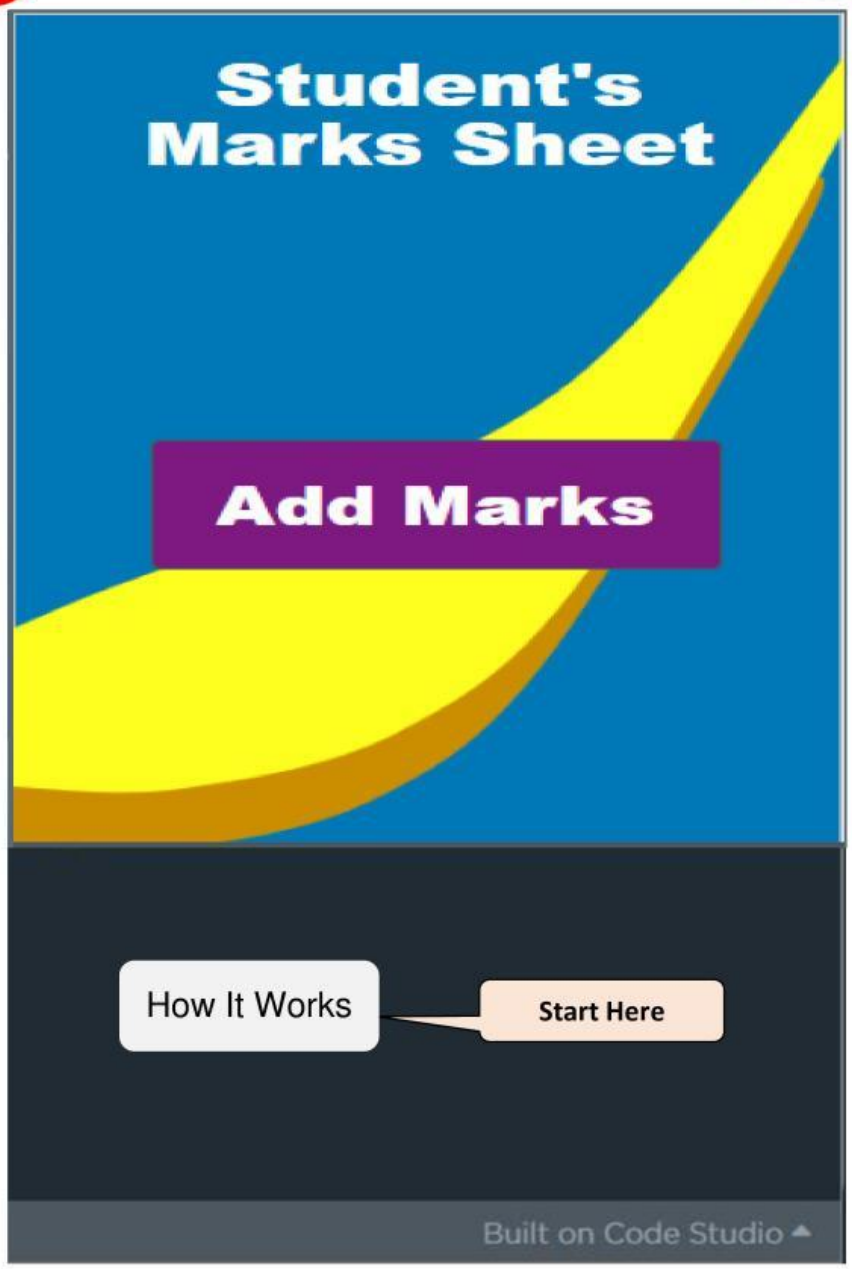
Project 91

DP EDUCATION

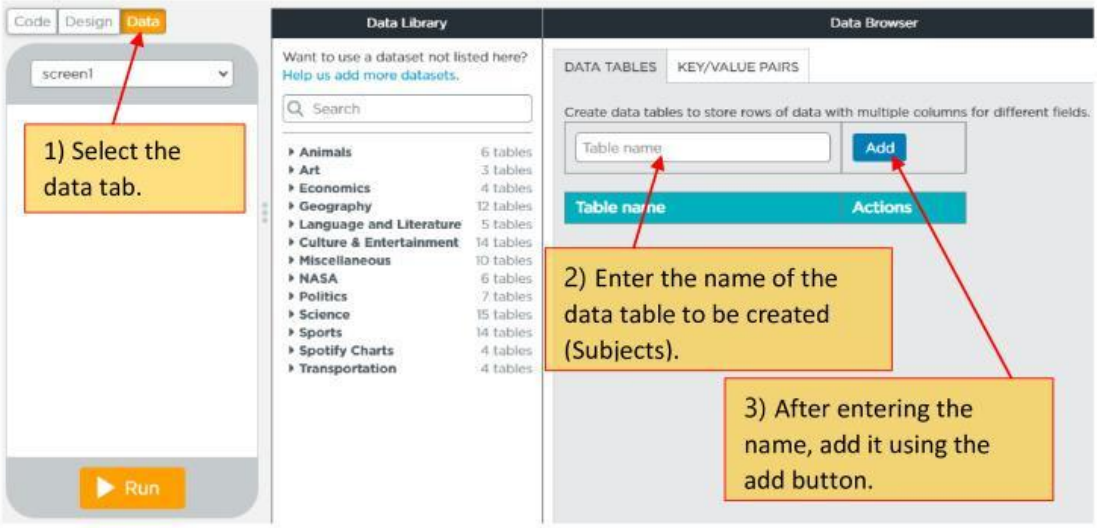
Coding School



- Let's create a report card to enter marks for students' subjects.
- Here, the name, year, subject and marks related to that subject of several children entered in a data table are