Republic of the Philippines
CEBU TECHNOLOGICAL UNIVERSITY
ARGAO CAMPUS
Ed Kintanar Street, Lamacan, Argao Cebu Philippines
Website: http://www.argao.ctu.edu.ph E-mail: cdargao@ctu.edu.ph
Phone No.: (032) 485-8290/485-5109 loc 1700Fax. N0.: (032)4858-290

Systems Integration and Architecture 1

Weeks 8 - 9

Information Technology Department

STUDENT ACTIVITY SHEET

The Development Life Cycle

Intended Learning Outcomes

At the end of the orientation, you should be able to:

- discuss about Software Development Life Cycle, ERP implementation life cycle and project management implementation strategies
- develop a use-case diagram
- create a data dictionary and;
- design an entity-relationship diagram

Materials
Computer, Student Activity Sheet, Google Suite apps, Mentimeter, Padlet

References
Hossein BIDGOLI. MIS. Course Technology CENGAGE. 2011

SDG Integration
SDG # 4 – ***Quality Education*** (Ensure inclusive and equitable quality education and promote life-long learning opportunities for all
SDG # 5 – *Gender Equality* (Achieve gender equality and empower all women)
SDG # 9 – ***Innovation and Infrastructure*** (Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation)

Name: _____ Course & Yr.: _____ Section: _____

Lesson Preparation

Business Idea: _____

RAMP	Assessment
R: Is it profitable? How much money do you think is needed to start this venture?	
A: What is your advantage? Who are your competitors?	
M: is there a big need for the product / service? What is your pricing strategy?	
P: How risky is the opportunity? If it succeeds, will there be a high reward?	

Concept Notes Presentation (40 minutes)

What is SDLC (Software Development Life Cycle)?

- is a structured process used in the software industry to design, develop, test, and maintain high-quality software.
- aims to produce software that meets or exceeds customer expectations, reaches completion within time and cost estimates, and is of high quality.
- A framework that describes the activities performed at each stage of a software development project.

Why is SDLC Important?

- It helps alleviate the complexity of developing a system from scratch.
- It helps transform an idea project into a functional and fully operational system.

SDLC Phases

1. Planning
2. Requirements Analysis
3. Design
4. Implementation/ Development
5. Testing & Integration
6. Deployment
7. Maintenance
8. Documentation

Planning

- This is the initial phase of the SDLC process that sets out to discover, identify, and define the scope of the project to decide the course of action and specifically address the issues that are going to be solved by the new system solution.

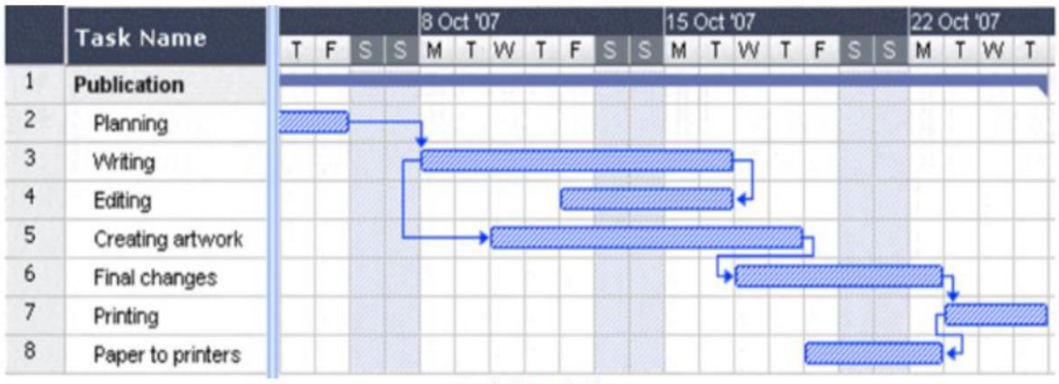
- Gather the team to brainstorm, set goals, and identify risks.

- Devise a set of business goals, requirements, specifications, and any high-level risks that might hinder the project's success

