Міністерство освіти і науки України

Криворізький національний університет

Факультет інформаційних технологій

Кафедра комп’ютерних систем та мереж

Пояснювальна записка

до кваліфікаційної роботи бакалавра

за спеціальністю 123 «Комп’ютерна інженерія»

на тему: МОБІЛЬНИЙ ТА ВЕБ-ДОДАТОК З ВИКОРИСТАННЯМ СУЧАСНИХ ТЕХНОЛОГІЙ ФРОНТЕНД- І БЕКЕНД- РОЗРОБКИ

Проектував \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Б. І. Каменев

Керівник роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. І. Кузнєцов

Нормоконтроль \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. І. Кузнєцов

Завідувач кафедри \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. І. Купін

Кривий Ріг

2025

Криворізький національний університет

Факультет інформаційних технологій

Кафедра комп’ютерних систем та мереж

Ступінь вищої освіти бакалавр

Спеціальність 123 «Комп’ютерна інженерія»

**ЗАТВЕРДЖУЮ**

Завідувач кафедри, голова циклової комісії

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. І. Купін

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ року

**З А В Д А Н Н Я**

**НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище, ім’я, по батькові)

1. Тема роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

керівник роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_,

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від “\_\_”\_\_\_\_ 20\_\_року №\_\_

2. Строк подання студентом роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Вихідні дані до роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно

розробити)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових

креслень)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Консультанти розділів роботи

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посада  консультанта | Підпис, дата | |
| завдання  видав | завдання  прийняв |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

7. Дата видачі завдання\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| №  з/п | Назва етапів роботи | Строк виконання етапів роботи | Примітка |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Студент** \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

**Керівник роботи** \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

РЕФЕРАТ

Змн

Арк.

№\_Документа№ документа

Підпис

Дата

Аркуш

КНУ.РБ.123.25.14.Р

КНУ.ПК.123.22.01.Р

Розробив

Каменев

Перевірив

Кузнєцов

Н.контроль

Кузнєцов

Затвердив

Купін

РЕФЕРАТ

Літера

Аркушів

КІ-21

Пояснювальна записка: 23 сторінок, 1 рисунків, 2 таблиць, 0 додаток, 4 використаних джерел.

Об’єкт аналізу та створеня – програмне забезпечення для MacOS для зручної роботи у багатовіконному режимі з мовною розкладкою для кожного вікна.

Проект складається з трьох розділів.

Перший розділ присвячений аналізу технології та середовища для створення ПЗ.

У другому розділі розкриті питання вибору операційної системи та середовища розробки програмного забезпечення.

Третій розділ присвячений моделюванню, а також аналізу недоліків. Вкінці розділу подано рекомендації щодо покращення роботи

ПЗ, MACOS, APPLE, SWIFT, REPL, APPKIT, UI, COCOA, IDE.

Explanatory note: 23 pages, 1 figures, 2 table, 0 appendix, 4 sources used.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.EN

Object of analysis and creation – software for MacOS for convenient work in multi-window mode with language layout for each window.

The project consists of three parts.

The first section is devoted to the analysis of technology and the environment for creating software.

The second section discusses the issues of choosing an operating system and software development environment.

The third section is devoted to modeling, as well as analysis of shortcomings. At the end of the section, recommendations are given for improving the work

S, MACOS, APPLE, SWIFT, REPL, APPKIT, UI, COCOA, IDE.

ЗМІСТ

Змн

Арк.

№\_Документа№ документа

Підпис

Дата

Аркуш

КНУ.РБ.123.25.14.З

КНУ.ПК.123.22.01.Р

Розробив

Каменев

Перевірив

Кузнєцов

Н.контроль

Кузнєцов

Затвердив

Купін

ЗМІСТ

Літера

Аркушів

КІ-21

[ПЕРЕЛІК СКОРОЧЕНЬ 8](#_Toc200007856)

[ВСТУП 9](#_Toc200007857)

[1. Аналіз технології та середовища для створення ПЗ 10](#_Toc200007858)

[1.1 Розвиток та особливості операційних систем MacOS 10](#_Toc200007859)

[‍1.1.1 macOS 10](#_Toc200007860)

[1.1.2 iOS (Apple) 12](#_Toc200007861)

[1.1.3 Веб-технології як платформа дистрибуції 13](#_Toc200007862)

[1.2 Аналіз підходів до зміни налаштувань джерел вводу 14](#_Toc200007863)

[1.2.1 На macOS 14](#_Toc200007864)

[1.2.2 На iOS 16](#_Toc200007865)

[1.3 Аналіз існуючих аналогів та альтернатив 17](#_Toc200007866)

[1.3.1 Для macOS 17](#_Toc200007867)

[1.3.2 Для iOS 18](#_Toc200007868)

[Висновок за розділом «Аналіз технології та середовища для створення ПЗ» 19](#_Toc200007869)

[2. Дослідження та вибір необхідних компонентів для створення програмного забезпечення 20](#_Toc200007870)

[2.1 Середовища розробки та мови програмування 20](#_Toc200007871)

[2.1.1 Xcode (для macOS та iOS додатків) 20](#_Toc200007872)

[2.1.2 Swift (для macOS та iOS додатків) 21](#_Toc200007873)

[2.1.3 Visual Studio Code (для веб-сайту) 21](#_Toc200007874)

[2.2 Фреймворки та бібліотеки 22](#_Toc200007875)

[2.2.1 Для macOS додатку 22](#_Toc200007876)

[2.2.2 Для iOS додатку 23](#_Toc200007877)

[2.3 Системи управління залежностями та контролю версій 25](#_Toc200007878)

[2.3.1 Для веб-сайту 26](#_Toc200007879)

[2.4. Аналіз нативного та кросплатформенного підходу 27](#_Toc200007880)

[Висновок за розділом «Дослідження та вибір необхідних компонентів для створення програмного забезпечення» 28](#_Toc200007881)

[3.1.1. Архітектура додатку для macOS 31](#_Toc200007882)

[3.1.2. Опис алгоритму роботи перемикання розкладок на macOS 33](#_Toc200007883)

[3.1.3. Проектування інтерфейсу користувача macOS додатку 36](#_Toc200007884)

[3.2 Проектування та реалізація iOS додатку 39](#_Toc200007885)

[3.2.1 Архітектура iOS додатку (Кастомна Клавіатура) 39](#_Toc200007886)

[3.2.2 Опис алгоритму роботи Custom Keyboard Extension 41](#_Toc200007887)

[3.2.3 Проектування інтерфейсу користувача iOS додатку (Кастомна Клавіатура) 43](#_Toc200007888)

[3.3 Проектування та реалізація Веб-сайту 46](#_Toc200007889)

[3.3.1 Архітектура Веб-сайту 46](#_Toc200007890)

[3.3.2 Проектування інтерфейсу користувача Веб-сайту 47](#_Toc200007891)

[Висновок за розділом 3 49](#_Toc200007892)

[Висновок 51](#_Toc200007893)

[Список використаних джерел 53](#_Toc200007894)

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.З

ПЕРЕЛІК СКОРОЧЕНЬ

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.ПС

S – Software

REPL – Read-Eval-Print-Loop

UI – user interface

ПЗ – Програмне забезпечення

IDE – Integrated Development Environment

VCS – Version Control System

HIG – Human Interface Guidelines

**ВСТУП**

Змн

Арк.

№\_Документа№ документа

Підпис

Дата

Аркуш

КНУ.РБ.123.25.14.В

КНУ.ПК.123.22.01.Р

Розробив

Каменев

Перевірив

Кузнєцов

Н.контроль

Кузнєцов

Затвердив

Купін

ВСТУП

Літера

Аркушів

КІ-21

Актуальність теми Сучасний цифровий світ характеризується інтенсивним використанням комп'ютерних систем та мобільних пристроїв у різних сферах діяльності. Користувачі все частіше взаємодіють з численними програмами одночасно, працюючи у багатозадачному режимі та перемикаючись між різними додатками. У цьому контексті, ефективна взаємодія з операційною системою та пристроями введення є ключовим фактором продуктивності та зручності.

Однією з поширених проблем, з якою стикаються користувачі, особливо ті, хто працює з декількома мовами, є необхідність постійного ручного перемикання розкладки клавіатури при зміні активного вікна або додатку. Це призводить до значних незручностей, витрати часу на корекцію помилок, зниження концентрації уваги та загальної продуктивності. Хоча деякі операційні системи (наприклад, Microsoft Windows) надають вбудовану підтримку "розкладки на вікно", macOS та більшість мобільних платформ (зокрема, iOS) традиційно цього не забезпечують, встановлюючи єдину системну розкладку для всіх активних програм. Це створює нагальну потребу в програмних рішеннях, які б автоматизували цей процес, адаптуючись до контексту роботи користувача на різних платформах.

