DSCI 551 Midterm Progress Report

Title

iRecycler

Topic

Information exchange, environment friendly, resources saving, social networking.

Motivation

Our team bought a lot of new things to meet our basic requirements after arriving at university, and we've seen multiple times before that information on resale of similar items were posted on different platforms, such as Facebook and Wechat, at much lower prices, which made us feel somewhat unfortunate. In addition, we realized that some information was posted in an unorganized manner, which took time to read and compare. As a result, this time we decided to create an information platform about re-leasing, re-sale and other demands within a certain area (for example ktown, downtown, DPS and other places near USC), hoping it can help corresponding users exchange information effectively and save time and money to some extent. Besides that, we also hope it can do something with the environment by promoting people to buy less new items and instead to utilize their own items to the greatest. Afterall, we expect this platform to be a convenient information exchange tool that can meanwhile enhance the awareness on resource recycling and environmental protection.

Plan & Milestone

- Brainstorm app features and UI design
    - Ex. account management, personal profile, filters/keywords during searches, post management, etc.
    - Outline how features interact with each other and constitute the app as a whole
- Build the webapp
    - Firebase for data management
    - Use lectures and online references
- Test the webapp
    - Invite random people to use the app
    - Ask for evaluations and feedbacks
- Improve the webapp
    - Use evaluations and feedbacks as guides
    - Try to add more attractive details

Team members and responsibility

Our team expects to share the responsibilities equally so that we can benefit from practicing and learning the same skills at the same time for all the tasks.

- Ying Wang - Project proposal, page design, firebase design and connection…

- Ziyue Chen - Project proposal, page design, firebase design and connection…

Design process

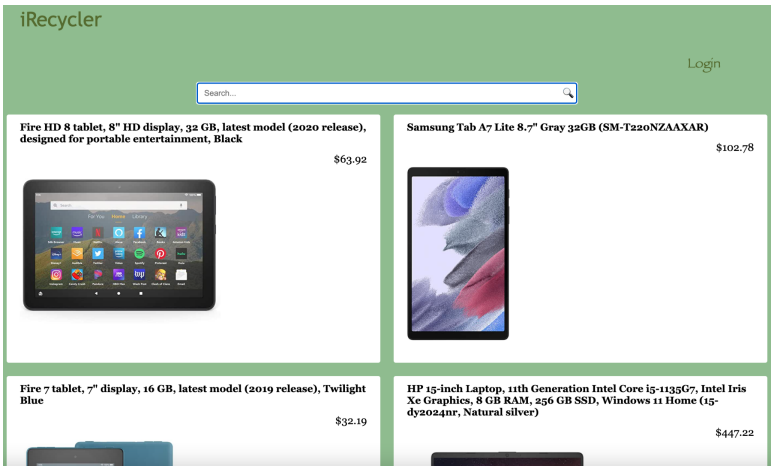
Up to this point, we have finished building the basic structures for our home page, login page, and posting page. Also, we have collected and uploaded our data to Firebase realtime database as registered users' information and their postings.

Home page

- Javascript, HTML, and CSS used
- Features
    - Search bar for inputting keywords
    - Posts containing keywords are matched and returned as individual grids below the search bar
    - iRecycler icon on the top left corner, and clicking on it will redirect to the home page again
- Code and picture

```html
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <meta http-equiv="X-UA-Compatible" content="ie=edge" />
        <title>Document</title>
        <link rel="stylesheet" href="homepage_style.css" />
        <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia">
    </head>
    <body>
        <div class="container">
            <div class="navbar">
                <a href="homepage.html"> iRecycler </a>
            </div>
            <nav>
                <ul id="login">
                    <li><a href="#">Login</a></li>
                </ul>
            </nav>
            <div id="searchWrapper">
                <input
                    type="text"
                    name="searchBar"
                    id="searchBar"
                    placeholder="Search..."
                />
            </div>
            <ul id="postList"></ul>
        </div>
        <script src="homepage.js"></script>
    </body>
</html>
```

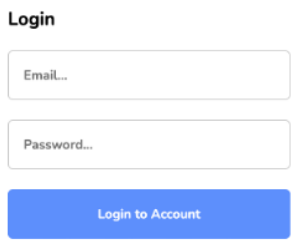
```javascript
const searchBar = document.getElementById('searchBar');
const postList = document.getElementById('postList');
let posts = [];

searchBar.addEventListener('keyup', (e) => {
    const searchString = e.target.value.toLowerCase();
    const filteredPosts = posts.filter((post) => {
        return (
            post.Item.toLowerCase().includes(searchString)
        );
    });
    displayPosts(filteredPosts);
});

const displayPosts = (posts) => {
    const htmlString = posts
        .map((post) => {
            return `
            <li class="post">
                <h2>${post.Item}</h2>
                <p>${post.Price}</p>
                <img src="${post.Image}"></img>
            </li>
        `;
        })
        .join('');
    postList.innerHTML = htmlString;
};

const loadPosts = async () => {
    try {
        const res = await fetch('https://retoolapi.dev/DX99sW/data'); // Will be replaced with firebase data api
        posts = await res.json();
        displayPosts(posts);
    } catch (err) {
        console.error(err);
    }
};

loadPosts();
```