updated through the APP or new marks are entered for a name in the data table.

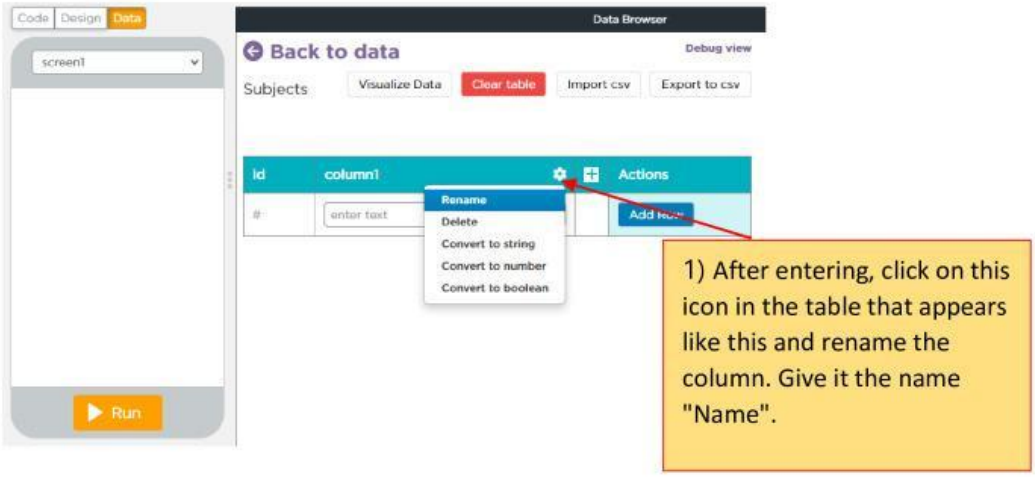
- For this, let's first see how to create the data table.

Step 1: While giving the name for the data table, let's first create the table containing the subjects. Name it as “Subjects”.

