

2022-1학기 캡스톤디자인 I

웹 기반 메타버스 구축 플랫폼

캡스톤디자인 중간 발표

2022.05.04

한밭대학교 컴퓨터공학과
최진아, 이혜진, 유선아

CONTENTS

01

소프트웨어 요구사항

02

진행 사항

03

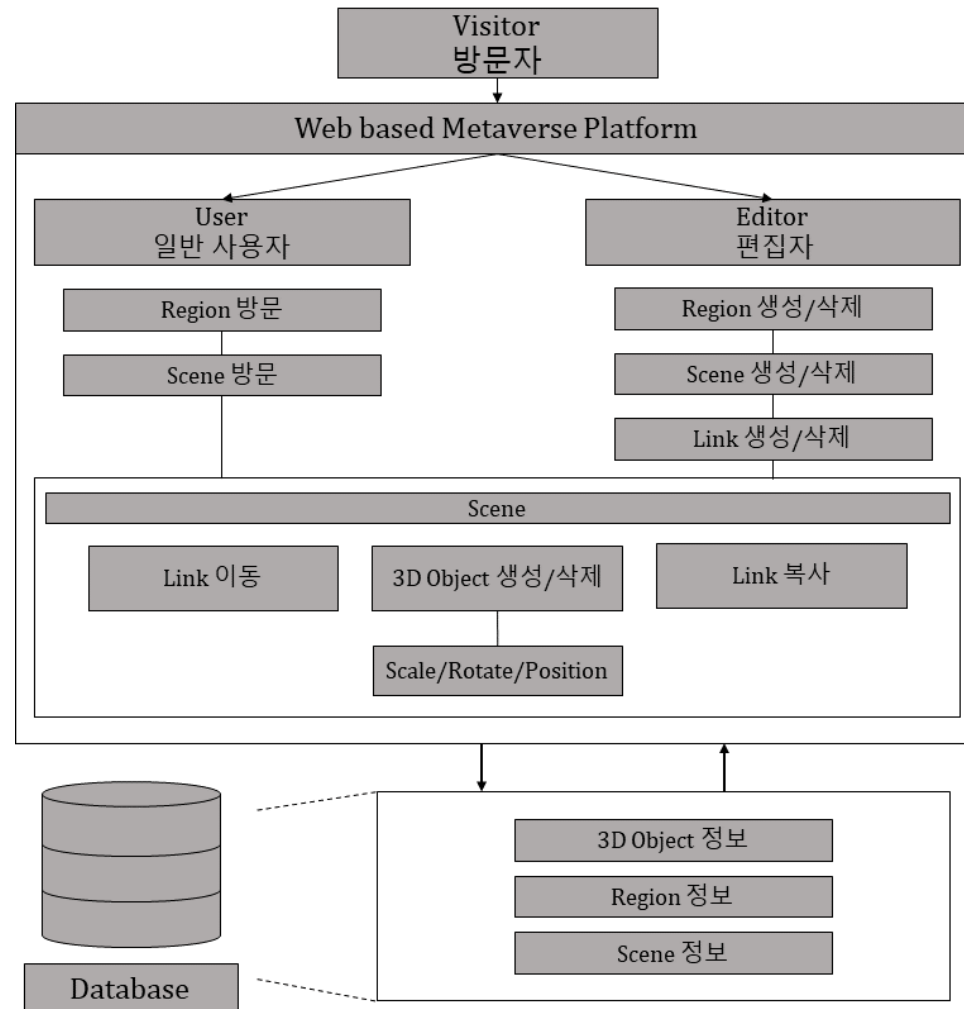
향후 계획

01

소프트웨어 요구사항
하

01 소프트웨어 요구사항

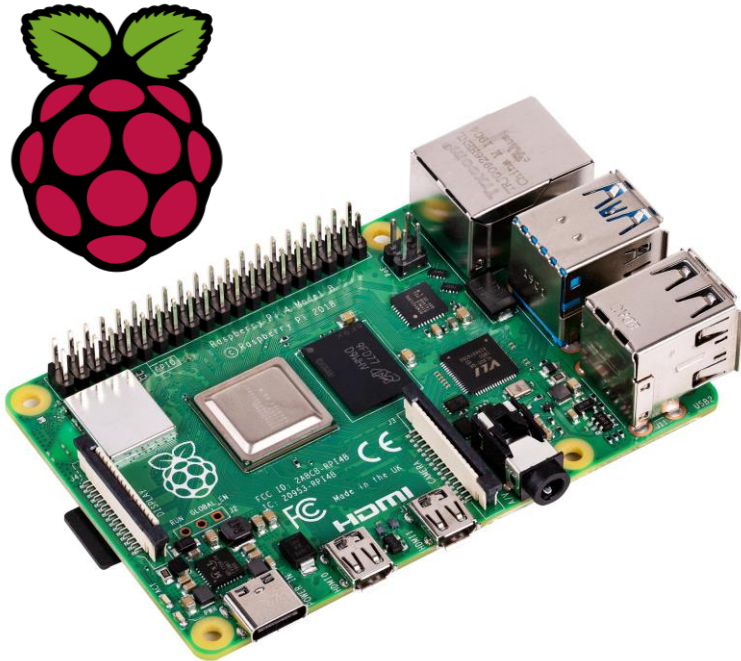
1) 시스템 개요



01 소프트웨어 요구사항

2) 시스템 장비

라즈베리파이 4



Ubuntu, mongoDB 사용



01 소프트웨어 요구사항

3) 기능

Region 생성 / 삭제
Scene 생성 / 삭제 / 연결

Region

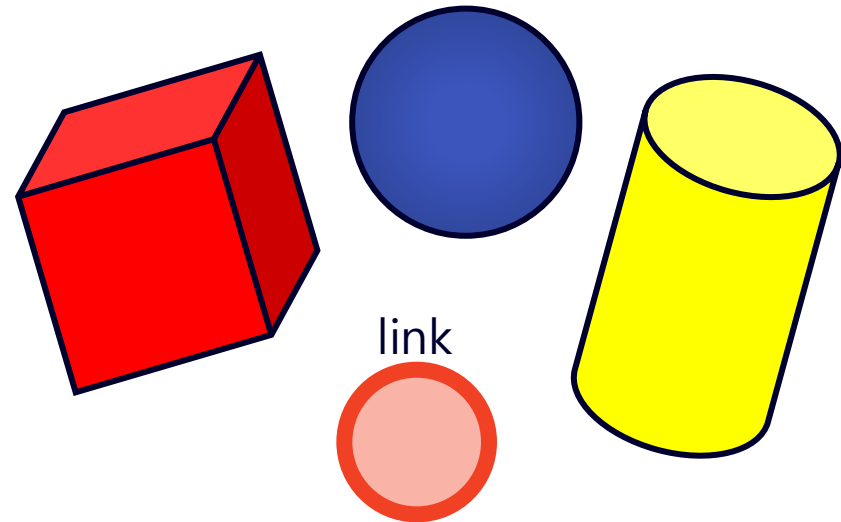
Scene

Scene

Scene

3D object 생성 / 삭제 / 편집
(위치, 회전, 모양, 색)

