 

**Project report:** Database application for invoice management



**Project Id :** 4

**Client :** OCP's Purchasing & Contract Management Unit

**Supervised by :** PR Karima ECHIHABI

**By :** ABERCHA Zineb

BOUDHAINE Ilyas

KEMMOUNE Anass

MOUIMI Oumaima

Table of Contents

INTRODUCTION

[PROJECT DESCRIPTION](#_Toc133166008)

[Step 1 : General Structure of the project](#_Toc133166009)

[Step 2 : User interface](#_Toc133166009)

[Step 3 : Data processing](#_Toc133166009)

[Step 4 : Data pre-processing](#_Toc133166010)

[Step 5 : Data visualization](#_Toc133166011)

[TEAM RESPONSIBILITIES](#_Toc133166012)

[CHALLENGES](#_Toc133166012)

[LESSONS LEARNED](#_Toc133166012)

# 

# INTRODUCTION

The primary goal of this project is to develop a robust and efficient Database Application that simplifies invoice management by automating data cleaning, extraction, and analysis from Excel sheets. The client has expressed a strong desire to optimize their invoicing processes and enhance overall operational efficiency. To achieve this objective, we have collectively decided to break down the project into three essential phases: Data pre-processing, Data processing, and Data visualizations.

The construction of this database application is one of the ways we want to give the client a number of advantages. One important benefit is the increased productivity brought about by the automation of data processing processes, which lowers the need for manual labor and speeds up the overall invoice management workflow. We also remove inaccuracies and deliver accurate information for decision-making by assuring data accuracy and integrity through complete data cleaning and standardization. As a result, the client is better able to make wise business decisions and uphold high standards of data integrity.

# PROJECT DESCRIPTION:

Step 1: General structure of the project

We have chosen to adopt the Model-View-Controller (MVC) architectural pattern for our invoice management application due to its numerous advantages in software development projects. It offers benefits such as code reusability, faster development process, and enhanced user experience. The MVC pattern is particularly suitable for our project as it ensures that modifications can be made without affecting the entire model, providing a robust and scalable solution.

In the context of our project, here is an overview of the main components and their respective roles within the MVC structure:

**Controller**: The controller acts as an interface between the Model and View components. It incorporates necessary libraries such as the excel Reader and vendor1. The excel Reader library enables the application to read Excel files, automate data extraction, processing, and analysis from the provided Excel sheets. The vendor1 library contains general styling for the user interface. The controller also includes files such as upload.php, which handles the data upload and ensures that the sheets are in the correct xlsx format. Additionally, the modifyCsv.php file modifies the first row of the sheet data to unify the column names using a hash map.

**Model**: The model component encompasses all the data-related logic within the application. It includes files such as tableInsert2.php and tableInsert1.php, which handle the insertion of the Excel tables into the database. The dbConfig.php file is responsible for establishing the SQL connection using mysqli. The finalExcel.sql file is used to create the final CSV file, and it utilizes the functions in the executeSqlFile.php file. The invoiced-ddl.sql file contains all the necessary queries to create the database, tables, and indexes and the views. Furthermore, the model component includes a file to clear the database after its usage and before re-uploading the sheets.

**View**: The view component handles the user interface logic of the application. It consists of a folder named "assets" containing subfolders for CSS, images, and JS files. The HTML and PHP files for login and sign up are included in this component. Additionally, the "charts" folder contains PHP files required for displaying and visualizing the graphs as specified by the project requirements.

Other directories within the application include "Exports," which stores the final CSV file as requested by the client for conversion to an Excel sheet, and "Uploads," which contains all the uploaded sheets that were previously downloaded using the upload.php file.

Step 2: User interface

In this step, we focus on providing a user-friendly interface that meets the project criteria and client requirements. We have taken into consideration the user registration and login processes, as well as the overall design and functionality of the interface.

