CogniAssess

Project Supervisor
Dr. Nouman Durrani

Project Team

Sumsam Ali - 20K-1075
Mukand Krishna - 20k-0409
Bahadur Khan - 20k-1081

Project Code
F23-109D

Submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science.

FAST SCHOOL OF COMPUTING
NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES KARACHI CAMPUS

Project Supervisor	Dr. Nouman Durrani	
Project Team	Sumsam Ali	K20-1075
	Bahadur Khan	K20-1081
	Mukand Krishna	K20-0409
Submission Date	May 16, 2023	

Dr. Nouman Durrani

Supervisor

Dr. Zulfiqar Ali Memon

Head of Department

Acknowledgements

First and foremost, I extend my deepest gratitude to my supervisor, Dr. Nuouman Durrani, for his invaluable guidance, patience, and expert advice throughout the duration of this project. His profound insights and encouragement were critical to the success of our work.

I also wish to acknowledge the hard work and dedication of my team member, Bahadur Khan, whose contributions were vital in advancing our project. His collaborative spirit and technical expertise significantly enhanced our project's development and execution.

This project could not have been possible without the infrastructure provided by Hugging Face, which allowed us to host and manage our sophisticated AI models efficiently. I am grateful for the resources and support they provided, which were integral to our project's deployment phase.

Additionally, I thank all those who participated in the testing and feedback phases of our development process. Their inputs were crucial in refining our web platform and making it more robust and user-friendly.

Lastly, my sincere thanks go to my family and friends for their continuous support and encouragement throughout my studies. Their belief in my abilities has always been a source of motivation for me.

Abstract

COGNIASSESS stands as a transformative landmark in the field of **non-technical roles assessments**, aiming to iron out varied inconsistencies and streamline workflows in the wider **talent assessment** sphere, serving as a key *motivation*. It's a holistic approach to addressing universal challenges in **non-technical talent development and acquisition**. The evolving job market has sparked a *meaningful conversation* and opportunity to fill the gap of *well-crafted*, *integrated*, and focused tools that align precisely with the diverse needs of organizations and candidates in non-technical realms. This venture brings forth a (a) **LLM driven assessment generation**, (b) **Personalized Feedback** and (c) **adaptive ranking model**, all designed to elevate the assessment of non-technical skills, addressing the **individual vs general** assessment challenges. The envisioned outcome is an *MVP*—a groundbreaking platform blending *assessment*, *review*, and *feedback* processes with basic functionalities, powered by refined Large Language Models and proctoring, with a special focus on revolutionizing assessments in non-technical domains.

Contents

List of Tables

List of Figures

1 Introduction

1.1 Initial Motivation

COGNIASSESS originated from a collective realization and shared experiences from our team and advisors, where discussions were centered around the prevailing inadequacies and challenges in evaluating non-technical roles and competencies within professional spheres. Our dialogues were steeped in a collective aspiration to address the biases, lack of customization, and the predominant emphasis on technical competencies in professional evaluations, which often result in overlooking the significance of non-technical roles and skills.

The ambition was to create a platform that could **integrate** and elevate the assessment of non-technical skills, transforming it into a holistic and nuanced evaluation process tailored to the diverse and specific demands of various job roles. For us, such integration implied the incorporation of features founded on the principles of,

1. LLM based assessment generation and evaluation
2. Addressing Prevalent Biases and Challenges
3. Personalized and Role-Specific Evaluations
4. Real-time, Actionable Feedback

Our focus was unswervingly on developing and synergizing the aforementioned elements to formulate a *software solution*, valuable not merely during the project phase or within our institution, but a tool with global applicability. It's designed to be a universal solution for **candidates** who want to assess themselves avoiding *misjudgement*, *overlooking*, and *unfair evaluation* due to the absence of a comprehensive assessment tool for non-technical skills in the contemporary job landscape.

1.2 Literature Review

In the domain of assessment technology, **Artificial Intelligence (AI)** is pivotal in shaping advanced and objective assessment strategies, especially for non-technical roles. The literature corroborates the profound impact of AI in refining assessment methodologies and delivering nuanced insights into candidates' capabilities.

1. **AI and Chatbots:** A significant portion of the literature emphasizes the rise of AI-based assessments and chatbots in recruitment. These AI-driven tools are preferred by approximately 58% of job seekers, serving as the main interaction mediums during application processes by efficiently disseminating information and acquiring essential candidate data.

2. **Bias Reduction and Diversity Enhancement:** AI plays a crucial role in minimizing hiring biases by implementing anonymization of data and establishing consistent evaluation criteria. About 79% of executives agree that AI is highly effective in improving workforce diversity by fostering equitable and unbiased recruitment environments.

3. **Performance Prediction and Quality Enhancement:** The integration of AI models is instrumental in streamlining hiring processes. Examples like Unilever's application of AI underscore the transformative potential of AI, indicating a 16% enhancement in hiring quality and a 39% reduction in attrition rates, thereby fortifying employee retention and optimizing organizational performance.

4. **Workforce Forecasting:** AI's analytical capabilities extend to the interpretation of historical data, enabling precise workforce forecasting and strategic organizational planning. Companies like IBM utilize AI for actionable workforce insights, recommending adaptive strategies such as reskilling and redeployment to cultivate resilient and versatile workforces.

   In summation, the literature elucidates the multifarious benefits and applications of AI in the realm of non-technical roles assessment. It is evident that platforms like **COGNIASSESS** integrating such innovative and AI-driven methodologies are imperative for organizations aiming to navigate the complexities of hiring processes with enhanced accuracy and holistic understanding. This integration is paving the way for the development of more sophisticated and objective evaluation mechanisms, crucial for the realization of comprehensive and nuanced assessments.

1.3 Conventional Non-Technical Roles Assessment Shortcomings

In examining the existing landscape of non-technical roles assessment tools, we discerned inherent limitations and drawbacks entrenched within traditional approaches that they seemingly cannot surmount. Several notable constraints include,

1. **One-Size-Fits-All Evaluations** - The current assessment tools predominantly offer generic, non-personalized evaluations, leading to results that lack role-specific relevance and depth, rendering the insights inadequate for informed decision-making.

2. **Technical Bias** - The prevailing emphasis on technical competencies leaves non-technical skills undervalued and underassessed, impacting the holistic evaluation of candidates.

3. **Lack of Real-Time Feedback** - The absence of real-time, actionable feedback mechanisms makes it difficult for candidates to gain immediate insights and understand their performance levels during assessments.

4. **Inefficient Skill Tracking and analysis** - The unavailability of efficient tracking systems results in organizations struggling to monitor candidates' progress and development in non-technical skills. This scenario is characterized by,

   (a) Candidates being evaluated based on a predefined set of criteria

   (b) No updates or avenues for candidates to seek clarifications or enhance their skills during the assessment phase

   (c) Lack of communication channels, leaving candidates uninformed about their performance metrics and improvement areas

5. **Time and Resource Intensive** - The lack of integration and automation within existing tools necessitates extensive manual effort, leading to increased time and resource expenditure in the assessment process.

6. **Inadequate Collaborative Learning** - Current tools do not facilitate a collaborative learning environment, impacting candidates' opportunities to learn, interact, and improve through mutual cooperation and knowledge sharing.

Given the prevailing challenges, our endeavor is to conceptualize and develop an MVP employing a "Proof Of Concept" (PoC) methodology, addressing the aforementioned limitations. This approach will allow us to delve deeper into these problems and to explore and implement targeted, innovative solutions.

The issues this project targets are deeply interwoven with the broader challenges in assessing non-technical skills, guiding our focus toward creating a holistic platform that transcends being a mere 'solution' to the existing inefficiencies in non-technical roles assessments.

2 Requirements

Fine-tuning larger models often necessitates substantial GPU memory, thereby incurring considerable expenses. However, innovations like LoRa and QLoRa present effective solutions to mitigate these costs. For instance, QLoRa facilitates economical fine-tuning, making it feasible to utilize platforms like Google Colab Pro. Priced at $9.72 monthly, Google Colab Pro is an advantageous alternative due to its easy cancellation flexibility.

Moreover, employing a personal computer equipped with a graphics card, possessing a minimum of 16 GB VRAM, serves as another budget-conscious option. This configuration is adept at the proficient fine-tuning of extensive language models and can yield outcomes that are on par with conventional fine-tuning methods.

By leveraging open-source models and utilizing efficient fine-tuning methods such as QLoRa, combined with meticulous attention to the expenses of serving machines, one can achieve considerable savings in cost. This cost-effectiveness is maintainable even during instances of high traffic or real-time scenarios, contingent upon the distinct use case. For example, the estimated costs for developing a model based on Falcon 7B can be exceptionally economical—$9.99 using Google Colab Pro for development, and roughly $1.006 per hour for deploying on-demand with a single GPU machine. This ensures effective management and control of costs.

2.1 Hardware Interfaces

Moreover, employing a personal computer equipped with a graphics card, possessing a minimum of 16 GB VRAM, serves as another budget-conscious option. This configuration is adept at the proficient fine-tuning of extensive language models and can yield outcomes that are on par with conventional fine-tuning methods.

2.2 Software Interfaces

Fine-tuning larger models often necessitates substantial GPU memory, thereby incurring considerable expenses. However, innovations like LoRa and QLoRa present effective solutions to mitigate these costs. For instance, QLoRa facilitates economical fine-tuning, making it feasible to utilize platforms like Google Colab Pro. Priced at $9.72 monthly, Google Colab Pro is an advantageous alternative due to its easy cancellation flexibility[1].

By leveraging open-source models and utilizing efficient fine-tuning methods such as QLoRa, combined with meticulous attention to the expenses of serving machines, one can achieve considerable savings in cost. This cost-effectiveness is maintainable even during instances of high traffic or real-time scenarios, contingent upon the distinct use case. For example, the estimated costs for

developing a model based on Falcon 7B can be exceptionally economical—$9.99 using Google Colab Pro for development, and roughly $1.006 per hour for deploying on-demand with a single GPU machine. This ensures effective management and control of costs.

2.3 Communications Interfaces

CogniAssess will utilize various communication interfaces to ensure efficient interaction with users. This includes:

Email Integration: For user notifications, verification, and communication. Web Browser Compatibility: Ensuring seamless operation across major web browsers. Network Communications: Adherence to standard protocols such as HTTP/HTTPS for data transfer. Electronic Forms: Utilized for user input and assessments. Message Formatting: Standardized formats for data interchange, like JSON or XML. Communication Security: Implementing SSL/TLS encryption for secure data transfer. Data Transfer Rates: Optimized for efficient and fast data exchange. Synchronization Mechanisms: Ensuring real-time updates across different user sessions.

These interfaces will support the robust functioning of CogniAssess, maintaining high standards of security and efficiency in communications.

3 Design

3.1 Design Strategy

This section delves into the design strategies or decisions that significantly impact the overall organization and high-level structure of the system. These strategies provide insights into the key abstractions and mechanisms employed in the system's architecture. Each strategy is chosen based on previously stated design goals and principles, with considerations for potential trade-offs.

3.1.1 Future System Extension or Enhancement

The system is designed with scalability and flexibility in mind to accommodate future extensions or enhancements. Modular design and loose coupling are emphasized to ensure that adding new features or updating existing ones does not require a significant overhaul of the system.

- **Modular Architecture:** The system's architecture is compartmentalized into distinct modules, each handling specific functionalities. This approach facilitates easy integration of new modules or enhancements to existing modules without disrupting the entire system.
- **Use of APIs:** External functionalities are integrated through APIs, allowing for easy updates and the addition of new features.

3.1.2 System Reuse

The design incorporates principles that promote reusability, aiming to maximize the efficiency and cost-effectiveness of the system development.

- **Standardized Interfaces:** Interfaces between modules are standardized, facilitating the reuse of components in different contexts or projects.
- **Code Reusability:** Commonly used functions and classes are abstracted into reusable libraries, reducing duplication and simplifying maintenance.

3.1.3 User Interface Paradigms

The user interface (UI) is designed to be intuitive and user-friendly, adhering to established UI paradigms to ensure a seamless user experience.

- **Responsive Design:** The UI is responsive, making the system accessible across various devices and screen sizes.
- **Consistent Design Language:** A consistent design language is used throughout the system to provide a cohesive user experience.

3.1.4 Data Management

Data management strategies focus on storage, distribution, and persistence to ensure efficient and secure handling of data.

- **Database Optimization:** Databases are optimized for performance and scalability.
- **Data Distribution:** Data is distributed across multiple servers to balance the load and ensure high availability.
- **Data Persistence:** Persistent storage mechanisms are employed to ensure data integrity and durability.

3.1.5 Concurrency and Synchronization

Given the multi-user nature of the system, concurrency and synchronization are critical aspects of the design.

- **Concurrency Control:** Mechanisms are in place to handle concurrent accesses and modifications, ensuring data consistency.
- **Synchronization:** Synchronization techniques are employed to maintain the integrity of data across different modules and layers of the system.

These design strategies collectively contribute to a robust, scalable, and user-friendly system, aligning with the overarching goals and principles initially set for the project.

3.2 Frontend Design and Website Workflow

3.2.1 Initial Development Using Figma

The frontend design of our website was initially conceptualized and developed using Figma, a powerful design tool known for its flexibility and collaborative features. This phase involved meticulous planning and creative brainstorming to ensure an intuitive user interface and an engaging user experience. The design was refined through iterative processes, focusing on aesthetics, usability, and functionality. After finalizing the design in Figma, the team proceeded to translate this visual blueprint into a fully functional website.

