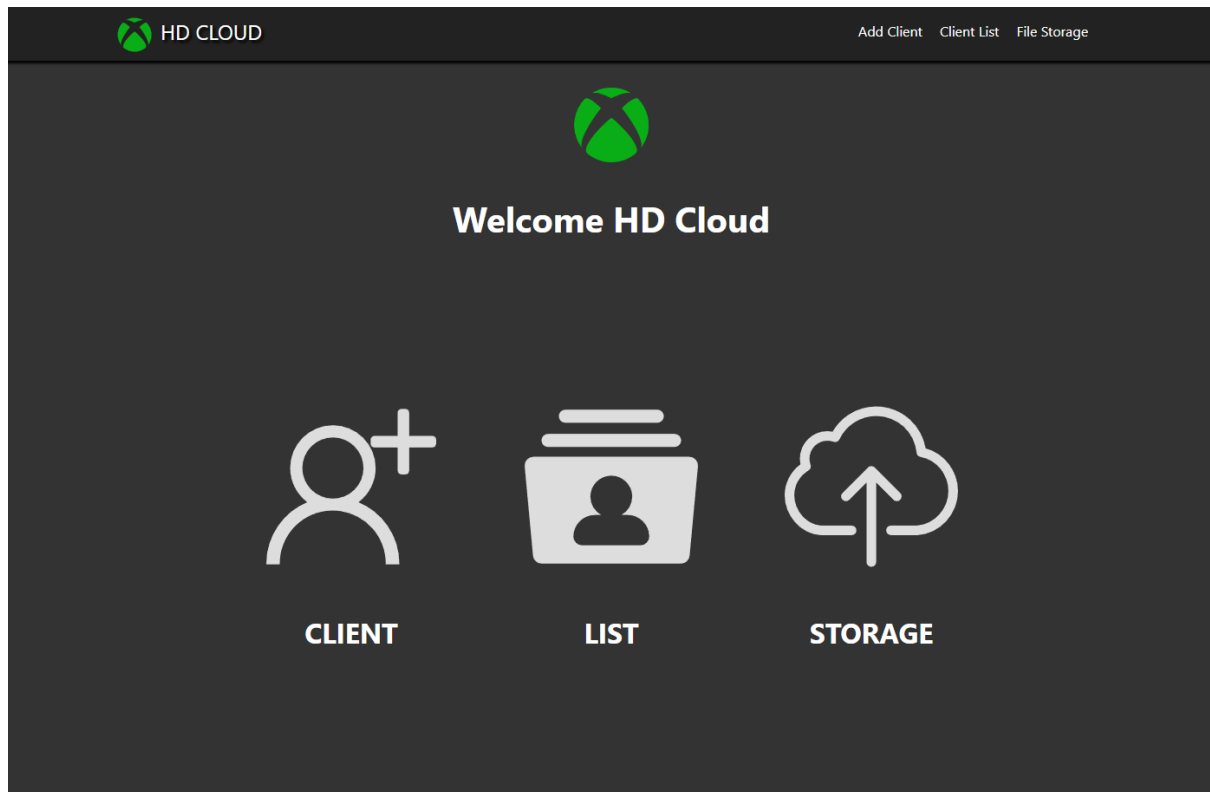


테스트결과서

1. React Test

1.1. 메인페이지



네이게이션 바의 링크와 메인페이지의 배너이미지 모두 다른 페이지로 정상적으로 연결됨

1.2. 고객 추가 페이지

HD CLOUD Add Client Client List File Storage

Add Client

이름
ex) 홍길동

성별
선택하세요

이메일
ex) example@gmail.com

출생년도
ex) 1991

등록하기

고객을 추가할 수 있는 페이지

Add Client

이름
홍길동

성별
여자

이메일 유효한 이메일을 입력하세요
test

출생년도
ex) 1991

등록하기

다른 모든 input란이 비어있거나 이메일에 '@'가 빠져있는 경우 오류 메시지가 라벨옆에 출력되어 유효하지 않은 데이터가 서버로 전송되는 것을 방지