**User Registration and Login:**

Users register once using a user-friendly interface that meets the project criteria and client requirements. The registration form includes required fields and validates the format of the email address and matching passwords. Upon successful registration, users can log in to the application. The login form verifies the user's credentials, ensuring secure access to the application. A User class is implemented to handle user-related operations and facilitate the login process.

**User Interface and View:**

The user interface contains all the necessary files for an intuitive and user-friendly experience. The interface includes forms that respect the requirements and validate the input data using JavaScript functions. Required fields are clearly indicated, and the format of the email address is checked.

**Database Interaction (database.php):**

Upon successful login, users are redirected to the database.php file.The database.php file handles interactions with the database.Users can upload two sheets through this interface.The uploaded sheets are processed and stored in the database.Users can view the uploaded data and download the final Excel file, which may contain a merged or transformed version of the original sheets.

**Dashboard Display (dashboard.php):**

The dashboard displays graphs representing invoice numbers per type d'achat, invoice numbers per echeance, and invoice numbers per entite.The graphs provide users with visual insights into the data and facilitate data analysis.

Step 3: Data pre-processing

In the data pre-processing step, we will perform these tasks :

**File Upload:** The upload.php file is responsible for handling the file upload and ensuring that the uploaded files are in the correct XLSX format. This file receives the 2 uploaded files from the user and checks if they have the expected file extension (.xlsx). It also performs additional validations, such as file size restrictions to ensure the uploaded files meet the requirements.

**Modify First Row:** Once the files are uploaded and confirmed to be in the correct XLSX format, the modifyCsv.php file is used to modify the first row of data in each sheet file. The purpose is to unify the names of the columns in the two sheets, likely to ensure consistency and compatibility with the database schema. This is achieved by using a hash that defines the corresponding column names for each sheet.

**Excel Reader Library:** The PHP files, tableInsert2.php and tableInsert1.php, utilize the Excel Reader library. This library provides functionality to read and parse the modified XLSX files, extracting the data from each sheet.

**Insert into Database:** The tableInsert2.php and tableInsert1.php files use the Excel Reader library to read the data from the modified XLSX files and prepare it for insertion into the database.

**Merge and Insert:** once the sheets are modified and merged, the PHP scripts "tableInsert2.php" and "tableInsert1.php" handle the insertion of the data into the database.

Step 4: Data processing

In this phase of the project, we will utilize embedded SQL queries to extract and analyze specific information from the database, focusing on the number of invoices per entity, the number of invoices per due date (échéance), and the number of invoices per ‘Type d’achat PO’.

To determine the number of invoices per entity, we will execute SQL queries that group the data based on the entity attribute. By applying the COUNT() function, we can calculate the total number of invoices associated with each entity. This analysis will provide a clear understanding of the invoice distribution across different entities within the organization.

Similarly, to ascertain the number of invoices per due date (échéance), we will employ embedded SQL queries that group the data based on the échéance attribute. By utilizing the COUNT() function, we can determine the count of invoices associated with each due date. This analysis will help identify the distribution of invoices based on their payment deadlines, allowing for better prioritization and planning of payment activities.

Furthermore, we will examine the number of invoices per ‘Type d’achat PO’. This analysis involves using embedded SQL queries to group the data based on the category attribute, which can represent different invoice types or classifications. By applying the COUNT() function, we can determine the count of invoices associated with each ‘Type d’achat PO’. This analysis will provide insights into the distribution of invoices across various ‘Type d’achat PO’, allowing for a better understanding of the invoice composition and potential patterns or trends.

In addition to the analysis of the number of invoices per entity, we can further refine the insights by filtering the data based on the due date (échéance). This will allow us to determine the number of invoices per entity, specifically those that are filtered by a specific échéance.

To calculate the number of invoices per entity filtered by échéance, we will utilize embedded SQL queries. This analysis will provide a detailed breakdown of the invoices associated with each entity for a specific échéance, enabling the identification of entities that may require immediate attention or prioritization.

