

# Building a Comprehensive Laptop Rental Management System on Salesforce

## 1. Introduction to the Laptop Rental Management System

### 1.1. Project Overview and Objectives

The digital age has seen an exponential rise in the demand for flexible access to technology. For businesses, educational institutions, and individuals alike, the need for temporary access to hardware like laptops is increasingly common. This project focuses on the development and implementation of a robust **Laptop Rental Management System** built entirely on the Salesforce platform. The primary objective is to streamline and automate the entire lifecycle of a laptop rental, from customer inquiry and booking to billing and inventory tracking, all within a centralized and scalable cloud-based environment.

This system aims to achieve several key objectives:

- **Centralized Data Management:** Consolidate all customer, laptop inventory, booking, and billing information into a single, accessible platform. This eliminates data silos and provides a unified view of all rental operations.
- **Streamlined Booking Process:** Automate the booking workflow, including capturing customer details, selecting laptop types and rental durations, and dynamically calculating rental costs.
- **Efficient Inventory Tracking:** Maintain real-time visibility into the availability of laptops, ensuring accurate stock levels and preventing overbooking.
- **Automated Communication:** Implement automated email notifications to customers upon successful bookings, enhancing the customer experience.
- **Role-Based Access Control:** Define distinct user roles (e.g., Owner, Agent) with appropriate permissions, ensuring data security and operational integrity.
- **Actionable Reporting and Analytics:** Provide comprehensive reports and dashboards to give stakeholders insights into rental trends, revenue, and inventory status, facilitating informed decision-making.
- **Scalability and Flexibility:** Leverage the power of the Salesforce platform to ensure the system can easily scale with business growth and adapt to evolving rental requirements.

By achieving these objectives, the Laptop Rental Management System will significantly enhance operational efficiency, improve customer satisfaction, and provide critical business intelligence, ultimately contributing to the success and growth of the laptop rental service.

### 1.2. Why Salesforce for Laptop Rentals?

Salesforce stands out as an ideal platform for developing a Laptop Rental Management System due to its inherent strengths as a leading cloud-based Customer Relationship Management (CRM) and platform-as-a-service (PaaS) solution. Its benefits extend far beyond traditional CRM functionalities, making it a powerful choice for custom application development:

- **Cloud-Native and Accessible:** As a cloud-based platform, Salesforce requires no on-premise hardware or software installation. This ensures the system is accessible from anywhere, at any time, with an internet connection, promoting remote work capabilities and business continuity.
- **Scalability:** Salesforce is designed to scale effortlessly. Whether the rental business handles a dozen bookings a month or thousands, the platform can accommodate increasing data volumes and user loads without performance degradation. This eliminates concerns about outgrowing the system.
- **Customization and Flexibility:** The platform offers extensive customization capabilities without requiring deep coding knowledge for many common functionalities. Through clicks, not code, administrators can define custom objects, fields, relationships, validation rules, and automation rules, tailoring the system precisely to the unique needs of a laptop rental business. For more complex logic, Apex and Visualforce provide powerful programming options.
- **Robust Security:** Salesforce provides enterprise-grade security features, including robust data encryption, multi-factor authentication, granular access controls (profiles, roles), and regular security updates. This ensures sensitive customer and financial data related to rentals is protected.
- **Integrated Automation Tools:** Features like Salesforce Flow (Process Builder, Workflow Rules, Flow Builder) and Apex enable powerful automation of business processes. This project utilizes Flow for dynamic pricing and Apex for automated email notifications, significantly reducing manual effort and potential errors.
- **Comprehensive Reporting and Analytics:** Salesforce offers powerful built-in reporting and dashboard capabilities. Users can easily create custom reports, slice and dice data, and visualize key performance indicators (KPIs) through interactive dashboards, providing immediate insights into rental performance, inventory, and revenue.
- **Ecosystem and AppExchange:** The vast Salesforce ecosystem and AppExchange (an online marketplace for business applications) mean that if there's a need for additional functionality not built into the core system (e.g., advanced payment gateway integrations, specialized inventory tracking), there's likely a pre-built solution or a development partner available.

- **Reliability and Uptime:** Salesforce maintains a high level of system uptime and reliability, ensuring that the rental business operations are rarely interrupted due to platform issues.

In summary, choosing Salesforce for this Laptop Rental Management System not only provides a highly customizable and efficient solution for current needs but also lays a strong, scalable, and secure foundation for future growth and evolving business requirements.

### 1.3. System Architecture Overview (High-Level)

The Laptop Rental Management System on Salesforce is structured around a foundational data model, layered with user interface elements, robust security, and powerful automation. At a high level, the architecture can be conceptualized as follows:

- **Core Data Model:**
  - **Custom Objects:** The primary building blocks are custom objects like **Consumer**, **Total Laptops**, **Laptop Bookings**, and **Billing Process**. These represent the core entities involved in a rental transaction.
  - **Custom Fields:** Each object contains specific fields to capture relevant data (e.g., Consumer's phone number, Laptop Booking's core type, Billing Process payment mode).
  - **Relationships:** Master-Detail and Lookup relationships link these objects, establishing how they relate to each other (e.g., a Laptop Booking is related to a specific Consumer and a Total Laptop record). This forms a connected network of data.
- **User Interface (UI):**
  - **Custom Tabs:** Provide direct access points for users to each custom object from within the Salesforce application.
  - **Lightning App:** The "LAPTOP RENTALS" Lightning App acts as a container, providing a cohesive user experience by grouping all relevant tabs and functionalities for rental operations.
- **Security and Access:**
  - **Profiles:** Define the baseline permissions and access levels for different types of users (e.g., "Owner" and "Agent"), controlling what objects, fields, and functionalities they can see and interact with.
  - **Roles:** Establish a hierarchical structure (e.g., Agent reports to Owner), which can be used for data visibility and reporting relationships (e.g., higher roles can see data owned by lower roles).

- **Validation Rules:** Enforce data quality and business logic at the point of data entry (e.g., ensuring critical fields are populated).
- **Automation Layer:**
  - **Salesforce Flow:** A record-triggered flow on the "Laptop Booking" object automates complex pricing calculations based on laptop model, core type, and rental duration. This dynamically updates the rental Amount field.
  - **Apex Trigger:** A custom Apex trigger on the "Laptop Booking" object initiates actions (like sending emails) after a booking record is created or updated.
  - **Apex Handler Class:** An Apex class (LaptopBookingHandler) contains the business logic for the trigger, specifically responsible for sending automated email notifications to customers with their booking details.
- **Reporting and Analytics:**
  - **Reports:** Custom reports are built to query and display data across various objects (e.g., "Total Laptops with Laptop Bookings and Consumer"), allowing users to analyze rental activities.
  - **Bucket Fields:** Enhance reporting by categorizing data (e.g., "Amount" into "types of versions" like Basic, Intermediate, High, Very High).
  - **Report Subscriptions:** Enable automated delivery of key reports (e.g., daily rental overview to the Owner).
  - **Dashboards:** Provide a visual summary of critical business metrics through charts and graphs, offering quick insights into rental performance and trends (e.g., "data analytics of laptops").

This layered architecture ensures that the system is not just a data repository but a dynamic tool that automates processes, secures information, and provides valuable insights to support the laptop rental business.

#### 1.4. Target Audience of this Guide

This comprehensive guide is primarily intended for individuals involved in the development, administration, and ongoing management of the Salesforce Laptop Rental Management System. Specifically, the target audience includes:

- **Salesforce Administrators:** Individuals responsible for the setup, configuration, customization, and maintenance of the Salesforce instance. This guide will serve as a step-by-step manual for replicating the described system components and understanding their underlying logic.

- **Salesforce Developers:** Although much of the system uses declarative (clicks, not code) tools, sections detailing Apex triggers and handler classes will be particularly relevant for developers looking to understand or extend the programmatic aspects of the solution.
- **Project Managers/Business Analysts:** Those overseeing the implementation or evolution of the rental management solution. This guide provides a detailed understanding of the system's capabilities, design choices, and how it addresses business requirements.
- **Technical Support Personnel:** For troubleshooting and understanding the system's behavior, especially concerning data flow, automation, and user access.
- **Aspiring Salesforce Professionals:** Individuals learning Salesforce who wish to understand a practical, end-to-end implementation of a business application on the platform.

While direct end-users of the Laptop Rental Management System (e.g., agents making bookings) are not the primary audience for the implementation details, understanding the system's architecture and capabilities as described here can indirectly benefit their operational efficiency and comprehension of how their actions within the system translate to business outcomes.

## **2. Salesforce Developer Account Setup & Initial Configuration**

This chapter provides a detailed walkthrough for setting up the foundational environment for our Laptop Rental Management System. This process is crucial as it creates the Salesforce Developer Edition organization (or "org") where all the system's customizations, automation, and data will reside.

### **2.1. Creating a Developer Org**

Before any development or configuration can begin, a Salesforce environment is required. A Developer Edition org is a free, full-featured, non-expiring environment that is perfect for building and testing custom applications like our rental system.

#### **2.1.1. Navigating to the Signup Page**

The first step is to locate the signup page for a Salesforce Developer Edition. This is a public-facing web page where prospective developers can register for their free environment. The URL for this page is <https://developer.salesforce.com/signup>.

1. Open your preferred web browser and navigate to the specified URL.
2. You will be presented with a form to "Sign up for your free Developer Edition," which is the entry point for creating your new Salesforce org.

