

Department of CSE, SDBCT
CS-606[Skill Development LAB]

SUSHILA DEVI BANSAL COLLEGE OF ENGINEERING, INDORE



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

LAB MANUAL

SUBJECT	Skill Development Lab
SUBJECT CODE	CS-606
SEMESTER	VIth
ENROLLMENT NO.	0829CS211062
SESSION	JAN-JUNE 2023

Prepared By:
Prof. Gouri Thakur

Department of CSE, SDBCT
CS-606[Skill Development LAB]

INDEX

S. NO	List of Experiment	Page No
1	Study of Software Product Life Cycle.	
2	Study of Agile Development Process.	
3	Study of Software Product Development Standards.	
4	Study of Software Design Patterns.	
5	Study of Architectural Patterns.	
6	Design UML Diagrams of a project using tools (Library Management System)	
7	Design Activity Diagrams of a project using tools (Library Management System)	
8	Design ER Diagrams of a project using tools (Library Management System)	
9	Design Sequence Diagrams of a project using tools (Library Management System)	
10	Implement Testing of a project and draw flow diagram of it (Library Management System)	
11	Design Class Diagrams of a project using tools (Library Management System)	

Department of CSE, SDBCT
CS–606[Skill Development LAB]

HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENTS:

- INTEL i3 or i5 processor
- 320GB HDD
- 4GB RAM

SOFTWARE REQUIREMENTS:

- Lucid or any editor to design graphics
- Online Draw.io tool
- Notepad

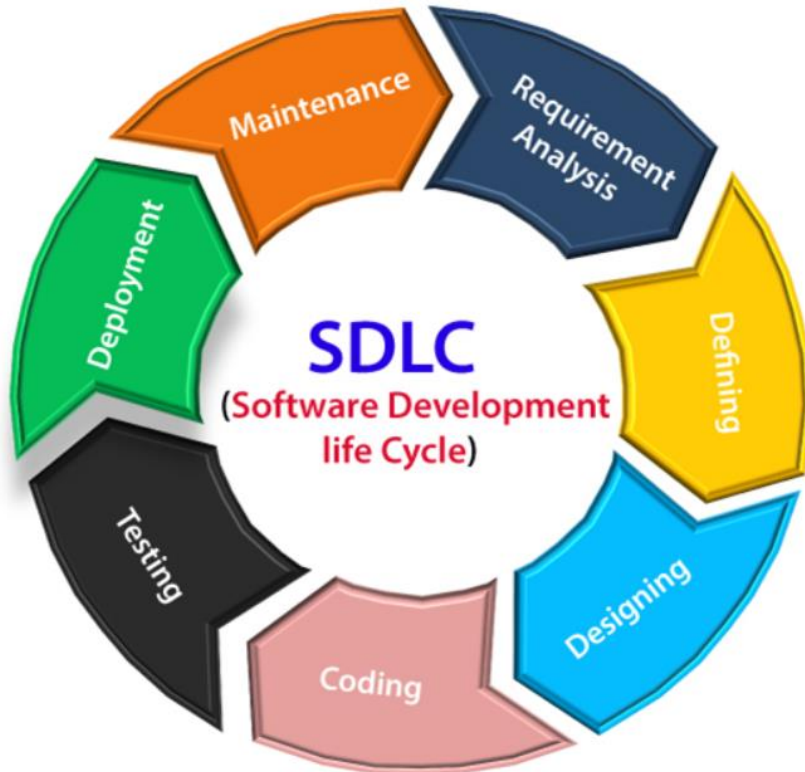
Department of CSE, SDBCT
CS-606[Skill Development LAB]

1.Study of Software Product Life Cycle.

Objective: Study of Software Product Life Cycle.

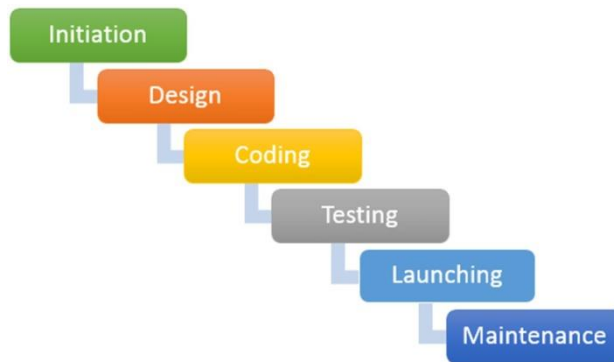
Introduction: Software development life cycle (SDLC) is a structured process that is used to design, develop, and test good-quality software. SDLC, or software development life cycle, is a methodology that defines the entire procedure of software development step-by-step.

The goal of the SDLC life cycle model is to deliver high-quality, maintainable software that meets the user's requirements. SDLC in software engineering models outlines the plan for each stage so that each stage of the software development model can perform its task efficiently to deliver the software at a low cost within a given time frame that meets users' requirements.



Product Development is the process of Designing, Building, Operating and Maintaining a Good Service. The life cycle of a Software Product is somehow similar to the life cycle of Software with some differences.

Department of CSE, SDBCT
CS-606[Skill Development LAB]



There are 7 Phases In Software Product Development Life Cycle :

1. **Planning:** This phase involves defining the project scope, goals, and requirements. It includes activities such as gathering requirements from stakeholders, analyzing feasibility, creating a project plan, and allocating resources.
2. **Analysis:** In this phase, the requirements gathered in the planning phase are analyzed in detail. The goal is to understand the needs of end-users and define the system's functionality. This may involve creating use cases, data models, and prototypes.
3. **Design:** During the design phase, the system architecture is planned and detailed. This includes defining the software's overall structure, interface design, data architecture, algorithms, and other technical specifications.
4. **Implementation (Coding):** In this phase, the actual coding of the software begins based on the design specifications. Developers write, test, and debug the code according to coding standards and best practices. This phase also involves integrating individual components and modules into a complete system.
5. **Testing:** Testing is a crucial phase where the software is evaluated to ensure it meets the specified requirements and functions correctly. Various testing techniques are employed, including unit testing, integration testing, system testing, and acceptance testing. Bugs and defects are identified and fixed during this phase.
6. **Deployment:** Once the software has been thoroughly tested and approved, it is deployed to the production environment. This involves installing the software, configuring it for use, and training end-users if necessary.
7. **Maintenance:** The maintenance phase involves ongoing support and updates to the software. This includes fixing bugs, addressing user feedback, making enhancements, and ensuring the software remains compatible with evolving technologies and requirements.

Result: The study of Software Product Life Cycle has been done successfully.

