FINAL PROJECT REPORT

11 / 30 / 2024

PREAPARED BY :

- VERSO Group No. 04
  - **Thamer S. Almalki**
  - **Mahmued A. Alardawi**
  - **Rawad A. Alghamdi**

King Abdulaziz University
Faculty of Computing and Information Technology Department of Computer Science.

CPCS-499 Senior Project, Supervised By:
Dr. Mohammed Dahab

DECLARATION OF AUTHORS

We hereby declare that this project report titled "VERSO" is the result of our original work, and We confirm that:

1. This report, submitted to the Faculty of Computing and Information Technology at King Abdulaziz University, is the result of our own research and investigations except where otherwise identified by references and that we have not plagiarized another's work. It is conducted under the supervision of Dr. Mohammed Dahab, Department of Computer Science.
2. We have not submitted this work or any portion of it to any other institution for any academic degree or qualification.

Thamer S. Almalki   Rawad A. Alghamdi   Mahmued A. Alardawi

[illegible]   [illegible]   [illegible]

DECLARATION OF SUPERVISOR

I, the undersigned, certify that I have reviewed this project report and approve it, recommending that the authors submit it to the final year project evaluation committee for final assessment and presentation. This report is submitted in partial fulfillment of the requirements for the degree of BS Computer Science at the Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University.

Dr. Mohammed Dahab

ABSTRACT

The VERSO project, initiated by the Computer Science Department at King Abdulaziz University and supervised by Dr. Mohammed Dahab, seeks to redefine the creation and experience of Arabic poetry through artificial intelligence. Developed by Thamer, Mahmued, and Rawad, VERSO is an AI-powered web application that generates and preserves Arabic poetry, blending linguistic artistry with computational power.

This platform respects the cadence of Arabic metrics while bringing poetry to life through visual art and allowing users to audibly experience their verses with virtual avatars. Within the CPCS-498 course framework, this report covers the project's goals, methodology, and challenges in merging AI with the intricacies of Arabic prosody. VERSO aims to connect the past with the future by celebrating Arabic poetry in the digital age.

مشروع "فيرسو"، الذي نشأ في قسم علوم الحاسب في جامعة الملك عبد العزيز تحت إشراف الدكتور محمد ذهب، يسعى إلى تنغيم تجربة الشعر العربي باستخدام الذكاء الاصطناعي وإحياء مصابيحه المطفأة. طوره أعضاء الفريق، ثامر ومحمود ورواد، وهو تطبيق ويب مدعوم بالذكاء الاصطناعي، من أجل الابتكار والحفاظ على روحية الشعر العربي، ودمج الفن اللغوي مع القوة الحاسوبية.

يعزز هذا التطبيق إيقاع اللغة العربية، ويجسد الشعر من خلال تأويله بصريا، ويتيح للمستخدمين الاستماع إلى قصائدهم عبر تجسيدها افتراضيا. ضمن إطار عمل مقرر CPCS-498، يتناول هذا التقرير أهداف المشروع ومنهجيته وتحديات تطويع الذكاء الاصطناعي لتعقيدات الشعر العربي. يهدف "فيرسو" إلى ربط الماضي بالمستقبل من خلال الاحتفال بالشعر العربي في العصر الرقمي.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I: Introduction

1.1 Introduction

This chapter provides the foundational background of the VERSO project, detailing the objectives and the problem definition. It explores the significance of AI in creative domains and how VERSO seeks to transform the landscape of Arabic poetry. We will discuss the project's vision and rationale, why this work is important, and the goals we aim to achieve.

1.2 Project Background

The Understanding and creation of poetry requires deep knowledge of the rules of Arabic prosody (AL-Arood). The structure complexity of these linguistics makes Arabic poetry a challenging art to implement digitally. The VERSO project will redefine how Arabic poetry is both created and experienced. This visionary approach addresses the unexplored intersection between Arabic literature and AI, aiming to craft a platform that will generate and immortalize Arabic poetry.

1.3 Problem Definition

In the field of computational linguistics, the generative capacity of artificial intelligence for poetic construction remains unexplored, particularly in the Arabic language with its complex prosodic systems. VERSO addresses this non-trivial problem by aiming to create a comprehensive AI system that generates authentic Arabic poetry.

Key Issues

- **Lack of Technological Integration**: Current digital offerings do not fully capture the essence of Arabic poetic forms.
- **Inadequate Representation**: There is a scarcity of platforms that support the creation and analysis of Arabic poetry using AI.
- **Limited Accessibility**: People interested in Arabic poetry and its creative process have few technological tools at their disposal

1.4 Project Aims

- To construct an AI framework that composes traditional Arabic poetry from user prompts, emphasizing adherence to the art form's classical metrics.
- To implement a visual transformation process that converts composed verses into bespoke artistic representations, enhancing the user's sensory experience.
- (optional) To integrate deepfake And Holographic technology, allowing users to generate personalized audiovisual recitations of their AI-crafted poems.

1.5 Project Scope

Inclusions

- **AI-Powered Arabic Poetry Generation**: Development of a web-based platform that uses AI to generate Arabic poetry from user inputs.
- **Prosody Analysis**: Development of an algorithm for analyzing the meter of Arabic poems to maintain adherence to traditional rules.
- **Artistic Representations**: Integration of a text-to-image AI model to transform generated poems into visual artworks.
- **(Optional) Personalized Recitations**: Implementation of deepfake technology to allow users to superimpose their faces and voices onto avatars for poem recitation. This includes both user-generated content and pre-trained options.
- **(Optional) Holographic Representation**: Creating a feature that uses holographic technology to represent the avatar created by the deepfake technology.

Exclusions

- The development of tools for non-Arabic languages.
- Integration with other forms of literature beyond poetry
- Real-time collaborative features for poetry writing or editing.
- An extensive library of pre-written poetry for analysis beyond user-generated content.

1.6 Proposed Challenges and Solutions

- **Data Acquisition and Quality**
  - **Description**: Obtaining a large and high-quality dataset of Arabic poetry for training the AI model.
  - **Solution**: Collaborate with Arabic literature scholars to curate a comprehensive dataset and use data augmentation techniques to enhance the quality and volume of training data.

- **Language Complexity**
  - **Description**: Addressing the linguistic intricacies of Arabic poetry, such as metaphors, rhymes, and prosody.
  - **Solution**: Employ advanced NLP techniques and consult with experts in Arabic prosody to ensure the AI model can accurately capture these complex elements.

- **Integration**
  - **Description**: Seamlessly integrating various AI technologies (NLP, text-to-image, deepfake) for a cohesive user experience.
  - **Solution**: Adopt a modular architecture to facilitate smooth integration and allow for individual component testing before full system assembly.

- **User Engagement**
  - **Description**: Ensuring the platform is engaging and meets the needs of a diverse user base.
  - **Solution**: Implement user-centered design principles, conduct usability testing, and gather user feedback for continuous improvement.

- **Ethical Considerations**
  - **Description**: Mitigating risks associated with deepfake technology, such as misuse for impersonation or deception.
  - **Solution**: Establish clear ethical guidelines, user agreements, and implement detection mechanisms for any misuse of the technology.

- **Platform Scalability**
    - **Description**: Ensuring the platform can manage a growing number of users and generated content.
    - **Solution**: Design the system for scalability from the outset, using cloud services and scalable databases to accommodate increased traffic and data storage needs.

- **Model Accuracy and Bias**
    - **Description**: Overcoming potential biases in the AI model and ensuring accurate
    - representation of various Arabic poetry styles.
    - **Solution**: Diversify the training dataset, perform regular model evaluations, and fine-tune the model to minimize biases and inaccuracies.

- **Technical Limitations**
    - **Description**: Navigating limitations of current AI and NLP technologies in capturing the full artistic essence of Arabic poetry.
    - **Solution**: Focus on iterative development and remain adaptable to incorporate new advancements in AI as they arise.

1.7 Conclusion

The VERSO project merges AI with Arabic poetry, creating a platform for generating and experiencing poetry in novel ways. It displays our dedication to enhance the cultural heritage through technology, offering a fresh perspective on traditional Arabic literary form.

CHAPTER II: Information Gathering and Literature Review

2.1 Introduction

In this chapter, we delve into previous work and studies relevant to AI in poetry generation, especially in the context of Arabic literature. By reviewing the latest developments and academic contributions in this domain, we identify the gaps and challenges that VERSO seeks to address. The literature review provides a comprehensive understanding of existing tools, and the artistic potential of AI.

2.2 Methods to Find Information

Literature Review

- **Strengths**

  - **Comprehensive Knowledge**: Literature reviews provide a broad overview of existing research, including different approaches and challenges associated with AI, NLP, and Arabic poetry. This wide scope helps us understand the current landscape and its complexities.

  - **Diverse Perspectives**: Offers access to various viewpoints, for a balanced understanding of the subject matter.

  - **Cost-Effectively**: Academic journals and publications are often easy to get at university libraries or online databases for less or no cost at all.

- **Weaknesses**
  - **Time Eating**: Filtering through vast amounts of literature could be time-intensive, needed efficient search and filtering strategies.
  - **Potential for Outdated Information**: Research in AI and NLP evolves very, very quickly! So, some literature does not reflect the latest advancements or techniques.
  - **Limited Practical Insights**: Academic articles might not always provide practical guidance on how to implement, particularly for complex, and detailed projects like VERSO.

Expert Consultation

- **Strengths**
  - **Up-to-date and Relevant Knowledge**: Experts offer access to the recent advancements. Current trends, and the best practices in their respective fields, often beyond what's found in published journals, papers.
  - **Customized Advice**: Experts can provide certain guidance and solutions, fitting to the unique challenges and targets of VERSO.
  - **Opportunities for Networking**: Building relationships with experts can eventually lead to continuous collaboration, support, and access to valuable resources.

- **Weaknesses**
  - **Availability & Cost**: Experts have limited time, and charge high, required careful planning, budget arrangement.
  - **Possible Bias**: Experts might carry their own ideas or preferences that could influence their advice. Making critical evaluation of their recommendations necessary.

Most Effective Approach

For the project VERSO, an approach combining both yet giving priority to expert consultation is recommended. Even though a complete literature survey is important for building a foundational grasp; expert consultation provides some key benefits:

- **Cultural Layers of Arabic Poetry**: Experts specialized in Arabic prosody and poetry, they can give deep insights about the cultural and linguistic subtleties-critical for generating authentic poetry,
- **Expertise that's Technical**: Experts in AI, NLP and connected technologies can suggest the latest developments.
- **Implementation Practically**: Experts capable of giving advice valuable for overcoming Hardships in implementation, optimizing performance of the model, integrating various AI components effectively.

