1. Директории, поддерживаемые всеми платформами

- Application Documents Directory: используется для хранения файлов, создаваемых пользователем или приложением и предназначенных для долговременного хранения.

- Application Support Directory: директория для файлов, которые поддерживают работу приложения, например, настройки или данные пользователя.

- Temporary Directory: используется для хранения временных файлов, которые могут быть удалены системой при необходимости.

2. Директории, поддерживаемые только на Android

- External Storage Directory: используется для хранения данных вне хранилища приложения, доступных другим приложениям.

- External Cache Directories: для хранения кэшированных данных на внешнем накопителе.

- Downloads Directory: папка загрузок, в которую сохраняются загруженные файлы.

3. Используемые библиотеки в рамках лабораторной работы

- `path\_provider`: используется для получения путей к различным директориям (например, документы, временные файлы).

- `dart:io`: для работы с файловой системой, чтения и записи файлов.

- `dart:convert`: для преобразования данных между строками и форматами JSON.

- `uuid`: для генерации уникальных идентификаторов (например, для ID объектов).

4. SQLite: Преимущества и недостатки

Преимущества:

- Легковесность: небольшая и встроенная база данных без необходимости настройки серверов.

- Кросс-платформенность: поддерживается практически на всех платформах.

- Простота использования: достаточно прост в интеграции и работе с данными.

Недостатки:

- Ограничения по масштабируемости: не подходит для приложений с большими объемами данных и высокой конкурентной нагрузкой.

- Отсутствие некоторых функций: не поддерживает некоторые возможности, такие как хранимые процедуры.

- Ограниченные средства администрирования: не обладает мощными инструментами управления, как другие базы данных (например, PostgreSQL или MySQL).

5. Обработка ошибок в зависимости от платформы

В рамках лабораторной работы обработка ошибок проводилась с учетом специфики платформы:

- Использование проверок платформы с помощью `Platform.isAndroid` и `Platform.isIOS`, чтобы убедиться, что определенные функции или доступ к директориям выполняются только на поддерживаемых платформах.

- Обработка ошибок чтения и записи файлов с использованием конструкции `try-catch`, чтобы поймать и обработать возможные исключения, такие как отсутствие доступа или ошибок файловой системы.

- Логирование ошибок с помощью `print()` или `ScaffoldMessenger` для информирования пользователя об ошибках и упрощения отладки.