Alexandria University
Faculty of Engineering
Computer and Systems Engineering Department

Database Systems
CSE-471
Lab 1

Lab 1: AI-Powered ER Diagram Generator

1 Assignment Description

1.1 Objective

This assignment is designed to deepen your understanding of Entity-Relationship (ER) modeling. You will develop a tool to automatically generate ER diagrams from a structured data format (JSON). Then, you will integrate Google's Gemini AI to convert a natural language system description into this structured JSON, creating a complete "text-to-diagram" pipeline.

1.2 Part 1: The JSON to ER Diagram Renderer

Your first task is to write a program that reads a specifically structured JSON file and generates a visual ER diagram as its output.

1.2.1 Task Description

You will implement a tool in Python that performs the following:

1. Parses a JSON file containing definitions for entities, attributes (including simple, primary key, **multivalued**, and **composite** attributes), and relationships.
2. Generates a graphical representation of the ER Diagram as an image file (e.g., `.png`).

1.2.2 The Structured JSON Format

Your program must correctly parse a JSON file with similar structure.

Listing 1: Structured JSON Schema for ER Diagrams

```
{
  "entities": [
    {
        "name": "Employee",
        "attributes": [
            { "name": "SSN", "isPrimaryKey": true },
            { "name": "Name", "composite": ["FirstName", "LastName"] },
            { "name": "Phone", "isMultiValued": true },
            { "name": "Address", "composite": ["Street", "City", "State", "ZIP"] }
        ]
    },
    {
      "name": "Department",
      "attributes": [
        { "name": "DeptID", "isPrimaryKey": true },
        { "name": "Name" },
        { "name": "Location", "isMultiValued": true }
      ]
    }
```

```
  ],
  "relationships": [
    { "entity1": "Employee", "entity2": "Department",
      "name": "Works_In", "cardinality": "N:1" }
  ]
}
```

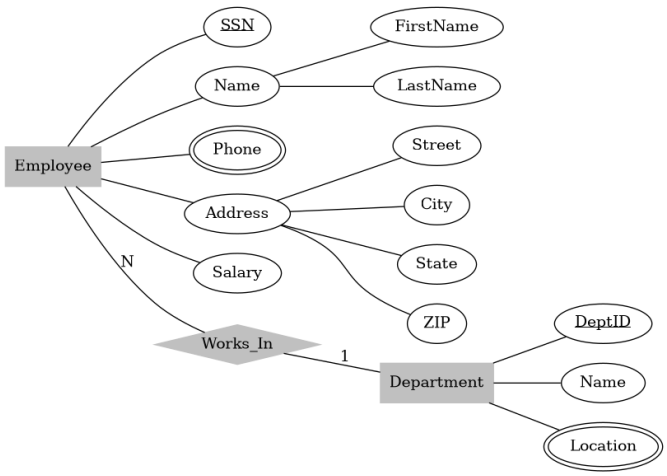

Figure 1: Json to Diagram Output Sample

1.3 Part 2: AI Integration with Gemini API

You will use the Gemini API to bridge the gap between a high-level requirement description and the structured JSON your tool from Part 1 can understand.

1.3.1 Task Description

You will write a script that takes a plain text description of a system and uses the Gemini API to automatically generate the JSON data in the **exact format specified in Part 1**.

1.3.2 Steps

1. **Get an API Key:** Obtain a free API key for the Gemini API from Google AI Studio.
2. **Prompt Engineering:** Design a high-quality prompt that instructs Gemini to analyze the text and output a JSON object that conforms to your predefined schema.
3. **API Call & Parsing:** Send the system description and your prompt to the Gemini API, receive the response, parse the JSON content, and save it.
4. **End-to-End Test:** Use the AI-generated JSON file as input for your renderer from Part 1 to create the final ER diagram.

Guidelines and Submission Details

- You should solve this assignment **individually**.
- Submit compressed file containing your python scripts and README file describing your work.
- You can use the internet or the reference to help you in answering the questions.
- **Assignment deadline is on Sunday 12th October**. Use the following form to submit your solution.