

Lecture – 1

What are CASE Tools?(software that helps automate SDLC steps)

→ Case tools are software applications that help the system analysts, designer and developers automate different steps of the systems development life cycle (SDLC) — like analysis, design, documentation and coding.

এইগুলার মাধ্যমে কাজের গতি বাড়ে এবং ভুল কম হয়।

Example of CASE Tools:

1. Diagramming Tools
2. Analysis Tools
3. Central Repository
4. Documentation Generators
5. Code Generators

Benefits of CASE Tools:

1. Increases productivity and speed
2. Reduces human error
3. Improves documentation quality
4. Makes project management easier

eXtreme Programming (XP): eXtreme Programming (XP) is an agile software development methodology that focuses on improving software quality and responding quickly to changing customer requirements.

এখানে ছোট ছোট ধাপে কাজ করা হয়, যেমন কোড লেখা, টেস্ট করা ইত্যাদি ফাংশনাল ইউনিটে ভাগ করা হয়।

Benefits of eXtreme Programming:

1. **Short incremental development cycles:** work is done in small and improved steps continuously.
(বড় প্রজেক্টকে ছোট ছোট অংশে ভাগ করে অল্প সময়ে তৈরি ও পরিমার্জন করা হয়)
2. **Automated testing:** tests are written before or during coding to ensure quality.
(Coding ও Testing একসাথে করা হয়)
3. **Pair programming:** two programmers work together on one computer — one writes code, the other reviews it instantly.
4. **Continuous feedback:** developers, managers & customers regularly review the software and give feedback.
(নিয়মিত মতামত)

Systems Development Life Cycle (SDLC) of Microsoft: Planning → Analysis → Design → Verification → Release → Response

Criticisms of SDLC:

1. **Forcing timed phases on intangible processes** (analysis & design) is doomed to fail.
SDLC প্রতিটি ধাপে নির্দিষ্ট সময়ের মধ্যে কাজ শেষ করার চাপ দেয়, কিন্তু analysis ও design হলো চিন্তাশীল ও পরিবর্তনশীল ধাপ — সময়ের আগে শেষ করার মানে হচ্ছে কাজ অসম্পূর্ণ থাকা।
2. **Too much formal process and documentation slows things down.**
SDLC-তে অনেক বেশি আনুষ্ঠানিক ধাপ ও ডকুমেন্টেশনের প্রয়োজন হয়, ফলে কাজের গতি ধীর হয়ে যায় এবং বাস্তবায়ন বিলম্বিত হয়।
3. **Cycles are not necessarily waterfalls.**
বাস্তব দুনিয়ায় ডেভেলপমেন্ট একসাথে ধাপে ধাপে হয় না; অনেক সময় ধাপগুলো একত্রে চলে বা পুনরায় হয়, কিন্তু SDLC-তে এই ধরনের flexibility দেখা যায় না।

Lecture 2

Six Sources of Software (Scenario Based Comparison Question):

1. **Information Technology Service Firms:** এই ধরনের কোম্পানিগুলো ক্লায়েন্টের প্রয়োজন অনুযায়ী custom software তৈরি করে দেয়।
তারা সাধারণত বড় প্রজেক্টে কাজ করে যেমন: Infosys, Accenture, Tata Consultancy Services (TCS).
 - Custom-made software for each client
 - Client pays for design, development & maintenance
 - Long-term contracts and professional project management
2. **Packaged Software Providers:** এরা এমন সফটওয়্যার তৈরি করে যা অনেক ইউজার একসাথে ব্যবহার করতে পারে।
এগুলোকে “off-the-shelf software” বলা হয় যেমন: Microsoft Office, Adobe Photoshop, QuickBooks.
 - Ready-made, mass produced
 - Same version for all customers
 - Installed locally on computers
3. **Vendors of Enterprise-wide Solution Software:** এরা বড় প্রতিষ্ঠানের জন্য integrated system সরবরাহ করে যা বিভিন্ন বিভাগ (HR, Finance, Inventory) একসাথে যুক্ত করে। যেমন: SAP ERP, Oracle E-Business Suite, Microsoft Dynamics.
 - Covers the whole organization
 - Central database
 - Complex installation and maintenance
4. **Cloud Computing:** কম্পিউটারের ইন্টারনেটের মাধ্যমে ব্যবহার করা হয় (Software as a Service – SaaS)।
ইউজারদের ইন্সটল করতে হয় না যেমন: Google Workspace, Salesforce, Dropbox, Zoom.
 - Hosted online
 - Pay-per-use or subscription
 - Accessible anywhere
5. **Open-source Software:** এমন সফটওয়্যার যার source code সবার জন্য উন্মুক্ত — যে কেউ দেখতে, পরিবর্তন করতে বা উন্নত করতে পারে। যেমন: Linux, MySQL, Apache, Mozilla Firefox.
 - Free or low-cost
 - Community-based development
 - Transparent source code
6. **In-house Development:** প্রতিষ্ঠানটি নিজস্ব টিম ও প্রোগ্রামার দিয়ে নিজেই সফটওয়্যার তৈরি করে।
যেমন: ব্যাংক, টেলিকম কোম্পানি বা বিশ্ববিদ্যালয় নিজেদের management system তৈরি করে।
 - Developed internally
 - Controlled entirely by the organization
 - Best suited for company-specific needs

