# **📄 1. Dev Onboarding Guide**

## **✅ Prerequisites**

| **Tool** | **Version / Notes** |
| --- | --- |
| Flutter SDK | >= 3.13.x (use flutter doctor) |
| Dart | Comes with Flutter |
| Node.js | >= 18.x (for tooling/scripts) |
| Firebase CLI | npm install -g firebase-tools |
| Supabase CLI | brew install supabase/tap/supabase |
| Python (Optional) | >= 3.8 (for LLM test scripts) |
| Git | >= 2.30 |

## **🚀 Setup Instructions**

# 1. Clone the repo

$ git clone https://github.com/your-org/defeah-app.git && cd defeah-app

# 2. Install Flutter dependencies

$ flutter pub get

# 3. Set up environment

$ cp .env.example .env.dev # fill with your API keys

# 4. Run Firebase auth/emulator locally (if needed)

$ firebase emulators:start

# 5. Launch app on device/emulator

$ flutter run

## **🧪 Running Tests**

# Run unit tests

$ flutter test

# Integration tests (if configured)

$ flutter drive --target=test\_driver/app.dart

## **✅ Linting & Formatting**

flutter format .

flutter analyze

Uses default Dart analyzer + flutter\_lints package. Warnings must be resolved before commit.

## **🎯 First Task for New Devs**

* Edit the prompt template in lib/services/llm\_prompt.dart
* Change wording of AI output response hint on the StudyGuide screen
* Submit a PR titled fix/prompt-tweak-llm-ui

## **⚠️ Common Pitfalls**

* Forgetting to add .env.dev
* Android Studio misconfiguring Flutter location (check flutter doctor)
* Permissions issue when accessing Firebase emulators

# **📄 2. Git Workflow Guide**

## **🔀 Branching Strategy**

| **Branch** | **Purpose** |
| --- | --- |
| main | Production-ready code |
| dev | Integrate tested features |
| feature/\* | New features (PRs to dev) |
| fix/\* | Bug fixes |
| hotfix/\* | Critical issues on main |

## **📝 Naming Conventions**

* feature/study-ui
* fix/verse-load-error
* hotfix/broken-build

## **✅ PR Guidelines**

* Title: use clear and conventional commit prefix (e.g., feat, fix, docs, refactor)
* Checklist before merging:  
  + All tests pass
  + No console.log/debug prints
  + PR description explains the change

## **⬇️ Commit Format (Recommended)**

feat(dailyverse): add multi-language support

fix(auth): resolve token refresh issue

## **✅ Merge Policy**

* Feature → dev: Needs 1 code review + CI pass
* dev → main: Tag with release version + full QA pass
* Squash commits before merge

## **🔥 Hotfixes**

* Branch from main → fix → PR into main and dev

## **🏷 GitHub Labels (Optional)**

* needs-review, ready-for-merge, blocked, high-priority

# **📄 3. CI/CD Pipeline Configuration**

## **🛠 CI Tasks**

| **Task** | **Triggered On** |
| --- | --- |
| Lint + Test | PR to dev, main |
| Build APK/Web | On PR to main |
| Deploy Supabase Edge | On merge to main |
| Notify Slack | On failed build |

## **✅ GitHub Actions (sample)**

.github/workflows/flutter.yml

name: Flutter CI

on:

pull\_request:

branches: ["main", "dev"]

jobs:

build:

runs-on: ubuntu-latest

steps:

- uses: actions/checkout@v3

- uses: subosito/flutter-action@v2

with:

flutter-version: '3.13.0'

- run: flutter pub get

- run: flutter analyze

- run: flutter test

## **✅ Deploy Backend (Supabase/Firebase)**

# Firebase

firebase deploy --only functions

# Supabase Edge

supabase functions deploy generateGuide

## **📣 Notifications**

* Optional Slack webhook for PRs & failed builds
* GitHub email alerts on failed deployments

# **📄 4. Environment Configuration**

## **🔐 Where to Store Secrets**

| **Platform** | **Secret Location** |
| --- | --- |
| Flutter (Client) | .env.dev (via flutter\_dotenv) |
| GitHub Actions | GitHub Secrets UI |
| Firebase | Firebase Console > Config |
| Supabase Edge | supabase secrets set |

## **🔑 Required Environment Variables**

| **Variable** | **Used In** | **Notes** |
| --- | --- | --- |
| OPENAI\_API\_KEY | Backend | LLM access |
| FIREBASE\_PROJECT\_ID | Frontend | Auth + Storage |
| SUPABASE\_URL | Frontend | Guide Storage/Fetch |
| SUPABASE\_ANON\_KEY | Frontend | Public API access |
| SUPABASE\_SERVICE\_KEY | Backend | Admin-level actions |
| ENV\_MODE | Both | dev, staging, prod |

## **🔄 Switching Between Envs**

# .env.dev → for local Flutter dev

# .env.staging → for testing

# .env.prod → used only in builds/deploy

flutter run --dart-define-from-file=.env.dev

## **🛡 Best Practices**

* Never commit .env.\* files
* Use .env.example with placeholder values
* Rotate API keys every 90 days
* Use .env.mock for unit tests without live API