Library Management System

By: Joel Joshy

Table of Contents

Project Plan

Introduction

System-As-Is

The current library management system at Cal Poly Pomona manages students, items (books/documentaries), creators (authors/producers) and loans through a manual process. Currently, it is very tedious, slow, and can be hard to track leading to potential errors or loss of data since it is done manually. It can be very time consuming to follow and update the details of each entity within the system.

System-To-Be

The system-to-be will improve the current library management system and give library staff better control of the multiple entities they are responsible for maintaining. The new system will work to automate multiple operations with a click of a button, reducing both operational costs and error. It will be easier to manage data of students, loans, items, and creators with this new system.

Organization of the Project

Project Manager	Analyst	Programmer	Tester
Joel	Joel	Joel	Joel

Methods and Techniques

Used classification techniques for the design of the project revolving around identifying meaningful phrases, omitting meaningless or unnecessary phrases within the project's scope, and classifying them as potential classes, attributes, and methods. A requirement specification in natural language was given as well as use-case diagrams. From there, the nouns and verbs were analyzed to identify entities, attributes and operations. Additionally, words outside the system's scope were identified and omitted from the design, and synonyms were also identified to prevent unnecessary duplication. Finally, the decision was made of which nouns represent classes in the system, which nouns represent attributes of the classes, and which verbs represent potential methods. Using all of this analysis, the domain and object models for the system were designed.

Requirements Specification

Functional Requirements

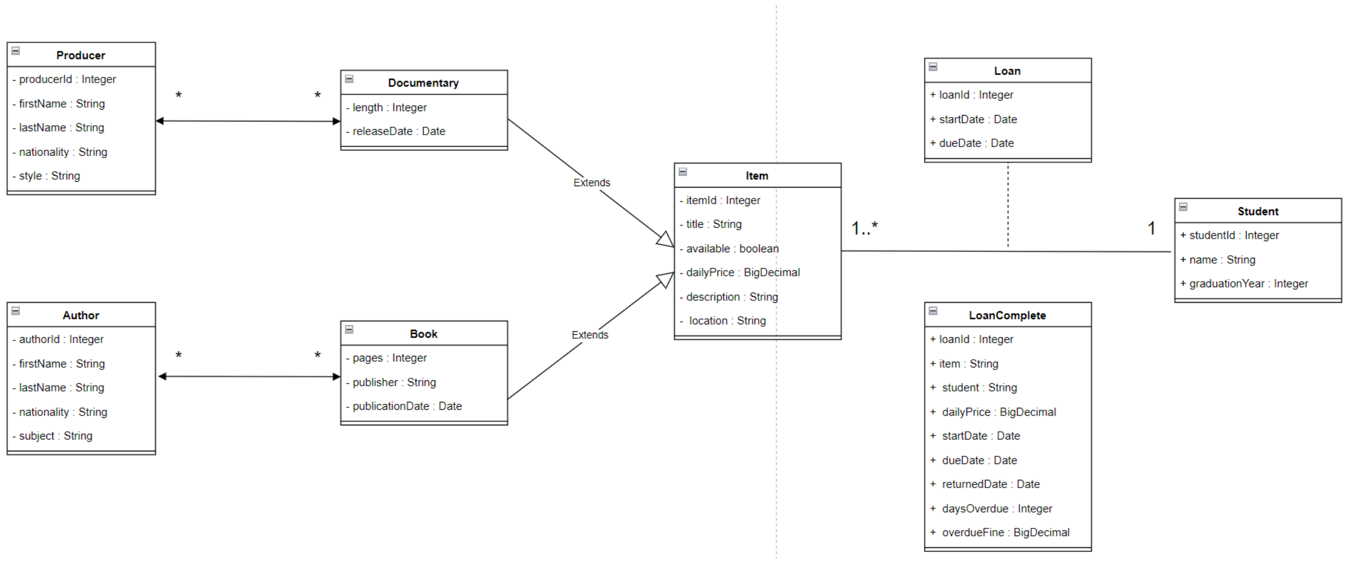
To improve the overall system, requirements must be met. The system was able to successfully meet the following requirements for the following entities:

- Student:
    - Create Student
    - Search/Update/Delete Student using Student Id
    - Display all Students
    - Let User know if there are empty fields and other requirements for fields
- Creators (Authors/Producers):
    - Create Creator
    - Search/Update/Delete Student using Creator Id
    - Displays all Creators
    - Displays the Items of a Creator once the Creator is searched
    - Lets User know if there are empty fields and other requirements for fields
- Items (Books/Documentaries):
    - Create Item
    - Search/Update/Delete Student using Item Id
    - Displays all Items
    - Displays a list of Authors to select from
    - When an Item is searched, the list of Creators is sorted so that it displays and highlights the Creators of the item at the top
    - Lets User know if there are empty fields and other requirements for fields
- Loan:
    - Generate loans consisting of student info, item info, loan start date, loan due date, loan daily price, status of loan (displays "Not Overdue" if the loan is not overdue or else it will display "Overdue"). These loans that are in progress will be saved in the loan_incomplete table
    - Calculate loan price including any potential fines due to late return. The fines will be the item's daily price plus 10% per day after the loan passes the due date.
    - Search Loan using Loan Id, Item Id, or Student Id
    - Show list of all loans that are overdue using a filter
    - Delete/Update Loan using Loan Id
    - When a loan is complete it will be saved in the loan_complete table which is a stand alone table because this table will not have any relations with the other tables since this table contains completed loans that are not used in the system anymore and we need to maintain accurate historic data for generating the revenue report. In other words, we do not want to alter the data for the completed loans

