CHALLENGE: Your mission, should you choose to accept it, is to make a URL shortener API.

The way you build this Django application is using TDD and BDD principles, so be sure to have both green and significant tests and also scenarios for BDD. Feel free to use any of the tools you are most familiar with.

CORE REQUIREMENTS

• We must be able to put a URL into the home page and get back a URL of the shortest possible length. • We must be redirected to the full URL when we enter the short URL (ex: http://myshortener.whatever/a => https://google.com). • There should be a page that shows the top 100 most frequently accessed URLs. • There must be a background job (resque, sidekiq, activejob, etc) that crawls the URL being shortened, pulls the <title> from the website and stores it. • Display the title with the URL on the top 100 board. • There must be a README that explains how to setup the application and the algorithm used for generating the URL short code. * NICE TO HAVE: Write a bot to populate your DB, and include it in the source code

DELIVERABLES

NOTES / HINTS

• Deploy to Heroku or any other free host you're comfortable with. • Code must be on GitHub. • Be verbose with your commit messages as this will allow us to understand some of the decisions you make throughout the process. o Should show the URL that the app is redirecting you to • The cURL examples above are not requirements. Feel free to use different parameter names, input types, and URLs.

• Research and understand how Bit.ly and TinyURL actually work before you decide on how to generate URLs of the shortest possible length. • You may use Django, Flask or any other framework for your application. • You may use HTTParty or any other HTTP library you'd like for fetching the <title> tag from the URL. • You can use whatever database you are comfortable with. For the purposes of this exercise, we recommend a relational database. • Don't forget things like validations and demonstrating good clean Object Oriented Programming.