

Urmă Tudor-Irinel

Fiteat : aplicație mobilă pentru un stil de viață sănătos

Universitatea Alexandru Ioan Cuza Iași

FACULTATEA DE INFORMATICĂ



LUCRARE DE LICENȚĂ

Fiteat

Realizată de : Urmă Tudor-Irinel

Sesiunea: Iulie, 2022

Coordonator: Lect. Dr. Cristian Vidrașcu

Urmă Tudor-Irinel

Fiteat : aplicație mobilă pentru un stil de viață sănătos

Universitatea Alexandru Ioan Cuza Iași

FACULTATEA DE INFORMATICA



**Fiteat : aplicație mobilă
pentru un stil de viață sănătos**

Realizată de : Urmă Tudor-Irinel

Sesiunea: Iulie, 2022

Coordonator: Lect. Dr. Cristian Vidrașcu

Urmă Tudor-Irinel

Fiteat : aplicație mobilă pentru un stil de viață sănătos

Anexa 4

Avizat,
Îndrumător Lucrare de Licență
Titlul, Numele și prenumele _____ Lect. Dr. Cristian Vidrașcu

Data _____ Semnătura _____


DECLARAȚIE privind originalitatea conținutului lucrării de licență

Subsemnatul(a) Urmă Tudor-Irinel domiciliul în Iași, născut(ă) la data de 11.09.2000 , identificat prin CNP 5000911226785 , absolvent(a) al(a) Universității „Alexandru Ioan Cuza” din Iași, Facultatea de Informatică specializarea Informatică , promoția 2022, declar pe propria răspundere, cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art.143 al. 4 si 5 referitoare la plagiat, că lucrarea de licență cu titlul: „ Fiteat : aplicație mobilă pentru un stil de viață sănătos” elaborată sub îndrumarea dl. / d-na Cristian Vidrașcu, pe care urmează să o susțin în fața comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea conținutului său într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice în vederea facilitării falsificării de către cumpărător a calității de autor al unei lucrări de licență, de diplomă sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Data azi,
23.06.2022.....

Semnătură student
.....

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „ Fiteat : aplicație mobilă pentru un stil de viață sănătos”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” Iași să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași, data

23.06.2022

Absolvent Urmă Tudor-Irinel



(semnătura în original)

ACORD PRIVIND PROPRIETATEA DREPTULUI DE AUTOR

Facultatea de Informatică este de acord ca drepturile de autor asupra programelor-calculator, în format executabil și sursă, să aparțină autorului prezentei lucrări, Urmă Tudor-Irinel. Încheierea acestui acord este necesară din următoarele motive: În viitor, doresc continuarea dezvoltării acestei aplicații și lansarea ei pentru a putea fi accesată de publicul larg.

Iași, data

23.06.2022

Absolvent Urmă Tudor-Irinel



(semnătura în original)

Cuprins

Cuprins	6
Introducere.....	8
Contribuții	9
1. Tehnologii Folosite	10
1.1. Dart.....	10
1.2. Flutter	10
1.3. Firebase.....	11
1.4. Python.....	12
1.5. NewsAPI.....	12
2. Dispozitive.....	13
2.1. Android Studio Emulator	13
2.2. Iphone 12 mini	13
3.Arhitectura aplicației.....	14
3.1. Firestore DB.....	22
3.2. Firebase Storage.....	26
3.3. Firebase Machine Learning	26
3.4. Firebase Auth.....	26
3.5. Flutter Application	27
3.6. Food Recognition Model.....	32
3.5. News API.....	36
4. Paginile aplicației.....	37
4.1. Înregistrare.....	37
4.2. Autentificare	38
4.2.1. Schimbare parolă uitată	39
4.3. Setare detalii și scop.....	40
4.4. Acasă	41
4.4.1. Știri din baza de date	42
4.4.2. Știri oferite de către News API	43

4.5. Jurnal caloric	44
4.5.1. Adăugare aliment	45
4.5.2. Adăugare exercițiu	47
4.5.3. Editare Aliment	48
4.5.4. Editare exercițiu	48
4.5.5. Finalizare	49
4.6. Progres.....	49
4.6.1. Adăugare greutate curenta	50
4.7. Mai multe.....	51
4.7.1. Schimbare poza de profil	52
4.7.2. Schimbare detalii și/sau scop	52
4.7.3. Schimbare parolă.....	53
5. Concluzii.....	54
5.1. Avantaje.....	54
5.2. Dezavantaje.....	55
6. Îmbunătățiri posibile ale aplicației	56
6.1. Îmbunătățirea modelului pentru recunoașterea mâncării	56
6.2. Istoric jurnal	56
6.3. Asocierea cu alte aplicații.....	57
Bibliografie	58

Introducere

Toată lumea dorește să ducă o viață cât mai îndelungată și fără probleme medicale.

Subnutriția și bolile carentiale sunt, au fost și vor fi o problemă cu care se confruntă o mare parte din populație. Astfel încetul cu încetul mai mulți oameni doresc să ducă un stil de viață mai sănătos.

Dar ce înseamnă un stil de viață sănătos?

Un stil de viață sănătos cuprinde o alimentație sănătoasă, activitate fizică și menținerea unei greutate care se află în parametri adecvați.

Acest răspuns ne duce acum cu gândul la întrebarea : „Ce înseamnă o alimentație sănătoasă?”. O alimentație sănătoasă constă în oferirea organismului „combustibilul” necesar pentru o bună funcționare.

„Combustibilul” pe care îl oferim trebuie să aibă două mari proprietăți:

- Cantitate
- Calitate

Calitatea unui aliment vine atât din valorile nutriționale pe care le oferă cât și din procesele prin care acesta a trecut, iar când vine vorba de cantitate, aceasta trebuie aleasă în funcție de individ și de dorințele acestuia.

De aici nu trebuie exclusă nici activitatea fizică care crește cantitatea „combustibilului” pe care o oferim.

Contribuții

Crearea unei aplicații tip jurnal caloric a pornit atât de la dorința de a ajuta populația când vine vorba de menținerea unui stil de viață sănătos, cât și din „problemele” pe care le-am avut în trecut.

Aplicația poate ajuta oamenii experimentați să își țină o evidență asupra alimentației și asupra activității fizice, iar pe cei „novici” îi va ajuta atât partea de evidență cât și partea de știri pe care o oferă aplicația , loc de unde pot învăța multe despre acest domeniu.

Partea de software este susținută de o aplicație mobilă și de un pachet ce are grijă de toată partea funcțională (autentificare, bază de date, depozitare, învățare automată).

Factorul de noutate al aplicației vine prin prisma știrilor la care are utilizatorul acces și prin recunoașterea de alimente printr-o simplă poză care mai apoi oferă valorile nutriționale ale acelui aliment.

Sumarizând, aplicația este mai mult decât un simplu jurnal caloric, este un mic ajutor spre atingerea unui stil de viață sănătos.

1. Tehnologii Folosite

1.1. Dart

Dart este un limbaj optimizat pentru dezvoltarea de aplicații rapide pe orice platformă. Acesta a fost creat de către Google în anul 2011. Scopul acestuia este de a oferi dezvoltarea de aplicații client-side (aplicații cu care utilizatorul interacționează în mod direct).

Inițial scopul acestuia era să înlocuiască Javascript-ul, dar după o perioadă de timp scopul s-a transformat, devenind un limbaj productiv pentru dezvoltarea de aplicații multi-platform.

Caracteristici care fac Dart un limbaj de ales când dorim să dezvoltăm o aplicație client-side :

1. Este un limbaj simplu de citit, scris și învățat.
2. Pune la dispoziție „Hot reload” (se observă în timp real schimbările aduse asupra codului)
3. Opțiunea de a rula rapid și fără întârziere aplicația pe orice dispozitiv.

Dart a fost special gândit să includă toate aceste caracteristici, acesta a devenit foarte popular odată cu apariția framework-ului Flutter.

1.2. Flutter

Înainte, dezvoltarea de aplicații mobile era destul de costisitoare deoarece era nevoie de două aplicații separate care să fie compatibile cu sistemele de operare existente. Acest lucru implica realizarea unei aplicații de două ori.

Pentru a rezolva această problemă, au fost construite mai multe framework-uri pentru crearea de aplicații „multi-platform” în HTML5 și Javascript printre acestea numărându-se și Flutter.

Flutter a devenit rapid popular în rândul dezvoltatorilor, întreprinderilor, antreprenorilor și utilizatorilor.

Flutter este un set de instrumente portabil pentru UI pentru crearea de aplicații native pe mobil, web și desktop, dintr-o singură bază de cod folosind limbajul Dart. Cu ajutorul acestuia putem crea o interfață cu un aspect plăcut și care se comporta normal pe orice platformă, utilizând o singură bază de cod.

Avantajele acestui framework:

- Un singur cod de bază pentru toate platformele
- Foarte multe librării
- Testare rapidă cu „hot reload”
- Principiul de „widget” pe care se bazează oferă multe oportunități când vine vorba de efecte vizuale

1.3. Firebase

Deținut de Google, Firebase este un pachet complet de produse care vă permite să construiți aplicații web și mobile, să îmbunătățiți calitatea aplicației și să vă ajutați clienții să-și dezvolte afacerea.

Firebase de la Google poate fi utilizat pentru următoarele:

- Gestionează toate datele în timp real din baza de date.
- Permite sincronizarea datelor în timp real pe toate dispozitivele - Android, iOS și web fără a reîmprospăta ecranul.
- Oferă integrare în Google Ads, AdMob, DoubleClick, Play Store, Data Studio, BigQuery și Slack, pentru a face dezvoltarea aplicației dvs. cu o gestionare și întreținere eficiente și precise.
- Totul, de la baze de date, analize la rapoarte de blocare, este inclus în Firebase. Deci, echipele de dezvoltare a aplicațiilor pot rămâne concentrate pe îmbunătățirea experienței utilizatorului.

Caracteristicile principale pe care le are Firebase:

1. Firebase Machine Learning
2. Firebase Authentication
3. Firebase Cloud Storage
4. Firebase Database

1.4. Python

Python este foarte răspândit printre programatori, motivele principale fiind popularitatea, ușurința înțelegerii și utilizării limbajului.

Ca o mică istorie , construirea acestui limbaj a pornit ca un hobby de Crăciun pentru Guido van Rossum în anul 1989, acesta a dorit să dezvolte un interpretor de cod simplu, intuitiv, open source, accesibil pe orice platformă. Ideea de a denumi limbajul „Python” i-a venit de la serialul Monty Python’s Flying Circus difuzat pe BBC.

Cele mai importante avantaje sunt evidențiate mai jos:

- ușor de învățat – limbajul Python este un limbaj care se înțelege foarte repede spre deosebire de alte limbaje de programare
- ușor de folosit – fiind simplu și minimalist, limbajul Python, se apropie de pseudocodul din limba engleză permițând astfel focalizarea asupra găsirii problemei și nu pe utilizarea limbajului în sine.
- gratuit și open source – redistribuirea limbajului este permisă, la fel și a codului sursă, precum și editarea sau modificarea acestuia pentru a obține noi programe, deci poate fi și este în mod constant îmbunătățit de comunitate.
- portabil – acest limbaj este folosit pe o multitudine de platforme și sisteme de operare.
- orientare puternică pe obiecte – pot fi create cu succes aplicații folosind Programarea Orientată pe Obiecte (POO), de aici și forța acestui limbaj de programare.

1.5. NewsAPI

News API este un simplu HTTP REST API pentru căutarea și preluarea articolelor live de pe tot internetul. Pentru a folosi acest API este nevoie de o cheie API - aceasta este o cheie unică care identifică cererile dvs. Acest API primește si cuvinte cheie pentru a extrage articole relevante pentru dumneavoastră.

2. Dispozitive

2.1. Android Studio Emulator

Emulatorul Android are ca scop să simuleze dispozitive Android. Cu ajutorul acestuia putem testa aplicații pe diferite dispozitive fără a fi nevoie să avem versiunea acestuia fizică.

Acest Emulator oferă toate funcționalitățile unui dispozitiv Android fizic.

Testarea pe un astfel de emulator este mai rapidă și mai ușoară decât pe un dispozitiv real. Un exemplu ar fi transferul de date care se face mult mai ușor pe un emulator.

2.2. Iphone 12 mini

Telefon personal folosit pentru camera, întrucât dispozitivul virtual nu are camera.