### 2.1.2. Required Details for Account Creations

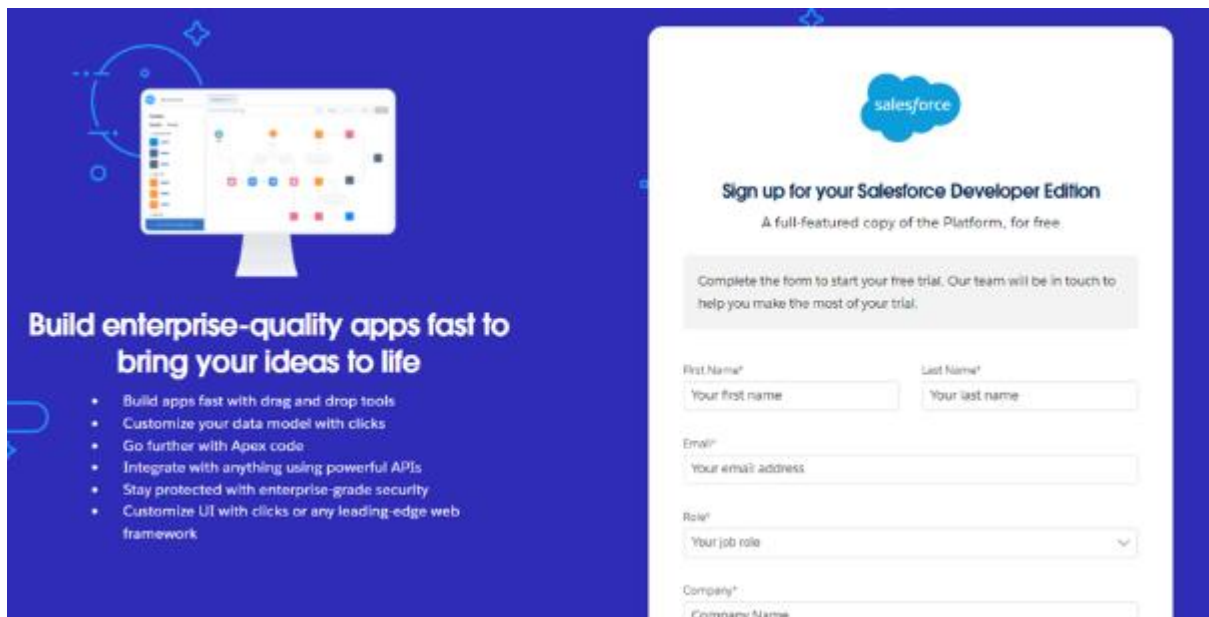
The signup form requires you to provide basic personal and company information. It's important to fill this out accurately to ensure your account is created successfully. The required details are:

1. **First Name & Last Name:** Your personal name.
2. **Email:** A valid, personal email address. This will be the primary contact email for your account and is used for activation and password resets.
3. **Role:** This specifies your professional role. "Developer" is the correct selection.
4. **Company:** The name of your company or, in this context, "College Name" as an example.
5. **Country:** Your geographical location, specified as "India" in the provided reference.
6. **Postal Code:** The corresponding postal or pin code for your location.

**Username:** A unique username for your Salesforce account. This should be a combination of your name and company, formatted like an email address (e.g., username@organization.com).

**Note:** This does not need to be a real, functional email address, but it must be unique across all Salesforce orgs.

After completing the form, click on the "Sign me up" button to submit your registration request.



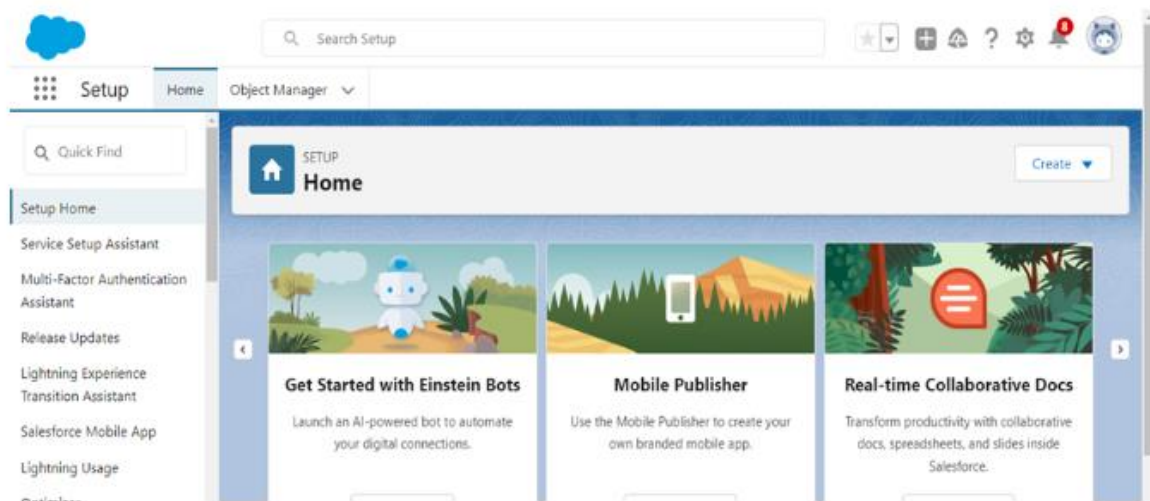
The image shows a promotional banner for Salesforce Developer Edition on the left and a signup form on the right. The banner features a blue background with a white monitor displaying a Salesforce interface. The text on the banner reads: "Build enterprise-quality apps fast to bring your ideas to life". Below this, there is a list of features: "Build apps fast with drag and drop tools", "Customize your data model with clicks", "Go further with Apex code", "Integrate with anything using powerful APIs", "Stay protected with enterprise-grade security", and "Customize UI with clicks or any leading-edge web framework". The signup form on the right has a white background with a blue Salesforce logo at the top. The text on the form reads: "Sign up for your Salesforce Developer Edition" and "A full-featured copy of the Platform, for free." Below this, there is a grey box with the text: "Complete the form to start your free trial. Our team will be in touch to help you make the most of your trial." The form fields are: "First Name\*" (with placeholder "Your first name"), "Last Name\*" (with placeholder "Your last name"), "Email\*" (with placeholder "Your email address"), "Role\*" (with placeholder "Your job role" and a dropdown arrow), and "Company\*" (with placeholder "Company Name").

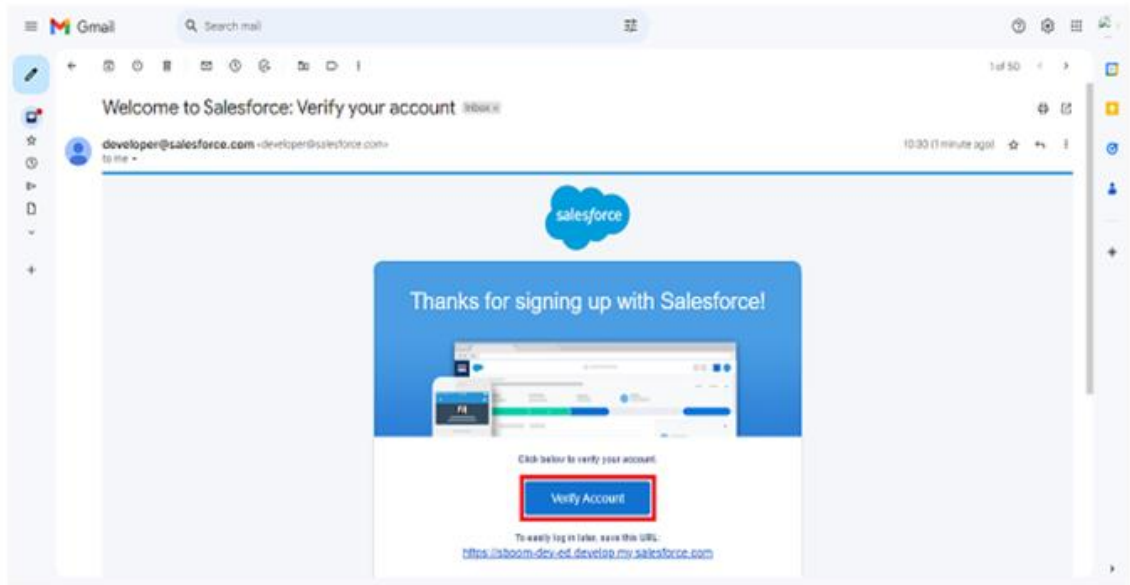
### 2.1.3. Account Activation and Password Setup

Upon successful submission, Salesforce will send an account activation email to the address you provided. This is a critical step to verify your identity and set your initial password.

1. You need to go to the inbox of the email address you registered with.
2. Locate the email from Salesforce with the subject "Welcome to Salesforce: Verify your account."
3. Inside the email, there will be a button or a link labeled "**Verify Account.**" Click this to proceed.
4. This action will redirect you to a page where you are prompted to set your password and a security question. You will be asked to:
  - Enter a **New Password**. Salesforce has strict password requirements (e.g., 8 characters, 1 letter, 1 number), and these are validated as you type.
  - **Confirm New Password**.
  - Answer a **Security Question**. This question is used for account recovery if you forget your password in the future.
5. Click "**Change Password**" to finalize the activation process.

Upon completion, you will be automatically logged into your brand-new Salesforce Developer Edition org, and your initial setup will be complete.





### Change Your Password

Enter a new password for lead@sb.com.  
Make sure to include at least:

- ✓ 8 characters
- ✓ 1 letter
- ✓ 1 number

\* New Password

Good

\* Confirm New Password

Match

Security Question

▼ In what city were you born?

\* Answer

asdfghjkl

Change Password

## 2.2. Understanding Salesforce Setup and Navigation



Once you've logged in for the first time, you will be redirected to the main Salesforce Setup page. This page is the command center for all administrative tasks, including building and configuring our custom application.

- The Setup page features a left-hand navigation menu and a central workspace. The left menu provides a search bar (Quick Find) to quickly locate settings and tools, as well as categorized links for various administrative functions.
- The Quick Find search bar is an essential tool for administrators. By typing a keyword (e.g., "users," "profiles," "objects," "flow"), you can instantly filter the menu and jump to the relevant page, saving a significant amount of time.