Більше того, у зв'язку з розвитком багатоплатформних рішень, де користувачі працюють як на десктопних, так і на мобільних пристроях, виникає потреба в уніфікованих та зручних інструментах, доступних на різних платформах, а також у сучасних методах їх дистрибуції. Розробка таких комплексних рішень, що включають десктопні, мобільні та вебкомпоненти, є актуальним завданням сучасної комп'ютерної інженерії, що вимагає глибокого розуміння як нативних технологій розробки, так і сучасних фронтенд- та бекенд-практик.

Метою кваліфікаційної роботи є розробка комплексного програмного забезпечення, що включає десктопний додаток для macOS, мобільний додаток для iOS, а також веб-сайт для їх дистрибуції, які автоматично запам'ятовують та перемикають мовну розкладку клавіатури для кожного активного вікна/додатку, використовуючи сучасні технології фронтенд- та бекенд-розробки.

**1. Аналіз технології та середовища для створення ПЗ**

Змн

Арк.

№\_Документа№ документа

Підпис

Дата

Аркуш

КНУ.РБ.123.25.14.01АТССПЗ

КНУ.ПК.123.22.01.Р

Розробив

Каменев

Перевірив

Кузнєцов

Н.контроль

Кузнєцов

Затвердив

Купін

Аналіз технології та середовища для створення ПЗ

Літера

Аркушів

КІ-21

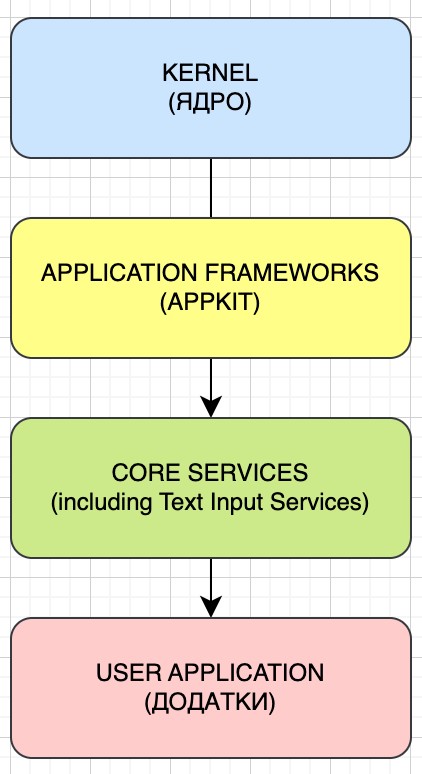
**1.1 Розвиток та особливості операційних систем MacOS**

Сучасний розвиток інформаційних технологій та глобалізація сприяють появі багатоплатформних програмних рішень, що функціонують у різних операційних середовищах. Для ефективної розробки такого програмного забезпечення необхідне глибоке розуміння архітектурних особливостей та можливостей кожної цільової платформи. У даному підрозділі буде розглянуто ключові характеристики операційних систем macOS та iOS, а також принципи функціонування веб-технологій як платформи для дистрибуції.

Операційна система від компанії Apple часто асоціюється користувачами з програмним забезпеченням для фахівців. На кожному кроці можна почути про те, що MacOS — це те, що потрібно кожному дизайнеру, програмісту та оптимізатору.

**‍1.1.1 macOS**

Операційна система macOS (раніше OS X), розроблена компанією Apple Inc., є ключовою платформою для настільних комп'ютерів та ноутбуків Apple. Вона відома своїм інтуїтивно зрозумілим інтерфейсом, стабільністю, високою продуктивністю та інтеграцією з іншими продуктами екосистеми Apple.



арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.01.АТССПЗ

Рисунок 1.1 – Спрощена архітектура macOS (з акцентом на взаємодію з введенням)

**Ключові особливості macOS:**

**Unix-подібне ядро:** macOS побудована на базі Darwin, Unixподібного ядра, що забезпечує високу стабільність, безпеку та потужні можливості для розробників, включаючи доступ до стандартних Unixінструментів.

**Інтерфейс Aqua:** Графічний інтерфейс користувача Aqua відрізняється естетичністю, плавними анімаціями та високою увагою до деталей, забезпечуючи приємний користувацький досвід.

**Інтеграція екосистеми**: Тісна інтеграція з іншими пристроями Apple (iPhone, iPad, Apple Watch) через iCloud, Continuity та Handoff, що створює єдине та зручне робоче середовище.

**Високі стандарти безпеки:** macOS реалізує численні механізми безпеки, такі як Gatekeeper, Sandbox, System Integrity Protection (SIP), що обмежують можливості програмного забезпечення до доступу до системних ресурсів. Це підвищує захист користувачів від шкідливих програм, але водночас вимагає явних дозволів (наприклад, Accessibility API) для додатків, які взаємодіють з низькорівневими системними функціями.

**Централізована розкладка клавіатури:** Однією з важливих особливостей macOS, що має безпосереднє відношення до даної роботи, є принцип єдиної

системної розкладки клавіатури. На відміну від деяких інших операційних систем (наприклад, Windows), macOS не надає вбудованої функціональності для автоматичного запам'ятовування та перемикання мовної розкладки для кожного окремого вікна або додатку. Зміна розкладки є системною, і вона застосовується до всіх активних програм одночасно. Ця архітектурна особливість обумовлює необхідність розробки спеціалізованого програмного забезпечення для емуляції бажаної поведінки.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.01.АТССПЗ

**1.1.2 iOS (Apple)**

Операційна система iOS є мобільною платформою, розробленою Apple Inc. для своїх пристроїв iPhone та iPad. Вона є найбільш поширеною мобільною ОС у світі після Android і відома своєю простотою використання, плавністю інтерфейсу та надійністю.

**Ключові особливості iOS:**

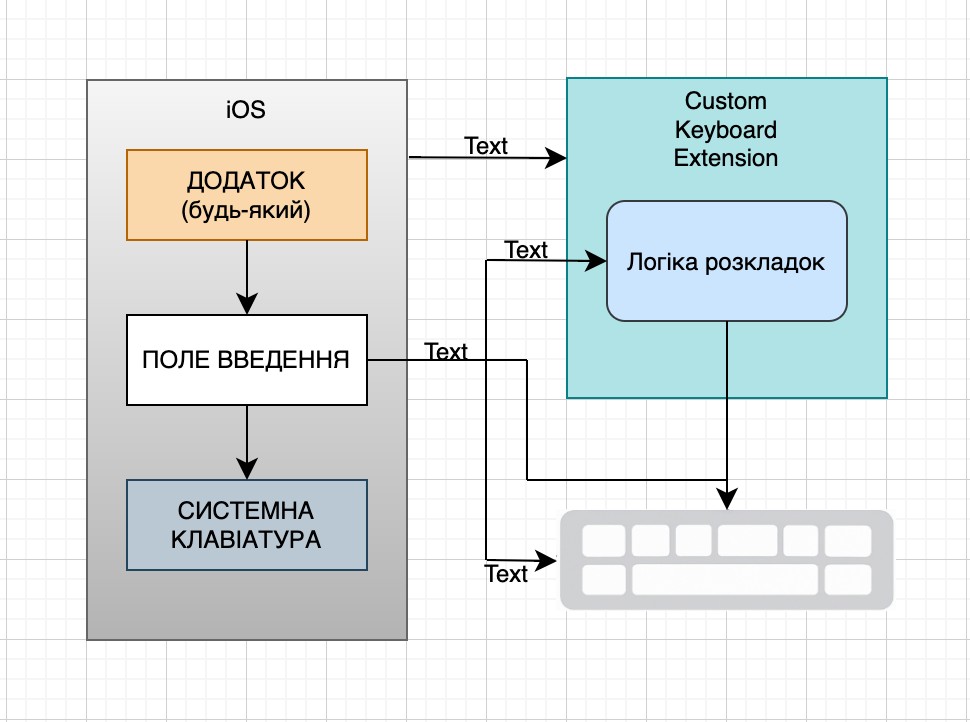
**Закрита та контрольована екосистема:** iOS характеризується високим рівнем контролю Apple над апаратним та програмним забезпеченням. Це забезпечує підвищену безпеку, конфіденційність даних користувачів та стабільну роботу додатків, але накладає певні обмеження на розробників.

**Оптимізована продуктивність:** Завдяки глибокій інтеграції програмного та апаратного забезпечення, iOS-пристрої забезпечують високу швидкість відгуку та плавність анімацій навіть на менш потужному "залізі" порівняно з конкурентами.

**Централізоване управління додатками:** Всі додатки на iOS встановлюються виключно через App Store, що гарантує їх перевірку на відповідність стандартам безпеки та функціональності.

**Суворий контроль над введенням:** Як і macOS, iOS реалізує суворий контроль над системним введенням. Стандартна клавіатура iOS є єдиною для всіх додатків. Можливість програмно керувати розкладками клавіатури для різних додатків або вікон є значно обмеженою або відсутньою у стандартних публічних API.

