[1]

Duolingo DataBase Model

Marlon Yecid Riveros Guio 20231020208
Brayan Estiven Aguirre Aristizabal 20231020156

Abstract -This project aims to reverse engineer core features of the Duolingo platform to understand its underlying database architecture. The context revolves around the growing demand for language learning tools and Duolingo's success as a popular solution that delivers an adaptive and engaging experience. We propose a detailed reverse engineering process to analyze and model the main entities, attributes, and relationships that form the Duolingo database, including user progress, lessons, and reward systems. The most relevant result is an entity relationship diagram and a complete and robust database model that breaks down many-to-many relationships, ensuring scalability and data integrity.

I. Introduction

Language learning has been one of the most learned topics in the world for some time now. Acquiring languages makes it easier to travel around the world, acquire different opportunities, etc. which opens up countless opportunities. Nowadays, learning languages is not very difficult because with the development of technology and the Internet, learning a language is as easy as accessing your cell phone or computer and asking for a language course. Thanks to events like these, various companies have dedicated themselves to making contact with a second language more accessible, which is why various platforms have emerged that allow access to them. Currently, the platform that stands out the most in this area is duolingo, which has gained fame and has specialized in language teaching.

Duolingo is an application that offers a wide range of languages to learn. This platform offers a complete user experience where we can learn in a modular way the different levels that surround a language. Particularly, each course is divided into stages, sections and lessons, which facilitates the understanding of the language being learned. It also offers a detailed report on how we progress with each course and also offers us a system of challenges and rewards that motivate the user to learn more every day. On the other hand, this tool uses pedagogical and psychological techniques that facilitate language learning and make learning something new more enjoyable.

However, the design of a language application like Duolingo is not open to everyone, which unfortunately makes the task of developing apps that surpass Duolingo more difficult, which is why in this paper we use techniques such as reverse engineering to understand the data structure behind this application. Understanding how the entities and components of this application are related and how the data flows in it shows a light for developers interested in manufacturing tools of such caliber as Duolingo, so this paper will give an introduction to a possible database model. data that duolingo could have. By detailing all this, we will be able to thoroughly understand how Duolingo works in order to open possible paths to the development of applications of this type.

The approach presented in this paper focuses on defining the critical entities and relationships within Duolingo's system architecture, such as users, courses, lessons, and progress metrics. These entities are essential in understanding how Duolingo stores and manipulates data to provide a seamless learning experience. For example, the user entity tracks personal details, course progress, and achievement status, while the lesson entity categorizes learning materials by language, difficulty, and interaction type. Additionally, we explore how Duolingo's reward systems, such as achievements and boosters, integrate into the learning process to create a motivating and addictive user experience.

Our methodology to meet the aforementioned objective is based on using the ER diagram as a modeling tool. Using this tool requires a complete understanding of the entities involved in the app and how they are related, so we use 10 steps that allow us to obtain this data. These 10 steps are 10 stages that make the development of a relational database more natural, which allows us to understand in a simple way how the entities are related, thus proposing attributes and phrases that allow us to naturally model the data structure of the database. the application. This will help us to easily find the possible duolingo database, thus facilitating the development of language-oriented applications.

In conclusion, this paper does not focus on studying the impact that duolingo has on people's lives, but rather offers a global understanding of how the data of this application is structured. Achieving this objective would open the doors to a future where we can all have simple tools at hand that facilitate the learning of more than one language in such a way that there is no excuse for learning comfortably. On the other hand, it delves into the understanding of reverse engineering and database modeling through simple tools but with a specific purpose. Below, the steps and findings that were found in pursuit of the objective of understanding how Duolingo works will be presented.

II. Methodology

For the realization of this project, a methodology known as the Entity Relationship Model (ERM) was used, an efficient methodology that included an in-depth analysis of the application to recognize the main components that are necessary for the construction of a solid and scalable database. Then, the main entities and their attributes were defined,

[1]

ensuring that each element necessary for the functionality of the application was represented. After this, the relationships between the entities were verified to describe the interactions within the application, defining which ones were related to each other and defining the types of relationships that each of these presented. Next, a first sketch of what would be the first entity relationship diagram was made.

Once the first outline was finished, a review of all the analysis carried out up to that point was carried out, in which small errors were corrected and the new entities were defined, in charge of breaking the "many to many" relationships that were presented in the first entity relationship diagram. The entity-relationship diagram was then validated by reviewing each component and testing the consistency of the relationships between entities, ensuring that all system interactions were correctly represented. The design was then refined by implementing each entity's data structure and properties, so that the model could be even more robust. The final result was a solid, efficient and adaptable model, ready to be implemented as a database for optimal management of users, courses, lessons and other components of the Duolingo system.

III. METHOD AND MATERIALS

Firstly, a technical discussion was held to reach an agreement on what the main components of the application would be. The process began with a deep analysis of Duolingo's core functionality to identify the components needed for a robust and scalable database. Key features analyzed included:

1. User registration and authentication.
2. Storage of user information
3. Catalog of available languages to learn.
4. Structuring lessons by language.
5. Classification of lessons by difficulty levels.
6. Storage of lesson content.
7. Show Conversations and expressions
8. Interactive Exercises
9. Achievement and Reward System
10. Show the Progress

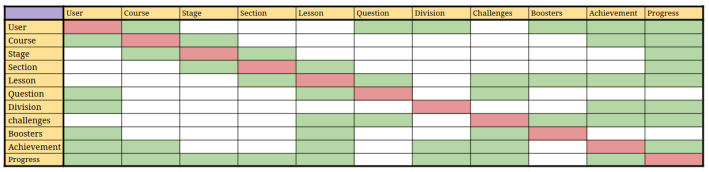
Once the essential components were identified, the main entities or tables that will form part of the database structure were defined. These entities include:

1. User
2. Course
3. Stage
4. Section
5. Lesson
6. Question
7. Division
8. Challenges
9. Boosters
10. Achievement
11. progress

In this step, the attributes or characteristics of each of the identified entities were detailed. The resulting attributes were:

1. User: name, nickname, joineDate, division, timesInTop, email, password, courses, achievements, followed, followers, gemsNumber, livesNumber, boostersNumber, isPremium.
2. Course: language, Stages, Sounds, id
3. Stage: Sections, id, level, minimumRequirements.
4. Section: Lessons, Guide, id, name.
5. Lesson: type, QuestionsNumber, EXP, time, Accuracy, id.
6. Question: Content, answer, isCorrect, type, id
7. Division: id, name, climbingZone. DescentZone, stayArea, users
8. Challenges: id, description, progress, goal, reward
9. Boosters: id, name, cost, benefit, maxNumber
10. Achievement: id, name, progress, goal.
11. Progress: id, Course, User, streakDays, DailyEXP, WeeklyEXP, TotalEXP, percentage

Detailed and exhaustive definitions were carried out on the interactions and connections that exist between the various entities involved:



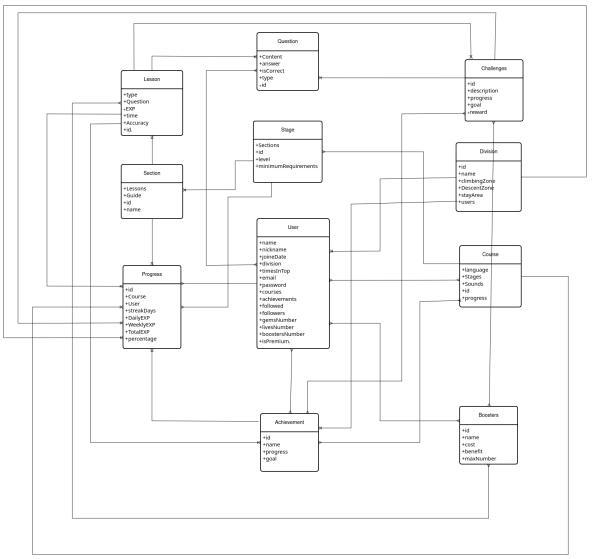
A few examples of sentences that helped us find existing relationships were:

1. User-Course: One user can take many courses and many courses can be completed by many users.
2. User-Question: One user can solve multiple questions and one question can be solved by multiple users.
3. User-Division: A user can be in a single division and in a division there can be multiple users.
4. complete multiple challenges, and one challenge can use multiple boosters.
5. Challenges-Achievements: A challenge can grant achievements upon completion, and an achievement can be related to multiple challenges.
6. Challenges-Progress: A progress record belongs to a challenge, and a challenge can have progress records for multiple users.
7. Achievement-Progress: A progress record belongs to an achievement, and an achievement can have progress records for multiple users.

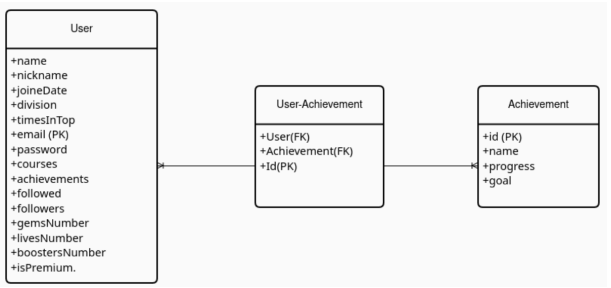
We continue with the classification of the relationships for the 25 relationships that were found:

1. User-Course: Many to many (* → *)
2. User-Question: Many to many (* → *)
3. User-Division: One to many (1 → *)
4. User-Boosters: Many to many (* → *)
5. User-Achievement: Many to many (* → *)
6. User-Progress: One to many (1 → *)
7. Course-Stage: One to many (1 → *)
8. Course-Achievement: Many to many (* → *)
9. Course-Progress: One to many (1 → *)
10. Stage-Section: One to many (1 → *)
11. Stage-Progress: One to many (1 → *)
12. Section-Lesson: One to many (1 → *)
13. Section-Progress: One to many (1 → *)
14. Lesson-Question: One to many (1 → *)
15. Lesson-Challenges: One to many (1 → *)
16. Lesson-Boosters: Many to many (* → *)
17. Lesson-Achievement: One to many (1 → *)
18. Lesson-Progress: One to many (1 → *)
19. Question-Challenges: One to many (1 → *)
20. Division-Achievement: One to many (1 → *)
21. Division-Progress: One to many (1 → *)
22. Challenges-Boosters: Many to many (* → *)
23. Challenges-Achievement: Many to many (* → *)
24. Challenges-Progress: One to many (1 → *)
25. Achievement-Progress: One to many (1 → *)

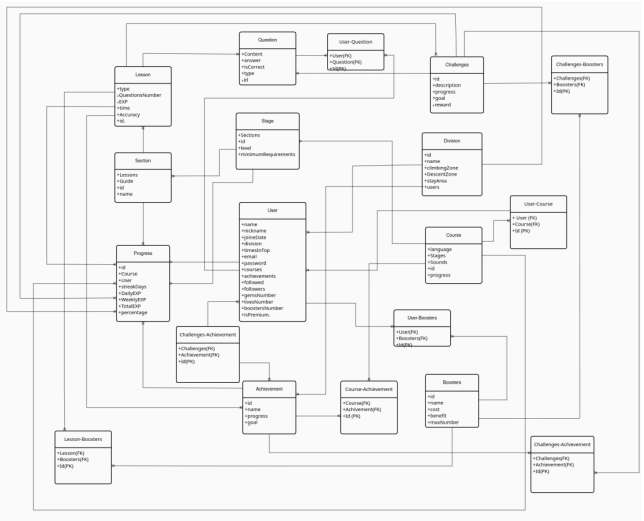
Then the first entity relationship diagram was designed, the result is shown in the image:



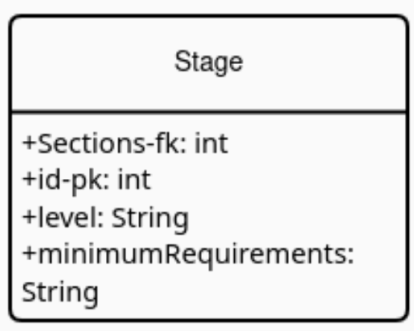
Once this was finished, the existing "many to many" entities were broken up. To do this, an additional entity was created, and the two original entities were connected to it with a "one to many" relationship, and taking as attributes in the new entity two foreign keys that would be our original entities.



Then we make a second diagram with the new entities, like this:



Finally, the data structure was categorized and the properties were added to each attribute of our entities.



To define the data structure we look at what function this attribute is fulfilling in the system, so we categorize it by "int, String, Date, etc."

And the restrictions and properties were provided to these attributes according to the functionalities analyzed above.

Stage:

1. Sections: Not null
2. Id: Not null, unique
3. Level: Varchar(2), Not null
4. minimumRequirements: Text (without limits), Not null

IV. Conclusions

The reverse engineering process presented in this article provided important insights into the structural and functional aspects of Duolingo's database and system architecture. Following the 10 steps outlined in this article, we successfully analyzed the core components of the Duolingo platform, showing how each entity—from users and lessons to achievements and progress metrics—interacts within the system. This systematic approach allowed us to gain a deeper understanding of the key relationships and constraints that make Duolingo an effective and scalable educational tool.

Each of the ten steps played a critical role in simplifying Duolingo's complex database model. From defining key components such as user registration, course structure, and progress tracking, to mapping relationships between entities such as lessons, milestones, and assignments, we were able to build a comprehensive entity relationship (ER) model. The detailed exploration of attributes and relationship types provided a clear framework that not only explains how Duolingo stores and processes data, but also serves as a guide for the design of similar databases in other educational or interactive platforms.

The standard division of components into phases, sections, and lessons, as well as the definition of one-to-many and many-to-many relationships, allowed us to draw an efficient and scalable data structure. This process is further enhanced by careful consideration of constraints and properties that ensure data integrity and consistency. Using this methodology, we have shown how these steps can be applied to simplify database modeling in general, providing a template for developers striving to create robust and dynamic systems based on user interaction and adaptive content delivery.

Ultimately, the database modeling and reverse engineering techniques presented here serve as valuable tools for understanding and replicating complex systems like Duolingo. Following this structured approach, developers can create more flexible, scalable, and user-centric applications, driving innovation in educational technology. The lessons learned from this analysis not only clarify the inner workings of Duolingo, but also provide a practical foundation for future projects in similar areas.