Far Western University
Faculty of Education
Bachelor in Computer Science Education

Course Title: Web Technology II
Course No.:CS.Ed 367
Semester: Sixth
Level: Undergraduate

Nature : Theory and Practical
Credit Hour:3
Teaching Hrs:48+16

1. Course Introduction

In addition to creating web sites and enhancing their basic programming skills, students will learn to embed PHP in HTML, to interact with MySQL databases through the PHP engine, accessibility issues, and the basics of (secure) file transfers, file management, and web server configuration.

2. Objectives

By the end of this course, students will be able to

- Understand of PHP and programming with PHP
- Work by using MySQL with PHP
- Use very simple regular expressions
- Put all of these ideas together to create web site

3. Specific Objectives and Contents

Specific Objectives	Contents
• Understand and explain importance of PHP • Understand basics of PHP syntax and programming • Embed PHP codes into web pages	**Unit I: PHP Fundamentals (8 Hrs)** 1.1. Introduction: What is PHP?, The history of PHP, What does PHP do?, PHP Installation and Configuration. 1.2. Language Basics: Lexical Structures, Variables, Data Types, Expressions and Operators 1.3. Flow Controls: If, switch, while, for, for each, try...catch, declare, exit, return, goto. 1.4. Including Code, Different styles of Embedding PHP in Web Pages
• Understand and demonstrate functions in PHP • Explain variable scopes, parameters and return values in functions • Handle strings and regular expressions in PHP	**Unit II: Functions Strings (8 Hrs)** 2.1. Defining Function, Calling Function, Variable Scope, Function Parameters, Returning Values, Variable Functions, Anonymous Functions 2.2. String Constants, Printing Strings, Accessing Characters, Cleaning Strings. 2.3. Encoding and Escaping Strings, Comparing Strings, Manipulating and Searching Strings, Regular Expression

• Demonstrate different types of arrays • Apply arrays in writing PHP programs • Understand Objects and other OOP concepts • Use OOP concepts in writing PHP program	**Unit III: Arrays and Objects (7 Hrs)** 3.1. Indexed Arrays, Associative Arrays, Accessing Array Elements, Storing Data, Extracting Multiple Values,multidimensional Arrays. 3.2. Converting between Arrays and Variables, Different Ways of Traversing Arrays, Sorting, Acting on Arrays. 3.3. Creating Objects, Accessing Properties and Methods, Declaring Classes 3.4. Constructors, Destructors, Inheritance, Interfaces, Abstract Classes
• Understand HTTP and Web server basics • Explain GET and POST in form processing • Exemplify file uploading and form validation • Demonstrate sessions and cookies	**Unit IV: Form Processing (7 Hrs)** 4.1. HTTP Basics, Server Variables, Getting Server Information 4.2. PHP Get & POST, Form Processing, Methods, Form Parameters, Form Validation, File Uploads, Setting Response Headers 4.3. Working with cookies, Setting cookie values, Reading cookie values, Unsetting cookie values, Working with sessions, SSL
• Understand MySQL and RDBMS • Connect PHP with MySQL and retrieve data from it • Demonstrate SQL operations by using PHP • Use complex SQL operations through PHP	**Unit V: Database Connectivity (6 Hrs)** 5.1. Using PHP to access Database, Relational Databases and SQL, PHP Data Objects 5.2. MySQL Object Interface, Retrieving Data for Display, SQLite 5.3. Performing basic database operation (DML) (Insert, Delete, Update, Select), Setting query parameter Executing query, 5.4. Cartesian Product and Join Operations, Prepared Statements
• Creating and drawing images suitable for web pages • Embedding images in web pages • Understand and implement security techniques with web pages	**Unit VI: Graphics and Security (6 Hrs)** 6.1. Embedding Images, Basic Graphics Concepts, Creating and Drawing Images, Images with Text 6.2. Dynamically Generated Buttons, Scaling Images, Color Handling 6.3. Security: Filter Input, Cross-Site Scripting, Escape Output, Session Fixation, File Upload, File Access
• Understand the basics of different frameworks and CMS systems used in PHP programs • Use basic functionalities of WordPress.	**Unit VII: Framework and CMS (6 Hrs)** 7.1. Frameworks: Introduction of CodeIgniter, Cake PHP 7.2. CMS: Introduction of WordPress, Joomla, Drupal, Magento 7.3. Wordpress Introduction: Using domain names, Hosting Options, Dashboard, Pages, Directory Permissions, Tags, Settings

4. Methodology and Techniques

Modes of instruction*:* Lecture, seminar, exercise course, guided personal study, tutorial, independent study, project work, Assignments in different topics, group discussion, reflective writing

Types of learning activities

Attending lectures, performing specific assignments, writing papers, independent and private study, reading books, journals and papers, providing constructive feedback, group study and peer discussion.

