

BSIT



**POLYTECHNIC UNIVERSITY OF THE
PHILIPPINES**

BSIT Capstone Project Guidelines

BSIT Capstone Project Guidelines



PUP QC Branch

Table of Contents

EXECUTIVE SUMMARY	4
Part 1. INTRODUCTION	6
1.1 CAPSTONE PROJECT OVERVIEW	6
1.2 CAPSTONE PROJECT SCOPE	7
1.3 CAPSTONE PROJECT OBJECTIVES	7
1.4 TIME TO BE SPENT ON THE CAPSTONE PROJECT	8
1.5 SELECTING A CAPSTONE PROJECT	8
1.6 CAPSTONE PROJECT ADVISER/PANEL QUALIFICATIONS	9
1.7 CAPSTONE PROJECT OWNERSHIP	11
1.8 GROUP COMPOSITION	12
1.9 REST OF THE DOCUMENT	13
Part 2 - CAPSTONE PROJECT SUBMISSIONS	14
2.1 CAPSTONE PROJECT PROPOSAL	14
2.2 CAPSTONE PROJECT LOGBOOK	14
2.3 CAPSTONE PROJECT	15
2.4 USB / CLOUD STORAGE	15
2.5 FINAL CAPSTONE PROJECT	15
PART 3 - CAPSTONE PROJECT TIMETABLE	16
3.1 CAPSTONE PROJECT SCHEDULE	16
3.2 SUBMISSIONS	18
3.3 CAPSTONE PROJECT FEES	19
3.4 CAPSTONE PROJECT GRADE	19
3.5 CAPSTONE PROJECT SEMINAR/ORIENTATION	19
PART 4 - REPORT WRITING	19
4.1 FORMAT OF THE CAPSTONE PROJECT	19
4.1.1 Other Format Requirements	20
4.2 CONTENTS OF THE CAPSTONE PROJECT	22
4.3 PREFACE MATERIAL	23
4.3.1 Title Page	23
4.3.2 Declaration	23
4.3.3 Abstract	24
4.3.4 Acknowledgements	25
4.3.5 Table of Contents	25
4.3.6 List of Acronyms	25
4.4 MAIN PARTS	26
4.4.1 Part 1 – Introduction	32
4.4.2 Part 2 – Literature Review	36
4.4.3 Part 3 – Methodology	42
4.4.4 Part 4 – Requirements Analysis	47
4.4.5 Part 5 – Business Process Architecture	51
4.4.6 Part 6 – Application Architecture	58

Capstone Project Guidelines for PUP BSIT Students

4.4.7	Part 7 - Data Architecture	64
4.4.8	Part 8 - Technology Architecture	69
4.4.9	Part 9 - Development Process	73
4.4.10	Part 10 - Implementation	76
4.4.11	Part 11 - Testing and Quality Assurance	80
4.4.12	Part 12 - Results and Evaluation	87
4.4.13	Part 13 - Conclusion	91
4.4.14	Part 14 - References	91
4.4.15	Part 15 - Appendices	93
4.4.16	Part 16 - Technical Documentation	96
PART 5 - CAPSTONE PROJECT EVALUATION		98
5.1	DEFENSE OF THE CAPSTONE PROJECT	99
5.1.1	Authorship/Mock and Defense	99
5.1.1.1	Authorship/Mock	99
5.2	CAPSTONE PROJECT ASSESSMENT	104
PART 6 - PITFALLS		107
Appendices		

CHED Directive

The Bachelor of Science in Information Technology (BSIT) program prepares students to be IT professionals, be well versed on application software installation, operation, development, maintenance and administration, and familiar with hardware installation, operation, and maintenance.

CMO #25 S 2015 (*Pls see Appendices*)

Bachelor of Science in Information Technology (BSIT)

The BS Information Technology program includes the study of the utilization of both hardware and software technologies involving planning, installing, customizing, operating, managing, and administering, and maintaining information technology infrastructure that provides computing solutions to address the needs of an organization.

The program prepares graduates to address various user needs involving the selection, development, application, integration, and management of computing technologies within an organization.

Capstone Project

BS Information Technology students must complete a capstone project such as a software/system development with emphasis on the IT infrastructure, or an IT Management project.

It is expressly understood that Computing Thesis and Capstone Projects need not require surveys, statistics and descriptive methods unless appropriate.

A **Capstone Project** is an undertaking appropriate to a professional field. It should significantly address an existing problem or need.

Capstone Project Guidelines for PUP BSIT Students

An Information Technology Capstone Project focuses on the infrastructure, application, or processes involved in implementing a Computing solution to a problem.

Scope of the Capstone Project

The Capstone Project should integrate the different courses, knowledge, and competencies learned in the curriculum. Students are encouraged to produce innovative results, generate new knowledge or theories, or explore new frontiers of knowledge or application areas.

Attached is the CMO #25 S. 2015 for your reference

EXECUTIVE SUMMARY

This document is to guide the group/individuals/individual on the final year's project. It first introduces what is expected from a project and highlights its objectives. It also describes on how to select a project and an adviser.

The various submissions from the project proposal to the final study have to be submitted to meet the project requirements. These are explained in detail along with the deadlines. Information is also available on what happens when the deadlines are not met. In a nutshell, Figures 1 and 2 show the major activities that should be carried out by the project team/individual during Capstone 1 and 2.

A detailed description of the project structure and content is provided. It identifies the areas that have to be addressed at each component of the study.

Capstone Project Guidelines for PUP BSIT Students

The method adopted to assess the project is described in detail. Finally the pitfalls one should be aware of are also presented.

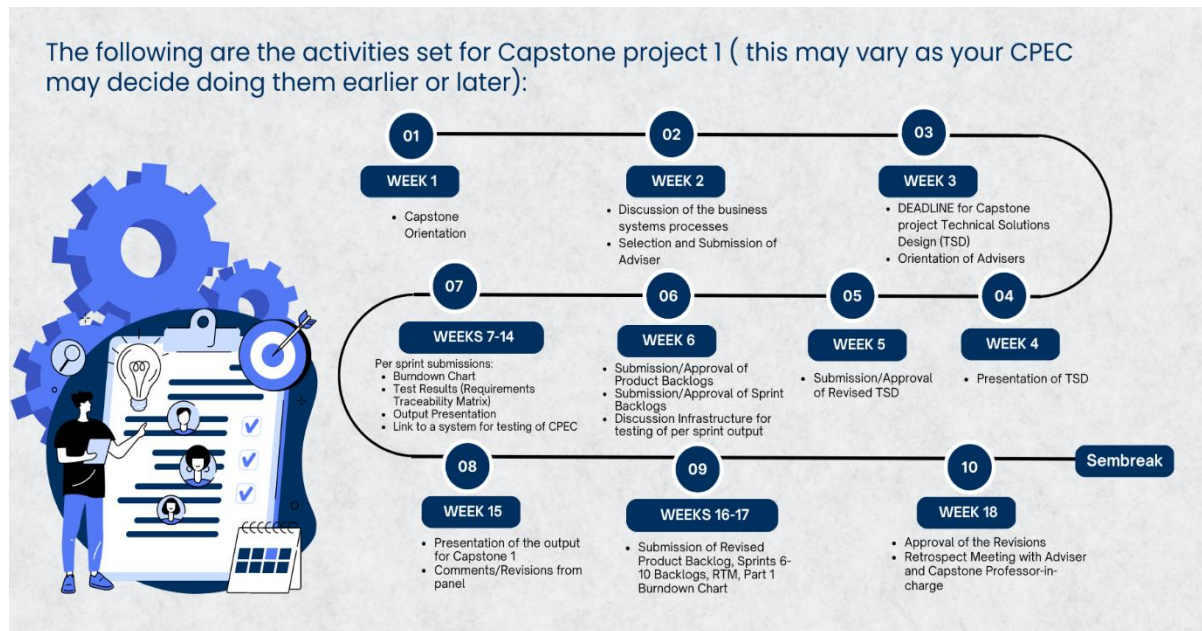


Figure 1: Capstone 1 Flow and Timeline of Activities

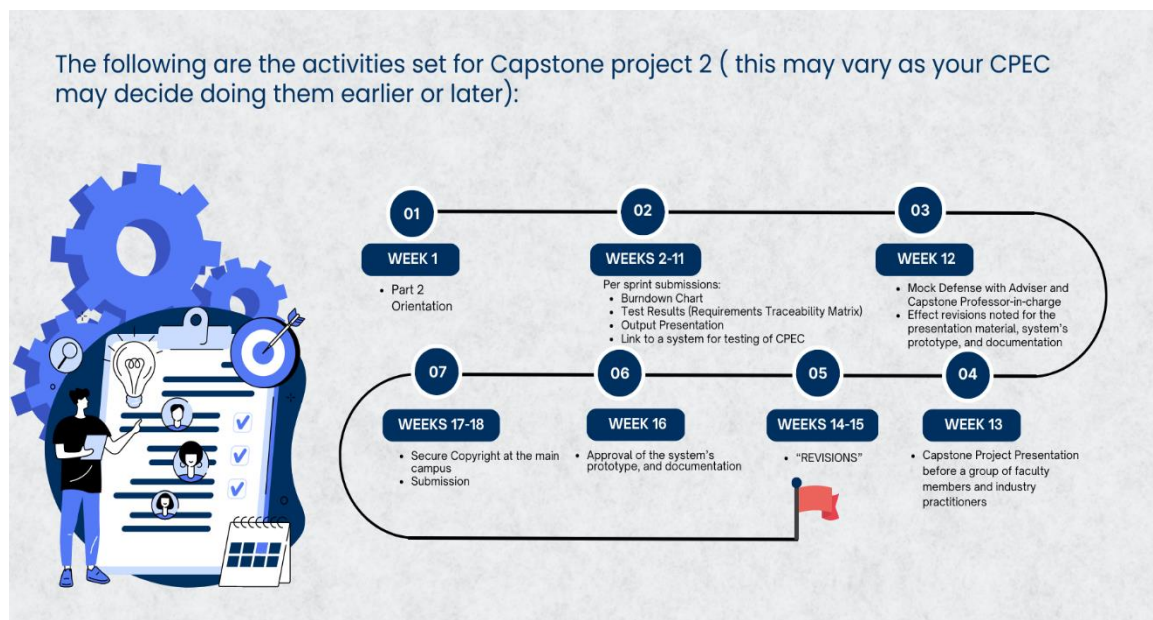


Figure 2: Capstone 2 Flow and Timeline of Activities

Part 1. INTRODUCTION

This part introduces what is expected from a final year capstone project and highlights its objectives. It also describes on how to select a capstone project and an adviser. A brief summary of the other parts is also given.

This document is to guide the group/individual on the fourth year capstone project and preparing the capstone project. Hence this document is presented to look like a mini capstone project. The cover page, abstract, acknowledgement, content, acronyms, chapters, references and appendices are examples to reflect the capstone project's appearances. During the academic year, any updates and amendments to this document will be published or will be given by your assigned Capstone Project professor.

1.1 CAPSTONE PROJECT OVERVIEW

The individual capstone project is by far the most important single piece of work in the BSIT degree program. It provides the opportunity for a group/individual to demonstrate independence and originality, to plan and organize a large capstone project over a long period, and to put into practice some of the techniques that have been taught throughout the course. The capstone project also aims to assess the group/individual's ability to communicate group's/individual's ideas and work. Whatever is your level of academic achievement so far, you can show your individuality and inspiration in this capstone project. It should be the most satisfying piece of work in your degree. It is equivalent to two modules in the BSIT degree program and is an extended piece of individual work, occupying group/individual's time from the end of the third year through to the end of the fourth year covering over two semesters of work.

A group/individual will select an adviser and a capstone project through the guidance of the Capstone Project Evaluation Committee (CPEC). A Capstone project is selected from

Capstone Project Guidelines for PUP BSIT Students

a workplace or an organization. Group/individual will have to develop a prototype and demonstrate that the Requirements are met through the system. Group/individual will have regular meetings with the Adviser to discuss capstone project work and produce a formal capstone project (report) in a structured way along the suggested guidelines. It should demonstrate that the relevant work has been carried out at a workplace under proper supervision.

1.2 CAPSTONE PROJECT SCOPE

Although the capstone project is done for a client (depending on the agreement with the CPEC and the adviser) a group/individual should remember that the purpose of the capstone project is to fulfill an examination requirement of the BSIT degree program. Hence satisfying a pilot client does not guarantee that the capstone project is successful as the client's expectation could be well below the expectation of the Capstone project Examination Committee of your school (CPEC). Also some clients may expect more than what is expected by the CPEC and hence the group/individual may fail to fulfill all the requirements of the CPEC within the allocated time.

1.3 CAPSTONE PROJECT OBJECTIVES

The capstone project encourages and rewards individual inventiveness and application of effort. The capstone project will develop group/individual's ability to:

- Construct a capstone project from initial ideas, via a thorough analysis of the problem;
- Plan, schedule, monitor and control own work;
- Work independently;
- Defend ideas in discussions and presentations;

- Use references - libraries and other information sources;
- Apply theories, tools and techniques from taught courses;
- Demonstrate the solution to the problem through developed software; and;
- Learn formal report writing.

1.4 TIME TO BE SPENT ON THE CAPSTONE PROJECT

The capstone project is equivalent to two modules. The group/individual is expected to spend, on an average, at least several hours per week amounting to a minimum total of 98 hours, excluding the time taken for report writing and preparation of presentation material. Effective time management is the group/individual's responsibility. Devote a regular time slot for the capstone project.

Work consistently throughout the year will help. Always keep track on the capstone project submission deadlines and plan on what has to be done to meet them.

1.5 SELECTING A CAPSTONE PROJECT

It is the responsibility of the group/individual or the teacher to identify a suitable capstone project. The capstone project should comprise a substantial amount of individual work to satisfy the CPEC that the capstone project objectives have been met as well as the time spent on the capstone project is justified.

The group/individual will work on a topic of interest which may have originated from group's/individual's work place or may be based on an organization's requirement or may be based on a Group/individual's idea that an organization would like to try out. It is important when choosing a capstone project to understand the way it will be assessed. A

Capstone Project Guidelines for PUP BSIT Students

good first-class capstone project involves a combination of sound background research, a solid implementation with substantial system functionality and a thorough evaluation of the capstone project's output in both absolute and relative terms. A crucial component (the most important single deliverable) is the capstone project which should be well-structured and well presented, detailing the capstone project's background, objectives and achievements. The very best capstone projects invariably cover some new ground, e.g. by developing a complex application which does not yet exist, or by enhancing some existing application or method to improve its functionality, performance etc.

Capstone project should involve the main activities associated with the design and implementation of a software engineering system: requirement analysis, specification, design, implementation, testing, evaluation, documentation and maintenance. At the end of the capstone project the group/individual must be able to certify that he has met all the requirements that are mentioned by the client and a client certification letter should be part of the capstone project.

1.6 CAPSTONE PROJECT ADVISER/PANEL QUALIFICATIONS

A group/individual should have a capstone project adviser. The adviser should be able to guide the group/individual throughout the capstone project.

The adviser must have completed a computing project successfully beyond the bachelor's degree project. As much as possible, the adviser should be a full-time faculty member of the HEI. Otherwise, a full time faculty co-adviser is required.

Advisers and Panel Members should have a degree in a Computing or allied programs, or must be domain experts in the area of study. At least one of the panel members must have a master's degree in Computing (preferably in the same field as the thesis or project)

Capstone Project Guidelines for PUP BSIT Students

or allied program. For IT at least one of the panel members should preferably have industry experience.

The adviser must be able to guide the students throughout the whole project life cycle, including the thesis/capstone project defense and possible project deployment.

Faculty advisers should preferably handle most five projects at one time, and in no case should exceed ten (10) projects. Panel members should preferably be limited to at most ten (10) projects and in no case should this exceed twenty projects in one semester, counting all projects in all HEIs.

In case of the participation of an external client, then the organization for which the project is intended should be represented as much as possible.

The group/individual should ensure that the adviser will not be away for very long periods and he is willing to spend the expected time with you. The adviser should also be able to go through all your capstone project documentations and provide feedback. The chosen adviser sometimes may not be familiar with the client domain and may not be able to guide you in that aspect. In such cases you are advised to have a second adviser.

This person need not be familiar with ICT and preferably from the client organization. Note that members of the CPEC are prohibited from being capstone project advisers. The group/individual should obtain the consent of the adviser to supervise the capstone project and group's/individual's consent should be indicated in the adviser consent form. Any change of capstone project adviser should be notified by resubmitting an adviser consent form.

It is a formal requirement that the group/individual regularly meets the capstone project adviser during the duration of the capstone project. The group/individual should work

Capstone Project Guidelines for PUP BSIT Students

independently but report the progress and seek guidance from the adviser to ensure the correctness of the approaches. The group/individual should agree on a timetable with information about methods of contact with the adviser at the start of the capstone project. Typically, a group/individual should expect to meet with the adviser for about half an hour per week. You may meet the second adviser on a monthly basis or as and when required. During the entire capstone project an adviser would typically spent around 10 hours with the group/individual for discussions in addition to the time spent to read and correct the documents to be submitted to the ICT Center of your school. Some advisers would want to meet the group/individual more often than this, especially at the start of the capstone project and at the writing stage. If the adviser requests, the group/individual would also have to regularly submit short progress reports on the capstone project work prior to the meeting (usually by email). A record of the meetings with the adviser should be formally recorded using the log form. When you go to see your adviser you should have prepared a written list of points you wish to discuss in the log form.

Take notes during the meeting so that you do not forget the advice you were given or the conclusions that were reached. Group/individual should obtain adviser's approval before each submission and ensure that the documentations meet the client and requirements.

Further, the adviser should be school accredited through attending the Annual IT Advisers' Seminar which will be called by PUPQC in the early stages of the first semester.

1.7 CAPSTONE PROJECT OWNERSHIP

- Capstone project ownership is given to the entire group/individual as one entity, regardless of workload. Each member is expected to do their share of the capstone project – no free riders.
- In special cases where capstone project ownership is in question; individuals / group/individual must meet with the CPEC.

Capstone Project Guidelines for PUP BSIT Students

- When submitted, your school already owns the capstone project of the group/individual.
- There are also legal documents that you need to comply as required by the school.

1.8 GROUP COMPOSITION

Students should preferably work in teams of two (2) to four (4) members depending on the complexity of the project. The adviser should be able to determine whether the team can complete the project on time.

During Capstone project 1, the students are already required to form their groups and such will require approval from the CPEC as the qualifications of these students will be verified.

1.8.1 Group Merging

In the event that when a group loses its members and becomes less than 2 members, the surviving members are allowed to join a group with the following conditions:

- The joining group/individual already passed the capstone project proposal;
- The accepting group has a written approval indicating the scope of work that will be given to the joining member/s; and
- The joining group has an approved “authority to be merged” from the CPEC which will determine pending obligations of the joining group/individual.

