Компетентності рівня junior для iOS Developer	**Загальна оцінка компетентності (max. 6)**	**GlobalLogic**	**OLEARIS**
1. Основи iOS розробки. Swift та Objective-C.			
1.1 Розуміння архітектури та компонентів для розробки додатків на iOS Знає основні архітектурні патерни, такі як MVC, MVVM, VIPER, та особливості їх використання в iOS-додатках. Розуміє структуру та компоненти iOS-додатків, включаючи UIView, UIViewController, UINavigationController, UITableView, UICollectionView, і їх ролі у створенні користувацьких інтерфейсів. Вміє працювати з основними UI-компонентами, налаштовувати їх для взаємодії з користувачем та забезпечувати коректне відображення даних. Розуміє принципи життєвого циклу додатка та його основних компонентів, включаючи методи для управління пам'яттю та обробки подій. Володіє навичками використання Auto Layout для створення адаптивних інтерфейсів, які коректно працюють на різних пристроях і орієнтаціях екрану.	**4**	2	2
1.2 Змінні та типи даних. Структури даних.			
1.2.1 Змінні та типи даних. Структури даних у Swift Знає основні типи даних у Swift, включаючи Int, Double, String, Bool, Array, Dictionary та Set. Вміє оголошувати змінні та константи, використовуючи ключові слова var і let. Розуміє концепцію типобезпечності та може працювати з опціональними типами для управління відсутніми значеннями. Вміє створювати та маніпулювати масивами, словниками та множинами, розуміє їхні основні властивості та методи. Розуміє різницю між структурою та класом у Swift, а також вміє використовувати структури для організації та зберігання даних у програмах.	**6**	3	3
1.2.2 Змінні та типи даних. Структури даних у Objective-C Знає основи оголошення та використання змінних у Objective-C, включаючи різні типи даних, такі як int, float, double, NSString, NSArray, та NSDictionary. Розуміє відмінності між об'єктними та примітивними типами даних, а також основи роботи з пам'яттю при використанні об'єктів. Вміє використовувати основні структури даних для зберігання та маніпуляції інформацією, включаючи масиви, словники, множини, і знає, як ефективно управляти їх життєвим циклом та продуктивністю у додатках.	**3**	1	2
1.3 Оператори та цикли, інструкції мови, умовні конструкції			
1.3.1 Оператори та цикли, інструкції мови, умовні конструкції у Swift Знає основи синтаксису Swift, включаючи оператори, цикли та умовні конструкції. Вміє використовувати різні типи операторів (арифметичні, логічні, порівняння), управляти потоком виконання коду за допомогою умовних інструкцій (if, switch) та створювати цикли (for, while, repeat-while) для повторення виконання коду. Розуміє, як правильно використовувати ці конструкції для досягнення ефективності та читабельності коду, а також як обробляти помилки та особливості Swift, такі як опціональні типи в умовних конструкціях.	**6**	3	3
1.3.2 Оператори та цикли, інструкції мови, умовні конструкції у Objective-C Знає основи операторів, циклів та умовних конструкцій в Objective-C. Вміє використовувати оператори для виконання арифметичних, логічних та порівняльних операцій. Розуміє, як реалізовувати умови та управляти виконанням коду за допомогою конструкцій if, else, switch, а також реалізовувати повторювальні операції за допомогою циклів for, while та do-while.	**3**	1	2

