**CÔNG TY CÔNG NGHỆ & KỸ THUẬT STECH**

**VIỆN NGHIÊN CỨU & SÁNG TẠO KHOA HỌC MÁY TÍNH – IRICS**

**A logo with orange letters

Description automatically generated**

**BÁO CÁO NGHIÊN CỨU FLUTTER**

**TASK 6 – NGHIÊN CỨU FLUTTER**

Người hướng dẫn**: MR. NGUYỄN TẤN HÙNG ANH**

**MR. VÕ THÀNH LUÂN**

Người thực hiện**: ĐINH NGỌC AN THƠ**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024**

**TÓM TẮT**

Flutter là một framework phát triển giao diện người dùng (UI) mã nguồn mở do Google phát triển, cho phép tạo các ứng dụng di động, web và desktop từ một codebase duy nhất. Được viết bằng ngôn ngữ Dart, Flutter nổi bật với tính năng hot reload giúp các nhà phát triển thấy ngay kết quả thay đổi mà không cần khởi động lại ứng dụng. Giao diện trong Flutter được xây dựng bằng cách sử dụng các widget, tất cả mọi thứ từ các nút bấm, văn bản đến bố cục đều là widget có thể tùy chỉnh và lồng vào nhau. Flutter cung cấp hiệu suất cao nhờ việc biên dịch mã nguồn thành mã máy gốc, và nó kết nối ứng dụng với các nền tảng gốc như Android và iOS, giúp phát triển ứng dụng nhanh chóng và hiệu quả.

**MỤC LỤC**

[CHƯƠNG I: PHÂN TÍCH CODE FILE PROJECT TASK 6 5](#_Toc168669405)

[1. history\_page.dart 5](#_Toc168669406)

[2. home\_page.dart 6](#_Toc168669407)

[3. result\_page.dart 9](#_Toc168669408)

[CHƯƠNG II: ĐIỂM KHÁC NHAU GIỮA CÁC PROJECT 11](#_Toc168669409)

[CHƯƠNG III: TỔNG HỢP 13](#_Toc168669410)

**DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT**

**DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ**

[Hình 1: phân tích code history\_page file project task 6…………………………………… 5](#_Toc168669449)

[Hình 2: phân tích code home\_page file project task 6 7](#_Toc168669450)

[Hình 4: hàm trả kết quả project task 6 9](#_Toc168669451)

[Hình 5: các class result\_page của project task 6 10](#_Toc168669452)

[Hình 6 : result\_page của file project task 6 11](#_Toc168669453)

CHƯƠNG I: PHÂN TÍCH CODE FILE PROJECT TASK 6

1. history\_page.dart

**A screen shot of a computer program

Description automatically generated**

*Hình 1: phân tích code history\_page file project task6*

- import :

Material : thư viện sử dụng các widget và các thành phần giao diện của Flutter

- class BmiHistory :

‘BmiHistory’ : là một class được gọi để chứa lịch sử các chỉ số BMT

‘\_bmiHistory’ : danh sách static chứa các chuỗi String là chỉ số BMI và được khởi tạo rỗng

‘bmiHistory’ : là một getter tĩnh trả về danh sách ‘\_bmiHistory’

‘addBmi(String bmi)’ : là một phương thức static( tĩnh) để thêm một chỉ số BMI vào danh sách ‘\_bmiHistory’

-class HistoryPage :

‘HistoryPage’ : là một StatelessWidget đại diện cho trang hiển thị lịch sử BMI

Phương thức ‘build(BuildContext context) : xây dựng giao diện của trang ‘HistoryPage’

‘Scaffold’ : là một widget cung cấp cấu trúc cơ bản cho một trang ( như appBar, body)

-‘appBar’ : có tiêu đề là “Lịch Sử BMI”

-‘body’ : chứa ‘container’ với màu nền được lấy từ chủ đề hiện tại của ứng dụng

- Bên trong ‘container’ là một ‘ListView.build’ để tại danh sách các mục ‘ListTile’ :

‘itemCount’ : là số lượng các mục trong danh sách lịch sử BMI

‘itemBuilder’ là hàm xây dựng từng mục ‘ListTile’ hiển thị chỉ số BMI từ danh sách ‘BmiHistory.bmiHistory’

