**MSSV: DE180292**

**Họ và Tên: Nguyễn Kim Đăng Khoa**

**Lớp: SE18D12**

**Exercise 1: Explain types of review.**

Based on Chapter 3, Section 3.2.3 of "Foundations of Software Testing (4th Edition)"

**1. Informal Review**

- Definition: A review process with no formal process, documentation, or defined roles.  
- Purpose: To quickly check documents, designs, or code for obvious issues or misunderstandings.  
- Characteristics:  
 • Performed by peers or colleagues.  
 • Communication is done via email, chat, or in-person.  
 • No official record of defects.  
- Use Case: Early-stage documents, design brainstorming, or quick peer feedback.  
- Strength: Fast and flexible.  
- Limitation: No measurable quality control or tracking.

**2. Walkthrough**

- Definition: A semi-formal review where the author leads the session and presents the content.  
- Purpose: To increase understanding, gather feedback, and identify issues.  
- Characteristics:  
 • Led by the author.  
 • Reviewers may ask questions.  
 • No formal preparation is required.  
- Use Case: Reviewing new designs or onboarding new team members.  
- Strength: Promotes understanding across teams.  
- Limitation: May lack deep defect detection.

**3. Technical Review**

- Definition: A formal review focused on technical quality.  
- Purpose: To evaluate the suitability of the software product technically.  
- Characteristics:  
 • Led by a qualified reviewer (not the author).  
 • Reviewers prepare using objectives or checklists.  
 • May include formal reporting.  
- Use Case: Architecture documents, source code, or algorithms.  
- Strength: Ensures conformance to standards.  
- Limitation: Requires skilled reviewers and time.

**4. Inspection**

- Definition: The most formal and structured review process.  
- Purpose: To detect defects systematically and improve quality.  
- Characteristics:  
 • Roles: Moderator, Author, Reader, Recorder, Reviewers.  
 • Uses checklists, defect logs, and metrics.  
 • Includes meetings and follow-ups.  
- Use Case: Critical documents like requirements or test plans.  
- Strength: Highly effective in early defect detection.  
- Limitation: Time-consuming and resource-intensive.

**Comparison Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Type | Formality | Led by | Preparation | Records Defects | Common Use |
| Informal Review | Low | Anyone | No | No | Peer review, early drafts |
| Walkthrough | Medium | Author | Optional | No | Knowledge transfer |
| Technical Review | High | Expert | Yes | Sometimes | Code/design validation |
| Inspection | Very High | Moderator | Yes | Yes | Contracts, critical requirements |

**Professional Insight**

In real-world software projects:  
- Inspection is highly effective and can detect up to 90% of defects in early phases.  
- Walkthroughs help ensure shared understanding among teams.  
- Technical Reviews validate performance, security, and maintainability.  
  
Choosing the right review type depends on the project phase, criticality of documents, and team maturity.

