## Архітектура додатку

Архітектура будь якого програмного забезпечення – це один з найважливіших модулів в розробці, так званий «наріжний камінь» додатку. Якісна архітектура забезпечить хорошу масштабуємість та гнучкість усього проекту, полегшить «читання коду» та розуміння логіки внутрішньої роботи. Погано ж продумана архітектура навпаки ускладнить майбутню розробку додатку, і чим більше проект буде удосконалюватись, тим складнішими і незграбними будуть нові доповнення, функціонал, тощо. В підсумку, в якийсь період розробки ми отримаємо програмний продукт з атрофованою гнучкістю та масштабуємістю, і тоді вигідніше буде розпочати розробку нового продукту, ніж підтримувати розробку старого. Саме тому архітектура найважливіший з модулів розробки, важливіше навіть ніж функціонал, так як якісна архітектура підтримує впровадження нових можливостей в роботу додатку.

Розробка мобільного додатку на платформи AndroidOS зумовлює використання та реалізацію патерна проектування MVC, а саме його нащадка MVP.

MVP(Model-View-Presenter) - один з найпоширеніших архітектурних шаблонів проектування, похідний від [MVC](https://uk.wikipedia.org/wiki/MVC), принцип полягає у розділенні даних, візуального відображення та поведінки обробки подій у різні класи, а саме: Модель даних(Model), Представлення(View) та Пред'явник(Presenter). Подібна структура дозволяє змінювати інтерфейс користувача з мінімальними впливами на роботу з внутрішніми даними, та навпаки, вносити зміни в модель даних без модифікування інтерфейсу користувача.

Дана архітектура наведена на рис. 3.1.



Рисунок 3.1. Архітектура додатку.

Мета шаблону – це забезпечення гнучкої архітектури програмного забезпечення, що в свою чергу полегшить майбутні зміни чи розширення додатку. Також надання можливості повторно використовувати окремі модулі та компоненти програми. Використовування цього патерну у великих системах забезпечує певну впорядкованість їх структури і підвищує зрозумілість завдяки зменшенню ступення складності.

Також, зважаючи на те, що основною вимогою до підсистеми «Розклад» була саме можливість працювати в режимі без доступу до мережі інтернет, то потрібно було побудувати таку архітектуру додатку, яка б могла виконувати таку вимогу. Проаналізувавши поставлене завдання, ми прийшли до висновку, що потрібно застосовувати локальну базу даних. Використовувати, СУБД для такого завдання немає потреби, адже можна обійтись використанням вже наданих платформою AndroidOS інструментів. Таким чином локальная база даних буде створена за допомоги SQLite, що дозволить нам реалізувати наш задум просто та швидко, без підключення сторонніх технологій.

Для нормального функціонування підсистеми в цілому потрібна серверна СУБД, яка, в свою чергу, буде забезпечувати збереження всіх даних, які необхідні для роботи додатку.

Дана архітектура наведена на рис. 3.2.



Рисунок 3.2. Архітектура підсистеми.