**SULE LAMIDO UNIVERSITY, KAFIN HAUSA**
**Faculty of Computing and Information Technology (FCIT)**

# B.Sc Software Engineering

***The Core Curriculum and Minimum Academic Standard for the Nigerian University System***

## Mission and vision Statement of the University:

**Mission:** To produce graduates, sound in character and learning, at undergraduate and graduate levels, imbued with the spirit of service and dedication to innovative application of knowledge aimed at improving the lives of people within Jigawa State, the nation and humanity in general.

**Vision:** To be a top class University that provides a stimulating teaching, learning and research environment in pursuit of solution to societal problems.

## University Motto: Knowledge for Service

## Principal Officers of the University

**Vice Chancellor**
Prof. Muhammad Ibrahim Yakasai

**Registrar**
Mal. Adamu Mohammed

**Bursar**
Mal.

**University Librarian**
Dr. Abdulkadir Idris

### STAFF LIST

| S/N | NAMES | QUALIFICATION | AREA OF SPECIALIZATION | DESIGNATION |
|---|---|---|---|---|
| **1** | Dr. Salisu Garba | **Ph.D** | Service Oriented Computing | **HOD** |

## List of Sule Lamido University CCMAS Technical Reviewer Committee:

1. Prof. Umar Ali (Chairman CCMAS)
2. Mal. Abdullahi Muhammad (Secretary)
3. Dr. Abdullahi Haruna Birniwa (Director of Acad. Planning)
4. Prof. Nasir Faruk (Director ICT/ Dean FCIT)
5. Dr. Iliyasu Yahaya (Dean Fac. of Humanities)
6. Dr. Ali Abdullahi Taura (Dean Fac. Of Education)
7. Dr. Mustapha Hussain (Dean FSMS)
8. Dr. Muhammad Sani Garko (Dean FANR)
9. Dr. Idris Zakariyya Kiri (Dean FNAS)
10. Dr. Muhammad Abdulkadir (Director SPS)
11. Dr. Dahiru Abdulkadir (Director SGES)

# B.Sc. Software Engineering

**Overview**

The software development industry presents huge opportunities within the context of an expanding global economy that is increasingly becoming digital. With the enormous potentials of this sector of the economy and the ever increasing need for large and complex software systems, there is great promise to grow a large crop of software engineers as a force for sustainable socio-economic development. In addition to its core Computer Science foundation, Software Engineering also involves human and technical processes, and therefore borrows and adapts from the field of project management as well as from traditional engineering practice.

**Philosophy**

The philosophy of Software Engineering focuses on producing graduates who have the required knowledge and skills to develop and maintain quality software systems of scale for governments, organisations and businesses that adequately fulfil the functional and nonfunctional requirements of the systems within time and budget constraints.

**Objectives**

The specific objectives of the Software Engineering programme for students are to:

1. provide them a solid foundation in computing in such areas as problem solving, algorithm design, data structures and programming basics;
2. demonstrate practical skills in requirements analysis, system design, software architecture, software metrics, verification and validation, and the software engineering process in general for the production of high quality software-based systems;
3. demonstrate expertise in programming in a number of different languages with emphasis on the production of robust, reliable, cost-effective and secure systems that are based on sound design and development principles;
4. train them to be able to effectively and efficiently manage the development of large, complex and critical software; and
5. enable them to have the requisite knowledge and skill base as well as adequate practical exposure and high ethical standards for the limitless professional career opportunities (including self-employment) in the software industry.

**Unique Features of the Programme**

Special efforts have been made to tailor the programme to the rapidly evolving software industry in Nigeria in particular and Africa in general especially in the following areas:

1. Development of skilled software engineers in mobile applications and web development
2. Rigorous training on how to effectively manage software projects in highly challenging circumstances
3. Grooming of software engineers with specialised skills in Software Engineering but very broad knowledge on the entire computing discipline.

**Employability Skills**

The critical importance and increasing proliferation of software systems in every aspect of human endeavour make it mandatory for today's software engineers to have all the necessary employability skills they require in today's competitive world. They include communication, teamwork and collaboration, negotiation and persuasion, problem solving, leadership, organisation, perseverance, motivation, confidence and the ability to work under pressure.

**21st Century Skills**

Cybersecurity students will be required to have the following 21st century skills:

i. Creative thinking;
ii. information literacy;
iii. media literacy
iv. flexibility;
v. social skills;
vi. problem solving;
vii. innovation skills; and
viii. social skills.

**Admission and Graduation Requirements**

**Admission requirements**

Candidates can be admitted into the Software Engineering degree programme by one of the following ways:

1. The Indirect/Preliminary Mode (4 years Degree Programme)
2. Direct Entry

**4 Year Degree Programme**

In addition to appropriate UTME-Score, a candidate must possess five Senior Secondary Certificate (SSC)-credits passes including English Language, Mathematics, Physics and any other relevant Science subjects in not more than two sittings.

**3 Year Degree Programme:**

**Direct Entry**

A minimum of a credit at the University/National Diploma or NCE with other five Senior Secondary Certificate (SSC) credit passes in relevant Science subjects three of which must be in English Language, Mathematics, Physics.

**Minimum duration**

The minimum duration of the Software Engineering degree programme is four academic sessions for UTME students, however, it is three academic sessions for candidates admitted to the 200 Level.

**Graduation requirements:**

To be eligible for the award of the Bachelor degree in Software Engineering, a student must have:

1. Passed all the core courses, university and faculty/school required courses and electives.
2. Accumulated a minimum of 120 course units for students admitted through UTME and 90 course units for students admitted to 200 level.
3. Attain a minimum CGPA of 1.00.

To graduate, a student must be found worthy in character throughout the period of his/her studentship and must accumulate the total units prescribed for the programme from Core, Faculty and General Studies courses as well as SIWES, Seminar and Final Year Project.

## Summary of Graduation Requirement for B.Sc. Software Engineering

| LEVEL | CCMAS 70% | | | | Additional 30% | | | Required units | |
|---|---|---|---|---|---|---|---|---|---|
| | Specialization | Faculty | General | Total | Core | Elect. | Total | Min | Max |
| 100 | - | 6 | 17 | 23 | 7 | - | 7 | 30 | |
| 200 | 5 | 15 | 8 | 28 | 2 | - | 2 | 30 | |
| 300 | 11 | 6 | 4 | 21 | 9 | - | 9 | 30 | |
| 400 | 10 | 5 | - | 15 | 15 | - | 15 | 30 | |
| Total | 26 | 32 | 29 | 87 | 33 | 0 | 33 | 120<br>DE (90) | |

## Global Course Structure:

### LEVEL 100

**First Semester**

| Course Code | Course Title | Credit Units | Course Status | LH | PH |
|---|---|---|---|---|---|
| GST 111 | Communication in English | 3 | C | 30 | 0 |
| COS 101 | Introduction to Computer Science | 2 | C | 30 | 45 |
| MTH 101 | Elementary Mathematics I | 2 | C | 30 | 0 |
| STA 111 | Descriptive Statistics | 3 | C | 45 | 0 |
| PHY 101 | General Physics I | 2 | C | 30 | 0 |
| PHY 107 | General Practical Physics I | 1 | C | 0 | 45 |
| SLU-COS 121 | Introduction to Application Packages | 2 | C | 15 | 45 |
| | **Total** | **15 Units** | | | |

**Second Semester**

| Course Code | Course Title | Credit Units | Course Status | LH | PH |
|---|---|---|---|---|---|
| GST 112 | Nigeria People and Culture | 2 | C | 30 | 0 |
| COS 102 | Introduction to Problem Solving | 3 | C | 30 | 45 |
| MTH 102 | Elementary Mathematics II | 2 | C | 30 | 0 |
| PHY 102 | General Physics II | 2 | C | 30 | 0 |
| PHY 108 | General Practical Physics II | 1 | C | 0 | 45 |
| SLU-IFT 132 | Introduction to Information System | 3 | C | 45 | 0 |
| SLU-STA 112 | Probability I | 2 | C | 30 | 0 |
| | **Total** | **15 Units** | | | |

**Summary:** Level 100 Student should register a total of **30 credit unit** for this session; **15 credit** for first semester and **15 credit** for second semester.

**LEVEL 200**

**First Semester**

| Course Code | Course Title | Unit(s) | Status | LH | PH |
|---|---|---|---|---|---|
| SEN 201 | Introduction to Software Engineering | 2 | C | 30 | 0 |
| COS 201 | Computer Programming I | 3 | C | 30 | 45 |
| CSC 203 | Discrete Structures | 2 | C | 30 | 0 |
| IFT 211 | Digital Logic Design | 2 | C | 15 | 45 |
| MTH 201 | Mathematical Method I | 2 | C | 30 | 0 |
| ENT 211 | Entrepreneurship and Innovation | 2 | C | 30 | 0 |
| SLU-SEN 203 | Software Engineering Processes | 2 | C | 15 | 45 |
| | **Total** | **15 Units** | | | |

**Second Semester**

| Course Code | Course Title | Unit(s) | Status | LH | PH |
|---|---|---|---|---|---|
| COS 202 | Computer Programming II | 3 | C | 30 | 45 |
| GST 212 | Philosophy, Logic, & Human Existence | 2 | C | 30 | 0 |
| MTH 202 | Mathematical Method II | 2 | C | 30 | 0 |
| IFT 212 | Computer Architecture and Organization I | 2 | C | 15 | 45 |
| INS 204 | System Analysis & Design | 3 | C | 30 | 45 |
| SEN 299 | SIWES I | 3 | C | 00 | 135 |
| | **Total** | **15 Units** | | | |

**Summary:** Level 200 Student should register a total of **30 credit unit** for this session; **15 credit** for first semester and **15 credit** for second semester.

