CS348 - Introduction to Database Management

University of Waterloo

Cheriton School of Computer Science

Project: My Test Bank

Authors:
Haoqi Shi (ID: 20603439)
Jiaqi Wang (ID: 20657135)
Xueyao Yu (ID: 20622323)
Zhengmin Zhang (ID: 20602385)

Date: March 11, 2020

Contents

1 Project Description

This project named **My Test Bank** is designed to be a cross-platform application that provides featured test bank look-up services to registered users. Upon registration, users will be able to look up past tests and practice problem sets for all kinds of courses provided at the University of Waterloo through our application. We also provide features such as shuffling the order of the problems and the options within a questions. We add error book as a new functionality. When a user is doing a quiz, the question he or she answers wrongly will be automatically added to an error book. Later on the user can start a new quiz only on the error book questions, and once he or she gets a question correctly, the question will be removed from the error book. The error book is designed for more targeted practise and aims to improve the efficiency and save user's time.

2 Project details

2.1 Target Users

The users of the application could be students who may want to access some practice questions or testing resources for their academic courses as well as administrators of the application to manage(upload/update/delete) the resources.

2.2 Key Features

The application interacts with users in different ways during each process. We have implemented all of them.

2.2.1 Overview

- Unique username and email pair.
- Unique practice problem set name.
- Allow users to contribute to the database and benefit all users by creating new practice problem sets.
- Questions and the order of the choices in each question are shuffled.
- Provide an overview of all the available practice problem sets to the users.
- Allow users to work on the problem and see the solution online.
- Notify users if a problem set is empty/invalid.
- Allow users to work on the problem they made mistakes before.

2.2.2 Feature Explanations

- New User Registration
  During the process of registration, the user enters their desired username, password, email address then clicks on the ”Create” button to finish registration. Then the corresponding user credential record will be stored to our backend database for future lookup and matching.

- User Login
  User enters the username and the corresponding password, and click the ”login” button. Then the system will perform a database query to check for the validity of the entered user credential.

- Add Books to the Database
  User enters the course name, course number, name of the practice problem set, and the problem content, including both the question and the answer, then click ”create”.

- Add Chapters to the Database
  After creating a book, User can add chapters to the current book. User enters the chapter name and chapter number, then click ”create”.

- Add Problems to the Database
  After creating a chapter, User can add problems to the current book. User enters the question description, options and the solution, then click ”create”.

- Work on Practice Problem Sets
  By clicking ”Start Quiz”, user could work on randomly shuffled practice questions with randomly shuffled choices. User can click on a certain choice and ”Next” to move to the next question, or instead, they could also click ”Show Solution” to see the solution for that question. Now our UI supports user to select and submit an option before seeing the correct answer. Unless the user clicks on ”Show Solution”, he or she will not be told the correct answer.

- Work on Mistaken Problem Sets
  By clicking the link ”error set” under each chapter, user can start a quiz on the problems he or she did wrongly before. Same as the normal quiz, order of the questions and options will be shuffled. After correctly answered a question at the first time, this question will be removed from the error set.

2.3 Dataset

Initially, We will create our own dataset by sorting out the public test bank resources and upload to GCP. After having the basic functionality and being able to provide this

application to the users, we could also populate our dataset by accepting the practice problem sets entered by registered users. We have gathered the test bank for some common courses in UWaterloo(e.g., ECON101, HRM200, etc.). We will reformat the text file into CSV and write a python script to load these books(with chapters, questions, options and the solution) into the database as public books, i.e., every registered user can access them.

2.4 Application Platform and User Interface

We will use MySQL as the platform via with Flask(Python). Users can interact with the application through web interface. Up until now, we are working on re-implementing the frontend with React, so in the near future we will be able to provide an mobile version of our application.

