Универзитет „Св. Кирил и Методиј“ во Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Мобилни информациски системи

Систем за изгубено–најдено

Дејан Станчевски *211027*
Берна Хоџин *211001*

Септември, 2025

Содржина

1. Вовед

Во секојдневниот студентски живот често се случува да се изгубат или најдат предмети како што се студентски картички, лични документи, УСБ-уреди или пак лични предмети од секојдневна употреба. Често процесот на нивно пронаоѓање е тежок, бидејќи недостасува централизирано место каде студентите можат да пријават изгубени или најдени предмети.

Токму затоа е создадена мобилната апликација „Изгубено & најдено“ (Lost & Found), која е наменета за студентите на ФИНКИ. Апликацијата овозможува едноставен начин за пријавување и прегледување на изгубени и пронајдените предмети. Со тоа се скратува времето на барање и се зголемува шансата предметите да бидат вратени кај нивните сопственици.

Оваа апликација претставува практичен и иновативен чекор кон дигитализација на овој процес, овозможувајќи им на студентите едноставна платформа која е секогаш достапна преку мобилните уреди.

2. Опис на апликација

Апликацијата „Изгубено & најдено“ е изработена со Dart и Flutter и е насочена кон сите уреди. Корисничкиот интерфејс е едноставен, чист и функционален, со цел да им овозможи на студентите лесна навигација и брз пристап до сите потребни функции.

На самиот почеток, корисникот мора да се најави или регистрира преку својата е-пошта и лозинка. Ова овозможува персонализирано искуство и пристап само до оние опции кои се дозволени за конкретниот корисник (како што е уредување или бришење на сопствено пријавени предмети).

По успешната најава, корисникот е пренасочен кон главниот екран на апликацијата кој е поделен на две основни секции (табови):

- Lost (Изгубено) – овде се прикажуваат сите предмети кои се пријавени како изгубени.
- Found (Најдено) – овде се прикажуваат сите предмети кои се пријавени како најдени.

Секој пријавен предмет содржи:
- Име (назив на предметот)
- Опис
- Место каде што е изгубен/најден
- Датум каде што е изгубен/најден
- Категорија
- Слика

- Boolean вредност која означува дали предметот е предаден
- Корисник (пријавувач)

Доколку корисникот кликне на некој предмет, се отвора Details екран, кој овозможува повеќе информации и дополнителни функционалности.

3. Функционалности

Апликацијата вклучува широк спектар на функционалности кои ја прават корисна и практична за секој студент:

3.1. Најава и регистрација

Секој корисник мора да има свој профил.

Регистрацијата се врши преку е-пошта и лозинка.

При најавување, корисникот има пристап до сите пријавени изгубени и најдени предмети.

3.2. Прегледување предмети

Во табот Lost се прикажуваат сите изгубени предмети.

Во табот Found се прикажуваат сите најдени предмети.

Секој предмет се прикажува со сите релевантни информации

3.3. Details екран

При кликнување на предмет, корисникот може да го види целиот опис и да стапи во контакт со пријавувачот.

Доколку предметот е пријавен од истиот корисник, тој може да го уредува, избрише или да ја промени состојбата на handover (дали предметот е предаден).

Кога состојбата е означена како предаден предмет, тој веќе не се прикажува во јавната листа.

3.4. Совпаѓања на предмети (Match finder)

Апликацијата нуди систем за автоматско пронаоѓање на совпаѓања.

Ако некој изгубен предмет има сличност со некој најден, тоа се прикажува во секцијата „Matching items“ во Details екранот.

3.5. Додавање нов предмет

Корисникот може да додаде предмет со кликнување на иконата „+“.

Ако е во табот Lost, предметот автоматски ќе биде обележан како изгубен.

Ако е во табот Found, предметот ќе биде обележан како најден.
При додавањето, корисникот може да ја промени категоријата ако е потребно.

