Зміст

[Структура програмного забезпечення 2](#_Toc166148436)

[Структурна схема 2](#_Toc166148437)

[Опис частин структурної схеми 4](#_Toc166148438)

[Програмне забезпечення 4](#_Toc166148439)

[Фоновий сервіс резервування 4](#_Toc166148440)

[Графічний інтерфейс 5](#_Toc166148441)

[Доменний рівень 6](#_Toc166148442)

[Програмний рівень 7](#_Toc166148443)

[Інфраструктурний рівень 7](#_Toc166148444)

[Реалізація структури програмного забезпечення 8](#_Toc166148445)

[Опис складу програми та репозиторію 8](#_Toc166148446)

[Графічний інтерфейс 9](#_Toc166148447)

[Діаграма класів 9](#_Toc166148448)

[Опис 9](#_Toc166148449)

[Фоновий сервіс резервування 10](#_Toc166148450)

[Діаграма класів 10](#_Toc166148451)

[Опис 10](#_Toc166148452)

[Доменний рівень 11](#_Toc166148453)

[Діаграма класів 11](#_Toc166148454)

[Опис 12](#_Toc166148455)

[Програмний рівень 13](#_Toc166148456)

[Діаграма класів 13](#_Toc166148457)

[Опис 14](#_Toc166148458)

[Інфраструктурний рівень 15](#_Toc166148459)

[Діаграма класів 15](#_Toc166148460)

[Опис 16](#_Toc166148461)

# Структура програмного забезпечення

## Структурна схема

На рисунку 1 зображена структурна схема проекту “Files Safe Reserve”.



Рисунок 1. Cтруктурна схема проекту “Files Safe Reserve”.

Як бачимо, програмне забезпечення “Files Safe Reserve” складається з двох частин:

* **Графічного інтерфейсу**, який буде реалізований у вигляді програми для персональних комп’ютерів.
* **Фонового сервісу** для резервування віртуальних сейфів після часу, який вказав користувач.

Вище вказані графічний інтерфейс і фоновий сервіс резервування будуть використовувати функціонал, який буде описаний у таких рівнях програмного забезпечення

* **Доменний рівень** – описує і втілює найважливіший функціонал програмного забезпечення і не використовує жодних сторонніх бібліотек.
* **Програмний рівень** – описує функціонал бізнес логіки проекту, але не реалізовує жодного з них, тобто налічує інтерфейси, моделі та ін. Таким чином, не залежить від конкретної реалізації архітектури бази даних, платформи операційної системи, тощо.
* **Інфраструктурний рівень** – імплементує функціонал програмного рівня і може прив’язуватися до сторонніх бібліотек і реалізацій спираючись на свої власні.

Відповідно, проекти графічного інтерфейсу і фонового сервісу резервування вміщують лише функціонал характерний їм, а функціонал програмного забезпечення “Files Safe Reserve” втілений окремо від них у доменному, програмному й інфраструктурному рівні.

## Опис частин структурної схеми

### Програмне забезпечення

Цілий проект складається з графічного інтерфейсу і фоновим сервісом резервування, як і зображено на рисунку 1. Описувати вигляд бази даних цього програмного забезпечення буде програмний рівень, а втілювати базу даних за конкретною архітектурою інфраструктурний рівень. Функціонал, що реалізує роботу з файлами та їхнє представлення, буде втілювати доменний рівень.

### Фоновий сервіс резервування

Він буде запускатися автоматично при запуску операційної системи та перевіряти чи необхідно зарезервувати віртуальний сейф відповідно з вказівок користувача. Якщо так, то відбувається резервування всього вмісту, який має бути зарезервований у віртуальний сейф і у логи записується дія резервування. Якщо ні, то резервування не відбувається і у логи нічого не записується. Цей рівень може використовувати доменний, програмний і інфраструктурний рівні, щоб реалізувати свою роботу. Схема роботи фонового сервісу резервування наведена на рисунку 2.



Рисунок 2. Схема роботи фонового сервісу резервування.

