# Smart Grader: An A.I. Driven Bidirectional Education Platform

Akankshya Biswal, Emi Harry, Gio Abou Jaoude, Johnathan Angamarca, Joseph Dougal, Jose Salcedo, Rithvik Subramanya, Tj Rabbani
Email:{ab94615n, eh26833n, ga97026n, ja73484p, jd91219n, js00759n, rs76020p, tr55885n}@pace.edu
Seidenberg School of CSIS, Pace University, New York

***Abstract*—The Computer Science Project I course at Pace University is a graduate conclusion to the Computer Science graduate degree. It involves the development of a market ready product that has gone through the complete Agile methodology. The product proposed is an alternative teaching aid to augment the student teacher relationship through the use of web design and natural language processing.**

**Keywords— web design, natural language processing, student education, teaching tools, Agile**

## I. Introduction

**Purpose** - This document aims to present a detailed description of the Smart Grader software, the process involved in its development, and its real-world application. It will explain the purpose and features of the software, the interfaces of the software, what the software will do, and the constraints under which it must operate.

**Document Conventions** - The format follows the Institute of Electrical and Electronics Engineers (IEEE) style for formatting research papers. The template follows standard Software Requirements Specification.

**Intended Audience and Reading Suggestions** - The entirety of this document is meant exclusively for the viewing of the current project team, who are responsible for the development of the software, and the supervising faculty members.

**Product Scope** - Smart Grader is cloud-based platform for A.I. assisted setting, and grading of homework assignments and exams. The platform also enables students to utilize the grading feature as a studying tool, with text lookup based on, and the ability to check the grammatical structure of students' answers to propose improvements, within the context of the subject matter.

**Project Scope** - To develop a minimum viable product (MVP) that can generate answers to math and essay questions, and automated assignment questions.

## II. Literature Review

In a world of increasing online presence, learning online is becoming more widely demanded. Educational platforms that can leverage the new landscape will dominate the field [1]. Many of these sites offer services that reflect or attempt to reproduce on site learning to limited success. As more platforms offer student experiences in the home, parents are seeking a greater involvement [2]. There exist weak points in the online learning process.

One of the greatest points of contention in the use of online learning platforms, and massive open online courses alike, is there ability to asses students capabilities and grasp of the material [3]. Of the top 27 online learning platforms, zero employ machine learning as part of the user experience [4]. The solution to these problems is a new form of natural language processing.

The Bidirectional Encoder Representations from Transformers (BERT) is a widely used model for generating answers to questions on trained text data [6]. The advanced neural network employs the properties of transfer learning to create a user specific model with every text. BERT has been used to solve many real world needs outside of academic contexts [5]. BERT is open source and available for all use [7].

## III. General Description

**Product Perspective** - The educational platform is meant as an aid to the online learning process. It is meant to fill the need for at-scale solutions for a growing educational space. Students, parents, teachers and teaching institutions will need an automated alternative to the current method of interacting with assignments individually.

**Product Functions** - Smart Grader will:

- allow the automatic grading of assignments, both technical and writing based.
- give parents a view of the grades their children

**User Classes and Characteristics**

- Student User: such as K12 students, college students, or adults learners, either independent, or under the supervision of an educator, their parent, guardian, or primary caregiver.
- Educator User: such as high school teachers, and college professors at all levels who want to automate their testing and grading process.
- Supervisory Users: such as, organizations, parents, guardians, or primary caregivers who have access to the admin dashboard to monitor the progress of their dependent or student.

**Operating Environment** - The software is being designed to operate on:

- Windows 10
- Mac OS
- iOS
- Android 25 and up

**Design and Implementation Constraints** - For the development of the MVP, Python was selected as the main programming language since it's open source, with a wide range of libraries that enable faster development.

Of the popular Python web frameworks (Django, Flask, Pyramid, CherryPy, etc.), Flask was chosen because its lightweight design ensures that the development process is not encumbered with bloatware, giving the free rein to plug and play different open-source packages as needed. SQLAlchemy is being used as the ORM (object relational mapping), since it allows users to interact with the database and makes adding CRUD (create read update delete) features into a web app very easy to do. Being an A.I driven system, the unique features of the system are powered by machine learning models through a distinct combination of spaCy (a natural language library) Pandas library (for data manipulation and analytics), and BERT.

