CV Analyzer - Software Engineering Project - 22916 2025A

Carefully read the instructions, as 10% of the grade is allocated to following them accurately.

Submission Guidlines

- Create a **private** GitHub repository for your project artifacts.
- When submitting, add me (omishali) as a collaborator so I can access it.
- Note that free private repos allow only three collaborators, so if your project involves more than two students, one collaborator should be removed upon submission.
- The project is due on Thursday, March 27th.
- Submit a single DOC file through the "Matalot" system, including your team names and a link to the repository.

Project Overview

Create a command-line python application that processes a CV in PDF format and performs ONE interesting analysis feature.

Evalution

The project will be evaluated based on:

1. Following the project instructions (10%)
2. Quality of conversations with the LLM - Your ability to apply prompt engineering techniques taught in class, explore alternatives, and use critical thinking to leverage the LLM's strengths while addressing its weaknesses (20%).
3. Quality of produced artifacts (70%)

Basic Requirements

- Command-line interface (no GUI)
- English language support only
- Choose either NLTK or Gemini LLM for the analysis (Gemini offers a free limited plan), and be sure to understand the differences before making your decision.
- Submit all LLM interactions that influenced your decisions.
  - Copy each LLM chat (e.g., ChatGPT, Claude) into a separate text file within the `chats/` directory of your project (using Select All and Copy-Paste). Links to Web pages are not allowed.
  - For each project phase described in the README, provide links to the corresponding LLM chats located in the `chats/` directory, along with a brief summary of the purpose or topic of each chat.

Project Phases

Each phase must be documented in the README as its own section. The documentation should be included directly in the README (marked below as "inline"), or with links to other parts of the project ("link"). Note that the project phases may not necessarily be executed in the order presented.

Phase 1: Requirements Engineering

- Consult with an LLM to define ONE significant and interesting CV analysis feature.
- Document the feature's requirements clearly (inline).
- Include acceptance criteria (inline).
- Document LLM interactions (link).

Phase 2: Architecture

- Define command-line interface specification (inline).
- Plan file system interactions, i.e., input/output (inline).
  - Your feature may use additional files for input and output.
- Identify relevant third-party libraries (inline).
- Define team member responsibilities (inline).
- Document LLM interactions (link).

Phase 3: Design

- Follow XP's principles for simple design.
- Provide a brief CRC description of the key classes (inline).
- Document LLM interactions (link).

Phase 4: Coding & Testing

Coding:
- Write clean, well-organized code
- Follow PEP 8 style guidelines
- DO NOT submit LLM interactions related to coding.
- Include a table in your README with clickable file names linking to the project and brief descriptions of their purpose (two columns: **File Name** and **Description**).

Testing:
- Implement one significant automated unit test (with multiple test cases) (inline & link).
- Create one meaningful automated system-level functional test (inline & link).
- Document LLM interactions (link).

Phase 5: Documentation

Create a concise and clear README.md with the following structure. The README should be well-formatted using the appropriate Markdown syntax.
- Project overview
- Easy-to-follow installation guide with no glitches – it should work perfectly.
- Usage examples with sample PDF input (and other files if required).
- Project phase documentation (as previously explained).