### Графічний інтерфейс

Через графічний інтерфейс користувач зможе керувати своїми віртуальними сейфами, а саме створити, змінити та видалити віртуальний сейф. Має бути можливість добавляти файли і каталоги у резервації і їх забирати з резервації, визначати час автоматичної резервації, визначити віртуальний сейф як сейф на знімному накопичувачі. Також, користувач має мати змогу резервувати файли з смартфорнів. На даний момент передбачаються такі сторінки, що мають бути присутні:

* **Домашня сторінка** – містить інформацію про програму та її оновлень релізних версій програми.
* **Сторінка списку віртуальних сейфів** – містить список всіх віртуальних сейфів з кнопками для переходу на їх сторінки і кнопку для створення віртуального сейфу.
* **Сторінка створення віртуального сейфу** – призначена для створення віртуального сейфу
* **Сторінка віртуального сейфу** – містить кнопки: для переходу у каталог віртуального сейфу, для зміни віртуального сейфу, для видалення віртуального сейфу, для резервування віртуального сейфу, для переходу до списку файлів і папок резервування, а також інформацію про віртуальний сейф.
* **Сторінка списку файлів і каталогів для резервування** – містить список файлів та каталогів, кожен з яких має кнопку для його відкриття, відкриття кореневої папки та видалення зі списку.
* **Сторінка для резервування смартфонів** – містить список віртуальних сейфів і підключених через провід смартфонів. Обравши смартфон, з якого буде відбуватися резервування, і віртуальний сейф, у який буде відбуватися резервування, стає доступна кнопка для резервування.
* **Сторінка налаштувань** – містить секцію для встановлення спільного встановлення автоматичного резервування віртуальних сейфів, але має бути можливість обрати, чи будуть перезаписуватися автоматичні резервування, що вже були налаштовані. Також, містить секцію зі списком комбінацій клавіш, кожну з яких можна змінити. Має бути можливість змінити скинути комбінації клавіш до заводських налаштувань.

На рисунку 3 наведено переходи між сторінками, але варто зауважити, що є сторінки, переходи на які відбуваються через бічну панель, тому перехід на них може відбуватися будь-звідки. Вони наведені нижче:

* **Домашня сторінка**
* **Сторінка списку віртуальних сейфів**
* **Сторінка для резервування смартфонів**
* **Сторінка налаштувань**



Рисунок 3. Переходи між сторінками.

### Доменний рівень

На цьому рівні описуються найосновніші сутності та реалізовується найосновніший функціонал програмного забезпечення і не використовується жодна бібліотека. Цей рівень має містити такі сутності:

* **Файл** – представляє файл операційної системи.
* **Каталог** – представляє каталог операційної системи.
* **Файлова система** – представляє файлову систему.
* **Сутність зі шляхом** (далі ipathed) – абстракція над файлом і каталогом (інтерфейс).

Ці сутності має надавати не лише шлях, а й назву файлу з чи без розширення або назву папки. Над файлом, каталогом чи ipathed має бути можливо здійснювати такі операції:

* **Створення**
* **Копіювання**
* **Переміщення**
* **Видалення**

### Програмний рівень

На цьому рівні є функціонал бізнес логіки проекту, але не реалізований, тобто на цьому рівні є лише інтерфейси, моделі та ін. Таким чином, не залежить від конкретної реалізації архітектури бази даних, платформи операційної системи, тощо. База даних має містити такі моделі:

* **Віртуальний сейф** – вміщає в собі деталі віртуального сейфу і модель резервації.
* **Деталі віртуального сейфу** – вміщають дані взаємодії з віртуальним сейфом і логи, що належать віртуальному сейфу.
* **Лог** – містить операції логу і інформацію, що описує стан логу загалом.
* **Операція логу** – певний тип операції у програмі. Наприклад, створення віртуального сейфу, добавлення файлу на резервацію та ін.
* **Резервація** – зберігає файли та каталоги, що мають бути резервовані, а також час останньої резервації.
* **Файл резервації** – файл, що мають резервуватися у віртуальному сейфі.
* **Каталог резервації** – каталог, що мають резервуватися у віртуальному сейфі.
* **Комбінація клавіш** – представлення комбінацій клавіш.

Також, там має зберігатися сутність комбінацій клавіш, функціонал будівника логів і взаємодії з телефоном.

### Інфраструктурний рівень

Реалізує функціонал програмного рівня, а саме створює контекст бази даних із заданих моделей. Обраною технологією представлення взаємодією з базою даних є ***Entity Framework Core***, а обраною архітектурою бази даних є ***SQLite***. Створює CRUD репозиторії для моделей, але для репозиторію комбінацій клавіш не має містити методи створення і видалення, адже комбінації клавіш мають створюватися разом з базою даних. Також, реалізує сервіси та будівника логів у програмному рівні, створюються конфігурації програми.