To begin building our application, the first major step is to start creating the core data structures, known as custom objects. The instructions in the screenshots consistently guide us to the **Object Manager** for this task.

- To navigate to the Object Manager, you will:
  1. Click the **gear icon** in the top-right corner of the screen.
  2. From the dropdown menu, select **"Setup."**
  3. This brings you back to the main Setup page if you were not already there.
  4. The left-hand navigation menu should now be visible. In the "Quick Find" search box, type Object Manager and click on the result.

The Object Manager is a central repository where you can view, create, and manage all the standard and custom objects in your org. This is where we will create the digital "containers" for our rental data.

### **3. Data Model Design: Custom Objects**

The foundation of any business application on Salesforce is its data model. This defines the core data structures and how they relate to each other. In Salesforce, these structures are called "objects," and for our Laptop Rental Management System, we will be creating custom objects to represent key business entities.

#### **3.1. Understanding Salesforce Objects: Standard vs. Custom**

Before we create our custom objects, it is important to understand the two main types of objects in Salesforce:

- **Standard Objects:** These are pre-built objects provided by Salesforce.com for common business functions, such as Accounts, Contacts, Opportunities, Leads, and Reports. While we could potentially adapt some of these for our purpose, they may not perfectly fit our unique rental process.

- **Custom Objects:** These are objects created by users to store data that is specific to a particular organization's unique needs. They are the heart of a custom application, supplying a structure for data that is not a part of the standard Salesforce data model. Our Laptop Rental Management System will be built primarily with custom objects.

### 3.2. Creating Core Custom Objects

Based on the provided screenshots, our project requires four primary custom objects: Consumer, Laptop Bookings, Billing Process, and Total Laptops. We will create each of these in the Object Manager. The process is consistent for all three.

#### 3.2.1. Creating the Consumer Object

The Consumer object will store all the details about the customers who rent our laptops. This is a crucial object as it is at the center of all rental activities.

1. Navigate to the **Object Manager** (if you're not there, use the gear icon > Setup > Quick Find: Object Manager).
2. In the top right corner of the Object Manager page, click the **"Create"** button, and from the dropdown, select **"Custom Object."** This will open the New Custom Object wizard.
3. Fill in the details for the new object:
  - **Label:** Consumer
  - **Plural Label:** consumers
  - **Record Name:** consumer\_name (The API name will auto-populate as consumer\_name\_\_c).
  - **Data Type:** Text (This is the default for a name field).
4. Scroll down to the "Optional Features" section. Ensure the following checkboxes are selected:
  - **Allow Reports**
  - **Allow Search**
  - **Track Field History**
5. Click **"Save."** This action creates the Consumer object and its default fields.

#### 3.2.2. Creating the Laptop Bookings Object

The Laptop Bookings object will serve as the central hub for each individual rental transaction. It will be the "child" in a relationship with Consumer, linking a booking to a specific person.

1. From the **Object Manager**, click "**Create**" > "**Custom Object.**"
2. Fill in the details:
  - **Label:** Laptop Bookings
  - **Plural Label:** Laptop Bookings
  - **Record Name:** Laptop BookingsName (The API name will auto-populate as Laptop\_BookingsName\_\_c).
  - **Data Type:** Text
3. In the "Optional Features" section, select the following checkboxes:
  - **Allow Reports**
  - **Allow Search**
  - **Track Field History**
4. Click "**Save.**"

### 3.2.3. Creating the Billing Process Object

The Billing Process object will be used to manage all payment-related information for a booking. It will be a child object to both the Consumer and Laptop Bookings objects, ensuring a clear trail of billing activity.

1. From the **Object Manager**, click "**Create**" > "**Custom Object.**"
2. Fill in the details:
  - **Label:** Billing Process
  - **Plural Label:** Billing Process
  - **Record Name:** Billing ProcessName (The API name will auto-populate as Billing\_ProcessName\_\_c).
  - **Data Type:** Text
3. In the "Optional Features" section, select the following checkboxes:
  - **Allow Reports**

- **Allow Search**
- **Track Field History**

4. Click "**Save.**"

### 3.2.4. Creating the Total Laptops Object

While not explicitly shown in an initial "Create Object" screenshot like the others, the Total Laptops object is implicitly defined in later steps as the object containing the total inventory and related rollup fields. For the sake of a complete and logical guide, we will create this object now.

1. From the **Object Manager**, click "**Create**" > "**Custom Object.**"
2. Fill in the details:
  - **Label:** Total Laptops
  - **Plural Label:** Total Laptops
  - **Record Name:** Total LaptopsName (The API name will auto-populate as Total\_LaptopsName\_\_c).
  - **Data Type:** Text
3. In the "Optional Features" section, select the following checkboxes:
  - **Allow Reports**
  - **Allow Search**
  - **Track Field History**
4. Click "**Save.**"

With these four custom objects created, we have successfully established the foundational data structure for our rental management system. The next step is to populate these objects with the specific fields and relationships that will hold all the necessary data.

## 4. Data Model Design: Custom Fields & Relationships

Now that our custom objects are in place, we must define the fields within them and the relationships that connect them. These fields are where the actual data will be stored, and

the relationships are what make the system an integrated database rather than a collection of isolated tables. This chapter details the creation of these fields and relationships.

#### 4.1. Creating Fields for the Consumer Object

The Consumer object will be the repository for all customer data. The following fields will be added to ensure we have comprehensive contact and demographic information. The process for creating fields is consistent across all objects: navigate to the object in the **Object Manager**, select the **"Fields & Relationships"** section, and click **"New."**

##### 4.1.1. Phone Number Field

This field will store the customer's contact number. The field is of type Phone.

1. From the Consumer object page in the Object Manager, click **"Fields & Relationships"** and then **"New."**
2. Select the **"Phone"** data type and click **"Next."**
3. Fill in the details:
  - **Field Label:** Phone number
  - **Field Name:** This will auto-generate.
4. Select the **"Required"** option checkbox to ensure this field is always populated.
5. Click **"Next"** and then **"Save & New"** to proceed to the next field.

##### 4.1.2. Email Field

This field will store the customer's email address. The process is similar to the phone number field.

1. Select the **"Email"** data type and click **"Next."**
2. Fill in the details:
  - **Field Label:** Email
  - **Field Name:** This will auto-generate.
3. Click **"Next"** and then **"Save & New."**

##### 4.1.3. Address Field

This field will capture the customer's address. We will use a Text Area data type to allow for multi-line input.

1. Select the **"Text Area"** data type and click **"Next."**
2. Fill in the details:

- **Field Label:** Address
  - **Field Name:** This will auto-generate.
3. Select the "**Required**" checkbox.
  4. Click "**Next**" and then "**Save & New.**"

#### 4.1.4. Consumer Status Picklist Field

To categorize our customers, we will use a Picklist field with pre-defined values. This ensures data consistency and makes it easy to segment customers for reporting.

1. Select the "**Picklist**" data type and click "**Next.**"
2. Fill in the details:
  - **Field Label:** Consumer Status
  - **Values:** Select "Enter values, with each value separated by a new line."
  - **Enter the values:**
    - Student
    - Employee
    - Others
  - Select the "**Required**" checkbox.
3. Click "**Next**" and then "**Save & New.**"

## 4.2. Creating Fields for the Laptop Bookings Object

The Laptop Bookings object is critical for capturing all the specifics of a rental. We will add fields to detail the type of laptop, rental duration, and the associated cost.

### 4.2.1. Laptop Names Picklist Field

This picklist will define the different laptop brands available for rent.

1. Navigate to the Laptop Bookings object in the Object Manager and click "**Fields & Relationships**" > "**New.**"
2. Select "**Picklist**" and click "**Next.**"
3. Fill in the details:
  - **Field Label:** Laptop Names
  - **Values:**
    - Dell

- Acer
  - Hp
  - Mac
  - Select the "**Required**" checkbox.
4. Click "**Next**" and then "**Save & New.**"

#### 4.2.2. Core Type Picklist Field

This field will specify the processor type of the rented laptop. This is an important detail for the pricing logic.

1. Select "**Picklist**" and click "**Next.**"
2. Fill in the details:
  - **Field Label:** Core Type
  - **Values:**
    - Core i3
    - Core i5
    - Core i7
    - Bionic Chip
  - Select the "**Required**" checkbox.
3. Click "**Next**" and then "**Save & New.**"

#### 4.2.3. How Many Months Picklist Field

To track the rental duration, this picklist will offer a selection of monthly increments.

1. Select "**Picklist**" and click "**Next.**"
2. Fill in the details:
  - **Field Label:** how many months
  - **Values:**
    - 1
    - 2
    - 3
    - 4

- 5

- Select the **"Required"** checkbox.

3. Click **"Next"** and then **"Save & New."**

#### 4.2.4. Email Field

An Email field on the booking record is a good practice to easily access the customer's email for communication.

1. Select the **"Email"** data type and click **"Next."**
2. Fill in the details:
  - **Field Label:** Email
  - **Field Name:** This will auto-generate.
3. Click **"Next"** and then **"Save."**

#### 4.2.5. Amount (Currency) Field

This field will store the total rental amount, which will be populated by our automation flow.

1. Select the **"Currency"** data type and click **"Next."**
2. Fill in the details:
  - **Field Label:** Amount
  - **Length:** 18
  - **Decimal Places:** 0
3. Click **"Next"** and then **"Save & New."**

### 4.3. Creating Fields for the Billing Process Object

This object will contain details relevant to the financial transaction.

#### 4.3.1. Payment Mode Picklist Field

This field will track how a customer paid for their rental, which is crucial for financial reporting.

1. Navigate to the Billing Process object and click **"Fields & Relationships" > "New."**
2. Select the **"Picklist"** data type and click **"Next."**
3. Fill in the details:
  - **Field Label:** Payment Mode



- **Values:**
  - Cash
  - Check
  - Credit card
  - Debit card
  - UPI
  - Phonepe
  - Gpay
  - Paytm
- Select the "**Required**" checkbox.