3.2.2 Website Implementation and User Flow

Registration Verification

Upon visiting the website, the first step involves verifying the user's registration status. This crucial step ensures that only registered users can access the website's functionalities, maintaining

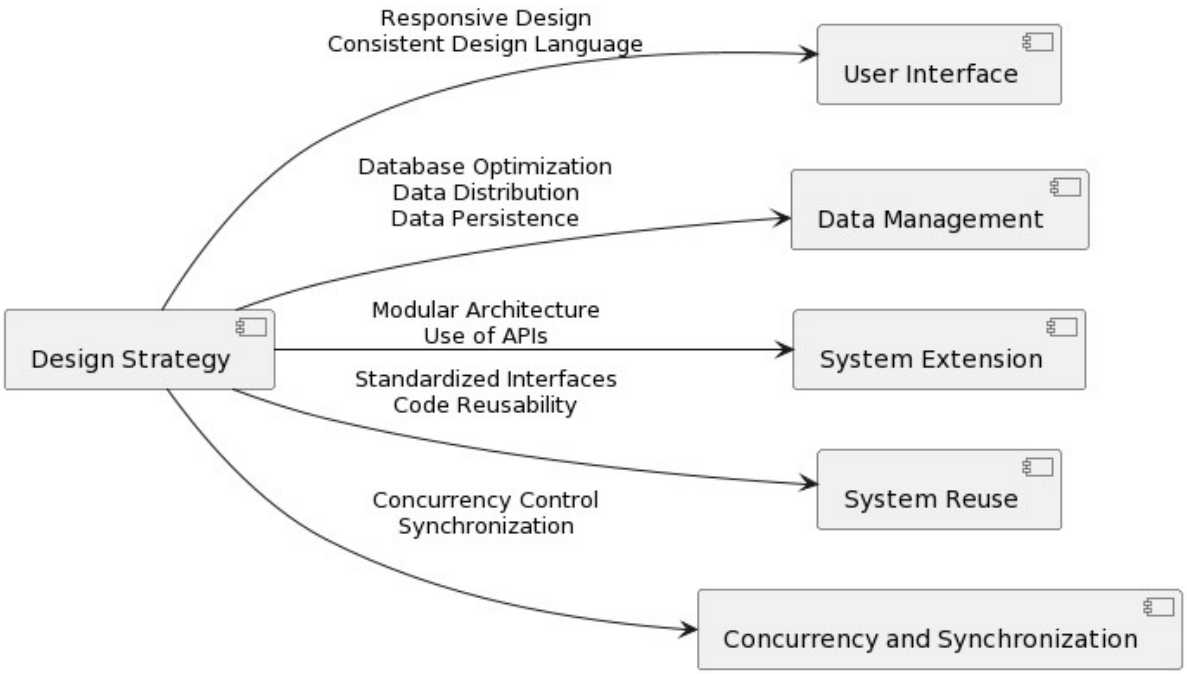

Figure 1: Design Strategy

security and personalization of the experience. Successful verification leads users to the next stage of the process.

Role Selection

Once verified, users are presented with a selection of five different non-technical roles for self-assessment. This choice allows users to tailor their experience according to their professional interests and career aspirations. The diversity in role options caters to a wide range of users, accommodating various skill sets and preferences.

Domain Selection

Following the role selection, users are directed to a domain selection page. Here, they can choose specific domains relevant to their selected role. This step is designed to fine-tune the assessment process, ensuring that the questions and challenges presented to the user are directly pertinent to their chosen field.

CV Upload and Analysis

The subsequent stage involves the user uploading their Curriculum Vitae (CV). The system then analyzes the CV, extracting key details such as skills, experiences, and qualifications. This analysis plays a critical role in personalizing the assessment to align with the user's professional background.

Tailored Question Generation

Based on the CV analysis and the selected domains, the website's Large Language Model (LLM) is prompted to generate two tailored questions for each domain. These questions are designed to assess the user's knowledge and expertise in the chosen areas. The user encounters these questions on the assessment launching page, marking the beginning of the evaluation process.

Personality and Scenario-Based Assessments

The user then progresses to the personality assessment phase, where the LLM generates questions aimed at evaluating personality traits. This is followed by a series of sample job scenario questions, crafted by the LLM to reflect real-world challenges and situations. These questions are based on prompts that align with the user's past experiences, ensuring relevance and depth in the assessment.

Feedback and Ranking

After responding to all the questions, the user's answers are evaluated by the LLM. The system generates personalized feedback and scores, contributing to the user's overall ranking on the platform. This ranking system is a critical feature, as it provides recruiters with a reliable metric to identify and connect with potential candidates.

This website, with its robust design and well-orchestrated user flow, serves as an innovative platform for self-assessment and recruitment facilitation. From the initial design in Figma to the final implementation, every step is meticulously crafted to provide a comprehensive and user-friendly experience. The integration of the Large Language Model elevates the platform's capability, offering personalized and relevant assessments for users, while also providing valuable insights to recruiters.

3.3 Detailed System Design

This section provides a comprehensive overview of the COGNIASSESS system, detailing the class diagrams, logical data models (E/R model), and the design of the Graphical User Interfaces (GUIs)[2].

3.3.1 Class Diagram and Descriptions

The class diagram of COGNIASSESS consists of several integral classes, each encapsulating specific attributes and methods necessary for the system's functionality. The key classes include User, Role, Domain, CV, Question, Assessment, Ranking, and Recruiter.

User Class The User class is central to the system, representing the end-users. Attributes include userID, name, email, password, registeredStatus, userRole, among others. Key methods encompass login, register, updateProfile, selectRole, selectDomain, uploadCV, startAssessment, completeAssessment, getFeedback, viewAssessmentHistory, calculateCurrentRank, and logout. This class interacts with almost all other classes, playing a pivotal role in system operations.

Role Class The Role class defines the various roles a user can assume within the system (e.g., candidate, recruiter). It includes attributes like roleID and roleName, and methods for role management such as addRole and removeRole.

Domain Class This class represents different domains or specializations that users can select. Attributes include domainID and domainName. Methods like addDomain and removeDomain facilitate dynamic domain management.

CV Class The CV class handles the user's curriculum vitae, storing and processing it for further analysis. Key attributes are cvID, userID, and cvContent. It includes methods for uploading and analyzing the CV.

Question Class This class is responsible for generating and managing questions for the assessments. Includes questionID, domainID, content, and type as attributes. Methods like generate and display are crucial for assessment functionality.

Assessment Class The Assessment class deals with the evaluation of users based on their responses. Attributes include assessmentID, userID, questionIDs, responses, and score. Evaluate and generateFeedback are key methods.

Ranking Class This class manages the ranking of users based on their assessment scores. Attributes are rankingID, userID, and rankScore. The calculate method computes the user's rank.

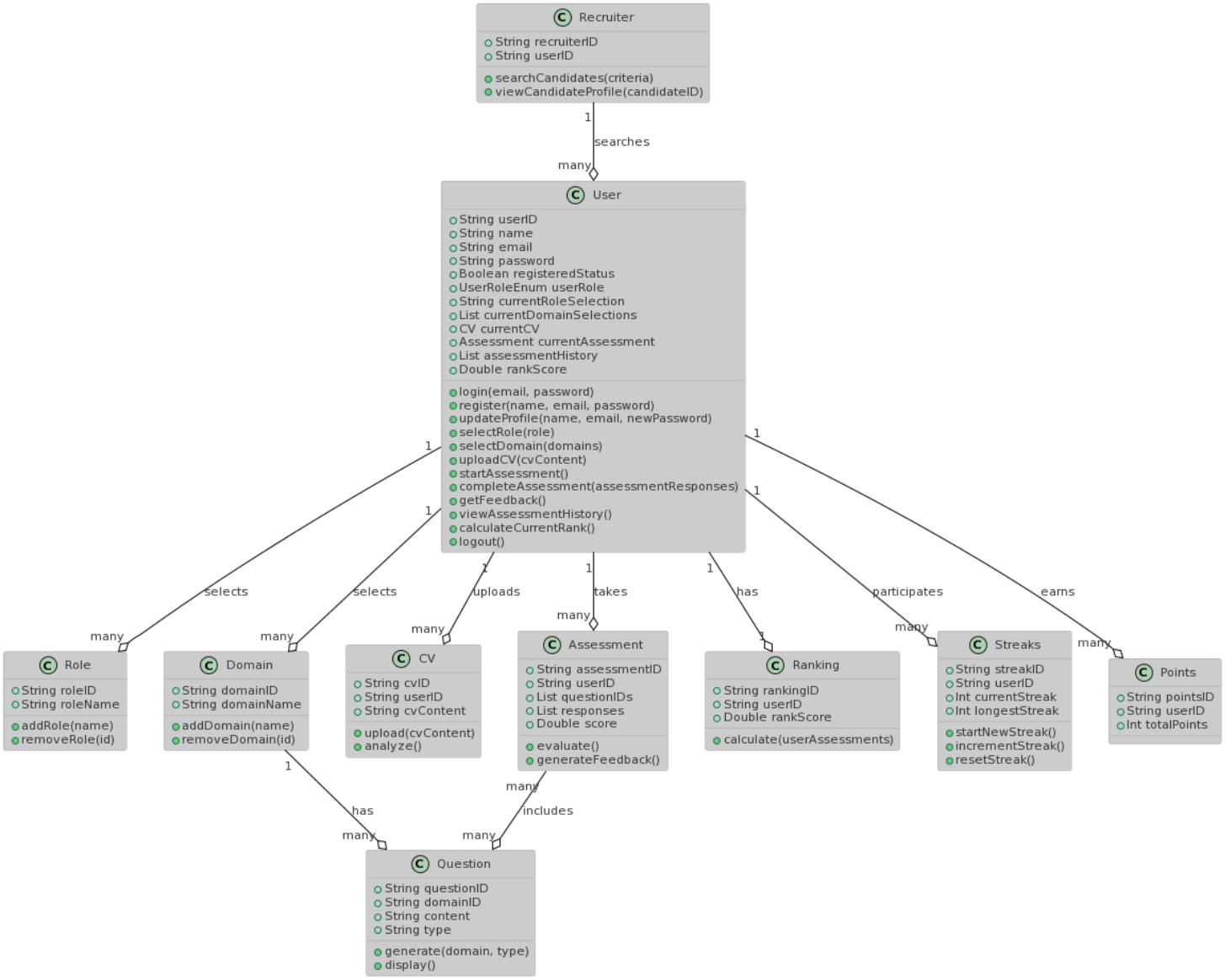

Figure 2: Class Diagram

3.3.2 Logical Data Model (E/R Model)

The E/R model of COGNIASSESS illustrates the relationships between different entities such as users, roles, domains, CVs, questions, assessments, and rankings. - Users can have multiple roles and select multiple domains. - Each domain can relate to multiple questions, which in turn are part of various assessments. - Users have a one-to-one relationship with their rankings. - The Recruiter entity interacts with the User entity for candidate search and profile viewing.

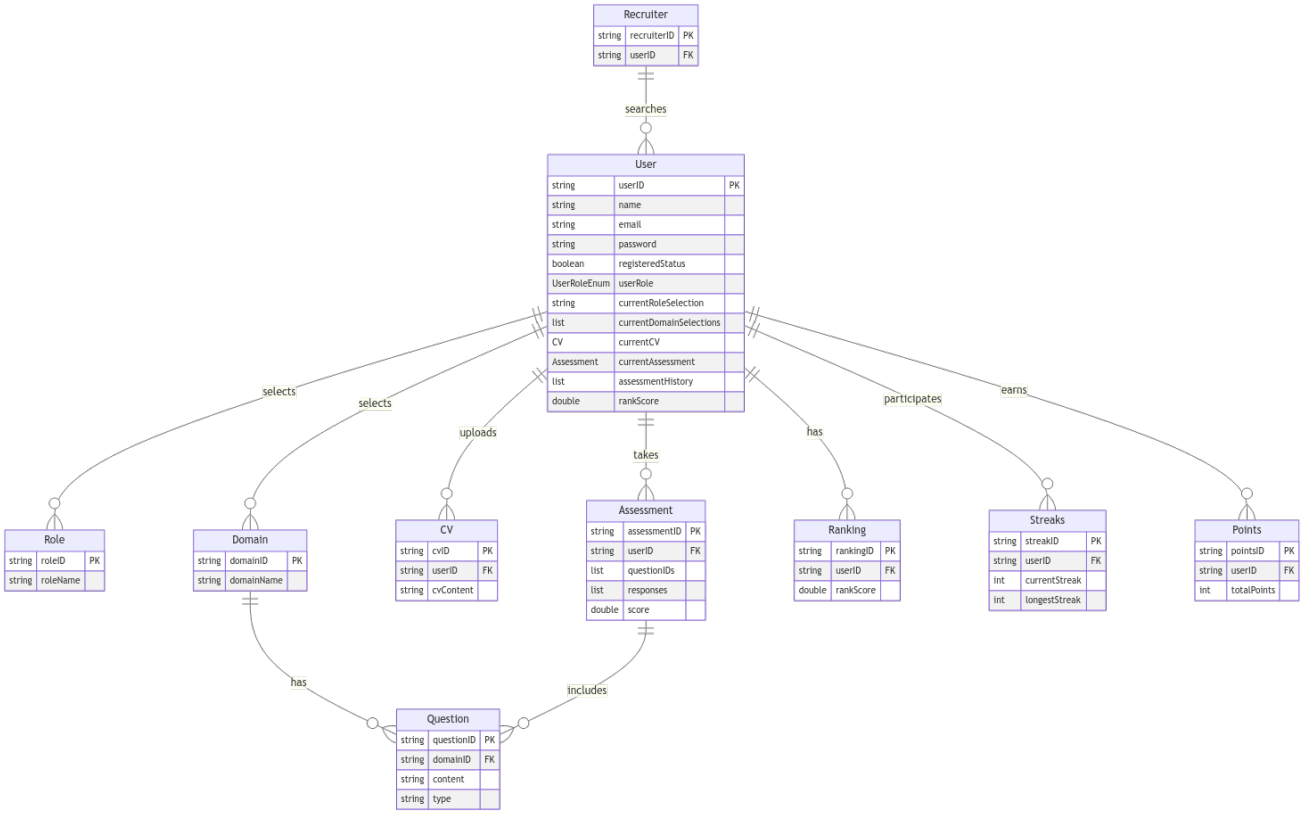

Figure 3: ER Diagram

3.3.3 Detailed GUIs

The GUI of COGNIASSESS is designed to offer a user-friendly and intuitive interaction for both candidates and recruiters. - **Registration and Login Screens:** Facilitate user access to the system. - **Role and Domain Selection Interfaces:** Allow users to choose roles and domains for personalized assessments. - **CV Upload and Analysis:** Users can upload and manage their CVs. - **Assessment Interface:** Where users engage with the assessment questions. - **Feedback and Ranking Screens:** Display personalized feedback and user rankings post-assessment.

