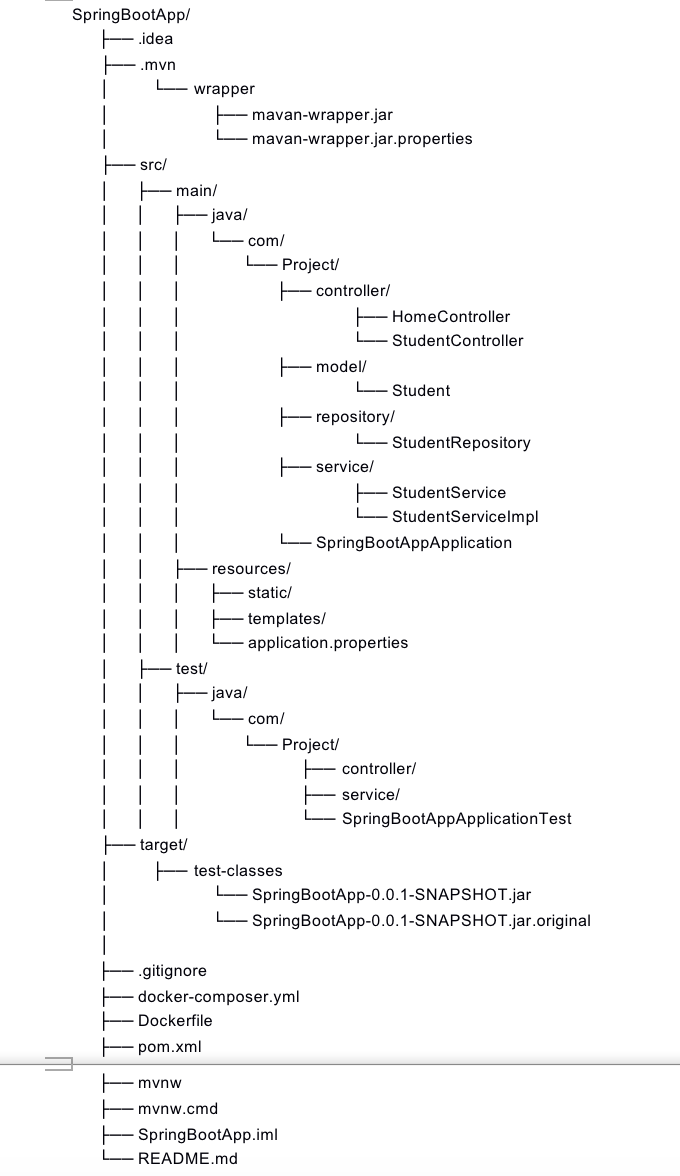
**6. Spring Boot - 2**

1. Checkout to the branch “spring-boot-app-v1” you created in spring boot - 1
2. Create a new branch “spring-boot-app-v2”
3. Propose a proper directory structure for your project created in spring boot part 1



1. Use API naming conventions to your endpoints
2. Use separate request and response types
3. Use exception handling where applicable
4. Return proper status codes
5. Use optionals where possible
6. Why do we need logging in our applications?

**Debugging and Troubleshooting:** Logs provide valuable insights into the inner workings of an application. When unexpected behavior or errors occur, developers can examine the logs to understand the sequence of events, variable values, and potential causes of the issue.

**Issue Identification and Resolution:** By analyzing logs, developers can pinpoint the root causes of bugs, glitches, or crashes, leading to quicker problem resolution and more efficient debugging.

**Performance Monitoring:** Logs offer a window into the performance of an application. They allow developers to monitor response times, resource utilization, and other metrics to identify performance bottlenecks and areas for optimization.

**Security and Auditing:** Logs play a role in security by documenting user activities and system events. In the event of security breaches or unauthorized access, logs can help trace the source of the breach and the actions taken by the attacker.

**Continuous Improvement:** Analyzing logs over time can reveal patterns and trends, helping developers make informed decisions about code improvements, feature enhancements, and overall application evolution.

**Documentation and Communication:** Logs serve as a historical record of an application's behavior, decisions, and changes. They facilitate communication between developers, operations teams, and stakeholders by providing a shared understanding of the application's state and events.

**Compliance and Legal Requirements:** Many industries and applications are subject to regulations that mandate the retention of logs for auditing and compliance purposes. Logs help demonstrate that an application is functioning correctly and in accordance with regulatory standards.

**Quality Assurance:** Logs can assist in quality assurance processes by providing evidence of correct functionality and validating that features work as expected.

**Remote Diagnostics:** When applications are deployed in remote environments, logs become a critical tool for diagnosing issues without direct access to the system.

**User Support and Customer Experience:** Logs aid in troubleshooting user-reported issues. When users encounter problems, support teams can analyze logs to understand the context and provide more accurate solutions.

1. What are different types of log levels?

**TRACE:** The most detailed level, providing fine-grained information about the application's internal workings. Typically used for diagnosing complex issues.

**DEBUG:** Used for debugging purposes. These messages contain information that helps developers track the flow of the program and identify issues.

**INFO:** Provides general information about the application's progress and important milestones. Useful for monitoring and getting a high-level overview.

**WARN:** Indicates potentially harmful situations that don't necessarily cause errors but might lead to issues if not addressed.

**ERROR:** Represents errors or exceptional conditions that need attention. These messages indicate that something has gone wrong, but the application can continue running.

**FATAL/CRITICAL:** Reserved for severe errors that could potentially crash the application or cause critical failures. These issues often require immediate attention.

1. Use a logging framework and add logs to your application
2. Push your updated project to “spring-boot-app-v2”
3. Add your codes and answer sheet to a directory named “spring-boot-basic-training-v2” and push it to your training github repository
4. Create a pull request to main branch and assign it to your trainer

**Resources**

<https://nordicapis.com/10-best-practices-for-naming-api-endpoints/>

<https://restfulapi.net/resource-naming/>

<https://www.geeksforgeeks.org/spring-boot-code-structure/>