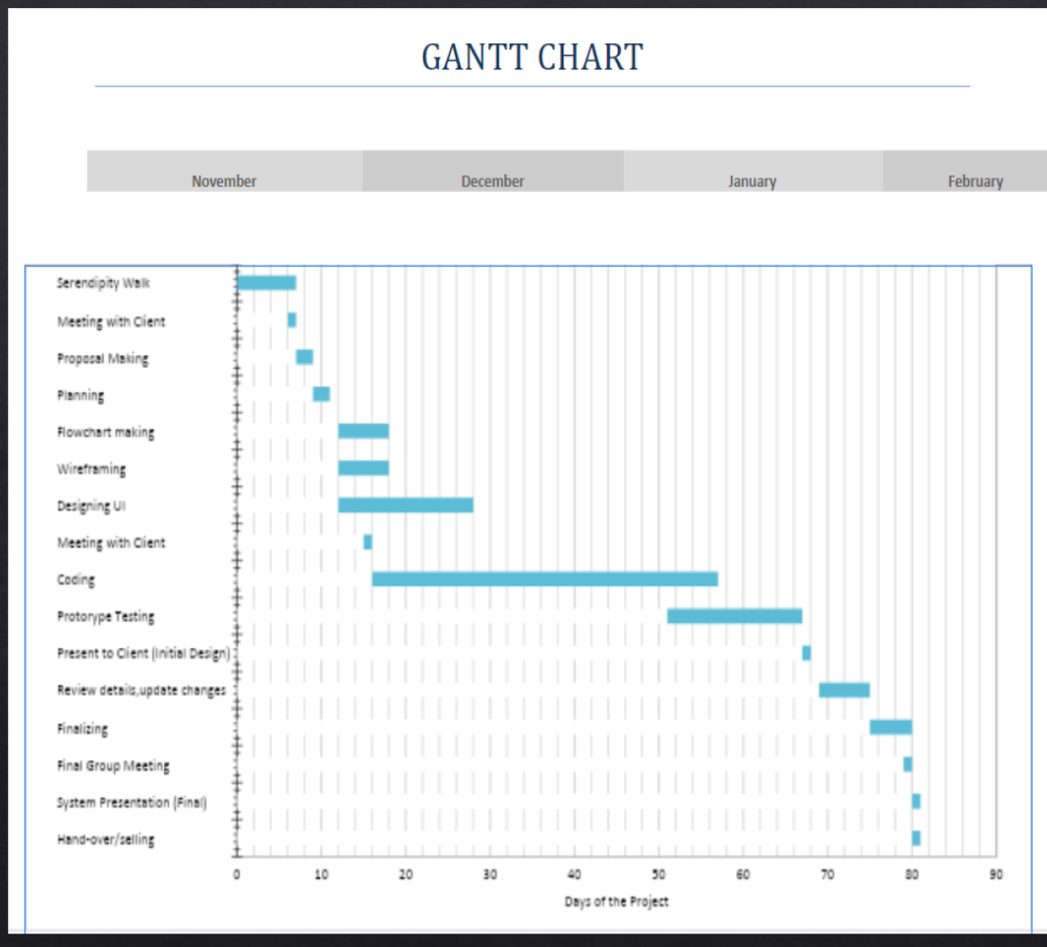
Every activity is related to one or more other activities. Every activity, except the first and last, has a relationship with a predecessor and a successor. Sequencing activities means placing the activities in the right order using the right relationships. There are four types of relationships:

1. **Finish to Start –** Cannot start the successor activity until its predecessor Is finished.
2. **Start to Start –** Cannot start the successor activity until its predecessor has started.
3. **Start to Finish –** Cannot finish the successor activity until its predecessor had started.
4. **Finish to Finish –** Cannot finish the successor activity until its predecessor has finished.

Relationships 1 and 2 are the most commonly used. Finish to Start is a sequential relationship and Start to Start is typically a parallel or over-lapping relationship.



