

**SOFTWARE ENGINEERING**  
**SOFTWARE TESTING LIFE CYCLE(SDLC)\**  
**WATERFALL MODEL**

It was introduced in 1970 by Winston Royce.

Waterfall Model is a sequential model that divides software development into pre-defined phases.

Each phase must be completed before the next phase can begin with no overlap between the phases.

Each phase is designed for performing specific activity during the SDLC phase.

Advantages:

These are some advantages of the Waterfall Model.

It is simple and easy to understand and use.

It is easy to manage.

It works well for smaller and low budget projects where requirements are very well understood.

Clearly defined stages and well understood.

It is easy to arrange tasks.

Process and results are well documented.

Disadvantages:

These are some disadvantages of Waterfall Model.

It is difficult to measure progress within stages.

Poor model for long and ongoing projects.

No working software is produced until late during the life cycle.

High amounts of risk and uncertainty.

Not a good model for long and object oriented projects.

Cannot accommodate changing requirements.

Different phases and Activities performed in each stage

### Requirement Gathering stage

During this phase, detailed requirements of the software system to be developed are gathered from client

### Design Stage

Plan the programming language, for Example Java, PHP, .net or database like Oracle, MySQL, etc. Or other high-level technical details of the project

### Built Stage

After design stage, it is built stage, that is nothing but coding the software

### Test Stage

In this phase, you test the software to verify that it is built as per the specifications given by the client.

### Deployment stage

Deploy the application in the respective environment

### Maintenance stage

Once your system is ready to use, you may later require change the code as per customer request.

## ITERATIVE MODEL

The iterative process model is a software development life cycle (SDLC) approach in which the initial development work is conducted based on initial requirements that are clearly defined, and subsequent features are added to this base software product through iterations until the final system is completed.

Planning --> Analysis--> Implementation --> Evaluation

The various phases of Iterative model are as follows:

#### 1. Requirement gathering & analysis:

In this phase, requirements are gathered from customers and checked by an analyst whether requirements will be fulfilled or not. Analyst checks that need will be achieved within budget or not. After all of this, the software team skips to the next phase.

#### 2. Design:

In the design phase, team design the software by the different diagrams like Data Flow diagram, activity diagram, class diagram, state transition diagram, etc.

### 3. Implementation:

In the implementation, requirements are written in the coding language and transformed into computer programmes which are called Software.

### 4. Testing:

After completing the coding phase, software testing starts using different test methods. There are many test methods, but the most common are white box, black box, and gray box test methods.

### 5. Deployment:

After completing all the phases, software is deployed to its work environment.

### 6. Review:

In this phase, after the product deployment, review phase is performed to check the behaviour and validity of the developed product. And if there are any error found then the process starts again from the requirement gathering.

### 7. Maintenance:

In the maintenance phase, after deployment of the software in the working environment there may be some bugs, some errors or new updates are required. Maintenance involves debugging and new addition options.

When to use the Iterative Model?

When requirements are defined clearly and easy to understand.

When the software application is large.

When there is a requirement of changes in future.

Advantage(Pros) of Iterative Model:

Testing and debugging during smaller iteration is easy.

A Parallel development can plan.

It is easily acceptable to ever-changing needs of the project.

Risks are identified and resolved during iteration.

Limited time spent on documentation and extra time on designing.

Disadvantage(Cons) of Iterative Model:

It is not suitable for smaller projects.

More Resources may be required.

Design can be changed again and again because of imperfect requirements.

Requirement changes can cause over budget.

Project completion date not confirmed because of changing requirements.

SOFTWARE ENGINEERING

SOFTWARE TESTING LIFE CYCLE(SDLC)

SPIRAL MODEL

The spiral model is a systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with elements of the Waterfall model. The spiral model is used by software engineers and is favored for large, expensive and complicated projects.

When to use the Spiral Model?

When a project is vast in software engineering, a spiral model is utilised.

A spiral approach is utilised when frequent releases are necessary.

When it is appropriate to create a prototype

When evaluating risks and costs is crucial

The spiral approach is beneficial for projects with moderate to high risk.

The SDLC's spiral model is helpful when requirements are complicated and ambiguous.

If modifications are possible at any moment

When committing to a long-term project is impractical owing to shifting economic priorities.

