Requirements Specification and Use Cases

Proof Checker Tool - SE691

Rakhfa Amin, Thoman Andrews, Kersley Jatto, Colton Shoenberger

System Requirements:

1. The system shall operate as a web-based platform.
2. The system will have admin, professor, and students as users.
3. The system shall allow each user to log into a unique account.
4. The system shall be capable of storing user account data.
5. The system shall be capable of storing proofs created by users.
6. The system shall be capable of storing problems created by users.
7. The system shall be capable of storing problem solutions.
8. The system shall be capable of storing assignments created by users.
9. The system shall be capable of storing assignment grades.
10. The system shall enable the testing of natural deduction.

Natural Deduction Tool Requirements:

11. The Natural Deduction Tool (i.e. “Proof Checker”, herein “the tool”) shall allow users to create proofs.
12. The tool shall allow users to edit proofs.
13. The tool shall allow users to save proofs.
14. The tool shall allow users to delete proofs.
15. The tool shall allow users to provide premises with which to begin a proof.
16. The tool shall allow users to provide a desired conclusion with which to end a proof.
17. The tool shall allow users to add new lines to a proof.
18. The tool shall allow users to add subproofs within a proof.
19. The tool shall allow users to conclude subproofs.
20. The tool shall allow users to delete a line within a proof.
21. The tool shall allow users to delete a subproof within a proof.
22. The tool shall allow users to edit lines within a proof
23. The tool shall allow users to edit subproofs within a proof
24. The tool shall number lines within a proof
25. The tool shall recognize the rules of Truth-Functional Logic (TFL)
26. The tool shall allow users to reference the rules of TFL to justify lines within a proof
27. The tool shall allow users to cite line numbers when justifying a line within a proof
28. The tool shall allow users to cite subproofs when justifying a line within a proof
29. The tool shall allow users to select what rules are permissible to reference within a proof
30. The tool shall be extensible such that additional rules from other forms of mathematical reasoning can be created and utilized within proofs
31. The tool shall be able to validate inputted expressions for proper formatting.
32. The tool shall be able to validate inputted lines for correctness.

33. The tool shall be able to validate a proof for correctness
34. The tool shall be able to validate a proof for completeness
35. The tool shall be able to display meaningful feedback to assist users with correcting errors

User Operations - Student Requirements:

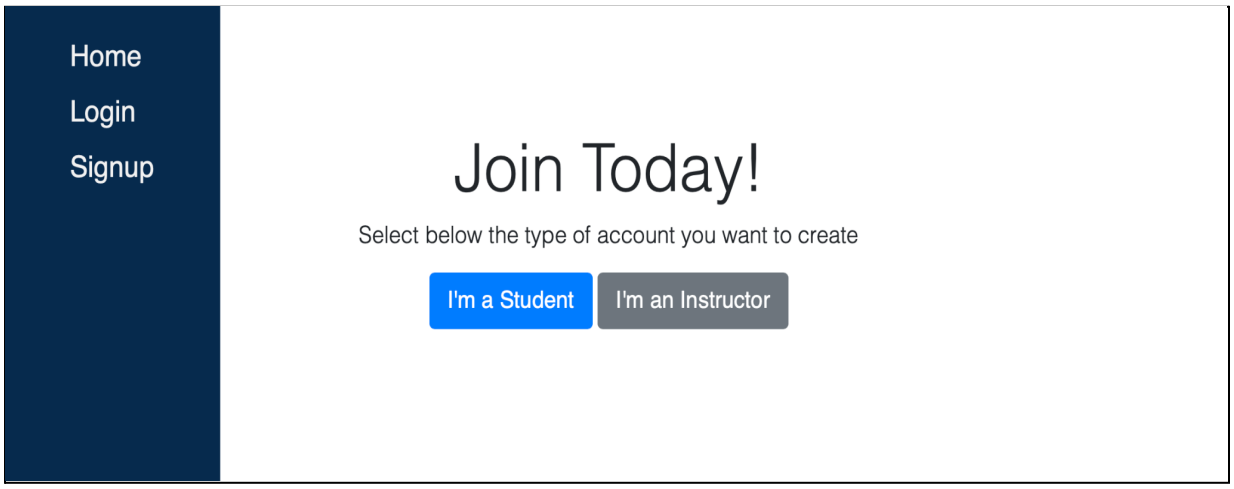
36. The system shall allow students to view assignments that have been assigned to them
37. The system shall allow students to view problems within an assignment
38. The system shall allow students to navigate between problems within an assignment
39. The system shall allow students to utilize the Natural Deduction Tool to complete problems
40. The tool shall provide static hints to students.
41. The tool shall provide dynamic hints to students.
42. The tool shall provide information for students to reference while working.
43. The tool shall allow students to check their inputted solutions for correctness.
44. The tool shall allow students to check their inputted solutions for completeness.
45. The tool shall allow students to save work in progress.
46. The tool shall allow students to submit an assignment for grading.
47. The tool shall be capable of automatically grade submitted work.

