Dlalat دلالات

Abdullah Mohammad Ahmad AbuZeidan 0201393

Abdulelah Mohd Ghaleb Santrisi 0205589

Omar Eihab Saleh Zamil 0204071

ARTIFICIAL INTELLIGENCE DEPARTMENT

KING ABDULLAH II SCHOOL OF INFORMATION TECHNOLOGY

THE UNIVERSITY OF JORDAN

ABSTRACT

This report outlines the analysis of the Holy Qur'an using a web chatbot service that employs a free-text question posed in Modern Standard Arabic (MSA) and a collection of Qur'anic passages. The report discusses the implementation of a system that utilizes the BERT model in Python to provide a ranked list of answer-bearing passages. The system accommodates both factoid and non-factoid questions.

ACKNOWLEDGMENTS

After thanking God for everything, we would also like to extend our appreciation to our graduation supervisors, (**Dr. Ruba Obiedat & Dr. Nadim Obied)**, for providing us the chance to work on the "Dlalat دلالات" project. Their guidance and support have played a crucial role in facilitating thorough research and effectively communicating our information and knowledge.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

MSA	Modern Standard Arabic
mAP	Mean Average Precision $\mathrm{mAP} = \frac{1}{N} \sum_{i=1}^{N} \mathrm{AP}_i$
mRR	Mean Reciprocal Recall $\mathrm{MRR} = \frac{1}{\lvert Q \rvert} \sum_{i=1}^{\lvert Q \rvert} \frac{1}{\mathrm{rank}_i}$
FAISS	Facebook AI Similarity Search

1. Chapter One: Introduction

1.1 Preamble:

The Holy Qur'an is a central text in Islam, containing spiritual, moral, and legal guidance for Muslims. With the advancement of technology, web chatbot services have become a popular medium for disseminating information and facilitating discussions. This report explores the development of a web chatbot that offers detailed insights into the Qur'an using the capabilities of the BERT model. The chatbot's primary function is to receive free-text questions in MSA and return a ranked list of passages that potentially contain relevant answers highlighting the answer if found.

1.2 Problem Statement:

The challenge is to design a system that effectively understands user questions, retrieves relevant Qur'anic passages, and ranks them based on their likelihood of containing pertinent answers, the system then searches these passages for answers. This involves leveraging the capabilities of the BERT model to retrieve coherent and informative responses while ensuring that the system remains accurate and respectful of the sacred text.

1.3 Project Aim and Objectives:

The main objectives of this project include:

- Developing a web chatbot service for Qur'anic analysis.
- Implementing a system that interprets free-text questions in MSA.
- Finding a collection of Qur'anic passages that cover various aspects of the Holy Qur'an.
- Utilizing the BERT model to retrieve detailed responses.
- Ranking passages based on their potential to contain relevant answers.
- Handling both factoid and non-factoid questions to provide informative responses.

1.4 Project Software and Hardware Requirements:

- Software requirement:

Table (1): Software requirements for Dlalat دلالات.

Software	**Requirement**
Browser	Microsoft Edge, Firefox, Safari, and Google Chrome.
Development Tools	Notepad++, Python,Transformers,PyTorch,FAISS

- Hardware requirement:

Table (2): Hardware requirements for Dlalat دلالات.

Hardware	**Requirement**
Computer	Core i5 - 1480 MHz Pentium minimum, V - 1 GHz or higher recommended Nvidia graphics card min. 8GB VRAM and cuda enabled
Memory (RAM)	16 GB
Hard disk	500 GB

1.5 Project Schedule:

Project management is declared as planning, organizing, securing, and managing resources to achieve specific goals. The following table displays the project management:

Table (3): Project Schedule Management in developing Dlalat دلالات.

Task	Description	Start time	End Time	Duration	Dependency
T1	Planning	10/27/23	10/31/23	5 Days	
T2	Information Gathering	10/30/23	10/03/23	3 Days	T1
T3	Analysis	11/03/23	11/14/23	10 Days	T2
T4	Design	11/04/23	11/12/23	7 Days	T3
T5	Implementation	11/05/23	12/16/23	30 Days	T4
T6	Testing	12/17/23	12/23/23	5 Days	T5
T7	Documentation	10/27/23	12/25/23	44 Days	T1, T2, T3, T4, T5, T6
T8	Submission	01/04/24	01/04/24	1 Day	T7

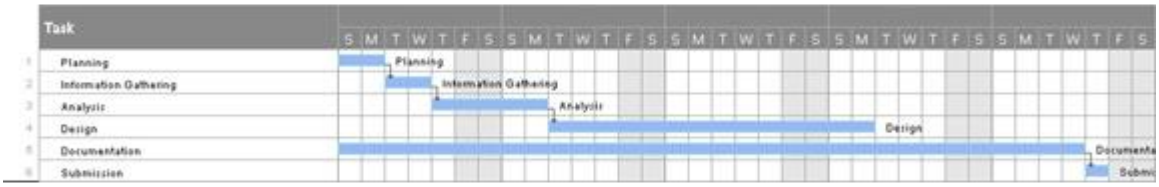


Figure (1): Gant Chart for Dlalat دلالات.

2. Chapter Two: Related Existing Systems

2.1 Introduction:

There are existing platforms that provide Qur'anic analysis, but the integration of the BERT model into a web chatbot service is a novel approach. Traditional methods involve manual indexing and searching, which can be time-consuming and may lack the flexibility to handle a wide range of questions.

2.2 Existing Systems:

ARABIC QUESTION ANSWERING ON THE HOLY QUR'AN (Malhas, 2023): The study show that a passage retriever over a BM25 index of Qur'anic passages expanded with two MSA resources significantly outperformed a baseline retriever over an index of Qur'anic passages only. Moreover, it empirically show that the fine-tuned CL-AraBERT reader model significantly outperformed the similarly finetuned AraBERT model, which is the baseline. In general, the CL-AraBERT reader performed better on single-answer questions in comparison to multi-answer questions.

Moreover, it has also outperformed the baseline over both types of questions. Furthermore, despite the integral contribution of fine-tuning with the MSA datasets in enhancing the performance of the readers, relying exclusively on those datasets (without MRC datasets in CA, e.g., QRCD) may not be sufficient for our reader models. This finding demonstrates the relatively high impact of the QRCD dataset (despite its modest size). As for the QA system, it consistently performed better on single-answer questions in comparison to multi-answer questions. However, the experiments provide enough evidence to suggest that a native BERT-based model architecture fine-tuned on the MRC task may not be intrinsically optimal for multi-answer questions.

LARSA22 at Qur’an QA 2022: Text-to-Text Transformer for Finding Answers to Questions from Qur’an (Mellahet al., 2022): The study propose a Transformer-based QA system using the mT5 Language Model (LM). We finetuned the model on the Qur'anic Reading Comprehension Dataset (QRCD) which was provided in the context of the Qur'an QA 2022 shared task. The QRCD dataset consists of question-passage pairs as input, and the corresponding adequate answers provided by expert annotators as output. Evaluation results on the same Dataset show that our best model can achieve 0.98 (F1 Score) on the Dev Set and 0.40 on the Test Set. We discuss those results and challenge.

2.3 System Advantages:

- Light weight: the system implements models that are smaller than models implemented in existing systems, existing systems need to be ran on a GPU for efficient predictions, we prioritize efficiency over performance.
- The system implements Faiss (Facebook AI Similarity Search) index. FAISS stores entries as vectors and retrieves them by different criteria like similarity and other metrics enabling much faster retrieval than traditional indexing techniques.
- None of the existing systems offer a UI for querying.

3. Chapter Three: System Requirements Engineering and Analysis

3.1 Datasets:

The primary dataset for this project consists of the Holy Qur'an translated into text format, it contains 1,266 thematic Qur'anic passages that cover the whole Holy Qur'an. Additionally, a collection of factoid and non-factoid questions related to various aspects of the Qur'an is compiled to assess the system's performance. The system relies on these datasets to provide accurate and informative responses.

- **Source and Composition:**

The primary dataset for this project is sourced from a JSON file named quran-data.json. This dataset includes Quranic verses, detailed with surah number, verse range, and passage text. Additional sources might include external databases for linguistic patterns, semantic databases, and other religious texts for comparative analysis.

- **Preprocessing and Augmentation:**

Text Preprocessing: Includes normalization, removing stop words, and standardizing text format. Data Augmentation: Expands the dataset by altering the original passages to create synthetic, yet realistic, variations.

1. **Text Cleaning:**

- **Remove Special Characters:** Uses a regular expression to replace specific special characters with a space.
- **Remove URLs:** Uses a regular expression to replace URLs with a space.
- **Handle User Mentions:** Uses a regular expression to replace questions handles (words starting with '@') with a space.
- **Remove Tabs**: Replaces tabs with a space.

- **Remove Line Jumps**: Replaces line jumps with a space.
- **Replace Multiple Spaces:** Replaces multiple consecutive spaces with a single space.
- **Remove Punctuation**: Uses a regular expression to remove any character that is not a word character or a whitespace character.

2. **Stopword Removal:**

- **Eliminate Common Words:** Remove common words (Arabic Stopwords) that do not contribute significantly to the meaning.

3. **Stemming:**

- **Normalization:** Reduce words to their base or root form to handle variations, by replacing different forms of Arabic letters with their standard forms.
- Employ the **Farasa stemmer** to perform Arabic text stemming, aiming to reduce words to their fundamental root or base forms within the Arabic language context.

4. **Tokenization:**

- **Split Text into Tokens:** Break down the text into individual words or tokens.

5. **The qur'anic passages** where embedded using our trained BERT model, these embeddings where the indexed using FAISS.

- **Data Quality and Integrity:**

The quality of the data is critical, given the religious and cultural significance of the text. Procedures are in place to ensure the integrity of the passages during translation and preprocessing.

3.2 Model Reports:

The system employs the Bert model (**CL-AraBERT**), a state-of-the-art language model developed by Google. The model is pretrained on a diverse range of classical Arabic texts. Fine-tuning the model using specific Qur'anic passages and question-answer pairs enhances its performance in retrieving answers related to the Holy Qur'an and extracting answers from the text**.**

- **Model Selection and Configuration:**

A combination of Transformer models, integrated with custom layers, is employed for this project. Model configurations include language specification (Arabic), training cycles, and BERT utilization.

- **Hyperparameter Tuning and Optimization:**

Utilizes **Learning Rate**: Experiment with learning rates, considering values such as 1e-5, 2e-5,4e-5, and 5e-5, to find an optimal rate for fine-tuning. **Batch Size**: Explore different batch sizes (e.g. 4,8, 16, 32) to balance training speed and memory requirements. **Epochs**: Determine the appropriate number of epochs, monitoring training and validation performance.

- **Training and Evaluation:**

- **Task A Training Objective (Passage Retrieval):**
  - **Pretraining:** The multiple negative ranking loss was used to pretrain the CL-AraBERT model on our dataset, the loss function is ideal if you only have positive query passage pairs as in our case. For each Ai, it uses all other Pj as negative samples, i.e., for Ai, we have 1 positive example (Pi) and n-1 negative examples (Pj). It then minimizes the negative log-likehood for softmax normalized scores, where Ai is the query and Pi is the corresponding positive passage, and Pj is a negative passage for all i≠j.
  - **Fine-tuning:** The triplet loss function was used to finetune the CL-AraBERT model. Given the triplets (anchor,positive,negative) where the anchor is the query, positive is the relevant passage and negative an irrelevant passage, the function maximizes the cosine similarity between the anchor and the positive example and minimizes the cosine similarity between the anchor and the negative example.

    Loss=max ((1-cosinesim(anchor, positive)) - (1-cosinesim(anchor, negative)) +margin, where margin is a hyper-parameter that specifies the minimum distance between the similarities between (anchor, positive) and (anchor, negative).

- **Task B training objective (QA):**

  Total span extraction loss: the sum of a Cross-Entropy for the start and end positions. The models are trained on the preprocessed and augmented dataset with continuous monitoring for performance and potential overfitting. Evaluation metrics ((mAP, mRR) for passage retrieval, F1-score for Question Answering) are recorded.

3.3 Approach:

- **Methodology**:

The project follows a comprehensive NLP methodology, starting from data acquisition, preprocessing, model training, and post-processing of results. Advanced techniques in data augmentation, semantic analysis, and linguistic pattern recognition are employed.

- **Custom model layers:**

To get fixed sized sentence vectors (embeddings) 2 extra layers were added to the CL-AraBERT model. The first added layer is a mean pooling layer, The mean pooling layer takes these word embeddings and computes the mean (average) of all the embeddings along a certain axis. This axis is often the axis corresponding to the sequence length, so that the mean is calculated across all word embeddings in the sequence. The second added layer is a dense layer with a tanh activation layer, the tanh function introduces non-linearity to the model, allowing the neural network to learn and represent more complex relationships in the data.

- **Indexing Passage Embeddings:**

After training and evaluating the encoder model, all of the passages were encoded using the encoder and stored in a FAISS index. FAISS is a python library used for creating indices that are optimized for fast retrieval of vector representations (embeddings). These indices can be trained to cluster vectors enabling even more efficient retrieval.

- **Technological Stack:**

The project relies on advanced NLP libraries, pandas for data manipulation, and custom modules for specific tasks, The system is designed to be scalable and robust, capable of handling large datasets and complex NLP tasks.

- **Future Scope and Scalability:**

The system is designed with scalability in mind, allowing for integration with additional datasets and more complex NLP models. Future enhancements could include multilingual support, integration with AI-driven translation services, and expansion to other religious texts for comparative studies.

4. Chapter Four: System Design & Implementation:

4.1 Introduction:

This chapter includes manly important figures that describe our application process, it will include context diagram, data flow diagram (DFD), use cases diagrams, sequences diagrams, class diagrams.

