Take-home Challenge Brief

Congratulations on receiving the Autodesk Infrastructure Engineer coding challenge, good luck!

At Autodesk, our Infrastructure engineers write code because we often need to debug applications and write tools and scripts to automate work.

Your task, should you accept it, is to complete the following 3 parts:

- Create a basic API
- Enable logging
- Test your service.

When your code is submitted, we will follow exactly the steps in your README in order to start the app and to run the tests. Please make sure your README lists all the steps required to install the app (and its dependencies), start the app, and run its tests.

Note: If your app does not start up or does not work when following your instructions, we will not attempt to debug your instructions.

Part 1: Writing an HTTP endpoint

Using the programming language and web framework of choice (e.g. Django, Flask, Rails, Spring, Struts, Express, etc.), please write a web app that accepts both a GET and a POST request to the / HTTP endpoint. This endpoint will have slightly different behavior depending on the Accept header (for a GET request).

In response to a GET request to /, your server should inspect the Accept header sent by the client and:

If the request does not send an Accept header, your server responds with:

```
<p>Hello, World</p>
```

If the incoming request sets an Accept header with value application/json, your server responds with:

```
{"message": "Hello, World"}
```

Part 2: Logging

While you are writing the above endpoint(s), please also include code which emits one log line per request at the debug log level. Your log lines should include a timestamp and the request URL. You should use your language's standard logging facility (e.g. the Python logging module, the Ruby Logger class, log4j for Java, etc.) and make sure to use the debug log level.

Please include instructions in the README for enabling the display of these log lines (by default the debug logs should not be shown).

The basic print function languages include does not count as a logger.

Part 3: Scripts and unit tests for validating correct behavior

Please provide sample cURL (or wget or httpie, etc) commands for quickly testing this behavior from the command line.

Please also include python/ruby/java/javascript unit tests for validating correct behavior using your chosen web framework's built-in testing facilities. The unit tests should not make network requests but should instead use your web framework's built-in ability to mock HTTP requests and route them to endpoints for unit testing.

Deliverables

1. Please create a Git repo of the application code with a README that includes:
    a. Setup instructions, including instructions on how to install prerequisite packages
    b. A command to start the application
    c. A command to run the unit tests for your application.
2. Email your zipped Git repo back to us!

Our commitment to you is that we will review your submission and get back to you quickly!