



ILAYS NATIONAL UNIVERSITY

JAAMACADDA UMMADDA ILAYS



**Faculty of
Computer Science**



**Bachelor of
Computer Science in
Software Engineering**



Ilays University, Lasanod,
Northeastern State, Somalia



www.ilaysuniversity.edu.so



info@ilaysuniversity.edu.so



| ILAYS NATIONAL UNIVERSITY

Program Overview

The Bachelor of Computer Science in Software Engineering at Ilay National University is a five-year academic pathway consisting of a 36-credit Foundation Year and a 144-credit degree stage. It prepares students to understand why software systems work, how they fail, and how to engineer them for real users over time. The program connects mathematical reasoning and computer science fundamentals with the practical disciplines of requirements analysis, design, implementation, verification, security and deployment.

The Foundation Year develops academic English, mathematics, critical thinking and digital readiness. In the degree stage, students study programming, algorithms, databases, operating systems, computer networks, cybersecurity and distributed computing. This common core is coordinated with the Computer Networks and Security program, enabling consistent treatment of shared subjects and access to common laboratory teaching. The software engineering track then develops a separate professional identity through web and mobile development, systems analysis, architecture, testing, secure software development and DevOps.

Learning becomes progressively more independent. Students begin with guided programming and laboratory exercises, move into requirements and design work, and then undertake integrated team projects. The final stage combines advanced software engineering with an internship, a professional portfolio and a two-part capstone project. Across these stages, students are expected to justify technical decisions, document their work, review code and evaluate software against user needs, quality criteria and ethical obligations.

Ilay National University positions the program for graduates who can contribute to software teams in businesses, education, public services and other organizations. Its broad computing base also supports postgraduate study and adaptation as tools change. This profile describes the intended educational design; individual course specifications and university regulations govern admission, assessment, formal prerequisites and award decisions.

Vision and Mission



VISION

To develop a nationally respected software engineering program that produces innovative, ethical and industry-ready graduates capable of designing dependable, secure and scalable software systems for social and economic development.



MISSION

To provide rigorous, practice-oriented education in computing and software engineering through strong foundations, laboratory work, collaborative projects, research, industry engagement, secure development practices and continuous adaptation to emerging technologies.

Rationale and Distinctive Features

Organizations increasingly depend on reliable digital services. This degree responds by combining computer science principles with the engineering practices used across the software lifecycle. The common core permits coordinated teaching and access to shared facilities; project work, internship and a two-part capstone give students opportunities to apply learning to authentic problems.

For Ilay National University, the program offers a coherent way to prepare graduates for both local digital service needs and further study. Students learn to clarify stakeholder requirements, make design decisions, implement maintainable systems and assess the consequences of failure, insecure data handling and poor usability. The

emphasis on practical work helps connect abstract computing concepts with the software used by businesses, public services and communities.



Program Objectives

- Develop strong foundations in programming, algorithms, mathematics, computer systems, databases, networks and cybersecurity.
- Develop competence in requirements engineering, systems analysis, software design and software architecture.
- Prepare students to build modern web, mobile, distributed and cloud-enabled software systems.
- Develop rigorous software testing, quality assurance, secure coding and software maintenance practices.
- Develop skills in DevOps, CI/CD, automation and professional software deployment workflows.
- Build research, project-management, teamwork, communication and entrepreneurship capabilities.
- Prepare graduates for professional software engineering roles, postgraduate study and technology entrepreneurship.

Program Learning Outcomes

By graduation, students are expected to demonstrate the following abilities through coursework, laboratory work, projects and professional practice:

- Analyze complex computing problems and formulate appropriate software-based solutions.
- Design and implement reliable, maintainable, secure and user-oriented software systems.
- Apply algorithms, data structures, databases, operating systems and networks effectively in software projects.
- Use professional development tools, version control, testing frameworks, cloud services and deployment pipelines.
- Evaluate software quality, security, performance and maintainability using appropriate methods.
- Work effectively in teams and communicate technical ideas to specialist and non-specialist audiences.
- Apply professional, ethical, legal and privacy principles in software engineering practice.
- Conduct research and complete a substantial software-oriented capstone project.

