1. 프로젝트 개요 - 동기,목표,유래
2. 환경,툴 - //완료// → 발표짧
3. 다이어그램 - 곧 완료 → 발표짧
4. 스토리보드
5. 핵심 기술 설명 - AWS-EC2 / AI / webSocket / 구글로그인
6. 시행착오 - 각자
7. 시연

1P (장성현)

안녕하세요 저희는 비트캠프 195기 노트 조 입니다. 최종프로젝트 발표 시작하겠습니다.

2P (장성현)

목차로는, 우선 프로젝트와 저희 팀에 대하여 전반적인 설명을 한 뒤, 프로젝트 개발 환경과 구조에 대해 설명드린 후 시연을 하도록 하겠습니다. 시연을 마친 후에는 저희 팀원들이

공부하고 적용시킨 기술들과 그 과정에서 겪었던 어려움에 대해 말씀드리려 합니다.

3P (장성현)

처음 말씀드렸다시피 저희 팀의 이름은 노트입니다. 노트라는 이름만으로는 저희가 어떤 의미를 담아 팀 이름을 정했는지 알 수 없으실 것 같습니다. 뒤에서 따로 설명하겠지만, 이번 프로젝트는 게임과 관련된 프로젝트이기 때문에 게임의 공지사항이라 할 수 있는 패치노트에서 이름을 따 왔습니다. 또 특이 사항으로, 팀원들 모두가 갤럭시 노트를 사용하고 있습니다. 우연의 일치지만 저희 팀의 이름과는 상관이 없을 겁니다. 아마도요.

4P (장성현)

팀을 결성하고, 팀원들과 이야기를 나눌 때에 모두가 게임을 취미로 즐긴다는 것을 알았습니다. 현재 발표하고 있는 제 개인적으로는, 비트캠프에서 자바를 공부하면서 자바를 활용한 여러 게임 프로젝트들을 봐왔기 때문에 게임개발을 제안했고, 팀원들도 승낙을 해 주제가 결정되었습니다. 그렇게 저희 팀은 웹이라는 점을 이용해 다중 접속이 가능한 게임을 개발하기로 했습니다. 또, 솔로 플레이를 위해 컴퓨터 대전도 공부하기로 했습니다.

5P (장성현)

지금 보고 계신 것은 팀의 간트차트, 일정표입니다. 중간중간 일정도 많이 바뀌고 계획보다 늦어지고 정신 없는 프로젝트였지만, 이렇게 저희가 달려온 일정을 보니 감회가 새로운 것 같습니다.

6P (장성현)

주제를 게임으로 정하고, 어떤 것을 만들어볼까 하던 중 닌텐도 스위치로 발매된 51가지 월드와이드 게임이 눈에 들어왔습니다. 가벼운 게임성을 가진 미니게임들을 모아놓아

언리얼이나 유니티 같은 전문적인 툴을 사용하지 않아도 접근하기 편할 것이라 생각해 모티브로 삼았습니다. 또, 웹 프로젝트를 기획하고 있기 때문에 2000년대에 많이 보였던 야후 꾸러기, 쥬니어 네이버 등의 성공한 웹 게임 예시들을 많이 참고했습니다.

7P (장성현)

프로젝트를 계획함에 있어서, 저희가 해결해야 할 과제는 크게 나누어 세 가지 였습니다.

하나는 AI를 만들어 컴퓨터와의 대전을 가능케 하는 것이고,

또 하나는 참여자가 실시간으로 서로의 정보를 받아와야 하기 때문에 웹소켓으로 연결해 주는 것.

마지막으로 프로젝트를 어디서든 접속할 수 있도록 외부 서버, 그 중에서도

아마존 서버를 연결하는 것이었습니다.

저희 조가 이 세 가지 과제를 무사히 끝마칠 수 있었는 지, 그 사이에 어떠한 과정이 있었을 지를 기대해주시기 바랍니다.

8P (신승엽)

안녕하세요 개발환경 파트를 맡은 신승엽입니다.

