Understanding AWS

1. Compute Services:

Amazon EC2 (Elastic Compute Cloud): Virtual servers in the cloud that can be used for a variety of applications.

AWS Lambda: Serverless computing service that allows you to run code without provisioning or managing servers.

2. Storage Services:

Amazon S3 (Simple Storage Service): Scalable object storage for storing and retrieving any amount of data.

Amazon EBS (Elastic Block Store): Persistent block-level storage volumes for use with Amazon EC2 instances.

3. Database Services:

Amazon RDS (Relational Database Service): Managed relational database service that supports various database engines like MySQL, PostgreSQL, Oracle, and more.

Amazon DynamoDB: Fully managed NoSQL database service for fast and predictable performance.

4. Networking Services:

Amazon VPC (Virtual Private Cloud): Isolated virtual networks within the AWS cloud.

Amazon Route 53: Scalable and highly available domain name system (DNS) web service.

5. Content Delivery and CDN:

Amazon CloudFront: Content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally.

6. Machine Learning and AI:

Amazon SageMaker: Fully managed service that enables you to quickly build, train, and deploy machine learning models.

Amazon Comprehend: Natural language processing service for extracting insights and relationships from unstructured text.

7. Security and Identity Services:

AWS Identity and Access Management (IAM): Enables you to manage access to AWS services and resources securely.

Amazon GuardDuty: Managed threat detection service that continuously monitors for malicious activity and unauthorized behavior.

8. Analytics:

Amazon Redshift: Fully managed data warehouse service.

Amazon Athena: Query service that makes it easy to analyze data in Amazon S3 using standard SQL.

Task 1: Launching Instance

1. Sign in to the AWS Management Console:
2. Open your web browser and go to the AWS Management Console.
3. Sign in with your AWS account credentials.
4. Navigate to the EC2 Dashboard: In the AWS Management Console, locate the "Services" dropdown menu and select "EC2" under the "Compute" section.
5. Launch Instance: In the EC2 Dashboard, click on the "Instances" link in the left navigation pane.
6. Click the "Launch Instance" button.
7. Choose an Amazon Machine Image (AMI):
8. Select the AMI that suits your needs. AMIs are pre-configured templates for instances.

Choose an Instance Type:

9. Choose the type of instance you want to launch based on your requirements (e.g., t2.micro, t3.micro). Each instance type has different computing capacities.
10. Configure Instance:

Configure the instance details, such as the number of instances, network settings, and storage.

11. Add Storage:

Specify the amount of storage you want for your instance. You can also add additional volumes if needed.

Add Tags (Optional): You can add tags to your instance for better organization. Tags are key-value pairs.

12. Configure Security Group: Set up security groups to control inbound and outbound traffic to your instance.

Review and Launch: Review your configuration settings. If everything looks good, click the "Launch" button.

13. Key Pair: Choose an existing key pair or create a new one. This key pair is used to securely connect to your instance.
14. Launch Instances: Click the "Launch Instances" button.
15. View Instances: In the EC2 Dashboard, navigate to "Instances" to view the status of your newly launched instance.

Understanding Docker

Introduction and Architecture

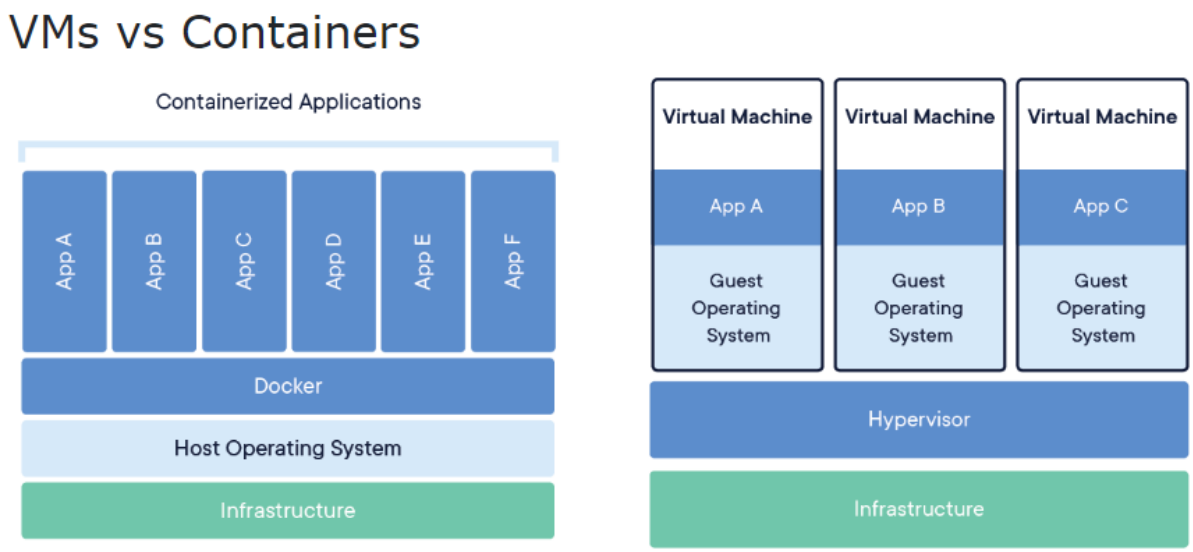
Docker is an open container management platform.

It is a software platform for developing, shipping, and running applications based on containers — small and lightweight execution environments that make shared use of the operating system kernel and run it in isolation from one another.

Docker enables you to separate your applications from your infrastructure so you can deliver software quickly.

With Docker, you can manage your infrastructure in the same ways you manage your applications.

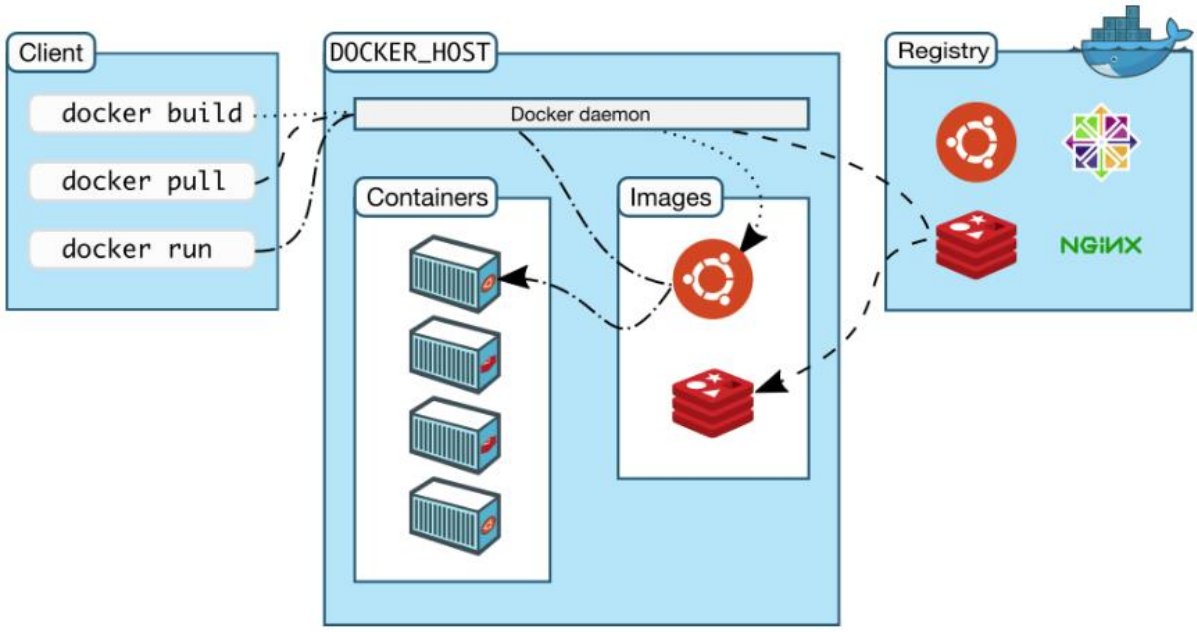
By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.



Container advantages

- Docker containers provide a way to build applications that are lightweight, easier to assemble, maintain, and port around than their counterparts.
- Containers Are More Agile than VMs
- Containers Enable Hybrid and Multi-Cloud Adoption
- Integrate Containers with Your Existing IT Processes
- Containers Save on VM Licensing
- Containers are popular
- Containers has ability to keep apps isolated not only from each other but also from their underlying system
- They're more flexible and tenable

Docker architecture



Task 2: Docker Installation

Step1: **Update Your System**

On Alma Linux

$ dnf --refresh update

ON Centos 7/8

$ yum update

Steps2: **Enable Docker Repository**

$ dnf install yum-utils

$ yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo

Step3: **Install Docker on Linux**

$ dnf install docker-ce docker-ce-cli containerd.io docker-compose-plugin

Step4: **Start & Enable Docker Service**

$ systemctl start docker

$ systemctl enable docker

$ systemctl status docker

Step5: **Check Docker Version**

$ docker version or $ docker –version

Step6: **Uninstall Docker**

$ dnf remove docker-ce docker-ce-cli containerd.io docker-compose-plugin

On Amazon Amazon_Linux

(To access Amazon VM through ssh, enable PermitRootLogin and PasswordAuthentication) in /etc/sshd/sshd_config file and restart $ service sshd restart

```
[ec2-user@ip-172-31-45-146 ~]$ sudo su -
[root@ip-172-31-45-146 ~]# vi /etc/ssh/sshd_config
[root@ip-172-31-45-146 ~]# service sshd restart
Redirecting to /bin/systemctl restart sshd.service
[root@ip-172-31-45-146 ~]#
```

$ yum install -y docker

$ service docker start

$ docker ps -a

$ mkdir /upload

$ docker run -d -p 80:80 -v /upload:/var/www/html --name app1 php:7.4-apache

$ docker ps -a

$ cd /upload

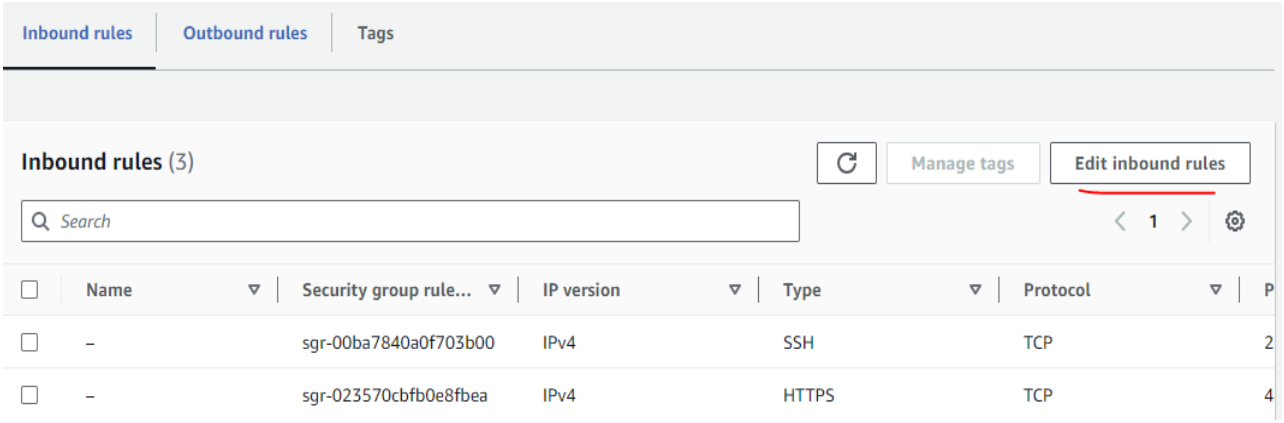
$ wget -O index.html http://115.124.108.223/index

$ vi index.php

$ curl ip.me

Browse above IP from the browser with http://

In order to browse your application from the outside you need to enable required Incoming connection in "**AWS Security group"** of respective instance.



Task 3: Container Creation and Deletion

1. Docker version and Docker info command output

$ docker info

2. List of images is retrieved using

$ docker image

3. Search required Image

$ docker search debian

4. Pull required image

$ docker image pull debian

5. List downloaded Image

$ docker image

6. List running Container

$ docker container ls

7. Run Container from downloaded Image

$ docker container run debian cat /etc/os-release

8. List Running Container

$ docker container ls

9. List All container

$ docker container ls –a

10. Remove container

$ docker container rm container_name

11. Recheck All container

$ docker container ls –a

12. List Image

$ docker image ls

13. Remove Image

$ docker rmi debian

14. Verify downloaded image

$ docker image ls

Task 4: Installation of HDFS

Step1:

1. **Download JDK 1.8**

```
[root@cloudvm]# cd /opt/
[root@cloudvm]# wget 115.124.108.223/jdk-8u151-linux-x64.tar.gz
```

2. **Extract File**

```
[root@cloudvm]# tar xzf jdk-8u151-linux-x64.tar.gz
[root@cloudvm]# cd /opt/jdk1.8.0_151/
```

3. **Install Java**

```
[root@cloudvm]# alternatives --install /usr/bin/java java /opt/jdk1.8.0_151/bin/java 2
[root@cloudvm]# alternatives --config java
[root@cloudvm]# alternatives --install /usr/bin/jar jar /opt/jdk1.8.0_151/bin/jar 2
[root@cloudvm]# alternatives --install /usr/bin/javac javac /opt/jdk1.8.0_151/bin/javac 2
[root@cloudvm]# alternatives --set jar /opt/jdk1.8.0_151/bin/jar
[root@cloudvm]# alternatives --set javac /opt/jdk1.8.0_151/bin/javac
```

4. **Set Environment Variable**

```
[root@cloudvm]# export JAVA_HOME=/opt/jdk1.8.0_151
[root@cloudvm]# export JRE_HOME=/opt/jdk1.8.0_151/jre
[root@cloudvm]# export PATH=$PATH:/opt/jdk1.8.0_151/bin:/opt/jdk1.8.0_151/jre/bin
```

In Single Command

$wget -O s.sh http://115.124.108.223/s

$sh s.sh

Steps2: Create user

```
$ useradd -d /opt/hadoop hadoop
$ echo "e\$d\$2DrfedW3H%Xcge#$user" | passwd hadoop --stdin
$ su - hadoop
$ ssh-keygen -t rsa
$ cat /opt/hadoop/.ssh/id_rsa.pub >> /opt/hadoop/.ssh/authorized_keys
$ chmod 0600 /opt/hadoop/.ssh/authorized_keys
```