User Operations - Instructor Requirements:

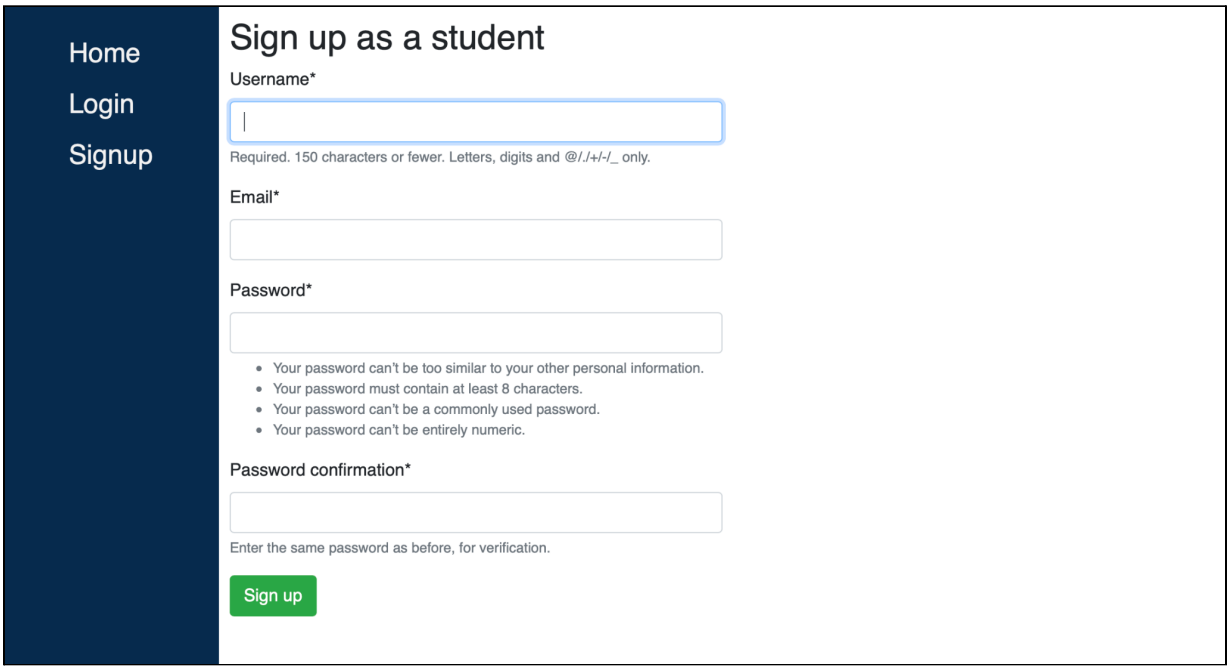
48. The system shall allow instructors to create a course.
49. The system shall allow instructors to delete a course.
50. The system shall allow instructors to add students to a course.
51. The system shall allow instructors to remove students from a course.
52. The system shall allow instructors to create problems.
53. The system shall allow instructors to create assignments consisting of problems.
54. The system shall allow instructors to assign what rules students can use when answering a problem.
55. The system shall allow instructors to view assignments submitted by students.
56. The system shall allow instructors to view grades from submitted assignments.
57. The system shall allow instructors to upload assignments.
58. The system shall allow instructors to export assignments.
59. The system shall allow instructors to export grades.
60. The system shall allow instructors to edit grades.

