



École Nationale Supérieure
d'Informatique et d'Analyse des Systèmes
- Rabat

End of Year Project Report

# IT Certifications Pipeline

**Student :**
Zakaria Ait Hssain

**Supervisor :**
Pr. M. Lazaar

**Jury :**
Pr. M. Lazaar & Pr. R. Faizi

University Year 2024-2025

# Acknowledgments

First and foremost, I wish to express my deep gratitude to Allah, the Almighty, the source of all wisdom and strength. It is by His will and guidance that I was able to move forward, overcome obstacles, and successfully complete this project. Without His light, none of my efforts would have been possible. I offer my sincerest thanks to Him for having illuminated my path from beginning to end.

I also extend my heartfelt appreciation to my dear family. Their unwavering support and constant encouragement have been an invaluable source of motivation. Every step of this journey was made easier and more meaningful thanks to their presence and their confidence.

I would also like to warmly thank all those who contributed, directly or indirectly, to the completion of this modest work.

A special mention goes to Mr. M. Lazaar, my supervisor, for his support and availability throughout this project. And to Mr. A. El Afia for his insightful advice.

I also wish to thank Mr. R. Faizi, member of the jury, for the time he dedicated to me, as well as for his pertinent and enriching remarks.

Finally, I thank all those who directly or indirectly contributed to the completion of this report.

# Abstract

This report describes the development of a complete data pipeline aimed at collecting, processing, and analyzing information about IT certifications from three major providers: Amazon Web Services (AWS), Microsoft, and CompTIA. In today's technology-driven world, professional certifications are becoming more important for proving skills and advancing careers. However, information about these certifications is often scattered and presented in different formats, making it hard to compare and analyze. This project solves that problem by creating a system that gathers and organizes certification data into a single, consistent format.

The project includes several key steps. First, it uses automated web scraping to collect data from provider websites. Then, the data is cleaned and standardized to fix inconsistencies and prepare it for analysis. Machine learning models are used to fill in missing details such as certification cost, level, duration, and domain. This helps create a more complete and reliable dataset.

Next, the project uses exploratory data analysis (EDA) to uncover trends and insights, like which providers offer the most certifications, how much they cost, how long exams take, and what languages they're available in. These results are displayed in a user-friendly dashboard built with Streamlit, allowing users to interact with the data and explore different aspects of the certification landscape.

The final outcome includes a cleaned and enriched dataset, reusable code for scraping and analysis, intelligent prediction models, and an interactive dashboard. This report explains how the system was built, the challenges that were faced (such as messy website structures and missing data), and offers ideas for future improvements like adding more providers or connecting the data with job market trends.

# General Introduction

In today's rapidly advancing digital world, professional certifications in Information Technology (IT) play a vital role in validating technical skills. They allow professionals to demonstrate expertise in specialized domains such as cloud computing, cybersecurity, networking, and data analytics. These certifications have become powerful tools for employability, career development, and professional recognition in the tech industry.

Organizations like Amazon Web Services (AWS), Microsoft, and CompTIA each offer dozens of certifications covering a wide range of technologies and proficiency levels. However, despite their increasing importance, comparing and understanding these certifications remains a complex task. Each provider publishes its data in different formats, structures, and terminologies, making cross-comparison and large-scale analysis difficult without considerable manual effort.

**Motivation and Problem Definition:**
The core problem addressed by this project is the fragmentation of IT certification data. Currently, there is no centralized, clean, and enriched source that enables professionals, students, or employers to easily browse, compare, and analyze available certifications across providers. This lack of structured information is especially problematic for:

- Individuals seeking to select the right and optimal certification for their goals,
- HR professionals trying to recommend targeted upskilling paths,
- Educational institutions aiming to align training with current industry standards.

This project responds to that challenge by designing an automated pipeline for collecting, processing, enriching, and analyzing certification metadata from the three major providers mentioned above.

**Relevance in Artificial Intelligence and Socio-Economic Impact**
This work is highly relevant in the field of artificial intelligence (AI), particularly through its use of machine learning to predict missing data such as certification cost, level, domain, and exam duration. By leveraging supervised learning models, the project enhances incomplete datasets and improves their usefulness for downstream analysis.

From a socio-economic perspective, the project supports digital inclusion and workforce development. As technology continues to reshape the labor market, certifications act as a bridge between learning and employment, especially for those without formal degrees. By making certification data more accessible, interpretable, and actionable, this project empowers individuals to make informed decisions and helps organizations align talent with business needs.

**Report Structure**
This report is structured as follows:

1. Chapter 1 presents the context, objectives, problem statement, and overall methodology.
2. Chapter 2 reviews the current literature and existing solutions in certification analysis.
3. Chapter 3 describes the data collection, cleaning, and machine learning enrichment processes.
4. Chapter 4 presents the results of exploratory data analysis (EDA).
5. Chapter 5 discusses the Streamlit dashboard implementation, challenges faced, and future directions.

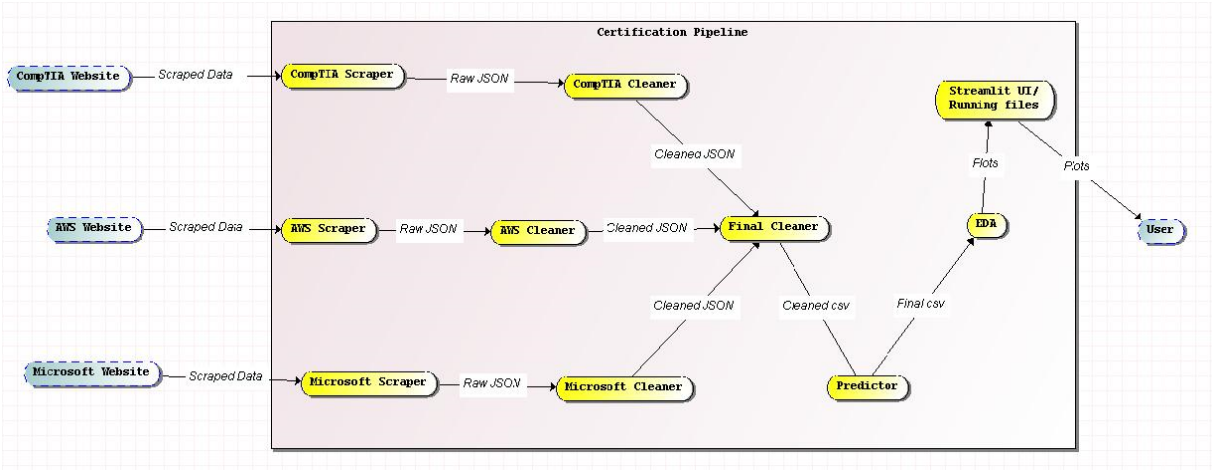


Figure 1: One Sense Communication Conceptual Model of the Pipeline

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Context and Methodology

## 1.1 Introduction

In today's digital economy, the demand for skilled IT professionals continues to grow at an unprecedented pace. As businesses accelerate their digital transformation strategies, there is increasing pressure on individuals to acquire and validate technical expertise in areas such as cloud computing, cybersecurity, networking, and data analysis. One of the most recognized ways to demonstrate such expertise is through industry certifications provided by well-established technology vendors.

Professional certifications have emerged as a global standard for skill validation. They offer structured, vendor-approved learning paths and assessments that are widely recognized by employers and educational institutions alike. Among the most prominent certification providers are Amazon Web Services (AWS), Microsoft, and CompTIA, each offering a comprehensive catalog of certifications tailored to various roles, domains, and experience levels.

These certifications are used not only by individuals seeking to enhance their career prospects, but also by organizations aiming to upskill their workforce and stay competitive in an evolving technological landscape. However, the abundance and variety of certifications available, each with its own structure, pricing, prerequisites, and targeted audience, can make it difficult for learners and employers to make informed decisions.

In this context, the present project focuses on the centralization, enrichment, and analysis of IT certification metadata from the three providers mentioned above. By collecting and standardizing key information such as certification titles, domains, levels, durations, costs, and available languages, the project seeks to offer a clearer and more comprehensive view of the IT certification ecosystem.

This introduction sets the stage for the rest of the chapter, which will define the problem in more detail, explain the motivation for the project, discuss its relevance in the field of artificial intelligence, outline its socio-economic impact, and present the overall methodology followed throughout its development.

## 1.2 Problem Statement and Motivation

**Problem Statement:**
Despite the growing significance of IT certifications in career advancement and workforce development, accessing, comparing, and analyzing certification offerings remains a challenging task. Each provider (AWS, Microsoft, and CompTIA) publishes its certification data independently, using unique formats, structures, and terminologies. This lack of standardization results in fragmented information that is difficult to process or compare at scale. For instance:

- Some providers present certifications as detailed web pages with embedded JavaScript, while others rely on static HTML layouts, meanwhile others combine both.
- The terminology used for levels (e.g., "Foundational" vs. "Beginner") and domains (e.g., "Cloud" vs. "Infrastructure") varies widely.
- Cost and duration details are inconsistently available, differently formatted, or missing altogether.

