TECHNISCHE UNIVERSITÄT
CHEMNITZ

Project Report
Datenbanken und Web-Techniken
Summer Semester 2021

School Grading Application

Mohammad Tanvir Islam
Matriculation # 676614
Master in Automotive Software Engineering

Contents

1. Introduction

To gain a better understanding of test results and student performance, a school would like to switch from a paper to a digital grading system. It should also encourage instructors and students to communicate with one another (and their parents).The new system should have a good user rights management system and a lot of options for users.

The major purpose will be to provide teachers with an easy and quick way to record the results of all of their students' tests, as well as to provide an overview of tests and subjects for both students and teachers, with some rudimentary export functionality. A rudimentary communication messaging system should also be developed. A detailed task description can be found on the following pages.

2. Project: School Grading System

A backend capable of aggregating and processing data is one of the project's key requirements. The aggregated data is stored in a database during processing and can later be seen by the end-user via an interactive interface. The frontend uses a REST-based interface to ingest data from the data store. The situation is even better shown in the diagram below.

Figure-1: Project Structure

The program is made up of four parts:

1. A database for storing data.
2. A backend for processing data.
3. An API-based application programming interface for delivering data.
4. A frontend for showing data.

3. Technology Used

3.1 Oracle Database, Stored Procedures

An Oracle database is a collection of data that is processed as if it were a single entity. A database's main function is to store and retrieve relevant data. The key to tackling information management issues is a database server. In a multiuser environment, a server securely handles a huge quantity of data so that several users may access the same data at the same time. All of this is done while maintaining a high level of performance. A database server also protects against unauthorized access and offers effective recovery options in the case of a failure.

A procedure is a group of PL/SQL statements that may be named and invoked. The call specifier (also referred to as call spec) describes a java method or a languages routine of third generation to call it from SQL and PL/SQL.

3.2 RESTful API

Restful web services are online resources that can be used to obtain specific data. These administrations and services essentially demonstrate how the REST API functions. REST, or Representational State Transfer, is a web service design style or set of guidelines.

REST can be used to modify or view assets on the server without having to perform any server-side operations. The customer requests an asset from the server, and the server responds (if there are no mistakes). The reaction is a representation of the resource that is now available on the server. It could be a JSON, XML, PDF, or DOC file, for example.

3.3 Nodejs

Node.js is a Chrome JavaScript runtime platform that makes it easy to develop fast, scalable network apps. Node.js utilizes an unblocking event based I/O architecture, making it light and powerful, ideal for retail applications running over dispersed devices with extensive data-intensive real-time.

Node.js is a cross-platform runtime environment for building server-side and networking applications that is free source. Node.js apps are written in JavaScript and run on OS X, Microsoft Windows, and Linux using the Node.js runtime.

Node.js also comes with a large library of JavaScript modules, which greatly facilitates the creation of Node.js web applications.

3.4 JSON

JSON (JavaScript Object Notation) is a data-exchange format that is simple to use. Reading and writing are simple tasks for humans. Machines can easily parse and produce it. It is based on a subset of the ECMA-262 3rd Edition - December 1999 JavaScript Programming Language Standard. JSON is a language-independent text format that employs standards known to programmers of the C family of languages, such as C, C++, C#, Java, JavaScript, Perl, Python, and many more. JSON is an appropriate data-transfer language because of these characteristics.

3.5 Angular

Angular is a TypeScript-based open-source JavaScript framework. It's run by Google, and its major goal is to create single-page apps. Angular has evident advantages as a framework, and it also provides a common structure for developers to work with. It allows users to develop big applications that are easy to manage.

3.6 JQuery

jQuery is a JavaScript library that allows you to "write less, do more." The goal of jQuery is to make using JavaScript on your website much easier. jQuery wraps a number of typical activities that require a lot of lines of JavaScript code into methods that can be called with only a single line of code.

3.7 Bootstrap

The most popular HTML, CSS, and JavaScript framework for creating a responsive website is Bootstrap. It is available for download and usage free of cost. It's a front-end framework that makes web development easier and faster. It contains design templates for typography, forms, buttons, tables, navigation, modals, picture carousels, and more, all built with HTML and CSS. It makes it easier to make responsive designs.

