사용자 친화적인 컴퓨터 견적 생성 웹 사이트 개발

조규철*, 이재윤O
*인하공업전문대학 컴퓨터정보과,
O인하공업전문대학 컴퓨터정보과
e-mail: kccho@inhatc.ac.kr*, dldi1021@naver.comO

Develop User Friendly Website for Computer Estimate

Cho Kyu Cheol*, Lee Jae YunO
*Dept. of Computer Science & Engineering, Inha Technical College,
ODept. of Computer Science & Engineering, Inha Technical College

● 요 약 ●

검색을 통해 찾은 컴퓨터 부품의 정보(가격, 호환 여부, 성능) 등을 자동으로 체크 해주고 이를 통해 만든 컴퓨터 견적을 여러 사람과 공유 및 열람을 할 수 있고 대화창과 같은 게시판을 통해 서로의 지식과 정보를 쉽게 공유할 수 있는 웹 사이트를 개발하였다. 이 사이트가 사용자들의 합리적인 소비로 이어지기를 기대한다.

키워드: Spring Boot, JPA, Axios, 컴퓨터 견적, Async/Await

I. Introduction

2000년대 이후 개인용 컴퓨터의 수요는 꾸준히 증가하였다. 잠시 스마트폰의 등장에 2010년대에는 주춤하긴 했지만 2020년대에 들어서 AI 산업, 암호화폐, 메타버스 등의 대두로 외장형 GPU에 대한 수요도 증가하여서 현재 PC와 외장형 GPU의 시장은 꾸준히 증가하고 있다.

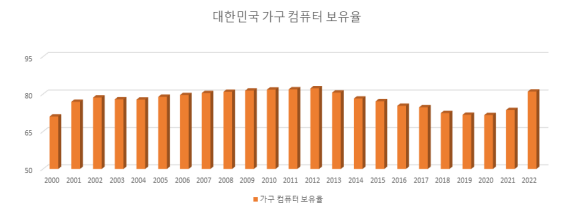

Fig. 1. 대한민국 가구 컴퓨터 보유율 [1]

우리는 컴퓨터를 구매할 때 부품에 대한 정보 없이 삼성, LG와 같은 대기업에서 출시하는 완제품 PC를 쉽게 살 수 있다. 자신의 용도에 맞춰서 견적을 만드는 게 제일 좋지만, 컴퓨터를 구성하는 모든 부품에 대한 정보를 찾는 것은 관련 지식이 없는 사람에게는 매우 복잡한 일이다. 완제품 PC는 이러한 조사 없이 가격대를 보고 대충 자신이 원하는 위치의 컴퓨터를 살 수 있고 A/S를 쉽게 받을 수 있다는 장점이 있지만, 직접 조립한 PC에 비해 가격이 높다는 단점을 가지고 있다.

브랜드 PC를 구매할 때도 부품에 대한 기본지식이 있어야 합리적인 소비를 할 수 있는데, 아직 이러한 지식이 부족한 사람들이 많다. 예를 들어 영상편집에는 높은 RAM 용량과 적절한 GPU 파워가 필요한데 이러한 사전지식 없이 그저 높은 사양이 필요하다는 이유로 게이밍 완제품 PC를 구매한다면 같은 돈을 사용해도 직접 조립한 PC보다 더 낮은 퍼포먼스를 얻을 수밖에 없을 것이다.

만약 견적의 호환 여부를 자동으로 검사해줘서 쉽게 견적을 생성하고 생성한 견적의 부품 정보를 쉽게 공유한다면 서로 채팅방에서 질문하듯이 대화하며 자신의 지식과 경험을 설명해줄 수 있는 사이트를 구현하는 것이 목적이다.

II. Development of Computer Estimate Site

2-1. Development Environment

본 사이트 개발에 사용된 IDE는 InteliJ이다. Framework는 Spring boot를 이용하였으며 DB는 MYSQL을 JPA를 사용해 연동하였으며 템플릿 엔진으로 Thymeleaf를 사용하여 동적 페이지를 생성하였다.