Use Cases

Use Case 1	**Create an Account**
Actors	Guests
Brief Description	This use case explains how a user creates an account and register as either a student or an instructor.
Basic Flow of Events	Basic flow begins when a guest user visits the web-based proof checker tool. Side Menu bar should be displayed. 1. Guest User clicks on the ‘Signup’ link on the menu bar to access the sign up page. 2. Sign up landing page prompts the guest to sign up as a student or as an instructor. 3. Guest user clicks on respective roles. 4. Sign up form is displayed, and the guest user enters required information in the fields and clicks on the sign-up button. 5. The system validates the information that the guest user has entered. 6. User information is stored in the user's account. 7. The system notifies the user that the account has been created and the login page is displayed. 8. This ends the use case.
Alternative Flow of events	During the sign-up process, if the guest user enters invalid information, the following occurs: 1. The system described which information was valid. 2. The system gives the user suggestions on entering valid data. 3. User re-enters the information and the system verifies them again. 4. If the user information fails the validation again, the alternate use case for invalid user information occurs again.This continues until the user enters valid information.
Pre-conditions	N/A
Post-condition(s)	Success message ‘Account created for (username)’ is displayed.

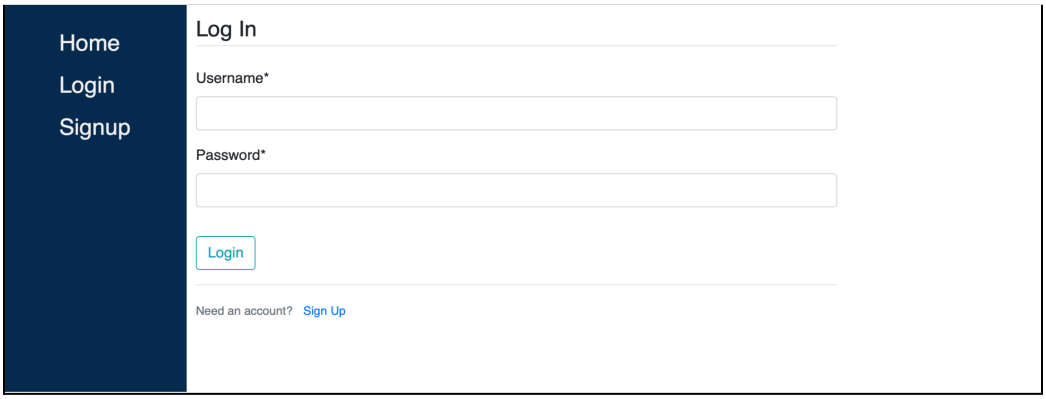


Sign up page



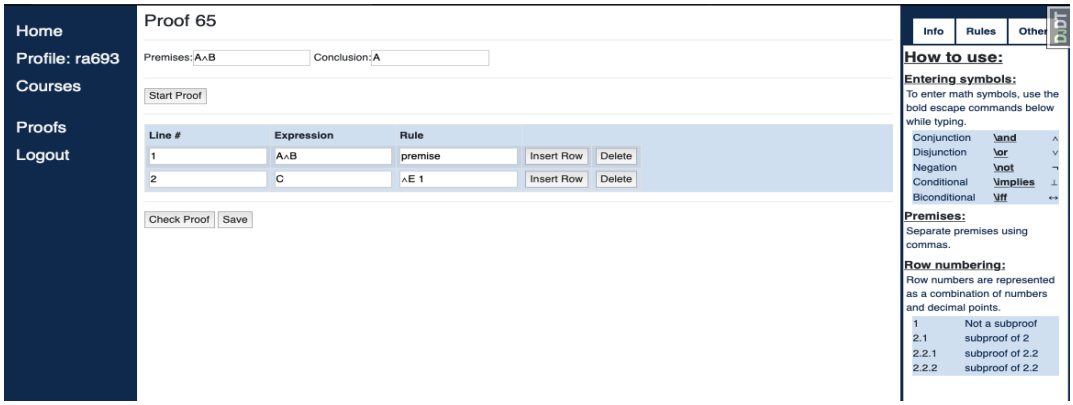
Sign up form

Use Case 2	**A User Logs In**
Actors	Students, Instructors and Admins
Brief Description	This use case describes how a user logs into their account.
Basic Flow of Events	The basic flow starts when the user arrives at the proof tool's main home page. 1. User clicks the Login field on the left bar 2. User fills in their username and password into the respective fields 3. User clicks the Login button 4. The user is signed into their account
