Dr. AMBEDKAR INSTITUTE OF TECHNOLOGY

Near Jnana Bharathi Campus, Bengaluru-560 056.
(An Autonomous Institution, Aided by Government of Karnataka)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Laboratory Report

Submitted in the Partial Fulfilment of the Cloud Programming Laboratory

Course Code: 18CSL77

Submitted by

SHIVANAND KERUR
1DA20CS135

Submitted to

Mr. Praveena M V
Asst.Professor
Dept. of CSE
Dr.AIT

Department of Computer Science & Engineering
2023-24

CONTENTS

SALESFORCE

Salesforce is an American cloud-based software company headquartered in San Francisco, California. It provides customer relationship management (CRM) service and also provides enterprise applications focused on customer service, marketing automation, analytics, and application development.

History

The company was founded on February 3, 1999 by former Oracle executive Marc Benioff, together with Parker Harris, Dave Hoelterhoff, and Frank Dominguez as a software as a service (SaaS) company, andwas launched publicly between September and November 1999.In June 2004, the company had its initial public offering on the New York Stock Exchange under the stock symbol CRM and raised US$110 million. Early investors include Larry Ellison, Magdalena Yesil, Halsey Minor, Stewart Henderson, Mark Iscaro, and Igor Sill, a founding member of Geneva Venture Partners. In October 2014, Salesforce announced the development of its Customer Success Platform to tie together Salesforce's services, including sales, service, marketing, analytics, community, and mobile apps. In October 2017, Salesforce launched a FacebookAnalytics tool for business-to-business marketers. In September 2018, Salesforce partnered with Apple intended on improving apps for businesses. In February 2020, co-chief executive officer Keith Block stepped down from his position in the company. Marc Benioff remained as chairman and chief executive officer. On December 1, 2020, it was announced that Salesforce would acquire Slack form $27.7 billion. The acquisitionclosed on July 21.In February 2021, Salesforce announced that CFO Mark Hawkins would be retiring from his position after six years of working for the company; however, retaining a position as CFO emeritus until October. Amy Weaver was selected as his replacement.

Salesforce.com's customer relationship management (CRM) service comprises several broad categories: Commerce Cloud, Sales Cloud, Service Cloud, Data Cloud (including Jigsaw), Marketing Cloud,Community Cloud (including Chatter), Manufacturing Cloud, Analytics Cloud, App Cloud, Vaccine Cloud,IoT and Work.com with over 100,000 customers.

Main services

Salesforce's main services are tools for case, task and issue management. It also gives customers tracking abilities for their raised cases and conversation features for social networking Web sites, provides analytical tools and other services including email alert, Google search, and access to customers' entitlement and contracts. They also partner with companies like IBM, Accenture, and Saggezza to help integrate Salesforce's cloud-based services into their businesses.

Lightning Platform

Lightning Platform (also known as Force.com) is a platform as a service (PaaS) that allows developersto create add-on applications that integrate into the main Salesforce.com application. [failed verification] These third-party applications are hosted on Salesforce.com's infrastructure.

Force.com applications are built using declarative tools, backed by Lightning [further explanation needed] and Apex, a proprietary Java-like programming language for Force.com, as well as Visualforce, a frameworkincluding an XMLsyntax typically used to generate HTML.

In 2015, a new framework for building user interfaces – Lightning Components – was introduced in beta. Lightning components are built using the open-source Aura Framework but with support for Apex as the server-side language instead of Aura's JavaScript dependency. This has been described as an alternative to, not necessarily a replacement for, Visualforce pages.

As of 2013, the Force.com platform has 1.4 million registered developers. Lightning Base Components is thecomponent library built on top of Lightning Web Components.

Experience Cloud

Experience Cloud (formerly Community Cloud) provides Salesforce customers the ability to create online web properties for external collaboration, customer service, channel sales, and other custom portals intheir instance of Salesforce. Tightly integrated to Sales Cloud, Service Cloud, and App Cloud, Experience Cloud can be quickly customized to provide a wide variety of web properties. Experience Cloud combines thefunctionality of the former Salesforce Customer and Partner Portals with some additional features.

Work.com

Work.com, previously Rypple, is a social performance management platform for managers and employees. It allows continuous coaching, real-time feedback, and recognition. It is aimed at sales management, customer service, marketing, and can be utilized by human resource departments.

Work.com, then known as "Rypple", was founded by Daniel Debow and David Stein, to create a simple wayof asking for feedback anonymously at work. The company was formed in May 2008 and their client list included Mozilla, Facebook, LinkedIn and the Gilt Groupe. Rypple aims to get employees to build and managetheir own coaching networks.

In September 2011, Rypple announced that they had hired Bohdan Zabawskyj as its Chief Technology Officer. In 2011, Rypple developed a more formalized management methodology called OKR ("Objectives and Key Results") for Spotify. Rypple also partnered with Facebook to create "Loops", short for "feedback loops", which gathers feedback from co-workers, including praise, progress against goals, and coaching fromsupervisors into one channel.

In December 2011, Salesforce.com announced that they would acquire Rypple. The transaction was completedin 2012 and Rypple was rebranded as Work.com in September 2012.

AppExchange

Launched in 2005, the Salesforce AppExchange is an online application marketplace for third-party applications that run on the Force.com platform. Applications are available for free, as well as via yearly or monthly subscription models. Applications available range from integrations with SharePoint to mobile approval management. As of June 2016, it features 2,948 applications which have driven 3+ million installs. The "AppExchange" is also a place customers can search for cloud consulting partners to help them implementthe technology in their own organization.

MyTrailhead

Launched in 2019, Salesforce's myTrailhead is an online training platform that can be customized forthe specific needs of its customers. The platform extends functionality to provide users with training content specific to their usage of Salesforce and enables them to create and publish their own training content and programs.

Technologies

Salesforce is powered by the Model–view–controller architecture.

Apex

Apex is a proprietary programming language provided by the Force.com platform to developers similar
to Java and C#. It is a strongly typed, object-oriented, case- insensitive programming language, following a dot-notation and curly-brackets syntax. Apex can be used to execute programmed functions during most processes on the Force.com platform including custom buttons and links, event handlers on record insertion, update, or deletion, via scheduling, or via the custom controllers of Visualforce or Lightning Experience pages.

Due to the multitenant nature of the platform, the language has strictly imposed governor limitations to guard against any code monopolizing shared resources. Salesforce provides a series of asynchronous processing methods for Apex to allow developers to produce longer-running and more complex Apex code.

Lightning

In 2014, Salesforce made public the front end of its platform, called Lightning. This component-basedframework is what the Salesforce mobile app is built on. Salesforce built on this framework in 2015 by releasing the Lightning Design System, an HTML style framework with default CSS styling built in. This framework allows customers to build their own components to either use in their internal instances or sell onthe AppExchange.

Lightning Experience, released in 2016, is the new redesigned interface in Salesforce for processes enhancement. Since then all the apps available on AppExchange need to be Lightning and those built on Classic have to migrate to Lightning as Classic is not to be updated any more by Salesforce. The platform offers an option for developers to employ migration techniques to enable the new user-friendly interface andswitch to Lightning.

What is Salesforce used for?

- Engage customers with relevant, empathetic digital marketing from anywhere.
- Sell smarter and grow your business faster from anywhere.
- Quickly launch and scale ecommerce built around your customer — from anywhere.
- Provide great customer service from anywhere.

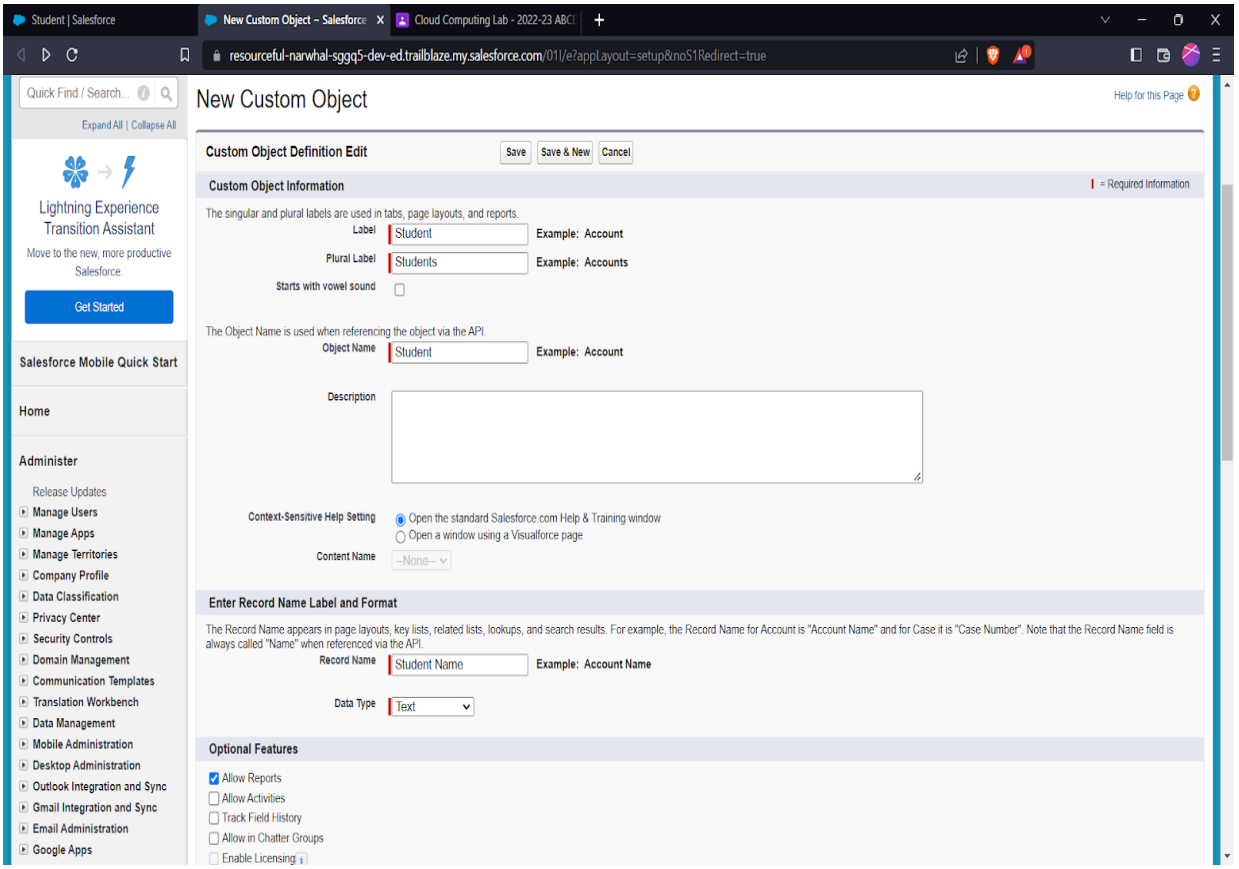
.

Part A - 1A - Creation of web applications on Salesforce Cloud Platform

Q1) Create a web application to enter student details like Name, USN, Semester,Section and CGPA to a database on Salesforce Cloud Platform

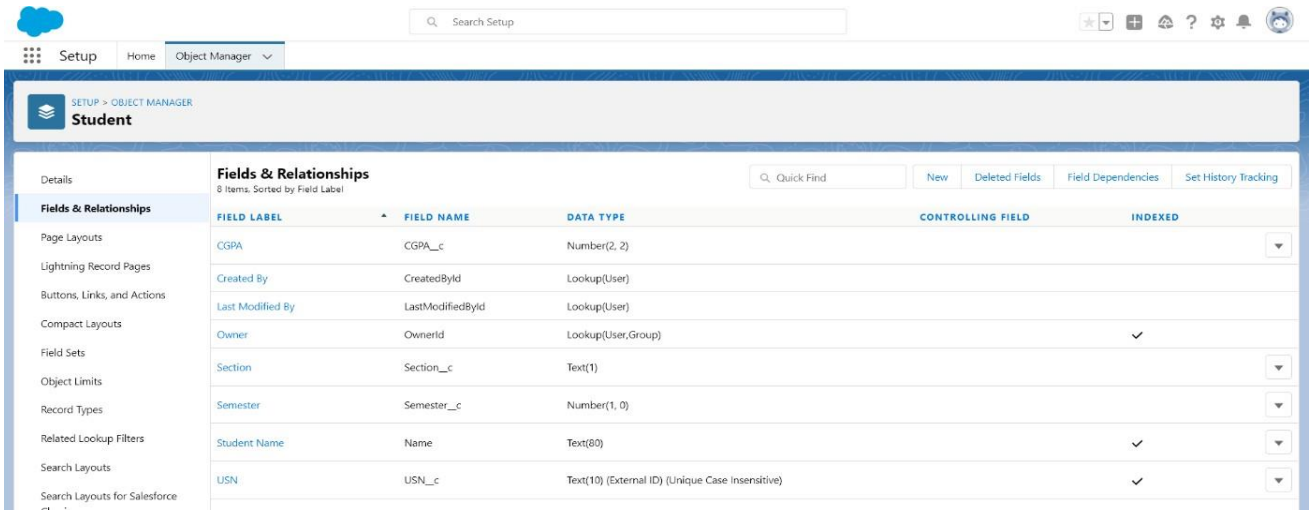
1. Launch your Salesforce Trailhead Playground by opening any module and Switch to Lightning Experience if you are currently in Salesforce Classic by clicking your picture in the right top cornerand then click on “Switch to Lightning Experience”.
2. Then go to Setup gear icon and click “Setup”.
3. Click on “Object Manager” and click “Create> Custom Object” to create new Custom Object.
4. Name the object “Student”.
5. Allow Reports and Allow Search.
6. Check the box in front of “Launch New Custom Tab Wizard after saving this custom object”.
7. To create a Tab for the Object: Select any Tab Style for the object “Student”. Click Next, Next, leavethe defaults and save.

   To add fields to the Object: Go to “Fields & Relationships” option of Student object and Click “New”.



Add the following fields one after the other:

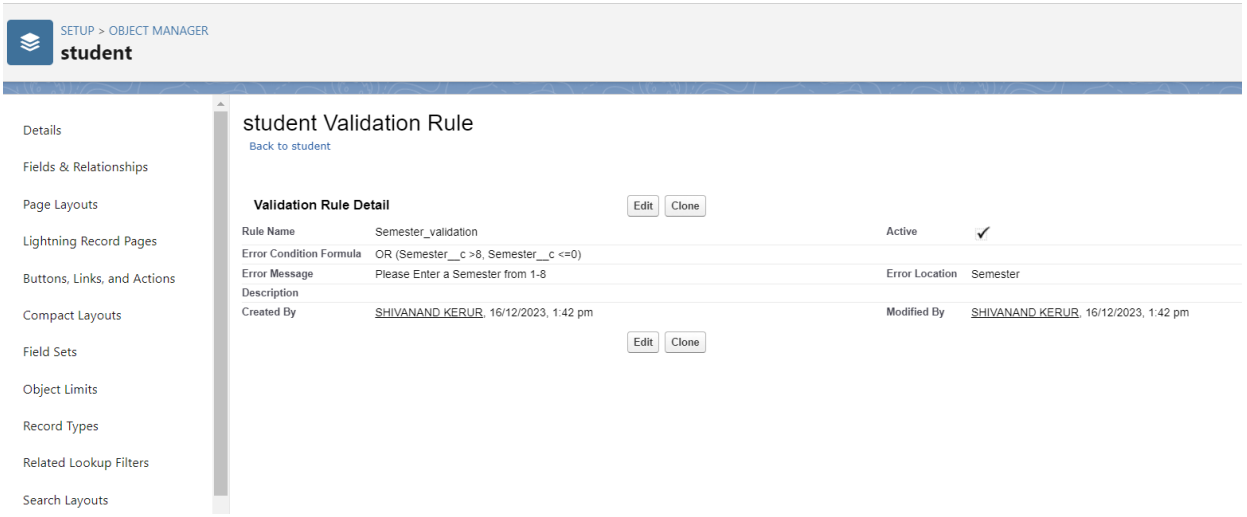
1. Field Label: USN (Length 10), Data Type: Text, provide an example USN as Help Text, make itas Required Field and Don't allow Duplicate Values and make it as Case Insensitive.
2. Field Label: Section (Length 1), Data Type: Text, make it as Required Field.
3. Field Label: Semester (Length 1, Decimal Place 0), Data Type: Number, make it as RequiredField.
4. Field Label: CGPA (Length 2, Decimal Places 2), Data Type: Number, make it as Required Field.



Semester Validation:

To add a rule to the Semester so that it should always be greater than 0 and less than or equal to 8:

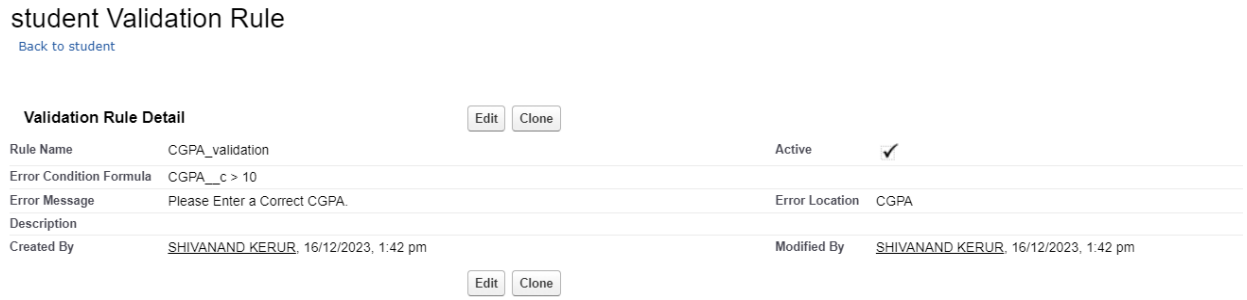
1. Go to Validation Rule of Student Object and click "New".
2. Name it as "Semester validation".
3. Error Condition Formula: OR (Semester__c >8, Semester__c <=0).
4. Error Message: Please Enter a Semester from 1-8.
5. Error Location: Field – Semester.
6. Click Save.



CGPA Validation:

To add a rule to the CGPA so that it should not take CGPA greater than 10:

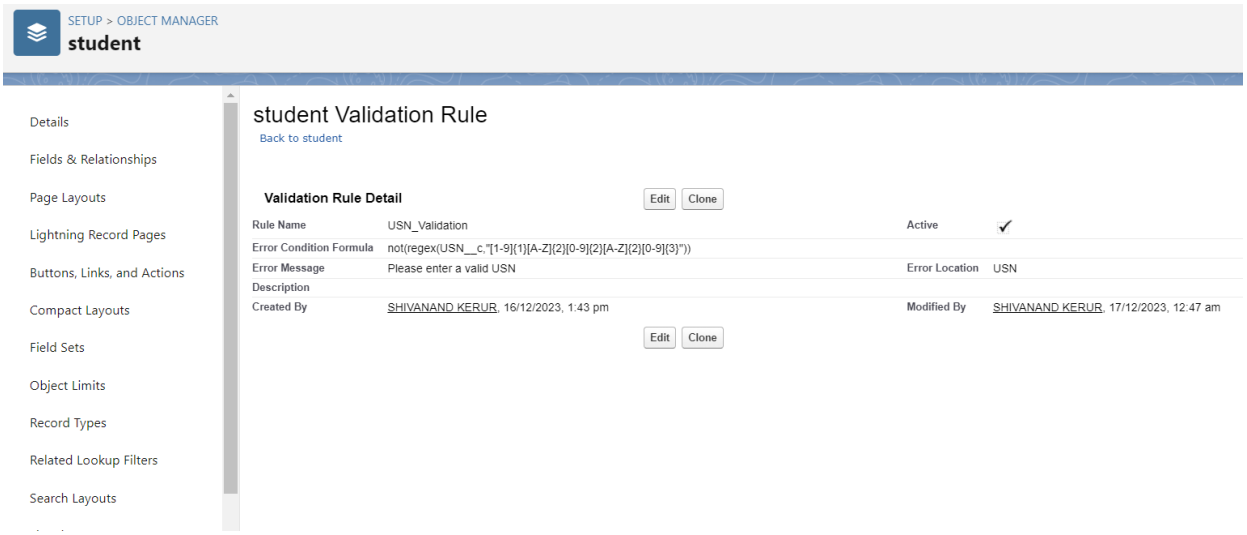
1. Go to Validation Rule of Student Object and click “New”
2. Name it as “CGPA validation”.
3. Error Condition Formula: CGPA__c > 10
4. Error Message: Please Enter a Correct CGPA.
5. Error Location: Field –CGPA
6. Click Save



USN Validation:

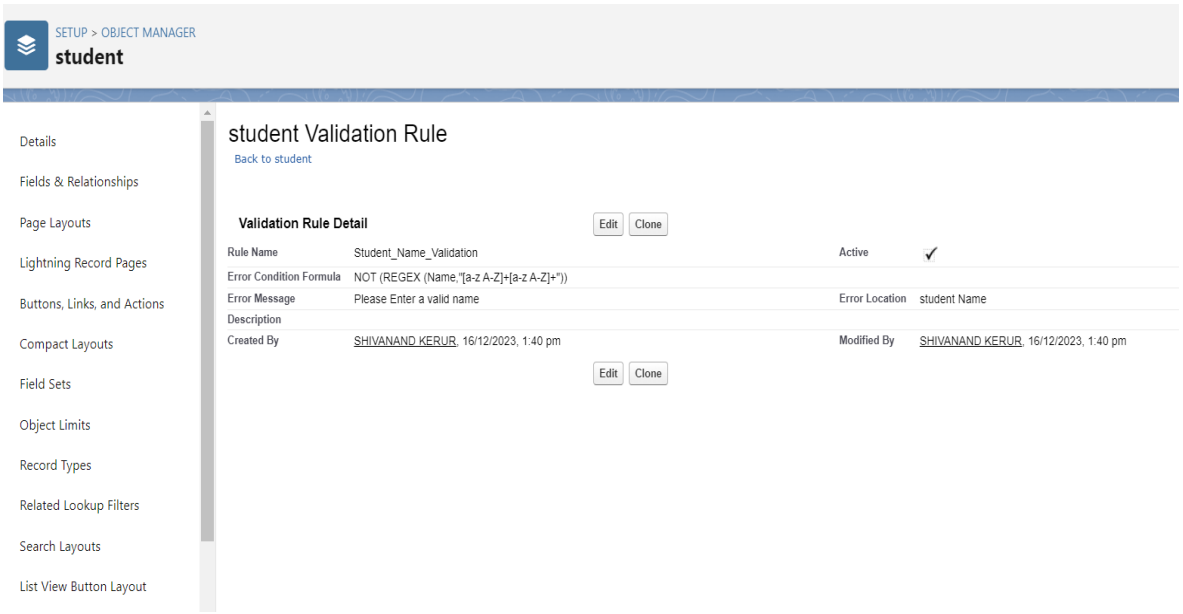
To add a rule to the USN so that it should validate only student’s serial number.

1. Go to Validation Rule of Flight Object and click “New”
2. Name it as “USN Validation”.
3. Error Condition Formula: NOT(REGEX(USN__c,"[1-5]{1} +[A-Z]{2} + [0-9]{2} +[A-Z]{2} + [0-9]{3}"))
4. Error Message: Please enter a valid USN.
5. Error Location: Field – USN
6. Click Save.



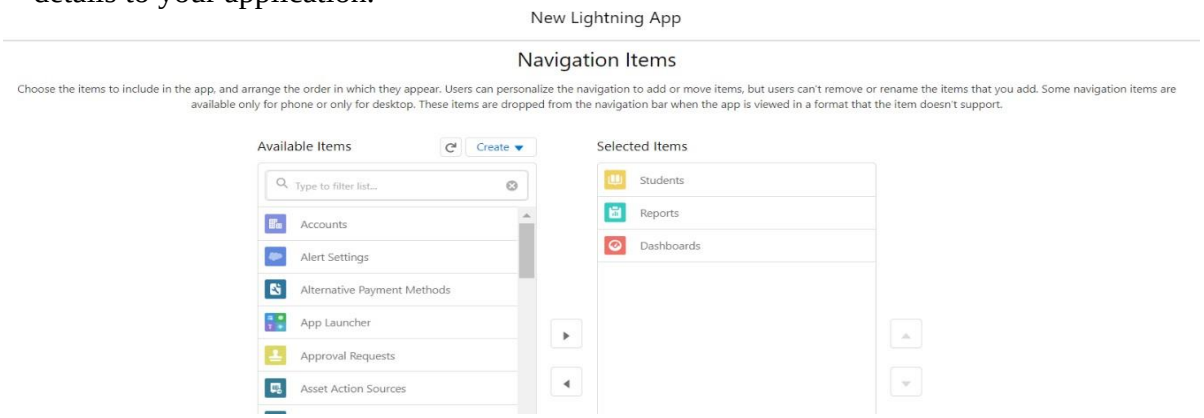
Name validation:

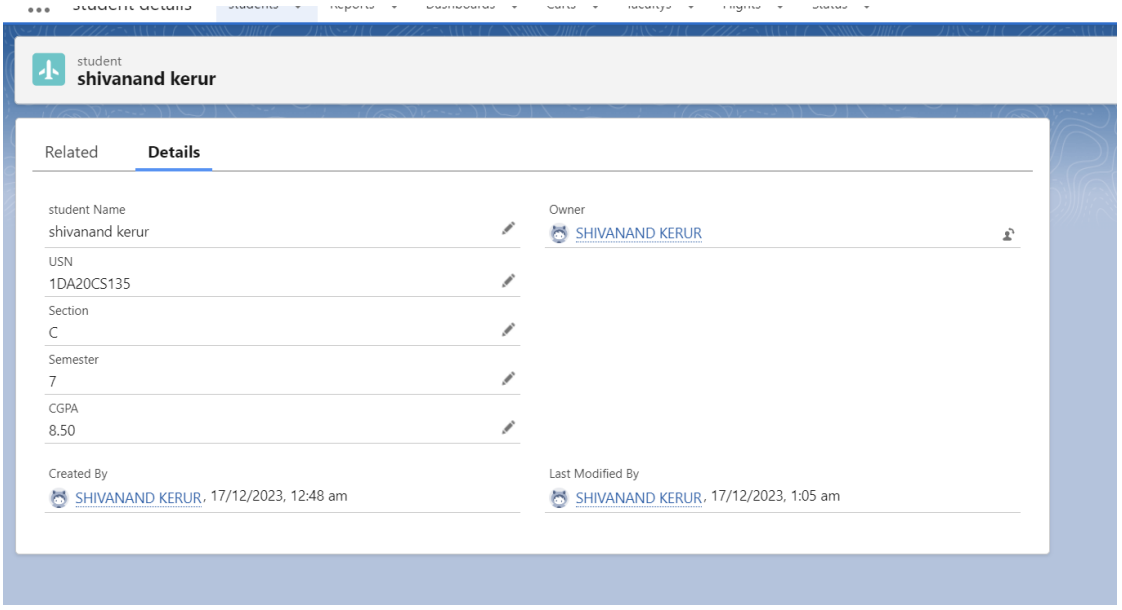
1. To add a rule to the Student Name so that the name should only start with letter but not digit:
2. Go to Validation Rule of Student Object and click “New”
3. Name it as “Student Name Validation”
4. Error Condition Formula: “NOT (REGEX (Name,"[a-z A-Z]+[a-z A-Z]+"))”
5. Error Message: “Please Enter a valid name”.
6. Error Location: Field –Student Name.
7. Click Save.

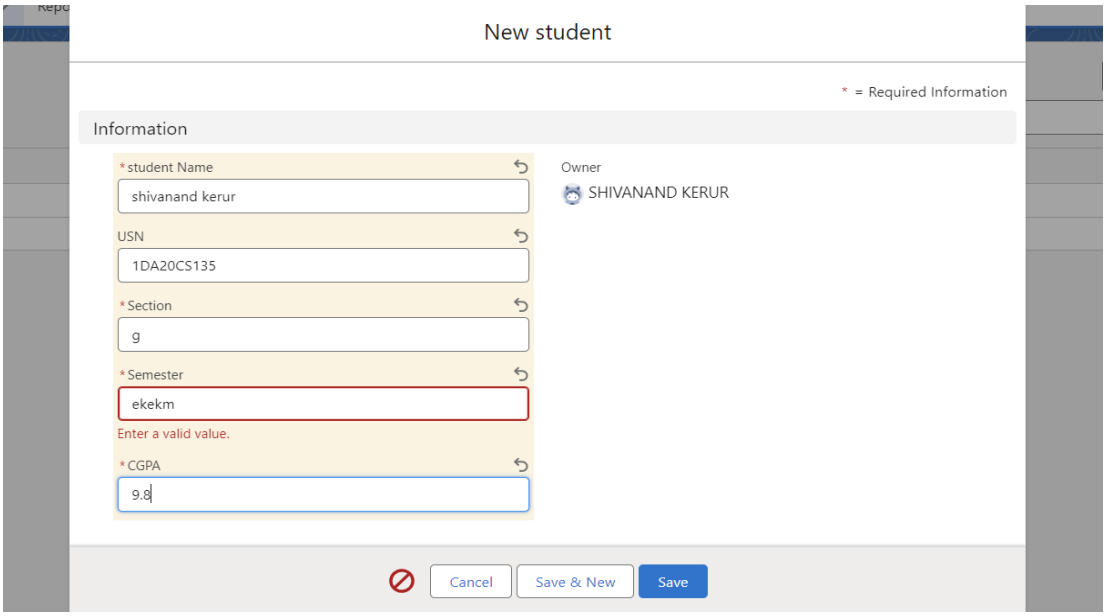


To create an application:

1. Go to “Setup” and type “App Manager” in Quick Find Box.
2. Click on “New Lightning App” to create a Lightning Application.
3. Name it as “Student Details”, give the description for your application.
4. Uploading Image and changing colours are optional, then click Next.
5. Navigation Style: Standard Navigation, click Next.
6. No need to add any Utility Bar, click Next.
7. Add the following Items: Students, Reports and Dashboards, click Next.
8. Assign it to System Administrator Profile by selecting System Administrator and pressing right arrow and then click Save & Finish.
9. Go to App Manager, select your application and select Students and click “New” to add some details to your application.







Make sure you will get error messages when you give invalid Name, USN, Semester and CGPA.

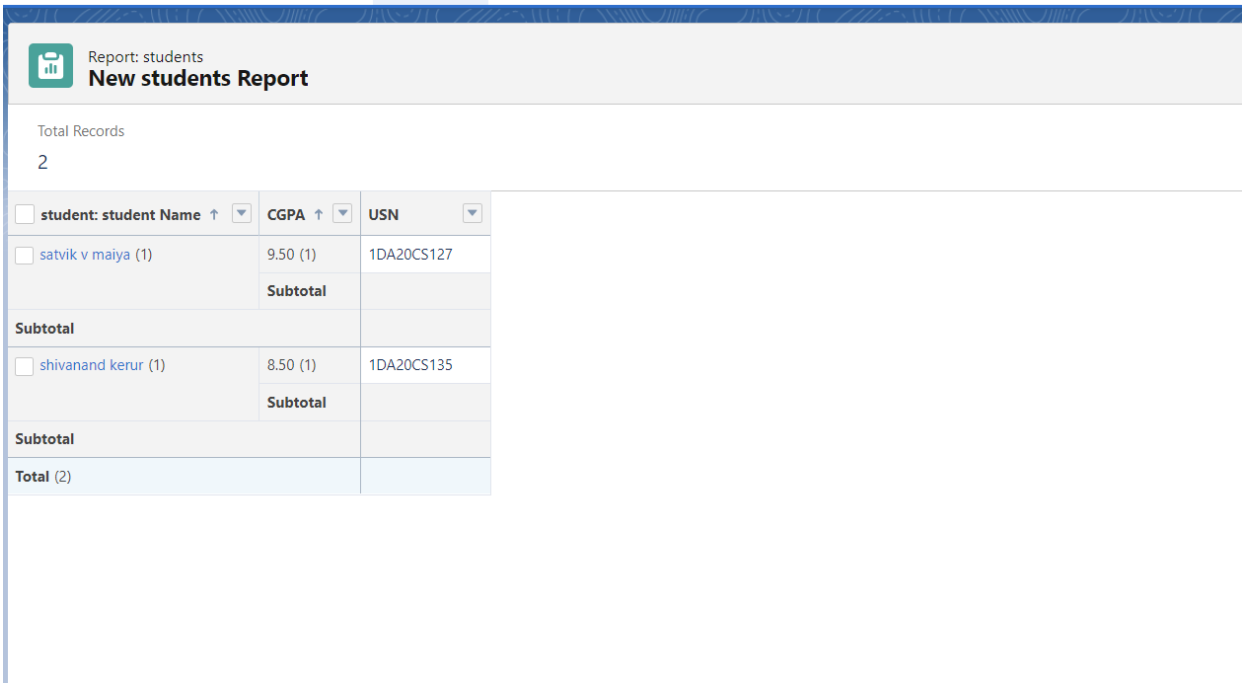
Reports and Dashboards:

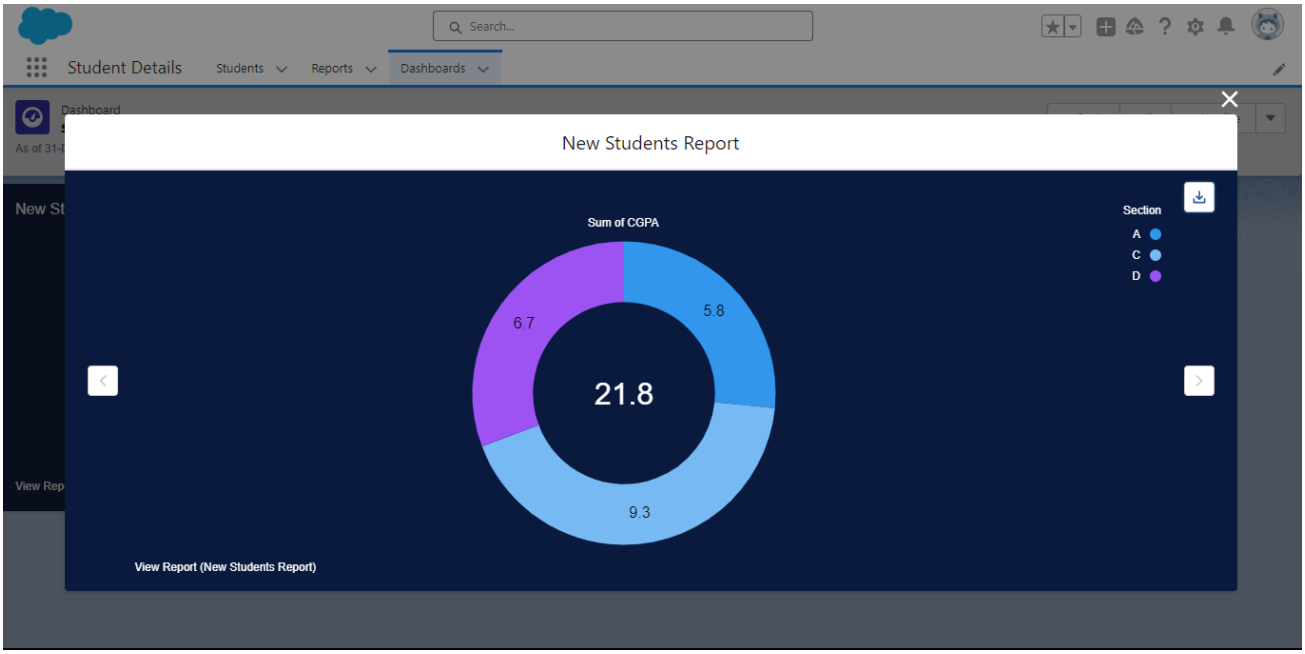
To Create a Students Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name and Click Save.
2. Click on “New Report” and from search bar Search for “Students” and then select it and then clickcontinue.
3. Add the required Columns to get the Completed Entered data.
4. If you Want the report to be grouped by any specific Fields then Search for the field in “Add groups”otherwise it is optional.
5. Click on save and name the report as “New Students Report” and then select the folder which youhave created.
6. Click Save and then Click Run

To Create a Students Dashboard:

1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Students Dashboard” and select folder that youhave created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run.



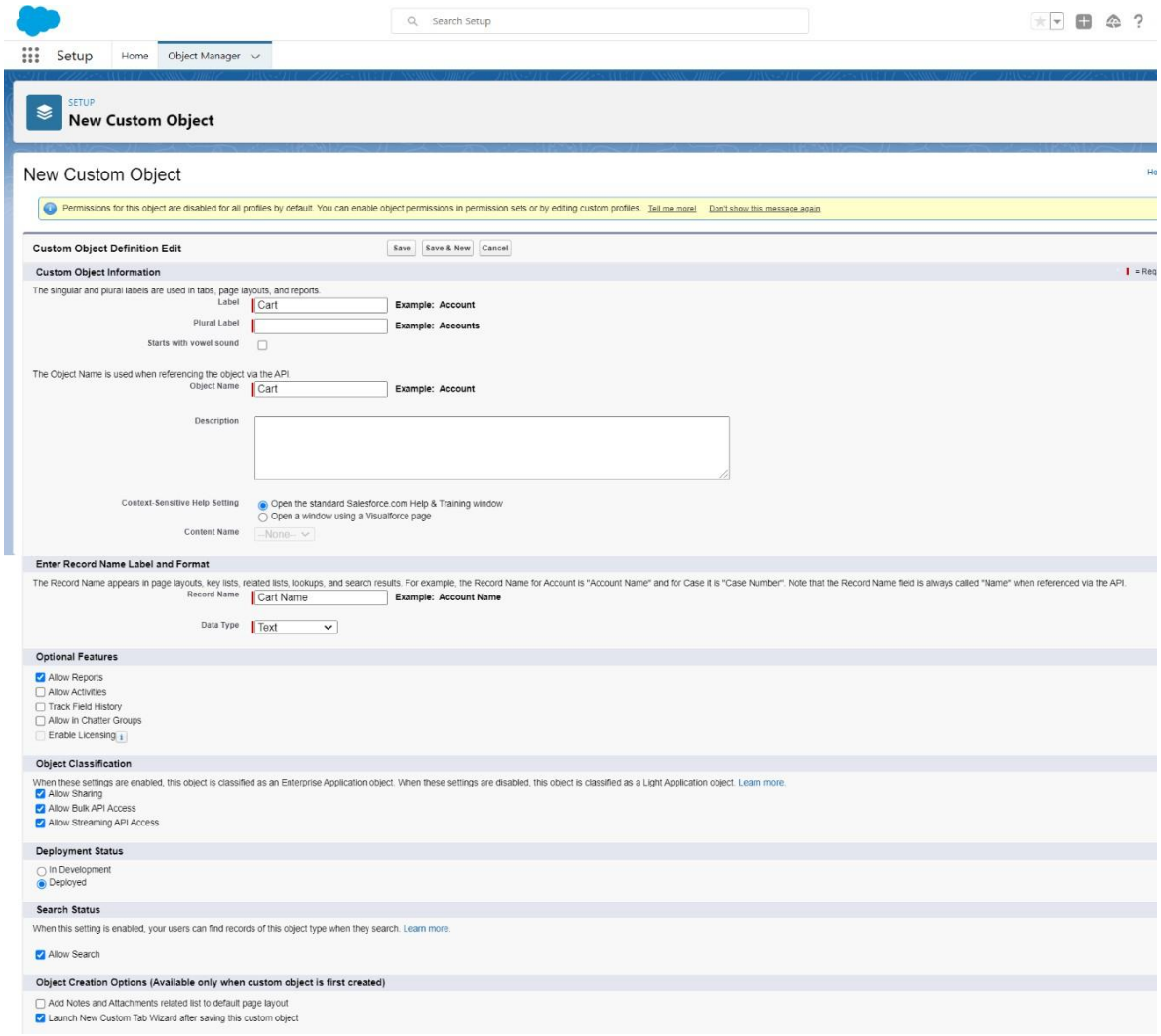


Q2) Create a Web Application to implement an online cart for adding items to a shopping cart and deleting it.

Launch your Salesforce Trailhead Playground by opening any module and Switch to Lightning Experienceif you are currently in Salesforce Classic by clicking your picture in the right top corner and then click on “Switch to Lightning Experience”.

Then go to Setup gear icon and click “Setup”.

Click on “Object Manager” and click “Create> Custom Object” to create new Custom Object.

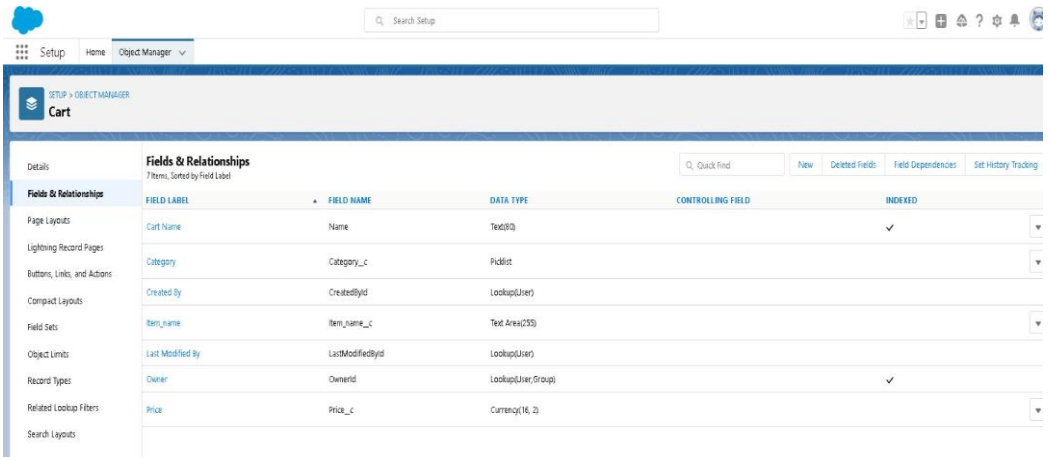


1. Name the object “Cart”.
2. Allow Reports and Allow Search.
3. Check the box in front of “Launch New Custom Tab Wizard after saving this custom object”
4. To create a Tab for the Object: Select any Tab Style for the object “Cart”. Click Next, Next, leavethe defaults and save.

To add fields to the Object:

Go to “Fields & Relationships” option of cart object and Click “New”.Add the following fields one after the other:

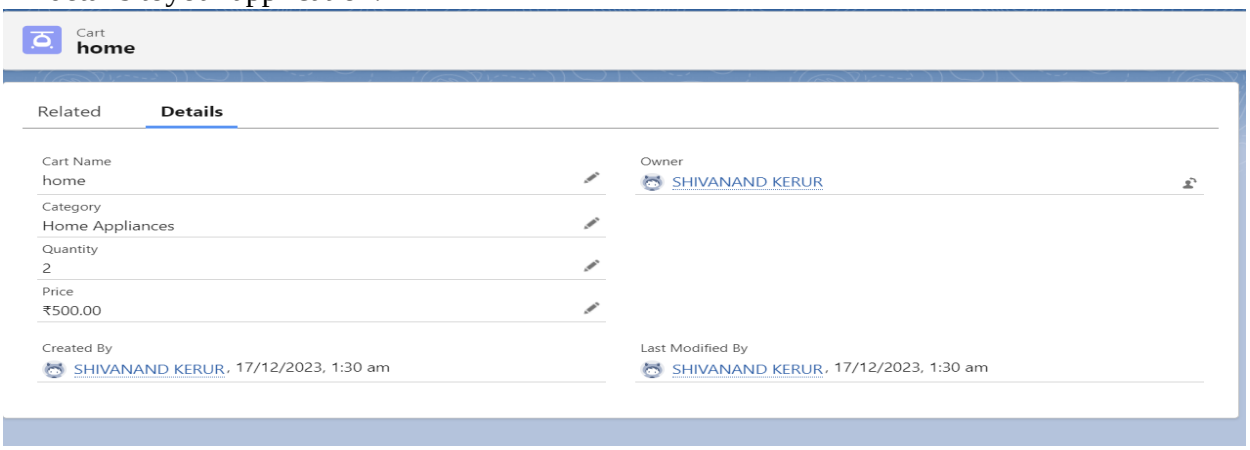
- Field Label: Item Name, Data Type: Text Area, make it as Required Field.

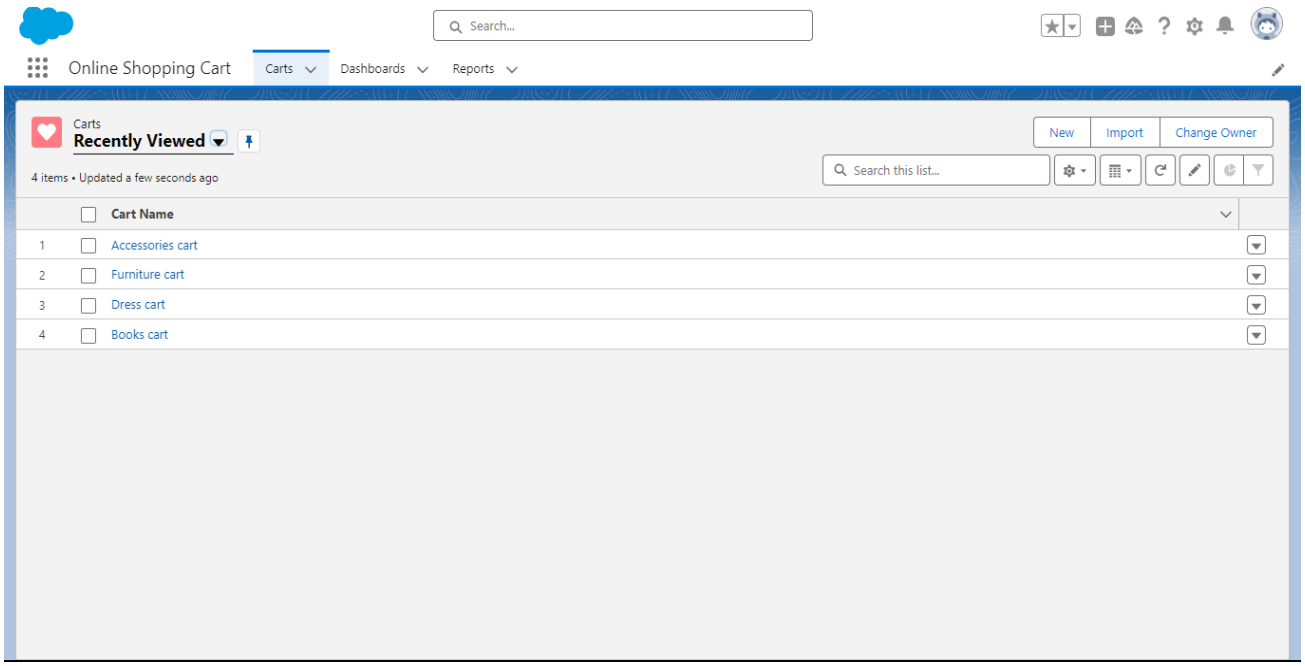


- Field Label: Category, Data Type: Picklist, click radio button in front of Enter values, with each value separated by a new line value are: Books, Electronics & Accessories, Furniture &Home Appliances, Fashion – Men, Fashion – Women, Fashion – Kids, Footwear and Others.
- Make it as Required Field and Restrict the values to the values in the picklist.
- Field Label: Quantity, Data Type: Number, make it as Required Field.
- Field Label: Price, Data Type: Currency (Length 16, Decimal Places 2), Make it as Required Field.

To create an application:

1. Go to “Setup” and type “App Manager” in Quick Find Box.
2. Click on “New Lightning App” to create a Lightning Application.
3. Name it as “Online Shopping Cart”, give the description for your application.
4. Uploading Image and changing colours are optional, then click Next.
5. Navigation Style: Standard Navigation, click Next.
6. No need to add any Utility Bar, click Next.
7. Add the following Items: Carts, Reports and Dashboards, click Next.
8. Assign it to System Administrator Profile by selecting System Administrator and pressing rightarrow and then click Save & Finish.
9. Go to App Manager, select your application and select Carts and click “New” to add some details toyour application.

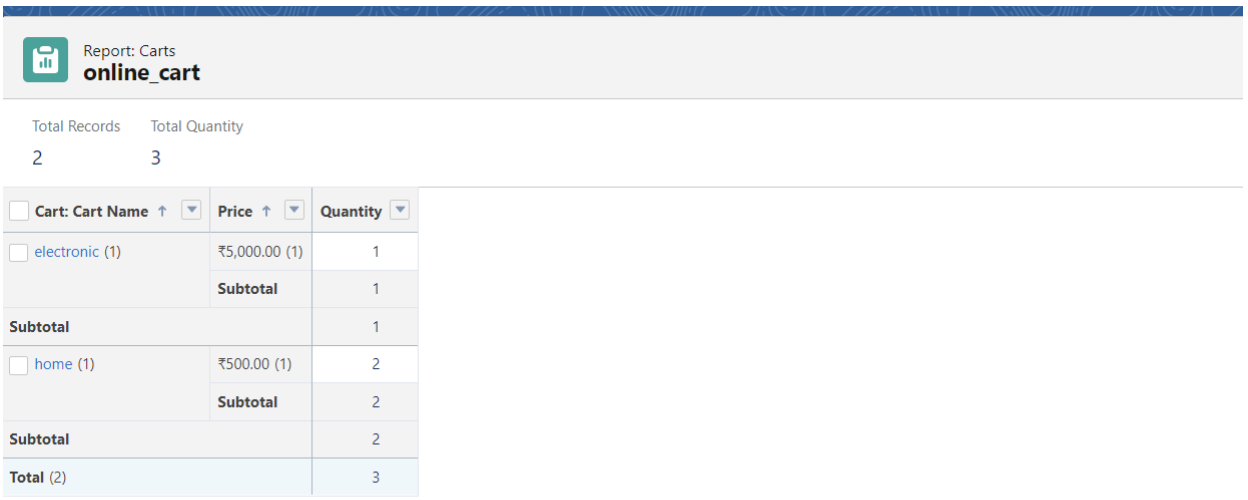




Reports and Dashboards:

To Create a Carts Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name and then click Save.
2. Click on “New Report” and from search bar Search for “Carts” and then select it and click Continue.
   • Add the required Columns to get the Completed Entered data.
3. If you Want the report to be grouped by any specific Fields then Search for the field in “Add groups”otherwise it is optional.
4. Click on save and name the report as “New Carts Report” and then select the folder which you havecreated.
5. Click Save and then Click Run



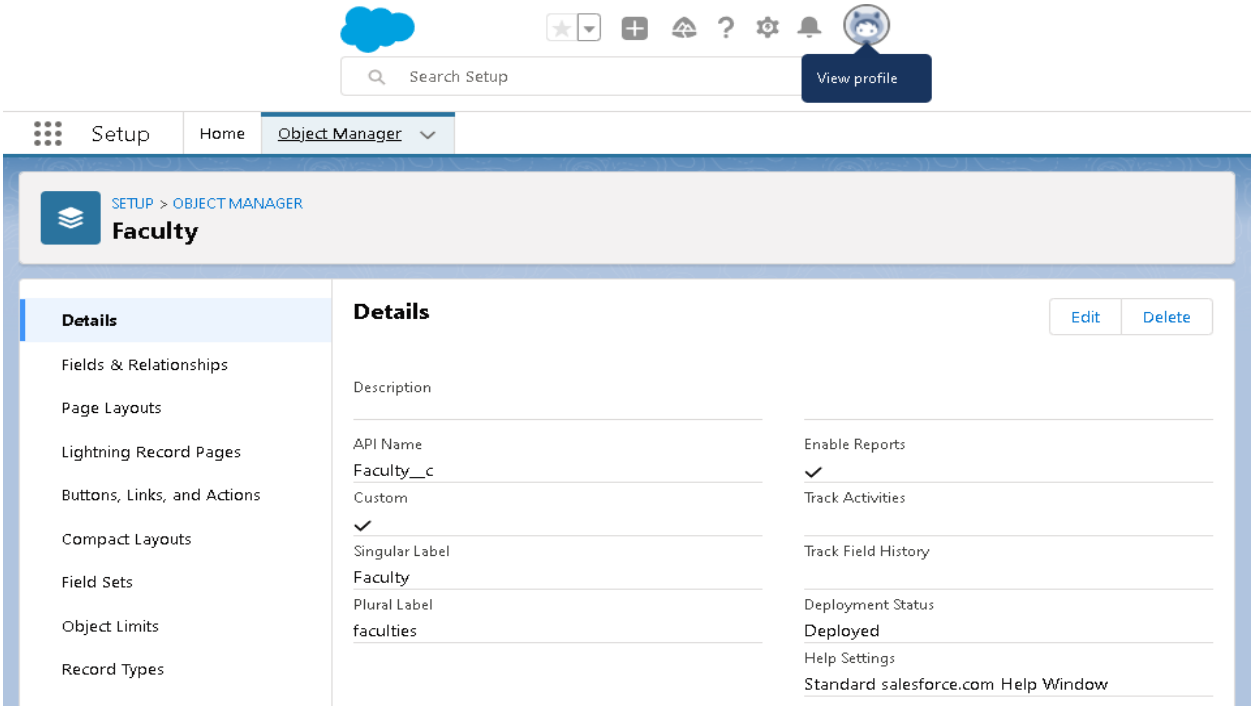
To Create a Carts Dashboard:

1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Shopping cart” and select folder that you havecreated Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run

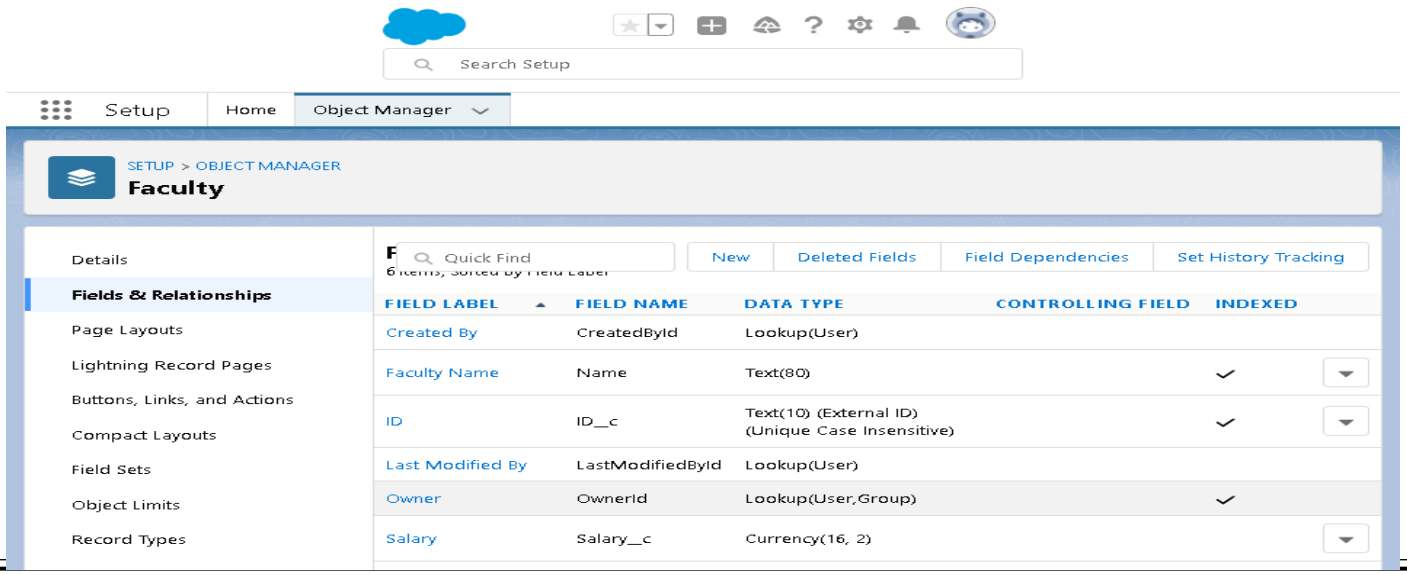
Q3) Create a web application to enter the faculty details like faculty ID, faculty name,and salary to a database and calculate the income tax to be paid by the faculty at the end of the financial year.

Launch your Salesforce Trailhead Playground by opening any module and Switch to Lightning Experienceif you are currently in Salesforce Classic by clicking your picture in the right top corner and then click on “Switch to Lightning Experience”.

Then go to Setup gear icon and click “Setup”.



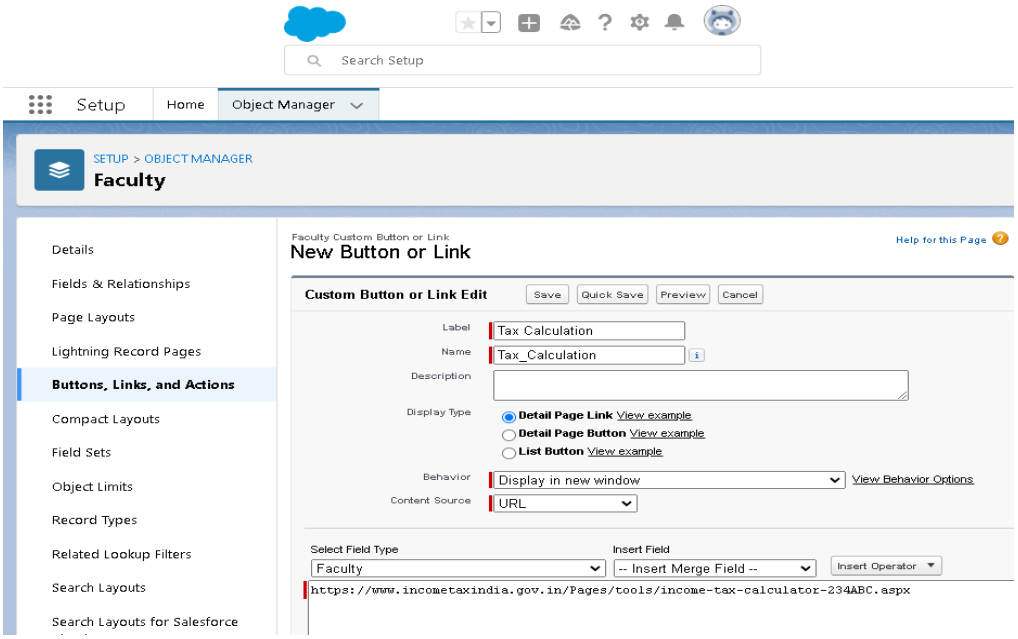
1. Click on “Object Manager” and click “Create> Custom Object” to create new Custom Object.
2. Name the object “Faculty”
3. Allow Reports and Allow Search.
4. Check the box in front of “Launch New Custom Tab Wizard after saving this custom object”
5. To create a Tab for the Object: Select any Tab Style for the object “Faculty”. Click Next, Next, leavethe defaults and save.



To add fields to the Object:

Go to “Fields & Relationships” option of Student object and Click “New”Add the following fields one after the other:

- Field Label: ID (Length 10), Data Type: Text, provide an example ID as Help Text, make it as Required Field, don’t allow Duplicate Values, make it as Case Insensitive and Set this field as the unique record identifier from an external system
- Field Label: Salary, Data Type: Currency (Length 16, Decimal Places 2), Make it as Required Field

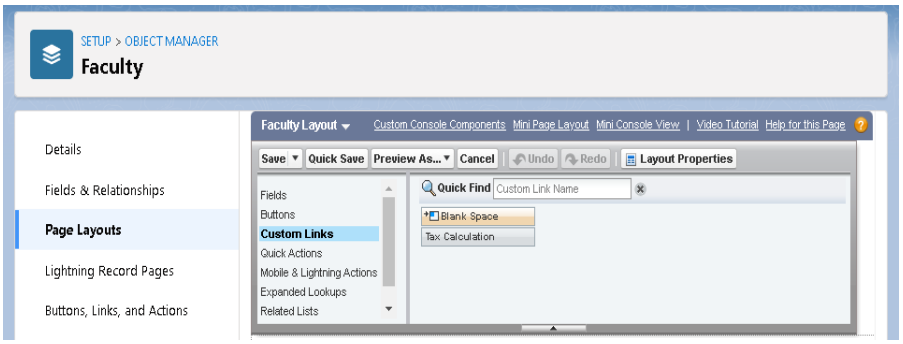


To calculate Income Tax to be paid:

1. Go to “Buttons, Links and Actions” of Faculty Object and click “New Button or Link”
2. Name it as “Tax Calculation”
3. Select the radio button “Detail Page Link” as it is a website link.
4. Behaviour: Display in new window.
5. Content Source: URL.

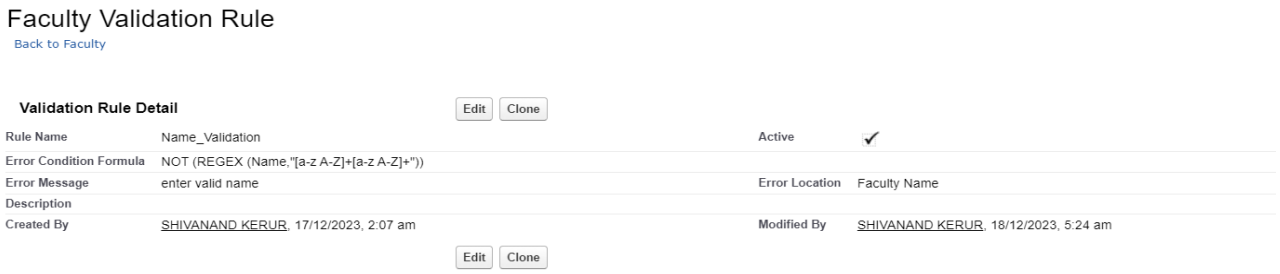
6. Field Type: Faculty
7. In the empty space provided, type https://www.incometaxindia.gov.in/Pages/tools/income-tax-calculator-234ABC.aspx
8. It is a link which redirects to the income tax calculation website.
9. Link Encoding: Unicode (UTF-8).
10. Click Save
11. Go to Page Layout, Click Faculty Layout.
12. Click Custom Links, Drag and drop the “Tax Calculation” link in the Custom Link area.