1.4 Функції.			
1.4.1 Функції у Swift Знає основи створення та використання функцій у Swift. Вміє оголошувати функції, визначати їх параметри та типи повертаємих значень. Розуміє принципи передачі значень і посилань, а також використання замикань (closures) для створення функцій з параметрами, які є функціями або замиканнями. Вміє використовувати функції для структуризації коду, підвищення його повторного використання та читабельності, а також для реалізації специфічної логіки в додатках.	**6**	3	3
1.4.2 Функції у Objective-C Знає основні принципи створення та використання функцій у Objective-C. Вміє оголошувати та визначати функції, правильно передавати параметри та повертати значення. Розуміє синтаксис та основні конструкції мови, такі як методи класів та екземплярів, а також функції з використанням категорій. Вміє ефективно організовувати код, розділяючи його на функціональні блоки для покращення читабельності та підтримки коду.	**3**	1	2
1.5 Помилки та виключення (Errors & Exceptions)			
1.5.1 Помилки та виключення (Errors & Exceptions) у Swift Знає основи обробки помилок у Swift, включаючи використання конструкцій do, try, catch для обробки виключень. Розуміє різницю між Error, NSError та custom error types. Вміє створювати власні типи помилок, використовувати throws для оголошення функцій, які можуть генерувати помилки, та забезпечувати коректну обробку помилок для підвищення стабільності і надійності коду. Розуміє, як обробляти помилки при виконанні асинхронних операцій та використовувати try? і try! для обробки необов'язкових помилок.	**4**	2	2
1.5.2 Помилки та виключення (Errors & Exceptions) у Objective-C Знає основні принципи обробки помилок та виключень у Objective-C. Розуміє різницю між помилками (NSError) та виключеннями (NSException), а також знає, як їх використовувати для забезпечення надійності та стабільності додатків. Вміє використовувати конструкції @try, @catch, @finally для обробки виключень, а також правильно налаштовувати та обробляти NSError об'єкти для виявлення та реагування на помилки. Вміє впроваджувати механізми обробки помилок у коді для покращення користувацького досвіду та забезпечення коректної роботи програми.	**3**	1	2
1.6 Фреймворк Flutter Знає основні можливості фреймворку Flutter для створення кросплатформених мобільних застосунків. Розуміє архітектуру Flutter, включаючи поняття виджетів (widgets), їх життєвий цикл та ієрархію. Вміє використовувати Flutter для створення інтерфейсів користувача, працювати з основними компонентами (наприклад, текстовими полями, кнопками, контейнерами). Знає, як налаштовувати навігацію між екранами та працювати з станами (state management). Вміє створювати та налаштовувати анімації, інтегрувати зовнішні бібліотеки та взаємодіяти з REST API для отримання та відправлення даних.	**0**	0	0
1.7 Фреймворк UIKit Знає основні компоненти фреймворку UIKit (UIView, UIViewController, UILabel, UIButton, UITableView, UICollectionView тощо) та їх функціонал для створення користувацьких інтерфейсів. Вміє використовувати UIKit для розробки та налаштування візуальних елементів, обробки подій користувача та анімацій. Розуміє принципи Auto Layout та вміє створювати адаптивні інтерфейси для різних розмірів екранів. Вміє працювати з інструментами Interface Builder та Storyboard для побудови інтерфейсу, налаштовувати переходи між екранами та використовувати Segues. Розуміє базові концепції навігації та управління станами у додатку за допомогою UINavigationController та UITabBarController.	**5**	2	3
1.8 Розуміння основних архітектурних патернів, таких як MVC, MVVM, VIPER та Clean Architecture Знає основи архітектурних патернів, таких як MVC, MVVM, VIPER та Clean Architecture. Розуміє, як ці патерни організовують код та розділяють відповідальності між різними компонентами додатку. Вміє використовувати ці патерни для побудови масштабованих і підтримуваних iOS-додатків. Має знання про основні переваги кожного патерну та може обрати найбільш підходящий для конкретного проєкту, забезпечуючи ефективну організацію коду та покращуючи якість програмного забезпечення.	**3**	2	1
2. Теоретичний блок. Розуміння ООП			