# Реалізація структури програмного забезпечення

## Опис складу програми та репозиторію

Програма складається з проектів у каталозі “src” і у каталозі “tests”, які вміщують відповідно проекти, які представляють частину з структурної схема на рисунку 1, та, тести для перевірки коду на справність. Нижче наведено назву частини структурної схеми і відповідний проект, що за цю частину відповідає.

* **Графічний інтерфейс** – FilesSafeReserve.UI,
* **Фоновий сервіс резервування** – FilesSafeReserve.Services,
* **Доменний рівень** – FilesSafeReserve.Domain,
* **Програмний рівень** – FilesSafeReserve.App,
* **Інфраструктурний рівень** – FilesSafeReserve.Infra.

Вище було описано роль кожної частини структурної схеми. Також, нижче наведено проекти тестів.

* **FilesSafeReserve.Domain.Tests** – тести для FilesSafeReserve.Domain.
* **FilesSafeReserve.Infra.Tests** – тести для FilesSafeReserve.Infra.

## Графічний інтерфейс

### Діаграма класів

Діаграма класів графічного інтерфейсу наведено на рисунку 4.

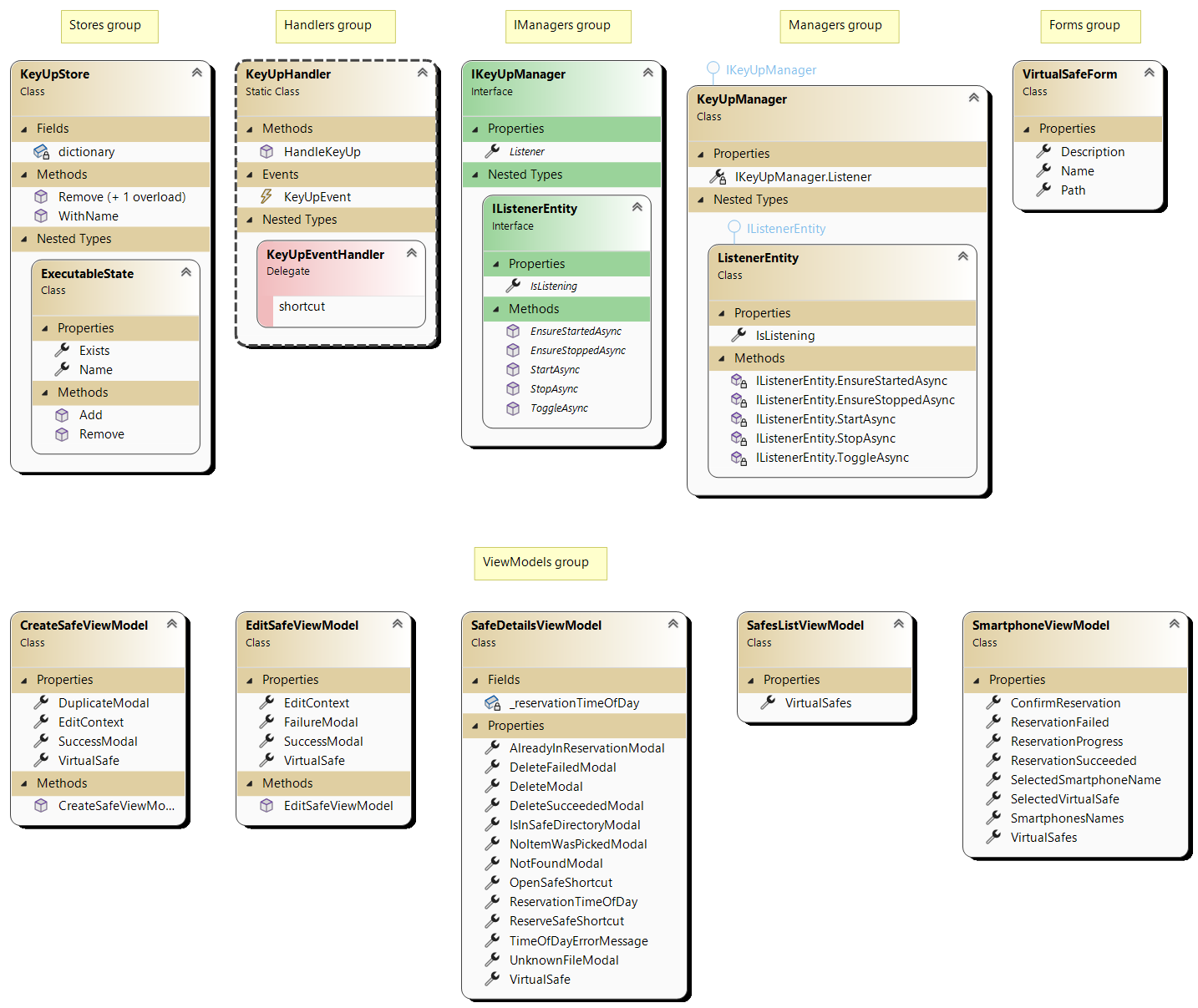


Рисунок 4. Діаграма класів графічного інтерфейсу.

### Опис

Це .NET MAUI Blazor Hybrid проект та у проекті зараз окрім стандартних використовуються такі NuGet package-і:

* CommunityToolkit.Maui.Core,
* Blazor.Bootstrap.

На даний момент графічний інтерфейс має view models для таких сторінок і відповідно такі сторінки:

* **Домашня сторінка** – ***HomePage***,
* **Сторінка списку віртуальних сейфів** – ***SafesListPage***,
