**Technical Interview Code Project**

Tasks:

Create a single sign-on API endpoint for application authentication. The API should receive a Username or Email and a password from any platform (web, mobile, desktop app, etc.). The API should handle all of the authentication process (database look up, password validation, verification).

The API should have a JSON response acknowledging if the authentication request succeeded or failed.

If the request succeeds, the JSON response should include a hash to use for future requests for authentication.

Minimum requirements (feel free to add data, functionality, etc. to any of these):

Database, API, and login functionality should be object oriented and show proper inheritance usage, dependency injection, and other fundamentals.

Use a MySQL database:

* Have at least two tables (Users, Logins):
  + Users columns:
    - Id
    - Username
    - Password
    - FirstName
    - LastName
    - Email
    - ActiveUser (Boolean to track deleted users, ‘1’ means the user has an active account and can log in)
  + Logins columns:
    - Id
    - LoginDateTime
    - Successful (boolean to track unsuccessful login attempts)
    - UserId
    - AuthenticationHash
    - IpAddress
    - Expires (date, can set a random expire time for user to reauthenticate)

Have a web-based login form that utilizes the API to authenticate a user and proves if an attempt was successful/unsuccessful.

Extra Credit:

* Lock account out on 5 failed login attempts and send an email to the user to unlock their account/reset their password
* Log suspicious log in attempts to the database in another table (based on IpAddress and time between login attempts). Example: if a user account tries to authenticate 3 times in a short amount of time, all from different IP addresses, it could mean someone is trying to brute force their account. Log the suspicion to the database.
* Have correct Primary, Foreign, and Unique keys set up on database tables
* Have API set up on a separate domain and the login form send a request to the API’s domain/endpoint
* Encrypt/encode API request data (base64 encode the request before sending to API, and the API base64 decodes the request before processing – this helps avoid encoding errors with quotes and other special characters in passwords)
* Provide a unit test for the code