**Exercise 2: Liệt kê ra các lỗi ở các requirements sau:**

**Requirement Defects and Improved Requirements – Student Management System**

**📌 Part 1: Defect Classification in Original Requirements**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Original Requirement | Defect Type | Explanation | Fix Suggestion |
| 1 | The system shall be easy to use for all teachers. | Ambiguous | ‘Easy to use’ is vague and subjective. | The system shall allow teachers to complete grading tasks within 3 steps. |
| 2 | Students should quickly find their grades. | Non-specific | 'Quickly' is undefined and non-measurable. | Students shall be able to view their grades within 2 seconds after login. |
| 3 | The app must always work on any computer. | Unrealistic / Non-specific | 'Always work' is absolute and 'any computer' is too broad. | The app shall support Windows 10+, macOS 11+, and Ubuntu 20.04+. |
| 4 | User data will be kept safe. | Unverifiable | 'Safe' is vague and unverifiable. | User data shall be encrypted using AES-256 and stored securely. |
| 5 | The interface shall be nice and professional. | Ambiguous | 'Nice' and 'professional' are subjective. | The interface shall follow the institution’s UI design standards and WCAG 2.1. |
| 6 | Teachers and admins can approve or reject grades and export reports. | Compound requirement | Multiple actions in one requirement. | Split into two: approve/reject grades, and export reports separately. |
| 7 | The grade list shall be updated on time. | Ambiguous / Non-specific | 'On time' is undefined. | The grade list shall be updated within 2 minutes of submission. |
| 8 | All users shall be satisfied with the system. | Unverifiable | 'Satisfied' is subjective and non-measurable. | Remove or replace with measurable user survey metric. |
| 9 | The system will never lose any student data. | Unrealistic | 'Never lose' is absolute and impractical. | The system shall back up student data every 10 minutes and log changes. |
| 10 | The UI should be responsive and beautiful. | Ambiguous + Non-specific | ‘Beautiful’ is subjective, ‘responsive’ is vague. | The UI shall support screens ≥768px and use responsive layout standards. |
| 11 | The system shall automatically assign student IDs logically. | Ambiguous | 'Logically' is undefined. | Student IDs shall be assigned sequentially with admission year prefix. |
| 12 | The application shall run well on Windows, macOS, and Linux. | Non-specific | 'Run well' is undefined. | The application shall start within 5 seconds and remain responsive under normal load. |
| 13 | Students may access their records from anywhere at any time. | Ambiguous / Unrealistic | 'Anywhere, anytime' lacks constraints. | Students shall access records 24/7 via web portal with 2FA authentication. |
| 14 | The code must be readable and easy to maintain. | Design vs Requirement | This refers to implementation, not system behavior. | Move to developer coding guidelines, not part of functional requirements. |
| 15 | Teachers can grade students and notify parents instantly. | Compound + Non-specific | Two actions in one; 'instantly' undefined. | Split: 1) Grade entry; 2) Notifications within 5 minutes. |
| 16 | The system should work fast with a large number of students. | Ambiguous + Non-specific | 'Fast' and 'large number' undefined. | The system shall respond within 2s for 95% of requests with 10,000 users. |
| 17 | It shall detect possible errors in transcripts intelligently. | Ambiguous + Unverifiable | 'Intelligently' is vague. | The system shall detect missing grades or duplicate entries. |
| 18 | The export feature shall support Excel, PDF, CSV, and more. | Non-specific + Compound | 'And more' is vague; multiple formats listed. | Support exporting to Excel (.xlsx), PDF, and CSV formats. |
| 19 | The app shall always show the best format for transcripts. | Ambiguous / Unverifiable | 'Best format' is subjective. | The app shall show transcripts in tabular layout with GPA summary. |
| 20 | It should never crash under any condition. | Unrealistic | Absolute and unverifiable statement. | The app shall handle errors gracefully and log crash reports. |

**✅ Part 2: Rewritten “Good” Requirements**

1. The system shall allow teachers to complete grading tasks within 3 steps.

2. Students shall be able to view their grades within 2 seconds after login.

3. The app shall support Windows 10+, macOS 11+, and Ubuntu 20.04+.

4. User data shall be encrypted using AES-256 and stored securely.

5. The interface shall follow the institution’s UI design standards and WCAG 2.1.

6. Teachers and admins shall approve/reject grades and export reports via dashboard actions.

7. The grade list shall be updated within 2 minutes of a teacher submitting changes.

8. The system shall achieve a user satisfaction score of at least 4.0/5 in quarterly surveys.

9. The system shall back up student data every 10 minutes and maintain change logs.

10. The UI shall support screen widths ≥768px and use responsive layout standards.

11. The system shall assign student IDs sequentially with admission year prefixes.

12. The application shall start within 5 seconds and respond under 1s in normal load.

13. Students shall access records 24/7 via web portal with 2FA authentication.

14. [Moved to developer documentation – not a requirement].

15. The system shall notify parents within 5 minutes after grades are submitted.

16. The system shall respond to 95% of requests within 2s with 10,000 users.

17. The system shall detect and report missing grades or duplicate course entries.

18. The export feature shall support Excel (.xlsx), PDF, and CSV file formats.

19. The app shall display transcripts in tabular layout with GPA summary.

20. The app shall handle all critical errors gracefully and log crash reports.