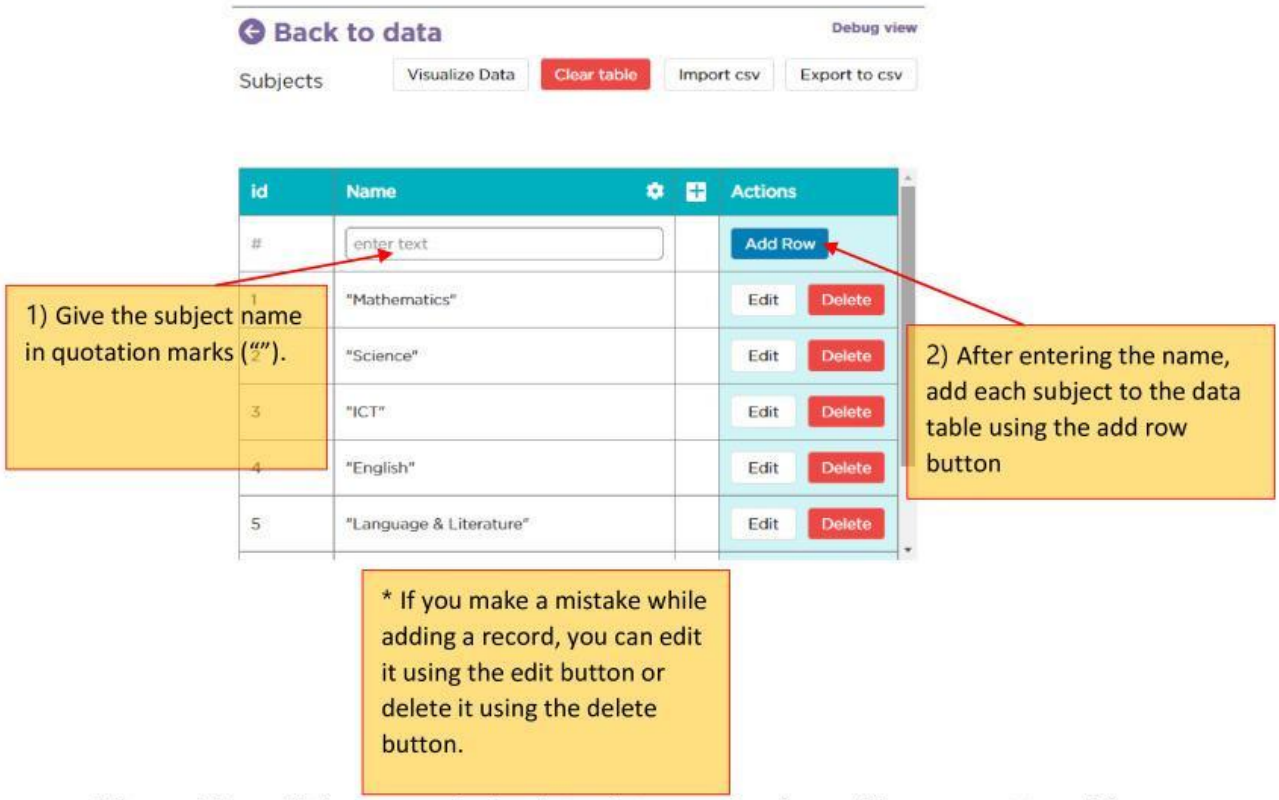


Then the data table added as below will be displayed below. You can enter the table by clicking on its name.

Step 2: After entering the data table, let's create its columns.

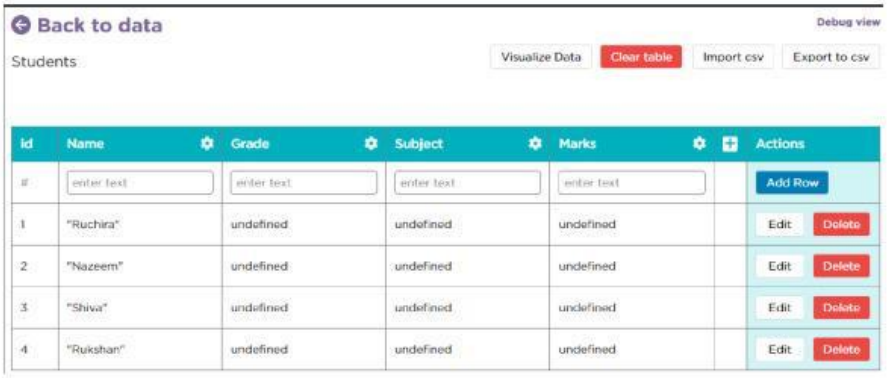


Step 3: After adding columns, let's see how to add data by adding rows to the table.

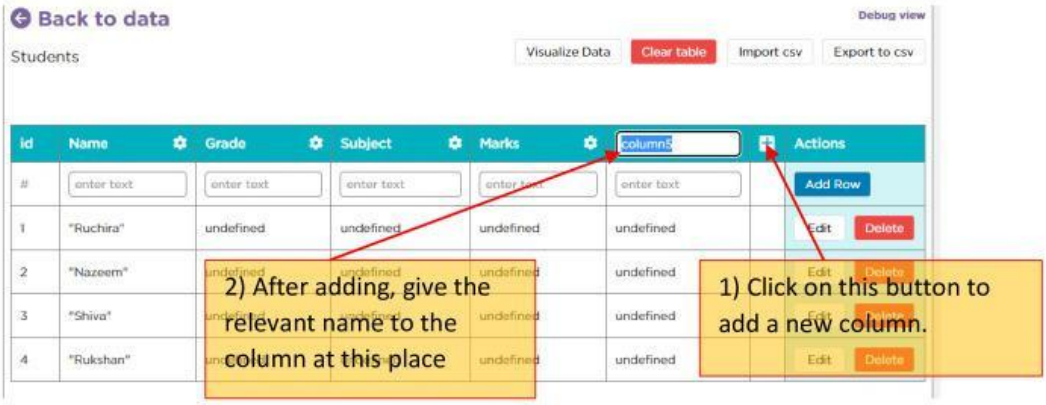


After adding all the records, back to data can be done. If you want to add more records, you can click on the table name as above and view the table again.

