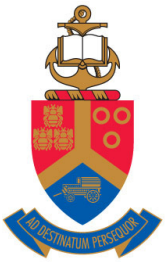

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Caracal Automated Marking Software Requirements Specification

Version: (1.0.0)

Edited by

Oratile Motswagosele
Mankgwanyane Tlaka
Lesego Makaleng
Kenneth Mangwane
Tlou Lebelo

University of Pretoria

2017

Project Client: Caracal Research

Contents

1 Introduction

E-learning is a topic that is receiving a huge amount of attention worldwide and especially in South Africa. Currently most e-learning in schools consists of taking paper material and making it available on electronic devices such as tablets.

LMS systems such as Moodle gives the option to design and enter electronic courses and presenting assessments in the form of multiple choice and other one-dimensional methods. The available commerical systems are available but cost is a problem.

2 Vision

Caracal Automated Marking is aimed at developing a metalanguage for the electronic assessment of grade 12 mathematical question in which the steps employed to arrive at the answer can be assessed, not only the final answer. The system should be ideally available as a Moodle plugin.

The proposed solution, ideally would be integrated into the Moodle open learning platform as a plugin. The system should accept three documents i.e. a document with mathematics question; a document with the memorandum and a document with student answers. The system should analyse answer document and give a mark.

At the end, the proposed system should ideally act as a methodology through which any given problem and matching memorandum can be entered.

3 Background

E-learning is a topic that receives a lot of attention worldwide and even more in Africa. The problem of educating dispersed people with a few resources is a headache that troubles many governments. New initiatives are being launched daily to try and solve the problem.

Taking a deeper look at the grade 12 question paper marking process, it is realised that most of the time, individuals are still needed who have the knowledge and skill to go through all the answered mathematics scripts and grade them correctly. This process has proven to be time costly and human error tends to creep up at a minimal rate.

4 Architecture requirements

4.1 Access channel requirements

The Caracal Automated Marking system will be web-based, and ideally available as a Moodle plug-in. Therefore it should be accessible on most platforms such as Android, Unix and iOS. Also, the browser support may be dependent on the implemented system plug-ins, however, currently Moodle 2.8 browser support is as follows:

- Google Chrome (Min. Version: 30.0 and Rec. Version: Latest)
- Mozilla Firefox (Min. Version: 25.0 and Rec. Version: Latest)
- Apple Safari (Min. Version 6 and Rec. Version: Latest)
- Microsoft Internet Explorer (Min. Version 9 and Rec. Version: Latest)

Also, Moodle provides accessibility plug-ins. With the requirements listed above, the system should therefore be accessible.

4.2 Integration requirements

1. Data Management
    - Apache Flink - This is a data streaming platform that will be used to manage the streaming of data such as memo and test documents, user registrations and marking requests.
2. Optical Character Recognition (OCR)

- Mathpix - This is an API used to read handwritten mathematical equations, graphs, matrices etc. It will be used to read math equations from the memos and test papers of the users.

- AbbyyOCR4 CLI Linux - This is another API which also reads handwritten characters from scanned documents and images. This API will be used to read all characters, especially for math questions such as sequence and series and also financial math questions for compound interest, simple interest etc.

4.3 Architecture constraints

- A web based solution. This means we need to use web-technologies such as php, html in our solution, especially for the user interface.
- Moodle programming. The solution should be rendered as a moodle plugin (ideally).

5 Functional requirements and application design

5.1 Modularization

1. **Access Module**
   Scope:
   The scope of this module will be to give users a graphical user interface, which will allow them to login and register and to also be able to have access to the system as whole. This module will be in charge of deploying a GUI on multiple browsers i.e. FireFox, Google Chrome, Microsoft Edge. It will provide a friendly user functionality of the whole system.

