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.

1. **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

1. **User Authentication**

**Title:** Restrict Platform Access 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 using firebase. 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:** Users

**Action:** Authenticate users.

1. **Messaging System**

**Title:** A user should be able chat with the seller regarding the product. more details.

**Description:** The system should facilitate seamless communication between users and sellers by enabling users to initiate and engage in real-time chat conversations with sellers regarding the products listed on the platform. This chat functionality enhances user interaction, promotes trust, and supports informed purchasing decisions.

**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:** Should be able to chat with sellers

1. **Categories**

**Title:** Data Retrieval by User Based on Categories.

**Description:** The system should empower users with the capability to fetch specific data based on predefined categories, allowing them to access relevant information tailored to their preferences and needs. This functionality enhances the user experience by providing a more personalized and efficient way to access desired content.

**Acceptance Criteria:**

**Actor:** Users

**Action:** fetch data based on category.

**Title:** Need to add data based on Categories.

**Description:** user should be able to add data based on categories.

**Acceptance Criteria:**

**Actor:** Users

**Action:** add data into product database

**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.

**10. Sorting**

**Description:** Users should be able to sort item listings based on price and date.

**Acceptance Criteria:**

Users can sort items based on price and date.

The sorting option is available on the item listing page.

**Actor:** user(buyer)

As a user, sort the items by the price.

As a user, sort the items by the date.

**Action:** Implement a UI control (e.g., dropdown or button) that allows users to select the desired sorting order and apply the sort to the displayed item list accordingly.

**11. User Registration**

**Description:** Enable users to create an account and join the platform.

**Acceptance Criteria:**

Users can access a registration page with fields for username, email, and password.

Upon submission, the system securely stores user registration data in the database.

**Actor:** User (New User)

**Action:** As a new user, fill out the required registration form and submits it to get registered.

**Method:** Design and implement a registration page where users can input their desired username, email, and password. Upon submission, the system validates the data, securely stores it, and sends a verification email to the provided email address.

**12. Payments**

**Description:** Facilitate secure and efficient payment processing for item transactions.

**Acceptance Criteria:**

Users should be able to securely enter their payment information for purchases.

Payment processing should be reliable, ensuring successful transactions.

**Actor:** User (Buyer)

**Action:**As a user, select an item for purchase and proceeds to the checkout to enter payment information and pay successfully.

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.

1. **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.

1. **User friendly and Accessibility**

**Description:** Ensure that the platform is accessible to all users.

**Acceptance Criteria:**

The platform should comply with relevant accessibility standards (e.g., WCAG).

Users should be able to use the platform conveniently.

**Actor:** User (Any User)

**Action:** The user accesses the platform using assistive technologies.

**4**. **Compatibility and Logging**

**Title:** Should be able to open the donHub website in any browser.

**Description:** The DonHub website should be designed and developed to ensure compatibility with all major web browsers. This compatibility ensures that users can access and use the website seamlessly, regardless of the browser they choose, enhancing the user experience and expanding the reach of the platform.

**Acceptance Criteria:**

**Actor:** Admin, Users

**Action:** Able to open the website in any web browser.

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

}

1. **DeleteByCondition**

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

**Acceptance Criteria:**

**Actor:** Users

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

**Method:** DeleteByCondition.

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

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

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

1. **FilterByCategory**

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

**Acceptance Criteria:**

**Actor:** Users

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

**Method:** FilterByCategory.

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

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

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

1. **Add data into the product table**

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

**Acceptance Criteria:**

**Actor:** Users, Admin

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

**Method:** Save.

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

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

**Sample Request:** http://localhost:8089/createProduct

{

"name":"Tanuja",

"description":"hey",

"sellerName\_pic":”s”,

"Price":20,

"image": pfw

}

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

1. **Update data into the product table**

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

**Acceptance Criteria:**

**Actor:** Users

**Action:** Update data into product table.

**Method:** Update.

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

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

**Sample Request:** http://localhost:8089/product

{

"name":"Tanuja",

"description":"hey",

"sellerName\_pic":”s”,

"Price":20,

"image": pfw

}

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

1. **FilterByPrice**

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

**Acceptance Criteria:**

**Actor:** Users

**Action:** Get the product data based on the price or get the products which are less than or equal to the price.

**Method:** getByPrice.

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

**URL:** http://localhost:8089/donHub/price/56

**Sample Request:** http://localhost:8089/donHub/price/56

**Sample Response:**

{

"name":"table",

“description”:”dining table”,

“condition”:”new”,

“sellername”:”monkey”,

"price":20,

"Address": pfw

}