In conclusion, both methods are valuable and all, however, giving priority to expert consultation ensures access to the current, relevant, and practical knowledge very much needed for navigating the complexities of integrating AI with Arabic poetry generation.

2.3 Poetry Generation

Poetry generation through computational means has been an area of increasing research over the past decade. Early attempts relied on rule-based systems to generate simple verses with predefined structures. With the advancement of machine learning (ML) and natural language processing (NLP), poetry generation has advanced significantly:

- **Rule-Based Systems:** These early systems used predefined grammatical structures and vocabulary to generate poetic text. However, they lacked the flexibility needed to capture the poetic meter and rhyme.
- **Machine Learning Models**: Recent advances in deep learning, particularly with transformer architectures like GPT-4, have advanced the way for more sophisticated poetry generation. These models can analyze large datasets of text and replicate patterns with higher fluency and accuracy.

Despite these advances, the application of such models to Arabic poetry remains underexplored due to its unique linguistic structure

2.4 Arabic Prosody Analysis

Arabic prosody, or (Al-Arood), governs the meter and rhythm of traditional Arabic poetry. Adherence to these rules requires an understanding of complex linguistic patterns. In recent years, computational linguistics research has attempted to codify these rules for digital analysis:

- **Manual Rules Implementation:** Studies have worked on implementing Al-Arood rules directly through code, creating prosody analysis tools capable of verifying meter and rhyme adherence.
- **Machine Learning for Prosody Analysis:** With machine learning models, researchers have automated prosody analysis, enabling identification of meter patterns across large datasets. Such tools, however, still require significant training on annotated datasets.

2.5 Artistic Representations in AI

The transformation of poetry into artistic representations leverages emerging AI techniques in text-to-image translation. Popular methods include:

- **Generative Adversarial Networks (GANs):** GANs are used to convert text input into images, offering unique interpretations of textual data.
- **Style Transfer and Neural Networks:** Style transfer techniques, through neural networks, allow the merging of artistic styles onto generated images.

2.6 (Optional) Personalized Recitations and Holography

- **(Optional) Deepfake Technology:** Deepfake technology enables users to superimpose their faces and voices onto avatars. The application of this technology in poetry projects allows personalized recitations that can be deeply immersive.
- **(Optional) Holographic Displays:** Holographic displays have gained traction in representing data three-dimensionally, providing viewers with a futuristic display of digital content. Integrating holography into poetic projects can deliver an innovative, immersive experience.

2.7 Conclusion

In summary, the literature indicates considerable progress in computational poetry generation and artistic representation. However, there is a lack of comprehensive tools dedicated to Arabic poetry due to its complex linguistic and prosodic structures. The VERSO project seeks to bridge this gap by leveraging the latest advancements in AI, artistic visualization, and personalized recitations to redefine Arabic poetry in the digital age.

CHAPTER III: Methodology

3.1 Introduction

The methodology chapter outlines the systematic approach undertaken in the VERSO project. By leveraging advanced artificial intelligence, natural language processing, and deep learning techniques, this chapter describes the technical framework and processes employed in developing the platform. The aim is to generate, analyze, and present Arabic poetry in an innovative way that honors traditional metrics and explores artistic representations.

3.2 Project Framework

The VERSO project is divided into multiple key stages to ensure systematic development and clear project structuring:

1. **Requirement Analysis:** Detailed discussions with stakeholders and potential users to understand their needs and expectations regarding Arabic poetry generation and presentation.
2. **Design and Planning:** Development of a comprehensive blueprint that defines the architecture and functionalities of the system, ensuring modularity and scalability.
3. **Development Phases:**
    - **AI-Powered Arabic Poetry Generation**: Creation of machine learning models trained on Arabic poetry datasets to generate verses that align with traditional prosodic rules.
    - **Prosody Analysis Engine**: Building a rule-based and machine learning hybrid algorithm to evaluate the generated poems for meter adherence.
    - **Artistic Representations**: Integration of text-to-image translation models to generate artistic visual representations of poems.

- **(Optional) Personalized Recitations:** Implementation of deepfake technology to allow users to superimpose their faces and voices onto avatars.
- **(Optional) Holographic Displays**: Incorporating holographic technology for immersive poetry presentations.

3.3 Data Collection and Preprocessing

- **Poetry Datasets:** Compilation of Arabic poetry datasets from classical and modern poets for training and analysis purposes. The data is carefully annotated for meter, rhyme, and style.
- **Text Preprocessing:** Standardization of poetry data by normalizing spelling, diacritics, and other linguistic aspects for consistency.
- **Image Datasets:** Collection of image datasets aligned with Arabic artistic themes to be used in artistic representations.

3.4 AI Models and Algorithms

- **Poetry Generation Model:** A transformer-based language model is trained using annotated Arabic poetry data. It leverages the rich patterns in the dataset to generate new poems while adhering to traditional structures.
- **Prosody Analysis Engine:** The analysis engine combines rule-based algorithms with machine learning to identify meter patterns and rhyme schemes, ensuring adherence to Al-Arood.
- **Text-to-Image Translation:** Implementation of GANs (Generative Adversarial Networks) to generate visual representations from textual inputs.
- **(Optional) Deepfake Technology:** Application of generative models to personalize recitations using user-provided facial and vocal data.

3.5 System Architecture

The architecture is designed for scalability and modularity, enabling each component to function independently or collaboratively. The core architecture consists of:

1. **Backend Server:** Manages data processing, poetry generation, prosody analysis, and artistic representation.
2. **Frontend Interface:** Web-based user interface allows seamless interaction with the platform.
3. **Database:** Stores poetry data, user-generated poems, and visual artworks.

3.6 Conclusion

This methodology ensures that the VERSO project addresses the requirements of AI-powered poetry generation, prosody analysis, and artistic representation. The framework provides a clear path from data collection to system evaluation, ensuring a solution for redefining Arabic poetry to the digital era.

CHAPTER IV: System Design

4.1 Introduction

This chapter details the technical architecture and design principles of the VERSO project. It aims to provide a comprehensive understanding of how the system's components are structured, how they interact, and how they ensure the platform's objectives are achieved efficiently.

4.2 Requirements Specification

Functional Requirements

- **AI-Powered Poetry Generator**
  - **Requirement:** The system must generate Arabic poetry based on user inputs.
  - **Input:** Accept text prompts from users.
  - **Output:** Produce poems that conform to traditional Arabic poetic forms.

- **Prosody Analysis Engine**
  - **Requirement:** The system must analyze the meter and rhyme of generated poems.
  - **Input:** Take the text of generated poems.
  - **Output:** Provide feedback on the poetic meter and rhyme accuracy.

- **Visual Transformation Process**
  - **Requirement:** The system must convert poems into bespoke artistic visual representations.
  - **Input:** Receive generated poems.
  - **Output:** Produce visual art representations based on poems.

- **Audiovisual Recitation**
    - **Requirement:** The system must use deepfake technology to create personalized recitations of poems.
    - **Input:** Accept user selection of avatars and generated poems.
    - **Output:** Provide an audiovisual recitation of poems.

Non-Functional Requirements

- **Performance**: Support up to 1000 concurrent users with response times not exceeding 3 seconds.
- **Security**
    - **Requirement:** Implement secure user authentication and data encryption.
    - **Requirement:** Comply with applicable data protection laws.
- **Usability**
    - **Requirement:** Provide an intuitive user interface for users without technical background.
    - **Requirement:** Support English and Arabic languages.
- **Reliability:** Achieve a minimum uptime of 99.5%, excluding scheduled maintenance.
- **Scalability:** Design the system to scale up for increased user numbers and data volume.

Data Requirements

- **Storage:** Utilize relational databases for storing user profiles, poem data, and metadata.
- **Privacy:** Adhere to GDPR and other relevant data protection regulations.
- **Backup:** Implement regular data backups to ensure data integrity and availability.

Software Requirements

- **Development Environment:** Develop using Python and JavaScript with Django and React frameworks.
- **Dependencies**
    - **Requirement:** Use TensorFlow or PyTorch for AI models.
    - **Requirement:** Use cloud services like AWS or Azure for hosting and storage.

Hardware Requirements

- **Server Specifications:** Operate on servers capable of handling intensive AI computations and data processing.
- **Client Devices:** Be accessible through modern web browsers on desktop and mobile devices.

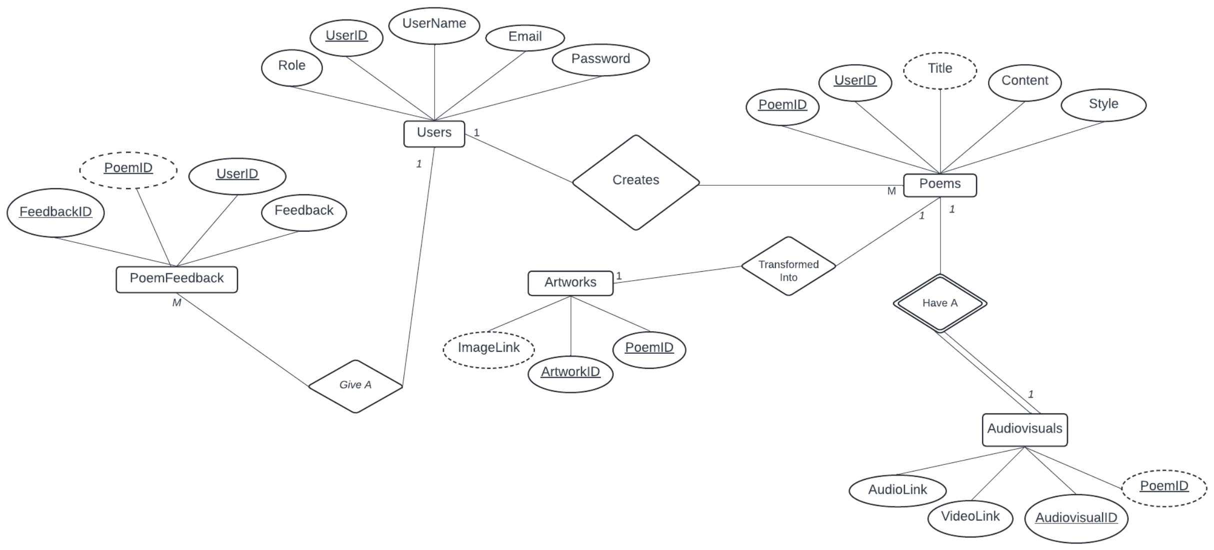
4.3 Design and Analysis

In the design phase, we sketch out what our project will look like and how it will work. Think of it as creating a map before building something. This map will help us make sure everything we plan to build fits together well and works right. The diagrams this document will contains:

- **Entity-Relationship (ER) Diagrams:** These are like family trees for our project's data, showing how different pieces of information are connected.
- **SQL Schema:** The SQL schema takes the relationships and attributes defined in the ER diagram and translates them into a database structure.
- **Use Case Diagrams:** These show all the things users can do with our project, like creating a poem or giving feedback.
- **Class Diagrams:** These are more detailed plans that show the various parts of our system and what each part does.
- **Sequence Diagrams:** These show the order of steps that happen when a user does something, like submitting a poem.

