Software Requirements Specification

for

**Distributed computer system for student project management**

**Version 1.0 approved**

**Prepared by**

**Milcu Cristian-Mihai**

**Muga Bogdan-Marius**

**Ratoiu Cristina-Irina**

**Facultatea de Automatica, Calculatoare si Electronica**

**08.03.2021**

**Table of Contents**

**Table of Contents [ii](#_30j0zll)**

**Revision History [ii](#_1fob9te)**

**1. Introduction [1](#_3znysh7)**

1.1 Purpose [1](#_2et92p0)

1.2 Document Conventions [1](#_tyjcwt)

1.3 Intended Audience and Reading Suggestions [1](#_3dy6vkm)

1.4 Product Scope [1](#_1t3h5sf)

1.5 References [1](#_4d34og8)

**2. Overall Description [2](#_2s8eyo1)**

2.1 Product Perspective [2](#_17dp8vu)

2.2 Product Functions [2](#_3rdcrjn)

2.3 User Classes and Characteristics [2](#_26in1rg)

2.4 Operating Environment [2](#_lnxbz9)

2.5 Design and Implementation Constraints [2](#_35nkun2)

2.6 User Documentation [2](#_1ksv4uv)

2.7 Assumptions and Dependencies [3](#_44sinio)

**3. External Interface Requirements [3](#_2jxsxqh)**

3.1 User Interfaces [3](#_z337ya)

3.2 Hardware Interfaces [3](#_3j2qqm3)

3.3 Software Interfaces [3](#_1y810tw)

3.4 Communications Interfaces [3](#_4i7ojhp)

**4. System Features [4](#_2xcytpi)**

4.1 System Feature 1 [4](#_1ci93xb)

4.2 System Feature 2 (and so on) [4](#_3whwml4)

**5. Other Nonfunctional Requirements [4](#_2bn6wsx)**

5.1 Performance Requirements [4](#_qsh70q)

5.2 Safety Requirements [5](#_3as4poj)

5.3 Security Requirements [5](#_1pxezwc)

5.4 Software Quality Attributes [5](#_49x2ik5)

5.5 Business Rules [5](#_2p2csry)

**6. Other Requirements [5](#_147n2zr)**

**Appendix A: Glossary [5](#_3o7alnk)**

**Appendix B: Analysis Models [5](#_23ckvvd)**

**Appendix C: To Be Determined List [6](#_ihv636)**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this document is to give a detailed description of the requirements for the “Distributed computer system for student project management” software. It will illustrate the purpose and complete declaration for the development of the system. It will also explain system constraints, interface and interactions with other external applications.

## Document Conventions

User- someone who interacts with the application

UI – user interface

camel Case(camelCase) - naming conventions for variables such that the first letter of the first word is lowercase and the other words are upper cases(rest of the letters are lowercase)

Pascal Case(PascalCase) - naming conventions for variables such that the first letter of every word is uppercase(other letters are lowercase)

underscore Prefix(\_underscore) - naming conventions for variables such that before the camelCase convention it is a \_(\_underscore) symbol

## Intended Audience and Reading Suggestions

The application was created in order to facilitate the transmission of documents and communication between students and teachers. Having educational purposes, the application document is recommended to be read by students, teachers and developers who deal with the implementation of the application.

## Product Scope

The system will allow students to submit their projects online to teachers (upload facility) and teachers to evaluate them. The computer system comprises an interface with the user (web browser or dedicated application), a database server that will store information about students, students, grades, application server that manages students, projects, notes and a Web server. The user has access to the service only through the interface with the user. The system will provide facilities to students and teachers after they successfully authenticate [password and identifier] (the identifier and password are already considered generated):

- viewing account information

- password change

- uploading projects by the student and entering the required information

- the teacher's visualization of the project files and the afferent information, date,

student name

- giving grades and comments by teachers

Depending on the type of authentication (student, teacher) the user will be allowed certain

functionalities.

## References

Udemy-Angular8-the Complete Guide

<https://angular.io/>

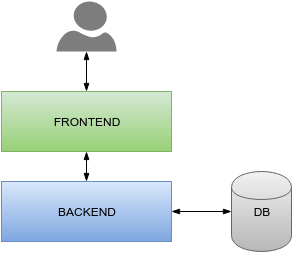
<https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-5.0>

<https://www.c-sharpcorner.com/UploadFile/8a67c0/C-Sharp-coding-standards-and-naming-conventions/>

<https://angular.io/guide/styleguide>

# Overall Description

## Product Perspective



The application is based on three tier architecture.

