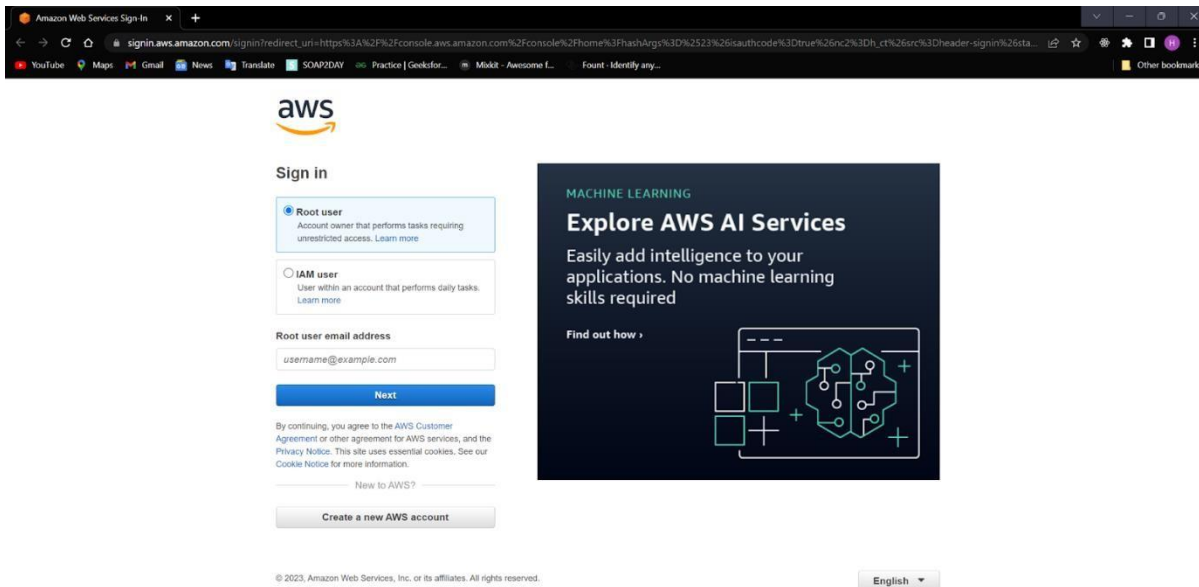


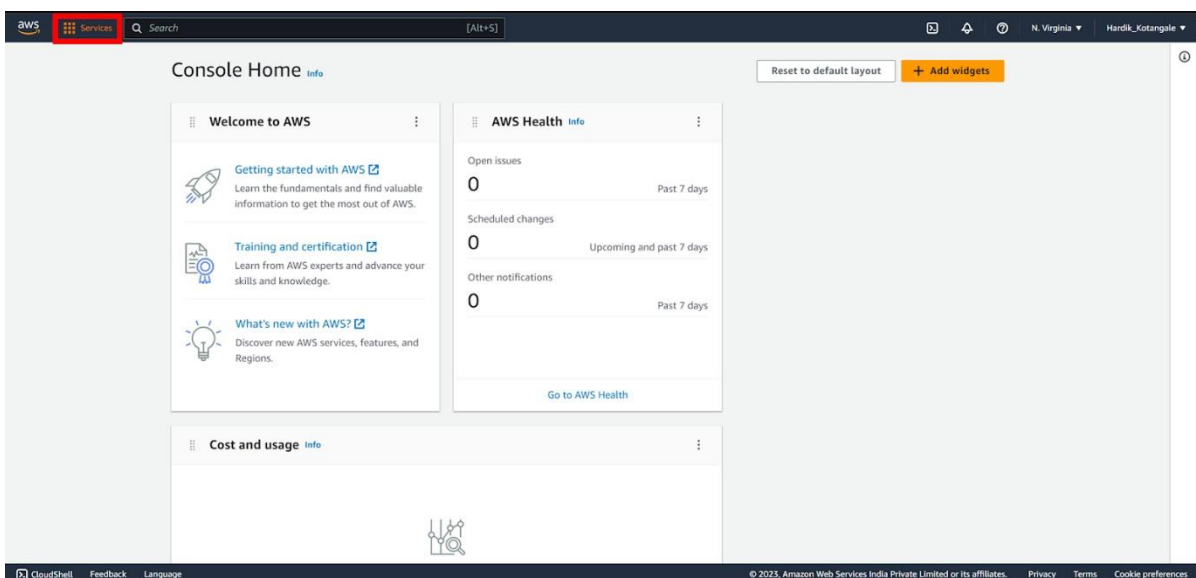
Implementation

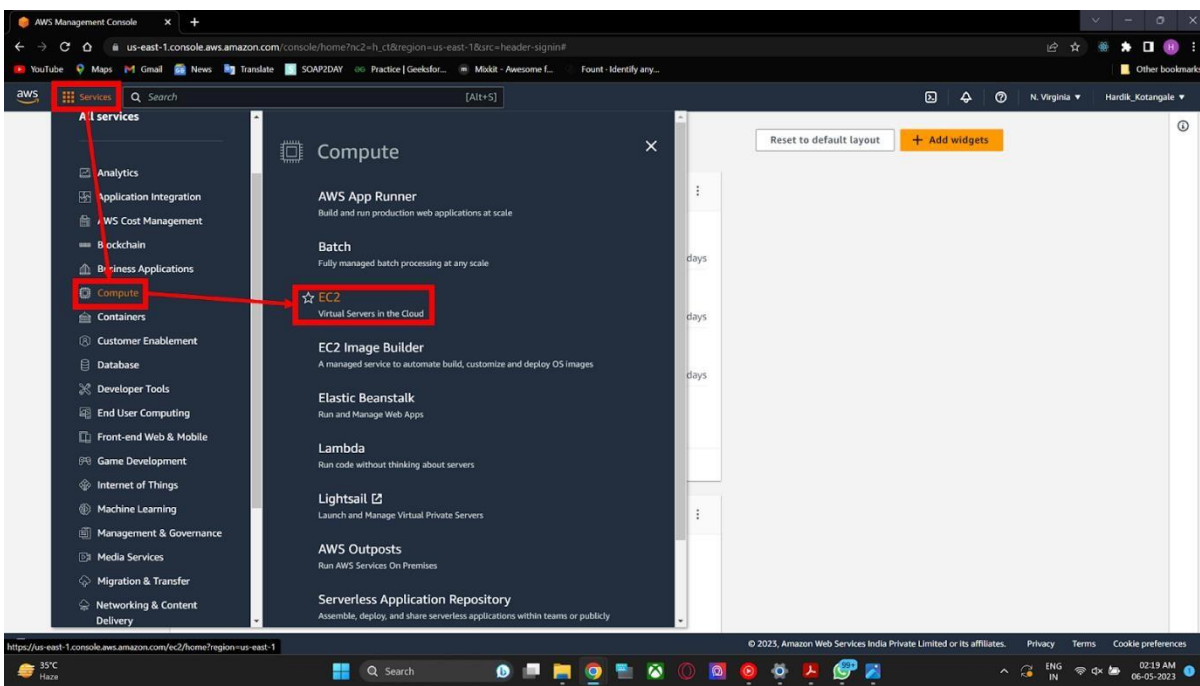
Step 1 :- Sign in to AWS Account

https://signin.aws.amazon.com/signin?redirect_uri=https%3A%2F%2Fconsole.aws.amazon.com%2Fconsole%2Fhome%3FhashArgs%3D%2523%26isauthcode%3Dtrue%26nc2%3Dh_ct%26src%3Dheader-signin%26state%3DhashArgsFromTB_eu-north-1_522611e77e2d291b&client_id=arn%3Aaws%3Asignin%3A%3A%3Aconsole%2Fcanvas&forceMobileApp=0&code_challenge=9IbjLkDwSPZ5p9SWe-PveqynNv5PaUI_IIBU2gzi6dU&code_challenge_method=SHA-256

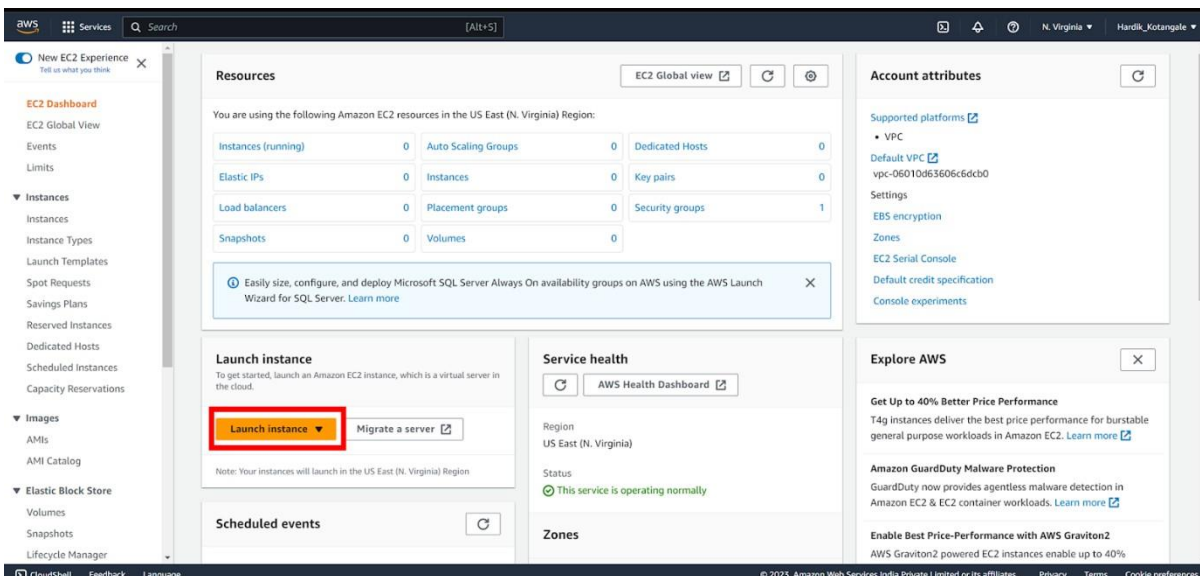


Step 2:-To run an instance, you can navigate to the AWS services menu, select the "Computing" option, and then choose "EC2". This will take you to the EC2 dashboard where you can launch and manage your instances.

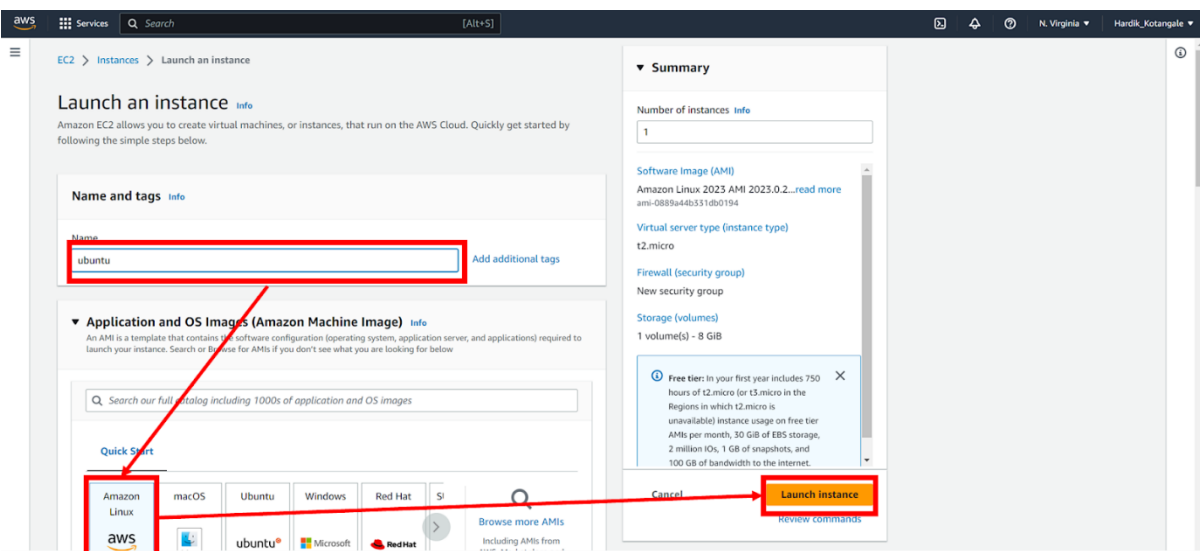




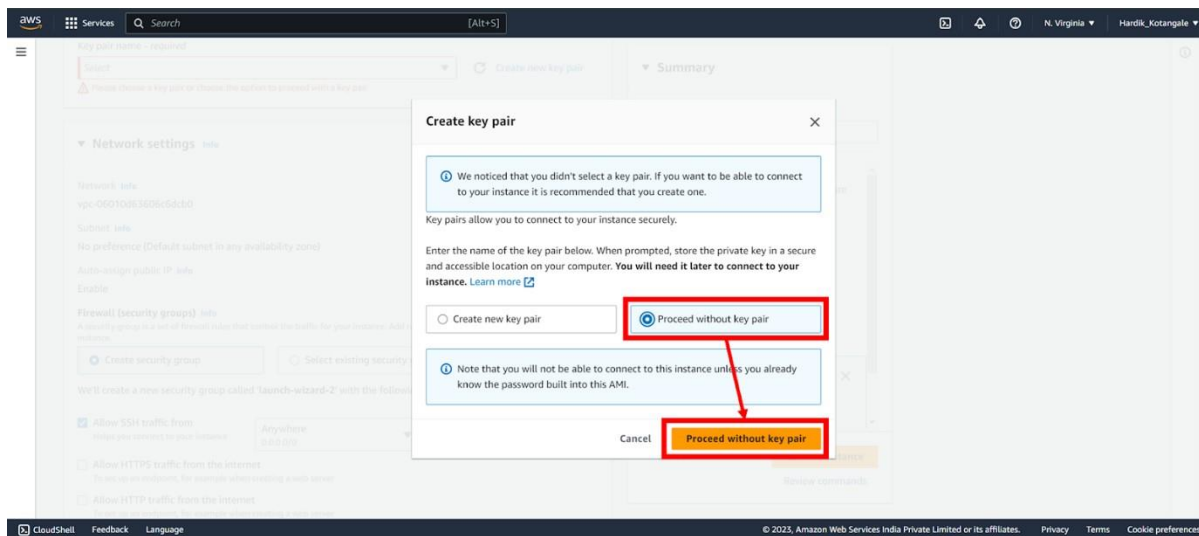
Step 3 :- To set up the instance, we have to click on "Launch Instance".