For a group with less than 4 members and are willing to stay together and continue the capstone project, the members should pay the corresponding capstone project fee of the missing member/s.

1.9 REST OF THE DOCUMENT

Part 2 describes the various submissions you have to make to meet the capstone project requirements.

Part 3 identifies the deadlines and describe what happens when they are not met.

Part 4 describes the capstone project structure and content.

Part 5 describes the capstone project assessment.

Part 6 describes the pitfalls one should be aware of.

Part 2 - CAPSTONE PROJECT SUBMISSIONS

This part describes the submissions one has to make to fulfill partial requirements of the capstone project.

2.1 CAPSTONE PROJECT PROPOSAL

Prior to commencing the capstone project work, the group/individual should submit a capstone project proposal using the Technical Solutions Design template in the latter parts that summarizes the intended work with capstone project goals. The group/individual should discuss the capstone project proposal with the chosen capstone project adviser, prior to submission. This document is placed in the Capstone Project as Part 2 which outlines the high level description of the project.

The group/individual is expected to look at some of the capstone project reports kept for reference at the ICT CENTER or library and get an idea of the type of work that has to be done. However note that the expected requirements from a capstone project do change annually and hence you need to go based on the guidelines given in this document and any subsequent web notices related to the capstone project. When selecting a capstone project you should verify from the client whether they had previously given such a capstone project to any other students, as implementing a similar capstone project for the same client is not allowed. Note that research based capstone projects will not be accepted.

2.2 CAPSTONE PROJECT LOGBOOK

The group/individual should maintain a logbook based on the log form that describes the work done during the capstone project. Regular entries should be recorded in the logbook including all meetings with the adviser, which should be countersigned by

Capstone Project Guidelines for PUP BSIT Students

him/her. Logbook entries should include decisions made at various stages, progress made, findings etc.. A professional and disciplined approach to planning and record maintenance should be adopted. The logbook should be submitted with the capstone project.

2.3 CAPSTONE PROJECT

The group/individual should start writing the documentation of the capstone project according to the guidelines given in Part 4.

2.4 USB / CLOUD STORAGE

Once the approval for the capstone project is obtained a USB or a cloud storage given by PUPQC consisting of the system should be prepared or uploaded. It should consist of developed software (complete program listing), test data and results, databases, manuals, capstone project and presentation slides. Each of them should be in appropriate folders. There should also be a README file in the root describing how to install and use the software. The capstone project, manuals and reports should be created in Portable Document Format (PDF). Group/individual need not include the software packages used such as the database management system. USB should be submitted by the specified date along with a detailed view of the Files and Folders of the USB. It should be labeled indicating the group/individual's index number, name and year. A printout of the directory contents showing the file names should be attached with appropriate comments so that the CPEC can identify the contents of the USB.

2.5 FINAL CAPSTONE PROJECT

Accepted capstone project shall be sewn, trimmed and bound and covered with **dark** cloth, leather or Rexene, preferably orange or the assigned color for BSIT capstone

Capstone Project Guidelines for PUP BSIT Students

project. On the **spine** of the capstone project, the **initials** and **surname** of the group/individual (at top of the spine), the **title** of the capstone project (abbreviated if necessary) (at the centre of the spine) and **year** (at the bottom of the spine) shall be given in gold lettering of suitable size. If the lettering will not fit across the spine it shall run along the spine reading from top to bottom.

The final capstone project should be submitted to the CPEC.

PART 3 - CAPSTONE PROJECT TIMETABLE

The group/individual should demonstrate the ability to work independently and meet all deadlines comfortably.

3.1 CAPSTONE PROJECT SCHEDULE

3.1.1 The following are the activities set for Capstone project 1 (this may vary as your CPEC may decide doing them earlier or later):

WEEK 1: Officially starts during student orientation

- Capstone Orientation

WEEK 2:

- Discussion of the business systems processes
- Selection and Submission of Adviser

WEEK 3:

- DEADLINE for Capstone project Technical Solutions Design (TSD)
- Orientation of Advisers

WEEK 4:

- Presentation of TSD

WEEK 5:

Capstone Project Guidelines for PUP BSIT Students

- Submission/Approval of Revised TSD

WEEK 6:

- Submission/Approval of Product Backlogs
- Submission/Approval of Sprint Backlogs
- Discussion Infrastructure for testing of per sprint output

WEEKS 7-14 (Completion of Sprints 1-4, 1 sprint is 2 weeks)

Per sprint submissions:

- Burndown Chart
- Test Results (Requirements Traceability Matrix)
- Output Presentation
- Link to system for testing of CPEC

WEEK 15:

- Presentation of the output for Capstone 1
- Comments/Revisions from panel

WEEK16 -17

- Submission of Revised Product Backlog, Sprints 6-10 Backlogs, RTM, Part 1
Burndown Chart

WEEK 18:

- Approval of the Revisions
- Retrospect Meeting with Adviser and Capstone Professor-in-charge

3.1.2 The following are the activities set for Capstone Project 1 (this may vary as your CPEC may decide doing them earlier or later):

WEEK 1:

- Part 2 Orientation

WEEKS 2-11: (Completion of Sprints 5 -10, 1 sprint is 2 weeks)

Per sprint submissions:

Capstone Project Guidelines for PUP BSIT Students

- Burndown Chart
- Test Results (Requirements Traceability Matrix)
- Output Presentation
- Link to system for testing of CPEC

WEEK 12:

- Mock Defense with Adviser and Capstone Professor-in-charge
- Effect revisions noted for the presentation material, system's prototype, and documentation

WEEK 13:

- Capstone Project Presentation before a group of faculty members and industry practioners

WEEKS 14-15:

- **"REVISIONS"**

WEEK 16:

- Approval of system's prototype, and documentation

WEEK 17-18:

- Secure Copyright at the main campus
- Submission

The specific dates depend on how PUPQC will be establishing the strategy.

3.2 SUBMISSIONS

All submissions have to reach the ICT CENTER or your professor-in-charge on or before the deadline. Submissions via delivery providers will be entertained on a case to case basis. Late submissions will need to comply to the policies of the university as the students will be given incomplete grades.

3.3 CAPSTONE PROJECT FEES

Capstone project can be done only by students registered for the 4th year of the BSIT degree Program. The capstone project fee is part of the fees set by the Accounting Office during assessment.

3.4 CAPSTONE PROJECT GRADE

Capstone project Grade will be given only to those who successfully completed the project and were able to comply to all the requirements.

3.5 CAPSTONE PROJECT SEMINAR/ORIENTATION

Members of the CPEC will conduct a seminar on the Final Year Capstone project for the group/individuals/individual who have submitted capstone project proposals. Objective is to elaborate this document and give the group/individual an opportunity to obtain any clarifications on their capstone project.

PART 4 - REPORT WRITING

The following guidelines are intended to assist with the writing of the capstone project and the presentation/demonstration. An outline of some important set of objective criteria is given to enable the groups/individual to ensure that their capstone project conforms to the required standard. The evaluation schemes indicated are applicable only if the capstone project conforms to the required standard.

4.1 FORMAT OF THE CAPSTONE PROJECT

The capstone project is a formal document and the structure and the general content requirements are described in section 4.3. This includes advice on how to organize the

Capstone Project Guidelines for PUP BSIT Students

work into parts, what should be covered in the main body of the work and what should go in the Appendices.

Groups/individuals/individual are strongly advised, while writing is in progress, to show each part to their advisers for necessary feedback especially on technical content. As for style and grammar, please follow the instructions given below to minimize correction time.

The format requirements are not overly restrictive (e.g. there is no requirement to use a particular font style for some parts of the capstone project). However, do not use too many different typefaces in the capstone project, or in general, too much time developing an elaborate visual presentation. It is better to keep the look of the capstone project simple and straightforward. (An elaborate presentation can in fact create a negative impression.)

By all means, use plotting/drawing packages to create graphs and figures. Hand-drawn figures will not be accepted.

4.1.1 Other Format Requirements

- The capstone project text (defined as everything except title page, table of contents, bibliography and appendices) should not normally exceed **200 pages excluding source codes**. On average the capstone project text is about 150 pages of 8.5 x 11 narrative text.
- The capstone project text should use **Arial 12-point** type and **1.5 spacing**. A typeface less than 10 points should **not** be used under any circumstances. Use **single sided** printing. **Left margin** is **37mm** (to allow space for binding), **top/bottom/right margin** is **25mm**. **Chapter heading 14pt bold**, **Section headings**

uppercase 13pt bold, subsection heading title 12pt bold, sub-sub-section (level 3) heading title 12pt bold.

- The total length (capstone project text together with appendices) of the capstone project should not be less than 150 **pages** except under unusual circumstances.
- All pages should be numbered with part 1 beginning on **page 1**. Use **Roman letters** for pages before that.
- Report writing style should use passive form. It is considered very bad style in a formal report to make explicit references to what the group/individual himself/herself did, as in “I decided...”. Scientific papers never use the first person in this way. The passive form as in ‘it was decided...’ is strongly preferred. In the capstone project, the first person could be used judiciously in the introduction and conclusions, but should be avoided everywhere else in the capstone project.
- Plagiarism is the presentation of another person’s thoughts or words as though they were group/individual’s owned. The group/individual should avoid this when writing his or her capstone project. All sentences or passages quoted in his or her report from other people’s work have to be specially acknowledged by clear cross-referencing to author, work and page(s). Direct quotations from published or unpublished work of others should always be clearly identified as such by being placed inside quotation marks, and full reference to their source should be provided in the proper form. Equally, if another person’s ideas or judgments are summarized, the group/individual should refer to that person in the main text of the capstone project, and include the work referred to in the bibliography. Failure to observe these rules may result in an allegation of cheating. All suspected cheating will be reported as an examination offense. Any illustrations, which are not the work of the group/individual can be used only with the explicit permission

of the originator and should be specially acknowledged. The capstone project is an important component of the degree and plagiarism in capstone project work is taken very seriously, and when discovered will imply severe penalties and consequences for the culprit's degree and possibly for the entire future career. A group/individual will fail the capstone project and the degree examination as a whole when plagiarism in capstone project work is discovered. Group/individual will not be allowed to repeat the capstone project and other degree components for a specified number of years. Therefore, it is important to give credit where it is due and acknowledge all work borrowed, and emphasize what the group/individual's distinct contribution has been in the capstone project.

4.2 CONTENTS OF THE CAPSTONE PROJECT

If you ever find time browsing the works of the past Capstone students, you will notice some good points and flaws in terms of the contents as specified. Remember that some employers would like to see a fresh graduate's output through the documented capstone project. As discussed earlier, this is the best venue to practice your writing skills besides being a future evidence of your productive college life. This can become part of your library collections!

The capstone project should be divided into a series of numbered parts and sections, each with titles. It consists of preface material, main chapters and appendices. A possible layout is described under section 4.3 and a summary of the contents is also given in this material.

4.3 PREFACE MATERIAL

This is the material that comes before the first part. This usually consists of a title page, declaration page, an abstract, an acknowledgement page, content page and a list of acronyms.

4.1 Title Page

This comprises the title of the capstone project, group/individual's name, BSIT registration no, index no, the name(s) of the adviser(s), the date of submission (Month and Year), and the following statement: "This capstone project is submitted in partial fulfillment of the requirement of the Degree of Bachelor of Science in Information Technology (External) of the <your school>". The title of the capstone project should be clear and describe the main area of work. Refer Appendix "F" for a sample title page.

4.3.2 Declaration

(See next page.)

The second page should contain the following signed declaration:

I certify that this capstone project does not incorporate, without acknowledgement, any material previously submitted for a Degree or Diploma in any University and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my capstone project, if accepted, to be made available for photocopying and for inter-library loans, and for the title and summary to be made available to outside organizations.

Signature of Group/individual

Date:../../....

.....

Name of Group/individual Members

Countersigned by:

Signature of adviser(s)

Date:../../....

.....

Name of Adviser(s)

4.3.3 Abstract

The abstract should encourage the reader to find out more, and has to identify the main technical aspects and distinguishing characteristics of the capstone project. The abstract should be brief but should provide a clear statement of what the capstone project is about, the scope of the work, a summary of the most important features/results set out in the capstone project and its relevance to the goals of the capstone project. The content should not exceed a page. The summary of the capstone project should be later reflected through the structure of the report presented in section 4.3.5. This will allow the reader of the abstract to quickly locate design, implementation and testing details of important features/results from the content page.

4.3.4 Acknowledgements

It is mandatory that a group/individual thank whoever has helped the group/individual technically or otherwise, during the capstone project. Your adviser and your client will obviously be pleased to be acknowledged as they would have invested quite a lot of time overseeing your progress. Acknowledgements should be brief and to the point and should be about half a page long, preferably a paragraph or two.

4.3.5 Table of Contents

Provides page numbers of all **sections** (up to 3 levels, e.g. 5.4.1), **tables** and **figures**. Part 1 begins on page 1. Use roman numerals for all previous pages (e.g. title page, declaration, abstract, acknowledgements, etc.). The overall structure should show a clear progression of logical thought. The structure of the capstone project should be seen to match the summary provided in the abstract. Choose self-explanatory section and sub-section titles relevant to the system under consideration (use titles such as “Requirements of a Human Resource System”, “Designing the Employee Module” instead of using generic titles such as “Requirement Definitions”, “Design” respectively).

4.3.6 List of Acronyms

Provide all abbreviations used in the capstone project in alphabetical order with their expected versions.

4.4 MAIN PARTS

Main parts should be written targeting non-technical readers such as a manager or sectional head of the client organization. Though the contents of Parts 2 onwards may seem technical, there are introductions that may give the reader the appropriate overview. Also, it is assumed that you have been taught about the underlying theories and applications of the capstone project such as the adoption of Agile Scrum methodology. All technical components are presented in appendices such as codes among other things.

The following is the list of the Capstone project contents:

1. Introduction

- Background of the Capstone Project
- Context and Scope
- Problem Statement
- Objectives and Goals
- Significance and Relevance
- Structure of the Document

2. Literature Review

- Agile Scrum Methodology Overview
- Enterprise Architecture Concepts
- Relevant Studies and Research
- Integration of Information Systems in Enterprise Environments

3. Methodology

- Agile Scrum Methodology in the Project
- Roles (e.g., Scrum Master, Product Owner, Development Team)
- Sprint Cycles (e.g., Sprint Planning, Daily Standups, Sprint Review)
- Scrum Artifacts (e.g., Product Backlog, Sprint Backlog)
- Integration Approach for Information Systems
- Introduction to TOGAF and the Four Architectural Domains

4. Requirements Analysis

- Stakeholder Identification
- Requirements Gathering Techniques
- User Stories and Use Cases
- Functional Requirements for Integration

5. Business Process Architecture

- Identification of Business Processes
- Business Process Diagrams
- Alignment of Integrated System with Business Processes
- Business Process Improvements

6. Application Architecture

- Components of Application Architecture
- Application Architecture Diagrams
- Integration of Software Modules
- Communication and Interaction Patterns

7. Data Architecture

- Data Sources and Types
- Data Flow Diagrams
- Data Storage and Management
- Data Synchronization Across Systems

8. Technology Architecture:

- Technology Stack and Infrastructure
- Network Topology and Configuration
- Software Technologies
- Scalability and Performance Considerations

9. Development Process

- Agile Scrum Roles and Responsibilities
- Sprint Planning and Backlog Management

Capstone Project Guidelines for PUP BSIT Students

- Sprint Execution and Deliverables
- Challenges Faced in the Development Process
- 10. Implementation
 - Technical Implementation Details
 - Tools and Technologies Used
 - Code Integration and Interoperability
 - Integration Testing and Debugging
- 11. Testing and Quality Assurance
 - Testing Strategies and Methodologies
 - Test Cases and Test Data
 - Test Results and Bug Reports
 - Quality Assurance Measures
- 12. Results and Evaluation
 - Project Outcomes and Deliverables
 - Alignment with Project Objectives
 - Stakeholder and User Feedback
 - Lessons Learned
- 13. Conclusion
 - Key Takeaways and Summary
 - Project Achievements and Contributions
 - Future Work and Enhancements
 - Closing Remarks
- 14. References
 - List of Cited Sources and References
- 15. Appendices

Appendix I Adviser Acceptance (Functional)

Appendix J Panel Evaluation and Signature (plus photo ops during defense)

Appendix K Capacity Planning (Estimates on Storage Consumption – 5 years)

Appendix L Pilot Companies' Background with proofs of interviews

Appendix M Cloud Copy of the Codes (1 year validity)

Appendix N IMRAD Format Summary

Appendix O Comparison of the EIS to existing EIS's (5 pages)

16. Supplementary Material

Technical Documentation

Scrum Artifacts (e.g., Burndown Charts, Sprint Backlogs) – attach all what you have been uploading in the google classroom

Technical Documentation Outline

16.1 System Architecture

Overview of the overall system architecture

Detailed diagrams illustrating system components and their interactions

Explanation of component responsibilities and relationships

16.2 Information Systems Integration:

Description of how the class's information systems were integrated

Diagrams showing the integration points and data flows

Data transformation and mapping processes

16.3 Application Design and Development

Detailed information about the software modules and components

Code snippets and code structure

Algorithms and data structures used

Libraries and frameworks utilized

16.4 Database Schema and Data Management

Database schema design

Entity-relationship diagrams (ERDs)

Data modeling and normalization

CRUD (Create, Read, Update, Delete) operations

16.5 Network Configuration

- Network topology and configuration details
- Security measures (e.g., firewalls, encryption)
- Protocols and communication methods used
- Load balancing and failover mechanisms

16.6 Deployment and Infrastructure

- Deployment strategies (e.g., cloud, on-premises)
- Hardware specifications and server configurations
- Configuration management (e.g., scripts, automation tools)
- Scalability and resource optimization

16.7 Security Measures

- Security protocols and measures implemented
- Authentication and authorization mechanisms
- Encryption and data protection
- Incident response and mitigation procedures

16.8 Testing and Quality Assurance:

- Detailed test cases and scenarios
- Test data and expected outcomes
- Test results and validation against requirements
- Performance testing and optimization efforts

16.9 System Monitoring and Maintenance:

- Tools and techniques for monitoring system health
- Logging and error handling mechanisms
- Scheduled maintenance and updates
- Disaster recovery and backup procedures

16.10 APIs and Integration Points:

- Documentation of APIs used for integration
- API endpoints, methods, and data formats
- How external systems interact with your project

16.11 User Documentation:

- User manuals and guides (if applicable)
- Instructions for system users
- Troubleshooting and FAQs

16.12 Known Issues and Troubleshooting

- List of known issues and their status
- Troubleshooting steps for common problems
- Contact information for technical support

16.13 Version Control and Source Code Repository

- Description of version control system (e.g., Git)
- Repository location and access details
- Branching and merging strategies

16.14 DevOps and Continuous Integration/Continuous Deployment (CI/CD)

- CI/CD pipeline setup and configurations
- Automation scripts and tools used
- Deployment strategies and rollback procedures

16.15 Licensing and Open Source Libraries

- Information about licenses for software and libraries used
- Credits and attributions for open-source components

16.16. Performance Metrics and Monitoring

- Metrics collected and monitored
- Tools and dashboards used for performance analysis

- Actions taken based on performance data

16.17 Capacity Planning (Estimates on Storage Consumption – 5 years)

4.4.1 Part 1 – INTRODUCTION

BACKGROUND OF THE CAPSTONE PROJECT

In this part, you provide the reader with the context of the project. Explain what led to the initiation of the project. This might include a brief history of the problem or the need the project aims to address.