The front end (presentation tier) is the highest level of the application and manages the UI. Our application has login methods and upload documents ,post comments. This information will be stored in the database together with the data about students and teachers through the logic tier.

The logic tier coordinates the application, the order processes, makes logical decisions and evaluations and makes calculations. The main role being to transfer and move information between the 2 surrounding layers.

The data tier stores and retrieves the information from a database or file system.Here is stored the information about users, notes, documents, comments.The data transfer is made through the logic tier back to the UI.

## Product Functions

With this application the users are able to connect to a platform which improves the study quality.

The main users will be students and teachers . They are going to access the platform using different credentials connected to the email address already generated by the faculty.

When a user connects as a teacher he is able to post assignments and evaluate students' work , he can post comments and upload documents related to the subject.

The student-user can see the assignments and upload a document as answer and address questions if something isn’t clear.

This application allows multiple classes to be created . When the purpose for which a class was created no longer exists, it can be deleted

## User Classes and Characteristics

The users , as already described above ,should be teachers and students. Often, due to the large number of students participating in a course, project / homework management is difficult.

That is why this application was created to facilitate the retention and management of student projects by their teachers.

Thus the loss and damage of documents can be substantially reduced.

## Operating Environment

The application consists of three main components: a web client, a web application server and a database server.

For the web client, any chromium-based web browser will work, as well as Firefox and Safari.

The web application server is built using the Windows 10 operating system, ASP.NET and IIS (Internet information services).

The database server uses Microsoft’s SQL Server Management Studio as a database management system (RDBMS).

## Design and Implementation Constraints

The application works only online. In this case, the only constraint is a stable internet connection.

## User Documentation

A user manual will be delivered along with the software, containing all the necessary data about application uses for both kinds of user.

## Assumptions and Dependencies

We assume that the accounts for both teachers and students are already registered in our database.