Raw login page

- Javascript, HTML, and CSS used
- Features
    - Spaces for inputting email and password combinations
    - Login button
- Code and picture

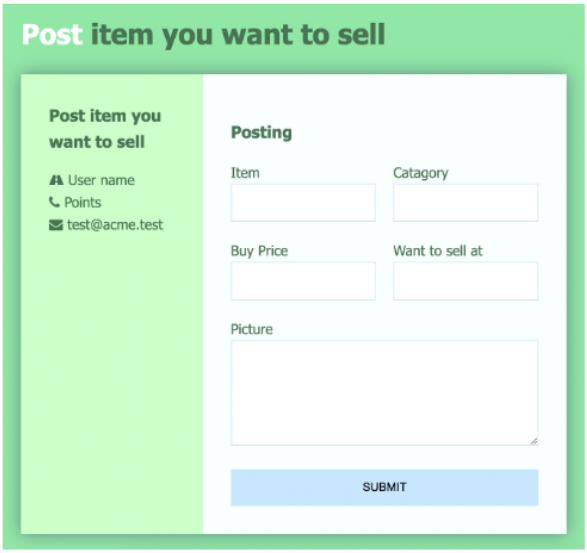
```html
<html>
<head>
  <title>Login page</title>
  <link href="https://fonts.googleapis.com/css?family=Nunito:400,600,700" rel="stylesheet">
  <link rel="stylesheet" href="register.css" />
</head>
<body>

  <div id="login_div" class="main-div">
    <h3>Login</h3>
    <input type="email" placeholder="Email..." id="email_field" />
    <input type="password" placeholder="Password..." id="password_field" />

    <button onclick="login()">Login to Account</button>
  </div>

  <div id="user_div" class="loggedin-div">
    <h3>Welcome User</h3>
    <p id="user_para">You're currently logged in.</p>
    <button onclick="logout()">Logout</button>
  </div>
```



Posting page

- Javascript, HTML, and CSS used
- Features
    - Spaces for inputting item name, category, buy price, sell price, and image
    - Submit button
- Connect to Firebase
    - Added Firebase configuration
    - When users fill in the blanks to post their items, the data will be uploaded to Firebase with the posting time
    - With the posting data stored, users will be able to see them by visiting the homepage, searching, or in their profiles
- Code and picture

```
  <link rel="stylesheet" href="post_style.css">
</head>
<body>
  <div class="container">
    <h1 class="brand"><span>Post</span> item you want to sell</h1>
    <div class="wrapper">
      <div class="company-info">
        <h3>Post item you want to sell</h3>
        <ul>
          <li><i class="fa fa-road"></i> User name</li>
          <li><i class="fa fa-phone"></i> Points</li>
          <li><i class="fa fa-envelope"></i> test@acme.test</li>
        </ul>
      </div>
      <div class="contact">
        <h3>Posting</h3>
        <div class="alert">Your post has been sent</div>
        <form id="contactForm">
          <p>
            <label>Item</label>
            <input type="text" name="item" id="item" required>
          </p>
          <p>
            <label>Catagory</label>
            <input type="text" name="catagory" id="catagory">
          </p>
          <p>
            <label>Buy Price</label>
            <input type="text" name="buy_price" id="buy_price" required>
          </p>
          <p>
            <label>Want to sell at</label>
            <input type="text" name="sell_price" id="sell_price">
          </p>
```



Database

- The data for this app, including user authentication and postings, will mainly come from users, but we also felt it necessary to collect some existing data to populate the database and for testing purposes.
- Data collection - user authentications
    - Generated fake usernames, emails, passwords, cities, zip codes, and etc. from fakenamegenerator.com
- Data collection - postings
    - Crawled item names, prices, and images from the first five pages of each product category on Amazon Warehouse using Selenium and BeautifulSoup.
- Data processing
    - Designed the structure of our data
    - Randomly assigned posting data to the users
    - Generated sell price by using extracted buy price multiplied by a random multiplier
    - Generated random timestamp for each posting data
    - Uploaded the final data to Firebase realtime database
- Database structure
    - User_name under users
    - User has attributes like user_name(key), age, city, email, gender, password, phone, zip code and posts including their history postings
    - Every postings has it unique post_id, other attributes including name, category, buy price, price want to sell, a multiplier, posting date, and picture of it

- Sample of database

Challenge faced

We are actively searching for solutions to solve these problems:

More connections Firebase

- Added existed user to Firebase authentication
- Make the authentication info uploaded to Firebase when users register for the first time, so with the authentication info stored users will be able to login again

Filter

- Want to allow users to apply filters to their searches based on price, category, post time, and etc.

Redirecting

- Want to connect the login page, home page, profile page, search page and ect. to allow the click operation to lead to the right page
    - For example, when the user is redirected to the homepage after login, we want to maintain their login state so that they can create new postings and see their posts in their profile page

Refinement

- Refine our all pages to make them as perfect as possible
    - For example, we want to display some of our newest postings on the home page by posting time
- Improve user experience after testing use by us designers

Expectation on the on-time completion of the project

From now on, we will keep resolving the challenges, and we want to work on the UI aspect of the pages so that they are more appealing to the users. Besides, we also want to brainstorm some methods to attract our users to use our app as usually as possible, like some promotion, point system or other motivational approaches.