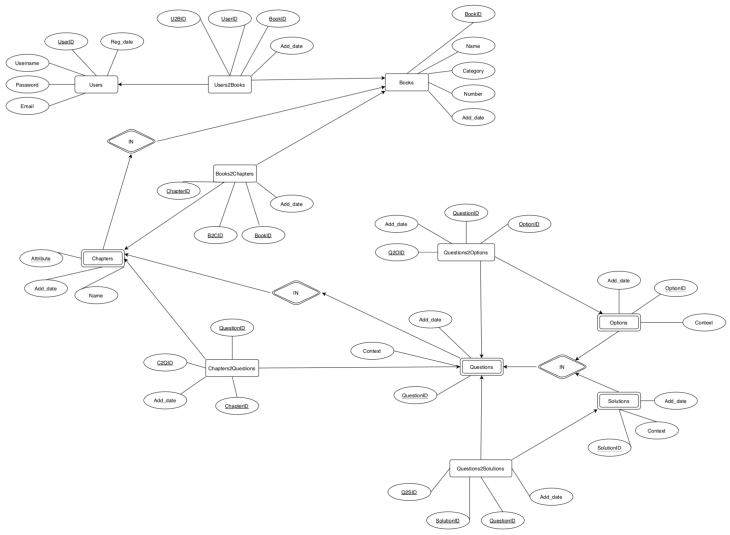
2.5 Design Database Schema

2.5.1 Data Assumptions

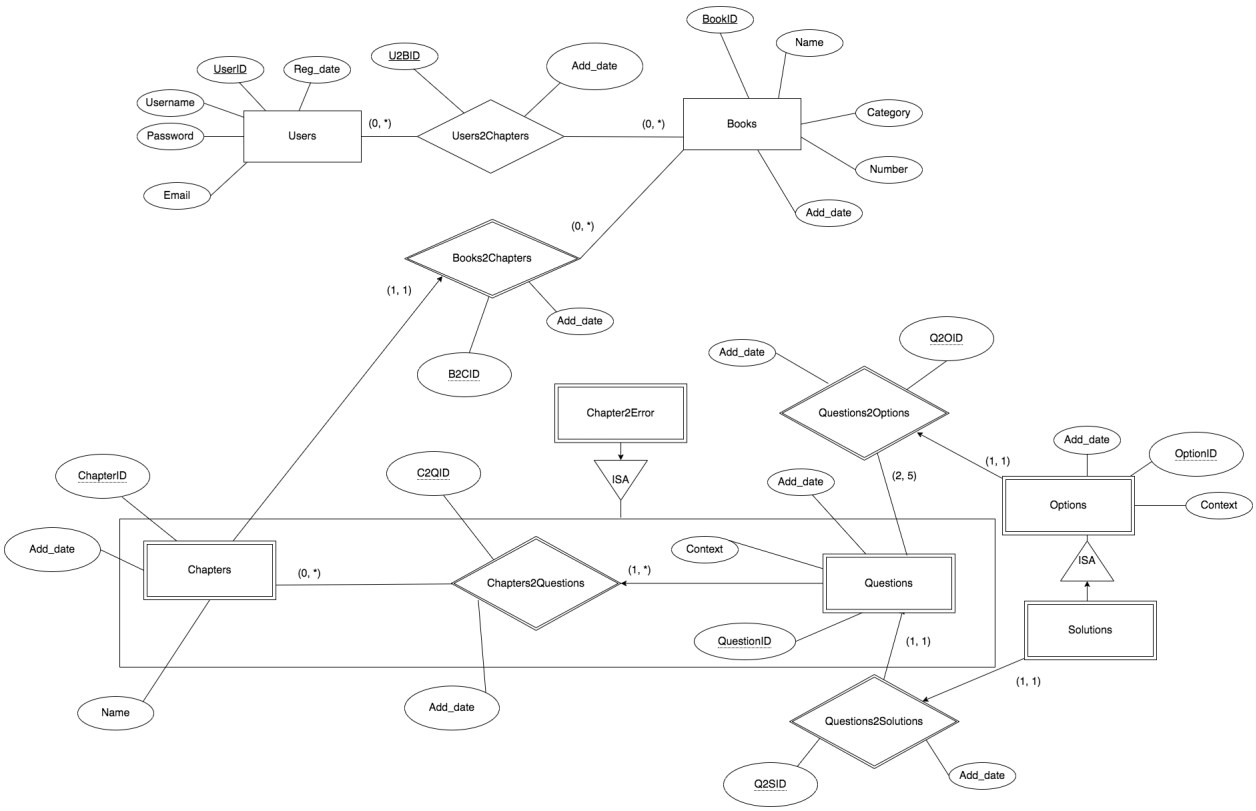
- Username and email are unique for each user.
- Password has to be longer than 6 characters including at least a letter and a number.
- Category and number pair are unique for each book, i.e., if book A's category is "CS" and number is "348", then "CS"+"350" and "SE"+"348" are valid while another "CS"+"348" is not allowed.
- A quiz can be started only when the selected chapter contains at least one question.