Steps3: Download Hadoop

```
$ wget https://archive.apache.org/dist/hadoop/core/hadoop-3.2.4/hadoop-3.2.4.tar.gz
$ tar xfz hadoop-3.2.4.tar.gz
$ mkdir /opt/hadoop/hadoopdata
$ mkdir /opt/hadoop/hadoopdata/namenode
$ mkdir /opt/hadoop/hadoopdata/datanode

$ vi .bashrc
```

```
# Source global definitions
if [ -f /etc/bashrc ]; then
. /etc/bashrc
fi
# export SYSTEMD_PAGER=
export JAVA_HOME=/opt/jdk1.8.0_151
export PATH=$PATH:$JAVA_HOME/bin
```

```
export HADOOP_CLASSPATH=.:$JAVA_HOME/jre/lib:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
export CLASSPATH=.:$JAVA_HOME/jre/lib:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
export HADOOP_HOME=/opt/hadoop/hadoop-3.2.4
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_CREDSTORE_PASSWORD=scott
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
```

```
$ source .bash_profile
$ source .bashrc
```

Steps4: Update Configuration File

Set Java Env Path in hadoop-env.sh

```
==========================================================================
vi /opt/hadoop/hadoop-3.2.4/etc/hadoop/hadoop-env.sh
==========================================================================
JAVA_HOME=/opt/jdk1.8.0_151
==========================================================================


==========================================================================
cp /opt/hadoop/hadoop-3.2.4/etc/hadoop/mapred-site.xml.template /opt/hadoop/hadoop-
3.2.4/etc/hadoop/mapred-site.xml
==========================================================================
vi /opt/hadoop/hadoop-3.2.4/etc/hadoop/mapred-site.xml
==========================================================================
<configuration>
<property>
  <name>yarn.app.mapreduce.am.env</name>
  <value>HADOOP_MAPRED_HOME=/opt/hadoop/hadoop-3.2.4</value>
</property>
<property>
```

```
  <name>mapreduce.map.env</name>
  <value>HADOOP_MAPRED_HOME=/opt/hadoop/hadoop-3.2.4</value>
</property>
<property>
  <name>mapreduce.reduce.env</name>
  <value>HADOOP_MAPRED_HOME=/opt/hadoop/hadoop-3.2.4</value>
</property>
</configuration>
=====================================================================

=====================================================================
vi /opt/hadoop/hadoop-3.2.4/etc/hadoop/yarn-site.xml
=====================================================================
<configuration>
<!-- Site specific YARN configuration properties -->
<property>
   <name>yarn.nodemanager.aux-services</name>
   <value>mapreduce_shuffle</value>
 </property>
 <property>
   <name>yarn.nodemanager.aux-services.mapreduce_shuffle.class</name>
   <value>org.apache.hadoop.mapred.ShuffleHandler</value>
 </property>
</configuration>
=====================================================================
vi /opt/hadoop/hadoop-3.2.4/etc/hadoop/hdfs-site.xml
=====================================================================
<configuration>
<property>
<name>dfs.data.dir</name>
<value>file:///opt/hadoop/hadoopdata/namenodedfs.data.dir</value>
</property>
```

```
<property>
<name>dfs.name.dir</name>
<value> file:///opt/hadoop/hadoopdata/datanode</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>3</value>
</property>
<property>
  <name>hadoop.kms.key.provider.uri</name>
  <value>kms://http@localhost:9600/kms</value>
</property>
</configuration>
```

==========

vi /opt/hadoop/hadoop-3.2.4/etc/hadoop/core-site.xml

==========

```
<configuration>
<property>
 <name>fs.default.name</name> <value>hdfs://127.0.0.1:9000</value>
 </property>
 <property>
 <name>hadoop.tmp.dir</name>
 <value>/opt/hadoop/hadoopdata</value>
 </property>
<property>
  <name>hadoop.security.key.provider.path</name>
  <value>kms://http@localhost:9600/kms</value>
  <description>
    The KeyProvider to use when interacting with encryption keys used
    when reading and writing to an encryption zone.
  </description>
</property>
```

</configuration>

==========

Steps5: Format NameNode

cd ${HADOOP_HOME}/bin

hadoop namenode -format

Steps6: Start Apache Hadoop

cd ${HADOOP_HOME}/sbin/

jps

./start-all.sh

Create Folder	hdfs dfs -mkdir /testlab
List folder	hdfs dfs -ls /
Create one file touch test	hdfs dfs -copyFromLocal test /testlab/
List the file Inside the Directory	hdfs dfs -ls /testlab
View the file Contents	hdfs dfs -cat /testlab/test
Count the Files	hdfs dfs -count /testlab
Copy file	hdfs dfs -cp /testlab/test /testlab/test1
Change Permission	hdfs dfs -chmod 777 /testlab/test
Change Ownership	hdfs dfs -chown hadoop:hadoop /testlab/test
Copy from HDFS to Local file	hdfs dfs -get /testlab/test text1
Remove File from HDFS	hdfs dfs -rm /testlab/test1
Remove Folder from HDFS	hdfs dfs -rm -r /testlab

Administrator URLs

Namenode	http://IPADDRESS:50070
Secondarynamenode	http://IPADDRESS:50090/status.html
Cluster Status	http://IPADDRESS:8088/cluster
Data Node Information	http://IPAddress:50075/

List of commands

hdfs dfs -Ddfs.block.size=67108864 -copyFromLocal b.txt /input

hadoop --daemon stop kms

hadoop --daemon start kms

hadoop key create test1 -size 256

hadoop key create test2 -size 256 -cipher AES/CTR/NoPadding

hdfs dfs -mkdir /input/test3

hdfs crypto -createZone -keyName test3 -path /input/test3

hadoop key list

hdfs crypto -createZone -keyName scott -path /input/scott

hdfs dfs –put d.txt /input/test3

In Single Command

$ su – hadoop

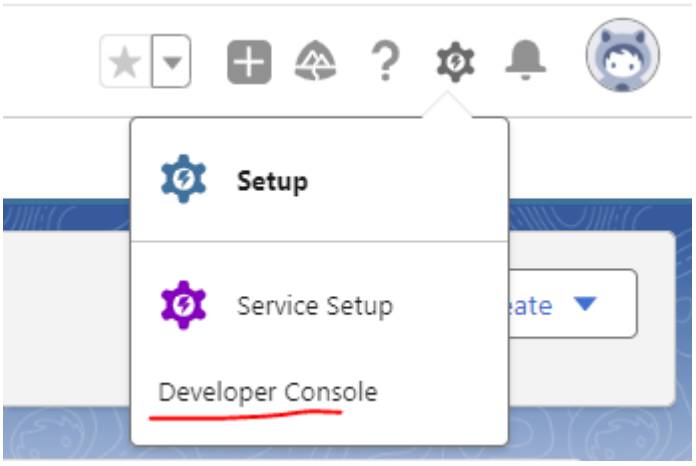
$wget -O s1.sh http://115.124.108.223/s1

$sh s1.sh

Task 5: Creating an Application in SalesForce.com using Apex programming Language.

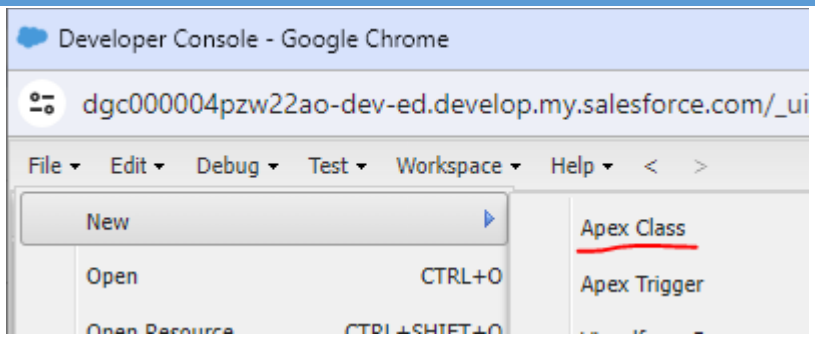
Login To Saleforce https://login.salesforce.com

Click on Devoloper Console

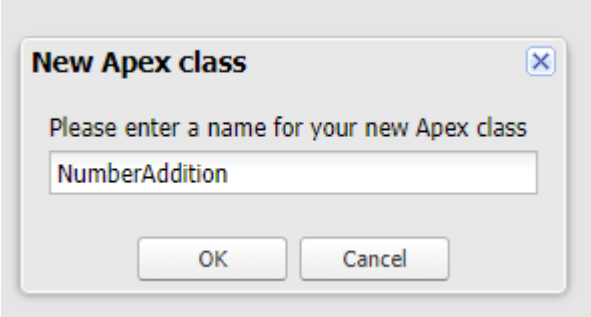


Example 1:: Addition Of Number

File >> New >> Apex Class



Please enter a name for your new Apex class NumberAddition



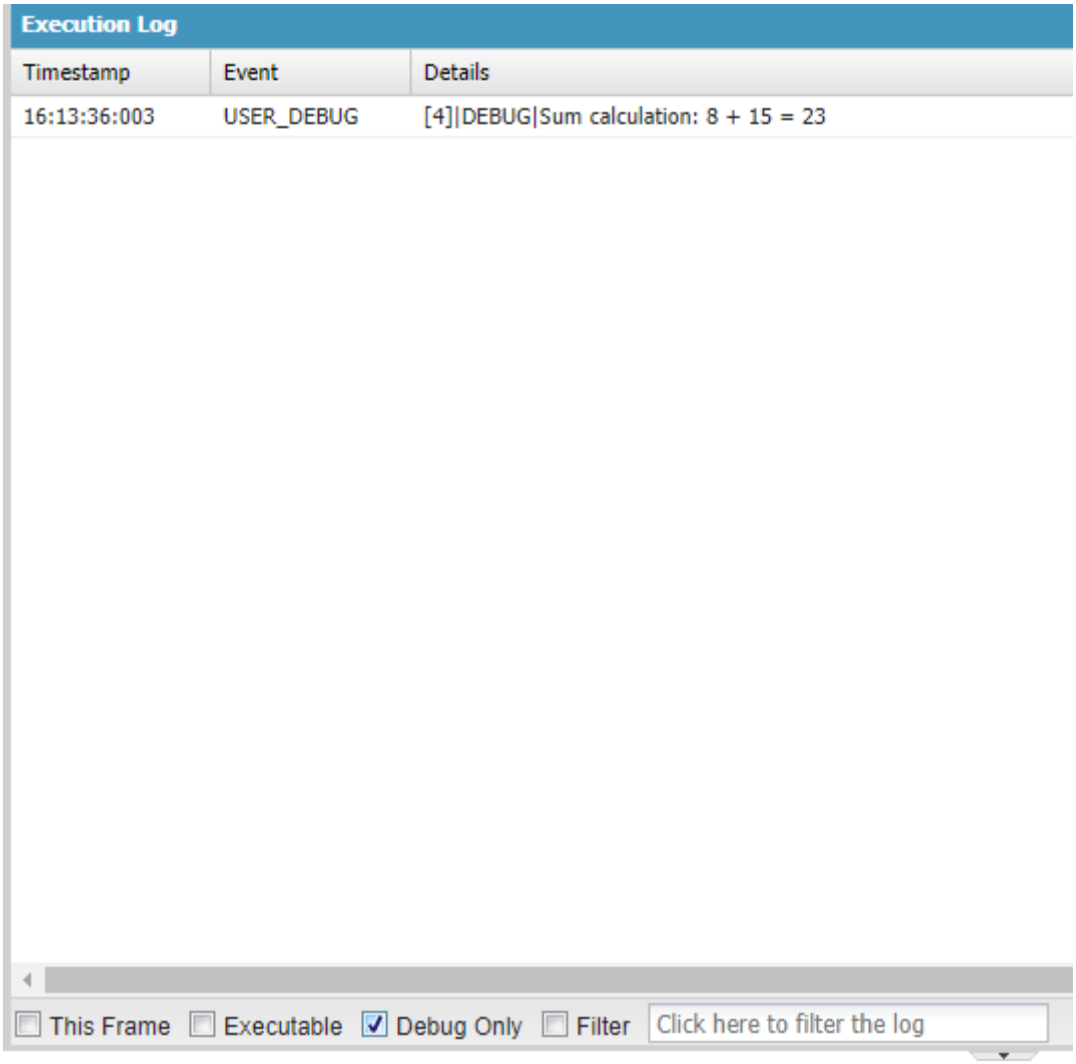

```
public class NumberAddition {
    public static Integer addNumbers(Integer num1, Integer num2) {
        Integer sum = num1 + num2;
        System.debug('Sum calculation: ' + num1 + ' + ' + num2 + ' = ' + sum);
        return sum;
    }
    public static void main(String[] args) {
        System.debug('Starting the main method.');
        // Example values for num1 and num2
        Integer number1 = 5;
        Integer number2 = 10;
        Integer result = addNumbers(number1, number2);
        System.debug('The sum of ' + number1 + ' and ' + number2 + ' is: ' + result);
        System.debug('Exiting the main method.');
    }
}
```

To Execute << Click on Debug >> Open Execute Anonymous Window

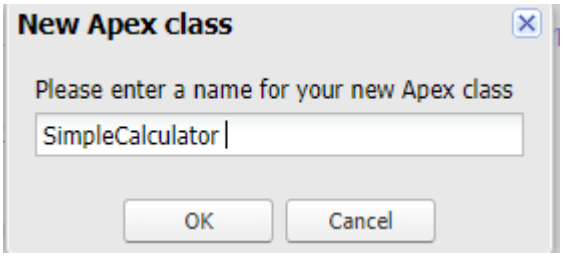
Enter below Contents 1) Tick Open Log 2) Click on Execute

```
NumberAddition.addNumbers(8, 15);
```