Discuss any previous work or research that might have contributed to the project's conception.

CONTEXT AND SCOPE

Here, you define the project's context, outlining the environment or domain in which it operates. Discuss any specific challenges, constraints, or limitations related to the context that will impact the project. Define the scope to make clear what the project includes and what it does not.

This helps manage expectations and prevent scope creep.

PROBLEM STATEMENT

State the problem the project is seeking to solve. Provide a clear and concise statement of the problem, preferably in a single sentence. Detail why this issue is important or urgent and how it affects stakeholders or the broader community. Consider using data or evidence to support your problem statement.

GOALS AND OBJECTIVES

List the specific goals and objectives of the capstone project. These should be clear, measurable and achievable outcomes that the project is aiming for. Objectives are what the project will achieve, while objectives are the desired results. Make sure they align with the problem statement.

IMPORTANCE AND RELEVANCE

Explain why the project is important and relevant. Discuss its potential impact and benefits. Highlight who benefits from the project's success and how the project addresses current challenges or needs on the ground. Consider presenting statistics, market trends or expert opinions to emphasize relevance.

DOCUMENT STRUCTURE

Provide readers with an overview of how the document is organized. Describe the sections, subsections, and chapters they can expect to find in the document. This acts as a road map for readers and helps them quickly locate specific information in the document. The "Introduction" section should be engaging and informative, attracting the reader's interest and providing essential context for the rest of the document. This is an opportunity to convince readers that the project is worth their time and attention, and that it is well designed to create a solid foundation for the detailed technical and methodological content that follows.

In summary, the "Introduction" section provides an overview of the background, context, problem statement, objectives, and significance of the university system project. It also outlines the structure of the document, offering readers a roadmap for understanding the project's intricacies and contributions to higher education management.

Here's a sample content for the "Introduction" section of your capstone project documentation, assuming that the project is a university system:

BACKGROUND OF THE CAPSTONE PROJECT

The development of our university system represents the culmination of our Bachelor of Science in Information Technology (BSIT) program. This capstone project is the result of the collaborative efforts of our team of dedicated students who have spent several

semesters honing their technical and project management skills. The project was conceived to address the complex challenges faced by modern universities in managing student records, optimizing faculty course scheduling, and enabling data-driven decision-making. The project's genesis is rooted in the increasing demand for efficient and integrated university management solutions to meet the evolving needs of higher education institutions.

CONTEXT AND SCOPE

In the context of higher education, universities today are tasked with managing an ever-growing volume of student data, maintaining an agile and adaptable course scheduling system, and providing stakeholders with access to meaningful insights. Our university system aims to streamline these processes, offering a comprehensive solution for university administrators, faculty members, and students. The scope of our project encompasses the development of a fully integrated system that addresses these critical university management challenges while promoting efficiency, transparency, and data-driven decision-making.

PROBLEM STATEMENT

Modern universities face multifaceted challenges in managing student data, scheduling courses, and making informed decisions. These challenges include:

1. Tedious and error-prone manual data entry and record-keeping processes.
2. Inefficient course scheduling that may lead to faculty workload imbalances and scheduling conflicts.
3. A lack of real-time, data-driven insights for decision-making, hindering strategic planning.

Our project seeks to resolve these challenges by creating a university system that automates student data management, optimizes faculty course scheduling, and

empowers administrators with a comprehensive dashboard for real-time data insights.

OBJECTIVES AND GOALS

The primary objectives and goals of our university system project are as follows:

1. Develop a student information management module to streamline student enrollment, academic records, and data reporting.
2. Create a faculty course scheduling system to optimize course allocation and faculty workload management.
3. Implement an administrative dashboard that offers real-time data visualization for informed decision-making.
4. Provide comprehensive user manuals and technical documentation for system management and maintenance.
5. Collect and incorporate feedback from stakeholders to continuously improve the system.

SIGNIFICANCE AND RELEVANCE

The significance of our university system project lies in its potential to transform university management and administration. By automating manual processes, optimizing faculty scheduling, and providing real-time insights, our system can improve operational efficiency, enhance data accuracy, and enable universities to make data-driven decisions. This project is highly relevant in the context of modern higher education institutions striving to meet the demands of a dynamic and data-centric educational landscape.

STRUCTURE OF THE DOCUMENT

This document is organized into several sections, each focusing on a specific aspect of the capstone project. The subsequent sections will delve into the project's technical and

architectural details, development process, testing, and quality assurance, as well as the results, evaluation, and lessons learned. By the end of this document, readers will gain a comprehensive understanding of our university system and its impact on the university's operational efficiency and effectiveness.

4.4.2 Part 2 – LITERATURE REVIEW

OVERVIEW OF THE AGILE SCRUM METHODOLOGY

Provide a comprehensive overview of the Agile Scrum methodology. Discuss its principles, values and practices. Explain how Agile Scrum is used in software development projects, highlighting its iterative and incremental nature, emphasis on customer collaboration, and adaptability to changing requirements change. Covers key roles and responsibilities within Scrum teams, such as Scrum Master, Product Owner, and Development Team. Cite research and industry standards related to Agile Scrum.

ENTERPRISE ARCHITECTURE CONCEPTS

Explains the basic concepts of Enterprise Architecture (EA). Discuss EA's role in aligning the business and technology components of an organization. Covers key EA frameworks and standards, such as TOGAF (Open Group Architecture Framework), and their relevance to project architecture. Describes the areas of architecture within EA, including business processes, applications, data, and technology. Emphasizes the importance of a coherent architecture to achieve organizational goals and facilitate the integration of information systems.

Related Studies and Research: Review previous studies and research related to your capstone project area. Summarize key findings, methods, and conclusions from scholarly articles, papers, and reports that address similar or related issues. Analyze how these studies inform your project, what gaps or opportunities they

identify, and what lessons can be learned. This demonstrates that your project builds on existing knowledge and research.

Integrating information systems into the business environment

Discuss the challenges and benefits of integrating information systems into the business environment. Explain how information systems integration can lead to improved efficiency, data consistency, and better decision making. Review common integration techniques and technologies.

Provides examples of successful information systems integration projects and their impact on organizations. Illustrates how integration aligns with EA concepts, emphasizing the importance of cohesion and consistency across architectural domains.

The “Literature Review” section should serve as an informative and coherent narrative, linking the methodology, architecture, and goals of the project to the existing body of knowledge. It demonstrates that your project is based on established principles and best practices, and that it solves an important and relevant problem in its field. Be sure to cite your sources accurately and refer to industry standards and authoritative publications where available.

Here's a sample "Related Literature" section for your university system project with citations for 15 literature sources included within the discussion

Related Literature

The development of a comprehensive university system is informed by a rich body of literature that encompasses various aspects, including software development methodologies, enterprise architecture, and higher education management. In this section, we review 15 key literature sources that underpin the conceptual framework of our project.

AGILE SCRUM METHODOLOGY OVERVIEW

1. Schwaber, K., & Sutherland, J. (2017) emphasized the principles and roles in Agile Scrum in "The Scrum Guide," laying the foundation for Agile project management.
2. The "Manifesto for Agile Software Development" by Beck et al. (2001) defined the values and principles of Agile methodologies, setting the stage for Agile practices.
3. In "Agile Software Development," Cockburn (2001) provided a comprehensive view of Agile practices, emphasizing their collaborative and iterative nature.
4. Cohn's work in "Succeeding with Agile: Software Development Using Scrum" (2010) offers practical guidance on implementing Scrum and highlights its effectiveness in software development projects.
5. Schwaber's "Agile Project Management with Scrum" (2004) provides deep insights into Scrum's application in project management, including Scrum roles and processes.

ENTERPRISE ARCHITECTURE CONCEPTS

1. The Open Group's "TOGAF® Standard, Version 9.2" (2018) offers structured guidance for enterprise architecture development, emphasizing the importance of defining architectural domains.

2. In "Enterprise Architecture as Strategy: Creating a Foundation for Business Execution," Ross, Weill, and Robertson (2006) underscored the role of enterprise architecture in achieving business objectives.
3. Lapalme's "Enterprise Architecture: A Disciplined Approach" (2001) provides an introduction to enterprise architecture concepts, helping in understanding its relevance to organizations.
4. Zachman's "A Framework for Information Systems Architecture" (1987) laid the early groundwork for organizing architectural concepts.

RELEVANT STUDIES AND RESEARCH

1. Smith and Johnson (2018) conducted a study on "Modernizing University Systems: Challenges and Opportunities" and emphasized the demand for integrated university systems that offer comprehensive solutions for student data management, faculty scheduling, and data-driven decision-making.
2. Brown et al. (2019) addressed the importance of automation in higher education administration in their study "Automation and Efficiency in Higher Education Administration: A Case Study."
3. Smith, A., & Johnson, L. (2018) conducted a study on "Managing Complexity in Higher Education: Using Data-Driven Decision Making," focusing on data-driven decision-making in higher education.
4. Martinez, R., et al. (2018) explored "Faculty Workload Management in Higher Education: Challenges and Opportunities," providing insights into faculty workload management, a critical aspect of our university system.
5. In "Improving Student Enrollment and Records Management: A Case Study of Higher Education Institutions," Johnson and Clark (2019) offered

insights into the challenges and opportunities related to student enrollment and records management in higher education.

INTEGRATION OF INFORMATION SYSTEMS IN ENTERPRISE ENVIRONMENTS:

"Enterprise Architecture as Strategy: Creating a Foundation for Business Execution" by Ross and Weill (2010) is revisited for its discussion on the role of enterprise architecture in aligning business strategy with IT.

These 15 literature sources, spanning Agile Scrum methodology, enterprise architecture, relevant studies, and information system integration within higher education, provide a solid foundation for our university system project. They inform our approach, align us with industry best practices, and help us address the evolving needs of higher education institutions.

4.4.3 Part 3 – METHODOLOGY

AGILE SCRUM METHOD IN PROJECTS

Describe how to apply Agile Scrum method in your project. Highlight its iterative and incremental approach to development, emphasizing how it promotes flexibility and collaboration. Explain why you chose Agile Scrum for your project and how it fits with the project goals. Refers to any adjustments or modifications made to meet the specific requirements of the project.

ROLES (e.g. Scrum Master, Product Owner, Development Team)

Details the key roles within your Agile Scrum team. Explain the responsibilities and tasks of each role, such as the Scrum Master's focus on enabling the team to grow and remove impediments, and the Product Owner's role in identifying and Prioritize requirements and responsibilities of the development team in implementing project features.

SPRINT CYCLES (e.g.sprint planning, daily projections, sprint reviews)

Provides an overview of the Agile Scrum sprint cycle. Describe key rituals, including sprint planning, daily planning (or Daily Scrum), and sprint reviews. Explain how these ceremonies help manage work, communicate, and track progress within the team. Highlight any unique considerations or adjustments specific to your project.

SCRUM ARTIFACTS (e.g., Product Backlog, Sprint Backlog)

Discuss the core artifacts used in Agile Scrum. Explain the purpose and content of the Product Backlog, which represents the project's requirements, and the Sprint Backlog, which contains the work selected for a particular sprint. Clarify how these artifacts are created, maintained, and leveraged throughout the project.

INTEGRATION APPROACH FOR INFORMATION SYSTEMS

Present your approach to integrating the information systems developed by the class. Describe the integration methodology, techniques, and tools used to ensure seamless collaboration among the systems. Discuss any data mapping, data transformation, or communication protocols used to enable integration.

INTRODUCTION TO TOGAF AND THE FOUR ARCHITECTURAL DOMAIN

Provides an introduction to TOGAF (Open Group Architecture Framework) and its relevance to your project. Explain how to apply the TOGAF architectural method in your project.

Introducing TOGAF's four architecture areas: Business Process Architecture, Application Architecture, Data Architecture, and Technology Architecture. Explain their importance and how they are addressed in your project. This "Methodology" section should provide a clear understanding of the framework and processes used to manage and execute your capstone project. It ensures that readers are

familiar with the project's methodology and the roles of team members, ceremonies, and artifacts that guided the development process. Additionally, it highlights the integration approach and the importance of TOGAF in achieving architectural consistency.

Here's a sample "Methodology" section for your university system project.

METHODOLOGY

The methodology section outlines the approach and processes used in the development of our university system project. The successful implementation of a comprehensive university system demands a well-structured approach that ensures alignment with project objectives, stakeholder needs, and industry best practices.

AGILE SCRUM METHODOLOGY IN THE PROJECT

In line with the project's goals of flexibility and adaptability, we have chosen to implement the Agile Scrum methodology. Agile Scrum is a collaborative and iterative framework that promotes regular communication, adaptability, and responsiveness to changing requirements. It is particularly well-suited for software development projects in higher education, given the evolving needs and dynamic nature of academic institutions.

Roles

1. Scrum Master: The Scrum Master, responsible for ensuring the Scrum process is followed, facilitating Scrum events, and removing impediments.
2. Product Owner: The Product Owner is the project's key stakeholder, representing the university's interests, defining requirements, and prioritizing features.

3. Development Team: The cross-functional development team comprises individuals with diverse skills and expertise, responsible for the actual development work.

SPRINT CYCLES

The project is divided into time-bound iterations known as sprints. Each sprint typically lasts two to four weeks, during which the team focuses on delivering a specific set of features or functionalities. The use of sprints enables us to maintain a dynamic approach and adjust project priorities as needed.

SCRUM ARTIFACTS

1. Product Backlog: The Product Backlog is a prioritized list of all project requirements and features. It serves as the single source of truth for the project and is continually refined.
2. Sprint Backlog: At the beginning of each sprint, a subset of items from the Product Backlog is selected and moved to the Sprint Backlog. These are the items to be developed during the sprint.

INTEGRATION APPROACH FOR INFORMATION SYSTEMS

The integration of various information systems within the university environment is a crucial component of this project. To ensure that data flows seamlessly between different modules, we are adopting a data-centric integration approach. This approach involves the creation of standardized APIs and data formats to enable the exchange of information between systems.

INTRODUCTION TO TOGAF AND THE FOUR ARCHITECTURAL DOMAINS

As part of the project's enterprise architecture framework, we are adhering to The Open Group Architecture Framework (TOGAF). TOGAF is a comprehensive approach to enterprise architecture, emphasizing the importance of defining four architectural domains:

1. Business Process Architecture: Identifying and modeling key university processes, such as student admissions, course registration, and faculty management.
2. Application Architecture: Defining the components, modules, and technologies that make up the university system, ensuring alignment with business processes.
3. Data Architecture: Designing data models, storage, and synchronization mechanisms for effective data management across the university system.
4. Technology Architecture: Determining the technology stack, infrastructure, network topology, and software technologies necessary for system scalability and performance.

By integrating these four architectural domains, our project aims to create a holistic and well-aligned university system that not only meets current requirements but also provides a foundation for future growth and adaptability.

This methodology provides a structured and agile approach to the development of our university system, ensuring that it aligns with Agile Scrum principles, data integration best practices, and the comprehensive framework of TOGAF. It allows for regular communication with stakeholders, adaptability to changing needs, and the delivery of a system that meets the evolving requirements of higher education institutions.

4.4.4 Part 4 – REQUIREMENTS ANALYSIS

STAKEHOLDER IDENTIFICATION

In this section, you should identify and describe the key stakeholders involved in the project.

Stakeholders are individuals or groups interested in or affected by the outcome of the project.

This can include end users, customers, project sponsors, managers, developers, and other stakeholders. Discuss the importance of analyzing stakeholders to understand their requirements and expectations.

REQUIREMENTS GATHERING TECHNIQUES

Explains the techniques and methods used to gather project requirements. This may include interviews, surveys, focus groups, observations or workshops. Details how these techniques are applied, who is involved in the requirements gathering process, and what specific information or data is collected. Discuss any difficulties encountered during this process and how they were resolved.

USER STORIES AND USE CASES

User stories and use cases are essential tools for capturing and documenting requirements from the end user perspective. Describe how to create user stories to show the feature(s) from the user's perspective. Provide sample user stories and explain how they are prioritized and managed. Discuss the use of use cases to describe different scenarios and interactions within the system, helping to capture functional requirements.

FUNCTIONAL REQUIREMENTS FOR INTEGRATION

This section should focus on specific functional requirements related to the integration of information systems developed by the class. Describe the detailed

features, functions, and behaviors required for successful integration. These requirements may include data exchange protocols, synchronization mechanisms, error handling, and user access controls related to embedded systems. Remember to mention any dependencies and constraints that affect the deployment.

In summary, this section aims to establish a clear understanding of the project requirements, ensure that they align with stakeholder needs, and present them in a structured manner.

Effective requirements gathering and stakeholder identification techniques are essential to gather accurate and complete information. User stories and use cases help translate these requirements into user-centric stories, and functional integration requirements ensure that information systems work seamlessly together, meeting project objectives.

Here's a sample "Requirements Analysis" section for your university system project:

The requirements analysis phase is a critical step in the development of our university system. It involves gathering, documenting, and analyzing the needs and expectations of stakeholders, which include university administrators, faculty, students, and staff. The goal of this phase is to establish a clear and comprehensive understanding of the system's functional and non-functional requirements.

STAKEHOLDER IDENTIFICATION

Identifying the key stakeholders is the first step in requirements analysis. In our university system project, stakeholders include:

1. University Administrators: They are responsible for overall management, strategic planning, and decision-making at the university.
2. Faculty Members: They need access to course management and grading systems to streamline teaching and assessment processes.
3. Students: Students require user-friendly interfaces for course registration, academic records, and other student services.
4. Administrative Staff: Administrative staff manage student enrollment, financial records, and other administrative functions.