Each cycle in the spiral is divided into four parts:

Objective setting: Each cycle in the spiral starts with the identification of purpose for that cycle, the various alternatives that are possible for achieving the targets, and the constraints that exists.

Risk Assessment and reduction: The next phase in the cycle is to calculate these various alternatives based on the goals and constraints. The focus of evaluation in this stage is located on the risk perception for the project.

Development and validation: The next phase is to develop strategies that resolve uncertainties and risks. This process may include activities such as benchmarking, simulation, and prototyping.

Planning: Finally, the next step is planned. The project is reviewed, and a choice made whether to continue with a further period of the spiral. If it is determined to keep, plans are drawn up for the next step of the project.

The development phase depends on the remaining risks. For example, if performance or user-interface risks are treated more essential than the program development risks, the next phase may be an evolutionary development that includes developing a more detailed prototype for solving the risks.

The risk-driven feature of the spiral model allows it to accommodate any mixture of a specification-oriented, prototype-oriented, simulation-oriented, or another type of approach. An essential element of the model is that each period of the spiral is completed by a review that includes all the products developed during that cycle, including plans for the next cycle. The spiral model works for development as well as enhancement projects.

These are following advantages and disadvantages of using Spiral Model

Advantages of Spiral Model:

Software is produced early in the software life cycle.

Risk handling is one of important advantages of the Spiral model, it is best development model to follow due to the risk analysis and risk handling at every phase.

Flexibility in requirements. In this model, we can easily change requirements at later phases and can be incorporated accurately. Also, additional Functionality can be added at a later date.

It is good for large and complex projects.

It is good for customer satisfaction. We can involve customers in the development of products at early phase of the software development. Also, software is produced early in the software life cycle.

Strong approval and documentation control.

It is suitable for high risk projects, where business needs may be unstable. A highly customized product can be developed using this.

Disadvantages of Spiral Model:

It is not suitable for small projects as it is expensive.

It is much more complex than other SDLC models. Process is complex.

Too much dependable on Risk Analysis and requires highly specific expertise.

Difficulty in time management. As the number of phases is unknown at the start of the project, so time estimation is very difficult.

Spiral may go on indefinitely.

End of the project may not be known early.

It is not suitable for low risk projects.

May be hard to define objective, verifiable milestones. Large numbers of intermediate stages require excessive documentation.

The spiral model is a systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with elements of the Waterfall model. The spiral model is used by software engineers and is favored for large, expensive and complicated projects.

When to use the Spiral Model?

When a project is vast in software engineering, a spiral model is utilised.

A spiral approach is utilised when frequent releases are necessary.

When it is appropriate to create a prototype

When evaluating risks and costs is crucial

The spiral approach is beneficial for projects with moderate to high risk.

The SDLC's spiral model is helpful when requirements are complicated and ambiguous.

If modifications are possible at any moment

When committing to a long-term project is impractical owing to shifting economic priorities.

Each cycle in the spiral is divided into four parts:

Objective setting: Each cycle in the spiral starts with the identification of purpose for that cycle, the various alternatives that are possible for achieving the targets, and the constraints that exists.

Risk Assessment and reduction: The next phase in the cycle is to calculate these various alternatives based on the goals and constraints. The focus of evaluation in this stage is located on the risk perception for the project.

Development and validation: The next phase is to develop strategies that resolve uncertainties and risks. This process may include activities such as benchmarking, simulation, and prototyping.

Planning: Finally, the next step is planned. The project is reviewed, and a choice made whether to continue with a further period of the spiral. If it is determined to keep, plans are drawn up for the next step of the project.

The development phase depends on the remaining risks. For example, if performance or user-interface risks are treated more essential than the program development risks, the next phase may be an evolutionary development that includes developing a more detailed prototype for solving the risks.

The risk-driven feature of the spiral model allows it to accommodate any mixture of a specification-oriented, prototype-oriented, simulation-oriented, or another type of approach. An essential element of the model is that each period of the spiral is completed by a review that includes all the products developed during that cycle, including plans for the next cycle. The spiral model works for development as well as enhancement projects.

These are following advantages and disadvantages of using Spiral Model

Advantages of Spiral Model:

Software is produced early in the software life cycle.

Risk handling is one of important advantages of the Spiral model, it is best development model to follow due to the risk analysis and risk handling at every phase.

Flexibility in requirements. In this model, we can easily change requirements at later phases and can be incorporated accurately. Also, additional Functionality can be added at a later date.

