Software Requirement Engineering

Assignment # 2

(**Deadline: FRIDAY | 25/04/2025 | 05:00PM**)

COURSE CODE: SE211 **INSTRUCTOR:** MUHAMMAD HUZAIFA SHAH

TOTAL MARKS: 02

Name: Mian M Owais Reg No: 2023-316

Assignment Instructions:
- This is an individual assignment.
- Read the assignment carefully, understand what is required and write appropriate answers.

Submission Method:
- ✓ Each student must submit a **Hard Copy** to the instructor in the office before the deadline.

General Instructions:
- **Late Submission Policy: 1% absolute deduction in overall.**
- Use of **AI tools** is **NOT** encouraged.

Assignment Title: Identifying and Correcting Requirement Errors

1. Common Types of Requirement Errors

Error Type	Brief Explanation
Ambiguous Requirements	Unclear terms open to multiple interpretations. E.g., 'user-friendly'. Avoid being precise.
Incomplete Requirements	Missing necessary information or context. Ensure who, what, when, how are covered.
Inconsistent Requirements	Conflicting statements. E.g., response time <2s vs. <5s. Harmonize with stakeholders.
Non-Testable Requirements	Cannot verify through testing. Must be measurable and specific.
Over-Specification / Gold Plating	Too much technical/implementation detail. Focus on 'what', not 'how'.
Vague / Subjective Language	Use of unclear terms like 'fast', 'nice'. Replace with measurable criteria.
Scope Creep / Creeping Requirements	Continuous growth of features. Stick to original scope or formally update.
Conflicting Stakeholder Interests	Different needs from different stakeholders. Use prioritization and negotiation.
Untraceable Requirements	No link to goals or sources. Maintain traceability with IDs, matrices.
Unrealistic Requirements	Technically or financially infeasible. Validate feasibility early.

Inflexible / Rigid Requirements	Too strict without need. E.g., platform lock-in. Allow flexibility if possible.
Unprioritized Requirements	No ranking of importance. Use MoSCoW, Kano, or Value-Cost techniques.
Hidden Requirements	Unspoken needs. E.g., legal compliance. Use thorough elicitation techniques.
Poorly Documented Requirements	Not in standard formats. Use SRS templates, diagrams, glossary.
Culturally Insensitive Requirements	Not suitable for diverse user bases. E.g., forcing legal names. Be inclusive.

2. Scenario: Smart City Property Management System (SCPMS)

SCPMS (Smart City Property Management System) is being developed to manage the registration, maintenance, and security of residential properties in smart urban environments. The system will integrate IoT devices, cloud-based services, and user portals for residents, maintenance staff, and administrators.

Disclaimer: Each requirement below is written independently and does not rely on any of the others. ***Your task is to identify the nature of the issue (if any) and revise the requirement using good practices where needed.***

3. Requirement Review Table (30 Requirements)

Requirement Statement	**Identify the Fault**	**Rewritten Requirement (Corrected)**
The system should be user-friendly.	Ambiguous Requirements	The system shall provide an intuitive interface complying with WCAG 2.1 standards.
User shall be notified.	Incomplete Requirements	The system shall send a push notification to the user's mobile app upon detecting a door unlock event.
The system should work fast.	Vague / Non-Testable	The system shall respond to resident queries within 1.5 seconds under normal load.
Code must be in Python.	Over-Specification / Gold Plating	The system shall be developed using industry-standard technologies suitable for scalability and security.
Sensors shall operate efficiently.	Vague / Subjective Language	Motion sensors shall operate with at least 95% detection accuracy and a latency of under 2 seconds.

User login shall be secure.	Vague / Subjective Language	User login shall use two-factor authentication (2FA) over encrypted HTTPS protocol.
The app must be beautiful.	Vague / Subjective Language	The app shall comply with the company's UI/UX design standards and user feedback mechanisms.
Admin can block access.	Incomplete	The system shall allow admin users to revoke access permissions of a resident from the dashboard.
Include AI assistant for help.	Scope Creep / Creeping Requirements	The system may include an AI-based help assistant in Phase 2, depending on resource availability.
Every button shall glow blue.	Over-Specification / Gold Plating	Every interactive button on the resident portal shall have hover and active states following the brand guidelines.
Notifications should be fast.	Vague / Subjective Language	The system shall deliver notifications within 10 seconds of triggering events.
Enable payments via all platforms.	Over-Specification/Gold Plating	The system shall support payments via Visa, MasterCard, and PayPal in Phase 1.
Add facial recognition.	Scope Creep / Creeping Requirements	Facial recognition shall be considered in the future after privacy and legal assessments.
Must support 10 million users instantly.	Unrealistic Requirements	The system shall initially support 100,000 users with scalability up to 10 million.
Show weather on dashboard.	Incomplete	The resident dashboard shall display current weather using data from the national API service.
Secure all data transmissions.	Incomplete Requirement	The system shall encrypt all user and device data using TLS 1.3 protocol.
Include multilingual support.	Incomplete Requirement	The portal shall support English and Spanish languages at launch.
No downtime allowed.	Unrealistic Requirements	System availability shall be 99.9% annually, with scheduled maintenance during non-peak hours.

Generate reports.	Incomplete / Ambiguous	The admin dashboard shall allow monthly report generation in PDF and Excel formats.
Keep residents happy.	Vague / Subjective Language, Non-Testable	The system shall implement a satisfaction feedback module to measure user experience post-maintenance.
System shall respond in less than 3 seconds.	Non-Testable Requirement	The system shall respond to user actions within 2 seconds under normal load conditions.
All resident data shall be backed up daily.	No fault	All resident data shall be backed up daily.
Dashboard shall be accessible via desktop and mobile.	No fault	Dashboard shall be accessible via desktop and mobile.
Support for biometric login (face or fingerprint).	Incomplete / Ambiguous	The system shall allow residents to log in using biometric authentication (face or fingerprint) on supported mobile devices.
Include QR code for visitor check-in.	No fault	Include QR code for visitor check-in.
System must run only on Windows 7.	Inflexible / Rigid Requirements	The system shall be cross-platform, compatible with Windows 10+, macOS, and Linux.
Smart lights must be yellow only.	Over-Specification / Gold Plating	Smart lights shall support adjustable colors including warm white (yellow) and daylight modes.
Resident shall get alerts via magic.	Unrealistic	The system shall notify residents via SMS and app alerts in case of anomalies.
Maintenance logs shall be kept hidden.	Vague / Incomplete	Maintenance logs shall be securely stored and accessible only to authorized users.
Send gift to residents on birthday.	Over-Specification / Unrealistic	The system shall notify admins about resident birthdays to enable manual greetings or actions.