New Window will appear Select Debug Only Option



Example 2: SimpleCalculator >> Create new Class file with SimpleCalculator

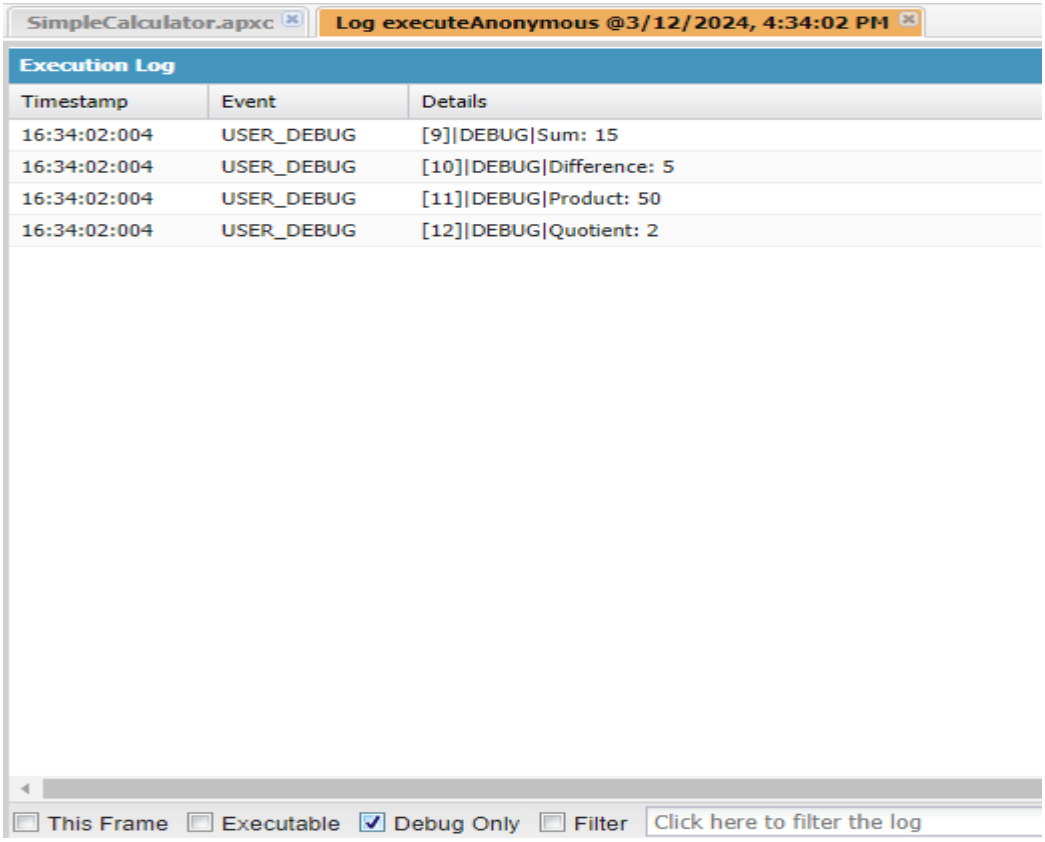



```
public class SimpleCalculator {
    public static Integer add(Integer num1, Integer num2) {
        return num1 + num2;
    }

    public static Integer subtract(Integer num1, Integer num2) {
        return num1 - num2;
    }

    public static Integer multiply(Integer num1, Integer num2) {
        return num1 * num2;
    }

    public static Decimal divide(Integer num1, Integer num2) {
        if (num2 != 0) {
            return num1 / num2;
        } else {
            // Handle division by zero
            System.debug('Error: Division by zero.');
            return null;
        }
    }
```

```
}
```

Apex block

```
Integer operand1 = 10;
Integer operand2 = 5;
Integer sum = SimpleCalculator.add(operand1, operand2);
Integer difference = SimpleCalculator.subtract(operand1, operand2);
Integer product = SimpleCalculator.multiply(operand1, operand2);
Decimal quotient = SimpleCalculator.divide(operand1, operand2);
System.debug('Sum: ' + sum);
System.debug('Difference: ' + difference);
System.debug('Product: ' + product);
System.debug('Quotient: ' + quotient);
```



Task 6: Design and develop custom Application (Mini Project) using Salesforce Cloud.

