Echos

“One place to learn and translate American Sign Language”

Coding and Integration Standards

Group 11

Prayag Savsani [AU1841035]
Darshan Navadiya [AU1841061]
Kishan Mehta [AU1841072]
Harsh P. Patel [AU1841075]
Dhruv Sharma [AU1841102]
Harin Khakhi [AU1841116]
Nilay Gandhi [AU1841118]
Sanket J Shah [AU1841120]

TABLE OF CONTENTS

1. Coding Standards

1.1 Naming conventions

The following three types of identifiers are to be used:

1. UpperCamelCase names capitalize the first letter of each word, including the first.
2. lowerCamelCase names capitalize the first letter of each word, except the first which is always lowercase, even if it's an acronym.
3. lowercase_with_underscores names use only lowercase letters, even for acronyms, and separate words with _.

For a detailed guide on the naming, do refer to the dart styling guide before starting with the coding.

1.2 Indentation and module separation

1. Indentation
   Flutter provides automatic formatting for the code on various IDEs. Setup the automatic formatting before starting to code.
2. Module separation
   Use the following separation protocols for the code files.
   /screens/myCode.dart : Add all the front-end files here
   /rewidgets/myCode.dart : Add all the reusable components here
   /services/myCode.dart : Add all backend services files here
   /models/myCode.dart : Add all the machine learning model files here
   /helpers/myCode.dart : Add all the helpers file here (camera, tflite etc)
   /utilities/myCode.dart : Add all the API keys here
   /assets : Add all assets like fonts, images here

2. Continuous Integration

2.1 How to get started?

Tool recommended: GitHub Desktop
Steps:

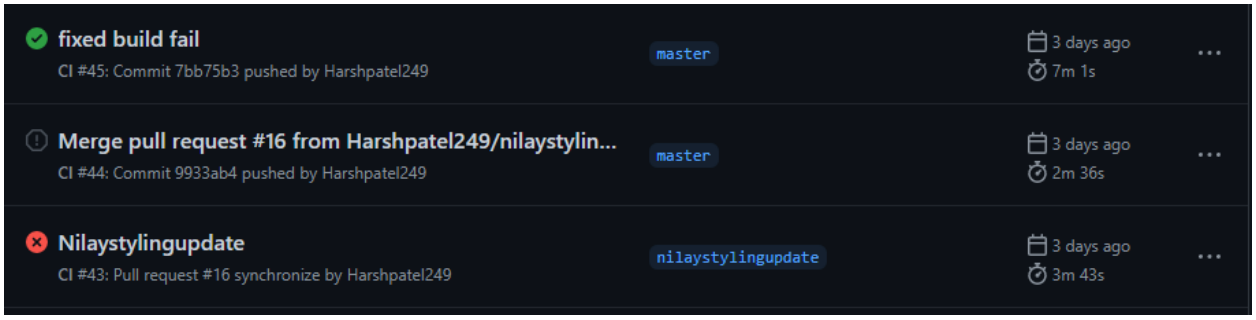
1. Clone the repository: repo
2. Create your own branch with proper naming convention.
3. Commit all your changes on your branch and once you are done with implementing the functionality, create a pull request.
4. Move to the integration process.

2.2 Integration process

The integration process requires the changes to pass the GitHub Actions build test and the following are the steps for the same:
Steps:

1. Create a pull request.
2. Resolve all the conflicts with the master branch and ask a peer to review the solutions so no content is lost.
3. After resolving the conflicts, create a merge request.
4. Wait for the GitHub Actions to test the changes triggered on the creation on merge request.
5. If the build test is passing, merge the changes.
6. If the build test fails, check the logs in the Actions section and resolve the errors and go to step 1 again.

Example screenshot of the Github Action run:



3. References

3.1 Naming conventions and indentations

Official dart styling guide: Link
Official flutter automatic formatting guide: Link

3.2 Continuous integration

Flutter actions provided on the GitHub Actions market: Link