The front-end design of the application is being developed with JavaScript, HTML, and CSS through Bootstrap framework, which is a project that was open sourced by Twitter for responsive mobile-first sites. The library includes various front-end technologies to help uses style their HTML components and add functionality to your web applications. Thus, a lightweight version of CSS/JavaScript/jQuery to reduce network load when using our website

Bootstrap is weaved into the Flask application by using Jinja2 web templating. The templating framework supports basic logic that allows users to build dynamic web applications. Opting for a readily scalable product, the application has been installed and is hosted on AWS's Ec2 service, running Ubuntu through Nginx. AWS offers a free tier Ec2 instance for one year, which is perfect for the duration of this project. Nginx is an open-source production grade web server that acts as a reverse proxy to our Flask web application. Following the same theme, Postgres is the relation database of choice, since AWS lists it as part of their RDS (Relation Database Service). However, running a separate instance by installing and configuring Postgres on Ubuntu Ec2 instance, has helped to cut cost and provides more flexibility.

**User Documentation** - Current documentation is available on GitHub, and OneDrive, since the work is still in progress.

**Assumptions and Dependencies** - The Smart Grader MVP is being developed on the following assumptions:

- The identified first adopters have a need for the product.
- The identified first adopters do not have any learning disabilities.
- All open source software being used in the development will remain as such.
- All project team members will remain on the team until the completion of the project in May 2022.

## IV. External Interface Requirements

Any web browser should be sufficient to interact with the website. The diagram below shows the modular front-end system logic, having separated the front-end into modules to optimize performance, increase browser rendering time, and ensures components can be readily updated without breaking the entire system.

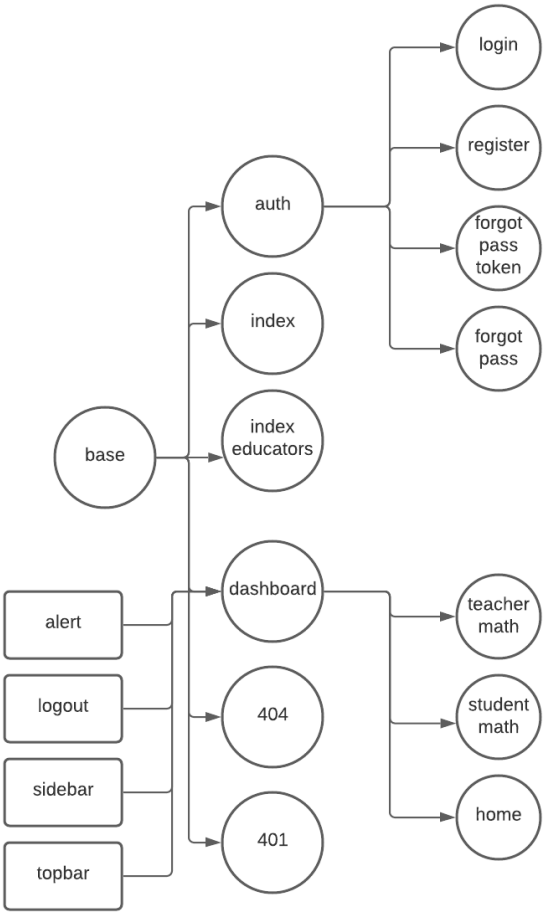


Fig. 1: Front-End Diagram

**User Interfaces** - The user interface for Smart Grader is compatible with any browser such as Internet Explorer, Mozilla Firefox, Safari, or Google Chrome through which users can access to the system.

The user interface could have been implemented using any tool or software package like Java Applet or Servlet, but instead it has been implemented using Python, CSS, HTML, and JavaScript through Bootstrap. The logic behind the interactions between the users and the software is as follows:

1) When the Users first navigates to the Smart Grader website, they will see student's landing page as shown below with a welcoming Hero section that consists of call to action prompts and two buttons whose colors reflect the product's logo.

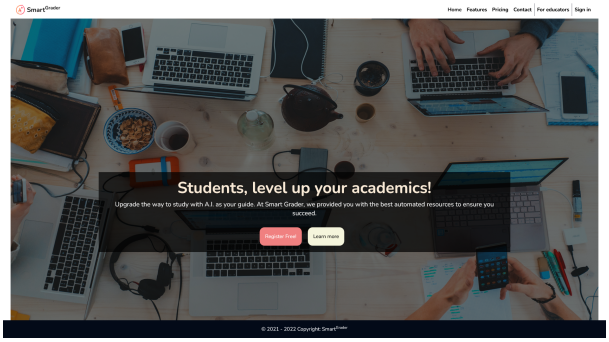


Fig. 2: Landing Page for Students

2) If the User falls under the Student class, and he/she clicks the button to the left, labeled ”Register Today”, the system navigates them to the registration page where they can create a free account. They are required to use the drop-down menu to select the student class for their registration to be successful.

