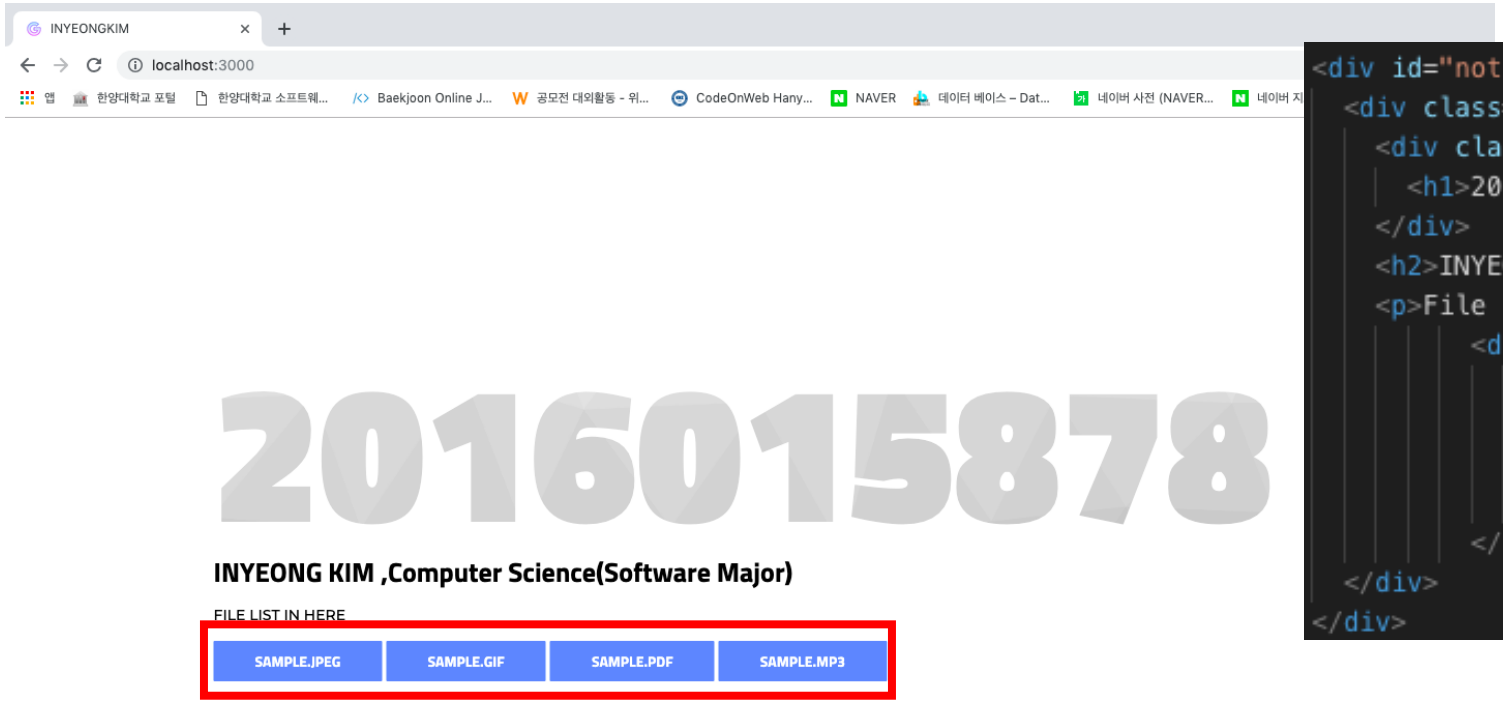
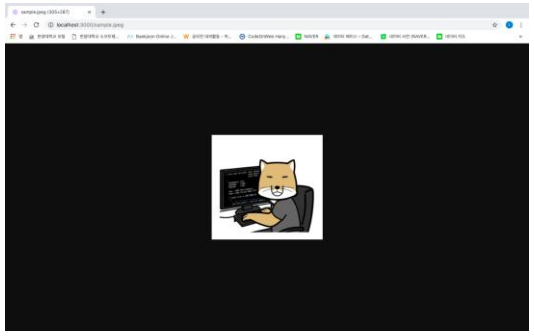
index.html

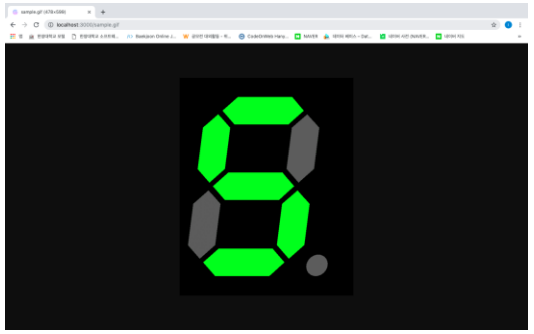



```
<div id="notfound">
  <div class="notfound">
    <div class="notfound-404">
      <h1>2016015878</h1>
    </div>
    <h2>INYEONG KIM ,Computer Science(Software Major)</h2>
    <p>File List in here</p>
          <div>
              <a href="./sample.jpeg">sample.jpeg</a>
              <a href="./sample.gif">sample.gif</a>
              <a href="./sample.pdf">sample.pdf</a>
              <a href="./sample.mp3">sample.mp3</a>
          </div>
  </div>
</div>
```