Gantt chart dependencies

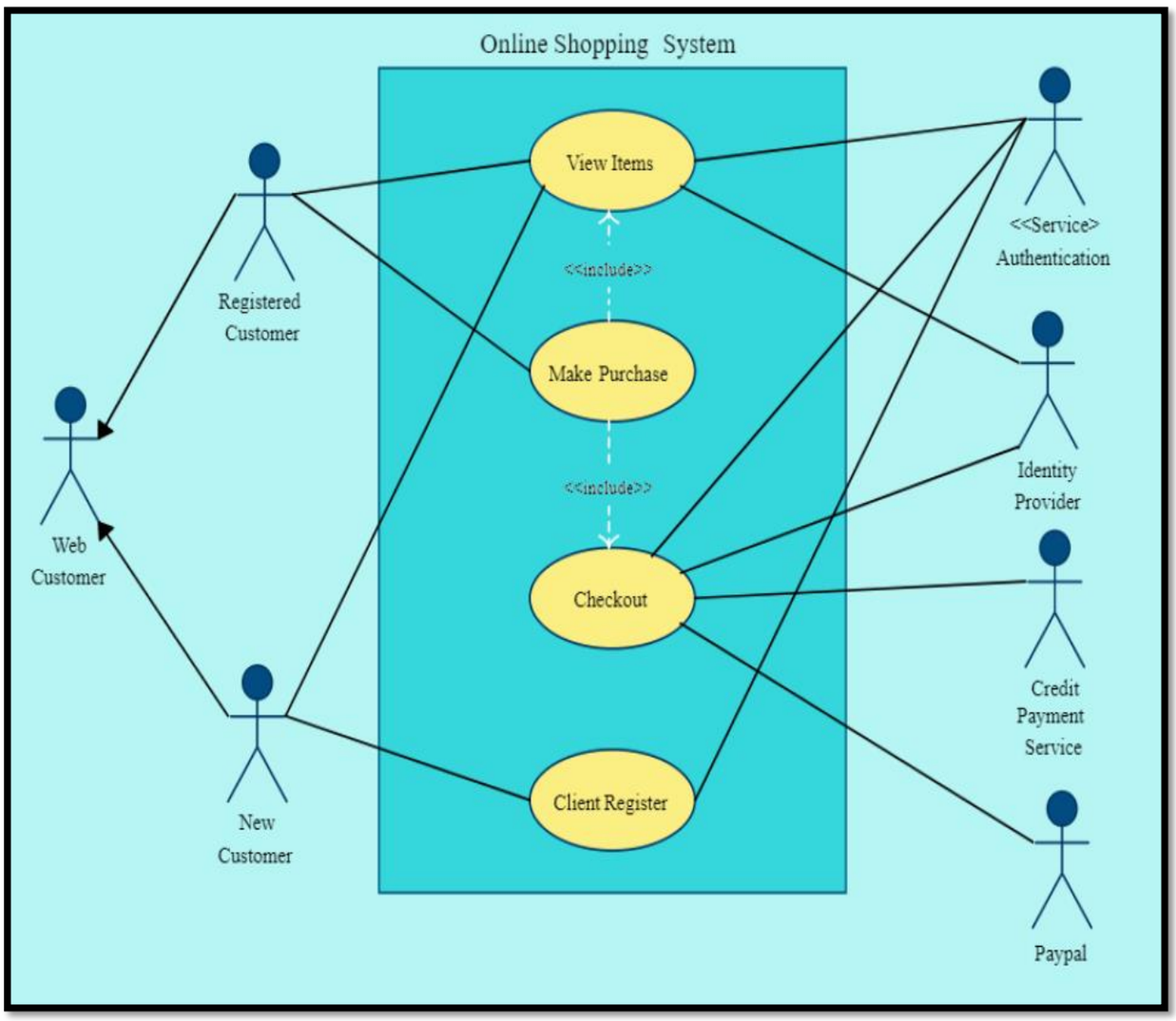


Project Mangement Tool

"A man who does not plan long ahead will find trouble at his door." – Confucius

Analysis

- The functional requirements of the system are considered and how the solution will meet the end user's expectations.



Steps to Create a Use Case Diagram

Identify Actors: Identify the primary actors involved. These are entities (could be users or external systems) interacting with the system.

Identify Use Cases: Determine the actions or services the system offers to the actors.

Create Use Case Relationships: Define relationships between actors and use cases. These connections illustrate how actors interact with the system.

Organize Use Cases: Group use cases and actors to represent different functionalities and roles within the system.

Refine and Review: Review the diagram to ensure it accurately represents the system's functionality and interactions.

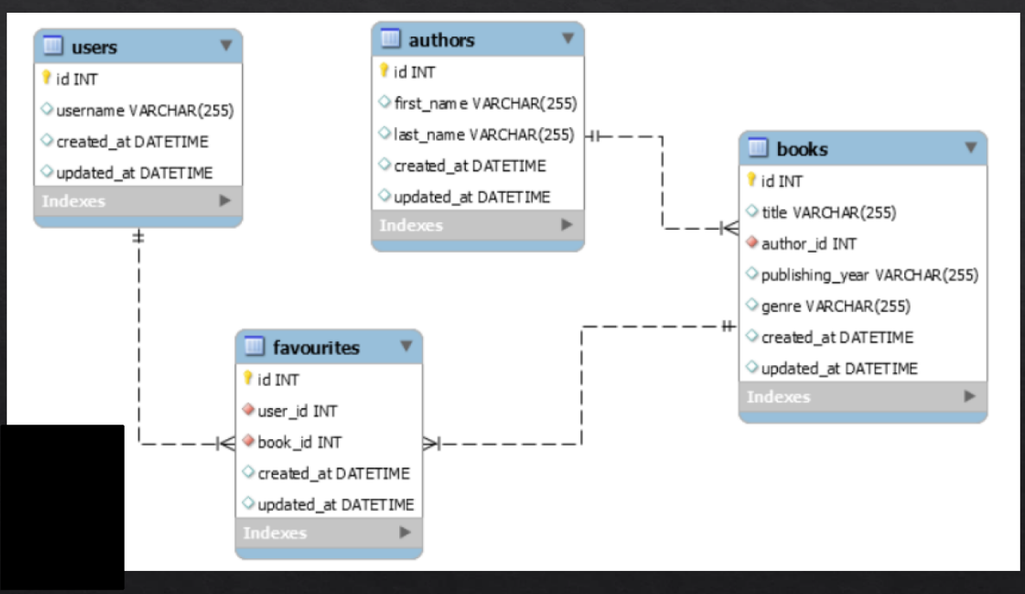
Design

1. Architecture
2. User Interface
3. Platforms
4. Programming
5. Communications
6. Security
7. Prototyping

Design

- Create a blueprint for the software.
- Determine how the software will be structured, how it will function, and how it will be implemented.

Entity Relationship Diagram (ERD)



To design a data dictionary and ERD, here are some steps that can be followed:

Identify the entities

- Identify the entities that will be included in the database. Entities are objects or concepts that have data associated with them.

Define the attributes

- Define the attributes for each entity. Attributes are characteristics or properties of an entity.

Determine the relationships

- Determine the relationships between the entities. Relationships describe how the entities are related to each other.