Graduate Attributes



A graduate should be able to explain the reasoning behind a technical decision as well as demonstrate a working system. The intended graduate profile combines analytical judgment, disciplined implementation, clear documentation, collaborative habits and responsibility for users and data. Practical evidence can include reviewed code, design artifacts, test results, a project portfolio and feedback from supervised work.

Why Choose This Program

The curriculum gives students broad computing preparation before deeper specialization, supporting movement between software roles as technologies change. Required work in web and mobile applications, software architecture, secure development, quality assurance and DevOps follows a connected progression. The internship and capstone provide opportunities to integrate these skills and present work to academic and professional audiences.

Admission and Entry Pathways

Applicants should meet the general undergraduate admission requirements set by Ilays National University. The Foundation Year provides preparation in English, mathematics, digital literacy and introductory computing. Entry with prior tertiary study, exemptions or transfer credit should be assessed individually under the university's approved recognition and transfer rules; this curriculum does not itself establish an automatic exemption.

Teaching and Learning Approach

Teaching combines lectures, guided laboratories, programming exercises, design studios, seminars, case discussions and team projects. Students use iterative development, code review, documentation and testing across successive courses. Practical activities should use accessible development tools and managed laboratory environments, with explicit attention to secure and ethical conduct.

In the early semesters, lecturers should model problem decomposition and give prompt feedback on small coding tasks. Intermediate courses can use shared repositories, milestones and peer review so that students experience the coordination required in a software team. Senior teaching should place more weight on design tradeoffs, system evaluation and clear communication to stakeholders. The course team should support students with different levels of prior computing experience while maintaining the same graduate outcomes.

Assessment Framework

For standard taught courses, the following proposed weighting gives students several ways to demonstrate learning while retaining substantial individual examination. It should be confirmed against Ilays National University assessment regulations and stated in each approved course specification before delivery.

Assessment Component	Weight	Evidence and Purpose
Attendance and Participation	10%	Prepared attendance and constructive engagement in taught and practical sessions
Individual Assignments and Practical Tasks	15%	Independent coding, analysis, written work or laboratory submissions
Group Assignments and Project Work	10%	Collaborative design, implementation and presentation with individual contribution recorded
Mid-Term Examination	25%	Individual assessment of concepts and applications covered in the first part of the semester
Final Examination	40%	Comprehensive individual assessment of course outcomes
TOTAL	100%	Standard taught-course model

Attendance and participation evidence should be recorded consistently; an approved accommodation or absence should be handled under university policy. Individual work must test each student's own understanding. Group work should use a shared rubric for the product alongside evidence of individual contribution, such as repository activity, peer review and a short oral explanation. The midterm and final examination should include applied tasks where appropriate, rather than relying solely on recall.

Courses centered on supervised practice, including the internship, Project I, Project II and the two-part capstone, may require approved course-specific assessment patterns instead of this standard model. Their specifications should identify deliverables, supervisor assessment, demonstration or defense, moderation, feedback, academic integrity rules and reassessment arrangements.

Technical submissions should be assessed for correctness and also for readability, testing, documentation, security and justified design choices where these are relevant to the learning outcomes. Project rubrics should distinguish product quality from team process and individual contribution. Formative review of drafts, prototypes and test plans can identify problems early enough for students to improve the final submission.

Student Support and Learning Resources

Delivery requires computer laboratories, dependable development environments, network and cloud access appropriate to each course, library resources and supervision for projects. Academic advising should help students plan prerequisites, monitor progress and select an elective. Students should receive formative feedback on code, design choices, writing and professional conduct.

Learning Environment and Staffing

The faculty should provide practical access to version control, integrated development environments, test automation, databases, virtualized or cloud based environments and accessible reference materials. Laboratory activities should be supported by staff able to supervise safe implementation and troubleshoot common toolchain issues. Specialist modules need appropriately qualified instructors and access to tools aligned with their learning outcomes; resource plans should be reviewed before each annual delivery cycle.

Industry Engagement and Professional Readiness

Industry engagement should be built into the program from the intermediate years, rather than reserved for the final internship. Ilays National University can invite software practitioners to review project briefs, discuss local implementation constraints and offer feedback on student demonstrations. Case work can draw on realistic needs in

education, commerce, public administration and community services. Such activities help students understand that a useful system must respond to stakeholders, available infrastructure, costs, privacy obligations and long-term maintenance capacity.

