포트폴리오

**교내 프로젝트 - 소프트웨어중심대학 사업단에서 실습이 병행된 프로젝트 수업을 대상으로 진행한 우수 결과물 시상 참여**

|  |  |
| --- | --- |
| **개발 기간** | 2020.11.02 ~ 2020.11.29 (기획 5일, 개발 23일) |
| **개발 환경** | Unity 2020.1.3f1 |
| **개발 인원** | 3명 |
| **요약** | 유니티를 이용하여 키보드와 마우스를 쓰는 1인칭 슈팅게임을 개발함 |
| **기능** | 1. 근접 공격  2. 좀비의 애니메이션, 스크립트, 생성지점  3. 게임 오버 시 페이드아웃, 재시작 |

**1. 개발 기획**

**1-1 게임 기획**

코로나 이슈에 착안하여 바이러스와 백신을 얻는 이야기에 영감을 받음

단순히 백신을 얻는 것이 아니라, 그 과정 속에서 외부의 위협에 대항하여 맞서는 방식을 착안

FPS방식을 선택하여 좀비(바이러스)가 퍼진 음침한 도시에서 백신을 얻으면 승리하는 게임을 기획

**1-2 개발 계획**

개발 인원수가 3명인 이점을 살려서 역할을 분담함(크게 맵 제작, 플레이어 제작, 좀비 제작으로 나눔)

