with anything related to this order (Submit cancellation request, View Shipping Status)
Performers: Businesses, Individual Buyers
Resources: User database, Internet
Performance Location: On view all orders page by buyer/business.
Precondition: EventHasOccurred (User requested login function) And OnPage(View all orders) And (OrderNumber is VALID)
Post-condition: DataAvailable (Individual order viewable and can interact with it).

19. **Authenticate valid email:** This function confirms that the user provided a valid email during the creation of their account. This is accomplished by sending a validation link to the provided email, and confirming that the link is opened, to validate the email.
    Input:
    Email Address (External)
    Event ‘User Clicked Submit Email Button’’ (External)
    Output:
    User will see page stating “Check your email’s inbox to validate email address” (External)
    Performers: Businesses, Individual Buyers, Individual Sellers
    Resources: Internet, Computer/Laptop, Website running
    Performance Location: Anywhere with internet, office of users
    Precondition: EventHasOccurred (User submitted email)
    Post-condition: EventProduced (Email validated)

20. **Sign up for alerts:** This function allows user to sign up to receive alerts when a new item matching their search criteria is listed for sale. User sets search criteria and enables alerts setting.
    Input:
    Alert Criteria (External),
    Output:
    User added to email list for criteria.
    Performers: Individual Buyers
    Resources: User database, Internet, Shipping service , Item database, Email Service
    Performance Location: Backend application server

Precondition: EventHasOccurred (User requested Item alerts) And EventHasOccured (User submitted item criteria)
Post-condition: EventProduced (User added to email list for specified criteria)

21. **View FAQ:** This function redirects user to the FAQ page. The FAQ page contains answers to frequently asked questions, and also provides relevant links (e.g. contact information)
    Input:
    Event 'User requested FAQ Page' (External)
    Output:
    User redirected to FAQ Page
    Performers: Businesses, Individual Sellers, Individual Buyers
    Resources: User database, Internet
    Performance Location: Frontend web interface
    Precondition: EventHasOccurred (User requested FAQ)
    Post-condition: EventProduced (FAQ Page displayed)

22. **View size chart:** This function shows the relevant item's size chart, if one exists. Users access this by clicking "View size chart" button under the item's picture.
    Input:
    Event 'User requested Size Chart' (External),
    Output:
    User is shown size chart for relevant item.
    Performers: Businesses, Individual Sellers, Individual Buyers
    Resources: User database, Internet, Item database
    Performance Location: Frontend web interface
    Precondition: User is viewing an item. Item has a size chart available.
    Post-condition: EventProduced (size chart displayed)

23. **Contact Seller:** This function provides a direct message interface for users wishing to contact the seller of an item.
    Input:
    Event 'User requested to contact seller' (External)
    Output:
    User taken to 'Write message' screen with seller contact information filled out

Performers: Businesses, Individual Sellers, Individual Buyers
Resources: User database, Internet, Simple contact form
Performance Location: Backend application server
Precondition: EventHasOccurred (User requested Contact Seller)
Post-condition: EventProduced (Write message screen shown)

24. **Report Listing:** This function provides the user a form to fill out that submits a report about the listing upon completion.
    Input:
    Event ‘User Pressed report button’ (External),
    Output:
    User redirected to ‘Report listing’ form
    Performers: Businesses, Individual Sellers, Individual Buyers
    Resources: User database, Internet
    Performance Location: Backend application server
    Precondition: EventHasOccurred (User Reported Listing)
    Post-condition: EventProduced (User redirected to report listing)

25. **Edit Preferences:** This function allows the user to modify the type of items and sellers they wish to be suggested by the system.
    Input:
    Valid Login [Function 1]
    User Preferences [External]
    User Request [External]
    Output:
    Updated suggestions
    Performers: Businesses, Individual Buyers
    Resources: User database, Internet, Computer/Laptop, Website running
    Performance Location: Anywhere with internet, office of users
    Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (User requested to edit preferences) And DataReceived (New Preferences) And ActionType (Edit Preferences)
    Post-condition: EventProduced (New suggestions created)

26. **Track Shipping Label**: The user should be able to be redirected to the appropriate shipping company’s site if they click on their created shipping number

Input:
Valid Shipping Label [Function 13]
User Request [Externa]
Output:
Redirect to shipping company's website [External]
Performers: Businesses, Individual Buyers
Resources: User database, Internet, Computer/Laptop, Website running
Performance Location: Anywhere with internet, office of users
Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (User requested tracking feature) And DataReceived (Shipping label) And ActionType (Redirect)
Post-condition: EventProduced (Redirected to external website)

