LNCT University B. Tech CSE

IV SEM

CS 401 Web Technology

COURSE OUTCOMES:

After Completing the course student should be able to

CO1	Use the various html tags to develop the user friendly web pages.
CO2	Use CSS to provide the styles to the web pages at various levels
CO3	Demonstrate characteristics of java scripts for dynamic web pages.
CO4	Install Node JS and mongo DB to deal with data.
CO5	Develop the modern web applications with client side and server side technologies.

Unit 1 Web 3.0 and HTML-Introduction: Concept of WWW, Internet and WWW, HTTP Protocol : Request and Response, Web browser and Web servers, Features of Web 2.0, Web 3.0 Overview:Markup Language,Basic Structure of HTML,Head Section and Elements of Head Section, Meta Tags, CSS Tags, Script Tag, Table Tag , Div Tag ,Anchor tags , Image Tag, ObjectTag,Iframe Tag , Forms , Form Tag , Attributes of Form , POST and GET Method Text input,Text area , Checkbox and Radio Button , Dropdown, List and Opt-group , File Upload and HiddenFields o Submit,HTML Validators,Introduction to HTML5 o Features ofHTML5 , HTML5DocType.

Unit 2 CSS Codes Introduction to Cascading Style Sheets ,Types of CSS , CSS Selectors , Universal Selector IDSelector, Tag Selector , Class Selector , Sub Selector , Child CombinatorySelectorCSS Properties,TypeProperties,BackgroundProperties Block Properties BoxPropertiesListProperties ,Border Properties,PositioningProperties,Real-timeI mplementation,ConverstionofTabletoCSS Layout , CSS MenuDesign(Horizontal, Vertical).

Unit 3 Javascript JQuery Introduction to Client Side Scripting , Introduction to Java Script , JavaScript Types , Variables inJS , Operators in JS , Conditions Statements , Java Script Loops , JS Popup Boxes , JS EventsJSArrays , Working with Arrays , JS Objects , JS Functions , Using Java Script in Real-time RelatedExamples Introduction to jQuery o jQuery Features, Installing jQuery , jQuery Syntax ,jQueryFunctionsand form UIdesigning.

Unit 4 Node JS and Angular JS server-side JS applications. Installing Node JS, Node JS Modules Create,publish,extend,manage,NodeJSHTTP,Express,MongoDBoverviewofstructuralframeworkAngularJS, ReactPHP: Introduction and basic syntax of PHP, decision and looping with examples, PHP and HTML, Arrays, Functions, Browser control and detection, string, Form processing,

UNIT5Moreonwebtechnologies Howwebsiteworksclient,server,uploading,FTP,HTTP,clientserverscripting languages,domainshosting,Intro to CMS word press, joomla,drupal, Case study of full stack development on web

Books Suggested:

- MasteringHTML,CSS,Javascript Web publishing HTML5B lackBook.
- JavaScript:The Definitive Guide by David Flanagan.

Suggested List of Experiments

Lab work includes web page preparation using all technologies in sequent. Static web page development using HTML and CSS, adding dynamism with CSS, Javascript, bootstrap, Node JS, Server side programming and then collectively create a mini project of web application

LNCT University B. Tech CSE

Analysis and Design of Algorithm (CS-402)

COURSE OUTCOMES:
After Completing the course student should be able to

CO1	Evaluate space and time complexity of merge sort algorithms.
CO2	Use greedy strategy to find minimum spanning tree using prim's algorithm.
CO3	Apply backtracking techniques for solving eight-queen problem.
CO4	Implement branch and bound methods to solve traveling salesman problem.
CO5	Solve knapsack problem using dynamic programming algorithm.

Course contents

Unit I: Definitions of algorithms and complexity, Time and Space Complexity; Time space tradeoff, various bounds on complexity, Asymptotic notation, Recurrences and Recurrences solving techniques, Introduction to divide and conquer technique, example: binary search, merge sort, quick sort, heap sort, strassen's matrix multiplication etc, Code tuning techniques: Loop Optimization, Data Transfer Optimization, Logic Optimization, etc.
Unit II : Study of Greedy strategy, examples of greedy method like optimal merge patterns, Huffman coding, minimum spanning trees, knapsack problem, job sequencing with deadlines, single source shortest path algorithm etc. Correctness proof of Greedy algorithms.
Unit III : Concept of dynamic programming, problems based on this approach such as 0/1 knapsack, multistage graph, reliability design, Floyd-Warshall algorithm etc.
Unit IV : Backtracking concept and its examples like 8 queen's problem, Hamiltonian cycle, Graph colouring problem etc. Introduction to branch & bound method, examples of branch and bound method like travelling salesman problem etc. Meaning of lower bound theory and its use in solving algebraic problem, introduction to parallel algorithms.
Unit V : Advanced tree and graph algorithms, NP-hard and NP-complete problems, Approximations Algorithms, Data Stream Algorithms, Introduction to design and complexity of Parallel Algorithms