* **Сторінка створення віртуального сейфу** – ***CreateSafePage***
* **Сторінка віртуального сейфу** – ***SafeDetailsPage***,
* **Сторінка списку файлів і каталогів для резервування** – немає,
* **Сторінка для резервування смартфонів** – ***SmartphonePage***,
* **Сторінка налаштувань** – ***SettingsPage***.

Кожна view model має приставку ViewModel без приставки Page, як і можна побачити з рисунку 4. Для заповнення і валідування даних віртуального сейфу використовується ***VirtualSafeForm***. ***KeyUpHandler*** відповідає за перехоплення події натискання клавіші клавіатури, якщо клавіша була натиснута, то він запускає методи що були додані до у ***KeyUpStore***. ***IKeyUpManager*** і ***KeyUpManager*** призначенні для запуску чи вимкнення прослуховування клавіатури.

## Фоновий сервіс резервування

### Діаграма класів

Діаграма класів фонового сервісу резервування наведено на рисунку 5.

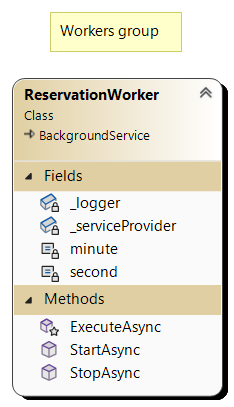


Рисунок 5. Діаграма класів фонового сервісу резервування.

### Опис

Це є проект типу Worker Service і не містить сторонніх NuGet package-ів окрім стандартних.

У проекті фонового сервісу резервування є лише один клас ***ReservationWorker***. Він запускається і працює на фоні, перевіряючи чи настав час резервувати якогось зі всіх віртуальних сейфів. Якщо час резервування настав або вже пройшов, то сервіс зарезервує віртуальний сейф, якщо ні, то ні.

## Доменний рівень

### Діаграма класів

Діаграма класів доменного рівня наведено на рисунку 6.

