Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий
институт

Кафедра «Информатика»
кафедра

ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ № 2

Реализация пользовательского интерфейса
Тема

Преподаватель _____ И. В. Ковалев
Подпись, дата — Инициалы, Фамилия

Студент КИ19-17/1Б, №031939174 _____ А. К. Никитин
Номер группы, зачетной книжки — Подпись, дата — Инициалы, Фамилия

Красноярск 2022

1 Задачи

Реализовать пользовательский интерфейс согласно 1 практической работе, заполнив данные статическими объектами.

2 Ход работы

Разработка велась на языке программирования Dart при помощи фреймворка Flutter.

На рисунке 1 представлено главное окно разработанного приложения, заполненное статическими данными.

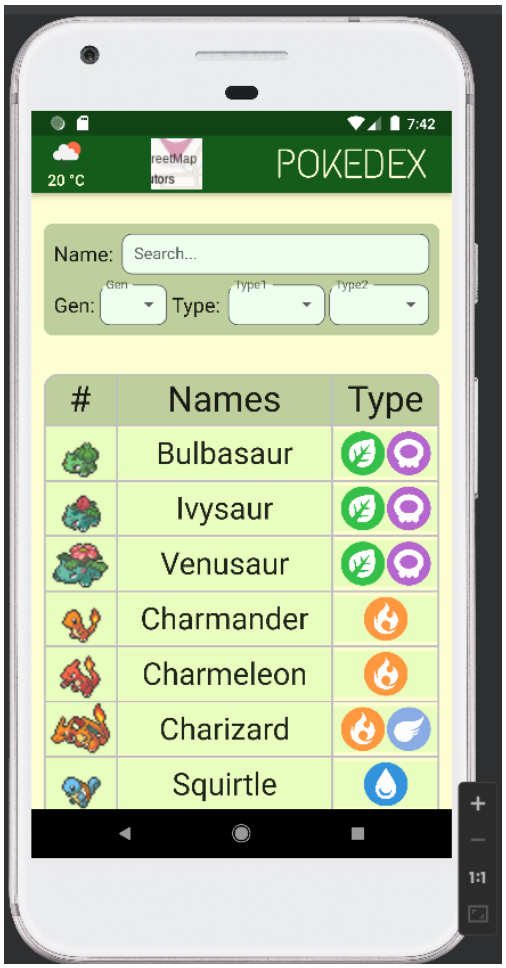

Рисунок 1 – Разработанная верстка приложения

При нажатии на один из найденных элементов таблицы, он подсвечивается и перемещается вверх списка.

Поле Name позволяет фильтровать список покемонов по совпадению в имени.

Поле Gen фильтрует покемонов по принадлежности к поколению.

Поля Type1 и Type2 позволяют явно указать тип покемона.

3 Листинг программы

Ниже представлены листинги со всем элементами системы.

Листинг 1 – Домашний класс

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

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

class HomePageState extends State<HomePage> {
    WeatherContainer weatherContainer = WeatherContainer(10, 800);
    Image mapImage = const Image(
        image: AssetImage('images/weather/sun.png'),
        height: 32,
        width: 32,
    );

    List<Pokemon> pokemonList = List.empty();
    List<Pokemon> mutablePokemonList = List.empty();

    TextEditingController searchController = TextEditingController();
    String? currentGen = "";
    String? currentType1 = "";
    String? currentType2 = "";

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

        getPokemons().then((result) {
            setState(() {
                pokemonList = result;
                mutablePokemonList = result;
            });
        });

        fetchWeather().then((weather) {
            setState(() {
                weatherContainer = weather;
```

```
            });
        });

        fetchMap().then((map) {
            setState(() {
                mapImage = map;
            });
        });

        searchController.addListener(_searchByName);
    }

    List<Pokemon> _searchPokemon() {
        List<Pokemon> newPokemonList = pokemonList
            .where(
                (element) =>

element.name.toLowerCase().startsWith(searchController.text.toLowerCase()))
            .where((element) => currentGen!.isEmpty || genToString(element.gen)
== currentGen!)
            .where((element) =>
        (currentType1!.isEmpty ||
            typeToString(element.type1) == currentType1! ||
            typeToString(element.type2) == currentType1!))
            .where((element) =>
        (currentType2!.isEmpty ||
            typeToString(element.type1) == currentType2! ||
            typeToString(element.type2) == currentType2!))
            .where((element) =>
        currentType1!.isEmpty ||
            currentType2!.isEmpty ||
            currentType1! == currentType2! && element.type1 == element.type2 ||
            currentType1! != currentType2! && element.type1 != element.type2)
            .toList();
        sort(newPokemonList);
        return newPokemonList;
    }

    void _searchByName() {
        setState(() {
            mutablePokemonList = _searchPokemon();
        });
```

```
    }

    void _searchByGen(String? value) {
        setState(() {
            currentGen = value;
            mutablePokemonList = _searchPokemon();
        });
    }

    void _searchByType1(String? value) {
        setState(() {
            currentType1 = value;
            mutablePokemonList = _searchPokemon();
        });
    }

    void _searchByType2(String? value) {
        setState(() {
            currentType2 = value;
            mutablePokemonList = _searchPokemon();
        });
    }