We'll use these diagrams to ensure that we build a system that's easy to use, can grow over time, and is dependable.

Entity-Relationship (ER) Diagram



4. Figure 0-1: Entity-Relationship (ER) Diagram

Overview

The ER diagram is a visual representation of the VERSO project's data structure. It defines how different entities such as Users, Poems, Artworks, Audiovisuals, and Poem Feedback are interrelated.

Key Components

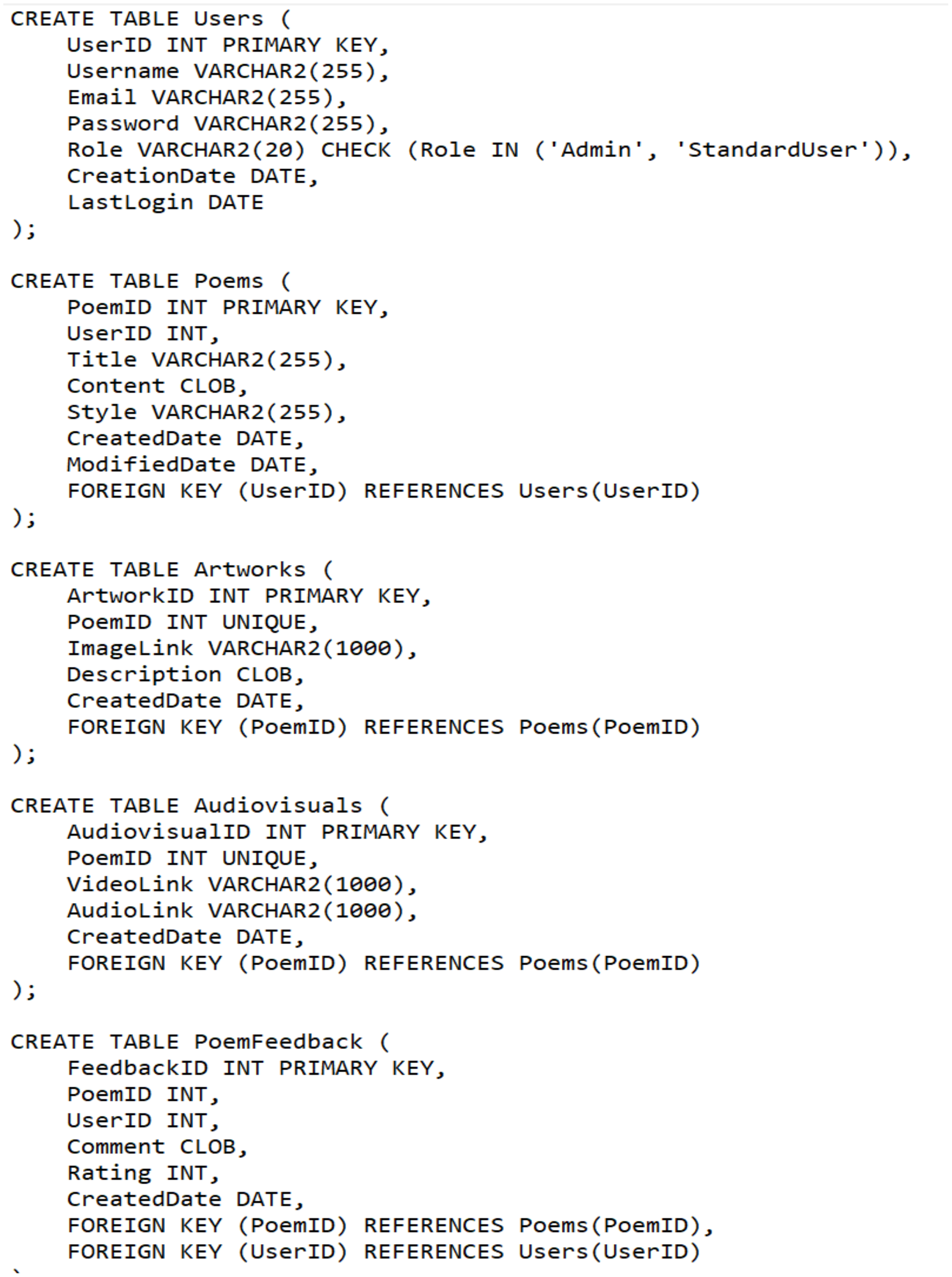
- **Users Entity**: Shows all registered users of the VERSO platform.
- **Poems Entity**: Lists all poems created by users. Each poem is linked to a user.
- **Artworks Entity**: Connects each poem to a corresponding artwork, depicting the poem visually.
- **Audiovisuals Entity**: Links poems to their audiovisual representations.
- **Poem Feedback Entity**: Captures user comments and ratings for poems.

Relationships

- **Creates Relationship**: Users can create multiple Poems.
- **Transformed Into Relationship**: Poems can be transformed into Artworks and Audiovisuals.
- **Give A Relationship**: Users can provide feedback on multiple Poems, and Poems can receive feedback from multiple Users.

The ER diagram helps in designing the database by clearly showing how various entities are associated with each other, which is essential for creating the tables and their relationships in the database.

SQL Schema



```sql
CREATE TABLE Users (
    UserID INT PRIMARY KEY,
    Username VARCHAR2(255),
    Email VARCHAR2(255),
    Password VARCHAR2(255),
    Role VARCHAR2(20) CHECK (Role IN ('Admin', 'StandardUser')),
    CreationDate DATE,
    LastLogin DATE
);

CREATE TABLE Poems (
    PoemID INT PRIMARY KEY,
    UserID INT,
    Title VARCHAR2(255),
    Content CLOB,
    Style VARCHAR2(255),
    CreatedDate DATE,
    ModifiedDate DATE,
    FOREIGN KEY (UserID) REFERENCES Users(UserID)
);

CREATE TABLE Artworks (
    ArtworkID INT PRIMARY KEY,
    PoemID INT UNIQUE,
    ImageLink VARCHAR2(1000),
    Description CLOB,
    CreatedDate DATE,
    FOREIGN KEY (PoemID) REFERENCES Poems(PoemID)
);

CREATE TABLE Audiovisuals (
    AudiovisualID INT PRIMARY KEY,
    PoemID INT UNIQUE,
    VideoLink VARCHAR2(1000),
    AudioLink VARCHAR2(1000),
    CreatedDate DATE,
    FOREIGN KEY (PoemID) REFERENCES Poems(PoemID)
);

CREATE TABLE PoemFeedback (
    FeedbackID INT PRIMARY KEY,
    PoemID INT,
    UserID INT,
    Comment CLOB,
    Rating INT,
    CreatedDate DATE,
    FOREIGN KEY (PoemID) REFERENCES Poems(PoemID),
    FOREIGN KEY (UserID) REFERENCES Users(UserID)
```

4. Figure 0-2: SQL Schema

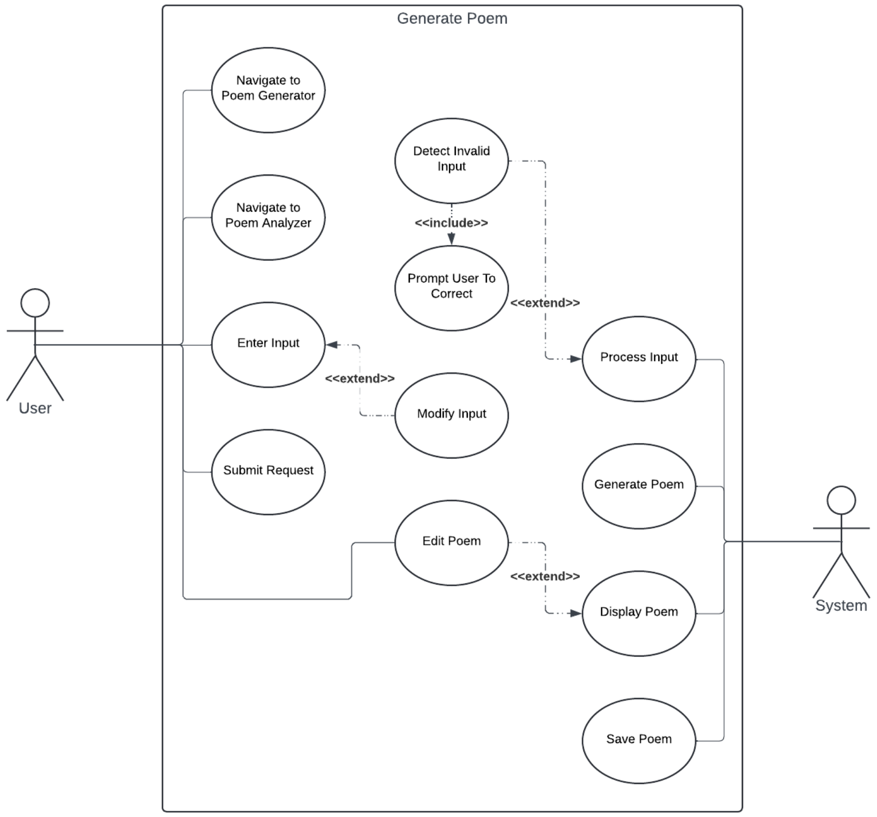
Tables

- **Users Table:** Corresponds to the Users entity in the ER diagram. It includes unique identifiers and personal information fields for authentication.
- **Poems Table:** Matches the Poems entity, containing details about the poems and a reference to the Users table to establish ownership.
- **Artworks Table:** Relates to the Artworks entity, holding information about the visual representation of poems with a unique link to the Poems table.
- **Audiovisuals Table:** Connects to the Audiovisuals entity, storing multimedia content related to poems.
- **Poem Feedback Table:** Corresponds to the Poem Feedback entity, capturing user feedback with references to both the Poems and Users tables for relational integrity.

The SQL schema includes the definition of primary keys for uniqueness, foreign keys for relational integrity, and appropriate data types for storing information.

Use Case Diagrams

Generate Poem Diagram



4. Figure 0-3: Generate Poem Diagram

Overview

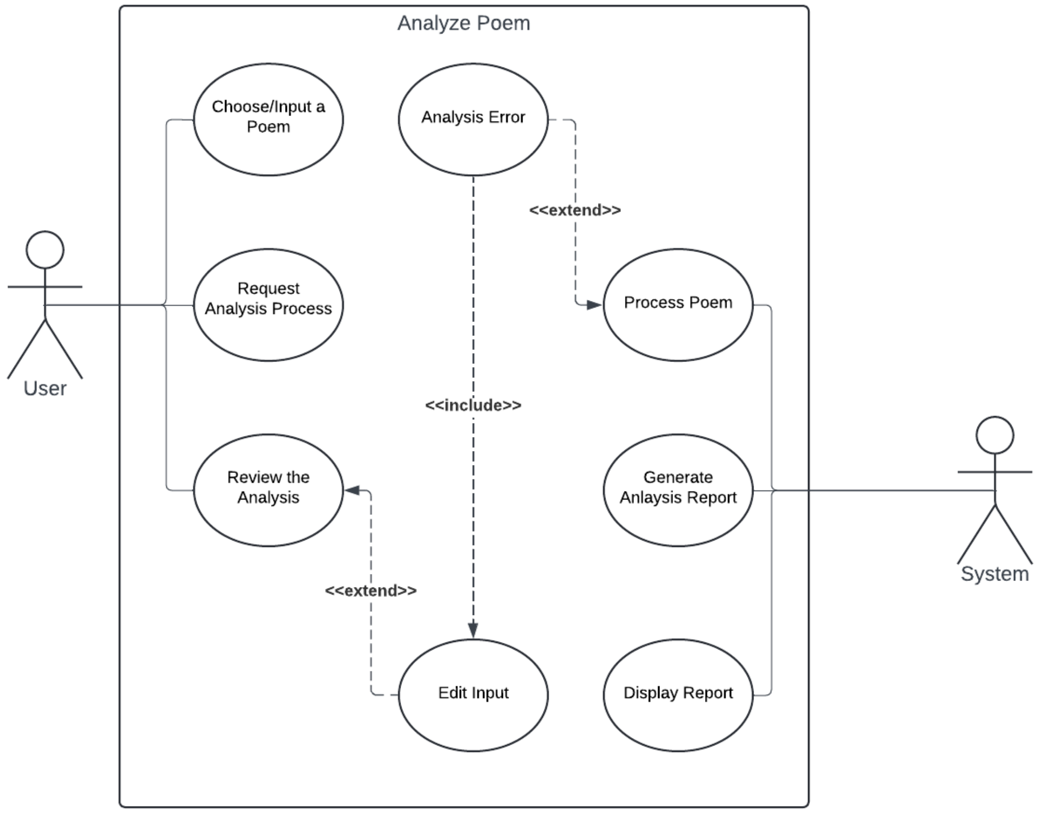
This diagram walks you through the steps of making a new poem with the VERSO system, from the initial idea to the final saved poem.

Main Steps

1. **Navigate to Poem Generator:** The user starts by going to the part of the VERSO application where they can create a poem.

2. **Enter Input:** The user types in their ideas or keywords that they want their poem to be about.
3. **Submit Request:** After entering their ideas, the user submits this information to the system to start the poem generation process.
4. **Detect Invalid Input:** The system checks to make sure the input from the user makes sense. If not, it asks the user to fix any issues.
5. **Modify Input:** If needed, the user makes corrections to their initial ideas or keywords.
6. **Generate Poem:** The system processes the corrected input and creates a new poem.
7. **Display Poem:** The new poem is shown to the user.
8. **Edit Poem (Optional):**The user can choose to edit the poem if they want to make changes.
9. **Save Poem:** Once the user is happy with the poem, they can save it.

Analyze Poem Diagram



4. Figure 0-4: Analyze Poem Diagram

Overview

This diagram guides you through the process within the VERSO system that allows a user to analyze a poem they have created or selected.

Main Steps

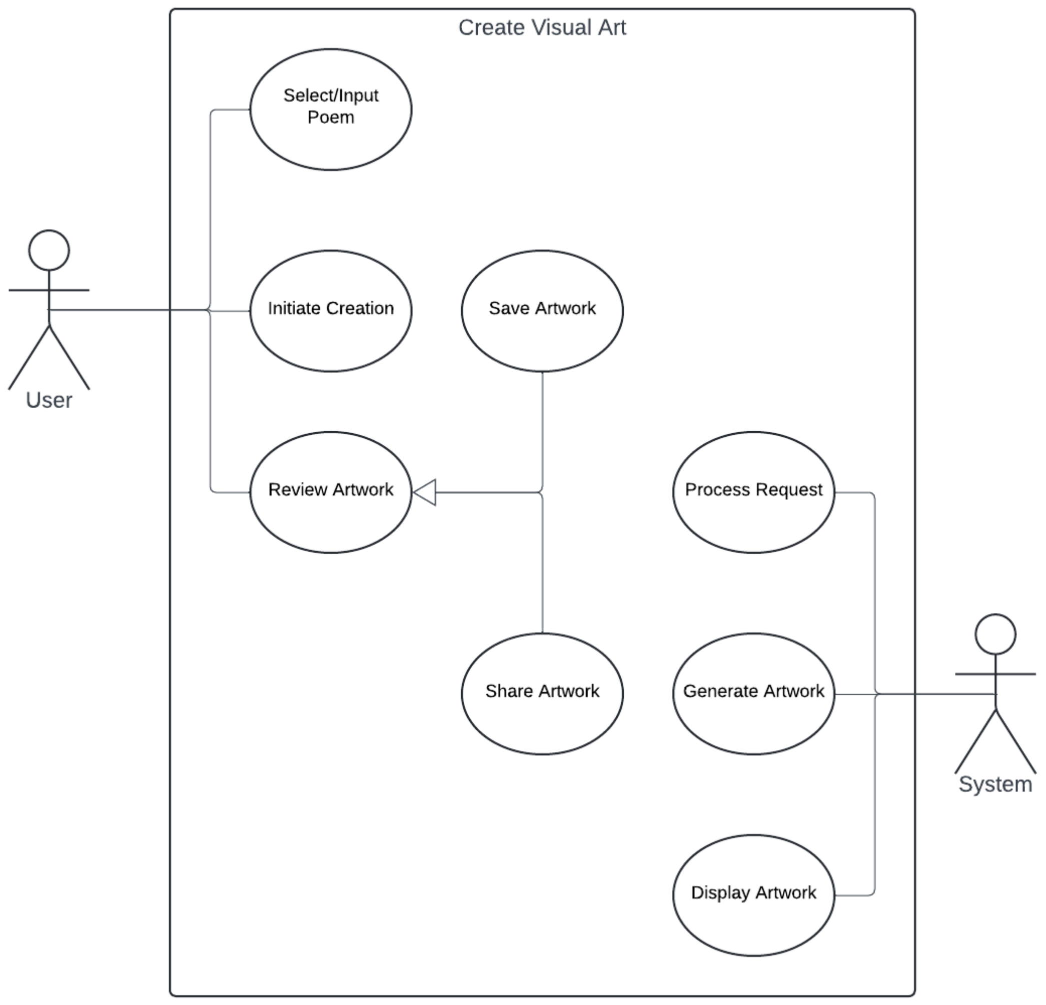
1. **Choose/Input a Poem:** The user begins by selecting or inputting a poem that they wish to have analyzed by the system.
2. **Request Analysis Process:** After choosing the poem, the user requests an analysis, signaling the system to begin evaluating the poem's content.

3. **Process Poem:** The system processes the poem, looking into its structure, style, and content.
4. **Generate Analysis Report:** Once the poem is processed, the system generates a detailed report on its analysis, which may include metrics, style, and suggestions for improvement.
5. **Display Report:** The analysis report is displayed to the user, allowing them to review the insights generated by the system.
6. **Review the Analysis:** With the report now visible, the user reviews the system's analysis of the poem.
7. **Edit Input (Optional):** If the user sees an opportunity to refine their poem based on the analysis, they can choose to edit the input directly from the analysis screen.

In the case of errors or the need for further refinement:

- **Analysis Error:** If there is an issue with the analysis process, such as incomplete data or a processable poem structure, the system indicates an error.
- **Prompt User to Correct (Optional):** The system may prompt the user to make necessary corrections if the input does not meet the criteria for a valid analysis.

Create Visual Art Diagram



4. Figure 0-5: Create Visual Art Diagram

Overview

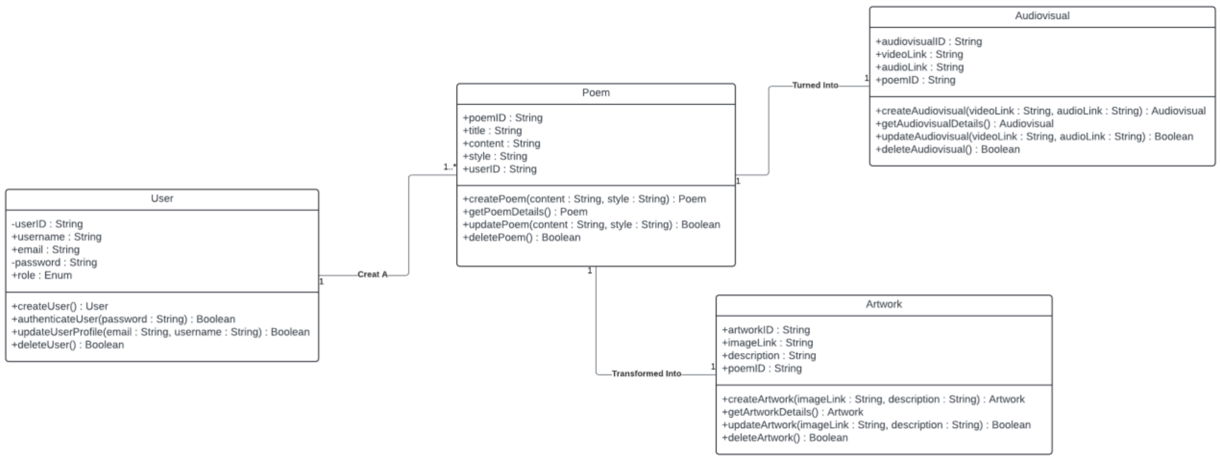
This diagram outlines the steps a user takes in the VERSO system to create a visual representation of their poem.

Main Steps

1. **Select/Input Poem:** The user picks an existing poem or inputs a new one they wish to visualize.

