SASVA Project Scan Report

Git URL: https://github.com/rahulshirsat30/task-manager.git | Branch: main

Specification:

Project-description: The Task Manager application is a backend service built using the Spring Boot framework, aimed at providing functionality for managing user-related operations. It allows users to create, read, update, and delete user profiles, facilitating the management of user data within an organization or system. The application serves as a RESTful API, enabling interaction with user data through standard HTTP methods. Its architecture leverages the Model-View-Controller (MVC) pattern, promoting separation of concerns and enhancing maintainability. The application is designed to be lightweight, efficient, and easy to integrate with other services or front-end applications. With a focus on simplicity and usability, it aims to provide a robust foundation for user management tasks. The use of Lombok annotations reduces boilerplate code, improving developer productivity. Overall, the Task Manager application is an essential tool for managing user data effectively and efficiently.

Features:

- **Feature:** User Management
  Feature-details: The application allows for the creation, retrieval, updating, and deletion of user profiles, enabling comprehensive user management.
- **Feature:** RESTful API
  Feature-details: The application exposes a RESTful API that adheres to standard HTTP methods, allowing clients to perform operations on user data.
- **Feature:** Dependency Injection
  Feature-details: Utilizes Spring's dependency injection to manage service components, promoting loose coupling and enhancing testability.
- **Feature:** Data Validation
  Feature-details: Ensures that user data is valid and conforms to expected formats before processing any operations.
- **Feature:** Lombok Integration
  Feature-details: Uses Lombok to minimize boilerplate code for model classes, simplifying the codebase and improving readability.

Technical-specification:

- **Platform/Technologies:** Spring Boot, Java, Maven
  Programming Languages: Java
  API Endpoints: GET /users, GET /users/{id}, POST /users, PUT /users/{id}, DELETE /users/{id}
  State Management: Managed through Spring services, with user data stored in memory for simplicity.
  Data Flow: Data is processed through controllers that interact with services, which in turn manage the user data.
  Data Persistence: Currently, user data is stored in memory using a list; persistent storage solutions can be integrated in the future.
  Integration: The application can be integrated with front-end applications via RESTful API calls.
  Validation: Basic validation is performed within the service methods to ensure data integrity.

Routing: Handled by Spring's @RequestMapping annotations in the controllers.
Security: No authentication or authorization is implemented; this can be added for production use.
Third-party components: Lombok for reducing boilerplate code.

File-groups:

- **Filegroup name:** Backend
  Summary: Contains the core backend components of the application, including the main application class, controllers, services, and models.
  Files: pom.xml, TaskmanagerApplication.java, UserController.java, User.java, UserService.java, application.properties

Folder-analysis:

- **Folder name:** src/main/java/org/product/taskmanager
  Summary: Contains the main Java source files for the Task Manager application.
  Purpose: This folder is the core of the application, housing all the Java classes that implement the business logic, controllers, and models necessary for the application to function. It follows the package structure that organizes the code into manageable components.
  Files: TaskmanagerApplication.java, UserController.java, User.java, UserService.java
- **Folder name:** src/main/resources
  Summary: Contains configuration files and resources for the application.
  Purpose: This folder holds configuration files, such as application.properties, which define properties that the Spring Boot application uses at runtime. It is essential for configuring the application context and other settings.
  Files: application.properties

File Descriptions:

Files:

- **Name:** .mvn/wrapper/maven-wrapper.properties
  Details:
  Filename: .mvn/wrapper/maven-wrapper.properties
  Description: This file configures the Maven Wrapper, specifying the version of Maven to use and the distribution URL for downloading it.
  Detailedsummary: The maven-wrapper.properties file is part of the Maven Wrapper setup, which allows developers to run Maven builds without requiring a local installation of Maven. This file specifies the version of the Maven Wrapper to use (3.3.2) and the distribution URL from which Maven (version 3.9.9) can be downloaded. The 'distributionType' is set to 'only-script', indicating that only the wrapper scripts will be included. This setup ensures that all developers working on the project use the same version of Maven, which helps maintain consistency across different environments.
  Importance: High
  References:

