Create a Web Api application (NET5.0 or NET6.0) that would calculate net salary given the gross value as input. The taxation rules in the country of Imaginaria as of date are as follows:

1. There is no taxation for any amount lower or equal to 1000 Imagiaria Dollars (IDR).
2. Income tax of 10% is incurred to the excess (amount above 1000).
3. Social contributions of 15% are expected to be made as well. As for the previous case, the taxable income is whatever is above 1000 IDR but social contributions never apply to amounts higher than 3000.
4. CharitySpent – Up to 10% of the gross income is allowed to be spent for charity causes. It then needs to be subtracted from the gross income base before the taxes are calculated.

Example 1: George has a salary of 980 IDR. He would pay no taxes since this is below the taxation threshold and his net income is also 980.

Example 2: Irina has a salary of 3400 IDR. She owns income tax: 10% out of 2400 => 240. Her Social contributions are 15% out of 2000 => 300. In total her tax is 540 and she gets to bring home 2860 IDR

Example 3: Mick has a salary of 2500 IDR. He has spent 150 IDR on charity causes during the year. His taxable gross income is 1500 – 150 = 1350 IDR owns income tax: 10% out of 1350 => **135**. His Social contributions are 15% out of 1350 => **202.5**. In total her tax is **337.5** and he gets to bring home **2162.5** IDR

Example 4: Bill has a salary of 3600 IDR. He has spent 520 IDR on charity causes during the year. His taxable gross income is 3600 – 360 = 3240 IDR owns income tax: 10% out of 2240 => **224**. His Social contributions are 15% out of 2000 => **300**. In total her tax is **524** and she gets to bring home **3076** IDR

***Directions***

The Api should have one Controller (Calculator) with one endpoint Calculate (POST) and should accept a TaxPayer contract with the following properties **FullName** , **DateOfBirth**, **GrossIncome**, **SSN**, **CharitySpent**.

*The following validation should apply:*

* FullName – at least two words separated by space – allowed symbols letters and spaces only (mandatory)
* SSN – a valid 5 to 10 digits number unique per taxpayer (mandatory) (e.g. 12345, 6543297811)
* GrossIncome – a valid number for the amount for gross income (mandatory)
* CharitySpent - a valid number for the amount of annual charity spent (optional)
* DateOfBirth - a valid date (optional)

The endpoint should return the response contract Taxes with the following properties:

**GrossIncome**: the amount of the gross income

**CharitySpent**: the entire amount of the charity spent (even if not entirely accounted for).

**IncomeTax**: the amount of the income tax

**SocialTax**: the amount of the social tax

**TotalTax**: the amount of the total tax to be paid

**NetIncome**: the amount remaining for the taxpayer after the taxes

All the calculated taxpayers should be kept in-memory cache for implementing response idempotency.

The good practices of the OOP should be used.

All the calculations should be covered by unit tests.

All the parameters of the tax paying calculations are specific to jurisdiction and are subject of change.

The solution should be extensible to other tax paying calculations to be implemented later on.

Add support for docker for Windows with Linux containers.

Please over-engineer it to make it as extensible and scalable as possible.