IT Learning Management System: A Digital Test-Taking System to Improve Conceptual IT Knowledge

CS4991 Capstone Report, 2022

Anthony Tiancheng Sun
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
Author Contact Info: anthonysun2007@gmail.com

Abstract

The IT development team of a service management company based in Herndon, Virginia needed an internal program that could be used to test and track a team member's knowledge of key concepts related to software development and the various programming tools utilized by the team. In order to address this issue, the team's tech lead and I developed a web-based IT Learning Management System (IT LMS) as a platform for team members to take multiple choice tests related to conceptual knowledge of software development principles and tools, while also tracking their test results over time.

We built the system in the Appian low-code development platform, with user data and test banks being stored through Appian's MySQL Cloud database. The program also allowed team members to identify knowledge gaps in relation to the software development tools used by the company, promoting growth and improvement as software developers. The system could be expanded to give better feedback to users based on their results, such as identifying areas for improvement.

1. Introduction

Eleanor Roosevelt once said that "unused ability, like unused muscles, will atrophy". Roosevelt's words rang true to the tech lead of a Herndon service management company when he observed that his software development team's abilities had deteriorated after a long period of focusing on project maintenance. The team had not engaged in challenging software development for some time, resulting in the erosion of their knowledge base.

In order to reverse this phenomenon, the tech lead needed a way for his team to consistently engage and challenge their software development skills, even if their current projects had entered a more relaxed phase. His solution was an IT LMS that would allow team members to take tests related to software development concepts, enabling them to constantly engage and grow their computer science knowledge base.

2. Related Works

One relevant work is Google's Web Fundamentals repository published on their Google Developers website[1]. The repository contains industry best practices for designing web applications in order to enhance usability and user experience. The repository recommends keeping primary tasks at the front of the design and leaving secondary tasks underneath in order to improve the flow of the application. I used this philosophy in the design of the IT LMS, where I made sure the main page put the important test-taking function front and center while leaving the secondary feature of viewing user metrics further down the webpage.

Another relevant work was Microsoft's Database Design Basics article published on their Microsoft Support website[2]. The article focuses on providing developers with insights on how to build robust and efficient databases that power applications for the web. Its recommendations on streamlining database designs by identifying and eliminating

unnecessary database objects inspired me to make similar simplifications to the IT LMS's database.

3. Process Design

The system's design process can be separated into three phases: Setting up the system's database, creating the home page, and designing the test-taking interface.

3.1 Database Design

The first step I took in developing the system was designing the backend database that would power the IT LMS. The database can be conceptually divided into two halves that are linked together with a test object: one half is dedicated to storing the test's question and answer choice data, and another half is dedicated to tracking if a user has taken the test and the answer choices they selected.

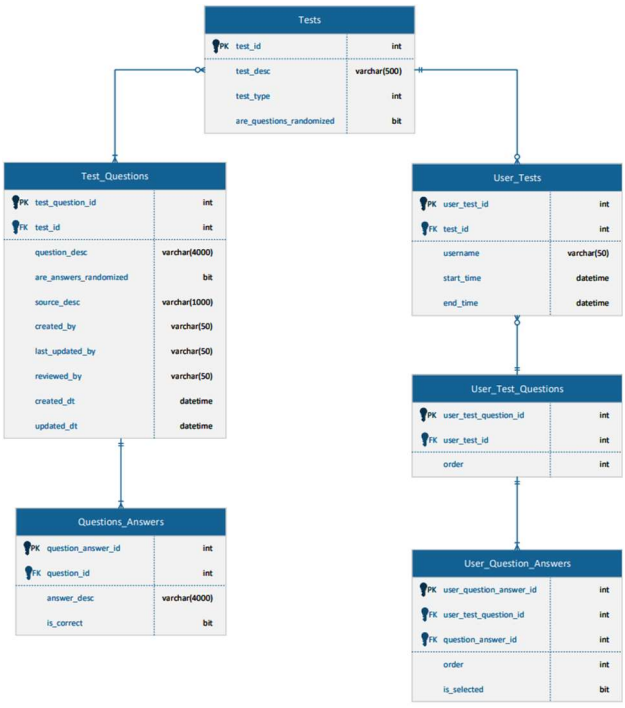

Figure 1: IT LMS Database Diagram

The tests users take are stored as 'Tests' tables in the database. The 'Tests' tables act as containers for test questions, and only store a basic text field for the test's name. Figure 1 shows additional data fields like a tag for questions that should be randomized, and a numerical tag for test categorization. These data fields represent features that were cut mid-development, and as a result, still remain in the database design. Each 'Tests' table is linked to multiple 'Test_Questions' tables that represent the questions asked by the test. Every table has a unique primary key which acts as the table's identifier. Tables are linked to each other through the use of foreign keys, which is when a table references another table's primary key.

'Test_Questions' tables are configured to store the question text as well as information on how the test question was created for documentation purposes. Each 'Test_Questions' table is linked to multiple 'Questions_Answers' tables that represent the answer choices to the question. These 'Question_Answers' tables store the answer text and whether or not the answer is correct.

User data is tracked through three tables that are allegorical to the tables we covered earlier: 'User_Tests', 'User_Test_Questions', and 'User_Question_Answers'. 'User_Tests' tables are connected to the 'Tests' table and track which user took the test and their start and end times. Linked to the 'User_Tests' tables are 'User_Test_Questions' that store the question number of the test. These tables are linked to multiple 'User_Question_Answers' tables that track whether or not the user chose certain answer choices for the question.