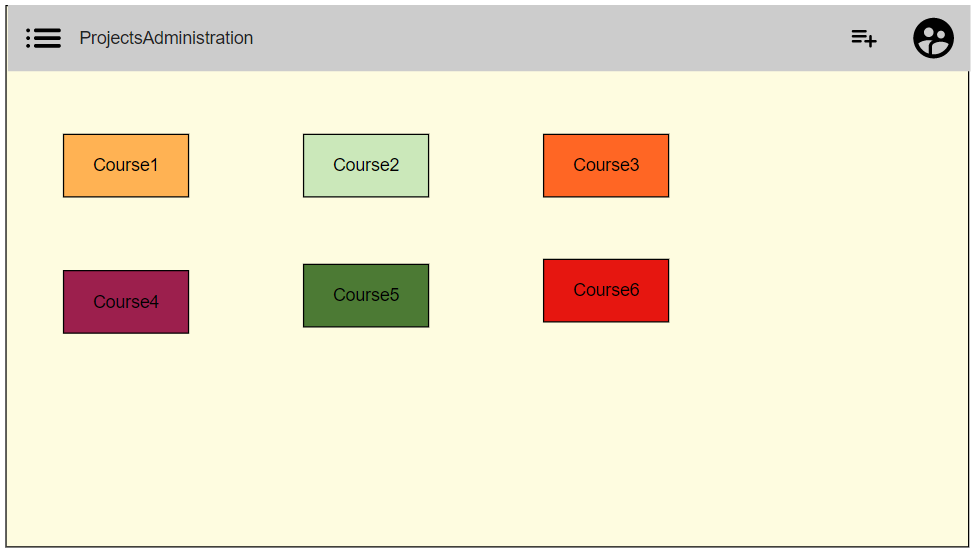
# External Interface Requirements

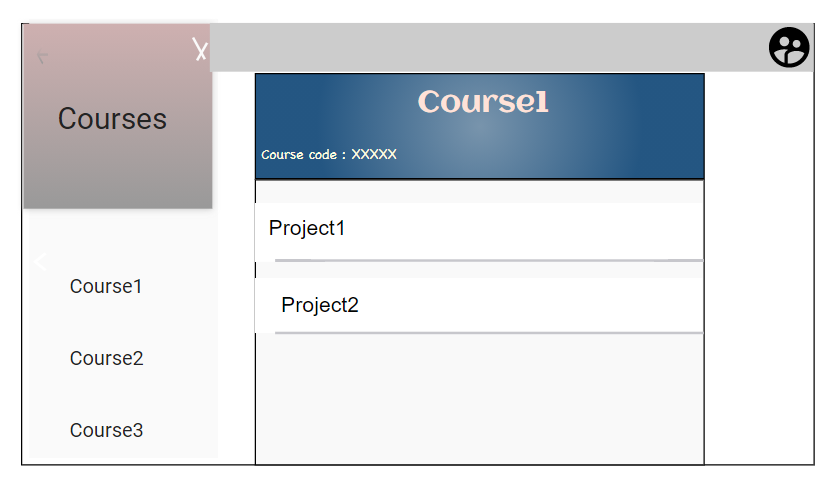
## User Interfaces

Login page:

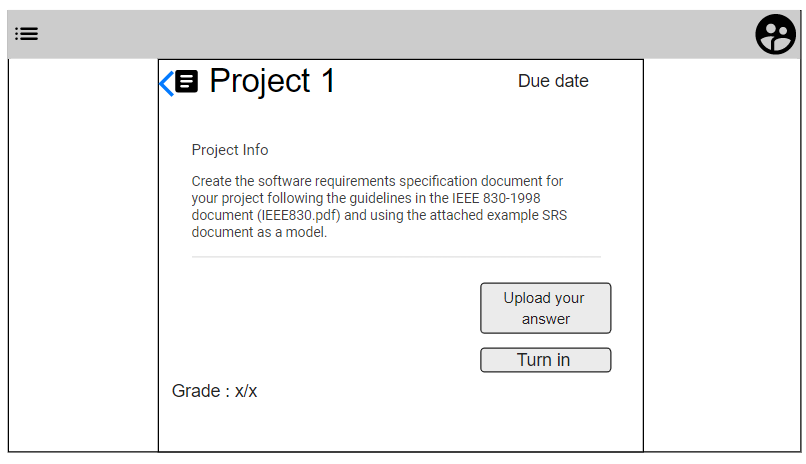


## Dashboard:

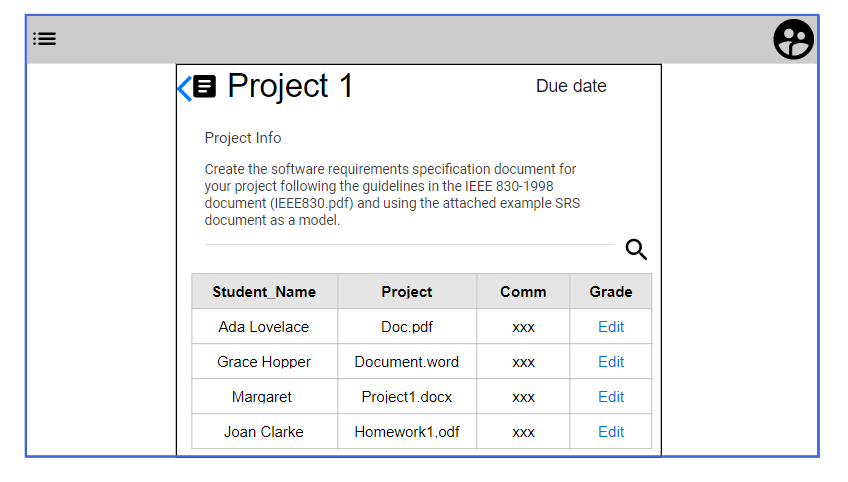




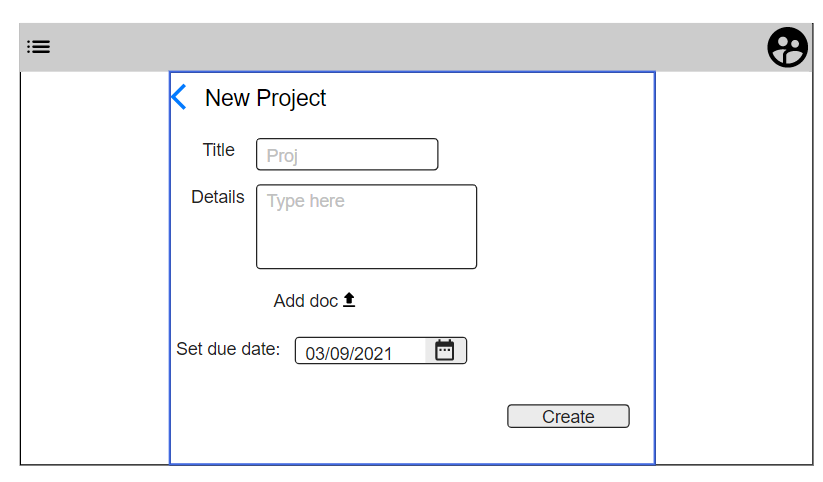
Project page ~ student view :



Project page ~ teacher view :



Project creation :



## Hardware Interfaces

No extra hardware interfaces needed for this application.

## Software Interfaces

Operating System: Windows 7 or better.

MacOS starting with 10.5.

## Communications Interfaces

The communication interface will have a standard client-server model. This model is used on web browser on a HTTP domain.

# System Features

## Login screen

4.1.1 Description and Priority

High priority. This screen should ensure secure access to our application.

4.1.2 Stimulus/Response Sequences

The user can enter the credentials associated with an existing account and it is prompted if the credentials were incorrect. If the credentials are correct the user is logged in.

4.1.3 Functional Requirements

REQ-1: The user can insert the email and the password.

REQ-2: The user’s password is hidden.

REQ-3: The focus is set on the email input field when accessing the webpage.

REQ-4: The user is prompted if the inserted credentials are incorrect.

REQ-5: After log in, the user is given to the rights according to the type of account (teacher or student).

## Student Dashboard

4.2.1 Description and Priority

High priority. Page where the student user is redirected after logging in the application. The dashboard contains courses, the student is enrolled in.

4.2.2 Stimulus/Response Sequences

Using this page, the student can access the courses or access the account information.

4.2.3 Functional Requirements

REQ-1: Courses are represented as a grid of tiles.

REQ-2: Students can access the courses by clicking on course tiles.

REQ-3: Navbar element at the top of the screen containing the application title.

REQ-4: Navbar button that opens a hamburger menu. Using this, the student can view the courses as a list.

REQ-5: Navbar button for enrolling. A student can enroll in a course by using the course's unique code.

REQ-6: Navbar button for accessing the account details.

REQ-7: Course tiles contain a button for “unrolling” from a course.

REQ-8: Modal for course adding via unique code.

REQ-9: Course adding modal focus on opening.

## Teacher Dashboard

4.3.1 Description and Priority

High priority. Page where the teacher is redirected after logging in the application. The dashboard contains the courses the teacher holds.

4.3.2 Stimulus/Response Sequences

Using this page, the teacher can access and manage the courses he holds.

4.3.3 Functional Requirements

REQ-1: Courses are represented as a grid of tiles.

REQ-2: Teachers can access the courses by clicking on course tiles.

REQ-3: Navbar element at the top of the screen containing the application title.

REQ-4: Navbar button that opens a hamburger menu. Using this, the student can view the courses as a list.

REQ-5: Navbar button for enrolling. A student can enroll in a course by using the course's unique code.

REQ-6: Navbar button for accessing the account details.

REQ-7: Navbar button for creating a new course.

REQ-8: Modal for course adding via title.

REQ-9: Course adding modal focus on opening.

## Account information screen

4.4.1 Description and Priority

Medium priority. Page where the student or teacher user can modify their settings on the account, such as changing password.

4.4.2 Stimulus/Response Sequences

Using this page, the user can view and edit the account information.

4.4.3 Functional Requirements

REQ-1: The user can see the account information.

REQ-2: The user needs to confirm their current password to be able to change it.

REQ-3: The user needs to change to the desired password.

REQ-4: Save button for changed information.

## Student course detail screen

4.5.1 Description and priority

High priority. Page where the student can see the course assignments in form of a list.

4.5.2 Stimulus/Response Sequences

The student can access the assignments detail screen by clicking on items inside the assignment list.