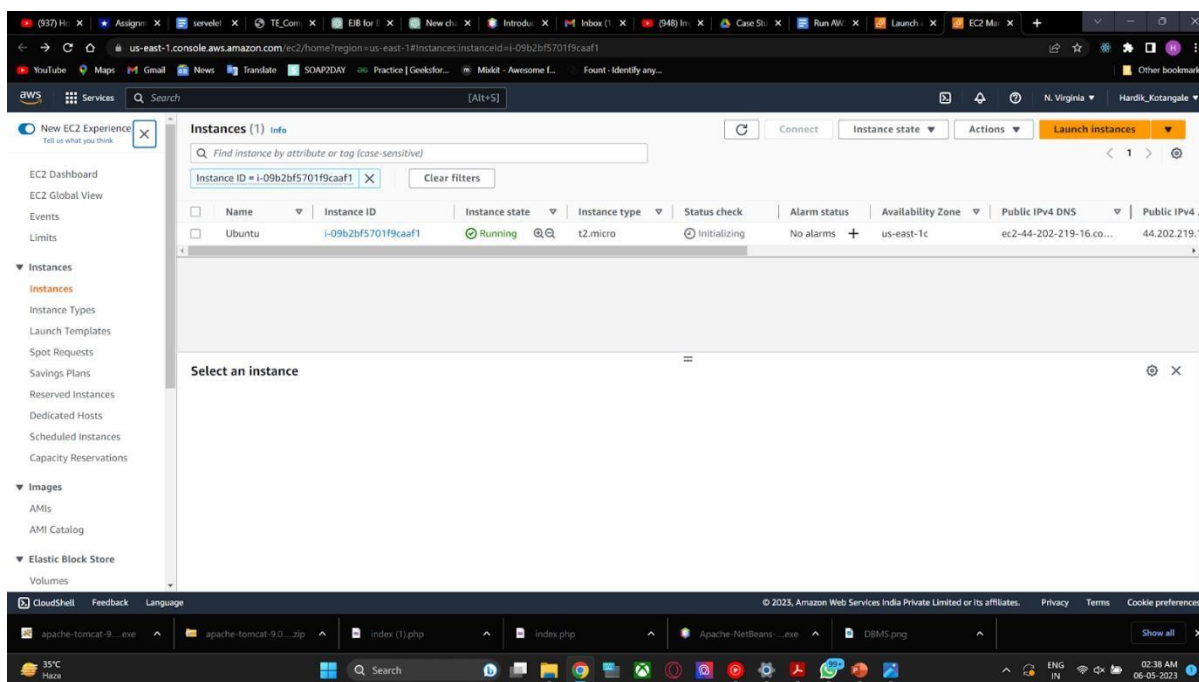
Step 4:- You need to provide a name for the instance. From there, you can select the operating system or application that you want to run on the instance. Then click on launch instance

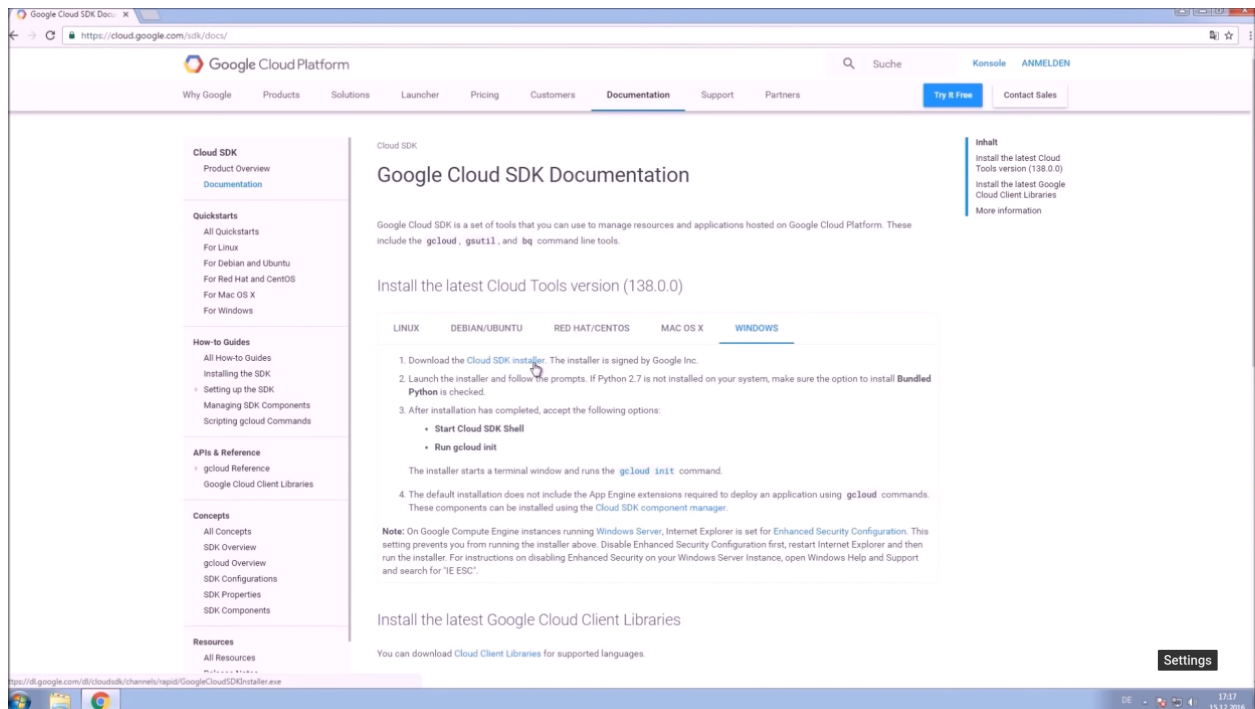
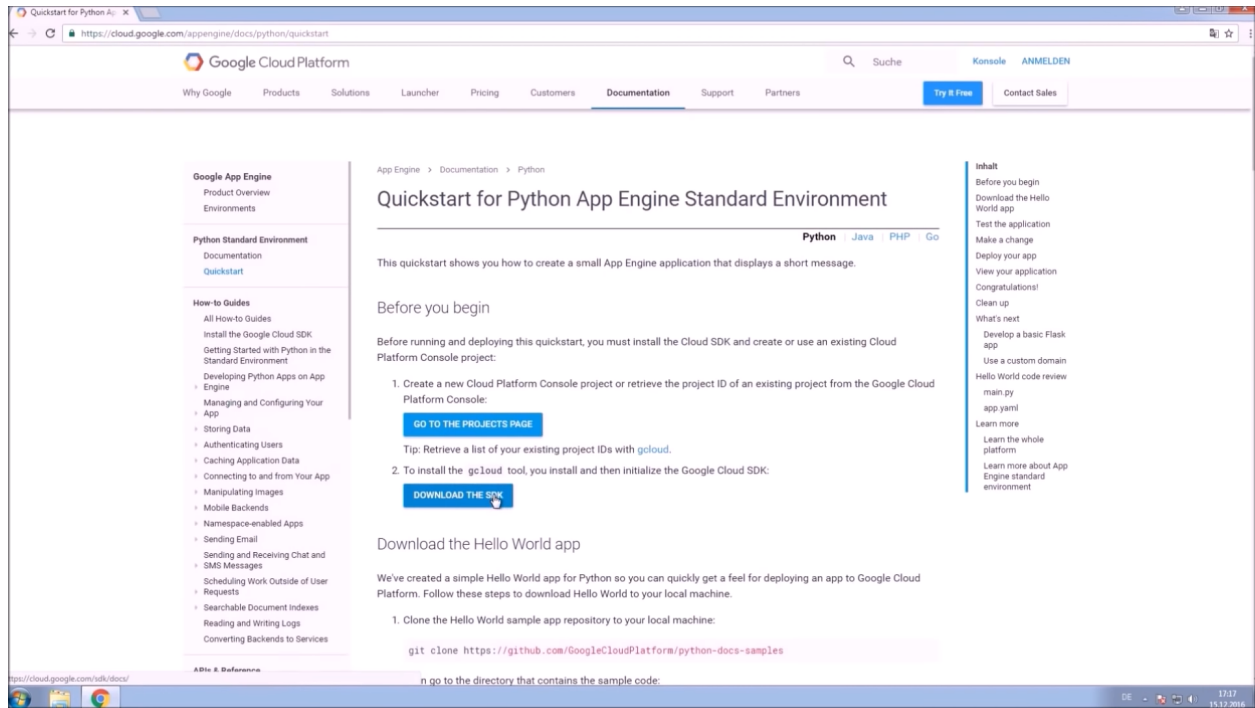