3.6. Филтрирање предмети

Предметите може да се филтрираат според:
- Датум на пријавување
- Име
- Категорија

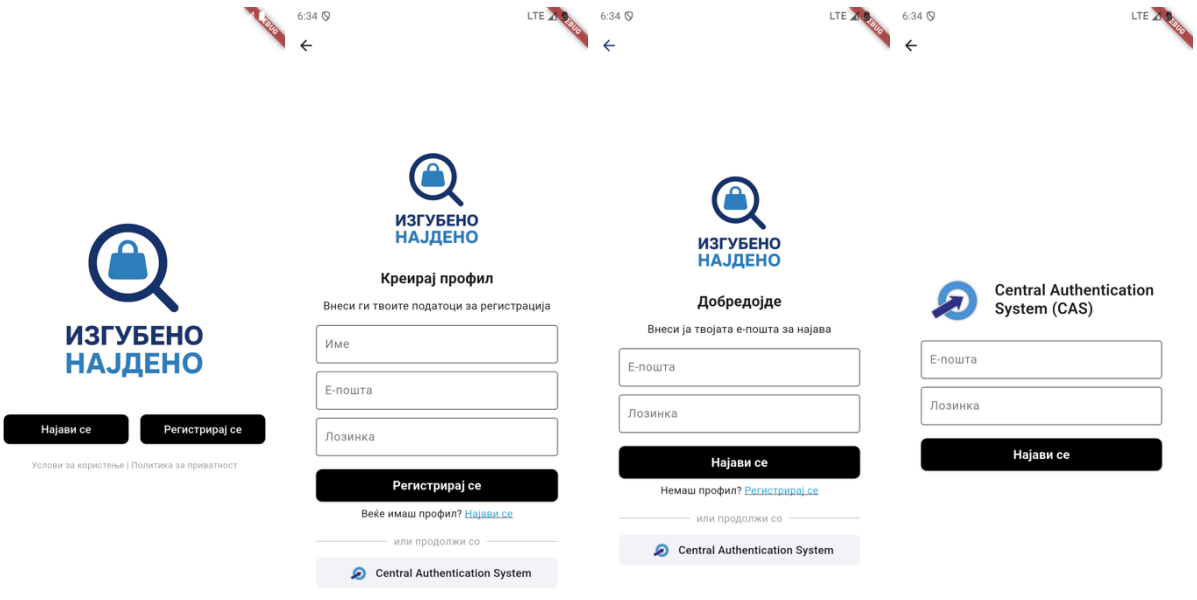
Покрај филтрирањето, постои можност и за сортирање на предметите според:
- Растечки или опаѓачки редослед по датум
- Азбучен редослед (A–Z или Z–A) според името на предметот

Оваа функционалност овозможува студентите полесно и побрзо да го најдат предметот што го бараат, без разлика дали пребаруваат по датум, име или категорија.