4.5.3 Functional Requirements

REQ-1: The student can see the assignments as items inside a list.

REQ-2: Each item of the list will contain assignment title and the due date.

REQ-3: The due date is replaced by a warning badge if the assignment is overdue.

## Teacher course detail screen

4.6.1 Description and priority

High priority. Page where the teacher can manage assignments.

4.6.2 Stimulus/Response Sequences

The teacher can see, add, edit and delete the assignments.

4.6.3 Functional Requirements

REQ-1: The teacher can see the assignments as a list.

REQ-2: The teacher can add a new assignment via modal.

REQ-3: Focus on modal “assignment name” field.

REQ-4: The teacher can add files via modal.

REQ-5: The teacher can add a due date.

REQ-6: The teacher can edit an assignment.

REQ-7: The teacher can delete an assignment.

## Student assignment detail screen

4.7.1 Description and priority

High priority. Page containing assignment information available for students.

4.7.2 Stimulus/Response Sequences

The student can see the assignment details, download files uploaded by the teacher and upload their assignment work.

4.7.3 Functional Requirements

REQ-1: The students can see the assignment title, and description.

REQ-2: The students can download the files uploaded by the teachers.

REQ-3: The students can upload their assignment (1 file limit).

REQ-4: The students can see their grade.

REQ-5: The students can see remarks left by teachers regarding their homework.

REQ-6: Button for submitting the files.

## Teacher assignment detail screen

4.8.1 Description and priority

High priority. Page containing assignment information available for teachers.

4.8.2 Stimulus/Response Sequences

The teacher can see assignment details, can see the students assignment work, can grade works and leave comments.

4.8.3 Functional requirements

REQ-1: The teachers should be able to see the assignment title and description.

REQ-2: The teachers should be able to download the files they have uploaded.

REQ-3: The teachers should see students' assignment works in a list.

REQ-4: The teachers should be able to grade students' work.

REQ-5: The teachers should be able to set a remark to the students' work.

# Other Nonfunctional Requirements

## Performance Requirements

Windows: 7 or better.

MacOS starting with 10.5.

Processor: 2.0 GHz Dual Core Processor

RAM: 2 GB or more

## Safety Requirements

All the data will be stored in a database from the server, so no data will be lost in the process.

## Security Requirements

On the login page the password will be hidden so it doesn’t get affected by third-party users. The password will be stored into the database.

## Software Quality Attributes

Developers will have to follow some rules and conventions for angular such that other developers understand the code for the front-end and can be found on the following link.

https://angular.io/guide/styleguide

Few angular rules and conventions that developers should follow are:

* modular programming such as doing a specific thing per file
* it’s easier to understand and reuse the code text when it is less lines of code(<400 lines)
* makes debugging much easier
* small functions makes the code easier to read(<75 lines)
* small functions makes the code easier to understand
* small functions makes the code easier to test/debug
* small functions makes the code easier to reuse for other developers
* names of the variables/files are simple and suggestive for further reuse of the code/files

Developers will have to follow some rules and conventions for C# such that other developers understand the code much easier, using the following link.

<https://www.c-sharpcorner.com/UploadFile/8a67c0/C-Sharp-coding-standards-and-naming-conventions/>

Few C# rules and conventions that developers should follow are:

* use native data types names, such as **int** instead of **Int32,** **Int64.**
* use **camelCase** method for local variables and method arguments
* do not use abbreviations in naming the variables
* use **PascalCase** for classes and method names
* use **camelCase** method with underscore(\_) for private member variables(private int \_result = 100)
* use **PascalCase** method for public member variables(public int Result = 100)

## Business Rules

In this web application teachers and students have different rights to access the same web page.

On the dashboard menu the teacher can create a class for students that will lately join the class with an invite made for students. The students have the option to join the class with the invitation given by their teachers and can disable the class if they finish it.

After the class is made(a new web page has been created), the teacher has the option to create new assignments for their students with a deadline and will give the students grades based on their work attached to the assignment. The students will then be able to attach files or documents to the desired assignment and lately will obtain the grade given by the teacher.

# Other Requirements

TBD

**Appendix A: Glossary**

**Appendix B: Analysis Models**

**Appendix C: To Be Determined List**