4.2 Context Diagram:

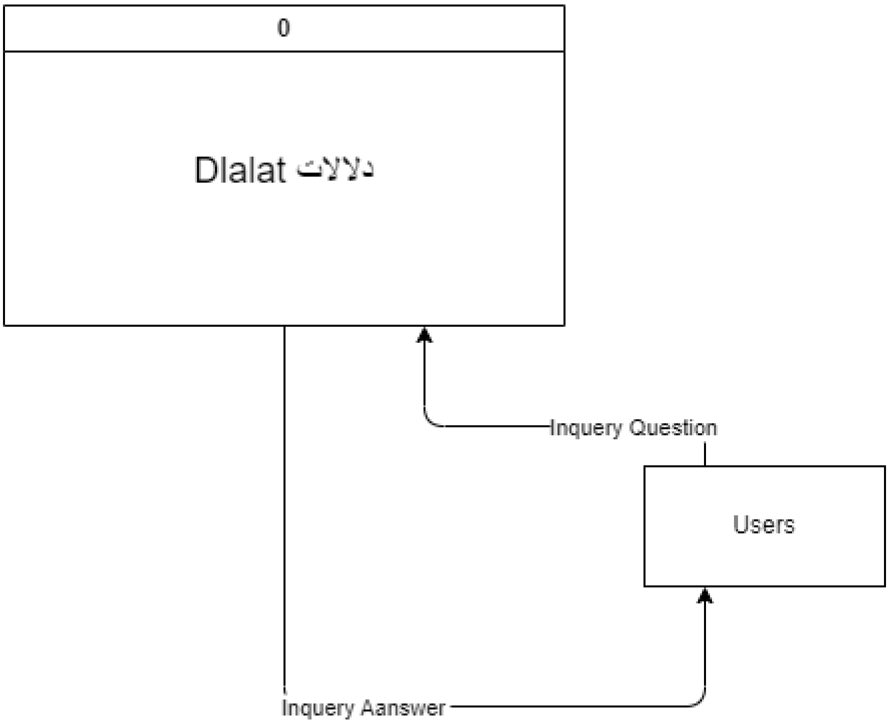


Figure (2): Context Level.

4.3 Data Flow Diagram (DFD):

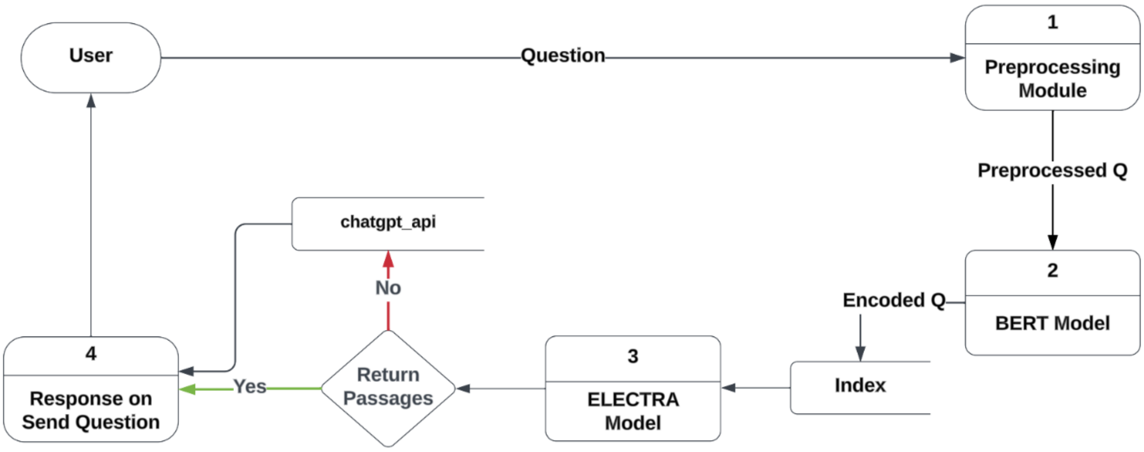


Figure (3): Data Flow Diagram (DFD).

4.4 UML Use Case Diagram:

- **Users Section:**

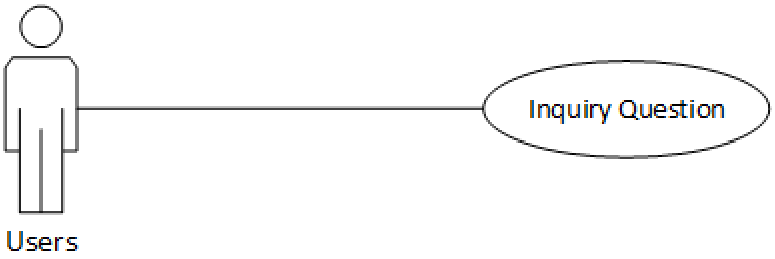


Figure (4): Users Use Case Diagram.

4.5 UML Sequence Diagram:

- **Users Section:**

  - **Inquiry Process:**

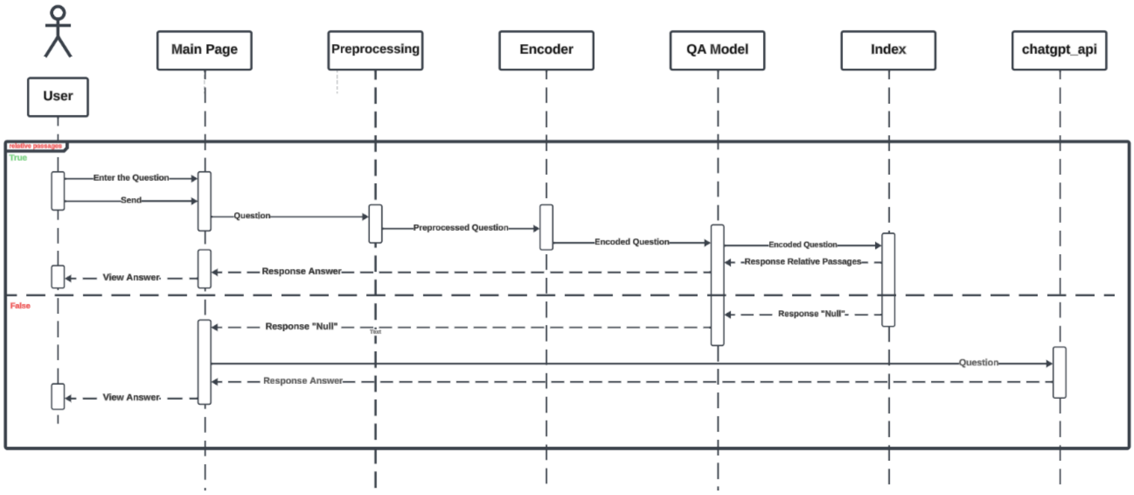


Figure (5): Inquiry Process Sequence Diagram.

- **sequence**:

1. The user enters a question in the interface.
2. The question is then preprocessed.
3. The preprocessed question is then encoded with our BERT model.
4. The most similar passages are then retrieved from the index.
5. The encoded question and encoded passages are run through the ELECTRA model to extract answers from the passages.
6. The answers are displayed to the user.

