# KOCAELİ ÜNİVERSİTESİ

## TEKNOLOJİ FAKÜLTESİ

## BİLİŞİM SİSTEMLERİ MÜHENDİSLİĞİ BÖLÜMÜ

Yazılım Geliştirme Proje-1 Final Raporu

2019-2020 Güz Dönemi

171307052 – ONUR KANTAR
161307066 – MUHAMMED BEDAVİ

## Özet:

Course Management System genellikle üniversitelerin sınav değerlendirmelerini ve notlandırma konularında öğretmenlere ve yöneticilere kolaylık sağlamaktadır. Bu raporda üniversitelerin kolaylıkla kullanabileceği bir ders yönetim sistemi anlatılmıştır. Proje WPF, MSSQL kullanılıp C# koduyla yazılmıştır. Ayrıca proje Admin paneli ve öğretmen paneli olmak üzere 2 kısımdan oluşmaktadır.

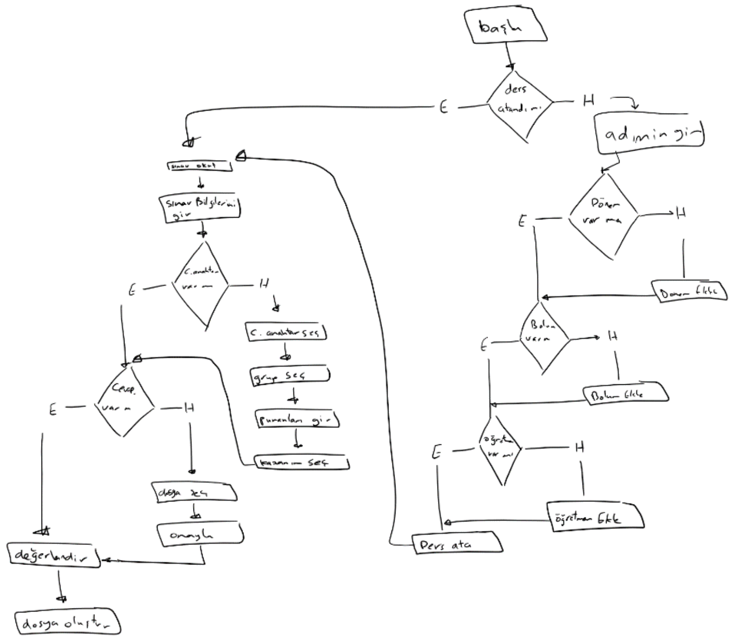
## Problem Tanımı:

Üniversitelerde öğretmenlerin en çok ihtiyaç duyduğu şeylerden bazıları da yaptığı sınavların kolay bir şekilde değerlendirilip, soruların hangi oranda doğru yapıldığı hangi kazanımın başarı oranının daha yüksek olduğu hakkında bilgi sahibi olmaktır. Bu bilgilere kolay ve hızlı bir şekilde sahip olmaları da çok önemlidir. Bizim geliştirdiğimiz bu uygulama bu sorunu tamamıyla çözüp bütün üniversitelerin fakültelerinde kolaylıkla kullanabilecektir. Bu sayede de üniversitelerin başarı oranı artacaktır.

## How we developed the system:

We started by analyzing the given project file and tried to reduce the big project to a much small problems to make solving them more easy, then we took every problem and started to studying it and finding the optimal algorithm to solving it (like how we crated the Create Exam window to take firstly the Answer Keys and based on them creating the Exam Groups and the Questions Count for each group which will help us when evaluation the students answers).

After finishing the pervious step, we created the first flow chart diagram for the full project with the basic procedures foreach section of the program, and with that part out of the way we had a pretty good imagination of what the project will be like and based on that we started the design process.

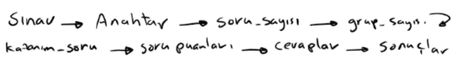


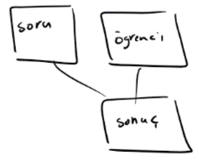
As a first step we started with designing the database with its tables and the relations between them, although in the project requirement there was no mention for the need to store the student data or their answers in the database, we decided to store them for three reasons:

1- Limit the student creation process to the admins to reduce the probability of wrong data entry by the teachers.
2- Reduce and auto solve the mistakes made by students on the optic paper regarding their personal data like first and last name which will make the exam evaluation process easier for the teacher
3- By storing all the students answers in the database we allow the results to be edited without the need to reenter all the data and evaluation it once more and in case of a lost excel file, it can be recreated with a click of a button.

