



경험정리 - 옮기기 완료

🕒 Created	@2024년 5월 9일 오전 10:32
🏷️ Tags	

burgerput Project

업무 역량과 커뮤니케이션 능력에 대한 그게 있어야 함

개요 :

프로젝트에 대한 간략한 설명 Burgerput Proejct를 하며 생긴 경험들을 정리해서 표로 만들 것이다.

▼ 이용자의 서비스 경험 개선 - 로딩 시간 줄이기

시스템 로직 중 하루에 한 번 외부 웹 페이지의 데이터 정보를 가져오는 작업이 필요했습니다.

기존의 로직은 사용자가 웹 페이지에 처음 진입하는 순간 로딩이 수행되고 해당 결과를 cookie에 저장해 로딩의 여부를 저장했습니다.

하지만 로딩의 속도가 5분 정도 소요되었기 때문에 배포 전 이용자는 불편을 느꼈습니다.

문제해결

서비스 경험 개선 시도

사용자의 피드백 수용

Action	<ol style="list-style-type: none">1. 이용자가 웹 페이지에 진입했을 때 로딩이 실행되는 것이 아닌 매일 아침 8시 35분에 서버에서 curl메세지를 이용해 로딩페이지 진입 후 로딩작업을 수행했습니다.2. 로딩로직 리팩토링을 수행했습니다.
--------	---

Result	1. 이용자는 로딩을 기다리지 않고 원하는 시간에 로딩없이 바로 서비스를 이용할 수 있었다. 2. 리팩토링을 통해 로딩의 속도를 120초에서 16초 8배 개선할 수 있었습니다.
배운점	1. 사용자의 피드백을 통해 서비스를 개선하고 이용이 편해졌다는 긍정적인 결과를 얻을 수 있었습니다. 2. 로직을 리팩토링하면서 목표를 위한 사전조사와 계획이 중요하다는 것을 깨달았습니다.

▼ 꼬리물기 질문들

로딩 로직 리팩토링을 어떻게 수행했나?

웹을 조작할 수 있는 셀레니움 라이브러리에서 웹 페이지가 모두 로딩될 때까지 대기 시간을 주는 메서드가 있었습니다.

해당 메서드는 20초로 지정되어 있었고 실제 로딩을 수행할 때 의도한 요소가 아닌 곳을 조사하게 되면 원하는 요소를 찾지 못해 20초 대기 후 요소를 찾을 수 없다는 에러가 발생했었습니다.

따라서 로직을 원하는 요소가 있는 곳만 조사할 수 있도록 웹 페이지를 분석해서 적용 시켰습니다.

? > 에러가 발생했다고 하는데 어떻게 처리를 했나?

요소를 검사하다가 noSuchElements에러가 발생하면 에러 메시지만 log로 띄우고 아무것도 반환하지 않도록 설정했습니다.

반환받은 값에 아무것도 없다면 데이터를 저장하지 않는 방식으로 에러를 처리했습니다.

매일 아침 로딩은 어떻게 수행했나?

curl명령어가 담긴 스크립트 파일을 crontab 설정을 통해 매일 아침 8시 35분에 수행되도록 설정했습니다.

? > crontab이 무엇인지 아는가?
스케줄링을 관리하는 프로그램입니다. 특정시간대에 사용자가 작성한 명령어나, 스크립트를 실행합니다. 반복적인 업무를 등록할때 활용하면 좋습니다.

? > curl이 무엇인가?
url를 사용해서 서버와 데이터를 주고받는 명령어 툴입니다. 프로젝트에서는 클라이언트에서 로딩을 수행하면 사용자가 로딩이 완료될 때 까지 기다려야 했습니다. 따라서 curl명령어를 이용해서 서버에서 요청하도록 수행했습니다.

웹 페이지에 처음 진입하는 순간 로딩이 실행되게 했다는데 어떻게 구현했었나?

기존의 로직은 cookie에 날짜를 넣는 것이었습니다. 쿠키에 저장된 날짜가 오늘과 같다면 로딩을 수행하지 않고 날짜가 다르면 로딩을 수행했습니다.

? > 결과를 cookie에 저장했다고 했는데 자세히 설명해줄 수 있나?


? > CORS문제를 마주한 적이 있나?
로컬 테스트를 할 때 마주한 적이 있습니다. 로컬 테스트를 할 때는

? > CORS란 무엇인가?

내 벨로그 링크

Loading Logic 리팩토링결과

속도를 8배나 개선했어요 너무 좋고~~

 <https://velog.io/@bbubboru22/Loading-Logic-리팩토링결과>

Burgerput Loading Logic Refactoring

유지보수 중

▼ 메모리 부족현상으로 인한 주기적 process kill - 다시 작성해야됨

🤔 Loading failed - process 과다문제

와랄라 입니다.