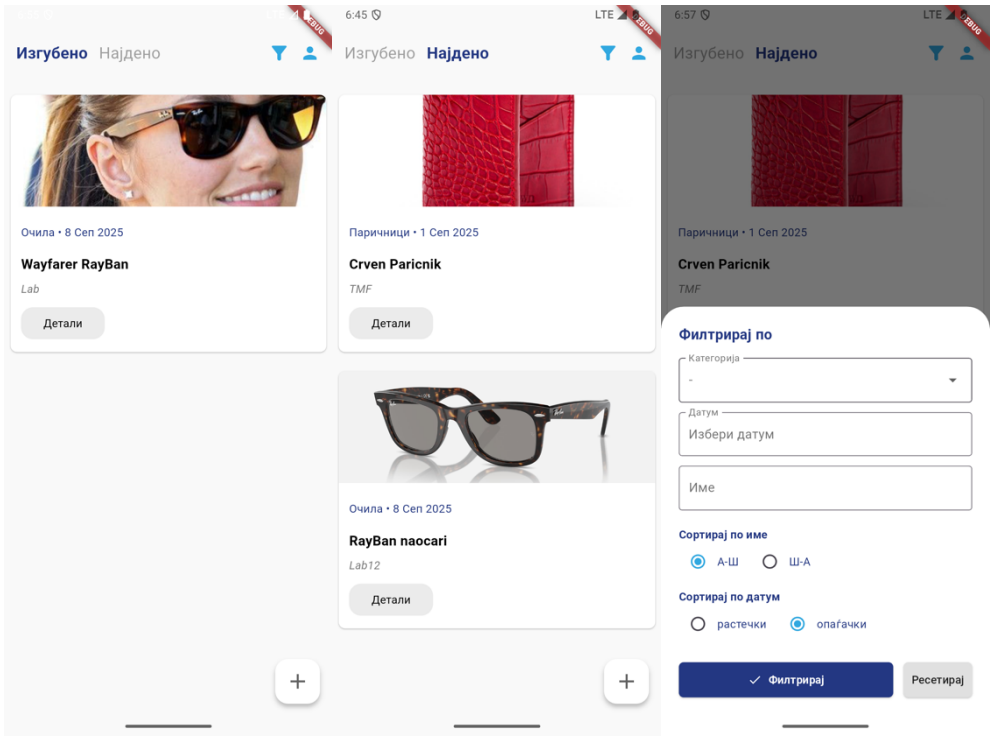
3.7. Примери од кориснички интерфејс

Апликацијата има едноставен, интуитивен и функционален интерфејс, прилагоден за студентите. Следниве слики и описи ги илустрираат клучните екрани:

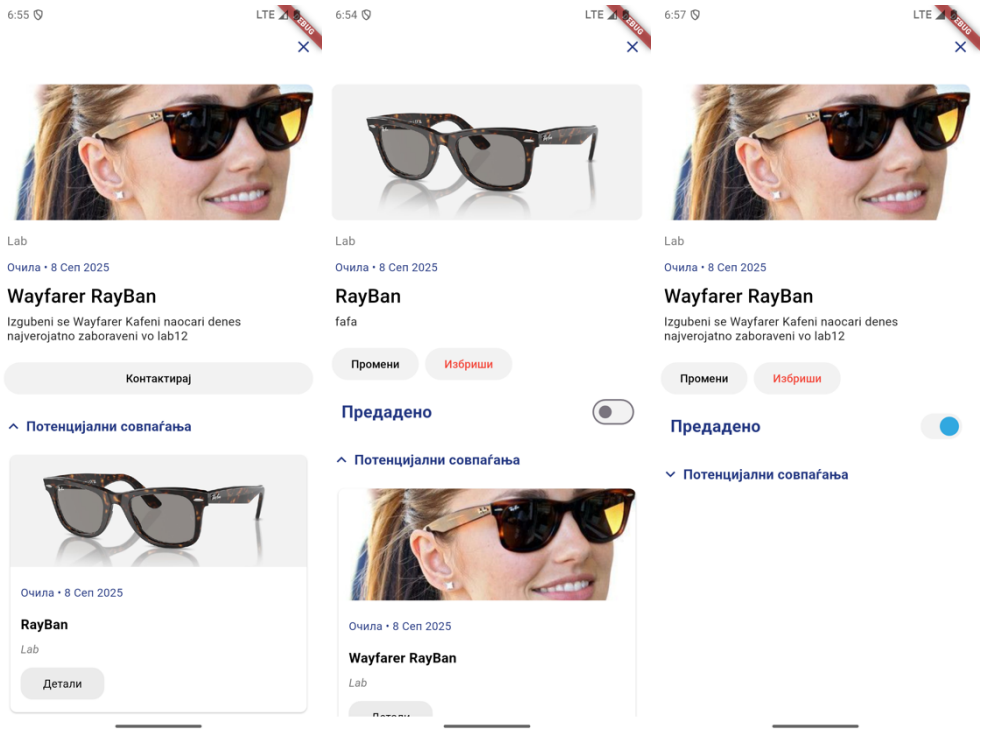
3.7.1. Почетен екран, регистрација, најава и CAS најава