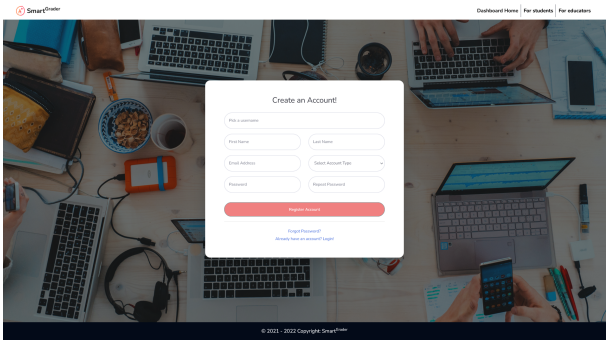


Fig. 3: General Registration Page

3) If on the other hand, the button labeled ”Learn more” is clicked, they are navigated to the ”Features” section of the student’s landing page. Their position is highlighted by the corresponding label on the navigation bar, as shown below.

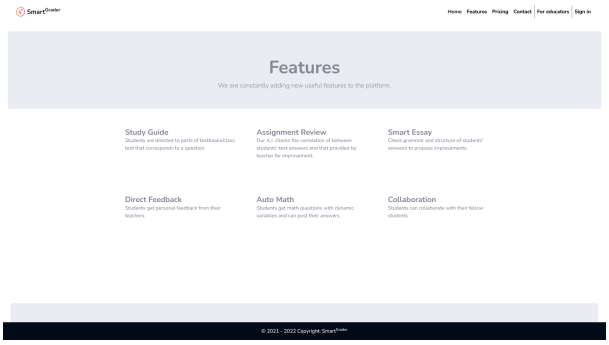


Fig. 4: Features for Students

4) Thereafter the student user can click on the ”Pricing” label also found on the navigation bar to be directed to the relevant Paywall that shows the pricing model.

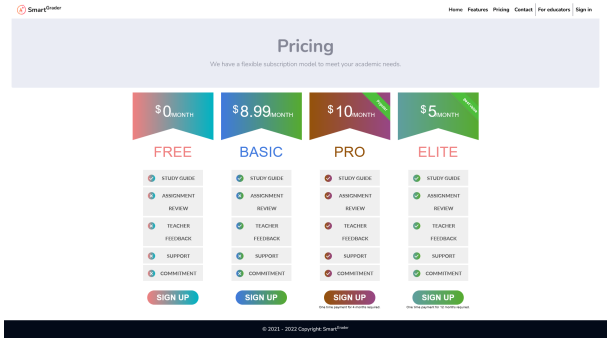


Fig. 5: Pricing for Students

5) Upon registration they are redirected to the dashboard.

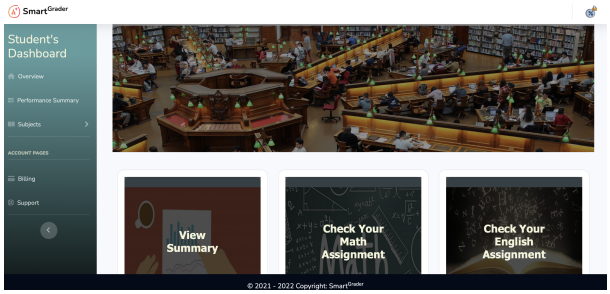


Fig. 6: Student Dashboard Screen

6) If the User falls under the Educator class, on the first landing page which has been tailored to students, to the right of the navigation bar, the user will click the ”For Educators” navigation button. Doing so will redirect the user to the landing page tailored for educators. On this page, the user will see two call to action buttons in the Hero section of the landing page.