Collaborative projects should mirror important parts of professional development without assuming that every student has access to a placement. Teams can work from documented requirements, maintain a shared repository, review code, track issues, test releases and present a deployment or handover plan. Guest reviews and demonstrations can expose students to changing requirements, security checks, accessibility concerns and release schedules. The department should distinguish proposed collaborations from established partnerships and obtain the necessary permissions before using an organization's real data or systems.

The fourth-year internship should operate through a learning agreement that names the workplace and academic supervisors, expected duties, attendance or activity records, confidentiality requirements and the method of evaluation. If suitable placements are limited, the university should provide an approved equivalent supervised professional project under its academic rules. Students should leave with evidence they can explain and defend: a requirements specification, design rationale, version history, test results, deployment notes, supervisor feedback and a reflective account of their contribution.

Employability work should also include technical communication, teamwork and professional conduct. Students should practice writing concise documentation, explaining tradeoffs to non-specialists and responding constructively to review. The final portfolio and capstone defense can bring these records together, allowing assessors to judge how independently a graduate can plan, build, test and maintain a software solution.

Academic Progression and Prerequisite Guidance

Programming and discrete mathematics support data structures and algorithms; architecture supports operating systems; database foundations support advanced data management; software engineering fundamentals support design, testing and senior projects. The department should publish formal prerequisite and co-requisite rules in individual course specifications and address any scheduling exceptions through approved academic procedures.



The pathway moves from readiness and core computing to software development, integrated quality work and professional practice. Projects, the internship and capstone provide evidence of growing independence at each stage.

Curriculum Structure and Contact Hours

The tables below reproduce the supplied curriculum. LH and PH indicate total lecture and supervised practical contact hours across each 16-week semester. Shared Core courses are intended to align in code, credits and contact hours with the parallel Computer Networks and Security program. Course sequencing and totals are shown as supplied.

CREDIT ARCHITECTURE

FOUNDATION	SHARED COMPUTING CORE	SPECIALIZATION
36 credits	102 credits	42 credits

180 credits total | 36 foundation + 144 degree

Within the 144-credit degree stage, 102 credits are shared with the parallel computing program and 42 credits establish the Software Engineering specialization. The Foundation Year contributes a further 36 credits to the full pathway. This distribution supports breadth in computing while reserving sustained space for discipline-specific study.

Academic Development across the Pathway

The Foundation Year establishes the language, quantitative and digital skills needed for degree study. Introductory programming, web development and hardware work make the transition into applied computing concrete. Its 36 credits are shown separately from the 144-credit degree stage so that the full 180-credit pathway is transparent.

Year I introduces core programming, discrete mathematics, probability and statistics, computer architecture, data structures, operating systems, databases and networks. Web Application Development I and Software Engineering Fundamentals give students their first specialization experience while the shared core remains dominant.

Year II extends technical breadth through Python, Linux, cybersecurity, algorithm analysis and cloud or distributed computing foundations. Systems Analysis and Design, full stack development, advanced programming and data management connect that foundation to a first computing project. Students begin to treat specifications, implementation and evaluation as parts of one workflow.

Year III advances software architecture, theory of computation, mobile development and quality assurance alongside research methods, cryptography, data analysis and distributed systems. The integrated project provides a setting for managing interfaces, responsibilities and testing across a larger solution.

Year IV focuses on artificial intelligence, cloud infrastructure, cyber law, advanced and secure software engineering, deployment and professional practice. Students propose and design the capstone before implementing and defending it. The internship and portfolio create additional evidence of workplace readiness, while the elective offers depth in an approved area.