Scene



01 소프트웨어 요구사항




4) 테스트

테스트 계획

테스트 수행

결과 분석

날짜	대상	인원	내용
2월 중(3일)	쌍별여자고등학교	7	프로토타입 테스트
3월~4월(2일)	유성여자고등학교	17	프로토타입 테스트
5월(3일)	한밭대학교 컴퓨터공학과 공학설계입문(01)	38	개발물 테스트
5월(3일)	한밭대학교 컴퓨터공학과 공학설계입문(02)	37	개발물 테스트
6월~7월(2일)	유성여자고등학교	17	개발물 테스트

-  진행 완료
-  진행 중
-  진행 예정

01 소프트웨어 요구사항

4) 테스트

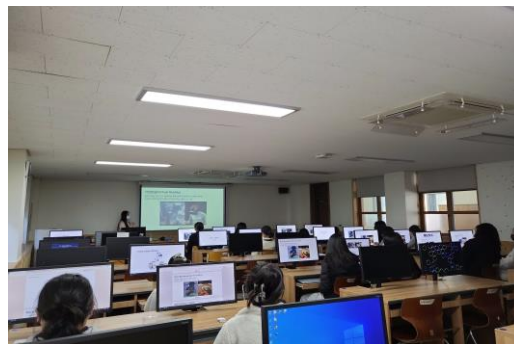
테스트 계획

테스트 수행

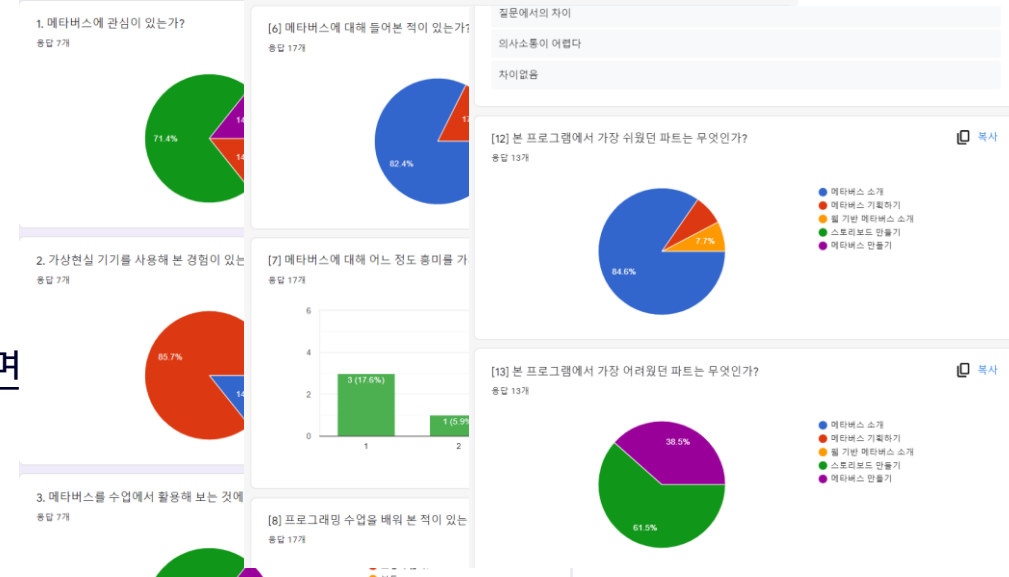
결과 분석



▲ 샌블여자고등학교(비대면) ▲ 유성여자고등학교(비대면)



▲ 유성여자고등학교(대면) ▲ 한밭대학교(대면)



▲ 설문조사 결과

테스트 계획

테스트 수행

결과 분석

- link에 다가가기만 해도 이동하는 기능
- 사용자 접속이 많은 경우 지연 시간이 발생함
- left, right 방향 편집
- edit을 할 수 있는 권한

- 직접 만들어보는 실습 활동이 많았으면 좋겠음	음
- 직접 만드는 실습이고 어려운 과정이 아니었음	음
- 비대면 수업 시 질문에 어려움을 겪음	논
- 공학 계열 진학에 도움을 얻음	G

