Teste de backend - Bcredi

About Bcredi

After 12 years of experience in the real estate backed-credit industry, we decided to leverage our expertise by developing a proprietary technology that enables partners to monetize and build customer loyalty throughout the 10+ years lifecycle of a credit loan.

Along with that, our mindset and efforts are focused on giving Brazilians homeowners access to more affordable credit products while providing financial education through gamification.

Nowadays, Bcredi's team is composed of 110 employees split into two locations: the headquarters in Curitiba and the strategic office in São Paulo.

As developers, we are always looking to improve our work and ourselves as professionals, following agile principles, code quality, good design and architecture to delivery high valuable software to make our lives easier. We give preference to small teams with qualified professionals over big teams with average professionals.

The challenge

This challenge is to build a small billing (or collection) management app.
The application will have a number of contracts with delayed installments. Each contract may have many delayed installments.

What is a contract?
When we close a deal, it becomes a contract. At Bcredi, loan contracts usually take 60-180 months to get paid, so they are under our management until then.

What is a delayed installment?
An installment is what the customer needs to pay each month until the end of the contract. When the customer delays a payment, we have a delayed installment.

Why do we work with delayed installment?

The reason to work with delayed installments is because this app is intended to manage contracts that are under default. So with this app we give the possibility for the user to schedule bank slips for the delayed installments of the contract.

How this app will be used?
Customers delay payments regularly, so we need to renegotiate payment conditions, schedule bank slips (“boletos bancários”) to customer pay, and so on. For this challenge, we want to generate bank slips for one or more delayed installments and keep track of what was paid.

Do I need to do a real bank slip implementation with barcode?
No. You don’t need to implement any barcode or anything else. Just save the data inside your database. No integrations or libraries are necessary for this feature.

Specification

You should implement an application with the following features:
- import two CSV files (one for contracts and another for delayed installments)
- an interface to list all contracts
- an interface to show a contract detail with all delayed installments
- an interface to select one or more delayed installments and create a bank slip.
- after creation, send an email with a bank slip to the customer.
- an API to receive payment hooks from other systems.

Data import feature

→ As a user, I want to import the contracts file regularly to have all system data up to date with other systems using a form with file upload.
→ As a user, I want to import the delayed installments file regularly to have all system data up to date with other systems using a form with file upload.

The csv files have the following formats:

contracts.csv

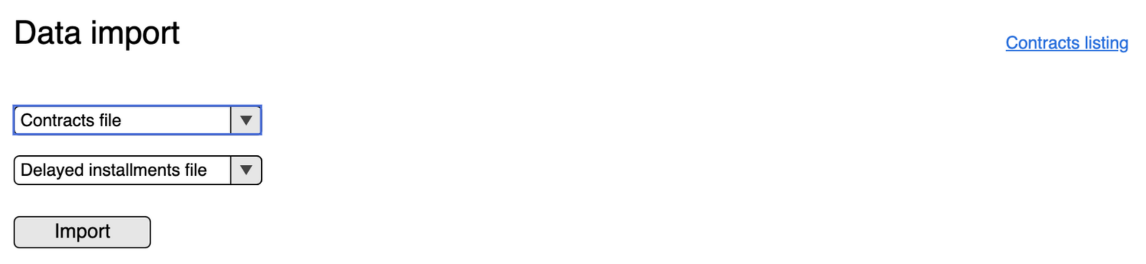
```
external_id, customer_name, customer_email, customer_cpf, loan_value, payment_term, realty_address
000003-5, João da Silva, joao@silva.com, 779.044.849-86, 150000.00, 120, "Av Silva Jardim, 496, Curitiba, PR"
```

52.8.102.3

```
000031-0, João das Neves, joao@neves.com, 258.280.504-19, 300000.00, 180, "Av Silva Jardim, 497, Curitiba, PR"
000040-A, José Johnson, jose@johnson.com, 741.574.644-94, 120000.00, 80, "Av Silva Jardim, 498, Curitiba, PR"
```

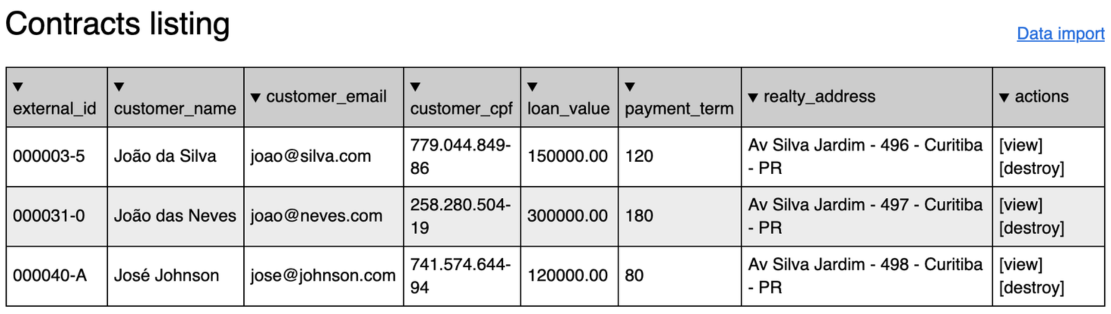
delayed_installments.csv

```
contract_id, installment_index, due_date, value
000003-5, 084/120, 2019-06-23, 1415.52
000003-5, 085/120, 2019-07-23, 1382.19
000031-0, 074/180, 2019-05-10, 2811.90
000031-0, 075/180, 2019-06-10, 2750.52
000031-0, 076/180, 2019-07-10, 2689.05
000040-A, 067/80, 2019-05-20, 793.39
000040-A, 068/80, 2019-06-20, 770.56
000040-A, 069/80, 2019-07-20, 747.98
```