Create the ERD

- Create the ERD using any available tool . The ERD should include all the entities, attributes, and relationships.

Create the data dictionary

- Create the data dictionary using a tool such as Word or Microsoft Excel. The data dictionary should include a description of each entity, attribute, and relationship.

Keep the ERD and data dictionary synchronized

- Keep the ERD and data dictionary synchronized by updating both whenever changes are made to the database.

Data Dictionary

Field Name	Data Type	Data Format	Field Size	Description	Example
License ID	Integer	NNNNNN	6	Unique number ID for all drivers	12345
Surname	Text		20	Surname for Driver	Jones
First Name	Text		20	First Name for Driver	Arnold
Address	Text		50	First Name for Driver	11 Rocky st Como 2233
Phone No.	Text		10	License holders contact number	0400111222
D.O.B	Date / Time	DD/MM/YYYY	10	Drivers Date of Birth	08/05/1956

Data Dictionary

A data dictionary in an ERD (Entity-Relationship Diagram) is a textual description of data objects and their inter-relationships. It is a detailed definition and documentation of data models.

- commonly used in confirming data requirements and for database developers to create and maintain a database system.

- describes the physical attributes of a data element, such as the data type (string, boolean, integer, etc.), maximum length, format and description.,

Steps to create Data Dictionary

- ✓ Identify the entities — Company, Department
- ✓ Define the attributes — Company: Company ID, Company Name; Department: Department ID, Department Name
- ✓ Determine the relationships — Company has many Departments (1:M)
- ✓ Create the data dictionary

Data Dictionary Example

Entity: Employee

Attribute: Employee ID

- Data Type: Integer
- Maximum Length: 10
- Format: Numeric
- Description: Unique identifier for each employee

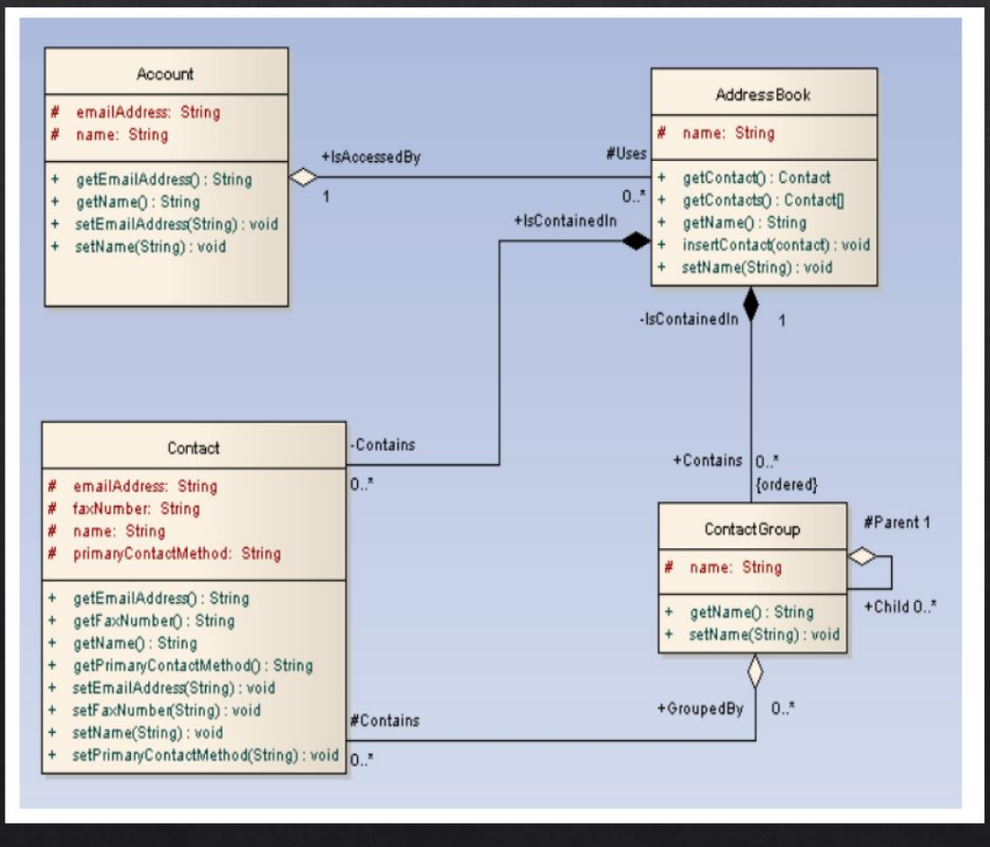
Attribute: First Name

- Data Type: String
- Maximum Length: 50
- Format: Text
- Description: First name of the employee

Attribute: Last Name

- Data Type: String
- Maximum Length: 50
- Format: Text
- Description: Last name of the employee

Class Diagram



Good design
is good business.

-- Thomas Watson Jr.

"Design is not just what it looks like & feels like. Design is how it works."