Foundation Year Semester I

No.	Code	Course Title	Type	Cr.	LH	PH
1	FE000	Freshman English	Shared Core	3	48	0
2	PC001	Pre-Calculus	Shared Core	3	48	0
3	SH002	Somali History and Culture I	Shared Core	2	32	0
4	CT003	Critical Thinking	Shared Core	2	32	0
5	CF004	Computer Fundamentals	Shared Core	3	32	32
6	CA005	Computer Applications / Digital Literacy	Shared Core	3	16	64
		SEMESTER TOTAL		16	208	96

Foundation Year Semester II

No.	Code	Course Title	Type	Cr.	LH	PH
1	AE007	Academic English I	Shared Core	3	48	0
2	CL008	Calculus	Shared Core	3	48	0
3	IS009	Islamic Studies	Shared Core	2	32	0
4	IP010	Introduction to Programming & Computing	Shared Core	4	32	64
5	WD011	Web Development Fundamentals	Shared Core	3	32	32
6	SH012	Somali History and Culture II	Shared Core	2	32	0
7	CM013	Computer Hardware Fundamentals & Maintenance	Shared Core	3	16	64
		SEMESTER TOTAL		20	240	160

Year I Semester I

No.	Code	Course Title	Type	Cr.	LH	PH
1	CSC101	Academic English II & Technical Communication	Shared Core	3	48	0
2	CSC102	Programming Fundamentals	Shared Core	3	32	64
3	CSC103	Discrete Mathematics	Shared Core	3	48	0
4	CSC104	Probability and Statistics	Shared Core	3	48	0
5	CSC105	Computer Organization & Architecture	Shared Core	3	32	32
6	SE106	Web Application Development I	SE Specialization	3	32	64
		SEMESTER TOTAL		18	240	160

Year I Semester II

No.	Code	Course Title	Type	Cr.	LH	PH
1	CSC107	Object-Oriented Programming	Shared Core	3	32	64
2	CSC108	Data Structures & Algorithms	Shared Core	3	32	32
3	CSC109	Operating Systems	Shared Core	3	32	32
4	CSC110	Database Systems I	Shared Core	3	32	32
5	CSC111	Computer Networks I	Shared Core	3	32	32
6	SE112	Software Engineering Fundamentals	SE Specialization	3	32	32
		SEMESTER TOTAL		18	192	224

Year II Semester I

No.	Code	Course Title	Type	Cr.	LH	PH
1	CSC201	Python Programming	Shared Core	3	32	64
2	CSC202	Systems Programming & Linux	Shared Core	3	32	64
3	CSC203	Fundamentals of Cybersecurity	Shared Core	3	32	32
4	CSC204	Internet & Web Security Fundamentals	Shared Core	3	32	32
5	SE205	Systems Analysis & Design	SE Specialization	3	32	32
6	SE206	Full-Stack Web Application Development	SE Specialization	3	32	64
		SEMESTER TOTAL		18	192	288

Year II Semester II

No.	Code	Course Title	Type	Cr.	LH	PH
1	CSC207	Design & Analysis of Algorithms	Shared Core	3	48	32
2	CSC208	Cloud & Distributed Computing Fundamentals	Shared Core	3	32	32
3	CSC209	Computing Ethics & Professional Practice	Shared Core	2	32	0
4	CSC210	Project I - Computing Project	Shared Core	4	0	96
5	SE211	Advanced Programming & Software Design	SE Specialization	3	32	64
6	SE212	Database Systems II & Data Management	SE Specialization	3	32	32
		SEMESTER TOTAL		18	176	256

Year III Semester I

No.	Code	Course Title	Type	Cr.	LH	PH
1	CSC301	Applied Cryptography	Shared Core	3	48	32
2	CSC302	Research Methods in Computing	Shared Core	3	32	32
3	CSC303	Data Analysis	Shared Core	3	32	64
4	CSC304	Linear Algebra for Computing	Shared Core	3	48	0
5	SE305	Theory of Computation	SE Specialization	3	48	0
6	SE306	Software Architecture & Design	SE Specialization	3	32	64
		SEMESTER TOTAL		18	240	192

Year III Semester II

No.	Code	Course Title	Type	Cr.	LH	PH
1	CSC307	Distributed Systems	Shared Core	3	32	32
2	CSC308	Internet of Things	Shared Core	3	32	64
3	CSC309	ICT Project Management & Entrepreneurship	Shared Core	3	32	32
4	CSC310	Project II - Integrated Computing Project	Shared Core	3	0	96
5	SE311	Mobile Application Development	SE Specialization	3	32	64
6	SE312	Software Testing & Quality Assurance	SE Specialization	3	32	64
		SEMESTER TOTAL		18	160	352