Department of CSE, SDBCT

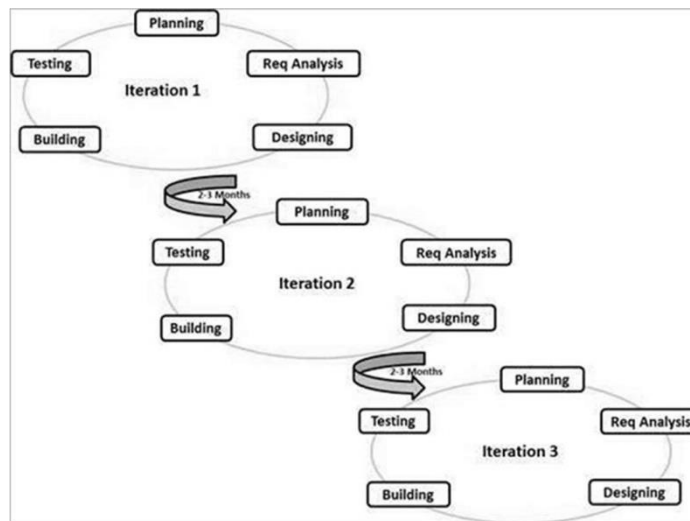
CS-606[Skill Development LAB]

2. Objective : Study of Agile Development Process.

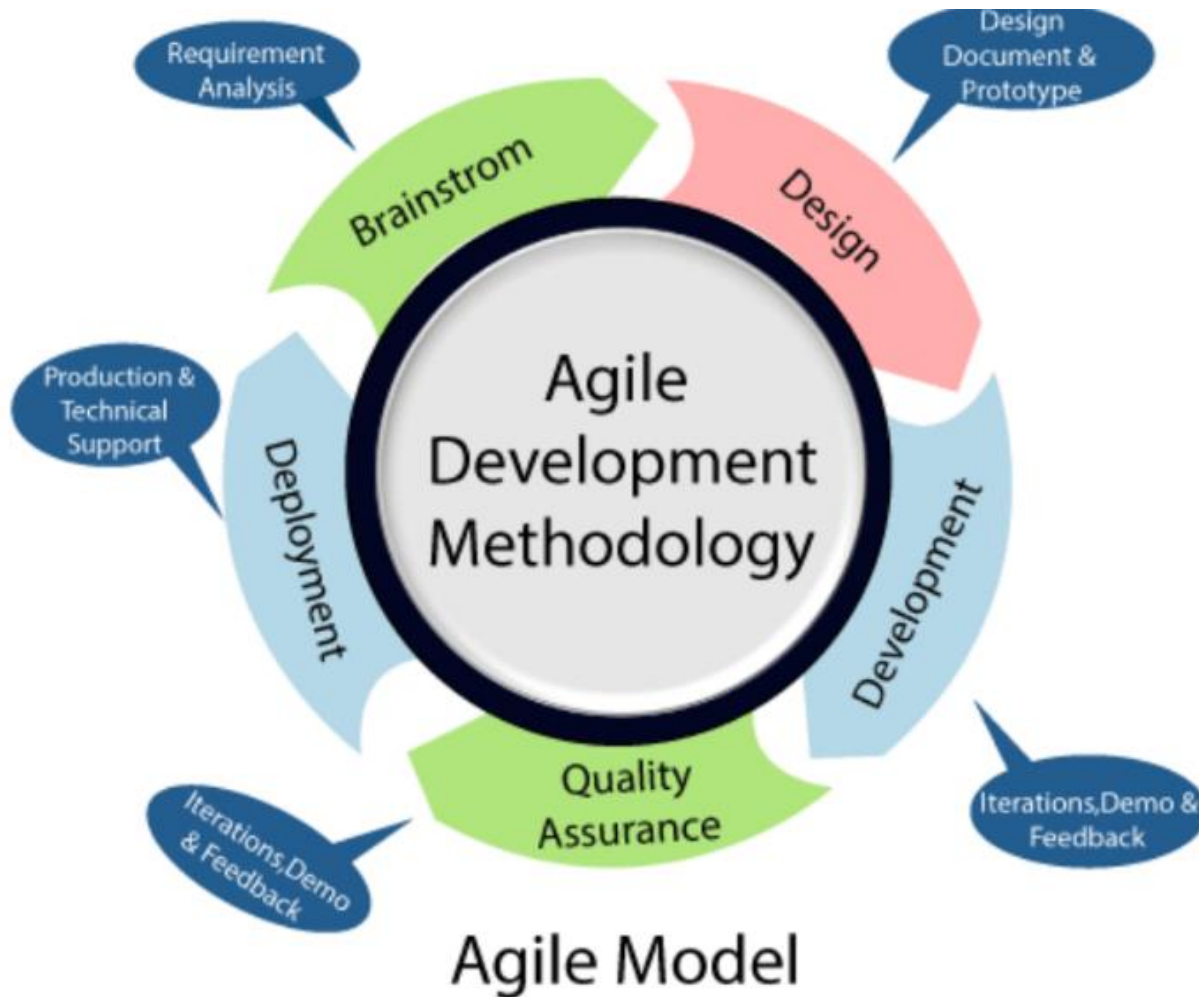
Introduction: Agile Software Development comprises various approaches under which Requirements and Solutions evolve through collaborative effort of Self Organizing and Cross Functional Teams and their End Users. It incorporates Adaptive Planning, Evolutionary Development, Early Delivery and Continual Improvement and encourages Rapid and Flexible Response to changes.

It is an Iterative and Incremental approach. It focuses on “Individuals and Interactions” over “Processes and Tools”, “Working Software” over “Comprehensive Documentation”, “Customer Collaboration” over “Contract Negotiation” and “Responding to a Change” over “Following a Plan”. Its basic Principles include Customer Satisfaction, Response to Changes, Faster Development, Collaboration of Developer and End Users, Simplicity, Focus on Quality and many more.

Agile SDLC Model: Agile Model breaks the Product into small Incremental Builds. These Builds are provided in Iterations. Every Iteration involves Teams from Cross Functional areas like Planning, Analysis, Design, Coding and Testing. At the end of every Iteration, a Working Product is displayed to the Customers and important Stakeholders. Agile Model believes that every Project needs to be handled differently.



Department of CSE, SDBCT
CS-606[Skill Development LAB]



The Phases of Agile Model are:

- **Requirement Gathering:** In this Phase, Requirements are defined. Business Opportunities, Planning, Effort and Development Time are also analyzed during this Phase. Based on this Information, Technical and Economic Feasibilities are evaluated.
- **Design:** Once the Requirements are finalized, Design Phase is initiated. Data Flow Diagrams and UML Diagrams are developed to specify the Operations of the Product.
- **Construction:** Incremental and Iterative Process of developing the Final Product.
- **Testing:** Quality Assurance Team examines the Product in this Phase. IT looks for Bugs and ensures Performance of the developed Product.

Department of CSE, SDBCT
CS-606[Skill Development LAB]

- **Deployment:** In this Phase, Product is released for operating in User's Environment and Platforms.
- **Feedback:** The last step after Release is Feedback. Development Team receives the Feedback about the Product and works on that Feedback for improvement.

Result: The study of Agile Development Process has been done successfully.

Department of CSE, SDBCT

CS-606[Skill Development LAB]

3.Introduction: A Software Standard is a standard Protocol or other common Format of a Document, File or Data Transfer accepted and used by one or more Software Developers while working on one or more Computer Programs. It enables Interoperability between different Programs developed by different Developers.