REQUIREMENTS GATHERING TECHNIQUES

To capture the diverse needs of stakeholders, we employ a combination of requirements gathering techniques:

1. Surveys: We distribute online surveys to university administrators, faculty, and students to collect quantitative and qualitative data on their requirements and pain points.
2. Interviews: We conduct one-on-one interviews with key stakeholders, including university administrators and faculty, to gain a deeper understanding of their specific needs and priorities.
3. Workshops: Requirements workshops are organized with administrative staff to collaboratively identify their system requirements, focusing on the administrative processes.
4. Observation: We observe the current manual processes and workflows in the university to identify areas where automation can bring efficiency.
5. Document Analysis: Existing university policies, procedures, and academic guidelines are analyzed to ensure that the system aligns with institutional regulations and standards.

USER STORIES AND USE CASES

Once requirements are gathered, they are translated into user stories and use cases for the system. User stories represent specific user scenarios and help in understanding how the system will be used. Use cases provide a high-level view of the system's functionalities from an end-user perspective.

FUNCTIONAL REQUIREMENTS FOR INTEGRATION

Functional requirements are defined to outline the specific functionalities of the university system. These requirements include:

1. **Student Enrollment:** The system should support online enrollment, class registration, and fee payment.
2. **Faculty Management:** Faculty members should be able to access course materials, record grades, and communicate with students.
3. **Administrative Functions:** Administrative staff should have tools for managing student records, financial transactions, and university resources.
4. **Student Services:** The system should provide access to academic records, library resources, and student support services.

NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements encompass performance, security, and scalability considerations. These requirements include:

1. **Performance:** The system should respond to user requests promptly, even during peak usage times.
2. **Security:** Data security measures should be in place to protect sensitive information such as student records and financial data.

3. Scalability: The system should be scalable to accommodate the growing user base and increased data volume.

In summary, the requirements analysis phase lays the foundation for the development of our university system. It involves identifying key stakeholders, gathering their needs using various techniques, and translating these into user stories, use cases, functional requirements, and non-functional requirements. This thorough analysis ensures that the system will meet the diverse needs of the university community while adhering to performance, security, and scalability standards.

4.4.5 PART 5 – BUSINESS PROCESS ARCHITECTURE

BUSINESS PROCESS ARCHITECTURE

Identification of Business Processes: Begin by identifying and defining the key business processes within the organization or domain relevant to your project. These processes are the fundamental steps and activities that drive the organization's operations.

It's crucial to work closely with stakeholders to ensure an accurate understanding of these processes. Discuss the significance of identifying these processes for your project and how they impact the organization's success.

BUSINESS PROCESS DIAGRAMS

Create and present business process diagrams that visually represent the identified processes. Use recognized notation such as Business Process Model and Notation (BPMN) to illustrate the flows, tasks, decisions, and interactions within these processes. These diagrams provide a clear and visual representation of how the business operates, making it easier for stakeholders to understand and analyze.

BUSINESS PROCESS ARCHITECTURE: EXAMPLE

Online Medical Support and Services System

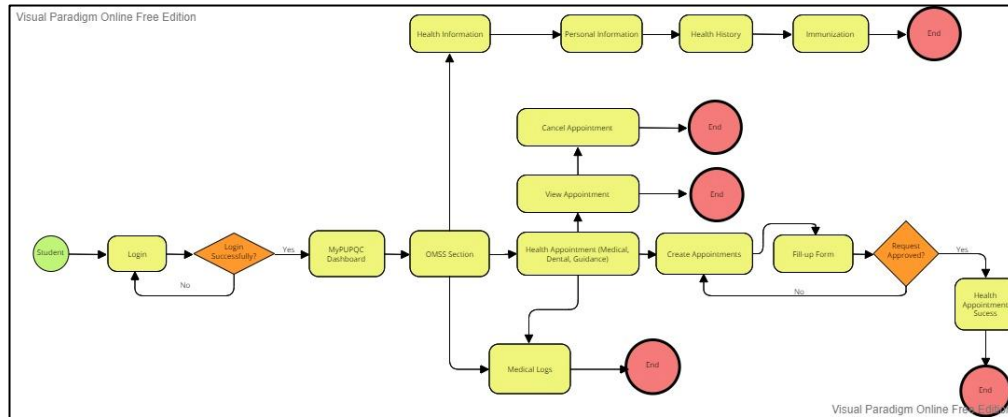


Figure 3. Student User's BPA for OMSSS

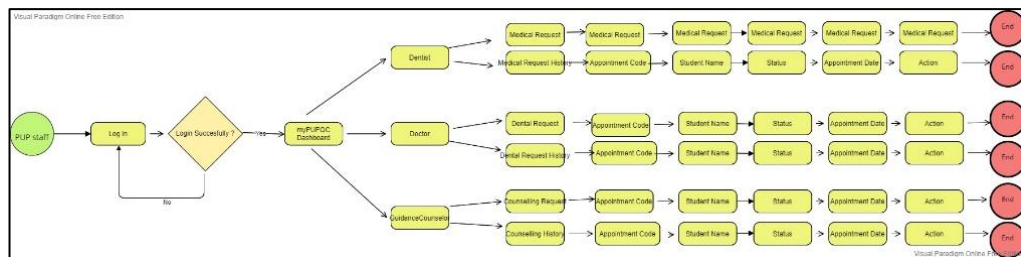


Figure 4. PUP Staff User's BPA for OMSSS

Online Document Request and Tracking System

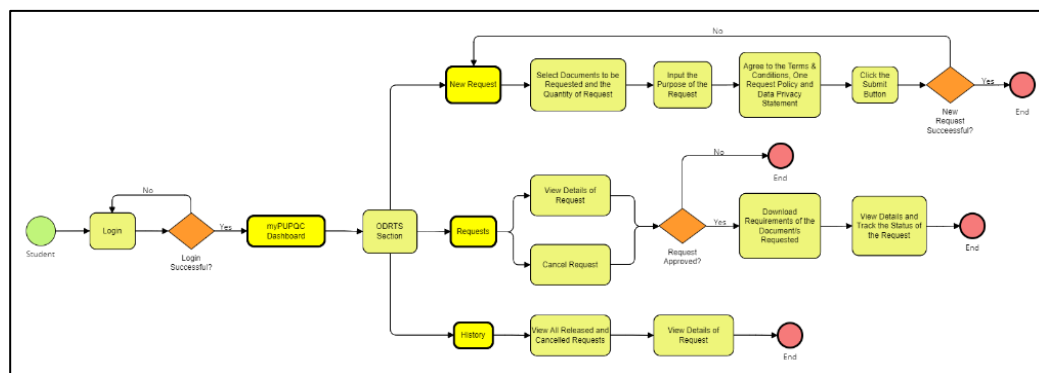


Figure 5. Student User's BPA for ODRTS

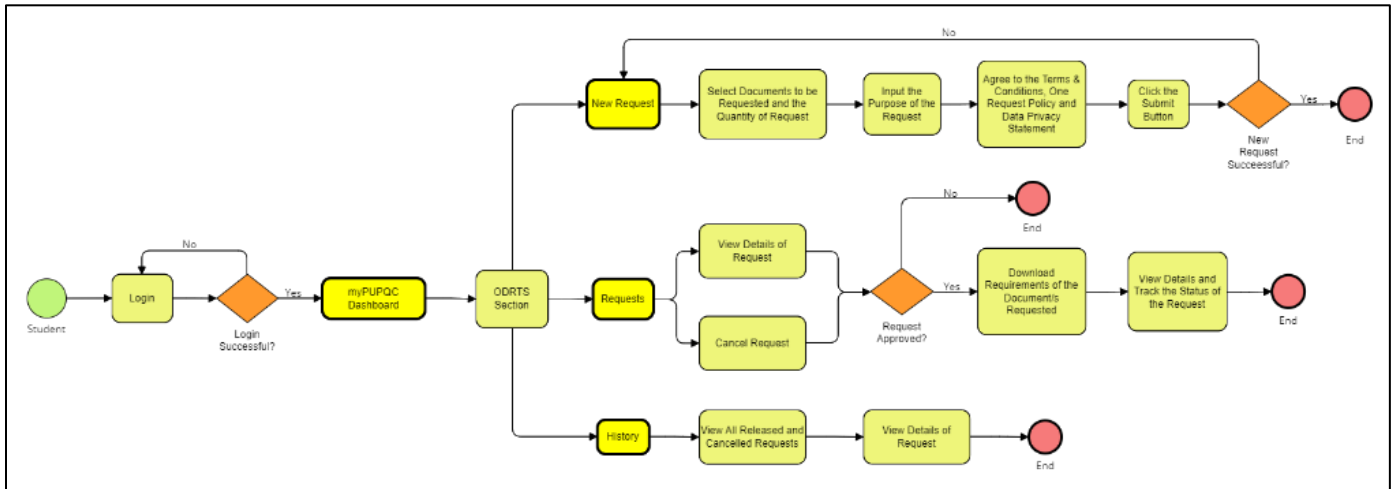


Figure 6. PUP Staff User's BPA for ODRTS

Capstone Project Guidelines for PUP BSIT Students

Electronic Venue Records for Scheduled Events and Reservation System (EVRSERS)

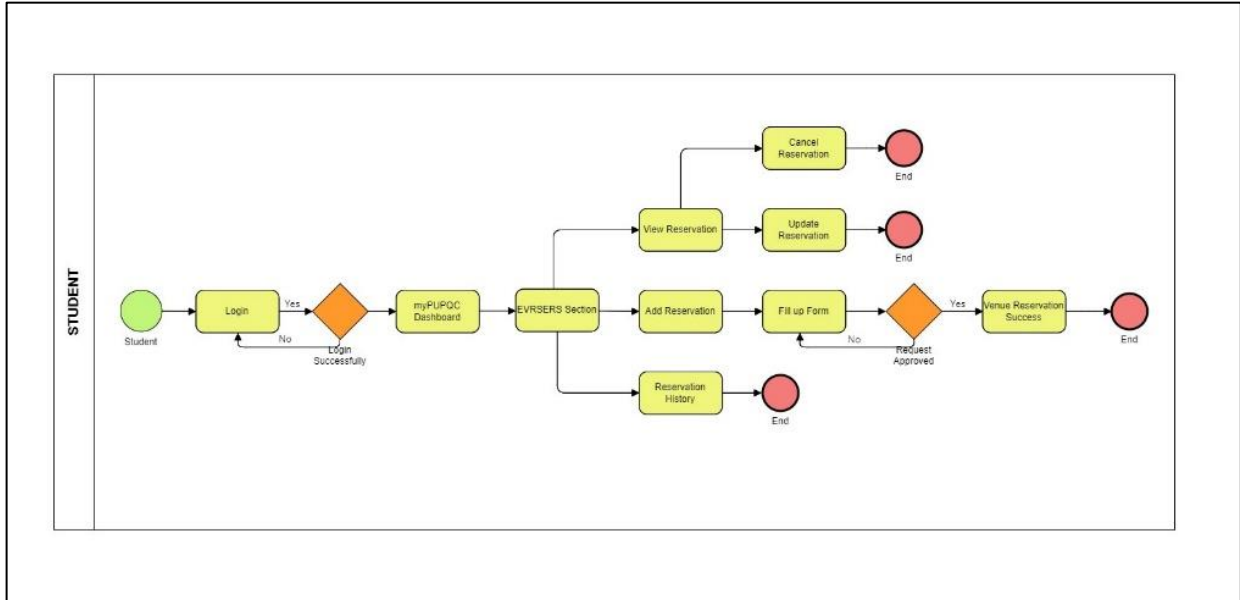


Figure 7. Student User's BPA for EVRSERS

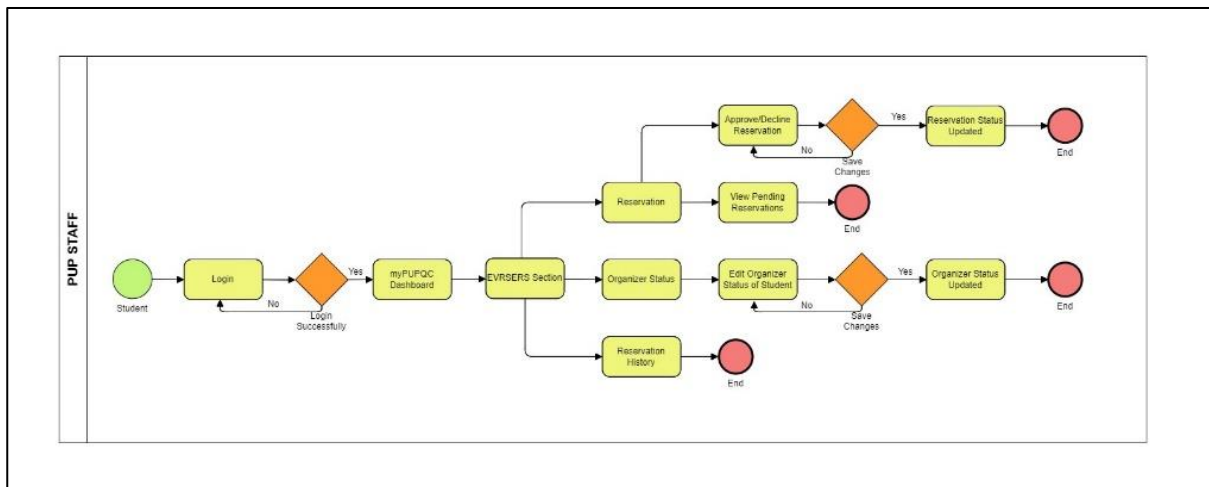


Figure 8. PUP Staff User's BPA for EVRSERS

ALIGNMENT OF INTEGRATED SYSTEMS WITH BUSINESS PROCESSES

Explain how integrated systems align with an organization's business processes. Detail the specific integration points where the system interacts with these processes. Describe how integrated systems support, improve, or automate aspects of business processes. Discuss how this alignment benefits the organization, such as improving efficiency, reducing errors, or enabling new features.

BUSINESS PROCESS IMPROVEMENT

Discusses how integrated systems have the ability to deliver improvements to an organization's business processes. Highlight any identified opportunities to streamline processes, reduce manual effort, or improve overall workflow. Explain how these improvements contribute to project goals and organizational goals. If possible, provide before and after scenarios to illustrate the impact of these improvements.

In short, "Business Process Architecture" is about understanding, documenting, and optimizing an organization's core business processes. It aims to create a clear link between the integrated system and these processes, ensuring that the project is aligned with the needs and operational goals of the organization. Accurate identification of business processes, visualization through diagrams, alignment, and potential improvements are all essential elements of this section.

Here's a sample "Business Process Architecture" section for your university system project:

BUSINESS PROCESS ARCHITECTURE:

The Business Process Architecture component of our university system project focuses on the identification, modeling, and optimization of key business processes within the higher education institution. This section outlines the critical steps in defining, documenting, and improving these processes to enhance efficiency and support the overall goals of the university.

IDENTIFICATION OF BUSINESS PROCESSES

To create an effective university system, it's crucial to identify and understand the key business processes that underpin the institution's operations. Some of the central business processes in a university context include:

1. **Student Enrollment:** This process encompasses admissions, course registration, fee payment, and student records management.
2. **Academic Administration:** Academic processes, such as course scheduling, curriculum management, and grading, fall under this category.
3. **Faculty Management:** This process covers recruitment, assignment, and performance evaluation of faculty members.
4. **Financial Management:** Managing budgeting, financial transactions, and auditing processes is essential for the university's financial stability.
5. **Student Services:** Providing services such as library access, counseling, and career guidance to students is a key component of the university's mission.

BUSINESS PROCESS DIAGRAMS

To visually represent the identified business processes, we create process diagrams. These diagrams use standardized symbols and notations to illustrate the flow of activities, decisions, and data within each process. For instance, a process diagram for "Student Enrollment" would depict the sequence of steps from application submission to enrollment confirmation.

ALIGNMENT OF INTEGRATED SYSTEM WITH BUSINESS PROCESSES

The university system's architecture should align seamlessly with the identified business processes. The integration of the system with these processes ensures a coherent and streamlined approach to university operations. For example, when a student registers for a course, the system should update records, trigger notifications to faculty, and process payment transactions within the enrollment process.

BUSINESS PROCESS IMPROVEMENTS

Our goal is not only to automate existing processes but also to improve them. By leveraging the capabilities of the university system, we aim to:

1. **Reduce Manual Effort:** Automate repetitive and manual tasks to save time and resources.
2. **Enhance Accuracy:** Minimize data entry errors and inaccuracies in student records and financial transactions.
3. **Streamline Communication:** Facilitate better communication between stakeholders, such as students, faculty, and administrators.
4. **Enable Data-Driven Decision-Making:** Ensure that data generated by the processes is available for analysis, supporting data-driven decision-making.

Through these process improvements, the university can operate more efficiently, allocate resources effectively, and provide a better experience for students and faculty.

In conclusion, the Business Process Architecture section outlines the steps taken to identify, model, and optimize key business processes within the university. Through process diagrams, alignment with the integrated system, and process improvements, the project aims to enhance operational efficiency and support

the university's mission of providing quality education and services to its community.

4.4.6 Part 6 - APPLICATION ARCHITECTURE

COMPONENTS OF APPLICATION ARCHITECTURE

Begin by identifying and defining the core components of the application architecture within your integrated system. These components may include user interfaces, server-side logic, databases, external APIs, and any other software modules or services relevant to the project. Explain the role and purpose of each component within the system.

APPLICATION ARCHITECTURE DIAGRAMS

Create architectural diagrams that visually represent the components and their relationships within the application architecture. Diagrams can use standard notations like UML (Unified Modeling Language) or any other appropriate visual representation. These diagrams provide a clear overview of the system's structure, making it easier for stakeholders to grasp its complexity.

