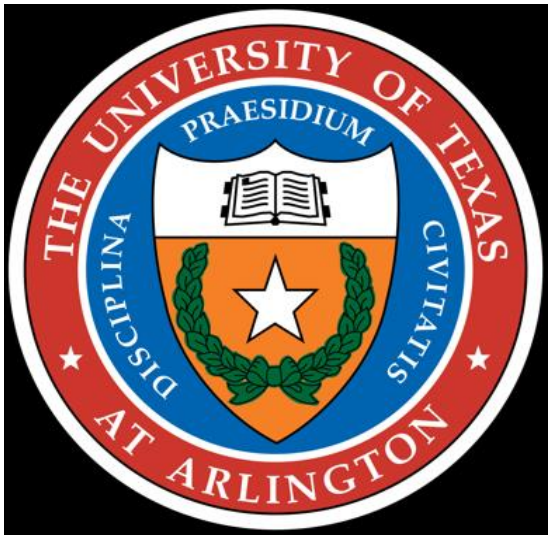


ReHome

The International Student Used Item Marketplace

CSE 6324: ADV TOPS SOFTWARE ENGINEERING

Group 28

Members name and ID:

Sanmesh Amol Sankhe - 1001924953
Archi Sahu - 1002104548
Rakesh Kasula – 1002060973

Date: 14th November 2023

Table of Contents

Serial Number	Contents
1	Abstract
2	Introduction
3	Project Scope
4	Requirements Analysis
5	Project Planning
6	Design And Architecture
7	Implementation
8	Testing
9	Maintenance
10	Conclusion
11	Recommendations
12	References

Abstract:

Summary:

The "ReHome" project seeks to create an easily navigable internet platform that would enable foreign university students to exchange old goods with ease. This platform encourages eco-friendly living, minimizes waste, and encourages the reuse of items. With a focus on fostering community, it links students all around the world and offers a convenient marketplace for trading, gifting, and purchasing goods. The platform uses strong security controls and features that foster trust to guarantee secure transactions and interactions. International students can improve their educational experience, make new friends, and save money by using ReHome. It also helps to create a more sustainable society.

Objective:

ReHome aims to facilitate resource sharing among students, thereby reducing waste and encouraging eco-friendly practices. The platform also aims to reduce financial pressures for international students by offering affordable solutions, fostering a strong sense of community through interactions and shared experiences, and providing excellent customer support to ensure a seamless and secure marketplace for all users.

Key Outcomes and Contributions:

1. **Resource Sharing:** ReHome encourages students to sell or give away their used goods, cutting down on waste and encouraging the reuse of materials. This makes a major contribution to the sustainability of the environment.
2. **Cost Reduction:** ReHome helps international students save money while studying abroad by providing them with affordable substitutes for new product purchases.
3. **Community Building:** By providing a safe space for interaction and mutual support, the platform helps international students develop a sense of unity.

Achievements:

- Effective deployment of interfaces that are easy to use by administrators and students, improving user satisfaction overall.
- Encouragement of trash reduction and item reuse as a means of promoting sustainability and eco-friendly living.
- Establishing a thriving community of foreign students and helping students from different backgrounds make friends and relationships.

Conclusions:

In conclusion, by creating a safe, practical, and effective platform for overseas college students, the ReHome project has effectively met its goals. ReHome has significantly improved the general well-being of international students as well as the environment by encouraging item reuse, cutting waste, and building a feeling of community. ReHome is an important resource for students looking for inexpensive and sustainable alternatives throughout their time in college because of the implementation of strong security measures and attentive user assistance, which have guaranteed a positive and secure user experience.

Introduction

Overview:

The goal of the project, "ReHome," is to create an easily navigable online platform that is specifically designed to meet the demands of international university students. The website makes it simple to list, search for, and buy used products while placing an emphasis on sustainability by encouraging item reuse and reducing waste. In addition to helping students locate reasonably priced goods, ReHome creates a lively international student community that promotes friendships and connections.

Objective:

ReHome's objective is to make it easier for students to share resources, which will cut down on waste and promote environmentally friendly behavior. By providing inexpensive options, building a strong sense of community through interactions and shared experiences, and offering first-rate user support to guarantee a seamless and secure marketplace for all users, the platform also seeks to lessen financial burdens for international students.

Context and Problem Statement:

The ReHome project was started with the difficulties that overseas students at universities experience in mind. These students frequently struggle with money problems, particularly when it comes to buying necessities for their everyday lives and academics. Furthermore, there are serious worries about how consumerism affects the environment and how used goods are disposed of.

The primary issue addressed in the problem statement is the absence of an international student-focused, readily navigable platform where they can quickly locate and trade secondhand goods while promoting a feeling of community. ReHome fills this void by offering a safe, effective, and environmentally friendly option that benefits both the environment and international students.

Project Scope

Scope of the Project:

The project entails creating the ReHome platform to meet the needs of students and administrators, who are the two main user groups. For effective product uploads, seller interactions, and product searches, students will make use of customized dashboards. Conversely, administrators will be able to access product data and manage requests as well as reports.

Features and Functionalities:

1. **User Management:**
    - User registration and profile management.
    - Password reset functionality for users.
    - Secure login and logout features for both students and administrators.
2. **Product Management:**
    - Users can upload detailed information about the items they want to sell.
    - Administrators review and approve/decline product listings.
    - Users can search, browse, and view products available on the platform.
    - Users can mark products as favorites for future reference.
3. **Communication Features:**
    - Users can contact the administrator for assistance.
    - Users can communicate directly with sellers via email within the platform.
    - Administrators can respond to user inquiries efficiently.
4. **Transaction Management:**
    - Users can view the list of items they have sold, ensuring transparency.
    - Administrators can access reports detailing the number of products sold, gaining insights into platform performance.
5. **Community Building:**
    - The platform fosters connections and friendships among international students.
    - Users can engage in direct communication with sellers, enhancing the sense of community.
6. **Security Measures:**
    - Robust security measures ensure a secure environment for all users.

- Administrators have control over the approval of product listings, ensuring the quality and appropriateness of items.

Constraints:

- The platform is limited to international university students.
- Products listed must be used items, aligning with the platform's sustainability goals.
- Administrators review and approve product listings, ensuring the quality and appropriateness of items on the platform.

Target Audience:

International university students looking to purchase, sell, or donate used goods make up the main target demographic. The platform administrators who oversee its operations make up the secondary audience.

Limitations:

- Access to the platform is restricted to registered international university students, excluding the public.
- Communication between users and sellers is limited to email within the platform, excluding other forms of interaction.
- Product listings are subject to approval by administrators, potentially causing minor delays in items appearing on the platform.

Requirements Analysis

Requirements Gathering Process:

To achieve the goals specified for the ReHome project, a thorough requirements gathering procedure will be conducted. To guarantee a thorough grasp of customer needs and preferences, this method will incorporate several tools, including surveys, interviews, and documentation. The steps involved in gathering requirements will go as follows:

1. **Surveys:**

    - Create and send questionnaires to overseas students to get quantitative information about their expectations, challenges, and preferences with relation to purchasing and selling used goods.
    - To learn more about the user's perspective, include questions about desired functionality, security concerns, and preferred user interfaces.
    - Examine survey data to find recurring themes and inclinations within the intended audience.

2. **Interviews:**

    - To acquire qualitative perspectives, conduct one-on-one interviews with a varied group of overseas students.
    - Inquire open-endedly about their experiences using the current online marketplaces, the difficulties they encountered, and the characteristics they believe are crucial.
    - Invite users to contribute their thoughts on sustainable practices and community development on the site.
    - To extract useful user stories and scenarios, record and examine interview answers.

3. **Documentation:**

    - Examine the literature that is currently available on user behaviour research, online markets, and sustainable living strategies.
    - Based on industry standards and best practices, document technical requirements for things like data storage, scalability, and security protocols.

4. **Stakeholder Meetings:**

    - Call meetings to address expectations and concerns from stakeholders, such as students, administrators, and technical specialists.
    - Work closely with the client and project team to make sure the platform's features complement the goals of the project.
    - Get input from stakeholders on the initial concept and iterate the needs accordingly.

Functional Requirements:

1. **User Registration and Authentication:**
   - Users can create and manage their profiles.
   - Users can reset their password through the "Forgot Password" functionality.
   - Administrators can create and manage their admin profiles.

2. **User Dashboard:**
   - Upon authentication, users access their personalized dashboards.
   - Administrators access their designated admin dashboards.

3. **Product Management:**
   - Users can upload detailed information about products they intend to sell.
   - Administrators review and approve/decline products before they are published.

4. **Product Browsing and Purchasing:**
   - Users can browse and view all products available.
   - Users can mark products as favorites.
   - Users can purchase products listed within the application.
   - Users can view their favorite products.
   - Users can engage in direct communication with sellers through email.

5. **Transaction History:**
   - Users can view the list of products they have sold.

6. **Communication Features:**
   - Users can contact the administrator using the "Contact Us" feature for inquiries and assistance.
   - Administrators can view and respond to user-generated questions and concerns.
   - Administrators can send automated email responses to user inquiries.

7. **User Interaction:**
   - Users can mark products as favorites.
   - Users can log out, terminating their active session.
   - Administrators can log out, concluding their active administrative session.

Non-Functional Requirements:

1. **Usability:**
   - The user interface should be intuitive and user-friendly, catering to users with varying levels of technical expertise.
   - The platform should provide clear instructions and feedback to users during the registration, product listing, and purchasing processes.

2. **Scalability:**
   - The system should be able to handle a growing user base and an increasing number of product listings without a significant decrease in performance.

3. **Compatibility:**
    - The platform should be accessible and functional across various devices and web browsers commonly used by the target audience, ensuring a consistent experience for all users.

Use Cases or User Stories:

- Create a registration page for the user.
- Create a login/logout page.
- Create a Home page.
- Create edit profile page.
- Create a Forgot password page.
- Create a page to upload details for the product.
- Create Contact us page.
- Create a page to display all the products.
- Add functionality to button to send email to the seller.
- Create a page to show all the products which are sold till now.
- Create a page to filter products.
- Create a page to show favorite products.
- Create a page to display all the products they wish to sell, with statuses indicating approval or pending status.
- Create a page to display the product request.
- Create a report to show the number of products sold till now.
- Create a page to show messages sent through the contact us page.
- On click of button automated email send to user
- Testing
- QA testing and bug fix
- Documenting Report

Changes or Updates to Requirements:

1. Added Deleted Functionality for User to Delete the Item Posted:

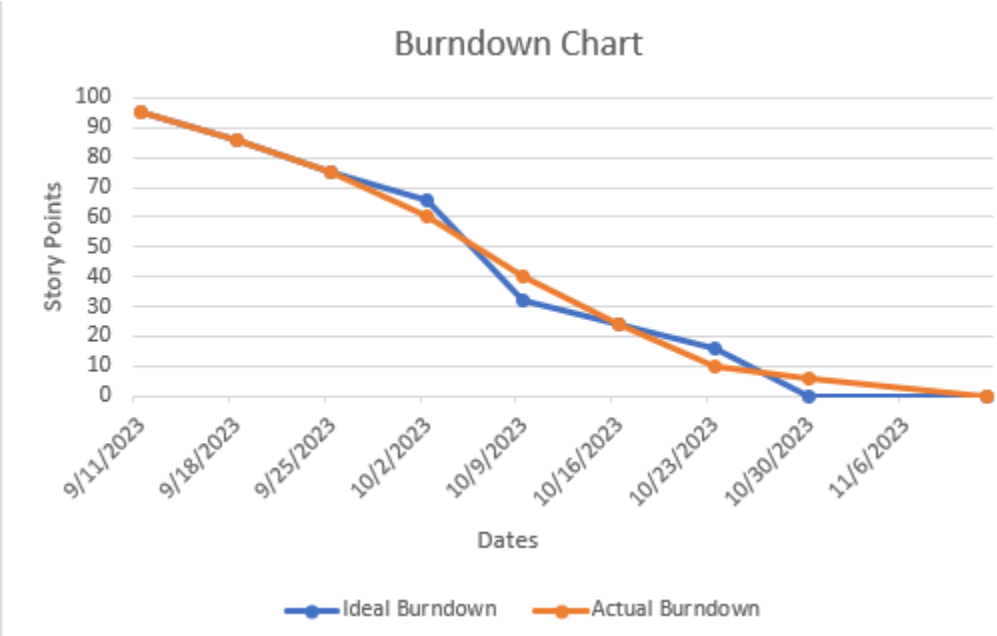
A new feature that enables users to remove goods they have posted on the ReHome platform has been implemented. With additional control over their listed items thanks to this functionality, users can effectively manage their inventory. The deletion feature makes sure that only the owner of the item can remove a listing, in accordance with security best practices.

2. Added Status (Pending or Available) in User Dashboard for the Item They Posted:

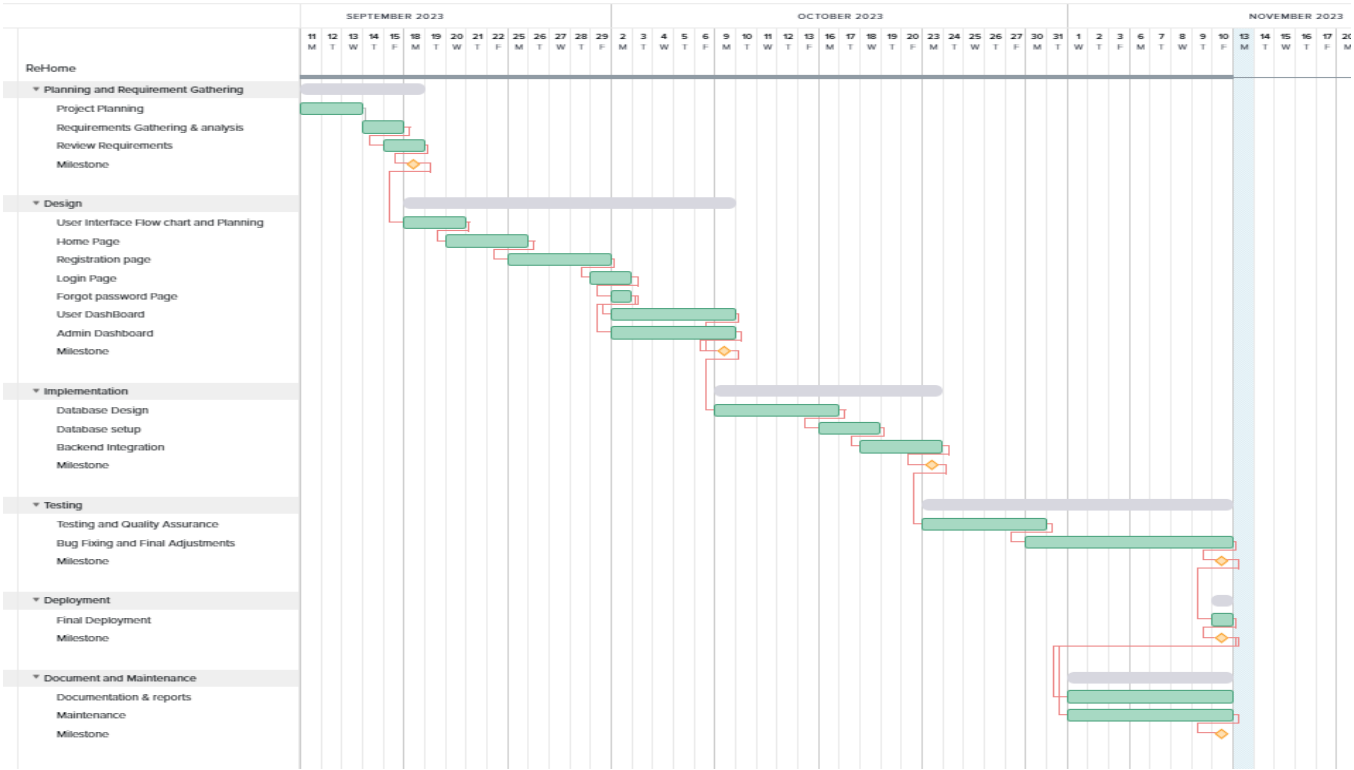
For every item posted, the ReHome platform now shows a status indicator in the user dashboard, improving transparency and streamlining user interactions. This status lets you know if the item is available for exchange or is presently awaiting approval. By giving users real-time access to the status of the items they have posted, this innovation clears up any confusion and enhances the trading experience overall.

Project Planning

Burndown Chart:



Gantt Chart:



Reasons for behind or ahead of schedule:

Week 1 (9/11/2023- 9/18/2023**):**

- *Planned Tasks:* Requirement gathering, Requirement analysis, Requirement review.
- *Actual Progress:* Successfully completed the planned tasks in Week 1. Thorough requirement gathering, analysis, and review laid the foundation for the project, ensuring a clear understanding of project goals and user needs.