It is good for large and complex projects.

It is good for customer satisfaction. We can involve customers in the development of products at early phase of the software development. Also, software is produced early in the software life cycle.

Strong approval and documentation control.

It is suitable for high risk projects, where business needs may be unstable. A highly customized product can be developed using this.

Disadvantages of Spiral Model:

It is not suitable for small projects as it is expensive.

It is much more complex than other SDLC models. Process is complex.

Too much dependable on Risk Analysis and requires highly specific expertise.

Difficulty in time management. As the number of phases is unknown at the start of the project, so time estimation is very difficult.

Spiral may go on indefinitely.

End of the project may not be known early.

It is not suitable for low risk projects.

May be hard to define objective, verifiable milestones. Large numbers of intermediate stages require excessive documentation.

### ***V- MODEL***

V-Model also referred to as the **Verification and Validation Model**. In this, each phase of SDLC must complete before the next phase starts. It follows a sequential design process same as the waterfall model. Testing of the device is planned in parallel with a corresponding stage of development.

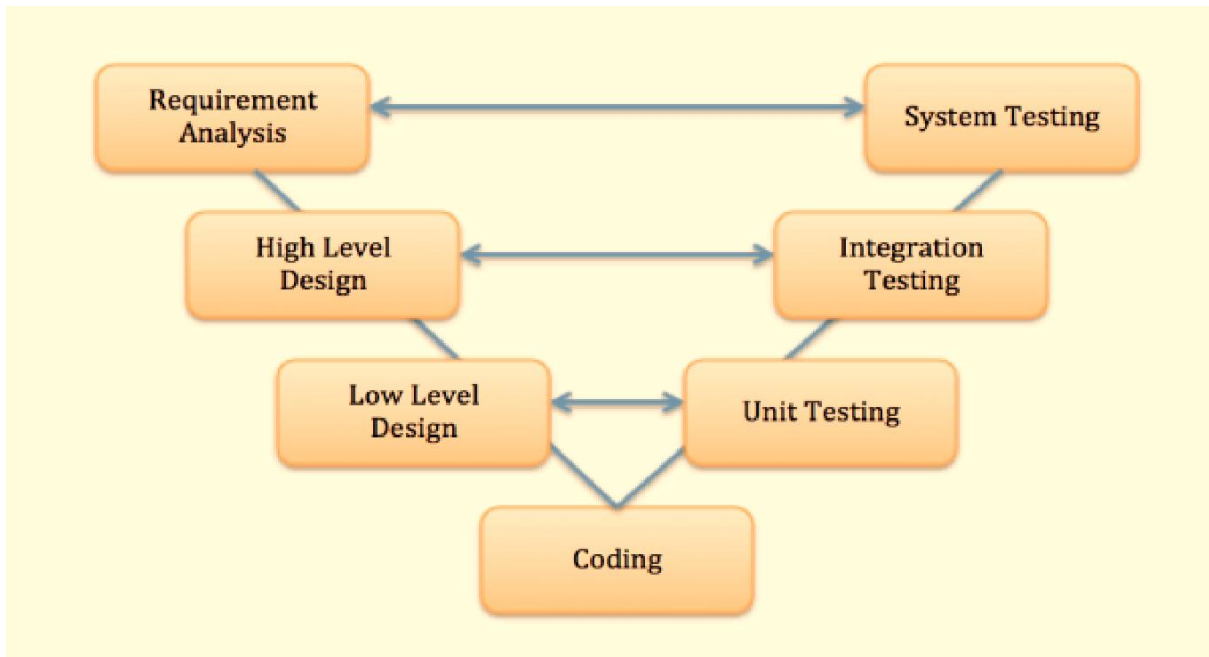
This is a highly-disciplined model and the next phase starts only after completion of the previous phase.

**Verification:** It involves a static analysis method (review) done without executing code. It is the process of evaluation of the product development process to find whether specified requirements meet.

**Validation:** It involves dynamic analysis method (functional, non-functional), testing is done by executing code. Validation is the process to classify the software after the completion of the development process to determine whether the software meets the customer expectations and requirements.

So V-Model contains Verification phases on one side of the Validation phases on the other side. Verification and Validation process is joined by coding phase in V-shape. Thus it is known as V-Model.