APPLICATION ARCHITECTURE DIAGRAM: AN EXAMPLE

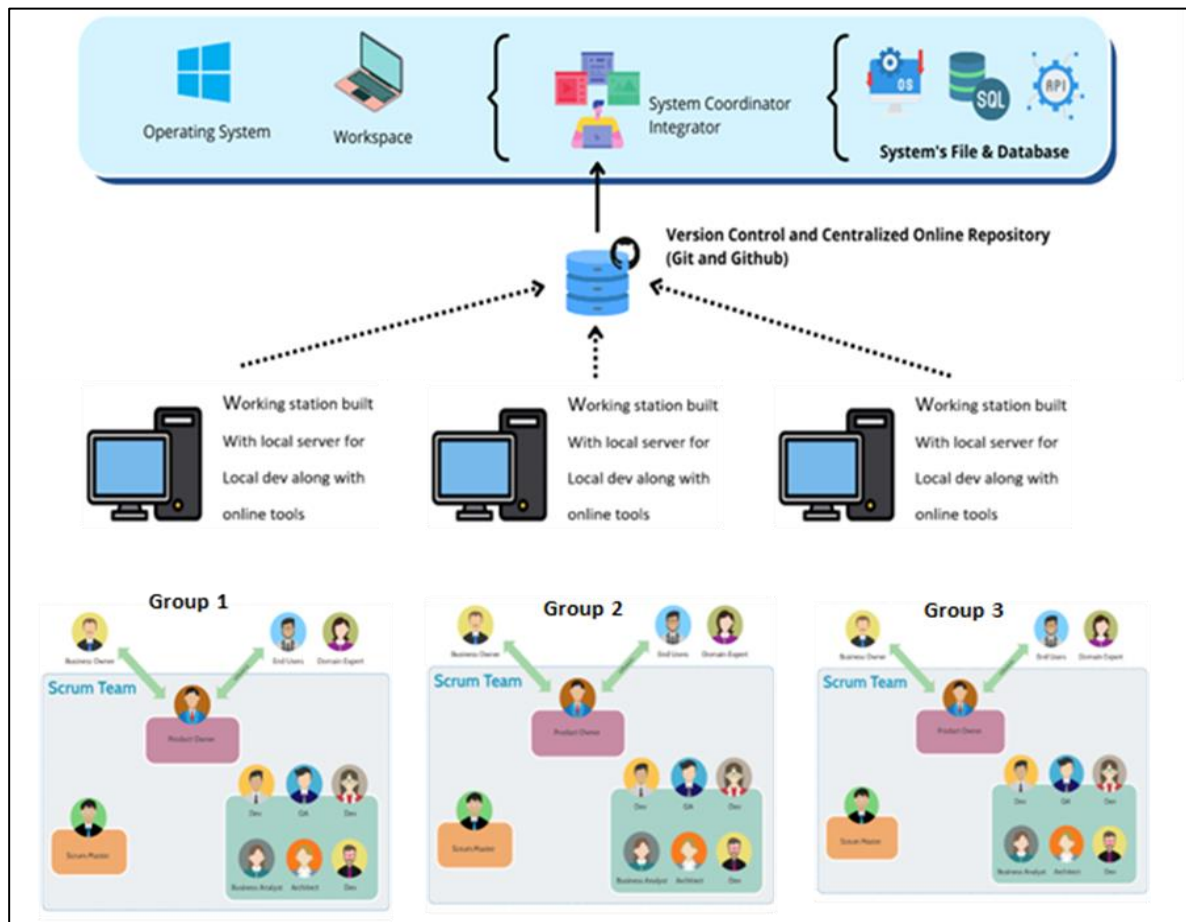


Figure 1. Application Architecture of myPUPQC

INTEGRATION OF SOFTWARE MODULES

Describe how the software modules within the integrated system are integrated and interact. Explain the communication protocols, data flows, and integration points between these modules. Discuss the mechanisms and patterns used to ensure that different components can work together cohesively.

COMMUNICATION AND INTERACTION PATTERNS

Detail the patterns and strategies used for communication and interaction within the application architecture. This includes how various modules and services exchange data, manage requests and responses, and handle different scenarios. Discuss any architectural decisions related to real-time communication, event-driven architectures, or other patterns.

In summary, the "Application Architecture" section is about defining and illustrating the architecture of the integrated system, highlighting its key components, their interactions, and communication patterns. Clear documentation of the application architecture is essential for effective project understanding, development, and future maintenance.

Here's a sample "Application Architecture" section for your university system project:

APPLICATION ARCHITECTURE

The Application Architecture component of our university system project delves into the design and structure of software applications that make up the integrated system. This section outlines the key components, communication patterns, and design principles that guide the development of the application architecture.

COMPONENTS OF APPLICATION ARCHITECTURE

The application architecture of our university system comprises several key components, each serving a specific purpose in the overall functionality:

1. **User Interface (UI):** The UI component includes web interfaces, mobile applications, and user dashboards that enable students, faculty, and administrators to interact with the system.
2. **Business Logic:** This component houses the core logic of the system, including enrollment processes, grading, faculty assignments, and financial transactions.
3. **Database Management System (DBMS):** The DBMS stores and manages data related to student records, course offerings, faculty details, and financial transactions.
4. **Application Programming Interface (API):** APIs provide the means for different system modules to communicate and exchange data, ensuring seamless integration.

APPLICATION ARCHITECTURE DIAGRAM

To visually represent the application architecture, we create architecture diagrams that illustrate the relationships and interactions between the components. These diagrams help in understanding how data flows, how user interactions are processed, and how different modules communicate with each other.

APPLICATION ARCHITECTURE DIAGRAM: AN EXAMPLE

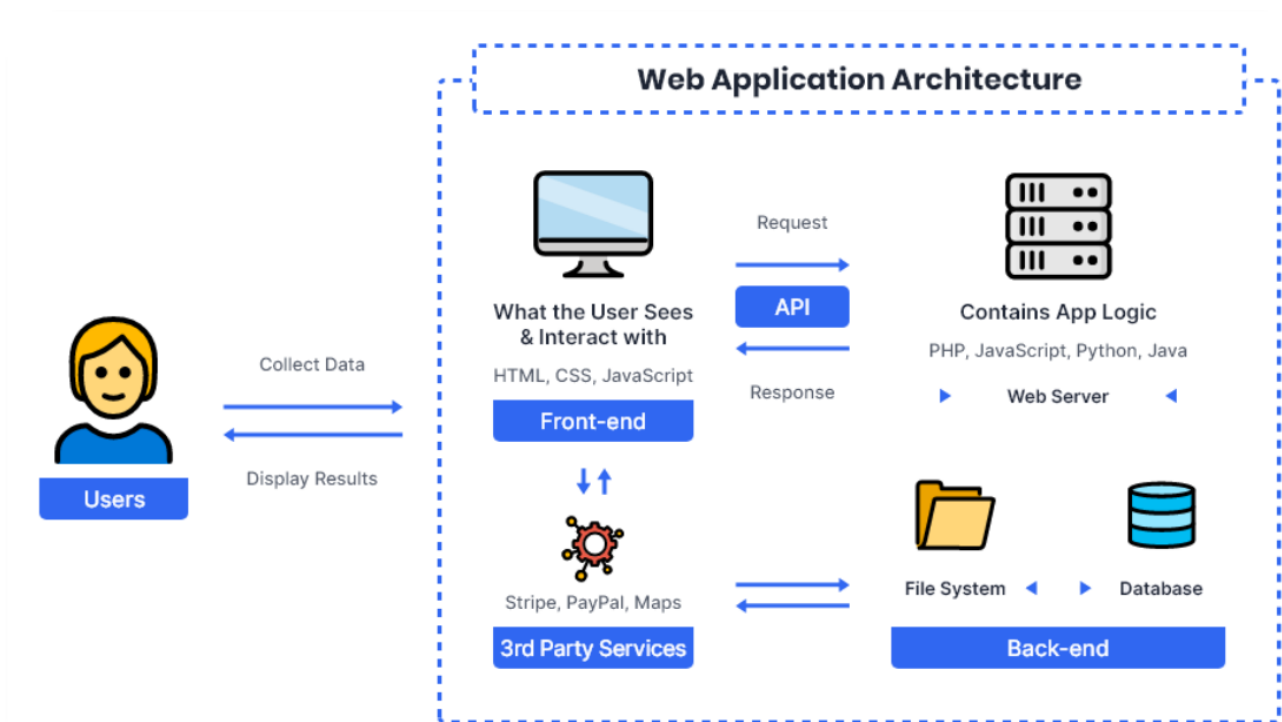


Figure 11. Web Application Architecture Diagram

INTEGRATION OF SOFTWARE MODULES

In a university system, various software modules need to work together seamlessly. Integration is achieved through the use of APIs and middleware. For instance, when a student registers for a course, the UI communicates with the business logic, which in turn interacts with the database and updates student records.

COMMUNICATION AND INTERACTION PATTERNS

Efficient communication and interaction patterns are essential to the application's performance and user experience. In our application architecture, we adopt the following patterns:

1. **REQUEST-RESPONSE:** Most user interactions involve sending a request to the system, which responds with the necessary data or performs the required action. For example, a student requests course registration, and the system responds with a confirmation.
2. **PUBLISH-SUBSCRIBE:** For real-time updates, we implement a publish-subscribe pattern. This is particularly useful for notifications about course changes, grade updates, and important university announcements.
3. **BATCH PROCESSING:** Certain tasks, such as end-of-semester grade calculations or financial reports, are processed in batches at specific intervals to minimize system load during peak times.

SCALIBILITY AND PERFORMANCE CONSIDERATIONS

To ensure that the application architecture is scalable and performs efficiently, we consider the following factors:

1. **LOAD BALANCING:** Load balancers distribute incoming traffic across multiple servers to prevent overloading and ensure consistent performance.
2. **CACHING:** Caching frequently accessed data reduces database load and speeds up responses to user requests.
3. **DATABASE INDEXING:** Proper database indexing and optimization techniques are applied to enhance data retrieval and storage efficiency.
4. **DATA STORAGE STRATEGIES:** Data is stored and organized in a way that ensures fast access and retrieval, particularly for student records and academic information.

In summary, the Application Architecture section outlines the composition of the software applications within our university system project. It provides insights into the components, their interactions, integration mechanisms, and considerations for scalability and performance. This architecture serves as the foundation for building a robust and user-friendly university system.

4.4.7 Part 7- DATA ARCHITECTURE

DATA SOURCES AND TYPES

Begin by identifying and describing the data sources used in the integrated system. These sources may include databases, external APIs, file systems, or other data stores. Detail the types of data they store, such as structured (relational databases), semi-structured (JSON or XML), or unstructured (documents, images). Explain the importance of identifying data sources for the project.

DATA FLOW DIAGRAMS

Create data flow diagrams that visually represent how data moves through an integrated system. Use standardized notation such as data flow diagrams (DFDs) or other appropriate visualization techniques. These diagrams provide an overview of how data is processed, transformed, and exchanged between different components of the system.

[illegible]

Describes the architecture and methods used to store and manage data in an integrated system.

Synchronizing data between systems Explains how data is synchronized between integrated systems to maintain data consistency and integrity. Discusses the mechanisms, protocols, and tools used for data synchronization. Addresses all applicable data mapping and transformation processes to ensure compatibility between systems. Emphasize the importance of synchronization for seamless data sharing and decision making.

The “Data Architecture” section aims to define and illustrate how data is managed, stored, and synchronized in the integrated system. Accurate documentation of data sources, types, data flows, storage and synchronization is essential to ensure the quality, consistency and accessibility of data that is essential for success of project.

Here's a sample content for the "Data Architecture" section of your capstone project documentation, assuming that the project is a university system:

Data Sources and Types

Our university system relies on a variety of data sources and types to support its functionality and operations. These data sources include:

1. STUDENT INFORMATION DATABASE

This database contains comprehensive information about students, including personal details, enrollment status, academic records, and course registrations.

2. FACULTY DATABASE

The faculty database stores data related to faculty members, including personal information, academic qualifications, courses they are teaching, and workload information.

3. COURSE CATALOG DATABASE

This database contains information about courses offered by the university, including course descriptions, prerequisites, schedules, and associated faculty members.

4. Administrative Data

Administrative data sources provide information on university resources, budgets, and various administrative processes.

5. External Data Sources

In addition to internal data, the system interfaces with external sources, such as educational content providers, for course materials and textbooks.

Data types in our system include structured data, such as student records and course information, semi-structured data like JSON files used for configuration, and unstructured data, including documents, images, and multimedia content used for educational resources.

DATA FLOW DIAGRAMS

Data flow diagrams illustrate how data moves within the university system. The diagrams show the flow of data from various sources to different components of the system and how data is processed and stored. They highlight key processes, data stores, data flows, and external entities, providing a visual representation of the system's data architecture.

DATA STORAGE AND MANAGEMENT

Data storage and management in our university system are critical for ensuring data accuracy, security, and accessibility. We utilize a PostgreSQL relational database management system (RDBMS) to store structured data, such as student and faculty records, course information, and administrative data. Data is organized in tables with defined relationships and constraints to maintain data integrity.

For semi-structured and unstructured data, such as configuration files and multimedia content, we employ a distributed file storage system that allows efficient storage and retrieval. This system accommodates data scalability and ensures redundancy and backup.

Data management practices include regular data backup and disaster recovery planning to safeguard against data loss and ensure business continuity.

DATA SYNCHRONIZATION ACROSS SYSTEMS

Data synchronization is critical to maintaining data consistency across different modules of our university system. To achieve this, we have implemented a data integration layer that synchronizes data changes between the various databases and systems. Data synchronization occurs in real-time, ensuring that all modules have access to the most up-to-date data. In cases where real-time synchronization is not feasible, scheduled batch processes are used to update data at regular intervals.

Data transformation and mapping processes are employed to ensure data compatibility and consistency between systems. This includes resolving data conflicts, standardizing data formats, and mapping data from one source to another.

In summary, our data architecture section highlights the importance of data sources and types in the university system, data flow diagrams to visualize data movement, data storage and management practices, and data synchronization methods. This architecture ensures that data remains accurate, accessible, and consistent across the system, providing a solid foundation for our university operations.

4.4.8 Part 8 -TECHNOLOGY ARCHITECTURE

TECHNOLOGY ARCHITECTURE

Technology stack and infrastructure: Start by describing the technology stack used in your project, including hardware and software components . Details the operating systems, programming languages, frameworks and libraries used. Discuss hardware infrastructure, including servers, storage, and any cloud services or virtualization technology. Explain the reasons for choosing this technology and infrastructure.

TECHNOLOGY ARCHITECTURE: AN EXAMPLE

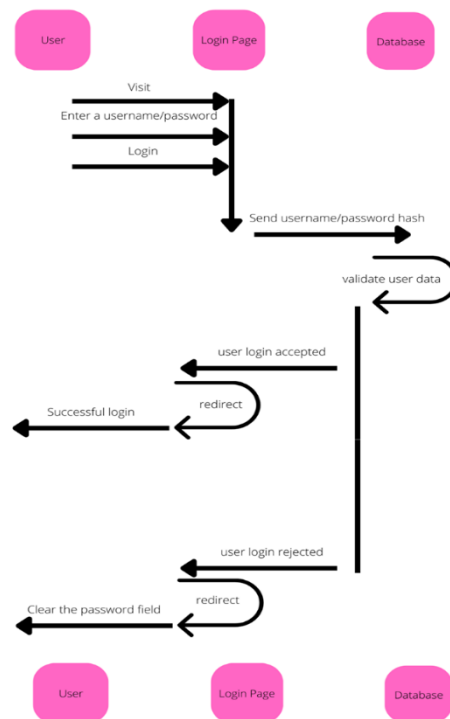


Figure 13. Technology Architecture for myPUPQC

NETWORK ARCHITECTURE AND CONFIGURATION

Introduces network architecture and configuration. Describes the network topology used, such as client-server, peer-to-peer, or hybrid. Explain network protocols, communication models, and any security measures in place. Discuss any load balancing or failover mechanisms implemented to ensure network reliability and performance.

SOFTWARE TECHNOLOGY

Details of software technologies used in integrated systems. This includes web servers, databases, middleware, and any other software component. Explain how these technologies contribute to the project goals and how they interact with each other. Discusses all considerations related to software licensing, version control, and dependencies.

SCALABILITY AND PERFORMANCE CONSIDERATIONS

Addresses the scalability and performance aspects of technology architecture. Explain how the system can scale to handle increased load, such as an increase in the number of users or data volume. Discusses strategies for optimizing system performance, including caching, indexing, and code optimization. Details any benchmarking or performance tests performed to validate the system's responsiveness.

In summary, the “Technology Architecture” section aims to define and explain the technology stack, infrastructure, and scalability and performance considerations in the integrated system.

A well-documented technology architecture ensures that the system is built on a solid foundation and can meet project and user needs.

Here's a sample content for the "Technology Architecture" section of your capstone project documentation, assuming that the project is a university system:

TECHNOLOGY STACK AND INFRASTRUCTURE

The technology architecture of our university system is built on a well-defined technology stack and infrastructure. We utilized a combination of open-source and industry-standard technologies to ensure scalability, security, and performance. Our technology stack includes:

1. **Operating System:** We deployed the system on Linux-based servers to ensure stability and security.
2. **Programming Languages:** Python was chosen as the primary programming language for backend development, while JavaScript was used for frontend development.
3. **Frameworks:** We leveraged the Django web framework for building the backend, ensuring rapid development, security, and modularity. React.js was chosen for the front end, providing a dynamic and responsive user interface.
4. **Database Management:** PostgreSQL, a powerful and open source relational database, is used to store and manage data.
5. **Web Server :** Gunicorn and Nginx are used to serve the application and handle HTTP requests.
6. **Cloud Infrastructure:** The system is hosted on Amazon Web Services (AWS) to ensure scalability, reliability, and ease of maintenance. AWS services, such as Amazon EC2 and Amazon RDS instances, are used to provide powerful database and storage capabilities.

NETWORK TOPOLOGY AND CONFIGURATION

Our network topology followed a client-server architecture, where clients (users) interacted with the system hosted on the server. Network configuration included:

1. Secure Socket Layer (SSL) encryption to secure data transmission.
2. Load balancing to distribute traffic across multiple server instances for improved performance and fault tolerance.
3. Firewall rules to control network access and protect against security threats.

SOFTWARE TECHNOLOGIES

We employed various software technologies to enhance the functionality and user experience of the university system:

1. Django REST framework: This powerful framework was used for building robust APIs that allowed seamless data interaction between the frontend and backend.
2. React.js: A popular JavaScript library, React.js has made creating dynamic and responsive user interfaces easy.
3. Redux: This state management library is used to manage complex state and data flows in the application.
4. Docker: containers are used to isolate and deploy applications, ensuring consistency between development and production environments.
5. Jenkins: This CI/CD tool is used for automated testing, builds, and deployments, streamlining our development and deployment processes.

SCALABILITY AND PERFORMANCE CONSIDERATIONS

To ensure scalability and performance of the university system, we have implemented the following considerations:

1. Dimensional scaling horizontal: The system is designed to be horizontally scalable, allowing us to add more server instances to accommodate increased user numbers.
2. Caching Mechanism: We have implemented caching to reduce database load and improve response time.
3. Database indexing: To optimize database queries, we have created appropriate indexes.
4. Performance Monitoring and Testing: Continuous performance monitoring and testing allows us to identify and resolve performance bottlenecks and ensure optimal system performance.

This section provides insight into the technological architecture that underpins our university system. It describes the technology stack, network configuration, software technology, and scalability and performance considerations. Our choice of technologies and infrastructure was carefully selected to support the system's robustness, security, and scalability.