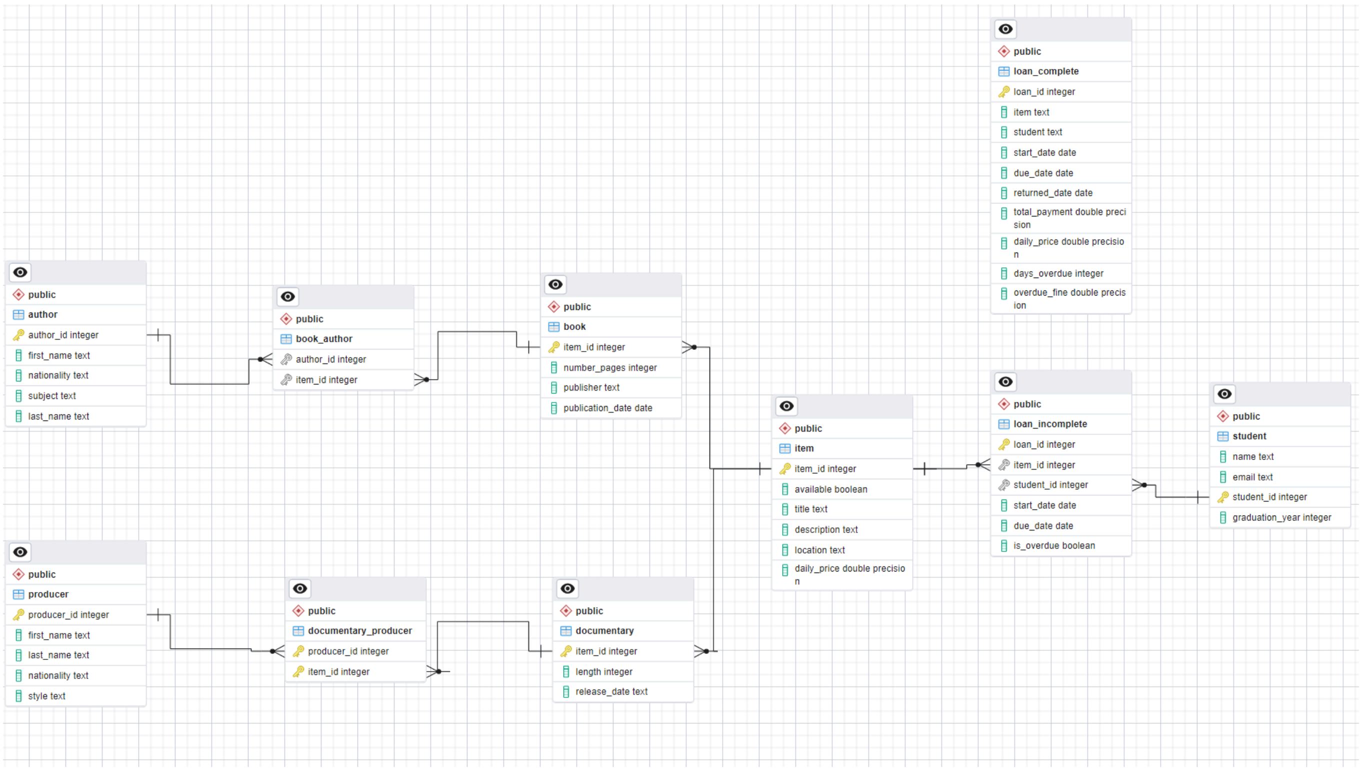
- Generates a receipt of a loan when the loan is generated
- Generate a loan report that has loan information filtered by loan id.
- Lets users know if Item is currently being used by another student, if chosen loan due date is more than six months later, if chosen loan due date is before the current day, if there are empty fields, and other requirements for fields.
- When the application opens, the system compares the current date to each loan's due date and if the due date of the loan has passed the current date, then the loan's isOverdue attribute is updated to overdue

Design

- Domain Model (Class Diagram)



- Data Logical Model



Implementation

Tools Used:

- Used JavaFX for programming the desktop frontend application
- Hibernate for ORM
- PostgreSQL for the Database Management System

GitHub Repository:

https://github.com/Joel5212/LibraryManagementSystem

Project Demonstration:

https://www.youtube.com/watch?v=s6Dq672TedI