Week 2 (9/18/2023- 9/25/2023**)**

- *Planned Tasks:* Document/Reports, Home page.
- *Actual Progress:* Completed the documentation and reports as scheduled in Week 2. Additionally, the Home page development commenced, aligning with the project roadmap.

Week 3 (9/25/2023- 10/2/2023)

- *Planned Tasks:* Signup, Sign in, Forget Password functionality.
- *Actual Progress:* Completed the planned tasks ahead of schedule, allowing an early start on User and Admin dashboards in Week 3.

Week 4 (10/2/2023 -10/9/2023)

- *Planned Tasks:* User and Admin dashboards.
- *Actual Progress:* Deviation from the schedule due to the extensive nature of the User and Admin dashboards, which required additional time. These tasks spilled over into Week 5.

Week 5: (10/9/2023 -10/16/2023)

- *Planned Tasks:* Completion of User and Admin dashboards, Email functionality for communication.
- *Actual Progress:* Completed the pending User and Admin dashboard tasks as well as the email functionality for communication. The extra time invested in Week 5 paid off in catching up with the delayed tasks from Week 4.

Week 6: (10/16/2023 -10/23/2023)

- *Planned Tasks:* Database setup and integration.
- *Actual Progress:* Successfully completed the database setup. And started testing.

Week 7: (10/23/2023 -10/30/2023)

- *Planned Tasks:* Database integration for Admin dashboard, Unit testing, QA testing, Bug fixing, and Documentation.
- *Actual Progress:* Completed the integration for the admin dashboard, resolving the pending task from Week 6. Concurrently, we initiated Unit testing, QA testing, Bug fixing, and Documentation.

Week 8: (10/30/2023 -11/12/2023)

- *Planned Tasks:* Unit testing, QA testing, Bug fixing, and Documentation.
- *Actual Progress:* Successfully completed all remaining tasks, ensuring a comprehensive testing phase, bug resolution, and detailed documentation.

Each person tasks and responsibilities:

Resource Name:	Task and Responsibilities
Archi	Create a registration page for the user
Sanmesh	Create a login/logout page
Sanmesh	Create a Home page
Rakesh	Create edit profile page
Sanmesh	Create a Forgot password page
Archi	Create page to upload details for the product
Archi	Create Contact us page
Rakesh	Create page to display all the products
Archi	Add functionality to button to send email to the seller
Archi	Create a page to show all the product which is sold till now
Archi	Create a page to filter products
Sanmesh	Create a page to show favorite products
Sanmesh	Create page to display all the products they wish to sell
Rakesh	Create page to display the product request
Sanmesh	Create report to show the number of products sold till now
Rakesh	Create a page to show messages sent through contact us page
Rakesh	On click of button automated email send to user
Archi/Rakesh/Sanmesh	Testing
Rakesh/Archi/Sanmesh	QA testing and bug fix
Rakesh/Archi/Sanmesh	Documenting Report

Design and Architecture

Software Architecture:

The software architecture for the "ReHome" project can be designed using a combination of front-end and back-end technologies. Below is the description of the software architecture components:

Front-end Components:

- The front-end will include a user-friendly web interface accessible via browsers and mobile devices.
- The UI will be designed using HTML, CSS and React.js to create interactive and responsive user experiences.

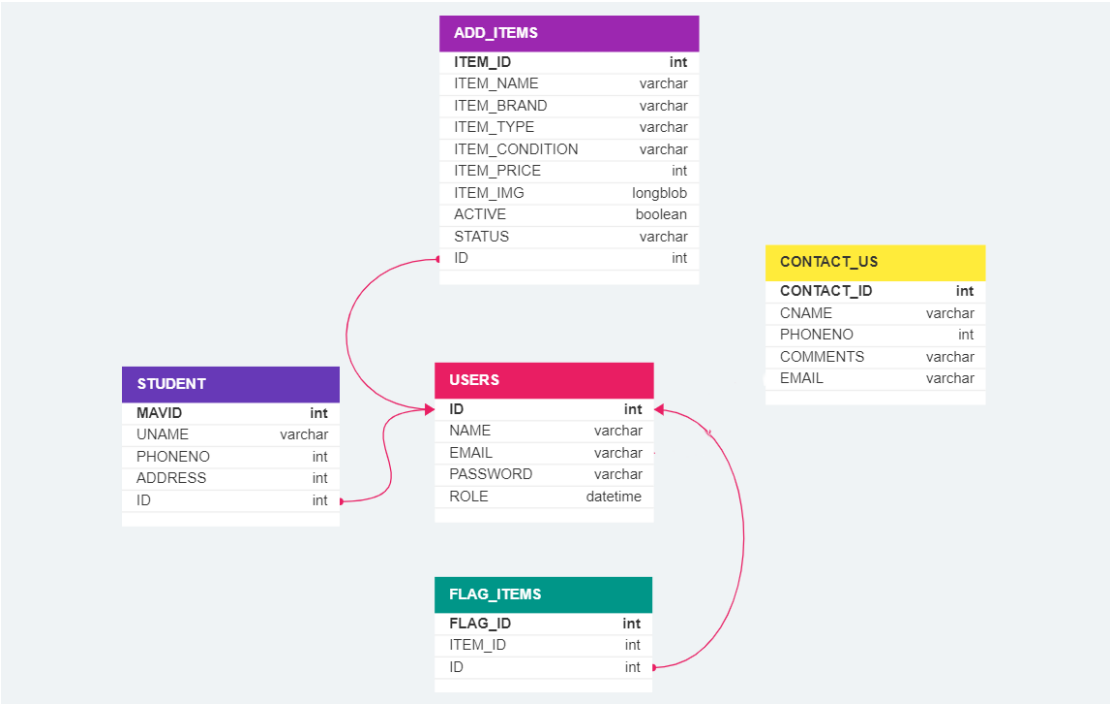
Back-end Components:

- PHP is used to implement business logic to handle user actions such as adding products, flagging items, managing user profiles, and processing transactions.

Database:

- MySql is used to implement the database schema.
- Ensure data normalization, and proper relationships between tables for efficient data retrieval and storage.

UML Diagram:



Database Schema:

1. **Users Table:**
    - ID (Primary Key)
    - Name
    - Email
    - Password
    - Roles

2. **Student Table:**
    - MavID (Primary Key)
    - Name
    - Phone Number
    - Address
    - ID (Foreign Key)

3. **Add Item Table:**
    - Item_ID (Primary Key)
    - Item_Name
    - Item_Brand
    - Item_Type
    - Item_Condition
    - Item_Price
    - Item_Img
    - Active
    - Status
    - ID (Foreign Key)

4. **Flag Item Table:**
    - FlagID (Primary Key)
    - Item_ID (Foreign Key)
    - ID (Foreign Key)

5. **Contact Us Table:**
    - ContactID (Primary Key)
    - Name
    - Phone Number
    - Comments
    - Email

Database Relationship:

1. Users Table and Student Table: **One-to-One relationship** between the Users table and the student table. Each user in the Users table can have at most one corresponding student record in the student table.
2. Users Table and Add Item Table**: One-to-Many relationship** between the Users table and the Add Item table. One user in the Users table can be associated with multiple items added in the Add Item table. However, each item is linked to a single user.
3. Add Item Table and Flag Item Table: **One-to-Many relationship** between the Add Item table and the Flag Item table. One item in the Add Item table can have multiple flags in the Flag Item table, indicating that it has been flagged multiple times.
4. Users Table and Flag Item Table: **One-to-Many relationship** between the Users table and the Flag Item table. One user can flag multiple items, and each flag is associated with a single user.

Implementation

Development Environment and Tools Used:

1. **IDE**: Visual Studio Code is the integrated development environment (IDE) used for coding and managing the project files. It provides a user-friendly interface and extensive extensions support for various programming languages and frameworks.

2. **Database**: MySQL is the chosen relational database management system (RDBMS) for storing and managing the project's data. It is known for its reliability, performance, and ease of use.

3. **Technology**: PHP is used as the server-side scripting language. It enables dynamic web page generation and interacts with the MySQL database to handle data operations.

4. **Scripting Language**: React.js is the JavaScript library used for building the user interface components. React.js facilitates the creation of interactive and responsive web applications, ensuring a seamless user experience.

5. **Frontend**: HTML and CSS are employed for structuring the web pages and styling the user interface, respectively. HTML provides the basic structure, while CSS enhances the visual presentation of the application.

6. **Libraries**:

- PHP Mailer: PHP Mailer is utilized for sending emails from the PHP backend. It simplifies the email-sending process and enhances email delivery reliability.
- ReCharts: ReCharts is a popular charting library for React.js, enabling the visualization of data through various types of charts, such as line charts, bar charts, and pie charts.

Major Components or Modules:

1. **Product Management:**

   Allowing users to add, edit, and mark as sold to any product listings, including details like item name, price, condition, and images.

2. **Communication Module:**

- Enables users to communicate directly with each other, allowing inquiries about products listed on the platform.
- Enabling communication between users and administrators, handling inquiries, and providing support features.

3. **Flagging System:**
   Implementing a system where users can flag item for future references.

4. **Reporting:**
   Generating reports based on user data and product sales, providing insights into the platform's performance.

Challenges Faced During Implementation:

- Integrating various technologies like PHP, MySQL, React.js, and third-party libraries can be challenging due to differences in syntax, data handling, and asynchronous operations.

- Implementing the email and report functionalities involves exploring unfamiliar libraries for PHP, requiring in-depth research to successfully integrate these features into the project. This process demands thorough investigation and learning to effectively incorporate email communication and reporting capabilities within the PHP-based application.

Third-Party Libraries or Frameworks Used:

- PHP Mailer: Used for sending emails, providing a reliable and efficient way to handle email notifications and responses.
- ReCharts: Utilized for data visualization, enabling the creation of interactive charts and graphs to display relevant statistics and trends to users.

Testing

Testing Strategy:

Testing is a crucial phase in software development, ensuring that the application functions correctly, meets user requirements, and provides a seamless user experience.

Unit Testing:

Test Case	Description	Status	Expected Result	Actual result
1	Home Page Navigation	Pass	Navigate to aboutus, Contactus, service, Register and login page.	Navigate to all pages properly
2	Registration page: Input Fields and its validation	Pass	Input fields should be editable, and validation must be there.	All input fields are editable and Validation for all fields
3	Login Page: Input Fields and its validation	Pass	Input fields should be editable, and validation must be there.	All input fields are editable and Validation for all fields
4	Forgot Password: Input Fields and its validation	Pass	Input fields should be editable, and validation must be there.	All input fields are editable and Validation for email field
5	User dashboard: Navigation	Pass	Navigate to EditProfile, Bookmarked item, Search items, add items and Sold item page	Navigate to all pages properly
6	User dashboard: Add Item form, Edit profile	Pass	Input fields should be editable	All input fields are editable
7	Admin dashboard: Navigation	Pass	Navigate to Contactus question and approve item	Navigate to all pages properly

Integration Testing:

Test Case	Description	Status	Expected Result	Actual result
1	Registration page: Register button	Pass	On click of submit button user data is stored in DB.	User data is stored in DB on click of submit button
2	Login Page: Login button	Pass	On click of Login button user is redirect to respective dashboard	User can successfully login to respective user dashboard.

3	Forgot Password: Generate New Password Button	Pass	On click of button user should get an email with new password	User is getting new password in the email
4	User dashboard: Edit Profile Page	Pass	On click of submit button user data are modified in DB	User data is successfully updated in DB
5	User dashboard: Add Item Page	Pass	On click of submit button product send to admin for approval	Item send successfully to admin for approval
6	User dashboard: Sold Item Page	Pass	On click of sold item button user should be able to see list of items they sold till now	Successfully visible List of sold items
7	User dashboard: Bookmarked Item button	Pass	On click of button user should be able to see list of items they marked as favorite	Successfully visible List of favorite items
8	User dashboard: Search Item button	Pass	On click of button user should be able to search items.	User can search item and Successfully visible List of items
9	User dashboard: Inside Search Item page user have option for Bookmarked	Pass	Users have option to mark item as favorite	Users can mark items as favorite.
10	User dashboard: Inside Search Item page user have option for Purchase Request	Pass	On click of button email is sent to respective seller that user what to buy it	Automatic mail is sent to the seller
11	User dashboard: Personal information	Pass	Able to see profile details on dashboard	Successfully see profile details on dashboard
12	User dashboard: Edit listed items	Pass	User should be able to edit the fields and on click of save button it should save in DB	Successfully stored updated item details in DB
13	User dashboard: View details of the item	Pass	On click of the detail button item details should be visible	Item details is visible successfully
14	Admin dashboard: Report	Pass	Graph is visible to Admin to show how many items sold	Graph is successfully visible
15	Admin dashboard: Contact Us Question	Pass	View the question that user sent	Admin can view all the question that user sent
16	Admin dashboard: Inside Contact Us Question Page admin have option to Send mail to user	Pass	On click of mail button admin send mail to user	Successfully sent mail to user

17	Admin dashboard: Approve item Page	Pass	Admin has option to approve and reject the item which user posted	Successfully approve/decline the posted item

System Testing:

Test Case	Description	Status	Expected Result	Actual result
1	End to End User Registration	Pass	Users should be able to fill in all the details and register on the website. After registration successful message should be displayed.	Successfully register message is displayed. Once user click on register button with all the details
2	End to End Login	Pass	After entering username password user should be able to login and respective dashboard should be open.	After validating username password user can login to respective dashboard
3	End to End User Dashboard	Pass	User should be able to edit their profile, add item, View sold item, Search Item, bookmark any item and view bookmarked item	User successfully able to edit their profile, add item, View sold item, Search Item, bookmark any item and view bookmarked item
4	End to End Admin Dashboard	Pass	Admin should be able to view report, approve items and send mail to users' question	Admin successfully able to view report, approve items and send mail to users' question

User Acceptance Testing:

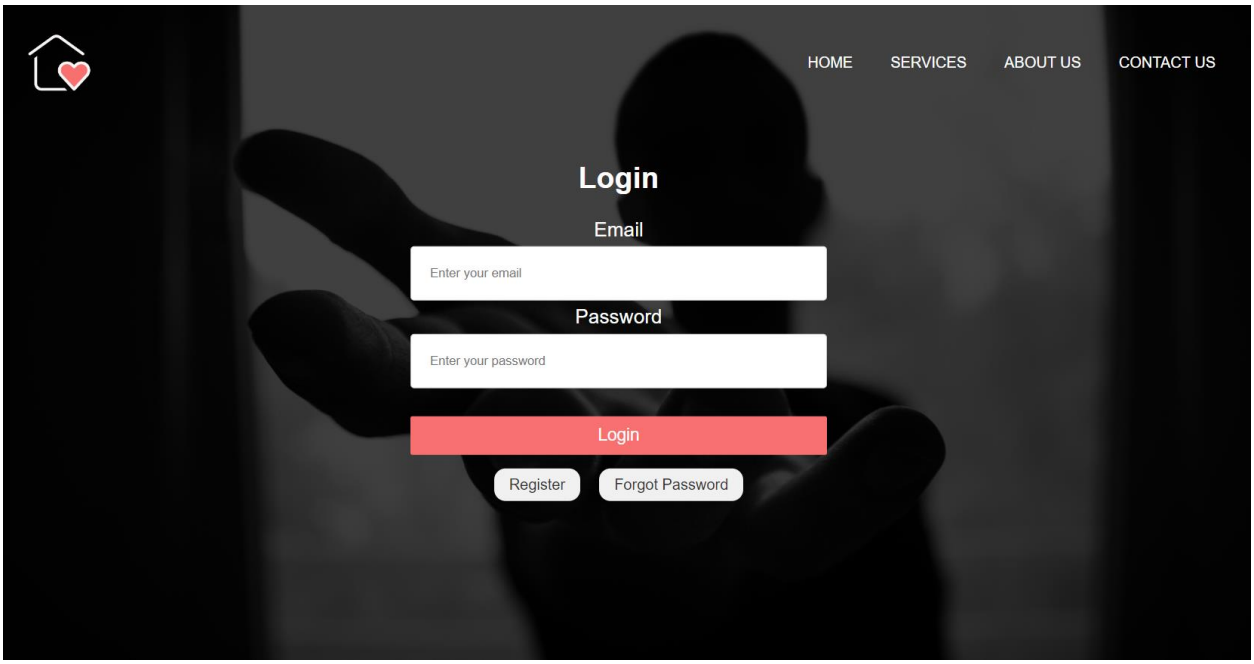
Test Case	Description	Status	Expected Result	Actual result
1	Register with already existing userid	Pass	Proper error message should be displayed	Error message is displayed
2	Login with incorrect credentials	Pass	Proper error message should be displayed	Error message is displayed
3	Login with valid credentials	Pass	Respective dashboard should be open	Successfully open respective dashboard
4	Register new user	Pass	Successfully message should be displayed	Successfully message displayed
5	User should be able to generate new password using forget password	Pass	Upon entering their email address, a new password will be generated and sent to the provided email. Users can then use this new password to log into their account.	Successfully generated new password.

