

프로젝트 명세서

NoSQL DB Data 암호화 입문

목차

1. 과제 개요	3
2. 과제 목표	4
3. 필수 지식 학습	5
4. 기능 명세	6
5. 산출물 제출	14
6. 심화 학습	14

1. 과제 개요



본 과제는 굉장히 가벼우면서도 파워풀한 Framework 인 Node.js 와 NoSQL 중 SSAFY 뿐만 아니라 현업에서의 사용성이 가장 많은 DBMS 중 하나인 Redis 를 사용하여 간단한 형태의 Data 암호화를 Server-Side 에서 실습 해보는 것이 주된 내용입니다.

2. 과제 목표

- 1) Node.JS 를 이해하고 Local 환경에 설치한다.
- 2) Node Library 세팅하여 web service 구현한다.
- 3) Redis DB 를 Local 환경에 설치한다.
- 4) View 화면 및 Server+DB 암호화 연동 부분을 실습한다.

3. 필수 지식 학습

아래 사이트들과 해당 키워드로 인터넷 Searching 하여 Node.JS 외의 다양한 서비스에 대해 익히고 NPM 과 같은 Package Manager 에 대해서도 학습합니다.

참고 자료

구분	제목	링크
이해	Node.JS	https://nodejs.org
이해	NPM	https://www.npmjs.com/
이해	Redis	https://redis.io/docs/install/install-redis/

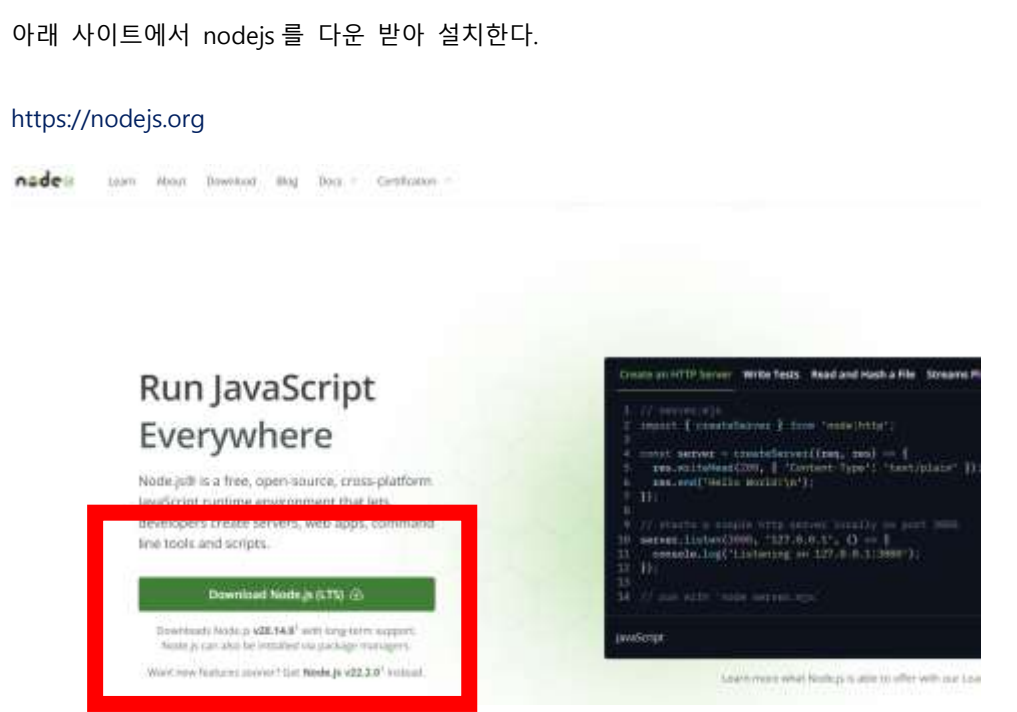
4. 기능 명세

1. 기능/과제 목록

Req.	Category
1	환경 구축 (nodejs 설치, npm 설치)
2	Index.js 생성 및 node init 진행
3	Redis 및 암호화 library 구현 및 web service 활성화
4	Redis DB 설치 및 서버 실행
5	암호와 Data DB Insert 및 View 구현