**Механізм Input Method Extension (IME) / Custom Keyboards:** Для реалізації функціоналу "розкладка на додаток" на iOS, стандартні API не підходять. Єдиним життєздатним підходом є розробка власної кастомної клавіатури (також відомої як Input Method Extension, IME). Ця клавіатура функціонує як окреме розширення, яке користувач може активувати замість стандартної системної клавіатури. Всередині такої кастомної клавіатури розробник отримує контроль над відображенням розкладки та її логікою, що дозволяє реалізувати запам'ятовування та перемикання розкладки для різних додатків. Це вимагає від користувача явного дозволу на використання сторонньої клавіатури та надання їй повного доступу.



арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.01.АТССПЗ

Рисунок 1.2 – Архітектура Custom Keyboard Extension в iOS

**1.1.3 Веб-технології як платформа дистрибуції**

Веб-технології відіграють ключову роль у сучасному цифровому світі, забезпечуючи універсальний доступ до інформації та програмного забезпечення через веб-браузери. У контексті даної дипломної роботи, вебсайт виступає як ефективна платформа для дистрибуції розроблених десктопного та мобільного додатків.

**Ключові аспекти веб-технологій для дистрибуції:**

**Універсальність доступу:** Веб-сайт доступний з будь-якого пристрою, що має веб-браузер та підключення до Інтернету, незалежно від операційної системи. Це робить його ідеальною платформою для широкого поширення програмного забезпечення.

**Гнучкість розгортання:** Веб-сайти не підлягають тим самим строгим правилам модерації, що й магазини додатків. Це дозволяє розробникам швидше публікувати оновлення та виправлення, а також пропонувати програмне забезпечення, яке, можливо, не відповідає всім вимогам App Store (наприклад, через використання низькорівневих системних API, що може бути інтерпретовано як порушення безпеки).

**Централізоване джерело інформації:** Веб-сайт слугує єдиним офіційним джерелом інформації про додатки, їх функції, системні вимоги та інструкції з встановлення.

**Простота завантаження:** Користувачі можуть легко перейти за посиланням та завантажити необхідні файли (наприклад, .dmg для macOS, .ipa для iOS або посилання на App Store, якщо додаток там буде опубліковано, але для кастомної клавіатури часто потрібна пряма дистрибуція або тестові збірки).

**Основні технології:** Для створення веб-сайту використовуються стандартні веб-технології:

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.01.АТССПЗ

* **HTML (HyperText Markup Language):** Визначає структуру та зміст веб-сторінок.
* **CSS (Cascading Style Sheets):** Використовується для візуального оформлення веб-сторінок, їх стилізації та адаптації під різні розміри екранів.
* **JavaScript:** Забезпечує інтерактивність на сторінці, дозволяє реалізовувати динамічні елементи, валідацію форм та інші клієнтські скрипти.

**Роль у проекті:** Веб-сайт виконуватиме функцію порталу для завантаження десктопного та мобільного додатків. Він може містити опис функціоналу, скріншоти, інструкції з використання та посилання для завантаження дистрибутивів. Завдяки цій платформі, користувачі матимуть зручний доступ до програмного забезпечення безпосередньо від розробника.**1.2 Недоліки MacOS**

Як і у будь-якого програмного забезпечення — у MacOS є свої недоліки.

**1.2 Аналіз підходів до зміни налаштувань джерел вводу**

У сучасних операційних системах управління джерелами введення (такими як розкладки клавіатури) є фундаментальним аспектом взаємодії користувача. Проте, архітектурні особливості різних платформ обумовлюють різні підходи до програмного контролю над цими налаштуваннями. У даному підрозділі буде розглянуто, як macOS та iOS дозволяють або обмежують маніпуляції з розкладками клавіатури, та які підходи є життєздатними для реалізації функціоналу автоматичного перемикання.

**1.2.1 На macOS**

**macOS** має специфічний підхід до управління розкладками клавіатури, який відрізняється від деяких інших операційних систем. За замовчуванням, macOS використовує єдину глобальну розкладку клавіатури, яка застосовується до всіх активних програм та вікон одночасно. Це означає, що при перемиканні між програмами, мовна розкладка залишається тією ж, якщо користувач її не змінить вручну.

Для реалізації функціоналу автоматичного перемикання розкладки для кожного вікна/додатку необхідно використовувати низькорівневі системні API, які дозволяють програмно:

* **Визначати активну програму (і вікно):** Це дозволяє ідентифікувати контекст, для якого потрібно запам'ятати або змінити розкладку.
* **Отримувати поточну розкладку клавіатури:** Необхідно знати, яка розкладка активна в момент перемикання програми, щоб її можна було зберегти.
* **Встановлювати бажану розкладку клавіатури:** Це дозволяє перемикати розкладку автоматично при активації відповідної програми.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.01.АТССПЗ

Основні API для реалізації на macOS:

* **Carbon Framework (Text Input Services - TIS):** Хоча фреймворк Carbon вважається застарілим і більшість його функцій замінені в новіших фреймворках (як Cocoa/AppKit), деякі його частини, зокрема Text Input Services (TIS), залишаються єдиним надійним способом програмного керування джерелами вводу на macOS.
* **TISCopyCurrentKeyboardInputSource()**: Ця функція дозволяє отримати об'єкт **TISInputSource**, який представляє поточну активну розкладку клавіатури. Це дає доступ до ідентифікатора розкладки

(наприклад “com.apple.keylayout.Ukrainian” або

“com.apple.keylayout.US”).

* **TISSelectInputSource(\_:)**: Ця функція дозволяє програмно встановити (перемкнути) певну розкладку клавіатури, передавши їй відповідний об'єкт **TISInputSource**.
* **Обґрунтування використання:** Використання TIS API є фундаментальним для даної роботи, оскільки воно забезпечує прямий програмний контроль над розкладками клавіатури, що є недоступним через більш сучасні та високорангові фреймворки.

**Accessibility API (через AppKit/Foundation):** Для відстеження активних програм та їхніх вікон, а також для отримання інформації про них (наприклад, назви програми), використовуються можливості Accessibility API. Додатки, які взаємодіють з Accessibility API, потребують відповідних дозволів від користувача (надаються в System Settings → Privacy & Security → Accessibility).

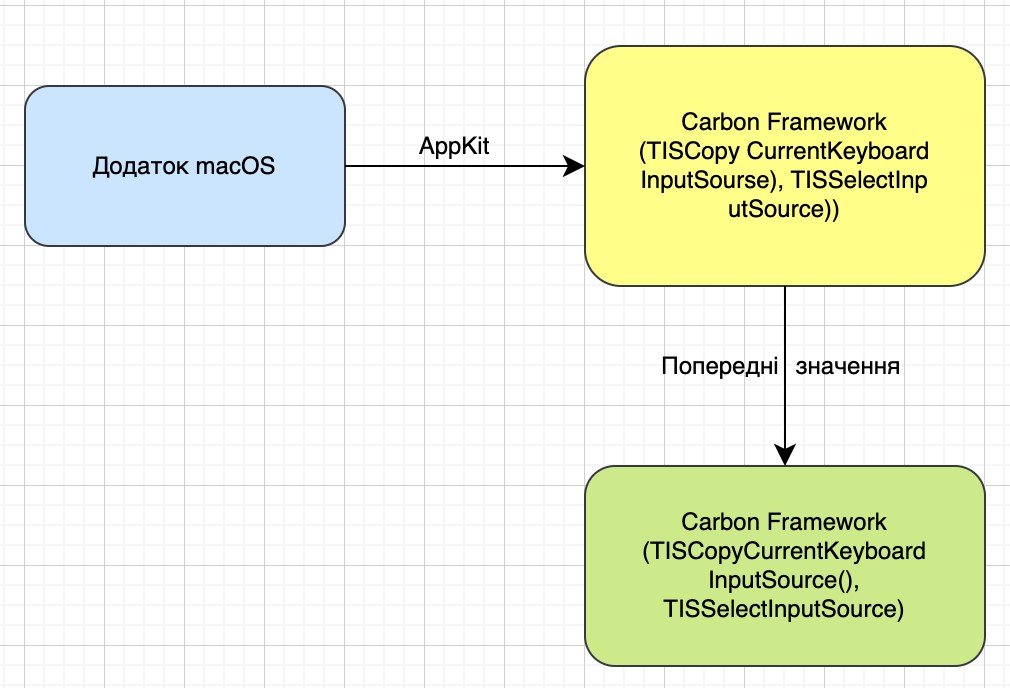


Рисунок 1.3 – Схема взаємодії macOS додатку з системними API для керування розкладкою

* NSWorkspace.shared.frontmostApplication: Це метод AppKit, який дозволяє отримати об'єкт NSRunningApplication, що представляє поточну активну (frontmost) програму. Він надає інформацію про її назву (localizedName) та інші властивості.
* AXUIElement (ієрархія): Для більш глибокого аналізу інтерфейсу активного вікна (наприклад, його заголовка), можна використовувати ієрархію AXUIElement з Accessibility API. Це дозволяє програмно досліджувати елементи користувацького інтерфейсу активного додатку.