References:

1. Coremen Thomas, Leiserson CE, Rivest RL, Introduction to Algorithms, Third edition, PHI.
2. Horowitz &Sahani, Analysis & Design of Algorithm, Fourth Edition Computer Science Press.
3. Dasgupta, algorithms, Fifth Edition, TMH
4. Ullmann; Analysis & Design of Algorithm, Addison-wesley publishing company,
5. Michael T Goodrich, RobartoTamassia, Algorithm Design, Wiely India
6. Rajesh K Shukla: Analysis and Design of Algorithms: A Beginner's Approach; Wiley

LNCT University B. Tech CSE

List of Experiments:

1. Write a program for Iterative and Recursive Binary Search.
2. Write a program for Merge Sort.
3. Write a program for Quick Sort.
4. Write a program for Strassen's Matrix Multiplication.
5. Write a program for optimal merge patterns.
6. Write a program for Huffman coding.
7. Write a program for minimum spanning trees using Kruskal's algorithm.
8. Write a program for minimum spanning trees using Prim's algorithm.
9. Write a program for single sources shortest path algorithm.
10. Write a program for Floye-Warshal algorithm.
11. Write a program for traveling salesman problem.
12. Write a program for Hamiltonian cycle problem**.**

LNCT University B. Tech CSE

Software Engineering and Project Management (CS-403)

COURSE OUTCOMES:

After Completing the course student should be able to

CO1	Compare software development models with their merits and demerits.
CO2	Construct software requirement specifications with functional and non-functional.requirements.
CO3	Apply boundary value analysis and equivalence partitioning testing techniques.
CO4	Calculate cyclomatic complexity for given program.
CO5	Apply cocomo model for estimating cost and efforts.

Course contents.

Unit I : The Software Product and Software Process Software Product and Process Characteristics, Software Process Models: Linear SequentialModel, Prototyping Model, RAD Model, Evolutionary Process Models like Incremental Model,Spiral Model, Component Assembly Model, RUP and Agile processes. Software Processcustomization and improvement, CMM, Product and ProcessMetrics

Unit II: Requirement Elicitation, Analysis, and Specification Functional and Non-functional requirements, Requirement Sources and Elicitation Techniques,Analysis Modeling for Function-oriented and Object- oriented software development, Use caseModeling, System and Software Requirement Specifications, Requirement Validation,Traceability

Unit III: Software Design The Software Design Process, Design Concepts and Principles, Software Modeling and UML,Architectural Design, Architectural Views and Styles, User Interface Design, Function- oriented Design, SA/SD Component Based Design, and DesignMetrics.

Unit IV : Software Analysis and Testing Software Static and Dynamic analysis, Code inspections, Software Testing, Fundamentals,Software Test Process, Testing Levels, Test Criteria, Test Case Design, Test Oracles, TestTechniques, Black-Box Testing, White-Box Unit Testing and Unit, Testing Frameworks,Integration Testing, System Testing and other Specialized, Testing, Test Plan, Test Metrics,Testing Tools. , Introduction to Object-oriented analysis, design and comparison with structuredSoftwareEngg.

Unit V : Software Maintenance & Software Project Management Need and Types of Maintenance, Software Configuration Management (SCM), Software ChangeManagement, Version Control, Change control and Reporting, Program ComprehensionTechniques, Re-engineering, Reverse Engineering, Tool Support. Project Management Concepts, Feasibility Analysis, Project and Process Planning,Resources Allocations, Software efforts,Schedule, and Cost estimations, Project Scheduling and Tracking, Risk Assessment andMitigation, Software Quality Assurance (SQA). Project Plan, ProjectMetrics.