These inconsistencies pose a significant problem for:

- Professionals trying to choose the most relevant certification for their career path.
- Employers attempting to assess certification value and align training with organizational goals.
- Educators and institutions designing curricula that reflect current industry standards.

Furthermore, manual comparison of certifications across providers is time-consuming and error-prone, especially when key fields like cost, duration, or or level are missing or unclear. Without a unified and enriched dataset, making informed decisions becomes difficult for all stakeholders.

**Motivation:**
The motivation for this project arises from the need to:

- Simplify access to certification information by centralizing it.
- Standardize and clean the data for consistency across providers.
- Enrich the dataset using machine learning to fill in missing attributes.
- Provide insights through visual analytics that can help individuals and organizations make informed, data-driven decisions about the certifications choice.

By automating the end-to-end pipeline, from web scraping to machine learning and dashboarding, this project not only addresses a real-world problem but also showcases the practical application of data science and artificial intelligence techniques in an educational and professional development context.

## 1.3 Relevance to Artificial Intelligence and Socio-Economic Impact

This project is not only a technical solution to a data management problem, it also highlights the powerful role that artificial intelligence (AI) can play in enhancing and interpreting incomplete, real-world datasets.

**Relevence to AI:**
In this project, AI is applied in the form of supervised machine learning algorithms to predict missing or incomplete metadata fields in the certification dataset, such as cost, exam duration, domain, and level. These predictions are based on existing patterns in the data and are generated using models such as decision trees and random forests, which are chosen for their interpretability and effectiveness on small to medium-sized datasets.

Moreover, feature selection techniques are used to identify the most informative attributes for each prediction task, improving model performance and transparency. This demonstrates how AI can be used to enhance the quality of real-world data, particularly when raw information is inconsistent or partially missing; a common challenge in many applied domains.

The project also exemplifies a human-centered AI approach, where the goal is not only automation but also human service: helping users (professionals, students, HR teams) make smarter, faster, and more informed decisions by enriching and visualizing data in meaningful ways.

**Socio-economic Impact:**
The socio-economic implications of this work are far-reaching. As industries around the world accelerate their digital transformation, the skills gap in IT continues to grow. Certifications serve as a practical and accessible path to upskilling, especially for those who do not have the opportunity to pursue formal higher education.

By providing a centralized, clean, and comparable repository of certification data, this project supports:

- Job seekers in identifying suitable certifications that align with their goals and backgrounds.
- Employers in making evidence-based decisions on employee training and recruitment.
- Educational institutions in aligning curricula with industry demands.

Additionally, the use of AI to equalize access to information supports broader goals such as digital inclusion, lifelong learning, and economic mobility. By lowering the barrier to understanding and comparing the certification landscape, this project contributes to a more informed and resilient digital workforce.

## 1.4 Project Objectives

The main objective of this project is to design and implement a complete, automated pipeline that enables the collection, enrichment, and analysis of IT certification data from leading providers. This pipeline should address the challenges of data fragmentation, missing information, and lack of comparability by delivering a unified, clean, and intelligent dataset that is easy to explore and interpret.

To achieve this, the project is structured around the following specific objectives:

**Functional Objectives:**

1. Automate the acquisition of certification metadata:

- Develop web scrapers using static (requests + BeautifulSoup) and dynamic (Selenium) tools to extract certification information from AWS, Microsoft, and CompTIA websites.

2. Clean and standardize the raw data:
   - Normalize field names, formats, and units (e.g., cost in USD, duration in minutes).
   - Remove duplicates, irrelevant entries, and empty fields.
   - Merge data from all providers into a single consistent schema.
3. Predict and complete missing fields using machine learning:
   - Apply supervised learning models to infer unknown values in key fields like: Certification Level, Domain of specialization, Exam Cost, and Exam Duration.
4. Generate insights using Exploratory Data Analysis (EDA):
   - Use Python libraries such as Seaborn and Matplotlib to visualize patterns, distributions, and trends in the enriched dataset.
5. Build an interactive dashboard for exploration:
   - Implement a web-based interface using Streamlit to allow users to run the full pipeline, view real-time results, and interact with visual reports.

**Technical Objectives:**

- Design a modular, reusable Python codebase for scraping, cleaning, and prediction (via OOP).
- Ensure integration and compatibility between different components of the pipeline.
- Enable future extensibility (e.g., support for additional providers, multilingual scraping).

By fulfilling these objectives, the project aims to deliver a valuable tool that can assist individuals, educators, and organizations in navigating the IT certification landscape with greater clarity and confidence.

## 1.5 Methodology Overview

To address the problem of fragmented and inconsistent IT certification data, this project adopts a structured and modular methodology that follows a complete data science lifecycle. The pipeline consists of four main phases: data collection, data cleaning, machine learning enrichment, and exploratory data analysis (EDA). Each phase is designed to be reusable, extensible, and automated for seamless integration into a user-friendly system.

**1. Data Collection (Web Scraping):**
The first phase involves extracting certification metadata from the official websites of AWS, Microsoft, and CompTIA. Two types of scraping techniques are used:

- Static scraping with requests and BeautifulSoup, for pages with server-rendered HTML structures (e.g., CompTIA and certifications overview page of AWS).
- Dynamic scraping using Selenium, to handle websites that rely heavily on JavaScript rendering or contain interactive elements (e.g., AWS and Microsoft).

Each scraper is designed as a Python class and writes its output in JSON format for easy integration.

**2. Data Cleaning and Standardization:**
The collected data from different providers varies significantly in terms of format, field names, and content. A set of cleaning python classes standardizes each raw data by:

- Normalizing column names, currency formats, time units, languages representation, levels, and domains.
- Removing duplicates and invalid records (e.g., records with missing names)
- Resolving semantic inconsistencies (e.g., merging similar domain names)
- Assigning a Provider label to each entry

All cleaned data is then merged into a unified dataset, to which additional cleaning steps are applied.

**3. Machine Learning for Data Enrichment:**
A major challenge is the presence of missing values in fields like Domain, Level, Cost, and Exam Duration. These gaps are addressed using supervised learning:

- Classification models (e.g., Decision Tree Classifier) are used to predict categorical fields like Domain and Level.
- Regression models (e.g., Decision Tree Regressor, Random Forest Regressor) are used to predict numeric fields such as Cost (USD) and Exam Duration (min).
- Feature selection is performed using SequentialFeatureSelector to identify the most relevant input variables for each prediction task. Models are trained and validated using cross-validation techniques to ensure accuracy and generalization.

**4. Exploratory Data Analysis (EDA):**
Once the dataset is complete and enriched, visual exploration is performed using Python libraries such as Seaborn and Matplotlib. Key analyses include:

- Distribution of costs and durations.
- Certification count by level and domain.
- Average cost and duration by provider.
- Most common domains in certifications.
- Most common languages in certifications.

These analyses help reveal trends, anomalies, and actionable insights in the data.

**-Streamlit-Based Dashboard:**
The final phase is the implementation of a user interface using Streamlit, a lightweight Python framework for building data apps. The dashboard allows users to:

- Execute each phase of the pipeline (scraping, cleaning, ML, EDA).
- Monitor progress and view real-time logs.
- Interact with dynamic charts and tables
- Download processed datasets and charts.

This complete methodology transforms scattered certification data into a powerful decision-support tool for learners, educators, and workforce planners.

## 1.6 Scope and Limitations

**Scope of the Project:**
This project focuses on building a complete pipeline to centralize, clean, enrich, and analyze IT certification metadata from three major providers:

- Microsoft
- CompTIA
- Amazon Web Services (AWS)

The final dataset includes 104 unique certifications and covers a variety of important attributes, such as:

- Certification title
- Provider
- Domain
- Level (e.g., beginner, intermediate, advanced)
- Languages offered
- Cost in USD
- Exam duration (in minutes)
- Recommended experience
- Official URL
- Description

The dataset captures a balanced range of certification types, with most entries classified at the intermediate level (65 certifications), followed by advanced (19) and beginner (18). A small number (2) are labeled as specialized. Common domains include productivity, business apps, cloud, and automation. The five most supported languages are English, Japanese, Spanish, German, and French.

The system built around this dataset includes:

- Web scrapers for each provider.
- Data cleaning and standardization routines for each provider and for the final raw dataset.
- Machine learning models for enriching missing attributes.
- Visual exploration of trends through EDA.
- A Streamlit dashboard for interaction.

