

Programming Challenge - Developer Position

Please read this document carefully from beginning to end. The purpose of this test is to assess your technical programming skills. The challenge consists of parsing this text file (<https://github.com/ByCodersTec/desafio-ruby-on-rails/blob/master/CNAB.txt>) and saving its information (financial transactions) into a database of your choice. This challenge should be completed by you at home. Take as much time as you need, but usually you shouldn't need more than a few hours.

Challenge Submission Instructions

1. First, fork this project to your Github account (create one if you don't have it).
2. Next, implement the project as described below in your local clone.
3. Finally, send the project or the fork/link to your bycoders contact with a copy to rh@bycoders.com.br.

Project Description

You received a CNAB file with financial transaction data from several stores. We need to create a way for this data to be imported into a database.

Your task is to create a web interface that accepts uploads of the CNAB file ([CNAB.txt - ByCodersTec/desafio-ruby-on-rails · GitHub](https://github.com/ByCodersTec/desafio-ruby-on-rails/blob/master/CNAB.txt)), normalizes the data, stores it in a relational database, and displays this information on the screen.

Your web application MUST:

1. Have a screen (via a form) to upload the file (extra points if you don't use a popular CSS Framework).
2. Parse the received file, normalize the data, and correctly save the information in a relational database. **Pay attention to the documentation** below.
3. Display a list of imported operations by store, including a total account balance.
4. Be written in your preferred programming language.
5. Be simple to configure and run, working in a Unix-compatible environment (Linux or Mac OS X). It should use only free or open-source languages and libraries.
6. Use Git with atomic and well-described commits.
7. Use PostgreSQL, MySQL, or SQL Server.

8. Have automated tests.
9. Use Docker compose (Extra points if you use it).
10. Include a README file describing the project and its setup.
11. Include information describing how to consume the API endpoint.

Your web application does NOT need to:

1. Handle authentication or authorization (extra points if it does, even more if authentication is via OAuth).
2. Be written using any specific framework (but there's nothing wrong with using them, use what you prefer).
3. Document the API (This is a plus and will earn extra points if you do it).

CNAB Documentation

Field Description	Start	End	Size	Comment
Type	1	1	1	Transaction type
Date	2	9	8	Date of occurrence
Value	10	19	10	Transaction amount. <i>Note:</i> The value in the file must be divided by one hundred (value / 100.00) to normalize it.
CPF	20	30	11	Beneficiary's CPF
Card	31	42	12	Card used in the transaction
Time	43	48	6	Time of occurrence in UTC-3 timezone
Store Owner	49	62	14	Store representative's name
Store Name	63	81	19	Store name

Transaction Types Documentation

Type	Description	Nature	Sign
1	Debit	Income	+
2	Boleto	Expense	-
3	Financing	Expense	-
4	Credit	Income	+
5	Loan Receipt	Income	+
6	Sales	Income	+
7	TED Receipt	Income	+
8	DOC Receipt	Income	+
9	Rent	Expense	-

Evaluation

Your project will be evaluated according to the following criteria:

1. Does your application meet the basic requirements?
2. Did you document how to configure the environment and run your application?
3. Did you follow the challenge submission instructions?
4. Quality and coverage of unit tests.

Additionally, we will try to assess your familiarity with standard libraries, as well as your experience with object-oriented programming based on your project's structure.

Good luck!