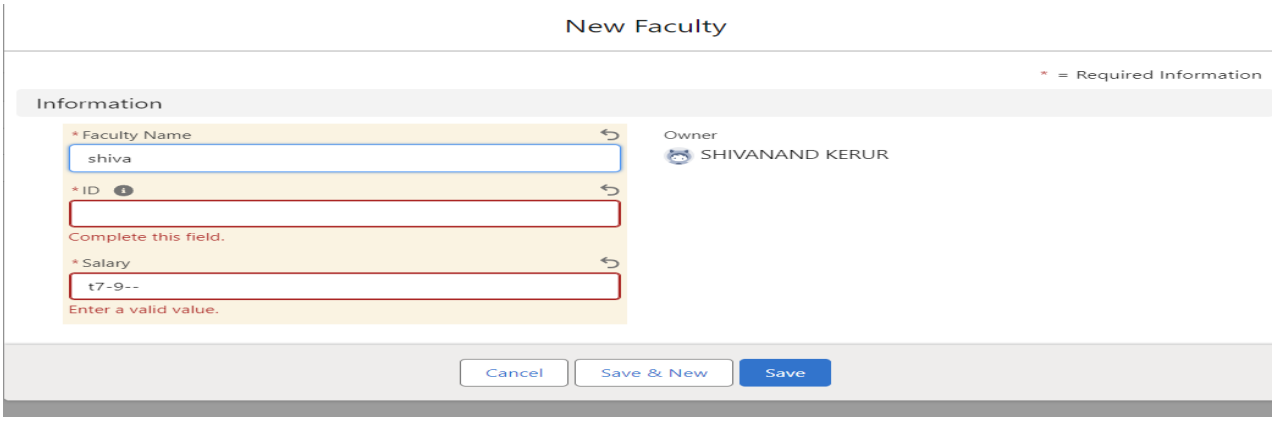
13. Click Save



To add a rule to the faculty name so that it should take only valid names:

1. Go to Validation Rule of Faculty object and click “New”
2. Name it as “Name Validation”.
3. Error Condition Formula: NOT (REGEX (Name,” [a-z] [A-Z] + [a-z] [A-Z] +”)).
4. Error Message: Please Enter a valid name.
5. Error Location: Field – Faculty name.
6. Click Save



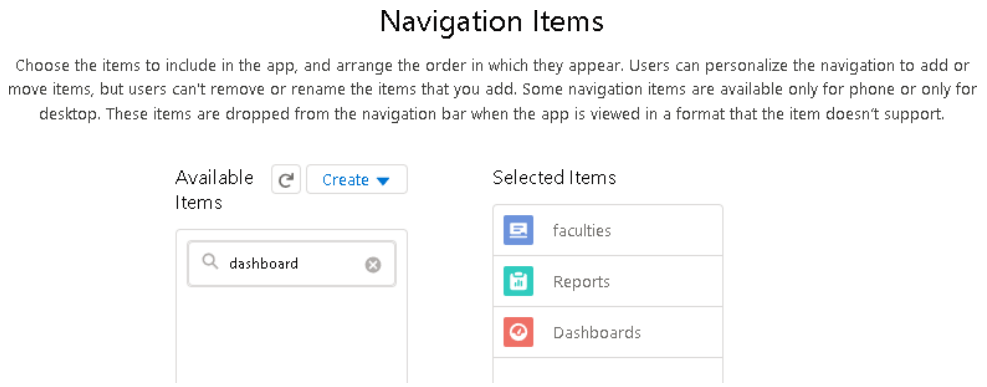


To create an application:

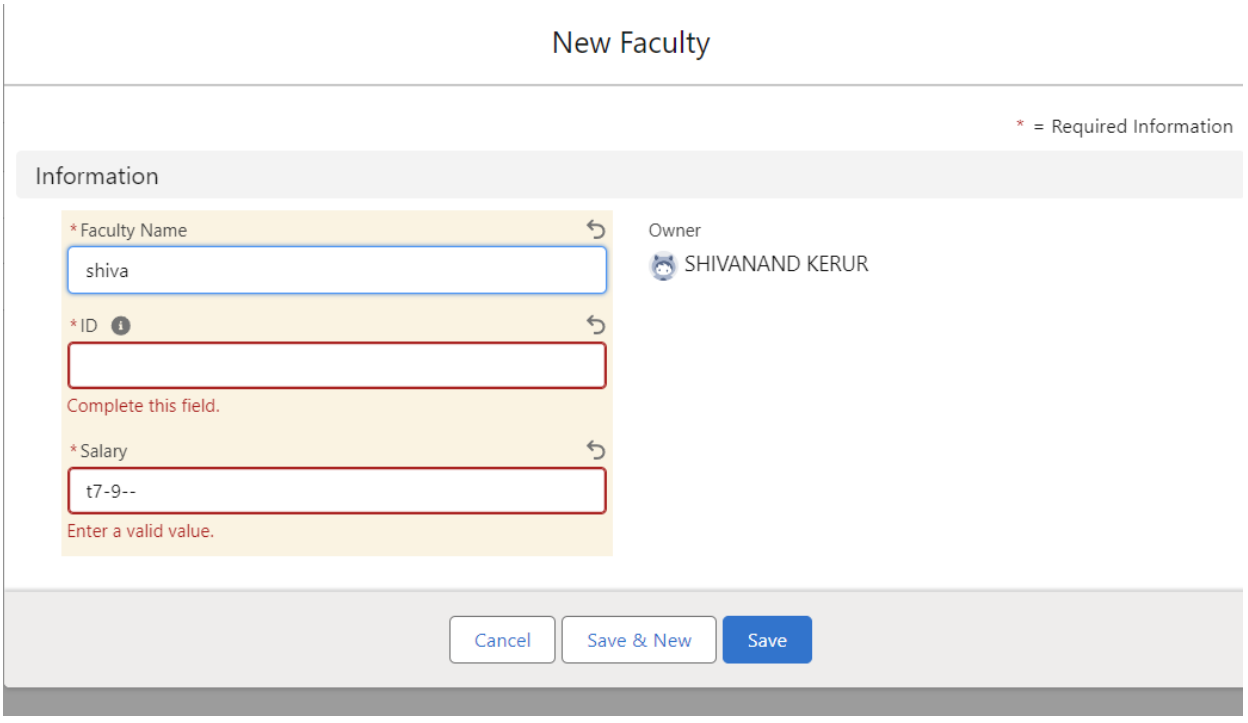
1. Go to “Setup” and type “App Manager” in Quick Find Box.
2. Click on “New Lightning App” to create a Lightning Application.
3. Name it as “Faculty Database”, give the description for your application.
4. Uploading Image and changing colours are optional, then click Next.
5. Navigation Style: Standard Navigation, click Next.
6. No need to add any Utility Bar, click Next.

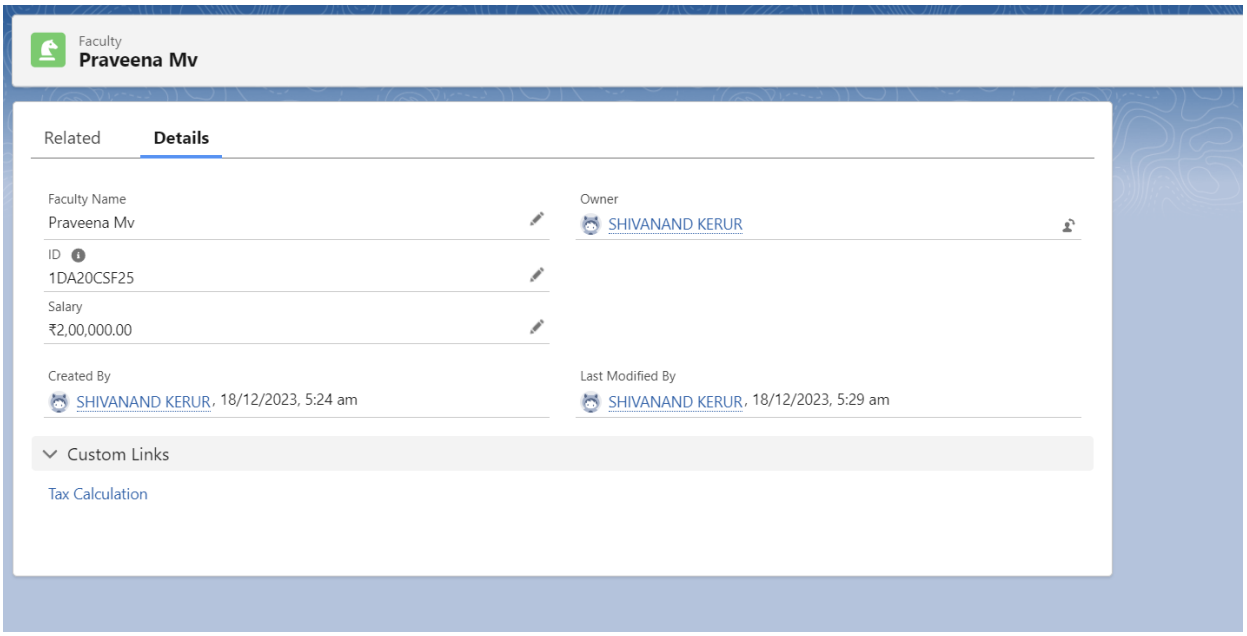
7. Add the following Items: Faculties, Reports and Dashboards, click Next.
8. Assign it to System Administrator Profile by selecting System Administrator and pressing right arrow and then click Save & Finish.

Go to App Manager, select your application and select Faculties and click “New” to add some details toyour application.



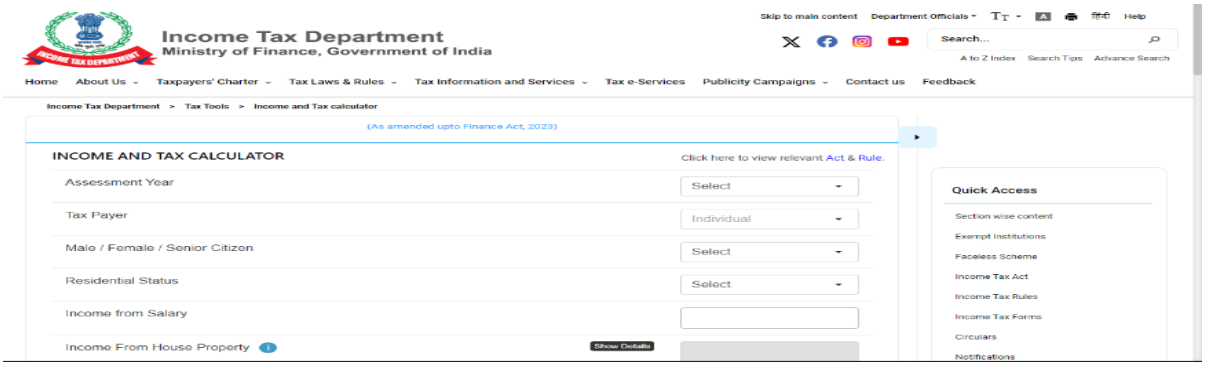
Click the entry you added, go to details. Make sure you will get an error message when you enter an invalidname and invalid id.





Press the “Tax Calculation” link to calculate income tax.

Click OK so that it will redirect you to the income tax calculator website



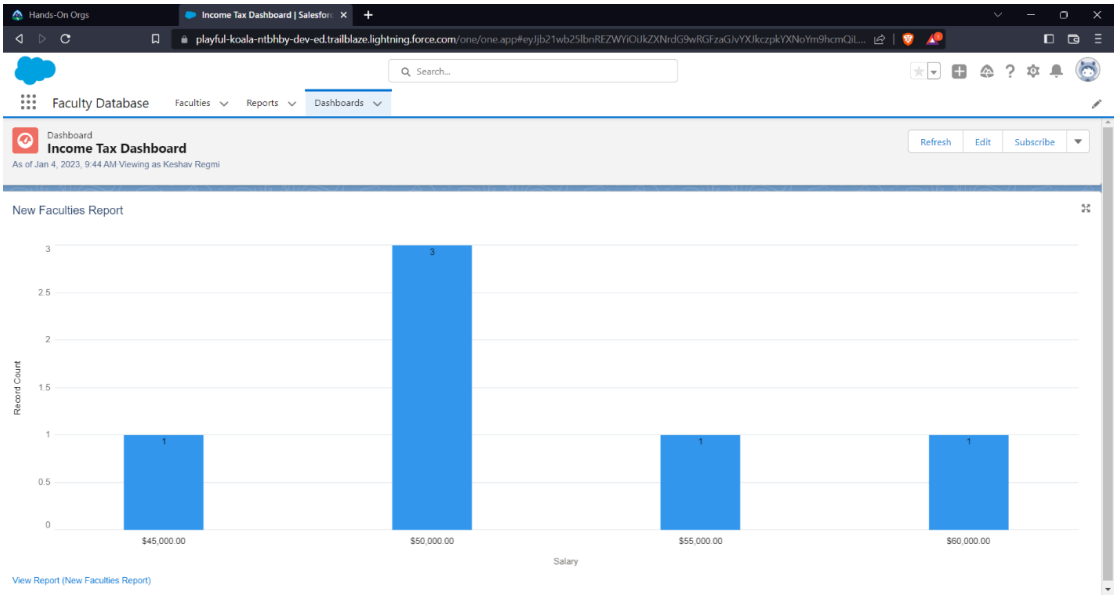
Enter the required Details and press “Calculate”.

Reports and Dashboards:

To Create a Faculty Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name and click on Save.
2. Click on “New Report” and from search bar Search for “Faculty” and then select it then Click Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups”otherwise it is optional.
5. Click on save and name the report as “New Faculties Report” and then select the folder which youhave created.
6. Click Save and then Click Run

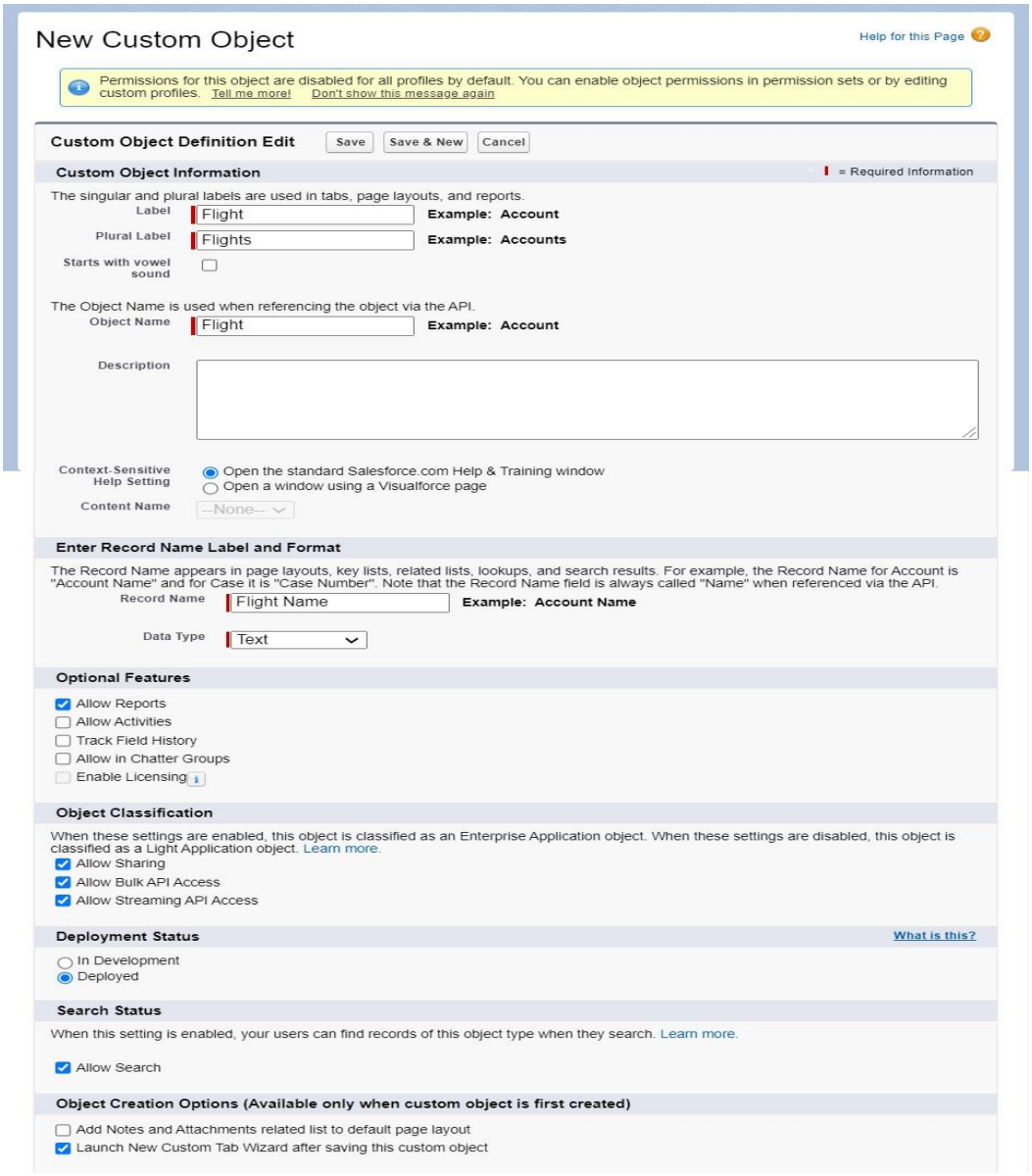
To Create a Faculty Dashboard:

1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Income Tax Dashboard” and select folder that youhave created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run
7. 

Q4) Create a web application to book a flight from a source to destination and store the status of flight, and departure timings on database.

Launch your Salesforce Trailhead Playground by opening any module and Switch to Lightning Experience if you are currently in Salesforce Classic by clicking your picture in the right top corner and then click on "Switch to Lightning Experience"

Then go to Setup gear icon and click "Setup".



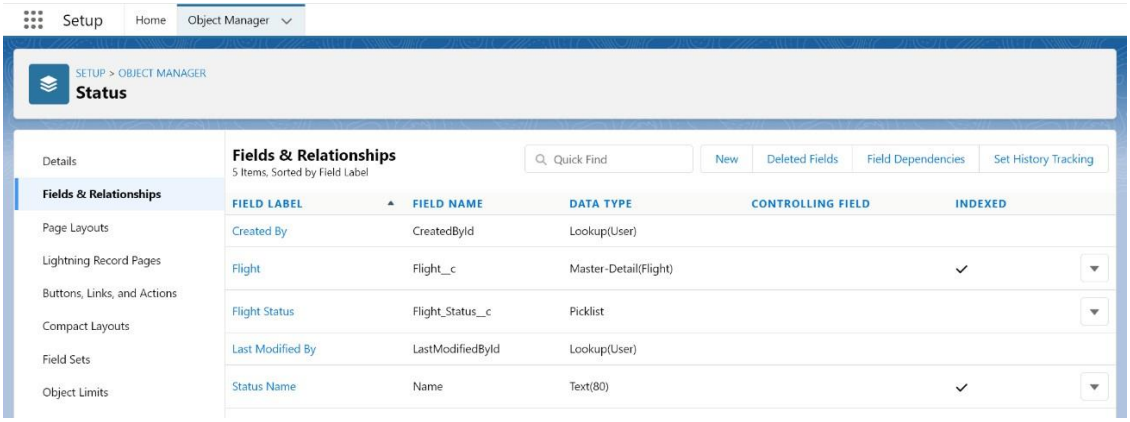
1. Click on "Object Manager" and click "Create > Custom Object" to create new Custom Object.
2. Name the object "Flight".
3. Allow Reports and Allow Search.
4. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object".
5. To create a Tab for the Object: Select any Tab Style for the object "Flight". Click Next, then, leave the defaults and save.

To add fields to the Object:

Go to “Fields & Relationships” option of Student object and Click “New”.

Add the following fields one after the other:

- Field Label: Source, Data Type: Text Area, make it as Required Field.
- Field Label: Destination, Data Type: Text Area, make it as Required Field.
- Field Label: Departure Timing, Data Type: Date/Time, make it as Required Field.

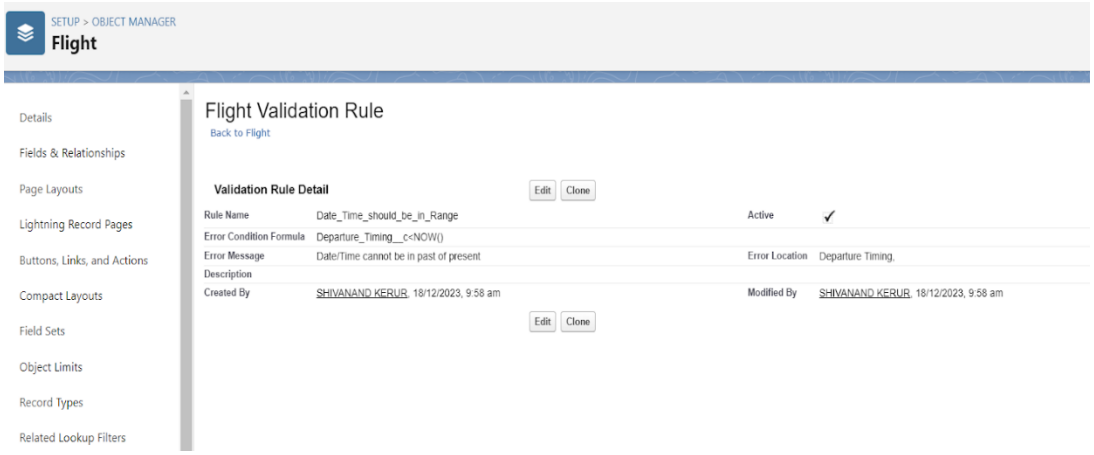


Validation Rules:

Date and time Validation:

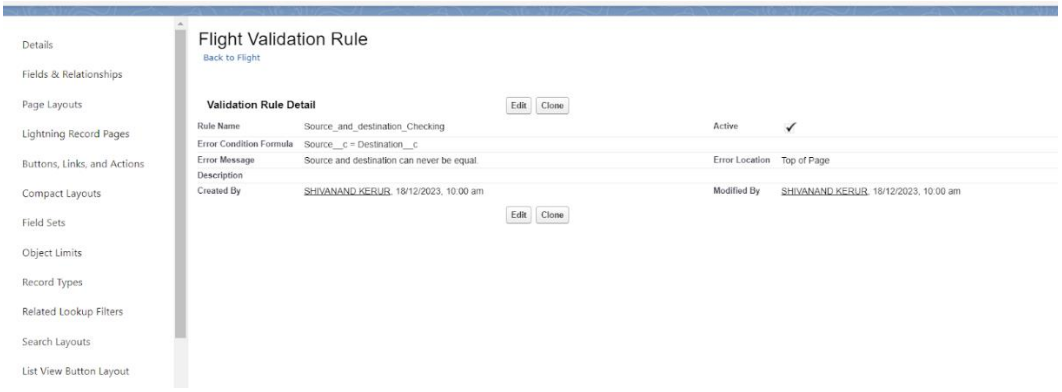
To add a rule to the departure timing so that it is greater than today’s date and the present time:

1. Go to Validation Rule of Flight Object and click “New”
2. Name it as “Date Time should be in Range”
3. Error Condition Formula: Departure_Timing__c<NOW ()
4. Error Message: Departure Date/Time cannot be in past of present.
5. Error Location: Field – Departure Timings.
6. Click Save



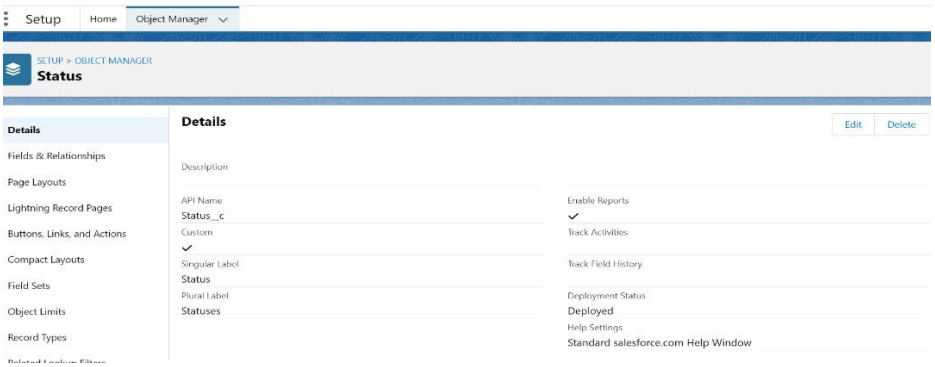
Source and Destination Validation:

To add a rule to the Source and destination so that source and destination should not be equal:

1. Go to Validation Rule of Flight Object and click “New”
2. Name it as “Source and destination Checking”
3. Error Condition Formula: Source__c = Destination__c
4. Error Message: Source and destination can never be equal.
5. Error Location: Top of the Page.
6. Click Save.
7. 

Create one more object to provide status of the flight:

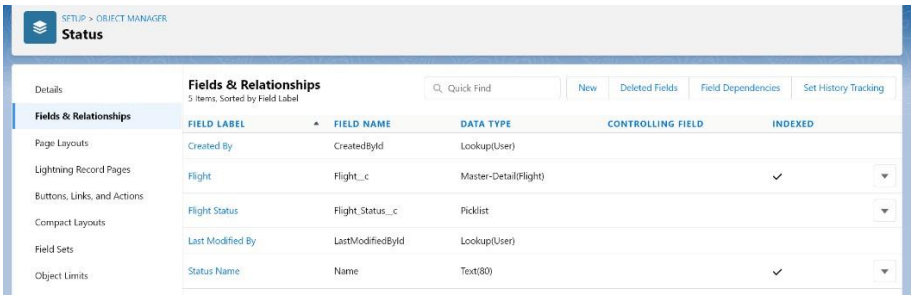
1. Name the Object “Status”
2. Allow Reports and Allow Search.
3. Check the box in front of “Launch New Custom Tab Wizard after saving this custom object”
4. Create a Tab for the Object.



To add fields to the Object: Go to “Fields & Relationships” option of Student object and Click “New”.

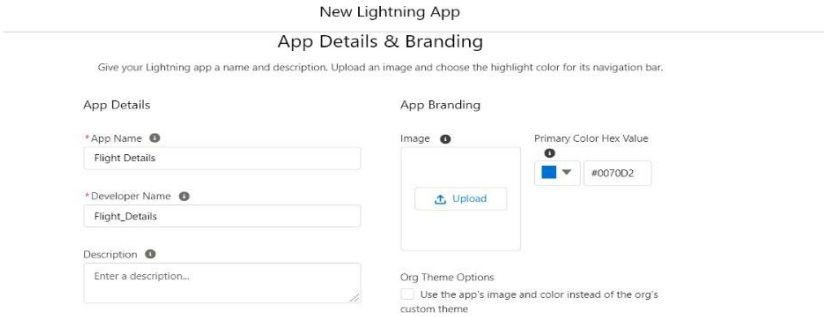
Add the following fields one after the other:

- Field Label: Flight Name, Data Type: Master-Detail Relationship, Related to: Flight. Sharing Setting: Read-Only. Leave the defaults and save. Master – Detail relationship is provided to enterstatus only to the existing flights.
- Field Label: Flight Status, Data Type: Picklist, click radio button in front of Enter values, with each value separated by a new line. Values are: Arrived, Cancelled, Delayed and Departed. Make it as Required Field and Restrict the values to the values in the picklist.

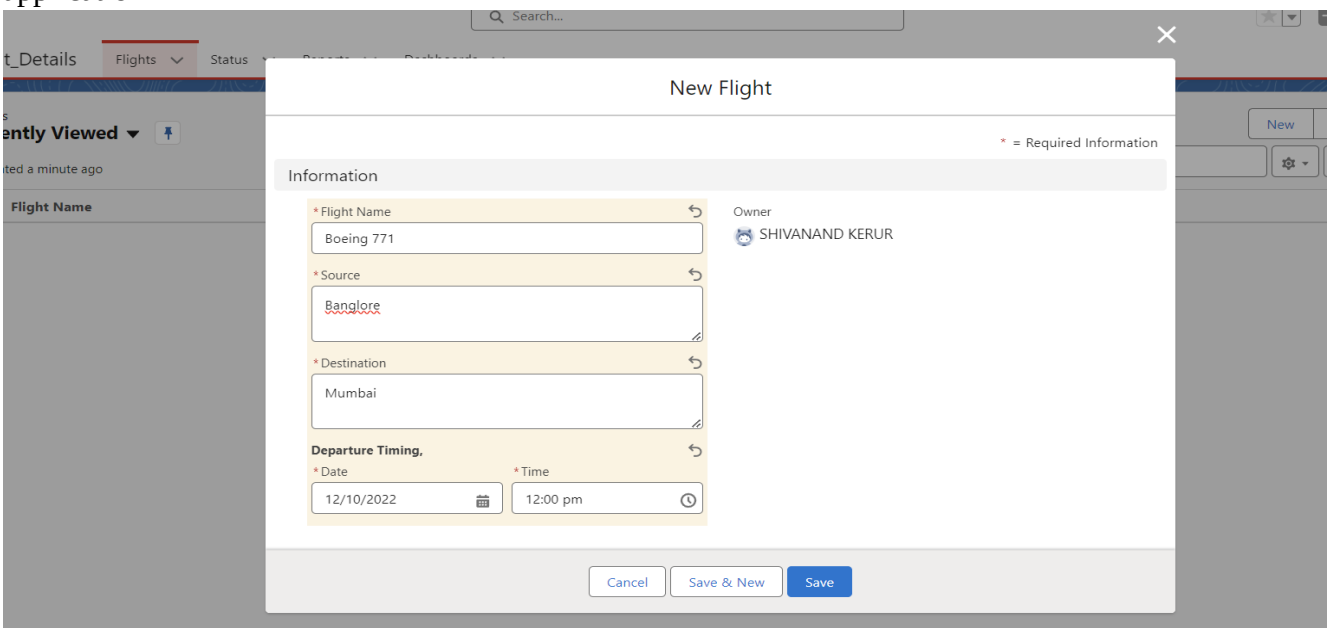


To create an application:

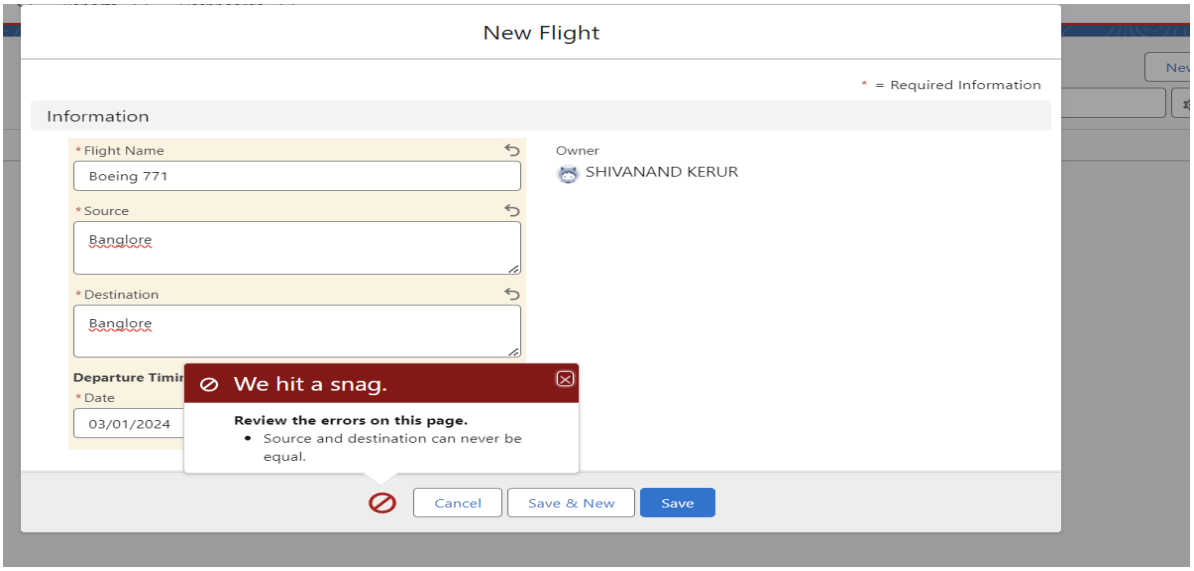
1. Go to “Setup” and type “App Manager” in Quick Find Box.
2. Click on “New Lightning App” to create a Lightning Application.
3. Name it as “Flight Details”, give the description for your application.
4. Uploading Image and changing colours are optional, then click Next.
5. Navigation Style: Standard Navigation, click Next.
6. No need to add any Utility Bar, click Next.
7. Add the following Items: Flights, Statuses, Reports and Dashboards, click Next.
8. Assign it to System Administrator Profile by selecting System Administrator and pressing right arrow and then click Save & Finish.



Go to App Manager, select your application and select Flights and click “New” to add some details to your application



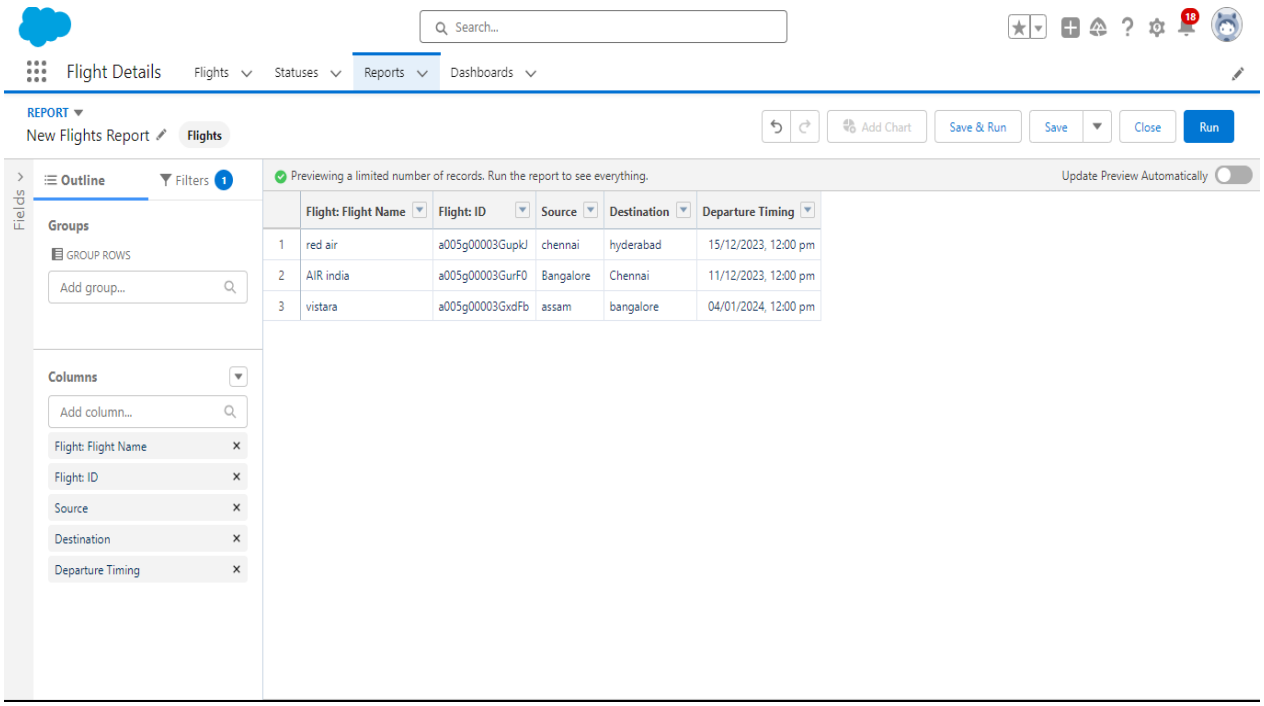
Make sure you will get an error message when you try to give the Departure Timing less than the current time and today’s date



Reports and Dashboards:

To Create a Flights Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name and then click Save.
2. Click on “New Report” and from search bar Search for “Flights” and then select it and Click Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Flights Report” and then select the folder which you have created.
6. Click Save and then Click Run.

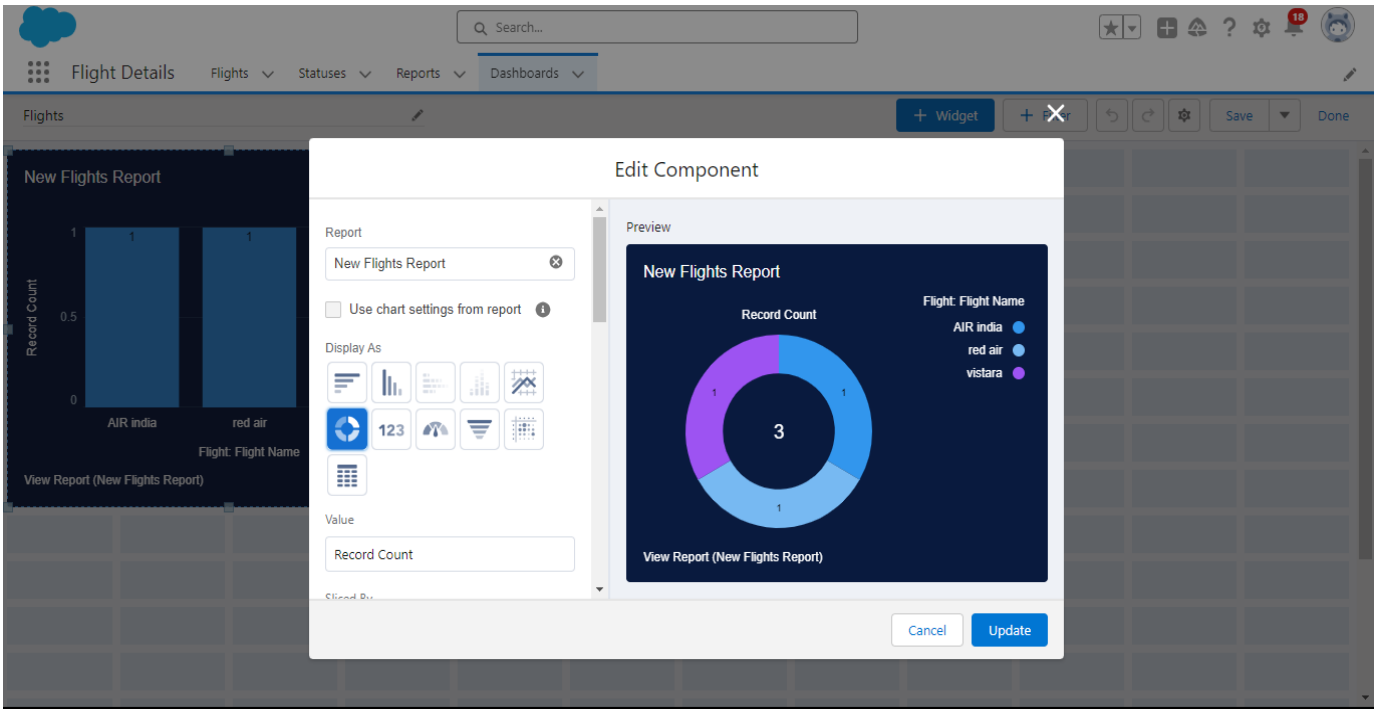


To Create a Status Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name and then click Save.
2. Click on “New Report” and from search bar Search for “Flights with Status” and then select it and then click Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Flights with status Report” and then select the folder which you have created.
6. Click Save and then Click Run

To Create a Status Dashboard:

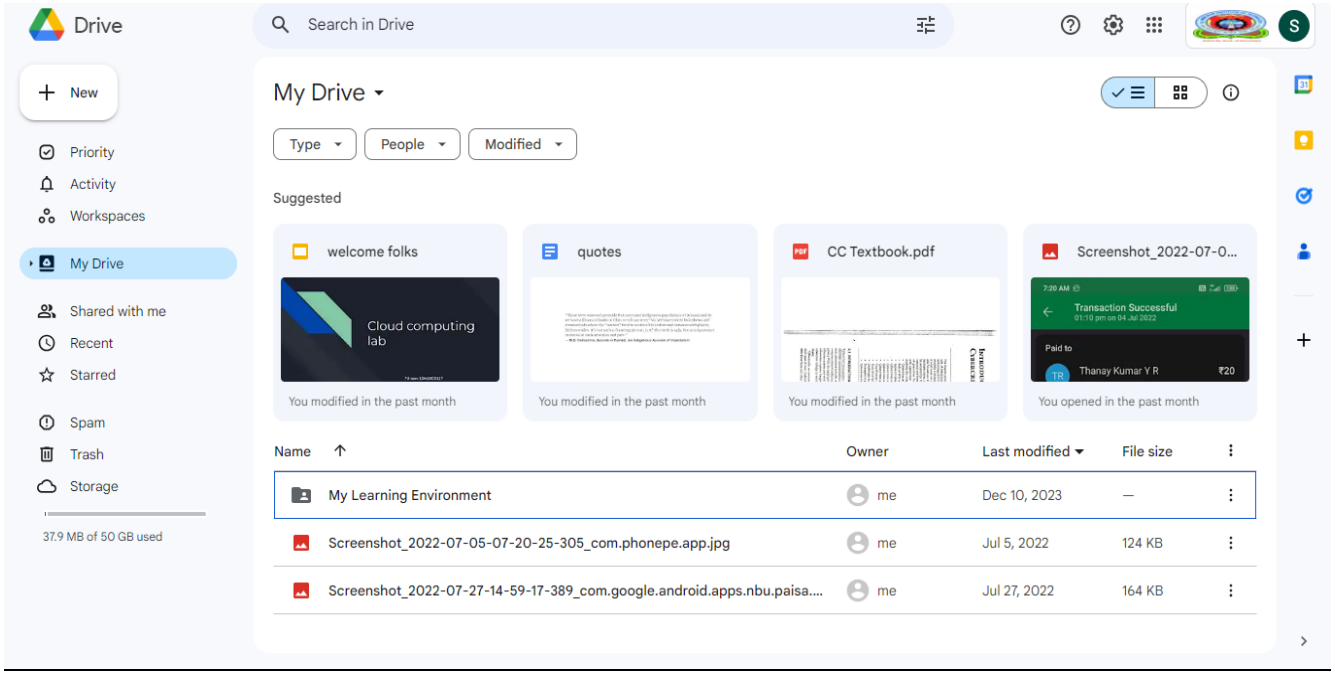
1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Status” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run



Q5) Create a collaborative learning environment for a particular learning topic using Google Apps. Google Drive, Google Docs and Google Slides must be used for hosting e-books, important articles and presentations.

Open http://drive.google.com/ and Sign In with your google account.

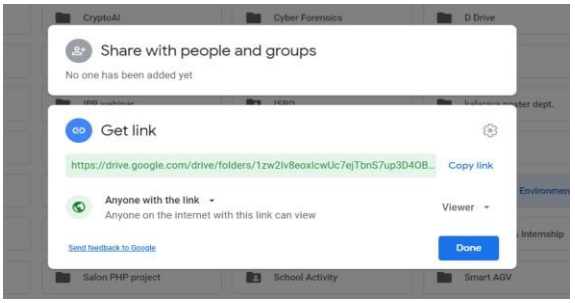
Create a new folder named “My Learning Environment” by clicking “New” button on the top left corner.



Right click on the folder created and tap “Share” then click on “Advanced”.

Under the “Who has Access” section click on “Change” of the first option.

Now check on the “On- Public on the web” option & set the Access to “Can View Only” and Click Save. This will make your folder to be accessible by anyone on the internet to view its contents and download them



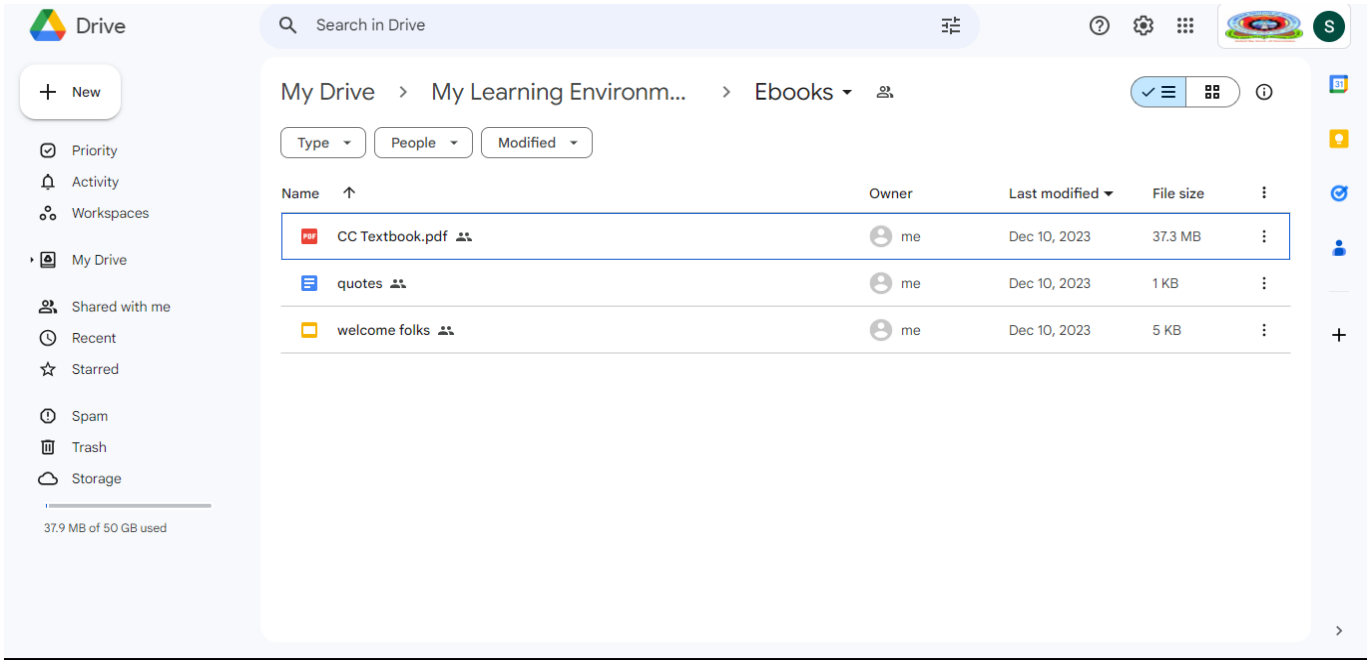
Copy the link and post it or share it to anyone you like.

Adding Learning Contents to your Environment:

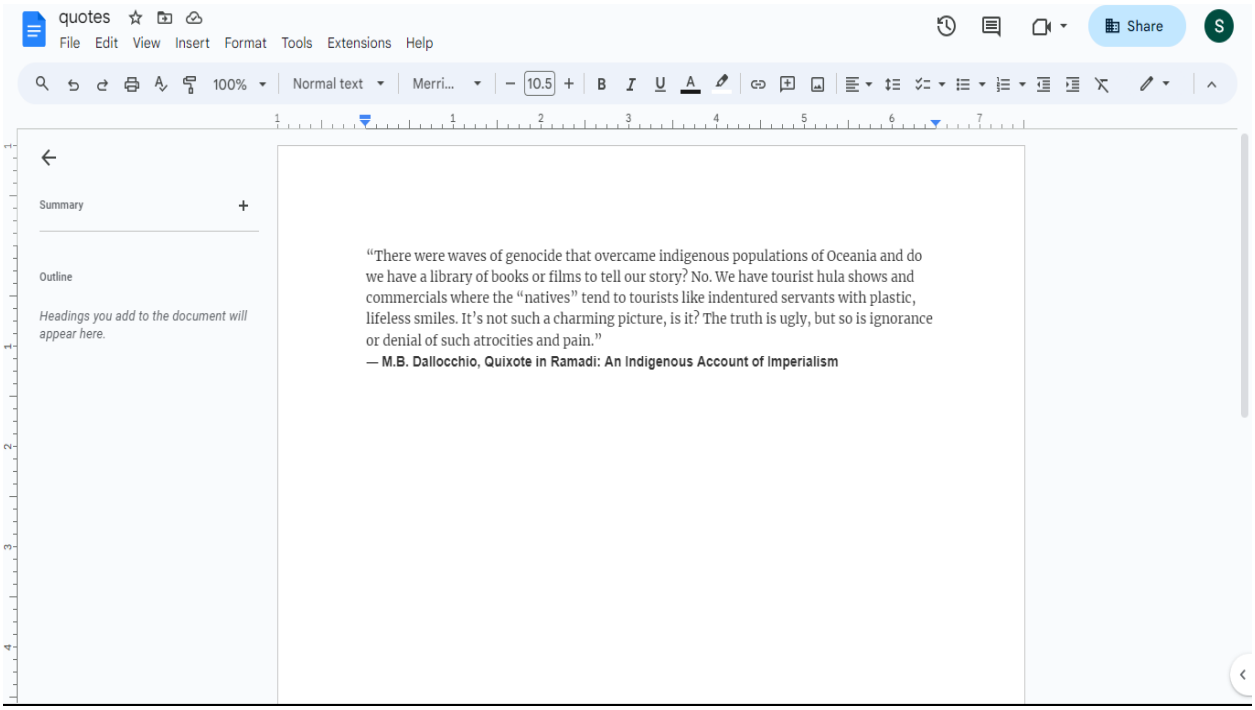
Double click on the folder you just created and click “New” button again.

Add these items:

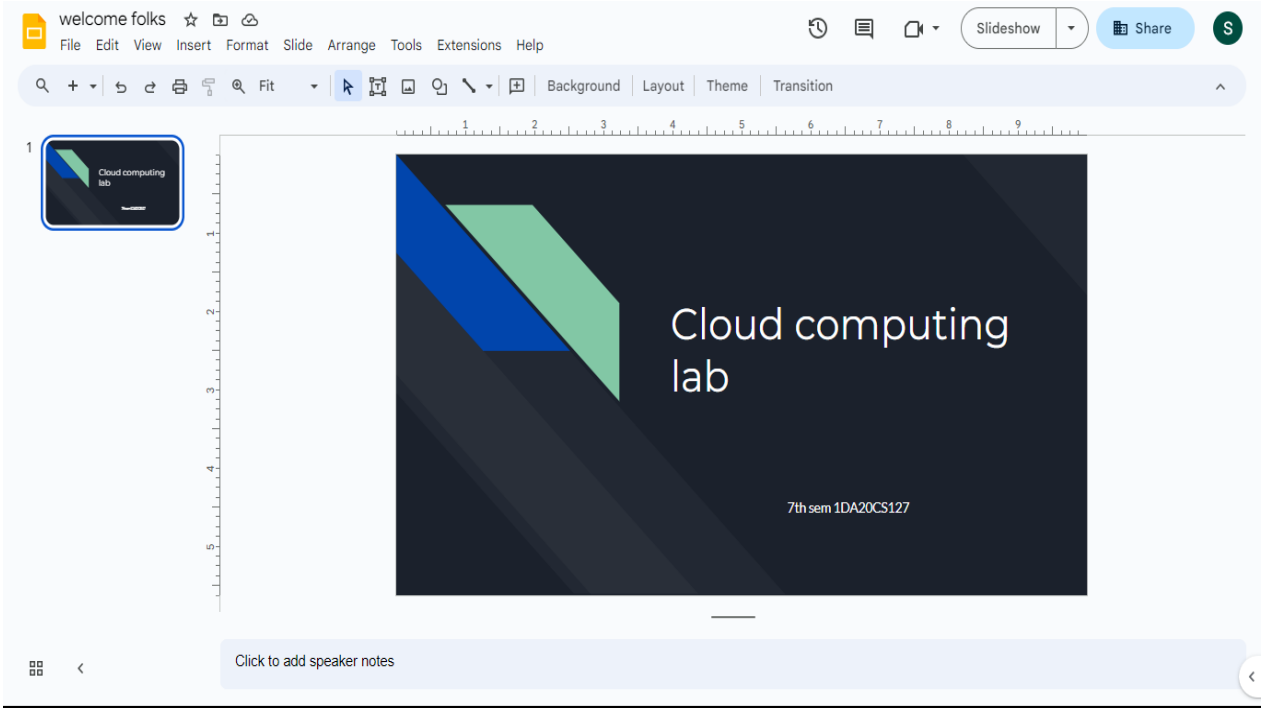
- Folder: Named “E-Books” where you will open the folder and click “New” and “Upload a file” like a Book from your hard drive



- Google Docs: Named “Important Quotes” where you will add some important links to the doc file. The file will keep saving hence you need to press save. click “Share” if you wish to share it.



- Google Slide: Named “Welcome Folks” make a few changes and add your content and choose your theme



You may also add many other items as necessary.

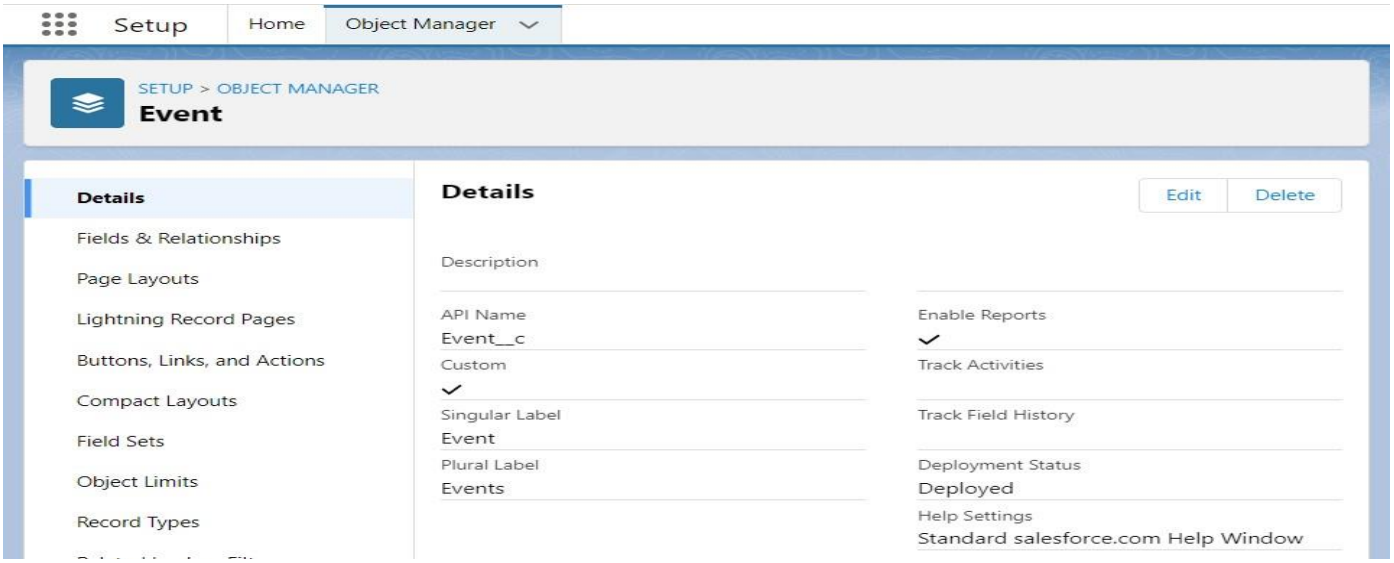
At any point in future if you wish to share this environment right click on the folder and click share. Copy the link and share.

Q6) Develop Department events registration app with an object containing event name, date/time, venue as parent relationship, another object containing student name, branch, event name, date/time, venue as child relationship.

Launch your Salesforce Trailhead Playground by opening any module and Switch to Lightning Experience if you are currently in Salesforce Classic by clicking your picture in the right top corner and then click on "Switch to Lightning Experience".

Then go to Setup gear icon and click "Setup".

1. Click on "Object Manager" and click "Create > Custom Object"to create new Custom Object.
2. Name the object "Event".
3. Allow Reports and Allow Search.
4. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object".
5. create a Tab for the Object:
6. Select any Tab Style for the object "Event". Click Next, Next, leave the defaults and save.

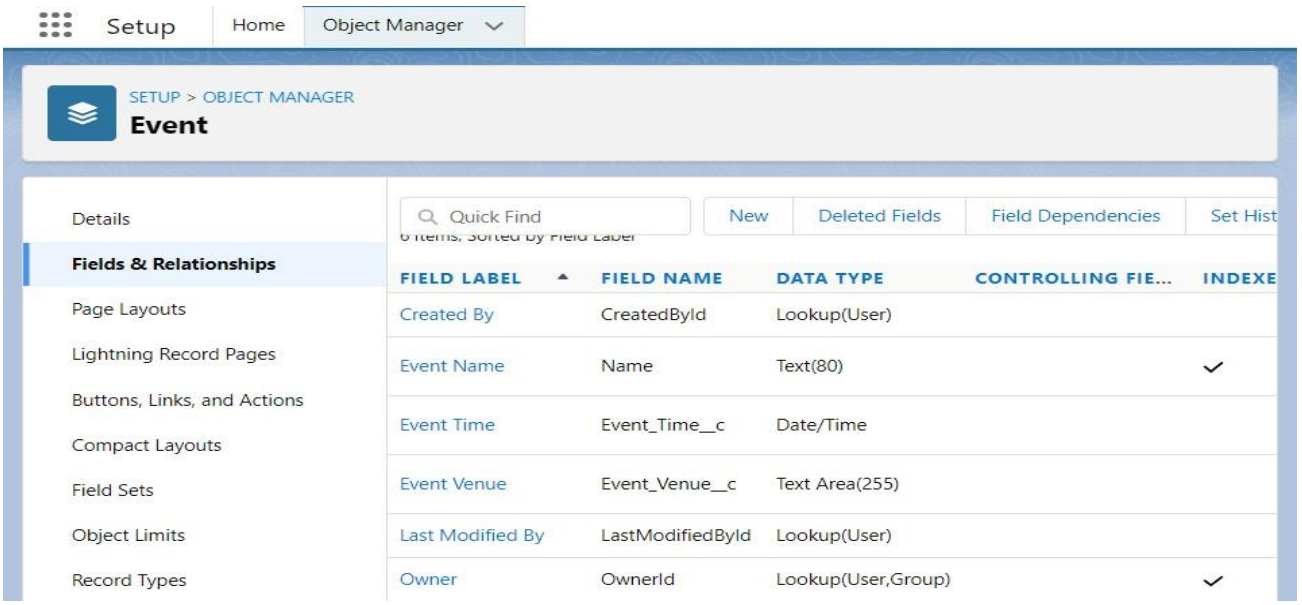


To add fields to the Object:

Go to "Fields & Relationships" option of Student object and Click "New".

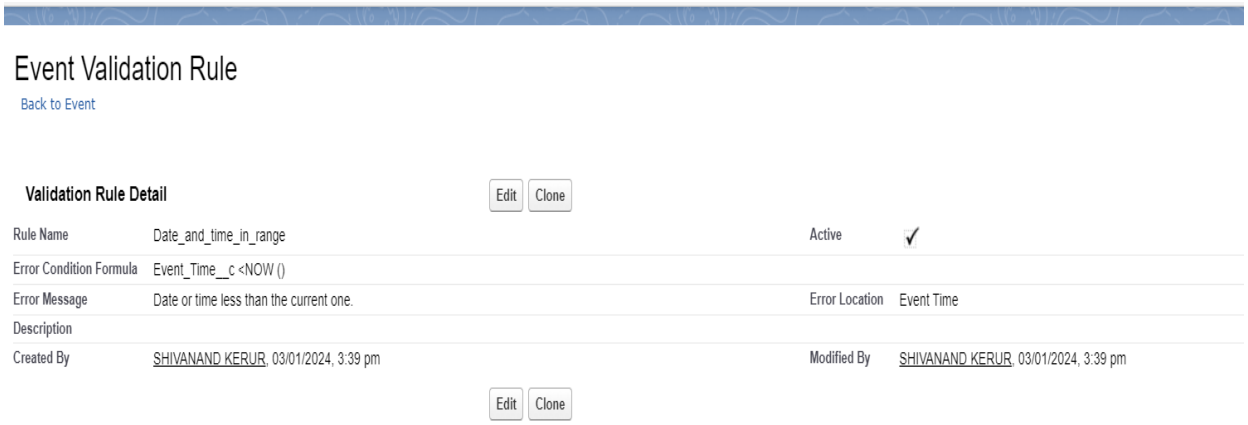
Add the following fields one after the other:

- Field Label: Event Time, Data Type: Date/Time, make it as Required Field.
- Field Label: Event Venue, Data Type: Text Area, make it as Required Field.



To add a rule to the Event Date/Time so that it is greater than today's date and the present time:

1. Go to Validation Rule of Event Object and click "New".
2. Name it as "Date and Time in Range".
3. Error Condition Formula: Event_Time__c <NOW ()
4. Error Message: Date or time less than the current one.
5. Error Location: Field – Event Time.
6. Click Save.



Create one more object to store student details:

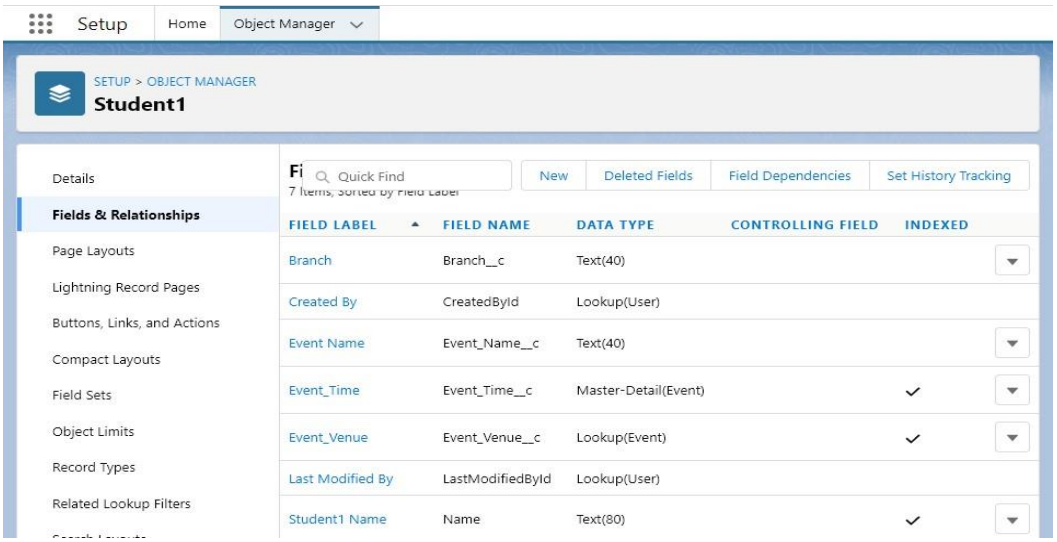
1. Name the Object "Student1".
2. Allow Reports and Allow Search.
3. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object".
4. Create a Tab for the Object.

To add fields to the Object:

Go to “Fields & Relationships” option of Student object and Click “New”.

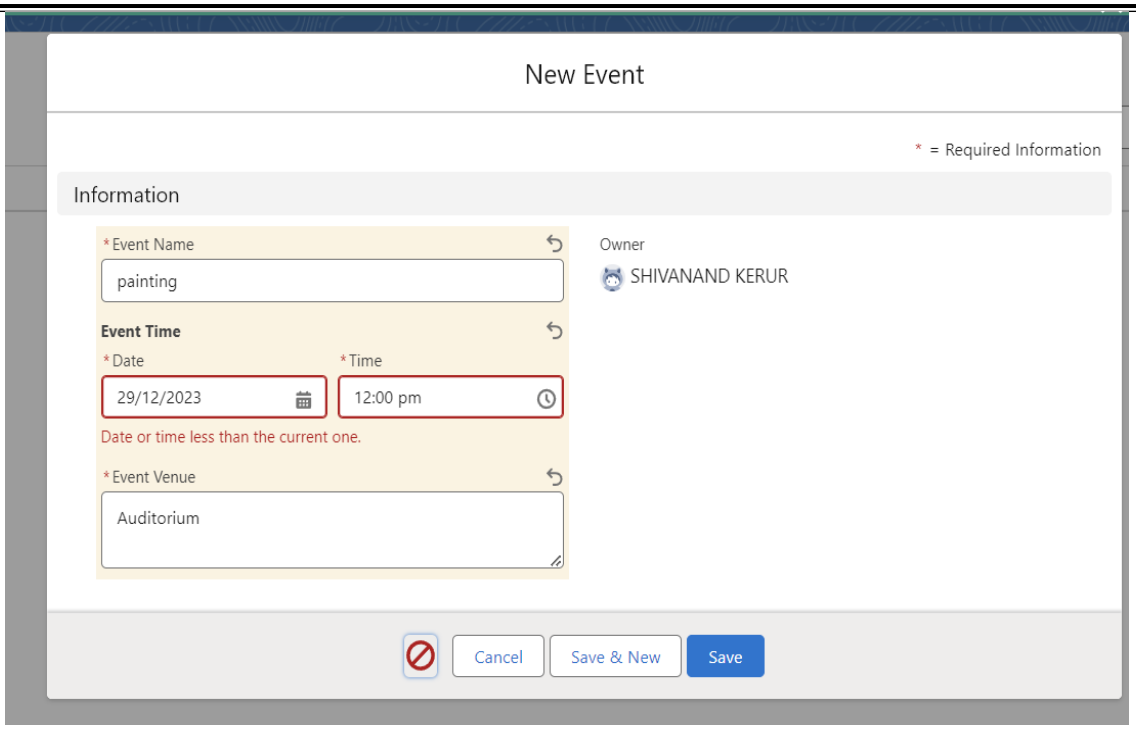
Add the following fields one after the other:

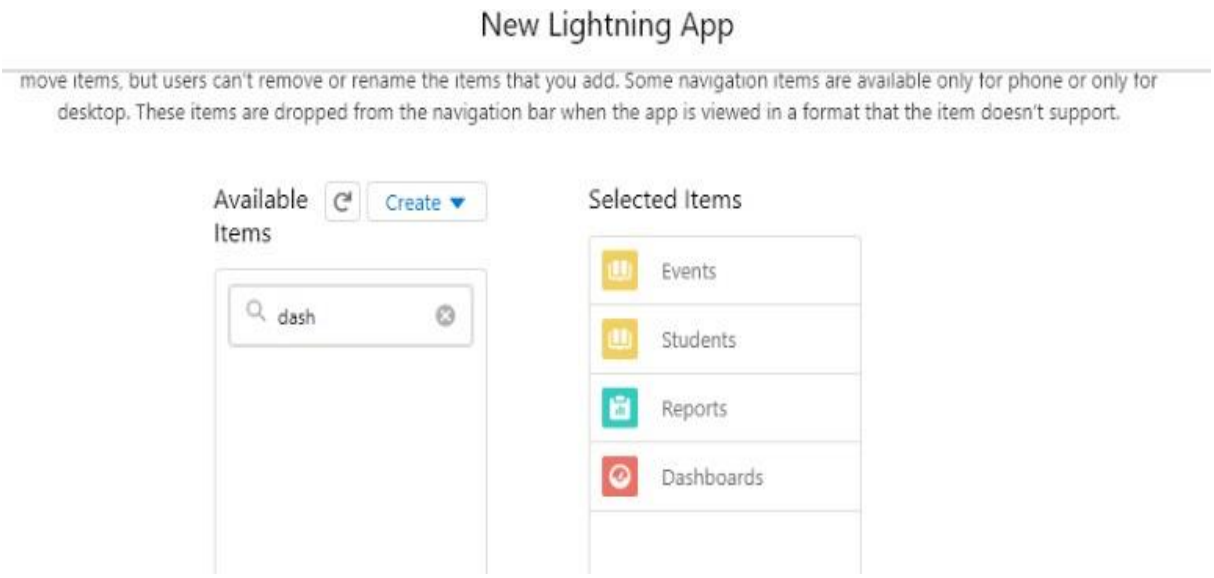
- Field Label: Event Name, Data Type: Text.
- Field Label: Event Time, Data Type: Master-Detail Relationship, Related to: Event. Sharing Setting: Read-Only. Leave the defaults and save.
- Field Label: Event Venue, Data Type: Look up Relationship, Related to: Event.
- Field Label: Branch, Data Type: Text, Make it as a Required Field.



