* **Workings of Google drive to make spread sheet and notes. (slip 1,9,10,11,22)**

**Creating a Spreadsheet**

1. **Open Google Drive**  
   Go to [Google Drive](https://drive.google.com).
2. **Click "New"**  
   In the top left corner, click the **"New"** button.
3. **Select "Google Sheets"**  
   From the dropdown menu, select **"Google Sheets"** to open a new spreadsheet.
4. **Name and Customize**  
   After Google Sheets opens, click on **"Untitled spreadsheet"** to give it a name. You can start entering data, use formulas, format cells, and more.

**Creating Notes (Google Docs)**

1. **Open Google Drive**  
   As above, go to [Google Drive](https://drive.google.com).
2. **Click "New"**  
   In the top left corner, click the **"New"** button.
3. **Select "Google Docs"**  
   From the dropdown, choose **"Google Docs"** to create a new document for your notes.
4. **Name and Start Typing**  
   When Google Docs opens, click **"Untitled document"** to rename it. Now you can start typing your notes and add formatting as needed.

These Google apps also automatically save your work, so you don’t have to worry about losing progress!

* **Create and host static web page using any cloud provider(slip 1, 16, 22)**

**1. Create an AWS Account**

* Go to [AWS](https://aws.amazon.com/) and sign up for a free-tier account if you haven’t already.

**2. Prepare Your Web Files**

* Develop your HTML, CSS, JavaScript, and any other static files for the website.
* Save these files in a single folder on your computer.

**3. Upload Files to Amazon S3 (Simple Storage Service)**

1. **Open the S3 Console**
   * Log in to AWS, navigate to the **S3** service in the AWS Management Console.
2. **Create a Bucket**
   * Click **Create Bucket** and give it a unique name (e.g., my-website-bucket).
   * Choose a region and keep it as **Public Access** (important for website hosting). AWS will warn you about making it public; confirm the setting.
3. **Upload Your Files**
   * Open your newly created bucket and click **Upload**.
   * Select your HTML, CSS, and JavaScript files and upload them.

**4. Configure the Bucket for Static Website Hosting**

1. **Enable Static Website Hosting**
   * In the bucket, go to **Properties**, scroll down, and choose **Static website hosting**.
   * Select **Enable**, and under **Index document**, type your main file (usually index.html).

* **Practical Implementation of Storage as a Service (using Own cloud).(slip 2, 17)**
* Practical Implementation of cloud security.(slip 2 ,18)
  + Open AWS Console
  + Click IAM
    - Create Users/Groups
    - Enable MFA for Users
    - Set Permissions (Least Privilege)
  + Click VPC
    - Set Up Custom VPC with Subnets
    - Configure Security Groups and ACLs
    - Enable VPC Flow Logs
  + Click S3 or RDS
    - Enable Encryption (at Rest)
    - Enable HTTPS (Data in Transit)
  + Click CloudTrail
    - Enable for Logging API Calls
  + Click CloudWatch
    - Set Alarms for Unusual Activity
  + Click GuardDuty
    - Enable Threat Detection
  + Click WAF & Shield
    - Set Up WAF for App Protection
    - Enable Shield for DDoS Protection
  + Click Backup
    - Enable Automated Database Backups
  + Click Security Hub
  + Run Compliance and Security Checks
* **Workings of Google drive to create exam form and generate result(slip 3,8,10,21)**

**1. Create the Exam Form with Google Forms**

* **Open Google Drive**
  + Go to [drive.google.com](http://drive.google.com/) and sign in.
* **Click New > Google Forms**
  + Select “Blank Form” to create a new form.
* **Set Up the Exam Questions**
  + Add fields for the student’s **name**, **email**, and **ID number**.
  + Use **Multiple Choice**, **Short Answer**, or **Paragraph** question types as needed.
  + Assign point values to each question under **Answer Key** if it’s a quiz form.
* **Enable Quiz Mode**
  + Go to **Settings > Quizzes** and toggle on **Make this a quiz** to allow automatic grading.
* **Distribute the Form**
  + Click **Send** to share the form link with students.

**2. Collect Responses in Google Sheets**

* **Link to Google Sheets**
  + In Google Forms, click **Responses > Create Spreadsheet** to generate a Google Sheets file where all responses will be collected.
* **View Collected Data**
  + Each response, including the score if it’s a quiz, will automatically populate in this linked Google Sheet.

**3. Generate Results in Google Sheets**

* **Calculate Total Scores (if needed)**
  + Use formulas like =SUM() to calculate total scores if they weren’t auto-scored in the form.
* **Analyze and Sort Scores**
  + Sort or filter by score, student name, or other details for easy result organization.
* **Visualize Data**
  + Create charts or graphs (e.g., grade distribution) under **Insert > Chart**.
* **Share Results with Students**
  + You can share results by creating a separate “Results” sheet and sending students individual links or by emailing them.
* **Practical Implementation of File sharing and Storage as a Service(slip 3,7,15,19,27)**
* **Practical Implementation of Storage as a Service (using Drive).(slip 11, 14)**

**1. Set Up Google Drive for File Storage**

* **Create a Google Account**
  + Go to [accounts.google.com](http://accounts.google.com/) if you don’t already have an account.
* **Access Google Drive**
  + Visit [drive.google.com](http://drive.google.com/) and log in to access your Drive.
* **Organize Files in Folders**
  + Create folders for different purposes (e.g., “Projects,” “Documents,” “Images”) to keep files organized.

**2. Upload and Manage Files**

* **Upload Files and Folders**
  + Click **New > File upload** or **Folder upload** to add files to Google Drive.
  + Use **Drag and Drop** directly into the Drive window to upload multiple files quickly.
* **Set File Permissions**
  + Right-click a file or folder, select **Share**, and set access as **Viewer**, **Commenter**, or **Editor** to control who can view, comment, or edit.