- **Name:** .gitattributes
  Details:
  Filename: .gitattributes
  Description: This file configures Git's handling of line endings for specific files in the repository.
  Detailedsummary: The .gitattributes file is used to define attributes for paths in the repository. In this case, it specifies that the file '/mvnw' should use LF (line feed) for end-of-line normalization, while files with the '.cmd' extension should use CRLF (carriage return and line feed). This ensures consistent line endings across different operating systems, which is crucial for collaboration in a multi-platform environment. Proper line ending management helps prevent issues related to version control and script execution, especially when working with shell scripts and batch files. This file plays a vital role in maintaining code quality and compatibility across various development environments.
  Importance: High
  References:
- **Name:** mvnw
  Details:
  Filename: mvnw
  Description: This file is a shell script that serves as the Apache Maven Wrapper, allowing users to run Maven commands without requiring a local Maven installation.
  Detailedsummary: The 'mvnw' file is a shell script that automates the process of downloading and executing Apache Maven. It checks for the JAVA_HOME environment variable, validates the Maven distribution URL, and handles downloading and installing Maven if it is not already present. The script also supports checksum validation for security, ensuring that the downloaded Maven distribution is not compromised. It adapts to different operating systems, including Windows and Unix-like systems, and provides verbose logging options for debugging. The script ultimately executes Maven commands using the installed version, streamlining the build process for Java projects.
  Importance: High
  References:
    - .mvn/wrapper/maven-wrapper.properties
- **Name:** mvnw.cmd
  Details:
  Filename: mvnw.cmd
  Description: This file is a batch script for the Apache Maven Wrapper, facilitating the download and setup of Maven in a project environment.
  Detailedsummary: The mvnw.cmd file is a batch script that serves as a wrapper for Apache Maven, allowing users to run Maven commands without requiring a pre-installed Maven. It checks for the Maven distribution URL, downloads the necessary files, validates checksums, and sets up the environment for Maven execution. The script also handles optional environment variables for repository URLs and credentials, ensuring a smooth setup process. It includes error handling for various stages, such as downloading and moving the Maven installation, and provides verbose logging options for debugging. Overall, it simplifies the Maven setup for developers by automating the installation process.
  Importance: High
  References:

        - .mvn/wrapper/maven-wrapper.properties
- **Name:** pom.xml
  Details:
    Filename: pom.xml
    Description: This file is a Maven Project Object Model (POM) configuration for a Spring Boot application, defining project metadata, dependencies, and build settings.
    Detailedsummary: The pom.xml file is essential for managing a Spring Boot project using Maven. It specifies the project coordinates such as groupId, artifactId, and version. It inherits from the Spring Boot starter parent, ensuring compatibility with Spring Boot features. The file includes dependencies for Spring Boot, Lombok for reducing boilerplate code, and Spring Boot testing utilities. The build section configures the Spring Boot Maven plugin, which simplifies the packaging and running of the application. Overall, this file orchestrates the project's structure, dependencies, and build process, making it crucial for development and deployment.
    Importance: High
    References:
- **Name:** src/main/java/org/product/taskmanager/TaskmanagerApplication.java
  Details:
    Filename: src/main/java/org/product/taskmanager/TaskmanagerApplication.java
    Description: This file serves as the entry point for the Task Manager application, utilizing Spring Boot to bootstrap the application context.
    Detailedsummary: The TaskmanagerApplication.java file is the main class of the Task Manager application, which is built using the Spring Boot framework. It is annotated with @SpringBootApplication, indicating that it is a Spring Boot application. The main method within this class calls SpringApplication.run(), which launches the application. This method sets up the application context, loads the necessary configurations, and starts the embedded web server if applicable. This file is crucial for initializing the application and serves as the starting point for the execution of the application.
    Importance: High
    References:
- **Name:** src/main/java/org/product/taskmanager/model/User.java
  Details:
    Filename: src/main/java/org/product/taskmanager/model/User.java
    Description: This file defines a User class that represents a user entity with attributes such as id, name, and email, utilizing Lombok annotations for boilerplate code reduction.
    Detailedsummary: The User.java file is part of the model layer in a Java application, specifically designed to represent a user entity. It includes three private fields: id (Long), name (String), and email (String). The class is annotated with Lombok annotations such as @Data, which generates getters, setters, and other utility methods automatically, reducing boilerplate code. The @NoArgsConstructor and @AllArgsConstructor annotations provide constructors for creating User objects with and without parameters. This class is essential for managing user data within the application, facilitating operations such as user creation, retrieval, and updates.
    Importance: High
    References:
- **Name:** src/main/java/org/product/taskmanager/controller/UserController.java
  Details:

Filename: src/main/java/org/product/taskmanager/controller/UserController.java
Description: This file defines the UserController class, which handles HTTP requests related to user management, including retrieving, creating, updating, and deleting users.
Detailedsummary: The UserController class is a Spring REST controller that manages user-related operations. It is annotated with @RestController and @RequestMapping to define the base URL for user-related endpoints. The class uses the UserService to perform operations such as fetching all users, retrieving a user by ID, creating a new user, updating an existing user, and deleting a user. Each method is mapped to specific HTTP methods (GET, POST, PUT, DELETE) and handles requests accordingly. The use of @Autowired allows for dependency injection of the UserService, ensuring that the controller can access the necessary business logic for user management.
Importance: High
References:

  - src/main/java/org/product/taskmanager/model/User.java
  - src/main/java/org/product/taskmanager/service/UserService.java

- **Name:** src/main/resources/application.properties
  Details:
  Filename: src/main/resources/application.properties
  Description: This file defines configuration properties for a Spring application, specifically setting the application name to 'taskmanager'.
  Detailedsummary: The 'application.properties' file is a key configuration file in a Spring application. It allows developers to define various properties that the application will use at runtime. In this specific file, the property 'spring.application.name' is set to 'taskmanager', which identifies the application within the Spring ecosystem. This name can be used for logging, monitoring, and other management tasks. The properties defined in this file can influence the behavior of the application, such as its connection to databases, message brokers, and other services. Proper configuration is essential for the application to function correctly and efficiently.
  Importance: High
  References:
- **Name:** src/main/java/org/product/taskmanager/service/UserService.java
  Details:
  Filename: src/main/java/org/product/taskmanager/service/UserService.java
  Description: This file defines the UserService class, which provides methods for managing user data, including creating, retrieving, updating, and deleting users.
  Detailedsummary: The UserService class is annotated with @Service, indicating that it is a Spring service component. It maintains a list of User objects and provides several methods: getAllUsers() returns all users, getUserById(Long id) retrieves a user by their ID, createUser(User user) adds a new user, updateUser(Long id, User updatedUser) updates an existing user's details, and deleteUser(Long id) removes a user from the list. The class uses an idCounter to assign unique IDs to new users, ensuring that each user has a distinct identifier.
  Importance: High
  References:
    - src/main/java/org/product/taskmanager/model/User.java
- **Name:** src/test/java/org/product/taskmanager/TaskmanagerApplicationTests.java
  Details:

Filename: src/test/java/org/product/taskmanager/TaskmanagerApplicationTests.java
Description: This file contains a test class for the Taskmanager application, ensuring that the application context loads correctly during testing.
Detailedsummary: The TaskmanagerApplicationTests class is a test suite for the Taskmanager application, utilizing the Spring Boot testing framework. It is annotated with @SpringBootTest, which indicates that it will load the application context for testing. The contextLoads() method is a simple test that checks if the application context can start without any issues. This is a fundamental test to ensure that the application is correctly configured and that all necessary components are available. It serves as a baseline test to prevent issues during further development and testing phases.
Importance: High
References:

- org/junit/jupiter/api/Test.java
- org/springframework/boot/test/context/SpringBootTest.java

Project Details:

Key	Value
Id	927016539
Node id	R_kgDON0EmWw
Name	task-manager
Full name	rahulshirsat30/task-manager
Private	false