    Function markFavorite(int id) {
        void inner() {
            Pokemon                                pokemon                                =
mutablePokemonList[mutablePokemonList.indexWhere((pokemon) =>
            pokemon.id == id)];
            setState(() {
                pokemon.isFavorite = pokemon.isFavorite ? false : true;
                sort(mutablePokemonList);
            });

            updateFavorite(pokemon);
        }

        return inner;
    }

    void sort(List<Pokemon> pokemons) {
        pokemons.sort((pokemon1, pokemon2) =>
            (pokemon2.isFavorite ? 10000 : 0 - pokemon2.id).compareTo(
```

```
                pokemon1.isFavorite ? 10000 : 0 - pokemon1.id));
    }

    @override
    Widget build(BuildContext context) {
        return Scaffold(
            backgroundColor: const Color(AppColors.mainBackground),
            appBar: CustomAppBar(map: mapImage, weather: weatherContainer),
            body: ListView(
                children: [
                    PokemonSearchContainer(
                        mutablePokemonList: mutablePokemonList,
                        searchController: searchController,
                        onGenChange: _searchByGen,
                        onType1Change: _searchByType1,
                        onType2Change: _searchByType2,
                    ),
                    PokemonTable(
                        mutablePokemonList:  mutablePokemonList,  markFavorite:
markFavorite),
                ],
            ));
    }
}
```

Листинг 2 – Окно навбара

```
class CustomAppBar extends StatefulWidget implements PreferredSizeWidget {
  WeatherContainer weather;
  Image map;
  CustomAppBar({Key? key, required this.weather, required this.map})
      : preferredSize = const Size.fromHeight(kToolbarHeight),
        super(key: key);

  @override
  final Size preferredSize;

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

class CustomAppBarState extends State<CustomAppBar> {
  @override
  Widget build(BuildContext context) {
```

```
    return AppBar(
      backgroundColor: const Color(AppColors.appBarBackground),
      title: Row(
          mainAxisAlignment: MainAxisAlignment.spaceBetween,
          children: <Widget>[
            Column(
              children: <Widget>[
                widget.weather.icon,
                Text(
                    '${widget.weather.temperature} °C',
                    style: const TextStyle(
                        color: Color(AppColors.appBarText), fontSize: 15)),
              ],
            ),
            widget.map,
            TextButton(
              onPressed: () {},
              child: const Text(
                'POKEDEX',
                style: TextStyle(
                    color: Color(0xFFFEFFD0),
                    fontFamily: "PokedexName",
                    fontSize: 40),
              ),
            ),
          ]),
    );
  }
}
```

Листинг 3 – Виджет поиска

```
class PokemonSearchContainer extends StatefulWidget {
    List<Pokemon> mutablePokemonList;
    TextEditingController searchController;
    Function(String?) onGenChange;
    Function(String?) onType1Change;
    Function(String?) onType2Change;

    PokemonSearchContainer({Key? key,
        required this.mutablePokemonList,
        required this.searchController,
        required this.onGenChange,
        required this.onType1Change,
```

```
        required this.onType2Change,
    }) : super(key: key);

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

class PokemonSearchContainerState extends State<PokemonSearchContainer> {

    @override
    Widget build(BuildContext context) {
        return Container(
            height: 110,
            margin: const EdgeInsets.fromLTRB(12.0, 30.0, 12.0, 25.0),
            decoration: const BoxDecoration(
                color: Color(AppColors.tableHeader),
                borderRadius: BorderRadius.all(Radius.circular(10.0))),
            child: Column(children: [
                Container(
                    padding:
                    const EdgeInsets.fromLTRB(10, 10, 10, 10),

                    child: Row(
                        mainAxisAlignment: MainAxisAlignment.spaceBetween,
                        children: <Widget>[
                            const Text(
                                'Name:',
                                style: TextStyle(fontSize: 20),
                            ),
                            Container(
                                padding: const EdgeInsets.only(left: 4.0),
                                width: 305,
                                height: 40,
                                child:                    SearchForm(searchList:
widget.mutablePokemonList, controller: widget.searchController)),
                        ])


                ),

                Row(
                    mainAxisAlignment: MainAxisAlignment.spaceBetween,
```

```
                    children: [
                        Container(
                            padding:
                            const EdgeInsets.fromLTRB(10, 0, 0, 10),
                            child: Row(
                                children: [
                                    const Text(
                                        'Gen:',
                                        style: TextStyle(fontSize: 20),
                                    ),
                                    Container(
                                        padding:   const   EdgeInsets.only(left:
4.0),
                                        width: 70,
                                        height: 40,
                                        child:  SelectForm(selectList:  genList,
onChange: widget.onGenChange, hint: "Gen")),
                                    ],
                            )
                        ),
                        Container(
                            padding:
                            const EdgeInsets.fromLTRB(4, 0, 10, 10),
                            child: Row(
                                children: [
                                    const Text(
                                        'Type:',
                                        style: TextStyle(fontSize: 20),
                                    ),
                                    Container(
                                        padding:   const   EdgeInsets.only(left:
7.0),
                                        width: 102,
                                        height: 40,
                                        child:  SelectForm(selectList:  typeList,
onChange: widget.onType1Change, hint: "Type1")),
                                    Container(
                                        padding:   const   EdgeInsets.only(left:
4.0),
                                        width: 102,
                                        height: 40,
```

