**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**
**ІМЕНІ ТАРАСА ШЕВЧЕНКА**
Факультет комп'ютерних наук та кібернетики
Кафедра теорії та технології програмування

**Кваліфікаційна робота**
**на здобуття ступеня бакалавра**
за спеціальністю 122 Комп'ютерні науки
на тему:

# РОЗРОБКА МОБІЛЬНОЇ ІНФОРМАЦІЙНО-ДОВІДКОВОЇ СИСТЕМИ ДЛЯ АГРОБІЗНЕСУ

Виконав студент 4-го курсу
Дуйловський Ігор Андрійович
Науковий керівник:
доцент, кандидат технічних наук
Ткаченко Олексій Миколайович

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент

Роботу розглянуто й допущено до захисту на засіданні кафедри теорії та технології програмування
«01» червня 2022 р., протокол № 10
Завідувач кафедри
Микола НІКІТЧЕНКО

**Київ – 2022**

## РЕФЕРАТ

Обсяг роботи 35 сторінок, 19 ілюстрацій, 13 джерел посилань.

ЗАСТОСУНОК, IOS, APPLE, CLASS, ZEPLIN, ДИЗАЙН, МАГАЗИН, ІНТЕРФЕЙС.

Назва роботи: Розробка мобільної інформаційно-довідкової системи для агробізнесу

Об’єктом роботи є процес розробки мобільного інформаційно-довідкового додатку. Предметом роботи є готовий додаток.

Мета роботи: розширити знання в області мобільної розробки, конкретно інструментарію для пристроїв компанії Apple[1], навчитися писати інформаційні додатки, розробити програму, яка буде максимально зручна та корисна користувачу.

План роботи: поглибити знання в створенні мобільних додатків, створити зручний та корисний додаток для агрономів. Розібратися як має виглядати інтерфейс користувача та MVC модель в даному наборі технологій. Розібратися в користувацьких пакетах та бібліотеках, які будуть використовуватись в процесі розробки.

Результат роботи: робочий мобільний додаток для агрономів. В якому можна подивитися дані для захисту культур, погоду, останні новини та статті в сфері агрономів. Програма вміє оновлювати дані з серверу, та взаємодіяти з різними API для знаходження своєчасної інформації для користувача.

**Зміст**

## СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

МП – мова програмування;

ПК – персональний комп’ютер;

ОС - операційна система;

IDE - Integrated Development Environment;

MVC - Model-view-controller;

ПІБ - прізвище ім'я по-батькові;

API - Application Programming Interface;

# ВСТУП

**Оцінка сучасного стану об’єкта розробки.** Мобільний ринок захоплює все більше і більше позицій через пришвидшення сучасного темпу життя. Далеко не всі в наш час мають стаціонарні комп'ютери та надають перевагу саме мобільним пристроям. Що особливо актуально для сільсько-господарської діяльності. В разі знаходження проблеми можна одразу подивитися інформацію про неї, а якщо за фото чи описом проблема не зрозуміла, завжди можна зателефонувати спеціалісту та отримати консультацію.

**Актуальність роботи та підстави для її виконання.** Агрокультурна галузь завжди буде актуальною, адже її продукти використовуються усюди. Якість вирощених культур впливає на смак їжі людей, на стан здоров'я усієї країни, а також на ціну при експорті за кордон. Тому надзвичайно важливо слідкувати, вчасно лікувати та вирішувати проблеми рослин. З ідентифікацією проблем та знаходженням способів їх вирішення допоможе мобільний додаток.

**Мета і завдання роботи.** Метою кваліфікаційної роботи є розширення знань в області та розробка мобільного додатку. Для цього поставленні такі завдання:

- Аналіз інструментарію, необхідного для розробки
- Створення ядра додатку
- Планування та реалізація користувацького інтерфейсу згідно з дизайном
- Підключення інтернет ресурсів та логіки до готового інтерфейсу
- Викладення готової програми в інтернет

**Об'єкт, методи й засоби розроблення.** Об’єктом роботи є реалізація мобільного додатку засобами мови програмування Swift[2],

використовуючи IDE Xcode[3]. Під час розробки використовувався шаблон MVC. Для розробки інтерфейсів використовувався Interface Builder[4], а для дизайну - програма Zeplin[5].

**Можливі сфери застосування.** Сфера застосування додатку - агробізнес. Як для агрономів і фермерів, так і для людей зацікавлених проблемами рослин.

**Подальший розвиток.** Програма розроблена з урахуванням подальшої підтримки та розвитку. Реалізовано зовнішнє оновлення даних, без необхідності оновлення додатку. Також в майбутньому може додаватися більший функціонал, наприклад можливість відразу купувати необхідні продукти, передивлятись відео чи особистий кабінет.

# РОЗДІЛ 1. ОГЛЯД НАБОРУ ІНСТРУМЕНТІВ ТА ОСОБЛИВОСТЕЙ ОБРАНОЇ СИСТЕМИ

## 1.1 Особливості компанії Apple з точки зору розробника

Компанія Apple не потребує представлення, в наш час їх продукція займає значну частку всього ринку мобільних пристроїв(див. рис. 1).

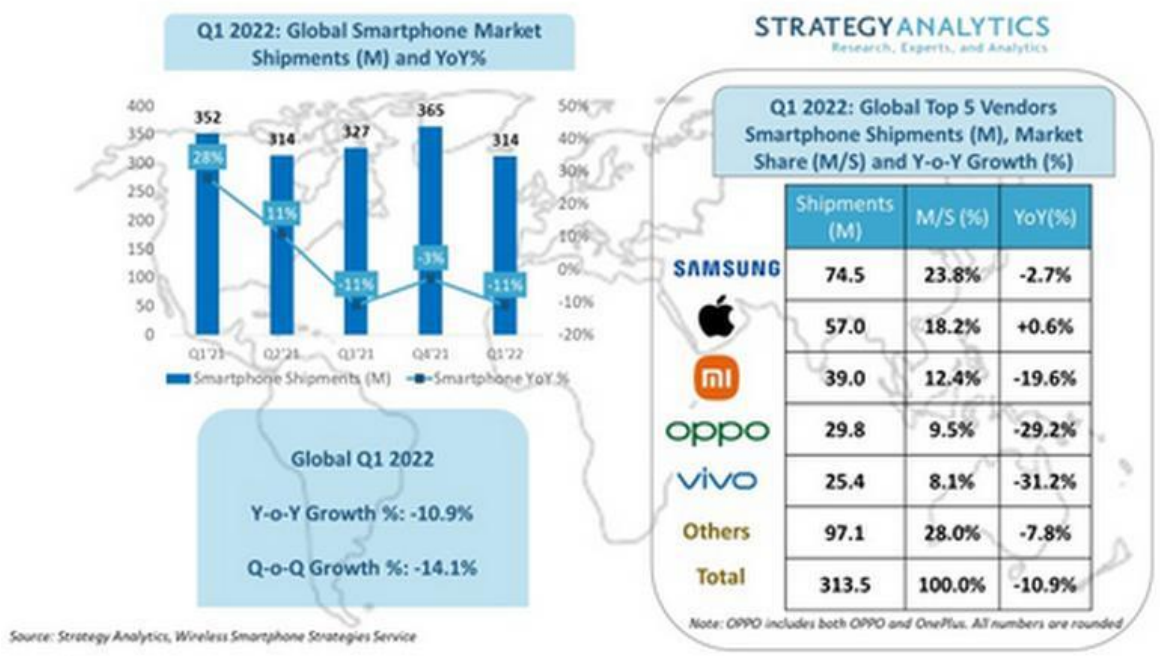



**Рисунок 1.** Аналітика мобільних виробників

Для своїх пристроїв програмісти компанії розробили спеціальну закриту систему: iOS[6]. На відміну від другої найбільш популярної системи: Android[7] вона має закритий програмний код та більш розвинені системи безпеки. Через це процес розробки додатків має свої особливості.

Під час розробки для системи Android потрібен лише .apk[8] файл та програма встановиться на пристрій. Для розробки під iPhone[9] потрібен сертифікат розробника, який встановлюється в його аккаунт. Для його отримання потрібно активувати Apple Developer Program, платну підписку, яка необхідна для доступу до Apple Developer Center[10](див. рис. 2), та поширення додатків у Appstore, спеціалізований магазин додатків. Після чого отримується право розробляти, але для поширення потрібен ще один

сертифікат - сертифікат публікації. Тільки тоді з'являється можливість поширити додаток. Після архівування програми та її загрузки автоматичні системи або люди перевіряють функціональність та працездатність додатку, правильність вказаних на його сторінці даних(див. рис. 3) та необхідність запитуваних дозволів. Ретельно перевіряється для чого, наприклад, додатку знадобилася геолокація пристрою чи список контактів користувача.

