Addis Ababa University

College of Natural Sciences

Dialogue System for Advising Ethiopian Public Universities Students

Dereje Tsegab Habte

A Thesis Submitted to the Department of Computer Science in Partial Fulfillment for the Degree of Master of Science in Computer Science

Addis Ababa, Ethiopia

August, 2017

Addis Ababa University

College of Natural Sciences

Dereje Tsegab Habte

Advisor: Yaregal Assabie (PhD)

This is to certify that the thesis prepared by *Dereje Tsegab Habte*, titled: *Dialogue System for Advising Ethiopian Public Universities Students* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

Name	Signature	Date
Advisor:		
Examiner:		
Examiner:		

Abstract

Academic advisors assist students in academic matters such as registration, warning, probation and other personal matters. Successful advising systems increase student retention, improve graduation rates and help students meet educational goals. Research in academic advising shows a trend of integrating electronic technologies that advisors must adopt to increase their roles in student education, while resources are limited and the proportion of students increases. The main objective of this research is to design a model of academic advising dialogue system for Ethiopian public universities.

We developed Higher Education Students Advising System (HESAS) as academic advising dialogue system, to allow advising session to be more centered on students and to help advisors and advisees in making better informed-decision and improved services. The design of HESAS is composed of six main components. These are user interface, spelling corrector, natural language understanding, dialogue manager, response generator and advisement service. The students used the system during the experimental phase. For the second experimental phase, the system performed well, obtaining close to 83%, on the traditional language processing measures of precision, recall and F1 score.

Keywords: natural language processing, dialogue system, conversational agents, academic advising

Dedication

I dedicate this research work to God Almighty my creator, my strong pillar, my source of inspiration, wisdom, knowledge and understanding. He has been the source of my strength throughout this research and on His wings only have I soared.

Acknowledgments

First and foremost, I would like to thank the Almighty God for His support and being with me in all directions of my life.

Next, I would like to express my sincere gratitude to my advisor Dr. Yaregal Assabie for his support, encouragement, guidance, constructive comments and suggestions throughout this research work. It is with his continuous follow up, encouragement and advice that this research came to realization. I appreciate his dedication from the beginning to the end to teach me for the first time how to conduct and write a research.

I would also like to take this opportunity to express my profound gratefulness to my mother Belaynesh Hamaro, my father Tsegab Habte, my sisters Tsedalu Tsegab, Beletu Tsegab, Bereket Tsegab and Addisalem Tsegab, for their continuous moral. My families have always given me their encouragement and support, I appreciate deeply. My special thank goes to my brother artist Dawit Tsegab (Dech) for supporting and preparing whatever I need. Dech, I never forget what you did for me. Thank you my dear brother and may God bless you forever and ever.

I thank, those two of my friends who were always with me while the thing goes tough: Eshetu Gusare and Sebahadin Nasir. Thank you my friends. I also thank my friends Mitiku Bajura and Mohammed Yegoraw for their encouragements.

I would also like to extend my gratitude to Jijiga University for sponsoring me to study my second degree. I also want to thank Graduate Committee of Department of Computer Science at Addis Ababa University for giving me the permission to do my research work. I also thank all of my classmates for working together in different projects and assignments in harmony.

Finally, I would like to thank all those people who have fruitful and valuable assistance and their finger prints in this research work. May God bless you all!

Table of Contents

List of Tables

List of Figures

List of Algorithms

Acronyms and Abbreviations

AI	Artificial Intelligence
AIML	Artificial Intelligence Markup Language
BDI	Beliefs, Desire and Intentions
CA	Conversational Agent
CS	ChatScript
CSS	Cascading Style Sheet
DB	Database
DM	Dialogue Manager
HESAS	Higher Education Students Advising System
FAQs	Frequently Asked Questions
FN	False Negative
FP	False Positive
FSA	Finite State Automata
GB	Giga Bit
HTML	Hyper Text Markup Language
KB	Knowledge Base
KE	Keyword Extraction
LU	Lexical Unit
MySQL	My Structured Query Language
NLG	Natural Language Generation
NLP	Natural Language Processing
NLU	Natural Language Understanding
ON	Outcome Negative

POS	Part of Speech
RAM	Random Access Memory
RG	Response Generator
SQL	Structured Query Language
TN	True Negative
TP	True Positive
XML	Extensible Markup Language

Chapter 1: Introduction

1.1 Background

Until the early 1900s, Ethiopia has a base of traditional education (Church) with its own script which is very elaborate and complex [1]. The development of modern education in Ethiopia is a recent phenomenon. The first public school to provide western style education was the Ecole Imperiale Menelik II, which was opened in October 1908. That the same year, Emperor Menelik II established a primary school in Harar. The regional governors also opened schools in Yirgalem, Gore and Harar [2].

In the 1950s, there were two universities founded in Ethiopia, i.e., Haile Selassie I University, which was established in 1950 [1] and Asmara University, was founded on December 20, 1958 [3]. By 2007, there were 7 universities which were expanding and 13 new public universities had started construction [1]. By 2012, the number of public universities had risen to 33.

Between 2008/09 and 2012/13, undergraduate enrollment for regular, evening, summer and distance programs had increased in both public (86%) and private universities from 310,702 to 553,484 [1]. First degree graduate increased from 56,109 to 79,073.

Nowadays, there are a total of 161 universities and colleges in Ethiopia including both public and private institutions [18]. The number of students enrolled in both public and private institutions has also increased. However, in most of public institutions students will not persist to graduation in the institution they began their studies. There are various reasons why students do not persist to graduation [4]. They may be experiencing financial difficulties, family problems, loneliness, or personal crises. Other reasons may also include poor academic performance, lack of self-confidence, poor management of studies and time, and lack of comprehensive academic advising.

Academic advisors can help retain students through their guidance and positive influence on the students [4]. Kulik *et al.* [6] studied the effects of four types of programs on the retention of college students. The four programs included (1) study and academic skills instruction, (2) academic advising and counseling, (3) academic support programs, and (4) enrollment in developmental and

remedial courses. They found that students who participated in the programs had a retention rate that was eight percent higher than those who didn't participate in the programs.

Ramirez and Evans [8] also identified several factors that contribute to attritions for minority students. These factors include inappropriate course selection, poor scheduling, low use of support services, and lack of comprehensive advising.

In higher education, academic advisors assist students' academic, professional, social, and personal matters [5]. Academic advising was initially a response for a need to guide students in selecting their course sequence and later evolved to include assistance in matter. The innovation trend in advising has been towards the use of communication technology such as email, social networking, instant message, podcast, mobile application, online videos and blogs [9, 10]. A quick survey of university websites shows that many have introduced the concept of eAdvising, that is, utilizing electronic means, usually web-based, to offer advising to students [11, 13].

White and Leonard [14] suggested that interactive advising modules for personalized discussions could be created using artificial intelligence (AI) techniques. Leonard [9], detailed the profound effect of technology use by advisor and referenced the idea of an expert-based interactive advising modules as possible future trend, but few institutions have shown interest in developing such a system. The reason is that it takes much longer to develop an expert-based, interactive advising module than it does to create, for example, online course registration [5]. An automated system is better solution for the innovation trend in advising and addresses the concern of not enough advisors, too many students, in an economically challenging environment [5]. Such a system will lessen the burden of academic advisors from several mundane tasks and free up more of their time for the deeper aspects of advising, such as career planning or managing extraordinary personal situations. To deliver improved academic advising services dialogue system for advising Ethiopian public universities students will be proposed.

Dialogue system or conversational agent (CA) is a computer program designed to carry out an open and coherent conversation with a human [15]. Dialogue systems have employed text, speech, graphics, gestures and other modes for communication. For the purpose of this research work a

text-based dialogue system will be used and advice students on curricula and extra-curricular related issues such as preparing students to plan, set goals, and make decision.

1.2 Motivation

The lack of proper advisement service will result in students to leave the university. Student advising is very important for higher education. Successful advising programs increase student retention, improve graduation rates and help students meet education goals [5]. There are two ways to deliver advising services—face-to-face meetings and technology [7]. In Ethiopian public universities the face-to-face advising service is usual. This advising service has limitations, for example, it doesn't address many students, increases the burden of academic advising, and time consuming. With the help of technology we can improve those limitations. The use of technology in academic advising may introduce a greater accountability and may provide better services to the students. According to Kramer and McCauley [17], technology in academic advising enables administrators to be student centered and it is helpful to advisors and advisees in that it contributes to assisting in making better-informed decision and improved services. Advising can be a very time consuming process leading to the need for automating some of its functions. Ideally, an automated advisor gives answers to a student's routine questions. The student can then meet with the advisor for further consultation. This combination of human and machine can save time for the human advisor [20]. This has motivated us to develop dialogue system for advising Ethiopian public universities students.

1.3 Statement of the Problem

Academic advising at higher education has become difficult for both advisors and students. Advising may be given to students at higher education mostly during registration as well as when students face different challenges. The task of student registration is time consuming for the academic advisor. During registration, students must first decide, together with their academic advisor, which courses they take to be able to proceed with the rest of the registration process. Each semester this process becomes cumbersome for the advisor because the advisor should follow up curriculum changes, should advise too many students, should know every aspect of academic

regulations for the students and follow up all the updates related to it, and should follow all the courses offered and the course schedule in the current semester.

The other difficulty in academic advising at higher education is advising of students who face various problems. In higher education, students face various problems such as academic dismissal, probation, warning, withdrawals, and other personal situations. These problems will increase attrition rate and decrease graduation rate. Research literature on student retention suggests that the contact with a significant person, that is, the person who is responsible for advising students, within an institution of higher education is a crucial factor in a student's decision to remain in the university [16]. Higher education academic advisors play a great role in the reduction of the number of students dismissed, probation, and warnings.

Though a lot of works have been done in the field of academic advising by many universities around the world, but in most of Ethiopian public universities, a dialogue system supporting students advising is not given much attention. This work will be done to decrease the gap between the advisor and advisees, to increase the number of students retention rate, to improve the current innovation trend in advising at Ethiopian public universities, to create an environment that students communicate as freely and openly as with their actual advisor using English language, and to allow students ask a wider range of questions and obtain immediate responses to these instead of waiting for peers or advisor to read and reply.

1.4 Objectives

General Objective

The main objective of this research is to design a model of academic advising dialogue system for Ethiopian public universities.

Specific Objectives

To meet the general objective, the following specific objectives are set in this research work.

- Studying and exploring the areas related to academic advising dialogue system.
- To model the linguistic features of English language.
- To collect curriculum and other related documents of Ethiopian public universities.

- Build a knowledge base from the collected documents.
- Designing the architecture of dialogue system for academic advising.
- Developing the prototype of dialogue system for advising Ethiopian public universities students.
- Conducting experiments to evaluate the performance of the proposed system.

1.5 Methods

The following methods will be used in order to achieve the above specified objectives.

Literature Review

An extensive review of relevant literatures will be done in order to acquire a deeper understanding of the research area and its problem domain. Previous research in the areas of academic advising dialogue system will be investigated to visualize the importance towards this research.

Data Source

The data to be used for knowledge base will be collected from different sources such as: curriculum, administrative policies and procedures document, students, and legislations of Ethiopian public universities.

Prototype Development and Evaluation

The proposed system should have a prototype development that contains a natural language interface that allows students to communicate as freely and openly as with their advisors.

The developed prototype will be evaluated. The evaluation of the developed prototype will be done iteratively to increase its performance. The errors encountered during the experimentation will be corrected and the experimentation will be done iteratively until the result is found to be satisfactory.

1.6 Scope and Limitations

Designing a dialogue system that can fully emulate human is a difficult task in Natural Language Processing (NLP). A dialogue system requires studies in linguistics and psychology to create a realistic human interaction experience. The proposed dialogue system is limited to linguistic study.

Basically, this research work will focus on advising undergraduate students of the Department of Computer Science at Ethiopian Public Universities related to both curricular and extra-curricular issues such as preparing students to plan, set goals, and make decision.

1.7 Application of Results

The result of this research work has the following contributions to the academic advising field.

- It allows Ethiopian public universities students to get advisement service anywhere and anytime.
- It protects the privacy of the students.
- Helps students to prepare plan and set goals.
- Allows students to make better-informed decision.
- Lessens the burden of academic advisors.

1.8 Organizations of the Rest of the Thesis

The rest of this research work is organized as follows:-

- Chapter 2: As the starting point, this chapter introduces about the models of academic advising and dialogue systems. The chapter describes the typical architecture or components of dialogue systems and the approaches to dialogue management.
- Chapter 3: This chapter discusses some related works that have been done on the areas of academic advising dialogue systems.
- Chapter 4: This chapter presents the detailed discussion on the models of proposed systems and the functions of each of its components.
- Chapter 5: This chapter presents the detail implementation of each of the components of the proposed system and the evaluations of the developed prototype.
- Chapter 6: This chapter concludes the thesis with the contributions of the research and future research directions.

Chapter 2: Literature Review

2.1 Introduction

In this Chapter, we present a review of literature that are related to this research work. We begin in Section 2.2 with the definition of few key concepts and terms. In Section 2.3, we present a review of academic advising and its model. Section 2.4, provides an introduction to natural language processing, the main problem involved in extracting meaning from human language and examines the kind of activities performed by NLP system. Section 2.5, Dialogue System, presents a brief introduction to dialogue system. Section 2.6, deals with an introduction to theories of dialogue management. Section 2.7, spelling error and correction, discusses the spelling errors as well as detection and correction of spelling error.

2.2 Preliminaries

In order to make the rest of this research coherent, a few concepts need to be explained, and clarified. This research work deals with the aspects of language technology. ***Language Technology*** is the process of analyzing textual input in terms of syntax and semantics [15].

Syntax is the structure of a language. It is the grammatical arrangement of words in a sentence to show its relationship to one another in a sentence.

Semantic is meaning of words/phrases/sentences/whole text. Normally semantic is restricted to "meaning out of context"—that is, meaning as it can be determined without taking context into account [15].

The system described herein is dialogue system that is limited to processing of written—or typed—interaction.

An ***utterance*** is a generic term referring to any mode of communication. That is, an utterance may be typed or signed for example [25].

A ***dialogue*** is a joint activity between two entities (e.g., a user and computer system). What makes dialogues different from other types of discourse is that they are concerned with grounding and turn-taking.

Grounding is to establish a base of mutual understanding, or common ground, well enough for the current purposes. In the case of dialogues, this often attributes to confirmation of understanding of a conversational counterpart's utterance. ***Turn-taking*** in a dialogue can be strict (no interruption is allowed) or flexible. For the purpose of this research, the interaction is typed on a keyboard, and the system described herein enforces strict turn-taking (i.e., interruption and parallel utterance are not allowed).

Domain is an area of knowledge, for a specific system. A ***genre*** is type of discourse with certain specific features and content.

2.3 Academic Advising

Academic advising has been part of higher education for many years. Initially in the higher education system, students followed a very structured schedule that was typically set up by their faculty advisor [4]. In the late 1800s, with the onset of the elective system, students were given more freedom to choose their classes. An academic advisor, assisted students in selecting what course they should take. In general, advisors focus curricular issues, including information and graduation requirements.

In Ethiopian public universities, academic advisors help students clarify their academic goals, assist students with institutional policies and procedures, and help acclimate students to university life.

Models of Academic Advising

From the literature, we select four models for academic advising [16]. These are prescriptive advising model, developmental advising model, integrated advising model, and engagement advising model.

Prescriptive Advising Model: The prescriptive advising model is characterized by an authoritarian relationships in which students follow the prescriptive regimen of their advisor concerning course selection, degree requirements, and registration, without assuming responsibility for decision making [21].

Developmental Advising Models: The developmental advising models rely on a shared responsibility between the students and the advisor in which the advisor directs the student to proper resources thus, facilitating the development of greater independence, decision making, and problem solving [22].

Integrated Advising Model: The integrated advising model combines elements of both prescriptive and developmental advising models [23].

Engagement Advising Model: The engagement advising model involves building a relationship between the student and advisor to enhance student self-efficacy for completing the degree requirements.

2.4 Natural Language Processing

In our time of information age, inspired by the application of computers, vast number of disciplines are inheriting the ideology of computers to help them carry-out detail studies in their domain. The use of computers even opened a track for the inauguration of new fields. One of them, resulted from the combination of computer science, linguistics and cognitive psychology disciplines, is Natural Language Processing (NLP). As a result of this, NLP is given different names like speech and language processing, human language technology, computational linguistics, and speech recognition and synthesis [15].