Add Client

이름

성별

여자
▼

이메일

출생년도 유효한 출생년도를 입력하세요

등록하기

출생년도 또한 현재년도보다 크거나 현재년도 - 120년(역대 최장수 기록 122세를 기준으로 잡음)보다 작게 입력했을 경우 오류 메시지를 출력

Switch etc Google Naver Kakao
Danawa NaverShop Torrent Xcloud XboxSt

HD CLOUD
54.193.132.251 내용:
등록이 성공하였습니다.
확인

Add Client

이름

성별

여자
▼

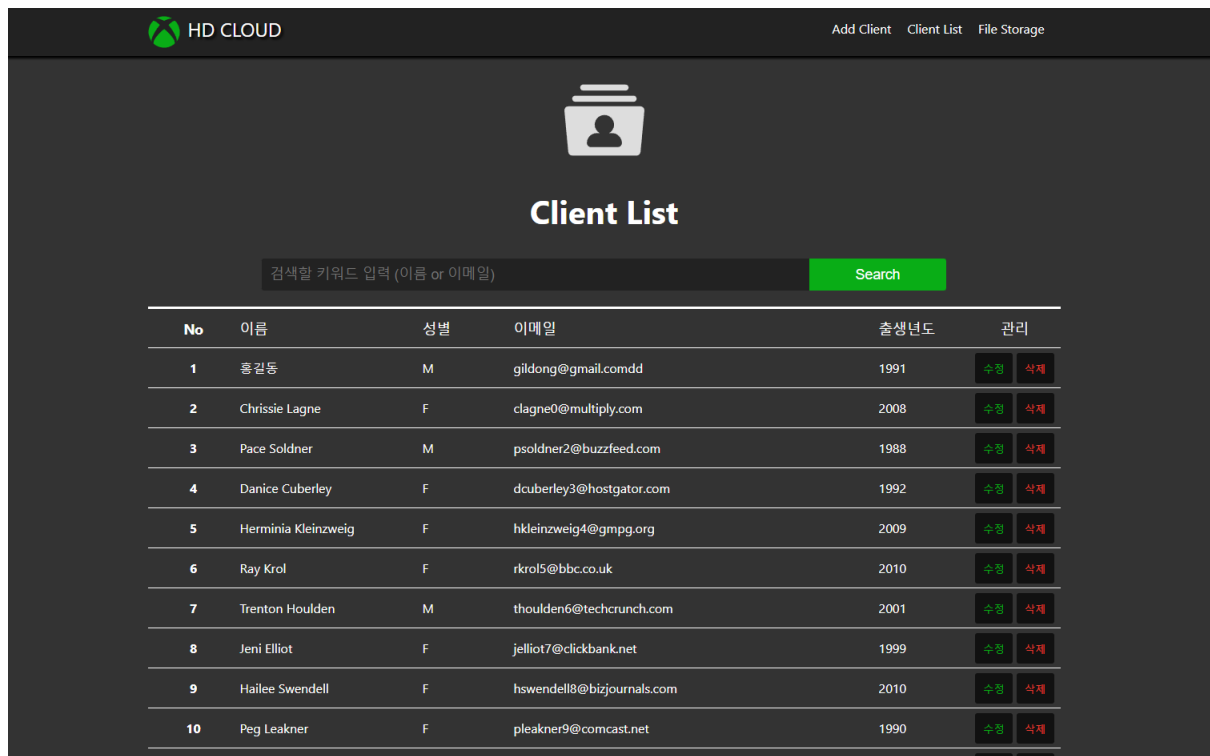
이메일

출생년도

등록하기

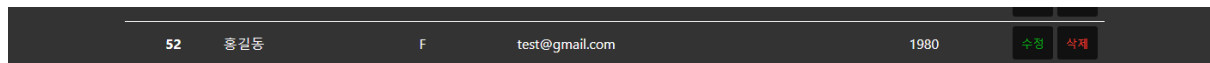
모두 유효한 값을 입력한 경우 등록에 성공했다는 메시지를 띄움

1.3. 고객 목록 페이지



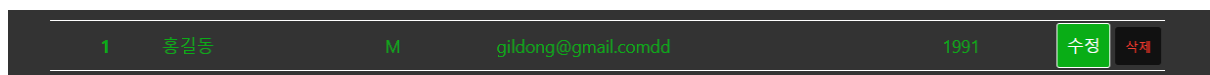
No	이름	성별	이메일	출생년도	관리
1	홍길동	M	gildong@gmail.comdd	1991	수정 삭제
2	Chrissie Lagne	F	clagne0@multiply.com	2008	수정 삭제
3	Pace Soldner	M	psoldner2@buzzfeed.com	1988	수정 삭제
4	Danice Cuberley	F	dcuberley3@hostgator.com	1992	수정 삭제
5	Herminia Kleinzweig	F	hkleinzweig4@gmpg.org	2009	수정 삭제
6	Ray Krol	F	rkrol5@bbc.co.uk	2010	수정 삭제
7	Trenton Houlden	M	thoulden6@techcrunch.com	2001	수정 삭제
8	Jeni Elliot	F	jelliot7@clickbank.net	1999	수정 삭제
9	Hailee Swendell	F	hswendell8@bizjournals.com	2010	수정 삭제
10	Peg Leakner	F	pleakner9@comcast.net	1990	수정 삭제

지금까지 추가한 고객들의 목록을 테이블 형태로 볼 수 있음

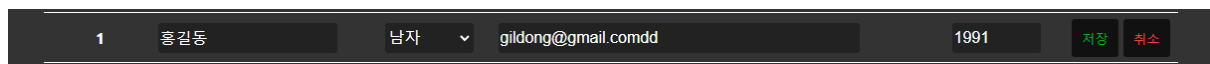


52	홍길동	F	test@gmail.com	1980	수정 삭제
----	-----	---	----------------	------	---------------------------------------