The left side of the model is Software Development Life Cycle – SDLC

The right side of the model is Software Test Life Cycle – STLC

The entire figure looks like a V, hence the name V – model

Apart from the V model, there are iterative development models, where development is carried in phases, with each phase adding a functionality to the software. Each phase comprises its independent set of development and testing activities.

### **There are the various phases of Verification Phase of V-model:**

**Business requirement analysis:** This is the first step where product requirements understood from the customer's side. This phase contains detailed communication to understand customer's expectations and exact requirements.

**System Design:** In this stage system engineers analyze and interpret the business of the proposed system by studying the user requirements document.

**Architecture Design:** The baseline in selecting the architecture is that it should understand all which typically consists of the list of modules, brief functionality of each module, their interface relationships, dependencies, database tables, architecture diagrams, technology detail, etc. The integration testing model is carried out in a particular phase.

**Module Design:** In the module design phase, the system breaks down into small modules. The detailed design of the modules is specified, which is known as Low-Level Design

**Coding Phase:** After designing, the coding phase is started. Based on the requirements, a suitable programming language is decided. There are some guidelines and standards for coding. Before checking in the repository, the final build is optimized for better performance, and the code goes through many code reviews to check the performance.

### There are the various phases of Validation Phase of V-model:

**Unit Testing:** In the V-Model, Unit Test Plans (UTPs) are developed during the module design phase. These UTPs are executed to eliminate errors at code level or unit level. A unit is the smallest entity which can independently exist, e.g., a program module. Unit testing verifies that the smallest entity can function correctly when isolated from the rest of the codes/ units.

**Integration Testing:** Integration Test Plans are developed during the Architectural Design Phase. These tests verify that groups created and tested independently can coexist and communicate among themselves.

**System Testing:** System Tests Plans are developed during System Design Phase. Unlike Unit and Integration Test Plans, System Tests Plans are composed by the client's business team. System Test ensures that expectations from an application developer are met.

**Acceptance Testing:** Acceptance testing is related to the business requirement analysis part. It includes testing the software product in user atmosphere. Acceptance tests reveal the compatibility problems with the different systems, which is available within the user atmosphere. It conjointly discovers the non-functional problems like load and performance defects within the real user atmosphere.

Alpha Testing	Beta Testing
It is done by internal testers of the organization.	It is done by real users.
It is an internal test, performed within the organization.	It is an external test, carried out in the user's environment.
Alpha Testing uses both black box and white box testing techniques	Beta Testing only uses the black box testing technique.
Identifies possible errors.	Checks the quality of the product.
Developers start fixing bugs as soon as they are identified.	Errors are found by users and feedback is necessary.
Long execution cycles.	It only takes a few weeks.
It can be easily implemented as it is done before the near end of development.	It will be implemented in the future version of the product.
It is performed before Beta Testing.	It is the final test before launching the product on the market.
It answers the question: Does the product work?	It answers the question: Do customers like the product?
Functionality and usability are tested.	Usability, functionality, security and reliability are tested with the same depth.

### When to use V-Model?

When the requirement is well defined and not ambiguous.

The V-shaped model should be used for small to medium-sized projects where requirements are clearly defined and fixed.

The V-shaped model should be chosen when sample technical resources are available with essential technical expertise.

### **Advantage of V-Model:**

Easy to Understand.

Testing Methods like planning, test designing happens well before coding.

This saves a lot of time. Hence a higher chance of success over the waterfall model.

Avoids the downward flow of the defects.

Works well for small plans where requirements are easily understood.

### **Disadvantage of V-Model:**

Very rigid and least flexible.

Not a good for a complex project.

Software is developed during the implementation stage, so no early prototypes of the software are produced.

If any changes happen in the midway, then the test documents along with the required documents, has to be updated.


