Digital NAO × TECMILENIO

Xóchitl Analí Cabañas Mota

NAO ID: 3319

October 5th, 2025

In-Mexico Program Backend Developer Certification

Server and Database Commands:

Backlog

Table 1

User Story	**Requirements**
1. As a project leader at the Innovation Center, I want to automate the integration of the top 3 researchers' information, so that manual data collection is minimized, errors are reduced, and processes are faster.	The system must enable the automation of data integration for the institution's researchers. It should store all relevant article information in a well-structured and easily accessible database. The system must ensure full documentation and version control through GitHub to track project progress and updates.
2. As a developer, I want to create a technical report about the Google Scholar API and a GitHub repository with project documentation, so that the team has a clear reference and a central place to manage versions and share information.	The system must allow developers to include endpoints, authentication methods, query parameters, response formats, usage limits, and code examples in the report.
3. As a Java developer, I want to design a data model that represents Google Scholar author information, so that the application can store and process author data efficiently.	The system must allow the storage of multiple authors and their associated articles in memory. The system must allow the model to be compatible with the MVC architecture for integration with controller and view components.
4. As a Java developer, I want to implement the controller and view components of the MVC application, so that author data can be fetched from the Google Scholar API and displayed to the user.	The system must allow the controller to perform GET requests to the Google Scholar API and process the responses. It must handle errors and exceptions gracefully. The system must allow the view to display author search results in a readable and user-friendly format.
5. As a database developer, I want to create a database and store researcher articles from the Google Scholar API, so that all relevant data is structured, accessible, and persistent.	The system must allow developers to create a database using a suitable DBMS such as MySQL, PostgreSQL, or SQLite. It must allow the creation of a table schema including id, title, authors, publication_date, abstract, link, keywords, and cited_by.

Table 2

Requirements	**Stages**	**Time estimation**	**Deliverables**
The system must enable the automation of data integration for the institution's researchers. It should store all relevant article information in a well-structured and easily accessible database. The system must ensure full documentation and version control through GitHub to track project progress and updates.	3	20	• A fully functional automated system that integrates and manages data for the institution's researchers. • A structured and easily searchable database containing all relevant article information. • Complete project documentation and version control available in GitHub. • A thoroughly tested solution that ensures proper operation and supports future data queries without requiring manual updates.
The system must allow developers to include endpoints, authentication methods, query parameters, response formats, usage limits, and code examples in the report.	1	2	• Technical report summarizing the Google Scholar API, including endpoints, authentication, query parameters, response formats, usage limits, and code examples. • GitHub repository with a README.md describing project purpose, key functionalities, and relevance. • Configured repository permissions allowing the Digital NAO team to access it.
The system must allow the storage of multiple authors and their associated articles in memory. The system must allow the model to be compatible with the MVC architecture for integration with controller and view components.	3	12	• Java class(es) representing the data model for Google Scholar author information. • Documentation explaining the model structure and how it integrates with the MVC architecture. • Unit tests verifying that the model can store multiple authors and articles correctly.
The system must allow the controller to perform GET requests to the Google Scholar API and process the responses. It must handle errors and exceptions gracefully. The system must allow the view to display author search results in a readable and user-friendly format.	2	8	• Controller class in Java that performs GET requests to the Google Scholar API and processes the responses. • View class that displays author search results clearly and user-friendly. • Error handling mechanism for API requests, with documentation on how errors are managed. • Integrated MVC application tested end-to-end (API → Controller → View) with sample author searches.

The system must allow developers to create a database using a suitable DBMS such as MySQL, PostgreSQL, or SQLite. It must allow the creation of a table schema including id, title, authors, publication_date, abstract, link, keywords, and cited_by.	3	8	• Database created in a chosen DBMS (MySQL, PostgreSQL, SQLite, etc.) with the defined schema. • Table containing fields: id, title, authors, publication_date, abstract, link, keywords, cited_by. • Integrated dataset for 2 researchers and 3 articles per researcher. • Documentation describing database structure, data integration process, and error handling. • Updated GitHub repository with Sprint 3 deliverables and access permissions for the Digital NAO team.