Similarly, we can extend this analysis to determine the number of invoices per ‘Type d’achat PO’, again filtered by échéance. By incorporating the échéance filter in the SQL queries, we can narrow down the data to a specific time period. The filtered data will be grouped by the ‘Type d’achat PO’ attribute, and the COUNT() function will be used to calculate the total number of invoices per ‘Type d’achat PO’. This analysis will help identify the distribution of invoices across different categories within a specific échéance period, allowing for a focused understanding of invoice composition and potential trends.

The use of embedded SQL queries ensures seamless integration within the Database Application, enabling efficient retrieval and aggregation of data. By extracting and analyzing the number of invoices per entity, per due date, and per ‘Type d’achat PO’, the client will gain valuable insights into their invoicing processes. These insights can aid in identifying entities with high invoice volumes, detecting any concentration of invoices around specific due dates, and understanding the distribution of invoices across different ‘Type d’achat PO’. Such information will facilitate the development of effective action plans to efficiently process and clear the invoices with suppliers, resulting in improved cash flow management and overall operational efficiency.

# Step 5: Data visualization

This step of the project involves visualizing the results obtained from the second step using Chart.js, a popular JavaScript library for creating interactive and visually appealing charts and graphs. We will leverage the capabilities of Chart.js to present the analyzed data in a meaningful and informative manner.

For the first graph depicting the number of invoices per entity, we will utilize a bar chart. Each bar on the chart will represent an entity, while the height of the bar will correspond to the total number of invoices associated with that entity. This visualization will allow for a clear comparison between entities, highlighting those with a higher volume of invoices.

Similarly, for the graph illustrating the number of invoices per ‘Type d’achat PO’, a disc chart can be utilized. The disc chart will divide the visualization into sectors, with each sector representing a different ‘Type d’achat PO’ of invoices. The size or angle of each sector will reflect the proportionate number of invoices attributed to that particular ‘Type d’achat PO’. This type of visualization allows for a quick assessment of the distribution of invoices among different ‘Type d’achat PO’.

For the graph that displays the number of invoices per échéance, a donut chart can effectively showcase the distribution of invoices across different due dates. Each segment of the donut will represent a specific échéance, and the size of the segment will correspond to the proportionate number of invoices associated with that due date. The donut chart provides a clear visual representation of the relative significance of each échéance in the dataset.

The dynamic nature of the charts allows users to interact with the data, gaining insights into the invoice distribution and trends for different échéances. They can easily compare the performance of entities or observe changes in the distribution of invoices across categories over different échéances.

By incorporating dynamic charts for each échéance, we enhance the visual representation of the data, making it more accessible and actionable. Users can quickly identify patterns and make informed decisions based on the visualized information.

Overall, Chart.js will enhance the project's deliverables by providing dynamic and interactive visualizations that make the extracted information easily digestible and actionable for the client.

# TEAM RESPONSIBILITIES:

**Anass Kemmoune:** Team Coordination – Communication with the client – Requirements gathering – Database design - Data insertion and preprocessing using PHP & mySQLI - Data Visualization using Chart.js- Stored Procedures – Application code Structure and modeling – Data processing and exportation- Debugging – User Integration to the DB- Frontend Development

**Ilyas Boudhaine:** Team Coordination - Requirements gathering - Database design - Data insertion and preprocessing using PHP & mySQLI - Multiple SQL script files (DDL – DCL -Indexes …)- Table Displaying - Data Visualization using Chart.js - Stored Procedures – Performance evaluation and application testing – data processing - Debugging.

**Oumaima Mouimi:** Database Design - BNCF normalization Minimal Cover Set- schemas -Log files – SQL Scripts - Data Generation scripts using Python & Excel – Login and register - Report writing – Frontend Development

**Zineb Abercha:** Database Design - BNCF normalization -Minimal Cover Set - schemas - Multiple SQL script files (DDL-DML – triggers -Indexes …) data modification – Transactions – Login and register – Dashboard design- Report writing – Frontend Development

# CHALLENGES:

The invoice management project faced several challenges throughout its development, considering both the project requirements and the time constraints. Here are some of the challenges encountered and how they were addressed:

1. Data Cleaning and Extraction: The project involved extracting data from Excel-based invoices, which can be challenging due to variations in formatting and data inconsistencies. To address this, robust data pre-processing techniques were implemented, including data validation, normalization, and error handling. Regular expressions and data parsing algorithms were utilized to ensure accurate data extraction.

