Colibri Mobile Application

iOS and Android

Version 1.1.8

Last updated on June 7, 2022

Table of Contents

Part 1: About Application

Part 2: Files & Folders

Part 3: App Debugging

Part 1: About Application

Application Overview

Colibri Mobile Application for iOS and Android is a mobile app developed and configured for the Colibri Social Media Platform. Functionalities such as Mobile OTP Login, Message system, Push notifications, Search queries and other cool features are embedded into the application. The app comes with clean User Experience Interface and easy to use functionalities.

This application is not a standalone application and must be connected to the Social Media Platform in order for it to function.

Application Package

Colibri Mobile Application has two (2) applications:

1. Colibri Mobile App (Android)
   Colibri app [app package name: com.skycap.colibri] for android users

2. Colibri Mobile App (iOS)
   Colibri app [bundle ID: com.skycap.colibri] for iOS users

Hardware and Software Requirements

Hardware required:

iOS devices (iPhone 4s or newer) and Android devices

Software required:

An android smart phone with operating system Jelly Bean, v16, 4.1.x or newer, and iOS smartphone with iOS 8 or newer. or IOS operating system (OS).

To download and use the functionalities of "Colibri" mobile application, you will require an Internet connection in your mobile phone.

APIs | Technologies Used

Colibri Mobile Application is built with the following technologies:

1. **Google's Flutter 3.0** as app development framework
2. **Dart** for client-optimized language
3. **Firebase Authentication** for Mobile OTP authentication
4. **Firebase Messaging** for sending Push Notifications

App Permissions

Colibri Mobile Application requires the following User Permissions:

1. **Internet Connectivity** to run app
2. **Read Storage** to upload Media files
3. **Written Storage** to save a file on storage
4. **Camera access** to take an upload videos
5. **GIPHY GIF API** for Gif in Posts, Replies, Comments and Messages

Part 2: Files & Folders

2.1. Android Folder [.../colibri/**android**]

This folder contains all technical information about the android part of the app since it is a cross platform package. Hence, editable information like- App package Name, App name, App logo are inside this folder.

There are hundreds of other files are inside the folder that are native android compiled. It is advisable not to alter those files.

2.2. Assets Folder [.../colibri/**assets**]

This folder contains all editable language files for the app.

2.3. Build Folder [.../colibri/**build**]

This is a temporary folder which is generated when we compile and run the app. It contains release apk, app bundle, debug apk. Final app file should be picked from this folder to upload into Playstore.

You can delete this folder anytime and rebuild this folder anytime using command **flutter run** OR **flutter build appbundle** from the terminal keeping Colibri folder open in terminal.

2.4. Fonts Folder [.../colibri/**fonts**]

This folder contains all the Font families and individual font files inside them. You can remove or add fonts in this folder but you have to register the font in [.../colibri/**pubspec.yaml**] file as per section 1.a.1.

2.5. Images Folder [.../colibri/**images**]

This folder contains all .svg icons and images such as logo, chat, search etc.

2.6. iOS Folder [.../colibri/**iOS**]

This is another special folder which contains the iOS configuration & code of the flutter application. You can change the iOS splash screen , name, bundle name etc. from this folder.

2.7. Lib Folder [.../colibri/**lib**]

This is the main folder where you will find Source code of all the Screens. Inside this file there is a **main.dart** file which is the initial file of the app which executes at first when the app is compiled & run.

2.8. Config Folder [.../colibri/**config**]

The config files contains the **API constants**, **App colors** and **Strings** files. Here you can change the App URLs and App colors.

2.9. Metadata Folder [.../colibri/**metadata**]

This folder contains keys to iOS and Android build.

2.10. Pubspec.yaml File [.../colibri/**pubspec.yaml**]

This is the register of all 3rd party Plugins, Fonts, Assets entry. This file is the one of the most crucial file of this package. It is suggested not to modify/ change this file code if you are not 100% sure about the code as this leads app crashing and unable to compile also.

2.11. Extra Files

All these are extra files generated by the system. You do not need to access them or delete them:

[.../colibri/analysis_options.yaml]

[.../colibri/pubspec.lock]

[.../colibri/README.md]

Part 3: App Debugging

Prerequisite

Basic knowledge of flutter. Follow the official flutter get started guide https://flutter.dev/docs/get-started/

System requirement: Android apps can be configured on Windows, macOS or Linux. macOS is mandatory for iOS app configurations.

Basic Knowledge of Android studio or Visual Studio and Xcode.
Firebase

Downloading and installing Flutter

Follow the flutter official docs to install Flutter on your system

https://flutter.dev/docs/get- started/install/

Firebase integration

Create a project on firebase:

Add android and iOS applications with the correct unique package name and bundleID. Follow the steps provided while creating the app. Replace the google-services.json and GoogleService-Info.plist files at the exact locations mentioned on firebase.