논문 투고 : FRONTIERS IN EDUCATION 2022 –
Grand challenges in Engineering Education



01 소프트웨어 요구사항

5) 성능, 6) 인터페이스, 7) 데이터

성능	고유번호	요구사항 정의 및 세부내용
웹 페이지 로딩 시간	PER-01	<ul style="list-style-type: none"> 플랫폼 이용 사용자의 화면에 가상 세계가 로딩되는 시간 각 가상 세계 내에 배치된 3D object들을 포함한 전체 웹 페이지가 사용자가 요구한 시점으로부터 10초 이내에 완벽하게 로딩되어야 함
동시 사용자 접속 수	PER-02	<ul style="list-style-type: none"> 웹 페이지 당 동시 사용자 수 50명 이상을 지원해야 하고 성능이 저하되지 않아야 함
데이터 형식 오류 응답	PER-03	<ul style="list-style-type: none"> 사용자로부터 발생한 모든 오류에 대해서 5초 이내에 오류 웹 페이지 또는 메시지를 전송해야 함
인터페이스	고유번호	요구사항 정의 및 세부내용
3D Object GUI	SIR-01	정의 : 사용자가 3D Object와 관련된 기능 사용 시의 인터페이스 <ul style="list-style-type: none"> 사용자는 구체적인 사용법 없이도 인터페이스만을 보고 3D Object를 생성, 추가, 편집할 수 있어야 함
사용자/편집자 구분	SIR-02	정의 : 사용자/편집자 구분 <ul style="list-style-type: none"> 사용자는 새로운 가상세계를 생성 또는 삭제하고 싶을 경우 편집자 모드를 통해서 원하는 대로 편집할 수 있어야 함
오류 웹페이지 안내	SIR-03	정의 : 사용자의 잘못된 입력으로 인한 오류 발생 시의 오류 웹페이지 <ul style="list-style-type: none"> 웹 플랫폼을 사용하면서 사용자가 잘못된 입력을 할 경우 발생한 오류에 관한 설명을 포함한 오류 웹 페이지를 사용자에게 제시해야 함
데이터	고유번호	요구사항 정의 및 세부내용
데이터베이스 구축	DAR-01	정의 : 사용자가 서비스를 이용하면서 생성된 데이터를 보관하기 위한 데이터베이스를 구축함 <ul style="list-style-type: none"> 내부 데이터베이스는 MongoDB를 사용함 데이터베이스는 관리자 외에 접근할 수 없음
데이터 저장	DAR-02	정의 : 사용자가 서비스를 이용하면서 생성된 데이터를 내부 데이터베이스에 저장함 <ul style="list-style-type: none"> 데이터베이스에 저장된 정보는 사용자가 삭제하지 않는 이상 보관함
데이터 수정	DAR-03	정의 : 사용자가 서비스를 이용하면서 생성된 데이터를 수정할 때 내부 데이터베이스에도 반영되어야 함 <ul style="list-style-type: none"> 데이터베이스에 저장된 정보는 사용자가 수정하는 즉시 반영되어야 함

01 소프트웨어 요구사항

8) 보안, 9) 제약사항, 10) 프로젝트 관리

보안	고유번호	요구사항 정의 및 세부내용
장비 보안 요구사항	SER-01	정의 : 시스템을 운영하는 통신 서버의 물리적 · 시스템적 보안에 대한 요구사항 <ul style="list-style-type: none"> 통신 서버에 물리적으로 접근할 수 없도록 분리된 공간에 위치해야 함 통신 서버에 접속하기 한 계정의 아이디와 패스워드는 관리자 외 알 수 없도록 관리해야 함
네트워크 보안 요구사항	SER-02	정의 : 네트워크 접근 통제, 네트워크 장비의 취약성 및 구성 설정에 대한 보안 요구사항 등 통신을 위해 사용하는 장비 및 접근과 관련한 요구사항 <ul style="list-style-type: none"> 유선 네트워크를 사용하여 무선 네트워크 사용으로 발생할 수 있는 취약점을 예방함
제약 사항	고유번호	요구사항 정의 및 세부내용
프레임워크	COR-01	<ul style="list-style-type: none"> JavaScript, Aframe, Node.js 등에서 기본으로 제공되는 라이브러리 사용
프로그래밍 언어	COR-02	<ul style="list-style-type: none"> 프론트엔드에는 JavaScript, HTML, CSS, Pug, Aframe을 사용함 백엔드에는 Node.js, JavaScript를 사용함
개발 방법론	COR-03	<ul style="list-style-type: none"> 알파 테스트와 애자일 방법론 사용 알파 테스트에서 받은 피드백 기반으로 애자일 방법론을 적용함 일정 주기로 실시하는 알파 테스트마다 기능들을 추가하며 프로토타입을 제작함
업무 모듈화 및 자원 활용 방안	COR-04	<ul style="list-style-type: none"> 유연성, 확장성을 확보할 수 있도록 모듈화 개발전략을 반영함 현재 보유하여 활용 가능한 H/W, S/W를 최대한 재활용하며, 추가 도입이 필요한 솔루션의 경우 본 용역에 포함하여 수행함
시스템 설계	COR-05	<ul style="list-style-type: none"> 기존 구축되어 있는 현행 시스템 구조 및 전체 표준화 호환성, 시스템 분산설계, 데이터 유형, 프로세스 환경 유형, 사용자 유형, 시스템 토폴로지가 고려되어 구조 설계가 이루어져야 함
프로젝트 관리	고유번호	요구사항 정의 및 세부내용
개발 장소 및 필요 장비	PMR-01	<ul style="list-style-type: none"> 개발 장소로 학교의 빈 강의실 사용 필요 장비로 라즈베리파이 등이 있음
프로젝트 수행조직	PMR-02	<ul style="list-style-type: none"> 프론트엔드 개발, 백엔드 개발은 데이터베이스의 데이터를 이용해 가상공간의 입출력을 처리함 데이터베이스 개발 및 관리는 데이터베이스를 개발하고 데이터를 관리함
프로젝트 일정계획	PMR-03	<ul style="list-style-type: none"> 주기적으로 알파테스트를 진행하며 개발을 진행함
단계별 산출물	PMR-04	<ul style="list-style-type: none"> 계획서, 중간보고서, 개발 및 커스터마이징 보고서 제출함

02

진행 사항

02 진행 사항

1) single-object 저장 / 불러오기



1

화면에 임의의 single-object 생성

vr item.pug : assets 정의

```

49 // - Object Make
50 a-assets
51   a-mixin(id='box' geometry='primitive: box; width: 0.5; height: 0.5; depth: 0.5' hoverable='' grabbable=''
    stretchable='' draggable='' droppable='' shadow='' event-set__dragdrop='_event: drag-drop;'
    event-set__hoveron='_event: hover-start; material.opacity: 0.7; transparent: true'
    event-set__hoveroff='_event: hover-end; material.opacity: 1; transparent: false'
    event-set__dragon='_event: dragover-start; material.wireframe: false' event-set__dragoff='_event:
    dragover-end; material.wireframe: false')
52
53   a-mixin(id='sphere' geometry='primitive: sphere; width: 0.5; height: 0.5; depth: 0.5' hoverable=''
    grabbable='' stretchable='' draggable='' droppable='' shadow='' event-set__dragdrop='_event: drag-drop;'
    event-set__hoveron='_event: hover-start; material.opacity: 0.7; transparent: true'
    event-set__hoveroff='_event: hover-end; material.opacity: 1; transparent: false'
    event-set__dragon='_event: dragover-start; material.wireframe: false' event-set__dragoff='_event:
    dragover-end; material.wireframe: false')
54
55   a-mixin(id='cylinder' geometry='primitive: cylinder; width: 0.5; height: 0.5; depth: 0.5' hoverable=''
    grabbable='' stretchable='' draggable='' droppable='' shadow='' event-set__dragdrop='_event: drag-drop;'
    event-set__hoveron='_event: hover-start; material.opacity: 0.7; transparent: true'
    event-set__hoveroff='_event: hover-end; material.opacity: 1; transparent: false'
    event-set__dragon='_event: dragover-start; material.wireframe: false' event-set__dragoff='_event:
    dragover-end; material.wireframe: false')