2. 기능/과제 상세

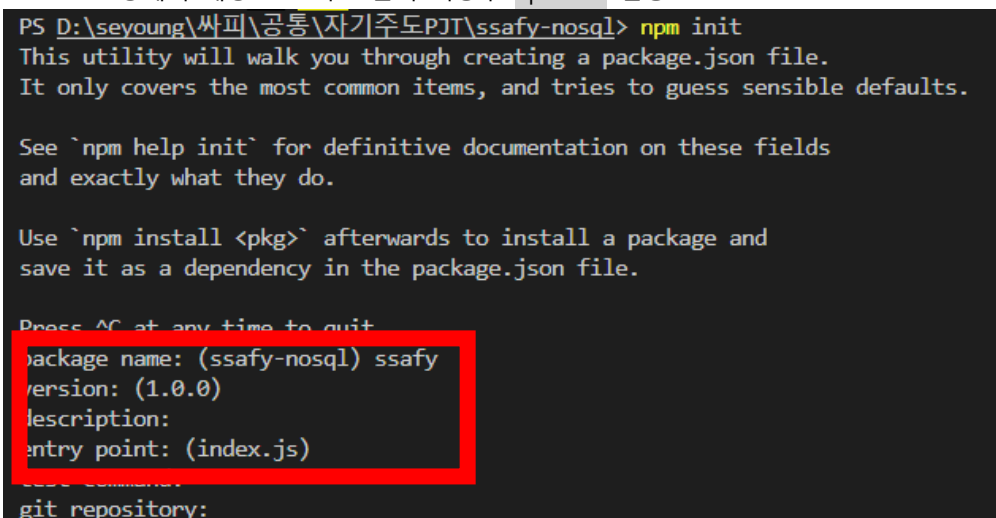
Req. 1. 환경 구축

Req. 1-1	환경 구축
기능 상세	<p>아래 사이트에서 nodejs 를 다운 받아 설치한다.</p> <p>https://nodejs.org</p>  <p>The screenshot shows the Node.js website with the heading 'Run JavaScript Everywhere'. Below the heading, it states 'Node.js® is a free, open-source, cross-platform JavaScript runtime environment that lets developers create servers, web apps, command line tools and scripts.' A green button labeled 'Download Node.js (LTS)' is highlighted with a red box. Below the button, it says 'Download Node.js v22.14.0 with long-term support. Node.js can also be installed via package managers. Want new features soon? Get Node.js v22.3.0 instead.'</p>

--	--

Req. 2. Index.js 생성 및 node init 진행

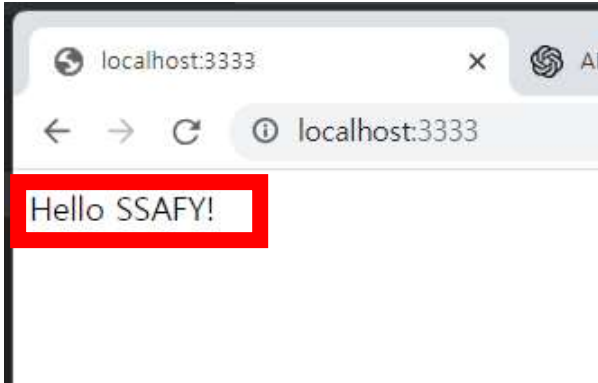
Req. 2-1	Index.js 생성 및 설정
기능 상세	<p>※ vscode(or any tools) 기본 사용법에 대한 사전 숙지가 필요하며, 이후 내용부터는 해당 tool에서의 작업분이 포함됨.</p> <p>project로 사용할 디렉토리를 하나 생성(예시는 'SSAFY-NOSQL')한후 해당 폴더 내부에 index.js 파일을 생성한다.</p> 

