askme: Tech Stack Implementation
Optimized KMP Development Environment with 4-Tier Storage

📋 Overview
- Complete Kotlin Multiplatform Mobile (KMP) development environment
- 4-Tier cloud-synchronized storage system for offline/online development
- 4+ API-based LLM provider integration with fallback system
- Cross-platform development capability (Android focus)
- Enterprise-grade security with credential protection
- Storage Required: 64GB USB + 44GB cloud

✅ **PHASE 1: Environment & Prerequisites Verification**

🎯 **Checkpoint 1: Hardware & Account Verification**
[x] 1.1 64GB+ USB drive available and mounted
[x] 1.2 Chromebook with Crostini Linux enabled and functional
[x] 1.3 Stable internet connection (test with `ping -c 3 google.com`)
[x] 1.4 4+ hours dedicated time allocated
[x] VALIDATION: All hardware requirements met
🎯 **Checkpoint 2: Cloud Storage Accounts**
[x] 2.1 Google Drive account (15GB free tier)
[x] 2.2 Box.com account (10GB free tier)
[x] 2.3 Mega.nz account (20GB free tier)
[x] 2.4 GitHub account for repository hosting
[x] VALIDATION: All cloud accounts accessible
🎯 **Checkpoint 3: USB Drive Structure Setup**
[x] 3.1 Mount USB drive: Settings → Linux → USB devices
[x] 3.2 Set USB path: `export USB_PATH="askme"`
[x] 3.3 Create directories: `mkdir -p $USB_PATH/{src,tools,docs,backups,logs}`
[x] 3.4 Create tools subdirs: `mkdir -p $USB_PATH/tools/{jdk17,android-studio,android-sdk}`
[x] 3.5 Make permanent: `echo 'export USB_PATH="askme"' >> ~/.bashrc`
[x] VALIDATION: `ls -la $USB_PATH` shows all directories
[x] SUCCESS CRITERIA: USB structure ready, 50GB+ available space
⚠️ **ROLLBACK 1:** If USB issues occur, unmount/remount via Chrome OS settings, verify path with `df -h`

✅ **PHASE 2: Core Development Tools**

🎯 **Checkpoint 4: JDK 17 Installation**
[x] 4.1 Update system: `sudo apt update && sudo apt upgrade -y`
[x] 4.2 Install OpenJDK 17: `sudo apt install openjdk-17-jdk`

[x] 4.3 Set JAVA_HOME: `echo 'export JAVA_HOME="/usr/lib/jvm/java-17-openjdk-amd64"' >> ~/.bashrc`
[x] 4.4 Reload environment: `source ~/.bashrc`
[x] VALIDATION: `java -version` shows 17.0.10+ and `echo $JAVA_HOME` shows correct path
[x] EXPECTED OUTPUT: `openjdk version "17.0.XX"`
🎯 **Checkpoint 5: Kotlin 1.9.10 via SDKMAN**
[x] 5.1 Install SDKMAN: `curl -s "https://get.sdkman.io" | bash`
[x] 5.2 Initialize: `source ~/.sdkman/bin/sdkman-init.sh`
[x] 5.3 Install Kotlin: `sdk install kotlin 1.9.10`
[x] 5.4 Set default: `sdk default kotlin 1.9.10`
[x] VALIDATION: `kotlin -version` shows exactly "1.9.10-release-459"
[x] BENCHMARK: SDKMAN installation < 5 minutes
🎯 **Checkpoint 6: Gradle 8.4 Installation**
[x] 6.1 Install Gradle: `sdk install gradle 8.4`
[x] 6.2 Set default: `sdk default gradle 8.4`
[x] VALIDATION: `gradle --version` shows Gradle 8.4 + Kotlin 1.9.10
[x] SUCCESS CRITERIA: Perfect version alignment confirmed
⚠️ **ROLLBACK 2:** If version conflicts occur, use `sdk uninstall kotlin 1.9.10` and `sdk uninstall gradle 8.4`, then reinstall
---

✅ **PHASE 3: Android Development Environment**

🎯 **Checkpoint 7: Android SDK Configuration**
[x] 7.1 Set Android SDK path: `export ANDROID_HOME="$USB_PATH/tools/android-sdk"`
[x] 7.2 Add to PATH: `export PATH="$PATH:$ANDROID_HOME/tools:$ANDROID_HOME/platform-tools"`
[x] 7.3 Make permanent: `echo 'export ANDROID_HOME="$USB_PATH/tools/android-sdk"' >> ~/.bashrc`
[x] 7.4 Download command line tools to `$ANDROID_HOME/tools/`
[x] VALIDATION: `echo $ANDROID_HOME` shows USB path
🎯 **Checkpoint 8: SDK Components Installation**
[x] 8.1 Install Platform 34: `$ANDROID_HOME/tools/bin/sdkmanager "platforms;android-34"`
[x] 8.2 Install Build Tools: `$ANDROID_HOME/tools/bin/sdkmanager "build-tools;34.0.0"`
[x] 8.3 Install Platform Tools: `$ANDROID_HOME/tools/bin/sdkmanager "platform-tools"`
[x] VALIDATION: `$ANDROID_HOME/tools/bin/sdkmanager --list | grep "platforms;android-34"`
[x] BENCHMARK: SDK installation < 15 minutes
[x] SUCCESS CRITERIA: All Android SDK components functional
⚠️ **ROLLBACK 3:** If SDK manager fails, delete `$ANDROID_HOME` directory and re-download command line tools

