Practical task for candidates seeking a Junior Software Engineer at Phoundry

This task is a chance for candidates to demonstrate their practical skills and will be weighted alongside technical interview; it's imperative to perform well at both.

The Challenge

Scenario

Last night an international team of archaeologists made a startling discovery: it appears that while The Great Library of Alexandria was irreversibly lost along with the troves of ancient knowledge it contained, scholars of the old had the foresight to create a backup! A dig in Syria uncovered what appears to be a vast network of underground catacombs housing hundreds of thousands of ancient scrolls, mostly intact.

This underground library stretches across territories controlled by multiple warring factions and the sheer amount of artifacts is impractical to move, especially considering heavy shelling in the area. It seems our only choice is to create a digital copy, but we don't have much time. A fleet of drones will be deployed to map the labyrinth, photograph and make 3D models of every inch of the complex.

You have been contracted by UNESCO to create a system that would organize and store assets collected by the robotic swarm. The drones can be trivially reprogrammed to communicate over any protocol you specify. We have complete confidence in your engineering prowess. Good luck!

Requirements

1. Your solution could have the form of:

- a web system powered by a server-side language (e.g. PHP, Python, Ruby) and an HTML front-end,
- a Java application/applet,
- A single page application written in HTML5 and JS optionally talking to a REST API written in a server-side language of your choice
- a mobile application
- a REST API and the underlying backend but no front-end
- whatever else you think is appropriate

1. You may use any frameworks/tools/libraries you wish
2. You may write whatever features and functionality you wish, based on the above-mentioned scenario
3. You might want to focus on a specific area (e.g. UI/UX, database, API) and mock other components of the system with dummy placeholders/hardcodes
4. Some ideas of what your solution might include:

- a complex interactive form (with validation), possibly with file upload
- a database of your choice, with a database schema and queries to operate on that database

- a well-designed and documented HTTP API
- a mobile-first solution (e.g. an adaptive layout)
- automated tests
- functional and intuitive navigation
- a visually appealing design
- user authentication and session management
- a data table view with sorting, filtering, pagination and keyword search
- a photo gallery

TIP: don't try and do everything - this is meant to be a simple exercise. Choose one or two areas to focus on - your goal is to demonstrate your strongest skills. This is an open-ended challenge with no right or wrong answers.

General directions

All writeup (as well as code comments, variable names, etc) should be in English.

Work submitted should be your own. Where other individuals' work is used and it's not obvious please add a comment referencing the original author. For example: if you use a third-party library like jQuery or Bootstrap we consider it obvious. If you found a snippet of code on Stack Overflow and integrated it into your solution - please reference the original source (by adding a comment in your code, for example).

Academic misconduct is conduct by which a candidate misrepresents his or her academic accomplishments, or impedes other candidates' opportunities of being judged fairly for their academic work. Knowingly allowing others to represent your work as their own is as serious an offense as submitting another's work as your own. Your submission will be automatically tested for plagiarism. If found you will be immediately disqualified from the selection process.

Evaluation criteria

When evaluating your solution we will be looking for evidence of the following:

- your capacity to start with a vague specification and move through the stages of: design, implementation, refactoring, testing, bug-fixing, documentation and delivery
- your ability to manage your time and prioritise
- your capacity for writing a program that works
- your code style and and code organisation. Pay attention to things like code repetition, readability, consistency and formatting, naming conventions, etc
- your proficiency with chosen tools and technologies
- your ability to independently overcome problems. To that end you may want to supplement your solution with a brief cover letter explaining the choices you faced, the issues you encountered and how you've overcome them, as well as rationale behind your decisions. While not strictly necessary such commentary is an additional opportunity show off your abilities. If you choose to do this please add a DEVNOTES.md file at the root of your repository.

Submission

You will submit your work via a public GitHub repository by sending us a repository link. You will not force push once you've sent us the link. Your solution will be licensed under MIT License: https://opensource.org/licenses/MIT

The repository should contain:

- source code of your solution
- any tools/dependencies required to build and run your solution (i)
- instructions on how to run your solution (ii)

i. You don't actually need to push dependencies that can be freely downloaded - it's enough to reference them. For example - don't upload your entire `node_modules` directory - just make sure `package.json` references all the dependencies. Similarly, you don't need to include PHP nor Apache, but you should state version numbers and custom configuration, if any.

ii. You may develop on any platform you choose, but your solution will be evaluated on OSX or Linux. Please provide detailed instructions in README.md at the root of your repository.