– Steve Jobs

Development

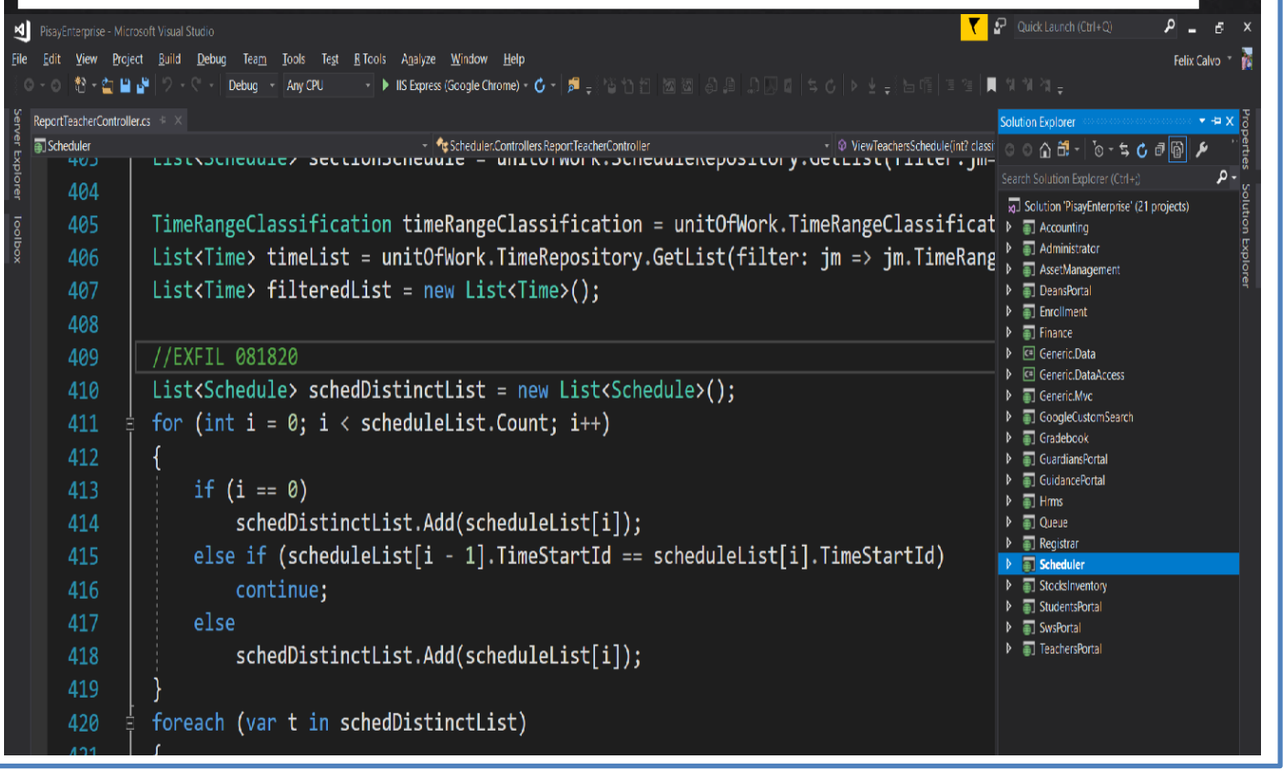
- This involves writing code and constructing and fine-tuning the technical and physical configurations necessary to build the overall information system.

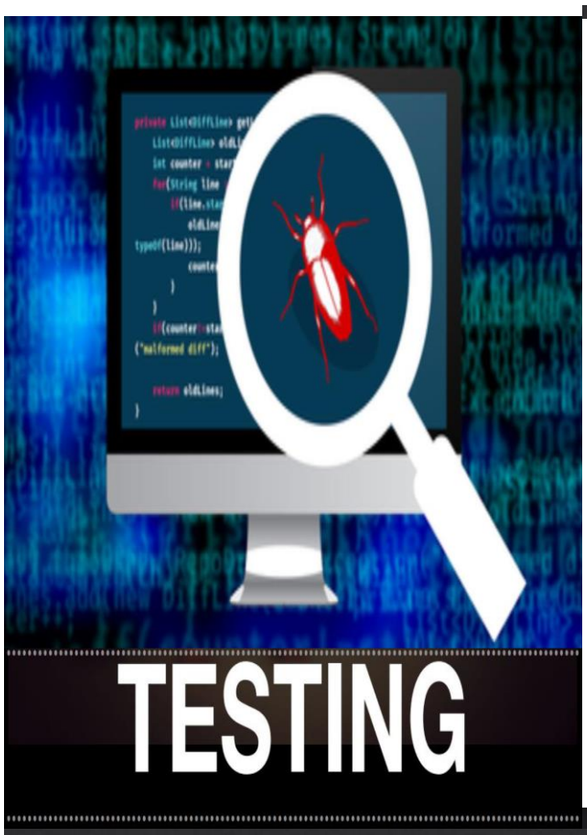
Development

- This involves writing code and constructing and fine-tuning the technical and physical configurations necessary to build the overall information system.

The top 10 languages in RedMonk's latest rankings are:

1. JavaScript
2. Python
3. Java
4. PHP
5. CSS
6. C++
7. C#
8. TypeScript
9. Ruby
10. C





Testing

- Quality check takes place.
- Validate whether the application addresses all user requirements and technical performance.