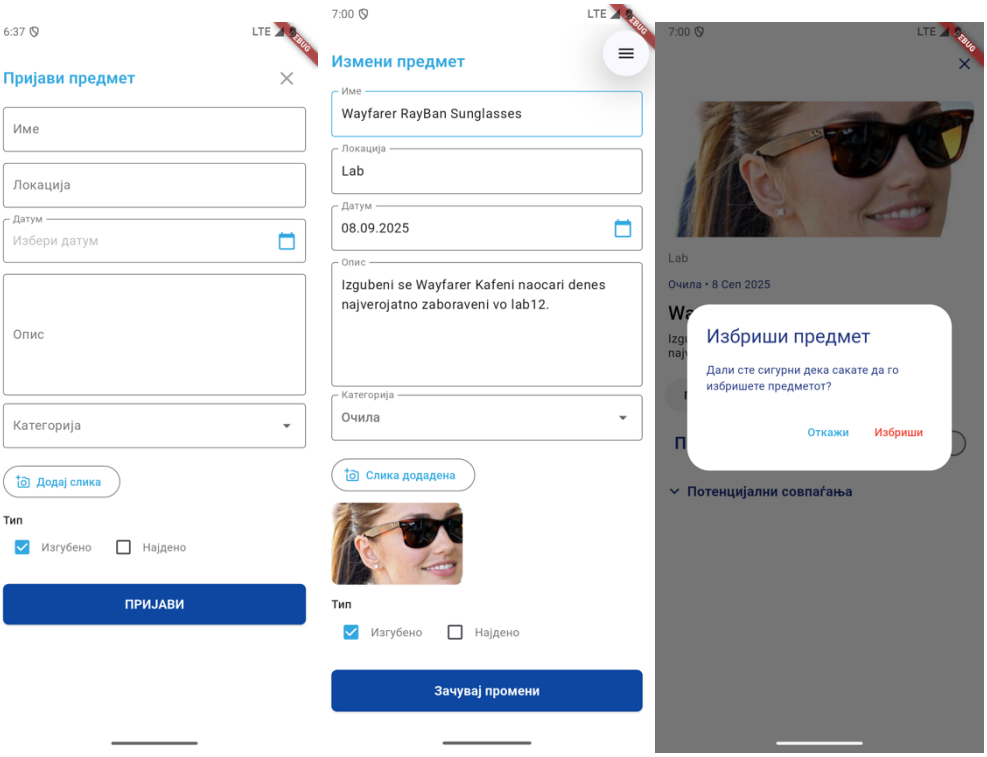
3.7.2. Главен екран и филтрирање



3.7.3. Детали



3.7.4. Додавање, измена и бришење



4. Технологии

Апликацијата е крос-платформска мобилна апликација изградена со Flutter и Dart. Главната цел е корисниците да можат да пријавуваат изгубени и пронајдени предмети, да пребаруваат и филтрираат предмети, како и да контактираат сопственици на предмети.

4.1. Архитектура

MVVM + StateNotifier (Riverpod) модел:

- **Models:**
  Item, Category, ItemType, state класи (AuthState, HomeScreenState, ItemFormState) за управување со UI.

```
class Item {
  final String id;
  final String name;
  final Category category;
  final String description;
  final DateTime dateIssueCreated;
  final String location;
  final ItemType type;
  final String? imageUrl;
  final bool handover;
  final String createdBy;

  Item({required this.id, required this.name, required this.category, ...});

  factory Item.fromMap(String id, Map<String, dynamic> data) { ... }
  Map<String, dynamic> toMap() { ... }
  Item copyWith({String? name, Category? category, ...}) { ... }
}

enum Category { MOBILE_PHONES("Мобилни телефони"), LAPTOPS("Лаптопи"), ... }
enum ItemType { LOST, FOUND }
```

- **State Management:**
  Riverpod за реактивно управување со state.
  StateNotifier за сложени state логики.
  StreamProvider.family за динамичко филтрирање и сортирање на листата на предмети.

- **Services:**
  AuthService – логика за автентикација (login, signup, logout).
  ItemService – CRUD операции за предмети и streaming на податоци.
  UserService – fetch на email за контактирање.