❖ In this way, create another new table as "Students" as follows.



4 columns have been used in this design. Name, Grade, Subject, Marks. Follow the steps below to add a new column.



❖ After creating both the data tables needed to create the app, let's start creating the app. The required screens are provided for you. Let's start coding.

❖ First, create two variables as follows to store the name entered in the search box and to check whether the entered name is included in the data table.

```
var stdName = "";
var register = false;
```

❖ After that, when the Add Marks button is clicked, create to switch to the screen where the marks are added.

❖ Let's use the subjects in the data table created above to provide options for the subject drop down.

```
readRecords("Subjects", {}, function(records) {
  var options = [];
  for (var i = 0; i < records.length; i++) {
    options.push(records[i].Name);
  }
  setProperty("drpSubjects", "options", options);
});
```

- The readRecord block accesses the data table we created as Subjects.
- Here, create an array named option. It is used to store data in the data table.
- The for loop goes through all the records and stores the value in the Name column of each record in the "options" array created above.

```
options.push(records[i].Name);
```

- For this, the push function in arrays is used. It can insert a new value into an array.

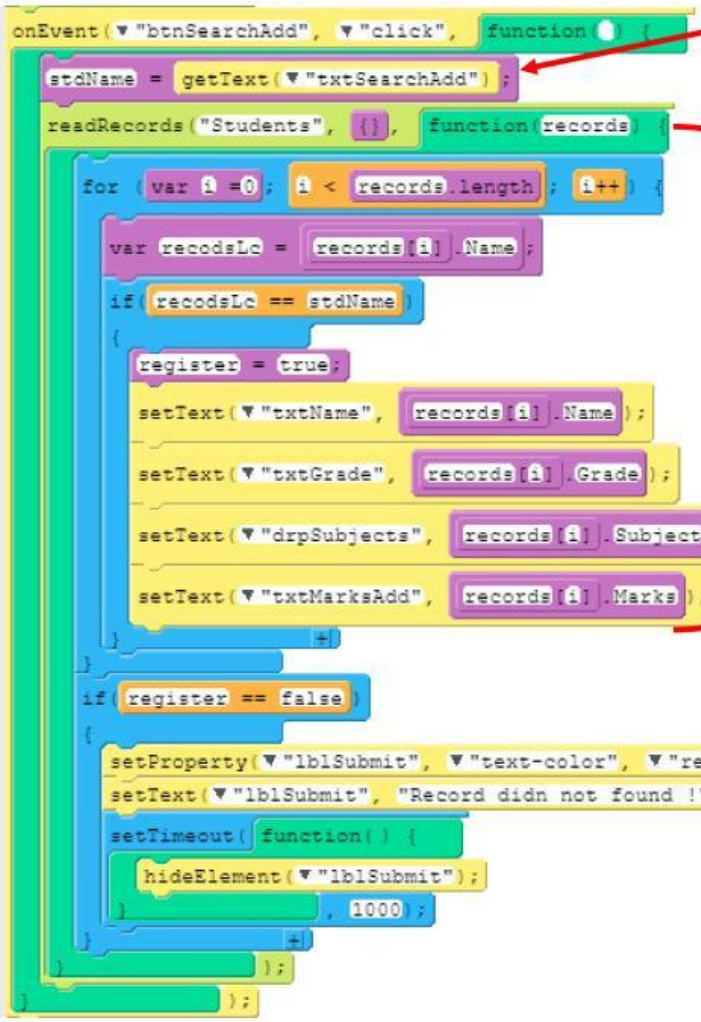
```
options.push
```

- Then this block is used to provide the included options array for the

```
setProperty("drpSubjects", "options", options);
```

dropdown.

- To update a child's details, first type the child's name in the search box and click the search button to see if that name is included in the students table. Code as below for that.