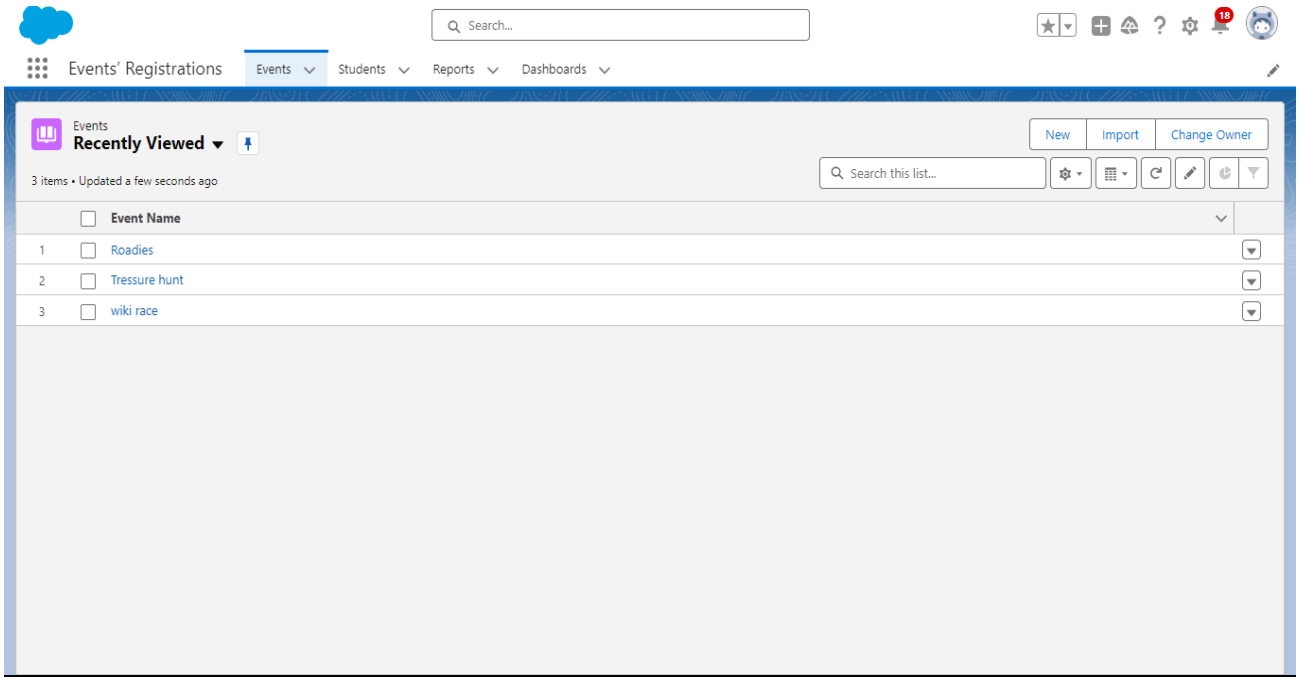
To create an application:

1. Go to “Setup” and type “App Manager” in Quick Find Box.
2. Click on “New Lightning App” to create a Lightning Application.
3. Name it as “Events’ Registrations”, give the description for your application.
4. Uploading Image and changing colours are optional, then click Next.
5. Navigation Style: Standard Navigation, click Next.
6. No need to add any Utility Bar, click Next.
7. Add the following Items: Events, Students, Reports and Dashboards, click Next.
8. Assign it to System Administrator Profile by selecting System Administrator and pressing right arrow and then click Save & Finish.


New Event
* = Required Information
Information
* Event Name
painting
Event Time
* Date
* Time
29/12/2023
12:00 pm
Date or time less than the current one.
* Event Venue
Auditorium
Owner
SHIVANAND KERUR
Cancel
Save & New
Save


New Lightning App
move items, but users can't remove or rename the items that you add. Some navigation items are available only for phone or only for desktop. These items are dropped from the navigation bar when the app is viewed in a format that the item doesn't support.
Available Items
Create
dash
Selected Items
Events
Students
Reports
Dashboards

Go to App Manager, select your application and select Events and click “New” to add some details to your application.

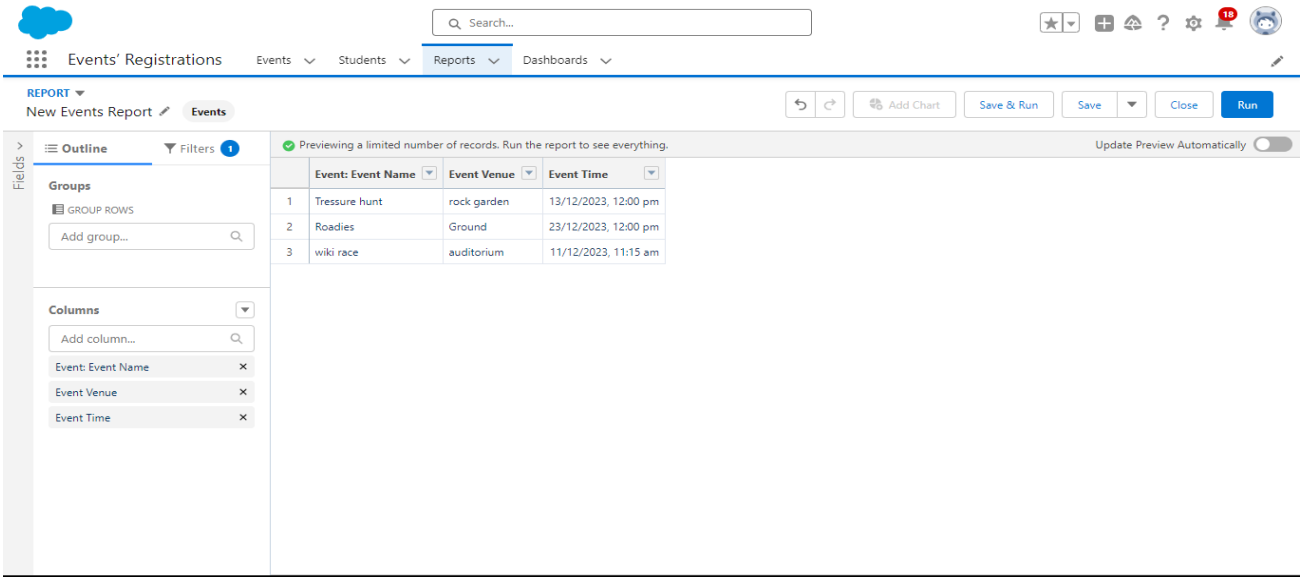


Make sure you will get an error message when you try to give the Event Time less than the current time and today’s date and also you can’t select events which are not there in the list.

Reports and Dashboards:

To Create an Event Report:

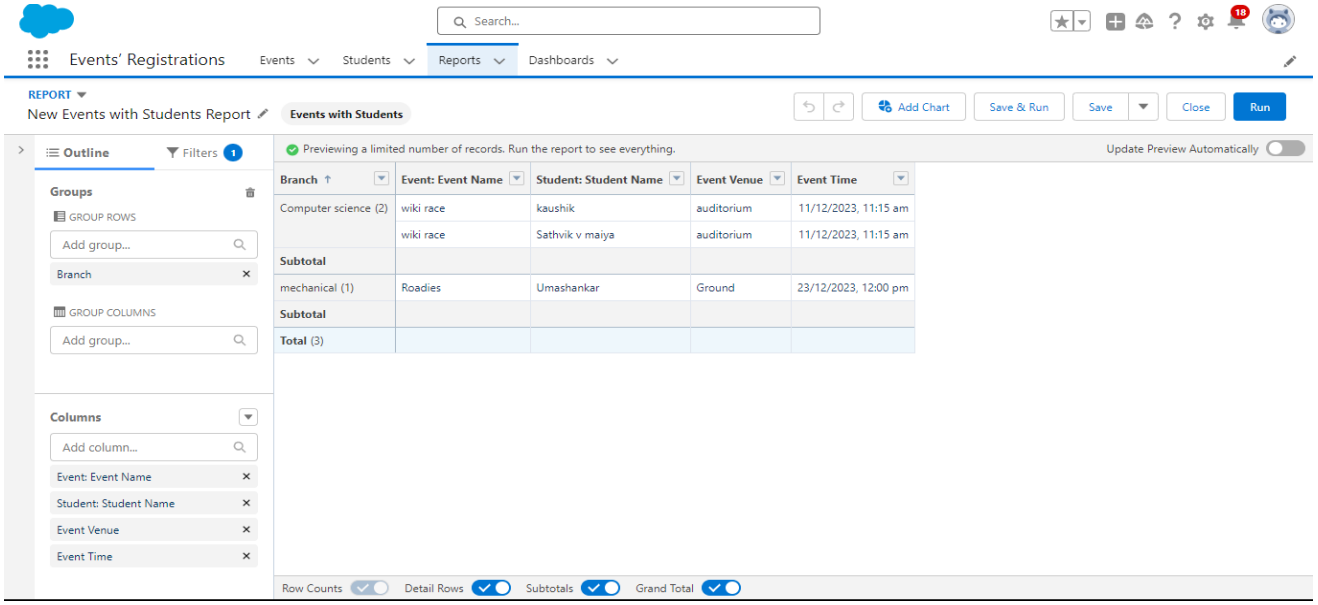
1. Go to “Reports tab” Click on “New Folder” And give it any name and Click Save.
2. Click on “New Report” and from search bar Search for “Events” and then select it then Click Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Events Report” and then select the folder which you have created.



6. Click Save and then Click Run

To Create a Students Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name and Click Save.
2. Click on “New Report” and from search bar Search for “Students with Events” and then select it then Click Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Events with Students Report” and then select the folder which you have created

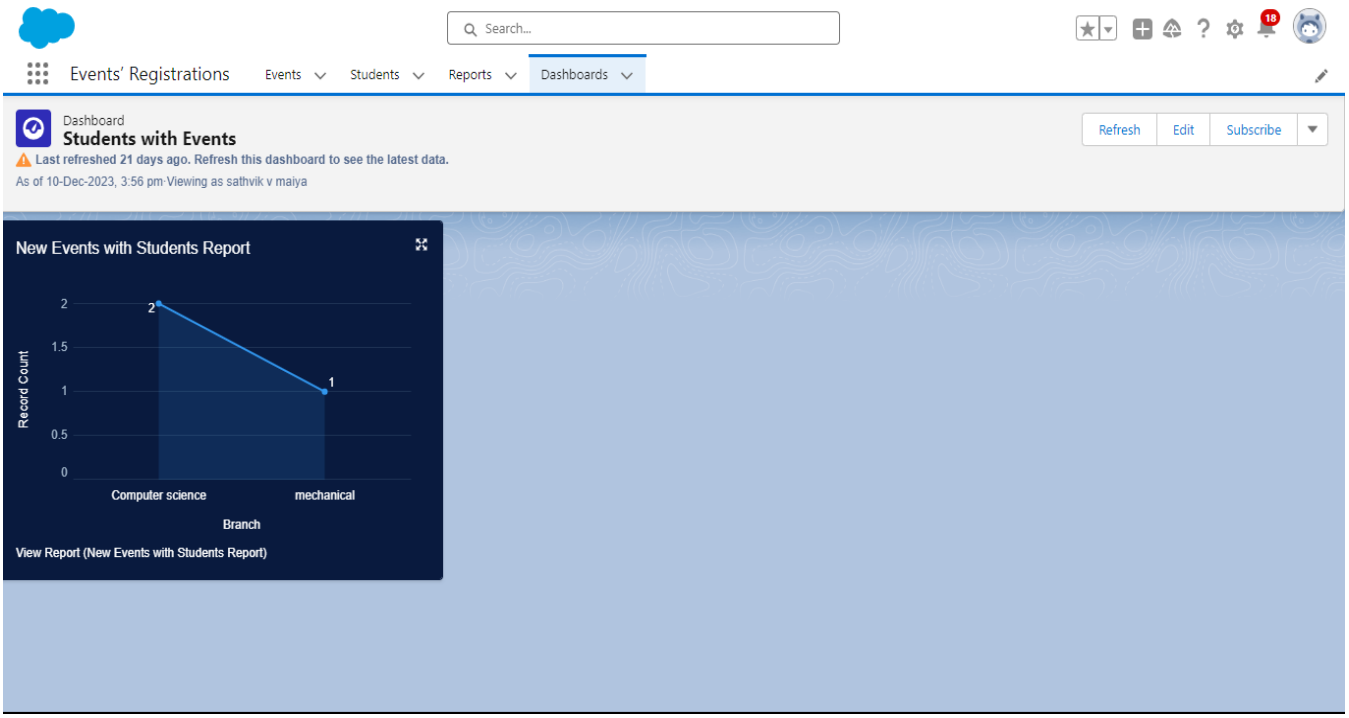


To Create an Event Dashboard:

1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Event Dashboard”
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run

To Create a Student1 Dashboard:

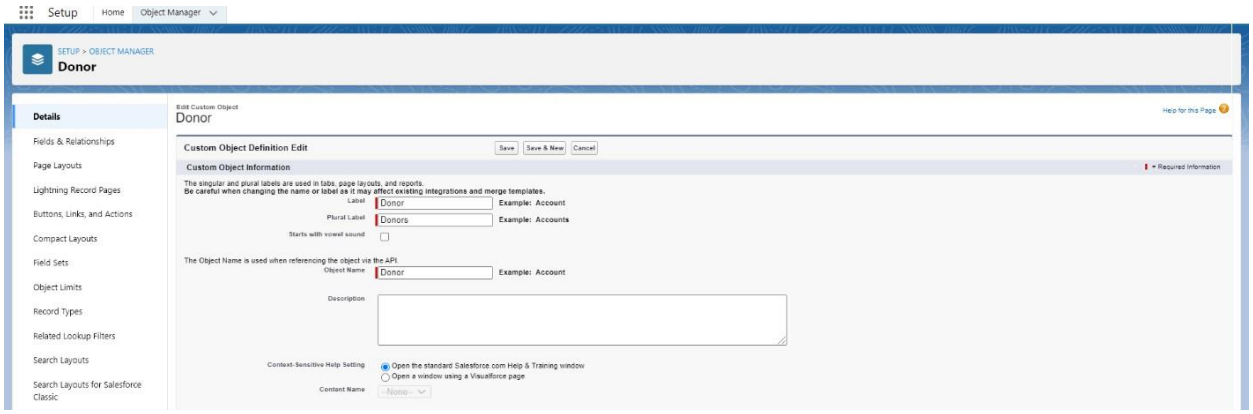
1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Students with Events Dashboard”
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run


Search...
Events' Registrations
Events
Students
Reports
Dashboards
Dashboard
Students with Events
Last refreshed 21 days ago. Refresh this dashboard to see the latest data.
As of 10-Dec-2023, 3:56 pm·Viewing as sathvik v maiya
Refresh
Edit
Subscribe
New Events with Students Report
Record Count
2
1.5
1
0.5
0
Computer science
mechanical
Branch
View Report (New Events with Students Report)

Q7) Develop Blood Donation registration app with an object which records donors name, age and blood group as parent relationship and another object containing haemoglobin level, donated or not details.

Launch your Salesforce Trailhead Playground by opening any module and Switch to Lightning Experience if you are currently in Salesforce Classic by clicking your picture in the right top corner and then click on "Switch to Lightning Experience"

Then go to Setup gear icon and click "Setup".



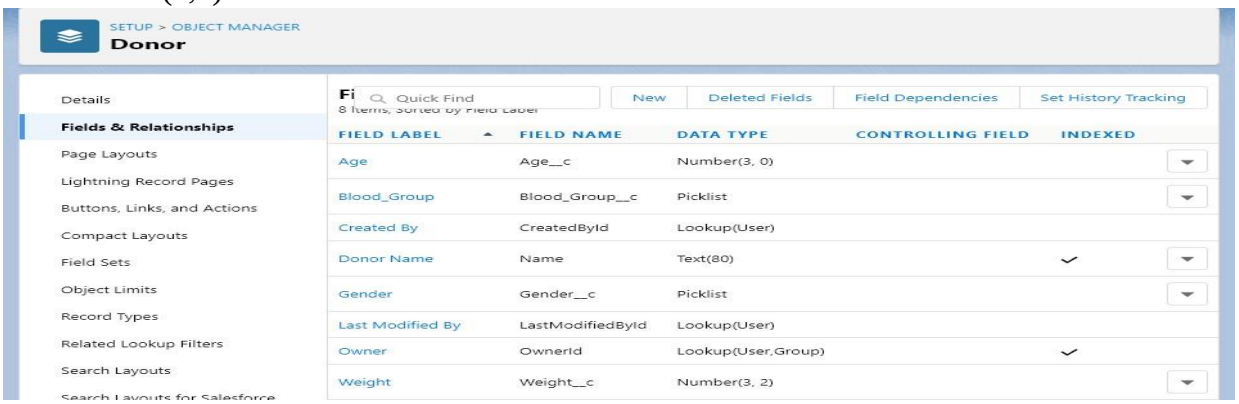
1. Click on "Object Manager" and click "Create > Custom Object" to create new Custom Object.
2. Name the object "Donor"
3. Allow Reports and Allow Search.
4. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object".
5. To create a Tab for the Object: Select any Tab Style for the object "Donor". Click Next, Next, leave the defaults and save.

To add fields to the Object:

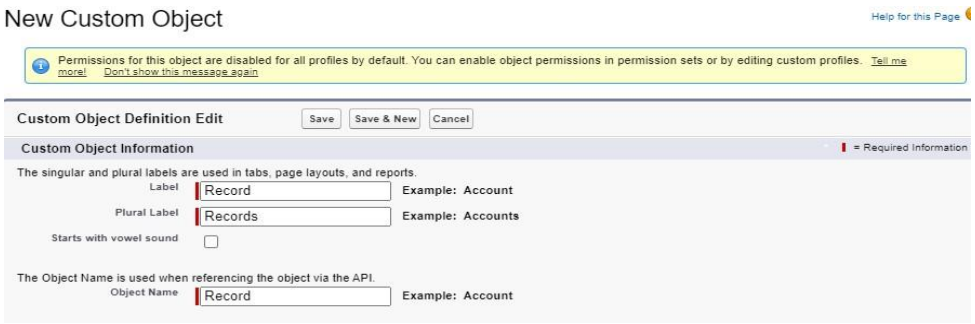
Go to "Fields & Relationships" option of Student object and Click "New".

Add the following fields one after the other:

- Field Label: Age, Data Type: Number (3,0).
- Field Label: Blood Group, Data Type: Picklist, click radio button in front of Enter values, with each value separated by a new line. Values are: A+ve, B+ve, A-ve, B-ve, O+ve, O-ve, AB+ve, AB-ve. Make it as Required Field and Restrict the values to the values in the Picklist.
- Field Label: Gender, Data Type: Picklist, click radio button in front of Enter values, with each value separated by a new line.Values are: Male, Female, Others. • Field Label: Weight, Data Type: Number (3,2).



Create one more object to store Collection details:



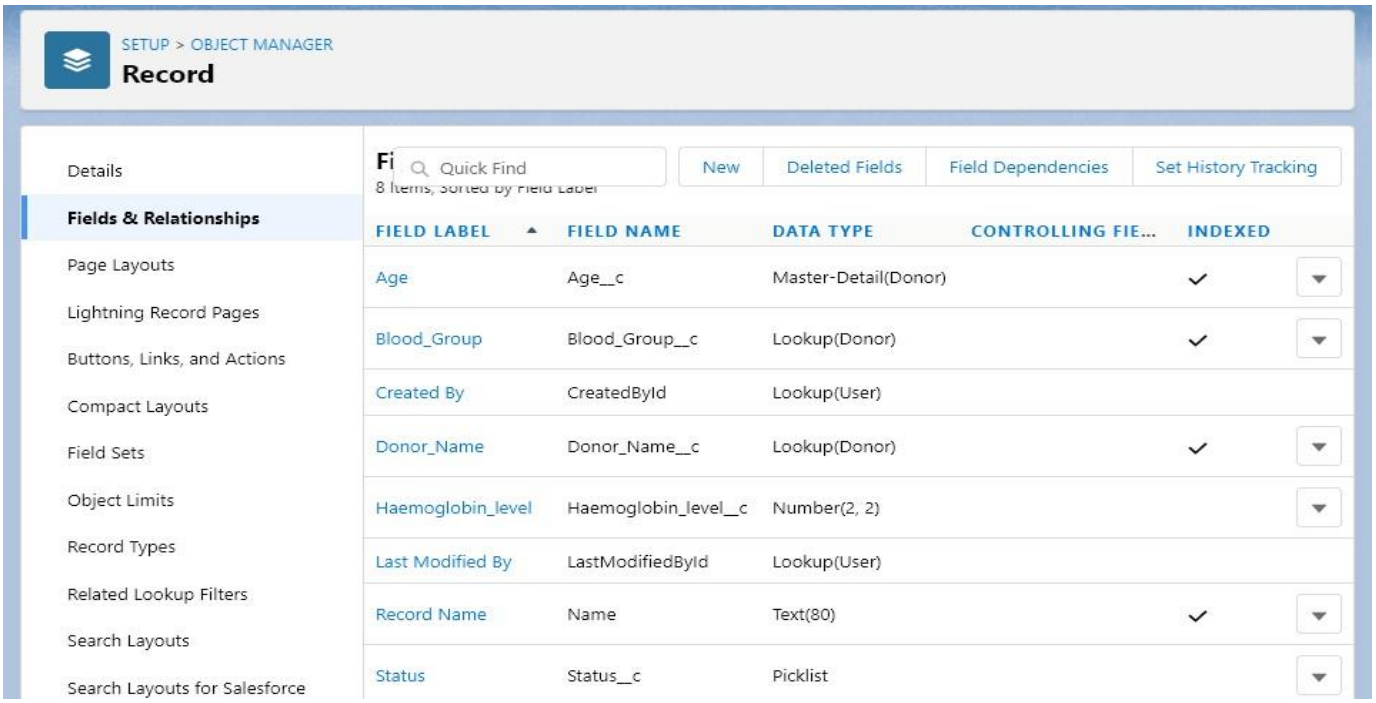
1. Name the Object "Record".
2. Allow Reports and Allow Search.
3. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object"
4. Create a Tab for the Object.

To add fields to the Object:

Go to "Fields & Relationships" option of Student object and Click "New".

Add the following fields one after the other:

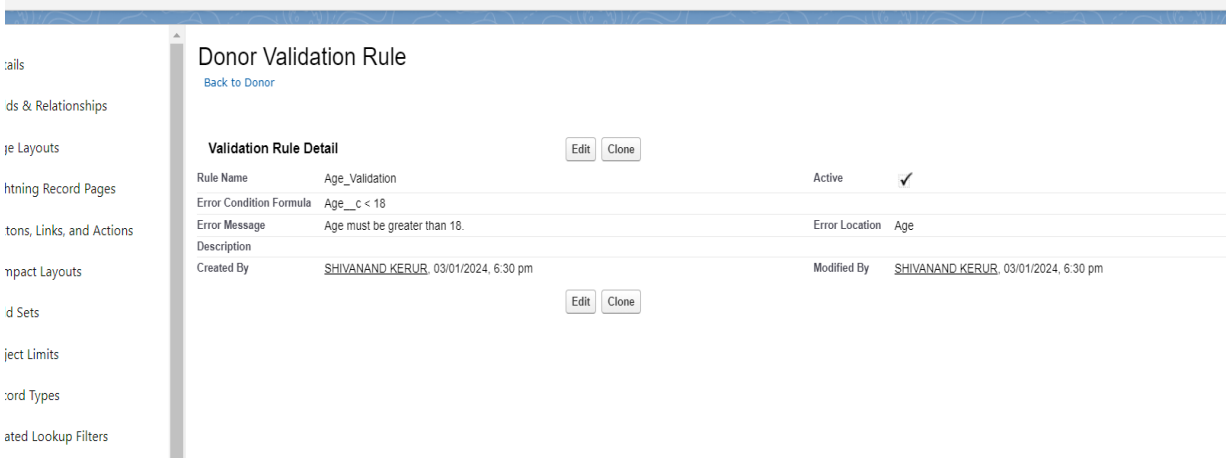
- Field Label: Haemoglobin level, Data Type: Number (2,2).
- Field Label: Donor Name, Data Type: Look up Relationship, Related to: Donor.
- Field Label: Blood Group, Data Type: Look up Relationship, Related to: Donor.
- Field Label: Age, Data Type: Master-Detail Relationship, Related to: Donor. Sharing Setting: Read-Only. Leave the defaults and save.
- Field Label: Status, Data Type: Picklist, Values: Donated, Not Donated, Make it as a Required Field



Validation Rules:

Age Validation:

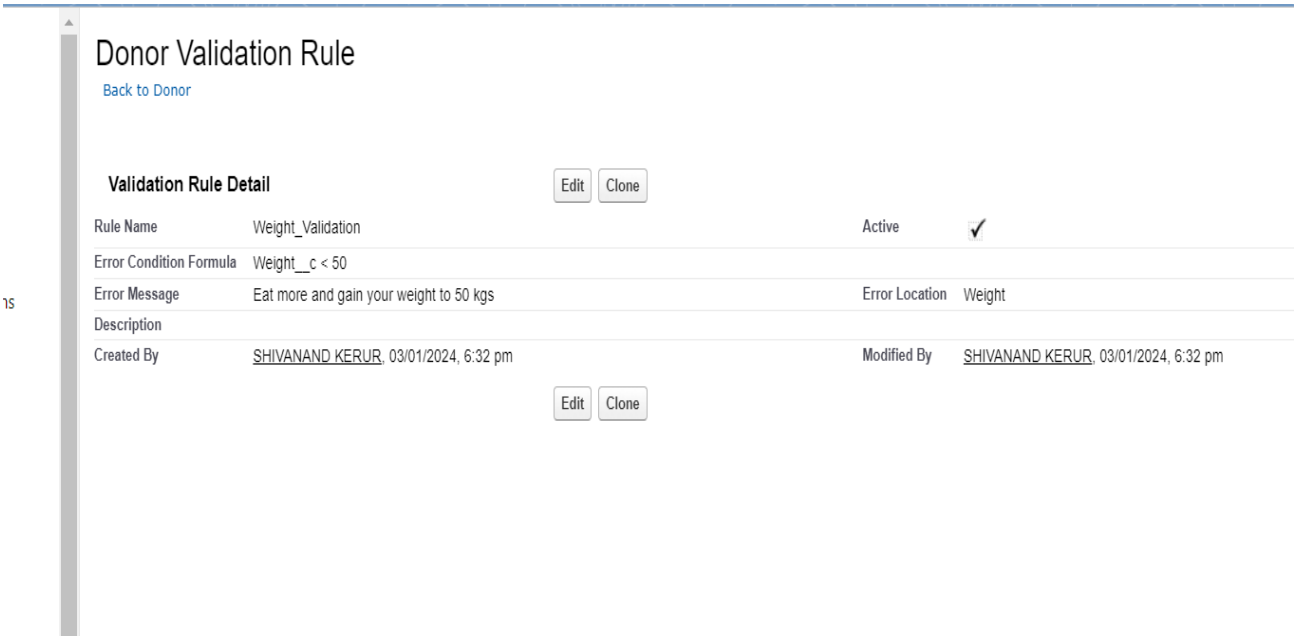
1. To add a rule to the Donor age so that it is greater than 18 years:
2. Go to Validation Rule of Donor Object and click “New”
3. Name it as “Age Validation”.
4. Error Condition Formula: age__c < 18.
5. Error Message: Age must be greater than 18.
6. Error Location: Field – Age.
7. Click Save



Weight Validation:

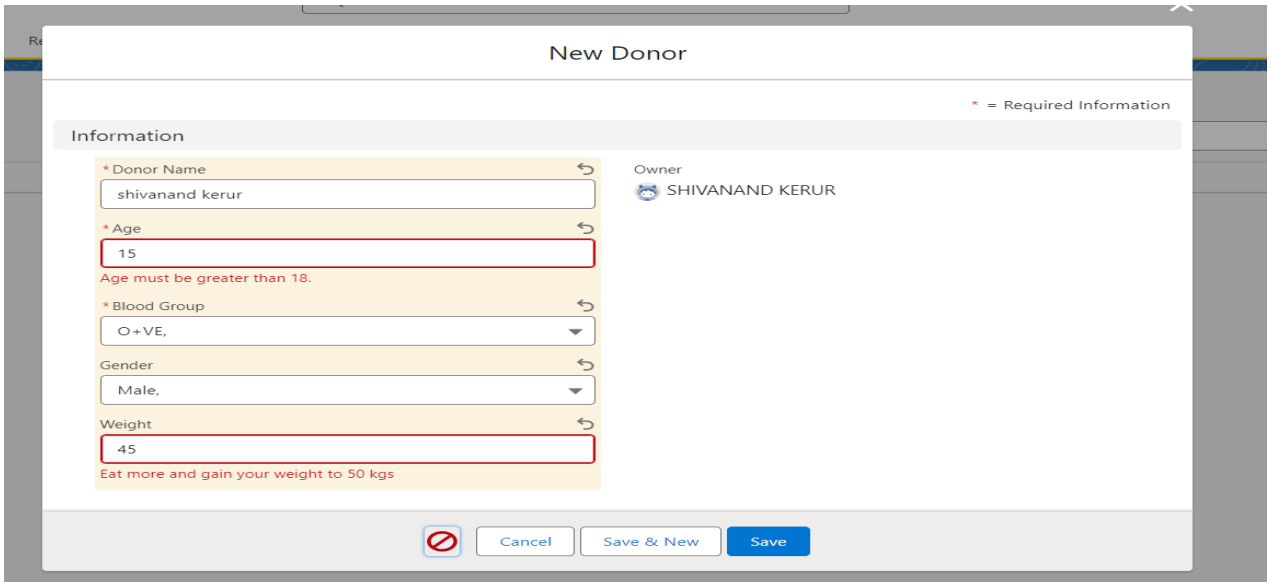
To add a rule to the Donor Weight so that it Should be greater than 50:

1. Go to Validation Rule of Donor Object and click “New”
2. Name it as “Weight Validation”.
3. Error Condition Formula: weight__c < 50.
4. Error Message: Eat more and gain your weight to 50 kgs.
5. Error Location: Field –weight.
6. Click Save



To create an application:

1. Go to “Setup” and type “App Manager” in Quick Find Box.
2. Click on “New Lightning App” to create a Lightning Application.
3. Name it as “Blood Donation”, give the description for your application.
4. Uploading Image and changing colours are optional, then click Next.
5. Navigation Style: Standard Navigation, click Next.
6. No need to add any Utility Bar, click Next.
7. Add the following Items: Donors, Records, Reports and Dashboards, click Next.
8. Assign it to System Administrator Profile by selecting System Administrator and pressing right arrow and then click Save & Finish.



Go to App Manager, select your application and select Donors and click “New” to add some details to your application.

Make sure you can’t donate when your age is less than 18 years and when your weight is not 50 kgs.

Reports and Dashboards:

To Create a Donor Report:

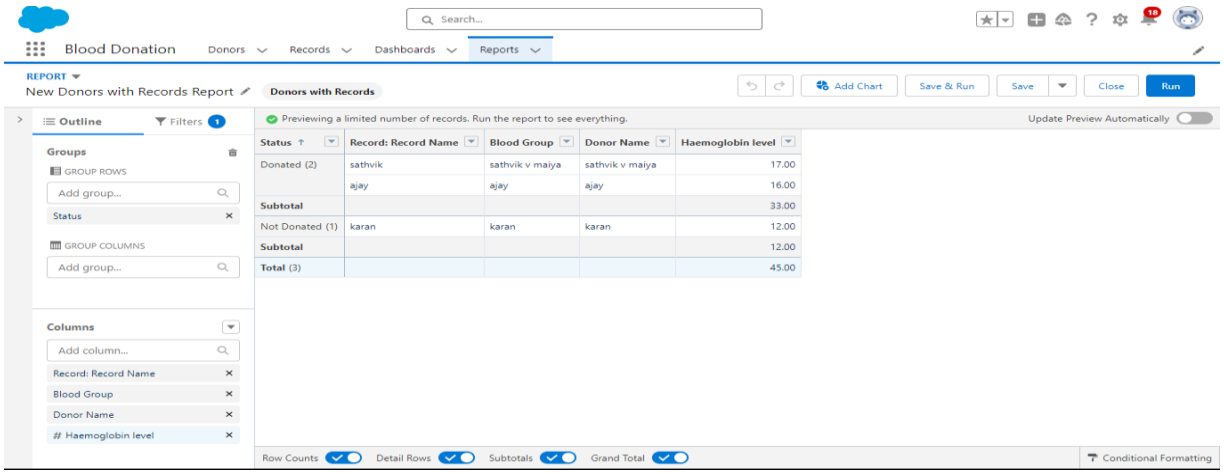
1. Go to “Reports tab” Click on “New Folder” And give it any name then click on Save.

2. Click on “New Report” and from search bar Search for “Donors” and then select it and then click on Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Donor Report” and then select the folder which you have created.
6. Click Save and then Click Run.

To Create a Records Report:

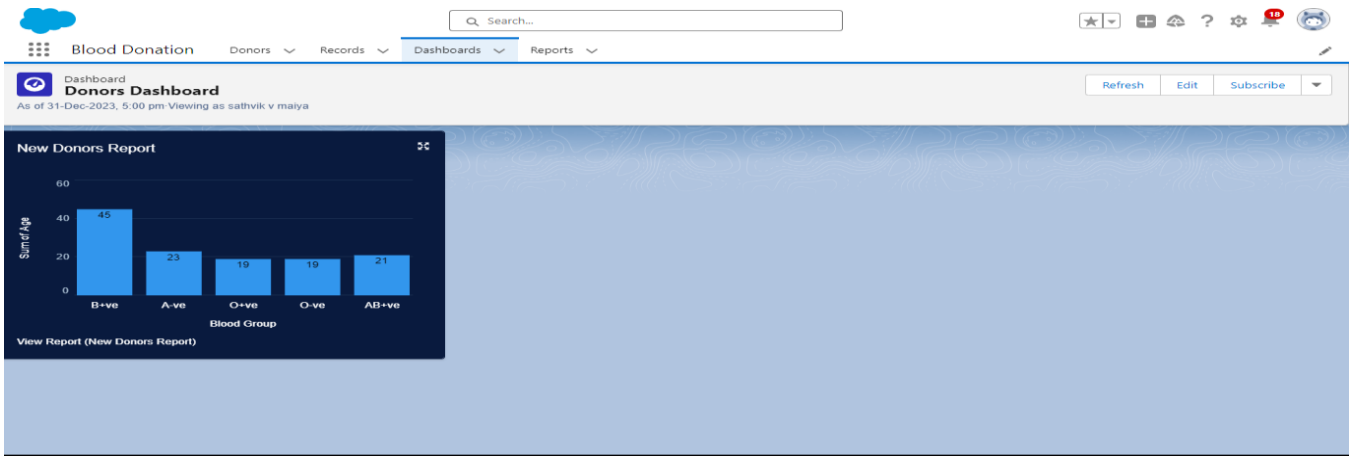
1. Go to “Reports tab” Click on “New Folder” And give it any name then click on Save.
2. Click on “New Report” and from search bar Search for “Donors with records” and then select it and then click on Continue.

3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Donors with record Report” and then select the folder which you have created.
6. Click Save and then Click Run.



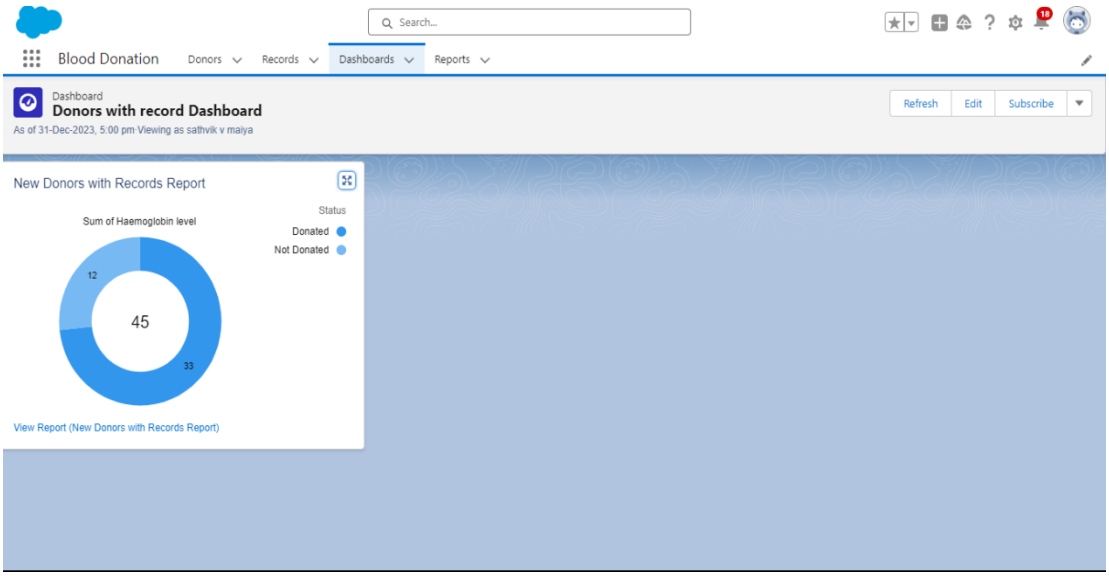
To Create a Donor Dashboard:

1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Donors Dashboard” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run



To Create a Records Dashboard:

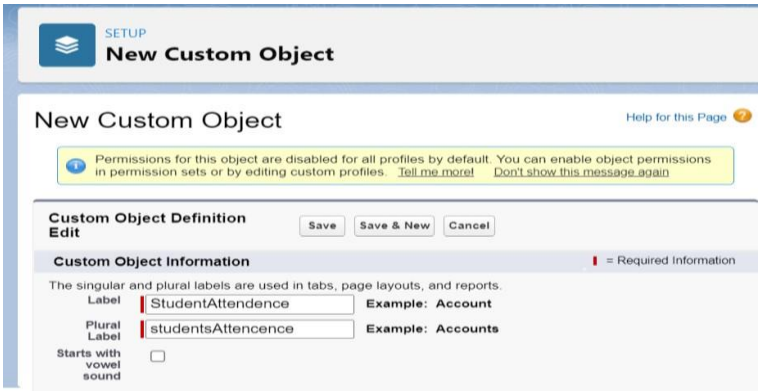
1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Donors with record Dashboard” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run


Search...
Blood Donation
Donors
Records
Dashboards
Reports
Dashboard
Donors with record Dashboard
As of 31-Dec-2023, 5:00 pm·Viewing as sathvik v maiya
Refresh
Edit
Subscribe
New Donors with Records Report
Sum of Haemoglobin level
Status
Donated
Not Donated
12
45
33
View Report (New Donors with Records Report)

Q8) Develop Attendance maintenance app with an object to record student details and attendance and a provide a link to college websites' results webpage.

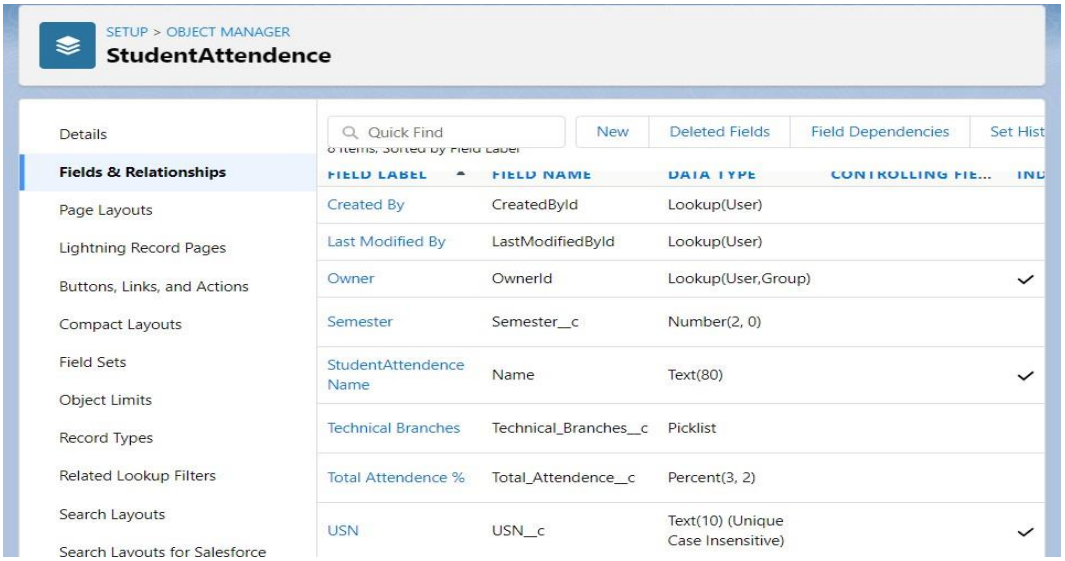
Launch your Salesforce Trailhead Playground by opening any module and Switch to Lightning Experience if you are currently in Salesforce Classic by clicking your picture in the right top corner and then click on "Switch to Lightning Experience"

Then go to Setup gear icon and click "Setup".



1. Click on "Object Manager" and click "Create > Custom Object" to create new Custom Object.
2. Name the object "Student".
3. Allow Reports and Allow Search.
4. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object".
5. To create a Tab for the Object: Select any Tab Style for the object "Student". Click Next, Next, leave the defaults and save.

To add fields to the Object:



Go to "Fields & Relationships" option of Student object and Click "New".

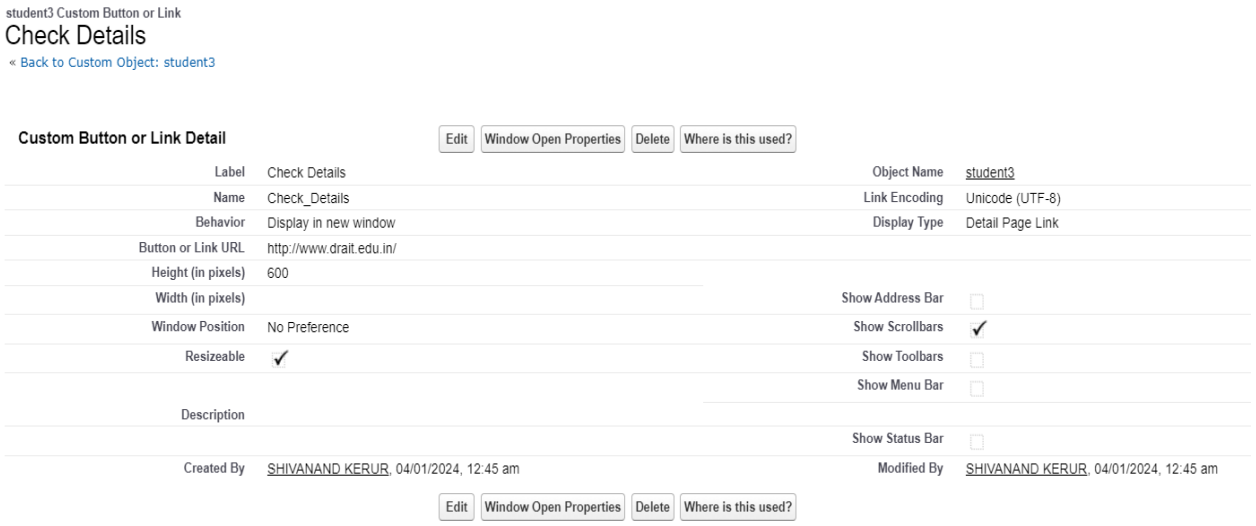
Add the following fields one after the other:

- Field Label: USN (Length 10), Data Type: Text, provide an example USN as Help Text, make it as Required Field and Don't allow Duplicate Values and make it as Case Insensitive.

- Field Label: Technical Branches, Data Type: Picklist, click radio button in front of Enter values, with each value separated by a new line. Values are: CSE, ISE, TCE, EEE, EC etc. Make it as Required Field and Restrict the values to the values in the picklist.
- Field Label: Total Attendance %, Data Type: Percent (3,2).
- Field Label: Semester, Data Type: Number (2,0).

To include Custom Links:

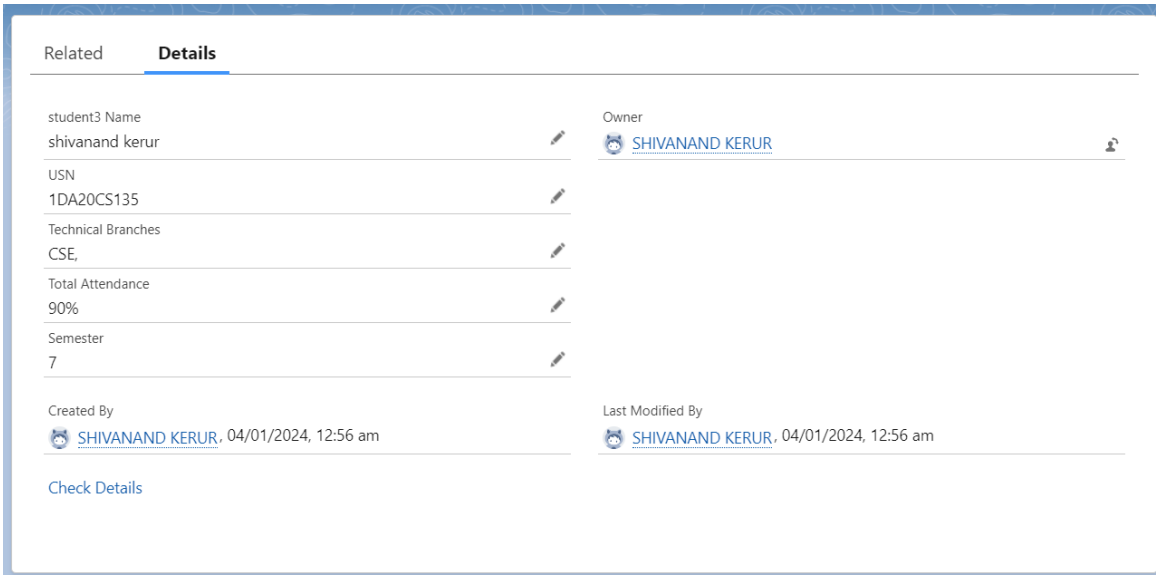
1. Go to “Buttons, Links and Actions” of “Student” Object and click “New Button or Link”.
2. Name it as “Check Details”.
3. Select the radio button “Detail Page Link” as it is a website link.
4. Behaviour: Display in new window.
5. Content Source: URL.
6. Field Type: Student.
7. In the empty space provided, type http://www.drait.edu.in/ It is a link which redirects to the income tax calculation website.
8. Link Encoding: Unicode (UTF-8).
9. Click Save
10. Go to Page Layout, Click Student Layout.
11. Click Custom Links, Drag and drop the “Check Details” link in the Custom Link area.
12. Click save.

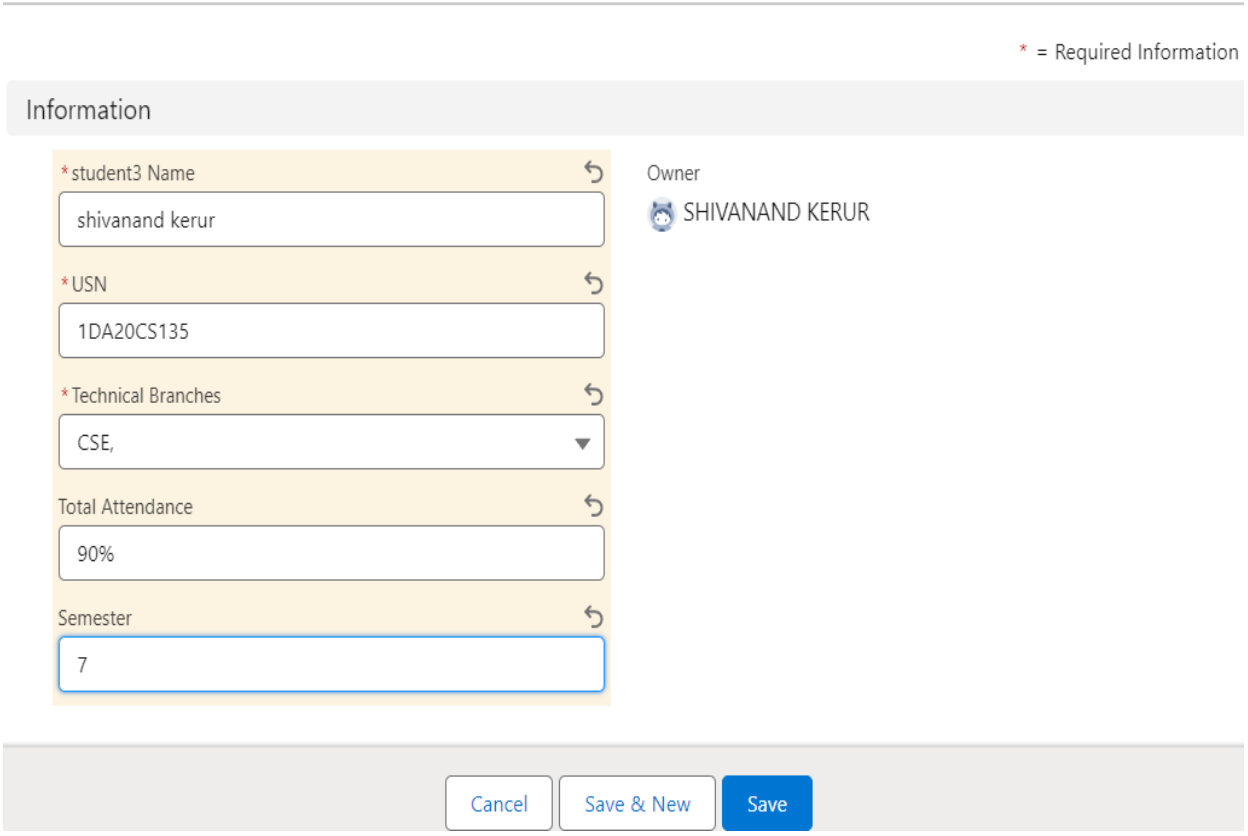


To create an application:

1. Go to “Setup” and type “App Manager” in Quick Find Box.
2. Click on “New Lightning App” to create a Lightning Application.
3. Name it as “Attendance Management”, give the description for your application.
4. Uploading Image and changing colours are optional, then click Next.
5. Navigation Style: Standard Navigation, click Next.
6. No need to add any Utility Bar, click Next.
7. Add the following Items: Student, Records, Reports and Dashboards, click Next.
8. Assign it to System Administrator Profile by selecting System Administrator and pressing right arrow and then click Save & Finish.

Go to App Manager, select your application and select Student and click “New” to add some details to your application



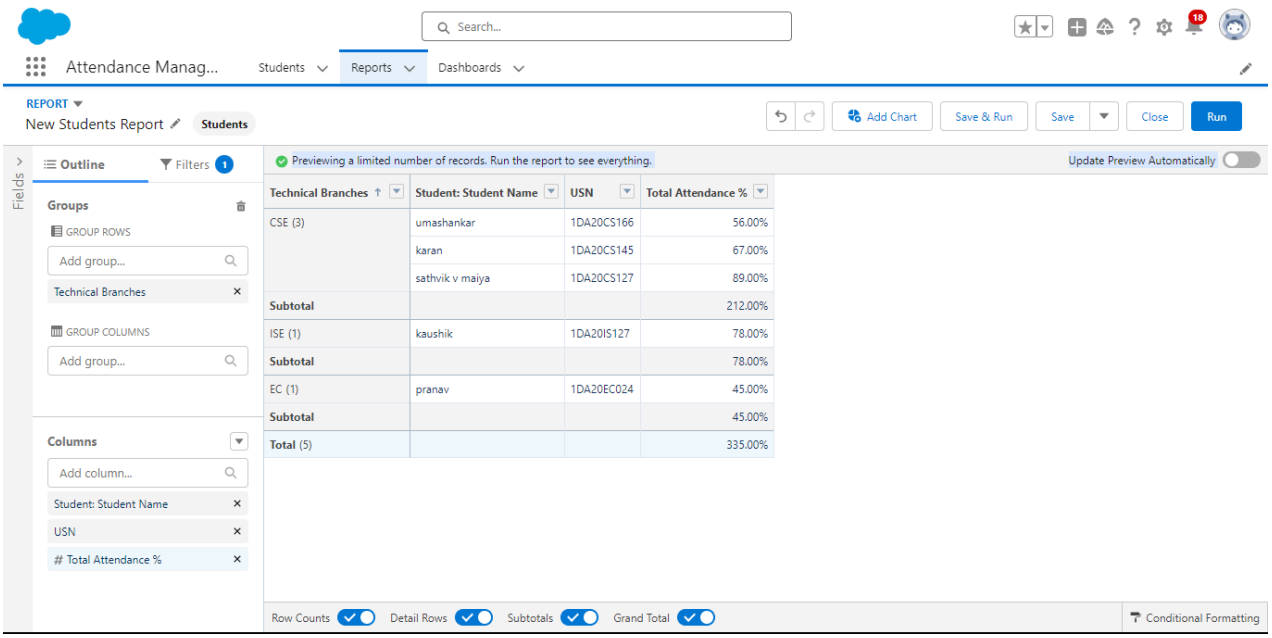


Make Sure you are redirected to the College Website When you click on the Check Details Link.

Reports and Dashboards:

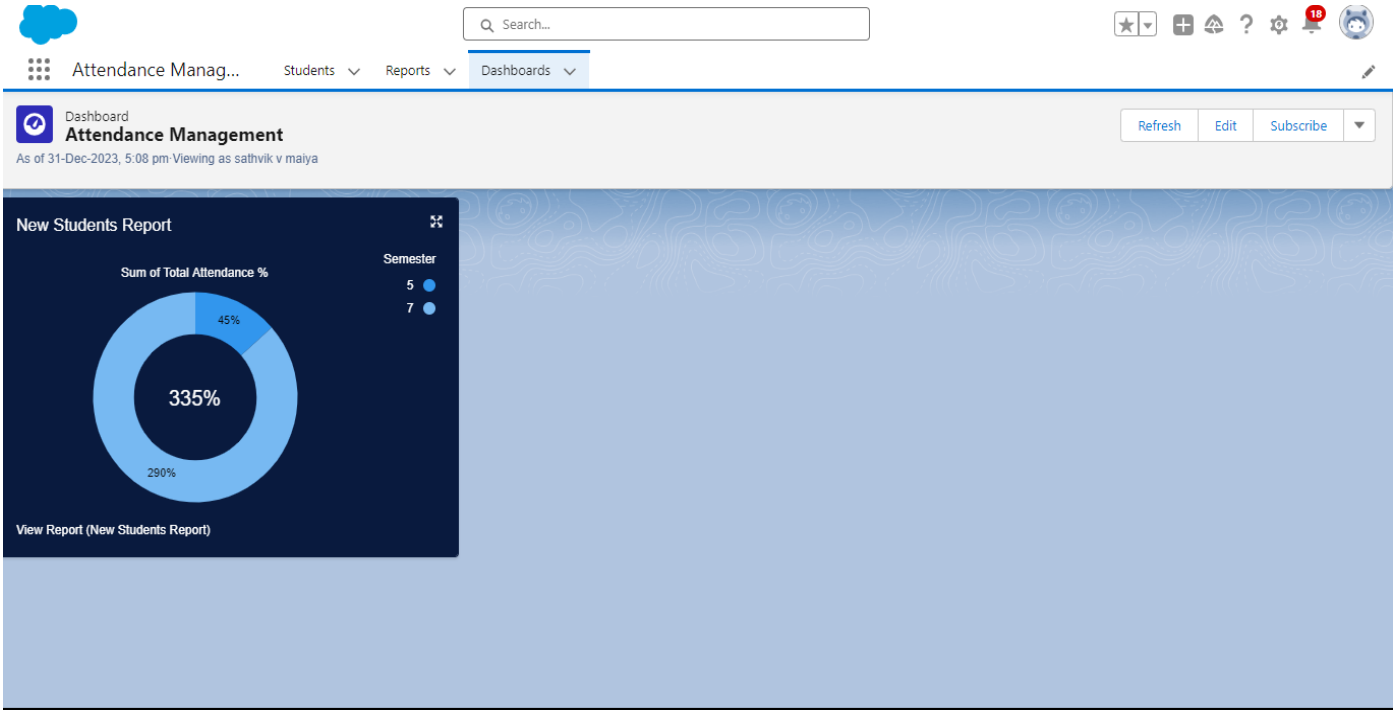
To Create a Student Report:

1. Go to "Reports tab" Click on "New Folder" And give it any name then click Save
2. Click on "New Report" and from search bar Search for "Attendance Management" and then select it then click Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in "Add groups" otherwise it is optional.
5. Click on save and name the report as "New Students Report" and then select the folder which you have created.
6. Click Save and then Click Run.



To Create a Student Dashboard:

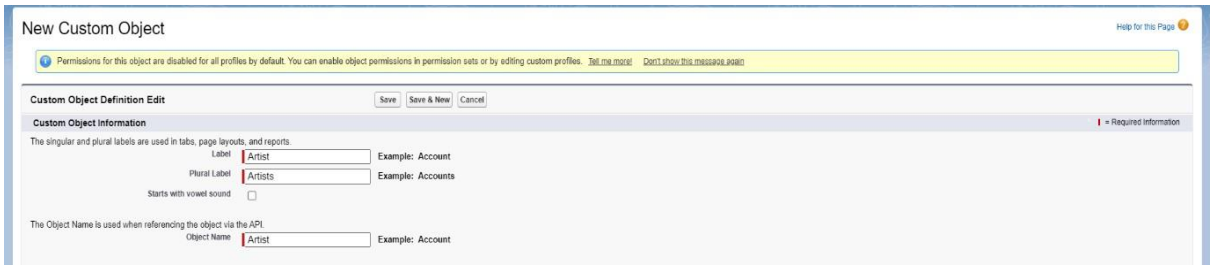
1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Attendance Management” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run



Q9) Create a web application with objects to maintain database of an art gallery which contains objects like artists, arts, inventory and provide a link to any of the art gallery website.

Launch your Salesforce Trailhead Playground by opening any module and Switch to Lightning Experience if you are currently in Salesforce Classic by clicking your picture in the right top corner and then click on "Switch to Lightning Experience".

Then go to Setup gear icon and click "Setup".



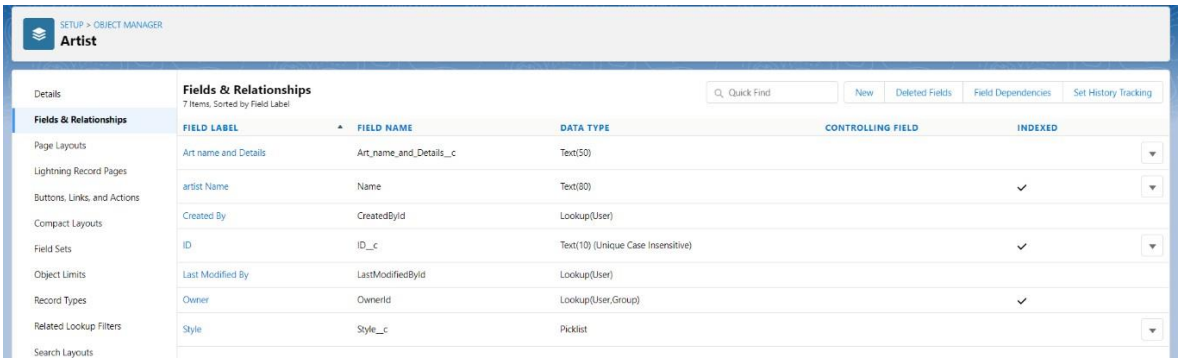
1. Click on "Object Manager" and click "Create > Custom Object" to create new Custom Object.
2. Name the object "Artist".
3. Allow Reports and Allow Search.
4. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object".
5. To create a Tab for the Object: Select any Tab Style for the object "Artist". Click Next, Next, leave the defaults and save.

To add fields to the Object:

Go to "Fields & Relationships" option of Student object and Click "New".

Add the following fields one after the other:

- Field Label: ID (Length 10), Data Type: Text, provide an example ID as Help Text, make it as required Field, don't allow Duplicate Values, make it as Case Insensitive and Set this field as the unique record identifier from an external system.
- Field Label: Art Name and Details, Data Type: Text, Make it as a required field.
- Field Label: Style, Data Type: Picklist. Values: Pencil Sketching, craft design, free hand, Human portrait, drawing, painting etc. Make it as Required Field and restrict values to the values in the picklist



Create one more object to store Collection details:

1. Name the Object "Art".

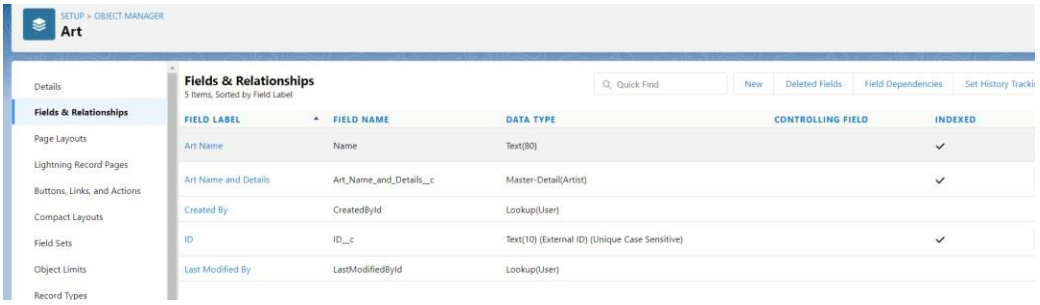
2. Allow Reports and Allow Search
3. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object".
4. Create a Tab for the Object.

To add fields to the Object:

Go to "Fields & Relationships" option of Student object and Click "New".

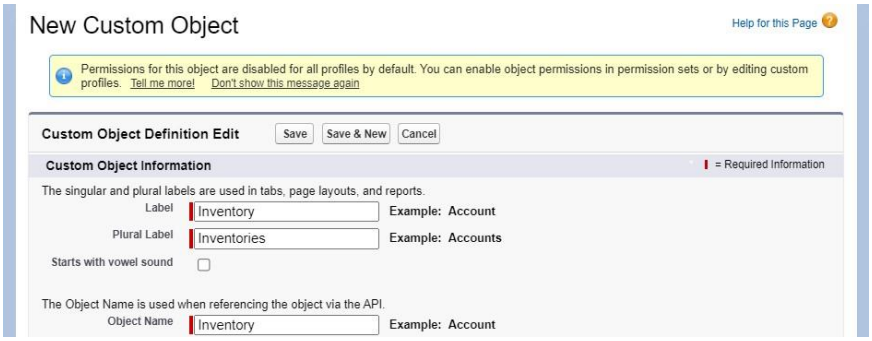
Add the following fields one after the other:

- Field Label: Art Name and Details, Data Type: Master-Detail Relationship, Related to: Artist. Sharing Setting: Read-Only. Leave the defaults and save.
- Field Label: ID (Length 10), Data Type: Text, provide an example ID as Help Text, make it as required Field, don't allow Duplicate Values, make it as Case Insensitive and Set this field as the unique record identifier from an external system.



Create one more object to store inventory details:

1. Name the Object "Inventory".
2. Allow Reports and Allow Search.
3. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object"
4. Create a Tab for the Object.



To add fields to the Object:

Go to "Fields & Relationships" option of Student object and Click "New".

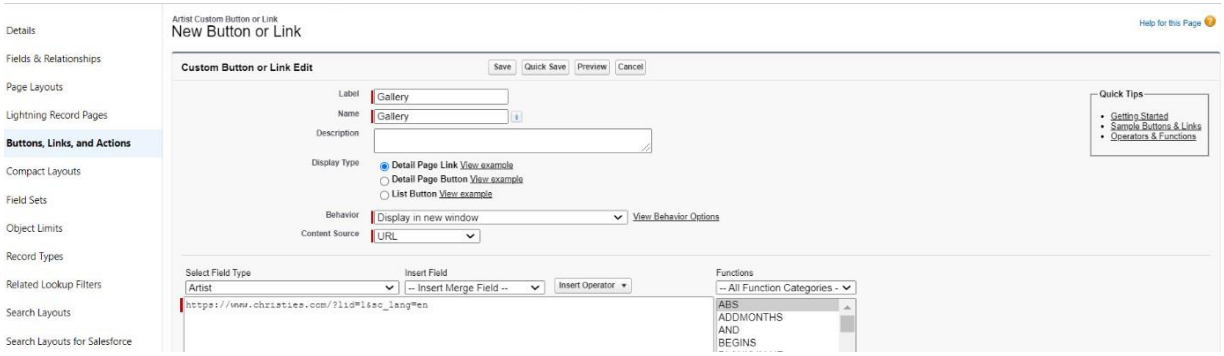
Add the following fields one after the other:

- Field Label: Quantity, Data Type: Number. Make it as a required field.