Comparison Table (Based on Six Sources of Software)

| Source | Customization | Cost | Time | Example | Best For |
|----------------------|---------------|-----------|--------|-------------|----------------------------|
| IT Service Firms | Very high | High | Long | Infosys | Unique business needs |
| Packaged Software | Low | Low | Short | MS Office | General users |
| Enterprise Vendors | Medium-high | Very high | Long | SAP ERP | Large organizations |
| Cloud Computing | Medium | Medium | Short | Salesforce | Flexible online access |
| Open Source | High | Free/Low | Medium | Linux | Tech-savvy users |
| In-house Development | Very high | High | Long | Bank system | Specialized internal needs |

Ranking of These 6 Sources of Software (অত্যন্ত ভালো):

1. Cloud Computing
2. Open-source Software
3. Packaged Software
4. IT Service Firms
5. In-house Development
6. Vendor of Enterprise-wide Solution Software

Outsourcing: Outsourcing means hiring another company or outside experts to perform services or build systems that your organization would otherwise do itself.

Outsourcing হলো এমন একটি প্রক্রিয়া যেখানে কোনো প্রতিষ্ঠান নিজের কাজ নিজে না করে, বাইরের কোম্পানি বা বিশেষজ্ঞদের দিয়ে সেই কাজ করিয়ে নেয়।

যেমন: কোনো ব্যাংক নিজে সফটওয়্যার না বানিয়ে সেটি অন্য কোম্পানিকে দিয়ে তৈরি করানো।

Advantages of Outsourcing

| Advantage | Explanation |
|------------------------|--------------------------------|
| Cost saving | কম খরচে উৎপাদন কার্য |
| Faster delivery | দ্রুত কাজ শেষ |
| Access to experts | দক্ষ ও অভিজ্ঞ জনবল ব্যবহার |
| Focus on core business | মূল ব্যবসায় ফোকাস দেওয়া যায় |
| Advanced technology | নতুন tools ব্যবহারের সুযোগ |

Lecture 1 & 2 – SDLC & Requirements

System Development Life Cycle (SDLC): Planning → Analysis → Design → Implementation → Maintenance

Functional Requirements

Define how the system should *do* — they describe the main features and functions.

Examples:

1. Users can log in & log out.
2. The system should have customer data.
3. The app should send a confirmation email after registration.
4. Students can view their exam results.

Non-Functional Requirements: Define how the system should *perform* — they describe the quality, performance, and constraints.

Examples:

1. Performance → The system should load within 3 seconds.
2. Security → Data must be secure.
3. Scalability → The system should handle 10,000 users at once.
4. Compatibility → The application must work with Windows, macOS and Linux.

Agile Methodology Key Principles

1. Adaptive rather than predictive.
2. Emphasize people rather than roles.
3. Self-adaptive process for flexibility and self-correction.

When to Use (Agile) Methodology

- **Changing Requirements:** Unpredictable or dynamic requirements.
- **Faster Delivery:** Need to deliver quickly and regularly.
- **Customer Involvement:** Customers understand the process and get involved.

Rational Unified Process (RUP)

Four phases:

1. **Inception** → Define scope, vision & feasibility (*Planning*)
2. **Elaboration** → Analyze problem domain and establish architecture (*Analysis phase*)
3. **Construction** → Build software based on architecture (*Coding & Testing*)
4. **Transition** → Deliver software to users and ensure it meets needs (*Deploy & Maintenance*)

RFP (Request for Proposal): When a company needs new software, it asks vendors to submit proposals (RFP process).