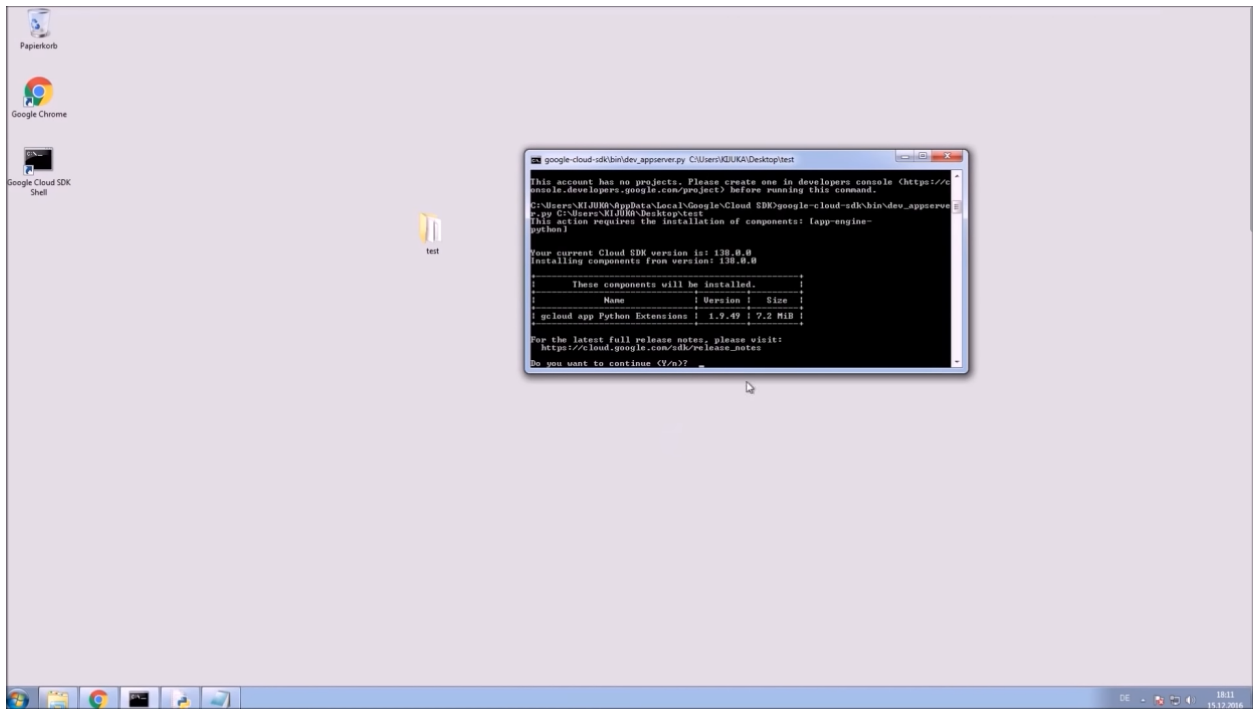
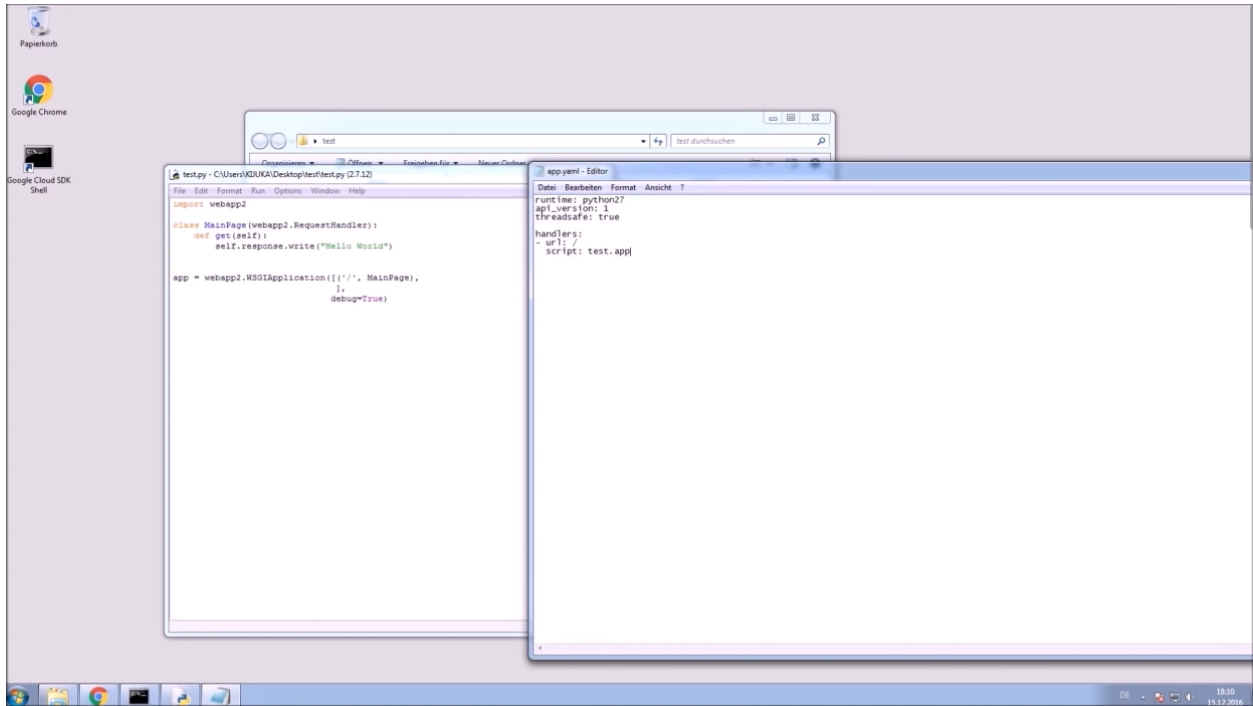
Step 5:-To proceed without a key pair, select the "Proceed without a key pair" option and click on it.



Step 6:- Monitor your instance







Google Cloud SDK Docs | You are now authenticated | Instances | localhost:8080/instances

Google App Engine

dev~None

Instances

	Latency (ms)	QPS	Total Requests	Runtime
default	0.0	0.00	0	python27

data42ed52c7ef56629fa5efcc2b4c8ded9

0.0.0

Möchten Sie diese Seite übersetzen? [Optionen](#)

[Übersetzen](#) [Nein](#)

Instances

Datastore Viewer

Datastore Indexes

Datastore Stats

Interactive Console

Memcache Viewer

Blobstore Viewer

Task Queues

Cron Jobs

XMPP

Inbound Mail

Full Text Search

18.13 15.12.2016

Google Cloud SDK Docs | You are now authenticated | Instances | localhost:8080

localhost:8080

Hello World

18.13 15.12.2016

CODE:-

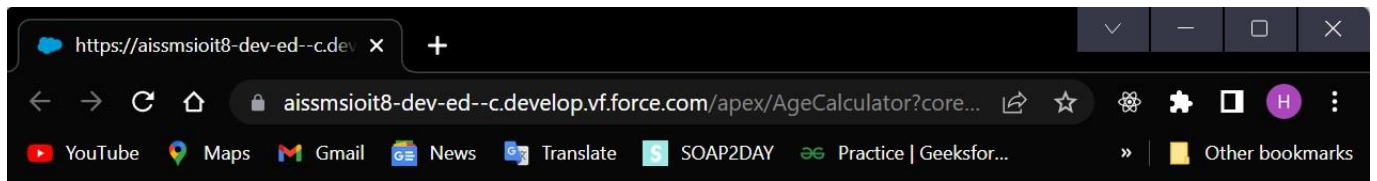
AgeCalculator.apxc

```
public class AgeCalculator {
    public Date birthdate {get; set;}
    public integer result {get; set;}
    // Define a method to calculate the age in years based on a birthdate
    public void calculateAge() {
        Date today = Date.today();
        Integer years = today.year() - birthdate.year();
        if (birthdate.month() > today.month() ||
            (birthdate.month() == today.month() && birthdate.day() > today.day())) {
            years--;
        }
        result = years;
    }
}
```

AgeCalculator.vfp

```
<apex:page controller="AgeCalculator">
<apex:sectionHeader subtitle="Age Calculator"/>
<apex:form >
<apex:pageBlock >
<apex:pageBlockButtons location="bottom">
<apex:commandButton value="Calculate" action="{!calculateAge}" reRender="res"/>
</apex:pageBlockButtons>
<apex:pageBlockSection title="Calculator">
<apex:inputText label="Date Of Birth" html-placeholder="Date Of Birth"
value="{!birthdate}"/>
<apex:outputText label="Age" value="{!result}" id="res"/>
</apex:pageBlockSection>
</apex:pageBlock>
</apex:form>
</apex:page>
```


OUTPUT:-



Age Calculator

▼ Calculator	
Date Of Birth	<input type="text" value="23/10/2013"/>
Age	9
<input type="button" value="Calculate"/>	

CODE:-

FormValidation.apxc

```
public class FormValidation {

    public String firstName { get; set; }
    public String lastName { get; set; }
    public String email { get; set; }
    public String phone { get; set; }

    public void submit() {

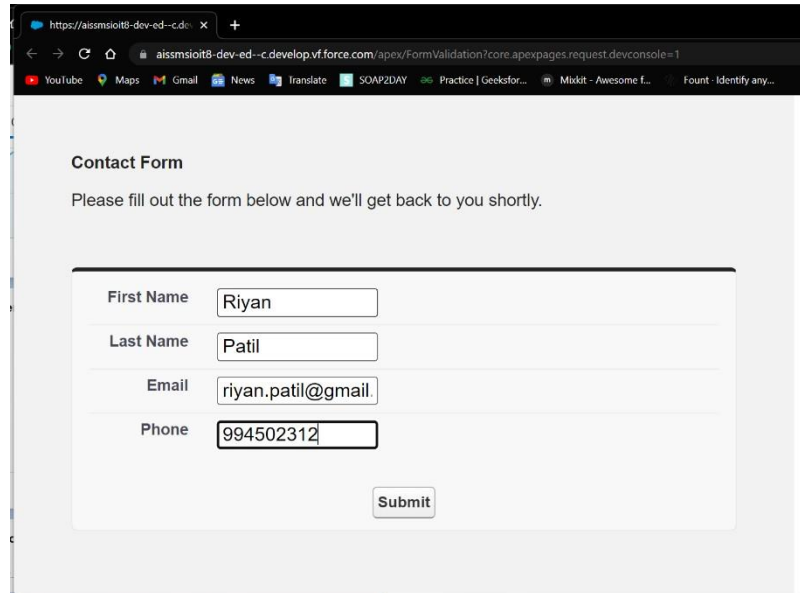
        // Perform form validation
        if (String.isBlank(firstName)) {
            ApexPages.addMessage(new ApexPages.Message(ApexPages.Severity.ERROR,
'First name is required.));
        }
        if (String.isBlank(lastName)) {
            ApexPages.addMessage(new ApexPages.Message(ApexPages.Severity.ERROR,
'Last name is required.));
        }
        if (String.isBlank(email) || !Pattern.matches('[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-
zA-Z]{2,}', email)) {
            ApexPages.addMessage(new ApexPages.Message(ApexPages.Severity.ERROR,
'Please enter a valid email address.));
        }
        if (String.isBlank(phone) || !Pattern.matches("\\d{10}'", phone)) {
            ApexPages.addMessage(new ApexPages.Message(ApexPages.Severity.ERROR,
'Please enter a valid phone number.));
        }

        // If there are no validation errors, save the form data
        if (ApexPages.getMessages().isEmpty()) {
            // Code to save the form data
            // ...
            ApexPages.addMessage(new ApexPages.Message(ApexPages.Severity.INFO, 'Form
submitted successfully!));
        }
    }
}
```

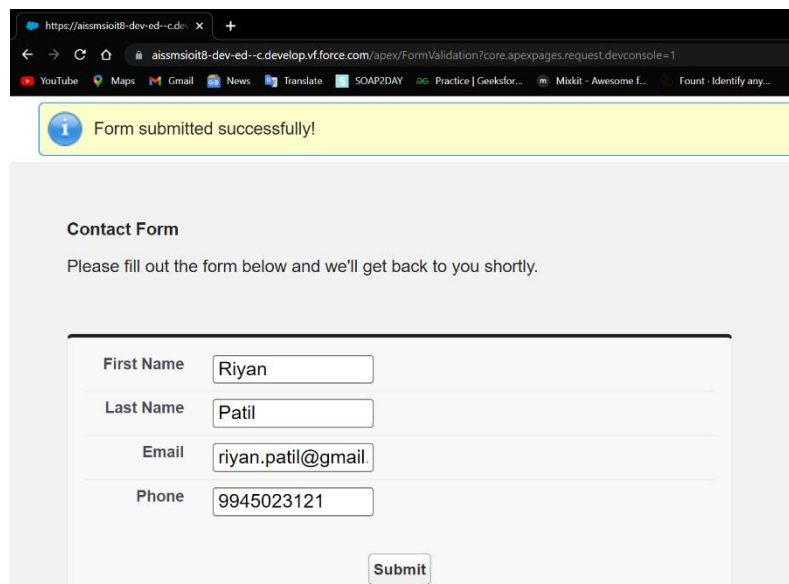
FormValidation.vfp

```
<apex:page controller="FormValidation" showHeader="false" sidebar="false" >
  <apex:form style="align-items:center;">
    <apex:pageMessages />
    <div style="padding: 40px; width: 50%; background-color: #f2f2f2;">
      <h2 style="margin-bottom: 10px;">Contact Form</h2>
      <p style="margin-bottom: 40px;">Please fill out the form below and we'll get back to
you shortly.</p>
      <apex:pageBlock >
        <apex:pageBlockSection columns="1" >
          <apex:inputText value="{ !firstName}" label="First Name" required="true"
style="width: 30%;" />
          <apex:inputText value="{ !lastName}" label="Last Name" required="true"
style="width: 30%;" />
          <apex:inputText value="{ !email}" label="Email" required="true" style="width:
30%;" />
          <apex:inputText value="{ !phone}" label="Phone" required="true" style="width:
30%;" />
        </apex:pageBlockSection> <div style="text-align: center; margin-top: 20px;">
          <apex:commandButton value="Submit" action="{ !submit}" styleClass="my-
button" />
        </div>
      </apex:pageBlock>
    </div>
  </apex:form>
</apex:page>
```

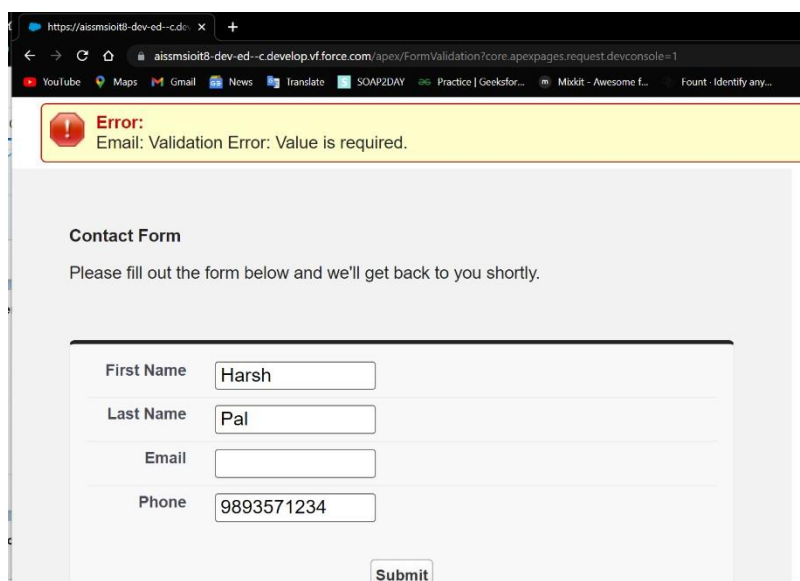
OUTPUT:-



The screenshot shows a web browser window with the URL `https://aissmsioit8-dev-ed--c-develop.vf.force.com/apex/FormValidation?core.apexpages.request.devconsole=1`. The page displays a "Contact Form" with the instruction "Please fill out the form below and we'll get back to you shortly." The form fields are filled with the following data: First Name: Riyan, Last Name: Patil, Email: riyan.patil@gmail, and Phone: 994502312. A "Submit" button is located at the bottom right of the form.



