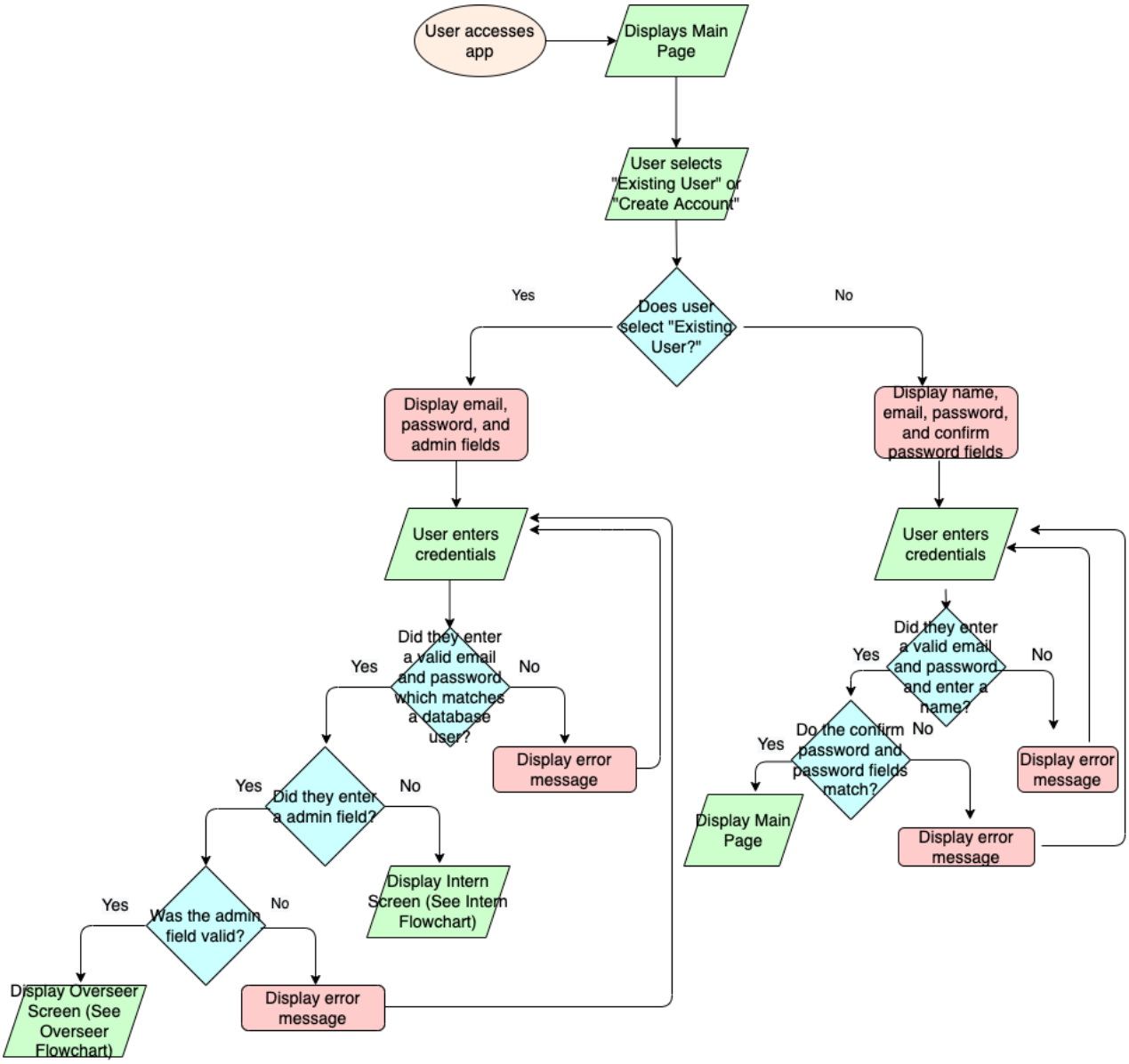
Criterion B: Design

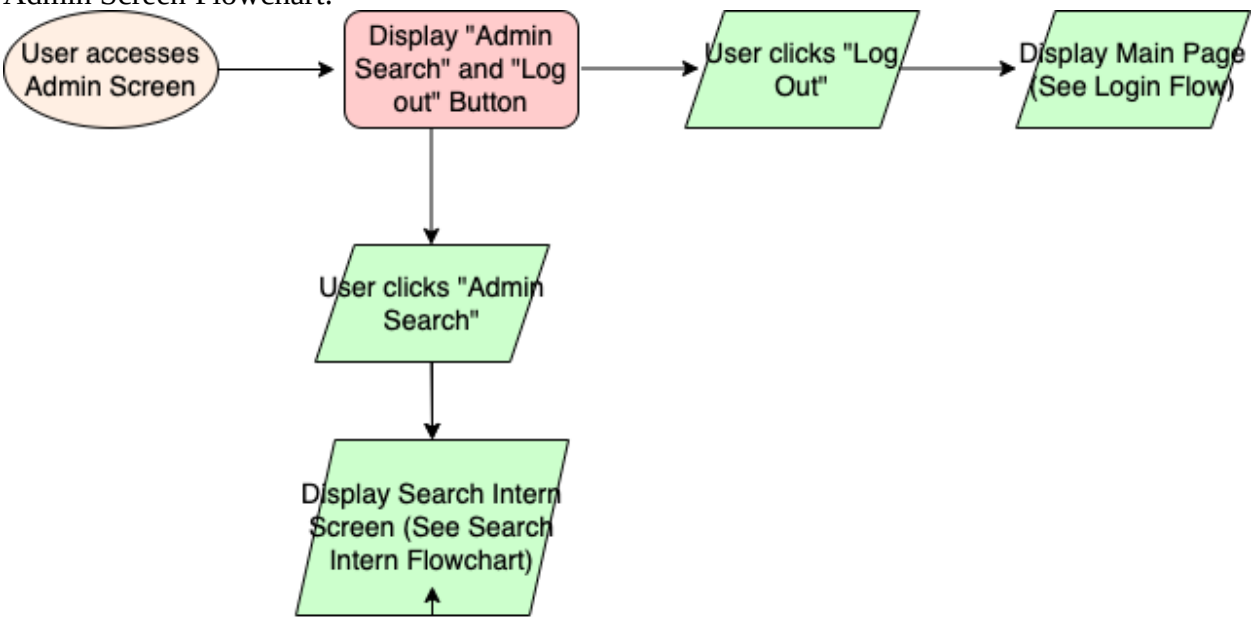
User Flow Diagrams:

See Appendix C for previous iterations.

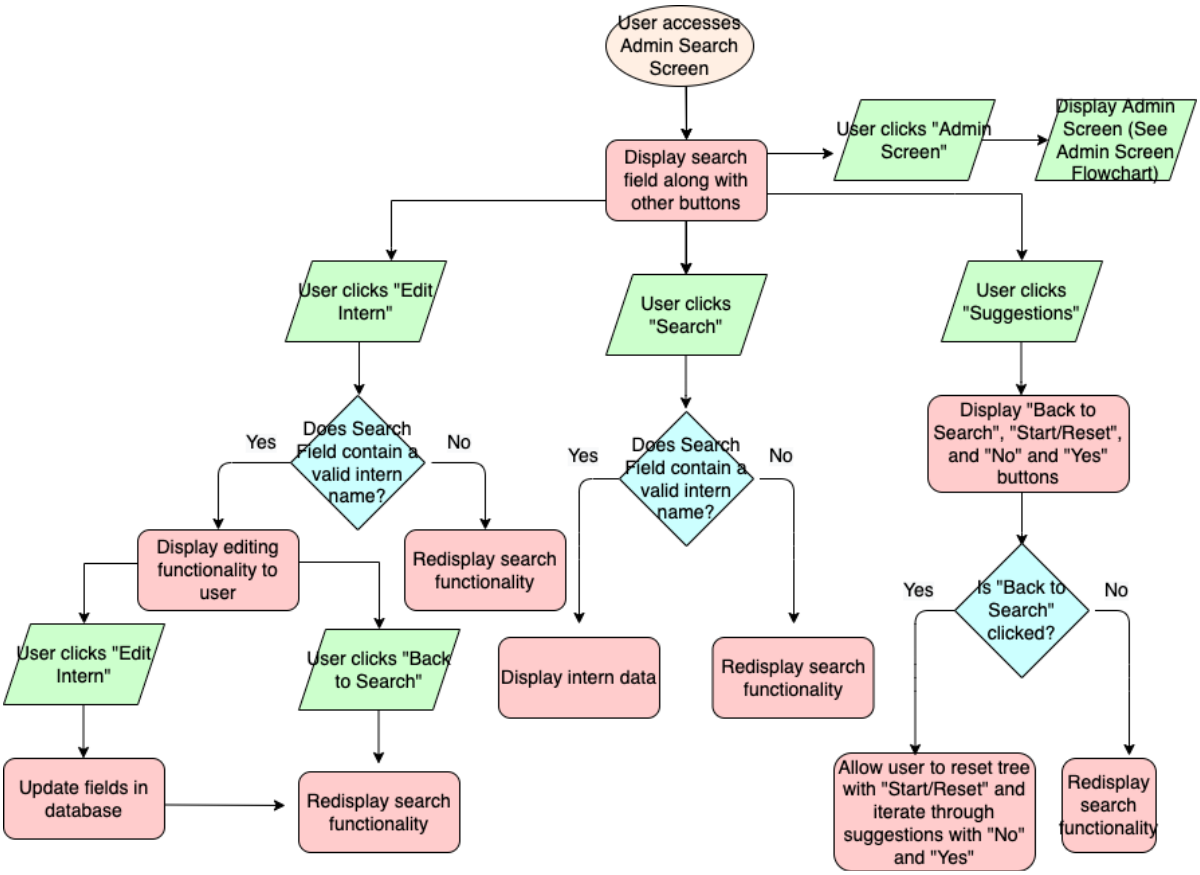
Log-in Flow:



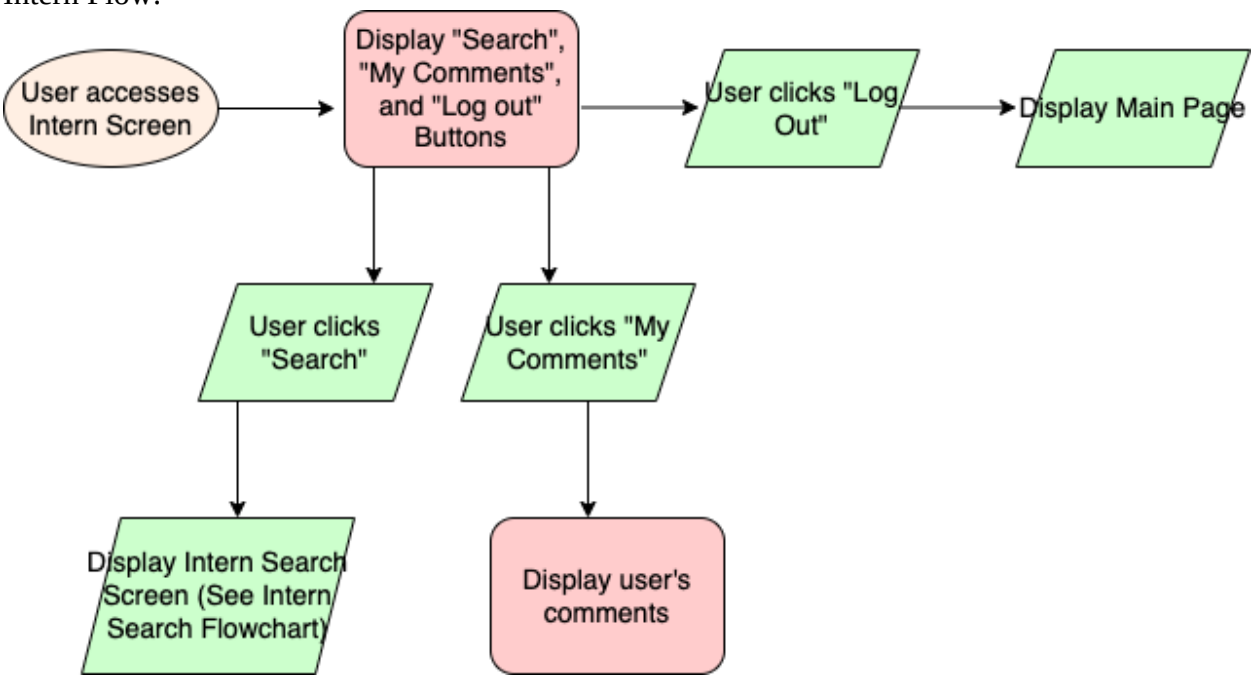
Admin Screen Flowchart:



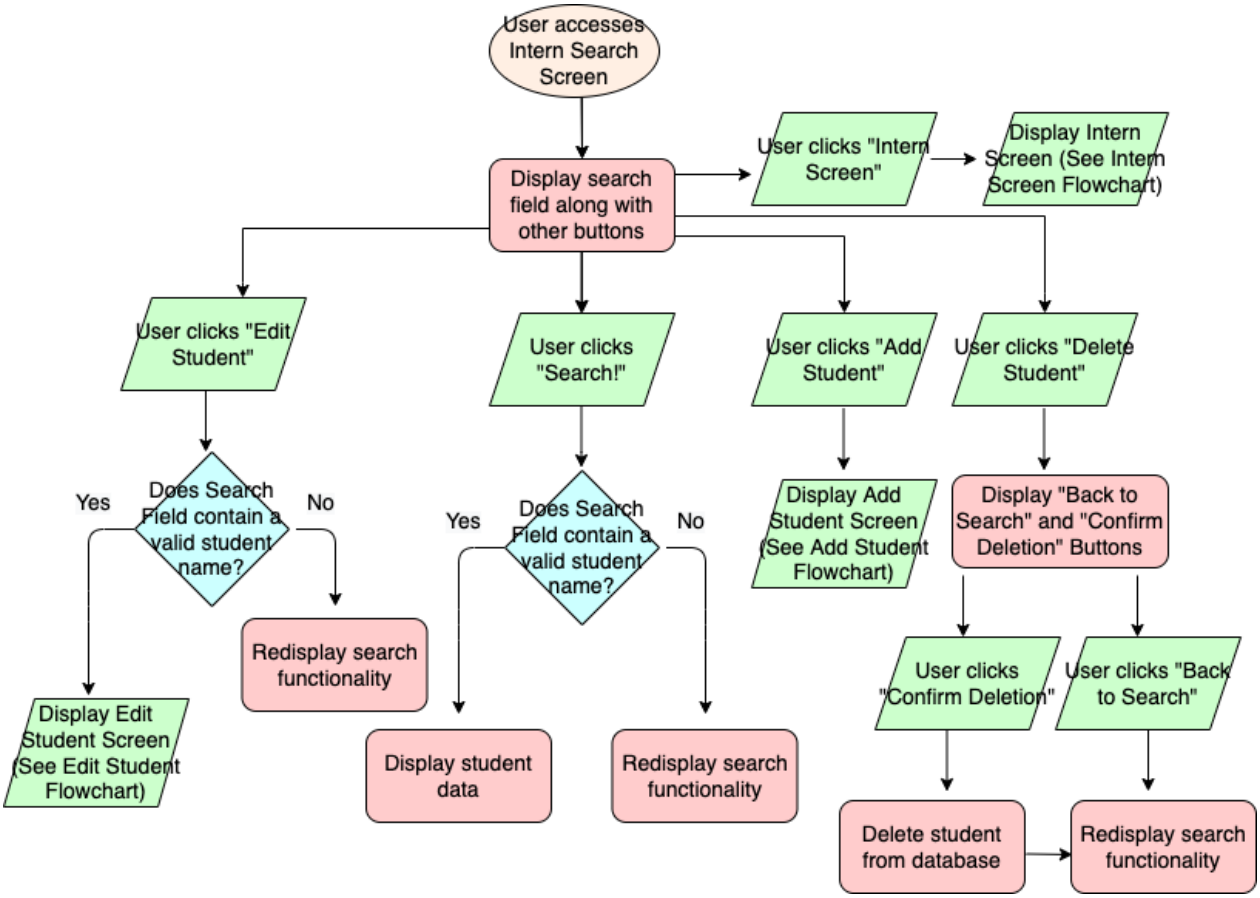
Admin Search Flowchart:



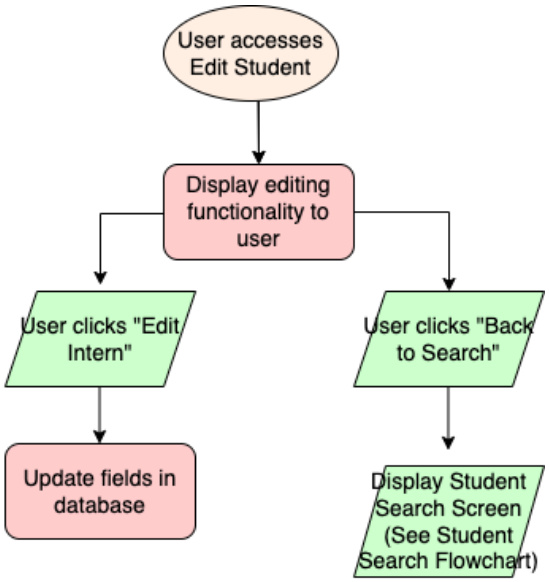
Intern Flow:



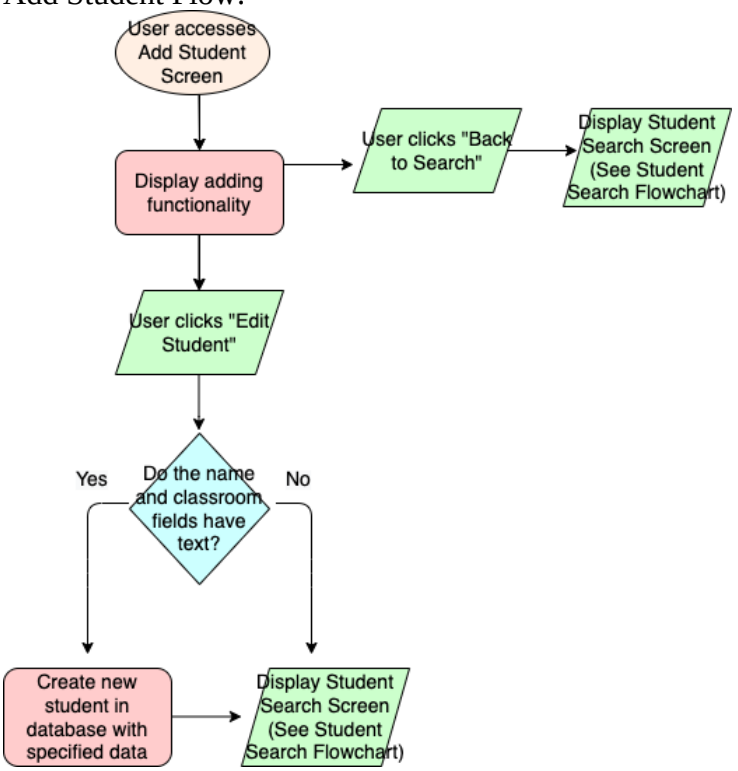
Student Search Flow:



Edit Student Flow:



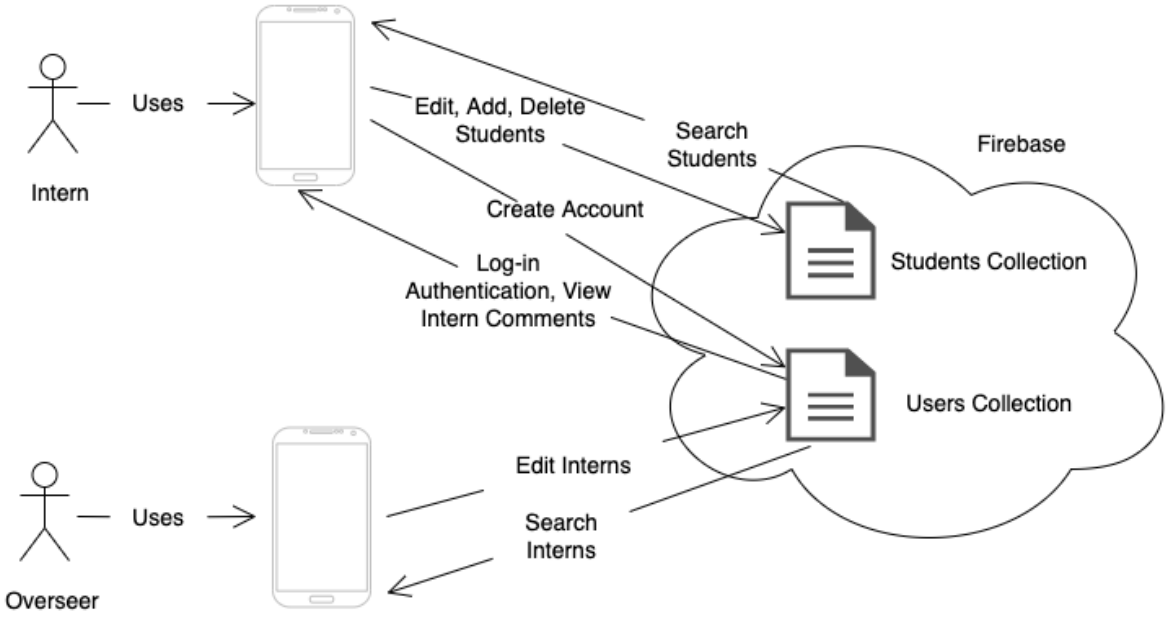
Add Student Flow:



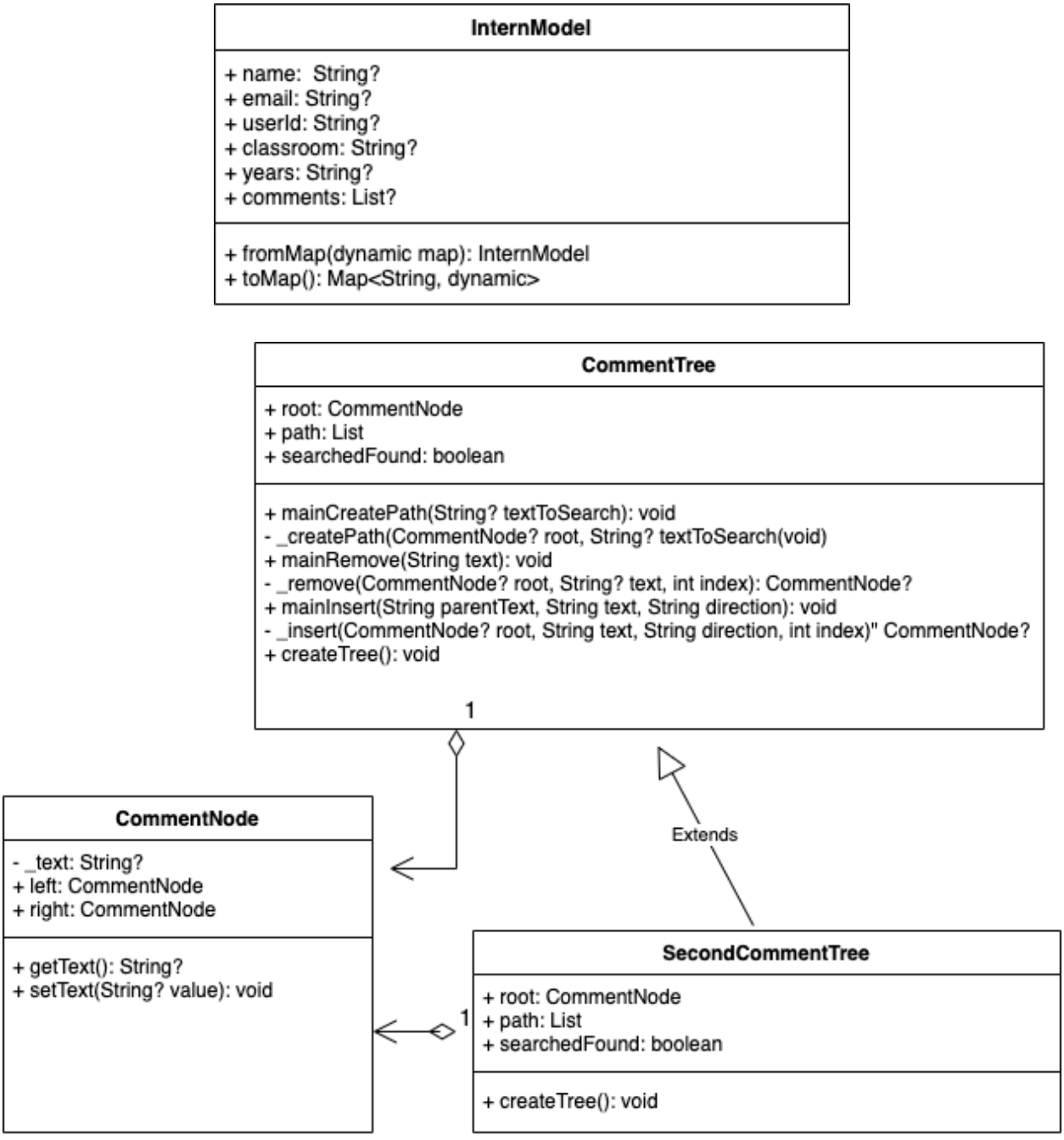
Data Flow:

See Appendix C for previous iterations.

The following diagram demonstrates how different users interact with the database, which holds two different collections: Students and Users. Both collections are composed of documents which are accessed by the users for several functionalities and were used due to their ease of accessibility and their clean segmentation between different types of data in students and interns.



UML Diagram:
See Appendix C for previous iterations.



I chose the inheritance relationship between the tree classes so I could reduce duplicated code and easily create instances of the classes in my OneYearSuggestion.dart and ExperiencedSuggestion.dart classes. Aggregation also was integral to my project, allowing me to

implement the CommentNode as a field in my trees. The UML Diagram was my best means to clearly communicate these relationships. Also, the InternModel class was instantiated in several other classes to show intern data, but it was impractical to fully show this in the UML Diagram.

Algorithms: See Crit C for in-depth explanations.

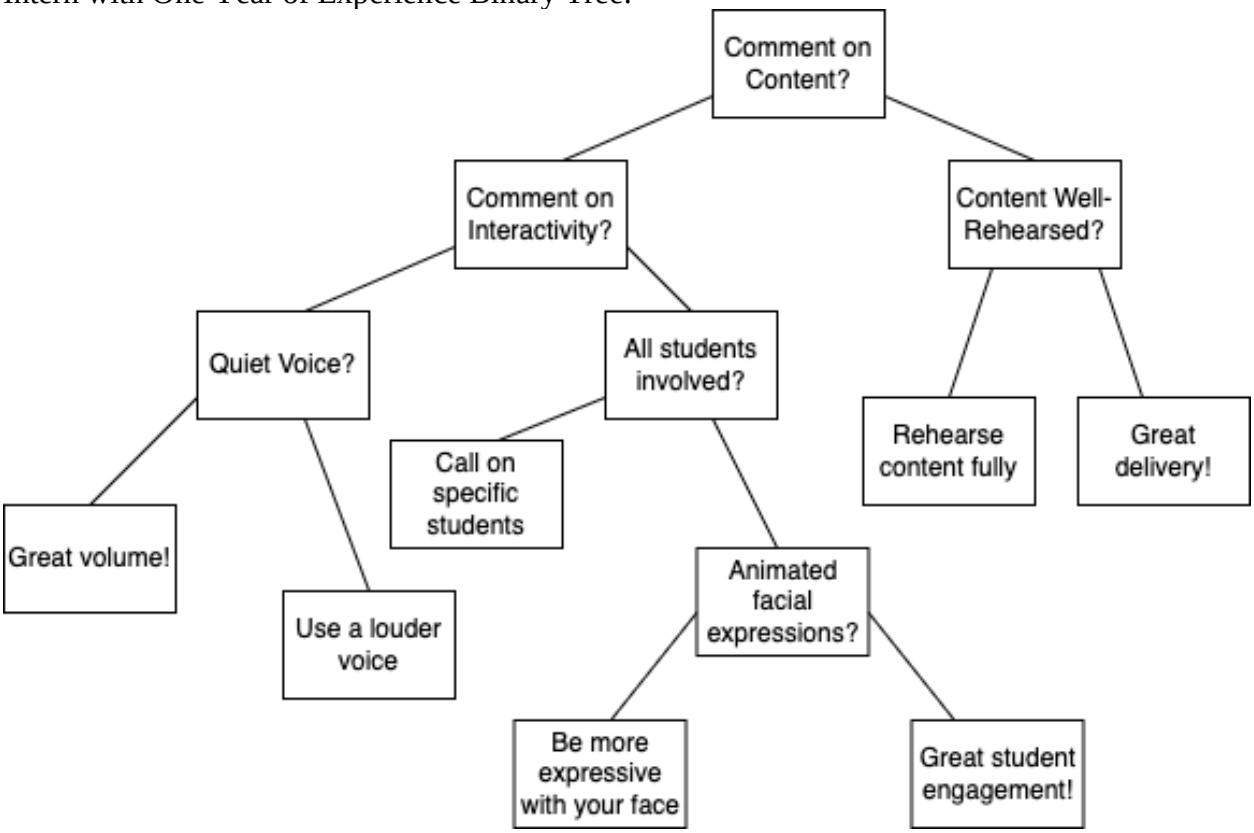
Algorithm	Functionality
Creating Path to Binary Tree Element	I utilized pre-order traversal to sift through the binary tree and add to a list path as it traversed down nodes. When it traversed upwards, these directions were removed. When the element was found, the list was not modified anymore so that the path was accurately reflected.
Insert in Binary Tree	To insert in the binary tree, an insertion function was recursively called as the tree traversed down its nodes using the created path. Once at the correct spot, a new node was created in the direction specified, with any nodes below rerouted to branch off the new node.
Remove in Binary Tree	To remove in the binary tree, a removal function was recursively called as the tree traversed to the correct node using the created path. Then, this node was set to null, nullifying its branches too as these need not branch off other questions in the tree.
Searching and Displaying Data	Utilizing a StreamBuilder and the Firestore collection, the documents of the database were iterated through. Once the current document was identified through checking the name of the document versus the search text field, its data was displayed in Text widgets.
Manipulating Data	Data was added, deleted, and edited utilizing the collections and the documents within them from Firestore. For editing, the document id specifically needed to be passed with Navigator through pages.