Software Standard consists of certain Terms, Concepts, Data Formats, Document Styles and Techniques agreed upon by Software Developers, so that their Software can understand the Data and Files created by other Computer Programs. A certain Protocol needs to be accepted and incorporated by a group of Developers who contribute to the Definition and Maintenance of the Standard to be considered as a Standard. Standard ensures Efficiency in Code Development, wider User Acceptance and increasing Applicability of the Software being developed.

HTML, TCP / IP, SMTP, FTP, XML and JSON are Software Standards that Application Designers must understand and follow, if their Software expects to interface with these Standards. Word and Excel Files are not still considered as a Standard, even when they are being widely used, only because of being Proprietary to Microsoft. W3C and ISO are some organizations that specify the Standards for Software.

A Standard can be a Closed Standard or an Open Standard. The Documentation for an Open Standard is Open to the Public and anyone can create Software that implements and uses the Standard. The Documentation and Specification for a Closed Standard is are not available to the Public. Standards ensure the Quality of Software and Products.

Software Quality: Quality of Software improves significantly due to Standards. This enhances Reliability and Usability of Software and reduces the Maintenance Efforts. Quality Management has three main activities as Quality Assurance, Quality Planning and Quality Control. It provides a check on Software and SDLC Process. It ensures that Deliverables must be Consistent with Standards.

Quality Assurance: Quality Assurance ensures the Development of a Framework of Organizational Procedures and Standards that lead to High Quality Software. It is the process of defining how Software Quality can be achieved and how the Development Organization knows that the Software has the required level of Quality.

Department of CSE, SDBCT

CS-606[Skill Development LAB]

Quality Planning: Quality Planning ensures the Selection of appropriate Procedures and Standards from this Framework and adapt for a specific Software Project. It is the process of developing a Quality Plan for the Project. The Quality Plan defines the Quality Requirements of Software and describes how these are to be assessed. Quality Plan specifies Introduction of Product, Product Plans, Process Descriptions, Quality Goals and Risks and Risk Management.

Quality Control: Quality Control ensures the Definition of Processes ensuring that Software Development follows the Quality Procedures and Standards. Quality Reviews are the most widely used method of validating the Quality of Software or a Product. Software Measurement specifies the Quality of a Software Product. Metrics are used to represent the Quality of Software or Products.

Documentation Standards: They are also important because they represent the Software and Software Processes. They must have a Consistent Appearance, Structure and Quality and therefore, be easy to Read and understand. There are three types of Documentation Standards, as:

- **Documentation Process Standards:** They define Process that should be followed for Document Production.
- **Document Standards:** They describe the Structure and Presentation of Documents.
- **Document Interchange Standards:** They ensure that all Electronic Copies of Documents are Compatible.

Software Development Best Practices: Best Practices are a set of proven approaches for Software Development. They are Industry Standards. Some Best Practices include:

- Develop iteratively to handle Risks effectively
- Manage Requirements, as they are prone to change with time
- UseComponentbasedArchitecturetoimproveMaintainability,Extensibilityand Reusability
- VisuallyModelSoftwareforClarityregardingInterfaceandRequirements
- VerifySoftwareQualitywithrespecttoFunctionality,PerformanceandReliability
- Control Changes to Software

Result: Study of Software Product Development Standards has been done successfully.

Department of CSE, SDBCT
CS-606[Skill Development LAB]

4.Introduction: In Software Engineering, a Design Pattern is a Reusable Solution to a commonly occurring Problem. It is not a Finished Design that can be transformed to Source or Machine Code; rather it is a Description or a Template to solve a Problem. Design Patterns gained popularity in 1994 by Erich Gamma, Richard Helm, Ralph Johnson and John Vlissides. They can speed up the Development Process by providing Tested and proven Development paradigms.

Types of Design Patterns: They are classified as:

Creational Patterns: They provide the capability to create Objects based on required criteria and in a controlled way. These include:

S.No.	Name	Specification
1	Abstract Factory	It provides an Interface for creating Families of related and dependent Objects without specifying their Concrete Classes.
2	Builder	It separates the Construction of a Complex Object from its Representation, allowing the same Construction process to create various Representations.
3	Object Pool	It avoids expensive Acquisition and Release of Resources by recycling Objects that are no longer in use.
4	Prototype	It specifies the kind of Objects to create using a Prototypical Instance and create new Objects from the Skeleton of an existing Object, thus enhancing Performance and reducing the Memory Requirements.
5	RAII	Resource Acquisition Is Initialization (RAII) ensures that Resources are properly released by tying them to the life span of Objects.
6	Singleton	It ensures that a Class has only one Instance and provides a Global point of Access to it.

Department of CSE, SDBCT
CS-606[Skill Development LAB]

Structural Patterns: They organize different Classes and Objects to form larger structures and provide new Functionality. These include:

S.No.	Name	Specification
1	Bridge	It decouples the Abstraction from its Implementation, thus allowing the two to vary independently.
2	Front Controller	It relates to the Design of Web Applications and provides a Centralized Entry Point for handling Requests.
3	Marker	It is an Empty Interface to associate MetaData with a Class.
4	Module	It groups several related Elements, such as Classes, Singletons, Methods, etc. into a single Conceptual Entity.
5	Proxy	It provides a Placeholder for another Object to control access to it.
6	Twin	It allows Modeling of Multiple Inheritance in Programming Languages that does not support this feature.

Behavioral Patterns: They identify Common Communication Patterns between Objects and realize these Patterns. These include:

S.No.	Name	Specification
1	Black Board	It combines disparate sources of Data using AI approaches.
2	Interpreter	It uses Representation to Interpret Sentences in a Language.
3	Iterator	It provides a way to access the Elements an Aggregate Object sequentially without exposing its underlying representation.
4	Mediator	It defines an Object that encapsulates how a set of Objects interact and promotes Loose Coupling.
5	NullObject	It avoids NullReferences by providing Default Object.
6	Servant	It defines Common Functionality for a group of Classes and also known as Helper Class or Utility Class and have all Static Methods and no Objects.
7	State	It allows an Object to alter its Behavior when its Internal State changes.

Department of CSE, SDBCT
CS-606[Skill Development LAB]

S.No.	Name	Specification
8	Strategy	It defines a family of Algorithms, encapsulates the mandmake them interchangeable.
9	Template Method	It defines the Skeleton of an Algorithm in operation and let Sub Classes redefine some steps without changing the Algorithm's Structure.

Apart from these, there are certain Concurrency Patterns like Lock, Schedulers, etc. that are being used while performing Concurrent Operations, Resource Sharing, Inter Process Communication and Synchronization.

Result: Study of Software Design Patterns has be end one successfully.