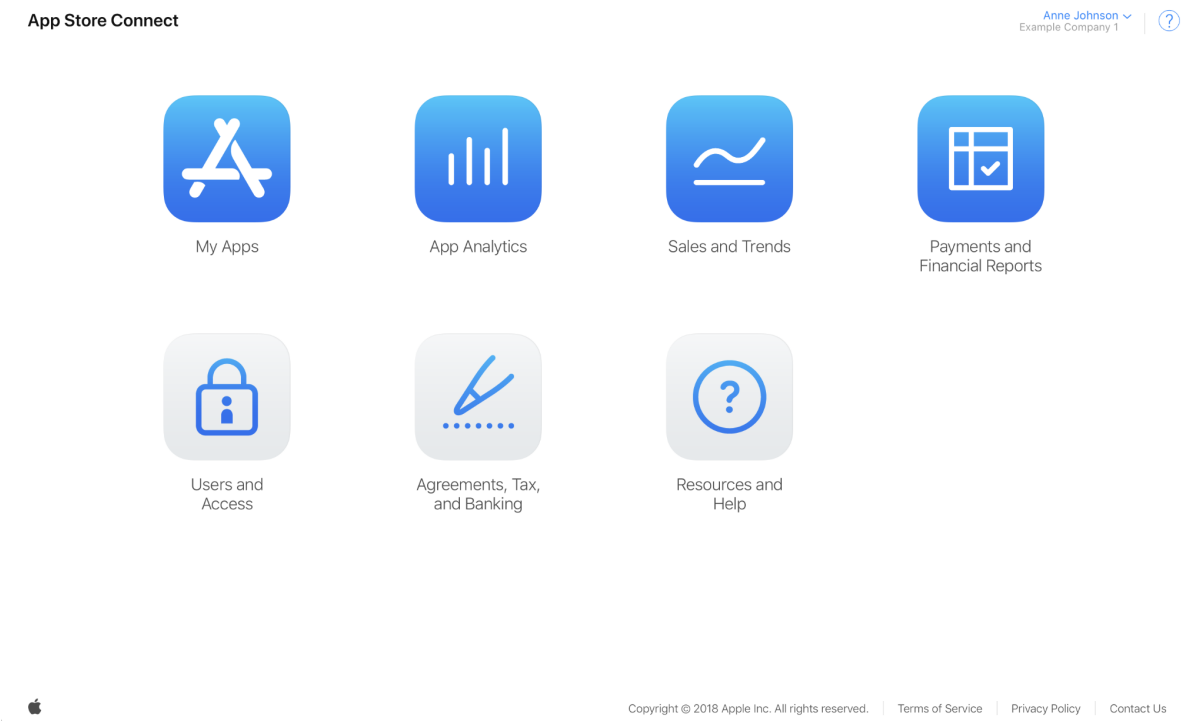



**Рисунок 2.** App Store Connect

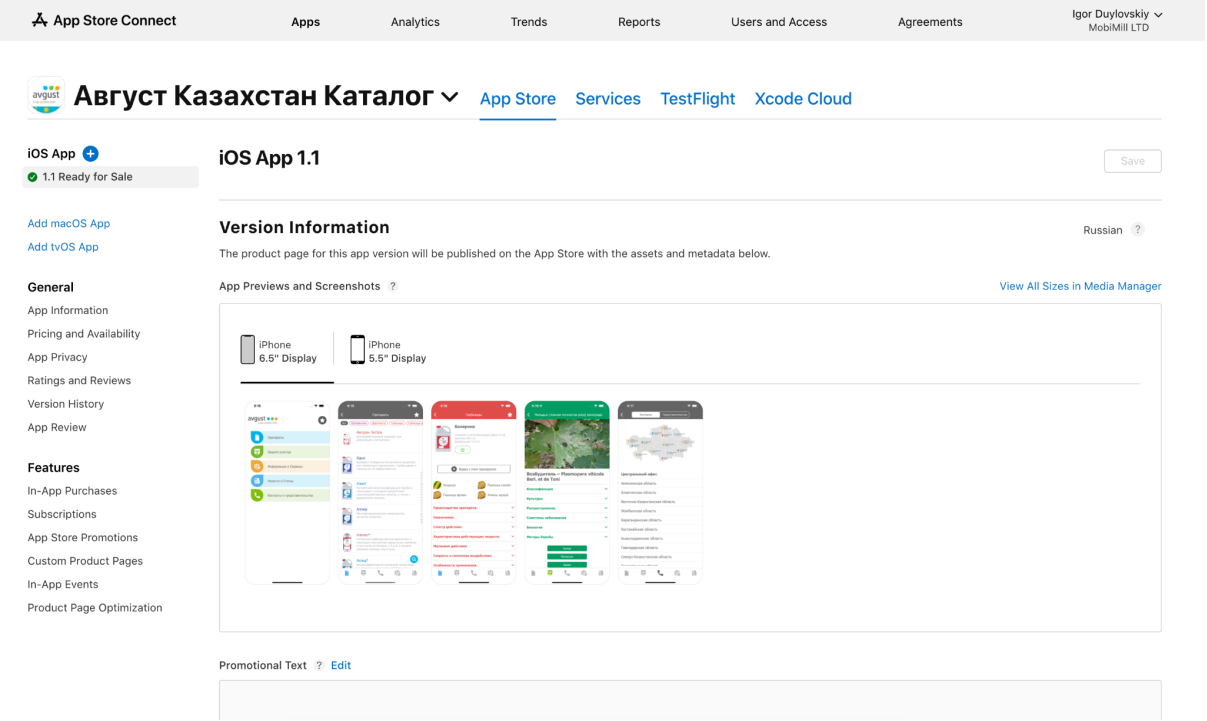



**Рисунок 3.** Сторінка застосунку

## 1.2 Огляд можливостей та інструментарію IDE XCode

Для написання коду в основному використовується IDE під назвою XCode. Вона доступна лише на пристроях компанії Apple, на різних моделях ПК та ноутбуків. Крім редактору коду вона допомагає з сертифікацією акаунтів, може автоматично створювати необхідні(якщо є аккаунт розробника). Також є досить зручний редактор інтерфейсів(див. рис. 4).

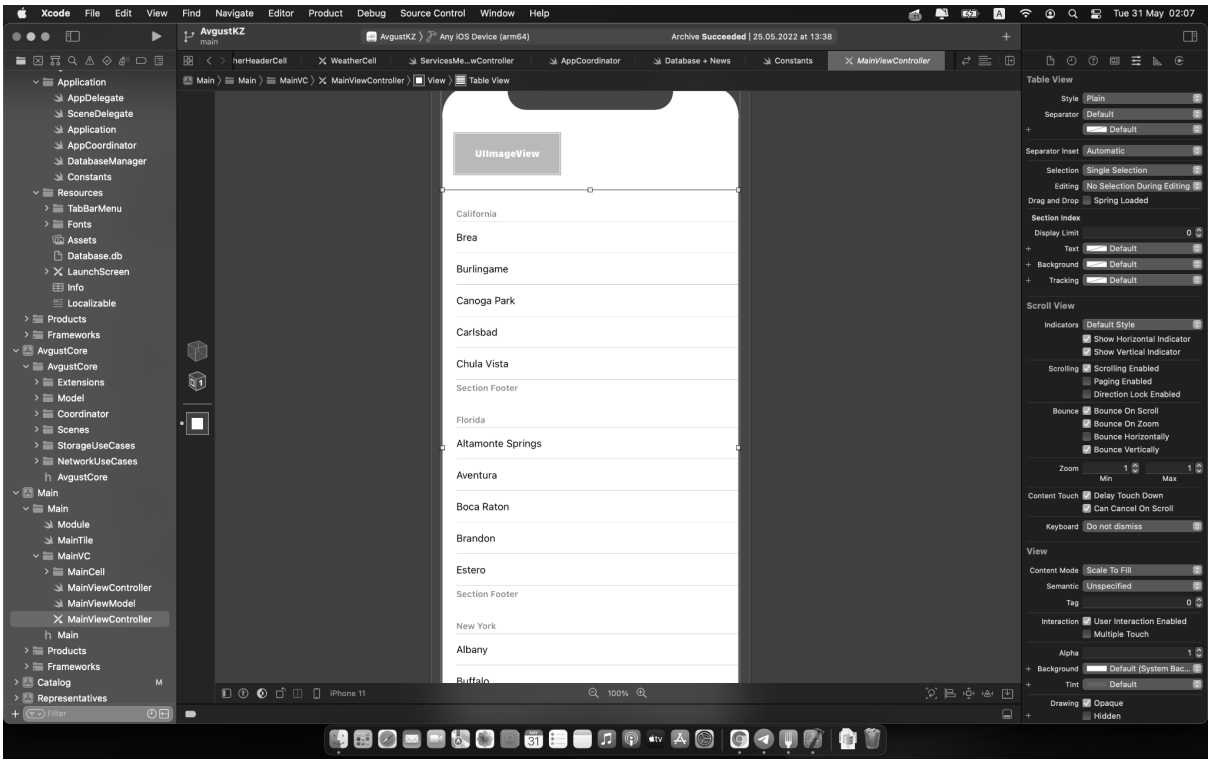



**Рисунок 4**. Конструктор інтерфейсів

В ньому можна досить зручно створювати, змінювати сторінки користувача та зв'язувати їх між собою. З правого боку панель з набором властивостей кожного елементу для точного налаштування.