Testing Levels

1. Unit Testing
2. Integration Testing
3. System Testing
4. Acceptance Testing

Deployment / Implementation

- Once the testing is completed and there are no open high priority issues, then comes the time to deploy the system.

Implementation Methods

1. Parallel
2. Phased
3. Pilot
4. Direct

Deployment

- Release the software to the end-users.

- Install the software and ensure that it is working correctly.

Maintenance

- Update the software to fix bugs, ensure reliability, and add new features or functionality.

Maintenance

- This stage is when the "fine tuning" of the software takes place. Once the build is deployed to production environment, any issues that the real users face are considered as Post-Production issues.

Documentation

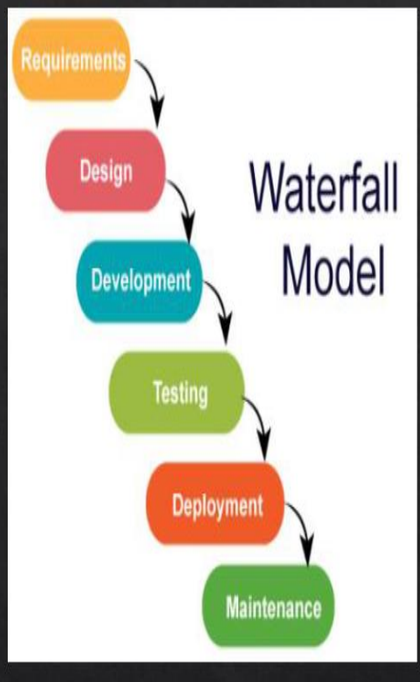
- All written documents and materials dealing with software product development.

- Explain product functionality, unify project-related information, and allow for discussing all significant questions arising between stakeholders and developers.

Types of Software Development Life Cycle (SDLC) Models

Waterfall Model/Methodology

- one of the oldest and most widely used SDLC models.
- follows a linear, sequential approach where each phase of the development process is completed before moving on to the next one.

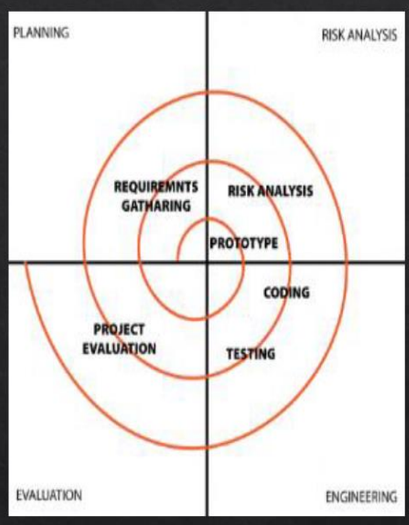


Types of Software Development Life Cycle (SDLC) Models

Spiral Model

- is a highly customizable SDLC model that is used by many leading software companies.

- involves repeating four phases (planning, risk analysis, engineering, and evaluation) until the project is finished.

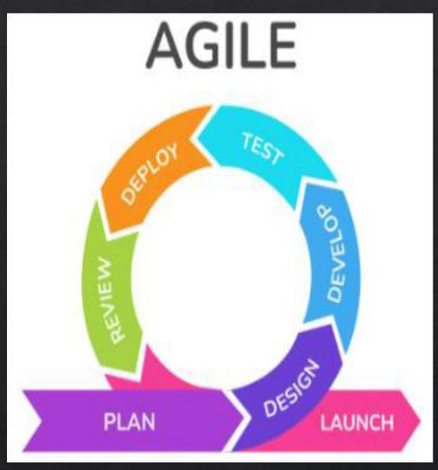


Types of Software Development Life Cycle (SDLC) Models

Agile Methodology

- is a flexible and iterative SDLC model that emphasizes collaboration, customer satisfaction, and rapid delivery of working software.

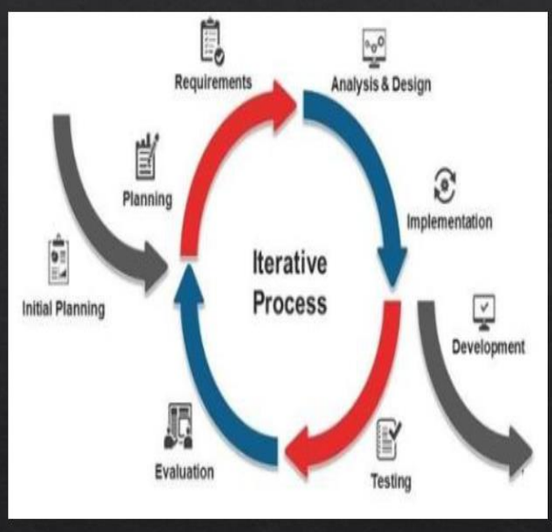
- involves breaking down the development process into small, manageable chunks called sprints.



Types of Software Development Life Cycle (SDLC) Models

Iterative Model

- is a flexible SDLC model that involves developing software in small increments.
- Each increment is tested and evaluated before moving on to the next one.