**Обґрунтування використання**: Accessibility API є ключовим для моніторингу системного стану та ідентифікації активного контексту, що дозволяє застосувати логіку перемикання розкладок.

**1.2.2 На iOS**

На відміну від десктопних операційних систем, iOS має більш обмежений доступ до системних налаштувань клавіатури для сторонніх розробників, що обумовлено високими стандартами безпеки та конфіденційності користувачів. iOS не надає прямого API для програмного перемикання системної розкладки клавіатури для різних додатків або для відстеження розкладки, яка була використана в конкретному полі введення.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.01.АТССПЗ

**Рішення для проекту: Розробка власної кастомної клавіатури (Custom**

**Keyboard Extension)**

Єдиним життєздатним та офіційно підтримуваним підходом для реалізації функціоналу "розкладка на додаток" на iOS є створення власної кастомної клавіатури, яка діє як розширення системи (Input Method Extension).

**Принцип роботи Custom Keyboard Extension:**

* Користувач встановлює розроблену кастомну клавіатуру як додаток, а потім активує її в системних налаштуваннях клавіатур.
* Після активації користувач може перемикатися на цю клавіатуру замість стандартної iOS-клавіатури в будь-якому додатку, що підтримує введення тексту.
* Вся логіка відображення клавіш, обробки натискань та, що найважливіше для даного проекту, управління внутрішньою розкладкою клавіатури, реалізується в межах цього розширення.

**Реалізація логіки перемикання в Custom Keyboard:**

* Оскільки кастомна клавіатура контролює свій власний стан, вона може відстежувати, в якому додатку (або навіть типі поля введення) вона активна.
* При зміні активного додатку, клавіатура може перевіряти, чи є для цього додатку збережена розкладка (наприклад, у власних налаштуваннях або локальному сховищі).

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.01.АТССПЗ

* Якщо збережена розкладка існує, клавіатура автоматично може змінити своє внутрішнє відображення клавіш та символів на цю розкладку. Якщо ні, вона може використовувати розкладку за замовчуванням або останню використану.
* Цей підхід дозволяє емулювати поведінку "розкладка на додаток" в межах контрольованого середовища власної клавіатури.

**Вимоги та обмеження Custom Keyboards:**

* Вимагає від користувача явного дозволу на встановлення та використання сторонньої клавіатури, а також надання їй "повного доступу" (Full Access) для деяких функцій (наприклад, доступ до мережі або shared container для обміну даними, якщо це потрібно).
* Розробка кастомної клавіатури вимагає знання фреймворку UIKit (для побудови UI клавіатури) та розуміння архітектури розширень iOS.
* Обмеження за часом роботи та ресурсами, які надаються розширенням

**1.3 Аналіз існуючих аналогів та альтернатив**

Перед початком розробки будь-якого програмного забезпечення критично важливим є аналіз існуючих рішень та альтернатив, щоб зрозуміти поточний стан ринку, виявити незадоволені потреби користувачів та обґрунтувати унікальність та цінність власної розробки. Даний підрозділ присвячений огляду аналогів, що надають або намагаються надати функціонал, схожий на автоматичне перемикання розкладки клавіатури на macOS та iOS.

**1.3.1 Для macOS**

На платформі macOS, незважаючи на її популярність серед багатомовних користувачів, відсутні вбудовані системні механізми для автоматичного перемикання розкладки клавіатури для кожного окремого активного вікна або додатку. Це є суттєвою відмінністю від операційної системи Microsoft Windows, де така функціональність реалізована на системному рівні, і користувач може налаштувати мовну розкладку індивідуально для кожної програми, яку він використовує.

Існуючі сторонні рішення на macOS, хоча і пропонують певний функціонал, не повністю задовольняють потребу в автоматичному перемиканні розкладок:

Вбудована функціональність (відсутня): Як було зазначено вище, macOS не має нативної підтримки "розкладки на вікно". Користувач може вручну перемикати розкладки через системне меню або гарячі клавіші, але система не запам'ятовує, яка розкладка була активна в конкретній програмі при останньому її використанні. Це є основним обґрунтуванням для розробки даного десктопного

додатку.

Karabiner-Elements:

- Опис: Це потужна та гнучка утиліта для macOS, яка дозволяє перепризначувати клавіші на клавіатурі, створювати складні модифікації поведінки клавіш, комбінації та макроси. Вона оперує на низькому рівні, перехоплюючи події клавіатури.

- Обмеження для даного проекту: Основний фокус KarabinerElements полягає у зміні поведінки клавіш та створенні власних правил введення. Вона не призначена і не надає функціоналу для запам'ятовування або автоматичного перемикання мовної розкладки для окремих програм або вікон на основі їх активності. Хоча її можна використовувати для створення деяких складних правил, які опосередковано впливають на розкладку, це не є її прямою функцією і не забезпечує автоматичного перемикання за принципом "одна програма - одна розкладка" без складних налаштувань.

Ukelele:

- Опис: Ukelele є спеціалізованим інструментом для macOS, призначеним для створення, редагування та встановлення власних розкладок клавіатури у форматі .keylayout. Користувач може змінити призначення клавіш, додати нові символи або створити повністю унікальну розкладку.

- Обмеження для даного проекту: Ukelele є інструментом для створення розкладок, а не для управління їх перемиканням в динамічному режимі. Він не надає функціоналу для моніторингу активних програм та автоматичної зміни розкладки на основі контексту. Його роль – це статична конфігурація розкладки.

Обґрунтування потреби: На основі аналізу існуючих рішень можна зробити висновок, що на macOS відсутній інтегрований або широко доступний сторонній інструмент, який би повною мірою задовольняв потребу в автоматичному та інтелектуальному перемиканні мовної розкладки клавіатури для кожного активного вікна/додатку. Існуючі утиліти або мають інше призначення, або їх функціонал є обмеженим у цьому аспекті. Це підтверджує актуальність та практичне значення розробки десктопного додатку, що вирішує цю проблему.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.01.АТССПЗ

**1.3.2 Для iOS**

На мобільній платформі iOS, управління клавіатурами також має свої особливості. Системна клавіатура є основною, і користувач може вручну перемикатися між встановленими клавіатурами через інтерфейс iOS

(наприклад, натискаючи на іконку глобуса).

**Вбудована функціональність:** iOS дозволяє користувачам встановлювати сторонні клавіатури з App Store та перемикатися між ними. Однак, як і на macOS, iOS не має вбудованого механізму, який би автоматично запам'ятовував та перемикав розкладку клавіатури для кожної конкретної програми або навіть для різних полів введення в межах однієї програми. Користувач повинен вручну обирати бажану клавіатуру (і, відповідно, її розкладку) щоразу, коли він хоче її

змінити.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.01.АТССПЗ

Існуючі кастомні клавіатури (приклади):

* **Gboard (клавіатура Google):** Пропонує розширений функціонал, такий як пошук, GIF-файли, голосовий ввід, підтримка кількох мов. Gboard дозволяє швидко перемикатися між мовами введення в межах самої клавіатури. Однак, вона не запам'ятовує обрану мову для конкретного додатку і не перемикає її автоматично при зміні програми.
* **SwiftKey Keyboard (Microsoft):** Також пропонує інтелектуальний набір тексту, передбачення слів та підтримку кількох мов. Подібно до Gboard, вона дозволяє перемикатися між мовами всередині клавіатури, але не автоматизує цей процес на основі активного додатку.
* **Інші сторонні клавіатури:** Багато інших кастомних клавіатур на App Store зосереджені на зміні зовнішнього вигляду, інтеграції смайликів, або інших додаткових функціях, але функціонал автоматичного запам'ятовування та перемикання розкладки для різних додатків зустрічається вкрай рідко або відсутній.

**Обґрунтування потреби:** Незважаючи на наявність широкого вибору кастомних клавіатур для iOS, жодна з них не надає системного рішення для автоматичного перемикання мовної розкладки на основі активного додатку. Це створює явну потребу в розробці спеціалізованої кастомної клавіатури, яка буде включати інтелектуальну логіку керування розкладками, адаптуючись до контексту роботи користувача в різних програмах, що й є метою даної роботи.