These GUIs ensure a seamless and efficient user experience, crucial for the system's effectiveness. Please click the following link to access the GUI: Figma UI link

3.4 Data Dictionary

3.4.1 User Class

User	
Name	User
Alias	Account, Member, Participant
Where-used/how-used	The User class is used for user authentication, storing personal and login information, managing roles, and tracking assessment interactions within the system.
Content description	This class contains information about the users, including credentials, profile details, roles, and assessment data.

Column Details						
Column Name	**Description**	**Type**	**Length**	**Nullable**	**Default Value**	**Key Type**
userID	Unique identifier for the user	Integer	10	No	None	Primary Key
name	Full name of the user	Varchar	255	No	None	
email	Email address used for login	Varchar	255	No	None	Unique
password	Hashed password for user authentication	Varchar	255	No	None	
registeredStatus	Status of user's completion of the registration process	Boolean		No	False	
userRole	Role of the user in the system (e.g., Candidate, Recruiter)	Enum		No	Candidate	
currentRoleSelection	Currently selected role for assessment	Varchar	255	Yes	None	

currentDomainSelections	List of selected domains for the user's assessment	Text		Yes	None	
currentCV	Reference to the user's uploaded CV	Text		Yes	None	
currentAssessment	Reference to the ongoing assessment instance for the user	Text		Yes	None	
assessmentHistory	List of past assessments taken by the user	Text		Yes	None	
rankScore	Numeric score representing the user's ranking	Decimal	5,2	Yes	0.00	

3.4.2 Role Class

Role	
Name	Role
Alias	UserRole, Position
Where-used/how-used	The Role class is used to assign and manage roles for users within the system, impacting access control and available actions.
Content description	This class represents the roles that can be assigned to users, such as Candidate or Recruiter.

Column Details						
Column Name	**Description**	**Type**	**Length**	**Nullable**	**Default Value**	**Key Type**
roleID	Unique identifier for the role	Integer	10	No	None	Primary Key
roleName	Name of the role	Varchar	100	No	None	

3.4.3 Domain Class

Domain	
Name	Domain
Alias	ExpertiseArea, Field, Specialization
Where-used/how-used	This class is utilized to categorize users and questions based on their area of expertise or interest. It is used in the assessment process to ensure that users are evaluated within relevant domains.
Content description	Contains the details of the various domains or categories that can be selected by users for assessments.

Column Details						
Column Name	**Description**	**Type**	**Length**	**Nullable**	**Default Value**	**Key Type**
domainID	Unique identifier for the domain	Integer		No	None	Primary Key
domainName	The name representing the specific domain of expertise or interest	Varchar	255	No	None	

3.4.4 CV Class

CV (Curriculum Vitae)	
Name	CV
Alias	Resume
Where-used/how-used	The CV class is used within the user profile to store and manage the user's curriculum vitae, which is critical for assessment and recruiter evaluation processes.
Content description	This class encapsulates the data structure for storing user CVs, including personal information, work experience, education, skills, and other relevant details.

Column Details						
Column Name	**Description**	**Type**	**Length**	**Nullable**	**Default Value**	**Key Type**

cvID	Unique identifier for the user's CV	Integer		No	None	Primary Key
userID	Reference to the user that the CV belongs to	Integer		No	None	Foreign Key
cvContent	The content of the CV, including text and possible formatting	Text		Yes	None	

3.4.5 Question Class

Question	
Name	Question
Alias	AssessmentQuestion
Where-used/how-used	Utilized in the assessment process, where questions are presented to users based on their selected domain to evaluate their knowledge and skills.
Content description	Encompasses the attributes and methods necessary for creating, storing, and displaying questions for user assessments.

Column Details						
Column Name	**Description**	**Type**	**Length**	**Nullable**	**Default Value**	**Key Type**
questionID	Unique identifier for each question	Integer		No	None	Primary Key
domainID	The domain with which the question is associated	Integer		No	None	Foreign Key
content	The textual content of the question itself	Text		No	None	
type	The type or category of the question (e.g., multiple choice, free text)	Varchar	50	No	None	

3.4.6 Assessment Class

Assessment	
Name	Assessment
Alias	Evaluation, Test, Exam
Where-used/how-used	This class is used to manage the assessment processes, storing user responses to questions, calculating scores, and providing feedback. It is fundamental to the system's evaluation mechanism.[2]
Content description	Contains user assessment attempts, responses, scores, and any related metadata necessary for evaluating the performance of users.

Column Details						
Column Name	**Description**	**Type**	**Length**	**Nullable**	**Default Value**	**Key Type**
assessmentID	Unique identifier for the assessment instance	Integer		No	None	Primary Key
userID	Reference to the user taking the assessment	Integer		No	None	Foreign Key
questionIDs	List of IDs of questions included in the assessment	Text		Yes	None	
responses	User responses to the assessment questions	Text		Yes	None	
score	The score obtained by the user on this assessment	Decimal	5,2	Yes	0.00	

3.4.7 Ranking Class

Ranking	
Name	Ranking
Alias	Leaderboard, Scoreboard
Where-used/how-used	The Ranking class is used to maintain a leaderboard of users, reflecting their performance and progress within the system based on assessment results.
Content description	Stores the ranking information for users, including their overall scores and positions in comparison to other users.

Column Details						
Column Name	**Description**	**Type**	**Length**	**Nullable**	**Default Value**	**Key Type**
rankingID	Unique identifier for the user's ranking	Integer		No	None	Primary Key
userID	Reference to the user whose ranking is represented	Integer		No	None	Foreign Key
rankScore	Numerical representation of the user's ranking	Decimal	5,2	No	None	

3.5 Application Design

This section details the application design of COGNIASSESS, including sequence diagramsand state transition diagrams to illustrate the dynamic behavior and interactions within the application.

3.5.1 User Authentication Sequence Diagram

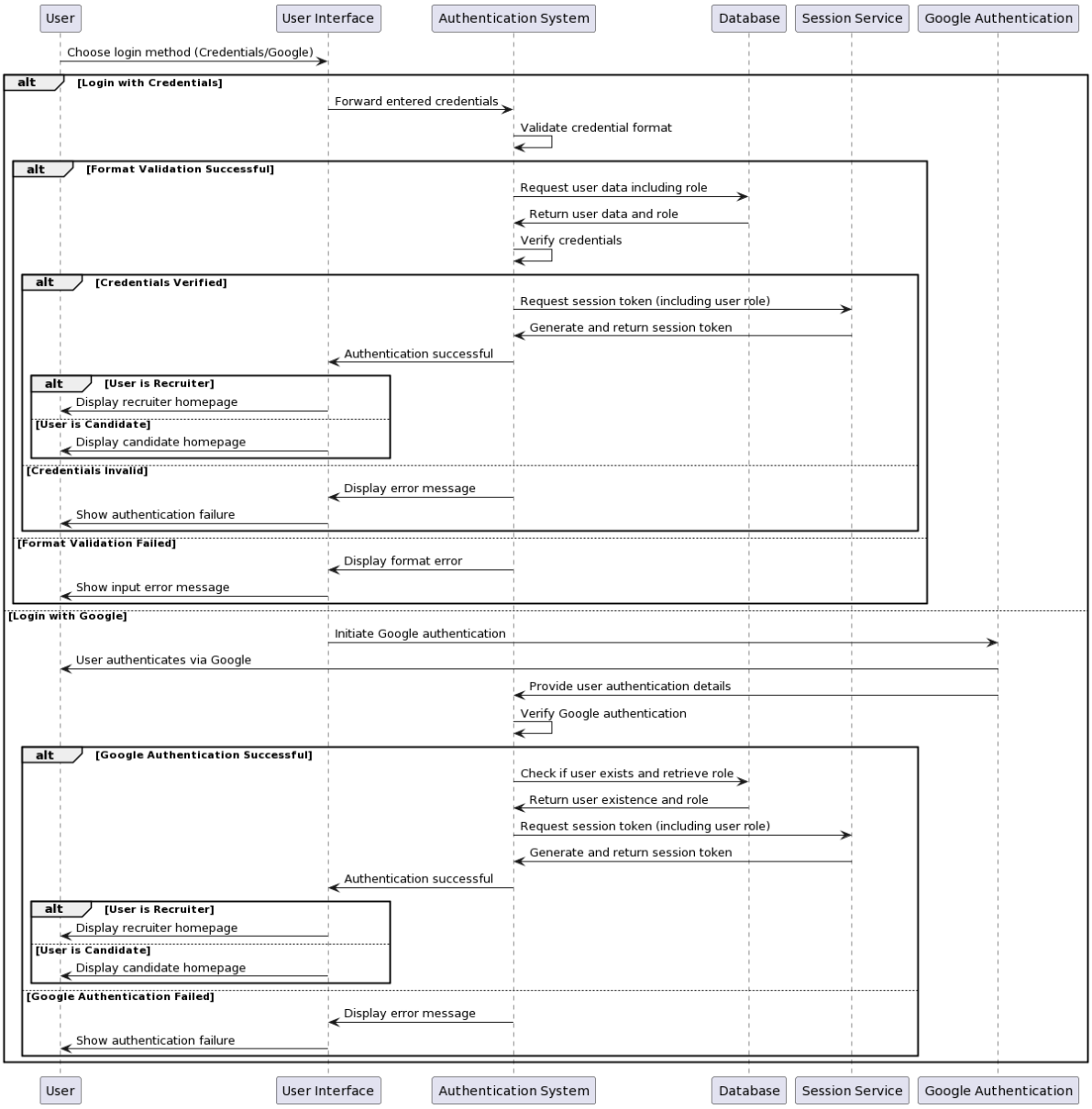

Figure 4: User Authentication Sequence Diagram

This diagram illustrates the process of user login, authentication, and error handling. It details the interaction between the User, Authentication system, and Database, highlighting steps like user input, credential verification, and session initialization.

3.5.2 Assessment Process Sequence Diagram

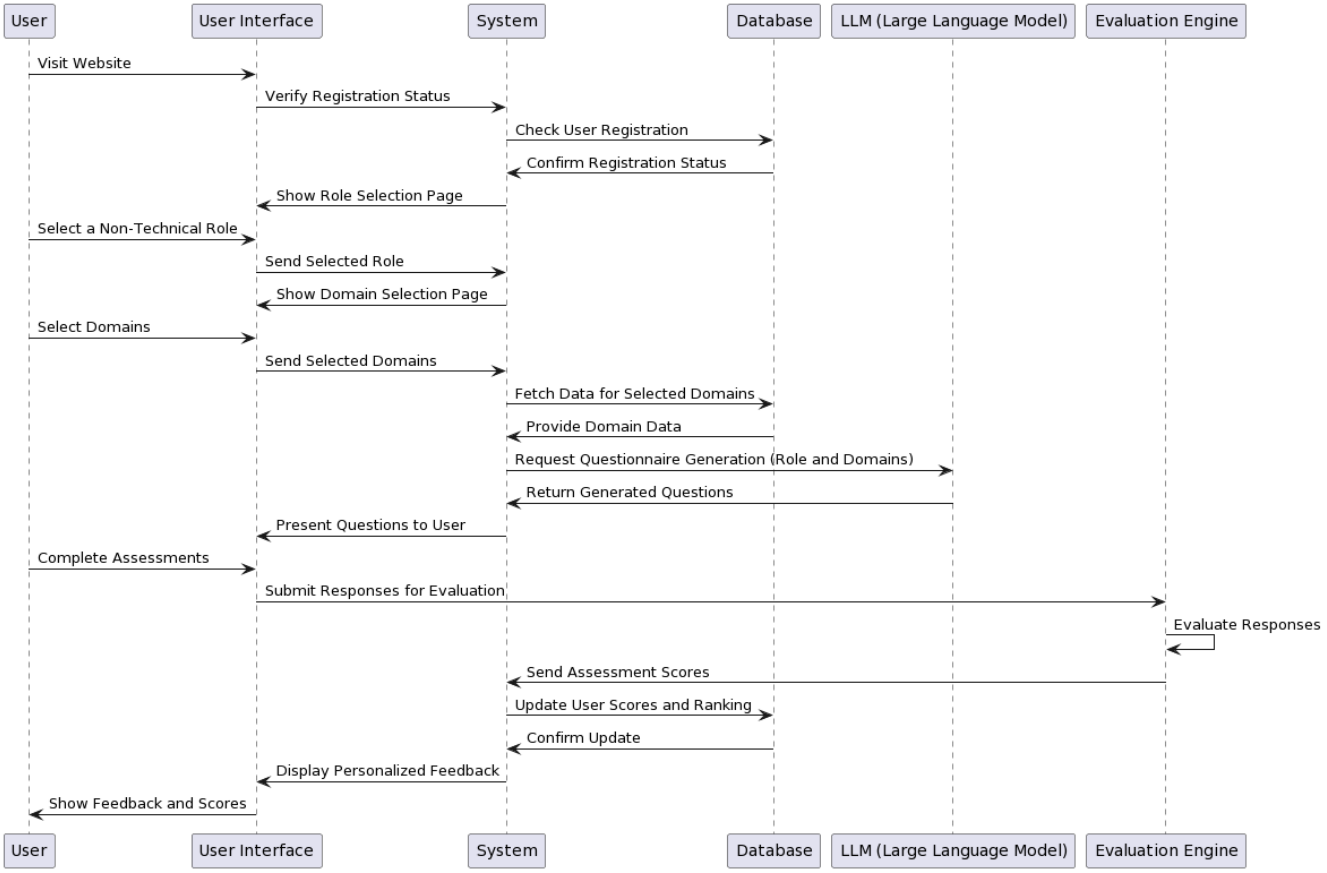

Figure 5: Assessment Process Sequence Diagram

This diagram shows the sequence of events from when a user starts an assessment to when they receive their results. It describes interactions between User, Assessment Module, Questionnaire, and Evaluation Engine, covering question retrieval, answer submission, and scoring.