I originally included a table that represented a user's account, but after reading Microsoft's Database Design Basics article[2] I decided to remove this table in order to follow the article's recommendation of streamlining database designs. I noticed that because the IT LMS is an internal company tool embedded into the IT team's online development environment, I could simply have the tool pull username data from the dev environment and remove the need for the application to handle user accounts itself. With this, the application has no need for a specific user table since security is handled outside of the application and usernames are already guaranteed to be unique. The IT LMS only needs to attribute data to a username so that it can be queried.

3.2 Home Screen and User Metrics

The IT LMS web application has two main functions: allowing users to take tests and showing users their historical test data. To allow users to easily access these functions, I designed the application to start at a home screen from which users can navigate to interfaces dedicated to either of these functions. Users can access the test-taking interface by selecting a test from a large dropdown field at the center of the main screen and clicking a button labeled 'Take Test'. The user metrics interface is right below the 'Take Test' button, and users can scroll down to view the interface in its entirety.

The user metrics interface is a table that lists a user's test-taking history. Each entry in the table contains the test's name, the user's start and end time, as well as the user's score on the test. This information is pulled from the database by searching it for 'User_Tests' tables that have stored the current user's username. Test scores are then calculated by looking through the linked user questions and user answers tables and seeing how many questions only had correct answer choices marked selected divided by the total number of questions in the test. Users can also click the name of a test on the history table to see a list of the test's questions, answers, and the answer choices they selected.

Above the table is a banner that lists the user's overall test data, such as total number of tests taken, average and median score, as well as the total number of time spent taking tests. This data is calculated from the scores and start/end times listed in the user's test history table.

3.3 Test-Taking Interface

Choosing to take a test from the dropdown field at the head of the main page will send the user to a test-taking interface. I designed the test-taking interface to resemble most online testing sites, with the question text being displayed at the top of the screen and answer choices being listed below, along with check boxes for the user to select their choices. At the bottom of the screen are two buttons, 'Next' and 'Previous', which allow users to navigate between questions. Between the two buttons is a text field and a 'Go' button, where users can input a question number and jump directly to that question.

The test-taking interface loads its questions and answers from the database by querying all 'Test_Questions' tables that are linked to the 'Tests' table which represents the test the user decided to take. These 'Test_Questions' tables have their numerical primary keys sorted into a list from lowest to greatest and their position in the list represents their question number in the test. The interface then uses this list to find the primary key of the question it is supposed to display and queries the question's text and its linked answer choices from the database.

At the last question of the test, the 'Next' button displayed on the bottom right of the page is replaced with a 'Submit' button. Clicking this button will trigger the system to record all answered questions, chosen answer choices, and the start/end times of the test as new database entries in their respective 'User_...' tables. The user is then sent back to the main page and the test history table is updated with a new entry of the test the user has just finished taking.

4. Results

The system was deployed in August of 2021 and is currently being used by all fourteen members of the company's IT development team. The development team's tech lead has added a new test to the system every quarter since deployment, with each test consisting of a mix of old questions from previous tests and brand-new questions. Each member of the development team is required to take the quarterly test sometime during the month in which it is released.

To date, the development team has only taken three tests since the deployment of the system. Because of this, there is not enough data to concretely measure improvements in the team's knowledge. However, the tech lead has described an overall improvement in the team's knowledge base since the deployment of the system. He also finds the quarterly testing a great way to benchmark and track a team member's conceptual understanding, which was

difficult to do with only code reviews before the introduction of the learning system.

5. Conclusion

The IT LMS is an Appian web application which provides users the ability to take multiple choice tests related to IT concepts and track their history of test scores. The system gives developers a way to consistently engage with IT concepts outside of coding, keeping their knowledge of topics fresh. It also gives IT team leaders a way to measure their developer's conceptual performance, helping managers execute a more holistic performance review of their team members. However, the benefits of the IT LMS are not limited to the IT field. Administrators of the IT LMS can load whatever questions best suit their needs into the database, ensuring that the system can be easily retooled to provide value to any team.

6. Future Work

The IT LMS could be further improved with the addition of graphs to the home screen to visually represent user metrics and improvement. In its current state, the IT LMS only shows a simple table of historical scores, and it can be difficult to see improvements over time. Representing the same data through a line graph would make it easier for users to see their progress and the overall trend of their scores.

The IT LMS could also add additional categories for question types to the backend database to provide users with better feedback. Each question could be marked as testing a certain type of IT skill or concept, and these categories could be reflected in the user metrics, allowing users to see the skills they are excelling in and the ones they are having difficulties with.

Questions could also be marked as having only one correct answer, or multiple correct answers. This would allow for the webpage to detect whether or not the question has several correct answer choices and use either multiple selection check boxes or single selection radio buttons for users to mark their answer choices depending on the situation, improving the user's test-taking experience.

References

[1] Google. Web Fundamentals. Retrieved from https://developers.google.com/web/fundamentals/.

[2] Microsoft. Database Design Basics. Retrieved from https://support.microsoft.com/en-us/office/database-design-basics-eb2159cf-1e30-401a-8084-bd4f9c9ca1f5