아까 고객 추가 페이지에서 테스트로 추가한 고객정보도 정상적으로 들어온 것을 확인 할 수 있음

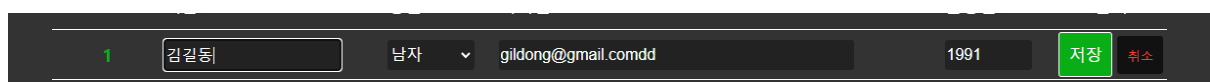


1	홍길동	M	gildong@gmail.comdd	1991	수정 삭제
---	-----	---	---------------------	------	---------------------------------------

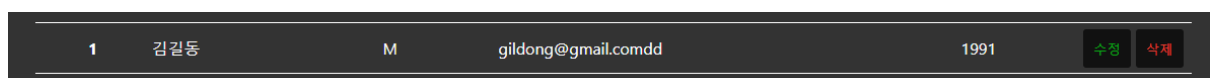


1	홍길동	남자	gildong@gmail.comdd	1991	저장 취소
---	-----	----	---------------------	------	---------------------------------------

고객 정보의 각 행에 있는 수정 버튼을 누르게 되면 수정모드로 들어가게 된다

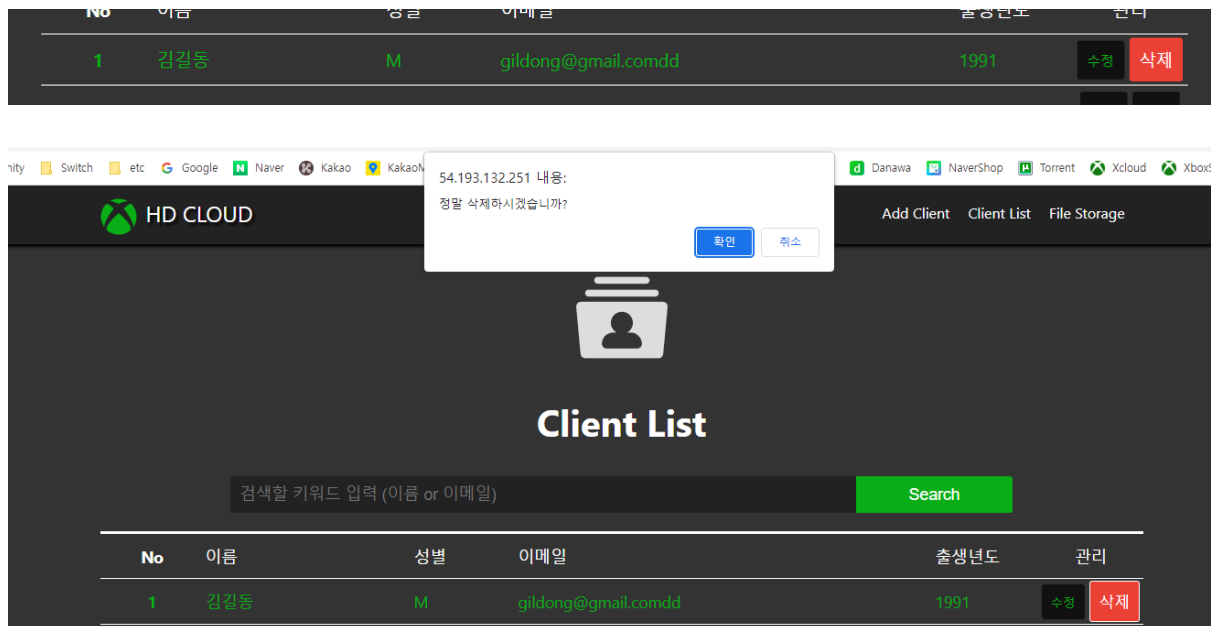


1	김길동	남자	gildong@gmail.comdd	1991	저장 취소
---	-----	----	---------------------	------	---------------------------------------



1	김길동	M	gildong@gmail.comdd	1991	수정 삭제
---	-----	---	---------------------	------	---------------------------------------

이름을 김길동으로 변경하고 저장을 누르게 되면 정상적으로 정보가 수정 된 것을 바로 확인할 수 있음



각 행에 있는 삭제 버튼을 누르게 되면 삭제하겠냐는 알림메시지를 보여준 뒤 한번더 확인인을 누를 시 데이터가 삭제되게 됨

검색할 키워드 입력 (이름 or 이메일)					
Search					
No	이름	성별	이메일	출생년도	관리
1	Chrissie Lagne	F	clagne0@multiply.com	2008	수정 삭제
2	Pace Soldner	M	psoldner2@buzzfeed.com	1988	수정 삭제
3	Danice Cuberley	F	dcuberley3@hostgator.com	1992	수정 삭제
4	Herminia Kleinzeig	F	hkleinzeig4@gmpg.org	2009	수정 삭제

