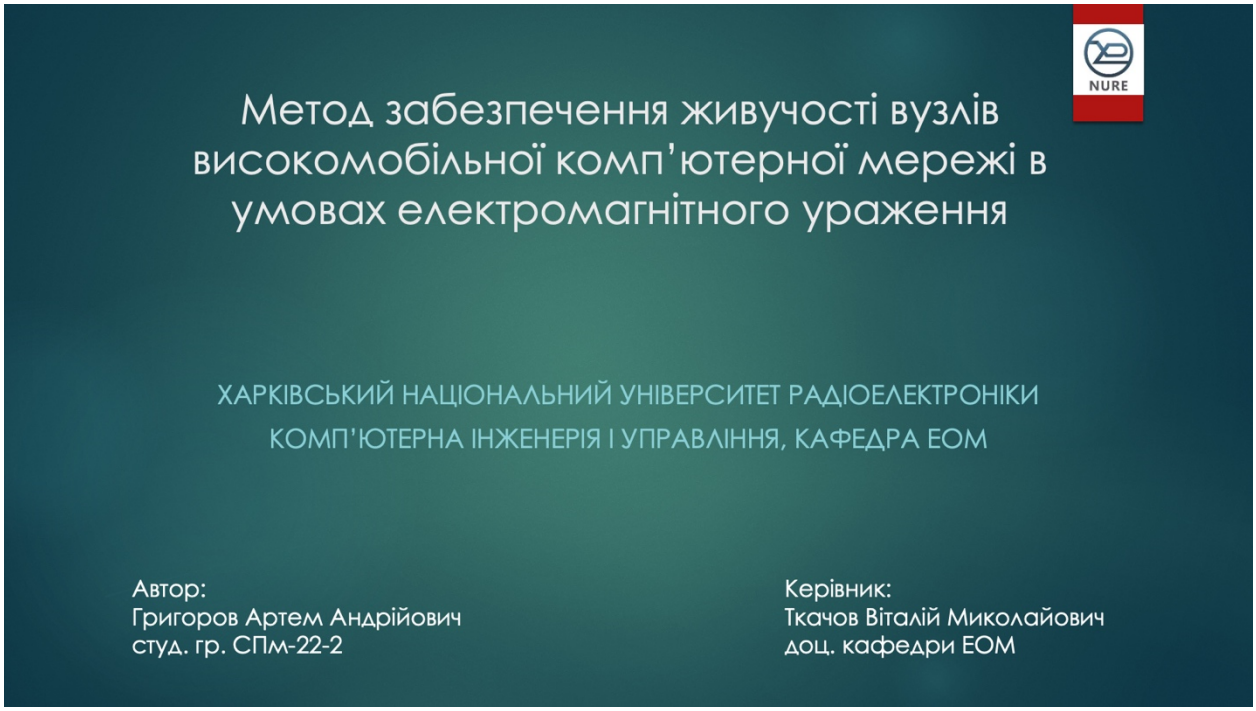
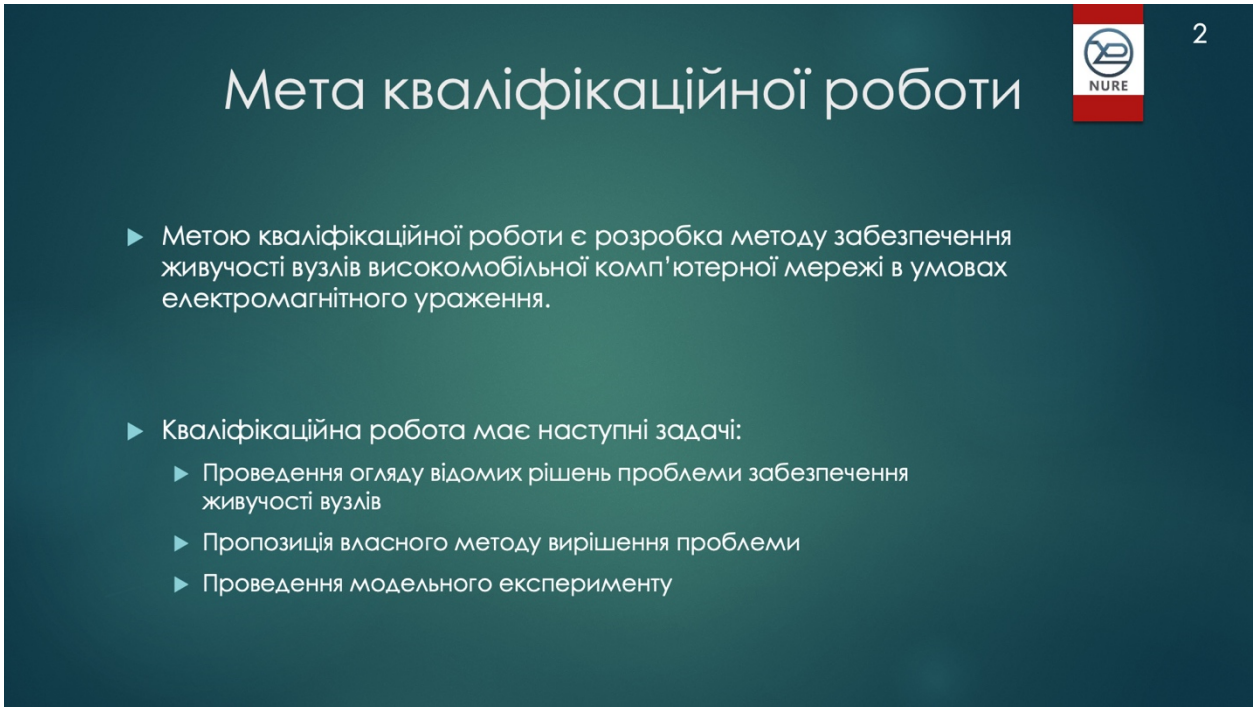
ДОДАТОК А
ГРАФІЧНИЙ МАТЕРІАЛ КВАЛІФІКАЦІЙНОЇ РОБОТИ