4. Click "**Next**" and then "**Save.**"

#### **4.4. Defining Relationships Between Objects**

Relationships are the connectors that link our objects together, allowing us to build a relational database model within Salesforce. We will create Master-Detail and Lookup relationships to connect our custom objects.

##### **4.4.1. Master-Detail Relationship: Laptop Booking to Consumer**

This relationship links each rental booking to a specific customer. This is a Master-Detail relationship because a booking is meaningless without a customer, making it a perfect use case for a tight link.

1. In the Laptop Bookings object, click "**Fields & Relationships**" > "**New.**"
2. Select the "**Master-Detail Relationship**" data type and click "**Next.**"
3. On the next screen, select Consumer from the Related To dropdown and click "**Next.**"
4. Fill in the details:
  - **Field Label:** Consumer
  - **Field Name:** This will auto-generate.
5. Click "**Next**", "**Next**" (accepting default visibility and page layout settings), and then "**Save.**"

##### **4.4.2. Lookup Relationship: Laptop Booking to Total Laptops**

This relationship links a booking to a specific laptop in our inventory. A Lookup is appropriate here because a Laptop Booking record might be deleted, but the Total Laptops inventory record should remain intact.

1. In the Laptop Bookings object, click "**Fields & Relationships**" > "**New.**"
2. Select the "**Lookup Relationship**" data type and click "**Next.**"
3. On the next screen, select Total Laptops from the Related To dropdown and click "**Next.**"
4. Fill in the details:
  - **Field Label:** Total No Of Laptops
  - **Field Name:** This will auto-generate.
5. Click "**Next**", "**Next**", and then "**Save.**"

#### **4.4.3. Master-Detail Relationship: Billing Process to Consumer**

This relationship links a billing record to the customer responsible for it. Like the Laptop Bookings relationship, this is a strong, Master-Detail connection.

1. In the Billing Process object, click "**Fields & Relationships**" > "**New.**"
2. Select the "**Master-Detail Relationship**" data type and click "**Next.**"
3. Select Consumer from the Related To dropdown and click "**Next.**"
4. Fill in the details:
  - **Field Label:** Name
  - **Field Name:** This will auto-generate.
5. Click "**Next**", "**Next**", and then "**Save.**"

#### **4.4.4. Lookup Relationship: Billing Process to Laptop Booking**

This relationship links a billing record to a specific booking record. A Lookup is appropriate because a billing record might be created after the booking is finalized, and a billing record's deletion should not necessarily affect the booking record.

1. In the Billing Process object, click "**Fields & Relationships**" > "**New.**"
2. Select the "**Lookup Relationship**" data type and click "**Next.**"
3. Select Laptop Bookings from the Related To dropdown and click "**Next.**"
4. Fill in the details:
  - **Field Label:** Laptop Booking

- **Field Name:** This will auto-generate.
5. Click "**Next**", "**Next**", and then "**Save**."

With these fields and relationships in place, we have completed the core data model. The system now has the necessary structure to store and link all the information related to a laptop rental. The next chapter will focus on advanced field types and business logic, which will add intelligence and automation to this data model.

## 5. Advanced Data Management: Formulas, Roll-ups, and Dependencies

With the core objects and their fields defined, we can now add intelligence to our data model using advanced field types. This chapter will cover the creation of formula fields, roll-up summary fields, and field dependencies, which automate calculations and enforce logical relationships between data.

### 5.1. Understanding Formula Fields

A **Formula Field** is a read-only field that automatically calculates its value based on an expression you define. The formula can reference other fields on the same record or on related records. This eliminates the need for manual calculations and ensures data accuracy.

### 5.2. Creating "Laptops Available" Formula Field (Total Laptops Object)

This formula will provide a real-time count of available laptops based on a total inventory count of 50, subtracting the number of laptops that have been delivered.

1. Navigate to the Total Laptops object in the Object Manager and click "**Fields & Relationships**" > "**New**."
2. Select the "**Formula**" data type and click "**Next**."
3. Fill in the details:
  - **Field Label:** Laptops Available
  - **Formula Return Type:** Number
  - **Decimal Places:** 0
4. Click "**Next**."
5. In the formula editor, enter the formula: 50 - Laptops\_delivered\_\_c.
6. Click "**Check Syntax**" to ensure there are no errors, then "**Next**" and "**Save**." This field will automatically calculate the number of available laptops.

### 5.3. Creating "Laptops Available" Formula Field (Laptop Booking Object)

This formula is another example of calculating availability, but this time, it directly references the Total Laptops object. This demonstrates a cross-object formula, pulling data from a parent record to a child record.

1. Navigate to the Laptop Bookings object and click "**Fields & Relationships**" > "**New.**"
2. Select "**Formula**" and click "**Next.**"
3. Fill in the details:
  - **Field Label:** Laptops Available
  - **Formula Return Type:** Number
  - **Decimal Places:** 0
4. Click "**Next.**"
5. In the formula editor, use the Insert Field button to select the rollup field we will create later on the Total Laptops object. The formula should be: 50 - Total\_no\_of\_laptops\_\_r.Laptops\_delivered\_\_c.
6. Click "**Check Syntax,**" then "**Next**" and "**Save.**"

#### 5.4. Understanding Roll-up Summary Fields

A **Roll-up Summary Field** is a read-only field that displays the count, sum, min, or max value of records in a related list. It's only available on the "master" side of a Master-Detail relationship. For our project, we need to count how many laptops have been booked to track inventory.

#### 5.5. Creating "Laptops delivered" Roll-up Summary Field (Total Laptops Object)

This field will count the number of Laptop Bookings records associated with a Total Laptops record. This is a crucial part of our inventory management logic.

1. Navigate to the Total Laptops object and click "**Fields & Relationships**" > "**New.**"
2. Select the "**Roll-up Summary**" data type and click "**Next.**"
3. Fill in the details:
  - **Field Label:** Laptops delivered
  - **Field Name:** This will auto-generate.
4. Click "**Next.**"
5. On the next screen, define the summary calculation:
  - **Summarized Object:** Laptop Bookings (this is the child object we're counting).

- **Roll-up Type:** Select the **"COUNT"** radio button.
  - Leave the filter criteria as default to include all records in the count.
6. Click **"Next"** and **"Save."**

### 5.6. Creating Cross-Object Formula Field "Amount" (Billing Process Object)

This formula will pull the total rental amount directly from the related Laptop Booking record, ensuring consistency between the booking and billing details.

1. Navigate to the Billing Process object and click **"Fields & Relationships" > "New."**
2. Select the **"Formula"** data type and click **"Next."**
3. Fill in the details:
  - **Field Label:** Amount
  - **Formula Return Type:** Currency
4. Click **"Next."**
5. In the formula editor, use Insert Field to select the Amount field on the Laptop Booking object. The formula should be: `Laptop_Booking__r.Amount__c`.
6. Click **"Check Syntax,"** then **"Next"** and **"Save."**

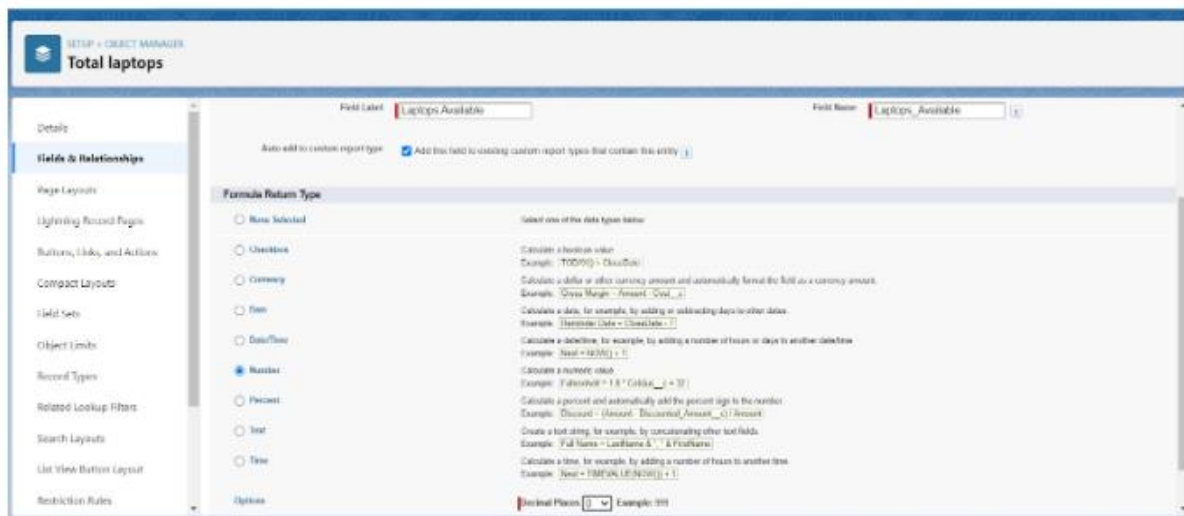
### 5.7. Implementing Field Dependencies: Laptop Name and Core Type

A **Field Dependency** links two picklist fields, where the values available in the dependent field are determined by the value selected in the controlling field. This is essential for creating a user-friendly and logical booking process. We will make "Core Type" dependent on "Laptop Names."

1. Navigate to the Laptop Bookings object and click **"Fields & Relationships."**
2. Click the **"Field Dependencies"** button in the top section.
3. Click **"New."**
4. Select Laptop Names as the **Controlling Field** and Core Type as the **Dependent Field** and click **"Continue."**
5. The next screen is a matrix where you define which core types are available for each laptop brand. Select the appropriate core types for each brand:
  - For Dell: Select Core i3, Core i5, Core i7.
  - For Acer: Select Core i3, Core i5, Core i7.
  - For Hp: Select Core i3, Core i5, Core i7.