4.2. Автентикација

Firebase Authentication: Сигурен систем за најава и регистрација со email/password.

CAS (Central Authentication System): Идентитетна интеграција за едноставна проверка на корисникот.

Иако во UI делот има додадени екрани за најава и регистрација, тие засега работат преку Firebase Authentication. Ако постои потреба од идентификација преку универзитетскиот систем (CAS) за продукциско користење, оваа функционалност сè уште не е интегрирана и има простор да се поврзе со CAS за целосна идентитетна проверка на студентите.

```
class AuthService {
  final FirebaseAuth _auth = FirebaseAuth.instance;

  Future<void> login(String email, String password) async { ... }
  Future<void> signUp(String email, String password, String name) async { ... }
  Future<void> logout() async => _auth.signOut();
  Stream<User?> get authStateChanges => _auth.authStateChanges();
} // AuthService со Firebase
```

```
final authProvider = StateNotifierProvider<AuthNotifier, AuthState>(
  (ref) => AuthNotifier(AuthService()),
);

class AuthNotifier extends StateNotifier<AuthState> {
  Future<void> login(String email, String password) async { ... }
  Future<void> signUp(String email, String password, String name) async { ... }
  Future<void> logout() async { ... }
} // AuthNotifier со StateNotifier и Provider
```

4.3. Главен екран и управување со предмети

HomeScreen: табови „Изгубено“ и „Најдено“, филтрирање по категорија, датум и име, сортирање (A-Z, Z-A).

ItemFormModal: креирање и уредување на предмети, избор на слика, категорија и датум.

ItemDetailsScreen: детални информации за предмет, сопственички акции (Edit/Delete/Handover) и контакт со пријавувачот.

Floating Action Button (FAB) се користи за додавање на нов предмет.

4.4. Модели и податоци

Item: id, name, category, description, dateIssueCreated, location, type, imageUrl, handover, createdBy.

Enums: Category, ItemType, NameSortMode, DateSortMode.

```
enum Category {
  MOBILE_PHONES("Мобилни телефони"),
  LAPTOPS("Лаптопи"),
  TABLETS("Таблети"),
  ...
  OTHER("Друго");
```

```
  final String displayName;
  const Category(this.displayName);
}
```

Методи: fromMap(), toMap(), copyWith() за лесно управување со state и immutability.

4.5. State management

Во апликацијата се користи Riverpod како state management библиотека, со StateNotifier за управување со посложени состојби. Подолу се претставени клучните провајдери и нивната намена:

Provider	*State*	*Опис*
authProvider	AuthState	Управување со login, signup, logout
homeScreenProvider	HomeScreenState	Избран таб, филтри и сортирање
itemFormProvider	ItemFormState	Логика за форма на предмет, контролери, селекции и статус
itemsListProvider	Stream<List<Item>>	Stream на предмети филтрирани и сортирани

4.5.1. authProvider – Управување со автентикација

Овој провајдер управува со login, signup и logout функционалноста преку FirebaseAuth.

```
final authProvider = StateNotifierProvider<AuthNotifier, AuthState>(
  (ref) => AuthNotifier(AuthService()),
);
```

AuthState чува: тековниот корисник (User?), дали се вчитува нешто (isLoading) и евентуални грешки (error).

AuthNotifier се грижи за логика: најава, регистрација, одјава, и автоматско слушање на промените во Firebase.

Благодарение на ова, UI може лесно да се „реактивира“ кога состојбата на корисникот ќе се промени.

4.5.2. homeScreenProvider – Управување со табови и филтри

```
final homeScreenProvider = StateNotifierProvider<HomeScreenNotifier, HomeScreenState>(
  (ref) => HomeScreenNotifier(),
);

class HomeScreenNotifier extends StateNotifier<HomeScreenState> {
```

```
  void setSelectedTab(ItemType tab) { state = state.copyWith(selectedTab: tab); }
  void setFilters({Category? category, DateTime? date, String? name, ...}) { ... }
}
```

HomeScreenState чува информации за: кој таб е активен (Lost или Found), избрани филтри (категорија, датум, име), како и начин на сортирање.

