# ETL Case:

Our learning team would like to consume data from a Learning Management System. They would like to see which users enrolled into which courses.

Using the following LMS documentation <https://canvas.instructure.com/doc/api/>, can you please explain how you could retrieve course data for users?

What are the development steps for the ETL?

What other data could be interesting to retrieve and why?

# Data Visualization Case:

A recruitment leader feels like they could be more efficient in their recruitment process and would like to understand what step to focus-on first. Create a dashboard that would help her answer this question that can be used in the future to optimize their operations.

Dataset: “case recruitment dataset.xlsx”

1. Authentication: Obtain an access token or API key to authenticate your requests to the Canvas API. Refer to the Canvas API documentation or contact your Canvas administrator to learn how to generate an access token.
2. API Endpoint: Determine the appropriate API endpoint to retrieve course data for users. According to the Canvas API documentation, you can use the "List enrollments for a user" endpoint (**GET /api/v1/users/:user\_id/enrollments**) to retrieve the enrollments of a specific user.
3. User ID: Identify the ID of the user for whom you want to retrieve course data. You will need the user's ID to make the API request.
4. Make the API Request: Use your preferred programming language or tool to make an HTTP GET request to the API endpoint mentioned above, providing the user ID as a parameter in the URL. Include the necessary headers and authentication information (access token or API key) in your request.
5. Process the Response: Handle the response from the API and extract the relevant course data for the user. The response may contain information about the courses in which the user is enrolled, such as course IDs, names, and other details.
6. Repeat for Multiple Users: If you need to retrieve course data for multiple users, you can loop through the user IDs and make separate API requests for each user.
7. Requirements Gathering: Understand the business requirements and data sources. Identify the data to be extracted, transformed, and loaded, as well as any specific data quality or integration requirements.
8. Data Source Analysis: Analyze the structure, format, and quality of the source data. Identify any data inconsistencies, missing values, or data quality issues that need to be addressed during the transformation process.
9. Extraction: Extract the data from the source systems or files. This may involve querying databases, accessing APIs, reading flat files, or any other method of retrieving data.
10. Data Cleaning and Validation: Cleanse the extracted data by handling missing values, removing duplicates, and performing data validation checks. This step ensures that the data is accurate, consistent, and conforms to the defined rules and standards.
11. Transformation: Apply transformations to the extracted data to convert it into the desired format for analysis or loading into the target system. This may involve data mapping, aggregation, filtering, joining, or any other data manipulation operations.
12. Data Loading: Load the transformed data into the target system, such as a database, data warehouse, or analytics platform. The loading process may include creating tables, defining schemas, and inserting or updating records.
13. Data Quality Assurance: Perform data quality checks and validations on the loaded data to ensure its accuracy and integrity. This step involves verifying data completeness, consistency, and conformity to the expected standards.
14. Error Handling and Logging: Implement mechanisms to handle errors, exceptions, and data inconsistencies encountered during the ETL process. Logging and error tracking help in identifying and resolving issues efficiently.
15. Scheduling and Automation: Set up a scheduling mechanism to run the ETL process at regular intervals or as per the defined schedule. Automation ensures that the data is kept up to date and the ETL pipeline operates reliably.
16. Testing and Validation: Conduct thorough testing to validate the ETL pipeline's functionality, data accuracy, and performance. Perform end-to-end testing, unit testing, and integration testing to identify and rectify any issues.
17. Documentation: Document the ETL process, including data flow diagrams, transformation rules, and technical specifications. Documentation helps in understanding and maintaining the ETL pipeline in the future.
18. Monitoring and Maintenance: Set up monitoring mechanisms to track the performance, data quality, and health of the ETL pipeline. Regular maintenance and updates may be required to accommodate changes in data sources, transformation rules, or target systems.