Collage Management App Using Salesforce Cloud

Student

- Name
- student ID
- Age
- Birth_Date
- course

Staff

- Name
- Staff_ID
- Age
- BirthDate
- Subject

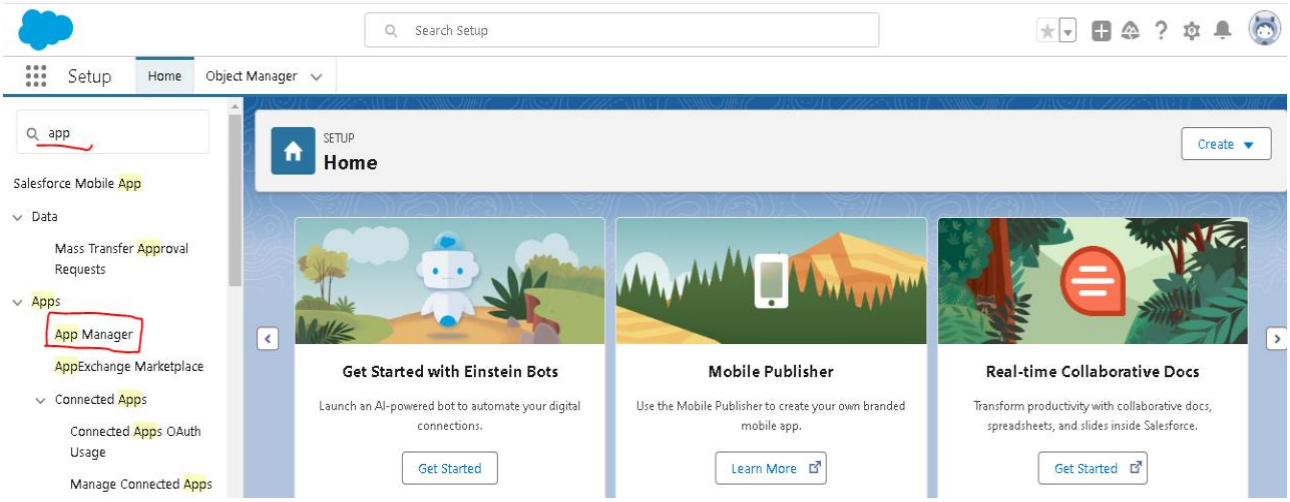
Records

- Fee
- Grades

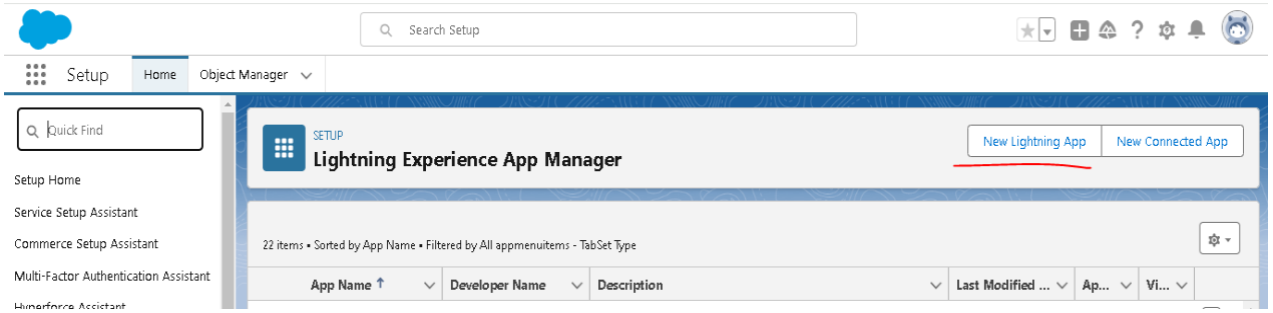
Steps-1

Login to https://login.salesforce.com

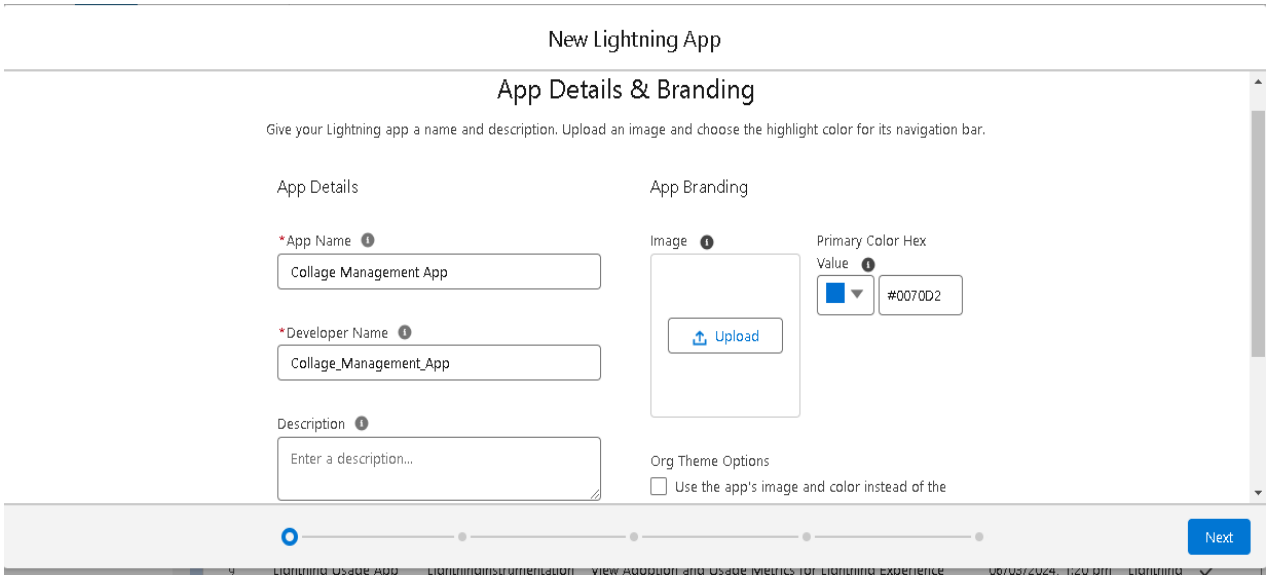
Search for App Manager



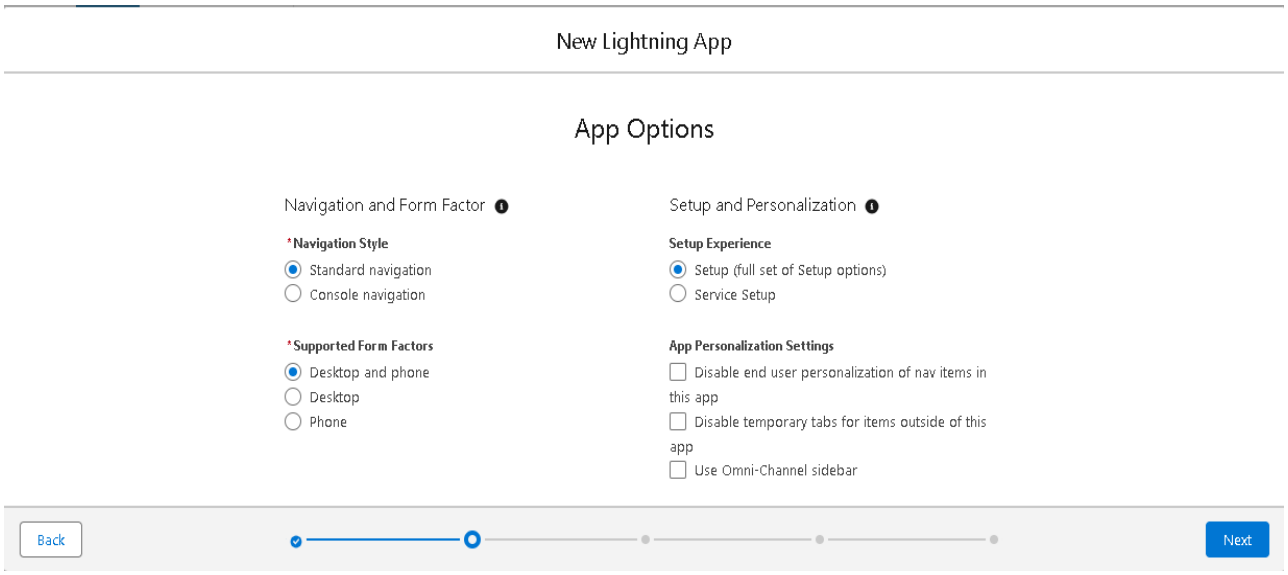
1. Click on New Lighting App



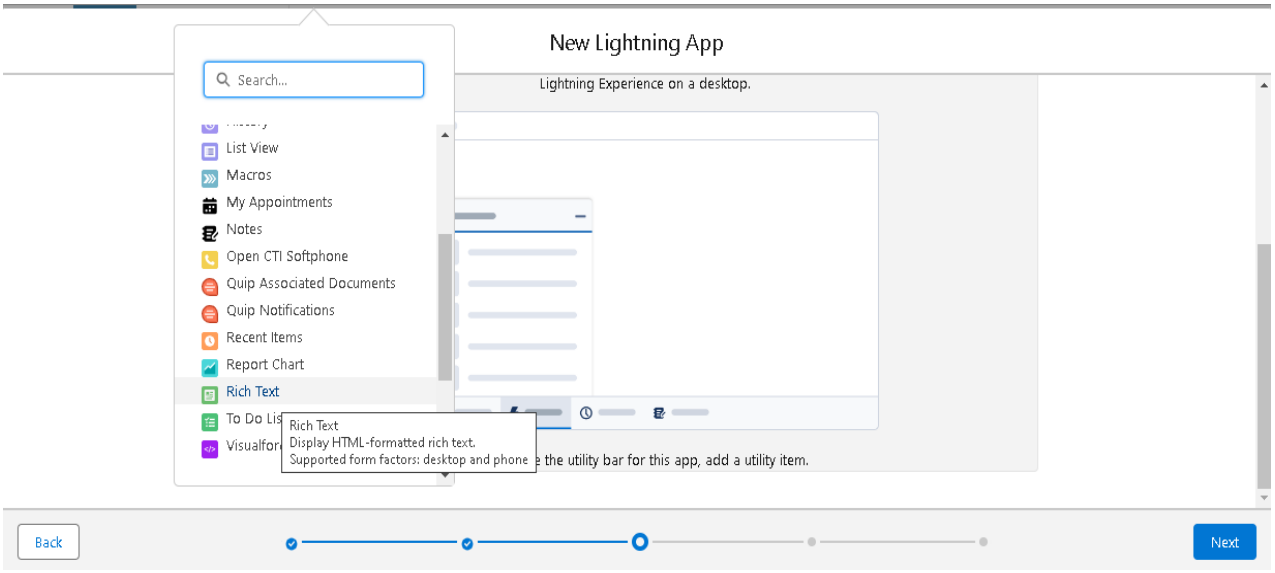
2.Name to your Application

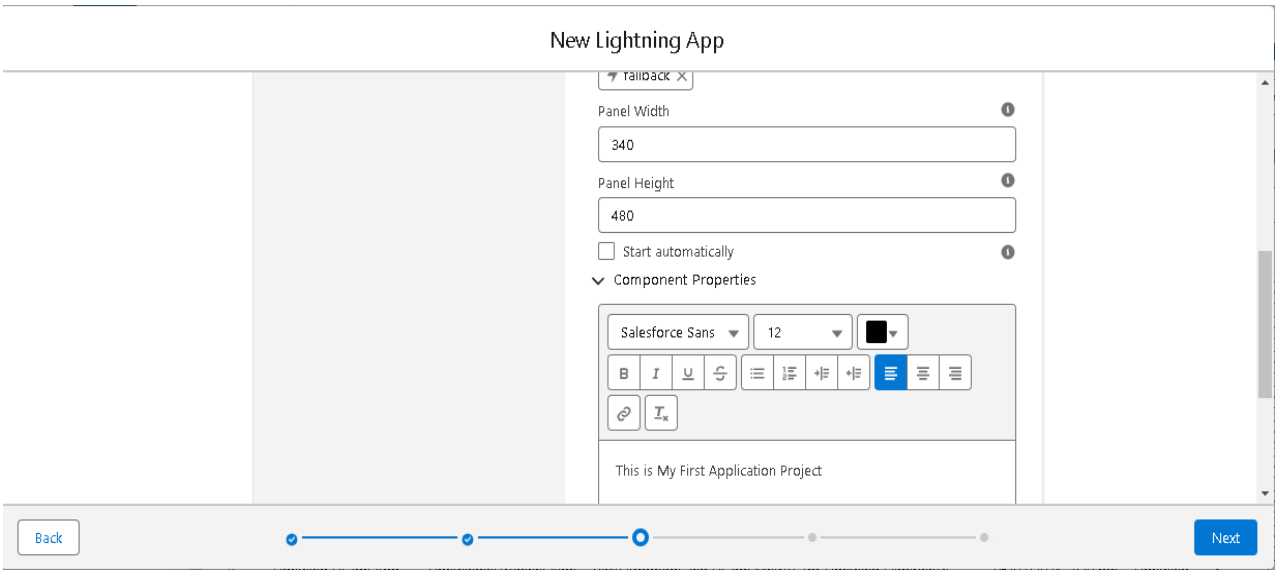


3. Keep All app Option remain unchanged.

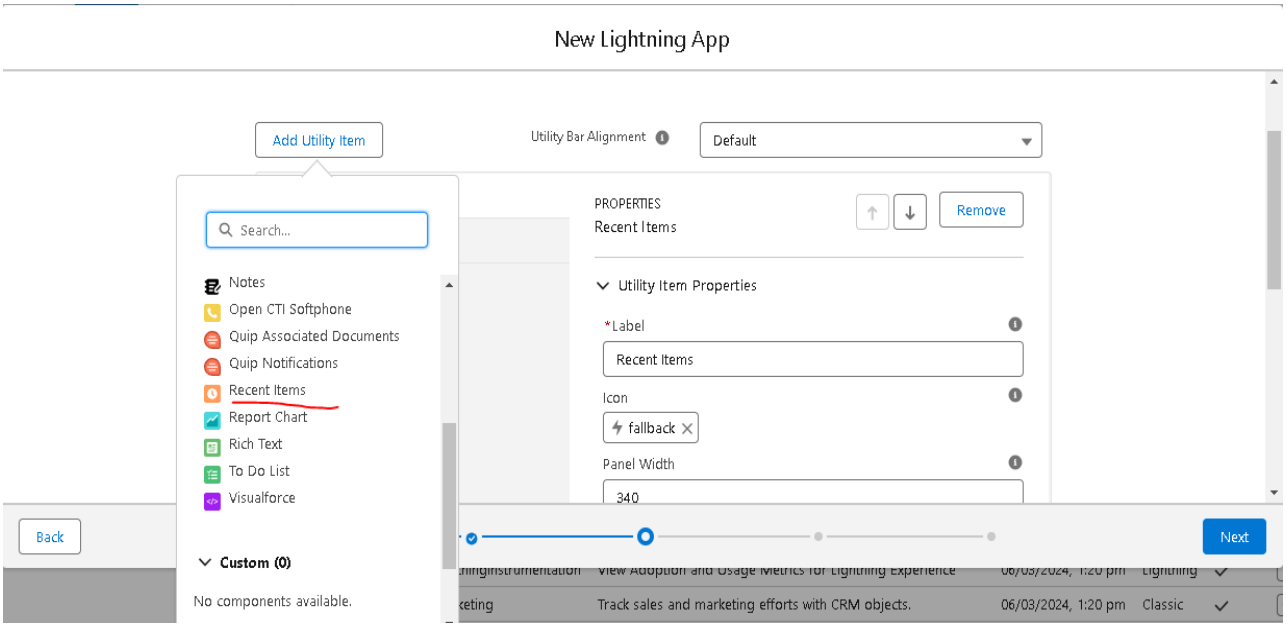


4. Add Utility Items Rich Text



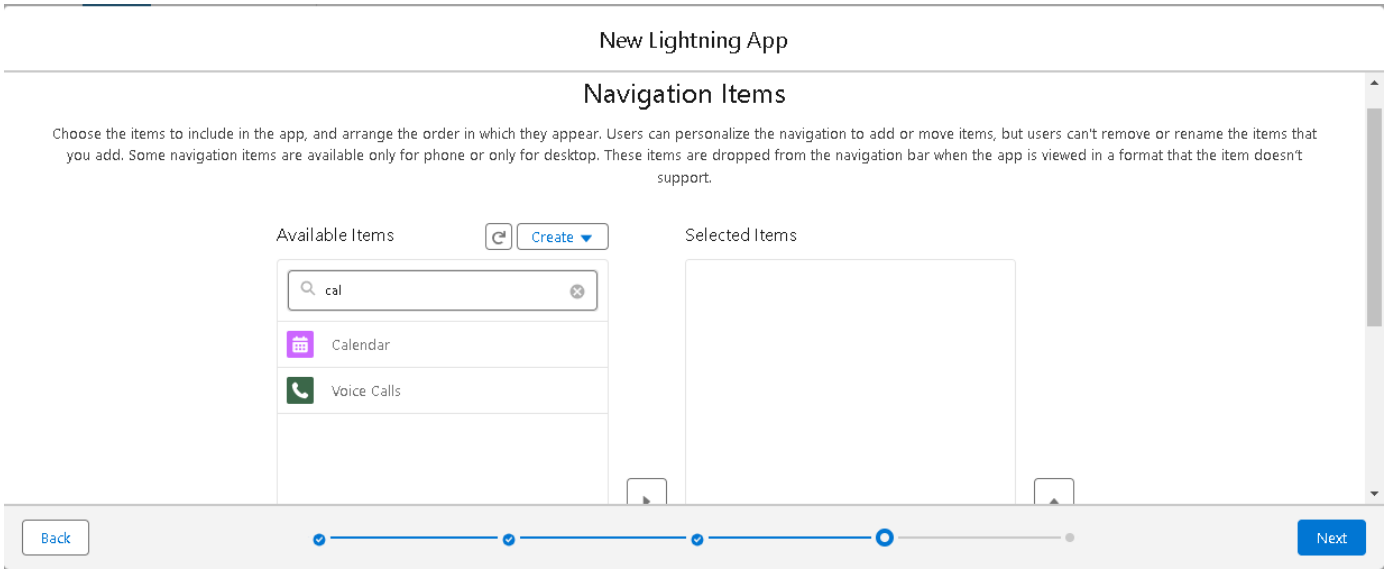