To give a link to any art gallery website:

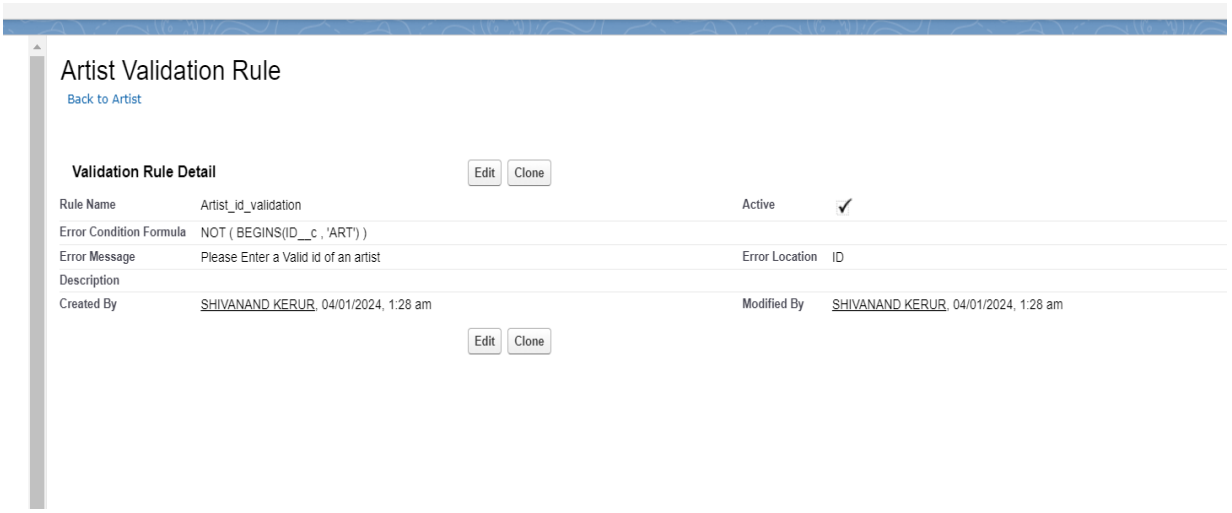
1. Go to "Buttons, Links and Actions" of Art Object and click "New Button or Link".

2. Name it as “Gallery”.
3. Select the radio button “Detail Page Link” as it is a website link.
4. Behaviour: Display in new window.
5. Content Source: URL 27. Field Type: Gallery.
6. In the empty space provided, type, http://www.aakritiartgallery.com/ Link
7. Encoding: Unicode (UTF-8).
8. Click Save.
9. Go to Page Layout, Click Art Layout.
10. Click Custom Links, Drag and drop the “Gallery” link in the Custom Link area.
11. Click save.



To add a rule to the Artist id so that it should take valid id:

1. Go to Validation Rule of Artist Object and click “New”.
2. Name it as “Artist id validation”.
3. Error Condition Formula: NOT(BEGINS(ID__c,’ART’)).
4. Error Message: Please Enter a Valid id of an artist.
5. Error Location: Field –id.
6. Click Save



To create an application:

1. Go to “Setup” and type “App Manager” in Quick Find Box.
2. Click on “New Lightning App” to create a Lightning Application.
3. Name it as “Art Gallery Database”, give the description for your application.
4. Uploading Image and changing colours are optional, then click Next.
5. Navigation Style: Standard Navigation, click Next.
6. No need to add any Utility Bar, click Next.

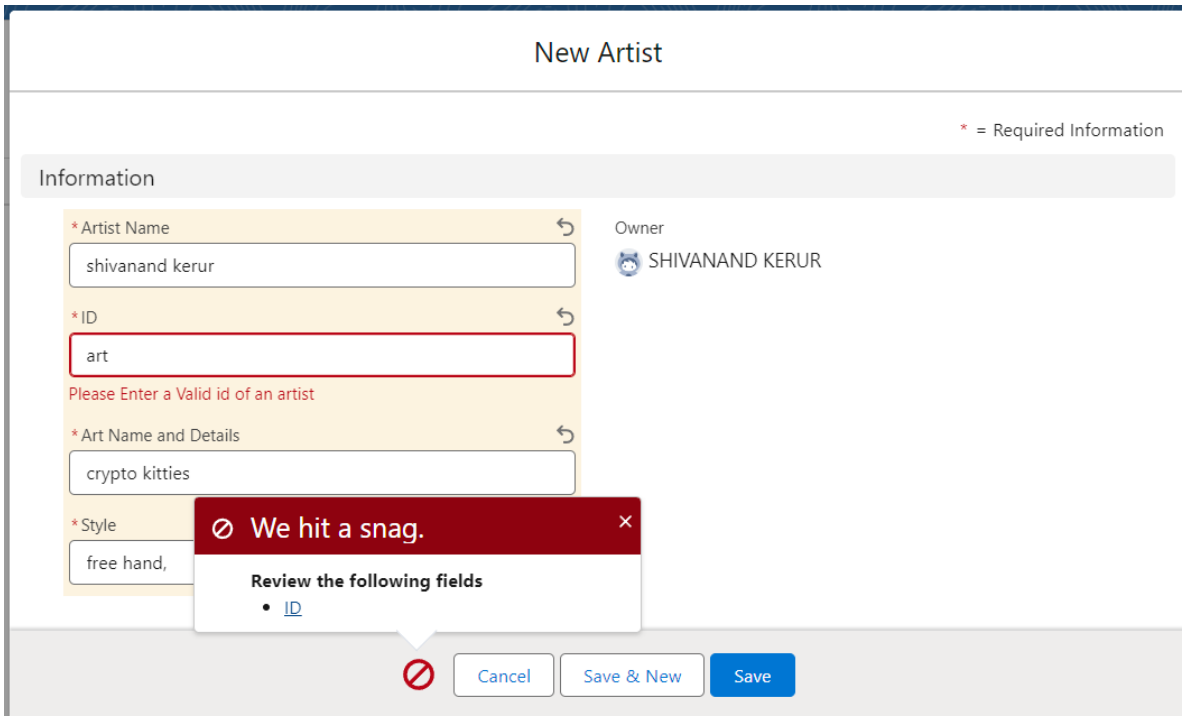
7. Add the following Items: Artists, Arts, Inventories, Reports and Dashboards, click Next.
8. Assign it to System Administrator Profile by selecting System Administrator and pressing right arrow and then click Save & Finish.

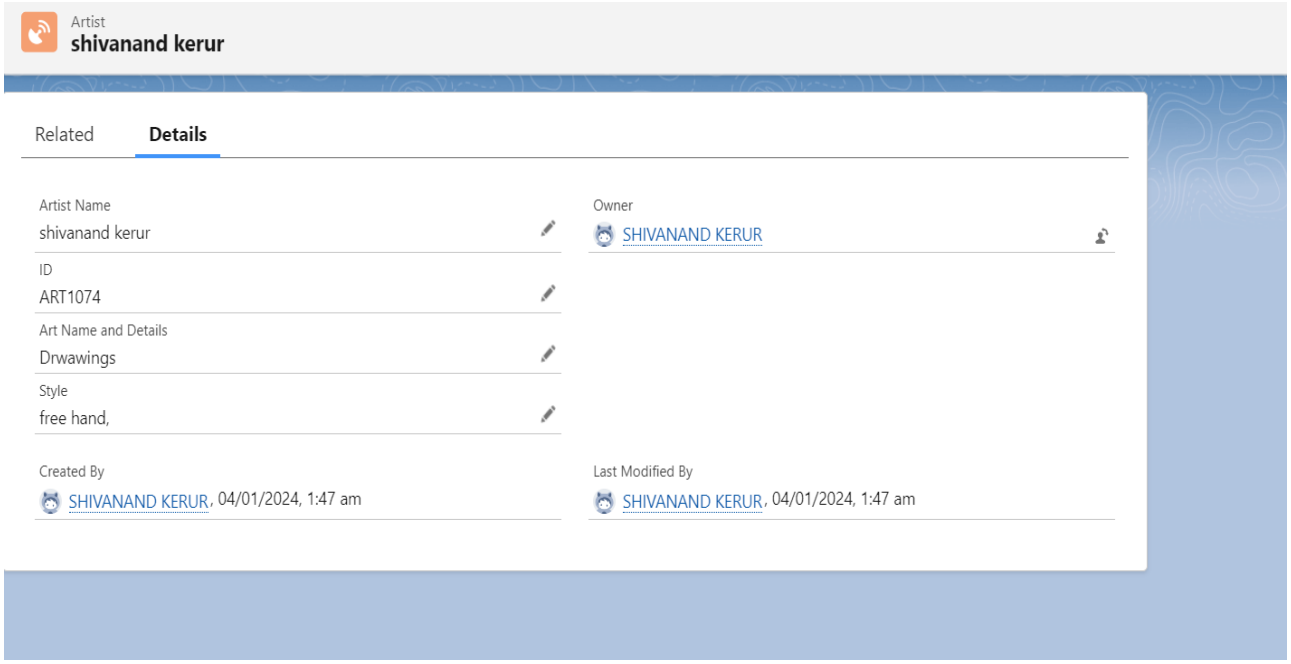
Go to App Manager, select your application and select Faculties and click “New” to add some details to your application.

Click the entry you added, go to details.

Press the “Gallery” link to check the details.

Click OK so that it will redirect you to the website. Make sure it should an error when an invalid id is given.

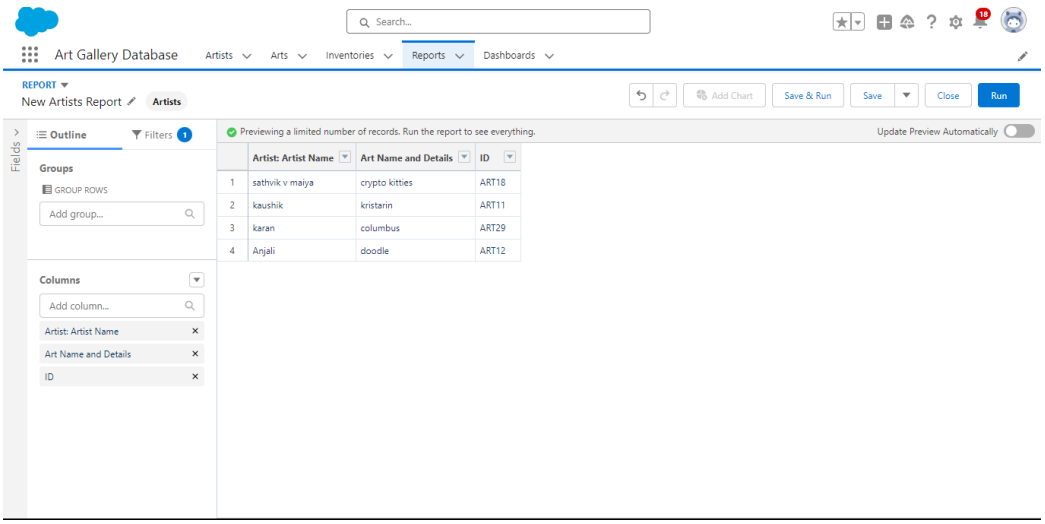




Reports and Dashboards:

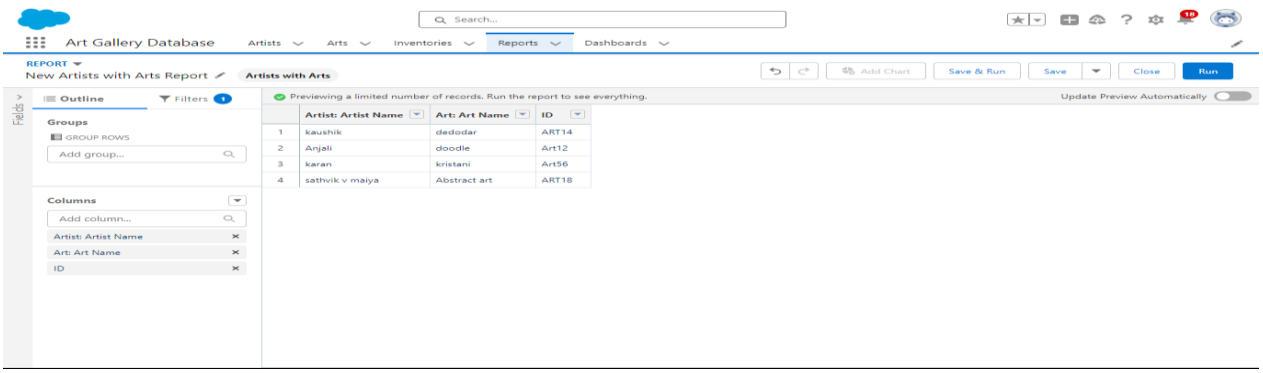
To Create an Artists Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name and Click Save.
2. Click on “New Report” and from search bar Search for “Artists” and then select it and then Click Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Artists Report” and then select the folder which you have created.
6. Click Save and then Click Run



To Create an Arts Report:

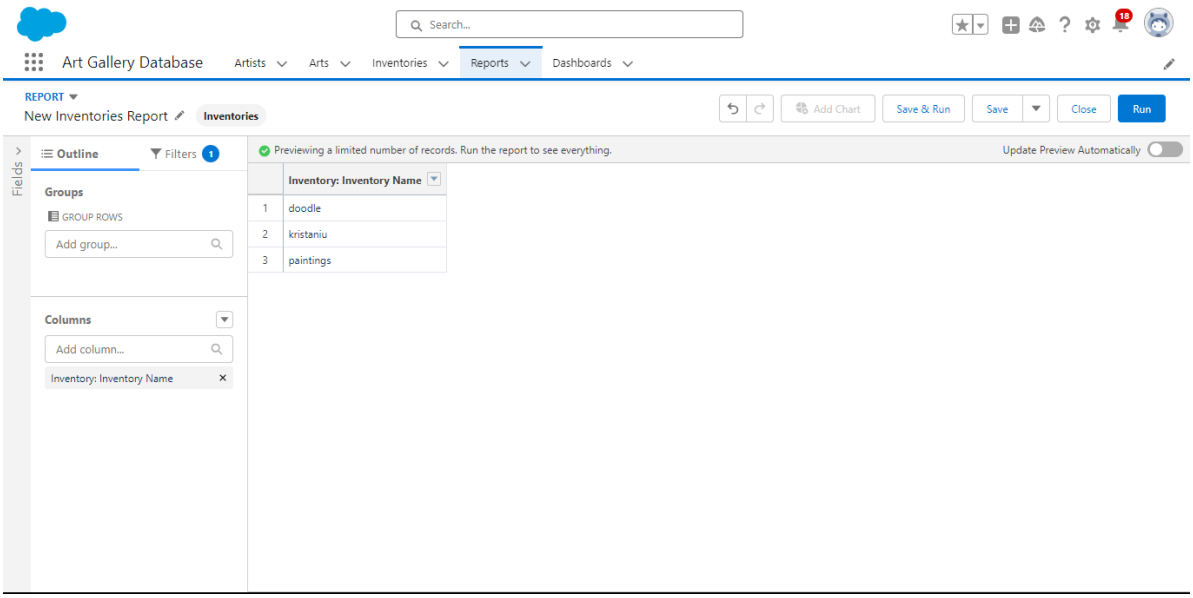
1. Go to “Reports tab” Click on “New Folder” And give it any name.
2. Click on “New Report” and from search bar Search for “Artists with Arts” and then select it.
3. Add the required Columns to get the Complete Entered data.



4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Artists with Arts Report” and then select the folder which you have created.
6. Click Save and then Click Run

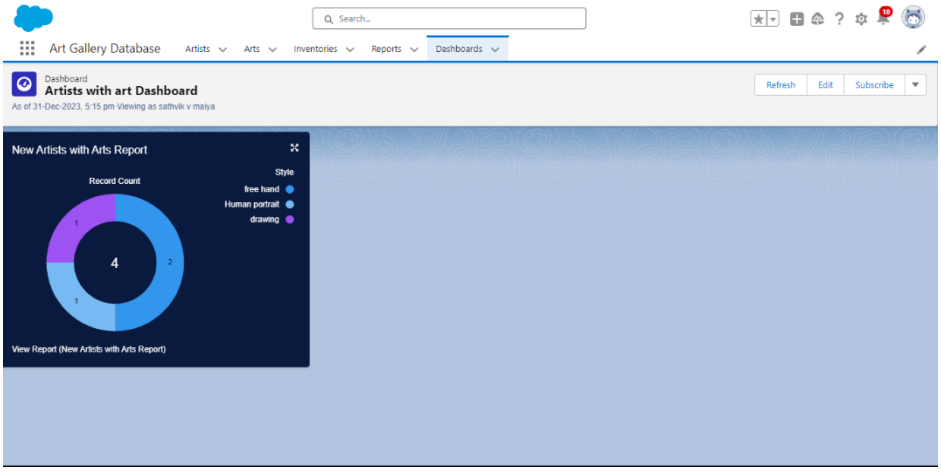
To Create an Inventory Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name.
2. Click on “New Report” and from search bar Search for “Inventories” and then select it.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Inventories Report” and then select the folder which you have created.
6. Click Save and then Click Run.



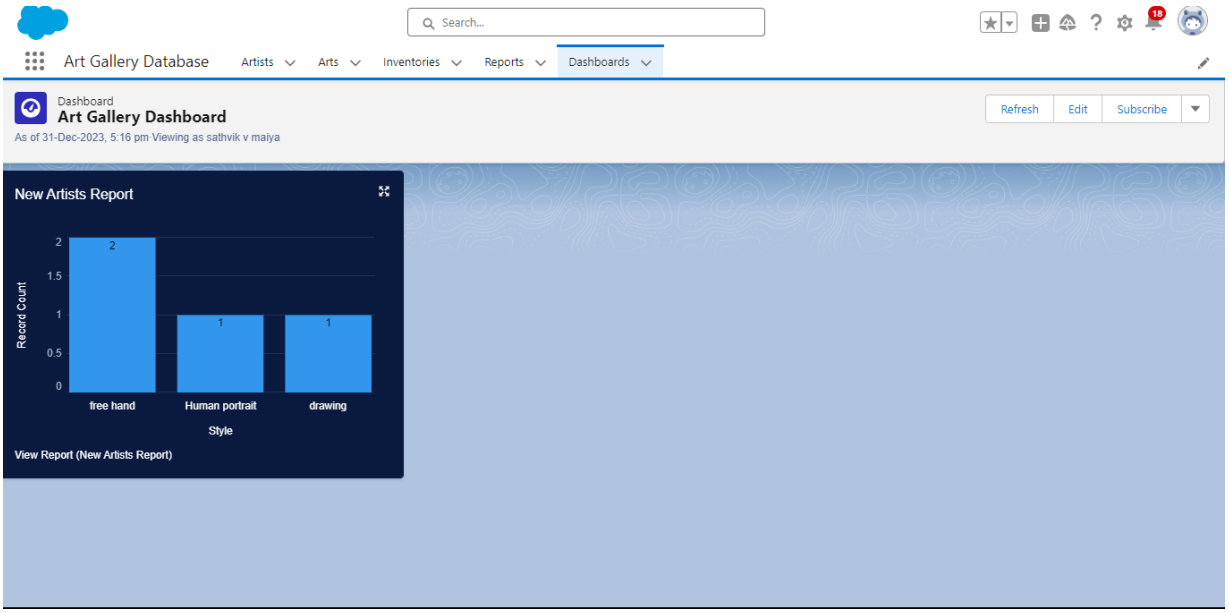
To Create an Artists Dashboard:

1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Art Gallery Dashboard” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run



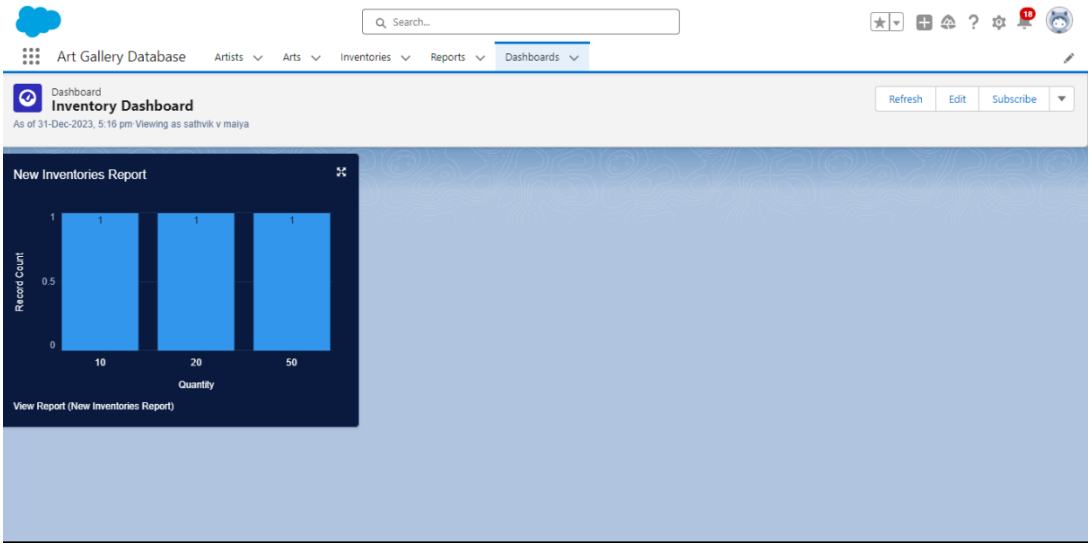
To Create an Arts Dashboard:

1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Artists with art Dashboard” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run



To Create an Inventory Dashboard:

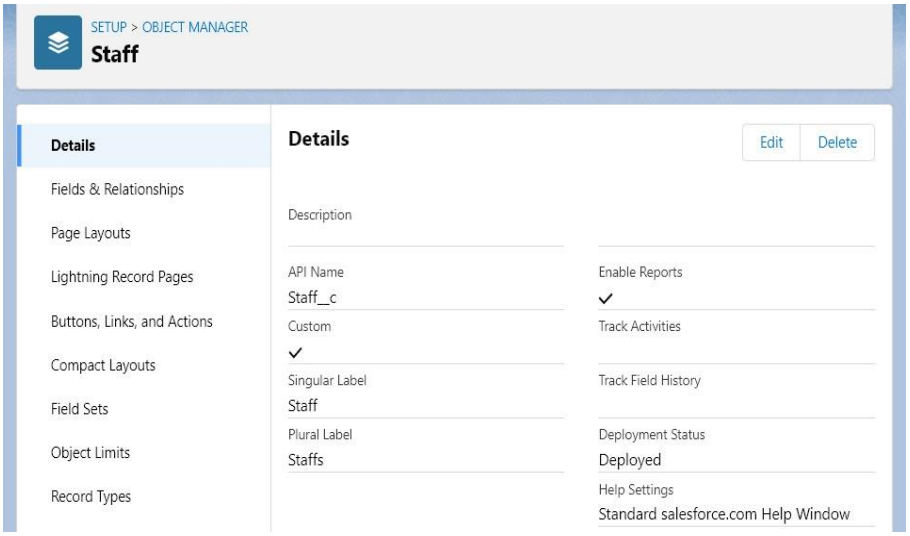
1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Inventory Dashboard” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run



Q10) Create a web application with objects to record details about staff, syllabus and activities of a department and provide a link to college website from any of the objects.

Launch your Salesforce Trailhead Playground by opening any module and Switch to Lightning Experience if you are currently in Salesforce Classic by clicking your picture in the right top corner and then click on "Switch to Lightning Experience"

Then go to Setup gear icon and click "Setup".



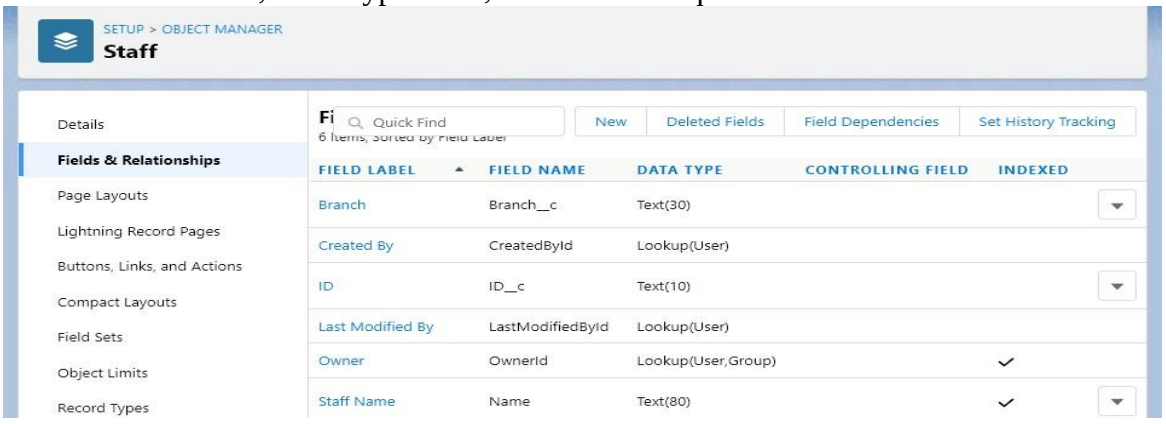
1. Click on "Object Manager" and click "Create > Custom Object" to create new Custom Object.
2. Name the object "Staff".
3. Allow Reports and Allow Search.
4. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object".
5. To create a Tab for the Object: Select any Tab Style for the object "Staff". Click Next, Next, leave the defaults and save.

To add fields to the Object:

Go to "Fields & Relationships" option of Student object and Click "New".

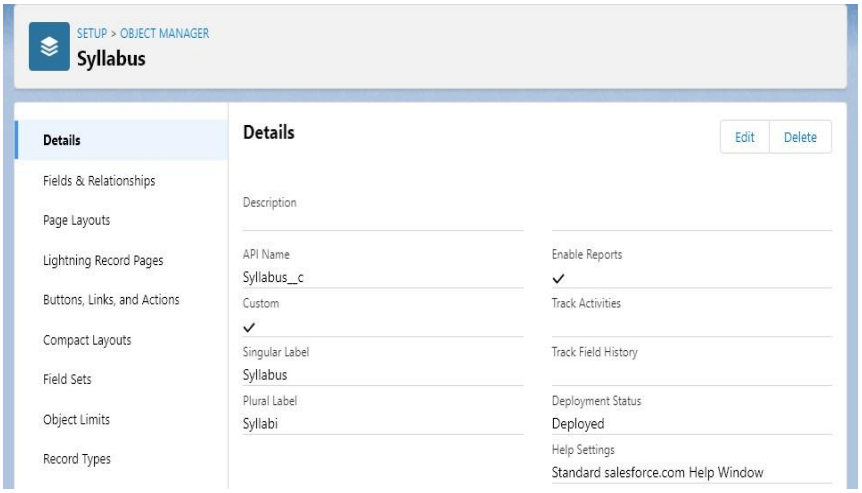
Add the following fields one after the other:

- Field Label: ID (Length 10), Data Type: Text, provide an example ID as Help Text, make it as Required Field, don't allow Duplicate Values, make it as Case Insensitive and Set this field as the unique record identifier from an external system.
- Field Label: Branch, Data Type: Text, Make it as a required field.



Create one more object to store Syllabus details:

1. Name the Object "Syllabus".
2. Allow Reports and Allow Search.
3. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object"
4. Create a Tab for the Object.

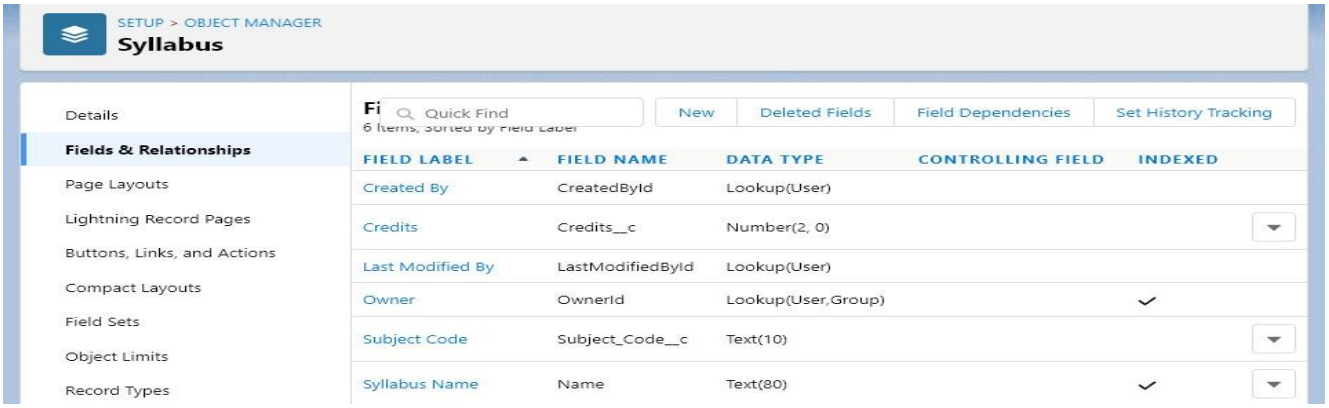


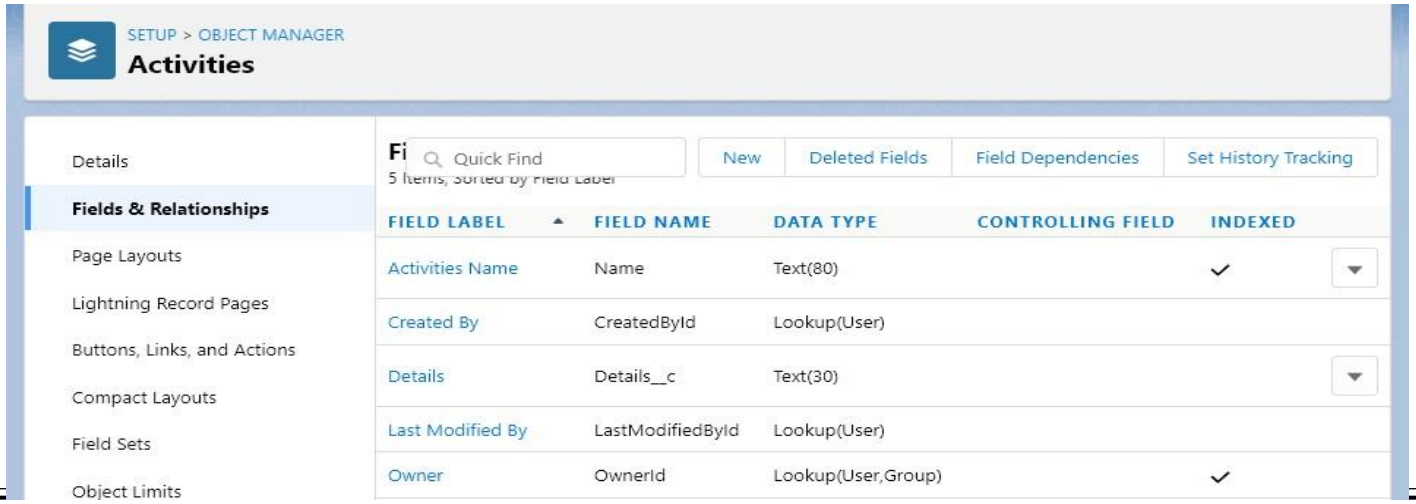
To add fields to the Object:

Go to "Fields & Relationships" option of Student object and Click "New".

Add the following fields one after the other:

- Field Label: Subject Code, Data Type: Text.
- Field Label: Credits, Data Type: Number.





Create one more object to store department activities' details:

1. Name the Object "Activities".
2. Allow Reports and Allow Search.
3. Check the box in front of "Launch New Custom Tab Wizard after saving this custom object"
4. Create a Tab for the Object.

To add fields to the Object:

Go to "Fields & Relationships" option of Student object and Click "New".

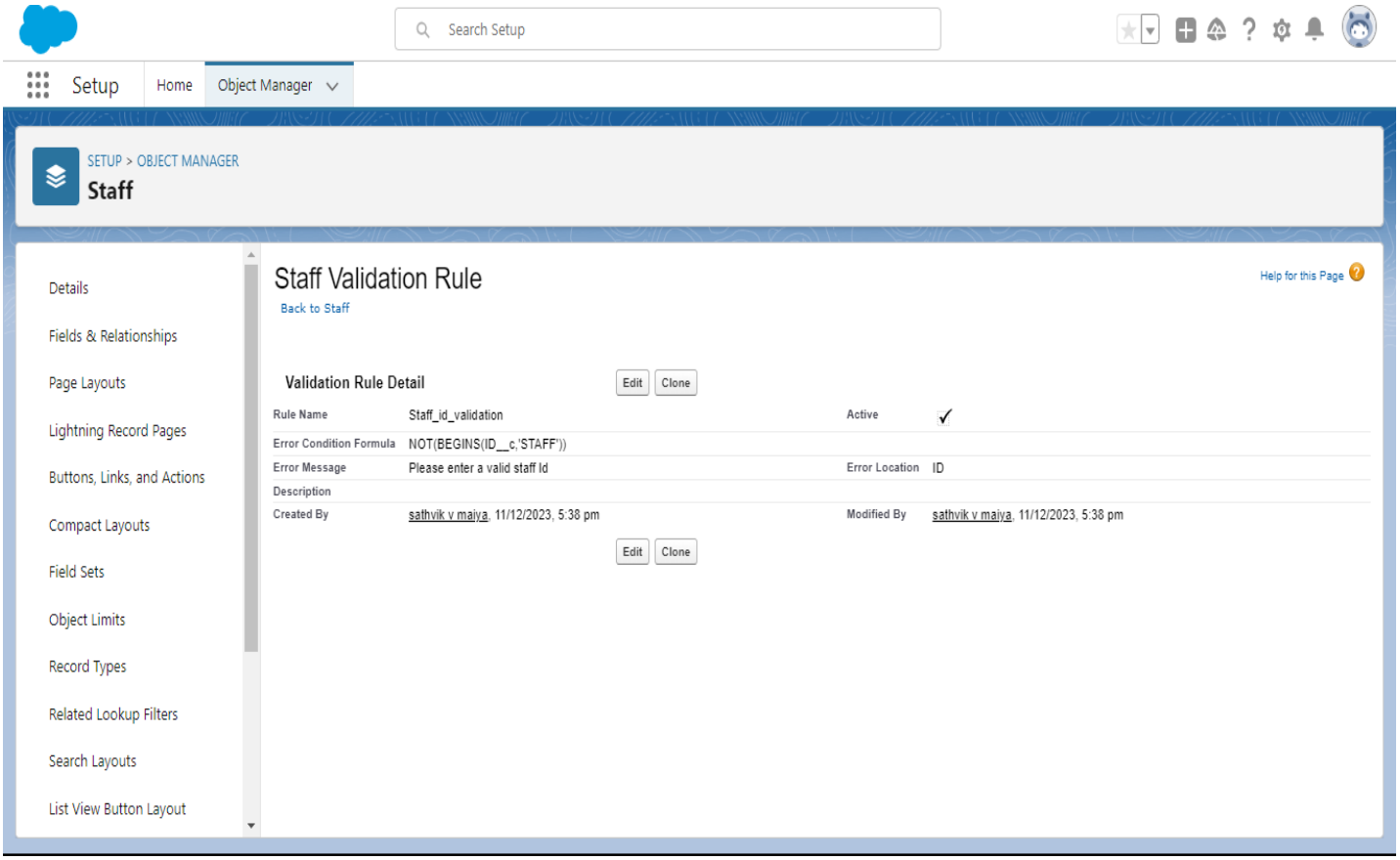
Add the following fields one after the other:

- Field Label: Details, Data Type: Text. Make it as a required field

Validation Rules:

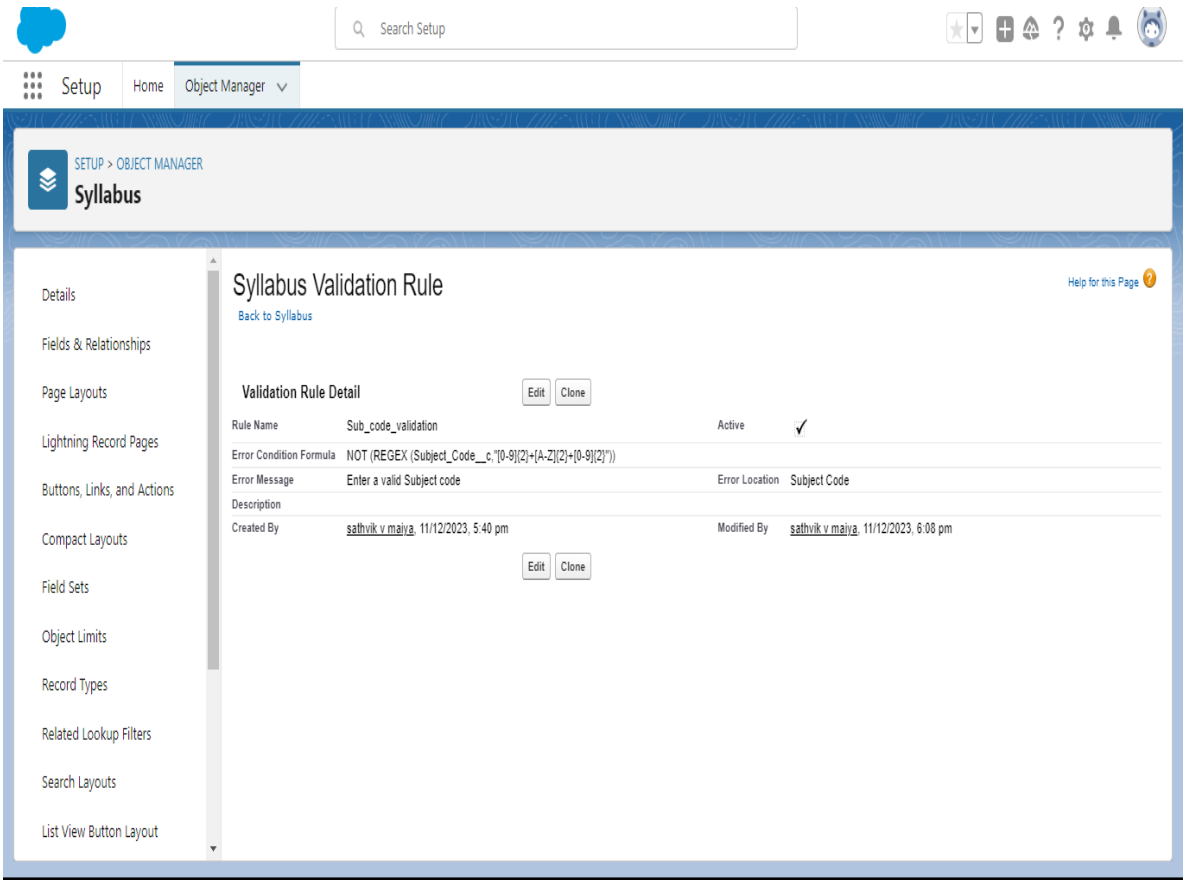
Staff id Validation:

1. To add a rule to the Staff is so that it should take only valid ids:
2. Go to Validation Rule of Staff Object and click "New"
3. Name it as "Staff id validation".
4. Error Condition Formula: NOT(BEGINS(ID__c,'STAFF')).
5. Error Message: Please enter a valid staff Id.
6. Error Location: Field – ID.
7. Click Save.



To add a rule to the Subject code so that it should not take invalid sub code:

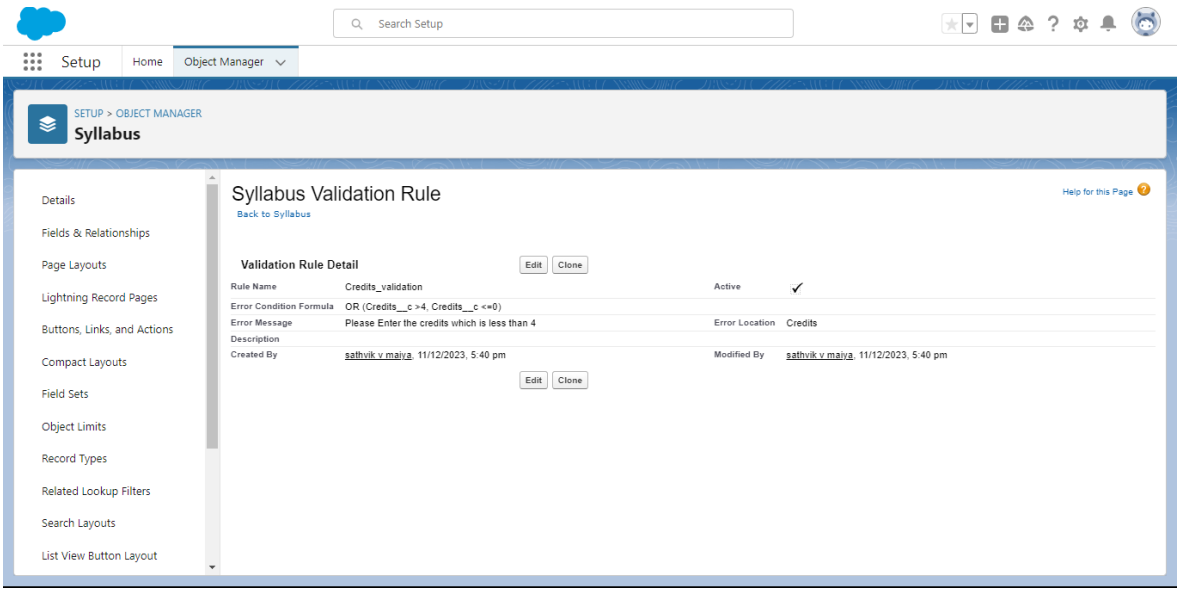
1. Go to Validation Rule of Syllabus Object and click “New”.
2. Name it as “Sub code validation”
3. Error Condition Formula: NOT (REGEX (Subject_Code__c, “[0-9]{2} +[a-z] [A-Z]{2} + [0-9]{2} +”)).
4. Error Message: Please Enter a valid Subject code.
5. Error Location: Field – Subject Code.
6. Click Save



To add a rule to the Credits so that it Should not take more than 4 credits:

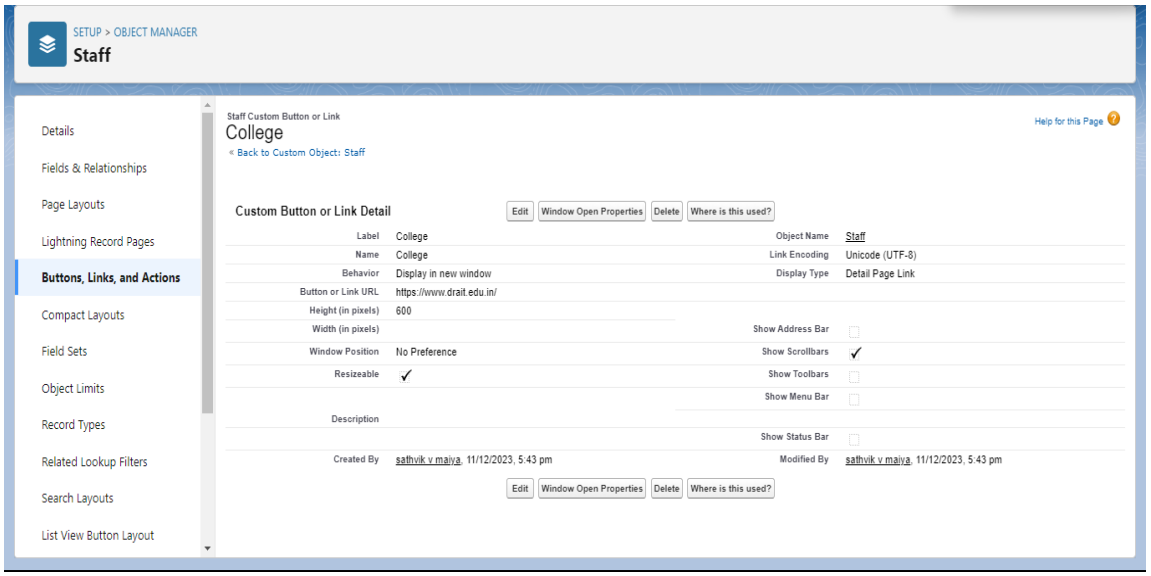
1. Go to Validation Rule of Syllabus Object and click “New”.
2. Name it as “Credits validation”.

3. Error Condition Formula: OR (credits__c >4, Credits__c <=0).
4. Error Message: Please Enter the credits which is less than 4.
5. Error Location: Field –Credit.
6. Click Save.



To give a link to college website:

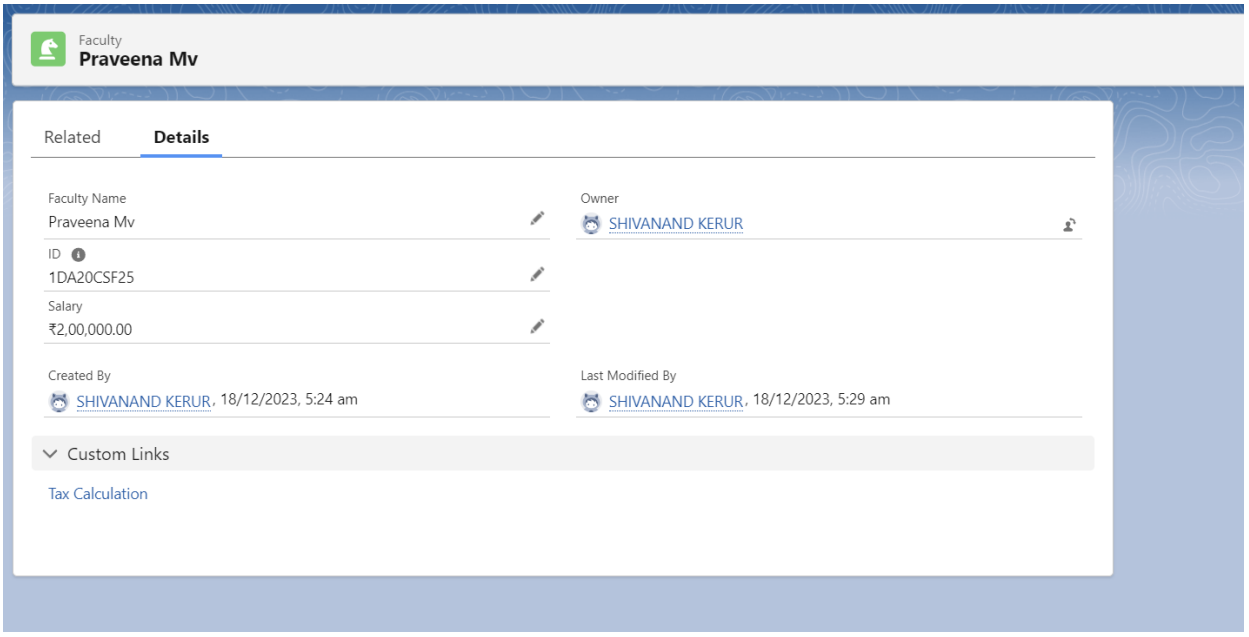
1. Go to “Buttons, Links and Actions” of Art Object and click “New Button or Link”.
2. Name it as “College”.
3. Select the radio button “Detail Page Link” as it is a website link.
4. Behaviour: Display in new window.
5. Content Source: URL.
6. Field Type: College.
7. In the empty space provided, type https://www.drait.edu.in/
8. Link Encoding: Unicode (UTF-8).
9. Click Save
10. Go to Page Layout, Click Activities Layout.
11. Click Custom Links, Drag and drop the “College” link in the Custom Link area.
12. Click Save.



To create an application:

1. Go to “Setup” and type “App Manager” in Quick Find Box.

2. Click on “New Lightning App” to create a Lightning Application.
3. Name it as “Department Details”, give the description for your application.
4. Uploading Image and changing colours are optional, then click Next.
5. Navigation Style: Standard Navigation, click Next.
6. No need to add any Utility Bar, click Next.
7. Add the following:
8. Items: Staff, Syllabuses, Activities, Reports and Dashboards, click Next.
9. Assign it to System Administrator Profile by selecting System Administrator and pressing right arrow and then click Save & Finish.



Go to App Manager, select your application and select Faculties and click “New” to add some details to your application.

Click the entry you added, go to details.

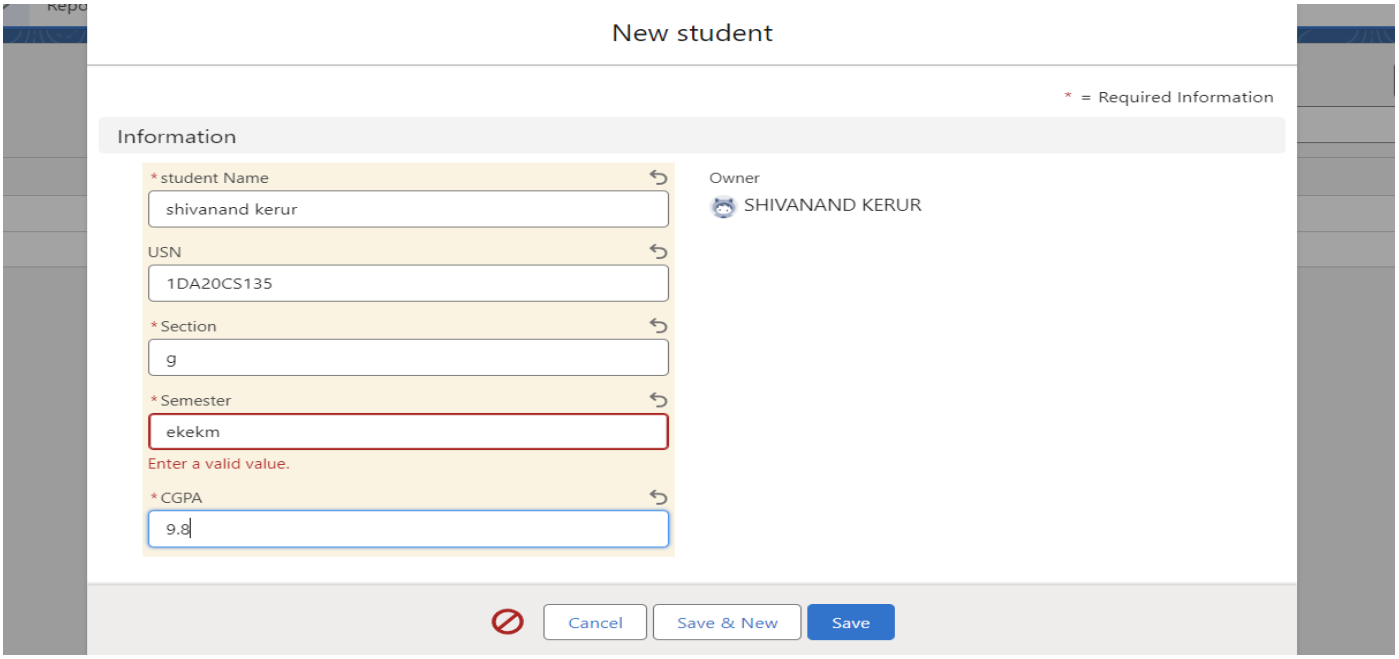
Press the “College” link to check the details.

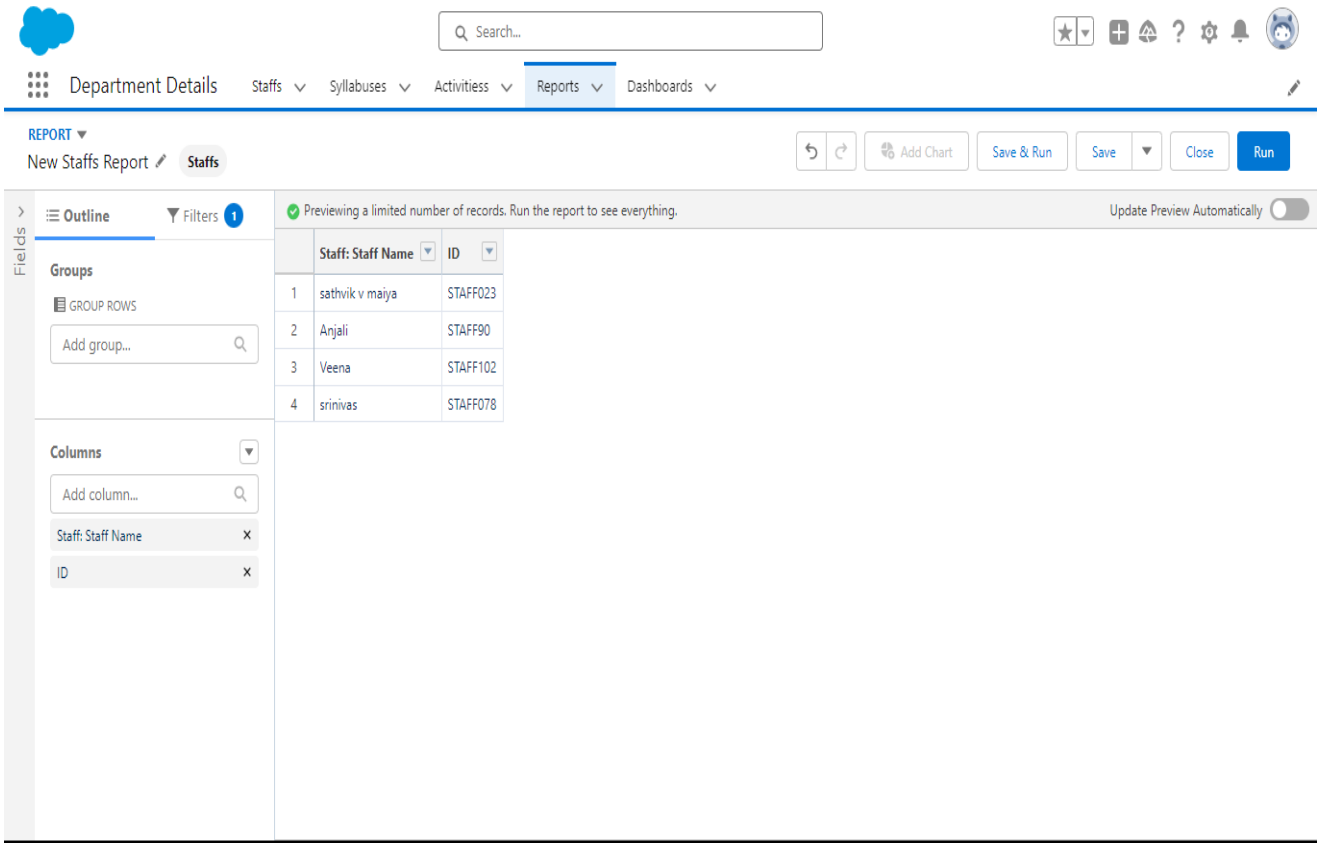
Click OK so that it will redirect you to the website. Make Sure You will get an error when u give invalid staff id, credits and Subject code.

Reports and Dashboards:

To Create a Staff Report:

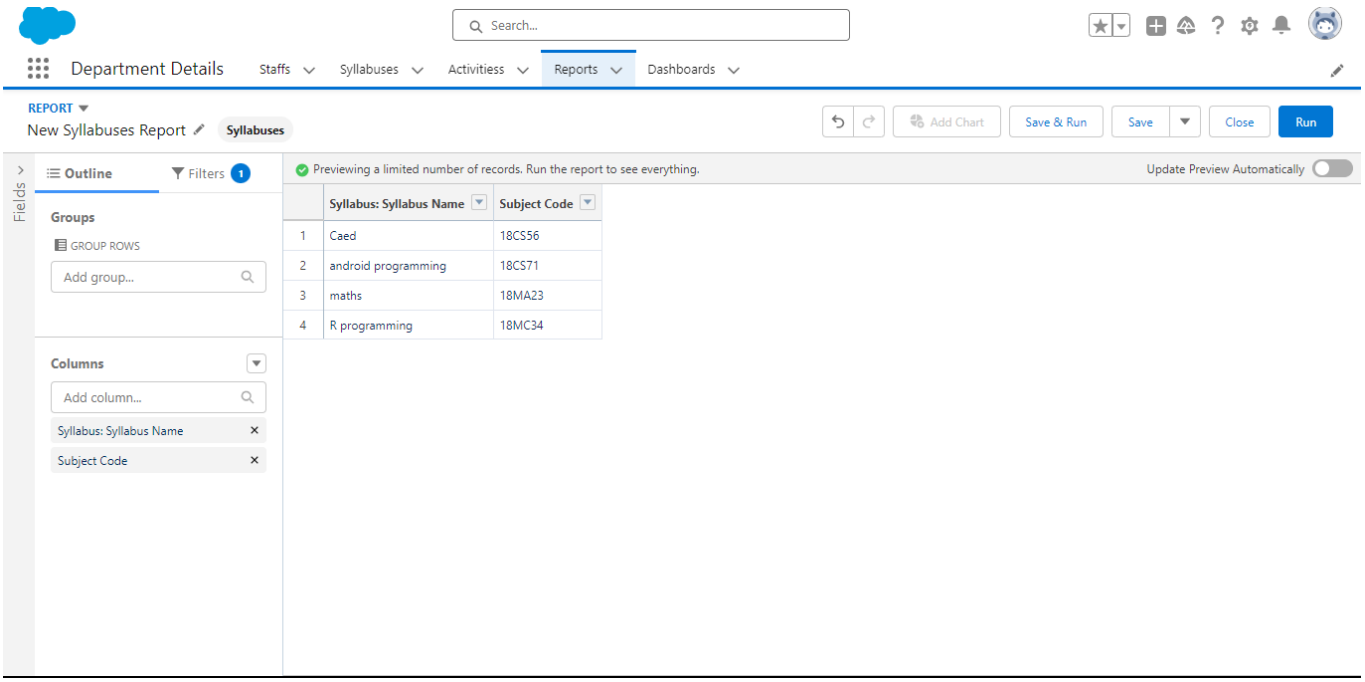
1. Go to “Reports tab” Click on “New Folder” And give it any name then select Save.
2. Click on “New Report” and from search bar Search for “Staffs” and then select it then select Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Staffs Report” and then select the folder which you have created.
6. Click Save and then Click Run.


New student
* = Required Information
Information
* student Name
shivanand kerur
USN
1DA20CS135
* Section
g
* Semester
ekekm
Enter a valid value.
* CGPA
9.8
Owner
SHIVANAND KERUR
Cancel
Save & New
Save


Search...
Department Details
Staffs
Syllabuses
Activitiess
Reports
Dashboards
REPORT
New Staffs Report
Staffs
Add Chart
Save & Run
Save
Close
Run
Fields
Outline
Filters 1
Previewing a limited number of records. Run the report to see everything.
Update Preview Automatically
Groups
GROUP ROWS
Add group...
Columns
Add column...
Staff: Staff Name
ID
Staff: Staff Name
ID
1 sathvik v maiya STAFF023
2 Anjali STAFF90
3 Veena STAFF102
4 srinivas STAFF078

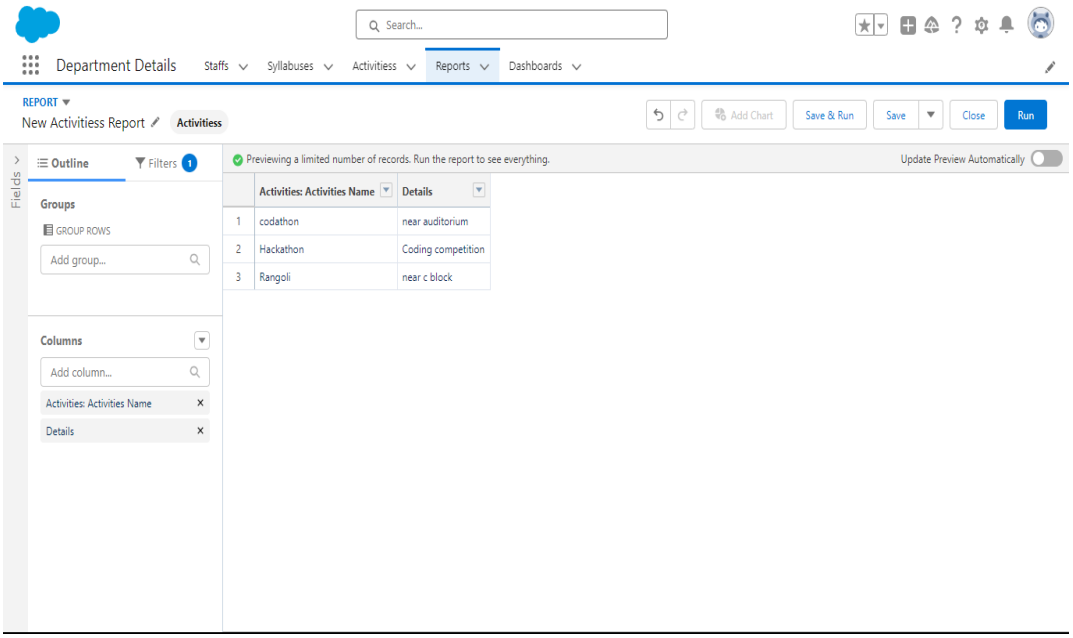
To Create a Syllabus Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name then select Save.
2. Click on “New Report” and from search bar Search for “Syllabus” and then select it then select Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Syllabus Report” and then select the folder which you have created.
6. Click Save and then Click Run



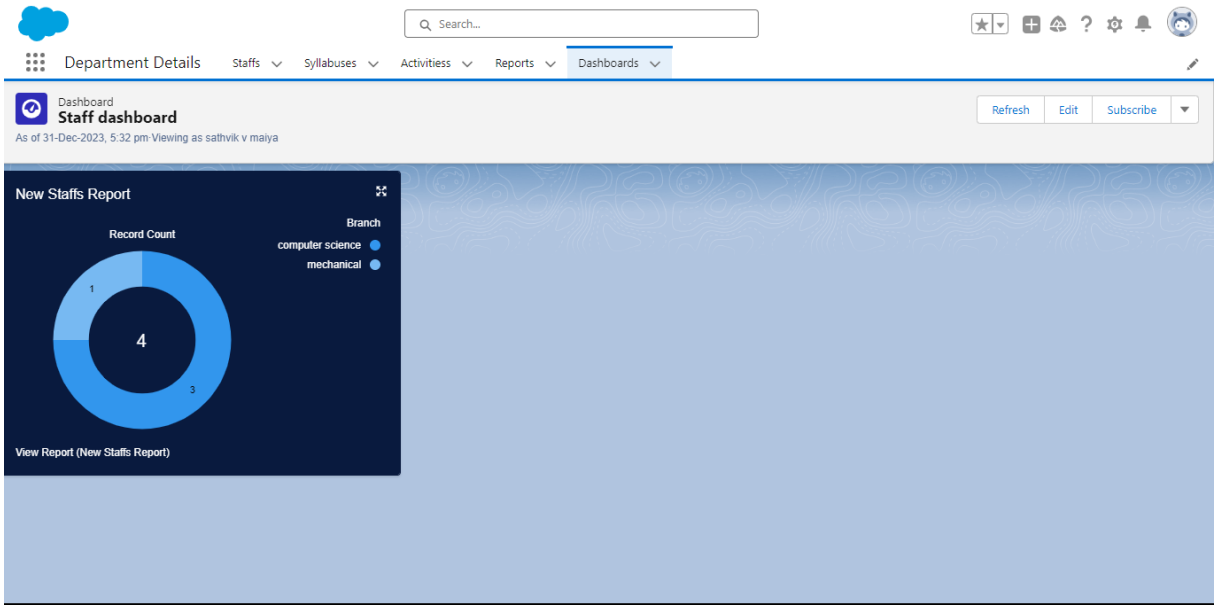
To Create an Activities Report:

1. Go to “Reports tab” Click on “New Folder” And give it any name then select Save.
2. Click on “New Report” and from search bar Search for “Activities” and then select it then select Continue.
3. Add the required Columns to get the Complete Entered data.
4. If you want the report to be grouped by any specific Fields then Search for the field in “Add groups” otherwise it is optional.
5. Click on save and name the report as “New Activities Report” and then select the folder which you have created.
6. Click Save and then Click Run



To Create a Staff Dashboard:

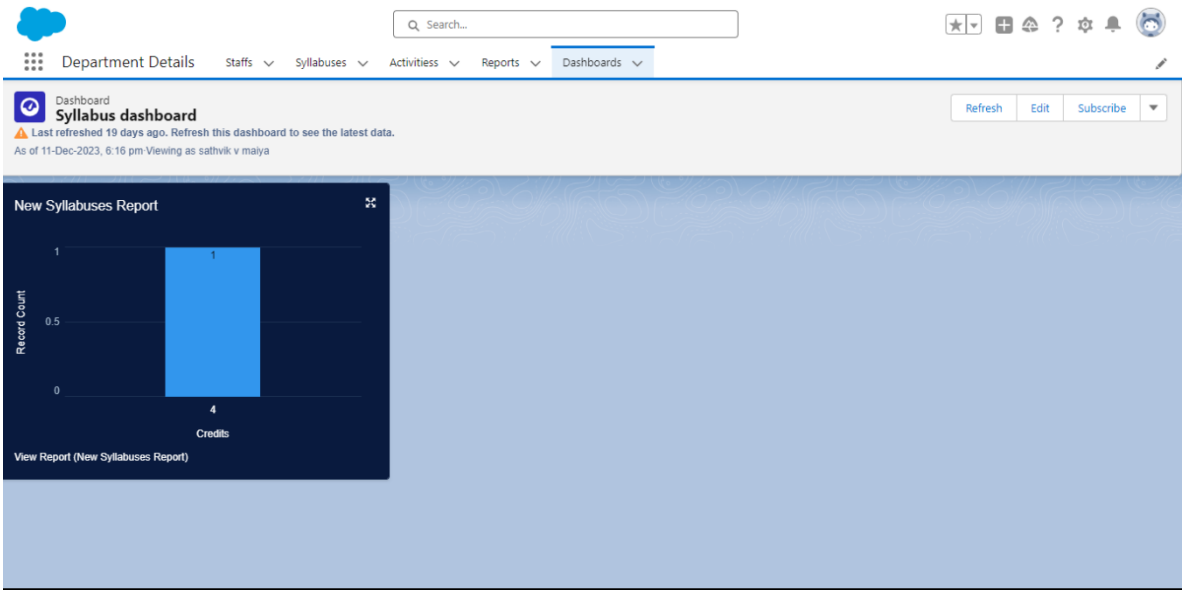
1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Staff dashboard” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run



To Create a Syllabus Dashboard:

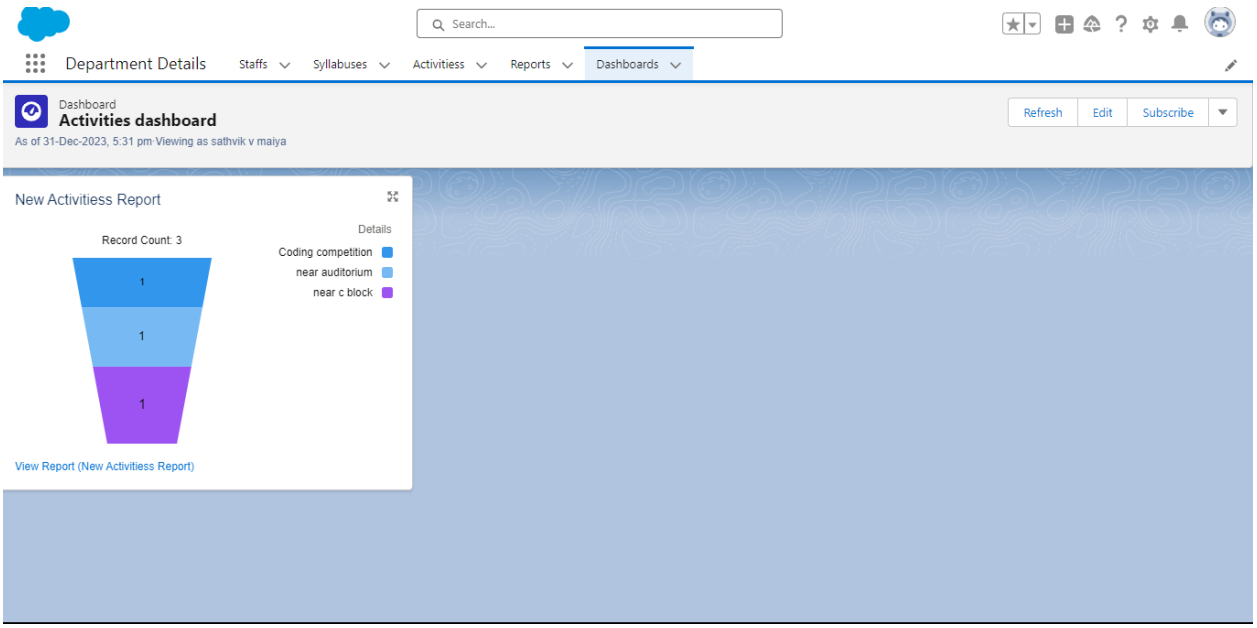
1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Syllabus dashboard” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.