4.6 UML Class Diagram:

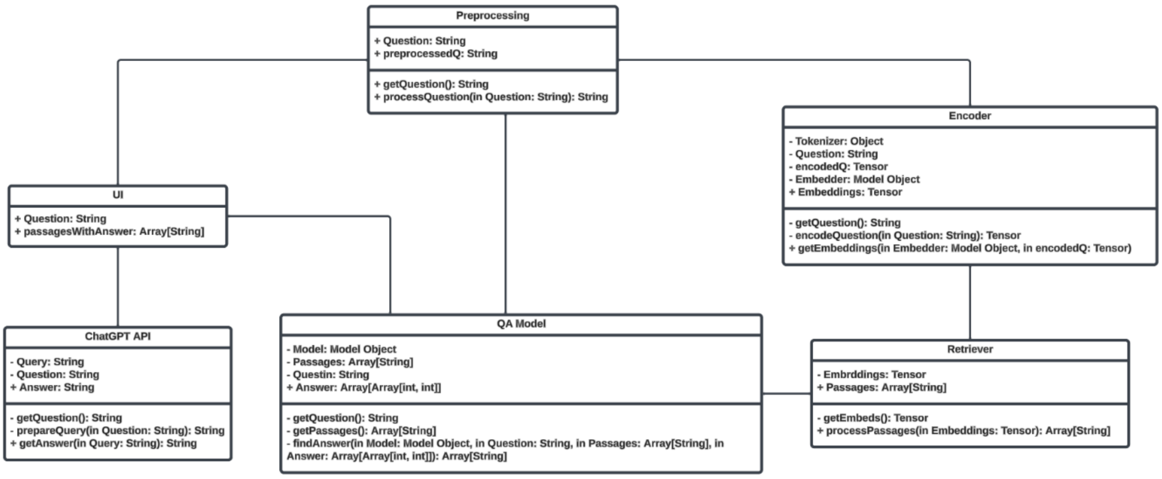

Preprocessing
+ Question: String
+ preprocessedQ: String
+ getQuestion(): String
+ processQuestion(in Question: String): String
Encoder
- Tokenizer: Object
- Question: String
- encodedQ: Tensor
- Embedder: Model Object
+ Embeddings: Tensor
- getQuestion(): String
- encodeQuestion(in Question: String): Tensor
+ getEmbeddings(in Embedder: Model Object, in encodedQ: Tensor)
UI
+ Question: String
+ passagesWithAnswer: Array[String]
ChatGPT API
- Query: String
- Question: String
+ Answer: String
- getQuestion(): String
- prepareQuery(in Question: String): String
+ getAnswer(in Query: String): String
QA Model
- Model: Model Object
- Passages: Array[String]
- Questin: String
+ Answer: Array[Array[int, int]]
- getQuestion(): String
- getPassages(): Array[String]
- findAnswer(in Model: Model Object, in Question: String, in Passages: Array[String], in Answer: Array[Array[int, int]]): Array[String]
Retriever
- Embrddings: Tensor
+ Passages: Array[String]
- getEmbeds(): Tensor
+ processPassages(in Embeddings: Tensor): Array[String]

Figure (6): Class Diagram.

4.7 Summary

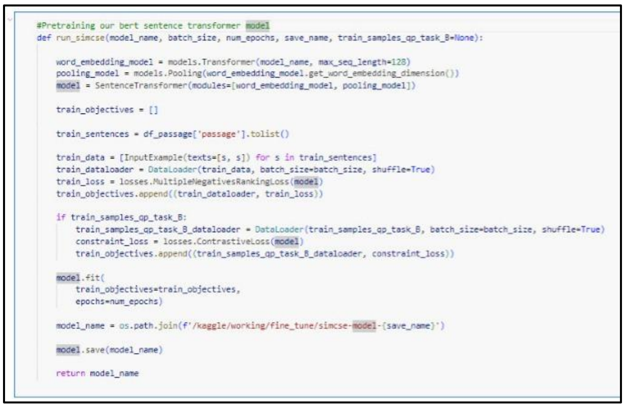
At final, this chapter define our application functionality and how the data transfer inside it the methodology of the system, that we build the context level and 0 level for the system, after that we identified use cases diagrams, sequence diagrams and class diagram for database relations.

5. Chapter Five: System Implementation:

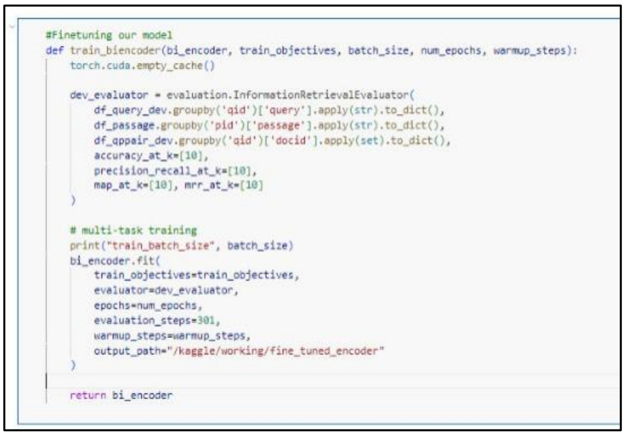
5.1 Introduction:

System implementation it’s an important phase is the system, at this stage designer start design the system from interface to implementation. Therefore, this chapter describe the database implementation relations, also the graphical user interface implementation figures.

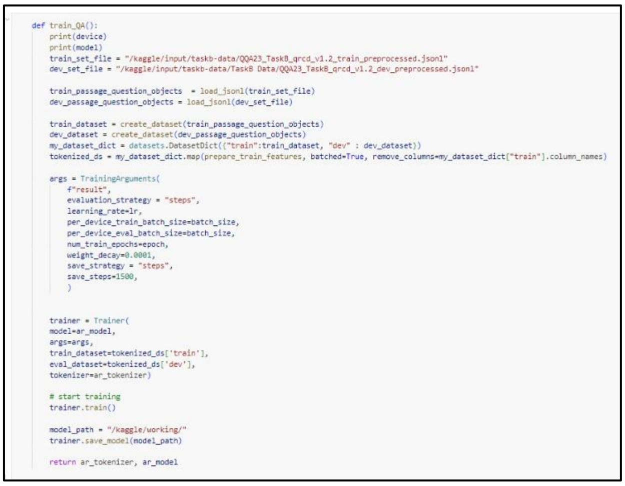
5.2 Code Implementation:



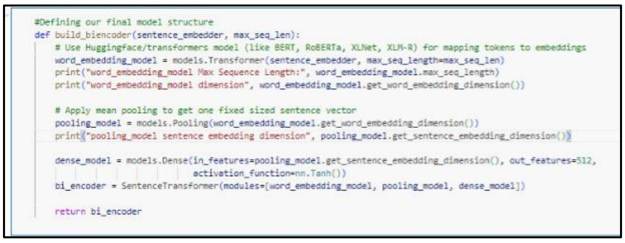
```
#Pretraining our bert sentence transformer model
def run_simcse(model_name, batch_size, num_epochs, save_name, train_samples_qp_task_B=None):

    word_embedding_model = models.Transformer(model_name, max_seq_length=128)
    pooling_model = models.Pooling(word_embedding_model.get_word_embedding_dimension())
    model = SentenceTransformer(modules=[word_embedding_model, pooling_model])

    train_objectives = []

    train_sentences = df_passage['passage'].tolist()

    train_data = [InputExample(texts=[s, s]) for s in train_sentences]
    train_dataloader = DataLoader(train_data, batch_size=batch_size, shuffle=True)
    train_loss = losses.MultipleNegativesRankingLoss(model)
    train_objectives.append((train_dataloader, train_loss))

    if train_samples_qp_task_B:
        train_samples_qp_task_B_dataloader = DataLoader(train_samples_qp_task_B, batch_size=batch_size, shuffle=True)
        constraint_loss = losses.ContrastiveLoss(model)
        train_objectives.append((train_samples_qp_task_B_dataloader, constraint_loss))

    model.fit(
        train_objectives=train_objectives,
        epochs=num_epochs)

    model_name = os.path.join(f'/kaggle/working/fine_tune/simcse-model-{save_name}')

    model.save(model_name)

    return model_name
```



```
#Finetuning our model
def train_biencoder(bi_encoder, train_objectives, batch_size, num_epochs, warmup_steps):
    torch.cuda.empty_cache()

    dev_evaluator = evaluation.InformationRetrievalEvaluator(
        df_query_dev.groupby('qid')['query'].apply(str).to_dict(),
        df_passage.groupby('pid')['passage'].apply(str).to_dict(),
        df_qppair_dev.groupby('qid')['docid'].apply(set).to_dict(),
        accuracy_at_k=[10],
        precision_recall_at_k=[10],
        map_at_k=[10], mrr_at_k=[10]
    )

    # multi-task training
    print("train_batch_size", batch_size)
    bi_encoder.fit(
        train_objectives=train_objectives,
        evaluator=dev_evaluator,
        epochs=num_epochs,
        evaluation_steps=301,
        warmup_steps=warmup_steps,
        output_path="/kaggle/working/fine_tuned_encoder"
    )

    return bi_encoder
```



