Team Lorem Ipsum Milestone 2 Report

The PCBee PC Part Picker Project

Team Members:

Thomas Cheng
Matthew Widjaja
Kevin Nguyen
Greg Bennett

GitHub Repository Link:

https://github.com/Team-Lorem-Ipsum/PCBee

Table of Contents

What are we

PCBee is a project created by our group to fill an online niche surrounding the creation of PCs. The product name is based on “bees” and is meant to invoke the idea that our product will have the agility and speed of a bee. The APIs chosen for this project are Best Buy’s API and GPT API.

Project Overview

PCBee is a website built for people with an eBay account who need to make a PC for themselves. The website functions as both an educational site as well as a product search site where people learn what sort of PC builds to fit their budget and their performance needs; where they can learn from the beginning how to make a PC to fit their chosen function; and most of all, a website where people can search for whatever product or computer component they need for their PC’s.[1]

The site contains two main sections, these are an educational AI chatbot and a pc parts product search. The chatbot converses and interacts with the client and acts as a second opinion providing feedback as well as information regarding general knowledge of building a PC. The PC parts product search is a manual search for specific items on eBay. The product search acts as the main section of the site where users can look for parts that are available on eBay such as GPUs, CPUs, and monitors. These two sections will work in tandem with one another to help the user build a computer that will match the needs of the user.

The site was made using Bootstrap, Node.js, and render.com. The computer languages used in the project are HTML, Javascript, and CSS. These stacks and languages were chosen because they fulfilled all the necessary components for this project.

SDLC Methodology Overview

SDLC

The group's SDLC model was a combination of Kanban and Extreme Programming (XP). The group mostly used Kanban to visualize and communicate with the team members on who works on what and who manages tasks. We took the pair programming aspect of XP, which accelerates the implementation and debugging processes.

[1] On eBay

Retrospective

The biggest change that happened during the project was the move to replace Best Buy API with eBay API. As the team realized during the API implementation process, Best Buy only gives out its API key to companies and organizations. Hence, the team had to search for alternatives and found that the eBay API served the same roles as the Best Buy API.

The team also decided to drop React and used Node.js and Render to create the back-end code and host the server respectively. During the implementation process, the team realized that Bootstrap was sufficient for creating the front end and that React was not necessary. Next, the sudden inclusion of back-end infrastructure was due to the team later learning that GPT API and eBay API require back-end features such as environment variables and session IDs. Node.js was chosen as it uses JavaScript, a language all of the team knew, and Render as it is free.

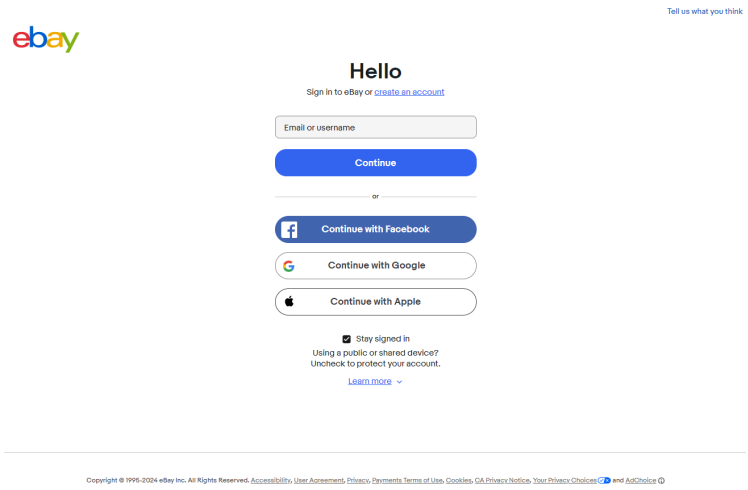
There were also changes in the API features. Other than feature changes to accommodate eBay API, the GPT API features had minor tweaks as well. We originally planned to have a dedicated PC recommendation mode where the AI will only send us a JSON of its PC recommendation. It turns out the GPT AI can't reliably output the JSON in the format we want, so we integrate it into the default chatbot mode. For eBay API, while the search feature remains, the other two are now replaced with "see related items" and "see seller analytics". These changes were made as eBay couldn't accommodate the previous filter items feature, and the image clicker feature didn't make use of eBay API.

Implemented Features and User Stories

eBay API:

Feature #1: Search PC parts.
Story #1: As a student, I want to easily pick out PC parts so that I can do a quick price search.

Feature #2: Search PC parts, see related items, and see seller analytics.
Story #2: As an accountant, I want to build a PC with parts from eBay as it has a wide selection of items and information on each seller.