2. **Initiate Creation:** With the poem selected, the user triggers the process to start creating visual art from the poem.
3. **Process Request:** The system takes the user's poem and processes it to understand its themes and mood for the artwork.
4. **Generate Artwork:** The system uses processed information to generate a piece of visual art that represents the poem.
5. **Display Artwork:** The newly created artwork is shown to the user.
6. **Review Artwork:** The user looks over the visual art and decides if it fits their vision.
7. **Save Artwork:** If the user is satisfied, they can save the artwork.
8. **Share Artwork (Optional):** The user has the option to share the artwork on various platforms or within the VERSO community.

Class Diagram


Audiovisual
+audiovisualID : String
+videoLink : String
+audioLink : String
+poemID : String
+createAudiovisual(videoLink : String, audioLink : String) : Audiovisual
+getAudiovisualDetails() : Audiovisual
+updateAudiovisual(videoLink : String, audioLink : String) : Boolean
+deleteAudiovisual() : Boolean
Turned Into
1
1
Poem
+poemID : String
+title : String
+content : String
+style : String
+userID : String
+createPoem(content : String, style : String) : Poem
+getPoemDetails() : Poem
+updatePoem(content : String, style : String) : Boolean
+deletePoem() : Boolean
1..*
User
-userID : String
+username : String
+email : String
-password : String
+role : Enum
+createUser() : User
+authenticateUser(password : String) : Boolean
+updateUserProfile(email : String, username : String) : Boolean
+deleteUser() : Boolean
Creat A
1
1
Transformed Into
1
Artwork
+artworkID : String
+imageLink : String
+description : String
+poemID : String
+createArtwork(imageLink : String, description : String) : Artwork
+getArtworkDetails() : Artwork
+updateArtwork(imageLink : String, description : String) : Boolean
+deleteArtwork() : Boolean

4. Figure 0-6: Class Diagram

Overview

This diagram lays out the blueprints for various parts of the VERSO system, kind of like an architect's plan for a building. It tells us what information each part deals with and what it can do.

Key Components

➢ **User Class**

- **Details:** Has information like user ID, username, email, and password.
- **Actions:** Can create a user, log in, change profile details, or delete a user account.

➢ **Poem Class**

- **Details:** Keeps track of each poem's ID, title, content, style, and the user who created it.
- **Actions:** Lets you make a new poem, see poem details, change them, or remove a poem from the system.

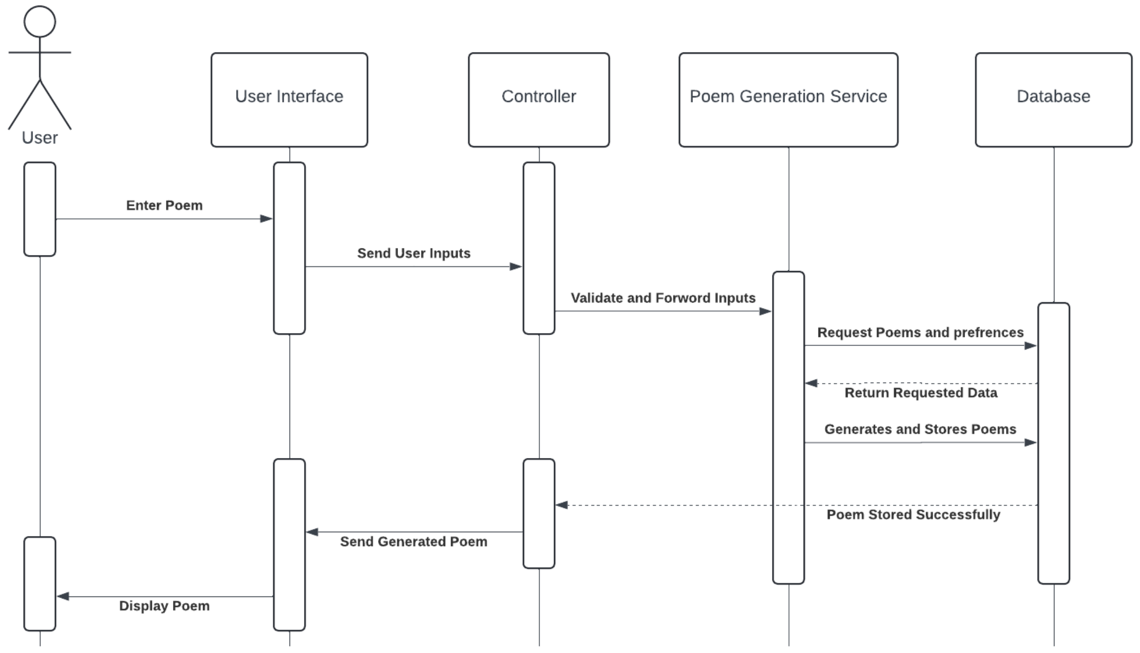
- **Artwork Class**
    - **Details:** Contains artwork ID, a link to the image, a description, and the poem ID it's connected to.
    - **Actions:** Can create new artwork, get details about it, update the info, or delete it.

- **Audiovisual Class**
    - **Details:** Holds IDs for audiovisual elements, like links to videos or audios, and links them to their poems.
    - **Actions:** Allows creation of new audiovisual content, viewing details, updating, or deleting them.

Relationships Among Classes

- **Create A:** Shows that a User can create a Poem.
- **Turned Into:** Indicates that a Poem can be turned into both Artwork and an Audiovisual piece.

Sequence Diagram



4. Figure 0-7: Sequence Diagram

Overview

This diagram is like a storyboard that shows the steps that happen when a user wants to create a poem using VERSO.

Steps Shown in the Diagram:

1. **Enter Poem:** The user starts by typing their poem into the VERSO interface.
2. **Send User Inputs:** After entering the poem, the user sends it to the system.
3. **Validate and Forward Inputs:** The system checks if the poem looks right, then sends it to the poem generation service.
4. **Request Poems and Preferences:** The poem generation service asks the database for any needed information to help create the poem.

5. **Return Requested Data:** The database sends back the information the poem service asked for.

6. **Generates and Stores Poems:** The poem service uses the information to create the poem and then saves it in the database.

7. **Poem Stored Successfully:** The database confirms that the poem has been saved.

8. **Display Poem:** Finally, the user gets to see their newly created poem on their screen.

4.4 Conclusion

The VERSO project's system architecture and design ensure modularity, scalability, and seamless interaction between components. This structure allows the system to fulfill its objective of redefining Arabic poetry through AI-powered generation, artistic representation, and immersive presentation.

CHAPTER V: Implementation

5.1 Introduction

In this chapter, we outline the detailed steps and timeline for implementing the VERSO project. The implementation plan is divided into key phases, each containing specific tasks that will guide the project toward its goals.

5.2 Tools

Software:

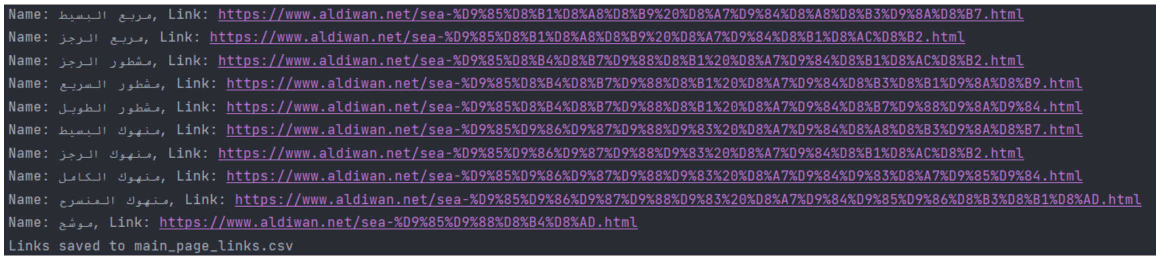
- **Django**: For backend development and API integration.
- **Vue.js**: To build a responsive and interactive frontend interface.
- **Google Colab**: Used for training and testing machine learning models, including fine-tuning and evaluation.
- **Selenium**: For web scraping data from sources like Diwan.net.
- **Pandas**: For handling and preprocessing large datasets of Arabic poetry.
- **ngrok**: To expose the local server for public access during testing and deployment.

Languages:

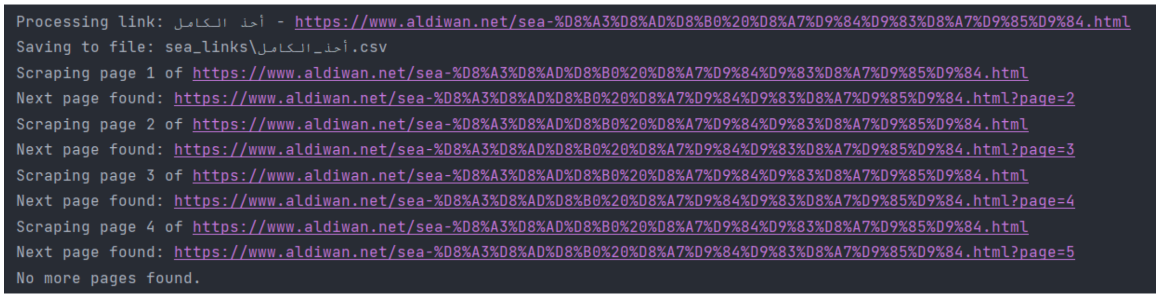
- **Python**: For backend logic, natural language processing (NLP), and AI model integration.
- **JavaScript**: For frontend interactivity and functionality.
- **HTML/CSS**: For structuring and styling the user interface.