Data Structures:

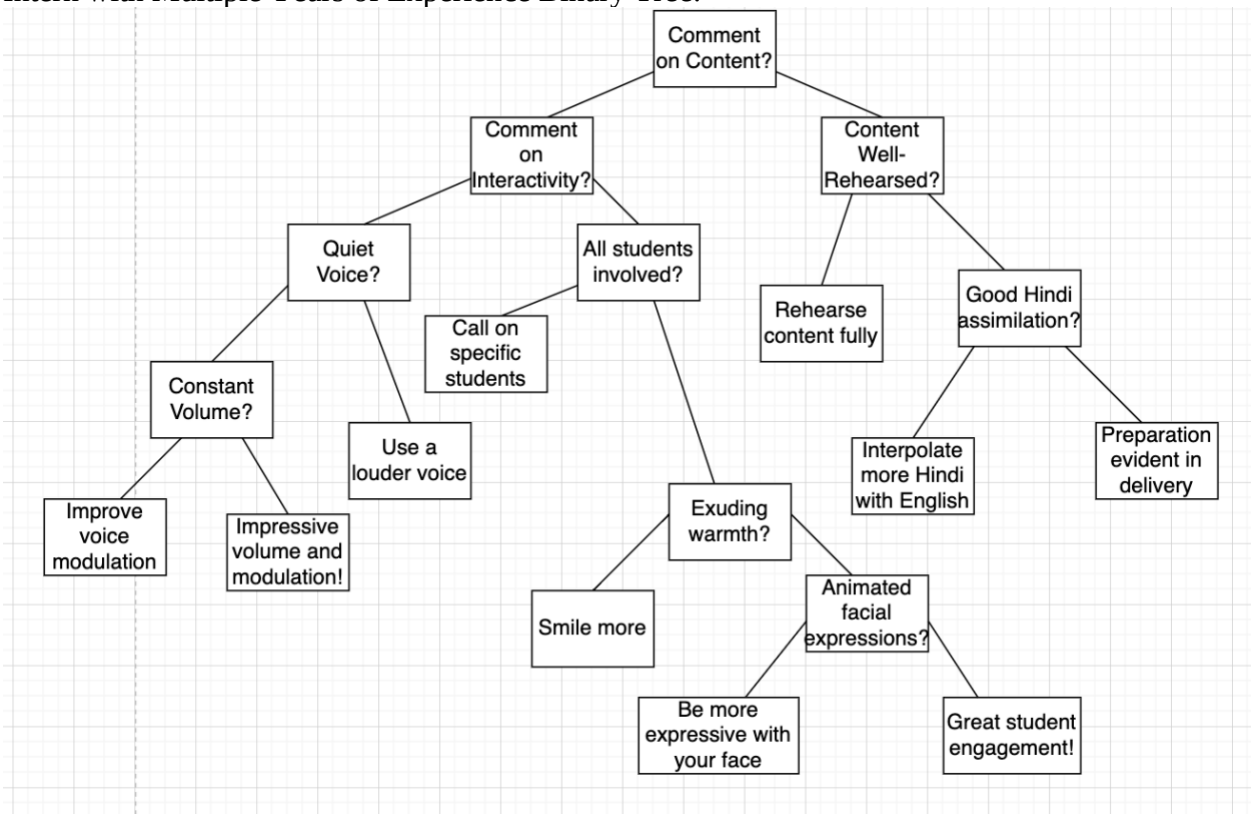
Data Structure	Reasoning/Explanation
Hierarchical Data Structure	I utilized a hierarchical data structure in my InternModel class, inspired by Backslash Flutter's video. This class served as a record-style data structure, having a fromMap and toMap method with the String? fields of userId, email, name, classroom, years and the object List? field of comments. The class allowed me to easily interact with Firestore to read and display values of the current, logged-in user, which proved integral to all intern functionality, especially the comment view which displayed data of the current user.
Binary Tree	Initially, I struggled with showing suggestions to the user, attempting to hard-code it through conditionals. The use of a Binary Tree proved to be much more dynamic, allowing for easy editing and implementation. The flow of the Binary Tree is documented below.

See Appendix C for previous iterations.
I iterate through the binary tree by going left for "no" and right for "yes."

Intern with One Year of Experience Binary Tree:



Intern with Multiple Years of Experience Binary Tree:



Test Plan:

Success Criterion Tested	Functionality	Test Process
1	Overseer Search	Checked by: (Logged in as overseer) • Overseer searches for intern in database • Overseer searches for intern not in database Succeeds if: • Data of corresponding intern only displays when intern found in database
2	Intern Search	Checked by: (Logged in as intern) • Intern searches for student in database • Intern searches for student not in database Succeeds if: • Data of corresponding student only displays when intern found in database
1, 3	Overseer intern-comment editing capabilities	Checked by: (Logged in as overseer) • Attempt to edit comments, years of experience, and classroom for two different interns Succeeds if: • After searching for intern, updated data displays • After logging in as the intern whose data was edited, they can view their updated data

2	Intern student-comment editing capabilities	Checked by: (Logged in as intern) • Attempt to edit one field of a pre-existing student • Attempt to add a new student with some fields entered • Attempt to edit some fields of the new student (at least one comment) • Attempt to delete student • Attempt to add one more student, with all fields entered Succeeds if: • After searching for student, updated data displays if edited or added • If deleted, student data does not appear upon search
3	Intern Access to Their Comments	Checked by: (Logged in as intern) • Viewing all comments from overseers even after it is updated once by the overseer
4	Suggestions	Checked by: (Logged in as overseer) • Click "Start/Reset", then "No" repeatedly till the suggestion stops changing for interns of 1 year of experience and 3 years of experience Succeeds if: • Suggestions for 3 years of experience ends at "Preparation evident in delivery" and "Good Hindi Assimilation?" is the preceding question • Suggestions for 1 year of Experience ends in "Great delivery!"
5	Creating Account	Should induce error message if "Create Account" pressed: • Empty name, email, password, or confirm password field • Password field has less than 6 characters • Confirm password and password fields don't match • Email field contains invalid email Succeeds if: • Account shows up in Firebase • User can log-in with set credentials
6	Login Authentication	Should induce error message: • Attempt log-in with correct username, wrong password • Attempt log-in with wrong username, correct password • Attempt log-in with invalid email • Attempt log-in with password shorter than 6 characters Should succeed and direct to respective screen:

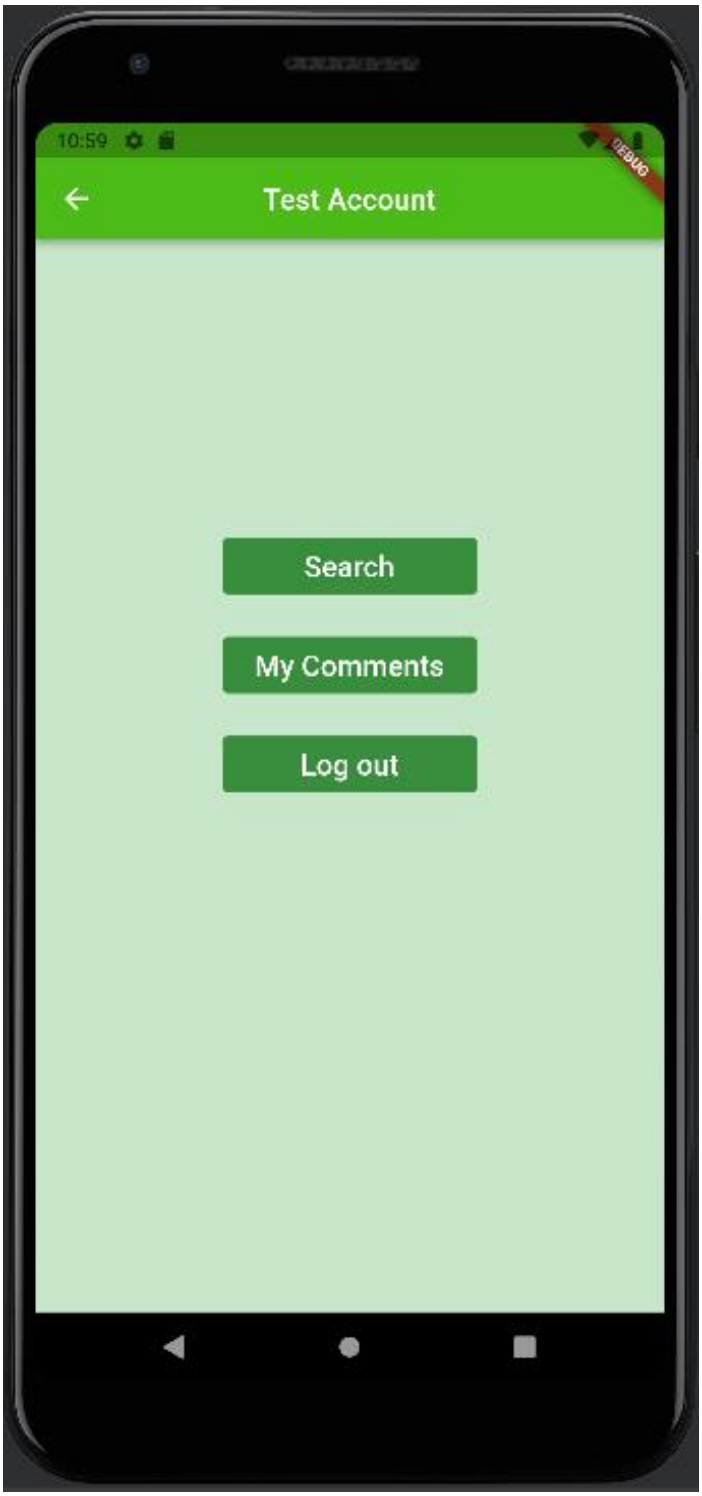
		• Log-in with correct credentials for a user and an empty admin field redirects to Intern Screen • Log-in with correct credentials for a user and correct credentials for admin field redirects to Admin Screen
6	Log-out	Checked by: • Click log-out button on Admin Screen • Click log-out button on Intern Screen Succeeds if: • Redirected to Home/Main Screen
7	Clean user interface for an optimal intern experience	Checked by: • Client, Advisor, and one other person will be given the app and a specific log-in and rough directions of a task to complete in the app Succeeds if: • App interface is easy and understandable and there are no problems with navigation of the app for the test subject

UI Flow:
See Appendix D for previous iterations.
The user starts out on this main page, with two buttons to choose from.

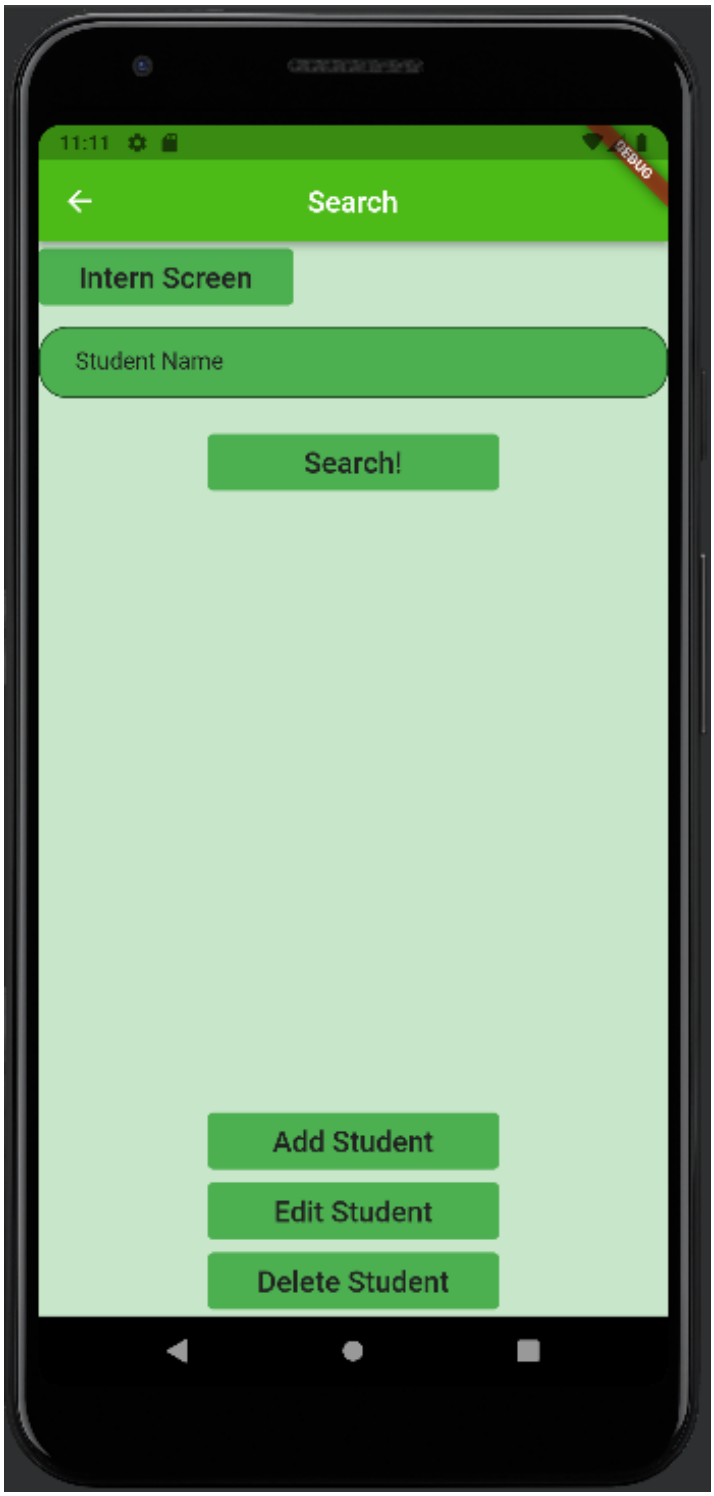
If the user clicks “Existing User”, they see this initial log-in screen. Some error validation is shown. If they enter the correct code for an admin, they are redirected to the Admin Screen, otherwise they reach the Intern Screen.

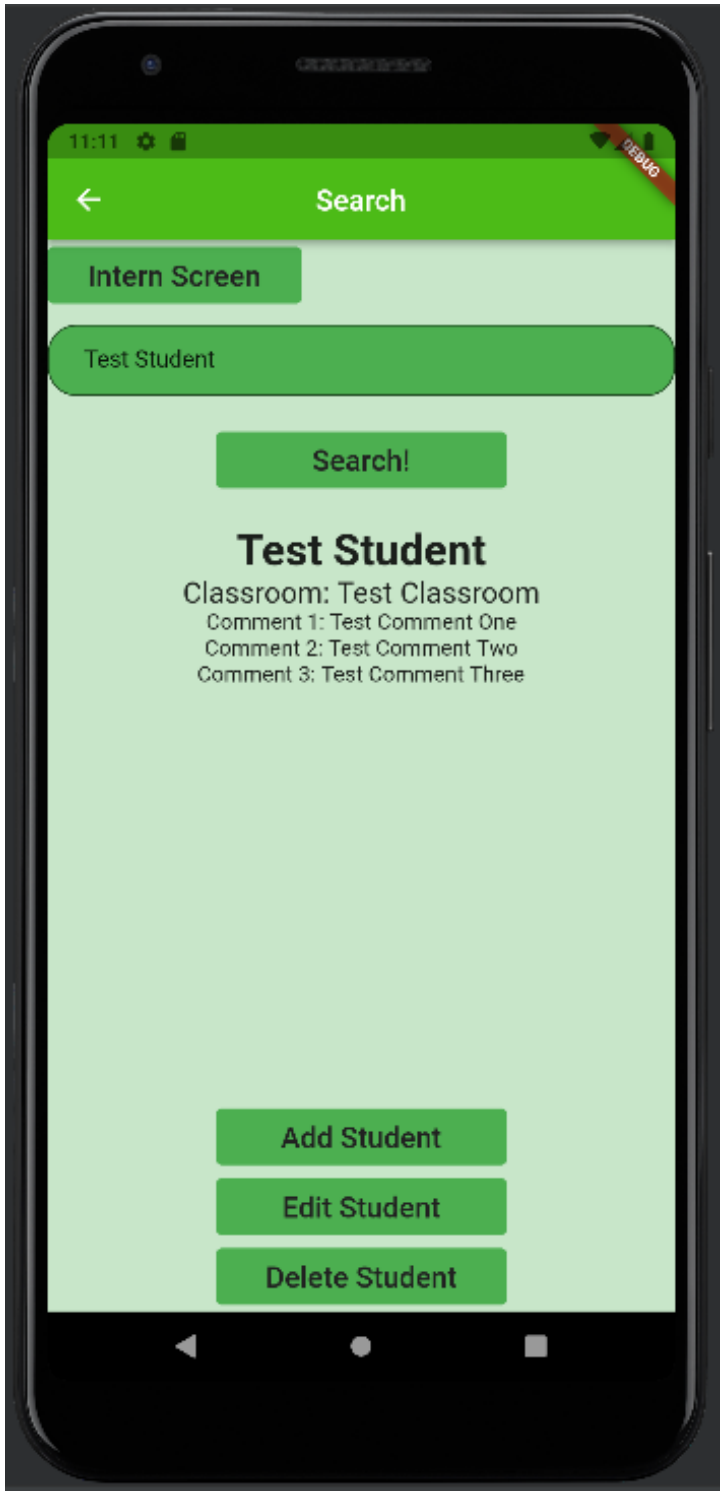
If they click “Create Account,” they are invited to type in their new credentials (error handling shown).