5. Evaluation Scheme

5.1 Internal Evaluation40%

Internal Evaluation will be conducted by the course teacher based on the following activities.

a) **Attendance and Participation in class activities**: **5+5=10 marks**

b) **Assignment I: Reflective Notes and Class presentation: 5+5=10 marks**

(Reflectivenoteson2to4questionsgivenbyteacherattheend of the every unit and presentation on any two questions among them)

c) **Assignment II**: **One Term paper/Essay/Project and Interview: 5+5=10 marks**

(Logical essay /term paper /project on the topics chosen by students and approved by the teacher, and an interview)

d) **Mid-term exam**: **10 marks**

5.2 External Evaluation (Final Examination) 40%

Types of questions	**Total questions to be asked**	**Number of questions to be answered and marks allocated**	**Total marks**
Group A: Multiple choice items	8 questions	8×1	8
Group B: Short answer questions	6 with 2 'or' questions	6×4	24
Group C: Long answer questions	1 with 1 'or' question	1×8	8

5.3 External Practical Evaluation (20%)

The Office of the Controller of Examinations will conduct the final practical examination at the end of the final examination.

After completing the end-of-semester theoretical examination, a practical examination

will be held. The external examiner will conduct the practical examination according to the following evaluation criteria. There will be an internal examiner to assist the external examiner. Three hours will be given for the practical examination. In this examination, Students must demonstrate their knowledge of the subject matter.

Evaluation System:

Practical	Weightage	Marks
Practical Report Copy	5	
Viva	5	
Practical Exam	10	20

Laboratory Work

Students should write programs and prepare a lab sheet for all of the units in the syllabus. Students
should be able to write PHP scripts by using various concepts discussed in class. Students have to create a dynamic website using core PHP concepts studied in this course.

Text Books

1. Kevin Tatore, Peter MacIntyre, Ramus Lerdorf, Programming PHP, O'Reilly Media, Third Edition, 2013

Reference Books

1. David Sklar, Learning PHP 5, A Pain-Free Introduction to Building Interactive Web Sites**,** O'Reilly Media,
2. Robin Nixon, “Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5”,
3. Luke Welling, PHP and MySQL Web Development, Addison-Wesley Professional O'Reilly Media

Far Western University
Faculty of Education
Bachelor's in Computer Science Education

Course Title: Python for AI
Course No.:CS.Ed. 368
Semester: Sixth
Level: Undergraduate

Nature : Theory and Practical
Credit Hour:3
Teaching Hrs:48+16

1. Course Introduction

This course provides a hands-on introduction to Python programming and its application in basic Artificial Intelligence. Students will learn how to write efficient Python code, understand basic AI concepts, and use popular libraries like NumPy, pandas, matplotlib, and scikit-learn to solve real-world problems. The course is project- and practice-oriented, integrating object-oriented programming and data manipulation techniques for building small intelligent systems.

2. Objectives

- Understand Python basics: syntax, variables, and data types.
- Write Python code using loops, conditionals, and functions.
- Handle files (open, read, write) and use standard modules.
- Use Python libraries and data structures (lists, tuples, sets, dictionaries).
- Apply object-oriented programming concepts: classes, objects, methods, and simple inheritance.
- Develop a problem-solving mindset through coding exercises.
- Build and evaluate simple AI models covering classification clustering and prediction.

3. Specific Objectives and Contents

Specific Objectives	Contents
• Remember core syntax, data types, and operators. • Understand how to run code in different IDEs. • Apply basic I/O in scripts.	**Unit 1: Introduction to Python Programming (5 hrs)** 1.1 Installing Python & IDEs (Anaconda, Jupyter, Google Colab) 1.2 Writing & running code 1.3 Variables, data types, operators 1.4 Input/output & simple programs
• Remember conditional and loop constructs. • Apply control structures to real-world scenarios. • Analyze code flow and evaluate error handling. • Create custom functions.	**Unit 2: Control Structures & Functions (5hrs)** 2.1 Conditional statements: if/elif/else 2.2 Loops: for, while 2.3 Defining & calling functions 2.4 Error handling & basic debugging
• Understand lists, tuples, sets, dictionaries. • Apply file I/O to read/write data. • Apply OOP to model simple entities. • Analyze inheritance hierarchies.	**Unit 3: Data Structures, File Handling & OOP (7 hrs)**