3.Arhitectura aplicației

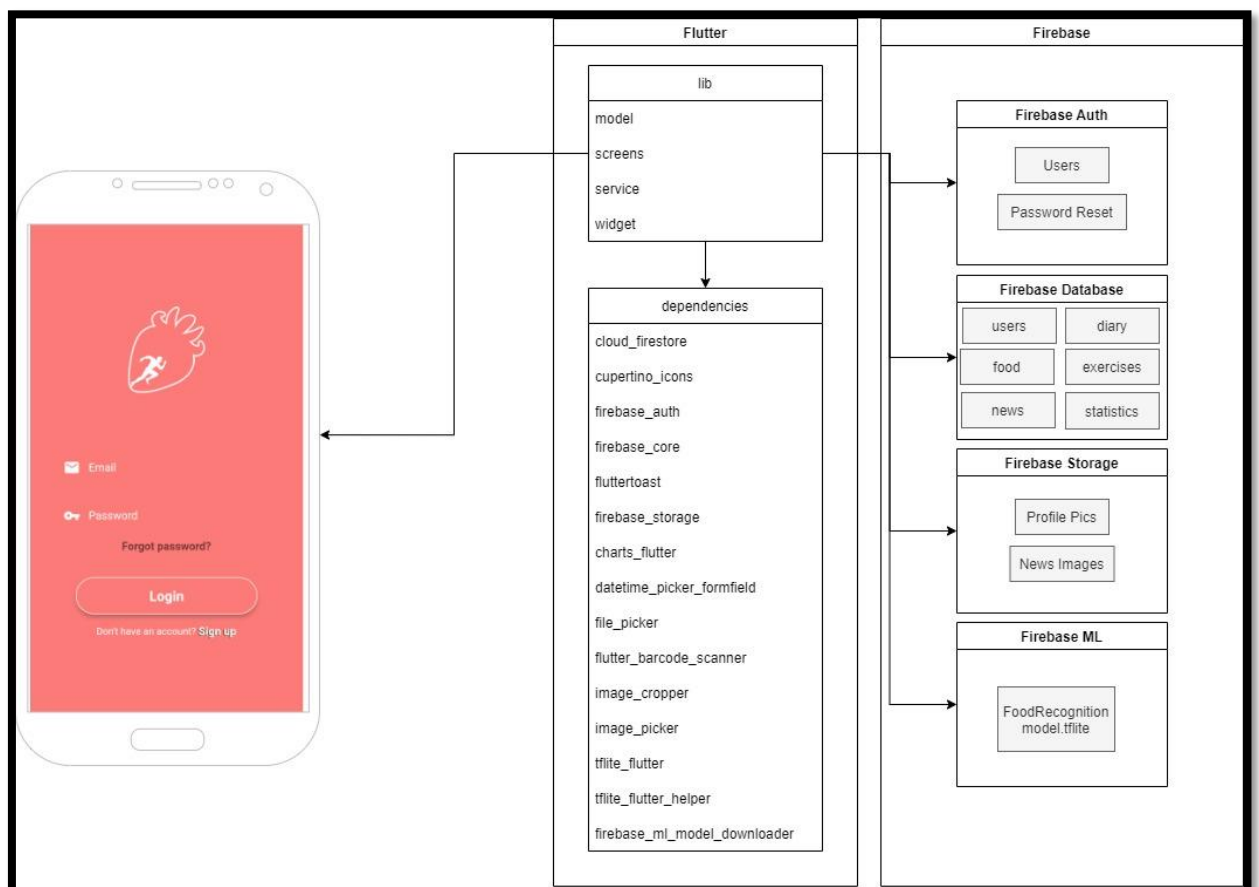


Fig. 1. Arhitectura aplicației

Diagrame C4

Modelul C4 este o tehnică de notare grafică pentru modelarea arhitecturii sistemelor software.

Nivel 1:

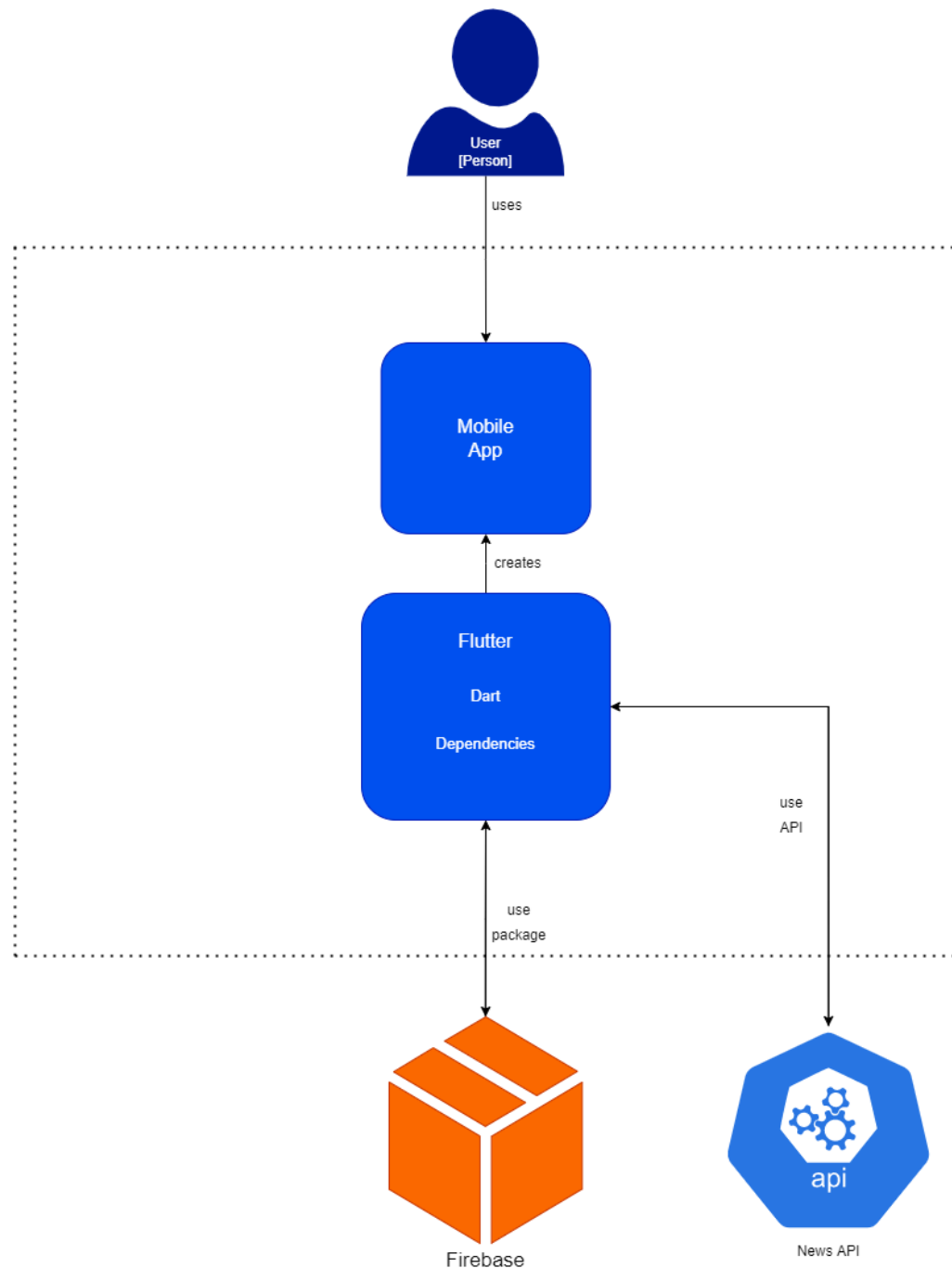


Fig. 2. Diagramă C4 Nivel 1

Nivel 2:

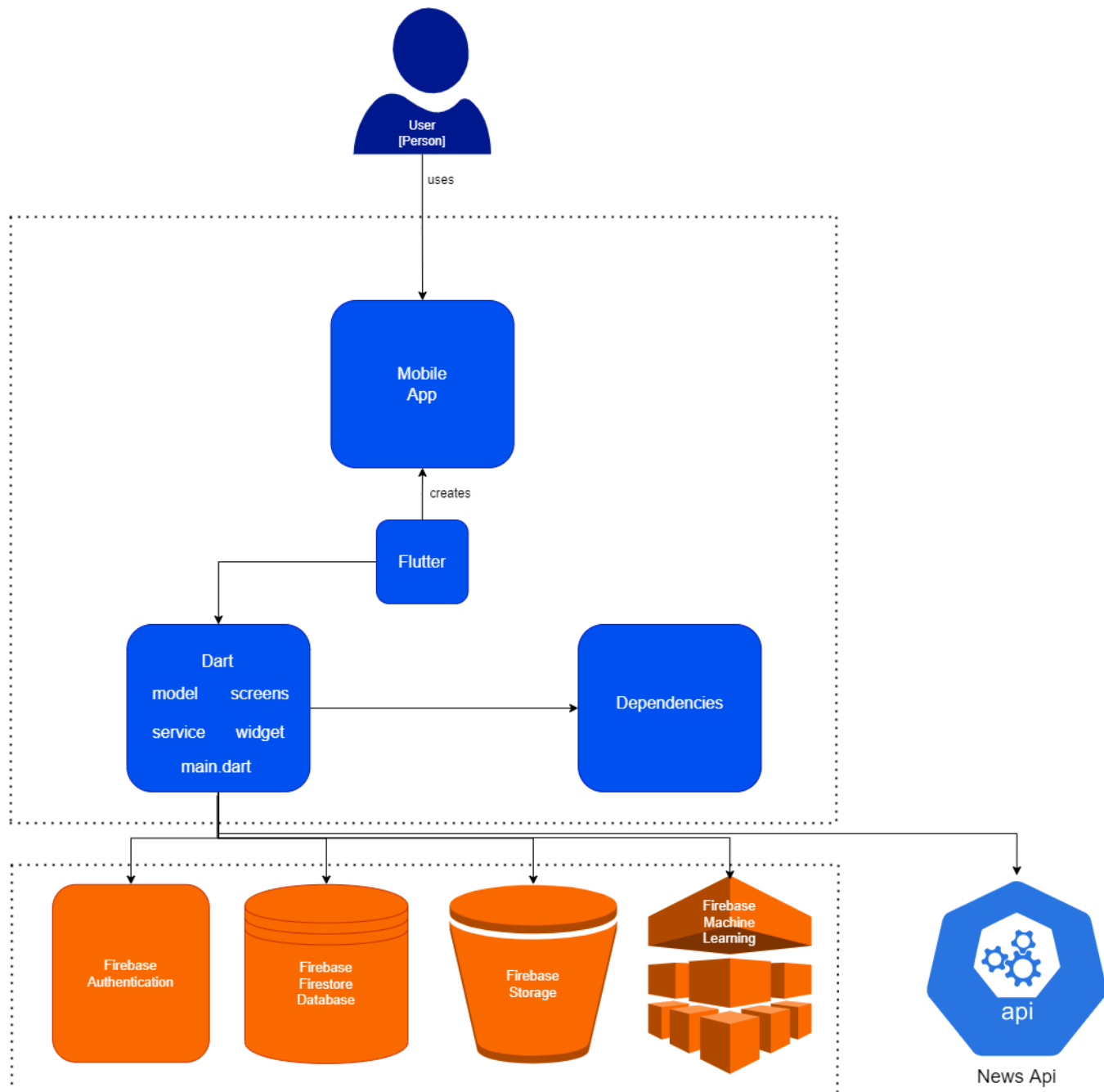


Fig. 3. Diagramă C4 Nivel 2

Nivel 3:

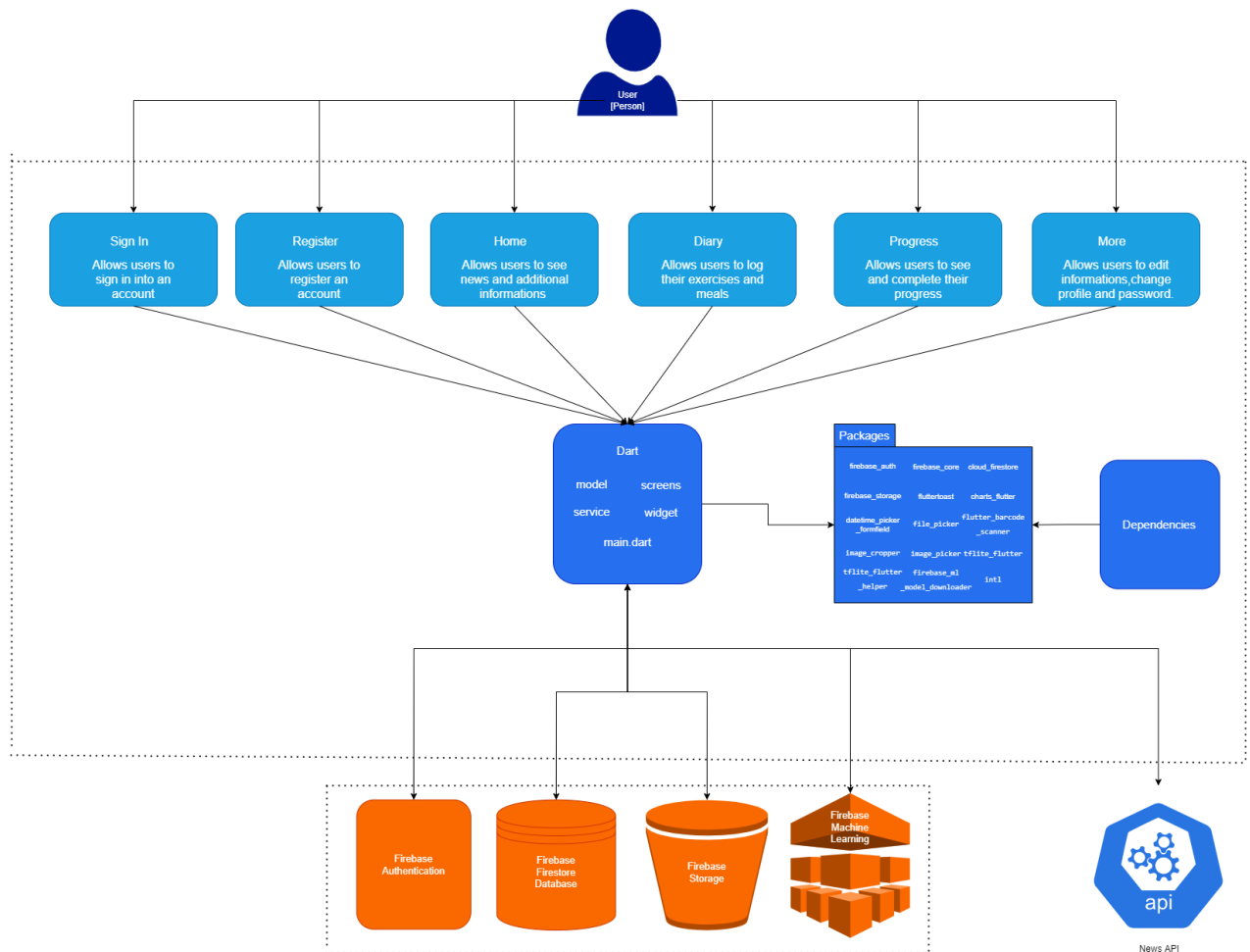


Fig. 4. Diagramă C4 Nivel 3

Nivel 4:

Sing In:

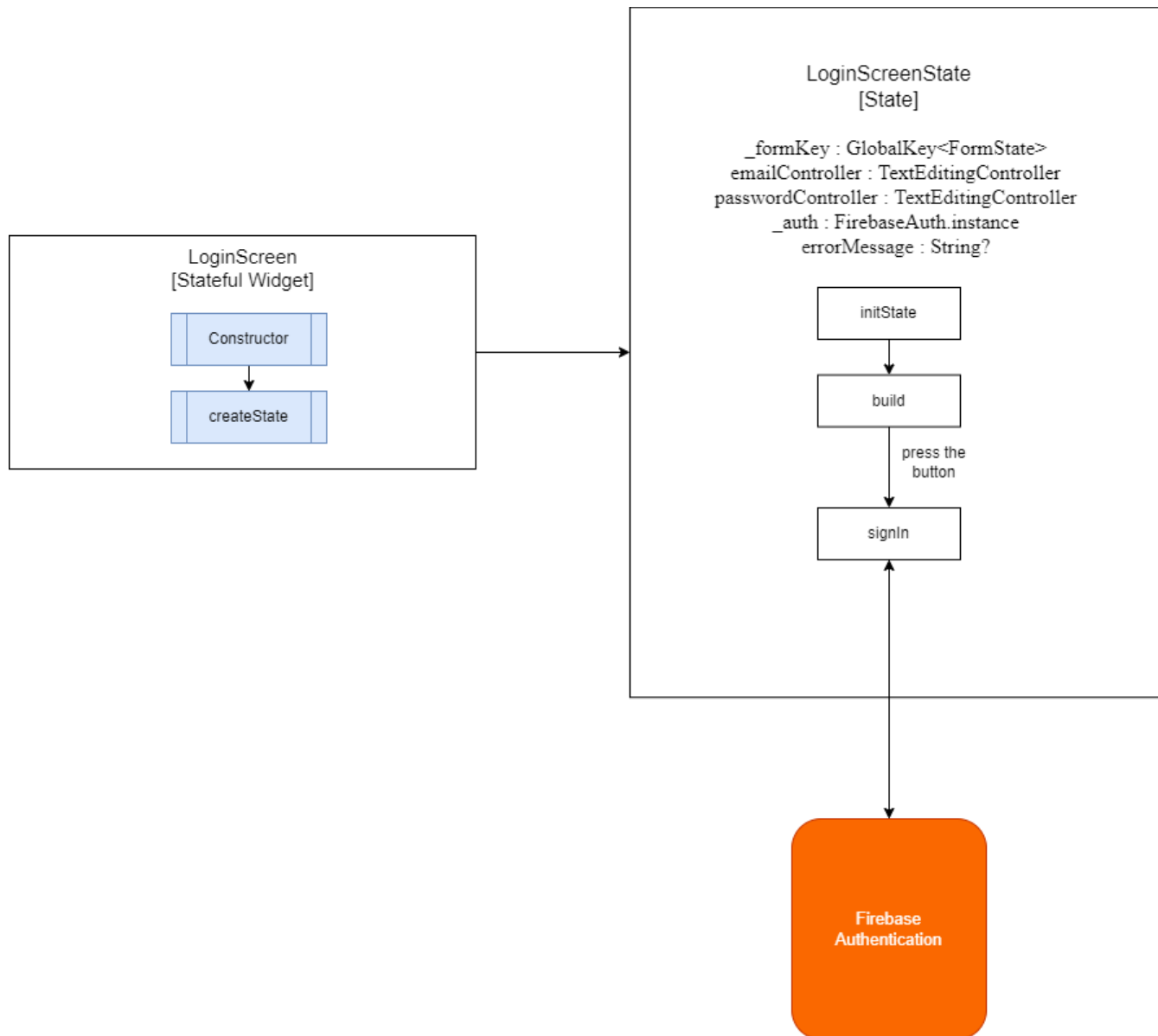


Fig. 5. Diagramă C4 Nivel 4 – Sign in

Register:

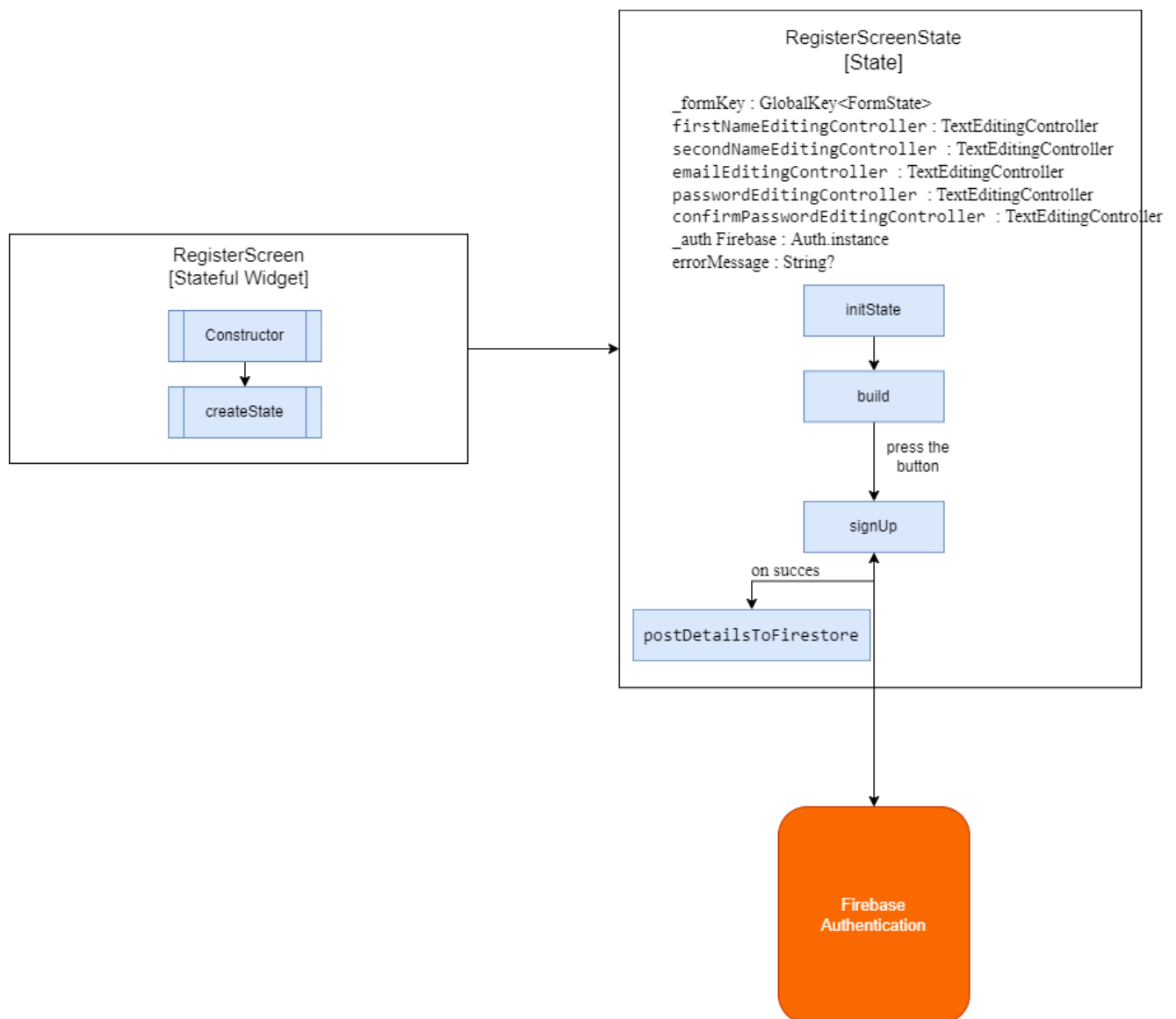


Fig. 6. Diagramă C4 Nivel 4 – Register

Home:

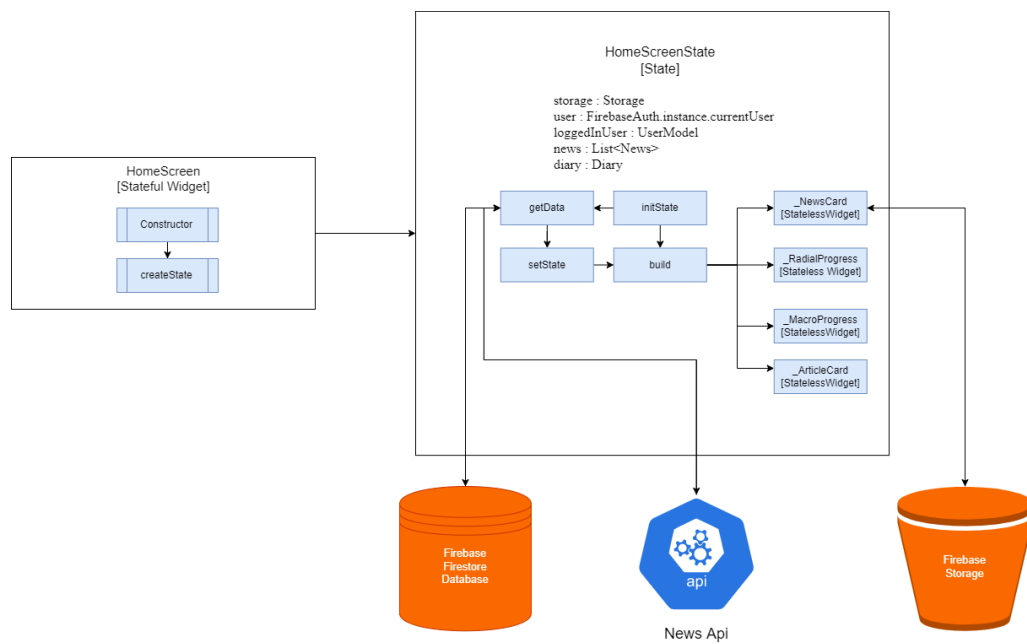


Fig. 7. Diagramă C4 Nivel 4 –Home

Diary:

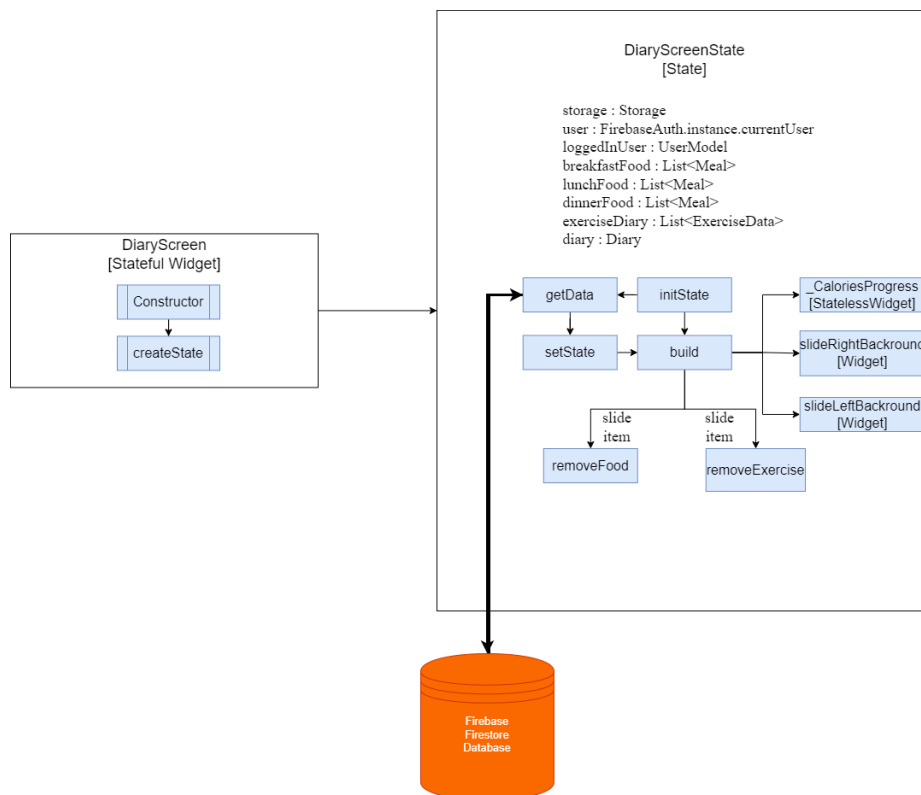


Fig. 8. Diagramă C4 Nivel 4 – Diary

Take a photo screen:

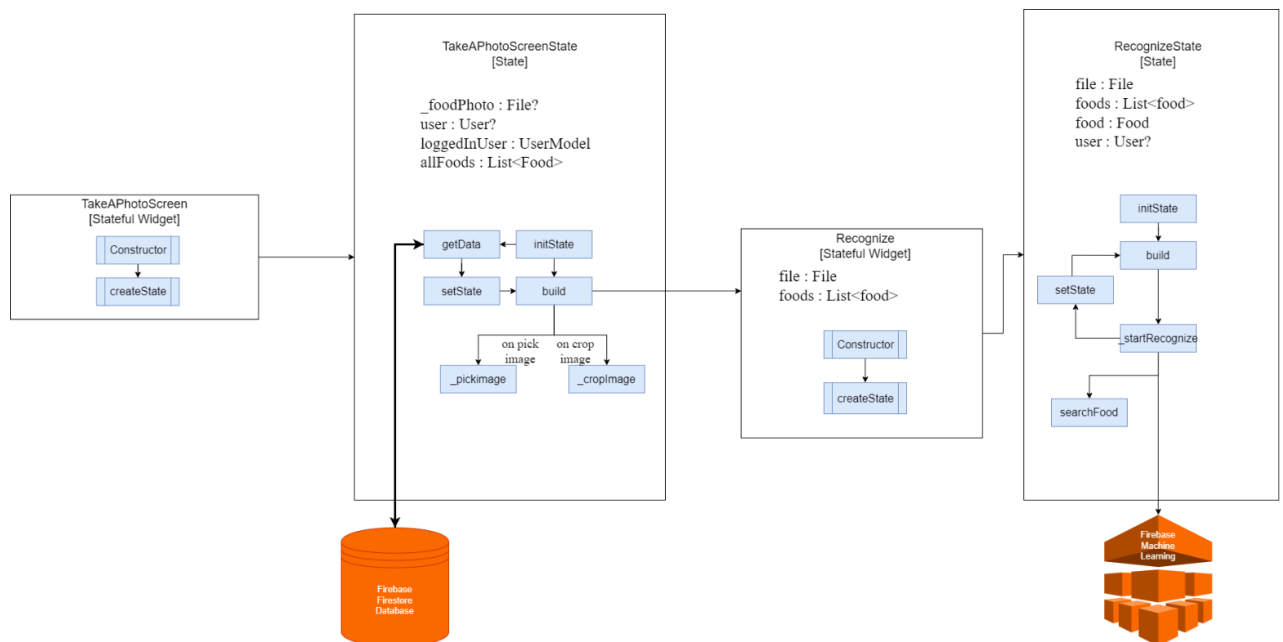


Fig. 9. Diagramă C4 Nivel 4 – Take a photo

Progress:

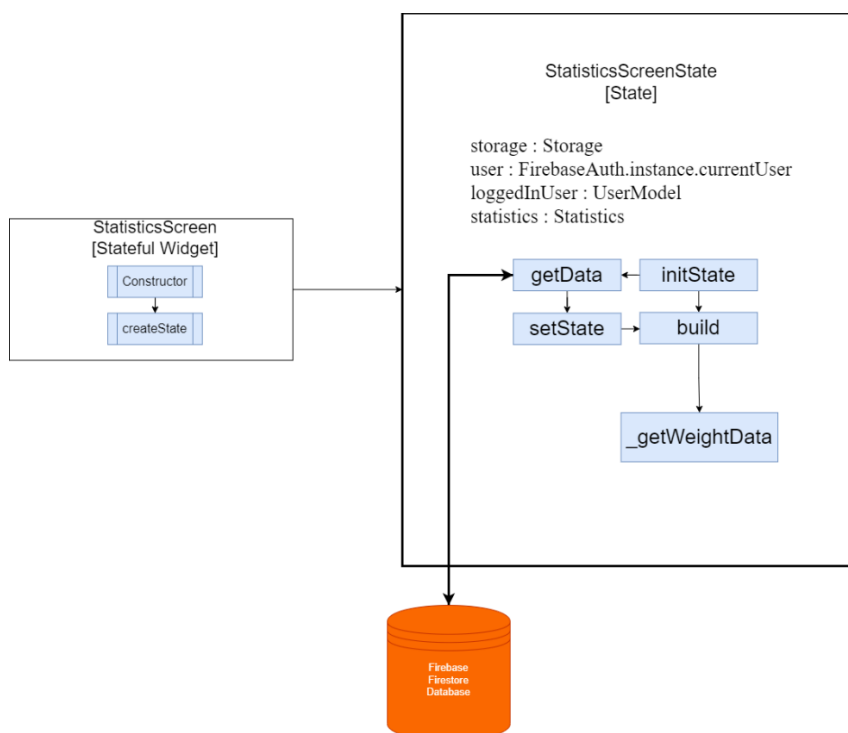


Fig. 10. Diagramă C4 Nivel 4 – Progress

More:

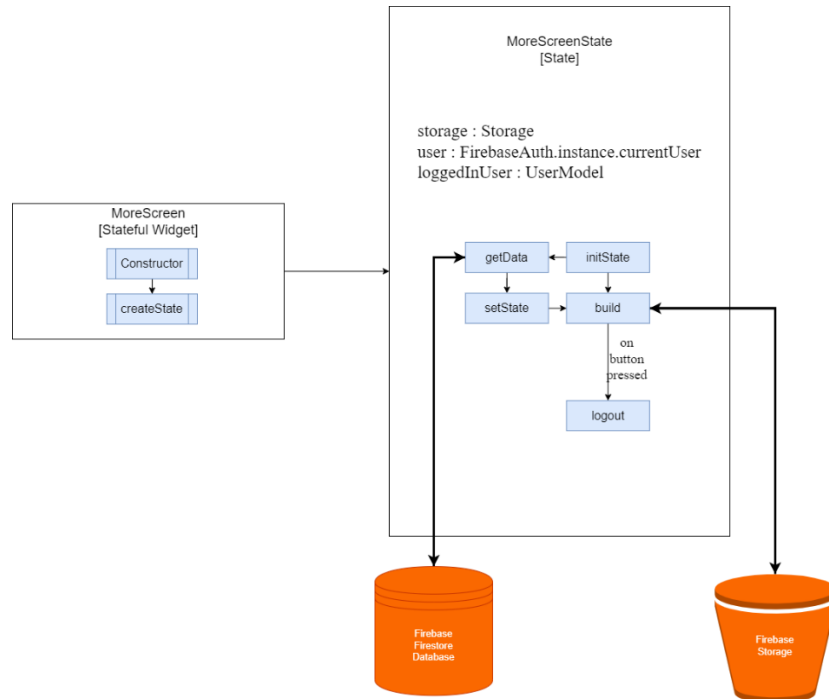


Fig.11. Diagramă C4 Nivel 4 – More

3.1. Firestore DB

Firestore DB este o bază de date de tip NoSQL care poate fi accesată direct de către aplicații folosind SDK-uri native. Aceasta face parte din pachetul pe care Firebase îl oferă.

Baza de date este o unealtă de colectare și organizare a informațiilor. Într-o bază de date putem stoca diferite informații despre persoane, obiecte, cerințe și multe altele.

Bazele de date de tip NoSQL (nerelaționale) stochează informațiile într-un mod diferit, nefolosind tabele relaționale. Acestea sunt utilizate în aplicații web și în Big Data tocmai datorită avantajelor pe care acestea le oferă. Marile avantaje sunt scalabilitatea și disponibilitatea ridicată. Totodată acestea permit stocarea informațiilor într-o manieră mai intuitivă și mai ușor de înțeles.

Baza de date pe care am construit-o conține 6 colecții de date, acestea fiind: diary, exercises, food, news, statistics și users. Fiecare colecție conține un document care este accesat printr-un id. Iar fiecare document reține informațiile despre fiecare element ce se află în colecția noastră. Mai jos voi prezenta fiecare colecție împreună cu scopul acesteia.

Colecția „diary” conține jurnalele calorice pentru utilizatorii care s-au înregistrat fiind salvate sub forma de documente. Fiecare jurnal poate fi accesat cu ajutorul uid-ului utilizatorului. Un document conține detalii legate de jurnalul utilizatorului curent.

Detaliile prezente sunt următoarele:

- Calorii pe care utilizatorul le-a consumat
- Calorii pe care utilizatorul le-a „ars”
- Proteine pe care utilizatorul le-a consumat
- Carbohidrați pe care utilizatorul le-a consumat
- Grăsimi utilizatorul le-a consumat
- Obiect de tip Map ce conține exercițiile utilizatorului din acea zi
- Obiect de tip Map ce conține alimentele consumate în acea zi organizate în funcție de mesele dintr-o zi(mic dejun, prânz, cină)
- Uid-ul utilizatorului

diary	I6pF8CLPpYVQe3q1TC3zVAXDa2i1	+ Add field
exercises	PpWpGTaghjUJ7EsIsWyBvagNBDB3	carbs: 214
food		exercise: 0 (number) ✎ 🗑
news		▼ exercises
statistics		fats: 38.2
users		food: 1598.6
		▶ meals: {breakfast: {10: 1.2}, din...}
		protein: 115.4
		uid: "I6pF8CLPpYVQe3q1TC3zVAXDa2i1"

Fig. 12. Colecția diary

Colecția „exercises” conține exercițiile stocate în baza de date pe care utilizatorul le poate adăuga în jurnalul lui caloric. Fiecare exercițiu are un id unic prin care putem accesa un document. Acest document conține detalii legate de exercițiul respectiv :

- Caloriile arse pe minut
- Id
- Nume

diary	0	+ Add field
exercises	1	caloriesPerMinute: 1
food	2	id: "0" (string) ✎ 🗑
news	3	name: "quickAdd"

Fig. 13. Colecția exercises

Colecția „food” conține alimentele stocate în baza de date pe care utilizatorul le poate adăuga în jurnalul caloric. Fiecare aliment deține un id unic, iar cu ajutorul acestuia putem accesa documentul în care se află detaliile alimentului :

- Denumire
- Informații adiționale
- Id/cod de bare
- Calorii
- Proteine
- Carbohidrați
- Grăsimi
- Mărimea Porției

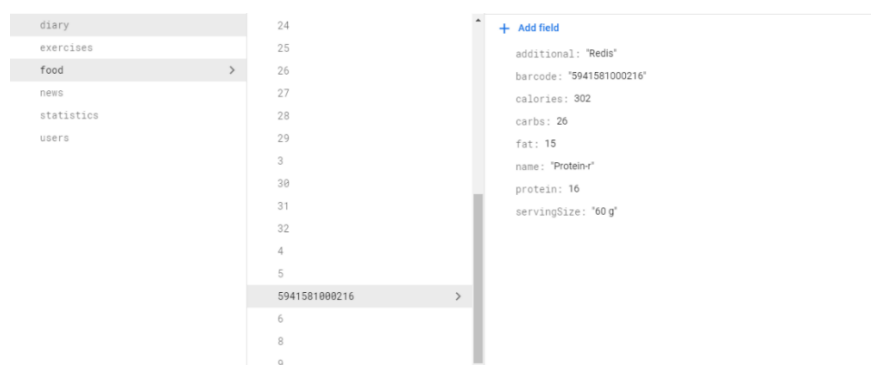


Fig. 14. Colecția food

Colecția „news” conține știrile ce se află în baza de date, știri ce le vor fi prezentate utilizatorului. La fel ca cele de mai sus , putem accesa documentele din interiorul colecției cu ajutorul unui id unic. Acest document conține următoarele informații:

- Titlul
- Tipul știrii
- Numele imaginii pe care o vom găsi în partea de depozitare
- Rezumat
- Titlurile principale
- Informații asignate fiecărui titlu

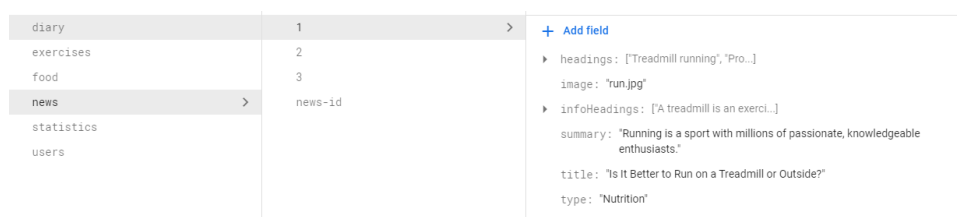


Fig. 15. Colecția news

Colecția „statistics” unde găsim informații ce ne vor ajuta pentru a afișa utilizatorului progresul pe care acesta l-a realizat. Fiecare document din interiorul acestei colecții se afla sub uid-ul utilizatorului. Acest document conține:

- Un array ce conține data înregistrării progresului curent
- Un array ce conține greutatea pe care acesta a înregistrat-o
- Uid-ul utilizatorului

date	weight
"01-04-2022"	69
"11-04-2022"	69.4
"15-04-2022"	69.8
"20-04-2022"	
"26-04-2022"	
"02-05-2022"	
"05-05-2022"	
"10-05-2022"	
"10-05-2022"	

Fig. 16. Colecția statistics

Colecția „users” conține detaliile fiecărui utilizator înregistrat. Accesul la documente se face prin prisma uid-ului , iar documentele conțin:

- Nume
- Prenume
- Email
- Sex
- Data de naștere
- Greutate
- Înălțime
- Nivelul de activitate
- Scop (dacă dorește sa slăbească, să se îngrașe sau își mențină greutatea)
- Scopul caloric (calculat cu ajutorul formulei si informațiile utilizatorului)
- Uid

diary	I6pF8CLPpYVQe3q1TC3zVAXDa2i1	+ Add field
exercises	PpWpGTaghjUU7EsIsWyBvagNBDB3	activityLevel: 1
food	uid	dob: "11-09-2000"
news		email: "tirynel@yahoo.com"
statistics		firstName: "Irinel"
users		gender: "Man"
		goal: -0.2
		goalcalories: 2271
		height: 175
		secondName: "Urma"
		uid: "I6pF8CLPpYVQe3q1TC3zVAXDa2i1"
		weight: 72.5

Fig. 17. Colecția users

3.2. Firebase Storage

Firebase Storage a fost creat pentru a ajuta cu stocarea și difuzarea rapidă și ușoară a conținutului generat de utilizatori. În cazul acestei aplicații vom depozita doar imagini.

