Capstone 발표

1page

안녕하세요 이번 코스타 sw300의 신입사원 교육의 결과 발표를 맡게된 노승업이라고 합니다

(인사)

-10sec

2page

간단하게 자기소개를 하고 넘어가도록 하겠습니다.

(애니메이션 종료 후)

(다음 애니메이션)

현재 한국산업기술대학교 ICT융합공학과의 SW트렉에서 공부하고 있으며 비타팀에서 근무하고있습니다.

(다음 애니메이션)

대학교 오기 전 고등학교에서부터 파이썬을 처음으로 시작하여서 주력언어로 사용하고 있으며,

회사 입사하여 C++를 기반으로 MFC를 공부하였고 이번 교육을 하는 동안에는 자바를 중심으로 교육을 들었습 니다.

이번 교육이 끝나고 현재는 C++언어와, 메켄토시에서 동작하는 스위프트 언어를 공부하고 있는 중입니다.

목차를 간단하게 설명드리고 발표 진행하도록 하겠습니다.

3page

첫번째로 교육소득 두번쨔로는 캡스톤 프로젝트 마지막으로 앞으로의 계획 간단하게 말씀드리고 발표 마치도록 하겠습니다.

(애니매이션 종료 후)
그러면 교육 소득에 대해 말씀드리겠습니다.

4페이지

저는 교육소득을 두가지 관점에서 찾아봤습니다.

원 내부에 저 노승업이 있고, 원 밖에는 제 3자가 저를 보는 관점에서는 어떤 교육소득이 있을까 입니다.

5페이지

이번 교육 소득에 있어, 어떤것을 얻었냐고 하신다면, 수료증이 아닐까 싶은데요,

수료증이라 하면 공부, 시험, 성적의 순서로 수료증을 취득할 수 있기 때문에, 아무래도 객관적으로 판단할 수 있 는 큰 축이 아닐까 싶습니다.

그러면, 이번 교육을 통해서 어떤 수료증을 얻었는지 알아보도록 하겠습니다.

6페이지

최 상단에 있는 SW300 밑으로 수료증이 있고 그 밑으로 이 과정을 수료하면서 얻은 9개의 수료증이 있습니다.

왼쪽부터 차례대로 데이터 베이스 서버에서 사용하는 SQL 학습시에는 AWS에서 사용하는 RDS를 사용하였고, MySQL 플렛폼을 사용하여 수업을 진행하여 취득하였습니다.

그 다음으로 소프트웨어 엔지니어링을 수료하였습니다. 소프트웨어 라이프사이클, 버전 컨트롤, oop, 즉 객체 지향 프로그래밍에 대해서 알아보는 시간등 간단하게 소프트웨어의 전반적인 프로세스들을 확인하는 시간을 가 졌습니다.

그 다음으로는 명세서를 통하여 다이어그램을 만들고 이를 이용하여 Class다이어그램등 추가로 도출하는 등,
UML을 사용하여 소프트웨어를 설계에 대해서 배웠고,

객체지향 설계를 안드로이드 어플리케이션을 만들어보고 설계를 통하여 소프트웨어 구조에 대한 수료증을 이수 하였습니다.

그리고 C++의 기초를 간단하게 알아보았고,

OS Power User라고 해서 Unix환경에서 쉘을 이용한 오류 찾고 수정하기, 윈도우에서는 파워쉘의 기초적인 이용방법, GUI 환경에서의 서버 관리 방법에 대해서 알아보는 시간을 가졌습니다.

그 다음으로 JAVA를 사용하여 백엔드 구축을 스프링 프레임워크를 통하여 구축하는 시간을 가졌는데요 기본적 인 REST API가 무엇인지, 어떻게 사용하는지, POST, GET, DELETE, UPDATE 등등에 대하여 알아보는 시 간을 가졌습니다.

이 다음으로는 HTML5, CSS에 대하여 기초적인 과정을 배우게 되었고, 그 다음으로는 자연스럽게 자바스크립 트로 들어가 HTML, CSS 수료증을 받고 이어서 자바스크립트 수료증을 받아 총 9개의 수료증을 이수하였습니 다.