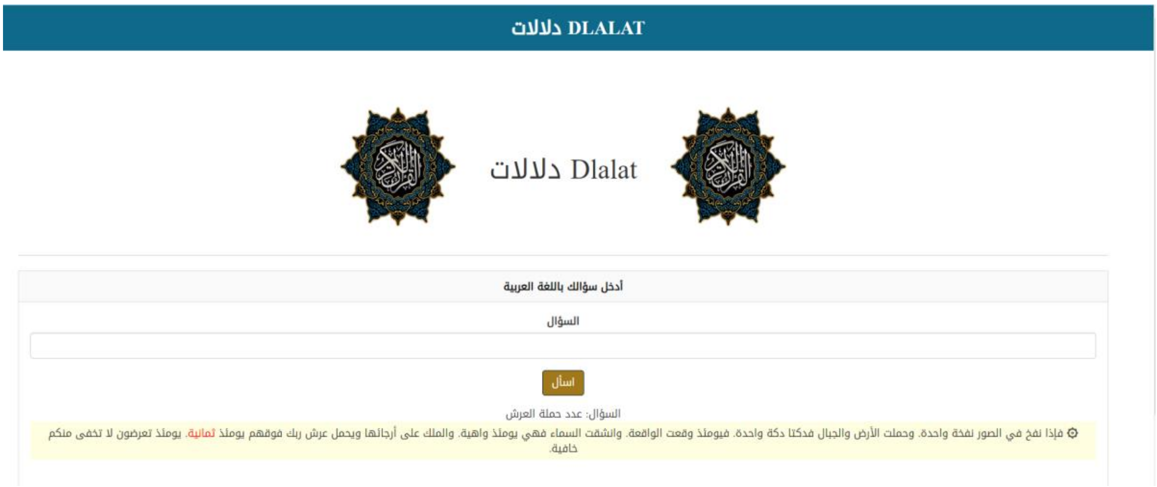
```
def train_QA():
    print(device)
    print(model)
    train_set_file = "/kaggle/input/taskb-data/QQA23_TaskB_qrcd_v1.2_train_preprocessed.jsonl"
    dev_set_file = "/kaggle/input/taskb-data/TaskB Data/QQA23_TaskB_qrcd_v1.2_dev_preprocessed.jsonl"

    train_passage_question_objects  = load_jsonl(train_set_file)
    dev_passage_question_objects = load_jsonl(dev_set_file)

    train_dataset = create_dataset(train_passage_question_objects)
    dev_dataset = create_dataset(dev_passage_question_objects)
    my_dataset_dict = datasets.DatasetDict({"train":train_dataset, "dev" : dev_dataset})
    tokenized_ds = my_dataset_dict.map(prepare_train_features, batched=True, remove_columns=my_dataset_dict["train"].column_names)

    args = TrainingArguments(
        f"result",
        evaluation_strategy = "steps",
        learning_rate=lr,
        per_device_train_batch_size=batch_size,
        per_device_eval_batch_size=batch_size,
        num_train_epochs=epoch,
        weight_decay=0.0001,
        save_strategy = "steps",
        save_steps=1500,
        )


    trainer = Trainer(
    model=ar_model,
    args=args,
    train_dataset=tokenized_ds['train'],
    eval_dataset=tokenized_ds['dev'],
    tokenizer=ar_tokenizer)

    # start training
    trainer.train()

    model_path = "/kaggle/working/"
    trainer.save_model(model_path)

    return ar_tokenizer, ar_model
```



```
#Defining our final model structure
def build_biencoder(sentence_embedder, max_seq_len):
    # Use Huggingface/transformers model (like BERT, RoBERTa, XLNet, XLM-R) for mapping tokens to embeddings
    word_embedding_model = models.Transformer(sentence_embedder, max_seq_length=max_seq_len)
    print("word_embedding_model Max Sequence Length:", word_embedding_model.max_seq_length)
    print("word_embedding_model dimension", word_embedding_model.get_word_embedding_dimension())

    # Apply mean pooling to get one fixed sized sentence vector
    pooling_model = models.Pooling(word_embedding_model.get_word_embedding_dimension())
    print("pooling_model sentence embedding dimension", pooling_model.get_sentence_embedding_dimension())

    dense_model = models.Dense(in_features=pooling_model.get_sentence_embedding_dimension(), out_features=512,
                               activation_function=nn.Tanh())
    bi_encoder = SentenceTransformer(modules=[word_embedding_model, pooling_model, dense_model])

    return bi_encoder
```

Figure (7): Code Implementation.

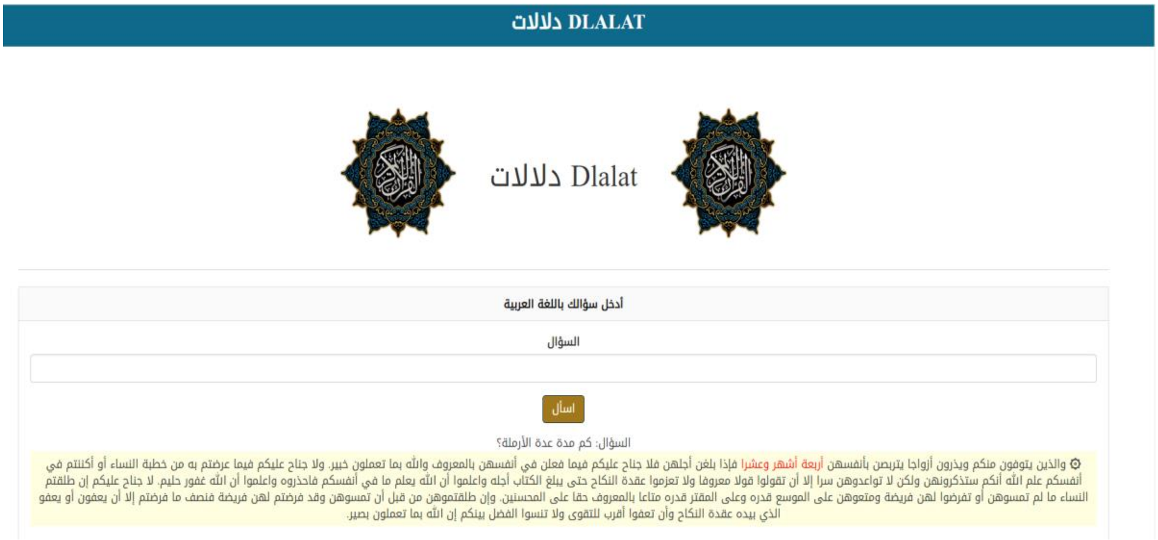
5.3 Graphical User Interface:



السؤال: عدد حملة العرش

⚙ فإذا نفخ في الصور نفخة واحدة. وحملت الأرض والجبال فدكتا دكة واحدة. فيومئذ وقعت الواقعة. وانشقت السماء فهي يومئذ واهية. والملك على أرجائها ويحمل عرش ربك فوقهم يومئذ ثمانية. يومئذ تعرضون لا تخفى منكم خافية.