4.4.9 Part 9 – DEVELOPMENT PROCESS

AGILE SCRUM ROLES AND RESPONSIBILITIES

Discuss the roles and responsibilities of Agile Scrum team members, including the Scrum Master, Product Owner, and Development Team. Explain the duties and expectations of each role, such as the Scrum Master's responsibilities in supporting the Scrum process and the Product Owner's role in managing the

product backlog. Emphasize the importance of collaboration and communication within the team.

SPRINT PLANNING AND BACKLOG MANAGEMENT

Explains how to conduct sprint planning meetings, including choosing which product backlog items will be included in a sprint withdraw. Discuss the process of setting sprint goals and creating a sprint backlog. Emphasize the importance of prioritizing and estimating user stories or tasks for planning purposes.

SPRINT EXECUTION AND DELIVERABLES

Describes the execution of sprints, including daily stand-up meetings (or daily Scrums), tracking sprint progress, and Complete the sprint deliverables. Discuss the development and testing activities that take place during each sprint. Present all sprint review meetings where finished features are demonstrated and feedback is collected. Emphasize the importance of incremental development and delivery of possible growth.

CHALLENGES ENCOUNTERED DURING DEVELOPMENT

Address any challenges or obstacles encountered during development. This may include issues related to scope changes, resource constraints, technical difficulties, or team dynamics.

Explains how to identify, manage and resolve these challenges. Discuss lessons learned and improvements made in the face of these challenges.

In summary, the “Development Process” section is intended to document the methodology, roles, activities, and challenges that are part of the project development process.

It provides an overview of how to apply the Agile Scrum framework and how teams manage projects from planning to execution. This section helps

stakeholders understand the project's progress and the team's adaptability to meet challenges.

Here is a sample content for the “Development Process” section of your capstone project document, assuming the project is an academic system:

ROLES AND RESPONSIBILITIES OF AGILE SCRUM

The development of our university system follows an Agile Scrum Framework, ensuring collaboration, adaptability and iterative progress. The roles within our Scrum team are clearly defined, with the Scrum Master responsible for supporting the Scrum process and removing impediments. The product owner plays an important role in identifying and prioritizing the features needed for the system. The development team consists of software engineers responsible for designing, coding, and testing various modules.

SPRINT PLANNING AND BACKLOG MANAGEMENT

Our projects are organized into Sprints, which typically last two weeks. Sprint planning meetings are held at the beginning of each Sprint to select product backlog items and define the Sprint Backlog. Prioritize user stories and tasks based on stakeholder needs and project goals.

Sprint backlog details the work to be completed during the sprint, ensuring transparency and alignment with project goals.

SPRINT EXECUTION AND DELIVERABLES

During each sprint, daily stand-up meetings (Daily Scrum) are held to monitor progress and remove possible impediments. The development team worked collaboratively to implement the features outlined in the sprint backlog. Incremental development is the key principle, with deliverable increments delivered at the end of each sprint. Our sprint reviews allow stakeholders to

provide feedback and validate completed work, ensuring that the project meets their expectations.

CHALLENGES ENCOUNTERED DURING DEVELOPMENT

Throughout the development process, we encountered a number of challenges.

Scope changes occurred due to growing stakeholder demands, requiring adjustments to our sprint plans and timelines. These changes are managed through effective communication and change management strategies. Technical challenges included integrating various modules within the system, which required careful planning and testing. Maintaining a balance between feature development and technical debt was also a recurring challenge that we addressed through continuous refactoring and code reviews. These challenges, while demanding, contributed to the team's growth and adaptability.

This section provides insights into how our university system was developed following an Agile Scrum framework. It outlines the roles and responsibilities of team members, the sprint planning and execution process, and how challenges were identified and addressed throughout the project. The iterative and collaborative nature of Agile Scrum facilitates responsive development, ensuring the project is aligned with the needs of stakeholders.

4.4.10 Part 10 – IMPLEMENTATION

The “Implementation” section of your capstone project document focuses on the technical aspects of how the project is actually developed and implemented. Below is a complete discussion of each aspect:

TECHNICAL IMPLEMENTATION DETAILS

Provides a detailed overview of how the project is implemented technically. This can include specific design patterns, coding methods, and software engineering principles used. Explain the technical architecture of an integrated system, including how data is transferred, the relationships between different components, and the high-level design of the software.

TOOLS AND TECHNOLOGIES USED

List and discuss the tools, technologies, and development environments used during the implementation phase. This will include everything from programming languages and frameworks to development and deployment tools. Explain why each tool or technology was chosen and how it contributes to successful project delivery.

CODE INTEGRATION AND INTEROPERABILITY

Details how to integrate different software modules or information systems developed by the class. Discuss the interoperability challenges and strategies used to ensure different components can work seamlessly together. Describe how to manage the code integration process, including version control and code reviews.

INTEGRATION TESTING AND DEBUGGING

Explains the testing strategies and techniques used to ensure that an integrated system functions correctly. Describe the types of testing performed, such as unit testing, integration testing, and end-to-end testing. Discuss test results and how to identify, document, and resolve problems or errors. Highlights debugging processes and best practices used to resolve issues.

In summary, the “Implementation” section provides an in-depth analysis of the technical

aspects of how the project is built and how the various components are integrated.

It demonstrates the team's technical expertise and the strategies used to ensure that the integrated system functions correctly and as intended. This section also highlights the tools, technologies, and testing procedures needed to successfully execute the project.

Here is a sample “Implementation” documentation in the Capstone project:

TECHNICAL IMPLEMENTATION DETAILS

The technical implementation of our university system involves the development of several modules to address various aspects of administration university administration. These modules are designed using a combination of programming languages and frameworks. The system's technical architecture follows a tiered approach, with a presentation layer, application layer, and data layer. The presentation layer is built using HTML, CSS, and JavaScript, while the application layer leverages the Python programming language with the Django web framework.

The data layer is based on the PostgreSQL relational database for data storage.

TOOLS AND TECHNOLOGIES USED

Our project uses a variety of development tools and technologies to support the implementation phase. These include text editors for code development, version control systems like Git for collaborative coding, integrated development environments (IDEs) for Python, and continuous integration (CI) tools like Jenkins for automated testing and deployment. The project is hosted on cloud-based infrastructure, specifically Amazon Web Services (AWS), which provides powerful scalability and storage.

CODE INTEGRATION AND INOPERABILITY

To ensure smooth code integration, we follow a modular and loosely coupled architecture.

Different modules, such as student information management and teacher lesson planning, are developed as separate components that communicate through clearly defined APIs. This approach allows us to maintain the independence of each module and ensure that changes in one area do not negatively affect other areas. Additionally, we used Git for version control, allowing us to manage code changes, track development progress, and collaborate effectively.

INTEGRATION TESTING AND DEBUGGING

During the implementation phase, extensive integration testing was performed to verify the interoperability of the various modules in the system. We performed automated and manual testing, including unit testing for each individual module and end-to-end testing to ensure correct data flow between components. Automated testing frameworks were used to continuously monitor code quality and integration. Once issues are identified, detailed debugging procedures are followed and code changes are rigorously tested to ensure errors are resolved.

This section provides insight into the technical aspects of the project's implementation, highlighting the tools, technologies, and practices used to create the university system.

The modular approach to code development and the comprehensive testing processes were key to ensuring a robust and scalable system. The project's hosting on AWS facilitated easy scalability and reliability.

4.4.11 Part 11 – TESTING AND QUALITY ASSURANCE

The "Testing and Quality Assurance" section in your capstone project documentation is essential for ensuring that the integrated system functions as expected and meets quality standards.

Here's a comprehensive discussion of each aspect

Testing and Quality Assurance: Testing Strategies and Methodologies: Describe the testing strategies and methodologies used in the project. Discuss the overall approach to testing, such as whether you followed a test-driven development (TDD) approach, conducted manual or automated testing, or used a combination of these methods. Explain the rationale behind the chosen strategy and how it aligned with the project's goals.

TEST CASES AND TEST DATA

Detail the test cases developed to validate the integrated system's functionality. Explain how these test cases are derived from requirements, user stories, or use cases.

Includes sample test cases to illustrate format and structure. Discusses the test data used, including expected results and special cases to verify system operation under various conditions.

TEST RESULTS AND BUG REPORTS

Presents test results, including successful test cases and any problems or defects encountered.

Discuss how to identify, record, and track errors. Includes bug reports with detailed information about each issue, such as its severity, reproduction steps, and affected components.

Explains the error resolution and prioritization process.

QUALITY ASSURANCE MEASURES

Explains the quality assurance measures taken to ensure the project meets quality standards.

Discuss code reviews, peer testing, and any other quality control processes.

Highlight any quality assurance tools or measures used to monitor project progress and quality.

Explain how the project will achieve its quality goals and correct any deficiencies.

In summary, the “Testing and Quality Assurance” section aims to demonstrate how rigorously the project was tested and how quality standards were maintained throughout the development process.

It presents the test methods, test cases and measures taken to ensure that the system meets the defined quality criteria.

Additionally, it demonstrates the team's commitment to delivering a high-quality integrated system.

Here is a sample Capstone Project Documentation for this part:

sample:

TESTING AND QUALITY ASSURANCE:

TESTING STRATEGIES AND METHODOLOGIES:

For our capstone project, we followed a comprehensive testing strategy that encompassed both manual and automated testing approaches. Our testing process was aligned with the Agile Scrum methodology, where testing was an integral part of each sprint cycle. Test-driven development (TDD) principles guided our approach, with unit tests being written before code implementation. We also performed system, integration, and acceptance testing as part of our iterative

development process. This strategy allowed us to identify and address issues early in the development lifecycle, resulting in a more robust and reliable system.

TEST CASES AND TEST DATA:

We developed a wide range of test cases to ensure that our integrated system met its requirements and performed as expected. Test cases were derived from user stories and use cases. Each test case was documented with clear steps, expected results, and criteria for success. As an example, one of our test cases ensured that user authentication worked correctly, and it included steps for entering valid and invalid credentials to verify that access control was functioning as intended. In addition to regular test data, we included edge cases to challenge the system's behavior under extreme conditions.

TEST RESULTS AND BUG REPORTS:

Throughout our testing process, we maintained detailed records of our test results and bug reports. Test results were documented for each test case, indicating whether the test passed or failed. In the case of failures, we logged bug reports that included information such as the severity of the issue, steps to reproduce, and the component affected. An example of a bug report involved a UI glitch that was categorized as a minor issue but had a visual impact on the user experience. These reports were tracked in our issue management system, and their resolution was prioritized based on severity.

QUALITY ASSURANCE MEASURES:

Our quality assurance measures extended beyond testing to include code reviews and continuous integration (CI). Code reviews were conducted regularly, allowing team members to provide feedback and ensure that the code adhered to coding standards and best practices. This process not only improved code quality but also facilitated knowledge sharing within the team. We utilized CI tools to automate

the build and testing process, ensuring that code changes did not introduce regressions. Quality metrics, such as code coverage and code complexity, were monitored to maintain a high level of code quality.

This section demonstrates our commitment to ensuring that our integrated system met the desired quality standards. Our testing and quality assurance efforts were integral to delivering a reliable and high-quality solution to our stakeholders.

4.4.12 Part 12 – RESULTS AND EVALUATION

The "Results and Evaluation" section of your capstone project documentation is essential for providing a comprehensive view of the project's outcomes and assessing its alignment with the initial objectives. This section also includes feedback from stakeholders and users, as well as lessons learned during the project's execution. Here's a detailed discussion of each aspect:

PROJECT OUTCOMES AND DELIVERABLES

Begin by summarizing the key project outcomes and deliverables. This should include a list of features or components developed, system functionality, and any documentation or artifacts produced as part of the project. Be specific and provide a clear overview of what was achieved during the project's lifecycle.

ALIGNMENT WITH PROJECT OBJECTIVES

Evaluate how well the project aligned with its initial objectives and goals. Assess whether the project successfully addressed the identified problem statement and fulfilled its intended purpose. Discuss how the project's outcomes contribute to the organization or domain's needs and whether they align with the expectations set at the beginning of the project.

STAKEHOLDER AND USER FEEDBACK

Include feedback from stakeholders and end-users who interacted with the integrated system. This feedback can be in the form of surveys, interviews, or usability testing results. Present both positive feedback and areas where improvements or adjustments may be necessary. Discuss how this feedback was collected, analyzed, and incorporated into the project.

Lessons Learned:

Share insights and lessons learned from the project's execution. Discuss what worked well, what challenges were encountered, and how those challenges were addressed. Consider aspects related to project management, technical development, communication, and team dynamics. Lessons learned should serve as valuable insights for future projects and contribute to continuous improvement.

In conclusion, the "Results and Evaluation" section is about summarizing the project's outcomes, assessing its alignment with initial objectives, and providing feedback from stakeholders and users. Lessons learned are valuable takeaways that contribute to the team's growth and improvement in future projects. This section serves as a reflective and forward-looking part of your capstone project documentation, showcasing the project's impact and the valuable knowledge gained from its execution.

Here's a sample content for the "Results and Evaluation" section of your capstone project documentation, assuming that the project is a university system:

PROJECT OUTCOMES AND DELIVERABLES

The development of our university system has yielded a range of outcomes and deliverables that significantly impact the efficiency and management of university operations. Key project deliverables include the successful implementation of a student information management module, a faculty course scheduling system, and an administrative dashboard. Additionally, comprehensive user manuals and technical documentation were provided to ensure effective system management and maintenance.

ALIGNMENT WITH PROJECT OBJECTIVES

Our project's objectives were centered around improving the management of student records, streamlining faculty course scheduling, and enhancing administrative decision-making within the university. We are delighted to report that our project outcomes align seamlessly with these objectives. The student information management module has greatly simplified the enrollment process, record-keeping, and reporting, reducing the administrative workload. The faculty course scheduling system has optimized course allocation and faculty workload management, significantly enhancing academic operations. The administrative dashboard empowers decision-makers with real-time data visualization, enabling data-driven insights. Our project has successfully addressed the identified challenges and met its primary objectives.

STAKEHOLDER AND USER FEEDBACK

To assess the impact of our university system, we actively gathered feedback from key stakeholders, including university administrators, faculty members, and students. This feedback was collected through surveys, interviews, and usability testing. The feedback indicates that the system has had a positive impact on the university community. University administrators have reported a significant

reduction in administrative tasks, which has allowed them to focus on strategic decision-making. Faculty members have praised the faculty course scheduling system for its user-friendly interface, making course allocation more efficient. Students have expressed satisfaction with the ease of accessing academic records and the streamlined enrollment process. While the feedback was predominantly positive, we have identified opportunities for further improvements, such as additional features to support faculty collaboration and enhanced student self-service capabilities. We are committed to addressing these suggestions in future iterations.

LESSONS LEARNED

Throughout the project's lifecycle, we encountered and successfully addressed various challenges. These challenges included scope changes driven by evolving stakeholder needs, requiring us to adapt our project plans and timelines accordingly. The effective management of these changes was facilitated through clear and open communication with stakeholders. Technical challenges, such as integrating various modules within the system, were successfully overcome through careful planning and testing. We also learned the importance of maintaining a balance between feature development and technical debt, which was addressed through continuous refactoring and rigorous code reviews. These lessons have enriched our understanding of project management and software development, equipping us with valuable insights for future projects.

In conclusion, this "Results and Evaluation" section demonstrates the positive impact of our university system project on university operations and the university community. The feedback from stakeholders and users provides valuable insights for future enhancements. The lessons learned underscore our commitment to

continuous improvement and the application of best practices in future endeavors.

4.4.13 Part 13 – RESULTS AND EVALUATION

PROJECT RESULTS AND DELIVERABLES

The development of our university system has delivered a comprehensive range of results and deliverables. These include implementing a student information management module, a course planning system for teachers, and an administration dashboard. In addition, we have provided detailed user manuals, technical documentation and complete user interface design. These products together provide a solid foundation to enhance the overall efficiency and management of university operations.

ALIGNS WITH PROJECT GOALS

Our project goals are to improve student records management, streamline faculty course planning, and improve decision making administrative arrangements in universities. We are pleased to report that our results are effectively consistent with these goals. The student information management module has simplified registration, record keeping and reporting. The department's course planning system optimized course assignments and faculty workload management. The admin dashboard supports data-driven decision making through real-time data visualization. The project was successful in solving the identified problems and achieving its main objectives.

STAKEHOLDER AND USER FEEDBACK

To assess the impact of our university system, we gathered feedback from key stakeholders, including administrators universities, lecturers and students. Feedback from administrators highlights improved efficiency in managing student records, helping to reduce administrative workload. Faculty members praise the

Faculty Course Planning System for its user-friendly interface and improved course assignments. Students appreciate the easy access to their academic records and the smooth registration process. While feedback has been largely positive, we have also identified areas for further improvement, such as additional features to support faculty collaboration and expanded self-service options for students.

LESSONS LEARNED

Throughout the project lifecycle, we encountered and resolved a number of challenges.

These challenges include changes in scope that require adjustments to project schedules and resource allocation. Collaborative communications and change management are tools to help mitigate these challenges. We also know the importance of clear documentation, code versioning, and comprehensive testing to ensure project quality and stability. These lessons have enriched our understanding of project management and software development, providing us with valuable insights for future projects.

This “Results and Evaluation” section illustrates how our university system project achieved its goals and produced tangible results. It also highlights the project's impact on stakeholders and users, and notes areas for improvement. Lessons learned highlight the valuable knowledge gained during the project and its application to future endeavors.

Here's a sample "Conclusion" section for your university system project:

The development of a comprehensive university system is a complex and multifaceted endeavor that requires careful planning, strategic alignment, and a deep understanding of the intricate operations of higher education institutions. This project has been guided by a commitment to improving efficiency, enhancing

user experiences, and supporting the core mission of our university. In this concluding section, we reflect on the journey of this project and highlight key takeaways.

PROJECT OBJECTIVES ATTAINED

Throughout the project's lifecycle, our primary objectives were to create a university system that streamlines administrative processes, supports faculty and students, and aligns with the strategic goals of our institution. We are pleased to report that we have successfully met these objectives.

The implementation of an Agile Scrum methodology enabled us to maintain flexibility and adaptability, ensuring that the system remained responsive to the evolving needs of the university community. The integration of information systems provided a unified platform for efficient data exchange, reducing redundancy, and improving data accuracy. Adherence to The Open Group Architecture Framework (TOGAF) principles helped us establish a well-structured and aligned architecture across four domains: Business Process, Application, Data, and Technology.

STAKEHOLDER SATISFACTION

Our engagement with key stakeholders, including university administrators, faculty, and students, has been integral to the project's success. Regular communication, feedback sessions, and user testing allowed us to tailor the system to their specific needs. As a result, we have received positive feedback from our stakeholders regarding the system's usability and effectiveness in addressing their requirements.

LESSONS LEARNED

The development of this university system has not been without its challenges and learning opportunities. We have encountered technical hurdles, coordination complexities, and the need for continuous adaptation to changing requirements. These challenges, however, have provided valuable lessons in project management, software development, and stakeholder collaboration.