- For Mac: Select Bionic Chip.

## 6. Click "Save."



This completes the advanced configuration of our data model. The system can now perform automated calculations and present a more intelligent user experience.

## 6. User Experience & Application Access (Lightning App & Tabs)

With the backend data model in place, we can now build the user-facing part of our application. This involves creating custom tabs for our objects and grouping them into a cohesive Lightning App. These steps are crucial for providing a user-friendly navigation experience for our team.

### 6.1. Understanding Salesforce Tabs

A **Tab** is a user interface element in Salesforce that provides a way to interact with an object's records. We will be using **Custom Tabs** for our custom objects. These tabs are the entry points that allow users to view, create, and edit records for our Consumer, Laptop Bookings, and other objects.

### 6.2. Creating Custom Tabs for Objects



We'll create a custom tab for each of our core objects. The process is consistent for all of them.

Navigate to the Salesforce Setup page (gear icon > Setup).

1. In the Quick Find box, type Tabs and select the **Tabs** menu item.
2. Scroll down to the "Custom Object Tabs" section and click "**New.**"
3. **For the Total Laptops object:**
  - In the Object dropdown, select Total Laptops.
  - Choose an appropriate **Tab Style** (icon) from the selector.
  - Click "**Next.**"
  - On the "Add to Profiles" page, keep the default settings for now. Click "**Next.**"
  - On the "Add to Custom Apps" page, uncheck any apps you don't want the tab to be visible in. Make sure it's available for the app we'll create later. Click "**Save.**"
4. **For the remaining objects (consumer, Laptop Booking, Billing Process):**
  - Repeat the process above, creating a new custom tab for each of them.
  - Ensure you choose a distinct icon for each tab to improve navigation.

### 6.3. Creating a Lightning Application: "LAPTOP RENTALS"

To give our users a centralized workspace, we will create a custom Lightning App that contains all our new tabs. This app will be the main entry point for the rental team.

#### 6.3.1. Initiating a New Lightning App

1. From the Setup page, use the Quick Find box to search for App Manager and select it.
2. In the App Manager, click "**New Lightning App**" in the top right corner. The New Lightning App wizard will launch.

#### 6.3.2. App Details and Branding

This step involves giving our app a name and basic branding.

1. In the "App Details & Branding" section:
  - **App Name:** LAPTOP RENTALS
  - **Developer Name:** This will auto-populate.



- **Description:** Provide a brief description of the app's purpose.
- You can optionally upload an image for the app's logo and choose a primary color.

2. Click "**Next.**"

### 6.3.3. Adding Navigation Items

Now, we'll select which tabs will be included in our app's navigation bar.

1. On the "Navigation Items" page, use the "Available Items" list to select the tabs we just created: Total Laptops, consumers, Laptop Booking, and Billing Process.
2. Move them to the "Selected Items" list using the arrow button. You can reorder them to your preference.
3. Click "**Next.**"

### 6.3.4. Assigning User Profiles to the App

Finally, we need to specify which users can see and access this new app.

1. On the "User Profiles" page, use the list of Available Profiles.
2. Find the System Administrator profile and move it to the Selected Profiles list. This ensures you, as the administrator, can see and test the app.
3. Click "**Save & Finish.**"

Our LAPTOP RENTALS app is now created and ready for use. You can access it by clicking the App Launcher (9 dots in the top left) and selecting it from the list.

## 7. User and Access Management: Profiles and Roles

Proper security is paramount for any business application. In Salesforce, this is managed through **Profiles** and **Roles**. Profiles control what users can *do* (their permissions), while Roles control what data they can *see* (hierarchy-based access). We will create custom profiles and roles for our rental team.

### 7.1. Understanding Salesforce Profiles

A **Profile** is a collection of settings and permissions that determines what a user can see and do within a Salesforce org. Every user must have exactly one profile. Instead of editing standard profiles, best practice is to clone them and then customize the cloned version.

### 7.2. Creating the "Owner" Profile

The Owner profile will have broad access to our rental management objects, allowing for full management of the business.

1. In the Setup Quick Find box, type Profiles and select **Profiles**.
2. Find and click on the "**Standard User**" profile.
3. Click the "**Clone**" button.
4. Enter the Profile Name: Owner.
5. Click "**Save.**"
6. Now, on the newly created "Owner" profile page, scroll down to the "Custom Object Permissions" section.
7. We need to grant the appropriate permissions to our four custom objects: Total Laptops, consumers, Laptop Bookings, and Billing Process. For the Owner, this will likely be a high level of access, including Read, Create, Edit, Delete, and potentially View All and Modify All. Ensure these are checked for each of our custom objects.
8. Click "**Save.**"

### 7.3. Creating the "Agent" Profile

The Agent profile will be for the day-to-day rental staff who handle bookings. They need to create and edit records but may not need the full administrative or deletion capabilities of the Owner.

1. From the Profiles page, find and click on the "**Standard Platform User**" profile.
2. Click "**Clone.**"
3. Enter the Profile Name: Agent.
4. Click "**Save.**"
5. On the new "Agent" profile page, scroll down to the "Custom Object Permissions" section.
6. Grant Read, Create, and Edit permissions for the four custom objects. Delete permission should likely be unchecked to prevent accidental data loss.
7. Click "**Save.**"

### 7.4. Understanding Salesforce Roles and Hierarchy

A **Role** is a position in a hierarchy that, in conjunction with sharing rules, primarily controls record visibility. Users at the top of the hierarchy can see data owned by users below them. Roles are not mandatory but are essential for managing data visibility in larger organizations.

### 7.5. Creating the "Owner" Role

The Owner is at the top of our rental management hierarchy.

1. In the Setup Quick Find box, type Roles and select **Roles**.
2. Click "**Set Up Roles**."
3. Click "**Expand All**" to see the full hierarchy.
4. Find a high-level role (e.g., CEO) and click "**Add Role**" under it.
5. Enter the Label: owner. The Role Name will auto-populate.
6. Click "**Save**."

## 7.6. Creating the "Agent" Role

The Agent role will be a subordinate role to the Owner. This ensures the Owner can see all records owned by the Agents.

1. From the Roles hierarchy page, find the owner role you just created and click "**Add Role**" under it.
2. Enter the Label: Agent. The Role Name will auto-populate.
3. Click "**Save**."

## 7.7. Creating Users and Assigning Roles/Profiles

Finally, we'll create the actual user accounts and assign them to the new roles and profiles.

### 7.7.1. Creating "vicky" (Owner) User

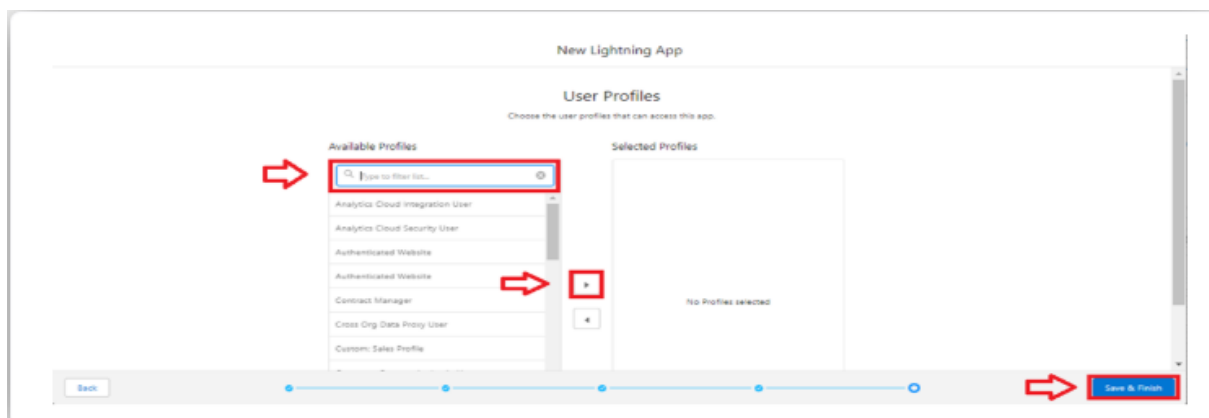
1. In the Setup Quick Find box, type Users and select **Users**.
2. Click "**New User**."
3. Fill in the required fields:
  - **First Name:** vicky
  - **Last Name:** y
  - **Email:** your.personal.email@example.com
  - **Username:** A unique username in the format text@text.text
  - **Alias:** An alias name
  - **Nickname:** A nickname
  - **Role:** Select the owner role.
  - **User License:** Salesforce

- **Profile:** Select the Owner profile.
4. Click "**Save.**"

### 7.7.2. Creating "ram" (Agent) User

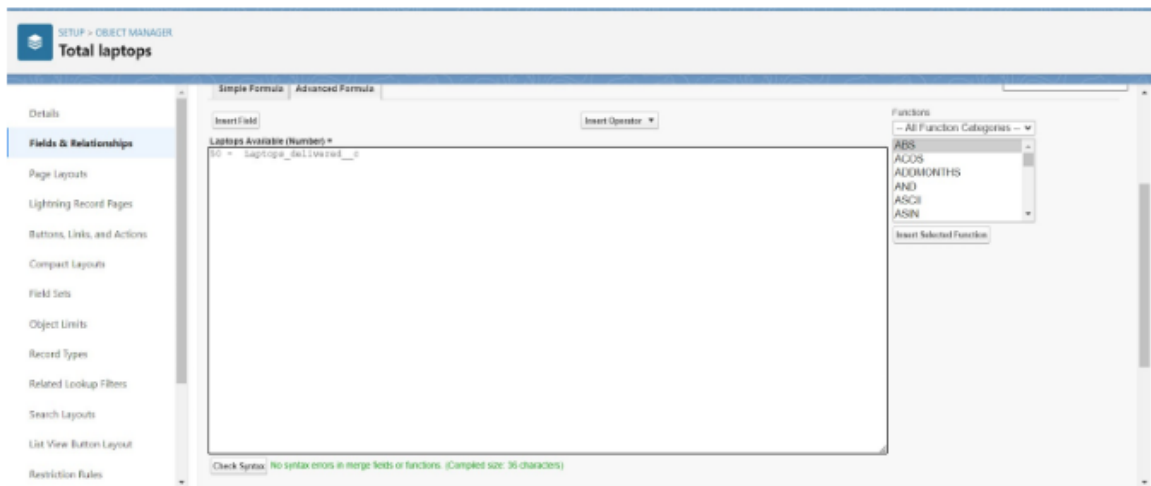
1. Repeat the process for creating a new user.
2. Fill in the details:
  - **First Name:** ram
  - **Last Name:** ram
  - **Email:** your.personal.email@example.com
  - **Username:** A unique username in the format text@text.text
  - **Alias:** An alias name
  - **Nickname:** A nickname
  - **Role:** Select the Agent role.
  - **User License:** Salesforce Platform
  - **Profile:** Select the Agent profile.
3. Click "**Save.**"

With users, roles, and profiles configured, our system now has a secure foundation for user access and data visibility. The next chapter will focus on automating the core business logic using Salesforce Flow.