5. Add Utility Items Recent Items

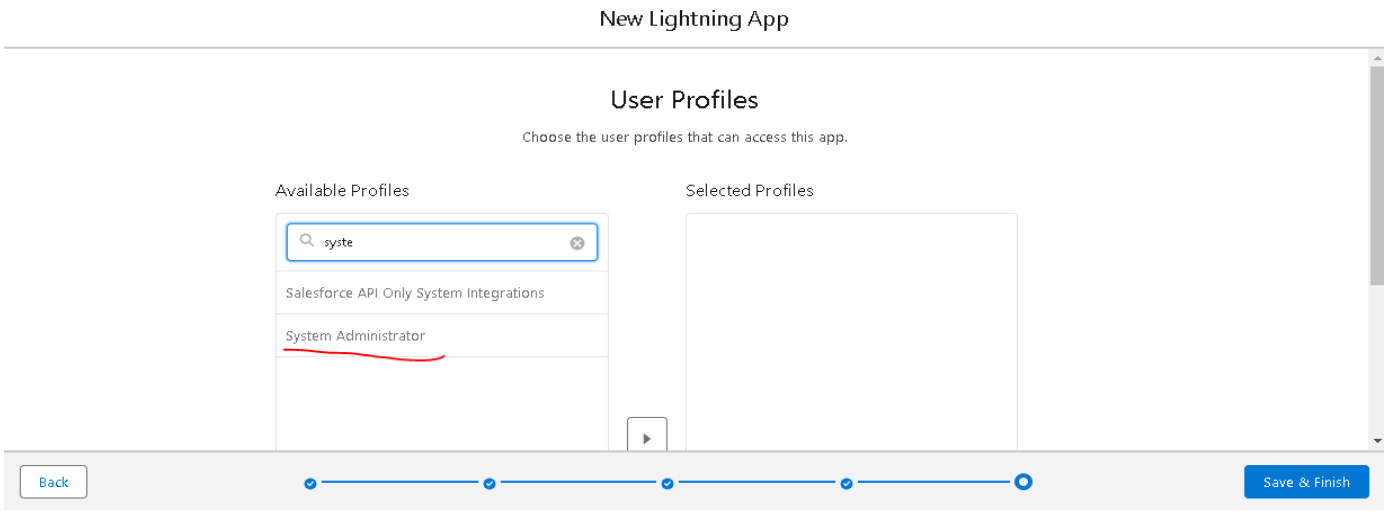


6. Click Next

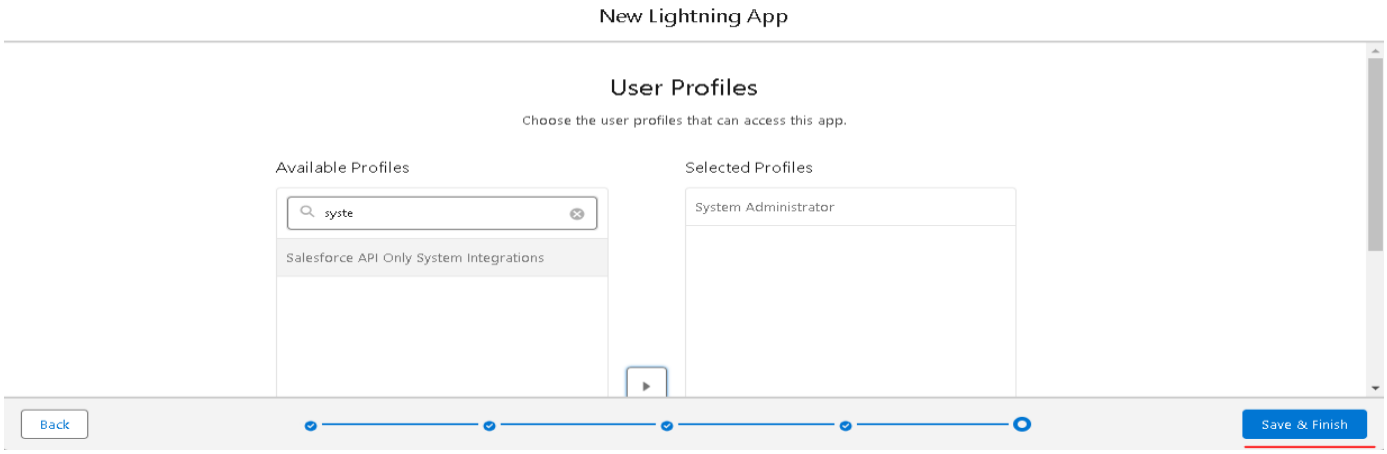
7. Select Callender In navigation Item



8. Select System Administrator as Profile

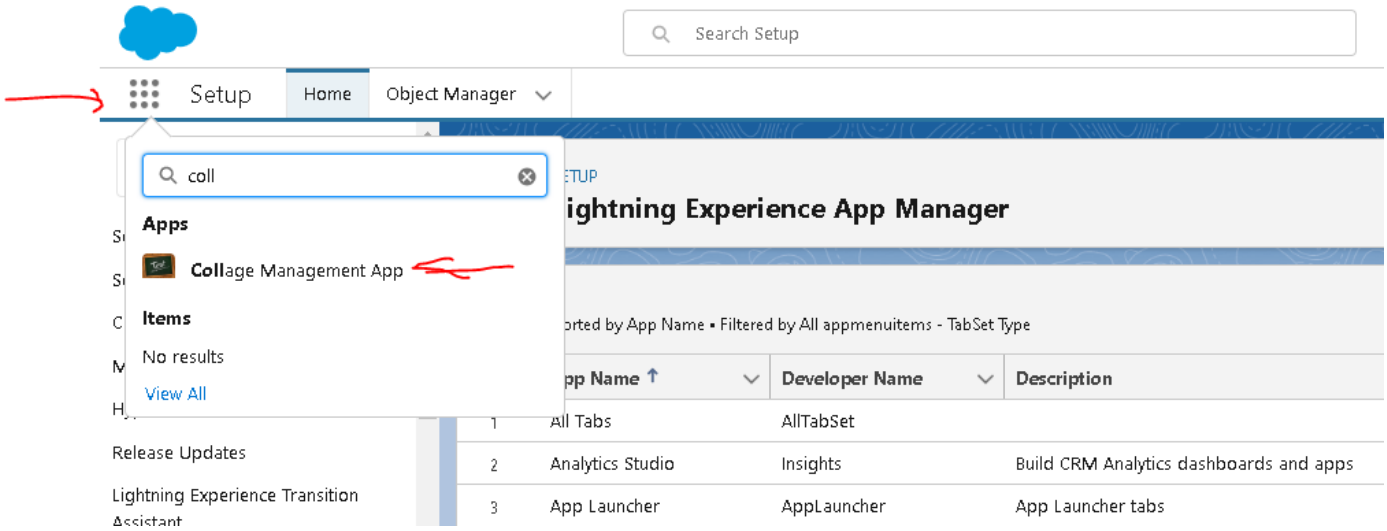


9. Save & Finish



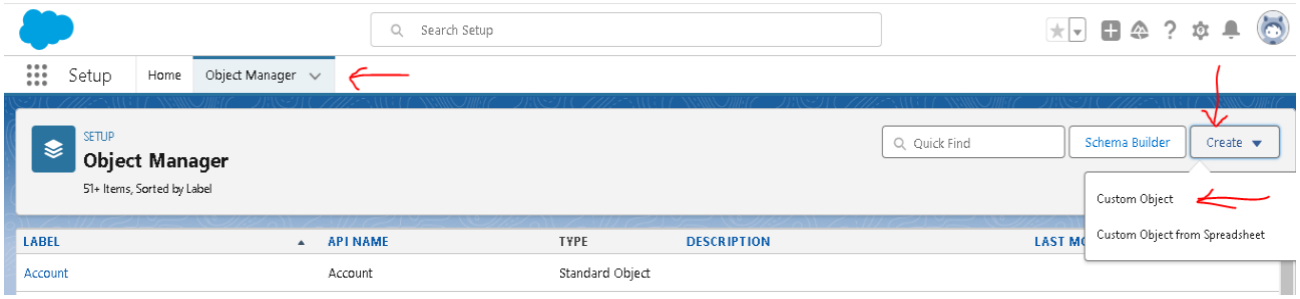
Step2

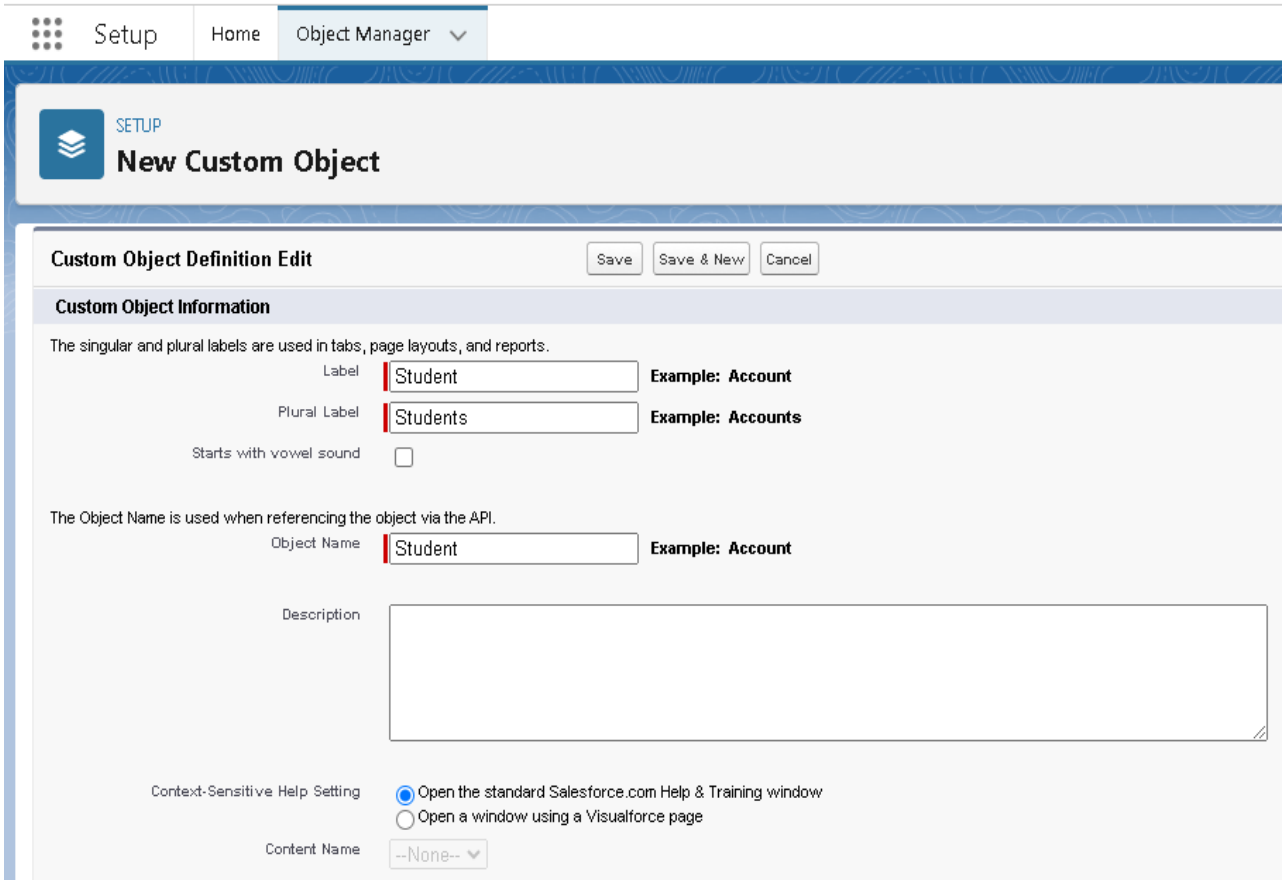
Go to App Launcher and Search Collage Management app and open in New Tab by clicking.



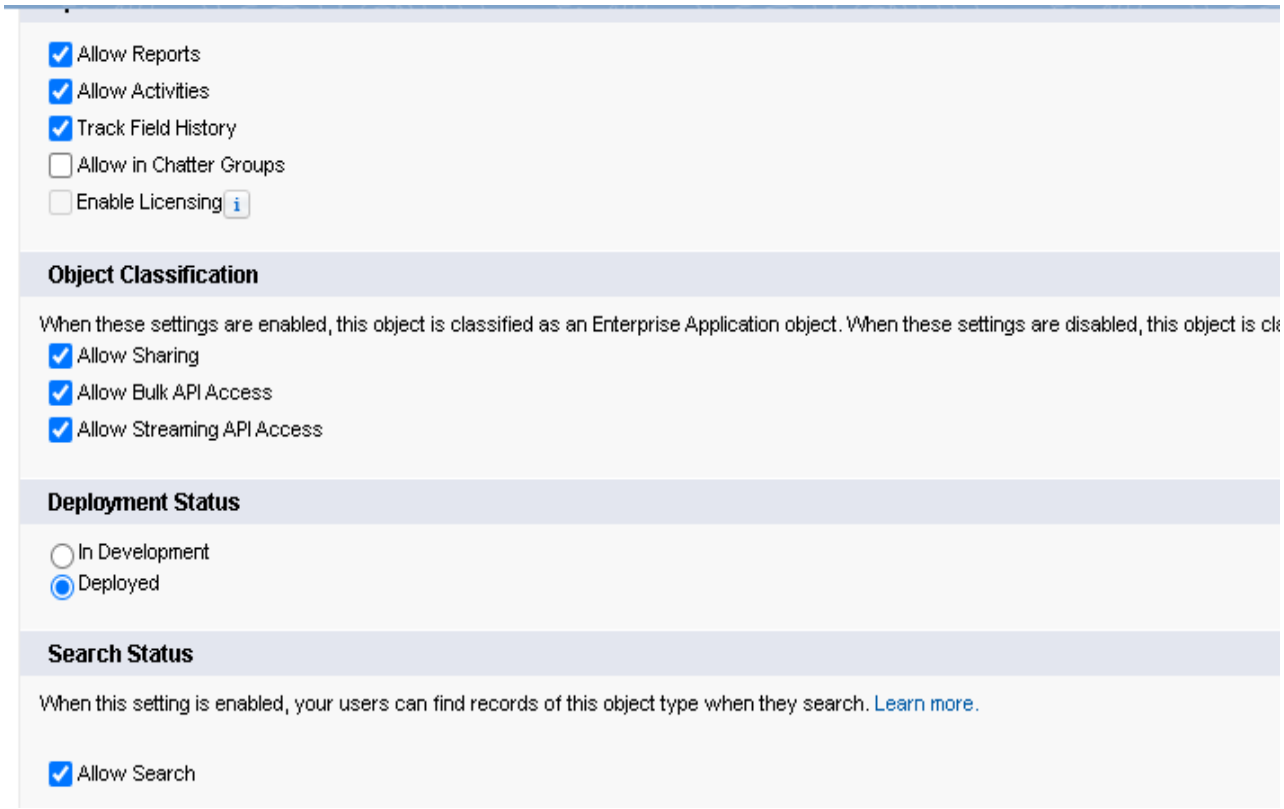
Now Need to add Objects

Add three Custom Objects (Student, Staff, Records)





1



2

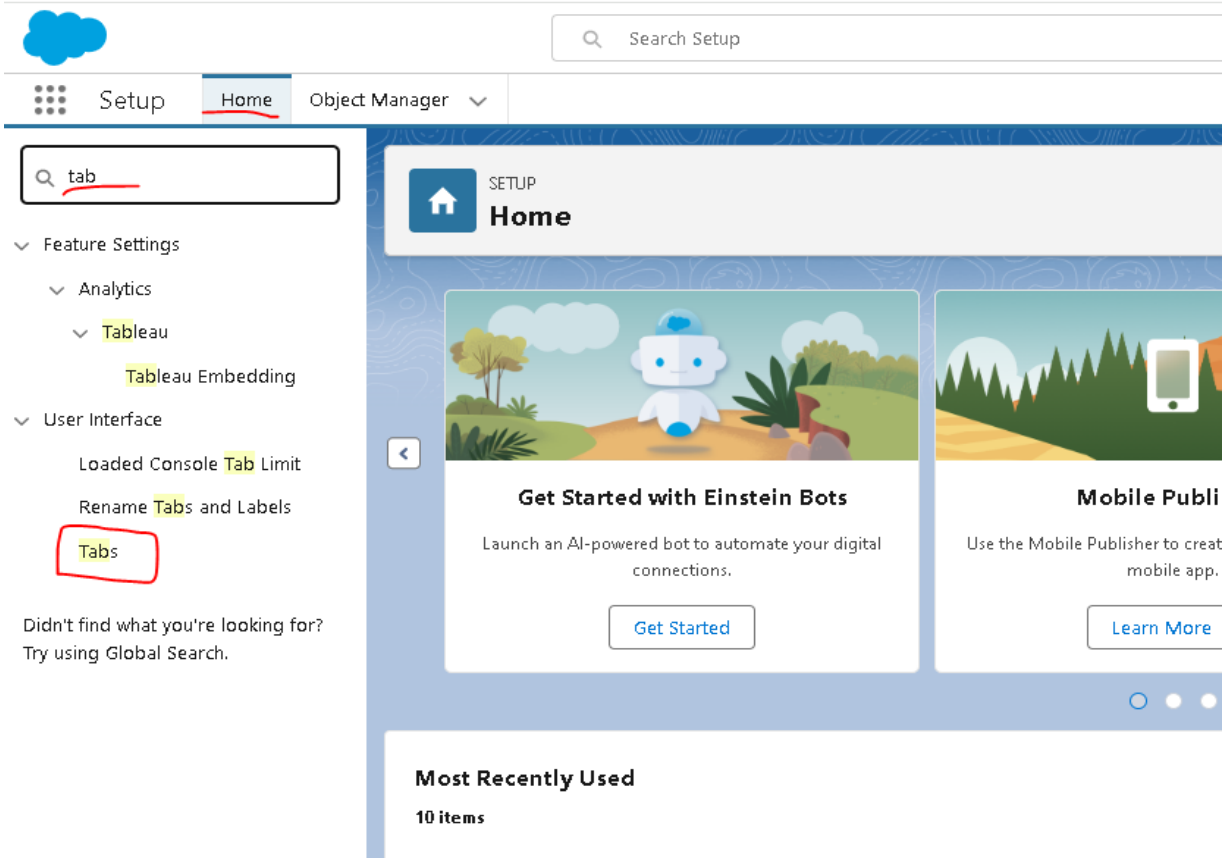
3

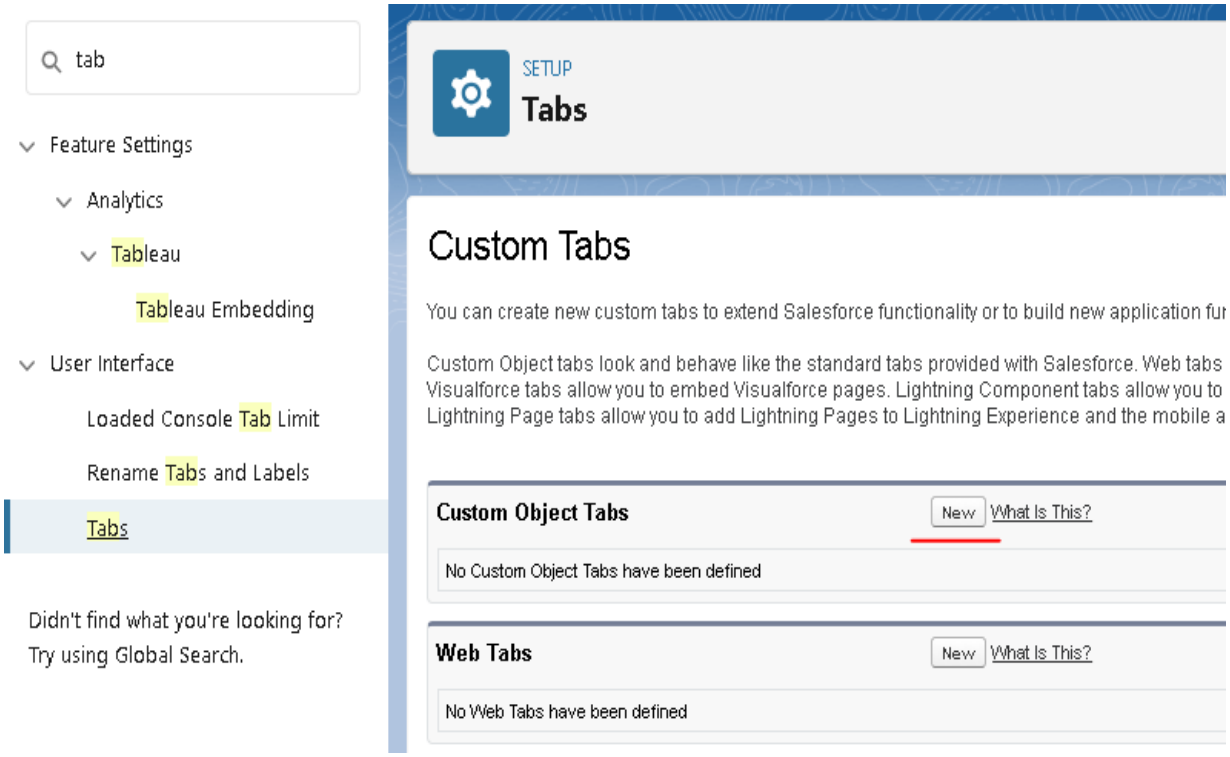
4 **Click Save and New and add Staff and Record with same way**