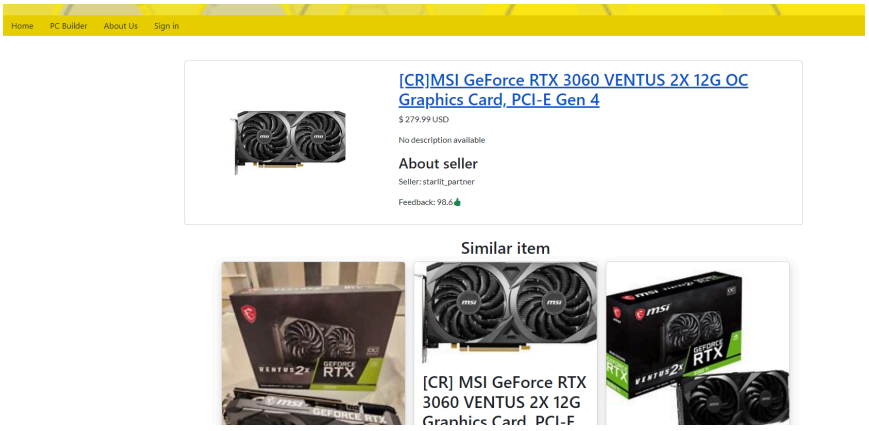


GPT API:

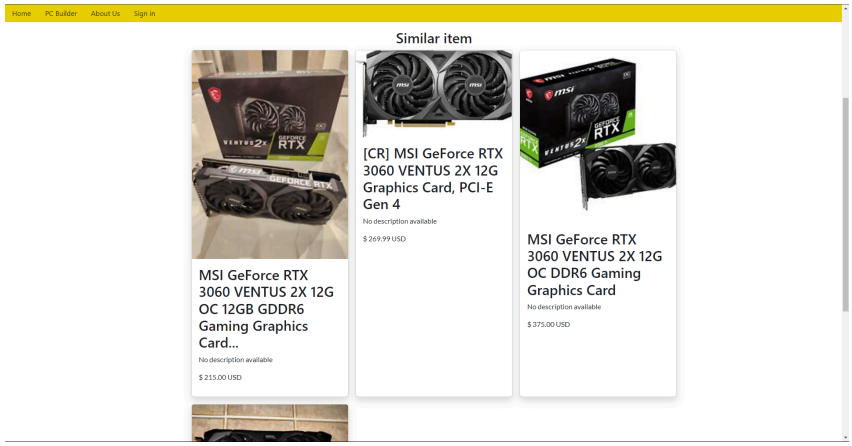
Feature #1: Teaches you what each PC part does.
Story #1: As a programmer, I need someone to speak to teach me about PC parts so that I can understand how my code interacts with hardware.

Feature #2: Recommends a PC build.
Story #2: As a software engineer, I want a quick suggestion for a PC build because I already work full time and can't spend time researching.

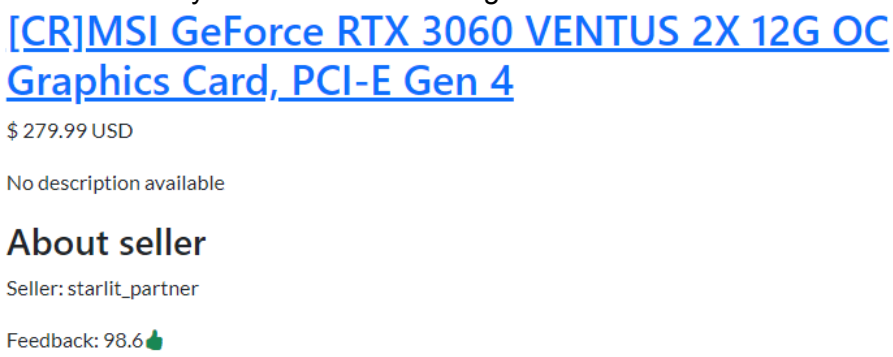
API Features (WIP)

- eBay:
  - PC part search:
    - We will use eBay's API to search for items and see their details.
    - 

