```
{
  "root": "build/web/"
}
```

```
# This file tracks properties of this Flutter project.
# Used by Flutter tool to assess capabilities and perform upgrades etc.
#
# This file should be version controlled.

version:
  revision: 4d9e56e694b656610ab87fcf2efbcd226e0ed8cf
  channel: stable

project_type: app

# Tracks metadata for the flutter migrate command
migration:
  platforms:
    - platform: root
      create_revision: 4d9e56e694b656610ab87fcf2efbcd226e0ed8cf
      base_revision: 4d9e56e694b656610ab87fcf2efbcd226e0ed8cf
    - platform: web
      create_revision: 4d9e56e694b656610ab87fcf2efbcd226e0ed8cf
      base_revision: 4d9e56e694b656610ab87fcf2efbcd226e0ed8cf

  # User provided section

  # List of Local paths (relative to this file) that should be
  # ignored by the migrate tool.
  #
  # Files that are not part of the templates will be ignored by default.
  unmanaged_files:
    - 'lib/main.dart'
    - 'ios/Runner.xcodeproj/project.pbxproj'
```

```
GEM
  remote: https://rubygems.org/
  specs:
    CFPropertyList (3.0.6)
      rexml
    addressable (2.8.6)
      public_suffix (>= 2.0.2, < 6.0)
    artifactory (3.0.15)
    atomos (0.1.3)
    aws-eventstream (1.3.0)
    aws-partitions (1.873.0)
    aws-sdk-core (3.190.1)
      aws-eventstream (~> 1, >= 1.3.0)
      aws-partitions (~> 1, >= 1.651.0)
      aws-sigv4 (~> 1.8)
      jmespath (~> 1, >= 1.6.1)
    aws-sdk-kms (1.75.0)
      aws-sdk-core (~> 3, >= 3.188.0)
      aws-sigv4 (~> 1.1)
    aws-sdk-s3 (1.142.0)
      aws-sdk-core (~> 3, >= 3.189.0)
      aws-sdk-kms (~> 1)
      aws-sigv4 (~> 1.8)
    aws-sigv4 (1.8.0)
      aws-eventstream (~> 1, >= 1.0.2)
    babosa (1.0.4)
    claide (1.1.0)
    colored (1.2)
    colored2 (3.1.2)
    commander (4.6.0)
      highline (~> 2.0.0)
    declarative (0.0.20)
    digest-crc (0.6.5)
      rake (>= 12.0.0, < 14.0.0)
    domain_name (0.6.20231109)
    dotenv (2.8.1)
    emoji_regex (3.2.3)
    excon (0.108.0)
    faraday (1.10.3)
      faraday-em_http (~> 1.0)
      faraday-em_synchrony (~> 1.0)
      faraday-excon (~> 1.1)
      faraday-httpclient (~> 1.0)
      faraday-multipart (~> 1.0)
      faraday-net_http (~> 1.0)
      faraday-net_http_persistent (~> 1.0)
      faraday-patron (~> 1.0)
      faraday-rack (~> 1.0)
      faraday-retry (~> 1.0)
      ruby2_keywords (>= 0.0.4)
    faraday-cookie_jar (0.0.7)
      faraday (>= 0.8.0)
      http-cookie (~> 1.0.0)
    faraday-em_http (1.0.0)
    faraday-em_synchrony (1.0.0)
    faraday-excon (1.1.0)
    faraday-httpclient (1.0.1)
    faraday-multipart (1.0.4)
      multipart-post (~> 2)
    faraday-net_http (1.0.1)
    faraday-net_http_persistent (1.2.0)
    faraday-patron (1.0.0)
```

```
faraday-rack (1.0.0)
faraday-retry (1.0.3)
faraday_middleware (1.2.0)
  faraday (~> 1.0)
fastimage (2.2.7)
fastlane (2.217.0)
  CFPropertyList (>= 2.3, < 4.0.0)
  addressable (>= 2.8, < 3.0.0)
  artifactory (~> 3.0)
  aws-sdk-s3 (~> 1.0)
  babosa (>= 1.0.3, < 2.0.0)
  bundler (>= 1.12.0, < 3.0.0)
  colored
  commander (~> 4.6)
  dotenv (>= 2.1.1, < 3.0.0)
  emoji_regex (>= 0.1, < 4.0)
  excon (>= 0.71.0, < 1.0.0)
  faraday (~> 1.0)
  faraday-cookie_jar (~> 0.0.6)
  faraday_middleware (~> 1.0)
  fastimage (>= 2.1.0, < 3.0.0)
  gh_inspector (>= 1.1.2, < 2.0.0)
  google-apis-androidpublisher_v3 (~> 0.3)
  google-apis-playcustomapp_v1 (~> 0.1)
  google-cloud-storage (~> 1.31)
  highline (~> 2.0)
  http-cookie (~> 1.0.5)
  json (< 3.0.0)
  jwt (>= 2.1.0, < 3)
  mini_magick (>= 4.9.4, < 5.0.0)
  multipart-post (>= 2.0.0, < 3.0.0)
  naturally (~> 2.2)
  optparse (~> 0.1.1)
  plist (>= 3.1.0, < 4.0.0)
  rubyzip (>= 2.0.0, < 3.0.0)
  security (= 0.1.3)
  simctl (~> 1.6.3)
  terminal-notifier (>= 2.0.0, < 3.0.0)
  terminal-table (~> 3)
  tty-screen (>= 0.6.3, < 1.0.0)
  tty-spinner (>= 0.8.0, < 1.0.0)
  word_wrap (~> 1.0.0)
  xcodeproj (>= 1.13.0, < 2.0.0)
  xcpretty (~> 0.3.0)
  xcpretty-travis-formatter (>= 0.0.3)
gh_inspector (1.1.3)
google-apis-androidpublisher_v3 (0.54.0)
  google-apis-core (>= 0.11.0, < 2.a)
google-apis-core (0.11.2)
  addressable (~> 2.5, >= 2.5.1)
  googleauth (>= 0.16.2, < 2.a)
  httpclient (>= 2.8.1, < 3.a)
  mini_mime (~> 1.0)
  representable (~> 3.0)
  retriable (>= 2.0, < 4.a)
  rexml
  webrick
google-apis-iamcredentials_v1 (0.17.0)
  google-apis-core (>= 0.11.0, < 2.a)
google-apis-playcustomapp_v1 (0.13.0)
  google-apis-core (>= 0.11.0, < 2.a)
google-apis-storage_v1 (0.29.0)
```

```
    google-apis-core (>= 0.11.0, < 2.a)
  google-cloud-core (1.6.1)
    google-cloud-env (>= 1.0, < 3.a)
    google-cloud-errors (~> 1.0)
  google-cloud-env (2.1.0)
    faraday (>= 1.0, < 3.a)
  google-cloud-errors (1.3.1)
  google-cloud-storage (1.45.0)
    addressable (~> 2.8)
    digest-crc (~> 0.4)
    google-apis-iamcredentials_v1 (~> 0.1)
    google-apis-storage_v1 (~> 0.29.0)
    google-cloud-core (~> 1.6)
    googleauth (>= 0.16.2, < 2.a)
    mini_mime (~> 1.0)
  googleauth (1.9.1)
    faraday (>= 1.0, < 3.a)
    google-cloud-env (~> 2.1)
    jwt (>= 1.4, < 3.0)
    multi_json (~> 1.11)
    os (>= 0.9, < 2.0)
    signet (>= 0.16, < 2.a)
  highline (2.0.3)
  http-cookie (1.0.5)
    domain_name (~> 0.5)
  httpclient (2.8.3)
  jmespath (1.6.2)
  json (2.7.1)
  jwt (2.7.1)
  mini_magick (4.12.0)
  mini_mime (1.1.5)
  multi_json (1.15.0)
  multipart-post (2.3.0)
  nanaimo (0.3.0)
  naturally (2.2.1)
  optparse (0.1.1)
  os (1.1.4)
  plist (3.7.0)
  public_suffix (5.0.4)
  rake (13.1.0)
  representable (3.2.0)
    declarative (< 0.1.0)
    trailblazer-option (>= 0.1.1, < 0.2.0)
    uber (< 0.2.0)
  retriable (3.1.2)
  rexml (3.2.6)
  rouge (2.0.7)
  ruby2_keywords (0.0.5)
  rubyzip (2.3.2)
  security (0.1.3)
  signet (0.18.0)
    addressable (~> 2.8)
    faraday (>= 0.17.5, < 3.a)
    jwt (>= 1.5, < 3.0)
    multi_json (~> 1.10)
  simctl (1.6.10)
    CFPropertyList
    naturally
  terminal-notifier (2.0.0)
  terminal-table (3.0.2)
    unicode-display_width (>= 1.1.1, < 3)
  trailblazer-option (0.1.2)
```

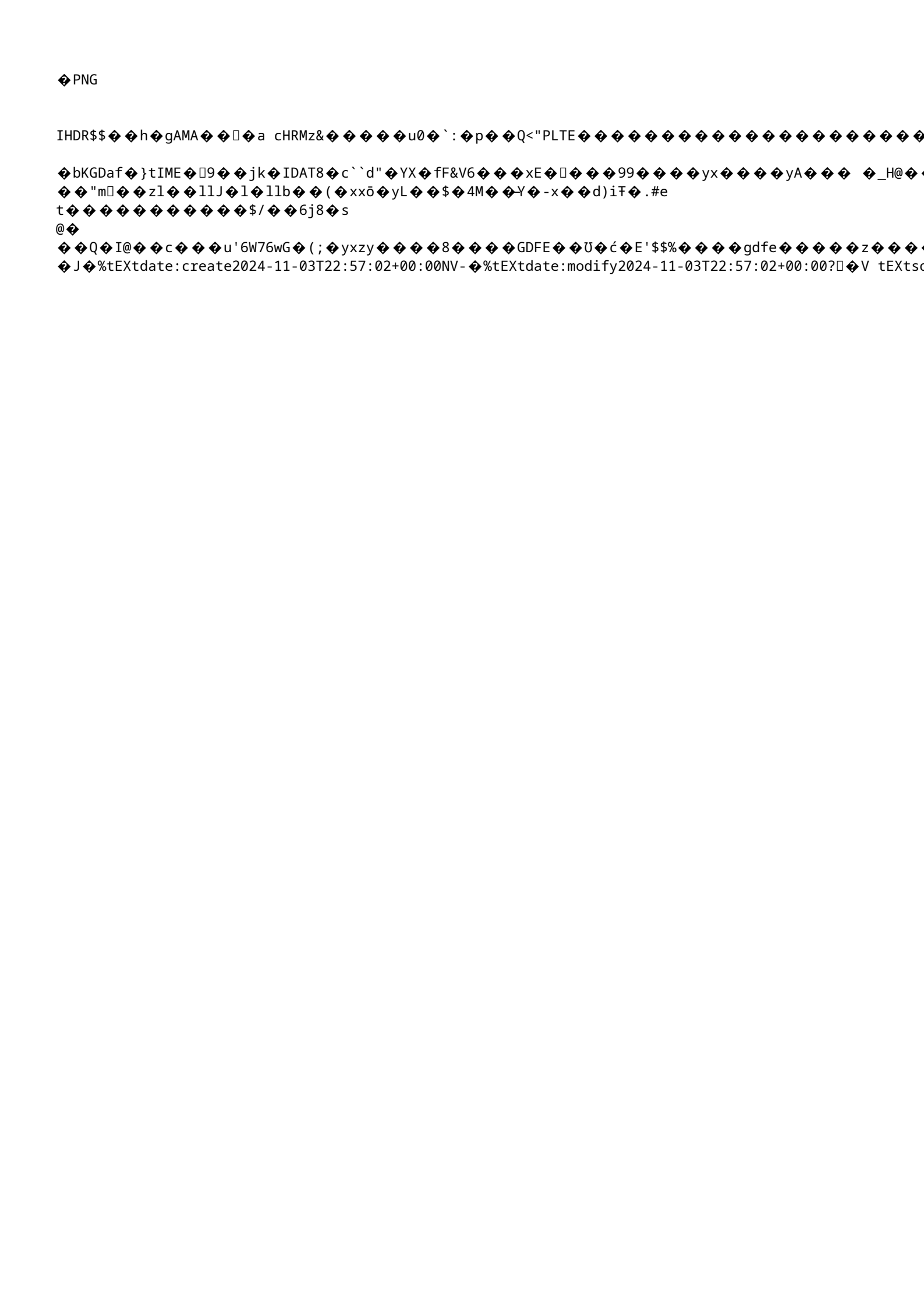
```
    tty-cursor (0.7.1)
    tty-screen (0.8.2)
    tty-spinner (0.9.3)
      tty-cursor (~> 0.7)
    uber (0.1.0)
    unicode-display_width (2.5.0)
    webrick (1.8.1)
    word_wrap (1.0.0)
    xcodeproj (1.23.0)
      CFPropertyList (>= 2.3.3, < 4.0)
      atomos (~> 0.1.3)
      claide (>= 1.0.2, < 2.0)
      colored2 (~> 3.1)
      nanaimo (~> 0.3.0)
      rexml (~> 3.2.4)
    xcpretty (0.3.0)
      rouge (~> 2.0.7)
    xcpretty-travis-formatter (1.0.1)
      xcpretty (~> 0.2, >= 0.0.7)

PLATFORMS
  arm64-darwin-23

DEPENDENCIES
  fastlane

BUNDLED WITH
   2.4.10
```

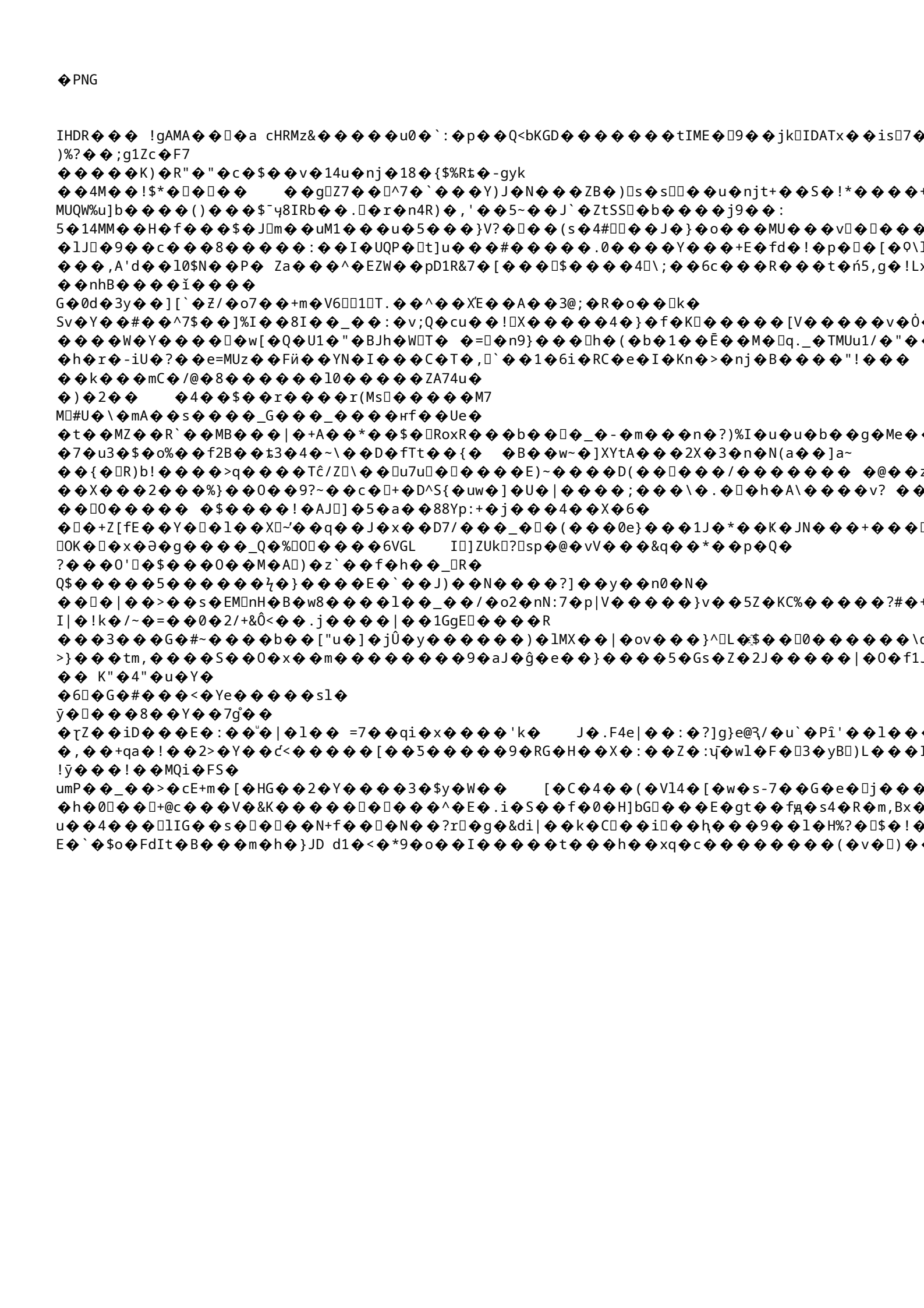
```
// This is a basic Flutter widget test.
//
// To perform an interaction with a widget in your test, use the WidgetTester
// utility in the flutter_test package. For example, you can send tap and scroll
// gestures. You can also use WidgetTester to find child widgets in the widget
// tree, read text, and verify that the values of widget properties are correct.

import "package:flutter_test/flutter_test.dart";


void main() {
  testWidgets("Counter increments smoke test", (final tester) async {
    // Build our app and trigger a frame.
/*    await tester.pumpWidget(const MyApp());

    // Verify that our counter starts at 0.
    expect(find.text('0'), findsOneWidget);
    expect(find.text('1'), findsNothing);

    // Tap the '+' icon and trigger a frame.
    await tester.tap(find.byIcon(Icons.add));
    await tester.pump();

    // Verify that our counter has incremented.
    expect(find.text('0'), findsNothing);
    expect(find.text('1'), findsOneWidget);*/
  });
}
```

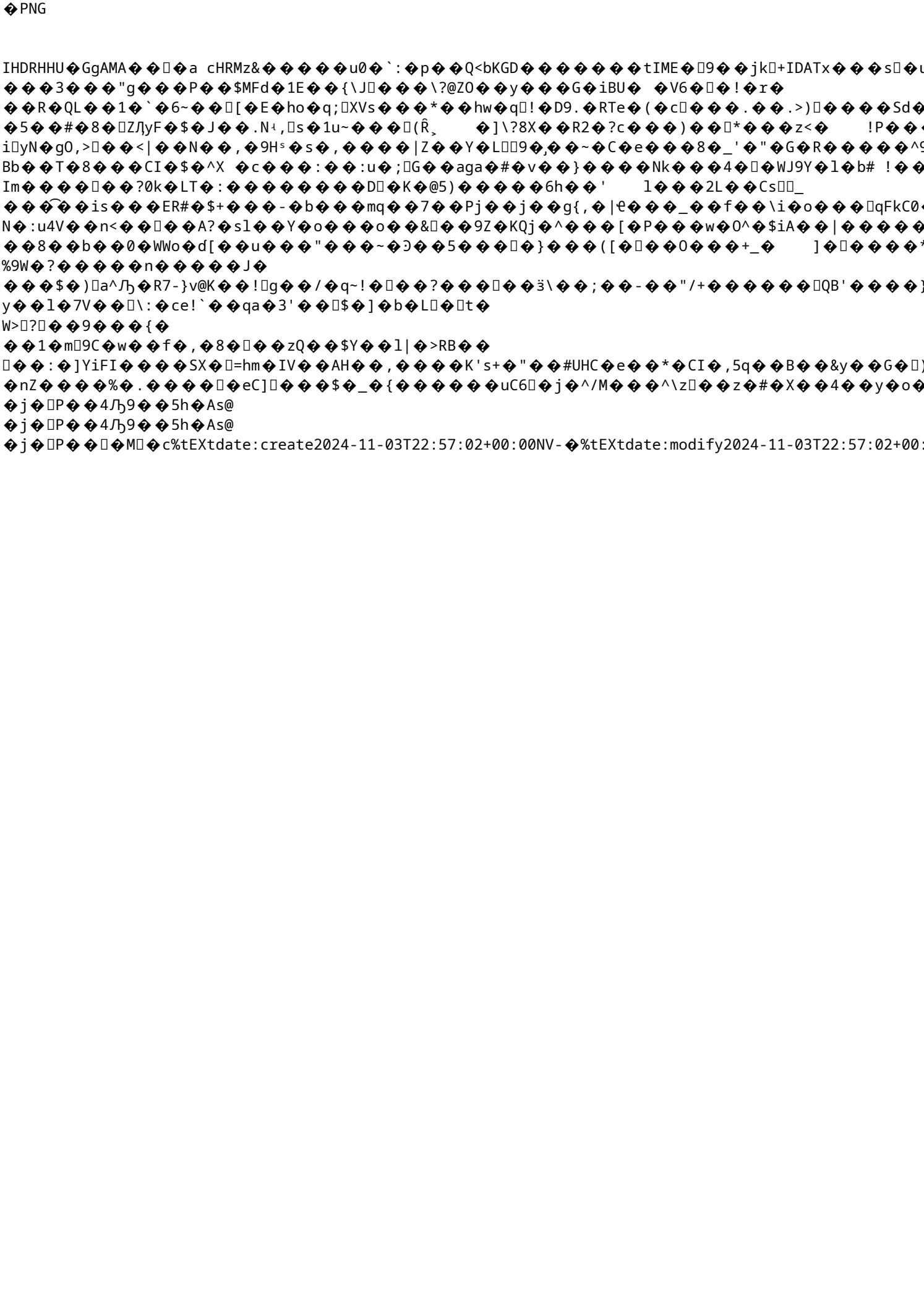
```
/* Reset default margins and paddings */
body {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

/* Styles for the loading screen */
#loading {
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: flex-start;
    position: fixed;
    width: 100%;
    height: 100%;
    background-color: #f0f0f0; /* Light gray background */
    padding: 10px;
    overflow-y: auto; /* Allows scrolling if content becomes too large */
}

/* Skeleton loader styles */
.skeleton-logo {
    width: 80%;
    max-width: 350px;
    height: 100px;
    background-color: #e0e0e0;
    border-radius: 8px;
    margin-bottom: 20px;
    animation: skeleton-loading 1.5s infinite alternate;
}

.skeleton-group-name {
    width: 90%;
    max-width: 600px;
    height: 25px;
    background-color: #e0e0e0;
    border-radius: 4px;
    margin: 20px 0 10px;
    animation: skeleton-loading 1.5s infinite alternate;
}

.skeleton-item {
    width: 90%;
    max-width: 600px;
    background-color: #e0e0e0;
    border-radius: 4px;
    padding: 10px;
    margin-bottom: 15px;
    animation: skeleton-loading 1.5s infinite alternate;
}

.skeleton-title {
    width: 70%;
    height: 20px;
    background-color: #d0d0d0;
    border-radius: 4px;
    margin-bottom: 10px;
    animation: skeleton-loading 1.5s infinite alternate;
}

.skeleton-description {
```

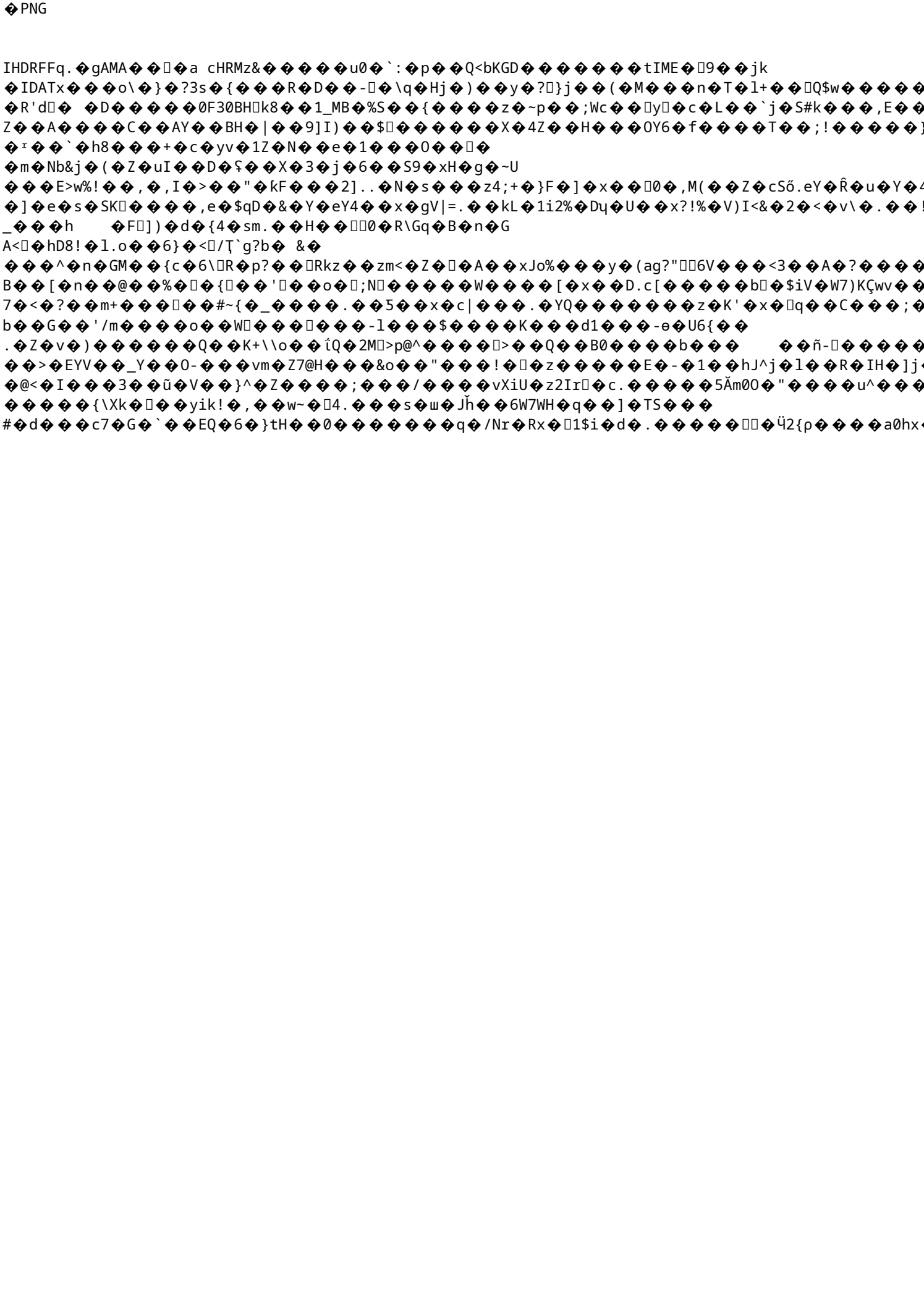
```
    width: 100%;
    height: 15px;
    background-color: #d0d0d0;
    border-radius: 4px;
    margin-bottom: 8px;
    animation: skeleton-loading 1.5s infinite alternate;
}

.skeleton-price {
    width: 30%;
    height: 20px;
    background-color: #c0c0c0;
    border-radius: 4px;
    margin-top: 10px;
    animation: skeleton-loading 1.5s infinite alternate;
}

/* Simplified animation for the skeleton loading effect */
@keyframes skeleton-loading {
    0% {
        background-color: #e0e0e0;
    }
    100% {
        background-color: #d0d0d0;
    }
}

/* Media query for smaller screens */
@media (max-width: 480px) {
    #loading {
        padding: 5px;
    }

    .skeleton-logo {
        width: 100%;
        height: 80px;
    }

    .skeleton-group-name {
        width: 100%;
    }

    .skeleton-item {
        width: 100%;
        padding: 8px;
    }

    .skeleton-title {
        width: 80%;
    }

    .skeleton-description {
        width: 90%;
    }

    .skeleton-price {
        width: 40%;
    }
}
```

```
<?xml version="1.0" encoding="utf-8"?>
<browserconfig>
    <msapplication>
        <tile>
            <square70x70logo src="/ms-icon-70x70.png" />
            <square150x150logo src="/ms-icon-150x150.png" />
            <square310x310logo src="/ms-icon-310x310.png" />
            <TileColor>#ffffff</TileColor>
        </tile>
    </msapplication>
</browserconfig>
```