**Limitations:**
While the final dataset is structured and comprehensive, several limitations exist:

- Provider coverage is limited to only three organizations. Though they are major players, many other certification bodies (e.g., Cisco, Oracle, Red Hat) are excluded.
- The dataset is static. It reflects the state of certification offerings at the time of scraping. Though we can re-scrape at anytime, changes on provider websites are not monitored in real time.
- Missing values remain in certain non-essential fields. For example, Recommended Experience is still missing in 13 entries, likely due to its absence in the source websites.
- Certain non-essential fields remain non-standardized due to their nature (e.g., paragraphs).
- Data accuracy relies on website structure stability. Some providers (like AWS) frequently change layout or content, which may break the scrapers and require manual maintenance.
- Prediction accuracy might be low for some targets due to the insufficient number of training records.
- No linkage to job market demand or salary insights. While the dataset provides rich certification metadata, it does not yet connect to labor market data (e.g., job openings, industry trends).

# Chapter 2

# Background and Literature Review

## 2.1 Importance of IT Certifications

In the digital age, IT certifications are essential for demonstrating technical skills and professional credibility. Unlike traditional academic degrees, certifications are often more specialized, focused on current industry technologies, and regularly updated to match market needs. They are valued by employers as reliable indicators of a candidate's ability to work with specific platforms, tools, or methodologies.

According to global industry surveys, certified professionals tend to earn higher salaries and experience faster career advancement compared to non-certified peers. Certifications are also frequently used by companies to benchmark internal skills and structure employee training programs.

## 2.2 Gaps in Certification Data Aggregation

Despite their value, certification offerings are distributed across a wide range of providers, each presenting information in inconsistent formats. This lack of centralization introduces several problems:

- Difficulty in comparing certifications across domains and providers.
- Lack of standardized naming and categorization.
- Incomplete or missing metadata (e.g., cost, language, duration).
- No unified access point for learners, educators, or recruiters.

Most academic and technical work in this area focuses on specific providers or platforms, or explores certifications in the context of curriculum mapping or labor market analysis. Few, if any, consolidate cross-provider metadata into a clean, enriched dataset that supports visual and analytical exploration.

## 2.3 Related Work

Prior research efforts relevant to this project include:

- Educational data mining and course recommendation systems.
- Certification relevance ranking using job market data.

However, the integration of web scraping, automated preprocessing, machine learning, and dashboard visualization into a single, user-oriented pipeline is a novel contribution of this project.

# Chapter 3

# Data Collection, Cleaning, and Machine Learning Enrichment

## 3.1 Overview of the Pipeline Architecture

This chapter presents the technical foundation of the IT certifications pipeline, describing how raw data is collected, cleaned, enriched using machine learning, and transformed into a structured dataset suitable for analysis and visualization.

The architecture follows a modular and sequential design, where each component of the pipeline is responsible for a distinct phase of data processing. This modularity ensures code reusability, easier debugging, and scalability for future enhancements such as the integration of additional providers or real-time updates.

### 3.1.1 General Flow of the Pipeline

The overall workflow is divided into five key stages:

1. Data Collection (Scraping): Certification metadata is scraped from the official websites of AWS, Microsoft, and CompTIA using provider-specific scrapers. The extracted data is saved in JSON format for downstream processing.

2. Data Cleaning and Standardization: Each dataset is processed using provider-specific cleaning routines to normalize field names, remove duplicates, convert units (e.g., cost and duration), and unify categorical values like "Level" and "Domain."

3. Merging and Structuring: Cleaned datasets from all providers are merged into a single DataFrame with a unified schema, forming the "pre-predictions" dataset.

4. Machine Learning Enrichment: Supervised learning models are used to predict missing values in key fields: Level, Domain, Cost (USD), and Exam Duration (min). The predicted values are inserted back into the dataset to create the final enriched version.

5. Visualization and Export: The final dataset is used for exploratory data analysis (EDA) and presented through an interactive Streamlit dashboard. The enriched dataset is also exported as a CSV file for external use.

This pipeline allows for both step-by-step execution (manual control through a user interface) and batch execution (via script-based automation).

### 3.1.2 Tools and Libraries Used

The following Python libraries and tools are used across the different stages of the pipeline:

**Web Scraping**


Figure 3.1: requests

- *requests*: Requests is a simple and elegant HTTP library for Python, designed to make sending HTTP requests more human-friendly and readable. It abstracts much of the complexity involved in using lower-level libraries like urllib, allowing developers to send HTTP requests with minimal code. With support for methods such as GET, POST, PUT, DELETE, and more. Its tagline, "HTTP for Humans," reflects its goal of providing a clean, intuitive API for working with web services and RESTful APIs in Python applications. I used it in my project to send HTTP requests for static pages (e.g., CompTIA) to get the HTML source code.


Figure 3.2: BeautifulSoup

- *BeautifulSoup:* BeautifulSoup is a Python library used for parsing HTML and XML documents. It creates a parse tree from page content, making it easy to extract data, navigate the document structure, and modify tags or attributes. Commonly used in web scraping, it works well with parsers like Python's built-in html.parser or external ones like lxml. BeautifulSoup is known for its simplicity and human-friendly API, allowing you to locate elements using tag names, classes, IDs, and CSS selectors with minimal code. In my project, It was used to extract information from static web pages, requested using requests, or from dynamic ones combined with Selenium.
- *Selenium:* Selenium is a powerful open-source tool for automating web browsers. It allows you to simulate user interactions such as clicking buttons, filling out forms,

Figure 3.3: Selenium

scrolling, and navigating between pages—just like a human would. Originally designed for automated testing of web applications, Selenium is also widely used in web scraping, especially for dynamic websites that rely heavily on JavaScript to load content. With Selenium, you can control real browsers like Chrome, Firefox, or Edge through their drivers. It provides APIs for multiple programming languages, including Python, Java, and C#. In Python, Selenium is often used in combination with libraries like BeautifulSoup or Pandas to extract and process data from fully rendered pages. It was used in this project to scrape JavaScript-rendered pages (e.g., AWS, Microsoft).

Figure 3.4: tqdm

- *tqdm*: tqdm is a Python library that provides fast, extensible progress bars for loops and iterable processing. Its name means "progress" in Arabic, reflecting its purpose. With minimal code changes, tqdm visually tracks progress, iteration speed, and estimated time of completion in the terminal, scripts, or Jupyter notebooks. It supports nested loops, pandas operations, file downloads, and more, making it a lightweight and powerful tool for monitoring long-running tasks.

### Data Handling

Figure 3.5: Pandas

- *Pandas:* Pandas is a powerful open-source Python library used for data manipulation and analysis. It provides flexible data structures, primarily DataFrame and Series, that make it easy to clean, filter, reshape, aggregate, and visualize tabular data. Built on top of NumPy, pandas supports fast, efficient operations on large datasets and integrates well with tools like Matplotlib, seaborn, and scikit-learn. It's widely used in data science, machine learning, finance, and academic research for tasks ranging from simple data cleaning to complex time series analysis. Throughout my project, I used it for data manipulation, cleaning, and transformation.

**Machine Learning**

*scikit-learn:* Scikit-learn (often imported as sklearn) is a popular open-source Python


Figure 3.6: scikit-learn

library for machine learning. Built on top of NumPy, SciPy, and matplotlib, it provides simple and efficient tools for data mining, data analysis, and modeling. Scikit-learn is known for its clean, consistent API and ease of use, making it ideal for beginners and professionals alike. It's widely used in academia and industry for prototyping and building machine learning workflows. In this IT certifications Pipeline, I used it in:

- Training classification and regression models (e.g., Decision Trees, Random Forests).
- SequentialFeatureSelector: For automated feature selection to improve model performance.
- KFold and StratifiedKFold: Cross-validation techniques to ensure model robustness.

**Visualization**

- *Matplotlib:* Matplotlib is a widely-used Python library for creating static, animated, and interactive plots. It provides full control over plot appearance, making it ideal

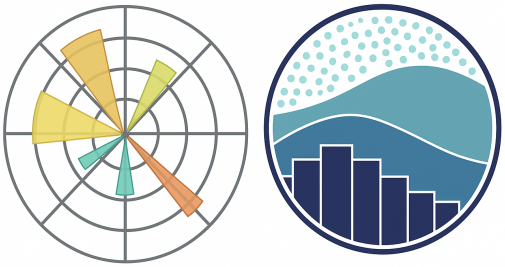

Figure 3.7: Matplotlib and Seaborn

for custom visualizations. With its core interface and its pyplot module (similar to MATLAB), Matplotlib supports a wide range of chart types, including line plots, bar charts, histograms, scatter plots, and more. It's highly flexible but can be verbose for complex plots.

*Seaborn:* Seaborn is a high-level data visualization library built on top of Matplotlib. It simplifies the creation of statistically informed visualizations, such as box plots, violin plots, and heatmaps. Seaborn integrates closely with pandas DataFrames and handles themes, color palettes, and plot aesthetics automatically, producing attractive and informative plots with minimal code. They were used for creating, saving or showing plots and charts in the Streamlit interface during exploratory data analysis.