   - use case diagrams

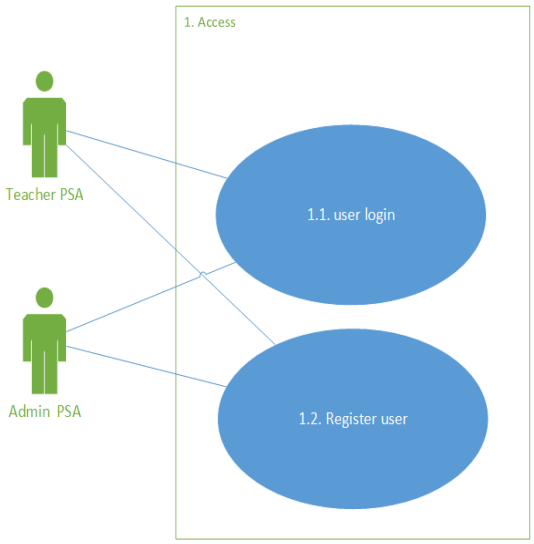

Figure 1: I/O module use case diagram

2. **User Management Module**
   Scope:
   This module will be in charge of managing user information and the different types of users. The different types of users of this system will be as follows:

   - Admin - The role of this user will be to have access to uploading question papers and memorandums, and general access to sensitive information of the system. This user can also delete/approve other users below this level i.e. Teachers from the system.
   - Teacher - The role of this user will be to upload the test papers of students i.e. their class of students, and recieve marks of the papers they have uploaded. This user cannot have direct access to the memo or question paper. This user also needs approval from the admin before being registered. This is to

prevent students registering as fake teachers and uploading tests to get access to the marking system .

- use case diagrams

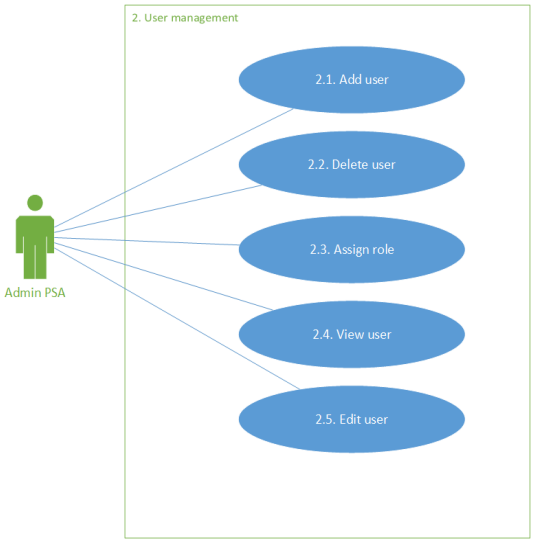

Figure 2: user management module use case diagram

3. **Notifications Module**
   Scope:
   This module will be responsible for giving feedback to the users of the system. A general example would be when a teacher uploads 100 papers to be marked. If the system can't handle the request immediately, the notifications module will have to notify that teacher when their papers are ready to be retrieved after marking is completed. The module will deploy notifications via notices on the system and also emails for more complex notifications like paper collection.

   - use case diagrams

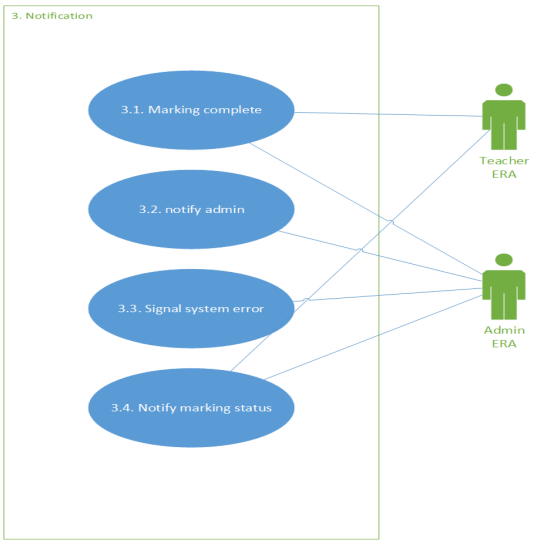

Figure 3: Notifications module use case diagram

4. **Data Streaming and Security Module**
   Scope:
   This module will be responsible for the handling and management of data in all its forms. The module

will firstly make sure that requests relating to data uploads and downloads are handled as quick as possible through an external data management tool API. The module will also make sure that before data is transported from server to client and vice versa, it will be decrypted and encrypted respectively. This module will basically be responsible for data integrity and data protection of the system.

- use case diagrams

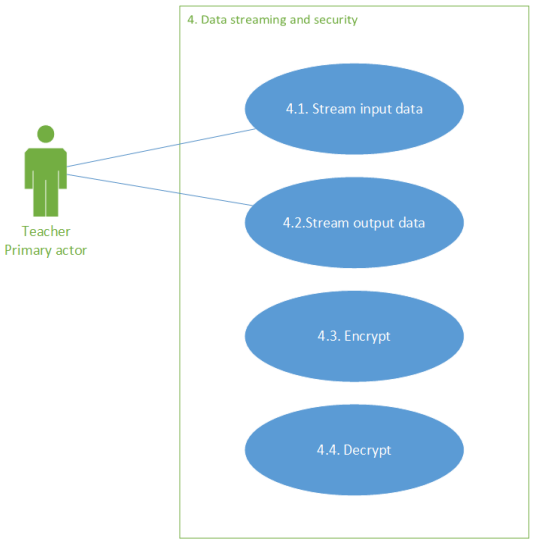

Figure 4: data and security module use case diagram

5. **Marking Module**
Scope:
This module will be responsible for the actual marking of uploaded scripts. This module will perform at least 3 functions, which are:

- Paper Slicing - This means that the paper will be analyzed, and broken into smaller pieces for OCR functionality.
- Comparing - This means after breaking the paper into smaller pieces, AI analytical algorithms will be used to properly compare and mark the answers against the memo.
- Persisting - This means after marking is complete, the marks and relevant answer sheet will be stored in the database.

- use case diagrams

Figure 5: marking module use case diagram use case diagram

6. **Reporting Module**

Scope:
The Scope of this module will be to create performance and statistical reports of the student marks and basically give feedback of how the marks went, what problems most students faced, suggestions etc. This module will compile a report using data from the database and analytical data algorithms.

- use case diagrams

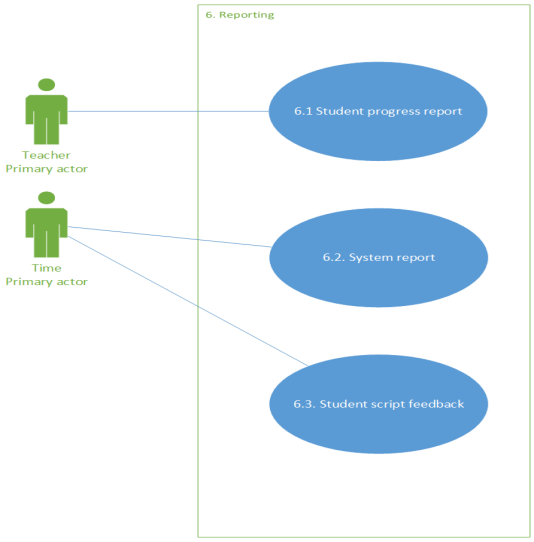

Figure 6: reporting module use case diagram

7. **I/O Module**

Scope:
This module will be in charge of the inputting and outputting of documents. One of the core functionalities for this module, will be to edit the answer sheet and append the marks calculated by the marking module to the answer sheet, and then outputting that document back to the user for download. Managing where documents are inputted to, formats and structure will also be amongst the scope of this module.

- use case diagrams

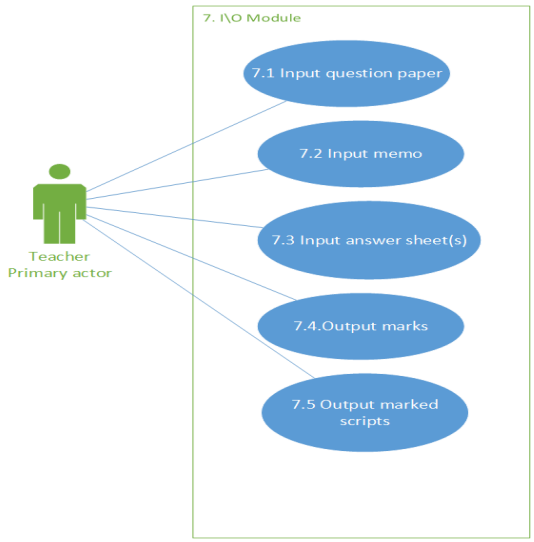

Figure 7: I/O module use case diagram

5.2 Use case prioritization

5.2.1 Critical:

- Register User
- Input Question Paper
- Input Memo
- Input Student Answer Script
- Save All Input Files
- Compare Student Answer Script against Memo
- Persist and save obtained mark
- Output Marks
- Output Marked Script