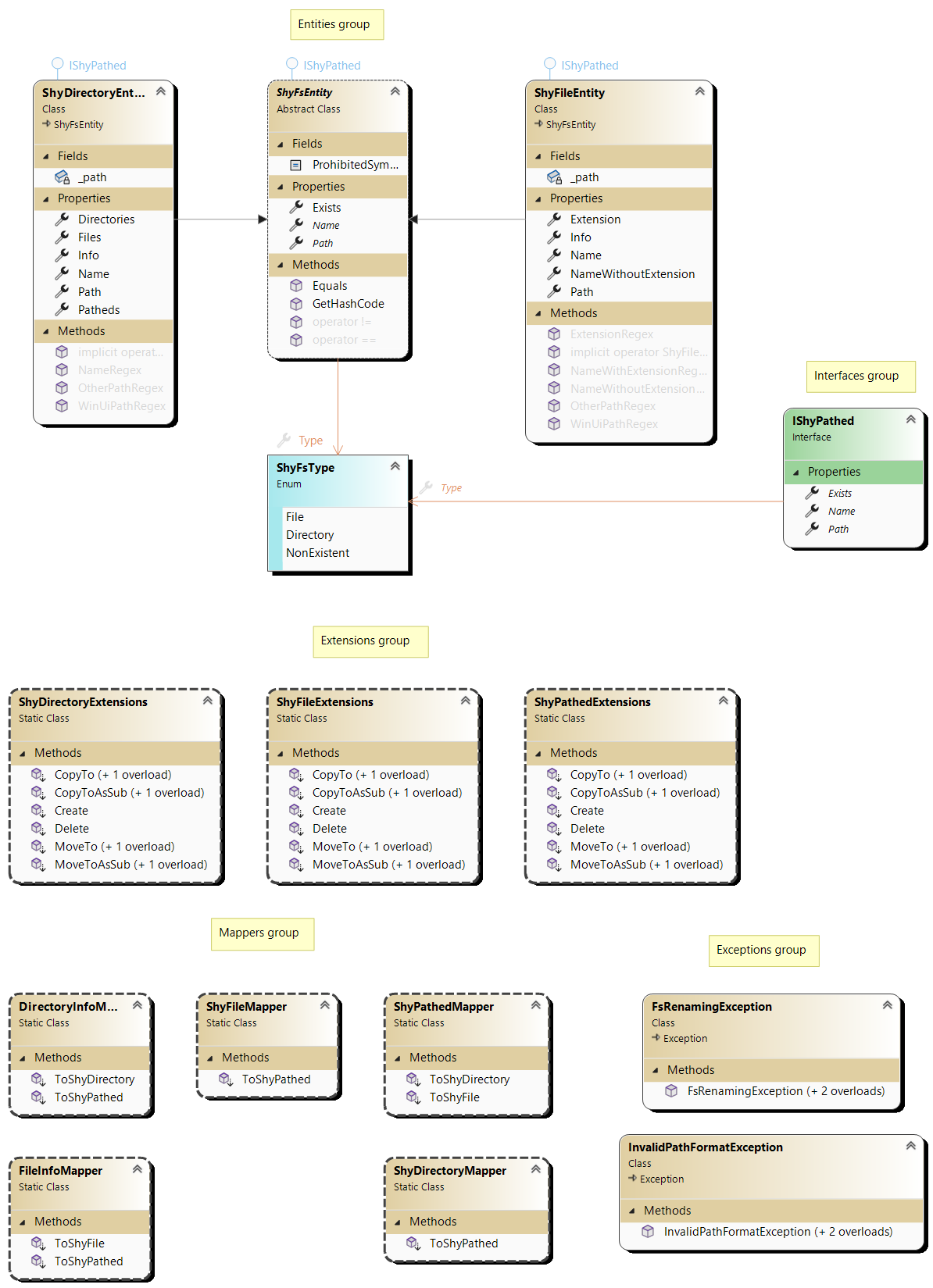


Рисунок 6. Діаграма класів доменного рівня.

### Опис

Це Class Library проект і не містить сторонніх NuGet package-і.

Нижче наведені сутності у відповідності з їх класами, про які було вказано вище при описі доменного рівня.

* **Файл** – ***ShyFileEntity***.
* **Каталог** – ***ShyDirectoryEntity***.
* **Файлова система** – ***ShyFsEntity***.
* **Сутність зі шляхом** – ***IShyPathed*** (інтерфейс).

Створенні extensions класи для реалізації дій над сутностями, які були згадані вище при описі доменного рівня, а саме створення, копіювання, переміщення, видалення. Це класи:

* **ShyFileExtensions** – стандартні операції над ***ShyFileEntity***,
* **ShyDirectoryExtensions** – стандартні операції над ***ShyDirectoryEntity***,
* **ShyPathedExtensions** – стандартні операції над ***IShyPathed***.

У рівні є користувацькі exception-и, які наведено нижче.

* **FsRenamingException** – помилка, яка виникає при неправильному переназиванні файлу, що не допускається файловою системою.
* **InvalidPathFormatException** – помилка, що виникає при неправильному форматі шляху до файлу, що не дозволений файловою системою.

## Програмний рівень

### Діаграма класів

Діаграма класів програмного рівня наведено на рисунку 7.



Рисунок 7. Діаграма класів програмного рівня.

### Опис

Це Class Library проект і не містить сторонніх NuGet package-і.

Нижче наведені моделі у відповідності з їх класами, про які було вказано вище при описі програмного рівня.