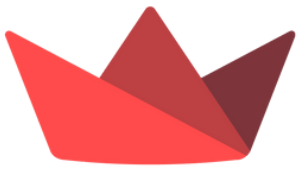

Figure 3.8: Streamlit

- *Streamlit:* Streamlit is an open-source Python library that enables users to create interactive web applications for data science and machine learning with minimal effort. Designed for simplicity and speed, Streamlit allows developers to turn Python scripts into shareable dashboards using intuitive commands. It integrates seamlessly with libraries like pandas, NumPy, and matplotlib, making it ideal for prototyping, visualizing data, and deploying models, all without requiring any front-end web development skills. I used it as a Web-based UI framework to create the interactive dashboard.

## 3.2 Data Collection (Scraping)

The scraping component of this project is responsible for gathering certification metadata directly from the websites of three providers: AWS, Microsoft, and CompTIA. Each one of these websites have two different pages that are handled:

1. The first page that only contains certifications boxes (usually the name and the image only). Each box has the url of the certification second page.
2. The second page (related to the URL contained in a specific certification box) contains all the metadata of the certification.

The data is collected via provider-specific web scrapers and saved in structured JSON format for further processing.

### 3.2.1 Target Providers and Coverage

The following providers were targeted:

- Amazon Web Services (AWS) – Certifications in cloud computing and infrastructure.
- Microsoft – Certifications covering Azure, Power Platform, business apps, and productivity tools.
- CompTIA – Certifications in cybersecurity, IT support, and networking.

Each provider has a dedicated Python scraper class inheriting from a common base class, its structure enforces modularity and reuse. Each derived scraper implements its own scraper() method, and has its own specific internal methods called by the scraper().

```
class BaseScraper:
    def __init__(self, name, url):
        self.name = name
        self.url = url
        self.data = None

    def _init_driver(self):
        #some webdriver options and configuration
        return webdriver.Chrome(options=options)

    def scraper(self):
        raise NotImplementedError('Each Site Should Implement Its \
        Own Scraper')

    def save_to_json(self, filename):
        #saving self.data to json if it's not None, using pandas

    def get_data(self):
        return self.data
```

### 3.2.2 Static vs. Dynamic Scraping

Depending on the complexity of the provider website, two different scraping techniques are used:

**CompTIA: Static Scraping (requests + BeautifulSoup)**
The CompTIA scraper uses the requests and BeautifulSoup libraries to fetch and parse HTML content directly:

```
response = rq.get(self.url)
bs = BeautifulSoup(response.text, 'html.parser')
```

Each certification page is visited to extract tabular metadata, which is converted to a pandas Series and appended to the result:

```
df = pd.read_html(StringIO(str(table)))[0]
certif_data = pd.Series(data=df['Value'].values, index=df['Field'].values)
```

**AWS & Microsoft: Dynamic Scraping (Selenium)**
Due to JavaScript-rendered elements, AWS and Microsoft require browser automation via Selenium. The browser is launched in headless mode and waits for page elements to load before scraping:

```
options = webdriver.ChromeOptions()
options.add_argument('--headless')
driver = webdriver.Chrome(options=options)
driver.get(certif_url)
```

Pages are parsed using BeautifulSoup, and content such as certification name, price, and duration is extracted. For Microsoft, pagination and layout changes are also handled:

```
button = driver.find_element(By.CSS_SELECTOR,
    'button.pagination-link[data-page="2"]')
button.click()
```

Microsoft's scraper supports two layout version (v1 and v2) based on the presence of HTML elements like #at-a-glance:

```
bs = BeautifulSoup(driver.page_source, 'html.parser')
        return self._v2_layout(driver, certif_url, bs)
        if bs.find('div', {'id': 'at-a-glance'})
        else self._v1_layout(driver, certif_url, bs)
```

### 3.2.3 Output Structure and Storage

After scraping, each provider's data is saved in JSON format using the method:

```
def save_to_json(self, filename):
    self.data.to_json(filename, orient='records', indent=2)
```

Saving as a JSON conserves any sequential format in scraped records features. The filenames follow this pattern: raw_provider_certifications.json, each JSON file stores a list of certifications, with fields such as:

| Field | Description |
|---|---|
| `Certification` | Certification name |
| `URL / Official Link` | Link to certification details |
| `Domain` | Area of expertise (e.g., cloud, security) |
| `Level` | Skill level (e.g., beginner, intermediate) |
| `Languages` | Languages offered for the exam |
| `Price / Cost (USD)` | Exam price |
| `Exam Duration` | Time limit for the exam |
| `Description` | Short summary of the certification |
| `Requirements` | Recommended experience or prerequisites |
| `Other Fields` | Variate from a provider to another, usually have no meaning, I drop them during the cleaning phase. |

Table 3.1: Fields extracted per certification and their descriptions

### 3.2.4 Execution Control

The entire scraping process is managed through a controller function in run_scraping.py:

```
sites = [CompTIA(), AWS(), Microsoft()]
for i, site in enumerate(sites):
    state.text(f"Scraping {site.name}...")
    site.scraper()
    site.save_to_json(f"raw_{site.name}_certifications.json")
```

This function dynamically invokes each scraper, tracks progress via the UI (Streamlit), and stores results locally. Here is the cleaning classes diagram:

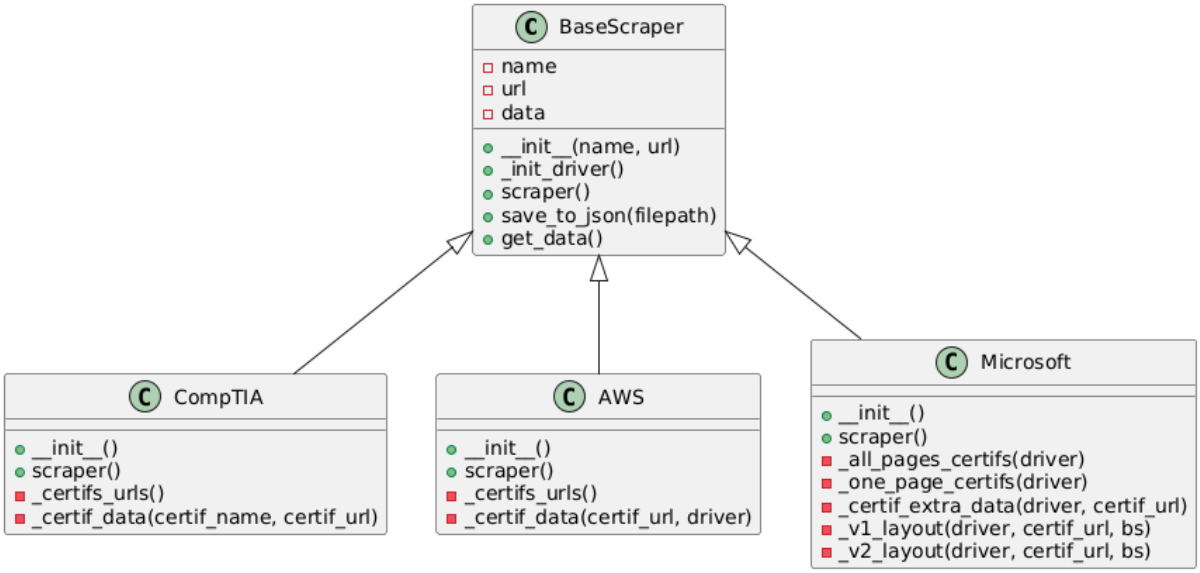


Figure 3.9: The Class Diagram of Scrapers

## 3.3 Data Cleaning and Transformation

Raw data scraped from different providers is often inconsistent in structure, terminology, and completeness. The cleaning phase is crucial to ensuring that this information can be unified and used reliably for analysis and modeling. This section details how the cleaning process standardizes and prepares the data for enrichment and visualization.

### 3.3.1 Initial Raw Data Issues

Upon inspection of the raw JSON files from AWS, Microsoft, and CompTIA, the following issues were commonly observed:

- Inconsistent field names and missing attributes across providers.
- Variations in units for exam durations (e.g., 50 vs. 50 minutes vs. 50min ).
- Different representations for cost (e.g., with or without currency symbols).
- Different representations for languages (e.g., en vs. english vs. English)
- Inconsistent naming of levels and domains (e.g., “Foundational” vs. “Beginner”).
- Duplicate, meaningless or empty records and attributes due to navigation issues during scraping.

### 3.3.2 Standardization Procedures

To address these problems, each provider’s data is passed through a dedicated cleaning class in cleaners.py. These classes inherit from a shared interface and follow the same structure. Each cleaning class performs various tasks, for instance:

- Drop invalid records: Rows with missing or duplicate certification names are removed.
- Drop invalid columns: columns with missing values ratio that is greater than 50%
- Provider tagging: Adds a Provider column to track origin.
- Cost formatting: Removes currency symbols using regular expressions and converts to float.
- Duration normalization: Converts text or mixed-format durations to integer minutes.
- Field alignment: Renames columns to match a common schema.