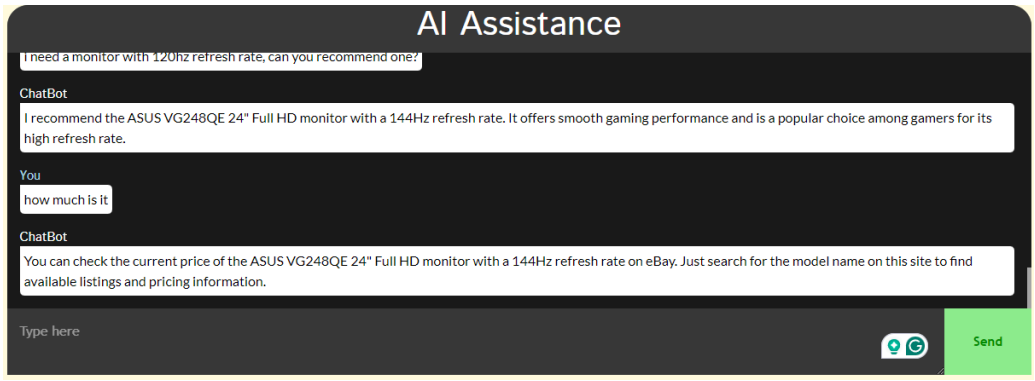
  - See related items:
    - We will use eBay's API to see other product listings in the same category that are available.



- See item rating:
  - We will use eBay's API to see the rating of the item.



- GPT API:
  - Educational for explaining PC parts:
    - The GPT chatbot can explain each PC part to help clear any confusion.
  - Recommends a PC build:
    - The GPT chatbot can also give builds with a specified feature (budget, range, etc.)



  - How to build a PC:
    - The GPT chatbot can help guide users on how to put parts together when building a PC.

Description of Tests (WIP)

- Unit Testing
    - eBay
        - Search
            - Entering “Intel i7 CPU” and expecting a product listing with no limit to the result.
            - Entering “iPhone15” and expecting a product listing with 10 limits to result.
        - Related products
            - Entering “Intel i7 CPU” and expecting other Intel chips or AMD chips.
            - Entering “Nvidia RTX3080” and expecting other Nvidia GPUs or AMD GPUs.
        - Seller analytics
            - Able to see the rating of the product
    - GPT
        - Producing product-related prompts
            - Entering prompts such as “How does a car work” will be responded to by AI with “Sorry my focus is on computers and PC parts”
        - Check whether or not it interacted with the user on the given chatbot environment
            - Expect GPT API to respond and send chat messages when the user asks questions
            - Expect GPT API to properly integrate with the chatbox messaging environment
        - Check if the chatbot responds to user inquiries regarding budget and fidelity
            - expect.not.toBe(x>800$) when given a prompt in which the user asks for a build setup with a budget of 800$.
            - expect.not.toBe(x<100$) when given a prompt in which the user asks for a build setup with a budget of 800$ (making sure the AI makes use of most of the budget).

Description of CI/CD Infrastructure (WIP)

The CI/CD for this project was implemented with GitHub Actions and was relatively simple. The main branch is protected and requires pull requests (PRs) to merge changes. Each PR also requires a reviewer to review all code that is being merged.

The .yml file runs several simple steps:

1. Install Node.js and Node Package Manager (NPM).

2. Install all Node dependencies.
3. Run the “start” script (hosts the server locally, in this case on a Ubuntu virtual machine).
4. Run the “test” script (using Jest to test the API functions).
5. Use cURL to deploy the site on Render using a webhook.

```yaml
name: Node.js CI

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

jobs:
  build:

    runs-on: ubuntu-latest

    strategy:
      matrix:
        node-version: [20.x]
        # See supported Node.js release schedule at https://nodejs.org/en/about/releases/

    steps:
    - uses: actions/checkout@v3
    - name: Use Node.js ${{ matrix.node-version }}
      uses: actions/setup-node@v3
      with:
        node-version: ${{ matrix.node-version }}
        cache: 'npm'
    - run: npm install
    - run: npm start
    - run: npm test
    - run: curl https://api.render.com/deploy/srv-cnqebo6d3nmc7393cqv0?key=L9jaKXy2aec
```

Data Flow Diagram

Project Takeaways

The project overall didn't experience a project-ending setback, but some difficulties caused some major scope switches. The first obstacle we overcame when we learned to always ensure the APIs you are going to use will work with your scope, budget, and schedule. We initially assumed Best Buy API was free but it turned out to have a paywall that we didn't have the schedule or budget to navigate, so we were forced to make a scope switch to eBay API. It is also important to check if the API endpoints you're going to utilize are still active and not deprecated.

The second obstacle was our lack of knowledge of how APIs work. We assumed both GPT API and eBay API could work with only the front-end, only to discover that they require back-end functionalities such as session IDs and environment files.

The third obstacle was our lack of experience and knowledge regarding back-end and server-side programming. We severely underestimated how difficult it would be to incorporate them into our project and ended up using a lot of slack in our project schedule. This could have been avoided during the planning phase had we tested out the stacks we planned on using.