# Introduction

KanScratch is an alternative to MIT’s Scratch website that enables Scratch to be used effectively in an academic environment. Specifically, this project aims to make it easy for students and teachers to interact with each other in Scratch through new assignment and classroom systems, as well as ensuring that the use of KanScratch will not violate any academic laws or regulations, such as FERPA.​

## purpose

The purpose of this project is to create an extension to Scratch that enables Scratch to be used effectively in an academic environment. Specifically, this project aims to make it easy for students and teachers to interact with each other in Scratch through new assignment and classroom systems, as well as ensuring that the use of KanScratch will not violate any academic laws or regulations, such as FERPA.

## Technology stack

This project will be using the following technologies:

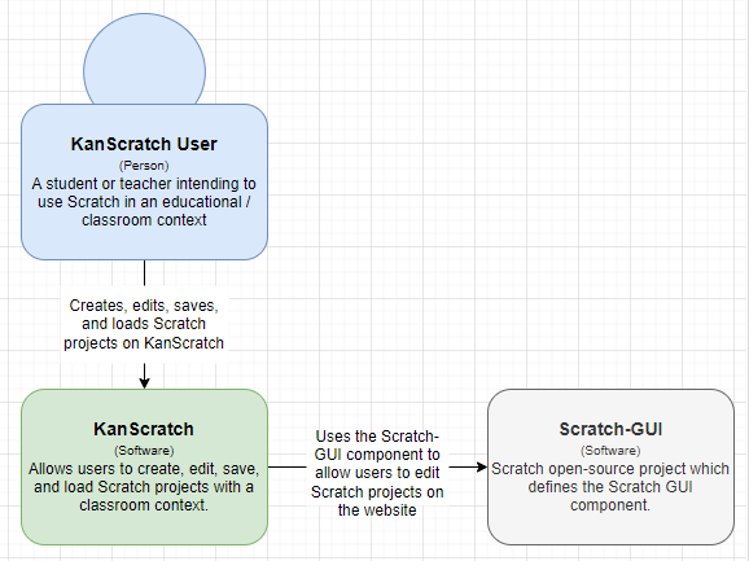
* React
  + Front End
* Django
  + Back End
* Django-Ninja
  + RESTful API
* PostgreSQL
  + Database
* Docker
  + Deployment / Environments

## C4 Diagrams

The KanScratch project will be modeled using C4 diagram models. At the current stage of development, we can abstractly model the project well using context diagrams and container diagrams. We will wait until a later stage in the project to start creating component diagrams and code diagrams.

### Context diagram

The figure below shows a context diagram representing the contexts in which the KanScratch project will be used.