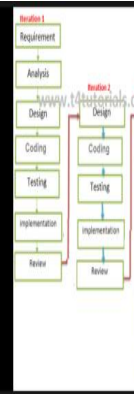
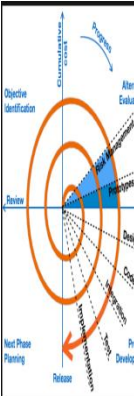
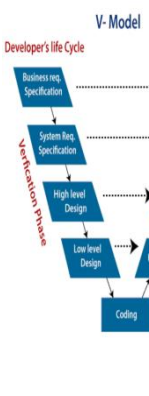

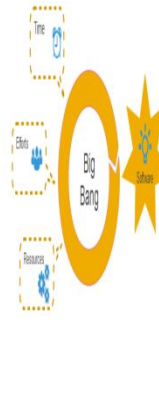
ALL MODELS

<b>Topics</b>	<b>Water Fall Model</b>	<b>Iteration Model</b>	<b>Spiral Model</b>	<b>V- Model</b>	<b>RAD - Model</b>	<b>Big Bang Model</b>
---------------	-------------------------	------------------------	---------------------	-----------------	--------------------	-----------------------

Definition	Waterfall Model	The iterative process model	The spiral model	V-Model	The Rapid Application Development (or RAD)	The Big Bang model
	<p>is a sequential model that divides software development into pre-defined phases.</p> <p>*Each phase is designed for performing specific activity during the SDLC phase.</p>	<p>is a software development life cycle (SDLC) approach in which the initial development work is conducted based on initial requirements that are clearly defined, and subsequent features are added to this base</p>	<p>is a systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with elements of the Waterfall model.</p>	<p>also referred to as the <b>Verification and Validation</b> Model. In this, each phase of SDLC must complete before the next phase starts. It follows a sequential design process same as the waterfall model. Testing of the device is planned in parallel with a corresponding stage of develop</p>	<p>model is based on prototyping and iterative model with no (or less) specific planning. In general, RAD approach to software development means putting lesser emphasis on planning tasks and more emphasis on development and coming up with a prototype.</p>	<p>is an SDLC model where we do not follow any specific process. The development just starts with the required money and efforts as the input, and the output is the software developed which may or may not be as per customer requirement.</p>

		software product through iterations until the final system is completed .		ment.		
<b>Phases</b>	1.Requirement Gathering 2.System Analysis 3.System Design 4.Coding 5.Implementation	1.Requirement 2.Design 3.Coding 4.Testing 5.Evaluation	1.Planning 2.Risk analysis 3.Product development 4.Planning or	1.Unit Testing 2.Integration 3.Testing System Testing 4.Accept	1.Business Modeling 2.Data Modeling 3.Process modeling 4.Application Generation 5.Testing	1.Time 2.Efforts 3.Resources

	entation 6. Testing 7. Deployment 8. Operation and Maintenance		Evaluation.	ance Testing	and Turnover.	
<b>When to use the Model?</b>	1. When the requirements are constant and not changed regularly. 2. A project is short 3. The situation is calm 4. Where the tools and technology used is consistent and is not changing 5. When resources are well prepared and are available to use.	1. When requirements are defined clearly and easy to understand. 2. When the software application is large. 3. When there is a requirement of changes in future.	1. Projects in which frequent releases are necessary. 2. Projects in which changes may be required at any time. 3. Long term projects that are not feasible due to altered economic priorities. 4. Medium to	1. Easy to Understand. 2. Testing Methods like planning, testing, designing happens well before coding. 3. This saves a lot of time. Hence a higher chance of success over the waterfall model. 4. Avoids the downwa	1. RAD should be used when there is a need to create a system that can be modularized in 2-3 months of time. 2. It should be used if there's high availability of designers for modeling and the budget is high enough to afford their cost along with the cost of automated code generating	This method is also used when the size of the developer team is small and when requirements are not defined, and the release date is not confirmed or given by the customer.

			<p>high risk projects .</p> <p>5. Projects in which cost and risk analysis is important.</p>	<p>rd flow of the defects.</p> <p>5. Works well for small plans where requirements are easily understood.</p>	<p>tools.</p> <p>3. RAD SD LC model should be chosen only if resources with high business knowledge are available and there is a need to produce the system in a short span of time (2-3 months).</p>	
<b>Diagram</b>						
<b>The Uses</b>	<p>The waterfall model use is to achieve goals based on the needs of their</p>	<p>The Iterative Model allows the accessing earlier</p>	<p>1. Used for risk management that combines the iterative</p>	<p>To set clear expectations and identify potential flaws throughout the</p>	<p>It allows you to incorporate updates based on usage rather than a rigid developme</p>	<p>It is Ideal for small projects with one or two developers working together and is</p>

	business.	phases, in which the variations made respectively.	development process model with elements of the Waterfall model. 2.The spiral model is used by software engineers and is favored for large, expensive and complicated projects.	process without needing a final build first.	not plan.	also useful for academic or practice projects.
<b>Advantages</b>	1.Before the next phase of development, each phase must be completed. 2.Suited	1.Testing and debugging during smaller iteration is easy. 2.A Parallel	1.High amount of risk analysis 2.Useful for large and mission-critical	1.Easy to Understand. 2.Testing Methods like planning, test design	1.Reduced development time. 2.Increases reusability of components 3.Quick initial reviews	1.There is no planning required. 2.Simple Model. 3.Few resources required. 4.Easy to manage.