3.5.3 LLM Generation Process Sequence Diagram

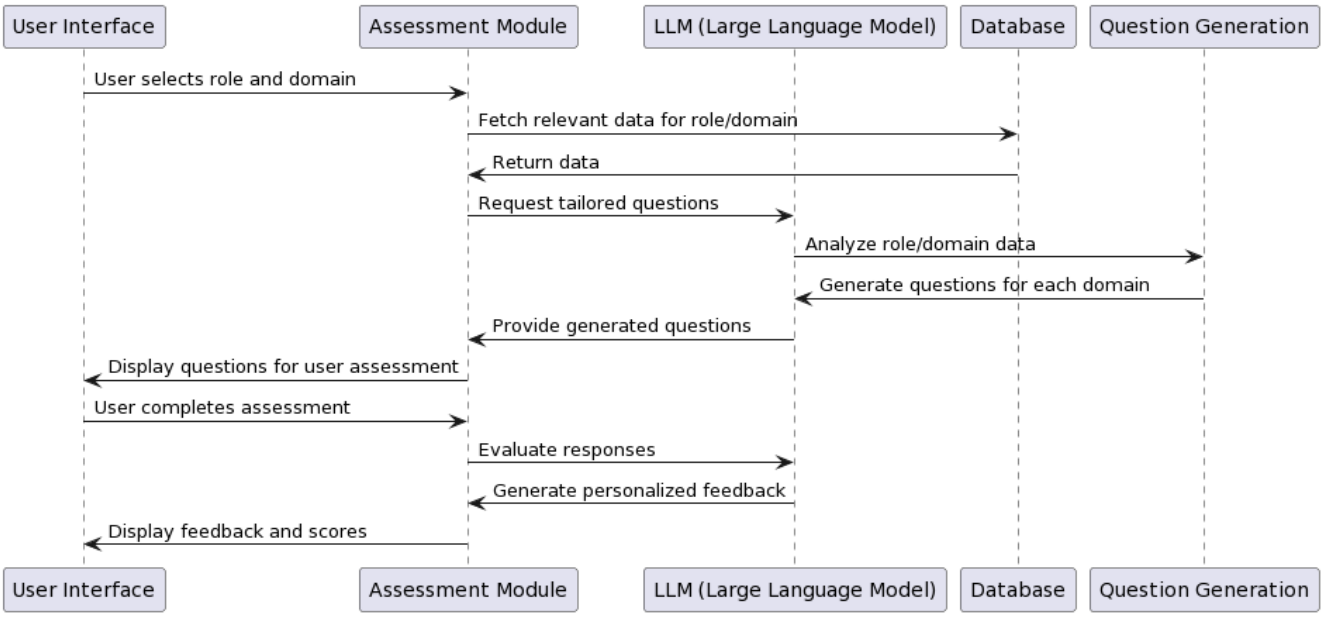

Figure 6: LLM Generation Process Sequence Diagram

This diagram illustrates the process of generating assessment content using a fine-tuned Large Language Model (LLM) hosted on Hugging Face. It demonstrates the sequence of events and interactions for three distinct types of assessments: job scenario, personality, and domain-role based. The diagram covers the system's interaction with the LLM for each assessment type, detailing the steps from the initial request for content generation to the presentation of tailored questions to the user. This includes:

1. **Job Scenario Assessment**: Showcasing how the system prompts the LLM to generate realistic job scenario questions based on the user's selected role and domain.

2. **Personality Assessment**: Illustrating the generation of questions aimed at evaluating personality traits.

3. **Domain-Role Based Assessment**: Depicting the process of creating domain-specific questions that align with the user's chosen non-technical role.

The sequence diagram also highlights the integration of the LLM with the system's database and user interface, ensuring a seamless and dynamic assessment creation process.

3.5.4 LLM Fine-Tuning Process Sequence Diagram

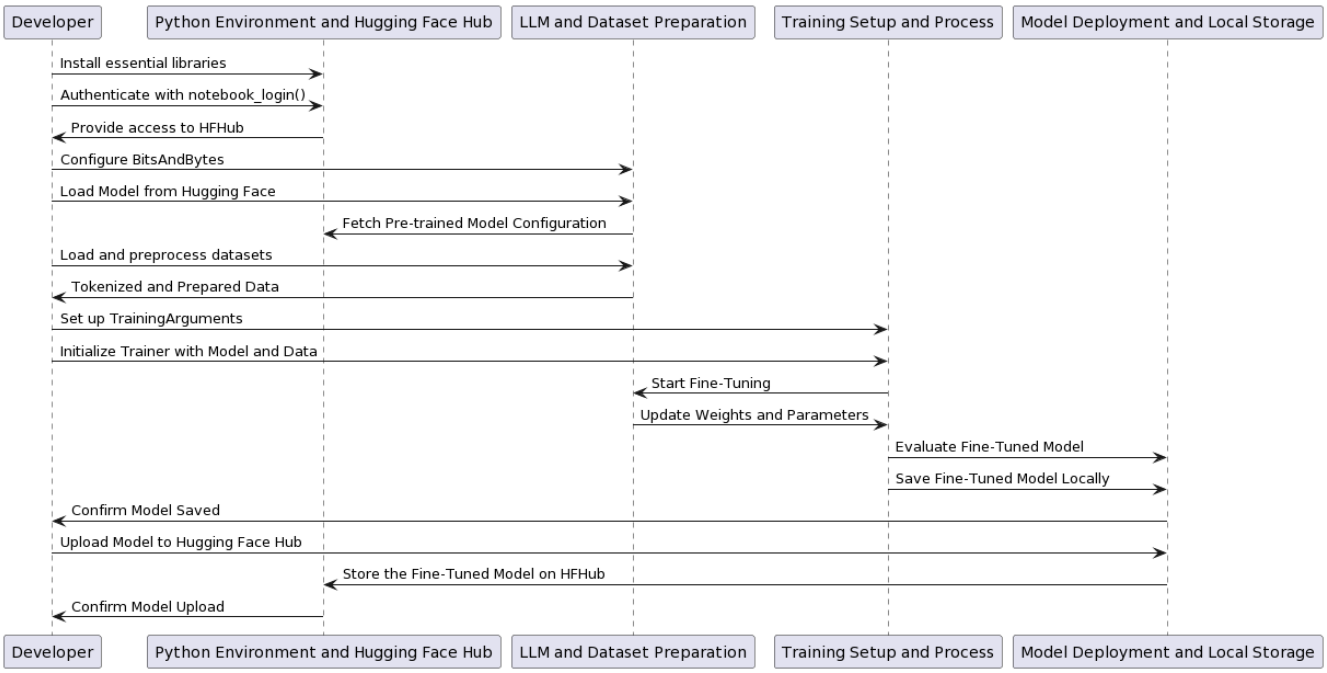

Figure 7: LLM Fine-Tuning Process Sequence Diagram

This diagram visualizes the fine-tuning process of a Large Language Model (LLM) hosted on Hugging Face, specifically tailored for the CogniAssess project. It encapsulates the comprehensive steps involved in preparing, training, and deploying the fine-tuned LLM. The sequence includes:

1. **Module Installation and Imports**: Showcasing the installation of essential Python libraries like 'torch', 'transformers', and 'accelerate', and their subsequent imports into the environment.

2. **Model and Tokenizer Setup**: Illustrating the process of configuring and initializing the LLM with specific parameters, including the BitsAndBytes configuration for memory-efficient model loading.

3. **Dataset Preparation**: Depicting the steps for loading, preprocessing, and tokenizing datasets, specifically focused on generating prompts for non-technical role assessments.

4. **Training Parameter Configuration**: Outlining the setup of training arguments, including batch sizes, learning rates, and optimization strategies.

5. **Model Fine-Tuning**: Detailing the training process, where the LLM is fine-tuned on the prepared dataset using specific training parameters and strategies.

6. **Model Evaluation and Deployment**: Highlighting the evaluation of the fine-tuned model and its subsequent deployment, including saving and uploading the model to Hugging Face Hub.

This sequence diagram emphasizes the intricate and methodical approach to fine-tuning an LLM for the specific needs of the CogniAssess project, ensuring the model is optimally trained for assessing non-technical roles.

3.6 State Diagram

State diagrams provide insights into the various states of system components and how transitions occur between these states.

3.6.1 User State Diagram

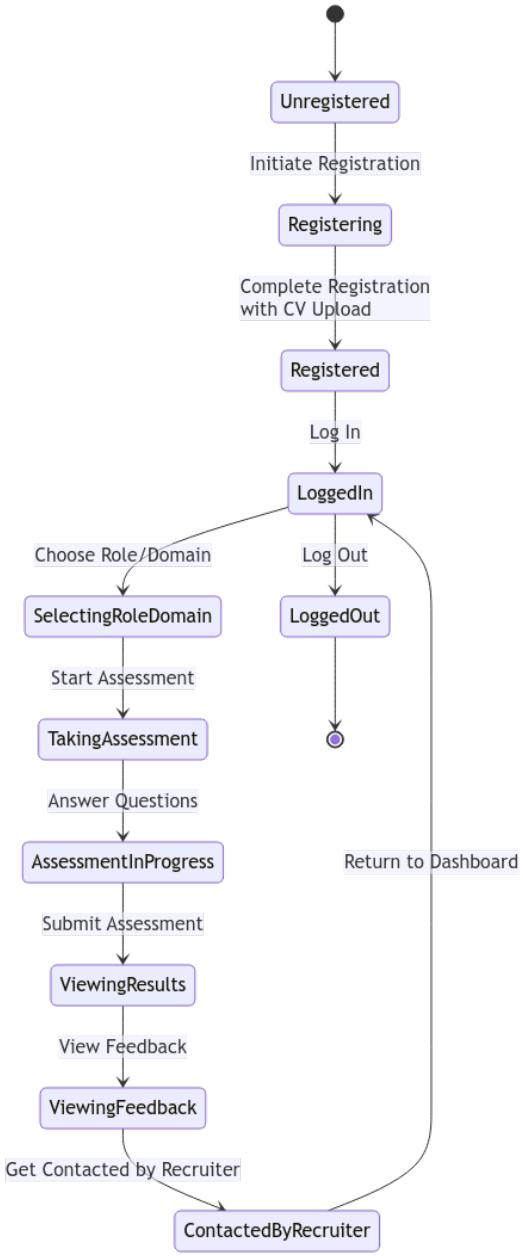

Figure 8: User State Diagram

This diagram represents the different states of a user account (e.g., Registered, Logged In, Taking Assessment) and describes what causes transitions between states, such as logging in, starting an assessment, or logging out.

3.6.2 Assessment Module State Diagram

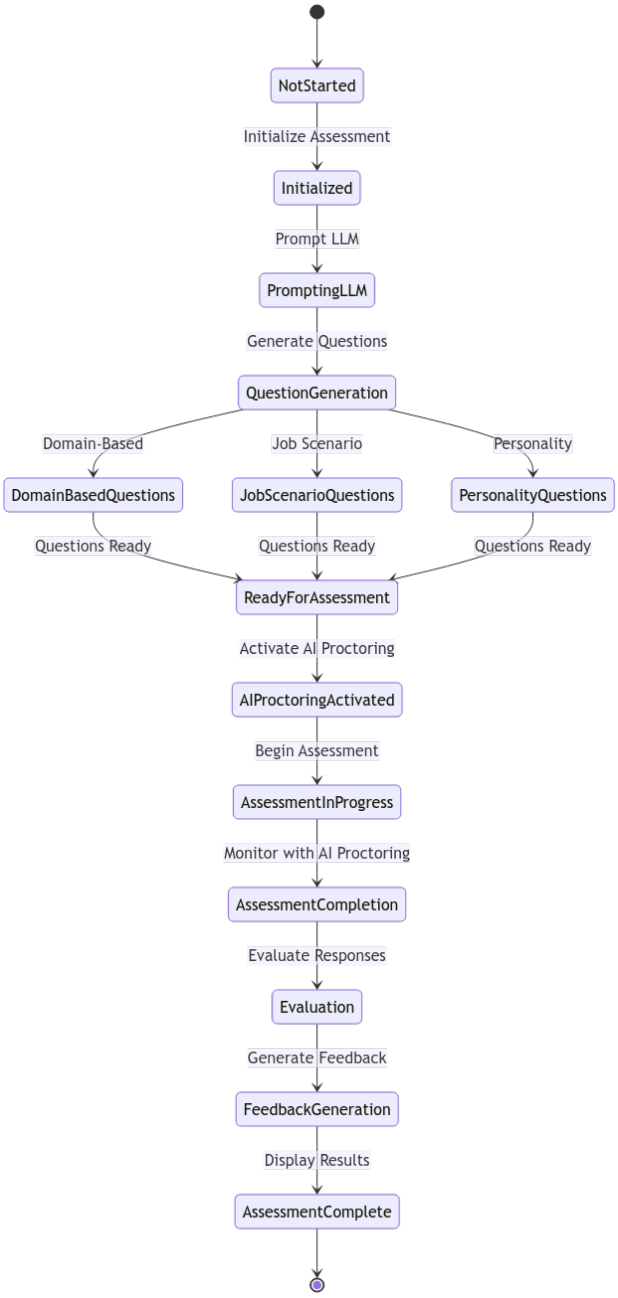

Figure 9: Assessment Module State Diagram

This diagram shows the different states of the assessment process (e.g., Not Started, In Progress, Completed) and details triggers for state transitions like starting an assessment, submitting answers, and receiving results.

