# Kbox “Java/Object Oriented Principles” Programming Assignment

## Guidelines and the purpose

This is an assignment designed specifically to test how much a candidate knows (and cares) about

- code quality
- object oriented principles
- clean coding best practices
- design patterns and
- written communication skills

The output of this assignment would be the base for evaluation and will be followed by a technical interview if qualified.

The assignment is designed so that it should not take more than 1-2 hours to complete, and 1-2 hours to refactor and polish. (Just as a benchmark: most of our successful hires in the past told us that they had spent no more than 2 hours in one sitting for the assignment.) That being said, it is crucial to present your abilities to understand a problem, design a solution to the problem and communicate your solution to your teammates, all within the scope of this assignment.

Thank you in advance for your interest in Kbox and for the time you have invested in the interview process with us.
We sincerely hope to see you as part of our team.

Umut Alp - Chief Technology Officer
(umut.alp@kboxglobal.com)

# Project Specification:

The solution should consist of a typical Spring Boot maven project listening on a certain port and exposing a handful API endpoints:

- Every API call in the project should have a user as parameter (for the sake of simplicity you can assume a numeric userId)

- Design an API endpoint to store the views of user profiles in a relational database, just like a typical social network application (please see more details in the “Instructions” section below)
  - UserX viewing the profile of UserY should be recorded
  - Date and time of the view should be recorded

- Design an API endpoint to list the users who viewed a particular user’s profile in the past
  - The endpoint should take the viewee’s userId as input
  - The list should include the viewer’s user id and also the date/time of the view
  - The list should NOT include more than 20 items
  - The list should NOT include views older than 30 days

- Design any relevant support code to ensure that the requirements in this specification document are met at all times

- Assuming you have millions of views every hour, try to come up with the most efficient database schema in terms of storage space, latency and throughput. Please briefly justify your design choices and the reasoning behind them, so that we can follow your train of thoughts here

# Instructions:

- Setup your project as a maven based Spring Boot project and make sure it can read from and write to a local file based database. (You can use any SQL compliant library here; H2, Derby, HsqlDb, Sqlite etc.)

- Create the source code to meet the requirements stated in the “Project Specifications” section above

- Create a design document that briefly discusses the chosen database schema and the reasoning behind it. Please make sure these questions are answered and discussed:
    - Do you delete any data from the database?
        - If yes, When? Why?
        - If no, Why?
    - Do you have any periodic task type of batch jobs to maintain data?
        - If yes, Why needed?
        - If no, Why not needed?
    - What type of compromise (for example; tradeoff between “storage on disk” vs “latency in retrieval”) did you see? What was your decision? Why?

- Also include the instructions to run the project and test if it works as expected

Please focus on clean/maintainable code, best possible data structures and corresponding database schemas. Use your good judgment to include the right amount of design patterns and documentation in your code. It should be structured but also not over engineered.