번호	요구사항 ID	기능사항 ID	기능사항 명
1	FR-101	FS-101	메인 로고 클릭 시 메인페이지 Redirect 기능
	React-router-dom 의 Link, Route, Routes 모듈을 사용했습니다. Link로 만든 페이지 좌측상단의 로고를 클릭할 시 Route로 컴포넌트화 되어있는 Main.js를 라우팅해서 페이지에 띄워줍니다.		
2	FR-102	FS-102	기부페이지 Redirect 기능
	React-router-dom 의 Link, Route, Routes 모듈을 사용했습니다. 페이지 하단 푸터의 Link로 만든 Donation링크를 클릭하면 Route로 컴포넌트화 되어있는 Donation.js를 라우팅해서 페이지에 띄워줍니다.		
3	FR-103	FS-103	사용설명페이지 Redirect 기능
	React-router-dom 의 Link, Route, Routes 모듈을 사용했습니다. 페이지 상단 NAV바의 Link로 만든 User Guide링크를 클릭하면 Route로 컴포넌트화 되어있는 UserGuide.js를 라우팅해서 페이지에 띄워줍니다.		
4	FR-104	FS-104	문의페이지 Redirect 기능
	React-router-dom 의 Link, Route, Routes 모듈을 사용했습니다. 페이지 하단 푸터의 Link로 만든 Contact Us 링크를 클릭하면 Route로 컴포넌트화 되어있는 Contact.js를 페이지에 띄워줍니다.		
5	FR-104-01	FS-104-01	문의메일 전송 기능
	Contact.js에서 handleSubmit 함수로 Axios를 통해 '/send-email'로 POST 요청을 보내면, Express 서버에서 sendMail.js가 실행되고, nodemailer로 네이버 SMTP 서버를 통해 이메일을 전송합니다. 성공 시 사용자에게 응답이 표시됩니다.		
6	FR-105	FS-105	개발팀 Github페이지 Redirect기능
	페이지 하단 푸터의 a링크태그로 만든 Team Github 링크를 클릭하면 개발팀 Github를 Redirect하여 새 창을 띄워줍니다.		
7	FR-106	FS-106	바이러스탐지 페이지 Redirect기능
	React의 Link, Route, Routes 모듈을 사용했습니다. 페이지 우측 상단에 Link로 만든 Virus Scan링크를 클릭 시 VirusScan.js컴포넌트가 페이지에 라우팅됩니다.		
8	FR-106-01	FS-106-01	바이러스 탐지하고 구조출력기능
	VirusScan.js에서 Axios로 Express 서버에 POST 요청을 보내면 서버의 uploadFileScan.js가 실행됩니다. 이 파일에서는 multer와 GridFSBucket 모듈을 사용해 파일 데이터를 데이터베이스에 저장하고 저장된 파일의 정보를 VirusScan.js로 반환해줍니다.		
9	FR-107	FS-107	레퍼런스 페이지로 Redirect기능
	Main.js에서 레퍼런스 사이트인 VirusTotal 사이트의 이미지를 띄우고 해당 이미지를 클릭하면 VirusTotal 사이트로 새 창이 열리게끔 a링크 태그로 연결시킵니다.		
10	FR-108	FS-108	패킹 페이지 Redirect기능
	React의 Link, Route, Routes 모듈을 사용했습니다. 페이지 우측 상단에 Link로 만든 Packing링크를 클릭 시 Packing.js컴포넌트가 페이지에 라우팅됩니다.		
11	FR-108-01	FS-108-01	파일 다운로드 기능
	Packing.js에서 Axios로 Express 서버에 POST 요청을 보내면, 서버의 uploadFilePack.js가 실행됩니다. 이 파일에서는 multer와 GridFSBucket을 사용해 파일 데이터를 데이터베이스에 저장한 후, 모듈 서버에서 convert_NorToEnc.py를 사용해 파일을 암호화하고, uploadDB_Enc.py로 암호화된 파일을 encrypted_files DB에 저장합니다. 이후 uploadFilePack.js에서 해당 DB의 파일 데이터와 정보를 가져와 Packing.js로 반환합니다.		