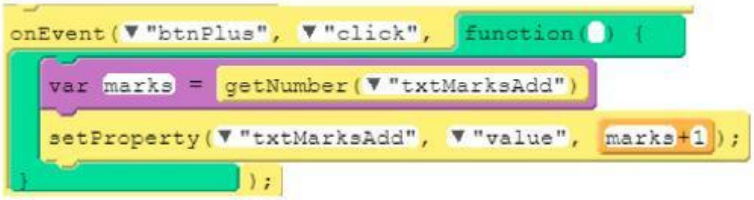
Putting the value entered into the search box into the variable created above

The records in the Student table are read and the value of the Name column of each record is put into the variable named recordLc in each loop of the for loop.

Then the if block checks whether the value of that variable is equal to the value entered in the search box. If there is a similar record, the values of each column are set to fields named txtName, txtGrade, drpSubjects, txtMarksAdd.

If the condition checked above is false, that is, if the input name is not included in the data table, use these codes to display a not found message and it will disappear after a second. A label as "lblSubmit" has been designed for this. Use it

- ❖ After coding in this way, if there is the search name in the data table, the values of that record will be displayed in the text field.
- ❖ Then code as follows for the operation of the plus button and the minus button when adding points to update a record that has been searched in this way.



Create a new variable as Marks and store the value of the text box to add the marks named "txtMarksAdd" using getNumber. getNumber stores the value of that textbox as a number in the variable

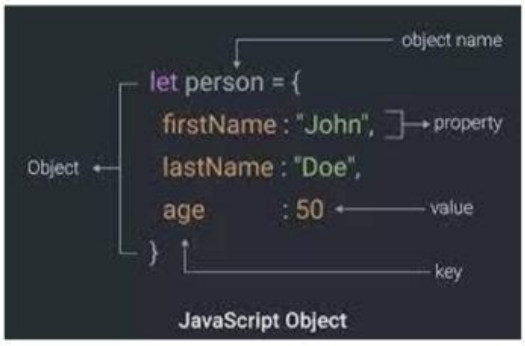
When the Plus button is clicked, the value of the textbox is coded as follows to increase by one.

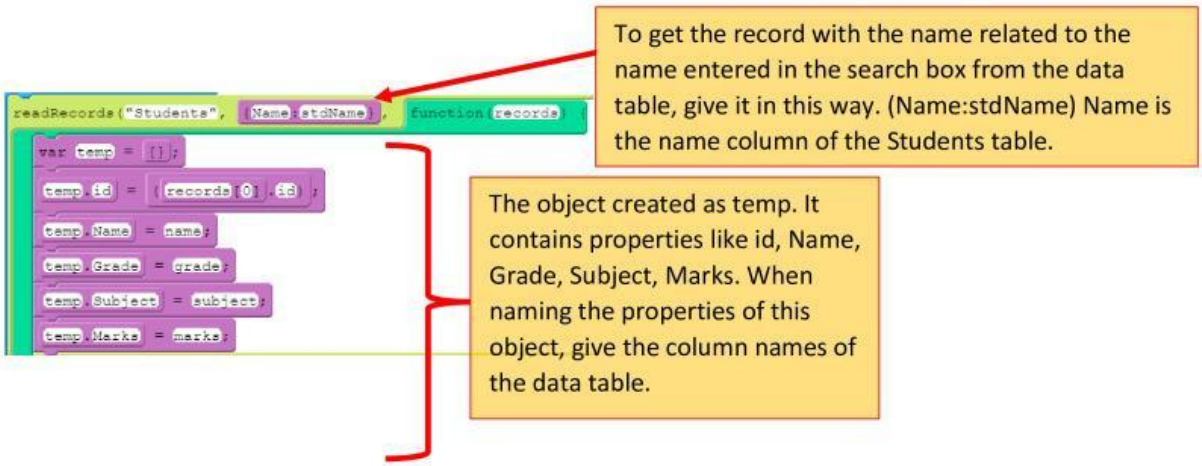
```
setProperty("txtMarksAdd", "value", marks+1);
```

- In this way, when the minus button is clicked, create the value of the text box to decrease by one.
- Then let's see how to submit a search and update record and save it in the data table.
- Use the onEventClick block as follows to update when the Submit button is clicked.

```
onEvent("btnSubmit", "click", function() {
  var name = getText("txtName");
  var grade = getText("txtGrade");
  var subject = getText("drpSubjects");
  var marks = getText("txtMarksAdd");
```

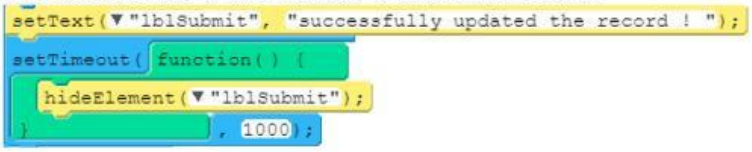
- Create 4 variables as name, grade, subject and marks and store the values of each textbox in variables by getText.
- Then if the searched name is in the data table, the above register variable will be set to true. Here, if it is true, the record related to that name is obtained by the readRecord block as follows and its values are stored in an object called temp.
- An object is an entity that contains properties with data types of various types. Below is an object created as person. An object contains a key and a value for one property. An object consists of several properties.