**LEVEL 300**
**First Semester**

| Course Code | Course Title | Unit(s) | Status | LH | PH |
|---|---|---|---|---|---|
| GST 312 | Peace and Conflict Resolution | 2 | C | 30 | 0 |
| ENT 312 | Venture Creation | 2 | C | 15 | 45 |
| CSC 301 | Data Structures | 3 | C | 15 | 45 |
| SEN 301 | Object-Oriented Analysis and Design | 2 | C | 15 | 45 |
| SLU-SEN 303 | Concept of Programming Languages | 3 | C | 30 | 45 |
| SLU-SEN 301 | Database Design and Management I | 3 | C | 30 | 45 |
| | **Total** | **15 Units** | | | |

**Second Semester**

| Course Code | Course Title | Unit(s) | Status | LH | PH |
|---|---|---|---|---|---|
| SEN 304 | Software Testing and Quality Assurance | 2 | C | 15 | 45 |
| SEN 322 | Software Engineering Innovation & New Technology | 2 | C | 15 | 45 |
| SEN 306 | Software Construction | 2 | C | 15 | 45 |
| CSC 308 | Operating Systems | 3 | C | 30 | 45 |
| SEN 399 | SIWES II | 3 | C | 0 | 135 |
| SLU-SEN 302 | Web Application Development | 3 | C | 15 | 45 |
| | **Total** | **15 Units** | | | |

**Summary:** Level 300 Student should register a total of **30 credit unit** for this session; **15 credit** for first semester and **15 credit** for second semester.

**LEVEL 400**
**First Semester**

| Course Code | Course Title | Unit(s) | Status | LH | PH |
|---|---|---|---|---|---|
| COS 409 | Research Methodology and Technical Report Writing | 3 | C | 30 | 0 |
| SEN 401 | Software Configuration Management and Maintenance | 2 | C | 15 | 45 |
| INS 401 | Project Management | 2 | C | 30 | 0 |
| SEN 497 | Final Year Project I | 3 | C | 0 | 135 |
| SLU-SEN 401 | Artificial Intelligence | 2 | C | 15 | 45 |
| SLU-SEN 403 | Human Computer Interface | 3 | C | 30 | 45 |
| | **Total** | **15 Units** | | | |

**Second Semester**

| Course Code | Course Title | Unit(s) | Status | LH | PH |
|---|---|---|---|---|---|
| SEN 410 | Software Architecture and Design | 2 | C | 15 | 45 |
| SEN 498 | Final Year Project I | 3 | C | 0 | 135 |
| SLU-SEN 402 | Mobile Applications Development | 2 | C | 15 | 45 |
| SLU-SEN 404 | Software Engineering Security | 2 | C | 15 | 45 |
| SLU-SEN 406 | Software Engineering Economics | 2 | C | 30 | 0 |
| SLU-SEN 408 | Open Source Software Development and Applications | 2 | C | 15 | 45 |
| SLU-SEN 412 | Distributed, Parallel and Cloud Computing | 2 | C | 15 | 45 |
| | **Total** | **15 Units** | | | |

**Summary:** Level 400 Student should register a total of **30 credit unit** for this session; **15 credit** for first semester and **15 credit** for second semester.

**Summary of TCU**

| S/No. | Level | UTME | DE |
|---|---|---|---|
| 1. | 100 | 30 | - |
| 2. | 200 | 30 | 30 |
| 3. | 300 | 30 | 30 |
| 4. | 400 | 30 | 30 |
| | **Total** | **120** | **90** |

## LEARNING OBJECTIVES AND COURSE CONTENT

### LEVEL 100 Courses

**GST 111: Communication in English** **(2 Units C: LH 15; PH 45)**

**Learning Outcomes:**

At the end of this course, students should be able to:

1. identify possible sound patterns in English Language;
2. list notable Language skills;
3. classify word formation processes;
4. construct simple and fairly complex sentences in English;
5. apply logical and critical reasoning skills for meaningful presentations;
6. demonstrate an appreciable level of the art of public speaking and listening; and
7. write simple and technical reports.

**Course Contents**

Sound patterns in English Language (vowels and consonants. Phonetics and phonology). English word classes (lexical and grammatical words, definitions, forms, functions, usages, collocations). Sentence in English (types: structural and functional, simple and complex). Grammar and Usage (tense, mood, modality and concord, aspects of language use in everyday life). Logical and Critical Thinking and Reasoning Methods (Logic and Syllogism, Inductive and Deductive Argument and Reasoning Methods, Analogy, Generalisation and Explanations). Ethical considerations, Copyright Rules and Infringements. Writing Activities: (Pre-writing, Writing, Post-writing, Editing and Proofreading; Brainstorming, outlining, Paragraphing, Types of writing, Summary, Essays, Letter, Curriculum Vitae, Report writing, Note making, etc. Mechanics of writing). Comprehension Strategies: (Reading and types of Reading, Comprehension Skills, 3RsQ). Information and Communication Technology in Modern Language Learning. Language skills for effective communication. Major word formation processes. Writing and reading comprehension strategies. Logical and critical reasoning for meaningful presentations. Art of public speaking and listening. Report writing.

**GST 112: Nigerian Peoples and Culture** **(2 Units C: LH 30)**

Learning Outcomes At the end of the course, students should be able to:

1. analyse the historical foundation of the Nigerian culture and arts in pre-colonial times;
2. list and identify the major linguistic groups in Nigeria;
3. explain the gradual evolution of Nigeria as a political unit;
4. analyse the concepts of Trade, Economic and Self-reliance status of the Nigerian peoples towards national development;
5. enumerate the challenges of the Nigerian State towards Nation building;
6. analyse the role of the Judiciary in upholding people's fundamental rights;
7. identify acceptable norms and values of the major ethnic groups in Nigeria; and
8. list and suggest possible solutions to identifiable Nigerian environmental, moral and value problems.

**Course Contents**

Nigerian history, culture and art up to 1800 (Yoruba, Hausa and Igbo peoples and culture; peoples and culture of the ethnic minority groups). Nigeria under colonial rule (advent of colonial rule in Nigeria; Colonial administration of Nigeria). Evolution of Nigeria as a political unit (amalgamation of Nigeria in 1914; formation of political parties in Nigeria; Nationalist movement and struggle for independence). Nigeria and challenges of nation-building (military intervention in Nigerian politics; Nigerian Civil War). Concept of trade and economics of selfreliance (indigenous trade and market system; indigenous apprenticeship system among Nigeria people; trade, skill acquisition and self-reliance). Social justice and national development (law definition and classification). Judiciary and fundamental rights.

Individual, norms and values (basic Nigeria norms and values, patterns of citizenship acquisition; citizenship and civic responsibilities; indigenous languages, usage and development; negative attitudes and conducts. Cultism, kidnapping and other related social vices). Re-orientation, moral and national values (The 3R's – Reconstruction, Rehabilitation and Re-orientation) Reorientation Strategies: Operation Feed the Nation (OFN), Green Revolution, Austerity Measures, War Against Indiscipline (WAI), War Against Indiscipline and Corruption (WAIC), Mass Mobilisation for Self-Reliance, Social Justice and Economic Recovery (MAMSER), National Orientation Agency (NOA). Current socio-political and cultural developments in Nigeria.

**COC 101: Introduction to Computer Science (3 Units: LH 30, PH 45)**

**Learning Outcomes**

At the end of the course, students should be able to:

1. explain basic components of computers and other computing devices;
2. describe the various applications of computers;
3. explain information processing and its roles in the society;
4. describe the Internet, its various applications and its impact;
5. explain the different areas of the computing discipline and its specializations; and
6. demonstrate practical skills on using computers and the internet.

**Course Contents**

Brief history of computing. Description of the basic components of a computer/computing device. Input/Output devices and peripherals. Hardware, software and human ware. Diverse and growing computer/digital applications. Information processing and its roles in society. The Internet, its applications and its impact on the world today. The different areas/programs of the computing discipline. The job specializations for computing professionals. The future of computing.

**Lab Work:** Practical demonstration of the basic parts of a computer. Illustration of different operating systems of different computing devices including desktops, laptops, tablets, smart boards and smart phones. Demonstration of commonly used applications such as word processors, spreadsheets, presentation software and graphics. Illustration of input and output devices including printers, scanners, projectors and smartboards. Practical demonstration of the Internet and its various applications. Illustration of browsers and search engines. How to access online resources.

**COC 102: Introduction to Problem Solving (2 Units C: LH 30; PH 0)**

Learning Outcomes: At the end of this course, students should be able to:

1. explain problem solving processes;
2. demonstrate problem solving skills;
3. describe the concept of algorithms development and properties of algorithms;
4. discuss the solution techniques of solving problem;
5. solve computer problems using algorithms, flowcharts, pseudocode, etc.; and
6. solve problems using programming language using C, PYTHON etc.

