THIRD YEAR GROUP PROJECTS

PART A: – Year III Group Project Proposal

i. The year III Group systems project is equivalent to 1 course unit
ii. Each group shall be made up of a minimum of **two** and a maximum of **three** members.
iii. This project shall be scheduled in the teaching timetable and shall be carried out and completed in the two semesters of the academic year: *the first semester will involve the writing of the project proposal and System Analysis and Design while the second semester will involve the implementation of the actual project, testing and a project report based on the proposal presented in the first semester*.
iv. The Year III student will be required to present their proposed idea of the project to be carried out, to the department's project coordinator for approval not later than the end of the second week of the first semester.
v. The proposed project ideas will undergo vetting by the departmental teaching staff, under the overall coordination of the Departmental Projects coordinator; the focus of the vetting will be **viability**, **scope** and **appropriateness**. The results of the successful and approved project ideas shall be communicated to the students at the end of the third week of the first semester.
vi. Students whose ideas have been approved will, under the guidance of their assigned supervisor begin their project proposals writing.
vii. A student shall be required to meet/consult the supervisor a minimum of nine (**9**) times in the semester when the project is carried out. The supervisor shall keep a "***Project Log book*** ". The logbook shall:
    - Captures the days and times such meetings occurred (as per the timetable).
    - Capture the comments of the supervisor at various stages of the project.

- Be signed by both the student and the supervisor on each of the days when such meetings took place.
- Be submitted as part of the bound and signed project documentation (in the appendix section).

viii. The final project proposals shall be submitted at the end of the twelfth (12) week of the semester (before project presentation date – as per the university exam timetable)

ix. Final project proposals submitted for examination in the department shall be duly signed by both the student(s) and the respective project supervisor.

x. The student SHALL submit two signed spiral bound project proposal documents to the supervisor, who will in turn sign and submit to the department's project coordinator by the end of week twelfth (12) of the first semester of the third year of study. Proposal submissions done after this period will only be accepted if there are compelling reasons that justified the delay and which must be accompanied with convincing proof/ evidence.

xi. Any project proposal submitted after the twelfth (12) week of the first semester of the third year of study shall not be accepted and consequently the affected student(s) shall be required to formally request to re-take the course unit in the following academic year.

xii. The submitted project proposal shall be examined by the supervisor, who will award a maximum of 50% using assessment tool in **Appendix Form YEAR III PROPOSAL SMG.**

xiii. A panel consisting of **NOT LESS THAN THREE** faculty members, excluding the project supervisor of the group being examined, shall examine the proposal, based on the laid down guidelines (**see Appendix Form YEAR III PROPOSAL PMG**) and independently award a score of up to 50%.

xiv. As part of the project examination, each group member will be required to clearly quantify and demonstrate their contribution to the final, submitted & examinable project proposal.

xv. The Year III project proposal shall comprise of the following chapters

i. Chapter 1 Introduction –see the format

ii. Chapter 2 Development methodology

iii. Chapter 3 System Analysis

iv. Chapter 4 System Design

Part B: – Year III Group Software Project Implementation

i. The Software Project implementation is equivalent to 1 course Unit.
ii. The student will be expected to develop the project prototype from the beginning of the second semester of the third year, based on the Software Project proposal presented in the previous semester.
iii. The students shall be expected to make contact and consult the assigned project supervisor a minimum of nine (9) times during the semester in which the project is carried out.
iv. The project supervisor shall keep a "*project log book* " that records the days and times such meeting occurred and submit the same to the project coordinator at the end of the twelfth (12) week of the second semester of the third year (*before project presentation date – as per the university examination timetable*)

Examination of Year III Software Project Implementation

i. The student SHALL submit two signed spiral bound project implementation reports to the supervisor, who will in turn sign and submit to the Departmental projects coordinator.
ii. Final project proposal documents **SHALL** be submitted at least 5 days before the scheduled day of oral examination. No project proposal document shall be submitted on the examination day.
iii. The student shall take responsibility in case of failure to submit the project proposal as laid down rules.
iv. The oral examination sessions of the software project proposal shall be organized and coordinated by the departmental projects' coordinator, who, working together with the

departmental examination coordinator & the Chair of Department shall specify; the date, time, venue and constitute the examining panels.

v. The date of the oral presentation shall fall within the specified examination period of semester within which the project was taken.