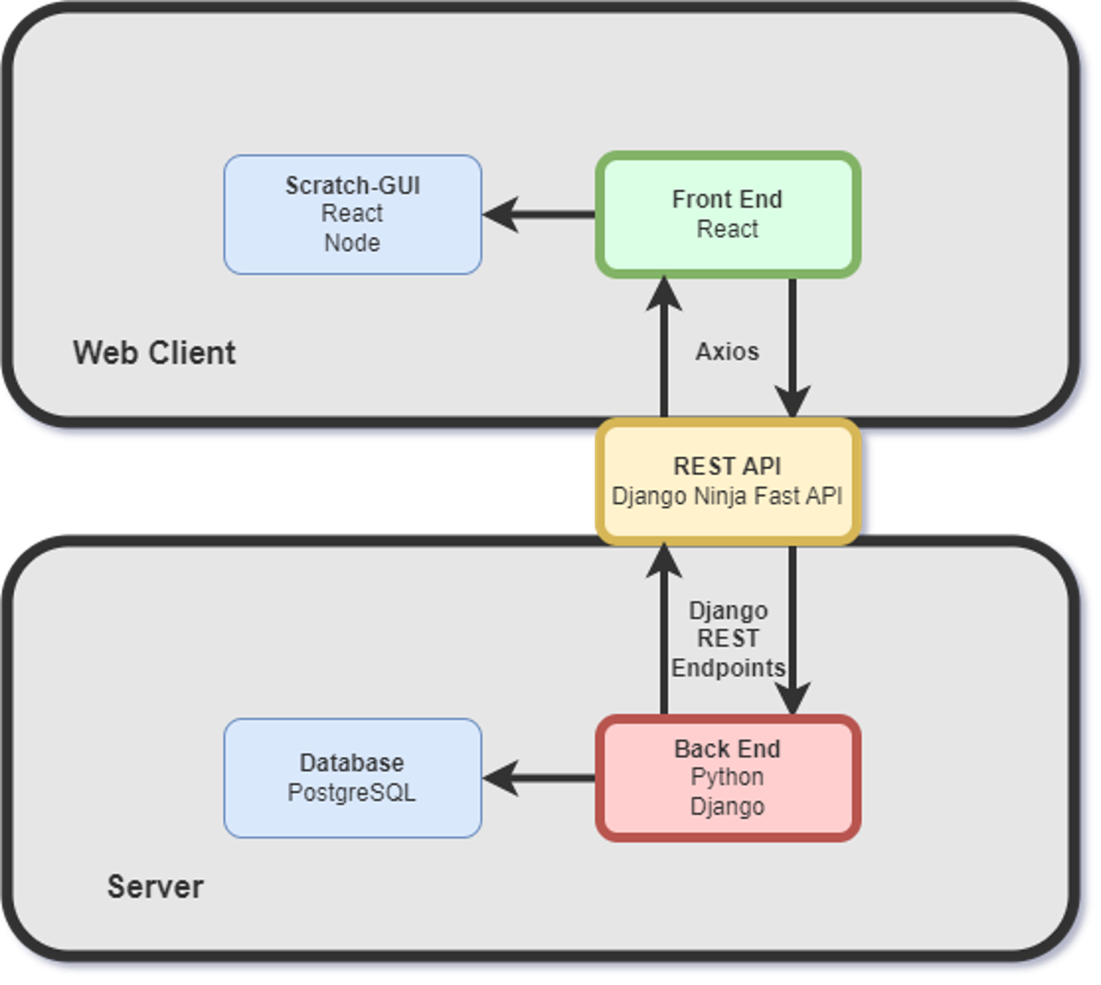


*Figure. System Context Diagram*

There are three main systems involved in the KanScratch project. Our focus is working on the KanScratch software (green) to meet the needs of a KanScratch user by applying the open-source Scratch-GUI to our own web application. It is important to keep these three high-level systems in mind while developing the web application because these systems must work in harmony to work as intended.

### Container diagram

The figure below shows a container diagram representing the high-level architecture of the KanScratch application.