Aici am stocat imaginile știrilor și imaginile de profil ale utilizatorilor.

3.3. Firebase Machine Learning

Firebase ML aduce experiența Google ML la aplicațiile Android și Apple într-un pachet puternic și totuși ușor de utilizat. În cadrul aplicației am folosit opțiunea de „Custom models”, loc unde am stocat modelul „FoodRecognition” antrenat de către mine ca mai apoi să îl folosesc în aplicație.

3.4. Firebase Auth

Firebase Authentication oferă servicii de backend, SDK-uri ușor de utilizat și biblioteci UI pentru a permite utilizatorilor să se autentifice în aplicația dumneavoastră. Prin intermediul acestuia puteți accepta autentificarea folosind parole, numere de telefon, furnizori de identitate federați populari precum Google, Facebook și Twitter și multe altele. Totodată acesta oferă și ajutor pentru trimiterea de email-uri către utilizator.

Scopul folosirii în intermediul aplicație a fost :

- Permitearea autentificării utilizatorilor
- Oferirea opțiunii de „am uitat parola”

3.5. Flutter Application

Această parte a arhitecturii realizează atât partea de interfață grafică a utilizatorului cât și funcționalitățile aplicației.

Scopul interfeței grafice este de a ajuta utilizatorul sa folosească aplicația implementată fără ca acesta sa dețină anumite cunoștințe în programare.

Partea de funcționalități se ocupa cu :

- Comunicarea cu pachetul Firebase
- Interpretarea și injectarea informațiilor primite prin comunicare
- Realizarea legăturii dintre interfață și Firebase

Pentru implementarea acestei părți am folosit limbajul Dart deoarece framework-ul Flutter are la bază acest limbaj.

Partea de implementare este împărțită în 4 sub-părți:

- Models
- Services
- Screens
- Widgets

„Models” este partea din implementare care ne ajută la comunicarea cu baza de date din interiorul pachetului Firebase. Pentru fiecare colecție avem un obiect care poate interpreta informațiile care „vin și pleacă”. Cu scopul de a se înțelege aceasta parte voi prezenta un exemplu concret.

Colecția Diary conține jurnalele salvate in baza de date. Fiecare document reprezintă un jurnal care are anumite informații adiționale asiguate(uid, calorii arse prin exerciții, calorii consumate, proteine, carbohidrați, grăsimi, exercițiile realizate, alimente consumate).

Astfel clasa Diary din aceste modele are ca rol sa extragă aceste informații in tipurile de date specifice și/sau sa le pregătească pentru a fi încărcate in baza de date.

```

class Diary {
  String? uid;
  double? exercise;
  double? food;
  double? fats;
  double? carbs;
  double? protein;
  Map<String, dynamic>? exercises;
  Map<String, Map<String, dynamic>>? meals;
  Diary(
    {this.uid, this.exercise, this.food, this.fats, this.carbs,
    this.protein, this.exercises, this.meals});
  factory Diary.fromMap(map) {
    return Diary(
      uid: map['uid'],
      exercise: map['exercise'].toDouble(),
      food: map['food'].toDouble(),
      protein: map['protein'].toDouble(),
      carbs: map['carbs'].toDouble(),
      fats: map['fats'].toDouble(),
      exercises: Map<String, dynamic>.from(map['exercises']),
      meals: Map<String, Map<String, dynamic>>.from(map['meals'])
    );
  }
  Map<String, dynamic> toMap() {
    return {
      'uid': uid,
      'exercise': exercise,
      'food': food,
      'protein': protein,
      'carbs': carbs,
      'fats': fats,

```

Fig. 18. diary.dart

„Services” este partea care se ocupă cu serviciile aplicației. Spre exemplu serviciul părții de storage. Acesta are ca scop să realizeze legătura dintre aplicație și Firebase Storage. Serviciul acesta se ocupă cu încărcarea și descărcarea fișierelor salvate în depozit. Partea de încărcare cuprinde schimbarea fotografiei de profil a utilizatorului și încărcarea acesteia pe depozit. Partea de descărcare cuprinde imagini salvate care sunt folosite în partea de interfață grafică (poza de profil, imagine știri).

```

class Storage{
  final firebase_storage.FirebaseStorage storage =
    firebase_storage.FirebaseStorage.instance;

  Future<void> uploadFile(
    String filePath,
    String fileName,
  )async{
    File file = File(filePath);

    try{
      await storage.ref('test/$fileName').putFile(file);
    } on firebase_core.FirebaseException catch (e){
      print(e);
    }
  }

  Future<firebase_storage.ListResult> listFiles() async{
    firebase_storage.ListResult results = await storage.ref('test').listAll();

    results.items.forEach((firebase_storage.Reference ref){
      print('found file: $ref');
    });

    return results;
  }

  Future<String> downloadURL(String imageName) async{
    String downloadURL = await storage.ref('test/${imageName}').getDownloadURL();

    return downloadURL;
  }

  Future<String> getProfilePicture(String uid) async{
    String imageUrl = 'user-profile/' + uid + '.png';
    String error = "";
    String downloadURL = "";
    try{
      await storage.ref(imageUrl).getDownloadURL();
    } on firebase_storage.FirebaseException catch(myError)
    {
      switch (myError.code)
      {
        case 'object-not-found':
          error = myError.toString();
        }
      }
  }
}

```

Fig. 19. storage_service.dart

„Screens” cuprinde toata partea de interfață grafică. Fiecare pagină a aplicației este creată de către un astfel de fișier. În interiorul acestora ne putem ocupa și de anumite funcționalități. Voi prezenta ca exemplu pagina pentru jurnalul caloric.

Această pagina cuprinde o afișare grafică a progresului curent și câteva funcționalități.

Printre funcționalitățile menționate anterior sunt ștergerea și editarea fie a alimentelor, fie a exercițiilor.

Prin ștergere se trimite această informație către baza de date și se realizează modificările specifice, iar pentru editare se deschide o alta pagină care primește alimentul respectiv ca parametru.

```
class DiaryScreen extends StatefulWidget {
  const DiaryScreen({Key? key}) : super(key: key);

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

class _DiaryScreenState extends State<DiaryScreen> {
  final Storage storage = Storage();
  User? user = FirebaseAuth.instance.currentUser;
  UserModel loggedInUser = UserModel();
  Diary diary = Diary();
  List<Meal> breakfastFood = [];
  List<Meal> lunchFood = [];
  List<Meal> dinnerFood = [];
  List<ExerciseData> exerciseDiary = [];

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

  Future<void> getData() async { ... }

  @override
  Widget build(BuildContext context) {
    final height = MediaQuery.of(context).size.height;
    final width = MediaQuery.of(context).size.width;

    final addFoodButton = Material( // Material ...
    final addExerciseButton = Material( // Material ...
    final finishButton = Material( // Material ...

    if (diary.food == null || loggedInUser.activitylevel == null)
      return Container( // CircularProgressIndicator // Center // Container ...
    else { ...
    }
  }

  void removeFood(String s, Meal dinnerFood) async { ...
  void removeExercise(ExerciseData exerciseDiary) async { ...
}
```

Fig. 20. diary_screen.dart

„Widgets” cuprinde widget-urile care sunt folosite în interiorul aplicației. Acestea au fie doar scop grafic (bara de progres circulară), fie scop grafic și funcționalitate (Date Picker). Date picker este un widget care este folosit în interiorul aplicației, acesta are ca scop selectarea unei date din calendar și afișarea acesteia pentru a putea fi folosită mai târziu.

```
class DatePickerWidget extends StatefulWidget {
  Color color;
  String userDate;
  Color buttonColor;
  TextEditingController dob;
  DatePickerWidget({Key? key ,required this.color,required
this.userDate,required this.buttonColor,required this.dob}): super(key:
key);
  @override
  _DatePickerWidgetState createState() => _DatePickerWidgetState(color :
color, userDate: userDate,buttonColor: buttonColor,dob: dob);
}

class _DatePickerWidgetState extends State<DatePickerWidget> {
  DateTime date = DateTime(0);
  Color color;
  String userDate;
  Color buttonColor;
  TextEditingController dob;
  _DatePickerWidgetState({required this.color,required
this.userDate,required this.buttonColor,required this.dob});

  String getText() {
    if (date.year == 0) {
      return userDate ;
    } else {
      return DateFormat('dd-MM-yyyy').format(date);
      // return '${date.month}/${date.day}/${date.year}';
    }
  }
  @override
  Widget build(BuildContext context) {...

  Future pickDate(BuildContext context) async {
    DateTime currentDate = DateTime.now();
    if(date.year !=0)
    {
      currentDate = date;
    }
  }
}
```

Fig. 21. date_picker_widget.dart

3.6. Food Recognition Model

La baza modelului antrenat stau modele de tip „Image Classification” care au ca scop gruparea unei poze sub un anumit nume.

O imagine reprezintă un aranjament de pixeli (puncte) puse într-o anumita ordine. Schimbarea ordinii sau culorii unui pixel duce și la schimbarea imaginii in sine.

Pentru ca calculatorul sa poate folosi această imagine acesta va trebui sa o spargă într-o matrice de pixeli care vor stoca codul de culoare ar pixelul de la poziția respectivă.

Aranjarea spațial trebuie păstrată atât pe plan vertical cât și pe plan orizontal. Greutățile vor fi reprezentate ca o matrice cu doua dimensiuni care vor lua pixeli împreună in direcția orizontală și verticală. Întrucât am luat in considerare atât direcția orizontală cât și cea verticală a greutateilor , rezultatul va fi cu un pixel mai mic pe ambele direcții.

Acest lucru îl face o rețea neuronală convoluțională. Din imaginea originală vor fi extrase caracteristici specifice fără a exista pierdere de informații despre aranjamentul imaginii.

Acest tip de rețea învață singură filtre fără a fi nevoie de menționări explicite. Aceste filtre ajută la extragerea de caracteristici din input.

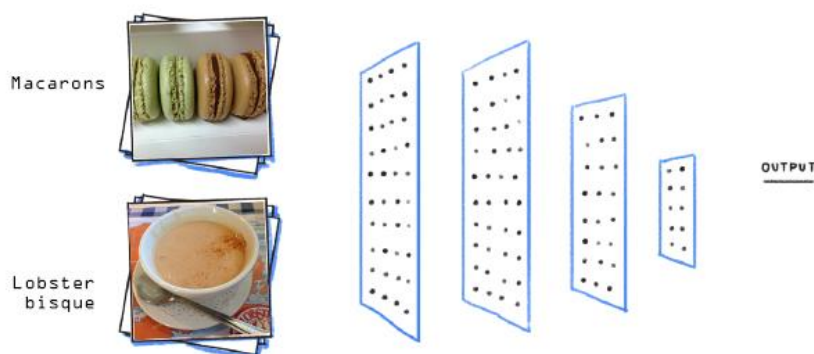


Fig. 22. Prezentare model rețea neuronală convoluțională

CNN captează aranjarea pixelilor și relația dintre aceștia dintr-o imagine. Aceste caracteristici ajuta la identificarea obiectului, locația și relația acestuia cu alte obiecte din imagine.

O rețea neuronală de acest tip are nevoie de 3 componente principale :

1. Un strat convoluțional :

În acest strat primim o imagine de o dimensiune și cu ajutorul unei matrice de greutate aplicăm un filtru asupra imaginii.

Primul pas este ca matricea de greutate să „treacă” peste imaginea primită drept input. Această „trecere” se realizează printr-o înmulțire a matricei de greutate cu imaginea. Imaginați-vă că matricea de greutate este o pensulă cu care se pictează un tablou. Pensula inițial pictează pe parte orizontală și tot coboară câte un nivel repetând pictatul pe orizontală. Valorile pixelilor sunt folosite din nou atunci când matricea de greutate se deplasează de-a lungul imaginii. Acest lucru permite practic partajarea parametrilor într-o rețea neuronală convoluțională.

Matricea de greutate se comportă ca un filtru într-o imagine care extrage anumite informații din matricea imaginii originale. O combinație de greutăți ar putea extrage margini, în timp ce alta ar putea avea o anumită culoare. În timp ce altul ar putea doar estompa zgomotul nedorit.

Ce a fost prezentat mai sus are dimensiunea de pas egală cu 1, deoarece ne plimbăm câte un pixel și sustragem o matrice egală cu dimensiunea matricei de greutate pornind de la acel pixel. Dacă dimensiunea pasului crește putem adăuga un strat de zerouri în jurul imaginii.

De reținut că adâncimea matricei de imagini trebuie să fie egală cu cea a matricei de greutate. În cazul modelului folosit adâncimea este de 3 deoarece codul de culoare al unui pixel este reținut cu ajutorul a 3 parametrii. Matricea rezultată după aplicarea unui filtru va avea adâncimea egală cu 1 (o singură dimensiune). Dacă se vor aplica mai multe filtre atunci numărul de matrice rezultate va fi egal cu numărul de filtre, iar acestea vor fi stivuite împreună.

2. Un strat pool

În unele cazuri imaginile au o dimensiune destul de mare și ar trebui să mai reducem numărul de parametrii antrenabili. Este important să se introducă aceste straturi de grupare între straturile convoluționale. Scopul acestora este de a reduce dimensiunea spațială a imaginii. Adunarea se face independent pe fiecare dimensiune de adâncime, prin urmare, adâncimea imaginii rămâne neschimbată.

De cele mai multe ori se aplică pooling maxim, după aplicarea acestuia imaginea încă păstrează informația inițială dar dimensiunea este redusă la jumate.