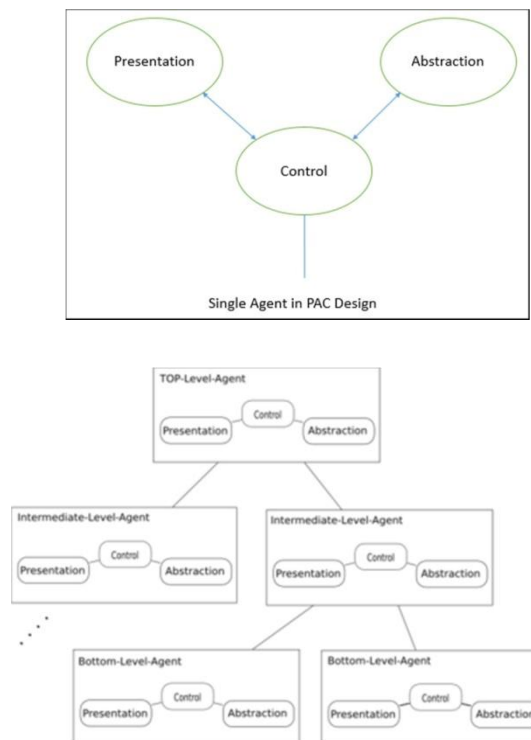
Department of CSE, SDBCT
CS-606[Skill Development LAB]

5.Introduction: Software Architectural Pattern is a Reusable Solution to a commonly occurring Problem in Software Architecture. It is similar to Software Design Pattern, but has a broader scope. It addresses Issues like Hardware Performance Limitations, High Availability and Minimization of Risks. It is a concept that solves some essential cohesive elements of a Software Architecture. A Software Architectural Style is a specific method of Construction.

An Architectural Style defines a Family of Systems in terms of Patterns of Structured Organization, a Vocabulary of Components and Connectors with constraints on how they can be combined. It is a named collection of Architectural Design Decisions that are applicable in a given Development Context.

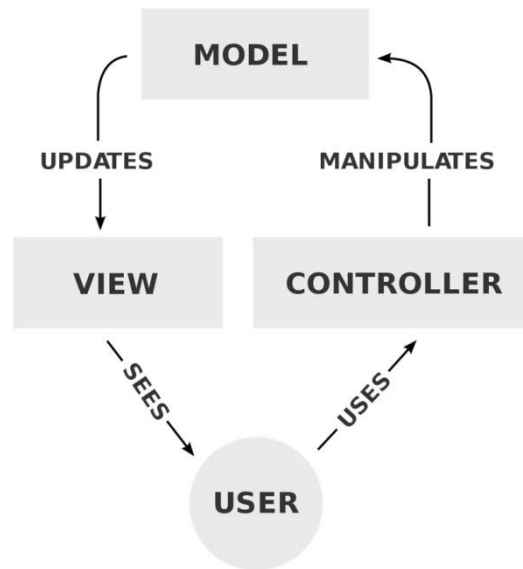
PAC: Presentation – Abstraction – Control (PAC) is an Interaction Oriented Software Architecture that separates an Interactive System into three types of Components, viz., the Abstraction Component that retrieves and processes the Data, the Presentation Component formats the Visual and Audio Presentation of Data and the Control Component handles Issues such as Flow of Control and Communication between the other two Components.

It uses hierarchical structure of Agents, each consisting of Presentation, Abstraction and Control parts. Agents communicate with each other through the Control Part only.



Department of CSE, SDBCT
CS-606[Skill Development LAB]

MVC: Model – View – Controller (MVC) is commonly used for developing User Interfaces that divides the related Programming Logic into three interconnected Elements. It separates the Internal Representation of Information from External Representation presented and accepted to User. It is commonly used in JAVAScript, Python, PHP, JAVA and C# for developing Web and Mobile Applications.



Model: It is the central component and application's dynamic Data Structure, independent of UI. It manages Data, Logic and Rules of the Application. It receives User Input from Controller.

View: It is Representation of Information in form of Charts, Tables and Diagrams. There can be multiple views of the same Information.

Controller: It accepts Input and converts it to Commands for Model or View. It responds to the User Inputs and performs Interactions on the Data Model Objects. It receives Input, optionally validates it and passes it to the Model.

It is widely used in Web Development. It supports Simultaneous Development.

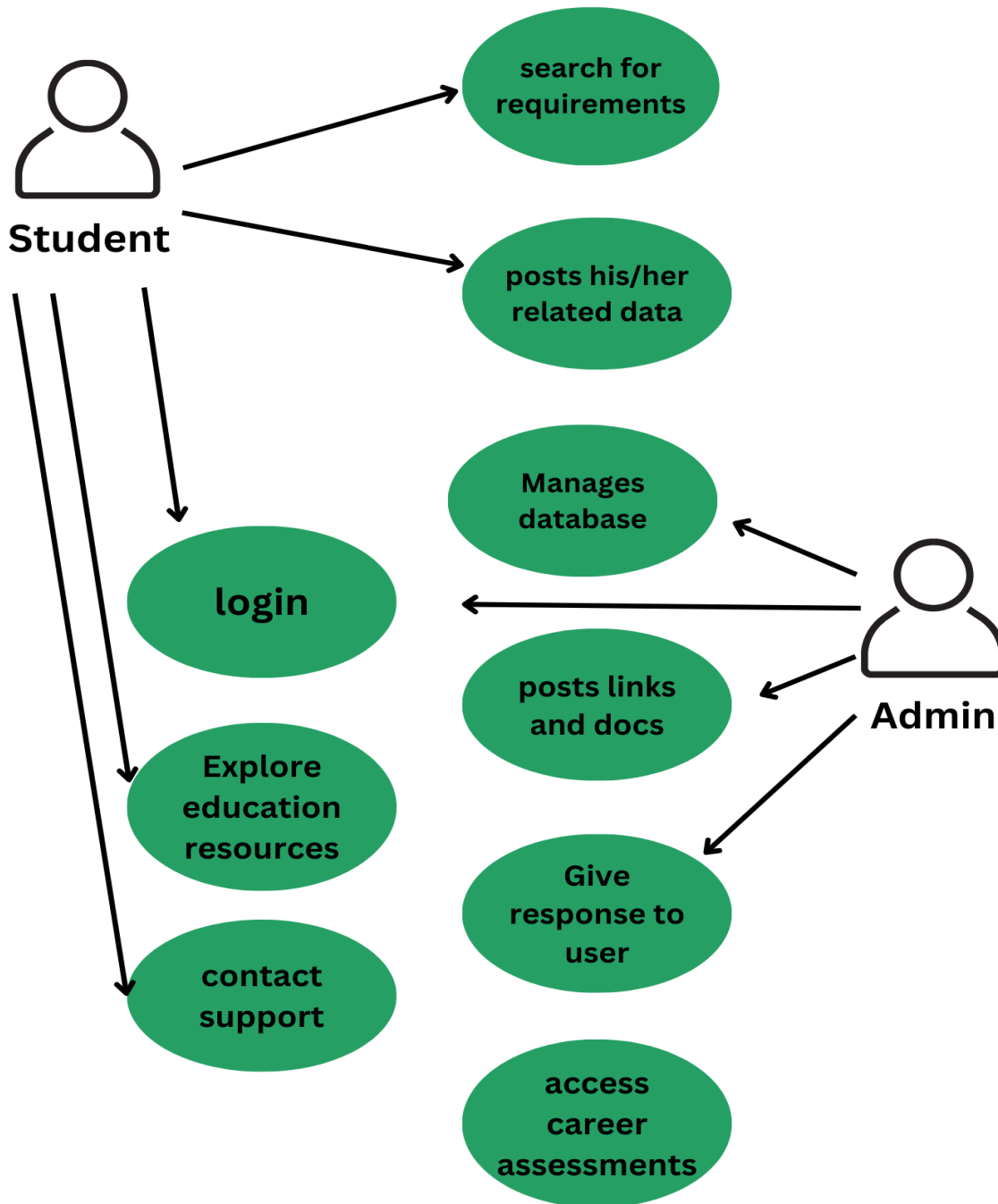
REST: Representational State Transfer (REST) defines a set of constraints for creating Web Services. It promotes Interoperability and aims at Fast Performance, Reliability, Reusability and Easy Update to Service. Roy Fielding defined REST in 2000. REST ensures Portability, Scalability, Simplicity and Modifiability.

