Exercise on Atomicity

- Design a banking application

It should support transfer of money from **`Customer A`** to **`Customer B`**. Test your code.

The model of a Customer, referred to as person looks as follows

When money is transferred from Person A, deduct the amount of money from Person A and increase the value of money in Person B.

Some instructions on setting this up

- Create an application called **`banking`** (**`python manage.py startapp banking`**)
- Use the Django form API - https://docs.djangoproject.com/en/4.1/topics/forms/
- Create the model class as shown in the ERD example above.
  - Use the auto_now feature for **`joining_date`** so that it created automatically by Django's ORM
  - Add a **`__str__`** method for easy visualization in either Admin or shell interfaces.
- Register the Customer class model with admin (this will also help us quickly visualize the data).
- Create two form classes; 1 will be for creating a new Customer, the second will be for supporting money transfer between 2 customers.

A few extra hints on money will be helpful.

Please note that money will be saved as cents. For example €2.80 will be saved as 280 cents in the database. Do not store any decimal form of money. Convert it to the nearest hundred where possible. Where possible, handle any expected errors you can think of.

The wireframe of the form in the frontend of the application looks like this: