CS 353 - Database Systems

2019 - 2020 Spring Semester

Project Proposal

"*CodeGiant*"

Team Number:	18	
Team Members:	Berke Oğuz	21601100
	Ibrahim Eren Tilla	21702537
	Alkım Önen	21703549
	Talha Burak Çuhadar	21703821

Table of Contents

1. Introduction

This is the project proposal of our “Coding Practice and Interview System”, *CodeGiant*. The proposal will consist of a brief description of the system that we are going to create, an explanation of both its functional and non-functional requirements alongside with its limitations, followed by the entity-relationship model as the conceptual design and end with the conclusion.

In the description section, the application system that we are proposing will be explained thoroughly, and it will be discussed how and why a database is needed and its main functionalities as a part of our system. In the requirements section, how and what the system provides to administrators, users and companies will be examined as well as pointing out the issues of performance, reliability, usability, capacity, and security for our system. After that, the limitations will be explicitly clarified. Then, the conceptual design will be examined in our entity-relationship diagram where the interaction between various entities can be seen. Our proposal will end with a short conclusion as a summary of what has been taken into account.

2. Description

As a “Coding Practice and Interview System”, our platform is going to be very similar to the current uses on the internet, like HackerRank and LeetCode. It aims to provide a user-friendly space where companies and developers can interact in a competitive environment. It will help people who want to practice their coding skills as well as helping companies that are seeking proficient programmers.

It will feature 3 different users: “Developers” (which will be referred to as “Users”, for they are the main focus of our system), “Administrators” (which will be referred to as “Admins” as an abbreviation) and “Company Representatives” (which will be referred to as “Representatives”). Out of these three user types, users and representatives will have their accounts to use the system, while admins will have a different view of it. All of these user types will be stored in the database.

A user will be able to create an account with his/her username, e-mail, and password. After logging in to the system, they will be able to create a question, select a track and practice coding, or select a company coding interview and try to solve its questions in time. All of the mentioned entities will be stored in the database as well. All questions and all tracks will feature a discussion section where puzzled users can ask questions and get answers from other users.

A company representative will be able to create a company track where he/she can choose and add from any subject to that particular track for recruitment. They will be able to see the profile of the applicant users and filter the ones that they do not think are suitable out of their recruitment process.

An administrator will be able to create a track that is specialized in a subject and arrange and update the existing tracks. They will be able to check and either confirm or deny the submitted questions. They will also be responsible for the assessment of the submitted answers to the existing questions. To make sure that there is no inappropriate behavior in the system, they will be able to ban particular users accordingly to their misbehavior.

3. Requirements

3.1 Functional Requirements

3.1.1 Users

The system provides users with coding practice and coding-interviews with different companies on a web-based platform. Users can solve different questions related to the programming language they choose. Users can ask questions on the discussion sections regarding the problem. Users can interview with company representatives of the companies that they applied for the recruitment. Users can create coding problems and submit to the system admin. Afterward, these coding problems can be published with admin permission.

The system has a leaderboard table so that users can see their rankings among other users who solve the questions on the same track. Also, users can manage their profiles.

3.1.2 Company Representatives

The system allows company representatives to create their interview questions and let them interview the candidates who apply for the recruitment. Company representatives can create their profile page and see the profiles of the users that applied to their company. Company representatives can see the scores of the users who applied for the recruitment.

3.1.3 Admins

The system allows admins to create new tracks and verify coding questions before making them public for the website. Admins can evaluate the questions that are sent by users and they can publish those questions on the website as coding practice questions. Admins are allowed to ban accounts and delete the inappropriate comments on the forum.

3.2 Non-Functional Requirements

3.2.1 Performance

Simple tasks such as navigating the app must be done within 2 seconds. Code sent by the user must be processed within 15 seconds. Interviews have limited time to complete. Therefore, the system must have as little time delay as possible to bring good performance to both users and company representatives.

3.2.2 Reliability

Codes sent from users must be run and send the output as expected. System maintenance should be done regularly to ensure the main functionalities work properly. Any occurring bug must be fixed within a day.

3.2.3 Usability

The user interface of the system must be easy to use and navigate through. The color design of the application must be easy to distinguish.

3.2.4 Capacity

Coding questions, forum comments, user scores, and user profile information will be held on the database system. Therefore, the database should be able to hold large amounts of questions, a forum, and user data.

3.2.5 Security

Users will sign-up to the website with their emails, usernames, and passwords. The database system will provide data privacy to user information.

3.2.6 Availability

The system should be highly available at all times with only toleration of 5 minutes downtime.

4. Limitations

User

- Users can run their code only once every 10 seconds to not bottleneck the system.
- Users can submit their code only once every 30 seconds to not bottleneck the system.
- Users must enroll in the system with a unique email and a username, and a password that complies with the security standard of minimum 8 characters with different letters.
- Input taken from users or input in a test case must be small enough to fit in the memory.
- Code run/submitted from the user must be small enough to compile within 3 seconds.
- Contact information of a user can be only shown to company representatives with the permission of the user.
- The code run by a user can only run for 5 seconds.