5.3 Data Collection

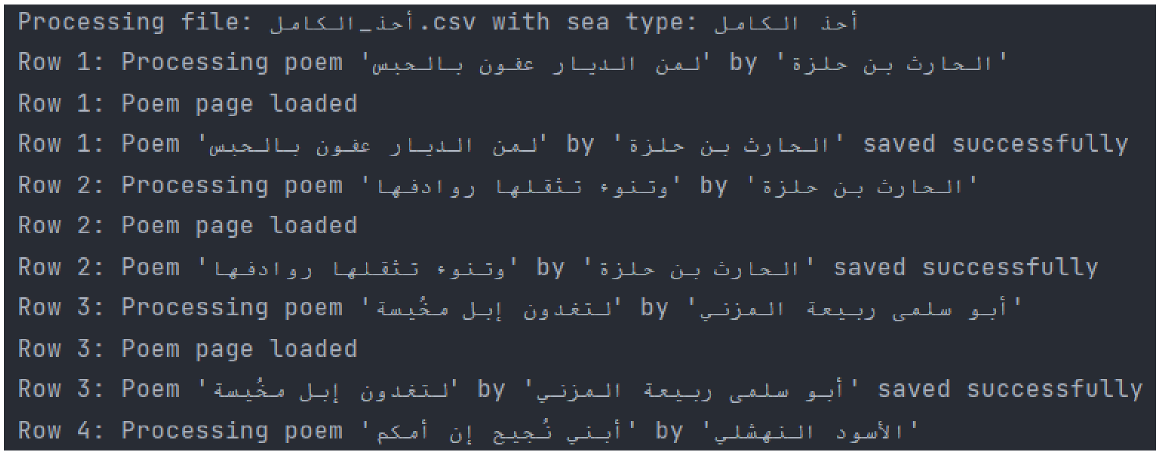
  - Compiled a large dataset of Arabic poems from sources like **Aldiwan.net** using a custom web scraper implemented with **Selenium**.



```
Name: مربع البسيط, Link: https://www.aldiwan.net/sea-%D9%85%D8%B1%D8%A8%D8%B9%20%D8%A7%D9%84%D8%A8%D8%B3%D9%8A%D8%B7.html
Name: مربع الرجز, Link: https://www.aldiwan.net/sea-%D9%85%D8%B1%D8%A8%D8%B9%20%D8%A7%D9%84%D8%B1%D8%AC%D8%B2.html
Name: مشطور الرجز, Link: https://www.aldiwan.net/sea-%D9%85%D8%B4%D8%B7%D9%88%D8%B1%20%D8%A7%D9%84%D8%B1%D8%AC%D8%B2.html
Name: مشطور السريع, Link: https://www.aldiwan.net/sea-%D9%85%D8%B4%D8%B7%D9%88%D8%B1%20%D8%A7%D9%84%D8%B3%D8%B1%D9%8A%D8%B9.html
Name: مشطور الطويل, Link: https://www.aldiwan.net/sea-%D9%85%D8%B4%D8%B7%D9%88%D8%B1%20%D8%A7%D9%84%D8%B7%D9%88%D9%8A%D9%84.html
Name: منهوك البسيط, Link: https://www.aldiwan.net/sea-%D9%85%D9%86%D9%87%D9%88%D9%83%20%D8%A7%D9%84%D8%A8%D8%B3%D9%8A%D8%B7.html
Name: منهوك الرجز, Link: https://www.aldiwan.net/sea-%D9%85%D9%86%D9%87%D9%88%D9%83%20%D8%A7%D9%84%D8%B1%D8%AC%D8%B2.html
Name: منهوك الكامل, Link: https://www.aldiwan.net/sea-%D9%85%D9%86%D9%87%D9%88%D9%83%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html
Name: منهوك المنسرح, Link: https://www.aldiwan.net/sea-%D9%85%D9%86%D9%87%D9%88%D9%83%20%D8%A7%D9%84%D9%85%D9%86%D8%B3%D8%B1%D8%AD.html
Name: موشح, Link: https://www.aldiwan.net/sea-%D9%85%D9%88%D8%B4%D8%AD.html
Links saved to main_page_links.csv
```



```
Processing link: أحذ الكامل - https://www.aldiwan.net/sea-%D8%A3%D8%AD%D8%B0%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html
Saving to file: sea_links\أحذ_الكامل.csv
Scraping page 1 of https://www.aldiwan.net/sea-%D8%A3%D8%AD%D8%B0%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html
Next page found: https://www.aldiwan.net/sea-%D8%A3%D8%AD%D8%B0%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html?page=2
Scraping page 2 of https://www.aldiwan.net/sea-%D8%A3%D8%AD%D8%B0%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html
Next page found: https://www.aldiwan.net/sea-%D8%A3%D8%AD%D8%B0%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html?page=3
Scraping page 3 of https://www.aldiwan.net/sea-%D8%A3%D8%AD%D8%B0%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html
Next page found: https://www.aldiwan.net/sea-%D8%A3%D8%AD%D8%B0%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html?page=4
Scraping page 4 of https://www.aldiwan.net/sea-%D8%A3%D8%AD%D8%B0%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html
Next page found: https://www.aldiwan.net/sea-%D8%A3%D8%AD%D8%B0%20%D8%A7%D9%84%D9%83%D8%A7%D9%85%D9%84.html?page=5
No more pages found.
```



```
Processing file: أحذ_الكامل.csv with sea type: أحذ الكامل
Row 1: Processing poem 'لمن الديار عفون بالحبس' by 'الحارث بن حلزة'
Row 1: Poem page loaded
Row 1: Poem 'لمن الديار عفون بالحبس' by 'الحارث بن حلزة' saved successfully
Row 2: Processing poem 'وتنوء تثقلها روادفها' by 'الحارث بن حلزة'
Row 2: Poem page loaded
Row 2: Poem 'وتنوء تثقلها روادفها' by 'الحارث بن حلزة' saved successfully
Row 3: Processing poem 'لتغدون إبل مخُيسة' by 'أبو سلمى ربيعة المزني'
Row 3: Poem page loaded
Row 3: Poem 'لتغدون إبل مخُيسة' by 'أبو سلمى ربيعة المزني' saved successfully
Row 4: Processing poem 'أبني نُجيج إن أمكم' by 'الأسود النهشلي'
```

- Merged and cleaned datasets from multiple files to create a unified and comprehensive corpus with over 112,000 entries.
- Performed preprocessing steps to remove duplicates, normalize whitespace, and handle missing data.

5.4 System Implementation

Text Processing:

- Developed a custom tokenizer (**ArabicPoemTokenizer**) to split Arabic poetry into diacritics and word units while maintaining meter-specific characteristics.
- Designed normalization rules to handle special cases in Arabic grammar (e.g., proper handling of demonstrative pronouns and exceptions like "الله" or "الرحمن").

Poetic Meter Analysis:

- Implemented the **ArabicPoeticUnitsBinary** module to convert tokenized poems into binary patterns and match them against predefined **Tafe'lat** (poetic feet).
- Integrated error-checking mechanisms to calculate matching percentages for poetic meters.

Fine-Tuning Silma-9B:

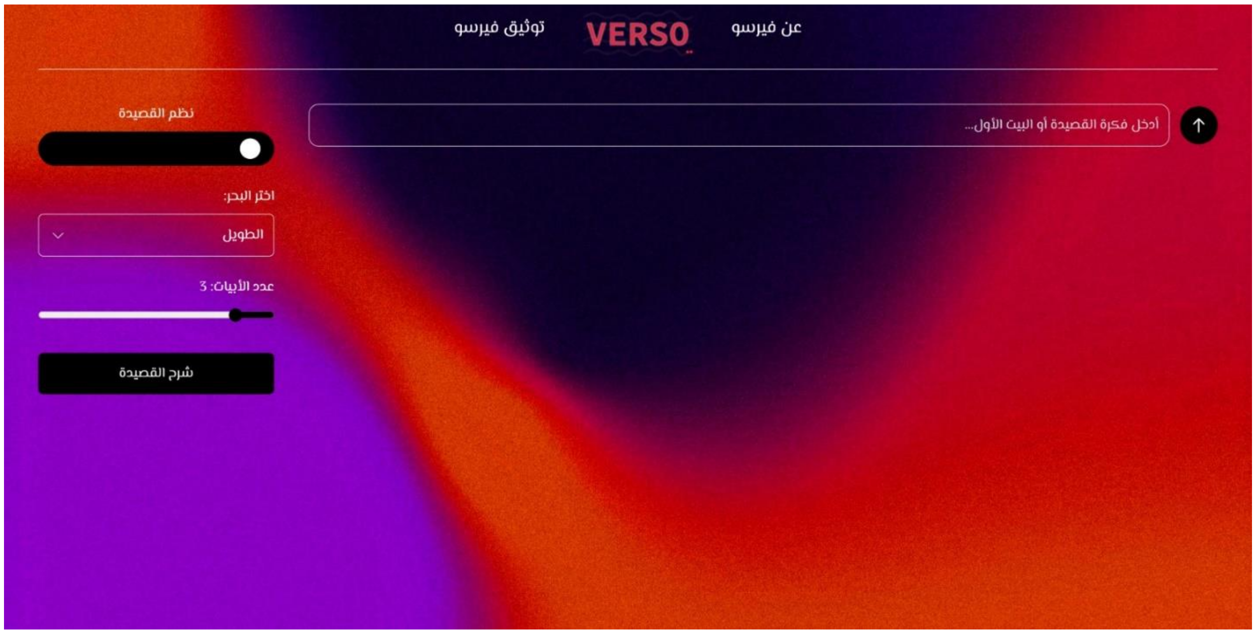
- Loaded the pre-trained **Silma-9B** model, designed specifically for Arabic language tasks.
- Applied **LoRA (Low-Rank Adaptation)** to fine-tune the model efficiently by focusing on the **attention layers** only, optimizing resource usage.
- Used **4-bit quantization** to reduce computational and memory requirements during fine-tuning, making it feasible to train on limited hardware resources like **Google Colab**.

System Design:

- Designed a modular architecture, ensuring each component (e.g., tokenization, meter analysis, and generation) is independent yet seamlessly integrated.
- Implemented tools for organizing and validating the dataset using the **fillter.py** script.

5.5 Preliminary Results and UI:

Interface:



1. Header Section

Navigation Links:

"عن فيرسو" (About VERSO): Likely a link to information about the application.

"توثيق فيرسو" (VERSO Documentation): Possibly a link to a page that provides user guides or API documentation.

Logo:

The word "VERSO" is prominently displayed in the center with a wavy underline, likely representing the application's branding.

2. Input Section

Main Text Input:

Placeholder text: "أدخل فكرة القصيدة أو البيت الأول..." (Enter the idea of the poem or the first verse...).

This is a text input where users can provide the topic or starting idea for the poem.

3. Settings Section

Poem Mode Toggle:

Label: "نظم القصيدة" (Poem Mode).

This is a toggle switch that might enable or disable poem generation.

Meter Dropdown:

Label: "اختر البحر:" (Select the meter).

Default option: "الطويل" (Al-Tawil, a traditional Arabic poetic meter).

This is a dropdown menu to select the poetic meter.

Verse Count Slider:

Label: "عدد الأبيات: 3" (Number of verses: 3).

A slider to adjust the number of verses in the generated poem.

4. Action Button

Explanation Button:

Text: "شرح القصيدة" (Explain the poem).

This button trigger a feature to analyze or explain the generated poem.

5. Background

The background has a vibrant gradient of colors, blending red, purple, and orange, which creates a visually appealing and artistic design.

This choice of colors aligns well with the creative nature of poetry and the artistic branding of the application.