WAS로 Tomcat, 부품 정보 크롤링을 위해서 Selenium[2]의 의존성을 추가하였다. JavaScript를 활용한 부품 정보 추가와 검색을 위해 Axios를 활용해 Ajax 비동기 통신을 구현하였으며 실시간 부품 가격 데이터와 실제 판매 페이지를 얻기 위해 네이버 쇼핑 API를 사용하였다.

2-2. Principal Functions

1. Swing을 활용한 컴퓨터 부품 조회기능

처음에 구현하려고 했던 것은 브라우저상에서의 사용자 컴퓨터 부품의 조회였지만, 브라우저상에서 부품 정보를 조회하는 것은 보안 문제가 있으므로 지원하지 않는다. CPU와 RAM의 정보를 단편적으로 얻을 수 있으나 이마저도 현재 사용 가능한 양만 얻을 수 있고 완전하게 얻을 수 있는 것은 GPU의 정보뿐이기 때문에 현재 부품의 정보를 가져올 때 브라우저를 활용하기에는 어려움이 있어 Java Swing을 이용해 외부 프로그램을 만들어서 이 프로그램을 통해 조회한 내용을 데이터베이스에 추가하는 방식으로 구현하였다.

```
Process process = Runtime.getRuntime().exec("wmic path " + wmicClass + " get " + property);
BufferedReader reader = new BufferedReader(new InputStreamReader( in:process.getInputStream()));
StringBuilder hardwareBuilder = new StringBuilder();
while ((line = reader.readLine()) != null) {
    if (line.contains( s:property)) {
        continue;
    }
    String hardware = line.trim(); // 공백 제거
    if (!hardware.isEmpty()) { // 빈 문자열은 추가하지 않음
        if (hardwareBuilder.length() > 0) { // 이미 하드웨어 정보가 추가된 경우
            hardwareBuilder.append( str:"/");
        }
        hardwareBuilder.append( str:hardware);
    }
}
reader.close();
return hardwareBuilder.toString();
```

Fig. 2. Wmic와 Process 클래스를 이용한 부품정보 조회 코드

이러한 부품의 조회는 Microsoft Windows 운영 체제에서 사용되는 WMI (Windows Management Instrumentation) 제공자[3]를 사용해서 하드웨어 부품 정보를 가져오는 방식으로 구현하였다. Process 클래스를 사용해서 wmic을 실행하고 부품별 클래스를 조회해서 이를 HashMap에 부품 종류를 Key로 부품 이름을 Value 등록하는 코드이다. 이 과정에서 Ram과 같이 여러 개가 존재하는 부품들은 "/"를 통해 구분하기 위해 사이에 "/"를 추가하여 구현하였다. 이는 문자열의 수정이 많아지게 되어 String을 +연산으로 붙이지 않고 StringBuilder를 사용하여 최적화를 진행하였다. StringBuilder는 문자열 연산이 이루어질 때 새로운 객체를 생성하지 않고 버퍼 내에서 수정이 발생하기 때문에 문자열 연산이 많을 때 최적화를 위해 사용하는 객체이다.

2. 견적 추가를 위한 크롤링 기능

사용자가 견적을 만들기 위해 여러 가지 부품들에 대한 정보를 얻기 위해 API를 사용하는 방법을 설계하였으나, 부품 호환 여부를 체크하는 기능을 구현하기 위한 정보가 있는 API가 없으므로 아마존에서 부품 정보를 크롤링 해오는 것으로 변경하였다. 아마존은 JavaScript를 사용하여 동적으로 페이지를 생성하기 때문에 Selenium을 사용하여 크롤링을 진행하였고 Xpath를 이용해 위치를 지정해서 필요한 정보들을 추출하였다.

```
elements = driver.findElements(By.xpath( xpath ));
"//ul[@class='a-unordered-list a-vertical a-spacing-mini']
                                    //li//span[@class='a-list-item']"
```

Fig. 3. 클래스와 Xpath를 이용해 크롤링을 지정하는 코드

3. 부품 조회기능 설명

부품 조회 페이지는 설계단계에서 부품을 검색한다는 것은 전 부품들을 대상으로 검색해야 해서 1차적으로 카테고리에서 나뉜 부품들을 대상으로 검색하는 것으로 처리하였다 처음 페이지에 들어갈 때, 오버헤드가 크게 발생하지만 검색하고 부품을 비교하는 동안은 DB에 부담을 주지 않는다는 장점이 있다.