```

vr item.pug : object 생성

```

87 a-entity(id='boxtest' loc='box1' mixin='box' position='0 0 0'
    rotation='0 0 0' color='red' material='color: red')

```

02 진행 사항

1) single-object 저장 / 불러오기



2

DB Schema 정의 및 저장

vrModel.js : Schema 정의

```

1 var mongoose = require('mongoose');
2 var Schema = mongoose.Schema;
3
4 var vrSchema = new Schema({
5   region_id: {type:mongoose.Schema.ObjectId},
6   scene_name:String,
7   image_file: String,
8   left_name: String,
9   up_name: String,
10  right_name: String,
11  down_name: String,
12  links:{left: {name:String, x:Number, y:Number, z:Number, yaw:Number, pitch:Number, roll:Number},
13          up:   {name:String, x:Number, y:Number, z:Number, yaw:Number, pitch:Number, roll:Number},
14          right:{name:String, x:Number, y:Number, z:Number, yaw:Number, pitch:Number, roll:Number},
15          down: {name:String, x:Number, y:Number, z:Number, yaw:Number, pitch:Number, roll:Number},
16          default: false},
17  boxtest:{pos: {x:Number, y:Number, z:Number, yaw:Number, pitch:Number, roll:Number, id:String, loc:String, mixin:String, color:String},
18            default: false},
19 });
20
21 module.exports = mongoose.model('vrItem', vrSchema);
  
```

vr.js : DB에 저장

```

136 if("boxtest" in req.body)
137 {
138   console.log('boxtest');
139   item.boxtest.pos.x = req.body.boxtest[0]['x']; //position
140   item.boxtest.pos.y = req.body.boxtest[0]['y'];
141   item.boxtest.pos.z = req.body.boxtest[0]['z'];
142
143   item.boxtest.pos.yaw = req.body.boxtest[1]['x']; //rotation
144   item.boxtest.pos.pitch = req.body.boxtest[1]['y'];
145   item.boxtest.pos.roll = req.body.boxtest[1]['z'];
146
147   item.boxtest.pos.id = req.body.boxtest[2]; //id
148   item.boxtest.pos.loc = req.body.boxtest[3]; //loc
149   item.boxtest.pos.mixin = req.body.boxtest[4]; //mixin
150   item.boxtest.pos.color = req.body.boxtest[5]; //color
  
```

02 진행 사항

1) single-object 저장 / 불러오기



3

DB에서 object 가져오기

vr item.pug : entity 불러오기

```
67 //test entity
68 if objectlist
69   a-entity(id='boxtest' loc='box1' mixin='box' position=objectlist.pos.x+ ' '+objectlist.pos.y+ ' '+objectlist.pos.z rotation=objectlist.pos.yaw+ ' '+objectlist.pos.pitch+ ' '
    +objectlist.pos.roll color='red' material='color: '+objectlist.pos.color)
```

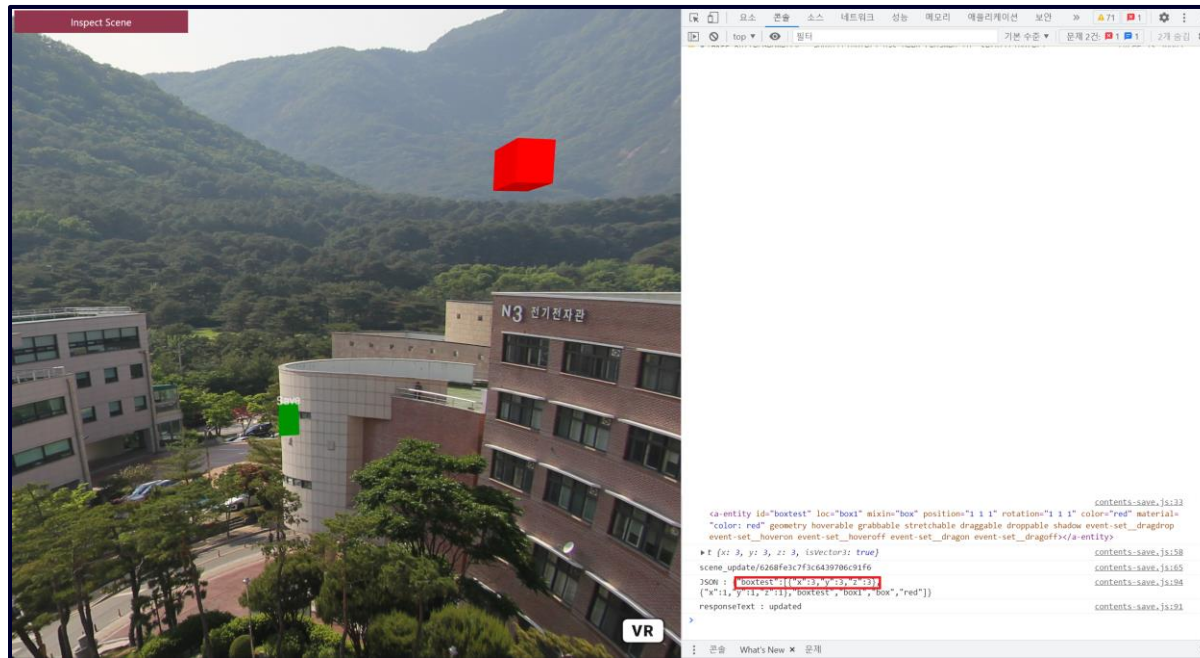
02 진행 사항

1) single-object 저장 / 불러오기



실행 결과

동작 모습



02 진행 사항

2) multiple-object 저장 / 불러오기



1

화면에 임의의 multiple-object 생성

vr item.pug : object 생성

```
83 a-entity(id='boxtest' loc='box1' position='-2 -2 0' rotation = '25 0 30'  
    geometry='primitive: box' material='color: red' )  
84 a-entity(id='boxtest' loc='sphere1' position='0 0 0' rotation = '0 0 0'  
    geometry='primitive: sphere' material='color: blue' )  
85 a-entity(id='boxtest' loc='cylinder1' position='2 -2 0' rotation = '20 -15 -15'  
    geometry='primitive: cylinder' material='color: yellow' )
```