2.1 Основні концепції ООП. Знає принципи використання об'єктів для організації та структурування даних та функціональності у програмі. Ідентифікує розрізнення між класом (шаблоном) та його екземплярами (об'єктами). Розуміє основні концепції ООП (інкапсуляція, спадкування та поліморфізм).	**6**	3	3
2.2 Об'єкти			
2.2.1 Об'єкти у Swift Знає основи роботи з об'єктами в Swift. Вміє створювати класи та структури, оголошувати властивості та методи, а також ініціалізувати об'єкти. Розуміє принципи інкапсуляції та може використовувати їх для захисту даних. Вміє працювати з іменованими ініціалізаторами та функціями в класах, а також розуміє різницю між класами та структурами в контексті передачі даних та їх мутації.	**6**	3	3
2.2.2 Об'єкти у Objective-C Знає основні принципи об'єктно-орієнтованого програмування в Objective-C. Вміє створювати та використовувати об'єкти, визначати та викликати методи, а також працювати з властивостями класів. Розуміє концепцію управління пам'яттю за допомогою ARC (Automatic Reference Counting) та відрізняє синтаксис Objective-C від інших мов. Здатний використовувати інтерфейси та реалізації класів для організації коду, а також управляти життєвим циклом об'єктів для забезпечення їхньої ефективної роботи та запобігання витокам пам'яті.	**3**	1	2
2.3 Класи, конструктори, властивості, модифікатори доступу			
2.3.1 Класи, конструктори, властивості, модифікатори доступу у Swift Знає основи створення класів у Swift, включаючи оголошення класів, конструкторів (ініціалізаторів) і властивостей. Вміє використовувати різні модифікатори доступу для контролю видимості класів і їх членів (публічні, приватні, внутрішні, захищені). Розуміє, як створювати та використовувати конструктори для ініціалізації об'єктів, а також як визначати і використовувати властивості для зберігання даних у класах. Вміє правильно організовувати код для забезпечення зручності і повторного використання класів	**5**	3	2
2.3.2 Класи, конструктори, властивості, модифікатори доступу у Objective-C Знає основи роботи з класами в Objective-C, включаючи створення та використання класів, конструкторів (ініціалізаторів) та властивостей. Вміє визначати та реалізовувати властивості класу з відповідними атрибутами (getter та setter), розуміє різницю між типами властивостей (strong, weak, copy тощо). Розуміє призначення та використання модифікаторів доступу (public, private, protected) для контролю видимості і доступу до елементів класу, що допомагає в організації коду та забезпеченні інкапсуляції.	**3**	1	2
2.4 Наслідування			
2.4.1 Наслідування у Swift Знає основи концепції наслідування у Swift, включаючи можливість створення підкласів для розширення функціональності базових класів. Вміє використовувати ключове слово class для визначення класів і override для перевизначення методів і властивостей в підкласах. Здатний визначати і використовувати ієрархії класів для організації коду і реалізації повторно використовуваних компонентів.	**6**	3	3
2.4.2 Наслідування у Objective-C Знає основи наслідування в Objective-C, включаючи створення базових та похідних класів. Вміє визначати та використовувати базові класи та їхні підкласи, розуміє концепцію переозначення методів (method overriding) та як це впливає на поведінку класів. Розуміє принципи інкапсуляції, модифікацію властивостей у підкласах, а також використання ключового слова super для доступу до методів базового класу. Вміє застосовувати наслідування для організації коду та повторного використання функціональності у проектах iOS.	**3**	1	2
2.5 Поліморфізм			
2.5.1 Поліморфізм у Swift Знає основи поліморфізму в Swift, включаючи використання протоколів та класів для реалізації поліморфізму. Вміє створювати базові класи та підкласи, використовувати протоколи для визначення спільного інтерфейсу та реалізовувати методи для конкретних типів даних. Розуміє, як поліморфізм допомагає створювати гнучкі та розширювані системи, що дозволяє використовувати один і той же інтерфейс для різних типів об'єктів.	**6**	3	3