1.


Метод забезпечення живучості вузлів високомобільної комп'ютерної мережі в умовах електромагнітного ураження
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ
КОМП'ЮТЕРНА ІНЖЕНЕРІЯ І УПРАВЛІННЯ, КАФЕДРА ЕОМ
Автор:
Григоров Артем Андрійович
студ. гр. СПм-22-2
Керівник:
Ткачов Віталій Миколайович
доц. кафедри ЕОМ

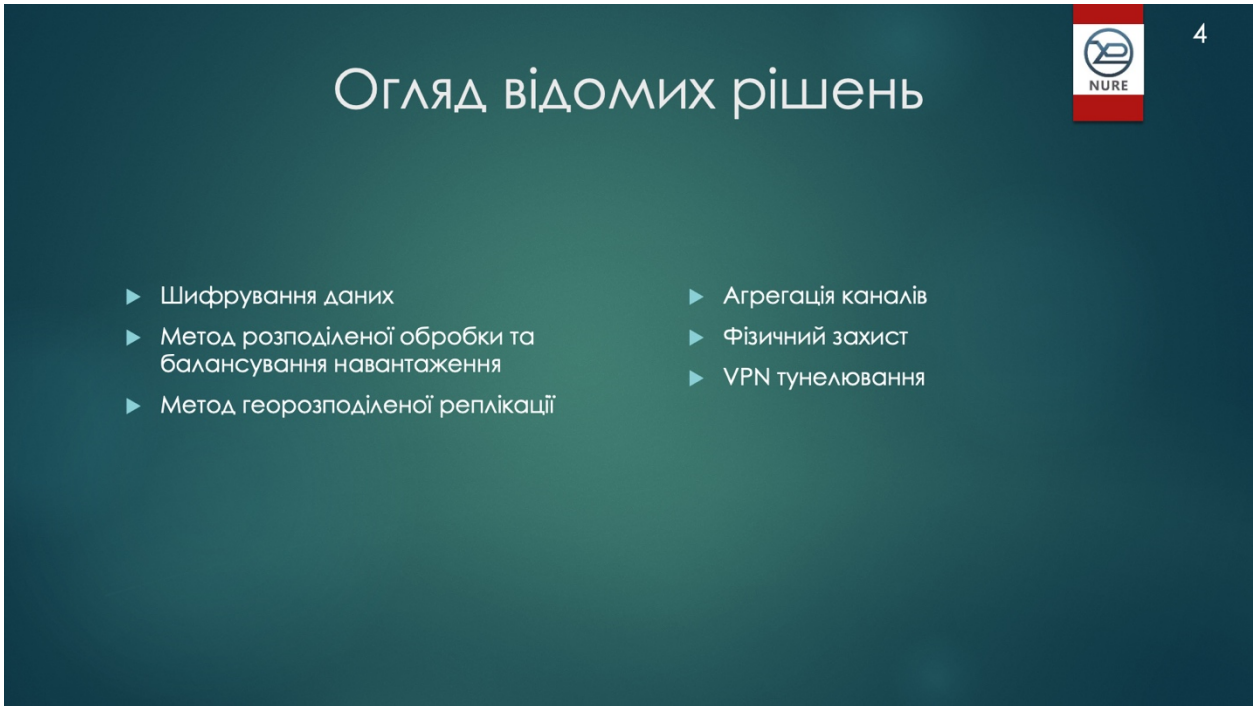
2.