1. home\_page.dart

**A screen shot of a computer program

Description automatically generated**

*Hình 2: phân tích code home\_page file project task 6*

-import :

/material : thư viện sử dụng các widget và thành phần giao diện của Flutter

/item\_count\_number\_button : dùng để sử dụng các widget ‘ItemCount’

/result\_page.dart và /history\_page.dart : để sử dụng các trang kết quá và lịch sử BMI

-void main() : điểm khởi đầu của ứng dụng, gọi runApp để chạy ứng dụng với widget

-class MyApp:

‘MyApp’ : là một StatelessWidget đại diện cho ứng dụng

‘build’ : trả về một ‘MatarialApp’ với một ‘ThemeData’ tùy chỉnh và home là trang HomePage

-class HomePage và \_HomePageState : ‘HomePage’ là một StatefulWidget tạo một trạng thái ‘\_HomePageState’

-class \_HomePageState : các biến trạng thái để lưu trữ giới tính, chiều cao, tuổi , cân nặng và chỉ số BMI

**-**build:

‘AppBar’ : có tiêu đề “BMI CALCULATOR “ và nút biểu tượng lịch sử để chuyển đến trang ‘HistoryPage’

‘body’ gọi hàm ‘\_buildUI’ để xây dựng giao diện chính

‘floatingActionButton’ : để tính toán BMI khi nhấn

-Widget \_buildUI : Xây dựng giao diện chính với các widget con : bộ chọn giới tính, nhập chiều cao, nhập cân nặng, tuổi và kết quả BMI

**A screen shot of a computer

Description automatically generated**Hình 3: widget home\_page của project task 6

**-** Widget \_genderSelector : widget để chọn giới tính nam hoặc nữ. Khi nhấn vào biểu tượng, cập nhật trạng thái ‘selectedGender’

**-**Widget \_heightInput : Widget để nhập chiều cao bằng ‘Slider’. Giá trị chiều cao thay đổi được hiển thị dưới dạng văn bản

**-**Widget \_weightAndAgeInput : widget để nhập cân nặng bằng ‘ItemCount’, là một widget với nút tăng/giảm

**-**Widget \_ageInput : Widget nhập tuổi bằng ‘ItemCount’

**-**Widget \_bmiResult : Widget để hiển thị kết quả BMI và nút để tính toán lại BMI

**-**void \_calculateBMI : Hàm tính toán chỉ số BMI dựa trên cân nặng và chiều cao để lưu kết quả vào lịch sử và chuyển đến trang kết quả ‘ResultsPage’

**-**String \_getResultText : Trả về văn bản kết quả dựa trên chỉ số BMI

**-**String \_getInterpretation : Trả về văn bản giải thích dựa trên chỉ số BMI

**-**void \_recalculate : Hàm để quay lại trang trước đó khi tính lại BMI

**A computer screen shot of text

Description automatically generated**

Hình 4: hàm trả kết quả project task 6

1. result\_page.dart

**A screen shot of a computer screen

Description automatically generated**

Hình 5: các class result\_page của project task 6

**-** import :

/material : thư viện cho phép sử dụng widget và các thành phần giao diện của Flutter

- class ResultPage:

‘class ResultPage’: là StatelessWidget dùng để hiển thị kết quả BMI

‘const’ constructor với các tham số bắt buộc là :

- ‘bmiResult’ : kết quả BMI dưới dạng chuỗi

- ‘resultText’ : chuỗi văn bản mô tả kết quả BMI

- ‘interpretaton’ : chuỗi giải thích kết quả BMT

- ‘onnecalcculate’ : hàm callback khi người dùng muốn tính toán lại BMI

-Var: ‘final String bmiResult ‘ : các biến thành viên của class ‘ResultPage’, được truyển vào từ bên ngoài khi khởi tạo

-‘build(BuildContext context)’ : Dùng để xây dựng giao diện của trang và ‘Scaffold’ cung cấp cấu trúc cơ bản cho trang (appBar và body)

**A screen shot of a computer program

Description automatically generated**

Hình 6 : result\_page của file project task 6

-body:

‘Container’ : có màu nền lấy từ chủ đề hiện tại của ứng dụng

‘center’ : giúp căn giữa các thành phần bên trong

‘Column’ : sắp xếp các thành phần theo chiều dọc

‘Text’ : hiển thị kết quả BMI với kích thước và độ đậm của phông chữ khác nhau

‘SizedBox’ : tạo khoảng cách giữa các thành phần

‘ElevatedButton’ : dùng để tính lại BMI. Khi nhấn vào sẽ gọi hàm ‘onRecalculate’

🡪 ‘ResultsPage’ là trang giao diện dùng để hiển thị kết quả BMT cùng với một nút để tính lại BMI. Trang gồm các Text hiển thị kết quả BMI để mô tả kết quả và giải thích, cùng với nút dùng để tính lại

CHƯƠNG II: ĐIỂM KHÁC NHAU GIỮA CÁC PROJECT

**-** BMI dựa trên thông tin giới tính, chiều cao, độ tuổi và cân nặng của người dùng để hiển thị kết quả và đưa ra lời khuyên sức khỏe. Ngoài ra còn lưu lại lịch sử các chỉ số BMI đã tính toán và cho phép xem lại lịch sử này

|  |  |  |
| --- | --- | --- |
|  | **Giống nhau** | **Khác nhau** |
| **myCard** | **-** package : material khai báo các widget cơ bản  **-** Sử dụng các widget : Column,ListTile  - Xây dựng Widget với phương thức trả về là Material App : cung cấp các tính năng để thiết kế giao diện cho ứng dựng Flutter  -Sử dụng Text để hiển thị văn bản cần thiết  -Sử dụng SizedBox : chỉnh kích thước cần  -‘debugShowCheckedModeBanned : false’ : ẩn biểu ngữ ‘DEBUG’ bên phải màn hình khi ứng dụng chạy ở chế độ riêng  - Dùng widget hiển thị ứng dụng là Scaffolf để thêm các thiết kế phổ biến như AppBars, Drawers  -@override : thể hiện một phương thức đang ghi đè lên một phương thức của lớp cha  -sử dụng setState để thông báo có sự thay đổi trạng thái cần cập nhật giao diện của người dùng  -onPressed  \*Quizzler và bmi điều hướng đến một trang khác để thể hiện tính năng khác của app | **-** myCard sử dụng widget: CircleAvatar: là widget hiển thị hình ảnh trong khung tròn. Divider: widget vẽ đường gạch ngang hoặc đường dọc |
| **xucxac** | - xucxac tạo class DicePage : để tạo widget có thể thay đổi trang thái là ‘DisPage’  - Có setState cập nhật biến sinh số ngẫu nhiên là Random().nextInt  - Sử dụng Image.asset để hiển thị hình ảnh của xúc xắc được lấy từ thư mục ‘images’  -Sử dụng items: để tạo danh sách các mục thả xuống với mỗi key là một tên của một loài động vật  -onChanged : () để xử lý sự kiện khi người dùng chọn mục khác trong nút thả xuống |
| **xyxophone** | **-**package : audioplayer cho phép ứng dụng có thể phát âm thanh  -Sử dụng các hàm để đặt thời gian phát lại âm thanh đã phát trước đó : startReplayTimer |
| **Quizzlzer** | **-** Sử dụng các hàm để đặt hiệu ứng pháo hoa nếu có số câu trả lời đúng cao hơn số câu trả lời sai  -Sử dụng onHover để theo dõi khi chuốt hover qua các widget thì các widget đổi mai bằng cách khai báo và sử dụng ‘isHovering’ |

CHƯƠNG III: TỔNG HỢP

- Hiểu cách hoạt động của Stateful Widgets, cách quản lý trạng thái với setState. Biết khi nào nên sử dụng Stateless Widget

- Sử dụng các widget nhập liệu ‘Slide’, ‘ItemCount’, ‘IconButton’. Và cách đảm bảo dữ liệu nhập vào là hợp lệ trước khi thực hiện tính toán

-Viết các hàm để tính toán chỉ số BMI từ các dữ liệu nhập vào, lưu trữ các chỉ số BMI vào trong một danh sách tĩnh

-Sử dụng gói thư viện bên ngoài như ‘**item\_count\_number\_button’** để thêm tính năng cho ứng dụng

-Clean code : cách viết một mã nguồn rõ ràng