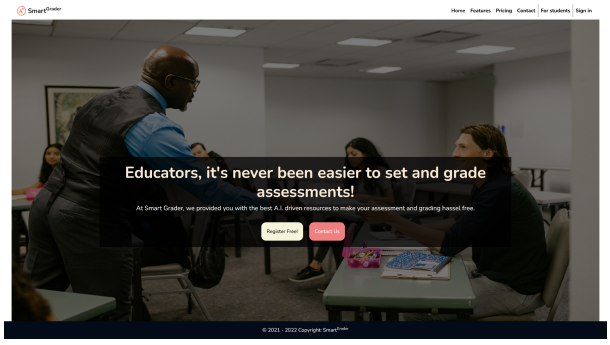


Fig. 7: Landing Page for Educators

7) The first button, ”Register Free”, when clicked will direct the user to the same registration page as shown on the previous page. They are required to use the drop-down menu to select the educator class for their registration to be successful. The second call to action button ”Contact Us”, when clicked will direct the user to the contact form, where they can place enquires. All information provided in this form, will be directed to the development team.

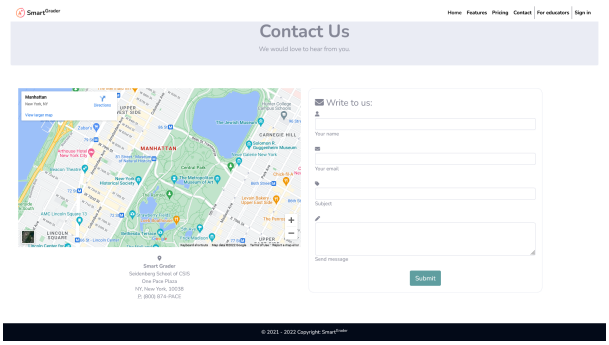


Fig. 8: Contact Section

8) Upon registration the educator user is redirected to the dashboard.

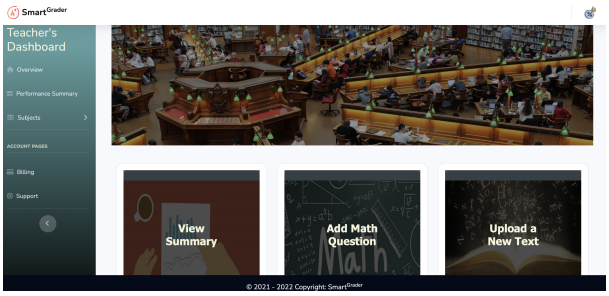


Fig. 9: Educator Dashboard Screen

9) If on the other hand, the ”Features” navigation button is clicked on the navigation, the user are navigated to the ”Features” section of the educator's landing page. Their position is highlighted by the corresponding label on the navigation bar, as shown below.

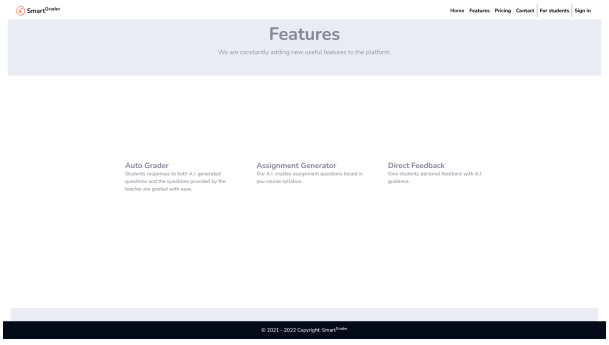


Fig. 10: Features for Educators

10) Thereafter the educator user can click on the ”Pricing” label also found on the navigation bar to be directed to the relevant Paywall that shows the pricing model for educators including a trial account, for free.

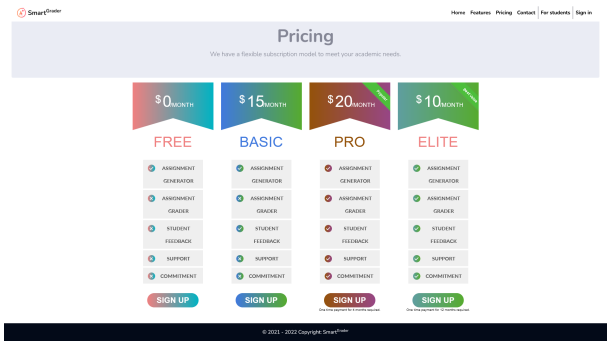


Fig. 11: Pricing for Educators

11) For consequent visits to the web application, when the user clicks on the ”Sign In” navigation button, the educator user is immediately directed to the dashboard. They have three features available to them: the AutoGrader, the Assignment Generator and Direct Feedback.