NLP is a theoretically motivated range of computational techniques for analyzing and representing naturally occurring texts at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of tasks or applications [26]. In NLP, the contribution of Computer Science is to develop an algorithm that specifies the internal representation of data and define the way they can be efficiently processed in a computer.

Basically, engaging in complex language behaviour requires various kinds of knowledge of language [15, 27]. This language knowledge is also called, as in [26], 'levels of language approach'. NLP must possess considerable knowledge about the language under consideration. Naturally, languages contain a number of ambiguities at one or more of the above levels.

One can say that input is ambiguous if there are alternative linguistic structures that can be built for it and ambiguity problem is resolved via disambiguation [27].

The idea of giving computers the ability to process human language is as old as the idea of computers themselves. Research in natural language processing has been going on for several decades dating back to the late 1940s [26]. During those times, it was a difficult task, even though not impossible, for it is based on human made rules. Preparing rules requires extensive involvement of talented linguistic experts and it is a time consuming work. In recent years, researches in the area of NLP are increasing rapidly.

Among the main reasons behind the scene are [26]:

- An increased availability of large amounts of electronic text
- Availability of computers with increased speed and
- The advent of the Internet

With over sixty years, natural language systems are still very complicated to design and they are still not perfect because human language is complex, and it is difficult to capture the entire linguistic knowledge for hundred percent accuracy in processing [27]. However, NLP technologies are becoming extremely vital in order to ease access into systems, thereby making those systems more user-friendly. As Bose and Ranjit [27] put it, increasing number of companies is investing in and deploying voice or speech recognition and processing technologies at an alarming rate to save money by replacing operators and to improve service to their customers. In the future, NLP is expected to further improve existing systems and hope that we will see technologically intelligent and more user-friendly systems.

NLP provides a good baseline for both theoretical and implementation of a range of applications. In fact, according to [26], any application associated with text is a candidate for NLP. This includes dialogue system, information extraction, text summarization, machine translation, speech recognition, etc.

2.5 Dialogue Systems

Ever since the dawn of the computer era, there has been a dream of creating a machine that is intelligent as a human being [28]. It was thought that the most obvious and natural way for the computer to manifest its intelligence was using natural language, either text or speech. Turing presented a famous test in the year 1950 [29]. The intention of the test is to validate if a machine can pass for being a human being. If so, the machine has accomplished true intelligence. When the test is performed, a person is placed behind a computer screen. He/she can write questions in natural language on the keyboard and answers appeared on the screen. The conversation goes on for a while and after finishing the person has to guess if he/she was talking to a man or a machine. If he/she was talking to a machine but guesses a man, the machine is considered intelligent. That is, intelligence is measured by the performance of the interaction in natural language.

Historically, two different research directions concerned dialogue systems can be identified [28]. The first one tries to develop a theory of dialogue that mimics human dialogue as much as possible. Much work is done in cooperation with human natural language community. Several logics have been proposed as a formal language for dialogue modeling and complex use of dialogue including different kinds of references and complex structures have been given a lot of attention [30, 31].

The second direction addresses the implementation of dialogue systems that can participate in a human dialogue. These systems have traditionally been simpler than the dialogue theory, and the performed dialogue has little in common with dialogue written/spoken between humans. These systems are typically database question-answering systems or frame-filling systems [30]. The dialogue in the former consists of the user asking questions to a database and the latter performs dialogue by asking the user for information and filling in the information gathered in a frame.

In the middle of the nineties, dialogue systems became a popular research area, both in academia and in industry. The interest in dialogue systems has led to better integrated systems where ideas from the dialogue theory have been integrated into actual performing dialogue systems [32].

2.5.1 Types of Dialogue System

In this section, we will give an informal classification of dialogue systems based on the genre of the dialogue that they model. We concentrate on what Allen *et al.* [33] define as practical dialogues, that is, interactions in which the dialogue is focused on accomplishing a concrete task.

Types of dialogue can vary over a number of different features. The initiative in a dialogue refers to which dialogue participant is driving the conversation at some point. The initiative may with the system, with the user, or with both, so-called mixed initiative. In this case both system and user can introduce new topics into the discourse.

Dialogue genres also vary with respect to domain. We now look at some dialogue genres which are subclasses of practical dialogue.

Information-Seeking Dialogue

A common application of dialogue system is as a front-end to a database. Here the dialogue system acts as a natural language interface, and such dialogues are referred to as information seeking dialogues [33]. The system tries to elicit enough information from the user as is needed to search for the information that the user wants. This means that the initiative in information-seeking dialogues is typically with the system.

The dialogues use a relatively restricted language, so that keyword spotting can often be used to extract the content of the user's utterances. Also, the dialogues follow standard patterns.

Examples of dialogue systems which model information-seeking dialogue are ATIS [34], in which a corpus of flight information dialogues has been collected, and the Philips automatic train timetable information system [35].

Negotiation Dialogue

The task of negotiation dialogues is that the dialogues participants come to agreement on an issue. Negotiation dialogues differ from many other user/system interactions because in a negotiation both parties will have their own goals and constraints [33]. An example is VerbMobil [36], which models human-human appointment negotiation dialogues. Negotiation is performed at many levels [37], namely at the domain level, and the meaning level.

Problem-Solving Dialogue

When the user and the system collaborate with the common goal of achieving a complex task, such dialogues are called problem-solving dialogues [33]. The system often models a domain expert who helps the user achieve the task at hand, but there can also be a mixed-initiative when the system introduces goals or informs the user of external events. Due to the collaborative nature of this genre, there will often be negotiation sub-dialogues [38]. Collaborative dialogues also contain many grounding utterance.

The TRAINS project has collected a corpus [39] of problem-solving dialogues in which the user collaborates with a planning assistant to complete a task in a railroad freight system.

The follow on project, TRIPS [40], shows how the model can be applied to a number of collaborative domains [41] including kitchen design and disaster relief management.

Tutorial Dialogue

The task in tutorial dialogue is that the user, or student, learns concepts or techniques in a given domain [33]. The system sets the user a task or an exercise which should be solved, and then aids the user in finding a solution. The initiative in tutorial dialogue is shared between user and system. The system poses the exercise at hand, but the user is free to raise questions for instance about unknown concepts. Both can initiate clarification sub-dialogues.

Tutoring should use flexible natural language in order to be effective [40]. However, the language can be restricted by concentrating on the domain at hand. Tutorial dialogue systems rely heavily on both domain reasoning and general pedagogical strategies to support the tutorial task. On a wider scale, tutorial dialogue can form part of a broader e-learning application.

There are a number of tutorial systems which use a dialogue interface [38]. The PACT Geometry Tutor [42] models tutoring with knowledge construction dialogues for the physics domain. These allow the student to build up his/her own knowledge by conversing with the system. AUTOTUTOR [43] also deals with physics domain, and includes prosodic features and facial expression in the model.

Academic Advising Dialogue

The task in academic advising dialogue is that the students get advisement service in a given domain. The system contains knowledge about the academic programs and policies and answer to a wider range of frequently asked questions in academic advising. An example is Instavice [44], Instant academic advice system, includes information about the academic programs, course schedules, answers to a wide range of academic FAQs, recommendations for the development of a course plan, and refers students to academic services.

The system provides students with an academic advising service that reflects a human interaction experience through an online text application. Albert [19], enhances the academic advising experience by offering students a service that is available at any time. ADS [12], Advising dialogue system, enables the students to get advisement services through natural language interaction.

2.5.2 Components of a Dialogue System

A dialogue system is typically broken down into modules or subsystems that provide the functionality necessary for natural language dialogue [38]. This functionality involves at least natural language understanding, natural language generation, and some interface to external knowledge source. A control module organizes the flow of the dialogue and facilitates communication between modules in such a way that they can interleave correctly. In this section we present the role played by each of these modules. Each modules is linked to a dialogue manager which controls the system. Figure 2.1 show the general architecture of a text-based dialogue system [19].

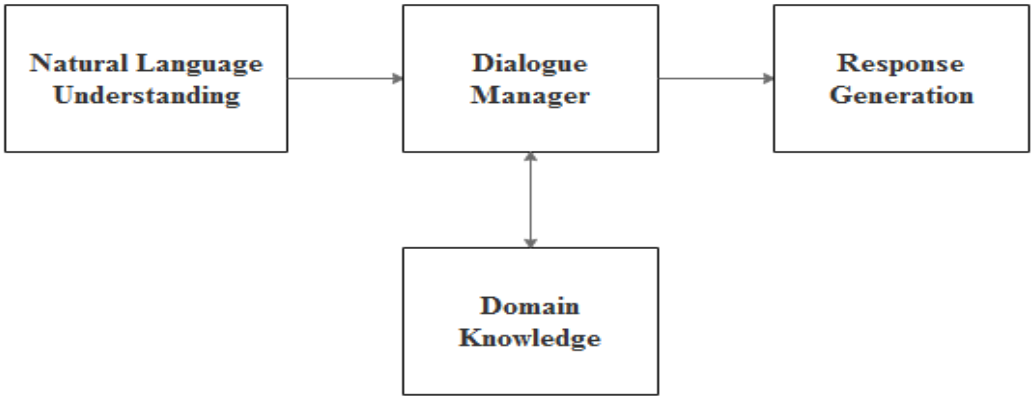


Figure 2.1: General Architecture of a Text-based Dialogue Systems

Natural Language Understanding

Natural language understanding (NLU) is concerned with the analysis of the utterances a user makes in a dialogue. This can be either typed input or the result of speech recognition, depending on what the medium of the dialogue is. The output of NLU is some representation of the meaning of the utterance that can then be used by the dialogue control module. The analysis should account for both the syntax and semantics of the utterance.

The syntactic analysis uses linguistics resources such as a lexicon and a grammar, both of which can be tailored for the domain of the dialogue. A typical representation of the syntactic structure of the utterance is a unification grammar such as in VerbMobil [36].
A semantic representation can be computed using compositional semantics, in which the meaning of a sentences is constructed as a function of the meaning of its constituent parts. The meaning can then be encoded in a framework such as discourse representation theory.

NLU in a dialogue context faces difficulties in addition to those in language processing in general [45]. Dialogues often contain utterances that are not grammatically well-formed in terms of a normal sentence grammar. For instance, dialogue participants often use incomplete sentences or sentence fragments, or use self-repair when they write. An NLU module must take this into account when analyzing the utterance.

A solution is to make parsing more robust by restricting the vocabulary that the system understands. When the dialogue task is simple enough, keyword spotting is sufficient to extract the required information from the utterance.

The two well-studied methods used in NLU are identified as pattern matching and statistical learning [19]. When pattern matching is used, the input sentences only needs to contain some keywords to be considered a match. The system is equipped with scripts that contain a sequence or a set of keywords. The sentences is searched for the keywords and if all keywords are represented and no other scripts matches the sentences better considered a match and a response can be generated automatically. The pattern matching method is easy to implement and intuitively understandable, however it can’t distinguish between similar sentences with different meanings.

The statistical NLU techniques generally rely on an analysis of word frequency and association across a wide corpus of appropriate texts. It also allows user inputs to be matched to expected inputs, and appropriate response generated. The statistical NLU techniques is very powerful, but the development and application is more complicated than typical pattern matching based approach.

Domain Knowledge

Dialogue manager usually needs to interface with some sort of external knowledge source such as database or an expert system [45]. This could be for example database for academic advising. The query or plans thus have to be converted from the internal representation used by the dialogue manager to the format used by the external domain specific system (e.g. SQL). This interfacing is handled by the domain specific modules. This can be handled by Natural Query Processing system. This system generate SQL query from natural language.

Enabling a dialogue system to communicate with outside knowledge sources allows it to abstract away from the task at hand and increases reusability and adaptability [38].

Natural Language Generation

The natural language generation (NLG) component of a dialogue system is responsible for generating the linguistic realization of the system's dialogue move. In general, it must decide what content to express (although this may come from the dialogue control module), how to structure this information, and finally how it is realized, i.e. its surface form.

The natural language generation (NLG) component host the process through which the output language is composed. There are two well-known basic methods to generate language: template-based and generative-based [15]. For the first method, the simplest and thus the most common, the dialogue designer preassembles sentences or prompts. The prompt usually contains parameters that the dialogue manager defines. The term generative-based refers to a process where an output sentence is constructed from words or even morphemes.

Language generation within a dialogue system must be able to relate utterances in order to generate correct and natural anaphors as a cohesive device. It also typically takes into account of a user model, and must be equipped to generate not only full sentences, but fragments, which often appear in dialogue.

Dialogue Manager

In order for all of these parts to function together properly, they need to be pulled by together and organized by a central controlling module. The exact function varies depending on the needs of the particular system.

The role of facilitating communication between modules is typically taken over by dialogue manager. In addition, the dialogue manager can maintain a representation of the dialogue context in order to motivate further action. The control of the dialogue flow should be based on a theory of dialogue. This means that the dialogue manager can have a domain dependent plan which guides its action. It can also have knowledge of how to utilize modules in order to achieve its dialogue goal. This means interleaving the computations performed by the natural language understanding and generation with the access to other knowledge sources. The dialogue goal can be, for instance, a task, a tutoring goal, or advising delivery.

2.6 Approaches to Dialogue Management

The job of dialogue manager is to control the flow of the interaction between the system and a dialogue participant based on some theory of dialogue. In this section we consider four types of design which support this role; finite state automata, the form-filling approach, the information state update approach and plan-based approach [15, 38].

2.6.1 Finite State Automata (FSA)

Systems which use the finite state automaton approach to dialogue management are characterized by a finite state machine which statically encodes all possible dialogues. The node represents dialogue states or dialogue context, and an edge represents an action such as a dialogue move. An FSA-based dialogue manager can retain very tight control over the dialogue process. This type of dialogue manger also has simple design and can be made deterministic.

However, such systems are flexible with respect to dialogue flow, and are not suited to support user initiative.

FSA-based dialogue managers are suited to simple tasks such as information elicitation, where a certain set of data must be collected by an agent in order to carry out some action. Contexts where the number of possible dialogues is small can be represented. Examples of finite state systems are the CSLU toolkit [48, 49], the DiaMant tool [50], or the Nuance demo banking system [51], which uses recursive transition networks.

2.6.2 Form-filling Approach

The form-filling approach (also known as the frame- or template-based approach) is more adaptable than finite-state [38]. The system tries to incrementally fill slots in a form by asking the user questions. When enough information is present in the form, the system can perform its task, such as a database lookup. A form-filling system works like a production system, where actions are determined by the current state of affairs.

The gain in flexibility over FSA systems is that the order in which slots in the form are filled is not strict, which allows some variation in the order in which information is elicited from the user. Over answering can be dealt with, since more than one slot may be filled by a single user utterance.

For example,

System: What is your destination?

User: London on Friday around 10 in the morning.

In this example, we see that the user gives more information than was requested.

Although the system has only asked the user for his/her destination in timetable application, the system can recognize that three pieces of information (namely destination, day and time) have been supplied, and can insert these in the suitable slots of the form. A form-filling system is better equipped to handle user-initiative, and this ability can be strengthened by modelling the task as sets of forms, or contexts [33].

Form-filling is suited to situations in which the information flow is mainly in the direction of the system, for instance, in timetable information systems, such as the Philips automatic train timetable

information system [35]. VoiceXML [52] is an extension of the form-filling approach using an XML-based specification language to model dialogues. It supports both information-seeking and menu type dialogues.

2.6.3 Information State Update Approach

The basic frame-based dialogue systems we have introduced so far are only capable of limited domain-specific conversations [15]. This is because the semantic interpretation and generation processes in frame-based dialogue systems are based only on what is needed to fill slots. In order to be usable for more than just form-filling applications, a conversational agent needs to be able to do things like decide when the user has asked a question, make a proposal, or reject a suggestion, and needs to be able to ground a user's utterance, ask clarification questions, and suggest plans. This approach is generally called information-state [53]. The information state update approach provides a method for specifying a theory of dialogue, and this theory has the following five components:

- An information state
- Representations for the information state
- A set of dialogue moves
- A set of update rules
- An update strategy

We will now examine each of these in turn.