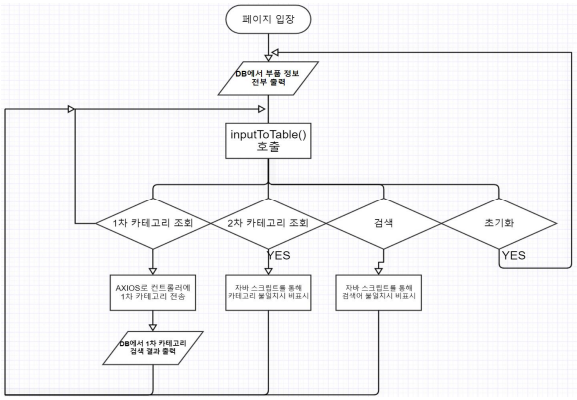

Fig. 4. 부품 조회 플로우차트

JavaScript를 활용하여 검색 결과에 따라서 페이징 또한 구현하였으며 카테고리별로 분류 후 검색 또한 가능하게 구현하였다.

4. 네이버 API 가격 조회기능

각 부품의 정보를 크롤링으로 가져왔지만, 아마존의 달러화 가격을 그대로 사용할 수 없고 사용할 때마다 아마존을 다시 크롤링할 수 없어서 네이버 쇼핑 API를 활용하여 부품의 이름을 매개로 네이버 쇼핑에서 가격을 가져오도록 구현하였다.

```
String apiURL =
"https://openapi.naver.com/v1/search/shop.json
?query=" + text+ "&display=3&exclude=used:cbshop";
```

Fig. 5. 네이버 쇼핑 API 조회 부분 코드

그림5는 ajax 요청을 통해 가져온 정보인 부품 가격을 반환해주는 컨트롤러 부분 코드이다. 매개변수를 통해 정확도 기준으로 3개의 검색 결과를 가져오도록 구현하였으며, 반환된 3개의 가격 중에서 가장 가격이 낮은 한 개의 가격만 남게 IF문을 사용하여 최저가를 검색하도록 구현하였다. 만약 네이버 쇼핑에 등록되어 있지 않아 조회된 값이 없으면 10,000,000의 초깃값이 그대로 남아있도록 해서 출력하기 전의 값을 체크해서 만약 초기값이라면 조회된 값이 없음 또한 출력할 수 있게 구현하였다.

```
function getData(searchText) {
  searchText = changeSearchText(searchText);

  function NaverItem(lowestPrice, lowestPriceLink) {
    this.lowestPrice = lowestPrice;
    this.lowestPriceLink = lowestPriceLink;
  }

  let itemInfo = new NaverItem(10000000, '#');

  return axios.get('/search/shop', {params: {query: searchText}})
    .then(response => {
      let items = response.data.items;
      items.forEach(item => {
        if (item.lprice < itemInfo.lowestPrice) {
          itemInfo.lowestPrice = item.lprice;
          itemInfo.lowestPriceLink = item.link;
        }
      });
      return itemInfo; // Promise 객체의 resolve 함수로 itemInfo 객체를 반환
    })
    .catch(error => {
      console.error(error);
      return itemInfo; // Promise 객체의 reject 함수로 itemInfo 객체를 반환
    });
}
```