2
Мета кваліфікаційної роботи
Метою кваліфікаційної роботи є розробка методу забезпечення живучості вузлів високомобільної комп'ютерної мережі в умовах електромагнітного ураження.
Кваліфікаційна робота має наступні задачі:
Проведення огляду відомих рішень проблеми забезпечення живучості вузлів
Пропозиція власного методу вирішення проблеми
Проведення модельного експерименту

3.


3
Предметна область
Високомобільна комп'ютерна мережа та її основні характеристики
Вузли мережі
Проблематика електромагнітного ураження мережі
Рисунок 1 - Високомобільна мережа на прикладі рою дронів

4.


4
Огляд відомих рішень
Шифрування даних
Метод розподіленої обробки та балансування навантаження
Метод георозподіленої реплікації
Агрегація каналів
Фізичний захист
VPN тунелювання

5.

5

Основні недоліки існуючих рішень

- Низька ефективність при локальних збоях
- Висока вартість реалізації та підтримки
- Складнощі синхронізації даних
- Підвищена вразливість до кібератак

6.

6

Метод забезпечення живучості. Архітектура

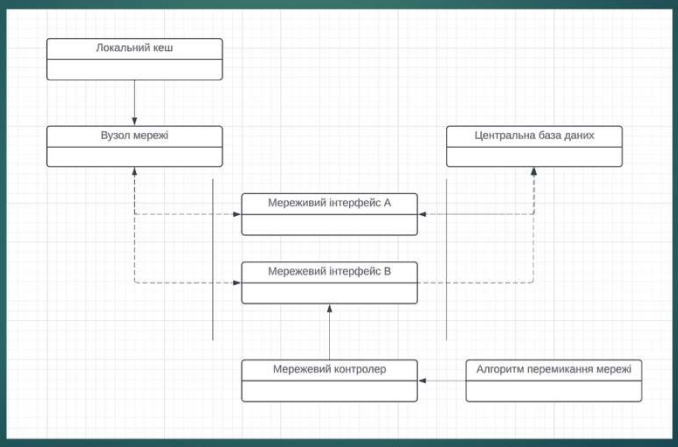


Рисунок 2 - Архітектура взаємодії компонентів методу

7.

7

Метод забезпечення живучості. Алгоритми пріорітезації

$$P_{\text{дрон}} = P_{\text{базовий}} + B_{\text{батарея}} + B_{з'єднання} + \cdots + B_N$$

$P_{\text{дрон}}$ - кінцевий пріоритет дрона;

$P_{\text{базовий}}$ - базовий пріоритет, присвоєний дрону. Це може бути статичне число або залежати від ролі дрона у місії;

$B_{\text{батарея}}$ - бонус, що враховує заряд батареї дрона. Це може бути обчислено як функція відсотка заряду батареї. Наприклад: Вбатарея = $k * V$, де k це коефіцієнт вагомості батареї, а V це відсоток заряду батареї;

$B_{з'єднання}$ - бонус за якість з'єднання дрона. Це може бути визначено, наприклад, через силу сигналу або стабільність з'єднання. Наприклад: $B_{з'єднання} = m * Q$, де m - коефіцієнт вагомості з'єднання, а Q - якість зв'язку;

B_N - інші бонусні коефіцієнти, які мають індивідуально підбиратися під умови використання вузлів і їх особливості.

$$P_{\text{маршрутизатор}} = P_{\text{базовий}} + P_{\text{інтерфейс}}$$

$P_{\text{маршрутизатор}}$- кінцевий пріоритет маршрутизатора;

$P_{\text{базовий}}$ - заданийпріоритет дрону;

$P_{\text{інтерфейс}}$ - пріоритет інтерфейсу, якщо інтерфейс активний.

8.

8

Теоретичне обгрунтування

Рисунок 3 - Приклад дрону, який виконує місію збору інформації, будучи вузлом високомобільної мережі

9.