27. **Browse Suggestions**: The user can look through the generated suggestions of sellers and items based on their preferences
    Input:
    User Preferences [Function 25]
    User Request [External]
    Output:
    List of suggested sellers and interesting items based on preferences
    Performers: Businesses, Individual Buyers
    Resources: User database, Internet, Computer/Laptop, Website running
    Performance Location: Anywhere with internet, office of users
    Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (User accessing suggestions list) And DataReceived (Preferences and suggestions) And ActionType (Display suggestions)
    Post-condition: EventProduced (Shown list of suggested sellers and items)

28. **Issue Refund:** Upon successful completion of cancellation request or return request, the buyer is issued their money back.
    Input:
    Valid cancellation [Function 15] OR
    Valid return [Function 29]
    Output:

Money refunded back to original method of payment
Performers: Businesses, Individual Buyers
Resources: User database, Internet, Computer/Laptop, Website running
Performance Location: Anywhere with internet, office of users
Precondition: EventHasOccurred (User requested login function) And (EventHasOccurred (Cancellation request validated) Or EventHasOccurred (Return request approved)) And DataReceived (Original method of payment) And ActionType (Refund)
Post-condition: EventProduced (Money refunded to original method of payment)

29. **Make Return Request**: If cancellation was unsuccessful or if the item was not as described, the buyer can make a request to return their item for a refund.
    Input:
    Invalid Cancellation [Function 15]
    Successful Shipment [External]
    User Request [External]
    Output:
    Request for return generated
    Performers: Businesses, Individual Buyers
    Resources: User database, Internet, Computer/Laptop, Website running
    Performance Location: Anywhere with internet, office of users
    Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (User requested to return product) And DataReceived (Item has successfully shipped) And ActionType (Create request)
    Post-condition: EventProduced (Return request generated)

30. **Create Return Label:** After the seller accepts the return request, the site generates a return label for the buyer who made the return request.
    Input:
    Successful Return Request [Function 29]
    User Request [External]
    Output:
    Return label generated

Performers: Businesses, Individual Buyers
Resources: User database, Internet, Computer/Laptop, Website running
Performance Location: Anywhere with internet, office of users
Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (Return request approved) And DataReceived (Seller's return address) And ActionType (Create label)
Post-condition: EventProduced (Return label generated)

31. **Pay for an Item:** After a buyer has bought an item, the buyer needs to pay for the item with their desired payment method.
    Input:
    Buy an Item [Function 3]
    Item to be paid for [External]
    Event 'User requested to pay for item' [External]
    Output:
    Update buyers payment history
    Performers: Businesses, Individual Buyers
    Resources: Items database, Internet, Computer/Laptop, Website running, Buyer Profile
    Performance Location: Anywhere with internet, office of users
    Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (Seller Listed this item) And EventHasOccurred (Buyer bought this item)and DataReceived (Payment amount) And ActionType (Pay for item)
    Post-condition: EventProduced (User has paid for item)

1.4.2. Scenario Specifications

1. **Sell an Item:**
   Sequence of Functions:
   Function 1: Authenticate and Authorize
   Function 2: List an Item
   Function 3: Generate shipping label.
   Function 3: Ship an Item (after it has been purchased and paid for)

Environment: This scenario occurs when a seller wants to list an item and have it sold. This includes the entire item lifecycle from initial listing to shipment to the buyer.
Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (User requested List an item function) And DataReceived (Item Description) And DataReceived (Item Price) And DataReceived (Item Condition) And DataReceived (Item Pictures) And EventHasOccurred(Buyer has bought the Item) And EventHasOccurred(Buyer has paid for item) And ActionType (Sell Item)
Post-condition: DataUpdated(Seller's items sold history) And DataUpdated(Seller store) And And EventHasOccurred(Item shipped)

2. **Purchase an item**
   Sequence of Functions:
   Function 1: Authenticate and Authorize
   Function 2: View an Item
   Function 3: Buy an Item
   Function 4: Pay for Item
   Function 5: View Shipping Status
   Function 6: Track Shipping Label
   Environment: This scenario occurs when a buyer has found an item that they want to purchase. The buyer then needs to pay for the item and will receive the item once it has been shipped by the seller.
   Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (User requested view an item function) And EventHasOccurred(Buyer has bought the Item) And EventHasOccurred(Buyer has paid for item) And ActionType (Purchase an item)
   Post-condition: DataUpdated(Buyer purchase history) And DataUpdated(Buyer's buyer profile) And EventHasOccurred(Item shipped)

3. **Search for item**
   Sequence of Functions:
   Function 1: Filter search results
   Function 2: Sort search results