저희는 프론트 개발환경으로 비주얼 스튜디오 코드를 사용하였으며, 사용하는 기술로는 HTML과 CSS, javascript를 기본 틀로 사용하며, 라이브러리 jquery, 비동기 처리기술인 ajax, 템플릿엔진인 thymeleaf,그리고 저희의 핵심이 기술인 웹소켓을 채택하여 사용하고 있습니다.

9P (신승엽)

다음은 백엔드입니다.

백엔드 개발환경은 스프링 부트와 메이븐을 사용하고 있으며, 라이브러리인 롬복과 프레임워크인 마이바티스를 사용하였습니다. 그리고 데이터베이스로는 오라클과 통합개발환경인 SQL Developer를 사용하였으며, 서버로는 톰캣과 EC2를 사용하였습니다.

다음으로는 협업 프로그램입니다.

우선 저희는 vscode와 스프링부트를 git과 github연동하여 사용하였고 구글 슬라이드, 스프레드, 독스를 이용하여 정보를 바로바로 공유하였으며, ppt만들때는 카카오 오븐을 같이 사용하였기에 넣어놨습니다.

이 페이지는 저희 프로젝트의 구조도로 이런 형식으로 동작하고 있습니다.

이번 파트는 여기까지입니다. 감사합니다.

25p (신승엽)

이번엔 저희의 핵심 기술중 하나인 AWS를 설명하려 합니다.

AWS란 미국의 IT기업인 만든 클라우드 컴퓨팅 플랫폼으로 그림과같이 아주 다양한 서비스를 하고 있으며 저희는 이 많은서비스중에서 EC2를 사용하였습니다.

이 페이지는 카카오톡의 플랫폼입니다. 우리나라는 AWS와 비슷한 플랫폼이 없을까? 라고 생각하며 찾아보던중 제일 눈에 띄고 익숙하여서 찾아왔습니다.

자 이제 AWS의 서비스중 하나인 EC2에 대하여 설명하겠습니다.

클라이언트가 ip를 통해 접속을 요청한다면 요청에 대하여 접속이 가능한지 EC2가 검사한 뒤 접속이 가능하다면 요청을 응답해 주는 방식입니다. 이 방식은 아마존에서 컴퓨터를 대여해주는 방식이기 때문에 서버컴퓨터를 구매하는 비싼 초기 비용이나 공간이 절약되며, 접근방식도 비교적 간단해 아주 좋은 서비스라고 할 수 있습니다.

28P (신승엽)

하지만 마냥 좋기만 한것은 아닙니다.

컴퓨터 하나를 빌려오는 것이며, 운영체제도 다르기 때문에 아주 여러가지 문제가 생깁니다.. 클라이언트가 요청을 했는데 포트가 다르거나 충돌이나서 실패하거나

요청은 인식하였으나 EC2자체의 인스턴스 오류로 실패하거나

아마존의 서버의 오류가 나서 실패하거나 생각보다 많은 문제가 실패하였습니다. 하지만 저는 실패를 했기에 여러가지 상황에 직면하였고 결론적으로 이렇게 예시를 만들고 그림을 넣어서 설명할수 있게 되었습니다. 오히려 실패가 저를 좀더 끌어올려준 것이라 생각합니다. 음.. 조금 늘어지는 감이 있어서 지루하셨을텐데 그럼에도 들어주셔서 감사합니다. 제 발표는 여기서 끝입니다.

29P (장성현)

처음 AI를 만들고자 했을 땐, 파이썬을 통해 강화학습을 실행하고, 그것을

자이썬을 통해 스프링으로 구현하고자 했습니다. 그러나 파이썬에 익숙하지 않은 상황에서

스스로 학습하는 인공지능을 만드는 것은 난이도가 너무 높았기 때문에, 생각을 조금

바꿔보기로 했습니다.

30P (장성현)

우리가 인공지능이라 부르는 것은 사실 넓은 범위이고, 학습시켜 발전시키는 머신러닝과

인간의 뉴런 구조처럼 만드는 것이 딥 러닝으로, 강화학습은 머신러닝에 속한 개념이었습니다. 이 중 답을 정해주는 지도학습에 영감을 받아 상황에 맞게 행동하는 알고리즘을 만들었습니다. 매 순간 필드의 상황을 분석하여 특정 경우에 행동의 가중치를 잡아주어 최선의 결과를 선택하도록 했습니다.