The screenshot shows the same web browser window as the previous one, but with a yellow confirmation message at the top: "Form submitted successfully!". The "Contact Form" is still visible below the message, with the same data as before: First Name: Riyan, Last Name: Patil, Email: riyan.patil@gmail, and Phone: 994502312. The "Submit" button is still present at the bottom right.



The screenshot shows the same web browser window, but with a red error message at the top: "Error: Email: Validation Error: Value is required." The "Contact Form" is still visible below the message, but the data has been changed: First Name: Harsh, Last Name: Pal, Email: (empty), and Phone: 9893571234. The "Submit" button is still present at the bottom right.

CODE:-

Calculator.apxc

```
public class Calculator {  
  
    public integer firstNumber {get; set;}  
    public integer secondNumber {get; set;}  
    public integer result {get; set;}  
    public void addition() {  
        result = firstNumber + secondNumber ;  
    }  
    public void subtraction() {  
        result = firstNumber - secondNumber ;  
    }  
    public void Multiplication() {  
        result = firstNumber * secondNumber ;  
    }  
    public void Division()  
        result = result = firstNumber / secondNumber  
    }  
}
```

Calculator.vfp

```
<apex:page controller="Calculator">  
<apex:sectionHeader subtitle="Basic Calculator"/>  
<apex:form >  
<apex:pageBlock >  
<apex:pageBlockButtons location="bottom">  
<apex:commandButton value="Addition" action="{!addition}" reRender="res"/>  
<apex:commandButton value="Subtraction" action="{!subtraction}" reRender="res"/>  
<apex:commandButton value="Multiplication" action="{!Multiplication}" reRender="res"/>  
<apex:commandButton value="Division" action="{!Division}" reRender="res"/>  
</apex:pageBlockButtons>  
<apex:pageBlockSection title="Calculator">  
<apex:inputText label="Enter First Number" html-placeholder="First Number"  
value="{!firstNumber}"/>  
<apex:inputText label="Enter Second Number" html-placeholder="Second Number"  
value="{!secondNumber}"/>  
<apex:outputText label="Result" value="{!result}" id="res"/>  
</apex:pageBlockSection>  
</apex:pageBlock>  
</apex:form>  
</apex:page>
```

OUTPUT:-

https://aissmsioit8-dev-ed--c.de... X

aissmsioit8-dev-ed--c.develop.vf.force.com/apex/Calculator?core.apexpages.request.devconsole=1

Basic Calculator

▼ Calculator

Enter First Number Enter Second Number

Result 5

Addition Subtraction Multiplication Division

https://aissmsioit8-dev-ed--c.de... X

aissmsioit8-dev-ed--c.develop.vf.force.com/apex/Calculator?core.apexpages.request.devconsole=1

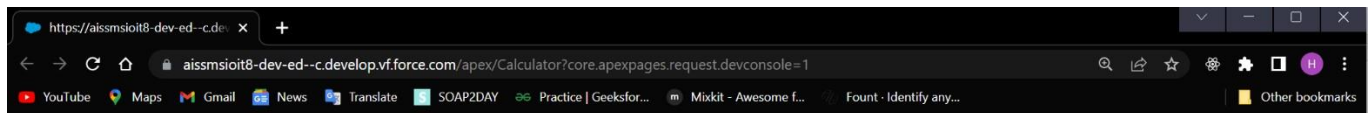
Basic Calculator

▼ Calculator

Enter First Number Enter Second Number

Result 2

Addition Subtraction Multiplication Division



Basic Calculator

▼ Calculator

Enter First Number

5

Enter Second Number

3

Result

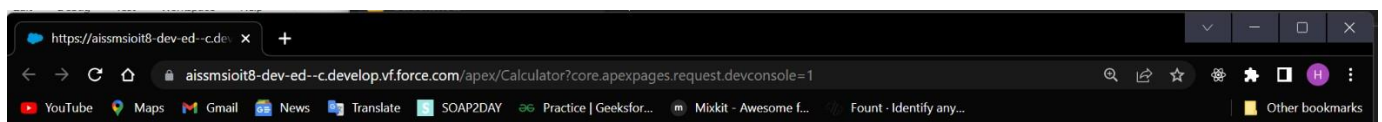
15

Addition

Subtraction

Multiplication

Division



Basic Calculator

▼ Calculator

Enter First Number

15

Enter Second Number

5

Result

3

Addition

Subtraction

Multiplication

Division

CODE:-

Game.apxc

```
public class Game {
    public integer counter {get;set;}
    public integer totalCleared {get;set;}
    public Integer secCount {get;set;}

    public Set<Integer> alreadyUsed {get;set;}
    public List<Integer> shuffledNumbersByCh {get;set;}
    Public List<List<Integer>> numberList {get;set;}
    public Game () {
        totalCleared = 1;
        shuffleNumbers();
    }
    public void shuffleNumbers() {
        //set the range of numbers from to N
        Integer lsitSize = 25;
        alreadyUsed= new Set<Integer>();
        shuffledNumbersByCh = new List<Integer>();
        while(shuffledNumbersByCh .size() < lsitSize){
            shuffledNumbersByCh .add(getRandom(lsitSize));
        }
        for(integer i=0; i< shuffledNumbersByCh.size(); i++) {
            shuffledNumbersByCh[i] = shuffledNumbersByCh[i]+1;
        }
        system.debug('Final shuffled Numbers'+shuffledNumbersByCh );
        numberList = new List<List<Integer>>();
        List<Integer> rowList = new List<Integer>();
        Integer j =0;
        for(integer i =0; i<=24; i++) {
            J= i+1;
            rowList.add(shuffledNumbersByCh[i]);
            if(Math.Mod(j,5) ==0) {
                numberList.add(rowList);
                rowList = new List<Integer>();
            }
        }
    }
    public Integer getRandom(Integer shuffledMaxNumber){
        Integer randomNum;
        do {
            randomNum = (math.random() * shuffledMaxNumber).intValue();
        }
```



```

    } while (alreadyUsed.contains(randomNum));
    alreadyUsed.add(randomNum);
    return randomNum;
}
public void updateCount() {
    counter = counter + 1;
    totalCleared = totalCleared + 1;
    system.debug('counter counter '+counter );
}
}

```

Game.vfp

```

<apex:page controller="Game" sidebar="false" showHeader="false" id="page1">
<apex:form id="form1">
<style>
    body{
        margin-top:50px;
    }
h1 {
    font-size:20px;
}
.grid-cell {
    width: 50px;
    height: 50px;
    border-radius: 3px;
    text-align:center;
    text-color:red;
    background: #FF7849;
    color:white;
    font-weight:bold;
    font-size:35px;
}
</style>
<script>
var startTimer;
function fun(rowIndex,columnIndex) {
    var myTable = document.getElementById('NumberTable');
    var previousValue = document.getElementById('page1:form1:displayCounter').value;
    var currentValue = myTable.rows[rowIndex].cells[columnIndex].innerHTML ;
    if(currentValue == 1) {
        startTimer();
    }
    if(currentValue == +previousValue +1) {
        myTable.rows[rowIndex].cells[columnIndex].innerHTML = "";
        var totalClear = document.getElementById('page1:form1:displayCounter1').value;

```

```

    if(totalClear == 25) {
        var sec = document.getElementById("displayTime").innerHTML;
        document.getElementById("displayTime").innerHTML = "Great ..You cleared all
the numbers in "+sec+". Refresh the Browser Url to start new Game";
        stopTime();
    }
    updateCounter();
}

}

function stopTime() {
    clearInterval(startTimer);
}

function startTime() {
    var sec = 0;
    startTimer = setInterval(
    function () {
        sec = sec + 1;
        document.getElementById("displayTime").innerHTML = sec + " Seconds!!!";
    }, 1000);
}
</script>
<body >
    <center>
<table border="1" cellpadding="0" cellspacing = "0" id="table1" style="padding-left:5px;">
<tr><td>
<table >
<tr><td >
<h1>

    <b style="color:red; ">Game Instructions : </b>Touch all the numbers from 1 to 25 in
Sequence to empty this Grid
The Counter will get started as soon as you start the game.Try to finish as early as possible.
</h1>
    </td>
</tr>
</table>
</td></tr>
<tr><td><h1>

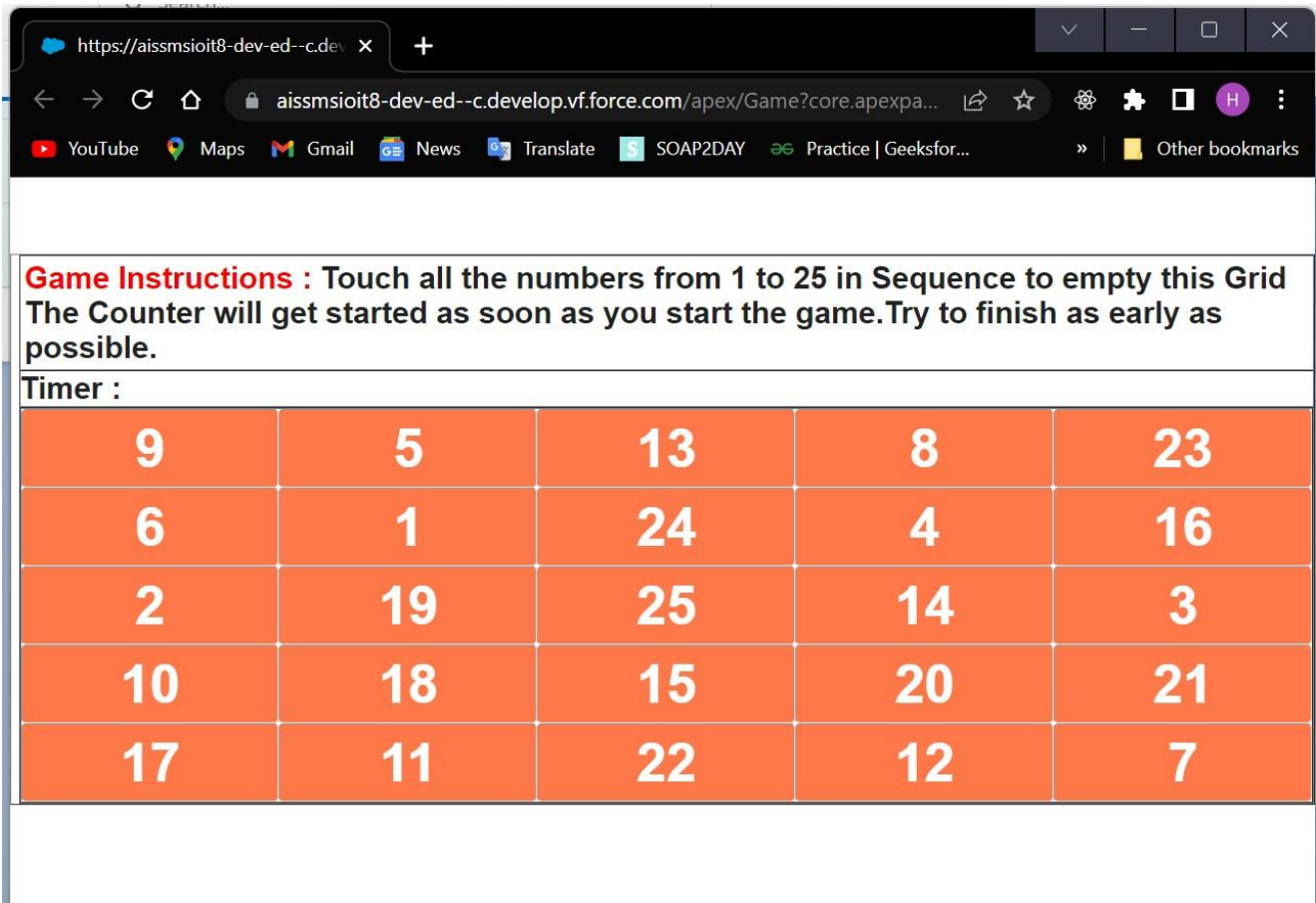
    Timer :<b id="displayTime" style="Color:red;"></b></h1></td></tr>
<tr><td>
<table border="1" cellpadding="0" cellspacing = "0" id="NumberTable" width="100%" >
<apex:variable var="rowIndex" value="{!0}"/>
<apex:repeat value="{!numberList}" var="rowNums">
<tr style="border:1px"><apex:variable var="columnIndex" value="{!0}"/>
    <apex:repeat value="{!rowNums}" var="num">