Company Representatives

- Company representatives must be verified before interviewing with users.
- There must be only one account for a company.

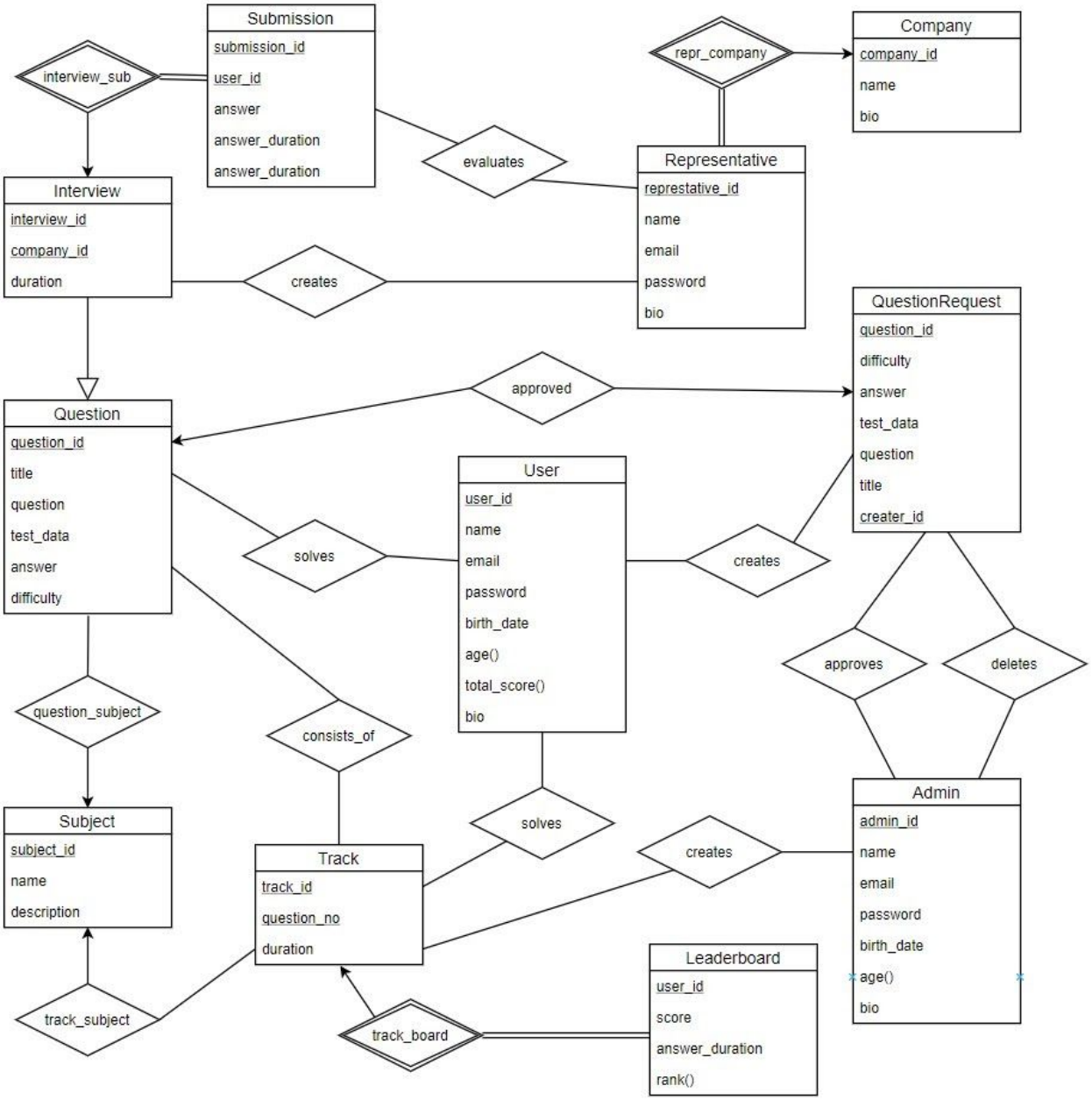
Question

- All questions should be unique.
- A developer can only submit 20 new questions a day.
- The description of a question must be at a maximum of 4000 characters long.
- The description of a question can only contain 2 images to help with the explanation.
- Questions can only have 15 test cases.
- Test cases' input size must be small enough to run in a maximum of 5 seconds in an optimized solution.

Leaderboard

- The leaderboard can only have the users who solved and submitted at least 10 questions.
- The leaderboard must be updated every day.

5. Conceptual Design (E/R Diagram)



6. Conclusion

CodeGiant is a web-based application. It is a coding practice platform designed for developers and also a recruitment environment for companies. The system that we intend to build tries to capture the specifications and the relations between developers, company representatives, admins and questions, tracks, interviews.

In this report, we described the purpose of the project. We gave some information about how the system is going to work and explained the importance of using a database management system as a part of our project. We presented the functional and non-functional requirements of our applications. Then we resolved the limitations of our project. In the end, we designed our base database system and constructed an entity relation diagram for it.

7. Website

All the documentation of our project in the future will be visible on the following link:

https://github.com/erentilla/CS353-Project/tree/master/docs