가장 첫번째에 있던 김길동 정보가 사라진 것이 확인 됨

Client List					
검색할 키워드 입력 (이름 or 이메일)					
Search					
No	이름	성별	이메일	출생년도	관리
1	Chrissie Lagne	F	clagne0@multiply.com	2008	수정 삭제

검색 기능을 이용하여 이름이나 이메일을 검색할 수 있음

Client List

No	이름	성별	이메일	출생년도	관리
1	홍길동	F	test@gmail.com	1980	수정 삭제

마지막에 추가한 홍길동이 정상적으로 검색되는 것을 확인

Client List

No	이름	성별	이메일	출생년도	관리
1	Marjory Coorington	F	mcooringtonc@miitbeian.gov.cn	2005	수정 삭제
2	Nicko Pesek	M	npesekm@nih.gov	2010	수정 삭제
3	Prinz Euplate	M	peuplate18@dot.gov	2008	수정 삭제

gov로 검색할 경우 이메일에 gov를 가진 3개의 정보만 검색되는 것을 확인

1.4. 파일 관리 페이지

HD CLOUD

[Add Client](#)
[Client List](#)
[File Storage](#)

File Management

업로드할 파일 선택 (파일당 20MB 제한)

선택된 파일 없음

UPLOAD

exe

SessionManager_4.exe

Download

png

주기율표.png

Download

exe

AOMEI Partition Assistant.exe

Download

txt

필독.txt

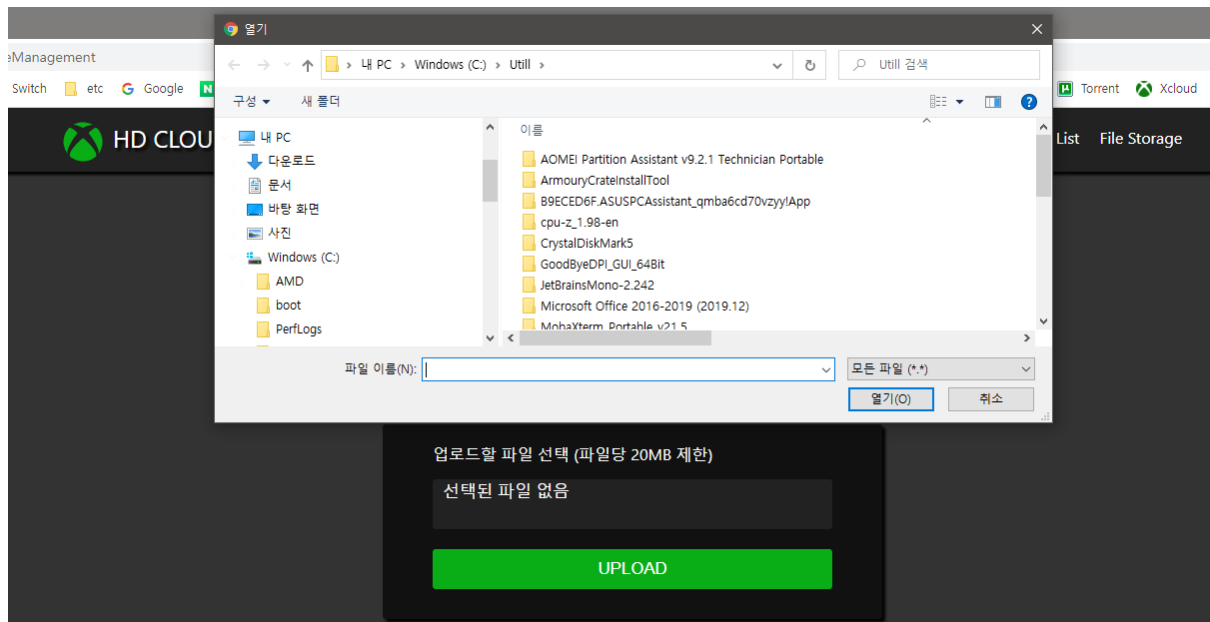
Download

EXE

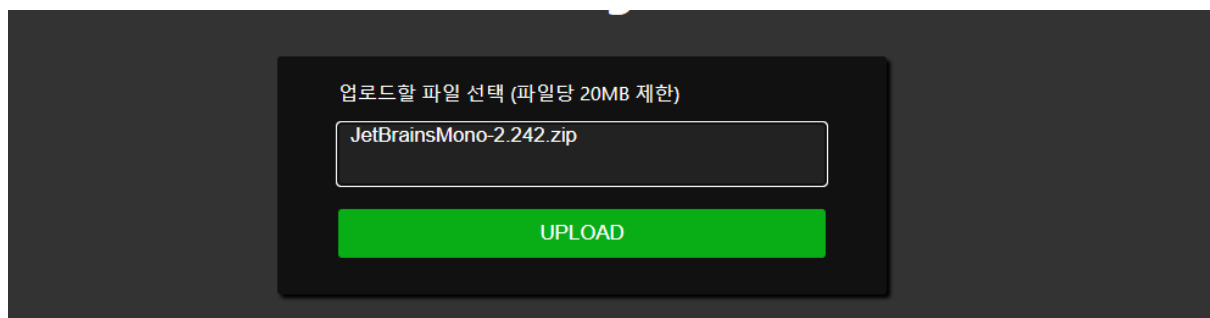
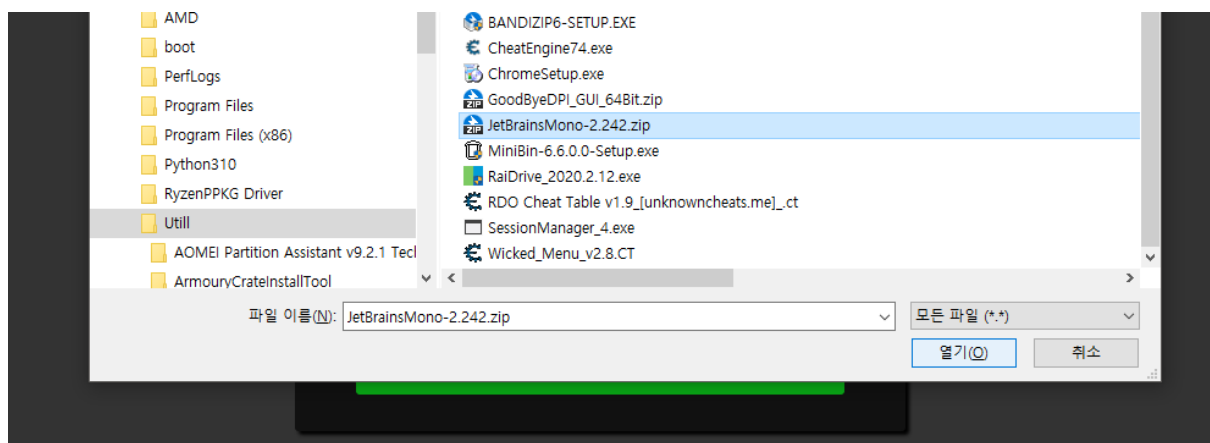
BANDIZIP6-SETUP.EXE