**Course Contents**

Core concepts of computing. Identification of problems. Types of problems (routine problems and non-routine problems). Problem-solving. Methods of solving computing problems. Algorithms and heuristics. Solvable and unsolvable problems. Solution techniques of solving problems; abstraction; analogy; brainstorming; trial and error; hypothesis testing; reduction; literal thinking; means-end analysis. Method of the focal object; morphological analysis; research; root cause analysis; proof; divide and conquer. General Problem-solving process. Solution formulation and design; flowchart; pseudocode; decision table; decision tree. Programming in any language. 210 Lab Work: Use of simple tools for algorithms and flowcharts; writing pseudocode; writing assignment statements, input-output

statements and condition statements; demonstrating simple programs using any programming language (Visual Basic, Python, C)

**MTH 101: Elementary Mathematics I (Algebra and Trigonometry) (2 Units C: LH 30)**
**Learning Outcomes:**
At the end of the course students should be able to:
1. understand the basic definition of Set, Subset, Union, Intersection, Complements and use of Venn diagrams;
2. solve quadratic equations;
3. solve trigonometric functions;
4. understand various types of numbers; and
5. solve some problems using the Binomial theorem.

**Course Contents**
Elementary set theory, subsets, union, intersection, complements, venn diagrams. Real numbers; integers, rational and irrational numbers, mathematical induction, real sequences and series, theory of quadratic equations, binomial theorem. Complex numbers; algebra of complex numbers; the Argand diagram. De-Moivre's theorem, nth roots of unity. Circular measure, trigonometric functions of angles of any magnitude, addition and factor formulae.

**MTH 102: Elementary Mathematics II (Calculus) (2 Units C: LH 30)**
**Learning Outcomes:**
At the end of the course, students should be able to:
1. distinguish types of rules in Differentiation and Integration;
2. describe the meaning of Function of a real variable, graphs, limits and continuity; and
3. solve some applications of definite integrals in areas and volumes.

**Course Contents**
Function of a real variable, graphs, limits and idea of continuity. The derivative, as limit of rate of change. Techniques of differentiation. Extreme curve sketching; Integration as an inverse of differentiation. Methods of integration, Definite integrals. Application to areas, volumes.

**STA 111: Descriptive Statistics: (3 Units C: LH 45)**
**Learning Outcomes**:
At the end of the course, students should be able to:
1. explain the basic concepts of descriptive statistics.
2. present data in graphs and charts.
3. differentiate between measures of location, dispersion and partition.
4. describe the basic concepts of Skewness and Kurtosis as well as their utility function in a given data set.
5. differentiate rates from ratio and how they are use.
6. compute the different types of index number from a given data set and interpret the output.

**Course content**
Statistical data. Types, sources and methods of collection. Presentation of data. Tables chart and graph. Errors and approximations. Frequency and cumulative distributions. Measures of location, partition, dispersion, skewness and Kurtosis. Rates, ratios and index numbers.

**SLU-PHY 101 General Physics I (Mechanics) (2 Units C: LH 30)**
**Learning Outcomes**:
At the end of the course students should be able to:
1. identify and deduce the physical quantities and their units;

2. differentiate between vectors and scalars;
3. describe and evaluate motion of systems on the basis of the fundamental laws of mechanics;
4. apply Newton's laws to describe and solve simple problems of motion;
5. evaluate work, energy, velocity, momentum, acceleration, and torque of moving or rotating objects;
6. explain and apply the principles of conservation of energy, linear and angular momentum.
7. describe the laws governing motion under gravity; and
8. explain motion under gravity and quantitatively determine behaviour of objects moving under gravity.

**Course Contents**

Space and time; units and dimension, Vectors and Scalars, Differentiation of vectors: displacement, velocity and acceleration; kinematics; Newton laws of motion (Inertial frames, Impulse, force and action at a distance, momentum conservation); Relative motion; Application of Newtonian mechanics; Equations of motion; Conservation principles in physics, Conservative forces, conservation of linear momentum, Kinetic energy and work, Potential energy, System of particles, Centre of mass; Rotational motion; Torque, vector product, moment, rotation of coordinate axes and angular momentum. Polar coordinates; conservation of angular momentum; Circular motion; Moments of inertia, gyroscopes and precession; Gravitation: Newton's Law of Gravitation, Kepler's Laws of Planetary Motion, Gravitational Potential Energy, Escape velocity, Satellites motion and orbits.

**SLU-PHY 102: General physics II (Electricity & magnetism) (2 Units C: LH 30)**

Learning Outcomes At the end of the course, students should be able to:

1. describe the electric field and potential, and related concepts, for stationary charges;
2. calculate electrostatic properties of simple charge distributions using Coulomb's law, Gauss's law, and electric potential;
3. describe and determine the magnetic field for steady and moving charges;
4. determine the magnetic properties of simple current distributions using Biot-Savart and Ampere's law;
5. describe electromagnetic induction and related concepts and make calculations using Faraday and Lenz's laws;
6. explain the basic physical of Maxwell's equations in integral form;
7. evaluate DC circuits to determine the electrical parameters;
8. determine the characteristics of ac voltages and currents in resistors, capacitors, and Inductors.

**Course Contents**

Forces in nature. Electrostatics (electric charge and its properties, methods of charging). Coulomb's law and superposition. Electric field and potential. Gauss's law. Capacitance. Electric dipoles. Energy in electric fields. Conductors and insulators. DC circuits (current, voltage and resistance. Ohm's law. Resistor combinations. Analysis of DC circuits. Magnetic fields. Lorentz force. Biot-Savart and Ampère's laws. Magnetic dipoles. Dielectrics. Energy in magnetic fields. Electromotive force. Electromagnetic induction. Self and mutual inductances. Faraday and Lenz's laws. Step up and step down transformers. Maxwell's equations. Electromagnetic oscillations and waves. AC voltages and currents applied to inductors, capacitors, and resistance.

**SLU-PHY 107: General Practical Physics I (1 Unit C: PH 45)**

Learning Outcomes At the end of the course, students should be able to:

1. conduct measurements of some physical quantities;
2. make observations of events, collect and tabulate data;
3. identify and evaluate some common experimental errors.
4. plot and analyse graphs; and 5. draw conclusions from numerical and graphical analysis of data.

**Course Contents**

This introductory course emphasises quantitative measurements, the treatment of measurement errors, and graphical analysis. A variety of experimental techniques should be employed. The experiments include studies of meters, the oscilloscope, mechanical systems, 100 electrical and mechanical resonant systems, light, heat, viscosity, etc., covered in PHY 101 and PHY 102. However, emphasis should be placed on the basic physical techniques for observation, measurements, data collection, analysis and deduction.

**PHY 108- General Practical Physics II** **(1 Unit C: PH 45)**

Learning Outcomes On completion, the student should be able to:
1. conduct measurements of some physical quantities;
2. make observations of events, collect and tabulate data;
3. identify and evaluate some common experimental errors;
4. plot and analyse graphs;
5. draw conclusions from numerical and graphical analysis of data; and
6. prepare and present practical reports.

**Course Contents**

This practical course is a continuation of PHY 107 and is intended to be taught during the second semester of the 100 level to cover the practical aspect of the theoretical courses that have been covered with emphasis on quantitative measurements, the treatment of measurement errors, and graphical analysis. However, emphasis should be placed on the basic physical techniques for observation, measurements, data collection, analysis and deduction.

**SLU-COS 121 Introduction to Application Packages (2 Units; C: LH 15, PH 90)**

**Learning Outcomes:** At the end of this course, students should be able to;
1) Understand the concept of application packages and their role in modern computing.
2) Explore popular application packages across different domains.
3) Learn essential features and functionalities of application packages.
4) Develop practical skills in using application packages for various tasks.
5) Analyze case studies to understand the real-world applications of different application packages.

**Course content**

Overview of application packages, Importance and benefits, Evolution and trends in application packages. Productivity Suites, Introduction to productivity suites, Microsoft Office Suite (Word, Excel, PowerPoint, Outlook), Google Workspace (Docs, Sheets, Slides, Gmail), Features and functionalities comparison. Graphic Design and Multimedia Packages, Adobe Creative Cloud Suite (Photoshop, Illustrator, InDesign, Premiere Pro), CorelDRAW Graphics Suite, Basics of graphic design and multimedia editing. Introduction to project management tools. Trello, Asana, Basecamp, Collaboration tools: Slack, Microsoft Teams, Zoom. Accounting and Finance Software, QuickBooks, Sage Intacct, Introduction to accounting principles. Introduction to Enterprise Resource Planning (ERP), Oracle ERP Cloud, Overview of ERP systems, Importance and benefits of ERP in businesses. Specialized Application Packages, Industry-specific software (e.g., AutoCAD, MATLAB, SPSS), Healthcare management software, Legal practice management software, Overview and applications in specialized domains

**Lab work:** Practical exercises in graphic design software, Case studies on effective project management using tools, Hands-on exercises in accounting software

**SLU-IFT 132 Introduction to Information System (3 Units: LH 45, P 45)**

**Learning Outcomes**

At the end of this course,
Students should be able to:

1. Explain system concepts and organisational processes;
2. Explain information systems principles and application in modern organisation;
3. Describe information technology security and related ethical issues; and
4. Explain database management and system development life cycle.

**Course Contents**

Roles and relevance of information systems in organisations to conduct business and solve problems. Information systems principles in modern organisations. Systems concepts; organisational processes; technological aspects of information systems. The internet. Information technology security. Ethical issues. Database management. Systems development life cycle.