문제해결

서비스 경험 개선 시도

사용자의 피드백 수용


Action	이용자가 웹 페이지에 진입했을 때 로딩이 실행되는 것이 아닌 매일 아침 8시 35분에 서버에서 curl메세지를 이용해 로딩페이지 진입후 로딩작업을 수행했다. 또한 로딩로직 리팩토링을 통해 로딩의 속도를 120초에서 16초로 약 8배나 줄일 수 있었습니다.
Result	이용자는 이용을 원하는 시간에 로딩없이 바로 서비스를 이용할 수 있었다.
배운점	사용자의 피드백을 통해 서비스를 개선하고 이용이 편해졌다는 긍정적인 결과를 얻을 수 있었다. 로직을 리팩토링하면서 목표를 위한 사전조사와 계획이 중요하다는 것을 깨달았다.

▼ 꼬리물기 질문들

내 벨로그 링크

Loading Logic 리팩토링결과

속도를 8배나 개선했어요 너무 좋고~~

 <https://velog.io/@bbubboru22/Loading-Logic-리팩토링결과>

Burgerput Loading Logic Refactoring

▼ Domain에서 @ID 애노테이션 수정하기 - 작성중

 [fix] JPA DTO @ID 수정

와랄라 입니다.

문제해결

서비스 경험 개선 시도

사용자의 피드백 수용


Action	이용자가 웹 페이지에 진입했을 때 로딩이 실행되는 것이 아닌 매일 아침 8시 35분에 서버에서 curl메세지를 이용해 로딩페이지 진입후 로딩작업을 수행했다. 또한 로딩로직 리팩토링을 통해 로딩의 속도를 120초에서 16초로 약 8배나 줄일 수 있었습니다.
Result	이용자는 이용을 원하는 시간에 로딩없이 바로 서비스를 이용할 수 있었다.
배운점	사용자의 피드백을 통해 서비스를 개선하고 이용이 편해졌다는 긍정적인 결과를 얻을 수 있었다. 로직을 리팩토링하면서 목표를 위한 사전조사와 계획이 중요하다는 것을 깨달았다.

▼ 꼬리물기 질문들

내 벨로그 링크

Loading Logic 리팩토링결과

속도를 8배나 개선했어요 너무 좋고~~

 <https://velog.io/@bbubboru22/Loading-Logic-리팩토링결과>

Burgerput Loading Logic Refactoring

▼ 컨트롤러의 역할 분리 - 다시 작성해야됨

● Machine과 Food 로딩결과 통합, JSON 데이터 저장, LoadingRefactoring
와랄라 입니다.

문제해결

서비스 경험 개선 시도

사용자의 피드백 수용

Action	이용자가 웹 페이지에 진입했을 때 로딩이 실행되는 것이 아닌 매일 아침 8시 35분에 서버에서 curl메세지를 이용해 로딩페이지 진입후 로딩작업을 수행했다. 또한 로딩로직 리팩토링을 통해 로딩의 속도를 120초에서 16초로 약 8배나 줄일 수 있었습니다.
Result	이용자는 이용을 원하는 시간에 로딩없이 바로 서비스를 이용할 수 있었다.
배운점	사용자의 피드백을 통해 서비스를 개선하고 이용이 편해졌다는 긍정적인 결과를 얻을 수 있었다. 로직을 리팩토링하면서 목표를 위한 사전조사와 계획이 중요하다는 것을 깨달았다.

▼ 꼬리물기 질문들

내 벨로그 링크

Loading Logic 리팩토링결과

속도를 8배나 개선했어요 너무 좋고~~

<https://velog.io/@bbubboru22/Loading-Logic-리팩토링결과>

Burgerput Loading Logic Refactoring

▼ 작성 가이드 - STAR기법 사용

어필 역량/ 주요 경험

Situation, Action, Result

Situation - 언제 무엇을 했는지, 어떤 일이 있어났는지, 거기서 내가 맡은 역할은 무엇인지, 해결해야 할 과제 문제는 무엇이었는지

Action - 그 문제 상황에서 내가 조치한 행동, 다른이와 차별화된 점은? 일을 수행하는데 우선순위는 어떻게 세웠나?

Result - 그 결과 어떤 성과를 이뤘나(객관적 사실, 정량적 수치 이용), 주변의 반응은 어땠나? 다시 한다면 어떻게 다르게 하고 싶나?

배운점 - 입사후 업무 수행 시 적용 가능한 점은 무엇이다. 이경험을 통해서 ~~를 깨달았다.

