Talks can be clicked on and opened for more details.
Participants can leave comments on talks.
Speakers bio can be viewed.

Workshop program:

1. Flutter introduction
2. Getting up and running
3. Create your first page
4. Flutter Dev tools
5. Your second page with simple navigation
6. Add Localization support
7. Start with State management
8. Add API calls
9. More State management

Assignment - Create a new project:

- File Structure, Platform projects, lib and test
- code analyses
- pubspec.yaml
- Run
- Hot reload
- Widget tree
- Rendering

Coding Session - Create a page:

- HomePage
- AppDevCon image on the top (assets folder)
- List of Talks
- Create Models
    - package json_serializable
    - pub.dev
    - dev dependencies
- StatelessWidget vs Statefull Widgets

Explanation - Flutter DevTools:

- Widget tree
- Performance
- Network tab
- Rendering hints: images, guidelines etc.

Coding Session - Talk Details & Theming:

- Create the feature folder "talk"
- Add page `TalkDetailsPage`
- Navigate to the Talks page like so:

```
Navigator.of(context).push(
        MaterialPageRoute(
          builder: (_) => TalkDetailPage(talk: talk),
        ),
      ),
```

- Create ThemeData, add colors constants and insets constants
- Use the Hero widget to animate the speakers thumbnail to the top of the TalkDetailsPage.

Assignment - Add localization:

- Also see: https://docs.flutter.dev/accessibility-and-localization/internationalization
- Add packages:
  - localizations: `flutter pub add flutter_localizations --sdk=flutter`
  - intl: `flutter pub add intl`
- Add localization delegates

```
return const MaterialApp(
  title: 'Localizations Sample App',
  localizationsDelegates: [
    GlobalMaterialLocalizations.delegate,
    GlobalWidgetsLocalizations.delegate,
    GlobalCupertinoLocalizations.delegate,
  ],
  supportedLocales: [
    Locale('en'), // English
    Locale('nl'), // Dutch
  ],
  home: MyHomePage(),
);
```

- Modify `pubspec.yaml` to enable generation

  ```
  flutter:
    generate: true # Add this line
  ```

- Add l10n.yaml configuration file

  ```
  arb-dir: localizations
  template-arb-file: app_en.arb
  synthetic-package: false
  output-dir: lib/generated
  output-localization-file: app_localizationsdart
  ```

- Create arb files in a new `localizations` folder: `app_en.arb` and `app_nl.arb`
- Run the command `flutter gen-l10n`
- Use the generated code in your app to localize texts

Coding Session - State Management:

- `flutter pub add provider`
- Create a `LocalizationModel` that inherits from `ChangeNotifier`
- Provide the `LocalizationModel` to the widget tree with `ChangeNotifierProvider`
- Consume the `LocalizationModel` in your MaterialApp widget
- Create an HomeAppBar Action button that switches the locale

Assignment - API integration:

- `flutter pub add http`
- Change the TalksService to implement dart http
- Construct the url like this:

```
final url = Uri.https(
      'wyjxbjikucgmxpgozvmi.supabase.co',
      '/rest/v1/talks',
      {'select': 'id,title,abstract,start_time,end_time,speakers(id,name,subtitle,image_url),stages(id,name)'},
    );

final response = await http.get(url, headers: DevApiConstants.defaultHeaders);
```

Assignment - State Management for Talks:

- Implement State management for Talks using e.g. `TalksModel`
- The TalksModel should have a `selectTalk(int talkId)` action that sets a `selectedTalk` property

Assignment - Navigation:

- See: https://pub.dev/documentation/go_router/latest/topics/Get started-topic.html
- Add go_router: `flutter pub add go_router`
- Create a router and add it to your app widget:

```
final _router = GoRouter(
  routes: [
    GoRoute(
      path: '/',
      builder: (context, state) => const HomePage(),
      routes: [
        GoRoute(
          path: 'talks/:talkId',
          builder: (context, state) => TalkDetailPage(talkId: int.parse(state.pathParameters['talkId'] ?? '-1')),
        ),
      ],
    ),
  ],
);

return MaterialApp.router(
  title: 'Dev Events',
  theme: AppTheme.defaultTheme(context),
  localizationsDelegates: AppLocalizations.localizationsDelegates,
  supportedLocales: AppLocalizations.supportedLocales,
  locale: localization.locale,
  routerConfig: _router,
);
```

Assignment - Add web support:

- In your project folder run: `flutter create --platforms web .`
- Run your app in your browser

Assignment Create a comment section:

- On the TalksDetailPage create a comment section.
- Comments for a talk can be retrieved by: `https://wyjxbjikucgmxpgozvmi.supabase.co/rest/v1/comments?talk_id=eq.${talk.id}`
- Create a comment form with an email address field and a comment field

Assignment - Integrate supabase for authentication:

- `flutter pub add supabase_flutter`
- Ask me for the api details
- Create an AuthModel to hold the auth state and business logic.
- Use the supabase package to login with oAuth provider Github.
- Use the `session.accessToken` to post comments to the backend:

```
curl --request POST \
  --url https://wyjxbjikucgmxpgozvmi.supabase.co/rest/v1/comments \
  --header 'Authorization: Bearer {ACCESS_TOKEN}' \
  --header 'Content-Type: application/json' \
  --header 'apikey: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6Ind5anhiamlrdWNnbXhwZ296dm1pIiwicm9sZSI6ImFub24iLCJpYXQiOjE2ODMzNzM0NDQsImV4cCI6MTk5ODk0OTQ0NH0.2YLZM998QBNiWVWsWbA86vae7dyJskA5OWoELhUlKJs' \
  --data '{
        "talk_id": 1,
        "content": "Test comment"
}'
```