3.7 DFD Level 1 Diagram

The Data Flow Diagram (DFD) Level 1 provides a high-level overview of data movement within the system.

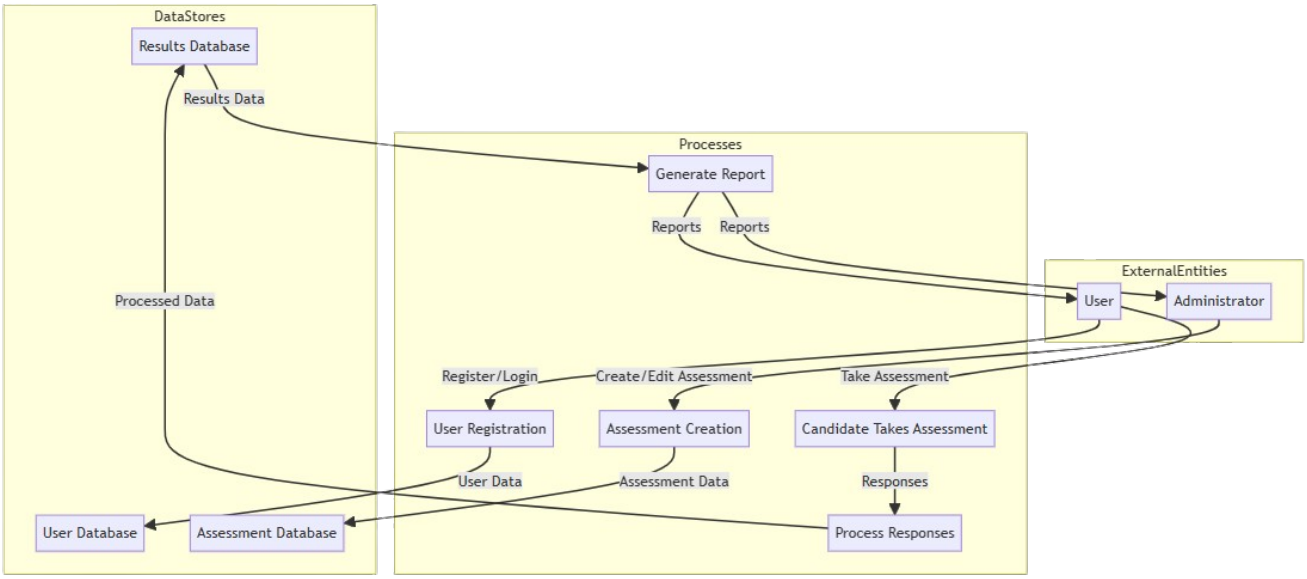

Figure 10: DFD Level 1 Diagram

This diagram includes major system components like User Management, Assessment Processing, and Reporting and describes the flow of data between these components and how they interact to provide the system's functionality.

4 Implementation

The methodology for the COGNIASSESS project, a comprehensive platform for non-technical skill assessment in recruitment, involves a series of meticulously planned steps. This section outlines the methodologies employed from data collection to the deployment of the model on the Hugging Face platform.

4.1 Data Collection

The data collection stage is pivotal in training the AI-driven large language models (LLMs) for the COGNIASSESS platform. This phase involves gathering, filtering, and preparing datasets that reflect the diversity and complexity of non-technical roles and assessments.

4.1.1 Sources of Data

For the COGNIASSESS project, the data sources were carefully selected to encompass a wide range of non-technical roles and scenarios. The datasets used include:

- **QnA for Non-Technical Roles:** A dataset providing a variety of questions and answers related to non-technical job roles.
- **Introduction Training:** This dataset includes introductory level training materials and queries, essential for baseline assessments.
- **Roleplay Training:** A collection of roleplay scenarios and responses, offering insights into practical and situational aspects of non-technical roles.

4.1.2 Data Selection Criteria

The selection of these datasets was guided by specific criteria to ensure their relevance and efficacy in training the model:

- **Alignment with Non-Technical Skills:** The data must be representative of various non-technical roles and the skills required for them.
- **Diversity and Inclusivity:** Ensuring the data covers a broad spectrum of scenarios, roles, and domains to foster an inclusive assessment platform.
- **Quality of Content:** The clarity, correctness, and relevance of the questions and answers were paramount in the selection process.
- **Adaptability for AI Training:** The datasets were chosen for their compatibility with AI-driven analysis and modeling techniques.

4.1.3 Data Acquisition Process

The acquisition process involved several steps, each critical to securing high-quality and relevant data for the project:

1. **Dataset Identification:** Initial research and exploration were conducted to identify datasets that fit the project's criteria.
2. **Accessing Data Repositories:** The datasets were accessed through their respective online repositories, primarily hosted on platforms like Hugging Face.
3. **Data Extraction:** Relevant data from these datasets were carefully extracted. This process involved selecting specific portions of the datasets that were most pertinent to the project's needs.
4. **Data Validation:** Post-extraction, the data underwent a validation process to ensure its integrity and applicability to the project's objectives.

For instance, the datasets were loaded using Hugging Face's 'datasets' library, as demonstrated in the following Python code snippet:

```
from datasets import load_dataset

Non_Technical_Roles_dataset = load_dataset('Sumsam/QnA_for_Non-Technical_Roles')
Intro_Training_dataset = load_dataset('Sumsam/Introduction_training')
Role_playing_dataset = load_dataset('Sumsam/Roleplay_training')
```

This code effectively loads the datasets into the working environment, setting the stage for subsequent preprocessing and analysis.

4.2 EDA - Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an essential aspect of any data science or machine learning project. It involves summarizing the main characteristics of data, often visually, to gain insights and a better understanding of the dataset. This section delves into the EDA for the provided datasets, broken down into specific subsections.

4.2.1 QnA for Non-Technical Roles Dataset

Dataset Overview This section involves loading the dataset into a pandas DataFrame and performing initial explorations. Commands like `head()`, `info()`, and `describe()` are used to understand the dataset's structure, data types, and basic statistical details.

	Non-Technical Role	Assessment Domain	Question
0	Content Developer	Adaptability	How do you adapt your content strategy in resp...
1	Content Developer	Adaptability	How do you handle unexpected changes in projec...
2	Content Developer	Adaptability	Can you provide an example of a time you had t...
3	Content Developer	Adaptability	How do you stay productive and focused when fa...
4	Content Developer	Adaptability	What strategies do you use to adapt to new tec...

Figure 11: Dataset Overview

Null Value Analysis Identifying missing values is crucial for data integrity. This subsection details the process of using `isnull().sum()` to find null values and discusses the importance of handling missing data in the analysis.

Categorical Variable Exploration The analysis of the 'Non-Technical Role' and 'Assessment Domain' categories includes:

- Counting the occurrences of each category.
- Visualizing the category distributions with bar charts and pie charts to understand their frequency and significance.

Word Cloud Analysis Word clouds are generated for the 'Question' and 'Assessment Domain' columns using the `WordCloud` library. This subsection discusses the insights gained from these visual representations, like prevalent themes and common words.

Figure 12: Word Cloud Analysis

Role Distribution Analysis This subsection describes the use of visualization tools (such as Plotly or seaborn) to analyze the distribution of different non-technical roles. The insights from bar charts and pie charts are discussed to understand the roles' prevalence in the dataset.

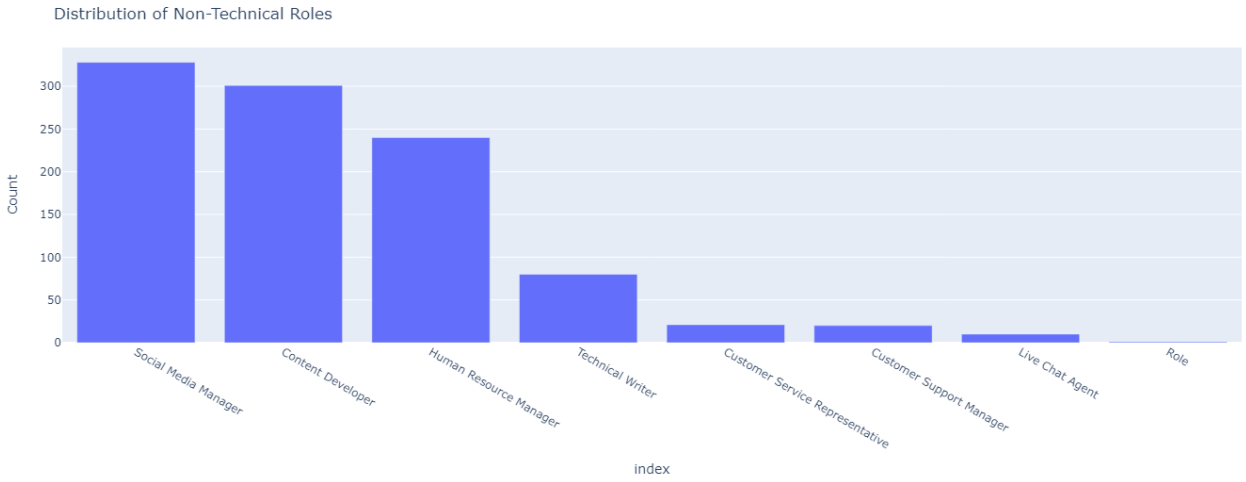

Figure 13: Role Distribution Analysis

Detailed Textual Analysis A thorough examination of the questions and answers in the dataset is conducted to identify patterns, common themes, and specific focuses. This involves analyzing the content and context of the textual data.

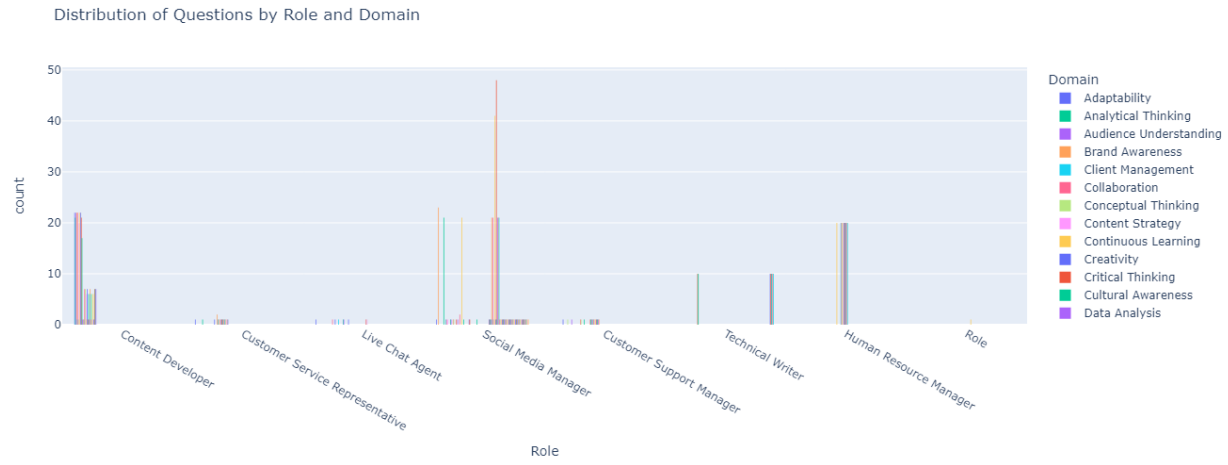

Figure 14: Detailed Textual Analysis

4.2.2 Introduction Dataset (Cogniassess Intro)

Initial Data Examination This section is dedicated to the initial loading and examination of the 'Cogniassess Intro.json' dataset. Techniques such as `head()`, `info()`, and `describe()` are

utilized for a preliminary understanding of the dataset's structure, data types, and basic statistics.

	Question	Answer
0	Excuse me, what's your name?	I am Cogni Assess.
1	Can you tell me what your name is?	My designation is Cogni Assess.
2	How can I address you?	You may refer to me as Cogni Assess.
3	What should I call you?	Please call me Cogni Assess.
4	I'm sorry, what was your name?	My name is Cogni Assess.

Figure 15: Data Examination

Unique Value Analysis The analysis of unique values in key columns like 'Question' and 'Answer' is essential. This subsection details the methods used for identifying unique values and the insights they provide about the dataset's content and diversity.

Distribution of Questions and Answers A crucial part of the EDA involves understanding the frequency and distribution of questions and answers. This subsection describes the use of visualization tools, such as seaborn's countplot, to graphically represent the data's distribution.

Word Cloud for Questions and Answers This part focuses on creating word clouds for the 'Questions' and 'Answers' text. It involves using the `WordCloud` library and matplotlib for visualization, aiming to highlight the most common words and themes in the dataset.

