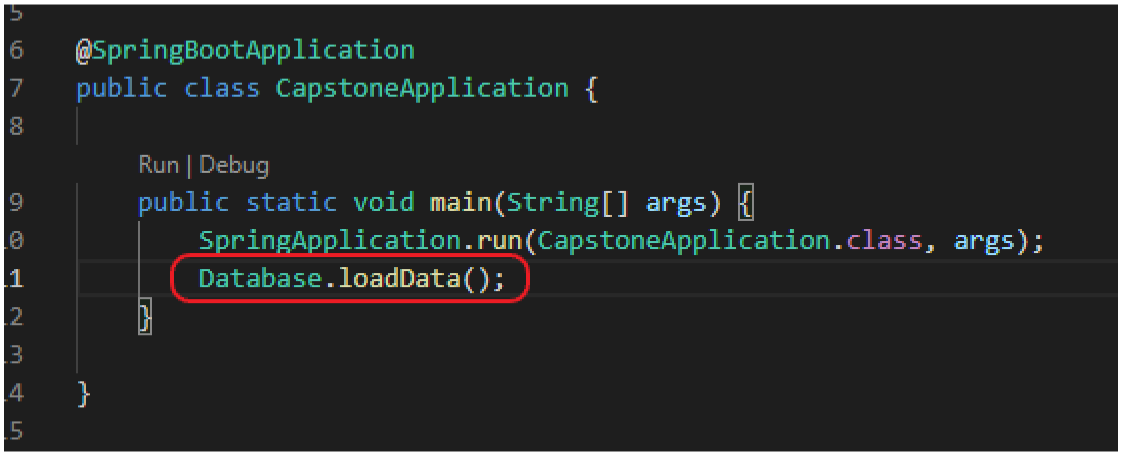
Capstone Project #2
Java/Spring Boot

This capstone project will provide an opportunity to design and build a multi-faceted solution that combines multiple technologies covered to date. Intended to be more substantial than a lab, the capstone aims to simulate the type of work more closely that you might encounter as part of a development team working on larger software engineering projects. The capstone project is considered "open book", self-paced, and to be completed on your own time. You may refer to the relevant videos and reference material (as well as use internet searches) as part of unit completion. Additionally, learners are encouraged to work as teams to complete. Having said that, each learner is still expected to submit an individual solution for the project to ensure that all team members are gaining necessary competency in the target technologies. Prior to beginning development, each learner is required to submit a "design description" (either 1 – 2 paragraphs or a few bullet points) outlining planned approach for build of project solution. Solution code will be provided for review after project completion/submission.

You will be using Java and Spring Boot to build a REST API according to the following requirements:

- In the GitHub repo located at https://github.com/KernelGamut32/capstone2-resources, you will find a Database.java source file
- Create a new Spring Boot application including the Spring Web starter
- Create a new file in your source directory called Database.java, and copy the "raw" contents of the Database.java file in the repository to your new project file
- Using the structure defined in that file (which includes sample data), create Java classes to match with the structure and hierarchy
- In your main method, add "Database.loadData();" to load the sample data into memory on application startup:



```
@SpringBootApplication
public class CapstoneApplication {

    Run | Debug
    public static void main(String[] args) {
        SpringApplication.run(CapstoneApplication.class, args);
        Database.loadData();
    }

}
```

- Create a RestController which includes the following routes:
    - HTTP GET against "/policies" which returns all policy data as JSON
    - HTTP GET against "/policies/<policy #>" which returns a specific policy as JSON
    - HTTP GET against "/policies/claims/<claim #>" which returns a specific claim as JSON

- OPTIONAL CHALLENGE #1: HTTP GET against "/policies/claims/paid" which returns all policies with at least 1 paid claim
- OPTIONAL CHALLENGE #2: HTTP GET against "/policies/<policy #>/totals" which returns annual premium for the policy and a total amount for all claims (paid or unpaid) for the policy as JSON:

  { "policyNumber": "XXXXXX", "annualPremium": #.##, "claimsTotal": #.## }

- Sample JSON has also been provided in https://github.com/KernelGamut32/capstone2-resources for review and verification