Possible Functionality:

User Input:

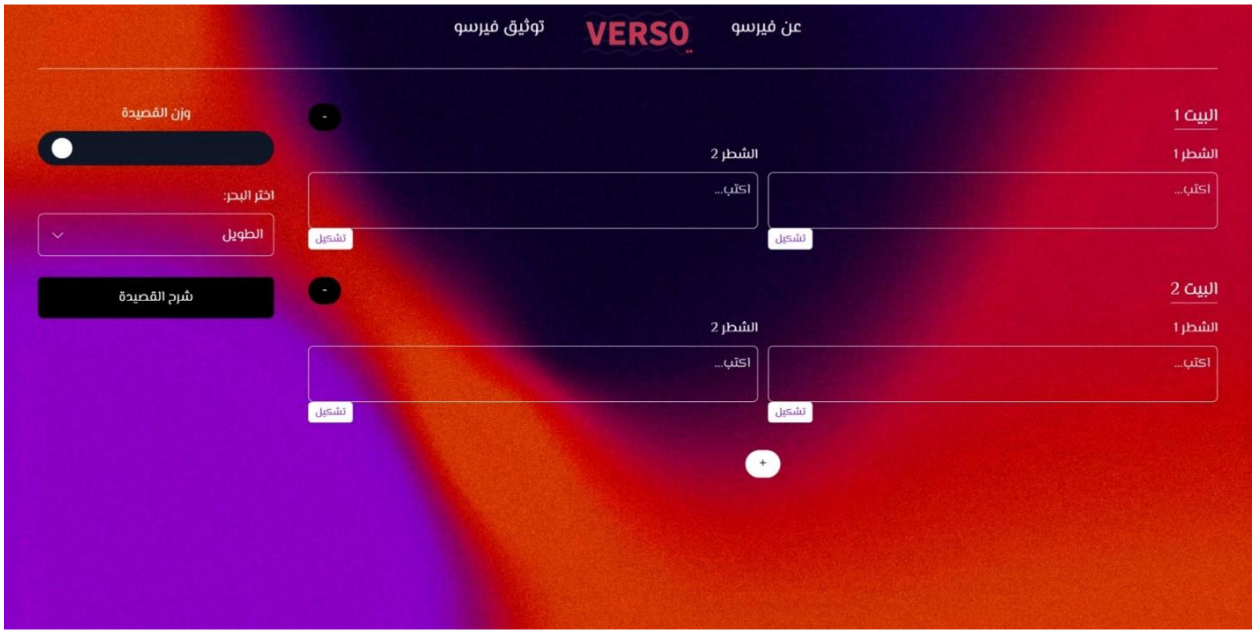
Users enter a topic or starting verse to generate or structure a poem.

Customization:

Users can customize the poetic structure by selecting the meter and adjusting the number of verses.

Analysis:

After generating the poem, users can use the "شرح القصيدة" (Explain the poem) button for analysis or explanation.



اختر البحر: الطويل

1. Header Section

Navigation Links:

"عن فيرسو" (About VERSO): A link to information about the application.

"توثيق فيرسو" (VERSO Documentation): A link to application guides or API documentation.

Logo:

The "VERSO" logo is still at the center with the same branding as before.

2. Input Section (Left Panel)

Poem Meter Toggle:

Label: "وزن القصيدة" (Poem Meter).

A toggle switch to enable or disable the meter system.

Meter Selection Dropdown:

Label: "اختر البحر:" (Select the meter).

Default option: "الطويل" (Al-Tawil, a traditional Arabic poetic meter).

Explanation Button:

Text: "شرح القصيدة" (Explain the poem).

This button likely triggers a feature that explains the poem.

3. Poem Composition Section (Right Panel)

Verses and Hemistich Input Fields:

Each verse is divided into two hemistiches (الشطر 1 and الشطر 2):

Input Fields: Users can write the first and second hemistich of each verse.

Placeholder Text: "اكتب..." (Write...).

Buttons: Each hemistich has a button labeled "تشكيل" (Diacritize), which likely applies diacritical marks (Tashkeel) to the text.

Example Structure:

البيت 1 (Verse 1):

الشطر 1 (Hemistich 1): Input field with a "تشكيل" button.

الشطر 2 (Hemistich 2): Input field with a "تشكيل" button.

البيت 2 (Verse 2):

Similar structure as Verse 1.

Add Verse Button:

A "+" button is at the bottom-right corner, likely for adding more verses dynamically.

Remove Verse Button:

A "-" button is next to each verse, allowing users to delete verses.

4. Background

The background retains the vibrant gradient from the previous design, with a visually appealing blend of red, purple, and orange.

New Features Added in This Version

Interactive Poem Writing:

The interface now allows users to compose individual verses, separated into hemistiches.

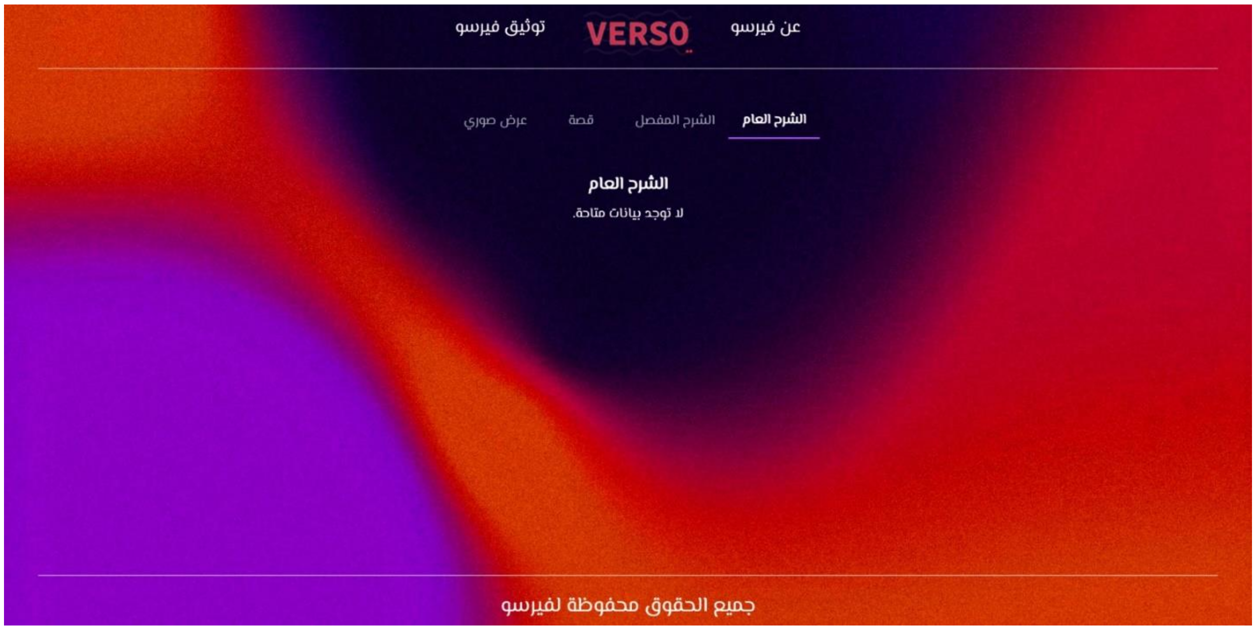
The diacritization feature provides precise control over the poem's phonetics.

Dynamic Verse Management:

Users can add (+) or remove (-) verses dynamically.

Explanation and Diacritization:

The "شرح القصيدة" (Explain the poem) button offers analysis, while the "تشكيل" (Diacritize) button focuses on accurate Arabic script.



الشرح العام

لا توجد بيانات متاحة.

جميع الحقوق محفوظة لفيرسو

1. Header Section

Navigation Links:

"عن فيرسو" (About VERSO): Link to an information page about the application.

"توثيق فيرسو" (VERSO Documentation): Link to a page with documentation or guides for using the application.

Logo:

The "VERSO" logo is centered, maintaining the branding consistency across all pages.

2. Content Section

Tabs for Analysis Types:

Several tabs allow the user to select the type of analysis or explanation for the poem:

"الشرح العام" (General Explanation): The currently selected tab, providing a general overview or summary of the poem.

"الشرح المفصل" (Detailed Explanation): Likely includes line-by-line or in-depth analysis of the poem.

"قصة" (Story): May provide a narrative or context related to the poem.

"عرض صوري" (Visual Display): Could generate or display visual interpretations of the poem.

Selected Tab Content:

Title: "الشرح العام" (General Explanation).

Content: "لا توجد بيانات متاحة." (No data available). This suggests that the feature hasn't generated content yet or no input was provided.

3. Footer Section

Copyright Notice:

Text: "جميع الحقوق محفوظة لفيرسو" (All rights reserved for VERSO).

This footer ensures legal and branding information is displayed at the bottom of the page.

4. Background

The vibrant gradient background persists in this design, featuring a blend of red, purple, and orange, giving the page an artistic and creative feel.

Potential Functionality

Navigation Between Tabs:

Users can switch between different types of explanations (general, detailed, story, or visual).

Content Updates:

The main content area dynamically updates based on the selected tab and the generated analysis.

Fallback Messaging:

If no data is available for the selected tab, the application displays a message ("لا توجد بيانات متاحة").

5.6 Difficulties during implementation:

1. **Quantization Impact on Model Quality**
   - The use of 4-bit quantization for SILMA-9B introduced a loss of precision, affecting the model's ability to generate nuanced and coherent poetry.
   - Training and fine-tuning required significant computational resources, which were limited in the development environment.
2. **Dataset Challenges**
   - The dataset collected from sources like Diwan.net contained inconsistencies, irrelevant data, and mismatched styles, requiring extensive cleaning and preprocessing.
   - Limited size of high-quality, labeled Arabic poetry datasets constrained the fine-tuning process.
3. **Integration Issues**
   - Connecting the quantized model with frontend and backend components required handling compatibility issues between various tools (e.g., Hugging Face, QLoRA, and Django).
   - Deploying the model on public servers was challenging due to infrastructure constraints, necessitating temporary solutions like ngrok.
4. **Fine-Tuning Risks**
   - Improper hyperparameter tuning during fine-tuning with QLoRA led to risks of overfitting, particularly when dealing with small or noisy datasets.
   - Tokenizer misalignment between the pretrained and fine-tuned states sometimes disrupted the output quality.