<table>
<tr><th>Key</th><th>Value</th></tr>
<tr><td><b>Owner</b></td><td>
<table>
<tr><th>Key</th><th>Value</th></tr>
<tr><td><b>Login</b></td><td>rahulshirsat30</td></tr>
<tr><td><b>Id</b></td><td>197733786</td></tr>
<tr><td><b>Node id</b></td><td>U_kgDOC8ktmg</td></tr>
<tr><td><b>Avatar url</b></td><td>https://avatars.githubusercontent.com/u/197733786?v=4</td></tr>
<tr><td><b>Gravatar id</b></td><td></td></tr>
<tr><td><b>Url</b></td><td>https://api.github.com/users/rahulshirsat30</td></tr>
<tr><td><b>Html url</b></td><td>https://github.com/rahulshirsat30</td></tr>
<tr><td><b>Followers url</b></td><td>https://api.github.com/users/rahulshirsat30/followers</td></tr>
<tr><td><b>Following url</b></td><td>https://api.github.com/users/rahulshirsat30/following{/other_user}</td></tr>
<tr><td><b>Gists url</b></td><td>https://api.github.com/users/rahulshirsat30/gists{/gist_id}</td></tr>
<tr><td><b>Starred url</b></td><td>https://api.github.com/users/rahulshirsat30/starred{/owner}{/repo}</td></tr>
<tr><td><b>Subscriptions url</b></td><td>https://api.github.com/users/rahulshirsat30/subscriptions</td></tr>
<tr><td><b>Organizations url</b></td><td>https://api.github.com/users/rahulshirsat30/orgs</td></tr>
<tr><td><b>Repos url</b></td><td>https://api.github.com/users/rahulshirsat30/repos</td></tr>
<tr><td><b>Events url</b></td><td>https://api.github.com/users/rahulshirsat30/events{/privacy}</td></tr>
<tr><td><b>Received events url</b></td><td>https://api.github.com/users/rahulshirsat30/received_events</td></tr>
<tr><td><b>Type</b></td><td>User</td></tr>
<tr><td><b>User view type</b></td><td>public</td></tr>
<tr><td><b>Site admin</b></td><td>false</td></tr>
</table>
</td></tr>
<tr><td><b>Html url</b></td><td>https://github.com/rahulshirsat30/task-manager</td></tr>
<tr><td><b>Description</b></td><td>Task manager to manage the task</td></tr>
<tr><td><b>Fork</b></td><td>false</td></tr>
<tr><td><b>Url</b></td><td>https://api.github.com/repos/rahulshirsat30/task-manager</td></tr>
<tr><td><b>Forks url</b></td><td>https://api.github.com/repos/rahulshirsat30/task-manager/forks</td></tr>
<tr><td><b>Keys url</b></td><td>https://api.github.com/repos/rahulshirsat30/task-manager/keys{/key_id}</td></tr>
<tr><td><b>Collaborators url</b></td><td>https://api.github.com/repos/rahulshirsat30/task-manager/collaborators{/collaborator}</td></tr>
<tr><td><b>Teams url</b></td><td>https://api.github.com/repos/rahulshirsat30/task-manager/teams</td></tr>
<tr><td><b>Hooks url</b></td><td>https://api.github.com/repos/rahulshirsat30/task-manager/hooks</td></tr>
</table>

Key	Value
Issue events url	https://api.github.com/repos/rahulshirsat30/task-manager/issues/events{/number}
Events url	https://api.github.com/repos/rahulshirsat30/task-manager/events
Assignees url	https://api.github.com/repos/rahulshirsat30/task-manager/assignees{/user}
Branches url	https://api.github.com/repos/rahulshirsat30/task-manager/branches{/branch}
Tags url	https://api.github.com/repos/rahulshirsat30/task-manager/tags
Blobs url	https://api.github.com/repos/rahulshirsat30/task-manager/git/blobs{/sha}
Git tags url	https://api.github.com/repos/rahulshirsat30/task-manager/git/tags{/sha}
Git refs url	https://api.github.com/repos/rahulshirsat30/task-manager/git/refs{/sha}
Trees url	https://api.github.com/repos/rahulshirsat30/task-manager/git/trees{/sha}
Statuses url	https://api.github.com/repos/rahulshirsat30/task-manager/statuses/{sha}
Languages url	https://api.github.com/repos/rahulshirsat30/task-manager/languages
Stargazers url	https://api.github.com/repos/rahulshirsat30/task-manager/stargazers
Contributors url	https://api.github.com/repos/rahulshirsat30/task-manager/contributors
Subscribers url	https://api.github.com/repos/rahulshirsat30/task-manager/subscribers
Subscription url	https://api.github.com/repos/rahulshirsat30/task-manager/subscription
Commits url	https://api.github.com/repos/rahulshirsat30/task-manager/commits{/sha}
Git commits url	https://api.github.com/repos/rahulshirsat30/task-manager/git/commits{/sha}
Comments url	https://api.github.com/repos/rahulshirsat30/task-manager/comments{/number}
Issue comment url	https://api.github.com/repos/rahulshirsat30/task-manager/issues/comments{/number}
Contents url	https://api.github.com/repos/rahulshirsat30/task-manager/contents/{+path}
Compare url	https://api.github.com/repos/rahulshirsat30/task-manager/compare/{base}...{head}
Merges url	https://api.github.com/repos/rahulshirsat30/task-manager/merges
Archive url	https://api.github.com/repos/rahulshirsat30/task-manager/{archive_format}{/ref}
Downloads url	https://api.github.com/repos/rahulshirsat30/task-manager/downloads
Issues url	https://api.github.com/repos/rahulshirsat30/task-manager/issues{/number}
Pulls url	https://api.github.com/repos/rahulshirsat30/task-manager/pulls{/number}
Milestones url	https://api.github.com/repos/rahulshirsat30/task-manager/milestones{/number}
Notifications url	https://api.github.com/repos/rahulshirsat30/task-manager/notifications{?since,all,participating}
Labels url	https://api.github.com/repos/rahulshirsat30/task-manager/labels{/name}
Releases url	https://api.github.com/repos/rahulshirsat30/task-manager/releases{/id}