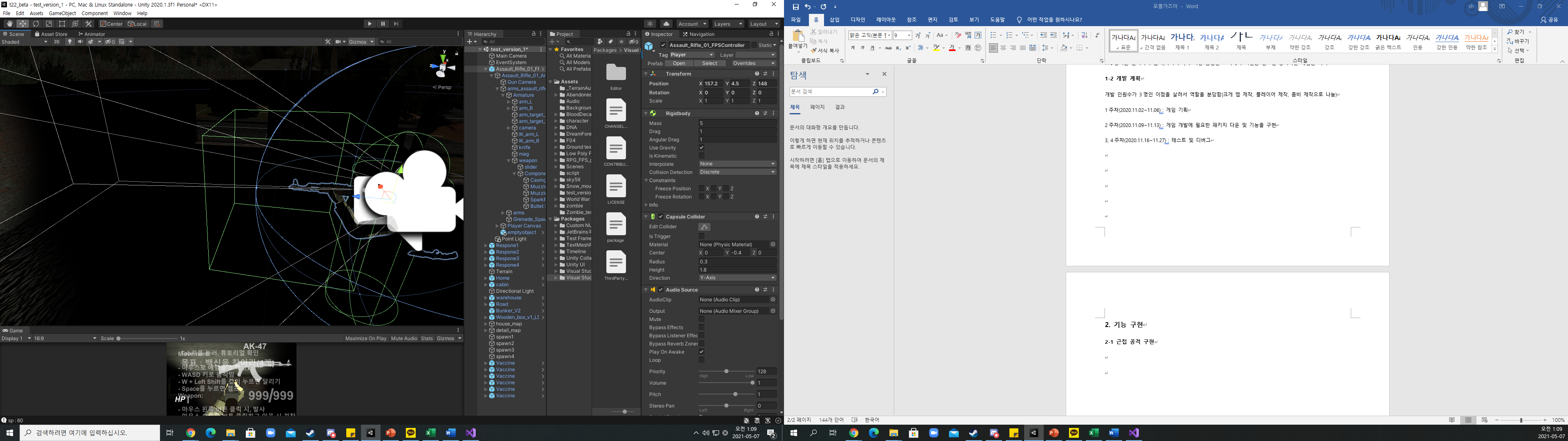
1주차(2020.11.02~11.06) : 게임 기획

2주차(2020.11.09~11.13) : 게임 개발에 필요한 패키지 다운 및 기능을 구현

3, 4주차(2020.11.16~11.27) : 테스트 및 디버그

**2. 기능 구현**

**2-1 근접 공격 구현**

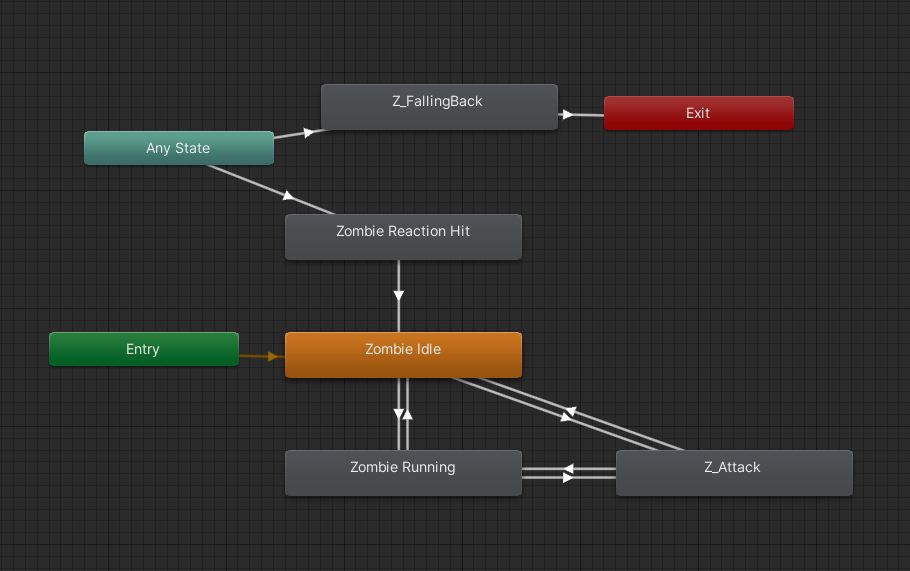
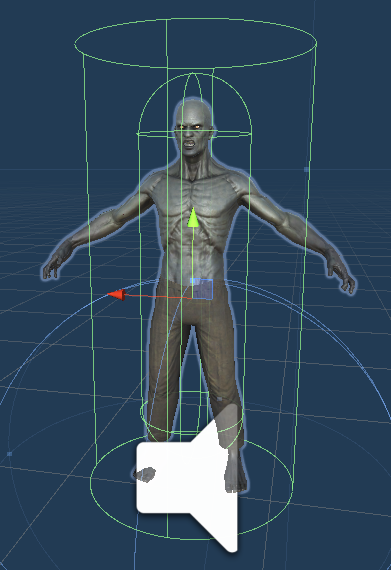


● 다음과 같이 플레이어 앞에 empty object를 생성하여 플레이어가 근접 공격키를 눌렀을 때, empty object의 콜라이더가 끄고 켜짐을 유도하여, 좀비들에게 충돌하게 하여 데미지를 줌

|  |  |
| --- | --- |
| Empty object에 이용된 코드 | |
|  | F키의 입력을 받아 empty object의  box collider를 활성화 및 비활성화  is\_knifed변수를 이용하여 근접공격에 1.5초의 딜레이를 줌 |

**2-2 좀비 구현**

● 좀비 모델 패키지를 받은 후 믹사모에서 애니메이션을 추출하여 애니메이터 컨트롤러를 구성함

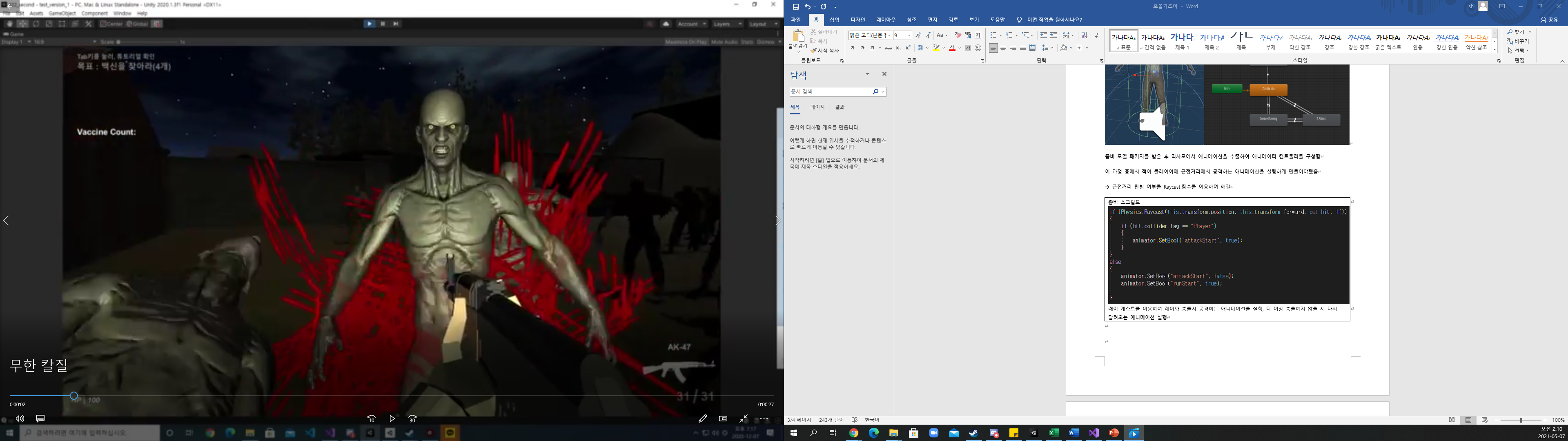


● 이 과정 중에서 적이 플레이어에 근접거리에서 공격하는 애니메이션을 실행하게 만들어야했음

🡪 근접거리 판별 여부를 Raycast함수를 이용하여 해결

|  |
| --- |
| 좀비 스크립트중의관련코드 |
| 래이 캐스트를 이용하여 레이와 충돌시 공격하는 애니메이션을 실행, 더 이상 충돌하지 않을 시 다시 달려오는 애니메이션 실행 |

● 좀비에 피 튀기는 효과, 사운드를 추가하여 타격감을 높임



좀비의 피격과, 그에 따른 처리를 다음과 같이 나타냈다.