요약하자면,

7페이지

이러한 테크닉을 배움으로서 전반적인 개발 프로세스에 대해 알아가는 좋은 시간이었습니다.
아는 내용이라 했더라도 현업에서 종사하시는 분들이 알려주고 질문의 기회도 굉장히 많아서

실제 대학교에서 프로젝트를 할 때 매우 큰 도움이 되었다는게 실제로 채감되었을 정도로 뜻깊은 시간을 가졌던 것 같습니다.

8페이지

이제는 외부적인게 아닌 내부적, 저의 내면에서 바뀐점이 있다면 어떤게 있을까에 대해 알려드리고자 합니다.

9페이지
지식이 늘었다, 이렇게 들어갈 수도 있었지만 확실하게 말씀드릴 수 있는건 아무래도

(애니메)

저의 경험에 대해 말씀드리는게 잘 말씀드릴 수 있을 것 같아서 경험에 대하여 말씀드리겠습니다.

10페이지

현재 개발직으로 들어와있는 제가 생각하기에
개발자는 머리와 컴퓨터로 유의미한 코드를 만드는 사람이라고 생각을 합니다.

11페이지
그렇다면 개발자는 이 코드를 어떻게 관리를 하여야할까요?

바로 첫번째는 save 저장이라고 생각합니다.

12페이지

휘발성 메모리에 올라가있는 저희의 코드를 파일로서 저장하여 어딘가에 저장을 잘 해놓는게 가장 중요하지 않나 싶습니다.

파일은 여려명이서 사용가능해야하며 공개 또는 비공개로 만들 수 있어야합니다.

13페이지

어떤 개발자는 이런 형식으로 저장을 할 수 있습니다.

하지만 이런 형식의 저장 방식은 위험부담이 큽니다.

실제로 교육중 유닉스 환경의 실습에서 저희가 겪은 비하인드 스토리가 있습니다. rm -rf 명령어를 혹시 아시는 분이 있으신가요? 실습을 하다가 실제로 데스크탑 폴더를 날려버리신 분들도 여럿 있었습니다. 이 명령어로 삭제할 경우 복구가 힘들기 때문에 소스코드가 날라갔다면 사실상 복구는 깃허브나, svn내부에 있는 코드를 다시 받아서 사용해야할 것입니다.

(대충 설명)

14페이지

그렇기 때문에, 저희는 이중 이상으로 어딘가에 저장해야 좀 안전하다고 볼 수 있다고 생각합니다.
단순 백업용도 라도, 깃과 깃허브를 이용해서 형상관리를 한다거나 하는 식으로,

15 페이지
여기서 교육소득을 마치고 이제 캡스톤 프로젝트에 대해서 발표하도록 하겠습니다

16 페이지
1프로젝트는 만들다보니까 시간상 안될것 같아서 빼버렸고, 개인 프로젝트를 중심으로 발표를 할까합니다.

17 페이지
뷰 js를 사용해서 프런트엔드 개발을 하게 되었고, 여태까지 배웠던 내용 최대한 살려서 해보려고 노력했습니다.

18 페이지
이번에 하기로 한 프런트엔드는 새로고침이 없는 웹 페이지를 만들기로 계획하였습니다.
새로고침을 하게 된다면 웹 페이지의 연속성이 깨진다고 생각하기 때문에 많은 부분에서 유저의 웹 사이트 경험 이 좋아지지 않을 수 있다고 생각했기 때문입니다. 되도록이면 모든 웹 페이지에 대한 컴포넌트를 첫 로딩때, 다 운로드 하고, 그 후에는 유저의 액션에 따른 데이터 값만 서버와 통신하여 얻는 형식으로 제작하고자 하였습니 다.

19 페이지
이번 프런트엔드 프로젝트는 뷰.js라는 프레임워크를 사용해서 개발하게 되었습니다.

20 페이지
대부분의 웹 페이지들이 모델, 뷰, 컨트롤러의 방식인 MVC 모델을 가지고 개발되었지만, 이번에는

MVVM, Model, View, View Model로 이루어진 프런트 엔드를 개발하게 되었고 이를 통하여 새로고침 없는 웹 페이지를 만들 수가 있었습니다.

