Technical Test

Complete the following three questions and provide answers in a zip file before the specified deadline.

Question 1:

Create a PHP function to identify if a provided integer argument is a valid fibonacci number.

- Return a boolean to indicate, if TRUE the provided argument was valid or FALSE for if it wasn't.

Question 2:

Create a PHP function to parse a CSV file.

- The argument for the function must be the path to the CSV file to parse
- Validate the CSV file in the following way
    - The CSV must only have 3 columns per row
    - The first column must be in the following format:
        - 3 - 5 alphabetic character
        - Hyphen
        - A numeric value
        - E,g. ABCD-123, EFG-4 or HIJKL-5678
    - The second column must be a valid unix timestamp
    - The third column must be a numeric value (decimal values are accepted)
- Ignore rows with invalid data (do not error out on invalid rows, simply ignore them)
- Calculate and return the sum of the third column of each valid row

Question 3:

Create a small API that integrates with MongoDB.

- The API must accept POST requests to `assessment-api/score/` expecting data in the JSON format.
    - Data from this request must be validated according to the schema defined below for the scores.
    - Valid data must be inserted into MongoDB.
- The API must accept GET requests to `assessment-api/score/%`
    - If an ID is provided as the third url parameter, it must attempt to retrieve the specified object from the MongoDB and return it in the JSON format.
    - If no ID is supplied it must provide a list of all the scores.

- The schema for the Scores:
    - ID (integer) - The identifier for this score
    - Title (string) - The title of the score entry
    - Score (float) - The score
    - Timestamp (timestamp) - The timestamp when this score was created
- Do not be concerned about CORS or JSONP
- Do not be concerned about authentication

General Requirements

- Comment all code
- Code style (spacing, etc) will be looked at
- And remember: work smarter not harder