3.8 HTML

HTML (Hypertext Markup Language) is the most widely used markup language for creating web pages and other data that can be seen in any web browser. Inside the web page content, HTML is written as HTML components with labels contained in edge sections (like html>). Most HTML labels are two by two, such as h1> and /h1>. When all other factors are equal, HTML components create the structure squares. HTML allows you to incorporate photos and items and can be used to create intuitive structures. It enables the creation of well-organized reports by interpreting auxiliary semantics for text, such as headers, paragraphs, lists, quotes, links, and other elements. It can include text written in dialects such as JavaScript and Cascading Style Sheets, which have an impact on the behavior of HTML pages. The first advantage of HTML is that it is widely used. The HTML language is supported by each program. It's easy to pick up and use. It is, of course, included in each window, so there is no need to purchase additional software.

3.9 ASP.NET

ASP.NET is a server-side web application framework that is open-source. Microsoft invented it at the turn of the century, and it runs on Windows. Developers may use ASP.NET to build online apps, web services, and dynamic content-driven websites. It's used to make HTML5, CSS, and JavaScript-based solutions that are simple, quick, and scalable to a large number of people.

4. System Requirement

- Oracle 12c Database
- Minimum 1GB Ram
- Windows 8/10 64 Bit OS
- NodeJS 14.17.1

5. Designs

5.1 Use Case

A use case portrays how a client utilizes a framework to achieve a specific objective. A use case chart comprises of the system, the related use cases and actors and related to these to one another to picture: what is being depicted? (System), who is utilizing the framework? (Actors) and what would the actors like to accomplish? (Use cases), subsequently, use cases help guarantee that the right framework is created by catching the necessities from the client's perspective.