References:

1. Pankaj Jalote ,"An Integrated Approach to Software Engineering", NarosaPub, 2005
2. RajibMall, "Fundamentals of Software Engineering" Second Edition, PHI Learning
3. R S. Pressman,"Software Engineering: A Practitioner's Approach", Sixth edition 2006, McGraw-Hill.
4. Sommerville,"SoftwareEnginerring",PearsonEducation.
5. Richard H.Thayer,"SoftwareEnginerring& Project Managements",WileyIndia
6. Waman S.Jawadekar,"Software Enginerring",TMH
7. BobHughes, M.Cotterell, RajibMall"SoftwareProjectManagement", McGrawHill

Practical and Lab work :

Lab work should include a running case study problem for which different deliverable sat the end of each phase of a software development life cycle are to be developed. This will includemodeling the requirements, architecture and detailed design. Subsequently the design models willbe coded and tested. For modeling, tools like Rational Rose products. For coding and testing, IDElike Eclipse, Net Beans, and Visual Studio can beused.

LNCT University B. Tech CSE

Compiler Design(CS-404)

COURSE OUTCOMES:
After Completing the course student should be able to

CO1	Develop an understanding of the basic theory underlying the different components and phases of a compiler.
CO2	Develop CFG and parse tree for given pseudocode
CO3	Develop an understanding for implementation of run time environment.
CO4	Apply tools for parse generators and code generators
CO5	Evaluate code optimization and data flow

Unit-I Introduction to compiling & Lexical Analysis Introduction of Compiler, Major data Structure in compiler, types of Compiler, Front-end and Back-end of compiler, Compiler structure: analysis-synthesis model of compilation, various phases of a compiler,Lexical analysis: Input buffering , Specification & Recognition of Tokens,Design of a Lexical Analyzer Generator, LEX.

Unit-II Syntax Analysis &Syntax Directed Translation Syntax analysis: CFGs, Top down parsing, Brute force approach, recursive descent parsing,transformation on the grammars, predictive parsing, bottom up parsing, operator precedence parsing, LR parsers (SLR,LALR, LR),Parser generation. Syntax directed definitions: Construction of Syntax trees,Bottom up evaluation of S- attributed definition, L-attribute definition, Top down translation, Bottom Up evaluation of inherited attributes Recursive Evaluation, Analysis of Syntax directed definition.

Unit-III Type Checking & Run Time Environment Type checking: type system, specification of simple type checker, equivalence of expression, types, type conversion, overloading of functions and operations, polymorphic functions. Run time Environment:storage organization, Storage allocation strategies, parameter passing, dynamic storage allocation , Symbol table, Error Detection & Recovery, Ad-Hoc and Systematic Methods.

Unit –IV Code Generation Intermediate code generation: Declarations, Assignment statements, Boolean expressions, Case statements, Back patching, Procedure calls Code Generation: Issues in the design of code generator, Basic block and flow graphs, Register allocation and assignment, DAG representation of basic blocks, peephole optimization, generating code from DAG.

Unit –V Code Optimization Introduction to Code optimization: sources of optimization of basic blocks, loops in flow graphs, dead code elimination, loop optimization, Introduction to global data flow analysis, Code Improving transformations ,Data flow analysis of structure flow graph Symbolic debugging of optimized code.

References:

1.Raghavan, ompilerDesign, TMH Pub.
2. A. V. Aho, R. Sethi, and J. D. Ullman. Compilers: Principles, Techniques and Tools , Pearson Education
3. Louden. Compiler Construction: Principles and Practice, Cengage Learning
4. A. C. Holub. Compiler Design in C , Prentice-Hall Inc., 1993.
5. Mak, writing compiler & Interpreters, Willey Pub.

Suggested List of experiments

- Implementation of LEXICAL ANALYZER for IF STATEMENT.
- Implemention of LEXICAL ANALYZER for ARITHMETIC EXPRESSION.
- Construction of NFA from REGULAR EXPRESSION.
- Construction of DFA from NFA.
- Implementation of SHIFT REDUCE PARSING ALGORITHM.
- Implementation of OPERATOR PRECEDENCE PARSER.
- Implementation of RECURSIVE DESCENT PARSER.
- Implemenation of CODE OPTIMIZATION TECHNIQUES.
- Implementation of CODE GENERATOR.
- Design Predictive Parser for the given language

LNCT University B. Tech CSE

Programming Elective - II Adv Java Programming (CS405 a-PE)

COURSE OUTCOMES:

After Completing the course student should be able to

CO1	Learn and practice Servlet and Jsp
CO2	Implement Modules of Spring Framework.
CO3	Create session factory in hibernate
CO4	Discuss the 3 logical layers of MVC
CO5	Apply RESTful web services and REST architecture to related domain

