The solution is composed of 5 projects, below is a description with the responsibility of each one of them

**IntelSyncStarter.Business**: Contains the application's business rule

* Folders
  + **DataInitialization**: File with the application's fake initialization data
  + **Implementations**: Classes with the implementation of the business rules (services)
  + **Interfaces**: Service interfaces

**IntelSyncStarter.Console**: Project responsible for calling the method to insert fake data and start the synchronization process

• In this project we have the appsettings.json file where it is possible to configure the data that will be processed per interaction (BatchSize) and the number of parallel processes (ParallelProcess)

• The Programs.cs file adds dependency injection, starts the process of creating fake data and starts the synchronization process

**IntelSyncStarter.Domain**: Groups the Project's Entities (Models) and enumerators

* Folders
  + Entities: Models used in the application
  + Enums: Enumerators used in the application, the "enumerators" were created as objects, so that it would be possible to print the strings in the project.

**IntelSyncStarter.Infrastructure**: Contains code for interacting with external objects, such as databases, files, etc.

* Implementations: Classes with the implementation of access to external resources (in this case, simulating storage in a database with two lists).
* Interfaces: Interface for implementing repositories (classes with code for external services).

**IntelSyncStarter.Test**: Project with the application's unit tests.

* **ValidateTokenTests.cs**: File with the token validation test code.