Finally, replace all the occurrence of bundle ID and package name inside the android and iOS folder.

Run the app **"flutter run"** to verify the configuration.
To know more – https://firebase.google.com/docs/flutter/setup

For Phone Auth:

Go to the Authentication tab from the Firebase dashboard side nav. Under the Sign-in method Enable the Phone method.

For more info:
android: https://firebase.google.com/docs/auth/android/phone-auth

iOS: https://firebase.google.com/docs/auth/ios/phone-auth

For more details on Android & Firebase integration check here:

https://help.deligence.com/knowledgebase/android-firebase-configuration/

IDE Setup

Install VS Code or Android Studio.

VS Code is a lightweight editor with Flutter app execution and debug support.

https://flutter.dev/docs/get-started/editor?tab=vscode

Android Studio provides the fastest tools for building apps on every type of Android device

https://developer.android.com/studio/intro

Setting up Source Code

Download ZIP File Source Code. Drag & Drop Downloaded Source Code Folder At VS Code Icon to open project OR Select downloaded folder and choose open with VS Code.

1.a. Edit Theme and Details

1.a.1 Change Fontsyle

1. Place your Font File (**.otf** or **.ttf**) in **<path_to_project_root>/lib/fonts**. folder.

2. Register font path at **<path_to_project_root>/pubspec.yaml**. Paste the font name at the bottom font section.

EXAMPLE: -family: FONTNAME

```
fonts:

 - assets: fonts/FONTNAME.ttf
```

3. Save 'pubspec.yaml' file to update new font

4. Open **<path_to_project_root>/lib/main.dart** and inside themeData provide this – font family: 'FONTNAME'

1.a.2 Change Color Values

1. Change Color values in **<path_to_project_root>/lib/core/config/colors.dart** inside you will find all the color values used in the project. Just replace the color value with your desired one.

2. Save the File using CTRL+S.

1.a.3 Change App/Package Name Label [Go to step 6.3]

1. Provide app name in "android:label" field inside **<path_to_project_root>/android/app/src/main/AndroidMainfest.xml**. Also provide your package name in "package" field

2. Provide package name in "package" field inside **<path_to_project_root>/android/app/src/debug/**

3. Provide package name in "package" field inside **<path_to_project_root>/android/app/src/profile/**

4. Provide application id name in "applicationId" field inside **<path_to_project_root>/android/app/build.gradle**

5. Provide package name in "package" field inside **<path_to_project_root>/android/app/src/main/kotlin/com/skycap/app**

1.a.4 Rename Package Directory [Automatic when step 6.3 is done]

1. Go to **<path_to_project_root>/android/app/src/main/kotlin** and rename the folder to your package name.

FOR EXAMPLE: the default **com > skycap > app** will need to be changed to **com > yourpackagename > app**

1.a.5 Change App Icon

1. Go to https://romannurik.github.io/AndroidAssetStudio/icons-launcher.html

2. Upload your logo and click the Download button at the top, right corner.

3. Copy files in download folder and overwrite to **<path_to_project_root>/android/app/src/main/res**. Make sure that the "launcher_icon.png" in each file matches the original image file for Colibri.

1.a.6 Change App and Splash Logos

1. Go to **<path_to_project_root>/android/app/src/main/res/drawable *app_icon.png**

2. Go to **<path_to_project_root>/images**

1.a.7 Edit Splash Screen

1. Go to **<path_to_project_root>/android/app/src/main/**

1.a.8 Change Other Configurations

1. Go to **<path_to_project_root>/lib/core/config/api_constants.dart** to edit **baseUrl** and **baseMediaUrl**

2. Go to **<path_to_project_root>/lib/core/config/strings.dart** to edit **@site_name**, **termsURL**, **privacyURL**, **aboutUS**, **cookiesPOLICY**, **affiliates**, and **adsShow**

1.a.9 Change App Version and Title

1. Go to **<path_to_project_root>/pubspec.yaml** and replace version number + version code: E.g. 1.x.x+23

2. Go to **<path_to_project_root>/android/app/build.gradle** and replace version number + version code: E.g. 1.x.x+23

3. Go to **<path_to_project_root>/android/local.properties** and replace version number + version code: E.g. 1.x.x+23

4. Change app title <path_to_project_root>/lib/main.dart *line 139

Please complete steps above for Android configuration before moving on to the iOS configuration below, as they both use the same files

Section 2: iOS Configuration

2.a. Edit Theme and Details

2.a.1 Change Fontsyle

1. Place your Font File (**.otf** or **.ttf**) in <path_to_project_root>/lib/fonts folder.

2. Register font path at <path_to_project_root>/pubspec.yaml. Paste the font name at the bottom font section.

EXAMPLE: -family: FONTNAME

fonts:

- assets: fonts/FONTNAME.ttf

3. Save 'pubspec.yaml' file to update new font

4. Open <path_to_project_root>/lib/main.dart and inside themeData provide this – font family: 'FONTNAME'

2.a.2 Change Color Values

1. Change Color values in **<path_to_project_root>/lib/core/config/colors.dart** inside you will find all the color values used in the project. Just replace the color value with your desired one.

2. Save the File using CTRL+S.

2.a.3 Change app Name Label

1. Provide app name inside **<path_to_project_root>/iOS/runner/info.plist;**

Part 4: Compile and Publish App [Android]

4.1 App Signing

1. Every app needs to be signed by a keystore which contains signing information by the developer/company.

2. The keystore file is present in **<path_to_project_root>/android/app/androidkey.jks**

3. The keystore should be same for each app update for respective apps. However, if you want to change keystore, please contact Play Console support team.

4. The default Keystore details provided by developer/company is shared in Secret Information section.

5. This link will assist you in setting up keystore https://docs.flutter.dev/deployment/android

4.2 Build Debug App [Test mode]

1. To test a Flutter app a real device or simulator must be connected with the computer.

2. Open project root folder at the terminal.

3. Execute the command "*flutter run*" to build a debug app. Press "r" to hot reload and "R" to hot restart the debug app while making changes to the code.

4.3 Build Release App [For Play Store]

1. Open project root folder at the terminal.

2. Once app debugging is successful, execute command "*flutter build appbundle*" to generate an app bundle to be submitted for playstore. You can also generate release ".apk"

4.4 Upload to Play Store

1. Click the link https://www.youtube.com/watch?v=5GHT4QtotE4 to see tutorial on how to configure and upload app to Play Store

Part 5: Compile and Publish App [iOS]

5.1 Installing/updating Pod File

Run following command in iOS path:

pod install
pod update
pod repo update
pod install --repo-update

5.2 Build Debug App [Test mode]

1. To test a Flutter app a real device or simulator must be connected with the computer.

2. Open project root folder at the terminal.

3. Execute the command "*flutter run*" to build a debug app. Press "r" to hot reload and "R" to hot restart the debug app while making changes to the code.

5.3 Build Release App

Please refer official documentation step wise provided by flutter team. It is updated and much useful: https://flutter.dev/docs/deployment/ios or watch this **YOUTUBE**

5.4 Publish in Apple store

Please refer official documentation step wise provided by flutter team. It is updated and much useful: https://flutter.dev/docs/deployment/ios

Part 6: Updating App

6.1 Download files

1. Download files from Codecanyon

6.2 Update application

1. Copy contents the **<Script Updates>/ 1.x.x** folder
2. Paste contents in your app respective folders

6.3 Configure file

1. Open in an IDE and navigate to **<path_to_project_root>/pubspec.yaml** file.
2. Go to flutter_icons section and change the **image_path** and **flutter_app_name**
3. Run the following commands [this would automatically change the logo, app name and package name throughout the app]**:**
   - **Change icon:** flutter pub run flutter_launcher_icons:main
   - **Change app name:** flutter pub run flutter_app_name

- **Change package name:** flutter pub run change_app_package_name:main com.newpackage.name

4. Open **<path_to_project_root>/lib/main.dart** and change the "title" at line **143.**
5. Open **<path_to_project_root>/lib/config/strings.dart** and change the @site_name@ at line **28**

6.4 Run project

1. In the terminal section, execute command *"flutter clean"*. Once completed, execute command *"flutter pub get"*.
2. Open emulator and run program to test before uploading.

Part 7: Troubleshooting

7.1 No data loading within app [Gif loader]

1. Make sure **Mod_Security** is disabled for your website via your cPanel.
2. Access the file via Cpanel core > api_req_init.php and add the following line of code **require_once("web_req_init.php");**
3. Restart or Hot Reload your mobile app

Part 8: How to

8.1 How to update android exit modal

1. Open **<path_to_project_root>/lib/core/extensions/context_exrensions.dart**
2. Navigate to the text "Do you want to exit Colibri" and make the required changes
3. Run *flutter clean* then *flutter pub get*

8.2 How to add Firebase Server key for push notifs?

1. Access Firebase then go to "Project settings"
2. In the "Cloud messaging API (Legacy) section, copy the entire Server Key
3. Go to your website admin panel, and under the push notifications section, add the key