### 3.3.3 Execution Script: run_cleaning.py

All cleaners are orchestrated by a single controller function in run_cleaning.py. This function:

- Instantiates the appropriate cleaner based on file name.
- Applies the cleaning logic and merges results into a single DataFrame.
- Saves the combined, cleaned dataset as raw_final_data.json.

### 3.3.4 Unified Dataset Final Cleaning

After cleaning each provider's data independently and merging them into raw_final_data.json, a second, more advanced cleaning phase is applied to standardize and refine the dataset. This is handled by the FinalDataCleaner class.

This final pass ensures consistency in field values, handles remaining formatting issues, and prepares the data for machine learning. The process is invoked as follows:

| Method | Purpose |
|---|---|
| `drop_missing_name_rows()` | Removes records with no certification name |
| `drop_duplicate_certifications()` | Eliminates repeated entries |
| `reorder_columns()` | Ensures consistent column ordering |
| `drop_empty_columns()` | Removes completely empty or unnecessary columns |
| `clean_duration()` | Converts duration values to consistent numeric format (in minutes) |
| `clean_cost()` | Standardizes and parses cost fields (e.g., removes symbols) |
| `clean_certification_name()` | Formats and trims certification names for consistency |
| `standardize_columns_names()` | Renames columns to follow a unified schema |
| `standardize_languages_column()` | Converts language listings to a standardized comma-separated format |
| `standardize_Level_column()` | Normalizes the values in the Level column (e.g., Beginner, Intermediate) |
| `standardize_Domain_column()` | Groups technical domains into consistent categories |
| `final_touches()` | Applies final type conversions and formatting fixes |
| `get_data()` | Returns the fully cleaned and processed dataset |

Table 3.2: Final cleaning methods and their purposes in the `FinalDataCleaner` class

```
final_cleaner = FinalDataCleaner("raw_final_data.json")
final_data = (final_cleaner
              .drop_missing_name_rows()
              .drop_duplicate_certifications()
              .reorder_columns()
              .drop_empty_columns()
              .clean_duration()
              .clean_cost()
              .clean_certification_name()
              .standardize_columns_names()
              .standardize_languages_column()
              .standardize_Level_column()
              .standardize_Domain_column()
              .final_touches()
              .get_data())

final_data.to_csv("pre_predictions_data.csv", index=False)

```

The result of this chain is written to pre_predictions_data.csv, which serves as the

input to the machine learning enrichment phase in Chapter 3.4.
The class Diagram for Cleaners:

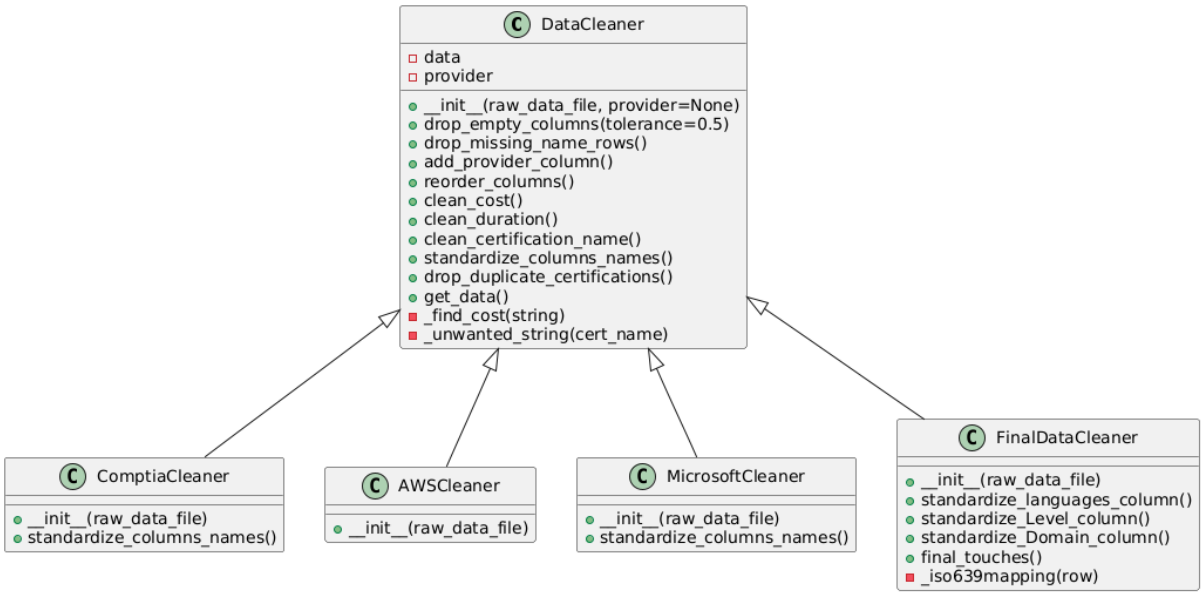


Figure 3.10: The Class Diagram of Cleaners

## 3.4 Machine Learning-Based Data Enrichment

After cleaning the dataset, several critical fields—such as Level, Domain, Cost (USD), and Exam Duration (min)—remain incomplete. These missing values can significantly hinder analysis and decision-making. To address this, the project applies supervised machine learning to intelligently predict and populate these missing fields based on patterns observed in the available data.

This phase is implemented using scikit-learn, and includes feature selection, feature encoding (one-hot encoding), model training, prediction, and injection of the results back into the dataset.

### 3.4.1 Fields Targeted for Prediction

The following fields are predicted using machine learning:

| Field | Type | Model Used |
|---|---|---|
| `Domain` | Categorical | Decision Tree Classifier |
| `Level` | Categorical | Decision Tree Classifier |
| `Cost (USD)` | Numerical | Decision Tree Regressor |
| `Exam Duration (min)` | Numerical | Random Forest Regressor |

Table 3.3: Target fields for prediction and the corresponding machine learning models

Each field uses a specific training set consisting of only rows where the target variable and features that can be encoded, so that the model can use them, are present, ensuring model reliability.

### 3.4.2 Feature Selection and Encoding

Before training, features are selected based on their predictive power using SequentialFeatureSelector, I look for the optimal number of features to keep, and the optimal direction ("forward" or "backward") using the following loop:

```
for n_features in tqdm(range(2, len(self.X.columns))):
    progressor.progress(step / total_steps)
    for direction in directions:
        selector = SequentialFeatureSelector(estimator= self.model,
        n_features_to_select=n_features, direction=direction, cv = self.cv)
        selector.fit(self.X, self.y)
        selected_features = self.X[self.X.columns[selector.get_support()]]
        score = cross_val_score(estimator=self.model,
        cv=self.cv, X= selected_features, y=self.y).mean()
```

Features considered are:

- Provider
- Domain
- Cost
- Duration
- Level

All categorical features are One-Hot encoded using get_dummies() pandas method, and non-numeric text fields are excluded. Cross-validation is performed using StratifiedKFold for classification to preserve class balance, and KFold for regression.

### 3.4.3 Dimensionality Reduction and Imbalanced Data

Dimensionality reduction was not applied in this project for the following reasons:

- The machine learning models used for prediction—namely Decision Trees and Random Forests—are non-linear by nature.
- These models do not rely on assumptions of linear relationships between features and target variables, and are therefore not sensitive to multicollinearity or high-dimensional input in the same way that linear models are.
- The feature space was already limited to a small number of relevant variables (typically two selected by feature selection), making additional dimensionality reduction unnecessary and potentially counterproductive.

The categorical data used for predicting missing values presents a remarkable imbalance:

| **Domain** | **Provider** | **Level** |
|---|---|---|
| Productivity (32) | Microsoft (77) | Intermediate (56) |
| Business Apps (13) | AWS (12) | Advanced (19) |
| Cloud (11) | CompTIA (15) | Beginner (11) |
| Security (6) | | Specialized (2) |
| Automation (5) | | |
| Data (4) | | |
| DevOps (2) | | |
| IT/Business (1) | | |

Table 3.4: Class distribution for Domain, Provider, and Level fields

The training data exhibits class imbalance, which necessitates careful model selection and validation techniques to mitigate biased predictions. In imbalanced datasets, machine learning models often favor the majority classes, resulting in poor predictive performance on minority classes. To address this issue, techniques such as stratified sampling and class weighting are employed to ensure fair and reliable model behavior across all categories.

For stratified sampling, the StratifiedKFold method from `scikit-learn` was used. This approach preserves the percentage of samples for each class in every training and validation split, maintaining the original distribution.

Regarding class weighting, the *class_ weight* parameter of the Decision Tree Classifier was set to *'balanced'* instead of the default *None*. This automatically assigns a weight to each class that is inversely proportional to its frequency in the training set, giving underrepresented classes more influence during model training.