2.5.2 ER-Diagram

- Milestone 1



- Milestone 2



Comments:

– User ID is the primary index of User, and (Password, Email) is the secondary index of User. This will support the future feature of recovering an account when user forgets his or her login username.

  - A User can have 0 or more books, and a Book can belong to 0 or more users. There are public or shared books among users.
  - The question in the error book of a chapter must be one of the all questions of that chapter. Thus Chapter-to-Error is an aggregated relation between Chapter and Question.

2.5.3 Relational Data Model

- Users(id, username, pw, email, reg_date)
  - username is a unique key
  - email is a unique key
  - (email, pw) is a secondary index for finding the username back.
- Books(id, category, number, name, add_date)
  - (category, number, name) is a unique key tuple. It happens that a course has more than one practise book. Two books of the same name in a course will be treated duplicated and are not allowed.
- Chapters(id, name, add_date)
- Questions(id, context, add_date)
- Options(id, context, add_date)
- Users_2_Books(id, user_id, book_id, add_date)
  User should be able to select the book he or she wants to practise on. So we need to record this relation and not allow any dangling user or book. That is why we need the foreign key constraint on UserID and BookID.
  - user_id is a foreign key pointing to id in Users
  - book_id is a foreign key pointing to id in Books
- Books_2_Chapters(id, book_id, chapter_id, add_date)
  User should be able to select a chapter from a book. User should not be able to select any chapter out of this book. So we need to record this relation and not allow any dangling book or chapter. That is why we need the foreign key constraint on ChapterID and BookID.
  - book_id is a foreign key pointing to id in Books
  - chapter_id is a foreign key pointing to id in Chapters

- Chapters_2_Questions(id, chapter_id, question_id, add_date)
  When user starts a quiz on a chapter, questions of that chapter should be loaded. So we need to record this relation and not allow any dangling question or chapter. That is why we need the foreign key constraint on ChapterID and QuestionID.
  - chapter_id is a foreign key pointing to id in Chapters
  - question_id is a foreign key pointing to id in Questions
- Questions_2_Options(id, question_id, option_id, add_date)
  A question has several options, and we need to record this relation. Our app supports user to add more options to a question that he or she finds confusing, so instead of recording all the options and the question together, we record the relation between one option and its corresponding question at a time to provide more flexibility.
  - question_id is a foreign key pointing to id in Questions
  - option_id is a foreign key pointing to id in Options
- Questions_2_Solutions(id, question_id, solution_id, add_date)
  A question should have a unique solution, and the solution will not change. That is why (question_id, solution_id) is a unique tuple. Also, since a question can only have one solution, question_id is a unique key.
  - question_id is a foreign key pointing to id in Questions
  - solution_id is a foreign key pointing to id in Options
  - question_id is a unique key
  - (question_id, solution_id) is a unique tuple
- Chapters_2_Errors(id, chapter_id, solution_id, add_date)
  An error is a question, and an error-chapter relation is a question-chapter relation, i.e., an error of a chapter is always a question of that chapter. So we have foreign key constraint to id pointing to the id of Chapters_2_Questions.
  - id is a foreign key pointing to id in Chapters_2_Questions
  - chapter_id is a foreign key pointing to id in Chapters
  - error_id is a foreign key pointing to id in Questions