	3.1 Lists, tuples, sets, dictionaries 3.2 File I/O: open/read/write 3.3 Using standard modules (math, random) 3.4 Classes & objects, constructor, methods and basic inheritance in python
• Apply NumPy operations to numerical data. • Analyze tabular data with pandas. • Create plots to evaluate patterns.	**Unit 4: Data Manipulation & Visualization (6hrs)** 4.1 NumPy arrays & operations 4.2 pandas DataFrame basics: loading & cleaning data 4.3 Handling CSV, JSON, and Excel files 4.4 Plotting with matplotlib & seaborn (line, bar, scatter) 4.5 Working with small datasets from Kaggle or UCI
• Remember & understand key AI definitions and types. • Explain Python's role in AI. • Apply basic scikit-learn workflows.	**Unit 5: Introduction to AI (8 hrs)** 5.1 Introduction to AI and its applications 5.2 Types of AI **5.2.1 By Capability:** Narrow (Weak) AI: task-specific systems (spam filters, voice assistants), General (Strong) AI: human-level cognition across domains (theoretical), Super AI: beyond human intelligence (speculative) Concept with simple example. **5.2.2 By Approach:** Rule-based systems ,machine learning: data-driven models(supervised, unsupervised, reinforcement and deep learning) with brief example. 5.3 Role of Python in AI development 5.4 Scikit-learn overview
• Apply preprocessing techniques and evaluate their effect. • Apply train_test_split, KNN, Linear Regression. • Evaluate models with accuracy, precision, confusion matrix. • Understand Naïve Bayes & Decision Trees conceptually.	**Unit 6: Machine Learning with Scikit-learn (12 hrs)** 6.1 Core Algorithms Overview • Linear Regression (prediction) • Decision Trees & Naïve Bayes (classification) • K-Means (clustering) 6.2 Data Preprocessing & Train/Test Split • Imputation, Scaling, Encoding • train_test_split 6.3 Model Development & Evaluation • Model training (.fit) & testing (.predict) • Metrics: accuracy, precision, confusion matrix 6.4 End-to-End Demo • Full pipeline on a small dataset
• Create a mini-AI solution by integrating all steps. • Analyze results. • Evaluate model effectiveness. • Communicate findings clearly.	**7. Unit VII: Mini AI Project (2 hrs)** 7.1 Project selection & dataset exploration 7.2 Preprocessing, model building & evaluation 7.3 Report writing & presentation

4. Methodology and Techniques

Modes of instruction: Lecture, seminar, exercise course, guided personal study, tutorial, independent study, project work, Assignments indifferent topics, group discussion, reflective writing.

Types of learning activities: Attending lectures, performing specific assignments, writing papers, independent and private study, reading books, journals and papers, providing constructive feedback, group study and peer discussion.

5. Evaluation Scheme

5.1 Internal Evaluation 40%

Internal Evaluation will be conducted by the course teacher based on the following activities.

a) Attendance and Participation in class activities: 5+5=10 marks

b) Assignment I: Reflective Notes and Class presentation: 5+5=10 marks

(Reflective notes on 2 to 4 questions given by teacher at the end of every unit and presentation on any two questions)

c)Assignment II: One Term paper/Essay/Project and Interview: 5+5=10 marks
(Logical essay/term paper/project on the topics chosen by students and approved by the teacher and an interview)

d)Mid-term exam: 10 marks

5.2 External Evaluation (Final Examination) 40%

Types of questions	**Total questions to be asked**	**Number of questions to be answered and marks allocated**	**Total Mars**
Group A: Multiple choice items	8 questions	8×1	8
Group B: Short answer questions	6 with2'or'questions	6×4	24
Group C: Long answer questions	1with1'or'question	1×8	8

5.3 Practical Evaluation (20%)

Office of the Controller of Examination will conduct final practical examination at the end of final examination.

After completing the end semester theoretical examination, practical examination will be held. The external examiner will conduct the practical examination according to the following evaluation criteria. There will be an internal examiner to assist the external examiner. Three hours' time will be given for the practical examination. In this examination Students must demonstrate the knowledge of the subject matter.

