Developer Technical Test

Requirements

Design and implement an HTTP API that controls an imaginary internet-connected coffee machine. Your solution should fulfil the following criteria:

1. When the endpoint `GET /brew-coffee` is called, the endpoint returns a `200 OK` status code with a status message and the current date/time in the response body as a JSON object, with the date/time formatted as an ISO-8601 value e.g.
   ```
   {
     “message”: “Your piping hot coffee is ready”,
     “prepared”: “2021-02-03T11:56:24+0900”
   };
   ```
2. On every fifth call to the endpoint defined in #1, the endpoint should return `503 Service Unavailable` with an empty response body, to signify that the coffee machine is out of coffee;
3. If the date is April 1st, then all calls to the endpoint defined in #1 should return `418 I’m a teapot` instead, with an empty response body, to signify that the endpoint is not brewing coffee today (see https://developer.mozilla.org/en-US/docs/Web/HTTP/Status/418);

Please either supply your solution as a Zip file, or create a public Git repository on e.g. GitHub or BitBucket, and share the URL.

Non-functional Requirements

Your solution should:

1. Be implemented in .NET Core; apart from that, you may pick whatever approach you like for the implementation, but be prepared to discuss the pros and cons of your choices;
2. Include tests for as many scenarios as you can come up with. They can be unit tests or integration tests or a combination of the two.

Extra Credit

Your solution delighted our customers, and now they are looking forward to the next product enhancements. Refactor your solution to deliver the following new requirement:

1. When the endpoint `GET /brew-coffee` is called, the endpoint should check a third-party weather service (e.g. https://openweathermap.org/api), and if the current temperature is greater than 30°C, the returned message should be changed to “`Your refreshing iced coffee is ready`”;
2. Original requirements #2 and #3 remain unchanged.

If you are going to attempt this part of the test, please supply the solution separately to the original – for instance, as a second Zip file, or as a Git branch. This is so that we can compare the two to see your refactoring more clearly.