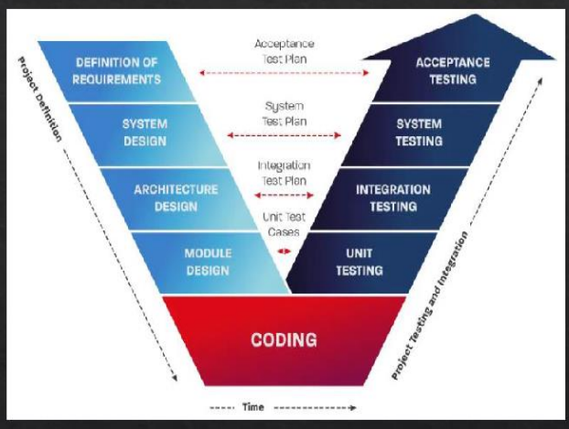


Types of Software Development Life Cycle (SDLC) Models

V-Model

- is a verification and validation SDLC model that involves testing and development phases that are planned in parallel.

- used to ensure that the software meets the requirements and specifications gathered in the planning stage.

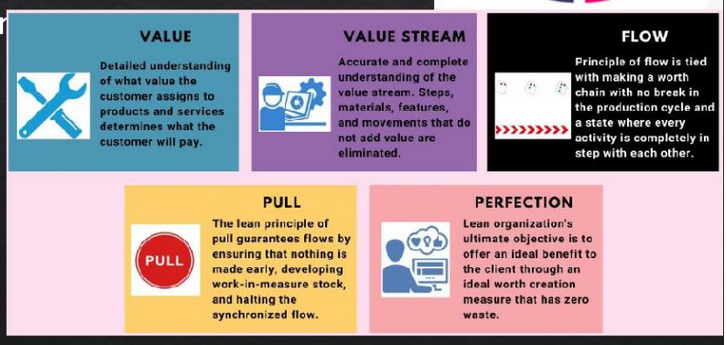


Types of Software Development Life Cycle (SDLC) Models

Lean Methodology

- is a customer-focused SDLC model that emphasizes delivering value to the customer as quickly as possible.

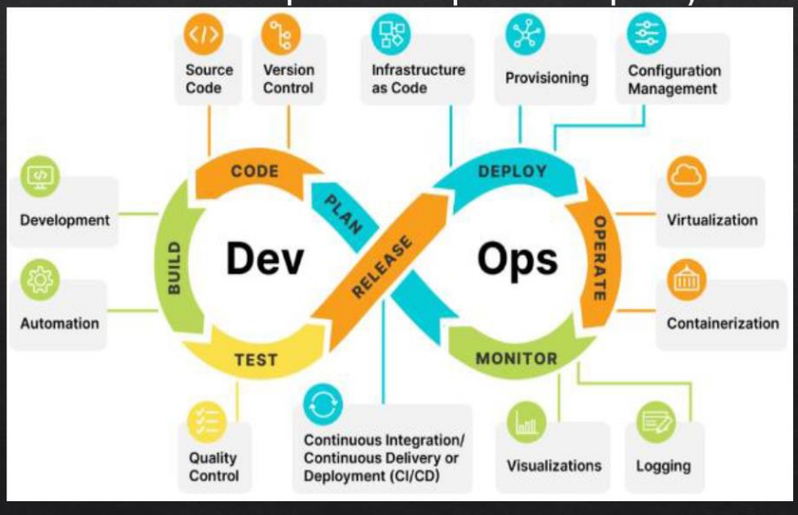
- involves eliminating waste and streamlining the developmer



Types of Software Development Life Cycle (SDLC) Models

DevOps

- is a collaborative SDLC model that involves integrating development and operations teams to improve the speed and quality of software delivery.



ERP IMPLEMENTATION LIFE CYCLE

ERP (Enterprise Resource Planning)

- implementation life cycle is a process that organizations follow to plan, design, develop, test, deploy, and support an ERP system.

- integrates various functions across the business, such as financial management, human resources, sales, and manufacturing, to deliver benefits such as increased productivity and efficiency.

Key Phases of the ERP implementation life cycle:

Discovery and Planning

- involves gathering the team to brainstorm, set goals, and identify risks. The team works together to devise a set of business goals, requirements, specifications, and any high-level risks that might hinder the project's success.

Design

- involves creating a blueprint for the ERP system. The team creates a detailed plan for the ERP system's architecture, user interface, and functionality.

Development

- is where the actual coding of the ERP system takes place. The team follows the blueprint created in the design phase to write the code for the ERP system.

Key Phases of the ERP implementation life cycle:

Testing

- involves testing the ERP system to ensure that it meets the requirements and specifications set in the planning phase. The team performs various tests to identify and fix any bugs or issues in the ERP system.

Deployment

- involves releasing the ERP system to the end-users. The team ensures that the ERP system is ready for production use and deploys it to the end-users.

Support

- involves providing ongoing support to the end-users of the ERP system. The team ensures that the ERP system is running smoothly and provides any necessary updates or maintenance.

