**ANSWERS FOR SAJEEL:**

**🔹 What are Requirement Patterns?**

Requirement patterns are **reusable templates** that describe common system needs in a structured way.  
They help software engineers avoid repeating the same analysis for similar problems.

**1. Communication Connectivity:** This pattern describes the device's ability to connect to external networks.

**Description:** A modern phone must support multiple wireless communication standards to enable data and voice services, such as cellular (e.g., 5G), Wi-Fi, and Bluetooth. This includes requirements for seamless switching between networks, handling data transfer, and maintaining a stable connection.

**Application to other devices:** This pattern is directly applicable to smartwatches or tablets, which also need to connect to the internet and other devices for their core functions.

**Example:** A smart speaker, like the Google Home or Amazon Echo, would use a version of this pattern, but focused specifically on Wi-Fi and its own cellular backup (if applicable) to stream audio and communicate with cloud-based services.

**2. Power Management and Efficiency:** This pattern addresses the device's power usage and battery life.

**Description:** A mobile phone must have an efficient power management system to maximize battery life while providing high performance. This involves optimizing power consumption for different tasks, from running background processes to high-intensity use like gaming, and supporting efficient charging mechanisms.

**Application to other devices:** All battery-powered or connected devices, including smartwatches, wireless headphones, and connected home sensors, have this requirement, though the specific goals will vary.

**Example:** A smart doorbell with a battery would require this pattern, with requirements to conserve energy and maximize battery life, potentially by only waking up and activating the camera when motion is detected.

**3.** **Software and Application Ecosystem: This pattern defines how the device supports and manages applications.**

**Description:** It covers the operating system, app installation and updates, compatibility with third-party apps, and integration with cloud services. It ensures users can expand device functionality through reliable software.

**Application to other devices:** This pattern applies to smart TVs, tablets, and wearables that also rely on dedicated software platforms and app ecosystems.

**Example:** A smartwatch running WearOS follows this pattern to manage apps, system updates, and connectivity with paired smartphones.

ANSWERS FOR NEHA

**🔹 What are Requirement Patterns?**

Requirement patterns are **reusable templates** that describe common system needs in a structured way.  
They help software engineers avoid repeating the same analysis for similar problems.

**1. User Interface and Interaction: This pattern covers how users interact with the device.**

**Description:** It includes requirements for the display (e.g., high-resolution touchscreen), input methods (touch, voice, physical buttons), and the overall user experience. The interface must be intuitive, responsive, and adaptable to various applications.

**Application to other devices:** This pattern is also relevant for tablets, e-readers, and smartwatches, each having its own unique interface requirements based on its size and primary functions.

**Example:** An electric toothbrush with a small built-in screen could use a very simple version of this pattern, with requirements focused on a few physical buttons and a basic visual display to show the user battery level and timer.

**2.** **Security and Privacy: This pattern ensures user data and operations remain safe from unauthorized access.**

**Description:** It includes requirements for encryption, biometric authentication (like fingerprints or facial recognition), secure app permissions, and regular security updates to protect sensitive data.

**Application to other devices:** This pattern is crucial for smartwatches, tablets, and IoT systems, which store and transmit personal data.

**Example:** A smart door lock uses encryption and two-factor authentication to prevent hackers from gaining control remotely.

### 3. ****Artificial Intelligence and Automation:** Enhances the device’s intelligence for smarter operation and decision-making.**

**Description:** This pattern includes features like voice assistants, predictive typing, adaptive brightness, and automated system optimization. It enables the device to learn user behavior and act intelligently.

**Application to other devices:** Applies to smart TVs, smart speakers, and even home appliances that rely on AI for better user experience.

**Example:** A smart speaker using voice recognition to adjust volume or suggest music applies this pattern.