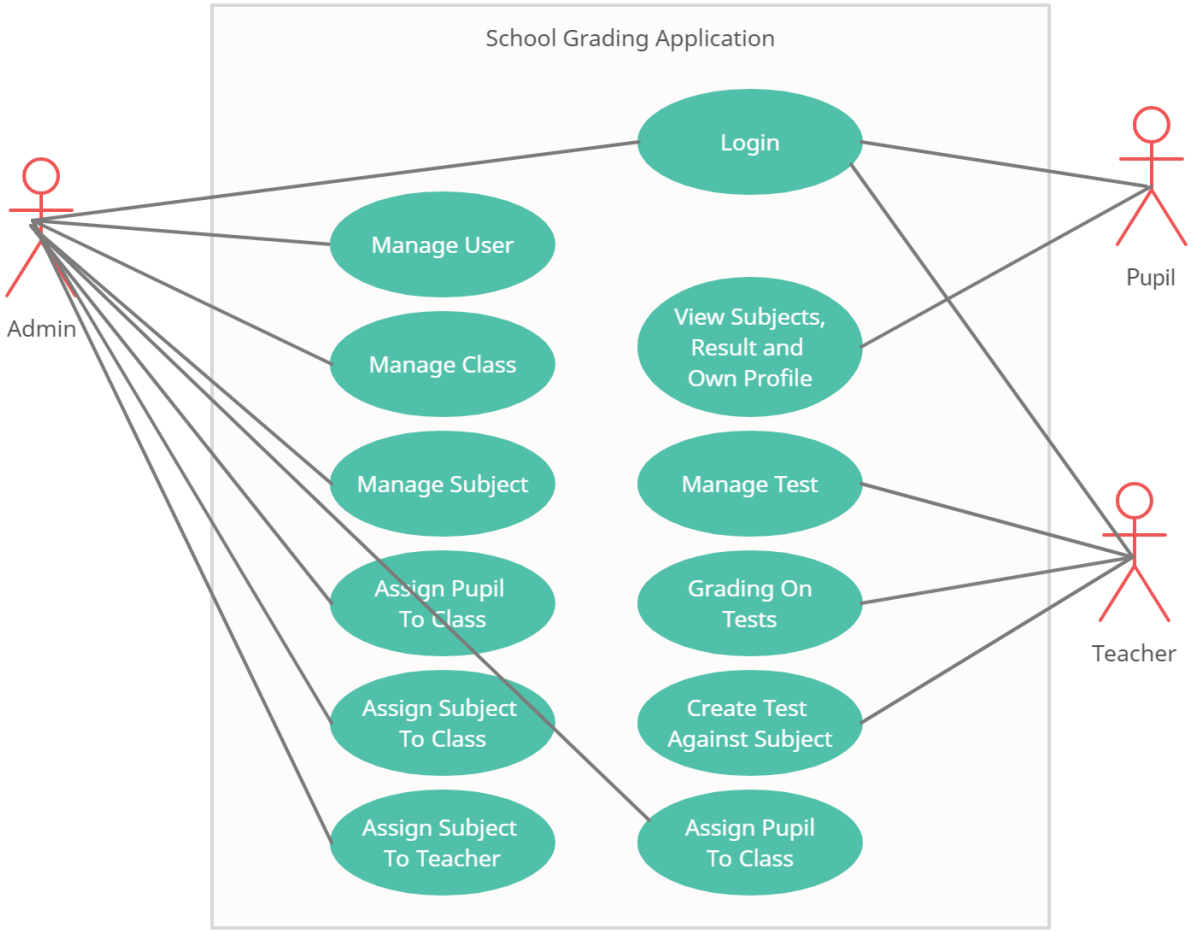

Figure-2: Use Case of School Grading Application

5.2 Project Architecture

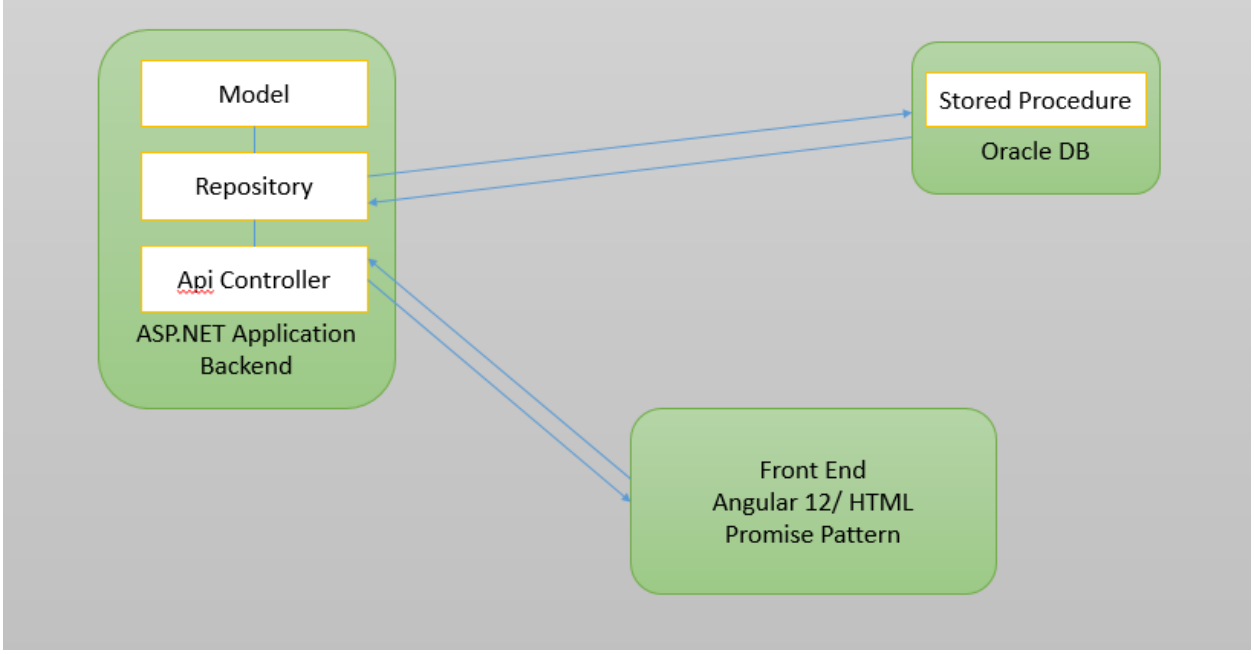

Figure-3: Project Structure

The above diagram depicts the **M**odel, **R**epository and **C**ontroller to the AP.NET high level architecture. The controller is an Api Controller implemented with a Asp.Net WebApi 2, the model is implemented with Repository, which calls database stored procedure, and in stored procedure, all business logics are implemented. This is the architectural overview of the Asp.Net WebApi2 Repository pattern.