It works in Client–Server Environment and provides Uniform Interface. WebServiceAPI's that adhere to the REST Architectural Constraints are called RESTful API's. A RESTful API has a Base URI to address and access it uniquely, standard HTTP Methods like GET, POST, etc. and a Media Type that defines State Transition Data Elements like JSON.

The current Representation tells the Client how to compose Requests for Transition to all the next available Application States. It is similar to a URI or a JAVA Applet.

Result: Study of Architectural Patterns has been done successfully.

6. Design UML Diagrams of a project using tools (career guidance website)



Use Case

Use Case Specification

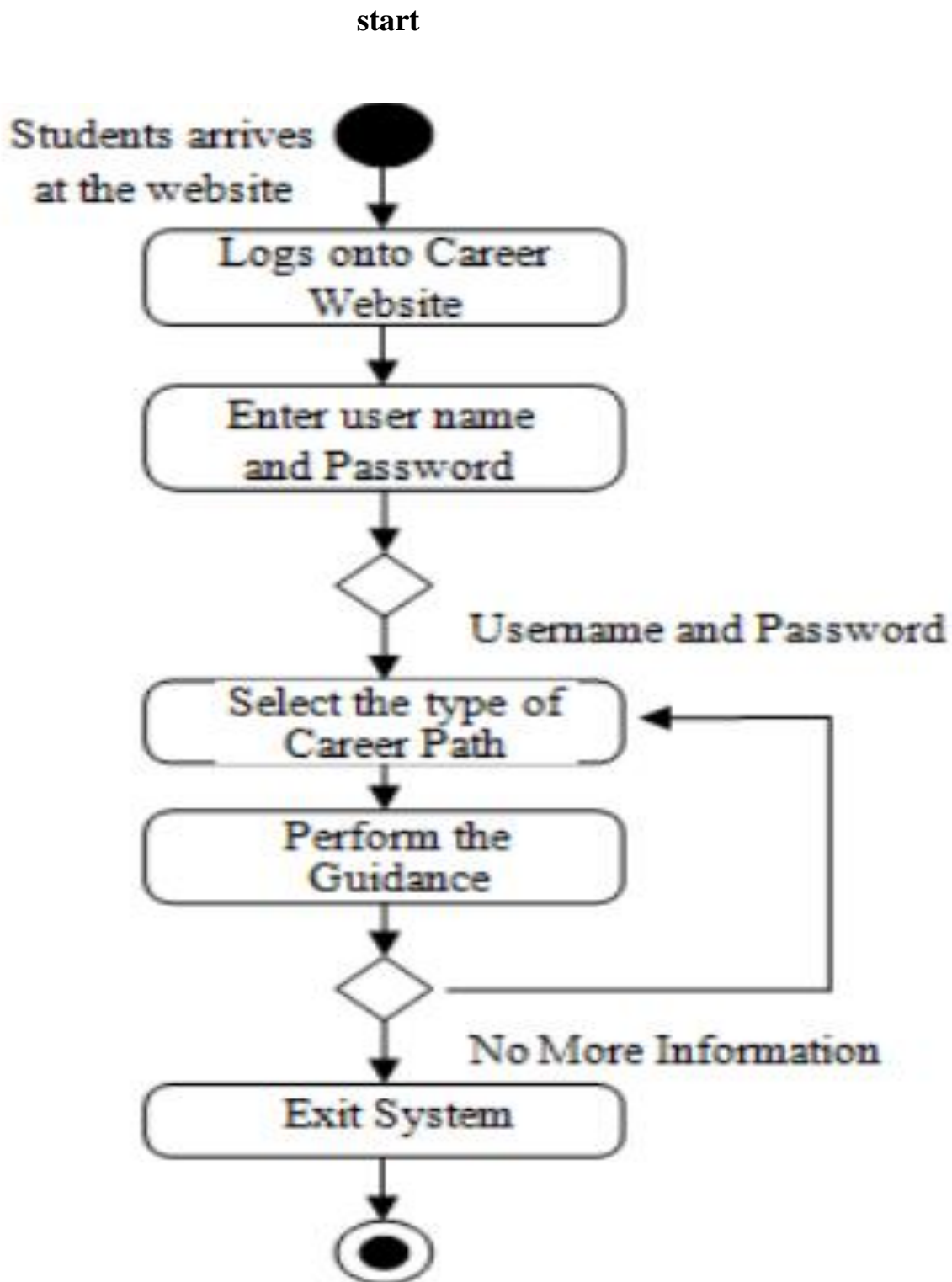
Table 3.2.1 Use case specification for Login.

Name Of Use Case	Login
Actor	User
Pre-condition	None.
Primary Flow of Events	<ul style="list-style-type: none">• Enter username• Enter password
Alternate Flow Of Events	Invalid Username or Password.
Post Condition	Login Successfully.
Use Case termination	By Back button.

Table 3.2.2 Use case specification for Search Books.