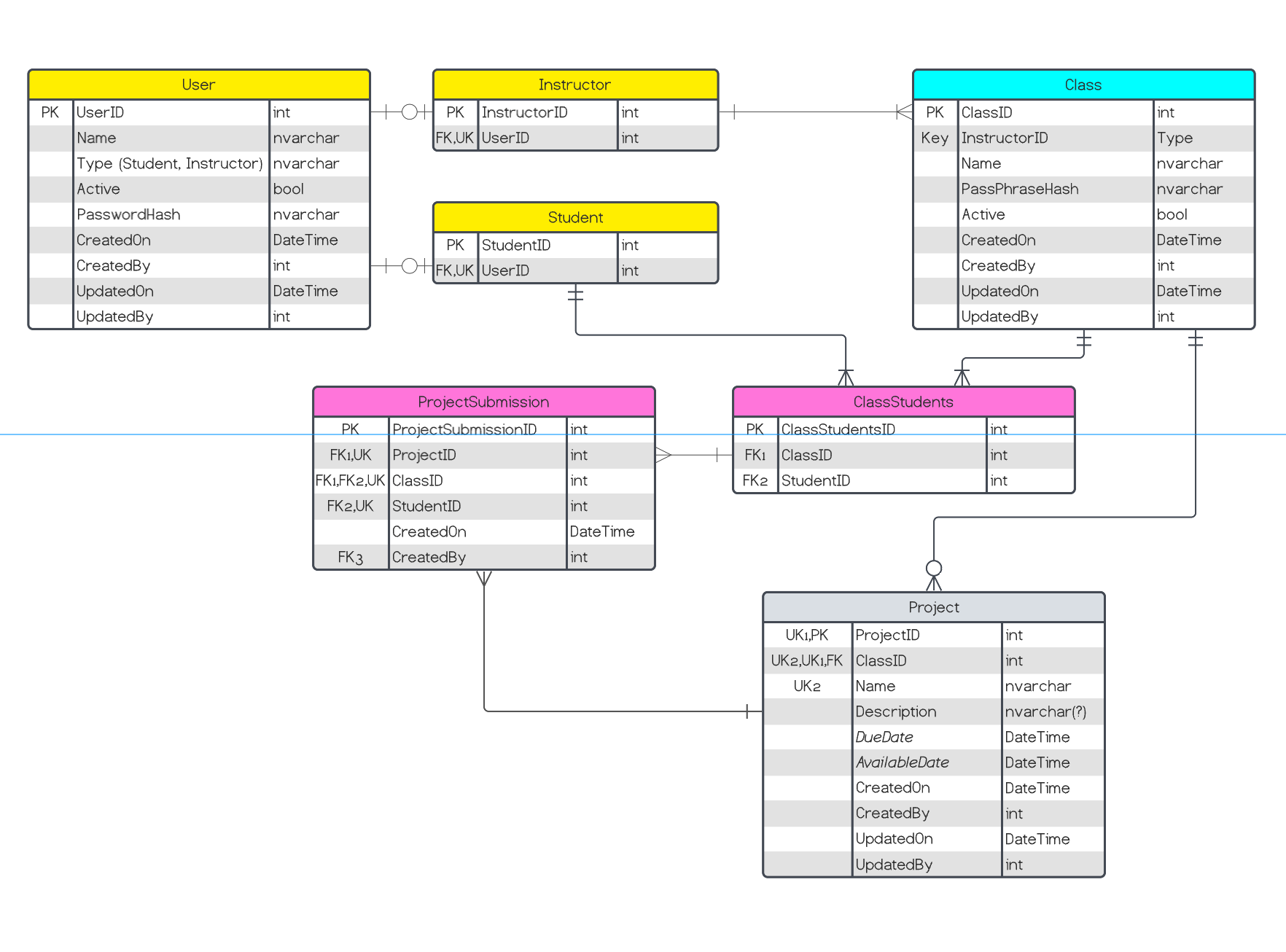
*Figure. Container Diagram*

Once the web client has downloaded the application from the web server, KanScratch will be prepared to follow the architecture pictured in the diagram. The web client consists of a React application which references the existing open-source scratch project. The reference will target the Scratch-GUI repository, which contains an edited version of the scratch blocks visual editor. The Scratch-GUI project will run its own Scratch-VM and Scratch-Blocks. If any changes are needed to the actual block logic or the virtual machine, then the referenced Scratch-GUI project will need to run the altered versions of those projects.

On the web server is a Python/Django application responsible for creating and maintaining the database for KanScratch. The database is code-first and is constructed on the server when the application is run without an already existing database.

The two projects communicate with each other using REST API hooks using Django Ninja. On the client, the REST API is consumed using FETCH get and post. The data being sent are JSON files.

