A Standardized Folder Structure for Mobile Application Projects

Abstract

In mobile application development, long-term maintainability problems often arise not from code quality but from poor project organization. This paper proposes a simple, scalable, and platform-agnostic folder structure that can be applied to most mobile application projects. The structure separates responsibilities clearly between development, design, marketing, backup management, and documentation, reducing operational friction and improving team efficiency.

1 Introduction

Mobile application projects evolve continuously through multiple versions, design iterations, and marketing cycles. Without a clear and enforced folder structure, projects tend to accumulate technical and organizational debt. This results in slower onboarding, version confusion, and increased risk during maintenance.

This paper introduces a minimal yet sufficient directory hierarchy intended to be used as a default structure for mobile application projects.

2 Proposed Root Directory Structure

The proposed structure consists of five top-level directories:

```
/Backups
/Code
/Design
/Marketing
/Other
```

Each directory has a single responsibility. Overlapping usage is explicitly avoided.

3 Backups Directory

3.1 Purpose

The `Backups` directory stores immutable snapshots of previous project versions. It is not part of the active development workflow.

3.2 Structure

```
/Backups
  /v1.0
    /Code
  /v1.1
    /Code
```

3.3 Rules

- Each version has its own directory.
- Only source code is stored inside backup versions.
- No design, marketing, or documentation files are included.
- Backup directories are read-only by convention.

4 Code Directory

4.1 Purpose

The `Code` directory contains only the active source code of the mobile application.

4.2 Rules

- No design assets, documents, or marketing materials are allowed.
- The directory reflects the platform structure (e.g., iOS, Android, Flutter).
- This is the only directory directly connected to build systems and CI pipelines.

5 Design Directory

5.1 Purpose

The `Design` directory stores all visual and UX-related materials used in the project.

5.2 Contents

- UI/UX design files (Figma, Sketch, Adobe XD)
- Icons and illustrations
- Brand guidelines
- Exported design assets

6 Marketing Directory

6.1 Purpose

The `Marketing` directory contains all materials related to user acquisition and promotion.

6.2 Contents

- Advertisement videos and images
- Store listing texts and screenshots
- Campaign performance metrics
- A/B testing results

7 Other Directory

7.1 Purpose

The `Other` directory acts as a controlled container for project-related materials that do not fit into the previous categories.

7.2 Contents

- Technical documentation
- Legal and compliance documents
- Roadmaps and planning files
- Meeting notes

8 Discussion

The strength of this structure lies in its strict separation of concerns. By preventing mixed usage of directories, the project remains understandable and maintainable even as it scales.

This model intentionally avoids deep nesting and unnecessary abstractions, favoring clarity over flexibility.

9 Conclusion

A disciplined folder structure is a low-cost, high-impact decision in mobile application development. The proposed hierarchy is simple enough to be adopted immediately, yet robust enough to support long-term growth. It can be used as a baseline standard for solo developers and teams alike.