* **Віртуальний сейф** – ***VirtualSafeModel***,
* **Деталі віртуального сейфу** – ***VirtualSafeDetailsModel***,
* **Лог** – ***LogModel***,
* **Операція логу** – ***LogOperationModel***,
* **Резервація** – ***ReservationModel***,
* **Файл резервації** – ***FileModel***,
* **Каталог резервації** – ***DirectoryModel***,
* **Комбінація клавіш** – ***ShortcutModel***.

Окрім моделі комбінації клавіш ще є сутність комбінації клавіш, яку представляє клас ***ShortcutEntity***. Функціонал будівника логів представляє інтерфейс ***ILogBuilder***, а функціонал взаємодії з телефоном – ***ISmartphoneService***.

Інтерфейс ***IModelBase<IdType>*** об’єднує різні моделі у одну групу, яка має властивість Id.

Також, присутні сутності, які представляють результат дії методу класу.

* **ResultEntity** – результат, який характеризується станом «успішно», що є bool типом даних і має вказуватися при кінці операції.
* **ValueResult** – результат, що характеризується поверненим результатом та станом «успішно», якщо результат було не повернуто або повернуто, але null чи default, то стан «успішно» є false. Якщо все виконано успішно, то стан «успішно» є true. Повернений результат є dynamic.
* **ValueResult<ValueType>** – результат, що характеризується поверненим результатом та станом «успішно», якщо результат було не повернуто або повернуто, але null чи default, то стан «успішно» є false. Якщо все виконано успішно, то стан «успішно» є true. Повернений результат є ValueType.
* **ObjResult** – результат, що характеризується поверненим результатом та станом «успішно», якщо результат було не повернуто або повернуто, але null чи default, то стан «успішно» є false. Якщо все виконано успішно, то стан «успішно» є true. Повернений результат є object.

## Інфраструктурний рівень

### Діаграма класів

Діаграма класів інфраструктурного рівня наведено на рисунку 7.