Req. 2-2	Node init 진행
기능 상세	<p>command 창에서 해당 프로젝트 폴더 이동후 npm init 실행.</p>  <p>Package name : (원하는 이름 지정, 이외는 optional) Version Description.....</p> <p>마지막 OK 클릭시 해당 폴더에 package.json 파일 생성됨.</p>

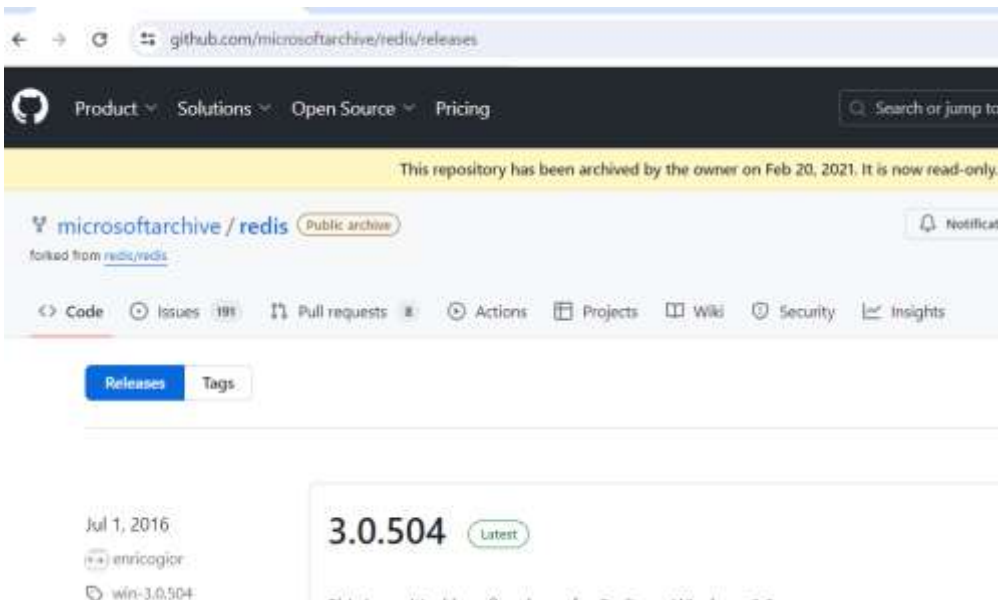
	<div> <div>SSAFY-NOSQL</div> <div>JS index.js</div> <div>{ } package.json</div> </div>	
--	---	--

Req. 3. library 구현 및 web service 활성화

Req. 3-1	Redis 및 암호화 library 구현
기능 상세	<p>command 창에서 <code>npm i express body-parser crypto redis</code> 실행.</p> <pre>PS D:\seyoung\싸피\공통\자기주도PJT\ssafy-nosql> npm i express body-parser crypto redis >> [.....] \ idealTree:ssafy-nosql: sill idealTree buildDeps</pre> <p>아까 생성한 index.js 파일 열고 아래의 구현 내용 작성.</p> <pre>const redis = require('redis'); const crypto = require('crypto'); const express = require('express'); const bodyParser = require('body-parser'); const key = crypto.randomBytes(32); // 32바이트 길이의 secret-키 const iv = crypto.randomBytes(16); // 초기화 벡터 생성 (16 바이트) const app = express(); const client = redis.createClient(); // JSON 파싱을 위한 미들웨어 설정 app.use(bodyParser.json()); app.use(bodyParser.urlencoded({ extended: true })); // 데이터를 암호화하는 함수 function encrypt(text) { const cipher = crypto.createCipheriv('aes-256-cbc', key, iv); let encrypted = cipher.update(text, 'utf8', 'hex'); encrypted += cipher.final('hex'); return { iv: iv.toString('hex'), encryptedData: encrypted }; } // 데이터를 복호화하는 함수 function decrypt(text) { const decipher = crypto.createDecipheriv('aes-256-cbc', key, iv); let decrypted = decipher.update(text, 'hex', 'utf8'); decrypted += decipher.final('utf8'); return decrypted; } app.get('/', function (req, res) { res.send("Hello SSAFY!"); }) // 서버 시작 const PORT = 3333; app.listen(PORT, () => { console.log(`서버가 포트 \${PORT}에서 실행 중입니다.`); });</pre> <p>port 번호는 임의로 지정 (예시는 3333).</p>

