Functional Requirements

1. **Admin functionalities**

**Title:** User Removal by Admin

**Description:** The system should provide the admin with the ability to remove a user from the database, thereby revoking the user's access to the website, in response to receiving fraud or abusive reports about the user.

**Acceptance Criteria:**

**Actor:** Admin

**Action:** As an admin, I should have the capability to access and utilize the "RemoveUser" functionality to address and resolve issues such as user harassment or fraudulent activities, thereby ensuring the safety and security of other users on the platform.

**Title:** As an admin I should be able to view all the users.

**Description:** The system should offer comprehensive functionality to empower administrators with the capability to monitor and access detailed information about all users who have visited the website. This includes capturing and presenting essential data related to user visits, ensuring transparency and accountability in tracking user activity within the platform. The system should facilitate a user-friendly interface that allows administrators to retrieve, filter, and analyse this information, contributing to the platform's security and user management.

**Acceptance Criteria:**

**Actor:** Admin

**Action:** As an admin, I should be able have the access to view all the user information.

1. **Notification**

**Title:** User should be able to get notifications on any product information.

**Description:** The system should provide users with the ability to receive notifications whenever new product information is added to the platform. This feature aims to enhance user engagement, keep users informed about product updates, and enable them to stay up-to-date with the latest offerings.

**Acceptance Criteria:**

**Actor:** Users

**Action:** View the information in notification icon.

**Title:** As a user I should be able to get notified if the product is sold out.

**Description:** The system should provide users with the ability to receive notifications whenever a product which user have added to cart is sold out.

**Acceptance Criteria:**

**Actor:** Users

**Action:** View the information in notification icon mentioning the product is sold out.

**3)** **Cart System**

**Title:** A user should be able to add favourite products into cart.

**Description:** User should be able to add favourite products to cart.

**Acceptance Criteria:**

**Actor:** Users

**Action:** Add data into cart.

**Title:** As a user I should be able to remove products into cart.

**Description:** User should be able to remove favourite products from cart.

**Acceptance Criteria:**

**Actor:** Users

**Action:** Remove data from cart

**7) Item Listing**

**Title**: Listing the item that is to be sold

**Description**: The user should be able to add the details of the products that are to be sold. The details including price, photos, description of the item and quality of the product should be added.

**Acceptance** **criteria**:

**Actors**: Student users/sellers

**Action**: As a user, I should be able to view the details of the product along with the description and photos.

**8) Item availability**

**Title:** Adding availability status

**Description**: The user should be able to add the status of the product to show whether it’s available or sold.

**Acceptance** **criteria**:

**Actor**: User/Seller

**Action**: As a user, I should be able to view whether a particular item is still available or sold so that I can decide on my purchase.

**9) Item search**

**Title**: User should be able to search for an item

**Description**: The user should be able to search for the items that he wishes to buy using a keyword.

**Acceptance criteria:**

**Actor**: User/Buyer

**Action**: As a buyer, I should be able to search for the item that I wish to purchase and select from the available options.

Non-Functional Requirements

1. **Security**

**Title:** Should be able to give access to only to Purdue Fort Wayne students

**Description:** The system should have the capability to restrict access to only Purdue Fort Wayne (PFW) students, ensuring that only individuals with valid PFW affiliations can use the platform. This requirement is essential for maintaining the exclusivity of the platform to the intended user base and for safeguarding the integrity of the community.

**Acceptance Criteria:**

**Actor:** Admin, Users

**Action:** Provide access to only university students.

2) **Error handling**

**Title**: The system must be built with error handling mechanisms in place.

**Description**: The system should be able to handle unexpected inputs or errors and continue functioning without crashing.

**Acceptance Criteria:**

Proper error messages should be provided, and the system should be able to resolve the error quickly.

Functional Requirements (planned for upcoming sprint)

**1) Add data into the user table**

**Description:** User should be able to add the data to the table.

**Acceptance Criteria:**

**Actor:** Users

**Action:** Add data into user table.

**Method:** Save.

As a user I should be able to add user’s data to the table.

**URL:** http://localhost:8089/donHub/createUser

**Sample Request:** http://localhost:8089/donHub/createUser

{

"name":"Keerthika",

"emailId":"abc@pfw.edu",

"profile\_pic":”s”,

"PhoneNo":2601234567,

"Address": pfw

}

**Sample Response:** Successfully added the record.

1. **Update data in the user table**

**Description:** Users should be able to update the new information.

**Acceptance Criteria:**

**Actor:** Users

**Action:** Update the data in the table.

**Method:** Update.

As a user I should be able to update user data to the table.

**URL:** http://localhost:8089/donHub/emailId

**Sample Request:** http://localhost:8089/donHub/emailId

{

"name": "keerthika",

"PhoneNo":2601234567,

}

**Sample Response:** Successfully added the record.

1. **FilterByCondition**

**Description:** Users should be filter the products based on condition.

**Acceptance Criteria:**

**Actor:** Users

**Action:** Get the product data based on the condition.

**Method:** getByCondition.

As a user I should be able to get the date based on the condition.

**URL:** http://localhost:8089/donHub/new

**Sample Request:** http://localhost:8089/donHub/new

Sample Response:

{

"name":"table",

“description”:”dining table”,

“condition”:”new”,

“sellername”:”monkey”,

"price":20,

"Address": pfw

}