4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run.



To Create an Activities Dashboard:

1. Go to “Dashboard tab” and then click on “New Folder” and give it any Name.
2. Click on “New Dashboard” and then name it as “Activities dashboard” and select folder that you have created Click on Create.
3. Click on the report that you have created and click on that and click select.
4. Select any style to represent the data in dashboard.
5. Add any filter(s), otherwise it is optional.
6. Click on Save and Click Run



CLOUD ANALYST

Cloud Analyst is a tool developed at the University of Melbourne whose goal is to support evaluation of social networks tools according to geographic distribution of users and data centres. In this tool, communities of users and data canters supporting the social networks are characterized and, based on their location; parameters such as user experience while using the social network application and load on the data centre are obtained/logged.

Cloud Analyst is developed by Bhathiya Wickremasinghe et al. at the CLOUDS Laboratory. It is built on top of CloudSim and separates the simulation experimentation from a programming task enabling one to concentrate on the simulation parameters rather than the technicalities of programming. Simulation in Cloud Analyst involves the following steps:

i. Defining and configuration of User Bases.

ii. Defining and configuring Data Centers

iii. Allocating of Virtual Machines in Data Centers.

iv. Review and Adjustment of various other parameters such as Packet size,Number of packets, Bandwidth, and Load balancing policies.

The Cloud Analyst enables us to model different scenarios of CSPs and User Bases,and provides a comprehensive output detailing the response time, Data Center processing time and total cost involved in the communication and computation.

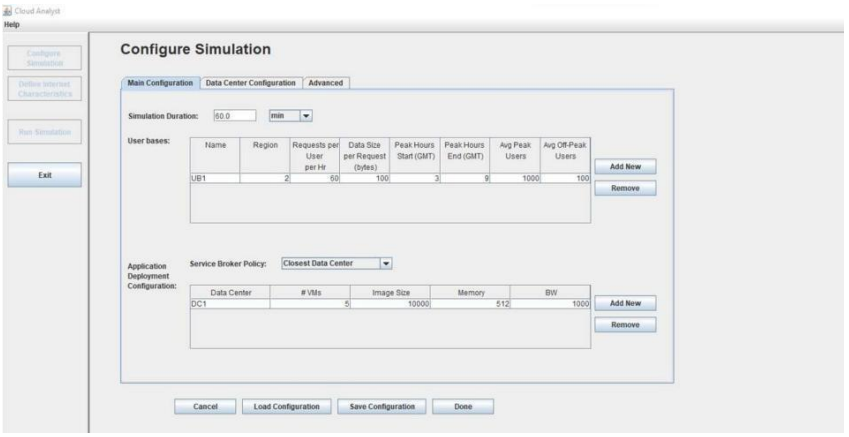
Installing and Running Cloud Analyst:

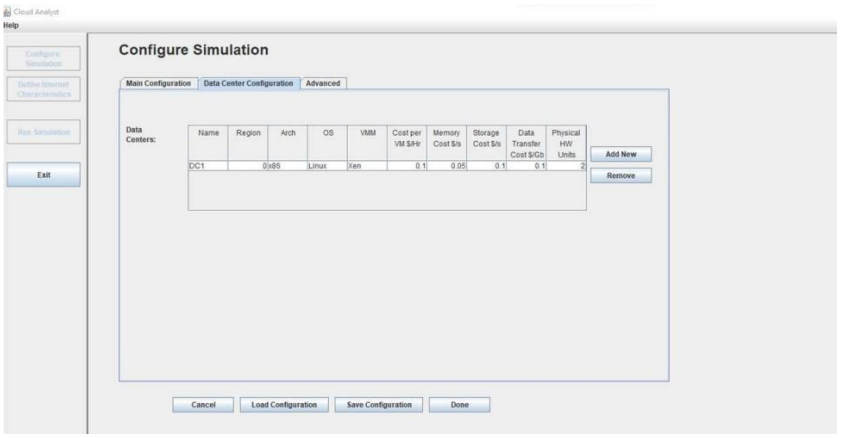
1. Download CloudAnalyst
2. Extract the files from the zip file which will give following folder structure.
3. Click on run.bat file.

4. The user can then configure the simulation which includes

- Simulation Duration
- Number of User Bases
- Service broker policy
- Data Center Configuration
- Load Balancing Policy

5. To run the simulation, click on Run Simulation.





Cloud Analyst Simulations

<table>
<tr><th colspan="2">Exp. No</th><th>Experiment List</th></tr>
<tr><td colspan="3">PART-A</td></tr>
<tr><td>1</td><td>a)</td><td>Creation of web applications on Salesforce cloud Platform.</td></tr>
<tr><td></td><td>b)</td><td>Use the following user base configuration to simulate following scenarios for the given data Centre and virtual machine configuration and answer to the following questions.
Scenario-1: Nearest data center with round robin policies
Scenario-2: Optimize response time with round robin policies

<table>
<tr><th>User base</th><th>Region</th><th>Data center</th><th>Peak-hour users</th><th>Off-peak hour users</th><th>Virtual machines</th></tr>
<tr><td>UB1</td><td>North America</td><td>--</td><td>1000</td><td>500</td><td rowspan="6">DC1-50</td></tr>
<tr><td>UB2</td><td>South America</td><td>--</td><td>1200</td><td>800</td></tr>
<tr><td>UB3</td><td>Europe</td><td>DC1</td><td>2000</td><td>1000</td></tr>
<tr><td>UB4</td><td>Africa</td><td>--</td><td>500</td><td>300</td></tr>
<tr><td>UB5</td><td>Asia</td><td></td><td>3000</td><td>300</td></tr>
<tr><td>UB6</td><td>Ocenia</td><td></td><td>1500</td><td>150</td></tr>
</table>

i) Tabulate the overall response time of all the scenarios and plot a line graph
ii) Plot a bar graph for the data processing time of all the scenarios
iii) Compare average response time by regions of all scenarios by plotting line graph
iv) Using Pie chart show the total cost spent for each scenario</td></tr>
<tr><td>2</td><td>a)</td><td>Install Virtual box/VMware Workstation with different flavours of Linux and execute some C programs</td></tr>
<tr><td></td><td>b)</td><td>Simulate the following scenarios for the given user base, data Centre and virtual machine configuration and answer to the given questions
<table>
<tr><th>Scenario</th><th>Scenario Description</th><th>Load Balancing algorithm</th><th>Service broker policy</th></tr>
<tr><td>1</td><td>One data center with 50 Virtual Machines for UB1</td><td rowspan="3">Nearest Data Centre</td><td rowspan="3">Round robin</td></tr>
<tr><td>2</td><td>Two data centers with 25 and 50 Virtual Machines respectively for UB1</td></tr>
<tr><td>3</td><td>Three data centers with 100,75 and 25 Virtual Machines respectively for UB1</td></tr>
</table>

i) Tabulate the overall response time and data processing of all the scenarios and plot the bar graph
ii) Plot a line graph of data center request servicing time of all the data centers for all the scenarios
iii) Compare average response time by regions of all scenarios by plotting line graph
iv) Mention the data centers used by the UB2,UB3, UB4 and UB5</td></tr>
</table>

3 a) Install Google App Engine. Create hello world app and other simple web applications using python/java.

b) Simulate the following scenarios for given data Centre, data Centre and virtual machine configuration and answer the following questions

Scenario 1: closest data center and round robin policies

Scenario 2: optimize response time and round robin policies

Use the following user base configuration for all the scenarios

User base	Region	Data center	Peak-hour users	Off-peak hour users	Virtual machines
UB1	North America	DC1, DC3	1000	500	DC1-50 DC3-100
UB2	South America	---	800	1200	
UB3	Europe	DC4	2000	1000	DC4-150
UB4	Africa	--	500	300	

i) Tabulate and compare the Average response time and data processing time of all the scenarios by plotting the line graph

ii) Tabulate the response time of user bases in all scenarios and compare these by plotting bar graph. Which user base is taking maximum time among three scenarios? Why

iii) Calculate the data transmission time from DC1 to UB2

iv) Plot the bar graph for data center cost of all scenarios

4 a) Create a RDS and launch in your custom VPC network using AWS.

b) Analyze the various service broker policies for the following configuration and answer the following questions.

Parameter	Value Used
UB Name	UB1
Region	2
Request Per User Per Hour	60
Data Size Per Request	100
Peak hour start(GMT)	3
Peak hour end (GMT)	9
Avg Peak Users	40000
Avg Off Peak Users	4000
DC 1 – No Of VM	75
DC 2 – No Of VM	50
DC 3 – No Of VM	25
VM Image Size	10000 MB
VM Memory	512 MB
VM Bandwidth	1000 bps
DC 1 – No Of Physical Machine	2
DC 2 – No Of Physical Machine	2
DC 3 – No Of Physical Machine	2
DC – Memory Per Machine	204800 Mb
DC – Storage Per Machine	100000000 Mb
DC – Available BW Per Machine	1000000
DC – No Of Processors Per Machine	4
DC – Processor Speed	10000 MIPS
DC – VM Policy	Time Shared
User Grouping Factor	1000
Request Grouping Factor	100
Executable Instruction Length	500
Load Balancing Policy	Throttled

a) Tabulate and compare the data processing time of service broker policies by plotting the line graph

b) Tabulate and compare response time of service broker policies by plotting the bar graph

c) Tabulate the cost for service broker policies and represent it using pie chart
d) Which service broker policy is best and why?

5 a) Create a file in one virtual machine and transfer it another virtual machine files from one virtual machine.

b) Analyze the various load balancing algorithms for the given user base, data Centre and virtual machine configuration and answer the following questions. Consider the following user base configuration for all load balancing algorithms

Number of User bases	06
Region for the user bases	UB1-South America, UB2-Asia, UB3-North America, UB4-Europe, UB5-Africa, UB6-Ocenia
Average peak users for all the user bases	10000
Average off-peak users for all the user bases	100
Peak hours' time	Depends on the region
Data centers in each user base	UB1-1, UB2-2, UB3-1, UB4-3, UB5-2, UB6-1
Virtual machines in each data center	6
Simulation time	10 mins
Service broker policy	Nearest data center

a) Tabulate and compare the data processing time of load balancing algorithms by plotting the line graph
b) Tabulate the response time of load balancing algorithms by plotting the bar graph
c) Tabulate the response time by region for load balancing algorithms and plot bar graph
d) Which load balancing algorithm is best and why?

1b)

Scenario 1

Results of the Simulation Completed at: 29/11/2023 15:04:27

Overall Response Time Summary

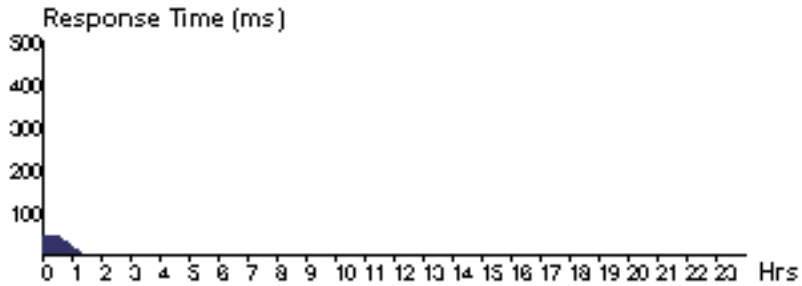
	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	264.08	37.63	657.65
Data Center processing time:	0.31	0.03	0.98

Response Time by Region

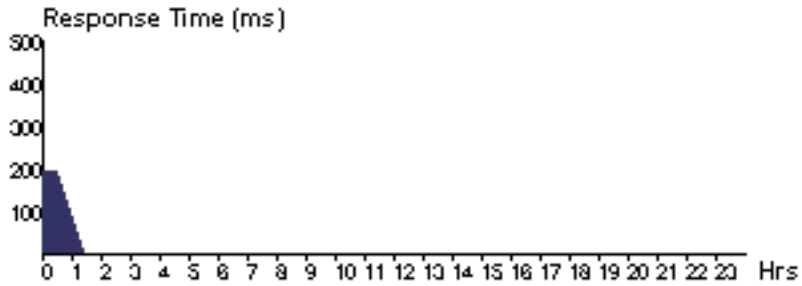
Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.09	37.63	61.88
UB2	200.50	150.23	256.19
UB3	299.77	223.73	376.71
UB4	500.47	387.65	657.65
UB5	500.31	390.15	640.14
UB6	199.56	144.13	243.14

User Base Hourly Response Times

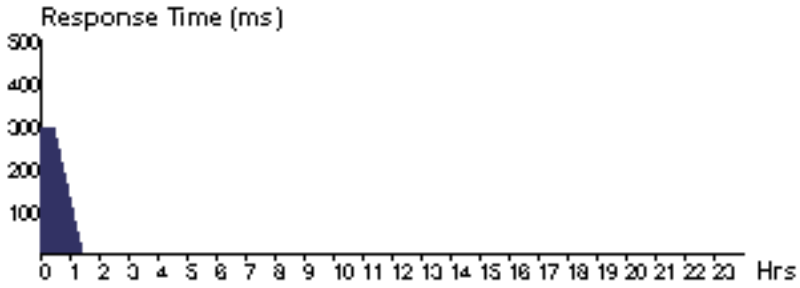
UB1



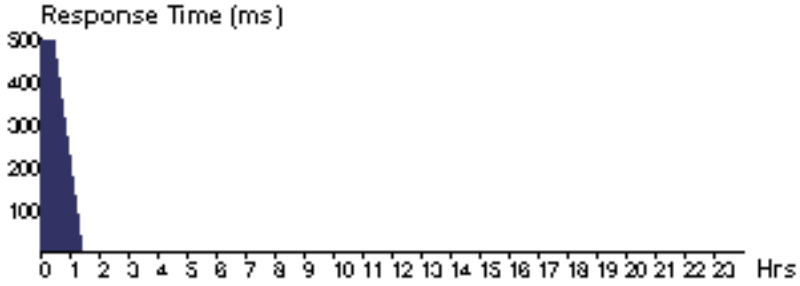
UB2



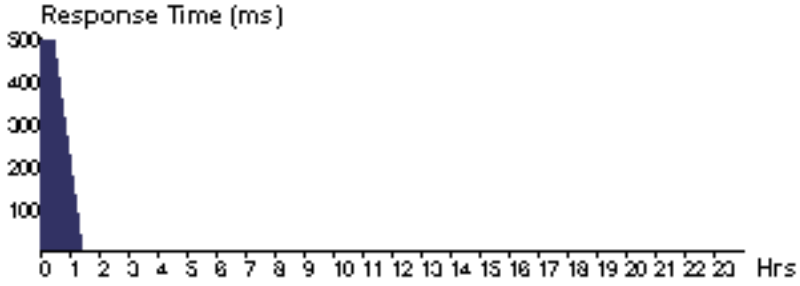
UB3



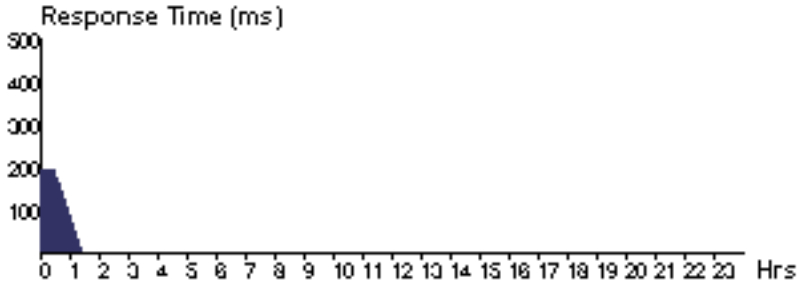
UB4



UB5



UB6

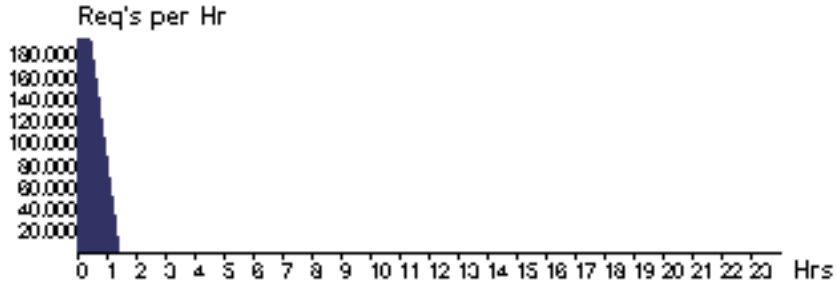


Data Center Request Servicing Times

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	0.31	0.03	0.98

Data Center Hourly Loading

DC1



Cost

Total Virtual Machine Cost ($):	0.50
Total Data Transfer Cost ($):	2.04
Grand Total: ($)	2.54

Data Center	**VM Cost $**	**Data Transfer Cost $**	**Total $**
DC1	0.50	2.04	2.54

Scenario 2

Results of the Simulation Completed at: 29/11/2023 15:09:03

Overall Response Time Summary

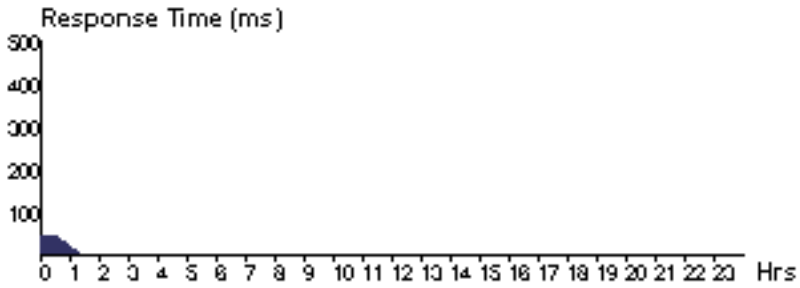
	Avg (ms)	**Min (ms)**	**Max (ms)**
Overall response time:	264.34	39.64	612.64
Data Center processing time:	0.31	0.03	0.98

Response Time by Region

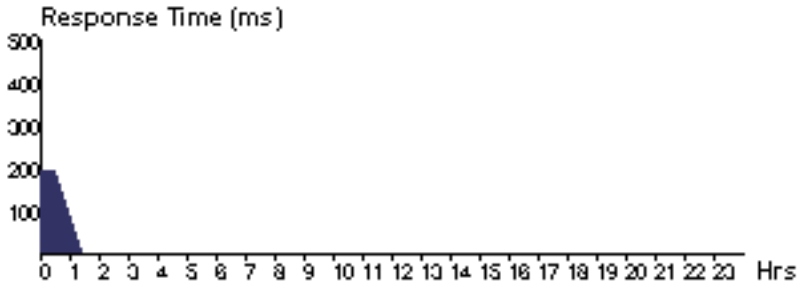
Userbase	**Avg (ms)**	**Min (ms)**	**Max (ms)**
UB1	50.16	39.64	62.88
UB2	200.20	154.18	250.25
UB3	300.09	231.25	381.20
UB4	501.32	392.66	612.64
UB5	500.88	375.13	610.15
UB6	200.60	152.14	252.14

User Base Hourly Response Times

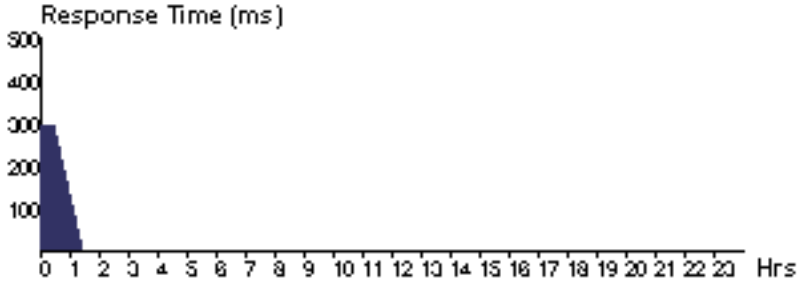
UB1



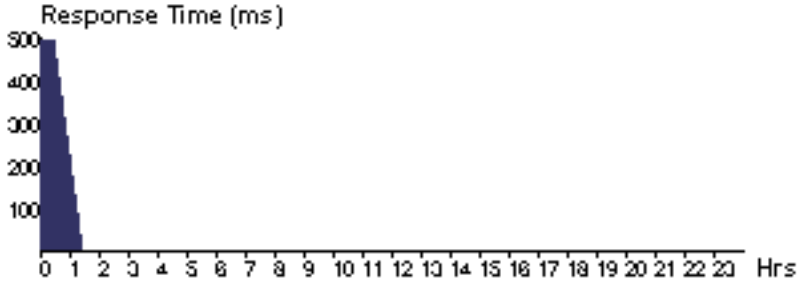
UB2



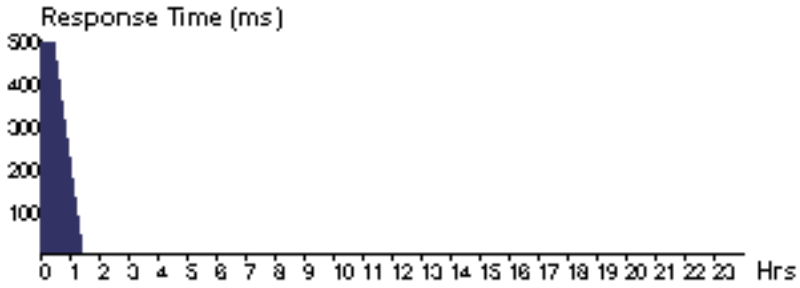
UB3



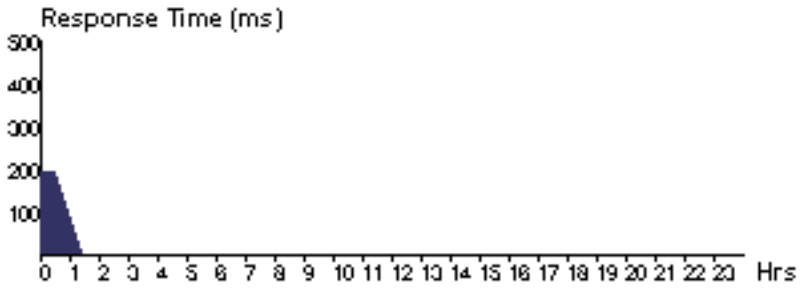
UB4



UB5



UB6

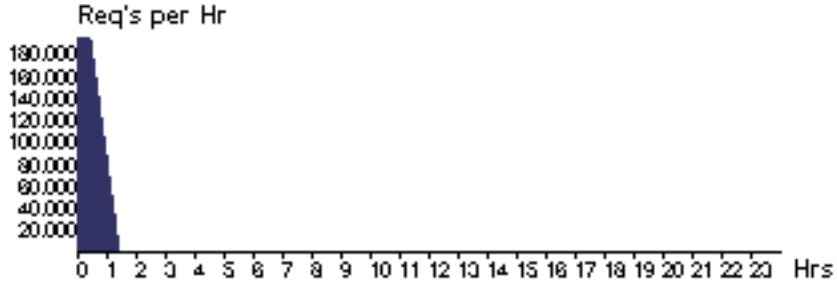


Data Center Request Servicing Times

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	0.31	0.03	0.98

Data Center Hourly Loading

DC1

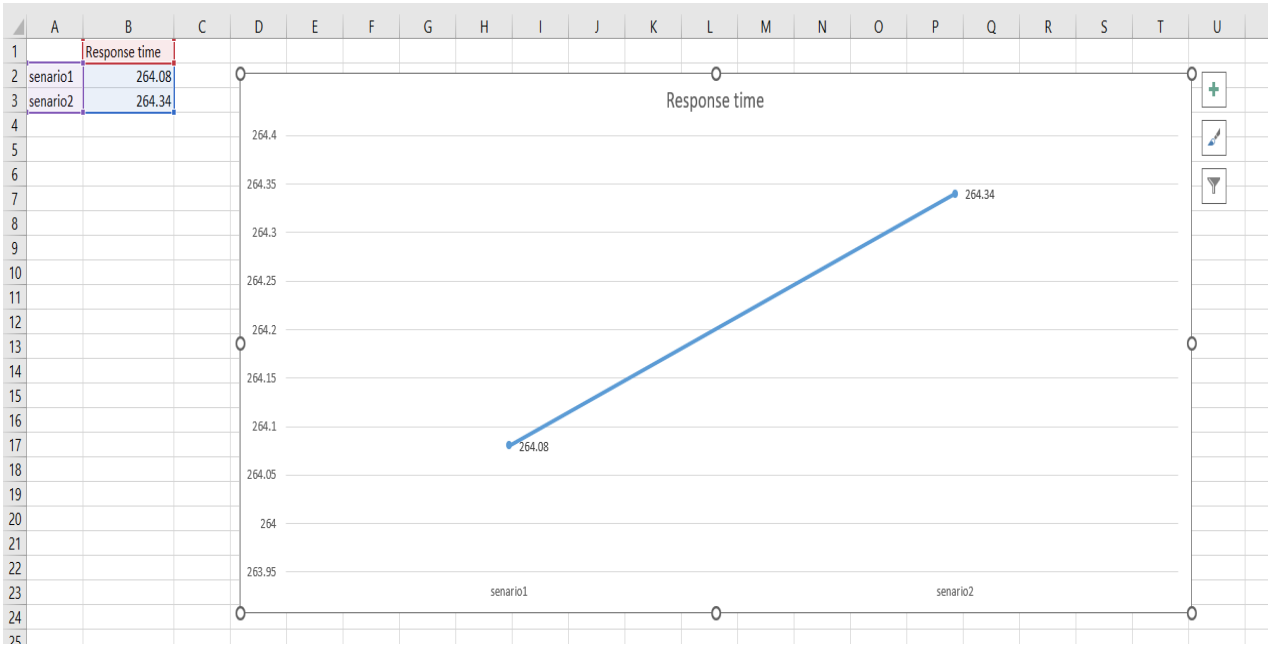


Cost

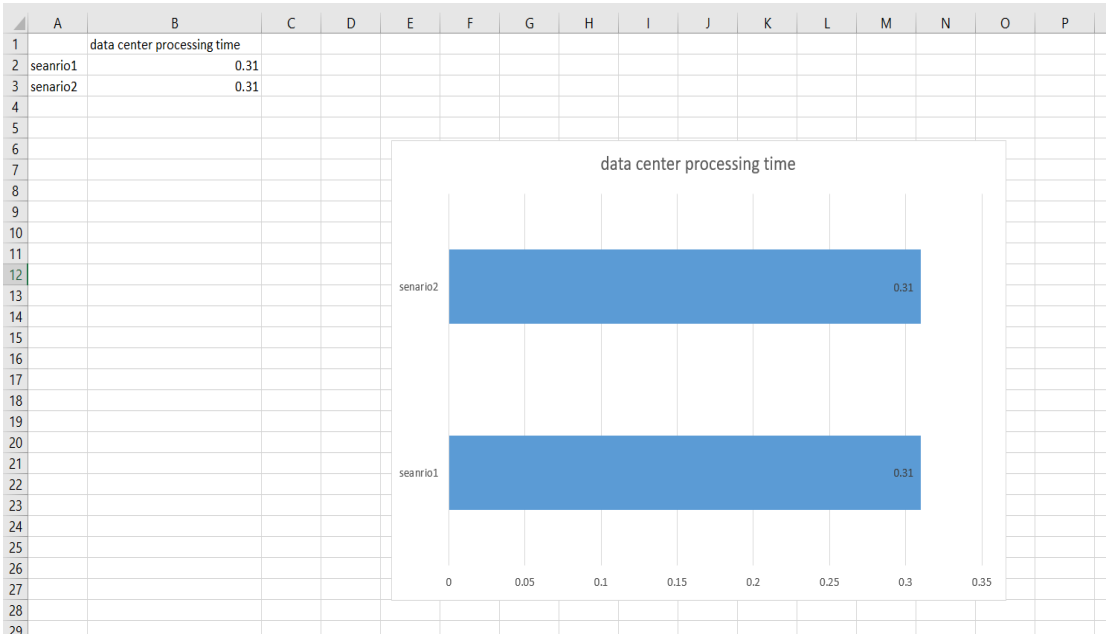
Total Virtual Machine Cost ($):	0.50
Total Data Transfer Cost ($):	2.04
Grand Total: ($)	2.54

Data Center	VM Cost $	Data Transfer Cost $	Total $
DC1	0.50	2.04	2.54

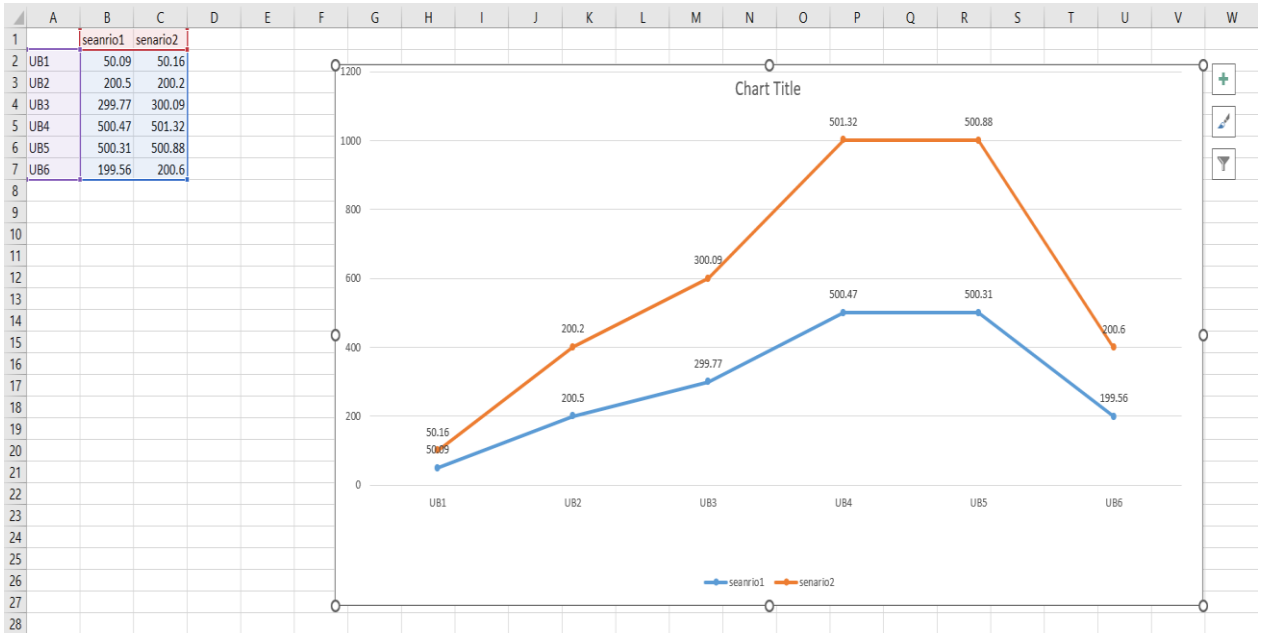
i) Tabulate the overall response time of all the scenarios and plot a line graph



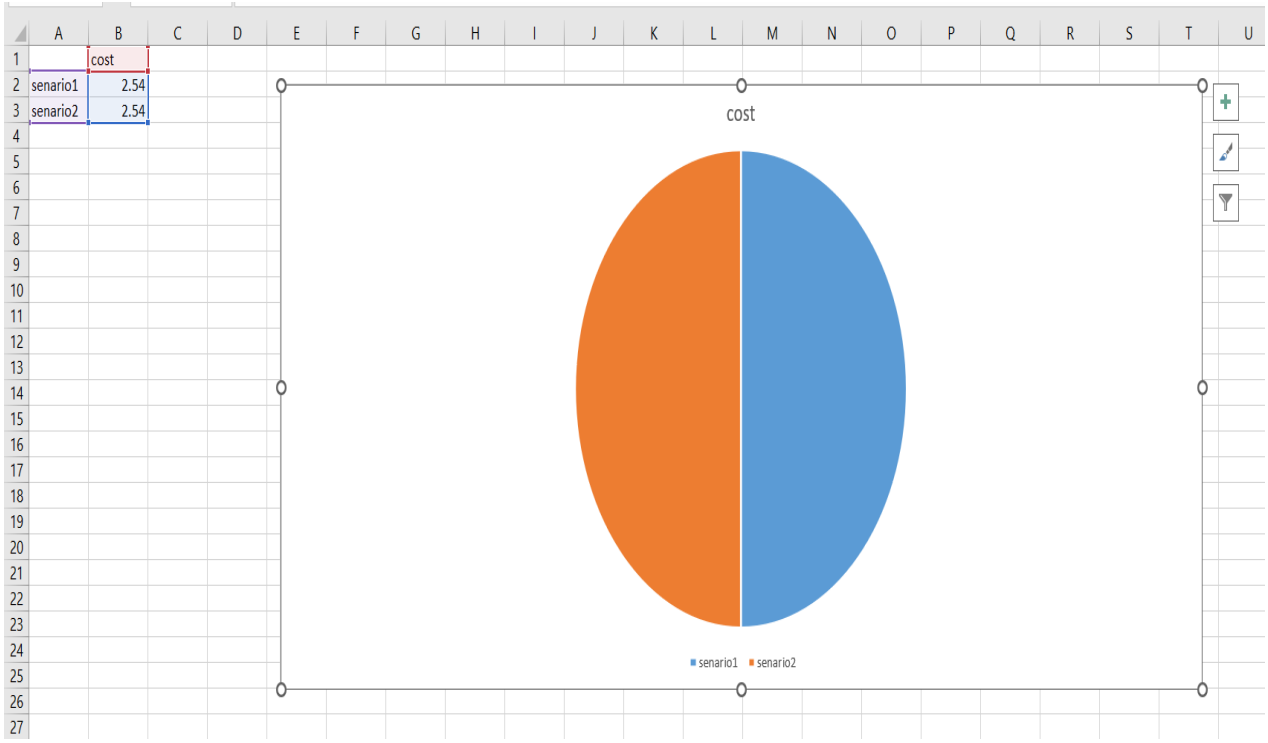
ii) Plot a bar graph for the data processing time of all the scenarios



iii) Compare average response time by regions of all scenarios by plotting line graph



iv) Using Pie chart show the total cost spent for each scenario



2b)

Scenario 1

Results of the Simulation Completed at: 27/11/2023 15:16:09

Overall Response Time Summary

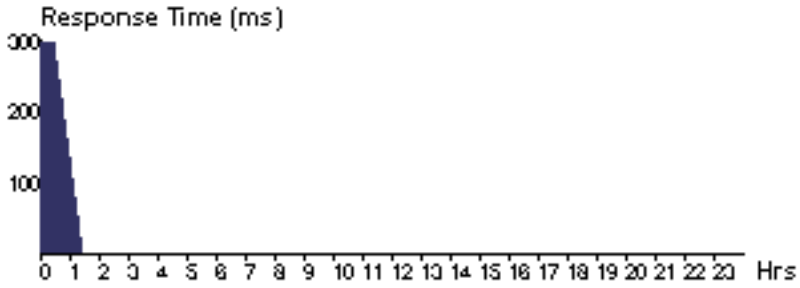
	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	300.06	237.06	369.12
Data Center processing time:	0.34	0.02	0.61

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	300.06	237.06	369.12

User Base Hourly Response Times

UB1

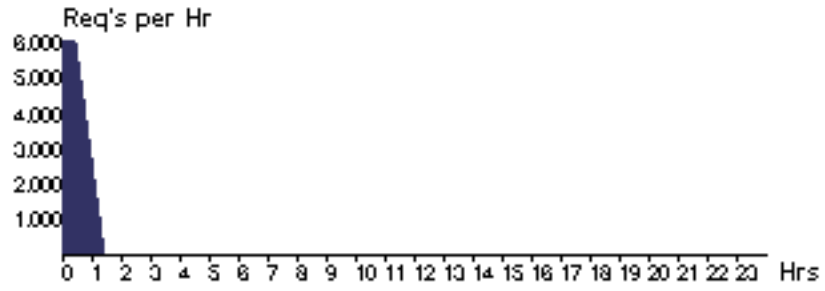


Data Center Request Servicing Times

Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	0.34	0.02	0.61

Data Center Hourly Loading

DC1



Cost

Total Virtual Machine Cost ($):	0.51
Total Data Transfer Cost ($):	0.06
Grand Total: ($)	0.57

Data Center	**VM Cost $**	**Data Transfer Cost $**	**Total $**
DC1	0.51	0.06	0.57

Scenario 2

Results of the Simulation Completed at: 27/11/2023 15:18:17

Overall Response Time Summary

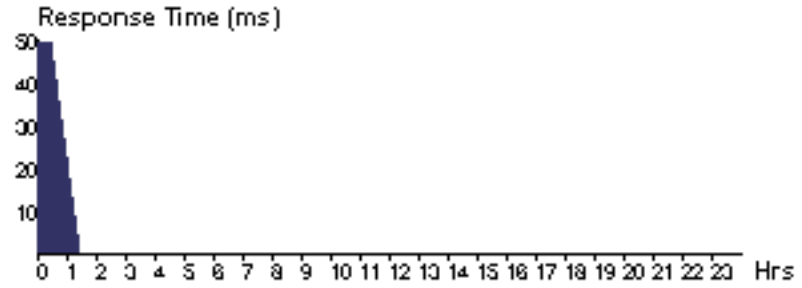
	Avg (ms)	**Min (ms)**	**Max (ms)**
Overall response time:	50.20	39.63	61.63
Data Center processing time:	0.59	0.03	1.08

Response Time by Region

Userbase	**Avg (ms)**	**Min (ms)**	**Max (ms)**
UB1	50.20	39.63	61.63

User Base Hourly Response Times

UB1

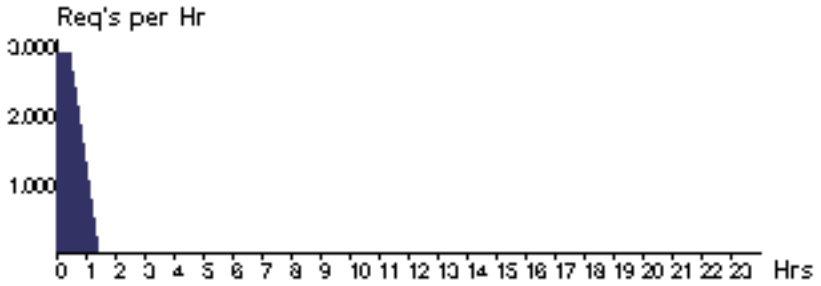


Data Center Request Servicing Times

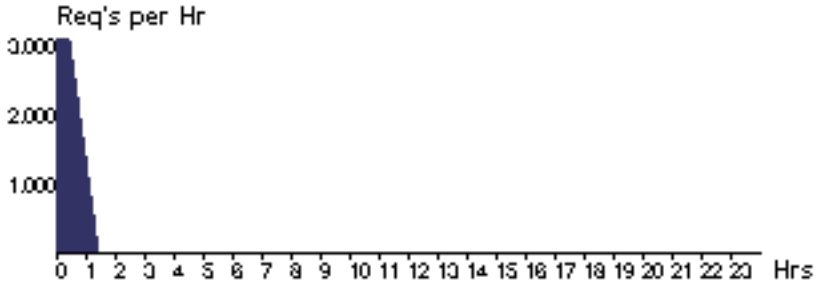
Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	0.69	0.03	1.08
DC2	0.49	0.05	0.88

Data Center Hourly Loading

DC1



DC2



Cost

Total Virtual Machine Cost ($):	3.04
Total Data Transfer Cost ($):	0.06
Grand Total: ($)	3.11

Data Center	**VM Cost $**	**Data Transfer Cost $**	**Total $**
DC2	0.51	0.03	0.54
DC1	2.54	0.03	2.57

Scenario 3

Results of the Simulation Completed at: 27/11/2023 15:20:10

Overall Response Time Summary

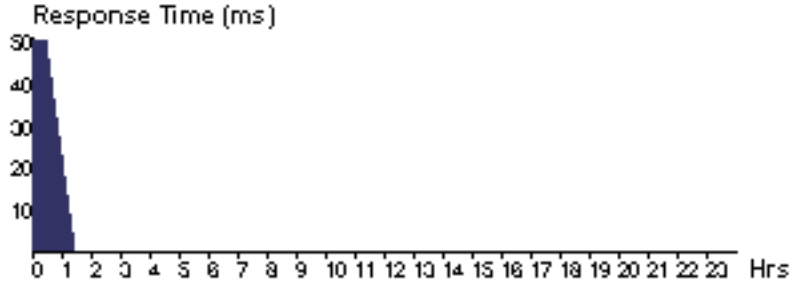
	Avg (ms)	**Min (ms)**	**Max (ms)**
Overall response time:	51.18	39.76	62.76
Data Center processing time:	1.56	0.13	2.63

Response Time by Region

Userbase	**Avg (ms)**	**Min (ms)**	**Max (ms)**
UB1	51.18	39.76	62.76

User Base Hourly Response Times

UB1

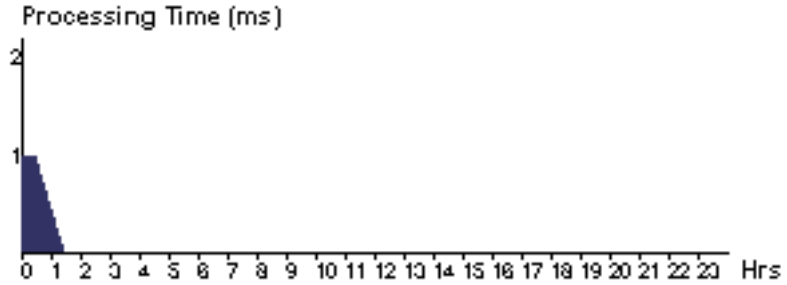


Data Center Request Servicing Times

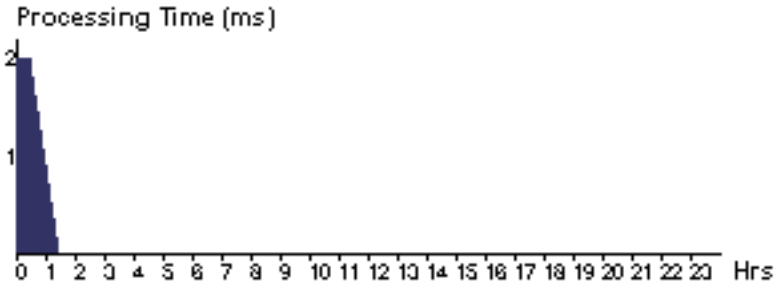
Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	1.56	0.50	2.00
DC2	2.18	0.19	2.63
DC3	1.01	0.13	1.38

Data Center Hourly Average Processing Times

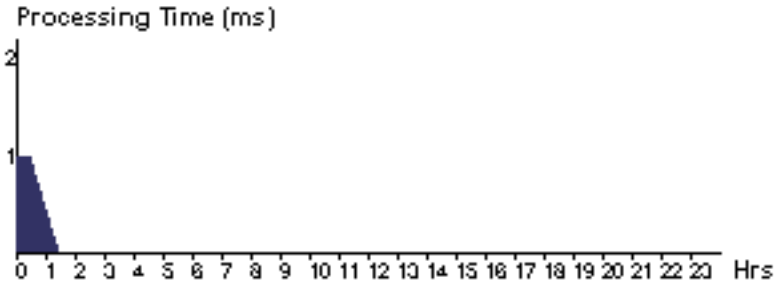
DC1



DC2

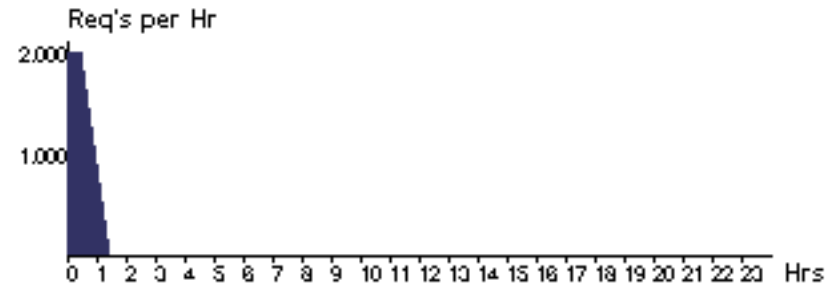


DC3

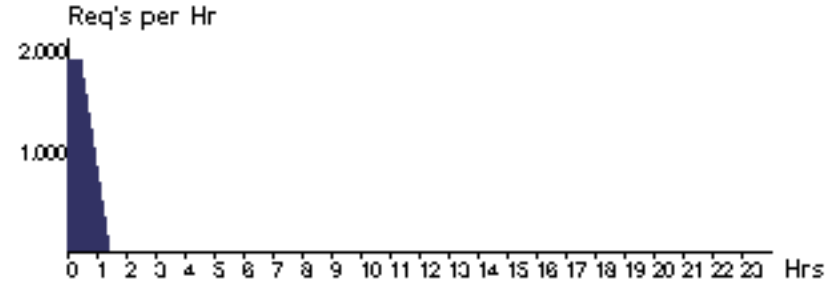


Data Center Hourly Loading

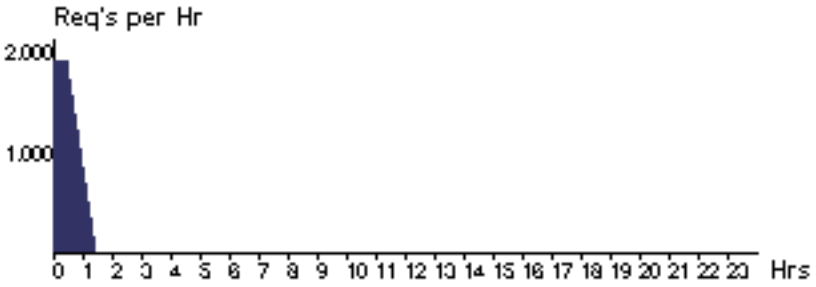
DC1



DC2



DC3

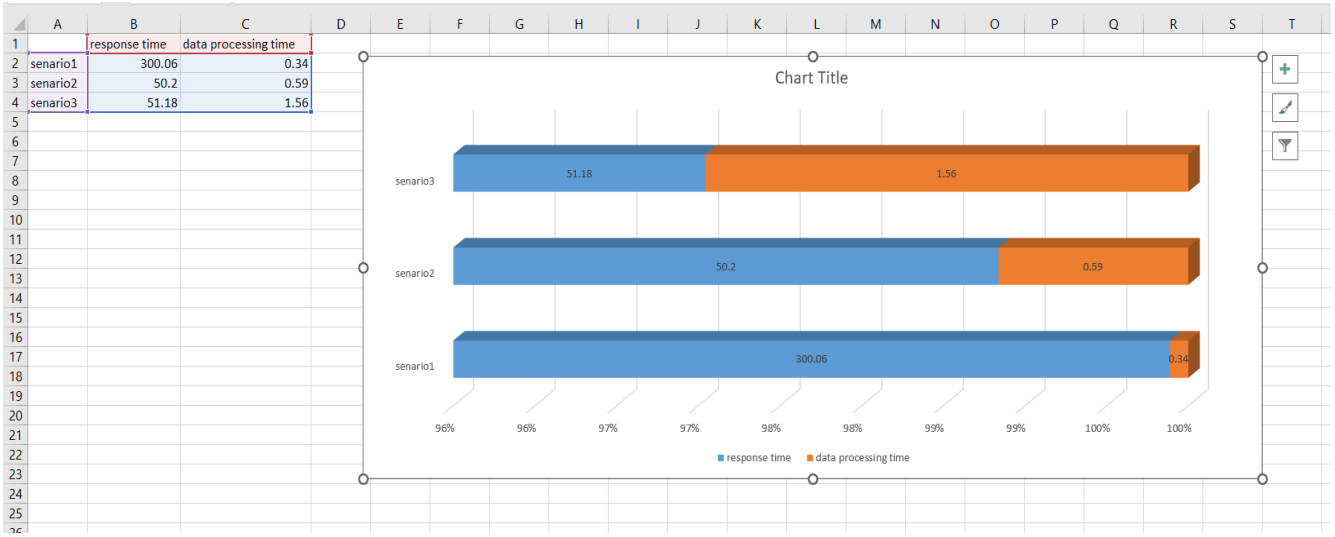


Cost

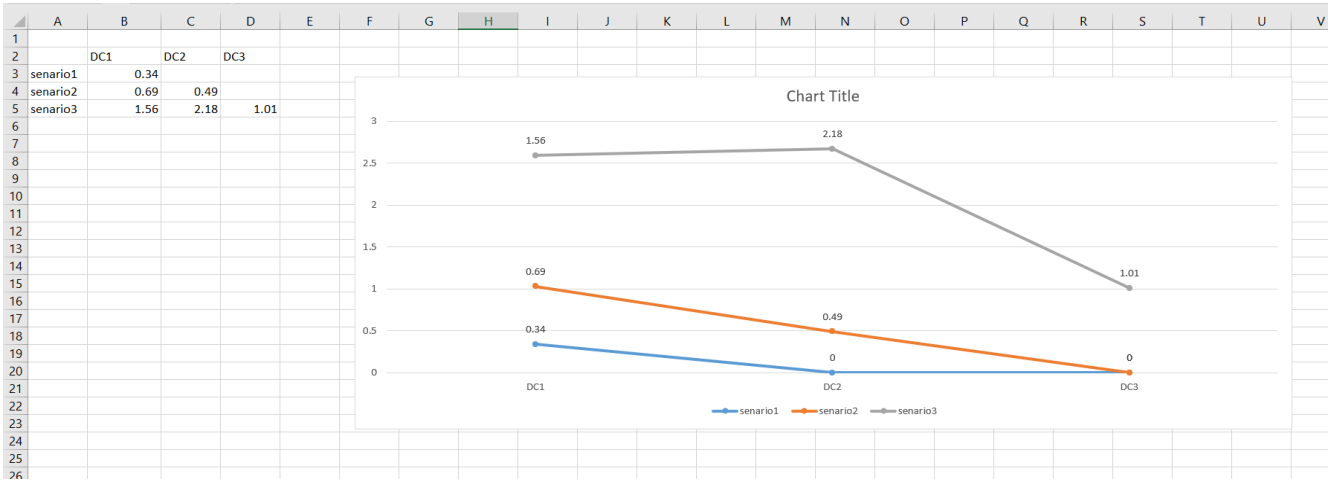
Total Virtual Machine Cost ($):	20.28
Total Data Transfer Cost ($):	0.06
Grand Total: ($)	20.34

Data Center	**VM Cost $**	**Data Transfer Cost $**	**Total $**
DC2	7.61	0.02	7.63
DC1	10.14	0.02	10.16
DC3	2.54	0.02	2.56

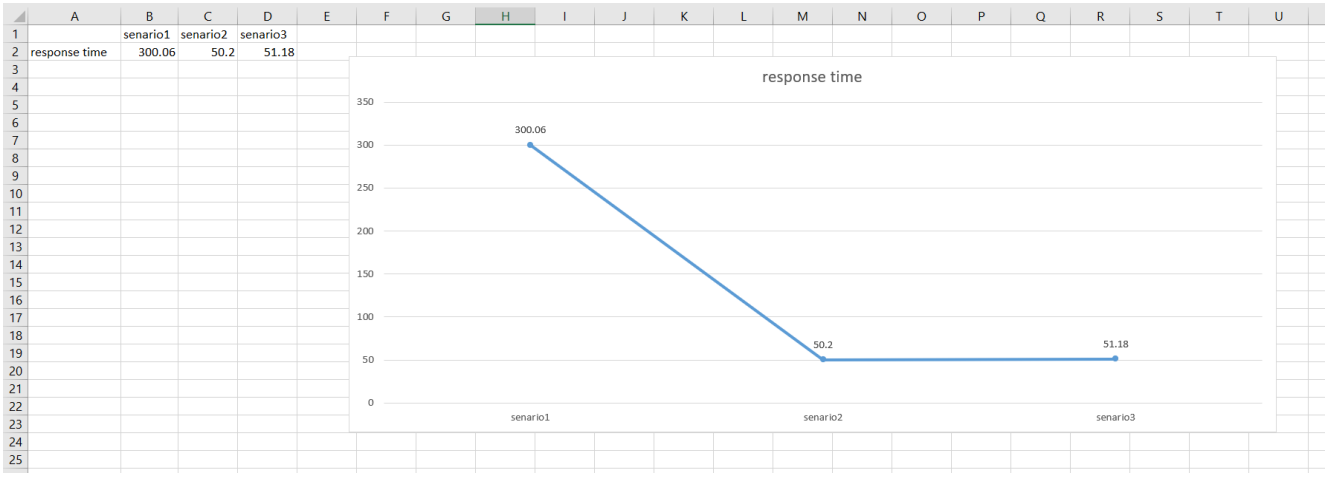
i) Tabulate the overall response time and data processing of all the scenarios and plot the bar graph



ii) Plot a line graph of data center request servicing time of all the data centers for all the scenarios



iii) Compare average response time by regions of all scenarios by plotting line graph



iv) Mention the data centers used by the UB2,UB3, UB4 and UB5

The data centers used by the UB2,UB3, UB4 and UB5 is zero(0).