---
✅ **PHASE 4: KMP Project Initialization**

🎯 **Checkpoint 9: Project Directory Setup**
[x] 9.1 Navigate to source: `cd $USB_PATH/src`
[x] 9.2 Create project: `mkdir -p askme && cd askme`
[x] 9.3 Create structure: `mkdir -p gradle src/{commonMain,androidMain}/kotlin`
[x] VALIDATION: `pwd` shows correct project path
🎯 **Checkpoint 10: Gradle Configuration Files**
[x] 10.1 Create `gradle/libs.versions.toml` with Kotlin 1.9.10, Gradle 8.4
[x] 10.2 Create `settings.gradle.kts` with project configuration
[x] 10.3 Create `build.gradle.kts` with KMP + Android setup
[x] 10.4 Create `gradle.properties` with Chromebook optimizations:
```
android.useAndroidX=true
org.gradle.jvmargs=-Xmx1024m
org.gradle.daemon=false
```
[x] VALIDATION: All configuration files created
🎯 **Checkpoint 11: Initial Build Test**
[x] 11.1 Initialize Gradle wrapper: `gradle wrapper --gradle-version 8.4`
[x] 11.2 Test version alignment: `./gradlew --version`
[x] 11.3 Create minimal AndroidManifest.xml and MainActivity.kt
[x] 11.4 Execute test build: `./gradlew compileKotlinMetadata`
[x] VALIDATION: Build successful with "BUILD SUCCESSFUL" message
[x] BENCHMARK: Initial build < 10 minutes
[x] SUCCESS CRITERIA: KMP project structure functional
⚠️ **ROLLBACK 4**: If build fails, check `gradle.properties` settings, verify all files in correct KMP structure (`src/commonMain/kotlin`)

✅ **PHASE 5: Version Control & Cloud Setup**

🎯 **Checkpoint 12: Git Configuration**
[x] 12.1 Install Git: `sudo apt install git`
[x] 12.2 Configure user: `git config --global user.name "your-username"`
[x] 12.3 Configure email: `git config --global user.email "your-email@example.com"`
[x] 12.4 Set default branch: `git config --global init.defaultBranch main`
[x] 12.5 Initialize repository: `git init`
[x] VALIDATION: `git config --list | grep user` shows correct configuration
🎯 **Checkpoint 13: rclone Cloud Storage Setup**
[x] 13.1 Install rclone: `curl https://rclone.org/install.sh | sudo bash`
[x] 13.2 Configure Google Drive remote: `rclone config` (name: `askme`)
[x] 13.3 Configure Box.com remote: `rclone config` (name: `askme-box`)
[x] 13.4 Configure Mega.nz remote: `rclone config` (name: `askme-mega`)
[x] VALIDATION: `rclone listremotes` shows all three remotes
[x] SUCCESS CRITERIA: 4-tier storage architecture ready

⚠️ **ROLLBACK 5**: If rclone config fails, use `rclone config delete <remote-name>` and reconfigure
---

✅ PHASE 6: API Provider Integration (4 Required)

🎯 Checkpoint 14: HTTP Dependencies
[x] 14.1 Add OkHttp to `build.gradle.kts`: `implementation("com.squareup.okhttp3:okhttp:4.12.0")`
[x] 14.2 Add JSON serialization: `implementation("org.jetbrains.kotlinx:kotlinx-serialization-json:1.6.0")`
[x] 14.3 Add coroutines: `implementation("org.jetbrains.kotlinx:kotlinx-coroutines-android:1.7.3")`
[x] 14.4 Sync dependencies: `./gradlew build`
[x] VALIDATION: Build successful with all HTTP dependencies
🎯 Checkpoint 15: API Client Structure
[x] 15.1 Create API package: `mkdir -p src/commonMain/kotlin/com/askme/api`
[x] 15.2 Create ClaudeClient.kt for Anthropic Claude 4 integration
[x] 15.3 Create OpenAIClient.kt for GPT-4 integration
[x] 15.4 Create GeminiClient.kt for Google Gemini integration
[x] 15.5 Create MistralClient.kt for Mistral AI integration
[x] 15.6 Create ApiManager.kt with fallback logic (Claude → OpenAI → Gemini → Mistral)
[x] VALIDATION: All 4 API clients created in proper KMP structure
🎯 Checkpoint 16: API Integration Test
[x] 16.1 Test compilation: `./gradlew compileKotlinMetadata`
[x] 16.2 Verify fallback system logic in ApiManager
[x] VALIDATION: 4-provider API system compiles successfully
[x] BENCHMARK: Build with API clients < 5 minutes
[x] SUCCESS CRITERIA: Complete LLM provider integration ready
⚠️ **ROLLBACK 6**: If API integration fails, check KMP directory structure, ensure all files in `src/commonMain/kotlin`