12	FR-201	FS-201	DB 저장된 파일 불러오는 기능
	사용자가 웹 어플리케이션을 사용해서 서비스 사이트에서 파일 업로드를 합니다.해당 파일은 MongoDB의 ProtectFiles/normalfile/files에 업로드 됩니다. 이 기능은 위의 DB를 모니터링 하여 DB에 파일이 업로드 될 때 해당 파일을 모듈 서버에 다운로드 하는 기능입니다. DB를 모니터링 하는 기능은 MonitorDB.py에 있는 Monitor_files() 함수이며 해당 파일을 모듈서버에 저장하는 기능은 downloadDB_Nor.py에 있는 store_file()함수입니다. Store_file()함수는 DB에 올라온 사용자 파일을 모듈 서버에 다운로드 하고 다운로드 된 이름을 그 파일의 시그니처ID(임의 생성값)으로 변경 후 날짜별로 폴더를 만들어 orgin_files 폴더에 저장합니다.		
	FR-202	FS-202	파일 PE 구조 각 섹션 값 DB 저장 기능

13	Module 서버에 다운로드 된 사용자 파일의 PE를 분석하고 MongoDB에 저장하는 기능입니다. 또한 암호화 된 파일의 PE를 분석하고 MongoDB에 저장하는 기능도 포함되어있습니다. 1. uploadDB_NorPE.py의 store_normal_file_pe_info()를 통해서 사용자가 올린 원본 파일 PE를 저장합니다. 2. uploadDB_EncPE.py의 store_encrypted_file_pe_info()를 통해서 암호화된 파일의 PE를 저장합니다. PE를 분석하는 부분은 check_Enc.py의 analyze_pe_sections()를 사용합니다.		
14	FR-203	FS-203	파일 암호화 여부 판단 기능
	사용자가 mongoDB에 업로드한 파일의 암호화 여부를 판단하는 기능입니다. 암호화 및 난독화를 적용했을 때에는 평문보다 암호문의 엔트로피가 높게 나온다는 사실을 기능으로 구현했습니다. 파일 업로드의 형식이 PE 파일이기 때문에 FS-202에서 PE구조를 분석하는 모듈과 연계하여 구현했습니다. 각 섹션의 값을 분석하여 각 섹션 별로 암호화가 적용되어 있는지의 여부를 판단하여 DB에 업로드 합니다. Check_Enc.py의 analyze_pe_sections()함수를 사용해 PE의 각 섹션 값을 추출하고 check_file_encryption() 함수를 통해 추출된 섹션의 암호화 여부를 판단합니다.		
15	FR-204	FS-204	프로텍터를 통한 파일 암호화 기능
	FS-202의 기능으로 파일 다운로드에 성공하게 되면 converNorToEnc.py의 encrypt_with_themida()라는 함수가 호출됩니다. Themida Protector는 console 명령어를 통해 protector를 실행할 수 있습니다. 이 점을 활용하여 windows bat 파일을 생성해 python코드 내에서 bat 파일을 통해 프로텍팅을 할 수 있게 했습니다. 해당 bat파일은 themida_encrypt.bat 파일입니다. themida_encrypt.bat은 먼저 파일이 64bit인지 32bit인지를 sigcheck.exe를 통해서 확인합니다. 만약 32bit 프로그램이면 themida.exe로 프로텍팅을 진행하고 64bit 프로그램이면 themida64.exe로 프로텍팅합니다. 해당 파일은 서버에 protected_files 디렉토리에 저장됩니다.		
16	FR-205	FS-205	암호화된 파일 DB 업로드 기능
	FS-204에 의해 프로텍팅으로 암호화된 파일을 MongoDB에 저장합니다. 같이 저장하는 정보는 signature_id, original_filename, encrypted_filename, original_upload_time, encrypted_upload_time, upload_ip, filedata입니다.저장하는 모듈은 uploadDB_Enc.py의 store_encrypted_file()를 사용합니다.		
