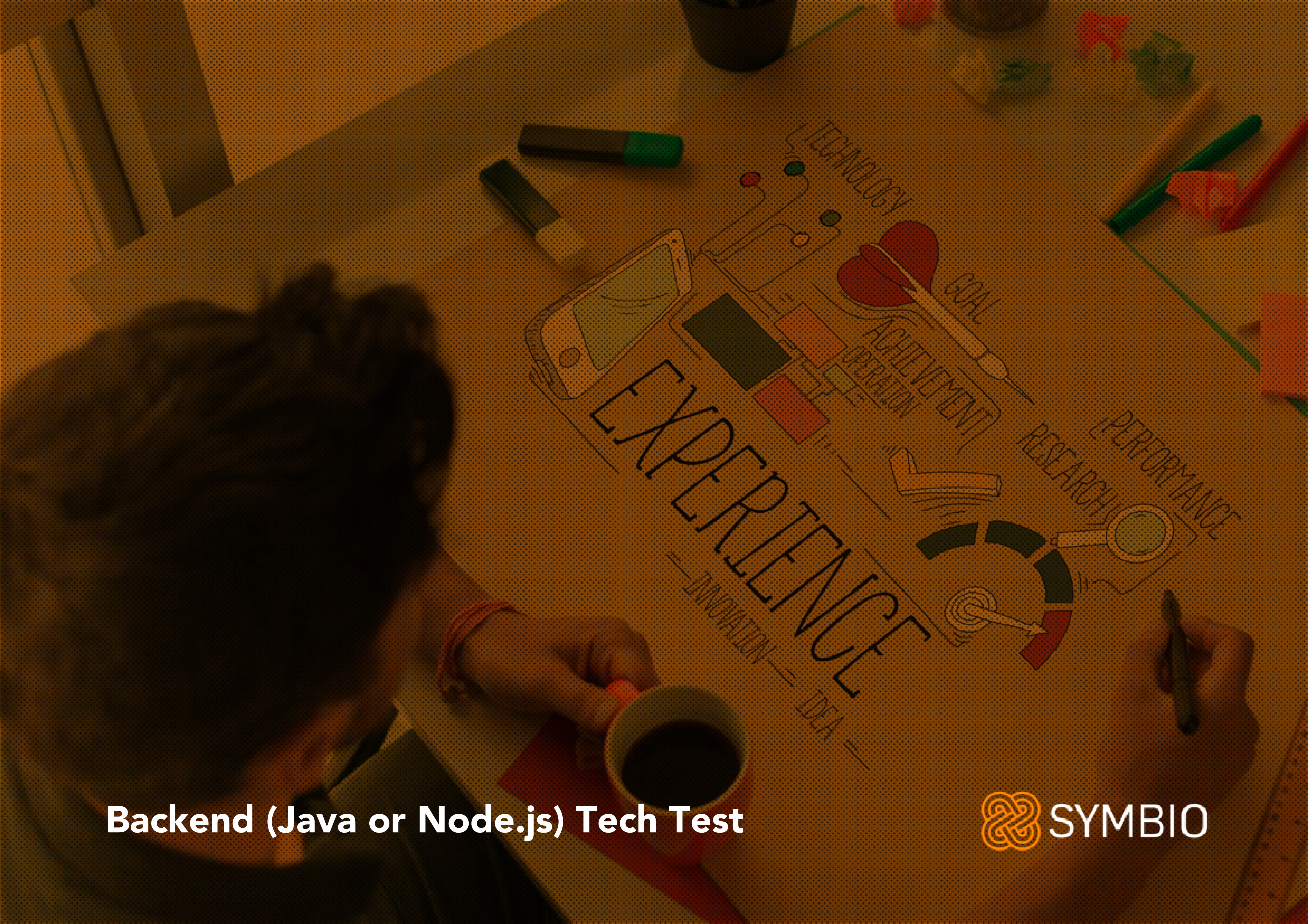

# Backend (Java or Node.js) Tech Test

SYMBIO

# INTRODUCTION

Congrats! If you're receiving this test it means that our Talent Manager liked you! Read carefully the test and if something is not clear enough just let us know. You have 7 calendar days to deliver this assignment. We're flexible so if you're having problems to deliver the test on time, please contact our Talent Manager. You can send the test on a zip file or upload it to GitHub.

*Feel free to comment your code on GitHub if you want.

.

# ASSIGNMENT

1. Design a REST API using your technology of preference (Java or Node.js) to manage a blog.

2. The REST API must have the basic CRUD operations to manage two entities, with at least the following fields:
    a. User: [user ID, username, e-mail, password]
    b. Post: [post ID, user ID, title, body, date]

You can add more fields, if you think they are necessary.

3. Your REST API must be capable to perform the following operations:
    a. Create user.
    b.User login.
    c. Edit user (username and password only).
    d. Get user by ID
    e. Create post (maximum 150 characters).
    f. Edit post.
    g. Delete post.
    h. Get post by ID.
    i. Search for posts using the Title or Body.

4. Anybody with anonymous access to the REST API can create a new user (**Item # 3.a**), login (**Item # 3.b**), get posts by ID (**Item # 3.h**) and search for posts (**Item # 3.i**). All the other actions can be performed **only by authenticated users**.

5. The user data and the posts must be persisted in a SQLite database file.

## BONUS

You don't have to implement the following requirement in the REST API if you don't want to, but they will be considered a bonus:

After you created your REST API, it became extremely popular and it started to be used by millions of users around the world, and you noticed that there are two bottlenecks in your system that are causing an unnecessary number of requests to the database, slowing down everything: when the user gets a post by ID (**Item # 3.h**) and when a search for posts is done (**Item # 3.i**).

To minimize the number of requests to the database, you decide to create a caching system in your API. Every time someone tries to get a post by ID or search by posts using Title or Body, your API must search for this information in the database and cache the results for 5 minutes. If another request with the same parameters is sent to the API within the 5 minutes after the first request, then your API must not query to database to retrieve the necessary information, but use the cache data instead.

**Points to Consider During the API Implementation:**

1. What HTTP verbs will be used for each endpoint?
2. What HTTP codes will be used for each response?
3. How is the payload and the response formatted?
4. Do you need to worry about HTTP headers in your requests?
5. How is the error handling done?
6. How would you document the API for a developer that is planning to use your service?