Expenses Input

Goal

Produce a simple web-app that satisfies the requirement below. Please organize, design, test, and document your code as if it were going into production, then send us a link to the hosted repository and server. More on deployment below.

The task

Imagine that you come back from 2 weeks of holidays on a Monday. On the team Kanban board, assigned to you, two tasks await:

User story 1:

> *As a user, I want to be able to enter my expenses and have them saved for later.*

As a user, in the application UI, I can navigate to an expenses page. On this page, I can add an expense, setting:

- *The date of the expense*
- *The value of the expense*
- *The reason of the expense*

When I click "Save Expense", the expense is then saved in the database. The new expense can then be seen in the list of submitted expenses.

User story 2:

> *As a user, I want to be able to see a list of my submitted expenses.*

As a user, in the application UI, I can navigate to an expenses page. On this page, I can see all the expenses I have already submitted in a tabulated list. On this list, I can see:

- *The date of the expense*
- *The VAT (Value Added Tax) associated to this expense. VAT is the UK's sales tax. It is 20% of the value of the expense, and is included in the gross amount entered by the user.*
- *The reason for the expense*

Your job

Create a RESTful full-stack application that solves the ticket:

1. Provide your solution to user story `1` and user story `2`
2. Create a README containing instructions on **how to build and run your app**.
3. **Host it**! When you're done, host it somewhere (e.g. on Amazon EC2, Heroku, Google AppEngine, etc.). Please do so **after** you have finished the challenge, meaning, submit us your repository within the allotted 3 hours, and when you have emailed us, you may spend as much time as you like to put it on a server (deploying is something that's learned on the job, so we don't want you to waste your challenge time deploying it)

Technical spec

The architecture will be split between a back-end and a web front-end. Feel free to use any other technologies provided that the general client/service architecture is respected.

Choose **one** of the following technical tracks that best suits your skillset:

1. **Full-stack**: include both front-end and back-end.
2. **Back-end track**: include a minimal front-end (e.g. a static view or API docs). Write, document and test your API as if it will be used by other services.
3. **Front-end track**: include a minimal back-end, or use the data service directly. If need be, mock the data service. Focus on making the interface as polished as possible.

README

Please write your README as if it were for a production service. Include the following:

- Description of the problem and solution. Since your hypothetical team lead isn't here with you to clarify requirements, write any assumptions you make.
- Whether the solution focuses on back-end, front-end or if it's full stack.
- Reasoning behind your technical choices, including architectural.
- Trade-offs you might have made, anything you left out, or what you might do differently if you were to spend additional time on the project.

Questions

What frameworks can I use?

That’s entirely up to you, as long as they’re Open Source Software (OSS). We’ll ask you to explain the choices you’ve made. Please pick something you're familiar with, as you'll need to be able to discuss it. If you choose to use a framework that results in boilerplate code in the repository, please detail in your README which code was written by you (as opposed to generated code).

What application servers can I use?

Anyone you like, as long as it’s available OSS. You’ll have to justify your decision. Please pick something you're familiar with, as you'll need to be able to discuss it.

What database should I use?

Whatever you want. You can use SQL or NoSQL, just explain your database choice.

What will you be grading me on?

Elegance, robustness, understanding of the technologies you use, tests, security.

Will I have a chance to explain my choices?

Feel free to comment your code, or put explanations in a pull request within the repo. We’ll be asking questions about why you made the choices you made.

Why doesn’t the test include X?

Good question. Feel free to tell us how to make the test better.