Coding challenge for Full-Stack position

Task #1

Create a webservice that makes use of a vocabulary of words defined in words.txt.

- This vocabulary file contains a single word per line, each word being composed of ASCII alphanumerical characters without any punctuation or blank characters.

The webservice is to expose an endpoint implementing a simple T9 matching functionality. The endpoint is to accept a non-empty sequence of decimal digits and yield a list of words, from the vocabulary above, matching that non-empty sequence.

The endpoint is to support both strict and prefix matching, depending on the client's request, the default matching mode being prefix matching.

For instance, imagine you have the following words in your vocabulary: "gone", "hoof" and "inoffensive". The characters in these three words, when translated through the T9 keypad layout, would give the decimal sequences "4663", "4663" and "46633367483", respectively.

Let's say the non-empty decimal digit sequence sent as a request is "4663".

- In strict matching mode, the endpoint must yield the word list "gone", "hoof".
- In prefix matching mode, the endpoint must yield the word list "gone", "hoof", and "inoffensive".

The list of matching words yielded must be in lexicographically increasing order, ignoring lower and upper cases.

Task #2

Create a lightweight web UI, loaded at a different endpoint of the same webservice, which has two UI elements:

- a text box accepting only digits,
- and a simple list box listing words underneath.

The web UI must use the service endpoint developed above, in Task #1, to achieve the following:

- The web UI is to present, in the list box, the list of words matching any non-empty sequence of digits typed into the textbox.
- If the input is only a sequence of digits without 0, prefix matching is to happen,
  - e.g. the digit sequence "4663" should display "gone", "hoof", and "inoffensive" in the listbox, given that 3-word vocabulary above.
- If the input is terminated by a 0 digit - which represents space - the digit sequence leading up to the terminating 0 digit is to be matched strict,
  - e.g. the digit sequence "46630" should display only "gone" and "hoof" in the listbox.

Task #3

Please dockerize your solution.

Task #4

Provide a brief documentation, as you would explain it to a fellow developer: how to build / run it, explanation of your choices, etc.

Non-functional requirements

- Do *not* deploy this to anywhere, and do *not* put your source files to any public service, e.g. GitHub.
  - Just send the source files, we'd like to read those, of course.
- A very simple solution with minimal amount of source files is deeply appreciated.

Not a requirement

- No particular tech stack is set: you can use whatever tech stack you are most comfy with.
- A fully automatic build system, with bells and whistles, integrated to a CI/CD pipeline is not required either.
- You do not have to store the vocabulary in an actual DB, the service endpoint may load it on startup.
- No need to make the web interface overly fancy.