

API RATE LIMITER(SUMMARY)

TEAM -4

**MANSI
MADHUMITA GANGA S
KEERTHANA R**

INDEX

- **Approach**
- **System Requirements**
- **Quick fixes (Debug)**
- **Execution Steps**

APPROACH

The implementation of our API Rate Limiter is based on the “Sliding Window” Rate-Limit Algorithm. The basic idea of the algorithm is:

If the number of requests served on **configuration key** in the last **time_window_sec** seconds is more than **number_of_requests** configured for it then discard, else the request goes through while we update the counter.

- The window is fixed to 1 minute.
- Default rate limit can be set at the time of object creation. This will only be used if a specific user is either anonymous or the rate limit for the user is not annotated.
- Custom annotations: Custom annotations are created for programmers to input the user rate limits for each API making it plug-and-play.
- Internally, 2 hashmaps are used to keep track of which API is being requested by which User. Each API+User combination has a separate hashmap.
- The user authentication, based on user limit and requested api, is done using JWT Authentication.

SYSTEM REQUIREMENTS

- Eclipse Enterprise Edition
- Jdk version-13.0.2
- Jar executable file
<http://www.java2s.com/Code/Jar/e/Downloadexecutablejararchetype10jar.htm>
- Apache-tomcat-7.0.105-windows-x64 or any latest version

QUICK FIXES

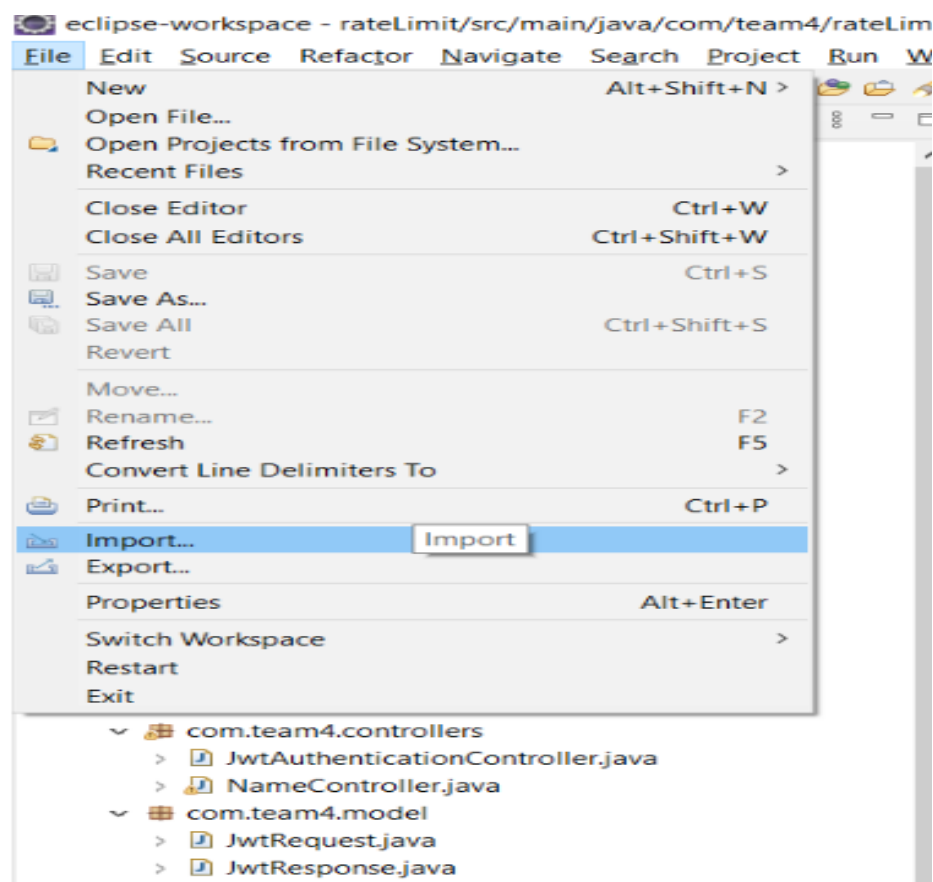
- The server may not be set-up and may throw “javax.servlet.http.HttpServlet” error.
Solution: Download the tomcat-servlet-api-9.0.4.jar (for the tomcat version installed in your system)
- org.springframework error

Solution: Right click on project -> build path -> configure build path -> JRE System Library -> Edit -> Alternate JRE, and change the version.