HomeScreenNotifier овозможува промена на активниот таб или сетирање/ресетирање на филтри.

Оваа логика овозможува динамично прилагодување на листата според критериумите на корисникот.

4.5.3. itemFormProvider – Управување со форма за предмети

```
final itemFormProvider = StateNotifierProvider<ItemFormNotifier, ItemFormState>(
  (ref) => ItemFormNotifier(),
);

class ItemFormNotifier extends StateNotifier<ItemFormState> {
  Future<void> submit(Item? initialItem, String userId) async { ... }
}
```

ItemFormState чува контролери за полињата на формата (име, локација, опис), избрана категорија, тип на предмет (Lost/Found), датум и слика.

ItemFormNotifier овозможува менување на секое од овие полиња, управување со upload на слика и логика за додавање/уредување предмети.

Ова ја прави формата реактивна – секоја промена веднаш се рефлектира во UI.

4.5.4. itemsListProvider – Stream со филтрирани предмети

```
final itemsListProvider = StreamProvider.family<List<Item>, ItemsFilterParams>(
  (ref, params) {
    return ItemService.itemsStream(params.type).map((items) {
      var filteredItems = items;
      if (params.category != null) { filteredItems = filteredItems.where((e) =>
e.category == params.category).toList(); }
      if (params.date != null) { filteredItems = filteredItems.where((e) =>
!e.dateIssueCreated.isBefore(params.date!)).toList(); }
      if (params.name != null && params.name!.trim().isNotEmpty) { ... }
      filteredItems.sort((a, b) { ... });
      return filteredItems;
    });
  },
);
```

Користи **StreamProvider.family** за да прикаже листа предмети врз основа на параметри (Lost/Found, категорија, датум, име).

Се користи **ItemService.itemsStream** кој враќа реални податоци од Firebase, а потоа се врши дополнително филтрирање и сортирање во меморија.

Ова обезбедува секогаш ажурирана листа на предмети, без потреба од рачно refresh-ирање.

4.6. UI / Компоненти

Custom Widgets: FormInput, PrimaryButton, ImagePickerButton, ItemCard, CasButton, LogoHeader.

Dialogs / Modals: EmailContactModal, ConfirmDeleteDialog, SortFilterBottomSheet.

Reusable Components: _ItemImage, _ItemHeader за детални екранови прикази.

```
ElevatedButton(
  onPressed: formState.isSubmitting ? null : () async {
    await formNotifier.submit(initialItem, userId);
    Navigator.of(context).pop();
  },
  child: Text(isEdit ? "Зачувај промени" : "ПРИЈАВИ"),
);
```

4.7. Навигација

Flutter Navigator: push, pushReplacement, pushAndRemoveUntil, pop.

Modal и **fullscreen** screens за форми и детали на предмети.

Navigation flow: SplashScreen → IndexScreen → Login/Signup → HomeScreen → ItemFormModal / ItemDetailsScreen.

```
Navigator.of(context).push(MaterialPageRoute(builder:              (_)              =>
ItemFormModal(preselectedType: ItemType.LOST)));
Navigator.of(context).pushAndRemoveUntil(
  MaterialPageRoute(builder: (_) => const IndexScreen()), (route) => false
);
```

5. Third services

За правилно функционирање на апликацијата „Изгубено & најдено“, искористени се повеќе надворешни сервиси кои овозможуваат сигурна работа со податоци, корисници и мултимедија.

5.1. Firebase Authentication

Сервис за автентикација кој обезбедува сигурна и лесна интеграција на логирање и регистрација во апликации. Со ова автентикацијата овозможува лична контрола врз содржината – само пријавувачот може да го уреди, избрише или да ја промени состојбата на сопствениот предмет. Секој корисник мора да се најави или регистрира за да има пристап до системот. Поддржано е логирање со е-пошта и лозинка. Со ова се обезбедува сигурност, бидејќи секој пријавен предмет е поврзан со конкретен корисник.