5.3 Entity Relationship Diagram (ERD):

The Entity Relationship Diagram (ERD) or Entity Relationship Diagram (ERD) is a type of structural diagram used in database schema. The Entity Relation model is based on the concept of major actual elements inside the system scope and their relationships.

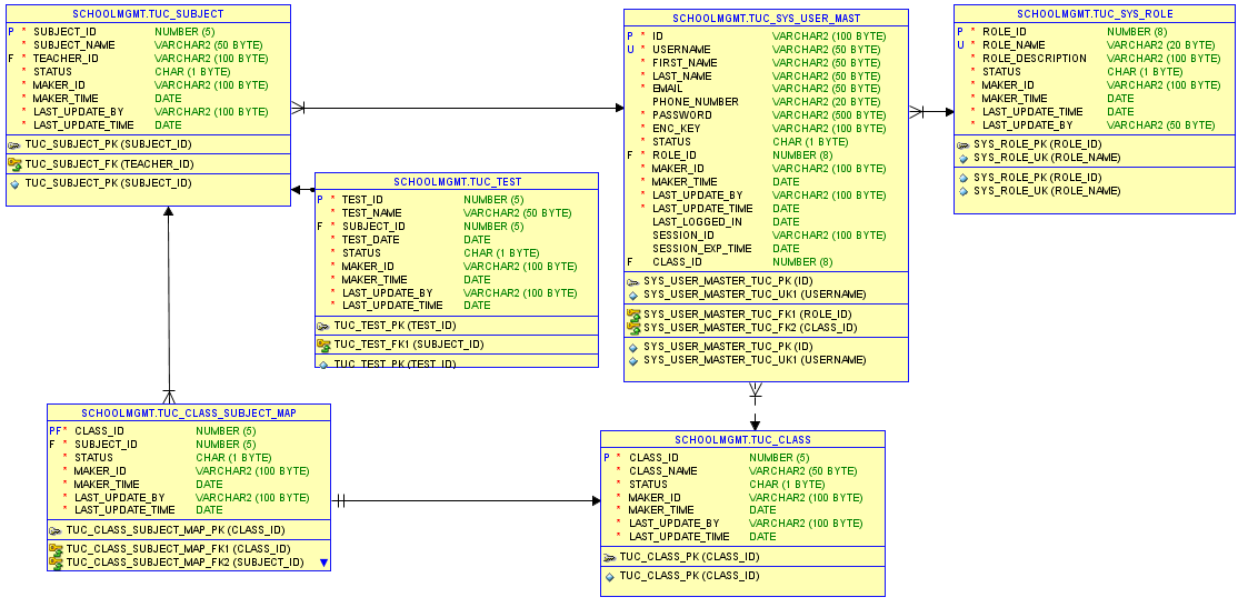

Figure-3: ERD of School Grading Application

6. References

[1] Richter D. (2021. May 31). "DBW Project" [PDF File]. Retrieved from https://www.tu-chemnitz.de/informatik/DVS/lehre/DBW/Projekt/DBW%20Project.pdf

[2] Richter D. (2021. June 25). "Project task questions and responses in Datenbanken und Web-Techniken 2021" [Text File]. Retrieved from https://www.tu-chemnitz.de/informatik/DVS/lehre/DBW/Projekt/DBW%20Project%20QA.txt

[3] OracleDatabase. (2021, June 28). Retrieved from https://en.wikipedia.org/wiki/Oracle_Database

[4] What is NodeJS? Executive Summary. (2021, June 28). Retrieved from https://nodejs.org/en/about/

[5] What is angular? (2021, June 28). Retrieved from https://angular.io/guide/what-is-angular

[6] What is RESTful ?. (2021, June 28). Retrieved from https://restfulapi.net/

[7] What is ASP.NET WebApi2? (2021, June 28). Retrieved from https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/tutorial-your-first-web-api

[8] jQuery. (2021, June 28). Retrieved from https://jquery.com/

[9] What is Stored Procedures?. (2021, June 28). Retrieved from https://searchoracle.techtarget.com/definition/stored-procedure

