Absolutely, Tathagata! Here's a structured summary of the \*\*main topics\*\* covered in your “Manual Testing Notes” document, with brief explanations for each to make it easy to grasp and revise:

---

### 🧪 \*\*Software Testing Basics\*\*

- \*\*What is Software Testing?\*\*

It’s the process of identifying bugs and ensuring the software meets client requirements before release.

- \*\*Quality in Software\*\*

Defined by how well software meets user expectations—bug-free, on time, within budget.

---

### 🔄 \*\*Software Development Life Cycle (SDLC)\*\*

- \*\*Phases\*\*: Requirement Analysis → Design → Development → Testing → Maintenance

- \*\*Models Covered\*\*:

- \*\*Waterfall\*\*: Sequential, best for small projects with fixed requirements.

- \*\*V-Model\*\*: Emphasizes verification & validation in parallel.

- \*\*Spiral & Incremental Models\*\*: Iterative and feature-based releases.

---

### 🧰 \*\*Types of Testing\*\*

#### 📌 Static Testing

- Done without executing code (e.g., document reviews, walkthroughs, inspections)

#### 📌 Dynamic Testing

- Executing the code and evaluating runtime behavior

#### 🔍 By Approach:

- \*\*White Box Testing\*\*: Internal logic testing (done by developers)

- \*\*Black Box Testing\*\*: Functionality testing without looking at code (done by testers)

- \*\*Grey Box Testing\*\*: Combination of both

#### 🧱 Levels of Testing:

1. \*\*Unit Testing\*\* – Developers test individual components

2. \*\*Integration Testing\*\* – Check module interactions

3. \*\*System Testing\*\* – Full system validation

4. \*\*Acceptance Testing (UAT)\*\* – End-user verification

---

### 🧠 \*\*Test Design Techniques (Black Box)\*\*

- \*\*ECP (Equivalence Class Partitioning)\*\* – Divide input data into valid/invalid classes

- \*\*BVA (Boundary Value Analysis)\*\* – Focus on edge input values

- \*\*Decision Table\*\* – For complex logic with multiple conditions

- \*\*State Transition\*\* – Behavior based on previous states

- \*\*Error Guessing\*\* – Based on tester’s intuition

---

### 🚦 \*\*Types of Testing Scenarios\*\*

- \*\*Smoke Testing\*\* – Basic check to ensure build is testable

- \*\*Sanity Testing\*\* – Focused testing on new changes

- \*\*Regression Testing\*\* – Ensure old functionalities work after changes

- \*\*Retesting\*\* – Re-executing failed tests

- \*\*Exploratory/Ad-hoc/Monkey Testing\*\* – No formal plans; done to uncover hidden bugs

---

### 📊 \*\*Test Artifacts\*\*

- \*\*Test Scenario\*\* – What to test

- \*\*Test Case\*\* – Steps, input, expected outcome

- \*\*Test Suite\*\* – Group of related test cases

- \*\*Test Plan\*\* – Strategy, scope, schedule of testing

- \*\*RTM (Requirement Traceability Matrix)\*\* – Ensures test coverage of requirements

---

### 🐞 \*\*Bug Lifecycle & Severity\*\*

- \*\*Error → Defect → Bug → Failure\*\* (based on who finds it and when)

- \*\*Bug Status Flow\*\*: New → Assigned → Fixed → Retest → Closed

- \*\*Severity\*\* (impact) vs. \*\*Priority\*\* (urgency)

---

### ⚙️ \*\*System Testing Examples\*\*

Includes GUI, usability, functional, non-functional (like load, stress, compatibility), and security testing.

---

### 📂 \*\*Real-World Project Workflow\*\*

- Covers practical testing steps: understanding the project, writing scenarios, execution, logging defects, and creating reports (e.g., for an e-commerce platform like OpenCart)