### 3.4.4 Prediction Results and Data Injection

Once the models are trained and evaluated, they are used to predict missing values in the main dataset. The run_predicting.py script orchestrates the entire enrichment step. Each missing field is then populated only for rows where it was originally missing, preserving known values.

The final output is the post_predictions_data.csv file, which contains a fully enriched, clean, and analysis-ready dataset with over **100 IT certifications**, complete with domain, level, duration, cost, and provider metadata.
The following figure represents the diagram of the class used for predictions:

### 3.4.5 Model Performance Evaluation

The performance of each predictive model was assessed using cross-validation, a robust technique for estimating how well a model generalizes to unseen data. Below are the cross-validation scores achieved for each target field:

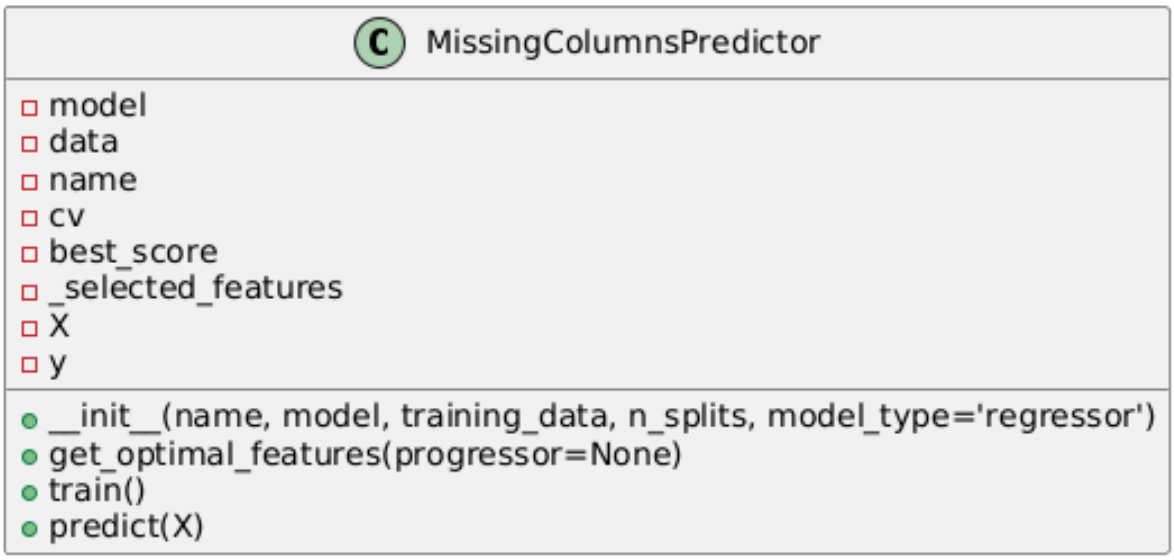


Figure 3.11: The Class Diagram of Predictor

| **Target Field** | **Best Cross-Validation Score** |
|---|---|
| Level (Classifier) | 0.869 |
| Domain (Classifier) | 0.459 |
| Cost (USD) (Regressor) | 0.930 |
| Exam Duration (min) (Regressor) | 0.837 |

Table 3.5: Best cross-validation scores for each predicted field

### 3.4.6 Interpretation

- Level (86.9%): The classification model performed very well in predicting the level of certifications, likely due to strong patterns between level and the features selected using SequentialFeatureSelector : ['Cost (USD)', 'Exam Duration (min)']
- Domain (45.9%): The domain prediction model had the lowest score. This is expected, as domain categories are more nuanced (complex to understand), and the training data records are not enough to fully grasp patterns. Additionally, domain overlaps (e.g., "cloud" vs. "automation") may confuse the model due to shared terminology and scope.
- Cost (93.0%): The model demonstrated excellent predictive performance for cost. This suggests that attributes selected provide strong signals for estimating price.
- Exam Duration (83.7%): The model also performed well in predicting exam duration, indicating consistent patterns in how providers assign durations based on other present attributes.

**Conclusion:** These results demonstrate that the machine learning models used in this project are highly effective for regression tasks (cost and duration), effective for

level prediction, and less reliable for domain prediction, which remains a challenging classification problem due to data overlap and label ambiguity.

# Chapter 4

# Exploratory Data Analysis (EDA)

## 4.1 Final Metadata Analysis

I will try in this section to uncover trends, anomalies, and relationships within the cleaned and enriched dataset. This phase uses Python libraries such as pandas, matplotlib, and seaborn to generate visual insights that aid in understanding the distribution and characteristics of IT certifications.

### 4.1.1 Overview of the Dataset

The final dataset consists of 104 certifications from three providers — Microsoft, CompTIA, and AWS, and includes features such as cost, duration, level, domain, languages, and provider.

calling the pandas method info() on the final dataset returns the following overview:

| **Column** | **Non-Null Count** | **Data Type** |
|---|---|---|
| Certification | 104 | `object` |
| Provider | 104 | `object` |
| Domain | 104 | `object` |
| Level | 104 | `object` |
| Languages | 104 | `object` |
| Cost (USD) | 104 | `int64` |
| Official Link | 104 | `object` |
| Description | 104 | `object` |
| Recommended Experience | 91 | `object` |
| Exam Duration (min) | 104 | `int64` |

Table 4.1: Overview of dataset columns and types

calling the pandas method describe() on the final dataset returns the following summary statistics for numerical fields:

| Statistic | Cost (USD) | Exam Duration (min) |
|---|---|---|
| Count | 104 | 104 |
| Mean | 162.85 | 85.63 |
| Standard Deviation | 85.84 | 37.14 |
| Minimum | 99.00 | 45.00 |
| 25th Percentile (Q1) | 100.00 | 50.00 |
| Median (Q2) | 150.00 | 91.00 |
| 75th Percentile (Q3) | 165.00 | 100.00 |
| Maximum | 392.00 | 180.00 |

Table 4.2: Summary statistics for numerical fields

### 4.1.2 Identifying Trends and Interesting Patterns

The exploratory data analysis (EDA) conducted on the final dataset reveals several meaningful trends, irregularities, and insights regarding IT certification offerings across providers. These analysis were made using the class EDA:

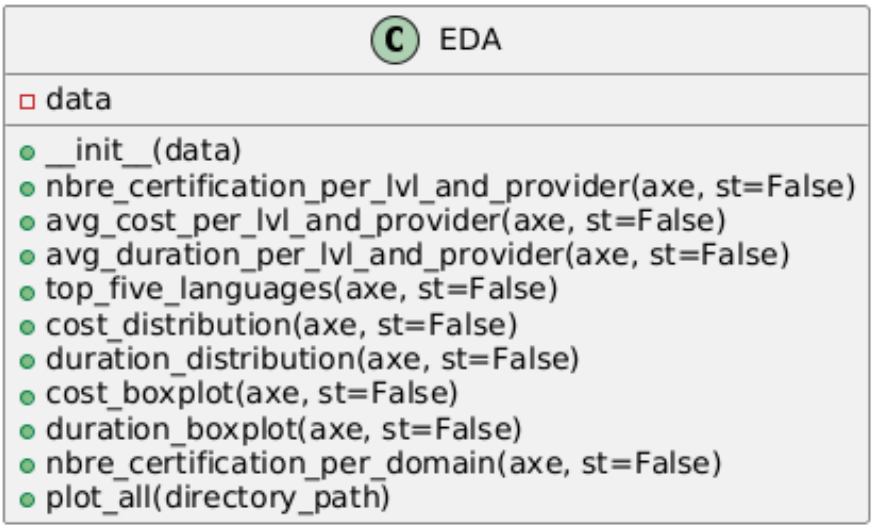


Figure 4.1: The Class Diagram of EDA

Notes: the st keyword argument is used all over my classes to integrate streamlit UI by setting it to True, so all the classes can be used without the UI. The plot_all() method is used to call all the plotting methods dynamically and save all the plots to the directory specified in directory_path argument.

**Observed Trends:**

1. **Average Cost per Level and Provider**

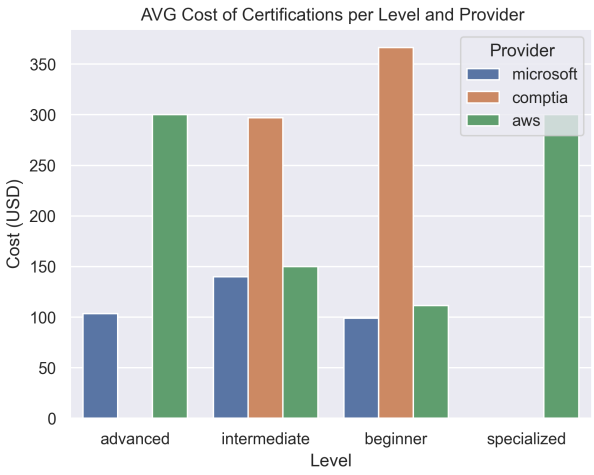


Figure 4.2: Average Cost per Level and Provider

- CompTIA certifications exhibit the highest average costs, particularly at the Beginner level..
- AWS shows a clear cost progression by level, with Advanced and Specialized certifications being the most expensive.
- Microsoft maintains moderate pricing, relatively stable across all levels.