Fig. 6. ajax를 통한 조회 후 최저가 선정

그림6의 getData()와 조회된 데이터를 테이블에 추가하는 함수인 그림7의 inputToTable()또한 비동기 통신을 통해 DB에서 정보를 조회하기 때문에 두 가지 비동기 통신 간의 순서를 지정해줘야 한다. 왜냐하면 데이터를 입력해주는 inputToTable()의 실행이 끝내기전에 getData()에서 행의 정보를 가져와서 검색하려고 하면 데이터 입력이 되어있지 않아 오류가 발생하기 때문이다.

```
async function inputToTableBody(row, tableBody, info) {
  if (info.cpuName.length <= MAX_CPUNAME_LENGTH) {
    row.insertCell(0).innerHTML = `<a href="${info.amazon_Link}">${info.cpuName}</a>`;
  } else {
    row.insertCell(0).innerHTML = `<a href="${info.amazon_Link}">${info.cpuName.substri
  }
  row.insertCell(1).innerHTML = info.brand;
  row.insertCell(2).innerHTML = info.cpuModel;
  row.insertCell(3).innerHTML = info.cpuSpeed;
  row.insertCell(4).innerHTML = info.cpu_socket;

  const priceCell = row.insertCell(5);
  priceCell.innerHTML = '로딩 중...'; // 가격 정보가 로드되기 전에 표시할 텍스트

  row.insertCell(6).innerHTML = `<a href="${info.amazon_Link}"><img src="${info.amazon_
  row.insertCell(7).innerHTML = '<a class="btn btn-primary" href="/bbs/updateEstimate?i

  const ItemInfo = await getData(info.cpuName); // 가격 정보 가져오기
```

Fig. 7. async와 await을 통한 비동기 통신 순서 지정

이 비동기 함수간의 순서를 지정할 때 async와 await을 활용하여 구현하였다. 콜백함수를 사용하는 방법도 있었지만 async/await은 무한 콜백을 방지할 수 있고 코드의 진행방식을 더욱 간결하고 직관적으로 표현하기 때문이다. 비동기 함수 간의 순서를 지정해서 아직 입력되지 않은 데이터의 가격 조회를 방지하였다. 또한 네이버 API에서 단순 문자열로 전달된 가격을 사용자가 쉽게 읽을 수 있도록 JavaScript의 Intl API를 활용해 구현한 formatCurrency()란 간단한 함수로 문자에서 화폐로의 서식 변환을 처리해주었다.

```
const currency = new Intl.NumberFormat('ko-KR',
                                {style: 'currency', currency: 'KRW' });
```

Fig. 8. Intl API를 활용한 서식 변환

5. 견적 정보 저장기능

견적 정보를 게시판 이용, 부품 조회를 하는 와중에도 항상 저장해야 하므로 견적 정보를 클래스 형태로 Session에 저장하는 방식으로 구현하였다. 클래스에는 각 부품의 이름만 저장되고 이 부품의 이름을 이용해서 각 부품의 테이블에 접근해서 부품 정보를 읽어오도록 구현하였다. 이를 통해 다른 페이지로 이동하여도 견적의 정보를 session에 계속 저장하고 있어서 계속 서비스에서 값을 이용할 수 있다.

6. 견적 호환 여부 체크 기능

견적에 정보가 저장된 Session에서 저장된 부품들의 정보와 지금 추가하려는 부품의 정보를 비교하여 호환 여부를 체크하여 JavaScript의 Alert 함수를 이용하여 사용자에게 부품 호환 여부를 알려주는 기능을 구현하였다.