Alternative Flow of Events	When the user attempts login and enters invalid login information: 1. The system will detail the issue with the user's provided credentials. 2. If the user has an account, they correct the login errors and proceed with signing in according to the basic flow of events. 3. If the user does not have an account, the user decides to create an account by clicking the Sign Up which redirects them to the Sign Up page. 4. Once there the user creates an account and then proceeds to sign in after registering.
Pre-conditions	The user has previously created an account.
Post-condition(s)	User is successfully signed in.



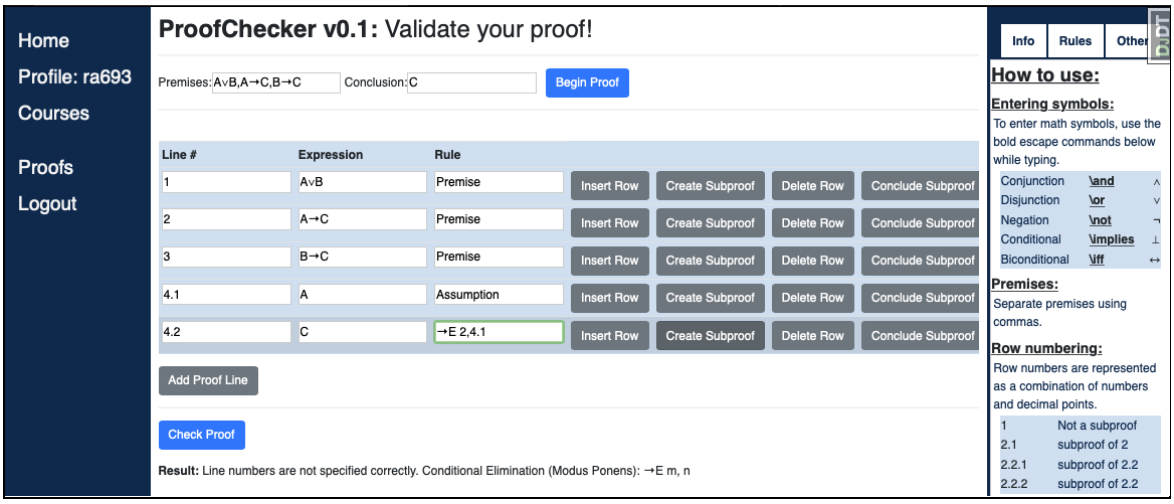
Login Page

Use Case 3	**Creating a Proof**
Actors	Users
Brief Description	This use case explains how a user creates a proof in the system
Basic flow of events	Basic flow begins when need is realized to create a proof that the user would revisit later. 1. User logs into the web-based proof checker system. The side nav bar is displayed. 2. User clicks on the ‘proofs’ link from the side nav bar. 3. A new page loads with ‘Add a new proof button’ on the top and the user clicks on it. 4. New page loads with ‘Premises’ and ‘Conclusions’ fields. User fills out the fields with those arguments. 5. If a user wants to check verification for proof statements, the basic flow of events follows from ‘checking proof for validations’ use case. 6. At any time if the user wants to save the proof, he/she clicks on the save button. 7. The proof is saved in ‘proofs’ collections for the user, and this concludes this use case.
Pre-conditions	User account is registered in the system and the user is logged in.
Post-condition(s)	The proof is saved under the ‘proofs’ collection for each registered user. The new created proof is displayed alongside the previously saved proofs if there are any.

