Creating an Application in SalesForce.com using Apex programming Language.

<https://developer.salesforce.com/signup>

**username :-**

[bhosaleprasanna0924@agentforce.com](mailto:bhosaleprasanna0924@agentforce.com)

**Program:-**

*// Apex Class to represent a Student*

public class stud {

public String name { get; set; }

public Integer age { get; set; }

public String email { get; set; }

public String course { get; set; }

*// Constructor*

public Stud(String name, Integer age, String email, String course) {

this.name = name;

this.age = age;

this.email = email;

this.course = course;

}

*// Method to return student info*

public String getStudentInfo() {

return 'Name: ' + name + ', Age: ' + age + ', Email: ' + email + ', Course: ' + course;

}

}

**Debug:-**

Stud s = new Stud('Aniket jagtap', 21, 'aniket@example.com', 'Computer Science');

System.debug(s.getStudentInfo());

**Explaination:-**

Sure! Here's a clear explanation of **Salesforce** and the **Apex programming language**:

**🔷 What is Salesforce?**

**Salesforce** is a **cloud-based Customer Relationship Management (CRM) platform** that helps businesses manage their relationships with customers, streamline processes, and improve profitability. It offers tools for:

* Sales and marketing automation
* Customer service and support
* Analytics and reporting
* App development (via the Salesforce Platform)

Salesforce is known for being:

* **Cloud-based** (accessible from anywhere)
* **Highly customizable** (using point-and-click tools or code)
* **Scalable** (suitable for small to enterprise-level businesses)

It also includes features like **automation (Flow, Process Builder)**, **AI tools (Einstein AI)**, and **third-party integrations**.

**🔷 What is Apex?**

**Apex** is the **programming language used in Salesforce** to build custom business logic, automate processes, and integrate with external systems.

**Key Features of Apex:**

* **Strongly typed** and **object-oriented**
* **Java-like syntax** (easy for Java developers to learn)
* Runs on the **Lightning Platform** (Salesforce's cloud runtime)
* Used in **triggers**, **classes**, **schedulers**, **web services**, and **unit tests**

**Common Use Cases:**

* Writing **Triggers** that respond to database events (e.g., after inserting a record)
* Creating custom **Apex Classes** for business logic
* Making **HTTP callouts** to external services
* Writing **batch jobs** for large data processing
* Building **REST/SOAP APIs**

**Example Apex Code:**

public class HelloWorld {

public static void sayHello() {

System.debug('Hello from Apex!');

}

}

**Summary:**

| **Feature** | **Salesforce** | **Apex** |
| --- | --- | --- |
| Purpose | Cloud-based CRM & app platform | Custom logic within Salesforce |
| Type | No-code/low-code + code platform | Programming language (like Java) |
| Usage | Manage customers, sales, service | Automate tasks, customize behavior |
| Executes | In the cloud (multi-tenant) | On Salesforce servers |

Case study on Microsoft azure to learn about Microsoft Azure is a cloud computing platform and infrastructure, created by Microsoft, for building, deploying and managing applications and services through a global network of Microsoft-managed data centers.

OR

Case study on Amazon EC2 and learn about Amazon EC2 web services.

Here’s a **case study format** for both Microsoft Azure and Amazon EC2 to help you learn and present them effectively. You can choose based on your interest—Azure for a broader cloud ecosystem or EC2 for compute-centric IaaS services.

**📘 Case Study: Microsoft Azure – Cloud Computing Platform**

**1. Introduction**

Microsoft Azure is a cloud computing platform and infrastructure launched by Microsoft in 2010. It offers a wide range of services including IaaS, PaaS, and SaaS and supports multiple programming languages and frameworks.

**2. Key Features**

* **Virtual Machines (VMs)**: Windows & Linux support
* **Azure App Services**: Host web apps without managing infrastructure
* **Azure SQL Database**: Managed relational SQL database service
* **Azure DevOps**: CI/CD tools
* **Azure AI & ML**: Pre-trained and customizable models
* **Global Reach**: 60+ Azure regions

**3. Use Case Example**

**Company**: Maersk – Global shipping company  
**Problem**: Needed to modernize legacy applications and enable data analytics.  
**Solution**: Migrated to Azure using Azure Kubernetes Service (AKS), Azure Synapse, and Logic Apps.  
**Outcome**: Improved operational efficiency, reduced IT costs by 30%, and enabled real-time shipping analytics.

**4. Benefits**

* Scalability
* Pay-as-you-go pricing
* Enterprise-grade security (ISO, GDPR, etc.)
* Integrated development tools (Visual Studio, GitHub, etc.)

**5. Challenges**

* Initial learning curve
* Complex pricing structure
* Latency in specific regions

**📘 Case Study: Amazon EC2 – Elastic Compute Cloud**

**1. Introduction**

Amazon EC2 (launched in 2006) is a part of AWS and provides scalable virtual servers in the cloud. It is one of the core components of AWS’s IaaS offerings.

**2. Key Features**

* **Elasticity**: Start/stop/scale instances as needed
* **Variety of Instance Types**: General purpose, compute-optimized, GPU-based, etc.
* **AMIs (Amazon Machine Images)**: Pre-configured OS and app environments
* **Auto Scaling & Load Balancing**
* **Integration**: Works seamlessly with S3, RDS, Lambda, etc.

**3. Use Case Example**

**Company**: Netflix – Streaming service giant  
**Problem**: Required scalable and highly available infrastructure for global video streaming.  
**Solution**: Migrated infrastructure to EC2 with auto-scaling, ELB, and custom AMIs.  
**Outcome**: Delivered uninterrupted streaming to millions of users globally with 99.99% uptime.

**4. Benefits**

* Highly customizable compute power
* Global availability
* Cost efficiency with Spot and Reserved Instances
* Tight integration with AWS services

**5. Challenges**

* Complex billing
* Requires manual OS and patch management unless automated
* Learning curve for managing at scale

**✅ Recommendation**

* Choose **Microsoft Azure** if you're interested in **end-to-end enterprise cloud ecosystems**, developer tools, and hybrid cloud capabilities.
* Choose **Amazon EC2** if you want to focus on **core computing services**, elasticity, and large-scale deployment scenarios like streaming or HPC.