## 8. Business Process Automation: Salesforce Flow

Salesforce Flow is a powerful automation tool that allows us to build complex business logic with clicks, not code. For our rental system, we will use a **Record-Triggered Flow** to automatically calculate the rental amount based on the user's selections. This eliminates manual pricing and ensures accuracy.



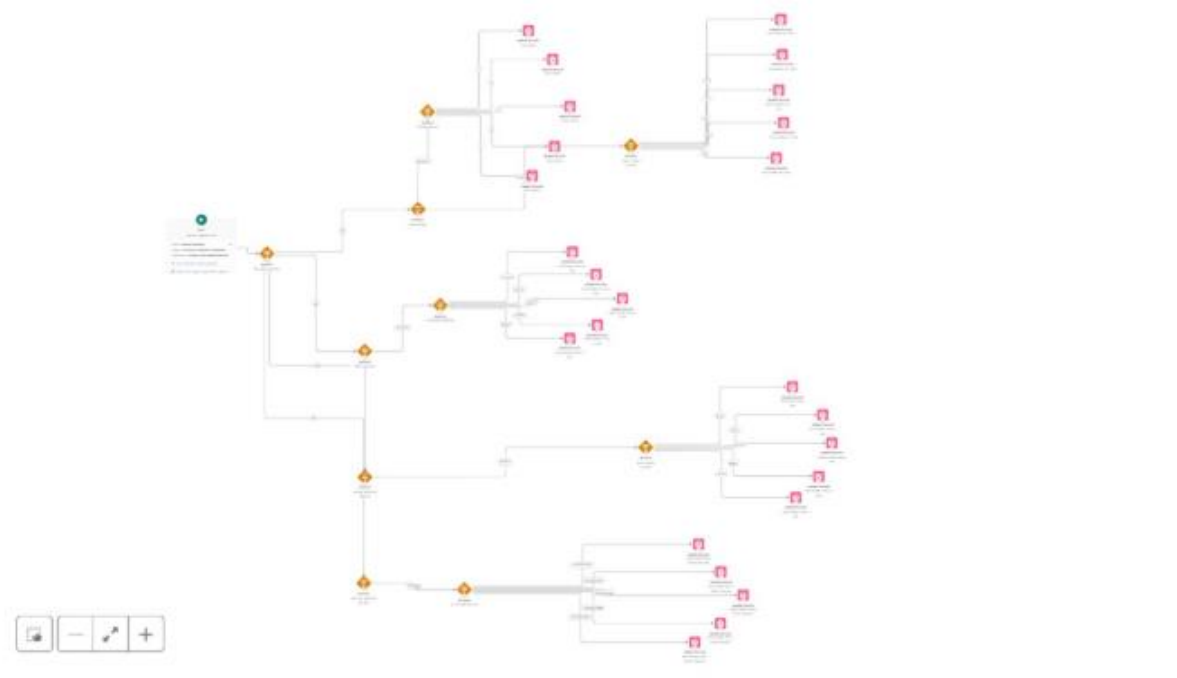
## 8.1. Introduction to Salesforce Flow (Record-Triggered Flow)

A Record-Triggered Flow runs in the background when a record is created, updated, or deleted. This is the perfect tool for our use case, as we need to update the Amount field on the Laptop Booking record as soon as a user selects the laptop and duration.

## 8.2. Initiating the Laptop Booking Flow

The first step is to create a new flow and define its trigger conditions.

1. In the Setup Quick Find box, type Flows and select **Flows**.
2. Click **"New Flow."**
3. From the "New Flow" wizard, select **"Record-Triggered Flow"** and click **"Create."**
4. On the next screen, configure the flow's trigger:
  - **Object:** Select Laptop Bookings.
  - **Trigger the Flow When:** Select **"A record is Created or Updated."**
  - **Optimize the Flow for:** Select **"Actions and Related Records."**
5. Click **"Done."** This will open the Flow Builder canvas.



### 8.3. Implementing Pricing Logic with Decisions and Updates

The core of our flow is a series of **Decision** elements that will determine the correct price for the rental. We will branch the flow based on the selected laptop, core type, and duration.

#### 8.3.1. Decision Element for Laptop Type

The first decision will check which laptop brand was selected.

1. On the Flow Builder canvas, click the "+" button and select "**Decision.**"
2. **Label:** Check Laptop Type
3. **Outcome 1:**
  - **Label:** Dell
  - **Condition:** Resource: \$Record.Laptop\_names\_\_c, Operator: Equals, Value: 'Dell'
4. **Outcome 2:**
  - **Label:** Acer
  - **Condition:** Resource: \$Record.Laptop\_names\_\_c, Operator: Equals, Value: 'Acer'
5. **Outcome 3:**
  - **Label:** HP

- **Condition:** Resource: \$Record.Laptop\_names\_\_c, Operator: Equals, Value: 'Hp'
6. **Outcome 4:**
- **Label:** Mac
  - **Condition:** Resource: \$Record.Laptop\_names\_\_c, Operator: Equals, Value: 'Mac'
7. Click "**Done.**"

### 8.3.2. Decision Elements for Core Type

For each laptop brand outcome, we need to create a nested decision to check the core type.  
For the "Dell" outcome:

1. After the "Dell" outcome, click the "+" and add another "**Decision**" element.
2. **Label:** Check Dell Core Type
3. **Outcome 1:**
  - **Label:** Dell Core i3
  - **Condition:** Resource: \$Record.core\_type\_\_c, Operator: Equals, Value: 'Core i3'
4. **Outcome 2:**
  - **Label:** Dell Core i5
  - **Condition:** Resource: \$Record.core\_type\_\_c, Operator: Equals, Value: 'Core i5'
5. **Outcome 3:**
  - **Label:** Dell Core i7
  - **Condition:** Resource: \$Record.core\_type\_\_c, Operator: Equals, Value: 'Core i7'
6. Click "**Done.**"
7. Repeat this for the Acer, HP, and Mac outcomes, with different core types as needed.

### 8.3.3. Decision Elements for Rental Duration (Months Selected)

For each core type outcome, we need to add a final decision to check the number of months.

1. After the Dell Core i3 outcome, add a new **"Decision"** element.
2. **Label:** Check Months for Dell i3
3. Create five outcomes, one for each month:
  - **Outcome 1:** Resource: \$Record.how\_many\_months\_\_c, Operator: Equals, Value: '1'
  - **Outcome 2:** Resource: \$Record.how\_many\_months\_\_c, Operator: Equals, Value: '2'
  - ... and so on up to 5 months.
4. Click **"Done."**
5. Repeat this process for all other core type outcomes (e.g., Dell Core i5, Dell Core i7, etc.).

#### 8.3.4. Updating Rental Amount Based on Decisions

The final step for each path in our nested decisions is to update the Amount field on the Laptop Booking record with the correct price.

1. After the Dell Core i3 and 1 month outcome, click the "+" and select **"Update Records."**
2. **Label:** Update Amount for Dell i3 (1 month)
3. **How to Find Records to Update:** Use the Laptop Booking record that triggered the flow.
4. **Set Field Values for the Record:**
  - **Field:** Amount
  - **Value:** 1000
5. Click **"Done."**
6. Repeat this Update Records step for every possible combination of Laptop, Core Type, and Duration, using the specific pricing values:
  - Dell i3: 1000, 2000, 3000, 4000, 5000
  - Dell i5: 1500, 2500, 3500, 4500, 5500
  - Dell i7: 2000, 4000, 6000, 8000, 10000



- (and so on for other brands/core types)
7. After all paths have an Update Records element, click **"Save"** in the top right, give the flow a name, and then **"Activate"** it.

This complex flow automates our entire pricing model, ensuring that every booking is priced accurately and instantly without any manual intervention from the rental agent.

## 9. Advanced Automation: Apex Triggers and Handlers

While Salesforce Flow handles our declarative automation needs, complex or programmatic requirements often call for Apex, Salesforce's proprietary programming language. For our system, we'll use an Apex Trigger to automatically send a welcome email to the customer whenever a new booking is created or an existing one is updated.

### 9.1. Introduction to Apex Triggers and Handler Classes

- An **Apex Trigger** is a piece of code that executes before or after a specific database event (like insert, update, delete). Triggers are ideal for performing actions in response to data changes.
- A best practice is to have a **Trigger Handler Class** to contain the business logic. The trigger itself should be lightweight and simply call the methods in the handler class. This makes the code easier to manage, test, and debug.

### 9.2. Creating the Apex Trigger

The trigger will be configured to run on our Laptop\_Bookings\_\_c object.