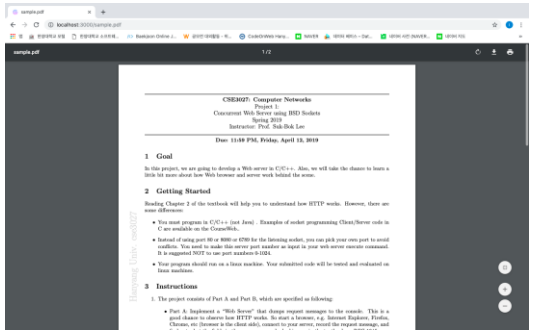
sample.jpe



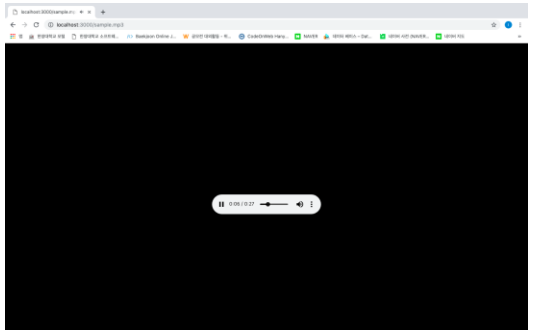
sample.gif



sample.pdf

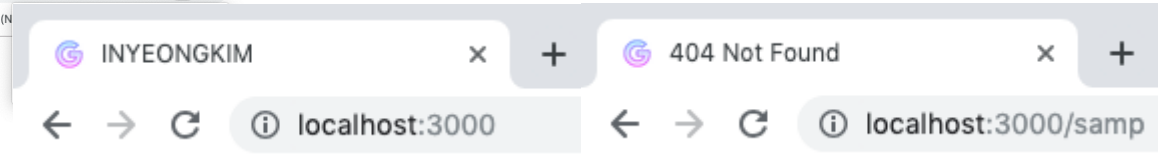


sample.mp3

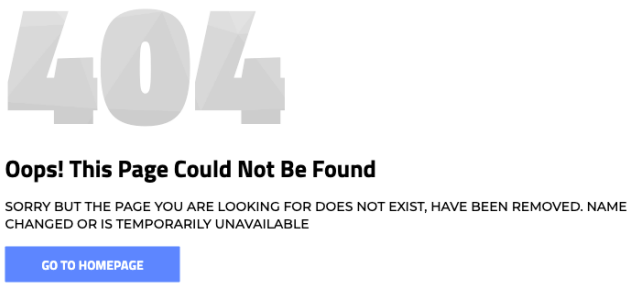


메인 페이지에서 저장된 파일의 목록을 확인가능하고, 버튼을 눌러 바로 이동 가능하도록 구현하였습니다.
(브라우저에 입력하여 접근하는 것 역시 가능합니다.)

404.html



Favicon 적용 모습입니다. 두 html파일 모두 확인 가능합니다.




```
<div id="notfound">
  <div class="notfound">
    <div class="notfound-404">
      <h1>404</h1>
    </div>
    <h2>Oops! This Page Could Not Be Found</h2>
    <p>Sorry but the page you are looking for do
    <a href="./">Go To Homepage</a>
  </div>
</div>
```

```
============================

# request :

GET /samp HTTP/1.1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintos
Accept: text/html,applicat
# rawPath : /samp
# path : ./samp
# contentType : text/html


# response :
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 3497
Connection: Alive



============================
```

잘못된 주소(디렉토리에 없는 파일) 입력 시 404.html 파일이 열립니다.
이 때 중간의 확인버튼을 누르면 index.html이 열리도록 구현했습니다.

make & socket reuse option

```
INYEONGs-MacBook-Air:project1_2016015878_Kim_Inyeong INYEONG$ ls
404.html        Makefile        README          favicon.ico     img             index.html      report.pptx     sample.gif      sample.jpeg     sample.mp3      sample.pdf      server.c
INYEONGs-MacBook-Air:project1_2016015878_Kim_Inyeong INYEONG$ make
gcc -c -o server.o server.c
gcc -o server server.o
INYEONGs-MacBook-Air:project1_2016015878_Kim_Inyeong INYEONG$ ls
404.html        README          img             report.pptx     sample.jpeg     sample.pdf      server.c
Makefile        favicon.ico     index.html      sample.gif      sample.mp3      server          server.o
INYEONGs-MacBook-Air:project1_2016015878_Kim_Inyeong INYEONG$ ./server 3000
```

make 명령어를 통해 server.c 파일을 컴파일 하는 과정과
./server 3000 명령으로 3000번 포트를 배정 받아 서버를 실행하는 과정입니다.

```c
    /* ctrl+c : release current port */
    int reuseAddress;
    reuseAddress = 1;
    setsockopt( sockfd, SOL_SOCKET, SO_REUSEADDR, (const char*)&reuseAddress, sizeof(reuseAddress) );
```

```
============================

^C
INYEONGs-MacBook-Air:project1_2016015878_Kim_Inyeong INYEONG$ ./server 3000
```