✅ PHASE 7: Security & Sync Implementation

🎯 Checkpoint 17: Security Configuration
[x] 17.1 master_sync.sh - 4-tier orchestrated synchronization with enhanced security filters
[x] 17.2 Copy sync script to USB: `cp master_sync.sh $USB_PATH/`
[x] 17.3 Make executable: `chmod +x $USB_PATH/master_sync.sh`
[x] 17.4 Remove sensitive files: `find $USB_PATH -name "local.properties" -delete`
[x] VALIDATION: No API keys found with `find $USB_PATH -name ".env" -o -name "_api_key"`
🎯 Checkpoint 18: Initial Sync Execution
[x] 18.1 Test dry run: `$USB_PATH/master_sync.sh` → option 7 (Test Sync)
[x] 18.2 Execute full backup: `$USB_PATH/master_sync.sh` → option 4

[x] 18.3 Monitor progress: `tail -f $USB_PATH/tiered_sync.log`
[x] VALIDATION: All 4 tiers populated, security filters active
[x] BENCHMARK: Full sync < 30 minutes depending on connection
[x] SUCCESS CRITERIA: 4-tier storage operational with zero sensitive files synced
⚠️ **ROLLBACK 7**: If sync fails, check rclone remotes with `rclone listremotes`, verify cloud storage quotas
---

✅ PHASE 8: Environment Optimization

🎯 Checkpoint 19: Environment Variables & Aliases
[x] 19.1 Set environment: `echo 'export askme_ENV="chromebook"' >> ~/.bashrc`
[x] 19.2 Create sync alias: `echo 'alias sync-tiers="$USB_PATH/master_sync.sh"' >> ~/.bashrc`
[x] 19.3 Create dev alias: `echo 'alias askme-dev="cd $USB_PATH/src/askme"' >> ~/.bashrc`
[x] 19.4 Reload: `source ~/.bashrc`
[x] VALIDATION: `echo $askme_ENV` shows "chromebook"
🎯 Checkpoint 20: Local Development Workspace
[x] 20.1 Create local workspace: `mkdir -p ~/askme-dev`
[x] 20.2 Create symlinks: `ln -s $USB_PATH/src/askme ~/askme-dev/`
[x] 20.3 Create development alias: `echo 'alias askme-dev="cd ~/askme-dev/askme"' >> ~/.bashrc`
[x] 20.4 VALIDATION: `askme-dev && pwd` shows correct path

✅ PHASE 9: Quality Tools & Final Validation

🎯 Checkpoint 21: Quality Tools Integration
[x] 21.1 Add Detekt plugin to `build.gradle.kts`
[x] 21.2 Add ktlint plugin to `build.gradle.kts`
[x] 21.3 Generate Detekt config: `./gradlew detektGenerateConfig`
[x] 21.4 Run quality checks: `./gradlew detekt ktlintCheck`
[x] 21.5 VALIDATION: Quality tools run without errors
🎯 Checkpoint 22: Final System Validation
[x] 22.1 Full project build: `./gradlew build`
[x] 22.2 Dependency validation: `./gradlew dependencies`
[x] 24.3 Version alignment check: All tools show correct versions
[x] 24.4 Sync architecture test: `sync-tiers` shows 4-tier menu
[x] 24.5 VALIDATION: Complete system functional

✅ FINAL VALIDATION CHECKLIST

Core Versions
[x] V.1 JDK version: `java -version` shows 17.0.10+

[x] V.2 Kotlin version: `kotlin -version` shows 1.9.10-release-459
[x] V.3 Gradle version: `gradle --version` shows 8.4 + Kotlin 1.9.10
[x] V.4 Android SDK: Platform 34 functional

Project Build
[x] V.5 KMP compilation: `./gradlew compileKotlinMetadata` succeeds
[x] V.6 Full build: `./gradlew build` succeeds
[x] V.7 Dependencies resolve correctly
[x] BENCHMARK: Full build < 6 minutes

API Integration (4 Providers Required)
[x] V.8 Claude 4 client functional
[x] V.9 OpenAI GPT-4 client functional
[x] V.10 Google Gemini client functional
[x] V.11 Mistral AI client functional
[x] V.12 Fallback system operational

Storage Architecture (4-Tier)
[x] V.13 TIER 1 (USB): Full environment < 50GB
[x] V.14 TIER 2 (Google Drive): Remote functional, < 14GB
[x] V.15 TIER 3 (Box.com): Remote functional, < 10GB
[x] V.16 TIER 4 (Mega.nz): Remote functional, < 20GB

Security Validation
[x] V.17 No API keys in cloud storage
[x] V.18 Security filters active and comprehensive
[x] V.19 Sync operations secure and filtered9.10-release-459
[x] V.3 Gradle version: `gradle --version` shows 8.4 + Kotlin 1.9.10
[x] V.4 Android SDK: Platform 34 functional