An information state: The information state is the description of the state of the discourse and its participant which is maintained by the dialogue manager. It stores dialogue level knowledge, such as the common context, linguistic and internal structure, or aspects of beliefs or obligations, depending on what theory of dialogue it formalizes. This context then forms the basis for the choice of action of the dialogue manager.

Representations for the information state: for each aspect of dialogue context that should be modelled, a representation must be chosen.

A set of dialogue moves: provide an abstract way from utterances and other dialogue actions to a description of their function. When a dialogue move is performed its content may result in a change being made to the state of the dialogue. Which dialogue moves a dialogue theory includes is influenced by the theory itself and the domain of the dialogue.

A set of update rules: as dialogue progresses, the information state which describes it must be updated to reflect the effect that actions of the dialogue participant have on the dialogue context. How these updates take place is governed by update rules. Update rule fire in reaction to observed dialogue moves, and are specified by applicability conditions and effects. If the conditions are satisfied by the information present in the information state, then the effects of the rule can be carried out. The effects are changes that will be made to the information state. Thus, update rules can be seen as transition between information states.

An update strategy: in order to control how updates are made to the information state, an update strategy must be declared. This is an algorithm which decides which update rules should be fired. Option for this algorithm includes allowing the first applicable rule to fire, allowing all applicable rules to fire, or choosing between rules based on probabilistic information.

2.6.4 Plan Based Approach

One of the earliest models of conversational agent behavior, and also one of the most sophisticated, is based on the use of AI planning techniques [15]. For example, the Rochester TRIPS agent [54] simulates helping with emergency management, planning where and how to supply ambulances or personnel in a simulated emergency situation. The same planning algorithms that reason how to get an ambulance from point A to point B can be applied to conversation as well. Since communication and conversation are just special cases of rational action in the world, these actions can be planned like any other. So, an agent seeking to find out some information can come up with the plan of asking the interlocutor for the information. An agent hearing an utterance can interpret a speech act by running the planner 'in reverse', using inference rules to infer what plan the interlocutor might have had to cause them to say what they said.

Using plans to generate and interpret sentences in this way require that the planner has good models of its beliefs, desires, and intentions (BDI), as well as those of the interlocutor. Plan-based models of dialogue are thus often referred to as BDI models. BDI models of dialogue were first introduced by Cohen, Perrault [55, 56] and their colleagues and students in a number of influential papers showing how speech acts could be generated, and interpreted. At the same time, Wilensky [57] introduced plan-based models of understanding as part of the task of interpreting stories.

Generally, the above dialogue management approaches are classified according to their increasing order of complexity. The last two approaches provide methods of modelling both mixed-initiative and complex dialogues. They required heavily coded solutions and are not readily suited for robust applications in real-world setting.

2.7 Spelling Error and Correction

Spelling errors are mostly common in text-based dialogue systems. In this section, we will see about the spelling error and correction. The section briefly introduces the types of spelling errors as well as detection and correction of spelling errors, mainly in typed text.

2.7.1 Types of Spelling Errors

Nowadays, there are two distinct types of spelling errors, namely, non-word error and real-word error. Damerau [58] indicates that 80 percent of misspellings are categorized by four rules, they are transposition of two letters, one letter extra, one letter missing and on letter wrong (Transposition, insertion, deletion and reverse).

Non-word Error

The definition of non-word is that a word does not have a meaning. According to Kukich [59], normally, typed text has three types of non-word errors, namely, (1) typographic errors, (2) cognitive errors and (3) phonetic errors. In the case of typographic errors, it is assumed that the typist knows the correct spelling but presses the wrong key by accident, and or presses the keys in wrong order, e.g., that → taht. In the case of cognitive errors, it is assumed that the typist does not have the knowledge or misunderstands the intended word, e.g., minute → minite.

In the case of phonetic errors it is assumed that the typist substitutes a phonetically correct but orthographically incorrect sequence of letters for the intended word. e.g. two → too.

Real-Word Error

According to Mitton [60], "real-word error is a valid word but not the intended word in the sentence, thus making the sentence syntactically or semantically ill formed or incorrectly.", e.g., there → their.

2.7.2 Error Detection and Correction

N-gram Analysis and Techniques

N-gram analysis is one of the popular methods for detecting non-word errors. Normally, it is used to detect errors made by Optical Character Recognizers (OCR) [59]. N-grams are n letters subsequences of words or strings.

One-letter n-grams are referred to as unigrams or monograms; two-letter n-grams are referred to as bigrams; and three-letter n-grams are seen as trigrams.

In order to pre-compile an n-gram table, a dictionary or corpus of text is usually required. The table stores n-gram's existence or frequency, any n-grams in an input string that have nonexistence or low frequencies are classified as probable misspellings. The table has a variety of forms, such as binary bigram array or binary trigram array. In the case of binary bigram array, it has two-dimensional array of size 26 * 26 whose elements represent all possible bigrams. For each element, if it occurs in at least one word (string) in predefined dictionary or text, then its value is set to 1, otherwise, set to 0. A binary trigram array could have three-dimensional array. Since these binary n-grams do not indicate the position of each n-gram within each word, they are non-positional binary n-gram arrays. Moreover, it is said that a set of positional binary n-gram arrays are able to detect error more accurately. Because each element in the positional binary n-gram arrays matches the exact position within each word. However, this raises the storage space problem due to the large capacity of the positional arrays. Since most misspellings do not contain any impossible n-grams, so n-gram analysis techniques are not good at detecting human generated errors but good at detecting machine-generated errors.

Dictionary Lookup Techniques

Dictionary lookup is another common technique for detecting non-word error. Because, it is a simple and quick task, it just simply lookup each word in a dictionary; any unfound words are considered as misspellings. However, response time and storage space raise issues when the size of the dictionary increases. In order to improve the access speed to a dictionary, several standard search techniques have been introduced, such as hash table [61] and finite-state automata [62].

Hash table is the most commonly used approach. It computes each input string's hash address and stores it at the address. To looks up the input string, if it is different from the retrieved word stored at the table or is null, then it is misspelling. Turba [63] outlined the main advantage and disadvantage of using hash tables; the nature of fast random access of hash code avoids the large number of comparisons needed for tree-based searches or sequential of the dictionary. However, an efficient hash function has to be devised to avoid collisions occurred in construction of the hash tables.

There is an alternative approach suggested by Peterson [64], a dictionary can be partitioned into three separate dictionaries for spelling correction. The first one has a few hundred most commonly used English words that are stored in the cache memory; it accounts for 50% of the complete dictionary access. The second one consists of a few thousand document specific words that are stored in the regular memory; it accounts for 45% of the access. The third one accounts for the remaining 5% of the access, has complete dictionary (probably stored on disk) that is stored in the second memory.

NLP Prototypes for Handling Ill-formed Input

Nowadays, context-dependent word detection and correction tools have natural language processing (NLP) capabilities, including robust natural language parsing, semantic understanding, pragmatic modeling, and discourse structure modeling. Most prototype NLP systems are parser driven and incorporate error-handling techniques. According to Kukich [59], a typical NLP system consists of a lexicon, a grammar, and a parsing procedure. The lexicon is the set of terms that are relevant to the application domain.

The terms are usually annotated with their potential parts of speech as well as morphological information. The grammar is a set of rules that specifies how words, parts of speech, and higher-order syntactic structures may be validly organized into well-formed sentence fragments and sentences. The parsing procedure often consists of looking up each word in the dictionary to determine its potential parts of speech and applying grammar rules to build up higher-order syntactic structures.

Chapter 3: Related Work

3.1 Introduction

In this Chapter, we will present some works that are done so far in the area related to academic advising dialogue system. The chapter briefly discusses about the approaches they followed, the implementation methodology and the evaluation result obtained. The related research works to be discussed are present advising systems with an expert-based system combined with natural language processing techniques for communication.

3.2 Expert-based Academic Advising Dialogue Systems

Leung *et al.* [65] proposed an intelligent counseling system that uses an ontology-based information retrieval engine to suggest relevant information to students. The goal of the research work was to develop a next-generation, student-centered learning environment that blends the strengths of distance learning, face-to-face teaching, and online learning. The system can provide 24 × 7 online counseling services to prospective and current students by simulating the role of a teacher/advisor when responding to enquiries in areas such as career guidance and development, study paths and methods, and program/course choices. The intelligent counseling system has two modules: academic counseling and academic advisement. The academic counseling module handles general queries from prospective students, on questions about career development, program/course information, and learning modes. The academic advisement module deals with specific questions mostly from current students, on program specifics, study plans, and graduation checks. The evaluation of intelligent counseling system yielded satisfactory performance in terms of credibility of its recommendations and usability.

McMahan and Bates [12] developed an automatic dialogue system for student advising using a rule-based approach. The implementation methodology of the system was concentrated on the expert system and natural language interface.

There is also a passive knowledge bank in which the natural language interface stores information for the expert system to use to make decisions, AIML (Artificial Intelligence Markup Language). AIML stores information in an XML-style document as input patterns, a topic category, and the appropriate response.

The expert system was approached as a rule-based decision engine, i.e., static and fast rules that decide what courses to suggest based on knowledge bank information. Most of the rules are inherited from the academic bulletin students use, but they choose to include more rules based possible information that can be derived from the conversation with the student.

The natural language interface has several major steps involved in it: dialogue management, a user interface, and enhancing the dialogue content. The state manager controls the flow of the conversation to ensure that enough information is obtained for the expert system to make decisions about course recommendations. The user interface was designed as a web application to interface naturally with a user. This user interface also has the capability to communicate to a back-end server. Once the data is input to the back end, it flows to the state manager that interacts with the AIML.

It was tested by several students and it was found to be simple and easy to use. The system was implemented using AIML, which is simple to implement, describe a category (we call a rule) as a pattern of words and wildcards and what to do when the rule matches the current input. However, AIML has weakness. It is inflexible. The possible input sentences are explicitly coded, it takes an exact match to trigger it. If a user makes a spelling error or uses a different syntax than what is expected, then it would fail. For example, if a user says “yeah” instead of “yes” and there are no measures in place to allow for this, then the system won’t recognize “yeah” as form of “yes”. In addition, it was limited to only yes/no type questions and a few phrases to manage state transitions.

Latorre-Navarro and Harris [19] proposed an intelligent natural language conversational system for academic advising. The main objective of the research work was to deploy an expert-based natural language academic advising system, to manage multiple straightforward advising tasks and allow advisors to engage in non-mundane educative tasks. To allow advising sessions to be more

centered on the development of the student, they developed Albert as academic advising dialogue system using an approach of pattern-matching techniques.

The design of Albert is composed of four main components. These are the web interface, dialogue manager, academic planner, and database. The advising dialogue manager in Albert includes the natural language understanding and generation system, and the task manager. The dialogue manager (DM) was built around ChatScript (CS), an open source scripting language.

The natural language generation component responds an answer that is taken from the knowledge base. The knowledge base has collection of information the system should return in a conversation. The NLU component of the DM produces a semantic representation of the natural language input that is appropriate for the dialogue task. The task manager component represents all the functions Albert executes to complement the dialogue task. These functions include user account management, database management, input validation, automatic updates, a scaling-up routine, and statistical data collection.

The academic planner in Albert was designed to offer students guidance and recommendations when preparing their course plans for each term. Students can enter their academic record in the academic planning process and receive recommendations on how to develop their academic plan up to graduation, based on courses completed, course prerequisites and all academic rules.

Students used Albert during three experimental phases. For the third phase, the system performed well, obtaining close to 80%. Albert allows unrestricted natural language communication utilizing state of the art natural language processing techniques.

3.3 Web-Based Higher Education Advising Systems

Albalooshi and Shatnawi [68] proposed a multidisciplinary web-based higher education advisory system that offer standard, advanced, and configuration management services that can easily be accessed by students, advisors, educational managers, and alumni.

The standard services was based on retrieving and organizing students' information in direct queries that require retrieval, formatting, and organizing of students' data.

Basically, the authors have five services in this category. These are student transcript and registration information, courses to be registered next semester, student's graduation progress, student progress dependency graph, and alumni services.

Advanced services in this category the system was not only does retrieving students' information but also it allows students to interact with the information such as GPA simulation. Educational managers was able to retrieve useful information such as students' statistical information and statistical information for managers.

Configuration management services these services was enable the system manager to configure the business rules and system parameters, or, on the other hand, to update, delete, or insert new records to the database, i.e. if a new degree program was introduced.

A multidisciplinary web-based higher education advisory system was evaluated by several users (students and advisors). The evaluation result was satisfactorily positive. The system provides the user with valuable, accurate information; can solve many advising issues. However, a Multidisciplinary Web-Based Higher Education Advisory System was only designed to perform a simple tasks. In addition to that the system was not ease of use and user friendliness.

Feghali *et al.* [16] proposed online advisor system that provides students with a very basic interface that contains no intelligence: it is an online transcript, showing the semester in which the student is registered with a listing of the courses took with their respective grades.

The Online Advisor creates academic schedules semester-to-semester and year-to-year by organizing information from many sources. The system provides information needed for course planning in an understandable and visually appealing way. In particular, the system displays the major and overall average, indicates which major, university and distribution requirements have been satisfied and which need to be completed. It displays the number of credits completed and the number still needed for graduation.

A database management system through a web application acts as an interface between applications and the Online Advisor. The online advisor is not 'intelligent' on its own. It only uses the rules that are stored in the database and displays them with the user in mind.

The online advisor system consists of three pages: the summary, the degree checklist and the degree plan pages. The summary is the one page that is mostly used by advisors and students that gives a quick snapshot of the most current situation of a student's file. The Degree Checklist displays the courses required from each student in order to fulfill his degree requirements.

The "degree plan" pages allows the student or the advisor to select the courses that he/she thinks must be taken in a specific semester.

The usability of the online advisor was assessed by several users. 79% of users stated that they were satisfied with the Online Advisor. 90% rated the Online Advisor as effective and efficient. 75% rated the Online Advisor as useful and helpful. However, the online advisor system was not intelligent.

3.4 Summary

In this Chapter, we reviewed the works that have been done related to this research work. We have discussed the approach they used, the implementation methodology, and finally, we discussed the evaluation result of the systems.

From the reviewed related works, we have understood that all the academic advising dialogue systems are based on the business process rules of the institutions. That is, each university has its own different business process rules. The business process rules of the Ethiopian public universities also differ from others. On the other hand, the Ethiopian public universities share similar business process rules such as the harmonized academic policy and the harmonized curriculum. For example, the requirements for academic dismissal, warning, semester pass, graduation, readmission, grading systems, etc., are similar at Ethiopian public universities.

Advising of students in those business process rules requires a lot of time and intellectual investments from the human agent burdened with such a responsibility.

As far as we know, there is no open source higher education students advising system, which helps us to easily customize our system. Thus, we need to design a dialogue system for advising Ethiopian public universities students.

Chapter 4: The Proposed Solution

4.1 Introduction

In this Chapter, we will discuss the overall design of our proposed system, that is, Dialogue System for Advising Ethiopian Public Universities Students. First, we will discuss the approach we followed in this study. Then, we will see the general overview of the proposed system architecture. Finally, we will present the detailed explanation of the modules/components and the subcomponents. After this, Dialogue System for Advising Ethiopian Public Universities Students, for short called as HESAS (Higher Education Students Advising System).

4.2 Architecture of the Proposed System

In this study, we followed the prescriptive model of academic advising for advising Ethiopian Public Universities Students. We have chosen this model, because of, most of Ethiopian Public Universities has experiencing this advising model. The design of HESAS used template-based approach for natural language understanding (NLU) and response generation (RG) and the knowledge base dialogue management approach that uses the information state update architecture. We have chosen template-based approach for NLU and RG, due to, the simplest and the most common, we only need to preassemble a sentences or prompts. The Information state update approach is best suited for this study. It contains many features for designing a dialogue agent such as asking clarification questions, making a proposal, rejecting misunderstood questions and suggest a plans, that is, why we have chosen this dialogue management approach for the purpose of this study.

Generally, most of the design of dialogue systems have similar components or modules. These components can be summarized as language understanding, dialogue manager, and response generation. Even though, the general basic architectures of dialogue systems are nearly similar, they substantially differ in their internal functionalities depending on the algorithm employed and the aims of the dialogue systems were designed. As many of the dialogue systems design, HESAS also share common components such as dialogue manager, response generator and knowledge base. As a result, it consists of spelling corrector, dictionary, natural language understanding

subcomponents and advisement service. The natural language understanding component performs three types of analysis of user's statement: syntactic, discourse, and semantic analysis as shown in Figure 4.1.