Figure (8): GUI (1).



السؤال: كم مدة عدة الأرملة؟

⚙ والذين يتوفون منكم ويذرون أزواجا يتربصن بأنفسهن أربعة أشهر وعشرا فإذا بلغن أجلهن فلا جناح عليكم فيما فعلن في أنفسهن بالمعروف والله بما تعملون خبير. ولا جناح عليكم فيما عرضتم به من خطبة النساء أو أكننتم في أنفسكم علم الله أنكم ستذكرونهن ولكن لا تواعدوهن سرا إلا أن تقولوا قولا معروفا ولا تعزموا عقدة النكاح حتى يبلغ الكتاب أجله واعلموا أن الله يعلم ما في أنفسكم فاحذروه واعلموا أن الله غفور حليم. لا جناح عليكم إن طلقتم النساء ما لم تمسوهن أو تفرضوا لهن فريضة ومتعوهن على الموسع قدره وعلى المقتر قدره متاعا بالمعروف حقا على المحسنين. وإن طلقتموهن من قبل أن تمسوهن وقد فرضتم لهن فريضة فنصف ما فرضتم إلا أن يعفون أو يعفو الذي بيده عقدة النكاح وأن تعفوا أقرب للتقوى ولا تنسوا الفضل بينكم إن الله بما تعملون بصير.

Figure (9): GUI (2).

5.4 Summary

In this chapter shows all implementation details for “Dlalat دلالات” including graphical user interface implementation.

The output of this chapter is very necessary for the next chapter, whereby the testing and evaluation take place.

6. Chapter Six: System Testing and Installation

6.1 Introduction:

In this chapter will we make the test system by certified test globally the Heuristic Evaluation Test and Cooperation Evaluation Test. This chapter present introduction in Section 6.1, heuristic evaluation is elaborate in Section 6.2 in addition the cooperative evaluation is stated in Section 6.3, Section 6.4, system installation. Furthermore, this chapter is finally summarized in Section 6.5.

6.2 Heuristics Evaluation:

Table (4): Summary of Violations by Heuristics.

Heuristic Numbering Scheme	Frequency	Ratio (%)
H1	5	7.576
H2	5	7.676
H3	5	7.676
H4	10	15.152
H5	7	10.606
H6	6	9.09
H7	8	12.121
H8	5	7.576
H9	5	7.576
H10	10	15.152
Total	**66**	**100%**

Table (5): Summary of Violations by Severity Rating For Participant (1).

Severity Rating	Frequency	Ratio (%)
0	15	0.23
1	12	0.181
2	10	0.151
3	9	0.14
4	20	0.30
Total	66	100%

- For more details, refer to Appendix A.

Table (6): Summary of Violations by Severity Rating for Participant (2).

Severity Rating	Frequency	Ratio (%)
0	9	0.14
1	12	0.181
2	15	0.23
3	10	0.151
4	20	0.30
Total	66	100%

- For more details, refer to Appendix A.

Table (7): Summary of Violations by Severity Rating for Participant (3).

Severity Rating	Frequency	Ratio (%)
0	12	0.181
1	10	0.151
2	15	0.23
3	9	0.14
4	20	0.30
Total	66	100%

- For more details, refer to Appendix A.

6.3 Cooperative Evaluation:

After using the tool and answering the Dlalat دلالات test, please indicate the extent to which you agree or disagree with each of the following statements regarding your experience with the system.

- Cooperative Evaluation:

Table (8): Participants Details

No.	Criteria	Participant 1	Participant 2	Participant 3
1.	**Gender**	Male	Female	Male
2.	**Age**	21	22	21
3.	**Educational Level**	Bachelor's	Bachelor's	Bachelor's
4.	**Programmer Taken**	Abdullah	Omar	Abdulelah
5.	**Institution**	JU	JU	JU

- **Pre-Evaluation Procedures:**

Participants were contacted through telephone conversations asking them the possibility to participate in the co-operative evaluation. A brief introduction to Dlalat دلالات was given to the participants 10 minutes before they started the evaluation, and participants were asked to read that introductory document. The document also has a list of tasks, which will be performed by the participants throughout the co-operative evaluation. Users were told that they need to think aloud when facing any problem in the system. They were also told that each task they perform is monitored and timed.

- **Evaluation Procedures:**

During the evaluation session, a moderator accompanied the users to do the co-operative evaluation. A comment shown in Appendix B was used by the moderator to write down the comments of each user for each task. Users were helped when they really face serious problems performing the tasks. The following tables show the comments pre-pared by the moderator for each participant.

Table (9): Cooperative Evaluation for Dlalat دلالات for Participant (1).

Task No.	Test	Time Taken to Complete the Task	Comments
A.	**Users**		
1.	Inquiry Process	120 seconds	

Table (10): Cooperative Evaluation for Dlalat دلالات for Participant (2).

Task No.	Test	Time Taken to Complete the Task	Comments
A.	**Users**		
1.	Inquiry Process	120 seconds	

Table (11): Cooperative Evaluation for Dlalat دلالات for Participant (3).

Task No.	Test	Time Taken to Complete the Task	Comments
A.	**Users**		
1.	Inquiry Process	120 seconds	

- It is important to compare the time taken by each participant to complete each single task com-pared to the default time allocated by the moderator as shown in next table.

Table (12): Task Completion Times in Minutes and Seconds.

Task No.	Default	Participant 1	Participant 2	Participant 3
A. Users Activity				
1.	120 seconds	120 seconds	120 seconds	120 seconds
Total Completion Time	120 seconds	120 seconds	120 seconds	120 seconds

- **Post-Evaluation Procedures:**

After completing the co-operative evaluation, participants were given a post-test questionnaire to fill in, which is shown in Appendix B. This questionnaire was important to capture their thoughts and feelings about Dlalat دلالات while they were still fresh. The questionnaire was then followed by a short interview and discussion, which mainly focused on the initial modified design of the Dlalat دلالات. Table (13) shows the responses of the 3 participants to the post-test questionnaire.

Table (13): Participants' Responses to the Post-Test Questionnaire.

No.	Statement	Participant 1	Participant 2	Participant 3	Average
1	Is the system stable?	5	4	4	**4.3**
2	Is the system ease of use?	5	5	5	**3**
3	Are the functionality of the system achieve user's needs?	5	4	5	**4.6**
Average		**5**	**4.3**	**4.6**	

6.4 System Installation:

- **Apache XAMPP Server For Windows:**

  https://www.apachefriends.org/index.html.

- **Flowcharts Online Maker:**

  https://draw.io/

- **Smart Sheets Online Maker:**

  https://www.smartsheet.com/

6.5 Summary:

This chapter showed intestine testing and evaluation for Dlalat دلالات. The heuristic evaluation was conducted system with 3 expert users.

