**CS673 Software Engineering**

**Team 4 - Team Builder**

**Software Design Document**

****

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| Raymond Chen | Configuration Leader | *Raymond Chen* | 09/22/2024 |
| Max Zink | Design and Implementation Leader | *Max Zink* | 09/22/2024 |
| Wenli Gai | Security Leader | *Wenli Gai* | 09/23/2024 |
| Jake Kelly | Team and Requirements leader | *Jake kelly* | 9/23/2024 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

[Introduction](#_87t9hln2vjz0)

[Software Architecture](#_3ipvmjgn6clp)

[Class Diagram](#_ky60nv8suxxm)

[UI Design (if applicable)](#_7ucksmkf6rzx)

[Database Design (if applicable)](#_tcmuor4nl1kz)

[Security Design](#_x18fj36s1121)

[Business Logic and/or Key Algorithms](#_mtfbusfb0eq3)

[Design Patterns](#_9zvwkmc4luo5)

[Any Additional Topics you would like to include.](#_15tmymhipvdv)

[References](#_50ojo9i46ytq)

[Glossary](#_8n34lvocupub)

# Introduction

In this section, give an overview of this document, and also address the design goals of your software system.

TeamBuilder is a team assignment tool to help create teams based on a few simple, but important questions that users will answer. This document outlines the architecture, class structure, user interface, database, and key algorithms of the TeamBuilder system.

# Software Architecture

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

The architecture follows a three-tiered design using Angular for the frontend, Spring Boot for the backend, and an H2 database for local data storage.

* Frontend: Angular will manage the user interface, handling user inputs via forms, and sending requests to the backend through RESTful APIs.
* Backend: Spring Boot will handle all business logic, including processing questionnaire data, team formation, and managing groups.
* Database: The H2 database will be used to store user responses and team assignments.

Architecture Diagram

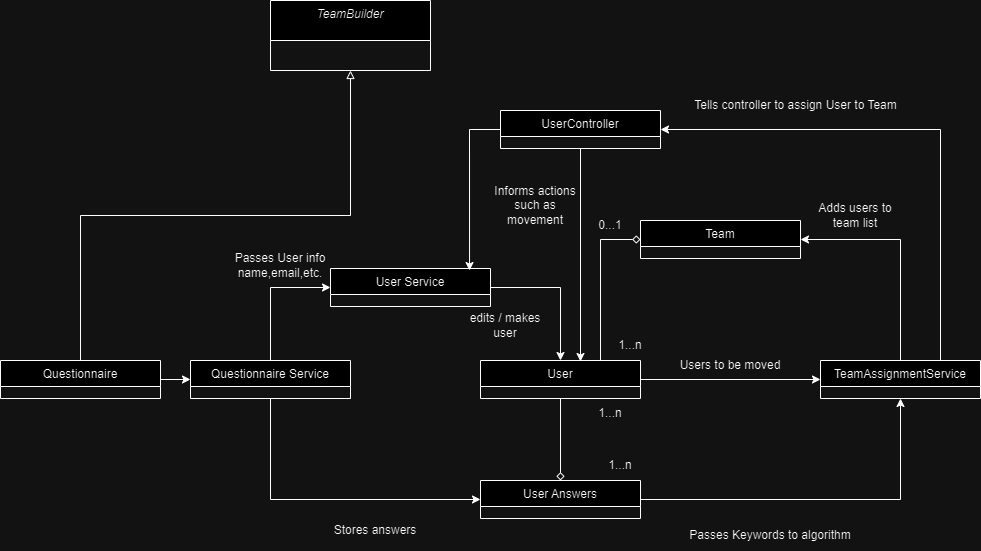


# Class Diagram

In this section, you will provide a detailed description of each component (or package) and use one or multiple class diagrams to show the main classes and their relationships in each component.

The system will be divided into several main packages:

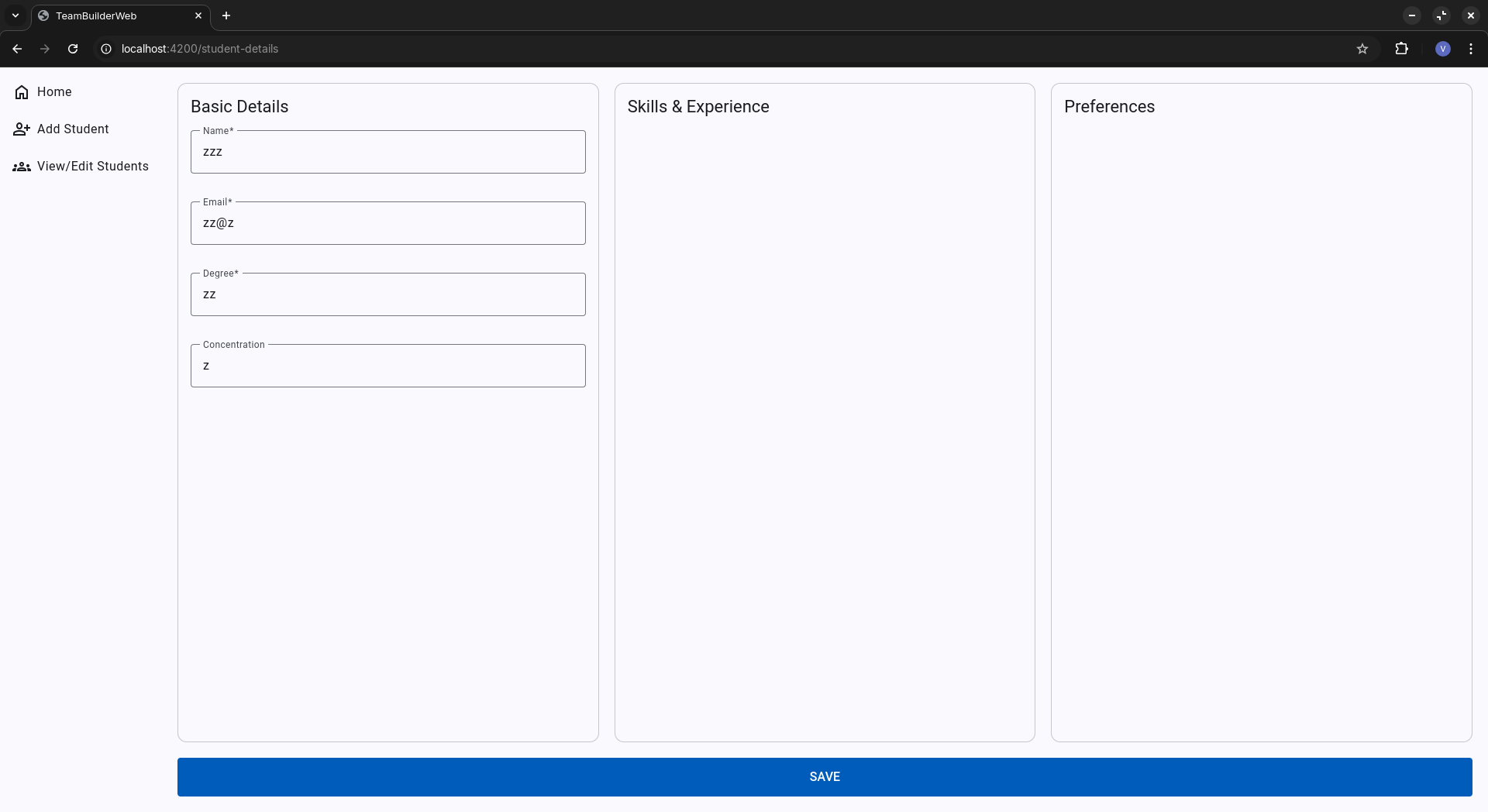
* + Controller:
    - UserController: Manages user interactions.
  + Service:
    - TeamAssignmentService: Implements the team formation algorithm.
    - UserService: Handles the user profile data.
    - QuestionnaireService: Handles the process of questionnaire data.
  + Model:
    - User: Represents a user and response.
    - Questionnaire: Represents the questionnaire filled by the user.
    - Team: Represents a team and list of users.

Class Diagram

# UI Design

The UI is divided into following main components:

* Navigation (Sidenav)
  + Location: The navigation is presented as a left-aligned sidenav, always visible
  + Items:
    - Home: Redirects users to the home page.
    - Add Student: Opens the form for adding new student profiles.
    - View/Edit Students: Allows viewing and editing existing student profiles.
* Form Layout for Adding Student Details: The Add Student page uses a form that is organized into three distinct columns:
  + Basic Details: Captures the student's name, email, degree, and concentration.
  + Skills & Experience: Can include fields for the student’s years of experience, relevant skills, and past projects.
  + Preferences: Allows the user to specify their preferred role in a team (e.g., Team Leader, QA Leader) and their availability.



# 

# Database Design (if applicable)

In this section, you shall describe any database schema if used in your software system.

The H2 database will have the following tables:

* Users Table:
  + id
  + name
  + email
  + role
  + answers (List)
  + team\_id
* User\_Answers Table:
  + id
  + answer
  + user\_id
* Team Table:
  + id
  + team\_name

In relation to our project as a whole, the database has a fairly straightforward approach. All currently unimplemented aspects of the database are colored in red.

A user is a table which includes all necessary information gathered by filling out the survey. This includes the user’s name, email and a list of their provided answers on the survey. Aside from data gathered through the survey, this table also includes a member’s team\_id and role. These two entries are of course dependent on Teambuilder’s algorithm executing, and will be empty if a User has yet to be assigned.

The User\_Answers table contains specific data from the survey. Answers don’t include information such as name, email, or role. Rather, answers contain specific keywords gathered from a user’s answers on our survey. For example, if a user were asked the question “what programming languages are you familiar with?” and the user answered C++, then a new entry would be added to User\_Answers with the answer keyword “C++” and a foreign key for the id of this user. This table allows us to have multiple of the same, or similar, keyword answers. Through these repetitions, with our algorithm, we would be able to group users together depending on the needs of a specific team.

Finally, the Teams table is a two field table used to group users. Aside from the id, Teams only includes the team name, which is of course used to differentiate the teams. This table is referenced by the User table, which stores the team’s id in its foreign key team\_id. Outside of the database, a “team” is a list of users. Above the team list we have the “teams” list which is a list of all created “team” lists.

# Security Design

In this section, you shall describe any security design in your software system.

TeamBuilder implements essential security measures to protect user data: Users log in securely with hashed passwords using Spring Security. Data transmitted between the frontend and backend is encrypted via HTTPS. Sensitive data (like passwords) is encrypted at rest in the database. All user inputs are validated and sanitized to prevent attacks like SQL injection and cross-site scripting (XSS). Session Management: Secure session handling using HTTP-only cookies with automatic logout after inactivity. Important actions, such as team updates are logged for monitoring and security reviews. Regular database backups ensure data can be restored in case of failure.

# Business Logic and/or Key Algorithms

In this section, you shall describe any key algorithms used in your software system, either in terms of pseudocode or flowchart, or sequence diagrams.

# Design Patterns

At this current iteration we have not actively utilized design patterns in the creation of our code. We will be reexamining where we can implement these patterns into our code. For example, one requirement of the project is a notification system that will alert users when there has been a change to their profile. In this case, we can, and will, implement an observer design pattern. This pattern allows us to watch actions, such as team assignment, and then, with connected functions, push notifications such as emails.

# Any Additional Topics you would like to include.

# References

# Glossary