SYLLABUS

1. Information regarding the programme

1.1 Higher education institution	**Babeş-Bolyai University**
1.2 Faculty	**Faculty of Mathematics and Computer Science**
1.3 Department	**Department of Computer Science**
1.4 Field of study	**Computers and Information Technology**
1.5 Study cycle	**Bachelor**
1.6 Study programme / Qualification	**Information Engineering**

2. Information regarding the discipline

2.1 Name of the discipline (en) (ro)	**Design Patterns / Șabloane de Proiectare**						
2.2 Course coordinator	**Lect. PhD. Arthur Molnar**						
2.3 Seminar coordinator							
2.4. Year of study	**4**	2.5 Semester	**8**	2.6. Type of evaluation	**C**	2.7 Type of discipline	**Optional DS**
2.8 Code of the discipline		**MLE8115**					

3. Total estimated time (hours/semester of didactic activities)

3.1 Hours per week	5	Of which: 3.2 course	2	3.3 seminar/laboratory	3
3.4 Total hours in the curriculum	70	Of which: 3.5 course	28	3.6 seminar/laboratory	42

Time allotment:	hours
Learning using manual, course support, bibliography, course notes	20
Additional documentation (in libraries, on electronic platforms, field documentation)	3
Preparation for seminars/labs, homework, papers, portfolios and essays	20
Tutorship	10
Evaluations	2
Other activities:	-

3.7 Total individual study hours	**55**
3.8 Total hours per semester	**125**
3.9 Number of ECTS credits	**5**

4. Prerequisites (if necessary)

4.1. curriculum	• Computer programming and programming languages • Object Oriented Programming

4.2. competencies	• Good programming skills in Java or C#

5. Conditions (if necessary)

5.1. for the course	• Lecture hall with projector
5.2. for the seminar /lab activities	• Computers with installed IDE for Java/C# development

6. Specific competencies acquired

Professional competencies	C2.2 Explaining the role, interaction and operation of hardware, software and communication components C3.1 Identifying classes of problems and solving methods that are specific to computing systems C3.3 Applying solution patterns using specific engineering tools and mehods
Transversal competencies	CT1 Honorable, responsible, ethical behavior, in the spirit of the law, to ensure the professional reputation CT3 Demonstrating initiative and pro-active behavior for updating professional, economical and organizational culture knowledge

7. Objectives of the discipline (outcome of the acquired competencies)

7.1 General objective of the discipline	• Enhance students' understanding of software design concepts through a pragmatic approach • Provide students with an environment in which they can explore the usage and usefulness of software design concepts in various business scenarios • Induce a realistic and industry driven view of software design concepts such as design patterns and their inherent benefits
7.2 Specific objective of the discipline	• Give students the ability to explore various object oriented programming languages. • Improve the students abilities to tackle business requirements . • Enhance the students understanding of business needs and business value. • Provide students with insights into ways of working towards achieving high quality software.

8. Content

8.1 Course	Teaching methods	Remarks
1. OOP Principles Recap: Recap presentation that mostly covers main OOP principles such as encapsulation, polymorphism, cohesion, coupling, aggregation, composition	description, explanation, example, case studies, dialogue,debate	
2. SOLID principles: base principles of high quality software: Single responsibility, Open-closed, Liskov substitution, Interface segregation and Dependency inversion		
3. Creational Patterns (Factory, Builder,		

Prototype, Singleton)		
4. Structural Patterns (Adapter, Bridge, Composite)		
5. Structural Patterns (Decorator, Facade, Flyweight)		
6. Structural Patterns (Proxy), Behavioural Patterns (Chain of Responsibility, Command)		
7. Behavioral Patterns (Iterator, Mediator, Memento)		
8. Behavioral Patterns (Observer, State, Strategy)		
9. Behavioral Patterns (Template, Visitor)		
10. Architectural Patterns (MVVM, MVP, MVC)		
11. Antipatterns: common responses to recurring problems that are usually ineffective and risk being highly counterproductive		
12. Dark Patterns		
13. Enterprise Integration Patterns		
14. Recap		

Bibliography

1. M. Fowler – Patterns of Enterprise Application Architecture, Addison Wesley, 2003
2.E. Freeman, E. Freeman, B. Bates – Head First Design Patterns, Oreilly, 2004
3. E. Gamma, R. Helm, R.Johnson, J. Vlissides – Design Patterns Elements of Reusable Object-Oriented Software, Addison Wesley, 1995
4. Robert C. Martin - Clean Architecture: A Craftsman's Guide to Software Structure and Design, Addison Wesley, 2017
5. Alex Xu - System Design Interview -- An Insider's Guide, Byte Code LLC, 2020

8.2 Seminar / laboratory	Teaching methods	Remarks
1. OOP Recap. Introduction to laboratory activities and grading	Explanation, dialogue, case studies	
2. SOLID principles.		
3. Creational design patterns		
4. Structural design patterns.		
5. Behavioural design patterns.		
6. Antipatterns. Dark Patterns.		
7. Architectural Patterns.		

Bibliography

1. M. Fowler – Patterns of Enterprise Application Architecture, Addison Wesley, 2003
2. E. Freeman, E. Freeman, B. Bates – Head First Design Patterns, Oreilly, 2004
3. E. Gamma, R. Helm, R.Johnson, J. Vlissides – Design Patterns Elements of Reusable Object- Oriented Software, Addison Wesley, 1995

8.2 Project	Teaching methods	Remarks
1. Selection of project theme	Explanation, dialogue, presentation	
2. Establishing project requirements		
3. Establishing project requirements (continued)		
4. Project architecture		
5. Project architecture is finalized		
6. Detailed Design, design patterns used		
7. Detailed Design, design patterns used (continued)		
8. First solution prototype		

9. Discussion of suitable design patterns for implementation		
10. Discussion of suitable design patterns for implementation (continued)		
11. Second prototype is completed		
12. Final check-in		
13. Project presentations		
14. Project presentations (continued)		

Bibliography

1. M. Fowler – Patterns of Enterprise Application Architecture, Addison Wesley, 2003
2. E. Freeman, E. Freeman, B. Bates – Head First Design Patterns, Oreilly, 2004
3. E. Gamma, R. Helm, R.Johnson, J. Vlissides – Design Patterns Elements of Reusable Object- Oriented Software, Addison Wesley, 1995
4. Robert C. Martin - Clean Architecture: A Craftsman's Guide to Software Structure and Design, Addison Wesley, 2017
5. Alex Xu - System Design Interview -- An Insider's Guide, Byte Code LLC, 2020

9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program

- The course observes the IEEE and ACM Curricula Recommendations for Computer Science studies.
- The course exists in the study program of all major universities in Romania and abroad.
- The content of the course is considered important for advanced programming skills within the software industry.

10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	Presentation during the semester	Grading based on presentation quality, thoroughness and suitability of examples selected.	25%
	Individual presentation (colloquium)		50%
10.5 Seminar/lab activities	Laboratory project: architecture & design pattern application		25%
Minimum performance standards			
➢ Students must observe the standards of academic integrity. ➢ A minimum passing grade is defined by attaining at least 50% (5/10) points in the final grade.			

Date	Signature of course coordinator	Signature of seminar coordinator
12.05.2022.	Lect. PhD. Arthur Molnar	Lect. PhD. Arthur Molnar

Date of approval

Signature of the head of department

Prof. dr. Laura Dioşan

Diosan

24.05.2022