21 페이지

개발 할 때의 방식은 다음과 같이 개발하게 되는데, 첫번째로 전체적인 웹 페이지의 구상을 먼저 합니다. 그리고

웹 페이지에서 동작을 수행하는 페이지의 각 요소, 컴포넌트를 구현하면 됩니다. 구현채만 존재한다고, 보시면 편하시고 쉽게 말하면,

웹 페이지 답게 내부의 데이터가 들어왔을 때, 그 데이터를 어떻게 표현해줄지에 대해 정의를 한다고 보면 될 것 같습니다.

23 페이지

이렇게 코드를 만들었다면 이제, 프로젝트를 어떤 방식으로 정리를 할지 결정하여야합니다.

vue.js는 프레임워크이기 때문에 기본 템플릿이 존재하는데요, vue.js는 다음과 같은 디렉터리 구조를 가지고 있습니다. 디렉터리의 명칭만 보고 추측한다면 거의 대부분 맞는데,

에셋 폴더에는 웹 페이지의 로고나 사진과 같은 파일들과 같이 정적인 파일들이 들어가게 됩니다.

컴포넌트 폴더에는 전 슬라이드에 보았던 구햔된 웹 페이지의 요소들이 들어가게되고, 약간 깊게 들어가면 각 요소에서 HTML POST, DELETE, UPDATE 통신이 일어나게됩니다.

플러그인 폴더에는 자신이 원하는 플러그인들이 정의되는 공간입니다.

가장 중요한 스토어 폴더에는 글로벌 변수나 함수들이 작성되게 되는데요, 최상위 클래스라고 생각하기면 편합니다. 이곳에서 프런트엔드 개발자들은 컴포넌트에 선언된 변수들을 만들어놓은 getter setter등의 함수를 이용하여 값을 가져오거나, 변경하거나할 수 있습니다.

앱 뷰 파일에는 만들어놓은 요소들을 if-else문을 통해서 상황별료 배치해놓으면 됩니다.

main.js는 특별히 손 댈 필요가 없는 부분이라서 넘기도록 하겠습니다.

24 페이지

이렇게 해서 만들어진 폴더를 vue 커멘트로 빌드를 하게되면 아주 간단한 형태의 폴더가 하나 만들어지게 됩니다. 컴파일되어있는 정적 웹 사이트라고 볼 수 있는데요, 이 폴더를 그대로 웹에 올리기만 하면 이제 프런트엔드의 준비는 끝났다고 보시면 될 것 같습니다.

25 페이지

서버에 해당하는 위치에 프런트를 올리는 기존 방식이 아닌, 서버리스 라고 불리우는 아마존의 정적 웹 호스팅 서비스인 S3에 올리게된다면 비용 감소가 있는 효과 및 서버의 사양에 상관없이 호스팅 할 수 있다는 장점을 가지고 있습니다. S3에 올리게 된다면, 이때 저의 모든 함수들은 잠자게 됩니다. 실제로 동작을 하지 않는 상태로 되는거죠. 그리고 여기서는 함수가 동작되어 실행하게 된다면 그 때, 비용을 부담하는 형태로 제작되어있습니다. 아시는 분은 아시겠지만 AWS의 람다와 비슷한 형태를 띄고 있습니다. 요청 백만건당 0.2달러로 저렴하게 이용할 수 있죠.

또한 인스턴트를 따로 만들어 관리하지 않아도 되기때문에 유지보수 면에서도 신경쓰지 않아도

된다는게 굉장히 편리합니다.

재해와 같은 상황에서도 분산서버에 따로 올라가있기 때문에 서버의 연속성도 유리합니다.

26 페이지

백엔드는 어떠한 방법으로 제작되었는지는 간단하게 설명해드리겠습니다.

27 페이지

만들었던 클래스 다이어그램을 ORM 객체로 변경하여서 비주얼 페러다임의 기능중에 하나인 코드 제너레이터를 사용하여 자바 클래스를 따로 만들 수도 있습니다.