Cum se observă în exemplu :

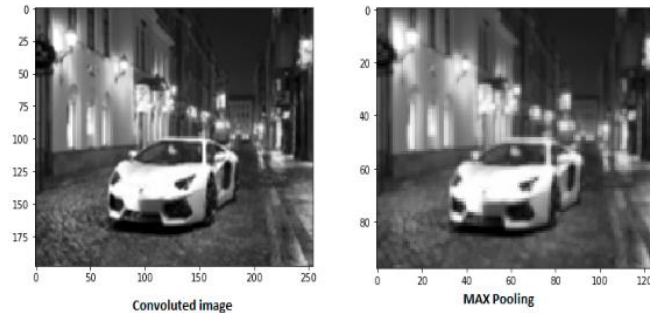


Fig. 23. Transformare imagine după aplicare pooling

3. Un strat de ieșire

Dimensiunile stratului de ieșire sunt controlate de 3 parametrii:

- Numărul de filtre - adâncimea volumului de ieșire va fi egală cu numărul de filtre aplicate.
- Pasul - când avem un pas de unu, ne mișcăm orizontal și în jos cu un singur pixel. Un pas mai mare duce implicit la un volum de ieșire mai mic.
- Umplerea cu zerouri (padding) - dacă adăugăm un strat de zerouri când realizăm o filtrare se va păstra dimensiunea imaginii inițială.

Pentru a calculat dimensiunea de ieșire putem folosi formula $([W-F4-2P]/S)+1$.

W - dimensiunea volumului de intrare

F - dimensiunea filtrului

P - numărul de umplutură aplicată

S - numărul de pași

După mai multe straturi de convoluție și umplutură, am avea nevoie de rezultat sub forma unei clase. Straturile de convoluție și de grupare ar putea doar să extragă caracteristici și să reducă numărul de parametri din imaginile originale. Cu toate acestea, pentru a genera ieșirea finală, trebuie să aplicăm un strat complet conectat pentru a genera o ieșire egală cu numărul de clase de care avem nevoie. Straturile de convoluție generează hărți de activare 3D, în timp ce avem nevoie doar de rezultat, indicând dacă o imagine aparține sau nu unei anumite clase. Stratul de ieșire are o funcție de pierdere, cum ar fi entropia încrucișată categorială, pentru a calcula eroarea în predicție.

Odată ce trecerea înainte este completă, propagarea înapoi începe să actualizeze greutatea și prejudiciile pentru reducerea erorilor și a pierderilor.

În cazul modelului „Food Recognition” avem trei grupuri de câte un strat convoluțional și unul „pool” (max pooling). După aceste grupări urmează un strat dropout.

Stratul dropout setează aleatoriu unitățile de intrare la 0 cu o frecvență a ratei la fiecare pas în timpul antrenamentului, ceea ce ajută la prevenirea overfitting-ului.

După aceste strat urmează un strat de aplatizare urmat de un strat dens. Stratul final este și cel de output care are dimensiunea egală cu numărul de clase.

```
Model: "sequential"
-----
Layer (type)                Output Shape              Param #
-----
conv2d (Conv2D)             (None, 224, 224, 32)      896
-----
max_pooling2d (MaxPooling2D) (None, 112, 112, 32)      0
-----
conv2d_1 (Conv2D)           (None, 112, 112, 32)      9248
-----
max_pooling2d_1 (MaxPooling2 (None, 56, 56, 32)        0
-----
conv2d_2 (Conv2D)           (None, 56, 56, 64)        18496
-----
max_pooling2d_2 (MaxPooling2 (None, 28, 28, 64)        0
-----
dropout (Dropout)           (None, 28, 28, 64)        0
-----
flatten (Flatten)           (None, 50176)             0
-----
dense (Dense)                (None, 128)               6422656
-----
dense_1 (Dense)              (None, 6)                 774
=====
Total params: 6,452,070
Trainable params: 6,452,070
Non-trainable params: 0
```

Fig. 24. Rețeaua Neuronală

Ca optimizator am folosit optimizatorul Adam cu un learning rate foarte mic ($lr = 0.000001$). Stratul de ieșire are o funcție de pierdere (entropia încrucișată categorială) pentru a calcula eroarea în predicție.

După ce se realizează un drum înainte complet, propagarea înapoi începe actualizarea greutăților și bias-urilor pentru reducerea pierderilor și erorilor.

Întreaga antrenare a fost realizată cu ajutorul Kaggle.

Kaggle este o filială a Google care funcționează ca o comunitate pentru oamenii de știință și dezvoltatori. Cei interesați de învățarea automată sau de alte tipuri de dezvoltare modernă se

Urmă Tudor-Irinel

Fiteat : aplicație mobilă pentru un stil de viață sănătos

pot alătura comunității de peste 1 milion de utilizatori înregistrați și pot vorbi despre modele de dezvoltare, să exploreze seturi de date sau rețea din 194 de țări separate din întreaga lume.

Așadar am folosit Kaggle pentru datele oferite , folosite la antrenare(poze alimente), și pentru Notebook-ul pe care acesta îl oferă.

3.5. News API

Acest API a fost folosit cu scopul de a extrage informații din zona „health”. Ca și cuvânt cheie am folosit cuvântul „food” pentru a extrage știri ce au legătură cu alimentația. Acest API oferă știri de actualitate, iar aceste știri vor fi oferite utilizatorului.

4. Paginile aplicației

4.1. Înregistrare

Această pagină îi oferă utilizatorului opțiunea de a-și crea un cont pentru ca mai apoi să utilizeze aplicația. Fiecare „greșală” a utilizatorului este alertată printr-un mesaj sugestiv.

Aceste „greșeli” pot fi :

- Loc liber în zonele unde se cer informații
- Introducerea unei informații invalide(ex: mail invalid)
- Parolele nu coincid

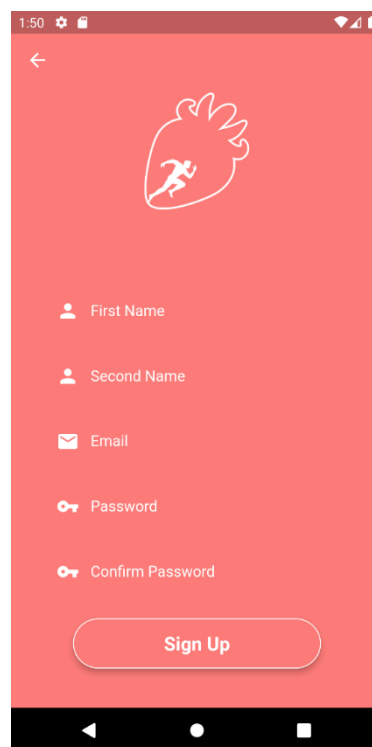


Fig. 25. Pagină de Înregistrare

4.2. Autentificare

Această pagina se ocupă cu autentificarea utilizatorului deja înregistrat. Dacă utilizatorul nu are cont poate apăsa pe opțiunea de „Sign up”, iar dacă acesta și-a uitat parola poate să o schimbe folosind opțiunea de „Forgot password”.

Similar cu pagina anterioară utilizatorul este anunțat când a făcut o „greșeală” ca :

- Email invalid
- Parolă invalidă
- Email/parolă greșită

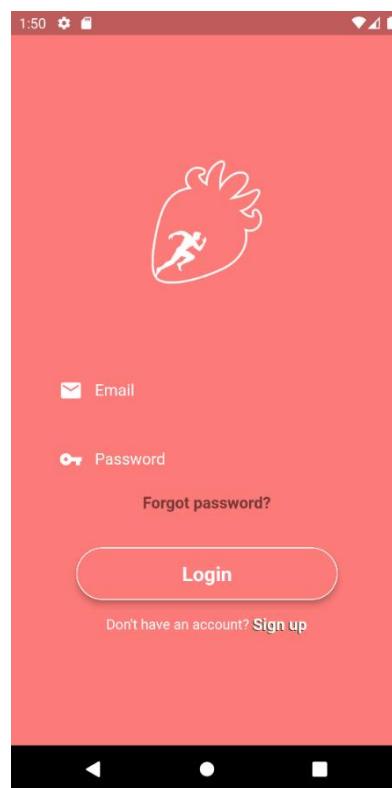


Fig. 26. Pagină de autentificare

4.2.1. Schimbare parolă uitată

Scopul acestei pagini este de a oferi utilizatorului șansa de a-și recupera parola pierdută. După ce acesta apasă pe butonul de resetare va primi un mesaj pe email unde va putea să își reseteze parola.

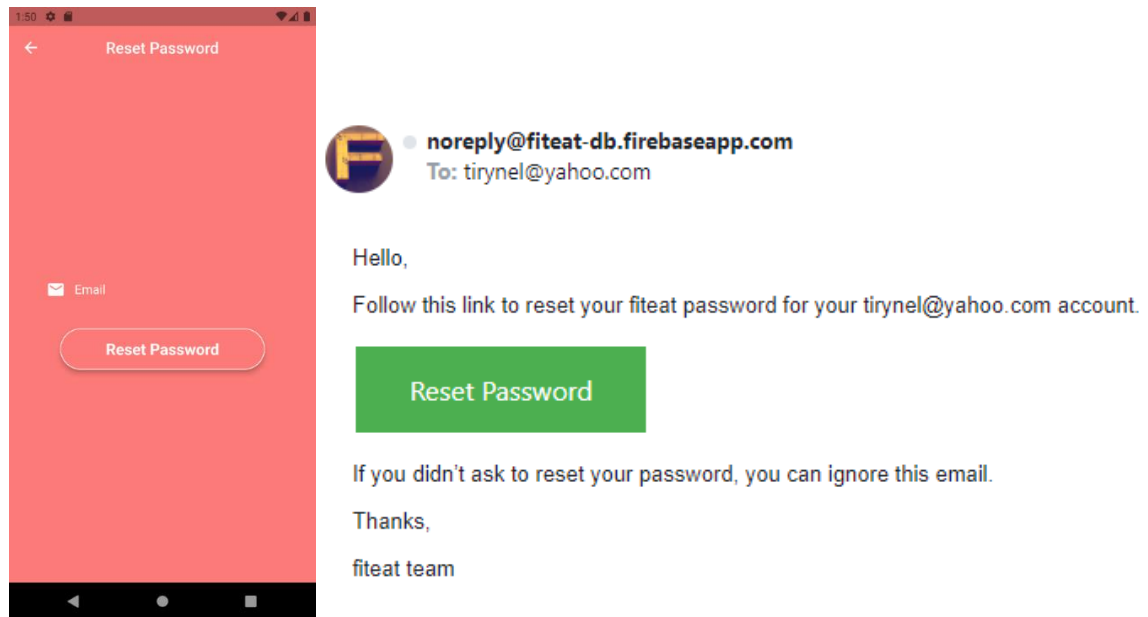


Fig. 27. Pagină de resetare parolă și model email de resetare

4.3. Setare detalii și scop

Această pagina apare după ce un utilizator s-a înregistrat. Aici utilizatorul își setează informații legate de el ca și scopul pe care dorește să îl aibă cu această aplicație.

Aplicația preia aceste informații, iar apoi îi oferă utilizatorului o sugestie cu privire la caloriile și macronutrienții pe care acesta ar trebui să îi consume.

Dacă utilizatorul nu completează corect acest formular va primi atenționări și sugestii pentru a repara „greșeala”.

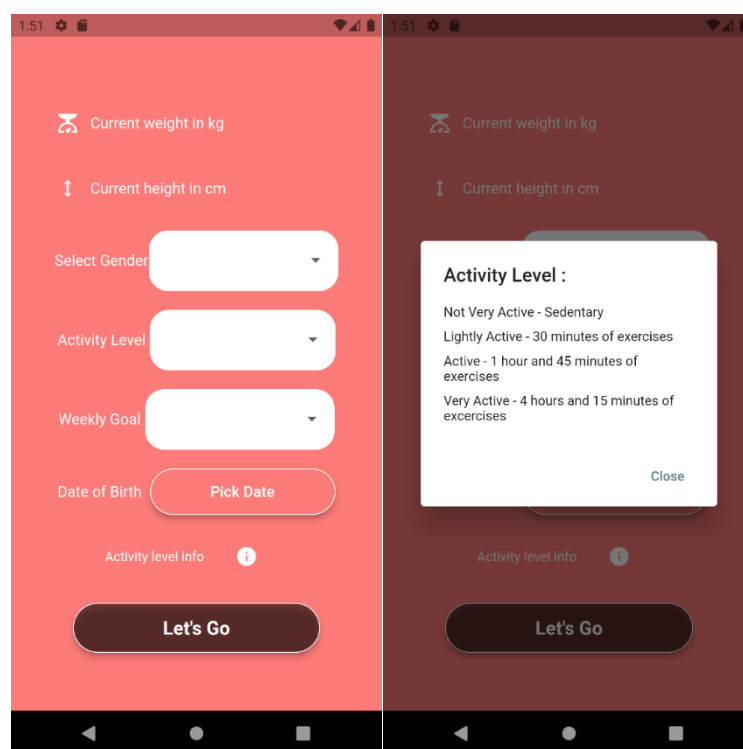


Fig. 28. Pagină de setare detalii și scop

4.4. Acasă

Aceasta este prima pagină cu care interacționează utilizatorul după ce s-a autentificat.