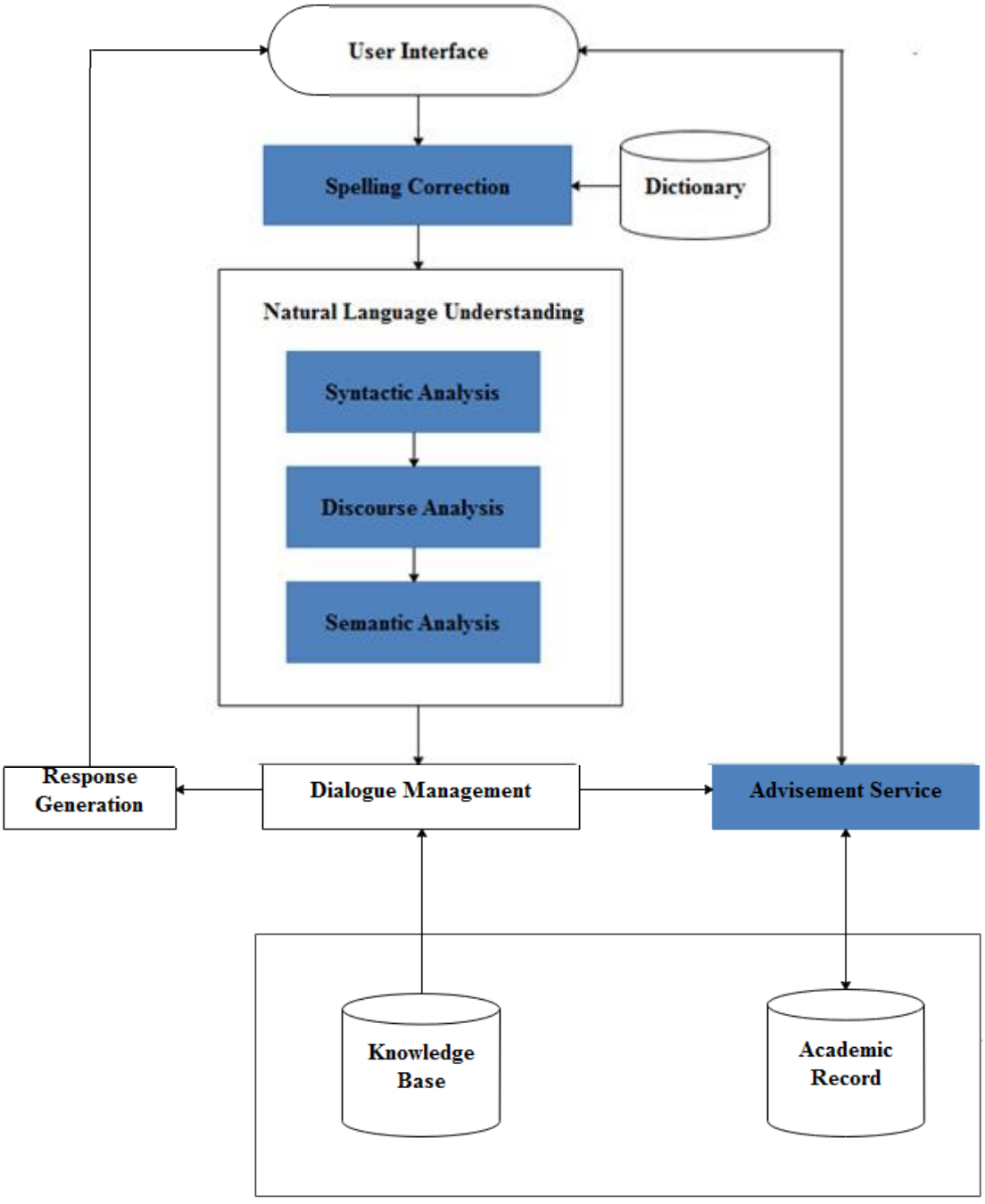


Figure 4.1: The Architecture of the Proposed System

4.3 Spelling Correction

The user interface is responsible for accepting the user's input and provide an output. It converts an internal representation (which user may not understand) to the user understandable form. The job of spelling corrector component is concerned with detecting misspelled words in the user's input and possibly assisting users in correcting them with the use of dictionary.

HESAS is a text-based system, thus spelling errors are very common, yet the users expect the system to handle them effectively. This work manages the typographical errors, which is described previously. This spelling correction routine uses a lexical distance measure using methods (inserting a character into a string, deleting a character from a string and replacing a character of a string by another character) to determine the likelihood of the possible corrections. If a correction candidate has a significant likelihood versus the other candidates, this value is replaced in the original input statement and this new statement is sent to CS (ChatScript). If there are candidates, but none of them is relatively significant, the system returns all the candidates to the user and prompts the user to choose a selection or rephrase the statement. If no candidates are found, the system terminates execution and responds to the user that statement was not recognized.

The spelling corrector uses a dictionary lookup techniques for detecting non-word errors. Because it is simple and quick task, it just simply lookup each word in a dictionary; any unfound words are considered as misspellings. So that, the dictionary contains list of English words. We used an existing dictionary and on top of that, we also added list of technical terms. This technical terms were continuously updated throughout the experimental phases. Totally, the dictionary consists of 15,827 words. Appendix A shows sample international English words. Appendix B shows list of technical terms that were added to international English words to enhance the spelling checking methodology for this study. Algorithm 4.1 shows how the spelling corrector detects the spelling error and fix it and Algorithm 4.2 shows tokenization.

Algorithm 4.1: Spelling Corrector

Input: User's statement with incorrectly spelled terms.

Desired Output: Suggesting the possible words for incorrectly spelled terms.

Initialize.

TokenizedWordList. Construct a list of all distinct tokens in the user input (see the Algorithm 4.2 for tokenization).

For each word W in TokenizedWordList, find W in the dictionary D

  If W not in D **then**, find all terms T similar to W, within

  a lexical distance of 2. Assign score K to each T.

    Then If a value TK is statistically significant versus all other TK

      Then replace TK value in place of W.

    Else if any value TK exists

      Then return all TK value to the user and prompts the user to choose a selection.

    Else

      End execution.

  Else

    W is the correct word.

End for

Algorithm 4.2: Tokenization

```
Input: User's statement with incorrectly spelled terms.
Desired output: List of tokenized terms.
Initialize. Get the input from the user and set all variables to
their initial state.
Do
   Read the user's statement character by character.
   If the token contains punctuation marks and special characters
     Then replace them with space.
   If statement contains space
     Then get the token before the space and store it into an array
     of tokens.
Until end of the statement
END
```

4.4 Natural Language Understanding

The job of natural language understanding is concerned with the analysis of the utterances a user makes in a dialogue. In order to perform analysis on the user's utterances, the NLU component should consist of three subcomponents. These are syntactic, discourse, and semantic analysis.

4.4.1 Syntactic Analysis

The syntactic analysis consists of determining the grammatical relationships among the words in the user's utterance. It is the first stage in the analysis of a user's statement, the syntactic analysis, starts with tokenization of the user's statement, that is, division of the input in a series of distinct lexical entities. There are two distinct steps in the implementations of syntactic analysis: part-of-speech tagging and parsing.

POS tagging consists of assigning to each tokens a part of speech indicating its grammatical functions, such as singular noun or comparative adjective. Parsing is the process of assigning a "phrase marker" to a sentence, that is, the process that given a sentence, produces a structure.

Generally, the syntactic analysis; firstly, performs tokenization. Then, it assigns to each tokens a part of speech tag. Finally, it performs chunking operations. The Algorithm 4.3 illustrates POS (part of speech) tagging and the Algorithm 4.4 shows how the chunking operation work.

Algorithm 4.3: A Part of Speech Tagging

```
Expected Input: Correctly spelled user's statement.
Desired Output: List of tagged tokens.
Initialize. Read the user's statement and assign all
variables to their initial states.
Perform Tokenization. See the Algorithm 4.2
Manual Tagging.
For each Word W in Tokens
   If W exist in a Manually Tagged Corpora MTC
           Then If multiple tag
              Then add it in a buffer.
           Else put the appropriate tag from MTC
End for
Read from buffer
If the tagging structures in the database (corpus) equals
current word's sentences structure
     Then Put appropriate tag.
Else
     Abandon for manual tagging.
Get the tagged tokens and store it into an array.
END
```

Algorithm 4.4: Chunking

```
Input: List of POS tagged user's statement.
Desired Output: phrase chunk with chunk type.
Initialize.
Begin
   Read POS(i) from list of POS tagged.
   Do
      While POS not equal to Chunk marker
        Chunk ← Chunk + Word(i).
        i ← i + 1.
        Read POS(i).
      End While
      Chunk type is defined by according to chunk marker.
      Add chunk and chunk type to an array.
      Chunk← NULL.
      i← i + 1.
      Read POS(i).
   While POS is empty
END
```

4.4.2 Discourse Analysis

Discourse analysis consists of determining the relationships among multiple sentences. An important component of discourse analysis is *reference resolution*, the task of determining the entity denoted by a referring expression.

4.4.3 Semantic Analysis

The natural language understanding finally performs semantic analysis, which is determining the meaning of the sentences. Typically, this consists of representing the statement in canonical formalism that maps statements with similar meaning to a single representation and that facilitates the inferences that can be drawn from the representation. WordNet, a lexical database has been used to provide lexical semantics for the words occurring in parsed sentences [54]. This system employs ChatScript (CS) which includes WordNet along with many predefined concepts, which allows easily to handle synonymous expressions. Conversely, the definitions for the technical terminologies the users must be used hand-coded (a written program).

The design of NLU contains several input templates and measures for responding to secondary topics such as information about the HESAS, control of the system and conversational systems. The structures in the NLU for these secondary topics use the same principles as the main topic, and the design of calls for off-topic dialogue to promptly converge to on-topic dialogue.

The input template design in NLU involved identifying the keywords, POS tags, noun-phrase chunks and lexical relations of each input statement and selecting the most significant features, to define the keywords of the template and any respective topic keywords. For example, the topic *course_information* includes as keywords, the list of all course names, course code, credit hours, ECTS, and term-phrase. The keywords for each input template, the words that convey the message, are extracted using CS. For example, in the input "*what is the course code of object oriented programming?"* the keywords are *what, course-code*, and *object-oriented-programming*. Using CS concepts and facts, the keywords are generalized to "*what"*, *course-code*, and *course-name*.

Clearly, this example is not the only way to ask the course code of the specified course. For example, the user could ask "*please tell me the course number of object oriented programming?*", or if the request is within the context of the previous input, the user could ask "*what is its course number?*".

For reference resolution tasks, the system uses the CS feature of rejoinders. Rejoinders are input templates, which follow parent templates that elicit some expected user response. Rejoinders also allow some input templates at the end of topics to assume certain keywords were implied. The entity recognition problem is managed by defining case-insensitive CS concepts with pre-classified POS tags. In addition, the system has concepts defined for all the technical terms that are not available in the CS or WordNet dictionaries.

There are two topics that the NLU script contains to respond when an input statement does not match with any template. These are antepenultimate topic (topic immediately preceding that one of a series which is next to the last one) and penultimate topic (the last topic in a template design).

The antepenultimate topic contains several on-topic templates with either, partial answers because the full statement was not recognized or with suggestions to help the user obtain the anticipated response.

This process is alike typical communications where if the receiver is doubtful of the input, it notifies the sender of the issue or requests a confirmation of the received message. This topic also contains several templates for pronoun resolution, where the system will either redirect the user to a specific topic for the response or alert the user that the statement was not recognized as written, to encourage the user to not repeat such statements. In general, this topic is designed to respond to any input statement that contains a significant keyword, such as a topic keyword, and return the most likely match based on a score of the significant keywords.

The penultimate topic contains a small collection of less significant keywords, such as verbs and common nouns, and some off-topic templates. Most of the templates in this topic are produced after analyzing the user log files and finding various users who did not understand how to use the system.

For example, many users would expect the system to have knowledge about popular culture or to successfully respond to one-word statements such as *course* and *enroll*. Ideally, this topic should not respond to users, however, it proved useful in the preliminary tests.

While these features enhance the template creating process, all template-based systems are limited by the amount of patterns for input matching.

4.5 Dialogue Management

The dialogue manager is the heart of HESAS because it coordinates the activity of all components, controlling the dialogue flow, and communicate with the knowledge base.

The dialogue manager controls the communication between modules/components. Modules/components are not able to pass messages directly to each other, for design as well as technical reasons. The design of the system is such that the dialogue manager is the mediator of all communication between system modules, and in this way is able to control all message passing and thus the order of module execution. In general, the dialogue manager should play many roles including extracting information from the knowledge base based on current input and discourse context, asking further information to submit an appropriate query, requesting to confirm unclear information and controlling generic conversation mechanism (e.g., multi-party dialogues, support a thousand of simultaneous users).

Algorithm 4.3 shows how the HESAS dialogue manager works. The algorithm keeps a queue of conversation acts it needs to generate. Acts are added to the queue based on grounding and obligations of the system.

Algorithm 4.5: A Dialogue Management

```
Input: Semantic representations of the user input.
Desired Output: Response to RG.
     While conversation is not finished
          If user has completed a turn
             Then interpret user's utterance
          If DM has obligations
             Then address obligations
          Else if DM has turn
             Then if DM has intended conversation acts
                  Then call generator to produce response to
          the user
             Else if some material is ungrounded
                  Then address grounding situation
             Else release turn or attempt to end conversation
          Else if no one has turn
             Then take turn
          Else if long pause
             Then take turn
     End while
```

Let us examine Algorithm 4.3, in this algorithm, Grounding means confirmation of understanding by the counterpart's of utterance. Confirmation can be explicitly, asking the user a direct question to confirm their understanding.

Here is an example of explicit confirmations from HESAS system. The (bold face) confirmation question is yes-no questions, using single sentence.

Student: I want to drop a course?

HESAS: **So, do you want to drop a course?**

Student: yes

Confirmation can be implicitly, rather than asking direct question, confirming to the user by repeating back what the system understood the user have said.

Turn in a dialogue is where the writer A write something, then the writer B write, and so on. No parallel utterance is allowed, meaning writer A or writer B cannot write multiple times without the intervention of A or B. Rule for turn-taking in this algorithm:

- If during this turn the current writer has selected A as the next writer then A must write next.
- If the current writer does not select the next writer, any other writer may take the next turn.
- If no one else takes the next turn, the current writer may take the next turn.

Obligation is defined in terms of modal operator often called permissible. An action is *obligatory* if it is not permissible not to do it. An action is *forbidden* if it is not permissible. Obligations are used in HESAS system to enable the system to correctly produce the second-pair part of an adjacency pair. That is, when a user requests something of the system, the system either accepts the request and give information to it or reject the request. HESAS rejects an utterance that it likely misunderstood, by giving the user prompts like *I'm sorry, I didn't understand that.*

The DM requires to extract the required information from the knowledge base (KB), that is, database (DB). The knowledge base is a database that stores advising information. Generally, the KB has information related to curricula and extra-circular issues. It consists of information related to curriculum for department of computer science, legislations of Ethiopian public universities, higher education glossary, introduction, default answers (no response matched) and information about HESAS.

KB contains a finite amount of answers that the system should return in a conversation. Similarly, the system input domain comprises infinitely possible style in which users can request each answer. To create KB, the output information was obtained from Ethiopian public universities documents such as curriculum and legislations and we have also obtained from different sites. The input templates were derived from a standard English subset of the input domain. The implementation of the KB has done using CS. To create the KB we used feature of CS called Fact. Facts enables us to create a database. Figure 4.2 illustrate the structure of HESAS KB.

