Mobile Application Development

Unit 1: Introduction to Mobile Application Development

BSIT – Detailed Lecture Notes

1.1 What is Mobile Application Development?

- **Mobile Application Development** is the process of designing, developing, testing, deploying, and maintaining software applications for mobile and portable devices such as smartphones and tablets.
- A mobile application (mobile app) is software designed to run on a mobile device. Examples include WhatsApp, Google Maps, banking, e-commerce, education and entertainment applications.
- **Simple definition:** Mobile Application Development is the process of creating software specifically designed to operate on mobile devices and their operating systems.

1.2 Why Do We Need Mobile Applications?

- Accessibility: users can access services from almost anywhere.
- Portability: mobile devices are easy to carry.
- Personalization: apps can adapt content and settings to users.
- Connectivity: apps can use Wi■Fi, cellular networks, Bluetooth and NFC.
- Device capabilities: apps can use cameras, microphones, GPS and sensors where permitted.
- **Example – University App:** A university app may provide login, courses, attendance, results, announcements and notifications.

1.3 Characteristics of Mobile Applications?

- Small and variable screens; interfaces must adapt to different sizes.
- Touch-based interaction: tap, swipe, drag, long press and pinch.
- High mobility: the user and device can change location.
- Wireless connectivity: Wi■Fi and cellular networks are common.
- Battery-powered operation requires efficient resource use.
- Memory, storage, processing and network quality can vary.
- Mobile hardware may include GPS, camera, microphone and motion sensors.

1.4 Mobile Application Development Platforms

- **Android:** Major mobile platform. Modern development commonly uses Kotlin/Java, Android SDK and Android Studio.
- **iOS:** Apple's mobile platform. Modern development commonly uses Swift/Objective■C, iOS SDK and Xcode.
- **Windows Mobile/Windows Phone:** Historically important Microsoft mobile platforms; no longer a major current smartphone platform.

1.5 Native Mobile Applications

- A **native application** is developed specifically for a particular operating system.
- Android applications can use Kotlin/Java and Android APIs; iOS applications can use Swift/Objective■C and Apple's frameworks.
- Advantages: strong platform integration, native capabilities and potentially excellent performance.
- Disadvantage: separate implementations for different platforms can increase development and maintenance effort.

1.6 Cross-Platform Applications

- Cross-platform development uses technologies that allow applications to target multiple platforms using shared code or a shared development approach.
- Examples: Flutter, React Native and .NET MAUI.
- Advantages: code reuse, faster multi-platform delivery and potentially lower maintenance cost.
- Limitation: platform-specific features may still require additional work.

1.7 Native vs Cross-Platform – Example

- **Native:** A company develops an Android version with Kotlin and an iOS version with Swift.
- **Cross-platform:** A company uses a framework such as Flutter to share much of the application code across platforms.
- The best choice depends on performance requirements, platform-specific features, budget, team skills and maintenance needs.

1.8 Mobile vs Desktop Applications

- Mobile applications usually have smaller/variable screens, touch input, battery operation, wireless connectivity and access to sensors.
- Desktop applications generally have larger screens, keyboard/mouse input and more stable power and network environments.
- **Example:** A mobile banking app can use the phone camera and biometric authentication directly, while a desktop app may not have the same hardware access.

1.9 Mobile Programming Languages

- Android: Kotlin and Java.
- iOS: Swift and Objective■C.
- Cross-platform: Dart (Flutter), JavaScript/TypeScript (React Native), C# (.NET MAUI).
- **Important:** Android is a platform; Kotlin is a programming language. A platform and a programming language are different concepts.

1.10 Mobile Platform Constraints

- **Battery:** unnecessary GPS, CPU and network activity can drain power.
- **Memory:** excessive memory usage can cause poor performance or crashes.
- **Processing:** expensive computations should be optimized.
- **Screen:** applications should adapt to different sizes and orientations.
- **Network:** connectivity may be slow, unavailable or change during use.
- **Storage:** applications should manage device space efficiently.
- **Security:** personal, authentication and financial data must be protected.

1.11 Challenges with Mobility and Wireless Communication

- **Network disconnection:** a user may enter an area with weak or no coverage. The app should handle failed requests gracefully.
- **Variable bandwidth:** network speed can change between Wi■Fi and cellular connections.
- **Latency:** delay in communication can make remote operations feel slow.
- **Network switching:** devices may move between Wi■Fi and cellular networks.
- **Wireless security:** sensitive information should be protected using appropriate secure communication.

1.12 Mobile Application Performance

- Performance means how efficiently and responsively an application performs.
- Good applications start quickly, respond to user input, manage memory efficiently, minimize unnecessary network requests and avoid freezes/crashes.
- Developers should optimize expensive processing and background operations.

1.13 Performance/Power Trade-Off

- The **performance/power trade-off** means balancing responsiveness or accuracy against battery consumption.
- **GPS example:** requesting location every second may provide frequent updates but consume more power. Requesting it less frequently may save battery but provide less frequent updates.
- The developer should select a strategy appropriate to the application's requirements.