Req. 3-2	web service 활성화
기능 상세	<p>command 창에서 <code>node index.js</code> 실행 후 browser 에서 <code>localhost:3333</code> 접근하여 "Hello SSAFY!" 문구 출력 및 server 실행 확인.</p> 

Req. 4. Redis DB 설치 및 서버 실행

Req. 4-1	Redis DB 설치 및 서버 실행
기능 상세	<p>아래 사이트에서 windows 용 무설치 Redis 파일 다운로드.. https://github.com/microsoftarchive/redis/releases</p> 

사이트 접속후 3.0.504 (Latest) 부분에서 **Redis-x64-3.0.504.zip** 다운로드.



3.0.504 Latest

This is a critical bug fix release for Redis on Windows 3.0.
If you are running a previous version of 3.0 in a cluster configuration you should upgrade to 3.0.504 urg
The fix resolves a problem with the cluster fail-over procedure.

This released is based on antirez/redis 3.0.5 plus Windows-specific fixes.

See the [release notes](#) for details.

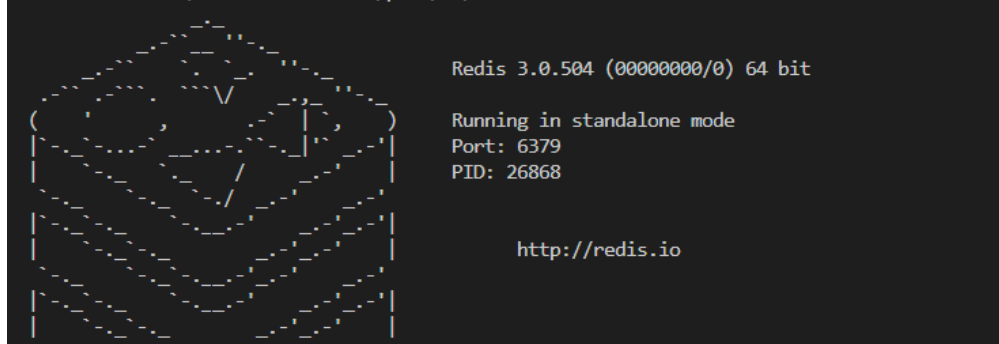
▼ Assets 4

Redis-x64-3.0.504.msi	6.42 MB
Redis-x64-3.0.504.zip	5.6 MB

이후 다운로드 받은 파일을 압축을 풀어두고 cmd 창 해당경로로 이동하여 .wredis-server 명령어로 redis server start 한다.

redis 는 기본적으로 windows 를 지원하지 않기 때문에 WSL 설치후 진행하여야 하나 본 명세서에서는 가장 간단한 무설치 형태를 선택하였습니다.

```
PS D:\seyoung\싸피\공통\자기주도PJT\ssafy-nosql\Redis-x64-3.0.504> .\redis-server
[26868] 15 Dec 16:53:14.417 # Warning: no config file specified, using the default config.
dis-x64-3.0.504\redis-server.exe /path/to/redis.conf
```