vi. The submitted project shall be examined by the supervisor, who will award a maximum of 50% using assessment tool in **Appendix Form YEAR III PROJECT SMG.**

vii. A panel consisting of **NOT LESS THAN THREE** faculty members, excluding the project supervisor of the group being examined, shall examine the project, based on the laid down guidelines (**see Appendix Form YEAR III PROJECT PMG**) and independently award a score of up to 50%.

viii. Each member of the examining panel shall sign their respective scoresheets for each group examined.

ix. Upon completion of the oral examination of the project, the student(s) shall;

a) Sign the examination attendance register.

b) Surrender the following items to examining panel, one appropriately labeled (name, registration number, year of presentation) CD/DVDs containing the following:

- A soft copy of the project report.
- The executable or installable version of the prototype system.
- The power point file used in the project presentation.

x. At the end of the oral examination, **each panel member** shall append their signatures and date on the examination attendance register, for the students examined by the panel.

xi. A final report shall be submitted as per the report guide provided below

A student shall ONLY be allowed to attempt the software project proposal a maximum of three (3) times. A student(s) who fails to successfully carry out the software project proposal within the allowed number of attempts, shall on the recommendation of the board of examiners of the department of Information Technology and approval by the School Board (School of Computing and Informatics) and the University Senate, be discontinued from the programme.

MASENO UNIVERSITY

COMPUTER SCIENCE DEPARTMENT

YEAR III PROJECT REPORT

STRUCTURE AND FORMAT

A) Title page/Cover page. The title page should have:-

i. Project Title

- A concise statement of the main topic and should identify the variables.
- Should be a reflection of the contents of the document.
- Fully explanatory when standing alone.
- Should not contain redundancies such as 'a study of…..or 'an investigation of……
- Abbreviations should not appear in the title.
- Scientific names should be in italics.
- Should contain 12 to 15 words.
- A good title should display understanding of the research problem.

ii. Author's name and affiliation

- Avoid use of words like 'By'….. 'from"…..
- Preferred order of names- start with 1st, middle followed by last name.
- Full names should be used, initials should be avoided.
- Titles like Dr. should not appear in the names.

Affiliation should be well illustrated i.e. '*A project submitted to the Department of Computer Science in the School of computing and informatics…………. in partial fulfillment of the requirements for the award of the degree of …….. Maseno University*

The year should follow at the bottom of the caption.

Note
The title, author and affiliation should all appear on one page (page 1) and centered.

B) Declaration – Should include both the candidate's and the supervisor's declaration and duly signed.

This project is my original work and has not been presented for a degree in any other University

…………………. ………………
Signature Date
Names Reg number
This project has been submitted for examination with my approval as University Supervisor

……………… ……………….
Signature Date
Names
Maseno,Kenya

C) Acknowledgements – appreciate those that directly contributed to the success of your project, e.g. project sponsor (if you had any), your lecturers, supervisors, technicians, fellow students, e.t.c.

D) Abstract- This is a brief statement of the problem, objectives of the study, target population, sampling technique and sample size, instruments, data collection, data processing and analysis, key findings and major recommendations. It is a summary of the project as a whole

- Should not exceed 400 words.

E) Table of contents – The rubric should be in title case and single spaced.
- The chapter titles should be in caps and bold.
- The subheadings should follow each chapter title and should be in title case.
- Subheading of rows should be – Chapters & Pages indicated once at the top of each column e.g.,

F) List of Tables

G) List of Figures

H) Acronyms and Abbreviations

I) Definition of terms

- Define terms in the text that are not common.

Note:

- Paragraphing should be consistent. Either leave space or indent between paragraphs.
- Spacing and indenting should not be used together.
- One sentence paragraphs are unacceptable.
- A paragraph should have a minimum of five sentences.

Table of contents should be followed by:

- List of figures/ tables- Should be labeled as per the chapters in which they are found e.g. the first figure in chapter one should be labeled as Figure 1.1
    - Paginate using roman numbers starting with the declaration page.

CHAPTER 1: INTRODUCTION

This chapter should include the following;

1.1 Background Information

Must indicate the background operation issues of the firm they want to create a system for, what do they engage in, what are some of the problems they experience or the industry in which the expected application solution is supposed to operate in.
The student should demonstrate good knowledge of the industry in focus.

1.2 Statement of the problem (What is problem that your project is addressing?)

Must indicate exactly the problem the project is attempting to solve.

