**CS673 Software Engineering** 

**Team 4 - Project Name**

**Software Design Document**

|  |  |  |  |
| --- | --- | --- | --- |
| Team Member | Role(s) | Signature | Date |
| Connor Richmond | Team Leader | *car* |  |
| Zak Kysar | Back up lead/Reqs | *zak* |  |
| Juan Sanchez | QA | *JPS* |  |
|  | Config |  |  |
| Rattikarn Dudley | Security | *Rk* |  |
| Molla Negash | Design | *MN* |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Author** | **Date** | **Change** |
| **1** | **Car** | **10/1** | **Added project overview and algorithms section** |
| **2** | **JPS** | **10/2** | **Added ER and Relational Schema under design patters** |

[Introduction](#_87t9hln2vjz0)

[Software Architecture](#_buttcq9i221r)

[Design Patterns](#_x18fj36s1121)

[Key Algorithms](#_mtfbusfb0eq3)

[Classes and Methods](#_7ucksmkf6rzx)

[References](#_15tmymhipvdv)

[Glossary](#_8n34lvocupub)

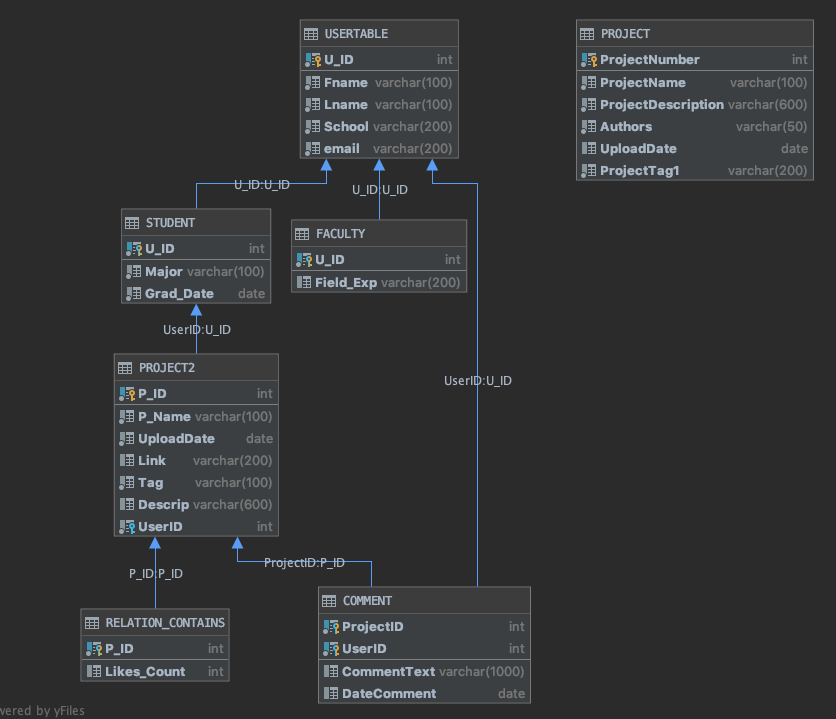
# Introduction

Our software system will provide search and archiving solutions for computer science based projects. We’re implementing a website that allows users to upload their own projects and search through those uploaded by others. We have an ER diagram for our database structure that stores all project info and authors. We will be implementing basic search functionality around project keywords that are manually input and also extracted from project titles and descriptions. All project information will be stored in a SQL Server database that will be viewable and searchable from project search and view pages. The system will be designed to be scalable to enable bug free project hosting as the number of projects stored increases moving forward.

# Software Architecture

*In this section, you will describe the decomposition of your software system, which include each component (which may be in terms of package or folder) and the relationship between components. You shall have a diagram to show the whole architecture, and class diagram for each component. The interface of each component and dependency between components should also be described. If any framework is used, it shall be defined here too. Database design should also be described if used.*

* Code Framework:
  + Website is hosted on a microsoft server.
  + HTML pages on the front end.
    - Where pages require back-end code, we’ll use .cshtml pages that have c# before the header of the html parent tag. These are known as Razor files.
  + To assist front end, and potentially back end, we’ll play around with Javascript and JQuery.
  + We’ll use CSS to style the html pages.
    - We’ll import Bootstraps css page on each html page to import their grid layout in css to make the site responsive.
* Components:
  + HTML Forms
  + SQL Server
  + CSS Styles
  + JavaScript navigation bar.



# Design Patterns

(In this section, you shall describe any design patterns used in your software system.)

The team has implemented structural design patterns at the database design level to organize two different types of users, which can be students and faculty. Users will have their respective distinctive attributes at the subclass level (STUDENT and FACULTY), and common attributes at the super class level USER

We are using the layered approach, the three core layers. Each layer provides a service to other layers. Each .cshtml file has these three layers. If a page does not require sql data, it will be a .html file, which only includes the first layer, presentation. It provides an easy way of writing well-organized and testable code.