For user “Test Account,” the following Intern Screen is shown. When “My Comments” is clicked, they are redirected to a screen with their intern comments.

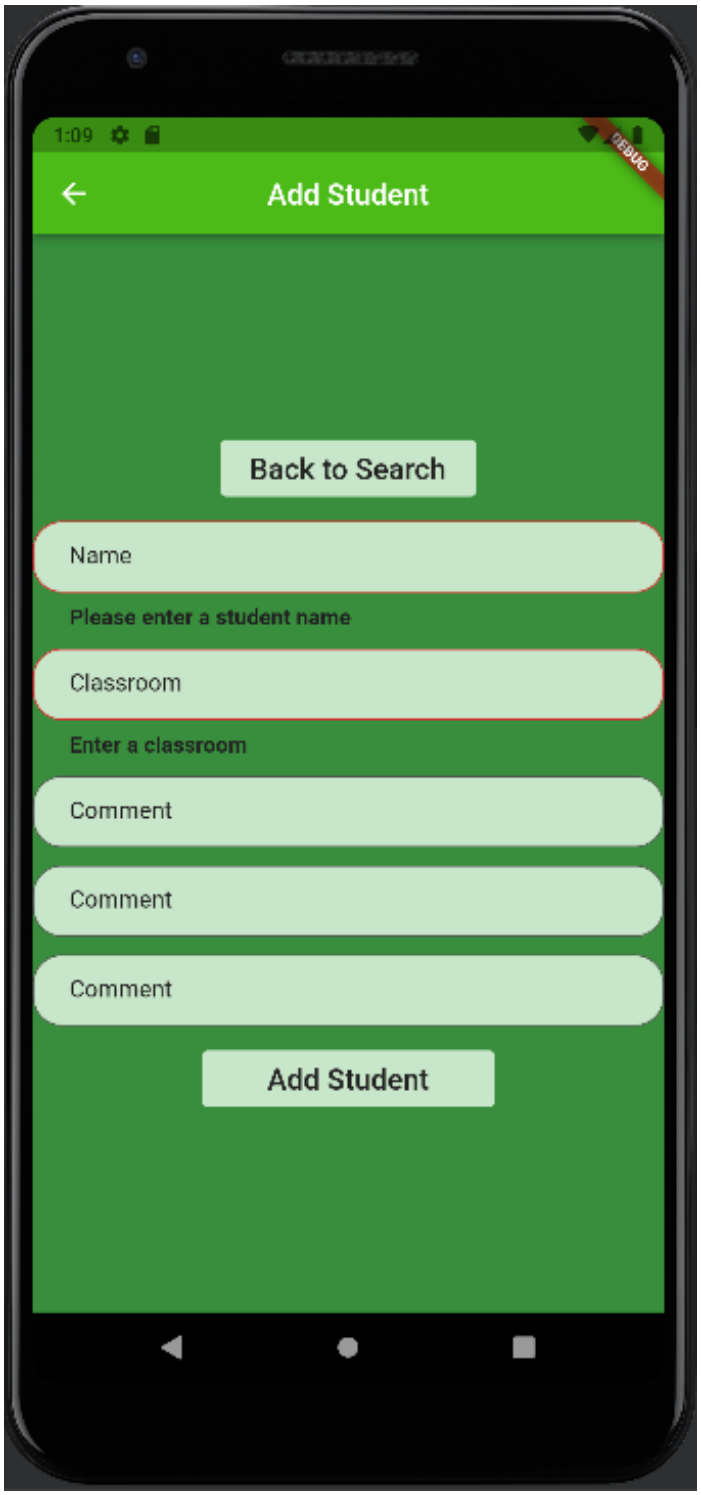


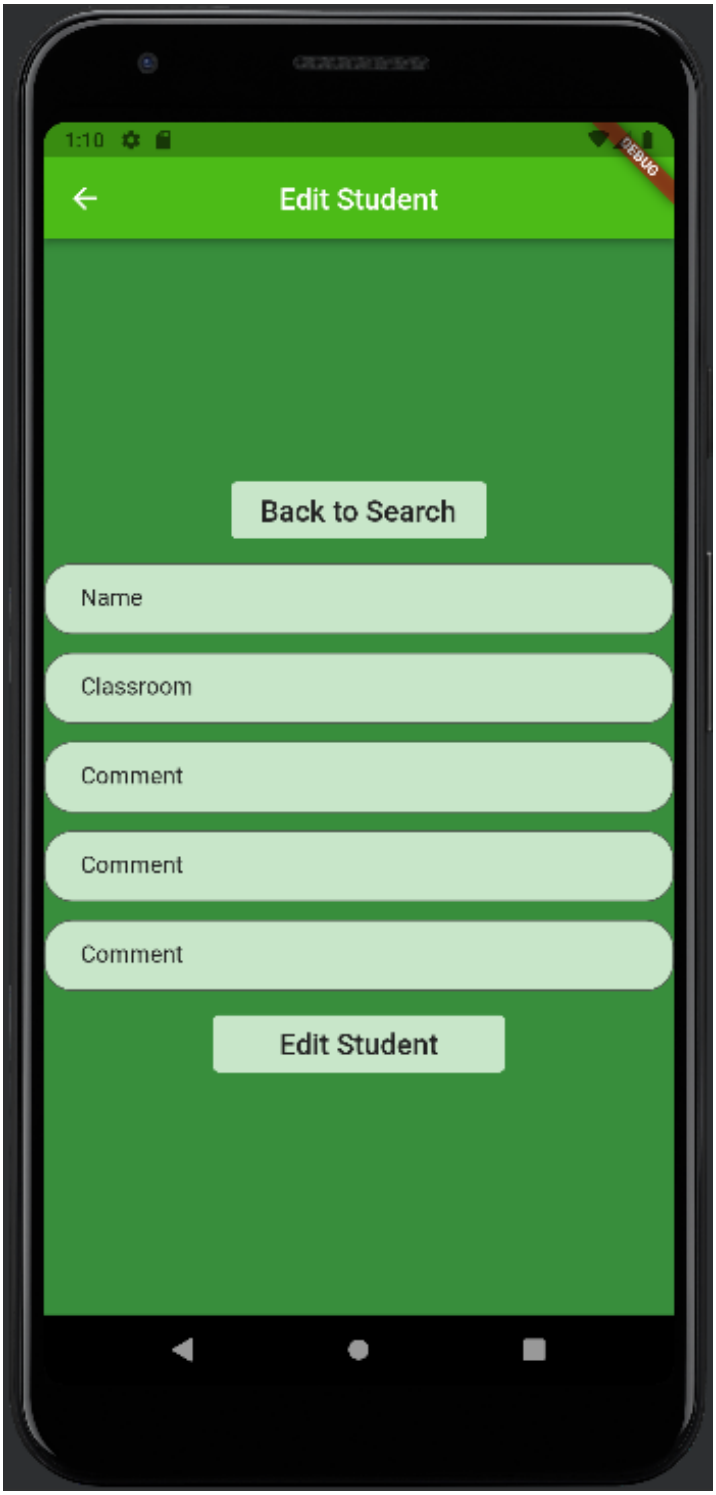
If “Search” is clicked on the Intern Screen, they are able to search for any students in the database.



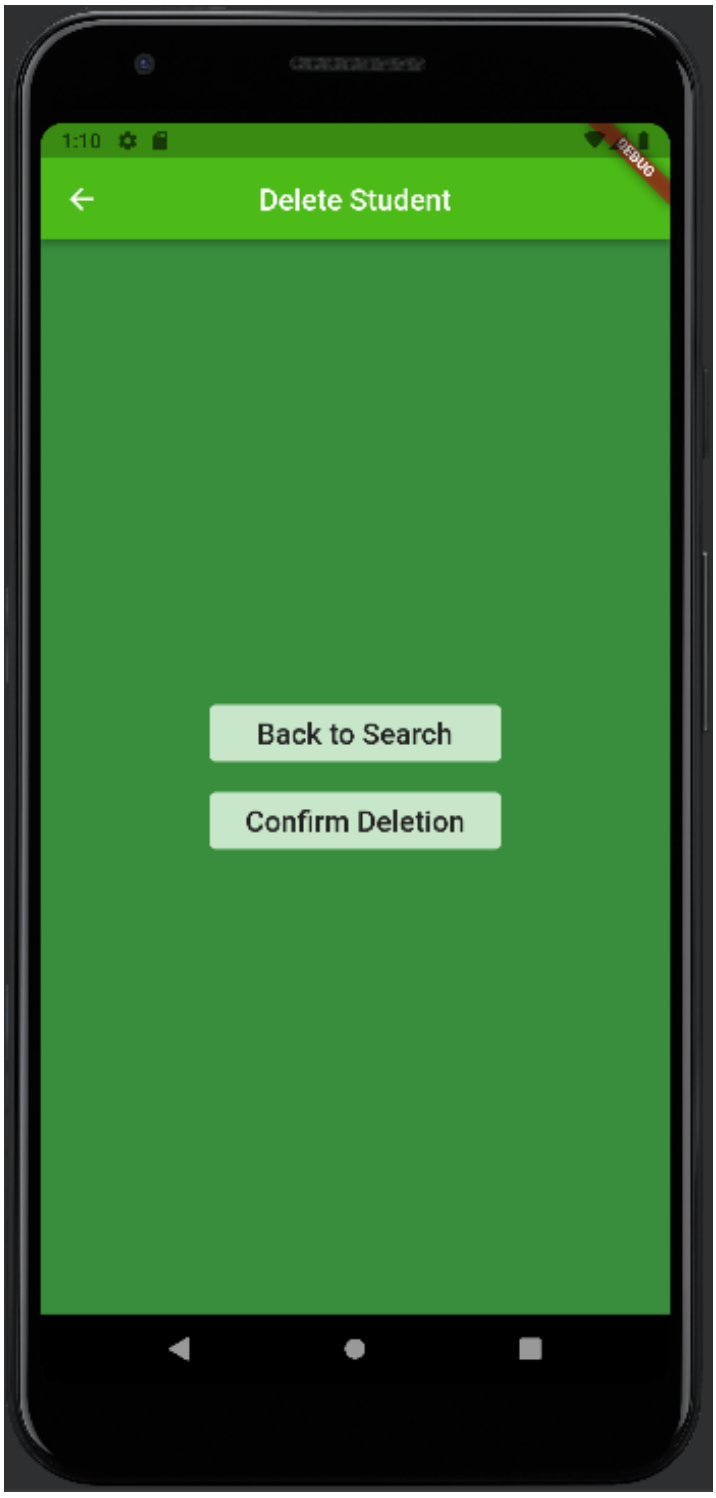


“Add Student” and “Edit Student” yield the following screens for the intern.