5.2.2 Important:

- Validate User Information/Credentials
- Validate Question Paper
- Validate Memo
- Validate Student Answer Script
- Slice Question Paper
- Slice Memo
- Slice Student Answer Script
- Convert Sliced Question Paper Sections To Suitable Format
- Convert Sliced Memo Sections To Suitable Format
- Convert Sliced Student Answer Script Sections To Suitable Format
- Associate Sliced Question Paper Section; Memo Section And Student Answer Script Section with Student Details

5.2.3 Nice-To-Have:

- Find Student Details From Answer Script
- Compile a report which includes information such as highest marks, average, lowest, overall performance etc.
- Learn Way Of Answering
- Reason For Given Mark

5.3 Use case/Services contracts

- Use case Pre/Post Conditions

Table 1: UC1.1 : loadDisplay()

Precondition: Internet connection and Chrome or Firefox browser	
Admin: User	System: Access Module
	0. Rendering and display service
3. Register, Sign-in, Upload scripts	1. Upload slot for soft-script.
Postcondition: no change.	

Figure 8: Access Module

Table 2: UC2.1 : getUser(userName:String)

Precondition: userName is a registered user	
Actor: Admin	System: User Module
	0. Display usermangement plage
1. Enter user details	3. Searches for requested person
2. Clicks Submit	4. Display Requested user
PostCondition: No change	

Figure 9: User management Module

Table 3: UC3.1 : sendNotification(userName)

Precondition: Mark uploaded Answer script.	
Admin: Marking Module	System: Notification Module
0. Marking module request Admin be notified about marking status	1. System receives notification request
	2. System finds the admin or user
	3. System sends notification to user
Postcondition: Specified user receives notification via email or SMS.	

Figure 10: Notification Module

Table 4: UC4.1 : uploadFiles(DataFile:ENC)

Precondition: Documents being uploaded are valid.	
Admin: User	System: Data and Security Module
0. Encrypt data files before storing.	1. Stream multiple file to server.
	2. Decrypt files
	3. Persist data for Validation
Postcondition: Data streamed successfully	

Figure 11: Data Streaming and Security Module

Table 5: UC5.1 : markScript(Memorandum:JSON ; AnswerSheet:JSON)

Precondition: Correct Memorandum and answer sheet upload.	
Admin: Access Module	System: Marking Module
0. Request user Marking	1. Slice the submitted memorandum
	2. Slice the submitted answer sheet
4. Receive feedback that marking is complete	3. System use Mathematical Processing Language compare the sliced mathematical steps.
Postcondition: Results persisted to database.	

Figure 12: Marking Module

Table 6: UC6.1 : getReport(userName:String)

Precondition: User exist within the system.	
Admin: Admin	System: Reporting Module
0. Request report about all the scripts ever marked under this user.	1. Extract data titled under this admin from the database
	2. Generate statistics about individual users (Students)
	3. Generate statistics about each mathematical script or question
Postcondition: Printout report to user via input–output module	

Figure 13: Reporting Module

Table 7: UC7.1 : submitDocuments(Memorandum:txt ; AnswerSheet:txt)

Precondition: user granted access i.e. user is registered	
Admin: Access Module	System: Input-Output Module
0. Upload memorandum and answer sheet	1. Validate if input documents are indeed scanned documents.
	2. Convert the document to digital data such as JSON.
	3. Validate if input is indeed a mathematical script
Postcondition: JSON object persisted to Marking Module and database simultaneous.	

Figure 14: I/O Module

5.4 Required functionality

The Caracal Automated Marking system will mostly rely on the the Optical Character Recognition (OCR), which has three most essential elements that are vital for the Caracal Automated Marking system. The elements are scanning, reading text and recognition. Therefore, as shown in Figure 1 below, the sequential or rather logical flow of the system will be input, scanning conversion, marking and output.