Також IDE предоставляє набір симуляторів різних моделей iPhone для тестування готової програми і має всі необхідні для запуску інструменти. Тож можна одним кліком запустити програму на одному з симуляторів, реальному телефоні(який підключено до комп'ютера) чи одразу створити та надіслати в App Store готовий продукт.

### 1.3 Мова програмування Swift

Для написання додатків під iOS використовують дві мови програмування: Swift та Objective C. Обрано перший варіант, бо це новіша МП, яку активно просуває та розвиває сама Apple.

Swift – це надійна та інтуїтивно зрозуміла мова програмування від Apple, за допомогою якої можна створювати програми для iOS, Mac, Apple TV та Apple Watch. Він надає розробникам небувалу свободу творчості.

Завдяки цій простій та зручній мові з відкритим кодом вам досить цікавої ідеї, щоб створити щось неймовірне. [11]

Як завіряють самі розробники, вона в 2,6 разів швидше за Objective C та у 8,4 рази швидше за Python. Також, у вже готові додатки на “C” можна встроїти код Swift, якщо це потрібно.

# РОЗДІЛ 2. СТВОРЕННЯ КОРЕНЕВИХ МОДУЛІВ

При створенні проекту IDE предоставляє нам пустий проект з одною пустою сторінкою. Створюється клас додатку, файл інтерфейсу(далі Xib) для екрану запуску та папка з ассетами, тобто необхідними матеріалами, як картинки чи кольори.

## 2.1 Структура проекту

Для подальшого розвитку додаток поділено на змістовні модулі. Вони представляють собою окремі проекти, які головний модуль використовує як бібліотеки. Це зроблено для подальшого розвитку програми та використання модулів для інших регіонів. Конкретно цей додаток створено саме для Казахстану, але в подальшому аналогічні будуть для Молдови та інших країн. Тоді замінюючи базу даних та деякі інші дані можна без змін використовувати вже готові модулі. Так як необхідні картинки, текст з локалізацією, шрифти, база даних та інша інформація зберігається саме у головному модулі.

Програма розбивається на 8 модулів(див. рис. 5):

- Головний модуль з необхідною інформацією
- Ядро, яке зберігає в собі моделі даних з бази, розширення елементів інтерфейсу, а також інтерфейси для ключових елементів програми
- Модуль з головною сторінкою
- Модуль “Каталогу”, де зберігаються сторінки з продуктами, культурами і шкідниками, та інші функції до цих сторінок(“Обране”, пошук т.п.)
- Модуль з контактною інформацією та представниками кожної області
- Модуль з новинами та статтями

- Модуль для роботи з базою даних
- А також для роботи з інтернетом

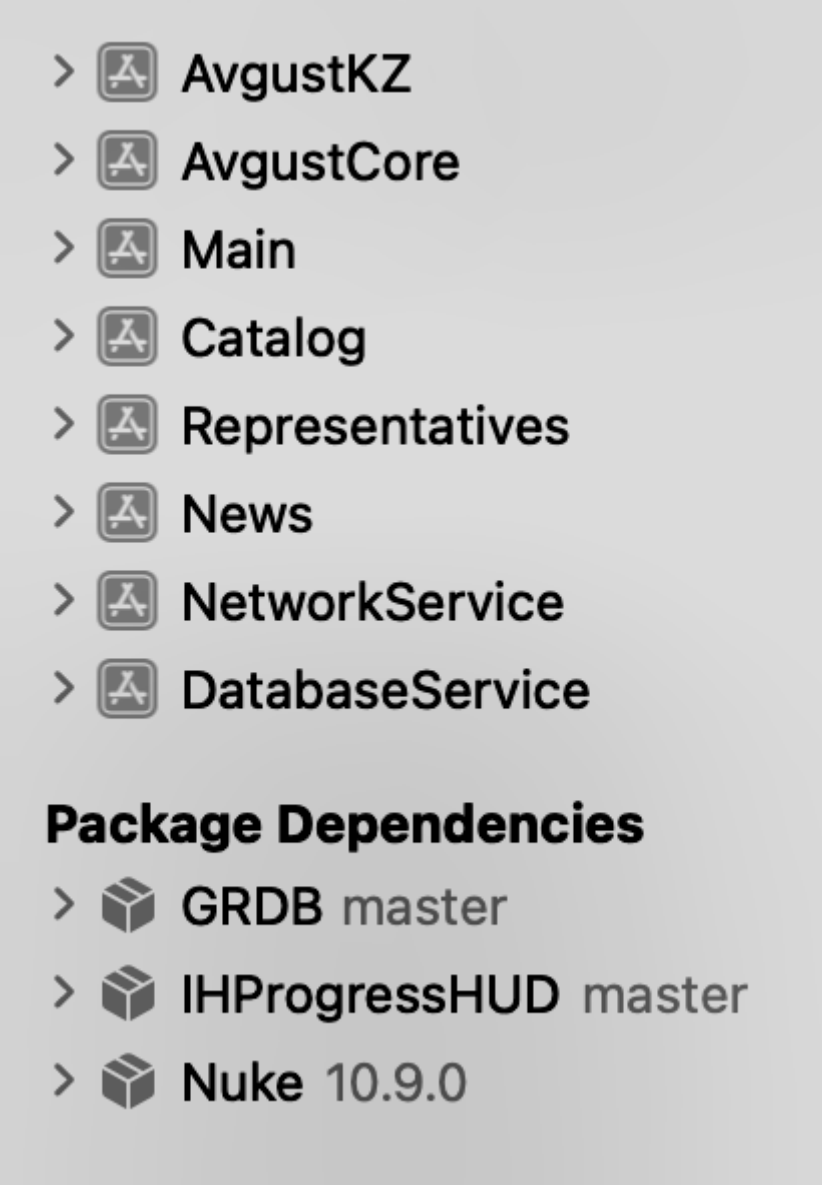



**Рисунок 5.** Структура застосунку

Також окремо підключено декілька користувацьких бібліотек для полегшення роботи з кодом. Для їх підключення використано встановлений в IDE менеджер бібліотек. Це відносно нова особливість, яка є кращою альтернативою застарілого CocoaPods[12]. “Подси” це окремий модуль додатку в якому трималися всі додані бібліотеки, як частина проекту. Для його встановлення потрібно окремо встановлювати деякі додатки та функціональні розширення, далі через консоль створювати

окремий файл з назвами усіх необхідних бібліотек та через консоль завантажувати їх. До того ж встановлення буде лише один раз, тобто у разі виходу оновлення, його необхідно завантажити заново.

В свою чергу встановлений менеджер позбавляє більшості з цих складностей. Необхідно лише нажати “Додати нову” і вказати посилання на необхідну бібліотеку. Далі вона буде встановлена, і автоматично оновлюється в разі необхідності. А також вони є окремою сутностю, а не модулем програми, тож помилки чи застереження з бібліотеки не будуть показувати розробнику. Це полегшує розуміння скільки проблем саме у додатку та дарує насолодження програми, написаної добре і без попереджень.

У додатку використовувались три користувацькі бібліотеки:

- GRDB - зручний спосіб роботи з базою даних
- IHProgressHUD - для відображення зручних і красивих віконець завантажень(як наприклад оновлення бази)
- Nuke - для відображення картинок з інтернету одною командою

**2.2 Робота з базою даних**

Створено систему взаємодії з базою даних. Її представляє класс DatabaseManager. Його задача перевіряти актуальність бази даних. При запуску додатку він перевіряє версії бази яка наявна в комплекті, початково встановленої та версію яка є на сервері. Після перевірки встановлює найновішу в файли додатку, а також додає на екран віконце прогресу завантаження, якщо потрібно.

Також для роботи з даними всередині бази створено окремий модуль. Клас DatabaseService спрощує написання запитів до бази. Він відкриває активну базу, встановлює локалізацію та має декілька допоміжних методів для опрацювання запитів. Використовуючи можливості бібліотеки GRDB,

він опрацьовує помилки в запитах та записує до дебаг консолі журнал роботи. Також він є реалізацією інтерфейсу UseCaseProvider. Він в свою чергу є наслідує низку інших інтерфейсів, кожен з яких відповідає за свою область запитів. Як приклад, інтерфейс ProductUseCase відповідає за екрани каталогу, тобто список продуктів, та всі необхідні для цього дані. З себе він представляє декілька методів з SQL запитом, який виконується засобами DatabaseService(див. рис. 6).

```
public func culturesForProduct(productId: Int) -> [Crop] {
    let query = """
        SELECT
            Culture_i18n.cultureId as id,
            Culture_i18n.name as name,
            Culture.picture
        FROM
            Culture
            LEFT JOIN Culture_i18n ON Culture.id = Culture_i18n.cultureId
            LEFT JOIN CultureForProduct ON Culture.id = CultureForProduct.cultureId
        WHERE
            Culture_i18n.i18n = '\(locale)' AND
            CultureForProduct.productId = '\(productId)'
        ORDER BY
            name ASC
    """

    return fetch(query)
}
```

**Рисунок 6.** Приклад методу з ProductUseCase

Цей запит потрібен для знаходження усіх культур, для яких використовується даний продукт. Метод приймає id продукту та повертає список усіх культур. Також, через локалізацію, необхідно використовувати змінну “locale” з DatabaseService. Для детальної інформації про культуру різними мовами створено окрему таблицю “Culture_i18n”. Аналогічні таблиці є для більшості сутностей. Частину схеми бази даних, як приклад приведено в додатку А.

### 2.3 Взаємодія з інтернет ресурсами

Аналогічно базі даних, реалізована робота з інтернет ресурсами. Створено головний клас NetworkService, який відповідальний за виконання та обробку інтернет запитів. Він реалізує інтерфейс NetworkProvider, який

в свою чергу залежить від трьох “кейсів”: для бази даних, новостей та статей, а також погоди. Також реалізована обробка помилок, створено декілька необхідних для цього “перерахувань”. Список методів запиту та список помилок з локалізацією.

### 2.4 Координатор для сторінок

Один з головних класів у додатку це клас Coordinator. Він відповідає за створення та перехід між сторінками. Створюється при запуску програми і розділяється на координатори для кожного з модулів. Також він відповідальний за створення панелі вкладок і зберігає в собі змінні для доступу до DatabaseService і NetworkService. При створенні сторінки передає себе, як параметр, та дозволяє викликати свої методи, тобто переходити на іншу сторінку. Також він має прямий доступ до NavigationController, який відповідає за верхню панель та всі сторінки.

### 2.5 Використання внутрішніх ресурсів

Створено файл локалізації(див. рис. 7) для всього тексту програми. Через особливості аудиторії користувачів доступна локалізація російською та казахською мовами. На жаль, переклад казахською ще не реалізовано. Відповідно на кожній сторінці використовується не текст, а саме посилання на відповідну строчку в файлі локалізації(див. рис. 8).

```
24 //MARK: - Main Module
25 "Main.Products" = "Препараты";
26 "Main.Crops" = "Защита культур";
27 "Main.Videos" = "Видео";
28 "Main.Services" = "Информация и Сервисы";
29 "Main.News" = "Новости и Статьи";
30 "Main.Contacts" = "Контакты и представительства";
31 "Main.Info" = "Информация";
32 "Main.Account" = "Личный кабинет";
```

**Рисунок 7.** Приклад з файлу локалізації

```
case .products:
    return "Main.Products".localized()
```

**Рисунок 8.** Використання локалізованої назви

Також в головному модулі програми додано файли з необхідними шрифтами і вбудована база даних.

В початкову папку “Assets” додані усі необхідні картинки, іконки, а також стандартні кольори. Це дозволяє легко використовувати кольори в інших модулях, а також змінювати колір елементу в усьому додатку, без необхідності змінювати його на кожному екрані.

**2.6 Модуль в модулі**

Для кожного з модулів програми створено клас “Module”. Саме він створює готові контролери. Всі методи кожного з модулів відповідають за створення ViewModel та ViewController, віддаючи вже готову до застосування сторінку координатору.

# РОЗДІЛ 3. СТВОРЕННЯ СТОРІНОК

У цьому розділі буде детально розібрано кожну сторінка кожного модулю.

## 3.1 Модуль каталогу та деталей продукту

Модуль каталогу найбільший та складається з 4 частин. Перша це список усіх продуктів та сторінка кожного. Сторінка зі списком створюється автоматично, після отримання списку всіх продуктів з бази даних. На екрані знаходиться елемент таблиця. Вона заповнюється шаблонними клітинками, які описані окремо(див. рис. 9,10). Для початку вказується кількість рядків у таблиці, далі для кожного рядку створюється копія клітинки та заповнюється даними відповідного продукту.

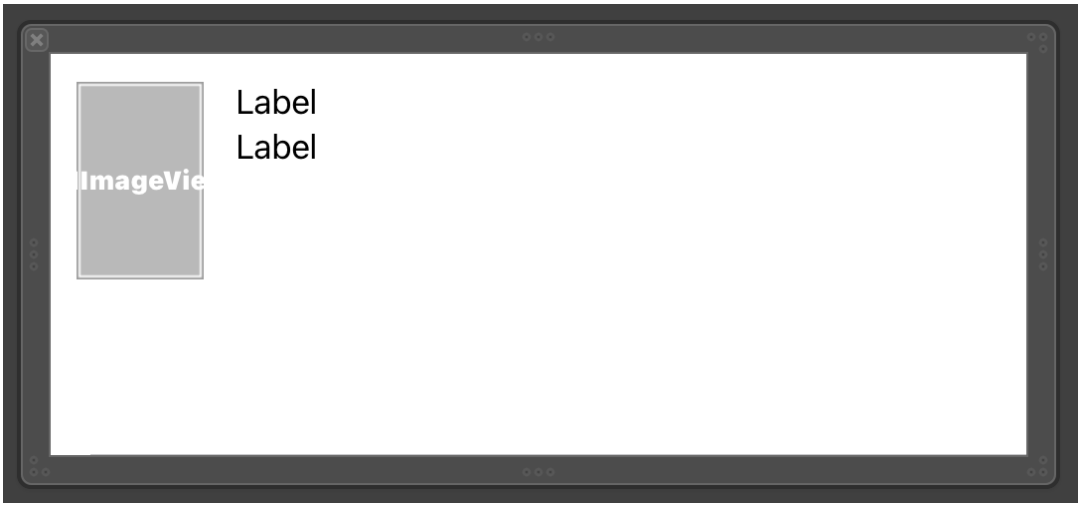



**Рисунок 9.** Вигляд шаблону клітинки

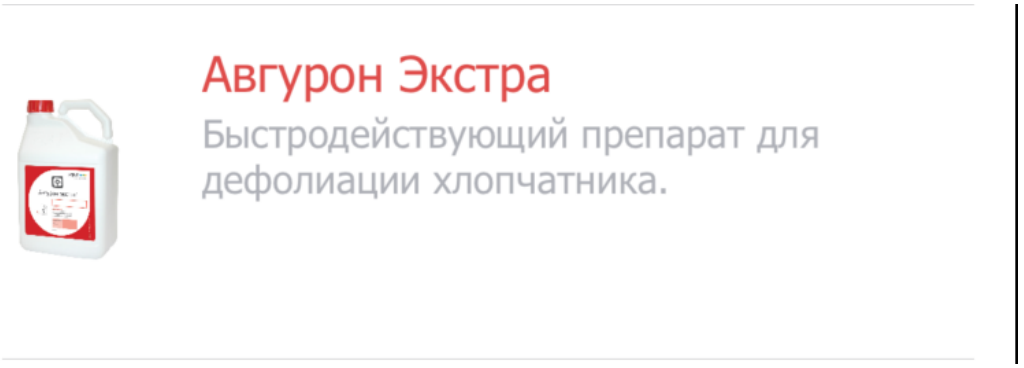



**Рисунок 10**. Вигляд заповненої клітини

Аналогічним чином виконана більшість елементів у додатку. В базі даних окрема таблиця відповідає за зображення продуктів. Вони знаходяться на сервері, а в базу знаходяться посилання на конкретне зображення. За допомогою бібліотеки Nuke зображення асинхронно завантажується до відповідного елементу клітинки. Так як кожна з них це окремий клас, в головному контролері необхідно лише викликати метод “configure" та надати йому необхідну інформацію. Метод налаштує зовнішній вигляд, кольори, шрифт та дані, передаючи повністю готову клітинку таблиці(див. рис. 11).