31P (장성현)

필드는 오목판과 오목판 밸류, 두 가지를 가지고 있습니다. 오목판은 말 그대로 돌이 놓아져 있는 현재 상황을 저장하며 오목판 밸류는 현재 상황에 따른 AI의 다음 선택의 가중치를 저장한 것입니다. 매 턴 AI는 공격 및 방어 가능 여부를 체크하여 밸류 필드를 재설정하고

연산을 모두 마친 뒤 가장 가중치가 높은 곳에 돌을 놓게 됩니다.

32P (장성현)

다음은 프로젝트에 적용된 핵심기술 중 하나인 웹 소켓입니다. 인터넷이 발전하면서 함께 발전된 기술인 웹 소켓은 실시간 양방향 통신이 가능하다는 특징이 있습니다. HTTP를 발전시켰기 때문에 기본포트와 시큐리티 포트 또한 일치합니다. WS를 사용하기 위해선 핸드쉐이크라 불리는 등록 과정이 필요합니다.

33P (장성현)

ajax 또한 서버에 요청을 보내고 응답을 받는 실시간 통신이지만, 웹소켓의 진가는 여기서 발휘됩니다. 여러 번의 요청을 보낼 때 ajax는 한 번의 요청에서 서버에 연결되었다가 응답을 받을 때 연결을 종료하지만, 웹 소켓은 한 번 연결해놓는다면 별도로 연결을 종료할 때 까지 과정을 생략할 수 있어 엄청난 효율을 보이게 됩니다. 그만큼 로딩 시간이나 서버에의 부담도 줄어듭니다.

34P (장성현)

소켓아이오로도 웹소켓을 사용할 수 있지만, 몇 가지 차이가 있습니다. 둘 모두 디펜던시를 추가해서 사용하지만, WS는 서버 주소 입력만으로 생성되는 데 반해 소켓아이오를 사용하기 위해선 별도의 파일을 추가해 설정해주어야 합니다. 또, WS는 데이터를 받을 때 스트링 자료형으로 받고 룸도 기본적으로 생성되지 않아 소켓아이오가 초심자에게는 더 접근성이 좋다고 생각합니다. 하지만 WS는 익숙해졌을 때 활용방면이 많기 때문에 각각의 장단이 있습니다.

35P (장성현)

저희 프로젝트에서는 로비와 게임 방에 입장할 때 자동으로 웹소켓에 연결이 되고, 명령어를 주고받는 방식을 통해 컨트롤러 단과 뷰 단을 연결했습니다. 또, 여러 게임방이 존재할 수 있기 때문에 구별을 위해 채널이라는 이름의 스태틱 해시맵을 만들었습니다.

36P (장성현)

웹소켓을 공부하고 프로젝트에 적용할 때, 공부 시간보다 익히는 시간이 더 오래 걸렸던 것 같습니다. 이론은 정말 간단한데, 적용하려고 하면 문제가 생겼습니다. 그 중 단순히 제 실수가 아닌 오류 두 개를 예를 들어보자면, 보통은 편의를 위해 스프링 내장 서버를 이용해 프로젝트를 점검하지만 웹소켓은 외부 서버에 연결되어 있어야 적용 가능했다는 것입니다. 또, 오토와이어를 적용한 인스턴스가 제대로 인식되지 않아 다른 컨트롤러에 있는 메소드나 매퍼를 사용할 수 없었습니다. 이는 스프링이 자동으로 빈을 연결해주지 못했기 때문으로, 어노테이션을 변경하고 수동으로 처리를 해줌으로서 해결했습니다.

정호/프로젝트 발표 분량: ppt 12p~24p

12p: 저희의 ERD는 총 3테이블로 구성되어있습니다. 회원쪽을 의미하는 MEMBER테이블, 게임방을 의미하는 GAMEROOM테이블, 관리자한정으로만 접근하는 공지사항 NOTICE테이블이 있습니다.

13p: 저희의 유즈케이스 다이어그램입니다. 오른쪽에 있는건 다이어그램을 설명하는 상세기술서입니다.