**SLU-STA 121 Probability I (3 Units; C: LH 45)**

**Learning Outcomes**

At the end of the course, students should be able to:

1. Explain the differences between permutation and combination;
2. Explain the concept of random variables and relate it to probability and distribution functions;
3. Describe the basic distribution functions; and
4. Explain the concept of exploratory data analysis.

**Course Contents**

Permutation and combination. Concepts and principles of probability. Random variables. Probability and distribution functions. Basic distributions: Binomial, geometric, Poisson, normal and sampling distributions; exploratory data analysis.

**LEVEL 200 Courses**

**ENT 211: Entrepreneurship and Innovation (2 Units C: LH 15; PH 45)**

**Learning Outcomes**

At the end of this course, students should be able to:

1. explain the concepts and theories of entrepreneurship, intrapreneurship, opportunity seeking, new value creation, and risk taking;
2. state the characteristics of an entrepreneur;
3. analyse the importance of micro and small businesses in wealth creation, employment, and financial independence;
4. engage in entrepreneurial thinking;
5. identify key elements in innovation;
6. describe stages in enterprise formation, partnership and networking including business planning;
7. describe contemporary entrepreneurial issues in Nigeria, Africa and the rest of the world; and
8. state the basic principles of e-commerce.

**Course Contents**

Concept of Entrepreneurship (Entrepreneurship, Intrapreneurship/Corporate Entrepreneurship,). Theories, Rationale and relevance of Entrepreneurship (Schumpeterian and other perspectives, Risk-Taking, Necessity and opportunity-based entrepreneurship and Creative destruction). Characteristics of Entrepreneurs (Opportunity seeker, Risk taker, Natural and Nurtured, Problem solver and change agent, Innovator and creative thinker). Entrepreneurial thinking (Critical thinking, Reflective Thinking, and Creative thinking). Innovation (Concept of innovation, Dimensions of innovation, Change and innovation, Knowledge and innovation). Enterprise formation, partnership and networking (Basics of Business Plan, Forms of business ownership, Business registration and forming alliances and joint ventures). Contemporary Entrepreneurship Issues (Knowledge, Skills and Technology, Intellectual property, Virtual office, Networking). Entrepreneurship in Nigeria (Biography of inspirational

Entrepreneurs, Youth and women entrepreneurship, Entrepreneurship support institutions, Youth enterprise networks and Environmental and cultural barriers to entrepreneurship). Basic principles of e-commerce.

**GST 212: Philosophy, Logic and Human Existence (2 Units C: LH 30)**

**Learning Outcomes**

At the end of this course, students should be able to

1. know the basic features of philosophy as an academic discipline;
2. identify the main branches of philosophy & the centrality of logic in philosophical discourse;
3. know the elementary rules of reasoning;
4. distinguish between valid and invalid arguments;
5. think critically and assess arguments in texts, conversations and day-to-day discussions;
6. critically asses the rationality or otherwise of human conduct under different existential conditions;
7. develop the capacity to extrapolate and deploy expertise in logic to other areas of knowledge, and
8. guide his or her actions, using the knowledge and expertise acquired in philosophy and logic.

**Course Contents**

Scope of philosophy; notions, meanings, branches and problems of philosophy. Logic as an indispensable tool of philosophy. Elements of syllogism, symbolic logic— the first nine rules of inference. Informal fallacies, laws of thought, nature of arguments. Valid and invalid arguments, logic of form and logic of content — deduction, induction and inferences. Creative and critical thinking. Impact of philosophy on human existence. Philosophy and politics, philosophy and human conduct, philosophy and religion, philosophy and human values, philosophy and character molding, etc.

**COS 201: Computer Programming I (3 Units C1: LH 30; PH 45)**

**Learning Outcomes**

At the end of this course, students should be able to:

1. identify different programming paradigms and their approaches to programming;
2. write programmes using basic data types and strings;
3. design and implement programming problems using selection;
4. design and implement programming problems using loops;
5. use and implement classes as data abstractions in an object-oriented approach;
6. implement simple exception handling in programmes;
7. develop programmes with input/output from text files; and
8. design and implement programming problems involving arrays.

**Course Contents**

Introduction to computer programming. Functional programming; Declarative programming; Logic programming; Scripting languages. Introduction to object-orientation as a technique for modelling computation. Introduction of a typical object-oriented language, such as Java. Basic data types, variables, expressions, assignment statements and operators. Basic object oriented concepts: abstraction; objects; classes; methods; parameter passing; encapsulation. Introduction to Strings and string processing; Simple I/O; control structures; Arrays; Simple recursive algorithms; inheritance; polymorphism. Lab work: Programming assignments involving hands-on practice in the design and implementation of simple algorithms such as finding the average, standard deviation, searching and sorting. Practice in developing and tracing simple recursive algorithms. Developing programmes involving inheritance and polymorphism.

**COS 202: Computer Programming II (3 Units C1: LH 30; PH 45)**

**Learning Outcome**
At the end of this course, students should be able to:
1. Demonstrate the principles of good programming and structured programming concepts;
2. Demonstrate string processing, internal searching, sorting, and recursion;
3. Demonstrate the basic use of OOP concepts: classes, objects, inheritance, polymorphism, data abstraction;
4. Apply the tools for developing, compiling, interpreting and debugging programmes; and
5. Demonstrate the use of syntax and data objects, operators. Central flow constructs, objects and classes programming, Arrays, methods, Exceptions, Applets and the Abstract, OLE, Persistence, Window Toolkit.

**Course Contents**
Review and coverage of advanced object-oriented programming - polymorphism, abstract classes and interfaces. Class hierarchies and programme organisation using packages/namespaces. Use of API – use of iterators/enumerators, List, Stack, Queue from API. Searching, and sorting. Recursive algorithms. Event-driven programming: event-handling methods, event propagation, exception handling. Applications in Graphical User Interface (GUI) programming.

**Lab work:**
Programming assignments leading to extensive practice in problem-solving and programme development with emphasis on object-orientation. Solving basic problems using 131 static and dynamic data structures. Solving various searching and sorting algorithms using iterative and recursive approaches. GUI programming.

**CSC 203: Discrete Structures (3 Units C: LH 45)**
**Learning Outcomes**
At the end of the course, students should be able to:
1. Convert logical statements from informal language to propositional and predicate logic expressions;
2. Describe the strengths and limitations of propositional and predicate logic;
3. Outline the basic structure of each proof technique (direct proof, proof by contradiction, and induction) described in this unit;
4. Apply each of the proof techniques (direct proof, proof by contradiction, and induction) correctly in the construction of a sound argument;
5. Apply the pigeonhole principle in the context of formal proof;
6. Compute permutations and combinations of a set and interpret the meaning in the context of the particular application;
7. map real-world applications to appropriate counting formalisms, such as determining the number of ways to arrange people around a table, subject to constraints on the seating arrangement, or the number of ways to determine certain hands in cards (e.g., a full house); and
8. Solve a variety of basic recurrence relations.

**Course Contents**
Structured approach to analysis and design of information systems for businesses. Software development life cycle. Structured top-down and bottom-up design. Dataflow diagramming. Entity relationship modelling. Computer aided software engineering. Input and output, prototyping design and validation. File and database design. Design of user interfaces. Comparison of structured and object-oriented design

**Lab Work:** system requirements gathering techniques;data modelling techniques (entity relationship modelling);process modelling techniques (data flow diagram); use of UML diagrams; system architectural design; user interface design.

**SEN 201: Introduction to Software Engineering (2 units C: LH 30)**

**Learning Outcomes:**

At the end of this course, students should be able to:

1. describe the concept of the software life cycle;
2. explain the phases of requirements analysis, design, development, testing and maintenance in a typical software life cycle;
3. differentiate amongst the various software development models;
4. utilise UML for object oriented analysis and design;
5. describe different design architectures;
6. explain the various tasks involved in software project management; and
7. describe the basic legal issues related to Software Engineering.

**Course Contents**

Software Engineering concepts and principles. Design, development and testing of software systems. Software processes: software lifecycle and process models. Process assessment models. Software process metrics. Life cycle of software system. Software requirements and specifications. Software design. Software architecture. Software metrics. Software quality and testing. Software architecture. Software validation. Software evolution: software maintenance; characteristics of maintainable software; re-engineering; legacy systems; software reuse. Software Engineering and its place as a computing discipline. Software project management: team management; project scheduling; software measurement and estimation techniques; risk analysis; software quality assurance; software configuration management. Software Engineering and law.

**MTH 201: Mathematical Methods 1 (2 Units C: LH 30)**

**Learning Outcomes:**

At the end of the course, students should be able to:

1. understand Real-valued functions of a real variable;
2. solve some problems using Mean value Theorem and Taylor Series expansion; and
3. evaluate Line Integral, Surface Integral and Volume Integrals.

**Course Contents**

Real-valued functions of a real variable. Review of differentiation and integration and their applications. Mean value theorem. Taylor series. Real-valued functions of two and three variables. Partial derivatives chain rule, extrema, lagrangian multipliers. Increments, differentials, and linear approximations. Evaluation of line, integrals. Multiple integrals.

