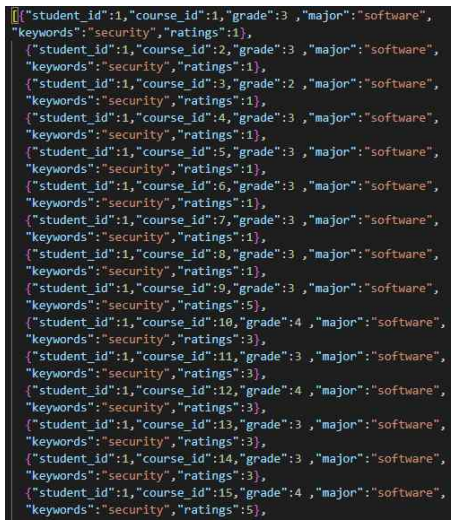
강의 추천 시스템 사용 방법

우선, 사용자의 학과major가 software, 관심분야Keywords가 Security or Network인 경우에만 사용할 수 있다. 프로토타입이니까 이정도 만으로 충분할 듯. (다른 과 다른 관심분야를 추가하려면 지금까지 한 작업을 한번 더 해야하며 설령 한번 더 한들 MF 알고리즘의 정확도 또한 보장할 수 없다 ㄷㄷ...)

1. 관심분야가 Security인지, Network인지 선택한다. (설명을 위해서 Security로 설정 후 설명)
   - Security인 경우 RatingData_Train_Security / RatingData_Test_Security / Result_Security 파일 사용
   - Network인 경우 RatingData_Train_Network / RatingData_Test_Network / Result_Network 파일 사용

2. RatingData_Train_Security 파일에 '학습'할 Data를 넣을 수 있고 현재 10명의 학생의 각각 40개의 강의에 대한 ratings을 저장해두고 있다.



RatingData_Train_Security에 들어있는 Data 중 student_id = 1의 Data 일부

student_id	course_id	grade	major (student 학과)	keywords (student 관심분야)	ratings
1	1~40	1~4	software	security	1~5

student_id, course_id를 index로 사용하기 때문에 2017000001, 21111 등 실제 학번이나 실제 강의 코드로 수정하면 안된다. 웬만하면 RatingData_Train 파일은 수정하지 않는 것이 좋을 듯 하다.

프로토타입이므로 값을 보기 좋게, 명확하게 내기 위해서 해당 Data들은 모두 일정한 값의 규칙을 가진다.

3. RatingData_Test_Security 파일은 추천 시스템을 사용한 결과를 얻고 싶은 student에 대한 Data이다.

RatingData_Test_Security에 들어있는 student_id = 2017000001에 대한 Data 일부

해당 파일의 Data는 2017000001이라는 학번을 가지고 있는 학생의 40개의 강의에 대한 ratings이 저장되어있다. 이때, course_id 1~27 까지는 ratings 값이 1~5 사이에 있지만 course_id 28~40 까지는 ratings 값이 0이다.

MF 알고리즘은 학생이 지금까지 수강한 course_id 1~27 까지의 ratings 값을 통해 아직 수강하지 않은 course_id 28~40 까지의 예상 ratings을 유추한다. 따라서 아직 수강하지 않은, 추천받고싶은 강의에 대한 ratings을 0으로 설정하는 것이다.

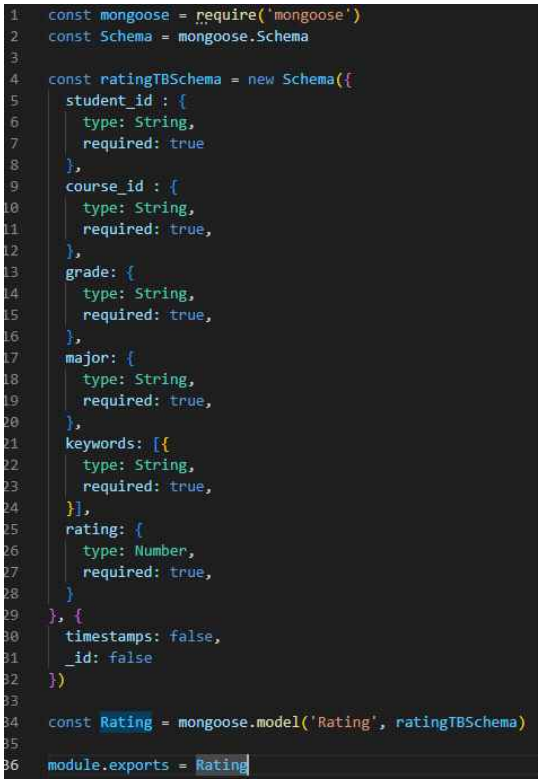
따라서 프론트엔드/백엔드에서 추천 시스템을 사용할 경우, RatingData_Test_Security 파일을 자신이 원하는 값으로 설정해야한다. 이때 앞서 RatingData_Train_Security 파일의 Data들은 모두 일정한 값의 규칙을 따른다고 설명했다.

Security Data 규칙	
course_id	ratings
1~8	1에 가까운 값
9	5
10~14	3에 가까운 값
15	5
16~27	2에 가까운 값
28	5
29~34	4에 가까운 값
35	5
36~39	3에 가까운 값
40	5

Network Data 규칙	
course_id	ratings
1~3	4에 가까운 값
4	5
5~8	3에 가까운 값
9	5
10~19	2에 가까운 값
20	5
21~26	1에 가까운 값
27	5
28~39	4에 가까운 값
40	5

해당 규칙을 최대한 따라서 RatingData_Test_Security의 Data를 채우고 MF 알고리즘을 돌리면, ratings 0을 넣은 course_id의 예상 ratings 값이 규칙과 비슷하게 이상적으로 나온다.

4. MF 알고리즘이 끝나면 ‘ratings 0을 넣은 강의들의 예상 ratings이 채워진 Data’가 Result_Security.json 파일을 생성하여 그곳에 저장된다. 그거 갖다 쓰면 된다. 참고로 모든 파일들의 Data 구조는 ratingTB.js 의 Data 구조를 따른다. 입력도 ratingTB.js 구조에 맞게 넣어야하고, 출력도 ratingTB.js 구조에 맞게 출력된다.



```
const mongoose = require('mongoose')
const Schema = mongoose.Schema

const ratingTBSchema = new Schema({
  student_id : {
    type: String,
    required: true
  },
  course_id : {
    type: String,
    required: true,
  },
  grade: {
    type: String,
    required: true,
  },
  major: {
    type: String,
    required: true,
  },
  keywords: [{
    type: String,
    required: true,
  }],
  rating: {
    type: Number,
    required: true,
  }
}, {
  timestamps: false,
  _id: false
})

const Rating = mongoose.model('Rating', ratingTBSchema)

module.exports = Rating
```

ratingTB.js 구조

student_id / course_id / grade / major / keywords / rating