Software Re-Use: আগের বানানো software বা কোড নতুন প্রজেক্টে ব্যবহার করাই Re-Use।

Types of Re-Use

1. **Object-Oriented Development** → একই Class/Object Project-এ use করা।
2. **Component-Based Development** → Use small ready-made components.

Benefits of Re-Use

1. Time saving
2. Less error
3. Less cost

3 Steps of Re-Use

1. **Abstraction** → Reusable code design
2. **Storage** → Save in library or repository
3. **Recontextualization** → Understandable code design যাতে অন্য developer-রা ব্যবহার করতে পারে

Approaches of Re-Use

| Approach | Meaning |
|-------------|---|
| Ad-hoc | Developer নিজের ইচ্ছা মতো Re-use করবে |
| Facilitated | Company encourages developers to Re-use |
| Managed | Company officially has a Re-use system (বাধ্যতামূলক ভাবে) |
| Designed | শুরু থেকেই Re-use এর জন্য বানানো হয় |

Why Select Off-the-Shelf Software

- **Cost:** কম খরচে প্রস্তুত সফটওয়্যার সমাধান মেলা যায়।
- **Functionality:** প্রয়োজনীয় বৈশিষ্ট্য আগেই থাকে।
- **Vendor Support:** সমস্যা হলে কোম্পানির থেকে সহায়তা পাওয়া যায়।
- **Vendor Viability:** দীর্ঘমেয়াদি চুক্তির নিরাপত্তা নিশ্চিত হয়।
- **Flexibility:** ইউজারের প্রয়োজন অনুযায়ী পরিবর্তন সম্ভব।
- **Documentation:** ব্যবহারকারীর সহজ নির্দেশ ও গাইড থাকে।

Lecture 3 – Project Management

Core Concepts

1. **Project:** নির্দিষ্ট মূল্য ও সময়ের ভিত্তিতে লক্ষ্য সম্পন্ন করার কাজ।
2. **Project Management:**
প্রকল্প শুরু, পরিকল্পনা, বাস্তবায়ন ও সমাপ্তির নিয়ন্ত্রিত প্রক্রিয়া।

Controlled Process:

- **Initiating** → Relationship with customer / project environment
- **Planning** → Define scope / risk analysis
- **Executing** → Execute plan / manage changes / maintain workbook
- **Closing** → Close, conduct reviews & close contract

Project Manager's Role

- **Customer and Management Expectations:** গ্রাহক ও ব্যবস্থাপনার চাহিদা বুঝে তা পূর্ণ করা।
- **Documentation & Communication:** প্রকল্প সম্পর্কিত সব নথিপত্র ও যোগাযোগ নিয়ন্ত্রণ।
- **Time & Resource Constraints:** নির্দিষ্ট সময় ও সীমিত সম্পদ ব্যবহার করে কাজ সম্পন্ন করা।
- **Methodologies & Tools:** Agile, Scrum, Gantt Chart ব্যবহার করা।
- **Managing People:** টিমের সদস্যদের কাজ সমন্বয় ও অনুপ্রাণিত করা।
- **Contractors and Vendors:** বাহ্যিক সংস্থার সাথে সম্পর্ক রক্ষা।
- **Organizational Change and Complexity:** সংগঠনের পরিবর্তন ও অভিযোজন নিয়ন্ত্রণ।

Lecture 3.2: Gantt Charts vs Network Diagrams

| Criteria | Gantt Chart | Network Diagram |
|-----------------------|---|--|
| Definition | A bar chart that shows the start and end dates of tasks along a timeline. | A diagram that shows relationships and sequence between project activities using nodes (circles) and arrows. |
| Representation | Uses horizontal bars to represent the time duration of each task. | Uses circles (nodes) for activities and arrows for their dependencies (sequence). |

| | | |
|-------------------|---|--|
| Focus | Focuses on time scheduling — when tasks start and finish. | Focuses on logical order and dependency — which task depends on which. |
| Purpose | To track project progress over time. | To analyze the workflow and sequence of tasks. |
| Complexity | Easy to read but becomes cluttered for large projects. | Better for complex projects to show inter-task relationships clearly. |
| Example | A Gantt chart may show Task A: 1–5 days, Task B: 6–10 days. | A Network Diagram shows Task A → Task B → Task C (dependency flow). |

Lecture 3.3 (Continued)

Gantt Chart:

টাইমলাইন ভিত্তিক চার্ট — project এর প্রতিটি কাজ কত দিন লাগবে এবং কখন শুরু ও শেষ হবে তা দেখায়।

Gantt Chart Example:

(Shows tasks from March to November for Task C1, C2, etc.)