FUTURE ENHANCEMENTS

While the university system has been successfully implemented, our journey does not end here. We recognize the need for continuous improvement and adaptation to changing technologies and educational practices. To that end, we have outlined some potential areas for future enhancements:

1. **Data Analytics:** Implementing advanced data analytics and reporting capabilities to provide insights for informed decision-making.
2. **Mobile Accessibility:** Expanding the system's accessibility through mobile applications to meet the preferences of an increasingly mobile-centric user base.
3. **Security Enhancements:** Continuously enhancing data security measures to safeguard sensitive information and protect against potential threats.

ACKNOWLEDGEMENTS

The successful completion of this project would not have been possible without the unwavering support and dedication of our project team, university stakeholders, and the broader community. We express our sincere appreciation to everyone who contributed to the project's success.

4.4.13 Part 13 - CONCLUSION

In closing, the development of the university system represents a significant milestone in our commitment to improving administrative processes, enhancing educational experiences, and aligning with our university's strategic objectives. The project's adherence to Agile Scrum principles, enterprise architecture best practices, and active stakeholder engagement has resulted in a system that empowers our university community and paves the way for future enhancements. We look forward to the continued evolution and success of our university system, shaping a brighter future for our institution and its community.

4.4.14 Part 14 – REFERENCES

LIST OF SITED SOURCES AND REFERENCES

Here's a sample "References" section for your university system project. Please note that you should format the references according to our school's preferred citation style (e.g., APA, MLA, Chicago, etc.):

References:

1. Beck, K., et al. (2001). Manifesto for Agile Software Development. Agile Alliance.
2. Brown, M., et al. (2019). Automation and Efficiency in Higher Education Administration: A Case Study. Journal of Higher Education Management, 41(3), 55-72.
3. Cohn, M. (2010). Succeeding with Agile: Software Development Using Scrum. Addison-Wesley.
4. Cockburn, A. (2001). Agile Software Development. Addison-Wesley.

5. Davis, A., & Patel, S. (2020). Challenges and Benefits of Information Systems Integration in Modern Organizations. *Journal of Enterprise Information Management*, 33(5), 677-695.
6. Johnson, L., & Clark, A. (2019). Improving Student Enrollment and Records Management: A Case Study of Higher Education Institutions. *Journal of Higher Education Technology*, 39(4), 67-82.
7. Lapalme, J. (2001). *Enterprise Architecture: A Disciplined Approach*. Prentice Hall.
8. Martinez, R., et al. (2018). Faculty Workload Management in Higher Education: Challenges and Opportunities. *Journal of Educational Administration*, 38(2), 125-140.
9. Ross, J. W., & Weill, P. (2010). *Enterprise Architecture as Strategy: Creating a Foundation for Business Execution*. Harvard Business Review Press.
10. Ross, J. W., Weill, P., & Robertson, D. C. (2006). *Enterprise Architecture as Strategy: Creating a Foundation for Business Execution*. Harvard Business Press.
11. Schwaber, K., & Sutherland, J. (2017). *The Scrum Guide*. Scrum.org.
12. Schwaber, K. (2004). *Agile Project Management with Scrum*. Microsoft Press.
13. Smith, A., & Johnson, L. (2018). Managing Complexity in Higher Education: Using Data-Driven Decision Making. *Journal of Higher Education Management*, 37(3), 45-60.
14. Smith, P., & Brown, M. (2017). Modernizing University Systems: Challenges and Opportunities. *Journal of Higher Education Administration*, 39(1), 23-38.
15. The Open Group. (2018). *TOGAF® Standard, Version 9.2*. The Open Group.

These references provide a comprehensive list of sources that have informed the development of the university system project, covering aspects of Agile Scrum

methodology, enterprise architecture, relevant studies, and information system integration within higher education.

4.4.15 Part 15 - APPENDICES

Here's a sample "Appendices" section for your university system project. This section typically includes supplementary materials, such as detailed process diagrams, sample data, user surveys, and any additional information that supports the main content of the document:

Appendix A: Process Diagrams

Figure A.1: Student Enrollment Process Diagram

Student Enrollment Process Diagram

Figure A.2: Academic Administration Process Diagram

Academic Administration Process Diagram

Figure A.3: Faculty Management Process Diagram

Faculty Management Process Diagram

Appendix B: User Surveys

Survey Questionnaire: A copy of the online survey questionnaire distributed to university administrators, faculty, and students to gather requirements and feedback.

[Link to Survey Questionnaire]

Survey Results: Summary of survey responses and data analysis.

[Link to Survey Results]

Appendix C: User Stories and Use Cases

Detailed user stories and use cases for the university system's functionalities.

[What is a User Story? Definition from WhatIs.com \(techtarget.com\)](#)

[Use Cases | Usability.gov](#)

Appendix D: Sample Data

Sample data sets used for system testing and demonstration purposes, including student records, course schedules, and financial transactions.

[data.world](#)

Appendix E: Stakeholder Interview Transcripts

Transcripts of key stakeholder interviews, providing insights into their requirements and expectations.

[Internal Stakeholder Interviews for User Research \(userinterviews.com\)](#)

Appendix F: System Performance Test Results

Detailed results of system performance testing, including response times, load handling, and scalability assessments.

[What is performance testing and how does it work? – TechTarget Definition](#)

Appendix G: Data Security Measures

An overview of the data security measures and protocols implemented to protect sensitive information.

[What is Data Security? Data Security Definition and Overview | IBM](#)

Appendix H: Mobile Application Prototype

Screenshots and descriptions of the mobile application prototype for user accessibility.

[How to Create a Mobile App Prototype \(2023\) - Business of Apps](#)

Appendix I: Adviser Acceptance (Functional)

A copy of the adviser's acceptance letter or email confirming their role in the project.

Appendix J: Panel Evaluation and Signature (plus photo ops during defense)

Capstone Project Guidelines for PUP BSIT Students

Evaluation forms completed by the panel members during the project defense, including their signatures.

Photographs taken during the defense to capture the memorable moments of the presentation.

Appendix K: Capacity Planning (Estimates on Storage Consumption – 5 years)

A detailed report on the capacity planning for the university system, including storage consumption estimates for the next five years.

Appendix L: Pilot Companies' Background with proofs of interviews

Background information about the pilot companies involved in testing the university system.

Proof of interviews, such as transcripts or audio recordings, to validate their feedback.

Appendix M: Cloud Copy of the Codes (1 year validity)

A cloud-based repository link to access the source code of the university system with a validity of one year.

Appendix N: IMRAD Format Summary

A concise summary of the IMRAD (Introduction, Methodology, Results, and Discussion) format used in the project's research and reporting.

Appendix O: Comparison of the EIS to existing EIS's (5 pages)

A 5-page document comparing the newly developed Education Information System (EIS) to existing EISs, highlighting the unique features and advantages of the project.

These appendices provide additional documentation and information related to your university system project, offering readers a comprehensive view of the project's development, research, and validation processes.

Technical Documentation Outline

Part 16. Technical Documentation

15.1. System Architecture:

- Overview of the overall system architecture

- Detailed diagrams illustrating system components and their interactions

- Explanation of component responsibilities and relationships

15.2. Information Systems Integration:

- Description of how the class's information systems were integrated

- Diagrams showing the integration points and data flows

- Data transformation and mapping processes

15.3. Application Design and Development:

- Detailed information about the software modules and components

- Code snippets and code structure

- Algorithms and data structures used

- Libraries and frameworks utilized

15.4. Database Schema and Data Management:

- Database schema design

- Entity-relationship diagrams (ERDs)

- Data modeling and normalization

- CRUD (Create, Read, Update, Delete) operations

15.5. Network Configuration:

- Network topology and configuration details

- Security measures (e.g., firewalls, encryption)

- Protocols and communication methods used

- Load balancing and failover mechanisms

15.6. Deployment and Infrastructure:

- Deployment strategies (e.g., cloud, on-premises)
- Hardware specifications and server configurations
- Configuration management (e.g., scripts, automation tools)
- Scalability and resource optimization

15.7. Security Measures:

- Security protocols and measures implemented
- Authentication and authorization mechanisms
- Encryption and data protection
- Incident response and mitigation procedures

15.8. Testing and Quality Assurance:

- Detailed test cases and scenarios
- Test data and expected outcomes
- Test results and validation against requirements
- Performance testing and optimization efforts

15.9. System Monitoring and Maintenance:

- Tools and techniques for monitoring system health
- Logging and error handling mechanisms
- Scheduled maintenance and updates
- Disaster recovery and backup procedures

15.10. APIs and Integration Points:

- Documentation of APIs used for integration
- API endpoints, methods, and data formats
- How external systems interact with your project

15.11. User Documentation:

- User manuals and guides (if applicable)
- Instructions for system users
- Troubleshooting and FAQs

15.12. Known Issues and Troubleshooting:

- List of known issues and their status

- Troubleshooting steps for common problems
- Contact information for technical support
- 15.13. Version Control and Source Code Repository
 - Description of version control system (e.g., Git)
 - Repository location and access details
 - Branching and merging strategies
- 15.14. DevOps and Continuous Integration/Continuous Deployment (CI/CD):
 - CI/CD pipeline setup and configurations
 - Automation scripts and tools used
 - Deployment strategies and rollback procedures
- 15.15. Licensing and Open Source Libraries:
 - Information about licenses for software and libraries used
 - Credits and attributions for open-source components
- 15.16. Performance Metrics and Monitoring:
 - Metrics collected and monitored
 - Tools and dashboards used for performance analysis
 - Actions taken based on performance data

PART 5 - CAPSTONE PROJECT EVALUATION

The capstone project work is evaluated based on the submissions made, the defense and the final capstone project. The capstone project defense consists of a presentation, a demonstration, software modification and a viva. Through all stages of the capstone project the CPEC should be satisfied that the group/individual has submitted group's/individual's own work and the capstone project objectives have been met as well as the outcome has justified the time spent on the capstone project. Verbal feedback for the capstone project will also be given on that day.

5.1 DEFENSE OF THE CAPSTONE PROJECT

The defense of the capstone project is consist of two stages. Authorship/Mock and Final defense.

The importance of both is the demonstration that the work belongs to the group/individual except where acknowledgements have been made and that the capstone project merits the award of the degree. The final defense stage assesses the group/individual's ability to communicate group's/individual's ideas and work.

Group/individual should be able to convince the CPEC that they have undertook a capstone project that is acceptable at the degree level (e.g. 98 hours of work) and have implemented a substantial software component by himself. The final defense of capstone projects will be done only **once** during an academic year. A group/individual will be allowed to resubmit and re defend the same capstone project again only if so recommended by the CPEC. .

5.1.1 Authorship/Mock and Defense

5.1.1.1 Authorship/Mock

This is conducted by the CPEC. Depending on the need the CPEC will test the group or individual's capability for a "deep dive" on their source codes.

This will assure the school that the students are knowledgeable on the codes. The committee might want to include a report, form, or simply remove some codes, or do some back end manipulation, etc. and will give instructions what to do with the system afterwards. The particular revision is time bounded. Some schools are allowing their students to revise until 30 minutes. Failure to do so, will not let them proceed to the next stage – Final Defense.

5.1.1.2 Final Defense Composition

Each presentation and defense panel is to be composed of one panel chairperson and panel members (2). Such panel will serve as the ultimate judge of capstone project's acceptability and quality. Only the following persons are allowed to be present during the presentation and defense:

- Capstone project group/individual members
- Panel chairperson and members
- BSIT coordinator and/or higher-ranking administrator
- Program secretary
- Presentation and defense liaisons (student)

5.1.1.2 Responsibilities

Punctuality

Tardiness will affect a capstone project member's grade. Members who incur tardiness of more than 15 minutes (from the appointed presentation start time) can be classified as failed.

Preparation

Capstone project members are responsible for preparing all the essential equipment to be used in the presentation and defense. Such equipment must be set up and functional before the scheduled start time of the presentation (do set up during the 10 minute break)

Students must mitigate risks of failures/plan contingencies. Technical difficulties such as software viruses and outright hardware failure are considered to be under their control and can be avoided. Such difficulties are thus not acceptable excuses no to proceed with the presentation and defense as scheduled and according to plan.

Attire

Group/individual members are required to come in corporate attire. This is a strict prerequisite for each member prior to being allowed to participate in the presentation.

Decorum

Students are mandated to abstain from the use of communications devices such as beepers, radios, and mobile phones during the presentation and defense.

The following acts are disallowed and commission of one or more of them shall be ground/s for appropriate disciplinary action:

- Leaving the room without authorization from the panelists
- Gross acts of disrespect through words or gestures, which tend to ridicule one or more panelists
- Direct assault upon panelists
- Making inappropriate remarks regarding the judgment handed down by panelists
- Any action unbecoming of a college student

The grades of students who are deemed guilty of these acts maybe withheld until the issue is resolved.

5.1.1.3 Presentation and Defense Proper

Presentation of the capstone project should be done in about 1 hour and 30 minutes: 25 minutes for group/individual presentation, 50 minutes for Q&A, and 15 minutes for grading (This can become longer or shorter depending on the

prerogative of the CPEC). Group/individual should bear in mind that the majority of the CPEC will not be familiar with the capstone project. Presentation should be clear, understandable and well structured. It should cover all significant aspects of the capstone project. It should be a walkthrough of the capstone project starting from the original objectives to the achieved results. The style and content of the presentation should be appropriate for an academic audience. Visual material should be of high standard and produced using Microsoft PowerPoint/Canva or any other presentation tool and may incorporate a selection of screen shots from software produced by the group/individual. Contingency arrangements should be made to ensure the availability of presentation material. Questions may be asked during the presentation.

Demonstration

Demonstration of the software should be limited to a maximum of 15 minutes (i.e. 10 minutes should be spent with the background prior to demo). Group/individual should take necessary actions to ensure that this part is tested prior to the viva date using the same equipment / environment you intend to use at the defense. The group/individual should confidently demonstrate the operations of the system. The demonstration should include the main aspects of the system. Group/individual should ensure that all aspects of the systems have been pre-tested and all such data brought for the demonstration. Typically, each database table must have a minimum of several records especially so. Questions may be asked during the demonstration. Group/individual is responsible to bring his or her own equipment and not more than 10 minutes will be given to set up the equipment.

Defense Proper

The group/individual will be asked questions (approximately 50 minutes) based on the presentation and demonstration. Group/individual should provide confident and sufficient responses to questions.

Modifications

The group/individual will be required to explain any part of the system code and also perform modifications such as changes to the database structures and reflecting them in the program interfaces; reports etc. and demonstrate the changes. Such changes should be demonstrated within a specified time period of 10 - 30 minute duration. In the event that the revision/s required will take a longer time to effect, the following are the considerations:

- Group/individuals/individual should ensure that they completely understand the revisions required of them by the panelists.
- Group/individuals/individual which garnered a passing grade during the presentation and defense are allowed to approach the panel chair for checking of revisions for a **maximum of three (3)** times only and is given seven (7) days to have the study accepted and approved. If after the third time, there are still some prescribed revisions which have not been satisfied, the group/individual will automatically get a status of “deferred”.
- Group/individuals/individual which garnered a grade withheld status during the presentation and defense will be given a week (7 days) to approach their panelists for checking of revisions. If the revisions have not been satisfied after the 1 week period, the group/individual will

automatically get a status of “deferred”. However, if all revisions are met within the time period, the group/individual will be given a grade of 3.0.

- Group/individuals/individual are required to set appointments with their panel chair / panelists.
“Ambush” consultation sessions will not be entertained.
- Group/individuals/individual must bring with them their copy of the presentation and defense revisions list.

5.2 CAPSTONE PROJECT ASSESSMENT

The capstone project will be initially assessed by the CPEC, as the capstone project components are submitted. The capstone project, capstone project management, technical achievement and defense will contribute to the overall capstone project grade. To pass the capstone project, the group/individual must satisfy the CPEC in all above aspects of the capstone project in the same academic year. A group/individual failing in any one of the above components is deemed to have failed the capstone project module.

A group/individual will have to qualify to be called for the capstone project defense by duly submitting all the capstone project components listed in this document. These submissions must fulfill a minimum acceptance level by the CPEC in terms of the number of pages and relevant content. For example a capstone project should consist of 150-200 pages of main parts and another 50-60 pages of appendices. However, being called for the capstone project presentation does not imply that the group/individual has passed in these components as such submissions must comply with the capstone project objectives and the deliverables.

Deferment and Grading Proper

A deferred final academic capstone project is equivalent to an incomplete grade. A student whose capstone project completion is deferred is given at most one semester's worth of an extension in order to finish the capstone project. Enrollment in the final academic subject for the succeeding semester is not necessary. At the end of the extension, the student will merit either a passing or a failing grade. In the latter case, re-enrollment in the final academic capstone project subject will be required during a subsequent term. Students may qualify for deferment, in lieu of an outright failing grade, in the following situations:

- Revisions required by the presentation and defense panel were not completed in the agreed time.
- Getting a grade of 50 – 69.99% under the PRESENTATION AND DEFENSE Documentation Grade /Systems Documentation Grade and wasn't able to complete the revisions in the agreed time.
- Getting a grade of 50 - 69.99% under the total PRESENTATION AND DEFENSE Score (Documentation / Systems Grade and wasn't able to complete the revisions in 1 week's time.

Grading Explained

The Presentation and Defense (P&D) is broken down into three (3) parts.

- Group/individual Documentation / System Grading
- Group/individual Presentation Grading
- Individual Presentation Grading

Capstone Project Guidelines for PUP BSIT Students

Example 1

Component	Panel Chair	Panel Member1	Panel Member 2
Technical (25%)	80	70	90
Usability (20%)	80	70	80
Screen Design (10%)	80	70	90
Completion (40%)	80	0	70
Average per Panelist	80	42	79.5
Overall Average	$201.5/3=67.17$		
Status	Grade withheld		

Example 2

Component	Panel Chair	Panel Member1	Panel Member 2
Technical (25%)	80	80	90
Usability (20%)	80	0	80
Screen Design (10%)	0	70	90
Completion (40%)	80	80	90
Average per Panelist	72	59	87.5
Overall Average	$218.5/3=72.83$		
Status	Pass 3.0		

Example 3

Component	Panel Chair	Panel Member1	Panel Member 2
Technical (25%)	70	70	70
Usability (20%)	0	70	70
Screen Design (10%)	70	70	70
Completion (40%)	0	0	70
Average per Panelist	24.5	42	70
Overall Average	136.5/3= 45.5		
Status	Failed		

** The criteria given in these examples may differ from the evaluation sheet being used by PUPQC

PART 6 - PITFALLS



Some of the most useful things to know about individual capstone projects are the common pitfalls. Why do some capstone projects go wrong? Here are some of the common causes of failure:

- Choosing/Starting the capstone project too late. Submit your capstone project proposal on time and start the capstone project as soon as you can. The longer you leave it the harder it is to get motivated, especially when all your friends seem to be flying ahead. You should aim to submit all capstone project components as listed under the submission schedule.

