Quiz Maker

System Design

<V 1.0>

<02.12.2018>

Atilla Özder

Mert Çetinkaya

Begüm Mina Bilgin

Selin Atalay

Prepared for

SE301 Software Engineering



Table of Contents

[1. Introduction 1](#_Toc433996772)

[1.1. Purpose of the System 1](#_Toc433996773)

[1.2. Design Goals 1](#_Toc433996774)

[1.3. Definitions, Acronyms, and Abbreviations 1](#_Toc433996775)

[1.4. References 1](#_Toc433996776)

[2. Current Software Architecture 1](#_Toc433996777)

[3. Proposed Software Architecture 1](#_Toc433996778)

[3.1. Overview 1](#_Toc433996779)

[3.2. System Decomposition 1](#_Toc433996780)

[3.3. Hardware Software Mapping 2](#_Toc433996781)

[3.4. Persistent Data Management 2](#_Toc433996782)

[3.5. Access Control and Security 2](#_Toc433996783)

[3.6. Global Software Control 2](#_Toc433996784)

[3.7. Boundary Conditions 2](#_Toc433996785)

[4. Subsystem Services 2](#_Toc433996786)

[5. References 2](#_Toc433996787)

SYSTEM DESIGN DOCUMENT

# Introduction

Design is the abstraction of an answer; it's the common description of the answer to an issue without details. Design is a view sample seen within the evaluation part to be a sample in a design part. After the design part we are able to reduce the time required for implementation.

## Purpose of the System

As we said within the Requirements Analysis Document, the aim of the system is offering to system administrator, instructors, student and users a central location for organizing varied quiz occasions. The aim of this technique supplies the next causes: No physical presence wanted for quizzes, no wastage of time throughout analysis, prompt availability of the outcomes, beginning and completion time are each recorded by the server, a collection of on-line assessments are provided by the software program for the good thing about the scholars. Normally the system is far more preferred as a consequence of these causes across various organizations and the world.

## Design Goals

The design objectives signify the specified qualities of quiz maker system and supply a constant set of standards that should be thought of when making design choices. Based on non-functionality requirements, the next design targets must achieved as a way to qualify the system as profitable:

* **Security**

The system security is one of the most important non-functional requirements.

* **Reliability**

The system has to carry out the quiz maker operations with no errors. The application developed needs to be extremely dependable and safe in order that details about any questions and many others shouldn’t be leaked earlier than the precise quiz is held.

* **End User**

Students can take quiz only throughout the previously allotted time slots, however can open application anytime to access different data. System should be able to deal with a number of users. This system should run on IOS operating system.

* **Performance**

The system has to be sturdy enough to manage any valid input from the users.

* **Flexibility:**

It requires Internet.

* **Implementation:**

System should be modifiable and readable. The number of errors must be minimized.

## Definitions, Acronyms, and Abbreviations

**Student**: The user of the system who will attend to the quiz. At the same time will be able to create a quiz.

**Admin**: The system administrator who will manage all data of system and who controls users.

**Instructor**: The user who will prepare and present the questions to the students and calculate their scores by entering the answers of the questions.

**Registered User:** Only public quizzes are allowed for this type of user to create. He/ She will be able to create public quizzes and join public quizzes.

**Visitor:** Visitor can only register and than login in the application.

## References

- There are many quiz maker applications. For example; <https://www.quiz-maker.com/>

<https://www.easypromosapp.com/quiz/>. But our system some different from others.

# Current Software Architecture

There are some websites and applications which have similar purposes as our system. These systems are allowing users to create or join some quizzes. The applications mentioned are iSpring QuizMaker[2], Easypromos Quiz[3] and Typeform Quiz[4], and the websites are Quiz-Maker[5] and SurveyMonkey[6].

iSpring QuizMaker is made for measuring the competency levels of staff members. Easypromos Quiz is made for testing the followers’ knowledge of a brand and products or a specific topic using a quiz. Typeform Quiz is based on business growth, personal or professional growth and it is designed to teachers or corporate trainers helping people with new knowledge and skills.

The website Quiz-Maker is a site where you can start building your own quizzes at the first page you see, you can prepare your own quizzes and share it with anyone you’d like to share. SurveyMonkey is a website where you can prepare your own surveys and share it with the people you want, it can be used like our system but the main idea is different.