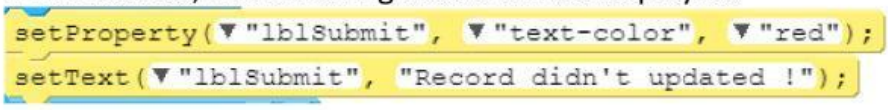
- Here, the values entered in the text box are stored in the properties of the object. This object is passed below to update the data table.

- If the update was done correctly, the success parameter will be passed into the function. Check it in the if block and if it is successful

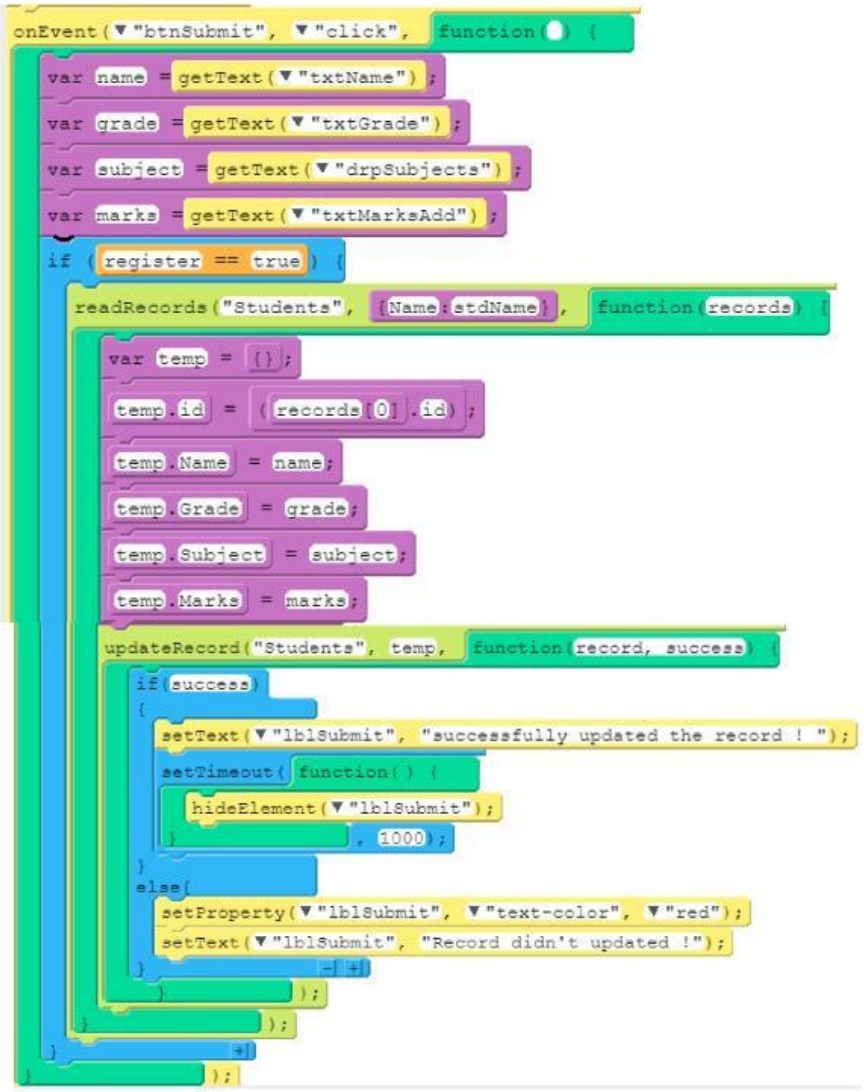


- This block is displayed and disappears in a second.

  If not success, the following blocks will be displayed.



- Below is the complete code for what should happen when the submit button is clicked.



- Create the app in this way and update the record through the app and go back to the data table to see if it has been updated. If the app has been created correctly, the data table should be updated when it is updated.

- Create to go back to the home screen when the back button is finally clicked.