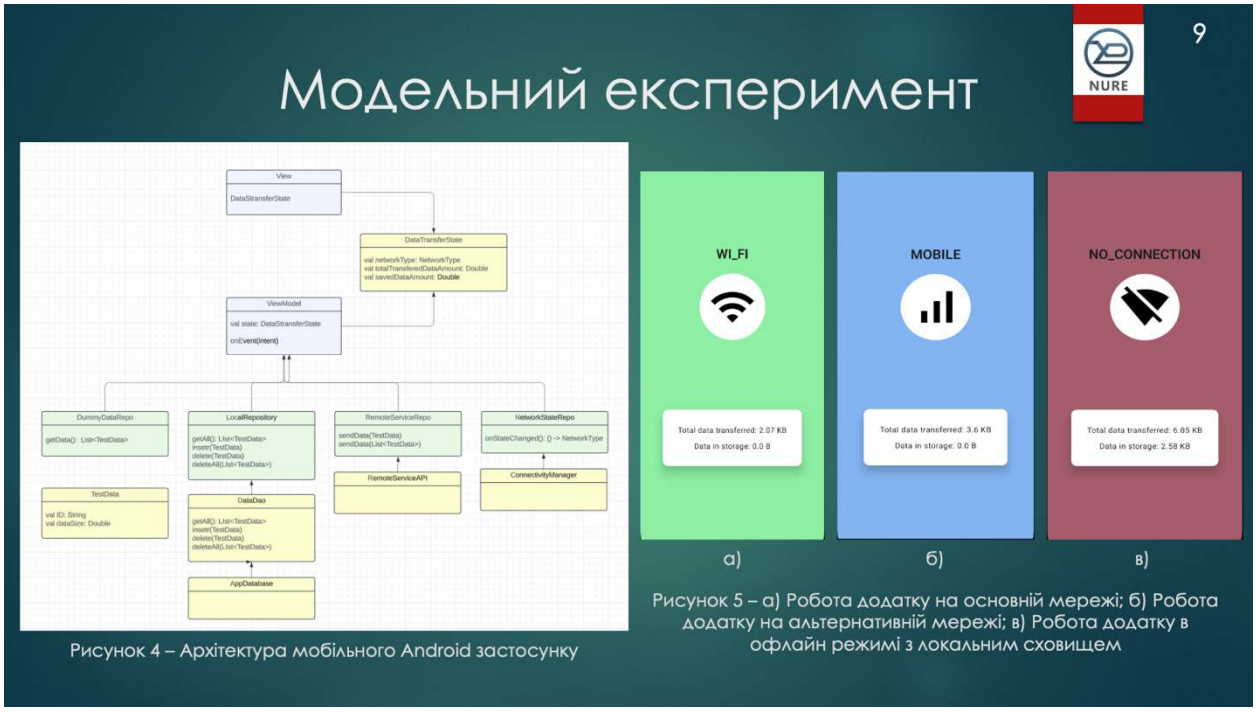


Рисунок 4 – Архітектура мобільного Android застосунку

Рисунок 5 – а) Робота додатку на основній мережі; б) Робота додатку на альтернативній мережі; в) Робота додатку в офлайн режимі з локальним сховищем

10.

№	Критеріальна база	Результат
1	Функціональність основної мережі	✅
2	Переключення на альтернативну мережу	✅
3	Кешування вхідних даних та ефективне використання локальної бази даних	✅
4	Видалення даних після відправки	✅
5	Повернення до основної мережі	✅
6	Відсутність всіх мереж	✅
7	Низький рівень сигналу	⚠️
8	Час перемикання мережі	✅