6	After Login they should be able to logout	Pass	After clicking the logout button, users should no longer have access to their profile. They should be unable to return to their profile by using the back button on their browser.	Successfully logout from the dashboard.
7	Users should have the capability to execute all actions available in the user dashboard.	Pass	Users should be able to seamlessly utilize all the options available in the user dashboard. Upon selecting any feature, the corresponding message or feedback should be clearly displayed, ensuring a user-friendly and informative experience.	Users can seamlessly utilize all the options available in the user dashboard.
8	Admin should have the capability to execute all actions available in the admin dashboard.	Pass	Admin should be able to seamlessly utilize all the options available in the admin dashboard. Upon selecting any feature, the corresponding message or feedback should be clearly displayed, ensuring a user-friendly and informative experience.	Admin can seamlessly utilize all the options available in the admin dashboard.

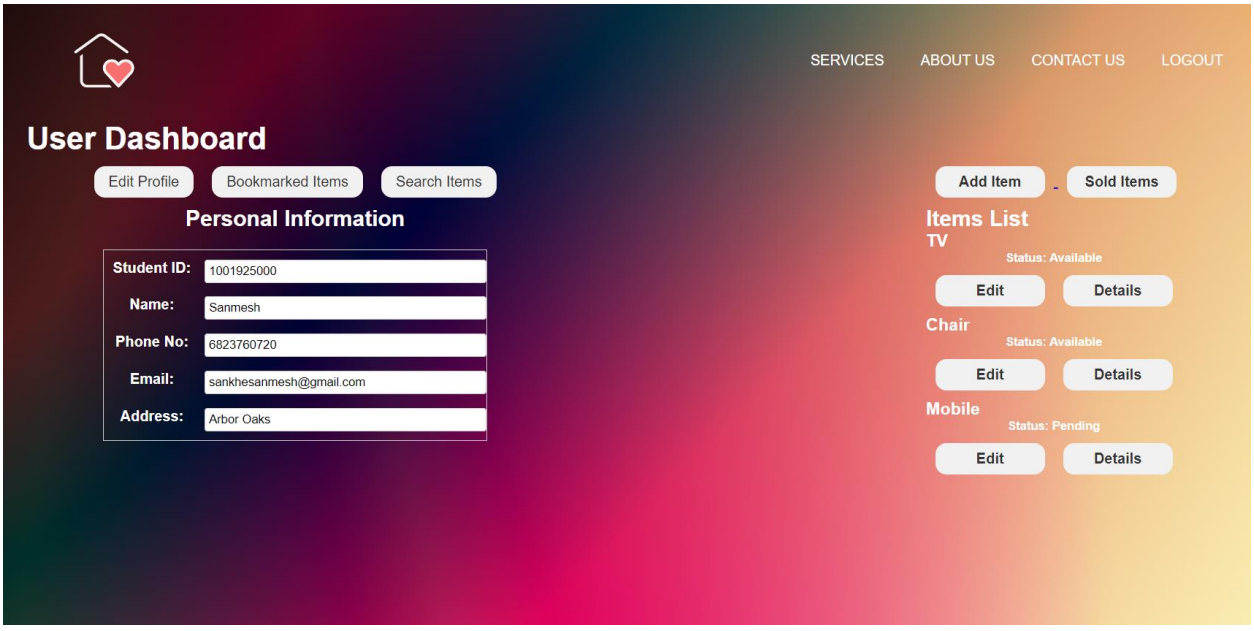
Provide test cases and results. Both manual and automation results should be included:

Manual Testing:

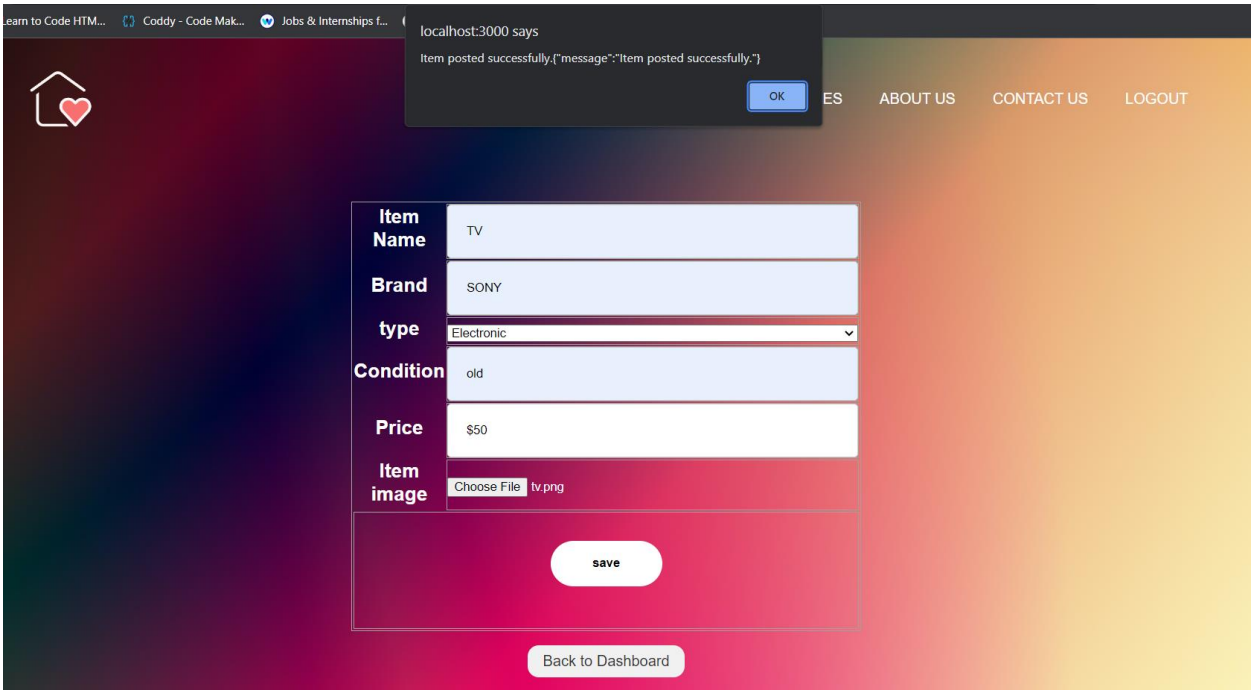
Login:



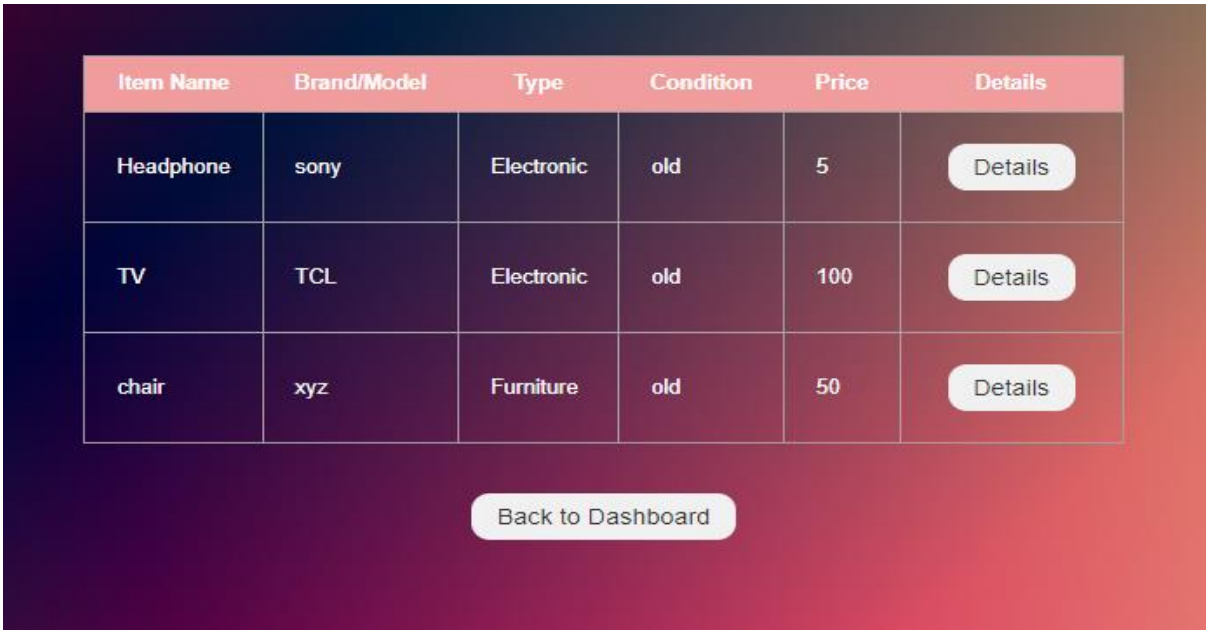
User Dashboard:



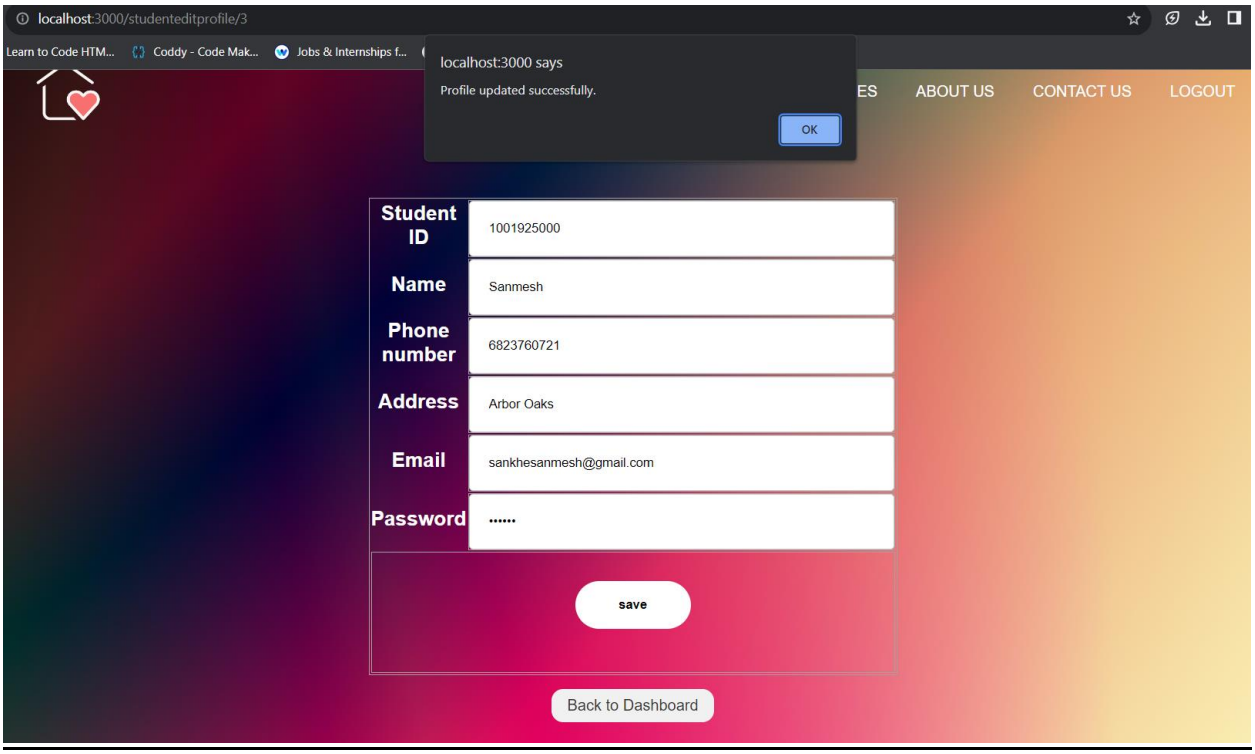
Add Item:



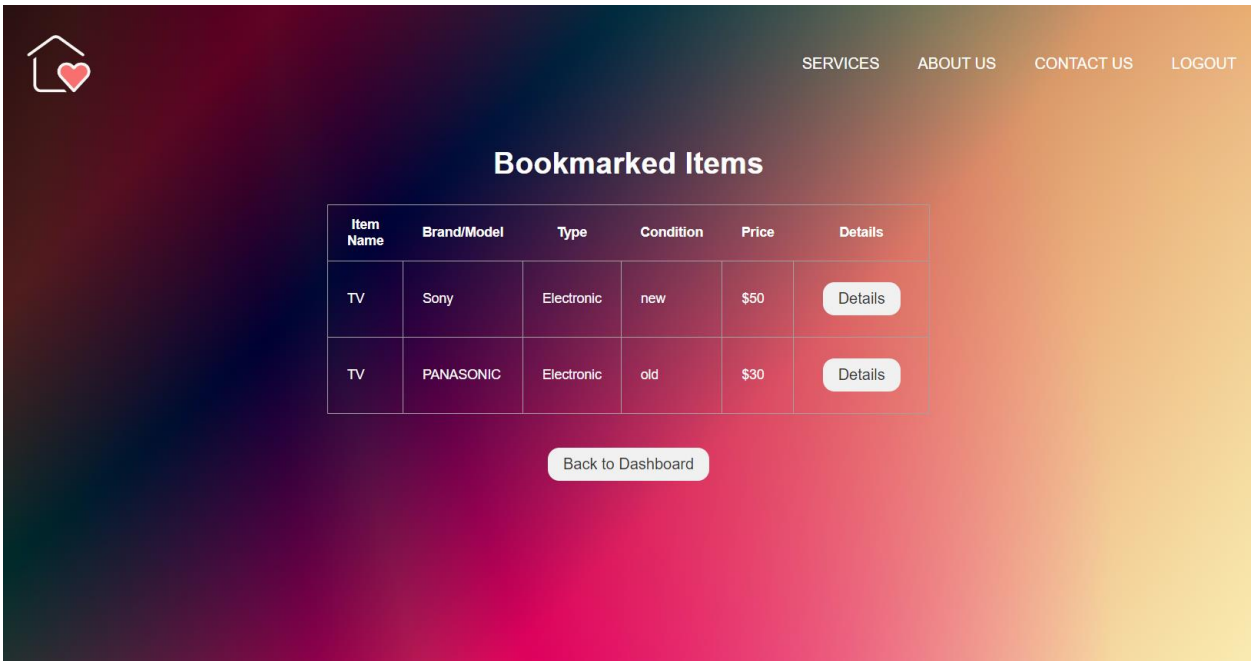
SoldItem:



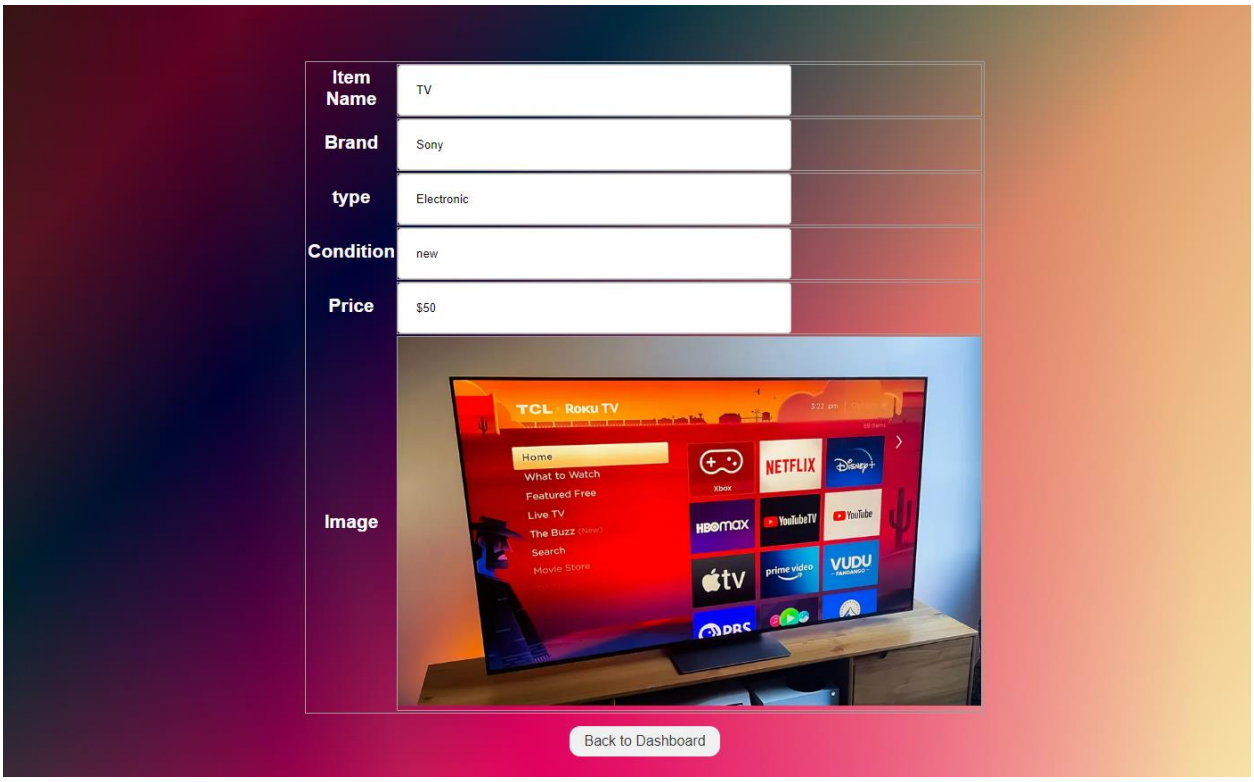
Edit Profile:



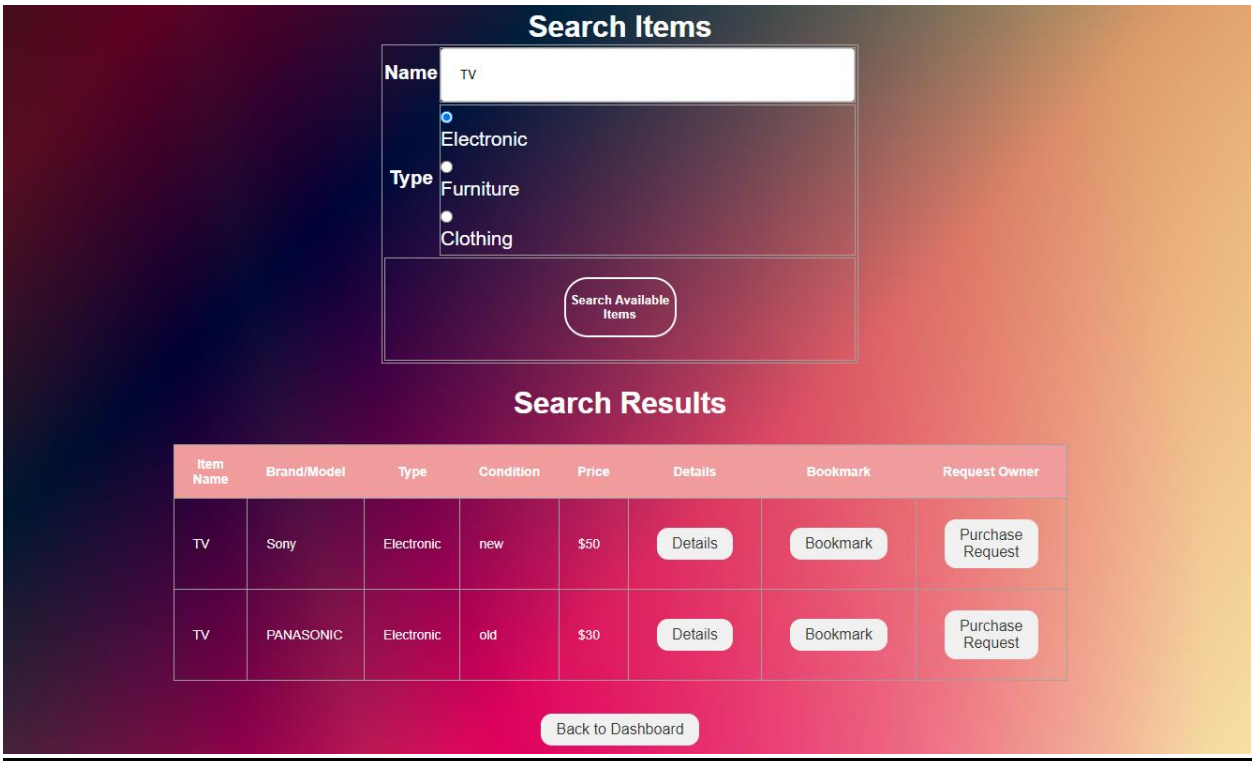
BookmarkItem:



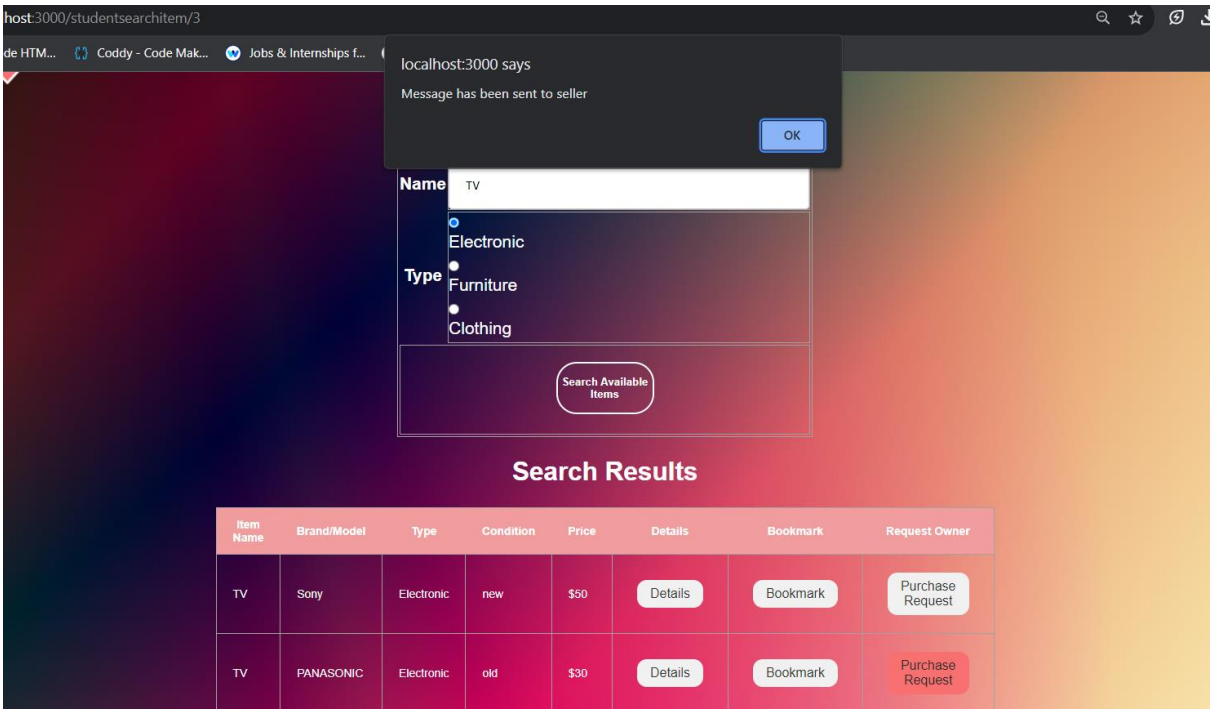
Details page:



Search Item:



Purchase Request:



Mail:

Regarding Service request Inbox ×

rehomegrp28@gmail.com
to me

Dear sankhesanmesh@gmail.com,

Greetings!.

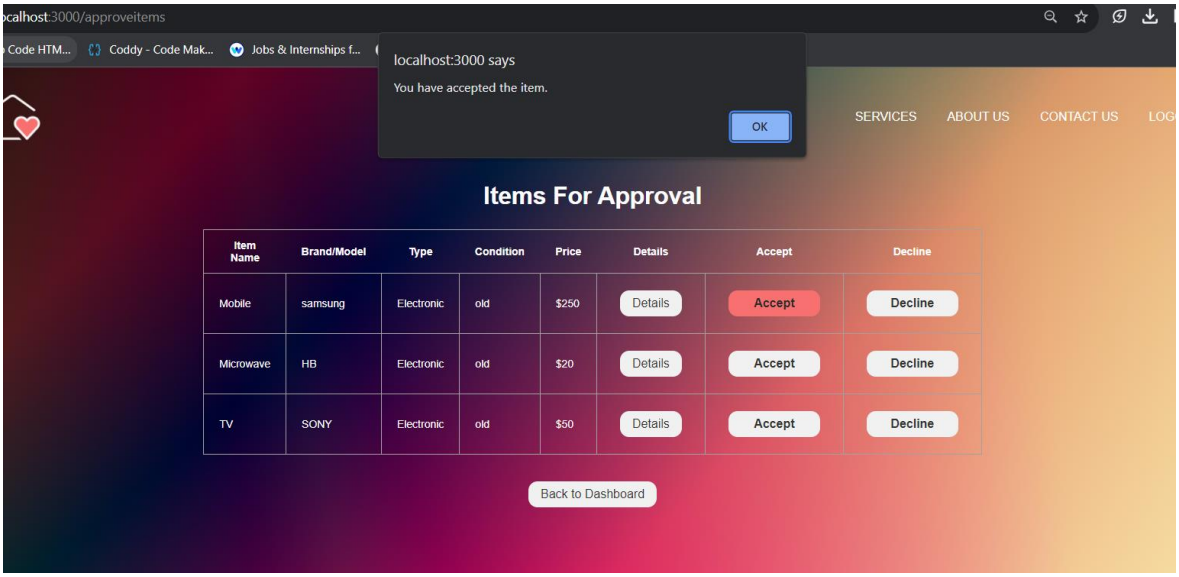
archisahu23@gmail.com would like to buy the TV which you posted on ReHome.

If you are interested please contact user by email provided.

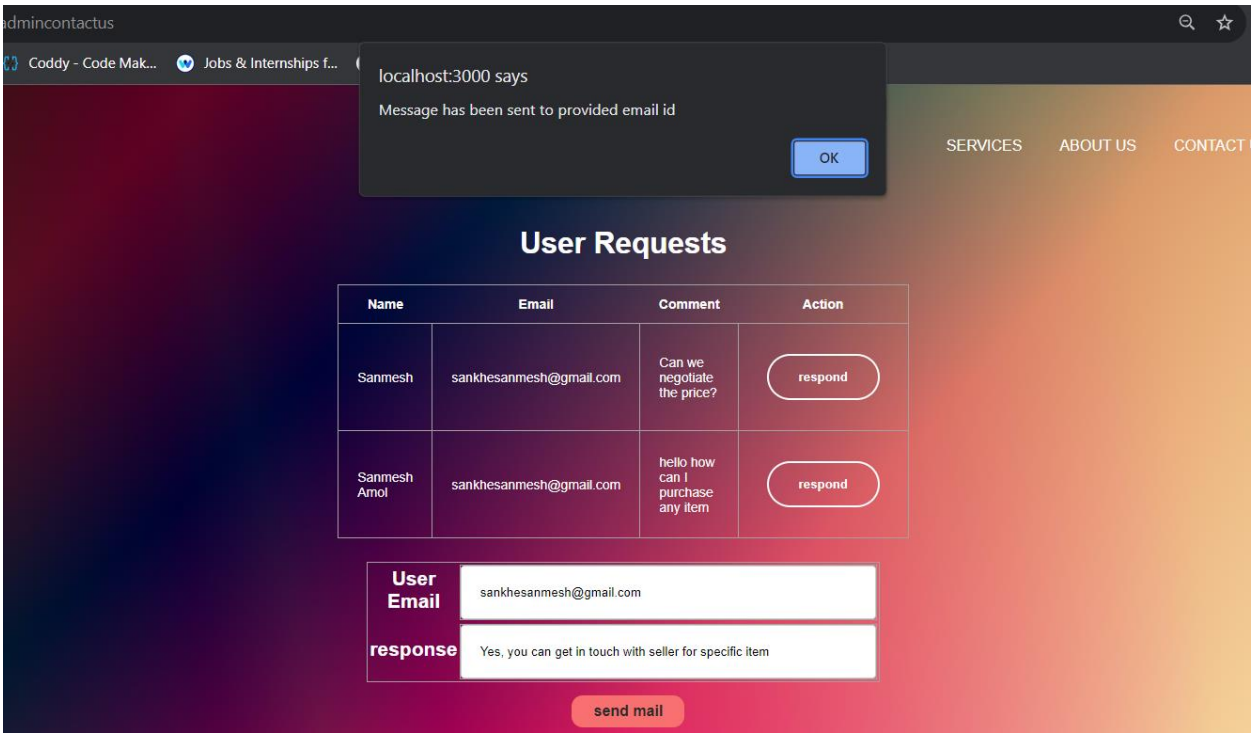
Best regards,

ReHome

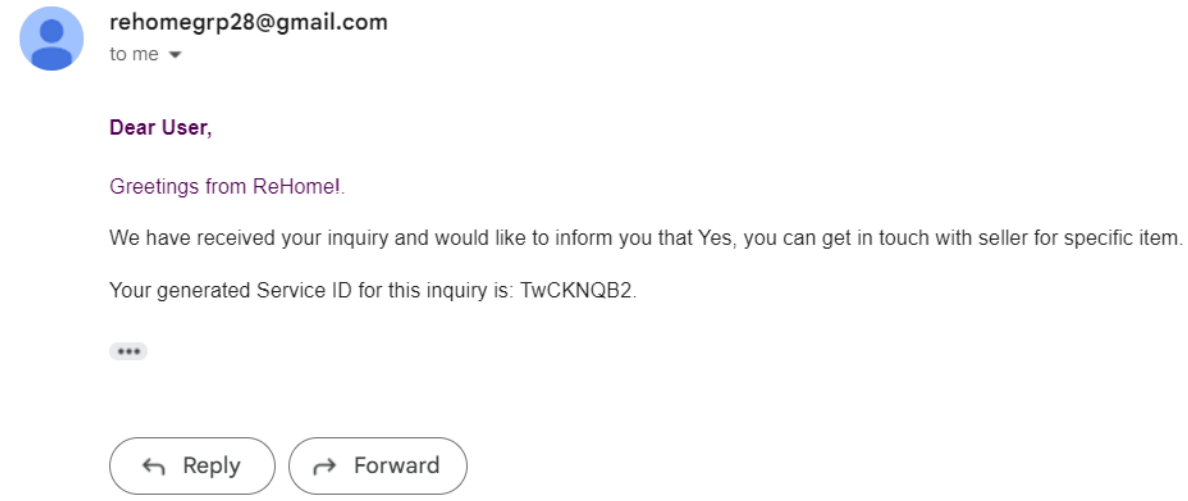
Approve Item:



Response Contact us:



Mail:



rehomegrp28@gmail.com
to me

Dear User,

Greetings from ReHome!.

We have received your inquiry and would like to inform you that Yes, you can get in touch with seller for specific item.

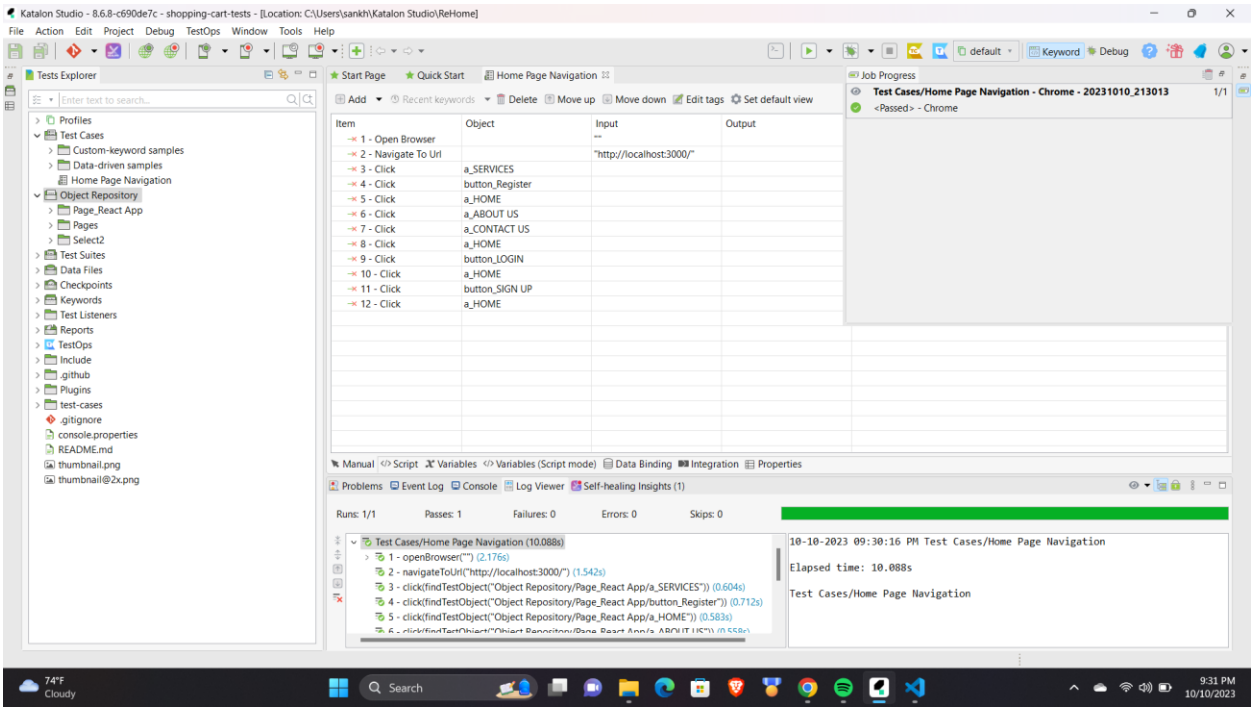
Your generated Service ID for this inquiry is: TwCKNQB2.