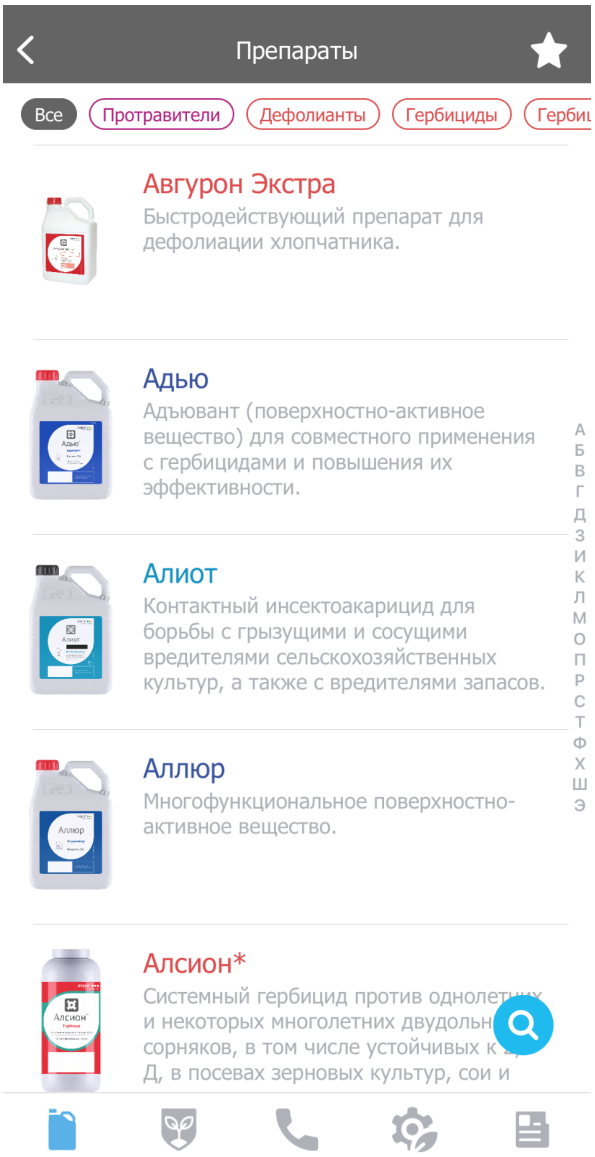



**Рисунок 11.** Вигляд сторінки зі списком продуктів

Також реалізовано малий пошук за алфавітним вказівником. В правій частині знаходиться маленька абетка, при тапу на яку таблиця опускається до необхідної букви. Літери для абетки беруться з першої букви кожного препарату. Якщо ж препарату з якоюсь літерою немає, її не буде і в абетці.

Другий варіант пошуку, розширений, це інший змістовий модуль. Відкривається інший екран з можливістю вибрати проблему, діючу речовину чи культуру для якої потрібні продукти. За допомогою делегата вибір користувача передається контролеру зі списком всіх продуктів, де залишаються лише відсортовані за запитом.

При тапу на зацікавивший препарат виконується перехід на сторінку деталей конкретного продукту(див. рис. 12). Він складається з такої ж таблиці з деякими надбудовами. Вся потрібна інформація береться з бази даних. В верхній частині мається скорочена інформація, інструкція, сертифікат, підтверджуючий що це реальний продукт та можливість додати до обраного. Далі список культур, з піктограмами і розділи з інформацією, які можна згорнути. Від попередньо описаного методу є деяке покращення. В цій таблиці мається декілька можливих клітинок, під різні цілі. Клітинка з культурами, чи клітинка з можливістю згорнути, як приклад.

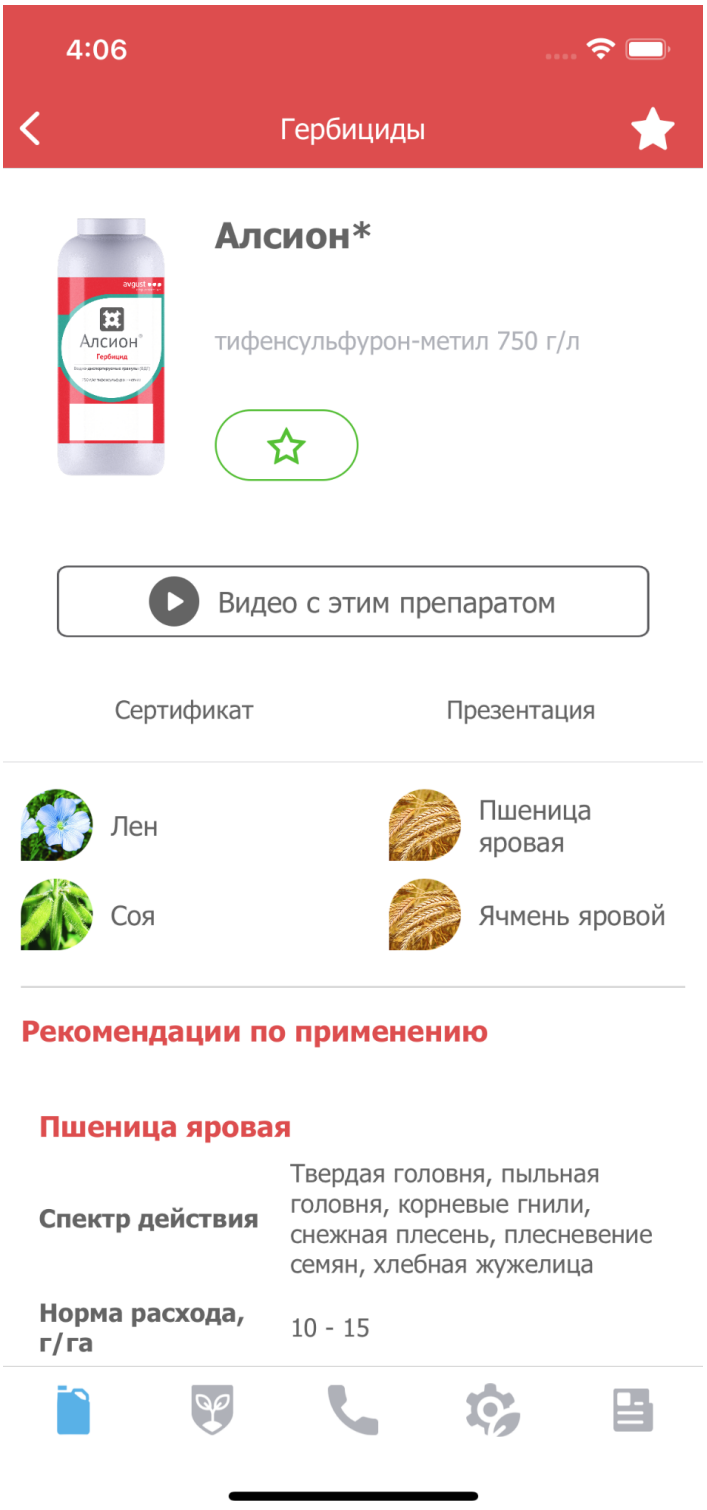



**Рисунок 12.** Деталі продукту

Третій змістовий модуль, це сторінка з культурами. Аналогічний список культур, з можливістю перейти до її деталей. Також ця сторінка є другою вкладкою в нижньому меню.

Деталі культури представляють собою екран з трьома можливостями. Переглянути список продуктів, які можуть взаємодіяти з нею, переглянути схему захисту(див. рис. 10), а також переглянути список шкідників, хвороб чи бур'яну.


**Рисунок 13.** Схема захисту, розбита на кожен етап росту культури

Останній модуль представляє собою таку ж сторінку як деталі продукту, тільки для шкідників(див. рис. 14). Там є колекція фото проблеми, інформація про неї, та список продуктів, проти неї. Кожний з продуктів веде до сторінки з деталями про себе.