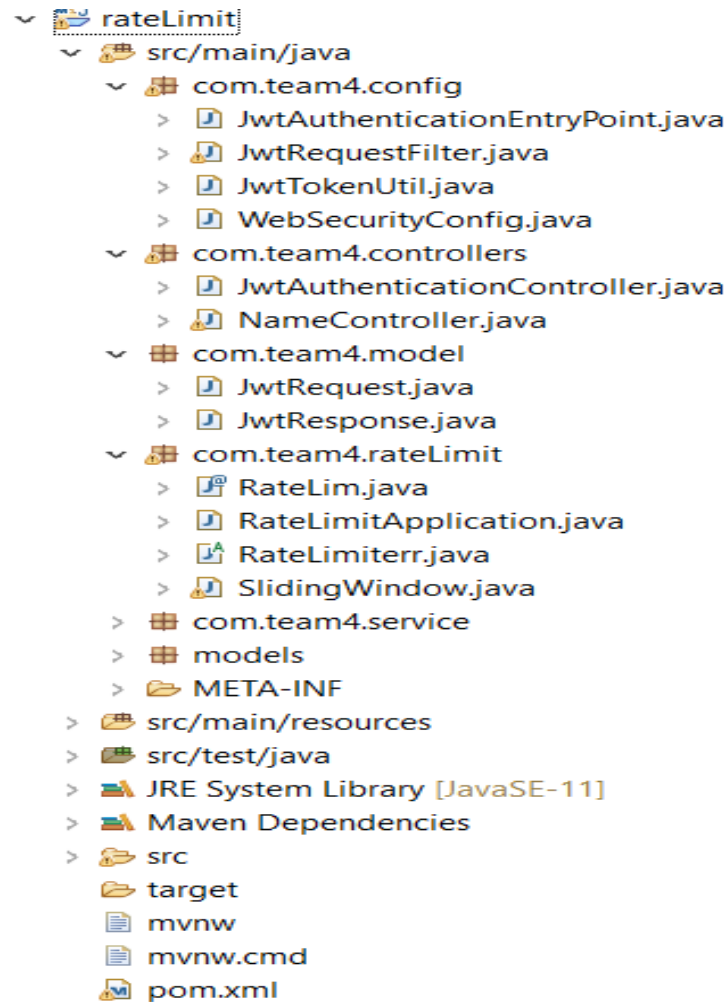
- JUnit Tests not running
Solution: Make sure JUnit4 is in classpath. If not, follow <https://javarevisited.blogspot.com/2012/10/5-ways-to-add-multiple-jar-to-classpath-java.html>
<https://github.com/junit-team/junit4/wiki/Download-and-Install>

EXECUTION STEPS

1. Unzip the folder ratelimit-master and import it in the Eclipse IDE

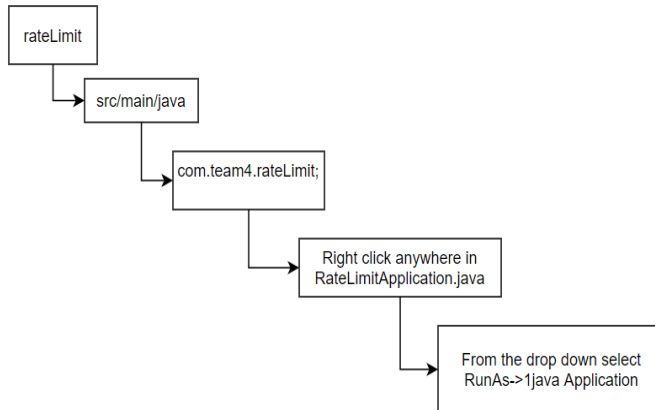


2. Under "Maven" select "Existing Maven" and browse for your unzipped folder.
3. Once the project gets loaded, it will come like this:



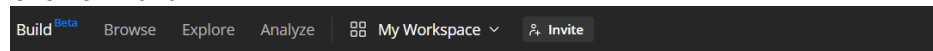
4. The normal Execution flow to run the application is:

Go on navigating through and finally under "RateLimitApplication.java",
RightClick->RunAs->1Java Application