	<p>for smaller projects where requirements are well defined.</p> <p>3.They should perform quality assurance test (Verification and Validation) before completing each stage.</p> <p>4.Elaborate documentation is done at every phase of the software's development cycle.</p> <p>5.Project is completely dependent on</p>	<p>development can plan.</p> <p>3.It is easily acceptable to ever-changing needs of the project.</p> <p>4.Risks are identified and resolved during iteration.</p> <p>5.Limited time spent on documentation and extra time on designing.</p>	<p>projects .</p>	<p>g happens well before coding.</p> <p>3.This saves a lot of time. Hence a higher chance of success over the waterfall model.</p> <p>4.Avoids the downward flow of the defects.</p> <p>5.Works well for small plans where requirements are easily understood.</p>	<p>occur</p> <p>4.Encourages customer feedback</p> <p>5.Integration from very beginning solves a lot of integration issues.</p>	<p>5.Flexible for developers.</p>
--	---	---	-------------------	--	---	-----------------------------------

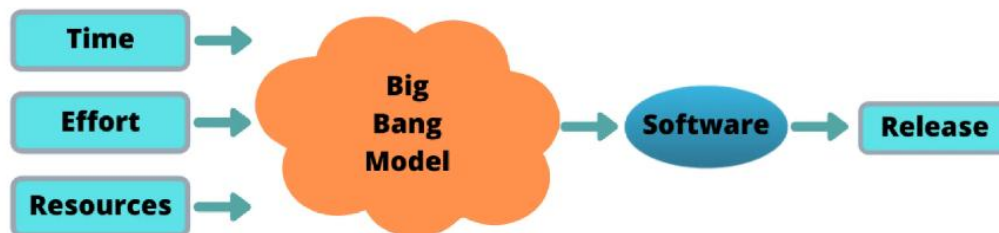
	project team with minimum client intervention. 6.Any changes in software is made during the process of the development.					
<b>Disadvantages</b>	Error can be fixed only during the phase. 2.It is not desirable for complex project where requirement changes frequently. 3.Testing period comes quite late	1.It is not suitable for smaller projects . 2.More Resources may be required. 3.Design can be changed again and again because	1.Can be a costly model to use. 2.Risk analysis needed highly particular expertise. 3.Does n't work well for smaller projects .	1.Very rigid and least flexible. 2.Not a good for a complex project. 3.Software is developed during the implementation stage, so no early prototyp	1.Depends on strong team and individual performances for identifying business requirements. 2.Only system that can be modularized can be built using RAD 3.Requires highly skilled developers/	1.There are high risk and uncertainty. 2.Not acceptable for a large project. 3.If requirements are not clear that can cause very expensive.

	<p>in the developmental process.</p> <p>4.Documentation occupies a lot of time of developers and testers.</p> <p>5.Clients valuable feedback cannot be included with ongoing development phase.</p> <p>6.Small changes or errors that arise in the completed software may cause a lot of problems.</p>	<p>of imperfect requirements.</p> <p>4.Requirement changes can cause over budget.</p> <p>5.Project completion date not confirmed because of changing requirements.</p>		<p>es of the software are produced.</p> <p>4.If any changes happen in the midway, then the test documents along with the required documents, has to be updated.</p>	<p>designers.</p> <p>4.High dependency on modeling skills</p> <p>5.Inapplicable to cheaper projects as cost of modeling and automated code generation is very high.</p>	
--	--	--	--	---	---	--

<b>Real Time Examples</b>	1.University Rankings  2.Student Scores	Culinary field, Professionals Typically perform repeated preparation, arrangement, and Presentation of their products.	Used for risk management that combines the iterative development process model with elements of the Waterfall model. The spiral model is used by software engineers and is favored for large, expensive and complicated projects .	Projects initiated from a request for proposals (RFPs), the customer has a very clear documented requirements. Military projects. Mission Critical projects, for example , in a Space shuttle.	Handling employee resignation.	“HACKATHON”
---------------------------	---	---	--	--	--------------------------------	-------------