개발 중 프로세스를 종료한 뒤 바로 동일포트번호에 배정이 안되는 문제를 발견했습니다.
이 문제를 socket에 SO_REUSEADDR옵션을 추가하는 것으로 해결했습니다.

Path setting

```c
char *token = strtok(buffer, " "); //GET
if (token != NULL){
    char *rawPath = strtok(NULL, " ");
    printf("# rawPath : %s\n", rawPath);
    if ( rawPath != NULL ) {
        char *path;
        /* set path */
        if(strcmp(rawPath, "/") == 0){ //root
            path = (char *)malloc(strlen("./index.html") + 1);
            strcpy(path, "./index.html");
        }else{
            path = (char *)malloc(strlen(rawPath) + 1);
            sprintf(path, ".%s", rawPath);
        }
```

Buffer의 내용(request)를 method명과 path로 구분하기 위해 strtok 함수를 이용했습니다.

이 때 예상치 못한 요청(favicon, index.html 파일에 구성된 png파일을 요청하는 내용, 이외 알 수 없는 상의 이유 등)으로 인해 strtok함수가 null을 읽으려 시도하여 Segment fault가 자주 발생했습니다.

이 문제를 NULL을 참조한 경우 아래 단계로 아예 진행이 불가능하도록 설정하여 해결했습니다.

```
==========

# request :
GET /sample.pdf HTTP/1.1
Host: localhost:3000
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintos
Accept: text/html,ap
# rawPath : /sample.pdf
# path : ./sample.pdf
# contentType : application/pdf


# response :
HTTP/1.1 200 OK
Content-Type: application/pdf
Content-Length: 112088
Connection: Alive



==========

# request :
GET /sample.gif HTTP/1.1
Host: localhost:3000
Connection: keep-alive
Purpose: prefetch
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macint
A
# rawPath : /sample.gif
# path : ./sample.gif
# contentType : image/gif


# response :
HTTP/1.1 200 OK
Content-Type: image/gif
Content-Length: 75870
Connection: Alive



==========
```

```
==========

# request :
GET /sample.jpeg HTTP/1.1
Host: localhost:3000
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macint
Accept: text/html,a
# rawPath : /sample.jpeg
# path : ./sample.jpeg
# contentType : image/jpeg


# response :
HTTP/1.1 200 OK
Content-Type: image/jpeg
Content-Length: 45415
Connection: Alive



==========

# request :
GET /sample.mp3 HTTP/1.1
Host: localhost:3000
Connection: keep-alive
Accept-Encoding: identity;q=1,
User-Agent: Mozilla/5.0 (Macin
chrome-proxy
# rawPath : /sample.mp3
# path : ./sample.mp3
# contentType : audio/mp3


# response :
HTTP/1.1 200 OK
Content-Type: audio/mp3
Content-Length: 443926
Connection: Alive



==========
```

pdf. gif. jpeg. mp3 파일 요청과 그에 대한 응답입니다.
각 형식에 따라 content type에 맞게 요청을 처리합니다.

```c
char *getContentType(char *contentPath){
    char *extents[] = { ".html", ".jpeg", ".gif", ".mp3", ".pdf", ".ico" };
    char *contentType[] = { "text/html", "image/jpeg", "image/gif", "audio/mp3", "application/pdf", "image/x-icon" };
    char *res = contentType[0];
    int len = (int)(sizeof(extents) / sizeof(extents[0]));

    for(int i=0; i<len; i++){
        if(strstr(contentPath, extents[i]) != NULL){
            res = (char *)malloc(strlen(contentType[i]) + 1);
            strcpy(res, contentType[i]);
            break;
        }
    }
    return res;
} //getContentType
```

```c
int file = open(path, O_RDONLY);
if(file < 0){ //not exist
    file = open("./404.html", O_RDONLY);
    contentType = (char *)malloc(strlen("text/html") + 1);
    strcpy(contentType, "text/html");
}

// get file size and reset cursor
int fileSize = lseek(file, 0, SEEK_END);
lseek(file, 0, SEEK_SET);
```

Path를 읽은 후 response에 파일 사이즈 lseek로 측정하고.
해당 커서를 다시 원위치로 맞추어 오류가 발생하지 않도록 구현했습니다.