

# COMP7007 Assignment 2

David Barnes ([d.j.barnes@kent.ac.uk](mailto:d.j.barnes@kent.ac.uk))

Date set: 21<sup>st</sup> November 2025

Deadline: 5<sup>th</sup> December 2025 at 12pm (mid-day)

## Background

In the second half of this module, we have covered the following topics:

- Retrieving data from an external source.
- Decoding retrieved data in JSON format.
- Using Java's streams and lambdas.
- Record types.
- JUnit testing

This assignment is designed to allow you to show your understanding and competence with all of those topics.

**If you have questions about any aspect of this assignment, please ask the setter via email.**

## The task

Write a Java program to retrieve from the US Geological Survey data on earthquakes. The program must allow the data to be queried and processed. Further details of the requirements are given below.

## AI policy for this assignment

You are permitted (but not required) to use AI tools to support the development of this program. You may prompt AI tools to help you develop, correct, and explain modest amounts of the program. It would be unacceptable simply to paste the text of this assignment brief into an AI tool to generate the whole thing. However, it is acceptable to ask a tool to write individual classes or methods and to explain and help you fix problems with the code.

Your usage should be from the perspective that you are in control of the development process, and you must adopt a critical approach when reviewing any generated code, identifying and correcting faulty code where necessary.

If you have concerns about whether your intended use of AI is appropriate at any point, please ask the assignment setter for guidance.

## The data

The US Geological Survey ([usgs.gov](https://usgs.gov)) provides worldwide data on earthquakes, accessible programmatically via an API documented [here](#):

<https://earthquake.usgs.gov/fdsnws/event/1/>

The `query` method supports parameters to customise time ranges, locations, earthquake magnitudes, and so on. Documentation and examples of these can be found via the link above.

When accessing the data, please fully respect any usage policy dictated by the USGS and refrain from sourcing very large amounts of data or making many requests for data over a short period. In other words, use the site in a reasonable manner when developing your program.

## The requirements

Your program must:

- Source earthquake data from the `usgs.gov` site in JSON format.
- Include at least one example of a record type to store earthquake-related data.
- Include a text-based interactive front end that allows a user:
  - To customise the queries. For instance, specify date ranges, magnitudes of interest, locations of interest for the data to be retrieved.
  - Allow local processing of the retrieved data.
- Evidence processing of retrieved data using ‘streams and lambdas’, in a similar way to the lecture examples involving the animal-monitoring project and the class examples involving the weather project. At a minimum, this must include use of the `filter` and `map` methods of `Stream`.
- Allow multiple queries and multiple processing of data within a single run of the program.

There is no requirement to store retrieved data on the local file system, although you may include that capability if you wish to.

Retrieval of data and its processing should be considered as two separate steps:

1. Data is retrieved using query parameters provided by the user.
2. The retrieved data is processed, possibly multiple times.

For instance, a user might request **retrieval** of global data for the past 7 days. They might then request local **processing** of the data to print out which locations suffered earthquakes with the highest magnitude. That processing would not require a separate retrieval. They might then request processing of the data to find the average magnitude. That processing would also be performed locally without further retrieval.

## JUnit testing

You must include with your project code a modest sample of JUnit tests to confirm the expected operation of some of the data processing aspects of your program. Note that **you are not required a complete set** of JUnit tests. You just need to demonstrate that you can write effective JUnit tests. Ten tests would be sufficient, for instance. You are not required to write tests that fail (that is, ones that evidence errors in the program).

## A developmental log

You must submit a ‘developmental log’ of the program’s development. This is an informal document whose primary purpose is to illustrate to your marker that you personally developed the program rather than handing the task completely over to an AI tool.

The document is informal, so it does not need to be a polished, grammatically correct, academic report. The idea is that each time you work on the program, you should make brief notes about what you did – preferably as you go along. Think of it as a bit like a diary/journal, so include the current date along with the notes for that date. For instance,

1<sup>st</sup> December

I needed to fix a problem with fetching data for a specific location, so I made a few changes to the Fetcher class, but that didn’t make any difference. I asked Copilot to explain what was wrong, and its answer was: “...” So, I tried that and it fixed it ok. What I was doing wrong was ... Now I can work on the code to analyse average time between quakes at a given location. [And so on.]

The log must be ‘contemporaneous’. In other words, written during the development process and not written after the program is complete. You may write the log by hand, if you prefer, but you will then need to scan or image the pages for upload.

## Javadoc

You must fully Javadoc comment the public features of the program at class and method level. In other words, write class and method comments. Commenting within the bodies of constructors and methods is not required.

## Mark allocation

The following is an indication of how marks will be allocated to the separate elements of the assessment:

- Usage of record types: 10%
- Usage of streams and lambdas: 10%
- Retrieval and decoding of data: 30%
- Interactive functionality: 20%
- JUnit testing: 20%
- Development log: 10%

## What to submit

- Your complete Java project as a zip file. This must include the JUnit tests.
- Your development log. No specific format is required.
- A README file describing how to use the interactive front end. This may be in either text (.txt) or Markdown (.md) format.

Note that failure to provide a credible development log and/or indications of inappropriate AI usage might result in an allegation of academic misconduct. Guidance on academic integrity and misconduct can be found [here](#):

<https://www.kent.ac.uk/education/documents/student-conduct-experience/academic-integrity-and-misconduct.docx>

Version: 2025.11.21.01