- Indicate why and how it is a problem. Give information to support this e.g. by use of statistics or evidence. This should be derived from background information to illustrate connectivity.

-It should show the magnitude and effects of the problem
-Must indicate exactly what the research problem is.
-This should be tied to your title/research area.
-This should have more than the physical problem

1.3 Proposed Solution (what is your proposed solution?)

-State your proposed solution. This research sought to …… not the prototype.
-The research component should be clearly captured within the solution
-The how (e.g. the prototype of the system) should not be mentioned.
-Students need to compare recent models-globally and regionally to come up with the best solution without giving an old technology/going backwards.

1.4 Objectives/Aims (what are aims and objectives of your project?)

-One general objective which should be in line with the title.

- Specific objectives- have to be in line with the variables the candidates hypothesize to influence the phenomenon being investigated.
- Should be related to the general objective.
- Should not be questions in the questionnaire.

-These should be the student’s objectives-not the client’s. It should be measurable within the 8 months of the projects
-They must cover research area, an implementation one and at least one to evaluate if they have achieved what they set out to do.
-They should be SMART (Specific, Measurable, Achievable, Reliable and Testable)
-They must be research project objectives
-They should be simple statements-not a paragraph

1.5 Research Questions (what are the research questions?)

- They should be in line with the specific objectives and equal in number.
- Have to be numbered (1, 2, 3…..) and should be questions and not statements.
- Gives us a breakdown of the areas that would be covered.

-They should be broad areas that will bring value

-They should not have short answers that can be answered in a few words

-Research questions should be limited to research areas-NOT the client's problem or proposed prototype

1.6 Justification (justify the need to undertake the proposed project)

- Should illustrate why the researcher is conducting the research and whom it shall benefit.
- Should indicate why the proposed solution will solve the client's problem.
- What is it contributing to that research area/Problem area

CHAPTER 2: DEVELOPMENT METHODOLOGY

This chapter should indicate;

2.1 Introduction: Introduction to the chapter detailing what the chapter covers.
2.2 Description of the methods for Systems Analysis.
2.3 Feasibility Study: A statement on the project feasibility should be presented.
2.4 Description of the methods for Requirements elicitation
2.5 Description of the methods for Data and Analysis
2.6 Description of the methods for Systems Coding and Testing

CHAPTER 3: SYSTEM ANALYSIS

This chapter should indicate;

3.1 Introduction: Introduction to the chapter detailing what the chapter covers.
3.2 Describe the application Systems Development Methodologies.
3.3 Feasibility Study: A statement on the project feasibility should be presented.
3.4 Requirements elicitation:

Data Collection:

-Describe the data collection tool, its preparation and how it was administered and attach it as an appendix. Use tools like Interviews, Observation, questionnaires, etc. This tool should be approved by the Supervisor before it is administered.

-The data collected must be relevant to the problem and based on the objectives of the research.

-It should be useful in deducing system requirements

3.5 Data and System Analysis:

Analyze the data collected using Statistical tools (Excel, SPSS) and represent the findings using any analytical tool (pie charts, bar graphs, line graphs, etc)

3.6 System Specification: Outline the systems requirements

In this section, you need to describe in clear, precise and non-ambiguous terms what the application will do. Let me remind you that an important objective of analyzing a problem is to determine two pieces of information: the information that must be supplied to the application, and the information that the application should produce to solve the problem. This information is what is referred to as specifications. Depending on the software process model that you use, problem analysis and specification document may contains models such as class diagrams, use cased diagrams, data flow diagrams, e.t.c.

CHAPTER 4: SYSTEM DESIGN

4.1 Specify Logical design and Physical design

-Logical designs we have tools like Rich pictures, Wire frames-Abstract representation of data flows, inputs and outputs of the system

-Physical design (OOSAD (Use specific standards-UML 2.x) and SSADAM diagram)-The actual inputs and outputs processes of the system (User Interface, Data Design, Process Designs).

4.2 System Architecture

-should capture include how you have designed the system the Client, server and middle tier

-Client/Server architecture – application should be structured like a web application with a defined client – the browser- and database plus script servers

-N-tier design – application should be divided into well-defined tiers like interface, business logic, data back end

-Other – should be explained by student

4.3 The technique should match the requirements analysis technique and will have direct implications on the kinds of diagrams required:

-Object oriented design

-Process oriented and Data Oriented design

4.4 Note that the end result should be a normalized database

4.5 Suggested minimum requirements

-Explanation of scope of the system – context diagram may be displayed and explained

-Communication of major processes – Detailed/Partitioned Data Flow diagram, should not be a copy of an unpartitioned DFD or a level 0 DFD from requirements analysis phase

-Data design – entity Relationship diagram and explanation of major entities

-Interface design – mock ups of forms

-Identification of components, subsystems

4.6 Other diagrams that may be present

-Use cases

-Sequence diagrams

-Class diagrams

-The design must match the functional requirements, stated requirements should translate to some tables in the data design, interface forms.