Course contents

UNIT-1: Servlet and JSP Servlet: Introduction to Servlets, Servlet lifecycle, Setting up a servlet environment, Writing and deploying a simple servlet, Servlet configuration (web.xml), Servlet context and config objects,RequestDispatcher and forward/include, Session management, Servlet filters.JSP: Introduction to JSP, JSP lifecycle, Writing and deploying a simple JSP, JSP scriptingelements (declarations, scriptlets, expressions), JSP directives , JSP implicit objects, JSP actiontags, JSP and JavaBeans,

UNIT-II: Spring and Spring boot introduction Introduction to Spring Framework, Spring architecture, Inversion of Control and Dependency Injection , Spring Bean lifecycle, Bean scopes and configurations, Spring Core annotations, ApplicationContext and BeanFactory.Introduction to Spring Boot, Differences between Spring and Spring Boot, Setting up a Spring Boot environment, Spring Boot starters, Spring Boot auto-configuration, Creating a Spring Boot application, Spring Boot annotations

UNIT-III: JPA/Hibernate Introduction to Spring Boot JPA/Hibernate, Setting up Spring Boot with JPA/Hibernate, Entity classes and mapping annotations (@Entity, @Table, @Column, etc.), Primary keys and generated values (auto,sequence, table), Relationships (One-to-One, One-to-Many, Many-to-One, Many-to-Many), Fetch types (Eager vs Lazy loading), JPQL, Native SQL queries with Hibernate, Spring Data JPA repositories, Query methods andcustom queries, Transaction management with JPA/Hibernate**.**

UNIT-IV: Model-View-Controller Introduction to Spring Boot MVC, Setting up a Spring Boot MVC application, Spring Boot MVC architecture (Model, View, Controller), Spring Boot MVC annotations, Handling HTTP requests and responses, Request parameters and path variables, Model attributes and forms, Thymeleaf template engine integration, Spring Boot REST controllers, JSON responses with Jackson, database integration, Exception handling in Spring Boot MVC**.**

UNIT-V: RESTful Webservice Introduction to RESTful web services, Basics of REST architecture (resources, URIs, HTTP methods), Introduction to Spring Boot RESTful services, Creating RESTful controllers with Spring Boot, Spring Boot annotations for REST, Request and Response entities (DTOs),Handling GET, POST, PUT, DELETE requests, Path variables and query parameters, Contentnegotiation (JSON, XML), Pagination and sorting in RESTful services.

Reference Books

- Cays Horstmann,Core Java vol-1,9 /e,Pearson education 2012
- Herbert schildt, The Complete Reference ,9/e TaTa Mc GrawHill 2014
- ScottIn Amber , The Object Primer,3/e, Cambridge 2004

Suggested List of experiments

Create spring boot application , implement spring framework with all possible options, set up MVC and simulate REST services

LNCT University B. Tech CSE

Elective II - Advance Python Programming (405 b)

COURSE OUTCOMES:

After Completing the course student should be able to:

CO1	Object oriented programming concept with python.
CO2	Working GUIs in Python
CO3	Introduction to web Framework- Django
CO4	Understand the views, models, forms of Django
CO5	Frontend integration and API Development with Django

Course Contents

UNIT – I Introduction to OOPs, Class definition and other operation in the class, objects, attributes, methods, constructor, polymorphism, inheritance, abstraction, encapsulation, decorators, generators.

UNIT – II Introduction to GUI programming, components and events, root component, different geometry methods, widgets such as Button, Entry, Label, Message Box, Text, Check Buttons and their functionality, Django Template, DTL.

UNIT – III Framework, introduction to Django, History, features, environment setup, virtual environment, MVC architecture vs MVT architecture, DJango MVT pattern, Create first project, directory structure, files inside the project, project vs app, Migrations, URL, URL mapping.

UNIT – IV Views in Django, Class based view, function based views, connect views with URL, Generic views, custom views, Middleware, Admin View, Django Forms, Form handling, custom validation, CSRF tokens, form security, custom template tags and filters.

UNIT – V Integration Django with Frontend, RESTful APIs with Django REST framework: (Serializers, ViewSets, Routers), Authentication and permissions

References:

1. Allen B. Downey, "Think Python: How to Think Like a Computer Scientist", 2nd Edition, Green Tea Press, 2015, ISBN: 978-9352134755.
2. Charles R. Severance, "Python for Everybody: Exploring Data Using Python 3", 1 st Edition, Shroff Publishers, 2017. ISBN: 978-9352136278.
3. Django for Beginners: Build websites with Python and Django, William S. Vincent, 2018
4. Lightweight Django, Mark Lavin, 2014.