Environment:
Execution Trigger: User clicks on search bar and enters keywords then presses enter
Completion Trigger: User finds the item they're looking for and clicks on it to bring up a more detailed view of the listing

4. **Edit an item listing**
   Sequence of Functions:
   Function 1: Edit listing description
   Function 2: Add/remove image
   Environment: There needs to be an item already listed that can be edited (item has not already been sold)
   Execution Trigger: Seller clicks 'Edit listing' button on their listing to open edit page
   Completion Trigger: Seller clicks 'Save listing' button on the edit page

5. **Create an account:**
   Function 1: Authenticate and Authorize
   Function 2: Authenticate email
   Environment: User must not be currently logged into an account
   Execution Trigger: Seller clicks 'Create Account'
   Completion Trigger: Seller clicks email validation link

6. **Edit Profile:**
   Function 1: Authenticate and Authorize
   Function 2: Edit Preferences
   Environment: User must currently be logged into valid account
   Execution Trigger: Seller clicks 'Edit Profile'
   Completion Trigger: Seller clicks 'Save Changes'

7. **Buy an item from watchlist and then decide to cancel**
   Sequence of Functions:
   Function 1: Authenticate and Authorize
   Function 2: View an Item (precondition - Authenticate and Authorize)
   Function 3: Add item to watchlist EventHasOccurred (precondition - User requested view an item function)

Function 4: Buy an Item (precondition - buyer has viewed item)
Function 5: Pay for Item (precondition - buyer has bought an item)
Function 6: View all orders (precondition - buyer is authenticated)
Function 7: Submit cancellation request (precondition - OrderNumber is valid)
Function 8: Issue Refund (precondition - cancellation request successful)
Environment: This scenario can occur when a buyer has found an item that they want to purchase, and adds it to their watchlist to purchase later if it hits a certain price. When it does, the buyer then purchases the item and pays for it. Before it has been shipped, the buyer decides they do not want the item anymore and submit a cancellation request. This scenario occurs on the buyer's laptop on the website.
Precondition: EventHasOccurred (User requested login function) And (User has account on website) And (User is interested in purchasing a product)
Post-condition: DataUpdated(Buyer purchase history) And DataUpdated(Buyer's buyer profile) And EventHasOccurred(Item cancelled) And EventHasOccurred(Issue Refund)

8. **Return an item after it has already been shipped**
   Sequence of Functions:
   Function 1: Authenticate and Authorize
   Function 2: View Order Details (precondition- user authenticated)
   Function 3: Submit cancellation request (precondition - OrderNumber is valid)
   Function 4: Contact Seller (precondition - user authenticated)
   Function 5: Request return (precondition - user authenticated)
   Function 6: Create return label (precondition - buyer has bought an item and requested return)
   Function 7: Issue refund (precondition - returned item received by seller)

   Environment: This scenario can occur when a buyer purchases an item from a seller, but then later decides that they do not want to keep the item for some reason. However, the item has already been shipped by the seller and cannot be cancelled, so the buyer

must go through the return process in order to get their money back.
Precondition: EventHasOccurred (User requested login function) And (User has account on website) And (User has bought an item) And (User wants to return item) And (Item has been shipped)
Post-condition: DataUpdated(Buyer purchase history) And DataUpdated(Buyer's buyer profile) And EventHasOccurred(Item returned) And EventHasOccurred(Issue Refund)

9. **Cancel an item after purchase**
   Sequence of Functions:
   Function 1: Authenticate and Authorize
   Function 2: View Order Details
   Function 3: Submit cancellation Request
   Environment: This scenario occurs when a buyer wants to cancel an item purchase while the item is in transit or before it's been packed and shipped by the seller.
   Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (User requested cancellation of order) And DataReceived (Item transit status) And EventHasOccurred(Buyer has purchased the Item) And Not EventHasOccurred(Buyer has received item) And ActionType (Send cancellation request)
   Post-condition: EventHasOccurred(Seller receives cancellation request)

10. **Generate personal suggestions**
    Sequence of Functions:
    Function 1: Authenticate and Authorize
    Function 2: Edit Preferences
    Function 3: Browser suggestions
    Environment: This scenario occurs when a buyer wants to allow the site to generate recommendations of sellers and items for them instead of searching for an item themselves.
    Precondition: EventHasOccurred (User requested login function) And EventHasOccurred (User requested suggestion generation) And DataReceived (User preferences) And ActionType (Update user preferences) And EventHasOccurred (User requested to browser generated suggestions) And DataReceived (Generated suggestion list)

Post-condition: EventHasOccurred(Buyer browses suggestions)

1.4.3 Essential Scenario

The essential scenario is **"Purchase an item"**. The entire point of an e-commerce system is to facilitate and oversee the exchange of money and goods between buyer and seller. As such, the essential scenario must either be "Sell an item" or "Purchase an item".