There is no system like our QuizMaker System which is based on education and providing instructors to create lecture quizzes to the students which are related with those classes and also to the registered user to create quizzes by their own and to make people join to the quizzes they have prepared.

# Proposed Software Architecture

We support a system that contains design patterns. According to this design pattern, we did a clear implementation. In addition, the strategy of the design pattern leads to our project. Finally, maintenance and operating become easier and proper.

Also, we support a system that has been implemented with the layered architecture. According to this layered architecture, there is a hierarchy of layer, each layer using services are offered by the lower layers. We implement our project with the layered architecture, all of the functions are called from services. Files of webservices contains all functions that are called web-service. All methods will be defined in here.

Our system is an object-oriented system. This is a technique of coding and we will use database in the part of back-end. The front-end part will include logical pages and the logical pages are using web services.

## Overview

The system is a user friendly system and all users can use the system easily. The system has four interfaces.

**Admin Interface:** It provides services to admin to manage the system. It allows admin to add courses, delete courses, approve instructors, reject instructors, delete quizzes and delete the other users (students and registered users).

**Instructor Subsystem:** It provides services that allows instructors to login, create lecture quizzes, update lecture quizzes, delete lecture quizzes, add students, edit their own profiles and change their password.

**Student Subsystem:** It provides services to the students which allows students to join lecture quizzes, create public quizzes, join public quizzes, edit their own profiles, update public quizzes and change password.

**Registered User Subsystem:** It provides services to the registered users which allows them to create public quizzes, join public quizzes, edit their own profiles, update public quizzes and change password.

**Visitor Subsystem:** It allows visitors to view quiz times and names.

**Login Component:**  It allows the system administrator, instructors, students or registered users login to the system.

**Registered Component:** It allows visitors to register the system.

**Create Lecture Quiz Component:** It allows instructors to add quizzes into the system.

**Update Lecture Quiz Component:** It allows instructors to update quizzes of their lectures.

**Delete Lecture Quiz Component:** It allows instructors and admins to delete quizzes of the students in the system.

**Add Student Component:** It allows instructors to add students for a course on the system. Instructors can search for students by their user name, e-mail or student id to add students to the system.

**Edit Profile Component:** It allows every user to do changes of their own information which are their first names, last names and e-mail addresses in the system.

**Change Password Component:** It allows every user to change their password in the system. (except for the visitor).

**Join Lecture Quiz Component:** It allows students to join lecture quizzes in the system.

**Create Public Quiz Component:** It allows students and registered users to create quizzes for everyone which are different from the lecture quizzes in the system.

