FIȘA DISCIPLINEI

1. Date despre program

Facultatea	Inginerie Electrică și Știința Calculatoarelor
Departamentul	Calculatoare
Domeniul de studii	Calculatoare si tehnologia informației
Ciclul de studii	Licență, dual
Programul de studii	Calculatoare

2. Date despre disciplină

Denumirea disciplinei	**Învățare prin muncă în specialitate - proiect I**					
Anul de studiu	III	Semestrul	V	Tipul de evaluare	P	
Regimul disciplinei	Categoria formativă a disciplinei DF - fundamentală, DS - de specializare, DC – complementară					DS
	Categoria de opționalitate a disciplinei: DOB – obligatorie, DOP – opțională, DFA - facultativă					DOB

3. Timpul total estimat (ore alocate activităților didactice)

I a) Număr de ore pe săptămână	4	Curs		Seminar		Laborator/ Lucrări practice IIS		Proiect IIS		Practică IIS	
						Laborator/ Lucrări practice IM		Proiect IM	4	Practică IM	
I b) Totalul de ore pe semestru din planul de învățământ	56	Curs		Seminar		Laborator		Proiect	56	Practică	

(*IIS – instituție de învățământ superior; IM – învățare prin muncă*)

Distribuția fondului de timp pe semestru	Ore IIS	Ore IM
II.Studiu individual		141
III. Examinări		3
IV. Alte activități (precizați):		

Total ore studiu individual (II + III)	Ore IIS		Ore IM	144
Total ore pe semestru (Ib+II+III+IV)	Ore IIS		Ore IM	200
Numărul de credite	Credite IIS		Credite IM	8

4. Competențe specifice acumulate

Competențe transversale	CT1 - munca in echipa; CT2 - se adresează unui public; CT3 - respecta obligațiile de confidențialitate;
Competențe profesionale	C1 - supraveghează dezvoltarea de software; C2 - analizează specificații software; C3 - aliniază software-ul la arhitecturile de sistem; C4 - definește arhitectura software; C5 - creează softuri;

5. Rezultatele învățării

Cunoștințe	Aptitudini	Responsabilitate și autonomie
Studentul analizează specificațiile unei componente software și justifică arhitectura propusă pentru ea, precum și modul în care aceasta se aliniază arhitecturii sistemului companiei.	Studentul definește arhitectura și implementează o componentă software de complexitate ridicată, integrată în produsul companiei, conform standardelor de calitate ale acesteia: teste automate, revizuire de cod, documentație.	Studentul lucrează autonom în echipa de proiect, coordonează dezvoltarea propriei componente (planificarea sarcinilor, revizuirea codului colegilor) și prezintă rezultatele în fața echipei sau a beneficiarului, respectând confidențialitatea.

6. Obiectivele disciplinei (reieșind din grila competențelor specifice acumulate)

Obiectivul general al disciplinei	Dezvoltarea capacității studentului de a proiecta și realiza în mod autonom o componentă software de complexitate ridicată, integrată într-un produs real al companiei partenere, și de a participa la coordonarea procesului de dezvoltare. Disciplina urmărește ca studentul: 1. să analizeze specificațiile și să definească arhitectura componentei, aliniată arhitecturii sistemului existent; 2. să implementeze, testeze și documenteze componenta conform standardelor de calitate ale companiei; 3. să își coordoneze activitatea și pe a colegilor implicați în componentă, să prezinte rezultatele și să respecte obligațiile de confidențialitate.

7. Conținutul predării și învățării

Curs	Nr. ore	Metode de predare	Observații
Bibliografie minimală recomandată			

Aplicații IM (proiect)	Nr. ore	Metode de predare	Observații
Stabilirea temei și analiza specificațiilor componentei	6	Mentorat, studiu de caz, învățare prin muncă	Tema se stabilește cu tutorele de practică și poate fi corelată cu tema lucrării de licență.
Definirea arhitecturii componentei și alinierea la arhitectura sistemului	10	Studiu de caz, mentorat, învățare prin muncă	
Implementarea componentei	24	Aplicații practice, învățare prin muncă	
Testarea automată, integrarea în sistem și revizuirea codului colegilor	10	Exercițiu aplicativ, învățare prin muncă	
Documentarea tehnică și prezentarea rezultatelor	6	Redactare ghidată, învățare prin muncă	
Bibliografie minimală recomandată			

1. I. Sommerville, Software Engineering, 10th ed. Pearson, 2016.
2. D. Thomas, A. Hunt, The Pragmatic Programmer: Your Journey to Mastery, 20th Anniversary ed. Addison-Wesley, 2019.
3. K. Wiegers, J. Beatty, Software Requirements, 3rd ed. Microsoft Press, 2013.
4. S. Chacon, B. Straub, Pro Git, 2nd ed. Apress, 2014. [Online]. Disponibil: git-scm.com/book
5. ISTQB, Certified Tester Foundation Level Syllabus, v4.0, 2023. [Online]. Disponibil: istqb.org
6. R. C. Martin, Clean Architecture: A Craftsman's Guide to Software Structure and Design. Prentice Hall, 2017.
7. M. Fowler, Refactoring: Improving the Design of Existing Code, 2nd ed. Addison-Wesley, 2018.

8. Evaluare

Tip activitate	Criterii de evaluare	Metode de evaluare	Pondere din nota finală
Evaluare finală	arhitectura este documentată, componenta este integrată și funcționează, testele automate trec, iar documentația tehnică este completă.	Demonstrația componentei, analiza codului și a documentației	40%
	studentul justifică arhitectura propusă și alinierea ei la sistemul existent.	Prezentare orală și întrebări	20%
Evaluare pe parcurs	sarcinile au fost planificate și livrate autonom, studentul a făcut revizuiri de cod, a prezentat progresul și a respectat confidențialitatea.	Fișa de evaluare completată de tutore și jurnalul de activitate	40%

Fișa disciplinei include, dacă este cazul, elemente adaptate persoanelor cu dizabilități, în funcție de tipul și gradul acestora.
.

Data completării	Grad didactic, nume, prenume, semnătura titularului de curs	Grad didactic, nume, prenume, semnătura titularului de aplicație
24.09.2026	-	conf. dr. ing. Ioan Ungurean

Data avizării	Grad didactic, nume, prenume, semnătura responsabilului de program
24.09.2026	conf. dr. ing. Ioan Ungurean

Data avizării în departament	Grad didactic, nume, prenume, semnătura directorului de departament
24.09.2026	Prof. dr. ing. Ovidiu-Andrei SCHIPOR

Data aprobării în consiliul facultății	Grad didactic, nume, prenume, semnătura decanului
25.09.2026	Prof. dr. ing. Laurentiu- Dan MILICI