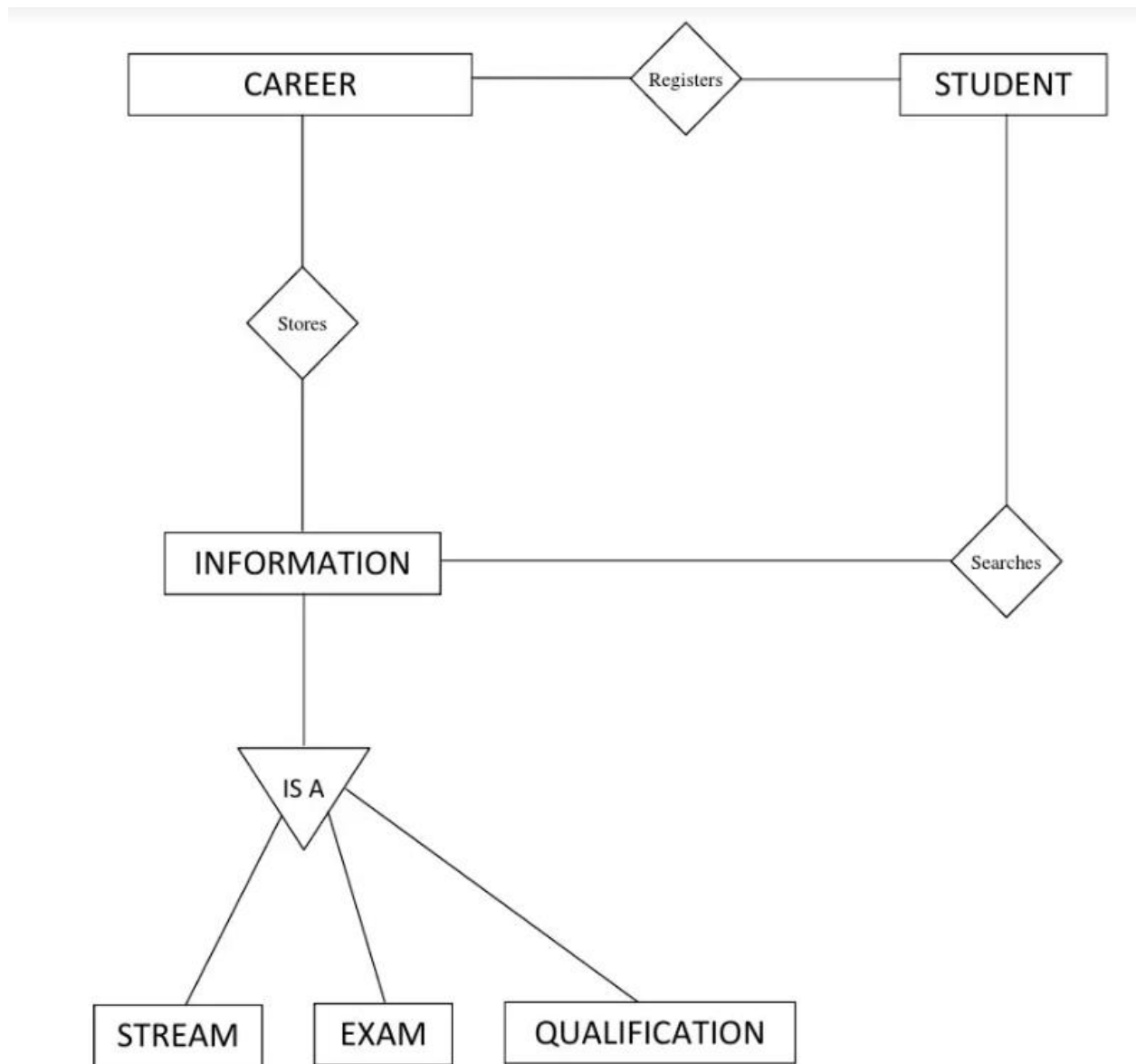
Name Of Use Case	Search for requirements
Actor	User
Pre-condition	Login.
Primary Flow of Events	<ul style="list-style-type: none">• ExploreEducation Resources• Access career assessment• Posts, links and docs
Alternate Flow Of Events	Something went wrong
Post Condition	Posts related details.
Use Case termination	By Back button.

7.Design Activity Diagrams of a project using tools (career guidance website)
Activity Diagram



8.Design ER Diagrams of a project using tools (career guidance system)

ER Diagram

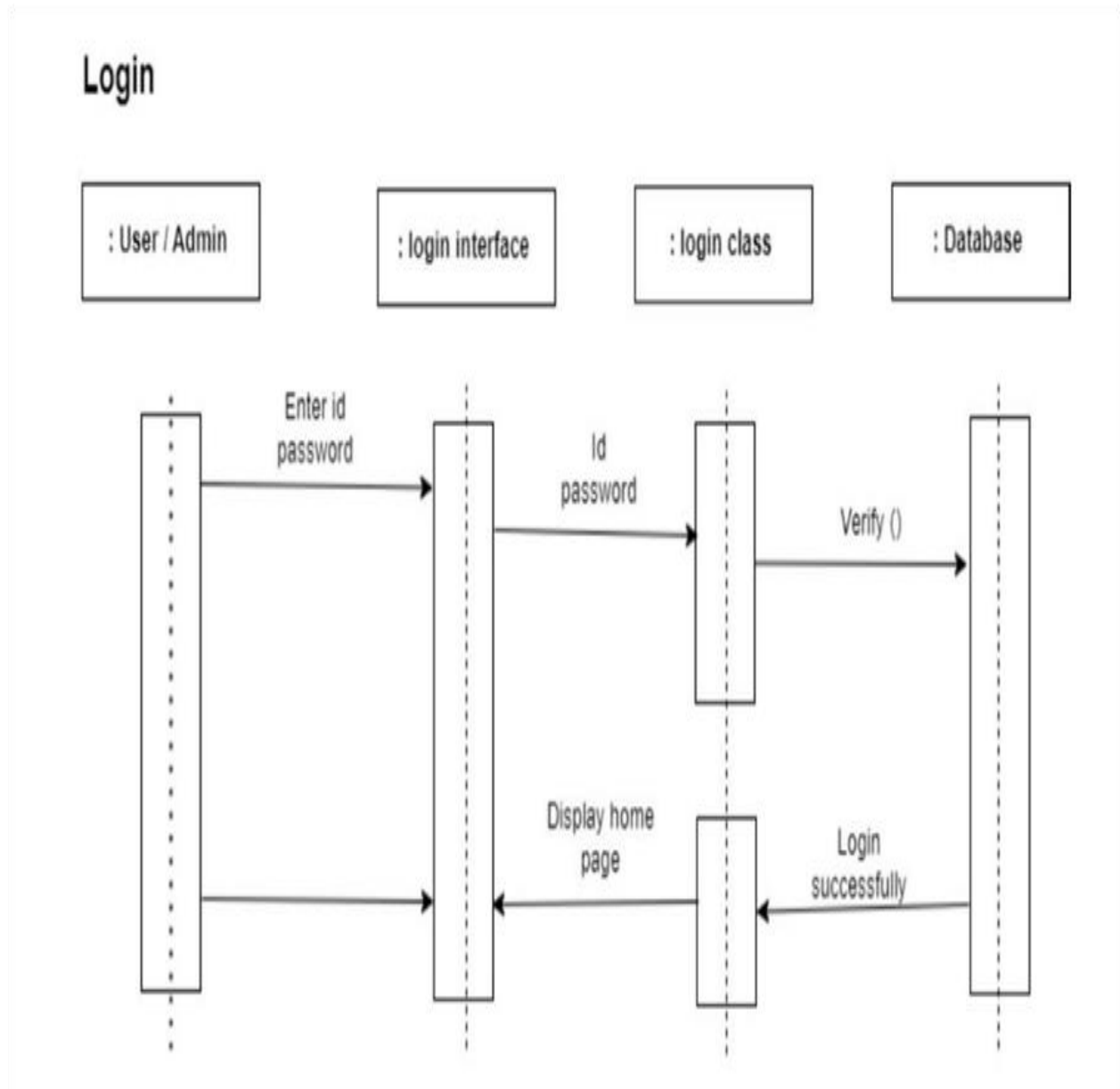


ER Diagram

9.Design Sequence Diagrams of a project using tools (career guidance system)