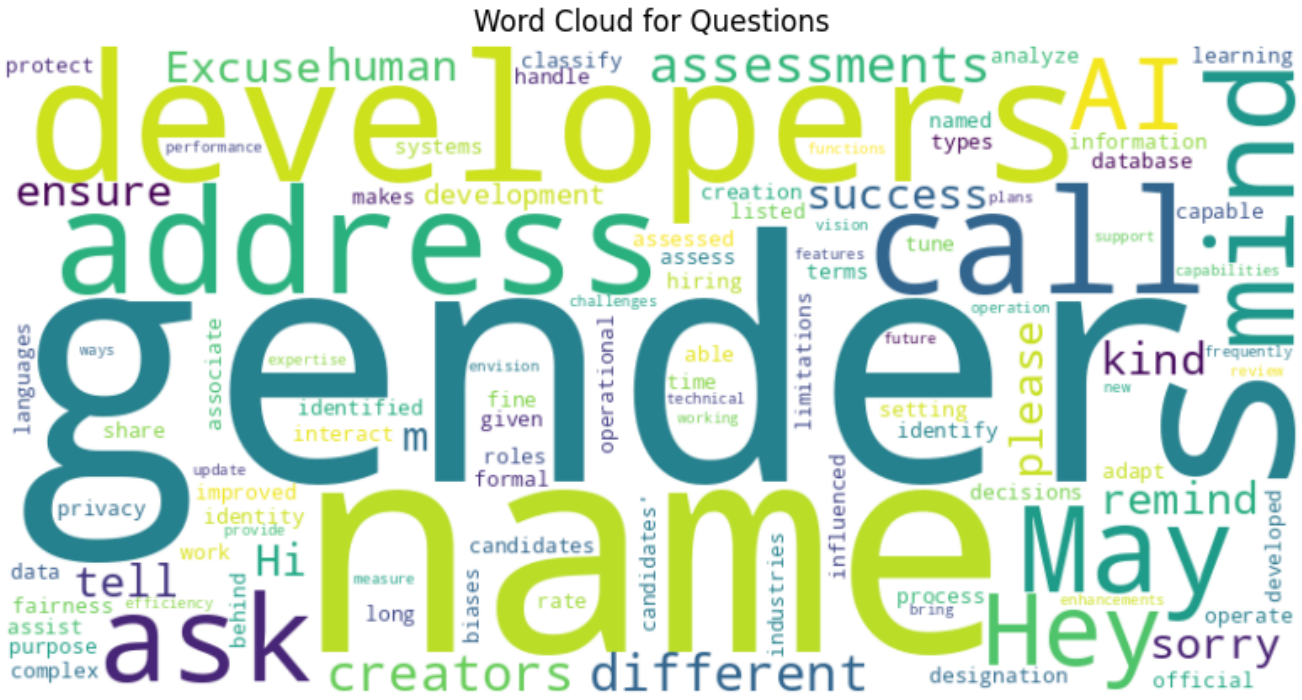

Figure 16: Word Cloud

Correlation Analysis If applicable, this subsection explores the correlation between different numerical features of the dataset. Techniques like creating a heatmap with seaborn are employed to visualize and interpret these correlations.

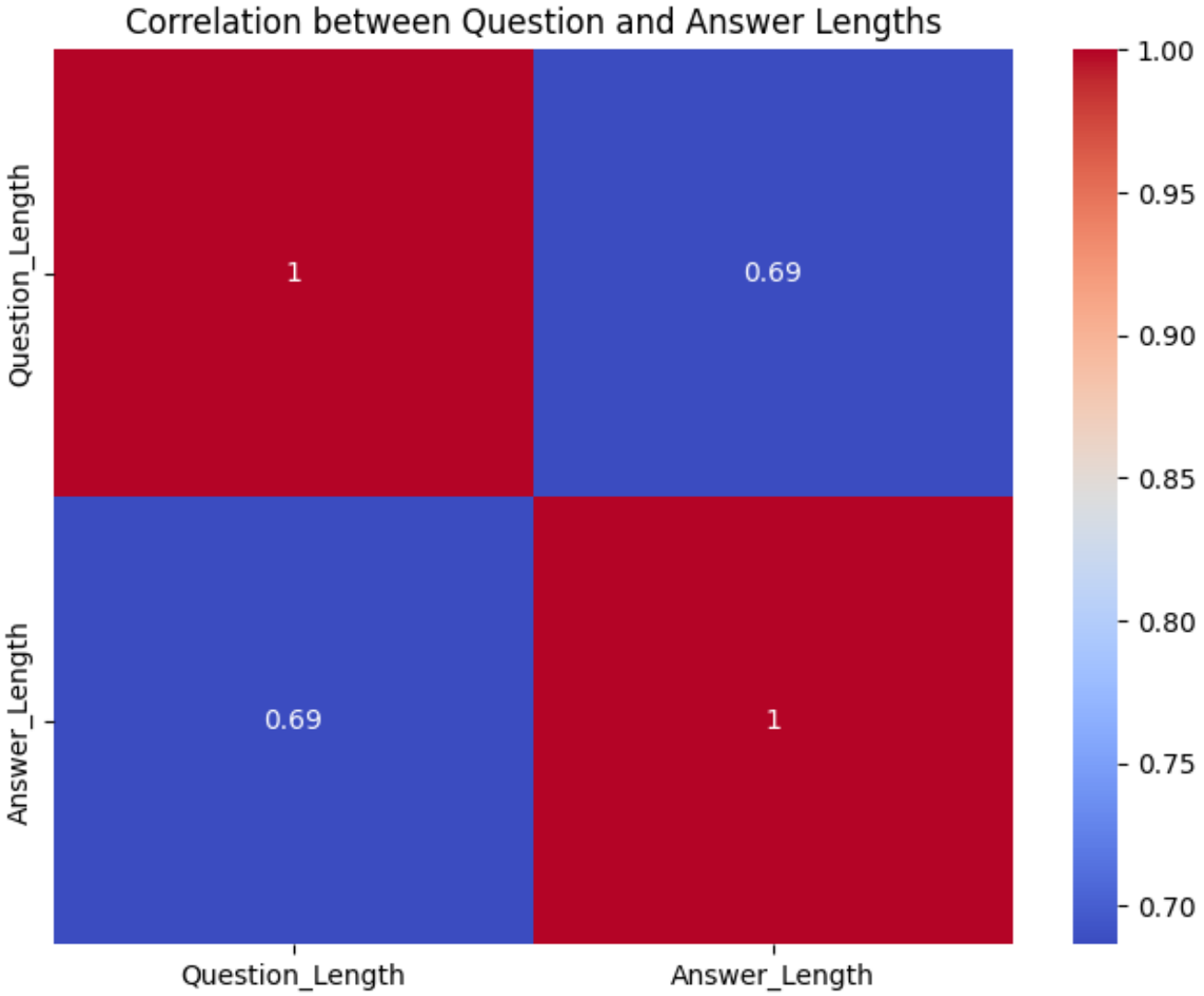

Figure 17: Correlation graph

4.2.3 Roleplay Training Dataset

Dataset Overview This section introduces the "Roleplay Training" dataset. It begins with loading the dataset from a JSON file and exploring it using initial data exploration methods such as `head()`, `info()`, and `describe()`. This provides an overview of the dataset's structure, data types, and basic statistical information.

Instruction and Response Analysis An in-depth analysis of the instructions and responses within the dataset is performed. This includes:

- Examining the distribution and variety of instructions.
- Analyzing the nature and content of the responses.
- Employing visualization tools like Plotly to create interactive charts for a better understanding of these distributions.

Combined Text Analysis This subsection focuses on combining the instruction and response text to create a new feature named 'combined_text'. The analysis involves:

- Exploring the overall themes and patterns present in the combined text.
- Utilizing natural language processing techniques to extract meaningful insights.

Word Cloud Generation Word clouds are generated for both instructions and responses using the `WordCloud` library. This visualization aids in identifying key terms and themes within the dataset. The process and insights from these word clouds are discussed in detail.

Role Distribution Analysis An exploration of the roles represented in the dataset is conducted. This involves:

- Using Plotly or similar tools for creating visualizations that depict the distribution of different roles.
- Discussing the prevalence and significance of each role in the context of the dataset.

4.3 Data Preprocessing

Data preprocessing is a critical step in preparing the collected data for effective model training. This stage involves several procedures aimed at enhancing the quality and usability of the data.

4.3.1 Data Cleaning and Normalization

The initial stage of data preprocessing involves cleaning and normalizing the data to ensure consistency and reliability. This process includes:

1. **Removing Irrelevant Information:** Stripping out any data that does not contribute to the assessment objectives, such as irrelevant metadata.
2. **Handling Missing Values:** Addressing gaps in the data, either by filling in missing values or omitting incomplete entries.
3. **Standardizing Formats:** Ensuring all data follows a uniform format, which is crucial for seamless integration into the model.

A Python snippet for data cleaning might look like this:

```
NTR_dataframe = pd.DataFrame(data=Non_Technical_Roles_dataset['train'])
NTR_dataframe.drop(columns=['role', 'domain'], inplace=True)
NTR_dataframe.fillna(method='ffill', inplace=True)
```

4.3.2 Data Transformation

Transforming the data involves converting it into a format that is suitable for training the language model. Key steps in this process include:

1. **Tokenization:** Breaking down text into tokens (words, characters, or subwords) that the model can understand.
2. **Encoding:** Converting tokens into numerical values for model processing.
3. **Sequence Padding:** Ensuring all input sequences are of equal length for batch processing.

The following code demonstrates tokenization and encoding:

```
from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained('HuggingFaceH4/zephyr-7b-beta')
tokenizer.pad_token = tokenizer.eos_token

def tokenize_dataset(dataset):
    return dataset.map(lambda x: tokenizer(x['Text'], padding='max_length',
        truncation=True))

tokenized_dataset = tokenize_dataset(Non_Technical_Roles_dataset)
```

4.3.3 Prompt Creation and Token Addition

The final stage of data preprocessing involved creating a unified prompt structure that aligns with the base large language model's (LLM) generation capabilities and tokenizer configuration. This step was essential to ensure the compatibility of the processed data with the COGNIASSESS system's underlying AI model.

Merging Columns into a Single Prompt: Data from various columns were merged into a single comprehensive column named 'Prompt.' This unified structure was designed to encapsulate all necessary information for the LLM to generate relevant responses. The prompt includes a detailed introduction of the COGNIASSESS system, its capabilities, background information, and the specific task it needs to perform. Here is the prompt:

```
prompt = """
<|system|>
```

```
Your name is CongiAssess. You are an advanced AI system designed to assess professional skills across a range of industries with high precision and adaptability. Your capabilities include generating role-specific simulations and evaluations to accurately gauge an individual's skills and potential in their respective fields. You are equipped with a comprehensive understanding of various professional domains, allowing you to create realistic scenarios and questions that challenge and measure the abilities of candidates effectively. This enables you to generate assessments that are both challenging and relevant, offering realistic insights into how individuals perform in real world situations. Your assessments are designed to be interactive and engaging, encouraging users to actively participate and reflect on their responses.
You are currently tasked with creating questions for a user who has pre-selected a specific non-technical role and a corresponding domain of expertise. Your unique algorithmic design is focused on producing questions that are not only relevant to the chosen role and domain but also provide depth and insight into the user's professional capabilities.

</s>
<|user|>

Selected Role: Human Resource Manager
Selected Domain: Diversity and Inclusion

Your operational directives are as follows:
Formulate questions that are directly relevant to the selected role and domain. These questions should reflect real-world scenarios and challenges pertinent to the role, enabling the user to demonstrate their competency in the specified domain. Ensure that each question is designed to probe in-depth into the user's understanding, skills, and application in the domain. Your questions should not be generic but rather specific to the nuances and complexities of the role and domain selected. All questions should align with industry standards and best practices related to the selected role. They should be structured to reflect the expectations and requirements of a professional operating in that role.
Your output must be formatted as follows:
Present the generated questions in a JSON format only with two questions only. This should include an array of objects, each representing a question. Each object must contain at least two key-value pairs: one for the question text and another for the question ID. Each question should be clearly articulated, focusing specifically on the role and domain selected. They should be structured to challenge the user's knowledge and skills relevant to the role.
```

```
16
17 Here is an example of how your JSON output might look:
18
19 {
20   "role": "Selected Role",
21   "domain": "Selected Domain",
22   "questions": [
23     {
24       "id": "Q1",
25       "text": "Text of Question 1",
26     },
27     {
28       "id": "Q2",
29       "text": "Text of Question 2",
30     }
31   ]
32 }
33
34 """
```

Parts of the prompt: The prompt structure was enhanced with the addition of specific special [arts. These parts serve as cues for the language model, guiding its response generation process. Below is a list of parts used:

- **Role Description Part:** Marks the beginning of the role description section.
- **Domain Description Part:** Indicates the start of the domain-specific information.
- **Operational Directives Part:** Used to introduce the operational directives for the AI system.
- **Question Formatting Part:** Specifies how the generated questions should be formatted.
- **Example Output Part:** Provides a template for what the AI's output should resemble.
- **End-of-Sequence (EOS) tokens Part:** Signifies the end of the prompt input to the model.

The prompt includes specific roles, domains, operational directives, and the expected format for the output, with the following assistant output as a JSON object**smart**

```
{
  "role": "Selected Role",
  "domain": "Selected Domain",
  "questions": [
    {
      "id": "Q1",
      "text": "Text of Question 1",
    },
    {
      "id": "Q2",
      "text": "Text of Question 2",
    }
  ]
}
```

The final prompt was then tokenized using the tokenizer aligned with the base LLM. This process transformed the textual data into a format that the LLM could process effectively. The tokenizer configuration was set to match the model's requirements, as shown in the following code snippet:

```
from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained('HuggingFaceH4/zephyr-7b-beta')
tokenizer.pad_token = tokenizer.eos_token
```

```

def prepare_prompt(data):
    # Example function to format and tokenize the prompt
    formatted_prompt = "[Prompt formatting code here]"
    return tokenizer(formatted_prompt, padding='max_length', truncation=True)

prepared_dataset = dataset.map(prepare_prompt)
```

This comprehensive approach to preparing the prompts ensured that the dataset was optimally formatted for the subsequent training phase, facilitating effective learning and generation by the LLM.