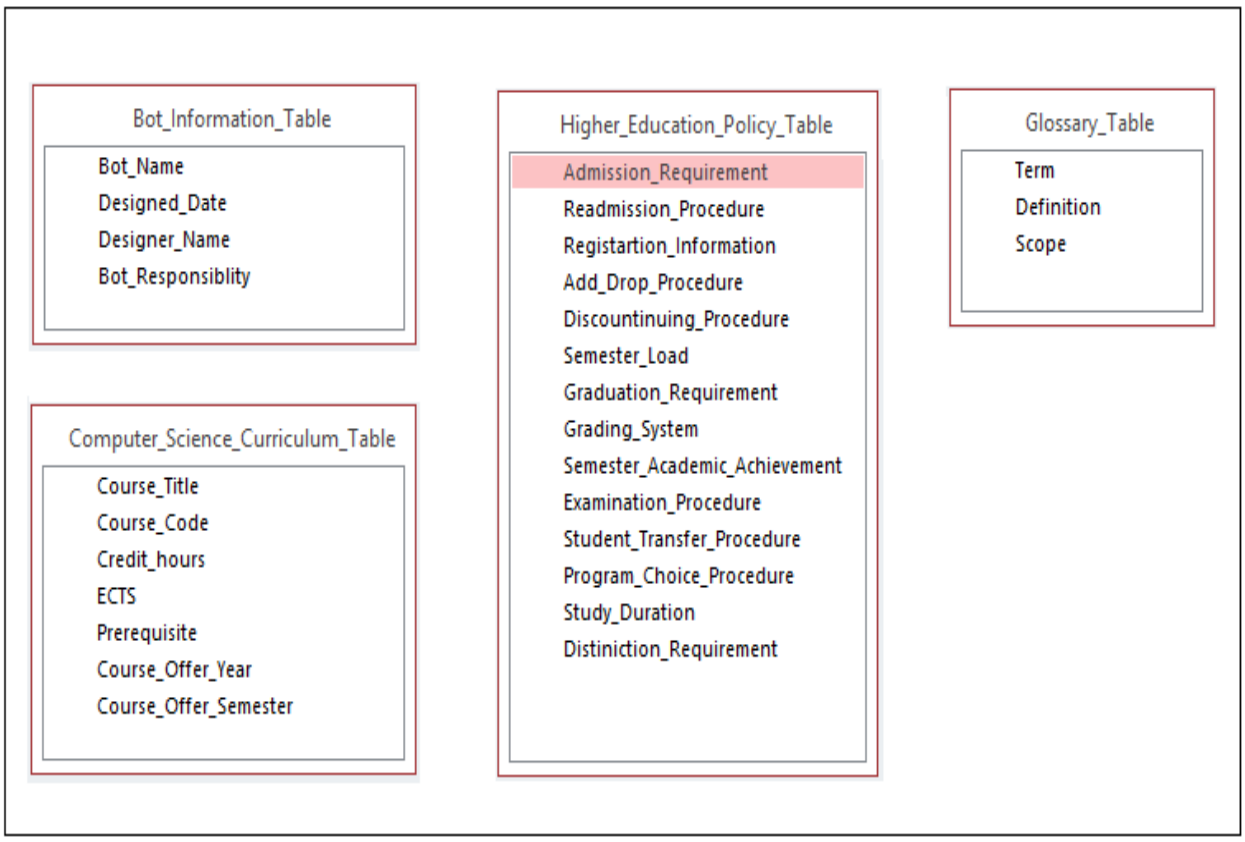


Figure 4.2: The Structure of Knowledge Base

4.6 Response Generation

The Response Generator (RG) is the final component of HESAS, which constructs the message or output to be given to the user. The output or response to be given by the user is given by the dialogue manager. The dialogue manager in turn extracts the response from the knowledge base of HESAS and sends it to RG. The job of RG is responding an answers to the user.

The RG consists of templates containing text, pointers, variables and other controls functions. As template-based system, the RG mostly responds with strings of answers taken straight from the KB. The answers are constructed using as tools logic functions, random generators, control structures and queries. The RG also contains a trivial amount of grammar rules, mostly for handling unrecognized input and extending off-topic (out of scope) dialogue. For example, when responding to unrecognized questions, the RG may either change the verb tense or rearrange the POS to let the user know that the answer to that question is not available.

4.7 Advisement Service

The Advisement Service component is concerned with registration, retrieving and organizing of the students' information. Basically, Advisement Service delivers two services. These are student transcript and registration information and course enrollment service. With this advisement service as the main feature, we referred to this system as the Advisement Service (AS) system.

4.7.1 Student Transcript and Registration Information

HESAS stores information about students' current semester registration and transcripts or the students' previous academic records in the academic record (AR) database. Students are able to obtain their current enrollment information and previous academic records. HESAS authenticates the authorization before accessing the AS system. A segment of template for viewing academic records is shown in Appendix C. The Algorithm 4.6 shows how the AS system works to view the students' previous academic records.

Algorithm 4.6: Viewing Academic Records and Current Enrollment Information

Input: Student information

Desired Output: Displaying student's previous academic records or current enrollment information.

Initialize. Get student's information and set all variables to their initial states.

[For showing previous academic record]

Search. Find the student's previous academic records from the AR database based on the information provided.

If the student existed **then**

Display his/her previous academic records.

Else

Inform the student that his/her record do not exist.

End

[For showing current enrollment information]

Search. Find the student's current enrollment information from the AR database based on the search criteria.

If the student is already registered for the current semester **then**

Show the student's current semester enrollment information.

Else

Inform the student that do not registered for the current semester.

End

4.7.2 Enrollment Service

The fundamental task of academic advisors is assisting students in academic matters. For a typical student, the academic advisor is someone who helps him/her choose which courses he/she should take and providing solutions related to academic issues. The students expect personal attention with precise answers for their unique questions. The theme throughout this work is that HESAS provides widespread solutions for frequently asked questions (FAQs).

HESAS includes the AS system which enables the Ethiopian Public Universities Students to register for the courses each academic semester. Courses that can be registered by the student in the next semester can be decided by the AS system based on the curriculum, courses completed, prerequisites approved and legislations of the Ethiopian Public Universities. The AS system contains all the rules and requisites for taking each of the courses of Computer Science department at Ethiopian Public Universities. The AS system allows the students to enter their information, based on that information, it analyzes the students previous academic records in order to allow course enrollment. Their course enrollment information should be stored in the AR database. The Algorithm 4.5 illustrate how the AS system work to register for the courses.

Algorithm 4.7: Registering Students for Courses

```
Input: Student's information and S a vector of courses to
register in the current semester.
Desired Output: Acknowledgement of successful course
registration.
  Initialize. Get the information and set all variables to
  their initial state including vector R.
  Search. Look the student from the AR database based on the
  search criteria.
     If student exist then
          [Considering First Year Student]
          If StudSemesterGPA ≥ 1.5 AND StudSemesterGPA < 1.75 OR
          StudSemesterGPA ≥ 1.75 then
              For each course Ci in S
               If prerequisite (Ci) is passed then
                  Add Ci to R
                  Increment i
               End for
               Add student information and R to AR database.
               Display acknowledgement.
          [Considering Second Year and Above Student]
          If StudSemesterGPA ≥ 1.75 AND StudCummulativeGPA ≥
          2.00 then
              For each course Ci in S
               If prerequisite (Ci) is passed then
                  Add Ci to R
                  Increment i
               End for
               Add student information and R to AR database.
               Display acknowledgement.
            Else
                Inform the student that registration was not
                succeeded.
        Else
               Inform the student not allowed to register.
END
```

The user interface and AS are built with PHP, a popular server-side scripting language that is well suited for web development. The user interface of AS runs outside the HESAS webpage as an independent webpage, allowing users to simultaneously use HESAS's messaging system and the AS system. With this feature, students can ask HESAS about academic rules, curriculum, details about the courses and related information as well as ask HESAS to initialize the AS webpages.

Students can initialize AS by requesting with expressions similar to *I want to register for the courses, I would like to see my previous grade, etc*. To prevent an unintended initialization, HESAS will ask users to confirm the request for AS before presenting the initial AS webpage. The AS system allows the students to view their previous academic records and to register for new courses. A sample student previous academic record is shown in Appendix D. Appendix E show a segment of the template registering for courses in AS. A sample code for courses enrollment is attached at Appendix F.

Academic Record (AR) is a database that stores information about the student's current semester registration and previous academic records. It can only contain information of the students of the department of Computer Science at Ethiopian Public Universities. The AS system can extract the students previous academic records from AR as well as stores the student's current semester registration information in AR database. Figure 4.3 show the database schema of AR.

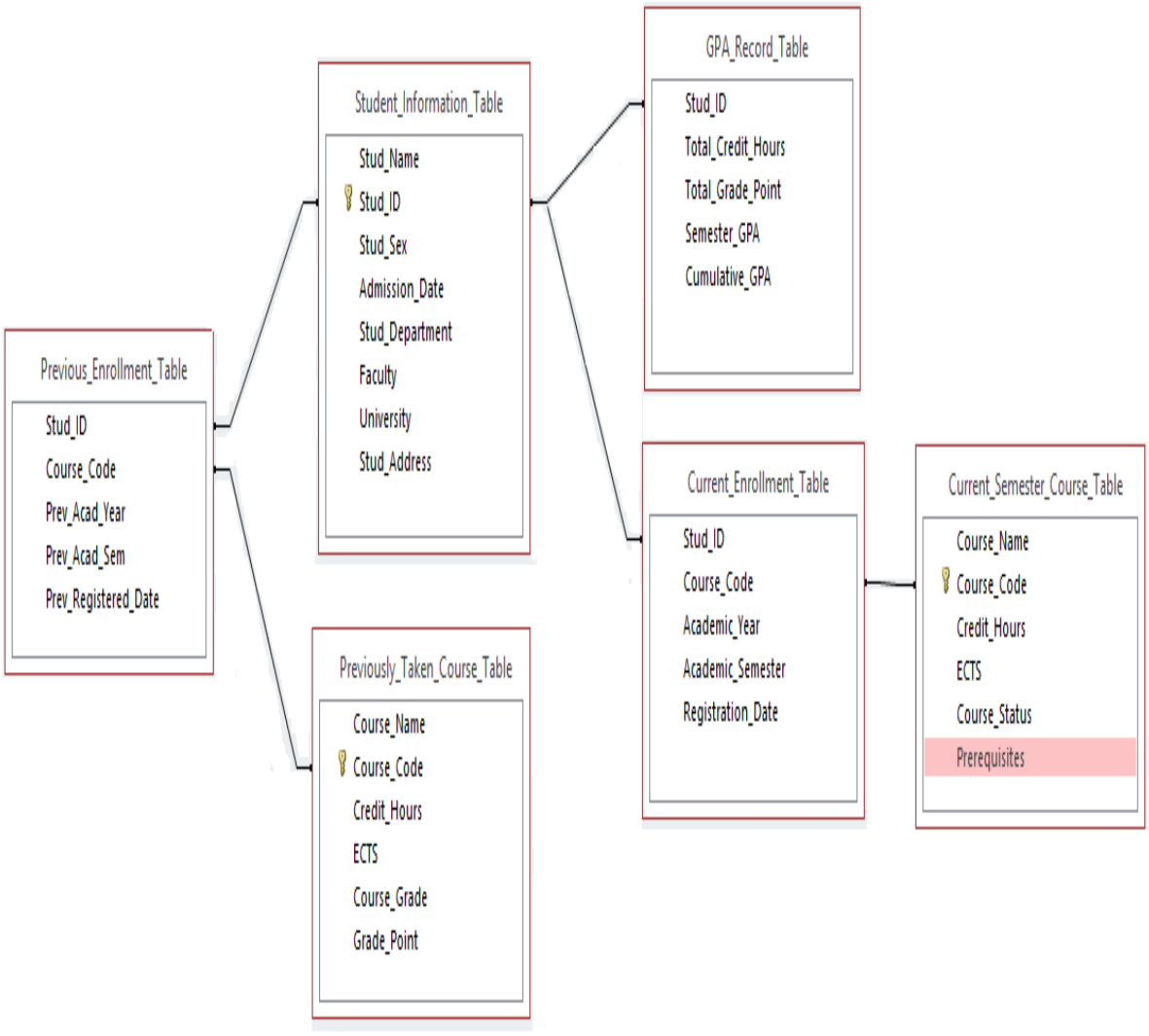


Figure 4.3: Academic Record Database Schema

Chapter 5: Experiment

5.1 Introduction

In this Chapter, we will present the implementation of the components and the experimentation aspects of our proposed system, HESAS. Firstly, we will give detail explanations regarding development environment used in developing HESAS. Secondly, we will discuss briefly the tools and programming languages that are used to develop our proposed system. Then, we will see the prototype of proposed system. Finally, in this Chapter, we will also discuss the evaluation metrics that are used to evaluate and experiment the performance of HESAS.

5.2 Data Source

Most of the time, natural language processing is a data-intensive field. The success or failure of most NLP application depends on the quality and availability of appropriate data. Most of the data we used for conducting the experiments are collected from various sources and in addition to that we have also prepared some of the data. For this experiment, data have been collected from four sources.

The first source is the Computer Science Program National Level Harmonized Modular Curriculum, which is prepared by Ethiopian public universities.

The second source of data is Senate Legislations, which is prepared by each of the Ethiopian Public Universities.

The third source of data is the students. We have also collected data from the students, while we are conducting the experimentation. The students has been participated in the experimentation in order to increase the performance of HESAS. Data collection can be done through the analysis of the user log files. Log file is a file that stores the user conversation history that made with the HESAS system. CS store the conversation log of the user.

The final source of the data is the web sites. We have collected the test data from different sites.

5.3 Implementation

Development Environment

HESAS is developed and tested in a laptop computer for the demonstration, testing and evaluation purpose. The computer has an Intel Pentium 4 processor, 4GB of RAM, 750GB of hard disk and is running Windows 8 Pro operating system.

Development Tools and Languages

For developing the prototype of our system, we have used programming languages such as PHP, HTML, JavaScript, CSS and ChatScript, and development tools such as XAMPP 7.0.13 all-in-one solution, consists of features such as Apache, MYSQL, and PHP. We have also used the latest released ChatScript version 6.9 as development tool.

Prototype

Figure 5.1 shows the user interface of the HESAS. In this user interface, the user or the student can ask HESAS, the natural language academic advising system, related to curricula, legislations and higher education term definitions.

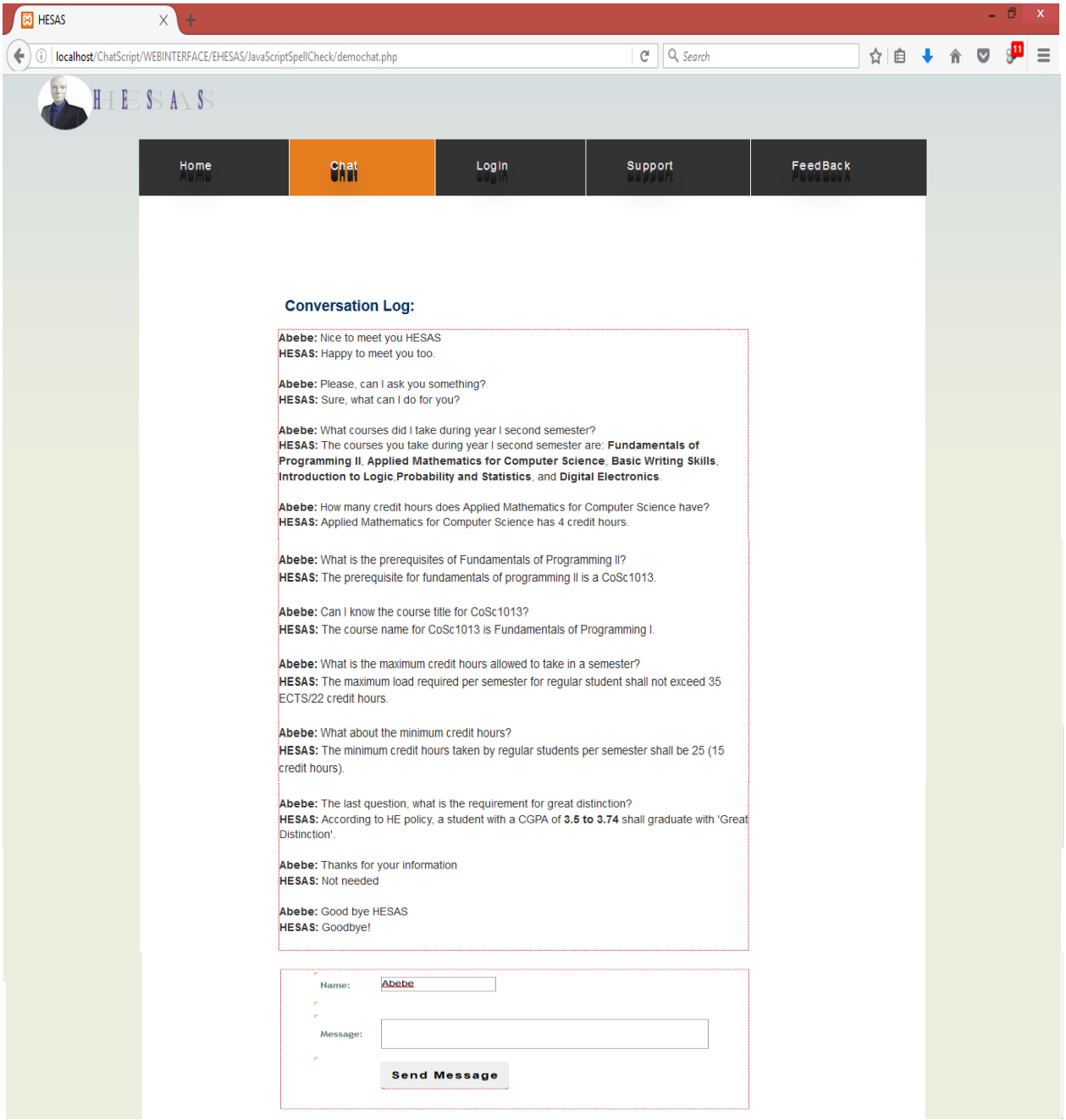


Figure 5.1: The User Interface of HESAS along with Conversation with the User

5.4 Evaluation Metrics

HESAS went through two incremental phases of testing. While testing during the first phase was not enough for significant analysis, it provide design decisions such as the need for advisement service system and a broad FAQ list on the system. We have described below what has been done during the two experimental phases.

