|  |  |  |
| --- | --- | --- |
| A picture containing pinwheel, table, lamp, vase  Description automatically generated | | A picture containing text  Description automatically generated |
| PROJECTREPORT | | |
| CARDIFF MET28/10/2020MARTIJN MEELDIJK |

## Testing

The following pages contain the test log for the test ran on the iValueU project.

# TEST CASES iValueU

Martijn Meeldijk

|  |  |
| --- | --- |
| Owner: | Martijn Meeldijk |
| Scope: | To validate all functions of the iValueU app |
| Originator: |  |
| Status: | Active |
| Document ID: |  |
| Location: | Cardiff, Wales |

**Change History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Issue** | **Date** | **Handled by** | **Comments** |
| Created.  Requirements fulfilled  Tested | 10-10-2020  27-10-2020  28-10-2020 | Martijn  Martijn  Martijn |  |

### Table of Contents

[Table of Contents 4](#_Toc54438747)

[1. Introduction 5](#_Toc54438748)

[2. Scope 5](#_Toc54438749)

[3. Requirements 5](#_Toc54438750)

[4. Related Documents 5](#_Toc54438751)

[5. Traceability Matrix 6](#_Toc54438752)

[6. Notes 6](#_Toc54438753)

[7. Test Cases - iValueU 7](#_Toc54438754)

[7.1 Test Item: Select Physical Movie 8](#_Toc54438755)

[7.2 Test Item: Select Emotional Movie 9](#_Toc54438756)

[7.3 Test Item: Start App 11](#_Toc54438757)

[7.4 Test Item: View Photo Diary 12](#_Toc54438758)

[7.5 Test Item: Add Photo 13](#_Toc54438759)

[7.6 Test Item: Bibliotherapy 15](#_Toc54438760)

[7.7 Test Item: Organisations 16](#_Toc54438761)

[7.8 Test Item: Register 17](#_Toc54438762)

# Introduction

The purpose of this test case is to test and validate the functionality of the iValueU features.

# Scope

To validate all known functions of the iValueU features.

# Requirements

1. *Home Screen*
   1. *Display IvalueU logo*
   2. *Create account*
      1. *Create an account given a ‘first name’, ‘last name’, ‘date of birth’, ‘password’, ‘theme’ and ‘language’*
2. *Physical wellbeing*
   1. *Selection between indoors & outdoors*
   2. *List of videos*
   3. *Video player*
3. *Emotional Wellbeing*
   1. *Selection between indoors & outdoors*
   2. *Nested list of videos per category*
   3. *Video player*
4. *Photo Diary*
   1. *List of Photo Diary entries*
   2. *Ability to add entries given a ‘title’, ‘description’ and a photo*
      1. *Title and Description are mandatory*
5. *More*
   1. *Bibliotherapy*
      1. *Info about the origin of the app*
   2. *Organisations*
      1. *Info about participating organisations*

# Related Documents

See diagrams on previous slides

# Traceability Matrix

The following table describes the relation mapping between requirements from the requirement specification documents and corresponding test items, which validate their implementation. REQ – requirement, PRV – previous tests, Bug # - Bug log identification number.

|  |  |
| --- | --- |
| Requirement | Validated by: |
| *REQ - 1*  *REQ - 1.1* | *7.3* |
| *REQ – 1.2*  *REQ – 1.2.1* | *7.8* |
| *REQ - 2*  *REQ - 2.1*  *REQ - 2.2*  *REQ - 2.3* | *7.1* |
| *REQ - 3*  *REQ - 3.1*  *REQ - 3.2*  *REQ - 3.3* | *7.2* |
| *REQ - 4*  *REQ - 4.1*  *REQ - 4.2*  *REQ - 4.2.1* | *7.4*  *7.5* |
| *REQ – 5.1*  *REQ – 5.1.1* | *7.7* |
| *REQ – 5.2*  *REQ – 5.2.1* | *7.8* |
| *BUG# 1* | *7.2* |

# Notes

Add any generally relevant notes here.

Bug# 1: The meditation animation is displayed in both indoors and outdoors emotional menus. This was due to an error in the categorisation algorithm.

# Test Cases *- iValueU*

Follow the checklist in order and perform each test. Check the Pass work Fail box to the left according to the results. Note addition of bug# number if or when a bug is submitted resulting from a failed test

| Test Run # | **Tester Name** | Date | Browser & Version | Execution Time | Did all the Tests Pass? |
| --- | --- | --- | --- | --- | --- |
| 7.1 | Martijn | 28-10-2020 | Flutter 1.22.0  Dart 2.10.0 | A minute | Yes |
| 7.2  BUG# 1 | Martijn | 28-10-2020 | Flutter 1.22.0  Dart 2.10.0 | A minute | No |
| 7.3 | Martijn | 28-10-2020 | Flutter 1.22.0  Dart 2.10.0 | A minute | Yes |
| 7.4 | Martijn | 28-10-2020 | Flutter 1.22.0  Dart 2.10.0 | A minute | Yes |
| 7.5 | Martijn | 28-10-2020 | Flutter 1.22.0  Dart 2.10.0 | A minute | Yes |
| 7.6 | Martijn | 28-10-2020 | Flutter 1.22.0  Dart 2.10.0 | A minute | Yes |
| 7.7 | Martijn | 28-10-2020 | Flutter 1.22.0  Dart 2.10.0 | A minute | Yes |
| 7.8 | Martijn | 28-10-2020 | Flutter 1.22.0  Dart 2.10.0 | A minute | Yes |
| 7.2 | Martijn | 28-10-2020 | Flutter 1.22.0  Dart 2.10.0 | A minute | Yes, bug fixed |

### Test Item: Select Physical Movie

**Scope:** *Test the selection of physical movies*

**Action:** User tap

**Test Notes and Preconditions:**

User has logged in to the system

User has chosen indoor or outdoor

**Verification Steps:** Verify if the navigation through the system is working, thumbnails are displayed, and videos are loaded correctly from YouTube.

**Scenario 1:** *P1*

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.1.1.1 | User opens physical menu | tap | Physical Menu displayed | Physical Menu displayed | P | no |
| 7.1.1.2 | User selects video from menu | tap | Video thumbnail displayed | Video thumbnail displayed | P | no |
| 7.1.1.3 | User taps video thumbnail | tap | Video P1 is displayed | Video P1 is displayed | P | no |

**Scope:** Test the selection of physical movies

**Scenario 2:** *P4*

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.1.2.1 | User opens physical menu | tap | Physical Menu displayed | Physical Menu displayed | P | no |
| 7.1.2.2 | User selects video from menu | tap | Video thumbnail displayed | Video thumbnail displayed | P | no |
| 7.1.2.3 | User taps video thumbnail | tap | Video P4 is displayed | Video P4 is displayed | P | no |

### Test Item: Select Emotional Movie

**Scope:** *Test the selection of emotional movies*

**Action:** User tap

**Test Notes and Preconditions:**

User has logged in to the system

User has chosen indoor or outdoor

**Verification Steps:** Verify if the navigation through the system is working, thumbnails are displayed, and videos are loaded correctly from YouTube.

**Scenario 1:** *E1-1*

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.2.1.1 | User opens emotional menu | tap | Emotional Menu displayed | Emotional Menu displayed | P | no |
| 7.2.1.2 | User selects category from menu | tap | List of videos is displayed | List of videos is displayed | P | no |
| 7.2.1.3 | User selects video from menu | tap | Video thumbnail is displayed | Video thumbnail is displayed | P | no |
| 7.2.1.4 | User taps video thumbnail | tap | Video E1-1 is displayed | Video E1-1 is displayed | P | no |

**Scope:** Test the selection of emotional movies

**Scenario 2:** *E2-4*

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.2.2.1 | User opens emotional menu | tap | Emotional Menu displayed | Emotional Menu displayed | P | no |
| 7.2.2.2 | User selects category from menu | tap | List of videos is displayed | List of videos is displayed | P | no |
| 7.2.2.3 | User selects video from menu | tap | Video thumbnail is displayed | Video thumbnail is displayed | P | no |
| 7.2.2.4 | User taps video thumbnail | tap | Video E2-4 is displayed | Video E2-4 is displayed | P | no |

**Scope:** Test the selection of emotional movies

**Scenario 2:** *Meditation animation*

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.2.2.1 | User opens emotional menu | tap | Emotional Menu displayed | Emotional Menu displayed | P | no |
| 7.2.2.2 | User selects category from menu | tap | List of videos is displayed | Meditation animation is displayed in both indoor and outdoor menus | F | no |
| 7.2.2.3 | User selects video from menu | tap | Video thumbnail is displayed | Video thumbnail is displayed | P | no |
| 7.2.2.4 | User taps video thumbnail | tap | Video E2-4 is displayed | Video E2-4 is displayed | P | no |

### Test Item: Start App

**Scope:** Start the app and see if it actually runs

**Action:** User tap

**Test Notes and Preconditions:** n/a

**Verification Steps:** Verify that the app started correctly

**Scenario 1:**

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.3.1.1 | Start app | tap | App started correctly | App started correctly | P | no |

### Test Item: View Photo Diary

**Scope:** *Test if photos, titles and descriptions are displayed correctly in the photo diary*

**Action:** User tap

**Test Notes and Preconditions:**

User has logged in to the system

User has entries in Photo Diary

**Verification Steps:** Verify if entries in Photo Diary are displayed correctly

**Scenario 1:**

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.4.1.1 | User opens Photo Diary Menu | tap | Photo Diary is displayed | Photo Diary is displayed | P | no |

### Test Item: Add Photo

**Scope:** Test if an entry is correctly added to the photo diary

**Action:** User tap and text input

**Test Notes and Preconditions:**

User is logged in

**Verification Steps:** Verify if photos are correctly added to the diary, and exceptions are caught.

*Little side note: The requirements say that the actual photo is not a mandatory field in the photo diary. So, we should be able to add an entry without adding a photo.*

**Scenario 1:**

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.5.1.1 | User opens Photo Diary menu | tap | Photo diary is opened | Photo diary is opened | P | no |
| 7.5.1.2 | User opens the ‘Add New data’ menu | tap (on the +) | Add new data menu is opened | Add new data menu is opened | P | no |
| 7.5.1.3 | User fills in the title | String: “Test 1” | n/a | n/a | P | no |
| 7.5.1.4 | User fills in the description | String: “Test 1” | n/a | n/a | P | no |
| 7.5.1.5 | User selects photo to add | Random photo | n/a | n/a | P | no |
| 7.5.1.6 | User presses ‘save’ | tap | Entry is saved and can be viewed | Entry is saved and can be viewed | P | no |

**Scope:** Add photo without filling in ‘Title’

**Scenario 2:**

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.5.2.1 | User opens Photo Diary menu | tap | Photo diary is opened | Photo diary is opened | P | no |
| 7.5.2.2 | User opens the ‘Add New data’ menu | tap (on the +) | Add new data menu is opened | Add new data menu is opened | P | no |
| 7.5.2.3 | User fills in the description | String: “Test 1” | n/a | n/a | P | no |
| 7.5.2.4 | User selects photo to add | Random photo | n/a | n/a | P | no |
| 7.5.2.5 | User presses ‘save’ | tap | Exception ‘no title’ is thrown | Exception ‘Title and description are mandatory is thrown | P | no |

**Scope:** Add photo without filling in ‘description’

**Scenario 3:**

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.5.3.1 | User opens Photo Diary menu | tap | Photo diary is opened | Photo diary is opened | P | no |
| 7.5.3.2 | User opens the ‘Add New data’ menu | tap (on the +) | Add new data menu is opened | Add new data menu is opened | P | no |
| 7.5.3.3 | User fills in the title | String: “Test 1” | n/a | n/a | P | no |
| 7.5.3.4 | User selects photo to add | Random photo | n/a | n/a | P | no |
| 7.5.3.5 | User presses ‘save’ | tap | Exception ‘no description’ is thrown | Exception ‘Title and description are mandatory is thrown | P | no |

### Test Item: Bibliotherapy

**Scope:** Test if the info in the bibliotherapy page is displayed correctly and the navigation to said page works as expected

**Action:** User tap

**Test Notes and Preconditions:**

User is logged in

**Verification Steps:** Verify if the bibliotherapy page is displayed correctly and the navigation to said page works as expected

**Scenario 1:**

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.6.1.1 | User opens the ‘more’ menu | tap | ‘more’ menu is displayed | ‘more’ menu is displayed | P | no |
| 7.6.1.2 | User selects ‘bibliotherapy’ | tap | ‘Bibliotherapy’ page is displayed correctly | ‘Bibliotherapy’ page is displayed correctly | P | no |

### Test Item: Organisations

**Scope:** Test if the organisations page works

**Action:** User Tap

**Test Notes and Preconditions:** user is logged in

**Verification Steps:** Verify if the organisations page is displayed correctly and the images on the page are displayed

**Scenario 1:**

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.7.1.1 | User opens the ‘more’ menu | tap | ‘More’ menu is displayed | ‘More’ menu is displayed | P | no |
| 7.7.1.2 | User selects ‘organisations’ from the list | tap | The ‘organisations’ page is displayed correctly with the right images | The ‘organisations’ page is displayed correctly with the right images | P | no |

### Test Item: Register

**Scope:** Test if user registration works as intended

**Action:** User Tap & Text input

**Test Notes and Preconditions:** user is not logged in, app is opened for the first time

**Verification Steps:** Verify if the registrations process works as intended and errors are caught.

**Scenario 1:**

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.8.1.1 | User opens the app | tap | App home screen is displayed with a pop-up asking for credentials | App home screen is displayed with a pop-up asking for credentials | P | no |
| 7.8.1.2 | User fills in ‘first name’ | text |  |  | P | no |
| 7.8.1.3 | User fills in ‘last name’ | text |  |  | P | no |
| 7.8.1.4 | User fills in ‘date of birth’ | text |  |  | P | no |
| 7.8.1.5 | User fills in ‘password’ | text |  |  | P | no |
| 7.8.1.6 | User selects ‘language’ | tap |  |  | P | no |
| 7.8.1.7 | User selects ‘theme’ | tap |  |  | P | no |
| 7.8.1.8 | User presses ‘confirm’ | tap | The User is now successfully registered, and the home screen is displayed with the correct language and theme | The User is now successfully registered, and the home screen is displayed with the correct language and theme | P | no |

**Scenario 2:**

**Scope:** Fill in form without first name

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.8.2.1 | User opens the app | tap | App home screen is displayed with a pop-up asking for credentials | App home screen is displayed with a pop-up asking for credentials | P | no |
| 7.8.2.2 | User fills in ‘last name’ | text |  |  | P | no |
| 7.8.2.3 | User fills in ‘date of birth’ | text |  |  | P | no |
| 7.8.2.4 | User fills in ‘password’ | text |  |  | P | no |
| 7.8.2.5 | User selects ‘language’ | tap |  |  | P | no |
| 7.8.2.6 | User selects ‘theme’ | tap |  |  | P | no |
| 7.8.2.7 | User presses ‘confirm’ | tap | Error is thrown | Error is thrown | P | no |

**Scenario 3:**

**Scope:** Fill in form without last name

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.8.3.1 | User opens the app | tap | App home screen is displayed with a pop-up asking for credentials | App home screen is displayed with a pop-up asking for credentials | P | no |
| 7.8.3.2 | User fills in ‘first name’ | text |  |  | P | no |
| 7.8.3.3 | User fills in ‘date of birth’ | text |  |  | P | no |
| 7.8.3.4 | User fills in ‘password’ | text |  |  | P | no |
| 7.8.3.5 | User selects ‘language’ | tap |  |  | P | no |
| 7.8.3.6 | User selects ‘theme’ | tap |  |  | P | no |
| 7.8.3.7 | User presses ‘confirm’ | tap | Error is thrown | Error is thrown | P | no |

**Scenario 4:**

**Scope:** Fill in form without date of birth

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.8.4.1 | User opens the app | tap | App home screen is displayed with a pop-up asking for credentials | App home screen is displayed with a pop-up asking for credentials | P | no |
| 7.8.4.2 | User fills in ‘first name’ | text |  |  | P | no |
| 7.8.4.3 | User fills in ‘last name’ | text |  |  | P | no |
| 7.8.4.4 | User fills in ‘password’ | text |  |  | P | no |
| 7.8.4.5 | User selects ‘language’ | tap |  |  | P | no |
| 7.8.4.6 | User selects ‘theme’ | tap |  |  | P | no |
| 7.8.4.7 | User presses ‘confirm’ | tap | Error is thrown | Error is thrown | P | no |

**Scenario 5:**

**Scope:** Fill in form without password

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.8.5.1 | User opens the app | tap | App home screen is displayed with a pop-up asking for credentials | App home screen is displayed with a pop-up asking for credentials | P | no |
| 7.8.5.2 | User fills in ‘first name’ | text |  |  | P | no |
| 7.8.5.3 | User fills in ‘last name’ | text |  |  | P | no |
| 7.8.5.4 | User fills in ‘date of birth’ | text |  |  | P | no |
| 7.8.5.5 | User selects ‘language’ | tap |  |  | P | no |
| 7.8.5.6 | User selects ‘theme’ | tap |  |  | P | no |
| 7.8.5.7 | User presses ‘confirm’ | tap | Error is thrown | Error is thrown | P | no |

**Scenario 6:**

**Scope:** Fill in form without language

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.8.6.1 | User opens the app | tap | App home screen is displayed with a pop-up asking for credentials | App home screen is displayed with a pop-up asking for credentials | P | no |
| 7.8.6.2 | User fills in ‘first name’ | text |  |  | P | no |
| 7.8.6.3 | User fills in ‘last name’ | text |  |  | P | no |
| 7.8.6.4 | User fills in ‘date of birth’ | text |  |  | P | no |
| 7.8.6.5 | User fills in ‘password’ | text |  |  | P | no |
| 7.8.6.6 | User selects ‘theme’ | tap |  |  | P | no |
| 7.8.6.7 | User presses ‘confirm’ | tap | Error is thrown | Error is thrown | P | no |

**Scenario 7:**

**Scope:** Fill in form without theme

| Test ID# | Action | Input Value | Expected Results | Actual Results | P/F | Automated |
| --- | --- | --- | --- | --- | --- | --- |
| 7.8.7.1 | User opens the app | tap | App home screen is displayed with a pop-up asking for credentials | App home screen is displayed with a pop-up asking for credentials | P | no |
| 7.8.7.2 | User fills in ‘first name’ | text |  |  | P | no |
| 7.8.7.3 | User fills in ‘last name’ | text |  |  | P | no |
| 7.8.7.4 | User fills in ‘date of birth’ | text |  |  | P | no |
| 7.8.7.5 | User fills in ‘password’ | text |  |  | P | no |
| 7.8.7.6 | User selects ‘language’ | tap |  |  | P | no |
| 7.8.7.7 | User presses ‘confirm’ | tap | Error is thrown | Error is thrown | P | no |

## User Manual

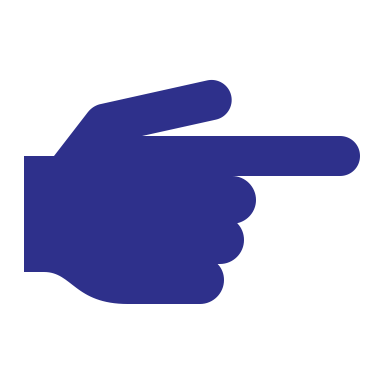
Welcome to iValueU. In this user manual, we’ll cover all the functionality contained in the iValueU app.

Topics we’ll cover:

#### CompassNavigation



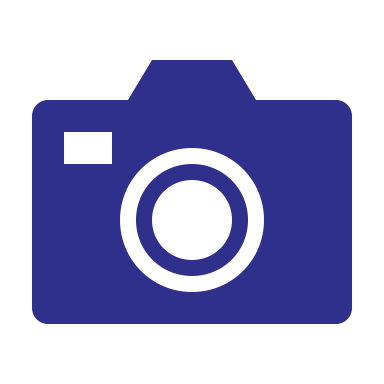
#### Logging in



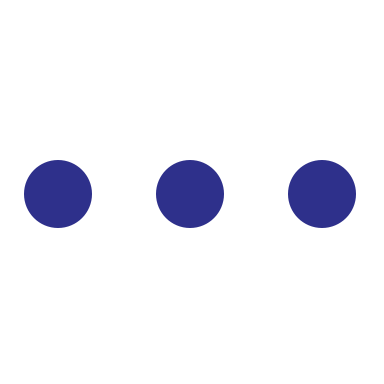
#### Video Selection



#### Video Player



#### Photo Diary



#### More

## Navigation

Graphical user interface, text, application

Description automatically generatedAt the bottom of the app screen, a navigation bar is provided. It contains five categories: iValueU, Physical, Emotional, Photo Diary and More. Tap the icons to navigate to the corresponding page. The icons will turn blue to indicate to which page you have navigated.

## Logging in

Graphical user interface, text, application

Description automatically generatedGraphical user interface, application

Description automatically generatedA screenshot of a cell phone

Description automatically generatedTo log in to the app, tap the ‘register’ button. Upon entering the registration menu, you’ll be asked to input your name, last name, a password, a theme, a language and your birth date. Tap the ‘register’ button at the bottom of the form to complete the registration. You are now successfully logged in.

If you forgot to fill in any of the fields, you’ll be prompted with an error message. Check if you didn’t forget something and try again.

Graphical user interface, application

Description automatically generated

## Video Selection

Both the Emotional and Physical icons on the navigation bar contain videos to aid you in your quest to peace of mind. Tap the maple leaf or the grey box to get started. You’ll be prompted to select whether you’d like to perform indoor or outdoor activities.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

After selecting if you’d like to perform indoor or outdoor activities, you can select a video. In the emotional section, the videos are grouped by different categories. In the physical section, you’ll be taken straight to the list of videos, as there aren’t that many. Tap a video to continue.

## A screenshot of a cell phone Description automatically generatedGraphical user interface, application Description automatically generatedVideo Player

A screenshot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generatedThe video player will play the selected video. Tap the video to show or hide the video player controls.

## Photo Diary

Use the photo diary to document your happiest memories and experiences. The photo diary can save a photo with a title and a description. Tap the “+” button in the top right corner to get started.

You’ll be greeted by a form in with you have to fill in the title and description of your photo. Afterwards, you can add a photo from your phone gallery or take a new one. Press the save button to save your happy memory!

Graphical user interface, application

Description automatically generatedGraphical user interface, text, application, Teams

Description automatically generated

If you forgot to fill in the title or description fields, the app will remind you to fill them in. A photo is not mandatory though, so you can also add entries without photos.

Graphical user interface, application

Description automatically generatedIf you did everything right, your entry will be saved in the photo diary.

A screenshot of a cell phone

Description automatically generated

## More

In the more category, you’ll find two subcategories: bibliotherapy and organisations. In the organisations tab, you can find information on the oranisations that participated in the creation of the app. In the bibliotherapy, you can find more info on Dan, the genius behind the app.

A screenshot of a cell phone

Description automatically generatedGraphical user interface, text, application

Description automatically generated

|  |
| --- |
|  |
| A screen shot of an open computer sitting on top of a table  Description automatically generated |
| THE CODE In the following appendix, you’ll find the source code for the project. |

#### Domain code

model/photo\_diary\_entry.dart

import 'dart:io';  
  
  
*/// One entry in the Photo Diary*class PhotoDiaryEntry{  
 PhotoDiaryEntry({  
 this.title,  
 this.description,  
 this.image  
 });  
 */// The title of the photo* String title;  
 */// The description of the photo* String description;  
 */// A [File] containing the image linked with the entry* File image;  
  
  
}

model/photo\_diary\_repository

import 'package:ivalueu\_app/model/photo\_diary\_entry.dart';  
  
  
*/// A singleton class emulating a database to store the entries of type [PhotoDiaryEntry]*class PhotoDiaryRepository {  
 static final PhotoDiaryRepository *\_singleton* = PhotoDiaryRepository.\_internal();  
  
 factory PhotoDiaryRepository() {  
 return *\_singleton*;  
 }  
 PhotoDiaryRepository.\_internal();  
  
 static var *\_allGroups* = <PhotoDiaryEntry>[  
  
  
 ];  
 */// Returns a [List] of [PhotoDiaryEntry] containing all entries in the diary* static List<PhotoDiaryEntry> *loadEntries*() {  
 return *\_allGroups*.toList();  
 }  
  
 */// Adds an entry of type [PhotoDiaryEntry] to the repository* static bool *addEntry*(PhotoDiaryEntry entry){  
 if(entry.title != null && entry.description!= null){  
 *\_allGroups*.add(entry);  
 return true;  
 }  
 return false;  
  
 }  
}

model/user.dart

*/// This class describes the user of our application*class User{  
 User({  
 this.firstName,  
 this.lastName,  
 this.dob,  
 this.password,  
 this.theme,  
 this.language,  
 });  
 */// First Name of the User* String firstName;  
 */// Last Name of the User* String lastName;  
 */// Users date of birth* DateTime dob;  
 */// Users password* String password;  
 */// Users theme of type enum [UserTheme]* UserTheme theme;  
 */// Users language of type enum [UserLanguage]* UserLanguage language;  
  
  
  
}  
  
enum UserTheme{  
 light,  
 dark  
}  
  
enum UserLanguage{  
 english,  
 welsh  
}

model/user\_repository.dart

import 'package:ivalueu\_app/model/user.dart';  
  
*/// A singleton class simulating a db to store a single [User].*class UserRepository {  
 static final UserRepository *\_singleton* = UserRepository.\_internal();  
  
 factory UserRepository() {  
 return *\_singleton*;  
 }  
 UserRepository.\_internal();  
  
 static User *user*;  
  
 */// Returns the active [User]  
 /// Returns null if the user is not logged in* static User *getUser*() {  
 return *user*;  
 }  
  
 */// Takes a [User] and sets it as the active user* static void *setUser*(User u){  
 *user* = u;  
 }  
}

model/video.dart

import 'package:flutter/foundation.dart';  
import 'package:ivalueu\_app/model/video\_repository.dart';  
  
*/// Category of the videos*enum VideoCategory {  
 physical,  
 emotional1,  
 emotional2,  
 emotional3,  
 emotional4,  
 meditation  
}  
  
*/// Location of the videos*enum Location {  
 indoors,  
 outdoors  
}  
  
*/// This class groups the sets of videos used in the emotional menu*class EmotionalGroup{  
 const EmotionalGroup({  
 @required this.name,  
 @required this.category,  
 @required this.icon  
});  
 */// Name of the grouping, has type [String]* final String name;  
 */// Category of the group of [Video]s this is later used to get the right set from the [VideosRepository]* final VideoCategory category;  
 */// [String] name of the asset used for displaying the group* final String icon;  
  
 */// Returns a [String] which contains the path to the asset for this group* String get iconAsset => 'assets/images/' + '$icon.png';  
  
}  
  
class Video {  
 const Video({  
 @required this.name,  
 @required this.category,  
 @required this.location,  
 @required this.url,  
 @required this.id,  
 @required this.icon,  
  
 }) : assert(name != null);  
 */// Name of the [Video] of type [String]* final String name;  
 */// Category of type enum [VideoCategory]* final VideoCategory category;  
 */// Location of type enum [Location]* final Location location;  
 */// Youtube link of type [String]* final String url;  
 */// Asset name of the thumbnail of type [String]* final String id;  
 */// Asset name of the icon associated with the video of type [String]* final String icon;  
  
 String get iconAsset => 'assets/images/$icon.png';  
 String get thumbnail => 'assets/videos/$id.png';  
 String get youtubeId => url.replaceAll('https://youtu.be/', '');  
  
 @override  
 String toString() => '$name (id=$name)';  
}

model/videos\_repository

import 'video.dart';  
  
  
*/// List of hardcoded videos  
///  
/// Could be replaced with a db in the future, but it's not in the scope of this assignment.*class VideosRepository {  
 static const *\_allVideos* = <Video>[  
 Video(  
 name: "Dilys and Lauren",  
 category: VideoCategory.emotional1,  
 location: Location.indoors,  
 url: "https://youtu.be/KB6CYtpNNEc",  
 id: "E1-1",  
 icon: "Ysgol Gynradd logo"  
 ),  
 Video(  
 name: "Megan and Alex",  
 category: VideoCategory.emotional1,  
 location: Location.indoors,  
 url: "https://youtu.be/7mn9U6pQ8l4",  
 id: "E1-2",  
 icon: "Ysgol Gynradd logo"),  
 Video(  
 name: "Jayden, Iestyn and Deacon",  
 category: VideoCategory.emotional1,  
 location: Location.indoors,  
 url: "https://youtu.be/YDxchDv6LI8",  
 id: "E1-3",  
 icon: "Ysgol Gynradd logo"),  
 Video(  
 name: "Cianna and Macy",  
 category: VideoCategory.emotional1,  
 location: Location.indoors,  
 url: "https://youtu.be/75Zt4H4S9ks",  
 id: "E1-4",  
 icon: "Ysgol Gynradd logo"),  
 Video(  
 name: "Theo and Evie",  
 category: VideoCategory.emotional2,  
 location: Location.indoors,  
 url: "https://youtu.be/qxb6EMwq8UQ",  
 id: "E2-1",  
 icon: "Maes yr haul logo"),  
 Video(  
 name: "Jasmine and Maddison",  
 category: VideoCategory.emotional2,  
 location: Location.indoors,  
 url: "https://youtu.be/-iLTNl8REIs",  
 id: "E2-2",  
 icon: "Maes yr haul logo"),  
 Video(  
 name: "Sophie and Phoebe",  
 category: VideoCategory.emotional2,  
 location: Location.indoors,  
 url: "https://youtu.be/ZaCtsmuZVA4",  
 id: "E2-3",  
 icon: "Maes yr haul logo"),  
 Video(  
 name: "Harry, Calla and Ashton",  
 category: VideoCategory.emotional2,  
 location: Location.outdoors,  
 url: "https://youtu.be/xWMDRBVzxYQ",  
 id: "E2-4",  
 icon: "Maes yr haul logo"),  
 Video(  
 name: "Ellie and Harley",  
 category: VideoCategory.emotional3,  
 location: Location.outdoors,  
 url: "https://youtu.be/888eyVvQY50",  
 id: "E3-1",  
 icon: "Deighton school logo"),  
 Video(  
 name: "Isabelle and Layla",  
 category: VideoCategory.emotional3,  
 location: Location.outdoors,  
 url: "https://youtu.be/xyklkqbopDY",  
 id: "E3-2",  
 icon: "Deighton school logo"),  
 Video(  
 name: "Olivia and Summer",  
 category: VideoCategory.emotional3,  
 location: Location.indoors,  
 url: "https://youtu.be/MQQSBqsBE9c",  
 id: "E3-3",  
 icon: "Deighton school logo"),  
 Video(  
 name: "Kason and Kan",  
 category: VideoCategory.emotional3,  
 location: Location.outdoors,  
 url: "https://youtu.be/IwvtEBI8xik",  
 id: "E3-4",  
 icon: "Deighton school logo"),  
 Video(  
 name: "Gracie and Sarah",  
 category: VideoCategory.emotional3,  
 location: Location.outdoors,  
 url: "https://youtu.be/pd1Gq9PqoGk",  
 id: "E3-5",  
 icon: "Deighton school logo"),  
 Video(  
 name: "Maddie and Nevie",  
 category: VideoCategory.emotional4,  
 location: Location.outdoors,  
 url: "https://youtu.be/QWSWluskpbM",  
 id: "E4-1",  
 icon: "Blaenycwn school logo"),  
 Video(  
 name: "Tyler and Bethany",  
 category: VideoCategory.emotional4,  
 location: Location.indoors,  
 url: "https://youtu.be/04SGYB1lrFE",  
 id: "E4-2",  
 icon: "Blaenycwn school logo"),  
 Video(  
 name: "Duy and Callum",  
 category: VideoCategory.emotional4,  
 location: Location.indoors,  
 url: "https://youtu.be/019r08HyrQo",  
 id: "E4-3",  
 icon: "Blaenycwn school logo"),  
 Video(  
 name: "Grace and Ellie",  
 category: VideoCategory.emotional4,  
 location: Location.outdoors,  
 url: "https://youtu.be/82f0kVf\_maQ",  
 id: "E4-4",  
 icon: "Blaenycwn school logo"),  
 Video(  
 name: "Jessica and Tina",  
 category: VideoCategory.emotional4,  
 location: Location.outdoors,  
 url: "https://youtu.be/Aq3aQwdz8JA",  
 id: "E4-5",  
 icon: "Blaenycwn school logo"),  
  
 Video(  
 name: "HITT Tutorial",  
 category: VideoCategory.physical,  
 location: Location.indoors,  
 url: "https://youtu.be/s7ABZA199rY",  
 id: "P1",  
 icon: "Ysgol Gynradd logo"),  
 Video(  
 name: "Yoga Tutorial 1",  
 category: VideoCategory.physical,  
 location: Location.indoors,  
 url: "https://youtu.be/8wwXmG5rvqE",  
 id: "P2",  
 icon: "Maes yr haul logo"),  
 Video(  
 name: "Yoga Tutorial 2",  
 category: VideoCategory.physical,  
 location: Location.indoors,  
 url: "https://youtu.be/OJasZcwsQGU",  
 id: "P3",  
 icon: "Deighton school logo"),  
 Video(  
 name: "Yoga Tutorial 3",  
 category: VideoCategory.physical,  
 location: Location.outdoors,  
 url: "https://youtu.be/5M9-ZJDr4p8",  
 id: "P4",  
 icon: "Blaenycwn school logo"),  
 ];  
  
 */// Returns a [List] of [Video]  
 ///  
 /// Only returns the videos matching the [VideoCategory] and [Location]* static List<Video> *loadVideos*(VideoCategory category, Location location) {  
 return *\_allVideos*.where((p) => p.category == category && p.location == location).toList();  
 }  
}  
  
*/// This class contains a hardcoded set of categories in which the emotional [Video]s are grouped  
///  
/// Again, this result could be achieved in a more flexible way using an RDBMS system, but this is beyond the scope of this project.*class EmotionalGroupRepository {  
 static const *\_allGroups* = <EmotionalGroup>[  
 EmotionalGroup(  
 name: "Meddwl iach trwy tapio m...", category: VideoCategory.emotional1, icon: "Ysgol Gynradd logo"),  
 EmotionalGroup(  
 name: "Picking out the Positives", category: VideoCategory.emotional2, icon: "Maes yr haul logo"),  
 EmotionalGroup(  
 name: "Deighton Destress Tool", category: VideoCategory.emotional3, icon: "Deighton school logo"),  
 EmotionalGroup(  
 name: "Blaencycwm Breathing Buddies",  
 category: VideoCategory.emotional4,  
 icon: "Blaenycwn school logo"  
 ),  
 EmotionalGroup(  
 name: "Meditation Animation",  
 category: VideoCategory.meditation,  
 icon: "Meditation"  
 ),  
 ];  
  
 */// Returns a [List] of [EmotionalGroup]s (sets of videos)  
 ///  
 /// Each group has a [VideoCategory] which indicates the set of videos to be loaded.* static List<EmotionalGroup> *loadGroups*(Location location) {  
 if(location == Location.indoors){  
 return *\_allGroups*.toList();  
 }  
 return *\_allGroups*.where((p) => p.category != VideoCategory.meditation).toList();  
  
 }  
}

screens/emotional/emotional.dart

import 'package:flutter/cupertino.dart';  
import 'package:ivalueu\_app/screens/emotional/location\_selection\_emotional.dart';  
  
  
*/// This widget contains the main screen of the emotional menu*class Emotional extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: const Text('IValueU'),  
 ),  
 child: SafeArea(  
 child: Padding(  
 padding: const EdgeInsets.all(20),  
 child: Center(  
 child: Column(  
 children: [  
 GestureDetector(  
 onTap: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(builder: (context) => EmotionalLocationSelection()),  
 );  
 },  
 child: Image.asset('assets/images/MapleLeafLarge.png')),  
 GestureDetector(  
 onTap: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(builder: (context) => EmotionalLocationSelection()),  
 );  
 },  
 child: Container(  
 width: 240.0,  
 height: 42.0,  
 decoration: BoxDecoration(  
 color: CupertinoColors.*systemGrey4*,  
 borderRadius: BorderRadius.circular(10),  
 boxShadow: [  
 BoxShadow(  
 color: CupertinoColors.*black*.withOpacity(0.1),  
 spreadRadius: 4,  
 blurRadius: 10),  
 ]),  
 child: Center(  
 child: Text(  
 'Emotional Wellbeing',  
 style: TextStyle(  
 fontSize: 18,  
 color: CupertinoColors.*activeBlue*,  
 height: 1,  
 ),  
 textAlign: TextAlign.center,  
 ),  
 ),  
 ),  
 ),  
 Padding(  
 padding: const EdgeInsets.all(40),  
 child: SizedBox(  
 width: 240,  
 child: Text(  
 "Use the Emotional Wellbeing self help tools including Mediation Animation made by the pupils with Sean Harris"),  
 ),  
 )  
 ],  
 )),  
 ),  
 ),  
 );  
 }  
}  
  
// user defined function

screens/emotional/emotional\_list.dart

import 'package:flutter/cupertino.dart';  
import 'package:flutter/foundation.dart';  
import 'package:ivalueu\_app/model/video.dart';  
import 'package:ivalueu\_app/model/video\_repository.dart';  
  
import 'emotional\_row\_item.dart';  
  
*/// This widget contains the list of groups of [Video]s from the emotional [VideoCategory]*class EmotionalList extends StatelessWidget {  
 const EmotionalList({  
 Key key,  
 this.location  
 }) : super(key: key);  
  
 final Location location;  
  
 @override  
 Widget build(BuildContext context) {  
 final groups = EmotionalGroupRepository.*loadGroups*(location);  
 return CustomScrollView(  
 semanticChildCount: groups.length,  
 slivers: <Widget>[  
 const CupertinoSliverNavigationBar(  
 largeTitle: Text('Emotional Wellbeing'),  
 ),  
 SliverSafeArea(  
 top: false,  
 minimum: const EdgeInsets.only(top: 8),  
 sliver: SliverList(  
 delegate: SliverChildBuilderDelegate(  
 (context, index) {  
 if (index < groups.length) {  
 return EmotionalRowItem(  
 index: index,  
 group: groups[index],  
 lastItem: index == groups.length - 1,  
 location: location,  
 );  
 }  
  
 return null;  
 },  
 ),  
   
 ),  
 )  
 ],  
 );  
 }  
}

screens/emotional/emotional\_row\_item.dart

import 'package:flutter/cupertino.dart';  
import 'package:ivalueu\_app/screens/emotional/meditation\_animation.dart';  
import 'package:ivalueu\_app/screens/physical/video\_list.dart';  
  
import 'package:ivalueu\_app/model/video.dart';  
import 'package:ivalueu\_app/Styles.dart';  
  
*/// This widget is rendered for each item in the list of emotional [Video] groups.*class EmotionalRowItem extends StatelessWidget {  
 const EmotionalRowItem(  
 {this.index, this.group, this.lastItem, this.location});  
  
 final EmotionalGroup group;  
 final int index;  
 final bool lastItem;  
 final Location location;  
  
 @override  
 Widget build(BuildContext context) {  
 final row = SafeArea(  
 top: false,  
 bottom: false,  
 minimum: const EdgeInsets.only(  
 left: 16,  
 top: 8,  
 bottom: 8,  
 right: 8,  
 ),  
 child: GestureDetector(  
 onTap: () {  
 if (group.category == VideoCategory.meditation) {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(builder: (context) => MeditationAnimation()),  
 );  
 } else  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(  
 builder: (context) => VideoList(  
 category: group.category,  
 location: location,  
 )),  
 );  
 },  
 child: SizedBox(  
 height: 110,  
 child: Column(  
 children: <Widget>[  
 ClipRRect(  
 borderRadius: BorderRadius.circular(4),  
 child: Image.asset(  
 group.iconAsset,  
 fit: BoxFit.cover,  
 width: 76,  
 height: 76,  
 ),  
 ),  
 Expanded(  
 child: Padding(  
 padding: const EdgeInsets.symmetric(horizontal: 12),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.start,  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: <Widget>[  
 Text(  
 group.name,  
 style: Styles.*VideoRowItemName*,  
 ),  
 const Padding(padding: EdgeInsets.only(top: 8)),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
  
 if (lastItem) {  
 return row;  
 }  
  
 return Column(  
 children: <Widget>[  
 row,  
 Padding(  
 padding: const EdgeInsets.only(  
 left: 100,  
 right: 16,  
 ),  
 ),  
 ],  
 );  
 }  
}

screens/emotional/location\_selection\_emotional.dart

import 'package:flutter/cupertino.dart';  
import 'package:ivalueu\_app/model/video.dart';  
import 'package:ivalueu\_app/screens/emotional/emotional\_list.dart';  
  
*/// This widget displays the indoor/outdoor selection screen*class EmotionalLocationSelection extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: const Text('Select location'),  
 ),  
 child: SafeArea(  
 child: Padding(  
 padding: const EdgeInsets.all(40),  
 child: Center(  
 child: Column(  
 children: [  
 Padding(  
 padding: const EdgeInsets.all(30),  
 child: Text("Can you perform indoor or outdoor activities?"),  
 ),  
 CupertinoButton(  
 onPressed: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(  
 builder: (context) => EmotionalList(  
 location: Location.indoors,  
 )),  
 );  
 },  
 child: Text(  
 'Indoors',  
 style: TextStyle(  
 color: CupertinoColors.*activeBlue*,  
 ),  
 ),  
 ),  
 CupertinoButton(  
 onPressed: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(  
 builder: (context) => EmotionalList(  
 location: Location.outdoors,  
 )),  
 );  
 },  
 child: Text(  
 'Outdoors',  
 style: TextStyle(  
 color: CupertinoColors.*activeBlue*,  
 ),  
 ),  
 ),  
 ],  
 )),  
 ),  
 ),  
 );  
 }  
}

screens/emotional/meditation\_animation.dart

import 'package:flutter/cupertino.dart';  
import 'package:youtube\_player\_flutter/youtube\_player\_flutter.dart';  
import 'package:flutter/material.dart';  
  
*/// The meditation animation page*class MeditationAnimation extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 YoutubePlayerController \_controller = YoutubePlayerController(  
 initialVideoId: "KB6CYtpNNEc",  
 flags: YoutubePlayerFlags(  
 autoPlay: true,  
 mute: true,  
 ),  
 );  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: Text("Meditation Animation"),  
 ),  
 child: SafeArea(  
 child: Center(  
 child: Column(  
 children: [  
 Padding(  
 padding: const EdgeInsets.all(30),  
 child: Container(  
 height: 200,  
 child: SingleChildScrollView(  
 scrollDirection: Axis.vertical,  
 child: Text(  
 "Preserved in the muddy estuaries of South-east wales are the footprints of cranes; large, raucous wetland birds who made these trails around 7,000 years ago when the Bristol Channel was a marshland connecting Wales to Somerset.Cranes became extinct in Wales around five hundred years ago as a result of habitat loss (the wetlands were drained) and hunting. In other words, as a result of human action.The cranes in this short animation were animated in four primary schools – in celebration of the return of the crane to Wales; a miracle enabled by a combination of science and community action. This shows that when we choose to, we can and do make a positive difference.The soundtrack features cranes and some other amazing birds that haven’t returned, amongst then the White Tailed Eagle, Dalmatian pelican and Great Northern Diver. Also the bittern which, like the crane, came back only more surreptitiously…Do we, as humans, have the sole right to decide which species survive or disappear?")),  
 ),  
 ),  
 YoutubePlayer(  
 controller: \_controller,  
 showVideoProgressIndicator: true,  
 liveUIColor: CupertinoColors.*systemRed*,  
 bottomActions: [  
 CurrentPosition(),  
 ProgressBar(isExpanded: true),  
 ],  
 )  
 ],  
 )),  
 ),  
 );  
 }  
}

screens/home/home.dart

import 'package:flutter/cupertino.dart';  
import 'package:ivalueu\_app/model/user.dart';  
import 'package:ivalueu\_app/model/user\_repository.dart';  
import 'package:ivalueu\_app/screens/home/register.dart';  
  
  
*/// This widget contains the iValueU home page.  
///  
/// It's a stateful widget so that it can refresh its state when the [User] is logged in.*class Home extends StatefulWidget {  
 @override  
 \_HomeState createState() => \_HomeState();  
}  
  
class \_HomeState extends State<Home> {  
 @override  
 Widget build(BuildContext context) {  
 User user = UserRepository.*user*;  
 Widget login;  
 if(user == null){  
 login = GestureDetector(  
 child: Text("Register",  
 style: TextStyle(  
 fontSize: 18,  
 color: CupertinoColors.*activeBlue*,  
 height: 1,  
 ),  
 textAlign: TextAlign.center,),  
 onTap: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(builder: (context) => Register()),  
 ).then((value) {  
 setState(() {  
  
 });  
 });  
 },  
 );  
 }  
 else{  
 login = GestureDetector(  
 child: Text("Log out",  
 style: TextStyle(  
 fontSize: 18,  
 color: CupertinoColors.*activeBlue*,  
 height: 1,  
 ),  
 textAlign: TextAlign.center,),  
 onTap: () {  
 setState(() {  
 UserRepository.*user* = null;  
 user = null;  
 });  
 },  
 );  
 }  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: const Text('IValueU'),  
 trailing: login  
 ),  
 child: SafeArea(  
 child: Padding(  
 padding: const EdgeInsets.all(20),  
 child: Center(  
 child: Column(  
 children: [  
 Image.asset('assets/images/MapleLeafLarge.png'),  
 Container(  
 width: 240.0,  
 height: 42.0,  
 decoration: BoxDecoration(  
 color: CupertinoColors.*systemGrey4*,  
 borderRadius: BorderRadius.circular(10),  
 boxShadow: [  
 BoxShadow(  
 color: CupertinoColors.*black*.withOpacity(0.1),  
 spreadRadius: 4,  
 blurRadius: 10),  
 ]),  
 child: Center(  
 child: Text(  
 'Wellbeing IValueU',  
 style: TextStyle(  
 fontFamily: 'Arial',  
 fontSize: 18,  
 color: CupertinoColors.*black*,  
 height: 1,  
 ),  
 textAlign: TextAlign.center,  
 ),  
 ),  
 ),  
 ],  
 )),  
 ),  
 ),  
 );  
 }  
}

screens/home/register.dart

import 'package:flutter/cupertino.dart';  
import 'package:flutter/material.dart';  
import 'package:ivalueu\_app/model/user.dart';  
import 'package:ivalueu\_app/model/user\_repository.dart';  
  
  
*/// This widget contains the registration form for making a new [User].  
///  
/// Upon creating an account, the [User] is immediately logged in.*class Register extends StatefulWidget {  
 @override  
 \_TestFormState createState() => \_TestFormState();  
}  
  
class \_TestFormState extends State<Register> {  
 final \_formKey = GlobalKey<FormState>();  
 User model = User();  
 final \_firstNameController = TextEditingController();  
 final \_lastNameController = TextEditingController();  
 final \_passwordController = TextEditingController();  
 Widget preview;  
 UserTheme theme;  
 UserLanguage language;  
 DateTime date;  
  
 @override  
 Widget build(BuildContext context) {  
 showThemePicker() {  
 showCupertinoModalPopup(  
 context: context,  
 builder: (BuildContext context) => CupertinoActionSheet(  
 title: const Text('Theme'),  
 message: const Text('Choose theme '),  
 actions: <Widget>[  
 CupertinoActionSheetAction(  
 child: const Text('Dark'),  
 onPressed: () {  
 theme = UserTheme.dark;  
 Navigator.*pop*(context);  
 },  
 ),  
 CupertinoActionSheetAction(  
 child: const Text('Light'),  
 onPressed: () {  
 theme = UserTheme.light;  
 Navigator.*pop*(context);  
 },  
 )  
 ],  
 cancelButton: CupertinoActionSheetAction(  
 child: const Text('Cancel'),  
 isDefaultAction: true,  
 onPressed: () {  
 Navigator.*pop*(context, 'Cancel');  
 },  
 )),  
 );  
 }  
  
 showMyDatePicker() {  
 showModalBottomSheet(  
 context: context,  
 builder: (BuildContext builder) {  
 return Column(  
 children: [  
 Container(  
 height: MediaQuery.*of*(context).copyWith().size.height / 3,  
 child: CupertinoDatePicker(  
 initialDateTime: DateTime(2000, 1, 1),  
 onDateTimeChanged: (DateTime newdate) {  
 date = newdate;  
 },  
 use24hFormat: true,  
 maximumDate: new DateTime(2022, 12, 30),  
 minimumYear: 1900,  
 maximumYear: 2022,  
 minuteInterval: 1,  
 mode: CupertinoDatePickerMode.date,  
 )),  
  
 ],  
  
 );  
 });  
 }  
  
 showLanguagePicker() {  
 showCupertinoModalPopup(  
 context: context,  
 builder: (BuildContext context) => CupertinoActionSheet(  
 title: const Text('Language'),  
 message: const Text('Choose Language '),  
 actions: <Widget>[  
 CupertinoActionSheetAction(  
 child: const Text('English'),  
 onPressed: () {  
 language = UserLanguage.english;  
 Navigator.*pop*(context);  
 },  
 ),  
 CupertinoActionSheetAction(  
 child: const Text('Welsh'),  
 onPressed: () {  
 language = UserLanguage.welsh;  
 Navigator.*pop*(context);  
 },  
 )  
 ],  
 cancelButton: CupertinoActionSheetAction(  
 child: const Text('Cancel'),  
 isDefaultAction: true,  
 onPressed: () {  
 Navigator.*pop*(context, 'Cancel');  
 },  
 )),  
 );  
 }  
  
 return CupertinoPageScaffold(  
 resizeToAvoidBottomInset: true,  
 navigationBar: CupertinoNavigationBar(  
 middle: const Text('Register'),  
 ),  
 child: SafeArea(  
 child: Padding(  
 padding: EdgeInsets.all(30),  
 child: Center(  
 child: Form(  
 key: \_formKey,  
 child: SingleChildScrollView(  
 child: Column(  
 children: <Widget>[  
 Container(  
 alignment: Alignment.*topCenter*,  
 child: MyTextFormField(  
 hintText: 'First Name',  
 controller: \_firstNameController,  
 onSaved: (String value) {  
 model.firstName = value;  
 },  
 ),  
 ),  
 Container(  
 alignment: Alignment.*topCenter*,  
 child: MyTextFormField(  
 hintText: 'Last Name',  
 controller: \_lastNameController,  
 onSaved: (String value) {  
 model.lastName = value;  
 },  
 ),  
 ),  
 Container(  
 alignment: Alignment.*topCenter*,  
 child: MyTextFormField(  
 hintText: 'Password',  
 controller: \_passwordController,  
 onSaved: (String value) {  
 model.password = value;  
 },  
 ),  
 ),  
 Column(  
 children: <Widget>[  
 const SizedBox(height: 15),  
 Text(  
 "Selected theme: ${theme.toString().substring(theme.toString().indexOf('.') + 1)}"),  
 const SizedBox(height: 15),  
 GestureDetector(  
 onTap: showThemePicker,  
 child: Center(  
 child: Text(  
 "Select Theme",  
 style: TextStyle(  
 fontFamily: 'Arial',  
 color: CupertinoColors.*activeBlue*,  
 height: 1,  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 Column(  
 children: <Widget>[  
 const SizedBox(height: 15),  
 Text(  
 "Selected language: ${language.toString().substring(language.toString().indexOf('.') + 1)}"),  
 const SizedBox(height: 15),  
 GestureDetector(  
 onTap: showLanguagePicker,  
 child: Center(  
 child: Text(  
 "Select Language",  
 style: TextStyle(  
 fontFamily: 'Arial',  
 color: CupertinoColors.*activeBlue*,  
 height: 1,  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 Column(  
 children: <Widget>[  
 const SizedBox(height: 15),  
 Text(  
 "Selected date: ${date.toString()}"),  
 const SizedBox(height: 15),  
 GestureDetector(  
 onTap: showMyDatePicker,  
 child: Center(  
 child: Text(  
 "Select Date",  
 style: TextStyle(  
 fontFamily: 'Arial',  
 color: CupertinoColors.*activeBlue*,  
 height: 1,  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 CupertinoButton(  
 onPressed: () async {  
 if (valid(\_firstNameController.text) &&  
 valid(\_lastNameController.text) && language != null && theme != null && date != null && valid(\_passwordController.text)) {  
 /\*if(\_image != null) \*/ /\*final File newImage =  
 await \_image.copy(  
 'assets/images/${\_titleController.text}.png');\*/  
 UserRepository.*setUser*(new User(  
 firstName: \_firstNameController.text,  
 lastName: \_lastNameController.text,  
 theme: theme,  
 language: language,  
 dob: date,  
 password: \_passwordController.text));  
 Navigator.*pop*(context);  
 } else {  
 showCupertinoModalPopup(  
 context: context,  
 builder: (BuildContext context) {  
 // return object of type Dialog  
 return Padding(  
 padding: const EdgeInsets.all(50),  
 child: CupertinoPopupSurface(  
 child: Padding(  
 padding: const EdgeInsets.all(20),  
 child: Text(  
 "All fields need to be filled in!"),  
 ),  
 ),  
 );  
 });  
 }  
 },  
 child: Text(  
 'Register',  
 style: TextStyle(  
 color: CupertinoColors.*activeBlue*,  
 ),  
 ),  
 ),  
 Container(child: preview),  
 ],  
 ),  
 ),  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
}  
  
bool valid(String input) {  
 return input.isNotEmpty;  
}  
  
class MyTextFormField extends StatelessWidget {  
 final String hintText;  
 final TextEditingController controller;  
 final Function onSaved;  
 final bool isPassword;  
 final bool isEmail;  
  
 MyTextFormField({  
 this.hintText,  
 this.controller,  
 this.onSaved,  
 this.isPassword = false,  
 this.isEmail = false,  
 });  
  
 @override  
 Widget build(BuildContext context) {  
 return Padding(  
 padding: EdgeInsets.all(8.0),  
 child: CupertinoTextField(  
 controller: controller,  
 placeholder: hintText,  
 obscureText: isPassword ? true : false,  
 onSubmitted: onSaved,  
 keyboardType: isEmail ? TextInputType.*emailAddress* : TextInputType.*text*,  
 ),  
 );  
 }  
}

screens/more/bibliotherapy.dart

import 'package:flutter/cupertino.dart';  
  
*/// This widget contains the information displayed in the Bibliotherapy page*class Bibliotherapy extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: Text("Bibliotherapy"),  
 ),  
 child: SafeArea(  
 child: Center(  
 child: Column(  
 children: [  
 Padding(  
 padding: const EdgeInsets.all(20),  
 child: Center(  
 child: SizedBox(  
 width: 200,  
 child: Image.asset('assets/images/dan anthony.jpg'),  
 ),  
 ),  
 ),  
 Padding(  
 padding: const EdgeInsets.all(30),  
 child: Container(  
 height: 250,  
 child: SingleChildScrollView(  
 scrollDirection: Axis.vertical,  
 child: Text(  
 "Hi my name is Dan, I am an author and I teach creative writing at Cardiff metropolitan University. I specialise in writing fiction for children. It was a pleasure to visit Deighton primary, Blaen y Cwm, Maes yr Haul and Calon y Cwmoedd. I tried to offer each school workshop that chimed with digital aspirations of the app and the well-being agenda implicit in the idea of bibliotherapy.In the mornings I used my own books as springboards into reading and writing in single-class workshops. My aim was to establish a rapport with young readers, emphasising our common ground: writes don’t know that publisher or readers will like their books, just as pupils don’t know how teachers will react to their work. By reading and sharing ideas we built up a culture in which story ideas became more important than ‘mistakes’ in their realisation. By the end of each session pupils had moved on from considering my stories to writing their own, much more interesting tales.During the afternoons, bibliotherapy book-review sessions were offered to pupils throughout the schools. The aim here was to encourage readers of all abilities and ages to get used to discussion their thoughts on work they’d read. As well as fiction we discussed non-fiction.Some books by Dan Anthony – The Rugby Zombies series, Steve’s Dreams series, The Bus Stop at the End of the World.")),  
 ),  
 ),  
  
 ],  
 )),  
 ),  
 );  
 }  
}

screens/more/more.dart

import 'package:flutter/cupertino.dart';  
import 'package:flutter/material.dart';  
import 'package:ivalueu\_app/screens/more/bibliotherapy.dart';  
import 'package:ivalueu\_app/screens/more/organisations.dart';  
  
  
*/// This widget contains the More subpage.  
///  
/// More links can be added by copy/pasting the [GestureDetector]s*class More extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: Text('More'),  
 ),  
 child: SafeArea(  
 child: Material(  
 child: ListView(  
 children: ListTile.*divideTiles*(  
 // <-- ListTile.divideTiles  
 context: context,  
 tiles: [  
 GestureDetector(  
 onTap: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(builder: (context) => Bibliotherapy()),  
 );  
 },  
 child: ListTile(  
 title: Text('Bibliotherapy'),  
 ),  
 ),  
 GestureDetector(  
 onTap: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(builder: (context) => Organisations()),  
 );  
 },  
 child: ListTile(  
 title: Text('Organisations'),  
 ),  
 ),  
 ]).toList(),  
 ),  
 )),  
 );  
 }  
}

screens/more/organisations.dart

import 'package:flutter/cupertino.dart';  
  
*/// This widget contains the information about the participating organisations*class Organisations extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: Text("Organisations"),  
 ),  
 child: SafeArea(  
 child: Center(  
 child: Column(  
 children: [  
 Center(  
 child: SizedBox(  
 child: Image.asset('assets/images/kuleuven.png'),  
 ),  
 ),  
 Center(  
 child: SizedBox(  
 child: Image.asset('assets/images/kuleuven1.png'),  
 ),  
 ),  
  
  
 ],  
 )),  
 ),  
 );  
 }  
}

screens/photo\_diary/form\_field.dart

import 'dart:io';  
  
import 'package:flutter/cupertino.dart';  
import 'package:flutter/material.dart';  
import 'package:image\_picker/image\_picker.dart';  
import 'package:ivalueu\_app/model/photo\_diary\_entry.dart';  
import 'package:ivalueu\_app/model/photo\_diary\_repository.dart';  
  
*/// This form is rendered when adding a new entry to the Photo Diary*class TestForm extends StatefulWidget {  
 @override  
 \_TestFormState createState() => \_TestFormState();  
}  
  
class \_TestFormState extends State<TestForm> {  
 final \_formKey = GlobalKey<FormState>();  
 PhotoDiaryEntry model = PhotoDiaryEntry();  
 final \_titleController = TextEditingController();  
 final \_descriptionController = TextEditingController();  
 File \_image;  
 Widget preview;  
  
 @override  
 Widget build(BuildContext context) {  
 void \_showDialog() {  
 // flutter defined function  
 showDialog(  
 context: context,  
 builder: (BuildContext context) {  
 // return object of type Dialog  
 return CupertinoPopupSurface(  
 child: new Text("Alert Dialog title"),  
  
 );  
 },  
 );  
 }  
 if (\_image != null) {  
 preview = Padding(  
 padding: const EdgeInsets.all(40),  
 child: Column(  
 children: [  
 Text("Selected Image:"),  
 Padding(  
 padding: const EdgeInsets.all(20),  
 child: Container(  
 width: 60,  
 height: 60,  
 decoration: BoxDecoration(  
 color: CupertinoColors.*systemGrey*,  
 shape: BoxShape.circle,  
 image: DecorationImage(  
 image: ResizeImage(FileImage(\_image), width: 200))),  
 )),  
 ],  
 ),  
 );  
 } else  
 preview = Container();  
  
 \_imgFromCamera() async {  
 FocusScope.*of*(context).unfocus();  
 File image = await ImagePicker.*pickImage*(  
 source: ImageSource.camera, imageQuality: 50);  
  
 setState(() {  
 \_image = image;  
 });  
 }  
  
 \_imgFromGallery() async {  
 FocusScope.*of*(context).unfocus();  
 File image = await ImagePicker.*pickImage*(  
 source: ImageSource.gallery, imageQuality: 50);  
  
 setState(() {  
 \_image = image;  
 });  
 }  
  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: const Text('Add Photo'),  
 ),  
 child: SafeArea(  
 child: Padding(  
 padding: EdgeInsets.all(30),  
 child: Center(  
 child: Form(  
 key: \_formKey,  
 child: Column(  
 children: <Widget>[  
 Container(  
 alignment: Alignment.*topCenter*,  
 child: MyTextFormField(  
 hintText: 'Title',  
 controller: \_titleController,  
 onSaved: (String value) {  
 model.title = value;  
 },  
 ),  
 ),  
 Container(  
 alignment: Alignment.*topCenter*,  
 child: MyTextFormField(  
 hintText: 'Description',  
 controller: \_descriptionController,  
 onSaved: (String value) {  
 model.description = value;  
 },  
 ),  
 ),  
 CupertinoButton(  
 onPressed: () {  
 \_imgFromGallery();  
 },  
 child: Text(  
 'Choose Photo',  
 style: TextStyle(  
 color: CupertinoColors.*activeBlue*,  
 ),  
 ),  
 ),  
 CupertinoButton(  
 onPressed: () {  
 \_imgFromCamera();  
 },  
 child: Text(  
 'Camera',  
 style: TextStyle(  
 color: CupertinoColors.*activeBlue*,  
 ),  
 ),  
 ),  
 CupertinoButton(  
 onPressed: () async {  
 if (valid(\_descriptionController.text) &&  
 valid(\_titleController.text)) {  
 /\*if(\_image != null) \*/ /\*final File newImage =  
 await \_image.copy(  
 'assets/images/${\_titleController.text}.png');\*/  
 PhotoDiaryRepository.*addEntry*(new PhotoDiaryEntry(  
 title: \_titleController.text,  
 description: \_descriptionController.text,  
 image: \_image));  
 Navigator.*pop*(context);  
 }  
 else{  
 showCupertinoModalPopup(context: context, builder: (BuildContext context) {  
 // return object of type Dialog  
 return Padding(  
 padding: const EdgeInsets.all(50),  
 child: CupertinoPopupSurface(  
 child: Padding(  
 padding: const EdgeInsets.all(20),  
 child: Text("Title and description are mandatory!"),  
 ),  
  
 ),  
 );  
 });  
 }  
 },  
 child: Text(  
 'Save',  
 style: TextStyle(  
 color: CupertinoColors.*activeBlue*,  
 ),  
 ),  
 ),  
 Container(child: preview),  
 ],  
 ),  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
}  
  
  
bool valid(String input) {  
 return input.isNotEmpty;  
}  
  
class MyTextFormField extends StatelessWidget {  
 final String hintText;  
 final TextEditingController controller;  
 final Function onSaved;  
 final bool isPassword;  
 final bool isEmail;  
  
 MyTextFormField({  
 this.hintText,  
 this.controller,  
 this.onSaved,  
 this.isPassword = false,  
 this.isEmail = false,  
 });  
  
 @override  
 Widget build(BuildContext context) {  
 return Padding(  
 padding: EdgeInsets.all(8.0),  
 child: CupertinoTextField(  
 controller: controller,  
 placeholder: hintText,  
 obscureText: isPassword ? true : false,  
 onSubmitted: onSaved,  
 keyboardType: isEmail ? TextInputType.*emailAddress* : TextInputType.*text*,  
 ),  
 );  
 }  
}

screens/photo\_diary/photo\_diary.dart

import 'package:flutter/cupertino.dart';  
import 'package:flutter/material.dart';  
import 'package:ivalueu\_app/model/photo\_diary\_entry.dart';  
import 'package:ivalueu\_app/screens/photo\_diary/form\_field.dart';  
import 'package:ivalueu\_app/model/photo\_diary\_repository.dart';  
import 'package:ivalueu\_app/screens/photo\_diary/photo\_diary\_row\_item.dart';  
  
*/// This widget displays all the [PhotoDiaryEntry]s in the [PhotoDiaryRepository]  
/// in the form of a reusible widget [PhotoDiaryRowItem]*class PhotoDiary extends StatefulWidget {  
 @override  
 \_PhotoDiaryState createState() => \_PhotoDiaryState();  
}  
  
class \_PhotoDiaryState extends State<PhotoDiary> {  
 @override  
 Widget build(BuildContext context) {  
 var entries = PhotoDiaryRepository.*loadEntries*();  
 if (entries.isEmpty){  
 entries.add(PhotoDiaryEntry(title: "No photos yet", description: "Use the Photo Diary tool to keep a record of all the wellbeing activities you have completed. Tap the + button to begin."));  
 }  
 return CustomScrollView(  
 semanticChildCount: entries.length,  
 slivers: <Widget>[  
 CupertinoSliverNavigationBar(  
 largeTitle: Text('Photo Diary'),  
 trailing: GestureDetector(  
 child: Icon(CupertinoIcons.*plus*),  
 onTap: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(builder: (context) => TestForm()),  
 ).then((value) {setState(() {});});  
 },  
 ),  
 ),  
 SliverSafeArea(  
 // BEGINNING OF NEW CONTENT  
 top: false,  
 minimum: const EdgeInsets.only(top: 8),  
 sliver: SliverList(  
 delegate: SliverChildBuilderDelegate(  
 (context, index) {  
 if (index < entries.length) {  
 return PhotoDiaryRowItem(  
 index: index,  
 entry: entries[index],  
 lastItem: index == entries.length - 1,  
 );  
 }  
  
 return null;  
 },  
 ),  
 ),  
 )  
 ],  
 );  
 }  
}  
  
// user defined function

screens/photo\_diary/photo\_diary\_row\_item.dart

import 'dart:io';  
  
import 'package:flutter/cupertino.dart';  
import 'package:ivalueu\_app/Styles.dart';  
import 'package:ivalueu\_app/model/photo\_diary\_entry.dart';  
import 'package:ivalueu\_app/model/photo\_diary\_repository.dart';  
  
*/// This class contains a single item from the [PhotoDiaryRepository]  
///  
/// By reusing this component, I was able to prevent typing very many lines of code.*class PhotoDiaryRowItem extends StatelessWidget {  
 PhotoDiaryRowItem({  
 this.index,  
 this.entry,  
 this.lastItem,  
 });  
  
 final PhotoDiaryEntry entry;  
 final int index;  
 final bool lastItem;  
  
 @override  
 Widget build(BuildContext context) {  
 File image;  
 image = entry.image;  
  
 Widget imageDisplay;  
 if(entry.image != null){  
 imageDisplay = Container(  
 width: 60,  
 height: 60,  
 decoration: BoxDecoration(  
 color: CupertinoColors.*systemGrey*,  
 shape: BoxShape.circle,  
 image: DecorationImage(  
 image: ResizeImage(FileImage(image), width: 200))),  
 );  
  
 }  
 else{  
 imageDisplay = Container(  
 width: 60,  
 height: 60,  
 decoration: BoxDecoration(  
 color: CupertinoColors.*systemGrey*,  
 shape: BoxShape.circle,  
  
 ));  
 }  
 final row = SafeArea(  
 top: false,  
 bottom: false,  
 minimum: const EdgeInsets.only(  
 left: 16,  
 top: 8,  
 bottom: 8,  
 right: 8,  
 ),  
 child: Row(  
 children: <Widget>[  
 Container(  
 child: imageDisplay,  
 ),  
 Expanded(  
 child: Padding(  
 padding: const EdgeInsets.symmetric(horizontal: 12),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.start,  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: <Widget>[  
 Text(  
 entry.title,  
 style: Styles.*VideoRowItemName*,  
 ),  
 const Padding(padding: EdgeInsets.only(top: 8)),  
 Text(  
 '${entry.description}',  
 style: Styles.*VideoRowItemPrice*,  
 )  
 ],  
 ),  
 ),  
 ),  
  
 ],  
 ),  
 );  
  
 if (lastItem) {  
 return row;  
 }  
  
 return Column(  
 children: <Widget>[  
 row,  
 Padding(  
 padding: const EdgeInsets.only(  
 left: 100,  
 right: 16,  
 ),  
 child: Container(  
 height: 1,  
 color: Styles.*VideoRowDivider*,  
 ),  
 ),  
 ],  
 );  
 }  
}

screens/physical/location\_selection\_physical.dart

import 'package:flutter/cupertino.dart';  
import 'package:ivalueu\_app/model/video.dart';  
import 'package:ivalueu\_app/screens/physical/video\_list.dart';  
  
*/// The location selection for the physical tab*class PhysicalLocationSelection extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: const Text('Select location'),  
 ),  
 child: SafeArea(  
 child: Padding(  
 padding: const EdgeInsets.all(40),  
 child: Center(  
 child: Column(  
 children: [  
 Padding(  
 padding: const EdgeInsets.all(30),  
 child: Text("Can you perform indoor or outdoor activities?"),  
 ),  
 CupertinoButton(  
 onPressed: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(  
 builder: (context) => VideoList(location: Location.indoors, category: VideoCategory.physical,)  
 ),  
 );  
 },  
 child: Text(  
 'Indoors',  
 style: TextStyle(  
 color: CupertinoColors.*activeBlue*,  
 ),  
 ),  
 ),  
 CupertinoButton(  
 onPressed: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(  
 builder: (context) => VideoList(location: Location.outdoors, category: VideoCategory.physical,)  
 ),  
 );  
 },  
 child: Text(  
 'Outdoors',  
 style: TextStyle(  
 color: CupertinoColors.*activeBlue*,  
 ),  
 ),  
 ),  
 ],  
  
 )),  
 ),  
 ),  
 );  
 }  
}

screens/physical/physical.dart

import 'package:flutter/cupertino.dart';  
import 'package:ivalueu\_app/screens/physical/location\_selection\_physical.dart';  
  
*/// The homepage for the physical tab*class Physical extends StatelessWidget {  
 @override  
 Widget build(BuildContext context) {  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: const Text('IValueU'),  
 ),  
 child: SafeArea(  
 child: Padding(  
 padding: const EdgeInsets.all(20),  
 child: Center(  
 child: Column(  
 children: [  
 GestureDetector(  
 onTap: (){  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(builder: (context) => PhysicalLocationSelection()),  
 );  
 },  
 child: Image.asset('assets/images/MapleLeafLarge.png')),  
 GestureDetector(  
 onTap: (){  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(builder: (context) => PhysicalLocationSelection()),  
 );  
 },  
 child: Container(  
 width: 240.0,  
 height: 42.0,  
 decoration: BoxDecoration(  
 color: CupertinoColors.*systemGrey4*,  
 borderRadius: BorderRadius.circular(10),  
 boxShadow: [  
 BoxShadow(  
 color: CupertinoColors.*black*.withOpacity(0.1),  
 spreadRadius: 4,  
 blurRadius: 10),  
 ]  
 ),  
 child: Center(  
 child: Text(  
 'Physical Wellbeing',  
 style: TextStyle(  
 fontFamily: 'Arial',  
 fontSize: 18,  
 color: CupertinoColors.*activeBlue*,  
 height: 1,  
 ),  
 textAlign: TextAlign.center,  
 ),  
 ),  
 ),  
 ),  
 Padding(  
 padding: const EdgeInsets.all(40),  
 child: SizedBox(  
 width: 240,  
 child: Text(  
 "Follow the HITT and Yoga Tutorials created by the pupils for you to follow and help improve your wellbeing."),  
 ),  
 )  
 ],  
 )),  
 ),  
 ),  
 );  
 }  
}

screens/physical/video\_list.dart

import 'package:flutter/cupertino.dart';  
import 'package:flutter/foundation.dart';  
import 'package:ivalueu\_app/model/video.dart';  
import 'package:ivalueu\_app/model/video\_repository.dart';  
import 'video\_row\_item.dart';  
  
*/// This widget contains a list of [Video]s which are taken from the [VideosRepository].  
///  
/// This component is reused for both physical and emotional menus*class VideoList extends StatelessWidget {  
 const VideoList({  
 Key key,  
 this.category,  
 this.location  
 }) : super(key: key);  
  
 final VideoCategory category;  
 final Location location;  
 @override  
 Widget build(BuildContext context) {  
 final videos = VideosRepository.*loadVideos*(category, location);  
 return CustomScrollView(  
 semanticChildCount: videos.length,  
 slivers: <Widget>[  
 const CupertinoSliverNavigationBar(  
 largeTitle: Text('Videos'),  
 ),  
 SliverSafeArea(  
 top: false,  
 minimum: const EdgeInsets.only(top: 8),  
 sliver: SliverList(  
 delegate: SliverChildBuilderDelegate(  
 (context, index) {  
 if (index < videos.length) {  
 return VideoRowItem(  
 index: index,  
 video: videos[index],  
 lastItem: index == videos.length - 1,  
 );  
 }  
  
 return null;  
 },  
 ),  
 ),  
 ) // END OF NEW CONTENT  
 ],  
 );  
 }  
}

screens/physical/video\_preview/dart

import 'package:flutter/cupertino.dart';  
import 'package:ivalueu\_app/model/video.dart';  
import 'package:youtube\_player\_flutter/youtube\_player\_flutter.dart';  
import 'package:flutter/material.dart';  
  
*/// This widget displays the actual [Video] and is reused for each video*class VideoPreview extends StatelessWidget {  
 const VideoPreview({this.video});  
 final Video video;  
  
 @override  
 Widget build(BuildContext context) {  
 YoutubePlayerController \_controller = YoutubePlayerController(  
 initialVideoId: video.youtubeId,  
 flags: YoutubePlayerFlags(  
 autoPlay: true,  
 mute: true,  
 ),  
 );  
 return CupertinoPageScaffold(  
 navigationBar: CupertinoNavigationBar(  
 middle: Text(video.name),  
 ),  
 child: SafeArea(  
 child: Center(  
 child: Column(  
 children: [  
 YoutubePlayer(  
 controller: \_controller,  
 showVideoProgressIndicator: true,  
 liveUIColor: CupertinoColors.*systemRed*,  
 bottomActions: [  
 CurrentPosition(),  
 ProgressBar(isExpanded: true),  
  
 ],  
 )  
 ],  
 )),  
 ),  
 );  
 }  
}

screens/physical/video\_row\_item.dart

import 'package:flutter/cupertino.dart';  
import 'package:ivalueu\_app/screens/physical/video\_preview.dart';  
  
import 'package:ivalueu\_app/model/video.dart';  
import 'package:ivalueu\_app/Styles.dart';  
  
*/// This widget creates one entry to be used in a list of videos  
///  
/// Is reused for evert video*class VideoRowItem extends StatelessWidget {  
 const VideoRowItem({  
 this.index,  
 this.video,  
 this.lastItem,  
 });  
  
 final Video video;  
 final int index;  
 final bool lastItem;  
  
 @override  
 Widget build(BuildContext context) {  
 final row = SafeArea(  
 top: false,  
 bottom: false,  
 minimum: const EdgeInsets.only(  
 left: 16,  
 top: 8,  
 bottom: 8,  
 right: 8,  
 ),  
 child: GestureDetector(  
 onTap: () {  
 Navigator.*push*(  
 context,  
 CupertinoPageRoute(  
 builder: (context) => VideoPreview(video: video,)),  
 );  
 },  
 child: SizedBox(  
 height: 110,  
 child: Column(  
 children: <Widget>[  
 ClipRRect(  
 borderRadius: BorderRadius.circular(4),  
 child: Image.asset(  
 video.iconAsset,  
 fit: BoxFit.cover,  
 width: 76,  
 height: 76,  
 ),  
 ),  
  
 Expanded(  
 child: Padding(  
 padding: const EdgeInsets.symmetric(horizontal: 12),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.start,  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: <Widget>[  
 Text(  
 video.name,  
 style: Styles.*VideoRowItemName*,  
 ),  
 const Padding(padding: EdgeInsets.only(top: 8)),  
 ],  
 ),  
 ),  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
  
 if (lastItem) {  
 return row;  
 }  
  
 return Column(  
 children: <Widget>[  
 row,  
 Padding(  
 padding: const EdgeInsets.only(  
 left: 100,  
 right: 16,  
 ),  
  
 ),  
 ],  
 );  
 }  
}

lib/main.dart

import 'package:flutter/cupertino.dart';  
import 'package:flutter/material.dart';  
import 'package:flutter/services.dart';  
import 'screens/app.dart';  
  
*/// Starts the app*void main() {  
 return runApp(MyApp()  
 );  
}  
GlobalKey<NavigatorState> mainNavigatorKey = GlobalKey<NavigatorState>();  
  
*/// The app widget. Our story starts here.  
///  
/// Contains some configuration.*class MyApp extends StatelessWidget {  
 // This widget is the root of your application.  
 @override  
 Widget build(BuildContext context) {  
 SystemChrome.*setPreferredOrientations*(  
 [DeviceOrientation.portraitUp, DeviceOrientation.portraitDown]);  
 return CupertinoApp(  
 home: MyHomePage(),  
 theme: CupertinoThemeData(brightness: Brightness.light),  
 debugShowCheckedModeBanner: false,  
 localizationsDelegates: [  
 DefaultMaterialLocalizations.*delegate*,  
 DefaultCupertinoLocalizations.*delegate*,  
 DefaultWidgetsLocalizations.*delegate*,  
 ],  
 );  
 }  
}

lib/app.dart

import 'package:flutter/cupertino.dart';  
import 'home/home.dart';  
import 'physical/physical.dart';  
import 'emotional/emotional.dart';  
import 'photo\_diary/photo\_diary.dart';  
import 'more/more.dart';  
  
*/// This widget contains the navigation of the app.  
///  
/// All other widgets are children of this widget.*class MyHomePage extends StatefulWidget {  
 MyHomePage({Key key, this.title}) : super(key: key);  
 final String title;  
 @override  
 \_MyHomePageState createState() => \_MyHomePageState();  
  
}  
  
class \_MyHomePageState extends State<MyHomePage> {  
  
 GlobalKey<NavigatorState> navigatorKey = GlobalKey<NavigatorState>();  
 @override  
 Widget build(BuildContext context) {  
  
 return CupertinoTabScaffold(  
 tabBar: CupertinoTabBar(  
 iconSize: 200,  
 items: <BottomNavigationBarItem>[  
 BottomNavigationBarItem(  
 icon: Image.asset("assets/images/IValueU Grey maple leaf.png"),  
 activeIcon: Image.asset("assets/images/IValueU Blue maple leaf.png"),  
 ),  
 BottomNavigationBarItem(  
 icon: Image.asset("assets/images/Physical Grey maple leaf.png"),  
 activeIcon: Image.asset("assets/images/Physical Blue maple leaf.png"),  
 ),  
 BottomNavigationBarItem(  
 icon: Image.asset("assets/images/Emotional Grey maple leaf.png"),  
 activeIcon: Image.asset("assets/images/Emotional Blue maple leaf.png"),  
 ),  
 BottomNavigationBarItem(  
 icon: Image.asset("assets/images/Photo Diary Grey maple leaf.png"),  
 activeIcon: Image.asset("assets/images/Photo Diary Blue maple leaf.png"),  
 ),  
 BottomNavigationBarItem(  
 icon: Image.asset("assets/images/More Grey maple leaf.png"),  
 activeIcon: Image.asset("assets/images/More Blue maple leaf.png"),  
 ),  
 ],  
 ),  
  
 tabBuilder: (context, index) {  
 CupertinoTabView returnValue;  
 switch (index) {  
 case 0:  
 returnValue = CupertinoTabView(builder: (context) {  
 return CupertinoPageScaffold(  
 child: Home(),  
 );  
 });  
 break;  
 case 1:  
 returnValue = CupertinoTabView(builder: (context) {  
 return CupertinoPageScaffold(  
 child: Physical(),  
 );  
 });  
 break;  
 case 2:  
 returnValue = CupertinoTabView(builder: (context) {  
 return CupertinoPageScaffold(  
 child: Emotional(),  
 );  
 });  
 break;  
 case 3:  
 returnValue = CupertinoTabView(builder: (context) {  
 return CupertinoPageScaffold(  
 child: PhotoDiary(),  
 );  
 });  
 break;  
 case 4:  
 returnValue = CupertinoTabView(builder: (context) {  
 return CupertinoPageScaffold(  
 child: More(),  
 );  
 });  
 break;  
 }  
 return returnValue;  
 },  
 );  
  
 }  
  
}