Redis 3.0.504 (00000000/0) 64 bit

Running in standalone mode
Port: 6379
PID: 26868

<http://redis.io>

Req. 5. 암호와 Data DB Insert 및 View 구현

Req. 5-1	Redis DB insert 구현
기능 상세	<p>index.js 파일에 client 부분 수정.</p> <pre>//const client = redis.createClient(); const client = redis.createClient(6379, 'localhost'); // 로컬 Redis 서버에 연결 client.connect();</pre> <p>redis 관련 function 구현부분 추가 작성.</p> <pre>// 비동기로 Redis에 데이터 저장 async function saveToRedis(key, data) { try { await client.set(key, JSON.stringify(data)); return true; } catch (err) { console.error('Redis 저장 중 에러 발생:', err); return false; } } // 암호화된 데이터를 Redis에 저장하는 API 엔드포인트 app.post('/insert', async (req, res) => { const { data } = req.body; const encryptedData = encrypt(data); // 암호화된 데이터를 Redis에 저장 const result = await saveToRedis('ssafyData', encryptedData); if (result) { res.status(200).send(`data : " + JSON.stringify(encryptedData) + "
 저장 완료!!`); } else { res.status(500).send('Insert error!'); } });</pre> <p>종료 초기화 부분 추가 작성</p> <pre>// node 종료시 호출 process.on('SIGINT', () => { client.quit(); console.log('서버 종료 및 Redis 클라이언트 종료'); process.exit(); });</pre>

Req. 5-1	View 화면 구현
기능 상세	<p>Index.html 파일을 생성하여 아래의 view 구현부분을 작성.</p> <pre> <!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <title>NoSQL 데이터 암호화</title> </head> <body> <h1>데이터 암호화 TEST</h1> <form id="dataForm"> <input type="text" id="dataInput" placeholder="데이터 입력"> <button type="submit">저장</button> </form> <div id="result"></div> <script> const form = document.getElementById('dataForm'); const input = document.getElementById('dataInput'); const resultDiv = document.getElementById('result'); form.addEventListener('submit', async (event) => { event.preventDefault(); const data = input.value; // 데이터를 서버에 전송 const response = await fetch('/insert', { method: 'POST', headers: { 'Content-Type': 'application/json' }, body: JSON.stringify({ data }) }); if (response.ok) { const result = await response.text(); resultDiv.textContent = result; } else { resultDiv.textContent = 'error!!'; } }); </script> </body> </html> </pre> <p>welcome 화면으로 index.html 을 지정하기 위해 아래 구문 추가</p>

	<pre>// index.html을 제공하기 위한 미들웨어 설정 app.use(express.static(__dirname)); // 현재 디렉터리 지정</pre> <p>node 서버 재시작 후 localhost:3333 재접속시 view 화면 정상접속 확인.</p>  <p>임의의 데이터 입력후 저장, reponse data 확인</p>  <p>(예시는 ㄹㅇㄴ). </p>
--	--

Req. 5-2	Redis DB 에서 Data 확인
기능 상세	<p>cmd 창에서 redis 폴더 이동후 redis-cli.exe -h localhost -p 6379 입력하여 redis client 진입한다</p> <pre>D:\seyoung\싸피\공통\자기주도PJT\ssafy-nosql\Redis-x64-3.0.504> redis-cli.exe -h localhost -p 6379</pre> <p>간단한 text 모델을 사용하였습니다. (예시 "text-davinci-003") 필요시 다양한 모델 리스트를 확인하여 적용 하시면 됩니다. prompt 를 바꿔서 질문을 수정 할 수 있습니다. 마지막의 여러 답변 중 첫번째 답변만 보이도록 지정한 부분도 응용해보세요.</p> <p>redis 에서 결과값 확인.</p>

	<pre>localhost:6379> keys * 1) "ssafyData" localhost:6379> get ssafyData "{\"iv\":\"c98c6b5a77a12f0b09bc00ac79859e2a\",\"encryptedData\":\"db255982d5c2991f2a04c76587cda7e4\"}"</pre>
--	---

5. 산출물 제출

https://lab.ssafy.com/s12-study/seasonal_fesw 의 “산출물 제출 가이드”.docx 참고

산출물 제출	<p>“산출물 제출 가이드”를 참고하여 소스 파일(총 2 개)와 README.md 파일을 제출한다. 최종적으로 작성한 index.js, index.html 파일이 해당하며 README.md 파일에는 수행한 command 창 의 결과를 캡쳐해서 올리고 정리한 내용들을 작성한다.</p>
--------	--

6. 심화 학습

기본 학습은 Data 암호화 후 Redis DB 에 저장하여 View 화면으로 보는 형태의 실습으로 이루어져 있습니다. 실제 암호화된 Data 의 복호화도 시도해서 다양한 형태의 암호복호화 시스템을 구현해 보세요.