**Висновок за розділом «Аналіз технології та середовища для створення ПЗ»**

У ході аналізу встановлено, що MacOS забезпечує високу стабільність, безпеку та інтуїтивно зрозумілий інтерфейс, що робить її привабливою для професійного використання. Однак, обмежений вибір програмного забезпечення та недостатня підтримка розважальних програм є суттєвими недоліками. Мова програмування Swift та фреймворк AppKit забезпечують ефективні засоби для створення продуктивних, безпечних і сучасних додатків. Також було розглянуто можливості інтеграції старих технологій через взаємодію з Objective-C та використання координаторів для організації складних сценаріїв роботи інтерфейсу. Проведений аналіз підтвердив доцільність використання обраних технологій для реалізації проєкту.

**2. Дослідження та вибір необхідних компонентів для створення програмного забезпечення**

Змн

Арк.

№\_Документа№ документа

Підпис

Дата

Аркуш

КНУ.РБ.123.25.14.02.ДВНКСПЗ

КНУ.ПК.123.22.01.Р

Розробив

Каменев

Перевірив

Кузнєцов

Н.контроль

Кузнєцов

Затвердив

Купін

Дослідження та вибір необхідних компонентів для створення програмного забезпечення

Літера

Аркушів

КІ-21

Ефективна розробка багатоплатформного програмного забезпечення вимагає ретельного вибору відповідних середовищ розробки, мов програмування, фреймворків та інструментів. Цей розділ присвячений обґрунтуванню вибору ключових компонентів, які будуть використані для реалізації десктопного додатку для macOS, мобільного додатку для iOS та веб-сайту

**2.1 Середовища розробки та мови програмування**

Вибір середовища розробки (IDE) та мови програмування є фундаментальним рішенням, що впливає на продуктивність розробки, якість коду та можливості інтеграції з цільовими платформами. Для даного проекту було обрано спеціалізовані інструменти, які найкраще відповідають вимогам нативної розробки для Apple-екосистеми та ефективної веб-розробки.

**2.1.1 Xcode (для macOS та iOS додатків)**

Призначення та роль у проекті:

* **Єдиний офіційний інструмент:** Xcode є єдиним офіційним та повноцінним інструментом, рекомендованим Apple для розробки нативних додатків під macOS та iOS. Його використання є безальтернативним для забезпечення глибокої інтеграції з операційними системами, доступу до всіх необхідних системних API та дотримання стандартів Apple.
* **Повний цикл розробки:** Xcode включає в себе редактор коду з підсвічуванням синтаксису та автодоповненням для Swift та Objective-C, вбудований інтерфейсний редактор (Interface Builder), потужний відладчик (debugger), інструменти для аналізу продуктивності (Instruments), симулятори для різних пристроїв (iPhone, iPad, Mac) та інтеграцію з Apple Developer Program для розгортання додатків.

**Робота на MacBook:** Весь процес розробки додатків для macOS та iOS відбувається на MacBook, що є оптимальним, оскільки Xcode працює виключно на macOS, а симулятори та можливість підключення реальних пристроїв Apple безпосередньо інтегровані в середовище.

**2.1.2 Swift (для macOS та iOS додатків)**

Опис: Swift – це сучасна, потужна, безпечна та інтуїтивно зрозуміла мова програмування, розроблена компанією Apple Inc. Вона призначена для створення програмного забезпечення на платформах Apple, Linux, а також може бути використана для серверної розробки. Swift була спроектована з акцентом на продуктивність, безпеку коду та зручність для розробника.

Призначення та роль у проекті:

- Основна мова для Apple-платформ: Swift є основною та рекомендованою мовою для розробки нативних додатків для macOS та iOS. Її використання дозволяє повною мірою скористатися можливостями екосистеми Apple та інтегруватися з останніми фреймворками, такими як SwiftUI та Combine.

- Безпека та продуктивність: Мова Swift мінімізує поширені помилки програмування (наприклад, роботу з nil), завдяки чому код стає більш надійним. Вона також забезпечує високу продуктивність, наближену до C++, що критично важливо для додатків, які взаємодіють з низькорівневими системними API.

- Сучасний синтаксис: Синтаксис Swift є лаконічним та виразним, що спрощує читання та написання коду, сприяючи швидшій розробці та легшій підтримці проекту.

- Уніфікована розробка: Використання Swift для обох нативних додатків (macOS та iOS) дозволяє, за потреби, обмінюватися деякими спільними компонентами або логікою, що не залежить від платформи, між проектами, використовуючи Swift Package Manager.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.02.ДВНКСПЗ

**2.1.3 Visual Studio Code (для веб-сайту)**

**Опис:** Visual Studio Code (VS Code) – це легкий, але надзвичайно потужний та гнучкий редактор коду, розроблений компанією Microsoft. Він є кросплатформенним, що дозволяє використовувати його на Windows, Linux та macOS. VS Code здобув широку популярність серед веброзробників завдяки своїй функціональності, швидкодії та великій екосистемі розширень.

**Призначення та роль у проекті:**

* **Універсальний редактор для веб-розробки:** VS Code ідеально підходить для розробки веб-сайту, який включає HTML, CSS та JavaScript. Він надає інтелектуальне автодоповнення, підсвічування синтаксису, форматування коду та підтримку для сучасних фронтендфреймворків та бібліотек (якщо вони будуть використовуватися).
* **Широкий набір розширень:** Екосистема розширень VS Code дозволяє значно розширити його функціонал: від підтримки різних мов програмування та фреймворків до інтеграції з системами контролю версій, інструментами для тестування та розгортання.
* **Вбудовані інструменти:** VS Code має вбудований термінал, що дозволяє виконувати команди командного рядка безпосередньо з редактора (наприклад, для запуску сервера розробки або встановлення npm-пакетів). Також присутня глибока інтеграція з Git для контролю версій.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.02.ДВНКСПЗ

**Робота на MacBook:** Як і Xcode, VS Code бездоганно працює на MacBook, що дозволяє виконувати всю розробку (як нативні додатки, так і веб-сайт) на одному пристрої, спрощуючи робочий процес та управління проектами.

**2.2 Фреймворки та бібліотеки**

Для реалізації комплексного програмного забезпечення, що включає десктопний, мобільний додатки та веб-сайт, необхідний вибір відповідних фреймворків та бібліотек. Вони забезпечують стандартизовані підходи до розробки, прискорюють процес створення програмного забезпечення та дозволяють використовувати вже протестовані та оптимізовані компоненти.

**2.2.1 Для macOS додатку**

Розробка десктопного додатку для macOS вимагає використання нативних фреймворків Apple, які надають доступ до системних функцій та забезпечують відповідність стандартам користувацького інтерфейсу macOS.

**AppKit:**

* **Призначення:** AppKit є основним фреймворком Apple для розробки графічних інтерфейсів користувача (GUI) для macOS-додатків. Він надає клас NSApplication, що забезпечує базову структуру додатку, керування життєвим циклом, роботу з RunLoop (циклом подій) та взаємодію з системними подіями. AppKit дозволяє створювати вікна, кнопки, текстові поля та інші елементи інтерфейсу, а також обробляти події від клавіатури та миші.
* **Роль у проекті:** Навіть для додатків, які функціонують переважно у системному меню (menubar app), AppKit є фундаментом, забезпечуючи роботу додатка як повноцінної програми macOS, взаємодію з системою та відображення елементів інтерфейсу (наприклад, іконки в меню-барі та контекстних меню).

**Carbon Framework (Text Input Services - TIS)**

* **Призначення:** Carbon Framework є застарілою, але критично важливою частиною macOS API для даного проекту. Зокрема, її підрозділ Text Input Services (TIS) надає функції для управління джерелами введення (Input Sources), до яких відносяться розкладки клавіатури.
* **Роль у проекті:** Функції TISCopyCurrentKeyboardInputSource() (для отримання поточної розкладки) та TISSelectInputSource(\_:) (для встановлення розкладки) є безальтернативними та унікальними для програмного керування розкладками на macOS. Незважаючи на те, що Carbon загалом застарілий, ці конкретні функції досі є основним і найбільш

надійним методом для виконання поставленої задачі, оскільки їхній функціонал не був повністю перенесений або замінений у новіших фреймворках, таких як Cocoa.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.02.ДВНКСПЗ

Accessibility API (через AppKit/Foundation):

* **Призначення:** Accessibility API надає програмний доступ до елементів користувацького інтерфейсу, що дозволяє додаткам взаємодіяти з іншими програмами та отримувати інформацію про їхній стан. Це є важливим для спеціальних можливостей, але також використовується для автоматизації та отримання контексту.
* **Роль у проекті:** За допомогою Accessibility API та класів, таких як NSWorkspace.shared.frontmostApplication, додаток може отримувати інформацію про активне (переднє) вікно та програму, яка ним володіє. Це дозволяє ідентифікувати, в якому додатку користувач працює в даний момент, і, відповідно, застосувати логіку перемикання розкладки. Для використання Accessibility API, додаток потребує явного дозволу користувача в системних налаштуваннях безпеки.

