**Технічний звіт до завдання 3**

**Розробка архітектури проекту та бази даних**

**Програмування для мобільних платформ**

**Командний проект BePresent**

**Виконали:**

* Бурдяк Олена
* Середня Ірина
* Чушак Христина
* Долуда Андрій
* Зозуля Олег

**Дата:** Травень 2025

**Зміст**

1. Вступ
2. Архітектура проекту
3. Схема бази даних
4. Реалізація бази даних
5. Тестові дані
6. Висновки

**Вступ**

BePresent - це Android-додаток для створення та управління списками подарунків (wishlists). Додаток дозволяє користувачам створювати дошки подарунків, додавати ідеї подарунків, резервувати подарунки для інших користувачів та керувати доступом до своїх списків.

Мета третього завдання - розробити архітектуру проекту та базу даних для забезпечення надійного зберігання та управління даними додатка.

**Архітектура проекту**

**Трирівнева архітектура**

Проект реалізований за принципом трирівневої архітектури:

**1. Data Access Layer (Рівень доступу до даних)**

* **Пакет:** com.example.bepresent.database
* **Компоненти:**
  + Room entities (room/)
  + DAO інтерфейси (dao/)
  + Type converters (typeConverters/)
  + Database manager (dbManager/)

**2. Business Logic Layer (Рівень бізнес-логіки)**

* **Пакет:** com.example.bepresent.viewmodel
* **Компоненти:**
  + ViewModels для кожного екрану
  + Управління станом UI
  + Обробка бізнес-логіки

**3. Presentation Layer (Рівень представлення)**

* **Пакет:** com.example.bepresent.ui
* **Компоненти:**
  + Compose UI екрани (screens/)
  + Переиспользуемые компоненти (components/)
  + Навігація (navigation/)
  + Теми та стилі (theme/)

**Використані технології**

* **Kotlin** - основна мова програмування
* **Jetpack Compose** - для побудови UI
* **Room Database** - для локального зберігання даних
* **Coroutines & Flow** - для асинхронного програмування
* **Navigation Compose** - для навігації між екранами
* **Material 3** - для дизайн-системи

**Схема бази даних**

**UML діаграма бази даних**

┌─────────────────────────┐ ┌─────────────────────────┐

│ GiftBoardRoom │────▷│ GiftRoom │

├─────────────────────────┤ ├─────────────────────────┤

│ + boardId: Int (PK) │ │ + giftId: Int (PK) │

│ + boardName: String │ │ + giftName: String │

│ + celebrationDate: Date │ │ + giftDescription: String│

│ + accessType: String │ │ + link: String │

│ + description: String │ │ + image: Bitmap? │

│ + creationDate: Date │ │ + reserved: Boolean │

└─────────────────────────┘ │ + reservoirUserId: Int? │

└─────────────────────────┘

│

▼

┌─────────────────────────┐

│ ReservedGiftsRoom │

├─────────────────────────┤

│ + reservedGiftId: Int(PK)│

│ + reservationDate: Date │

│ + receiverId: Int │

│ + receiverName: String │

│ + reservedGiftName: String│

│ + reservedGiftDescription│

│ + link: String │

│ + image: Bitmap? │

└─────────────────────────┘

**Опис таблиць**

**1. GiftBoardRoom (giftBoards)**

**Призначення:** Зберігання інформації про дошки подарунків

| **Поле** | **Тип** | **Опис** |
| --- | --- | --- |
| boardId | Int (PK) | Унікальний ідентифікатор дошки |
| boardName | String | Назва дошки |
| celebrationDate | Date? | Дата святкування |
| accessType | String | Тип доступу (public/friends/private) |
| description | String | Опис дошки |
| creationDate | Date? | Дата створення |

**2. GiftRoom (gifts)**

**Призначення:** Зберігання інформації про подарунки

| **Поле** | **Тип** | **Опис** |
| --- | --- | --- |
| giftId | Int (PK) | Унікальний ідентифікатор подарунка |
| giftName | String | Назва подарунка |
| giftDescription | String | Опис подарунка |
| link | String | Посилання на подарунок |
| image | Bitmap? | Зображення подарунка |
| reserved | Boolean | Чи зарезервований подарунок |
| reservoirUserId | Int? | ID користувача, який зарезервував |