[10] ASP.NET-CORS. (2021, June 28). Retrieved from https://www.c-sharpcorner.com/article/enable-cors-in-asp-net-webapi-2/

[11] Requests: HTTP for Humans™. (2021, June 28). Retrieved from https://requests.readthedocs.io/en/master/

[12] Bootstrap (front-end framework). (2021, June 28). Retrieved from https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)

[13] Font Awesome (front-end framework). (2021, June 28). Retrieved from https://en.wikipedia.org/wiki/Font_Awesome

[14] DataTables | Table plug-in for jQuery (2021, June 28). Retrieved from https://datatables.net/

7. Appendix

Stored Procedures:

The PL/SQL language of the Oracle database consists of stored procedures used to create applications in databases. IT specialists write and test code by utilizing stored programs in the Oracle database, which are then compiled into stored methods. The fundamental structure of the PL/SQL block, including declarative, executable and exception handling components, is followed by a stored procedure in Oracle.

```
procedure sp_tuc_sys_user_mast_i(p_activity     in char,
                                 p_username     in tuc_sys_user_mast.username%type,
                                 p_first_name   in tuc_sys_user_mast.first_name%type,
                                 p_last_name    in tuc_sys_user_mast.last_name%type,
                                 p_email        in tuc_sys_user_mast.email%type,
                                 p_phone_number in tuc_sys_user_mast.phone_number%type,
                                 p_password     in tuc_sys_user_mast.password%type,
                                 p_role_id      in tuc_sys_user_mast.role_id%type,
                                 p_user_id      in tuc_sys_user_mast.maker_id%type,
                                 p_out          out number,
                                 p_err_code     out varchar2,
                                 p_err_msg      out varchar2);
procedure sp_tuc_sys_user_mast_gk(p_username in nvarchar2,
                                  p_user_id  in nvarchar2,
                                  p_out      out number,
                                  p_err_code out nvarchar2,
                                  p_err_msg  out nvarchar2,
                                  T_CURSOR   out sys_refcursor);

procedure sp_tuc_sys_user_mast_ga(p_username  in nvarchar2,
                                  p_user_type in number,
                                  p_user_id   in nvarchar2,
                                  p_out       out number,
                                  p_err_code  out nvarchar2,
                                  p_err_msg   out nvarchar2,
                                  T_CURSOR    out sys_refcursor);
```

```
procedure sp_sys_verify_user(p_username in nvarchar2,
                             p_password in nvarchar2,
                             p_out      out number,
                             p_err_code out nvarchar2,
                             p_err_msg  out nvarchar2,
                             T_CURSOR   out sys_refcursor);

procedure sp_sys_change_password(p_username     in nvarchar2,
                                 p_password_old in nvarchar2,
                                 p_password_new in nvarchar2,
                                 p_user_id      in nvarchar2,
                                 p_out          out number,
                                 p_err_code     out nvarchar2,
                                 p_err_msg      out nvarchar2);

procedure sp_sys_reset_password(p_username  in nvarchar2,
                                p_password  in out nvarchar2,
                                p_out_email out nvarchar2,
                                p_user_id   in nvarchar2,
                                p_out       out number,
                                p_err_code  out nvarchar2,
                                p_err_msg   out nvarchar2);

procedure sp_pass_encrypt(p_user_id  in varchar2,
                          p_password in varchar2,
                          p_enc_key  in varchar2,
                          p_enc_pass out varchar2);

procedure sp_check_session(p_session_id in TUC_SYS_USER_MAST.session_id%type,
                           p_username   out TUC_SYS_USER_MAST.username%type,
                           p_role_name  out tuc_sys_role.role_name%type,
                           p_out        out number,
                           p_err_code   out nvarchar2,
                           p_err_msg    out nvarchar2);

function is_session_expired(p_username in varchar2) return number;

procedure sp_tuc_sys_role_ga(p_user_id  in nvarchar2,
                             p_out      out number,
                             p_err_code out nvarchar2,
                             p_err_msg  out nvarchar2,
