**ООО «Натлекс»**

185000 Карелия г. Петрозаводск, ул. Кирова, д. 8Б (4-ый этаж, помещение 7)

[www.natlex.team](https://natlex.team/) • [info@natlex.ru](mailto:info@natlex.ru)

**Test Task for backend developer**

1) Add REST CRUD API for Sections and GeologicalClasses. Each Section has structure:

{

“name”: “Section 1”,

“geologicalClasses”: [

{ “name”: “Geo Class 11”, ”code”: “GC11” },

{ “name”: “Geo Class 12”, ”code”: “GC12” }, ...

]

}

2) Add API GET /sections/by-code?code=... that returns a list of all Sections that have geologicalClasses with the specified code.

3) Add APIs for importing and exporting XLS files. Each XLS file contains headers and list of sections with it’s geological classes. Example:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Section name | Class 1 name | Class 1 code | Class 2 name | Class 2 code | Class M name | Class M code |
| Section 1 | Geo Class 11 | GC11 | Geo Class 12 | GC12 | Geo Class 1M | GC1M |
| Section 2 | Geo Class 21 | GC21 | Geo Class 22 | GC22 |  |  |
| Section 3 | Geo Class 31 | GC31 |  |  | Geo Class 3M | GC3M |
| Section N | Geo Class N1 | GCN1 | Geo Class N2 | GCN2 | Geo Class NM | GCNM |

Files should be processed asynchronously, results should be stored id DB.

* API POST /import (file) returns ID of the Async Job and launches importing.
* API GET /import/{id} returns result of importing by Job ID ("DONE", "IN PROGRESS", "ERROR")
* API GET /export returns ID of the Async Job and launches exporting.
* API GET /export/{id} returns result of parsed file by Job ID ("DONE", "IN PROGRESS", "ERROR")
* API GET /export/{id}/file returns a file by Job ID (throw an exception if exporting is in process)

Requirements:

* Technology stack: Spring, Hibernate, Spring Data, Spring Boot, Gradle/Maven.
* All data (except files) should be in JSON format.
* In export and import use Apache POI for parsing.
* (Optional) Basic Authorization should be supported.