2.5.2 Поліморфізм у Objective-C Знає основи поліморфізму в Objective-C, включаючи концепцію перевизначення методів у підкласах і використання протоколів для досягнення гнучкості коду. Вміє створювати та використовувати базові класи і підкласи, переозначати методи та викликати їх через посилання на базові класи. Розуміє, як використовувати поліморфізм для покращення масштабованості та підтримуваності коду, забезпечуючи його адаптивність до змін.	**3**	1	2
3. Розширені знання у формі iOS Development			
3.1 Розуміння принципів розробки користувальницького інтерфейсу з використанням Auto Layout технології та інструментів візуального проєктування (Storyboard і XIBs) Знає основи технології Auto Layout для створення адаптивних і масштабованих інтерфейсів на iOS. Вміє налаштовувати обмеження (constraints) для елементів інтерфейсу, щоб забезпечити коректне відображення на різних розмірах екранів та орієнтаціях. Розуміє принципи використання Storyboard та XIBs для візуального проєктування інтерфейсів, вміє створювати та налаштовувати екрани та переходи між ними. Вміє використовувати ці інструменти для швидкої розробки інтерфейсу та інтеграції з кодом, забезпечуючи узгодженість дизайну та функціональності.	**5**	2	3
3.2 Основи роботи зі специфічними компонентами iOS: UITableView, UICollectionView, UINavigationController, та іншими ключовими елементами інтерфейсу Знає основи роботи з ключовими компонентами інтерфейсу в iOS, такими як UITableView, UICollectionView та UINavigationController. Вміє створювати та налаштовувати таблиці та колекції для відображення даних у додатках, а також реалізовувати навігацію між екранами за допомогою UINavigationController. Розуміє принципи роботи делегатів і джерел даних (data sources) для UITableView та UICollectionView, а також вміє налаштовувати відповідні методи для обробки користувацького вводу та відображення контенту.	**5**	2	3
3.3 Зберігання та керування даними за допомогою Core Data Знає основи використання Core Data для зберігання та керування даними в iOS-додатках. Розуміє концепцію об'єктно-реляційного мапінгу (ORM), основні елементи Core Data, такі як моделі даних, контексти, об'єкти та зберігання. Вміє створювати та налаштовувати моделі даних, працювати з запитами, зберігати та отримувати дані за допомогою Core Data. Здатний вирішувати базові завдання з обробки даних, включаючи збереження нових записів, оновлення та видалення існуючих, а також обробку помилок, що виникають при роботі з даними.	**4**	1	3
3.4 Робота з багатозадачністю та асинхронним виконанням коду завдяки технології Grand Central Dispatch (GCD). Управління потоками та забезпечення безпечного доступу до спільних ресурсів. Знає основи Grand Central Dispatch (GCD) та його роль у забезпеченні багатозадачності та асинхронного виконання коду в iOS. Вміє використовувати GCD для виконання завдань у фонових потоках, розподілу роботи між різними чергами та синхронізації результатів. Розуміє принципи роботи з чергами (queues), як асинхронно обробляти завдання, і як уникати блокування основного потоку додатка. Розуміє, як створювати та керувати паралельними задачами для покращення продуктивності додатка. Вміє використовувати механізми синхронізації, такі як серійні черги та блокування, для забезпечення безпечного доступу до спільних ресурсів і уникнення умов гонки (race condition). Розуміє важливість синхронізації даних між потоками для запобігання помилок та підвищення стабільності додатка.	**5**	2	3
3.5 Методи аутентифікації та авторизації користувачів з використанням механізмів OAuth та JWT. Збереження конфіденційної інформації за допомогою сховища Keychain. Знає основи аутентифікації та авторизації користувачів за допомогою OAuth та JWT. Вміє налаштовувати процеси авторизації, інтегруючи їх у мобільні додатки для безпечного доступу до ресурсів. Розуміє принципи роботи токенів OAuth та JWT, а також їх застосування для управління сесіями користувачів. Вміє використовувати Keychain для збереження конфіденційної інформації, забезпечуючи високий рівень безпеки даних, таких як паролі та токени, на iOS-пристроях.	**3**	1	2
3.6 Використання Core Animation для створення анімацій та покращення візуального досвіду користувача Знає основи Core Animation у iOS, включаючи ключові компоненти та класи, такі як CALayer, CAAnimation, і CAKeyframeAnimation. Вміє створювати базові анімації для елементів інтерфейсу, такі як плавні переходи, ефекти зникнення та переміщення, а також налаштовувати властивості анімацій, такі як тривалість, затримка та типи кривих анімації. Розуміє, як використовувати Core Animation для покращення візуального досвіду користувача, оптимізувати продуктивність анімацій та забезпечити їх плавність у додатках.	**3**	1	2