However, the primary responsibility of such a system should be to its users, which typically consists of far more buyers than sellers. Therefore, the most important scenario to consider for the system is the purchase of items. Without this scenario, the entire application fails to make sense.

1.5 Qualities and Constraints

1. Reliable

- **Description:** The service must be reliable and have fail safes in place to prevent financial loss to users.
- **Category:** Performance
- **Stakeholder source:** Software Testers
- **Scope:** Entire System
- **How the system should be evaluated:** Regression testing with each update

2. Scalable

- **Description:** The service must be able to handle a large influx of users.
- **Category:** Usability
- **Stakeholder source:** Consumers
- **Scope:** Entire System
- **How the system should be evaluated:** Simulated stress testing

3. Secure

- **Description:** System must securely store and transmit transaction data
- **Category:** Usability
- **Stakeholder source:** Software Testers, Engineers
- **Scope:** Transaction system
- **How the system should be evaluated:** Regression Testing with each update, vulnerability testing

4. Flexible

- **Description:** System must allow a wide variety of listing types, to capture wide user audience.
- **Category:** Usability
- **Stakeholder source:** Software Architect
- **Scope:** Listing new item
- **How the system should be evaluated:** System should be able to handle arbitrary items and pictures.

5. Accessible

- **Description:** Service should work with a wide variety of devices
- **Category:** Usability
- **Stakeholder source:** Business Analyst
- **Scope:** Device integration
- **How the system should be evaluated:** Accessibility testing with various devices

6. Fast

- **Description:** Service should be able to load new items in real-time while browsing.
- **Category:** Performance
- **Stakeholder source:** Software Engineers
- **Scope:** Database access
- **How the system should be evaluated:** Automated timed-testing

7. Consistent

- **Description:** Service should follow a consistent layout and theme.
- **Category:** Usability
- **Stakeholder source:** Business Analyst
- **Scope:** Website UI
- **How the system should be evaluated:** Meets Business Analyst's desire.

8. Clear Navigation

- **Description:** Users should be able to easily navigate to different parts of the website.
- **Category:**
- **Stakeholder source:** Consumers
- **Scope:** Website UI
- **How the system should be evaluated:** User feedback on ease of navigation.

9. Redundant

- **Description:** Data should be stored in more than one location for faster access, and reliability.
- **Category:** Performance
- **Stakeholder source:** Software Architect
- **Scope:** Database Storage and access
- **How the system should be evaluated:** Stress/Failure testing to simulate storage failures.

10. Optimized

- **Description:** Users should only see most relevant information in an efficient manner.
- **Category:** Performance
- **Stakeholder source:** Software Engineers
- **Scope:** Database access
- **How the system should be evaluated:** User feedback on relevancy of results.

1.6. Deployment Environment

When deploying this system, there are many considerations that must be taken into account for both producers and consumers.

First, in order to protect customer security, all application servers will be running on company premises. The backend server that does all processing will be deployed onsite on Linux servers and inaccessible from outside the company network. Also, the user database will be kept on premises and similarly isolated from the outside world. The front end servers are also deployed on premises, but they are accessible to the public via HTTP. Because the front end servers are in the company network, they are able to request backend processing for the user while ensuring that users cannot directly interact with company data and cause harm. This allows for a separation of critical functions from the user and provides the customers (businesses, individual sellers, and individual buyers) with as much security and privacy as possible.

Because of the potential scale of this application, we will need to make sure the system has enough throughput to process customers' actions. This means that both front end and back end servers will be deployed in a distributed fashion with smart routing and load balancing to

ensure maximum reliability. Additionally, the database will have sufficient replication to protect integrity. The aim of these deployment specifications is to maximize uptime of the application and minimize wait time for all customers (businesses, individual sellers, individual buyers).

Another deployment constraint is related to making development and maintenance as easily as possible. All backend and front end servers will be deployed in a Linux environment. This makes development easier for software developers since the environment is standardized and Linux is a development friendly operating system. Additionally, the standardization of deployment environment makes it easier for software testers to identify any issues with deployment and find bugs in the code.

Finally one last constraint for customer security will be to use a third party library/service like Stripe for payment processing. Customers are very concerned with security, and double so with important information like credit card information or PII. Using something like Stripe for payment allows us to outsource this security to another service that specializes in this domain. Because we do not have the resources and knowledge to process payments as securely as we want, it makes sense to utilize the services of a third party with specialization in payments. This will allow us to effectively cater to customers' demands for security of their payment information.