**MTH 202: Mathematical Methods II (2 Units C: LH 30)**

**Learning Outcomes**:

At the end of the course, students should be able to:

1. define the following: order and degree of a differential equation;
2. describe some techniques for solving first and second order linear and non-linear equations; and
3. solve some problems related to geometry and physics.

**Course Contents**

Derivation of differential equations from primitive, geometry, physics etc. order and degree of differential equation. Techniques for solving first and second order linear and non-linear equations. Solutions of systems of first order linear equations. Finite linear difference equations. Application to geometry and physics.

**IFT 211: Digital Logic Design (2 Units C: LH 15; PH 45)**

**Learning Outcomes:**

At the end of this course, students will be able to:

1. explain why everything is data, including instructions, in computers;
2. describe how negative integers, fixed-length numbers, and non-numeric data are represented;
3. convert numerical data from one format to another;
4. describe computations as a system characterised by a known set of configurations with transitions from one unique configuration (state) to another (state);
5. describe the distinction between systems whose output is only a function of their input (combinational) and those with memory/history (sequential);
6. describe a computer as a state machine that interprets machine instructions;
7. articulate that there are many equivalent representations of computer functionality, including logical expressions and gates, and be able to use mathematical expressions to describe the functions of simple combinational and sequential circuits; and
8. design the basic building blocks of a computer: arithmetic-logic unit (gate-level), registers (gate-level), central processing unit (register transfer-level), and memory (register transfer-level).

**Course Contents**
Introduction to information representation and number systems. Boolean algebra and switching theory. Manipulation and minimisation of completely and incompletely specified Boolean functions. Physical properties of gates: fan-in, fan-out, propagation delay, timing diagrams and tri-state drivers. Combinational circuits design using multiplexers, decoders, comparators and adders. Sequential circuit analysis and design, basic flip-flops, clocking and timing diagrams. Registers, counters, RAMs, ROMs, PLAs, PLDs, and FPGAs. Lab Work: Simple combinational gates (AND, OR, NOT, NAND, NOR); Combinational circuits design using multiplexers, decoders, comparators and adders. Sequential circuit analysis and design using basic flip-flops (S-R, J-K, D, T flip-flops); Demonstration of registers, counters, RAMs, ROMs, PLAs, PLDs, and FPGAs

**IFT 212: Computer Architecture and Organisation (2 Units C: LH 15; PH 45)**
**Learning Outcomes**
At the end of this course, students will be able to:
1. explain the organisation of the classical von Neumann machine and its major functional units;
2. construct simple assembly language programme segments;
3. describe how fundamental high-level programming constructs are implemented at the machine-language level;
4. discuss the concept of control points and the generation of control signals using hardwired or microprogrammed implementations;
5. describe how the use of memory hierarchy (cache, virtual memory) is used to reduce the effective memory latency; and
6. explain the concept of interrupts and describe how they are used to implement I/O control and data transfers.

**Course Contents**
Principles of computer hardware and instruction set architecture. Internal CPU organisation and implementation. Instruction format and types, memory, and I/O instructions. Dataflow, arithmetic, and flow control instructions, addressing modes, stack operations, and interrupts. Data path and control unit design. RTL, microprogramming and hardwired control. The practice of assembly language programming. Memory hierarchy. Cache memory, Virtual memory. Cache performance. Compiler support for cache performance. I/O organisations.

**Lab work:** Practical demonstration of the architecture of a typical computer. Illustration of different types of instructions and how they are executed. Simple Assembly Language programming. Demonstration of interrupts. Programming assignments to practice MS-DOS batch programming, Assembly Process, Debugging, Procedures, Keyboard input, Video Output, File and Disk I/O, and Data

Structure. Demonstration of Reduced Instruction Set Computers. Illustration of parallel architectures and interconnection networks.

**INS 204: Systems Analysis and Design (3 Units C: LH 30; PH 45)**

Learning Outcomes At the end of this course, students should be able to:

1. describe system requirements gathering techniques;
2. explain data modelling technique (entity relationship modelling);
3. explain process modelling technique (data flow diagram);
4. describe system architectural design;
5. describe process and database design; and
6. explain user interface design.

**Course Contents**

Structured approach to analysis and design of information systems for businesses. Software development life cycle. Structured top-down and bottom-up design. Dataflow diagramming. Entity relationship modelling. Computer aided software engineering. Input and output, prototyping design and validation. File and database design. Design of user interfaces. Comparison of structured and object-oriented design. Lab work: Practical exercises on software development life cycle (SDLC) activities with different case studies. Use of different information systems case studies to apply the knowledge of structured top-down and bottom –up design, dataflow diagram and entity relationship models.

**SEN 299: SIWES I (3 Units C: PH 135)**

**Learning Outcomes**

At the end of this training, students should be able to:

1. assess their knowledge of cybersecurity, its application, preventive measures and defence of the cyber environment;
2. explain how a typical cybersecurity unit/department of an organisation operates;
3. describe the various assignments carried out and the skills acquired during the SIWES period; and
4. submit a comprehensive report on the knowledge acquired and the experience gained during the exercise.

**Course Contents**

Students are attached to private and public organisations for a period of three months during the second year session long break with a view to making them acquire practical experience and to the extent possible, develop skills in all areas of Cybersecurity. Students are supervised during the training period and shall be expected to keep records designed for the purpose of monitoring their performance. They are also expected to submit a report on the experience gained and defend their reports.

**SLU-SEN 203: Software Engineering Processes (2 Units C: LH: 15, PH: 45)**

**Learning Outcomes** At the end of the course, students should be able to:

1. define and explain key software engineering concepts such as software processes, life cycle models, and process management.
2. select and apply appropriate software process models (e.g., Waterfall, Agile, Spiral) to different software development projects.
3. assess the maturity of software processes using models like CMMI and identify areas for improvement.
4. use software engineering tools, such as version control systems, issue tracking systems, and build tools, to support software development processes.
5. apply software measurement techniques to assess the quality, efficiency, and effectiveness of software processes and products.
6. understand and apply ethical principles and professional standards in software engineering.

**Course Contents**
Software process definition – software process management and infrastructure, Software life cycles – categories of software processes, software life cycle models, software process adaptation, practical considerations; Software process assessment and improvement – software process assessment methods, software process improvement models, and continuous and staged software process rating; Software measurement – software process and product measurement, quality of measurement results, and software process measurement techniques; Software engineering process tools, version control systems, and testing frameworks.

**LEVEL 300 Course**
**GST 312: Peace and Conflict Resolution (2 Units C: LH 30)**
**Learning Outcomes** At the end of the course, students should be able to:
1. analyse the concepts of peace, conflict, and security;
2. list major forms, types, and root causes of conflict and violence;
3. differentiate between conflict and terrorism;
4. enumerate security and peace building strategies; and
5. describe roles of international organisations, media, and traditional institutions in peace building.

**Course Contents**
Concepts of Peace, Conflict, and Security in a multi-ethnic nation. Types and Theories of Conflicts: Ethnic, Religious, Economic, Geo-political Conflicts; Structural Conflict Theory, Realist Theory of Conflict, Frustration-Aggression Conflict Theory. Root causes of Conflict and Violence in Africa: Indigene and settlers Phenomenon; Boundaries/border disputes; Political disputes; Ethnic disputes and rivalries; Economic Inequalities; Social disputes; Nationalist Movements and Agitations; Selected Conflict Case Studies – Tiv-Jukun; Zango Kataf, Chieftaincy, and Land disputes, etc. Peace Building, Management of Conflicts and Security: 72 Peace & Human Development. Approaches to Peace & Conflict Management --- (Religious, Government, Community Leaders, etc.). Elements of Peace Studies and Conflict Resolution: Conflict dynamics assessment Scales: Constructive & Destructive. Justice and Legal framework: Concepts of Social Justice; The Nigeria Legal System. Insurgency and Terrorism. Peace Mediation and Peace Keeping. Peace & Security Council (International, National, and Local levels) Agents of Conflict resolution – Conventions, Treaties Community Policing: Evolution and Imperatives. Alternative Dispute Resolution, ADR. a) Dialogue b). Arbitration, c). Negotiation d). Collaboration, etc. Roles of International Organisations in Conflict Resolution. (a). The United Nations, UN, and its Conflict Resolution Organs. (b). The African Union & Peace Security Council (c). ECOWAS in Peace Keeping. Media and Traditional Institutions in Peace Building. Managing Post-Conflict Situations/Crisis: Refugees. Internally Displaced Persons, IDPs. The role of NGOs in Post-Conflict Situations/Crisis

**ENT 312: Venture Creation (2 Units C: LH 15; PH 45)**
**Learning Outcomes**
At the end of this course, students, through case study and practical approaches, should be able to:
1. describe the key steps in venture creation;
2. spot opportunities in problems and in high potential sectors regardless of geographical location;
3. state how original products, ideas, and concepts are developed;
4. develop business concepts for further incubation or pitching for funding;
5. identify key sources of entrepreneurial finance;
6. implement the requirements for establishing and managing micro and small enterprises;
7. conduct entrepreneurial marketing and e-commerce;
8. apply a wide variety of emerging technological solutions to entrepreneurship; and

9. appreciate why ventures fail due to lack of planning and poor implementation.