5.2. Firebase Realtime Database & Cloud Firestore

Firebase нуди два сервиси за работа со податоци – Realtime Database и Firestore. И двата се NoSQL бази на податоци кои обезбедуваат работа со структуриран JSON формат. Firestore овозможува понапредни пребарувања и филтрирања според категорија, датум или име.
Realtime Database овозможува веднаш да се ажурираат промените во реално време кај сите корисници. Со ова ги чуваме информациите за сите изгубени и најдени предмети, корисниците (ID, пријавени предмети, состојба на предметите). Исто така секое додавање или менување предмет е веднаш синхронизирано кај сите корисници.

5.3. Firebase Storage

Облак сервис наменет за складирање на мултимедијални содржини, како што се слики, видеа или документи. Сликите се клучни за полесно препознавање на предметите, а Firebase Storage нуди сигурен и брз начин за нивно чување и преземање. Сите слики од изгубените и најдените предмети се прикачуваат и складираат во Firebase Storage. Секоја слика е поврзана со конкретен предмет во базата на податоци преку линк.

5.4. Cloudinary

Сервис за обработка, оптимизација и управување со слики и видеа. Овој сервис овозможува сликите да бидат прикажани со висок квалитет, но истовремено да не го успоруваат апликацискиот перформанс. Оптимизација на слики за побрзо вчитување. Автоматско скалирање и прилагодување на сликите според уредот. Во нашата апликација овој сервис се користи бидејќи нуди бесплатен план, и сликите се хостирани на Cloudinary, наместо директно во Firebase Storage, што ја намалува оптовареноста на Firebase и го подобрува перформансот.

5.5. Image Picker

Flutter плагин кој овозможува лесна селекција на слики од галеријата или сликање со камера. Сликата е клучен елемент за секој предмет и значително го олеснува неговото идентификување. При додавање нов предмет, корисникот може да избере слика од уредот или да направи нова фотографија.

5.6. URL Launcher

Плагин кој овозможува отворање линкови или директно стартување на други апликации од мобилниот уред. Оваа функционалност ја прави комуникацијата меѓу студентите едноставна и директна, што е основна цел на апликацијата. Со помош на ова се остварува контактирање со пријавувачот на предмет преку телефон или е-пошта. Отворање надворешни линкови доколку се потребни.

5.7. Flutter Riverpod

Современ state management пакет за Flutter, кој овозможува лесно управување со податоци и состојби низ целата апликација. Го прави кодот поорганизиран и стабилен, со што се олеснува одржувањето и понатамошното надградување на апликацијата. Управување со состојби на предметите (дали се изгубени, најдени или предадени).Следење на состојбите при промени – пример: кога некој предмет се уредува или брише, сите промени веднаш се прикажуваат во интерфејсот.

6. Заклучок

Апликацијата „Изгубено & најдено“ претставува практично и современо решение за чест проблем во студентската заедница – губење и пронаоѓање на лични предмети. Со воведување на оваа апликација, студентите на ФИНКИ добиваат централизирана платформа каде што можат брзо да пријават изгубени предмети или пак да пријават доколку пронашле нешто што не е нивно.

Главната предност на апликацијата е тоа што целиот процес е едноставен, брз и транспарентен. Од технички аспект, апликацијата користи модерен стек на технологии, со Flutter како развоен алат и Firebase како сигурен бекенд за чување на податоци, корисници и слики. Дополнителните third-party сервиси (Cloudinary, Image Picker, URL Launcher, Riverpod) овозможуваат стабилна работа, брз перформанс и одлично корисничко искуство. Овој проект не е само вежба во изработка на мобилна апликација, туку и реално корисна алатка за студентите.

Во целост, „Изгубено & најдено“ покажува како технологијата може да се искористи за олеснување на секојдневните проблеми и како со добра идеја и соодветни алатки може да се создаде корисна апликација која им носи реална корист на студентите.