Foundation:

* **Призначення:** Foundation є базовим фреймворком для розробки на платформах Apple. Він надає фундаментальні класи та інтерфейси для роботи з даними, колекціями (масивами, словниками), рядками, датами, файловою системою, мережею, а також механізмами для роботи з потоками, таймерами (Timer) та іншими основними функціями.

**Роль у проекті**: Foundation є невід'ємною частиною будь-якого Swiftдодатку, забезпечуючи базову інфраструктуру для зберігання налаштувань, обробки логіки, роботи з часом та взаємодії з операційною системою на базовому рівні.

**2.2.2 Для iOS додатку**

Розробка мобільного додатку для iOS вимагає використання специфічних фреймворків Apple, орієнтованих на мобільні пристрої, з урахуванням сенсорного інтерфейсу та обмежених системних ресурсів.

**SwiftUI:**

* **Призначення:** SwiftUI – це декларативний фреймворк від Apple для побудови користувацьких інтерфейсів на всіх платформах Apple, включаючи iOS. Він дозволяє розробникам описувати бажаний стан інтерфейсу, а система автоматично оновлює його при зміні даних. Це значно спрощує та прискорює розробку UI.
* **Роль у проекті:** SwiftUI буде використовуватися для створення сучасного, адаптивного та інтуїтивно зрозумілого інтерфейсу мобільного додатку. Хоча основна логіка управління розкладками буде реалізована через UIKit, SwiftUI ідеально підходить для розробки основного інтерфейсу додатку-контейнера для клавіатури, вікон налаштувань та інших елементів UI.

**UIKit:**

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.02.ДВНКСПЗ

* **Призначення:** UIKit є фундаментальним фреймворком для розробки користувацьких інтерфейсів на iOS. Він надає класи для створення вікон, представлень (views), контролерів представлень (view controllers), обробки подій та багатьох інших елементів, які формують мобільний додаток.
* **Роль у проекті:** UIKit є обов'язковим для розробки кастомних клавіатур (Input Method Extension). Хоча SwiftUI може бути використаний для загального інтерфейсу, створення самого розширення клавіатури та його компонентів, а також взаємодія з текстовими полями інших додатків, вимагає використання UIKit. Це єдиний офіційний шлях для отримання контролю над віртуальною клавіатурою на iOS для реалізації логіки перемикання розкладок.

**Foundation:**

* **Призначення:** Як і для macOS, Foundation є базовим фреймворком для iOS-додатків. Він надає основні типи даних, колекції, можливості роботи з файловою системою, мережею та інші утиліти, необхідні для будь-якої програми.

**Роль у проекті:** Використовується для зберігання налаштувань клавіатури, управління даними про розкладки, обробки строк та інших базових операцій, які не залежать від інтерфейсу.

**2.2.3 Для веб-сайту**

Для розробки веб-сайту використовуються стандартні вебтехнології, які забезпечують кросбраузерну сумісність та доступність на різних пристроях.

**HTML (HyperText Markup Language):**

* **Призначення:** Стандартна мова розмітки, що використовується для створення структури та змісту веб-сторінок. Вона визначає заголовки, параграфи, списки, посилання, зображення, форми та інші елементи, з яких складається веб-сторінка.
* **Роль у проекті:** HTML буде основою для всіх сторінок веб-сайту, надаючи їхню базову структуру для відображення інформації про додатки, посилань для завантаження, контактних даних тощо.

**CSS (Cascading Style Sheets):**

* **Призначення:** Мова стилів, що використовується для опису візуального представлення документа, написаного на HTML. CSS контролює кольори, шрифти, розміри, розташування елементів, анімації та адаптацію дизайну під різні пристрої (responsive design).
* **Роль у проекті:** CSS забезпечить естетичний та функціональний дизайн веб-сайту, роблячи його привабливим та зручним для користувачів на різних екранах (десктоп, мобільні пристрої).

**JavaScript:**

* **Призначення:** Мова програмування, що дозволяє додавати

інтерактивність та динамічну поведінку на веб-сторінки. Вона може використовуватися для валідації форм, анімацій, взаємодії з API, завантаження контенту без перезавантаження сторінки та багато іншого.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.02.ДВНКСПЗ

* **Роль у проекті:** JavaScript може бути використаний для реалізації таких функцій, як плавне прокручування, анімації, або, якщо сайт має динамічну частину, для взаємодії з бекендом (наприклад, для підрахунку завантажень). Для простого сайту дистрибуції його використання може бути мінімальним.

**Фронтенд-фреймворки/бібліотеки (опціонально):**

* **Обґрунтування вибору:** Для сайту, що в основному служить для дистрибуції та статичної інформації, використання складних фронтенд-фреймворків, таких як React, Vue.js або Angular, може бути надмірним. Якщо ж планується більш інтерактивний інтерфейс, динамічне оновлення контенту, або складні компоненти, тоді їх використання може бути виправданим.
* **Рішення для проекту:** Для поточної задачі, що полягає у забезпеченні інформації та посилань для завантаження, достатньо чистого HTML, CSS та JavaScript. Це дозволить створити легкий, швидкий та простий у підтримці веб-сайт. Якщо пізніше буде потреба у складній інтерактивності, можна розглянути використання легких бібліотек або фреймворків.

**Бекенд-технології (опціонально):**

* **Обґрунтування вибору:** Для сайту, основним завданням якого є дистрибуція, бекенд може бути відсутнім взагалі (якщо це статичний сайт, розміщений на хостингу, який підтримує тільки статичні файли). Якщо ж виникає потреба у функціоналі, що вимагає серверної логіки (наприклад, підрахунок завантажень, система зворотного зв'язку, динамічне оновлення контенту з бази даних), тоді необхідно буде використовувати бекенд-технології.

**Рішення для проекту:** На початковому етапі для функціоналу дистрибуції бекенд може не знадобитися. Веб-сайт може бути реалізований як статичний ресурс. У разі потреби розширення функціоналу (наприклад, для збору статистики завантажень), може бути використаний Node.js з фреймворком Express.js, як популярний та гнучкий стек для серверної розробки на JavaScript, що дозволить зберігати дані у базі даних (наприклад, SQLite або MongoDB).

**2.3 Системи управління залежностями та контролю версій**

Для розробки нативних додатків під macOS та iOS використовуються інструменти, інтегровані в екосистему Apple, а також загальноприйняті системи контролю версій.

**Swift Package Manager (SPM):**

* **Призначення: Swift** Package Manager (SPM) – це офіційний, нативний інструмент для управління залежностями в проектах Swift. Він дозволяє автоматично завантажувати, компілювати та пов'язувати сторонні бібліотеки (пакети Swift) з проектом. SPM інтегрований безпосередньо в Xcode, що спрощує його використання.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.02.ДВНКСПЗ

* Роль у проекті: SPM буде використовуватися для інтеграції будь-яких сторонніх Swift-бібліотек або модулів, які можуть знадобитися для реалізації функціоналу як десктопного, так і мобільного додатків (наприклад, для роботи з даними, мережею або для певних UIкомпонентів, якщо це не буде реалізовано нативно).

**CocoaPods (якщо використовуєш):**

* **Призначення: CocoaPods** – це децентралізований менеджер залежностей для проектів Objective-C та Swift. Він є популярним вибором для управління великою кількістю сторонніх бібліотек, які можуть бути написані як на Objective-C, так і на Swift.
* **Роль у проекті:** Якщо у проекті виникне потреба у використанні бібліотек, які доступні лише через CocoaPods (або якщо їх зручніше інтегрувати саме так), то цей інструмент буде задіяний паралельно зі Swift Package Manager.

**Git:**

* **Призначення:** Git – це розподілена система контролю версій (DVCS) з відкритим вихідним кодом. Вона дозволяє розробникам відстежувати зміни в коді, повертатися до попередніх версій, працювати над проектом паралельно без конфліктів та ефективно співпрацювати в команді.

**Роль у проекті:** Git є єдиним інструментом контролю версій, що використовується для всіх частин проекту – як для коду macOS та iOS додатків, так і для коду веб-сайту. Він забезпечує збереження історії змін, можливість створення окремих гілок для розробки нових функцій, легке об'єднання коду та відновлення у разі виникнення проблем.

**2.3.1 Для веб-сайту**

Для управління залежностями у веб-розробці та для контролю версій також використовуються загальноприйняті стандарти та інструменти.

**npm (Node Package Manager) або Yarn:**

* **Призначення:** npm (Node Package Manager) та Yarn є менеджерами пакетів для JavaScript. Вони використовуються для встановлення, управління та оновлення бібліотек, фреймворків, утиліт та інших залежностей, які необхідні для фронтенд- та, за потреби, бекендчастини веб-проекту. npm зазвичай встановлюється разом з Node.js, а Yarn є альтернативним менеджером з покращеною продуктивністю та надійністю.
* **Роль у проекті:** Ці менеджери пакетів будуть використовуватися для інтеграції будь-яких сторонніх JavaScript-бібліотек (наприклад, для

каруселі, анімацій, або якщо буде обрано такий фреймворк як React/Vue) або інструментів розробки (наприклад, для мініфікації CSS/JS) у веб-сайт.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.02.ДВНКСПЗ

**Git:**

* **Призначення:** Як і для нативних додатків, Git є основною системою контролю версій для веб-сайту.