1. In the Setup menu, click the gear icon and select **"Developer Console."** This opens a new browser window.
2. In the Developer Console, click on the **"File"** menu, then **"New"**, and select **"Apex Trigger."**
3. Enter the trigger details:
  - **Name:** LaptopBooking
  - **sObject:** Laptop\_Bookings\_\_c (this is the API name of our object)
4. Click **"Submit."**
5. The trigger code editor will open. Enter the following code:

Apex

```
trigger LaptopBooking on Laptop_Bookings__c (after insert,after update) {  
    if(trigger.isAfter && (trigger.isInsert || trigger.isUpdate)) {
```

```

        LaptopBookingHandler.sendEmailNotification(trigger.new);
    }
}

```

This code defines the trigger to run after insert and after update events. The if statement ensures the logic only runs after the records have been successfully saved to the database. It then calls the sendEmailNotification method from our LaptopBookingHandler class, passing the new records (trigger.new) to it.

## 6. Click "Save."



## 9.3. Developing the Apex Handler Class (LaptopBookingHandler)

The handler class contains the actual logic for sending the email. This is where we will construct the email content using the data from the booking record.

1. In the Developer Console, click on the "File" menu, then "New", and select "Apex Class."
2. Enter the class name: LaptopBookingHandler.
3. Click "OK."
4. Enter the code:

Apex

```
public class LaptopBookingHandler {
```

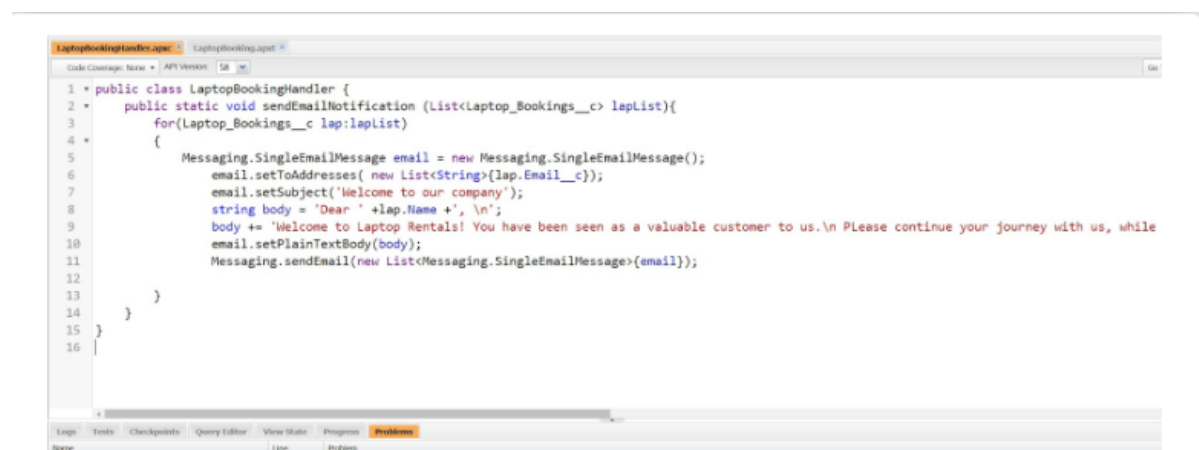
```

public static void sendEmailNotification(List<Laptop_Bookings__c> lapList){
    for(Laptop_Bookings__c lap:lapList) {
        Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
        email.setToAddresses(new List<String>{lap.Email__c});
        email.setSubject('Welcome to our company');
        String body = 'Dear Customer, \n' +
            'Welcome to Laptop Rentals! You have been seen as a valuable customer to us.\n' +
            'Please continue your journey with us, while we try to provide you with good
quality resources.\n' +
            'Laptop Amount = ' + lap.Amount__c + '\n' +
            'Core type = ' + lap.Core_type__c + '\n' +
            'Laptop type = ' + lap.Laptop_names__c;
        email.setPlainTextBody(body);
        Messaging.sendEmail(new List<Messaging.SingleEmailMessage>{email});
    }
}
}
}

```

This class defines a static method that takes a list of Laptop\_Bookings\_\_c records. It then iterates through the list, creates an email for each record, and populates the subject and body with relevant details from the record using the API names of our fields (Email\_\_c, Amount\_\_c, Core\_type\_\_c, Laptop\_names\_\_c). Finally, it sends the email.

## 5. Click "Save."



Now, whenever a booking is created or updated, our Apex trigger will fire, and the handler class will send a personalized email to the customer. This automates a crucial communication step and improves the customer experience.

## **10. Reporting and Analytics: Gaining Business Insights**

After building our application and automating its processes, the final step is to leverage the data we are collecting to gain valuable business insights. Salesforce's reporting and dashboard features allow us to visualize key performance indicators (KPIs) and make data-driven decisions.

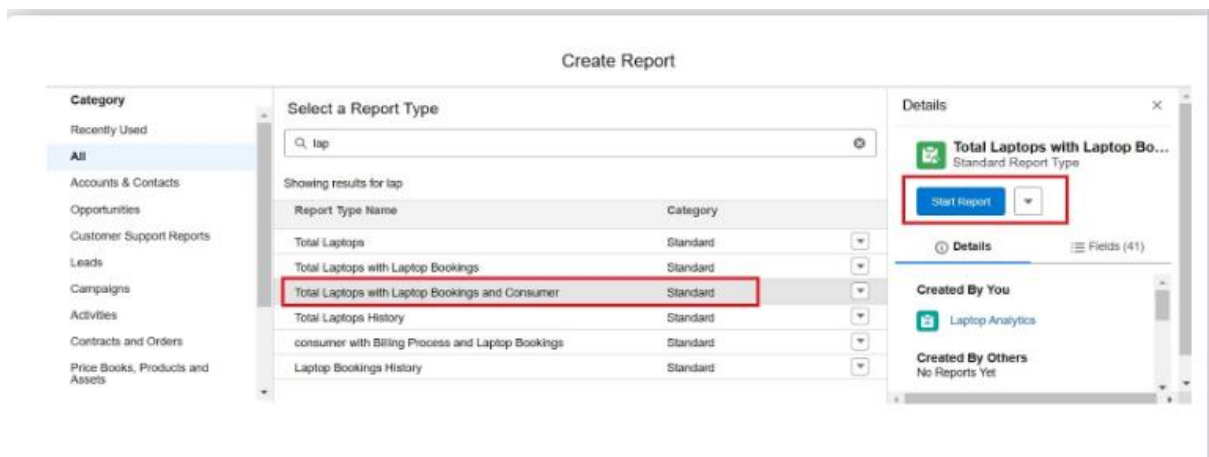
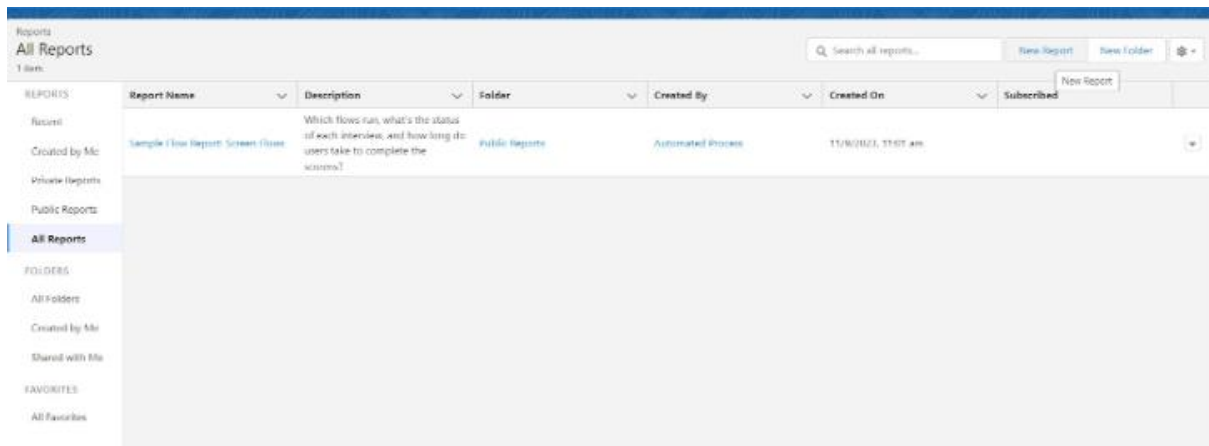
### **10.1. Introduction to Salesforce Reports**

A **Report** is a list of records that meet criteria you define. We will create reports to analyze our rental data, such as which laptops are being rented most frequently or our total revenue.

### **10.2. Creating a Standard Report for Laptop Bookings**

First, we need to create a few sample records in our Laptop Bookings object to have data to report on. Once that's done, we can create our report.

1. In the App Launcher (9 dots), select the **LAPTOP RENTALS** app.
2. Click on the **Reports** tab.
3. Click "**New Report.**"
4. In the "Choose Report Type" search bar, type lap and select the report type that best combines our data, like "**Total Laptops with Laptop Bookings and Consumer**".
5. Click "**Start Report.**"
6. The report builder will open. Drag and drop fields like Consumer Name, Amount, Laptop Names, and Laptops Available from the "Fields" pane on the left to the report preview.



### 10.3. Enhancing Reports with Bucket Fields

A **Bucket Field** allows you to categorize and group report data without creating a new formula field. We will use this to group our rental amounts into different tiers.

#### 10.3.1. Defining "Types of Versions" Bucket

We'll categorize the Amount field.

1. On the report preview, click the dropdown arrow on the Amount column header and select **"Bucket This Column."**
2. **Bucket Name:** types of versions
3. Define the ranges:
  - Range  $\leq 900$ : Label it basic
  - Range  $> 900$  to 1500: Label it intermediate
  - Range  $> 1500$  to 10000: Label it high
  - Range  $> 10000$ : Label it very high
4. Click **"Apply."**

Edit Bucket Column

\* Field  
Amount ×

\* Bucket Name  
types of versions

	Range	Bucket	
Add ▶	<= 900	basic	×
Add ▶	> 900 to 1500	intermediate	×
Add ▶	> 1,500 to 10000	high	×
	> 10,000	very high	×

☒ Treat empty Amount values in the report as zeros.