**Course Contents**
Opportunity Identification (Sources of business opportunities in Nigeria, Environmental scanning, Demand and supply gap/unmet needs/market gaps/market research, Unutilised resources, Social and climate conditions, and technology adoption gap). New business development (business planning, market research). Entrepreneurial finance (venture capital, equity finance, microfinance, personal savings, small business investment organisations, and business plan competition). Entrepreneurial marketing and e-commerce (Principles of marketing, customer acquisition & retention, B2B, C2C and B2C models of e-commerce, first mover advantage, e-commerce business models and successful e-commerce companies,). Small business management/family business: Leadership & Management, basic bookkeeping, nature of family business and family business growth model. Negotiation and business communication (Strategy and tactics of negotiation/bargaining, traditional and modern business communication methods). Opportunity discovery demonstrations (business idea generation presentations, business idea contest, brainstorming sessions, idea pitching). Technological solutions (the concept of market/customer solution, customer solution, and emerging technologies, business applications of new technologies- Artificial Intelligence (AI), Virtual/Mixed Reality (VR), Internet of Things (IoT), Blockchain, Cloud Computing, renewable energy, etc. digital business and e-commerce strategies).

**CSC 301: Data Structures (3 Units C: LH 30, PH 45)**
**Learning Outcomes**
At the end of this Course, students should be able to:
1. discuss the appropriate use of built-in data structures;
2. apply object-oriented concepts (inheritance, polymorphism, design patterns, etc.) in software design;
3. implement various data structures and their algorithms, and apply them in implementing simple applications;
4. choose the appropriate data structure for modelling a given problem;
5. analyse simple algorithms and determine their efficiency using big-O notation; and
6. apply the knowledge of data structures to other application domains like data compression and memory management.

**Course Contents**
Primitive types, Arrays, Records Strings and String processing, Data representation in memory, Stack and Heap allocation, Queues, TREES. Implementation Strategies for stack, queues, trees. Run time Storage management; Pointers and References, linked structures. Lab work: Writing C+ /C++ functions to perform practical exercises and implement using the algorithms on arrays, records, string processing, queues, trees, pointers and linked structures.

**CSC 308: Operating System (3 Units C: LH 30; PH 45)**
**Learning Outcomes**
At the end of this course, students should be able to:
1. Recognise operating system types and structures;
2. Describe OS support for processes and threads;
3. Recognise CPU scheduling, synchronisation, and deadlock;
4. Resolve OS issues related to synchronisation and failure for distributed systems;
5. Explain OS support for virtual memory, disk scheduling, I/O, and file systems;
6. Identify security and protection issues in computer systems; and
7. use C and Unix commands, examine behaviour and performance of Linux, and develop various system programmes under Linux to make use of OS concepts related to process synchronisation, shared memory, mailboxes, file systems, etc.

**Course Contents**
Fundamentals of operating systems design and implementation. History and evolution of operating systems. Types of operating systems. Operating system structures. Process management: processes, threads, CPU scheduling, process synchronisation. Memory management and virtual memory. File systems; I/O systems; Security and protection; Distributed systems; Case studies.

**Lab work:** Practical hands-on engagement to facilitate understanding of the material taught in the course. All the process, memory, file and directory management issues will be demonstrated under the LINUX operating system. Also UNIX commands will be briefly discussed. Alternatively, hands-on exposure may be through the use of operating systems developed for teaching, like TempOS, Nachos, Xinu or MiniOS. Another possibility is through programming exercises that implement and simulate algorithms taught. Simulation of CPU scheduling algorithms, producer-consumer problem, memory allocation algorithms, file organisation techniques, deadlock algorithms and disk scheduling algorithms.

**SEN 301: Object-Oriented Analysis and Design (2 Units C: LH 15; PH 45)**
**Learning Outcomes**
At the end of this course, students should be able to:
1. explain the concept of the object-oriented approach to modelling;
2. describe the conceptual model of the UML-based software development life cycle;
3. demonstrate how to use the major UML diagrams for object-oriented analysis and design;
4. demonstrate the use of UML-based CASE tools.

**Course Contents**
Object-oriented approach to information system development, particularly in reference to the earlier stages of analysis and design. Importance of modelling, principles of modelling, object- oriented modelling, conceptual model of the Unified Modelling Language (UML), architecture, software development life cycle. The principles and basic concepts of object orientation and the different aspects of object-oriented modelling as represented by the UML technique. Case study of a typical UML-based CASE tool. Lab Work: Practical exercises on different requirements specification and design activities; developing problem statements, SRS documents and Use Case Diagrams; designing UML Activity diagrams, UML Class diagrams and State Chart diagrams; drawing partial layered, logical architecture diagram with UML package diagram notation; Designing Component and Deployment diagrams.

**SEN 304: Software Testing & Quality Assurance (2 Units C: LH 15; PH 45)**
**Learning Outcomes**
At the end of this Course, students should be able to:
1. state the critical importance of software testing in ensuring software quality;
2. explain the difference between validation and verification and their different techniques;
3. describe the concept of quality assurance and differentiate between process assurance and product assurance;
4. describe the different statistical approaches to quality control.

**Course Contents**
The importance of Software Testing. Understanding Verification and Validation. How to assure it and verify it, and the need for a culture of quality. Avoidance of errors and other quality problems. Inspections and reviews. Testing, verification and validation techniques. Process assurance vs. Product assurance. Quality process standards. Product and process assurance. Problem analysis and reporting. Statistical approaches to quality control

**Lab Work:** Debugging tools; unit testing – black box and white testing techniques; integration and system testing tools; other testing tools – performance testing, load testing, stress testing, regression testing, security testing; manual testing vs automated testing.

**SEN 306: Software Construction (2 Units C: LH 15; PH 45)**
**Learning Outcomes**
At the end of this Course, students should be able to:
1. explain the importance of Software Construction and the key construction decisions;
2. describe the key issues in design including key design concepts, levels of design and Abstract Data Types (ADTs);
3. discuss best practices in dealing with routines, fundamental data types and different types of statements; and
4. describe how to ensure software quality through developer testing, debugging and software craftsmanship.

**Course Contents**
Definition of Software Construction; Its importance; Key construction decisions – choice of programming language, selection of major construction decisions. Design in construction – Key design concepts, levels of design, design heuristics. Abstract Data Types (ADTs). Working Classes. High Quality Routines. The Pseudo Code Programming Process. Fundamental Data

**SEN 322: Software Engineering Innovation and New Technology (2 Units C: LH 15)**
**Learning Outcomes**
At the end of this course, students should be able to:
1. explain business models;
2. identify some entrepreneurial opportunities available in Software Engineering;
3. describe business plan and business startup process;
4. explain business feasibility and strategy;
5. explain marketing strategies; and
6. discuss business ethics and legal issues.

**Course Contents**
Software entrepreneurial process. Principles of software business ownership. Identifying software market opportunities. Entrepreneurial software marketing. Software business communication and negotiation techniques. Feasibility analysis. Entrepreneurial financing. Legal issues. Software business plan development. Risk management.

**SEN 399: SIWES II (3 Units C: PH 135)**
**Learning Outcomes**
Upon the completion of the training, the students should be able to:
1. interact with experts, thus making them gain extra knowledge outside the school environment;
2. compare classwork with real-life working experience in their various areas of specialisation;
3. determine their level of competence;
4. acquire the more practical knowledge and skills;
5. provide a detailed written report on their industrial experience; and
6. defend their project to a panel of examiners.

**Course Contents**
Students are attached to private and public organisations for a period of three months during the third year session-long break with a view to making them acquire additional practical experience in all areas of Cybersecurity over and above what is gained in CYB 299. Students are supervised during the training period and shall be expected to keep records designed for the purpose of monitoring their performance. They are also expected to submit a report on the experience gained and defend their reports.

**SLU-SEN 302: Web Application Development (2 Units C: LH 15; PH 45)**
**Learning Outcomes**

At the end of the course the students should be able to:
1. design and implement simple client-side and server-side web applications;
2. demonstrate hands-on skills in PHP and Python programming uses open-source software;
3. compare and contrast web programming with general-purpose programming; and
4. develop a fully functioning website and deploy it on a web server.

**Course Contents**
Introduction to framework-based web development using a contemporary language like PHP and ASP.net. Principles of web pages (dynamic and static) and website design. The tool used in web development. Client-side and server-side languages. Creation of interactive, dynamic websites using a common web architecture and object-based database access. Design, implementation, and testing of web-based applications including related software, databases, interfaces, and digital media. Standard object models, and the use of server-side programmes for database and file access; testing, software quality assurance; and the process of publishing Web sites. Hands-on PHP and Python programme using open-source software (Apache, PHP, Python, JavaScript, and MySQL). Programming for web development includes control structures, objects, functions, and the use of composite data types. Deploying dynamic content using JavaScript. Designing and developing dynamic web pages and creating, validating, transforming, and formatting data using PHP.

**Lab Work:** Simple PHP programming. Design of simple web pages. Creation of dynamic websites. Design of client-side and server-side programmes. Demonstration of web-based applications with database access. Use of JavaScript to develop dynamic content. Use of Python to develop dynamic web pages.

**SLU-CSC 303: Database Design and Management I (3 Units: LH 30; PH 45)**
**Learning Outcomes**
1. explain key concepts in information storage and retrieval, including data models, data structures, and indexing techniques.
2. design and implement database systems using appropriate data models and database management systems.
3. write complex SQL queries to retrieve, manipulate, and analyze data from relational databases.
4. understand and apply security measures to protect sensitive data, including access control, encryption, and backup and recovery procedures.
5. identify and address performance bottlenecks in database systems, including indexing, query optimization, and database tuning.
6. evaluate the impact of emerging technologies, such as NoSQL databases and big data analytics, on information storage and retrieval.