5. Now open the Postman Application and do the following.

Click on Build

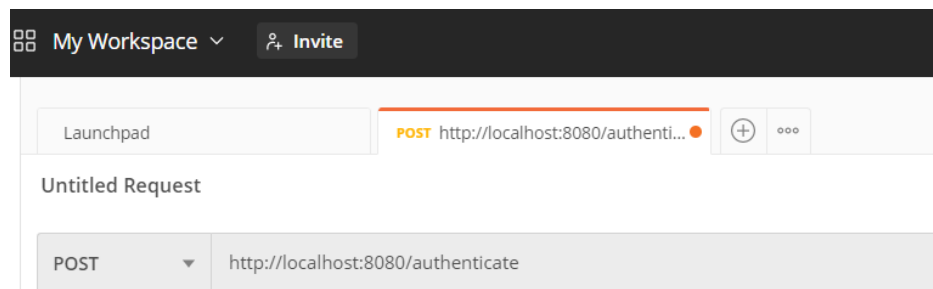


6. Now on the dashboard, click on “Create Request”

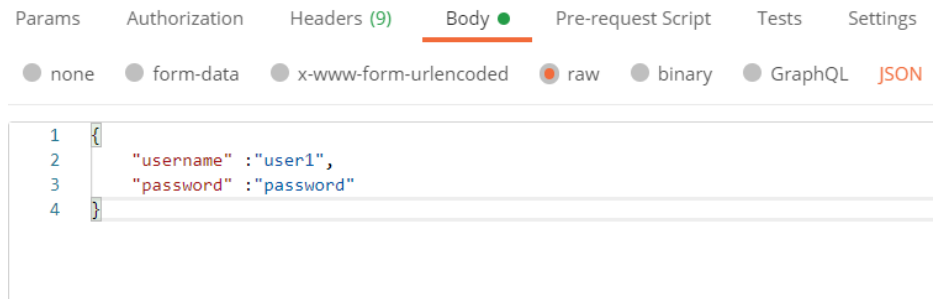
Start something new

 Create a request

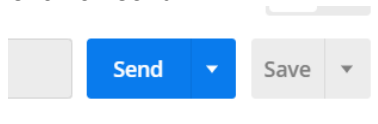
7. Now in the URL field, type the following
(Tip: Make sure u keep it as “POST”)



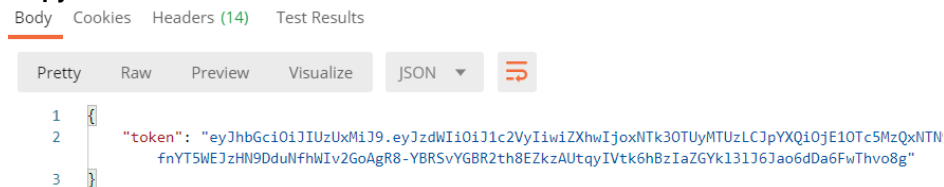
8. Now write the following under “body”
(Tip: Select the “JSON” from the dropdown and check “raw”)



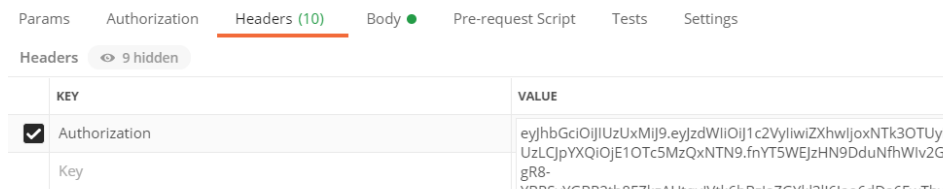
9. Click on send



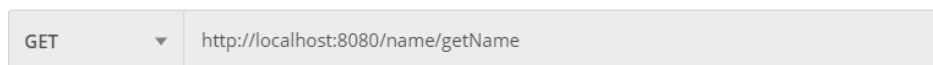
10. Now you get a token (JWT Token) which is used for authenticating the user. Copy this token



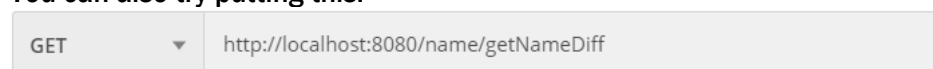
11. Now under the Header section, select key=" Authorization" (It comes from drop down) and paste the token u had copied in step 10 as the value.



12. Now go to URL section and paste as follows (Tip: This time select "GET")



You can also try putting this:



13. Hit send again and you will get the status of your request.

14. To check the api-limiting aspect, keep on hitting "SEND" until you get status:429(Too many requests).