14p: 저희의 시퀀스 다이어그램은 총 3개입니다. 이메일로 가입하는 회원가입 시퀀스, 그리고 구글 ID로 로그인하는 구글로그인 시퀀스, 게임방에 접속하는 게임방 시퀀스 총 3개로 되어있습니다.

15p: 저희의 클래스 다이어그램입니다. 크게 멤버, 게임룸, 공지, 채팅 dto로 구성되어있습니다.(수정)

16p: 저희 사이트의 페이지 구성도입니다. 처음 접속시 메인페이지가 뜨고, 거기서 버튼으로 회원가입, 로비, 마이페이지로 이동합니다. 로비페이지에서 공지페이지, 게임페이지로 갈 수 있도록 구성하였습니다.

17p: 저희조의 스토리보드 규칙입니다. 기본 글씨체, 크기 같은 것들과, 표현방식과 설명방식, 페이지 규칙들이 있습니다.

18p: 저희조의 문서 버전입니다. 초기 작성에서 1.0버전에서 수정과 내용추가 등을 거쳐 지금의 버전에 이르렀습니다.

19p: 로비 페이지입니다. 룰버튼 클릭시 룰페이지로 이동합니다. 메인버튼을 누르면 아까 보셨던 메인 페이지로 돌아갑니다. 공지버튼을 누르면 공지페이지로 이동하게 됩니다. create버튼을 누르면 방생성 페이지가 뜨게 됩니다. 5번은 방 목록을 보여주는 곳입니다. 이미지가 뜨고 옆에 방이름과 자리여부를 확인할 수 있습니다. 6번은 접속자 목록을 보여줍니다.

20p: 마이페이지입니다. 1번은 유저의 프로필 이미지를 보여주고, 2번은 닉네임을 보여줍니다. 3번은 프로필 선택버튼으로 새로 이미지를 바꾸려 할때 이용할 수 있습니다. 4번에서 나머지는 변경하실수는 없고, 오직 비밀번호만 변경할 수 있게끔 하였습니다. 5번은 업데이트 완료 버튼으로, 누른 후 로비페이지로 이동합니다. 6번의 Reset버튼을 누르면 현재 저장되어있던 닉네임과 패스워드 값을 전부 지우게 됩니다. 7번의 버튼을 누르게 되면, 확인창을 한번 띄우게 되고, 확인을 누르게 되면 회원탈퇴가 됩니다.

21p: 게임페이지입니다. 1번은 뒤로가기 버튼이고, 2번은 오목판, 3번, 4번은 호스트와 게스트의 채팅을 보여줍니다.

22p: 이제 프로젝트 시연을 하겠습니다.

23p: 소셜로그인을 설명하기에 앞서 Oauth인증방식에 대해서 설명하겠습니다. 대부분의 소셜 로그인에서 쓰이는 Oauth인증이란, 웹사이트 계정 인증에 타 서비스 계정을 사용하는 방식을 뜻합니다. Oauth 인증 방식의 과정은 다음과 같습니다. 구성으로는 사용자, 접속서버, 권한을 부여해주는 권한서버, 그리고 소셜로그인 쪽 서버가 있습니다. 먼저 사용자가 접속서버에 인증요청을 하고, 접속서버는 사용자에게 인증수단을 보냅니다. 사용자는 인증을 완료하고 접속서버에 권한허가 요청을 보냅니다. 접속서버는 그 요청을 인증서버에 보내고, 인증서버에서 확인한 후에, 토큰으로 값을 전달합니다. 그러면 서버에서 토큰을 저장했다가, 소셜로그인 서버(카카오, 구글등)에 보내고, 서버에서 토큰이 유효한지 확인후, 접속서버에 자원을 보내는 방식입니다.

24p: 앞에서 설명한 oauth2.0의 로그인 원리를 가지고 하는 소셜 로그인 과정입니다. 사용자가 인증요청을 하고, 인증 후에 권한 요청을 보내고, 인증서버에서 토큰과 유저정보를 받습니다. 그러면 서버에서 DB에 있는 정보를 조회에서 인증을 하게되고 사용자의 인증을 처리하는 과정입니다.