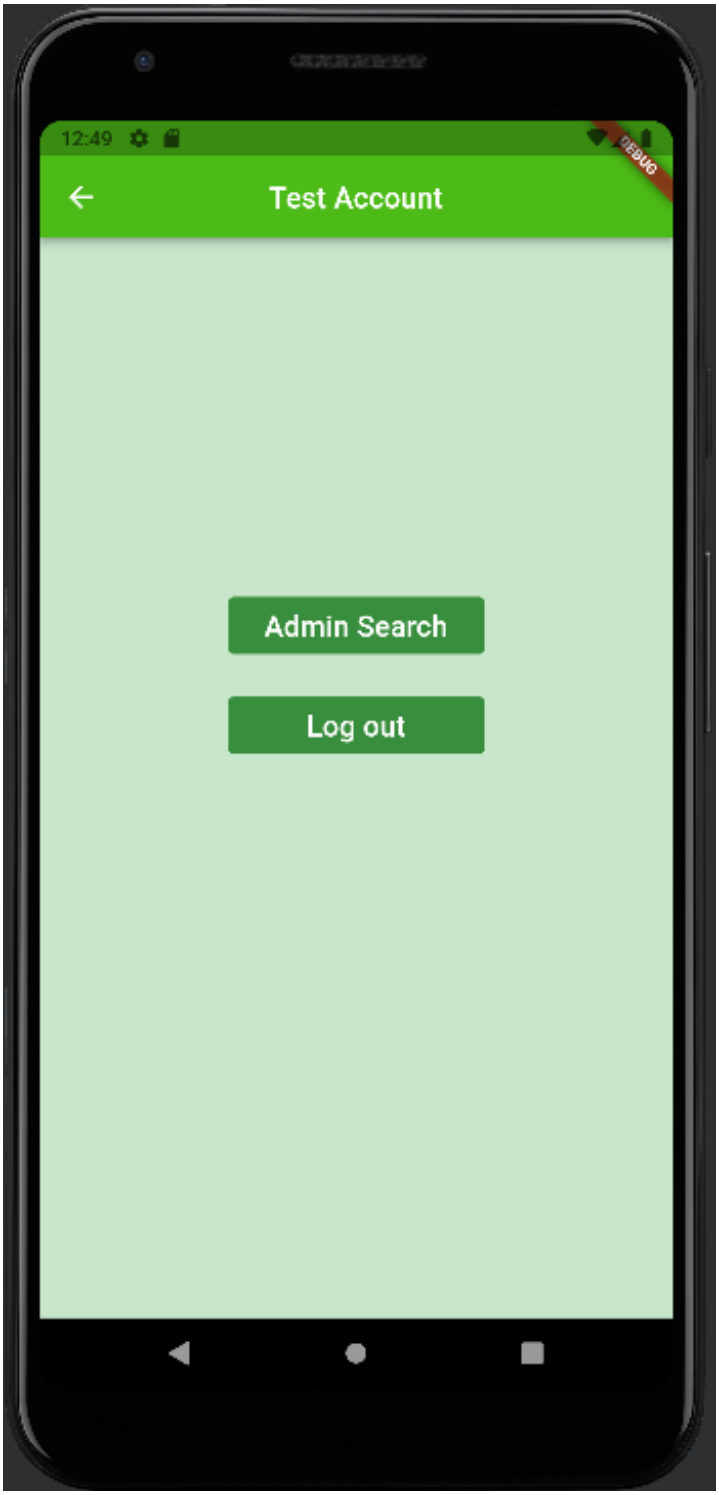




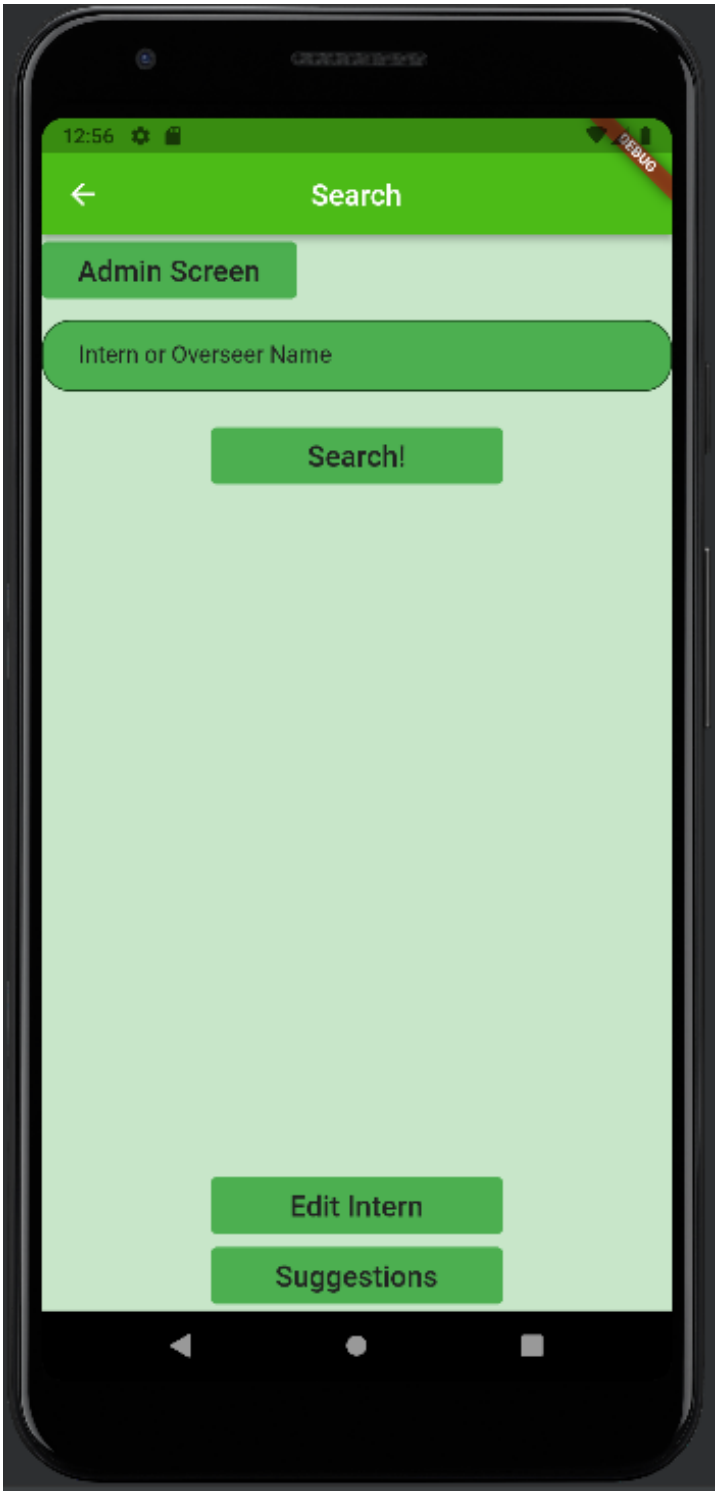
“Delete Student” shows the user a confirmation screen for their deletion.