Таблиця 1 – Критеріальна база оцінки ефективності

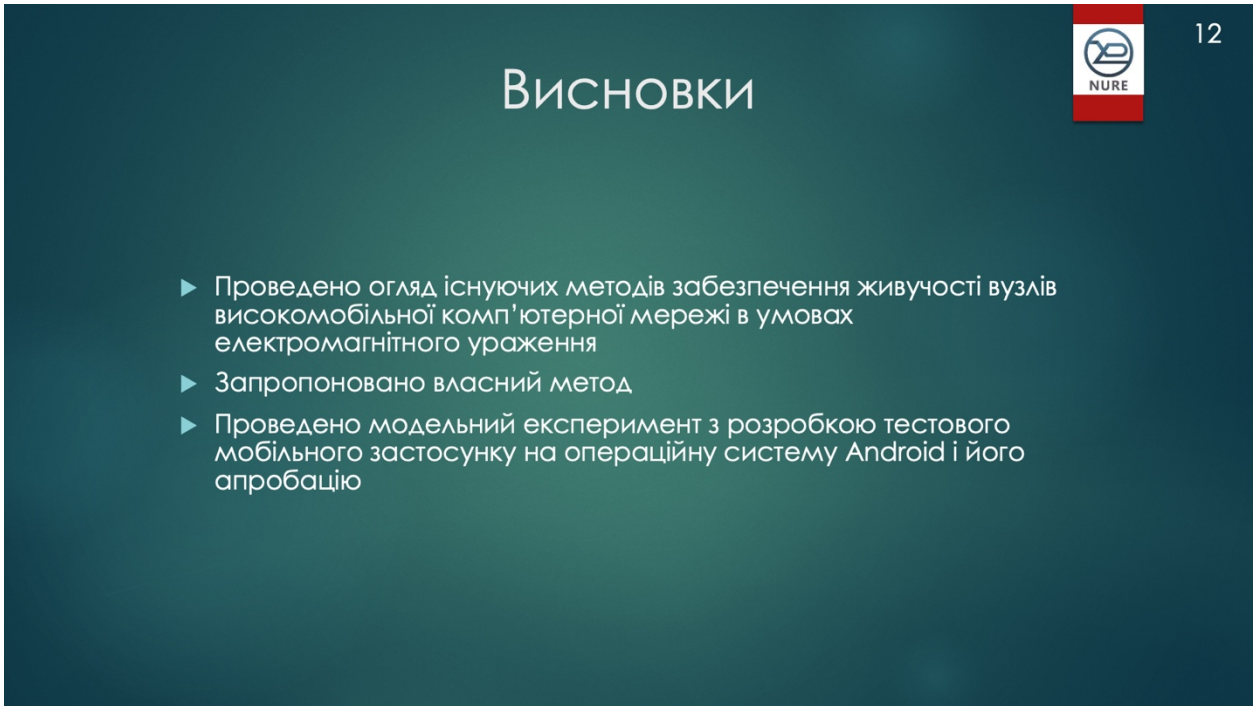
№	1	2	3	4	5	6	7	8	9	10
Час	1.49	3.04	3.53	3.17	2.13	1.94	0.36	2.00	2.05	3.31

Таблиця 2 – Результати швидкості переключення додатку на альтернативну мережу

11.


11
Апробація
NURE
АВТОМАТИЗАЦІЯ ПРОЦЕСУ ВІДНОВЛЕННЯ ВИСОКОМОБІЛЬНОЇ МЕРЕЖІ ПІСЛЯ ЕЛЕКТРОМАГНІТНОГО ВПЛИВУ
09.11.2023 12:05
[1. Інформаційні системи і технології]
Автор: Григоров Артем Андрійович, студент, Харківський національний університет радіоелектроніки, м. Харків
ORCID: 0009-0007-1420-8661
Рисунок 6 – Заголовок опублікованої статті
Посилання на публікацію: http://www.konferenciaonline.org.ua/ua/article/id-1431/

12.


12
Висновки
NURE
Проведено огляд існуючих методів забезпечення живучості вузлів високомобільної комп'ютерної мережі в умовах електромагнітного ураження
Запропоновано власний метод
Проведено модельний експеримент з розробкою тестового мобільного застосунку на операційну систему Android і його апробацію

ДОДАТОК Б
ПРОГРАМИ ДЛЯ РОЗРАХУНКУ

Б.1 Реалізація основних компонентів мобільного Android застосунку

Б.1.1 Репозиторії

Лістинг 1.1.1 – Реалізація моніторингу мережі в NetworkStateRepository

```
class NetworkStateRepository(private val connectivityManager:
ConnectivityManager) {

    fun registerNetworkStateListener(
        onLost: () -> Unit,
        onCapabilitiesChanged: (NetworkCapabilities) -> Unit,
    ) {
        connectivityManager.registerDefaultNetworkCallback(object
:
            ConnectivityManager.NetworkCallback() {

            override fun onAvailable(network: Network) {
                // Empty
            }

            override fun onLost(network: Network) {
                onLost()
            }

            override fun onCapabilitiesChanged(
                network: Network,
                networkCapabilities: NetworkCapabilities
            ) {
                onCapabilitiesChanged(networkCapabilities)
            }

            override fun onLinkPropertiesChanged(network:
Network, linkProperties: LinkProperties) {
                // Empty
            }
        })
    }
}
```