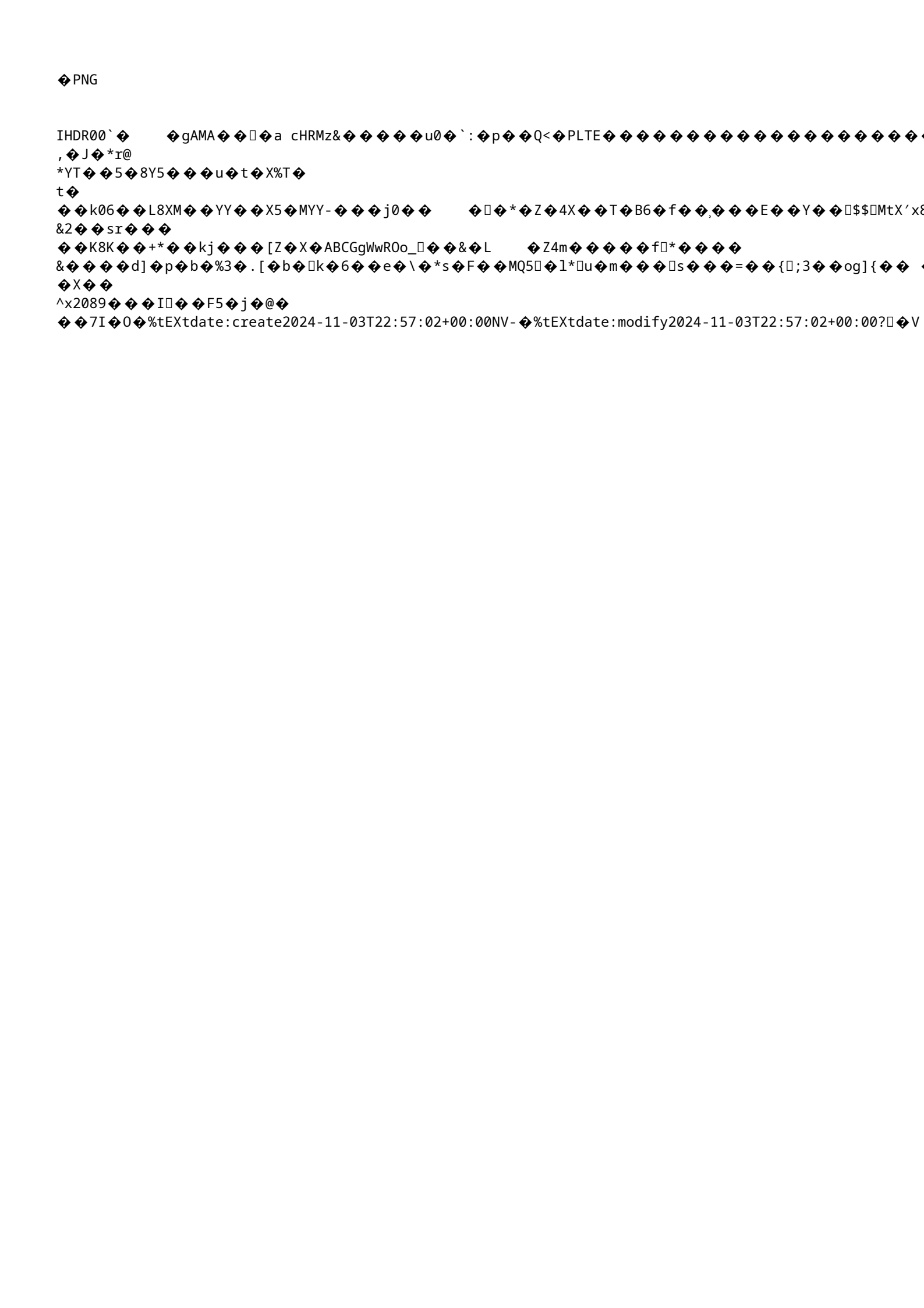


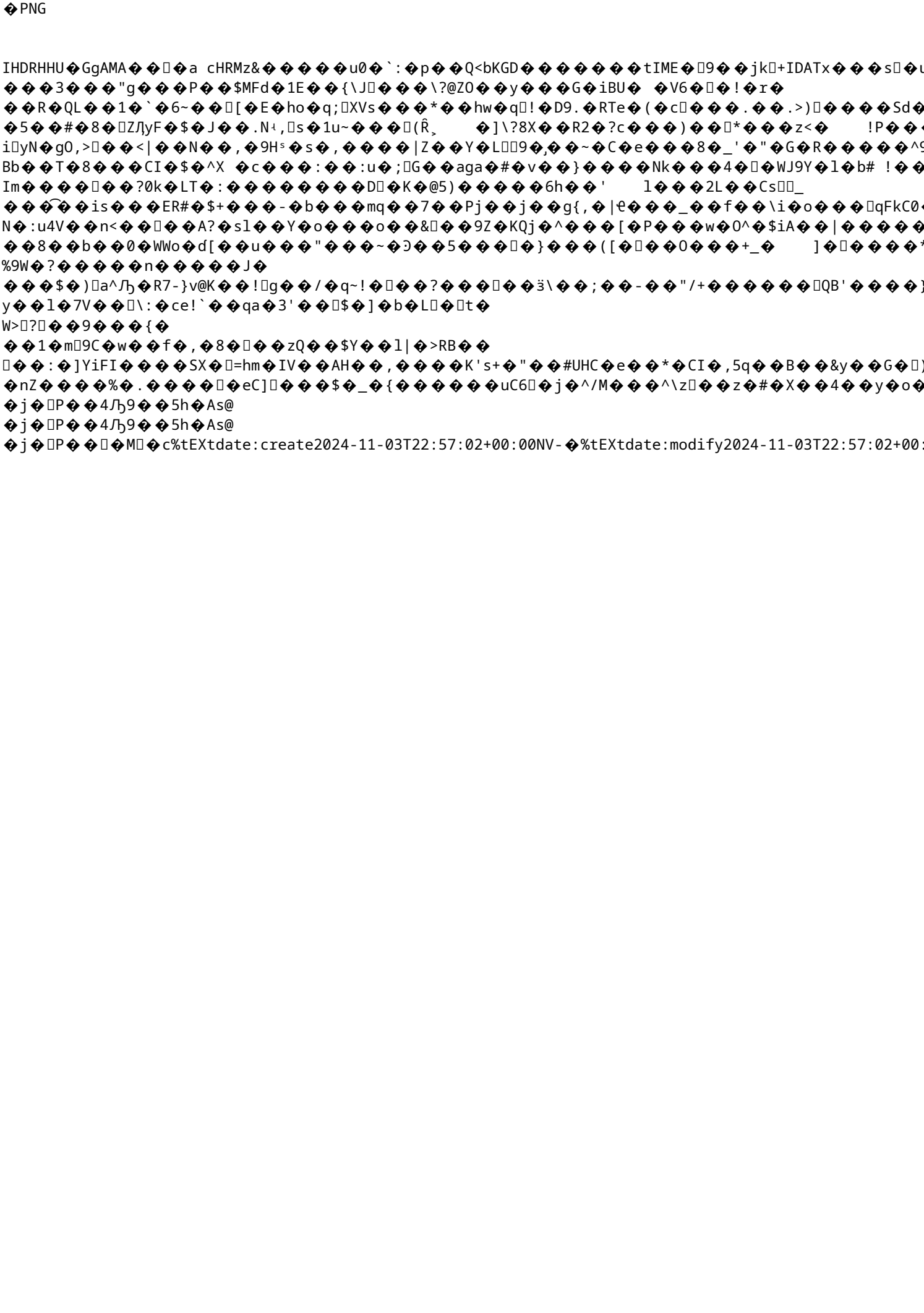


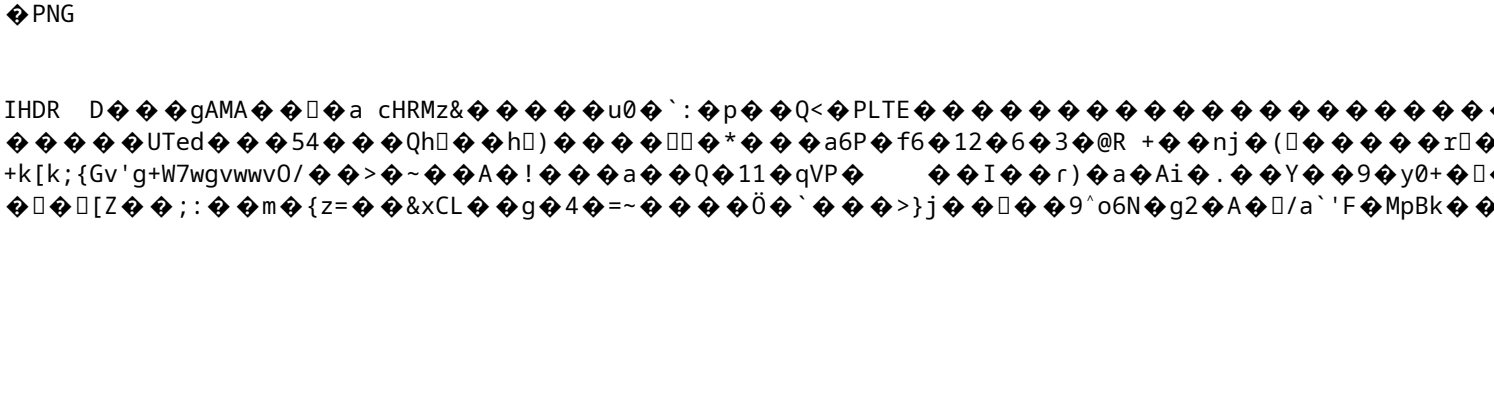


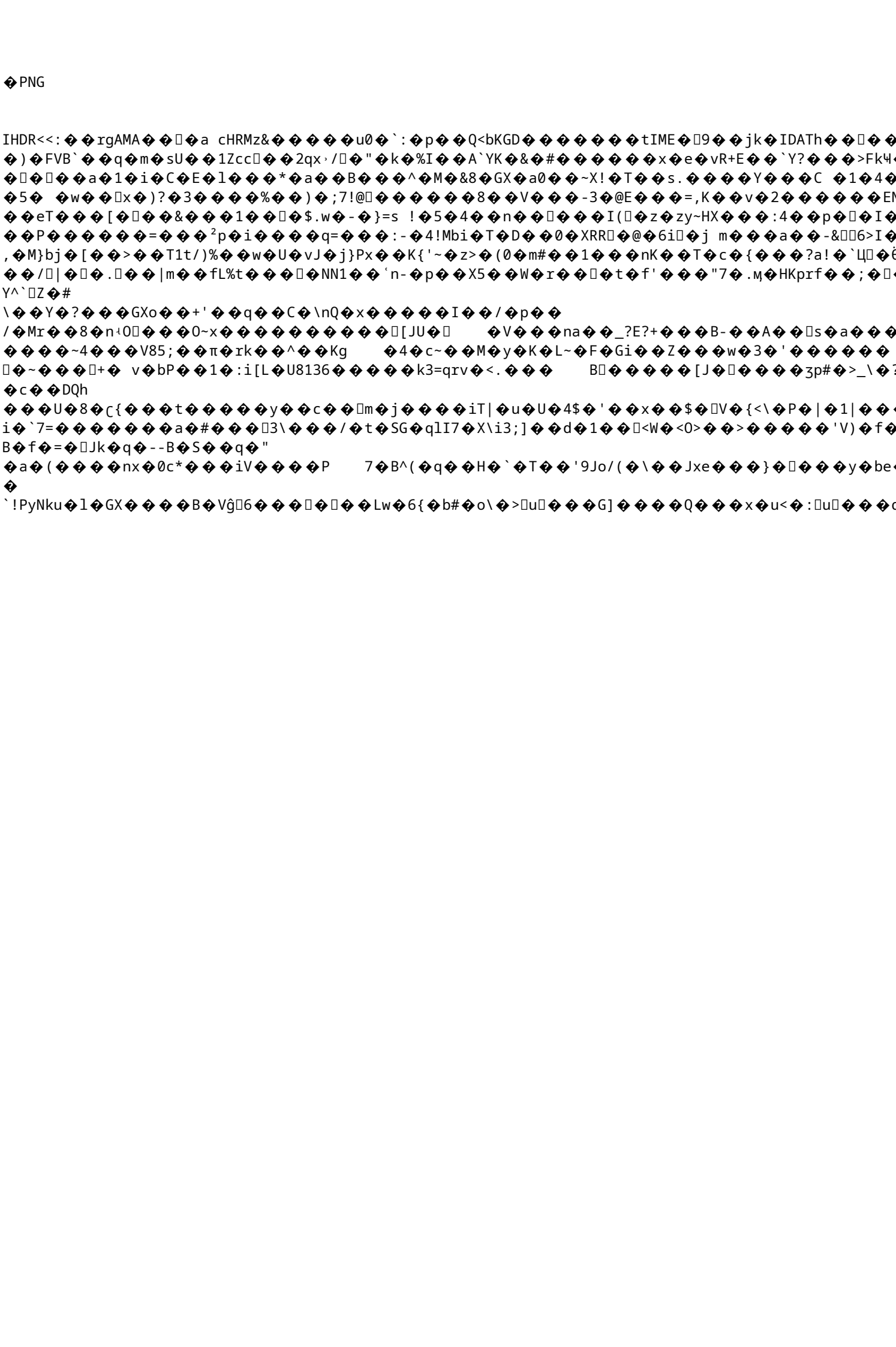


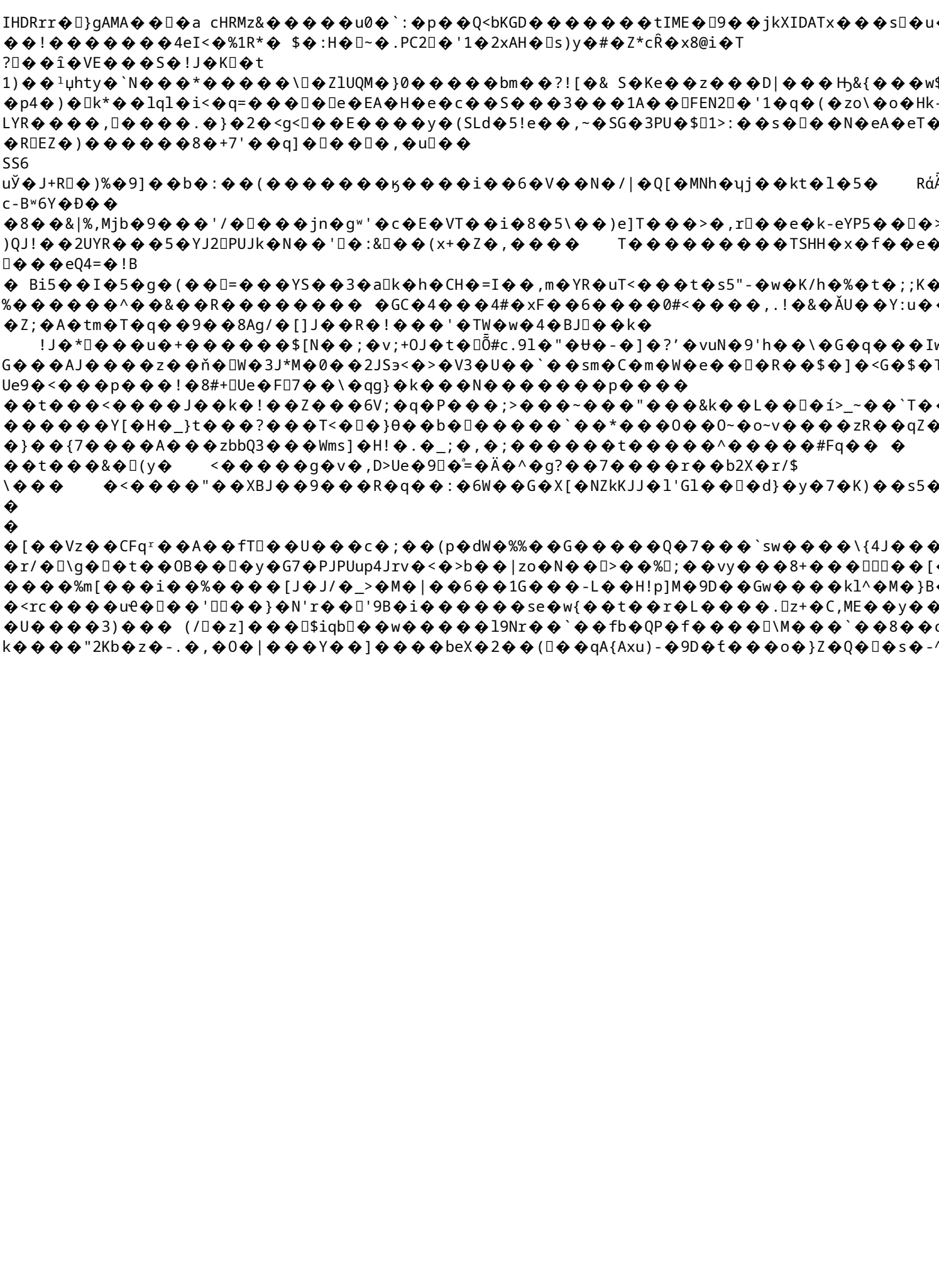
h(��











```
<!DOCTYPE html>
<html lang="en">
<head>
    <!-- Base URL for the application -->
    <base href="$FLUTTER_BASE_HREF"/>
    <meta charset="UTF-8"/>
    <meta http-equiv="X-UA-Compatible" content="IE=Edge"/>
    <!-- Google Sign-In Client ID -->
    <meta name="google-signin-client_id"
          content="484842176385-uoacb8pcvp1l6tonf6qd0rf13kh7suml.apps.googleusercontent.com"/>
    <!-- Mobile App Capabilities -->
    <meta name="apple-mobile-web-app-capable" content="yes"/>
    <meta name="apple-mobile-web-app-status-bar-style" content="black"/>
    <meta name="apple-mobile-web-app-title" content="onelenyk.dev"/>
    <!-- Icons -->
    <link rel="apple-touch-icon" href="/icons/apple-icon.png"/>
    <link rel="apple-touch-icon" sizes="57x57" href="/icons/apple-icon-57x57.png"/>
    <link rel="apple-touch-icon" sizes="60x60" href="/icons/apple-icon-60x60.png"/>
    <link rel="apple-touch-icon" sizes="72x72" href="/icons/apple-icon-72x72.png"/>
    <link rel="apple-touch-icon" sizes="76x76" href="/icons/apple-icon-76x76.png"/>
    <link rel="apple-touch-icon" sizes="114x114" href="/icons/apple-icon-114x114.png"/>
    <link rel="apple-touch-icon" sizes="120x120" href="/icons/apple-icon-120x120.png"/>
    <link rel="apple-touch-icon" sizes="144x144" href="/icons/apple-icon-144x144.png"/>
    <link rel="apple-touch-icon" sizes="152x152" href="/icons/apple-icon-152x152.png"/>
    <link rel="apple-touch-icon" sizes="180x180" href="/icons/apple-icon-180x180.png"/>
    <link rel="icon" type="image/png" sizes="192x192" href="/icons/android-icon-192x192.png"/>
    <link rel="icon" type="image/png" sizes="32x32" href="/icons/favicon-32x32.png"/>
    <link rel="icon" type="image/png" sizes="96x96" href="/icons/favicon-96x96.png"/>
    <link rel="icon" type="image/png" sizes="16x16" href="/icons/favicon-16x16.png"/>
    <meta name="msapplication-TileColor" content="#ffffff"/>
    <meta name="msapplication-TileImage" content="/icons/ms-icon-144x144.png"/>
    <meta name="theme-color" content="#ffffff"/>
    <!-- Title -->
    <title>onelenyk.dev</title>
    <!-- Manifest for PWA -->
    <link rel="manifest" href="manifest.json"/>
    <script>
        // The value below is injected by flutter build, do not touch.
        var serviceWorkerVersion = null;
    </script>
    <!-- Include Flutter Web App JavaScript -->
    <script defer src="flutter.js"></script>
    <!-- Inline CSS for image loader styles -->
    <style>
        /* Reset default margins and paddings */
        body {
            margin: 0;
            padding: 0;
            box-sizing: border-box;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            background-color: #111315;
        }
        /* Styles for the blinking image */
       #blinking-image {
            width: 270px;
            height: 130px;
               object-fit: contain;
            animation: blink 2s infinite ease-in-out;
        }
```

```
        /* Blinking animation */
        @keyframes blink {
            0% { opacity: 1; }
            50% { opacity: 0.5; }
            100% { opacity: 1; }
        }
    </style>
</head>
<body>
<!-- Loading screen with blinking image -->
<img id="blinking-image" src="assets/assets/svg/mainlogo.svg" alt="Logo">
<!-- Flutter initialization script -->
<script>
    window.addEventListener("load", function () {
        var loading = document.getElementById("blinking-image");
        _flutter.loader
            .loadEntrypoint({
                serviceWorker: {
                    serviceWorkerVersion: serviceWorkerVersion,
                },
                onEntrypointLoaded: function (engineInitializer) {
                    engineInitializer
                        .initializeEngine({
                            useColorEmoji: true,
                        })
                        .then(function (appRunner) {
                            appRunner.runApp();
                        })
                        .then(function () {
                          window.setTimeout(function () {
                           loading.remove();
                           }, 600);
                           });
                },
            });
    });
</script>
</body>
</html>
```

```
{
  "name": "onelenyk.dev",
  "short_name": "onelenyk.dev",
  "start_url": ".",
  "display": "standalone",
  "background_color": "#0175C2",
  "theme_color": "#0175C2",
  "description": "The onelenyk project site",
  "orientation": "portrait-primary",
  "prefer_related_applications": false,
  "icons": [
    {
      "src": "\/android-icon-36x36.png",
      "sizes": "36x36",
      "type": "image\/png",
      "density": "0.75"
    },
    {
      "src": "\/android-icon-48x48.png",
      "sizes": "48x48",
      "type": "image\/png",
      "density": "1.0"
    },
    {
      "src": "\/android-icon-72x72.png",
      "sizes": "72x72",
      "type": "image\/png",
      "density": "1.5"
    },
    {
      "src": "\/android-icon-96x96.png",
      "sizes": "96x96",
      "type": "image\/png",
      "density": "2.0"
    },
    {
      "src": "\/android-icon-144x144.png",
      "sizes": "144x144",
      "type": "image\/png",
      "density": "3.0"
    },
    {
      "src": "\/android-icon-192x192.png",
      "sizes": "192x192",
      "type": "image\/png",
      "density": "4.0"
    }
  ]
}
```

```
# Generated by pub
# See https://dart.dev/tools/pub/glossary#lockfile
packages:
  _fe_analyzer_shared:
    dependency: transitive
    description:
      name: _fe_analyzer_shared
      sha256: "0b2f2bd91ba804e53a61d757b986f89f1f9eaed5b11e4b2f5a2468d86d6c9fc7"
      url: "https://pub.dev"
    source: hosted
    version: "67.0.0"
  _flutterfire_internals:
    dependency: transitive
    description:
      name: _flutterfire_internals
      sha256: "37a42d06068e2fe3deddb2da079a8c4d105f241225ba27b7122b37e9865fd8f7"
      url: "https://pub.dev"
    source: hosted
    version: "1.3.35"
  analyzer:
    dependency: transitive
    description:
      name: analyzer
      sha256: "37577842a27e4338429a1cbc32679d508836510b056f1eedf0c8d20e39c1383d"
      url: "https://pub.dev"
    source: hosted
    version: "6.4.1"
  archive:
    dependency: transitive
    description:
      name: archive
      sha256: cb6a278ef2dbb298455e1a713bda08524a175630ec643a242c399c932a0a1f7d
      url: "https://pub.dev"
    source: hosted
    version: "3.6.1"
  args:
    dependency: transitive
    description:
      name: args
      sha256: bf9f5caeea8d8fe6721a9c358dd8a5c1947b27f1cfaa18b39c301273594919e6
      url: "https://pub.dev"
    source: hosted
    version: "2.6.0"
  async:
    dependency: transitive
    description:
      name: async
      sha256: "947bfcf187f74dbc5e146c9eb9c0f10c9f8b30743e341481c1e2ed3ecc18c20c"
      url: "https://pub.dev"
    source: hosted
    version: "2.11.0"
  auto_route:
    dependency: "direct main"
    description:
      name: auto_route
      sha256: eb33554581a0a4aa7e6da0f13a44291a55bf71359012f1d9feb41634ff908ff8
      url: "https://pub.dev"
    source: hosted
    version: "7.9.2"
  auto_route_generator:
    dependency: "direct dev"
    description:
```

```
    name: auto_route_generator
    sha256: "11067a3bcd643812518fe26c0c9ec073990286cabfd9d74b6da9ef9b913c4d22"
    url: "https://pub.dev"
  source: hosted
  version: "7.3.2"
barcode:
  dependency: transitive
  description:
    name: barcode
    sha256: ab180ce22c6555d77d45f0178a523669db67f95856e3378259ef2ffeb43e6003
    url: "https://pub.dev"
  source: hosted
  version: "2.2.8"
bidi:
  dependency: transitive
  description:
    name: bidi
    sha256: "9a712c7ddf708f7c41b1923aa83648a3ed44cfd75b04f72d598c45e5be287f9d"
    url: "https://pub.dev"
  source: hosted
  version: "2.0.12"
bloc:
  dependency: transitive
  description:
    name: bloc
    sha256: "106842ad6569f0b60297619e9e0b1885c2fb9bf84812935490e6c5275777804e"
    url: "https://pub.dev"
  source: hosted
  version: "8.1.4"
boolean_selector:
  dependency: transitive
  description:
    name: boolean_selector
    sha256: "6cfb5af12253eaf2b368f07bacc5a80d1301a071c73360d746b7f2e32d762c66"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.1"
build:
  dependency: transitive
  description:
    name: build
    sha256: "80184af8b6cb3e5c1c4ec6d8544d27711700bc3e6d2efad04238c7b5290889f0"
    url: "https://pub.dev"
  source: hosted
  version: "2.4.1"
build_cli_annotations:
  dependency: transitive
  description:
    name: build_cli_annotations
    sha256: b59d2769769efd6c9ff6d4c4cede0be115a566afc591705c2040b707534b1172
    url: "https://pub.dev"
  source: hosted
  version: "2.1.0"
build_config:
  dependency: transitive
  description:
    name: build_config
    sha256: bf80fcfb46a29945b423bd9aad884590fb1dc69b330a4d4700cac476af1708d1
    url: "https://pub.dev"
  source: hosted
  version: "1.1.1"
build_daemon:
```

```
  dependency: transitive
  description:
    name: build_daemon
    sha256: "79b2aef6ac2ed00046867ed354c88778c9c0f029df8a20fe10b5436826721ef9"
    url: "https://pub.dev"
  source: hosted
  version: "4.0.2"
build_resolvers:
  dependency: transitive
  description:
    name: build_resolvers
    sha256: "339086358431fa15d7eca8b6a36e5d783728cf025e559b834f4609a1fcfb7b0a"
    url: "https://pub.dev"
  source: hosted
  version: "2.4.2"
build_runner:
  dependency: "direct dev"
  description:
    name: build_runner
    sha256: "644dc98a0f179b872f612d3eb627924b578897c629788e858157fa5e704ca0c7"
    url: "https://pub.dev"
  source: hosted
  version: "2.4.11"
build_runner_core:
  dependency: transitive
  description:
    name: build_runner_core
    sha256: e3c79f69a64bdfcd8a776a3c28db4eb6e3fb5356d013ae5eb2e52007706d5dbe
    url: "https://pub.dev"
  source: hosted
  version: "7.3.1"
built_collection:
  dependency: transitive
  description:
    name: built_collection
    sha256: "376e3dd27b51ea877c28d525560790aee2e6fbb5f20e2f85d5081027d94e2100"
    url: "https://pub.dev"
  source: hosted
  version: "5.1.1"
built_value:
  dependency: transitive
  description:
    name: built_value
    sha256: c7913a9737ee4007efedaffc968c049fd0f3d0e49109e778edc10de9426005cb
    url: "https://pub.dev"
  source: hosted
  version: "8.9.2"
cached_network_image:
  dependency: "direct main"
  description:
    name: cached_network_image
    sha256: "4a5d8d2c728b0f3d0245f69f921d7be90cae4c2fd5288f773088672c0893f819"
    url: "https://pub.dev"
  source: hosted
  version: "3.4.0"
cached_network_image_platform_interface:
  dependency: transitive
  description:
    name: cached_network_image_platform_interface
    sha256: "35814b016e37fbdc91f7ae18c8caf49ba5c88501813f73ce8a07027a395e2829"
    url: "https://pub.dev"
  source: hosted
```

```
  version: "4.1.1"
cached_network_image_web:
  dependency: transitive
  description:
    name: cached_network_image_web
    sha256: "6322dde7a5ad92202e64df659241104a43db20ed594c41ca18de1014598d7996"
    url: "https://pub.dev"
  source: hosted
  version: "1.3.0"
carousel_slider:
  dependency: "direct main"
  description:
    name: carousel_slider
    sha256: "9c695cc963bf1d04a47bd6021f68befce8970bcd61d24938e1fb0918cf5d9c42"
    url: "https://pub.dev"
  source: hosted
  version: "4.2.1"
characters:
  dependency: transitive
  description:
    name: characters
    sha256: "04a925763edad70e8443c99234dc3328f442e811f1d8fd1a72f1c8ad0f69a605"
    url: "https://pub.dev"
  source: hosted
  version: "1.3.0"
checked_yaml:
  dependency: transitive
  description:
    name: checked_yaml
    sha256: feb6bed21949061731a7a75fc5d2aa727cf160b91af9a3e464c5e3a32e28b5ff
    url: "https://pub.dev"
  source: hosted
  version: "2.0.3"
clock:
  dependency: transitive
  description:
    name: clock
    sha256: cb6d7f03e1de671e34607e909a7213e31d7752be4fb66a86d29fe1eb14bfb5cf
    url: "https://pub.dev"
  source: hosted
  version: "1.1.1"
cloud_firestore:
  dependency: "direct main"
  description:
    name: cloud_firestore
    sha256: a0f161b92610e078b4962d7e6ebeb66dc9cce0ada3514aeee442f68165d78185
    url: "https://pub.dev"
  source: hosted
  version: "4.17.5"
cloud_firestore_platform_interface:
  dependency: transitive
  description:
    name: cloud_firestore_platform_interface
    sha256: "6a55b319f8d33c307396b9104512e8130a61904528ab7bd8b5402678fca54b81"
    url: "https://pub.dev"
  source: hosted
  version: "6.2.5"
cloud_firestore_web:
  dependency: transitive
  description:
    name: cloud_firestore_web
    sha256: "89dfa1304d3da48b3039abbb2865e3d30896ef858e569a16804a99f4362283a9"
```

```
    url: "https://pub.dev"
  source: hosted
  version: "3.12.5"
code_builder:
  dependency: transitive
  description:
    name: code_builder
    sha256: f692079e25e7869c14132d39f223f8eec9830eb76131925143b2129c4bb01b37
    url: "https://pub.dev"
  source: hosted
  version: "4.10.0"
collection:
  dependency: transitive
  description:
    name: collection
    sha256: ee67cb0715911d28db6bf4af1026078bd6f0128b07a5f66fb2ed94ec6783c09a
    url: "https://pub.dev"
  source: hosted
  version: "1.18.0"
convert:
  dependency: transitive
  description:
    name: convert
    sha256: b30acd5944035672bc15c6b7a8b47d773e41e2f17de064350988c5d02adb1c68
    url: "https://pub.dev"
  source: hosted
  version: "3.1.2"
crypto:
  dependency: transitive
  description:
    name: crypto
    sha256: "1e445881f28f22d6140f181e07737b22f1e099a5e1ff94b0af2f9e4a463f4855"
    url: "https://pub.dev"
  source: hosted
  version: "3.0.6"
cupertino_icons:
  dependency: "direct main"
  description:
    name: cupertino_icons
    sha256: ba631d1c7f7bef6b729a622b7b752645a2d076dba9976925b8f25725a30e1ee6
    url: "https://pub.dev"
  source: hosted
  version: "1.0.8"
dart_style:
  dependency: transitive
  description:
    name: dart_style
    sha256: "99e066ce75c89d6b29903d788a7bb9369cf754f7b24bf70bf4b6d6d6b26853b9"
    url: "https://pub.dev"
  source: hosted
  version: "2.3.6"
device_info_plus:
  dependency: transitive
  description:
    name: device_info_plus
    sha256: "77f757b789ff68e4eaf9c56d1752309bd9f7ad557cb105b938a7f8eb89e59110"
    url: "https://pub.dev"
  source: hosted
  version: "9.1.2"
device_info_plus_platform_interface:
  dependency: transitive
  description:
```

```
    name: device_info_plus_platform_interface
    sha256: "282d3cf731045a2feb66abfe61bbc40870ae50a3ed10a4d3d217556c35c8c2ba"
    url: "https://pub.dev"
  source: hosted
  version: "7.0.1"
dhttpd:
  dependency: "direct main"
  description:
    name: dhttpd
    sha256: "2e24765d7569b8e0a02a441e3cf96f09cca69dfecba646e7e9f6b3ab45a2f3fe"
    url: "https://pub.dev"
  source: hosted
  version: "4.1.0"
diff_match_patch:
  dependency: transitive
  description:
    name: diff_match_patch
    sha256: "2efc9e6e8f449d0abe15be240e2c2a3bcd977c8d126cfd70598aee60af35c0a4"
    url: "https://pub.dev"
  source: hosted
  version: "0.4.1"
dio:
  dependency: transitive
  description:
    name: dio
    sha256: "5598aa796bbf4699afd5c67c0f5f6e2ed542afc956884b9cd58c306966efc260"
    url: "https://pub.dev"
  source: hosted
  version: "5.7.0"
dio_web_adapter:
  dependency: transitive
  description:
    name: dio_web_adapter
    sha256: "33259a9276d6cea88774a0000cfae0d861003497755969c92faa223108620dc8"
    url: "https://pub.dev"
  source: hosted
  version: "2.0.0"
dotted_border:
  dependency: "direct main"
  description:
    name: dotted_border
    sha256: "108837e11848ca776c53b30bc870086f84b62ed6e01c503ed976e8f8c7df9c04"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.0"
equatable:
  dependency: "direct main"
  description:
    name: equatable
    sha256: c2b87cb7756efdf69892005af546c56c0b5037f54d2a88269b4f347a505e3ca2
    url: "https://pub.dev"
  source: hosted
  version: "2.0.5"
fake_async:
  dependency: transitive
  description:
    name: fake_async
    sha256: "511392330127add0b769b75a987850d136345d9227c6b94c96a04cf4a391bf78"
    url: "https://pub.dev"
  source: hosted
  version: "1.3.1"
ffi:
```

```
  dependency: transitive
  description:
    name: ffi
    sha256: "16ed7b077ef01ad6170a3d0c57caa4a112a38d7a2ed5602e0aca9ca6f3d98da6"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.3"
file:
  dependency: transitive
  description:
    name: file
    sha256: a3b4f84adafef897088c160faf7dfffb7696046cb13ae90b508c2cbc95d3b8d4
    url: "https://pub.dev"
  source: hosted
  version: "7.0.1"
file_saver:
  dependency: "direct main"
  description:
    name: file_saver
    sha256: d375b351e3331663abbaf99747abd72f159260c58fbbdbca9f926f02c01bdc48
    url: "https://pub.dev"
  source: hosted
  version: "0.2.13"
firebase_auth:
  dependency: "direct main"
  description:
    name: firebase_auth
    sha256: cfc2d970829202eca09e2896f0a5aa7c87302817ecc0bdfa954f026046bf10ba
    url: "https://pub.dev"
  source: hosted
  version: "4.20.0"
firebase_auth_platform_interface:
  dependency: transitive
  description:
    name: firebase_auth_platform_interface
    sha256: a0270e1db3b2098a14cb2a2342b3cd2e7e458e0c391b1f64f6f78b14296ec093
    url: "https://pub.dev"
  source: hosted
  version: "7.3.0"
firebase_auth_web:
  dependency: transitive
  description:
    name: firebase_auth_web
    sha256: "64e067e763c6378b7e774e872f0f59f6812885e43020e25cde08f42e9459837b"
    url: "https://pub.dev"
  source: hosted
  version: "5.12.0"
firebase_core:
  dependency: "direct main"
  description:
    name: firebase_core
    sha256: "26de145bb9688a90962faec6f838247377b0b0d32cc0abecd9a4e43525fc856c"
    url: "https://pub.dev"
  source: hosted
  version: "2.32.0"
firebase_core_platform_interface:
  dependency: transitive
  description:
    name: firebase_core_platform_interface
    sha256: e30da58198a6d4b49d5bce4e852f985c32cb10db329ebef9473db2b9f09ce810
    url: "https://pub.dev"
  source: hosted
```

```
  version: "5.3.0"
firebase_core_web:
  dependency: transitive
  description:
    name: firebase_core_web
    sha256: "362e52457ed2b7b180964769c1e04d1e0ea0259fdf7025fdfedd019d4ae2bd88"
    url: "https://pub.dev"
  source: hosted
  version: "2.17.5"
fixnum:
  dependency: transitive
  description:
    name: fixnum
    sha256: b6dc7065e46c974bc7c5f143080a6764ec7a4be6da1285ececdc37be96de53be
    url: "https://pub.dev"
  source: hosted
  version: "1.1.1"
flutter:
  dependency: "direct main"
  description: flutter
  source: sdk
  version: "0.0.0"
flutter_animate:
  dependency: transitive
  description:
    name: flutter_animate
    sha256: "7c8a6594a9252dad30cc2ef16e33270b6248c4dedc3b3d06c86c4f3f4dc05ae5"
    url: "https://pub.dev"
  source: hosted
  version: "4.5.0"
flutter_bloc:
  dependency: "direct main"
  description:
    name: flutter_bloc
    sha256: b594505eac31a0518bdcb4b5b79573b8d9117b193cc80cc12e17d639b10aa27a
    url: "https://pub.dev"
  source: hosted
  version: "8.1.6"
flutter_cache_manager:
  dependency: transitive
  description:
    name: flutter_cache_manager
    sha256: "400b6592f16a4409a7f2bb929a9a7e38c72cceb8ffb99ee57bbf2cb2cecf8386"
    url: "https://pub.dev"
  source: hosted
  version: "3.4.1"
flutter_colorpicker:
  dependency: transitive
  description:
    name: flutter_colorpicker
    sha256: "969de5f6f9e2a570ac660fb7b501551451ea2a1ab9e2097e89475f60e07816ea"
    url: "https://pub.dev"
  source: hosted
  version: "1.1.0"
flutter_dotenv:
  dependency: "direct main"
  description:
    name: flutter_dotenv
    sha256: b7c7be5cd9f6ef7a78429cabd2774d3c4af50e79cb2b7593e3d5d763ef95c61b
    url: "https://pub.dev"
  source: hosted
  version: "5.2.1"
```

```
  flutter_emoji:
    dependency: "direct main"
    description:
      name: flutter_emoji
      sha256: "3cc794d2459738e6489cee63f50e85de3732257042f425611d255c8049ce1b81"
      url: "https://pub.dev"
    source: hosted
    version: "2.5.1"
  flutter_highlight:
    dependency: transitive
    description:
      name: flutter_highlight
      sha256: "7b96333867aa07e122e245c033b8ad622e4e3a42a1a2372cbb098a2541d8782c"
      url: "https://pub.dev"
    source: hosted
    version: "0.7.0"
  flutter_hooks:
    dependency: "direct main"
    description:
      name: flutter_hooks
      sha256: cde36b12f7188c85286fba9b38cc5a902e7279f36dd676967106c041dc9dde70
      url: "https://pub.dev"
    source: hosted
    version: "0.20.5"
  flutter_keyboard_visibility:
    dependency: transitive
    description:
      name: flutter_keyboard_visibility
      sha256: "4983655c26ab5b959252ee204c2fffa4afeb4413cd030455194ec0caa3b8e7cb"
      url: "https://pub.dev"
    source: hosted
    version: "5.4.1"
  flutter_keyboard_visibility_linux:
    dependency: transitive
    description:
      name: flutter_keyboard_visibility_linux
      sha256: "6fba7cd9bb033b6ddd8c2beb4c99ad02d728f1e6e6d9b9446667398b2ac39f08"
      url: "https://pub.dev"
    source: hosted
    version: "1.0.0"
  flutter_keyboard_visibility_macos:
    dependency: transitive
    description:
      name: flutter_keyboard_visibility_macos
      sha256: c5c49b16fff453dfdafdc16f26bdd8fb8d55812a1d50b0ce25fc8d9f2e53d086
      url: "https://pub.dev"
    source: hosted
    version: "1.0.0"
  flutter_keyboard_visibility_platform_interface:
    dependency: transitive
    description:
      name: flutter_keyboard_visibility_platform_interface
      sha256: e43a89845873f7be10cb3884345ceb9aebf00a659f479d1c8f4293fcb37022a4
      url: "https://pub.dev"
    source: hosted
    version: "2.0.0"
  flutter_keyboard_visibility_web:
    dependency: transitive
    description:
      name: flutter_keyboard_visibility_web
      sha256: d3771a2e752880c79203f8d80658401d0c998e4183edca05a149f5098ce6e3d1
      url: "https://pub.dev"
```

```
  source: hosted
  version: "2.0.0"
flutter_keyboard_visibility_windows:
  dependency: transitive
  description:
    name: flutter_keyboard_visibility_windows
    sha256: fc4b0f0b6be9b93ae527f3d527fb56ee2d918cd88bbca438c478af7bcfd0ef73
    url: "https://pub.dev"
  source: hosted
  version: "1.0.0"
flutter_lints:
  dependency: "direct dev"
  description:
    name: flutter_lints
    sha256: "9e8c3858111da373efc5aa341de011d9bd23e2c5c5e0c62bccf32438e192d7b1"
    url: "https://pub.dev"
  source: hosted
  version: "3.0.2"
flutter_markdown:
  dependency: "direct main"
  description:
    name: flutter_markdown
    sha256: "04c4722cc36ec5af38acc38ece70d22d3c2123c61305d555750a091517bbe504"
    url: "https://pub.dev"
  source: hosted
  version: "0.6.23"
flutter_quill:
  dependency: "direct main"
  description:
    name: flutter_quill
    sha256: "49070012fad3f6794ca4c1fe76d16f0610ff1be6edee360ab935f95de898dbd1"
    url: "https://pub.dev"
  source: hosted
  version: "8.4.3"
flutter_shaders:
  dependency: transitive
  description:
    name: flutter_shaders
    sha256: "34794acadd8275d971e02df03afee3dee0f98dbfb8c4837082ad0034f612a3e2"
    url: "https://pub.dev"
  source: hosted
  version: "0.1.3"
flutter_staggered_grid_view:
  dependency: "direct main"
  description:
    name: flutter_staggered_grid_view
    sha256: "19e7abb550c96fbfeb546b23f3ff356ee7c59a019a651f8f102a4ba9b7349395"
    url: "https://pub.dev"
  source: hosted
  version: "0.7.0"
flutter_svg:
  dependency: "direct main"
  description:
    name: flutter_svg
    sha256: "7b4ca6cf3304575fe9c8ec64813c8d02ee41d2afe60bcfe0678bcb5375d596a2"
    url: "https://pub.dev"
  source: hosted
  version: "2.0.10+1"
flutter_test:
  dependency: "direct dev"
  description: flutter
  source: sdk
```

```
  version: "0.0.0"
flutter_to_pdf:
  dependency: "direct main"
  description:
    name: flutter_to_pdf
    sha256: "753510410c324dabb4e99d4c9eb8a4810fb5a1e4f67817c19ed2014620234903"
    url: "https://pub.dev"
  source: hosted
  version: "0.1.2"
flutter_web_plugins:
  dependency: transitive
  description: flutter
  source: sdk
  version: "0.0.0"
font_awesome_flutter:
  dependency: "direct main"
  description:
    name: font_awesome_flutter
    sha256: "275ff26905134bcb59417cf60ad979136f1f8257f2f449914b2c3e05bbb4cd6f"
    url: "https://pub.dev"
  source: hosted
  version: "10.7.0"
freezed:
  dependency: "direct main"
  description:
    name: freezed
    sha256: a434911f643466d78462625df76fd9eb13e57348ff43fe1f77bbe909522c67a1
    url: "https://pub.dev"
  source: hosted
  version: "2.5.2"
freezed_annotation:
  dependency: "direct main"
  description:
    name: freezed_annotation
    sha256: c2e2d632dd9b8a2b7751117abcfc2b4888ecfe181bd9fca7170d9ef02e595fe2
    url: "https://pub.dev"
  source: hosted
  version: "2.4.4"
frontend_server_client:
  dependency: transitive
  description:
    name: frontend_server_client
    sha256: f64a0333a82f30b0cca061bc3d143813a486dc086b574bfb233b7c1372427694
    url: "https://pub.dev"
  source: hosted
  version: "4.0.0"
get_it:
  dependency: "direct main"
  description:
    name: get_it
    sha256: d85128a5dae4ea777324730dc65edd9c9f43155c109d5cc0a69cab74139fbac1
    url: "https://pub.dev"
  source: hosted
  version: "7.7.0"
glob:
  dependency: transitive
  description:
    name: glob
    sha256: "0e7014b3b7d4dac1ca4d6114f82bf1782ee86745b9b42a92c9289c23d8a0ab63"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.2"
```

```
glowy_borders:
  dependency: "direct main"
  description:
    name: glowy_borders
    sha256: a0ace5b193047deab00ed2646065b0ef91bf1efef5b3579851bcc6bc7e0748b4
    url: "https://pub.dev"
  source: hosted
  version: "1.0.2"
google_fonts:
  dependency: "direct main"
  description:
    name: google_fonts
    sha256: b1ac0fe2832c9cc95e5e88b57d627c5e68c223b9657f4b96e1487aa9098c7b82
    url: "https://pub.dev"
  source: hosted
  version: "6.2.1"
google_identity_services_web:
  dependency: transitive
  description:
    name: google_identity_services_web
    sha256: "5be191523702ba8d7a01ca97c17fca096822ccf246b0a9f11923a6ded06199b6"
    url: "https://pub.dev"
  source: hosted
  version: "0.3.1+4"
google_sign_in:
  dependency: "direct main"
  description:
    name: google_sign_in
    sha256: "0b8787cb9c1a68ad398e8010e8c8766bfa33556d2ab97c439fb4137756d7308f"
    url: "https://pub.dev"
  source: hosted
  version: "6.2.1"
google_sign_in_android:
  dependency: transitive
  description:
    name: google_sign_in_android
    sha256: "1ad54110a151d3ebbfe9be5a0c2b715aeabddb079e54efd84e56c49605b5474a"
    url: "https://pub.dev"
  source: hosted
  version: "6.1.31"
google_sign_in_ios:
  dependency: transitive
  description:
    name: google_sign_in_ios
    sha256: "83f015169102df1ab2905cf8abd8934e28f87db9ace7a5fa676998842fed228a"
    url: "https://pub.dev"
  source: hosted
  version: "5.7.8"
google_sign_in_platform_interface:
  dependency: transitive
  description:
    name: google_sign_in_platform_interface
    sha256: "1f6e5787d7a120cc0359ddf315c92309069171306242e181c09472d1b00a2971"
    url: "https://pub.dev"
  source: hosted
  version: "2.4.5"
google_sign_in_web:
  dependency: "direct main"
  description:
    name: google_sign_in_web
    sha256: "042805a21127a85b0dc46bba98a37926f17d2439720e8a459d27045d8ef68055"
    url: "https://pub.dev"
```

```
  source: hosted
  version: "0.12.4+2"
graphs:
  dependency: transitive
  description:
    name: graphs
    sha256: "741bbf84165310a68ff28fe9e727332eef1407342fca52759cb21ad8177bb8d0"
    url: "https://pub.dev"
  source: hosted
  version: "2.3.2"
highlight:
  dependency: transitive
  description:
    name: highlight
    sha256: "5353a83ffe3e3eca7df0abfb72dcf3fa66cc56b953728e7113ad4ad88497cf21"
    url: "https://pub.dev"
  source: hosted
  version: "0.7.0"
http:
  dependency: "direct main"
  description:
    name: http
    sha256: b9c29a161230ee03d3ccf545097fccd9b87a5264228c5d348202e0f0c28f9010
    url: "https://pub.dev"
  source: hosted
  version: "1.2.2"
http_multi_server:
  dependency: transitive
  description:
    name: http_multi_server
    sha256: "97486f20f9c2f7be8f514851703d0119c3596d14ea63227af6f7a481ef2b2f8b"
    url: "https://pub.dev"
  source: hosted
  version: "3.2.1"
http_parser:
  dependency: transitive
  description:
    name: http_parser
    sha256: "2aa08ce0341cc9b354a498388e30986515406668dbcc4f7c950c3e715496693b"
    url: "https://pub.dev"
  source: hosted
  version: "4.0.2"
i18n_extension:
  dependency: transitive
  description:
    name: i18n_extension
    sha256: db45cd88cf3114f5b9368d975aebebe4ac37fa634fbc5643634289cdfd4d3631
    url: "https://pub.dev"
  source: hosted
  version: "9.0.2"
image:
  dependency: transitive
  description:
    name: image
    sha256: f31d52537dc417fdcde36088fdf11d191026fd5e4fae742491ebd40e5a8bea7d
    url: "https://pub.dev"
  source: hosted
  version: "4.3.0"
intl:
  dependency: "direct main"
  description:
    name: intl
```

```
    sha256: "3bc132a9dbce73a7e4a21a17d06e1878839ffbf975568bc875c60537824b0c4d"
    url: "https://pub.dev"
  source: hosted
  version: "0.18.1"
io:
  dependency: transitive
  description:
    name: io
    sha256: "2ec25704aba361659e10e3e5f5d672068d332fc8ac516421d483a11e5cbd061e"
    url: "https://pub.dev"
  source: hosted
  version: "1.0.4"
js:
  dependency: transitive
  description:
    name: js
    sha256: f2c445dce49627136094980615a031419f7f3eb393237e4ecd97ac15dea343f3
    url: "https://pub.dev"
  source: hosted
  version: "0.6.7"
json_annotation:
  dependency: "direct main"
  description:
    name: json_annotation
    sha256: "1ce844379ca14835a50d2f019a3099f419082cfdd231cd86a142af94dd5c6bb1"
    url: "https://pub.dev"
  source: hosted
  version: "4.9.0"
json_serializable:
  dependency: "direct dev"
  description:
    name: json_serializable
    sha256: ea1432d167339ea9b5bb153f0571d0039607a873d6e04e0117af043f14a1fd4b
    url: "https://pub.dev"
  source: hosted
  version: "6.8.0"
leak_tracker:
  dependency: transitive
  description:
    name: leak_tracker
    sha256: "7f0df31977cb2c0b88585095d168e689669a2cc9b97c309665e3386f3e9d341a"
    url: "https://pub.dev"
  source: hosted
  version: "10.0.4"
leak_tracker_flutter_testing:
  dependency: transitive
  description:
    name: leak_tracker_flutter_testing
    sha256: "06e98f569d004c1315b991ded39924b21af84cf14cc94791b8aea337d25b57f8"
    url: "https://pub.dev"
  source: hosted
  version: "3.0.3"
leak_tracker_testing:
  dependency: transitive
  description:
    name: leak_tracker_testing
    sha256: "6ba465d5d76e67ddf503e1161d1f4a6bc42306f9d66ca1e8f079a47290fb06d3"
    url: "https://pub.dev"
  source: hosted
  version: "3.0.1"
lints:
  dependency: transitive
```

```
  description:
    name: lints
    sha256: cbf8d4b858bb0134ef3ef87841abdf8d63bfc255c266b7bf6b39daa1085c4290
    url: "https://pub.dev"
  source: hosted
  version: "3.0.0"
logging:
  dependency: transitive
  description:
    name: logging
    sha256: c8245ada5f1717ed44271ed1c26b8ce85ca3228fd2ffdb75468ab01979309d61
    url: "https://pub.dev"
  source: hosted
  version: "1.3.0"
lottie:
  dependency: "direct main"
  description:
    name: lottie
    sha256: a93542cc2d60a7057255405f62252533f8e8956e7e06754955669fd32fb4b216
    url: "https://pub.dev"
  source: hosted
  version: "2.7.0"
mailto:
  dependency: "direct main"
  description:
    name: mailto
    sha256: f8c5ce39e0eaa94a856795b2855af7f66aac37f7c3b70ac5c26ab00b94685445
    url: "https://pub.dev"
  source: hosted
  version: "2.0.0"
markdown:
  dependency: "direct main"
  description:
    name: markdown
    sha256: ef2a1298144e3f985cc736b22e0ccdaf188b5b3970648f2d9dc13efd1d9df051
    url: "https://pub.dev"
  source: hosted
  version: "7.2.2"
markdown_widget:
  dependency: "direct main"
  description:
    name: markdown_widget
    sha256: "216dced98962d7699a265344624bc280489d739654585ee881c95563a3252fac"
    url: "https://pub.dev"
  source: hosted
  version: "2.3.2+6"
matcher:
  dependency: transitive
  description:
    name: matcher
    sha256: d2323aa2060500f906aa31a895b4030b6da3ebdcc5619d14ce1aada65cd161cb
    url: "https://pub.dev"
  source: hosted
  version: "0.12.16+1"
material_color_utilities:
  dependency: transitive
  description:
    name: material_color_utilities
    sha256: "0e0a020085b65b6083975e499759762399b4475f766c21668c4ecca34ea74e5a"
    url: "https://pub.dev"
  source: hosted
  version: "0.8.0"
```

```
meta:
  dependency: transitive
  description:
    name: meta
    sha256: "7687075e408b093f36e6bbf6c91878cc0d4cd10f409506f7bc996f68220b9136"
    url: "https://pub.dev"
  source: hosted
  version: "1.12.0"
mime:
  dependency: transitive
  description:
    name: mime
    sha256: "801fd0b26f14a4a58ccb09d5892c3fbdeff209594300a542492cf13fba9d247a"
    url: "https://pub.dev"
  source: hosted
  version: "1.0.6"
multi_split_view:
  dependency: "direct main"
  description:
    name: multi_split_view
    sha256: d68e129bff71fc9e6b66de59e1b79deaf4b91f30940130bfbd2d704c1c713499
    url: "https://pub.dev"
  source: hosted
  version: "2.4.0"
nested:
  dependency: transitive
  description:
    name: nested
    sha256: "03bac4c528c64c95c722ec99280375a6f2fc708eec17c7b3f07253b626cd2a20"
    url: "https://pub.dev"
  source: hosted
  version: "1.0.0"
octo_image:
  dependency: transitive
  description:
    name: octo_image
    sha256: "34faa6639a78c7e3cbe79be6f9f96535867e879748ade7d17c9b1ae7536293bd"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.0"
package_config:
  dependency: transitive
  description:
    name: package_config
    sha256: "1c5b77ccc91e4823a5af61ee74e6b972db1ef98c2ff5a18d3161c982a55448bd"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.0"
package_info_plus:
  dependency: "direct main"
  description:
    name: package_info_plus
    sha256: df3eb3e0aed5c1107bb0fdb80a8e82e778114958b1c5ac5644fb1ac9cae8a998
    url: "https://pub.dev"
  source: hosted
  version: "8.1.0"
package_info_plus_platform_interface:
  dependency: transitive
  description:
    name: package_info_plus_platform_interface
    sha256: ac1f4a4847f1ade8e6a87d1f39f5d7c67490738642e2542f559ec38c37489a66
    url: "https://pub.dev"
```

```
  source: hosted
  version: "3.0.1"
pasteboard:
  dependency: transitive
  description:
    name: pasteboard
    sha256: "1c8b6a8b3f1d12e55d4e9404433cda1b4abe66db6b17bc2d2fb5965772c04674"
    url: "https://pub.dev"
  source: hosted
  version: "0.2.0"
path:
  dependency: transitive
  description:
    name: path
    sha256: "087ce49c3f0dc39180befefc60fdb4acd8f8620e5682fe2476afd0b3688bb4af"
    url: "https://pub.dev"
  source: hosted
  version: "1.9.0"
path_drawing:
  dependency: transitive
  description:
    name: path_drawing
    sha256: bbb1934c0cbb03091af082a6389ca2080345291ef07a5fa6d6e078ba8682f977
    url: "https://pub.dev"
  source: hosted
  version: "1.0.1"
path_parsing:
  dependency: transitive
  description:
    name: path_parsing
    sha256: "45f7d6bba1128761de5540f39d5ca000ea8a1f22f06b76b61094a60a2997bd0e"
    url: "https://pub.dev"
  source: hosted
  version: "1.0.2"
path_provider:
  dependency: transitive
  description:
    name: path_provider
    sha256: fec0d61223fba3154d87759e3cc27fe2c8dc498f6386c6d6fc80d1afdd1bf378
    url: "https://pub.dev"
  source: hosted
  version: "2.1.4"
path_provider_android:
  dependency: transitive
  description:
    name: path_provider_android
    sha256: "6f01f8e37ec30b07bc424b4deabac37cacb1bc7e2e515ad74486039918a37eb7"
    url: "https://pub.dev"
  source: hosted
  version: "2.2.10"
path_provider_foundation:
  dependency: transitive
  description:
    name: path_provider_foundation
    sha256: f234384a3fdd67f989b4d54a5d73ca2a6c422fa55ae694381ae0f4375cd1ea16
    url: "https://pub.dev"
  source: hosted
  version: "2.4.0"
path_provider_linux:
  dependency: transitive
  description:
    name: path_provider_linux
```

```
    sha256: f7a1fe3a634fe7734c8d3f2766ad746ae2a2884abe22e241a8b301bf5cac3279
    url: "https://pub.dev"
  source: hosted
  version: "2.2.1"
path_provider_platform_interface:
  dependency: transitive
  description:
    name: path_provider_platform_interface
    sha256: "88f5779f72ba699763fa3a3b06aa4bf6de76c8e5de842cf6f29e2e06476c2334"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.2"
path_provider_windows:
  dependency: transitive
  description:
    name: path_provider_windows
    sha256: bd6f00dbd873bfb70d0761682da2b3a2c2fccc2b9e84c495821639601d81afe7
    url: "https://pub.dev"
  source: hosted
  version: "2.3.0"
pdf:
  dependency: "direct main"
  description:
    name: pdf
    sha256: "05df53f8791587402493ac97b9869d3824eccbc77d97855f4545cf72df3cae07"
    url: "https://pub.dev"
  source: hosted
  version: "3.11.1"
petitparser:
  dependency: transitive
  description:
    name: petitparser
    sha256: c15605cd28af66339f8eb6fbe0e541bfe2d1b72d5825efc6598f3e0a31b9ad27
    url: "https://pub.dev"
  source: hosted
  version: "6.0.2"
platform:
  dependency: transitive
  description:
    name: platform
    sha256: "5d6b1b0036a5f331ebc77c850ebc8506cbc1e9416c27e59b439f917a902a4984"
    url: "https://pub.dev"
  source: hosted
  version: "3.1.6"
plugin_platform_interface:
  dependency: transitive
  description:
    name: plugin_platform_interface
    sha256: "4820fbfdb9478b1ebae27888254d445073732dae3d6ea81f0b7e06d5dedc3f02"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.8"
pool:
  dependency: transitive
  description:
    name: pool
    sha256: "20fe868b6314b322ea036ba325e6fc0711a22948856475e2c2b6306e8ab39c2a"
    url: "https://pub.dev"
  source: hosted
  version: "1.5.1"
popover:
  dependency: "direct main"
```

```
  description:
    name: popover
    sha256: "5cba40e04115cbbf15c35e00767b91e8bf3f769763a34beb2f8a1b9e8b5fc876"
    url: "https://pub.dev"
  source: hosted
  version: "0.3.0+1"
provider:
  dependency: transitive
  description:
    name: provider
    sha256: c8a055ee5ce3fd98d6fc872478b03823ffdb448699c6ebdbbc71d59b596fd48c
    url: "https://pub.dev"
  source: hosted
  version: "6.1.2"
pub_semver:
  dependency: transitive
  description:
    name: pub_semver
    sha256: "40d3ab1bbd474c4c2328c91e3a7df8c6dd629b79ece4c4bd04bee496a224fb0c"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.4"
pubspec_parse:
  dependency: transitive
  description:
    name: pubspec_parse
    sha256: c799b721d79eb6ee6fa56f00c04b472dcd44a30d258fac2174a6ec57302678f8
    url: "https://pub.dev"
  source: hosted
  version: "1.3.0"
qr:
  dependency: transitive
  description:
    name: qr
    sha256: "5a1d2586170e172b8a8c8470bbbffd5eb0cd38a66c0d77155ea138d3af3a4445"
    url: "https://pub.dev"
  source: hosted
  version: "3.0.2"
quiver:
  dependency: transitive
  description:
    name: quiver
    sha256: ea0b925899e64ecdfbf9c7becb60d5b50e706ade44a85b2363be2a22d88117d2
    url: "https://pub.dev"
  source: hosted
  version: "3.2.2"
rxdart:
  dependency: transitive
  description:
    name: rxdart
    sha256: "5c3004a4a8dbb94bd4bf5412a4def4acdaa12e12f269737a5751369e12d1a962"
    url: "https://pub.dev"
  source: hosted
  version: "0.28.0"
screenshot:
  dependency: "direct main"
  description:
    name: screenshot
    sha256: "63817697a7835e6ce82add4228e15d233b74d42975c143ad8cfe07009fab866b"
    url: "https://pub.dev"
  source: hosted
  version: "3.0.0"
```

```
  scroll_to_index:
    dependency: transitive
    description:
      name: scroll_to_index
      sha256: b707546e7500d9f070d63e5acf74fd437ec7eeeb68d3412ef7b0afada0b4f176
      url: "https://pub.dev"
    source: hosted
    version: "3.0.1"
  shelf:
    dependency: transitive
    description:
      name: shelf
      sha256: ad29c505aee705f41a4d8963641f91ac4cee3c8fad5947e033390a7bd8180fa4
      url: "https://pub.dev"
    source: hosted
    version: "1.4.1"
  shelf_static:
    dependency: transitive
    description:
      name: shelf_static
      sha256: c87c3875f91262785dade62d135760c2c69cb217ac759485334c5857ad89f6e3
      url: "https://pub.dev"
    source: hosted
    version: "1.1.3"
  shelf_web_socket:
    dependency: transitive
    description:
      name: shelf_web_socket
      sha256: "073c147238594ecd0d193f3456a5fe91c4b0abbcc68bf5cd95b36c4e194ac611"
      url: "https://pub.dev"
    source: hosted
    version: "2.0.0"
  sky_engine:
    dependency: transitive
    description: flutter
    source: sdk
    version: "0.0.99"
  source_gen:
    dependency: transitive
    description:
      name: source_gen
      sha256: "14658ba5f669685cd3d63701d01b31ea748310f7ab854e471962670abcf57832"
      url: "https://pub.dev"
    source: hosted
    version: "1.5.0"
  source_helper:
    dependency: transitive
    description:
      name: source_helper
      sha256: "6adebc0006c37dd63fe05bca0a929b99f06402fc95aa35bf36d67f5c06de01fd"
      url: "https://pub.dev"
    source: hosted
    version: "1.3.4"
  source_span:
    dependency: transitive
    description:
      name: source_span
      sha256: "53e943d4206a5e30df338fd4c6e7a077e02254531b138a15aec3bd143c1a8b3c"
      url: "https://pub.dev"
    source: hosted
    version: "1.10.0"
  sprintf:
```

```
  dependency: transitive
  description:
    name: sprintf
    sha256: "1fc9ffe69d4df602376b52949af107d8f5703b77cda567c4d7d86a0693120f23"
    url: "https://pub.dev"
  source: hosted
  version: "7.0.0"
sqflite:
  dependency: transitive
  description:
    name: sqflite
    sha256: a43e5a27235518c03ca238e7b4732cf35eabe863a369ceba6cbefa537a66f16d
    url: "https://pub.dev"
  source: hosted
  version: "2.3.3+1"
sqflite_common:
  dependency: transitive
  description:
    name: sqflite_common
    sha256: "3da423ce7baf868be70e2c0976c28a1bb2f73644268b7ffa7d2e08eab71f16a4"
    url: "https://pub.dev"
  source: hosted
  version: "2.5.4"
stack_trace:
  dependency: transitive
  description:
    name: stack_trace
    sha256: "73713990125a6d93122541237550ee3352a2d84baad52d375a4cad2eb9b7ce0b"
    url: "https://pub.dev"
  source: hosted
  version: "1.11.1"
stream_channel:
  dependency: transitive
  description:
    name: stream_channel
    sha256: ba2aa5d8cc609d96bbb2899c28934f9e1af5cddbd60a827822ea467161eb54e7
    url: "https://pub.dev"
  source: hosted
  version: "2.1.2"
stream_transform:
  dependency: transitive
  description:
    name: stream_transform
    sha256: "14a00e794c7c11aa145a170587321aedce29769c08d7f58b1d141da75e3b1c6f"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.0"
string_scanner:
  dependency: transitive
  description:
    name: string_scanner
    sha256: "556692adab6cfa87322a115640c11f13cb77b3f076ddcc5d6ae3c20242bedcde"
    url: "https://pub.dev"
  source: hosted
  version: "1.2.0"
synchronized:
  dependency: transitive
  description:
    name: synchronized
    sha256: "539ef412b170d65ecdafd780f924e5be3f60032a1128df156adad6c5b373d558"
    url: "https://pub.dev"
  source: hosted
```

```
  version: "3.1.0+1"
term_glyph:
  dependency: transitive
  description:
    name: term_glyph
    sha256: a29248a84fbb7c79282b40b8c72a1209db169a2e0542bce341da992fe1bc7e84
    url: "https://pub.dev"
  source: hosted
  version: "1.2.1"
test_api:
  dependency: transitive
  description:
    name: test_api
    sha256: "9955ae474176f7ac8ee4e989dadfb411a58c30415bcfb648fa04b2b8a03afa7f"
    url: "https://pub.dev"
  source: hosted
  version: "0.7.0"
timeago:
  dependency: "direct main"
  description:
    name: timeago
    sha256: "054cedf68706bb142839ba0ae6b135f6b68039f0b8301cbe8784ae653d5ff8de"
    url: "https://pub.dev"
  source: hosted
  version: "3.7.0"
timing:
  dependency: transitive
  description:
    name: timing
    sha256: "70a3b636575d4163c477e6de42f247a23b315ae20e86442bebe32d3cabf61c32"
    url: "https://pub.dev"
  source: hosted
  version: "1.0.1"
typed_data:
  dependency: transitive
  description:
    name: typed_data
    sha256: facc8d6582f16042dd49f2463ff1bd6e2c9ef9f3d5da3d9b087e244a7b564b3c
    url: "https://pub.dev"
  source: hosted
  version: "1.3.2"
url_launcher:
  dependency: "direct main"
  description:
    name: url_launcher
    sha256: "9d06212b1362abc2f0f0d78e6f09f726608c74e3b9462e8368bb03314aa8d603"
    url: "https://pub.dev"
  source: hosted
  version: "6.3.1"
url_launcher_android:
  dependency: transitive
  description:
    name: url_launcher_android
    sha256: f0c73347dfcfa5b3db8bc06e1502668265d39c08f310c29bff4e28eea9699f79
    url: "https://pub.dev"
  source: hosted
  version: "6.3.9"
url_launcher_ios:
  dependency: transitive
  description:
    name: url_launcher_ios
    sha256: e43b677296fadce447e987a2f519dcf5f6d1e527dc35d01ffab4fff5b8a7063e
```

```
    url: "https://pub.dev"
  source: hosted
  version: "6.3.1"
url_launcher_linux:
  dependency: transitive
  description:
    name: url_launcher_linux
    sha256: e2b9622b4007f97f504cd64c0128309dfb978ae66adbe944125ed9e1750f06af
    url: "https://pub.dev"
  source: hosted
  version: "3.2.0"
url_launcher_macos:
  dependency: transitive
  description:
    name: url_launcher_macos
    sha256: "769549c999acdb42b8bcfa7c43d72bf79a382ca7441ab18a808e101149daf672"
    url: "https://pub.dev"
  source: hosted
  version: "3.2.1"
url_launcher_platform_interface:
  dependency: transitive
  description:
    name: url_launcher_platform_interface
    sha256: "552f8a1e663569be95a8190206a38187b531910283c3e982193e4f2733f01029"
    url: "https://pub.dev"
  source: hosted
  version: "2.3.2"
url_launcher_web:
  dependency: transitive
  description:
    name: url_launcher_web
    sha256: "772638d3b34c779ede05ba3d38af34657a05ac55b06279ea6edd409e323dca8e"
    url: "https://pub.dev"
  source: hosted
  version: "2.3.3"
url_launcher_windows:
  dependency: transitive
  description:
    name: url_launcher_windows
    sha256: "44cf3aabcedde30f2dba119a9dea3b0f2672fbe6fa96e85536251d678216b3c4"
    url: "https://pub.dev"
  source: hosted
  version: "3.1.3"
uuid:
  dependency: "direct main"
  description:
    name: uuid
    sha256: a5be9ef6618a7ac1e964353ef476418026db906c4facdedaa299b7a2e71690ff
    url: "https://pub.dev"
  source: hosted
  version: "4.5.1"
vector_graphics:
  dependency: transitive
  description:
    name: vector_graphics
    sha256: "32c3c684e02f9bc0afb0ae0aa653337a2fe022e8ab064bcd7ffda27a74e288e3"
    url: "https://pub.dev"
  source: hosted
  version: "1.1.11+1"
vector_graphics_codec:
  dependency: transitive
  description:
```

```
    name: vector_graphics_codec
    sha256: c86987475f162fadff579e7320c7ddda04cd2fdeffbe1129227a85d9ac9e03da
    url: "https://pub.dev"
  source: hosted
  version: "1.1.11+1"
vector_graphics_compiler:
  dependency: transitive
  description:
    name: vector_graphics_compiler
    sha256: "12faff3f73b1741a36ca7e31b292ddeb629af819ca9efe9953b70bd63fc8cd81"
    url: "https://pub.dev"
  source: hosted
  version: "1.1.11+1"
vector_math:
  dependency: transitive
  description:
    name: vector_math
    sha256: "80b3257d1492ce4d091729e3a67a60407d227c27241d6927be0130c98e741803"
    url: "https://pub.dev"
  source: hosted
  version: "2.1.4"
visibility_detector:
  dependency: transitive
  description:
    name: visibility_detector
    sha256: dd5cc11e13494f432d15939c3aa8ae76844c42b723398643ce9addb88a5ed420
    url: "https://pub.dev"
  source: hosted
  version: "0.4.0+2"
vm_service:
  dependency: transitive
  description:
    name: vm_service
    sha256: "3923c89304b715fb1eb6423f017651664a03bf5f4b29983627c4da791f74a4ec"
    url: "https://pub.dev"
  source: hosted
  version: "14.2.1"
watcher:
  dependency: transitive
  description:
    name: watcher
    sha256: "3d2ad6751b3c16cf07c7fca317a1413b3f26530319181b37e3b9039b84fc01d8"
    url: "https://pub.dev"
  source: hosted
  version: "1.1.0"
web:
  dependency: transitive
  description:
    name: web
    sha256: "97da13628db363c635202ad97068d47c5b8aa555808e7a9411963c533b449b27"
    url: "https://pub.dev"
  source: hosted
  version: "0.5.1"
web_socket:
  dependency: transitive
  description:
    name: web_socket
    sha256: "3c12d96c0c9a4eec095246debcea7b86c0324f22df69893d538fcc6f1b8cce83"
    url: "https://pub.dev"
  source: hosted
  version: "0.1.6"
web_socket_channel:
```

```
    dependency: transitive
    description:
      name: web_socket_channel
      sha256: "9f187088ed104edd8662ca07af4b124465893caf063ba29758f97af57e61da8f"
      url: "https://pub.dev"
    source: hosted
    version: "3.0.1"
  win32:
    dependency: transitive
    description:
      name: win32
      sha256: "68d1e89a91ed61ad9c370f9f8b6effed9ae5e0ede22a270bdfa6daf79fc2290a"
      url: "https://pub.dev"
    source: hosted
    version: "5.5.4"
  win32_registry:
    dependency: transitive
    description:
      name: win32_registry
      sha256: "21ec76dfc731550fd3e2ce7a33a9ea90b828fdf19a5c3bcf556fa992cfa99852"
      url: "https://pub.dev"
    source: hosted
    version: "1.1.5"
  xdg_directories:
    dependency: transitive
    description:
      name: xdg_directories
      sha256: "7a3f37b05d989967cdddcbb571f1ea834867ae2faa29725fd085180e0883aa15"
      url: "https://pub.dev"
    source: hosted
    version: "1.1.0"
  xml:
    dependency: transitive
    description:
      name: xml
      sha256: b015a8ad1c488f66851d762d3090a21c600e479dc75e68328c52774040cf9226
      url: "https://pub.dev"
    source: hosted
    version: "6.5.0"
  yaml:
    dependency: transitive
    description:
      name: yaml
      sha256: "75769501ea3489fca56601ff33454fe45507ea3bfb014161abc3b43ae25989d5"
      url: "https://pub.dev"
    source: hosted
    version: "3.1.2"
sdks:
  dart: ">=3.4.0 <4.0.0"
  flutter: ">=3.22.0"
```

```
{
  "hosting": {
    "rewrites": [
      {
        "source": "/**",
        "destination": "/index.html"
      }
    ],
    "headers": [
      {
        "source": "/icons/**",
        "headers": [
          {
            "key": "Cache-Control",
            "value": "public, max-age=31536000"
          }
        ]
      },
      {
        "source": "/assets/**",
        "headers": [
          {
            "key": "Cache-Control",
            "value": "public, max-age=31536000"
          }
        ]
      }
    ],
    "source": ".",
    "ignore": [
      "firebase.json",
      "**/.*",
      "**/node_modules/**"
    ],
    "frameworksBackend": {
      "region": "europe-west1"
    }
  }
}
```

```
targets:
  $default:
    builders:
      json_serializable:
        options:
          explicit_to_json: true
```

```
extensions:
  - provider: true
```

```
# Miscellaneous
*.class
*.log
*.pyc
*.swp
.DS_Store
.atom/
.buildlog/
.history
.svn/
migrate_working_dir/

# IntelliJ related
*.iml
*.ipr
*.iws
.idea/

# The .vscode folder contains launch configuration and tasks you configure in
# VS Code which you may wish to be included in version control, so this line
# is commented out by default.
#.vscode/

# Flutter/Dart/Pub related
**/doc/api/
**/ios/Flutter/.last_build_id
.dart_tool/
.flutter-plugins
.flutter-plugins-dependencies
.packages
.pub-cache/
.pub/

# Symbolication related
app.*.symbols

# Obfuscation related
app.*.map.json

# Android Studio will place build artifacts here
/android/app/debug
/android/app/profile
/android/app/release
/build/3c6e5cf9bd2369c57f1061727dea4d2d.cache.dill.track.dill
/build/4f2a58af27abc49ecf55570043bbf938.cache.dill.track.dill
/build/02937c812a0f2091606219ae0b6b7efc.cache.dill.track.dill
/build/225a8c8a7ed1097aff942637eb31fa37/_composite.stamp
/build/c91cf747aeec407f5049ae21c76ef551/_composite.stamp
/build/225a8c8a7ed1097aff942637eb31fa37/gen_dart_plugin_registrant.stamp
/build/c91cf747aeec407f5049ae21c76ef551/gen_dart_plugin_registrant.stamp
/build/225a8c8a7ed1097aff942637eb31fa37/gen_localizations.stamp
/build/c91cf747aeec407f5049ae21c76ef551/gen_localizations.stamp
/build/last_build_run.json
/build/
/.firebase/
/assets/.env
/dotenv.env
/report.xml
.env
.firebase/
/build
build/
```

```
# This file configures the analyzer, which statically analyzes Dart code to
# check for errors, warnings, and lints.
#
# The issues identified by the analyzer are surfaced in the UI of Dart-enabled
# IDEs (https://dart.dev/tools#ides-and-editors). The analyzer can also be
# invoked from the command line by running `flutter analyze`.

# The following line activates a set of recommended lints for Flutter apps,
# packages, and plugins designed to encourage good coding practices.
include: package:flutter_lints/flutter.yaml

linter:
  rules:
    - always_declare_return_types
    - always_put_control_body_on_new_line
    - always_put_required_named_parameters_first
    - always_require_non_null_named_parameters
    - always_use_package_imports
    - annotate_overrides
    - avoid_annotating_with_dynamic
    - avoid_bool_literals_in_conditional_expressions
    - avoid_catches_without_on_clauses
    - avoid_catching_errors
    - avoid_classes_with_only_static_members
    - avoid_double_and_int_checks
    - avoid_dynamic_calls
    - avoid_empty_else
    - avoid_equals_and_hash_code_on_mutable_classes
    - avoid_escaping_inner_quotes
    - avoid_field_initializers_in_const_classes
    - avoid_function_literals_in_foreach_calls
    - avoid_implementing_value_types
    - avoid_init_to_null
    - avoid_js_rounded_ints
    - avoid_multiple_declarations_per_line
    - avoid_null_checks_in_equality_operators
    - avoid_positional_boolean_parameters
    - avoid_print
    - avoid_private_typedef_functions
    - avoid_redundant_argument_values
    - avoid_relative_lib_imports
    - avoid_renaming_method_parameters
    - avoid_return_types_on_setters
    - avoid_returning_null
    - avoid_returning_null_for_future
    - avoid_returning_null_for_void
    - avoid_returning_this
    - avoid_setters_without_getters
    - avoid_shadowing_type_parameters
    - avoid_single_cascade_in_expression_statements
    - avoid_slow_async_io
    - avoid_type_to_string
    - avoid_types_as_parameter_names
    - avoid_types_on_closure_parameters
    - avoid_unnecessary_containers
    - avoid_unused_constructor_parameters
    - avoid_void_async
    - avoid_web_libraries_in_flutter
    - await_only_futures
    - camel_case_extensions
    - camel_case_types
    - cancel_subscriptions
```

```
- cascade_invocations
- cast_nullable_to_non_nullable
- close_sinks
- comment_references
- constant_identifier_names
- control_flow_in_finally
- curly_braces_in_flow_control_structures
- depend_on_referenced_packages
- deprecated_consistency
- directives_ordering
- do_not_use_environment
- empty_catches
- empty_constructor_bodies
- empty_statements
- exhaustive_cases
- file_names
- flutter_style_todos
- hash_and_equals
- implementation_imports
- invariant_booleans
- iterable_contains_unrelated_type
- join_return_with_assignment
- leading_newlines_in_multiline_strings
- library_names
- library_prefixes
- library_private_types_in_public_api
- lines_longer_than_80_chars
- list_remove_unrelated_type
- literal_only_boolean_expressions
- missing_whitespace_between_adjacent_strings
- no_adjacent_strings_in_list
- no_default_cases
- no_duplicate_case_values
- no_logic_in_create_state
- no_runtimeType_toString
- non_constant_identifier_names
- noop_primitive_operations
- null_check_on_nullable_type_parameter
- null_closures
- omit_local_variable_types
- one_member_abstracts
- only_throw_errors
- overridden_fields
- package_api_docs
- package_names
- package_prefixed_library_names
- parameter_assignments
- prefer_adjacent_string_concatenation
- prefer_asserts_in_initializer_lists
- prefer_asserts_with_message
- prefer_collection_literals
- prefer_conditional_assignment
- prefer_const_constructors
- prefer_const_constructors_in_immutables
- prefer_const_declarations
- prefer_const_literals_to_create_immutables
- prefer_constructors_over_static_methods
- prefer_contains
- prefer_double_quotes
- prefer_equal_for_default_values
- prefer_expression_function_bodies
- prefer_final_fields
```

```
    - prefer_final_in_for_each
    - prefer_final_locals
    - prefer_final_parameters
    - prefer_for_elements_to_map_fromIterable
    - prefer_foreach
    - prefer_function_declarations_over_variables
    - prefer_generic_function_type_aliases
    - prefer_if_elements_to_conditional_expressions
    - prefer_if_null_operators
    - prefer_initializing_formals
    - prefer_inlined_adds
    - prefer_int_literals
    - prefer_interpolation_to_compose_strings
    - prefer_is_empty
    - prefer_is_not_empty
    - prefer_is_not_operator
    - prefer_iterable_whereType
    - prefer_mixin
    - prefer_null_aware_method_calls
    - prefer_null_aware_operators
    - prefer_spread_collections
    - prefer_typing_uninitialized_variables
    - prefer_void_to_null
    - provide_deprecation_message
#    - public_member_api_docs
    - recursive_getters
    - require_trailing_commas
    - sized_box_for_whitespace
    - slash_for_doc_comments
    - sort_child_properties_last
    - sort_constructors_first
    - sort_pub_dependencies
    - sort_unnamed_constructors_first
    - test_types_in_equals
    - throw_in_finally
    - tighten_type_of_initializing_formals
    - type_annotate_public_apis
    - type_init_formals
    - unawaited_futures
    - unnecessary_await_in_return
    - unnecessary_brace_in_string_interps
    - unnecessary_const
    - unnecessary_getters_setters
    - unnecessary_lambdas
    - unnecessary_new
    - unnecessary_null_aware_assignments
    - unnecessary_null_checks
    - unnecessary_null_in_if_null_operators
    - unnecessary_nullable_for_final_variable_declarations
    - unnecessary_overrides
    - unnecessary_parenthesis
    - unnecessary_raw_strings
    - unnecessary_statements
    - unnecessary_string_escapes
    - unnecessary_string_interpolations
    - unnecessary_this
    - unrelated_type_equality_checks
    - unsafe_html
    - use_build_context_synchronously
    - use_full_hex_values_for_flutter_colors
    - use_function_type_syntax_for_parameters
    - use_if_null_to_convert_nulls_to_bools
```

```
- use_is_even_rather_than_modulo
- use_key_in_widget_constructors
- use_late_for_private_fields_and_variables
- use_named_constants
- use_raw_strings
- use_rethrow_when_possible
- use_setters_to_change_properties
- use_string_buffers
- use_test_throws_matchers
- use_to_and_as_if_applicable
- valid_regexps
- void_checks
```

```
#!/bin/bash

# reset_version.sh

# Save the current directory
current_dir=$(pwd)

# Change to the root directory where pubspec.yaml is located
cd ..

# Path to the fruits.txt list in the root directory
fruits_file="fastlane/fruits.txt"

# If current version is not in the list, start from the first fruit
default_version="1.0.0"
default_fruit=$(head -n 1 $fruits_file)
default_build="0"

#!/bin/bash


# Default version format
new_version="${default_version}-${default_fruit}+${default_build}"

# Update the version in pubspec.yaml to the default value
sed -i '' "s/version: .*/version: $new_version/g" pubspec.yaml

# Print the new version
echo "Version reset to $new_version"

# Return to the original directory
cd "$current_dir"
```

```
# This file contains the fastlane.tools configuration
# You can find the documentation at https://docs.fastlane.tools
#
# For a list of all available actions, check out
#
#     https://docs.fastlane.tools/actions
#
# Uncomment the line if you want fastlane to automatically update itself
# update_fastlane

default_platform(:web)

platform :web do
  desc "Deploy new version build"
  lane :deploy do
    # add actions here: https://docs.fastlane.tools/actions
      sh("./fruit_version_release.sh")
      sh("./fruit_version_build.sh")
      sh("flutter build web")
      sh("firebase deploy")
  end

  desc "Update current version build"
  lane :deployCurrent do
      # add actions here: https://docs.fastlane.tools/actions
      sh("./fruit_version_build.sh")
      sh("flutter build web")
      sh("firebase deploy")
  end


desc "Deploy current version build to Demo with a random channel"
lane :deployCurrentToDemo do
    # Your existing actions
    sh("./fruit_version_build.sh")
    sh("flutter build web")
    # Generate a random channel ID using shell commands
    channel_id = sh("echo $(date +%s)$(openssl rand -hex 5)").strip
    sh("firebase hosting:channel:deploy #{channel_id}")
end
  desc "Print current build version"
  lane :printCurrent do
        # add actions here: https://docs.fastlane.tools/actions
      sh("./fruit_print_version.sh")
  end
end
```

```
#!/bin/bash

# Save the current directory and move to the root directory
current_dir=$(pwd)
cd ..

# Path to the fruits list in the root directory
fruits_file="${current_dir}/fruits.txt"

# Extract the current version, fruit name, and build number from pubspec.yaml
version_line=$(grep 'version:' pubspec.yaml)
current_fruit=$(echo $version_line | sed 's/version: //' | sed 's/.*-//' | sed 's/+.*//')
build_number=$(echo $version_line | sed 's/.*+//')

# Increment the major version
current_major=$(echo $version_line | sed 's/version: //' | sed 's/\..*//')
major=$((current_major + 1))

# Get the next fruit name
next_fruit=""
found_current=0
while read fruit; do
    if [ "$found_current" -eq 1 ]; then
        next_fruit=$fruit
        break
    fi
    if [ "$fruit" = "$current_fruit" ]; then
        found_current=1
    fi
done < $fruits_file

# Check if next fruit is found, otherwise start from the first fruit
if [ -z "$next_fruit" ]; then
    next_fruit=$(head -n 1 $fruits_file)
fi

# Combine the new version, fruit name, and build number
new_version="$major.0.0-$next_fruit+$build_number"

# Update the version in pubspec.yaml
sed -i '' "s/version: .*/version: $new_version/g" pubspec.yaml

# Print the new version
echo "Version updated to $new_version"

# Return to the original directory
cd "$current_dir"
```

```
#!/bin/bash

# increment_build_number.sh

# Save the current directory and move to the root directory
current_dir=$(pwd)
cd ..

# Extract the current version, fruit name, and build number from pubspec.yaml
version_line=$(grep 'version:' pubspec.yaml)
version=$(echo $version_line | sed 's/version: //' | sed 's/+.*//')
build_number=$(echo $version_line | sed 's/.*+//')

# Increment the build number
new_build_number=$((build_number + 1))

# Update the version in pubspec.yaml
new_version="${version}+${new_build_number}"
sed -i '' "s/version: .*/version: $new_version/g" pubspec.yaml

# Print the new version
echo "Build number updated to $new_version"

# Return to the original directory
cd "$current_dir"
```

```
#!/bin/bash

# print_current_version.sh

# Save the current directory and move to the root directory
current_dir=$(pwd)
cd ..

# Extract and print the current version from pubspec.yaml
current_version=$(grep 'version:' pubspec.yaml | sed 's/version: //')
echo "Current version: $current_version"

# Return to the original directory
cd "$current_dir"
```

Apple
Banana
Cherry
Date
Elderberry
Fig
Grape
Honeydew
Icaco
Jackfruit
Kiwi
Lemon
Mango
Nectarine
Orange
Papaya
Quince
Raspberry
Strawberry
Tangerine
Ugli Fruit
Vanilla Bean
Watermelon
Xigua
Yellow Watermelon
Zucchini
Apricot
Blackberry
Cantaloupe
Dragonfruit
Eggfruit
Feijoa
Guava
Huckleberry
Ilama
Jujube
Kumquat
Lime
Mulberry
Nashi Pear
Olive
Peach
Quararibea cordata
Rambutan
Starfruit
Tomato
Uva Ursi
Voavanga
White Sapote
Ximenia
Yuzu
Zapote
Acerola
Bilberry
Currant
Dewberry
Emu Apple
Finger Lime
Gac
Horned Melon
Indian Fig
Jaboticaba

Kiwano
Lychee
Mandarin
Noni
Otaheite Apple
Pear
Queen Anne Cherry
Red Banana
Salak
Tamarillo
Umbu
Velvet Apple
Winter Melon
Xoconostle
Yangmei
Zalzalak

```
<?xml version="1.0" encoding="UTF-8"?>
<testsuites>
  <testsuite name="fastlane.lanes">


      <testcase classname="fastlane.lanes" name="0: default_platform" time="0.000317">

      </testcase>


      <testcase classname="fastlane.lanes" name="1: ./fruit_version_build.sh" time="0.019917">

      </testcase>


      <testcase classname="fastlane.lanes" name="2: flutter build web" time="43.912456">

      </testcase>


      <testcase classname="fastlane.lanes" name="3: firebase deploy" time="13.974227">

      </testcase>

  </testsuite>
</testsuites>
```

```
# app_identifier("[[APP_IDENTIFIER]]") # The bundle identifier of your app
# apple_id("[[APPLE_ID]]") # Your Apple Developer Portal username


# For more information about the Appfile, see:
#     https://docs.fastlane.tools/advanced/#appfile
```

fastlane documentation

Installation

Make sure you have the latest version of the Xcode command line tools installed:

```sh
xcode-select --install
```

For _fastlane_ installation instructions, see [Installing _fastlane_](https://docs.fastlane.tools/#insta

Available Actions

web

web deploy

```sh
[bundle exec] fastlane web deploy
```

Deploy new version build

web deployCurrent

```sh
[bundle exec] fastlane web deployCurrent
```

Update current version build

web deployCurrentToDemo

```sh
[bundle exec] fastlane web deployCurrentToDemo
```

Deploy current version build to Demo with a random channel

web printCurrent

```sh
[bundle exec] fastlane web printCurrent
```

Print current build version

This README.md is auto-generated and will be re-generated every time [_fastlane_](https://fastlane.tools

More information about _fastlane_ can be found on fastlane.tools.

The documentation of _fastlane_ can be found on docs.fastlane.tools.

```
source "https://rubygems.org"

gem "fastlane"
```

```
import "dart:convert";
import "package:flutter/services.dart" show rootBundle;

import "env/environment.dart";

class BuildConfig {
  BuildConfig({
    required this.isProduction,
    this.enableLogging = true,
  });

  factory BuildConfig.fromJson(final Map<String, dynamic> json) => BuildConfig(
        isProduction: json["isProduction"],
        enableLogging: json["enableLogging"],
      );
  final bool isProduction;
  final bool enableLogging;
}

Future<BuildConfig> loadBuildConfig() async {
  final isProduction = getIsProduction();
  return BuildConfig(isProduction: isProduction);
}
```

```dart
import "package:flutter/material.dart";
import "package:flutter/widgets.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/app_style.dart";

class A4PageContainer extends StatelessWidget {
  // Background color of the container
  A4PageContainer({required this.child, this.color = AppColors.background});

  final Widget child; // Child widget to be placed inside the container
  final Color color;

  @override
  Widget build(final BuildContext context) {
    // Get the total available height of the screen
    final screenHeight = MediaQuery.of(context).size.height;

    // A4 aspect ratio: 210 mm (width) : 297 mm (height) => 1 : 1.414
    final aspectRatio = 210 / 297;

    // Calculate the width based on the height and aspect ratio
    final pageWidth = screenHeight * aspectRatio;

    return Center(
      child: Container(
          width: pageWidth,
          height: screenHeight,
          color: color,
          child: Center(child: child)),
    );
  }
}


class A4PageWidthContainer extends StatelessWidget {
  // Background color of the container
  A4PageWidthContainer({required this.child, this.color = Colors.white});

  final Widget child; // Child widget to be placed inside the container
  final Color color;

  @override
  Widget build(final BuildContext context) {
    // Get the total available height of the screen
    final screenHeight = MediaQuery.of(context).size.height;

    // A4 aspect ratio: 210 mm (width) : 297 mm (height) => 1 : 1.414
    final aspectRatio = 210 / 297;

    // Calculate the width based on the height and aspect ratio
    final pageWidth = screenHeight * aspectRatio;

    return Center(
      child: Container(
          width: pageWidth,
          color: color,
          child: Center(child: child)),
    );
  }
}
```

```dart
import "package:flutter_bloc/flutter_bloc.dart";
import "package:get_it/get_it.dart";
import 'package:menyusha/app/data/firebase/menu/menu_payload_repository.dart';
import "package:menyusha/app/features/main/screen/menyusha/admin/admin_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/view_menu/private_menu_state.dart";
import "package:menyusha/app/features/main/screen/menyusha/public/view_menu/public_menu_state.dart";

import "../../../../../../data/firebase/auth/auth_state.dart";
import "../../../../../../data/firebase/menu/menu_payload.dart";
import "../../login/auth_cubit.dart";

///cubit logic
class PrivateMenuCubit extends Cubit<PrivateMenuState> {
  PrivateMenuCubit(this.id)
      : super(
          PrivateMenuState(
            activeMenu: null,
          ),
        ) {
    init(id);
  }

  late final AuthenticationCubit authorizationCubit =
      getIt<AuthenticationCubit>();

  final String id;
  final getIt = GetIt.instance;

  final MenuPayloadRepository repository = MenuPayloadRepository();

  Future<void> initialize() async {
    // initialize();
  }

  Future<void> init(final String id) async {
    final item = await repository.getItem(id);
    emit(state.copyWith(activeMenu: item));
  }

  Future<void> signOut() async => authorizationCubit.signOut();

  Future<void> createNewMenu({required final MenuPayload payload}) async =>
      repository.createMenu(payload);

  Future<void> deleteMenu({required final MenuPayload payload}) async =>
      repository.deleteMenu(payload);

  Future<void> updateMenu({required final MenuPayload payload}) async =>
      repository.updateMenu(payload);

  Future<void> saveMenu({required final Menu menu}) async {
    final activeMenu = state.activeMenu;

    if (activeMenu == null) return;

    final menuToUpdate = activeMenu.copyWith(menu: menu);
    updateMenu(payload: menuToUpdate);
    emit(state.copyWith(activeMenu: menuToUpdate));

  }

  Future<void> switchMode() async {
```

```
    emit(state.copyWith(
        viewMode: state.viewMode == Mode.VIEW ? Mode.EDIT_JSON : Mode.VIEW));
  }
}
```

```dart
import "dart:convert";

import "../../../../../../data/firebase/menu/menu_payload.dart";


class PrivateMenuState {
  PrivateMenuState({
    required this.activeMenu,
    this.viewMode = Mode.VIEW,
  });

  final MenuPayload? activeMenu;

  final Mode viewMode;

  // The copyWith method
  PrivateMenuState copyWith({
    final MenuPayload? activeMenu,
    final Mode? viewMode,
  }) =>
      PrivateMenuState(
        activeMenu: activeMenu ?? this.activeMenu,
        viewMode: viewMode ?? this.viewMode,
      );
}

enum Mode {
  VIEW,
  EDIT_JSON,
  EDIT_UI,
  ;

  String toJson() => name;
  // Convert string back to enum
  static DesignPreset fromJson(String json) => DesignPreset.values.byName(json);
}
```

```dart
import "package:auto_route/auto_route.dart";
import "package:flutter/cupertino.dart";
import "package:flutter/material.dart";
import "package:flutter/widgets.dart";
import "package:flutter_bloc/flutter_bloc.dart";
import "package:get_it/get_it.dart";
import "package:menyusha/app/data/firebase/auth/auth_state.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/view_menu/private_menu_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/view_menu/private_menu_state.dart";
import "package:menyusha/app/features/main/screen/menyusha/login/auth_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/sample_screen.dart";
import "package:url_launcher/url_launcher.dart";

import "../../../../../../data/firebase/menu/menu_payload.dart";
import "../../../../../../root/app_router.dart";
import "../../../base/responsive_state.dart";
import "../../a4_page_container.dart";
import "../../menu/menyusha_renderer.dart";
import "../../theme/app_style.dart";

@RoutePage()
class PrivateMenuScreen extends StatefulWidget {
  PrivateMenuScreen({super.key, @PathParam('id') required this.id});

  final String id;

  final getIt = GetIt.instance;

  // Initialize the cubit with the passed id
  late final PrivateMenuCubit cubit = getIt<PrivateMenuCubit>(param1: id);

  @override
  _PrivateMenuScreenState createState() => _PrivateMenuScreenState(cubit);
}

class _PrivateMenuScreenState extends ResponsiveState<PrivateMenuScreen,
    PrivateMenuState, PrivateMenuCubit> {
  _PrivateMenuScreenState(super.cubit);

  @override
  void onStateChange(
    final BuildContext context,
    final PrivateMenuState state,
  ) {}

  @override
  Widget buildDesktopLayout(
    final BuildContext context,
    final PrivateMenuState state,
  ) =>
      A4PageContainer(
        child: buildBody(state: state),
      );

  @override
  Widget buildMobileLayout(
    final BuildContext context,
    final PrivateMenuState state,
  ) =>
      buildBody(state: state);

  Widget buildDescriptionPreview() {
```

```
  return Row(
    mainAxisSize: MainAxisSize.max,
    mainAxisAlignment: MainAxisAlignment.start,
    children: [
      Flexible(
        child: Text(
          'Превью:',
          style: AppStyles.body3Style,
        ),
      ),
      const SizedBox(width: 8.0),
    ],
  );
}

Widget buildDescription() {
  return Row(
    mainAxisSize: MainAxisSize.max,
    mainAxisAlignment: MainAxisAlignment.start,
    children: [
      Flexible(
        child: Text(
          'Меню:',
          style: AppStyles.body2Style,
        ),
      ),
      const SizedBox(width: 8.0),
    ],
  );
}

Widget buildBlueFilledButton(
        {required final Widget child,
        required final VoidCallback onPressed,
        final bool enabled = true,
        final EdgeInsetsGeometry? padding =
            const EdgeInsets.symmetric(horizontal: 4.0),
        final bool wrapContent = false}) =>
    SizedBox(
      height: wrapContent ? null : 44,
      child: ElevatedButton(
        onPressed: enabled ? onPressed : null,
        style: ElevatedButton.styleFrom(
          backgroundColor: enabled
              ? AppColors.blueAccent
              : AppColors.blueAccent
                  .withOpacity(0.5), // Dimmed color when disabled
          padding: const EdgeInsets.symmetric(vertical: 16.0),
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(8),
          ),
        ),
        child: Padding(
          padding: padding ?? EdgeInsets.zero,
          child: child,
        ),
      ),
    );

Widget buildMenuItem({required final MenuPayload item}) {
  return Padding(
    padding: const EdgeInsets.only(top: 8.0),
```

```
    child: buildBlueFilledButton(
        wrapContent: true,
        child: Padding(
          padding: const EdgeInsets.all(4),
          child: Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: [
              Column(
                mainAxisAlignment: MainAxisAlignment.start,
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  Text(
                    item.title,
                    style: AppStyles.blueFilledButtonTextStyle,
                  ),
                  Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      Text(
                        style: AppStyles.blueFilledButtonTextStyle,
                        "Design Preset: ${item.menu.designPreset.name} Public id: ${item.publicId}",
                      )
                    ],
                  ),
                ],
              ),
              IconButton(
                icon: Icon(
                  Icons.delete,
                  color: AppColors.white,
                ),
                onPressed: () {
                  cubit.deleteMenu(payload: item);

                  final router = AutoRouter.of(context);
                  router.back();
                },
              ),
            ],
          ),
        ),
        onPressed: () {
          launchInternalPage(item.publicId, isNewTab: true);
        }),
  );
}

Future<void> launchInternalPage(String id, {bool isNewTab = true}) async {
  final baseUrl = Uri.base.origin.toString(); // Gets the base URL of your app
  final url = '$baseUrl/$id'; // Appends the id to your base URL
  await launchUrl(
    Uri.parse(url),
    webOnlyWindowName: isNewTab ? '_blank' : '_self',
  );
}

Widget buildItemBody({required final PrivateMenuState state}) {
  Widget widget;

  switch (state.viewMode) {
    case Mode.VIEW:
      {
```

```
        widget = MenuRendererWidget(
          menu: state.activeMenu!.menu,
        );
      }
    case Mode.EDIT_JSON:
      {
        widget = MenuEditorWidget(
          menu: state.activeMenu!.menu,
          onSave: (menu) {
            cubit.saveMenu(menu: menu);
          },
        );
      }
    case Mode.EDIT_UI:
      {
        widget = MenuEditorWidget(
          menu: state.activeMenu!.menu,
          onSave: (menu) {
            cubit.saveMenu(menu: menu);
          },
        );
      }
  }
  return Column(
    children: [
      Row(
        mainAxisAlignment: MainAxisAlignment.end,
        children: [
          AppDesign.buildBlueFilledButtonText(
              text: "PREVIEW",
              enabled: state.viewMode != Mode.VIEW,
              onPressed: () {
                cubit.switchMode();
              }),
          const SizedBox(width: 8.0),
          AppDesign.buildBlueFilledButtonText(
              text: "EDIT JSON",
              enabled: state.viewMode != Mode.EDIT_JSON,
              onPressed: () {
                cubit.switchMode();
              }),
          const SizedBox(width: 8.0),
          AppDesign.buildBlueFilledButtonText(
              text: "EDIT UI",
              enabled: state.viewMode != Mode.EDIT_UI,
              onPressed: () {
                cubit.switchMode();
              }),
        ],
      ),
      buildDescriptionPreview(),
      SizedBox(height: 8.0),
      widget,
    ],
  );
}

Widget buildBody({required final PrivateMenuState state}) {
  return TemplateScreenAdmin(
    child: SingleChildScrollView(
      child: Column(
        children: [
```

```
            buildDescription(),
            SizedBox(height: 16.0),
            state.activeMenu != null
                ? buildMenuItem(item: state.activeMenu!)
                : Container(),
            SizedBox(height: 8.0),
            state.activeMenu != null
                ? buildItemBody(state: state)
                : Container(),
          ],
        ),
      ),
    );
  }
}
```

```dart
import "dart:convert";
import "package:menyusha/app/data/firebase/auth/auth_state.dart";

import "../../../../../data/firebase/menu/menu_payload.dart";

class AdminState {
  AdminState({
    required this.items,
    required this.authState,
  });

  final List<MenuPayload> items;

  final AuthSuccess? authState;

  // The copyWith method
  AdminState copyWith({
    final List<MenuPayload>? items,
    final AuthSuccess? authState
  }) =>
      AdminState(
        items: items ?? this.items,
        authState: authState ?? this.authState,
      );
}
```

```dart
import "package:auto_route/annotations.dart";
import "package:auto_route/auto_route.dart";
import "package:flutter/cupertino.dart";

@RoutePage()
class AdminContainerScreen extends StatefulWidget {
  const AdminContainerScreen({super.key});

  @override
  State<AdminContainerScreen> createState() => _AdminContainerScreenState();
}

class _AdminContainerScreenState extends State<AdminContainerScreen> {
  @override
  Widget build(final BuildContext context) =>
      const AutoRouter();
}
```

```dart
import "package:flutter_bloc/flutter_bloc.dart";
import "package:get_it/get_it.dart";
import "package:menyusha/app/data/firebase/user/user_payload.dart";

import "../../../../../data/firebase/auth/auth_state.dart";
import "../../../../../data/firebase/menu/menu_payload.dart";
import "../../../../../data/firebase/menu/menu_payload_repository.dart";
import "../login/auth_cubit.dart";
import "admin_state.dart";

///cubit logic
class AdminCubit extends Cubit<AdminState> {
  AdminCubit() : super(AdminState(items: [], authState: null)) {
  }

  final getIt = GetIt.instance;

  late final AuthenticationCubit authorizationCubit =
      getIt<AuthenticationCubit>();

  final MenuPayloadRepository repository = MenuPayloadRepository();

  Future<void> listenUserItems(final UserPayload payload) async {
    repository.getItemsLive().listen((final items) async {
      emit(state.copyWith(items: items));
    });
  }
  Future<void> signOut() async => authorizationCubit.signOut();

  Future<void> createNewMenu({required final MenuPayload payload}) async =>
      repository.createMenu(payload);

  Future<void> deleteMenu({required final MenuPayload payload}) async =>
      repository.deleteMenu(payload);

  Future<void> updateMenu({required final MenuPayload payload}) async =>
      repository.updateMenu(payload);
}
```

```dart
import "dart:ui";

import "package:auto_route/annotations.dart";
import "package:auto_route/auto_route.dart";
import "package:flutter/cupertino.dart";
import "package:flutter/material.dart";
import "package:flutter/widgets.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/common/link_utils.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/list_menu/list_menu_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/list_menu/list_menu_state.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/sample_screen.dart";
import "package:url_launcher/url_launcher.dart";

import "../../../../../../data/firebase/menu/menu_payload.dart";
import "../../../../../../root/app_router.dart";
import "../../../base/responsive_state.dart";
import "../../a4_page_container.dart";
import "../../menu/menyusha_renderer.dart";
import "../../theme/app_style.dart";
import "../create_menu/create_dialog.dart";
import "create_menu_cubit.dart";
import "create_menu_state.dart";

@RoutePage()
class CreateMenuScreen extends StatefulWidget {
  CreateMenuScreen({super.key});

  final getIt = GetIt.instance;
  late final CreateMenuCubit cubit = getIt.get<CreateMenuCubit>();

  @override
  _CreateMenuScreenState createState() => _CreateMenuScreenState(cubit);
}

class _CreateMenuScreenState extends ResponsiveState<CreateMenuScreen,
    CreateMenuState, CreateMenuCubit> {
  _CreateMenuScreenState(super.cubit);

  @override
  void onStateChange(
    final BuildContext context,
    final CreateMenuState state,
  ) {}

  @override
  Widget buildDesktopLayout(
    final BuildContext context,
    final CreateMenuState state,
  ) =>
      A4PageContainer(
        child: buildBody(state: state),
      );

  @override
  Widget buildMobileLayout(
    final BuildContext context,
    final CreateMenuState state,
  ) =>
      buildBody(state: state);
```

```
  Widget buildDescription() {
    return Row(
      mainAxisSize: MainAxisSize.max,
      mainAxisAlignment: MainAxisAlignment.start,
      children: [
        Flexible(
          child: Text(
            'Створення меню',
            style: AppStyles.body2Style,
          ),
        ),
        const SizedBox(width: 8.0),
      ],
    );
  }

  Widget buildBody({required final CreateMenuState state}) => TemplateScreenAdmin(
      child: SingleChildScrollView(
        child: Column(
          children: [
            buildDescription(),
            SizedBox(height: 16.0),
            CreateMenuPayloadWidget(
              onCreate: (newPayload) {
                state.activeMenu != null
                    ? cubit.updateMenu(payload: newPayload)
                    : cubit.createNewMenu(payload: newPayload);
              },
              onPayloadChange: (previewPayload) {
                cubit.onPreviewChanged(payload: previewPayload);
              },
            ),

            if (state.activeMenu != null) MenuRendererWidget(
                    menu: state.activeMenu!.menu,
                  ) else if (state.previewMenu != null) MenuRendererWidget(
              menu: state.previewMenu!.menu,
            ) else Container(),
          ],
        ),
      ),
    );
}
```

```dart
import "dart:convert";
import "package:menyusha/app/data/firebase/auth/auth_state.dart";

import "../../../../../../data/firebase/menu/menu_payload.dart";

class CreateMenuState {
  CreateMenuState({
    required this.items,
    required this.activeMenu,
    required this.previewMenu,
  });

  final List<MenuPayload> items;
  final MenuPayload? activeMenu;
  final MenuPayload? previewMenu;
  // The copyWith method
  CreateMenuState copyWith({
    final List<MenuPayload>? items,
    final MenuPayload? activeMenu,
    final MenuPayload? previewMenu,
  }) =>
      CreateMenuState(
        items: items ?? this.items,
        activeMenu: activeMenu ?? this.activeMenu,
        previewMenu: previewMenu ?? this.previewMenu,
      );
}
```

```
import "package:flutter/material.dart";

import "../../../../../../data/firebase/menu/menu_payload.dart";
import "../../theme/app_style.dart";

class CreateMenuPayloadDialog extends StatefulWidget {
  final Function(MenuPayload) onCreate;

  const CreateMenuPayloadDialog({Key? key, required this.onCreate})
      : super(key: key);

  @override
  _CreateMenuPayloadDialogState createState() =>
      _CreateMenuPayloadDialogState();
}

class _CreateMenuPayloadDialogState extends State<CreateMenuPayloadDialog> {
  final TextEditingController titleController = TextEditingController();
  final TextEditingController titleSrcController = TextEditingController();
  final TextEditingController publicIdController = TextEditingController();
  final TextEditingController userIdController = TextEditingController();

  DesignPreset selectedPreset = DesignPreset.BANDANA;
  List<Position> positions = [];

  @override
  Widget build(BuildContext context) {
    return AlertDialog(
      title: Text('Create New MenuPayload'),
      content: SingleChildScrollView(
        child: Column(
          mainAxisSize: MainAxisSize.min,
          children: [
            TextField(
              controller: titleController,
              decoration: InputDecoration(labelText: 'Title Source'),
            ),
            // Public ID
            TextField(
              controller: publicIdController,
              decoration: InputDecoration(labelText: 'Public ID'),
            ),
            // User ID
            TextField(
              controller: userIdController,
              decoration: InputDecoration(labelText: 'User ID'),
            ),
            // Design Preset Dropdown
            DropdownButton<DesignPreset>(
              value: selectedPreset,
              onChanged: (preset) {
                setState(() {
                  selectedPreset = preset!;
                });
              },
              items: DesignPreset.values.map((preset) {
                return DropdownMenuItem(
                  value: preset,
                  child: Text(preset.name),
                );
              }).toList(),
            ),
```

```
          // Positions
          // Title Source
          TextField(
            controller: titleSrcController,
            decoration: InputDecoration(labelText: 'Image title link'),
          ),

          ElevatedButton(
            onPressed: () => _addPositionDialog(),
            child: Text('Add Position'),
          ),
          SingleChildScrollView(
            child: Column(
              mainAxisSize: MainAxisSize.min,
              children: positions.map((pos) {
                return ListTile(
                  title: Text(pos.title),
                  subtitle:
                      Text("Group: ${pos.group}, Price: \$${pos.price}"),
                  trailing: IconButton(
                    icon: Icon(Icons.delete),
                    onPressed: () => setState(() => positions.remove(pos)),
                  ),
                );
              }).toList(),
            ),
          ),
        ],
      ),
    ),
    actions: [
      TextButton(
        onPressed: () => Navigator.pop(context),
        child: Text('Cancel'),
      ),
      AppDesign.buildBlueOutlinedButtonText(
          text: "Create",
          onPressed: () {
            final newPayload = MenuPayload(
              id: "",
              publicId: publicIdController.text,
              userId: userIdController.text,
              title: titleController.text,
              menu: Menu(
                designPreset: selectedPreset,
                titleSrc: titleSrcController.text,
                positions: positions,
              ),
            );
            widget.onCreate(newPayload);
            Navigator.pop(context);
          })
    ],
  );
}

Future<void> _addPositionDialog() async {
  final TextEditingController groupController = TextEditingController();
  final TextEditingController titleController = TextEditingController();
  final TextEditingController descriptionController = TextEditingController();
  final TextEditingController priceController = TextEditingController();
  final TextEditingController outputController = TextEditingController();
```

```
    await showDialog(
      context: context,
      builder: (context) => AlertDialog(
        title: Text('Add Position'),
        content: SingleChildScrollView(
          child: Column(
            mainAxisSize: MainAxisSize.min,
            children: [
              TextField(
                controller: groupController,
                decoration: InputDecoration(labelText: 'Group'),
              ),
              TextField(
                controller: titleController,
                decoration: InputDecoration(labelText: 'Title'),
              ),
              TextField(
                controller: descriptionController,
                decoration: InputDecoration(labelText: 'Description'),
              ),
              TextField(
                controller: priceController,
                decoration: InputDecoration(labelText: 'Price'),
                keyboardType: TextInputType.numberWithOptions(decimal: true),
              ),
              TextField(
                controller: outputController,
                decoration: InputDecoration(labelText: 'Output'),
              ),
            ],
          ),
        ),
        actions: [
          TextButton(
            onPressed: () => Navigator.pop(context),
            child: Text('Cancel'),
          ),
          ElevatedButton(
            onPressed: () {
              final newPosition = Position(
                id: UniqueKey().toString(),
                group: groupController.text,
                title: titleController.text,
                description: descriptionController.text,
                price: double.parse(priceController.text),
                output: outputController.text,
              );
              setState(() {
                positions.add(newPosition);
              });
              Navigator.pop(context);
            },
            child: Text('Add Position'),
          ),
        ],
      ),
    );
  }
}

/*
```

```
class CreateMenuPayloadWidget extends StatefulWidget {
  final Function(MenuPayload) onCreate;
  final Function(MenuPayload) onPayloadChange;

  const CreateMenuPayloadWidget({
    Key? key,
    required this.onCreate,
    required this.onPayloadChange,
  }) : super(key: key);

  @override
  _CreateMenuPayloadWidgetState createState() =>
      _CreateMenuPayloadWidgetState();
}

class _CreateMenuPayloadWidgetState extends State<CreateMenuPayloadWidget> {
  final TextEditingController titleController = TextEditingController();
  final TextEditingController titleSrcController = TextEditingController();
  final TextEditingController publicIdController = TextEditingController();
  final TextEditingController userIdController = TextEditingController();

  DesignPreset selectedPreset = DesignPreset.BANDANA;
  List<Position> positions = [];

  bool showPositionForm = false;

  @override
  void initState() {
    super.initState();
    _addListeners();
  }

  void _addListeners() {
    titleController.addListener(_notifyChange);
    titleSrcController.addListener(_notifyChange);
    publicIdController.addListener(_notifyChange);
    userIdController.addListener(_notifyChange);
  }

  void _notifyChange() {
    final updatedPayload = _buildMenuPayload();
    widget.onPayloadChange(updatedPayload);
  }

  MenuPayload _buildMenuPayload() {
    return MenuPayload(
      id: "",
      publicId: publicIdController.text,
      userId: userIdController.text,
      title: titleController.text,
      menu: Menu(
        designPreset: selectedPreset,
        titleSrc: titleSrcController.text,
        positions: positions,
      ),
    );
  }

  @override
  Widget build(BuildContext context) {
    return SingleChildScrollView(
      child: Column(
```

```
children: [
  Text(
    'Create New MenuPayload',
    style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
  ),
  TextField(
    controller: titleController,
    decoration: InputDecoration(labelText: 'Title'),
  ),
  TextField(
    controller: publicIdController,
    decoration: InputDecoration(labelText: 'Public ID'),
  ),
  TextField(
    controller: userIdController,
    decoration: InputDecoration(labelText: 'User ID'),
  ),
  TextField(
    controller: titleSrcController,
    decoration: InputDecoration(labelText: 'Image Title Link'),
  ),
  DropdownButton<DesignPreset>(
    value: selectedPreset,
    onChanged: (preset) {
      setState(() {
        selectedPreset = preset!;
        _notifyChange();
      });
    },
    items: DesignPreset.values.map((preset) {
      return DropdownMenuItem(
        value: preset,
        child: Text(preset.name),
      );
    }).toList(),
  ),
  ElevatedButton(
    onPressed: () {
      setState(() {
        showPositionForm = true;
      });
    },
    child: Text('Add Position'),
  ),
  if (showPositionForm) _positionForm(),
  Column(
    children: positions.map((pos) {
      return ListTile(
        title: Text(pos.title),
        subtitle: Text("Group: ${pos.group}, Price: \$${pos.price}"),
        trailing: IconButton(
          icon: Icon(Icons.delete),
          onPressed: () => setState(() {
            positions.remove(pos);
            _notifyChange();
          }),
        ),
      );
    }).toList(),
  ),
  Row(
    mainAxisAlignment: MainAxisAlignment.end,
```

```
          children: [
            TextButton(
              onPressed: () {
                // Handle cancel action, e.g., clear fields or navigate back
              },
              child: Text('Cancel'),
            ),
            ElevatedButton(
              onPressed: () {
                final newPayload = _buildMenuPayload();
                widget.onCreate(newPayload);
                // Optionally clear fields after creation
              },
              child: Text('Create'),
            ),
          ],
        ),
      ],
    ),
  );
}

Widget _positionForm() {
  final TextEditingController groupController = TextEditingController();
  final TextEditingController titleController = TextEditingController();
  final TextEditingController descriptionController = TextEditingController();
  final TextEditingController priceController = TextEditingController();
  final TextEditingController outputController = TextEditingController();

  return Card(
    margin: EdgeInsets.symmetric(vertical: 10),
    child: Padding(
      padding: EdgeInsets.all(10),
      child: Column(
        children: [
          Text(
            'Add Position',
            style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
          ),
          TextField(
            controller: groupController,
            decoration: InputDecoration(labelText: 'Group'),
          ),
          TextField(
            controller: titleController,
            decoration: InputDecoration(labelText: 'Title'),
          ),
          TextField(
            controller: descriptionController,
            decoration: InputDecoration(labelText: 'Description'),
          ),
          TextField(
            controller: priceController,
            decoration: InputDecoration(labelText: 'Price'),
            keyboardType: TextInputType.numberWithOptions(decimal: true),
          ),
          TextField(
            controller: outputController,
            decoration: InputDecoration(labelText: 'Output'),
          ),
          Row(
            mainAxisAlignment: MainAxisAlignment.end,
```

```
              children: [
                TextButton(
                  onPressed: () {
                    setState(() {
                      showPositionForm = false;
                    });
                  },
                  child: Text('Cancel'),
                ),
                ElevatedButton(
                  onPressed: () {
                    final newPosition = Position(
                      id: UniqueKey().toString(),
                      group: groupController.text,
                      title: titleController.text,
                      description: descriptionController.text,
                      price: double.parse(priceController.text),
                      output: outputController.text,
                    );
                    setState(() {
                      positions.add(newPosition);
                      showPositionForm = false;
                      _notifyChange();
                    });
                  },
                  child: Text('Add'),
                ),
              ],
            ),
          ],
        ),
      ),
    );
  }

  @override
  void dispose() {
    titleController.dispose();
    titleSrcController.dispose();
    publicIdController.dispose();
    userIdController.dispose();
    super.dispose();
  }
}

*/

class CreateMenuPayloadWidget extends StatefulWidget {
  final Function(MenuPayload) onCreate;
  final Function(MenuPayload) onPayloadChange;

  const CreateMenuPayloadWidget({
    Key? key,
    required this.onCreate,
    required this.onPayloadChange,
  }) : super(key: key);

  @override
  _CreateMenuPayloadWidgetState createState() => _CreateMenuPayloadWidgetState();
}

class _CreateMenuPayloadWidgetState extends State<CreateMenuPayloadWidget> {
```

```
final TextEditingController titleController = TextEditingController();
final TextEditingController titleSrcController = TextEditingController();
final TextEditingController publicIdController = TextEditingController();
final TextEditingController userIdController = TextEditingController();

DesignPreset selectedPreset = DesignPreset.BANDANA;
List<Position> positions = [];
bool showPositionForm = false;

@override
void initState() {
  super.initState();
  _addListeners();
}

void _addListeners() {
  titleController.addListener(_notifyChange);
  titleSrcController.addListener(_notifyChange);
  publicIdController.addListener(_notifyChange);
  userIdController.addListener(_notifyChange);
}

void _notifyChange() {
  widget.onPayloadChange(_buildMenuPayload());
}

MenuPayload _buildMenuPayload() {
  return MenuPayload(
    id: "",
    publicId: publicIdController.text,
    userId: userIdController.text,
    title: titleController.text,
    menu: Menu(
      designPreset: selectedPreset,
      titleSrc: titleSrcController.text,
      positions: positions,
    ),
  );
}

@override
Widget build(BuildContext context) {
  return SingleChildScrollView(
    padding: const EdgeInsets.all(16.0),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        Text(
          'Create New MenuPayload',
          style: AppStyles.body3Style,
        ),
        const SizedBox(height: 12.0),
        _buildInputField(controller: titleController, label: 'Title'),
        _buildInputField(controller: publicIdController, label: 'Public ID'),
        _buildInputField(controller: userIdController, label: 'User ID'),
        _buildInputField(controller: titleSrcController, label: 'Image Title Link'),
        _buildDesignPresetDropdown(),
        const SizedBox(height: 12.0),
        _buildAddPositionButton(),
        if (showPositionForm) _positionForm(),
        const SizedBox(height: 12.0),
        _buildPositionList(),
```

```
        _buildActionButtons(),
      ],
    ),
  );
}

Widget _buildInputField({required TextEditingController controller, required String label}) {
  return Padding(
    padding: const EdgeInsets.symmetric(vertical: 8.0),
    child: TextField(
      controller: controller,
      decoration: AppStyles.inputDecoration.copyWith(labelText: label),
      style: AppStyles.labelTextStyle,
    ),
  );
}

Widget _buildDesignPresetDropdown() {
  return DropdownButton<DesignPreset>(
    value: selectedPreset,
    onChanged: (preset) {
      setState(() {
        selectedPreset = preset!;
        _notifyChange();
      });
    },
    items: DesignPreset.values.map((preset) {
      return DropdownMenuItem(
        value: preset,
        child: Text(preset.name, style: AppStyles.bodyStyle),
      );
    }).toList(),
  );
}

Widget _buildAddPositionButton() {
  return AppDesign.buildBlueOutlinedButtonText(
    onPressed: () => setState(() => showPositionForm = true),
    text: 'Add Position',
  );
}

Widget _positionForm() {
  final TextEditingController groupController = TextEditingController();
  final TextEditingController titleController = TextEditingController();
  final TextEditingController descriptionController = TextEditingController();
  final TextEditingController priceController = TextEditingController();
  final TextEditingController outputController = TextEditingController();

  return Card(
    margin: const EdgeInsets.symmetric(vertical: 10.0),
    child: Padding(
      padding: const EdgeInsets.all(10.0),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text('Add Position', style: AppStyles.body2Style),
          _buildInputField(controller: groupController, label: 'Group'),
          _buildInputField(controller: titleController, label: 'Title'),
          _buildInputField(controller: descriptionController, label: 'Description'),
          _buildInputField(
            controller: priceController,
```

```
            label: 'Price',
          ),
          _buildInputField(controller: outputController, label: 'Output'),
          Row(
            mainAxisAlignment: MainAxisAlignment.end,
            children: [
              AppDesign.buildBlueOutlinedButtonText(
                onPressed: () => setState(() => showPositionForm = false),
                text: 'Cancel',
              ),
              const SizedBox(width: 8.0),
              AppDesign.buildBlueOutlinedButtonText(
                onPressed: () {
                  final newPosition = Position(
                    id: UniqueKey().toString(),
                    group: groupController.text,
                    title: titleController.text,
                    description: descriptionController.text,
                    price: double.tryParse(priceController.text) ?? 0.0,
                    output: outputController.text,
                  );
                  setState(() {
                    positions.add(newPosition);
                    showPositionForm = false;
                    _notifyChange();
                  });
                },
                text: 'Add',
              ),
            ],
          ),
        ],
      ),
    ),
  );
}

Widget _buildPositionList() {
  return Column(
    children: positions.map((pos) {
      return ListTile(
        title: Text(pos.title, style: AppStyles.bodyStyle),
        subtitle: Text("Group: ${pos.group}, Price: \$${pos.price}", style: AppStyles.footerStyle),
        trailing: IconButton(
          icon: const Icon(Icons.delete),
          onPressed: () {
            setState(() {
              positions.remove(pos);
              _notifyChange();
            });
          },
        ),
      );
    }).toList(),
  );
}

Widget _buildActionButtons() {
  return Row(
    mainAxisAlignment: MainAxisAlignment.end,
    children: [
      AppDesign.buildBlueOutlinedButtonText(
```

```
          onPressed: () {
            // Handle cancel action, e.g., clear fields or navigate back
          },
          text: 'Cancel',
        ),
        const SizedBox(width: 8.0),
        AppDesign.buildBlueOutlinedButtonText(
          onPressed: () {
            widget.onCreate(_buildMenuPayload());
            // Optionally clear fields after creation
          },
          text: 'Create',
        ),
      ],
    );
  }

  @override
  void dispose() {
    titleController.dispose();
    titleSrcController.dispose();
    publicIdController.dispose();
    userIdController.dispose();
    super.dispose();
  }
}

// Assuming the MenuPayload, Menu, DesignPreset, and Position classes are defined elsewhere.
```

```dart
import "package:flutter_bloc/flutter_bloc.dart";
import "package:get_it/get_it.dart";
import "package:menyusha/app/data/firebase/user/user_payload.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/list_menu/list_menu_state.dart";

import "../../../../../../data/firebase/auth/auth_state.dart";
import "../../../../../../data/firebase/menu/menu_payload.dart";
import "../../../../../../data/firebase/menu/menu_payload_repository.dart";
import "../../login/auth_cubit.dart";
import "../admin_state.dart";
import "create_menu_state.dart";

///cubit logic
class CreateMenuCubit extends Cubit<CreateMenuState> {
  CreateMenuCubit() : super(CreateMenuState(items: [], activeMenu: null, previewMenu: null)) {
    initialize();
  }

  final getIt = GetIt.instance;

  late final AuthenticationCubit authorizationCubit =
      getIt<AuthenticationCubit>();

  final MenuPayloadRepository repository = MenuPayloadRepository();

  Future<void> initialize() async {
    // initialize();
  }

  Future<void> signOut() async => authorizationCubit.signOut();

  Future<void> createNewMenu({required final MenuPayload payload}) async {
    final created = await repository.createMenu(payload);
    emit(state.copyWith(activeMenu: created));
  }

  Future<void> deleteMenu({required final MenuPayload payload}) async =>
      repository.deleteMenu(payload);

  Future<void> updateMenu({required final MenuPayload payload}) async {
    final created =
        await repository.updateMenu(payload.copyWith(id: state.activeMenu?.id));
    emit(state.copyWith(activeMenu: created));
  }

  Future<void> onPreviewChanged({required final MenuPayload payload}) async {
    emit(state.copyWith(previewMenu: payload));
  }
}
```

```dart
import "dart:ui";

import "package:auto_route/annotations.dart";
import "package:auto_route/auto_route.dart";
import "package:flutter/cupertino.dart";
import "package:flutter/material.dart";
import "package:flutter/widgets.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/common/link_utils.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/list_menu/list_menu_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/list_menu/list_menu_state.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/sample_screen.dart";
import "package:url_launcher/url_launcher.dart";

import "../../../../../../data/firebase/menu/menu_payload.dart";
import "../../../../../../root/app_router.dart";
import "../../../base/responsive_state.dart";
import "../../a4_page_container.dart";
import "../../theme/app_style.dart";
import "../admin_cubit.dart";
import "../create_menu/create_dialog.dart";

@RoutePage()
class ListMenuScreen extends StatefulWidget {
  ListMenuScreen({super.key});

  final getIt = GetIt.instance;
  late final ListMenuCubit cubit = getIt.get<ListMenuCubit>();

  @override
  _ListMenuScreenState createState() => _ListMenuScreenState(cubit);
}

class _ListMenuScreenState
    extends ResponsiveState<ListMenuScreen, ListMenuState, ListMenuCubit> {
  _ListMenuScreenState(super.cubit);

  @override
  void onStateChange(
    final BuildContext context,
    final ListMenuState state,
  ) {}

  @override
  Widget buildDesktopLayout(
    final BuildContext context,
    final ListMenuState state,
  ) =>
      A4PageContainer(
        child: buildBody(state: state),
      );

  @override
  Widget buildMobileLayout(
    final BuildContext context,
    final ListMenuState state,
  ) =>
      buildBody(state: state);

  Widget buildDescription() {
    return Row(
```

```
    mainAxisSize: MainAxisSize.max,
    mainAxisAlignment: MainAxisAlignment.start,
    children: [
      Flexible(
        child: Text(
          'Список меню:',
          style: AppStyles.body2Style,
        ),
      ),
      const SizedBox(width: 8.0),
    ],
  );
}

Widget buildBlueFilledButton(
        {required final Widget child,
        required final VoidCallback onPressed,
        final bool enabled = true,
        final EdgeInsetsGeometry? padding =
            const EdgeInsets.symmetric(horizontal: 4.0),
        final bool wrapContent = false}) =>
    SizedBox(
      height: wrapContent ? null : 44,
      child: ElevatedButton(
        onPressed: enabled ? onPressed : null,
        style: ElevatedButton.styleFrom(
          backgroundColor: enabled
              ? AppColors.blueAccent
              : AppColors.blueAccent
                  .withOpacity(0.5), // Dimmed color when disabled
          padding: const EdgeInsets.symmetric(vertical: 16.0),
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(8),
          ),
        ),
        child: Padding(
          padding: padding ?? EdgeInsets.zero,
          child: child,
        ),
      ),
    );

Widget buildMenuItem({required final MenuPayload item}) {
  return Padding(
    padding: const EdgeInsets.only(top: 8.0),
    child: buildBlueFilledButton(
        wrapContent: true,
        child: Padding(
          padding: const EdgeInsets.all(4),
          child: Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: [
              Expanded(
                child: Column(
                  mainAxisAlignment: MainAxisAlignment.start,
                  crossAxisAlignment: CrossAxisAlignment.start,
                  children: [
                    Text(
                      item.title,
                      style: AppStyles.blueFilledButtonTextStyle,
                    ),
                    Column(
```

```
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        Text(
                          style: AppStyles.blueFilledButtonTextStyle,
                          "Design Preset: ${item.menu.designPreset.name} Public id: ${item.publicId}",
                        )
                      ],
                    ),
                  ],
                ),
              ),
              IconButton(
                icon: Icon(
                  Icons.delete,
                  color: AppColors.white,
                ),
                onPressed: () => cubit.deleteMenu(payload: item),
              ),
            ],
          ),
        ),
        onPressed: () {
          final router = AutoRouter.of(context);
          // Use pushAndRemoveUntil with the root route
          router.navigate(
            PrivateMenuRoute(id: item.id),
          );
        }),
  );
}

Widget buildBody({required final ListMenuState state}) {
  final reachLimit = (state.items.length >= 3);

  return TemplateScreenAdmin(
      child: Column(
    children: [
      buildDescription(),
      Column(
        children: state.items.map((item) {
          return buildMenuItem(item: item);
        }).map((item) {
          return item;
        }).toList(),
      ),
      SizedBox(height: 64.0),
      AppDesign.buildBlueOutlinedButtonText(
          text: reachLimit ? "Ліміт досягнуто" : "Створити нове меню",
          enabled: !reachLimit,
          onPressed: () {
            final router = AutoRouter.of(context);
            // Use pushAndRemoveUntil with the root route
            router.navigate(
              CreateMenuRoute(),
            );
          }),
      AppDesign.buildBlueOutlinedButtonText(
          text: "DESIGN PREVIEW",
          enabled: !reachLimit,
          onPressed: () {
            final router = AutoRouter.of(context);
            // Use pushAndRemoveUntil with the root route
```

```
              router.navigate(
                PreviewRoute(),
              );
            }),
        AppDesign.buildBlueOutlinedButtonText(
            text: "DESIGN SAMPLE",
            enabled: !reachLimit,
            onPressed: () {
              final router = AutoRouter.of(context);
              // Use pushAndRemoveUntil with the root route
              router.navigate(
                SampleRoute(),
              );
            })
      ],
    ));
  }
}
```

```
import "dart:convert";
import "package:menyusha/app/data/firebase/auth/auth_state.dart";

import "../../../../../../data/firebase/menu/menu_payload.dart";

class ListMenuState {
  ListMenuState({
    required this.items,
  });

  final List<MenuPayload> items;
  // The copyWith method
  ListMenuState copyWith({
    final List<MenuPayload>? items,
  }) =>
      ListMenuState(
        items: items ?? this.items,
      );
}
```

```dart
import "package:flutter_bloc/flutter_bloc.dart";
import "package:get_it/get_it.dart";
import "package:menyusha/app/data/firebase/user/user_payload.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/list_menu/list_menu_state.dart";

import "../../../../../../data/firebase/auth/auth_state.dart";
import "../../../../../../data/firebase/menu/menu_payload.dart";
import "../../../../../../data/firebase/menu/menu_payload_repository.dart";
import "../../login/auth_cubit.dart";
import "../admin_state.dart";

///cubit logic
class ListMenuCubit extends Cubit<ListMenuState> {
  ListMenuCubit() : super(ListMenuState(items: [])) {
    initialize();
  }

  final getIt = GetIt.instance;

  late final AuthenticationCubit authorizationCubit =
      getIt<AuthenticationCubit>();

  final MenuPayloadRepository repository = MenuPayloadRepository();

  Future<void> listenUserItems(final UserPayload payload) async {
    repository.getItemsLive().listen((final items) async {
      emit(state.copyWith(items: items));
    });
  }

  Future<void> initialize() async {
    // initialize();
    authorizationCubit.stream.listen((final authorizationState) async {
      if (authorizationState is AuthLoading) {
      } else if (authorizationState is AuthInitial) {
      } else if (authorizationState is AuthFailed) {
      } else if (authorizationState is AuthSuccess) {
        await listenUserItems(authorizationState.userPayload!);
      } else {}
    });
  }

  Future<void> signOut() async => authorizationCubit.signOut();

  Future<void> createNewMenu({required final MenuPayload payload}) async =>
      repository.createMenu(payload);

  Future<void> deleteMenu({required final MenuPayload payload}) async =>
      repository.deleteMenu(payload);

  Future<void> updateMenu({required final MenuPayload payload}) async =>
      repository.updateMenu(payload);
}
```

```dart
import "dart:convert";

import "../../../../../../data/firebase/menu/menu_payload.dart";

class PublicMenuState {
  PublicMenuState({
    required this.isInitialization,
    required this.menuLoadingFailed,
    required this.loadedMenu,
  });

  final MenuPayload? loadedMenu;
  final bool isInitialization;
  final bool menuLoadingFailed;

  // The copyWith method
  PublicMenuState copyWith({
    final MenuPayload? activeMenu,
    final bool? isInitialization,
    final bool? menuLoadingFailed,
  }) =>
      PublicMenuState(
        loadedMenu: activeMenu ?? this.loadedMenu,
        isInitialization: isInitialization ?? this.isInitialization,
        menuLoadingFailed: menuLoadingFailed ?? this.menuLoadingFailed,
      );
}
```

```dart
import "dart:convert";

import "package:auto_route/auto_route.dart";
import "package:flutter/material.dart";
import "package:get_it/get_it.dart";
import "package:menyusha/app/features/main/screen/menyusha/menu/menyusha_renderer.dart";
import "package:menyusha/app/features/main/screen/menyusha/public/view_menu/public_menu_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/public/view_menu/public_menu_state.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/app_style.dart";
import "package:menyusha/app/root/app_router.dart";
import "package:uuid/uuid.dart";

import "../../../base/base_screen.dart";
import "../../../base/responsive_state.dart";
import "../../a4_page_container.dart";
import "../../theme/theme_manager.dart";

import 'dart:html' as html;

@RoutePage()
class PublicMenuScreen extends StatefulWidget {
  PublicMenuScreen({super.key, @PathParam('id') required this.id});

  final String id;

  final getIt = GetIt.instance;

  // Initialize the cubit with the passed id
  late final PublicMenuCubit cubit = getIt<PublicMenuCubit>(param1: id);

  @override
  _PublicMenuScreenState createState() => _PublicMenuScreenState(cubit);
}

///ui
class _PublicMenuScreenState extends ResponsiveState<PublicMenuScreen,
    PublicMenuState, PublicMenuCubit> {
  _PublicMenuScreenState(super.cubit);

  @override
  void onStateChange(
    final BuildContext context,
    final PublicMenuState state,
  ) {
    if (state.menuLoadingFailed) {
      final router = AutoRouter.of(context);
      router.navigate(
        NotFoundRoute(),
      );
    }
  }

  @override
  Widget buildDesktopLayout(
    final BuildContext context,
    final PublicMenuState state,
  ) =>
      buildBody(state: state);

  @override
  Widget buildMobileLayout(
    final BuildContext context,
```

```
    final PublicMenuState state,
  ) =>
      buildBody(state: state);

  Widget buildBody({required final PublicMenuState state}) {
    if (state.loadedMenu == null) {
      return A4PageContainer(child: AppDesign.buildLogoLoader());
    } else {
      final theme =
          MenuThemeManager.getTheme(state.loadedMenu!.menu.designPreset);

      return SingleChildScrollView(
        child: Column(
          children: [
            Container(
                child: Title(
              title: state.loadedMenu!.title,
              color: Colors.black,
              child: MenuRendererWidget(
                menu: state.loadedMenu!.menu,
              ),
            )),

            A4PageWidthContainer(
              child: Padding(
                padding: const EdgeInsets.symmetric(horizontal: 32.0),
                child: Column(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: [
                    AppDesign.buildLogoSmall(context),
                    AppDesign.buildCopyrightTextYear(theme.backgroundColor)
                  ],
                ),
              ),

              color: theme.backgroundColor,
            ),
          ],
        ),
      );
    }
  }
}

Color getContrastingColor(Color color) {
  return color.computeLuminance() > 0.5 ? Colors.black : Colors.white;
}

Color invertColor(Color color) {
  return Color.fromARGB(
    color.alpha,
    255 - color.red,
    255 - color.green,
    255 - color.blue,
  );
}



class SkeletonLoader extends StatelessWidget {
  @override
  Widget build(BuildContext context) => Center(
```

```
        child: SingleChildScrollView(
          child: Padding(
            padding: const EdgeInsets.all(10.0),
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.center,
              children: [
                // Skeleton Logo
                SkeletonWidget(
                  width: 350,
                  height: 100,
                  borderRadius: BorderRadius.circular(8.0),
                  animationType: AnimationType.blink,
                ),
                SizedBox(height: 16),

                // Group 1 Header
                SkeletonWidget(
                  width: MediaQuery.of(context).size.width * 0.9,
                  height: 40,
                  borderRadius: BorderRadius.circular(4.0),
                ),
                SizedBox(height: 15),

                // Skeleton Items
                for (int i = 0; i < 2; i++) ...[
                  SkeletonItem(),
                ],

                // Group 2 Header
                SkeletonWidget(
                  width: MediaQuery.of(context).size.width * 0.9,
                  height: 40,
                  borderRadius: BorderRadius.circular(4.0),
                ),
                SizedBox(height: 15),

                // Skeleton Items
                for (int i = 0; i < 3; i++) ...[
                  SkeletonItem(),
                ],

                SizedBox(height: 20),
              ],
            ),
          ),
        ),
      );
}

class SkeletonWidget extends StatelessWidget {
  final double width;
  final double height;
  final BorderRadius borderRadius;
  final AnimationType animationType;

  const SkeletonWidget({
    required this.width,
    required this.height,
    required this.borderRadius,
    this.animationType = AnimationType.loading,
  });
```

```
  @override
  Widget build(BuildContext context) {
    return AnimatedContainer(
      duration: Duration(milliseconds: 1300),
      width: width,
      height: height,
      decoration: BoxDecoration(
        color: Colors.grey[300],
        borderRadius: borderRadius,
      ),
      child: animationType == AnimationType.blink
          ? Opacity(
              opacity: 0.8,
              child: Container(color: Colors.grey[200]),
            )
          : null,
    );
  }
}

class SkeletonItem extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.symmetric(vertical: 8.0),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          SkeletonWidget(
            width: MediaQuery.of(context).size.width * 0.7,
            height: 20,
            borderRadius: BorderRadius.circular(4.0),
          ),
          SizedBox(height: 10),
          SkeletonWidget(
            width: MediaQuery.of(context).size.width,
            height: 15,
            borderRadius: BorderRadius.circular(4.0),
          ),
          SizedBox(height: 8),
          SkeletonWidget(
            width: MediaQuery.of(context).size.width * 0.3,
            height: 20,
            borderRadius: BorderRadius.circular(4.0),
          ),
        ],
      ),
    );
  }
}

enum AnimationType {
  loading,
  blink,
}
```

```dart
import "package:flutter_bloc/flutter_bloc.dart";
import "package:get_it/get_it.dart";
import 'package:menyusha/app/data/firebase/menu/menu_payload_repository.dart';
import "package:menyusha/app/features/main/screen/menyusha/public/view_menu/public_menu_state.dart";

///cubit logic
class PublicMenuCubit extends Cubit<PublicMenuState> {
  PublicMenuCubit(this.id)
      : super(
          PublicMenuState(
              loadedMenu: null,
              isInitialization: true,
              menuLoadingFailed: false),
        ) {
    init(id);
  }

  final String id;
  final getIt = GetIt.instance;

  final MenuPayloadRepository repository = MenuPayloadRepository();

  Future<void> init(final String id) async {
    final item = await repository.getItemByPublicId(id);

    if (item == null) {
      emit(state.copyWith(
          activeMenu: item, isInitialization: false, menuLoadingFailed: true));
    } else {
      emit(state.copyWith(
          activeMenu: item, isInitialization: false, menuLoadingFailed: false));
    }
  }
}
```

```dart
import "package:flutter/material.dart";
import "package:google_fonts/google_fonts.dart";

import "../../../../../data/firebase/menu/menu_payload.dart";

class MenuThemeManager {
  static MenuTheme commonLightTheme() {
    final TextStyle textStyle =
    GoogleFonts.openSans(fontSize: 14, color: Colors.white);

    return MenuTheme(
      titleStyle: textStyle.copyWith(
        fontSize: 24,
        fontWeight: FontWeight.bold,
      ),
      groupTitleStyle: textStyle.copyWith(
        fontSize: 22,
        fontWeight: FontWeight.bold,
      ),
      itemTitleStyle: textStyle.copyWith(
        fontSize: 18,
        fontWeight: FontWeight.w600,
      ),
      itemDescriptionStyle: textStyle.copyWith(
        fontSize: 14,
        fontWeight: FontWeight.w100,
      ),
      itemPriceStyle: textStyle.copyWith(
        fontSize: 18,
        fontWeight: FontWeight.bold,
      ),
      backgroundColor: Colors.black, // Example background color
    );
  }

  static MenuTheme commonDarkTheme() {
    final TextStyle textStyle = const TextStyle(
      fontSize: 14,
      color: Colors.black,
    );

    return MenuTheme(
      titleStyle: textStyle.copyWith(
        fontSize: 24,
        fontWeight: FontWeight.bold,
      ),
      groupTitleStyle: textStyle.copyWith(
        fontSize: 22,
        fontWeight: FontWeight.bold,
      ),
      itemTitleStyle: textStyle.copyWith(
        fontSize: 18,
        fontWeight: FontWeight.w500,
      ),
      itemDescriptionStyle: textStyle.copyWith(
        fontSize: 16,
        fontWeight: FontWeight.normal,
      ),
      itemPriceStyle: textStyle.copyWith(
        fontSize: 18,
```

```
        fontWeight: FontWeight.bold,
      ),
      backgroundColor: Colors.white, // Example background color
    );
  }

  static MenuTheme getTheme(DesignPreset designPreset) {
    switch (designPreset) {
      case DesignPreset.SANTOKU:
        break;
      case DesignPreset.BBQ:
        break;
      case DesignPreset.BOSTON:
        break;
      case DesignPreset.VOLGA:
        break;
      case DesignPreset.CUSTOM:
        break;
      case DesignPreset.BANDANA:
        break;
    }
    return commonLightTheme();
  }
}

class MenuTheme {
  final TextStyle titleStyle;
  final TextStyle groupTitleStyle;
  final TextStyle itemTitleStyle;
  final TextStyle itemDescriptionStyle;
  final TextStyle itemPriceStyle;

  final Color backgroundColor; // Background color (optional)

  MenuTheme({
    required this.titleStyle,
    required this.groupTitleStyle,
    required this.itemTitleStyle,
    required this.itemDescriptionStyle,
    required this.itemPriceStyle,
    required this.backgroundColor,
  });
}
```

```
import "package:auto_route/auto_route.dart";
import "package:flutter/material.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";

import "../../../../../common/util/svg_manager.dart";
import "../../../../../root/app_router.dart";
import "../login/auth_cubit.dart";

class ImageLoader extends StatefulWidget {
  @override
  _ImageLoaderState createState() => _ImageLoaderState();
}

class _ImageLoaderState extends State<ImageLoader>
    with SingleTickerProviderStateMixin {
  late AnimationController _controller;

  @override
  void initState() {
    super.initState();
    _controller = AnimationController(
      vsync: this,
      duration: Duration(milliseconds: 750),
    )..repeat(reverse: true);
  }

  @override
  void dispose() {
    _controller.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) => Center(
        child: FadeTransition(
          opacity: _controller,
          child: AppDesign.buildLogo(context),
        ),
      );
}

class AppDesign {
  static Color getContrastingColor(Color color) =>
      color.computeLuminance() > 0.5 ? Colors.black : Colors.white;

  static Widget buildCopyrightTextYear(Color? bgColor) => Text(
        "□ 2024",
        style: AppStyles.copyrightStyle.copyWith(
            color: bgColor == null ? null : getContrastingColor(bgColor)),
      );

  static Widget buildCopyrightSection() => Column(
        children: [
          buildCopyrightTextYear(null),
          SvgManager.bandana(width: 8, height: 10, color: AppColors.fontOnBackground),
        ],
      );

  static Widget buildLogoLoader() =>
      Container(width: 270, height: 130, child: ImageLoader());
```

```
static Widget buildLogoSmall(BuildContext context) => GestureDetector(
      onTap: () {
        final router = AutoRouter.of(context);
        router.navigate(HelloRoute());
      },
      child: Padding(
        padding: const EdgeInsets.symmetric(vertical: 16.0),
        child: SizedBox(
          height: 20,
          child: SvgManager.logo(width: 8, height: 10),
        ),
      ),
    );

static Widget buildLogo(BuildContext context) => GestureDetector(
      onTap: () {
        final router = AutoRouter.of(context);
        router.navigate(HelloRoute());
      },
      child: Padding(
        padding: const EdgeInsets.symmetric(vertical: 16.0),
        child: SizedBox(
          height: 40,
         width: 100,
         child: SvgManager.logo(width: 100, height: 40),
        ),
      ),
    );

static Widget buildUserButton(BuildContext context) {
  final getIt = GetIt.instance;
  final cubit = getIt.get<AuthenticationCubit>();
  final state = cubit.state;

  return AppDesign.buildBlueOutlinedButtonText(
      text: state.login ?? "",
      onPressed: () {
        cubit.signOut();
        final router = AutoRouter.of(context);
        router.navigate(
          MobileEraRoute(),
        );
      });
}

static Widget buildBlueOutlinedButtonText(
        {required final String text,
        required final VoidCallback onPressed,
        final bool enabled = true,
        final bool wrapContent = true}) =>
    CustomOutlinedButton(
      text: text,
      onPressed: onPressed,
      styleType: ButtonStyleType.blue,
      enabled: enabled,
      wrapContent: wrapContent,
    );

static Widget buildBlueFilledButtonText(
        {required final String text,
        required final VoidCallback onPressed,
        final bool enabled = true,
```

```
        final bool wrapContent = true}) =>
    CustomFilledButton(
      text: text,
      onPressed: onPressed,
      styleType: ButtonStyleType.blue,
      enabled: enabled,
      wrapContent: wrapContent,
    );

static Widget buildBlueFilledButton(
        {required final Widget child,
        required final VoidCallback onPressed,
        final bool enabled = true,
        final EdgeInsetsGeometry? padding =
            const EdgeInsets.symmetric(horizontal: 4.0),
        final bool wrapContent = true}) =>
    CustomFilledButtonWidget(
      child: child,
      onPressed: onPressed,
      styleType: ButtonStyleType.blue,
      enabled: enabled,
      wrapContent: wrapContent,
    );

static Widget buildBeigeFilledButtonText(
        {required final String text,
        required final VoidCallback onPressed,
        final bool enabled = true,
        final bool wrapContent = true}) =>
    CustomFilledButton(
      text: text,
      onPressed: onPressed,
      styleType: ButtonStyleType.beige,
      enabled: enabled,
      wrapContent: wrapContent,
    );

static Widget buildBeigeOutlinedButtonText(
        {required final String text,
        required final VoidCallback onPressed,
        final bool wrapContent = true,
        final bool enabled = true}) =>
    CustomOutlinedButton(
      text: text,
      onPressed: onPressed,
      styleType: ButtonStyleType.beige,
      enabled: enabled,
      wrapContent: wrapContent,
    );

static Widget buildBeigeFilledButton(
        {required final Widget child,
        required final VoidCallback onPressed,
        final bool enabled = true,
        final bool wrapContent = true}) =>
    CustomOutlinedButtonWidget(
      child: child,
      onPressed: onPressed,
      styleType: ButtonStyleType.beige,
      enabled: enabled,
      wrapContent: wrapContent,
    );
```

```
  static Widget buildBeigeOutlinedButton(
          {required final Widget child,
          required final VoidCallback onPressed,
          final bool enabled = true,
          final bool wrapContent = true}) =>
      CustomOutlinedButtonWidget(
        child: child,
        onPressed: onPressed,
        styleType: ButtonStyleType.beige,
        enabled: enabled,
        wrapContent: wrapContent,
      );

  /// Outlined input field with onChange callback
  static Widget buildOutlinedInputField({
    required final String hintText,
    required final ValueChanged<String> onChanged,
    final bool obscureText = false,
    final VoidCallback? onIconPressed,
    final TextInputType? keyboardType,
    final TextEditingController? controller,
  }) =>
      TextField(
        controller: controller,
        obscureText: obscureText,
        keyboardType: keyboardType,
        onChanged: onChanged,
        decoration: InputDecoration(
          hintText: hintText,
          hintStyle: AppStyles.placeholderStyle,
          border: OutlineInputBorder(
            borderRadius: BorderRadius.circular(8),
            borderSide: BorderSide(color: AppColors.blueAccent),
          ),
          focusedBorder: OutlineInputBorder(
            borderRadius: BorderRadius.circular(8),
            borderSide: BorderSide(color: AppColors.blueAccent),
          ),
          enabledBorder: OutlineInputBorder(
            borderRadius: BorderRadius.circular(8),
            borderSide: BorderSide(color: AppColors.blueAccent),
          ),
          suffixIcon: onIconPressed != null
              ? IconButton(
                  icon: Icon(
                    obscureText ? Icons.visibility : Icons.visibility_off,
                    color: AppColors.blueAccent,
                  ),
                  onPressed: onIconPressed,
                )
              : null,
        ),
        style: AppStyles.inputTextStyle,
      );
}

class AppStyles {
  static final TextStyle inputTextStyle = GoogleFonts.openSans(
    fontSize: 14,
    color: AppColors.blueAccent,
  );
```

```
static final TextStyle beigeOutlinedButtonTextStyle = GoogleFonts.openSans(
  fontSize: 14,
  color: AppColors.beigeAccent,
  fontWeight: FontWeight.bold,
);

static final TextStyle beigeFilledButtonTextStyle = GoogleFonts.openSans(
  fontSize: 14,
  color: AppColors.white,
  fontWeight: FontWeight.bold,
);

static final TextStyle blueFilledButtonTextStyle = GoogleFonts.openSans(
  fontSize: 14,
  fontWeight: FontWeight.bold,
  color: AppColors.white,
);
static final TextStyle blueOutlinedButtonTextStyle = GoogleFonts.openSans(
  fontSize: 14,
  fontWeight: FontWeight.bold,
  color: AppColors.blueAccent,
);

static final TextStyle titleStyle = GoogleFonts.openSans(
  fontSize: 14,
  fontWeight: FontWeight.w600,
  color: AppColors.fontOnBackground,
  height: 1.4,
);

static final TextStyle bodyStyle = GoogleFonts.openSans(
  fontSize: 14,
  fontWeight: FontWeight.w400,
  color: AppColors.fontOnBackground,
  height: 1.4,
);

static final TextStyle body3Style = GoogleFonts.openSans(
  fontSize: 18,
  fontWeight: FontWeight.bold,
  color: AppColors.blueAccent,
  height: 1.4,
);
static final TextStyle body2Style = GoogleFonts.openSans(
  fontSize: 24,
  fontWeight: FontWeight.bold,
  color: AppColors.blueAccent,
  height: 1.4,
);

static final TextStyle footerStyle = GoogleFonts.openSans(
  fontSize: 12,
  fontWeight: FontWeight.w200,
  fontStyle: FontStyle.italic,
  color: AppColors.fontOnBackground,
  height: 1.4,
);

static final TextStyle copyrightStyle = GoogleFonts.openSans(
  fontSize: 12,
  fontWeight: FontWeight.w600,
```

```
    color: AppColors.fontOnBackground,
    height: 1.4,
  );

  static final TextStyle labelTextStyle = GoogleFonts.openSans(
    fontSize: 14,
    fontWeight: FontWeight.w600,
    color: AppColors.fontOnBackground,
  );

  static final InputDecoration inputDecoration = InputDecoration(
    border: OutlineInputBorder(
      borderRadius: BorderRadius.all(Radius.circular(8)),
    ),
    contentPadding: EdgeInsets.symmetric(horizontal: 12, vertical: 16),
  );

  static final TextStyle placeholderStyle = TextStyle(
    fontSize: 14,
    color: AppColors.blueAccent,
  );
}

class AppColors {
  static const Color background = primaryBlack;

  static const Color blueAccent = Color(0xFF4A6FA5);
  static const Color beigeAccent = Color(0xFFE6D2B5);
  static const Color redBeigeAccent = Color(0xFFE2DAD6);

  static const Color white = Colors.white;
  static const Color black = Colors.black;

  //NEW
  static const Color primaryFont =
      Color(0xFF111315); // Dark black for main font
  static const Color secondaryFont =
      Color(0xFF292C2D); // Dark grey for secondary text
  static const Color tertiaryFont =
      Color(0xFF989898); // Light grey for less emphasis
  static const Color fontOnBackground = primaryWhite;


  static const Color primaryDarkGrey = Color(0xFF2A2C2D);
  static const Color primaryBlack = Color(0xFF111315);
  static const Color primaryWhite = Color(0xFFFFFFFF);
  static const Color primaryMediumGrey = Color(0xFF676767);
  static const Color primaryLightGrey = Color(0xFFABABAB);

  // Accent Colors
  static const Color accentPeach = Color(0xFFF5D8C8);
  static const Color accentLightGrey = Color(0xFFDFE2E3);
  static const Color accentSoftBlue = Color(0xFFC6CCD9);
  static const Color accentBlueGrey = Color(0xFF709ED0);
}

enum ButtonStyleType { blue, beige }

class CustomFilledButton extends StatelessWidget {
  final String text;
  final VoidCallback onPressed;
  final bool enabled;
```

```
  final bool wrapContent;
  final ButtonStyleType styleType;
  final EdgeInsetsGeometry? padding;

  const CustomFilledButton({
    required this.text,
    required this.onPressed,
    this.enabled = true,
    this.wrapContent = true,
    this.styleType = ButtonStyleType.blue,
    this.padding,
    Key? key,
  }) : super(key: key);

  Color get backgroundColor => enabled
      ? (styleType == ButtonStyleType.blue
          ? AppColors.blueAccent
          : AppColors.beigeAccent)
      : (styleType == ButtonStyleType.blue
          ? AppColors.blueAccent.withOpacity(0.5)
          : AppColors.beigeAccent.withOpacity(0.5));

  TextStyle get textStyle => enabled
      ? (styleType == ButtonStyleType.blue
          ? AppStyles.blueFilledButtonTextStyle
          : AppStyles.beigeFilledButtonTextStyle)
      : (styleType == ButtonStyleType.blue
          ? AppStyles.blueFilledButtonTextStyle
          : AppStyles.beigeFilledButtonTextStyle);

  @override
  Widget build(BuildContext context) => SizedBox(
        height: wrapContent ? null : 44,
        child: ElevatedButton(
          onPressed: enabled ? onPressed : null,
          style: ElevatedButton.styleFrom(
            backgroundColor: backgroundColor,
            padding: const EdgeInsets.symmetric(vertical: 16.0),
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(8),
            ),
          ),
          child: Padding(
            padding: padding ?? const EdgeInsets.symmetric(horizontal: 8.0),
            child: Text(
              text,
              style: textStyle,
            ),
          ),
        ),
      );
}

class CustomOutlinedButton extends StatelessWidget {
  final String text;
  final VoidCallback onPressed;
  final bool enabled;
  final bool wrapContent;
  final ButtonStyleType styleType;
  final EdgeInsetsGeometry? padding;

  const CustomOutlinedButton({
```

```
    required this.text,
    required this.onPressed,
    this.enabled = true,
    this.wrapContent = true,
    this.styleType = ButtonStyleType.blue,
    this.padding,
    Key? key,
  }) : super(key: key);

  Color get borderColor => enabled
      ? (styleType == ButtonStyleType.blue
          ? AppColors.blueAccent
          : AppColors.beigeAccent)
      : (styleType == ButtonStyleType.blue
          ? AppColors.blueAccent.withOpacity(0.5)
          : AppColors.beigeAccent.withOpacity(0.5));

  TextStyle get textStyle => enabled
      ? (styleType == ButtonStyleType.blue
          ? AppStyles.blueOutlinedButtonTextStyle
          : AppStyles.beigeOutlinedButtonTextStyle)
      : (styleType == ButtonStyleType.blue
          ? AppStyles.blueOutlinedButtonTextStyle
          : AppStyles.beigeOutlinedButtonTextStyle);

  @override
  Widget build(BuildContext context) => SizedBox(
        height: wrapContent ? null : 44,
        child: OutlinedButton(
          onPressed: enabled ? onPressed : null,
          style: OutlinedButton.styleFrom(
            padding: const EdgeInsets.symmetric(vertical: 16.0),
            side: BorderSide(color: borderColor, width: 2.0),
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(8.0),
            ),
          ),
          child: Padding(
            padding: padding ?? const EdgeInsets.symmetric(horizontal: 8.0),
            child: Text(text, style: textStyle),
          ),
        ),
      );
}

class CustomFilledButtonWidget extends StatelessWidget {
  final Widget child;
  final VoidCallback onPressed;
  final bool enabled;
  final bool wrapContent;
  final ButtonStyleType styleType;
  final EdgeInsetsGeometry? padding;
  final ButtonStyle? customStyle;

  const CustomFilledButtonWidget({
    required this.child,
    required this.onPressed,
    this.enabled = true,
    this.wrapContent = true,
    this.styleType = ButtonStyleType.blue,
    this.padding,
    this.customStyle,
```

```
    Key? key,
  }) : super(key: key);

  Color get backgroundColor => enabled
      ? (styleType == ButtonStyleType.blue
          ? AppColors.blueAccent
          : AppColors.beigeAccent)
      : (styleType == ButtonStyleType.blue
          ? AppColors.blueAccent.withOpacity(0.5)
          : AppColors.beigeAccent.withOpacity(0.5));

  @override
  Widget build(BuildContext context) => SizedBox(
        height: wrapContent ? null : 44,
        child: ElevatedButton(
          onPressed: enabled ? onPressed : null,
          style: customStyle ??
              ElevatedButton.styleFrom(
                backgroundColor: backgroundColor,
                padding: const EdgeInsets.symmetric(vertical: 16.0),
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(8),
                ),
              ),
          child: Padding(
            padding: padding ?? const EdgeInsets.symmetric(horizontal: 8.0),
            child: child,
          ),
        ),
      );
}

class CustomOutlinedButtonWidget extends StatelessWidget {
  final Widget child;
  final VoidCallback onPressed;
  final bool enabled;
  final bool wrapContent;
  final ButtonStyleType styleType;
  final EdgeInsetsGeometry? padding;
  final ButtonStyle? customStyle;

  const CustomOutlinedButtonWidget({
    required this.child,
    required this.onPressed,
    this.enabled = true,
    this.wrapContent = true,
    this.styleType = ButtonStyleType.blue,
    this.padding,
    this.customStyle,
    Key? key,
  }) : super(key: key);

  Color get borderColor => enabled
      ? (styleType == ButtonStyleType.blue
          ? AppColors.blueAccent
          : AppColors.beigeAccent)
      : (styleType == ButtonStyleType.blue
          ? AppColors.blueAccent.withOpacity(0.5)
          : AppColors.beigeAccent.withOpacity(0.5));

  @override
  Widget build(BuildContext context) => SizedBox(
```

```
        height: wrapContent ? null : 44,
        child: OutlinedButton(
          onPressed: enabled ? onPressed : null,
          style: customStyle ??
              OutlinedButton.styleFrom(
                padding: const EdgeInsets.symmetric(vertical: 16.0),
                side: BorderSide(color: borderColor, width: 2.0),
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(8.0),
                ),
              ),
          child: Padding(
            padding: padding ?? const EdgeInsets.symmetric(horizontal: 8.0),
            child: child,
          ),
        ),
      );
}
```

```dart
import "package:auto_route/auto_route.dart";
import "package:flutter/cupertino.dart";
import "package:flutter/material.dart";
import "package:menyusha/app/features/main/screen/menyusha/a4_page_container.dart";
import "package:menyusha/app/features/main/screen/menyusha/hello/hello_screen.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/app_style.dart";

import "../../../../../common/util/svg_manager.dart";

@RoutePage()
class SampleScreen extends StatefulWidget {
  const SampleScreen({super.key});

  @override
  State<SampleScreen> createState() => _SampleScreenState();
}

class _SampleScreenState extends State<SampleScreen> {
  Widget buildContent() =>
      Container(
        child: Column(
          children: [
            SizedBox(height: 16.0),
            Placeholder(),
            SizedBox(height: 16.0),
          ],
        ),
        color: Colors.lightGreen,
      );

  Widget buildBody() =>
      TemplateScreen(child: buildContent());

  @override
  Widget build(final BuildContext context) =>
      A4PageContainer(child: buildBody());
}

class TemplateScreen extends StatelessWidget {
  const TemplateScreen({required this.child, super.key});

  final Widget child;

  Widget buildContent() =>
      Container(
        child: Column(
          children: [
            const SizedBox(height: 16.0),
            child,
            const SizedBox(height: 16.0),
          ],
        ),
      );

  Widget buildBody(final BuildContext context) =>
      Center(
        child: Padding(
          padding: const EdgeInsets.symmetric(horizontal: 20.0, vertical: 10.0),
          child: SingleChildScrollView(
            child: Column(
              children: [
                AppDesign.buildLogo(context),
```

```
                buildContent(),
                AppDesign.buildCopyrightSection(),
              ],
            ),
          ),
        ),
      );

  @override
  Widget build(final BuildContext context) =>
      buildBody(context);
}


class TemplateScreenAdmin extends StatelessWidget {
  const TemplateScreenAdmin({required this.child, super.key});

  final Widget child;

  Widget buildContent() =>
      Container(
        child: Column(
          children: [
            const SizedBox(height: 16.0),
            child,
            const SizedBox(height: 16.0),
          ],
        ),
      );

  Widget buildBody(final BuildContext context) =>
      Center(
        child: SingleChildScrollView(
          child: Column(
            children: [
              Padding(
                padding: const EdgeInsets.symmetric(horizontal: 4.0),
                child: Row(
                  mainAxisAlignment: MainAxisAlignment.spaceBetween,
                  children: [
                    AppDesign.buildLogo(context),
                    AppDesign.buildUserButton(context),
                  ],
                ),
              ),
              buildContent(),
              AppDesign.buildCopyrightSection(),
            ],
          ),
        ),
      );

  @override
  Widget build(final BuildContext context) =>
      buildBody(context);
}
```

```dart
import "package:auto_route/auto_route.dart";
import "package:flutter/cupertino.dart";
import "package:flutter/material.dart";
import "package:menyusha/app/common/info_block.dart";
import "package:menyusha/app/features/main/screen/menyusha/a4_page_container.dart";
import "package:menyusha/app/features/main/screen/menyusha/hello/hello_screen.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/app_style.dart";

import "../../../../../common/util/svg_manager.dart";

@RoutePage()
class PreviewScreen extends StatefulWidget {
  const PreviewScreen({super.key});

  @override
  State<PreviewScreen> createState() => _PreviewScreenState();
}

class _PreviewScreenState extends State<PreviewScreen> {
  Widget _buildCopyrightSection() {
    return Column(
      children: [
        Text(
          "□ 2024",
          style: AppStyles.copyrightStyle,
        ),
        SvgManager.bandana(width: 8, height: 10),
      ],
    );
  }

  Widget wrap({required final Widget child}) => Container(
        child: child,
        constraints: new BoxConstraints(
          minWidth: 400.0,
        ),
        decoration:
            BoxDecoration(borderRadius: BorderRadius.all(Radius.circular(8))),
      );

  @override
  Widget build(final BuildContext context) => A4PageContainer(
        child: SingleChildScrollView(
          child: Column(
            children: [
              SizedBox(
                height: 16,
              ),
              wrap(child: AppDesign.buildLogo(context)),
              SizedBox(
                height: 16,
              ),
              wrap(child: AppDesign.buildLogoSmall(context)),
              SizedBox(
                height: 16,
              ),
              wrap(
                child: AppDesign.buildBlueOutlinedButtonText(
                    text: "Ліміт досягнуто",
                    enabled: true,
                    onPressed: () {
                      final router = AutoRouter.of(context);
```

```
            // Use pushAndRemoveUntil with the root route
          }),
    ),
    SizedBox(
      height: 16,
    ),
/*            wrap(
      child: AppDesign.buildBlueFilledButton(
          child: Text(
            "PREVIEW",
            style: AppStyles.blueFilledButtonTextStyle,
          ),
          enabled: true,
          onPressed: () {}),
    ),*/
    SizedBox(
      height: 16,
    ),
    wrap(
      child: AppDesign.buildBeigeFilledButtonText(
          onPressed: () {}, text: "Registration", wrapContent: true),
    ),
    SizedBox(
      height: 16,
    ),
    wrap(
      child: AppDesign.buildBeigeOutlinedButtonText(
          text: "Ліміт досягнуто",
          onPressed: () {
            final router = AutoRouter.of(context);
            // Use pushAndRemoveUntil with the root route
          }),
    ),
    wrap(
      child: Padding(
        padding: const EdgeInsets.all(8.0),
        child: AppDesign.buildOutlinedInputField(
            hintText: "Email",
            keyboardType: TextInputType.emailAddress,
            onChanged: (final value) => {},
            controller: TextEditingController()),
      ),
    ),
    wrap(child: _buildCopyrightSection()),
    SizedBox(
      height: 16,
    ),
    wrap(
        child: InfoBlock(
      color: AppColors.beigeAccent,
      child: SizedBox(
        width: 100,
        height: 100,
      ),
    )),

    SizedBox(
      height: 16,
    ),
    wrap(
        child: InfoBlock(
          color: AppColors.blueAccent,
```

```
      child: SizedBox(
        width: 100,
        height: 100,
      ),
    )),
SizedBox(
  height: 16,
),
wrap(
  child: Container(
    decoration: BoxDecoration(
        color: AppColors.background,
        borderRadius: BorderRadius.circular(12),
        border: Border.all(
          color: AppColors.beigeAccent,
        )),
    child: SizedBox(
      width: 100,
      height: 100,
    ),
  ),
),
SizedBox(
  height: 16,
),
wrap(
  child: Container(
    decoration: BoxDecoration(
        color: AppColors.background,
        borderRadius: BorderRadius.circular(12),
        border: Border.all(
          color: AppColors.blueAccent,
        )),
    child: SizedBox(
      width: 100,
      height: 100,
    ),
  ),
),

Padding(
  padding: const EdgeInsets.all(16.0),
  child: Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      Text(
        "Primary Colors",
        style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
      ),
      SizedBox(height: 8),
      Row(
        mainAxisAlignment: MainAxisAlignment.spaceAround,
        children: primaryColors.map((color) => ColorBox(color)).toList(),
      ),
      SizedBox(height: 16),
      Text(
        "Accent Colors",
        style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
      ),
      SizedBox(height: 8),
      Row(
        mainAxisAlignment: MainAxisAlignment.spaceAround,
```

```
                    children: accentColors.map((color) => ColorBox(color)).toList(),
                  ),
                ],
              ),
            ),

          ],
        ),
      ),
    );
}



class ColorBox extends StatelessWidget {
  final ColorDetails colorDetails;

  ColorBox(this.colorDetails);

  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        Container(
          width: 50,
          height: 50,
          decoration: BoxDecoration(
            color: colorDetails.color,
            borderRadius: BorderRadius.circular(8),
          ),
        ),
        SizedBox(height: 4),
        Text(
          colorDetails.hexCode,
          style: TextStyle(fontSize: 12),
        ),
      ],
    );
  }
}

class ColorDetails {
  final Color color;
  final String hexCode;

  ColorDetails(this.color, this.hexCode);
}

List<ColorDetails> primaryColors = [
  ColorDetails(Color(0xFF111315), "#111315"),
  ColorDetails(Color(0xFF292C2D), "#292C2D"),
  ColorDetails(Color(0xFFFFFFFF), "#FFFFFF"),
  ColorDetails(Color(0xFF676767), "#676767"),
  ColorDetails(Color(0xFF767676), "#767676"),
];

List<ColorDetails> accentColors = [
  ColorDetails(Color(0xFFCFDDBB), "#CFDDBB"),
  ColorDetails(Color(0xFFE4CDED), "#E4CDED"),
  ColorDetails(Color(0xFFC2DBE9), "#C2DBE9"),
  ColorDetails(Color(0xFFF1CBD0), "#F1CBD0"),
  ColorDetails(Color(0xFFC9CAEF), "#C9CAEF"),
```

```
];
```

```
import "dart:convert";

import "package:auto_route/auto_route.dart";
import "package:flutter/material.dart";
import "package:flutter_bloc/flutter_bloc.dart";
import "package:flutter_markdown/flutter_markdown.dart";
import "package:font_awesome_flutter/font_awesome_flutter.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/common/hover_button.dart";
import "package:menyusha/app/common/info_block.dart";
import "package:menyusha/app/common/responsive_util.dart";
import "package:markdown/markdown.dart" as md;
import "package:menyusha/app/features/main/screen/menyusha/a4_page_container.dart";

import "../../../../../data/firebase/menu/menu_payload.dart";
import "../../base/base_screen.dart";
import "../../base/mobile_frame.dart";
import "../../base/responsive_state.dart";
import "../theme/theme_manager.dart";

class MenuRendererWidget extends StatelessWidget {
  const MenuRendererWidget({
    super.key,
    required this.menu,
  });

  final Menu menu;

  @override
  Widget build(final BuildContext context) {
    final theme =
    MenuThemeManager.getTheme(menu.designPreset);

    return SelectionArea(
      child: A4PageContainer(
        color: theme.backgroundColor,
        child: SingleChildScrollView(
          child: Padding(
            padding: const EdgeInsets.all(32.0),
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.center,
              children: [
                buildTitle(),
                SizedBox(height: 16),
                buildDishGroupWidget(menu.groupedPositions),
              ],
            ),
          ),
        ),
      ),
    );
  }

  Widget buildTitle() {
    final theme = MenuThemeManager.getTheme(menu.designPreset);

    if (false) {
      return Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          SizedBox(
```

```
          width: 350,
          height: 100,
          child: Placeholder(
            color: Colors.redAccent,
            child: Center(
              child: Text(
                "Add image",
                style: theme.titleStyle,
              ),
            ),
          ),
        )
      ],
    );
  } else {
    return Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        SizedBox(
          child: Image.network(menu.titleSrc),
          width: 350,
          height: 100,
        )
      ],
    );
  }
}

Widget buildDishGroupWidget(final Map<String, List<Position>> groupedDishes) {
  final theme = MenuThemeManager.getTheme(menu.designPreset);

  if (groupedDishes.isEmpty) {
    return Padding(
      padding: const EdgeInsets.all(8.0),
      child: GestureDetector(
        child: SizedBox(
          width: 350,
          height: 100,
          child: Placeholder(
            color: Colors.red,
            child: Center(
              child: Text(
                "No Dishes",
                style: theme.titleStyle,
              ),
            ),
          ),
        ),
      ),
    );
  } else {
    return Column(
      children: groupedDishes.entries.map((final entry) {
        final group = entry.key;
        final dishes = entry.value;

        return Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
```

```
                Text(group, style: theme.groupTitleStyle),
              ],
            ),
            ...dishes.map((final dish) => buildDishItem(dish)).toList(),
            SizedBox(height: 16), // Add some space between groups
          ],
        );
      }).toList(),
    );
  }
  }

  Widget buildDishItem(final Position dish) {
    final theme = MenuThemeManager.getTheme(menu.designPreset);

    return Padding(
      padding: const EdgeInsets.symmetric(vertical: 8.0),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: [
              Text(
                dish.title,
                style: theme.itemTitleStyle,
              ),
              Text(
                '\₴${dish.price}',
                style: theme.itemPriceStyle,
              ),
            ],
          ),
          SizedBox(height: 4),
          Text('${dish.description} ${dish.output}',
              style: theme.itemDescriptionStyle),
        ],
      ),
    );
  }
}

class MenuEditorWidget extends StatefulWidget {
  const MenuEditorWidget({
    super.key,
    required this.menu,
    required this.onSave,
  });

  final Menu menu;
  final ValueChanged<Menu> onSave;

  @override
  _MenuEditorState createState() => _MenuEditorState();
}

class _MenuEditorState extends State<MenuEditorWidget> {
  late TextEditingController _jsonController;

  @override
  void didUpdateWidget(MenuEditorWidget oldWidget) {
    super.didUpdateWidget(oldWidget);
```

```
  // Check if the menu has changed
  if (oldWidget.menu != widget.menu) {
    setState(() {
      // Update the text in the controller with the new menu data
      _jsonController.text = prettyPrintJson(widget.menu.toJson());
    });
  }
}

@override
void initState() {
  super.initState();
  // Initialize with the current menu and set the controller to display the formatted JSON
  _jsonController = TextEditingController(
    text: prettyPrintJson(widget.menu.toJson()),
  );
}

@override
void dispose() {
  _jsonController.dispose();
  super.dispose();
}

// Call this when the user clicks Save to return the updated model
void _handleSave() {
  try {
    // Parse the JSON input and update the menu model
    final updatedJson = jsonDecode(_jsonController.text);
    final updatedMenu = Menu.fromJson(updatedJson);

    // Trigger the save callback with the updated model
    widget.onSave(updatedMenu);
  } catch (e) {
    // Handle JSON parsing errors
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Invalid JSON: ${e.toString()}')),
    );
  }
}

// Call this when the user clicks Cancel to reset the input to the original model
void _handleCancel() {
  setState(() {
    _jsonController.text = prettyPrintJson(widget.menu.toJson());
  });
}

// Helper function to format JSON
String prettyPrintJson(Map<String, dynamic> json) {
  const encoder = JsonEncoder.withIndent('  '); // Indentation with 2 spaces
  return encoder.convert(json);
}

@override
Widget build(BuildContext context) {
  final theme = MenuThemeManager.getTheme(widget.menu.designPreset);

  return A4PageContainer(
    color: theme.backgroundColor,
    child: Padding(
      padding: const EdgeInsets.all(16.0),
```

```
  child: Column(
    crossAxisAlignment: CrossAxisAlignment.start,
    mainAxisAlignment: MainAxisAlignment.start,
    children: [
      // Input field for JSON
      Expanded(
        child: TextField(
          controller: _jsonController,
          maxLines: null,
          // Allows TextField to grow vertically
          expands: true,
          // Expands to fill the available height
          textAlign: TextAlign.start,

          style: theme.itemDescriptionStyle,
          decoration: InputDecoration(
            border: OutlineInputBorder(),
            labelText: 'Edit Menu JSON',
          ),
        ),
      ),
      SizedBox(height: 16),
      // Save and Cancel buttons
      Row(
        mainAxisAlignment: MainAxisAlignment.end,
        children: [
          HoverButton(
            onTap: _handleCancel,
            onDoubleTap: () {},
            child: const Padding(
              padding: EdgeInsets.all(4),
              child: Text(
                "cancel",
                style: TextStyle(
                  fontWeight: FontWeight.bold,
                  color: Color(0xFFDFD3C3),
                  fontSize: 16,
                ),
              ),
            ),
          ),
          SizedBox(width: 16),
          HoverButton(
            onTap: _handleSave,
            onDoubleTap: () {},
            child: const Padding(
              padding: EdgeInsets.all(4),
              child: Text(
                "Save",
                style: TextStyle(
                  fontWeight: FontWeight.bold,
                  color: Color(0xFFDFD3C3),
                  fontSize: 16,
                ),
              ),
            ),
          ),
        ],
      ),
    ],
  ),
),
```

```
    );
  }
}
```

```dart
import "dart:convert";

import "package:auto_route/auto_route.dart";
import "package:flutter/material.dart";
import "package:flutter_bloc/flutter_bloc.dart";
import "package:flutter_markdown/flutter_markdown.dart";
import "package:font_awesome_flutter/font_awesome_flutter.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/sample_screen.dart";
import "package:menyusha/app/features/main/screen/notfound/not_found.dart";
import "package:menyusha/app/root/app_router.dart";
import "package:uuid/uuid.dart";

import "../../../../../common/hover_button.dart";
import "../../../../../common/util/svg_manager.dart";
import "../../base/responsive_state.dart";
import "../theme/app_style.dart";

@RoutePage()
class HelloScreen extends StatefulWidget {
  HelloScreen({super.key});

  final getIt = GetIt.instance;
  final NullCubit cubit = NullCubit();

  @override
  _HelloScreenState createState() => _HelloScreenState(cubit);
}

class _HelloScreenState
    extends ResponsiveState<HelloScreen, NullState, NullCubit> {
  _HelloScreenState(super.cubit);

  @override
  void onStateChange(BuildContext context, NullState state) {}

  @override
  Widget buildDesktopLayout(BuildContext context, NullState state) =>
      buildBody();

  @override
  Widget buildMobileLayout(BuildContext context, NullState state) =>
      buildBody();

  Widget buildBody() {
    return TemplateScreen(
      child: Column(
        children: [
          _buildTextSection(),
          const SizedBox(height: 16.0),
          _buildRegistrationRow(),
          const SizedBox(height: 66.0),
          _buildFooterText(),
          const SizedBox(height: 32.0),
        ],
      ),
    );
  }

  Widget _buildTextSection() {
    return Column(
```

```
      children: [
        Text(
          'Вітаю на сторінці нашого проєкту — онлайн-меню, створеного для кафе, ресторанів та інших закл
          style: AppStyles.titleStyle,
          textAlign: TextAlign.start,
        ),
        const SizedBox(height: 8.0),
        Text(
          'Ми допомагаємо легко налаштовувати меню, ділитися ним з клієнтами та підтримувати актуальніст
          style: AppStyles.bodyStyle,
          textAlign: TextAlign.start,
        ),
      ],
    );
  }

  Widget _buildRegistrationRow() {
    return Row(
      mainAxisSize: MainAxisSize.min,
      children: [
        Flexible(
          child: Text(
            'Щоб почати створення власного меню, пройдіть просту реєстрацію. Натисніть кнопку, щоб розпо
            style: AppStyles.bodyStyle,
          ),
        ),
        const SizedBox(width: 8.0),
        _buildHoverButton(),
      ],
    );
  }

  Widget _buildHoverButton() {
    return AppDesign.buildBlueFilledButtonText(
        text: "Увійти",
        onPressed: () {
          final router = AutoRouter.of(context);
          router.navigate(LoginRoute());
        });
  }

  Widget _buildFooterText() {
    return Text(
      'Застосунок знаходиться у розробці, але скоро ви зможете легко створювати меню, ділитися ним та он
      style: AppStyles.footerStyle,
      textAlign: TextAlign.start,
    );
  }
}
```

```dart
import "dart:convert";

import "package:auto_route/auto_route.dart";
import "package:flutter/material.dart";
import "package:flutter_bloc/flutter_bloc.dart";
import "package:flutter_markdown/flutter_markdown.dart";
import "package:font_awesome_flutter/font_awesome_flutter.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/features/main/screen/menyusha/login/auth_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/login/login_screen.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/sample_screen.dart";

import "../../../../../data/firebase/auth/auth_state.dart";
import "../../base/responsive_state.dart";
import "../a4_page_container.dart";
import "../theme/app_style.dart";

@RoutePage()
class RegistrationScreen extends StatefulWidget {
  RegistrationScreen({super.key});

  final getIt = GetIt.instance;
  late final AuthenticationCubit cubit = getIt.get<AuthenticationCubit>();

  @override
  _RegistrationScreenState createState() => _RegistrationScreenState(cubit);
}

class _RegistrationScreenState extends ResponsiveState<RegistrationScreen,
    AuthState, AuthenticationCubit> {
  _RegistrationScreenState(super.cubit);

  final TextEditingController emailController = TextEditingController();
  final TextEditingController passwordController = TextEditingController();

  @override
  void onStateChange(
    final BuildContext context,
    final AuthState state,
  ) {}

  @override
  Widget buildDesktopLayout(
    final BuildContext context,
    final AuthState state,
  ) =>
      Container(
        child: A4PageContainer(
          child: buildBody(state: state),
          color: Colors.white,
        ),
        color: Colors.white,
      );

  @override
  Widget buildMobileLayout(
    final BuildContext context,
    final AuthState state,
  ) =>
      buildBody(state: state);
```

```
Widget buildBody({required final AuthState state}) => TemplateScreen(
      child: Column(
        mainAxisSize: MainAxisSize.min,
        children: [
          buildLoginFormBody(context: context, state: state),
          SizedBox(height: 16),
          DividerWidget(width: 200, height: 1),
          SizedBox(height: 16),
          buildAuthMethods(),
        ],
      ),
    );

/// Builds the login form with email and password fields
Widget buildLoginFormBody(
    {required final BuildContext context, required final AuthState state}) {
  final bool isEnabledLogin;
  if (state is AuthInitial) {
    isEnabledLogin = state.email.isNotEmpty && state.password.isNotEmpty;
  } else {
    isEnabledLogin = false;
  }

  final bool isLoginLoading;
  if (state is AuthLoading) {
    isLoginLoading = true;
  } else {
    isLoginLoading = false;
  }

  return Column(
    children: [
      Padding(
        padding: const EdgeInsets.all(8.0),
        child: AppDesign.buildOutlinedInputField(
            hintText: "Email",
            keyboardType: TextInputType.emailAddress,
            onChanged: (final value) => {widget.cubit.typeEmail(value)},
            controller: emailController),
      ),
      Padding(
        padding: const EdgeInsets.all(8.0),
        child: AppDesign.buildOutlinedInputField(
            hintText: "Password",
            obscureText: true,
            onChanged: (final value) => {widget.cubit.typePassword(value)},
            controller: passwordController),
      ),
      Padding(
        padding: const EdgeInsets.all(8.0),
        child: Column(
          children: [
            AppDesign.buildBeigeFilledButton(
              onPressed: () {
                if (state is AuthInitial) {
                  widget.cubit.registerWithEmailAndPassword(
                      state.email, state.password);
                  return;
                } else {
                  return;
                }
              },
```

```
                child: SizedBox(
                  width: 150,
                  child: Center(
                      child: isLoginLoading
                          ? Center(
                              child: const SizedBox(
                              height: 20,
                              width: 20,
                              child: CircularProgressIndicator(
                                strokeWidth: 2,
                                color: Colors.white,
                              ),
                            ))
                          : Text("Confirm",
                              style: AppStyles.beigeFilledButtonTextStyle)),
                ),
              ),
            ],
          ),
        ),
        SizedBox(height: 8.0),
      ],
    );
  }

  /// Builds social authentication methods, such as Google login
  Widget buildAuthMethods() => Padding(
        padding: const EdgeInsets.all(8.0),
        child: Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            AppDesign.buildBlueFilledButton(
              onPressed: widget.cubit.signInWithGoogle,
              child: Center(
                child: Icon(
                  FontAwesomeIcons.google,
                  color: Colors.white,
                  size: 30,
                ),
              ),
            ),
          ],
        ),
      );
}
```

```dart
import "dart:convert";

import "package:auto_route/auto_route.dart";
import "package:flutter/cupertino.dart";
import "package:flutter/material.dart";
import "package:flutter_bloc/flutter_bloc.dart";
import "package:flutter_hooks/flutter_hooks.dart";
import "package:flutter_markdown/flutter_markdown.dart";
import "package:font_awesome_flutter/font_awesome_flutter.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/common/hover_button.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/sample_screen.dart";
import "package:menyusha/main.dart";

import "../../../../../common/info_block.dart";
import "../../../../../data/firebase/auth/auth_state.dart";
import "auth_cubit.dart";
import "../../../../../root/app_router.dart";
import "../../base/responsive_state.dart";
import "dart:math" as math;

import "../a4_page_container.dart";
import "../theme/app_style.dart";

@RoutePage()
class LoginScreen extends StatefulWidget {
  LoginScreen({super.key});

  final getIt = GetIt.instance;

  late final AuthenticationCubit cubit = getIt<AuthenticationCubit>();

  @override
  _LoginScreenState createState() => _LoginScreenState(cubit);
}

class _LoginScreenState
    extends ResponsiveState<LoginScreen, AuthState, AuthenticationCubit> {
  _LoginScreenState(super.cubit);

  final TextEditingController emailController = TextEditingController();
  final TextEditingController passwordController = TextEditingController();

  @override
  void initState() {
    // Check initial auth state when the widget starts
    if (widget.cubit.state is AuthSuccess) {
      final router = AutoRouter.of(context);
      WidgetsBinding.instance.addPostFrameCallback((_) {
        router.navigate(AdminContainerRoute());
      });
    }
  }

  @override
  void onStateChange(
    final BuildContext context,
    final AuthState state,
  ) {
    if (state is AuthSuccess) {
      // Navigate to the home screen or any other screen on successful login
```

```
    final router = AutoRouter.of(context);

    router.navigate(AdminContainerRoute());
    print("AuthenticationSuccess: Navigating to home");
  } else if (state is AuthFailed) {
    // Show a snackbar or dialog with the error message
    final snackBar = SnackBar(
      backgroundColor: Colors.transparent,
      elevation: 0,
      behavior: SnackBarBehavior.floating,
      content: Column(
        mainAxisSize: MainAxisSize.min,
        mainAxisAlignment: MainAxisAlignment.center,
        crossAxisAlignment: CrossAxisAlignment.center,
        children: [
          InfoBlock(
            width: 350,
            padding: EdgeInsets.symmetric(horizontal: 16, vertical: 8),
            color: Colors.black26.withAlpha(80),
            child: Text(state.error ?? 'Authentication Failed',
                style: GoogleFonts.robotoMono(
                    color: AppColors.blueAccent,
                    fontWeight: FontWeight.bold,
                    letterSpacing: 2,
                    fontSize: 12)),
          ),
        ],
      ),
    );

    ScaffoldMessenger.of(context).showSnackBar(snackBar);
    print("AuthenticationFailure: ${state.error}");
    widget.cubit
        .restoreFailedLogin(emailController.text, passwordController.text);
  } else if (state is AuthSignedOut) {
    context.router.replace(HelloRoute());
  }
}

@override
Widget buildDesktopLayout(
  final BuildContext context,
  final AuthState state,
) =>
    A4PageContainer(
      child: buildBody(context: context, state: state),
    );

@override
Widget buildMobileLayout(
  final BuildContext context,
  final AuthState state,
) =>
    buildBody(context: context, state: state);

Widget buildBody(
        {required final BuildContext context,
        required final AuthState state}) =>
    TemplateScreen(
      child: Column(
        mainAxisSize: MainAxisSize.min,
        children: [
```

```
          buildLoginFormBody(context: context, state: state),
          SizedBox(height: 16),
          DividerWidget(width: 200, height: 1),
          SizedBox(height: 16),
          buildAuthMethods(),
        ],
      ),
    );

Widget buildLoginFormBody(
    {required final BuildContext context, required final AuthState state}) {
  final bool isEnabledLogin;
  if (state is AuthInitial) {
    isEnabledLogin = state.email.isNotEmpty && state.password.isNotEmpty;
  } else {
    isEnabledLogin = false;
  }

  final bool isLoginLoading;
  if (state is AuthLoading) {
    isLoginLoading = true;
  } else {
    isLoginLoading = false;
  }

  return Column(
    children: [
      Padding(
        padding: const EdgeInsets.all(8.0),
        child: AppDesign.buildOutlinedInputField(
            hintText: "Email",
            keyboardType: TextInputType.emailAddress,
            onChanged: (final value) => {widget.cubit.typeEmail(value)},
            controller: emailController),
      ),
      Padding(
        padding: const EdgeInsets.all(8.0),
        child: AppDesign.buildOutlinedInputField(
            hintText: "Password",
            obscureText: true,
            onChanged: (final value) => {widget.cubit.typePassword(value)},
            controller: passwordController),
      ),
      Padding(
        padding: const EdgeInsets.all(8.0),
        child: Column(
          children: [
            isLoginLoading
                ? AppDesign.buildBlueFilledButton(
                    onPressed: () {},
                    child: SizedBox(
                        width: 150,
                        child: Center(
                            child: const SizedBox(
                          height: 20,
                          width: 20,
                          child: CircularProgressIndicator(
                            strokeWidth: 2,
                            color: Colors.white,
                          ),
                        ))),
                    wrapContent: true,
```

```
                    enabled: isEnabledLogin)
                : AppDesign.buildBlueFilledButtonText(
                    text: "Увійти",
                    onPressed: () {
                      if (state is AuthInitial) {
                        widget.cubit.signInWithEmailAndPassword(
                            state.email, state.password);
                        return;
                      } else {
                        return;
                      }
                    }),
            SizedBox(height: 8.0),
            AppDesign.buildBeigeFilledButtonText(
                onPressed: () {
                  final router = AutoRouter.of(context);
                  router.navigate(RegistrationRoute());
                },
                text: "Registration",
                wrapContent: true),
            if (!buildConfig.isProduction)
              Column(
                children: [
                  SizedBox(height: 8.0),
                  AppDesign.buildBeigeFilledButtonText(
                      onPressed: () {
                        if (state is AuthInitial) {
                          widget.cubit
                              .adminSignIn(state.email, state.password);
                          return;
                        } else {
                          return;
                        }
                      },
                      text: "Admin sign in",
                      wrapContent: true)
                ],
              )
          ],
        ),
      ),
      SizedBox(height: 8.0),
    ],
  );
}

Widget buildAuthMethods() => Padding(
      padding: const EdgeInsets.all(8.0),
      child: Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          AppDesign.buildBlueFilledButton(
              onPressed: widget.cubit.signInWithGoogle,
              child: Center(
                child: Icon(
                  FontAwesomeIcons.google,
                  color: Colors.white,
                  size: 30,
                ),
              ),
              wrapContent: true),
        ],
```

```
        ),
      );
}

class DividerWidget extends StatelessWidget {
  const DividerWidget({
    Key? key,
    required this.width,
    this.height = 1,
  }) : super(key: key);

  final double width;
  final double height;

  @override
  Widget build(BuildContext context) {
    return Container(
      width: width,
      height: height,
      decoration: BoxDecoration(
        borderRadius: BorderRadius.circular(100),
        color: Colors.white,
      ),
    );
  }
}

class WidgetSwitcher extends HookWidget {
  final Widget widget1;
  final Widget widget2;
  final bool showWidget;

  WidgetSwitcher(
      {required this.widget1, required this.widget2, required this.showWidget});

  @override
  Widget build(BuildContext context) {
    return Center(
      child: Column(
        children: [
          widget1,
          Visibility(
            visible: !showWidget,
            child: Column(
              children: [
                SizedBox(
                  height: 8,
                ),
                DividerWidget(
                  width: 64,
                ),
                SizedBox(
                  height: 8,
                ),
                widget2
              ],
            ),
          ),
          /*   AnimatedSwitcher(
            duration: const Duration(seconds: 1),
            transitionBuilder: (Widget child, Animation<double> animation) {
              return FadeTransition(
```

```
                opacity: animation,
                child: child,
              );
            },
            child: showWidget
                ? widget1
                : Column(
                    children: [
                      widget1,
                      SizedBox(
                        height: 8,
                      ),
                      DividerWidget(),
                      SizedBox(
                        height: 8,
                      ),
                      widget2
                    ],
                  ),
          ),*/
        ],
      ),
    );
  }
}
```

```dart
import "dart:async";

import "package:firebase_auth/firebase_auth.dart";
import "package:flutter_bloc/flutter_bloc.dart";
import "package:get_it/get_it.dart";
import "package:google_sign_in/google_sign_in.dart";
import "package:menyusha/app/data/firebase/user/user_payload.dart";
import "package:menyusha/app/data/firebase/user/user_payload_repository.dart";
import "package:menyusha/app/data/firebase/user_manager.dart";

import "../../../../../data/firebase/auth/auth_state.dart";

class AuthenticationCubit extends Cubit<AuthState> {
  AuthenticationCubit(this._firebaseAuth)
      : super(AuthState.authInitial(true, "", "")) {
    init();
  }
  final getIt = GetIt.instance;

  late final UserPayloadRepository userRepository =
      getIt<UserPayloadRepository>();

  late final UserManager userManager = getIt<UserManager>();

  final FirebaseAuth _firebaseAuth;

  StreamSubscription<User?>? _authStateSubscription;

  final GoogleSignIn _googleSignIn = GoogleSignIn();

  bool isLoggedIn() => _firebaseAuth.currentUser != null;

  void init() {
    _authStateSubscription = _firebaseAuth.authStateChanges().listen(
      (final user) async {
        if (user == null) {
          print("User is currently signed out!");
          emit(AuthState.authInitial(true, "", ""));
        } else {
          // Optional: fetch or create user payload if needed
          final userPayload = await userRepository
              .getUserByUID(user!.uid);

          loginUser(user,userPayload!);
        }
      },
      onError: (final error) {
        print("Error listening to auth state changes: $error");
        emit(AuthState.authFailed(error.toString()));
      },
    );
  }

  void loginUser(final User user, final UserPayload userPayload) {
    print("User is signed in!");
    userManager.saveUser(userPayload);
    emit(AuthState.authSuccess(user, userPayload));
  }
  // login and registration

  Future<void> typeEmail(final String value) async {
    if (state is AuthInitial) {
```

```
    final authState = state as AuthInitial;
    emit(
      authState.copyWith(email: value),
    );
  } else {
    emit(AuthInitial(true, "", ""));
  }
}

Future<void> typePassword(final String value) async {
  if (state is AuthInitial) {
    final authState = state as AuthInitial;
    emit(
      authState.copyWith(password: value),
    );
  } else {
    emit(AuthInitial(true, "", ""));
  }
}

// login
Future<void> restoreFailedLogin(
    final String email, final String password) async {
  if (state is AuthInitial) {
    final authState = state as AuthInitial;
    emit(
      authState.copyWith(email: email, password: password),
    );
  } else {
    emit(AuthInitial(true, email, password));
  }
}

Future<void> switchLogin() async {
  if (state is AuthInitial) {
    final authState = state as AuthInitial;
    emit(
      authState.copyWith(
        loginOpened: !authState.loginOpened,
      ),
    );
  } else {
    emit(AuthInitial(true, "", ""));
  }
}

Future<void> signOut() async {
  await _firebaseAuth.signOut();
  await _googleSignIn.signOut();
  await userManager.logout();
  emit(AuthState.authSignedOut()); // Emitting initial state after sign out
  switchLogin();
}

Future<void> signInWithGoogle() async {
  emit(AuthState.authLoading());
  try {
    final GoogleSignInAccount? googleUser = await _googleSignIn.signIn();
    if (googleUser == null) {
      emit(AuthState.authFailed("Google sign-in canceled by user"));
      return;
    }
```

```
    final GoogleSignInAuthentication googleAuth =
        await googleUser.authentication;

    final AuthCredential credential = GoogleAuthProvider.credential(
      accessToken: googleAuth.accessToken,
      idToken: googleAuth.idToken,
    );

    final UserCredential userCredential =
        await _firebaseAuth.signInWithCredential(credential);

    // Optional: fetch or create user payload if needed
    final userPayload = await userRepository
        .findOrCreateUserPayloadByUID(userCredential.user!.uid);

    loginUser(userCredential.user!,userPayload!);
  } catch (e) {
    emit(AuthState.authFailed(_processSignInError(e)));
  }
}

Future<void> adminSignIn(
    final String email, final String password) async {
  emit(AuthState.authLoading());
  try {
    final userCredential = await _signIn("lenyk5665@gmail.com", "123123");
    final userPayload = await userRepository
        .getUserByUID(userCredential.user!.uid);
    loginUser(userCredential.user!,userPayload!);
  } catch (e) {
    emit(AuthState.authFailed(_processSignInError(e)));
  }
}

Future<void> signInWithEmailAndPassword(
    final String email, final String password) async {
  emit(AuthState.authLoading());
  try {
    final userCredential = await _signIn(email, password);
    final userPayload = await userRepository
        .getUserByUID(userCredential.user!.uid);
    loginUser(userCredential.user!,userPayload!);
  } catch (e) {
    emit(AuthState.authFailed(_processSignInError(e)));
  }
}

Future<void> signInOrRegisterWithEmailAndPassword(
    final String email, final String password) async {
  emit(AuthState.authLoading());
  try {
    final userCredential = await _attemptSignIn(email, password);

     final userPayload = await userRepository
         .findOrCreateUserPayloadByUID(userCredential.user!.uid);
    loginUser(userCredential.user!,userPayload!);
  } catch (e) {
    emit(AuthState.authFailed(_processSignInError(e)));
  }
}
```

```
Future<void> registerWithEmailAndPassword(
    final String email, final String password) async {
  emit(AuthState.authLoading());
  try {
    final userCredential = await _attemptRegister(email, password);
    final userPayload = await userRepository
        .createUser(userCredential.user!.uid);
    loginUser(userCredential.user!, userPayload!);
  } catch (e) {
    emit(AuthState.authFailed(_processSignInError(e)));
  }
}

String _processSignInError(final error) {
  if (error is FirebaseAuthException) {
    return _handleFirebaseAuthException(error);
  }
  return "Unexpected error: ${error.toString()}";
}

String _handleFirebaseAuthException(final FirebaseAuthException exception) {
  final code = exception.code;
  return _handleRegistrationError(code);
}

String _handleRegistrationError(final String? code) {
  switch (code) {
    case "email-already-in-use":
      return "The email address is already in use by another account.";
    case "invalid-email":
      return "The email address is not valid.";
    case "operation-not-allowed":
      return "Email/password accounts are not enabled.";
    case "weak-password":
      return "The password is too weak.";
    default:
      return _errorCodeToMessage(code);
  }
}

String _errorCodeToMessage(final String? code) {
  switch (code) {
    case "invalid-email":
      return "Invalid email";
    case "user-not-found":
      return "User not found. Please register.";
    case "wrong-password":
      return "Invalid password. Please try again.";
    case "weak-password":
      return "The password is too weak.";
    case null:
      return "Unknown error";
    default:
      return "Unexpected error: $code";
  }
}

Future<UserCredential> _attemptRegister(
    final String email, final String password) async {
  try {
    return await _register(email, password);
  } on FirebaseAuthException catch (e) {
```

```
      throw e;
    }
  }

  Future<UserCredential> _attemptSignIn(
      final String email, final String password) async {
    try {
      return await _signIn(email, password);
    } on FirebaseAuthException catch (e) {
      if (e.code == "user-not-found") {
        return _register(email, password);
      }
      throw e;
    }
  }

  Future<UserCredential> _signIn(
          final String email, final String password) async =>
      _firebaseAuth.signInWithEmailAndPassword(
        email: email,
        password: password,
      );

  Future<UserCredential> _register(
          final String email, final String password) async =>
      _firebaseAuth.createUserWithEmailAndPassword(
        email: email,
        password: password,
      );

  @override
  Future<void> close() {
    _authStateSubscription?.cancel();
    return super.close();
  }
}
```

```dart
import "package:auto_route/auto_route.dart";
import "package:flutter/material.dart";
import "package:flutter_bloc/flutter_bloc.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/common/info_block.dart";
import "package:menyusha/app/features/main/screen/base/mobile_frame.dart";

import "../../../../common/hover_button.dart";
import "../../../../root/app_router.dart";
import "../base/base_screen.dart";
import "../base/responsive_state.dart";

@RoutePage()
class NotFoundScreen extends StatefulWidget {
  NotFoundScreen({super.key});

  final getIt = GetIt.instance;

  final cubit = NullCubit();

  @override
  _NotFoundScreenState createState() => _NotFoundScreenState(cubit);
}

class _NotFoundScreenState
    extends ResponsiveState<NotFoundScreen, NullState, NullCubit> {
  _NotFoundScreenState(super.cubit);

  @override
  Widget buildDesktopLayout(
          final BuildContext context, final NullState state) =>
      buildBody(state: state);

  @override
  Widget buildMobileLayout(final BuildContext context, final NullState state) =>
      buildBody(state: state);

  Widget buildBody({required final NullState state}) => MobileFrame(
          child: SingleChildScrollView(
  child: Column(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      const Text(
        '404',
        style: TextStyle(fontSize: 72, fontWeight: FontWeight.bold, color: Colors.black),
      ),
      const SizedBox(height: 16),
      const Text(
        'The page you are looking for does not exist.',
        style: TextStyle(fontSize: 24, color: Colors.black),
        textAlign: TextAlign.center,

      ),
      const SizedBox(height: 32),
      HoverButton(
        onTap: () {
          final router = AutoRouter.of(context);
          router.navigate(HelloRoute());
        },
        onDoubleTap: () {},
        child: Padding(
```

```
          padding: EdgeInsets.symmetric(horizontal: 12, vertical: 6),
          child: Text(
            "home",
            style: GoogleFonts.robotoMono(
              fontSize: 14,
              color: Colors.white,
              fontWeight: FontWeight.bold,
            ),
          ),
        ),
      ),
    ],
  ),
          ),
        );
}

// Define a NullState that doesn't hold any specific data
class NullState {}

// Define a NullCubit that doesn't perform any actions
class NullCubit extends Cubit<NullState> {
  NullCubit() : super(NullState());
}
```

```dart
import "package:auto_route/auto_route.dart";
import "package:flutter/cupertino.dart";
import "package:flutter/material.dart";
import "package:flutter_bloc/flutter_bloc.dart";
import "package:font_awesome_flutter/font_awesome_flutter.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/features/main/screen/notfound/not_found.dart";

import "base/base_screen.dart";
import "base/responsive_state.dart";
import "menyusha/theme/app_style.dart";

@RoutePage()
class MobileEraScreen extends StatefulWidget {
  MobileEraScreen({super.key});

  @override
  _MobileEraScreenState createState() => _MobileEraScreenState();
}

class _MobileEraScreenState extends State<MobileEraScreen> {
  @override
  Widget build(final BuildContext context) => BaseEntryPointScreen(
        child: Container(
          child: AutoRouter(),
          color: AppColors.background,
        ),
      );
}
```

```
import "package:flutter/material.dart";

class MobileFrame extends StatelessWidget {
  const MobileFrame({
    required this.child,
    super.key,
  });

  final Widget child;

  @override
  Widget build(final BuildContext context) => SizedBox(
        width: 375,
        height: 667,
        child: Center(child: child),
      );
}
```

```dart
import "package:flutter/material.dart";
import "package:flutter_svg/svg.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/app_style.dart";

class BaseEntryPointScreen extends StatelessWidget {
  const BaseEntryPointScreen({
    required this.child,
    this.backgroundColor = AppColors.background,
    super.key,
  });

  final Widget child;
  final Color backgroundColor;

  @override
  Widget build(final BuildContext context) => Container(
    color: backgroundColor,
    child: Scaffold(
      body: Stack(
        children: [
          Center(child: Container(
              child: AppDesign.buildLogo(context))),
          Container(
            child: child,
          ),
        ],
      ),
      backgroundColor: backgroundColor,
    ),
  );
}
```

```dart
import "package:flutter/widgets.dart";
import "package:flutter_bloc/flutter_bloc.dart";
import "package:menyusha/app/common/responsive_util.dart";
import "package:menyusha/app/common/util/unit.dart";

abstract class ResponsiveState<T extends StatefulWidget, S, C extends Cubit<S>>
    extends State<T> {

  ResponsiveState(this.cubit);
  final C cubit;

  @override
  Widget build(final BuildContext context) => BlocConsumer<C, S>(
    bloc: cubit,
    builder: _buildResponsiveLayout,
    listener: onStateChange,
  );

  void onStateChange(final BuildContext context, final S state) => Unit;

  Widget buildDesktopLayout(final BuildContext context, final S state);
  Widget buildMobileLayout(final BuildContext context, final S state);

  Widget _buildResponsiveLayout(final BuildContext context, final S state) =>
      ResponsiveUtil.isDesktopFull(context)
          ? buildDesktopLayout(context, state)
          : buildMobileLayout(context, state);
}
```

```dart
import "package:flutter/widgets.dart";
import "package:flutter_bloc/flutter_bloc.dart";

import 'responsive_state.dart';

class TestState {

}

class TestCubit extends Cubit<TestState> {
  TestCubit(super.initialState);
}

class MyResponsiveWidget extends StatefulWidget {
  MyResponsiveWidget({super.key});

  late final TestCubit cubit = TestCubit(TestState());

  @override
  MyResponsiveWidgetState createState() => MyResponsiveWidgetState(cubit);
}

class MyResponsiveWidgetState extends ResponsiveState<MyResponsiveWidget, TestState, TestCubit> {
  MyResponsiveWidgetState(super.cubit);

  @override
  void onStateChange(
    final BuildContext context,
    final TestState state,
  ) {

  }
  @override
  Widget buildDesktopLayout(
    final BuildContext context,
    final TestState state,
  ) => Container();

  @override
  Widget buildMobileLayout(
    final BuildContext context,
    final TestState state,
  ) => Container();
}
```

```dart
import "package:flutter/widgets.dart";
import "package:flutter_svg/flutter_svg.dart";

class BackgroundSvgContainer extends StatelessWidget {
  final Widget child;
  final String svgPath;

  const BackgroundSvgContainer({
    Key? key,
    required this.child,
    required this.svgPath,
  }) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Stack(
      children: [
        Positioned.fill(
          child: SvgPicture.asset(
            svgPath,
            fit: BoxFit.cover,
          ),
        ),
        child,
      ],
    );
  }
}
```

```dart
import "package:flutter_dotenv/flutter_dotenv.dart";

// DotEnv dotenv = DotEnv() is automatically called during import.
// If you want to load multiple dotenv files or name your dotenv object differently, you can do the foll
// DotEnv another_dotenv = DotEnv()

Future<void> environmentInit() async => await dotenv.load(fileName: "dotenv.env");

String? getGithubToken() => dotenv.env["githubToken"];

String? getGithubLogin() => dotenv.env["login"];

String? getGithubRepository() => dotenv.env["repository"];

bool getIsProduction() => dotenv.getBool("isProduction");
```

```dart
import "package:dotted_border/dotted_border.dart";
import "package:flutter/material.dart";

class RoundedContainerWidget extends StatelessWidget {
  const RoundedContainerWidget({super.key, required this.child, this.padding});

  final Widget child;
  final EdgeInsetsGeometry? padding;

  @override
  Widget build(final BuildContext context) => Container(
        padding: padding ?? const EdgeInsets.all(16),
        width: 350,
        child: DottedBorder(
          color: Colors.white,
          borderType: BorderType.RRect,
          radius: const Radius.circular(12),
          padding: const EdgeInsets.all(8),
          strokeWidth: 1.5,
          dashPattern: const [8, 6],
          child: ClipRRect(
            borderRadius: const BorderRadius.all(Radius.circular(12)),
            child: child,
          ),
        ),
      );
}
```

```dart
import 'package:flutter/material.dart';
import 'package:flutter_svg/flutter_svg.dart';
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/common/util/svg_manager.dart";

class PoweredByFlutterWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Row(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        // Using Flutter logo from assets, make sure you have added a Flutter logo SVG in your assets fo
        SvgManager.flutter(width: 24, height: 24),
        SizedBox(width: 10),
        // Provides some spacing between the logo and the text
        Text(
          'Powered by Flutter',
          style: GoogleFonts.robotoMono(
              fontSize: 14,
              fontWeight: FontWeight.bold,
          ),
        ),
      ],
    );
  }
}
```

```dart
import "dart:async";

import "package:flutter/material.dart";

import "package:menyusha/app/common/hover_button.dart";

class ExperienceWidget extends StatefulWidget {
  const ExperienceWidget(
      {required this.child1, required this.child2, super.key});

  final Widget child1;
  final Widget child2;

  @override
  _ExperienceWidgetState createState() => _ExperienceWidgetState();
}

class _ExperienceWidgetState extends State<ExperienceWidget>
    with SingleTickerProviderStateMixin {




  @override
  Widget build(final BuildContext context) => HoverButton(
        color: Colors.transparent,
        onTap: () {},
        onDoubleTap: () {},
        onHover: (final isHovered) {

        },
        child: Column(
          children: <Widget>[
            widget.child1,
            widget.child2
          ],
        ),
      );
}
```

```dart
import "package:flutter/cupertino.dart";

import "package:menyusha/app/common/ui/border_paint.dart";

class AnimatedPointBorderContainer extends StatefulWidget {

  const AnimatedPointBorderContainer({
    required this.child, super.key,
    this.borderWidth = 3.0,
    this.borderRadius = 10.0,
    this.animationDuration = const Duration(seconds: 10),
  });
  final Widget child;
  final double borderWidth;
  final double borderRadius;
  final Duration animationDuration;

  @override
  _AnimatedPointBorderContainerState createState() =>
      _AnimatedPointBorderContainerState();
}

class _AnimatedPointBorderContainerState
    extends State<AnimatedPointBorderContainer>
    with SingleTickerProviderStateMixin {
  late AnimationController _controller;

  @override
  void initState() {
    super.initState();
    _controller = AnimationController(
      vsync: this,
      duration: widget.animationDuration,
    )..repeat();
  }

  @override
  void dispose() {
    _controller.dispose();
    super.dispose();
  }

  @override
  Widget build(final BuildContext context) => AnimatedBuilder(
      animation: _controller,
      builder: (final context, final child) => CustomPaint(
          painter: BorderPainter(
              _controller.value, widget.borderWidth, widget.borderRadius,),
          child: Container(
            padding: EdgeInsets.all(widget.borderWidth + widget.borderRadius),
            child: widget.child,
          ),
        ),
    );
}
```

```dart
import "dart:async";

import "package:flutter/material.dart";

import "package:menyusha/app/common/hover_button.dart";

class ExpandableWidget extends StatefulWidget {
  const ExpandableWidget(
      {required this.child1, required this.child2, super.key});

  final Widget child1;
  final Widget child2;

  @override
  _ExpandableWidgetState createState() => _ExpandableWidgetState();
}

class _ExpandableWidgetState extends State<ExpandableWidget>
    with SingleTickerProviderStateMixin {
  late AnimationController _controller;
  late Animation<double> _sizeAnimation;
  bool _isExpanded = false;
  bool _isHovered = false;

  @override
  void initState() {
    super.initState();

    final durationMillis = _isHovered ? 600 : 300;
    _controller = AnimationController(
      vsync: this,
      duration: Duration(milliseconds: durationMillis),
    );

    final endValue = _isHovered ? 0.2 : 1.0;
    _sizeAnimation =
        Tween<double>(begin: 0, end: endValue).animate(_controller);
  }

  @override
  void dispose() {
    _controller.dispose();
    super.dispose();
  }

  void _toggleExpand() {
    setState(() {
      _isExpanded = !_isExpanded;
      if (_isExpanded) {
        _controller.forward();
      } else {
        _controller.reverse();
      }
    });
  }

  Timer? _debounce;

  void toggleHover(final bool isHovered) {
    if (_debounce?.isActive ?? false) _debounce!.cancel();
    _debounce = Timer(Duration(milliseconds: _isHovered ? 500 : 1000), () {
      _toggleHover(isHovered);
```

```
    });
  }

  void _toggleHover(final bool isHovered) {
    setState(() {
      _isHovered = isHovered;

      final durationMillis = _isHovered ? 300 : 300;
      _controller.duration = Duration(milliseconds: durationMillis);

      if (_isHovered) {
        _controller.forward();
        return;
      } else {
        if (_isExpanded) {
          return;
        } else {
          _controller.reverse();
          return;
        }
      }
    });
  }

  @override
  Widget build(final BuildContext context) => HoverButton(
        color: _isHovered ? Colors.grey : Colors.transparent,
        onTap: _toggleExpand,
        onDoubleTap: () {},
        onHover: (final isHovered) {
          toggleHover(isHovered);
        },
        child: Column(
          children: <Widget>[
            widget.child1,
            SizeTransition(
              sizeFactor: _sizeAnimation,
              child: widget.child2,
            ),
          ],
        ),
      );
}
```

```
class BorderPainter extends CustomPainter {
  final double progress;
  final double borderWidth;
  final double borderRadius;

  BorderPainter(this.progress, this.borderWidth, this.borderRadius);

  @override
  void paint(Canvas canvas, Size size) {
    final paint = Paint()
      ..color = Colors.grey
      ..strokeWidth = borderWidth
      ..style = PaintingStyle.stroke;

    final rect = Rect.fromLTWH(borderWidth / 2, borderWidth / 2,
        size.width - borderWidth, size.height - borderWidth);
    final rrect = RRect.fromRectAndRadius(rect, Radius.circular(borderRadius));
    canvas.drawRRect(rrect, paint);

    final pointPaint = Paint()
      ..color = Colors.lightBlue
      ..strokeWidth = borderWidth
      ..style = PaintingStyle.fill;

    Offset point = _calculatePoint(progress, size, borderRadius, borderWidth);
    canvas.drawCircle(point, borderWidth / 2, pointPaint);
  }

  Offset _calculatePoint(
      double progress, Size size, double borderRadius, double borderWidth) {
    final adjustedWidth = size.width - borderWidth;
    final adjustedHeight = size.height - borderWidth;
    final totalLength =
        2 * (adjustedWidth + adjustedHeight - 2 * borderRadius) +
            2 * pi * borderRadius;
    final position = totalLength * progress;

    // Top edge
    if (position < adjustedWidth - borderRadius) {
      return Offset(borderWidth / 2 + position, borderWidth / 2);
    }
    // Right upper corner
    else if (position < adjustedWidth + pi / 2 * borderRadius) {
      return Offset(
              adjustedWidth - borderRadius + borderWidth / 2, borderWidth / 2) +
          Offset(
              cos(pi / 2 -
                      (position - (adjustedWidth - borderRadius)) /
                          borderRadius) *
                  borderRadius,
              sin(pi / 2 -
                      (position - (adjustedWidth - borderRadius)) /
                          borderRadius) *
                  borderRadius);
    }
    // Right edge
    else if (position <
        adjustedWidth +
            pi / 2 * borderRadius +
            adjustedHeight -
            2 * borderRadius) {
      return Offset(
```

```
      size.width - borderWidth / 2,
      borderWidth / 2 +
          borderRadius +
          position -
          adjustedWidth -
          pi / 2 * borderRadius);
}
// Bottom right corner
else if (position <
    adjustedWidth + adjustedHeight - borderRadius + pi * borderRadius) {
  return Offset(adjustedWidth - borderRadius + borderWidth / 2,
          adjustedHeight - borderRadius + borderWidth / 2) +
      Offset(
          cos(position -
                  adjustedWidth -
                  adjustedHeight +
                  borderRadius -
                  pi / 2 * borderRadius) /
              borderRadius *
              borderRadius,
          sin(position -
                  adjustedWidth -
                  adjustedHeight +
                  borderRadius -
                  pi / 2 * borderRadius) /
              borderRadius *
              borderRadius);
}
// Bottom edge
else if (position <
    2 * adjustedWidth +
        adjustedHeight -
        2 * borderRadius +
        pi * borderRadius) {
  return Offset(
      adjustedWidth -
          (position - adjustedWidth - adjustedHeight + pi * borderRadius) +
          borderWidth / 2,
      size.height - borderWidth / 2);
}
// Bottom left corner
else if (position <
    2 * adjustedWidth +
        2 * adjustedHeight -
        2 * borderRadius +
        1.5 * pi * borderRadius) {
  return Offset(borderWidth / 2 + borderRadius,
          adjustedHeight - borderRadius + borderWidth / 2) +
      Offset(
          cos(1.5 * pi -
                  (position -
                      2 * adjustedWidth -
                      adjustedHeight +
                      1 * borderRadius)) /
              borderRadius *
              borderRadius,
          sin(1.5 * pi -
                  (position -
                      2 * adjustedWidth -
                      adjustedHeight +
                      1 * borderRadius)) /
              borderRadius *
```

```
                borderRadius);
    }
    // Left edge
    else if (position <
        2 * (adjustedWidth + adjustedHeight - 2 * borderRadius) +
            1.5 * pi * borderRadius) {
      return Offset(
          borderWidth / 2,
          adjustedHeight -
              (position -
                  2 * adjustedWidth -
                  2 * adjustedHeight +
                  1.5 * pi * borderRadius) +
              borderWidth / 2);
    }
    // Top left corner
    else {
      return Offset(borderWidth / 2 + borderRadius, borderWidth / 2) +
          Offset(
              cos(2 * pi -
                      (position -
                          2 *
                              (adjustedWidth +
                                  adjustedHeight -
                                  2 * borderRadius) -
                          1.5 * pi * borderRadius)) /
                  borderRadius *
                  borderRadius,
              sin(2 * pi -
                      (position -
                          2 *
                              (adjustedWidth +
                                  adjustedHeight -
                                  2 * borderRadius) -
                          1.5 * pi * borderRadius)) /
                  borderRadius *
                  borderRadius);
    }
  }

  @override
  bool shouldRepaint(covariant CustomPainter oldDelegate) {
    return true; // Always repaint for a continuous animation
  }
}
```

```
import "dart:math";

import "package:flutter/material.dart";

class BorderPainter extends CustomPainter {

  BorderPainter(this.progress, this.borderWidth, this.borderRadius);
  final double progress;
  final double borderWidth;
  final double borderRadius;

  @override
  void paint(final Canvas canvas, final Size size) {
    final paint = Paint()
      ..color = Colors.grey
      ..strokeWidth = borderWidth
      ..style = PaintingStyle.stroke;

    final rect = Rect.fromLTWH(borderWidth / 2, borderWidth / 2,
        size.width - borderWidth, size.height - borderWidth,);
    final rrect = RRect.fromRectAndRadius(rect, Radius.circular(borderRadius));
    canvas.drawRRect(rrect, paint);

    final pointPaint = Paint()
      ..color = Colors.lightBlue
      ..strokeWidth = borderWidth
      ..style = PaintingStyle.fill;

    final point = _calculatePoint(progress, size, borderRadius, borderWidth);
    canvas.drawCircle(point, borderWidth / 2, pointPaint);
  }

  Offset _calculatePoint(
      final double progress, final Size size, final double borderRadius, final double borderWidth,) {
    final adjustedWidth = size.width - borderWidth;
    final adjustedHeight = size.height - borderWidth;
    final totalLength =
        2 * (adjustedWidth + adjustedHeight - 2 * borderRadius) +
            2 * pi * borderRadius;
    final position = totalLength * progress;

    // Top edge
    if (position < adjustedWidth - borderRadius) {
      return Offset(borderWidth / 2 + position, borderWidth / 2);
    }
    // Right upper corner
    else if (position < adjustedWidth + pi / 2 * borderRadius) {
      return Offset(
          adjustedWidth - borderRadius + borderWidth / 2, borderWidth / 2,) +
          Offset(
              cos(pi / 2 -
                  (position - (adjustedWidth - borderRadius)) /
                      borderRadius,) *
                  borderRadius,
              sin(pi / 2 -
                  (position - (adjustedWidth - borderRadius)) /
                      borderRadius,) *
                  borderRadius,);
    }
    // Right edge
    else if (position <
        adjustedWidth +
```

```
        pi / 2 * borderRadius +
        adjustedHeight -
        2 * borderRadius) {
  return Offset(
      size.width - borderWidth / 2,
      borderWidth / 2 +
          borderRadius +
          position -
          adjustedWidth -
          pi / 2 * borderRadius,);
}
// Bottom right corner
else if (position <
    adjustedWidth + adjustedHeight - borderRadius + pi * borderRadius) {
  return Offset(adjustedWidth - borderRadius + borderWidth / 2,
      adjustedHeight - borderRadius + borderWidth / 2,) +
      Offset(
          cos(position -
              adjustedWidth -
              adjustedHeight +
              borderRadius -
              pi / 2 * borderRadius,) /
              borderRadius *
              borderRadius,
          sin(position -
              adjustedWidth -
              adjustedHeight +
              borderRadius -
              pi / 2 * borderRadius,) /
              borderRadius *
              borderRadius,);
}
// Bottom edge
else if (position <
    2 * adjustedWidth +
        adjustedHeight -
        2 * borderRadius +
        pi * borderRadius) {
  return Offset(
      adjustedWidth -
          (position - adjustedWidth - adjustedHeight + pi * borderRadius) +
          borderWidth / 2,
      size.height - borderWidth / 2,);
}
// Bottom left corner
else if (position <
    2 * adjustedWidth +
        2 * adjustedHeight -
        2 * borderRadius +
        1.5 * pi * borderRadius) {
  return Offset(borderWidth / 2 + borderRadius,
      adjustedHeight - borderRadius + borderWidth / 2,) +
      Offset(
          cos(1.5 * pi -
              (position -
                  2 * adjustedWidth -
                  adjustedHeight +
                  1 * borderRadius),) /
              borderRadius *
              borderRadius,
          sin(1.5 * pi -
              (position -
```

```
                    2 * adjustedWidth -
                    adjustedHeight +
                    1 * borderRadius),) /
                borderRadius *
                borderRadius,);
    }
    // Left edge
    else if (position <
        2 * (adjustedWidth + adjustedHeight - 2 * borderRadius) +
            1.5 * pi * borderRadius) {
      return Offset(
          borderWidth / 2,
          adjustedHeight -
              (position -
                  2 * adjustedWidth -
                  2 * adjustedHeight +
                  1.5 * pi * borderRadius) +
              borderWidth / 2,);
    }
    // Top left corner
    else {
      return Offset(borderWidth / 2 + borderRadius, borderWidth / 2) +
          Offset(
              cos(2 * pi -
                  (position -
                      2 *
                          (adjustedWidth +
                              adjustedHeight -
                              2 * borderRadius) -
                      1.5 * pi * borderRadius),) /
                  borderRadius *
                  borderRadius,
              sin(2 * pi -
                  (position -
                      2 *
                          (adjustedWidth +
                              adjustedHeight -
                              2 * borderRadius) -
                      1.5 * pi * borderRadius),) /
                  borderRadius *
                  borderRadius,);
    }
  }

  @override
  bool shouldRepaint(covariant final CustomPainter oldDelegate) {
    return true; // Always repaint for a continuous animation
  }
}
```

```dart
import "package:flutter/material.dart";

class InfoBlock extends StatelessWidget {
  // Constructor to initialize the child widget
  const InfoBlock({required this.child, super.key,
    this.color, this.width, this.padding, this.radius = 12,});
  final Widget child;

  final Color? color;

  final EdgeInsetsGeometry? padding;

  final double? radius;
  final double? width;

  @override
  Widget build(final BuildContext context) => Container(
      width: width,
      decoration: BoxDecoration(
        color: color ?? Colors.grey.shade900,
        borderRadius: BorderRadius.circular(radius ?? 12),
      ),
      padding: padding ?? const EdgeInsets.all(8),
      child: child,
    );
}
```

```
import "package:cached_network_image/cached_network_image.dart";
import "package:carousel_slider/carousel_slider.dart";
import "package:flutter/material.dart";

class ImageSlider extends StatefulWidget {

  const ImageSlider({required this.imgList, super.key});
  final List<String> imgList;

  @override
  State<StatefulWidget> createState() => _ImageSliderState(imgList: imgList);
}

class _ImageSliderState extends State<ImageSlider> {

  _ImageSliderState({required this.imgList});
  final List<String> imgList;
  int _current = 0;

  @override
  Widget build(final BuildContext context) => Container(
      color: Colors.transparent,
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        mainAxisSize: MainAxisSize.min,
        children: [
          Expanded(
            child: CarouselSlider(
              options: CarouselOptions(
                  enlargeCenterPage: true,
                  viewportFraction: 1,
                  aspectRatio: 1,
                  onPageChanged: (final index, final rs) {
                    setState(() {
                      _current = index;
                    });
                  },),
              items: imgList
                  .map((final item) => InteractiveViewer(

                    panEnabled: false, // Set it to false
                    boundaryMargin: const EdgeInsets.all(100),
                    minScale: 0.5,
                    maxScale: 2,
                    child: CachedNetworkImage(
                          imageUrl: item,
                          fit: BoxFit.contain,
                          width: double.infinity,
                          progressIndicatorBuilder: (final context, final url, final progress) =>
                              Center(
                            child: CircularProgressIndicator(
                              color: Colors.black,
                              value: progress.progress,
                            ),
                          ),
                        ),
                  ),)
                  .toList(),
            ),
          ),
        ],
      ),
```

```
    );
}

class CarouselPage extends StatelessWidget {

  const CarouselPage({required this.imgList, super.key});
  final List<String> imgList;

  @override
  Widget build(final BuildContext context) => Scaffold(
      appBar: AppBar(
        title: const Text("Image Carousel"),
      ),
      body: const Center(
          child: ImageSlider(
        imgList: [],
      ),),
    );
}
```

```
const Unit = _unitFunction;

void _unitFunction() {}
```

```dart
import "package:flutter/widgets.dart";
import "package:flutter_svg/flutter_svg.dart";

class SvgManager {
  static const String _basePath = "assets/svg/";

  static SvgPicture logo({double? width, double? height}) {
    return _getIcon('mainlogo', width: width, height: height);
  }

  static SvgPicture bandana({double? width, double? height, Color? color}) {
    return _getIcon('bnd_white', width: width, height: height, color:color);
  }

  static SvgPicture flutter({double? width, double? height}) {
    return _getIcon('flutter', width: width, height: height);
  }

  // Private method to return an SVG icon by its name
  static SvgPicture _getIcon(String name,
      {double? width, double? height, Color? color}) {
    return SvgPicture.asset('$_basePath$name.svg',
        width: width, height: height, color: color);
  }
}
```

```dart
import "package:flutter/material.dart";

class GapRoundedBlock extends StatelessWidget {

  // Constructor to initialize the child widget
  const GapRoundedBlock(
      {required this.child, super.key, this.backGroundColor,this.strokeColor, this.width, this.padding,}
  final Widget child;

  final Color? backGroundColor;
  final Color? strokeColor;

  final EdgeInsetsGeometry? padding;

  final double? width;

  @override
  Widget build(final BuildContext context) => Container(
      width: width,
      padding: padding ?? const EdgeInsets.all(8),
      decoration: BoxDecoration(
          color: backGroundColor,
          borderRadius: BorderRadius.circular(12),
          border: Border.all(
            color: strokeColor ?? Colors.white24,
          ),),
      child: child,
    );
}
```

```dart
import "dart:math" as math;

import "package:flutter/material.dart";
import "package:google_fonts/google_fonts.dart";

// Add the WordPositionCalculator class here (from previous example)

class TextCloud extends StatefulWidget {
  const TextCloud({super.key});

  @override
  _TextCloudState createState() => _TextCloudState();
}

class _TextCloudState extends State<TextCloud> {
  final List<String> words = [
    "Android",
    "Flutter",
    "Firebase",
    "Kotlin",
    "Java",
    "Ktor",
    "REST API",
    "Jetpack Compose",
    "Dart",
    "FFmpeg",
    "ChatGPT",
  ];

  @override
  Widget build(final BuildContext context) => LayoutBuilder(
      builder: (final context, final constraints) => FutureBuilder<List<Widget>>(
          future: _calculatePositions(words, constraints),
          builder: (final context, final snapshot) {
            if (snapshot.connectionState == ConnectionState.done) {
              return Stack(children: snapshot.data ?? []);
            } else {
              // Show a loading spinner or a placeholder while waiting
              return const Center(child: CircularProgressIndicator());
            }
          },
        ),
    );

  Future<List<Widget>> _calculatePositions(
      final List<String> words, final BoxConstraints constraints,) async {
    final random = math.Random();
    final placedRects = <Rect>[];
    final positionedWords = <Widget>[];

    for (final word in words) {
      final fontSize = random.nextInt(15) + 15.0;
      final color = Colors.primaries[random.nextInt(Colors.primaries.length)];

      final textStyle = GoogleFonts.robotoMono(
        fontSize: fontSize,
        fontWeight: FontWeight.bold,
        color: color,
      );
      final textPainter = TextPainter(
        text: TextSpan(text: word, style: textStyle),
        maxLines: 1,
```

```
        textDirection: TextDirection.ltr,
      )..layout();

      final textSize = textPainter.size;
      Offset position;
      bool doesOverlap;
      do {
        position = Offset(
          random.nextDouble() * (constraints.maxWidth - textSize.width),
          random.nextDouble() * (constraints.maxHeight - textSize.height),
        );

        final textRect = position & textSize;
        doesOverlap = placedRects.any((final rect) => rect.overlaps(textRect));

        if (!doesOverlap) {
          placedRects.add(textRect);
        }
      } while (doesOverlap);

      positionedWords.add(
        Positioned(
          left: position.dx,
          top: position.dy,
          child: Container(
              padding: const EdgeInsets.all(4),
              decoration: BoxDecoration(
                color: textStyle.color?.withAlpha(95),
                borderRadius: BorderRadius.circular(50),
              ),
              child: Text(word, style: textStyle),),
        ),
      );
    }

    return positionedWords;
  }
}

class WordPosition {

  WordPosition(this.word, this.position);
  final String word;
  final Offset position;
}
```

```
class DateUtil {

  DateUtil(this.date);
  DateTime date;
  DateTime now = DateTime.now();

  List<String> weekDays = [
    "Sunday",
    "Monday",
    "Tuesday",
    "Wednesday",
    "Thursday",
    "Friday",
    "Saturday",
  ];

  String getTime([final bool showWeek = true]) {
    if (showWeek && date.isBefore(DateTime(now.year, now.month, now.day))) {
      return getWeek();
    }

    return "${date.hour < 10 ? '0' : ''}${date.hour}:${date.minute < 10 ? '0' : ''}${date.minute}";
  }

  String getWeek() => weekDays[date.weekday].substring(0, 3);
}
```

```dart
import "package:flutter/material.dart";

class HoverButton extends StatefulWidget {
  const HoverButton({
    required this.child,
    required this.onTap,
    required this.onDoubleTap,
    super.key,
    this.hoverColor = Colors.grey,
    this.color = Colors.black,
    this.radius = 12,
    this.onHover,
    this.clickable = true,
  });

  final Widget child;

  final Color hoverColor;
  final Color color;
  final double radius;
  final VoidCallback onTap;
  final Function? onHover;
  final VoidCallback onDoubleTap;

  final bool clickable;

  @override
  _HoverButtonState createState() => _HoverButtonState();
}

class _HoverButtonState extends State<HoverButton> {
  bool _isHovering = false;

  @override
  Widget build(final BuildContext context) => AnimatedContainer(
        duration: const Duration(milliseconds: 200),
        decoration: BoxDecoration(
          color: _isHovering ? widget.hoverColor : widget.color,
          borderRadius: BorderRadius.circular(widget.radius),
        ),
        child: InkWell(
          onHover: (final value) {
            if (widget.onHover != null) {
              widget.onHover!(value);
            }

            setState(() {
              _isHovering = value;
            });
          },
          onDoubleTap: widget.onDoubleTap,
          onTap: () {
            if (widget.clickable) {
              widget.onTap();
            }
          },
          child: widget.child,
        ),
      );
}
```

```
import "dart:html";

import "package:flutter/services.dart";

void openLink(final String url) {
  window.open(url, "_blank");
}


Future<void> downloadPdfFromAssets(final String assetPath) async {
  try {
    // Load the PDF file from assets
    final data = await rootBundle.load(assetPath);
    final bytes = data.buffer.asUint8List();
    final blob = Blob([bytes], "application/pdf");
    final url = Url.createObjectUrlFromBlob(blob);
    final anchor = AnchorElement(href: url)
      ..setAttribute("download", "lenyk_resume.pdf")
      ..click();
    // Clean up the object URL
    Url.revokeObjectUrl(url);
  } catch (e) {
    print("Error downloading PDF: $e");
  }
}
```

```dart
import "package:flutter/material.dart";

class ResponsiveUtil {
  static bool isMobile(final BuildContext context) =>
      MediaQuery.of(context).size.width < 600;

  static bool isTablet(final BuildContext context) {
    final width = MediaQuery.of(context).size.width;
    return width >= 600 && width < 1024;
  }

  static bool alwaysBehaveLikeMobile = false;

  static bool isDesktop(final BuildContext context) =>
      MediaQuery.of(context).size.width >= 1024;

  static bool isDesktopFull(final BuildContext context) {
    var alwaysReturnMobile = alwaysBehaveLikeMobile;
    if(alwaysReturnMobile){
      return false;
    }
    final desktopWidth = MediaQuery.of(context).size.width >= 1024;
    final desktopHeight = MediaQuery.of(context).size.height >= 720;
    return desktopWidth && desktopHeight;
  }

  static bool isBigEnough(final BuildContext context, final double value) =>
      MediaQuery.of(context).size.width >= value;

  // Additional utility functions can be added here
  // Example: Getting the screen orientation
  static bool isPortrait(final BuildContext context) =>
      MediaQuery.of(context).orientation == Orientation.portrait;

  static bool isLandscape(final BuildContext context) =>
      MediaQuery.of(context).orientation == Orientation.landscape;

  // Example: Getting the device pixel ratio
  static double devicePixelRatio(final BuildContext context) =>
      MediaQuery.of(context).devicePixelRatio;

// Add more functions as per your requirement
}
```

```
import "package:flutter/material.dart";

class BackBtn extends StatelessWidget {
  const BackBtn({
    super.key,
  });

  @override
  Widget build(final BuildContext context) => IconButton(
      onPressed: () {
        Navigator.maybePop(context);
      },
      icon: const Icon(
        Icons.arrow_back_ios,
      ),
    );
}
```

```dart
import "package:cloud_firestore/cloud_firestore.dart";
import "package:freezed_annotation/freezed_annotation.dart";

class TimestampSerializer implements JsonConverter<DateTime, dynamic> {
  const TimestampSerializer();

  @override
  DateTime fromJson(final timestamp) => timestamp.toDate();

  @override
  Timestamp toJson(final DateTime date) => Timestamp.fromDate(date);
}

class TimestampSerializerNullable implements JsonConverter<DateTime?, dynamic> {
  const TimestampSerializerNullable();

  @override
  DateTime? fromJson(final timestamp) {
    // Handle the case where timestamp is null
    if (timestamp == null) return null;
    return timestamp is Timestamp ? timestamp.toDate() : null;
  }

  @override
  dynamic toJson(final DateTime? date) {
    // Handle the case where date is null
    if (date == null) return null;
    return Timestamp.fromDate(date);
  }
}

class TimestampSerializer2 implements JsonConverter<Timestamp, int> {
  const TimestampSerializer2();

  @override
  int toJson(final Timestamp timestamp) => timestamp.millisecondsSinceEpoch;

  @override
  Timestamp fromJson(final int millis) => Timestamp.fromMillisecondsSinceEpoch(millis);
}

class NullableTimestampSerializer implements JsonConverter<Timestamp?, int?> {
  const NullableTimestampSerializer();

  @override
  int? toJson(final Timestamp? timestamp) =>
      timestamp?.millisecondsSinceEpoch;

  @override
  Timestamp? fromJson(final int? millis) =>
      millis != null ? Timestamp.fromMillisecondsSinceEpoch(millis) : null;
}
```

```
import "package:auto_route/auto_route.dart";
import "package:flutter/material.dart";
import "package:get_it/get_it.dart";

import "../data/firebase/user_manager.dart";
import "../features/main/screen/menyusha/admin/container/admin_container.dart";
import "../features/main/screen/menyusha/admin/create_menu/create_menu_screen.dart";
import "../features/main/screen/menyusha/admin/list_menu/list_menu_screen.dart";
import "../features/main/screen/menyusha/admin/view_menu/private_menu_screen.dart";
import "../features/main/screen/menyusha/hello/hello_screen.dart";
import "../features/main/screen/menyusha/login/login_screen.dart";
import "../features/main/screen/menyusha/login/registration_screen.dart";
import "../features/main/screen/menyusha/public/view_menu/public_menu_screen.dart";
import "../features/main/screen/menyusha/theme/preview_screen.dart";
import "../features/main/screen/menyusha/theme/sample_screen.dart";
import "../features/main/screen/mobileera_screen.dart";
import "../features/main/screen/notfound/not_found.dart";

part "app_router.gr.dart";

@AutoRouterConfig(replaceInRouteName: "Screen,Route")
class AppRouter extends _$AppRouter implements AutoRouteGuard {
  final getIt = GetIt.instance;
  late final UserManager userManager = getIt<UserManager>();

  @override
  List<AutoRoute> get routes =>
      [
        AutoRoute(
          page: MobileEraRoute.page,
          path: "/",
          children: [
            AutoRoute(page: HelloRoute.page, path: "", initial: true),
            //
            AutoRoute(page: RegistrationRoute.page, path: "registration"),
            AutoRoute(page: LoginRoute.page, path: "login"),
            //
            AutoRoute(page: AdminContainerRoute.page, path: "admin", children: [
              AutoRoute(page: ListMenuRoute.page, path: ""),
              AutoRoute(page: CreateMenuRoute.page, path: "create"),
              AutoRoute(page: PreviewRoute.page, path: "preview"),
              AutoRoute(page: SampleRoute.page, path: "sample"),
              AutoRoute(page: PrivateMenuRoute.page, path: ":id"),
            ]),
            //
            AutoRoute(page: PublicMenuRoute.page, path: ":id"),
            //
            AutoRoute(page: NotFoundRoute.page, path: "not-found"),
            RedirectRoute(path: "*", redirectTo: "not-found"),
          ],
        ),
      ];

  @override
  void onNavigation(final NavigationResolver resolver,
      final StackRouter router) {
    final isUserLoggedIn = userManager.isUserLoggedInSync();

    if (resolver.route.name == NotFoundRoute.name ||
        resolver.route.name == MobileEraRoute.name ||
        resolver.route.name == HelloRoute.name ||
        resolver.route.name == PublicMenuRoute.name) {
```

```
      // we continue navigation
      resolver.next();
    } else if (resolver.route.name == LoginRoute.name ||
        resolver.route.name == RegistrationRoute.name) {
      if (isUserLoggedIn) {
        push(AdminContainerRoute());
      } else {
        resolver.next();
      }
    } else if ((resolver.route.name == AdminContainerRoute.name ||
        resolver.route.name == ListMenuRoute.name ||
        resolver.route.name == CreateMenuRoute.name ||
        resolver.route.name == PrivateMenuRoute.name ||
        resolver.route.name == SampleRoute.name ||
        resolver.route.name == PreviewRoute.name) &&
        isUserLoggedIn) {
      resolver.next();
    } else {
      print(resolver.route.name);
      push(NotFoundRoute());
    }
  }
}
```

```
// GENERATED CODE - DO NOT MODIFY BY HAND

// **************************************************************************
// AutoRouterGenerator
// **************************************************************************

// ignore_for_file: type=lint
// coverage:ignore-file

part of 'app_router.dart';

abstract class _$AppRouter extends RootStackRouter {
  // ignore: unused_element
  _$AppRouter({super.navigatorKey});

  @override
  final Map<String, PageFactory> pagesMap = {
    AdminContainerRoute.name: (routeData) {
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: const AdminContainerScreen(),
      );
    },
    CreateMenuRoute.name: (routeData) {
      final args = routeData.argsAs<CreateMenuRouteArgs>(
          orElse: () => const CreateMenuRouteArgs());
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: CreateMenuScreen(key: args.key),
      );
    },
    HelloRoute.name: (routeData) {
      final args = routeData.argsAs<HelloRouteArgs>(
          orElse: () => const HelloRouteArgs());
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: HelloScreen(key: args.key),
      );
    },
    ListMenuRoute.name: (routeData) {
      final args = routeData.argsAs<ListMenuRouteArgs>(
          orElse: () => const ListMenuRouteArgs());
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: ListMenuScreen(key: args.key),
      );
    },
    LoginRoute.name: (routeData) {
      final args = routeData.argsAs<LoginRouteArgs>(
          orElse: () => const LoginRouteArgs());
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: LoginScreen(key: args.key),
      );
    },
    MobileEraRoute.name: (routeData) {
      final args = routeData.argsAs<MobileEraRouteArgs>(
          orElse: () => const MobileEraRouteArgs());
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: MobileEraScreen(key: args.key),
      );
```

```
    },
    NotFoundRoute.name: (routeData) {
      final args = routeData.argsAs<NotFoundRouteArgs>(
          orElse: () => const NotFoundRouteArgs());
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: NotFoundScreen(key: args.key),
      );
    },
    PreviewRoute.name: (routeData) {
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: const PreviewScreen(),
      );
    },
    PrivateMenuRoute.name: (routeData) {
      final pathParams = routeData.inheritedPathParams;
      final args = routeData.argsAs<PrivateMenuRouteArgs>(
          orElse: () => PrivateMenuRouteArgs(id: pathParams.getString('id')));
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: PrivateMenuScreen(
          key: args.key,
          id: args.id,
        ),
      );
    },
    PublicMenuRoute.name: (routeData) {
      final pathParams = routeData.inheritedPathParams;
      final args = routeData.argsAs<PublicMenuRouteArgs>(
          orElse: () => PublicMenuRouteArgs(id: pathParams.getString('id')));
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: PublicMenuScreen(
          key: args.key,
          id: args.id,
        ),
      );
    },
    RegistrationRoute.name: (routeData) {
      final args = routeData.argsAs<RegistrationRouteArgs>(
          orElse: () => const RegistrationRouteArgs());
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: RegistrationScreen(key: args.key),
      );
    },
    SampleRoute.name: (routeData) {
      return AutoRoutePage<dynamic>(
        routeData: routeData,
        child: const SampleScreen(),
      );
    },
  };
}

/// generated route for
/// [AdminContainerScreen]
class AdminContainerRoute extends PageRouteInfo<void> {
  const AdminContainerRoute({List<PageRouteInfo>? children})
      : super(
          AdminContainerRoute.name,
```

```
          initialChildren: children,
        );

  static const String name = 'AdminContainerRoute';

  static const PageInfo<void> page = PageInfo<void>(name);
}

/// generated route for
/// [CreateMenuScreen]
class CreateMenuRoute extends PageRouteInfo<CreateMenuRouteArgs> {
  CreateMenuRoute({
    Key? key,
    List<PageRouteInfo>? children,
  }) : super(
          CreateMenuRoute.name,
          args: CreateMenuRouteArgs(key: key),
          initialChildren: children,
        );

  static const String name = 'CreateMenuRoute';

  static const PageInfo<CreateMenuRouteArgs> page =
      PageInfo<CreateMenuRouteArgs>(name);
}

class CreateMenuRouteArgs {
  const CreateMenuRouteArgs({this.key});

  final Key? key;

  @override
  String toString() {
    return 'CreateMenuRouteArgs{key: $key}';
  }
}

/// generated route for
/// [HelloScreen]
class HelloRoute extends PageRouteInfo<HelloRouteArgs> {
  HelloRoute({
    Key? key,
    List<PageRouteInfo>? children,
  }) : super(
          HelloRoute.name,
          args: HelloRouteArgs(key: key),
          initialChildren: children,
        );

  static const String name = 'HelloRoute';

  static const PageInfo<HelloRouteArgs> page = PageInfo<HelloRouteArgs>(name);
}

class HelloRouteArgs {
  const HelloRouteArgs({this.key});

  final Key? key;

  @override
  String toString() {
    return 'HelloRouteArgs{key: $key}';
```

```
  }
}

/// generated route for
/// [ListMenuScreen]
class ListMenuRoute extends PageRouteInfo<ListMenuRouteArgs> {
  ListMenuRoute({
    Key? key,
    List<PageRouteInfo>? children,
  }) : super(
          ListMenuRoute.name,
          args: ListMenuRouteArgs(key: key),
          initialChildren: children,
        );

  static const String name = 'ListMenuRoute';

  static const PageInfo<ListMenuRouteArgs> page =
      PageInfo<ListMenuRouteArgs>(name);
}

class ListMenuRouteArgs {
  const ListMenuRouteArgs({this.key});

  final Key? key;

  @override
  String toString() {
    return 'ListMenuRouteArgs{key: $key}';
  }
}

/// generated route for
/// [LoginScreen]
class LoginRoute extends PageRouteInfo<LoginRouteArgs> {
  LoginRoute({
    Key? key,
    List<PageRouteInfo>? children,
  }) : super(
          LoginRoute.name,
          args: LoginRouteArgs(key: key),
          initialChildren: children,
        );

  static const String name = 'LoginRoute';

  static const PageInfo<LoginRouteArgs> page = PageInfo<LoginRouteArgs>(name);
}

class LoginRouteArgs {
  const LoginRouteArgs({this.key});

  final Key? key;

  @override
  String toString() {
    return 'LoginRouteArgs{key: $key}';
  }
}

/// generated route for
/// [MobileEraScreen]
```

```
class MobileEraRoute extends PageRouteInfo<MobileEraRouteArgs> {
  MobileEraRoute({
    Key? key,
    List<PageRouteInfo>? children,
  }) : super(
          MobileEraRoute.name,
          args: MobileEraRouteArgs(key: key),
          initialChildren: children,
        );

  static const String name = 'MobileEraRoute';

  static const PageInfo<MobileEraRouteArgs> page =
      PageInfo<MobileEraRouteArgs>(name);
}

class MobileEraRouteArgs {
  const MobileEraRouteArgs({this.key});

  final Key? key;

  @override
  String toString() {
    return 'MobileEraRouteArgs{key: $key}';
  }
}

/// generated route for
/// [NotFoundScreen]
class NotFoundRoute extends PageRouteInfo<NotFoundRouteArgs> {
  NotFoundRoute({
    Key? key,
    List<PageRouteInfo>? children,
  }) : super(
          NotFoundRoute.name,
          args: NotFoundRouteArgs(key: key),
          initialChildren: children,
        );

  static const String name = 'NotFoundRoute';

  static const PageInfo<NotFoundRouteArgs> page =
      PageInfo<NotFoundRouteArgs>(name);
}

class NotFoundRouteArgs {
  const NotFoundRouteArgs({this.key});

  final Key? key;

  @override
  String toString() {
    return 'NotFoundRouteArgs{key: $key}';
  }
}

/// generated route for
/// [PreviewScreen]
class PreviewRoute extends PageRouteInfo<void> {
  const PreviewRoute({List<PageRouteInfo>? children})
      : super(
          PreviewRoute.name,
```

```
          initialChildren: children,
        );

  static const String name = 'PreviewRoute';

  static const PageInfo<void> page = PageInfo<void>(name);
}

/// generated route for
/// [PrivateMenuScreen]
class PrivateMenuRoute extends PageRouteInfo<PrivateMenuRouteArgs> {
  PrivateMenuRoute({
    Key? key,
    required String id,
    List<PageRouteInfo>? children,
  }) : super(
          PrivateMenuRoute.name,
          args: PrivateMenuRouteArgs(
            key: key,
            id: id,
          ),
          rawPathParams: {'id': id},
          initialChildren: children,
        );

  static const String name = 'PrivateMenuRoute';

  static const PageInfo<PrivateMenuRouteArgs> page =
      PageInfo<PrivateMenuRouteArgs>(name);
}

class PrivateMenuRouteArgs {
  const PrivateMenuRouteArgs({
    this.key,
    required this.id,
  });

  final Key? key;

  final String id;

  @override
  String toString() {
    return 'PrivateMenuRouteArgs{key: $key, id: $id}';
  }
}

/// generated route for
/// [PublicMenuScreen]
class PublicMenuRoute extends PageRouteInfo<PublicMenuRouteArgs> {
  PublicMenuRoute({
    Key? key,
    required String id,
    List<PageRouteInfo>? children,
  }) : super(
          PublicMenuRoute.name,
          args: PublicMenuRouteArgs(
            key: key,
            id: id,
          ),
          rawPathParams: {'id': id},
          initialChildren: children,
```

```
        );

  static const String name = 'PublicMenuRoute';

  static const PageInfo<PublicMenuRouteArgs> page =
      PageInfo<PublicMenuRouteArgs>(name);
}

class PublicMenuRouteArgs {
  const PublicMenuRouteArgs({
    this.key,
    required this.id,
  });

  final Key? key;

  final String id;

  @override
  String toString() {
    return 'PublicMenuRouteArgs{key: $key, id: $id}';
  }
}

/// generated route for
/// [RegistrationScreen]
class RegistrationRoute extends PageRouteInfo<RegistrationRouteArgs> {
  RegistrationRoute({
    Key? key,
    List<PageRouteInfo>? children,
  }) : super(
          RegistrationRoute.name,
          args: RegistrationRouteArgs(key: key),
          initialChildren: children,
        );

  static const String name = 'RegistrationRoute';

  static const PageInfo<RegistrationRouteArgs> page =
      PageInfo<RegistrationRouteArgs>(name);
}

class RegistrationRouteArgs {
  const RegistrationRouteArgs({this.key});

  final Key? key;

  @override
  String toString() {
    return 'RegistrationRouteArgs{key: $key}';
  }
}

/// generated route for
/// [SampleScreen]
class SampleRoute extends PageRouteInfo<void> {
  const SampleRoute({List<PageRouteInfo>? children})
      : super(
          SampleRoute.name,
          initialChildren: children,
        );
```

```
  static const String name = 'SampleRoute';

  static const PageInfo<void> page = PageInfo<void>(name);
}
```

```dart
import "package:flutter/material.dart";
import "package:get_it/get_it.dart";
import "package:google_fonts/google_fonts.dart";
import "package:menyusha/app/data/firebase/services/firestore_service.dart";
import "package:menyusha/app/data/firebase/user/user_payload_repository.dart";
import "package:menyusha/app/features/main/screen/menyusha/a4_page_container.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/admin_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/create_menu/create_menu_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/list_menu/list_menu_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/admin/view_menu/private_menu_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/login/auth_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/public/view_menu/public_menu_cubit.dart";
import "package:menyusha/app/features/main/screen/menyusha/theme/app_style.dart";
import "package:menyusha/app/root/app_router.dart";
import "package:menyusha/main.dart";

import "../data/firebase/user_manager.dart";

class RootComponent extends StatelessWidget {

  RootComponent({super.key}) {
    getIt
      ..registerSingleton<AppRouter>(AppRouter())
      ..registerSingleton<UserManager>(UserManager())
      ..registerSingleton<FirestoreService>(FirestoreService())
      ..registerSingleton<AuthenticationCubit>(
          AuthenticationCubit(firebaseAuth))
      ..registerSingleton<ListMenuCubit>(ListMenuCubit())
      ..registerSingleton<AdminCubit>(AdminCubit())
      ..registerSingleton<CreateMenuCubit>(CreateMenuCubit())
      ..registerFactoryParam<PublicMenuCubit, String, void>(
        (id, _) => PublicMenuCubit(id),
      )
      ..registerFactoryParam<PrivateMenuCubit, String, void>(
        (id, _) => PrivateMenuCubit(id),
      )
      ..registerSingleton<UserPayloadRepository>(
        UserPayloadRepository(),
      );
  }
  final getIt = GetIt.instance;

  late final _router = getIt.get<AppRouter>();

  ThemeData _buildTheme(final brightness) {
    final baseTheme = ThemeData(brightness: brightness);

    return baseTheme.copyWith(
      textTheme: GoogleFonts.robotoTextTheme(baseTheme.textTheme),
    );
  }

  @override
  Widget build(final BuildContext context) => MaterialApp.router(
        title: "Nazar Lenyk",
        theme: _buildTheme(Brightness.light),
        routerConfig: _router.config(placeholder: (context) => AppDesign.buildLogoLoader()),
        debugShowCheckedModeBanner: !buildConfig.isProduction,
      );
}
```

```dart
import 'package:cloud_firestore/cloud_firestore.dart';
import "package:freezed_annotation/freezed_annotation.dart";

mixin BaseModel {
  String? get id;

  Map<String, dynamic> toJson();
  BaseModel copyWithId({final String? id});
}
```

```dart
import 'package:cloud_firestore/cloud_firestore.dart';

import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:get_it/get_it.dart';

import "../services/firestore_service.dart";
import "base_model.dart";

class FirestoreRepository<T extends BaseModel> {
  FirestoreRepository(final String collectionPath,
      final T Function(Map<String, dynamic>) fromJson) {
    collection = service.getCollection<T>(collectionPath, fromJson);
  }

  final GetIt getIt = GetIt.instance;

  late final FirestoreService service = getIt.get<FirestoreService>();
  late final CollectionReference<T> collection;

  Future<T?> createItem(final T item) async {
    try {
      final docRef = await collection.add(item);
      final tempItem = await getItem(docRef.id);
      await updateItem(tempItem!);
      return tempItem;
    } catch (e) {
      print("Error creating item: $e");
      return null;
    }
  }

  // Read an item by ID
  Future<T?> getItem(final String id) async {
    try {
      final snapshot = await collection.doc(id).get();

      if (snapshot.exists) {
        final snap = snapshot.data()!;
        return snap.copyWithId(id: snapshot.id) as T;
      }
    } catch (e) {
      print("Error getting item: $e");
      return null;
    }
    return null;
  }

  // Read all items
  Future<List<T>> getItems() async {
    try {
      final snapshot = await collection.get();

      return snapshot.docs
          .map((final doc) => doc.data()!.copyWithId(id: doc.id) as T)
          .toList();
    } catch (e) {
      print("Error getting items: $e");
      return [];
    }
  }

  // Update an item
```

```
  Future<T?> updateItem(final T item) async {
    try {
      final r = await collection.doc(item.id);

      await r.set(item);
      await getItem(r.id);
    } catch (e) {
      print("Error updating item: $e ${item.toJson()}");
    }
  }

  // Delete an item
  Future<void> deleteItem(final String id) async {
    try {
      await collection.doc(id).delete();
    } catch (e) {
      print("Error deleting item: $e");
    }
  }

  // Get a live stream of items
  Stream<List<T>> getItemsLive() {
    return collection.snapshots().map((final snapshot) {
      return snapshot.docs
          .map((final doc) => doc.data()!.copyWithId(id: doc.id) as T)
          .toList();
    });
  }
}
```

```dart
import "package:firebase_auth/firebase_auth.dart";
import "package:freezed_annotation/freezed_annotation.dart";
import "package:menyusha/app/data/firebase/user/user_payload.dart";

part 'auth_state.freezed.dart';

@freezed
abstract class AuthState with _$AuthState {
  AuthState._();
  factory AuthState.authInitial(final bool loginOpened, final String email, final String password, ) = A

  factory AuthState.authSuccess(final User? user, final UserPayload? userPayload) = AuthSuccess;

  factory AuthState.authFailed(final String? error) = AuthFailed;

  factory AuthState.authLoading() = AuthLoading;

  factory AuthState.authSignedOut() = AuthSignedOut;

  String get login => getLogin();

  String getLogin() {
    var email = "";
    if (this is AuthSuccess) {
      email = (this as AuthSuccess).user?.email ?? "";
    } else {
      email = "";
    }
    return getLocalPartOfEmail(email);
  }

  String getLocalPartOfEmail(final String email) {
    // Check if the email contains '@' symbol
    if (email.contains('@')) {
      // Split the email by '@' and return the first part
      return email.split('@')[0];
    } else {
      // If email does not contain '@', return an empty string or handle as needed
      return '';
    }
  }
}


/*
sealed class AuthState {
  const AuthState._();

  // Factory constructors to create specific states
  factory AuthState.authInitial(bool loginOpened) = AuthInitial;
  factory AuthState.authSuccess(User? user) = AuthSuccess;
  factory AuthState.authFailed(String? error) = AuthFailed;
  factory AuthState.authLoading() = AuthLoading;
  factory AuthState.authSignedOut() = AuthSignedOut;


  // Shared methods
  String get login => getLogin();

  String getLogin() {
    var email = "";
    if (this is AuthSuccess) {
```

```
      email = (this as AuthSuccess).user?.email ?? "";
    } else {
      email = "";
    }
    return getLocalPartOfEmail(email);
  }

  String getLocalPartOfEmail(final String email) {
    if (email.contains('@')) {
      return email.split('@')[0];
    } else {
      return '';
    }
  }
}

// AuthState subclasses
class AuthInitial extends AuthState {
  final bool loginOpened;
  const AuthInitial(this.loginOpened) : super._();
}

class AuthSuccess extends AuthState {
  final User? user;
  const AuthSuccess(this.user) : super._();
}

class AuthFailed extends AuthState {
  final String? error;
  const AuthFailed(this.error) : super._();
}

class AuthLoading extends AuthState {
  const AuthLoading() : super._();
}

class AuthSignedOut extends AuthState {
  const AuthSignedOut() : super._();
}

*/
```

```
// coverage:ignore-file
// GENERATED CODE - DO NOT MODIFY BY HAND
// ignore_for_file: type=lint
// ignore_for_file: unused_element, deprecated_member_use, deprecated_member_use_from_same_package, use_

part of 'auth_state.dart';

// **************************************************************************
// FreezedGenerator
// **************************************************************************

T _$identity<T>(T value) => value;

final _privateConstructorUsedError = UnsupportedError(
    'It seems like you constructed your class using `MyClass._()`. This constructor is only meant to be

/// @nodoc
mixin _$AuthState {
  @optionalTypeArgs
  TResult when<TResult extends Object?>({
    required TResult Function(bool loginOpened, String email, String password)
        authInitial,
    required TResult Function(User? user, UserPayload? userPayload) authSuccess,
    required TResult Function(String? error) authFailed,
    required TResult Function() authLoading,
    required TResult Function() authSignedOut,
  }) =>
      throw _privateConstructorUsedError;
  @optionalTypeArgs
  TResult? whenOrNull<TResult extends Object?>({
    TResult? Function(bool loginOpened, String email, String password)?
        authInitial,
    TResult? Function(User? user, UserPayload? userPayload)? authSuccess,
    TResult? Function(String? error)? authFailed,
    TResult? Function()? authLoading,
    TResult? Function()? authSignedOut,
  }) =>
      throw _privateConstructorUsedError;
  @optionalTypeArgs
  TResult maybeWhen<TResult extends Object?>({
    TResult Function(bool loginOpened, String email, String password)?
        authInitial,
    TResult Function(User? user, UserPayload? userPayload)? authSuccess,
    TResult Function(String? error)? authFailed,
    TResult Function()? authLoading,
    TResult Function()? authSignedOut,
    required TResult orElse(),
  }) =>
      throw _privateConstructorUsedError;
  @optionalTypeArgs
  TResult map<TResult extends Object?>({
    required TResult Function(AuthInitial value) authInitial,
    required TResult Function(AuthSuccess value) authSuccess,
    required TResult Function(AuthFailed value) authFailed,
    required TResult Function(AuthLoading value) authLoading,
    required TResult Function(AuthSignedOut value) authSignedOut,
  }) =>
      throw _privateConstructorUsedError;
  @optionalTypeArgs
  TResult? mapOrNull<TResult extends Object?>({
    TResult? Function(AuthInitial value)? authInitial,
    TResult? Function(AuthSuccess value)? authSuccess,
```

```
    TResult? Function(AuthFailed value)? authFailed,
    TResult? Function(AuthLoading value)? authLoading,
    TResult? Function(AuthSignedOut value)? authSignedOut,
  }) =>
      throw _privateConstructorUsedError;
  @optionalTypeArgs
  TResult maybeMap<TResult extends Object?>({
    TResult Function(AuthInitial value)? authInitial,
    TResult Function(AuthSuccess value)? authSuccess,
    TResult Function(AuthFailed value)? authFailed,
    TResult Function(AuthLoading value)? authLoading,
    TResult Function(AuthSignedOut value)? authSignedOut,
    required TResult orElse(),
  }) =>
      throw _privateConstructorUsedError;
}

/// @nodoc
abstract class $AuthStateCopyWith<$Res> {
  factory $AuthStateCopyWith(AuthState value, $Res Function(AuthState) then) =
      _$AuthStateCopyWithImpl<$Res, AuthState>;
}

/// @nodoc
class _$AuthStateCopyWithImpl<$Res, $Val extends AuthState>
    implements $AuthStateCopyWith<$Res> {
  _$AuthStateCopyWithImpl(this._value, this._then);

  // ignore: unused_field
  final $Val _value;
  // ignore: unused_field
  final $Res Function($Val) _then;
}

/// @nodoc
abstract class _$$AuthInitialImplCopyWith<$Res> {
  factory _$$AuthInitialImplCopyWith(
          _$AuthInitialImpl value, $Res Function(_$AuthInitialImpl) then) =
      __$$AuthInitialImplCopyWithImpl<$Res>;
  @useResult
  $Res call({bool loginOpened, String email, String password});
}

/// @nodoc
class __$$AuthInitialImplCopyWithImpl<$Res>
    extends _$AuthStateCopyWithImpl<$Res, _$AuthInitialImpl>
    implements _$$AuthInitialImplCopyWith<$Res> {
  __$$AuthInitialImplCopyWithImpl(
      _$AuthInitialImpl _value, $Res Function(_$AuthInitialImpl) _then)
      : super(_value, _then);

  @pragma('vm:prefer-inline')
  @override
  $Res call({
    Object? loginOpened = null,
    Object? email = null,
    Object? password = null,
  }) {
    return _then(_$AuthInitialImpl(
      null == loginOpened
          ? _value.loginOpened
          : loginOpened // ignore: cast_nullable_to_non_nullable
```

```
              as bool,
      null == email
          ? _value.email
          : email // ignore: cast_nullable_to_non_nullable
              as String,
      null == password
          ? _value.password
          : password // ignore: cast_nullable_to_non_nullable
              as String,
    ));
  }
}

/// @nodoc

class _$AuthInitialImpl extends AuthInitial {
  _$AuthInitialImpl(this.loginOpened, this.email, this.password) : super._();

  @override
  final bool loginOpened;
  @override
  final String email;
  @override
  final String password;

  @override
  String toString() {
    return 'AuthState.authInitial(loginOpened: $loginOpened, email: $email, password: $password)';
  }

  @override
  bool operator ==(Object other) {
    return identical(this, other) ||
        (other.runtimeType == runtimeType &&
            other is _$AuthInitialImpl &&
            (identical(other.loginOpened, loginOpened) ||
                other.loginOpened == loginOpened) &&
            (identical(other.email, email) || other.email == email) &&
            (identical(other.password, password) ||
                other.password == password));
  }

  @override
  int get hashCode => Object.hash(runtimeType, loginOpened, email, password);

  @JsonKey(ignore: true)
  @override
  @pragma('vm:prefer-inline')
  _$$AuthInitialImplCopyWith<_$AuthInitialImpl> get copyWith =>
      __$$AuthInitialImplCopyWithImpl<_$AuthInitialImpl>(this, _$identity);

  @override
  @optionalTypeArgs
  TResult when<TResult extends Object?>({
    required TResult Function(bool loginOpened, String email, String password)
        authInitial,
    required TResult Function(User? user, UserPayload? userPayload) authSuccess,
    required TResult Function(String? error) authFailed,
    required TResult Function() authLoading,
    required TResult Function() authSignedOut,
  }) {
    return authInitial(loginOpened, email, password);
```

```
}

@override
@optionalTypeArgs
TResult? whenOrNull<TResult extends Object?>({
  TResult? Function(bool loginOpened, String email, String password)?
      authInitial,
  TResult? Function(User? user, UserPayload? userPayload)? authSuccess,
  TResult? Function(String? error)? authFailed,
  TResult? Function()? authLoading,
  TResult? Function()? authSignedOut,
}) {
  return authInitial?.call(loginOpened, email, password);
}

@override
@optionalTypeArgs
TResult maybeWhen<TResult extends Object?>({
  TResult Function(bool loginOpened, String email, String password)?
      authInitial,
  TResult Function(User? user, UserPayload? userPayload)? authSuccess,
  TResult Function(String? error)? authFailed,
  TResult Function()? authLoading,
  TResult Function()? authSignedOut,
  required TResult orElse(),
}) {
  if (authInitial != null) {
    return authInitial(loginOpened, email, password);
  }
  return orElse();
}

@override
@optionalTypeArgs
TResult map<TResult extends Object?>({
  required TResult Function(AuthInitial value) authInitial,
  required TResult Function(AuthSuccess value) authSuccess,
  required TResult Function(AuthFailed value) authFailed,
  required TResult Function(AuthLoading value) authLoading,
  required TResult Function(AuthSignedOut value) authSignedOut,
}) {
  return authInitial(this);
}

@override
@optionalTypeArgs
TResult? mapOrNull<TResult extends Object?>({
  TResult? Function(AuthInitial value)? authInitial,
  TResult? Function(AuthSuccess value)? authSuccess,
  TResult? Function(AuthFailed value)? authFailed,
  TResult? Function(AuthLoading value)? authLoading,
  TResult? Function(AuthSignedOut value)? authSignedOut,
}) {
  return authInitial?.call(this);
}

@override
@optionalTypeArgs
TResult maybeMap<TResult extends Object?>({
  TResult Function(AuthInitial value)? authInitial,
  TResult Function(AuthSuccess value)? authSuccess,
  TResult Function(AuthFailed value)? authFailed,
```

```
    TResult Function(AuthLoading value)? authLoading,
    TResult Function(AuthSignedOut value)? authSignedOut,
    required TResult orElse(),
  }) {
    if (authInitial != null) {
      return authInitial(this);
    }
    return orElse();
  }
}

abstract class AuthInitial extends AuthState {
  factory AuthInitial(
          final bool loginOpened, final String email, final String password) =
      _$AuthInitialImpl;
  AuthInitial._() : super._();

  bool get loginOpened;
  String get email;
  String get password;
  @JsonKey(ignore: true)
  _$$AuthInitialImplCopyWith<_$AuthInitialImpl> get copyWith =>
      throw _privateConstructorUsedError;
}

/// @nodoc
abstract class _$$AuthSuccessImplCopyWith<$Res> {
  factory _$$AuthSuccessImplCopyWith(
          _$AuthSuccessImpl value, $Res Function(_$AuthSuccessImpl) then) =
      __$$AuthSuccessImplCopyWithImpl<$Res>;
  @useResult
  $Res call({User? user, UserPayload? userPayload});
}

/// @nodoc
class __$$AuthSuccessImplCopyWithImpl<$Res>
    extends _$AuthStateCopyWithImpl<$Res, _$AuthSuccessImpl>
    implements _$$AuthSuccessImplCopyWith<$Res> {
  __$$AuthSuccessImplCopyWithImpl(
      _$AuthSuccessImpl _value, $Res Function(_$AuthSuccessImpl) _then)
      : super(_value, _then);

  @pragma('vm:prefer-inline')
  @override
  $Res call({
    Object? user = freezed,
    Object? userPayload = freezed,
  }) {
    return _then(_$AuthSuccessImpl(
      freezed == user
          ? _value.user
          : user // ignore: cast_nullable_to_non_nullable
              as User?,
      freezed == userPayload
          ? _value.userPayload
          : userPayload // ignore: cast_nullable_to_non_nullable
              as UserPayload?,
    ));
  }
}

/// @nodoc
```

```
class _$AuthSuccessImpl extends AuthSuccess {
  _$AuthSuccessImpl(this.user, this.userPayload) : super._();

  @override
  final User? user;
  @override
  final UserPayload? userPayload;

  @override
  String toString() {
    return 'AuthState.authSuccess(user: $user, userPayload: $userPayload)';
  }

  @override
  bool operator ==(Object other) {
    return identical(this, other) ||
        (other.runtimeType == runtimeType &&
            other is _$AuthSuccessImpl &&
            (identical(other.user, user) || other.user == user) &&
            (identical(other.userPayload, userPayload) ||
                other.userPayload == userPayload));
  }

  @override
  int get hashCode => Object.hash(runtimeType, user, userPayload);

  @JsonKey(ignore: true)
  @override
  @pragma('vm:prefer-inline')
  _$$AuthSuccessImplCopyWith<_$AuthSuccessImpl> get copyWith =>
      __$$AuthSuccessImplCopyWithImpl<_$AuthSuccessImpl>(this, _$identity);

  @override
  @optionalTypeArgs
  TResult when<TResult extends Object?>({
    required TResult Function(bool loginOpened, String email, String password)
        authInitial,
    required TResult Function(User? user, UserPayload? userPayload) authSuccess,
    required TResult Function(String? error) authFailed,
    required TResult Function() authLoading,
    required TResult Function() authSignedOut,
  }) {
    return authSuccess(user, userPayload);
  }

  @override
  @optionalTypeArgs
  TResult? whenOrNull<TResult extends Object?>({
    TResult? Function(bool loginOpened, String email, String password)?
        authInitial,
    TResult? Function(User? user, UserPayload? userPayload)? authSuccess,
    TResult? Function(String? error)? authFailed,
    TResult? Function()? authLoading,
    TResult? Function()? authSignedOut,
  }) {
    return authSuccess?.call(user, userPayload);
  }

  @override
  @optionalTypeArgs
  TResult maybeWhen<TResult extends Object?>({
```

```
    TResult Function(bool loginOpened, String email, String password)?
        authInitial,
    TResult Function(User? user, UserPayload? userPayload)? authSuccess,
    TResult Function(String? error)? authFailed,
    TResult Function()? authLoading,
    TResult Function()? authSignedOut,
    required TResult orElse(),
  }) {
    if (authSuccess != null) {
      return authSuccess(user, userPayload);
    }
    return orElse();
  }

  @override
  @optionalTypeArgs
  TResult map<TResult extends Object?>({
    required TResult Function(AuthInitial value) authInitial,
    required TResult Function(AuthSuccess value) authSuccess,
    required TResult Function(AuthFailed value) authFailed,
    required TResult Function(AuthLoading value) authLoading,
    required TResult Function(AuthSignedOut value) authSignedOut,
  }) {
    return authSuccess(this);
  }

  @override
  @optionalTypeArgs
  TResult? mapOrNull<TResult extends Object?>({
    TResult? Function(AuthInitial value)? authInitial,
    TResult? Function(AuthSuccess value)? authSuccess,
    TResult? Function(AuthFailed value)? authFailed,
    TResult? Function(AuthLoading value)? authLoading,
    TResult? Function(AuthSignedOut value)? authSignedOut,
  }) {
    return authSuccess?.call(this);
  }

  @override
  @optionalTypeArgs
  TResult maybeMap<TResult extends Object?>({
    TResult Function(AuthInitial value)? authInitial,
    TResult Function(AuthSuccess value)? authSuccess,
    TResult Function(AuthFailed value)? authFailed,
    TResult Function(AuthLoading value)? authLoading,
    TResult Function(AuthSignedOut value)? authSignedOut,
    required TResult orElse(),
  }) {
    if (authSuccess != null) {
      return authSuccess(this);
    }
    return orElse();
  }
}

abstract class AuthSuccess extends AuthState {
  factory AuthSuccess(final User? user, final UserPayload? userPayload) =
      _$AuthSuccessImpl;
  AuthSuccess._() : super._();

  User? get user;
  UserPayload? get userPayload;
```

```
  @JsonKey(ignore: true)
  _$$AuthSuccessImplCopyWith<_$AuthSuccessImpl> get copyWith =>
      throw _privateConstructorUsedError;
}

/// @nodoc
abstract class _$$AuthFailedImplCopyWith<$Res> {
  factory _$$AuthFailedImplCopyWith(
          _$AuthFailedImpl value, $Res Function(_$AuthFailedImpl) then) =
      __$$AuthFailedImplCopyWithImpl<$Res>;
  @useResult
  $Res call({String? error});
}

/// @nodoc
class __$$AuthFailedImplCopyWithImpl<$Res>
    extends _$AuthStateCopyWithImpl<$Res, _$AuthFailedImpl>
    implements _$$AuthFailedImplCopyWith<$Res> {
  __$$AuthFailedImplCopyWithImpl(
      _$AuthFailedImpl _value, $Res Function(_$AuthFailedImpl) _then)
      : super(_value, _then);

  @pragma('vm:prefer-inline')
  @override
  $Res call({
    Object? error = freezed,
  }) {
    return _then(_$AuthFailedImpl(
      freezed == error
          ? _value.error
          : error // ignore: cast_nullable_to_non_nullable
              as String?,
    ));
  }
}

/// @nodoc

class _$AuthFailedImpl extends AuthFailed {
  _$AuthFailedImpl(this.error) : super._();

  @override
  final String? error;

  @override
  String toString() {
    return 'AuthState.authFailed(error: $error)';
  }

  @override
  bool operator ==(Object other) {
    return identical(this, other) ||
        (other.runtimeType == runtimeType &&
            other is _$AuthFailedImpl &&
            (identical(other.error, error) || other.error == error));
  }

  @override
  int get hashCode => Object.hash(runtimeType, error);

  @JsonKey(ignore: true)
  @override
```

```
@pragma('vm:prefer-inline')
_$$AuthFailedImplCopyWith<_$AuthFailedImpl> get copyWith =>
    __$$AuthFailedImplCopyWithImpl<_$AuthFailedImpl>(this, _$identity);

@override
@optionalTypeArgs
TResult when<TResult extends Object?>({
  required TResult Function(bool loginOpened, String email, String password)
      authInitial,
  required TResult Function(User? user, UserPayload? userPayload) authSuccess,
  required TResult Function(String? error) authFailed,
  required TResult Function() authLoading,
  required TResult Function() authSignedOut,
}) {
  return authFailed(error);
}

@override
@optionalTypeArgs
TResult? whenOrNull<TResult extends Object?>({
  TResult? Function(bool loginOpened, String email, String password)?
      authInitial,
  TResult? Function(User? user, UserPayload? userPayload)? authSuccess,
  TResult? Function(String? error)? authFailed,
  TResult? Function()? authLoading,
  TResult? Function()? authSignedOut,
}) {
  return authFailed?.call(error);
}

@override
@optionalTypeArgs
TResult maybeWhen<TResult extends Object?>({
  TResult Function(bool loginOpened, String email, String password)?
      authInitial,
  TResult Function(User? user, UserPayload? userPayload)? authSuccess,
  TResult Function(String? error)? authFailed,
  TResult Function()? authLoading,
  TResult Function()? authSignedOut,
  required TResult orElse(),
}) {
  if (authFailed != null) {
    return authFailed(error);
  }
  return orElse();
}

@override
@optionalTypeArgs
TResult map<TResult extends Object?>({
  required TResult Function(AuthInitial value) authInitial,
  required TResult Function(AuthSuccess value) authSuccess,
  required TResult Function(AuthFailed value) authFailed,
  required TResult Function(AuthLoading value) authLoading,
  required TResult Function(AuthSignedOut value) authSignedOut,
}) {
  return authFailed(this);
}

@override
@optionalTypeArgs
TResult? mapOrNull<TResult extends Object?>({
```

```
    TResult? Function(AuthInitial value)? authInitial,
    TResult? Function(AuthSuccess value)? authSuccess,
    TResult? Function(AuthFailed value)? authFailed,
    TResult? Function(AuthLoading value)? authLoading,
    TResult? Function(AuthSignedOut value)? authSignedOut,
  }) {
    return authFailed?.call(this);
  }

  @override
  @optionalTypeArgs
  TResult maybeMap<TResult extends Object?>({
    TResult Function(AuthInitial value)? authInitial,
    TResult Function(AuthSuccess value)? authSuccess,
    TResult Function(AuthFailed value)? authFailed,
    TResult Function(AuthLoading value)? authLoading,
    TResult Function(AuthSignedOut value)? authSignedOut,
    required TResult orElse(),
  }) {
    if (authFailed != null) {
      return authFailed(this);
    }
    return orElse();
  }
}

abstract class AuthFailed extends AuthState {
  factory AuthFailed(final String? error) = _$AuthFailedImpl;
  AuthFailed._() : super._();

  String? get error;
  @JsonKey(ignore: true)
  _$$AuthFailedImplCopyWith<_$AuthFailedImpl> get copyWith =>
      throw _privateConstructorUsedError;
}

/// @nodoc
abstract class _$$AuthLoadingImplCopyWith<$Res> {
  factory _$$AuthLoadingImplCopyWith(
          _$AuthLoadingImpl value, $Res Function(_$AuthLoadingImpl) then) =
      __$$AuthLoadingImplCopyWithImpl<$Res>;
}

/// @nodoc
class __$$AuthLoadingImplCopyWithImpl<$Res>
    extends _$AuthStateCopyWithImpl<$Res, _$AuthLoadingImpl>
    implements _$$AuthLoadingImplCopyWith<$Res> {
  __$$AuthLoadingImplCopyWithImpl(
      _$AuthLoadingImpl _value, $Res Function(_$AuthLoadingImpl) _then)
      : super(_value, _then);
}

/// @nodoc

class _$AuthLoadingImpl extends AuthLoading {
  _$AuthLoadingImpl() : super._();

  @override
  String toString() {
    return 'AuthState.authLoading()';
  }
```

```
@override
bool operator ==(Object other) {
  return identical(this, other) ||
      (other.runtimeType == runtimeType && other is _$AuthLoadingImpl);
}

@override
int get hashCode => runtimeType.hashCode;

@override
@optionalTypeArgs
TResult when<TResult extends Object?>({
  required TResult Function(bool loginOpened, String email, String password)
      authInitial,
  required TResult Function(User? user, UserPayload? userPayload) authSuccess,
  required TResult Function(String? error) authFailed,
  required TResult Function() authLoading,
  required TResult Function() authSignedOut,
}) {
  return authLoading();
}

@override
@optionalTypeArgs
TResult? whenOrNull<TResult extends Object?>({
  TResult? Function(bool loginOpened, String email, String password)?
      authInitial,
  TResult? Function(User? user, UserPayload? userPayload)? authSuccess,
  TResult? Function(String? error)? authFailed,
  TResult? Function()? authLoading,
  TResult? Function()? authSignedOut,
}) {
  return authLoading?.call();
}

@override
@optionalTypeArgs
TResult maybeWhen<TResult extends Object?>({
  TResult Function(bool loginOpened, String email, String password)?
      authInitial,
  TResult Function(User? user, UserPayload? userPayload)? authSuccess,
  TResult Function(String? error)? authFailed,
  TResult Function()? authLoading,
  TResult Function()? authSignedOut,
  required TResult orElse(),
}) {
  if (authLoading != null) {
    return authLoading();
  }
  return orElse();
}

@override
@optionalTypeArgs
TResult map<TResult extends Object?>({
  required TResult Function(AuthInitial value) authInitial,
  required TResult Function(AuthSuccess value) authSuccess,
  required TResult Function(AuthFailed value) authFailed,
  required TResult Function(AuthLoading value) authLoading,
  required TResult Function(AuthSignedOut value) authSignedOut,
}) {
  return authLoading(this);
```

```
  }

  @override
  @optionalTypeArgs
  TResult? mapOrNull<TResult extends Object?>({
    TResult? Function(AuthInitial value)? authInitial,
    TResult? Function(AuthSuccess value)? authSuccess,
    TResult? Function(AuthFailed value)? authFailed,
    TResult? Function(AuthLoading value)? authLoading,
    TResult? Function(AuthSignedOut value)? authSignedOut,
  }) {
    return authLoading?.call(this);
  }

  @override
  @optionalTypeArgs
  TResult maybeMap<TResult extends Object?>({
    TResult Function(AuthInitial value)? authInitial,
    TResult Function(AuthSuccess value)? authSuccess,
    TResult Function(AuthFailed value)? authFailed,
    TResult Function(AuthLoading value)? authLoading,
    TResult Function(AuthSignedOut value)? authSignedOut,
    required TResult orElse(),
  }) {
    if (authLoading != null) {
      return authLoading(this);
    }
    return orElse();
  }
}

abstract class AuthLoading extends AuthState {
  factory AuthLoading() = _$AuthLoadingImpl;
  AuthLoading._() : super._();
}

/// @nodoc
abstract class _$$AuthSignedOutImplCopyWith<$Res> {
  factory _$$AuthSignedOutImplCopyWith(
          _$AuthSignedOutImpl value, $Res Function(_$AuthSignedOutImpl) then) =
      __$$AuthSignedOutImplCopyWithImpl<$Res>;
}

/// @nodoc
class __$$AuthSignedOutImplCopyWithImpl<$Res>
    extends _$AuthStateCopyWithImpl<$Res, _$AuthSignedOutImpl>
    implements _$$AuthSignedOutImplCopyWith<$Res> {
  __$$AuthSignedOutImplCopyWithImpl(
      _$AuthSignedOutImpl _value, $Res Function(_$AuthSignedOutImpl) _then)
      : super(_value, _then);
}

/// @nodoc

class _$AuthSignedOutImpl extends AuthSignedOut {
  _$AuthSignedOutImpl() : super._();

  @override
  String toString() {
    return 'AuthState.authSignedOut()';
  }
```

```
@override
bool operator ==(Object other) {
  return identical(this, other) ||
      (other.runtimeType == runtimeType && other is _$AuthSignedOutImpl);
}

@override
int get hashCode => runtimeType.hashCode;

@override
@optionalTypeArgs
TResult when<TResult extends Object?>({
  required TResult Function(bool loginOpened, String email, String password)
      authInitial,
  required TResult Function(User? user, UserPayload? userPayload) authSuccess,
  required TResult Function(String? error) authFailed,
  required TResult Function() authLoading,
  required TResult Function() authSignedOut,
}) {
  return authSignedOut();
}

@override
@optionalTypeArgs
TResult? whenOrNull<TResult extends Object?>({
  TResult? Function(bool loginOpened, String email, String password)?
      authInitial,
  TResult? Function(User? user, UserPayload? userPayload)? authSuccess,
  TResult? Function(String? error)? authFailed,
  TResult? Function()? authLoading,
  TResult? Function()? authSignedOut,
}) {
  return authSignedOut?.call();
}

@override
@optionalTypeArgs
TResult maybeWhen<TResult extends Object?>({
  TResult Function(bool loginOpened, String email, String password)?
      authInitial,
  TResult Function(User? user, UserPayload? userPayload)? authSuccess,
  TResult Function(String? error)? authFailed,
  TResult Function()? authLoading,
  TResult Function()? authSignedOut,
  required TResult orElse(),
}) {
  if (authSignedOut != null) {
    return authSignedOut();
  }
  return orElse();
}

@override
@optionalTypeArgs
TResult map<TResult extends Object?>({
  required TResult Function(AuthInitial value) authInitial,
  required TResult Function(AuthSuccess value) authSuccess,
  required TResult Function(AuthFailed value) authFailed,
  required TResult Function(AuthLoading value) authLoading,
  required TResult Function(AuthSignedOut value) authSignedOut,
}) {
  return authSignedOut(this);
```

```
  }

  @override
  @optionalTypeArgs
  TResult? mapOrNull<TResult extends Object?>({
    TResult? Function(AuthInitial value)? authInitial,
    TResult? Function(AuthSuccess value)? authSuccess,
    TResult? Function(AuthFailed value)? authFailed,
    TResult? Function(AuthLoading value)? authLoading,
    TResult? Function(AuthSignedOut value)? authSignedOut,
  }) {
    return authSignedOut?.call(this);
  }

  @override
  @optionalTypeArgs
  TResult maybeMap<TResult extends Object?>({
    TResult Function(AuthInitial value)? authInitial,
    TResult Function(AuthSuccess value)? authSuccess,
    TResult Function(AuthFailed value)? authFailed,
    TResult Function(AuthLoading value)? authLoading,
    TResult Function(AuthSignedOut value)? authSignedOut,
    required TResult orElse(),
  }) {
    if (authSignedOut != null) {
      return authSignedOut(this);
    }
    return orElse();
  }
}

abstract class AuthSignedOut extends AuthState {
  factory AuthSignedOut() = _$AuthSignedOutImpl;
  AuthSignedOut._() : super._();
}
```

```dart
import "package:firebase_auth/firebase_auth.dart";
import "package:menyusha/app/data/firebase/user/user_payload.dart";

import "../repository/firestore_repository.dart";

class UserPayloadRepository extends FirestoreRepository<UserPayload> {
  UserPayloadRepository() : super("payloads", UserPayload.fromJson);

  // Create
  Future<UserPayload?> createUser(final String uid) async {
    try {
      final newPayload = UserPayload(
        id: "",
        uid: uid,
      );

      final createdPayload = await createItem(newPayload);
      return createdPayload;
    } catch (e) {
      print("Error creating user: $e");
      return null;
    }
  }

  // Read
  Future<UserPayload?> getUserByUID(final String uid) async {
    try {
      final querySnapshot =
          await collection.where("uid", isEqualTo: uid).limit(1).get();
      if (querySnapshot.docs.isNotEmpty) {
        final userDoc = querySnapshot.docs.first;
        return userDoc.data();
      } else {
        return null;
      }
    } catch (e) {
      print("Error finding user by UID: $e");
      return null;
    }
  }

  Future<UserPayload?> findOrCreateUserPayloadByUID(final String uid) async {
    final existingUser = await getUserByUID(uid);
    return existingUser ?? await createUser(uid);
  }

  // Update
  Future<UserPayload?> updateUser(final UserPayload user) async {
    try {
      await updateItem(user);
      print("User updated successfully");
      return user;
    } catch (e) {
      print("Error updating user: $e");
      return null;
    }
  }

  // Delete
  Future<void> deleteUser(final String id) async {
    try {
      await deleteItem(id);
```

```
      print("User deleted successfully");
    } catch (e) {
      print("Error deleting user: $e");
    }
  }
}
```

```dart
import "package:cloud_firestore/cloud_firestore.dart";
import "package:uuid/uuid.dart";

import "../repository/base_model.dart";


class UserPayload implements BaseModel {
  UserPayload({
    required this.id,
    required this.uid,
  });

  // Convert JSON to UserPayload
  factory UserPayload.fromJson(Map<String, dynamic> json) => UserPayload(
      id: json['id'] as String,
      uid: json['uid'] as String,
    );

  @override
  final String id;
  final String uid;


  // Copy with new values, including id
  UserPayload copyWith({
    String? id,
    String? uid,
  }) =>
      UserPayload(
        id: id ?? this.id,
        uid: uid ?? this.uid,
      );

  // Implement copyWithId from BaseModel
  @override
  UserPayload copyWithId({final String? id}) => copyWith(id: id);

  // Convert UserPayload to JSON
  @override
  Map<String, dynamic> toJson() => {
    'id': id,
    'uid': uid,
  };
}
```

```dart
import "package:menyusha/app/data/firebase/menu/menu_payload.dart";
import "package:menyusha/app/data/firebase/user/user_payload.dart";

import "../repository/firestore_repository.dart";

class MenuPayloadRepository extends FirestoreRepository<MenuPayload> {
  MenuPayloadRepository() : super("menu_payload", MenuPayload.fromJson);

  // Method to get all items by a specific user ID
  Future<List<MenuPayload>> getItemsByUserId(String userId) async {
    try {
      final snapshot =
          await collection.where('userId', isEqualTo: userId).get();

      return snapshot.docs
          .map((doc) => doc.data()!.copyWithId(id: doc.id) as MenuPayload)
          .toList();
    } catch (e) {
      print("Error getting items by user ID: $e");
      return [];
    }
  }

  // Method to get a live stream of items filtered by user ID
  Stream<List<MenuPayload>> getItemsLiveByUserId(String userId) {
    return collection
        .where('userId', isEqualTo: userId)
        .snapshots()
        .map((snapshot) {
      return snapshot.docs
          .map((doc) => doc.data()!.copyWithId(id: doc.id) as MenuPayload)
          .toList();
    });
  }

  // Function to create a new MenuPayload
  Future<MenuPayload?> createMenu(MenuPayload payload) async =>
      createItem(payload);

  Future<MenuPayload?> updateMenu(MenuPayload payload) async =>
      updateItem(payload);

  Future<void> deleteMenu(MenuPayload payload) async => deleteItem(payload.id);

  // Method to get a single item by publicId
  Future<MenuPayload?> getItemByPublicId(String publicId) async {
    try {
      final snapshot = await collection
          .where('publicId', isEqualTo: publicId)
          .limit(1)
          .get();

      if (snapshot.docs.isNotEmpty) {
        final doc = snapshot.docs.first;
        return doc.data()!.copyWithId(id: doc.id) as MenuPayload;
      } else {
        print("No item found with publicId: $publicId");
        return null;
      }
    } catch (e) {
      print("Error getting item by publicId: $e");
      return null;
```

```
    }
  }
}
```

```dart
import "../repository/base_model.dart";

enum DesignPreset {
  BANDANA,
  SANTOKU,
  BBQ,
  BOSTON,
  VOLGA,
  CUSTOM;

  // Convert enum to string
  String toJson() => name;

  // Convert string back to enum
  static DesignPreset fromJson(String json) => DesignPreset.values.byName(json);
}

class MenuPayload implements BaseModel {
  MenuPayload({
    required this.id,
    required this.publicId,
    required this.userId,
    required this.title,
    required this.menu,
  });// Generate a unique ID if not provided

  @override
  final String id;
  final String publicId;
  final String userId;
  final String title;
  final Menu menu;

  MenuPayload copyWith({
    String? id,
    String? publicId,
    String? userId,
    String? title,
    Menu? menu,
  }) =>
      MenuPayload(
        id: id ?? this.id,
        publicId: publicId ?? this.publicId,
        userId: userId ?? this.userId,
        title: title ?? this.title,
        menu: menu ?? this.menu,
      );

  Map<String, dynamic> toJson() => {
    'id': id,
    'publicId': publicId,
    'userId': userId,
    'title': title,
    'menu': menu?.toJson(),
  };

  factory MenuPayload.fromJson(Map<String, dynamic> json) {
    try {
      return MenuPayload(
        id: json['id'] as String,
        publicId: json['publicId'] as String,
```

```
        userId: json['userId'] as String,
        title: json['title'] as String,
        menu: Menu.fromJson(json['menu'] as Map<String, dynamic>),
      );
    } catch (e) {
      throw FormatException("Invalid JSON format for MenuHolder: $e");
    }
  }

  @override
  BaseModel copyWithId({String? id}) {
    return copyWith(id: id);
  }
}

class Menu {
  Menu({
    required this.designPreset,
    required this.titleSrc,
    required this.positions,
  });

  final DesignPreset designPreset;
  final String titleSrc;
  final List<Position> positions;

  Map<String, List<Position>> get groupedPositions {
    final grouped = <String, List<Position>>{};
    for (final dish in positions) {
      if (grouped.containsKey(dish.group)) {
        grouped[dish.group]!.add(dish);
      } else {
        grouped[dish.group] = [dish];
      }
    }
    return grouped;
  }
  Menu copyWith({
    DesignPreset? designPreset,
    String? titleSrc,
    List<Position>? positions,
  }) =>
      Menu(
        designPreset: designPreset ?? this.designPreset,
        titleSrc: titleSrc ?? this.titleSrc,
        positions: positions ?? this.positions,
      );
  Map<String, dynamic> toJson() => {
    'designPreset': designPreset.toJson(),
    'titleSrc': titleSrc,
    'positions': positions.map((pos) => pos.toJson()).toList(),
  };
  factory Menu.fromJson(Map<String, dynamic> json) {
    return Menu(
      designPreset: DesignPreset.fromJson(json['designPreset'] as String),
      titleSrc: json['titleSrc'] as String,
      positions: (json['positions'] as List<dynamic>)
          .map((item) => Position.fromJson(item as Map<String, dynamic>))
          .toList(),
    );
  }
}
```

```
class Position {
  Position({
    required this.id,
    required this.group,
    required this.title,
    required this.description,
    required this.price,
    required this.output,
  });
  final String id;
  final String group;
  final String title;
  final String description;
  final double price;
  final String output;
  Position copyWith({
    String? id,
    String? group,
    String? title,
    String? description,
    double? price,
    String? output,
  }) =>
      Position(
        id: id ?? this.id,
        group: group ?? this.group,
        title: title ?? this.title,
        description: description ?? this.description,
        price: price ?? this.price,
        output: output ?? this.output,
      );
  Map<String, dynamic> toJson() => {
    'id': id,
    'group': group,
    'title': title,
    'description': description,
    'price': price,
    'output': output,
  };
  factory Position.fromJson(Map<String, dynamic> json) {
    return Position(
      id: json['id'] as String,
      group: json['group'] as String,
      title: json['title'] as String,
      description: json['description'] as String,
      price: (json['price'] as num).toDouble(),
      output: json['output'] as String,
    );
  }
}
```

```dart
import 'dart:html';
import 'dart:convert';

import "package:menyusha/app/data/firebase/user/user_payload.dart";

class UserManager {
  static const String _userKey = 'user';
  bool _isUserLoggedIn = false;

  UserManager() {
    // Initialize the logged-in status from localStorage
    _isUserLoggedIn = window.localStorage.containsKey(_userKey);
  }

  // Save user data to local storage
  Future<void> saveUser(UserPayload user) async {
    final StorageService storage = StorageService();
    String userJson = jsonEncode(user.toJson());
    await storage.setItem(_userKey, userJson);
    _isUserLoggedIn = true;
  }

  // Get user data from local storage
  Future<UserPayload?> getUser() async {
    final StorageService storage = StorageService();
    String? userJson = await storage.getItem(_userKey);
    if (userJson != null) {
      Map<String, dynamic> userMap = jsonDecode(userJson);
      return UserPayload.fromJson(userMap);
    }
    return null;
  }

  // Check if user is logged in asynchronously
  Future<bool> isUserLoggedIn() async {
    final StorageService storage = StorageService();
    return await storage.containsKey(_userKey);
  }

  // Check if user is logged in synchronously
  bool isUserLoggedInSync() {
    return _isUserLoggedIn;
  }

  // Log out user by removing user data from local storage
  Future<void> logout() async {
    final StorageService storage = StorageService();
    await storage.removeItem(_userKey);
    _isUserLoggedIn = false;
  }
}

class StorageService {
  // Save item to localStorage
  Future<void> setItem(final String key, final String value) async {
    window.localStorage[key] = value;
  }

  // Get item from localStorage
  Future<String?> getItem(final String key) async => window.localStorage[key];

  // Remove item from localStorage
```

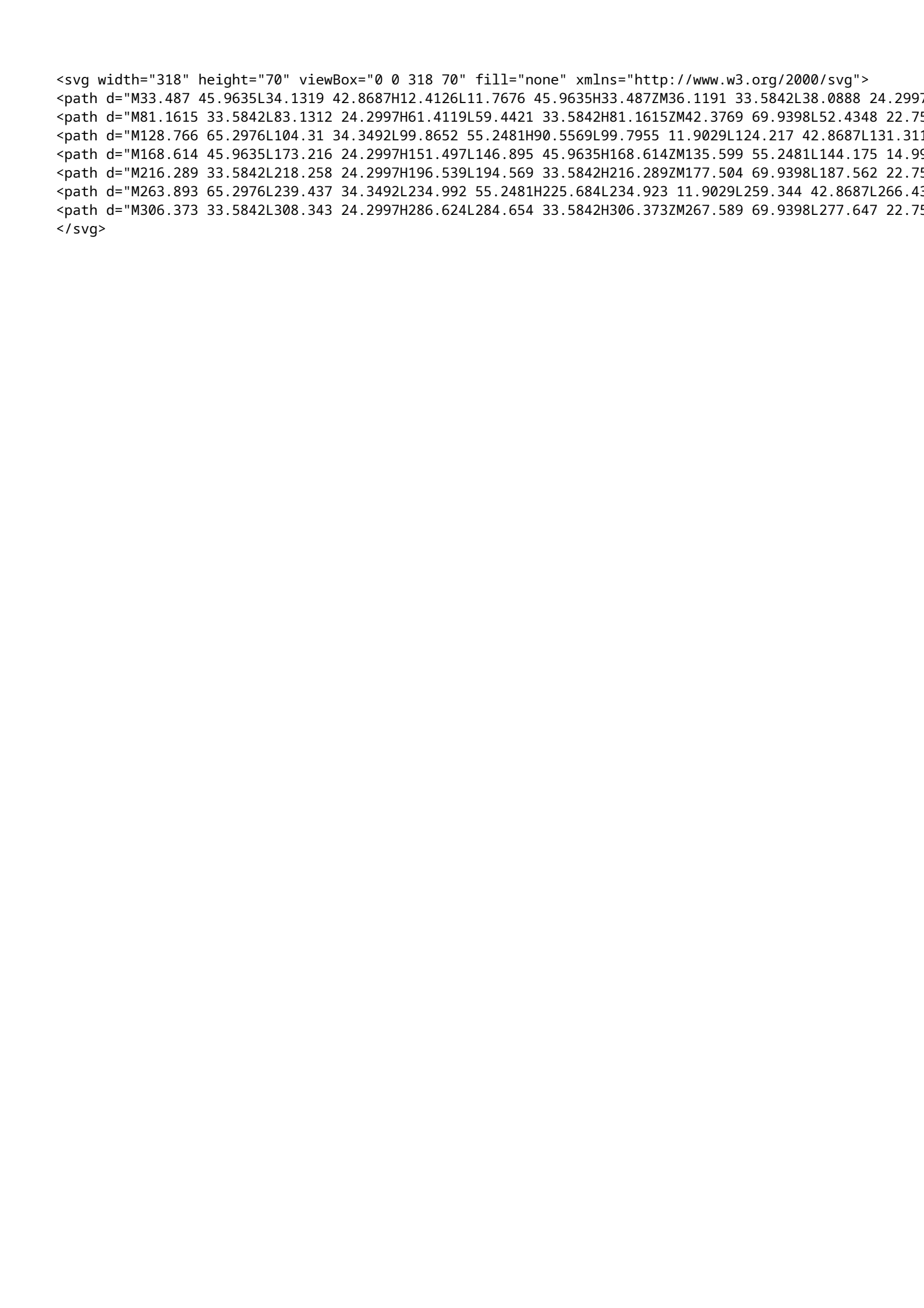
```
  Future<void> removeItem(final String key) async {
    window.localStorage.remove(key);
  }

  // Check if item exists in localStorage
  Future<bool> containsKey(final String key) async =>
      window.localStorage.containsKey(key);
}

void main() async {
  final userManager = UserManager();

  // Example Usage
  final user = UserPayload(id: "test", uid: "TEST UID");

  // Save user
  await userManager.saveUser(user);

  // Check if user is logged in
  var isLoggedIn = await userManager.isUserLoggedIn();
  print('Is user logged in? $isLoggedIn');

  // Get user info
  final UserPayload? fetchedUser = await userManager.getUser();
  if (fetchedUser != null) {
    print('User Name: ${fetchedUser.uid}');
  }

  // Log out user
  await userManager.logout();
  isLoggedIn = await userManager.isUserLoggedIn();
  print('Is user logged in after logout? $isLoggedIn');
}
```

```dart
import "package:cloud_firestore/cloud_firestore.dart";

import "../repository/base_model.dart";

class FirestoreService {
  final FirebaseFirestore _firestore = FirebaseFirestore.instance;

  CollectionReference<T> getCollection<T extends BaseModel>(
          final String path, final T Function(Map<String, dynamic>) fromJson) =>
      _firestore.collection(path).withConverter<T>(
            fromFirestore: (final snapshot, final _) =>
                fromJson(snapshot.data()!),
            toFirestore: (final model, final _) => model.toJson(),
          );
}
```

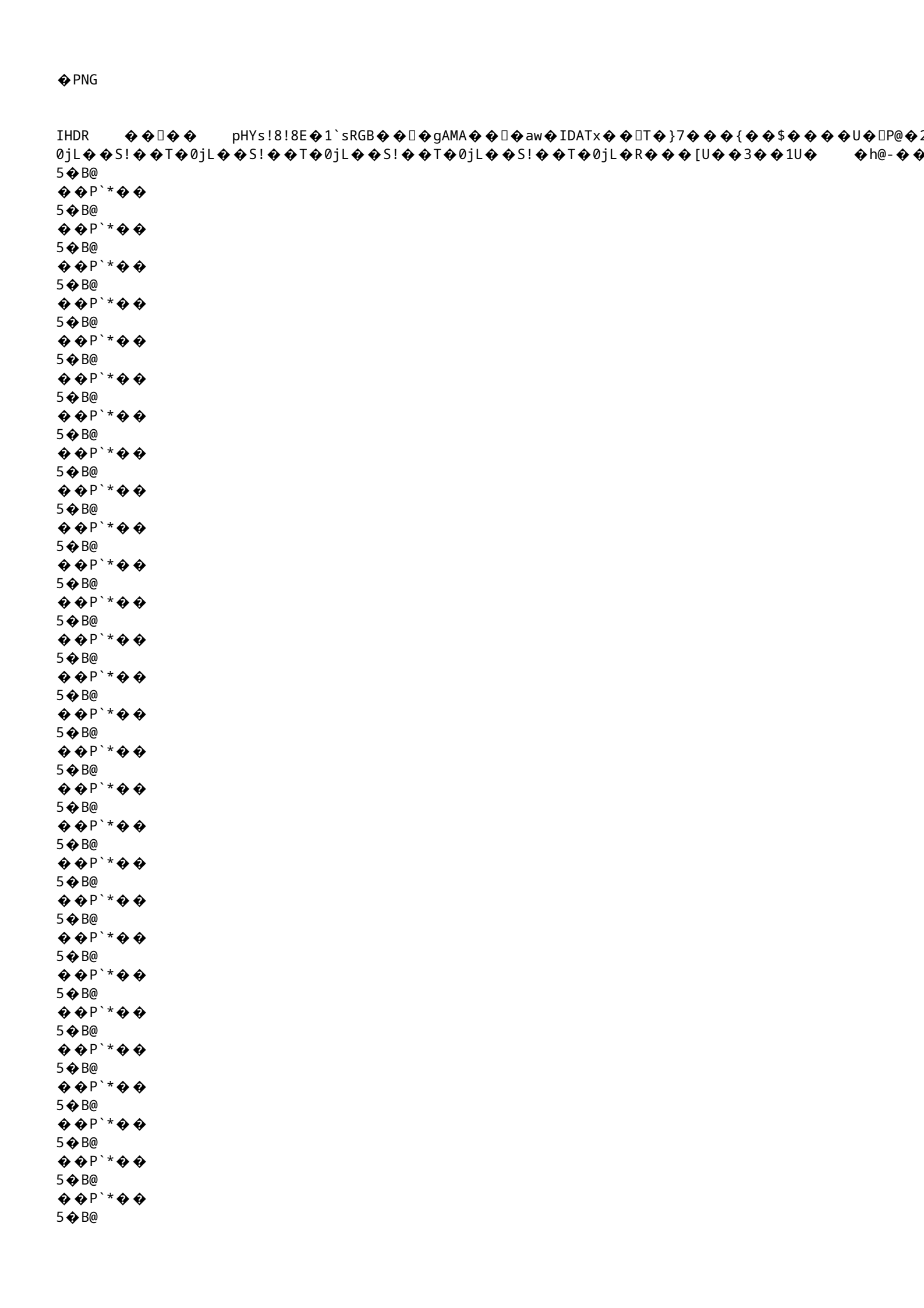
```dart
// File generated by FlutterFire CLI.
// ignore_for_file: lines_longer_than_80_chars, avoid_classes_with_only_static_members
import "package:firebase_core/firebase_core.dart" show FirebaseOptions;
import "package:flutter/foundation.dart"
    show defaultTargetPlatform, kIsWeb, TargetPlatform;
import "package:flutter_dotenv/flutter_dotenv.dart";

/// Default [FirebaseOptions] for use with your Firebase apps.
///
/// Example:
/// ```dart
/// import 'firebase_options.dart';
/// // ...
/// await Firebase.initializeApp(
///   options: DefaultFirebaseOptions.currentPlatform,
/// );
/// ```
class DefaultFirebaseOptions {
  static FirebaseOptions get currentPlatform {
    if (kIsWeb) {
      return web;
    }
    switch (defaultTargetPlatform) {
      case TargetPlatform.android:
        throw UnsupportedError(
          "DefaultFirebaseOptions have not been configured for android - "
          "you can reconfigure this by running the FlutterFire CLI again.",
        );
      case TargetPlatform.iOS:
        throw UnsupportedError(
          "DefaultFirebaseOptions have not been configured for ios - "
          "you can reconfigure this by running the FlutterFire CLI again.",
        );
      case TargetPlatform.macOS:
        throw UnsupportedError(
          "DefaultFirebaseOptions have not been configured for macos - "
          "you can reconfigure this by running the FlutterFire CLI again.",
        );
      case TargetPlatform.windows:
        throw UnsupportedError(
          "DefaultFirebaseOptions have not been configured for windows - "
          "you can reconfigure this by running the FlutterFire CLI again.",
        );
      case TargetPlatform.linux:
        throw UnsupportedError(
          "DefaultFirebaseOptions have not been configured for linux - "
          "you can reconfigure this by running the FlutterFire CLI again.",
        );
      default:
        throw UnsupportedError(
          "DefaultFirebaseOptions are not supported for this platform.",
        );
    }
  }

  static FirebaseOptions web = FirebaseOptions(
    apiKey: dotenv.env["fbApiKey"] ?? "",
    appId: dotenv.env["fbAppId"] ?? "",
    messagingSenderId: dotenv.env["fbMessagingSenderId"] ?? "",
    projectId: dotenv.env["fbProjectId"] ?? "",
    authDomain: dotenv.env["fbAuthDomain"] ?? "",
    storageBucket: dotenv.env["fbStorageBucket"] ?? "",
```

```
    measurementId: dotenv.env["fbMeasurementId"] ?? "",
  );
}
```

```dart
import "package:firebase_auth/firebase_auth.dart";
import "package:firebase_core/firebase_core.dart";
import "package:flutter/material.dart";
import "package:flutter_web_plugins/url_strategy.dart";

import "package:menyusha/app/env/environment.dart";
import "package:menyusha/app/root/root_component.dart";
import "package:menyusha/firebase_options.dart";

import "app/BuildConfig.dart";

///Public firebase
late final FirebaseApp firebaseApp;

///Public firebase
late final FirebaseAuth firebaseAuth;
late final BuildConfig buildConfig;

Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  usePathUrlStrategy();
  await environmentInit();
  buildConfig = await loadBuildConfig();
  firebaseApp = await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );

  firebaseAuth = FirebaseAuth.instanceFor(app: firebaseApp);

  runApp(RootComponent());
}
```


<svg width="318" height="70" viewBox="0 0 318 70" fill="none" xmlns="http://www.w3.org/2000/svg">
<path d="M33.487 45.9635L34.1319 42.8687H12.4126L11.7676 45.9635H33.487ZM36.1191 33.5842L38.0888 24.2997
<path d="M81.1615 33.5842L83.1312 24.2997H61.4119L59.4421 33.5842H81.1615ZM42.3769 69.9398L52.4348 22.75
<path d="M128.766 65.2976L104.31 34.3492L99.8652 55.2481H90.5569L99.7955 11.9029L124.217 42.8687L131.311
<path d="M168.614 45.9635L173.216 24.2997H151.497L146.895 45.9635H168.614ZM135.599 55.2481L144.175 14.99
<path d="M216.289 33.5842L218.258 24.2997H196.539L194.569 33.5842H216.289ZM177.504 69.9398L187.562 22.75
<path d="M263.893 65.2976L239.437 34.3492L234.992 55.2481H225.684L234.923 11.9029L259.344 42.8687L266.43
<path d="M306.373 33.5842L308.343 24.2997H286.624L284.654 33.5842H306.373ZM267.589 69.9398L277.647 22.75
</svg>


<svg width="517" height="60" viewBox="0 0 517 60" fill="none" xmlns="http://www.w3.org/2000/svg">
<path d="M7.875 1.75V54.85L4.725 51.775H47.25V58H0.825V1.75H7.875ZM73.6143 58.975C68.6643 58.975 64.2893
<path d="M317.803 1.75V58H303.028V7.3L306.328 7.675L284.578 58H268.828L247.003 7.825L250.378 7.45V58H235
</svg>



[□���Ң
��dzX��zd/����q,��.��Z���P�W�+��R8a'ư���w�m@@@?Ru"���Y|&4?Vr���C□в
B������U\�����2|C@����w���b��,����R'U□4-W;6`z`ˀ���u
hhvX���-B���m����|□k
��p$������□ow��+□f□7��ahɳ3��+{□7þ*��gT i~□N8&X$□X5
⁎⁎⁎

```
name: menyusha
description: menyusha project

publish_to: 'none' # Remove this line if you wish to publish to pub.dev

version: 2.0.0-Banana+10

environment:
  sdk: ">=3.0.0 <4.0.0"

dependencies:
  flutter:
    sdk: flutter
  flutter_svg: ^2.0.10+1

  cupertino_icons: ^1.0.2
  dhttpd: ^4.0.1
  firebase_core: ^2.23.0
  firebase_auth: ^4.19.6
  google_sign_in: ^6.1.6
  google_sign_in_web: ^0.12.4+2
  cloud_firestore: ^4.13.2
  flutter_bloc: ^8.1.3
  get_it: ^7.6.4
  auto_route: ^7.8.4
  font_awesome_flutter: ^10.6.0
  flutter_emoji: ">= 2.0.0"
  carousel_slider: ^4.2.1
  freezed: ^2.4.5
  freezed_annotation: ^2.4.1
  json_annotation: ^4.8.1
  mailto: ^2.0.0
  url_launcher: ^6.2.1
  http: ^1.1.0
  cached_network_image: ^3.3.0
  google_fonts: ^6.2.1
  flutter_staggered_grid_view: ^0.7.0
  flutter_markdown: ^0.6.18+3
  markdown_widget: ^2.3.1
  equatable: ^2.0.5
  timeago: ^3.6.0
  lottie: ^2.7.0
  flutter_quill: ^8.1.1
  dotted_border: ^2.1.0
  intl: ^0.18.1
  glowy_borders: ^1.0.2
  flutter_to_pdf: ^0.1.1
  file_saver: ^0.2.9
  screenshot: ^3.0.0
  pdf: ^3.10.7
  package_info_plus: ^8.0.0
  flutter_dotenv: ^5.1.0
  multi_split_view: ^2.4.0
  markdown: ^7.1.1
  popover: ^0.3.0
  flutter_hooks: ^0.20.5
  uuid: ^4.4.2

dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_lints: ^3.0.1
```

```
  build_runner: ^2.4.6
  auto_route_generator: ^7.3.2
  json_serializable: ^6.7.1
flutter:

  uses-material-design: true
  assets:
    - dotenv.env
    - assets/images/
    - assets/svg/
    - assets/animation/
#  fonts:
#    - family: Bandana
#      fonts:
#        - asset: assets/fonts/bandana.ttf

flutter_web_plugins:
  sdk: flutter
```

onelenyk Project

Overview
onelenykdev is an innovative Flutter project designed to provide a robust and scalable platform for vari

Features
- **Firebase Integration**: Utilize Firebase Auth for user authentication and Firestore for real-time da
- **State Management**: Implemented using Flutter Bloc for efficient state management across the app.
- **Responsive Design**: Crafted with a focus on responsive design, ensuring a smooth user experience ac
- **Advanced UI/UX**: Implementations include custom widgets, theme management, and interactive componen

Code Structure
- `lib/main.dart`: Entry point of the application initializing Firebase and setting up the root componen
- `lib/app/`: Contains the core functionality of the app, including routing, state management, and UI co
- `lib/app/data/`: Data layer of the app, featuring Firebase services and business logic.
- `lib/app/features/`: Modularized app features, each encapsulating specific functionalities.
- `lib/app/common/`: Common utilities and shared widgets for the app.

Setup and Installation
To get started with the OneLenykCo project, ensure you have Flutter installed and set up on your machine

```bash
git clone https://github.com/your-repository/onelenykdev.git
cd onelenykdev
flutter pub get
flutter run
```
Contributing
Contributions to the OneLenykCo project are welcome! Whether it's bug fixes, feature additions, or impro

License
This project is licensed under the MIT License - see the LICENSE.md file for details.

```
{
  "projects": {
    "default": "menyusha-897b9"
  }
}
```