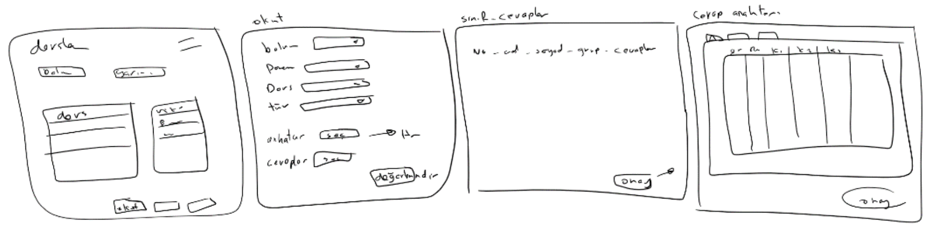
**Our exam creation process consists of:**

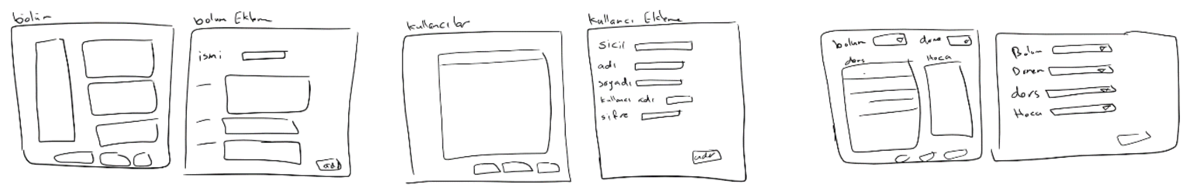
1- Selecting an assignment
2- Creating an exam with a type and date
3- Based on the answer keys creating exam groups
4- Foreach question in the answer keys creating a question and assigning to it the wanted outcomes
5- Foreach question and student in a given exam group creating a result with the student’s answer





After finishing the design of the database, we started to design the program UI by sketching the wanted windows and panel with the basic functionality of each one.





With the ending of the early designing of the system we started the research phase for the project which consisted of these parts:

A- The Platform: according to our past skills and the given period of the project we decided to use the .Net platform on windows with C# language to develop the project with more advanced programing techniques and design patterns.

B- The UI Type: due to the lack of UI scaling feature in windows form and the fact that it is using ancient technology for displaying content we decided to use WPF with MahApp.Metro framework to achieve a good looking, responsive and scalable UI with minimum effort in addition to the ability to design the UI separated from the code behind which gave us flexibility in making updates and improvement to the UI without the need to change the back-end side of the forms.

C- Database: as mentioned above we decided to go with C# as the programming language for the project we didn't find a better solution than MS SQL Serve to handle the database for the following reasons:
  - Being developed by Microsoft assures the compatibility and optimization for Windows
  - Better compatibility and speeds when working with C#

## Used Technologies:

1- Visual Studio 2017 for C# development.
2- Ms SQL Server for database.
3- Dapper for connecting the C# (back-end) to the database with its API.
4- MahApps.Metro as a WPF library for UI design.
5- MahApps.Metro.IconPack as an icon library for UI design.
6- Squirrel.Windows for deploying the app as an oneClick executable.
7- Office.Core for writing data to Excel.
8- XAML language to create the UI for WPF app.
9- GitHub for version control and hosting the app releases online.

## Programing Techniques:

### - Business Logic:

A- We opted for the use of a class library (CMSLibrary) to separate the bossiness logic from the UI completely and to give us the flexibility to change the UI in the future without the need to change anything in the library.
B- In CMSLibrary we have an IDataConnection interface specifying the required functions for a DataConnection class whether it's a SQL or MySQL or any different Database which will make it easier to migrate to a new database if needed with writing more than a one class
C- We used Enums for our in-program options which made the development process easier and error free.
D- We used Data Annotations for validating the user input in the app in real-time with superior decrees in code-repetition according to classic if-else validation.
E- The using of models to represent every database table in the back-end with some special property made our job much easier without the need to make

unnecessary queries to the database furthermore made the data exchanging between different UI element easier and more efficient.

F- Making all the database's info dynamic and not hard coded which make connecting to a new database easier.

**- The User Interface:**

A- Using of converters to change values in the UI in real-time to make it more dynamic.

B- Using of requesters (Interface) to loose couple windows to each other so any window can call any other window as long as it implements its requester interface (Data Exchange).

C- Using of dependency property to send data to children user controls.

D- Using of User Control for all the dashboard and small elements which allowed us to use these controls in multiples places without the need to rewrite them and with a one place for editing which had a huge effect on our work-flow while designing the UI (UserControls Folder).

E- Using MahApps theme manager to change the app mode (Dark – Light) and accent colors on the fly in runtime.