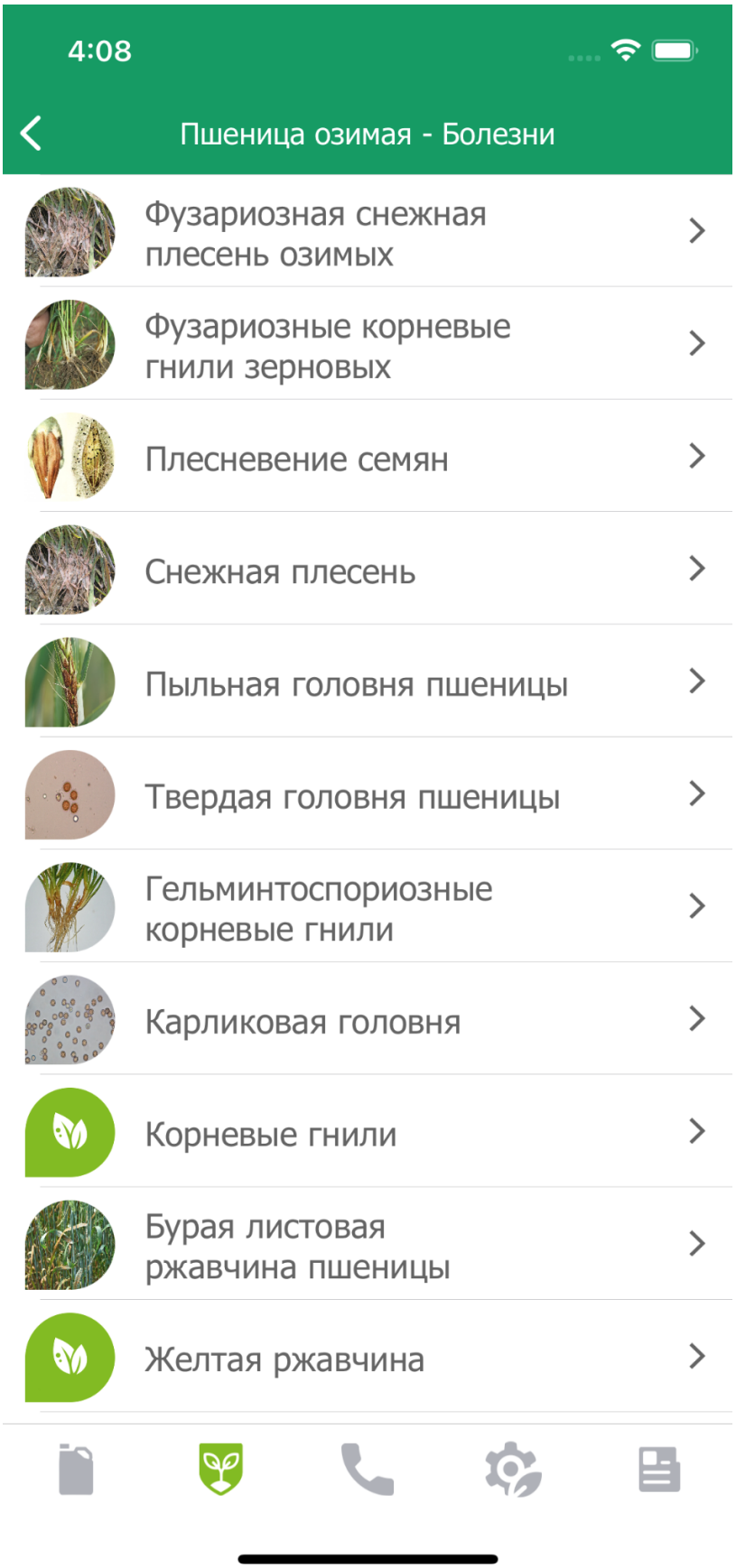



**Рисунок 14.** Список захворювань озимої пшениці

### 3.2 Розділ з контактною інформацією

Третя вкладка меню це контактна інформація. При вході до цієї сторінки користувач бачить карту Казахстану, та список областей. Якщо натиснути на карту, список відсортується під необхідний регіон. Це досягнуто “двойним дном”. Під однією картою знаходиться друга, де кожен регіон позначено окремим кольором. При натисканні перша карта ігнорується, а колір в місці тапу порівнюється з прописаними а базі(див. рис. 15).

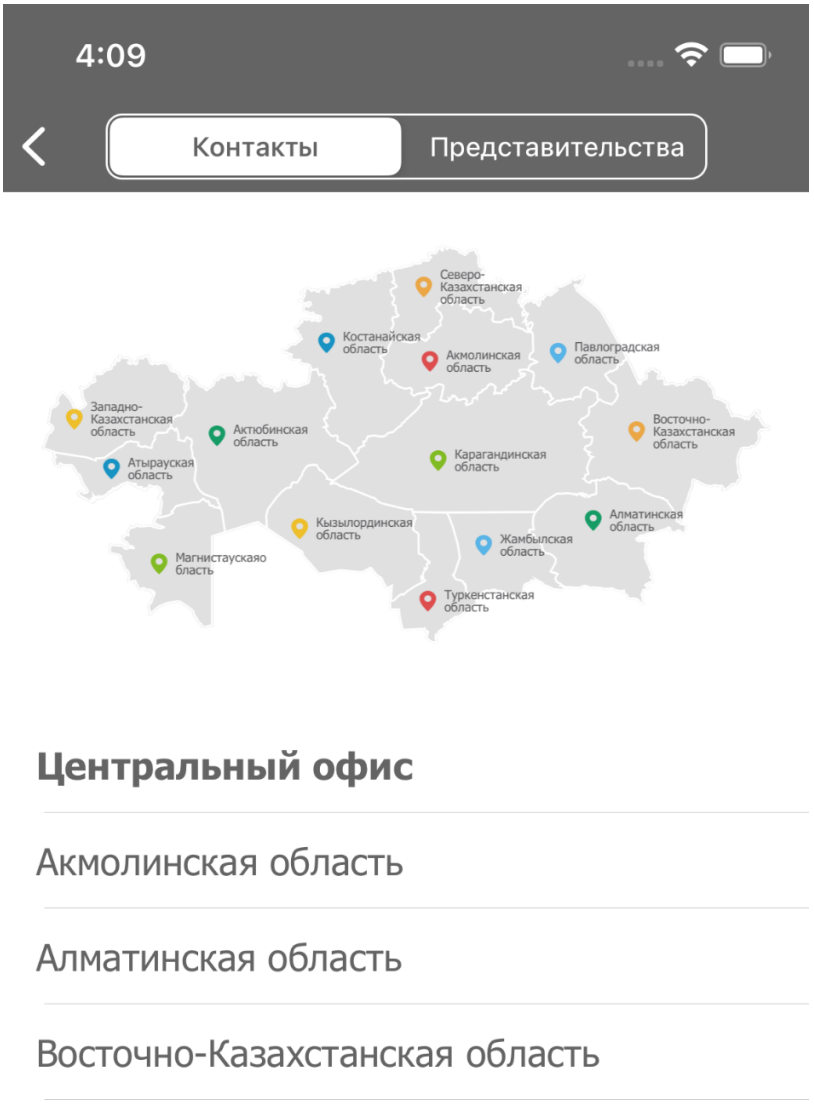



**Рисунок 15.** Сторінка контактів

При тапу на область у списку відкривається екран з доступними представниками(див. рис. 16).

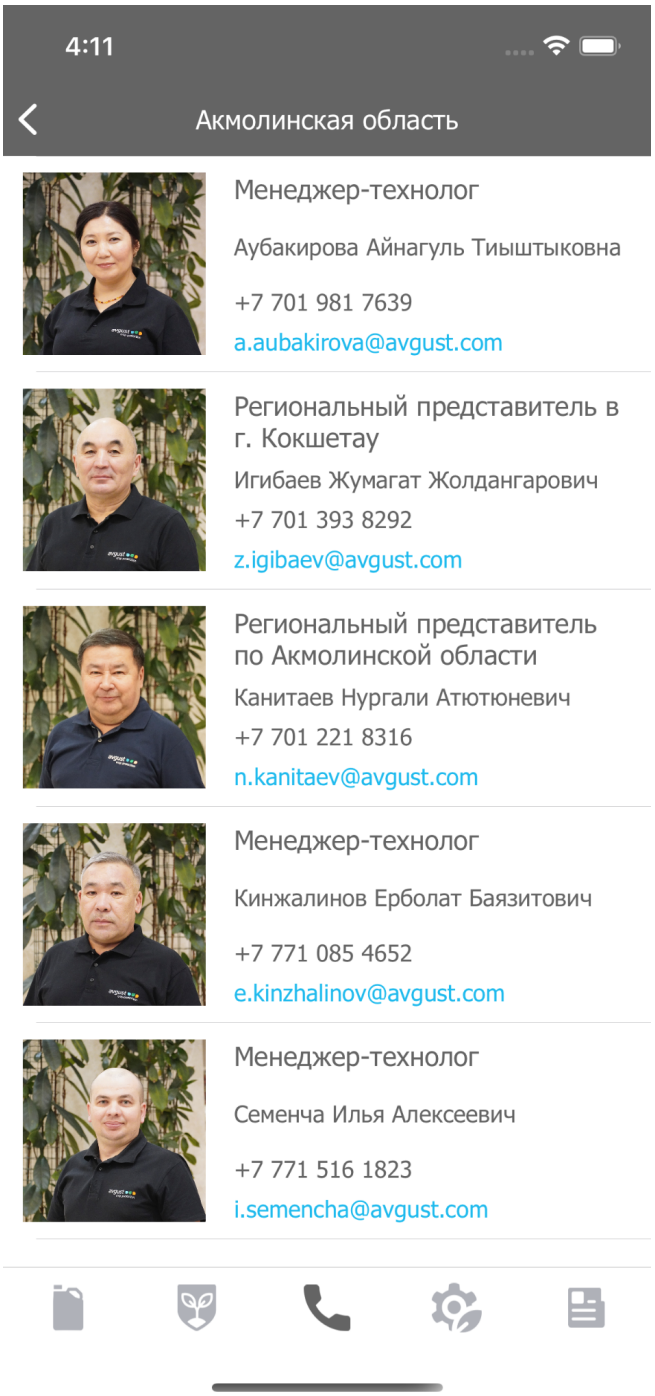



**Рисунок 16.** Список контактів

Дається коротка інформація про них: посада, ПІБ, номер телефону, фото та пошта. При натисканні на потрібну людину з'являється декілька опцій: подзвонити, написати на пошту чи додати в контакти(див. рис. 17).