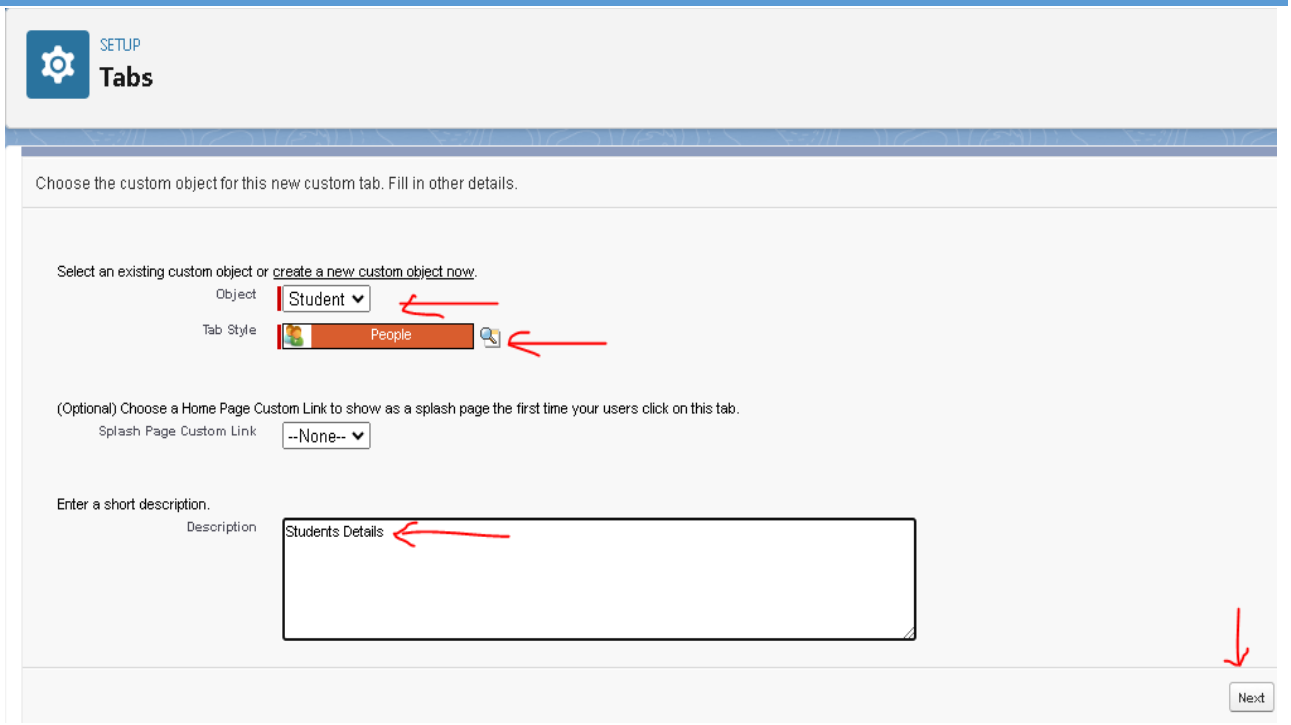
5

Step3

Add Tab for Student, Staff and Records using below Steps

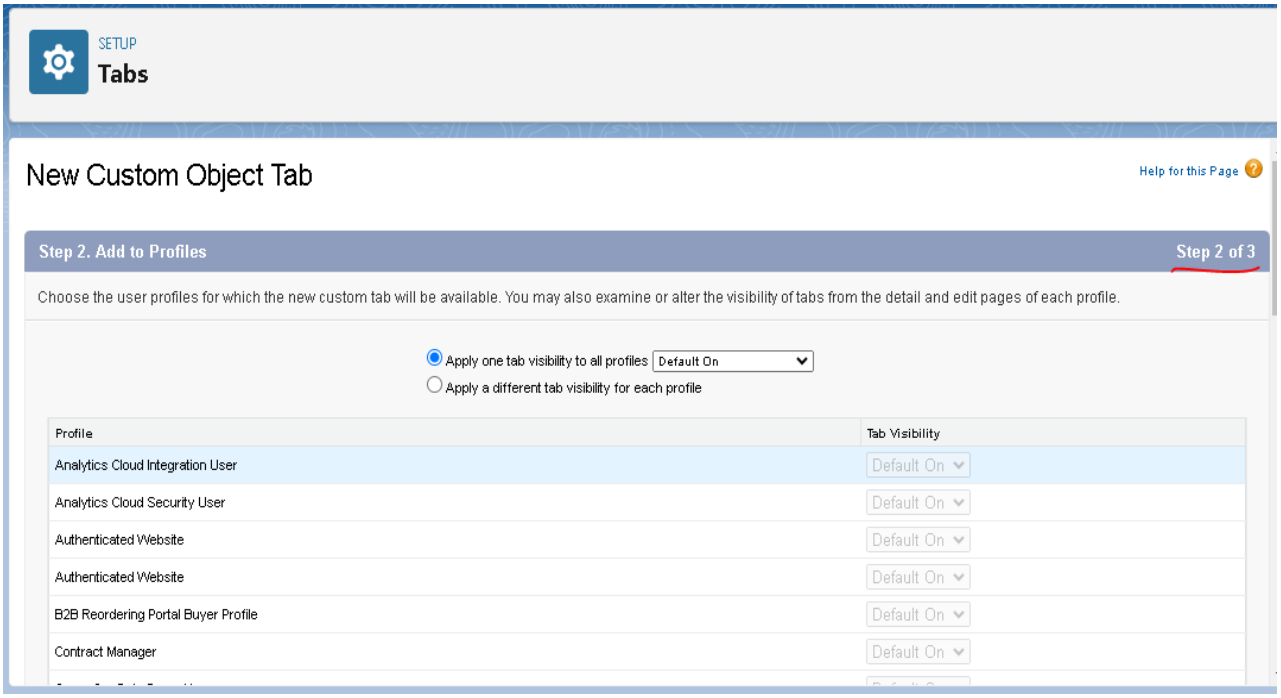





SETUP
Tabs
Choose the custom object for this new custom tab. Fill in other details.
Select an existing custom object or create a new custom object now.
Object
Student
Tab Style
People
(Optional) Choose a Home Page Custom Link to show as a splash page the first time your users click on this tab.
Splash Page Custom Link
--None--
Enter a short description.
Description
Students Details
Next

1

2


SETUP
Tabs
New Custom Object Tab
Help for this Page
Step 2. Add to Profiles
Step 2 of 3
Choose the user profiles for which the new custom tab will be available. You may also examine or alter the visibility of tabs from the detail and edit pages of each profile.
Apply one tab visibility to all profiles
Default On
Apply a different tab visibility for each profile
Profile
Tab Visibility
Analytics Cloud Integration User
Analytics Cloud Security User
Authenticated Website
Authenticated Website
B2B Reordering Portal Buyer Profile
Contract Manager
Default On

3

4

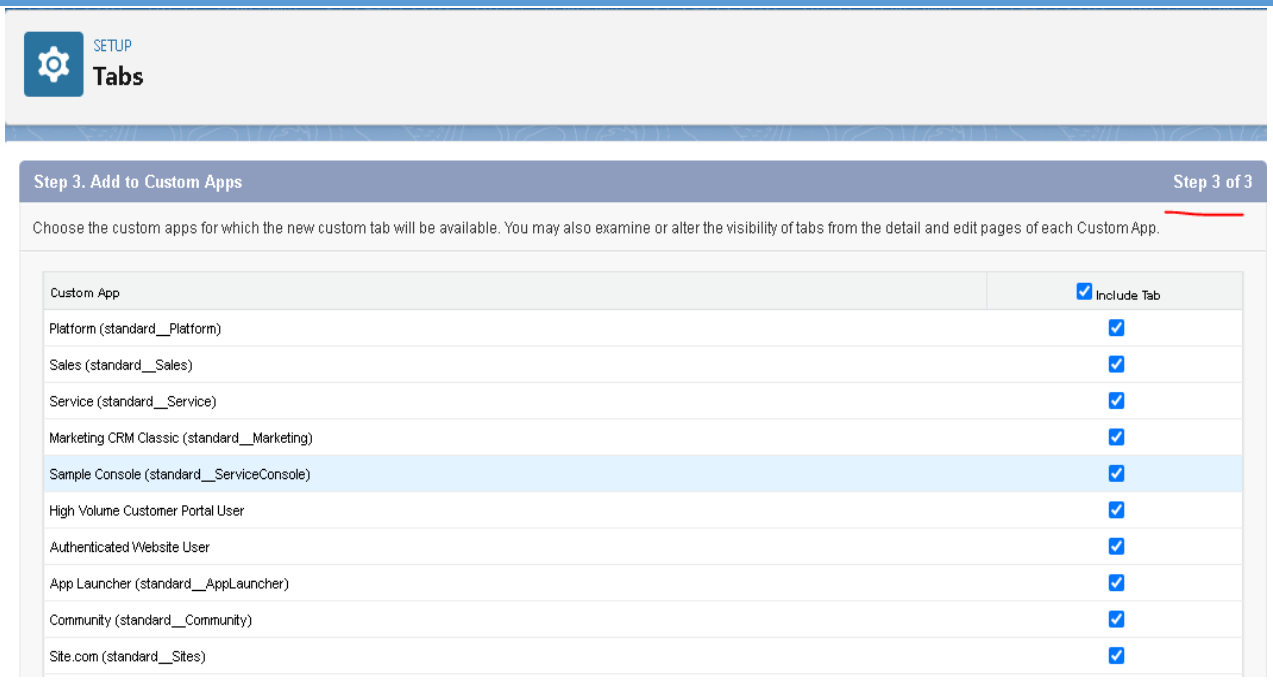
5

6

7

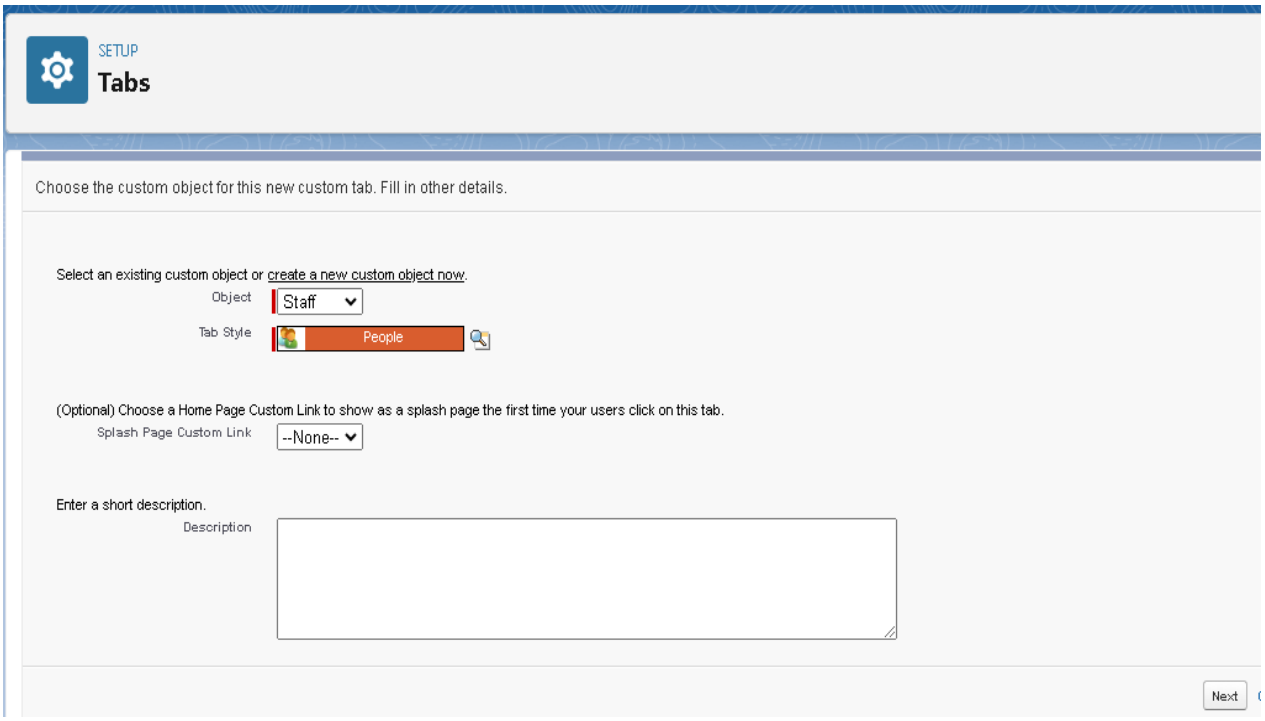
8

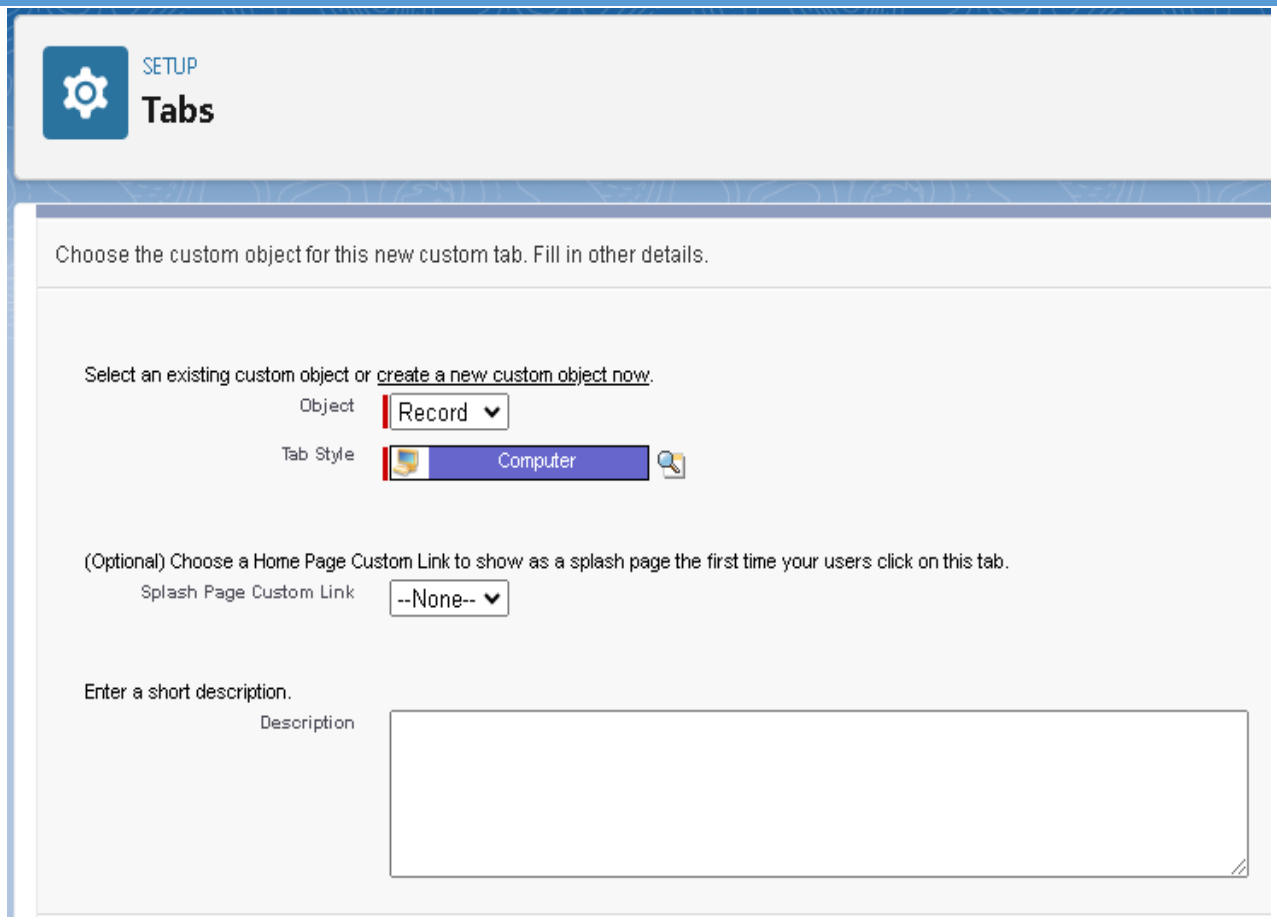
9



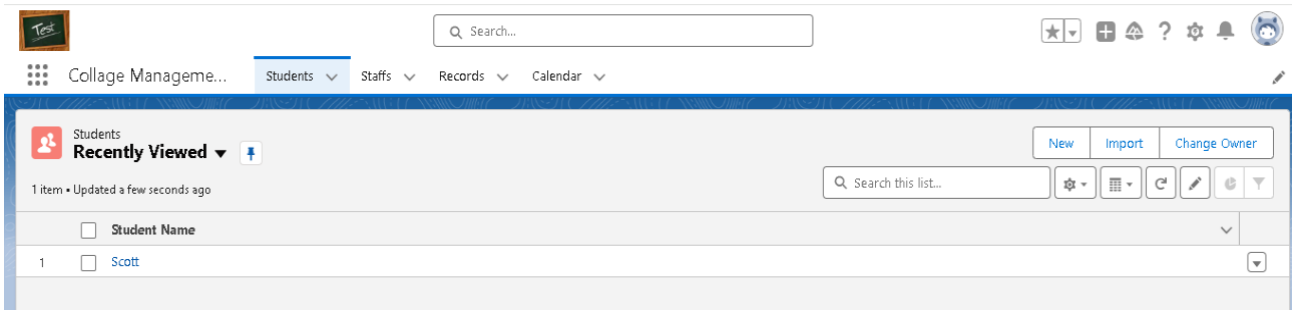
Click on the Save

Add Staff and Records in same way.