Management Implementation Strategies

- involve setting clear goals, defining key variables, determining roles and responsibilities, conducting proper research, mapping out any risks, improving internal communication, creating a healthy budget, and developing procedures or policies.

Management Implementation Strategies:

Set Clear Goals and Define Key Variables

- Clearly define the goals and objectives of the project.
- Identify the key variables that will impact the success of the project.
- Communicate the goals and variables to all team members.

Determine Roles, Responsibilities, and Relationships

- Define the roles and responsibilities of each team member.
- Establish clear lines of communication and reporting.
- Ensure that everyone understands their role and how it fits into the overall project.

Delegate the Work

- Assign tasks to team members based on their skills and expertise.
- Ensure that the workload is distributed evenly.
- Monitor progress to ensure that tasks are completed on time and to the required standard.

Conduct Proper Research

- Conduct research to ensure that the project is feasible and that the resources are available to implement it.
- Analyze the market, competitors, and customer needs to identify opportunities and potential risks.
- Use the research to inform the project plan and strategy.

Management Implementation Strategies:

Map Out Any Risks

- Identify potential risks and develop contingency plans to mitigate them.
- Consider the impact of risks on the project timeline, budget, and resources.
- Monitor risks throughout the project and adjust plans as necessary.

Improve Internal Communication

- Establish clear lines of communication and reporting.
- Ensure that everyone has the support and knowledge they need to implement the strategy successfully.
- Create procedures or policies that help teams better achieve their goals and allocate the resources needed to implement the strategy.

Create a Healthy Budget

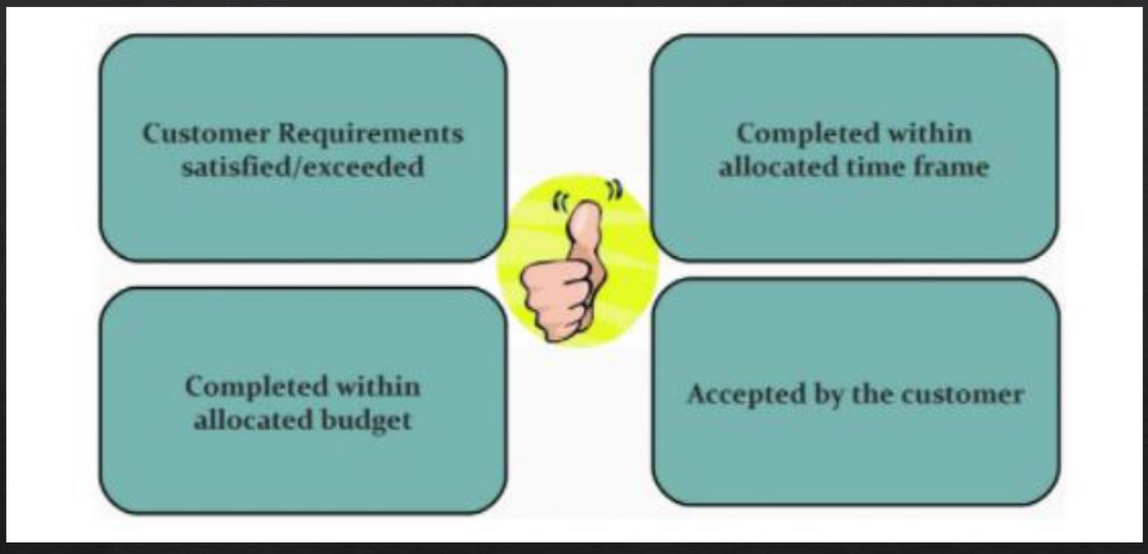
- Allocate the necessary funds to each stage of the implementation process.
- Ensure that the budget is realistic and achievable.
- Monitor spending throughout the project and adjust plans as necessary.

Management Implementation Strategies:

Develop Procedures or Policies

- Create a set of procedures or policies that help teams better achieve their goals.
- Ensure that everyone understands the procedures or policies and how they fit into the overall project.
- Allocate the resources needed to implement the procedures or policies.

Project Success



Guided Practice

Create a Use-Case Diagram for an online shopping system taking into consideration the suggestions below:

ACTORS

Customer: Interacts with the system to browse and purchase items.
Admin: Manages products, reviews orders, and handles user accounts.

USE CASES

Browse Items: Allows the customer to view products.
Add to Cart: Enables the customer to add items to their shopping cart.
Place Order: Allows the customer to finalize a purchase.
Manage Products: Permits the admin to add, edit, or remove products.
Review Orders: Allows the admin to view and manage customer orders.

RELATIONSHIPS

Customer interacts with Browse Items, Add to Cart, and Place Order use cases.
Admin interacts with Manage Products and Review Orders use cases.

Draw the diagram in the box provided below:

CompuSkill (Performance)

Create a data dictionary and draw an ER diagram for this scenario.

A company has multiple departments, and each department has multiple employees. Each employee has a unique employee ID, name, and job title. Each department has a unique department ID, name, and location.

INSERT THE TABULAR FORMAT OF THE DATA DICTIONARY IN THIS AREA.

DRAW THE ERD IN THIS BOX.