**3. ReservedGiftsRoom (reservedGifts)**

**Призначення:** Зберігання інформації про зарезервовані подарунки

| **Поле** | **Тип** | **Опис** |
| --- | --- | --- |
| reservedGiftId | Int (PK) | Унікальний ідентифікатор резервації |
| reservationDate | Date? | Дата резервації |
| receiverId | Int | ID отримувача подарунка |
| receiverName | String | Ім'я отримувача |
| reservedGiftName | String | Назва зарезервованого подарунка |
| reservedGiftDescription | String | Опис подарунка |
| link | String | Посилання на подарунок |
| image | Bitmap? | Зображення подарунка |

**Реалізація бази даних**

**Database Manager**

@Database(

entities = [GiftRoom::class, GiftBoardRoom::class, ReservedGiftsRoom::class],

version = 3,

exportSchema = false

)

@TypeConverters(TypeConverter::class)

abstract class DatabaseManager : RoomDatabase() {

abstract fun giftDao(): GiftDao

abstract fun giftBoardDao(): GiftBoardDao

abstract fun reservedGiftsDao(): ReservedGiftDao

companion object {

@Volatile

private var instance: DatabaseManager? = null

fun getDatabase(context: Context): DatabaseManager =

instance ?: synchronized(this) {

instance ?: buildDatabase(context).also { instance = it }

}

}

}

**DAO інтерфейси**

**GiftDao**

* insertGift() - додавання нового подарунка
* getAllGifts() - отримання всіх подарунків
* deleteAllGifts() - видалення всіх подарунків

**GiftBoardDao**

* insertGiftBoard() - створення нової дошки
* getAllGiftBoard() - отримання всіх дошок
* deleteAllGiftBoard() - видалення всіх дошок

**ReservedGiftDao**

* insertReservedGift() - додавання резервації
* getAllReservedGifts() - отримання всіх резервацій
* deleteAllReservedGifts() - видалення всіх резервацій

**Type Converters**

Для конвертації складних типів даних:

class TypeConverter {

// Date <-> Long

@TypeConverter

fun fromTimestamp(value: Long?): Date?

@TypeConverter

fun dateToTimestamp(date: Date?): Long?

// Bitmap <-> ByteArray

@TypeConverter

fun fromBitmap(bitmap: Bitmap?): ByteArray?

@TypeConverter

fun toBitmap(bytes: ByteArray?): Bitmap?

// List<String> <-> JSON String

@TypeConverter

fun fromStringList(value: List<String>?): String?

@TypeConverter

fun toStringList(value: String?): List<String>?

}

**Тестові дані**

**TestDataPopulator**

Створено клас TestDataPopulator для автоматичного заповнення бази даних тестовими даними:

**Тестові дошки (5 записів)**

1. **"Мій 25-й день народження"** - публічна дошка
2. **"Новий рік 2025"** - для друзів
3. **"Весілля Ані та Максима"** - приватна дошка
4. **"8 березня для мами"** - для друзів
5. **"Випускний 2025"** - публічна дошка

**Тестові подарунки (20 записів)**

Різноманітні подарунки від технічних гаджетів до книг та хобі:

* MacBook Air M2
* AirPods Pro (зарезервовано)
* Книги та курси
* Спортивне обладнання
* Кімнатні рослини
* Настільні ігри

**Тестові резервації (5 записів)**

Приклади зарезервованих подарунків з різними датами резервації.

**Автоматичне заповнення**

class MainActivity : ComponentActivity() {

override fun onCreate(savedInstanceState: Bundle?) {

// ...

lifecycleScope.launch {

val testDataPopulator = TestDataPopulator(database)

testDataPopulator.populateWithTestData()

}

// ...

}

}

**Висновки**

**Досягнуті результати**

1. **Розроблено повноцінну схему бази даних** з трьома основними таблицями та їх зв'язками
2. **Реалізовано трирівневу архітектуру** проекту з чітким розділенням відповідальності
3. **Створено базу даних Room** з підтримкою міграцій та type converters
4. **Заповнено базу даних тестовими даними** для демонстрації функціональності
5. **Налаштовано автоматичне управління базою даних** через DatabaseManager

**Переваги реалізації**

* **Масштабованість:** Архітектура дозволяє легко додавати нові функції
* **Надійність:** Використання Room забезпечує стабільну роботу з даними
* **Тестованість:** Чітке розділення шарів спрощує тестування
* **Продуктивність:** Оптимізовані запити та кешування даних

**Подальший розвиток**

1. Додавання зв'язків між таблицями через Foreign Keys
2. Реалізація повнотекстового пошуку
3. Додавання індексів для оптимізації запитів
4. Реалізація синхронізації з хмарною базою даних

Архітектура та база даних повністю готові для подальшої розробки функціональності додатка BePresent.