The Heuristic and cooperative evaluation have also shown competitive and acceptable performance for the system indicating that the system is easy to use and has fewer usability problems.

Chapter Seven: Project Conclusion and Future Work

7.1 Introduction:

The conclusion chapter encapsulates the journey of developing a web chatbot designed to provide insights into the Holy Qur'an. This innovation harnesses the power of the BERT model to interpret questions in Modern Standard Arabic (MSA) and offer ranked Qur'anic passages as responses, blending technological advancement with spiritual enlightenment.

7.2 Overall Weaknesses:

While the chatbot represents a significant stride in digital religious studies, it exhibits certain limitations. These include challenges in comprehending the deep contextual and theological nuances of the Qur'anic text, potential biases in the BERT model, and the inherent complexities of translating sacred texts into a format understandable by AI. Moreover, ensuring the respect and reverence due to the sacred text in every interaction remains a delicate balance.

7.3 Overall Strengths:

Despite these challenges, the chatbot stands as a testament to technological innovation in the service of religious education. Its strengths lie in its ability to make Qur'anic teachings more accessible to a wider audience, provide quick and relevant responses to users’ queries, and support a diverse range of question types. Additionally, its user-friendly interface and the integration of advanced AI technology make it a pioneering tool in religious studies and digital humanities.

7.4 Future Work:

Looking ahead, several enhancements are envisioned for the chatbot. These include improving its understanding of complex theological concepts, expanding its database to include more diverse interpretative readings of the Qur'an, and integrating more languages to reach a broader audience. Additionally, continuous updates to the BERT model and user feedback mechanisms will be crucial in evolving the chatbot's accuracy and relevance. Collaborations with Islamic scholars and technologists could further refine its capabilities, ensuring it remains a respectful and informative portal to Qur'anic wisdom.

7.5 Summary:

In this chapter, we wrote weaknesses and strengths in order with the hope to reduce the weaknesses and focus on future work.

References

MALHAS, R.R., 2023. ARABIC QUESTION ANSWERING ON THE HOLY QUR'AN (Doctoral dissertation).

Malhas, R. and Elsayed, T., 2020. Ayatec: building a reusable verse-based test collection for arabic question answering on the holy qur’an. ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 19(6), pp.1-21.

Mellah, Y., Touahri, I., Kaddari, Z., Haja, Z., Berrich, J. and Bouchentouf, T., 2022, June. LARSA22 at Qur’an QA 2022: Text-to-Text Transformer for Finding Answers to Questions from Qur’an. In Proceedinsg of the 5th Workshop on Open-Source Arabic Corpora and Processing Tools with Shared Tasks on Qur'an QA and Fine-Grained Hate Speech Detection (pp. 112-119).

Chai, Christine. (2022). Comparison of text preprocessing methods. Natural Language Engineering.

Budnik, Ruslan. (2023). Risks and Prospects of Creativity Tokenization. Journal of Digital Technologies and Law.

Ghosh, Kripabandhu & Bhattacharya, Arnab. (2017). Stopword Removal: Why Bother? A Case Study on Verbose Queries.

Wibowo, Sastya & Toyib, Rozali & Muntahanah, Muntahanah & Darnita, Yulia. (2022). Time complexity in rejang language stemming.

Akram, Muhammad & Moosa, Wajid & Najiba, (2023). From Data Quality to Model Performance.

Emmert-Streib, Frank & Moutari, Salissou & Dehmer, Matthias. (2023). Model Selection

Kuo, Kevin & Thaker, Pratiksha & Khodak, Mikhail & Ngyuen, John & Jiang, Daniel & Talwalkar, Ameet & Smith, Virginia. (2022). On Noisy Evaluation in Federated Hyperparameter Tuning.

Training, Tonex & Masum, Mostafizur Rahman. (2023). Scientific Foundations of Test and Evaluation Training by Tonex.

Elov, Botir & Khamroeva, Shahlo & Khusainova, Zilola. (2023). The pipeline processing of NLP. E3S Web of Conferences.

Kanwal Mehreen, KDnuggets, (April 18, 2023), A Guide to Top Natural Language Processing Libraries.

APPENDIX A

Heuristic Evaluation – A System Checklist

Disclaimer: This list is a simplified one of the original list which was developed by Xerox Corporation (© Usability Analysis & Design, Xerox Corporation, 1995) and was downloaded from ftp://cs.uregina.ca/pub/class/305/lab2/example-he.html. It has been simplified to suite the purpose it is used for, which is to evaluate the Dlalat دلالاتin order to identify current problems as experienced by the users, which is part of our graduation project that is submitted to King Abdullah II School for Information Technology, The University of Jordan. The number of questions was reduced; however, the individual questions were left intact.

Please fill in the evaluation form below, which is a form of checklist, by writing “X” in the appropriate place which mostly describes the best answer to the corresponding criterion. This form is to be filled after you have investigated the system interface i.e. have looked at, and examined the interface. The answer to each criterion is either:

- "**0**" which means “I don’t agree that this is a usability problem at all”.
- "**1**" which means “Cosmetic problem only: need not be fixed unless extra time is available on project”.
- "**2**" which means “Minor usability problem: fixing this should be given low priority”.
- "**3**" which means ”Major usability problem: important to fix, so should be given high priority”.
- "**4**" which means ”Usability catastrophe: imperative to fix this before product can be released".

Thank you for your willingness to evaluate this system. Your time and effort are highly appreciated.

H1. Visibility of System Status

The system should always keep user informed about what is going on, through appropriate feedback within reasonable time.

Number	Review Checklist	0 1 2 3 4	Comments
1.1	Does every display begin with a title or header that describes screen contents?	( ) ( ) ( ) ( ) ( )	
1.2	Do menu instructions, prompts, and error messages appear in the same place(s) on each menu?	( ) ( ) ( ) ( ) ( )	
1.3	Is there some form of system feedback for every operator action?	( ) ( ) ( ) ( ) ( )	
1.4	Are responses times appropriate to the users cognitive processing?	( ) ( ) ( ) ( ) ( )	
1.5	Is there visual feedback in menus or dialog boxes about which choices are selectable?	( ) ( ) ( ) ( ) ( )	

H2. Match between System and the Real World

The system should speak the user’s language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

Number	Review Checklist	0 1 2 3 4	Comments
2.1	Are icons concrete and familiar?	( )( )( )( )( )	
2.2	Are menu choices ordered in the most logical way, given the user, the item names, and the task variables?	( )( )( )( )( )	
2.3	Do related and interdependent fields appear on the same screen?	( )( )( )( )( )	
2.4	When prompts imply a necessary action, are the words in the message consistent with that action?	( )( )( )( )( )	
2.5	On data entry screens, are tasks described in terminology familiar to users?	( )( )( )( )( )	

H3. User Control and Freedom

Users should be free to select and sequence tasks (when appropriate), rather than having the system does this for them. Users often choose system functions by mistake and will need a clearly marked “emergency exit” to leave the unwanted state without having to go through an extended dialogue. Users should make their own decisions (with clear information) regarding the costs of exiting current work. The system should support undo and redo.