5.7 Appropriate Solutions:

1. **Quantization and Model Optimization**
   - Used **8-bit quantization** as an alternative to 4-bit to retain better precision while maintaining a manageable computational footprint.
   - Employed **QLoRA** for efficient fine-tuning with minimal resource usage.
2. **Data Cleaning and Preprocessing**
   - Developed scripts in Python and Pandas to clean, preprocess, and align poetry data, ensuring consistency across training datasets.
   - Utilized **web scraping techniques (Selenium)** to expand the dataset, focusing on high-quality poetic content.
3. **Infrastructure Improvements**
   - Leveraged **Google Colab** as a cost-effective environment for training and fine-tuning the model.
   - Used **ngrok** to expose local servers for testing and deployment without needing dedicated hosting infrastructure.
4. **Fine-Tuning Strategies**
   - Experimented with multiple hyperparameters to identify optimal configurations for learning rate, batch size, and dropout values.
   - Ensured alignment between tokenization methods in pretraining and fine-tuning stages to maintain model compatibility.

5.8 Conclusion:

The implementation of the project presented several challenges, particularly in handling quantized models, dataset quality, and infrastructure limitations. By leveraging advanced tools like QLoRA, Hugging Face, and Django, we successfully integrated SILMA-9B into a poetry generation framework. The project demonstrated the potential of large-scale language models in handling complex and creative tasks like Arabic poetry, showcasing their ability to balance creativity with structural constraints.

Through iterative problem-solving and optimization, we achieved a system capable of generating structured, meaningful, and stylistically coherent poetry. The project not only highlights the importance of robust datasets and precise fine-tuning but also underscores the value of scaling in AI for nuanced language tasks. Future work could focus on expanding datasets, optimizing deployment pipelines, and exploring larger models for even greater creative capabilities.

CHAPTER VI: Testing

6.1 Introduction

Testing was a critical phase to ensure the accuracy, functionality, and reliability of our system. Below is a detailed description of our testing plan, approach, types of testing performed, and the conclusions drawn:

6.2 Unit Testing

- **ArabicPoemTokenizer**:
  - Input: A sample poem text.
  - Output: Tokenized text with correct diacritics and structure.
  - Verified accuracy by comparing output tokens with manual annotations.
- **ArabicPoeticUnitsBinary**:
  - Input: Binary representations of poetry rhythms.
  - Output: Matched patterns against predefined **tafeelat**.
  - Checked match percentages for correctness.

6.3 Integration Testing:

- Combined tokenization and poetic meter classification to test end-to-end functionality.
- Ensured seamless integration with the **Silma-9B** model for poetry generation.

6.4 Performance Testing:

- Measured training times and resource utilization during fine-tuning on **Google Colab** with **4-bit quantization**.
- Monitored GPU memory usage and training speed with tools like **pynvml**.

6.5 Quality Assurance for Poetry Generation:

- Assessed generated poetry for:
  - Linguistic coherence.
  - Alignment with traditional Arabic poetry styles.
  - Consistency in meter.

6.6 Test Cases Considered:

1. **Tokenization**:
   - **Valid Input**: Simple poems with standard diacritics.
   - **Edge Cases**: Texts with missing or incorrect diacritics, mixed-language content.
2. **Meter Classification**:
   - **Valid Input**: Poems adhering strictly to traditional Arabic meter rules.
   - **Edge Cases**: Poems with irregular rhythms or ambiguous structures.
3. **Poetry Generation**:
   - **Valid Input**: Clear and concise titles (e.g., "الوطن" ,"الحب").
   - **Edge Cases**: Long and complex titles or single-word prompts.

6.7 Testing Tools:

- **Python Unit Test Framework**: For modular testing.
- **Hugging Face Trainer**: For monitoring fine-tuning loss and checkpointing.
- **Manual Evaluation**: For assessing the quality of generated poems and tokenized output.

6.8 Conclusion:

Testing validated the system’s accuracy, reliability, and adherence to Arabic poetic styles. Robust results across tokenization, meter classification, and poetry generation confirm its readiness for practical use.

CHAPTER VII: Conclusion

7.1 Introduction

This concluding chapter summarizes the VERSO project's ambitions, its potential impact on the field, and outlines the next steps for its evolution and broader application.

7.2 Project Summary

The VERSO project aims to revolutionize the way Arabic poetry is created and experienced by integrating advanced AI technology with traditional cultural expressions.

7.3 Future Work

- **Research Opportunities:** Identify areas for further research, such as expanding the AI model to other forms of poetry or languages. Suggest collaborative research initiatives with academic institutions.
- **Development Plans:** Outline plans for enhancing the platform's features, including improving AI accuracy, expanding the user interface, and integrating additional multimedia elements.
- **Potential Applications:** Discuss the potential for applying the VERSO project's technologies in other cultural, educational, or creative fields.

7.4 Final Thoughts

Conclude with an optimistic outlook on how the VERSO project will continue to evolve and influence both technological innovation and cultural preservation. Encourage ongoing support and collaboration from stakeholders and the community to drive future developments.

By effectively summarizing the project and outlining a vision for the future, this chapter not only wraps up the report but also sets a strategic path forward, inviting continued engagement and innovation.

References

[1] DeepLearning.AI. (n.d.). Natural language processing. Retrieved from https://www.deeplearning.ai/resources/natural-language-processing/

[2] Deepfakesweb. (n.d.). Home page. Retrieved from https://deepfakesweb.com/

[3] GeeksforGeeks. (n.d.). LSTM-based poetry generation using NLP in Python. Retrieved from https://www.geeksforgeeks.org/lstm-based-poetry-generation-using-nlp-in-python/

[4] GitHub. (2024, February 22). How AI code generation works. Retrieved from https://github.blog/2024-02-22-how-ai-code-generation-works/

[5] Hitachi Solutions. (n.d.). Natural language processing. Retrieved from https://global.hitachi-solutions.com/blog/natural-language-processing/

[6] Inamdar, F. M., Ambesange, S., Mane, R., Hussain, H., Wagh, S., & Lakhe, P. (Year). Voice cloning using artificial intelligence and machine learning: A review. [Journal Name, Volume(Issue), Page numbers if available]. ResearchGate. https://www.researchgate.net/publication/376626376_Voice_Cloning_Using_Artificial_Intelligence_and_Machine_Learning_A_Review

[7] Kesarwani, V. (2018). Automatic poetry classification using natural language processing (master's thesis). University of Ottawa, School of Electrical Engineering and Computer Science, Faculty of Engineering. Retrieved from https://ruor.uottawa.ca/items/1a1245e5-7b52-40ed-9bcd-7aecb7c8369a/full

[8] OpenAI. (n.d.). Text generation. Retrieved from https://platform.openai.com/docs/guides/text-generation/

[9] **Sawant, R., Shaikh, A., Sabat, S., & Bhole, V. (2021). Text to image generation using GAN. In Proceedings of the International Conference on IoT Based Control Networks & Intelligent Systems - ICICNIS 2021 (pp. 1-7). A.C Patil College of Engineering, Navi Mumbai, India. Retrieved from https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3882570**

[10] **Speechify. (n.d.). How to create AI deepfake videos. Retrieved from https://speechify.com/blog/how-to-create-ai-deepfake-videos/**

[11] **TechTarget. (n.d.). Deepfake. Retrieved from https://www.techtarget.com/whatis/definition/deepfake**

[12] **Imam Mohammad Ibn Saud Islamic University. (2008). العروض والقافية: للسنة الثالثة الثانوية بالمعاهد العلمية [Arud and Qafiya: For the third secondary year in scientific institutes]. Riyadh: Imam Mohammad Ibn Saud Islamic University. ISBN 9960-04-744-X. Retrieved from https://units.imamu.edu.sa/deanships/elearn/Documents/e-learning_1/hs3/%D8%A7%D9%84%D8%B9%D8%B1%D9%88%D8%B6-%D9%88%D8%A7%D9%84%D9%82%D8%A7%D9%81%D9%8A%D8%A9-%D8%AB%D8%A7%D9%84%D8%AB%D8%A9-%D8%AB%D8%A7%D9%86%D9%88%D9%8A-%D9%811-%D9%812.pdf**

Appendix A: Overview of Al-Arood and Al-Bihar in Arabic Poetry

A.1 Introduction

This appendix provides a detailed overview of Al-Arood, the science of Arabic prosody, and Al-Bihar, the specific rhythmic patterns used in Arabic poetry. It offers insights into the historical background, the primary patterns of Al-Bihar, their applications, and cultural significance

A.2 Introduction to Al-Arood

Al-Arood refers to the structured science of Arabic prosody, essential for crafting and analyzing Arabic poetry. Originating in the 8th century by Al-Khalil ibn Ahmad al-Farahidi, Al-Arood organizes Arabic poetry into rhythmic patterns or 'seas' (Al-Bihar), which dictate the poetic meter and help maintain the linguistic purity of Arabic through its verse.

A.3 The Sixteen Al-Bihar

Al-Bihar (singular: Bahr) are the meters of Arabic poetry, each with unique rhythmic patterns:

1. **Al-Tawil**: Known for its dignity and formality, used in solemn contexts.
2. **Al-Bassit**: Offers simplicity and flexibility, suitable for varied themes.
3. **Al-Wafir**: Characterized by its depth and complex structures.
4. **Al-Kamil**: Represents completeness and is used for expansive themes.
5. **Ar-Rajs**: Known for its quick and lively rhythm, suitable for joyous themes.
6. **Al-Khafif:** Light and swift, often used for delicate and playful subjects.
7. **Al-Hazaj**: Tight and intense, perfect for enthusiastic expressions.
8. **Al-Muttakarib:** Features closeness in rhythmic structure, used for impactful messages.

9. **Al-Munsarih**: Provides a sense of release and liberation.
10. **Al-Muktatab**: Steady and balanced, typically used for narrative poetry.
11. **Al-Muktadarak:** A reverse form of Al-Muktatab, embodying a reflective mood.
12. **Al-Madid**: Flowing and extensive, ideal for storytelling.
13. **Al-Mujtath:** Rare and intricate, used for intellectually stimulating themes.
14. **Al-Ramel**: Notable for its rhythmic pace, used in formal poetry.
15. **Al-Khabab:** Light and airy, often associated with fleeting beauty.
16. **Al-Saria**: Fast and flowing, suited for dramatic narratives.

A.4 Applications of Al-Bihar

Al-Bihar serve various roles in Arabic literature:

- **Creative Writing**: Poets choose specific Bahrs to match the emotional and thematic needs of their work.
- **Literary Analysis**: Scholars analyze the use of Al-Bihar to gain deeper insights into a poet’s technique and the poem’s thematic elements.
- **Education**: Students of Arabic literature study Al-Bihar to understand the foundational elements of Arabic poetic form.

A.5 Cultural Significance

The study and application of Al-Arood and Al-Bihar connect contemporary Arabic poets and literary scholars to the rich traditions of Arabic literature. This connection not only preserves but also enriches the cultural heritage by enabling the transmission of traditional forms infused with modern sensibilities.

END

Thank You