                             T_CURSOR   out sys_refcursor);

procedure sp_class_subject_map(p_activity   in char,
                               p_subject_id in tuc_class_subject_map.subject_id%type,
                               p_class_id   in tuc_class_subject_map.class_id%type,
                               p_user_id    in TUC_SYS_USER_MAST.maker_id%type,
                               p_out        out number,
                               p_err_code   out varchar2,
                               p_err_msg    out varchar2);
```

```
procedure sp_tuc_test(p_activity   in char,
                      p_test_id    in out tuc_test.test_id%type,
                      p_test_name  in tuc_test.test_name%type,
                      p_subject_id in tuc_test.subject_id%type,
                      p_test_date  in tuc_test.test_date%type,
                      p_user_id    in tuc_test.maker_id%type,
                      p_out        out number,
                      p_err_code   out varchar2,
                      p_err_msg    out varchar2);

procedure sp_tuc_test_gk(p_test_id  in nvarchar2,
                         p_user_id  in nvarchar2,
                         p_out      out number,
                         p_err_code out nvarchar2,
                         p_err_msg  out nvarchar2,
                         T_CURSOR   out sys_refcursor);

procedure sp_tuc_test_ga(p_in         in nvarchar2,
                         p_subject_id number,
                         p_user_id    in nvarchar2,
                         p_out        out number,
                         p_err_code   out nvarchar2,
                         p_err_msg    out nvarchar2,
                         T_CURSOR     out sys_refcursor);
procedure sp_tuc_class(p_activity   in char,
                       p_class_id   out tuc_class.class_id%type,
                       p_class_name in tuc_class.class_name%type,
                       p_user_id    in tuc_class.maker_id%type,
                       p_out        out number,
                       p_err_code   out varchar2,
                       p_err_msg    out varchar2);
procedure sp_tuc_class_gk(p_class_id in nvarchar2,
                          p_user_id  in nvarchar2,
                          p_out      out number,
                          p_err_code out nvarchar2,
                          p_err_msg  out nvarchar2,
                          T_CURSOR   out sys_refcursor);
procedure sp_tuc_class_ga(p_in       in nvarchar2,
                          p_user_id  in nvarchar2,
                          p_out      out number,
                          p_err_code out nvarchar2,
                          p_err_msg  out nvarchar2,
                          T_CURSOR   out sys_refcursor);
procedure sp_tuc_subject(p_activity     in char,
                         p_subject_id   out tuc_subject.subject_id%type,
                         p_subject_name in tuc_subject.subject_name%type,
                         p_teacher_id   in tuc_sys_user_mast.id%type,
                         p_user_id      in tuc_subject.maker_id%type,
                         p_out          out number,
                         p_err_code     out varchar2,
                         p_err_msg      out varchar2);
procedure sp_tuc_subject_gk(p_subject_id in nvarchar2,
                            p_user_id    in nvarchar2,
                            p_out        out number,
                            p_err_code   out nvarchar2,
                            p_err_msg    out nvarchar2,
                            T_CURSOR     out sys_refcursor);
procedure sp_tuc_subject_ga(p_in       in nvarchar2,
                            p_user_id  in nvarchar2,
                            p_out      out number,
                            p_err_code out nvarchar2,
                            p_err_msg  out nvarchar2,
                            T_CURSOR   out sys_refcursor);
```

API Documentation:

RESOURCES	END POINTS
Login	POST api/Login
User	POST api/User/Register
	POST api/User/UpdateUser
	GET api/User/GetUserList?userType={userType}
	GET api/User/Get?id={id}
Class	GET api/Class/GetClassList
	GET api/Class/GetClassInfo?id={id}
	POST api/Class/ModifyClassInfo
	POST api/Class/AddNewClass
	DELETE api/Class/DeleteClass?id={id}
Subject	GET api/Subject/GetSubjectList
	GET api/Subject/GetSubjectList?classId={classId}
	GET api/Subject/GetSubjectInfo?id={id}
	POST api/Subject/ModifySubjectInfo
	POST api/Subject/AddNewSubject
	DELETE api/Subject/DeleteSubject?id={id}
	POST api/Subject/ClassSubjectMap
	DELETE api/Subject/ClassSubjectMapRemove?classId={classId}&subjectId={subjectId}
Test	GET api/Test/GetTestList?subjectId={subjectId}
	GET api/Test/GetTestInfo?testId={testId}
	POST api/Test/ModifyTestInfo
	POST api/Test/AddNewTest
	DELETE api/Test/DeleteTest?id={id}