Download

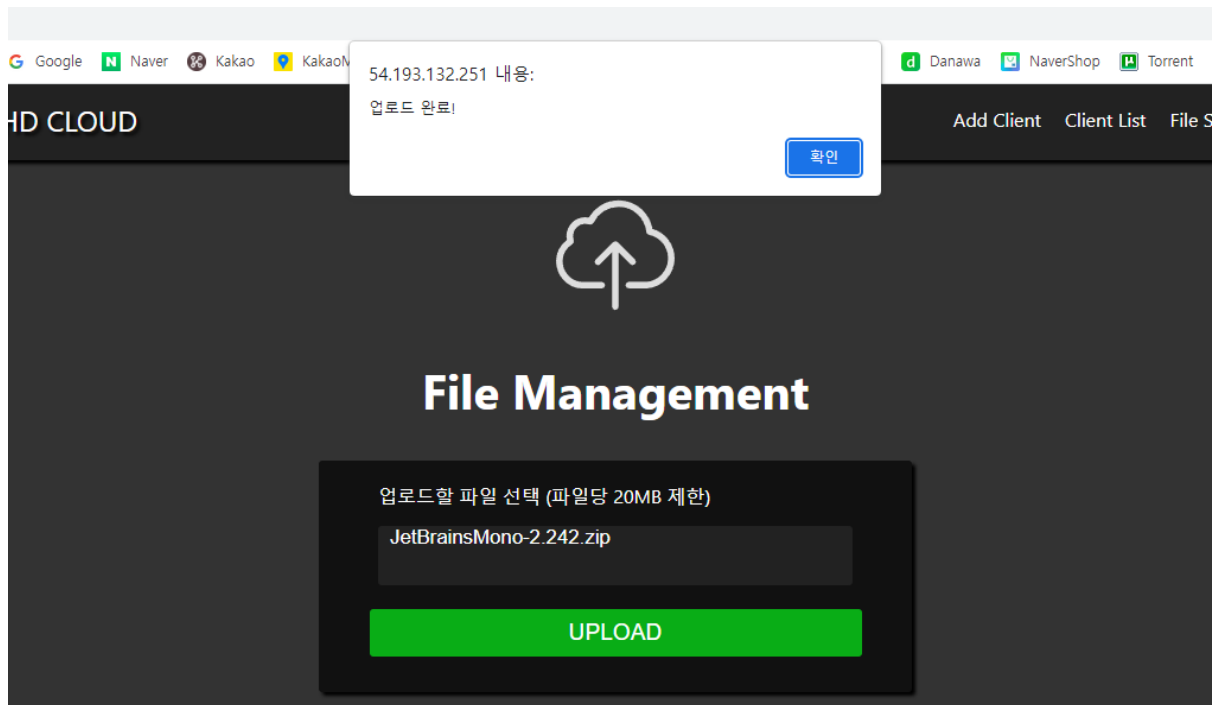
파일을 업/다운로드하는 기능을 이용할 수 있는 페이지



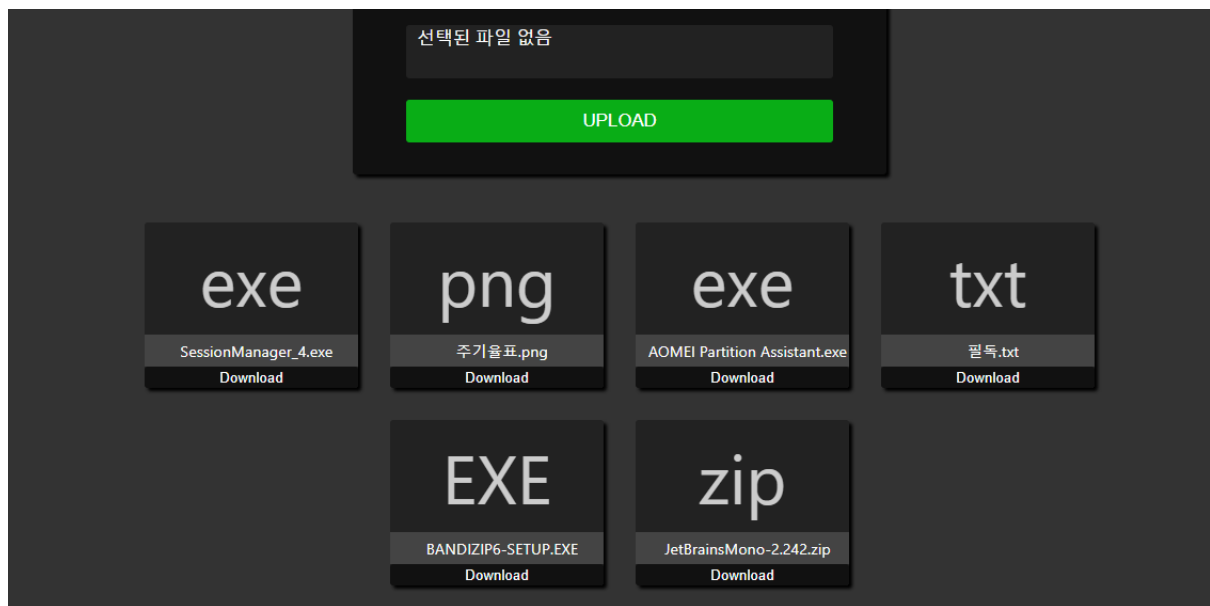
업로드 할 파일을 선택하는 inputBox를 클릭하면 파일을 선택하는 창이 뜬



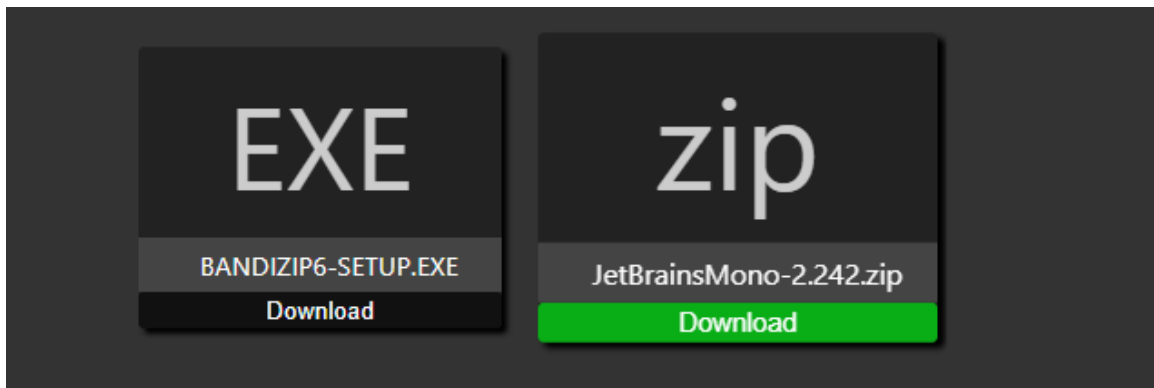
파일을 선택하고 열기를 누르면 inputBox에 해당 파일이 정상적으로 선택 됨