1. Presentation layer (Users Browser):

a. HTML

b. Java Script

c. Jquery

d. Css

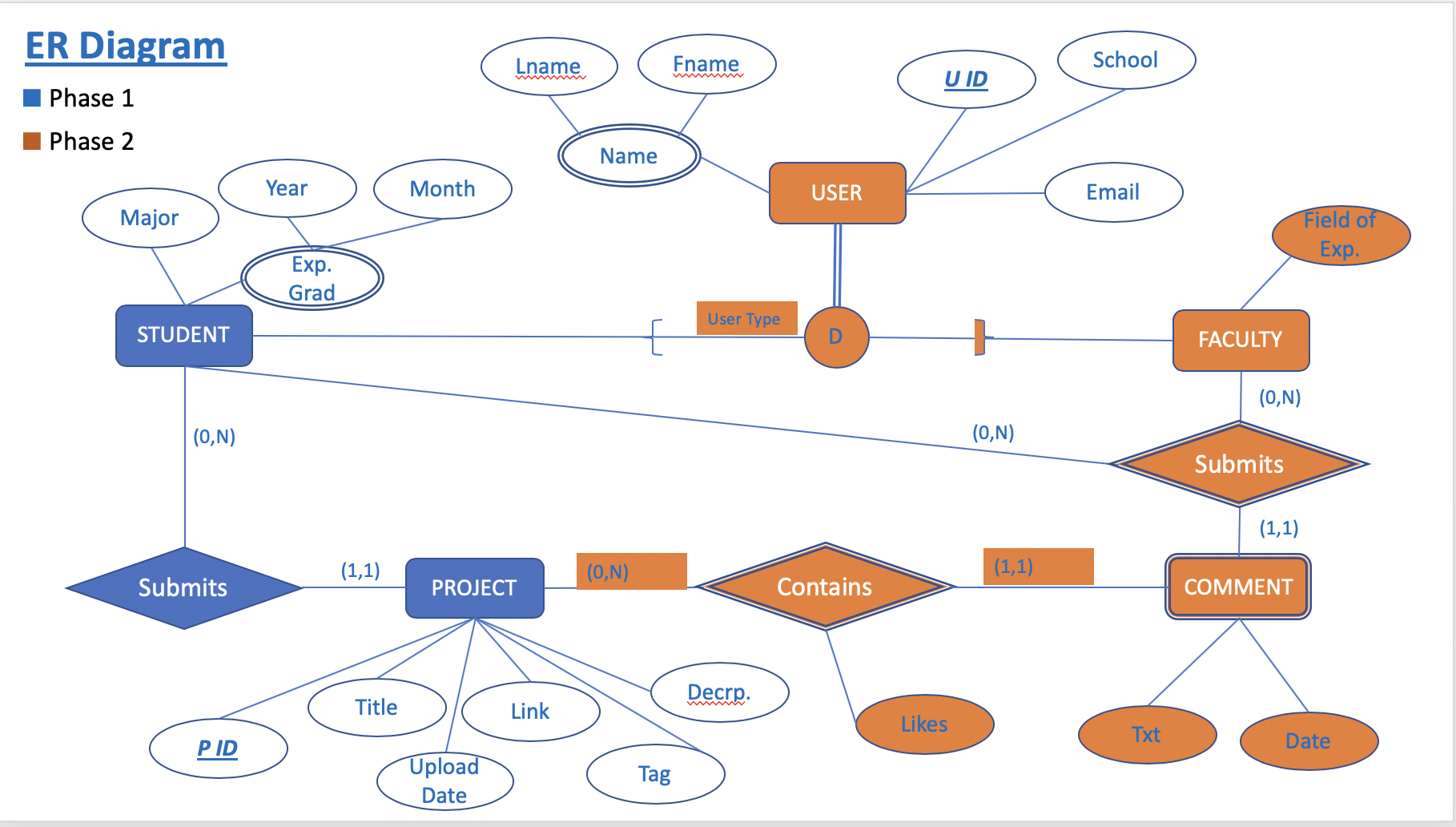
2. Application Layer

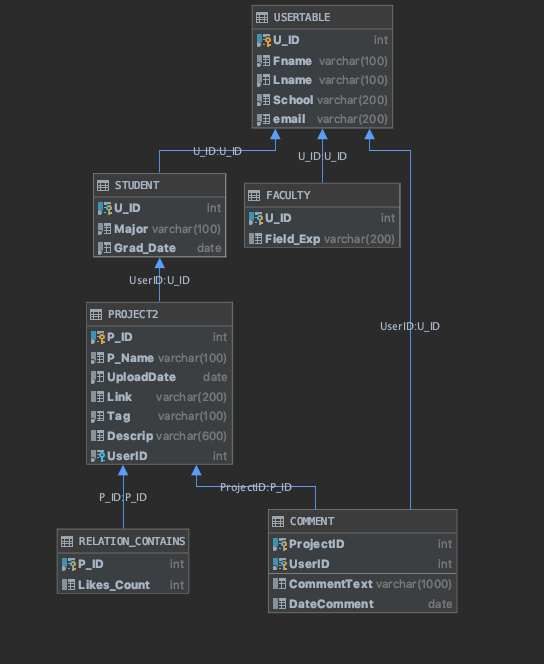
a. C# Razor (.cshtml files)

b. Communicates to presentation layer via URL Querying

3. Data Layer

a. Sql server. Receives commands from Application Layer.





# Key Algorithms

One of our main algorithms will be implementing keyword searching for uploaded projects. Returning relevant projects to the user’s search criteria will be crucial to the project portal’s success. There are 2 majors algorithms that we’ll need to implement proper keyword searching.

* Automated keyword tagging from project descriptions
  + Using simple NLP techniques to detect relevant search terms from user uploaded project descriptions.
  + By removing stop words we will extract relevant search terms that go beyond what users would upload manually.
* Keyword search of currently stored projects
  + By utilizing manual and extracted keyword tagging we’ll be able to properly index projects and provide a positive search experience for the user.
  + By prioritizing keywords and potentially tracking views we can return the most relevant and trending projects to the user searching our database.

Advanced search

Or how many views

Pseudo code to come soon once database and add project functionality is implemented.

# Classes and Methods

* Project Class
  + Add Method
  + Delete Method
* Authentication Method
  + LogIn Method
  + LogOut Method
* Search Class
  + SeachByName Method
  + SeachByKeywod Method

There are automated ways to get methods. She says you can put comments into your code, and this can be generated. “Generate API documentation.” This sounds like a duzy. Is there a tool that does this with html? Doxygen Manual

Find a tool that will calculate the lines of code and stuff for us.

# References

# Glossary