Cum se poate vedea și în imagine pagina este împărțită în 3 părți :

- Informații - aici găsim data curentă, poza de profil și progresul
- Știri - știri interesante din domeniu
- Bara de navigare – zona unde putem naviga către altă pagina

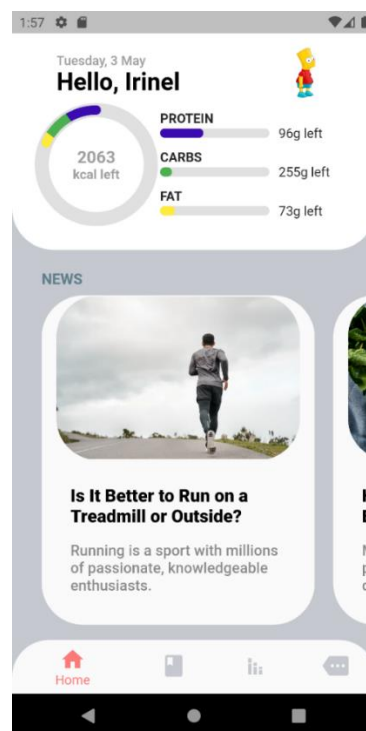


Fig. 29. Pagină acasă

4.4.1. Știri din baza de date

După accesarea unei știri aceasta se va deschide într-un format similar cu poza de mai jos, pentru a ne plimba prin informație avem opțiunea de scroll . Când am terminat de citit apăsăm pe săgeata din stânga sus pentru a reveni la pagina „Acasă”.

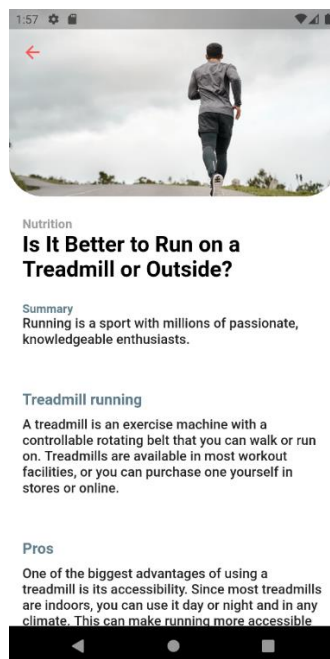


Fig. 30. Pagină de știri aflate in baza de date

4.4.2. Știri oferite de către News API

Acest tip de știre va avea prezentat doar un sumar in interiorul aplicației, iar dacă utilizatorul dorește să citească mai mult despre aceasta știre o va putea face acționând butonul „Read More...” care îl va direcționa pe pagina articolului.

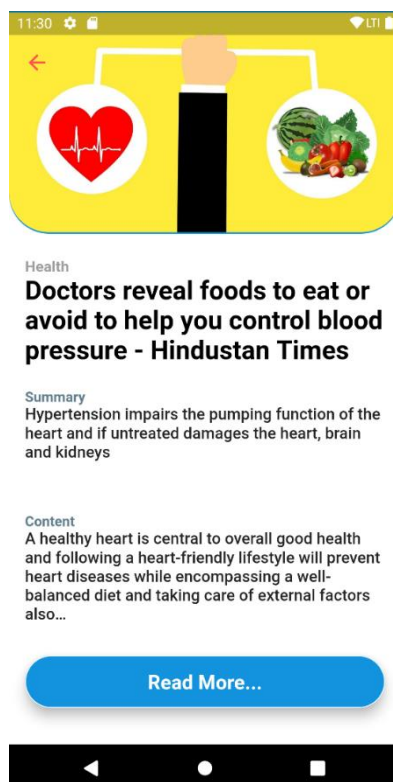


Fig. 31. Pagină de știri extrasă cu ajutorul API-ului News API

4.5. Jurnal caloric

Pagina cea mai complexă și importantă din această aplicație este pagina de jurnal caloric.

Aici ne putem observa progresul și cât mai avem nevoie pentru a-l completa.

Fiecare aliment și exercițiu adăugat în jurnal va fi prezentat aici împreună cu informațiile specifice. Putem șterge sau edita informații legate de una dintre acestea printr-o mișcare spre stânga sau dreapta.

Se observă în partea de jos 3 butoane cu scopuri specifice.

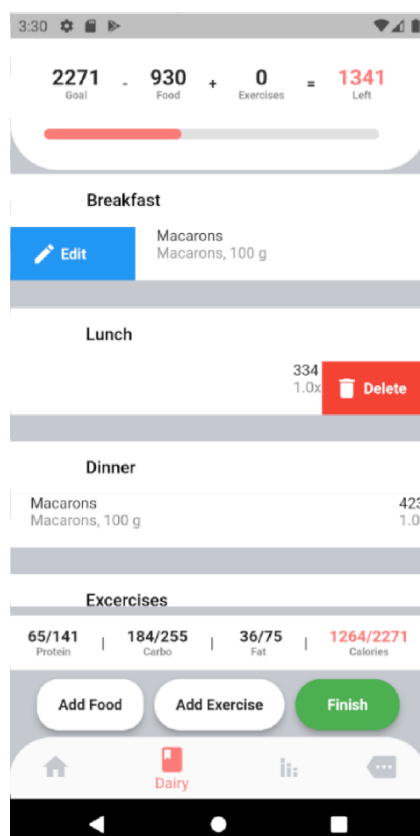


Fig. 32. Pagină de jurnal caloric

4.5.1. Adăugare aliment

La această pagină se ajunge prin apăsarea butonului „Add Food”. Scopul acestei pagini se regăsește și în nume, adăugarea de alimente în jurnalul caloric.

Utilizatorul are 5 metode prin care poate adăuga un aliment :

1. Scanare cod de bare – utilizatorul scanează un codul de bare al alimentului, iar apoi există 2 cazuri :
 - a. Alimentul este înregistrat și utilizatorul îl poate folosi
 - b. Alimentul nu este înregistrat și i se oferă utilizatorului opțiunea de a-l crea pentru a-l folosi mai târziu fie el fie alți utilizatori
2. Adăugare rapidă – utilizatorul știe exact câte calorii are alimentul pe care l-a mâncat și dorește să se grăbească cu adăugarea caloriilor
3. Fotografiere – utilizatorul fotografiază alimentul pe care l-a consumat, iar apoi așteaptă ca aplicația să îl recunoască ca mai apoi să îi ofere o sugestie. Dacă sugestia este corectă utilizatorul poate adăuga alimentul în jurnal.
4. Căutare - utilizatorul caută alimentul dorit fie prin scroll, fie cu ajutorul barei de căutare, iar mai apoi selectează alimentul dorit
5. Creare aliment – utilizatorul nu găsește alimentul dorit așa că îl poate adăuga manual pentru a-l folosi el sau alți utilizatori ai aplicației.

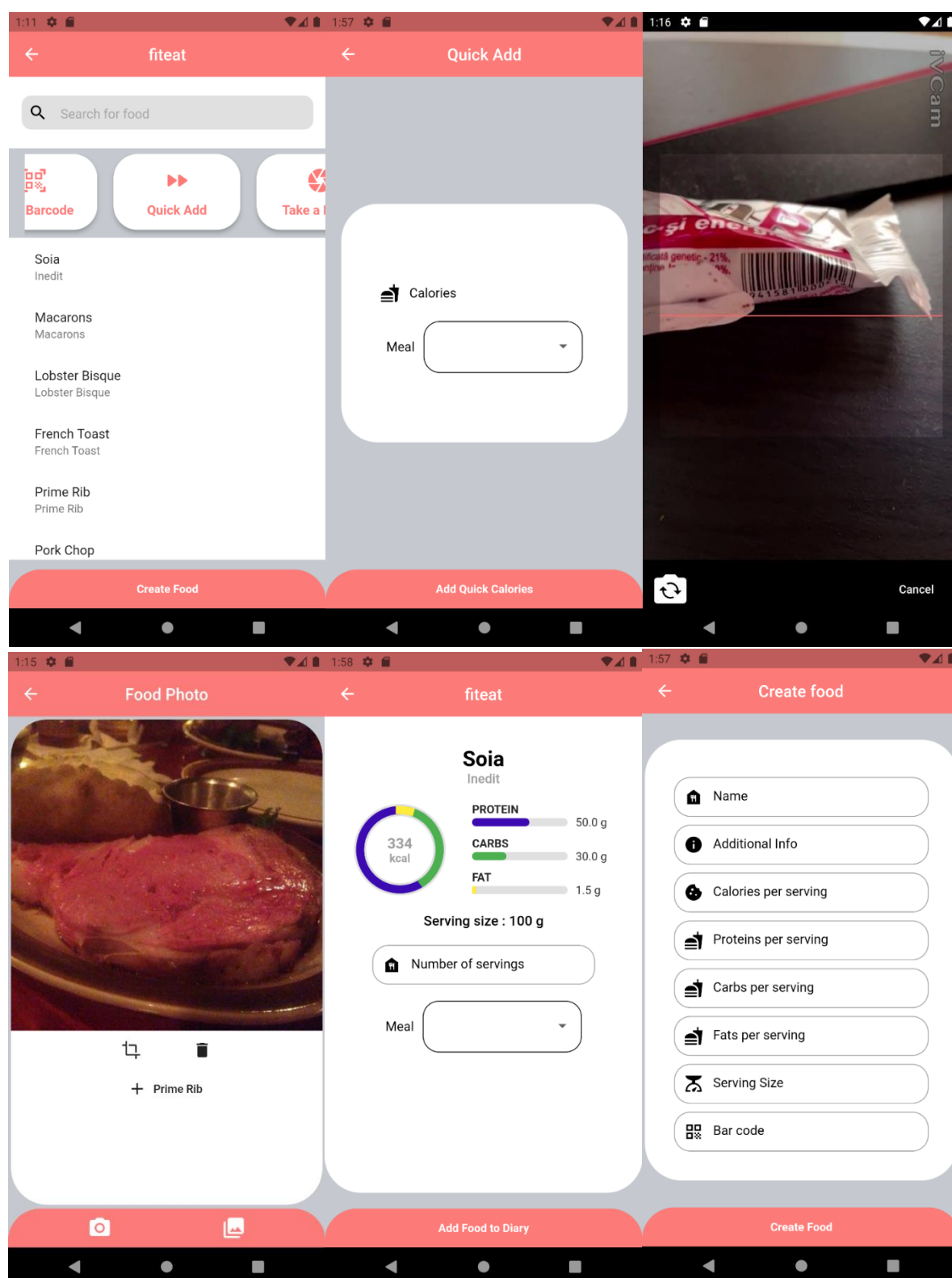


Fig. 33. Pagini pentru adăugare mâncare

4.5.2. Adăugare exercițiu

Această pagină are scop similar cu cea anterioară doar că aceasta se ocupă de exerciții.

Metode de adăugare a unui exercițiu :

- Adăugare rapidă - utilizatorul știe exact cate calorii a consumat realizând exercițiul și adaugă rapid cu aceasta opțiune
- Căutare - utilizatorul caută exercițiul dorit fie prin scroll, fie cu ajutorul barei de căutare, iar mai apoi îl selectează
- Creare exercițiu - utilizatorul nu găsește exercițiul dorit așa că îl poate adăuga manual pentru a-l folosi el sau alți utilizatori ai aplicației.

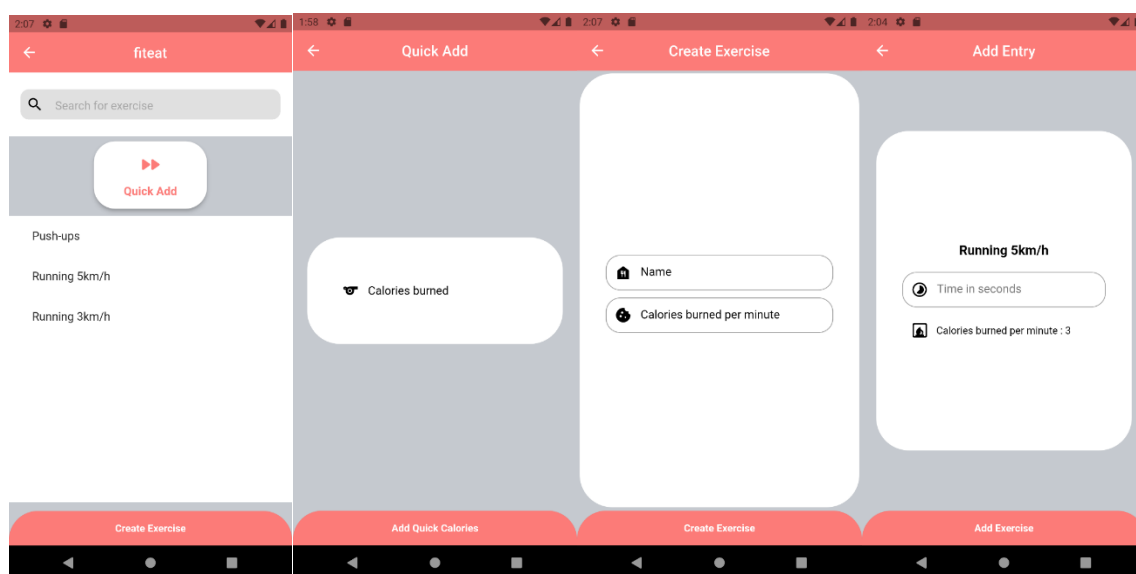


Fig. 34. Pagini pentru adăugare exercițiu

4.5.3. Editare Aliment

Scopul acestei pagini este editarea unui aliment deja existent în jurnal , se pot observa detalii curente și masa când a fost servit alimentul pentru a ajuta utilizatorul.