Key	Value
Deployments url	https://api.github.com/repos/rahulshirsat30/task-manager/deployments
Created at	2025-02-04T09:04:37Z
Updated at	2025-02-04T14:22:27Z
Pushed at	2025-02-04T14:22:23Z
Git url	git://github.com/rahulshirsat30/task-manager.git
Ssh url	git@github.com:rahulshirsat30/task-manager.git
Clone url	https://github.com/rahulshirsat30/task-manager.git
Svn url	https://github.com/rahulshirsat30/task-manager
Homepage	null
Size	0
Stargazers count	0
Watchers count	0
Language	Java
Has issues	true
Has projects	true
Has downloads	true
Has wiki	true
Has pages	false
Has discussions	false
Forks count	0
Mirror url	null
Archived	false
Disabled	false
Open issues count	0
License	null
Allow forking	true
Is template	false
Web commit signoff required	false
Topics	null

<table>
<tr><th>Key</th><th>Value</th></tr>
<tr><td><b>Visibility</b></td><td>public</td></tr>
<tr><td><b>Forks</b></td><td>0</td></tr>
<tr><td><b>Open issues</b></td><td>0</td></tr>
<tr><td><b>Watchers</b></td><td>0</td></tr>
<tr><td><b>Default branch</b></td><td>main</td></tr>
<tr><td><b>Permissions</b></td><td><table>
<tr><th>Key</th><th>Value</th></tr>
<tr><td><b>Admin</b></td><td>true</td></tr>
<tr><td><b>Maintain</b></td><td>true</td></tr>
<tr><td><b>Push</b></td><td>true</td></tr>
<tr><td><b>Triage</b></td><td>true</td></tr>
<tr><td><b>Pull</b></td><td>true</td></tr>
</table></td></tr>
<tr><td><b>Temp clone token</b></td><td></td></tr>
<tr><td><b>Allow squash merge</b></td><td>true</td></tr>
<tr><td><b>Allow merge commit</b></td><td>true</td></tr>
<tr><td><b>Allow rebase merge</b></td><td>true</td></tr>
<tr><td><b>Allow auto merge</b></td><td>false</td></tr>
<tr><td><b>Delete branch on merge</b></td><td>false</td></tr>
<tr><td><b>Allow update branch</b></td><td>false</td></tr>
<tr><td><b>Use squash pr title as default</b></td><td>false</td></tr>
<tr><td><b>Squash merge commit message</b></td><td>COMMIT_MESSAGES</td></tr>
<tr><td><b>Squash merge commit title</b></td><td>COMMIT_OR_PR_TITLE</td></tr>
<tr><td><b>Merge commit message</b></td><td>PR_TITLE</td></tr>
<tr><td><b>Merge commit title</b></td><td>MERGE_MESSAGE</td></tr>
</table>

<table>
<tr><th>Key</th><th>Value</th></tr>
<tr><td><b>Security and analysis</b></td><td>
<table>
<tr><th>Key</th><th>Value</th></tr>
<tr><td><b>Secret scanning</b></td><td><table><tr><th>Key</th><th>Value</th></tr><tr><td><b>Status</b></td><td>enabled</td></tr></table></td></tr>
<tr><td><b>Secret scanning push protection</b></td><td><table><tr><th>Key</th><th>Value</th></tr><tr><td><b>Status</b></td><td>enabled</td></tr></table></td></tr>
<tr><td><b>Dependabot security updates</b></td><td><table><tr><th>Key</th><th>Value</th></tr><tr><td><b>Status</b></td><td>disabled</td></tr></table></td></tr>
<tr><td><b>Secret scanning non provider patterns</b></td><td><table><tr><th>Key</th><th>Value</th></tr><tr><td><b>Status</b></td><td>disabled</td></tr></table></td></tr>
<tr><td><b>Secret scanning validity checks</b></td><td><table><tr><th>Key</th><th>Value</th></tr><tr><td><b>Status</b></td><td>disabled</td></tr></table></td></tr>
</table>
</td></tr>
<tr><td><b>Network count</b></td><td>0</td></tr>
<tr><td><b>Subscribers count</b></td><td>1</td></tr>
</table>