17	FR-301	FS-301	빅데이터를 통한 악성코드 블랙리스트 작성
	Malwarebazzar에서 가져온 2020년도부터 보고된 악성코드 dataset.csv를 통해 mongoDB vsapi DB에 info collection을 구축하는 기능입니다. Dataset.csv에서는 md5 값으로 보고된 악성코드를 구분합니다. 기존 dataset.csv에 있는 정보와의 virustotal api를 사용해 나온 정보와의 불균형을 해결하기 위해서 dataset.csv에서 악성코드 정보의 md5를 모듈에서 읽어와 virustotal api 기능 중 hash값으로 search하는 기능을 통해 info collection에 업데이트 합니다. process_hash.py의 process_hash() 함수를 통해 실행합니다. 이 과정에서 처리된 해시 값은 processed_hashes.json 파일로 저장되고 해당 파일에서 같은 해시 값이 있으면 DB에 해당 해시 데이터가 있는 것이므로 그 해시 값은 건너뜁니다. virustotal api는 각 사용자마다 api key로 query할 수 있는 사용량이 제한되어있기 때문에 windows server에 내장되어 있는 task schedular를 사용하여 매달 3일동안 사용량의 한계만큼 hash를 검색해 데이터를 재가공 할 수 있게 trigger를 걸어두었습니다.		
18	FR-305	FS-305	API 출력 정보 DB 서버 저장 기능
	Virustotal api를 통해 나온 데이터를 vsapi DB의 info collection에 저장하는 기능입니다. virustotal_api.py의 search_file_by_hash() 함수를 이용해 파일을 탐색하면 그 파일의 details정보와 behavior의 정보가 가공되지 않은 형태로 저장됩니다. 해당 정보를 data_converter.py의 convert_data()함수를 이용하여 mongoDB 저장구조에 맞게 만들어둔 template.json파일의 형식대로 api를 통해 가져온 정보를 가공을 합니다. 그 후 DB_handler.py의 upload_to_mongodb()함수를 이용해 해당 정보를 vsapi DB의 info collection에 업데이트합니다.		
19	FR-308	FS-308	블랙리스트 참조 기능
	사용자가 웹에 파일을 업로드하여 바이러스 탐지 기능을 사용하려고 할 때 해당 파일의 해시 값을 받아와 기존에 만들어둔 vsapi의 info collection에 해당 정보가 있는지 확인하는 기능입니다. 이 기능을 통해 사용자는 좀 더 빠르게 파일에 대한 분석 결과를 받을 수 있습니다. DB_handler.py의 check_hash_in_mongodb()를 통해서 블릭리스트 즉, info collection을 참하게 됩니다.		
	FR-311	FS-311	DB 서버에 올라온 파일 탐지 기능

<table>
<tr><td>20</td><td colspan="3">Vsapi DB에 files에 업로드 된 파일을 탐지하는 기능입니다. 사용자가 웹을 통해 파일을 업로드 하면 DB_handler.py의 watch_for_file_uploads()가 감지하여 process_new_file.py의 process_new_file()함수가 작동합니다. watch_for_file_uploads()는 올라온 파일의 데이터를 가져와 process_new_file()에 전달합니다. 해당 모듈은 windows server의 service로 동작하고 있습니다.
process_new_file()는 아래와 같이 동작합니다.
1. 가져온 md5 해시 값을 통해 vsapi info collection에 있는지 확인 (FS-308)
2. 해당 되는 데이터가 없다면 virustotal_api.py의 search_file_by_hash()를 통해 virustotal에 hash값으로 접근할 수 있는 정보인지 확인
3. 해당 되는 데이터가 없다면 virustotal_api.py의 upload_file_to_virustotal()을 이용해 해당 파일을 업로드
3-1. 위 과정 이후 search_file_by_hash_with_retry() 함수를 이용해 virustotal에 업로드 된 파일의 정보를 확인할 수 있도록 함
4. data_converter.py를 통해 데이터를 정제
5. vsapi DB의 info collection에 정제된 데이터 업데이트</td></tr>
<tr><td rowspan="2">21</td><td>FR-312</td><td>FS-312</td><td>파이썬 모듈 서버 구축</td></tr>
<tr><td colspan="3">SDD-001 문서의 모듈 서버 구성 참조</td></tr>
<tr><td rowspan="2">22</td><td>FR-313</td><td>FS-313</td><td>DB 서버 구축</td></tr>
<tr><td colspan="3">SDD-001 문서의 DB 서버 구성 참조</td></tr>
<tr><td rowspan="2">23</td><td>FR-314</td><td>FS-314</td><td>웹 서버 구축</td></tr>
<tr><td colspan="3">SDD-001의 Web서버 구성 참조</td></tr>
</table>