Developer Challenge 2.2

### Version History:

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Notes | Author |
| 12 OCT 2021 | 0.01 | Initial version | val.adrien.li |
|  |  |  |  |

# Introduction

This document outlines a technical assessment given to **backend** developer candidates.

This challenge aims to test most, if not all, of the following items:

|  |  |
| --- | --- |
| Language Knowledge | |
| Python 3 | Flask (backend web framework) |
|  | virtualenv/pyenv (environment isolation) |
|  | unittest (test-driven development) |
| NoSQL | JSON/CSV (data format) |

|  |  |
| --- | --- |
| Architecture & Design | |
| Web Development | REST APIs and HTTP Requests (Flask) |
|  | Authentication & Authorisation (Flask) |

|  |  |
| --- | --- |
| Other | |
| Collaboration | Documentation and git (.gitignore, committing, branching) |
| Code Quality | Object-oriented programming principles, error handling, commenting |

# Background

Independent reviews form an important part of consumer decision-making on Steam, an online distribution platform and storefront for gaming products. Users can purchase games and write reviews that are then posted publicly, allowing other users to rate the validity of their review by marking the reviews as “funny”, “helpful”, or “unhelpful”. Searching for helpful reviews is especially useful for discerning customers.

An open dataset of reviews has been provided via [Kaggle](https://www.kaggle.com/j457zhan/warframe-steam-user-reviews) (<https://www.kaggle.com/j457zhan/warframe-steam-user-reviews>) and a subset provided in *warframe\_reviews.csv* and *warframe\_reviews.json*. It has been requested that a tool be created to allow users to programmatically view and post reviews.

# Requirements

The application should be a Flask server that runs on Python 3.7.x and exposes certain endpoints. Data can be stored in any format. Authentication can use any method. It should allow users to:

1. Return reviews posted on a certain date
2. Return reviews with a “helpful” count over a certain number
3. Post a new review
4. Delete reviews if the user is authenticated as an admin

The application should also include:

1. A .git folder showing commit history;
2. A test suite with instructions on running the tests; and,
3. A readme file with notes on the design and development of the application.

# Submission

The recommended time for this challenge is 3-4 hours. If it’s not complete by then, don’t worry – the expectation isn’t a perfect application, simply enumerate outstanding items in the README.

The project should be submitted as a hosted private repository (such as on Github) and the link emailed to val.adrien.li@accenture.com.

# Appendices:

## Appendix I: Sample Submission

A sample project can be found at [https://github.com/val-something/dev\_challenge\_2\_](https://github.com/val-something/dev_challenge_2_2)2.

Contact <mailto:val.adrien.li@accenture.com> for access if not granted.