I have used the MVC (Model-View-Controller) design pattern for my assessment.

Here is a breakdown of the classes I have implemented and how they align with the MVC pattern:

RecordsController.java: This class handles the RESTful API endpoints for managing records. It acts as the controller, receiving incoming requests, and coordinating the interaction between the model and the view.

FileReaderService.java: This class provides a method to read records from a file. It can be considered part of the model layer, as it handles the logic for retrieving data and updating the database accordingly.

Record.java: This class represents a record model with its attributes and getters/setters. It serves as the model, encapsulating the data and business logic related to records.

RecordService.java: This class contains the business logic for managing records. It can be seen as part of the model layer, as it deals with the processing and manipulation of record data.

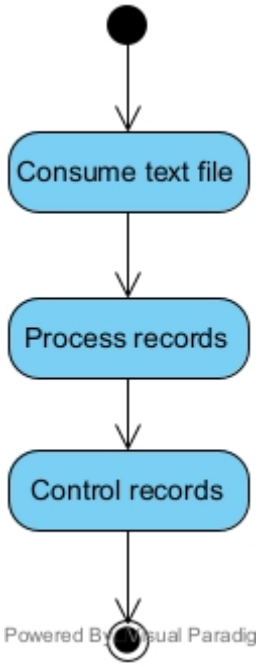
SecurityConfig.java: This class configures security settings for the application. It is not directly related to the MVC pattern but serves as an additional component for managing authentication, which can be considered a cross-cutting concern.

By adopting the MVC pattern, I can achieve a clear separation of concerns. The model handles the data and business logic, the view (in this case, the RESTful API) presents the data to the client, and the controller handles the request handling and coordination between the model and the view.

The MVC pattern is a popular choice for web application development as it promotes modularity, maintainability, and testability. It allows for easier code organization, separation of concerns, and potential reusability of components.

For the documentation requirement, I will provide a class diagram and an activity diagram. The class diagram will illustrate the structure and relationships between the classes in my application, while the activity diagram will demonstrate the flow of activities or actions in a specific scenario.

Activity Diagram



Class Diagram