2. **Average Exam Duration per Level and Provider**

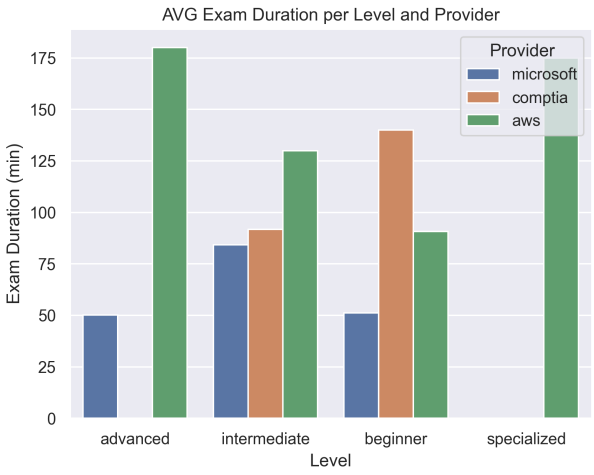


Figure 4.3: Average Duration per Level and Provider

- AWS exams, especially at Advanced and Specialized levels, are the longest—reaching up to 180 minutes.
- Microsoft exams tend to be more consistent, averaging between 50–100 minutes.
- CompTIA's beginner-level exams are surprisingly long, which might indicate more intensive content.

3. **Cost Distribution**

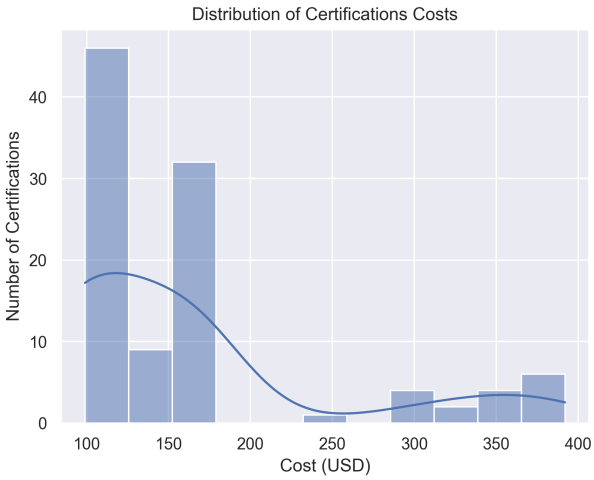


Figure 4.4: Cost Distribution

- Most certifications cost between $100 and $160, with a few outliers reaching up to $392.
- The distribution is skewed toward lower prices, with a long tail extending toward higher-cost certifications.

4. **Exam Duration Distribution**

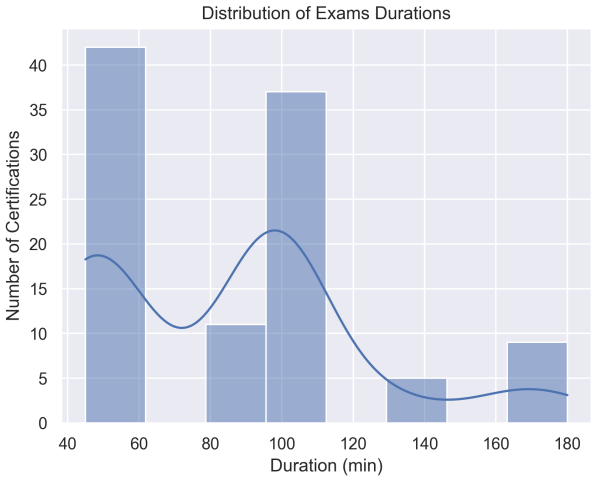


Figure 4.5: Duration Distribution

- The exam duration histogram reveals two peaks: around 60 minutes and 100 minutes, suggesting common time standards used by providers.
- Extreme durations (e.g., 180 minutes) are rare and mostly specific to AWS.

5. **Domain Distribution**

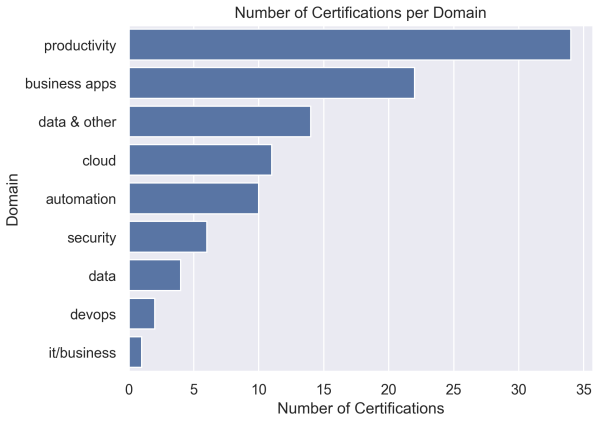


Figure 4.6: Domain Distribution

- The Productivity domain is heavily dominant (due to dominance of Microsoft certifications).
- Technical domains such as DevOps, Security, and IT/Business are underrepresented, which may influence model performance or bias downstream analysis.

6. **Level Distribution**

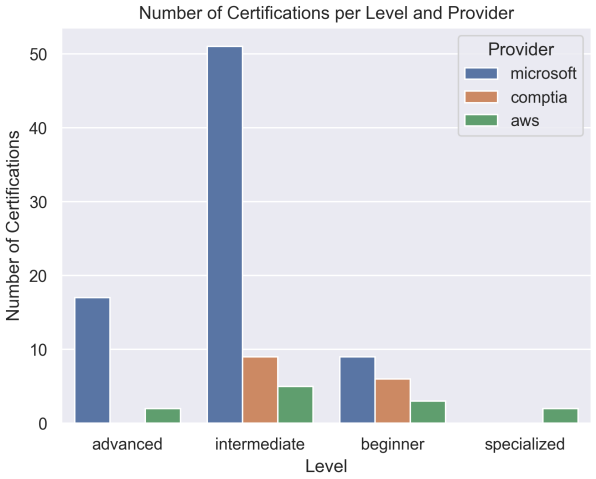


Figure 4.7: Level Distribution

- The majority of certifications fall under the Intermediate level, particularly from Microsoft.
- Specialized and Beginner levels are less frequent, indicating a market focus on mid-level professional certifications.

7. **Top Languages Offered**

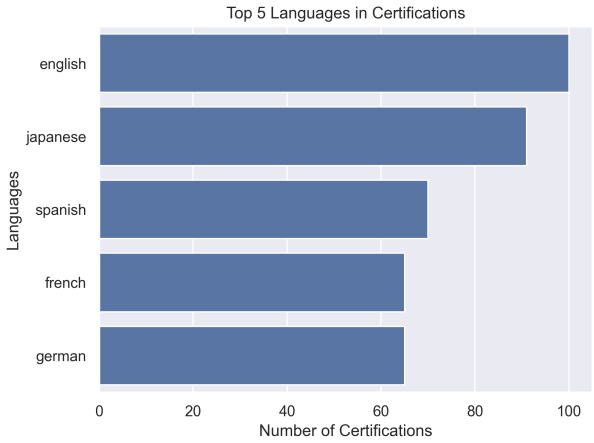


Figure 4.8: Top Languages Offered

- English is universal, followed by Japanese, Spanish, French, and German.
- This suggests global accessibility, although some important languages (e.g., Arabic) are minimally supported.

### 4.1.3 Detected Anomalies

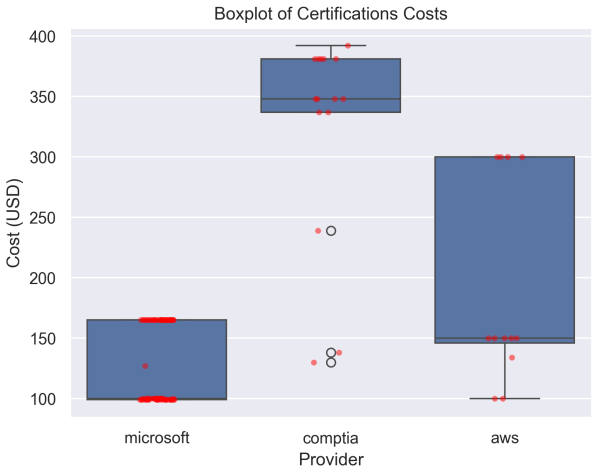


Figure 4.9: Cost Boxplot

**Visual observations**

Microsoft:

- Tight interquartile range (IQR), mostly between $100 and $165.
- Contains a few mild outliers above $165.
- Indicates stable and moderately priced certifications.
- Lowest median.