1.14 Mobile Application Development Lifecycle

- **Requirements:** identify users and required functions.
- **Design:** plan UI, navigation, architecture and data.
- **Development:** implement the application.

- **Testing:** verify functionality, performance, security and compatibility.
- **Deployment:** release the application to users.
- **Maintenance:** fix bugs, improve features and issue updates.
- **Lifecycle:** Requirements → Design → Development → Testing → Deployment → Maintenance.

1.15 Types of Mobile Applications

- Educational: learning, quizzes and classrooms.
- Business: inventory, CRM and employee management.
- Entertainment: games, music and video.
- Communication: messaging and video calling.
- E-commerce: shopping, payments and delivery.
- Location-based: maps, ride-hailing and nearby services.

1.16 Location-Aware Applications

- A **location-aware application** uses geographical location information to provide location-related functionality.
- Technologies may include GPS, Wi■Fi positioning and cellular information.
- Examples: navigation, ride-hailing, delivery tracking, local weather and nearby-store applications.
- Location features require attention to permissions, privacy and battery consumption.

1.17 Emerging Technologies

- **Artificial Intelligence:** chatbots, recommendations, image and voice recognition.
- **Internet of Things:** smartphones communicate with smart devices.
- **Augmented Reality:** digital information is overlaid on the real world.
- **Cloud Computing:** remote storage, processing, authentication and synchronization.
- **5G:** high-speed mobile connectivity and potentially lower latency.

1.18 Basic Mobile Application Architecture

- A useful introductory model is: **User Interface** → **Application Logic** → **Data/Services**.
- **User Interface:** screens, buttons, text, images and input controls.
- **Application Logic:** business rules and processing.
- **Data/Services:** local files/databases, APIs, servers and web services.
- **Banking example:** UI collects login data → application logic validates it → app communicates with a banking service → result is displayed.

1.19 Key Terms

- **Mobile Application:** software designed for a mobile device.
- **Mobile Platform:** operating system and associated software environment.
- **Native Application:** application developed for a particular platform.
- **Cross-Platform Application:** application designed to target multiple platforms through a shared development approach.
- **SDK:** Software Development Kit containing tools, libraries, APIs and resources.
- **API:** Application Programming Interface used for software-to-software communication.
- **UI:** User Interface.
- **GPS:** Global Positioning System for geographic positioning.
- **Latency:** communication delay.

1.20 Case Study – Food Delivery App

- UI: restaurant list, product details, cart and checkout.
- Location: customer address and delivery tracking.
- Networking: menus, orders and status updates.
- Notifications: order confirmation and delivery updates.
- Power management: avoid unnecessary continuous location tracking.
- Security: protect account and payment information.

Important Short Questions

- 1. What is Mobile Application Development?
- 2. Define a mobile application.
- 3. What is a mobile platform?
- 4. What is a native application?
- 5. What is cross-platform development?
- 6. What is an SDK?
- 7. What is an API?
- 8. What is latency?
- 9. What is a location-aware application?
- 10. What are mobile platform constraints?
- 11. Why is battery consumption important?

- 12. What is the performance/power trade-off?
- 13. Name three mobile programming languages.
- 14. What are emerging technologies in mobile development?

Important Long Questions

- 1. Explain Mobile Application Development and discuss its importance.
- 2. Discuss the major characteristics of mobile applications.
- 3. Explain the major constraints faced by mobile application developers.
- 4. Discuss challenges associated with mobility and wireless communication.
- 5. Explain the performance/power trade-off with suitable examples.
- 6. Compare native and cross-platform mobile application development.
- 7. Explain major mobile application development platforms.
- 8. Explain the mobile application development lifecycle.
- 9. What are location-aware applications? Explain with examples.
- 10. Discuss emerging technologies in mobile application development.

Unit 1 – Quick Revision

- Mobile apps provide portability, connectivity, personalization and access to device capabilities.
- Major platforms: Android, iOS and historically Windows Mobile/Windows Phone.
- Native apps target a particular platform; cross-platform approaches target multiple platforms.
- Major constraints: battery, memory, processing, screen, network, storage and security.
- Wireless challenges: disconnection, bandwidth variation, latency, switching and security.
- Performance must be balanced against power consumption.
- Lifecycle: Requirements $\rightarrow$ Design $\rightarrow$ Development $\rightarrow$ Testing $\rightarrow$ Deployment $\rightarrow$ Maintenance.
- Location-aware apps use geographical information for context-aware services.
- Emerging areas: AI, IoT, AR, cloud computing and 5G.

Reference Material

- Reto Meier, *Professional Android Application Development*, Wrox Programmer to Programmer, 2015.
- Conway, J., Hillegass, A., & Keur, C., *iOS Programming: The Big Nerd Ranch Guide*, 5th Edition, 2014.
- Phillips, B. & Hardy, B., *Android Programming: The Big Nerd Ranch Guide*, 2nd Edition, 2014.