Phase One

For the first experimental phase, we developed HESAS with some instructions and a handful of FAQ example. The development did not have AS system. The main NLU script has approximately 1000 input templates, for roughly 800 unique responses. During this phase, five students have participated to test the system. The experimentation was done by making the system available in a laptop computer. We connected the computer with another computer through the network cable and access the IP address of that computer were the system is placed.

The students experimented the system with random statements, with many using the example FAQs. Therefore, to help students about the information in HESAS, the FAQ was increased in the subsequent implementation. Students also showed for the chat system by asking from a wide range of topics. The FAQ list would also influence the students to focus on these topics. The FAQ lists are shown in Appendix G.

During this phase, the statements that were not recognized had two significant reasons. As expected, the first reason was that the information was not included in the design. The data collected provided information for the subsequent designs. The second reason was the students would approach HESAS as if it were a standard search engine, that is, they used mostly one-word entries. This result prompted adding to HESAS the ability to recognize keywords of interest to inform the user about the information available on the topic.

Appendix H shows the sample feedback given by students, who used the system during the experimentation.

Phase Two

For the second experimental phase, the system contained all the necessary features that we have described in Chapter 4 including the spelling corrector and the AS system. The main NLU script had approximately 3000 input templates, with over 1200 unique responses. Students did not participated in this experimentation phase, we have done it by ourselves.

The second phase introduced the measure to evaluate HESAS's input responses. To evaluate HESAS, we have used the evaluation metrics from information retrieval, POS tagging, parsing systems, named entity recognition systems and word sense disambiguation systems, namely, precision, recall and the F1 measures [15]. The following values can help us easily calculate the evaluation metrics for our systems. These values are True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN). The four values are categorized into two these are: Outcome Negative (ON) and Outcome Positive (OP)

- Outcome negative (ON) includes false negatives (FN), i.e., statements not recognized by the system and true negatives (TN), i.e., statements outside the design scope.
- Outcome positive (OP) includes correct responses or true positives (TP), and false positives (FP).

Precision is defined as

$$P = \frac{TP}{TP+FP}$$

to measure the percentage of outcomes that were correct.

Recall is defined as

$$R = \frac{TP}{TP+FN}$$

to measure the percentage of positive outcomes given the statements with expected outcome positive.

The unbiased F-measure, or F1 measure, is the harmonic mean of precision and recall.

$$F_1 = \frac{2PR}{P+R}$$

5.5 Test Result

To determine the number of TP, TN, FP and FN, it is necessary to manually evaluate the OP and ON statements. For this evaluation, a FP outcome is a response that is not relevant to the input statement. This definition of FP error conditions the system's capabilities of responding to the statement with respect to the available information. For example, if a user asks, "What is the max and min credit hours taken in a semester?", the system response "The minimum and maximum load required for a full-time student per semester shall be 25 ECTS (15 credit hours) and 35 ECTS/22 credit hours respectively" is not classified FP as the answer is relevant and it is the best response available in the system, if for the same input the system replied "The minimum and maximum total credit required to graduate for four year bachelor program shall be 240 ECTS/145 credit hours and 250 ECTS/151 credit hours respectively" the response is classified as FP, given it was not generated by the expected template by design.

A FN outcome occurs when the input statement has an available answer, yet the system did not respond correctly. Following the same example as above, if a user asks, "What is the semester load allowed to take?", the system response "I don't remember" is classified FN, as this question should have been responded as stated previously. TN outcome occurs when the input statement is outside the design scope. The FN outcomes include cases caused by technical difficulties in other areas of the system; however, with respect to the overall system performance they are still incorrect responses.

It is possible to estimate the FP and FN results by selecting a random sample of OP and ON statements. To select the samples, a number is assigned to each of the 1224 OP statements and to each of the 200 ON statements. Each statement is tested for an outcome FP, FN and TP respectively.

Table 5.1: System Performance Estimation Results

Total Statements	Outcomes		Values			Metrics		
	ON	OP	FN	TP	FP	Precision	Recall	F1 Measure
1424	200	1224	100	548	120	82.03	84.6	83.29

The tests returned 120 false positive statements, for 1224 of the OP samples and 100 false negatives for 200 of the ON samples. To reduce the ON responses, in the first phase during revision of the log files, false negatives were continuously identified and updated in the NLU system. As the data collection increased, the number of false negatives decreased.

The results for precision and recall are in line with the design goal of providing students with high precision answers and minimizing false positives. At the same time, the results show that over 75% of the queries to the system, obtained a reply that is effective versus an input-not-recognized type answer. This result is valuable.

5.6 Discussion

In this chapter, we presented the implementation and experimentation aspects associated with HESAS. In the implementation aspects, we have detailed discuss about the development environment, the tools and programming languages that we used to develop the HESAS system as well as we discussed how the HESAS components were implemented.

The experimentation of HESAS has been done in two phases. During phase 1, we analyzed the user log files and the data shows that, as usual in electronic text communications, students prefer to ask questions using the minimum amount of words possible, instead of through a Standard English sentence. Therefore, HESAS should include methods to allow this user preference, while concurrently allowing students to use complete expressions.

Phase 2 introduced the system performance measures as well as the evaluation results. We used evaluation metrics such as precision, recall and f1-measures to evaluate HESAS. We obtained the results of precision 82%, recall 84% and f1-measures 83%. The results show that the system performance metrics are all close to 83%. These results are estimated minimums; given the error is an estimated maximum. As compared to the recent studies conducted in e-learning systems HESAS offers a high-level performance for the complexity of the task [66]. Regarding the main NLP tasks in HESAS, specifically keyword extraction, question answering and natural language interfaces, recently published systems that require data corpora for training, obtain results that show the performance of HESAS is highly competitive [67]. In any case, the results show HESAS offers a competitive performance.

The enhancements developed throughout the experiments proved successful in increasing system performance and user interest for the system. The results of the second experimentation phase showed HESAS has developed into a practical and valuable application for academic advising.

Indeed, much effort is required in installing the service as a useful option versus visiting a human advisor and also the implementation of AS required much efforts to be a tool for an essential process that all Ethiopian Public Universities students must complete each academic semester and to confirm the viability of HESAS as a real world solution.

Chapter 6: Conclusion and Future Work

6.1 Conclusion

The fundamental objective of this research work is to provide students with an automated online advising experience that is as close as possible to traditional human interaction. In this research work, we proposed a model for dialogue system for advising Ethiopian Public Universities Students, based on which we designed a dialogue system which we called it as HESAS. HESAS is designed based on template based approach. HESAS composed of six main components. These are user interface, spelling corrector, natural Language understanding, dialogue manager, advisement service, and response generator. The implementation of HESAS is done with ChatScript, PHP, JavaScript and CSS languages.

The system offers Ethiopian Public Universities an instrument to potentially increase student integration, retention, satisfaction and performance, by providing additional advising services without the need for additional personnel. This achievement is possible by allowing academic advisors to allocate resources to tasks typically classified as developmental and educative advising, instead of repetitive time-intensive tasks related to prescriptive advising.

After two major experimental evaluation periods, the system has proved to be usable and a valuable application for academic advising, based on positive feedback from the users, the advising personnel and the measures. HESAS yielded satisfactory performance, showed an estimated precision of 82%, a recall of 84% and a F1 measure of 83%.

6.2 Future Works

In this research work, we have presented the efforts exerted to design and implement academic advising dialogue system for Ethiopian Public Universities. However, developing a full-fledged Dialogue System for Advising Ethiopian Public Universities Students requires the involvement of several professionals from different disciplines such as computer science, psychology, linguistics and other related fields. As a result, additional features, improvements, and modifications should be incorporated to come up with an effective and efficient academic advising dialogue system.

Hence, we proposed the following recommendations for the future research directions:

- The immediate developments for this system include expanding AS for multiple semesters and adding a computerized method for submitting the academic record.
- The advising corpus will be expanded to provide the data to integrate statistical methods into the dialogue manager. Additional future updates include an escalation mechanism to forward selected student conversations to the advisors, adding course information from other academic departments and traditional advising tasks.
- The spelling dictionary will also be expanded by adding more technical terms.
- Building domain specific advising dialogue system for specific applications. The advising dialogue system can be easily customized and implemented to satisfy the needs of some organizations and institutions for specific application domains such as: health, education, and other domain specific.

References

[1] Education in Ethiopia, retrieved from https://en.wikipedia.org/wiki/Education_in_Ethiopia, Last accessed on March 2016.

[2] Helen Tesfaye, "Contributing Factors for Female Students' Attrition in Science Education: The Case of Faculty of Science at Addis Ababa University", Unpublished Master's Thesis, Institute of Gender Studies, Addis Ababa University, Ethiopia, July 2010.

[3] Student selection and retention at the University of Asmara, retrieved from http://www.rug.nl/research/portal/files/2975412/c2.pdf, Last accessed on July 14, 2016.

[4] Mary E. Taylor, "The Changing Advising Needs of Undergraduate Students", Unpublished Master's Thesis, Department of Educational Leadership and Policy Studies, Virginia Polytechnic Institute and State University, Virginia, July 26, 2000.

[5] V. N. Gordon, W. R. Habley, T. J. Grites, and Associates, Academic Advising: A Comprehensive Handbook, National Academic Advising Association, 2nd ed. San Francisco: Jossey-Bass, 2008.

[6] C. C. Kulik, J. A. Kulik, and B. J. Schwalb, "College Programs for High-Risk and Disadvantaged Students: A Meta-Analysis of Findings", Review of Educational Research, Vol. 53, No. 3, 1983, pp. 397–414.

[7] T. Gaines, "Technology and Academic Advising: Student Usage and Preferences", NACADA Journal, Vol. 3, No. 1, 2014, pp. 43-49.

[8] G. M. Ramirez and R. J. Evans, "Solving the Probation Puzzle: A Student Affirmative Action Program", NACADA Journal, Vol. 8, No. 2, 1988, pp. 34-45.

[9] M. J. Leonard, "Advising Delivery: Using Technology", in Academic Advising: A Comprehensive Handbook, National Academic Advising Association, 2nd ed. San Francisco: Jossey-Bass, pp. 292-306, 2008.

[10] National Academic Advising Association, "Advising Technology Innovation Awards", retrieved from https://www.nacada.ksu.edu/programs/Awards/EPublication.htm, Last accessed on April 2016.

[11] H. S. Hart, R. B. Hussey, M. J. Leonard, J. Levin, and S. M. Winck, "eLion: Penn State's Comprehensive Web-Based Academic Advising System", in The Mentor: An Academic

Advising Journal, retrieved from http://dus.psu.edu/mentor/bookstore/elion-advising-system, Last accessed May 2016.

[12] B. C. McMahan and R. A. Bates, "An Automatic Dialog System for Student Advising", Journal of Undergraduate Research at Minnesota State University, Mankato, Vol. 10, Article 6, 2010.

[13] Academic Advising and Career Center, "eAdvising", University of Michigan-Flint, Flint, Michigan, retrieved from http://www.umflint.edu/advising/eadvising.htm, Last accessed on March 2016.

[14] E. R. White and M. J. Leonard, "Faculty Advising and Technology", in Faculty Advising Examined: Enhancing the Potential of College Faculty as Advisors, G. L. Kramer, E. D. Bolton, and MA: Anker Publishing Company, 2003.

[15] D. Jurasky and J. H. Martin, "Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition", 2nd Ed, New Jersey: Pearson Education, 2008.

[16] Feghali, T., Zbib, I., and Hallal, S., "A Web-based Decision Support Tool for Academic Advising", Journal of Education Technology and Society, Vol. 14, No. 1, 2011, p. 82-84.

[17] G. L. Kramer and M. McCauley, "High Tech and High Touch: Integrating Information Technology in the Advising Process", retrieved from http://www.nacada.ksu.edu/Monographs/index.htm, Last accessed on March 2016.

[18] List of Universities and Colleges in Ethiopia, retrieved from https://en.wikipedia.org/wiki/List_of_universities_and_colleges_in_Ethiopia, Last accessed on March 2016.

[19] E. M. Latorre-Navarro and J. G. Harris, "An Intelligent Natural Language Conversational System For Academic Advising", International Journal of Advanced Computer Science and Applications *(*IJACSA), Vol. 6, No. 1, 2015.

[20] T. M. Rao, S. Coleman, and C. Hollenbeck, "Advisor: An Expert System for Student Advisement", Proceedings of the 15th Annual Conference on Computer Science, New York: ACM, 1987, pp.32-35.

[21] B. B. Crookston, "A Developmental View of Academic Advising as Teaching", Journal of College Student Personnel, Vol. 13, 1972, pp. 12-17.

[22] M. C. Chando, "Predicting Advising Style Preference from Student Characteristics", Unpublished Doctoral dissertation, University of Memphis, 1997.

[23] L. D. Heisserer and P. Parette, "Advising At-Risk Students in College and University Settings", Journal of College Student, Vol. 36, No. 1, 2002, pp. 69-84.

[24] D. Yarbrough, "The Engagement Model for Effective Academic Advising With Undergraduate College Students and Student Organizations", Journal of Humanistic Counseling, Education and Development, Vol. 41, No. 1, 2002, pp. 61-68.

[25] S. Russel and P. Norvig, "*Artificial Intelligence. A Modern Approach*", New Jersey: Prentice Hall, 1995.

[26] D. Liddy, "Natural Language Processing. In Encyclopedia of Library and Information Science", 2nd Ed, NY. Marcel Decker, Inc., 2001.

[27] Bose and Ranjit, "Natural Language Processing: Current State and Future Directions", International Journal of the Computer, the Internet and Management, Vol. 12, No.1, pp. 1 – 11, January – April, 2004.

[28] K. Eliasson, "The Use of Case-Based Reasoning in a Human-Robot Dialogue System", Unpublished Undergraduate Thesis, Department of Computer and Information Science, Linkoping University, 2006.

[29] A. Turing, "Computing Machinery and Intelligence. Mind", Vol. 59, No. 236, pp. 433-460, 1950.

[30] H. Kamp and U. Reyle, "From Discourse to Logic: Introduction to Model Theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory", Kluwer Academic Publishers, 1993.

[31] J. Groenendijk and M. Stokhof, "On the Semantics of Questions and the Pragmatics of Answers", Unpublished PhD Thesis, pp. 143-170, 1984.

[32] J. Alexanderson, E. Maier, and N. Reithinger, "A Robust and Efficient Three-Layered Dialogue Component for Speech-to-Speech Translation System", Technical Report 50, Federal Ministry of Education, Science, 1994.