The system accept documents as input, with the integrated OCR technology the input documents have to be scanned and stored temporarily as images for marking. The images are then converted into words and characters that are recognizable. Next, the system marks the document, stores the marks and produce the output. The output is preferably the marked document as a portable document file. Therefore the overall functionality of the Caracal Automated Marking will be:

- Input - Three documents (i.e Question paper, Memo and the Answer sheet)
- Validation and Scanning
- Conversion
- Marking
- Output - Marked document as portable document file

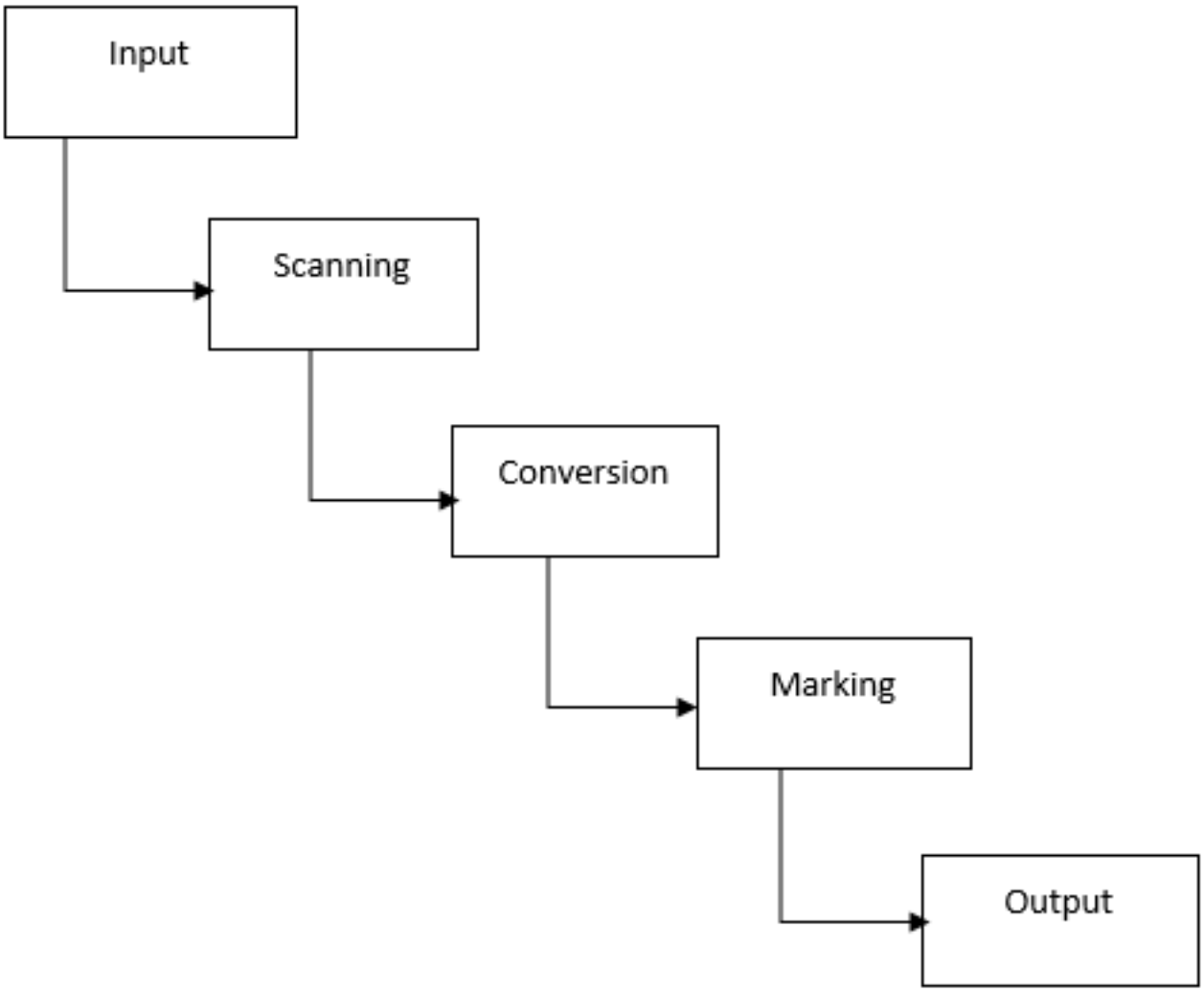

Figure 15: Caracal Automated Marking logical flow

5.5 Process specification

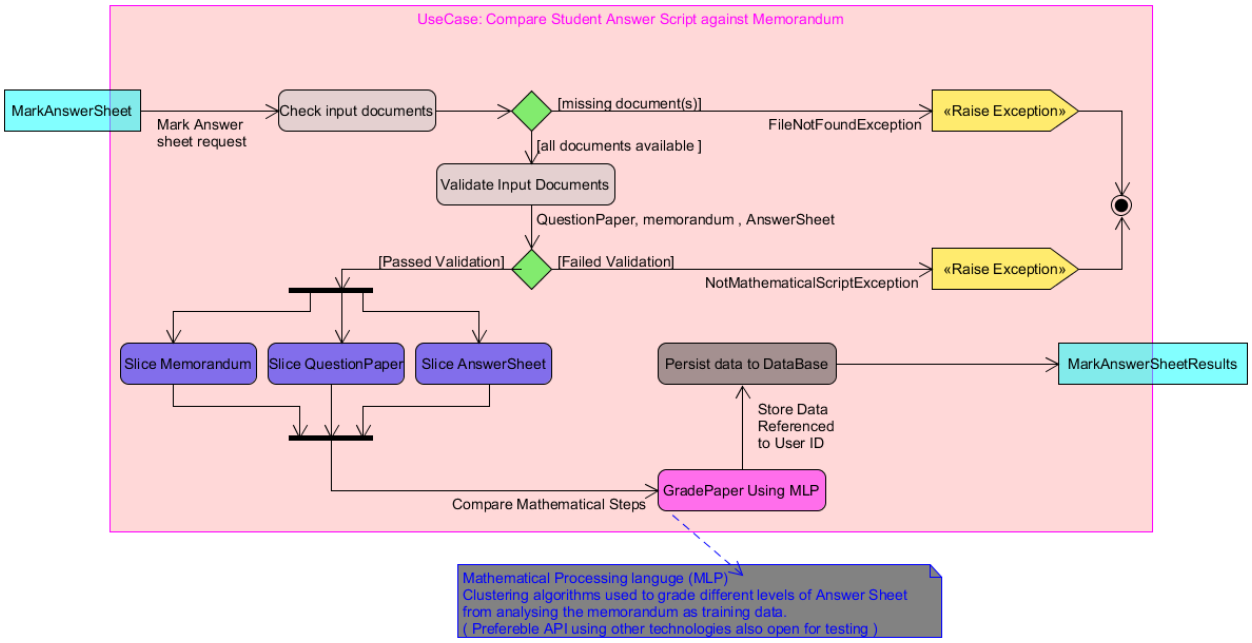

Figure 16: Caracal Automated Marking Process Specification Diagram

5.6 Domain Model

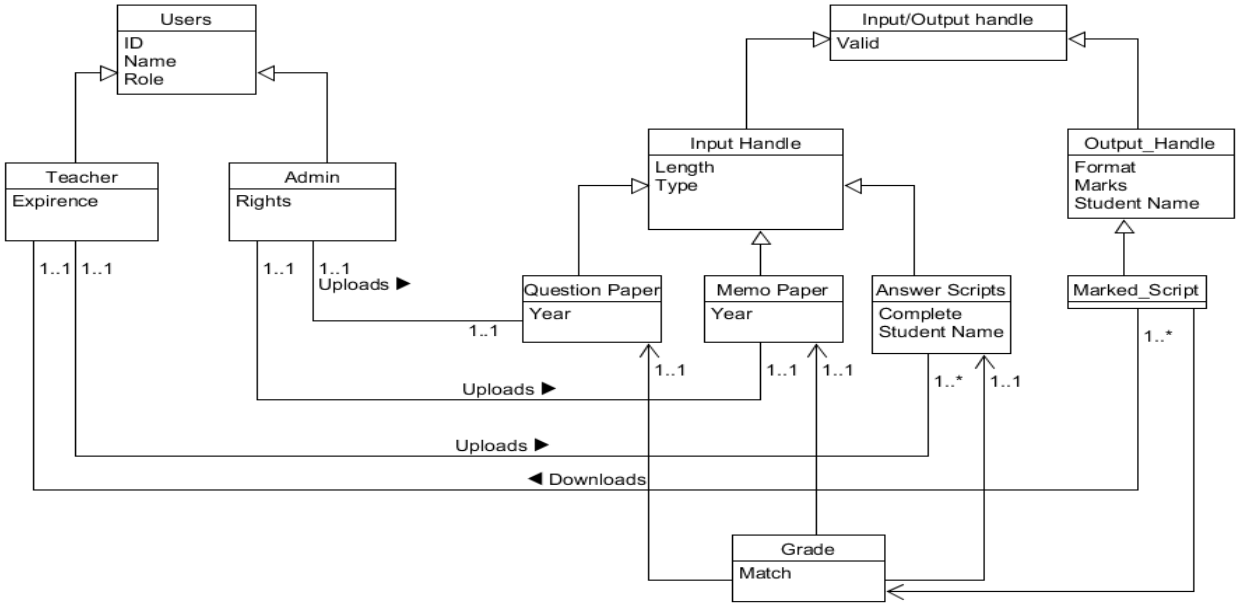

Figure 17: Caracal Automated Marking Domain Model

5.7 Traceability Matrix

5.7.1 Use Cases

- UC1.1: User Login