3b)

Scenario 1

Results of the Simulation Completed at: 27/11/2023 15:26:50

Overall Response Time Summary

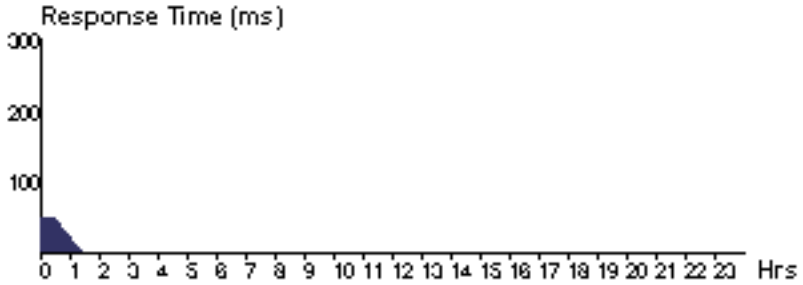
	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	126.75	40.66	378.76
Data Center processing time:	2.64	0.23	4.55

Response Time by Region

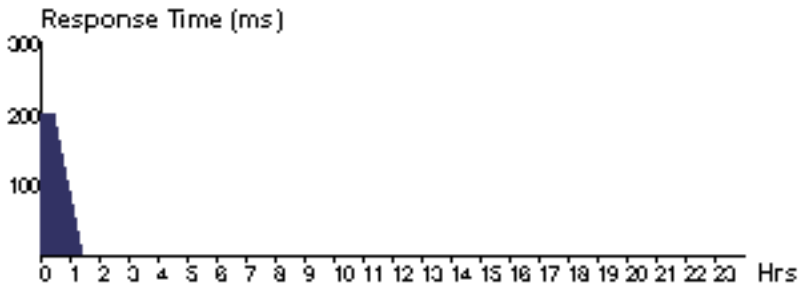
Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	51.56	40.66	63.55
UB2	201.66	151.82	254.76
UB3	53.76	41.30	67.08
UB4	302.60	252.77	378.76

User Base Hourly Response Times

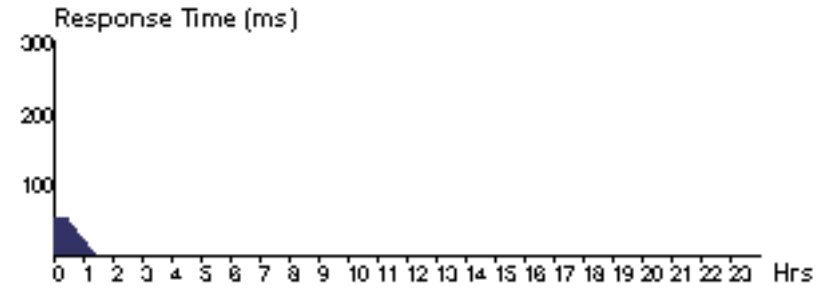
UB1



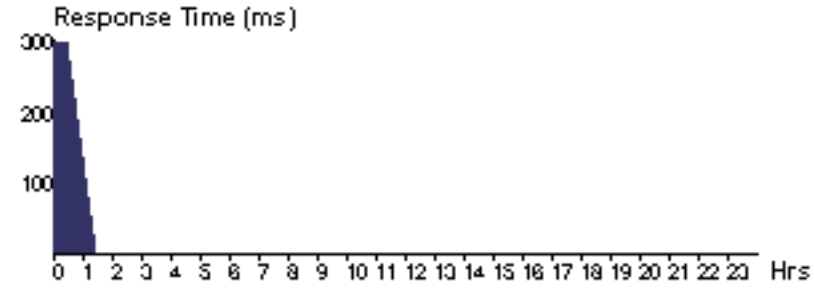
UB2



UB3



UB4

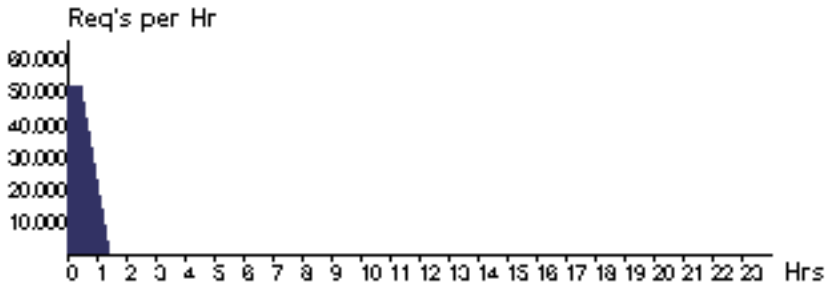


Data Center Request Servicing Times

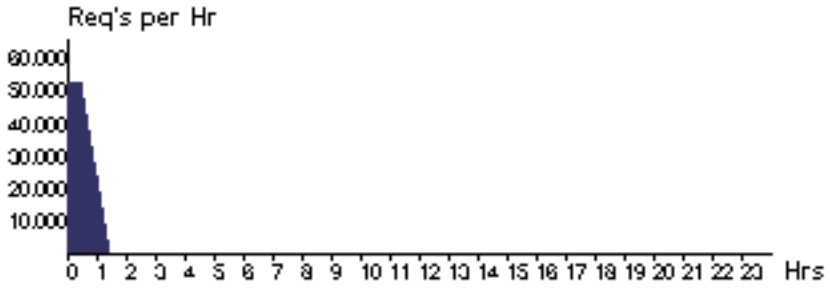
Data Center	**Avg (ms)**	**Min (ms)**	**Max (ms)**
DC1	0.78	0.23	1.40
DC3	2.64	0.59	3.28
DC4	4.11	0.38	4.55

Data Center Hourly Loading

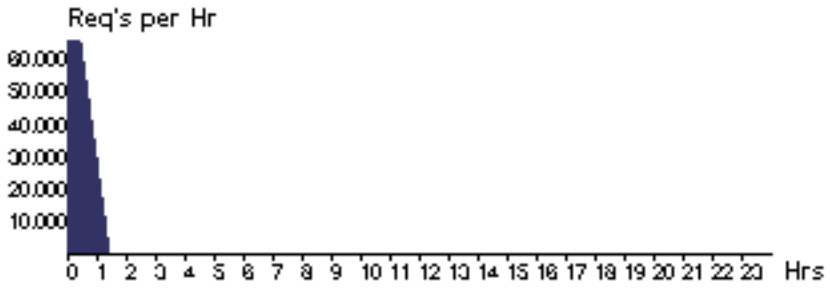
DC1



DC3



DC4



Cost

Total Virtual Machine Cost ($):	30.11
Total Data Transfer Cost ($):	1.78
Grand Total: ($)	31.89

Data Center	**VM Cost $**	**Data Transfer Cost $**	**Total $**
DC1	5.02	0.55	5.56
DC4	15.06	0.69	15.74
DC3	10.04	0.55	10.59

Scenario 2

Results of the Simulation Completed at: 27/11/2023 15:28:33

Overall Response Time Summary

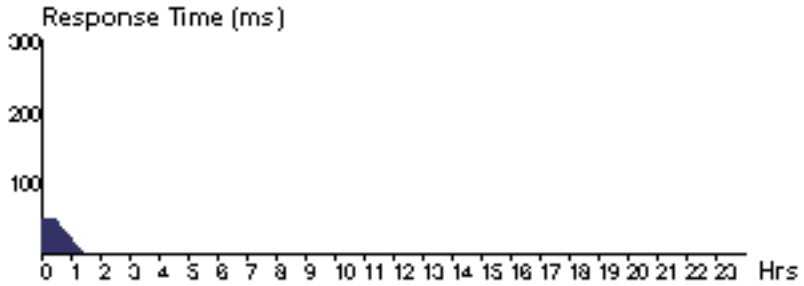
	Avg (ms)	**Min (ms)**	**Max (ms)**
Overall response time:	126.77	38.92	371.26
Data Center processing time:	2.58	0.22	4.55

Response Time by Region

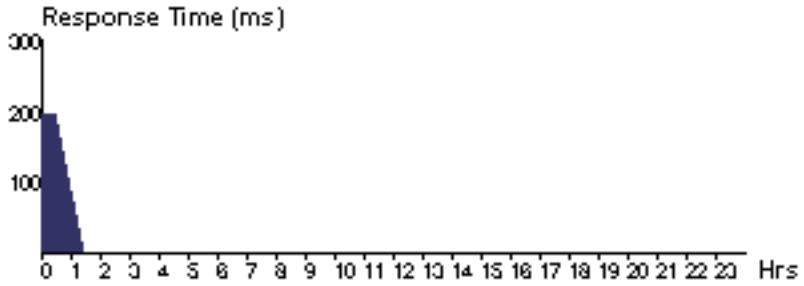
Userbase	**Avg (ms)**	**Min (ms)**	**Max (ms)**
UB1	51.32	38.92	65.05
UB2	201.70	153.66	253.63
UB3	53.78	41.79	67.82
UB4	303.53	248.27	371.26

User Base Hourly Response Times

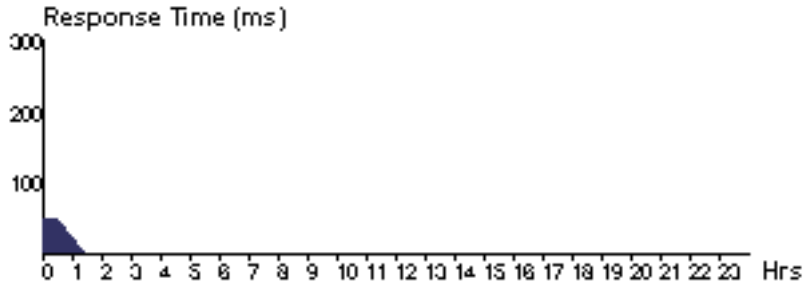
UB1



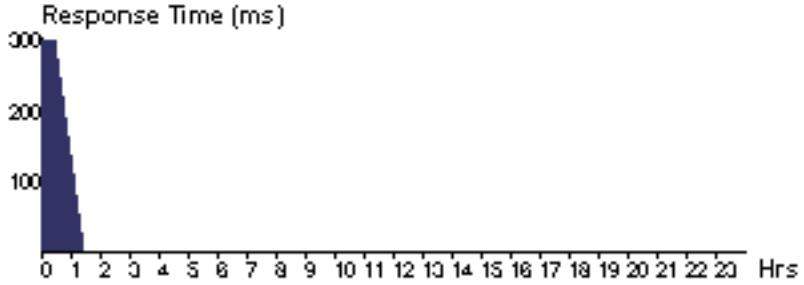
UB2



UB3



UB4

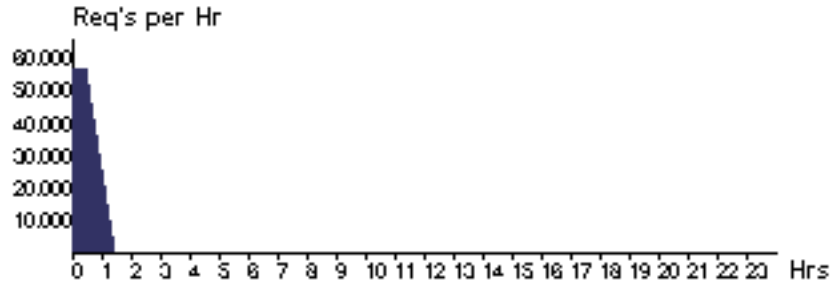


Data Center Request Servicing Times

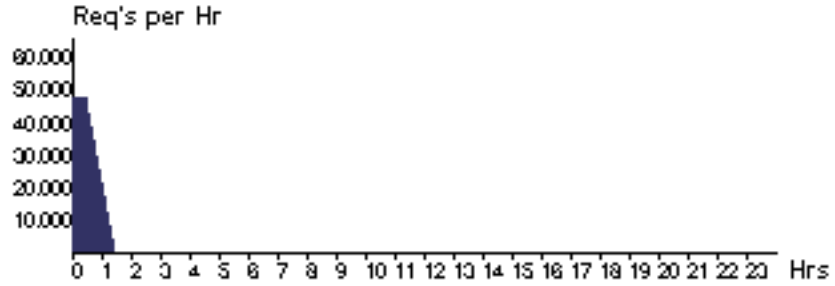
Data Center	**Avg (ms)**	**Min (ms)**	**Max (ms)**
DC1	0.79	0.22	1.41
DC3	2.63	0.60	3.28
DC4	4.11	0.38	4.55

Data Center Hourly Loading

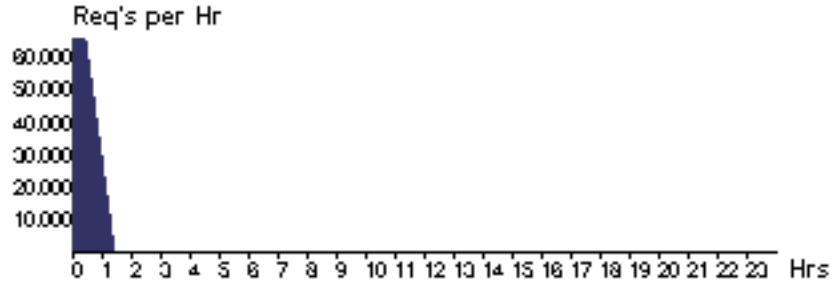
DC1



DC3



DC4

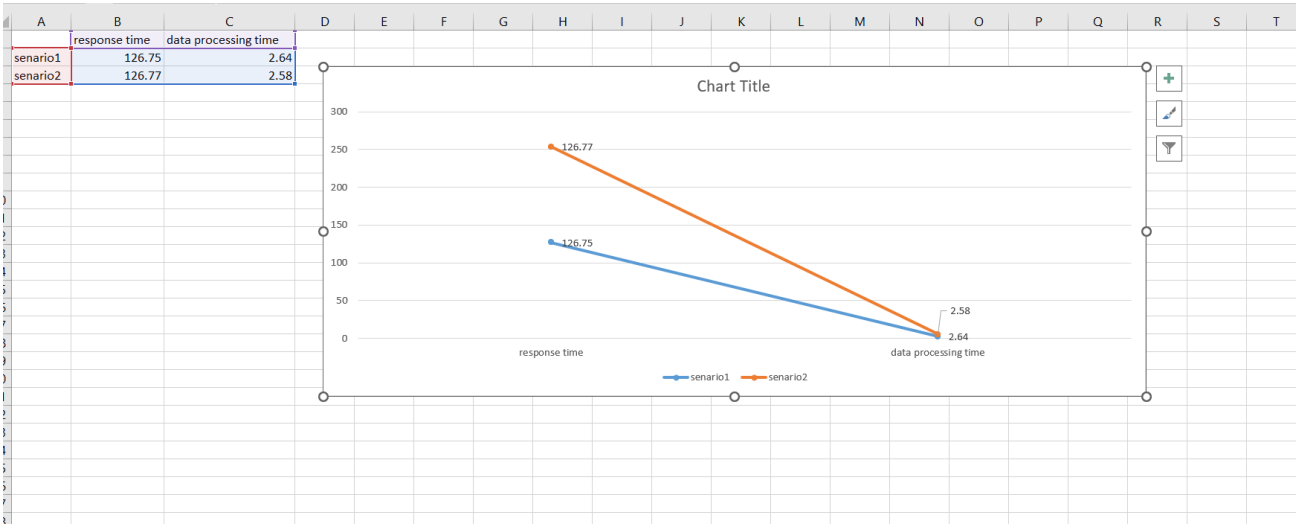


Cost

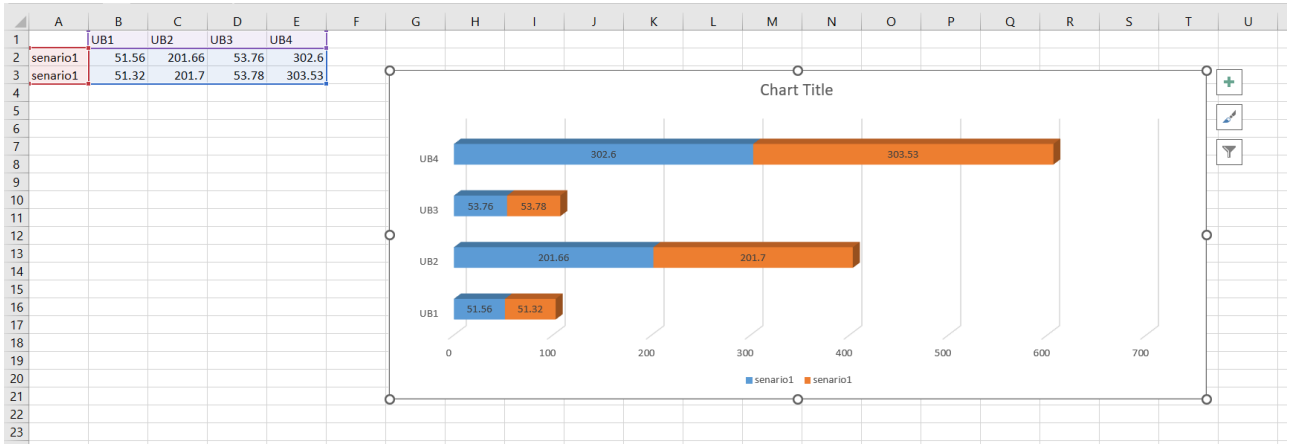
Total Virtual Machine Cost ($):	30.11
Total Data Transfer Cost ($):	1.78
Grand Total: ($)	31.89

Data Center	**VM Cost $**	**Data Transfer Cost $**	**Total $**
DC1	5.02	0.60	5.62
DC4	15.06	0.69	15.74
DC3	10.04	0.50	10.54

i) Tabulate and compare the Average response time and data processing time of all the scenarios by plotting the line graph



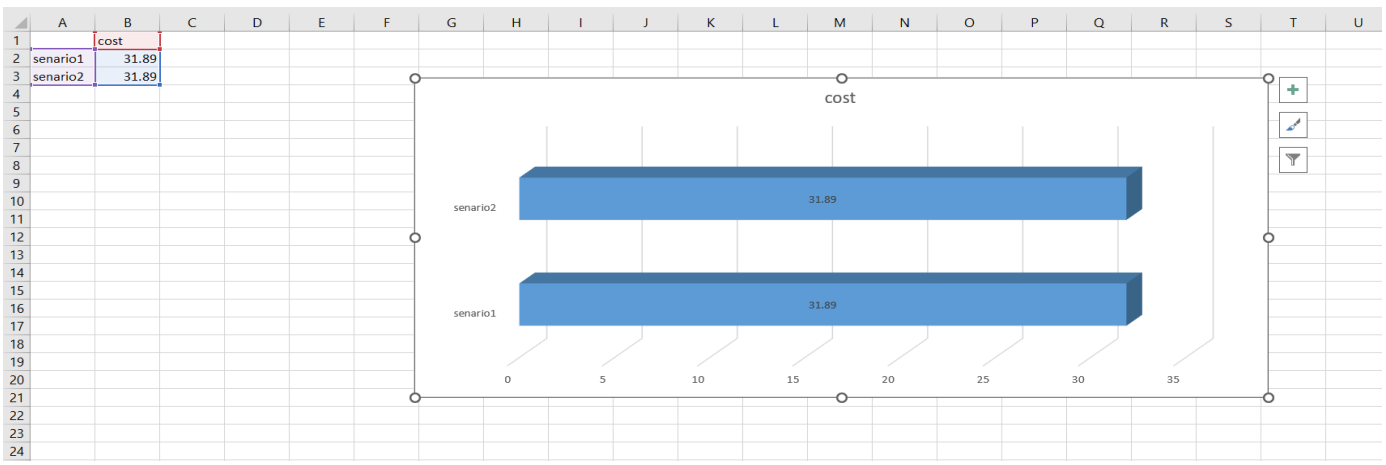
ii) Tabulate the response time of user bases in all scenarios and compare these by plotting bar graph. Which user base is taking maximum time among two scenarios? Why



UB4 user base is taking maximum time among two scenarios because it has less users at the peak hours.

iii) Calculate the data transmission time from DC1 to UB2

iv) Plot the bar graph for data center cost of all scenarios



4b)

Scenario 1

Results of the Simulation Completed at: 27/11/2023 15:38:13

Overall Response Time Summary

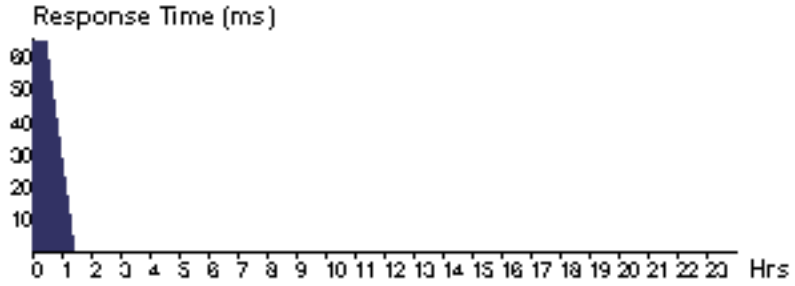
	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	65.48	44.82	100.78
Data Center processing time:	15.89	0.05	48.25

Response Time by Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	65.48	44.82	100.78

User Base Hourly Response Times

UB1

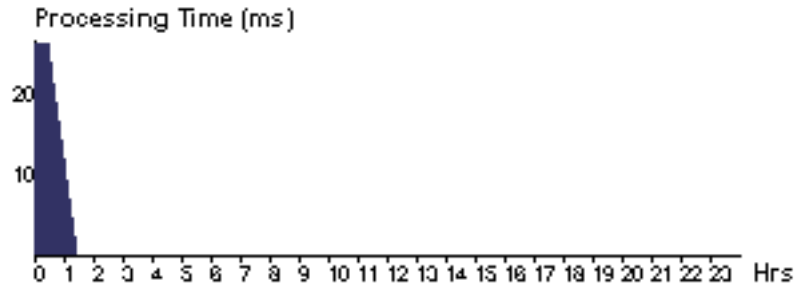


Data Center Request Servicing Times

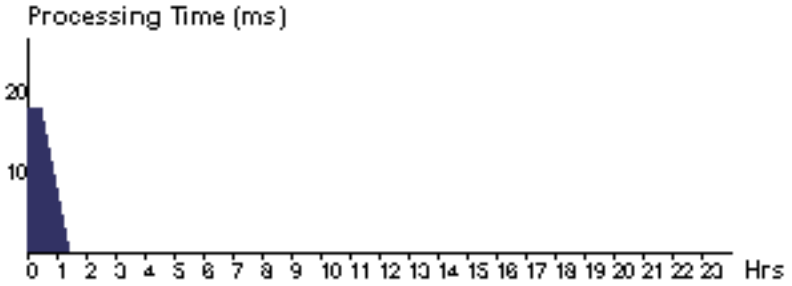
Data Center	Avg (ms)	Min (ms)	Max (ms)
DC1	26.23	1.68	48.25
DC2	18.32	0.82	31.38
DC3	2.65	0.05	5.40

Data Center Hourly Average Processing Times

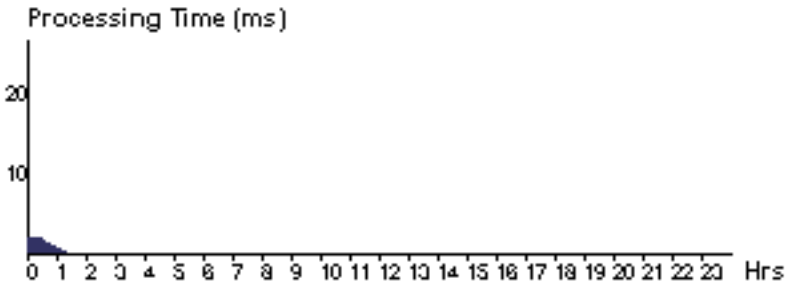
DC1



DC2

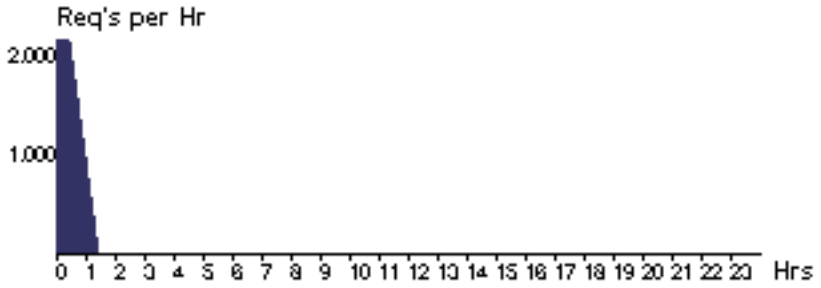


DC3

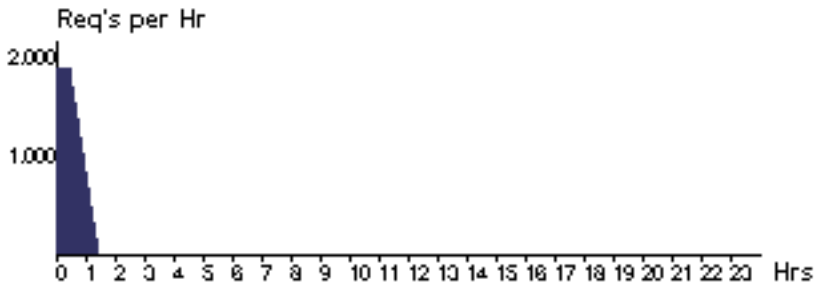


Data Center Hourly Loading

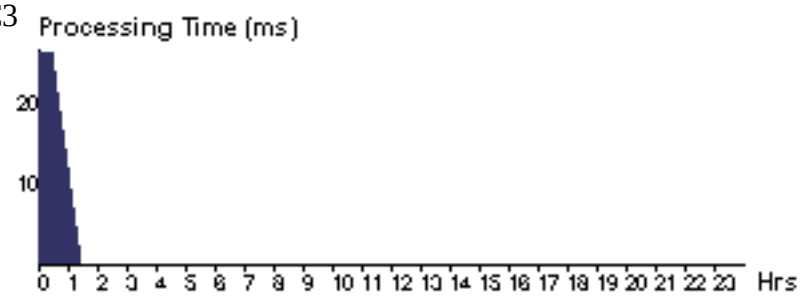
DC1



DC2



DC3

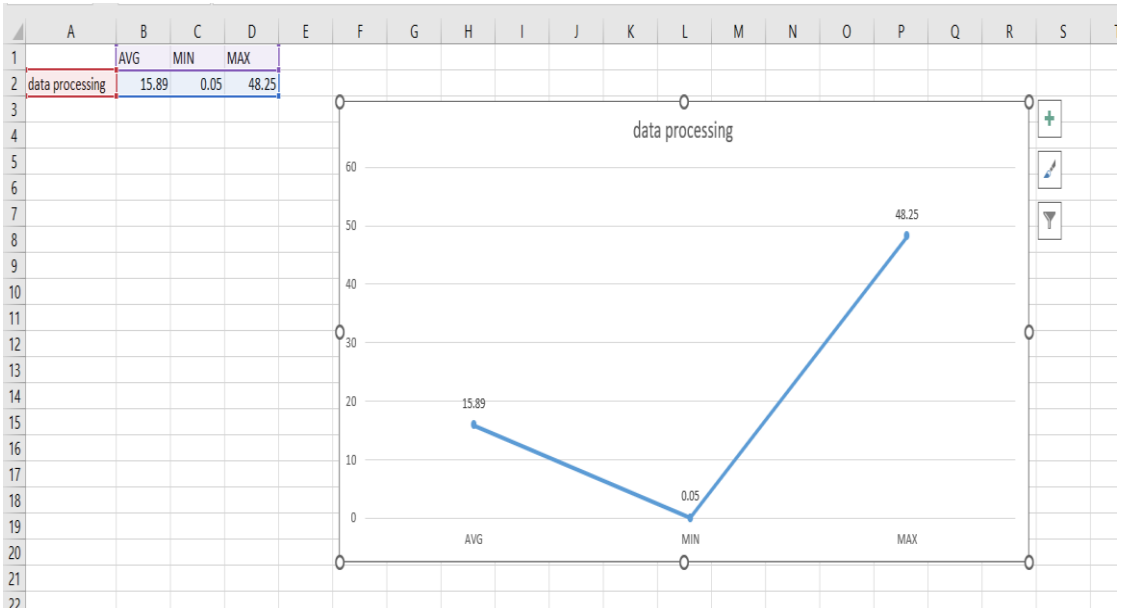


Cost

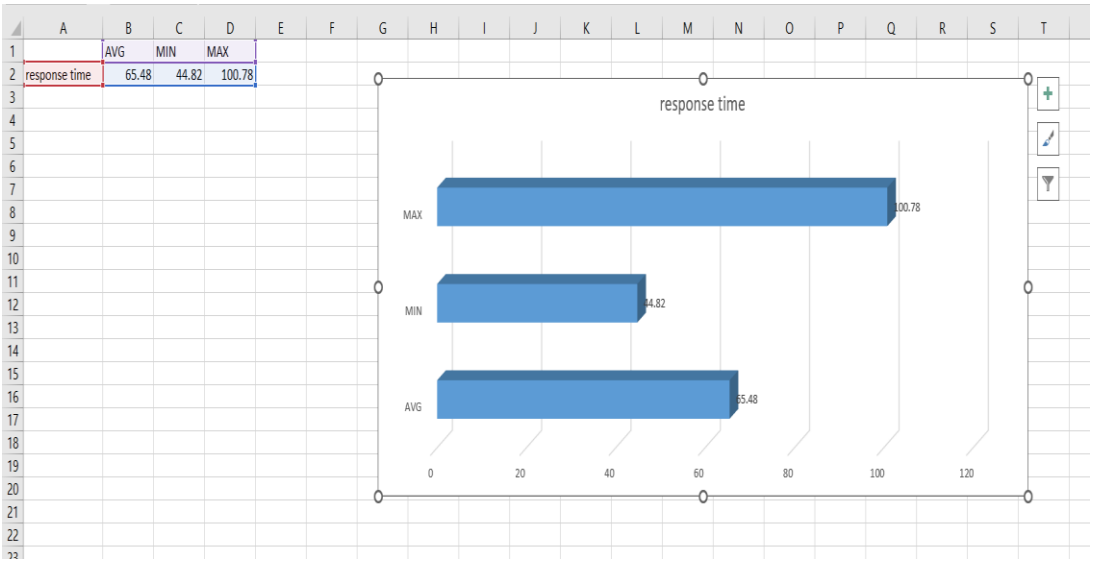
Total Virtual Machine Cost ($):	13.18
Total Data Transfer Cost ($):	0.06
Grand Total: ($)	13.24

Data Center	**VM Cost $**	**Data Transfer Cost $**	**Total $**
DC2	5.07	0.02	5.09
DC1	7.61	0.02	7.63
DC3	0.51	0.02	0.53

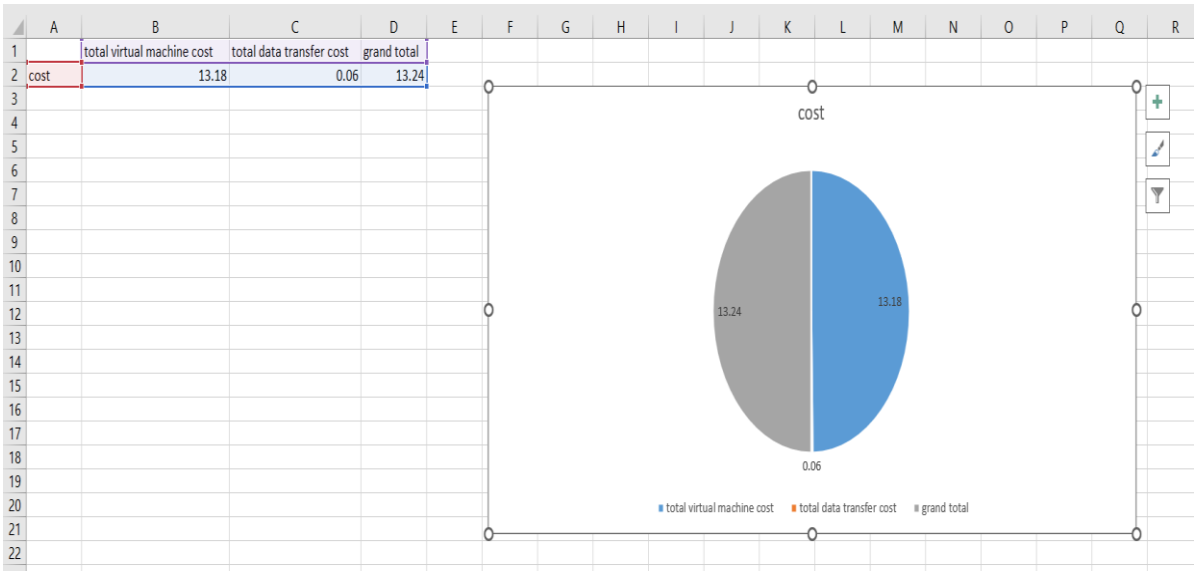
i) Tabulate and compare the data processing time of service broker policies by plotting the line graph



ii) Tabulate and compare response time of service broker policies by plotting the bar graph



iii) Tabulate the cost for service broker policies and represent it using pie chart



iv) Which service broker policy is best and why?
Throttling policy is best because it will enhance the performance of the system .

5b)

Scenario 1

Results of the Simulation Completed at: 27/11/2023 15:54:58

Overall Response Time Summary

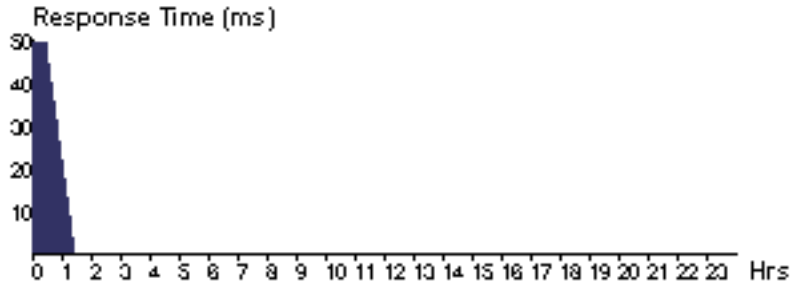
	Avg (ms)	Min (ms)	Max (ms)
Overall response time:	50.15	37.62	60.88
Data Center processing time:	0.48	0.02	0.90

Response Time by Region

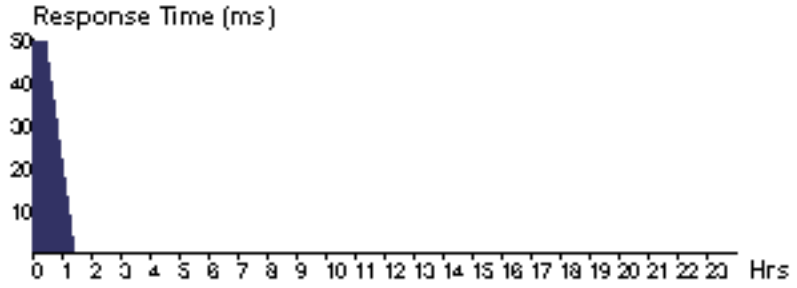
Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.03	37.62	60.12
UB2	50.10	42.13	60.88
UB3	49.97	42.13	60.88
UB4	49.69	39.13	60.38
UB5	50.60	42.66	57.66
UB6	50.51	42.59	59.88

User Base Hourly Response Times

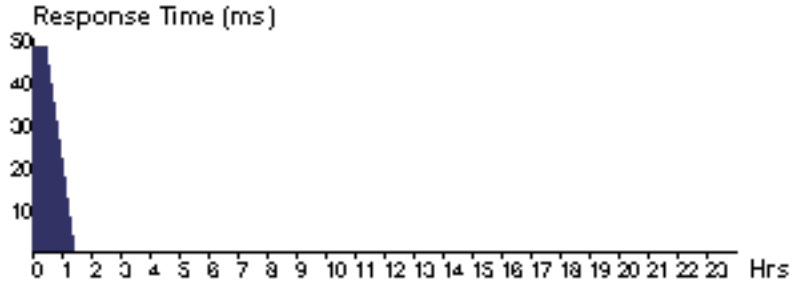
UB1



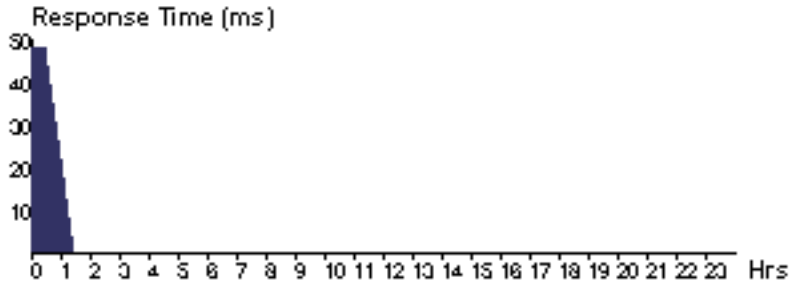
UB2



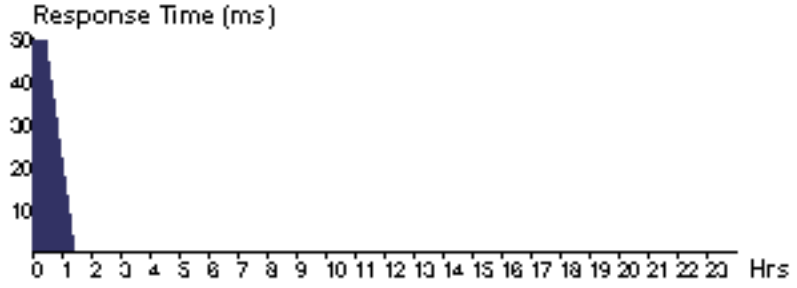
UB3



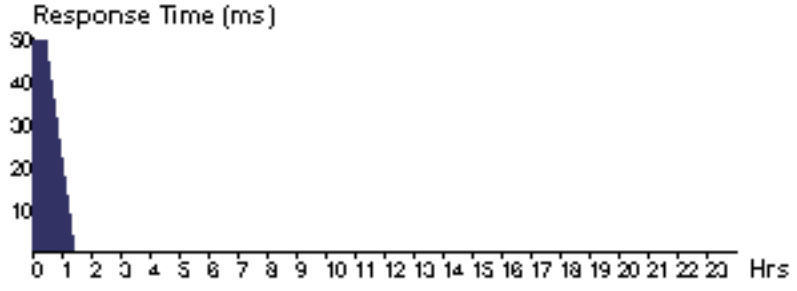
UB4



UB5



UB6

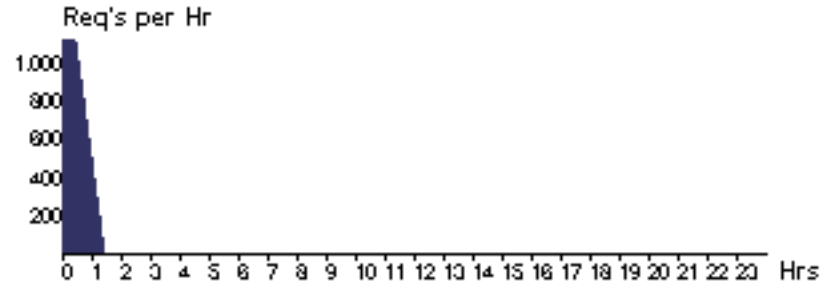


Data Center Request Servicing Times

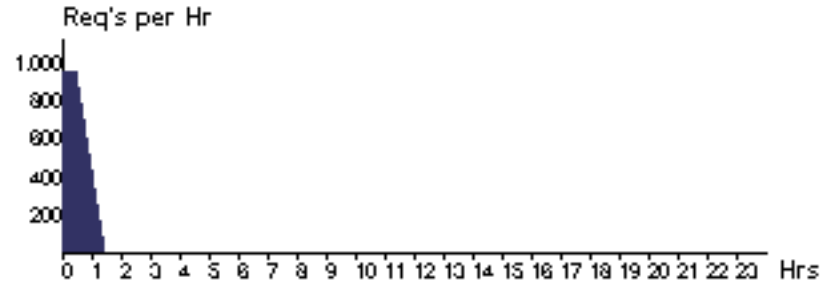
Data Center	Avg (ms)	Min (ms)	Max (ms)
DC10	0.48	0.09	0.88
DC1	0.42	0.03	0.86
DC2	0.47	0.07	0.88
DC3	0.48	0.13	0.88
DC4	0.50	0.02	0.88
DC5	0.48	0.07	0.88
DC6	0.42	0.08	0.88
DC7	0.47	0.13	0.88
DC8	0.53	0.07	0.90
DC9	0.58	0.03	0.89

Data Center Hourly Loading

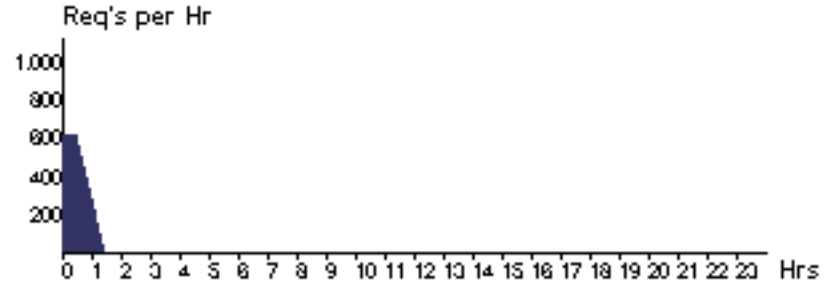
DC1



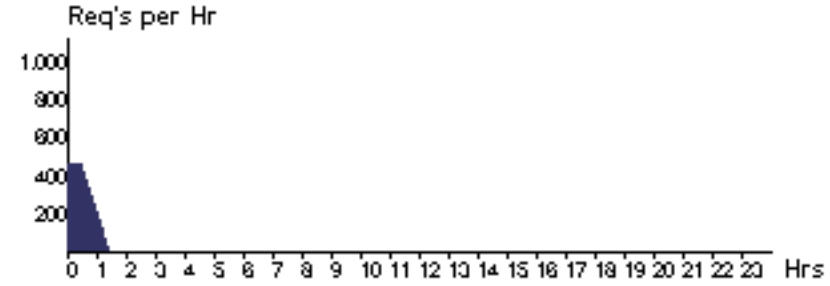
DC10



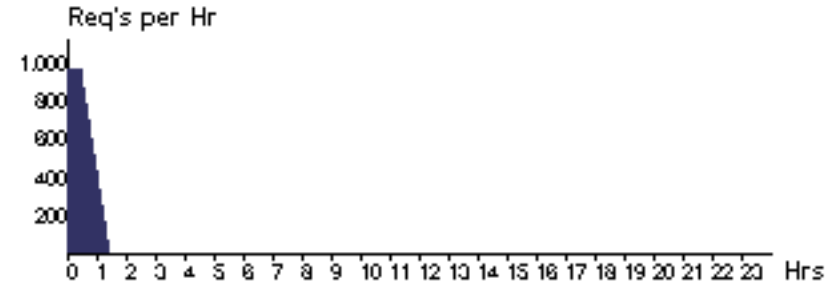
DC2



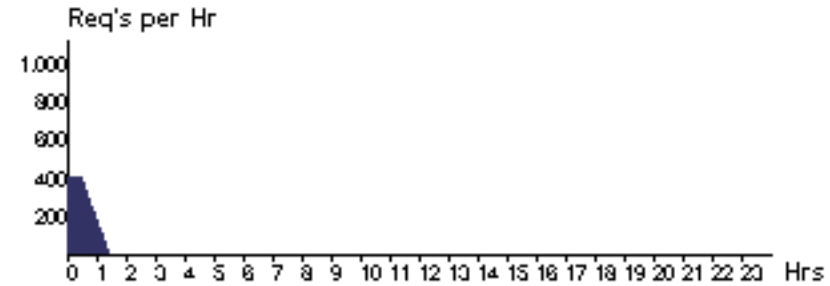
DC3



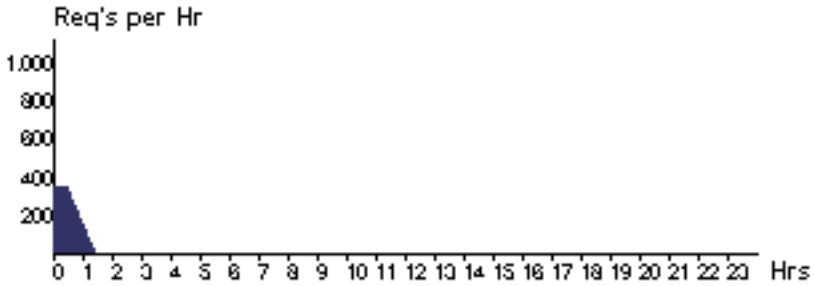
DC4



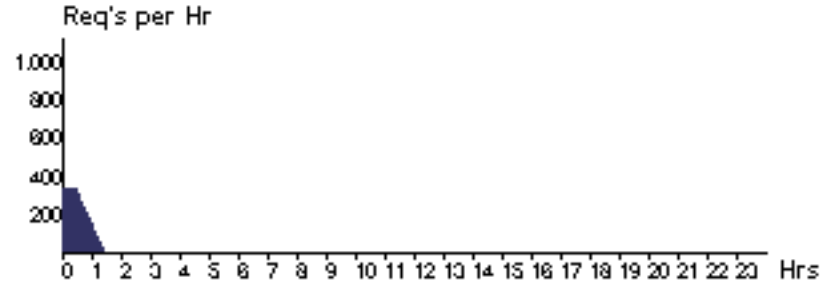
DC5



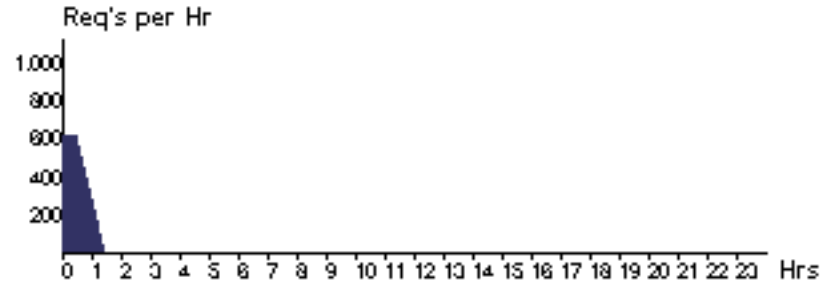
DC6



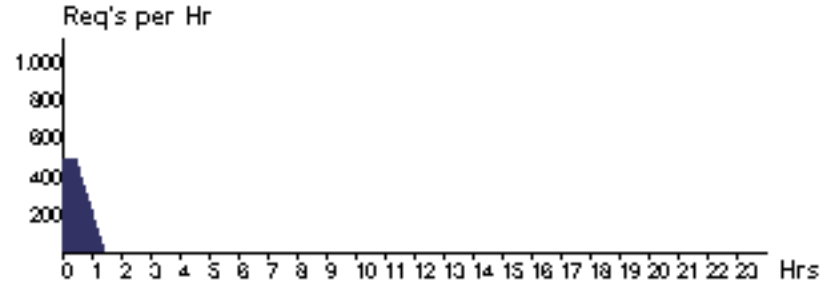
DC7



DC8



DC9

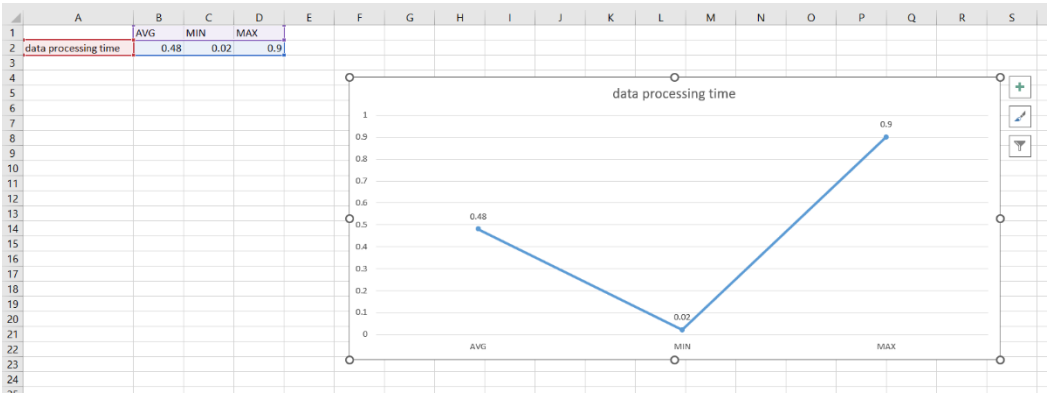


Cost

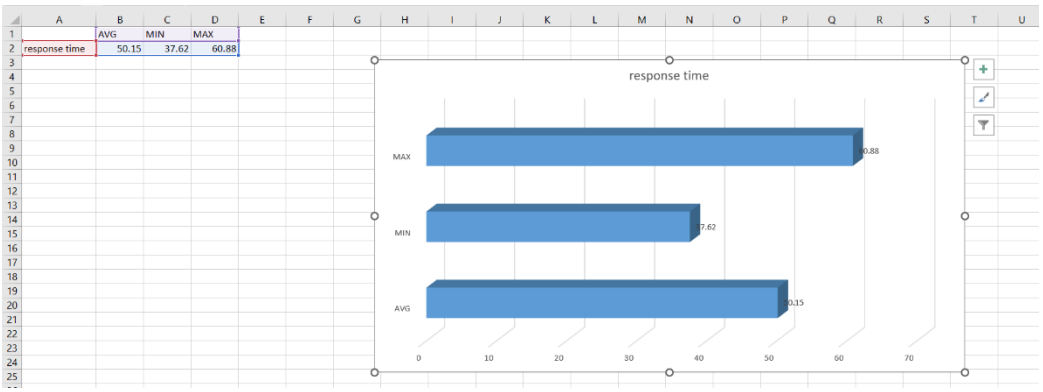
Total Virtual Machine Cost ($):	0.84
Total Data Transfer Cost ($):	0.07
Grand Total: ($)	0.90

Data Center	**VM Cost $**	**Data Transfer Cost $**	**Total $**
DC10	0.08	0.01	0.09
DC2	0.08	0.01	0.09
DC1	0.08	0.01	0.10
DC4	0.08	0.01	0.09
DC3	0.08	0.00	0.09
DC6	0.08	0.00	0.09
DC5	0.08	0.00	0.09
DC8	0.08	0.01	0.09
DC7	0.08	0.00	0.09
DC9	0.08	0.01	0.09

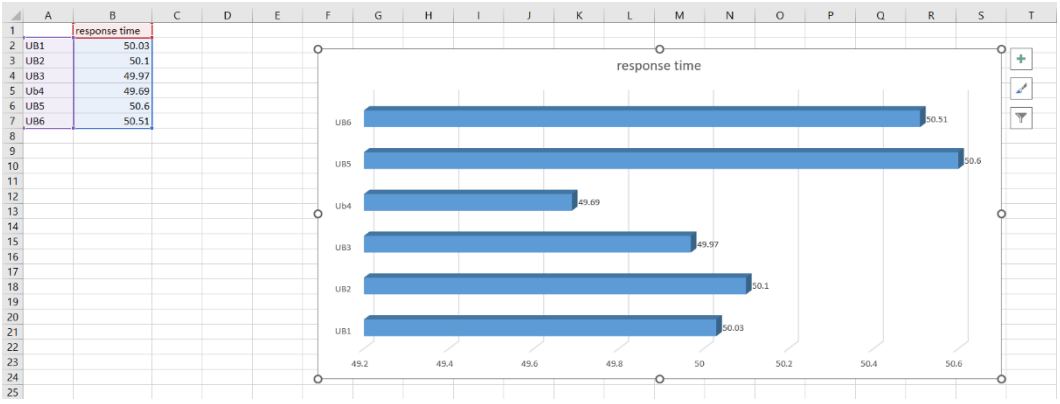
i) Tabulate and compare the data processing time of load balancing algorithms by plotting the line graph



ii) Tabulate the response time of load balancing algorithms by plotting the bar graph



iii) Tabulate the response time by region for load balancing algorithms and plot bar graph



iv) Which load balancing algorithm is best and why?
Round robin algorithm is the best because it is scalable and reliable .

2A - Install Virtualbox/VMware Workstation with different flavours of linux and execute some C programs

Virtual Box

VirtualBox is a powerful x86 and AMD64/Intel64 virtualization product for enterprise as well as home use. Not only is VirtualBox an extremely feature rich, high-performance product for enterprise customers, it is also the only professional solution that is freely available as Open-Source Software under the terms of the GNU General Public License (GPL) version 2. See "About VirtualBox" for an introduction.

Presently, VirtualBox runs on Windows, Linux, Macintosh, and Solaris hosts and supports a large number of guest operating systems including but not limited to Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), DOS/Windows 3.x, Linux (2.4, 2.6, 3.x and 4.x), Solaris and Open Solaris, OS/2, and OpenBSD.

VirtualBox is being actively developed with frequent releases and has an ever- growing list of features, supported guest operating systems and platforms it runs on. VirtualBox is a community effort backed by a dedicated company: everyone is encouraged to contribute while Oracle ensures the product always meets professional quality criteria.

History

VirtualBox was first offered by Innotek GmbH from Weinstadt, Germany, under a proprietary software license, making one version of the product available at no cost for personal or evaluation use, subject to the VirtualBox Personal Use and Evaluation License (PUEL). In January 2007, based on counsel by LiSoG, Innotek GmbH released VirtualBox Open-Source Edition (OSE) as free and open-source software, subject to the requirements of the GNU General Public License (GPL), version 2. Innotek GmbH also contributed to the development of OS/2 and Linux support in virtualization and OS/2 ports of products from Connectix which were later acquired by Microsoft. Specifically, Innotek developed the "additions" code in both Windows Virtual PC and Microsoft Virtual Server, which enables various host–guest OS interactions like shared clipboards or dynamic viewport resizing.

Sun Microsystems acquired Innotek in February 2008.Following the acquisition of Sun Microsystems by Oracle Corporation in January 2010, the product was re- branded as "Oracle VM VirtualBox". In December 2019, VirtualBox started supporting only hardware-assisted virtualization, dropping support for Software-based one.

Features

- Snapshots of the RAM and storage that allow reverting to a prior state.
- Screenshots and screen video capture
- "Host key" for releasing the keyboard and mouse cursor to the host system if captured (coupled) to the guest system, and for keyboard shortcuts to features such as configuration, restarting, and screenshot. By default, it is the right-side CTRL key.
- Mouse pointer integration, meaning automatic coupling and uncoupling of mouse cursor when moved inside and outside the virtual screen, if supported by guest operating system.
- Seamless mode – the ability to run virtualized applications side by side with normal desktop applications Shared clipboard
- Shared folders through "guest additions" software

- Special drivers and utilities to facilitate switching between systems
- Ability to specify amount of shared RAM, video memory, and CPU execution cap
- Ability to emulate multiple screens
- Command line interaction (in addition to the GUI)
- Public API (Java, Python, SOAP, XPCOM) to control VM configuration and execution
- Nested paging for AMD-V and Intel VT (only for processors supporting SLAT and with SLAT enabled)
- Limited support for 3D graphics acceleration (including OpenGL up to (but not including) 3.0 and Direct3D 9.0c via Wine's Direct3D to OpenGL translation)
- SMP support (up to 32 virtual CPUs per virtual machine), since version 3.0
- Teleportation (aka Live Migration)
- 2D video output acceleration (not to be mistaken with video decoding acceleration), since version 3.1
- EFI has been supported since version 3.1 (Windows 7 guests are not supported).

VM Ware

VMware, Inc. is an American cloud computing and virtualization technology company with headquarters in California. VMware was the first commercially successful company to virtualize the x86 architecture. VMware's desktop software runs on Microsoft Windows, Linux, and macOS, while its enterprise software hypervisor for servers, VMware ESXi, is a bare-metal hypervisor that runs directly on server hardware without requiring an additional underlying operating system. VMware's most notable products are its hypervisors. VMware became well known for its first type 2 hypervisor known as GSX. This product has since evolved into two hypervisor product lines: VMware's type 1 hypervisors running directly on hardware and their hosted type 2 hypervisors. VMware software provides a completely virtualized set of hardware to the guest operating system. VMware software virtualizes the hardware for a video adapter, a network adapter, and hard disk adapters. The host provides pass-through drivers for guest USB, serial, and parallel devices. In this way, VMware virtual machines become highly portable between computers, because every host looks nearly identical to the guest. In practice, a system administrator can pause operations on a virtual machine guest, move or copy that guest to another physical computer, and their resume execution exactly at the point of suspension. Alternatively, for enterprise servers, a feature called vMotion allows the migration of operational guest virtual machines between similar but separate hardware hosts sharing the same storage (or, with vMotion Storage, separate storage can be used, too). Each of these transitions is completely transparent to any users on the virtual machine at the time it is being migrated.

VMware's products predate the virtualization extensions to the x86 instruction set, and do not require virtualization-enabled processors. On newer processors, the hypervisor is now designed to take advantage of the extensions. However, unlike many other hypervisors, VMware still supports older processors. In such cases, it uses the CPU to run code directly whenever possible (as, for example, when running user-mode and virtual 8086 mode code on x86). When direct execution cannot operate, such as with kernel-level and real-mode code, VMware products use binary translation (BT) to re-write the code dynamically. The translated code gets stored in spare memory, typically at the end of the address space, which segmentation mechanisms can protect and make invisible. For these reasons, VMware operates dramatically faster than emulators, running at more than 80% of the speed that the virtual guest operating system would run directly on the same hardware. In one study VMware claims a slowdown over native ranging from 0–6 percent for the VMware ESX Server.

Products:

Desktop software

VMware Workstation, introduced in 1999, was the first product launched by VMware. This software suite allows users to run multiple instances of x86 or x86-64 -compatible operating systems on a single physical personal computer. Workstation Pro version 15.5.1 was released in Nov 2019.

VMware Fusion provides similar functionality for users of the Intel Mac platform, along with full compatibility with virtual machines created by other VMware products.

VMware Workstation Player is freeware for non-commercial use, without requiring a license, and available for commercial use with permission. It is similar to VMware Workstation, with reduced functionality.

Server software

VMware ESXi, an enterprise software product, can deliver greater performance than the freeware VMware Server, due to lower system computational overhead. VMware ESXi, as a "bare-metal" product, runs directly on the server hardware, allowing virtual servers to also use hardware more or less directly. In addition, VMware ESXi integrates into VMware vCenter, which offers extra services.

Cloud management software

VMware vRealize Suite – a cloud management platform purpose-built for a hybrid cloud.

VMware Go is a web-based service to guide users of any expertise level through the installation and configuration of VMware vSphere Hypervisor.

VMware Cloud Foundation – Cloud Foundation provides an easy way to deploy and operate a private cloud on an integrated SDDC system.

VMware Horizon View is a virtual desktop infrastructure (VDI) product.

Application management

The VMware Workspace Portal was a self-service app store for workspace management.

Storage and availability

VMware's storage and availability products are composed of two primary offerings:

VMware vSAN (previously called VMware Virtual SAN) is software-defined storage that is embedded in VMware's ESXi hypervisor. The vSphere and vSAN software run on industry-standard x86 servers to form a hyper- converged infrastructure (or HCI). However, network operators need to have servers from HCL (Hardware Compatibility List) to put one into production. The first release, version 5.5, was released in March 2014. The 6th generation, version 6.6, was released in April 2017.New features available in VMware vSAN 6.6 include native data at rest encryption, local protection for stretched clusters, analytics, and optimized solid-state drive performance. The VMWare 6.7 version was released in April 2018, Users now have improved monitoring tools and new workflows, it is closer to feature parity. The vCenter Server Appliance architecture is moving around to an easy deployment method.

VMware Site Recovery Manager (SRM) automates the failover and failback of virtual machines to and from a secondary site using policy-based management.

Networking and security products

VMware NSX is VMware's network virtualization product marketed using the term software-defined data centre (SDDC). The technology included some acquired from the 2012 purchase of Nicira. Software Defined Networking (SDN) allows the same policies that govern Identity and Access Management (IAM) to dictate levels of access to applications and data through a totally converged infrastructure not possible with legacy network and system access methods.

Other products

Workspace ONE allows mobile users to access apps and data.

The VIX (Virtual Infrastructure eXtension) API allows automated or scripted management of a computer virtualized using either VMware's vSphere, Workstation, Player, or Fusion products. VIX provides bindings for the programming languages C, Perl, Visual Basic, VBscript and C#.

Herald is a communications protocol from VMware for more reliable Bluetooth communication and range finding across for mobile devices. Herald code is available under an Open-source license and was implemented in the Australian Government's COVID Safe app for contact tracing on 19 December 2020.

Guest OS

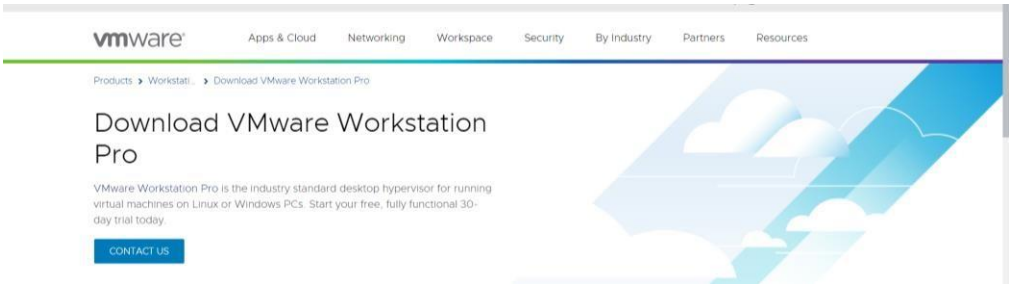
A guest operating system (guest OS) is an operating system (OS) that is secondary to the OS originally installed on a computer, which is known as the host operating system. The guest OS is either part of a partitioned system or part of a virtual machine (VM) setup. A guest OS provides an alternative OS for a device.

In disk partitioning, a guest OS is simply another instance of the same operating system that can boot up for controlling a certain partitioned memory set. A virtual machine (VM) process is much different, in that a guest OS can be a different OS alternative. In VM setups, a guest OS is delivered through a virtual machine environment through a tool called a hypervisor. Again, the machine will typically have a host OS, where the guest OS will operate "within" the host OS. This can lead to limitations on file saving and other operations within the guest OS, depending on whether the guest OS is said to be "persistent."

Part of the emergence of guest operating systems in VM systems has to do with the benefits provided by virtualization. These revolutions in computing coincide with the more general concept of cloud computing, where resources are delivered, rather than hosted, in physical local hardware setups. In addition, a guest OS often takes advantage of a lean OS to build, where memory requirements are further alleviated. VM setups can help with licensing issues, system requirements, and more, making these an attractive part of outsourced computing services.

Step 1: To download and install the VMware product visit the official website of VMware.

https://www.vmware.com/in/products/workstation-pro/workstationproevaluation.html



Step 2: Click on Download VM WorkStation for Windows. The installation file gets downloaded in the specified location and is now ready for installation.

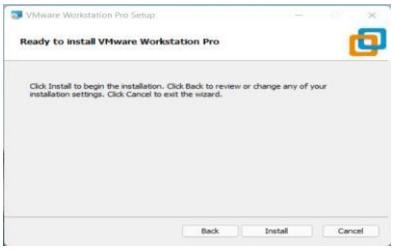
Step 3: Click on the download file to install the VMWare Workstation 16 Pro. Popup will appear

Step 4: Once the initialization gets completed, Click Next

Step 5: Accept the terms and click on Next. In the next screen, it will ask for some additional features, it is not mandatory to check this box. Click on Next.

Step 6: On the next screen, some checkboxes are populated, Check them as per your requirement. Click on Next.

Step 7: At this step, VMware Workstation is ready to install. Click on Install.

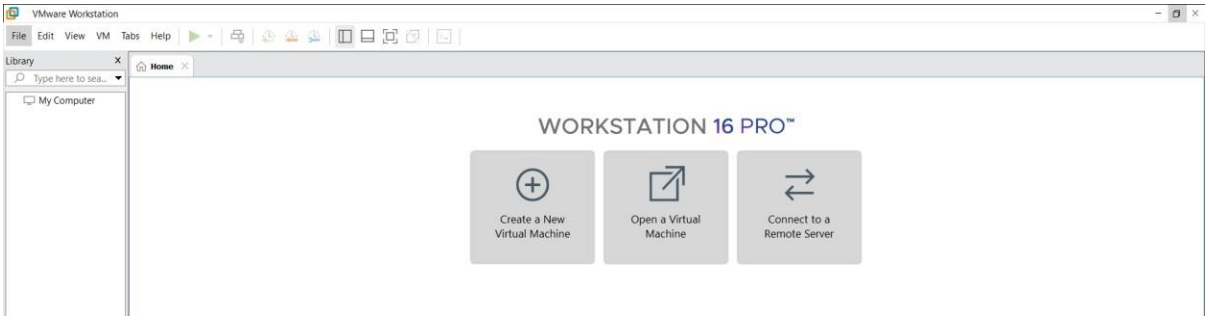


Step 8: Once the installation gets completed you will see the following dialogue box. Click on Finish. If you have purchased the product and have a license key, then you can click on License to enter the key.

Step 9: Upon Finish, the window will close, and You can see VMware Workstation installed icon on your Desktop. Double Click on the Icon to open the application.

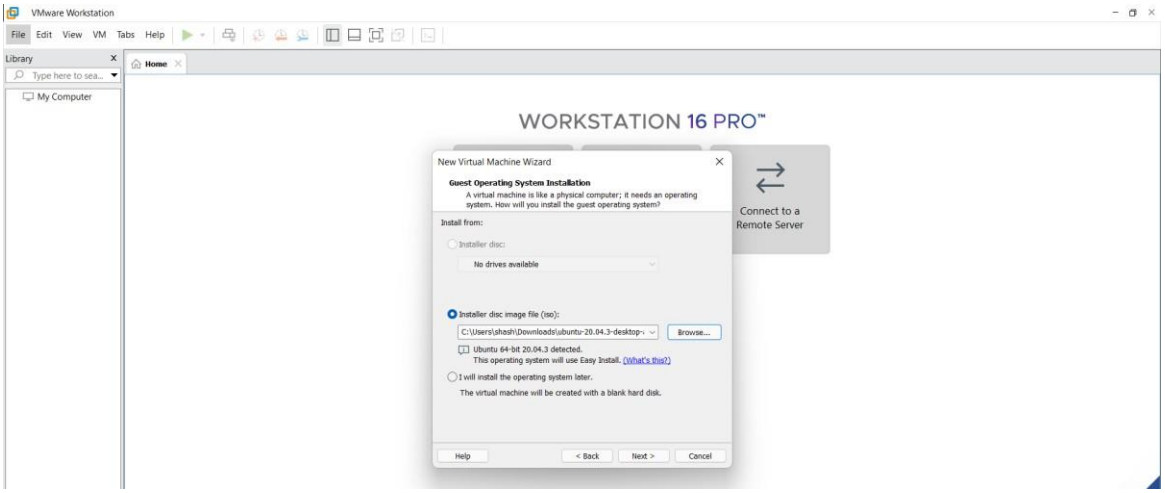
Step 10: For the first time opening, if you have not entered the License key in the previous steps, then it will ask for a license key.

Finally, this will open a window of VMware Workstation Pro.



Step 11: Click on File New Virtual Machine. A New Virtual Machine Wizard will appear. Click on Typical.

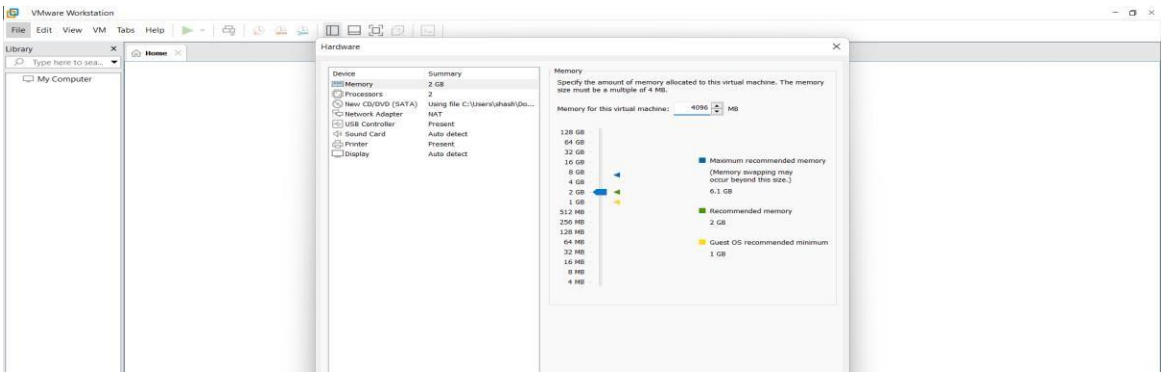
Step 12: Select the ISO File and click on Next



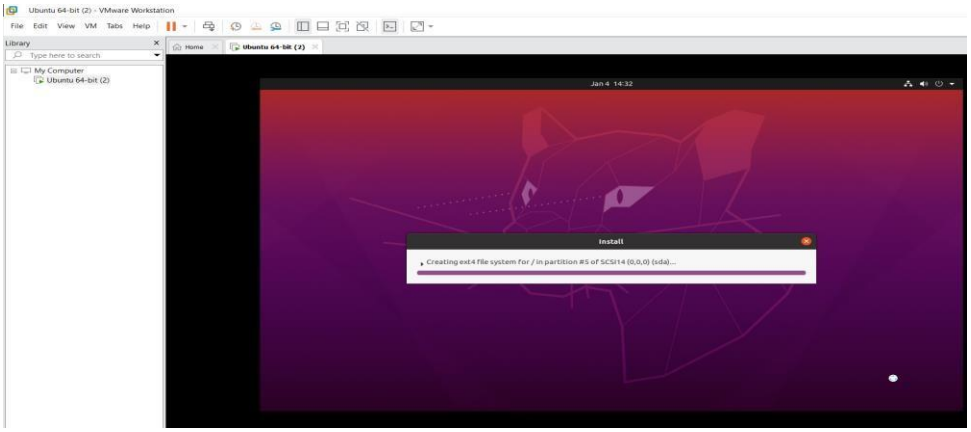
Step 13: Fill in the required details such as username, password and click on next.

Step 14: Name the virtual machines.

Step 15: Allocate the memory and select Split virtual disk into multiple files and click on next. Click on Customize Hardware. Set the memory size to 4GB



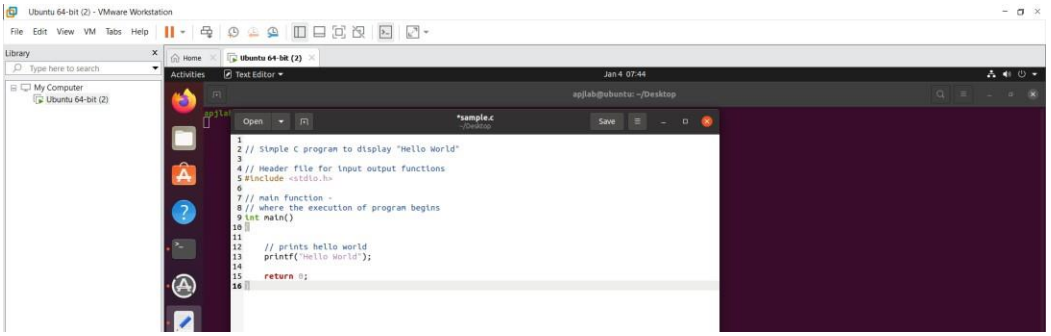
Step 16 : Click on Finish. You can see that Ubuntu gets installed in the VM Ware Workstation.

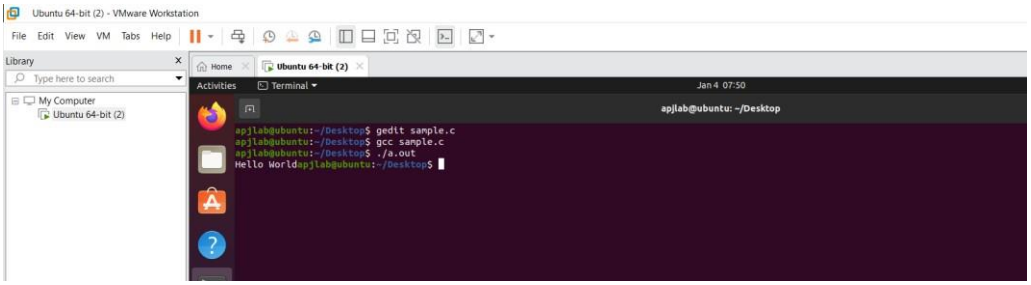


Step 17: Login and Open Terminal and Execute a simple C Program

Step 18: Using gedit write a simple C Code. Use the following commands on the terminal:

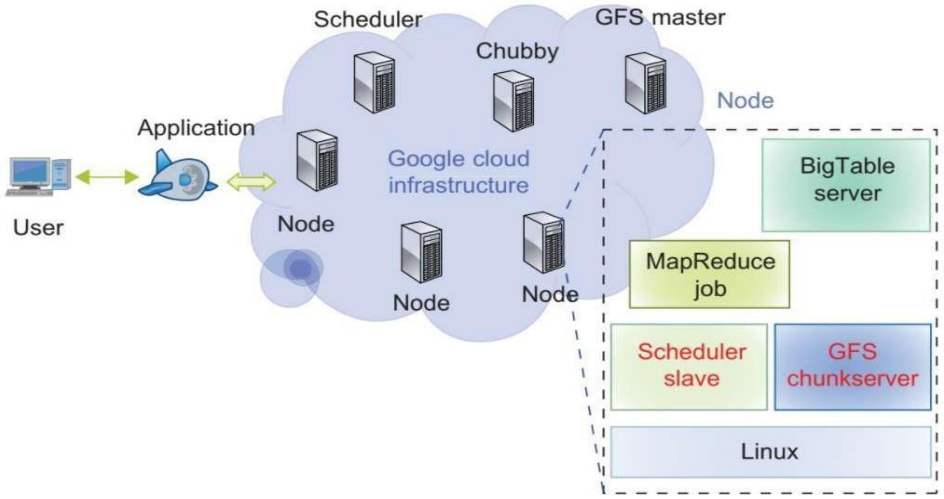
- To Open the Editor - gedit sample.c
- To Compile the code - gcc sample.c
- To Get the output - ./a.out





3A - Install Google App Engine. Create hello world app and other simple web applications using python/java

Google App Engine (often referred to as GAE or simply App Engine) is a cloud computing platform as a service for developing and hosting web applications in Google-managed data centers. Applications are sandboxed and run across multiple servers. App Engine offers automatic scaling for web applications—as the number of requests increases for an application, App Engine automatically allocates more resources for the web application to handle the additional demand. The service is free up to a certain level of consumed resources and only in standard environment but not in flexible environment. Fees are charged for additional storage, bandwidth, or instance hours required by the application. It was first released as a preview version in April 2008 and came out of preview in September 2011.



