What is this assignment?
This assignment is specifically designed to test the MVC skills of a developer candidate (which is probably you, the reader :))

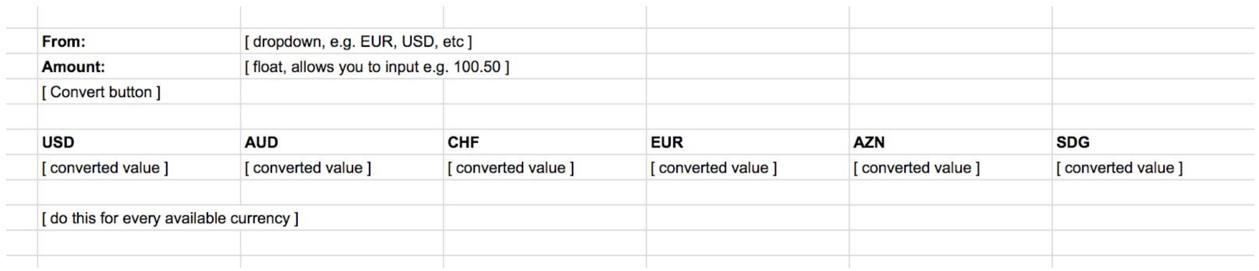
What techniques should be used?
We would prefer you use an MVC framework such as Symfony or CakePHP, combined with MySQL or MariaDB.

How can I share my code?
Setup a git repository (for example on Github or Gitlab) and share it with us. We would prefer you include a README file that helps us to get your repo working and we would prefer you use migrations above an .SQL file.

How important is the look and feel of the application?
We would encourage you to use e.g. bootstrap so that it looks presentable without too much of an effort. Don't spend too much time on making it look great, we're more interested in the backend functionalities. If however you feel like creating a real frontend gem, feel free to :)

What are the 'bonus/optional' questions?
They are suggestions for the application that are not required but would make for a better application. We encourage you to consider adding these but don't worry if you don't.

The actual assignment
We want to have a simple web page, secured with IP (subnet) restrictions and username/password authorization, that allows you to choose a FROM currency (e.g. USD, EUR, YEN.. should be in a pulldown) and enter a number (float). When you click on the 'Convert' button, a table is shown in which the FROM is converted to various TO currencies. It would look like this:



From:	[dropdown, e.g. EUR, USD, etc]				
Amount:	[float, allows you to input e.g. 100.50]				
[Convert button]					
USD	**AUD**	**CHF**	**EUR**	**AZN**	**SDG**
[converted value]	[converted value]	[converted value]	[converted value]	[converted value]	[converted value]
[do this for every available currency]					

In order to do this, you must create a task that is able to import all exchange rates from http://www.floatrates.com/json-feeds.html into a MySQL-compatible database. The task has to be able to update rates on demand and should also be easy to put in a cronjob. Use a log file on the server or logging framework to log some stuff that may be useful when debugging the application in a live environment.

Before the user can use this functionality, he should login to the application using a username and a password. The user can login only if his IP is added to the authorization database. There should also be a ‘Remember me’ checkbox. **Bonus/optional:** create a ‘Forgot password’ button that allows the user to reset their password.

Also create a task that allows you to add and remove users and IP’s to the application so that they are able to login to the frontend. **Bonus/optional:** create a simple admin screen in the frontend that allows you to do this as well.

Please take these things into consideration:

1. Take into account code quality and maintainability
2. Validate data as if it were real users using the system
3. Use logical git commits and commit messages so we can see your working process

And some more bonus/optional things:

1. Use Elasticsearch to store (some of) the data to increase efficiency or store historical data
2. Supply a docker-compose.yml file that allows you to run the application using Docker, including the database server