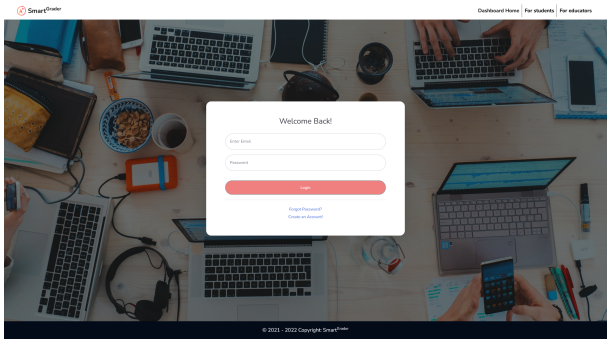


Fig. 12: Login Screen

12) In the advent that the user has forgotten the log in credentials, the forgot password can be clicked on the login form which redirects the user to the reset form.

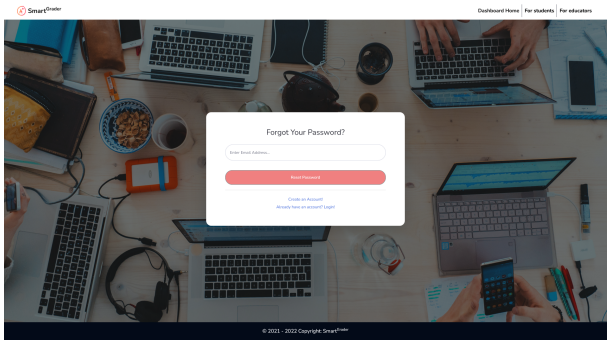


Fig. 13: Password Reset

13) When done, both user class (Student and Educator) can click on the lock button at the top right-end off the navigation bar, a drop-down menu appears from which either user class can log out of the system. Doing so redirects the user to the landing page.

**Hardware Interfaces** - Given that the Smart Grader application must run over the internet, being cloud-based, all

the hardware used to access the application must be able to connect to the internet. Thus the hardware interface for the system are:

- Internet Modem
- Wireless adapter (minimum capacity 3G)
- WAN – LAN
- Ethernet Cross-Cable

**Communications Interfaces** - The Smart Grader system will use the HTTPS protocol for communication over the internet and for the intranet communication will be through TCP/IP protocol suite.

**Software Interfaces** - Smart Grader requires any browser such as Internet Explorer, Mozilla Firefox, Safari, or Google Chrome to be installed on the system.

## V. System Features

The high level summary of the functional requirements of the software includes ensuring that users can create a login with a specific username and password. Users passwords must be encrypted at rest with the ability reset passwords. The software should create a math formulas on-the-fly with minimal user input. Users under the Educator category should be able to update a math formula with an expression. Math formulas should resolve themselves. Users with the Student category should be able to submit an answer for a math expressions. The software should automatically grade submitted answers and return scores. Users with the Student category should be able to generate report with grades for all assignments done. Below is a detailed breakdown of the unique system features.

- *Auto Grader*
  - **Description and Priority**: Students responses to both A.I. generated questions and the questions provided by the teacher are graded with ease. On the priority scale, this feature ranks 9 in benefit, 7 in risk, 6 in cost, and 9 in penalty, giving it an average priority score of 7.3
  - **Stimulus/Response Sequences**: The Auto Grader should be prompted automatically, upon the submission of an assignment from the student user.
  - **Functional Requirements**: The mathematical question grading requirements should only involve the math class features. The text based questions will involve dedicated NLP engines.
- *Study Guide*
  - Description and Priority : Non-committed
  - Stimulus/Response Sequences : Non-committed
  - Functional Requirements : Non-committed
- *Assignment Generator*
  - Description and Priority : Non-committed
  - Stimulus/Response Sequences : Non-committed
  - Functional Requirements : Non-committed
- *Assignment Review*
  - Description and Priority : Non-committed
  - Stimulus/Response Sequences : Non-committed
  - Functional Requirements : Non-committed
- *Smart Essay*
  - Description and Priority : Non-committed
  - Stimulus/Response Sequences : Non-committed
  - Functional Requirements : Non-committed
- *Direct Feedback*
  - Description and Priority : Non-committed
  - Stimulus/Response Sequences : Non-committed
  - Functional Requirements : Non-committed

## VI. Non-Functional Requirements

*Performance requirements*

The A.I. quality requirements will shift as the scope of the product expands.

*Safety requirements*

Smart Grader will be a browser based product that will pose little risk to the safety of the users.