**3. Enable File Sharing and Collaboration**

* **Share Files via Link**
  + Right-click a file, choose **Get link**, and set the link’s permission to **Anyone with the link** to share it with a broad audience.
* **Collaborate in Real-Time**
  + Google Drive apps (Docs, Sheets, Slides) allow real-time collaboration. Open a file, click **Share**, add collaborators by email, and set permissions.
  + Collaborators can leave comments, make edits, or chat within the document.
* **Working and Implementation of Software as a service (Google).(slip 4)**
* **Create Google form for generating Certificates for Workshop(slip 4 ,6 ,20, 25)**

**1. Create Google Form**

* Open Google Drive > **New** > **Google Form**.
* Add fields: **Name**, **Email**, **Workshop Attended**, **Completion Date**.

**2. Install Certify'em**

* Go to **Add-ons** > **Get add-ons**, search for **Certify'em** and install it.

**3. Configure Certify'em**

* Open the form > **Add-ons** > **Certify'em** > **Create Certificates**.
* Select a template, map form fields (e.g., {{Name}}), and customize certificate.

**4. Email Certificates**

* Set up **automatic email delivery** to send certificates to participant emails.

**5. Share Form**

* Share the form link with participants. Certify'em will generate and email certificates automatically upon form submission.
* Working and Implementation of Infrastructure as a services.(slip 5 ,24)
* Demonstrate how to managing cloud computing Resources.(slip 12)

**1. Set Up an AWS Account**

* Go to [AWS](https://aws.amazon.com/) and sign up for an account if you don’t already have one.

**2. Launch an EC2 Instance (Virtual Server)**

* **Log in to AWS Console**.
* Navigate to the **EC2** service.
* Click **Launch Instance**.
* Choose an **Amazon Machine Image (AMI)** (e.g., Ubuntu, Windows).
* Select an **Instance Type** (e.g., t2.micro for testing).
* Configure the **Instance Details** (network, IAM role, etc.).
* Add **Storage** if needed (EBS volume).
* Configure **Security Group** (allow SSH/HTTP/HTTPS).
* Click **Launch** and create a new **Key Pair** for SSH access.
* **Developing python Application (addition of 10 numbers) Using Google App Engine.(slip 5,8,12,13)**

**Step 1: Set Up GitHub**

1. **Create a GitHub account**: Go to [GitHub](https://github.com) and sign up if you don’t have an account.
2. **Create a repository**:
   * After logging in, click the **+** in the upper-right corner, select **New repository**.
   * Name the repository (e.g., python-app-engine), then click **Create repository**.

**Step 2: Set Up Your Python Application**

1. **Create a Python file** (e.g., app.py) locally with your code.
2. **Push the Python file to GitHub**:
   * Initialize the repository locally with git init.
   * Add, commit, and push your code to the repository.

**Step 3: Set Up Google Cloud**

1. **Sign in to Google Cloud Console**: Go to Google Cloud Console.
2. **Enable App Engine**:
   * In the Cloud Console, search for **App Engine** and click **Enable** to activate it for your project.

**Step 4: Clone the Repository on Google Cloud Console**

1. Open **Cloud Shell** by clicking the terminal icon in the top-right corner.
2. Clone your GitHub repository using:

bash

Copy code

git clone <your-github-repository-link>

1. If needed, delete old directories using:

bash

Copy code

rm -rf old-directory-name

* Write a program for Web feed.
* Working and Implementation of identity management. (slip 7,16,27)
* Show the implementation of cloud on single sign on.(slip 17 ,21)

**Step 1: Set Up AWS Identity and Access Management (IAM)**

1. **Log in to AWS Console**:
   * Go to the AWS Management Console and log in with your administrator account.
2. **Create IAM Users**:
   * Go to **IAM** service in the AWS Console.
   * Click on **Users** > **Add user**.
   * Set a username, select **Programmatic access** (for API access) and/or **AWS Management Console access** (for console login).
   * Assign permissions either directly or by adding the user to a group.

**Step 2: Create IAM Groups and Assign Permissions**

1. **Create IAM Groups**:
   * Go to **IAM** > **Groups** > **Create New Group**.
   * Name the group (e.g., Admins, Developers, ReadOnly).
   * Attach predefined IAM policies (e.g., AdministratorAccess, PowerUserAccess, ReadOnlyAccess) or create custom policies based on the user's needs.
2. **Assign Users to Groups**:
   * Add the appropriate IAM users to the groups based on their roles.

**Step 3: Enable Multi-Factor Authentication (MFA)**

1. **Enable MFA for IAM Users**:
   * Go to **IAM** > **Users** > Select a user > **Security Credentials**.
   * Under **Multi-Factor Authentication (MFA)**, click **Assign MFA device**.
   * Choose either a **Virtual MFA device** (using an app like Google Authenticator) or a **Hardware MFA device**.

**Step 4: Create and Attach IAM Policies**

1. **Define IAM Policies**:
   * Go to **IAM** > **Policies** > **Create Policy**.
   * Define permissions using either the **Visual editor** or **JSON**.
2. **Attach Policies to Users/Groups**:
   * Attach the appropriate policies to users or groups (either predefined or custom policies).

**Step 5: Configure Single Sign-On (SSO)**

**Enable AWS Single Sign-On (SSO)**:

* + In AWS Console, go to **AWS SSO** > **Enable**.
  + Integrate AWS SSO with your corporate identity provider (e.g., Active Directory, Okta).
  + Set up user access to various AWS applications and accounts via SSO.