```

```
<td class="grid-cell" onClick="fun('{!rowIndex}','{!columnIndex}')"> {!num}</td>
<apex:variable var="columnIndex" value="{!columnIndex+1}" />
</apex:repeat><apex:variable var="rowIndex" value="{!rowIndex+1}" />
</tr>
</apex:repeat>
</table>
</td></tr></table>
</center>
```

```
<apex:inputtext value="{!counter}" id="displayCounter" style="display:none;" />
<apex:inputtext value="{!totalCleared}" id="displayCounter1" style="display:none;" />
</body>
<button onclick="window.location.reload()" id="startGame" style="display:none;">Start
New Game</button>
<apex:actionFunction action="{!updateCount}" name="updateCounter"
render="displayCounter,displayCounter1">
</apex:actionFunction>
</apex:form>
</apex:page>
```

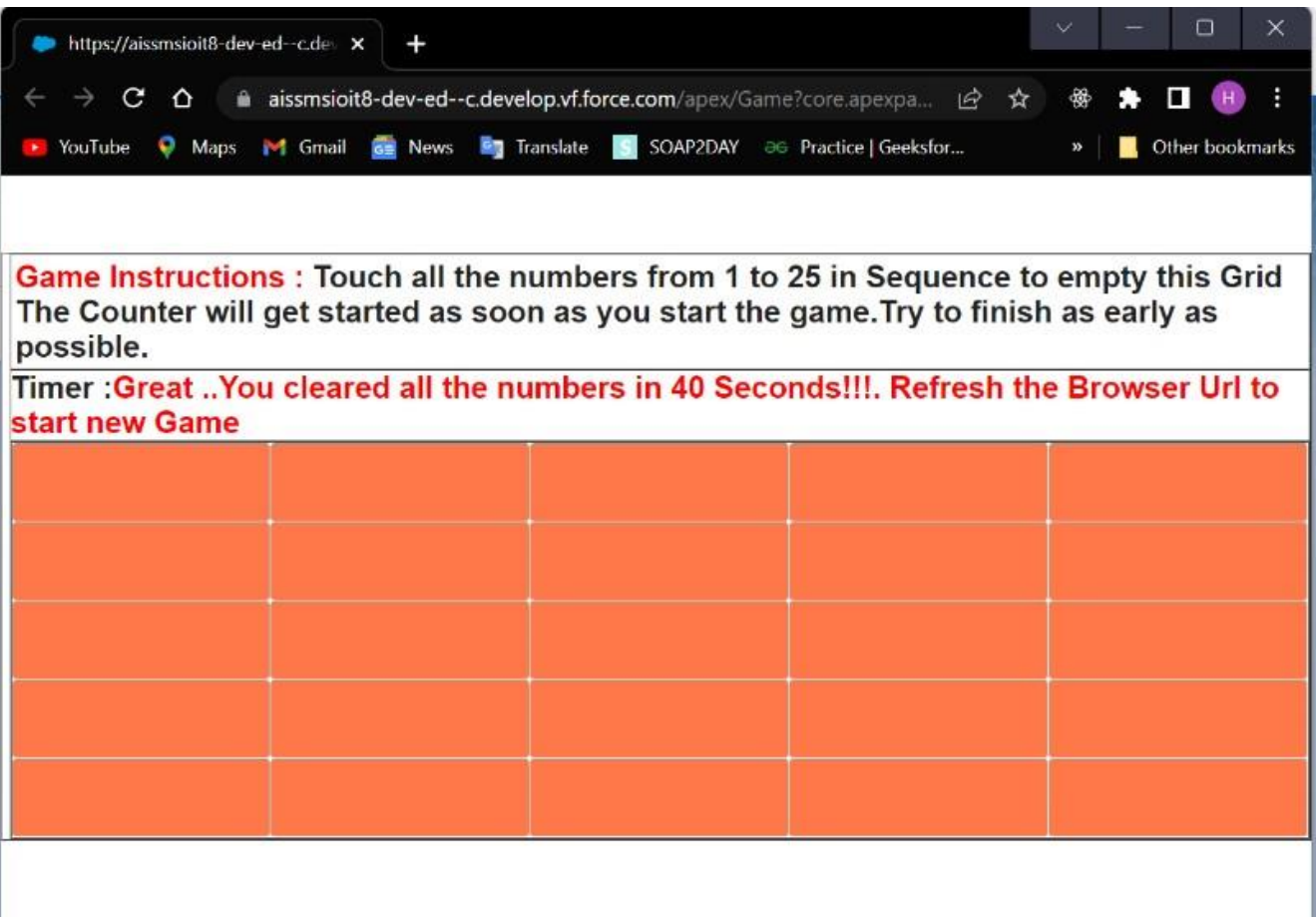
OUTPUT:-



Game Instructions : Touch all the numbers from 1 to 25 in Sequence to empty this Grid
The Counter will get started as soon as you start the game.Try to finish as early as possible.

Timer :

9	5	13	8	23
6	1	24	4	16
2	19	25	14	3
10	18	15	20	21
17	11	22	12	7

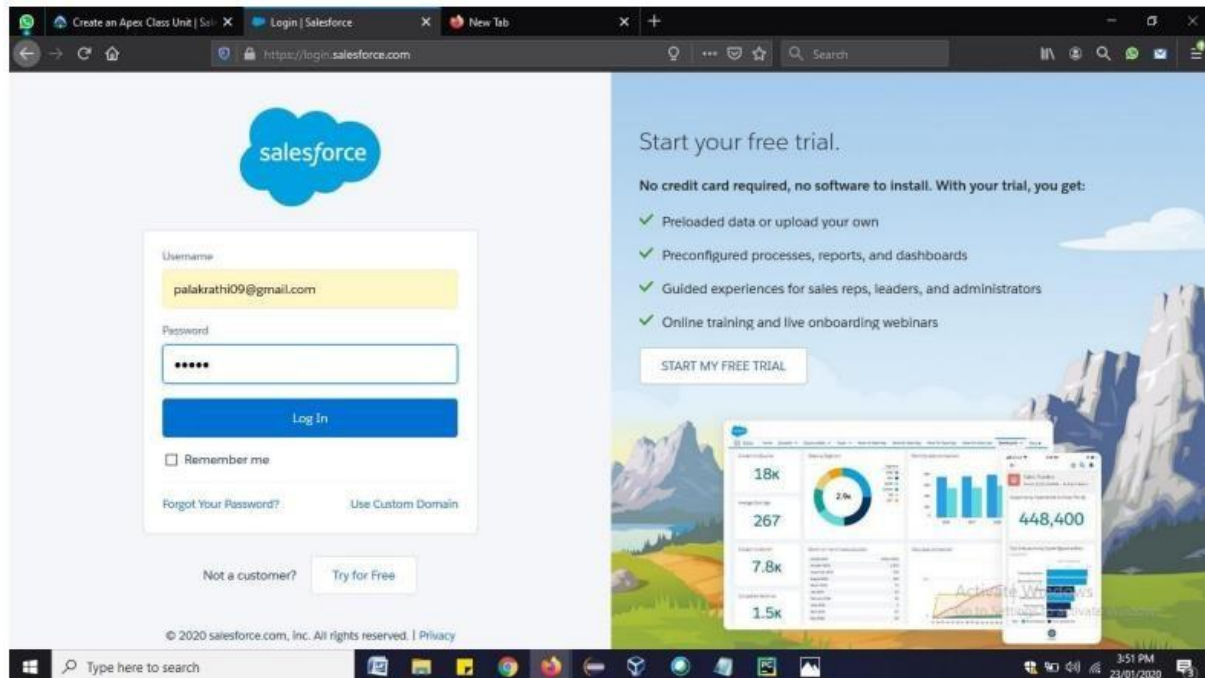


Game Instructions : Touch all the numbers from 1 to 25 in Sequence to empty this Grid
The Counter will get started as soon as you start the game.Try to finish as early as possible.

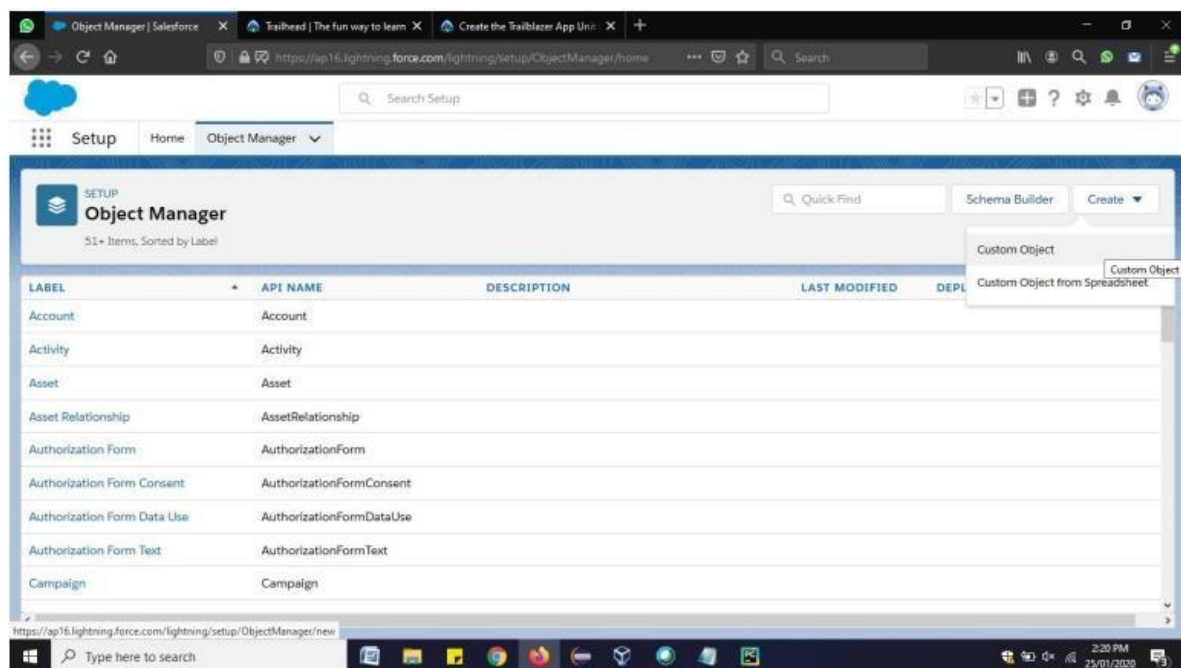
Timer :Great ..You cleared all the numbers in 40 Seconds!!!. Refresh the Browser Url to start new Game

Implementation:

Step 1: Log into Salesforce Developer account.



Step 2: Open Salesforce Lightning platform and click on Object Manager => Create => Custom Object.



Step 3: Fill in the required fields and under Optional Features, select Allow Reports and Allow Activities. Click Save.

The screenshot shows the 'New Custom Object' page in Salesforce Setup. The 'Custom Object Information' section is active, showing fields for Label, Plural Label, Object Name, and Description. The Label is 'Student_Detail' and the Plural Label is 'Student_Details'. The Object Name is 'Student_Detail'. The Description field is empty. A yellow banner at the top states: 'Permissions for this object are disabled for all profiles by default. You can enable object permissions in permission sets or by editing custom profiles. [Tell me more](#) [Don't show this message again](#)'. Buttons for 'Save', 'Save & New', and 'Cancel' are visible.

Setup Home Object Manager

SETUP New Custom Object

Help for this Page

Permissions for this object are disabled for all profiles by default. You can enable object permissions in permission sets or by editing custom profiles. [Tell me more](#) [Don't show this message again](#)

Custom Object Definition Edit Save Save & New Cancel

Custom Object Information ⓘ = Required Information

The singular and plural labels are used in tabs, page layouts, and reports.

Label Example: Account

Plural Label Example: Accounts

Starts with vowel sound ☐

The Object Name is used when referencing the object via the API.

