Create compact RESTful web service for payments processing.

Please note that it's better to only implement half of requirements but do absolutely best you can.

#### Requirements

##### Payment creation

Client should be able to create payment of one of 3 types - TYPE1, TYPE2, TYPE3. Fields 'amount' (positive decimal), 'currency' (EUR or USD), 'debtor\_iban' and 'creditor\_iban' (texts) are mandatory for all types.

Additional type-specific requirements:

1. TYPE1 is only applicable for EUR payments, has additional field 'details' (text) which is mandatory;
2. TYPE2 is only applicable for USD payments, has additional field ‘details’ (text) which is optional.
3. TYPE3 is applicable for payments in both EUR and USD currency, has additional field for creditor bank BIC code (text) which is mandatory.

##### Payment cancelation

Client should be able to cancel the payment. It is possible to cancel payment only on the day of creation before 00:00. When cancel happens, cancelation fee should be calculated and saved along the payment in database.

Cancelation fee is calculated as: *h \* k*

Where *h* - number of full hours (2:59 = 2h) payment is in system; *k* - coefficient (0.05 for TYPE1; 0.1 for TYPE2, 0.15 for TYPE3). Result is an amount in EUR.

##### Payments querying

It should be possible to query all payments that aren't canceled as well as filter them by amount. Query should return payment IDs.

There should also be an option to query specific payment by ID, and it should return payment ID and cancelation fee.

##### Client country logging

Resolve clients country (use any external web service to resolve it by user IP) and write it to the log (that’s OK if it will fail sometimes). Information about clients country won't be required anywhere in business logic.

##### Notification

Imagine other (micro)service should be notified about the fact we saved valid TYPE1 or TYPE2 payment. The fact that external service was (un-)successfully notified should be saved to our database.

Send HTTP GET requests to resources of your choice (should be different for TYPE1 and TYPE2 payments). Consider any 2XX response status code as a success.

### What you'll get karma points for:

* Application efficiency - please be careful with what you make client wait for;
* Testability (proved with tests :) );
* Appropriate usage of design patterns - real-life services you're going to work on will be much bigger - *sometimes* it might be a good idea to get abstract;
* Writing “defensive” code - it's really important to be careful working with people's money;
* Clean code - imagine your code will be reviewed by Robert Martin;
* Ability to argue on various “why”s about framework/design/code/whatever decisions.

It would also be nice to be able to launch app and get all dependencies up and running in one command.

Do not hesitate to ask questions!