Work Here

We are a company that ...

Our development team consists of developers who loves what they do. Our agile development processes and our search for the best development practices provide a great environment for professionals who like to create quality software in good company.

We are always looking for good programmers who love to improve their work. We give preference to small teams with qualified professionals over large teams with average professionals.

This repository contains a problem used to evaluate the candidate skills. It's important to notice that satisfactorily solving the problem is just a part of what will be evaluated. We also consider other programming disciplines like documentation, testing, commit timeline, design and coding best practices.

Hints:

- Carefully read the specification to understand all the problem and artifact requirements before start.
- Check the recommendations and reference material at the end of this specification.

How to participate

1. Make a fork of this repository on Github. If you can't create a public fork of this project, make a private repository;
2. Follow the instructions of this file;
3. Deploy your project on a host service (we recommend Heroku);
4. Apply for the position at our career with:
    - Link to the fork on Github;
    - Link to the project in a the deployed host service.

Specification

You should implement a Python application that receives call detail records and calculates monthly bills for a given telephone number.

This Python application must provide a HTTP REST API to attend the requirements.

1. Receive telephone call detail records

There are many telecommunications platform technologies that can potentially be clients of this system. Each one has its own communication flow. It's not safe to believe that when an already sent record can be resent or retrieved later. This context requires system flexibility in receiving information to avoid record loss.

There are two call detailed record types: **Call Start Record** and **Call End Record**. To get all information of a telephone call you should use the records pair.

Call Start Record information:

- **record identifier**: Record unique identificator;
- **record type**: Indicate if it's a call start or end record;
- **record timestamp**: The timestamp of when the event occured;
- **call identifier**: Unique for each call record pair;
- **origin phone number**: The subscriber phone number that originated the call;
- **destination phone number**: The phone number receiving the call.

The Call End Record has the same information
excepting **origin** and **destination** fields.

The phone number format is *AAXXXXXXXXX*, where *AA* is the area code and *XXXXXXXXX* is the phone number. The phone number is composed of 8 or 9 digits.

Examples

1. Call Start Record

```
{
  "id":  // Record unique identificator;
  "type":  // Indicate if it's a call "start" or "end" record;
  "timestamp":  // The timestamp of when the event occured;
  "call_id":  // Unique for each call record pair;
  "source":  // The subscriber phone number that originated the call;
```

```
  "destination":  // The phone number receiving the call.
}
```

1. Call End Record

```
{
   "id":  // Record unique identificator;
   "type":  // Indicate if it's a call "start" or "end" record;
   "timestamp":  // The timestamp of when the event occured;
   "call_id":  // Unique for each call record pair.
}
```

2. Get telephone bill

To get a telephone bill we need two information: the subscriber telephone number (required); the reference period (month/year) (optional). If the reference period is not informed the system will consider the last closed period. In other words it will get the previous month. It's only possible to get a telephone bill after the reference period has ended.

The telephone bill itself is composed by subscriber and period attributes and a list of all call records of the period. A call record belongs to the period in which the call has ended (eg. A call that started on January 31st and finished in February 1st belongs to February period).

Each telephone bill call record has the fields:

- destination
- call start date
- call start time
- call duration (hour, minute and seconds): e.g. 0h35m42s
- call price: e.g. R$ 3,96

3. Pricing rules

The call price depends on fixed charges, call duration and the time of the day that the call was made. There are two tariff times:

1. Standard time call - between 6h00 and 22h00 (excluding):
    - Standing charge: R$ 0,36 (fixed charges that are used to pay for the cost of the connection);

    - Call charge/minute: R$ 0,09 (there is no fractioned charge. The charge applies to each completed 60 seconds cycle).
2. Reduced tariff time call - between 22h00 and 6h00 (excluding):
    - Standing charge: R$ 0,36
    - Call charge/minute: R$ 0,00 (hooray!)

It's important to notice that the price rules can change from time to time, but an already calculated call price can not change.

Examples

1. For a call started at 21:57:13 and finished at 22:10:56 we have:
    - Standing charge: R$ 0,36
    - Call charge:
        - minutes between 21:57:13 and 22:00 = 2
        - price: 2 * R$ 0,09 = R$ 0,18
    - Total: R$ 0,18 + R$ 0,36 = R$ 0,54

4. Sample data

Insert the following calls to your app after it is deployed to a working environment (eg. Heroku). This sample data will be used in your evaluation, so do this as the last step before submitting the project.

These calls are between the numbers 99988526423 (source) and 9993468278 (destination).

- call_id: 70, started at 2016-02-29T12:00:00Z and ended at 2016-02-29T14:00:00Z.
- call_id: 71, started at 2017-12-12T15:07:13Z and ended at 2017-12-12T15:14:56Z.
- call_id: 72, started at 2017-12-12T22:47:56Z and ended at 2017-12-12T22:50:56Z.
- call_id: 73, started at 2017-12-12T21:57:13Z and ended at 2017-12-12T22:10:56Z.
- call_id: 74, started at 2017-12-12T04:57:13Z and ended at 2017-12-12T06:10:56Z.

- call_id: 75, started at 2017-12-12T21:57:13Z and ended at 2017-12-13T22:10:56Z.
- call_id: 76, started at 2017-12-12T15:07:58Z and ended at 2017-12-12T15:12:56Z.
- call_id: 77, started at 2018-02-28T21:57:13Z and ended at 2018-03-01T22:10:56Z.

Project Requirements:

- Provide a working environment with your project (eg. Heroku)
- Use Python >= 3.X
- Choose any Python web framework you want to solve the problem
- Every text or code must be in English
- Use PEP-8 for code style
- Write the project documentation containing:
  - Description;
  - Installing and testing instructions;
  - Brief description of the work environment used to run this project (Computer/operating system, text editor/IDE, libraries, etc).
- Provide an API documentation (in english);
- Variables, code and strings must be all in English.

Recommendations

- Write tests!
- Practice the 12 Factor-App concepts;
- Use programming good practices;
- Use a good Python Coding Style;
- Use git best practices (https://www.git-tower.com/learn/git/ebook/en/command-line/appendix/best-practices), with clear messages (written in English);
- Be aware when modeling the database;

- Be careful with REST API details. They can bite you!

Have fun!