Object Name Example: Account

Description

The screenshot shows the 'New Custom Object' page in Salesforce Setup, with the 'Optional Features' section expanded. The 'Record Name' field is 'Student_Detail Name' and the 'Data Type' is 'Text'. Under 'Optional Features', 'Allow Reports' and 'Allow Activities' are checked, while 'Track Field History' and 'Allow in Chatter Groups' are unchecked. Under 'Object Classification', 'Allow Sharing', 'Allow Bulk API Access', and 'Allow Streaming API Access' are all checked. The 'Deployment Status' section is partially visible at the bottom.

Setup Home Object Manager

SETUP New Custom Object

Enter Record Name Label and Format

The Record Name appears in page layouts, key lists, related lists, lookups, and search results. For example, the Record Name for Account is "Account Name" and for Case it is "Case Number". Note that the Record Name field is always called "Name" when referenced via the API.

Record Name Example: Account Name

Data Type

Optional Features

- ☒ Allow Reports
- ☒ Allow Activities
- ☐ Track Field History
- ☐ Allow in Chatter Groups

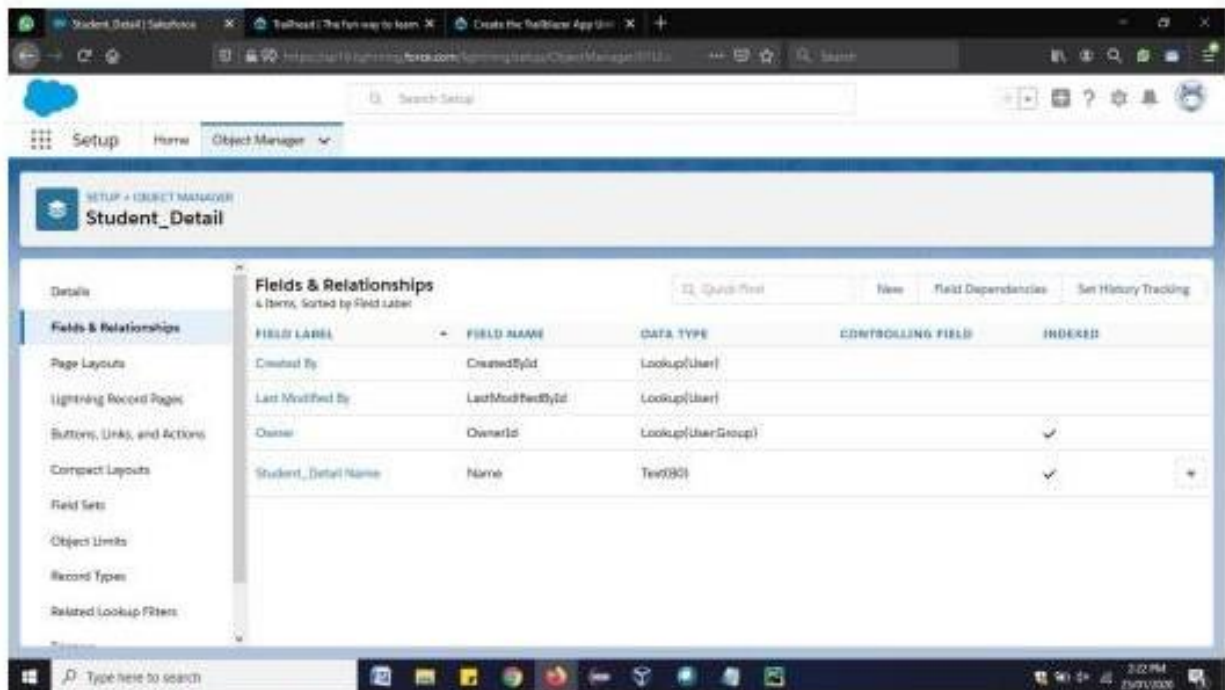
Object Classification

When these settings are enabled, this object is classified as an Enterprise Application object. When these settings are disabled, this object is classified as a Light Application object. [Learn more](#)

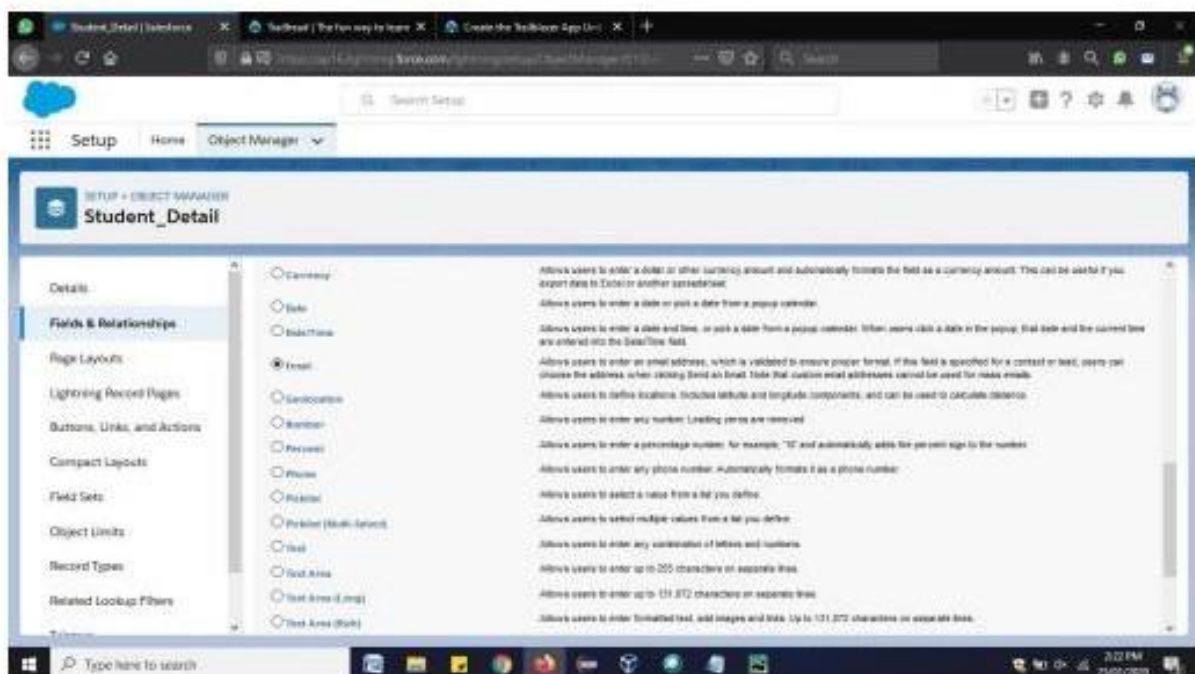
- ☒ Allow Sharing
- ☒ Allow Bulk API Access
- ☒ Allow Streaming API Access

Deployment Status [What is this?](#)

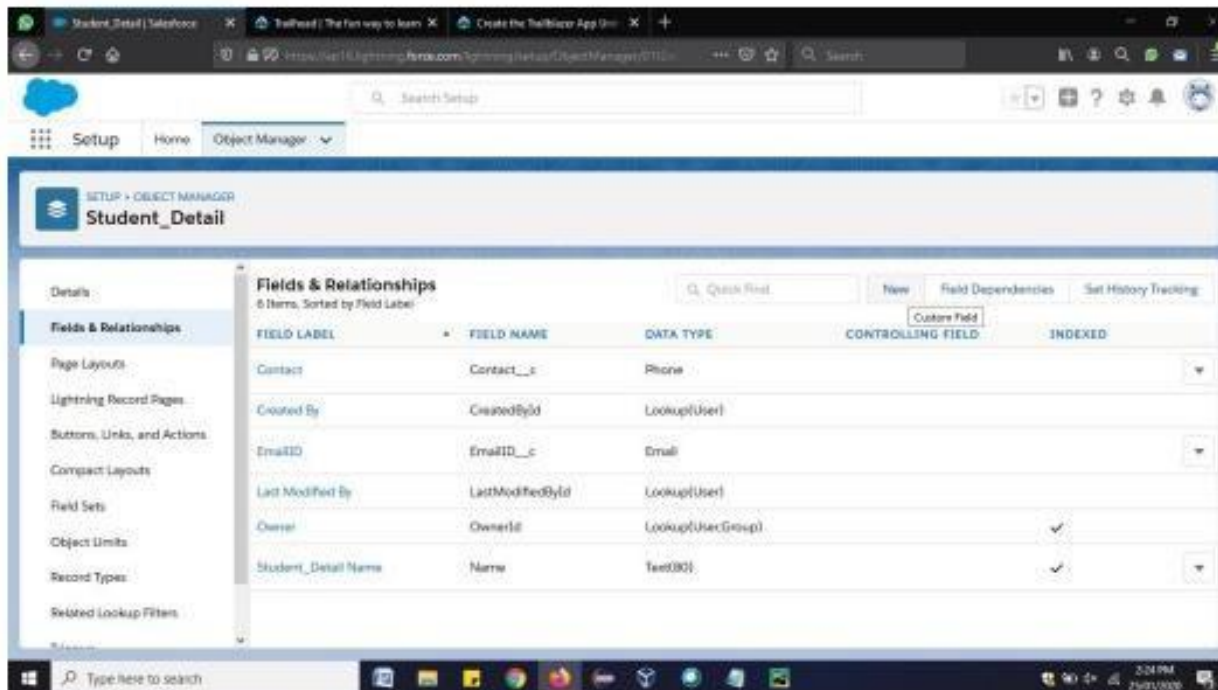
Step 4: Now, Click on Fields & Relations => New.



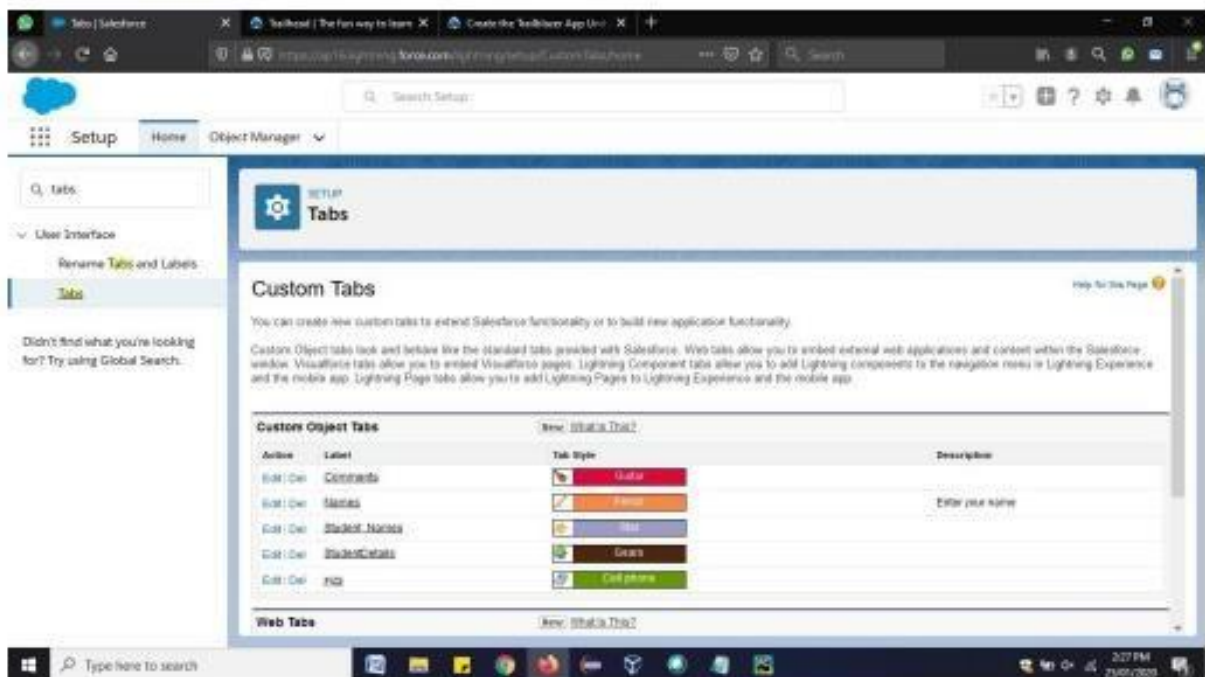
Step 5: Then select option "Email" and click Next-> Next -> Save.



