**Approach:**

* A Spring Boot application was configured with a custom AuthenticationController that exposes a REST endpoint /authenticate.
* JWT token generation was handled using a securely generated Base64 key, injected via application.properties.
* Spring Security was configured to allow unauthenticated access to /authenticate, while securing all other endpoints.
* Basic authentication (-u user:pwd) was used to verify credentials.
* On success, a signed JWT token is generated and returned as a JSON response.

**Execution:**

* The application was run using mvn spring-boot:run.
* A curl command or Postman request was made to /authenticate with basic credentials.
* On valid input, a JWT token was returned.

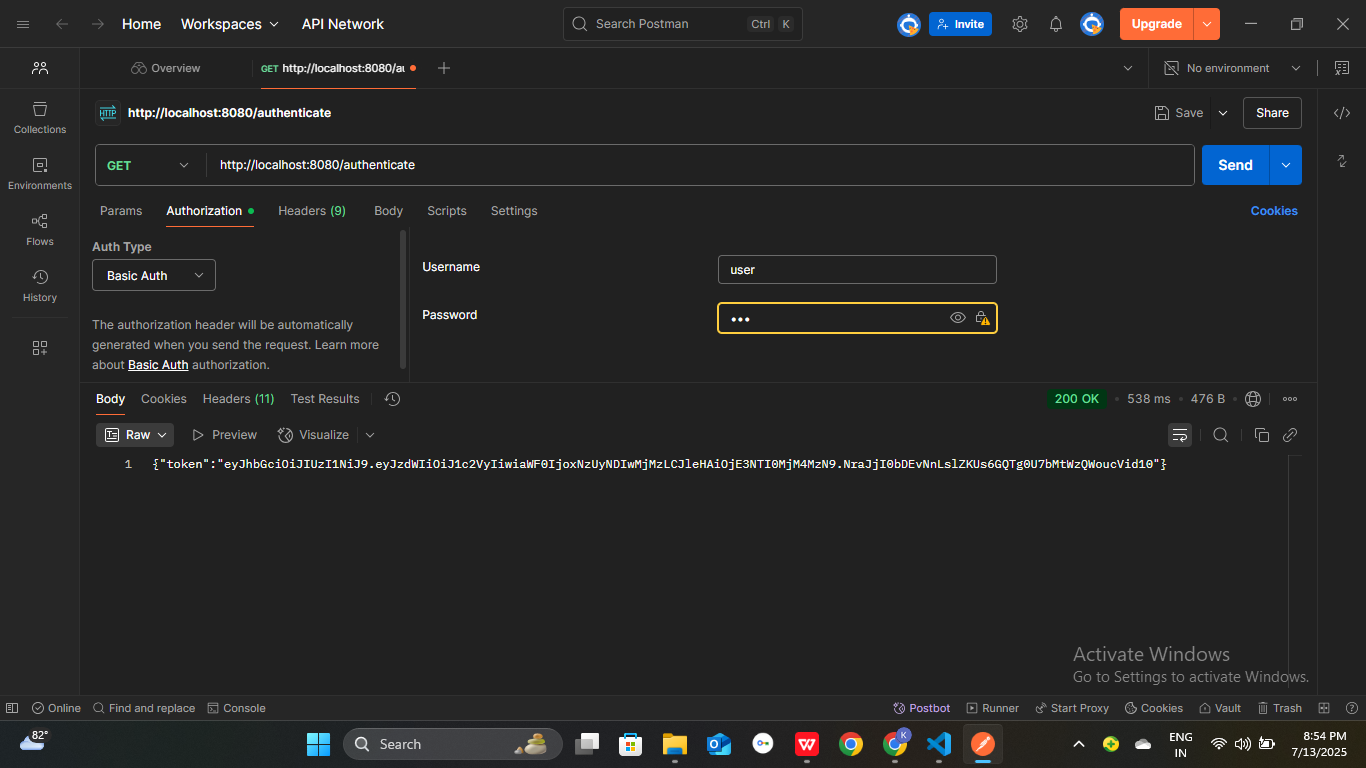
**Result:**

* The response contained a JSON object with a token field.
* This token can now be used in the Authorization: Bearer <token> header to access protected endpoints.
* Logs showed debug information about the controller invocation, JWT creation, and any failures.

**Verification:**

* Token structure matched the standard JWT format (xxxxx.yyyyy.zzzzz).
* HTTP response code 200 OK confirmed successful token generation.
* Header inspection (via browser dev tools or Postman) confirmed content-type as application/json.

**Output:**

****