[취업고민/자소서] Step2. 경험정리표 완성하기

꿈리의 합격 자소서 완성을 위한 7가지 단계 Step1. 스펙이력표 완성하기 Step2. 경험정리표 완성하기 S...

<https://m.blog.naver.com/flipstory4u/221525726766>

세명에서 가장 소중한 분야의 경험정리표				
분야	상황 (어떤 일을 했는지)	상황 (어떤 일을 했는지)	상황 (어떤 일을 했는지)	상황 (어떤 일을 했는지)
1	상황 (어떤 일을 했는지)	상황 (어떤 일을 했는지)	상황 (어떤 일을 했는지)	상황 (어떤 일을 했는지)
2	상황 (어떤 일을 했는지)	상황 (어떤 일을 했는지)	상황 (어떤 일을 했는지)	상황 (어떤 일을 했는지)

나는 Task를 situation이랑 합쳤다. 어느정도 세분화가 진행된 상태에서 수행하기 때문에 situation이 task와 동일한 기능을 수행했기 때문

▼ 이용자의 서비스 경험 개선 - 로딩 시간 줄이기

시스템 로직 중 하루에 한 번 외부 웹 페이지의 데이터 정보를 가져오는 작업이 필요했습니다.

기존의 로직은 사용자가 웹 페이지에 처음 진입하는 순간 로딩이 수행되고 해당 결과를 cookie에 저장해 로딩의 여부를 저장했습니다.

하지만 로딩의 속도가 5분 정도 소요되었기 때문에 배포 전 이용자는 불편을 느꼈습니다.

문제해결

서비스 경험 개선 시도

사용자의 피드백 수용

Action	이용자가 웹 페이지에 진입했을 때 로딩이 실행되는 것이 아닌 매일 아침 8시 35분에 서버에서 curl메세지를 이용해 로딩페이지 진입후 로딩작업을 수행했다. 또한 로딩로직 리팩토링을 통해 로딩의 속도를 120초에서 16초로 약 8배나 줄일 수 있었습니다.
Result	이용자는 이용을 원하는 시간에 로딩없이 바로 서비스를 이용할 수 있었다.
배운점	사용자의 피드백을 통해 서비스를 개선하고 이용이 편해졌다는 긍정적인 결과를 얻을 수 있었다. 로직을 리팩토링하면서 목표를 위한 사전조사와 계획이 중요하다는 것을 깨달았다.

▼ 꼬리물기 질문들

로딩 로직 리팩토링을 어떻게 수행했나?

웹을 조작할 수 있는 셀레니움 라이브러리에서 웹 페이지가 모두 로딩될 때까지 대기시간을 주는 메서드가 있었습니다.

해당 메서드는 10초로 지정되어 있었고 실제 로딩을 수행할 때 의도한 요소가 아닌 곳을 조사하게 되면 원하는 요소를 찾지 못해 10초 대기 후 요소를 찾을 수 없다는 에러가 발생했었습니다.

따라서 로직을 원하는 요소가 있는 곳만 조사할 수 있도록 웹 페이지를 분석해서 적용시켰습니다.

? > 웹 페이지를 어떻게 분석한건가?

웹 페이지는 ul li태그로 구성되어 있었고 모든 li태그는 동일한 class를 사용하고 있었지만 ul태그에서 원하는 요소가 담긴 부분은 특정 클래스를 사용하고 있었습니다.

기존에 모든 li를 검사하던 코드를 필요한 요소들이 담긴 ul태그를 골라낸 후 그 안에 있는 li태그만을 검사했습니다.

? > 에러가 발생했다고 하는데 어떻게 처리를 했나?

요소를 검사하다가 noSuchElements에러가 발생하면 에러 메세지만 log로 띄우고 아무것도 반환하지 않도록 설정했습니다.

반환받은 값에 아무것도 없다면 데이터를 저장하지 않는 방식으로 에러를 처리했습니다.

매일 아침 로딩은 어떻게 수행했나

crontab 설정과 curl설정

웹 페이지에 처음 진입하는 순간 로딩이 실행되게 했다는데 어떻게 구현했었나?

cookie값을 검사해서 쿠키에 오늘 날짜가 들어있지 않으면 로딩이 실행되도록 설정했다.


결과를 cookie에 저장했다고 했는데 자세히 설명해줄 수 있나?

다음에 적기

내 벨로그 링크

Loading Logic 리팩토링결과

속도를 8배나 개선했어요 너무 좋고~~

 <https://velog.io/@bbubboru22/Loading-Logic-리팩토링결과>

**Burgerput
Loading Logic
Refactoring**