Jenkins Flutter CI/CD Pipeline – Project Summary and Documentation

1. Project Overview

You built a robust, open-source Jenkins CI/CD pipeline for a Flutter Android application. The system uses a custom Docker image that has **Flutter** and the **Android SDK** pre-installed, with all correct permissions for the Jenkins user. The pipeline:

- Checks out code from GitHub
- Installs Dart & Flutter dependencies
- Runs static analysis and tests
- Builds the release APK
- Archives the APK for download

2. Issues Faced and How They Were Solved

Error Message	Why it Happened	Root Cause	How We Solved It
`couldn't find remote ref`	Jenkins tried to build a `master` branch	Git repo default branch was `main`	Set job to use the `main` branch
`flutter: not found`	Jenkins couldn't run Flutter CLI	Flutter SDK missing or not in PATH	Installed Flutter in Docker & adjusted PATH
`cannot access /opt/android-sdk`	No Android SDK found	SDK not installed or path was wrong	Installed SDK and set ANDROID_HOME
`Permission denied /opt/flutter`	Couldn't write cache in Flutter dirs	Jenkins user didn't own preinstalled Flutter	Added chown to give Jenkins user ownership
`SDK directory is not writable`	Gradle couldn't auto-install NDK/tools	Jenkins user couldn't write to android-sdk	Gave Jenkins user write access to SDK directory
`Invalid version constraint: ^latest`	pubspec.yaml used `^latest` (invalid)	Only real version numbers accepted	Used published version (e.g. `^2.32.0`)
`await only in an async function`	Used `await` in `void main()`	Dart syntax rules	Marked main as `Future<void> main() async`
SMTP/email send fails	Email notifications don't work	SMTP not configured in Jenkins	(Remains to fix) – needs real SMTP config

3. Documentation and Best Practices

Environment Setup

- **Docker image** contains all required SDKs, with `chown` set for Jenkins user.
- **Pipeline environment variables**: Sets `FLUTTER_HOME` and `ANDROID_HOME` and updates `PATH`.
- **All critical dirs writable** by the Jenkins user.

Jenkinsfile Pipeline Flow

1. **Show Tool Versions/Staging**: Early failure if any tool missing
2. **Dependencies**: `flutter pub get`
3. **Code Analysis**: `flutter analyze`
4. **Run Tests**: `flutter test`
5. **Build APK**: `flutter build apk --release`
6. **Archive Artifacts**: Stores APK as download
7. **Email Notification**: (if SMTP configured)

Maintenance Tips

- Always use real package versions in `pubspec.yaml`, not `latest`.
- Run `flutter pub get` after changing dependencies, and commit/push both `pubspec.yaml` and `pubspec.lock`.
- For permission errors, check Dockerfile for correct folder ownership before switching to `USER jenkins`.
- Update Jenkins `PATH` if adding new tools.
- For any build errors: check Jenkins logs, echo tool paths early, ensure effective Dockerfile runs.

How to Troubleshoot

- Tool not found? Echo `$PATH` and run `which` commands in a Jenkins stage.
- Permission denied? Check folder ownership and Dockerfile `chown` step.
- Dependency issues? Ensure all needed Dart/Flutter packages are committed in `pubspec.yaml` and `pubspec.lock`.
- Failing email? Verify SMTP server config in Jenkins.

4. Long-Term Project Memory

- **Keep this summary in your repo's README for fast onboarding and troubleshooting.**
- Extend as you add Firebase/AppDistribution deployment or iOS support.
- Always document new errors and their fixes in a similar table for future reference.

If you want to create a PDF, you can simply copy this Markdown to a file and use an online MD-to-PDF converter or run:

```
from fpdf import FPDF
# ...python export code as shown in your reference
```

But: Exporting a PDF directly requires a custom script or tool not built into Jenkins or basic CI. Simply save this doc as README or process it with pandoc/fpdf as shown above if you want a print/PDF export for presentations or personal study.

⁂