3.7 Налаштування та обробка пуш-сповіщень, включаючи взаємодію з Apple Push Notification Service (APNs) Знає, як налаштувати пуш-сповіщення в iOS-додатках, включаючи конфігурацію Apple Push Notification Service (APNs). Розуміє процес реєстрації пристрою для отримання сповіщень, створення і реєстрації апаратного токена та отримання дозволу від користувача. Вміє обробляти отримані сповіщення у додатку, включаючи обробку фонових та активних сповіщень, а також налаштування відповідної логіки для відображення або взаємодії з ними.	**4**	1	3
3.8 Навички роботи з платформою App Store Connect для завантаження і керування додатками в App Store. Налаштування безпеки та підпису додатка через Code Signing Знає основи роботи з App Store Connect, включаючи процес завантаження додатків, управління версіями та перегляд статистики. Розуміє, як налаштувати підписування додатків (Code Signing) для забезпечення їхньої безпеки та відповідності вимогам Apple. Вміє створювати і керувати сертифікатами, профілями розробника та релізу, а також вирішувати проблеми, пов'язані з процесом підписання та публікації додатків в App Store.	**3**	1	2
3.9 Знання специфікацій пристрою та розуміння апаратних можливостей і особливостей пристроїв iOS. Знає основні специфікації та апаратні можливості iOS пристроїв, включаючи різні моделі iPhone та iPad, їх екранні роздільності, процесори, обсяг оперативної пам'яті, камери та інші апаратні компоненти. Розуміє, як ці характеристики впливають на продуктивність і функціональність додатків. Вміє адаптувати додатки під різні пристрої та екранні роздільності, а також використовувати специфічні апаратні можливості для оптимізації користувацького досвіду.	**4**	2	2
3.10 Розуміння оптимізації швидкодії додатків, ефективне використання ресурсів пристрою з використанням інструментів Time Profiler, Leaks та Memory Graph в Xcode та GCD Знає основні інструменти для оптимізації швидкодії додатків на iOS, такі як Time Profiler, Leaks та Memory Graph в Xcode. Вміє використовувати ці інструменти для аналізу продуктивності додатка, виявлення витоків пам’яті та оптимізації використання ресурсів. Розуміє принципи роботи Grand Central Dispatch (GCD) для ефективного управління паралельними завданнями та асинхронним виконанням коду, що допомагає зменшити затримки та покращити загальну швидкодію додатка. Вміє створювати ефективні асинхронні функції для покращення відгуку та продуктивності додатків.	**2**	1	1
4. Знання Web-технологій			
4.1 Загальні питання веб технологій Розуміє клієнт-серверну архітектуру. Знає основні мережеві протоколи, розуміє принцип роботи протоколу HTTP, його основні методи, статусні коди та їх значення. Вміє працювати з заголовками HTTP, формувати HTTP-запити, вказуючи метод, шлях, заголовки та дані. Може обробляти HTTP-відповіді, читати статусні коди, заголовки та вміст відповіді. Розуміє принципи взаємодії із зовнішніми API з використанням HTTP-запитів. Розуміє, як здійснювати взаємодію з RESTful API через HTTP-запити та обробляти дані у форматі JSON для обміну інформацією з серверами.	**3**	1	2
4.2 Налаштування роботи з мережевими запитами та обробки відповідей API Знає основи роботи з мережевими запитами в iOS, включаючи використання URLSession для відправки запитів і отримання відповідей від серверів. Розуміє основні HTTP методи (GET, POST, PUT, DELETE) і статусні коди відповідей. Вміє обробляти JSON або XML дані, отримані від API, та конвертувати їх у моделі даних Swift. Володіє навичками обробки помилок мережевих запитів та забезпечення плавного користувацького досвіду при взаємодії з API.	**4**	2	2
4.3 Взаємодія з віддаленими серверами завдяки URLSession або бібліотеці Alamofire. Знає основи роботи з URLSession та Alamofire для здійснення HTTP-запитів до віддалених серверів. Вміє налаштовувати і виконувати запити GET, POST, PUT, DELETE, обробляти відповіді сервера та управляти помилками. Розуміє, як використовувати URLSession для стандартних запитів і як бібліотека Alamofire спрощує цей процес за рахунок більш високого рівня абстракції та додаткових можливостей. Вміє інтегрувати отримані дані у додаток та обробляти їх відповідно до вимог проєкту.	**4**	2	2
5. Testing			