## **BIG BANG MODEL**

- The Big bang model is an SDLC paradigm that begins from scratch. It is the most basic SDLC (Software Development Life Cycle) paradigm because it requires very minimal planning.
- However, it demands more finances, code, and time. The big bang model was named after the “Great Big Bang,” which formed galaxies, stars, planets, etc.
- Similarly, to produce a product, this SDLC model combines time, effort, and resources.
- The product is gradually produced as the customer’s requirements arrive, yet, the ultimate product may not meet the actual requirements.
- This model is ideal for small projects like academic projects or practical projects. One or two developers can work together on this model.
- Below is a diagrammatic illustration of the big bang model.



### **When to use Big Bang Model?**

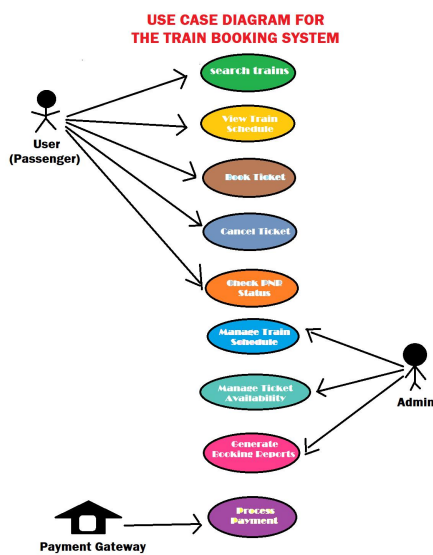
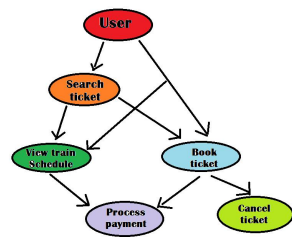
As we discussed above, this model is required when this project is small like an academic project or a practical project. This method is also used when the size of the developer team is small and when requirements are not defined, and the release date is not confirmed or given by the customer.

## Advantage(Pros) of Big Bang Model:

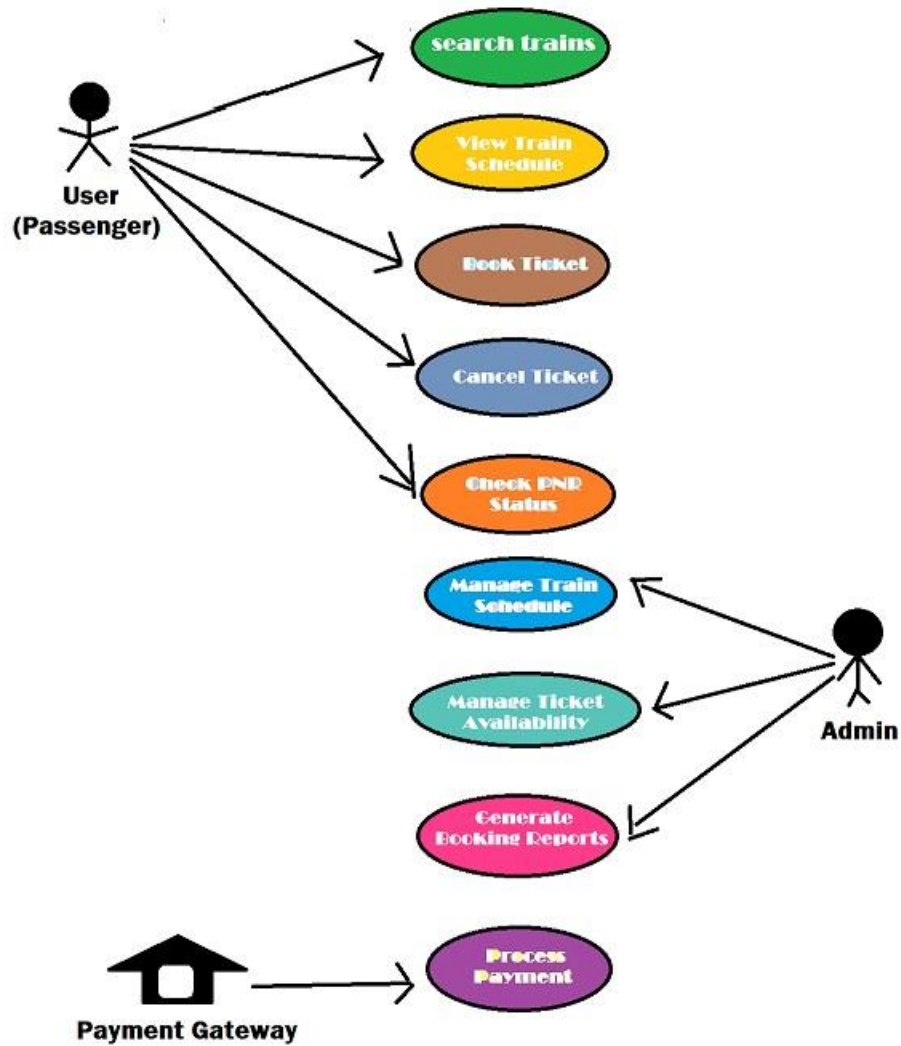
- There is no planning required.
- Simple Model.
- Few resources required.
- Easy to manage.
- Flexible for developers.