02 진행 사항

2) multiple-object 저장 / 불러오기



2

DB Schema 정의 및 저장

vrModel.js : array Schema 정의

```
17   boxtest:[new mongoose.Schema({
18       x:Number, y:Number, z:Number,
19       yaw:Number, pitch:Number, roll:Number,
20       id:String, loc:String, geometry:String,
        color:String })],
```

vr.js : jdata에 object attribute 저장

```
51 | // jarray에 entity array로 넣기
52 | for(elem2 of el2)
53 | {
54 |     jarray.push([elem2.getAttribute('position'), elem2.
55 |         getAttribute('rotation'), elem2.getAttribute('id'), elem2.
56 |         getAttribute('loc'), elem2.getAttribute('geometry').
57 |         primitive, elem2.getAttribute('material').color]);
58 | }

// jdata에서 key가 boxtest이고, value가 jarray로 넣기
jdata['boxtest'] = jarray;
```

vr.js : DB에 저장

```
141 | // ##### DB에 entity 저장하기 #####
142 | if("boxtest" in req.body)
143 | {
144 |     // boxtest에 이미 저장된 요소들 undefined가 될 때까지 비우기
145 |     while(item.boxtest.shift() !== undefined)
146 |
147 |     console.log('boxtest try save');
148 |     // console.log('length' + String(req.body.boxtest.length));
149 |
150 |     // boxtest value에 entity 하나씩 push하기
151 |     for(let i = 0; i < (req.body.boxtest.length); i++){
152 |         item.boxtest.push({
153 |             x: req.body.boxtest[i][0]['x'], // position
154 |             y: req.body.boxtest[i][0]['y'],
155 |             z: req.body.boxtest[i][0]['z'],
156 |             yaw: req.body.boxtest[i][1]['x'], // rotation
157 |             pitch: req.body.boxtest[i][1]['y'],
158 |             roll: req.body.boxtest[i][1]['z'],
159 |             id: req.body.boxtest[i][2], // id
160 |             loc: req.body.boxtest[i][3], // loc
161 |             geometry: req.body.boxtest[i][4], // material
162 |             color: req.body.boxtest[i][5] // color
163 |         });
164 |     }
165 |     console.log('boxtest finish save');
166 | }
```

02 진행 사항

2) multiple-object 저장 / 불러오기



3

DB에서 object 가져오기

vr.js : boxtest를 objectList로 render

```

57 // boxtest는 entityList로 가져옴
58 entityList = vritem.boxtest;
59
60 // linkList, objectList render하기
61 return res.render("vr_item", { vid:req.params.vid,
  vrimage_id: vritem.image_file, arrowList:linkList,
  objectList:entityList });
  
```

vr item.pug : object 가져오기

```

75 // fetch entities
76 - var i = -1
77 while ++i < objectList.length
78   a-entity(position=objectList[i].x+ ' '+objectList[i].y+ ' '+objectList[i].z
    rotation=objectList[i].yaw+ ' '+objectList[i].pitch+ ' '+objectList[i].roll
    id=objectList[i].id loc=objectList[i].loc geometry='primitive: '+objectList[i].
    geometry material='color: '+objectList[i].color)
  