28 페이지

ORM 객체를 살짝 쉽게 표현하면 다음과 같이 표현할 수 있습니다. 유저에는 닉네임 아이디 패스워드가 있고 각 개체마다 고유힌 아이디가 존재하게됩니다.
코멘트에도 코멘트, 어떤 유저가 댓글을 썻는지 알기위해 유저의 아이디, 생성시간 등이 존재하고 스레드 즉, 게시물에는 제목, 만든 유저의 닉네임, 내부 내용, 생성시간, 각 게시물의 고유 아이디 등이 존재합니다.

시간상 모든 기능을 구현하기에는 시간이 너무 촉박하다고 생각해서 유저와 코멘트 부분은 벡엔드에는 구현이 되어있지만 프런트엔드에서는 기능을 구현을 하지 못한게 아쉽습니다.

29 페이지

스프링으로 구현된 벡엔드 서버를 .jar 파일로 빌드를 하게된다면, 내부에 서버를 열어주는 톰켓도 내부에 같이 빌드가 되게됩니다.그러면 간단하게 가상 서버에 자바만 설치하고 명령어 한줄만 입력해준다면 JVM이라는 가상 머신 위에서 빌드가 보장됩니다.

30 페이지

전체적으로 제가 원하는 동작 구성을 짧게 한페이지에 표현해보았는데요,
발표에서 생략된 부분이 많이 있지만 전체적인 구상은 다음과 같습니다. 고객이 웹 페이지에 들어오게된다면 유저의 브라우져에서 기본적인 액션이 해당 페이지의 index.html이라는 웹 페이지를 요청하게됩니다. 그러면 s3에서는 index.html을 주게되는데, 이 웹 페이지에는 if-else로 이루어진 한 페에지에 모든 요소들이 포함된 사이트를 넘겨주게됩니다.

만약 유저가 데이터를 요청하게된다면 브라우져에서 특정 컴포넌트의 데이터를 요청하게됩니다. 서버에 GET 통신을 하여서 데이터를 가져오게되고, 혹은 POST통신을 위해서 서버에 데이터를 추가할 수도 있습니다. 그렇게 데이터가 컴포넌트에 업데이트가 된다면 웹 페이지의 전체 문서는 변경되지 않고, 컴포넌트 내부에 데이터만 변경되므로 페이지의 구조가 변경되거나, 또는 데이터가 추가되어 유저에게 보여지게됩니다.

이는 싱글 페이지 기법으로 페이스북이나, 인스타그램에서 사용되는 방식으로 많은 기업에서 쓰이는 기술이라고 합니다.

31 페이지

이번 프로젝트를 하면서 서버단의 기능을 구현하는건 어렵지 않았지만 프런트엔드에서 구현이 저로서는 굉장히 힘들었다고 생각합니다. 아쉽게도 전체적인 기능이 구현되어있지 않아서

게시판을 생각하고 만들었다가 사이트 자체로만 본다면 그저 공유 To-Do 사이트가 되어버렸습니다.

32 페이지

과유불급, 투머치는 좋지 않다. 저에게 주어진 시간을 최대한 활용할 수 있도록 계획을 좀 구체적으로 짜 놓아야되지 않았나 싶습니다. 2.5베의 법칙이라고 5일의 시간이 주어졌다면 하루 안에 다이어그램부터 프로토타입을 끝내고 이틀동안 백엔드와 프런트 구현이 끝나고 나머지 이틀 동안은 테스트 코드를 짜면서 버그 수정을 하며 끝내야 했었는데 구현 부분에서 디테일을 잡겠다고 기간보다 오랫동안 붙잡았던게 실수였던것 같습니다.

33 페이지

메인, 로그인, 회원가입, 스레드 추가, 검색등 시연

34 페이지

해당 프로젝트를 capstone 기간동안에는 마무리 짓지 못했지만, 개인 시간을 들여 마무리를 짓도록하고, 그 외 새로운 언어에 대한 공부를 넓게 많은 경험을 쌓는것을 목표로 공부하는것이 계획입니다.
친구들과 모여 devops의 프로세스에 맞게 사이트 프로젝트로 새로운 웹 어플리케이션을 추가적으로 개발할 수 있도록 할 것입니다.

이상으로 발표 마치겠습니다.