*Security requirements*

The security of the user is a top concern. Smart Grader will be subject to NY and federal laws concerning data privacy. To name the most relevant:

- New York Senate Privacy Act 2021
- Family Educational Rights and Privacy Act
- Children's Online Privacy Protection Rule
- Federal Trade Commission Act

*Software quality attributes*

At scale, Smart Grader may hold hundred of concurrent users. The quality of of the software will be held to this standard.

*Other requirements*

Relevant legal disclaimers indemnifying Smart Grader for the content that is generated by users of the platform (teachers) that do not explicitly operate under the supervision of Smart Grader will be issued.

## VII. Backend Structure

*Math Backend Structure*

The mathematical grading works on the principal of symbolic logic. Many versions of this solution to grading math based questions exist online. A direct solution was used for simplicity.

The logic of the mathematical expression is encapsulated in a python class that contains a an array of the variables. The object consists of one expression and one array. Each time a variable is replaced with a definitive number, the array and expression are updated as such. If only one variable remains in the expression, it can be transformed into an equation that solves for the missing variable. If no variables remain, the expression can be reduced to a real number.

As of the latest version, the symbolic logic solution does not accept imaginary values as inputs into the instantiation of the object. Future iterations will allow for imaginary numbers as well as more intricate number sets.

*Essay Backend Structure*

The natural language processing backend structure of SmartGrader relies on the software library spaCy and the open tool BERT. All software and tools used are open source.

spaCy was used to handle the lower scale language handling tasks including tokenization and text similarity. This helped break larger texts into smaller subtexts that BERT could handle. After answers are produced by BERT, spaCy handle scoring of the individual generated answers. The student proposed solution is then weighted against all the generated answers for a question. The average is rounded up to a value of one or zero.

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained neural network used to answer question using a small text as a reference. BERT works by retraining itself on each new text in a form of transfer learning. This means that BERT has a set token length. Effectively, BERT can only retrain itself in a text with 512 unique words. All BERT documentation is available if needed.

## VIII. Retrospective

The development process was aided by several team building steps including:

- Organizing a weekly scheduled time for team meetings.
- Employing a fully integrated and utilized Github.
- Consolidating all documents onto one inclusive OneDrive folder.
- Promoting and advertising a robust Wikipage for the team.

The predominant sentiment among the team is that there still exist bottlenecks in the development process that can be aided with a few steps:

- Organizing even more team meetings in a dynamic fashion on an at need basis.
- Seeking more customer feedback during the quality assurance stage.

The current list of scheduled improvements to the employment process involve:

- Solidifying cohort structure with distinct titles and responsibilities therein.
- Beta-testing the current product.

## IX. References


[1] Cara Bafile *Online Grades Provide Access and Accountability*, https://www.educationworld.com/a_admin/admin/admin467.shtml

[2] Julia A. Osteen *Effects of an Online Grading System on Parent Involvement*, http://itm.coe.uga.edu/archives/fall2005/josteen.htm

[3] Matt Crosslin, et al. *Creating Online Learning Experiences*, https://uta.pressbooks.pub/onlinelearning/chapter/chapter-9-assessment-issues/

[4] Kyriaki Raouna *27 Best Online Learning Platforms*, https://www.learnworlds.com/online-learning-platforms/

[5] Sergey Parakhin, Oleg Smirnov, Eugene Steinberg *Finding a needle in a haystack: building a question answering system for online store*, https://blog.griddynamics.com/question-answering-system-using-bert/

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, https://arxiv.org/abs/1810.04805

[7] jacobdevlin-google *google-research/bert*, https://github.com/google-research/bert


## X. Appendix

### 1) UML CLASS DIAGRAM

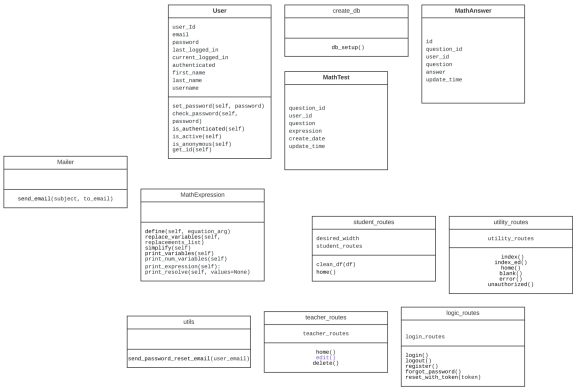


Fig. 14: UML Class Diagram