**Student Name:** Joseph MUTANGANA

**Student ID:** 29061

**Course Name:** Software engineering

**SECTION A:**

1. a

2. b

3. b

4. b

5. b

6. e

7. b

8. c

9. c

10. a

**SECTION B:**

11. **Functional Requirements:** Defines what the system should do. while

**Non-Functional Requirements** defines how system performs.

12. **Software reliability:** Is the ability of software to perform without failure

It is important to have the reliability software because it has the standard and it will be accepted by users and customers and bring good feedback.

13. Requirements, Design, Implementation, Testing, Deployment, Maintenance.

14. **Verification:** Checking whether the products is built right. While **Validation** checks if product meets requirements

15. Purpose of software maintenance: Adding new feature and keep it up to date

Types Corrective, Adaptive, Perfective, Preventive

16. **Advantages**: Flexibility, faster delivery.  
  **Disadvantages**: Scope creep, less documentation.

17. **Modularization** breaks the software into smaller parts/modules.  
  It improves maintainability and reduces complexity.

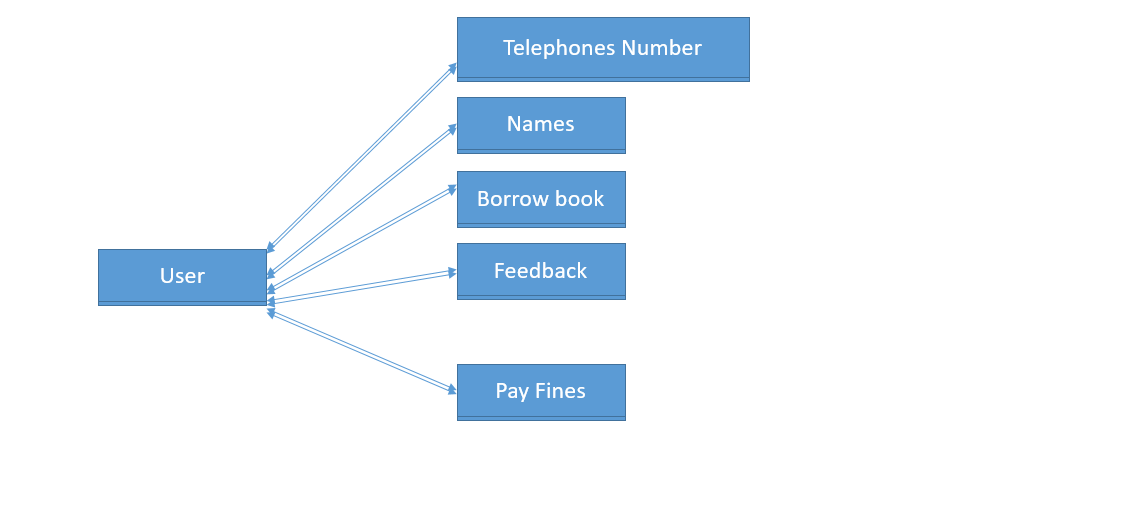
**18. Black-box testing** checks functionality without code knowledge.  
  **White-box testing** checks internal logic using code.

19. **Risk management** identifies and handles potential project problems.  
  Example: Delay due to changing requirements.

20 **Version control** helps track changes, revert mistakes, and manage collaboration safely.

**SECTION C:**

21. **Recommended model**: Spiral or Agile — because banking systems require risk analysis, flexibility, and security.

22. 

23. **Testing type**: Input validation or boundary testing (often part of black-box testing or system testing).

24. **Methodology**: Agile — it handles changing requirements through short, iterative cycles.

25. **Model's drawback**: Waterfall — it assumes clear requirements from the start.

26. **Git helps** by restoring deleted code using commit history or branches.

27. **Non-functional** — scalability describes how well the system performs under load, not what it does.

28. **Relevant non-functional requirements**: Security, privacy, confidentiality, and compliance.

29. **Performance testing** should be done during **system testing** before deployment.

30. **Tools**: Git, Slack, Jira, Zoom, Google Docs.  
  **Practices**: Daily standups, clear documentation, version control, time zone planning.