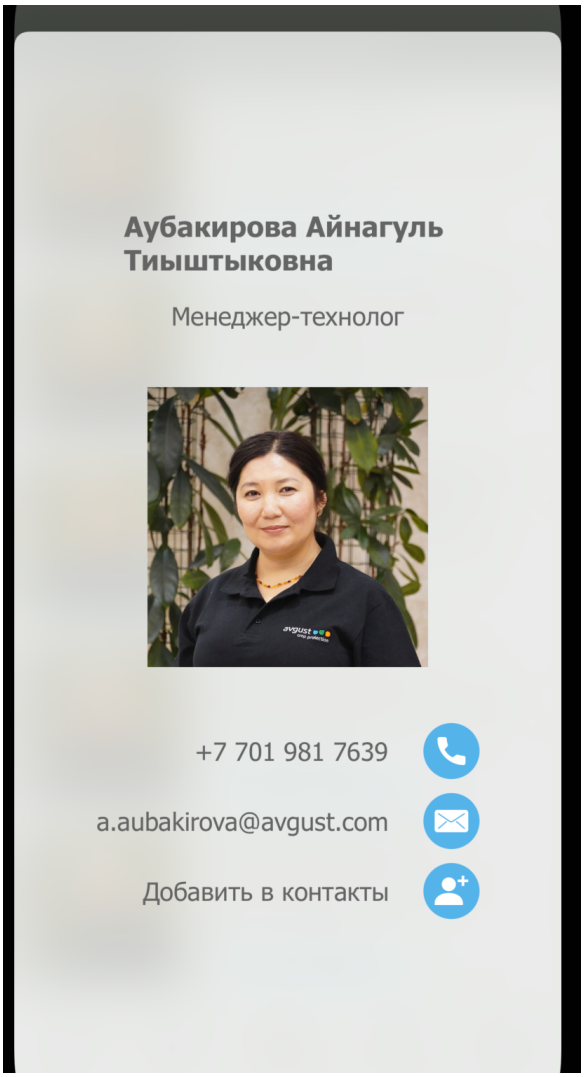



**Рисунок 17.** Сторінка представника

Якщо ж свайпнути вліво відкриється список дистриб'юторів з аналогічним функціоналом.

### 3.3 Розділ з погодою

Четверта вкладка містить в собі сервіси та інформацію. Тут знаходиться сторінка “Про компанію”, деякі необхідні знання для агронома при використанні продукції, а також сервіси, з яких реалізовано лише один “Погода”.

При виборі цього розділу відкривається велика мапа(див. рис. 18). Це спеціальний елемент інтерфейсу, який дозволяє використовувати

вбудовані в iPhone Apple Maps. На відміну від Google Maps, для їх використання не потрібно реєструватися, підключати сторонні API та окремо завантажувати необхідні матеріали. Даний елемент використовує вбудовані в телефон мапи і працює одразу, без налаштування. При вході буде запит на використання геолокації. Якщо не дозволити, користувач може натиснути на карту, з'явиться маркер і кнопка “Дізнатись погоду”. Після натискання відкриється вікно з погодними умовами в вибраній місцевості.

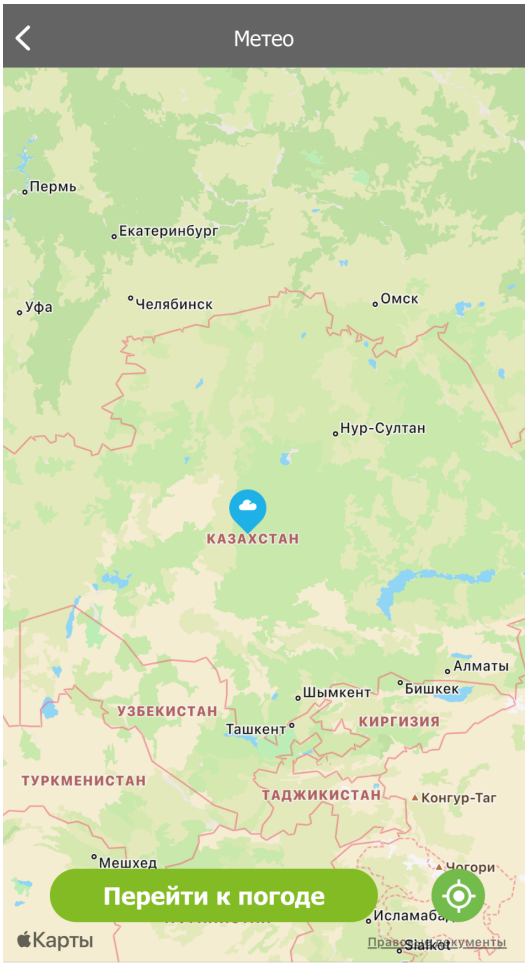



**Рисунок 18.** Вигляд мапи від Apple

Дані для цієї сторінки беруться за допомогою API OpenWeatherMap. Використовується безкоштовний пакет, який дає найновішу інформацію в точці. Для його використання треба зареєструвати додаток та отримати ключ. Він використовується при запиті. Також в запиті необхідно вказати

широту та довготу. У відповідь програма отримує json з докладною інформацією про погоду на вказаній точці. Дані конвертуються в зручний формат, відкидаються не потрібні, та таблично показуються користувачу.

### 3.4 Розділ з новинами та статтями

І остання вкладка це новини та статті. Аналогічно до контактів, в верхній частині є перемикач між ними. Знову в табличному вигляді представляються заголовки та зображення, при натисканні відкривається уся стаття чи новина. На даному етапі дані беруться з бази, але вже реалізований функціонал для завантаження їх з серверу. Відображення відбувається дуже простим способом. В текстове поле передається текст з веб-форматуванням, тобто з тегами. Для цього поля зроблена надбудова, яка і дозволяє приводити вебтекст до тексту з необхідним форматуванням.

# РОЗДІЛ 4. МАГАЗИН ДОДАТКІВ APP STORE

Останній етап розробки це поширення готового додатку у магазин. Для цього у кожної програми є унікальний bundle identifier, у додатку він повинен співпадати зі вказаним на відповідній сторінці в магазині. Тоді з'являється можливість викласти готовий build на сервера Apple. На сторінці програми необхідно заповнити багато полів з інформацією. Опис, назва, фото додатку, контактна інформація розробника, теги за якими можна знайти додаток, посилання на сайт компанії та права на використання. При завершенні перевірки завантаженого додатку, автоматично з'явиться іконка з нього. Тоді можна починати тестування.

## 4.1 Особливості тестування: TestFlight

Через закритість ОС тестування відбувається складніше, ніж у системи Android. Тут у нагоду приходить TestFlight[13], платформа розроблена для тестування додатків на системі iOS.

TestFlight – офіційний продукт компанії Apple для відкритого бета-тестування, призначений для iOS-пристроїв, щоб полегшити процес збору кодів тестових пристроїв. Даний сервіс дозволяє легко запросити користувачів протестувати додатки для iOS, watchOS і tvOS, перш ніж вони будуть випущені в App Store.

Для тестування потрібно завантажити додаток з аналогічною назвою, та мати доступ до програми.

Тестування поділяється на два рівні: зовнішнє та внутрішнє. Внутрішнє проводиться всередині компанії, до нього можно додати будь-якого члена організації, який має право тестувати продукти. При викладенні нової версії він автоматично отримує сповіщення, та можливість завантажити її.

Зовнішнє тестування дає можливість запросити до тестування будь-кого. Для цього необхідно запросити необхідну людину та заповнити

форму зворотнього зв'язку, щоб тестувальники могли зв'язатися з розробником та передати результати. На відміну від внутрішнього тестування, зовнішнє не дає нову версію, як тільки вона з'являється. Розробник повинен сам вказати версію, яку потрібно тестувати, або не вказувати, якщо тестування не потрібно.

### 4.2 Викладення готового продукту на платформу

Після проходження всіх кроків, заповнення всіх даних та вибору потрібної версії, нарешті можна викласти готовий додаток до магазину для всіх користувачів. Після підтвердження додаток переходить у ще одну, більш детальну перевірку. Якщо все добре, то додаток з'являється у магазині і будь-хто може його завантажити. Якщо ж з'являються проблеми, на пошту приходить лист з відмовою та описом, що саме не відповідає стандартам компанії. Це може бути як дрібничкою, наприклад недостатньо детальний опис навіщо додаток використовує геолокацію, так і серйозною несправністю.