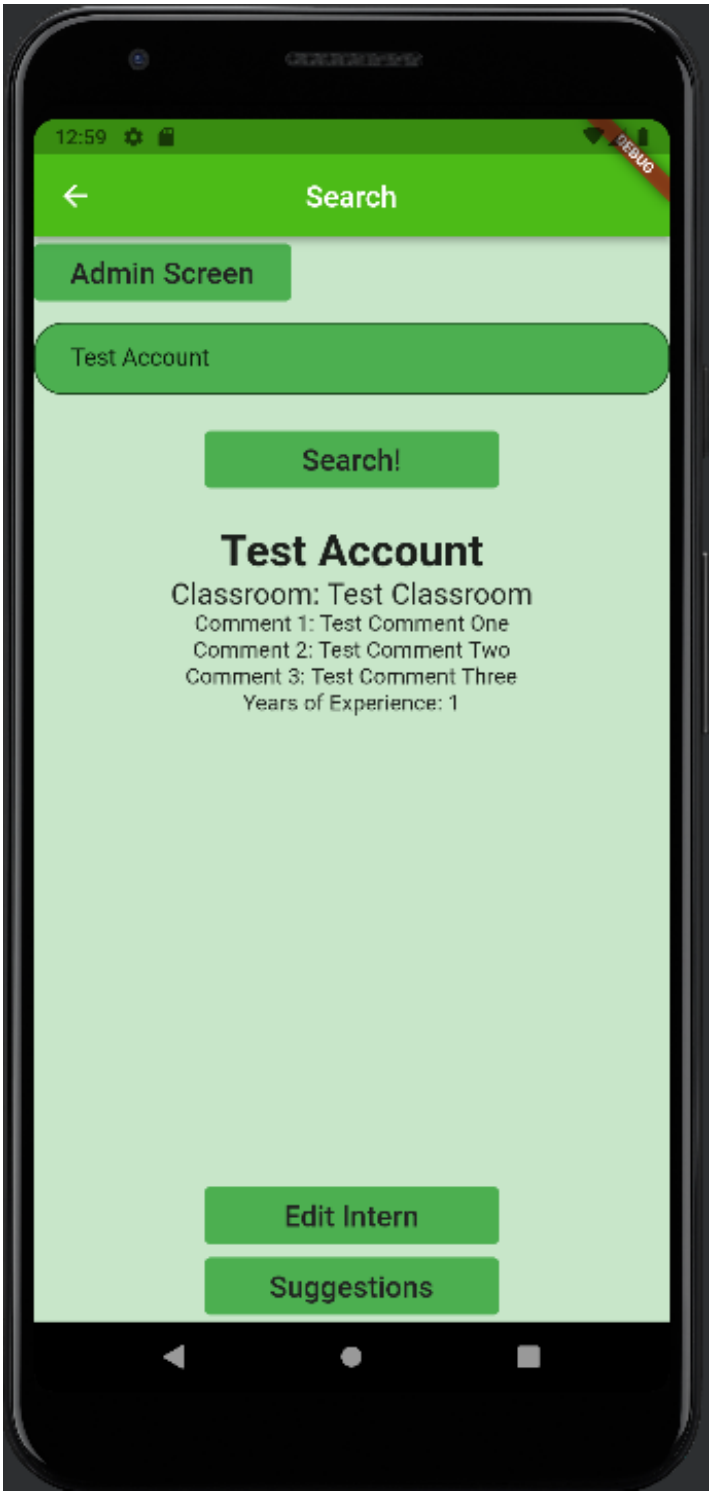


If the user is an admin, they arrive at this screen (admin Test Account).

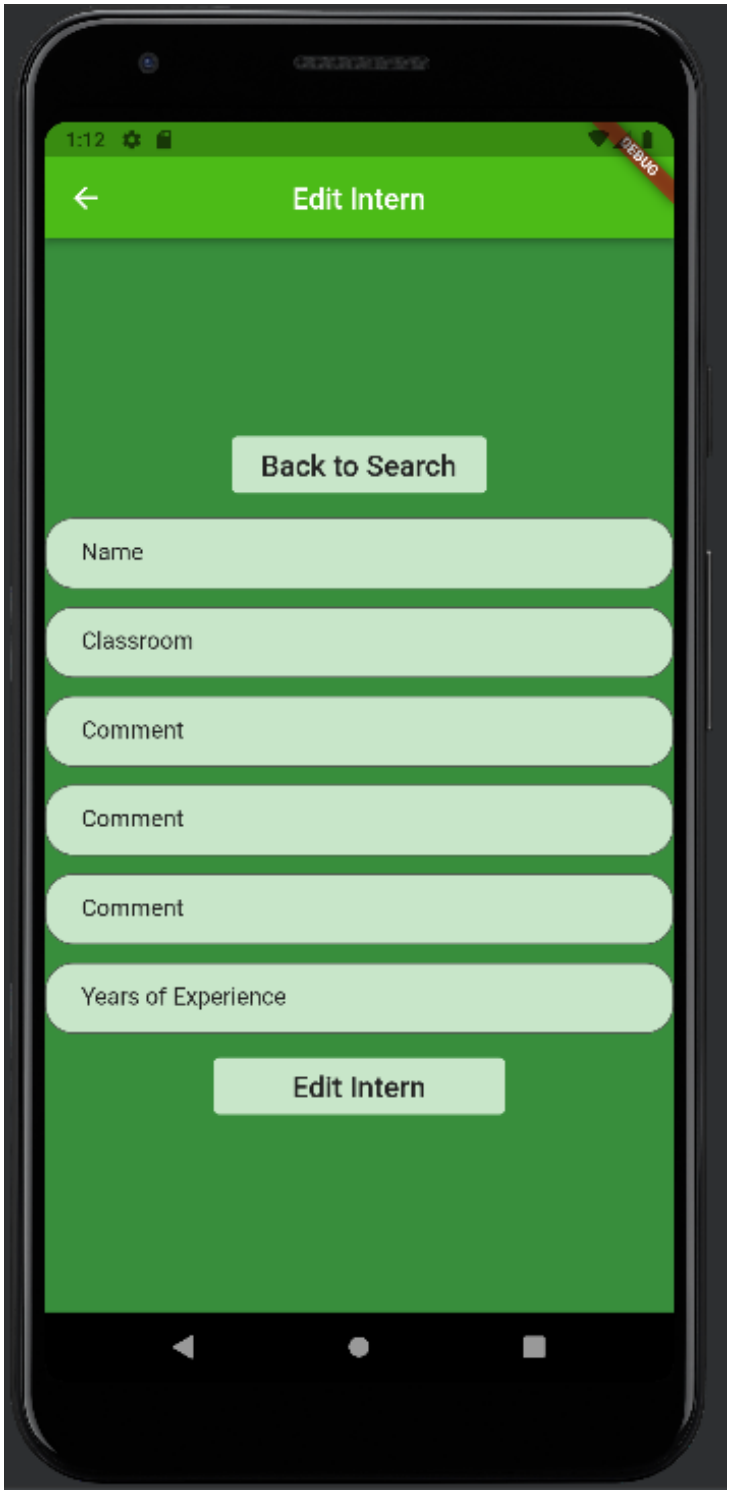


Once the admin clicks “Admin Search,” they are able to search the database for interns.

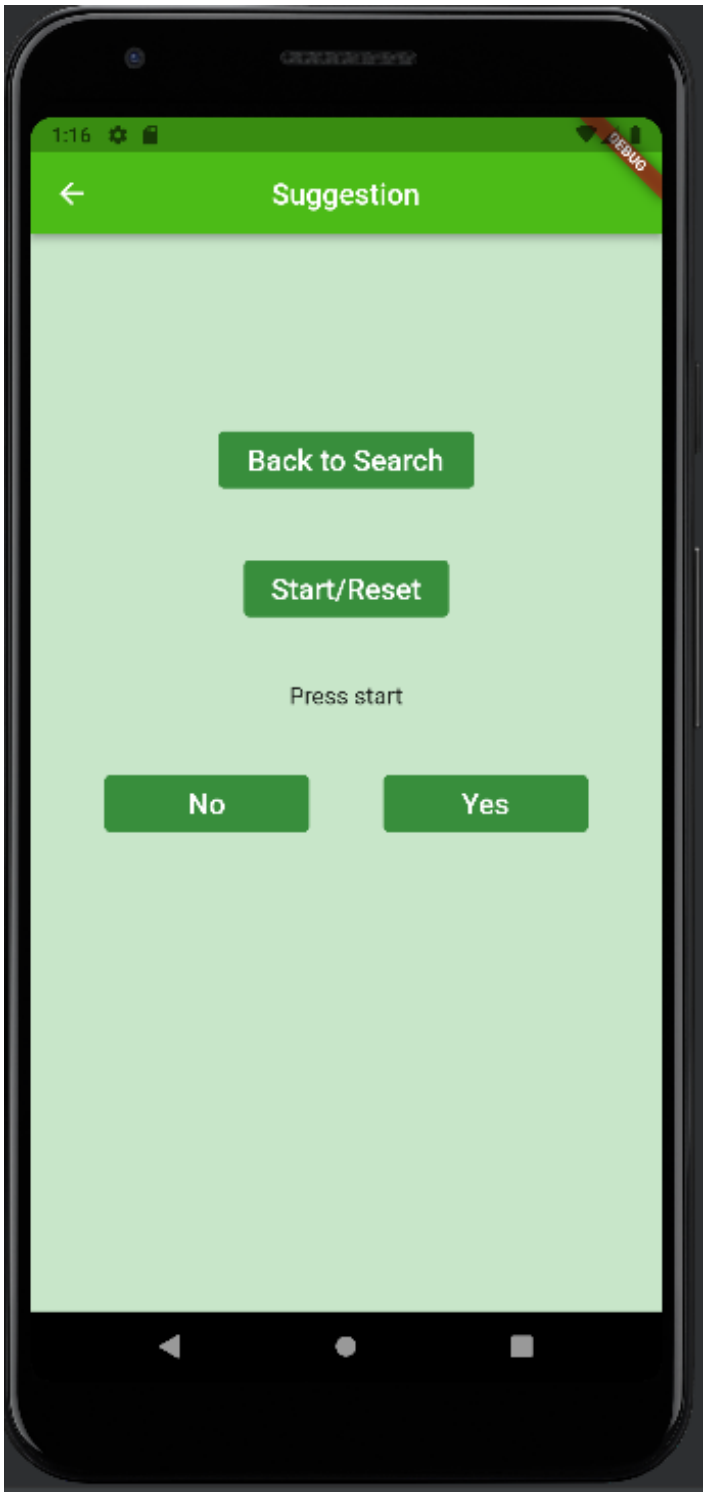




If the admin clicks “Edit Intern,” they are given text fields which they can edit if they choose.



If the user clicks “Suggestions,” they can sift through a tree of suggestions to ascertain he perfect comment.



Word Count: 446