Digital NAO x TECMILENIO

Xóchitl Analí Cabañas Mota

NAO ID: 3319

September 18th, 2025

In-Mexico Program Backend Developer Certification

Back End in Java for Information Processing:

Backlog

Table 1

User Story	Requirements
1. As a user, I want the system to convert JSON data into CSV format so that I can analyze it in Excel or other programs.	The system must allow users to load JSON files and generate valid CSV outputs that are compatible with tools such as Excel.
2. As a user, I want the application to save the CSV file on my computer so that I can reuse it for future analysis.	The system must allow users to select a save location, generate files with user-friendly names, and ensure they are stored without accidental overwriting.
3. As a user, I want the program to validate JSON file formatting so that I can avoid errors during the conversion.	The system must validate the JSON structure before processing and notify the user with clear messages when the file is invalid, preventing the conversion until corrections are made.
4. As an end user, I want the application to have a simple interface to select a JSON file and export it to CSV without using complex commands.	The system must provide a graphical interface that allows users to browse and select JSON files, execute the export action with a button, and display confirmation or error messages in an intuitive way.
5. As a developer, I want unit tests to ensure the quality and reliability of the code.	The system must include automated unit tests that cover the JSON reader, the CSV writer, and error handling functions, ensuring that code quality is maintained throughout development.

Table 2

Requirements	**Stages**	**Time estimation**	**Deliverables**
The system must allow users to load JSON files and generate valid CSV outputs that are compatible with tools such as Excel.	2	3	A working Java class that reads JSON input and generates a valid CSV file. Sample input/output files (JSON and CSV) to demonstrate the conversion process.
The system must allow users to select a save location, generate files with user-friendly names, and ensure they are stored without accidental overwriting.	3	3	An implementation that saves the CSV file to the local computer. Screenshots or a short demo showing the save location selection and the generated file stored with a user-friendly name.
The system must validate the JSON structure before processing and notify the user with clear messages when the file is invalid, preventing the conversion until corrections are made.	2	3	Code module that validates JSON formatting before processing. Error logs or screenshots showing clear error messages when invalid JSON is provided.
The system must provide a graphical interface that allows users to browse and select JSON files, execute the export action with a button, and display confirmation or error messages in an intuitive way.	3	8	A graphical user interface (GUI) that allows users to select a JSON file and export it to CSV. Screenshots or demo video showing file selection, export button, and success/error messages.
The system must include automated unit tests that cover the JSON reader, the CSV writer, and error handling functions, ensuring that code quality is maintained throughout development.	2	3	Unit test cases covering the JSON reader, the CSV writer, and error handling. Test reports or console output showing successful execution of all unit tests.