Contracts listing feature
→ As a user, I want to see all the contracts that were imported using the *Data Import* feature.
→ As a user, I want to click and go to a page with contract detail.
→ As a user, I want to delete any contract from the listing.

To list all contracts, it can be a simple table. Each contract needs a link to see more details, like all delayed installments for the contract, and another link to delete the contract from the database.



Contracts listing

Data import

▼ external_id	▼ customer_name	▼ customer_email	▼ customer_cpf	▼ loan_value	▼ payment_term	▼ realty_address	▼ actions
000003-5	João da Silva	joao@silva.com	779.044.849-86	150000.00	120	Av Silva Jardim - 496 - Curitiba - PR	[view] [destroy]
000031-0	João das Neves	joao@neves.com	258.280.504-19	300000.00	180	Av Silva Jardim - 497 - Curitiba - PR	[view] [destroy]
000040-A	José Johnson	jose@johnson.com	741.574.644-94	120000.00	80	Av Silva Jardim - 498 - Curitiba - PR	[view] [destroy]

Contract detail feature

→ As a user, I want to see all informations about the contract.
→ As a user, I want to see a list of delayed installments for the contract.
→ As a user, I want to see a list of bank slips.

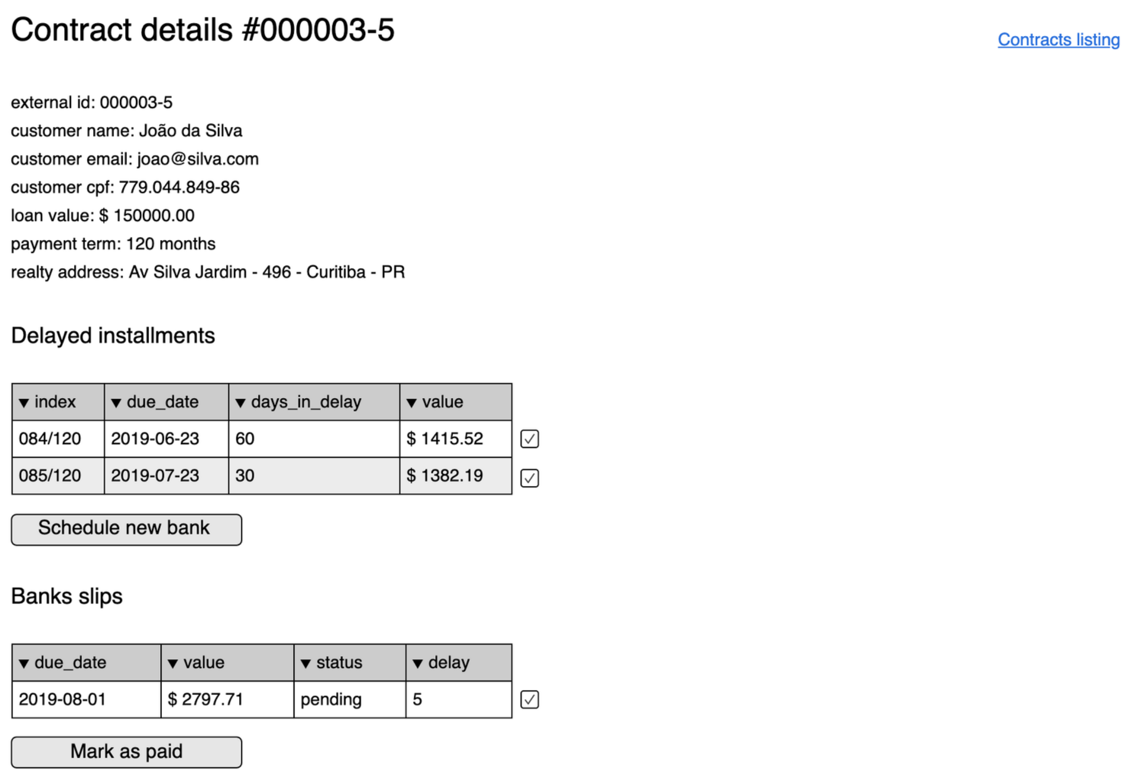
For the installments list, the following columns are needed:

- installment index (085/120)
- due date
- value
- days in delay (differente between due date and today)

For the bank slips list, the following columns are needed:

- due date
- value
- status (pending or paid)
- delay (when current date <= due date, then delay is 0; when current date > due date, then show the difference between them)

52.8.102.3



Contract details #000003-5

external id: 000003-5
customer name: João da Silva
customer email: joao@silva.com
customer cpf: 779.044.849-86
loan value: $ 150000.00
payment term: 120 months
realty address: Av Silva Jardim - 496 - Curitiba - PR

Delayed installments

▾ index	▾ due_date	▾ days_in_delay	▾ value	
084/120	2019-06-23	60	$ 1415.52	☑
085/120	2019-07-23	30	$ 1382.19	☑

Banks slips

▾ due_date	▾ value	▾ status	▾ delay	
2019-08-01	$ 2797.71	pending	5	☑

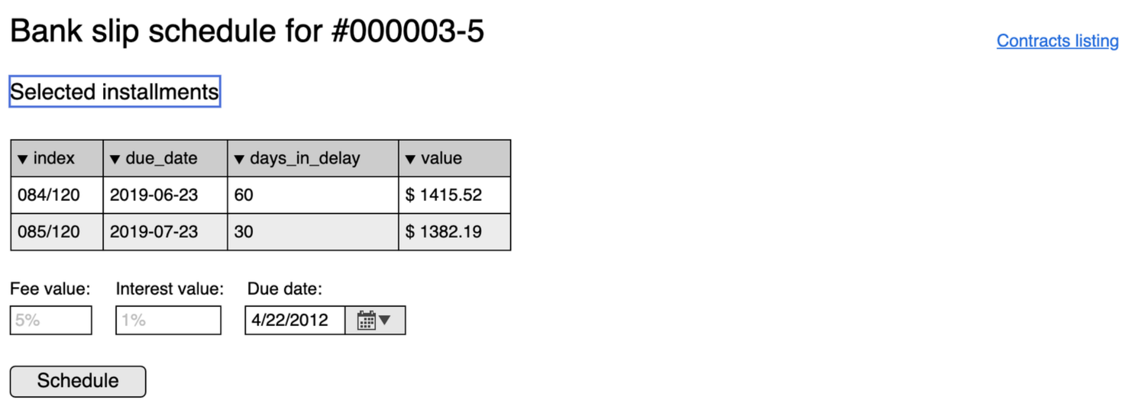
Bank slip schedule feature
→ As a user, when I am seeing the contract detail, I want to select one or more delayed installments, click on the button with the label schedule and open a new page with the selected installments.
→ As a user, when opening the page with selected installments, I want to fill the form with: fee value, interest value, and due date. After fill the form, I want to submit it.
→ As a user, when submit the form, I want to send an automated email as plain text with bank slip info.

Data specs:

fee (% percentual): Applied over the installments sum.
interest (% percentual): Compound interest over the installments sum.
due date (date YYYY-mm-dd): Limit date to the customer pay.

PS: you don't need implement a real bank slip (with barcode, etc.), only save in database and send a simple plain text/html email. For real cases a payment gateway may be used or a library that already do that.

52.8.102.3


Bank slip schedule for #000003-5

▼ index	▼ due_date	▼ days_in_delay	▼ value
084/120	2019-06-23	60	$ 1415.52
085/120	2019-07-23	30	$ 1382.19

Fee value: 5%
Interest value: 1%
Due date: 4/22/2012

Example:

Contract: 000003-5
Today is: 2019-07-24

Installments selected:

```
1, 000003-5, 084/120, 2019-06-23, 1415.52
2, 000003-5, 085/120, 2019-07-23, 1382.19
```

Sum: 1415.52 + 1382.19 = 2797.71

user applied the following values:
→ **fees (percentual based on total):** 5%
→ **interest (percentual for each day in delay):** 1%
→ due date: 2019-08-10

Then, the bank slip will be saved in database with the following values:

- due date: 2019-08-10
- value: 2797.71 + (2797.71 * 0,05) + (1415.52 * $(1 + 0,01)^{30}$) + (1382.19 * $(1 + 0,01)^{1}$)
- status: pending
- installments: [1, 2]

Data checks:

- valid status are: pending, paid.
- due date aways need to be greater than today.
- value need to be equal or greater than the sum of selected installments.

52.8.102.3

The installments probably will be a many_to_many relationship between **bank slips** and **installments.**

Payment notification feature
→ As an external system, I want to notify the system using a POST request, update the bank slip status to paid and remove the delayed installment from the listing.

Project requirements

- This project needs to be written in: Elixir, Ruby, Node, Python, Java or C#;
- Every text or code must be in English;
- Write the project documentation containing:
  - Description;
  - Installing and testing instructions;
- Automated tests.

Recommendations

- 12 Factor-App concepts;
- SOLID design principles;
- Be aware of best practices when modeling the database.
- Keep it simple.

52.8.102.3