5.1 Основи тестування Знає основні принципи тестування, його типи (Unit testing, Integration testing, End to End testing) та важливість для розробки програмного забезпечення, здатний виконувати кожен з перелічених типів. Ідентифікує відмінності різних типів тестування в рівні складності та обсязі тестових сценаріїв. Вміння писати невеликі, швидкі та автоматичні тестові сценарії Unit testing для перевірки правильності роботи окремих модулів або методів. Вміє використовувати Xcode для написання і виконання юніт-тестів (Unit Testing) та тестів користувацького інтерфейсу (UI Testing). Розуміє процес інтеграційного тестування та використання фреймворків, які допомагають тестувати взаємодію між різними компонентами програмного забезпечення. Вміє проводити тестування програмного забезпечення в реальному середовищі з урахуванням всіх компонентів системи. Практикує автоматизацію процесу тестування для забезпечення ефективності і повторюваності тестових сценаріїв. Здатний аналізувати результати тестування, виявляти помилки та робити відповідні корективи, співпрацювати з розробниками для забезпечення якості та правильності коду.	**2**	1	1
6. Інструментарій розробника			
6.1 Системи контролю версій - GIT Розуміє git flow - методологію роботи з гілками для ефективного управління процесом розробки. Здатний створювати репозиторії та ініціалізувати проєкти з використанням Git. Розуміє та використовує команди Git (commit, push, pull, fetch, checkout, add, status, revert та інші). Вміє працювати з гілками (branches) для розробки функціональності відокремлено від основної гілки (наприклад, розробка нових функцій у власних гілках та їх об'єднання в основну гілку). Знайомий з процедурою створення пулл-реквестів для обговорення та злиття змін з різних гілок	**6**	3	3
6.2 Навички використання IDE. IDE Xcode. Вміє працювати з популярними IDE, такими як Xcode. Використовує редактор коду, налаштування розширень та плагінів для поліпшення продуктивності розробки. Застосовує вбудовані інструменти IDE для налагодження коду, аналізу помилок, рефакторингу та контролю якості коду. Знає основні функціональні можливості середовища розробки Xcode, включаючи створення та управління проєктами, написання та налагодження коду, а також використання інструментів для тестування та профілювання. Вміє працювати з інтерфейсом користувача Xcode, розуміє, як використовувати редактор коду, менеджер проєктів, Interface Builder для розробки інтерфейсів користувача, та інші інструменти для підвищення продуктивності розробки. Має базові навички налаштування та оптимізації середовища розробки для ефективної роботи над iOS-додатками.	**5**	3	2
7. Розмовна англійська			
7.1. Правильна побудова фрази з узгодженням часу Вміє використовувати відповідні часові форми дієслів і структури речень для точного вираження часу в комунікації. Здатний правильно використовувати часові форми, такі як Present Simple, Present Continuous, Past Simple, Past Continuous, Future Simple, для передачі правильного значення в повідомленні.	**4**	2	2
7.2 Професійна it термінологія в розмові з замовником Знає і використовує професійні IT-терміни та скорочення, які використовуються в IT-індустрії, для чіткого та зрозумілого спілкування зі замовником. Вміє перекладати складні технічні концепції на доступну мову для замовника, не знайомого з IT-галуззю.	**5**	3	2
7.3 Професійна it термінологія у діловому звіті Знає і використовує спеціалізовану IT-термінологію та фразеологію при написанні ділових звітів. Вміє передати інформацію про технічні питання та проєкти зрозуміло та конкретно, використовуючи адекватні IT-терміни та абревіатури.	**4**	2	2
7.4 Професійна термінологія для iOS Development Знає та розуміє специфічну термінологію, що використовується в контексті iOS Development Розуміє та використовує специфічні терміни, патерни, алгоритми, фреймворки, бібліотеки та інші інструменти, які використовуються в iOS Development	**4**	2	2