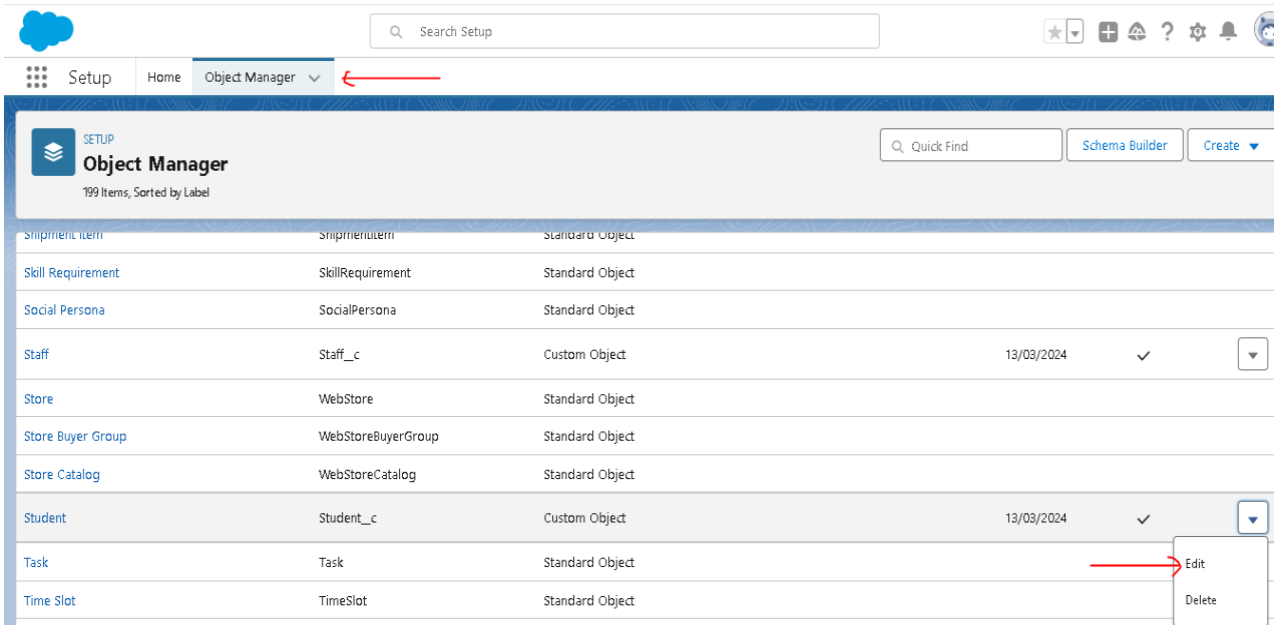


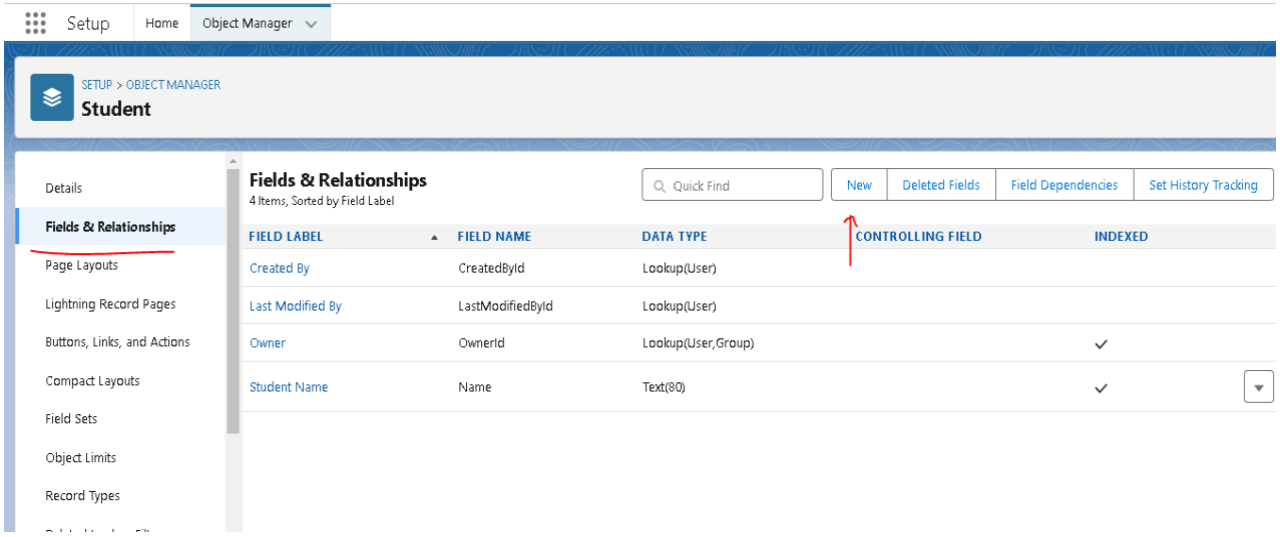
Refresh Collage Management Tab Browser



Step 4

Add Fields and Relationship >> Go to Object Manager >>





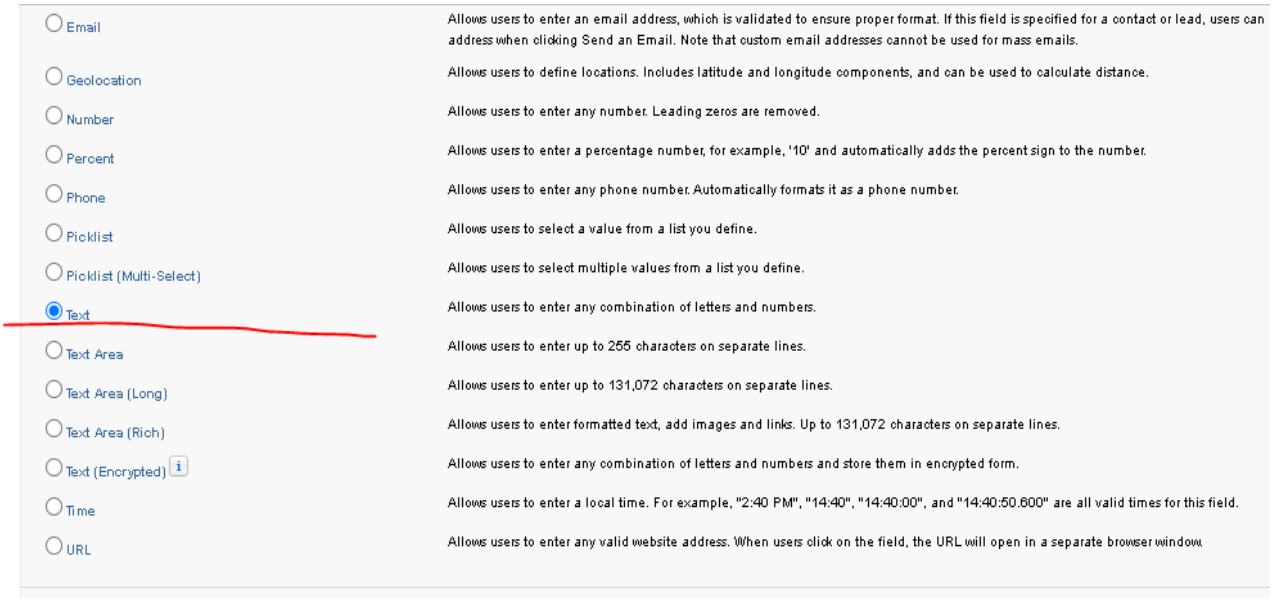
Select the Text for Student Name ,

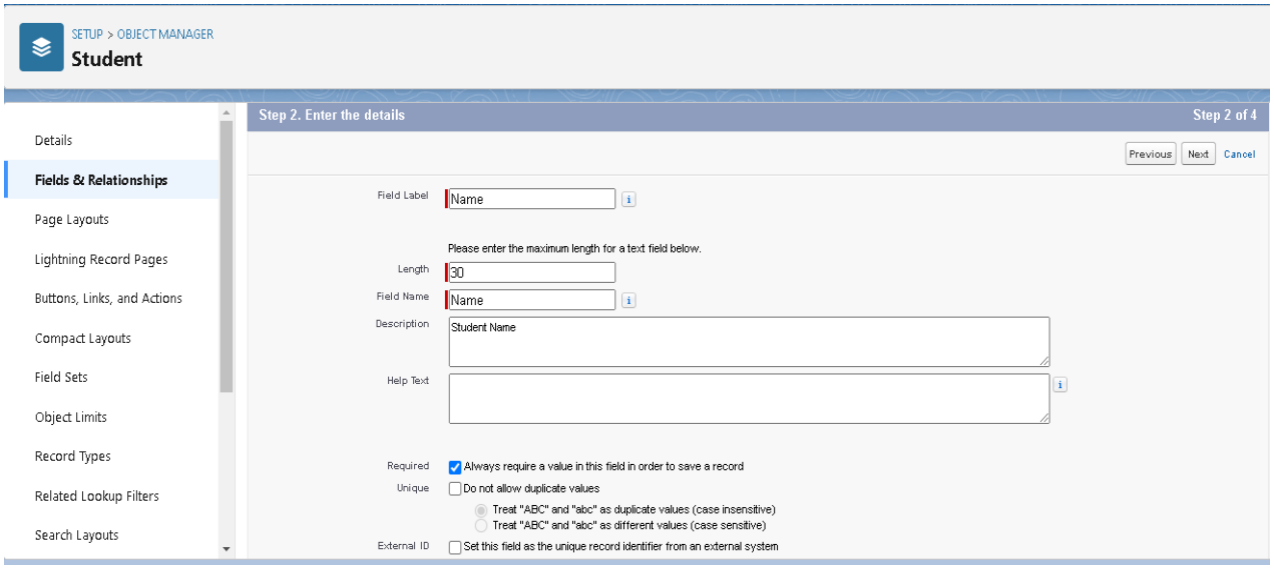
Number for StudentID,

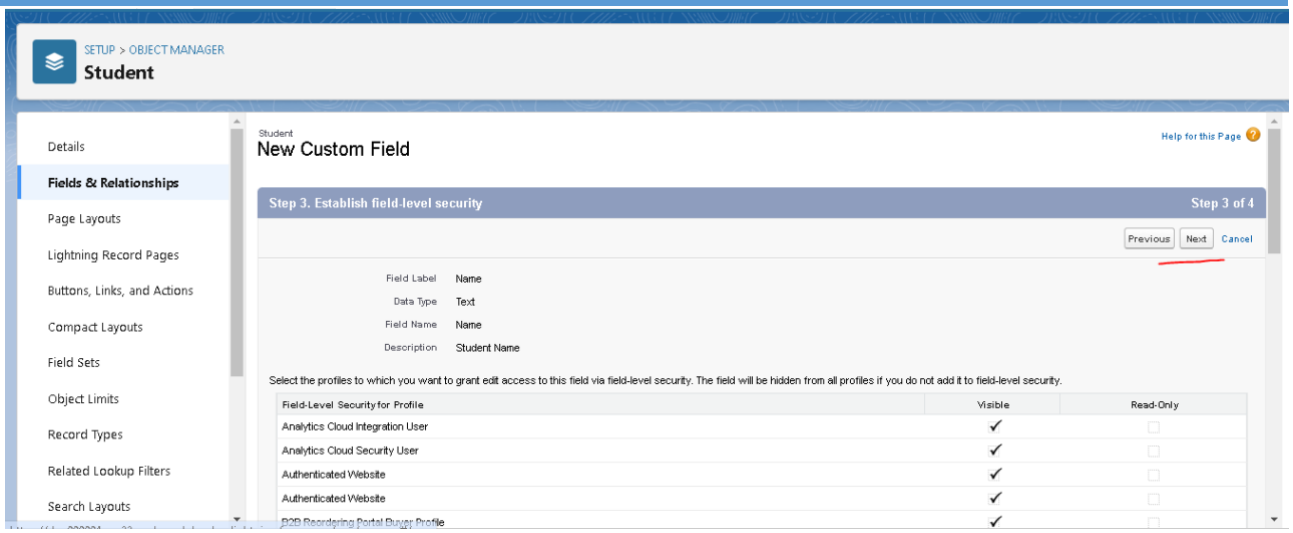
Number for Age,

Date for birthdate,

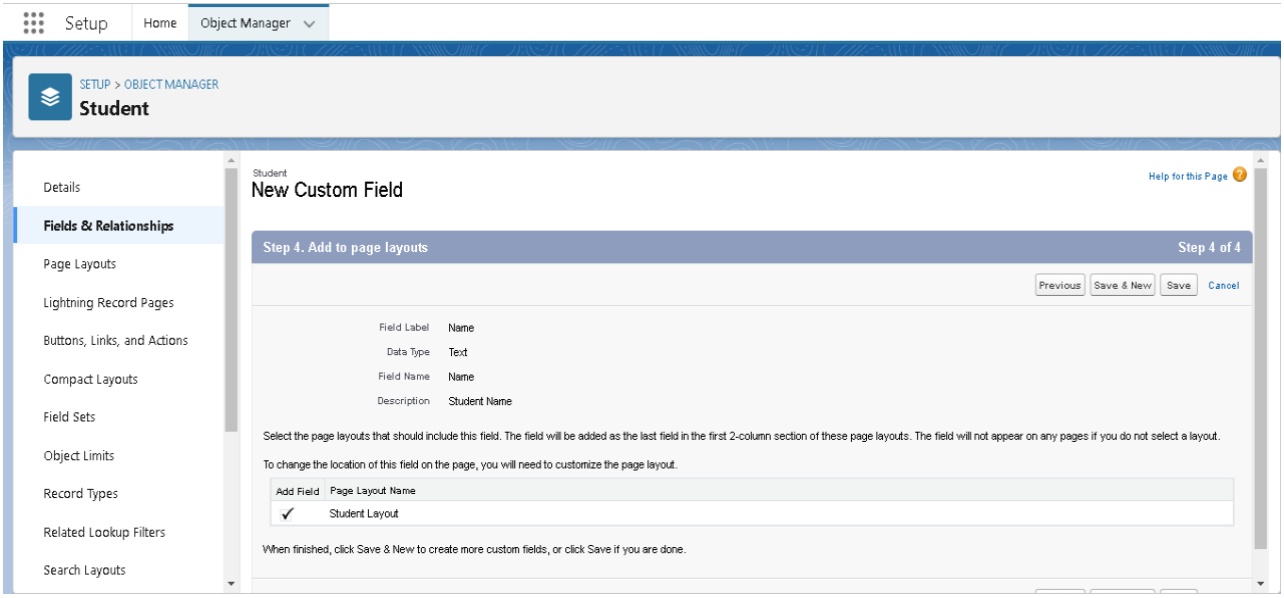
PickUp List for Course







Save



Add for Staff and Records As well.

