System Design Principles

In your own words, describe:

1. The two steps in the Software Design Process:
    a. Architectural design
    b. Detailed design
2. System Design Principles:
    i. Abstraction
    ii. Coupling and cohesion
    iii. Decomposition and modularization
    iv. Encapsulation and information hiding
    v. Separation of interface and implementation
    vi. Completeness and sufficiency
    vii. Separation of concerns
    viii. Uniformity
    ix. Verifiability

1. .
    1. Architectural design: Starts with the big picture of the system and lays out basic processes. It then breaks down into smaller segments until it can be easily implemented.
    2. Detailed design: The smaller level of architectural design(the ones that can be implemented). Focuses on specific components needed for each buildable level of software. Defines interface, algorithms, data structures, and communication
2. System Design Principles
    1. Abstraction: Larger idea or simplification of a complex system. Does not show what needs to be implemented exactly.
    2. Coupling and cohesion: How much stuff inside of a class is calling methods from other classes is coupled. But Cohesion deals with the relation within a class or module. Aim to reduce coupling and have high cohesion.
    3. Decomposition and modularization: Process of breaking down the higher level concept into parts that can be easily deployed. Reusable parts

4. **Encapsulation and information hiding**: Keeps the implementation details of the system hidden, but you know its working
5. **Separation of interface and implementation**: The information and implementation is hidden from the interface or fronted, related to the principle above.
6. **Completeness and sufficiency**: Ensures everything that is made is sufficient and delivers a working product and there is nothing unnecessary or complicated that doesn't need to be there. Design what you need now, not what you think you might need.
7. **Separation of concerns**: View the system from each user's view, design the system according to each view. Separate these views to work on one at a time.
8. **Uniformity**: Want everything to work the same way across the system. Everyone uses the same style of comments and naming conventions of variables and similar syntax
9. **Verifiability**: Testability, must be easily tested and have test cases for the design. Make sure when you design the sytstem you test along the way. Very important for security