Лістинг 1.1.2 – Реалізація репозиторію управління локальними даними LocalStorageRepository

```
class LocalStorageRepository(context: Context) {

   private val dao = AppDatabase.getDatabase(context).dao()

   fun getAll(): List<TestData> = dao.getAll()

   fun insert(data: TestData) = dao.insert(data)

   fun delete(data: TestData) = dao.delete(data)

   fun deleteAll() = dao.deleteAll()
}
```

Б.1.2 База даних

Лістинг 1.2.1 – Реалізація бази даних AppDatabase

```
@Database(entities = [TestData::class], version = 1)
abstract class AppDatabase : RoomDatabase() {

   abstract fun dao(): TestDataDao

   companion object {

       @Volatile
       private var INSTANCE: AppDatabase? = null

       fun getDatabase(context: Context): AppDatabase {
           return INSTANCE ?: synchronized(this) {
               val instance = Room.databaseBuilder(
                   context.applicationContext,
                   AppDatabase::class.java,
                   "test_database"
               ).build()
               INSTANCE = instance
               instance
           }
       }
   }

}
```

Б.1.3 Інші компоненти

Лістинг 1.3.1 – Стан додатку DataTransferedState

```
data class DataTransferState(
   val networkType: NetworkType = NetworkType.NO_CONNECTION,
   val totalTransferDataAmount: Double = 0.0,
   val localSavedDataAmount: Double = 0.0,
)
```

Лістинг 1.3.2 – Модель тестових даних реалізована як Kotlin Data Class

```
@Entity
data class TestData(
   @PrimaryKey val id: String,
   @ColumnInfo(name = "data_amount") val dataAmount: Double,
)
```

Б.1.4 Інтерфейс користувача (UI)

Лістинг 1.4.1 – Головний екран додатку

```
@Composable
fun MainScreen(state: DataTransferState) {
    Column(
        modifier = Modifier
            .fillMaxSize()
            .background(
                color = when (state.networkType) {
                    NetworkType.WI_FI -> Color(0xFF66f29c)
                    NetworkType.MOBILE -> Color(0xFF74b6f7)
                    NetworkType.NO_CONNECTION ->
Color(0xFFb0586b)
                }
            ),
        verticalArrangement = Arrangement.Center,
        horizontalAlignment = Alignment.CenterHorizontally,
    ) {
        Text(
            text = state.networkType.name,
            style = TextStyle(
                fontSize = 28.sp,
                fontWeight = FontWeight.SemiBold,
            )
        )
        Box(
            modifier = Modifier
                .padding(top = 24.dp)
                .size(140.dp)
                .background(
                    color = Color.White,
                    shape = CircleShape,
```

Продовження лістингу 1.4.1

```
                )
        ) {
            Image(
                modifier = Modifier
                    .align(Alignment.Center)
                    .size(100.dp),
                painter = when (state.networkType) {
                    NetworkType.WI_FI -> painterResource(id =
R.drawable.baseline_wifi_24)
                    NetworkType.MOBILE -> painterResource(id =
R.drawable.baseline_signal_cellular_alt_24)
                    NetworkType.NO_CONNECTION ->
painterResource(id = R.drawable.baseline_signal_wifi_off_24)
                },
                contentDescription = null,
            )
        }
        Card(
            modifier = Modifier.padding(top = 148.dp),
            colors = CardDefaults.cardColors(containerColor =
Color.White),
            elevation =
CardDefaults.elevatedCardElevation(defaultElevation = 12.dp),
        ) {
            Text(
                modifier = Modifier
                    .padding(horizontal = 32.dp)
                    .padding(top = 32.dp),
                text = "Total data transferred:
${DataUtil.convertBytes(state.totalTransferDataAmount)}",
            )
            Text(
                modifier = Modifier
                    .padding(top = 12.dp)
                    .padding(bottom = 32.dp)
                    .align(Alignment.CenterHorizontally),
                text = "Data in storage:
${DataUtil.convertBytes(state.localSavedDataAmount)}",
            )
        }
    }
}
```

Лістинг 1.4.2 – Превью головного екрану

```
@Preview(showBackground = true)
@Composable
fun MainScreenPreview() {
    NetworkSwitchTheme {
        MainScreen(state = DataTransferState())
    }
}
```