Evaluation System

Practical	Weightage	Marks
Practical Report Copy	5	20
Viva	5	
Practical Exam	10	

Lab Work

Students should implement the following types of programs under instructor supervision. Aim to complete **10–12** of these over the semester:

1. **Basic Python Programming**
   Variable declarations, and simple I/O.
2. **Control Structures**
   Programs using if/else branches and for/while loops.
3. **Functions & Error Handling**
   Modular functions with parameters/return values, plus basic try/except.
4. **Data Structures**
   Operations on lists, tuples, sets, and dictionaries.
5. **File Handling**
   Reading from and writing to text or CSV files.
6. **Object-Oriented Programming**
   Defining classes, creating objects, and demonstrating simple inheritance.
7. **NumPy Array Operations**
   Creating arrays and computing summary statistics (mean, median, std).
8. **pandas DataFrame Manipulation**
   Loading, cleaning, filtering, and summarizing tabular data.
9. **Data Visualization**
   Plotting line, bar, and scatter charts with matplotlib; boxplots with seaborn.
10. **Machine Learning Pipeline**
    Building a scikit-learn Pipeline for preprocessing and a classifier.
11. **Regression Modeling**
    Training and evaluating a linear regression model (R^2, MSE).
12. **Classification Modeling**
    Training and evaluating a KNN or decision-tree classifier (accuracy, confusion matrix).
13. **Mini-Project Integration**
    End-to-end implementation on a small dataset: preprocessing, model training/testing, evaluation.

Textbooks

1. Downey, A. (2015). Think Python. O'Reilly.
2. Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly.

References Books

1. VanderPlas, J. (2016). Python Data Science Handbook. O'Reilly.
2. Scikit-learn documentation: https://scikit-learn.org
3. Real Python tutorials: https://realpython.com
4. Google Colab: https://colab.research.google.com

Far Western University
Faculty of Education
Bachelor's in Computer Science Education

Course Title: Software Engineering
Course No.:CS.Ed. 369
Semester: Sixth
Level: Undergraduate

Nature : Theory and Practical
Credit Hour:3
Teaching Hrs:48+16

1. Course Introduction

An introduction to engineering principles for building, testing, and maintaining high-quality software systems. Topics include software definitions and types, a rapid recap of life-cycle models, requirements engineering, structured and object-oriented design, architecture and patterns, configuration & change management, testing strategies, and project management. Emphasis is on hands-on use of CASE tools, version control, and team-based mini-projects.

2. Objectives

- Define software engineering and distinguish it from other engineering disciplines.
- Understand and apply software life-cycle models.
- Elicit and specify functional and non-functional requirements.
- Model and design systems using structured and object-oriented techniques.
- Design software architecture and apply design patterns.
- Manage configuration control and versioning.
- Plan and execute testing strategies and quality assurance.
- Estimate and monitor project effort, schedule, and risks.

3. Specific Objectives and Contents

Specific Objectives	Contents
● Recall & Explain life-cycle stages and models. ● Understand software engineer's role and how it differs from other disciplines.	**Unit I: Software Engineering Overview (5 Hrs)** 1.1 Introduction to Software Engineering: definitions, roles, comparison to other engineering fields 1.2 Types of Software: system, application, embedded, domain-specific 1.3 Life-Cycle Models Recap: Waterfall, Iterative, Agile. 1.4 Quality Attributes & Process Frameworks: CMMI, DevOps (overview)
● Explain functional vs non-functional requirements ● Analyze stakeholder needs ● Create an SRS outline.	**Unit II: Requirements Engineering (6 Hrs)**

	2.1 Introduction to Requirements Engineering 2.2 Functional vs Non-Functional Requirements 2.3 Elicitation Techniques: interviews, use-cases, user stories 2.4 SRS Structure & Templates 2.5 Requirements Validation & Traceability
• Model & Design data and processes using structured & OO tools • Map between different representations.	**Unit III: Structured & OO Analysis & Design (8 Hrs)**
	3.1 Introduction to Analysis & Design 3.2 Structured Analysis: DFDs, Context Diagrams 3.3 Data Modeling: ERDs & normalization 3.4 OO Modeling: UML use-case, class, sequence diagrams 3.5 Mapping & Decomposition: DFD to UML
• Apply & Evaluate architectural styles and OO design patterns for robust systems.	**Unit IV: Architecture & OO Design Patterns (8 Hrs)**
	4.1 Introduction to Software Architecture 4.2 Architectural Styles: layered, client-server, microservices 4.3 Component & Deployment Diagrams 4.4 SOLID Principles 4.5 Common Patterns: Singleton, Factory, MVC
• Manage versions and changes effectively using modern tools and workflows.	**Unit V: Configuration & Change Management (5 Hrs)**
	5.1 Introduction to Configuration Management 5.2 Git Fundamentals: clone, commit, push, pull 5.3 Branching & Merging Strategies 5.4 Change Request Workflows 5.5 CI/CD Overview
• Plan & Execute tests at various levels. • Analyze & Evaluate quality metrics.	**Unit VI: Testing Strategies & Quality Assurance (8 Hrs)**
	6.1 Introduction to Testing Strategies 6.2 Black-Box vs White-Box Techniques 6.3 Test Automation Intro (JUnit/PyTest) 6.4 Quality Metrics & Reviews