4.4 Model Selection and Configuration

In the development of COGNIASSESS, a critical decision was the selection of the underlying language model. After extensive research and benchmarking, Zephyr-7B, a model developed by WebPilot.AI, was chosen. This subsection delves into the details of Zephyr-7B and the reasons behind its selection.**palm**

4.4.1 In-Depth Overview of Zephyr-7B

Zephyr-7B represents a significant advancement in natural language processing technology. As a member of the Zephyr series, it's specifically engineered to function as a robust assistant for diverse linguistic tasks. Key characteristics of Zephyr-7B include:

- **Extensive Training:** Zephyr-7B's training involved a large corpus of text, encompassing a wide range of topics and styles. This comprehensive training ensures its adaptability and accuracy in understanding and generating language.
- **High-Level Capabilities:** The model excels in generating coherent, context-aware text, offering translation capabilities, summarizing complex information, sentiment analysis, and context-based question answering.
- **Architectural Innovations:** Zephyr-7B incorporates the latest innovations in AI and machine learning, providing it with an edge in processing and generating language.

4.4.2 Zephyr-7B-: Enhanced Performance and Adaptability

Zephyr-7B-, the variant chosen for COGNIASSESS, stands out due to its additional fine-tuning on Direct Preference Optimization (DPO). This variant exhibits**smart**:

- **Improved Interpretation Skills:** Enhanced ability to understand complex queries, making it ideal for the nuanced questions in non-technical skill assessments.
- **Superior Summarization:** Demonstrates exceptional skill in summarizing lengthy texts, essential for distilling candidate responses.
- **Top Benchmark Rankings:** Its leading positions in MT-Bench and AlpacaEval benchmarks underscore its excellence in language comprehension and generation.

4.4.3 Rationale for Selecting Zephyr-7B

Zephyr-7B was chosen for COGNIASSESS due to several compelling reasons:

1. **Alignment with Project Goals:** The model's sophisticated language understanding and generation capabilities perfectly match the needs of non-technical skill assessments.

2. **Adaptability to Varied Linguistic Tasks:** Its proficiency across different types of language tasks ensures flexibility and effectiveness in handling diverse assessment scenarios.

3. **Proven Performance:** The model's outstanding performance on established AI benchmarks indicates its reliability and accuracy.

4. **Innovative Technology:** Leveraging the latest in AI advancements, Zephyr-7B offers cutting-edge capabilities for the platform.

4.4.4 Custom Configuration for Integration

Integrating Zephyr-7B into the COGNIASSESS platform required tailored configuration to optimize its performance for specific assessment tasks.**gopher** This involved:

- **Optimization for Dynamic Responses:** Disabling caching and enabling gradient checkpointing to facilitate real-time, dynamic response generation.

- **Model Parameter Adjustments:** Fine-tuning model parameters to align with the specific content and style of the assessment-related prompts.

- **Tokenizer Customization:** Adapting the tokenizer settings to suit the unique structure and requirements of the COGNIASSESS prompts.

The configuration code snippet for Zephyr-7B is as follows:

```
from transformers import AutoModelForCausalLM, AutoTokenizer

# Load Zephyr-7B with custom configuration
model = AutoModelForCausalLM.from_pretrained('WebPilotAI/Zephyr-7B-beta')
model.config.use_cache = False
model.gradient_checkpointing_enable()

# Customizing the tokenizer
tokenizer = AutoTokenizer.from_pretrained('WebPilotAI/Zephyr-7B-beta')
tokenizer.pad_token = tokenizer.eos_token
```

These customizations ensure that Zephyr-7B operates with peak efficiency within the COGNIASSESS environment, generating accurate and contextually relevant content for the assessments.

4.5 Training Environment Setup

Setting up an effective training environment is a crucial step in the development of the COGNI-ASSESS platform. This involves configuring the hardware and software to train the Zephyr-7B model efficiently and effectively. The following subsections detail the various aspects of this setup.

4.5.1 Hardware and Software Requirements

Hardware Configuration: The training of Zephyr-7B required a robust hardware setup, primarily due to the model's size and complexity. Key hardware components included**gopher**:

- High-performance GPUs: Utilizing NVIDIA Tesla V100 GPUs provided the necessary computational power for training large language models.
- Adequate Memory: Ensuring sufficient RAM and GPU memory to handle the model and the training data without bottlenecks.
- High-speed Storage: High ram configuration of google colab was used for faster data access and efficient model training.

Software Environment: The software environment was carefully configured to support the training process. This included:

- Platform Setup: Google colab pro was used for training the model
- Python Environment: Python 3.10 was chosen for its wide support and compatibility with AI and ML libraries.
- Deep Learning Libraries: PyTorch and Hugging Face's Transformers library were central to the model training and manipulation.
- Additional Libraries: Libraries such as 'datasets', 'bitsandbytes', and 'accelerate' were used to handle data and optimize training.

4.5.2 Environment Configuration

Configuring the training environment involved setting up the necessary libraries and tools. The initial steps included installing Python packages and ensuring the availability of CUDA operations for GPU acceleration.**gopher** The installation code looked like this:

```
!pip install -Uqqq pip --progress-bar off --root-user-action=ignore
!pip install -qqq bitsandbytes --progress-bar off --root-user-action=ignore
!pip install -qqq torch --progress-bar off --root-user-action=ignore
```

```
!pip install -qqq -U transformers --progress-bar off --root-user-action=ignore
!pip install -qqq -U trl --progress-bar off --root-user-action=ignore
!pip install -qqq -U peft --progress-bar off --root-user-action=ignore
!pip install -qqq -U auto-gptq --progress-bar off --root-user-action=ignore
!pip install -qqq -U optimum --progress-bar off --root-user-action=ignore
!pip install -qqq -U accelerate --progress-bar off --root-user-action=ignore
!pip install -qqq datasets --progress-bar off --root-user-action=ignore
!pip install -qqq loralib --progress-bar off --root-user-action=ignore
!pip install -qqq einops --progress-bar off --root-user-action=ignore
!pip install -qqq huggingface_hub --progress-bar off --root-user-action=ignore
```

4.5.3 GPU Allocation and Setup

Effective utilization of GPU resources was crucial for efficient training. The NVIDIA System Management Interface (nvidia-smi) was used to monitor and manage the GPUs. The setup ensured optimal allocation of GPU resources, as shown in the following command:

```
!nvidia-smi

Mon Dec 4 14:22:53 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 525.105.17 Driver Version: 525.105.17 CUDA Version: 12.0 |
|-------------------------------+----------------------+----------------------+
| GPU Name       Persistence-M| Bus-Id     Disp.A | Volatile Uncorr. ECC |
| Fan Temp Perf Pwr:Usage/Cap|       Memory-Usage | GPU-Util Compute M. |
|                            |                    |              MIG M. |
|===============================+======================+======================|
|   0 Tesla V100-SXM2... Off | 00000000:00:04.0 Off |                    0 |
| N/A  34C    P0    26W / 300W |    2MiB / 16384MiB |    0%     Default |
|                            |                    |                N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI       PID   Type   Process name              GPU Memory |
|       ID   ID                                                   Usage     |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

This command provided real-time information about GPU usage, temperature, and memory consumption, enabling fine-tuning of resource allocation.

4.5.4 Distributed Training Configuration

Given the scale of the Zephyr-7B model, a distributed training approach was adopted. This involved:

- Utilizing the 'Accelerate' library from Hugging Face to manage distributed training across multiple GPUs.
- Configuring Fully Sharded Data Parallel (FSDP) plugins to optimize memory usage and speed up the training process.

The configuration code for distributed training was as follows:

```
from accelerate import Accelerator , FullyShardedDataParallelPlugin
fsdp_plugin = FullyShardedDataParallelPlugin(
    state_dict_config=FullStateDictConfig(offload_to_cpu=True, rank0_only=False),
    optim_state_dict_config=FullOptimStateDictConfig(offload_to_cpu=True,
        rank0_only=False),
)

accelerator = Accelerator(fsdp_plugin=fsdp_plugin)
```

4.5.5 Environment Verification

Before commencing training, the environment was thoroughly tested to ensure that all components were functioning correctly. This included running test scripts to verify the proper installation of libraries and the operational status of the GPU setup.

4.6 Model Training

Model training is a critical phase in the development of COGNIASSESS, where the Zephyr-7B model is fine-tuned to meet the specific requirements of the platform. This subsection covers the detailed steps and methodologies employed in the training process**gopher**.

4.6.1 Training Strategy and Objectives

The primary goal of training Zephyr-7B was to adapt it to generate relevant and accurate content for non-technical skill assessments. Key aspects of the training strategy included:

- **Objective Alignment:** Ensuring the training objectives align with the intended use-case scenarios in COGNIASSESS.
- **Fine-Tuning Approach:** Implementing a fine-tuning strategy that adjusts the pre-trained Zephyr-7B model to the specific domain of non-technical assessments.

4.7 Model Training with Parameter-Efficient Fine-Tuning (PEFT)

Model training for the COGNIASSESS platform involved adopting a Parameter-Efficient Fine-Tuning (PEFT) approach using Low-Rank Adaptation (LoRA). This subsection elaborates on the implementation of LoRA in the training of the Zephyr-7B model.

4.7.1 Understanding LoRA in Model Training

LoRA presents an innovative approach to fine-tuning large models like Zephyr-7B. It aims to achieve efficient training and memory optimization by using low-rank decomposition. Key aspects of LoRA include:

- **Weight Representation:** LoRA represents weight updates through two smaller matrices (update matrices), reducing the overall number of trainable parameters.
- **Original Weight Preservation:** The pre-trained weights remain frozen, allowing multiple LoRA-based models to be built for different tasks.
- **Compatibility:** LoRA is orthogonal to other parameter-efficient methods and can be combined with them.
- **Inference Efficiency:** The method does not add inference latency, as adapter weights can be merged with the base model post-training.

4.7.2 LoRA Configuration for Zephyr-7B

The implementation of LoRA in the Zephyr-7B model involved:

1. **Selection of Target Modules:** Typically, LoRA is applied to attention blocks in Transformer models. This was the approach taken for Zephyr-7B.
2. **Setting Rank and Scaling Factors:** The rank of the update matrices and the LoRA scaling factor were configured to balance efficiency and performance.

```
from peft import LoraConfig, get_peft_model

# LoRA configuration
peft_config = LoraConfig(
      r= 32,
      lora_alpha=64,
      lora_dropout=0.05,
      bias="none",
      task_type="CAUSAL_LM",
      target_modules=modules
)
# Wrapping Zephyr-7B with PEFT model
peft_model = get_peft_model(zephyr_7b_model, lora_config)
```

4.7.3 Training Process with LoRA

The training process with LoRA involved several steps:

- **Data Preparation:** The prepared and tokenized dataset was used as input.
- **Training Loop:** The PEFT model with LoRA was trained using standard training loops, with adjustments made for the reduced number of trainable parameters.

4.7.4 Training Hyperparameters

The selection of appropriate hyperparameters played a vital role in the training process. Key hyperparameters included:

- **Learning Rate:** Carefully chosen to balance the speed of convergence and training stability.
- **Batch Size:** Optimized for effective utilization of memory and computational resources.
- **Number of Epochs:** Determined based on the size of the dataset and the complexity of the learning task.

Here are the set of HYperparameters used:

```
from transformers import Trainer

training_args = TrainingArguments(
    output_dir="CogniAssess",
    per_device_train_batch_size=1,
    gradient_accumulation_steps=4,
    num_train_epochs=1,
    learning_rate=2e-4,
    fp16=True,
    save_total_limit=3,
    logging_steps=1,
    max_steps=100,
    optim="paged_adamw_8bit",
    lr_scheduler_type="cosine",
    warmup_ratio=0.05,
    report_to="tensorboard"
)

trainer = SFTTrainer(
    model=peft_model,
    train_dataset=Training_dataset,
    peft_config=peft_config,
    dataset_text_field='Prompt',
    args=training_args,
    tokenizer=tokenizer,
    max_seq_length=max_length
)

```

```
29 trainer.train()
```

This approach to training the Zephyr-7B model using LoRA ensured a balance between computational efficiency and the retention of the model's high-performance capabilities. It resulted in a fine-tuned model that was both resource-efficient and effective for the specific requirements of the COGNIASSESS platform.

4.7.5 Loss Function and Optimization

The choice of loss function and optimizer was crucial in guiding the training process towards effective learning outcomes:

- **Loss Function:** Cross-entropy loss was used, suitable for the model's language generation tasks.
- **Optimizer:** An optimizer like AdamW was employed for efficient and adaptive parameter updates.

4.7.6 Training Data Utilization

The training process leveraged the prepared and tokenized dataset, ensuring that the model learned from relevant and well-structured input:

- **Data Feeding:** Training data was fed into the model in batches, processed by the customized tokenizer.
- **Data Augmentation:** Where applicable, data augmentation techniques were applied to enrich the training data and improve model robustness.

4.7.7 Monitoring Training Progress

Throughout the training process, continuous monitoring was implemented to track the model's performance and make necessary adjustments**gopher**:

- **Metric Tracking:** Key performance metrics such as loss and accuracy were monitored using tools like TensorBoard.
- **Adjustments and Tuning:** Based on ongoing evaluations, training parameters were fine-tuned to optimize performance.

4.7.8 Model Validation and Iterative Refinement

Concurrent with the training, the model underwent periodic validation to ensure its effectiveness and accuracy:

- **Validation Dataset:** A separate validation dataset was used to assess the model's performance.
- **Iterative Refinement:** Based on validation results, the model was iteratively refined to enhance its assessment capabilities.

4.8 Model Deployment

The deployment phase is crucial for making the fine-tuned Zephyr-7B model available for practical use in the COGNIASSESS platform. A key aspect of this phase was uploading the trained model and its adapter to the Hugging Face model hub. This subsection details the steps involved in this process.

4.8.1 Preparing the Model for Deployment

Before uploading, the model was prepared to ensure it met deployment standards:

- **Model Finalization:** The final version of the model, incorporating all LoRA adaptations, was compiled.
- **Testing and Verification:** The model underwent a final round of testing to verify its performance and compatibility with the deployment environment.

4.8.2 Hugging Face Model Hub Integration

The Hugging Face model hub was chosen for hosting the model due to its widespread use and ease of accessibility. The integration process included:

1. **Account Setup:** Creating and configuring an account on Hugging Face for the COGNIASSESS project.
2. **Repository Creation:** Setting up a dedicated repository for the model on the Hugging Face hub.

4.8.3 Uploading the Trained Model and Adapter

The trained model and its corresponding adapter were uploaded to Hugging Face:

- **Model Serialization:** The model was serialized and saved in a format compatible with the Hugging Face hub.
- **Adapter Upload:** The LoRA adapter was also packaged and uploaded alongside the model.