Google App Engine primarily supports Go, PHP, Java, Python, Node.js, .NET, and Ruby applications, although it can also support other languages via "custom runtimes". Python web frameworks that run on Google App Engine include Django, CherryPy, Pyramid, Flask, web2py and webapp2, as well as a custom Google-written webapp framework and several others designed specifically for the platform that emerged since the release. Any Python framework that supports the WSGI using the CGI adapter can be used to create an application; the framework can be uploaded with the developed application. Third-party libraries written in pure Python may also be uploaded.

Google App Engine supports many Java standards and frameworks. Core to this is the servlet 2.5 technology using the open-source Jetty Web Server, along with accompanying technologies such as JSP. Java Server Faces operates with some workarounds. A newer release of App Engine Standard Java in Beta supports Java8, Servlet 3.1 and Jetty9.Though the integrated database, Google Cloud Datastore, may be unfamiliar to programmers, it is accessed and supported with JPA, JDO, and by the simple low-level API. There are several alternative libraries and frameworks you can use to model and map the data to the database such as Objectify, Slim3 and Jello framework. The Spring Framework works with GAE. However, the Spring Security module (if used) requires workarounds. Apache Struts 1 is supported, and Struts 2 runs with workarounds. The Django web framework and applications running on it can be used on App Engine with modification. Django-nonrel aims to allow Django to work with non-relational databases and the project includes support for App Engine.

Reliability and support

- All billed App Engine applications have a 99.95% uptime SLA.
- App Engine is designed in such a way that it can sustain multiple datacentre outages without any downtime. This resilience to downtime is shown by the statistic that the High Replication Datastore saw 0% downtime over a period of a year.
- Paid support from Google engineers is offered as part of Premier Accounts.

Bulk downloading

SDK version 1.2.2 adds support for bulk downloads of data using Python. The open-source Python projects gaebar, approcket, and gawsh also allow users to download and back up App Engine data. No method for bulk downloading data from GAE using Java currently exists.

Restrictions

- Developers have read-only access to the filesystem on App Engine. Applications can use only virtual filesystems, like gae-filestore.
- App Engine can only execute code called from an HTTP request (scheduled background tasks allow for self-calling HTTP requests).
- Users may upload arbitrary Python modules, but only if they are pure-Python; C and Pyrex modules are not supported.
- Java applications may only use a subset (The JRE Class White List) of the classes from the JRE standard edition. This restriction does not exist with the App Engine Standard Java8 runtime.
- A process started on the server to answer a request can't last more than 60 seconds (with the 1.4.0 release, this restriction does not apply to background jobs anymore).
- Does not support sticky sessions (a.k.a. session affinity), only replicated sessions are supported including limitation of the amount of data being serialized and time for session serialization.

App Engine provides more infrastructure to make it easy to write scalable applications, but can only run a limited range of applications designed for that infrastructure. App Engine's infrastructure removes many of the system administration and development challenges of building applications to scale to hundreds of requests per second and beyond. Google handles deploying code to a cluster, monitoring, failover, and launching application instances as necessary. While other services let users install and configure nearly any *NIX compatible software, App Engine requires developers to use only its supported languages, APIs, and frameworks. Current APIs allow storing and retrieving data from the document-oriented Google Cloud Datastore database; making HTTP requests; sending e-mail; manipulating images; and caching. Google Cloud SQL can be used for App Engine applications requiring a relational MySQL compatible database backend. Per-day and per-minute quotas restrict bandwidth and CPU use, number of requests served, number of concurrent requests, and calls to the various APIs, and individual requests are terminated if they take more than 60 seconds or return more than 32MB of data.

Google App Engine's integrated Google Cloud Datastore database has a SQL-like syntax called "GQL" (Google Query Language). GQL does not support the Join statement. Instead, one-to-many and many-to-many relationships can be accomplished using ReferenceProperty (). Google Firestore is the

successor to Google Cloud Datastore and replaces GQL with a document-based query method that treats stored objects as collections of documents.

Google App Engine Features:

- Blob store for serving large data objects;
- GAE Cloud Storage for storing data objects;
- Page Speed Service for automatically speeding up webpage load times;
- URL Fetch Service to issue HTTP requests and receive responses for efficiency and scaling; and
- Memcache for a fully managed in-memory data store.

Benefits of GAE

- Ease of setup and use. GAE is fully managed, so users can write code without considering IT operations and back-end infrastructure. The built-in APIs enable users to build different types of applications. Access to application logs also facilitates debugging and monitoring in production.
- Pay-per-use pricing. GAE's billing scheme only charges users daily for the resources they use. Users can monitor their resource usage and bills on a dashboard.
- Scalability. Google App Engine automatically scales as workloads fluctuate, adding and removing application instances or application resources as needed.
- Security. GAE supports the ability to specify a range of acceptable Internet Protocol (IP)addresses. Users can allow list specific networks and services and blocklist specific IP addresses.

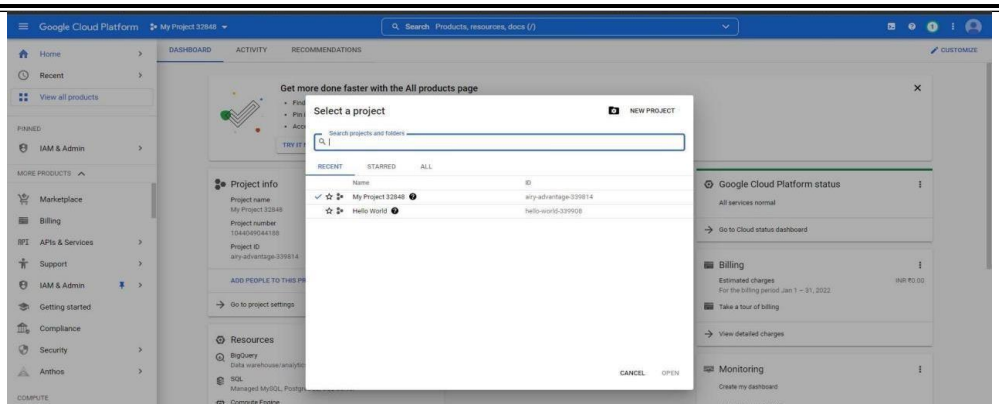
GAE challenges

- Lack of control. Although a managed infrastructure has advantages, if a problem occurs in the back-end infrastructure, the user is dependent on Google to fix it.
- Performance limits. CPU-intensive operations are slow and expensive to perform using GAE. This is because one physical server may be serving several separate, unrelated app engine users at once who need to share the CPU.
- Limited access. Developers have limited, read-only access to the GAE filesystem.
- Java limits. Java apps cannot create new threads and can only use a subset of the Java runtime environment standard edition classes.

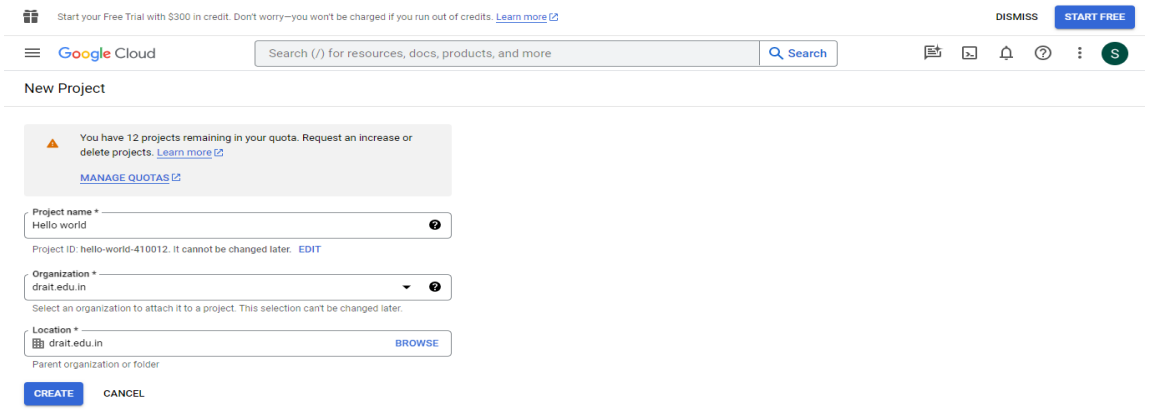
Install Google App Engine. Create hello world app and other simple web applications using python/java

Step 1: Create an account on Google App Engine.

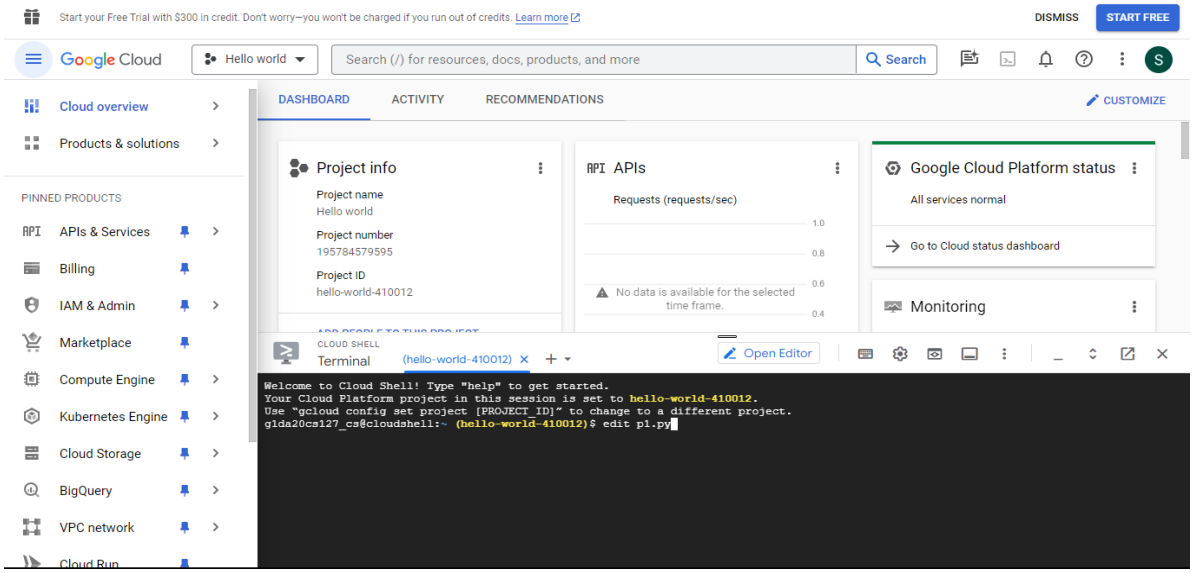
Step 2: Log in to your Google App Engine Account.



Step 3: Create a New Project and give a Project Name and click on create.



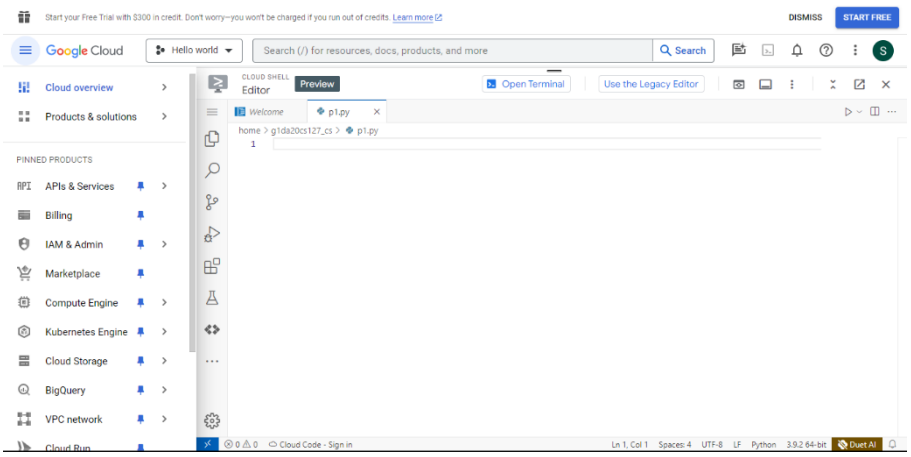
Step 4: Once the project is created click on Activate Cloud Shell on the top right corner.



Step 5: Once the cloud shell gets activated type the following command in the terminal and choose the region as South Asia

gcloud app create

Click on Open Editor.



Step 6: Click on File, then, New File.

Step 7: Name the File as main.py and type the following

code:*from flask import Flask*

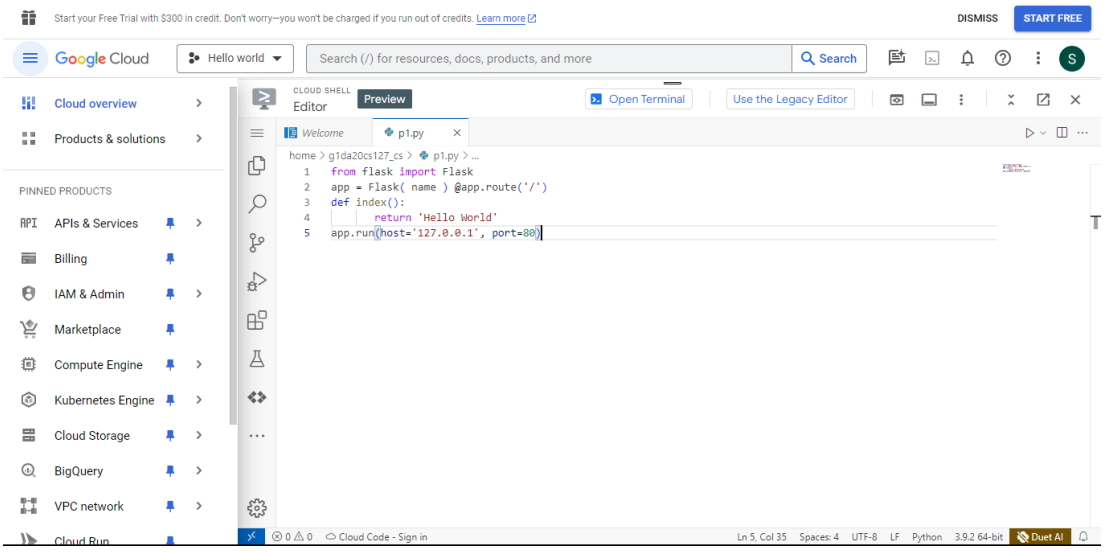
app = Flask(name)

@app.route('/')

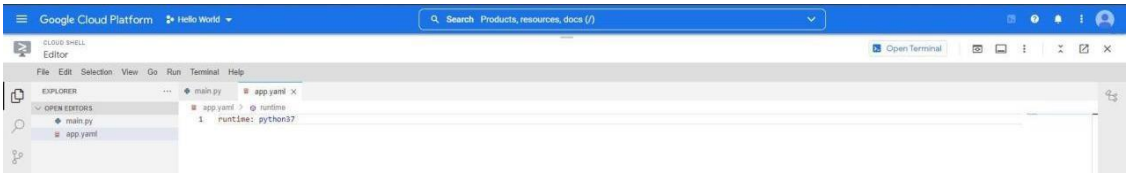
def index():

return 'Hello World'

app.run(host='127.0.0.1', port=80)

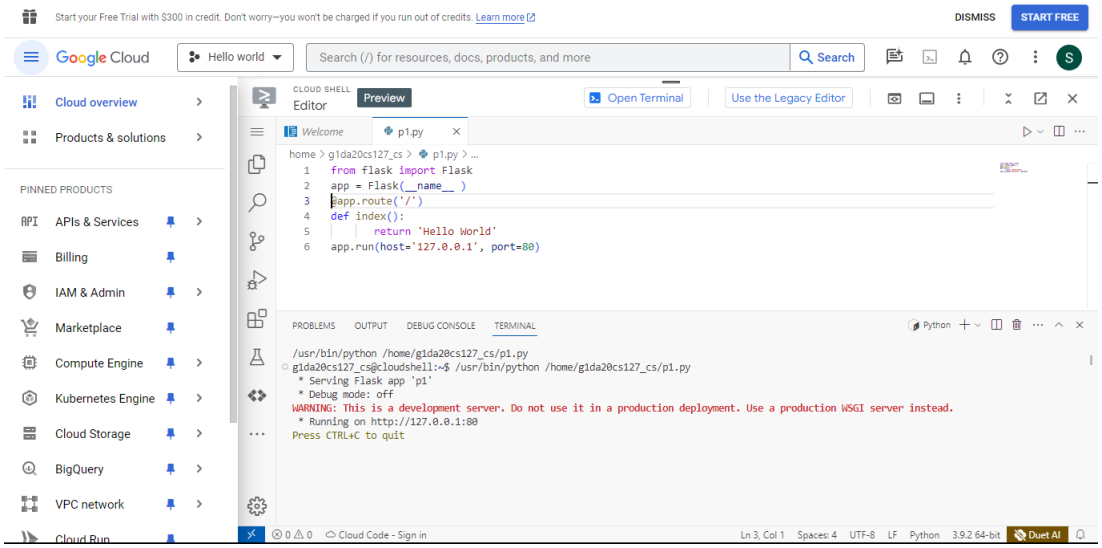


Step8: Create a app.yaml file



Step 9: Go to terminal and type the following command

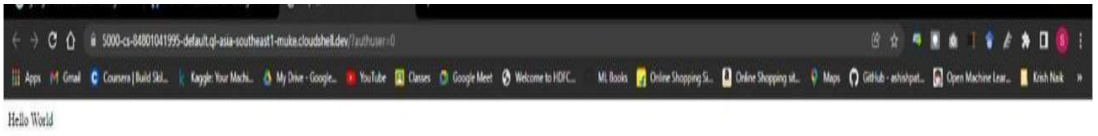
python main.py



Note: If Flask is not installed, please use the following command

pip install flask

Step 10: Open the https link

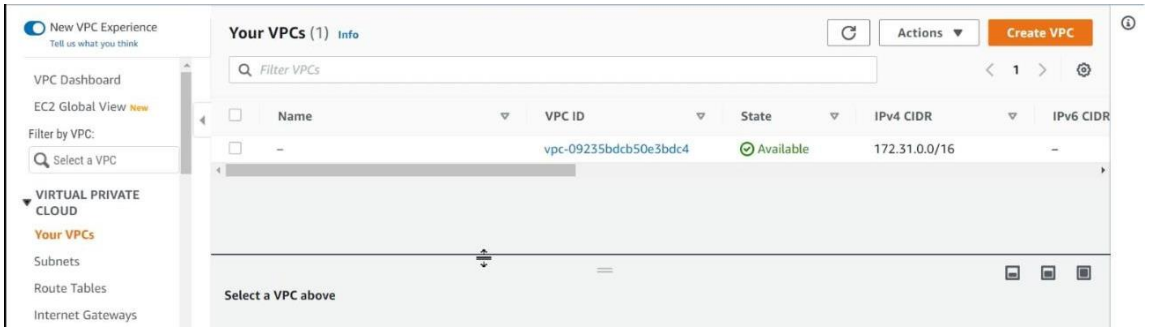


4A - Create a RDS and launch in your custom VPC network

Step 1: Open “AWS Management Console”. Click on “VPC” service.

Step 2: Click “Subnets” on the left panel.

Step 3: Now you can see there is one subnet group (Public Subnet) created in your VPC (Your VPC id/11.0.0.0/16). Now Click on “Create Subnet” button.



Step 4: Give a Name to the subnet and select you own VPC from the “VPC drop down”. Select an “availability zone” (Which is not used by “Public Subnet” of your VPC). Give CIDR block range. Click on “Yes, Create” button.

Step 5: Now you can see the subnet is created in your VPC.

Step 6: Click on “Services”. Click on “RDS” service.

Step 7: Click on “Subnet Groups” on the left panel. (Note: Before creating the subnet groups you need to note down your VPC ID and subnets for that VPC).

Step 8: Click on “Create DB Subnet Group” button.

Step 9: Give a name to the “Subnet”. Select your own “VPC ID”. Select the “Availability Zone” and “subnet”. Click on “Add” button.

Step 10: Now select another “Availability Zone” and “Subnet”. Click on “Yes, Create” button. (Note: Before clicking on “Yes, create” check that two subnets are added or not)

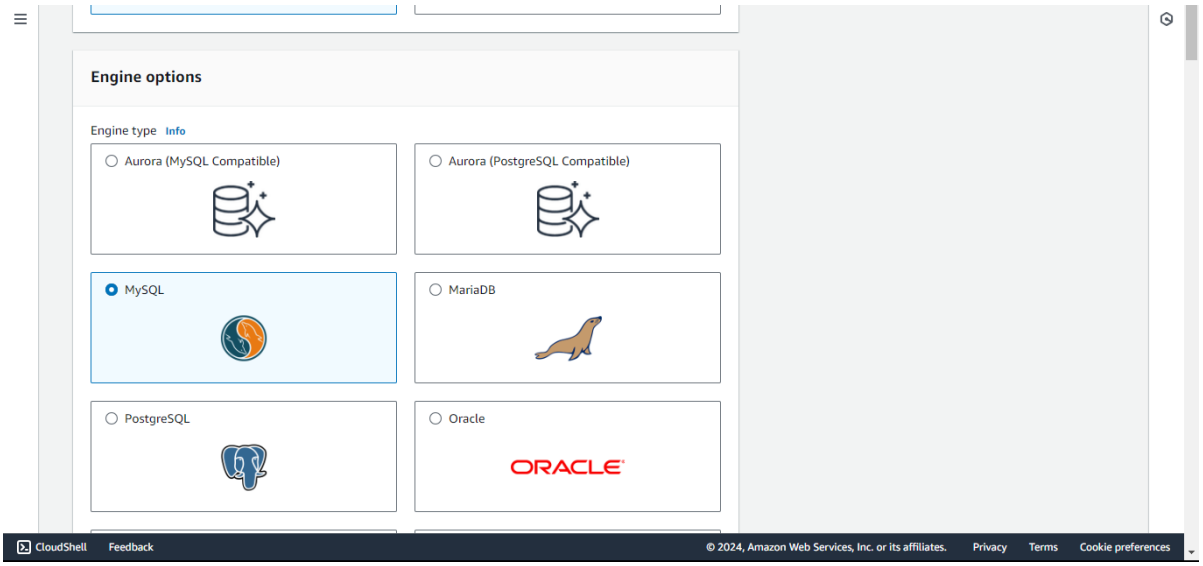
Step 11: Now you can see that “DB subnet group” is created.

Step 12: Click on “RDS Dashboard” on the left panel. Click on “Get Started” button

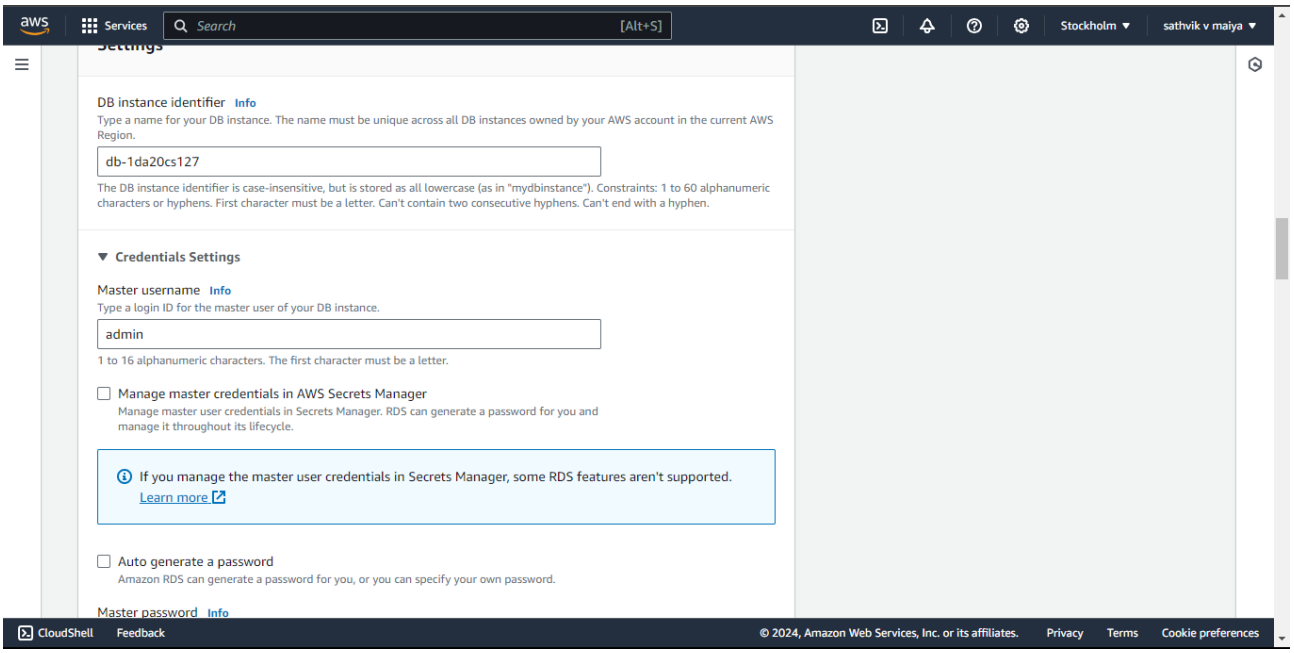
Step 13: Select your desired “Database Engine” from the list and click on “Select” button.

Step 14: If you want multi-AZ Deployment, select the “first radio button” otherwise select the “second radio button”. Click on “Next” button.

Step 15: Select your desired DB instance from the “DB Instance Class” drop down. Select Multi-AZ option from the drop down. Select the storage type from the drop down and give 95 your desired storage space range. Enter the Details (DB instance Identifier, Master user name, password and confirm password). Click on “Next” button.

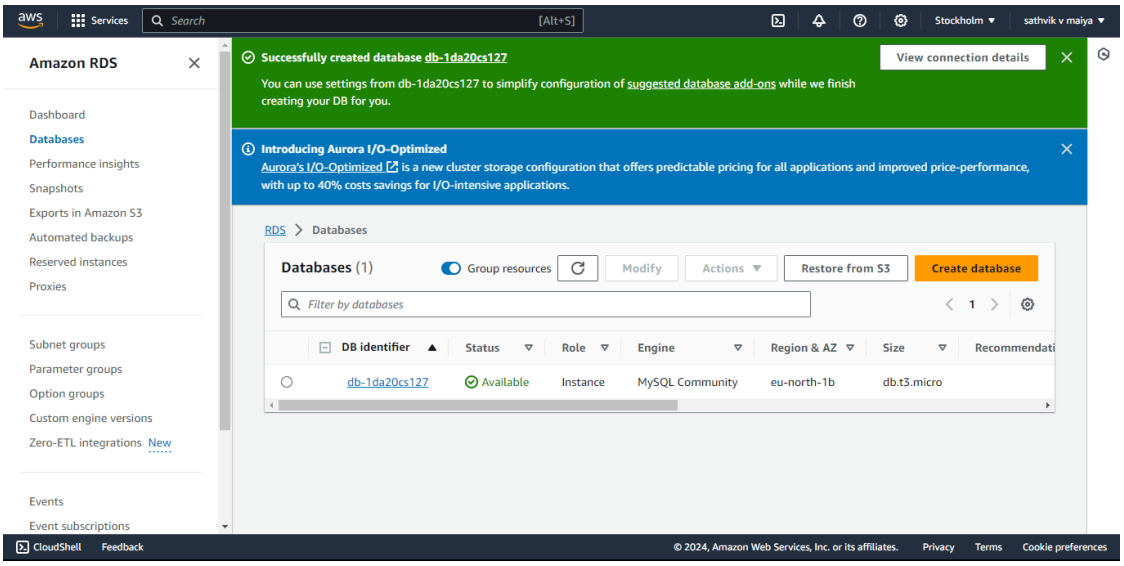


Step 16: Select your own VPC ID from the “VPC Dropdown”. Select the “public accessibility” from the dropdown. Select your desired Availability Zone from the dropdown. Give a name to your Database and check the database port is and mention the “Backup retention period” as per your needs and if you want a Time frame for your backup, configure the time frame from “Backup Window” Dropdown. Click on “Launch DB Instance” button.

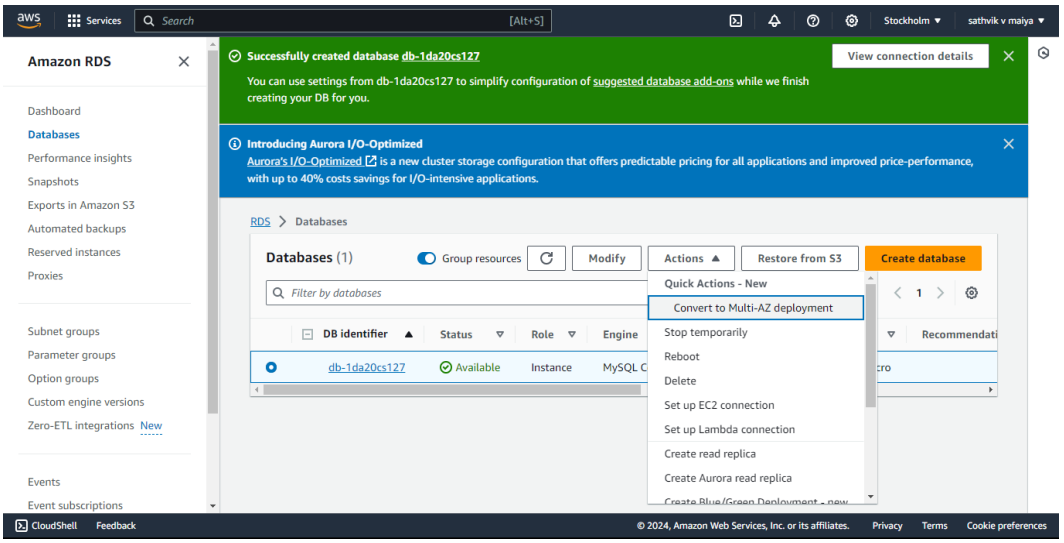


Step 17: Click on “Close” button.

Step 18: Now you can see one DB instance is created. (It will take 5-10min to create the database instance). You will have the DB name, VPC ID, End Point (which is used to connect to the DB Instance from your EC2 instance) and etc…



Note: To connect to the Database from your Ec2 Instance, you need the following.



a). RDS end point.

b) Database Name.

c) Master username.

d) Master Password.

e) Port Number.

5A - Create a file in one virtual machine and transfer/share it with the host machine.

Virtualization and File Sharing

Virtualization

Virtualization is the process of running a virtual instance of a computer system in a layer abstracted from the actual hardware. Most commonly, it refers to running multiple operating systems on a computer system simultaneously. To the applications running on top of the virtualized machine, it can appear as if they are on their own dedicated machine, where the operating system, libraries, and other programs are unique to the guest virtualized system and unconnected to the host operating system which sits below it.

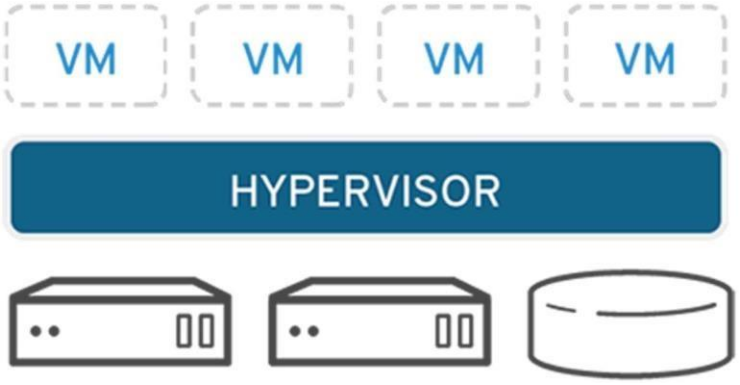
There are many reasons why people utilize virtualization in computing. To desktop users, the most common use is to be able to run applications meant for a different operating system without having to switch computers or reboot into a different system. For administrators of servers, virtualization also offers the ability to run different operating systems, but perhaps, more importantly, it offers a way to segment a large system into many smaller parts, allowing the server to be used more efficiently by a number of different users or applications with different needs. It also allows for isolation, keeping programs running inside of a virtual machine safe from the processes taking place in another virtual machine on the same host.

Working of Virtualization:

Software called hypervisors separate the physical resources from the virtual environments—the things that need those resources. Hypervisors can sit on top of an operating system (like on a laptop) or be installed directly onto hardware (like a server), which is how most enterprises virtualize. Hypervisors take your physical resources and divide them up so that virtual environments can use them.

Resources are partitioned as needed from the physical environment to the many virtual environments. Users interact with and run computations within the virtual environment (typically called a guest machine or virtual machine). The virtual machine functions as a single data file. And like any digital file, it can be moved from one computer to another, opened in either one, and be expected to work the same.

When the virtual environment is running and a user or program issues an instruction that requires additional resources from the physical environment, the hypervisor relays the request to the physical system and caches the changes—which all happens at close to native speed (particularly if the request is sent through an open- source hypervisor based on KVM, the Kernel-based Virtual Machine).



File Sharing

Transferring Files to and from Virtual Machines can be done in the following ways:

- Creating a Shared Folder in VirtualBox
- Dragging and Dropping Files in VirtualBox
- Managing Files with NextCloud

Creating a Shared Folder in VirtualBox

A shared folder is a folder that makes its files available on both the guest machine and the host machine at the same time. Creating a shared folder between the guest and the host allows you to easily manage files that should be present on both machines. The course virtual machines are ready to use shared folders right away, but if you are using the virtual machine on your personal computer, then you will need to specify which folder to use as shared storage.

Shared Folders on SCS Lab Computers using Course VMs:

If you are using a course VM on a lab computer, it is likely that a shared folder has already been set up for you. On the desktop of your course VM, you should notice a folder titled Shared Folders. Inside this folder, you will find any folders that have been shared between the course VM and lab computers. You should see two folders that have already been configured for you: Z_DRIVE and Temp. Z_DRIVE gives you access to your Windows Account Z:\ drive. This is storage that is persistent to your SCS account and available as a network drive on the lab computers. Temp gives you access to the folder found at D:\temp on the lab computer. Files stored in this folder are local to the machine, meaning that they can be accessed faster but will delete from the system when you log out. If you are working with data that you will need to use again, use the Z_DRIVE for your shared folder. If you need a faster read/write speed, use the Temp folder, but remember to back up your files, or they will be deleted when you log off the computer.

Shared Folders on Personal Computers

If you are using your own personal machine, you will need to configure VirtualBox to look in the right place for your shared files. First, click on the guest machine you intend to share files with. From there, you can select the guest Settings and navigate to Shared Folders on the left side menu. To create a new shared folder, either click the New Folder icon on the right menu or right-click the empty list of shared folders and click Add Shared Folder. From here, there are six options:

- **Folder Path**: The folder name on the host machine. Click the drop-down menu and navigate to the folder you would like to share.
- **Folder Name**: This is the name of the folder as it will appear on the guest machine.
- **Read-Only**: If you check read-only, the guest machine will be unable to write changes to the folder. This is valuable when you only want to send files to the virtual machine, but do not want to risk having the files modified by the guest.
- **Auto-Mount**: When any external storage is connected to a computer, it must be mounted in order to be used. It is recommended that you turn on auto-mounting unless you are familiar with the process of mounting a drive yourself.

- **Mount Point**: Unless you already know about mount points, leave this blank.
- **Make Permanent**: If you check this, the shared folder will be a permanent machine folder. If it is not checked, the folder will not be shared after a shutdown.

On the course virtual machines, when you load into the desktop, you should see a folder labelled Shared Folders. In there, you will see any folders that are currently mounted and being shared.

Dragging and Dropping Files in VirtualBox

If you only need to transfer a few files quickly, you can simply drag and drop the files in. On the top bar of the running guest machine, click on Devices > Drag and Drop and make sure that Bidirectional is selected. This means that you will be able to drag files from the host to the guest and from the guest to the host. Once bidirectional drag and drop is checked, you should be able to begin dragging and dropping files.

NOTE: Sometimes when dragging files into the course VM, you may not be able to drag them into the file browser directly. If you encounter this issue, you should drag your files onto the Desktop and move the files around from there. You should see the cursor change when it is ready to drop files.

You can also drag files from the guest machine into the host. To do this, simply open the file browser on the host to where you would like to drop the files and drag the files from the virtual machine into the file browser of the host. File transfers should be pretty quick; if the virtual machine seems stuck when transferring, simply cancel the transfer and try again.

Managing Files with NextCloud

On any virtual machine, including VirtualBox, VMWare, or the virtual machines hosted on the SCS OpenStack, you can access the SCS NextCloud services to move files between multiple machines and your SCS Windows Account storage. NextCloud offers you all of your SCS storage in one remote location, similar to how you might use other file hosting services like Dropbox or Google Drive. Before trying to use NextCloud, you should check that you can access the service by logging in here. If you can access the NextCloud services, you can browse the various file storage services available to you:

- **Linux Home**: These are the files from your SCS Linux Account
- **Windows Home**: These are the files from your SCS Windows Account and your lab Z:\ drive.
- **NextCloud**: In addition to the other storage accounts provided to you by the SCS, you can also upload up to 20GB of files directly to NextCloud.

With NextCloud, you can upload your files from any machine with an internet connection and download them onto any other machine with an internet connection. For example, you can move project files off of your virtual machine, onto the NextCloud storage, and then download them on your personal laptop. Alternatively, you can upload files from your personal PC onto the NextCloud storage, place them into the Windows home folder, and access those files from either the lab Z:\ drive or download them on a virtual machine like VirtualBox or OpenStack.

Uploading Files to NextCloud from a Lab Computer

If you would like to upload files from a lab computer, the easiest way to do this is to place the files you would like to transfer into your Z:\ drive. These files will be automatically backup into your NextCloud storage under the Windows home folder. After that, you can move them into the main NextCloud storage or choose to keep them in your Z:\drive.

Uploading Files to NextCloud from a VM or Other PC

If you would like to upload files from either a VM or any other computer, you can log in to the NextCloud service using any of the available interfaces, such as the web interface. Press the "+" icon in the top left of the file browser and select Upload File. From here, you can choose to keep it in the main NextCloud storage, move it into your Windows Account storage (the Windows home folder), or into your Linux Account storage (the Linux Home folder).

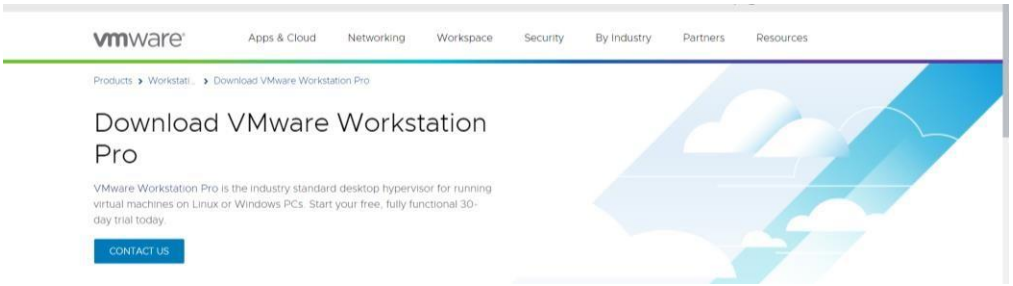
Downloading NextCloud Files to a VM or Other PC

Once your files are uploaded, you will be able to download those files onto any machine, which can connect to NextCloud. First, log in to your preferred NextCloud interface (e.g., the web interface). Navigate to the folder which contains the files you would like to download. Once you are in the target folder, click the checkbox next to each file you would like to download. Above the file listing, you should notice the context bar changing to tell you how many files you have selected and a button labelled Actions. Click Actions > Download.

If you selected a single file, it will prompt you to confirm the download. If you have chosen more than one file, NextCloud will place all of the selected files into a zip archive. Before you can use the files, you will need to extract them from the archive. Once you have downloaded your file or extracted your archive, you are ready to use your files on your machine.

Step 1: To download and install the VMware product visit the official website of VMware.

https://www.vmware.com/in/products/workstation-pro/workstationproevaluation.html



Step 2: Click on Download VM WorkStation for Windows. The installation file gets downloaded in the specified location and is now ready for installation.

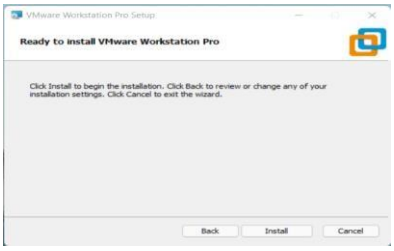
Step 3: Click on the download file to install the VMWare Workstation 16 Pro. Popup will appear

Step 4: Once the initialization gets completed, Click Next

Step 5: Accept the terms and click on Next. In the next screen, it will ask for some additional features, it is not mandatory to check this box. Click on Next.

Step 6: On the next screen, some checkboxes are populated, Check them as per your requirement. Click on Next.

Step 7: At this step, VMware Workstation is ready to install. Click on Install.

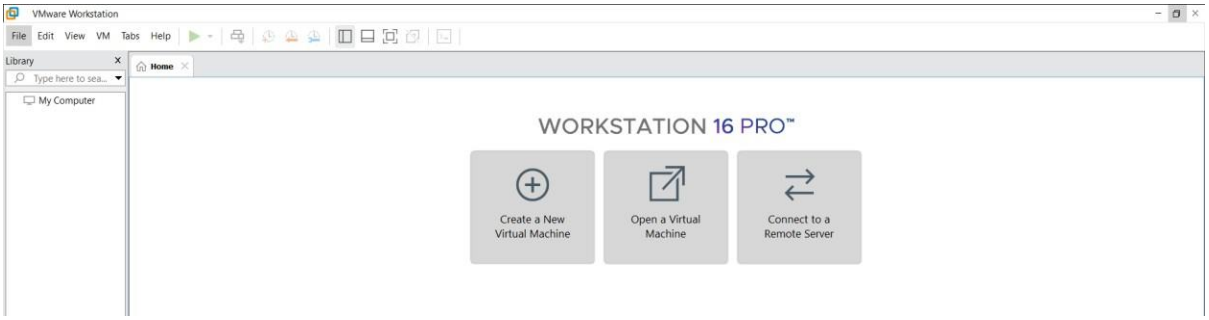


Step 8: Once the installation gets completed you will see the following dialogue box. Click on Finish. If you have purchased the product and have a license key, then you can click on License to enter the key.

Step 9: Upon Finish, the window will close, and You can see VMware Workstation installed icon on your Desktop. Double Click on the Icon to open the application.

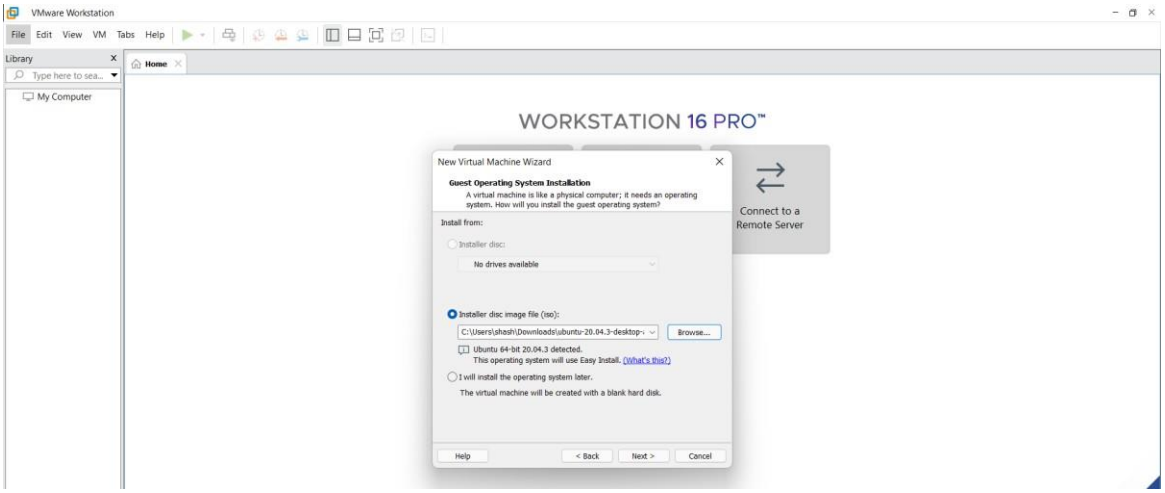
Step 10: For the first time opening, if you have not entered the License key in the previous steps, then it will ask for a license key.

Finally, this will open a window of VMware Workstation Pro.



Step 11: Click on File New Virtual Machine. A New Virtual Machine Wizard will appear. Click on Typical.

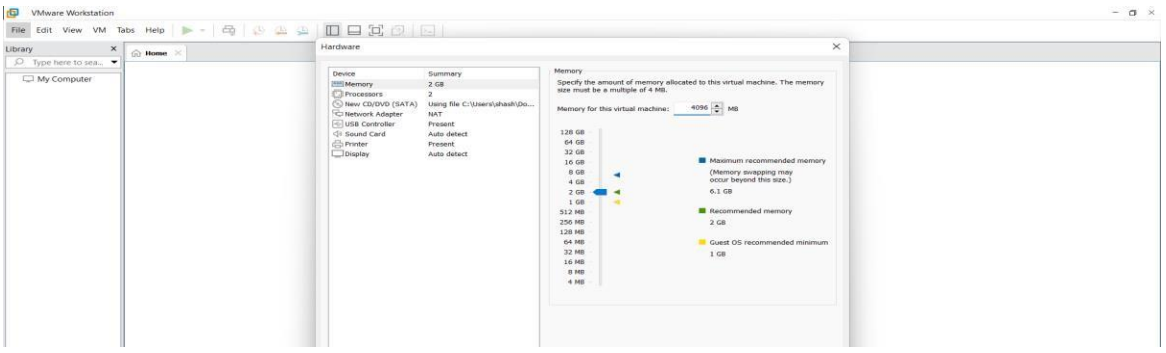
Step 12: Select the ISO File and click on Next



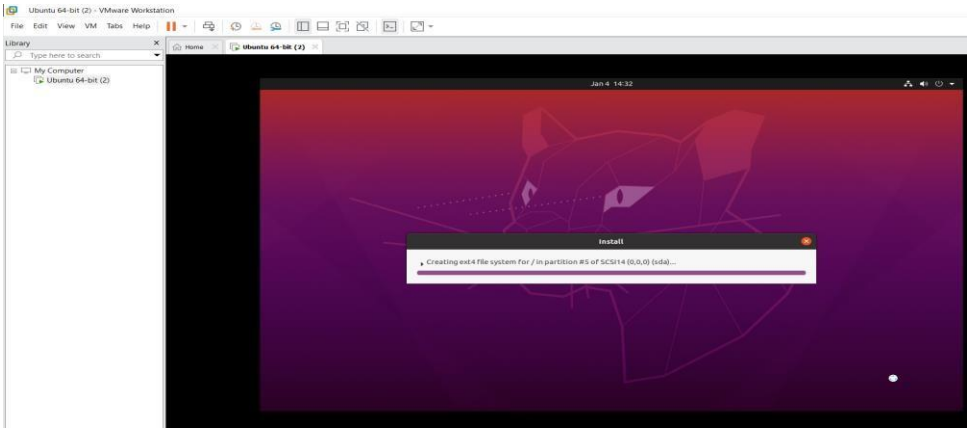
Step 13: Fill in the required details such as username, password and click on next.

Step 14: Name the virtual machines.

Step 15: Allocate the memory and select Split virtual disk into multiple files and click on next. Click on Customize Hardware. Set the memory size to 4GB



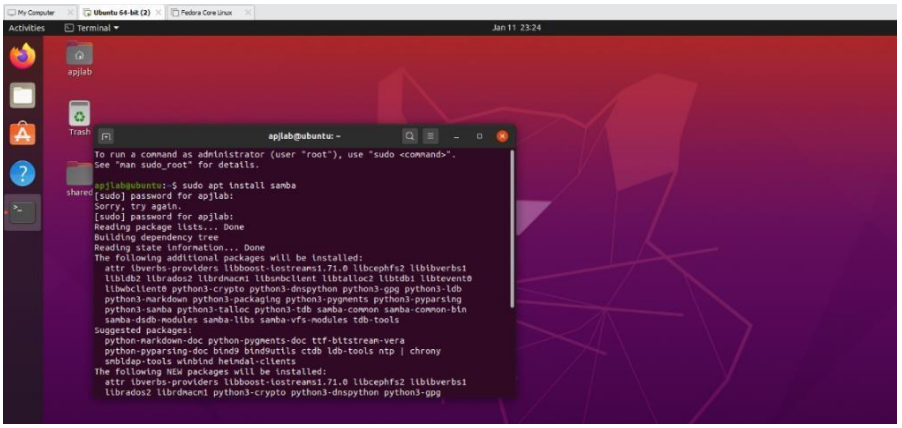
Step 16 : Click on Finish. You can see that Ubuntu gets installed in the VM Ware Workstation.

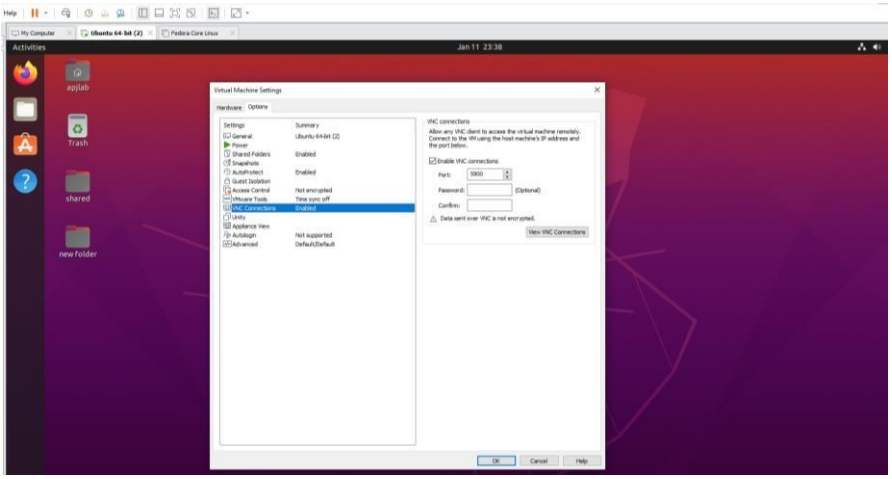


Step 17: Open the terminal and type the following command:

sudo apt install samba

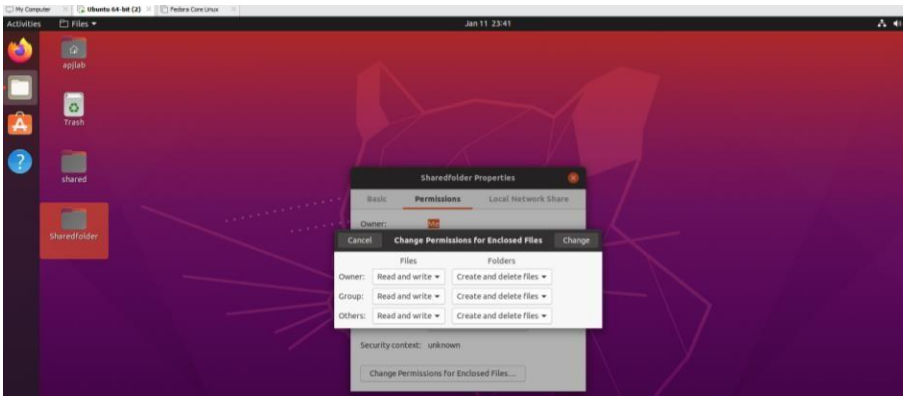
Wait for the installation to complete and close the terminal.





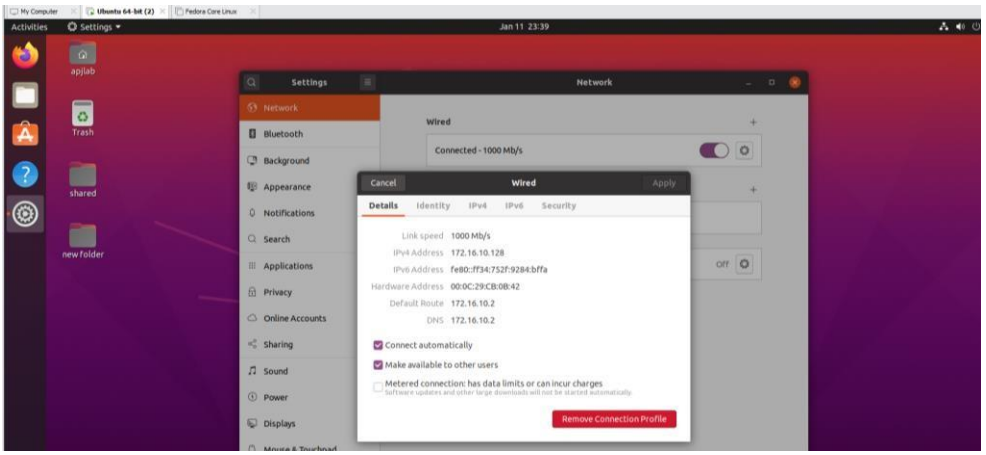
Step 18: Click on VM on top and click on Settings. A window appears. Click on Options and Enable Shared Folders, Auto Protect, VNC Connections and click on OK

Step 19: Click on permissions and set Read and Write Permissions for Owner, Group and Others.

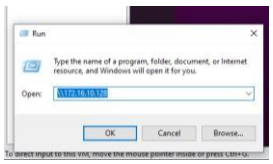


Step 20: Close all the windows and click on Settings

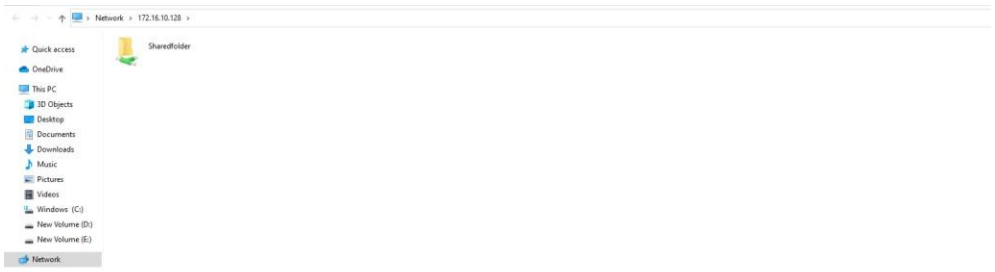
Step 21: Click on Wired Settings and window appears as shown in the below picture. Copy the IPv4 address.



Step 22: Open RUN on Windows and type the IP Address with the "\\". For example, if the IPv4 Address copied is 172.16.10.128 in the Run window enter \\172.16.10.128



Step 23: A window appears showing the folder that was shared. The user can now access the folder.



Dr. AMBEDKAR INSTITUTE OF TECHNOLOGY

Near Jnana Bharathi Campus, Bengaluru-560 056.
(An Autonomous Institution, Aided by Government of Karnataka)

MINI PROJECT REPORT
ON

"PETSHOP"

BACHELOR OF ENGINEERING
IN

COMPUTER SCIENCE AND ENGINEERING

SUBMITTED
BY

SHIVANAND
KERUR
1DA20CS135

UNDER THE GUIDANCE OF

Mr. Praveena M V
Asst.Prof, Dept. of CSE
Dr.AIT

Department of Computer Science & Engineering
2023-24

Dr. AMBEDKAR INSTITUTE OF TECHNOLOGY

Near Jnana Bharathi Campus, Bengaluru-560 056.
(An Autonomous Institution, Aided by Government of Karnataka)

Aided By Govt. of Karnataka

CERTIFICATE

This is to certify that the project entitled **"PETSHOP"** submitted in the partial fulfillment of the requirement of the 7th semester Cloud Computing laboratory curriculum during the year 2023-24 is a result of bonafide work carried out by-

SHIVANAND
KERUR
1DA20CS135

Signature of the guide

_____

Mr. Praveena M V
Asst.Prof, Dept. of CSE
Dr.AIT

1. Internal Examiner _____

2. External Examiner _____

Dr. Siddaraju,
Head of Department
Department of CSE, Dr.AIT

ACKNOWLEDGEMENT

The sense of contention and elation that accompanies the success of this seminar and the report could be incomplete without mentioning the names of people who have helped me in accomplishing them, people whose constant guidance, support and encouragement resulted in the realization.

I consider myself privileged to express our gratitude and respect towards all those who guided me through the project, **"PETSHOP".**

I take this opportunity to thank, **Dr. M Meenakshi, Principal, Dr. Ambedkar Institute of Technology,** Bengaluru for his support and encouragement.

I am grateful to **Dr. Siddaraju, Head of Department, CSE, Dr. Ambedkar Institute of Technology**, Bengaluru for providing encouragement and support.

I consider ourselves privileged to express our gratitude and respect towards our guide **Mr.Praveena.M.V** for constant guidance and support for the completion of the project.

Lastly, I thank all the members of the staff both teaching and non-teaching, friends and last but not the least our parents and family, for helping me directly or indirectly in the completion of the project.

SHIVANAND KERUR

ABSTRACT

Online Pet store management refers to the overall process of organizing, coordinating, and controlling the operations of a pet store. It involves managing inventory, sales, and customer information, as well as ensuring the availability of high-quality products and services for pets and their owners.

Effective pet store management requires a deep understanding of the pet industry, including the latest trends and developments, as well as customer needs and preferences. Key tasks involved in pet store management include ordering and receiving products, managing inventory levels, pricing and promotion of products, and providing excellent customer service.

In addition, it is important for pet store managers to keep up with industry regulations and guidelines, maintain a clean and safe store environment, and maintain positive relationships with suppliers and vendors. A well-managed pet store can provide a satisfying experience for pet owners while generating revenue and profits for the business.

CONTENTS

INTRODUCTION

A Online Pet Management System is a large complex structure involving many component entities like users, pet products and different services required for pets. Design and develop a user friendly efficient computerized software titled “Pet Store Management” for a better management of the pets. To provide user a comfortable web site experience with good UI.

To optimize and improve workflow automation. Secured data storage with high level of authentication and user confidentiality.To Provide better database and GUI based interface and interaction to the customer.

A System Requirements Specification (SRS) (also known as Software Requirement Specification is a document or set of documentation that describes the features and behavior of a system or software application. It includes a variety of elements (see below) that attempts to define the intended functionality required by the customer to satisfy their different users.

In addition to specifying how the system should behave, the specification also defines at a high-level, the main business processes that will be supported, what simplifying assumptions have been made and what key performance parameters will need to be met by the system.

The hardware and software requirements of a computer system are required to install and use the software efficiency in the SRS. The minimum requirements need to be met for the program to run efficiently all the times on the system are as follows:

.

CHAPTER 2

LITERATURE REVIEW

2.1 JAVASCRIPT

JavaScript stands as a cornerstone of modern web development, empowering websites and applications with dynamic, interactive functionalities. Initially conceived as a scripting language to add interactivity to static web pages, JavaScript has evolved into a versatile programming language that runs not only in web browsers but also on servers, enabling developers to create responsive, feature-rich experiences across diverse platforms.

One of JavaScript's key strengths lies in its versatility and ubiquity. As a client-side scripting language, it empowers developers to manipulate web page elements, handle user interactions, and dynamically update content without requiring a page refresh. Beyond the browser, JavaScript has expanded its reach with the rise of server-side development through platforms like Node.js, allowing developers to build robust and scalable backend systems.

JavaScript’s flexibility and ease of use have made it a foundational tool for web development. Its ecosystem boasts a rich collection of libraries, frameworks, and tools such as React, Angular, and Vue.js, enabling developers to expedite development processes and create highly responsive and interactive web applications. Furthermore, its compatibility with HTML and CSS facilitates seamless integration within web projects, cementing its position as a linchpin in the web development landscape.

As the digital realm continues to evolve, JavaScript remains a driving force behind the dynamic and immersive experiences users encounter daily, contributing to the continuous innovation and evolution of web technologies.

Features of Javascript are as follows:

- Versatile Scripting Language
- Client-Side Interaction
- Cross-Browser Compatibility
- Asynchronous Programming
- Server-Side Development
- Rich Ecosystem
- Event-Driven and Functional Paradigm
- Dynamic Typing
- Security
- Continuous Evolution

Advantages of Javascript are:

- **Versatility:** JavaScript's versatility allows it to function across various platforms, enabling both client-side and server-side scripting.

- **Enhanced User Experience:** It facilitates dynamic content updates without page reloads, leading to more interactive and responsive user interfaces.

- **Wide Browser Support:** Supported by all major browsers, ensuring consistent functionality and user experience across different platforms.
- **Extensive Ecosystem:** JavaScript boasts a vast ecosystem of libraries, frameworks, and tools that aid in rapid development and provide solutions for various functionalities.
- **Asynchronous Capabilities:** Utilizing asynchronous programming, JavaScript efficiently handles operations that might take time to complete without blocking other processes.
- **Easy to Learn:** With a syntax similar to other programming languages, JavaScript is relatively easy for beginners to pick up and start coding.
- **Community Support*:*** A large and active community constantly contributes to resources, forums, and open-source projects, fostering learning and problem-solving.
- **Server-Side Development:** Platforms like Node.js enable JavaScript to be used for server-side scripting, making it possible to build scalable and efficient backend systems.
- **Enhanced Interactivity:** Offers functionalities like event-driven programming, allowing the creation of responsive and interactive web applications.
- **Continuous Updates:** Regular updates and enhancements through ECMAScript standards ensure JavaScript remains relevant and adaptable to modern web development practices.

Disadvantages of Javascript are as follows:

- **Browser Compatibility:** Despite improvements, discrepancies in JavaScript interpretation among browsers can lead to compatibility issues, requiring additional code for cross-browser support.
- **Security Vulnerabilities:** As it executes on the client-side, JavaScript is susceptible to security threats like cross-site scripting (XSS) if not properly sanitized or validated, allowing attackers to manipulate web pages.
- **Single-threaded Execution:** JavaScript operates on a single thread, potentially causing performance bottlenecks in complex applications that require heavy computational tasks, as they might slow down the user interface.
- **Lack of Static Typing:** Being dynamically typed, JavaScript doesn't enforce strict typing, which can lead to errors that might be caught at runtime instead of during development.

2.2 RENDER

A cloud-based rendering platform, often referred to as a Render Cloud, represents a cutting-edge solution for industries heavily reliant on visual rendering processes. At its core, this platform harnesses the scalability and computational power of cloud infrastructure to revolutionize the rendering workflow. It operates as a remote rendering service accessible through an internet connection, eliminating the need for extensive local hardware and offering users a dynamic, scalable environment for rendering tasks..

One of the key advantages of a Render Cloud platform lies in its scalability. By leveraging cloud resources, users can tap into a virtually limitless pool of computing power. This scalability is particularly advantageous for rendering tasks that demand significant computational resources, enabling users to expedite project timelines and handle complex, high-resolution rendering jobs with remarkable efficiency. Such scalability empowers users to scale up or down based on project demands, optimizing resource utilization and minimizing costs.

Moreover, Render Cloud platforms often employ distributed rendering methodologies. This approach involves breaking down rendering tasks into smaller segments, distributing these segments across multiple cloud servers or nodes, and processing them simultaneously. The distributed rendering capability significantly accelerates the overall rendering process, harnessing parallel computing to complete tasks in a fraction of the time it would take using traditional local processing methods. This speed and efficiency enable users to achieve quicker turnaround times and meet tight project deadlines.

Interoperability is another crucial aspect of a Render Cloud platform. It's designed to seamlessly integrate with various rendering engines, offering compatibility with popular software tools used across industries. This flexibility allows users to choose the rendering engine that best suits their project requirements, ensuring compatibility and streamlining the workflow. Whether utilizing V-Ray, Arnold, Blender Cycles, or other rendering engines, the platform offers adaptability to cater to diverse rendering needs.

In addition to technical capabilities, a Render Cloud platform often incorporates collaboration and management tools. These features facilitate seamless communication and interaction among team members, allowing for real-time collaboration, feedback, and version control. Project management functionalities further enhance workflow efficiency, enabling users to organize, share, and manage rendered outputs across distributed teams or stakeholders, irrespective of geographical locations.

2.3 VISUAL STUDIO

Visual Studio stands as a robust and comprehensive integrated development environment (IDE) meticulously crafted by Microsoft, tailored for software development across diverse platforms. This powerful toolset offers developers a cohesive and feature-rich environment, encompassing a suite of tools, services, and functionalities to streamline the entire development lifecycle. With its user-friendly interface and extensive compatibility, Visual Studio caters to a wide spectrum of programming languages, frameworks, and technologies, empowering developers to code, debug, test, and deploy applications efficiently. Whether working on web applications, mobile apps, cloud solutions, or desktop software, Visual Studio provides an all-encompassing ecosystem equipped with debugging tools, code editors, project management capabilities, and integrations, fostering a seamless and productive development experience for individuals and teams alike.