• Estimate & Monitor project effort. • Schedule and asses project risks. • Apply process improvement.	**Unit VII: Project Management & Process Improvement (8 Hrs)**
	7.1 Introduction to Project Management 7.2 Estimation Techniques: COCOMO II, function points 7.3 Scheduling & Monitoring: Gantt, CPM 7.4 Risk Management & Mitigation 7.5 Process Improvement & Retrospectives

4. Methodology and Techniques

Modes of instruction: Lecture, seminar, exercise course, guided personal study, tutorial, independent study, project work, Assignments indifferent topics, group discussion, reflective writing.

Types of learning activities: Attending lectures, performing specific assignments, writing papers, independent and private study, reading books, journals and papers, providing constructive feedback, group study and peer discussion.

5.Evaluation Scheme

5.1 Internal Evaluation40%

InternalEvaluationwillbeconductedby thecourseteacherbasedon thefollowing activities.

a) Attendance and Participation in class activities: 5+5=10 marks

b) Assignment I: Reflective Notes and Class presentation: 5+5=10 marks

(Reflective notes on 2 to 4 questions given by teacher at the end of every unit and presentation on any two questions)

c)Assignment II : One Term paper/Essay/Project and Interview: 5+5=10marks

(Logical essay/term paper/project on the topics chosen by students and approved by the teacher and interview)

d)Mid-termexam: 10 marks

5.2 External Evaluation (Final Examination) 40%

Types of questions	Total questions to be asked	Number of questions to be answered and Marks allocated	Total marks

Group A: Multiple choice items	8 questions	8×1	8
Group B: Short answer Questions	6 with 2'or' questions	6×4	24
Group C: Long answer Questions	1 with 1'or' question	1×8	8

5.3 Practical Evaluation (20%)

Office of the Controller of Examination will conduct final practical examination at the end of final examination.

After completing the end semester theoretical examination, practical examination will be held. The external examiner will conduct the practical examination according to the following evaluation criteria. There will be an internal examiner to assist the external examiner.Threehours'timewillbegivenforthepracticalexamination.Inthisexamination Students must demonstrate the knowledge of the subject matter.

Evaluation System

Practical	**Weightage**	**Marks**
Practical Report Copy	5	20
Viva	5	
Practical Exam	10	

4. Lab Work

Students should complete 10–12 guided exercises and a team mini-project over 45 practical hours, including:

- Requirements drafting and SRS outline
- DFDs, ERDs, and UML diagrams with CASE tools
- Design pattern pseudocode exercises
- Git version control workflows
- Unit and integration testing exercises
- CI/CD pipeline setup demonstration
- Team mini-project: requirements to prototype

Textbook

1. Sommerville, I. (2021). *Software engineering* (10th ed.). Pearson.
2. Pressman, R. S., & Maxim, B. R. (2020). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill Education.

Reference Books

1. Fowler, M. (2003). *UML distilled: A brief guide to the standard object modeling language* (3rd ed.). Addison-Wesley.
2. Wiegers, K. E., & Beatty, J. (2013). *Software requirements* (3rd ed.). Microsoft Press.
3. Git SCM. (n.d.). *Git documentation*. Retrieved [Month Day, Year], from https://git-scm.com/doc
4. JUnit. (n.d.). *JUnit 5 User Guide*. Retrieved [Month Day, Year], from https://junit.org/junit5/docs/current/user-guide/
5. pytest developers. (n.d.). *pytest documentation*. Retrieved [Month Day, Year], from https://docs.pytest.org/en/stable/
6. Boehm, B. W., Abts, C., & Chulani, S. (2000). *Software cost estimation with COCOMO II*. Prentice Hall.