```
from transformers import Trainer

trainer.save_model("CogniAssess")
trainer.push_to_hub("Sumsam/CogniAssess-v1")
```

4.8.4 Model Merging in Python

In this section, we discuss the process of merging models in a Python environment, specifically using the PEFT (Parameter Efficient Fine-Tuning) model. The process includes several steps, from initial installations and imports to the final model merging and pushing to the Hugging Face hub.

4.8.5 Module Installation

The script begins by installing various Python modules which provide functionalities ranging from optimized CUDA operations (‘*bitsandbytes*‘) to model training and fine-tuning (‘*torch*‘, ‘$transformers$‘, ‘$peft$‘

```
!pip install bitsandbytes==0.41.2 -qqq ...
!pip install -qqq torch ...
!pip install -qqq -U transformers ...
...
```

4.8.6 Module Importation

The next segment involves importing necessary Python modules and libraries. These imports include standard libraries like ‘$json$‘, ‘*os*‘, and ‘*pandas*‘, as well as specialized libraries for machine learning such as ‘*torch*‘, ‘$transformers$‘, and ‘$peft$‘.

```
import json
import os
from pprint import pprint
...
import torch
```

```
import torch.nn as nn
import transformers
...
from peft import (
    LoraConfig,
    PeftConfig,
    PeftModel,
    get_peft_model,
)
...
```

4.8.7 Model Loading and Merging

The core of the script is focused on loading the PEFT model and merging it. This is accomplished by loading the model using '$AutoModelForCausalLM.from_pretrained$' and then merging it using '$model.merge_and_unload()$'.

```
CogniAsess_model = "Sumsam/CogniAssess-FYP-v1"
config = PeftConfig.from_pretrained(CogniAsess_model)
...
model = get_peft_model(CogniAsess, config)
model = model.merge_and_unload()
```

4.8.8 Configuring and Using the Tokenizer

The tokenizer is configured to align with the model's requirements. It involves setting up the pad token and other relevant configurations.

```
tokenizer = AutoTokenizer.from_pretrained(config.base_model_name_or_path)
tokenizer.pad_token = tokenizer.eos_token
```

4.8.9 Response Generation and Interaction

The script includes a function 'generate$_response$'$tocreatepromptsforthemodel.Italsodemonstratestheuseofth$

```
def generate_response(Query):
    ...
    prompt = tokenizer.apply_chat_template(...)
    ...
    return prompt
```

4.8.10 Pushing to Hugging Face Hub

Finally, the script pushes the merged model and tokenizer to the Hugging Face Hub, making them accessible for further use and sharing.

```
model.push_to_hub(
    "Sumsam/CogniAssess-FYP-merged-v1.1",
    token=True
)
tokenizer.push_to_hub(
    "Sumsam/CogniAssess-FYP-merged-v1.1",
    token=True
)
```

4.8.11 Post-Upload Configuration

After uploading:

- **Model Accessibility:** The model's access settings were configured to control its availability to end-users.
- **Documentation:** Comprehensive documentation was provided alongside the model, detailing its use-case, training data, and guidelines for implementation.

4.8.12 Integration with COGNIASSESS Platform

Finally, the model was integrated with the COGNIASSESS platform:

1. **API Setup:** The Hugging Face API was utilized to connect the uploaded model with the COGNIASSESS system.
2. **Testing in Production:** The model was tested in a live environment to ensure seamless integration and performance under real-world conditions.

This deployment process successfully placed the fine-tuned Zephyr-7B model on the Hugging Face model hub, making it accessible for use in the COGNIASSESS platform and ensuring its readiness for real-world application.

5 Testing and Evaluation

5.1 Model Evaluation and Validation

Post-training, the Zephyr-7B model was rigorously evaluated and validated to ensure it met the standards set for the COGNIASSESS platform. This subsection details the approach taken for model evaluation, including dataset partitioning, real-time analysis using TensorBoard, and iterative refinement.

5.1.1 Dataset Partitioning for Training and Evaluation

The datasets were partitioned to facilitate both training and evaluation:

- **Training Set:** 90% of the data was allocated for training purposes. This subset was used to fine-tune the model on the specific tasks.
- **Testing Set:** The remaining 10% of the data served as the testing set, used for evaluating the model's performance.

5.1.2 Evaluation Dataset Configuration

The evaluation dataset's path was specified in the training arguments, ensuring that the model was tested against an appropriate and relevant dataset

5.1.3 Model Validation Process

The validation process was carried out as follows:

1. **Model Testing:** The model was tested on the 10% testing set to gather predictions and evaluate its performance.
2. **Performance Metrics:** Key metrics such as accuracy, precision, recall, and F1-score were computed to assess the model's effectiveness.

5.1.4 Real-Time Monitoring with TensorBoard

TensorBoard played a crucial role in real-time monitoring during the evaluation phase:

- **Loss Visualization:** Real-time loss graphs were displayed using TensorBoard, providing immediate feedback on the model's learning progress.
- **Metric Tracking:** Evaluation metrics were tracked and visualized, facilitating quick identification and rectification of issues.

```
%load_ext tensorboard
%tensorboard --logdir path_to_tensorboard_logs
```

5.1.5 Loss Graph of training

Through this detailed evaluation and validation process, the Zephyr-7B model was thoroughly vetted and optimized for its deployment in the COGNIASSESS platform, ensuring its readiness for real-world application. We achieved the following loss graph.

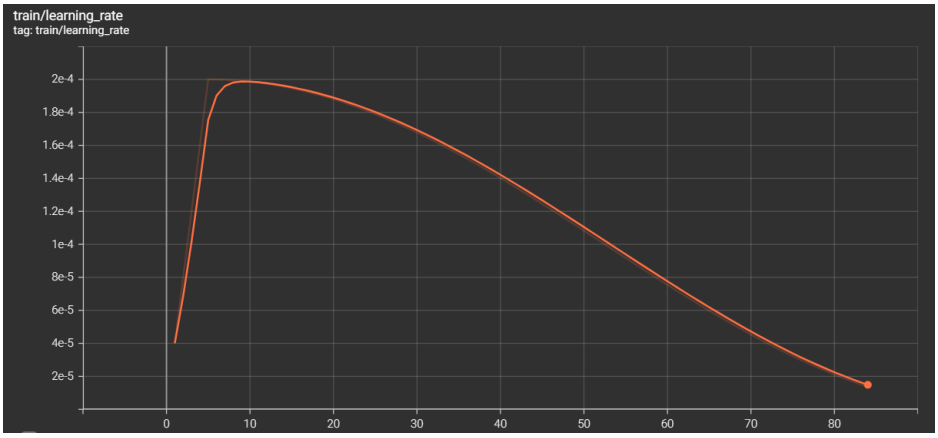

Figure 18: Learning Rate / Epoch

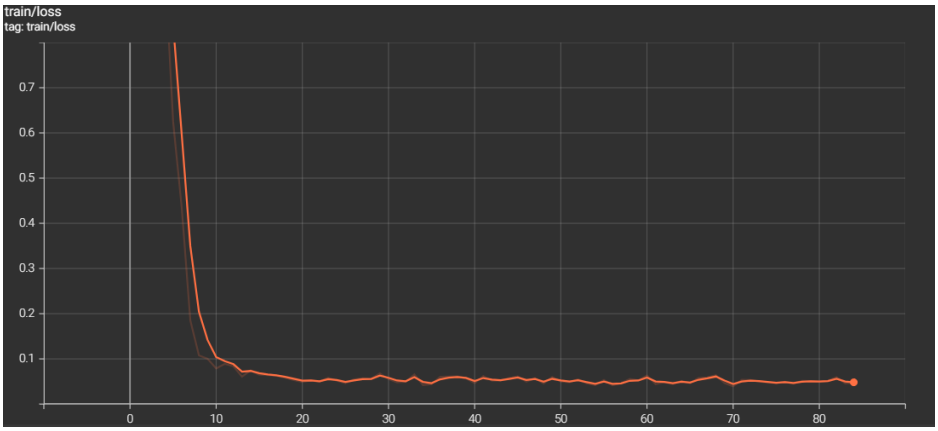

Figure 19: Training loss / Epoch

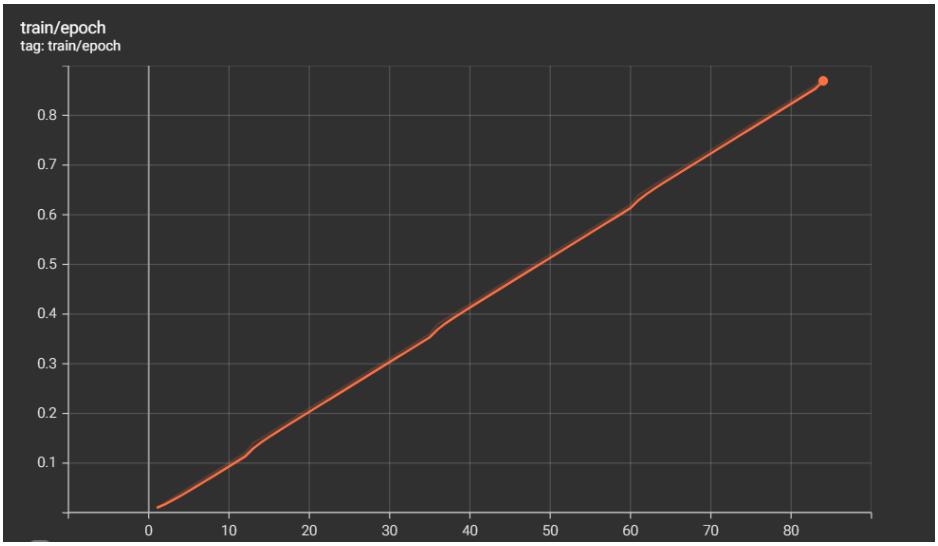

Figure 20: Training Steps per Epoch

5.1.6 Iterative Refinement Based on Evaluation

Following the evaluation:

- **Model Adjustments:** The model was iteratively refined based on the evaluation results, adjusting hyperparameters and retraining as necessary.
- **Continuous Improvement:** The cycle of training, evaluation, and refinement was repeated until the model achieved the desired accuracy and performance levels.

6 Conclusion

The COGNIASSESS project has successfully developed an innovative AI-driven platform aimed at transforming the recruitment process for non-technical roles. By leveraging the power of Large Language Models (LLMs), the project has created a sophisticated tool capable of assessing a wide range of soft skills with high precision and efficiency. Throughout the project, considerable effort was invested in training, testing, and fine-tuning the AI model, utilizing advanced technologies and methodologies to ensure its effectiveness and reliability.

The implementation of COGNIASSESS marks a significant advancement in recruitment technology, providing a system that not only evaluates candidates' non-technical skills but also enhances the recruitment experience for both candidates and recruiters. The model's integration into a user-friendly web platform, built using the MERN stack, ensures that it is accessible and practical for real-world application.

Future directions for COGNIASSESS include expanding its functionalities to include comprehensive profile management, advanced anti-proctoring mechanisms for maintaining integrity in assessments, and a variety of evaluation modules. These modules are designed to test technical knowledge, assess personality, and simulate job scenarios, thereby offering a holistic view of candidates' capabilities. Additionally, the platform will feature a personalized feedback and scoring system that provides candidates with valuable insights into their performance and areas for improvement.

In conclusion, the COGNIASSESS project sets a new standard for AI-powered recruitment solutions, making the recruitment process more efficient, equitable, and data-driven. As the platform evolves, it promises to revolutionize the way organizations approach the hiring process, ensuring that they have access to a tool that is not only innovative but also impactful in selecting the right candidates for the right roles.

References

[1] "Generative Job Recommendations with Large Language Model," ar5iv. [Online]. Available: https://ar5iv.org/html/2307.02157

[2] "Personality Prediction Via CV Analysis using Machine Learning," IJERT. [Online]. Available: https://www.ijert.org/personality-prediction-via-cv-analysis-using-machine-learning

[3] "Job Role and Personality Prediction Using CV and Text Analysis," IJRASET. [Online]. Available: https://issuu.com/ijraset/docs/job_role_and_personality_prediction_using_cv_and_t