Sequence Diagram

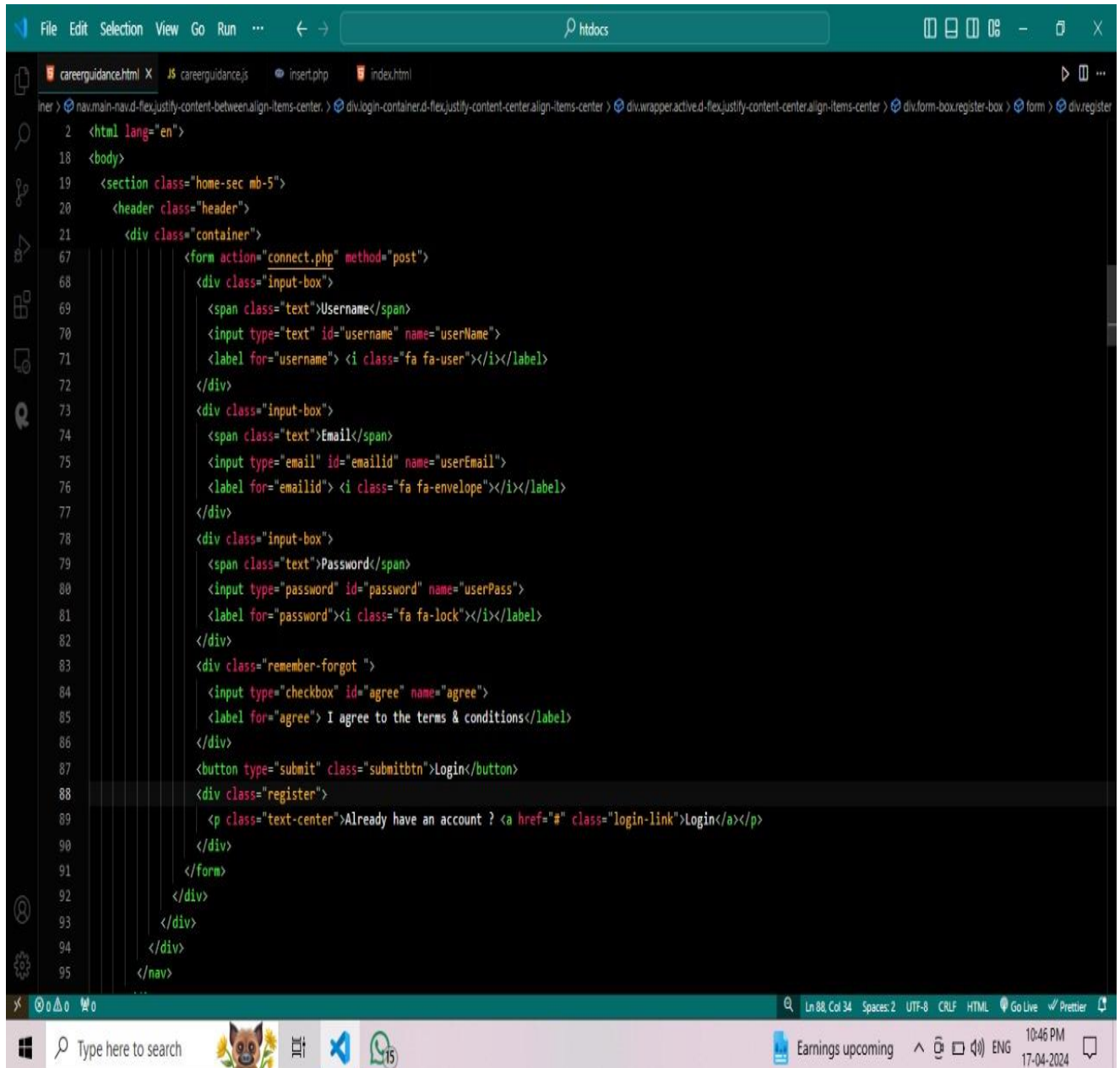
Login



Sequence diagram

10. Implement Testing of a project and draw flow diagram of it (career guidance website) (Basis Path testing)

Procedure login..