-There should be a statement on overall system architecture

4.7 Design phase report contents

-Summary of overall system architecture

-A partitioned DFD showing various flows

-Details of entities identified from the requirements and their attributes

-Data design - Entity relationship diagram
-Interface design – mockups of various forms

CHAPTER 5: SYSTEM CODE GENERATION AND TESTING

5.1 SYSTEM CODE GENERATION-Actual coding

5.2 TESTING- Should describe all tests subjected to your system and the results

OVERVIEW OF THE CHAPTER

-The system being developed must be based on the system specifications defined in the previous

-Must use state of the art/latest technology

Main phases are Code generation, integration, testing and documentation

-**Code Generation**: Has to be done according to the tools and requirements stated earlier

-**Software Integration**: All components that were developed must be integrated- everything provisioned at the design and requirements specifications

Testing

Objectives of testing

- Build quality in the system developed
- Demonstrate the working capabilities of your system
- Assess progress and suitability of the system

What is testing?

Testing entails subjecting the system developed by the students to set of valid inputs to validate the outputs of the system against pre meditated set of outputs.

-Identify, design then implement the testing.

-Clearly state the Validation (e.g.Usability, Blackbox, Acceptance) and verification tests (e.g.Smoke testing, Stress test, White box) conducted.

TESTING SCOPES THAT CAN BE ADOPTED

Gui Testing: Attempt to cover all the functionality of the system and fully exercise the GUI itself. A GUI has many operations that need to be tested. The objective at this point will be test sequencing of events and relevance in terms of colors and appearance of the GUI interface

Usability testing is a technique used in user-centered interaction design to evaluate a product by testing it on users. This can be seen as an irreplaceable usability practice, since it gives direct input on how real users use the system.

The students should be able to test the system with their friends and peers and report on how he peers felt about the system. A form or documentary evidence should accompany the report to show this.

Performance testing is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage.

The system for the students should actually be subjected in stressful conditions such as multiple entries of data, quick succession of activities to determine its performance level. This should then be reported back to the group for assessment by the supervisors.

White box testing: tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user. The student should submit the flow charts so that set of test cases are subjected to the flow and response of the system determined from the flow chart. This would then help compare the internal performance against expected general performance of the system.

Load testing: Load testing is a generic term covering Performance Testing and Stress Testing. A process of testing the behavior of the Software by applying maximum load in terms of Software accessing and manipulating large input data. It can be done at both normal and peak load conditions. This type of testing identifies the maximum capacity of Software and its behavior at peak time.

Regression Testing: Similar in scope to a functional test, a regression test allows a consistent, repeatable validation of each new release of a product or Web site. Such testing ensures reported product defects have been corrected for each new release and that no new

quality problems were introduced in the maintenance process. Though regression testing can be performed manually an automated test suite is often used to reduce the time and resources needed to perform the required testing.
Whenever a change in a software application is made it is quite possible that other areas within the application have been affected by this change. To verify that a fixed bug hasn't resulted in another functionality or business rule violation is Regression testing. The intent of Regression testing is to ensure that a change, such as a bug fix did not result in another fault being uncovered in the application.

Stress testing: Testing conducted to evaluate a system or component at or beyond the limits of its specified requirements to determine the load under which it fails and how. A graceful degradation under load leading to non-catastrophic failure is the desired result. Often Stress Testing is performed using the same process as Performance Testing but employing a very high level of simulated load.
This testing type includes the testing of Software behavior under abnormal conditions. Taking away the resources, applying load beyond the actual load limit is Stress testing.

Smoke testing: A quick-and-dirty test that the major functions of a piece of software work without bothering with finer details. Originated in the hardware testing practice of turning on a new piece of hardware for the first time and considering it a success if it does not catch on fire.