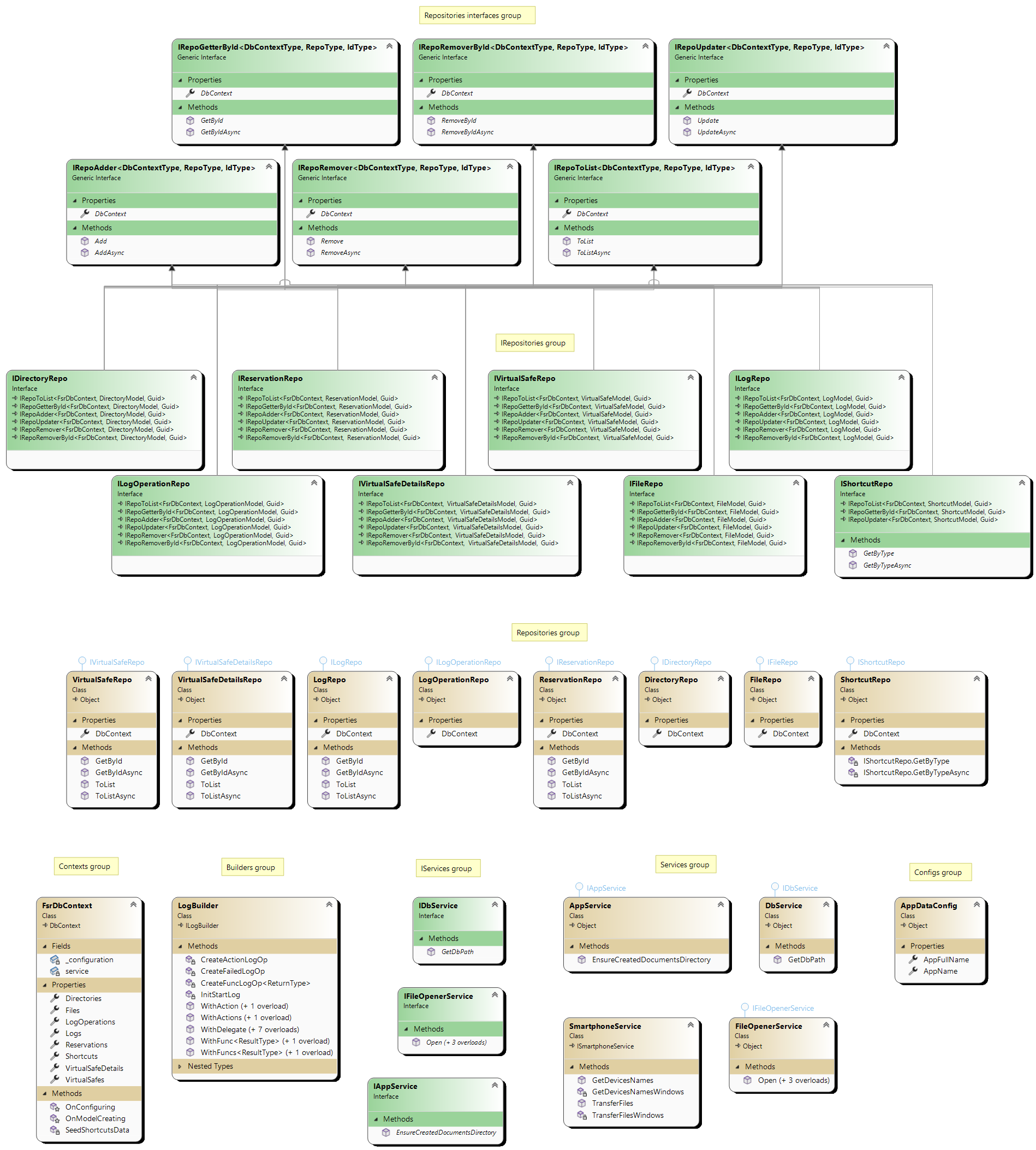


Рисунок 8. Діаграма класів інфраструктурного рівня.

### Опис

Це Class Library проект, у якому зараз використовуються такі NuGet package-і:

* Microsoft.EntityFrameworkCore,
* Microsoft.EntityFrameworkCore.Design,
* Microsoft.EntityFrameworkCore.Tools,
* Microsoft.EntityFrameworkCore.Sqlite.

Контекст бази даних реалізований у класі ***FsrDbContext***, а шлях до згенерованої бази даних знаходиться за допомогою ***DbService*** (інтерфейс IDbService), який використовує клас ***AppDataConfig***. База даних генерується за допомогою класу ***AppService***, який реалізує інтерфейс ***IAppService***. Інтерфейси ***ILogBuilder*** і ***ISmartphoneService*** з програмного рівня реалізуються у класах ***LogBuilder*** і ***SmartphoneService***.

Також, створено інтерфейси, що реалізують стандартну реалізацію CRUD репозиторіїв для ***EF Core***. Вони наведені нижче.

* **IRepoAdder** – зберігає стандартний функціонал додавання нового запису у таблицю бази даних.
* **IRepoUpdater** – зберігає стандартний функціонал оновлення запису у таблиці бази даних.
* **IRepoRemover** – зберігає стандартний функціонал видалення запису з таблиці бази даних.
* **IRepoRemoverById** – зберігає стандартний функціонал видалення запису з таблиці бази даних за id полем.
* **IRepoGetterById** – зберігає стандартний функціонал отримання запису з таблиці бази даних за id полем.
* **IRepoToList** – зберігає стандартний функціонал отримання всіх записів з таблиці бази даних.

Відповідно, використовуючи вище описані інтерфейси створив такі інтерфейси репозиторіїв та їхні імплементації у класах, що наведено нижче.

* **IVirtualSafeRepo** і **VirtualSafeRepo** – реалізують взаємодію із записами таблиці віртуальних сейфів.
* **IVirtualSafeDetailsRepo** і **VirtualSafeDetailsRepo** – реалізують взаємодію із записами таблиці деталей віртуальних сейфів.
* **ILogRepo** і **LogRepo** – реалізують взаємодію із записами таблиці логів.
* **ILogOperationRepo** і **LogOperationRepo** – реалізують взаємодію із записами таблиці операцій логів.
* **IReservationRepo** і **ReservationRepo** – реалізують взаємодію із записами таблиці резервацій.
* **IFileRepo** і **FileRepo** – реалізують взаємодію із записами таблиці файлів резервації.
* **IDirectoryRepo** і **DirectoryRepo** – реалізують взаємодію із записами таблиці каталогів резервації.
* **IShortcutRepo** і **ShortcutRepo** – реалізують взаємодію із записами таблиці комбінацій клавіш.

Варто зазначити, що не всі з вище перелічених інтерфейсів і їхніх імплементацій використовують всі інтерфейси стандартних реалізацій репозиторіїв. Деякі використовують лише деякі стандартні реалізації, а деякі мають власні особливі методи, які описані в інтерфейсі і реалізовані у відповідному класі, або мають стандартні реалізації, які переписати і класі, що імплементовує інтерфейс.