Network Diagram: কার্যগুলোর relationship ও order দেখায় — কোন কাজ শেষ না হলে পরবর্তী কাজ শুরু হবে না।

Example: Design System → Write Programs → Test Programs → Write Documentation → Install System

Backward / Forward Pass Math:

$EF = ES + \text{duration} - 1$ (for forward pass)

$LS = LF - \text{duration} + 1$ (for backward pass)

Collected from University slide.

Network Diagram Example (Critical Path Calculation)

Collected from:

<https://youtu.be/4oDLMSilExs?si=N40-mepzrGh26uUA>

Here:

A → B → D → G → H = Critical Path

C → E → F = Non-Critical Path

Slack = LS – ES

C = (11–5) = 6

E = (12–6) = 6

F = (16–7) = 9

Terminologies:

LS = Latest Start

LF = Latest Finish

ES = Earliest Start

EF = Earliest Finish

Duration Box:

ES EF

LS LF

Critical Path: EF = LF

- **Backward pass:** এর সময় ছোট সংখ্যা নেয়া হয়
- **Forward pass:** এর সময় বড় সংখ্যা নেয়া হয়

ES/EF (Forward pass): এর মাধ্যমে কাজ কখন শুরু হবে
LS/LF (Backward pass): এর মাধ্যমে কাজ কখন শেষ হবে

Lecture 3.4

Network Diagram

Definition: “কোন কাজের পর কোন কাজ হবে” (Dependency ভিত্তিক চার্ট)

Diagram Example

Design System → Write Programs → Test Programs → Install System
↘ Write Documentation ↗

Step-by-Step Explanation

| Step | Work | Explanation (Bangla) |
|------|---------------------|---|
| A | Design System | প্রকল্পের প্রথম ধাপ, সিস্টেম ডিজাইন করা। |
| B | Write Programs | সিস্টেম ডিজাইনের শেষ হলে প্রোগ্রাম লেখা শুরু হয়। |
| C | Test Programs | প্রোগ্রাম লেখা শেষ হলে তা পরীক্ষা করা হয়। |
| D | Write Documentation | একই সময়ে বা পরে ডকুমেন্টেশন লেখা হয়। |
| E | Install System | টেস্ট ও ডকুমেন্টেশন শেষ হলে সিস্টেম ইনস্টল করা হয়। |

Observation

- A → B → C → E হলো মূল পথ (main workflow)
- A → D → E হলো একটি বিকল্প বা সমান্তরাল পথ (parallel task)

PERT Calculations

PERT (Program Evaluation and Review Technique) নির্ভর করে নিম্নলিখিত তিনটি সময় অনুমানের ওপর —

Three Time Estimates:

1. Optimistic (Least)
2. Pessimistic (Greatest)
3. Realistic

Example Table

| Activity | Time Estimate (weeks) | Expected Time (ET) | Formula: $(O + 4R + P) \div 6$ |
|----------------------------|-----------------------|--------------------|--------------------------------|
| 1. Requirements Collection | 1, 5, 9 | 5 | |
| 2. Screen Design | 5, 6, 7 | 6 | |
| 3. Report Design | 3, 6, 9 | 6 | |
| 4. Database Design | 1, 2, 3 | 2 | |
| 5. User Documentation | 2, 6, 7 | 5.5 | |
| 6. Programming | 3, 5, 6 | 5 | |
| 7. Testing | 1, 3, 5 | 3 | |
| 8. Installation | 1, 1, 1 | 1 | |

Path Calculation

Diagram Overview: Critical path and non-critical path are calculated based on **Earliest Time (TE)** and **Latest Time (TL)**.

Critical Path: Path with zero slack.

Non-Critical Path: Path with positive slack (> 0).