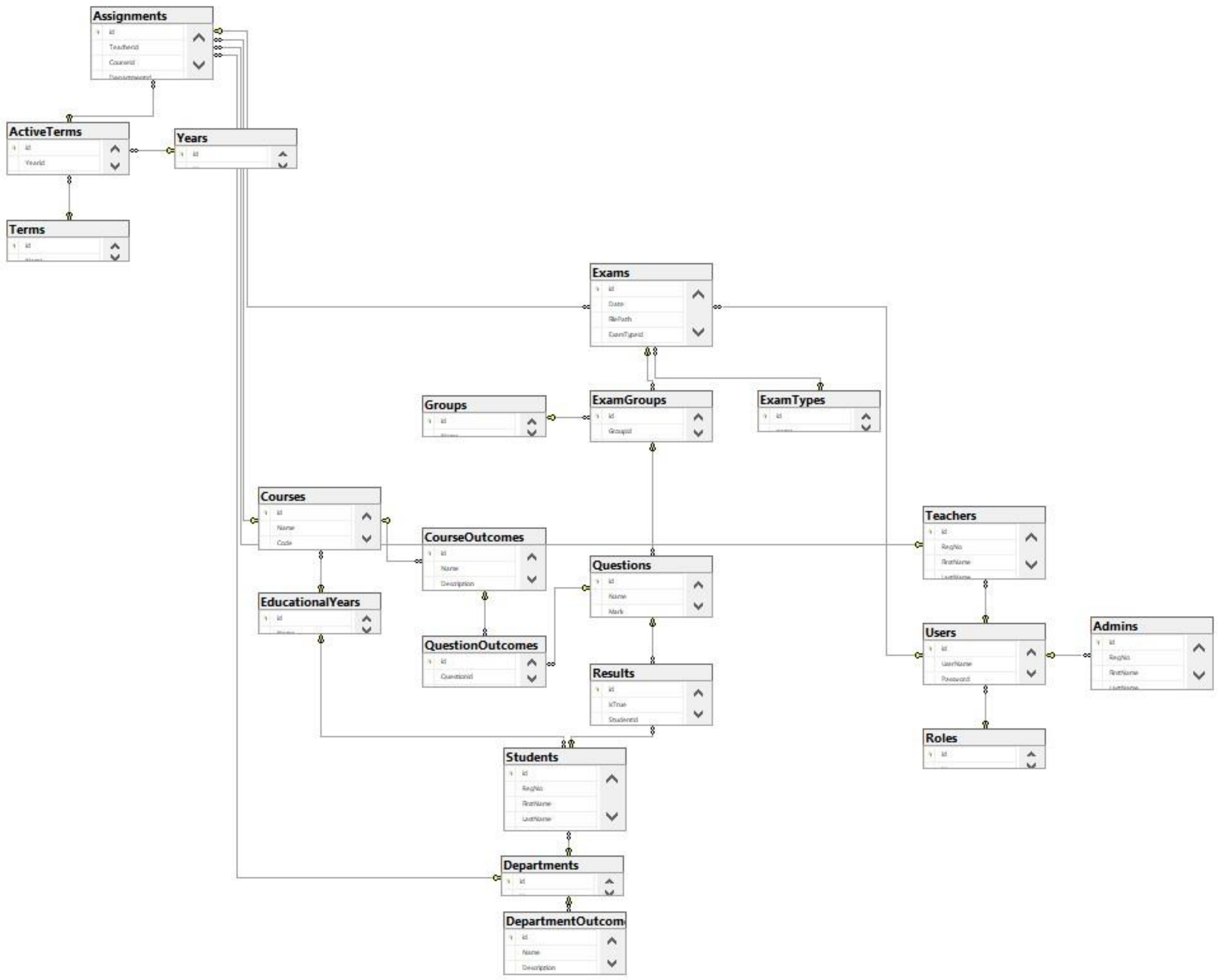
F- Using flyouts to display the Help section in a neat way.

**- Database:**

A- We used stored procedures for the added security against SQL injections attacks and for the gained speed thanks to caching.

B- We used Microsoft SQL Server Management Studio to managing our SQL server.

## ER Diagram:


Assignments
Id
TeacherId
CourseId
DepartmentId
ActiveTerms
Id
YearId
Years
Id
Terms
Id
Exams
Id
Date
FilePath
ExamTypeId
Groups
Id
ExamGroups
Id
GroupId
ExamTypes
Id
Courses
Id
Name
Code
CourseOutcomes
Id
Name
Description
Teachers
Id
RegNo
FirstName
Questions
Id
Name
Mark
EducationalYears
Id
QuestionOutcomes
Id
QuestionId
Users
Id
UserName
Password
Admins
Id
RegNo
FirstName
Results
Id
IsTrue
StudentId
Roles
Id
Students
Id
RegNo
FirstName
LastName
Departments
Id
DepartmentOutcom
Id
Name
Description