**Course Contents**
Information storage & retrieval, Information management applications, Information capture and representation, analysis & indexing, search, retrieval, information privacy; integrity, security; scalability, efficiency and effectiveness. Introduction to database systems: Components of database systems DBMS functions, Database architecture and data independence use of database query language.

**SLU-SEN 303: Concepts of Programming Languages (3 Units: LH 35; PH 45)**
**Learning Outcomes**
1. explain the basic concepts of programming languages, including syntax, semantics, and data types.
2. apply various programming paradigms, such as imperative, object-oriented, functional, and logic programming, to solve real-world problems.
3. design and implement efficient algorithms and data structures using appropriate programming languages.

4. analyze and debug programs using debugging tools and techniques.
5. write clear, concise, and well-structured code that adheres to best practices and coding standards.
6. evaluate the impact of language design choices, such as type systems, memory management, and concurrency mechanisms, on program performance and reliability.

**Course Contents**
Preliminaries, evolution of programming languages, paradigms, language design considerations, language processing including syntax and semantic analysis, naming, binding, type checking, expression and assignment statement, statement-level control structures, subprograms, abstract data types, support for object-oriented languages, concurrency, exception handling, functional and logic programming.

## LEVEL 400 Courses

**COS 409: Research Methodology and Technical Report Writing (2 Units C: LH 30)**
**Learning Outcomes**
At the end of this course, students should be able to:
1. explain research, types, approaches and significance of research;
2. discuss statement of problem, research methods, methodology, research process, criteria and strategy for good research;
3. discuss scientific investigation, problem formulation, and technique of the research problem;
4. discuss the guidelines for constructing questionnaire/schedule, guidelines for successful interviewing, research proposal and research plan; and
5. explain types of reports, technical report writing, procedures and guidelines.

**Course Contents**
Foundations of Research. Types of Research. Research Approaches. Significance of Research. Research Methods versus Methodology. Research Process. Criteria and Strategy for Good Research. Problems Encountered by Researchers in Nigeria. Principles of Scientific Research. Scientific investigation. Problem formulation. Definition and technique of the Research Problem. Selection of Appropriate Method for Data Collection- Primary Data and Secondary Data. Guidelines for Constructing Questionnaire/Schedule. Guidelines for Successful Interviewing. Difference between Survey and Experiment. Eloping Research Proposal and Research Plan. Formulation of working hypothesis and Testing. Literature review. Procedure for reviewing related relevant studies and referencing cited works. Types of Reports. Technical Report Writing. Layout and mechanics of writing a Research Report. Standard Techniques for Research Documentation. Sampling Design. Different Types of Sample Designs. Steps in Sampling Design. Criteria of Selecting a Sampling Procedure. Methods of analysis. Processing and Analysis of Data Elements/Types of Analysis. Interpretation and Presentation of results. How to prepare References and Bibliography.

**SEN 401: Software Configuration Management & Maintenance (2 Units C: LH 15; PH 45)**
**Learning Outcomes**
At the end of this course, students should be able to:
1. state the importance of software configuration management;
2. explain the typical processes in software configuration management; and
3. describe the key issues in software maintenance

**Course Contents**
Management of the software configuration management process – organisation context for software configuration management, constraints and guidance for software configuration management process. Planning for software configuration management, software configuration management plan, and

surveillance of software configuration management. Software configuration identification and software library. Software configuration control – requesting, evaluating and approving software changes, implementing software changes, and deviations and waivers. Software configuration status accounting – software configuration status information and reporting. Software configuration auditing. Key issues in software maintenance – technical issues, management issues, maintenance cost estimation, and software maintenance measurement. Maintenance process – maintenance processes and activities. Techniques for maintenance – program comprehension, re-engineering, reverse engineering, migration, and retirement.

**Lab Work:** Practical demonstration of software configuration management processes. Working with software configuration management software. Illustration of software maintenance processes and activities. Working with software maintenance software. Illustration of software re-engineering and reverse engineering techniques.

**INS 401: Project Management** **(2 Units C: LH 30)**

**Learning Outcomes**

At the end of this course, students should be able to:
1. describe project management planning;
2. describe project scheduling;
3. explain the management of project resources;
4. discuss project procurement, monitoring, and execution; and
5. explain project communication and time management.

**Course Contents**

Introduction to Project Management. The Project Management Lifecycle: Project management and systems development or acquisition, The project management context, Technology and techniques to support the project management lifecycle, and Project management processes. Managing Project Teams: Project team planning, Motivating team members, Leadership, power and conflict in project teams, and managing global project teams; Managing Project Communication and enhancing team communication; Project Initiation and Planning. Managing Project Scope: Project initiation, How organisations choose projects, Activities, and Developing the project charter; Managing Project Scheduling: Common problems in project scheduling, and Techniques for project scheduling; Managing Project Resources: Types of resources (human, capital, time), and Techniques for managing resources. Project quality and tools to manage project quality. Managing project risk and tools for managing project risk. Managing Project Procurement: Alternatives to systems development, External acquisition, Outsourcing-domestic and offshore, Steps in the procurement process, and managing the procurement process. Project Execution, Control, and Closure: Managing project execution, monitoring progress, and managing change, Documentation, and communication, and common problems in project execution; Managing Project Control and Closure: Obtaining information, Cost control, Change control, Administrative closure, Personnel closure, Contractual closure, and Project auditing.

**SEN 410: Software Architecture and Design** **(2 Units C: LH 15; PH 45)**

**Learning Outcomes**

At the end of this course, students should be able to:
1. describe design patterns, frameworks and architectures;
2. explain design of distributed systems and component based design; and
3. describe the techniques of designing for qualities such as reliability, performance, safety, security and reusability.

**Course Contents**

An in-depth look at software design. Continuation of the study of design patterns, frameworks, and architectures. Survey of current middleware architectures. Design of distributed systems using

middleware. Component based design. Measurement theory and appropriate use of metrics in design. Designing for quality attributes such as reliability, performance, safety, security, reusability, etc. Measuring internal qualities and complexity of software. Evaluation and evolution of designs.

**Lab Work:** Practical demonstration of the use of design patterns, frameworks and architectures. Practical simulation of distributed systems. Illustration of component based design. Working with software design software. Use of software metrics measuring software.

**SEN 497: Final Year Project I** **(3 Units C: PH 135)**

**Learning Outcomes:**

At the end of this course, students should be able to:

1. identify researchable project topics in Software Engineering;
2. search and review literature pertinent to identified problem statements;
3. acknowledge and reference sources of information used in the research report;
4. conceptualise and design a research methodology to address an identified problem;
5. determine tools for analysing data collected based on research objectives;
6. write a coherent report on the research conducted;
7. take instruction to accomplish the set goals for the project with the guidance of the research supervisor; and
8. orally present the written project report.

**Course Contents**

An independent or group investigation to address a Software Engineering problem under the supervision of a lecturer. Before registering, the student must submit a written proposal to the supervisor for review. The proposal should give a brief outline of the project, estimated schedule of completion, and computer resources needed. A formal written report is essential and an oral presentation may also be required. At the end of the semester, the introduction, literature review and methodology employed should be submitted for grading.

**SEN 498: Final Year Project II** **(3 Units C: PH 135)**

**Learning Outcomes**

Upon completion of the project, students should be able to:

1. demonstrate technical skills in Software Engineering;
2. demonstrate generic transferable skills such as communication and team work;
3. produce a technical report in the chosen project;
4. defend the written project report; and
5. appreciate the art of carrying out a full-fledged research.

**Course Contents**

This is a continuation of SEN 497. This contains the implementation and the evaluation of the project. A formal written report (chapters 4-5) has to be approved by the supervisor. A final report comprising chapters 1-5 will be submitted to the department for final grading. An oral presentation is required.

**SLU-SWE 404: Software Engineering Security** **(2 Units: LH 20; PH 30)**

**Learning Outcomes**

1. explain fundamental security concepts such as threat, vulnerability, risk, and attack vectors.
2. apply security principles, such as least privilege, defense in depth, and fail-safe defaults, to design and implement secure systems.
3. conduct risk assessments, identify vulnerabilities, and implement appropriate security controls to mitigate risks.
4. understand and apply various security technologies, including cryptography, authentication, authorization, and intrusion detection systems.

5. develop incident response plans and respond effectively to security incidents, including data breaches and cyberattacks.
6. stay up-to-date with the latest security trends and technologies, and adapt their security practices accordingly.

**Course Contents**
History and terminology, security mindset, design principles, system/security life–cycle, security implementation mechanisms, information assurance analysis model, disaster recovery, and forensics; Security mechanisms–cryptography, authentication, redundancy, and intrusion detection; Operational issues–trends, auditing, cost/benefit analysis, asset management, standards, enforcement, legal issues, and disaster recovery; Policy–creation of policies, maintenance of policies, prevention, avoidance, incident response (forensics), and domain integration (physical, network, internet, etc.); Attacks – social engineering, denial of service, protocol attacks, active and passive attacks, buffer overflow attacks, and malware; Security domains–security awareness and possible domains; Forensics–legal systems, digital forensics and its relationship to other forensic disciplines, rules of evidence, search and seizure, digital evidence, and media analysis; Security services; Threat analysis model; Vulnerabilities.