Unit Testing: Functional and reliability testing in an Engineering environment. Producing tests for the behavior of components of a product to ensure their correct behavior prior to system integration.
This type of testing is performed by the developers before the setup is handed over to the testing team to formally execute the test cases. Unit testing is performed by the respective developers on the individual units of source code assigned areas. The developers use test data that is separate from the test data of the quality assurance team.

Acceptance Testing: Testing to verify a product meets customer specified requirements. A customer usually does this type of testing on a product that is developed externally.
This is arguably the most importance type of testing as it is conducted by the Quality Assurance Team who will gauge whether the application meets the intended specifications and satisfies the clients requirements. The QA team will have a set of pre written scenarios and Test Cases that will be used to test the application.
Black Box Testing: Testing without knowledge of the internal workings of the item being tested. Tests are usually functional.
Functional Testing: Validating an application or Web site conforms to its specifications and correctly performs all its required functions. This entails a series of tests which perform a feature by feature validation of behavior, using a wide range of normal and erroneous input data. This can involve testing of the product's user interface, APIs, database management, security, installation, networking, etc Functional testing can be performed on an automated or manual basis using black box or white box methodologies.
This is a type of black box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to conform to the functionality it was intended for. Functional Testing of the software is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.
Integration Testing: The testing of combined parts of an application to determine if they function correctly together is Integration testing.
Testing in which modules are combined and tested as a group. Modules are typically code modules, individual applications, client and server applications on a network, etc. Integration Testing follows unit testing and precedes system testing.
There are two methods of doing Integration Testing Bottom-up Integration testing and **Top down Integration testing**.
Bottom-up integration: this testing begins with unit testing, followed by tests of progressively higher-level combinations of units called modules or builds.
Top-Down integration: in this testing, the highest-level modules are tested first and progressively lower-level modules are tested after that.

Security Testing: Security testing involves the testing of Software in order to identify any flaws ad gaps from security and vulnerability point of view.

Portability/Compatibility Testing: Testing to ensure compatibility of an application or Web site with different browsers, OSs, and hardware platforms. Compatibility testing can be performed manually or can be driven by an automated functional or regression test suite. Portability testing includes the testing of Software with intend that it should be re-useable and can be moved from another Software as well.
Conformance Testing: Verifying implementation conformance to industry standards. Producing tests for the behavior of an implementation to be sure it provides the portability, interoperability, and/or compatibility a standard defines.

Identification of Test case generation
A test case is a set of conditions or variables under which the student determines if the application is working. It includes Test case ID, Condition to be tested, Expected results and actual results. Include a column that shows if the test passed or failed.
The students should generate a minimum of 20 test cases to enable the examiners assess the full functionality of the system. The test cases should cover each unit or module of the system. The assumption in this scenario is that the students will have five modules and so 4 test cases will be applied to each module to determine its suitability.
A table should be generated to indicate the set of inputs and the valid outputs to be produced from the system. These inputs will help form test cases and outputs as assessment criteria for suitability of the system.

Software testing tools in the market:
-Selenum
-HP Quick Test Professional
-IBM Rational Functional Tester
-Silk Test
-TestComplete
-Testing Anywhere

-WinRunner

-LoadRunner

-Visual Studio Test Professional

-WATIR

CHAPTER 6: CONCLUSIONS AND RECOMMENDATIONS

- Did you solve your client's problem and to what extent?

- Should be derived from the conclusions

REFERENCES

- List all the materials e.g. books, journals, conference papers, internet materials cited in the report.
- Use either Harvard referencing style or APA referencing style
- Consult a minimum of 10 peer reviewed references, i.e. journals. Students are encouraged to adhere to the age of the references not be more than 5 years except in a few exceptions

APPENDICES

Should contain materials that are peripheral to the body of the report, e,g.

- Test cases and test data
- Users' manual
- Instruments e.g. a sample questionnaire
- Budget
- Work plan
- Source Code (Not all source code as they can consume many pages. Save part of the source in a CD). Submit that CD together with the project report.

MASENO UNIVERSITY

COMPUTER SCIENCE DEPARTMENT

Project Student Log.

Student's Names………………………………………….. Registration No………………………

Unit code. ………………………Unit title……………... ……………………………………...

Project Title………………………………………………………………..……………………

Supervisor …………………………………………………………………………………….

This form is to be filled by the supervisor on each consultation date and duly signed. It is to be kept by the student, and it should be submitted to the project coordinator at the end of the project period. The student should consult the supervisor at least once a week.

Date	Comment(s)	Sign Supervisor	Sign Student