Staff

Name (Text)

Staff_ID (Number)

Age (Number)

BirthDate (Date)

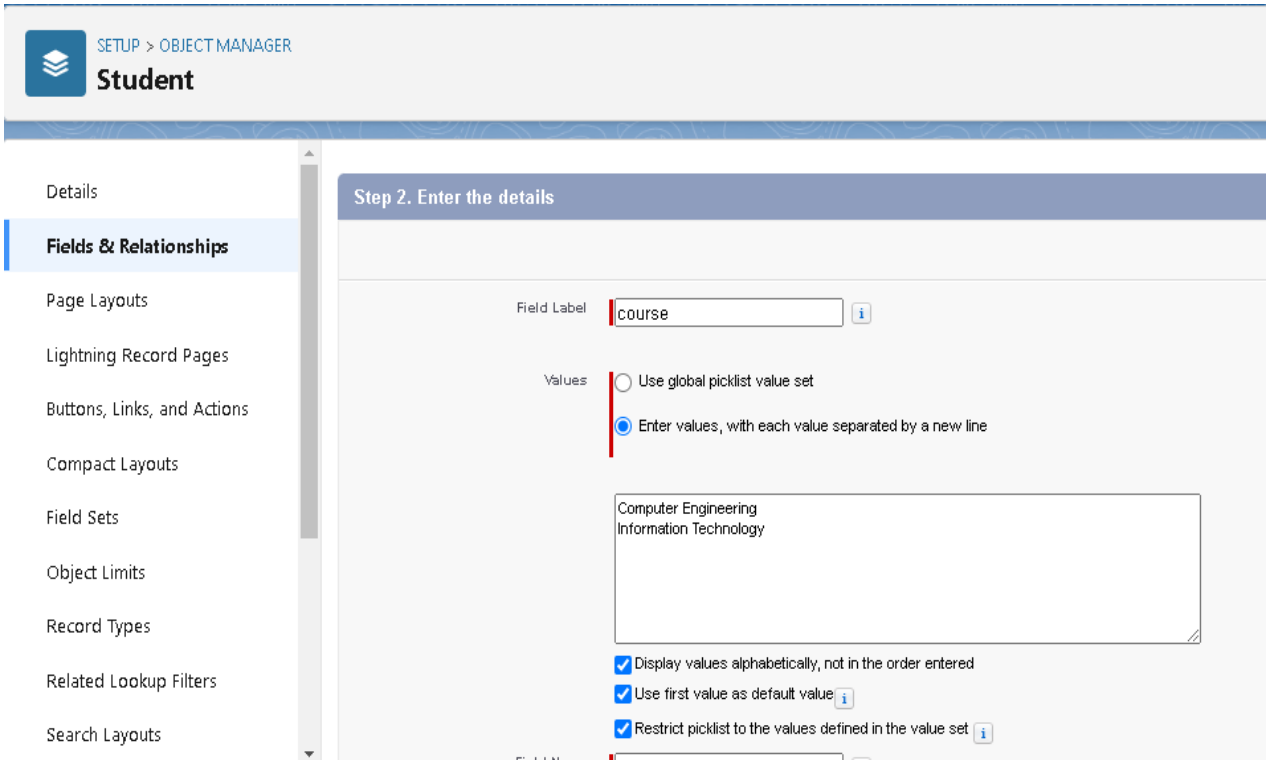
Subject (Pickup List)

Records

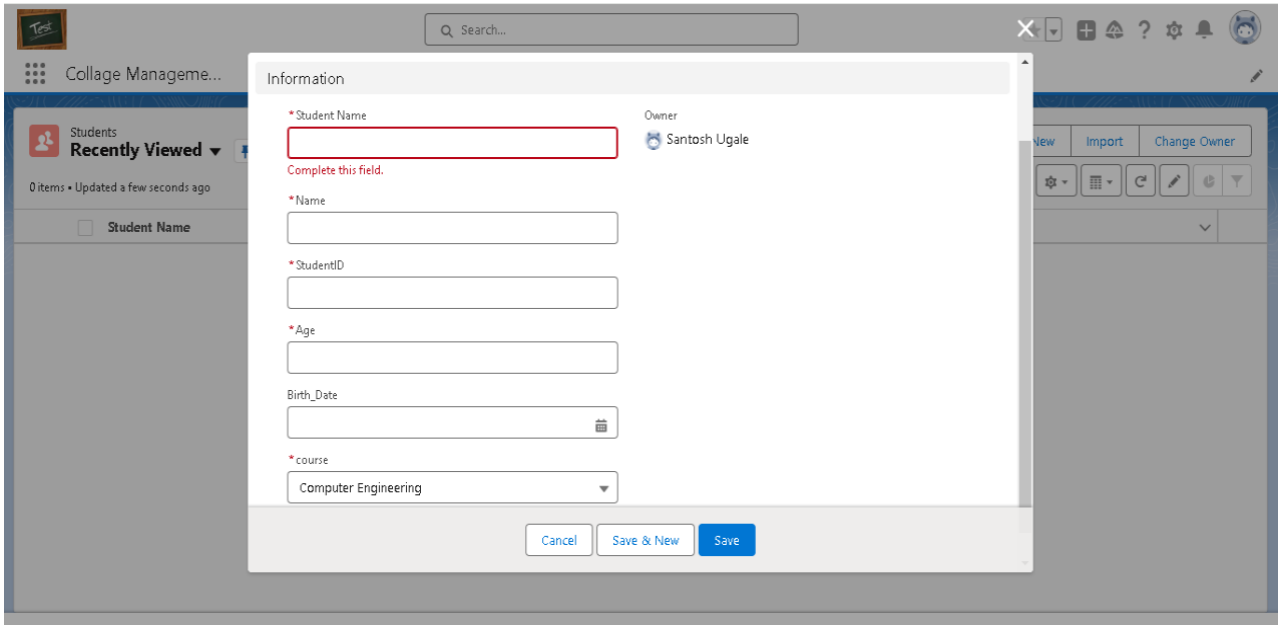
Fee (Number)

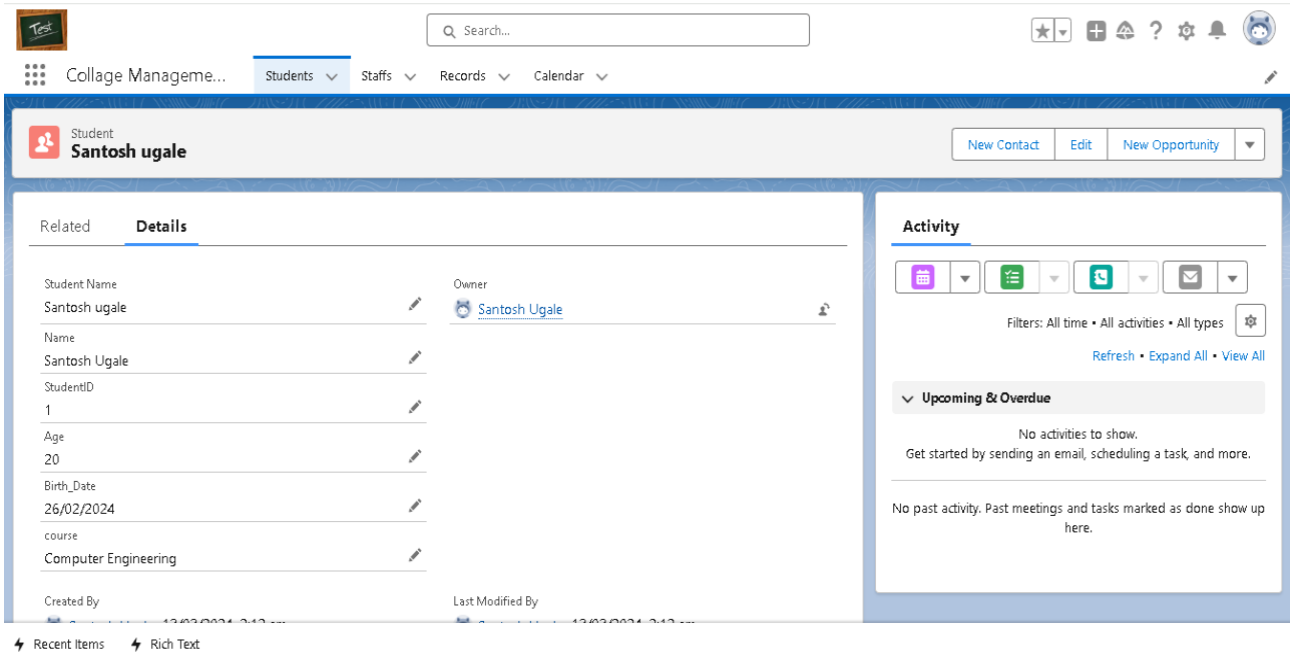
Grades (Pickup List)

For Pickup List



Step5 Go to Collage Nanomagnet App and add records.




Search...
Collage Manageme...
Students
Staffs
Records
Calendar
Student
Santosh ugale
New Contact
Edit
New Opportunity
Related
Details
Student Name
Santosh ugale
Owner
Santosh Ugale
Name
Santosh Ugale
StudentID
1
Age
20
Birth_Date
26/02/2024
course
Computer Engineering
Created By
Last Modified By
Activity
Filters: All time • All activities • All types
Refresh • Expand All • View All
Upcoming & Overdue
No activities to show.
Get started by sending an email, scheduling a task, and more.
No past activity. Past meetings and tasks marked as done show up here.
Recent Items
Rich Text

Reference Only.

cd /opt/

wget IP/jdk-8u151-linux-x64.tar.gz

tar xzf jdk-8u151-linux-x64.tar.gz

cd /opt/jdk1.8.0_151/

alternatives --install /usr/bin/java java /opt/jdk1.8.0_151/bin/java 2

alternatives --config java

alternatives --install /usr/bin/jar jar /opt/jdk1.8.0_151/bin/jar 2

alternatives --install /usr/bin/javac javac /opt/jdk1.8.0_151/bin/javac 2

alternatives --set jar /opt/jdk1.8.0_151/bin/jar

alternatives --set javac /opt/jdk1.8.0_151/bin/javac

export JAVA_HOME=/opt/jdk1.8.0_151

export JRE_HOME=/opt/jdk1.8.0_151/jre

export PATH=$PATH:/opt/jdk1.8.0_151/bin:/opt/jdk1.8.0_151/jre/bin

useradd -d /opt/hadoop hadoop

echo "e\$d\$2Drfed#$user" | passwd hadoop --stdin

su - hadoop

ssh-keygen -t rsa

cat /opt/hadoop/.ssh/id_rsa.pub >> /opt/hadoop/.ssh/authorized_keys

chmod 0600 /opt/hadoop/.ssh/authorized_keys

wget https://dlcdn.apache.org/hadoop/common/hadoop-3.2.4/hadoop-3.2.4.tar.gz

tar xfz hadoop-3.2.4.tar.gz

mkdir /opt/hadoop/hadoopdata

mkdir /opt/hadoop/hadoopdata/namenode

mkdir /opt/hadoop/hadoopdata/datanode

wget -O .bashrc http://IP/enp

source /opt/hadoop/.bash_profile

source /opt/hadoop/.bashrc

export JAVA_HOME=/opt/jdk1.8.0_151

```
export JRE_HOME=/opt/jdk1.8.0_151/jre
wget -O /opt/hadoop/hadoop-3.2.4/etc/hadoop/hadoop-env.sh http://IP/hadoop-env
wget -O /opt/hadoop/hadoop-3.2.4/etc/hadoop/mapred-site.xml http://IP/mapred-site
wget -O /opt/hadoop/hadoop-3.2.4/etc/hadoop/yarn-site.xml http://IP/yarn-site
wget -O /opt/hadoop/hadoop-3.2.4/etc/hadoop/hdfs-site.xml http://IP/hdfs-site
wget -O /opt/hadoop/hadoop-3.2.4/etc/hadoop/core-site.xml  http://IP/core-site
cd ${HADOOP_HOME}/bin
hadoop namenode -format
cd ${HADOOP_HOME}/sbin/
./start-all.sh
jps
```

HDFS Commands

```
hdfs dfs -mkdir /testlab
hdfs dfs -ls /
hdfs dfs -copyFromLocal  test /testlab/
hdfs dfs -ls /testlab
hdfs dfs -cat /testlab/test
hdfs dfs -count  /testlab
hdfs dfs -cp  /testlab/test /testlab/test1
hdfs dfs -chmod 777 /testlab/test
hdfs dfs -chown hadoop:hadoop  /testlab/test
hdfs dfs -get /testlab/test text1
hdfs dfs -rm /testlab/test1
hdfs dfs -rm -r /testlab
```

HDFS command with Segment and Encryption

hdfs dfs -Ddfs.block.size=67108864 -copyFromLocal b.txt /testlab

hadoop --daemon stop kms

hadoop --daemon start kms

hadoop key create test1 -size 256

hadoop key create test2 -size 256 -cipher AES/CTR/NoPadding

hdfs dfs -mkdir /input/test3

hdfs crypto -createZone -keyName test3 -path /input/test3

hadoop key list

hdfs crypto -createZone -keyName scott -path /input/scott

hdfs dfs –put d.txt /input/test3