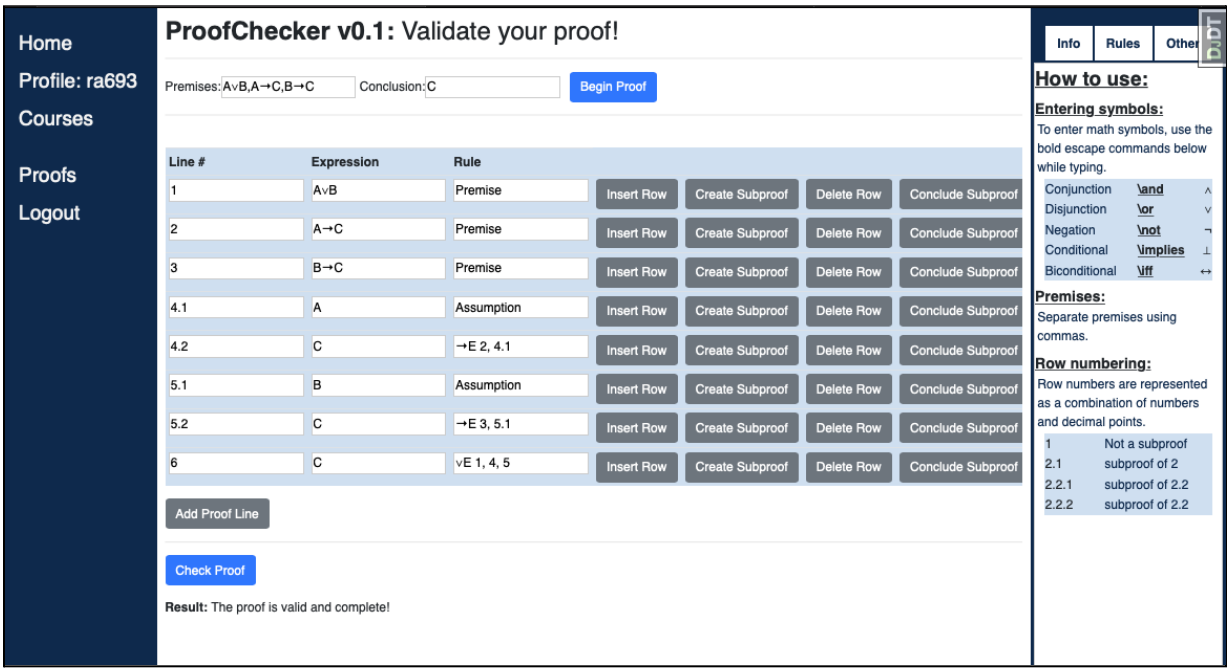


Add a new proof page

Use Case 4	**Checking Mathematical Proof for Validation**
Actors	Users
Brief Description	This use case explains how a user validates a mathematical proof
Basic flow of events	Basic flow begins when a user visits the web-based proof checker tool and the home page is displayed. 1. User clicks on the 'ProofChecker' link in the homepage. 2. The proofchecker page is displayed. There are two fields to enter premises and conclusion, respectively. User enters those arguments and clicks on 'Begin Proof' button. 3. A table with Line no, Expression and Rule columns appears. The line no is automatically generated. For the first row, the tool also pre-fills expression and rule fields. 4. Users are also provided with following buttons for each row: a. Insert row: Insert a row after the current row. b. Create Subproof: creates a subproof under a proof statement. c. Delete Row: deletes the current row. d. Conclude Subproof: It would conclude the current subproof. The line numbers are updated accordingly. 5. Users can check the validation of the proof by clicking on 'Check Proof' button. 6. The tool checks the validation of each proof line using TFL logic and displays a response to users. If the statement is valid the response 7. If any line is invalid, the system displays error messages that also specify line number and error type. 8. Users re-enters proof statements for validation. 9. If all the lines are valid and proof is complete, this use case concludes.
Pre-conditions	N/A
Post-condition(s)	Success message 'The proof is valid and complete!' is displayed.