```

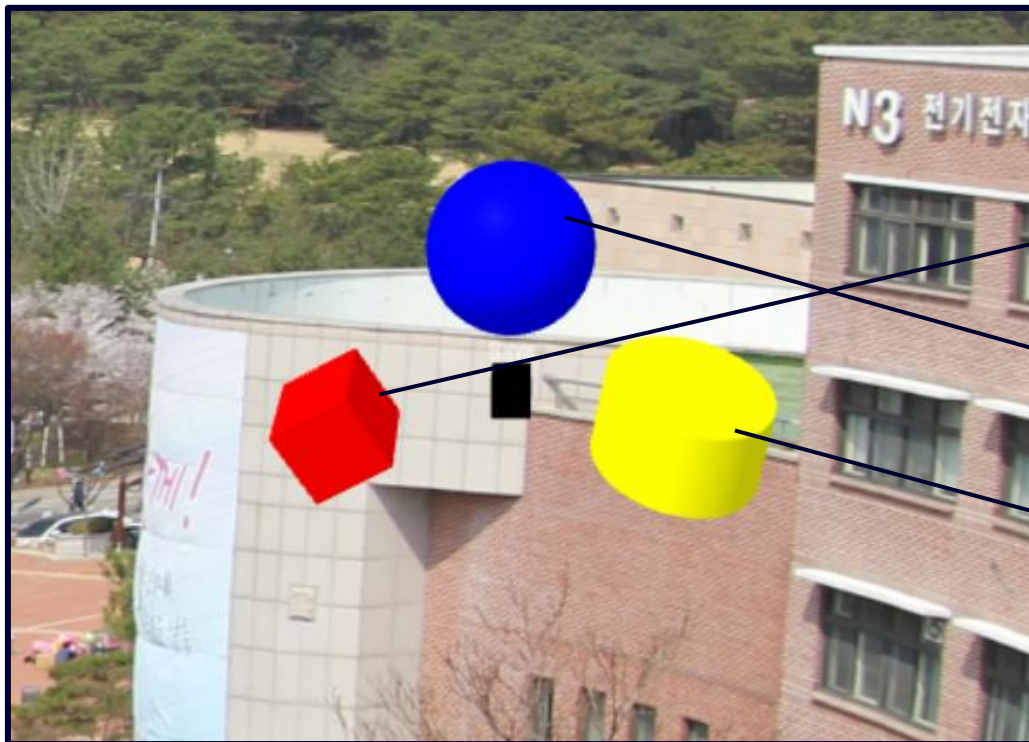
02 진행 사항

2) multiple-object 저장 / 불러오기

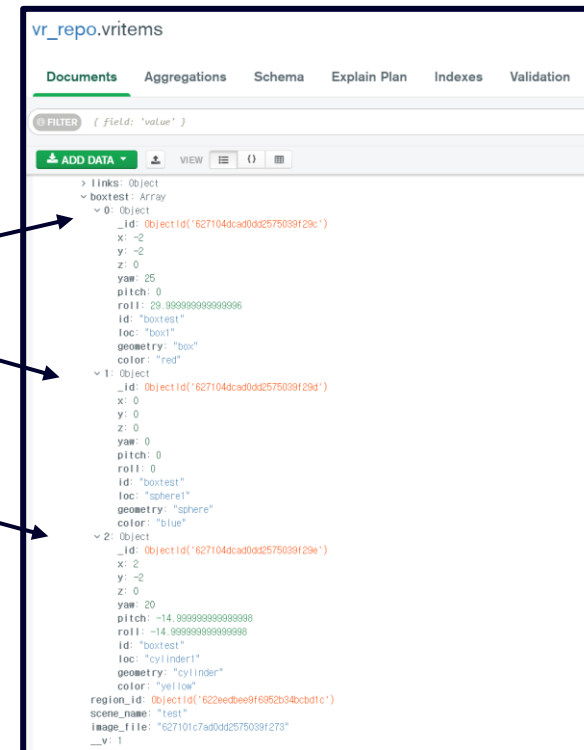


실행 결과

동작 모습



DB



02 진행 사항

3) Dropbox(object 생성, 편집)

dropdown-event.js

Dropbox UI 동작

```
1  window.closeDropdown = function() {  
2      console.log("closedropdown");  
3      var mydropdown = document.getElementById("mydropdown");  
4      mydropdown.emit('closedropdown');  
5      var myoptions = document.getElementById("myoptions");  
6      myoptions.emit('closedropdown');  
7  }  
8  }  
9  
10 window.openDropdown = function() {  
11     console.log("opendropdown");  
12     var mydropdown = document.getElementById("mydropdown");  
13     mydropdown.emit('opendropdown');  
14     var myoptions = document.getElementById("myoptions");  
15     myoptions.emit('opendropdown');  
16 }  
17 }
```

02 진행 사항

3) Dropbox(object 생성, 편집)

pressed-gui-button.js

Dropbox 메뉴 클릭에 따른 동작(object 생성)

```

1 AFRAME.registerComponent('pressed-gui-button', {
2
3   init: function() {
4     //var sceneEl = document.querySelector('a-scene');
5     var objectEl = document.querySelector('#objects');
6     var markerEl = document.querySelector('#marker');
7     var boxIndex = 0;
8     var sphereIndex = 0;
9     var cylinderIndex = 0;
10
11     // Add boxe when spacebar is pressed.
12     this.el.addEventListener('click', function (e) {
13       //if (e.keyCode !== 32) return;
14       //console.log(this.id);
15
16       // 각 오브젝트 유형에 따른 설정 지정
17       switch(this.id) {
18         case "box":
19           var newEl = document.createElement('a-box');
20           newEl.setAttribute('color', 'red');
21
22           newEl.setAttribute('class', this.id);
23           newEl.setAttribute('id', this.id + boxIndex);
24           boxIndex++;
25
26           newEl.setAttribute('mixIn', this.id);
27
28           break;
29         case "sphere":
30           var newEl = document.createElement('a-sphere');
31           newEl.setAttribute('color', 'blue');
32
33           newEl.setAttribute('class', this.id);
34           newEl.setAttribute('id', this.id + sphereIndex);
35           sphereIndex++;
36
37           newEl.setAttribute('mixIn', this.id);
38

```

```

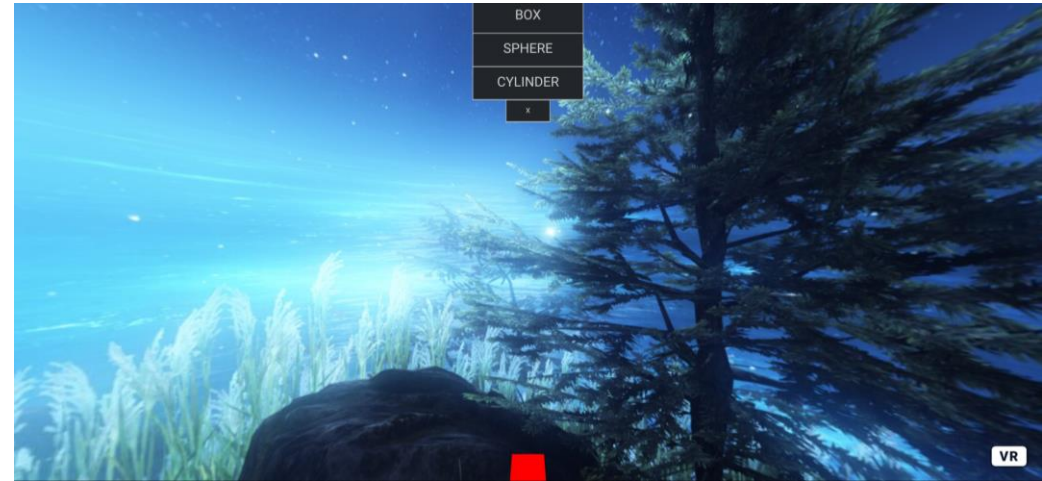
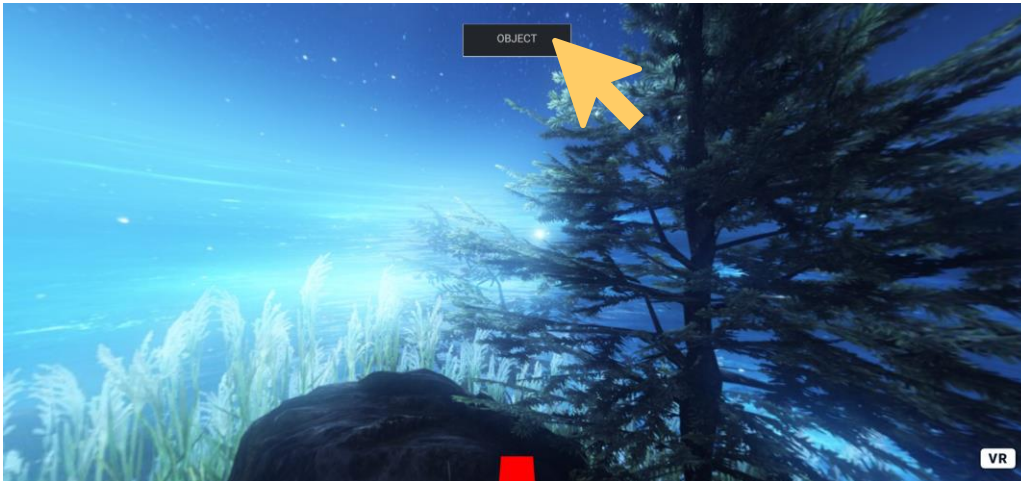
39           break;
40         case "cylinder":
41           var newEl = document.createElement('a-cylinder');
42           newEl.setAttribute('color', 'yellow');
43
44           newEl.setAttribute('class', this.id);
45           newEl.setAttribute('id', this.id + cylinderIndex);
46           cylinderIndex++;
47
48           newEl.setAttribute('mixIn', this.id);
49
50           break;
51         default: break;
52       }
53
54       // 오브젝트 생성
55       //sceneEl.appendChild(newEl);
56       objectEl.appendChild(newEl);
57
58       // 각 오브젝트가 클릭이 되도록 리스너 달기
59       newEl.classList.add('clickable');
60       newEl.addEventListener('click', handleClickEvent);
61
62       // 오브젝트가 생성되었을 때의 위치 수정
63       var position = markerEl.object3D.getWorldPosition();
64       //position.y = 0.5;
65       newEl.setAttribute('position', position);
66     });
67   }
68 });
69
70 // 해당 오브젝트가 맞는지 확인하는 용
71 function handleClickEvent (event)
72 {
73   var obj = event.target;
74
75   console.log(obj.id);
76   console.dir(obj);
77   // - console.log(obj.getAttribute("position"));
78 }