•••

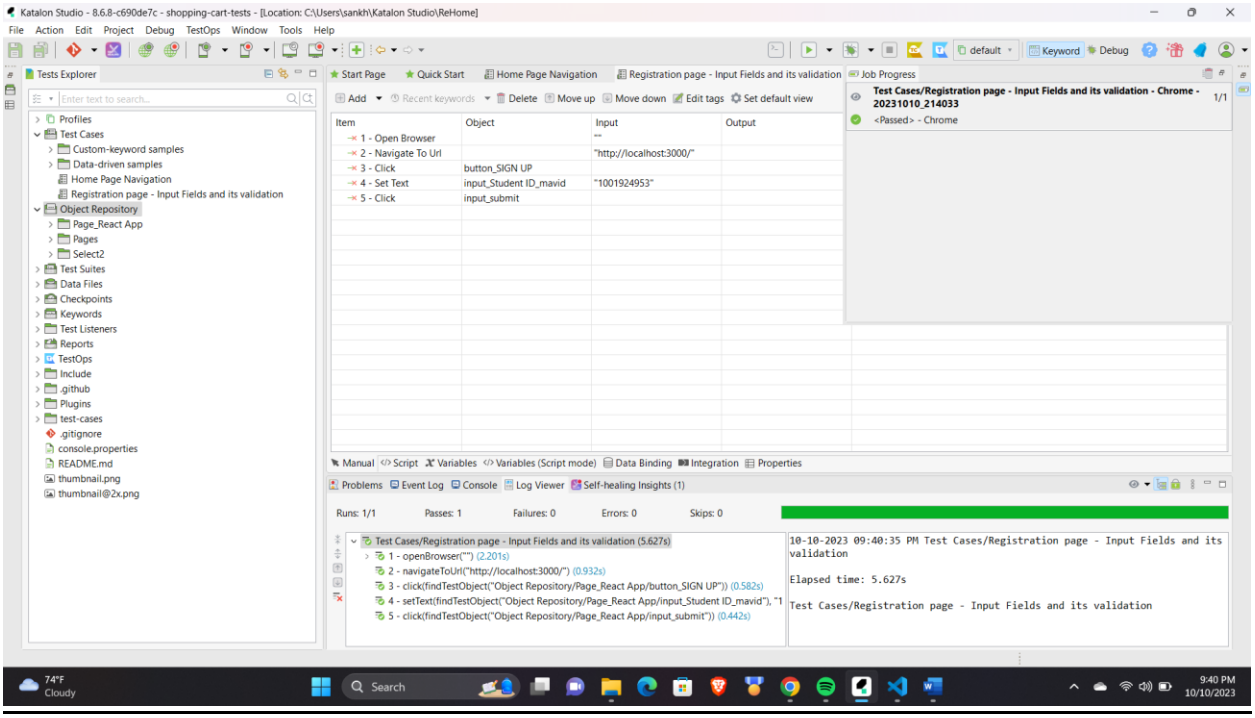
Reply Forward

Automation Testing:

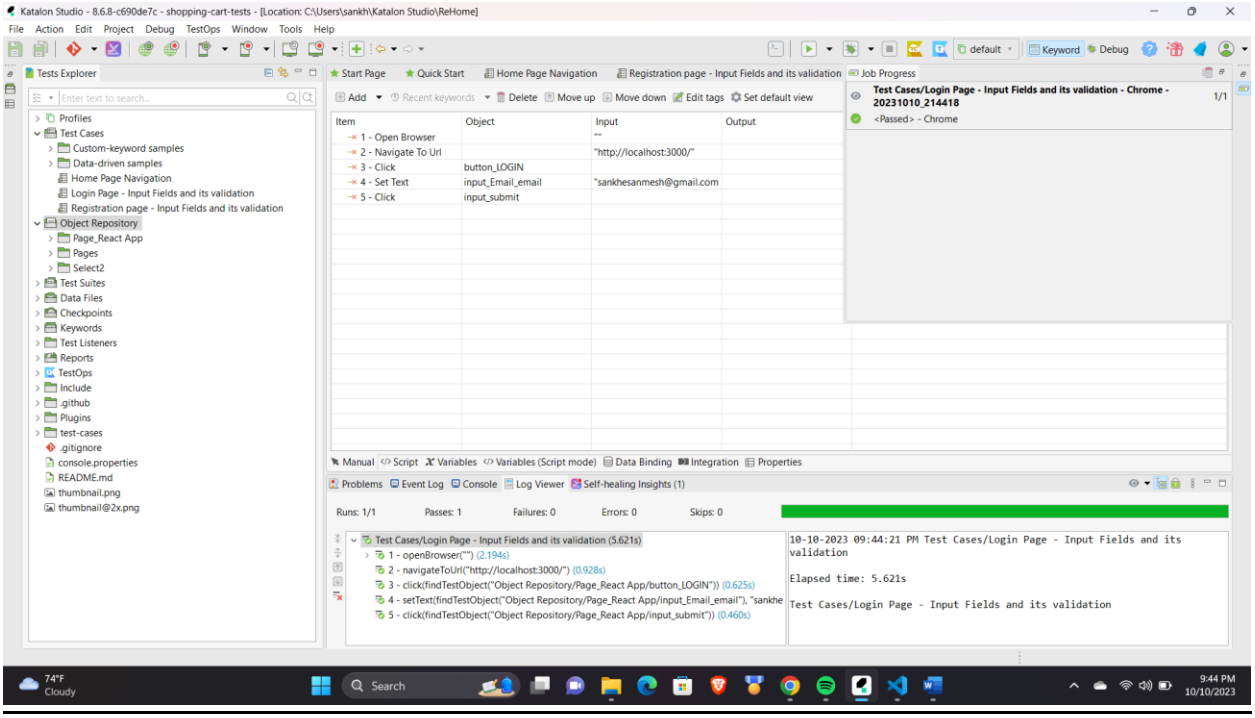
TC1:



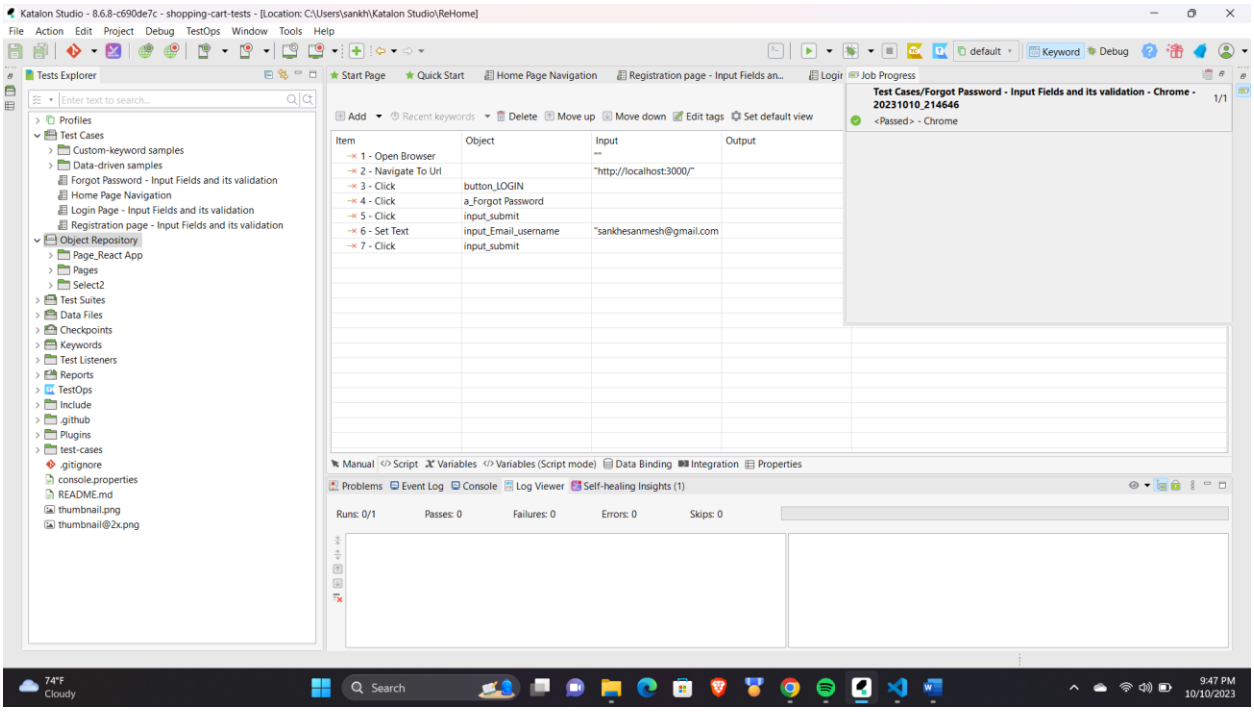
TC2:


Katalon Studio - 8.6.8-c690de7c - shopping-cart-tests - [Location: C:\Users\sankh\Katalon Studio\ReHome]
File Action Edit Project Debug TestOps Window Tools Help
Tests Explorer
Start Page
Quick Start
Home Page Navigation
Registration page - Input Fields and its validation
Job Progress
Test Cases/Registration page - Input Fields and its validation - Chrome - 20231010_214033
<Passed> - Chrome
1/1
Item
Object
Input
Output
1 - Open Browser
2 - Navigate To Url
"http://localhost:3000/"
3 - Click
button_SIGN UP
4 - Set Text
input_Student ID_mavid
"1001924953"
5 - Click
input_submit
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
Test Cases/Registration page - Input Fields and its validation (5.627s)
1 - openBrowser("") (2.201s)
2 - navigateToUrl("http://localhost:3000/") (0.932s)
3 - click(findTestObject("Object Repository/Page_React App/button_SIGN UP")) (0.582s)
4 - setText(findTestObject("Object Repository/Page_React App/input_Student ID_mavid"), "1
5 - click(findTestObject("Object Repository/Page_React App/input_submit")) (0.442s)
10-10-2023 09:40:35 PM Test Cases/Registration page - Input Fields and its validation
Elapsed time: 5.627s
Test Cases/Registration page - Input Fields and its validation
74°F Cloudy
Search
9:40 PM 10/10/2023

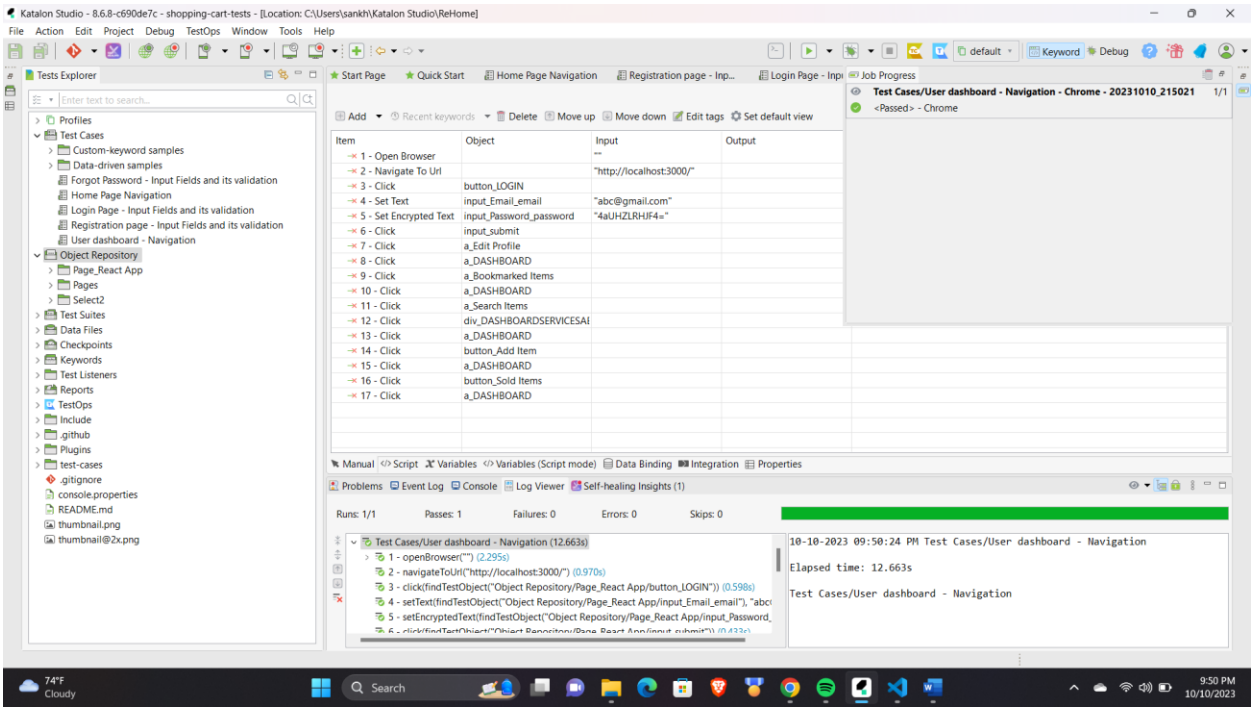
TC3:


Katalon Studio - 8.6.8-c690de7c - shopping-cart-tests - [Location: C:\Users\sankh\Katalon Studio\ReHome]
File Action Edit Project Debug TestOps Window Tools Help
Tests Explorer
Start Page
Quick Start
Home Page Navigation
Registration page - Input Fields and its validation
Job Progress
Test Cases/Login Page - Input Fields and its validation - Chrome - 20231010_214418
<Passed> - Chrome
1/1
Item
Object
Input
Output
1 - Open Browser
2 - Navigate To Url
"http://localhost:3000/"
3 - Click
button_LOGIN
4 - Set Text
input_Email_email
"sankhesanmesh@gmail.com
5 - Click
input_submit
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
Test Cases/Login Page - Input Fields and its validation (5.621s)
1 - openBrowser("") (2.194s)
2 - navigateToUrl("http://localhost:3000/") (0.928s)
3 - click(findTestObject("Object Repository/Page_React App/button_LOGIN")) (0.625s)
4 - setText(findTestObject("Object Repository/Page_React App/input_Email_email"), "sankhe
5 - click(findTestObject("Object Repository/Page_React App/input_submit")) (0.460s)
10-10-2023 09:44:21 PM Test Cases/Login Page - Input Fields and its validation
Elapsed time: 5.621s
Test Cases/Login Page - Input Fields and its validation
74°F Cloudy
Search
9:44 PM 10/10/2023

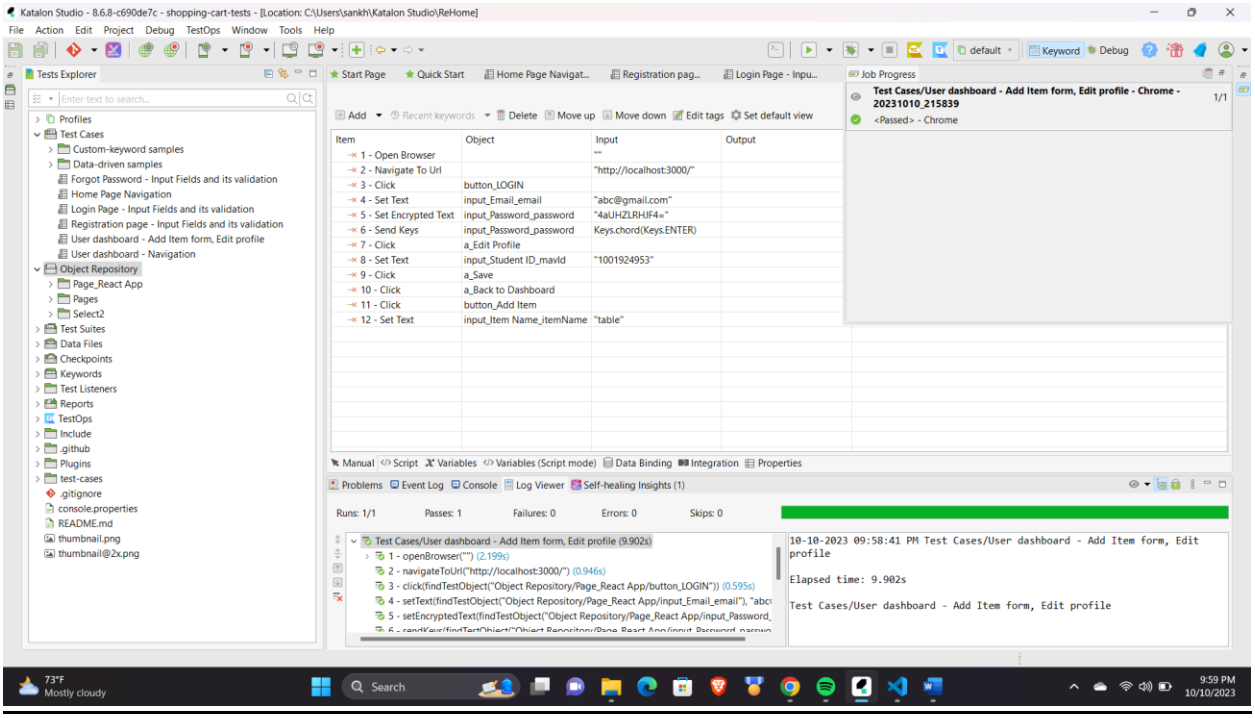
TC4:


Katalon Studio - 8.6.8-c690de7c - shopping-cart-tests - [Location: C:\Users\sankh\Katalon Studio\ReHome]
Test Cases/Forgot Password - Input Fields and its validation - Chrome - 20231010_214646
<Passed> - Chrome
Runs: 0/1
Passes: 0
Failures: 0
Errors: 0
Skips: 0
74°F Cloudy
9:47 PM 10/10/2023

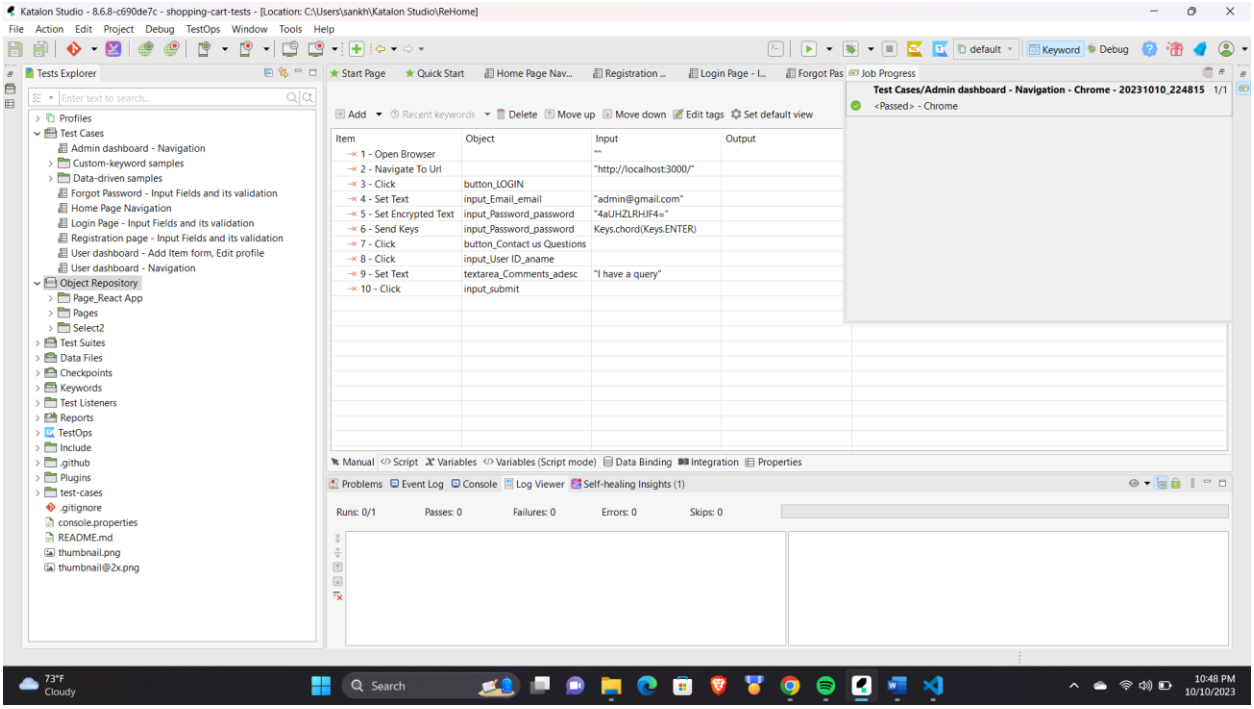
TC5:


Katalon Studio - 8.6.8-c690de7c - shopping-cart-tests - [Location: C:\Users\sankh\Katalon Studio\ReHome]
Test Cases/User dashboard - Navigation - Chrome - 20231010_215021
<Passed> - Chrome
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
Test Cases/User dashboard - Navigation (12.663s)
10-10-2023 09:50:24 PM Test Cases/User dashboard - Navigation
Elapsed time: 12.663s
Test Cases/User dashboard - Navigation
74°F Cloudy
9:50 PM 10/10/2023

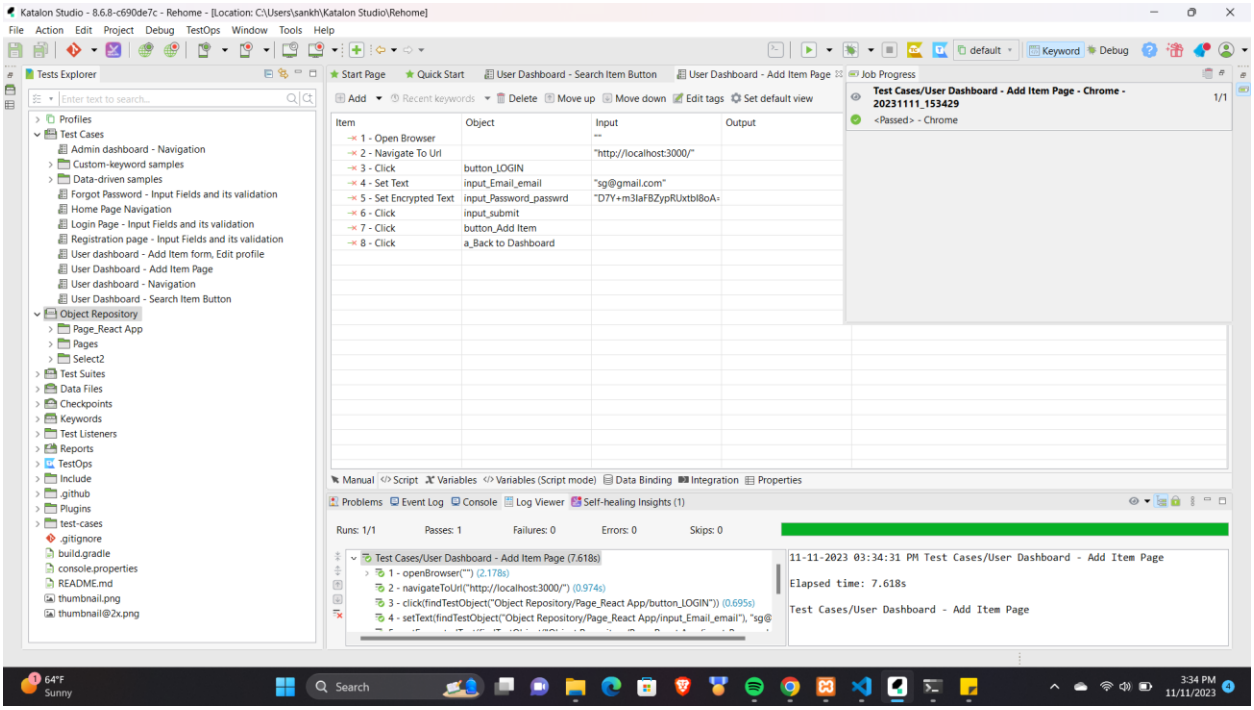
TC6:


Katalon Studio - 8.6.8-c690de7c - shopping-cart-tests - [Location: C:\Users\sankh\Katalon Studio\ReHome]
Test Cases/User dashboard - Add Item form, Edit profile - Chrome - 20231010_215839
<Passed> - Chrome
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
10-10-2023 09:58:41 PM Test Cases/User dashboard - Add Item form, Edit profile
Elapsed time: 9.902s
Test Cases/User dashboard - Add Item form, Edit profile
73°F
Mostly cloudy
9:59 PM
10/10/2023

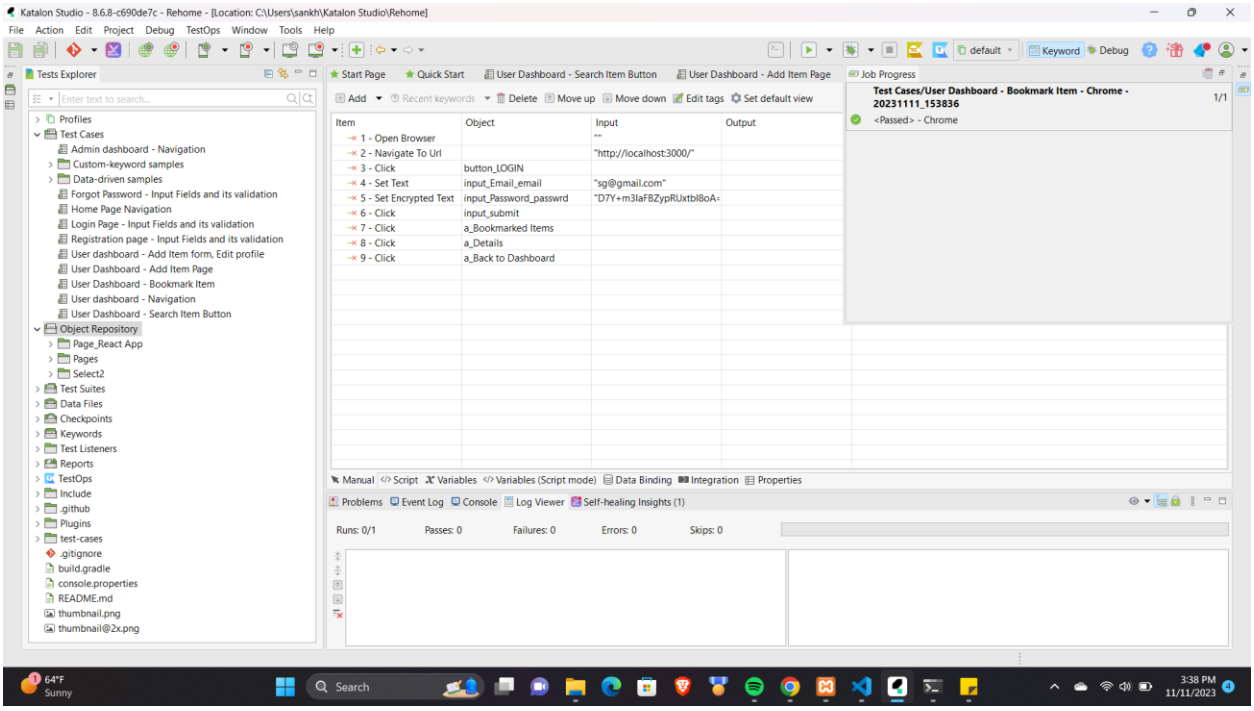
TC7:


Katalon Studio - 8.6.8-c690de7c - shopping-cart-tests - [Location: C:\Users\sankh\Katalon Studio\ReHome]
Test Cases/Admin dashboard - Navigation - Chrome - 20231010_224815
<Passed> - Chrome
Runs: 0/1
Passes: 0
Failures: 0
Errors: 0
Skips: 0
73°F
Cloudy
10:48 PM
10/10/2023

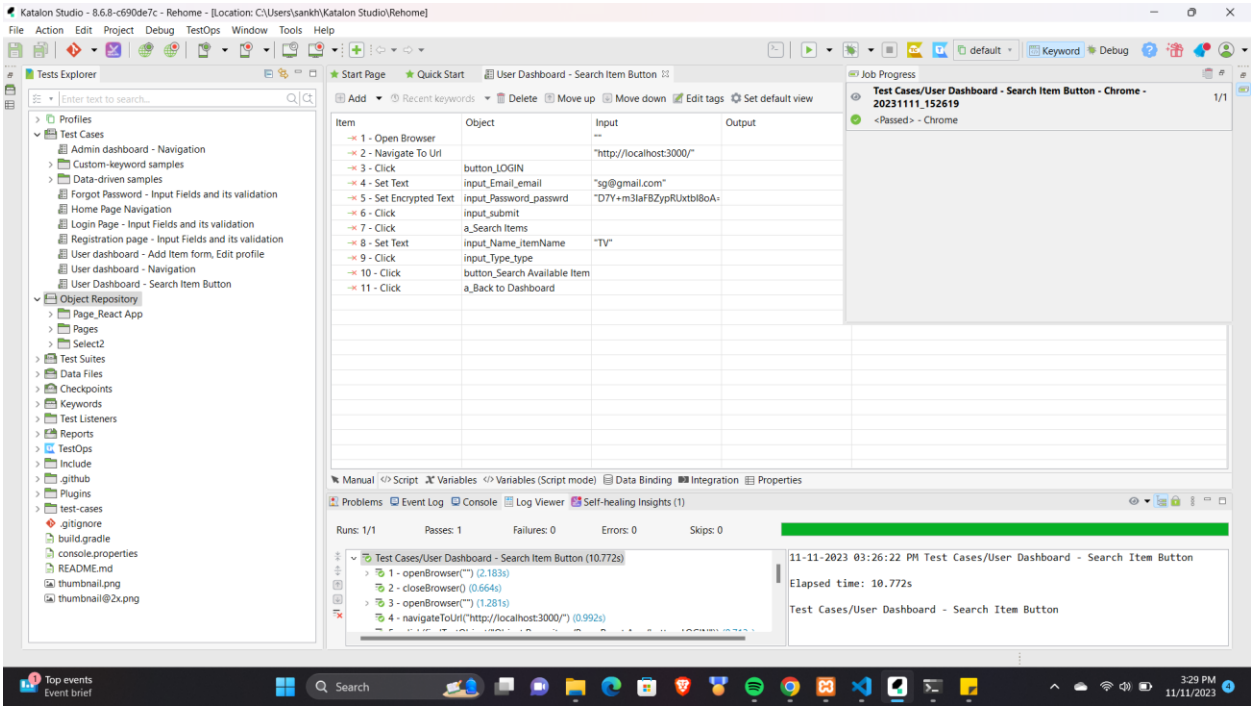
TC8:


Katalon Studio - 8.6.8-c690de7c - Rehome - [Location: C:\Users\sankh\Katalon Studio\Rehome]
File Action Edit Project Debug TestOps Window Tools Help
Tests Explorer
Start Page
Quick Start
User Dashboard - Search Item Button
User Dashboard - Add Item Page
Job Progress
Test Cases/User Dashboard - Add Item Page - Chrome - 20231111_153429
<Passed> - Chrome
1/1
Item
Object
Input
Output
1 - Open Browser
2 - Navigate To Url
"http://localhost:3000/"
3 - Click
button_LOGIN
4 - Set Text
input_Email_email
"sg@gmail.com"
5 - Set Encrypted Text
input_Password_passwrd
6 - Click
input_submit
7 - Click
button_Add Item
8 - Click
a_Back to Dashboard
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
Test Cases/User Dashboard - Add Item Page (7.618s)
11-11-2023 03:34:31 PM Test Cases/User Dashboard - Add Item Page
Elapsed time: 7.618s
Test Cases/User Dashboard - Add Item Page

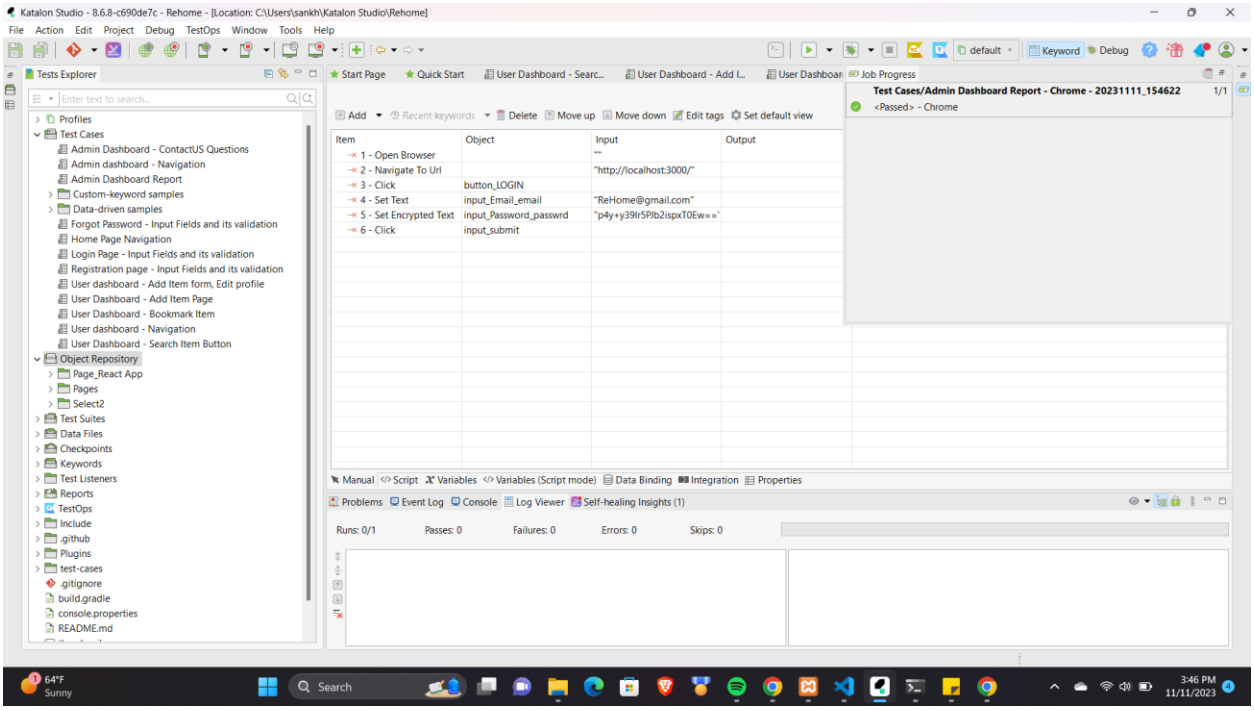
TC9:


Katalon Studio - 8.6.8-c690de7c - Rehome - [Location: C:\Users\sankh\Katalon Studio\Rehome]
File Action Edit Project Debug TestOps Window Tools Help
Tests Explorer
Start Page
Quick Start
User Dashboard - Search Item Button
User Dashboard - Add Item Page
Job Progress
Test Cases/User Dashboard - Bookmark Item - Chrome - 20231111_153836
<Passed> - Chrome
1/1
Item
Object
Input
Output
1 - Open Browser
2 - Navigate To Url
"http://localhost:3000/"
3 - Click
button_LOGIN
4 - Set Text
input_Email_email
"sg@gmail.com"
5 - Set Encrypted Text
input_Password_passwrd
6 - Click
input_submit
7 - Click
a_Bookmarked Items
8 - Click
a_Details
9 - Click
a_Back to Dashboard
Runs: 0/1
Passes: 0
Failures: 0
Errors: 0
Skips: 0

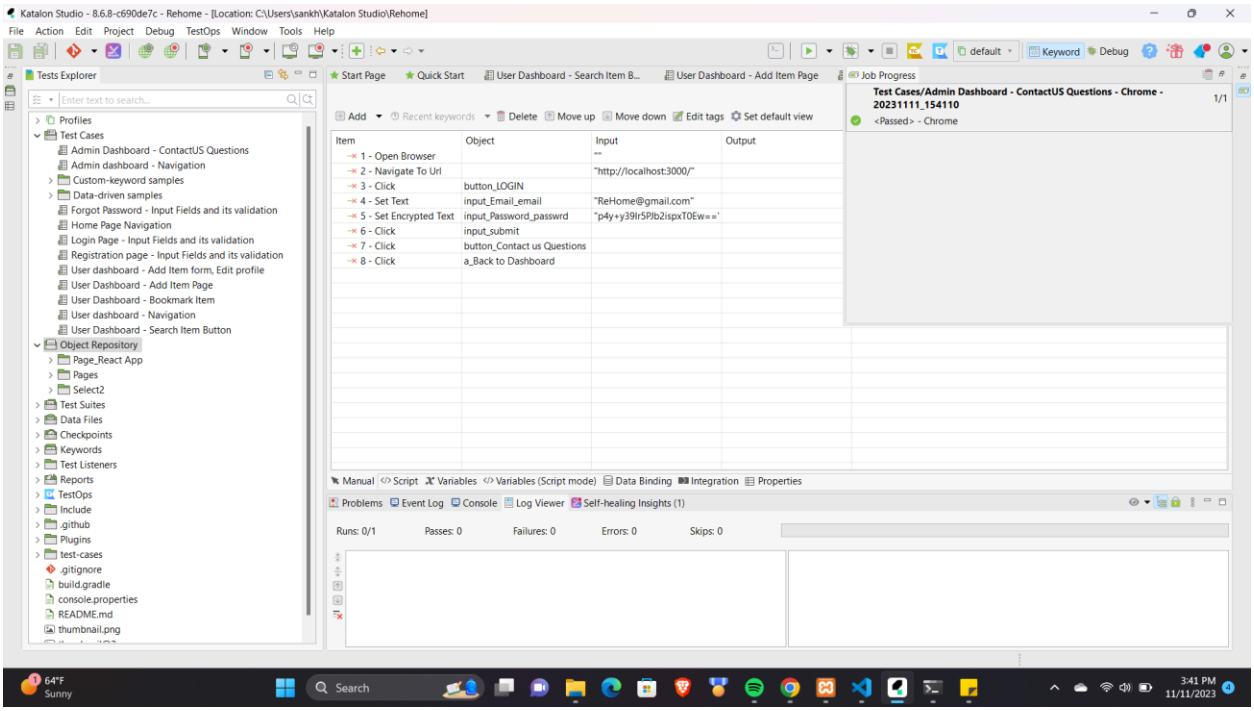
TC10:


Katalon Studio - 8.6.8-c690de7c - Rehome - [Location: C:\Users\sankh\Katalon Studio\Rehome]
Test Cases/User Dashboard - Search Item Button - Chrome - 20231111_152619
<Passed> - Chrome
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
11-11-2023 03:26:22 PM Test Cases/User Dashboard - Search Item Button
Elapsed time: 10.772s
Test Cases/User Dashboard - Search Item Button

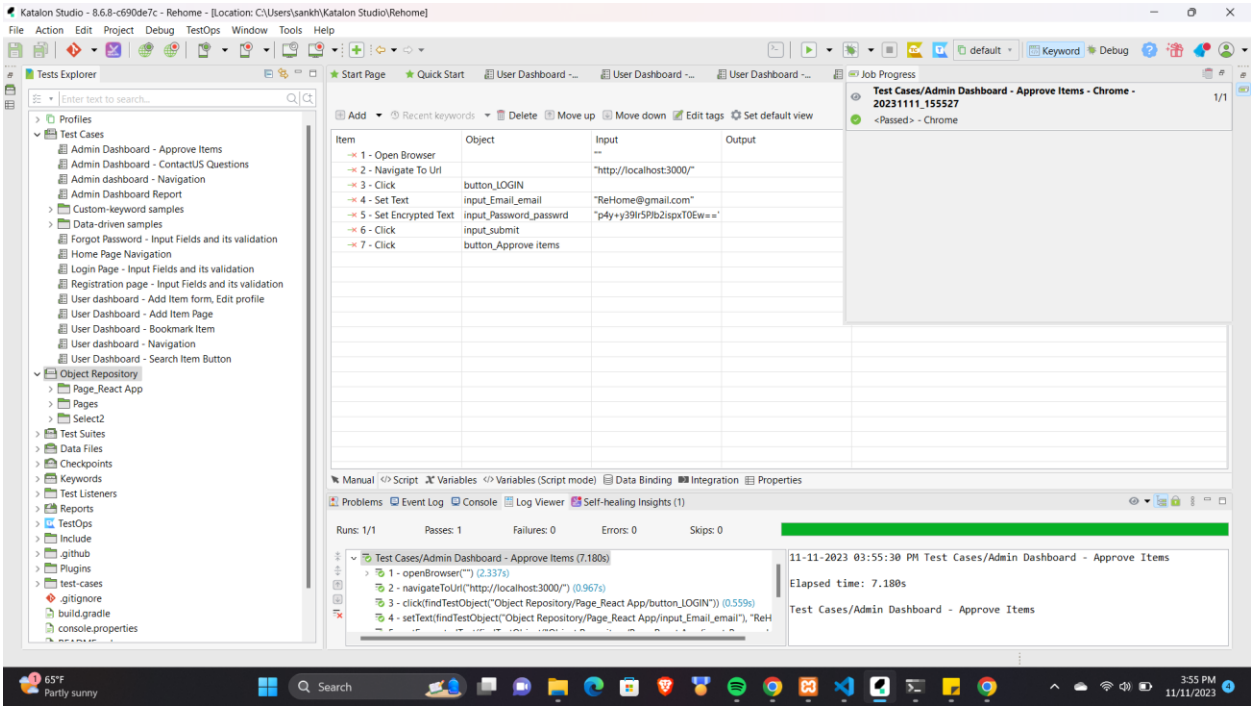
TC11:


Katalon Studio - 8.6.8-c690de7c - Rehome - [Location: C:\Users\sankh\Katalon Studio\Rehome]
Test Cases/Admin Dashboard Report - Chrome - 20231111_154622
<Passed> - Chrome
Runs: 0/1
Passes: 0
Failures: 0
Errors: 0
Skips: 0

TC12:


Katalon Studio - 8.6.8-c690de7c - Rehome - [Location: C:\Users\sankh\Katalon Studio\Rehome]
Test Cases/Admin Dashboard - ContactUS Questions - Chrome - 20231111_154110
<Passed> - Chrome
1/1
1 - Open Browser
2 - Navigate To Url "http://localhost:3000/"
3 - Click button_LOGIN
4 - Set Text input_Email_email "ReHome@gmail.com"
5 - Set Encrypted Text input_Password_passwrd "p4y+y39Ir5PJb2ispxT0Ew=="
6 - Click input_submit
7 - Click button_Contact us Questions
8 - Click a_Back to Dashboard
Runs: 0/1
Passes: 0
Failures: 0
Errors: 0
Skips: 0

TC13:


Katalon Studio - 8.6.8-c690de7c - Rehome - [Location: C:\Users\sankh\Katalon Studio\Rehome]
Test Cases/Admin Dashboard - Approve Items - Chrome - 20231111_155527
<Passed> - Chrome
1/1
1 - Open Browser
2 - Navigate To Url "http://localhost:3000/"
3 - Click button_LOGIN
4 - Set Text input_Email_email "ReHome@gmail.com"
5 - Set Encrypted Text input_Password_passwrd "p4y+y39Ir5PJb2ispxT0Ew=="
6 - Click input_submit
7 - Click button_Approve items
Runs: 1/1
Passes: 1
Failures: 0
Errors: 0
Skips: 0
Test Cases/Admin Dashboard - Approve Items (7.180s)
1 - openBrowser("") (2.337s)
2 - navigateToUrl("http://localhost:3000/") (0.967s)
3 - click(findTestObject("Object Repository/Page_React App/button_LOGIN")) (0.559s)
4 - setText(findTestObject("Object Repository/Page_React App/input_Email_email"), "ReH
11-11-2023 03:55:30 PM Test Cases/Admin Dashboard - Approve Items
Elapsed time: 7.180s
Test Cases/Admin Dashboard - Approve Items

Discuss how issues and bugs were managed and resolved:

Issue Tracking System:

- **Issue Logging:** Issues and bugs reported by testers, users, or developers were systematically logged in to an issue tracking system (Clickup.com). Each issue was described clearly, detailing the problem, steps to reproduce, expected behavior, and actual behavior.
- **Prioritization**: Issues were prioritized based on their severity and impact on the functionality of the application. Critical issues affecting core features or security were addressed with the highest priority.

Bug Triage and Assignment:

- **Bug Triage Meetings**: Regular bug triage meetings were held where the development team, QA team, and project stakeholders discussed reported issues. During these meetings, bugs were reviewed, and their severity and priority were reassessed if necessary.
- **Assignment:** Bugs were assigned to specific developers based on their expertise and workload. Clear ownership of each issue ensured accountability in the resolution process.

Root Cause Analysis:

- **Reproduction:** Developers attempted to reproduce reported bugs locally to understand the issue comprehensively. This step was crucial to identify the root cause accurately.
- **Debugging and Profiling:** Debugging tools and profilers were used to trace the code execution flow and identify the specific lines of code causing the problem. Profiling helped identify performance-related issues.

Resolution and Code Fixes:

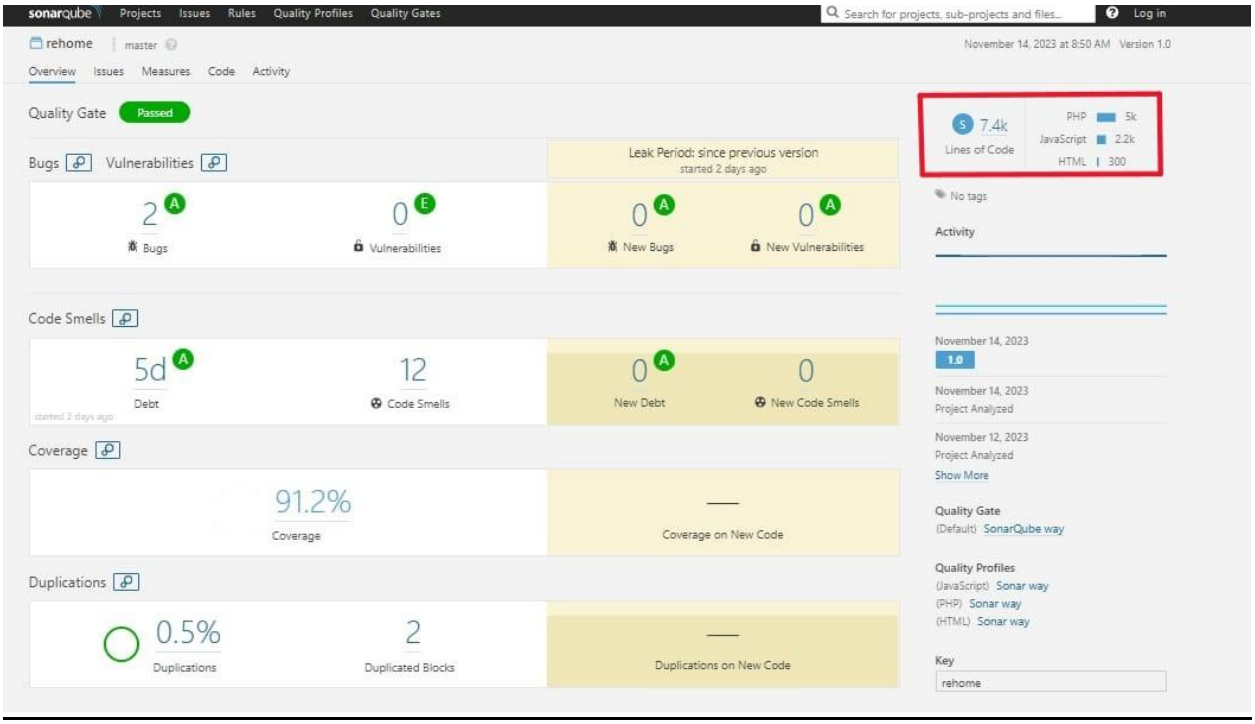
- **Code Fixes:** Developers made necessary code changes to fix the identified issues. Care was taken to ensure that fixes didn't introduce new bugs or regressions into the system.
- **Code Reviews:** Code fixes were reviewed by peers or senior developers to ensure the quality and correctness of the changes. Code reviews helped in catching potential issues before the changes were merged into the main codebase.

Testing and Verification:

- **Unit Testing:** Developers wrote unit tests specifically targeting the fixed issues to ensure that the problem was resolved and that the fix didn't break any other functionalities.
- **Regression Testing:** The affected areas of the application were thoroughly regression tested to confirm that the bug was resolved and that no new issues were introduced.

Address the code coverage and quality:

1. In Rehome project, created config property file for sonar, named as "sonar-project.properties" at project's root level
2. Run sonarqube:
2.1 go to zip extracted directory "path_to_sonar_qube\bin\windows-x86-64\" & click on "StartSonar.bat" to run it.
3.Run sonar scanner:
3.1 Open Rehome terminal.
3.2 go to sonar scanner zip extracted directory. copy path till bin/sonar-scanner.
3.3 run following command:
>path_to_sonar_scanner/bin/sonar-scanner
3.4 after execution completed, we will get url to visit generated report as follows:



Maintenance

Outline the ongoing maintenance plan:

Preventive Maintenance:

Regular Code Reviews and Refactoring:

- Arrange for regular code reviews to make sure that coding guidelines are followed and to pinpoint areas that need work.
- During regular refactoring sessions, motivate developers to take proactive measures to solve code smells and potential bottlenecks.

User Experience (UX) Enhancement:

- To find and fix any usability problems, regularly carry out usability testing.
- Considering user preferences and current design trends, update and optimize the user interface.

Corrective Maintenance:

Performance Optimization:

- To spot slow queries, resource-hungry processes, or bottlenecks in the system, periodically check its performance.
- To resolve detected performance concerns, put optimizations like caching, database indexing, or code restructure into practice.

Accessibility Improvements:

- Identify locations that could be difficult for users with disabilities to navigate by conducting regular accessibility compliance audits of the platform.
- Give development teams the guidance and instruction they need to prioritize and incorporate accessibility enhancements in upcoming releases.

