You have been called in to help with the program that data entry staff use to input data that is then uploaded into the database.

The program writes out CSV files, which are uploaded into a database table with the following columns:

* ID int not
* FieldName nvarchar(255)
* FieldData nvarchar(255)

User will typically enter a few fields at a time, and upload them.  
The program can run by “dotnet run [filename]” e.g. “dotnet run Users.txt”.

* The program has been recently enhanced to perform type checking for fields like date of birth and salary, as some staff were entering invalid values. This has been largely hailed as a success as it has prevented some staff from entering invalid values.

However, the program needs some enhancement, and you have been called in to help.

* **Bug**: some users have been reporting that they are getting database errors when uploading the files that they have created using the program - and that it is something to do with a primary key.  
  You talk to the previous developer to ask whether this is a known issue, and he said:  
  “oh yes, I found that some files had duplicate IDs in them. It uploads them to a staging table, you see, and it has got a primary key. I think it was probably a threading issue, so I put in an Interlocked.Increment to get the nextId instead of just a normal incremenet - so I think that should have sorted it!”.  
  However, it doesn’t seem to have sorted it - you’ve checked some users have definitely encountered the issuse since upgrading to the latest version, so you need to investigate further.  
  You can assume that the IDs don’t have to be unique across multiple files (because it uploads each one to its own ‘staging table’) but they do have to be unique within each file.  
  Each file is created by a single invocation of the program.  
  Users have said, though, that the program usually works fine - they have demonstrated using it to successfully create files with unique IDs - so you need to find out what conditions cause it to fail as well as coming up with a fix.
* Refactoring: some of the other technical team have wondered if there is any refactoring that the program would benefit from, to improve the general quality of the code and the layout of the program as a whole. However it’s up to you to decide if anything along these lines is necessary and if so what.
* Enhancement: some users have complained that sometimes, they run the program for quite a long time, and if their computer crashes, they lose the whole file. Can you do anything to improve this situation, so that if their computer crashes they don’t lose the whole file?
* Unit tests: some developers have said there is an upcoming project to ask for more advanced parsing, and they have wondered if some unit tests would be a good idea to protect existing functionality - and any bug fix you are able to come up with - from regressing - as well as to test any new functionality.