|  |
| --- |
|  |
| 좀비의 피를 Random함수를 이용해 각각에게 랜덤한 HP를 부여함  피격의 경우를 근접공격과 총알로 정해 각각에게 HP감소치를 다르게 부여함  죽을 경우 sp\_point스크립트에 현재 살아있는 좀비 수의 변수에 정보를 전달  시체가 남아있는 모습을 보여주기 위해 이동속도를 0으로 만들고 collider를 꺼주었다. |

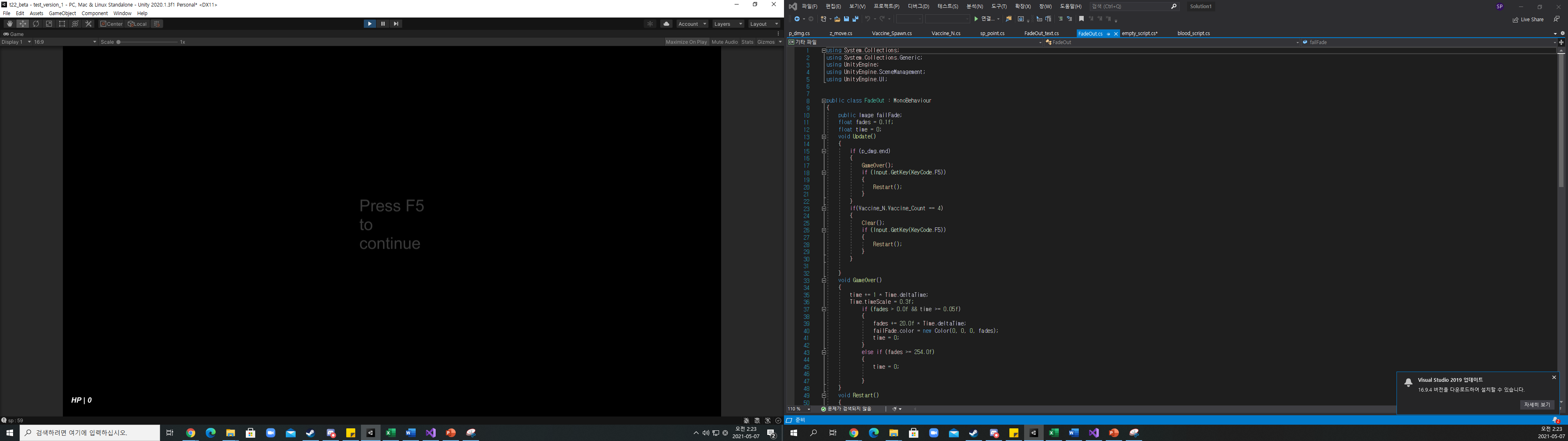
● 좀비에게 피격을 받은 주인공의 체력감소는 다음과 같이 구현했다.

|  |
| --- |
|  |
| OnCollisionStay함수를 이용하여 “좀비와 얼마나 접촉하여 있었냐”에 따라 시간에 반비례하여 체력이 줄게 하였습니다. |

●좀비의 생성지점

|  |
| --- |
|  |
| 이런 좀비들을 생성하기위해 빈 오브젝트에 sp\_point스크립트를 넣어 2초마다 프리팹으로 생성되게 하였다  죽은 좀비가 생성지점의 좀비가 몇 마리 살았는지 알려주는 변수에 접근하게 하여, 일정 수의 좀비를 유지할 수 있게 했다. |

3. 게임 오버화면



타임스케일과 UI를 이용하여 게임오버화면을 구성했다.

**3. 프로젝트 진행 후기**

|  |  |
| --- | --- |
| **프로젝트를 진행하면서 어려웠던 점** | **1. 팀원과의 소통 부족.**  팀원이 많음을 이점으로 서로 역할을 분담하여 개인적인 부담을 줄여주는 장점이 있었지만, 서로 간의 일에만 집중하여 소통이 부족했음. 대략적인 계획은 있었지만, 서로의 손과 발이 맞지 않아 계획에 차질이 생기기도 함.  **2. 서로의 작업물을 합치기**  서로 작업한 것들을 하나로 합치기 시작했을 때, 서로가 서로의 작업물을 제대로 이해하지 못해 큰 어려움을 겪음. 그래도 그 과정에서 대화와 토론을 통해 더 좋은 스크립트나 모델을 구할 수 있었음  **3. 두루뭉실한 계획**  계획이 있지만, 너무 애매모호하여 계획을 진행하기가 어려웠음. |
| **깨달은 점** | 1. **팀원과의 적극적인 소통을 하여 진행하는 프로젝트를 이해해야함.** 2. **서로 코드를 주고받을 때 주석을 통해 상대방이 이해하기 쉽게 해야함.** 3. **계획의 구체화(분석, 설계, 구현, 디버그)가 필요.** |

참고 : 발표 ppt자료

<https://drive.google.com/file/d/1psEdSNLDTpag4XnXk8yg66yB_e-XDpba/view?usp=sharing>