Capstone Project Guidelines for PUP BSIT Students

- Failing to meet your adviser regularly. If you arrange a meeting with your adviser, turn up at the agreed time. You gain no sympathy from anyone if you lose contact with your adviser and produce a poor capstone project as a result. Your adviser will be happy to help you but they can do nothing if they are unaware that you are having trouble.
- Failing to plan a fall-back position if the planned work is not completed on time. Try to plan your capstone project in stages so that if things go wrong in a later stage you have an alternative plan to fall back on.
- Trying to satisfy an external customer at the expense of your grades. Do not let any outside interests interfere with your work. The guidance for your capstone project should come from your adviser, not your prospective employer or client. While it is important to satisfy the client you should remember that the capstone project is evaluated to meet the degree requirements. Sometimes client's expectations may be far beyond or far below a degree level capstone project.
- Over/Under Ambition. Try to be realistic about what you can achieve in the time available. A good capstone project requires a lot of input from you and should prove to be technically challenging throughout. At the same time, however, it is better to do a small job well than failing to do a complex job at all. Your adviser will advise you on his or her expectations of the capstone project and this will help you to set your sights accordingly.

CHED CMO no. 25 Series of 2015

 COMMISSION ON HIGHER EDUCATION	<p>Republic of the Philippines OFFICE OF THE PRESIDENT COMMISSION ON HIGHER EDUCATION</p>	 OFFICIAL RELEASE CHED Central Office RECORDS SECTION C.P. Garcia Ave., U.P. Diliman, Q.C.
<p>CHED MEMORANDUM ORDER (CMO) NO. 25 ; Series of 2015</p>		
<p>SUBJECT : REVISED POLICIES, STANDARDS, AND GUIDELINES FOR BACHELOR OF SCIENCE IN COMPUTER SCIENCE (BSCS), BACHELOR OF SCIENCE IN INFORMATION SYSTEMS (BSIS), AND BACHELOR OF SCIENCE IN INFORMATION TECHNOLOGY (BSIT) PROGRAMS</p>		
<p>-----</p> <p>In accordance with the pertinent provisions of Republic Act (RA) No. 7722, otherwise known as the "Higher Education Act of 1994," in pursuance of an outcomes-based quality assurance system as advocated under CMO 46 s. 2012, and by virtue of the Commission en banc Resolution No. 268-2015 dated May 25, 2015 the following policies, standards and guidelines (PSGs) are hereby adopted and promulgated by the Commission.</p>		
<p>ARTICLE I INTRODUCTION</p>		
<p>Section 1</p>	<p>Rationale</p> <p>Based on the Guidelines for the Implementation of CMO 46 s 2012, this PSG implements the "shift to learning competency-based standards/outcomes-based education." It specifies the 'core competencies' expected of graduates of <i>Bachelor of Science in Computer Science (BSCS)</i>, <i>Bachelor of Science in Information Systems (BSIS)</i>, and <i>Bachelor of Science in Information Technology (BSIT)</i>, "regardless of the type of HEI they graduate from." However, in "recognition of the spirit of outcomes-based education and ... of the typology of HEIs," this PSG also provides "ample space for HEIs to innovate in the curriculum in line with the assessment of how best to achieve learning outcomes in their particular contexts and their respective missions ..."</p> <p>The field of computing is ever dynamic; its advancement and development had been rapid and its involvement is a continuous process (O'Brien, 2008). To face the challenges of advancement, the Commission recognizes the need to be responsive to the current needs of the country. It is essential and important that the country's computing capability be continually developed and strengthened to be at par globally.</p> <p>It is the objective of the Commission to develop and promote the Policies, Standards and Guidelines (PSG) for BSCS, BSIS and BSIT, to provide a minimum standard for Higher Education Institutions (HEIs) offering or intending to offer these programs. The PSG is developed with consultations from all stakeholders, from the academe to industry (Sarmiento, 2009).</p> <p>The PSG contains provisions that cultivate the culture of excellence in offering these programs. This is in line with the vision of the Commission to have HEIs produce competent graduates that shall cater to the needs of the industry. The PSG is also designed for all HEIs to exercise their innovativeness and creativity in the development of their curricula in the offering of BSCS, BSIS, and BSIT programs (RA 7722, 1994).</p>	
<p>Higher Education Development Center Building, C.P. Garcia Ave., UP Campus, Diliman, Quezon City, Philippines Web Site: www.ched.gov.ph Tel. Nos. 441-1177, 385-4391, 441-1169, 441-1149, 441-1170, 441-1216, 392-5296</p>		

ARTICLE II AUTHORITY TO OPERATE

Section 2 Government Recognition

All Higher Education Institutions (HEIs) including private HEIs, State Universities and Colleges (SUCs), and Local Universities and Colleges (LUCs) intending to offer BSCS, BSIS, and BSIT must first secure proper authority from the Commission in accordance with this PSG. All HEIs with existing BSCS, BSIS, and/or BSIT programs are required to shift to outcomes-based approach pursuant to this PSG and must inform the Commission of such shift. SUCs and LUCs should likewise strictly adhere to the provisions in these policies, standards and guidelines.

ARTICLE III GENERAL PROVISIONS

Section 3 The succeeding articles provide minimum standards and other requirements and prescriptions. The minimum standards for each program are expressed as minimum sets of desired program outcomes which are given in Article IV Section 6. The Commission designed **sample** curricula to attain such outcomes and these are shown in Article V Section 9. The total number of units for each program is here prescribed as the "minimum unit requirement" under Section 13 of RA 7722. In designing the curricula, the Commission employed curriculum maps which are shown in Article V Section 10 as **sample** curriculum map.

Using a learner-centered/outcomes-based approach, the Commission provided sample curricula delivery methods shown in Article V Section 11. The sample course syllabi given in Article V Section 12 show some of these methods.

Based on the curricula and the means of their delivery, the Commission determined the physical resource requirements for the library, laboratories and other facilities and the human resource requirements in terms of administration and faculty, as indicated in Article VI.

Section 4 The HEIs are allowed to design curricula suited to their own contexts and missions provided that they can demonstrate that the same leads to the attainment of the required minimum set of outcomes, albeit by a different route. In the same vein, they have latitude in terms of curriculum delivery and in terms of specification and deployment of human and physical resources as long as they can show that the attainment of the program outcomes and satisfaction of program educational objectives can be assured by the alternative means they propose.

The HEIs can use the **CHED Implementation Handbook for Outcomes-Based Education (OBE)** and the **Institutional Sustainability Assessment (ISA)** as a guide in complying with Sections 16, 17 and 22 of Article VII, hereof.



This PSG is based on the 10-year basic education system and on the existing General Education (GE) program. It reflects the reform towards outcomes-based education as well as international trends in computer science, information systems and information technology curricula. However, this does not yet include necessary changes as a consequence of the K-12 reform. The latter shall be addressed subsequently.

ARTICLE IV PROGRAM SPECIFICATIONS

Section 5 Program Description

5.1 Degree Name

A. Bachelor of Science in Computer Science (BSCS)

Graduates of this program shall be conferred the degree of **Bachelor of Science in Computer Science (BSCS)**.

B. Bachelor of Science in Information Systems (BSIS)

Graduates of this program shall be conferred the degree of **Bachelor of Science in Information Systems (BSIS)**.

C. Bachelor of Science in Information Technology (BSIT)

Graduates of this program shall be conferred the degree of **Bachelor of Science in Information Technology (BSIT)**.

5.2 Nature of the Field of Study

5.2.1 Bachelor of Science in Computer Science (BSCS)

The BS Computer Science program includes the study of computing concepts and theories, algorithmic foundations and new developments in computing. The program prepares students to design and create algorithmically complex software and develop new and effective algorithms for solving computing problems.

The program also includes the study of the standards and practices in Software Engineering. It prepares students to acquire skills and disciplines required for designing, writing and modifying software components, modules and applications that comprise software solutions.

5.2.2 Bachelor of Science in Information Systems (BSIS)

The BS Information Systems Program includes the study of application and effect of information technology to organizations. Graduates of the program should be able to implement an information system, which considers complex technological and organizational factors affecting it. These include components, tools, techniques, strategies, methodologies, etc.



Graduates are able to help an organization determine how information and technology-enabled business processes can be used as strategic tool to achieve a competitive advantage. As a result, IS professionals require a sound understanding of organizational principles and practices so that they can serve as an effective bridge between the technical and management/users communities within an organization. This enables them to ensure that the organization has the information and the systems it needs to support its operations.

5.2.3 Bachelor of Science in Information Technology (BSIT)

The BS Information Technology program includes the study of the utilization of both hardware and software technologies involving planning, installing, customizing, operating, managing and administering, and maintaining information technology infrastructure that provides computing solutions to address the needs of an organization.

The program prepares graduates to address various user needs involving the selection, development, application, integration and management of computing technologies within an organization.

5.3 Program Goals

The BSCS, BSIS, and BSIT graduates are expected to become globally competent, innovative, and socially and ethically responsible computing professionals engaged in life-long learning endeavours. They are capable of contributing to the country's national development goals.

5.4 Specific Professions/careers/occupations for Graduates

A. Bachelor of Science in Computer Science (BSCS)

Primary Job Roles

- Software Engineer
- Systems Software Developer
- Research and Development computing professional
- Applications Software Developer
- Computer Programmer

Secondary Job Roles

- Systems Analyst
- Data Analyst
- Quality Assurance Specialist
- Software Support Specialist

B. Bachelor of Science in Information Systems (BSIS)

Primary Job Roles

- Organizational Process Analyst
- Data Analyst



- Solutions Specialist
- Systems Analyst
- IS Project Management Personnel

Secondary Job Roles

- Applications Developer
- End User Trainer
- Documentation Specialist
- Quality Assurance Specialist

C. Bachelor of Science in Information Technology (BSIT)

Primary Job Roles

- Web and Applications Developer
- Junior Database Administrator
- Systems Administrator
- Network Engineer
- Junior Information Security Administrator
- Systems Integration Personnel
- IT Audit Assistant
- Technical Support Specialist

Secondary Job Roles

- QA Specialist
- Systems Analyst
- Computer Programmer

5.5 Allied Fields

In general, subject to the specific provision below, the following may be considered as allied fields:

1. Basic Sciences, Math and Engineering
2. Programs that have at least 50% of core and professional courses of a specific ITE program
3. Any program deemed to be an allied program by the TPITE such as the following:

A. Bachelor of Science in Computer Science (BSCS)

- Applied Mathematics
- Computer Engineering
- Electrical Engineering
- Electronics Engineering
- Entertainment and Multimedia Computing
- Mathematics
- Physics
- Statistics



B. Bachelor of Science in Information Systems (BSIS)

- Applied Mathematics
- Industrial Engineering
- Information Management
- Library and Information Science
- Statistics
- Informatics

C. Bachelor of Science in Information Technology (BSIT)

- Computer Engineering
- Electrical Engineering
- Electronics Engineering
- Informatics
- Information Management

Section 6 Program Outcomes

The minimum standards for the BSCS, BSIS, and BSIT programs are expressed in the following minimum set of graduate outcomes. The graduate outcomes common to all programs, and those common to the discipline are further mapped into the expanded graduate outcomes specific to the sub-disciplines of CS, IS, and IT, as outlined in Section 6.3.

6.1 Common to all programs in all types of schools

The graduates have the ability to

- a) articulate and discuss the latest developments in the specific field of practice. (Philippine Qualifications Framework (PQF) level 6 descriptor) (Graduate Outcomes: CS10, IS10, IT13)
- b) effectively communicate orally and in writing using both English and Filipino (Graduate Outcomes: CS08, IS08, IT10)
- c) work effectively and independently in multi-disciplinary and multi-cultural teams. (PQF level 6 descriptor) (Graduate Outcomes: CS07, IS07, IT08)
- d) act in recognition of professional, social, and ethical responsibility (Graduate Outcomes: CS09, IS09, IT12)
- e) preserve and promote "*Filipino historical and cultural heritage*" (based on RA 7722)

6.2 Common to the discipline

The graduates of BSCS, BSIS, and BSIT must have the ability to

- a) analyze complex problems, and identify and define the computing requirements needed to design an appropriate solution (Graduate Outcomes: CS02, IS02-03, IT03)
- b) apply computing and other knowledge domains to address real-world problems (Graduate Outcomes: CS01, IS01, IT01)
- c) design and develop computing solutions using a system-level perspective (Graduate Outcomes: CS03-05, IS04-05, IT05)
- d) utilize modern computing tools (Graduate Outcomes: CS06, IS06, IT07)



6.3 Specific to a sub-discipline and a major

A. Bachelor of Science in Computer Science (BSCS)

Graduate Attribute	Graduate Outcomes Code	Graduate Outcomes
Knowledge for Solving Computing Problems	CS01	Apply knowledge of computing fundamentals, knowledge of a computing specialization, and mathematics, science, and domain knowledge appropriate for the computing specialization to the abstraction and conceptualization of computing models from defined problems and requirements.
Problem Analysis	CS02	Identify, analyze, formulate, research literature, and solve complex computing problems and requirements reaching substantiated conclusions using fundamental principles of mathematics, computing sciences, and relevant domain disciplines
Design/Development of Solutions	CS03	An ability to apply mathematical foundations, algorithmic principles and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices
	CS04	Knowledge and understanding of information security issues in relation to the design, development and use of information systems
	CS05	Design and evaluate solutions for complex computing problems, and design and evaluate systems, components, or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.
Modern Tool Usage	CS06	Create, select, adapt and apply appropriate techniques, resources and modern computing tools to complex computing activities, with an understanding of the limitations to accomplish a common goal
Individual & Team Work	CS07	Function effectively as an individual and as a member or leader in diverse teams and in multidisciplinary settings
Communication	CS08	Communicate effectively with the computing community and with society at large about complex computing activities by being able to comprehend and write effective reports, design documentation, make effective presentations, and give and understand clear instructions
Computing Professionalism and Ethics	CS09	An ability to recognize the legal, social, ethical and professional issues involved in the utilization of computer technology and be guided by the adoption of appropriate professional, ethical and legal practices
Life-Long Learning	CS10	Recognize the need, and have the ability, to engage in independent learning for continual development as a computing professional

B. Bachelor of Science in Information Systems (BSIS)

Graduate Attribute	Graduate Outcomes Code	Graduate Outcomes
Knowledge for Solving Computing Problems	IS01	Apply knowledge of business processes, computing, mathematics and social sciences appropriate to Information Systems



Problem Analysis	IS02	Analyze a problem, identify and define the computing requirements with respect to organizational factors appropriate to its solution and plan strategies for their solution
	IS03	Evaluate information systems in terms of general quality attributes and possible trade-offs presented within the given requirement
Design/Development of Solutions	IS04	Design, implement, and evaluate information systems, processes, components, or programs and to source cost-benefit efficient alternatives to meet desired needs, goals and constraints
	IS05	Use knowledge and understanding of enterprises in modelling and design of information systems
Modern Tool Usage	IS06	Deploy and use effectively skills, tools and techniques necessary for information systems practice
Individual and Team Work	IS07	Function effectively on teams (recognizing the different roles within a team and different ways of organizing teams) to accomplish a common goal
Communication	IS08	Communicate effectively with a range of audiences. Communication skills includes technical writing, presentation and negotiation, and numeracy
Computing Professionalism and Ethics in the Society	IS09	Recognize the legal, social, ethical and professional issues involved in the exploitation of computer technology and be guided by the adoption of appropriate professional, ethical and legal practices both in the local and global community
Life-Long Learning	IS10	Recognize the need for and engage in an independent and life-long learning, planning self-learning and improving performance as the foundation for on-going professional development

C. Bachelor of Science in Information Technology (BSIT)

Graduate Attribute	Graduate Outcomes Code	Graduate Outcomes
Knowledge for Solving Computing Problems	IT01	Apply knowledge of computing, science, and mathematics appropriate to the discipline
	IT02	Understand best practices and standards and their applications
Problem Analysis	IT03	Analyze complex problems, and identify and define the computing requirements appropriate to its solution
	IT04	Identify and analyze user needs and take them into account in the selection, creation, evaluation and administration of computer-based systems
Design/Development of Solutions	IT05	Design, implement, and evaluate computer-based systems, processes, components, or programs to meet desired needs and requirements under various constraints
	IT06	Integrate IT-based solutions into the user environment effectively
Modern Tool Usage	IT07	Apply knowledge through the use of current techniques, skills, tools and practices necessary for the IT profession
Individual and Team Work	IT08	Function effectively as a member or leader of a development team recognizing the different roles within a team to accomplish a common goal
	IT09	Assist in the creation of an effective IT project plan
Communication	IT10	Communicate effectively with the computing community and with society at large about complex computing activities through logical writing, presentations, and clear instructions



Computing Professionalism and Social Responsibility	IT11	Analyze the local and global impact of computing, information technology on individuals, organizations, and society
	IT12	Understand professional, ethical, legal, security and social issues and responsibilities in the utilization of information technology.
Life-Long Learning	IT13	Recognize the need for and engage in planning self-learning and improving performance as a foundation for continuing professional development

6.4 Common to a horizontal type as defined in CMO 46 s 2012

- Graduates of professional institutions demonstrate a service orientation in one's profession
- Graduates of colleges participate in various types of employment, development activities, and public discourses particularly in response to the needs of the communities one serves
- Graduates of universities participate in the generation of new knowledge or in research and development projects

Graduates of State Universities and Colleges must, in addition, have the competencies to support "national, regional and local development plans." (RA 7722)

A PHEI, at its option, may adopt mission-related program outcomes that are not included in the minimum set.

Section 7 Minimum Performance Indicators

Graduate attributes can be assessed through set of performance indicators provided in the following table.

Graduate Attribute	Performance Indicators
Knowledge for Solving Computing Problems	Completed and successfully defended Capstone Project /Thesis in line with the discipline.
Problem Analysis	Documented software/hardware requirements specifications following computing industry standards.
Design/Development of Solutions	Designed and developed a computing solution using object-oriented approach.
Modern Tool Usage	Used an integrated development environment.
Individual & Team Work	Worked in a group to develop a machine project.
Communication	Presented a proposed solution in class or in a public forum.
Computing Professionalism and Ethics	Immersed/exposed in an actual working environment in industry.
Life-Long Learning	Created a report on a conducted independent learning activity.

An institution may enhance the minimum performance indicators using an industry or globally accepted reference competency inventory.