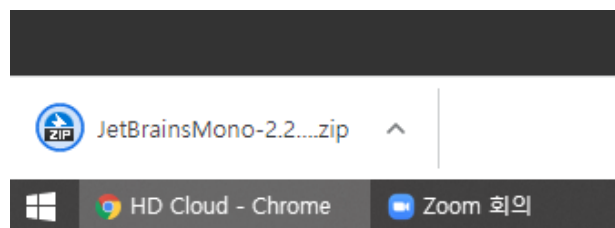
UPLOAD버튼을 누르게 되면 업로드 완료 메시지가 출력



파일 목록 카드들의 마지막에 방금 올린 파일이 추가 된 것을 확인할 수 있음



방금 업로드 한 파일을 다운로드를 누르게 되면



정상적으로 파일이 다운되는 것을 확인

2. Springboot Test (아래 Test code 모두 Pass함)

2.1. Client model test code

```
package com.cloud.file_management_system_backend.client.model;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

class ClientTest {
    @Test
    void createTest() {
        Client client = Client.builder()
            .clientName("tester")
            .birthYear(1990)
            .gender('M')
            .email("test@test.com")
            .id(1001L)
            .build();

        assertEquals(client.getClientName(), "tester");
        assertEquals(client.getBirthYear(), 1990);
        assertEquals(client.getGender(), 'M');
        assertEquals(client.getEmail(), "test@test.com");
        assertEquals(client.getId(), 1001L);
    }
}
```

2.2. Client repository test code

```
package com.cloud.file_management_system_backend.client.repository;

import com.cloud.file_management_system_backend.client.model.Client;
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.boot.test.mock.mockito.SpyBean;

import static org.junit.jupiter.api.Assertions.assertEquals;

@SpringBootTest
class ClientRepositoryTest {

    @SpyBean
    private ClientRepository clientRepository;

    @Test
    void saveTest() {
        clientRepository.save(Client.builder()
            .email("test@test.com")
            .gender('M')
            .birthYear(2000)
            .clientName("tester")
            .build());
    }

    @Test
    void findTest() {
        Client client = Client.builder()
            .email("test@test.com")
            .gender('M')
            .birthYear(2000)
            .clientName("tester")
            .build();

        assertEquals(clientRepository.findById(1L).get().getClientName(), client.getClientName());
        assertEquals(clientRepository.findById(1L).get().getGender(), client.getGender());
        assertEquals(clientRepository.findById(1L).get().getEmail(), client.getEmail());
        assertEquals(clientRepository.findById(1L).get().getBirthYear(), client.getBirthYear());
    }

    @Test
    void findAllTest() {
        assertEquals(clientRepository.findAll().size(), 4);
    }

    @Test
    void deleteTest() {
        assertEquals(clientRepository.findAll().size(), 4);
        clientRepository.deleteById(1L);
        assertEquals(clientRepository.findAll().size(), 3);
    }

    @Test
    void searchTest() {
        Client client1 = Client.builder()
            .email("test@test.com")
            .gender('M')
            .birthYear(2000)
            .clientName("tester")
            .build();
        clientRepository.save(client1);
        Client client2 = Client.builder()
```

```

        .email("admin@test.com")
        .gender('M')
        .birthYear(2000)
        .clientName("admin")
        .build();
clientRepository.save(client2);
Client client3 = Client.builder()
        .email("good@test.com")
        .gender('M')
        .birthYear(2000)
        .clientName("admin")
        .build();
clientRepository.save(client3);

assertEquals(clientRepository.findByClientNameContaining("admin").size(), 2);
assertEquals(clientRepository.findByEmailContaining("good@test.com").size(), 1);
    }
}

```

2.3. Client service test code

```

package com.cloud.file_management_system_backend.client.service;

import com.cloud.file_management_system_backend.client.model.Client;
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.boot.test.mock.mockito.SpyBean;

import static org.junit.jupiter.api.Assertions.assertEquals;

@SpringBootTest
class ClientServiceTest {
    @SpyBean
    private ClientService clientService;

    @Test
    void saveTest() {
        clientService.saveClient(
            Client.builder()
                .email("test@test.com")
                .gender('M')
                .birthYear(2000)
                .clientName("tester")
                .build()
        );
    }

    @Test
    void findTest() {
        Client client = Client.builder()
            .email("test@test.com")
            .gender('M')
            .birthYear(2000)
            .clientName("tester")
            .build();
        assertEquals(clientService.findClient(1L).getClientName(), client.getClientName());
        assertEquals(clientService.findClient(1L).getGender(), client.getGender());
        assertEquals(clientService.findClient(1L).getEmail(), client.getEmail());
        assertEquals(clientService.findClient(1L).getBirthYear(), client.getBirthYear());
    }
}

