Experiment 2: Cloud-based E-learning System (Detailed Step-by-Step)

AIM: To design and deploy a simple cloud-based e-learning system using GCP (Firebase Hosting, Firestore, and Cloud Storage).

DESCRIPTION: A lightweight e-learning web app that lists courses stored in Firestore. Course materials (PDFs/videos) are hosted in Cloud Storage (Firebase Storage). The frontend fetches course metadata and displays clickable links to materials.

PREREQUISITES:

• Google account with access to Firebase / Google Cloud Console
• Node.js and npm installed
• Firebase CLI installed (npm install -g firebase-tools)
• Basic HTML and JavaScript knowledge
• A GitHub repository (optional)

DETAILED PROCEDURE (STEP-BY-STEP):

Step 1 — Create Firebase Project

1. Open https://console.firebase.google.com/ and click 'Add project'.
2. Enter project name (e.g., elearning-demo) and click Continue.
3. Follow prompts and create the project. Wait while Firebase provisions resources.
4. In Project settings > General, under 'Your apps' register a Web app () and note the firebaseConfig values.

Insert screenshot for Step 1 here: ______________________________

Step 2 — Enable Firestore and Firebase Storage

1. In Firebase Console -> Build -> Firestore Database -> Create Database. Choose 'Start in test mode' for practice and select a region.
2. In Firebase Console -> Build -> Storage -> Get Started. Choose the same location or a nearby location.
3. Make a note of the Storage bucket name (usually your-project-id.appspot.com).

Insert screenshot for Step 2 (Firestore & Storage) here: ______________________________

Step 3 — Initialize Firebase locally

1. Open terminal/command prompt and install Firebase CLI if not installed:
```
npm install -g firebase-tools
```
2. Login to Firebase CLI:
```
firebase login
```
3. Create a project folder and initialize hosting:
```
mkdir elearning-demo
cd elearning-demo
firebase init hosting
```
When prompted, select your Firebase project, set 'public' as the public directory, and choose Yes for single-page app if asked.

Insert screenshot for Step 3 (firebase init) here: ______________________________

Step 4 — Upload course materials to Storage

Option A: Upload via Firebase Console:
1. Open Firebase Console -> Storage -> Files -> Upload files. Upload your PDFs/videos.
2. Click the uploaded file and copy the 'File URL' (Make sure file is readable by those with the link or configure rules).

Option B: Upload via gsutil (Cloud SDK required):
```
gsutil cp path/to/your-material.pdf gs://YOUR_BUCKET_NAME/path/to/your-material.pdf
```
Insert screenshot for Step 4 (uploaded file URL) here: ______________________________

Step 5 — Create Firestore 'courses' collection and add documents

1. In Firebase Console -> Firestore -> Start collection. Enter collection id: courses.
2. Add a document with fields:
title (string) - e.g., 'Intro to Cloud Computing'
fileUrl (string) - paste the Storage File URL from Step 4
3. Save the document. Repeat to add multiple courses.
Insert screenshot for Step 5 (courses collection entry) here: ______________________________

Step 6 — Create frontend files (index.html and script.js)

Create a file public/index.html with the following content:
```
<!DOCTYPE html>
<html>
<head>
  <title>E-learning System</title>
  <script src="https://www.gstatic.com/firebasejs/9.22.2/firebase-app.js"></script>
  <script src="https://www.gstatic.com/firebasejs/9.22.2/firebase-firestore.js"></script>
  <script src="script.js"></script>
</head>
<body>
  <h2>E-learning Courses</h2>
  <ul id="courseList"></ul>
</body>
</html>
```

Create a file public/script.js with the following content (replace firebaseConfig with your project's values):
```
const firebaseConfig = {
  apiKey: "YOUR_API_KEY",
  authDomain: "YOUR_PROJECT_ID.firebaseapp.com",
  projectId: "YOUR_PROJECT_ID",
  storageBucket: "YOUR_PROJECT_ID.appspot.com",
  messagingSenderId: "YOUR_SENDER_ID",
  appId: "YOUR_APP_ID"
};

firebase.initializeApp(firebaseConfig);
const db = firebase.firestore();

function loadCourses() {
  const list = document.getElementById('courseList');
  db.collection("courses").get().then((querySnapshot) => {
    querySnapshot.forEach((doc) => {
      const course = doc.data();
      const li = document.createElement("li");
      const link = document.createElement("a");
      link.href = course.fileUrl;
      link.innerText = course.title;
      link.target = "_blank";
      li.appendChild(link);
      list.appendChild(li);
    });
  });
}

window.onload = loadCourses;
```
Insert screenshot for Step 6 (code files in project folder) here: ______________________________

Step 7 — Deploy to Firebase Hosting

```
firebase deploy
```
2. After deployment, note the Hosting URL shown in the CLI (e.g., https://.web.app).
Insert screenshot for Step 7 (deploy output and hosting URL) here: ______________________________

Step 8 — Test the App

1. Open the Hosting URL in a web browser.

2. You should see the list of courses (titles) loaded from Firestore with links to the materials.
3. Click a course link to open the PDF/video in a new tab.
4. In Firebase Console -> Firestore -> courses, verify read counts or check that the documents are present.
5. Take screenshots of the hosted page and Firestore entries for your report.

Insert screenshot for Step 8 (hosted app and Firestore view) here: ____________________________

RESULT: The e-learning web app lists courses pulled from Firestore and links to course materials hosted in Cloud Storage. The app is accessible via the Firebase Hosting URL.

TROUBLESHOOTING & NOTES:

• If Firestore reads fail, ensure Firestore rules allow reads (for demo, test mode should allow reads).
• If storage URLs return 403, check file permission or generate a signed URL for restricted content.
• If firebase deploy fails, ensure you are logged in (firebase login) and selected the correct project during firebase init.
• To secure the app, enable Firebase Authentication and update Firestore security rules to require authentication.

END OF PROCEDURE - Insert your screenshots in the marked placeholders