LNCT University B. Tech CSE

LIST OF EXPERIMENTS

1. Define a class Student with attributes name and marks. Include methods to assign marks and display the student's name and marks. Write a Python script to create multiple student objects, assign marks to each, and display their information.
2. Define a class Animal with a method speak(). Create two subclasses Dog and Cat, each with its own implementation of the speak() method. Write a Python script to demonstrate polymorphism by calling the speak() method on objects of both subclasses.
3. Create a Django project with an app that includes a form for user registration (username, email, passwo rd). Define a template to display the form and handle form submission in a view. Display a success message upon successful form submission.
4. Write a Python script to create a GUI inwdow with a Label and a Button. When the button is clicked, display a messag e box with the text from the Label.
5. Define a simple model Book with fields title and author in the MyApp app. Create and apply migrations for the model. Document the commands used and verify the changes in the database.
6. Create a Django template in MyApp that displays a list of books. Define a view that passes a list of book titles to the template. Map the view to a URL and displ ay the list in the browser.
7. Create a custom template tag that returns the current date and time. Create a custom template filter th at converts text to uppercase. Use these custom tags and filters in a template and test them in the browser.
8. Add custom validation to the Post form to ensure that the title is at least 10 characters long. Display appropriate error mes sages for invalid input. Test the validation by submitting the form with various inputs.
9. Modify the Article serializer to include a method field that returns the word count of the article content. Customize the vie wset to filter articles based on the author's name. Test the customized API endpoints.
10. Create a Django project and app named FrontendApp. Develop a simple HTML form that allows users to submit their names. Use Dj ango views to process the form and display a greeting message with the submitted name. Style the form using CSS and add basic JavaScript validation.
11. Define a class named Circle with an attribute radius. Include methods to calculate the area and circumference of the circle. Write a Python script to create an instance of the class, set the radius, and display the area and circumference.
12. Define a class Book with a constructor that initializes the attributes title, author, and price. Write a Python script to cre ate multiple book objects and display their details.
13. Write a Python script to create a GUI window with a button. When the button is clicked, display a message "Button Clicked!" in the console.
14. Write a Python script to create a GUI window with three buttons arranged using the pack(), grid(), and place() geometry metho ds. Experiment with each method to understand their behavior.
15. Define a simple model Book with fields title and author in the MyApp app. Create and apply migrations for the model. Document the commands used and verify the changes in the database.
16. Define a view in MyApp that returns a simple HttpResponse with the text "Hello, Django!". Map this view to a URL in the urls.py file of MyApp. Test the URL mapping by running the Django development server and accessing the URL in a browser.
17. Create a Django project and app named MyApp. Define a function-based view that returns a simple HttpResponse with the text "Welcome to Function-Based Views!". Connect this view to a URL and test it in the browser.
18. Define a custom view that filters posts based on a keyword passed as a URL parameter. Display the filtered lis t of posts. Connect this view to a URL and test it in the browser.
19. Create a form for the Post model using Django's forms.ModelForm. Define a view to handle form submission and save the data to the database. Connect the view to a URL and test the form in the browser.
20. Create a custom middleware that logs the URL of each request to the console. Add this middleware to the MIDDLEWARE setting an d test it by accessing different URLs in the browser.

INNOVATION PRACTICES (CS-406)

The course is designed to give an overview on various aspects of innovation, creativity, design thinking, business models, incubation and entrepreneurship.

Case studies: Expert Lectures, MOOCs to Analyze the Current Business Scenario, Innovation and Creativity, Innovation in Current Environment, Types of Innovation , Hackathons, IPRs-Patents, copyrights, filing and design process, KAPILA scheme

Design thinking : A systematic method of solving problems. This method is unique that it starts and ends with humans. The design thinkers start by observing, interviewing or just plain experiencing a situation. Then they proceed to improve the situation of the humans by solving problems for them.

Activates Suggested:

Virtual Lab Creative Design, Prototyping & Experiential Lab

Swayam/NPTEL Course on dEsign thinking, Innovation

Industry Modules on Design Thinking Infosys,Wipro, CISCO

Note : Submission of report at EO