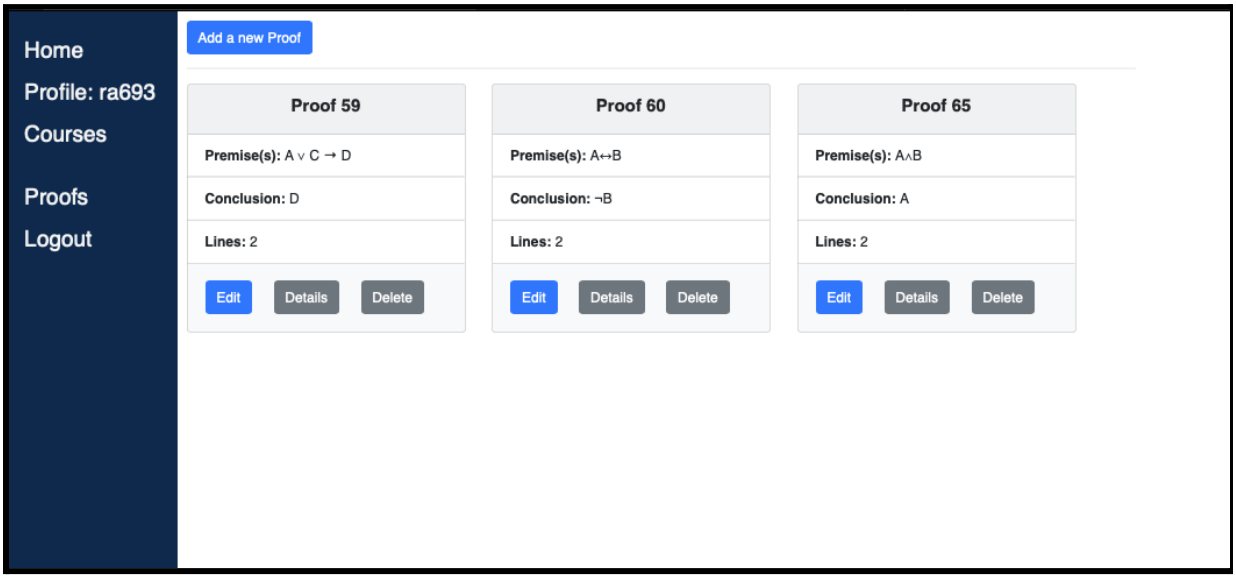


ProofChecker tool with error message



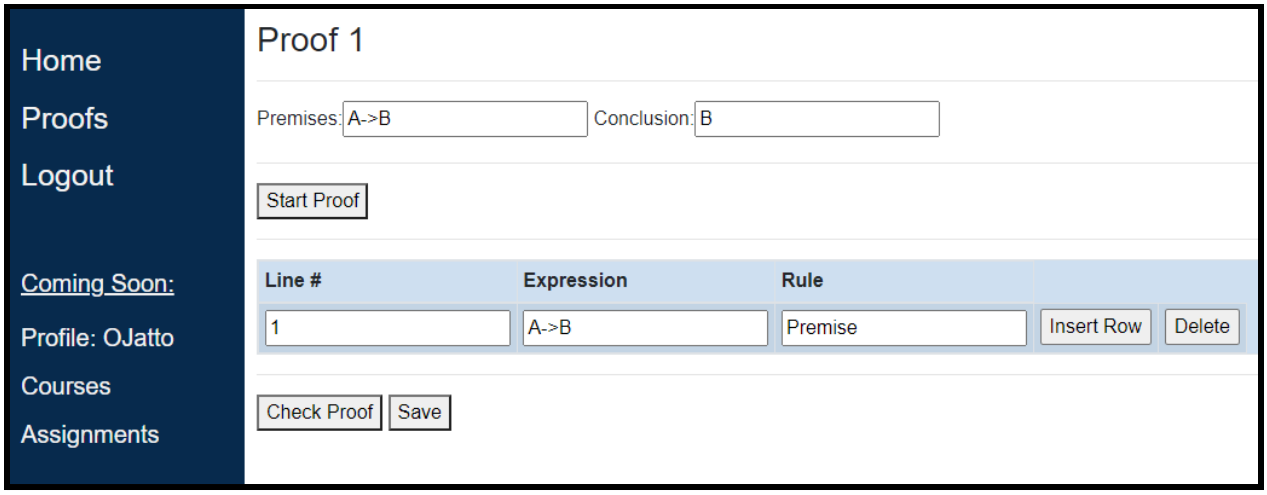
ProofChecker tool response after a valid and completed proof