- UC1.2: Register User

- UC2.1: Add User
- UC2.2: Delete User
- UC2.3: Assign Role
- UC2.4: View User
- UC2.5: Edit User
- UC3.1: Notify Marking Status
- UC3.2: Notify Admin
- UC3.3: Signal System Error
- UC4.1: Stream Input Data
- UC4.2: Stream Output Data
- UC4.3: Encrypt
- UC4.4: Decrypt
- UC5.1: Compare Script with Memo
- UC5.2: Slice Question Paper
- UC5.3: Slice Memo
- UC5.4: Slice Answer Sheet
- UC5.5: Convert Sliced Question Paper
- UC5.6: Convert Sliced Memo
- UC5.7: Convert Sliced Answer Sheet
- UC5.8: Store Results
- UC6.1: Produce Student Performance Report
- UC6.2: Produce System Report
- UC6.3: Student Script Feedback
- UC7.1: Input Question Paper
- UC7.2: Input Memo
- UC7.3: Input Answer Sheet(s)
- UC7.4: Output Marks
- UC7.5: Output Marked Script

5.7.2 Requirements

- R01: Present Usable and Responsive Interface
- R02: Enable Login Functionality for Users
- R03: Allow New Users to Register
- R04: Allow for Document Input
- R05: Answer Sheet Marking
- R06: Notifications
- R07: Output: Marked Answer Sheet

Requirement	Priority	UC 1.1	UC 1.2	UC 2.1	UC 3.1	UC 5.1	UC 5.8	UC 7.1	UC 7.2	UC 7.3	UC 7.4	UC 7.5
R01	1	X	X									
R02	2	X		X				X	X	X		
R03	2	X	X	X								
R04	1					X		X	X	X		
R05	1					X	X				X	X
R06	3				X							
R07	2						X				X	X
UC Priority		1	2	2	1	2	1	1	1	1	1	2

Table 1: Traceability Matrix

6 Open Issues

The most critical issue is if the client is willing to pay for the technologies that we are going to use or should we develop everything from scratch. Another issue is the format of the question paper and memo, how is the question paper and memo going to be structured and is the structure going to be conbsistent or should our system adapt to any structure that it is presented. Will the system mark the final answer or should it also mark the steps followed to get the answer.

Should the system write the marks on the answer sheet or should it print a copy with the marks of the student. Will the scripts be entered one at the time then marked or will the system accept a load of scripts and then mark every sheet.