1. Importing Data from Similar Database: Importing additional information from a similar database introduces the challenge of data retrieval and integration. Using the "PK Identifiant GED" as a key to import data such as "Blocage," "catégorie," "chef de projet," "Entité," "Entité Site," and "Entité G" requires establishing a connection to the database, querying the relevant data, and integrating it into the invoice dataset accurately.

1. Handling Performance and Efficiency: Processing a large volume of invoices and performing data transformations within a reasonable timeframe requires efficient coding practices. Optimizing the code for performance and considering factors such as processing speed and memory usage are important challenges to address.
2. Data Consistency and Accuracy: Ensuring data consistency and accuracy throughout the processing phase is crucial. Validating the imported data, cross-checking against existing records, and performing data quality checks can help identify and resolve any inconsistencies or discrepancies.
3. Choosing the Right Visualizations: Selecting the most appropriate visualization techniques to effectively represent the data and communicate the desired insights can be challenging. It requires understanding the nature of the data, the

relationships between variables, and the audience's needs. Choosing the wrong visualization can lead to misinterpretation or confusion.

# LESSONS LEARNED:

The project of invoice management offers several valuable lessons that can be learned:

* Effective Planning and Requirement Gathering: Proper planning and thorough requirement gathering are crucial to ensure the project's success. It is essential to have a clear understanding of the client's needs, goals, and expectations from the outset. This helps in setting realistic project timelines, allocating resources effectively, and delivering a solution that aligns with the client's requirements.
* Data Quality and Preprocessing: The importance of data quality and preprocessing cannot be overstated. Cleaning, validating, and transforming the data before analysis is critical to obtain accurate and reliable results. Investing time and effort in data cleaning and ensuring data consistency at the early stages of the project can save significant time and effort later on.
* Effective Database Design: A well-designed database is the foundation of any successful database application. Ensuring proper normalization, appropriate indexing, and efficient query design can enhance performance, scalability, and data integrity. Careful consideration of the database schema and optimization techniques can significantly impact the overall effectiveness of the system.
* Integration and Data Synchronization: Integrating data from multiple sources and ensuring data synchronization is an essential aspect of many database applications. Understanding the complexities of data integration, establishing reliable data connections, and implementing robust synchronization mechanisms are critical to maintaining data accuracy and consistency.
* Time Management and Deadline Adherence: Time management is a crucial skill, especially when working on projects with strict deadlines. The ability to manage time effectively, prioritize tasks, and adhere to project timelines is essential for successful project completion. This project highlights the importance of allocating time appropriately, setting realistic deadlines, and actively managing tasks to meet project milestones.
* Communication and Collaboration: Effective communication and collaboration among team members and stakeholders are essential for project success. Regular communication, status updates, and feedback sessions help in ensuring everyone is aligned, addressing challenges promptly, and making necessary adjustments along the way. Open and transparent communication channels facilitate a smooth project workflow.
* Flexibility and Adaptability: Projects often encounter unforeseen challenges or changes in requirements. The ability to adapt and be flexible in response to these changes is critical. Being open to adjusting the project plan, accommodating new requirements, and proactively finding solutions to emerging issues demonstrate agility and resilience.
* Documentation and Knowledge Sharing: Documenting the project processes, decisions, and lessons learned is invaluable for future reference and knowledge sharing. Proper documentation ensures continuity, facilitates troubleshooting, and helps in onboarding new team members or maintaining the system in the long run.

By reflecting on these lessons learned, future projects can benefit from improved planning, enhanced data management practices, efficient time management, effective communication, and a culture of continuous learning and improvement.