```


02 진행 사항

3) Dropbox(object 생성, 편집)

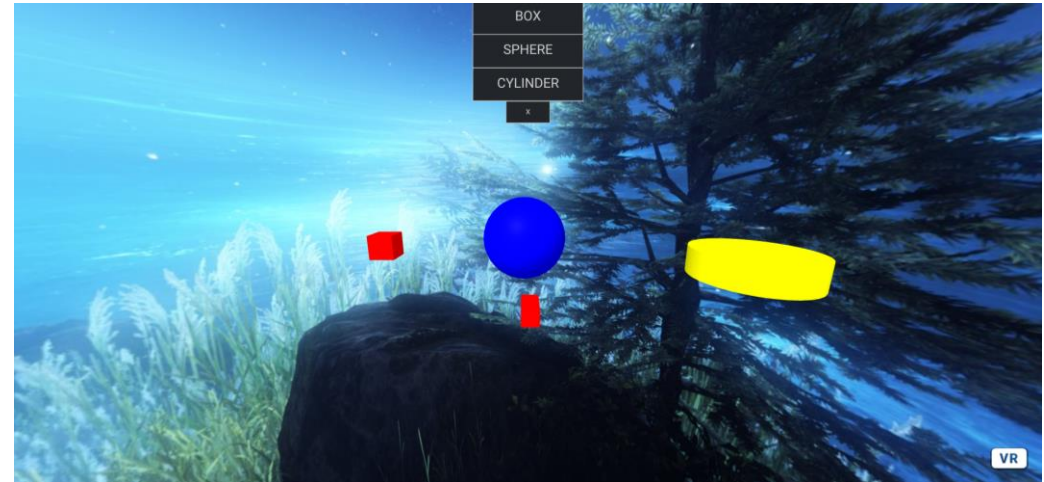
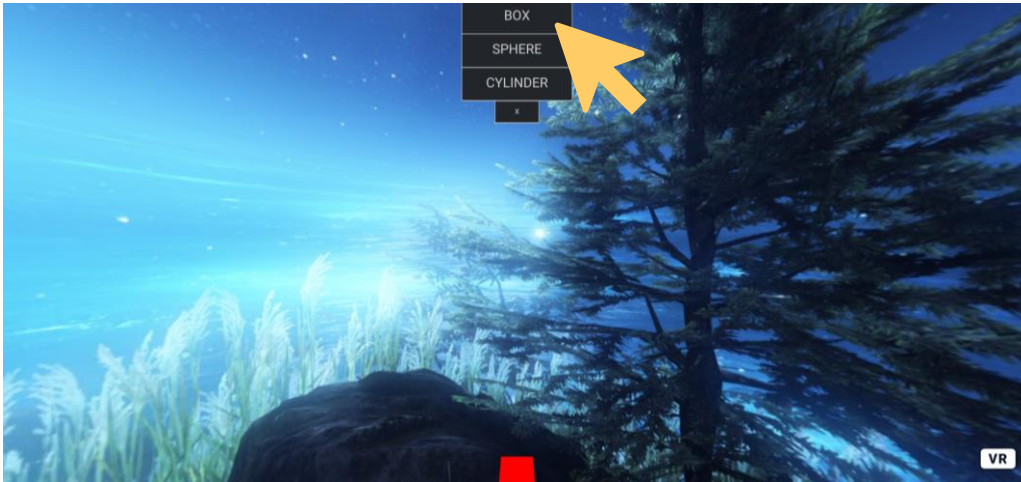
동작 모습



02 진행 사항

3) Dropbox(object 생성, 편집)