### 4.3 Статистика та прибуток

Як і інші подібні платформи App Store предоставляє можливість переглянути успішність додатку. У акаунті, якщо звісно є доступ, можна переглянути досить докладну статистику, кількість завантажень за період, кількість сесій, кількість помилок та прибутку(див. рис. 19).

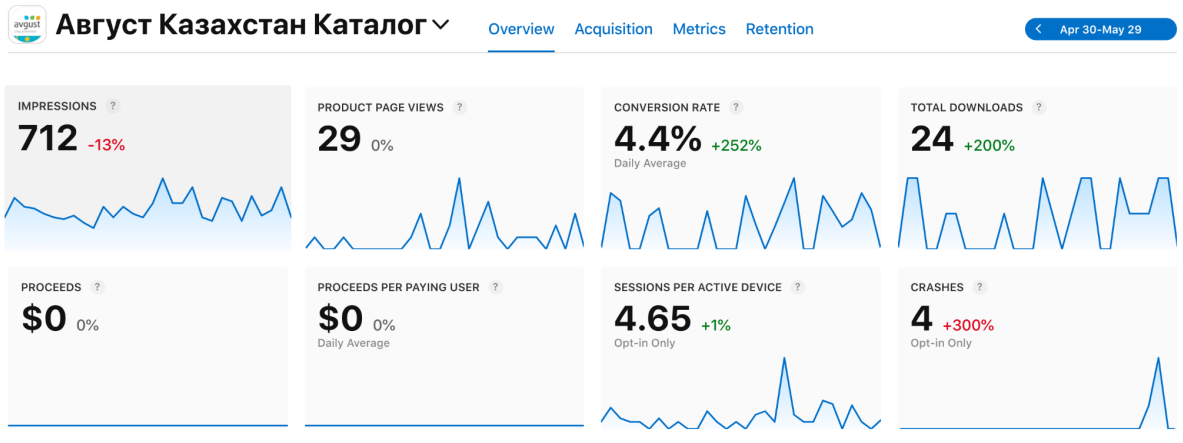



**Рисунок 19.** Статистика розробленого додатку

# ВИСНОВКИ

Агрокультурна галузь завжди була і буде вкрай необхідною для світової економіки та людей. Тому важливо проводити дослідження для селекції та захисту культур, особливо на фоні поточного кризису зерна, а також просто і зрозуміло доносити цю інформацію до фермерів. Для цього буде в нагоді розроблений додаток.

Довідникові системи завжди будуть необхідні та популярні. Через це спеціалізовані програми будуть розвиватись та конкурувати за аудиторію, змагаючись в простоті та зручності для користувача.

Даний проект також є важливим досвідом та підґрунтям для подальшого розвитку, бо це перший повністю самостійний проект на даному наборі технологій. Це знадобиться як для поточної роботи, так і для подальшого кар'єрного росту в галузі.

За час розробки детально оглядено можливості мови програмування Swift та інших інструментів для розробки саме під систему iOS. Також оглядено необхідні кроки для поширення додатку в магазин App Store, для внутрішнього і зовнішнього тестування та, як результат, розповсюдження на широку аудиторію.

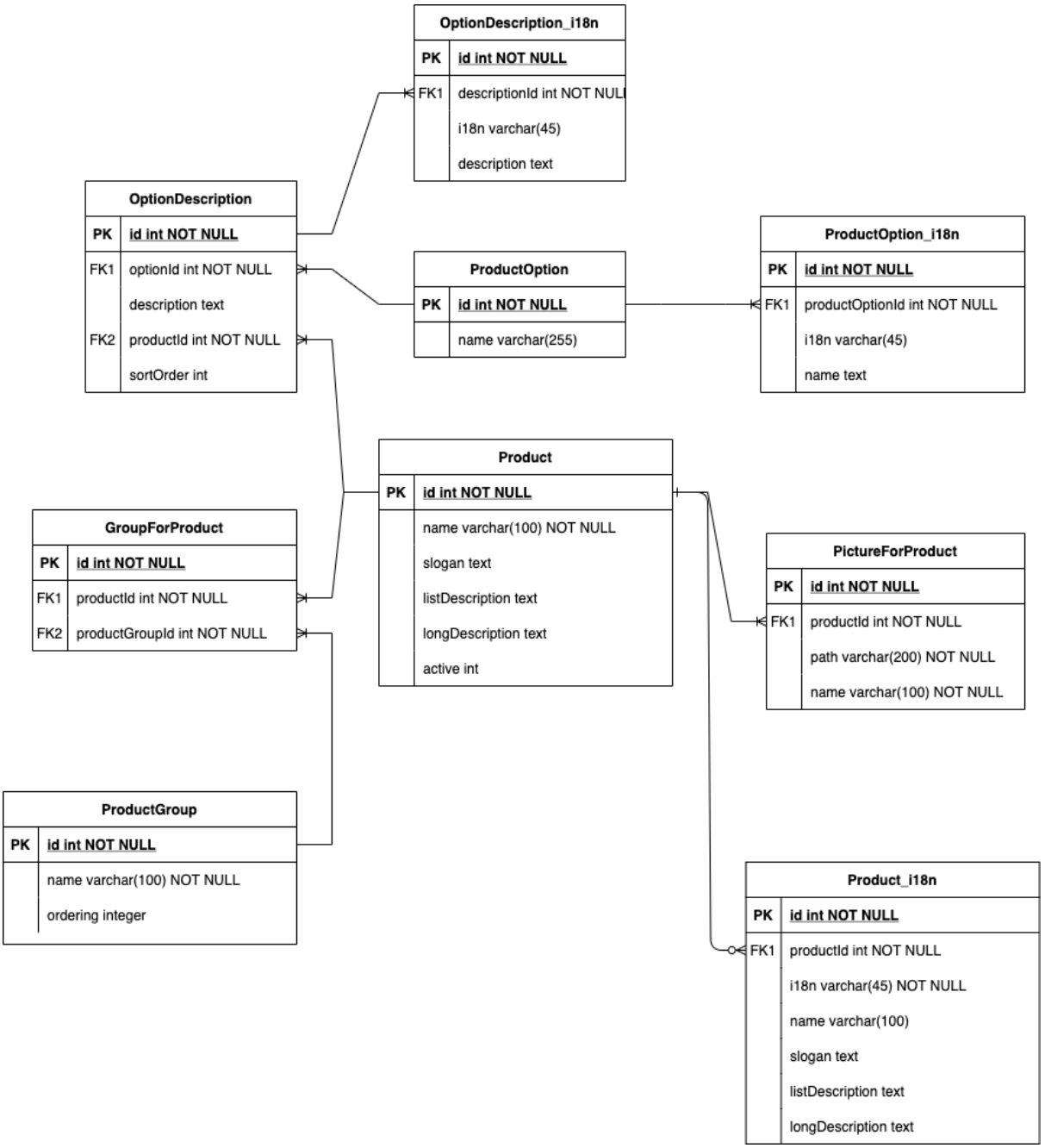
## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ


1. Apple Україна [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://www.apple.com/ua/
2. Swift - Apple [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://www.apple.com/ua/swift/
3. Xcode 14 Overview [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://developer.apple.com/xcode/
4. A Bit About Interface Builder [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://medium.com/swlh/a-bit-about-interface-builder-ceffaf484580
5. Zeplin [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://zeplin.io
6. Wallace W. Beginning iPhone Development with Swift 5: Exploring the iOS SDK / Wang Wallace. – San Diego, CA, USA: Apress, 2019. – 647 с.
7. Платформа, расширяющая возможности - Android [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://www.android.com/intl/ua_ua/
8. Що таке файл APK та як його відкрити [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://uk.myservername.com/what-is-an-apk-file
9. iPhone – Apple (UA) [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://www.apple.com/ua/iphone/
10. Developer Program [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://developer.apple.com/programs/
11. Swift – Apple (UA) - нова мова програмування від Apple [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://smychnyk.name/swift-apple-ua.html

12. CocoaPods [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://cocoapods.org
13. Бета-тестування за допомогою TestFlight - QATestLab [Електронний ресурс]: [Вебсайт]. - Режим доступу: https://training.qatestlab.com/blog/technical-articles/beta-testing-via-testflight/

# ДОДАТОК А

Частина схеми реляційної бази даних на прикладі продукту. Тут розглянута лише частина з усіх зв'язків сутності продукт. Опис продукту, його група, посилання на зображення на сервері, а також деяка детальна інформація про нього.



# ДОДАТОК Б

Статистика застосунку взята з аналітики App Store Connect