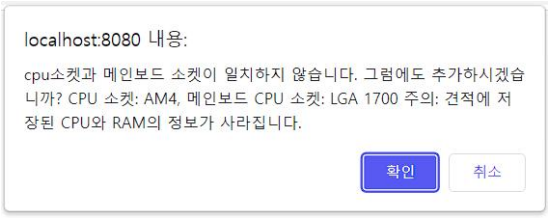

Fig. 9. 호환여부 체크 알림 메세지

7. 게시판 기능

견적들을 이용해서 글을 작성할 수 있는 기능 또한 구현하였다. 견적에 대한 정보와 대화로 주고받은 정보가 최대한 서로 같이 노출되는 것을 중점으로 개발을 진행하였다. 저장되어있는 견적을 이용하여 바로 글을 작성할 수 있고 견적 없이 바로 글을 작성할 수도 있다. 이러한 게시판에서 사용자는 대화방에 들어온 것처럼 서로 댓글을 통해 대화를 나눌 수 있다. 처음에는 글의 본문만 보이지만 본문을 클릭 시 대화창처럼 커지게 되어 마치 채팅방처럼 여러 사용자의 의견을 쉽게 확인할 수가 있다. 또한 부품을 댓글에 첨부하는 것도 가능하다. 부품 상세 조회 페이지로 연결된 이미지가 댓글에 추가되며 이렇게 부품 정보를 댓글에 시각적으로 표시함으로써 더욱 사용자 간의 직관적인 대화를 가능하게 하였다. 또한, 게시글의 견적에서 부품을 내 견적으로 가져오거나 이 견적에서 사용된 부품의 정보를 조회할 수 있다.

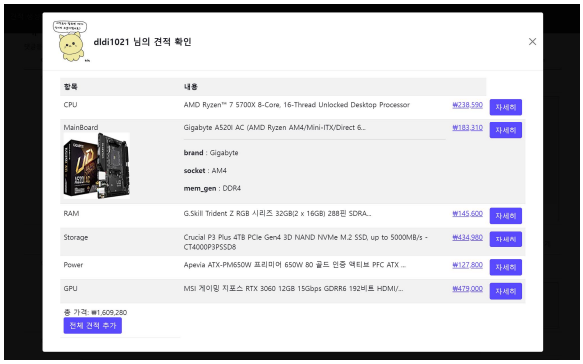

Fig. 10. 게시판의 견적 출력 모달

8. Pageable 클래스를 이용한 Page 구현

JPA(Java Persistence API)에는 Pageable이란 클래스가 있어서 페이징의 구현이 매우 간결하다. PageRequest 객체를 생성할 때 페이지 위치, 페이지 크기, 정렬 기준이 될 컬럼명, 오름차/내림차순 정렬 여부들만 설정하면 자동으로 페이징이 된 데이터를 Page 자료형으로 반환한다. 이 Page 자료형에는 Count 질의를 통해 전달된 데이터의 개수 또한 들어있어서 이 정보를 통해 더욱 쉬운 페이징이 가능하다.

```
PageRequest pageRequest = PageRequest.of(page,
                                pageSize,
                                Sort.by("writeDate").descending());
```

Fig. 11. 페이징 컨트롤러 부분

페이징 구현을 위해 제공하는 Slice라는 자료형을 사용할 수도 있었는데 Page 자료형과는 달리 생성 시 Count 질의를 날리지 않고 다음 페이지가 존재하는지 아닌지만 반환하기 때문에 무한 스크롤 등을 구현할 때 더욱 유리한 자료형이라 Page를 사용하는 방식으로 개발을 진행하였다. 이처럼 JPA에서 지원하는 자료형과 클래스 덕분에 더욱 쉽게 페이징 구현이 가능했다.

III. Conclusions

컴퓨터 견적을 맞추기 위해 여러 가지 부품들의 정보를 아는 것은 중요한 일이다. 컴퓨터의 특성상 대체로 수년간 사용할 제품이므로 한번 구매할 때 자신이 필요로 하는 단계의 제품을 신중히 구매해야 한다. 또한 무조건 새 컴퓨터를 구매하는 것이 아니라 호환 여부를 파악해서 필요한 부품만 바꿀 수도 있어 합리적인 소비를 하기 위해서는 이러한 지식과 정보를 가지고 있어야 한다. 이러한 정보가 필요한 사용자들에게 본 웹 사이트가 합리적인 소비에 도움이 되길 기대한다.

REFERENCES

[1] https://www.index.go.kr/unity/potal/main/EachDtlPageDetail.do?idx_cd=1345
[2] https://www.selenium.dev/
[3] https://learn.microsoft.com/ko-kr/windows/win32/wmisdk/wmic