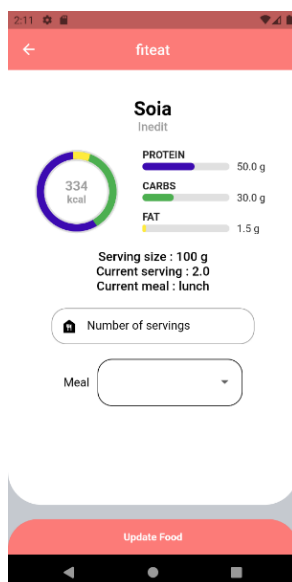


Fig. 35. Pagină pentru editarea unui aliment

4.5.4. Editare exercițiu

Scopul acestei pagini este editarea unui exercițiu deja existent în jurnal , se poate observa timpul curent petrecut pentru a ajuta utilizatorul.

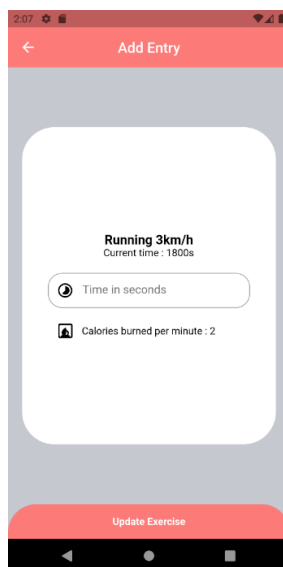


Fig. 36. Pagină pentru editarea unui exercițiu

4.5.5. Finalizare

Scopul acestei pagini este de a începe o nouă „pagină” a jurnalului, putem observa ca după finalizare se remarcă un mesaj fie de încurajare, fie de motivare a utilizatorului în funcție de progresul pe care acesta îl are.

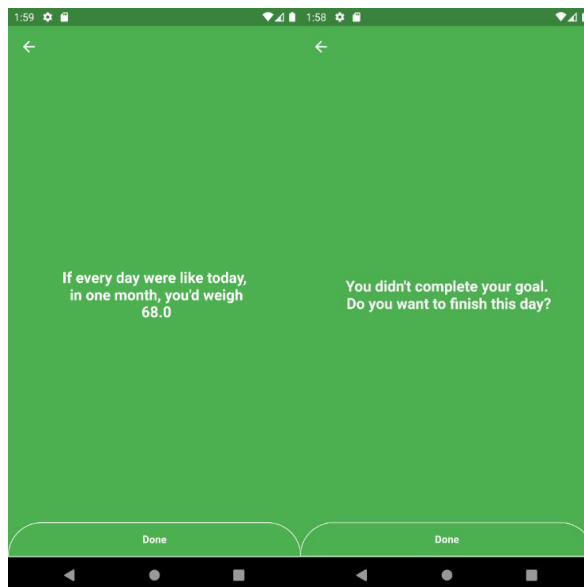


Fig. 37. Pagină pentru finalizare

4.6. Progres

Această pagină prezintă cu ajutorul unui grafic evoluția greutății utilizatorului de la crearea contului. Utilizatorul poate adăuga un nou progres acționând butonul „Add Progress”.

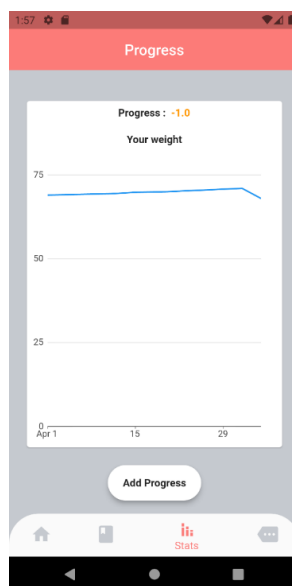


Fig. 38. Pagină pentru progres

4.6.1. Adăugare greutate curentă

O pagină simplă care are ca scop înregistrarea progresului curent al utilizatorului când vine vorba de greutatea acestuia.

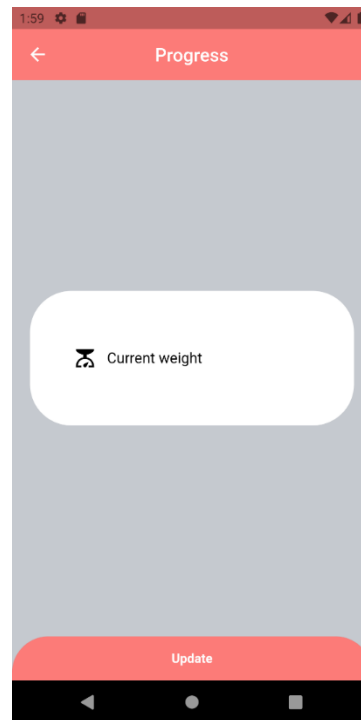


Fig. 39. Pagină pentru adăugare greutate curentă

4.7. Mai multe

În această pagină pot fi editate informații legate de contul utilizatorului.

Această pagină oferă 4 alegeri :

- Schimbare fotografie de profil
- Schimbare parola
- Schimbare scop/detalii
- Deconectare

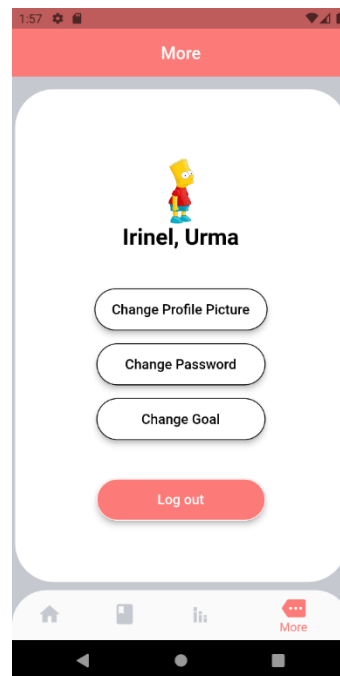


Fig. 40. Pagină „mai multe”

4.7.1. Schimbare poza de profil

Cu ajutorul acestei pagini utilizatorul își poate schimba fotografia de profil selectând o imagine din galeria foto.

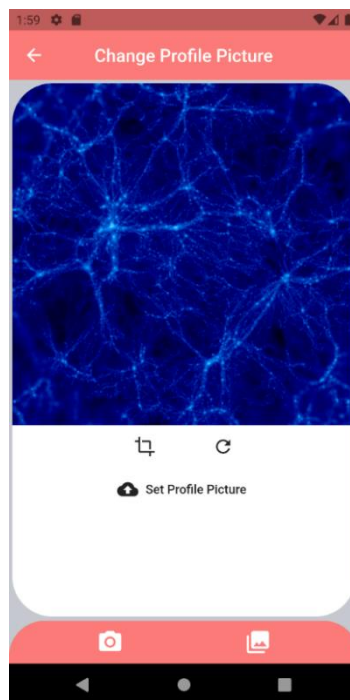


Fig. 41. Pagină pentru schimbarea pozei de profil

4.7.2. Schimbare detalii și/sau scop

Această pagină este împărțită în 2 segmente :

- Detalii și scop curent – aici utilizatorul poate observa detalii curente și poate trage o concluzie dacă dorește, sau nu, să modifice un anumit detaliu și/sau scop.
- Modificări - aici utilizatorul poate aduce modificări, după orice modificare caloriile utilizatorului vor fi recalculat în funcție de detalii schimbate.

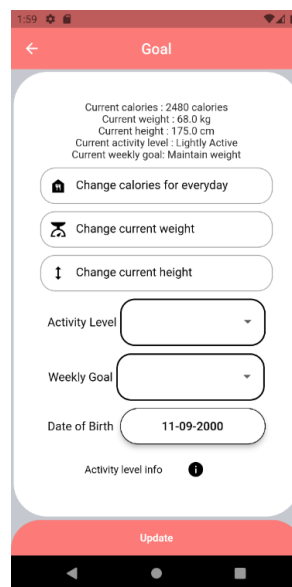


Fig. 42. Pagină pentru schimbare de detalii și/sau scop

4.7.3. Schimbare parolă

În această pagină i se oferă utilizatorului opțiunea de a schimba parola curentă.

În cazul unei greșeli (parolele nu coincid / parola invalidă) utilizatorul va fi anunțat.

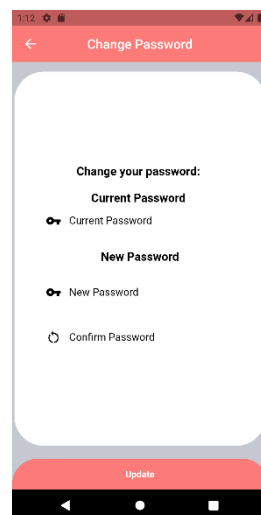


Fig. 43. Pagină pentru schimbare parolă curentă

5. Concluzii

Capitolele prezentate mai sus ne ajută să extragem concluzii, cât și o listă de avantaje și dezavantaje ale aplicației.

În această lucrare de licență a fost prezentată aplicația „fiteat” o aplicație tip jurnal caloric la care au fost adăugate mai multe funcționalități cu scopul de a ajuta utilizatorul care folosește aplicația.

Pornind de la o simplă idee , cea a jurnalului caloric, am ajuns la această aplicație care se transformă în acest ajutor pentru persoanele ce își doresc un stil de viață sănătos și echilibrat.

Această aplicație poate deveni ușor un produs pentru acest tip de oameni.

5.1. Avantaje

Unul din avantajele acestei aplicații este ca poate fi ușor folosită pe telefoane cu sistem de operare IOS și Android datorită framework-ului Flutter care oferă această posibilitate.

Zona de știri este un alt avantaj pentru utilizator , care poate găsi informații interesante despre acest domeniu. Informații care pot fi legate de antrenamente, de alimente sau chiar sugestii.

Partea de recunoaștere este destul de importantă și oferă utilizatorului o cale mai rapidă să își contorizeze alimentele consumate. Cu același scop este și partea de scanare a codului de bare și adăugarea rapidă prezentă atât la alimente cât și la partea de exerciții.

Statisticile prezentate pot ajuta utilizatorul prin motivarea acestuia să continue procesul sau sa mai aducă modificări planului. Decizie pe care utilizatorul o poate lua in funcție de fluctuația greutatei lui și scopul pe care acesta îl setase.

Faptul că aplicația este una mobila ajută utilizatorul să poate modifica informații din jurnal foarte rapid întrucât oricine in perioada aceasta deține un telefon smart.

5.2. Dezavantaje

Dezavantajul principal pe care îl putem observa este lipsa istoricului despre care am vorbit și la partea de îmbunătățiri.

Un alt dezavantaj este formula pe care aplicația o folosește în calculul kaloriilor de care are nevoie utilizatorul deoarece aceasta este o formula generală și nu se aplică perfect pe toate cazurile.

6. Îmbunătățiri posibile ale aplicației

6.1. Îmbunătățirea modelului pentru recunoașterea mâncării

Pornind de la modelul deja antrenat pot fi adăugate modificări astfel încât acesta să poată recunoaște mai multe alimente.

Când vine vorba de recunoașterea unei imagini și catalogarea acesteia va fi întotdeauna un element care poate fi îmbunătățit.

Fiecare parte din antrenarea unui model este importantă, atât rețeaua neuronală aleasă cât și elementele trimise drept input. Din căutările mele am reușit să găsesc un dataset ce conține 101 de alimente fiecare aliment având asigurate un număr de 1000 de poze din care putem alege.

Notebook-ul oferit de către Kaggle nu reușește să facă față la un număr așa mare de date, iar modelul antrenat de către mine poate recunoaște 6 alimente.

Referitor strict la acest model pot fi îmbunătățite :

- Procentajul de succes
- Numărul de alimente pe care le poate recunoaște

6.2. Istoric jurnal

Ar fi foarte interesant ca utilizatorul să poată verifica cum s-a descurcat în această săptămână, lună sau chiar an. Sau poate doar își dorește ca își realizeze o masă cu ceva ce a mâncat cu o zi înainte și nu își mai aduce aminte.

Ca soluție la aceste cerințe ar fi adăugarea unui istoric unde utilizatorul se poate „plimba” și își poate acoperi nevoile.

6.3. Asocierea cu alte aplicații

Această opțiune o consider destul de importantă. O asociere între aplicația „fiteat” și o alta aplicație poate aduce beneficii ca :

- Rapiditate în adăugarea exercițiilor - spre exemplu cei de la Under Armour au realizat o aplicație în care îți poți adăuga exercițiile realizate în ziua respectivă, iar mai apoi aplicația „fiteat” poate prelua aceste informații de la această aplicație. Un alt exemplu ar fi aplicațiile ce contorizează pașii, de unde, aplicația „fiteat” poate lua numărul de pași și cu ajutorul unei formule poate adăuga numărul de calorii pe care utilizatorul le-a consumat.
- Sincronizarea progresului cu celelalte aplicații - Similar cu aplicația de mai sus, putem primi de la aplicații o înregistrare cu progresul curent și îl putem adăuga în aplicația noastră.

Bibliografie

- [1] https://www.alevia.com.ro/cum-sa-ai-un-stil-de-viata-sanatos/?gclid=CjwKCAjwve2TBhByEiwAaktM1C5fOhdIsegaOndq2lDEkGJ6fZkd3bItYpB8vUHDnbQ9YWxeQxsoJxoCWNMQAvD_BwE
- [2] <https://www.timis.dev/dart/beginner/introducere-in-dart>
- [3] <https://www.appify.digital/post/flutter-app-development>
- [4] <https://www.tristatetechnology.com/blog/firebase-backend-mobile-app/>
- [5] <https://www.pythonisti.ro/despre-limbajul-de-programare-python.php>
- [6] <https://ro.theastrologypage.com/kaggle>
- [7] <https://www.oracle.com/ro/database/nosql/what-is-nosql/>
- [8] <https://firebase.google.com/docs>
- [9] <https://courses.analyticsvidhya.com/courses/take/convolutional-neural-networks-cnn-from-scratch/>