Number	Review Checklist	0 1 2 3 4	Comments
3.1	When a user's task is complete, does the system wait for a signal from the user before processing?	( ) ( ) ( ) ( ) ( )	
3.2	Are users prompted to confirm commands that have drastic, destructive consequences?	( ) ( ) ( ) ( ) ( )	
3.3	Are character edits allowed in data entry fields?	( ) ( ) ( ) ( ) ( )	
3.4	If menu lists are long (more than seven items), can users select an item either by moving the cursor or by typing a mnemonic code?	( ) ( ) ( ) ( ) ( )	
3.5	If the system uses a pointing device, do users have the option of either clicking on menu items or using a keyboard shortcut?	( ) ( ) ( ) ( ) ( )	

H4. Consistency and Standards

Users should not have to wonder whether different words, situations, or actions mean the same thing.

Follow platform conventions.

Number	Review Checklist	0 1 2 3 4	Comments
4.1	Has a heavy use of all uppercase letters on a screen been avoided?	( )( )( )( )( )	
4.2	Are icons labeled?	( )( )( )( )( )	
4.3	Are there no more than twelve to twenty icon types?	( )( )( )( )( )	
4.4	Does each window have a title?	( )( )( )( )( )	
4.5	Is vertical and horizontal scrolling possible in each window?	( )( )( )( )( )	
4.6	Are menu choice lists presented vertically?	( )( )( )( )( )	
4.7	Are menu titles either centered or left-justified?	( )( )( )( )( )	
4.8	Are menu items left-justified, with the item number or mnemonic preceding the name?	( )( )( )( )( )	
4.9	Do embedded field-level prompts appear to the right of the field label?	( )( )( )( )( )	
4.10	Are attention-getting techniques used with care?	( )( )( )( )( )	

H5. Help Users Recognize, Diagnose, and Recover From Errors

Error messages should be expressed in plain language (NO CODES).

Number	Review Checklist	0 1 2 3 4	Comments
5.1	Is sound used to signal an error?	( )( )( )( )( )	
5.2	Are error messages worded so that the system, not the user, takes the blame?	( )( )( )( )( )	
5.3	Do error messages suggest the cause of the problem?	( )( )( )( )( )	
5.4	Do error messages indicate what action the user needs to take to correct the error?	( )( )( )( )( )	
5.5	If the system supports both novice and expert users, are multiple levels of error-message detail available?	( )( )( )( )( )	
5.6	If an error is detected in a data entry field, does the system place the cursor in that field or highlight the error?	( )( )( )( )( )	
5.7	Do error messages inform the user of the error's severity?	( )( )( )( )( )	

H6. Error Prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place.

Number	Review Checklist	0 1 2 3 4	Comments
6.1	Are menu choices logical, distinctive, and mutually exclusive?	( ) ( ) ( ) ( ) ( )	
6.2	Are data inputs case-blind whenever possible?	( ) ( ) ( ) ( ) ( )	
6.3	Does the system prevent users from making errors whenever possible?	( ) ( ) ( ) ( ) ( )	
6.4	Does the system warn users if they are about to make a potentially serious error?	( ) ( ) ( ) ( ) ( )	
6.5	Do data entry screens and dialog boxes indicate the number of character spaces available in a field?	( ) ( ) ( ) ( ) ( )	
6.6	Do fields in data entry screens and dialog boxes contain default values when appropriate?	( ) ( ) ( ) ( ) ( )	

H7. Recognition Rather Than Recall

Make objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Number	Review Checklist	0 1 2 3 4	Comments
7.1	For question and answer interfaces, are visual cues and white space used to distinguish questions, prompts, instructions, and user input?	( ) ( ) ( ) ( ) ( )	
7.2	Are inactive menu items grayed out or omitted?	( ) ( ) ( ) ( ) ( )	
7.3	Do data entry screens and dialog boxes indicate when fields are optional?	( ) ( ) ( ) ( ) ( )	
7.4	Are prompts, cues, and messages placed where the eye is likely to be looking on the screen?	( ) ( ) ( ) ( ) ( )	
7.5	Are field labels close to fields, but separated by at least one space?	( ) ( ) ( ) ( ) ( )	
7.6	Have items been grouped into logical zones, and have headings been used to distinguish between zones?	( ) ( ) ( ) ( ) ( )	
7.7	Are borders used to identify meaningful groups?	( ) ( ) ( ) ( ) ( )	
7.8	Is color coding consistent throughout the system?	( ) ( ) ( ) ( ) ( )	

H8. Flexibility and Minimalist Design

Accelerators-unseen by the novice user-may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions. Provide alternative means of access and operation for users who differ from the "average" user (e.g., physical or cognitive ability, culture, language, etc.)

Number	Review Checklist	0 1 2 3 4	Comments
8.1	If menu lists are short (seven items or fewer), can users select an item by moving the cursor?	( )( )( )( )( )	
8.2	If the system uses a pointing device, do users have the option of either clicking on fields or using a keyboard shortcut?	( )( )( )( )( )	
8.3	On data entry screens, do users have the option of either clicking directly on a field or using a keyboard shortcut?	( )( )( )( )( )	
8.4	On menus, do users have the option of either clicking directly on a menu item or using a keyboard shortcut?	( )( )( )( )( )	
8.5	In dialog boxes, do users have the option of either clicking directly on a dialog box option or using a keyboard shortcut?	( )( )( )( )( )	

H9. Aesthetic and Minimalist Design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Number	Review Checklist	0 1 2 3 4	Comments
9.1	Are all icons in a set visually and conceptually distinct?	( ) ( ) ( ) ( ) ( )	
9.2	Does each icon stand out from its background?	( ) ( ) ( ) ( ) ( )	
9.3	Does each data entry screen have a short, simple, clear, distinctive title?	( ) ( ) ( ) ( ) ( )	
9.4	Are field labels brief, familiar, and descriptive?	( ) ( ) ( ) ( ) ( )	
9.5	Are there pop-up or pull-down menus within data entry fields that have many, but well-defined, entry options?	( ) ( ) ( ) ( ) ( )	

H10. Help and Documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Number	Review Checklist	0 1 2 3 4	Comments
10.1	Are on-line instructions visually distinct?	( )( )( )( )( )	
10.2	If menu choices are ambiguous, does the system provide additional explanatory information when an item is selected?	( )( )( )( )( )	
10.3	Is the help function visible; for example, a key labeled help or a special menu?	( )( )( )( )( )	
10.4	Navigation: Is information easy to find?	( )( )( )( )( )	
10.5	Presentation: Is the visual layout well designed?	( )( )( )( )( )	
10.6	Conversation: Is the information accurate, complete, and understandable?	( )( )( )( )( )	
10.7	Is the information relevant?	( )( )( )( )( )	
10.8	Can users easily switch between help and their work?	( )( )( )( )( )	
10.9	Is it easy to access and return from the help system?	( )( )( )( )( )	
10.10	Can users resume work where they left off after accessing help?	( )( )( )( )( )	

Appendix B

Cooperative for Dlalat دلالات

Task No.	Test	Time Taken to Complete the Task	Comments
A.	**Users**		
1.	Inquiry Process		

APPENDIX C

Dlalat دلالات Usability Test (Post-test Questionnaire)

Gender: M / F **Age:** _____

Educational Level: _______________________________

Programmer Taken: _____________________________

Institution: ____________________________________

After using the system and answering Dlalat دلالات test, please indicate the extent to which you agree or disagree with each of the following statements regarding to your experience with the system.

No.	Statements	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
		1	2	3	4	5
1.	Is the app stable?					
2.	Is the app ease of use?					
3.	Are the functionality of the app achieve user's needs?					