**Join Public Quiz Component:** It allows students and registered users to join public quizzes in the system.

**Update Public Quiz Component:** It allows students and registered users to update quizzes in the system.

## System Decomposition

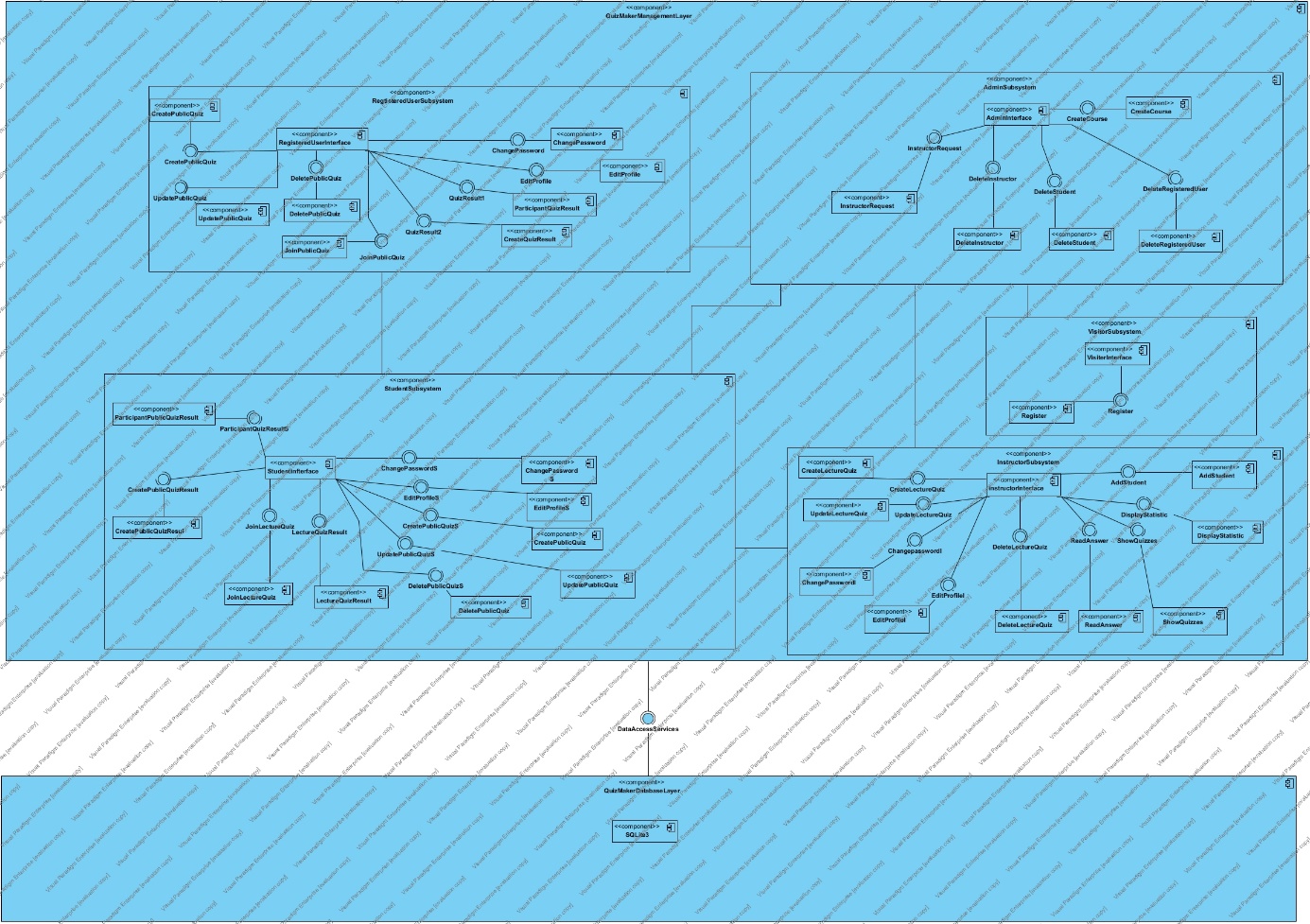


Figure 1: Coupling view of Subsystem Decomposition

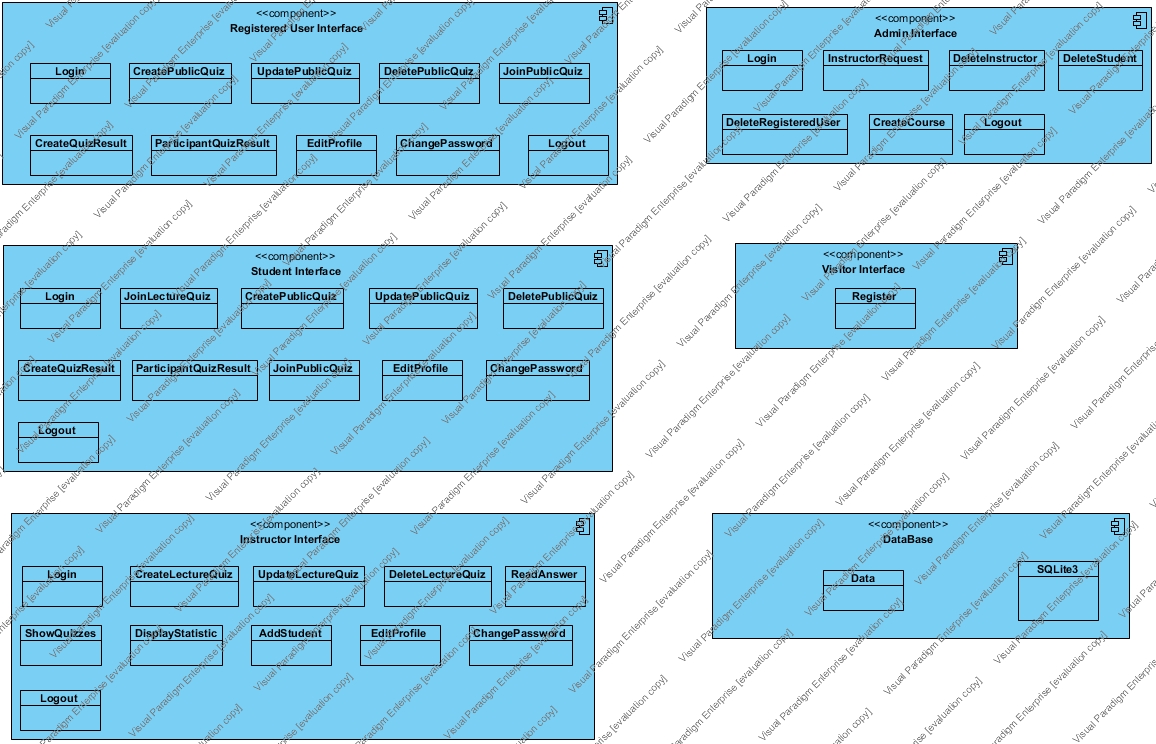
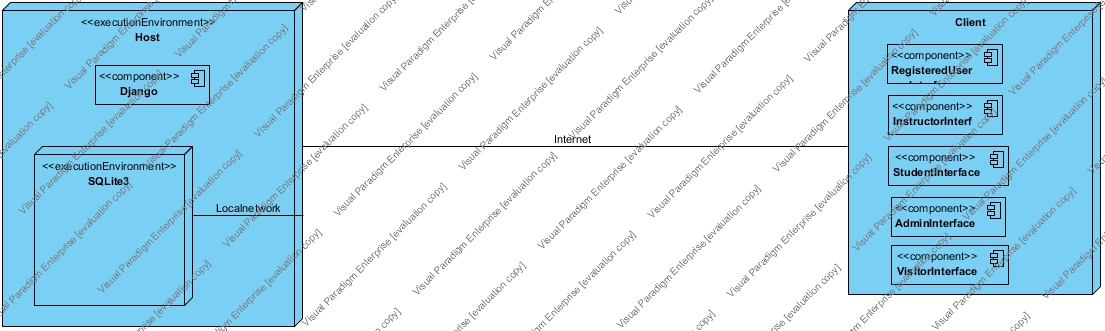
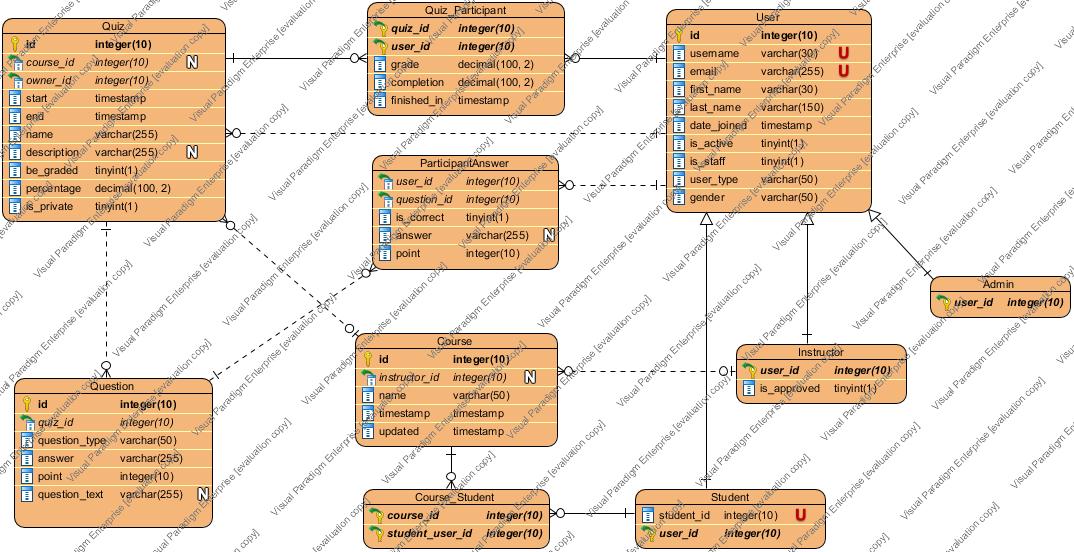


Figure 2: Cohesion view of Subsystem Decomposition

## Hardware Software Mapping



## Persistent Data Management



***User, Instructor, Student, Admin***

These tables store all the information about users. User tables have ten columns which are id, username, email, first\_name, last\_name, date\_joined, is\_active, is\_staff, user\_type and gender. ID is a unique integer primary key which is generated by the system. Username and email are also unique keys.