Year IV Semester I

No.	Code	Course Title	Type	Cr.	LH	PH
1	CSC401	Artificial Intelligence & Machine Learning	Shared Core	3	32	64
2	CSC402	Cloud Computing, Virtualization & Infrastructure Automation	Shared Core	3	32	64
3	CSC403	Cyber Law, Ethics & Digital Privacy	Shared Core	3	48	0
4	CSC404	Capstone Project I - Proposal & Design	Shared Core	3	16	64
5	SE405	Advanced Software Engineering	SE Specialization	3	32	64
6	SE406	Secure Software Development	SE Specialization	3	32	64

No.	Code	Course Title	Type	Cr.	LH	PH
		SEMESTER TOTAL		18	192	320

Year IV Semester II

No.	Code	Course Title	Type	Cr.	LH	PH
1	CSC407	Internship / Professional Practice	Shared Core	4	0	192
2	CSC408	Capstone Project II - Implementation & Defense	Shared Core	6	16	192
3	CSC409	Capstone Seminar & Professional Portfolio	Shared Core	2	16	32
4	SE410	DevOps & CI/CD	SE Specialization	3	32	64
5	SE411	Software Engineering Elective	SE Specialization	3	32	32
		SEMESTER TOTAL		18	96	512

Credit and Contact Hour Summary

Stage / Semester	Credits	LH	PH	Status
Foundation - Semester I	16	208	96	Shared
Foundation - Semester II	20	240	160	Shared
Year I - Semester I	18	240	160	Degree
Year I - Semester II	18	192	224	Degree
Year II - Semester I	18	192	288	Degree
Year II - Semester II	18	176	256	Degree
Year III - Semester I	18	240	192	Degree
Year III - Semester II	18	160	352	Degree
Year IV - Semester I	18	192	320	Degree
Year IV - Semester II	18	96	512	Degree
FOUNDATION TOTAL	36	448	256	
DEGREE TOTAL	144	1488	2304	
OVERALL PATHWAY	180	1936	2560	

Elective Provision

The final semester includes SE411, a three-credit Software Engineering Elective. The department should approve its offered topic, syllabus, staffing and laboratory provision before enrolment. Potential subjects in the source curriculum include advanced web technologies, human computer interaction, computer graphics, advanced databases, data mining, natural language processing, computer vision, cloud application engineering, software product management and advanced mobile development. An offered topic should avoid substantial duplication of required courses.

Internship and Capstone

The fourth-year internship connects students with supervised professional practice and should have a learning agreement, workplace supervisor, academic supervisor and documented evaluation. Capstone Project I develops a proposal and design; Capstone Project II implements, evaluates and defends the solution. The department should

publish project scope, milestones, supervision arrangements, ethical review requirements where relevant and assessment criteria.

Graduate Profile and Career Pathways

Graduates can contribute to software development, web and mobile engineering, quality assurance, systems analysis, DevOps, cloud application delivery and technical product teams. The degree also provides a basis for postgraduate study and technology entrepreneurship. Specific employment outcomes depend on the graduate's demonstrated skills, experience and local opportunities.

Potential direction	Relevant program preparation
Software developer or engineer	Programming, design, databases and team projects
Quality assurance or test engineer	Testing methods, automation and evidence of verification
Systems analyst	Requirements, modeling and stakeholder communication
DevOps or cloud application associate	Version control, CI/CD, cloud and deployment practice
Technical founder or postgraduate researcher	Entrepreneurship, research methods and capstone

Research Innovation and Community Relevance

Faculty supervised projects may address practical challenges in education, commerce, public administration and community information services. Suitable work should define users and evaluation criteria before development and address accessibility, privacy and sustainability. Research methods and capstone supervision help students distinguish a functional prototype from a well evidenced and maintainable solution.

Quality Assurance and Review

The department should review course specifications, assessment outcomes, student feedback, internship reports, capstone quality and employer input at regular intervals. Proposed changes to credits, course codes or shared core content should be coordinated with the parallel program and approved through the university's academic governance process.

Graduation Requirements

- Successfully complete the 36-credit Foundation Year where applicable.
- Successfully complete all 144 bachelor-level credits, including the required shared-core and specialization courses.
- Successfully complete the Internship / Professional Practice requirement.
- Successfully complete and defend the Capstone Project.
- Meet the university's minimum cumulative academic standing and all other institutional graduation requirements.