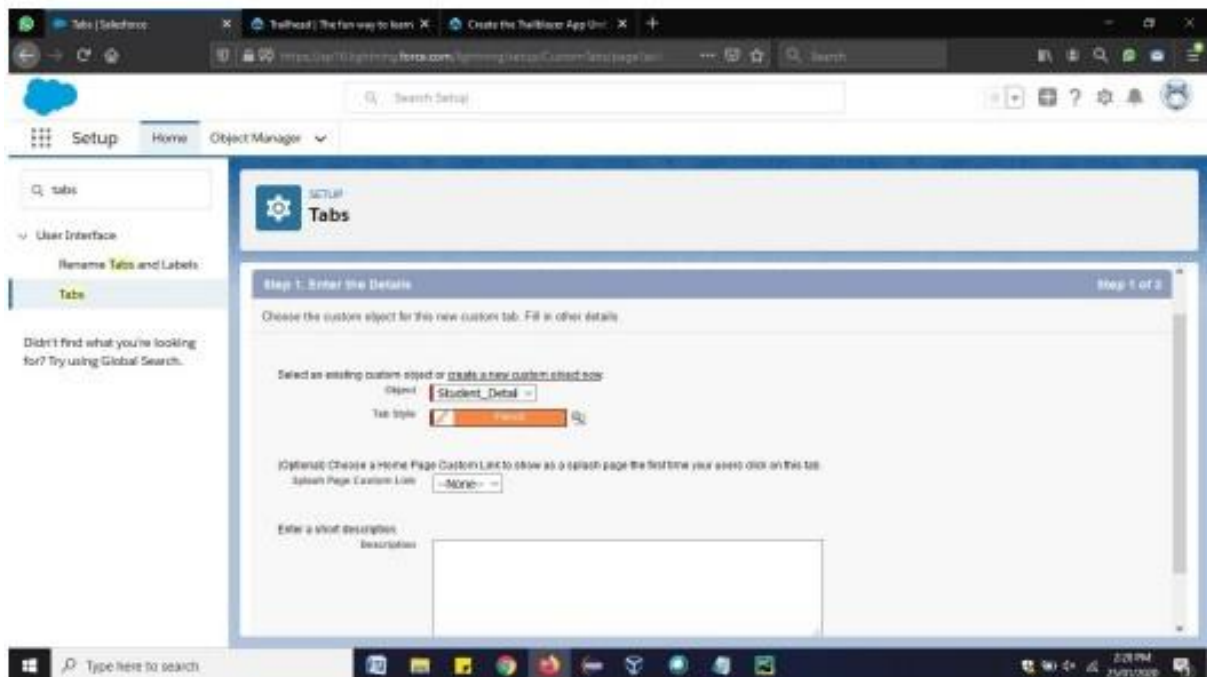
Step 6: Similarly. Repeat steps 4 and 5 to add more fields like Phone, Date of Birth. This is how the custom object will have the various fields.



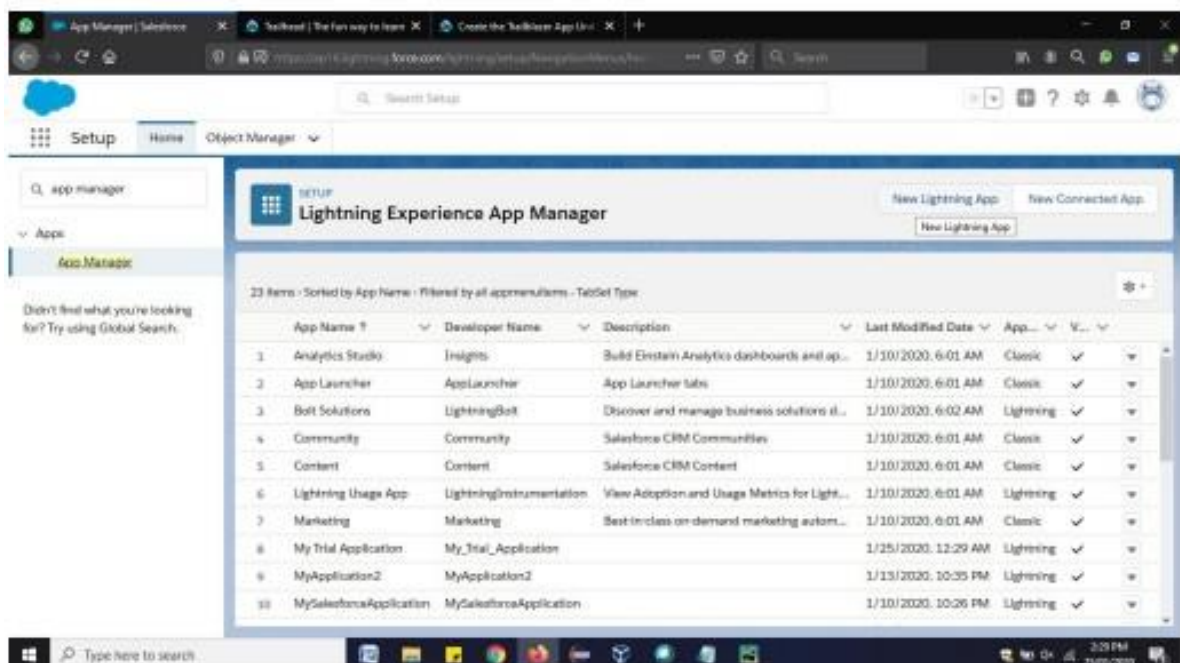
Step 7: Now go to Home => search for "Tabs". Click on New.



Step 8: Enter the Object name and select any icon for tab style. Leave all defaults as it is. Click **Next**, **Next**, and **Save**.



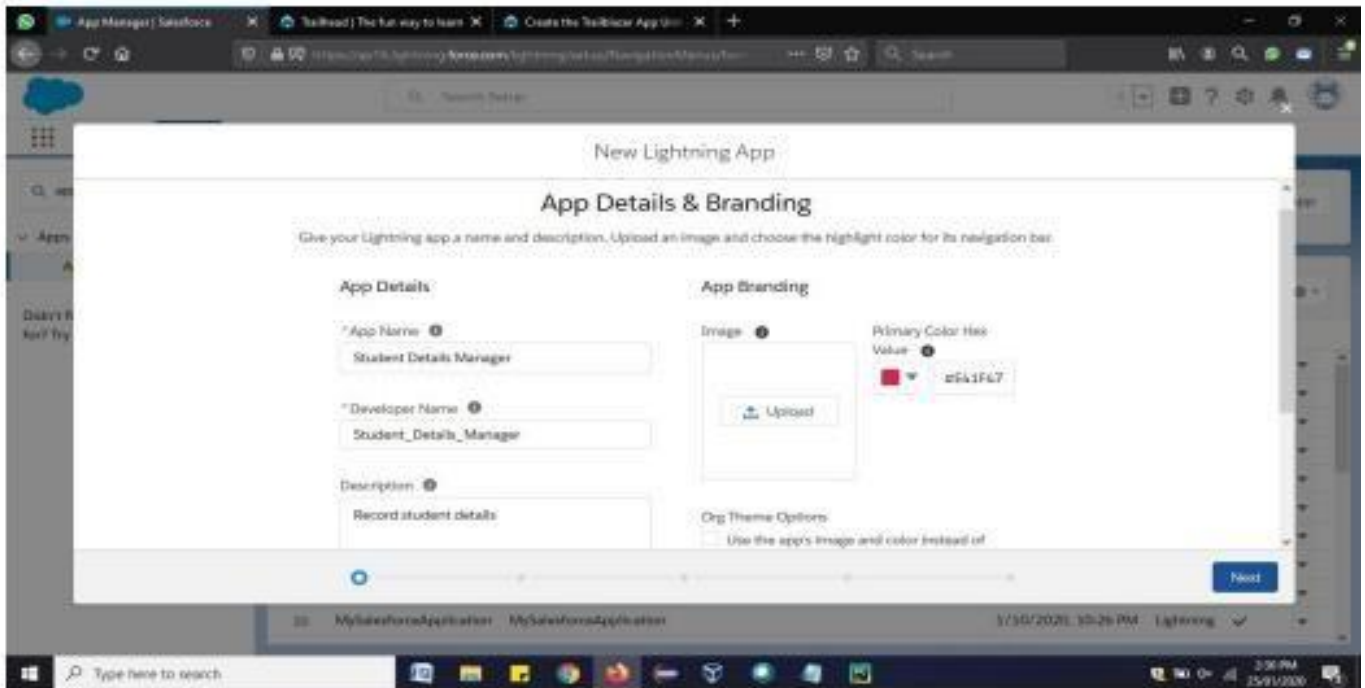
Step 9: In Setup, click **Home**. Enter “App Manager” in Quick Find and select **App Manager**. Click **New Lightning App**.



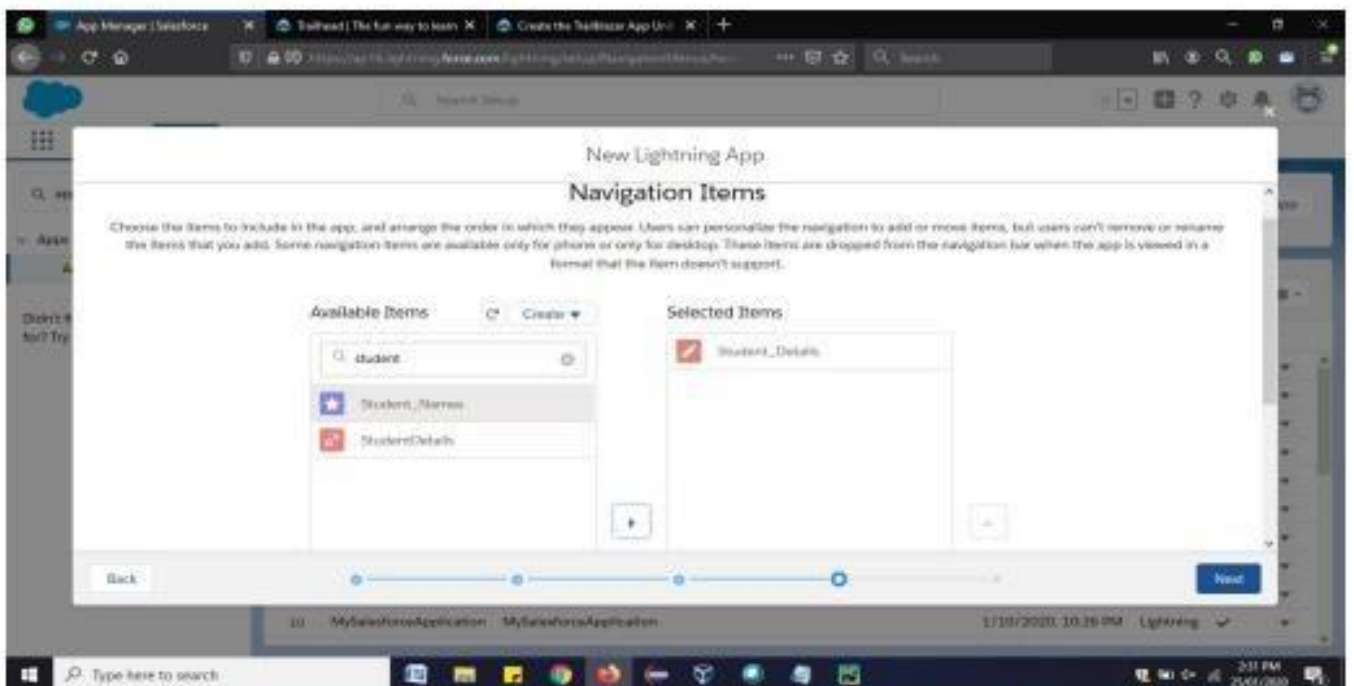
Step 10: Define the new Lightning app as follows:

App Name: Student Details Manager

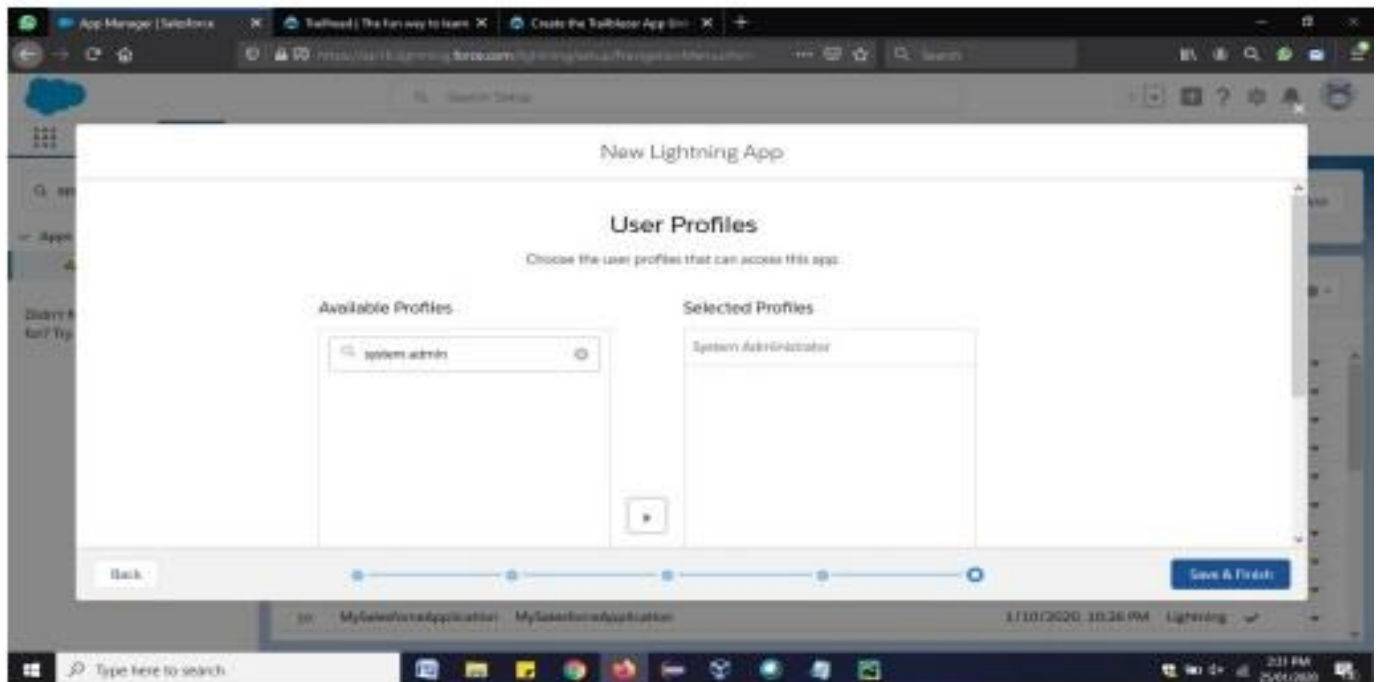
Developer Name: Student_Details_Manager. Click **Next**.