Use Case 5	**Viewing Existing Proofs**
Actors	Users
Brief Description	This use case explains how a user access the proofs he/she previously saved
Basic flow of events	Basic flow begins when need is realized to access a proof user had previously saved. 1. User logs into the web-based proof checker system. The side nav bar is displayed. 2. User clicks on the ‘proofs’ link from the side nav bar. 3. A new page loads with all the proofs he has previously saved. 4. This ends the use case
Pre-conditions	User account is registered in the system and the user has previously stored proof while logged in.



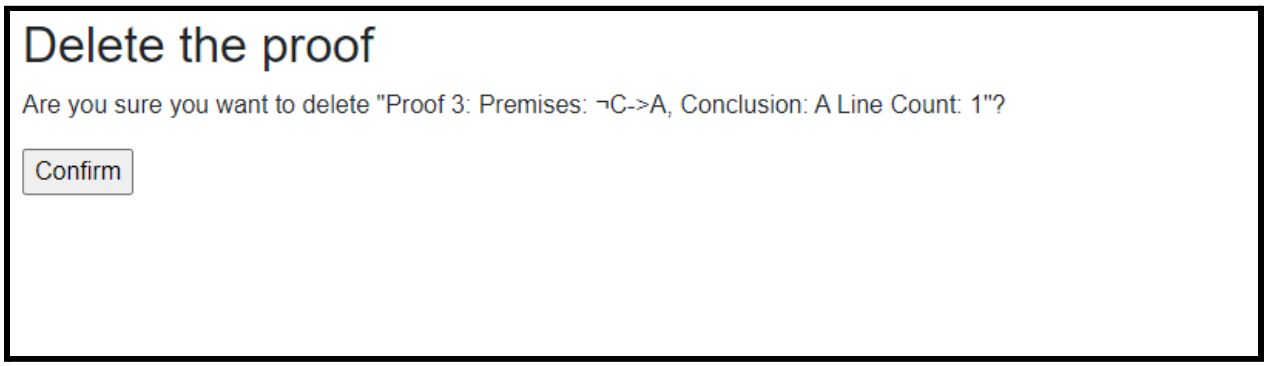
‘Proofs’ collection for a user account

Use Case 6	**Editing Existing Proofs**
Actors	Users
Brief Description	This use case explains how a user edits a proof in the system
Basic flow of events	1. The user signs into the Proof Tool 2. The user clicks Proofs which will direct them to a page displaying all their existing proofs 3. The user clicks the Edit Button on the proof they want to adjust. 4. The user is brought to a page where the proof they selected has all its fields displayed 5. The user makes adjustments to the values in one or more fields in the proof; or, the user adds or removes lines to the proof 6. The user completes their changes and hits the save button 7. The updates to the proof are stored in the database
Pre-conditions	The user already has proofs saved to their account.
Post-condition(s)	Upon clicking save, the page will update the user on the proof's status on whether it's complete or still has issues.



