**Job Description:** C#/.NET Backend Developer – Home Test 2.2021  
  
Answer 3 out of the following questions

# **The morse code server**

Build a web application (in the technology of your choice) with the following endpoints:

* POST /morse?msg=text
* POST /setup?sound=[1 - 10]
* GET /log?n=5

The *morse* endpoint should accept a string as a query string parameter and then play the string as morse code on the host computer.

In other words, if you are running this on your machine, when running:

POST /morse?msg=SOS

You should hear … --- … played. You can play the sounds however you like, including using the Beep() function or via the SpeechSynthesizer from the System.Speech package. You can assume that the input is composed of letters and digits only.

The *setup* endpoint can be called to modify the sounds in some manner. After it is called, all future morse calls should sound differently. For example, if using the Beep function, you can change the frequency of the tone.

The *log* endpoint accepts a number and returns most recent *played* messages in both textual and morse format.

Considerations:

* Playing a morse code can take a while (multiple seconds). Requests that arrive near one another should not cause the messages they play to be intermixed.
* Messages are added to the log only when they have successfully been played.
* Assuming that the log can grow to be very large, how would you handle this situation now?
* How do you handle request timing out if they are waiting to play a sound for too long?

**Bonus**: Containerize the application so we can run it in Docker.

## **Broadcast server**

Build a server application that would listen to the network. A client will connect to the server and is able to send and receive data.

The server will support **multiple concurrent connections** and will broadcast the data received from one client to all the other clients.

For example ( > indicate console input, < indicates program output):

|  |  |  |
| --- | --- | --- |
| Client 1 | Client 2 | Client 3 |
| > telnet 127.0.0.1 9999  > Foo  < Bar  < 123 | > telnet 127.0.0.1 9999  < Foo  < Bar > 123 | > telnet 127.0.0.1 9999  < Foo  > Bar  < 123 |

Clients can join and leave at any time, but will only see the data that is sent to the server while it is connected to the server. Pay attention to error handling and concurrency.

## **Phone book**

Implement the following class, representing a phone book. The data is persisted to disk as a file.

public class PhoneBook

{

    public class Entry

    {

        public string Name;

        public string Phone;

        public string Type; // home | work, etc

    }

    public PhoneBook(string filename);

    public Entry GetByName(string name);

    // will overwrite the entry if there is already

    // one with the same Name

    public void InsertOrUpdate(Entry e);

    // Get a list of all the entries, in Name order

    public IEnumerable<Entry> Iterate();

}

**All the data for the phone book should be held in a single file.**

**Adding or updating a new entry must not overwrite the entire file.**

**Aside from the filename, this class cannot have any fields or members.**

## **Search in log**

The following log file (<https://zenodo.org/record/3227177/files/Windows.tar.gz?download=1>) is a (2GB) compressed log file. You can see the format of the log here (<https://github.com/logpai/loghub/blob/master/Windows/Windows_2k.log>). Note: the uncompressed file is about 27GB in size.

Download the file and decompress it, then write a program that would be run with a filename as well as a start and end dates and output all the log lines in the file that fall within the given date range.

FindEntries Windows.log 2016-09-28 2016-09-30

**You may not read through all the records in the file.**

## **Fix the bug**

When running the [following program](https://gist.github.com/ayende/ca1de0ea5da2a9c80c37c9855c13e297) on this file (<https://zenodo.org/record/3227177/files/Android.tar.gz?download=1>), we sometimes get an error message and are unable to complete the running of the program successfully.

The program code is here <https://gist.github.com/ayende/ca1de0ea5da2a9c80c37c9855c13e297>

**Fix the program so it would complete its task with no errors.**

**Explain the manner in which you found and fixed the problem.**