**Роль у проекті:** Використання Git для веб-проекту дозволяє відстежувати всі зміни в HTML, CSS, JavaScript файлах, керувати різними версіями сайту, співпрацювати над його розробкою (якщо це командний проект) та легко розгортати нові версії на хостинг.

**2.4. Аналіз нативного та кросплатформенного підходу**

При розробці програмного забезпечення одним з ключових рішень є вибір між нативним та кросплатформенним підходом. Нативний підхід передбачає використання специфічних для кожної платформи мов програмування, фреймворків та інструментів, що забезпечує максимальну продуктивність, глибоку інтеграцію з операційною системою та відповідність її стандартам. Кросплатформенні рішення, своєю чергою, дозволяють писати код один раз для кількох платформ, що може прискорити розробку, але часто накладає обмеження на доступ до низькорівневих системних функцій та на продуктивність.

Для даного проекту вибір підходу був обумовлений специфічними вимогами до функціоналу, який потребує глибокої взаємодії з операційною системою та її механізмами введення.

**Нативний підхід для macOS:**

* **Обґрунтування:** Для десктопного додатку на macOS обрано суто нативний підхід, використовуючи мову Swift та фреймворки AppKit, Carbon Framework (зокрема Text Input Services) та Accessibility API. Цей вибір є безальтернативним та єдиним можливим шляхом для реалізації функції автоматичного перемикання розкладок клавіатури на рівні операційної системи. Зазначена функціональність вимагає глибокого доступу до низькорівневих системних API (таких як TIS та Accessibility API), які не можуть бути повноцінно досягнуті за допомогою кросплатформенних фреймворків (наприклад, Electron, React Native Desktop або Flutter Desktop) без значних обмежень, складнощів або порушення системних дозволів. Нативний підхід гарантує максимальну продуктивність, стабільність роботи додатку та бездоганну інтеграцію з користувацьким інтерфейсом та системними процесами macOS.

**Нативний підхід для iOS:**

* **Обґрунтування:** Для мобільного додатку на iOS також обрано нативний підхід, використовуючи мову Swift та фреймворки SwiftUI і UIKit. Це дозволяє розробляти високопродуктивний додаток з оптимальним інтерфейсом, який повністю відповідає Apple Human Interface Guidelines. Найважливіше – це дає змогу використовувати механізми Custom Keyboard

Extensions (власних клавіатур), що є ключовим для реалізації функціоналу управління розкладками на мобільній платформі. Кросплатформенні рішення, хоча і спрощують процес розробки та дозволяють використовувати єдину кодову базу, часто мають обмежений доступ до специфічних системних функцій та розширень, які є критично важливими для даного проекту (як-от реалізація власної клавіатури).

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.02.ДВНКСПЗ

**Веб-сайт:**

* **Вибір:** Для веб-сайту використовується стандартний веб-стек, що включає HTML, CSS та JavaScript. У даному проекті складні фронтенд-фреймворки або бекенд-системи не є критично необхідними, оскільки основне завдання сайту – це інформаційна підтримка та дистрибуція додатків.

**Обґрунтування:** Завданням веб-сайту є інформаційна підтримка проекту, надання користувачам можливості дізнатися більше про додатки та завантажити їх дистрибутиви. Це завдання не вимагає глибокої інтеграції з операційною системою чи доступу до низькорівневих функцій. Веб-технології є універсальними, доступними з будь-якого пристрою та браузера, і найбільш підходящими для створення публічного інформаційного ресурсу. Використання Visual Studio Code дозволяє ефективно та швидко розробляти фронтенд-частину веб-сайту.

**Висновок за розділом «Дослідження та вибір необхідних компонентів для створення програмного забезпечення»**

У процесі дослідження було проведено комплексний аналіз та обґрунтований вибір інструментів, фреймворків та підходів, необхідних для розробки всіх компонентів програмного комплексу: нативних застосунків під macOS та iOS, а також веб-сайту.

Основну увагу приділено середовищу розробки Xcode, його можливостям, перевагам і недолікам як центрального інструменту для десктопної та мобільної розробки в екосистемі Apple. Обґрунтовано вибір Swift як основної мови програмування та SwiftUI/AppKit як ключових фреймворків для створення інтерфейсу та логіки додатків на платформах Apple. Детально розглянуто особливості використання Carbon Framework (Text Input Services) та Accessibility API (AXUIElement), які є критично важливими для реалізації функції автоматичного керування розкладками на macOS, а також механізми Custom Keyboard Extensions у UIKit для забезпечення аналогічного функціоналу на iOS. Для розробки веб-сайту обрано Visual Studio Code та стандартний веб-стек (HTML, CSS, JavaScript), як найбільш оптимальні для інформаційного порталу та дистрибуції.

Аналіз нативного та кросплатформенного підходів показав безальтернативність нативного підходу для розробки функціоналу з глибокою інтеграцією в системи macOS та iOS, оскільки він забезпечує необхідний рівень

доступу до системних API та оптимальну продуктивність. Для веб-сайту обрано стандартний веб-підхід, що відповідає його призначенню.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.02.ДВНКСПЗ

Дотримання Apple Human Interface Guidelines та використання офіційних інструментів і фреймворків Apple забезпечує відповідність високим стандартам якості, зручності користування, безпеки та стабільності. У підсумку було обрано найефективніші рішення для створення стабільного, продуктивного та багатоплатформного програмного комплексу, здатного вирішити поставлені завдання.

**3.Проектування та реалізація програмного забезпечення**

Змн

Арк.

№\_Документа№ документа

Підпис

Дата

Аркуш

КНУ.РБ.123.25.14.03.ПТРПЗ

КНУ.ПК.123.22.01.Р

Розробив

Каменев

Перевірив

Кузнєцов

Н.контроль

Кузнєцов

Затвердив

Купін

Проектування та реалізація програмного забезпечення

Літера

Аркушів

КІ-21

**3.1. Архітектура програмного комплексу**

Розроблюваний програмний комплекс є багатокомпонентною системою, призначеною для вирішення проблеми неефективного управління розкладками клавіатури на різних платформах. Він складається з трьох ключових частин: десктопного додатку для macOS, мобільного додатку для iOS (реалізованого у вигляді Custom Keyboard Extension) та вебсайту, що слугує для дистрибуції та інформаційної підтримки.

Кожен компонент відіграє специфічну, але взаємодоповнюючу роль у загальному функціонуванні продукту:

* **macOS Додаток:** Є основним рішенням для користувачів десктопних систем Apple. Його роль полягає в автоматичному перемиканні розкладки клавіатури відповідно до активного додатку або вікна на macOS, забезпечуючи максимальну продуктивність та зручність роботи з текстом. Він глибоко інтегрується з операційною системою для відстеження активних вікон та керування джерелами введення.
* **iOS Додаток (Custom Keyboard Extension):** Розроблений для мобільних користувачів, що потребують аналогічної функціональності на пристроях Apple iOS. Основна роль цього компоненту полягає в наданні кастомної клавіатури, яка дозволяє запам'ятовувати та автоматично активувати відповідну розкладку для кожного окремого мобільного додатку, тим самим усуваючи необхідність ручного перемикання.
* **Веб-сайт:** Виступає як центральна точка входу та інформаційний портал для всього програмного комплексу. Його основна роль — забезпечення легкого доступу до інформації про додатки, їхні можливості, переваги, а також надання зручного механізму для завантаження дистрибутивів macOS та iOS додатків. Він також слугує як джерело новин та підтримки користувачів.

Пряма програмна взаємодія (наприклад, синхронізація налаштувань через хмарний сервіс або спільна база даних) між macOS, iOS додатками та вебсайтом на поточному етапі реалізації не передбачається. Проте, всі три компоненти є частинами єдиного продуктового бачення, спільно вирішуючи проблему ефективного управління розкладками на різних платформах та надаючи користувачам цілісне, багатоплатформне рішення. Об'єднання цих компонентів в єдиний програмний комплекс дозволяє охопити ширший спектр користувачів та забезпечити комплексну підтримку для їхніх повсякденних завдань.



арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

Рисунок 3.1 – Загальна архітектурна схема програмного комплексу.

**3.1.1. Архітектура додатку для macOS**

Архітектура додатку для macOS розроблена з урахуванням необхідності стабільного та ефективного моніторингу системних подій, глибокої взаємодії з операційною системою для керування розкладками клавіатури, та мінімізації впливу на загальну продуктивність системи. Додаток побудований за модульним принципом, що забезпечує чітке розділення відповідальності між компонентами, спрощує розробку, тестування та подальшу підтримку.