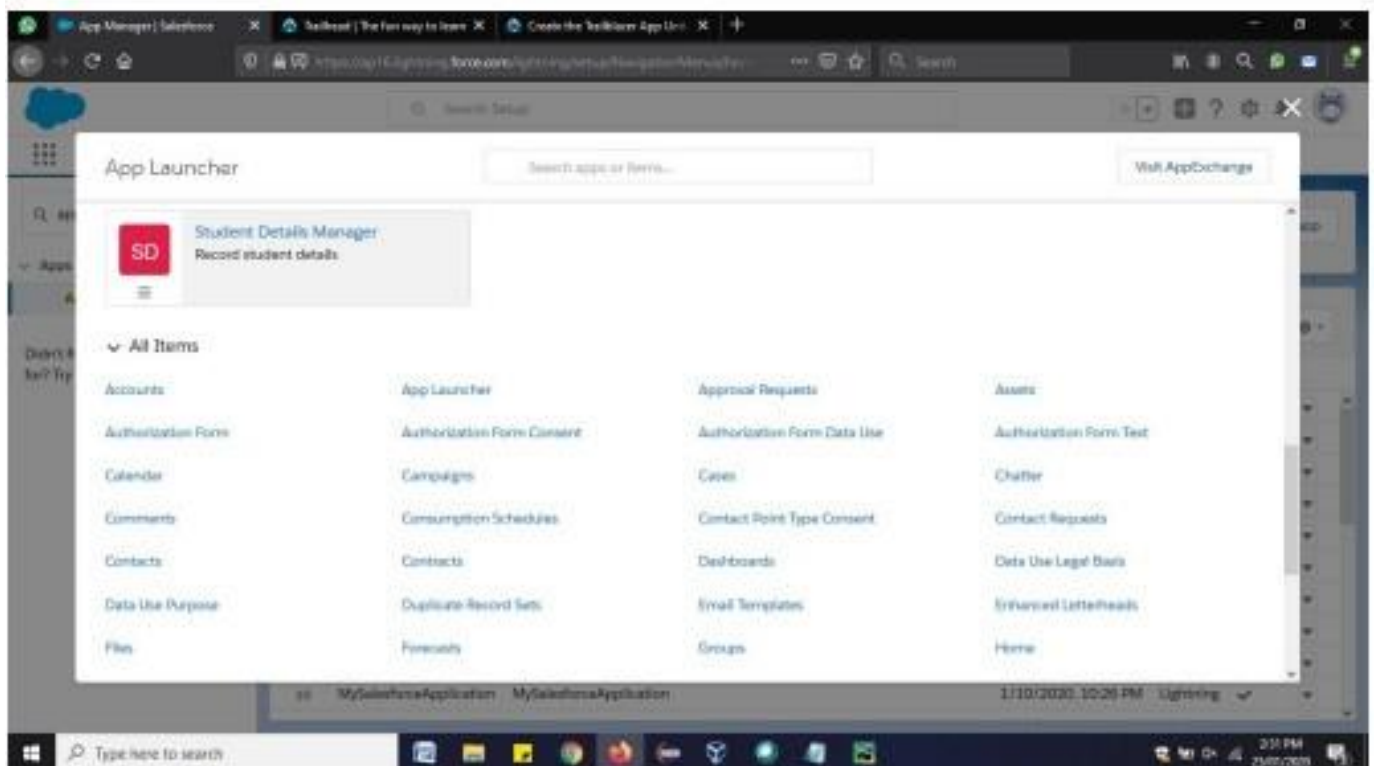
Step 11: On the App Options screen, leave the defaults as is and click **Next**. On the Utility Items screen, leave the defaults as is and click **Next**. On the Navigation Items screen, select **Student_Detail** and move them to the Selected Items box. Then click **Next**.

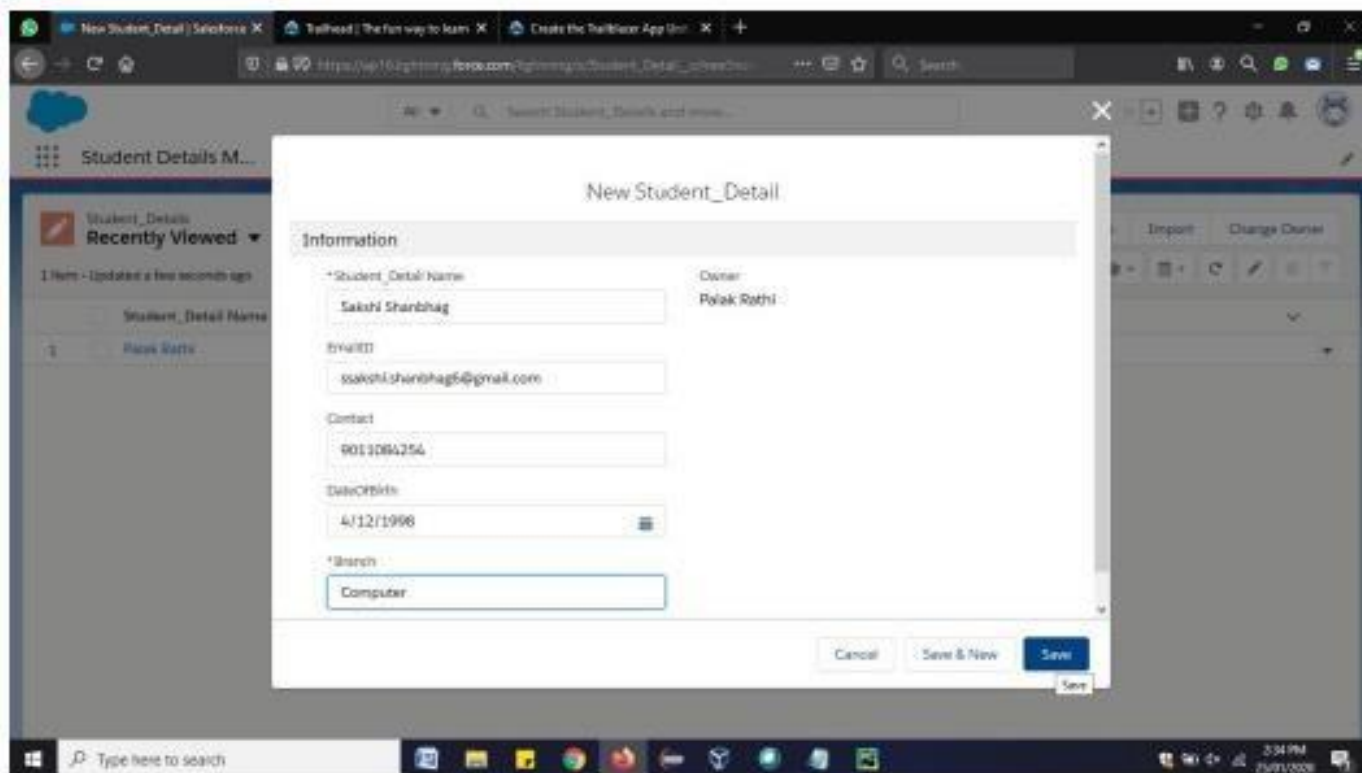


Step 12: On the Assign to User Profiles screen, select **System Administrator** and move it to Selected Profiles. Then click **Save & Finish**.

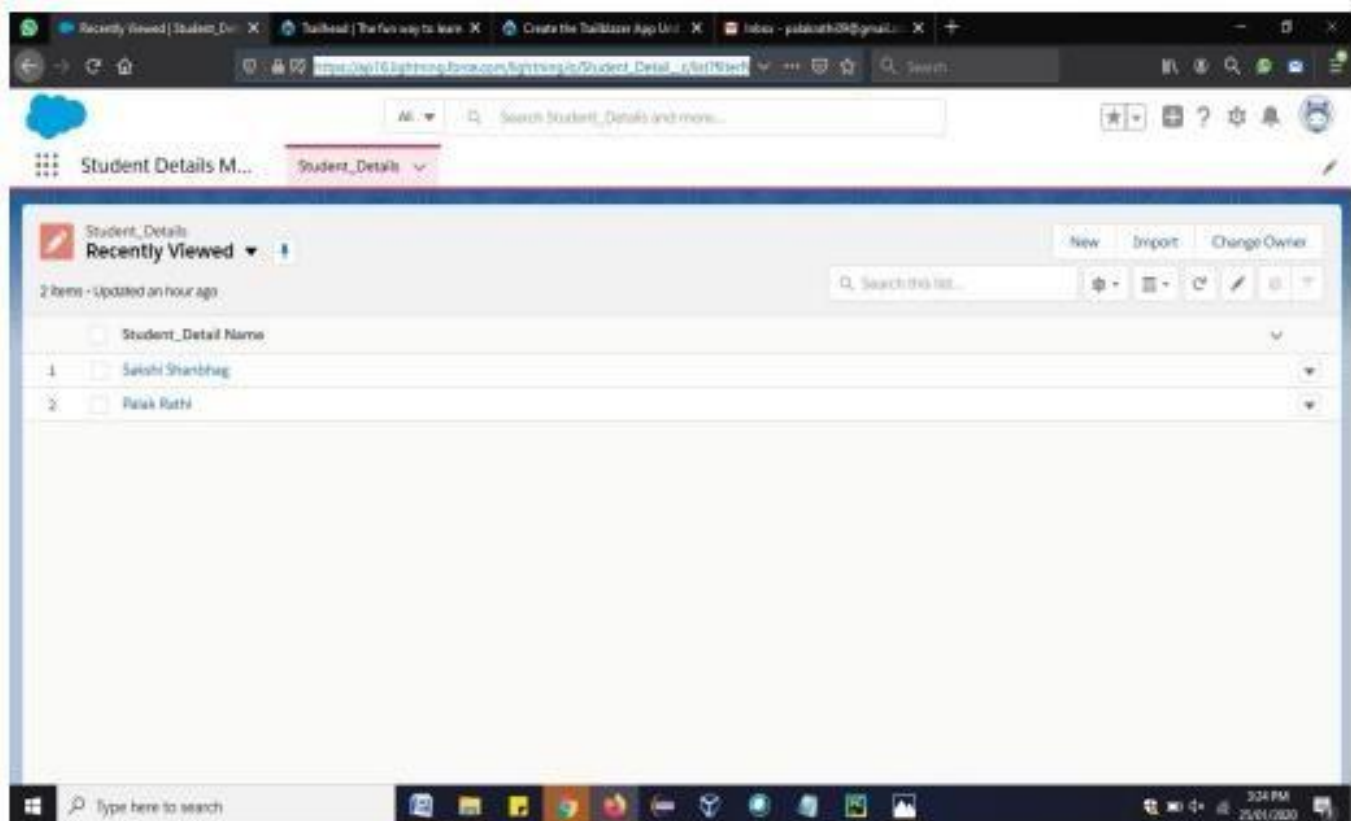


Step 13: Click on App launcher and Open the custom application created.





Step 14: Click on Student_Details => New => fill the specified details and copy the URL.



Step 15: Open Google Chrome new Tab => More Tools => Developer Tools and then paste the URL of the application, copied in the previous step.