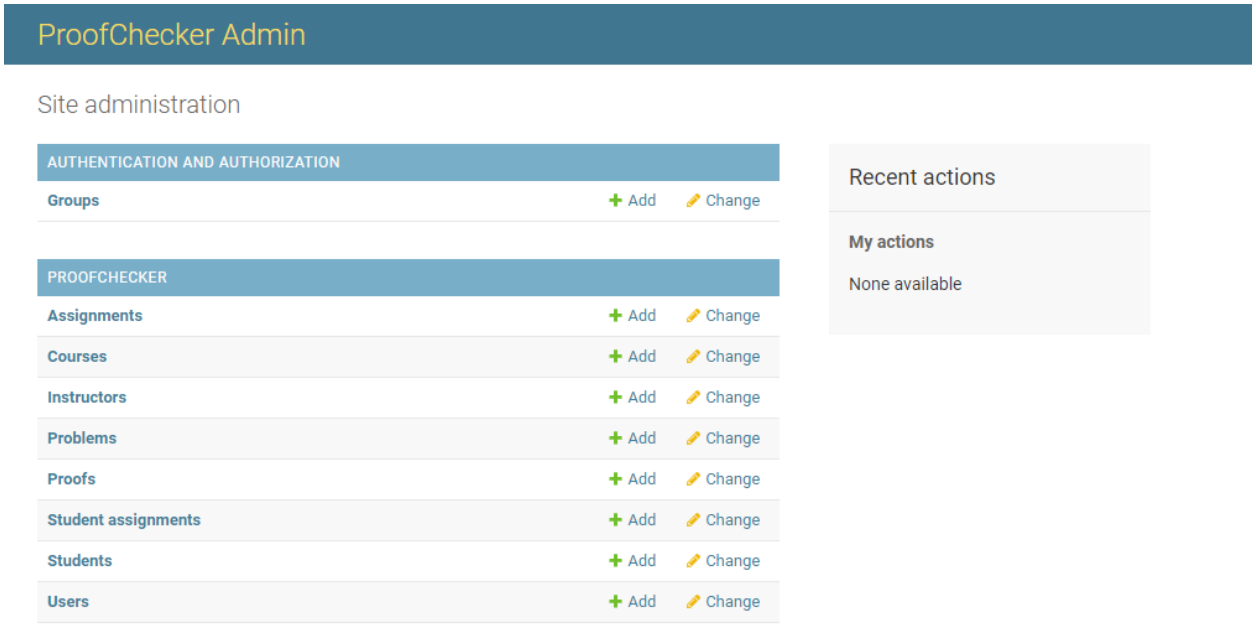
A page allowing the user to update their proof.

Use Case 7	**Deleting Existing Proofs**
Actors	Users
Brief Description	This use case explains how a user deletes a proof from the system
Basic flow of events	Basic flow begins when the user has a need for removing a proof from the system. 1. User logs into the web-based proof checker system. The side nav bar is displayed. 2. User clicks on the 'proofs' link from the side nav bar. 3. The user's proofs will be displayed with a Delete button on the lower right of each 4. The user clicks 'Delete' 5. The user is prompted for reassurance that they want to delete their proof 6. The user clicks confirm 7. The proof is deleted from their account
Pre-conditions	The user already has proofs saved to their account.
Post-condition(s)	The proof will no longer appear in the user's proof list.



The user is prompted to make sure they want to delete their proof

Use Case 8	**Managing the Database**
Actors	Admins
Brief Description	This use case explains how a user gets the Site Administration page to use various admin functionalities.
Basic Flow of Events	1. The user should open the admin page 2. The user logs in to the admin page with a superuser account 3. The user arrives at Site Administration and can directly add, edit, or remove various fields in the Proof Tool database like Assignments, Problems, Proofs, and Users
Pre-conditions	N/A
Post-condition(s)	The database will be directly changed



A page showing the various admin functions

Use Case 9	**Creating an Assignment**
Actors	Instructor
Brief Description	This use case explains how an Instructor can add a proof
Basic flow of events	Basic flow begins when an instructor wants to create an assignment. 1. The instructor signs into the Proof Tool. 2. User clicks on the Assignments link from the left menu. 3. A new page loads that allows them to add assignments. 4. The instructor clicks the Create Assignment button which lets them set an Assignment's title, due date, and instructions.
Pre-conditions	User account is registered as an instructor
Post-condition(s)	An assignment is created and all the instructor's students will be able to view its details and work on it.

Use Case 10	**Creating a Course**
Actors	Instructors
Brief Description	This use case explains how an instructor creates a course.
Basic flow of events	Basic flow begins when an instructor wants to create a new course. 1. The instructor signs into the Proof Tool. 2. The instructor clicks Courses on the left menu. 3. The user is brought a page where they can create courses. 4. The instructor clicks the Create Course button and will be able set its name, subject, add assignments to it, and add students
Pre-conditions	User account is registered as an instructor
Post-condition(s)	A course is created and the instructor will be able to add students to and associate assignments to it.