[33] J. Allen, D. Byron, M. Dzikovska, George Ferguson, Lucian Galescu, and A. Stent, "Towards Conversational Human-Computer Interaction", AI Magazine, 2001.

[34] T. Hemphill, J. Godfrey, and R. Doddington, "The ATIS Spoken Language Systems Pilot Corpus", In Proceedings of DARPA Speech and Natural Language Workshop, pp. 96-101, 1990.

[35] H. Aust, M. Oerder, F. Seide, and V. Steinbiss, "The Philips Automatic Train Timetable Information System", In Speech Communication, Vol. 17, pp. 249-262, 1995.

[36] W. Wahlster, "Verbmobil: Foundations of Speech-to-Speech Translation", Springer, Berlin, Germany, 2000.

[37] R. Cooper, S. Ericsson, I. Lewin, and J. Rupp, "Dialogue Moves in Negotiative Dialogue", Technical Report Deliverable D1.2, Siridus, University of Gothenburg, 2000.

[38] M. Buckley, "An Agent-Based Platform for Dialogue Management", In Proceedings of the Tenth ESSLLI Student Session, Edinburgh, Scotland, pp. 33-45, 2005.

[39] A. Heeman and J. Allen, "The TRAINS 93 dialogues", Technical note 94-2, University of Rochester, Rochester, New York, 1995.

[40] J. Allen, G. Ferguson, and A. Stent, "An Architecture for More Realistic Conversational Systems", In Proceedings of Intelligent User Interfaces (IUI-01), Santa Fe, NM, 2001.

[41] J. Allen, D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent, "An Architecture for a Generic Dialogue Shell", Journal of Natural Language Engineering, Vol. 6, No. 3, pp. 1-16, 2000.

[42] V. Aleven, R. Koedinger, and K. Cross, "Tutoring Answer Explanation Fosters Learning with Understanding", In Proceedings of AIED-99, Amsterdam, pp. 199-206, 1999.

[43] C. Graesser, K. Wiemer-Hastings, P. Wiemer-Hastings, and R. Kreuz, "Autotutor: A Simulation of A Human Tutor", Cognitive Systems Research, Vol. 1, pp. 35-51, 1999.

[44] E. M. Latorre-Navarro and J. G. Harris, "A Natural Language Conversational System for Online Academic Advising", in Proceedings of the Twenty-Seventh International Florida Artificial Intelligence Research Society Conference, Vol. 6, No. 1, 2015.

[45] S. Arora, K. Batra, and S. Singh, "Dialogue System: A Brief Review", retrieved from *https://arxiv.org/pdf/1306.413.pdf, Last accessed on August 8, 2016.*

[46] A. Fiedler and M. Gabsdil, "Supporting Progressive Refinement of Wizard-Of-Oz Experiments", In Proceedings of the ITS - Workshop on Empirical Methods for Tutorial Dialogue Systems, Spain, pp. 62-69, 2002.

[47] A. Ranta, "Grammatical Framework—a Type-Theoretical Grammar Formalism", Journal of Functional Programming, Vol. 14, No. 2, 2004, pp. 145-189.

[48] F. McTear, "Modelling Spoken Dialogues with State Transition Diagrams: Experiences with the CSLU Toolkit", In Proceedings of the 5th International Conference on Spoken Language Processing, Sydney, Australia, 1998.

[49] CSLU Toolkit, retrieved from http://cslu.cse.ogi.edu/toolkit/, Last accessed on August 8, 2016.

[50] G. Fliedner and D. Bobbert, "DiaMant: A Tool for Rapidly Developing Spoken Dialogue Systems", In Proceedings of the 7th Workshop on the Semantics and Pragmatics of Dialogue (DiaBruck), Wallerfangen, Germany, 2003.

[51] NUANCE Developers' Toolkit, retrieved from http://www.nuance.com/, Last accessed on August 8, 2016.

[52] VoiceXML, retrieved from *http://www.voicexml.org/, Last accessed on August 8, 2016.*

[53] D. Traum and S. Larsson, "The Information State Approach to Dialogue Management", In J. van Kuppevelt and R. Smith, editors, Current and New Directions in Discourse and Dialogue. Kluwer, 2003.

[54] Allen, J., Ferguson, G., and Stent, A., An architecture for more realistic conversational systems. In *IUI '01: Proceedings of the 6th international conference on Intelligent user interfaces*, Santa Fe, New Mexico, 2001, United States, pp. 1– 8. ACM Press.

[55] P. R. Cohen and C. R. Perrault, "Elements of a Plan Based Theory of Speech Acts. *Cognitive Science"*, Vol. 3, No. 3, pp. 177– 212, 1979.

[56] C. R. Perrault and J. Allen, "A plan-based Analysis of Indirect Speech Acts", *American Journal of Computational Linguistics*, Vol. *6*, pp. 167–182, 1980.

[57] R. Wilensky, *Planning and Understanding: A Computational Approach to Human Reasoning*. Addison-Wesley, 1983.

[58] F. J. Damerau, “A Technique for Computers to Detect and Correct Spelling Errors.”, Commun ACM Vol. 7, pp. 171-176, 1964.

[59] K. Kukich, “Techniques for Automatically Correcting Words in Text”, Computing Surveys, Vol. 24, no. 4, pp. 377-439, 1992.

[60] R. Mitton, “Spelling Correctors and the Misspellings of Poor Spellers”, Inf. Process. Manage, Vol. 23, no. 5, pp495-505, 1987.

[61] D. K. Mark, W.C. Kenneth, and A.G. Willian, “A Spelling Correction Program Based on a Noisy Channel Model”, International Conference on Computational Linguistics, Proceedings of the 13th conference on Computational linguistics, Vol. 2, pp. 205-210, 1990.

[62] A. V. Aho and M. J. Corasick, “Fast Pattern Matching: An Aid to Bibliographic Search”, Commun. ACM, Vol. 18, pp. 333-340, 1975.

[63] T. N. Turba, “Checking for Spelling and Typographical Errors in Computer-based Text”, SIGLAN-SIGOA Newslett, pp. 51-60, 1981.

[64] J. L. Peterson, “Computer Programs for Detecting and Correcting Spelling Errors”, Communication. ACM 23, 12, pp. 676-684, 1980.

[65] C. M. Leung, E. Y. M. Tsang, F. S. S. Lam and D. P. C. Wai, “Intelligent Counseling System: A 24 x 7 Academic Advisor,” EDUCAUSE Quart. 33, no. 4, 2010, retrieved from *http://www.educause.edu/edUCaUSe+Quarterly/edUCaUSeQuarterlymagazineVolum/intelligentCounselingSystema24/219101, Last accessed on June 30, 2017.*

[66] A. Olney, M. Louwerse, E. Matthews, J. Marineau, H. Hite-Mitchell, and A. Graesser, “Utterance Classification in AutoTutor”, HLT-NAACL Building Educational Application. Using NLP, *Assoc for Computational Linguistics*, Vol. 2, pp.1-8, USA, 2003.

[67] E. Sneiders, “Automated FAQ Answering with Question-Specific Knowledge Representation for Web Self-Service”, *Proceedings of 2nd International Conference Human System Interaction*, IEEE, Catania, Italy, pp. 298-305, May 21-23, 2009.

[68] F. Albalooshi1 and S. Shatnawi, “HE-Advisor: A Multidisciplinary Web-Based Higher Education Advisory System”, Global Journal of Computer Science and Technology, Vol. 10, Issue 7, Ver. 1.0, September 2010.

Appendixes

Appendix A: Sample for International English Words

Aaron
Aaron's
Abba
Abba's
Abbe
Abbe's
Abbey
Abbey's
.
.
.
.
.
zucchini
zucchini's
zucchinis
zwieback
zwieback's
zygote
zygote's

Appendix B: List of Technical Terms

CESt1023
CESt-M1023
CoSc1011
CoSc1012
CoSc1014
CoSc2043
CoSc2044
CoSc2045
CoSc2061
CoSc2071
CoSc2072
CoSc2082
CoSc2083
CoSc2084
CoSc3062
CoSc3063
CoSc3091
CoSc3092
CoSc3101
CoSc3111
CoSc3112
CoSc3112

CoSc3131

CoSc3131

CoSc3141

CoSc4132

CoSc4142

CoSc4151

CoSc4152

CoSc4161

CoSc4162

CoSc4163

CoSc4171

CoSc4181

CoSc4191

CoSc-M1011

CoSc-M2041

CoSc-M2051

CoSc-M2061

CoSc-M2071

CoSc-M2081

CoSc-M3091

CoSc-M3101

CoSc-M3111

CoSc-M3121

CoSc-M3131

CoSc-M3141

CoSc-M4152

CoSc-M4161

CoSc-M4171

CoSc-M4181

CoSc-M4191

EEng2041

EEng2042

EnLa1011

EnLa1012

EnLa-M1013

Math1012

Math1015

Math2051

Math2081

Mgmt3101

Mgmt-M3101

Phil1024

Stat3071

.

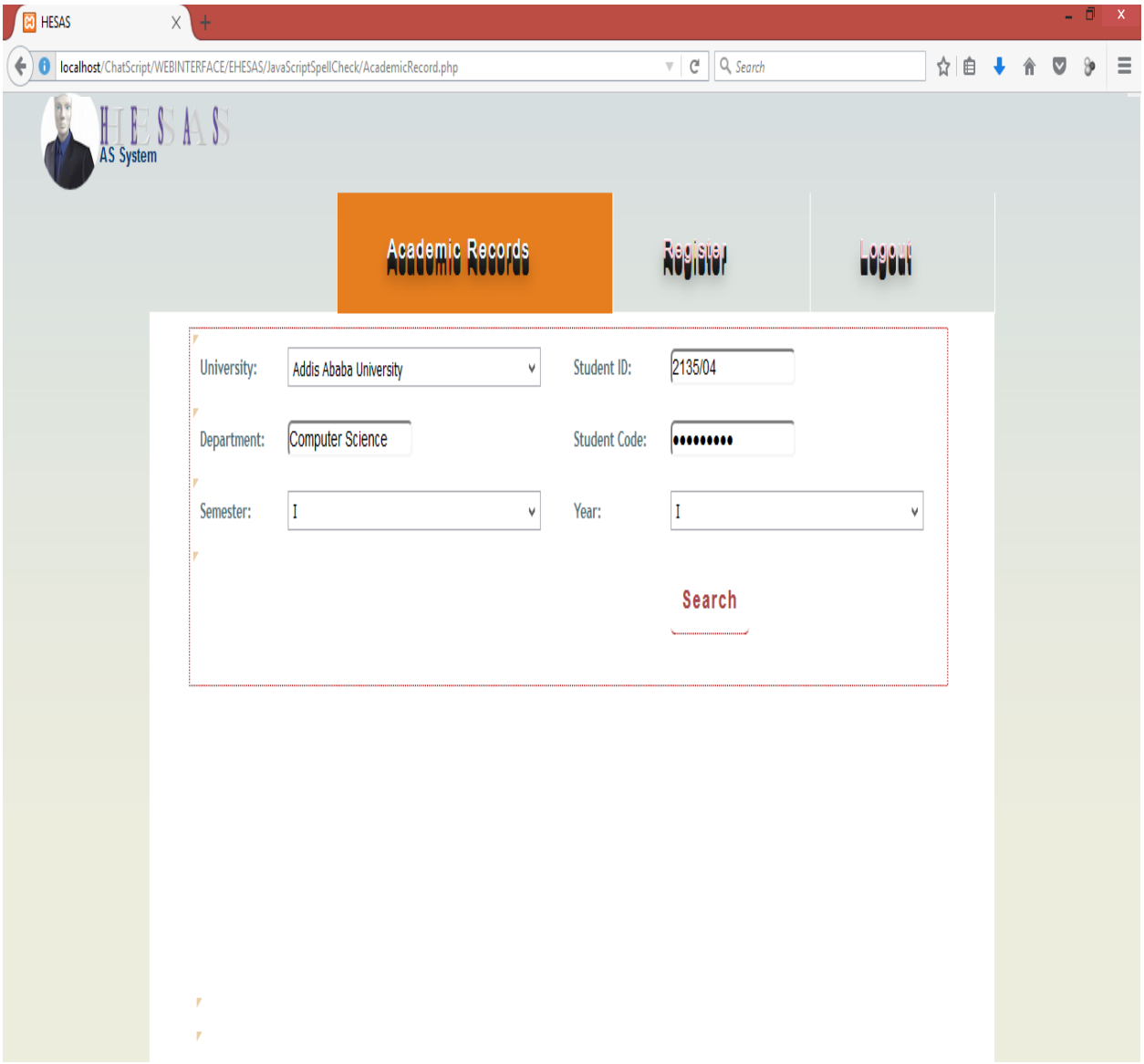
.

.

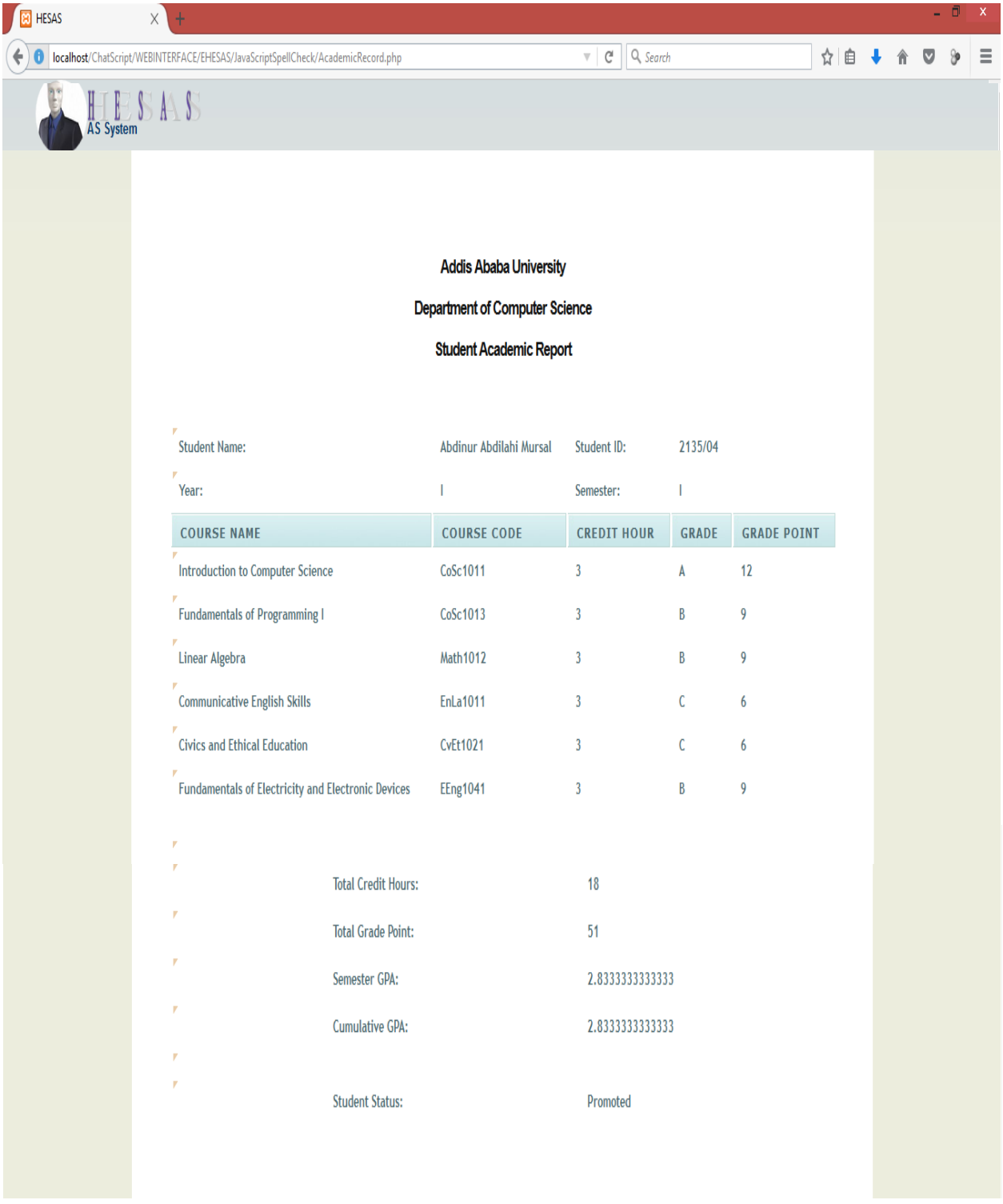
.