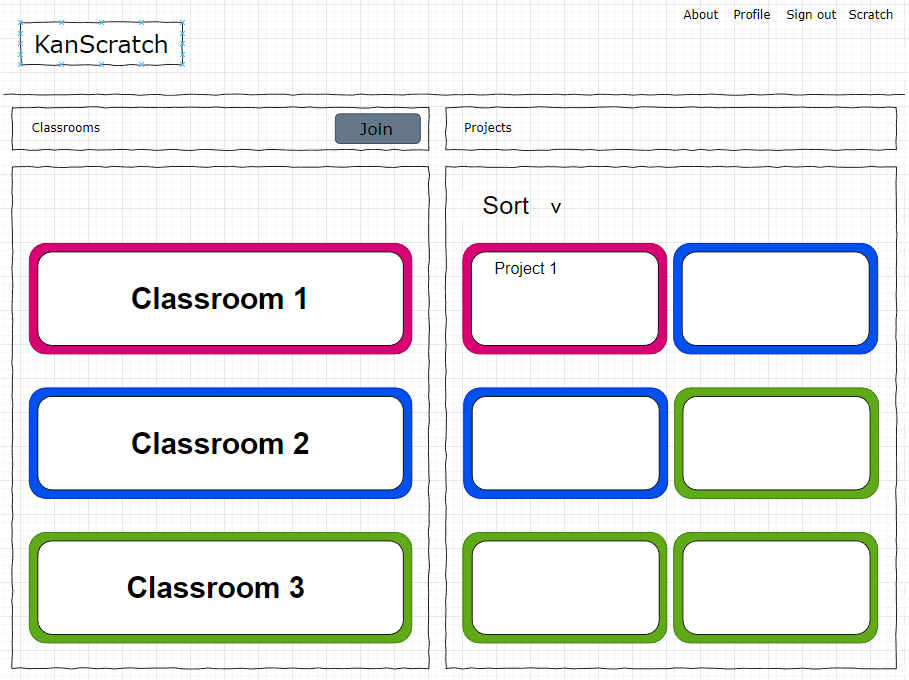
## Database diagram



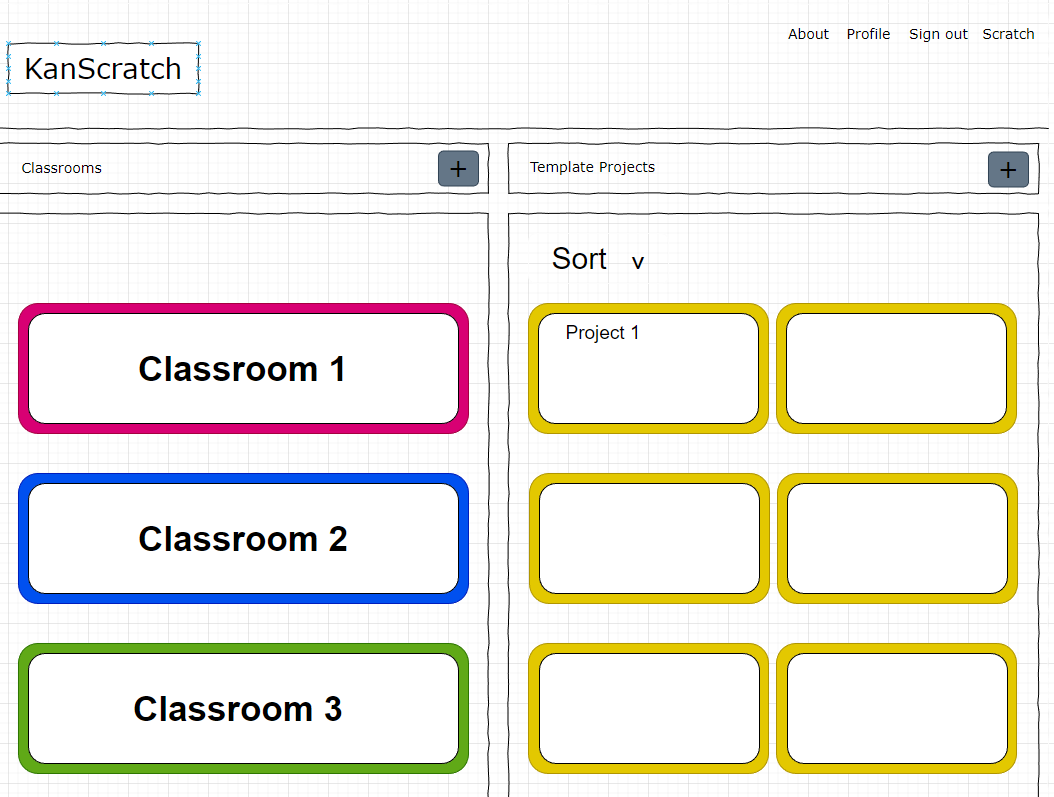
## UI Mockups

This section will show various UI mockups made that will be implemented into KanScratch when we start website design.

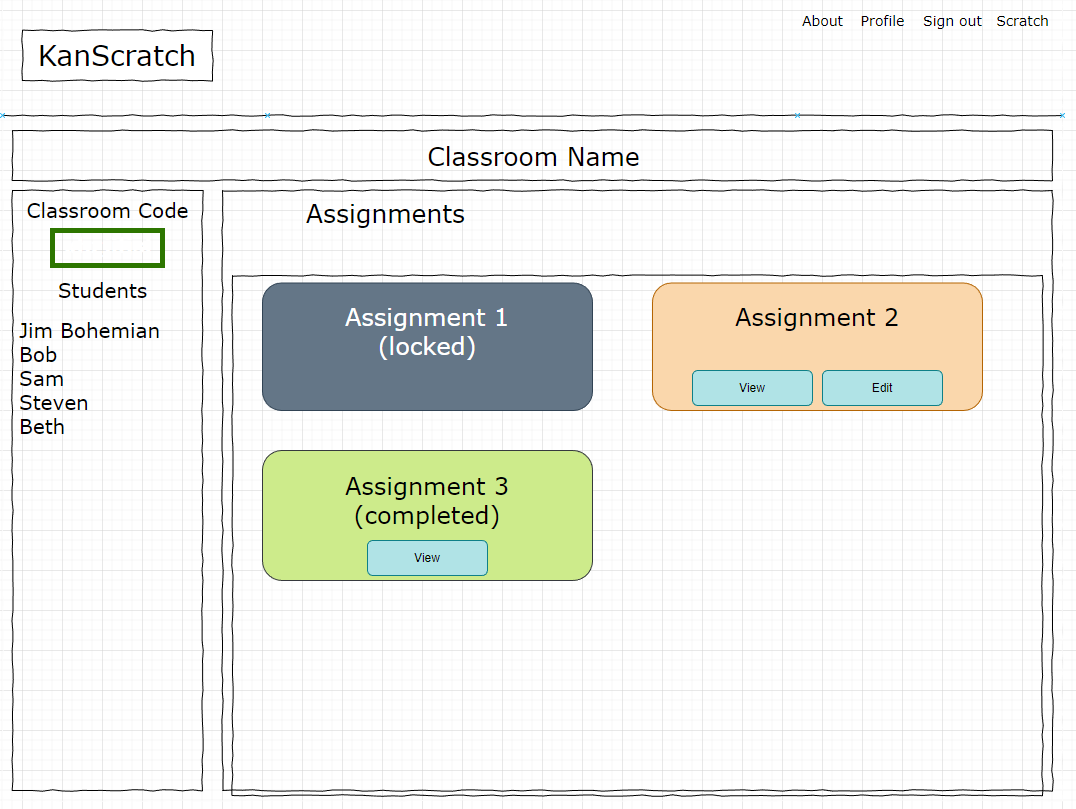
### Main Student view:



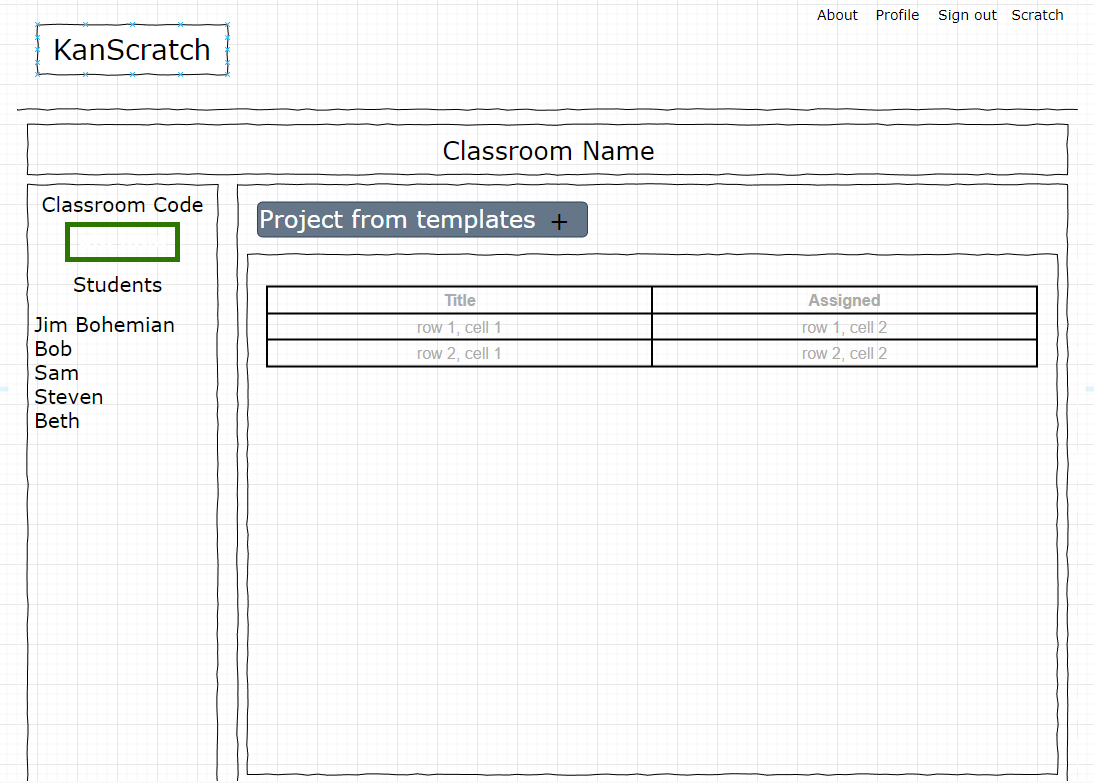
### Main teacher view:



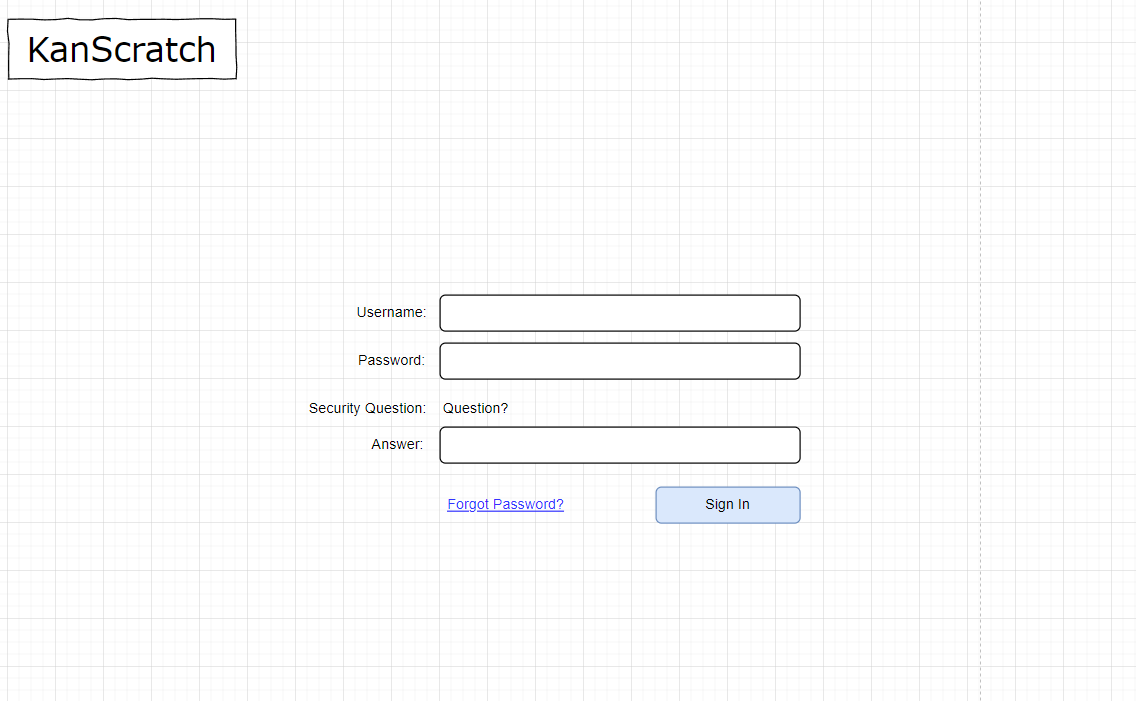
### Student class view:



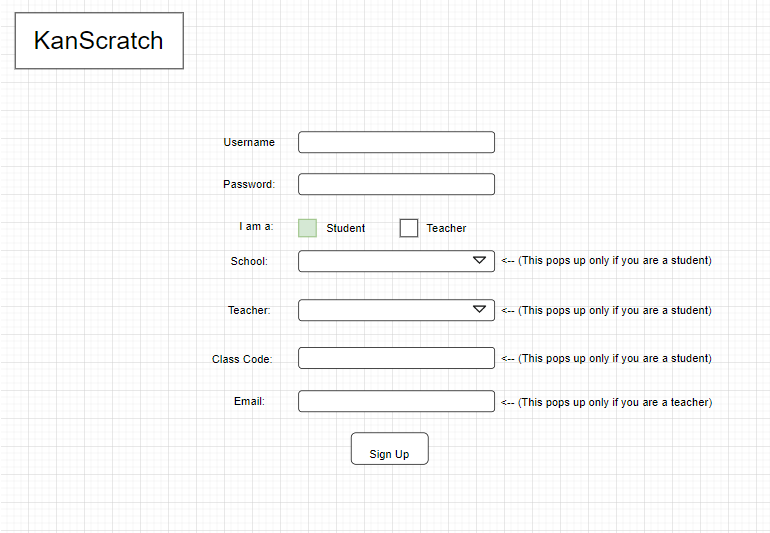
### Teacher class view:



### Sign in view:



### Sign up view:



## Primary Goal for the semester

Our primary goal for this semester is to create a functioning, secure, and well-designed website. To accomplish this, we need to focus most of our semester working on the implementation of the Scratch-GUI react components into the KanScratch react web application. While this is being worked on, the classroom and teacher-student relationship portions of KanScratch can be worked on. Ideally, once Scratch-GUI and the classroom systems are completed, all that will be left is saving projects to our database and general website design.

To create a successful classroom system, we need to have a sign up / login system that is easy for students and teachers to use. The teacher portion of KanScratch should allow a system to create classes and assign projects to a class, and the student portion of KanScratch should allow a system to work on, save, and load projects within a classroom.