CompTIA:

- High concentration of values near the upper end ($340–$390).
- Very small IQR, but at a much higher price point than the other providers.
- Shows multiple extreme outliers below the main cost group.
- Highest median.

AWS:

- Broad spread of prices with the widest IQR.
- Median appears lower than CompTIA but slightly higher than Microsoft.
- Several outliers toward the high end ($300), but also some in the lower range ($100–$150).

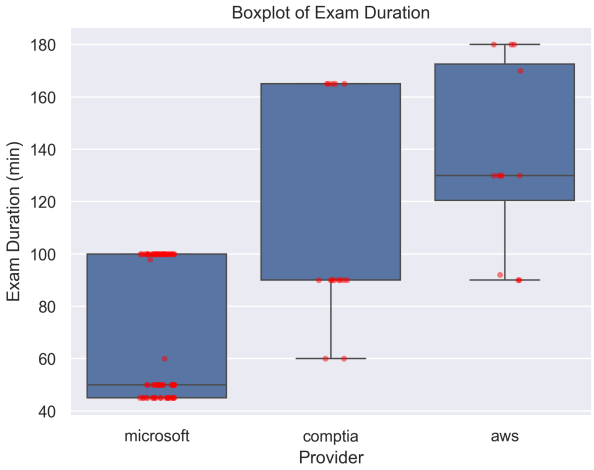


Figure 4.10: Duration Boxplot

**Visual observations**

Microsoft:

- Exam durations are tightly clustered around 45–100 minutes.
- Majority of exams are short to moderate in length.

CompTIA:

- Wide range of durations, spanning from 60 minutes to over 160 minutes.
- The broadest spreads among the three providers.
- Presence of multiple moderate and extreme outliers.

AWS:

- Durations skewed toward the high end (120–180 minutes).
- Suggests that AWS certifications are among the longest in the dataset.

**Final conclusions**

- CompTIA certifications are significantly more expensive on average than Microsoft and AWS, especially for entry-level certifications.
- Microsoft's pricing is highly consistent, reflecting a more uniform pricing policy across certifications.
- AWS shows the greatest variability, likely due to the range of roles and domains it covers (foundational to specialty).

- AWS exams are clearly more time-intensive, often lasting 2–3 hours.
- Microsoft offers faster assessments, which may make them more accessible, especially for upskilling.
- CompTIA exams lack duration standardization across their certifications.

- Pricing and duration correlate somewhat with provider reputation and technical depth. For example, AWS's long durations align with its high-cost, role-based certifications.
- Candidates may choose certifications based not only on cost or content, but also on time commitment. Longer exams can imply higher difficulty, endurance requirements, or greater assessment scope.
- Outliers in both boxplots indicate inconsistent pricing/duration policies for certain certifications

# Chapter 5

# Streamlit Dashboard, Challenges, and Future Work

## 5.1 Streamlit Dashboard Implementation

To improve accessibility and user interaction, the project includes a lightweight web interface built with Streamlit, a Python-based framework for creating interactive data apps. The dashboard allows users to execute each step of the pipeline and visualize results dynamically—without writing any code.

### 5.1.1 Features of the Dashboard

The UI is designed for modularity and user-friendliness, making it accessible to students, HR managers, and career advisors alike. It contains:

- Run Buttons: Trigger scraping, cleaning, prediction, and EDA processes on demand.
- Progress Feedback: Real-time logs and progress bars track execution steps.
- Visualization Panels: Display EDA charts such as cost distributions, exam durations, domain frequency, and more.
- File Download: Users can download the cleaned and enriched dataset (post_predictions_data.csv) directly from the interface.

## 5.2 Challenges Faced

Throughout the development of this project, several technical and methodological challenges were encountered:

1. **Technical Challenges:**
    - Website Structure Variability: Websites layouts are unique for each provider. Each site contains multiple pages. The Microsoft certification site uses multiple page layouts and has a messy HTML structure, and AWS site dynamically renders certification grids. This required flexible scraping logic and frequent updates to selectors.
    - JavaScript-Heavy Pages: Dynamic pages had to be scraped using Selenium, increasing complexity and runtime.

- Data Inconsistency: Fields like cost, duration, or domain were often missing, inconsistently formatted, or embedded deep in subpages.
- Scraper Blocking and Browser Errors: During scraping, sites occasionally returned error messages such as "This browser is no longer supported". These issues were often triggered by scraping patterns being detected and rate-limited. Solution included adding time delays, and rotating user-agent strings.

2. **Machine Learning Challenges:**

- Class Imbalance: Some fields (e.g., Level, Domain) had skewed distributions, requiring stratified sampling and class weighting.
- Sparse Inputs: Features such as certification descriptions or prerequisites were too inconsistent to use as reliable predictors, which limits the models capabilities.
- Low Predictive Power for Some Fields: The domain classifier, in particular, struggled due to label ambiguity and absence of enough features and training records.

## 5.3 Future Work

This project establishes a strong foundation for intelligent certification analytics. Several opportunities exist to extend and enhance its capabilities:

**Functional Extensions:**

- Add More Providers: Extend scraping to include Cisco, Oracle, Red Hat, Google Cloud, and others.
- Integrate Labor Market Data: Link certifications to job demand and salary insights.
- Multilingual Scraping: Expand the scope to include localized certification pages and pricing.

**Machine Learning Enhancements:**

- Advanced Models: Replace decision trees with ensemble models like XGBoost and include textual features using transformer-based language models.
- Interactive Filtering: Enable dashboard filtering by role, cost range, domain, and duration.

**Automation and Deployment:**

- Scheduled Updates: Re-run scraping periodically to make the data real-time based.
- Deployment: Host the dashboard for public access.

# General Conclusion

This project presented the design and implementation of a complete data pipeline for the collection, enrichment, and analysis of IT certification metadata from major providers; namely Microsoft, AWS, and CompTIA. Through a structured methodology involving automated web scraping, data cleaning, machine learning-based completion, and exploratory data analysis, I succeeded in transforming fragmented online information into a centralized, structured, and interactive knowledge base.

The pipeline not only standardized disparate data sources but also used supervised learning models to predict missing values such as cost, level, domain, and exam duration. These models performed particularly well for numeric fields like cost and duration, and moderately for categorical fields such as level and domain. The use of techniques like stratified sampling and class weighting helped to mitigate the impact of class imbalance on model performance.

Furthermore, an interactive Streamlit dashboard was developed to facilitate access to insights derived from the enriched dataset. Users can launch pipeline components, visualize trends, and download data without technical expertise, extending the tool's relevance to a wide audience including students, educators, and HR professionals.

Despite its achievements, the project encountered several technical and methodological challenges, such as dynamic page handling, scraper blocking, and missing metadata, which were addressed with adaptable design and feature engineering. However, limitations persist, including a restricted number of providers, lack of real-time data and integration with job market trends.

Future work could enhance this pipeline by adding support for more providers, automating periodic updates, integrating labor market data, and applying more advanced AI models to improve prediction accuracy.

In summary, this project demonstrates how data science and artificial intelligence can be harnessed to bring clarity and structure to a complex and dynamic domain, ultimately empowering informed decision-making in education and career development.

*Completed by the will and power of Allah.*

# Bibliography


[1] Ryan Mitchell *Web Scraping With Python: Collecting More Data From The Modern Web*. O'REILLY April 2018 978-1-491-98557-1

[2] *Feature Selection* https://www.geeksforgeeks.org/feature-selection-techniques-in-machine-learning/

[3] *Cross Validation* https://www.geeksforgeeks.org/cross-validation-machine-learning/

[4] *Class Weighting* https://medium.com/@rafaelnduarte/class-weight-smote-random-over-and-under-sampling-bca603378e02

[5] *Dimensionality Reduction* https://www.ibm.com/think/topics/dimensionality-reduction

[6] *Requests Documentation*. https://requests.readthedocs.io/en/latest/

[7] Richardson, L. (2007). *Beautiful Soup Documentation*. https://www.crummy.com/software/BeautifulSoup/

[8] *Selenium Documentation*. https://www.selenium.dev/documentation/

[9] *Pandas Documentation*. https://pandas.pydata.org/docs/ Sep 20, 2024 Version: 2.2.3

[10] *Scikit-Learn Documentation*. scikit-learn.org January 2025. scikit-learn 1.6.1

[11] Michael Waskom. *Seaborn Documentation*. https://seaborn.pydata.org 2012-2024

[12] *Streamlit Documentation*. https://docs.streamlit.io/

[13] *Amazon Web Services – Certification Website*. https://aws.amazon.com/certification/

[14] *Microsoft Learn – Certification Website*. https://learn.microsoft.com/en-us/credentials/browse/?credential_types=certification

[15] *CompTIA - Certification website*. https://www.comptia.org/certifications

[16] *My GitHub Repository for This Project*. https://github.com/zakariaaithssain/it-certifications-pipeline