```
                                        child: SelectForm(selectList: typeList,
onChange: widget.onType2Change, hint: "Type2")),
                                    ],
                                )
                            ),
                        ],
                    )
                ]),
            );
    }
}
```

Листинг 4 – Виджет таблицы

```
class PokemonTable extends StatefulWidget {
  List<Pokemon> mutablePokemonList;

  Function(int) markFavorite;

  PokemonTable({
    Key? key,
    required this.mutablePokemonList,
    required this.markFavorite,
  }) : super(key: key);

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

class PokemonTableState extends State<PokemonTable> {
  @override
  Widget build(BuildContext context) {
    return Container(
        margin: const EdgeInsets.symmetric(vertical: 12, horizontal: 12),
        child: _createTable(widget.markFavorite, widget.mutablePokemonList));
  }
}

Table _createTable(markFavorite, mutablePokemonList) {
  return Table(
      columnWidths: const {
        0: FixedColumnWidth(72),
        1: FlexColumnWidth(2),
        2: FlexColumnWidth(1),
```

```
      },
      defaultVerticalAlignment: TableCellVerticalAlignment.middle,
      border: TableBorder.all(
          width: 2,
          color: const Color(AppColors.tableBorders),
          borderRadius: const BorderRadius.only(
              topLeft: Radius.circular(15.0), topRight: Radius.circular(15.0))),
      children: [
        _createHeader(),
        ..._createRows(markFavorite, mutablePokemonList),
      ]);
}

TableRow _createHeader() {
  return TableRow(children: [
    Container(
        padding: const EdgeInsets.all(5),
        decoration: const BoxDecoration(
            color: Color(AppColors.tableHeader),
            borderRadius: BorderRadius.only(topLeft: Radius.circular(15.0))),
        child: const Center(
            child: (Text(
          '#',
          style: TextStyle(
            fontSize: 35,
          ),
        )))),
    Container(
        padding: const EdgeInsets.all(5),
        decoration: const BoxDecoration(color: Color(AppColors.tableHeader)),
        child: const Center(
            child: (Text(
          'Names',
          style: TextStyle(
            fontSize: 35,
          ),
        )))),
    Container(
        padding: const EdgeInsets.all(5),
        decoration: const BoxDecoration(
            color: Color(AppColors.tableHeader),
            borderRadius: BorderRadius.only(topRight: Radius.circular(15.0))),
```

```
        child: const Center(
            child: (Text(
          'Type',
          style: TextStyle(
            fontSize: 35,
          ),
        ))))
  ]);
}

List<TableRow>     _createRows(Function(int)     markFavorite,     List<Pokemon>
mutablePokemonList) {
  return mutablePokemonList
      .map((pokemon) {
        return TableRow(children: [
          TableRowInkWell(
              onTap: markFavorite(pokemon.id),
              child: Container(
                color: pokemon.isFavorite
                    ? const Color(AppColors.tableFavoriteColor)
                    : const Color(AppColors.tableBackgroundColor),
                child: Center(
                    child: Image(
                        image: AssetImage("images/pokemon/${pokemon.id}.png"),
                        height: 54,
                        width: 72,
                        fit: BoxFit.fill)),
              )),
          TableRowInkWell(
              onTap: markFavorite(pokemon.id),
              child: Container(
                  padding:  const  EdgeInsets.symmetric(vertical:  0,  horizontal:
10),
                  color: pokemon.isFavorite
                      ? const Color(AppColors.tableFavoriteColor)
                      : const Color(AppColors.tableBackgroundColor),
                  child: SizedBox(
                      height: 50,
                      child: FittedBox(
                        fit: BoxFit.scaleDown,
                        child: Center(
                            child: Text(
```

```
                        pokemon.name,
                        textAlign: TextAlign.center,
                        style: const TextStyle(
                          fontSize: 30,
                        ),
                      )),
                    )))),
          TableRowInkWell(
              onTap: markFavorite(pokemon.id),
              child: Container(
                  color: pokemon.isFavorite
                      ? const Color(AppColors.tableFavoriteColor)
                      : const Color(AppColors.tableBackgroundColor),
                  child:    Row(mainAxisAlignment:    MainAxisAlignment.center,
children: [
                    Image(
                      image: AssetImage(

"images/type/${typeToString(pokemon.type1).toLowerCase()}.png"),
                      height: 45,
                      width: 45,
                    ),
                    if (pokemon.type1 != pokemon.type2) ...[
                      Image(
                        image: AssetImage(

"images/type/${typeToString(pokemon.type2).toLowerCase()}.png"),
                        height: 45,
                        width: 45,
                      ),
                    ],
                  ])))
        ]);
      })
      .toList()
      .cast<TableRow>();
}
```