```

```

@Test
void findAllTest() {
    assertEquals(clientService.findAllClient().size(), 4);
}

@Test
void deleteTest() {
    assertEquals(clientService.findAllClient().size(), 4);
    clientService.deleteClient(1L);
    assertEquals(clientService.findAllClient().size(), 3);
}

@Test
void searchTest() {
    Client client1 = Client.builder()
        .email("user@test.com")
        .gender('M')
        .birthYear(2000)
        .clientName("user")
        .build();
    clientService.saveClient(client1);
    Client client2 = Client.builder()
        .email("user@test.com")
        .gender('M')
        .birthYear(2000)
        .clientName("admin")
        .build();
    clientService.saveClient(client2);
    Client client3 = Client.builder()
        .email("good@test.com")
        .gender('M')
        .birthYear(2000)
        .clientName("good")
        .build();
    clientService.saveClient(client3);

    assertEquals(clientService.searchClient("admin").size(), 1);
    assertEquals(clientService.searchClient("user@test.com").size(), 2);
}
}

```

2.4. FileData model test code

```

package com.cloud.file_management_system_backend.file.model;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.*;

class FileDataTest {
    @Test
    void name() {
        FileData fileData = FileData.builder()
            .hashName("sfe876s6df567we8f7")
            .fileName("test.exe")
            .filePath("c:\\test\\sfe876s6df567we8f7")
            .id(55L)

```

```

        .build();

        assertEquals(fileData.getHashName(), "sfe876s6df567we8f7");
        assertEquals(fileData.getFileName(), "test.exe");
        assertEquals(fileData.getFilePath(), "c:\\test\\sfe876s6df567we8f7");
        assertEquals(fileData.getId(), 55L);
    }
}

```

2.5. FileData repository test code

```

package com.cloud.file_management_system_backend.file.repository;

import com.cloud.file_management_system_backend.file.model.FileData;
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.boot.test.mock.mockito.SpyBean;

import static org.junit.jupiter.api.Assertions.assertEquals;

@SpringBootTest
class FileDataRepositoryTest {

    @SpyBean
    private FileDataRepository fileDataRepository;

    @Test
    void saveTest() {
        fileDataRepository.save(
            FileData.builder()
                .hashName("sfe876s6df567we8f7")
                .fileName("test.exe")
                .filePath("c:\\test\\sfe876s6df567we8f7")
                .build()
        );
    }

    @Test
    void findTest() {
        FileData fileData = FileData.builder()
            .hashName("sfe876s6df567we8f7")
            .fileName("test.exe")
            .filePath("c:\\test\\sfe876s6df567we8f7")
            .build();

        assertEquals(fileDataRepository.findById(1L).get().getFileName(), fileData.getFileName());
        assertEquals(fileDataRepository.findById(1L).get().getHashName(), fileData.getHashName());
        assertEquals(fileDataRepository.findById(1L).get().getFilePath(), fileData.getFilePath());
    }

    @Test
    void findAllTest() {
        assertEquals(fileDataRepository.findAll().size(), 1);
    }

    @Test
    void deleteTest() {
        assertEquals(fileDataRepository.findAll().size(), 1);
        fileDataRepository.deleteById(1L);
        assertEquals(fileDataRepository.findAll().size(), 0);
    }
}

```

```
}  
}
```

2.6. FileData service test code

```
package com.cloud.file_management_system_backend.file.service;  
  
import com.cloud.file_management_system_backend.file.model.FileData;  
import org.junit.jupiter.api.Test;  
import org.springframework.boot.test.context.SpringBootTest;  
import org.springframework.boot.test.mock.mockito.SpyBean;  
  
import static org.junit.jupiter.api.Assertions.assertEquals;  
  
@SpringBootTest  
class FileDataServiceTest {  
    @SpyBean  
    private FileDataService fileDataService;  
  
    @Test  
    void saveTest() {  
        fileDataService.saveFileData(  
            FileData.builder()  
                .hashName("sfe876s6df567we8f7")  
                .fileName("test.exe")  
                .filePath("c:\\test\\sfe876s6df567we8f7")  
                .build()  
        );  
    }  
  
    @Test  
    void findTest() {  
        FileData fileData = FileData.builder()  
            .hashName("sfe876s6df567we8f7")  
            .fileName("test.exe")  
            .filePath("c:\\test\\sfe876s6df567we8f7")  
            .build();  
        assertEquals(fileDataService.findFileData(1L).getFilePath(), fileData.getFilePath());  
        assertEquals(fileDataService.findFileData(1L).getFileName(), fileData.getFileName());  
        assertEquals(fileDataService.findFileData(1L).getHashName(), fileData.getHashName());  
    }  
  
    @Test  
    void findAllTest() {  
        assertEquals(fileDataService.findAllFileData().size(), 1);  
    }  
  
    @Test  
    void deleteTest() {  
        assertEquals(fileDataService.findAllFileData().size(), 1);  
        fileDataService.deleteFileData(1L);  
        assertEquals(fileDataService.findAllFileData().size(), 0);  
    }  
}
```