2.4 HTML

The Hypertext Markup Language, or HTML is the standard markup language for documents designedto be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets and scripting languages such as JavaScript.

2.5 CSS

Cascading Style Sheets is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

2.6 REACT.JS

ReactJS stands as a popular JavaScript library renowned for building user interfaces with a declarative and component-based approach. Its virtual DOM system enhances performance by selectively updating components, ensuring efficient rendering. React's reusable components, state management, and one-way data flow streamline development, enabling rapid creation of interactive and scalable web applications..

2.7 NODE.JS

Node.js is a powerful, open-source JavaScript runtime environment built on Chrome's V8 JavaScript engine. It enables server-side execution of JavaScript, allowing developers to create scalable and high-performance applications. Node.js operates on a non-blocking, event-driven architecture, optimizing throughput and scalability for web applications.

2.8 EXPRESS.JS

Express.js, often referred to simply as Express, is a minimal and flexible web application framework for Node.js. It provides a robust set of features to develop web applications and APIs. Express simplifies the process of building web applications by offering a streamlined, unopinionated framework that allows developers to structure their applications as needed.

2.9 MONGODB

MongoDB is a versatile, document-oriented NoSQL database renowned for its flexibility, scalability, and ease of use. It diverges from traditional relational databases by storing data in a flexible, JSON-like format called BSON (Binary JSON), allowing for dynamic and schema-less data models.

2.10 TAILWIND

Tailwind CSS is a utility-first CSS framework that provides a highly customizable set of low-level utility classes. It diverges from traditional CSS frameworks by focusing on building user interfaces using utility classes rather than predefined components. This approach offers flexibility and allows for rapid development by providing a comprehensive library of utility classes that cover styling, layout, positioning, and responsiveness

CHAPTER 3 REQUIREMENT SPECIFICATION

The hardware and software components of a computer system that are required to install and use software efficiently are specified in the SRS. The minimum system requirements need to be met for the programs to run at all times on the system.

3.1 HARDWARE REQUIREMENTS

The hardware requirements specify the necessary hardware which provides us the platform to implement ourprograms.

- 2.2 GHz processor (Pentium).
- 8 GB RAM (System Memory).
- 20 GB of hard-drive space.
- VGA capable of 1024 x 768 screen resolution.
- Necessary computer peripherals such as keyboard etc.

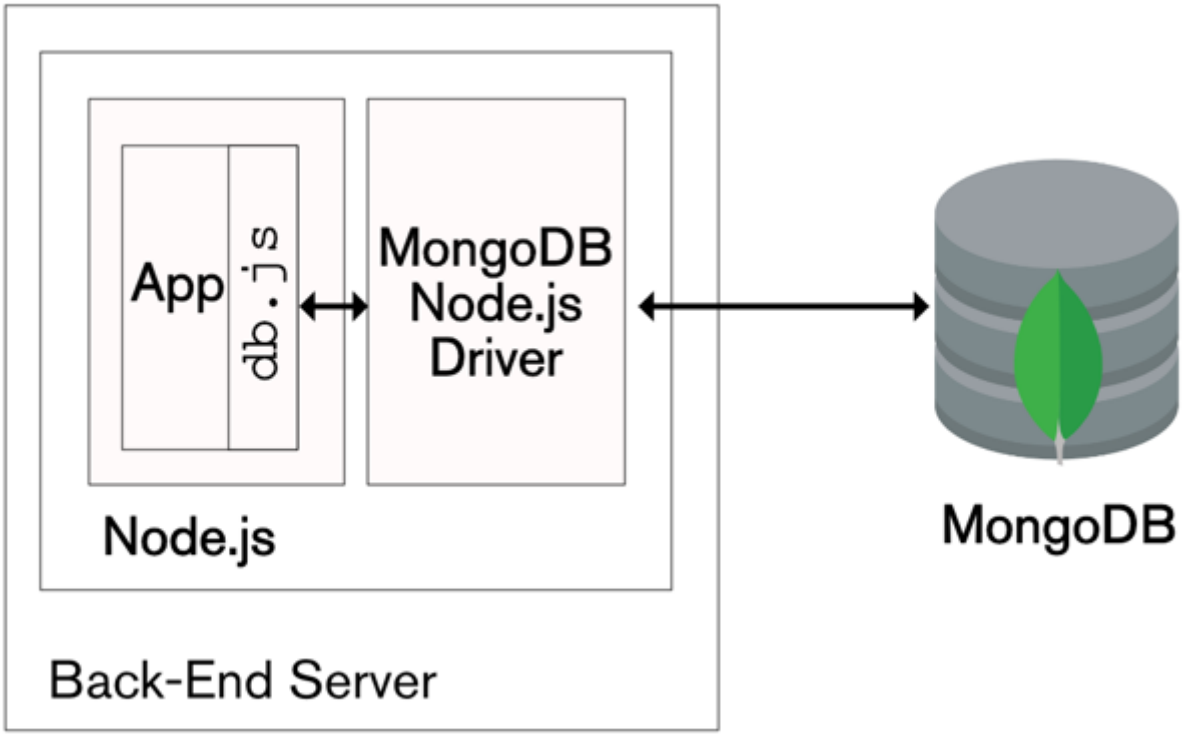
3.2 SOFTWARE REQUIREMENTS

The software requirement specifies the pre-installed software needed to run the code being implemented in this project.

- Windows Operating System
- Node
- Npm
- GitHub
- VSCode
- JavaScript

CHAPTER 4

DESIGN



To develop a secure and intuitive pet management system with robust authentication and streamlined CRUD functionality for efficient pet and item management.

The first step involves designing the project architecture, followed by the creation of the backend. The backend is primarily responsible for managing CRUD operations, including creating, deleting, and updating data, while ensuring secure authentication through the use of JWT tokens. Once the backend development is complete, thorough testing with Postman is conducted to verify its functionality. With a confirmed backend, the project then progresses to the frontend, where the implementation involves utilizing AJAX concepts and leveraging jQuery for DOM manipulation.

The home section features two distinct buttons for user authentication—login and signup. The signup section collects essential user information, including name, email (with email validation), password (minimum 8 characters), and a unique phone number. Moving on to the services segment, there are two pathways: one for regular users and another for administrators. For regular users, a dedicated dashboard allows them to explore and interact with pet-related items, such as pet toys, pet medicines, and pet food. Each item supports CRUD operations, providing the user with the ability to create, update, and delete entries. Additionally, the dashboard includes detailed views of pets, pet foods, and pet toys, along with filtering options for pet breeds and other attributes. Administrators have a separate dashboard for managing pet-related models, enabling them to perform CRUD operations on pet, pet toy, pet medicine, and pet food items. The filtering functionality extends to pet breeds and specific attributes for pets, as well as brand, taste, and price for pet food. To streamline the user experience, an order gateway displays all placed orders, while a payment gateway facilitates secure checkout for order

4.1 FRONTEND:

The roadmap for the pet management system begins with an emphasis on user interface design. Crafting an intuitive frontend involves creating components for user authentication, pet and item management, and implementing AJAX concepts for seamless data interaction. The frontend ensures a secure and engaging experience by incorporating robust authentication measures, including JWT tokens. The commitment to responsive design guarantees accessibility and consistent functionality across various devices, prioritizing an inclusive user experience.

4.2 BACKEND:

The backend of the pet management system is designed with a comprehensive architectural blueprint. It outlines RESTful APIs for CRUD operations, managing data related to pets, pet toys, pet medicines, and pet food. The API design includes routes and endpoints for user authentication, item handling, and user roles (regular users and administrators). MongoDB is chosen as the database, with a focus on crafting efficient schemas optimized for real-time updates. Real-time communication is facilitated through WebSocket connections, ensuring seamless interaction between clients and the server. Stringent security measures, including input validation and encryption protocols, fortify the application against potential vulnerabilities, ensuring a secure user experience.

4.3 DATABASE:

The database architecture for the pet management system delves into MongoDB schema design, emphasizing collections and document structures for users, pets, items, and orders. The choice between embedding and referencing data is carefully considered to meet specific application requisites. Indexing strategies are implemented to expedite queries for efficient data retrieval, with caching mechanisms optimizing performance for frequently accessed data. Scaling strategies, such as sharding and replication, are outlined to handle data and user volume surges. The data modeling framework is designed to evolve and accommodate future enhancements, ensuring scalability and adaptability as the system grows.

4.4 DEPLOYMENT:

The deployment process involves steps, processes, and activities to make the pet management system available to users. Software deployment includes release, installation, testing, and performance monitoring. Cloud deployment, leveraging models like SaaS, PaaS, and IaaS, is considered to enhance scalability and accessibility. Thorough testing, both backend and frontend, is conducted to ensure the system's functionality and security. Once confirmed, the system is deployed to the chosen hosting environment, providing users with a secure and intuitive platform for efficient pet and item management.

CHAPTER 5

DEVELOPMENT

index.js

```
import utils from "/js/util.js";

let cart;
//petfood starts from here

const loadPetFood = function () {
  fetch("/service/petfood", {
    method: "GET",
  })
    .then((res) => res.text())
    .then((res) => {
      utils.insertHtml("#main-content", res);
      utils.onclickEvent("#btn-filter-petfood", loadFoodItems);
      loadFoodFilter();
      loadFoodItems();
    });
};

const loadFoodItems = () => {
  let filter = getFoodFilter();
  loadData("/petfood", filter).then((data) => {
    loadSnippet("/fooditem")
      .then((res) => {
        var html;
        utils.insertHtml("#pro-container", "");
        for (const petfood of data.petfood) {
          html = utils.insertProperty(res, "id", petfood._id);
          html = utils.insertProperty(html, "name", petfood.foodname);
          html = utils.insertProperty(html, "brand", petfood.brand);
          html = utils.insertProperty(html, "price", petfood.price);
          if (petfood.images.length > 0)
            html = utils.insertProperty(html, "image", petfood.images[0].url);
          utils.appendHtml("#pro-container", html);
        }
      })
      .then((res) => {
        $(".pro > button").each((i, ele) => {
          ele.addEventListener("click", loadFoodDescription);
        });
      });
  });
};
```

```
const loadFoodDescription = (e) => {
  getServiceById("petfood", e.target.id).then((data) => {
    loadSnippet("/fooddescription")
      .then((res) => {
        let html = utils.insertProperty(res, "id", data.petfood._id);
        html = utils.insertProperty(html, "name", data.petfood.foodname);
        html = utils.insertProperty(html, "brand", data.petfood.brand);
        html = utils.insertProperty(html, "price", data.petfood.price);
        html = utils.insertProperty(html, "flavor", data.petfood.flavor);
        html = utils.insertProperty(
          html,
          "description",
          data.petfood.description
        );
        if (data.petfood.images.length > 0)
          html = utils.insertProperty(
            html,
            "image",
            data.petfood.images[0].url
          );
        utils.insertHtml("#main-content", html);
      })
      .then((res) => {
        $(`#desc-cart`).each((i, ele) => {
          if (getItemIndex(cart.petfood, data.petfood._id) != -1) {
            $(`#desc-cart`).show();
            $(`#${data.petfood._id}`).hide();
          }
          ele.addEventListener("click", (e) => {
            loadCart();
          });
        });
        $(`#${data.petfood._id}`).each((i, ele) => {
          ele.addEventListener("click", (e) => {
            $(`#desc-cart`).show();
            $(`#${data.petfood._id}`).hide();

            const index = getItemIndex(cart.petfood, e.target.id);
            if (index != -1) cart.petfood[index].quantity += 1;
            else
              cart.petfood.push({
                id: e.target.id,
                quantity: 1,
              });

            localStorage.setItem("cart", JSON.stringify(cart));
          });
        });
      });
  });
};

const loadFoodFilter = () => {
  fetch(utils.apiurl + "/petfood/getflavors")
```

```
    .then((res) => res.json())
    .then((res) => {
      let htmlData = "";
      for (let i = 0; i < res.flavors.length; i++)
        htmlData += `<input type="checkbox" name="flavor${i}"
value="${res.flavors[i]}">
      <label for="flavor${i}">${res.flavors[i]}</label><br />`;
      utils.insertHtml("#filter-flavors", htmlData);
    });

  fetch(utils.apiurl + "/petfood/getbrands")
    .then((res) => res.json())
    .then((res) => {
      let htmlData = "";
      for (let i = 0; i < res.brands.length; i++)
        htmlData += `<input type="checkbox" name="brand${i}" value="${res.brands[i]}">
      <label for="brand${i}">${res.brands[i]}</label><br />`;
      utils.insertHtml("#filter-brands", htmlData);
      getFoodFilter();
    });
};

const getFoodFilter = () => {
  let filter = {
    price: [0, 1000000],
    brand: [],
    flavor: [],
  };

  $("#filter-flavors > input").each((i, ele) => {
    if (ele.checked) filter.flavor.push(ele.value);
  });
  $("#filter-brands > input").each((i, ele) => {
    if (ele.checked) filter.brand.push(ele.value);
  });
  if ($("#lp").val()) filter.price[0] = parseInt($("#lp").val());
  if ($("#up").val()) filter.price[1] = parseInt($("#up").val());
  return filter;
};

const loadPets = function () {
  //  showLoading('#main-content', 'var(--primary-accent)', 5);
  fetch("/service/pets", {
    method: "GET",
  })
    .then((res) => res.text())
    .then((res) => {
      utils.insertHtml("#main-content", res);
      utils.onclickEvent("#btn-filter-pets", loadPetItems);
      loadPetFilter();
      loadPetItems();
      // loaddescription
    });
};
```

```
const loadPetItems = () => {
  let filter = getPetFilter();
  loadData("/pets", filter).then((data) => {
    loadSnippet("/petitem")
      .then((res) => {
        var html;
        utils.insertHtml("#pro-container", "");
        for (const pet of data.pets) {
          html = utils.insertProperty(res, "id", pet._id);
          html = utils.insertProperty(html, "name", pet.petClass);
          html = utils.insertProperty(html, "breed", pet.breed);
          html = utils.insertProperty(html, "price", pet.price);
          if (pet.images.length > 0)
            html = utils.insertProperty(html, "image", pet.images[0].url);
          utils.appendHtml("#pro-container", html);
        }
      })
      .then((res) => {
        $(".pro > button").each((i, ele) => {
          ele.addEventListener("click", loadPetDescription);
        });
      });
  });
};

const loadPetDescription = (e) => {
  getServiceById("pet", e.target.id).then((data) => {
    loadSnippet("/petdescription")
      .then((res) => {
        let html = utils.insertProperty(res, "id", data.pet._id);
        html = utils.insertProperty(html, "name", data.pet.petClass);
        html = utils.insertProperty(html, "brand", data.pet.breed);
        html = utils.insertProperty(html, "price", data.pet.price);
        html = utils.insertProperty(html, "description", data.pet.description);
        if (data.pet.images.length > 0)
          html = utils.insertProperty(html, "image", data.pet.images[0].url);
        utils.insertHtml("#main-content", html);
      })
      .then((res) => {
        $(`#desc-cart`).each((i, ele) => {
          if (getItemIndex(cart.pet, data.pet._id) != -1) {
            $(`#desc-cart`).show();
            $(`#${data.pet._id}`).hide();
          }
          ele.addEventListener("click", (e) => {
            loadCart();
          });
        });
        $(`#${data.pet._id}`).each((i, ele) => {
          ele.addEventListener("click", (e) => {
            $(`#desc-cart`).show();
            $(`#${data.pet._id}`).hide();
```

```
            const index = getItemIndex(cart.pet, e.target.id);
            if (index != -1) cart.pet[index].quantity += 1;
            else
              cart.pet.push({
                id: e.target.id,
                quantity: 1,
              });

            localStorage.setItem("cart", JSON.stringify(cart));
          });
        });
      });
  });
};

const loadPetFilter = () => {
  fetch(utils.apiurl + "/pets/getbreeds")
    .then((res) => res.json())
    .then((res) => {
      let htmlData = "";
      for (let i = 0; i < res.breeds.length; i++)
        htmlData += `<input type="checkbox" name="breeds${i}"
value="${res.breeds[i]}">
      <label for="breeds${i}">${res.breeds[i]}</label><br />`;
      utils.insertHtml("#filter-breeds", htmlData);
    });

  fetch(utils.apiurl + "/pets/getpetClass")
    .then((res) => res.json())
    .then((res) => {
      let htmlData = "";
      for (let i = 0; i < res.petClass.length; i++)
        htmlData += `<input type="checkbox" name="petClass${i}"
value="${res.petClass[i]}">
      <label for="petClass${i}">${res.petClass[i]}</label><br />`;
      utils.insertHtml("#filter-petClass", htmlData);
      getPetFilter();
    });
};

const getPetFilter = () => {
  let filter = {
    price: [0, 1000000],
    breed: [],
    petClass: [],
  };
  $("#filter-breeds > input").each((i, ele) => {
    if (ele.checked) filter.breed.push(ele.value);
  });
  $("#filter-petClass > input").each((i, ele) => {
    if (ele.checked) filter.petClass.push(ele.value);
  });
  if ($("#lp").val()) filter.price[0] = parseInt($("#lp").val());
  if ($("#up").val()) filter.price[1] = parseInt($("#up").val());
```

```
  return filter;
};

const loadPetToy = function () {
  fetch("/service/pettoy", {
    method: "GET",
  })
    .then((res) => res.text())
    .then((res) => {
      utils.insertHtml("#main-content", res);
      utils.onclickEvent("#btn-filter-pettoy", loadPetToyItems);
      loadPetToyFilter();
      loadPetToyItems();
      // loaddescription
    });
};

const loadPetToyItems = () => {
  let filter = getPetToyFilter();
  loadData("/pettoy", filter).then((data) => {
    loadSnippet("toyitem")
      .then((res) => {
        var html;
        utils.insertHtml("#pro-container", "");
        for (const pettoy of data.pettoy) {
          html = utils.insertProperty(res, "id", pettoy._id);
          html = utils.insertProperty(html, "name", pettoy.Toyname);
          html = utils.insertProperty(html, "brand", pettoy.brand);
          html = utils.insertProperty(html, "price", pettoy.price);
          if (pettoy.images.length > 0)
            html = utils.insertProperty(html, "image", pettoy.images[0].url);
          utils.appendHtml("#pro-container", html);
        }
      })
      .then((res) => {
        $(".pro > button").each((i, ele) => {
          ele.addEventListener("click", loadPetToyDescription);
        });
      });
  });
};

const loadPetToyDescription = (e) => {
  getServiceById("petToy", e.target.id).then((data) => {
    loadSnippet("/toydescription")
      .then((res) => {
        let html = utils.insertProperty(res, "id", data.pettoy._id);
        html = utils.insertProperty(html, "name", data.pettoy.Toyname);
        html = utils.insertProperty(html, "brand", data.pettoy.brand);
        html = utils.insertProperty(html, "price", data.pettoy.price);
        html = utils.insertProperty(
          html,
          "description",
          data.pettoy.description
```

```
        );
        if (data.pettoy.images.length > 0)
          html = utils.insertProperty(html, "image", data.pettoy.images[0].url);
        utils.insertHtml("#main-content", html);
      })
      .then((res) => {
        $(`#desc-cart`).each((i, ele) => {
          if (getItemIndex(cart.pettoy, data.pettoy._id) != -1) {
            $(`#desc-cart`).show();
            $(`#${data.pettoy._id}`).hide();
          }
          ele.addEventListener("click", (e) => {
            loadCart();
          });
        });
        $(`#${data.pettoy._id}`).each((i, ele) => {
          ele.addEventListener("click", (e) => {
            $(`#desc-cart`).show();
            $(`#${data.pettoy._id}`).hide();

            const index = getItemIndex(cart.pettoy, e.target.id);
            if (index != -1) cart.pettoy[index].quantity += 1;
            else
              cart.pettoy.push({
                id: e.target.id,
                quantity: 1,
              });

            localStorage.setItem("cart", JSON.stringify(cart));
          });
        });
      });
  });
};

const loadPetToyFilter = () => {
  fetch(utils.apiurl + "/pettoy/getpetClass")
    .then((res) => res.json())
    .then((res) => {
      let htmlData = "";
      for (let i = 0; i < res.petClass.length; i++)
        htmlData += `<input type="checkbox" name="petClass${i}"
value="${res.petClass[i]}">
       <label for="petClass${i}">${res.petClass[i]}</label><br />`;
      utils.insertHtml("#filter-petClass", htmlData);
      getPetFilter();
    });
  fetch(utils.apiurl + "/pettoy/getbrands")
    .then((res) => res.json())
    .then((res) => {
      let htmlData = "";
      for (let i = 0; i < res.brands.length; i++)
        htmlData += `<input type="checkbox" name="brand${i}" value="${res.brands[i]}">
       <label for="brand${i}">${res.brands[i]}</label><br />`;
```

```
      utils.insertHtml("#filter-brands", htmlData);
      getFoodFilter();
    });
};

const getPetToyFilter = () => {
  let filter = {
    price: [0, 1000000],
    brand: [],
    petClass: [],
  };
  $("#filter-petClass > input").each((i, ele) => {
    if (ele.checked) filter.petClass.push(ele.value);
  });
  $("#filter-brands > input").each((i, ele) => {
    if (ele.checked) filter.brand.push(ele.value);
  });
  if ($("#lp").val()) filter.price[0] = parseInt($("#lp").val());
  if ($("#up").val()) filter.price[1] = parseInt($("#up").val());
  return filter;
};

const loadPetmed = function () {
  //  showLoading('#main-content', 'var(--primary-accent)', 5);
  loadSnippet("/service/petmedicine").then((res) => {
    utils.insertHtml("#main-content", res);
    utils.onclickEvent("#btn-filter-petmed", loadMeditems);
    loadMeditems();
  });
};

const loadMeditems = function () {
  loadData("/petmedicine").then((data) => {
    loadSnippet("/meditem")
      .then((res) => {
        var html;
        for (const petmedicine of data.petmedicine) {
          html = utils.insertProperty(res, "id", petmedicine._id);
          html = utils.insertProperty(html, "name", petmedicine.medname);
          html = utils.insertProperty(html, "brand", petmedicine.brand);
          html = utils.insertProperty(html, "price", petmedicine.price);
          html = utils.insertProperty(html, "image", petmedicine.images[0].url);
          utils.appendHtml("#pro-container", html);
        }
      })
      .then((res) => {
        $(".pro > button").each((i, ele) => {
          ele.addEventListener("click", loadMedDescription);
        });
      });
  });
};

const loadMedDescription = (e) => {
```

```
  getServiceById("petmedicine", e.target.id).then((data) => {
    loadSnippet("/meddescription")
      .then((res) => {
        let html = utils.insertProperty(res, "id", data.petmedicine._id);
        html = utils.insertProperty(html, "name", data.petmedicine.medname);
        html = utils.insertProperty(html, "brand", data.petmedicine.brand);
        html = utils.insertProperty(html, "price", data.petmedicine.price);
        html = utils.insertProperty(html, "dosage", data.petmedicine.dosage);
        html = utils.insertProperty(
          html,
          "description",
          data.petmedicine.description
        );
        if (data.petmedicine.images.length > 0)
          html = utils.insertProperty(
            html,
            "image",
            data.petmedicine.images[0].url
          );
        utils.insertHtml("#main-content", html);
      })
      .then((res) => {
        $(`#desc-cart`).each((i, ele) => {
          if (getItemIndex(cart.petmedicine, data.petmedicine._id) != -1) {
            $(`#desc-cart`).show();
            $(`#${data.petmedicine._id}`).hide();
          }
          ele.addEventListener("click", (e) => {
            loadCart();
          });
        });
        $(`#${data.petmedicine._id}`).each((i, ele) => {
          ele.addEventListener("click", (e) => {
            $(`#desc-cart`).show();
            $(`#${data.petmedicine._id}`).hide();

            const index = getItemIndex(cart.petmedicine, e.target.id);
            if (index != -1) cart.petmedicine[index].quantity += 1;
            else
              cart.petmedicine.push({
                id: e.target.id,
                quantity: 1,
              });

            localStorage.setItem("cart", JSON.stringify(cart));
          });
        });
      });
  });
};

const loadPlaceOrder = () => {
  loadSnippet("/placeorder").then((res) => {
    let cartHtml = $("#cart-table").clone().html();
```

```
    utils.insertHtml("#main-content", res);
    utils.insertHtml("#cart-table", cartHtml);
    utils.onclickEvent("#btn-ordernow", async () => {
      await placeOrder();
      loadOrder();
    });
  });
};

const placeOrder = async () => {
  let order = {
    orderItems: [],
    itemsPrice: 0,
  };
  order.shippingInfo = {
    address: $("#address").val(),
    city: $("#city").val(),
    state: $("#state").val(),
    country: $("#country").val(),
    pinCode: parseInt($("#pincode").val()),
    phoneNo: parseInt($("#phone").val()),
  };
  order.paymentInfo = {
    id: s4() + "-" + s4(),
    status: $("input:checked", "#payment").val(),
  };
  for (let key of Object.keys(cart)) {
    for (let element of cart[key]) {
      let prod = {};
      prod[key] = element.id;
      order.orderItems.push({
        product: prod,
        quantity: element.quantity,
      });

      let res = await getServiceById(key, element.id);
      if (key == "pet") {
        order.itemsPrice += res.pet.price;
      } else if (key == "petfood") {
        order.itemsPrice += res.petfood.price;
      } else if (key == "petmedicine") {
        order.itemsPrice += res.petmedicine.price;
      } else if (key == "pettoy") {
        order.itemsPrice += res.pettoy.price;
      }
    }
  }
  order.taxPrice = (order.itemsPrice * 18) / 100;
  order.shippingPrice = 50 + (order.itemsPrice * 2) / 100;
  order.totalPrice =
    Math.round(
      (order.itemsPrice + order.taxPrice + order.shippingPrice) * 100
    ) / 100;
  await loadData("/order/new", order);
```

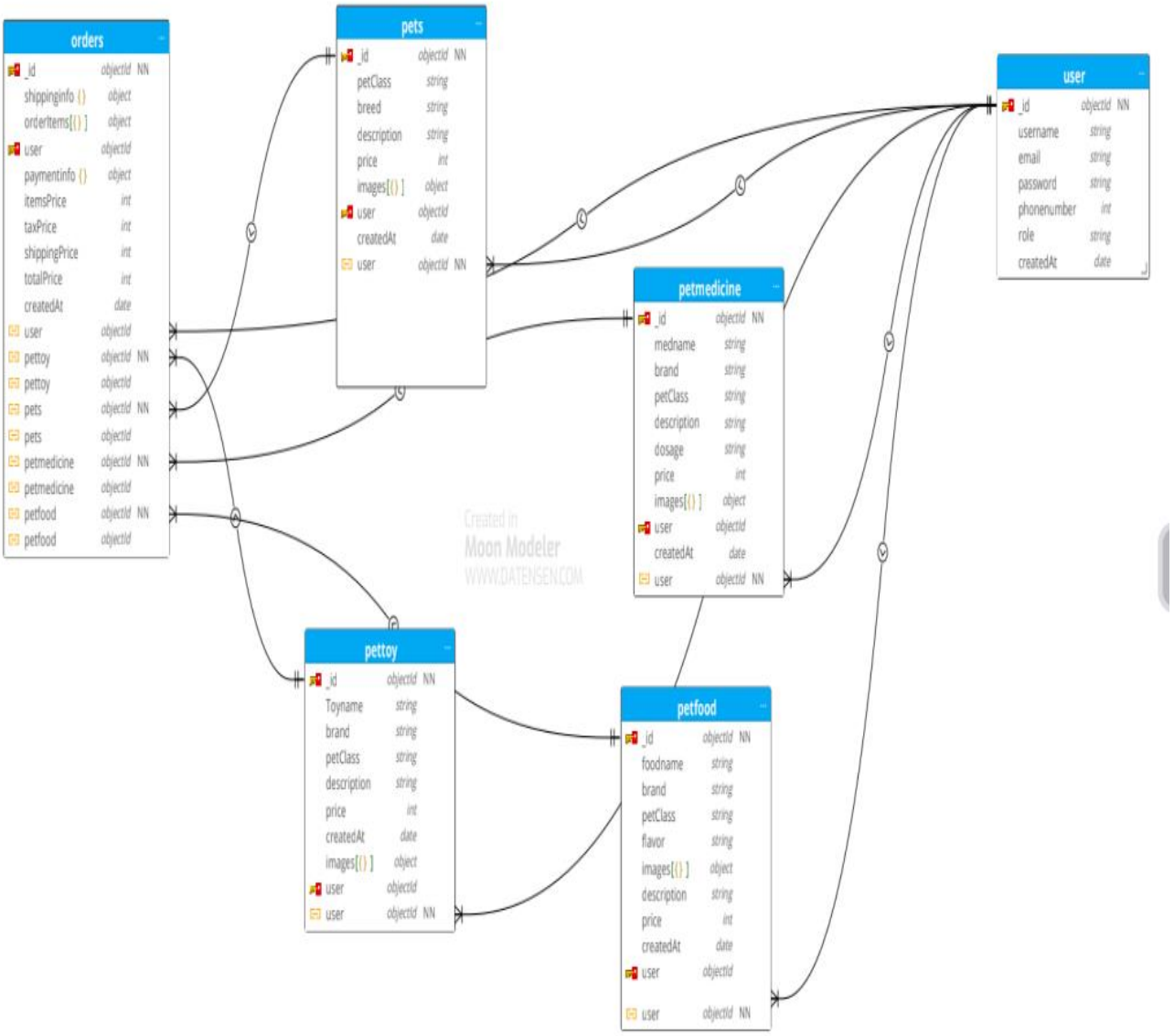
```
  localStorage.removeItem("cart");
  resetCart();
};

const loadOrder = async () => {
  let orderPage = await loadSnippet("/myorder");
  let snippet = `<tr>
        <td>{{#}}</td>
        <td>{{item}}</td>
        <td>{{address}}</td>
        <td>{{tax-price}}</td>
        <td>{{shipping-price}}</td>
        <td>{{amount}}</td>
        </tr>`,
    html,
    item,
    i = 1;

  let data = await fetch(utils.apiurl + "/orders/me", {
    credentials: "include",
  }).then((data) => data.json());
  utils.insertHtml("#main-content", orderPage);
  // utils.insertHtml('#order-desc', JSON.stringify(data));

  for (let [i, order] of data.orders.entries()) {
    html = utils.insertProperty(snippet, "#", i + 1);
    item = `<ol>`;
    for (let orderItem of order.orderItems) {
      let key = Object.keys(orderItem.product);
      let row = await getServiceById(key, orderItem.product[key]);
      var JSON_Obj = row;
      for (var w in JSON_Obj) {
      }
      if (key == "petfood")
        item += `<li>${JSON_Obj[key].foodname} ${JSON_Obj[key].brand}</li>`;
      else if (key == "petmedicine")
        item += `<li>${JSON_Obj[key].medname} ${JSON_Obj[key].brand}</li>`;
      else if (key == "pettoy")
        item += `<li>${JSON_Obj[key].Toyname} ${JSON_Obj[key].brand}</li>`;
      else if (key == "pet")
        item += `<li>${JSON_Obj[key].breed} ${JSON_Obj[key].petClass}</li>`;
    }
    item += `</ol>`;
    html = utils.insertProperty(html, "item", item);
    html = utils.insertProperty(
      html,
      "address",
      `${order.shippingInfo.address}, ${order.shippingInfo.city},
${order.shippingInfo.state} - ${order.shippingInfo.pinCode},
${order.shippingInfo.country}<br /> Phone: ${order.shippingInfo.phoneNo}`
    );
    html = utils.insertProperty(html, "tax-price", order.taxPrice);
    html = utils.insertProperty(html, "shipping-price", order.shippingPrice);
    html = utils.insertProperty(html, "amount", order.totalPrice);
```

```
    utils.appendHtml("#order-table", html);
  }
};
const getItemIndex = (arr, id) => {
  let index = -1;
  arr.every((element, i) => {
    if (element.id == id) {
      index = i;
      return false;
    }
    return true;
  });
  return index;
};

const loadCart = () => {
  loadSnippet("/cart").then((res) => {
    let snippet = `<tr>
                                <td >{{name}}</td>
                                <td>{{brand}}</td>
                                <td>{{quantity}}</td>
            </tr>`,
      html; // add/remove item in cart
    utils.insertHtml("#main-content", res);
    Object.keys(cart).forEach((key, index) => {
      cart[key].forEach((element, index) => {
        getServiceById(key, element.id).then((res) => {
          var JSON_Obj = res;
          for (var key in JSON_Obj) {
          }
          if (key == "petfood") {
            html = utils.insertProperty(
              snippet,
              "name",
              JSON_Obj[key].foodname
            );
            html = utils.insertProperty(html, "brand", JSON_Obj[key].brand);
          } else if (key == "petmedicine") {
            html = utils.insertProperty(snippet, "name", JSON_Obj[key].medname);
            html = utils.insertProperty(html, "brand", JSON_Obj[key].brand);
          } else if (key == "pettoy") {
            html = utils.insertProperty(snippet, "name", JSON_Obj[key].Toyname);
            html = utils.insertProperty(html, "brand", JSON_Obj[key].brand);
          } else if (key == "pet") {
            html = utils.insertProperty(snippet, "name", JSON_Obj[key].breed);
            html = utils.insertProperty(html, "brand", JSON_Obj[key].petClass);
          }
          html = utils.insertProperty(html, "quantity", element.quantity);
          utils.appendHtml("#cart-table", html);
        });
      });
    });
    utils.onclickEvent("#btn-placeorder", loadPlaceOrder);
```

```
  });
};
const getServiceById = (route, id) => {
  return fetch(`${utils.apiurl}/${route}/${id}`).then((data) => data.json());
};

const loadData = (route, data) => {
  return fetch(utils.apiurl + route, {
    method: "POST",
    credentials: "include",
    headers: {
      Accept: "application/json",
      "Content-Type": "application/json",
    },
    body: JSON.stringify(data),
  }).then((data) => data.json());
};

const loadSnippet = (route) => {
  return fetch(route, {
    method: "GET",
  }).then((res) => res.text());
};

const logout = () => {
  fetch(utils.apiurl + "/logout", {
    method: "GET",
    credentials: "include",
  }).then((res) => (window.location.href = "/"));
};
const s4 = () => {
  return Math.floor((1 + Math.random()) * 0x10000)
    .toString(16)
    .substring(1);
};

const resetCart = () => {
  cart = {
    pet: [],
    petfood: [],
    pettoy: [],
    petmedicine: [],
  };
  if (localStorage.getItem("cart"))
    cart = JSON.parse(localStorage.getItem("cart"));
};

$(() => {
  utils.onclickEvent("#s1", loadPetFood);
  utils.onclickEvent("#s2", loadPetmed);
  utils.onclickEvent("#s3", loadPetToy);
  utils.onclickEvent("#s4", loadPets);
  resetCart();
  utils.onclickEvent("#btn-dropdown", () => $("#dropdown").toggle());
```

```
  utils.onclickEvent("#cart", loadCart);
  utils.onclickEvent(
    "#dashboard",
    () => (window.location.href = "/dashboard.html")
  );
  utils.onclickEvent("#myorder", loadOrder);
  utils.onclickEvent("#profile", loaduserprofile);
  utils.onclickEvent("#logout", logout);
});


const loadupdateprofile = (e) => {
  loadSnippet("service/userprofile").then((res) => {
    utils.insertHtml("#main-content", res);
  });
  fetch(utils.apiurl + "/me", {
    method: "GET",
    credentials: "include",
    headers: {
      Accept: "application/json",
      "Content-Type": "application/json",
    },
  })
    .then(function (response) {
      const data = response.json();
      return data;
    })
    .then((data) => {
      $("#username").prop("readonly", false);
      $("#email").prop("readonly", false);
      $("#password").prop("readonly", false);
      $("#phonenumber").prop("readonly", false);
      $("#username").val(data.user.name),
        $("#email").val(data.user.email),
        $("#phonenumber").val(data.user.phonenumber);
        $(`#btn-upd`).hide();
      utils.onclickEvent("#btn-apply", updateprofile);
    });
};

const updateprofile = () => {
  const data = {
    name: $("#username").val(),
    email: $("#email").val(),
    phonenumber: parseInt($("#phonenumber").val()),
  };
  fetch(utils.apiurl + "/me/update", {
    method: "PUT",
    credentials: "include",
    headers: {
      Accept: "application/json",
      "Content-Type": "application/json",
    },
    body: JSON.stringify(data),
```

```
  })
    .then((res) => res.json())
    .then((res) => {
        loadSnippet("service/userprofile").then((res) => {
        utils.insertHtml("#main-content", res);
        loaduserprofile();
      });
    });
};

const loaduserprofile = async (e) => {
  fetch(utils.apiurl + "/me", {
    method: "GET",
    credentials: "include",
    headers: {
      Accept: "application/json",
      "Content-Type": "application/json",
    },
  })
    .then(function (response) {
      const data = response.json();
      return data;
    })
    .then((data) => {


      loadSnippet("service/userprofile").then((res) => {
        let html = utils.insertProperty(res, "id", data.user._id);
        html = utils.insertProperty(html, "name", data.user.name);
        html = utils.insertProperty(html, "email", data.user.email);
        html = utils.insertProperty(html, "password", data.user.password);
        html = utils.insertProperty(html, "phonenumber", data.user.phonenumber);
        utils.insertHtml("#main-content", html);
        utils.onclickEvent("#btn-upd", loadupdateprofile);
        $('#btn-apply').hide();
      });
    });
};
```

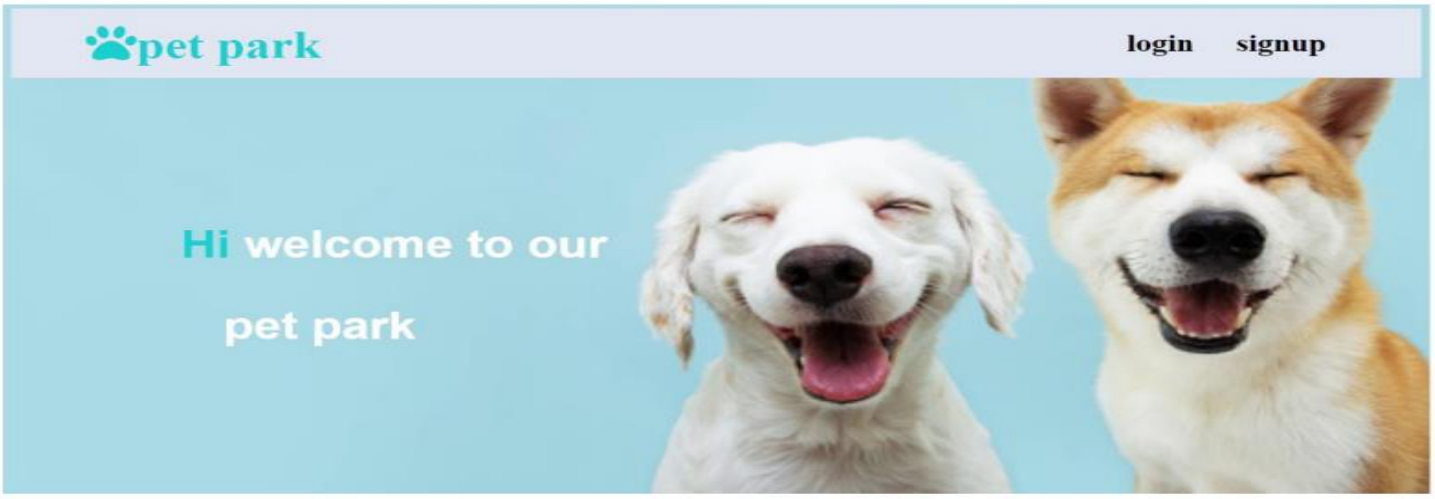


Backend strucuture

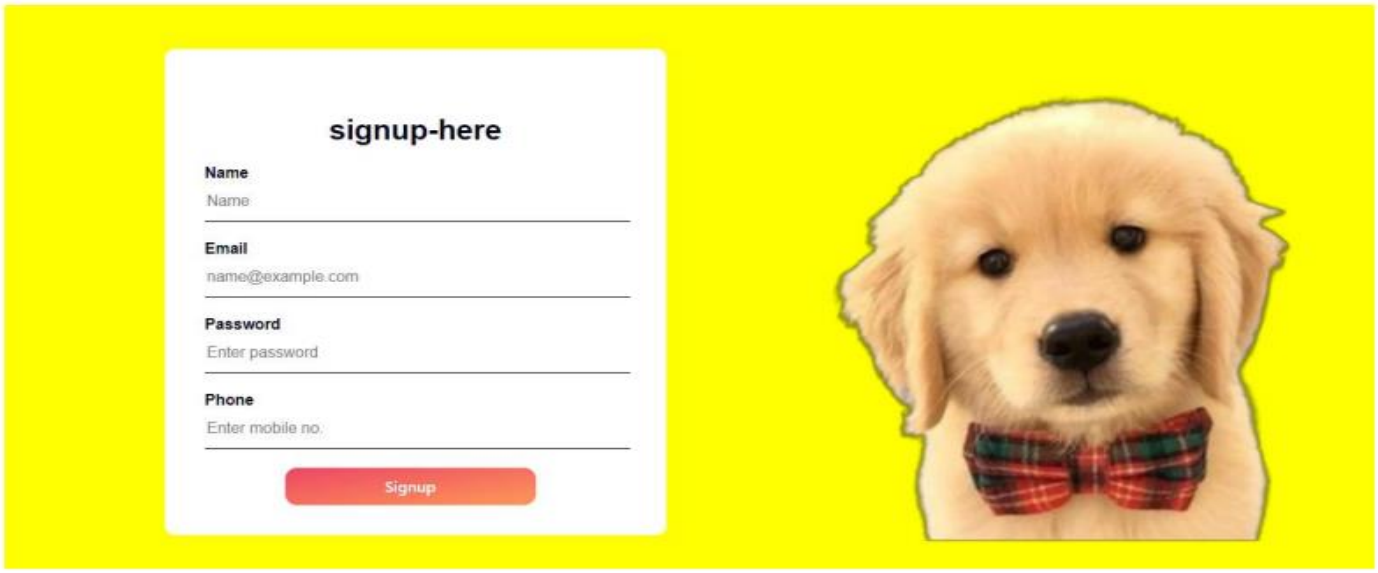
RESULTS

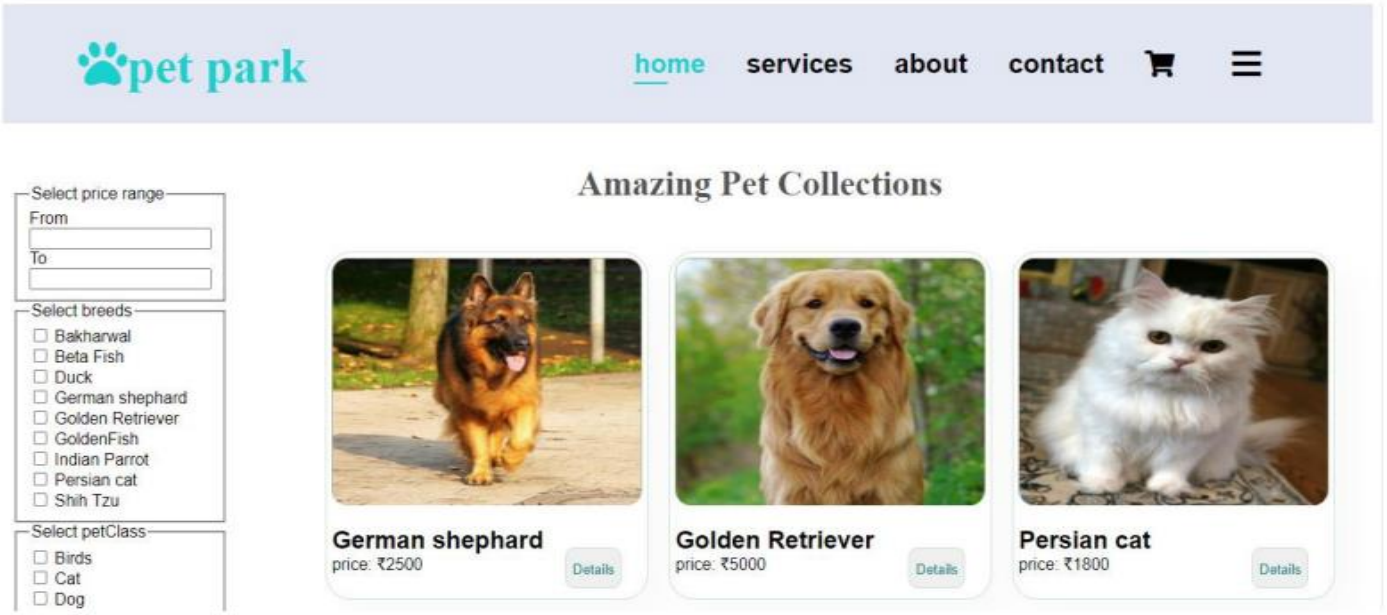
Evaluating the success of a petshop project involves various key areas. Firstly, understanding user engagement is crucial, focusing on metrics such as active user count, session duration, and interaction levels within the app. Additionally, monitoring user retention rates over time helps in assessing the app's ability to retain its user base. Ensuring top-notch performance, including responsiveness, loading times, and overall reliability, is paramount. Feature adoption metrics highlight which functionalities resonate most with users, guiding future updates and improvements.

FRONTEND SCREENSHOT

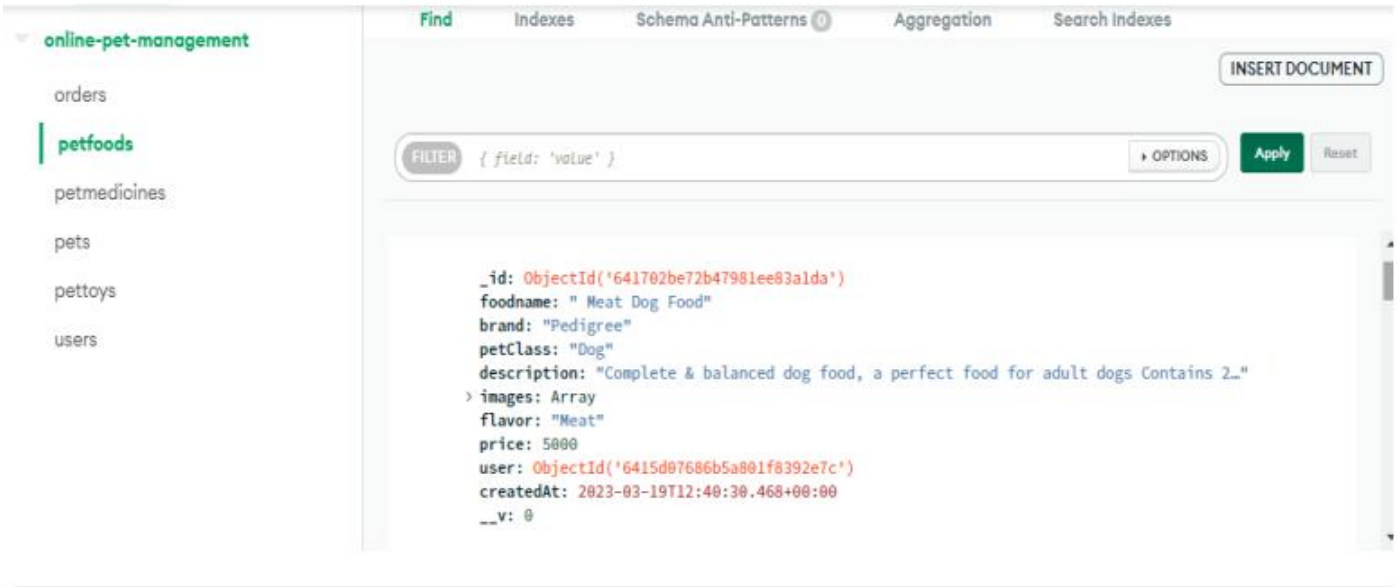


Login page





BACKEND SCREENHSOT



Users document

CHAPTER 7

DEPLOYMENT

Software deployment includes all of the steps, processes, and activities that are required to make a software system or update available to its intended users. Today, most IT organizations and software developers deploy software updates, patches and new applications with a combination of manual and automated processes. Some of the most common activities of software deployment include software release, installation, testing, deployment, and performance monitoring.

Software development teams have innovated heavily over the past two decades, creating new paradigms andworking methods for software delivery that are designed to meet the changing demands of consumers in an increasingly connected world. In particular, software developers have created workflows that enable faster and more frequent deployment of software updates to the production environment where they can be accessedby users.

7.1 CLOUD DEPLOYEMENT

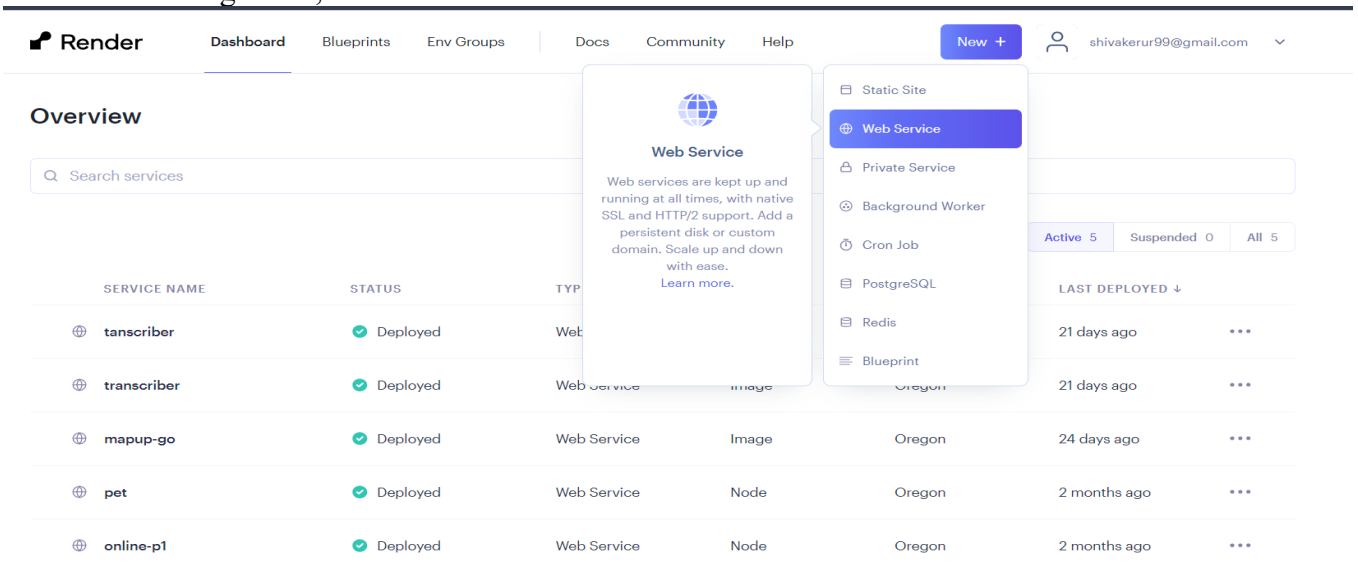
Cloud deployment is the process of deploying an application through one or more hosting models—softwareas a service (SaaS), platform as a service (PaaS) and/or infrastructure as a service (IaaS)—that leverage the cloud. This includes architecting, planning, implementing and operating workloads on cloud.

Advantages of Cloud Deployment:

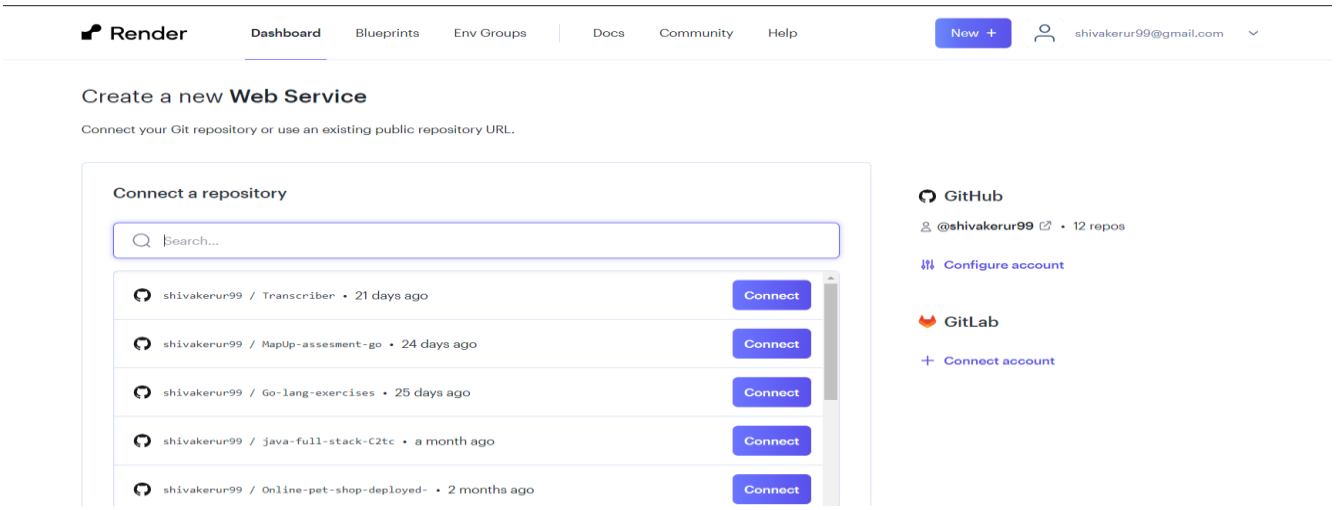
- Faster and simplified deployments. Automate builds that deploy code, databases and applicationreleases, including resource provisioning.
- Cost savings. Control costs using consumption-based pricing and eliminate capex-heavy on-premisesenvironments.
- Platform for growth. Leverage the global infrastructure provided by cloud service providers (CSPs) toseamlessly expand the business into other geographies.
- New digital business models. Exploit the continuous release of features and services by CSPs, incubatenew technologies and innovate digital business models.
- Business resiliency. Architect for the availability and fault-tolerance CSPs offer and ensure disasterrecovery and business continuity of applications to make the business resilient.
- Agility and scalability. Use autoscaling and scalability to meet peak demands of the business withoutprovisioning for excess capacity.
- Geographic reach. Access applications from any location, on any device, leveraging the connectivitybackbone of CSPs.
- Operational efficiency. Use the inherent automation enabled by cloud to increase operational efficiency and reduce human effort.
- A competitive edge. Leverage infrastructure as code and development, security and operations toreduce the time to market for new features and stay ahead of the competition.

7.1 Deploying on Render:

Step 1: Sign up and Create a New Service
Sign up for an account on Render.com if you haven't already.
Once signed in, click on "Create a New Service" on the Render dashboard.



Step 2: Connect Your GitHub Repository
Connect your GitHub repository to Render by clicking on "Add Git Repository."
Grant Render access to the repository containing your app's code.

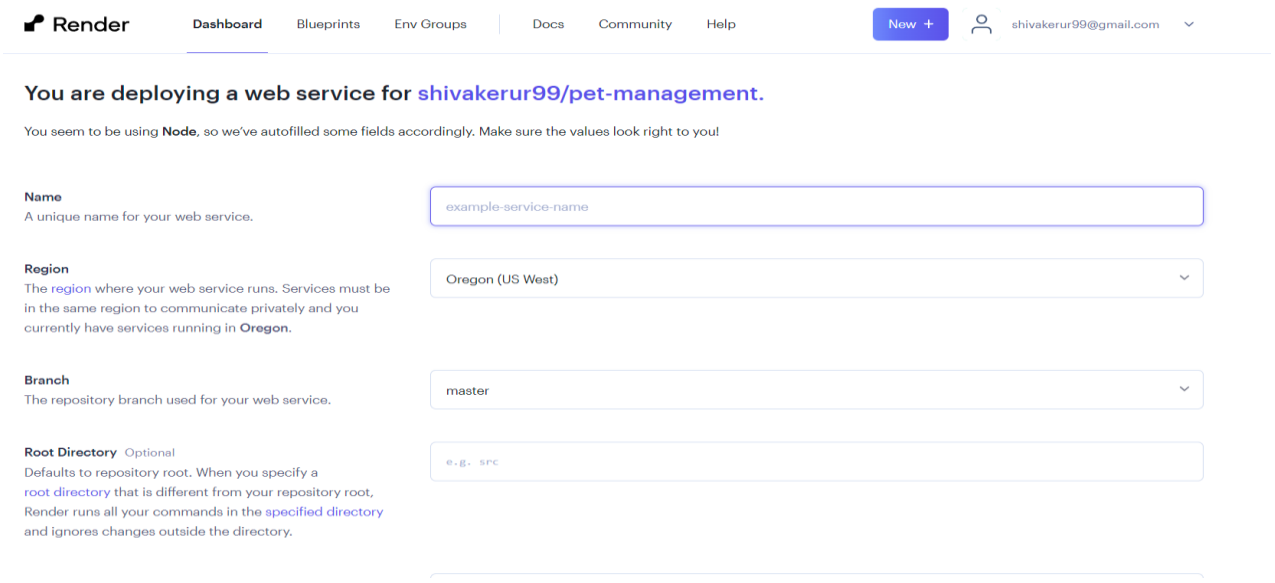


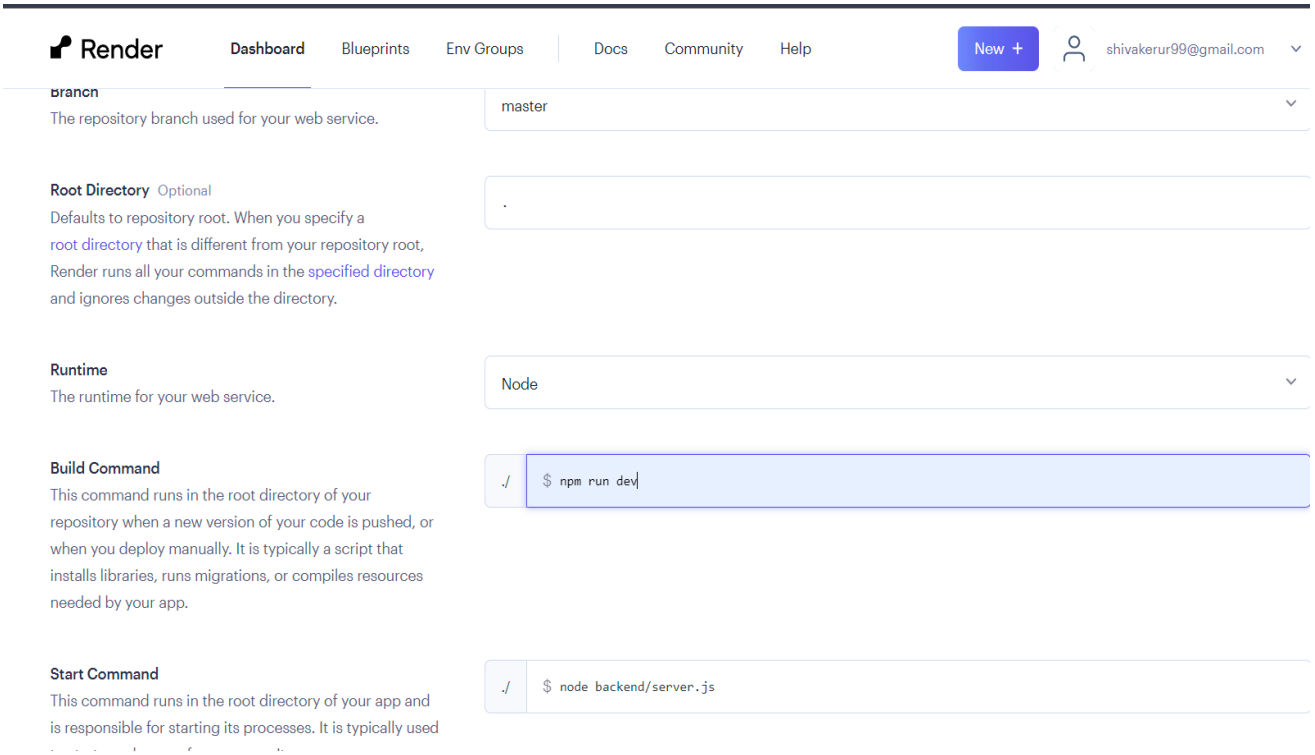
Step 3: Configure the Deployment Settings
Choose your connected Git repository from the dropdown.
In the "start Command" field, enter the command 'npm start'

Step 4: Add Environment Variables

Click on the “Environment Variables” tab.

Add any environment variables, such as API keys or database connection strings.

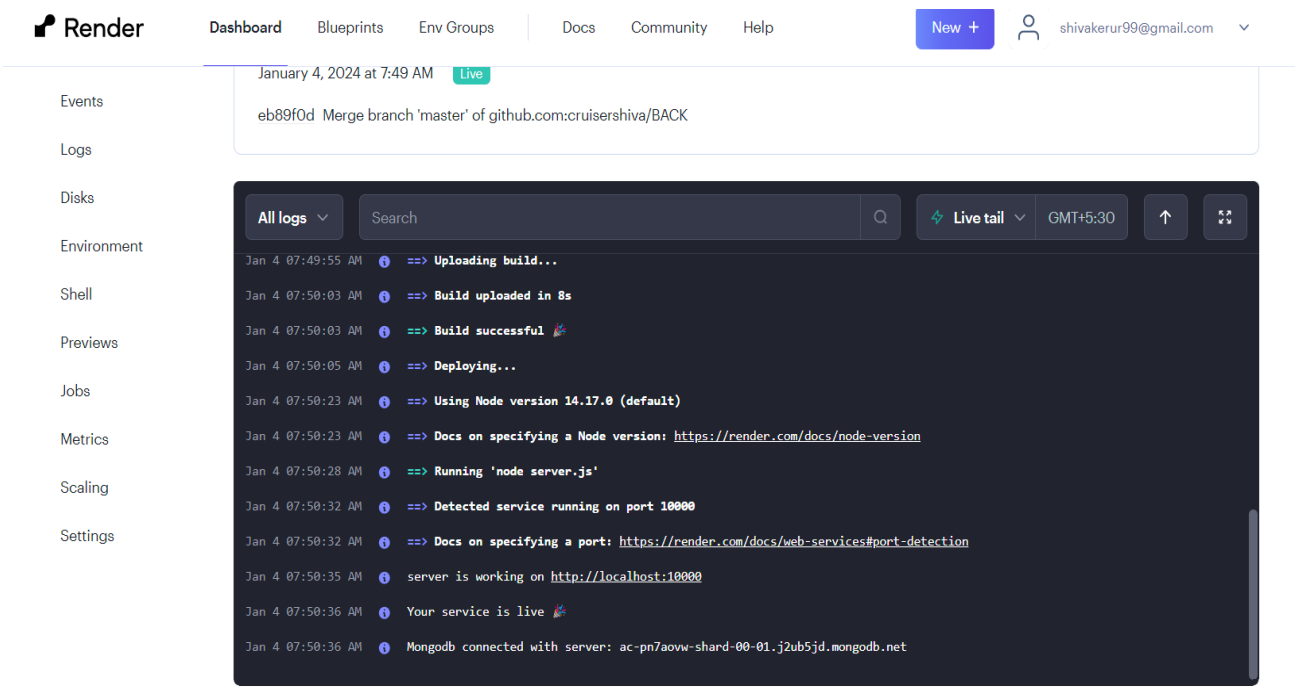




Step 5: Deploy Your App

Click on “Create Service” to deploy your MERN app on Render.

The deployment process will begin, and you’ll be able to monitor the progress and logs.



Step 6: Verify and Test

Once the deployment is complete, Render will provide you with a unique URL for app.

Open the URL in a browser to ensure everything is working as expected.

The app is published at URL: Complete Responsive Petshop (online-p1.onrender.com)

CONCLUSION

The success of the pet management system is intricately tied to a comprehensive evaluation across various dimensions. User engagement, retention, performance, security, scalability, and community growth collectively contribute to the project's overall viability. To ensure sustained success, developers must adopt a multifaceted approach, continually analyzing user behavior, prioritizing improvements, and staying competitive in a dynamic market.

Regularly seeking and incorporating user feedback is essential for refining the system and meeting evolving user needs. Upholding high standards of performance and security safeguards user data and maintains trust. The adaptability of the pet management system to changing requirements ensures its relevance and competitiveness over time.

In essence, a thriving pet management system is one that consistently evolves, prioritizes user experience, and remains responsive to the demands of its user base. This commitment to ongoing improvement and adaptation guarantees a sustainable and flourishing presence in the digital landscape, establishing the petshop as a trusted and user-friendly platform for efficient pet and item management

BIBLIOGRAPHY

Books:

Distributed and Cloud Computing: From Parallel Processing to the Internet of Things, Kai Hwang

URLs:

https:/www.geeksforgeeks.org/mern-stack/

https://www.mongodb.com/languages/mern-stack-tutorial

https://medium.com/@ansari028amaan/a-guide-to-deploy-your-mern-app-on-render-f6da3b1e9ccc

INDUSTRY CERTIFICATION IN SALESFORCE