Path Calculation Results

| Activity | TE | TL | Slack (TL – TE) | On Critical Path |
|----------|------|----|-----------------|------------------|
| 1 | 5 | 5 | 0 | ✓ |
| 2 | 11 | 11 | 0 | ✓ |
| 3 | 11 | 11 | 0 | ✓ |
| 4 | 13 | 13 | 0 | ✓ |
| 5 | 18.5 | 21 | 2.5 | ✗ |
| 6 | 18 | 18 | 0 | ✓ |
| 7 | 21 | 21 | 0 | ✓ |
| 8 | 22 | 22 | 0 | ✓ |

Slack → বাড়তি সময়

উদাহরণ: উপরের চিত্রে আমরা দেখছি (Activity 5 → TL = 21, TE = 18.5)।

তাহলে **Slack** = $(21 - 18.5) = 2.5$ days.

অর্থাৎ Activity 5 প্রকল্প ২.৫ দিন দেরি করলেও প্রকল্প ঠিক সময়ে শেষ করা যাবে।

Critical Path → যে কাজগুলির Slack = 0, তারা প্রকল্পের মূল Critical Path এ থাকে। তাদের এক ধাপ দেরি মানে পুরো প্রকল্প দেরি হবে।

Forward & Backward Pass:

- **TE (Forward Pass):** প্রথম থেকে শেষ পর্যন্ত যাওয়া সময়।
- **TL (Backward Pass):** শেষ থেকে প্রথম পর্যন্ত যাওয়া সময়।

Symbol Table

| Symbol | Meaning | Bangla Explanation |
|------------------------------|---------------|---|
| ET | Elapsed Time | এই কাজ করতে যত সময় লাগে |
| TE (Forward Pass) | Earliest Time | যত দ্রুত কাজটা শেষ করা যায় |
| TL (Backward Pass) | Latest Time | যত দেরিতে শেষ করলেও প্রকল্প Ready হতে দেরি হবে না |
| SLACK | TL – TE | কাজে দেরি করার সুযোগ আছে |
| Critical Path | Slack = 0 | এই কাজ দেরি করলে প্রকল্পে বিলম্ব হবে |
| Non-Critical / Flexible Path | Slack > 0 | এই কাজ দেরি করার সুযোগ রয়েছে |

Lecture 4

Identifying and Selecting IS Development Projects

Definition: Information System (IS) Development Project মানে হচ্ছে এমন একটি প্রজেক্ট যা কোনো প্রতিষ্ঠানের জন্য ব্যবসায়িক উন্নয়নের উদ্দেশ্যে তৈরি করা হয় — যেমন নতুন সফটওয়্যার বানানো, ডাটাবেস তৈরি, বা সিস্টেম আপগ্রেড করা।

Top-Down and Bottom-Up Sources:

| Type | Description | Example |
|---------------------------|--|---|
| Top-Down Planning | Top-down source are projects identified by top management or by a diverse steering committee. (প্রজেক্টের আইডিয়াটা আসে Top Management বা Steering Committee থেকে) | CEO বলল, “একটা নতুন ERP সিস্টেম দরকার।” |
| Bottom-Up Planning | Bottom-up source ideas come from manager, department or development group workers. | HR টিম বলল, “একটা attendance tracking অ্যাপ দরকার।” |

Selecting IS Development Projects (প্রজেক্ট নির্বাচন): অর্থাৎ — বিভিন্ন প্রকল্প আইডিয়াগুলোর মধ্যে থেকে এমন প্রজেক্ট নির্বাচন করা যা প্রতিষ্ঠানের জন্য সবচেয়ে বেশি লাভ দেখায় এবং কৌশলগত (strategy) মানদণ্ডে মেলে।

বিবেচনার মানদণ্ড:

- কোম্পানির সাথে মিল আছে কি না (Strategic fit)
- খরচ ও লাভ (Cost-benefit ratio)
- ঝুঁকি (Risk)
- সময় (Time)
- সম্পদের প্রাপ্যতা (Resources)

Project Selection Decision Diagram:

Perceived and Real Needs
↓
List of Potential and Ongoing Projects
↓
Current Organizational Environment
↓
Evaluation Criteria



Project Selection Decision

- Accept Project
- Reject Project
- Delay Project
- Reduce Project
- End-User Development
- Proof of Concept