***Course***

Course table stores all the information about courses. The table has five columns which are id, instructor\_id, name, timestamp, updated. ID is a unique integer primary key and instructor\_id is an integer foreign key which cannot be null.

***Question***

Question table stores all information about questions. The table has five columns which are id, quiz\_id, question\_type, answer, point and question\_type. ID is a unique integer primary key and quiz\_id is an integer foreign key.

***Quiz***

Quiz table stores all information about quizzes. The table has ten columns which are id, course\_id, owner\_id, start, end, name, description, be\_graded, percentage, is\_private. ID is a unique integer primary key. course\_id is an integer foreign key of course table. owner\_id is an integer foreign key of user table.

## Access Control and Security

Quiz maker is a mobile application running with swift. It is created with object-oriented programing. So, there are classes. It is a classical mobile application. It is a multiuser application that has more than one actor. The actors are admin, instructor, student, registered user and visitor.

There have been object oriented architecture. So, we implemented classes which are organized by actors. In this project, we are setting the roles in databases. When a registered user tries to login to the system and if s/he makes a successful login, the user will be redirected to the main page as user. User accesses services by the actor role.

User accesses to the system with a uniquely defined e-mail, user name and password at registration page.

## Global Software Control

The system controls the authentication with a defined unique mail, username and password. The synchronization is satisfied among queries on server. Thus, dynamic content will be displayed.

## Boundary Conditions

Quiz Maker has to be uploaded to App Store to work actively. Uploading stage does project compiler and our application can be uploaded on app store and after uploading the project, we do some changes for our project and we take a screen shot for uploading the application to app store and we publish on app store. Now this project waiting for a confirmation from app store officials. App store officials approve project, then the project can be published on app store.

The errors could be originated from attempting to login with an invalid e-mail or password or trying to update the personal information with empty information. All these exceptions are getting caught by the system and handled properly.

When the system is terminated by system our remove application on app store. All users have to download it in app store to continue to use this application but these users can’t get an update in addition, other users don’t download this application.

# Subsystem Services

The subsystem decomposition of Quiz Maker application, we divide the system into smaller subsystems with strong coherences.

The subsystem separation shows the entities of following subsystem

* Visitor
* Registered User Subsystem
* Instructor Subsystem
* Student Subsystem
* Admin Subsystem
* Database Subsystem

**Registered User Subsystem**

This subsystem manages quiz activity with user type which is registered user.

This subsystem provided by:

* Login
* CreatePublicQuiz
* UpdatePublicQuiz
* DeletePublicQuiz
* EditProfile
* ChangePassword
* ShownAnswer
* DisplayAnswer
* Logout

**Instructor Subsystem**

This subsystem provides managing quiz activity for student.

This subsystem is provided by:

* Login
* CreateLectureQuiz
* UpdateLectureQuiz
* DeleteLectureQuiz
* EditProfile
* ChangePassword
* ReadAnswer
* AddStudent(with search student)
* ShownAnswer
* DisplayAnswer
* Logout

**Student Subsystem**

This subsystem provides to manage quiz activities and participant quiz activities.

This subsystem is provided by:

* Login
* JoinLectureQuiz
* CreatePublicQuiz
* JoinPublicQuiz
* EditProfile
* ChangePassword
* QuizResultforCreated
* ShownAnswer
* DeletePublicQuiz
* Logout

**Admin Subsystem**

This subsystem provides to manage all system.

This subsystem provided by:

* Login
* CreateCourse
* UpdateCourse
* DeleteCourse
* ApproveInstructor
* RejectInstructor
* DeleteRegisteredUser
* DeleteStudent
* DeleteInstructor
* DeleteQuiz
* Logout

**Visitor Subsystem**

This subsystem provides to registeractivity.

This subsystem provided by:

* Register

# References

1. Bruegge B. & Dutoit A.H.. (2010). *Object-Oriented Software Engineering Using UML, Patterns, and Java*, Prentice Hall, 3rd ed.
2. <https://www.ispringsolutions.com/ispring-quizmaker>
3. <https://www.easypromosapp.com/quiz/>
4. <https://www.typeform.com/quizzes/>
5. <https://www.quiz-maker.com/>
6. <https://www.surveymonkey.com/mp/quiz/>