**SLU-SEN 401: Artificial Intelligence (2 Units C: LH 15; PH 45)**

**Learning Outcomes**
At the end of this course, students should be able to:
1. explain AI fundamentals, concepts, goals, types, techniques, branches, applications, AI technology and tools;
2. discuss intelligent agents, their performance, examples, faculties, environment and architectures, and determine the characteristics of a given problem that an intelligent system must solve;
3. describe the Turing test and the "Chinese Room" thought experiment, and differentiate between the concepts of optimal reasoning/behaviour and human-like reasoning/behaviour;
4. describe the role of heuristics and the trade-offs among completeness, optimality, time complexity, and space complexity;
5. analyse the types of search and their applications in AI and describe the problem of combinatorial explosion of search space and its consequences;
6. demonstrate knowledge representation, semantic network and frames along with their applicable uses;
7. practice Natural Language Processing, translate a natural language (e.g., English) sentence into a predicate logic statement, convert a logic statement into clause form, apply resolution to a set of logic statements to answer a query; and
8. analyse programming languages for AI and expert systems technology, and employ application domains of AI.

**Course Contents**
Overview of Artificial Intelligence. History of AI. Goals of AI. AI Technique. Types of AI. Branches and applications of AI. Advantages and Disadvantages. Introduction to Intelligent Agents. Agent Performance, Examples of Agents, Agent Faculties, Rationality, Agent Environment. Agent Architectures. Search. General Classes of AI Search Algorithm Problems. Problem Solving by Search. Types of AI Search Techniques and Strategies. Introduction to the types of problems and techniques in AI. Problem-Solving methods. Major structures used in AI programmes. Knowledge Representation. KR and Reasoning Challenges. KR Languages. Knowledge representation techniques such as predicate logic, non-monotonic logic, and probabilistic reasoning. Semantic Network - types of relationships, semantic network inheritance, types and components. Introduction to Frames. Natural Language Processing (NLP). Introduction to natural language understanding and various syntactic and semantic structures. Introduction to Expert Systems - characteristics, components, types, requirements, technology, development. Programming Languages for AI. Introduction to computer image recognition.

**Lab work**: Group practical in (i) Turing test practical - Students can act out their own version of the Turing test (iii) Facial recognition practical to aid in teaching students how machine learning works with students simulating a facial recognition algorithm. Practical applications of NLP in groups – (i) Question Answering focuses on building systems that automatically answer the questions asked by humans in a natural language (ii) Spam detection application for detecting unwanted e-mails getting to a user's inbox (iii) Sentiment analysis/opinion mining should be used on the web to analyse the attitude, behaviour, and emotional state of the sender, implemented through a combination of NLP and statistics (iv) Practical exercise of machine translation used to translate text or speech from one natural language to another natural language such as the Google Translator (v) Developing a model to provide word processor software for the spelling correction (vi) Developing a model for speech recognition for converting spoken words into text (vii) Implementing a Chatbot to provide the staff/student's chat services. OR Group Practical exercise on agents and its environment using simulation of a colony of ants foraging for food; model simulating a message between agents; model simulating the flocking behaviour of birds; model to apply standard search algorithm to the classic search problem of missionaries and cannibals, and how to use communicating agents for searching networks. 46 Some computer AI animation exercises for any branch of AI. Practical exercise on simple robots coupling and programming. Group project of building a lawn robot for trimming grasses, or any simple design and implementation of robotics.

**SLU-SWE 412: Distributed, Parallel & Cloud Computing (2 Units: LH 20; PH 30)**

**Learning Outcomes**

1. define and explain key concepts such as parallelism, concurrency, and distributed systems.
2. design and implement parallel and distributed algorithms using appropriate programming paradigms and tools.
3. analyze the performance of parallel and distributed systems and identify optimization opportunities.
4. effectively use cloud computing platforms (IaaS, PaaS, SaaS) to deploy and manage applications.
5. identify and address security and privacy challenges in parallel and distributed systems, including data protection, access control, and threat mitigation.
6. apply advanced techniques such as big data analytics, machine learning, and artificial intelligence in the context of parallel and distributed computing.

**Course Contents**

Analysis and Design of Parallel and Distributed Algorithm; Language/Operating Systems for parallel processing; GPGPU computing; Architecture of parallel distributed systems, Tools for parallel computing, Parallel (Distributed) database systems, Networking aspects of parallel/distributed computing, parallel/distributed scientific computing Applications; High performance computing Applications in molecular sciences; Multimedia applications for parallel/distributed systems; Grid networks, services and applications; Distributed File Systems; Hyper Scale/Hyper Converged Distributed Storage Design, Storage I/O Protocols, Cloud as a Service, Cloud Infrastructure, Management and operations, Performance, Scalability, Reliability, Virtualization, Cloud Provisioning Orchestration, Architecture support, Development Tools, Platforms and Applications, Legal aspects and Service Level Agreement, Mobile computing advances in the Cloud, Performance optimization.

**SLU-SWE 408: Open-Source Software Development & Application (2 Units: LH 20; PH 30)**

**Learning Outcomes**

1. explain key open-source concepts, including open-source licenses, community norms, and ethical considerations.
2. effectively use version control systems like Git to manage code, collaborate with others, and track changes.
3. contribute to open-source projects by understanding the development process, following community guidelines, and writing high-quality code.

4. design, develop, and deploy open-source applications using various technologies and frameworks.
5. identify and address security vulnerabilities in open-source software and implement best practices for secure development.
6. collaborate with other developers, participate in discussions, and contribute to the open-source community.

**Course Contents**
History Introduces concepts, principles and applications of open-source software, Discusses about open-source software development process. Covers economy, business, societal and intellectual property aspects of open-source software. Obtain hands-on experiences on open-source software and related tools through developing various open-source software applications such as mobile applications, Web applications building on existing open-source frameworks and application development platforms.

**SLU-SWE 406: Software Engineering Economics (2 Units: LH 15; PH 30)**
**Learning Outcomes**
1. apply fundamental economic principles, such as supply and demand, opportunity cost, and marginal analysis, to software engineering projects.
2. assess the economic feasibility of software projects using techniques like cost-benefit analysis, ROI, and payback period.
3. make informed decisions about project scope, resource allocation, and risk management based on economic considerations.
4. use various estimation techniques, such as function point analysis and work breakdown structure, to accurately estimate project costs and timelines.
5. identify, assess, and mitigate risks associated with software projects.
6. analyze the impact of economic factors, such as globalization, outsourcing, and technological advancements, on software development practices.

**Course Contents**
Software engineering economics fundamentals; lifecycle economics; Risk and uncertainty goals, estimates and plans, estimation techniques, addressing uncertainty, prioritization, decisions under risk and uncertainty; Economic analysis methods – for profit decision analysis, minimum acceptable rate of return, return on investment and capital employed, cost-benefit analysis, cost-effectiveness analysis, break-even analysis, business case, multiple attribute evaluation, and optimization analysis; Practical considerations – the "good enough" principle, friction – free economy, ecosystems, and offshoring and outsourcing.

**SLU-SEN 402: Mobile Application Development (2 Units C: LH 15; PH 45)**
**Learning Outcomes**
At the end of the course the students should be able to:
1. identify the basic knowledge on mobile application environment and technology;
2. explain the concepts and processes of mobile application development;
3. discuss design and development issues specific to mobile applications;
4. design and develop mobile applications, using development tools and environments;
5. evaluate the performance of a mobile application and give its result; and
6. appreciate perspectives of mobile applications and their impact.

**Course Contents**
Introduction to developing mobile applications. Mobile operating systems capabilities, application architecture, and major components, such as activities, services, broadcast receivers, etc. Development of interactive applications using widget libraries, web-based services. Basic concepts of 2D graphics and animation. An SQL database engine, and multithreading. Multiplatform mobile application

development. Mobile application basics and features; Android application basics, UI design. Data storage; networking application design. Advanced application design (sensors, camera, GPS, Audio, etc.), graphics and games, webbased hybrid application design. Design and implement a simple mobile application for a given mobile platform. Metrics and methods to evaluate the performance of mobile applications. Mobile application perspectives and impact. Lab Work: Demonstration of a Simple Mobile Application. Design and Development of interactive mobile applications. Demonstration of multiplatform mobile application development. Development of Android applications including UI design and data storage design. Demonstration of advanced mobile application design. Illustration of metrics for measuring the performance of mobile applications.

**SLU-CSC 431: Human-Computer Interface (HCI) (2 Units: LH 30)**

**Learning Outcomes**

1. explain key concepts in human-computer interaction, including usability, user experience, and user-centered design.
2. apply user-centered design principles to create user-friendly and effective interfaces.
3. conduct various user research methods, such as interviews, surveys, and usability testing, to gather insights into user needs and preferences.
4. design intuitive and visually appealing user interfaces that are accessible to a wide range of users.
5. evaluate user interfaces using a variety of methods, including heuristic evaluation, cognitive walkthroughs, and usability testing.
6. stay up-to-date with the latest trends and technologies in human-computer interaction, such as virtual reality, augmented reality, and artificial intelligence.

**Course Contents**

Foundations of HCI, principles of GUI, GUI toolkits; voice interface design, human-centred software evaluation and development; GUI design and programming, user experience.