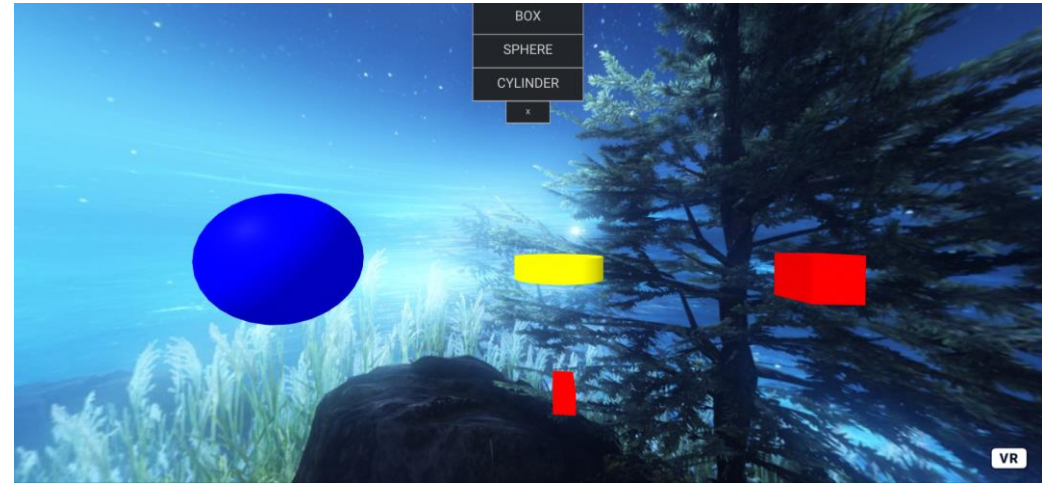
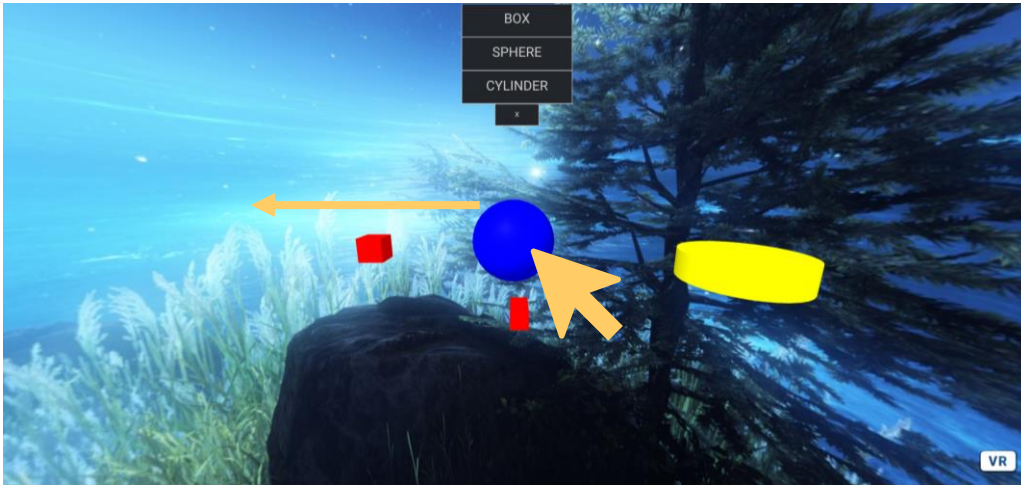
동작 모습



02 진행 사항

3) Dropbox(object 생성, 편집)

동작 모습



03

향후 계획

03 향후 계획

1) 역할 분담



03 향후 계획

2) 일정표

현재 진행 상황



	1월	2월	3월	4월	5월	6월
프로젝트 계획 수립						
프로젝트 준비						
프로토타입에 대한 연구 논문 작성						
프로젝트 수행(개발)						
알파 테스트 및 유지보수						
프로젝트 마무리						

- 기존 위치, 회전 구현했던 코드 develop (+ x, y, z축을 이용한 UI 기능 구현)
- object 색 변경 기능
- 개발 진행 중 & 후, 단위 테스트 및 알파 테스트 진행
- 기능이 모두 구현되면, pug를 사용하지 않고 NestJS/Next.js 사용하여 리팩토링 계획

03 향후 계획

3) 중요 마일스톤 및 진행정도

진행한 마일스톤

1. 기능 요구사항에 대한 중요 마일스톤

- 1.1. Region 생성 및 삭제 및 씬(Scene) 연결 - 프로토타입
- 1.2. 씬(Scene) 생성/읽기/삭제 - 프로토타입 및 360도 이미지 없이 디폴트 배경 씬 생성에 대한 소스코드 추가
 - 1.2.1 내부 GUI button을 통한 3D Object 생성 소스코드 작성 완료
 - 1.2.2 생성된 3D Object DB 연결 및 저장 소스코드 작성 완료
 - 1.2.3 3D Object Position 조정 소스코드 작성 완료
- 1.3. 씬(Scene) 편집
 - 1.3.1 내부 GUI button을 통한 3D Object 생성 소스코드 작성
 - 1.3.2 생성된 3D Object DB 연결 및 저장 소스코드 작성
 - 1.3.3 3D Object Position/Rotate/Scale 조정 소스코드 작성
- 1.4. Region 공유 - link 공유를 위한 button 소스코드 작성

2. 성능 요구사항에 대한 중요 마일스톤

- 2.1. 웹 페이지 로딩 시간 - 웹 페이지 로딩 시간 테스트
- 2.2. 동시 사용자 접속 수 - 알파테스트를 통한 동시 사용자 수 테스트
- 2.3. 데이터 형식 오류 응답 시간 - 플랫폼에서 발생할 수 있는 오류들에 대한 예외처리

3. 인터페이스 요구사항에 대한 중요 마일스톤

- 3.1. 3D Object GUI - 3D Object 사용시의 필요 기능 정리
- 3.2. 오류 웹페이지 안내 - 사용자 입장에서 발생할 수 있는 오류들에 대한 오류 테스트

4. 데이터 요구사항에 대한 중요 마일스톤

- 4.1. 데이터베이스 구축 및 저장 - 데이터베이스 서버 구축 및 오픈, 연결 소스코드 작성
- 4.2. 데이터 수정 - 저장 데이터 불러오기 및 수정 소스코드 작성

5. 테스트 요구사항에 대한 중요 마일스톤

- 5.1. 테스트 계획 수립 - 플랫폼을 활용한 테스트 수업 계획 작성
- 5.2. 테스트 수행 - 계획에 맞춘 테스트 수업 진행
- 5.3. 단위 테스트 - 각 기능에 대한 테스트 수행
- 5.4. 알파 테스트 - 해당 플랫폼을 활용한 수업 진행 후 테스트 결과 취합

감사합니다