Cancel
Apply

### 10.3.2. Grouping Reports by Bucket

Now, we'll use this new bucket field to create a summary report.

1. Drag the new **Types of Versions** bucket field from the "Fields" pane into the "Group Rows" section of the report builder.
2. The report will now be grouped by these categories, with subtotals for each.
3. Click **"Save & Run"** to see the final report.

### 10.4. Automating Report Delivery: Subscriptions

Report subscriptions allow us to automatically send report results to key stakeholders on a schedule. We'll set this up for the owner to receive a daily report.

1. From the report you just saved, click the dropdown arrow next to the "Edit" button and select **"Subscribe."**
2. On the "Edit Subscription" page:
  - **Frequency:** Daily
  - **Time:** 8:00 am
  - **Recipients:** Click **"Edit Recipients"** and select the "vicky" user (our owner).
  - **Run Report As:** Select Another Person and choose the "vicky" user to ensure the report runs with the owner's permissions.
3. Click **"Save."**

aitam195-dev-ed.develop.lightning.force.com/lightning/r/Report/00Od200000AFnc9EAD/view?queryScope=userFolders

Search...

LAPTOP RENTALS Total Laptops Consumer Laptop Bookings Billing Process Laptop Bookings by Consumer

Report: Total Laptops with Laptop Bookings and Consumer  
**Laptop Bookings by Consumer**

Enable Field Editing Add Chart Edit

Total Records: 7 Total Amount: ₹45,800

Types of versions	Amount	Total Laptops: Total Laptops Name	Laptop Bookings: Laptop Bookings Name	consumer: Consumer Name
High (7)	₹2,600	Dell	Dell i3 Booking1	Uma
	₹5,000	Hp	Hp Booking1	Maresh
	₹6,000	Acer	Acer Bookin g	Uma
	₹10,000	Mac	Mac User	Uma Maresh
	₹6,000	Dell	Dell Booking	Uma
	₹10,000	Dell	Dell Booking2	Maresh
	₹6,200	Hp	Hp User	Uma Maresh
Subtotal	₹45,800			
Total (7)	₹45,800			

## 10.5. Visualizing Data with Dashboards

Dashboards provide a visual representation of key metrics from multiple reports.

### 10.5.1. Creating a Dashboard Folder

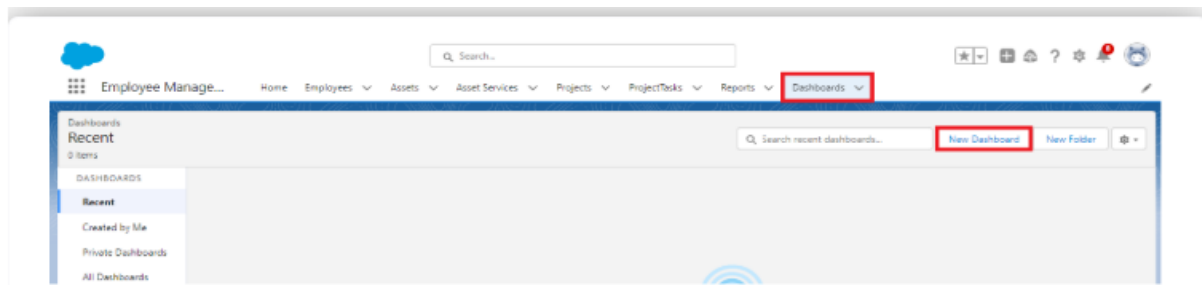
Folders are essential for organizing our reports and dashboards.

1. In the App Launcher, select the **Dashboards** tab.
2. Click **"New Folder."**
3. **Folder Label:** total rent amount

Click **"Save."**

### 10.5.2. Creating a New Dashboard

1. In the Dashboards tab, click **"New Dashboard."**
2. **Name:** data analytics of laptops
3. **Description:** total amount of data in dashboards
4. **Folder:** Click Select Folder and choose the total rent amount folder.
5. Click **"Create."**



### 10.5.3. Adding Report Components to Dashboard

We'll now add a component to our dashboard to visualize the rental amount data.

1. On the new dashboard, click the **" + Component "** button.
2. In the "Add Component" wizard, search for and select the report you just created.
3. Configure the component:
  - **Display As:** Choose a donut chart.
  - **Value:** Sum of Amount
  - **Sliced By:** types of versions
4. Click **"Add."**
5. Click **"Save"** and then **"Done"** to finalize the dashboard.

### New Dashboard

\* Name

data analytics of laptops

Description

total amount of data in dashboards

Folder

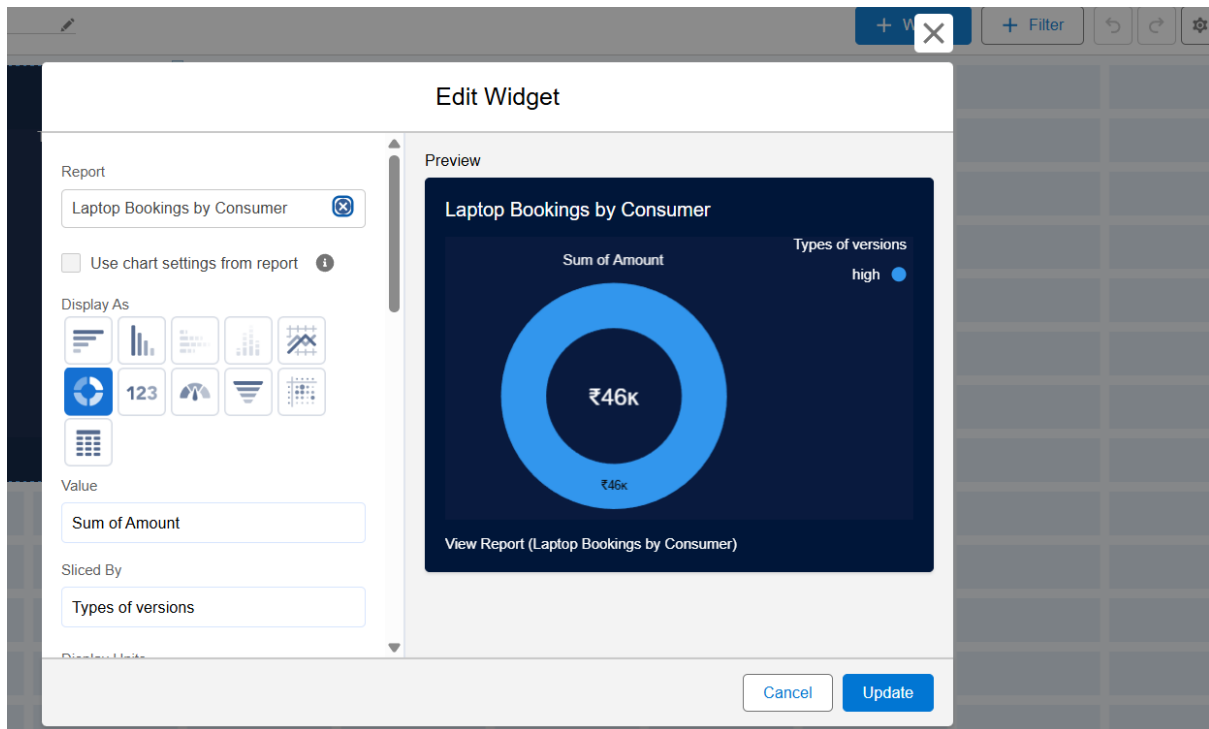
total rents amount

Select Folder

Cancel

Create





## 11. Conclusion and Future Enhancements

We have successfully built a comprehensive Laptop Rental Management System on the Salesforce platform, from the ground up.

### 11.1. Summary of Implemented Functionalities

- **Data Model:** Created custom objects for **Consumers**, **Laptop Bookings**, **Total Laptops**, and **Billing Process**, and linked them with Master-Detail and Lookup relationships.
- **Fields:** Added custom fields to each object, including picklists, email, phone, and currency fields.
- **Advanced Logic:** Implemented a field dependency, formula fields for inventory tracking, and a rollup summary field to count bookings.
- **User Experience:** Created a custom Lightning App, LAPTAP RENTALS, with dedicated tabs for easy navigation.
- **Security:** Defined custom **Owner** and **Agent** profiles and roles to manage access and data visibility.
- **Automation:** Built a powerful **Salesforce Flow** to automatically calculate rental prices and an **Apex Trigger and Handler** to send automated email notifications to customers.

- **Analytics:** Created detailed reports with bucket fields to categorize data, set up automated report subscriptions for the owner, and built a dashboard to visualize key business metrics.

### 11.2. Benefits of the Salesforce Laptop Rental System

This system provides a modern, cloud-based solution that replaces manual processes with automation. It ensures data accuracy, improves operational efficiency, and provides invaluable insights that were previously unavailable. The streamlined booking process, automated communication, and real-time inventory tracking all contribute to a better experience for both the rental business and its customers.

### 11.3. Potential Future Enhancements

While the current system is robust, it can be expanded to include more advanced features, such as:

- **Integrated Inventory Management:** Track individual laptop assets with their serial numbers.
- **Customer Portal:** Allow customers to log in, view their bookings, and make payments online.
- **Payment Gateway Integration:** Connect with payment processors like Stripe or PayPal to handle transactions directly within Salesforce.
- **Service Cloud Integration:** Use a case management system to track laptop repairs or customer support inquiries.
- **More Advanced Flow Logic:** Create flows to automatically generate invoices or contracts.

## CONCLUSION

This project is a testament to the power and flexibility of the Salesforce platform. It demonstrates how a custom business application can be built efficiently to meet specific business needs, providing a solid foundation for future growth and innovation.