## Disadvantage(Cons) of Big Bang Model:

- There are high risk and uncertainty.
- Not acceptable for a large project.
- If requirements are not clear that can cause very expensive.



## USE CASE DIAGRAM FOR THE TRAIN BOOKING SYSTEM

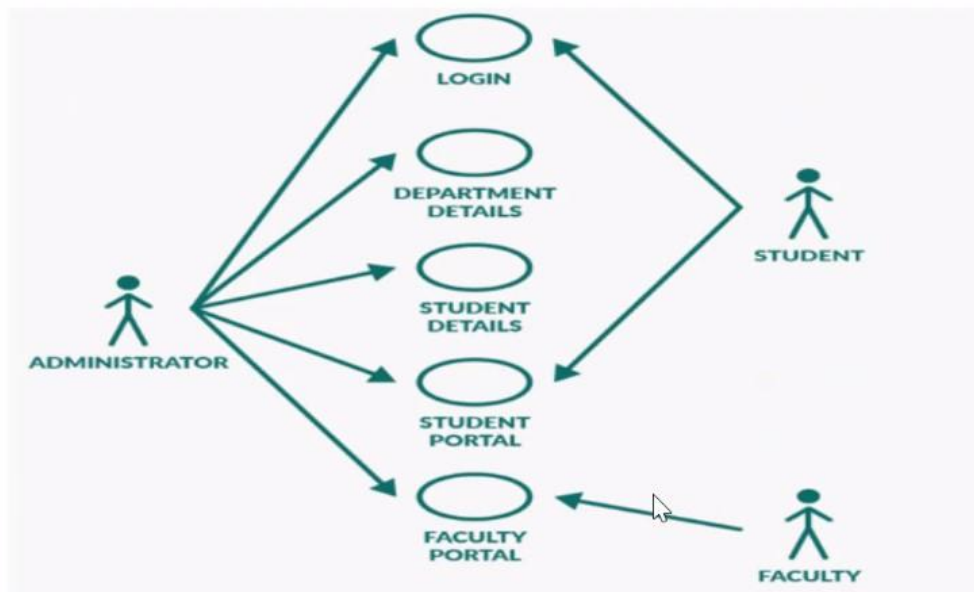




## LOGIN USE CASE EXAMPLE

1. **Introduction** - This use case outlines the steps that need to be followed in order to login into the system
2. **Actors**
3. **Pre-Condition**
4. **Post-Condition**
5. **Basic Flow**
6. **Alternate Flow**
7. **Special Requirements**
8. **Associated Use Case(s)**

## UML(UNIFIED MODELING LANGUAGE) USE CASE DIAGRAM







## IDENTIFICATION OF USE CASES

In the university registration system, we can identify use cases for every actor as follows:

# Use Case	Actor(s)	Description
1 Login	Administrator, Faculty, Student	Login Change Password Forgot Password
2 Department Details	Administrator	Add Department Edit Department Details Delete Department View Department Details
3 Student Details	Administrators	Add New Student Edit Existing Student Delete Existing Student View Existing Student
4 Student Portal	Administrator, Student	Add Student Details Add/Drop Courses View Progress/History
5 Faculty Portal	Administrator, Faculty	Generate Report Program-wise Generate Report Semester-wise