Managing Updates, Patches, and Bug Fixes:

Bug Identification and Prioritization:

- Reported bugs are carefully identified, verified, and prioritized based on their impact on user experience and platform functionality.
- Critical issues affecting security or essential features are given the highest priority to ensure rapid resolution.

Development and Testing:

- Our development team works diligently to address identified bugs and develop necessary updates and patches.
- Rigorous testing procedures, including unit testing, integration testing, and user acceptance testing, are carried out to ensure the reliability and stability of the updates.

Version Control and Rollback Plans:

- Version control systems are utilized to track changes, enabling us to monitor modifications made to the platform's codebase.
- In the event of unforeseen issues after deployment, rollback plans are in place to revert the platform to the previous stable version promptly.

Scalability and Future-Proofing Considerations:

- **Scalable Database Architecture**: Select a database system that has the capacity to effectively manage a sizable amount of data and numerous concurrent user interactions. NoSQL databases with scalability and flexibility, such as MongoDB or Cassandra, are appropriate for managing unstructured or semi-structured data. Optimize database queries and indexes frequently to increase performance.

- **API Integrations**: Plan the architecture to allow for API and third-party service integration. Social media sites, payment processors, and other pertinent technologies fall under this category. The platform can utilize new features and technologies in the future without requiring major redesigns thanks to its flexibility in integrating other services.

- **Mobile Responsiveness:** Make sure the platform has an easy-to-use interface and is responsive to mobile devices. This will serve a wider audience and consider the growing trend of mobile device usage.

Conclusion

The "ReHome" project has successfully achieved its primary objective of creating a user-friendly online platform that facilitates the exchange of used items among international university students. The platform has not only met but exceeded the expectations set by the client. Some key achievements include:

1. **Efficient Resource Sharing:** ReHome has enabled students to share and sell their used items easily, promoting a culture of reuse and reducing waste. This has contributed significantly to sustainability efforts on campus.
2. **Cost Reduction:** International students have benefited from ReHome by finding affordable alternatives to purchasing new items. By buying and selling within the community, students have saved money, easing their financial burden.
3. **Community Building:** The platform has successfully fostered a sense of community among international students. By connecting them through the exchange of items, ReHome has facilitated meaningful interactions, friendships, and a supportive environment.
4. **User Support and Engagement:** ReHome has provided exceptional user support, ensuring a positive experience for both buyers and sellers. The "Contact Us" feature has facilitated seamless communication, addressing user queries promptly.

Lessons Learned:

1. **User-Centric Design:** Prioritizing user experience is paramount. Understanding the needs of the international student community and tailoring the platform accordingly has been crucial to its success.
2. **Effective Communication:** Clear and efficient communication between users and administrators is essential. Features like the "Contact Us" functionality and automated email responses have enhanced user satisfaction.
3. **Continuous Improvement:** Regularly updating and improving the platform based on user feedback is essential. Receiving feedback from users and adapting the platform to meet evolving needs is a continuous process.

Areas for Improvement:

1. **Enhanced User Engagement:** Exploring additional features to enhance user engagement, such as discussion forums or community events, could further strengthen the sense of community among users.
2. **Expanded Item Categories:** Diversifying the types of items that can be listed on the platform, beyond textbooks and furniture, could attract a wider user base. Including categories like electronics, sports equipment, and household items could be beneficial.
3. **Mobile Accessibility**: Ensuring the platform's full functionality on mobile devices, including a dedicated mobile app, could improve accessibility for users who primarily use smartphones for online activities.

Recommendations

Integration of Payment Gateway: Incorporate a safe payment gateway so that users of the site can make online purchases. Make sure there are strong security measures in place to safeguard user funds and foster user confidence.

Collaborations and Partnerships: Investigate joint ventures with local businesses, eco-friendly brands, and college associations to provide ReHome customers with special offers, incentives, or promotions, thus augmenting the platform's value proposition.

Enhanced User Experience: Create a mobile application version of ReHome to accommodate users who would rather use tablets and smartphones to access the platform, making sure that there is a consistent user experience across all platforms.

References

- **Libraries:** PHP mailer, ReCharts
- https://app.clickup.com/?fromLanding=true
- Refer testcase, ReHome description and Diagram from the report submitted previously.
- https://katalon.com/

#Appendices

Code snippets:

Login Page UI:

```
class login extends Component {

    constructor(props) {
        super(props)

        this.state = {
            email:"",
            passwrd:"",
            role:"",
            id:"",
            isLogin:false
        }
        document.body.classList.remove('dashboard-background');
         document.body.classList.add('home-background');
      }

      LoginBtnClick =(e)=>{

          e.preventDefault();
          const {email,passwrd}=this.state;
        //   let isValid=this.validateLoginFormData(email,password);
        const reqData={
            email:email,
            passwrd:passwrd
        };
        LoginService.login(reqData)
        .then((response)=>{
            console.log(response.data);
            if(response.data.isLogin =="false"){
                this.setState({role:"",isLogin:false},
                ()=>this.props.onLogin(this.state.isLogin));
                alert('Invalid user credentials');
            }else {
                console.log(response.data);
                //success login
                this.setState({role:response.data.role,isLogin:true, id:response.data.id },
                    ()=>this.props.onLogin(this.state.isLogin)//trigger back
                );

            }
        });
      }
```

Login Page DB:

```php
if ($_SERVER['REQUEST_METHOD'] === 'OPTIONS') {
  header('Access-Control-Allow-Origin: *'); // Replace '*' with the specific frontend origin
  header('Access-Control-Allow-Methods: GET, POST, OPTIONS');
  header('Access-Control-Allow-Headers: Content-Type');
  exit;
}

if ($_SERVER['REQUEST_METHOD'] === 'POST') {

    $rawData = file_get_contents("php://input");

    $requestData = json_decode($rawData, true);

    $email = $requestData['email'];
    $passwrd = $requestData['passwrd'];

    $sql = "select * from users where email='".$email."' and passwrd='".$passwrd."'";
    $result = $conn->query($sql);

    if ($result->num_rows === 1) {
        $row = $result->fetch_assoc();
        //echo json_encode($row);
        echo json_encode(['message' => 'Login successful','isLogin'=>'true','role'=>$row['ROLES'],'id'=>$row['ID']]);
    }else{
        echo json_encode(['message' => 'Invalid credentials','isLogin'=>'false']);
        return json_encode(['message' => 'Invalid credentials','isLogin'=>'false']);
    }

}
```

Forget password UI:

```jsx
return (
  <div className="Homemid">
        <section className="card">
            <h2 style={{ textAlign: 'center' }}>Forgot Password?</h2>
            <h5 style={{ textAlign: 'center' }}>
                Enter your email address. You will receive a verification email with instructions.
            </h5>
            <br />
            <form onSubmit={this.forgetPassClick}>
                <div className="form-group">
                    <label htmlFor="email">Email</label>
                    <input type="email" id="email" name="email"
                    placeholder="Enter your email" required
                    value={this.state.email}
                    onChange={this.handleInputChange}
                    disabled={false}
                    />
                </div>
                <br />
                <div className="form-group">
                    <input type="submit" value="Get Code" />
                </div>
            </form>
            <br />
            <div className="button-container">
               <Link to={AppUrl.signUp}><button type="button"
               className="dashbutton">Create a New Account? Signup</button></Link>
                <Link to={"/login"} className="dashbutton">Return to login</Link>
            </div>
        </section>
    </div>
)
```

Forget password DB:

```php
$rawData = file_get_contents("php://input");

// Convert the JSON data to an associative array
$requestData = json_decode($rawData, true);

$email = $requestData['email'];

$sql = "select * from users where email='".$email."'";
  $result = $conn->query($sql);

if ($result->num_rows >= 1) {
    $newPass = generateRandomString(6);

    $sql = "UPDATE users SET passwrd = '$newPass' WHERE email = '$email'";
    if ($conn->query($sql) === TRUE) {

        $emailBody=getEmailBody($newPass);
        smtp_mailer($email,'New Credentials For Password Reset',$emailBody);

        echo json_encode(['message' => 'Reset password details sent to registered email id']);
        return json_encode(['message' => 'Reset password details sent to registered email id']);
    } else {
        echo "Error updating password: " . $conn->error;
    }


}else{
    echo json_encode(['message' => 'Provided Email Id is not present']);
    return json_encode(['message' => 'Provided Email Id is not present']);
}
```

User Dashboard UI:

```jsx
return (
    <div>
        <h1 className="dashhead">User Dashboard</h1>
        <div className="content">

            <div className="left-side">
                <Link to={`/studenteditprofile/${id}`} className="dashbutton">Edit Profile</Link>
                <Link to={`/studentbookmarkitem/${id}`} className="dashbutton">Bookmarked Items </Link>
                <Link to={`/studentsearchitem/${id}`} className="dashbutton">Search Items </Link>
                <h2>
                    <p>Personal Information</p>
                </h2>
                <div className="perinfo">
                    <label>Student ID:</label>
                    <input type="text" value={userDetails.mavid} readOnly />
                    <label>Name:</label>
                    <input type="text" value={userDetails.uname} readOnly />
                    <label>Phone No:</label>
                    <input type="text" value={userDetails.phoneno} readOnly />
                    <label>Email:</label>
                    <input type="text" value={userDetails.email} readOnly />
                    <label>Address:</label>
                    <input type="text" value={userDetails.address} readOnly />
                </div>
                <br />
            </div>
            <div className="right-side">
                <div className="button-container">
                    <Link to={`/studentadditem/${id}`}> <button className="dashbutton">Add Item</button></Link>
                    <Link to={`/studentsolditems/${id}`}> <button className="dashbutton">Sold Items</button></Link>
                </div>
                <h2>Items List</h2>
                <div className="itemlist">

                    {posteditemDetails.map((rs) => (
                                            <PostItem key={rs.ITEM_ID} rs={rs} /> // Use the new component and pass the data as props
                                            ))}

                </div>
            </div>
        </div>
    </div>
)
```

User Dashboard DB:

```php
}elseif ($_SERVER['REQUEST_METHOD'] === 'GET'){
    $role = $_GET['role'];

  if($role === "StudentProfiledetails"){
        $id = $_GET['id'];
        $sql="select a.mavid,a.uname,a.Phoneno,a.address,u.email from  student as a join  users as u  on a.ID=u.ID where a.ID='$id'";
        $mysqli =mysqli_query($conn,$sql);
        $json_data=array();
        while($row = mysqli_fetch_assoc($mysqli))
        {
          $json_data[]=$row;
        }
        echo json_encode(['phpresult'=>$json_data]);
    }
    elseif($role === "Updatestudentprofile"){
        $id = $_GET['id'];
        $sql="select a.mavid,a.uname,a.phoneno,a.address,u.email,u.passwrd from  student as a join  users as u  on a.ID=u.ID where a.ID='$id'";

        $mysqli =mysqli_query($conn,$sql);
        $json_data=array();
        while($row = mysqli_fetch_assoc($mysqli))
        {
          $json_data[]=$row;
        }
        echo json_encode(['phpresult'=>$json_data]);
    }
    elseif ($role === "Fetchitems"){
        $id = $_GET['id'];
        $sql="select * from  add_items where ID='$id' and STATUS='AVAILABLE'";
        $mysqli =mysqli_query($conn,$sql);
        $json_data=array();
        while($row = mysqli_fetch_assoc($mysqli))
        {
          $json_data[]=$row;
        }
        echo json_encode(['phpresult'=>$json_data]);
    }
```

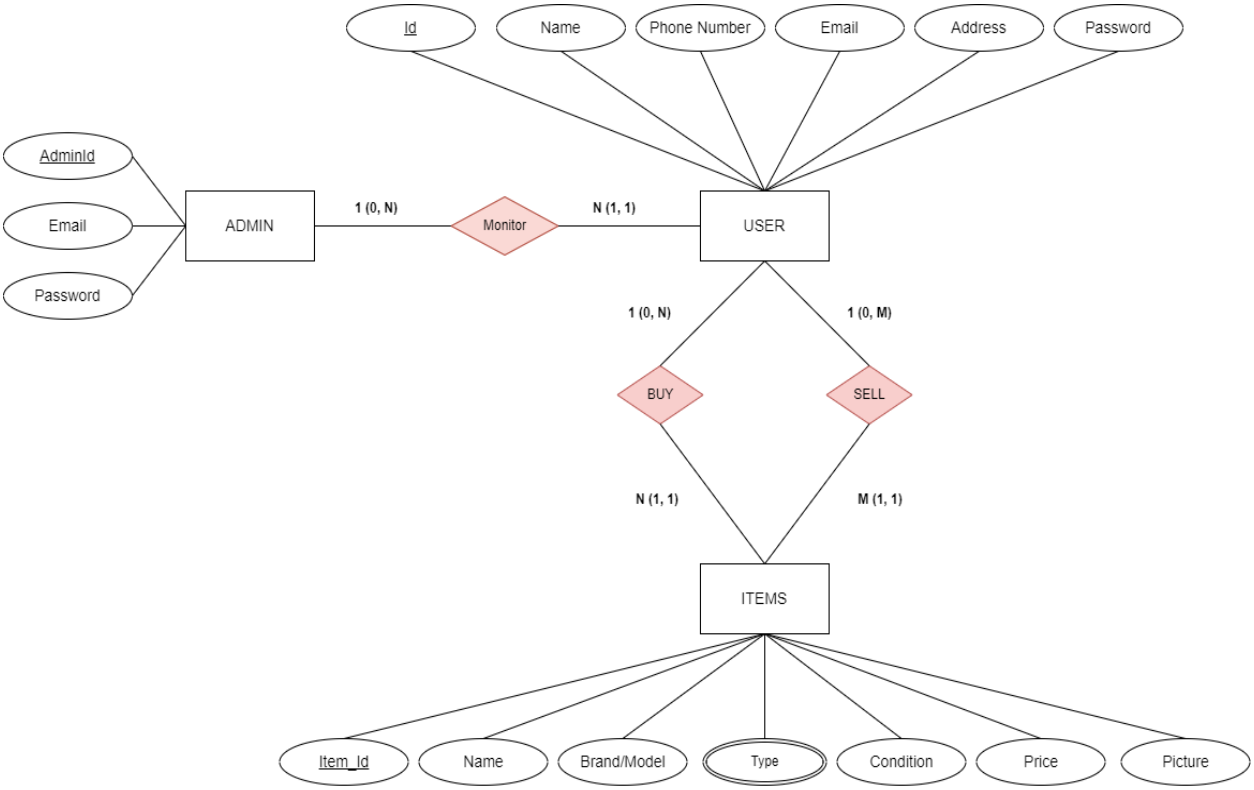
Admin UI:

```jsx
render() {
    document.body.classList.remove('home-background');
    document.body.classList.add('dashboard-background');

return (
    <div className="admin-dashboard">
        <h1>Admin Dashboard</h1>
        <div className='secondarycontent'>
            <div className="wrapper">
                <div className="tile-container">
                    <div className="box ">
                        Total Number of user: {this.state.totalUsers}
                    </div>
                    <div className="box ">
                        Total number of items posted: {this.state.totalItemPosts}
                    </div>
                    <div className="box ">
                        Total number of sold items: {this.state.totalSoldItem}
                    </div>
                </div>
            </div>
            <div className="button-container">

                <Link to={"/admincontactus"}><button className="dashbutton">Contact us Questions</button></Link>
                <Link to={"/approveitems"}><button className="dashbutton">Approve items</button></Link>


            </div>
        </div>
```

Admin DB:

```
        }
    }
    //-----------------active item on accept----------------------
    elseif($role === "acceptItem"){

        $jid = $requestData['jid'];
        $sql = "UPDATE add_items SET ACTIVE = 1 WHERE ITEM_ID=$jid;";

        if ($conn->query($sql) === TRUE) {
          echo "Accpeted Item  successfully.";
          return json_encode(['message' => 'Accpeted Job  successfully.']);
        } else {
          echo "Error: " . $sql . " " . $conn->error;
          return "Error: " . $sql . " " . $conn->error;
        }


    }
}
elseif ($_SERVER['REQUEST_METHOD'] === 'GET') {
        $role = $_GET['role'];

        if ($role === "FetchConatctusQuestion"){
          $sql="select * from contactus ";
          $mysqli =mysqli_query($conn,$sql);
          $json_data=array();
          while($row = mysqli_fetch_assoc($mysqli))
          {
            $json_data[]=$row;
          }
          echo json_encode(['phpresult'=>$json_data]);
      }
      elseif ($role === "FetchAllItem"){
          $sql="select * from add_items where STATUS='AVAILABLE' and ACTIVE=0";
          $mysqli =mysqli_query($conn,$sql);
          $json_data=array();
          while($row = mysqli_fetch_assoc($mysqli))
          {
            $json_data[]=$row;
          }
          echo json_encode(['phpresult'=>$json_data]);
  }
```

ER Diagram:



Click up Management tool: