

FINAL PROJECT REPORT

SEMESTER 2, ACADEMIC YEAR: 2024-2025

CT312H: MOBILE PROGRAMMING

- **Project/Application name:** Coffee Information App
- **GitHub link:** <https://github.com/24-25Sem2-Courses/ct312hm01-project-qbinh1423>
- **Link youtube:** <https://youtu.be/Py-nawmpLOk>
- **Student ID 1:** B2111972
- **Student Name 1:** Nguyen Tran Quang Binh
- **Student ID 2:** B2111964
- **Student Name 2:** Bui Ngoc Truc
- **Class/Group Number (e.g, CT312HM01):** CT312HM01

I. Introduction


- Project/application description: The Coffee Information App is an application that provides detailed information about coffees, including their origins, characteristics, descriptions, as well as brewing methods and coffee-related drinks. In addition, users can explore information about coffee brewing tools and addresses of coffee shops. The application also integrates a favorite feature that helps users save their favorite coffees and addresses and supports a dark/light interface for an easier user experience.
- A task assignment sheet for each member if working in groups.

Task assignment	Quang Binh	Ngoc Truc
Interface design	user	admin
Backend	user, admin	admin, user

II. Details of implemented features

1. Feature / Application page 1: Login/Register

- **Description:** The Login/Register page allows users to Register a new account if they do not have one and login to the application using their registered account.
- **Screenshots:**




Email

Password

Login

Don't have account? [Register](#)



Name

Email

Password

Confirm

Already have an account? [Let's login](#)

- **Implementation details:**

- + List the widgets used for this feature/page. Is there any special widget (not introduced in the lesson) used? If so, state them.
 - List of widgets: AuthScreen, AppBanner, AuthCard, Card, Form, SingleChildScrollView SnackBar, CircularProgressIndicator
- + Does the feature use any libraries or plugins? If so, state them and state the role of those libraries/plugins.
 - ValueListenableBuilder: Used to listen for change events in the data submission widget and update the interface while the form is submitting data.
- + Does this feature use a shared state management solution? If so state briefly how your solution works and describe the code architecture.
 - Provider: Used for state management, access authentication methods.
 - PocketBase: Used as backend, Authentication management, account creation and login session management.
- + Does this feature read or store any data? Locally or remotely? If so, state the data table structure. If you are using a REST API, briefly describe that API (how to call, input, output).
 - Collection: users
 - Use Provider to manage AuthManager state.

- AuthManager keeps the user logged in and notifies the UI when it changes.
- AuthService handles operations with PocketBase and notifies AuthManager when the authentication state changes.
- AuthService listens for authentication state changes via `pb.authStore.onChange.listen()`.
- When the authentication state changes, it updates AuthManager, which in turn notifies the UI.

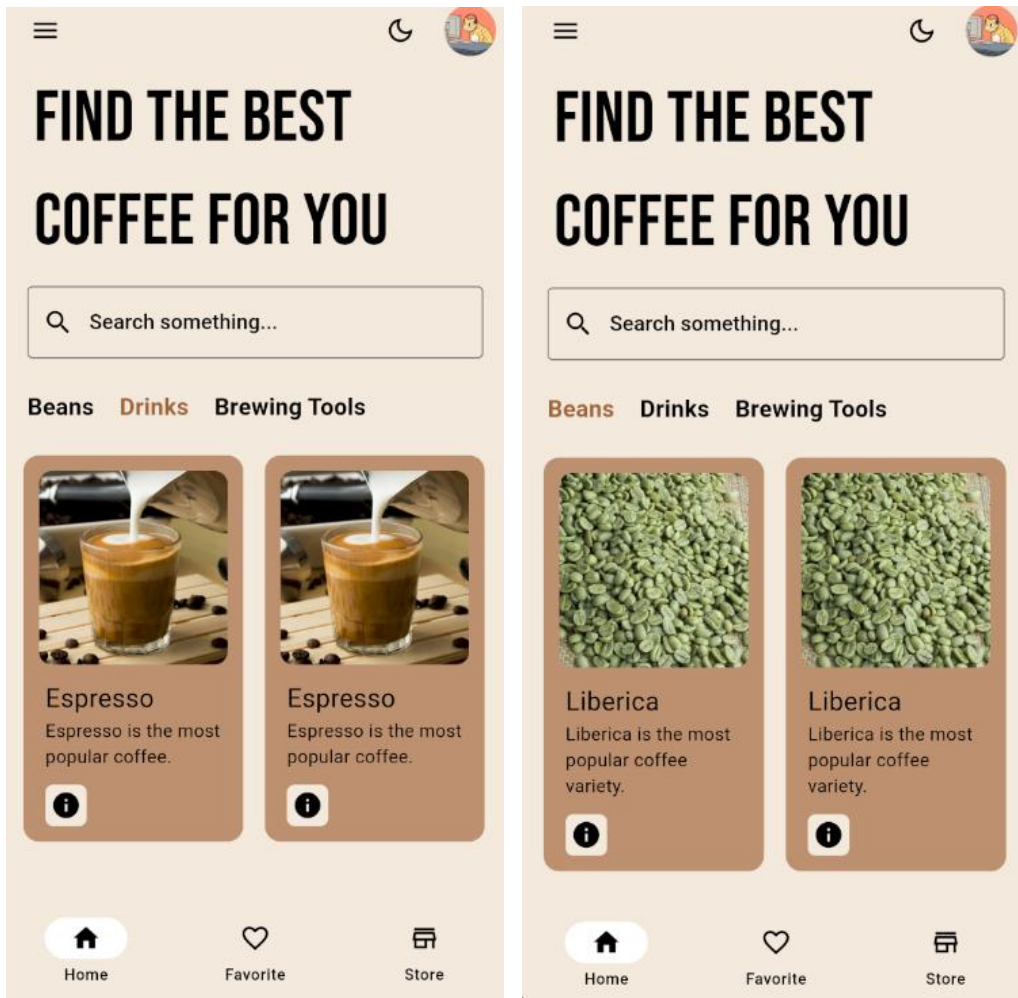
Name *
users
Type: Auth

Fields
API Rules
Options

T id	Nonempty	⚙️
🔒 password	Nonempty Hidden	⚙️
T tokenKey	Nonempty Hidden	⚙️
✉ email	Nonempty	⚙️
👁 emailVisibility		⚙️
👁 verified		⚙️
T name		⚙️
🖼 avatar	Single	⚙️
T role		⚙️
T phone		⚙️
📅 created	Create	⚙️

2. Feature / Application page 2: List page

- **Description:** The List Page allows users to browse through a categorized list of coffee beans, drinks, and brewing tools. It displays items based on user selection and enables navigation to detailed item pages for more information, such as the favorite page and store page.
- **Screenshots:**



- **Implementation details:**

- + List the widgets used for this feature/page. Is there any special widget (not introduced in the lesson) used? If so, state them.
 - Standard Flutter Widgets: Scaffold, AppBar, ListView.builder, Padding, SizedBox, Column, Row, GestureDetector, TextField
 - Custom Widgets: BeansCard, DrinksCard, BrewingToolsCard, CoffeeType
 - Special Widgets: ValueListenableBuilder
- + Does the feature use any libraries or plugins? If so, state them and state the role of those libraries/plugins.
 - Provider, Google Fonts, Pocketbase
- + Does this feature use a shared state management solution? If so state briefly how your solution works and describe the code architecture.
 - State Management: Provider

- + Does this feature read or store any data? Locally or remotely? If so, state the data table structure. If you are using a REST API, briefly describe that API (how to call, input, output).
 - Data is stored in PocketBase. UI listens to changes in Provider and updates dynamically.
 - Collection: bean, tool, drink

Name *

bean

Type: Base

Fields

API Rules

T	id		Nonempty	⚙️
T	name			⚙️
T	origin			⚙️
T	altitude			⚙️
T	climate			⚙️
T	caffeine			⚙️
T	description			⚙️
🖼️	beanImage	Single	▼	⚙️
🔗	userId	users	Single ▼	⚙️
📅	created	Create	▼	⚙️
📅	updated	Create/Update	▼	⚙️

Name *
drink

Type: Base

Fields

API Rules

T	id		Nonempty	
T	name			
T	origin			
T	ingredients			
T	description			
T	caffeine			
	tool	tool	Single ▾	
	drinkImage		Single ▾	
	userId	users	Single ▾	
	created		Create ▾	
	updated		Create/Update ▾	





Name *

Type: Base

tool

Fields

API Rules

T	id	Nonempty		
T	name			
T	origin			
T	type			
T	material			
T	description			
	toolImage	Single		
	userId	users	Single	
	created	Create		
	updated	Create/Update		

3. Feature / Application page 3: Coffee detail page

- **Description:** This page displays detailed information about a particular coffee. Information includes: Coffee image, origin, altitude, climate, caffeine content. Users can add coffee to their favorites list or share coffee information with others.
- **Screenshots:**



- **Implementation details:** students should answer the following questions:
 - + List the widgets used for this feature/page. Is there any special widget (not introduced in the lesson) used? If so, state them.
 - Scaffold, AppBar, IconButton, Padding, Icon, DrawerUser, Provider, ThemeButton, CircleAvatar, Column, Expanded, ListView, GestureDetector, Dialog, ClipRRect, Image.asset, SizedBox, Text, GoogleFonts, ListTile, FontAwesomeIcons, Row, Container, IconButton, SingleChildScrollView, BoxDecoration, BorderRadius, TextAlign.
 - + Does the feature use any libraries or plugins? If so, state them and state the role of those libraries/plugins.
 - provider: Manages the state for ThemeProvider, which helps to toggle between light and dark mode.
 - google_fonts: Apply custom fonts.
 - font_awesome_flutter: Use icons from FontAwesome.

- + Does this feature use a shared state management solution? If so state briefly how your solution works and describe the code architecture.
 - State Management: Provider, ChangeNotifier,
 - Provider is wrapped around the root widget using MultiProvider to give access to the theme across the entire app.
- + Does this feature read or store any data? Locally or remotely? If so, state the data table structure. If you are using a REST API, briefly describe that API (how to call, input, output).
 - Collection: bean
 - Displays information about coffee beans on this page. Data is stored on Pocketbase

Name *

bean

Type: Base

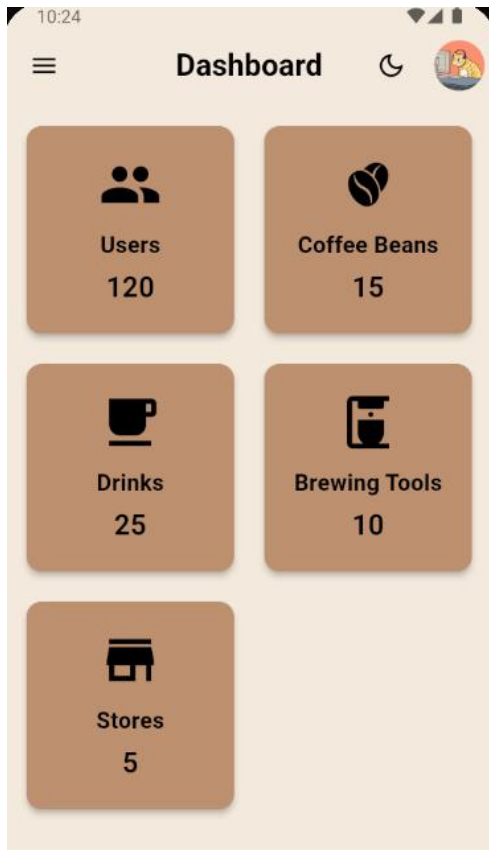
Fields

API Rules

T	id		Nonempty	⚙️
T	name			⚙️
T	origin			⚙️
T	altitude			⚙️
T	climate			⚙️
T	caffeine			⚙️
T	description			⚙️
🖼️	beanImage		Single ▾	⚙️
🔗	userId	users	Single ▾	⚙️
📅	created		Create ▾	⚙️
📅	updated		Create/Update ▾	⚙️

4. Feature / Application page 4: Dashboard page

- **Description:** The Dashboard page displays key statistics about the application, including the number of users, coffee beans, drinks, brewing tools, and stores added. It provides an overview of the application's data in a grid format.
- **Screenshots:**



- **Implementation details:** students should answer the following questions:
 - + List the widgets used for this feature/page. Is there any special widget (not introduced in the lesson) used? If so, state them.
 - Scaffold, AppBar, IconButton, Padding, GridView.count, Card, DashboardCard, CircleAvatar, ThemeButton, Image.asset, Column, SizedBox, Text, RoundedRectangleBorder, BorderRadius.circular
 - + Does the feature use any libraries or plugins? If so, state them and state the role of those libraries/plugins.
 - Libraries: material, provider
 - + Does this feature use a shared state management solution? If so state briefly how your solution works and describe the code architecture.
 - Use Provider (ThemeProvider) to manage the light/darkness of the mode.
 - `themeProvider.toggleTheme()` toggle dark/light mode on button press.
 - + Does this feature read or store any data? Locally or remotely? If so, state the data table structure. If you are using a REST API, briefly describe that API (how to call, input, output).
 - Collection: users, tool, drink, bean, stores
 - The dashboard fetches statistical data dynamically from PocketBase. It retrieves real-time counts for users, coffee beans, drinks, brewing tools,

Name *

bean

Type: Base

Fields

API Rules

T	id	Nonempty	
T	name		
T	origin		
T	altitude		
T	climate		
T	caffeine		
T	description		
	beanImage	Single	
	userId	users	Single
	created	Create	
	updated	Create/Update	

Name *
users

Type: Auth

Fields

API Rules

Options

T id

Nonempty



password

Nonempty Hidden



T tokenKey

Nonempty Hidden



email

Nonempty



emailVisibility



verified



T name



avatar

Single



T role



T phone



created

Create



Name *

drink

Type: Base

Fields

API Rules

T	id	Nonempty		
T	name			
T	origin			
T	ingredients			
T	description			
T	caffeine			
	tool	tool	Single	
	drinkImage		Single	
	userId	users	Single	
	created		Create	
	updated		Create/Update	

Name *

store

Type: Base

Fields

API Rules

T	id	Nonempty	
T	name		
T	location		
T	phone		
T	description		
	startTime		
	endTime		
	storeImage	Single ▼	
	userId	users	Single ▼
	created	Create ▼	
	updated	Create/Update ▼	

- **Implementation details:** students should answer the following questions:
 - + List the widgets used for this feature/page. Is there any special widget (not introduced in the lesson) used? If so, state them.
 - ListView, ElevatedButton, CircleAvatar, TextButton, Image, Container, FittedBox, ImagePicker, SnackBar, AlertDialog, Provider, ImagePicker
 - + Does the feature use any libraries or plugins? If so, state them and state the role of those libraries/plugins.
 - Libraries: material, provider, image_picker, dart:io
 - + Does this feature use a shared state management solution? If so state briefly how your solution works and describe the code architecture.
 - Use Provider to manage coffee data
 - ThemeProvider to manage light/dark themes
 - + Does this feature read or store any data? Locally or remotely? If so, state the data table structure. If you are using a REST API, briefly describe that API (how to call, input, output).
 - Collection: bean

- This feature integrates with PocketBase to store and retrieve coffee bean details. When users enter coffee information and save it, the app sends a request to PocketBase to create or update the record. Images are uploaded, and their URLs are stored in the database.

Edit collection

Name *
beanType: Base

Fields

API Rules

T idNonempty

T name

T origin

T altitude

T climate

T caffeine

T description

beanImage

Single

userId

users

Single

created

Create

updated

Create/Update