Existing and Available Resources

Weighted Multicriteria Analysis

Formula:

$$\text{Score} = \sum (\text{Weight} \times \text{Rating})$$

| Criteria | Weight | Alternative A | Alternative B | Alternative C |
|-----------------------------|--------|---------------|---------------|---------------|
| Requirements | | | | |
| Real-time data entry | 18 | 5×18=90 | 5×18=90 | 5×18=90 |
| Automatic reorder | 18 | 5×18=90 | 5×18=90 | 5×18=90 |
| Real-time data query | 14 | 5×14=70 | 5×14=70 | 5×14=70 |
| Subtotal | 50 | 250 | 250 | 250 |
| Constraints | | | | |
| Developer costs | 15 | 4×15=60 | 5×15=75 | 3×15=45 |
| Hardware costs | 15 | 4×15=60 | 4×15=60 | 3×15=45 |
| Operating costs | 15 | 5×15=75 | 1×15=15 | 5×15=75 |
| Ease of training | 5 | 5×5=25 | 3×5=15 | 3×5=15 |
| Subtotal | 50 | 220 | 165 | 180 |
| Total | 100 | 342 | 415 | 430 |

Highest total score = Selected project

Electronic Commerce (E-Commerce)

| Type | Meaning | Example |
|------------|----------------------|---------------------------|
| B2C | Business → Customer | Amazon, Daraz |
| B2B | Business → Business | Dell ↔ Intel |
| B2E | Business → Employees | HR portal, payroll system |

Illustration:

Individual → Business → Business
 (B2C → B2B → B2E)

Types of Planning Matrices

| Matrix Type | সম্পর্ক | Example |
|------------------------------|--------------------------------------|--|
| Location to Function | কোন স্থান কোন কাজ করে | যেমন ধরো “Sales” শাখা, অথবা “Support” শাখা |
| Location to Unit | কোন স্থান কোন বিভাগ করে | যেমন সদর দপ্তর HR, ফ্যাক্টরি Production |
| Unit to Function | কোন বিভাগ কোন কাজ করে | Finance → Budgeting, HR → Recruitment |
| Function to Objective | কোন কাজ কোন লক্ষ্য পূরণে সাহায্য করে | Sales → Increase revenue |

| | | |
|--|--|---|
| Function to Process | কোন কাজের সাথে কোন প্রক্রিয়া যুক্ত | Sales → Order Entry, Invoice Process |
| Function to Data Entity | কোন ফাংশন কোন ডেটা ব্যবহার করে | Payroll uses Employee Data |
| Process to Data Entity | কোন প্রক্রিয়াতে কোন ডেটা দরকার | Order Process → Order Table, Customer Table |
| Process to Information System | কোন সিস্টেম এই প্রক্রিয়া সাপোর্ট করে | Order Entry → Sales System |
| Data Entity to Information System | কোন সিস্টেম এই ডেটা ব্যবহার করে | Customer Table → CRM System |
| Information System to Objective | কোন ইনফরমেশন সিস্টেম কোন লক্ষ্য অর্জনে সাহায্য করে | ERP System → Increase efficiency |

Lecture 5 – Assessing Project Feasibility

Definition: Feasibility assessment means determining whether a proposed information system project is possible, practical, and worthwhile for the organization.

Feasibility মানে হলো — কোনো ইনফরমেশন সিস্টেম বাস্তবে তৈরি করা সহজ কি না, প্রতিষ্ঠানের জন্য লাভজনক কি না, এবং তা প্রযুক্তিগতভাবে বাস্তবসম্মত কি না, তা মূল্যায়ন করা।

Purpose: To evaluate if the project should continue, be modified, or be rejected before investing large resources.

Main Types of Feasibility

| Type | Explanation | Bangla |
|--|---|---|
| Economic Feasibility | Checks cost vs. benefit (is it financially worthwhile?) | খরচ ও লাভ বিশ্লেষণ করা হয়। |
| Technical Feasibility | Can we build it with existing technology & skills? | প্রয়োজনীয় প্রযুক্তি ও দক্ষতা আছে কি না দেখা হয়। |
| Operational Feasibility | Will it work well and be accepted by users? | সিস্টেমটি ব্যবহার যোগ্য ও ইউজাররা গ্রহণ করবে কি না। |
| Scheduling Feasibility | Can it be finished on time? | নির্দিষ্ট সময়ের মধ্যে শেষ করা যাবে কি না। |
| Legal & Contractual Feasibility | Any legal or policy restrictions? | কোনো আইনি বা নীতিগত সমস্যা আছে কি না। |
| Political Feasibility | Will management and employees support it? | কর্তৃপক্ষ ও কর্মচারীরা সমর্থন করবে কি না। |