7.5 Неформальне спілкування (small talks) Вміє вести неформальну розмову, розпочинати діалоги та підтримувати невимушену атмосферу. Знає загальновживану термінологію та фрази, які використовуються у неформальних розмовах, таких як загальні питання про погоду, цікаві події, особисті захоплення, спорт, фільми тощо.	**2**	0	2

Коментарі до таблиці

Оцінки компетентностей вказані станом на 2024/2025 рр., по шкалі від 0 до 3.

«3» - компетентність вкрай важлива, якщо кандидат нею не володіє, його подальше просування на вакансію

«2» - компетентність, знання якої обов'язково знадобиться кандидату при подальшому професійному зростанні.

«1» - некритична компетентність, якою можна знехтувати.

«0» - неактуальна компетентність.

Зауваження від компаній

GlobalLogic

Знання Objective-C є перевагою, але не обов’язковою вимогою для Junior iOS Developer, якщо це не передбачено специфікою проєкту.

Знання Flutter також може бути перевагою, проте не є обов’язковим для iOS-розробника та залежить від потреб конкретного проєкту.

3.1. Розуміння принципів розробки користувацького інтерфейсу з використанням Auto Layout, Storyboard та XIB. Також бажане розуміння побудови інтерфейсів за допомогою SwiftUI.

3.2. Основи роботи з UITableView, UICollectionView, UINavigationController та іншими компонентами інтерфейсу. Додатково рекомендується розуміння відповідних підходів у SwiftUI, а також базових концепцій ObservableObject, ObservedObject і Combine.

3.3. Зберігання та керування даними за допомогою Core Data — достатньо загального розуміння, оскільки це часто залежить від специфіки проєкту.

3.5. Методи автентифікації та авторизації користувачів із використанням OAuth та JWT — достатньо загального розуміння принципів роботи.

3.6. Використання Core Animation для створення анімацій та покращення користувацького досвіду — бажано мати загальне уявлення про можливості технології.

3.7. Налаштування та обробка push-сповіщень, включаючи взаємодію з Apple Push Notification Service (APNs) — достатньо загального розуміння принципів роботи.

OLEARIS

1.1. Для рівня Junior достатньо розуміння архітектурного патерну MVC. Знання інших архітектурних підходів (MVVM, VIPER тощо) буде перевагою.

1.4. Знання Objective-C залежить від наявності проєктів, що використовують цю мову. Для Junior iOS Developer не є обов’язковою вимогою.

1.5. Розуміння принципів обробки помилок є важливим, однак вміння працювати з кастомними помилками не є обов’язковим для цього рівня.

4.2. Для початкового рівня достатньо розуміння роботи з JSON та базових принципів взаємодії з API.

4.3. Перевагою є знання нативного інструменту URLSession. Досвід роботи зі сторонніми бібліотеками, зокрема Alamofire, буде додатковим плюсом.

5.1. Основи тестування не є обов’язковою компетенцією для Junior-фахівця, однак їх наявність є суттєвою перевагою.

6.2. Для цього рівня достатньо впевненого користування Xcode. Знання розширень і плагінів не є пріоритетною вимогою.