.

Appendix C: Segment of Template for Viewing Academic Records



Appendix D: Sample for Previous Academic Record



Addis Ababa University

Department of Computer Science

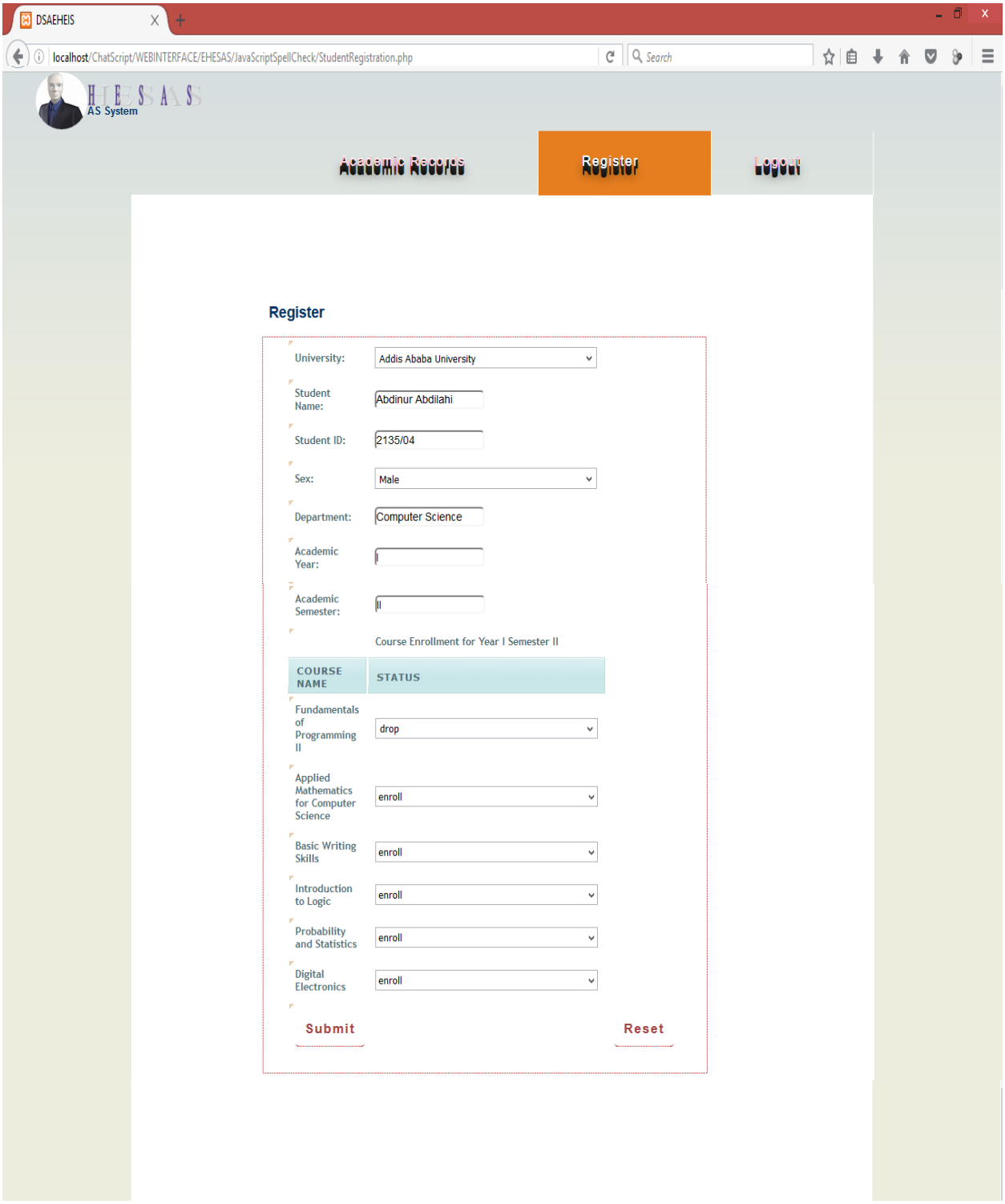
Student Academic Report

Student Name:	Abdinur Abdilahi Mursal	Student ID:	2135/04
Year:	I	Semester:	I

COURSE NAME	COURSE CODE	CREDIT HOUR	GRADE	GRADE POINT
Introduction to Computer Science	CoSc1011	3	A	12
Fundamentals of Programming I	CoSc1013	3	B	9
Linear Algebra	Math1012	3	B	9
Communicative English Skills	EnLa1011	3	C	6
Civics and Ethical Education	CvEt1021	3	C	6
Fundamentals of Electricity and Electronic Devices	EEng1041	3	B	9

Total Credit Hours:	18
Total Grade Point:	51
Semester GPA:	2.8333333333333
Cumulative GPA:	2.8333333333333
Student Status:	Promoted

Appendix E: A Segment of the Template for Registering for Courses in AS



Appendix F: Sample Code for Courses Enrollment

```php
<?php
/* Student Registration php */
class StudentRegistration{

    public $credit_hours;
    public $course_Code;
    public $course_ECTS;

  private function connectToDB(){

        $con = mysql_connect('localhost','root','');

    if(!$con){
        ?>
        <script type="text/javascript">
          alert('Could Not Connect:');
        </script>
        <script language="JavaScript" type="text/javascript" >
         location.href="StudRegistration.php";
        </script>
       <?
    }

       mysql_select_db('HESAS_AS',$con);
  }//end of function connect
```

```php
private function check_Student_Existance($stud_id, $stud_university, $stud_dept){  //A function that checks the existance of students.
  $stud_exist = false;
  $select_from_StudentInformationTable = "SELECT * FROM StudentInformationTable WHERE university='$stud_university' and stud_id='$stud_id' and
                                          stud_department='$stud_dept'";
  $Stud_query_result = mysql_query($select_from_StudentInformationTable);
  if($Stud_query_result){
    $stud_exist = true;
  }
  else {
    $stud_exist = false;
  }
  return $stud_exist;
}//end of check_Student_Existance() method
```

```
private function check_Stud_Previous_Academic_Records($stud_id, $current_sem, $current_year){ //A function that checks the students previous academic
                                                                                              //records, in order to allow the students to register for
                                                                                              the courses for the current semester....
 $select_From_GPARecordTable = "";
 $isAllowed = false;
 $prev_sem;
 $prev_year;

 if($current_year == 1 && $current_sem == 2){
    $select_From_GPARecordTable = "SELECT * FROM GPARecordTable WHERE stud_id = '$stud_id' and academic_year = 'I' and academic_sem = 'I'";
 }
 else if($current_year == 2 && $current_sem == 1){
   $select_From_GPARecordTable = "SELECT * FROM GPARecordTable WHERE stud_id = '$stud_id' and academic_year = 'I' and academic_sem = 'II'";
 }
 else if($current_year == 2 && $current_sem == 2){
   $select_From_GPARecordTable = "SELECT * FROM GPARecordTable WHERE stud_id = '$stud_id' and academic_year = 'II' and academic_sem = 'I'";
 }
 else if($current_year == 3 && $current_sem == 1){
   $select_From_GPARecordTable = "SELECT * FROM GPARecordTable WHERE stud_id = '$stud_id' and academic_year = 'II' and academic_sem = 'II'";
 }
 else if($current_year == 3 && $current_sem == 2){
   $select_From_GPARecordTable = "SELECT * FROM GPARecordTable WHERE stud_id = '$stud_id' and academic_year = 'III' and academic_sem = 'I'";
 }
 else if($current_year == 4 && $current_sem == 1){
   $select_From_GPARecordTable = "SELECT * FROM GPARecordTable WHERE stud_id = '$stud_id' and academic_year = 'III' and academic_sem = 'II'";
 }

  $query_result = mysql_query( $select_From_GPARecordTable);

  if($query_result){
      $semester_gpa = 0.0;
      $Cumulative_GPA = 0.0;
     while($row = mysql_fetch_array($query_result)){

      $semester_gpa = $row['stud_sgpa'];
      $Cumulative_GPA = $row['stud_cgpa'];
      $prev_sem = $row['academic_sem'];
      $prev_year = $row['academic_year'];

      $semester_gpa = floatval($semester_gpa);
      $Cumulative_GPA = floatval($Cumulative_GPA);
     }

   if (($semester_gpa>=1.5) && ($prev_sem == "I") && ($prev_year == "I")){ //For First year first Semester....
        $isAllowed = true;
   }
   else if(($semester_gpa>=1.75 && $Cumulative_GPA>=1.75) && ($prev_sem == "II" && ($prev_year == "I")){//For First year second Semester...
        $isAllowed = true;
   }
  else if(($semester_gpa>=1.75 && $Cumulative_GPA>=1.75) && ($prev_sem == "II" && $prev_year == "I")){//For second year and above...
        $isAllowed = true;
   }
  else if(($semester_gpa>=1.75 && $Cumulative_GPA>=2.00) && (($prev_sem == "I" && $prev_year == "II") || ($prev_sem == "II" && $prev_year == "II") || ($
    prev_sem == "I" && $prev_year == "III") || ($prev_sem == "II" && $prev_year == "III") )){//For second year and above...
        $isAllowed = true;
   }

  }

 return $isAllowed;

}//end of check_Stud_Previous_Academic_Records() method
```

```
private function check_prerequisites($Course_title){// A function that checks the prerequisites of each of the courses to be registered...
        $isCoursePassed = false;
        $select_From_CS_CurriculumTable = "SELECT * FROM CS_CurriculumTable WHERE course_title = '$Course_title'";
        $query_result = mysql_query($select_From_CS_CurriculumTable);
        if($query_result){
        while($row = mysql_fetch_array($query_result)){
         $this->prerequistes = $_POST['prerequistes'];
        }
    }
    if($prerequistes!= "None"){
      $select_From_PreviouslyTakenCourseRecordTable = "SELECT * FROM PreviouslyTakenCourseRecordTable WHERE course_code = '$prerequistes'";
      $query_result = mysql_query($select_From_PreviouslyTakenCourseRecordTable);
      if($query_result){
        while($row = mysql_fetch_array($query_result)){
         $this->prev_course_grade = $_POST['course_grade'];
        }
        if($prev_course_grade!= "F" || $prev_course_grade!= "Fx" || $prev_course_grade!= "NG" || $prev_course_grade!= "I" || $prev_course_grade!= "DO" ||
          $prev_course_grade!= "W"){
          $isCoursePassed = true;
        }
      }
    }
   if($prerequistes== "None"){
        $isCoursePassed = true;
    }
    return $isCoursePassed;

}//end of check_prerequisites() method
```

```
public function register_Student_for_Current_Semester($University_Name,$StudName,$StudID,$StudSex,$StudDep,$AcademicYear,$AcademicSem,$List_of_Courses,
                                                      $List_of_CoursesStatus){
                    $List_of_Failled_Courses = array();
                    failed = 0;
                    register = 0;
                    $List_of_registered_courses = array();

                    connectToDB();

                if(check_Student_Existance($StudID, $University_Name, $StudDep) && check_Stud_Previous_Academic_Records($StudID, $AcademicSem,
                    $AcademicYear)){

                   for ($i = 0; $i<count($List_of_Courses); $i++){

                    if(!check_prerequisites($List_of_Courses[$i])){
                      $List_of_Failled_Courses[$failed] =  $List_of_Courses[$i];
                      $failed++;
                     }
                    else{
                      getCourseInformation($List_of_Courses[i]);
                      $Course_title = $List_of_Courses[$i];
                      $course_status = $List_of_CoursesStatus[$i];
                      $sql = "INSERT INTO Current_semester_course_Record(stud_id, course_name,course_code, credit_hour, academic_year, academic_sem,
                        Date_of_reg, status)VALUES ('$StudID','$Course_title','$course_Code','$credit_hours',
                       '$AcademicYear','$AcademicSem','$reg_Date', '$course_status')";
                       //mysw
                       $query_enroll_result = mysql_query($sql);
                       if($query_enroll_result){
                         $List_of_registered_courses[$register] = $Course_title;
                       }
                    }
                   }//end of for loop
                }//end of outer if

                else{
                        ?>
                        <script type="text/javascript">
                         alert('You are unable to register for the current semester, due to, poor academic performance in the previous semester!');
                        </script>
                        <script language="JavaScript" type="text/javascript" >
                         location.href="StudRegistration.php";
                        </script>
                        <?

                }//end of else

}//end of register_Student_for_Current_Semester() method


}//end of class StudentRegistration

?>
```

Appendix G: Sample List of FAQs

1. I want to choose my courses for the next semester.
2. What courses did I select for next semester?
3. What courses do you recommend I take?
4. How many credits does Object Oriented programming have?
5. What is the prerequisites of Internet Programming?
6. What is the course number of Complier Design?
7. Can I take Complier Design without the prerequisite?
8. How many courses do I have to take?
9. What does assessment methodology?
10. I want to graduate with great distinction, what does its requirement?
11. Is it ok if I take less than 15 credit hours?
12. Which programming courses am I required to take?
13. Does Object Oriented Programming have a lab?
14. How many times can I repeat a course?
15. What is the HE drop rule?
16. What does HE stands for?
17. My GPA is below 2.00, am I kicked out of the major?
18. Can I know make-up examination procedure?
19. I made Fx in a major course, can I sit for re-exam?
20. Tell me the mark interval for B+?
21. What does that mean breaking examination rule?
22. What is the procedure of make-up examination?
23. What will happen if i drop below 15 credits?
24. What is the max number of hours I can take in a semester?
25. What major courses must I take?
26. What is probation shall mean?

27. What is the procedure of re-examination?
28. Define major?
29. What GPA do I need to complete undergraduate program?
30. Is that possible to graduate with F grade?
31. Can I take online courses?
32. What are elective I courses?
33. How many ECTS does elective I have?
34. How do I register for the courses?
35. Can I see my academic records?
36. I want to drop a course?
37. What is the deadline for dropping a course?
38. What is add/drop means?
39. Would you like to introduce me courses taken at 3rd year semester 2?
40. How can I apply for readmission?
41. How can I complete the clearance?
42. Can you show me grading system?
43. What is the requirements for academic dismissal?
44. I want to know the pass point of year 1 first semester?
45. What does I do to apply for continuing education programs?
46. What is the course title for CoSc1011?
47. Does fundamentals of programming I has a lab?
48. What tool do you recommend me to practice a lab for C++?
49. Please tell me a reading materials for Introduction to Computer Science
50. How many number of hours do I have to take in year 1 1st sem?
51. What is the procedure to withdraw?
52. What must I do to dropout?
53. What is the criteria for full time admission in undergraduate program at Ethiopian higher education institution?
54. I want to know the semester academic achievements policy?

55. What tutoring service is available?
56. What is the minimum and maximum total credits required to graduate in undergraduate program?
57. Can I know methods of examination in Public universities?
58. Please tell me the methods of examination in HE?
59. Can you tell me the transfer procedure
60. What GPA do I need to pass to the next semester?
61. What does the requirements to graduate with very great distinction?

Appendix H: Sample from Students Feedback

1. General, how did you get HESAS, the academic advising system?

   I think HESAS is a great tool to be used in conjunction with traditional face-to-face advising system. HESAS helped to reduce the time needed to lift semesterly advising holds and allowed to focus more on one-on-one student interactions.

2. What positive features can you highlight?

   HESAS's ability to recognize the course information, rules and procedures of the Ethiopian public universities and term definitions was extremely helpful. Allows public universities students to know about the curriculum and the rules and regulations of Ethiopian public universities.

3. What negative features can you highlight?

   The negative features of HESAS is allowing course selection only for one semester and lack of handling all the academic policy of public universities.

4. In what ways might this advising system be improved?

   Incorporating the course selection process for more semester would be a good way to improve the system.

Final Comments/Suggestions/Recommendations:

Overall, I was very pleased with my interactions with HESAS.

Declaration

I, the undersigned, declare that this research is my original work and has not been presented for a degree in any other university, and that all source of materials used for the thesis have been duly acknowledged.

Declared by:

Name: _____

Signature: _____

Date: _____

Confirmed by advisor:

Name: _____

Signature: _____

Date: _____