Bravi
Back End - Technical Assessment

Hello candidate,

Thank you for your application to Bravi. For this stage we invite you to complete our Technical Assessment.

These are some of the things we will be evaluating:

- Problem-solving skills
- How you test your application
- Good practices (clean code, code organization)
- How you use git (good practices)
- English code / commits / documentation

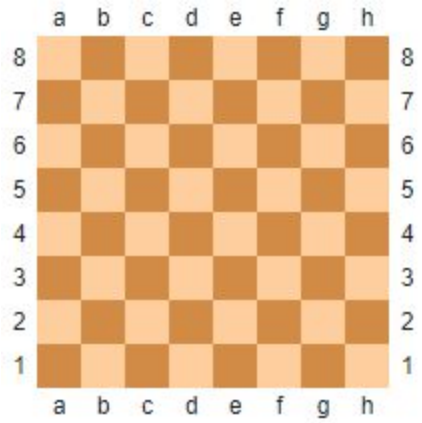
We expect you to explain your project in the readme file. Also feel free to include any other technical information you think is relevant (ex: notes of things you could have done differently).

Solve the following challenge using Python (better with Django) and Git.

Chess Board - Finding out knight moves

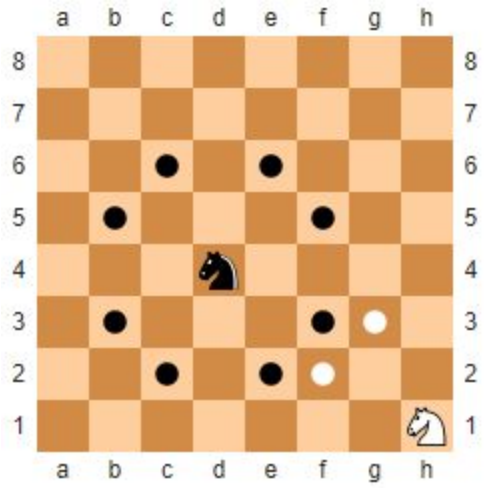
Some context on chess

A chessboard is the type of checkerboard used in the board game chess, consisting of 64 squares (eight rows and eight columns). The squares are arranged in two alternating colors (light and dark). https://en.wikipedia.org/wiki/Chessboard



A chessboard generally use [Algebraic notation], basically rows are marked 1 to 8 from bottom to top and columns a to h from left to right. With that players can make a move using coordinates.

The knight (♘ ♞ /ˈnaɪt/) is a piece in the game of chess, representing a knight (armored cavalry). It is normally represented by a horse's head and neck. It moves to a square that is two squares away horizontally and one square vertically, or two squares vertically and one square horizontally. The complete move therefore looks like the letter L.



Challenge

You must develop an application that allows the registration of chess pieces (type/name and color). In addition, given a location on a coordinate chosen by the user and the piece id, if it is a knight, find out all possible locations where the knight can move in 2 turns.

You are expected to build an API.

API

The api should:

- receive the piece type/name and the color and return the piece id;
- receive cell coordinate (in Algebraic notation) and the piece id and return an array with all possible locations where the knight can move within 2 turns.

Bonus

You must focus on the main application features specified above and it is not mandatory that you implement the bonus features. However you can get some extra points for implementing them correctly:

- PEP8
- Lint
- Coverage
- add docker/docker-compose
- allow user to dynamically add rows/columns to the chessboard
- save logs of all requests and results in a database

Submit

Once completed you should push your code to GitHub (or another git service) and send us the link.

May the code be with you!