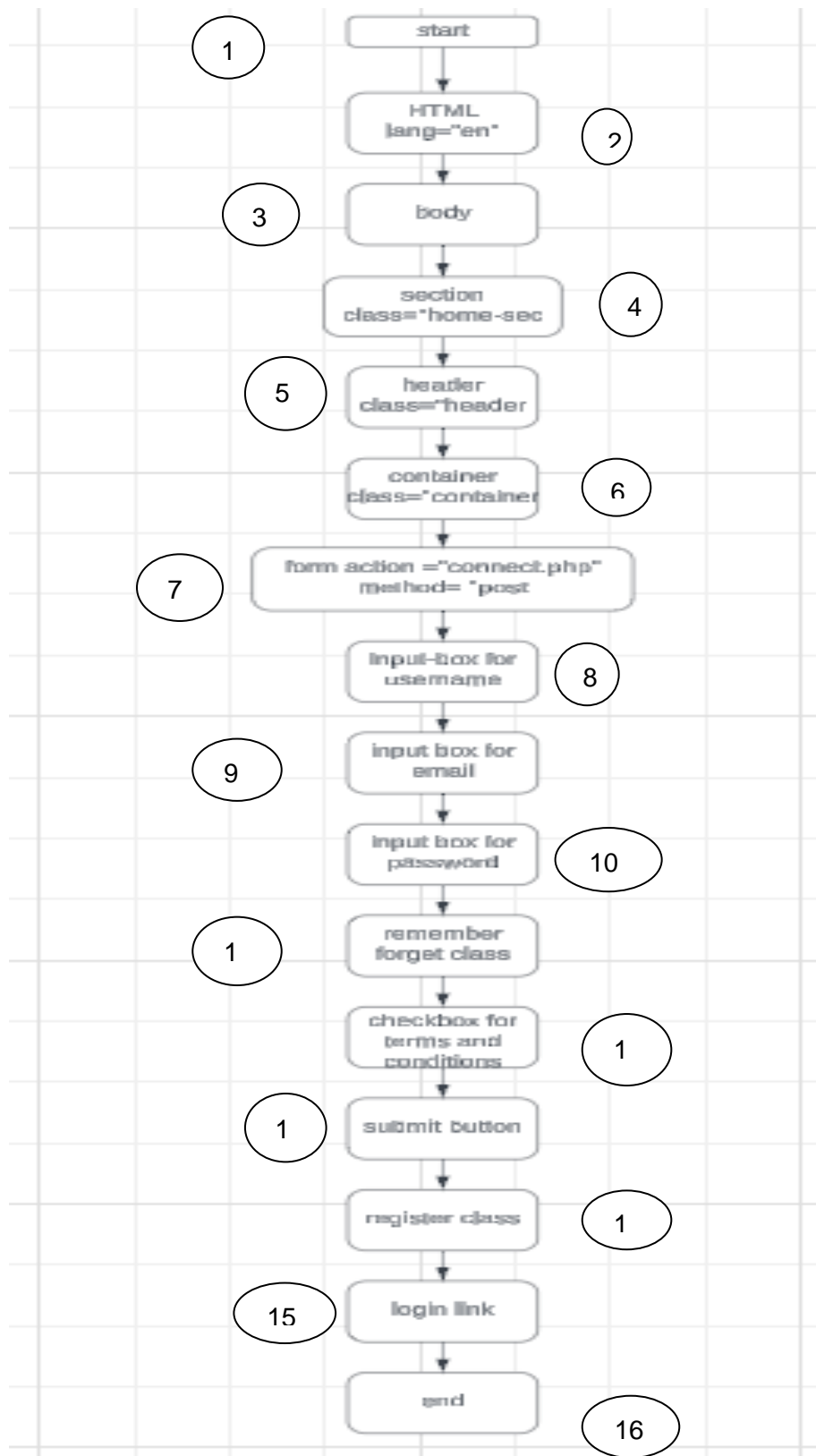
```
File Edit Selection View Go Run ... htdocs
careerguidance.html x JS careerguidance.js insert.php index.html
inner > nav.main-nav.d-flex.justify-content-between.align-items-center > div.login-container.d-flex.justify-content-center.align-items-center > div.wrapper.active.d-flex.justify-content-center.align-items-center > div.form-box.register-box > form > div.register

2 <html lang="en">
18 <body>
19 <section class="home-sec mb-5">
20 <header class="header">
21 <div class="container">
67 <form action="connect.php" method="post">
68 <div class="input-box">
69 <span class="text">Username</span>
70 <input type="text" id="username" name="userName">
71 <label for="username"> <i class="fa fa-user"></i></label>
72 </div>
73 <div class="input-box">
74 <span class="text">Email</span>
75 <input type="email" id="emailid" name="userEmail">
76 <label for="emailid"> <i class="fa fa-envelope"></i></label>
77 </div>
78 <div class="input-box">
79 <span class="text">Password</span>
80 <input type="password" id="password" name="userPass">
81 <label for="password"> <i class="fa fa-lock"></i></label>
82 </div>
83 <div class="remember-forgot">
84 <input type="checkbox" id="agree" name="agree">
85 <label for="agree"> I agree to the terms & conditions</label>
86 </div>
87 <button type="submit" class="submitbtn">Login</button>
88 <div class="register">
89 <p class="text-center">Already have an account ? <a href="#" class="login-link">Login</a></p>
90 </div>
91 </form>
92 </div>
93 </div>
94 </div>
95 </nav>
```

Ln 88, Col 34 Spaces: 2 UTF-8 CRLF HTML Go Live Prettier

Type here to search Earnings upcoming 10:46 PM 17-04-2024

Flow Graph



Cyclomatic Complexity...

$V(G) = \text{Number of Regions} = 1.$

$$V(G) = E - N + 2$$

Where $V(G)$ = cyclomatic complexity,

$E = \text{Number of Edges} = 15,$

$N = \text{Number of Nodes} = 16.$

Hence,

$$V(G) = 1.$$

Also

$$V(G) = P + 1$$

Where $P = \text{Number of Predicate nodes} = 1.$

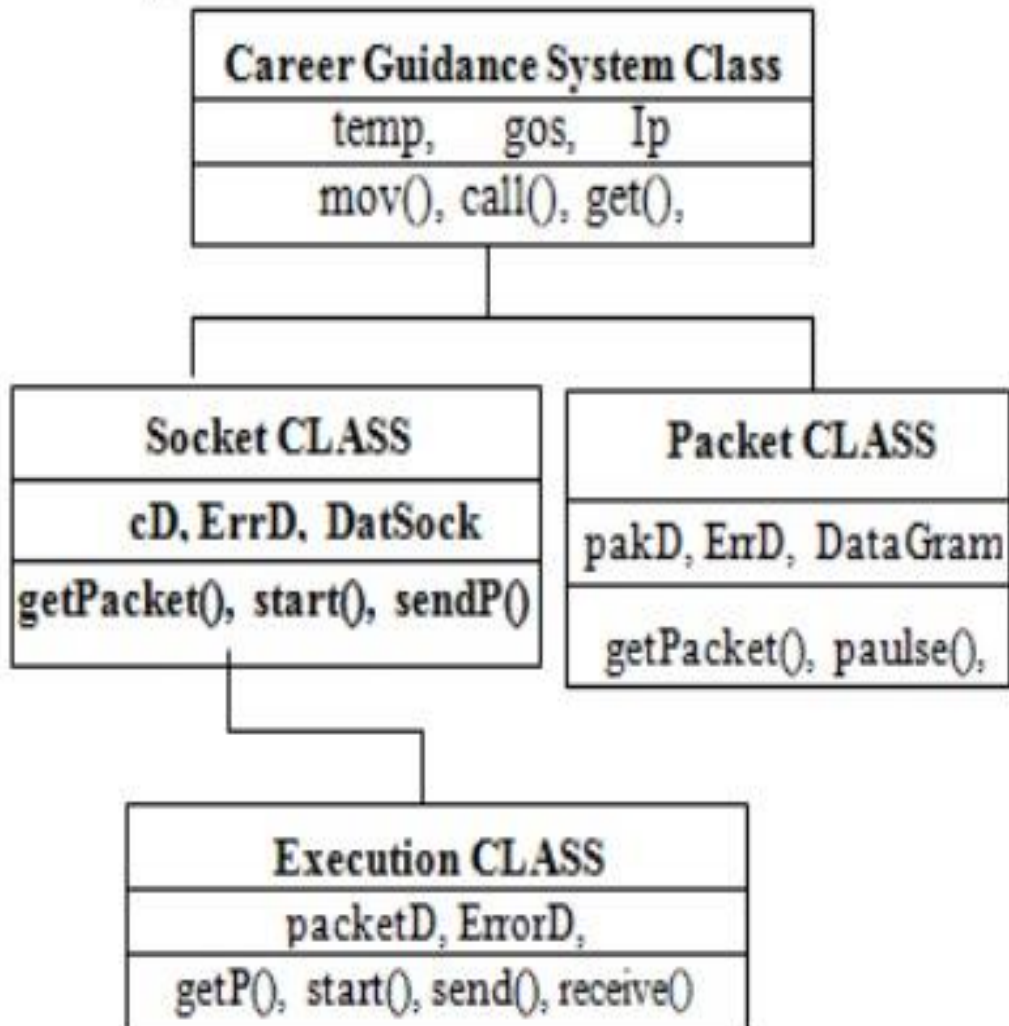
$$V(G) = 1.$$

Path

1. 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16

11.Design Class Diagrams of a project using tools (Library Management System)

CLASS DIAGRAM



Class Diagram