Основні логічні частини (модулі) macOS додатку та їхнє призначення:

**1. Головний модуль додатку (Main Application Logic/Core):**

* **Призначення:** Цей модуль є ядром додатку. Він відповідає за ініціалізацію всіх інших модулів, керування життєвим циклом додатку (запуск, зупинка), координацію їхньої взаємодії та обробку загальних подій.

Він слугує центральним диспетчером, що забезпечує злагоджену роботу всіх компонентів.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

* **Використовувані API/Фреймворки:** Взаємодіє з базовими компонентами AppKit та Foundation для ініціалізації та управління основним циклом додатку.

**2. Модуль моніторингу системних подій (System Event Monitor):**

* **Призначення:** Це ключовий модуль для реалізації основного функціоналу. Його завдання – безперервне відстеження зміни активного додатку або вікна на macOS. Він реагує на події переключення контексту, надаючи інформацію про поточний активний об'єкт.
* **Використовувані API/Фреймворки:** Активно використовує NSWorkspace для отримання інформації про активний додаток (NSWorkspace.shared.frontmostApplication) та Accessibility API (зокрема, AXUIElement та AXIsProcessTrustedWithOptions) для отримання доступу до даних про активне вікно та забезпечення необхідних системних дозволів.

**3. Модуль управління розкладками (Input Source Manager):**

* **Призначення:** Відповідає за програмне визначення поточної розкладки клавіатури та її автоматичну зміну. Він отримує запити від Модуля моніторингу системних подій та на основі даних з Модуля зберігання налаштувань виконує необхідні операції.
* **Використовувані API/Фреймворки:** Для взаємодії з системою введення використовує низькорівневі функції Carbon Framework, зокрема Text Input Services (TIS). Це дозволяє отримати список доступних розкладок (TISCopyInputSourceList()), визначити поточну активну розкладку (TISCopyCurrentKeyboardInputSource()) та здійснити програмне перемикання на потрібну

(TISSelectInputSource()).

**4. Модуль зберігання налаштувань (Preferences/Settings Module):**

* Призначення: Цей модуль відповідає за збереження та завантаження всіх користувацьких налаштувань. Сюди входять: мапінг "додатокрозкладка" (тобто, яка розкладка має бути активною для певного додатку), стан активації автоматичного перемикання, а також будьякі інші конфігураційні параметри.
* **Використовувані API/Фреймворки:** Для постійного зберігання даних використовується UserDefaults (для простих ключ-значення пар) або може бути реалізований інший механізм персистентності для складніших структур даних.

**5. Модуль інтерфейсу користувача (UI Module - Menu Bar Item):**

* Призначення: Відповідає за візуальне представлення додатку та взаємодію користувача з ним. Оскільки додаток є фоновою службою, основною точкою взаємодії є іконка в меню-барі macOS. Цей модуль керує відображенням іконки, її контекстним меню (з опціями активації/деактивації, доступу до налаштувань) та, за наявності, окремим

вікном налаштувань.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

* **Використовувані API/Фреймворки:** AppKit (для NSStatusItem, NSMenu, NSWindow та інших UI-елементів) є основним фреймворком для реалізації цього модуля.

**Взаємодія компонентів:**

Взаємодія між модулями є цілісною та логічною. Головний модуль додатку ініціалізує Модуль моніторингу системних подій. При виявленні зміни активного додатку або вікна, Модуль моніторингу передає відповідну інформацію до Головного модуля або безпосередньо до Модуля управління розкладками. Модуль управління розкладками, у свою чергу, звертається до Модуля зберігання налаштувань, щоб визначити бажану розкладку для нового контексту. Якщо потрібна зміна, Модуль управління розкладками використовує Carbon TIS для виконання операції перемикання. Модуль інтерфейсу користувача дозволяє користувачеві в будь-який момент налаштувати параметри, які зберігаються Модулем зберігання налаштувань та використовуються іншими модулями.

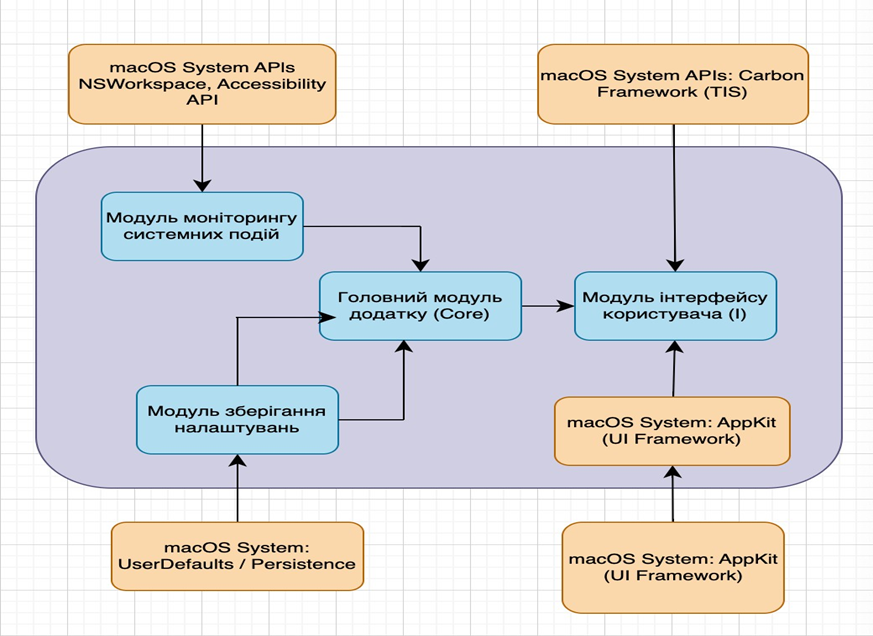


Рисунок 3.2 – Внутрішня архітектура macOS додатку.

**3.1.2. Опис алгоритму роботи перемикання розкладок на macOS**

Основна функціональність macOS додатку полягає в автоматичному перемиканні розкладки клавіатури, що ґрунтується на динамічному моніторингу активного контексту користувача (додатку або окремого вікна) та застосуванні попередньо збережених налаштувань. Алгоритм роботи додатку є послідовним процесом, який забезпечує безперервне та точне управління джерелами введення.

Етапи алгоритму роботи додатку:

**1. Ініціалізація та надання дозволів:**

* При запуску додатку відбувається його ініціалізація, під час якої встановлюються необхідні системні слухачі.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

* Критично важливим кроком є перевірка та запит необхідних дозволів від користувача, зокрема доступу до Accessibility API (API доступності). Цей дозвіл є обов'язковим, оскільки він надає додатку можливість отримувати інформацію про активні вікна та додатки, а також програмно взаємодіяти з елементами інтерфейсу (що потрібно для точного визначення контексту). Якщо дозвіл не надано, додаток пропонує користувачеві перейти до системних налаштувань безпеки та конфіденційності для його активації, без цього дозволу функціональність автоматичного перемикання неможлива.

**2. Моніторинг активних додатків/вікон:**

* Після ініціалізації додаток починає безперервний моніторинг змін в активному робочому просторі macOS. Тригером для запуску логіки перемикання розкладки є зміна активного додатку або активного вікна всередині додатку.
* Для визначення активного додатку використовується фреймворк AppKit, зокрема об'єкт NSWorkspace.shared.frontmostApplication. Це дозволяє отримати ідентифікатор (bundle identifier) додатка, який наразі перебуває на передньому плані.
* Для більш гранульованого контролю та можливості розрізняти розкладки в межах одного додатку (наприклад, для різних вікон або вкладок), використовується Accessibility API. За допомогою AXUIElement додаток може отримати доступ до інформації про активне вікно, включаючи його назву, роль та інші атрибути, що дозволяє більш точно визначити контекст роботи.

**3. Визначення поточної розкладки:**

- Після ідентифікації нового активного контексту додаток запитує інформацію про поточну активну розкладку клавіатури в системі. Це здійснюється за допомогою функцій Carbon Framework (Text Input

Services – TIS), зокрема функції

TISCopyCurrentKeyboardInputSource(). Ця функція повертає об'єкт, що містить деталі про активне джерело введення, включаючи його унікальний ідентифікатор.

**4. Зіставлення з налаштуваннями:**

* Отримавши ідентифікатор активного додатку/вікна, додаток звертається до свого Модуля зберігання налаштувань. Цей модуль містить базу даних (або збережені у UserDefaults відповідності), де користувач раніше встановив, яка розкладка має бути активною для певного додатку чи, можливо, навіть конкретного вікна (якщо реалізовано більш глибоке зіставлення).
* Додаток перевіряє, чи існує збережене правило для поточного активного контексту.

**5. Логіка прийняття рішення щодо перемикання:**

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

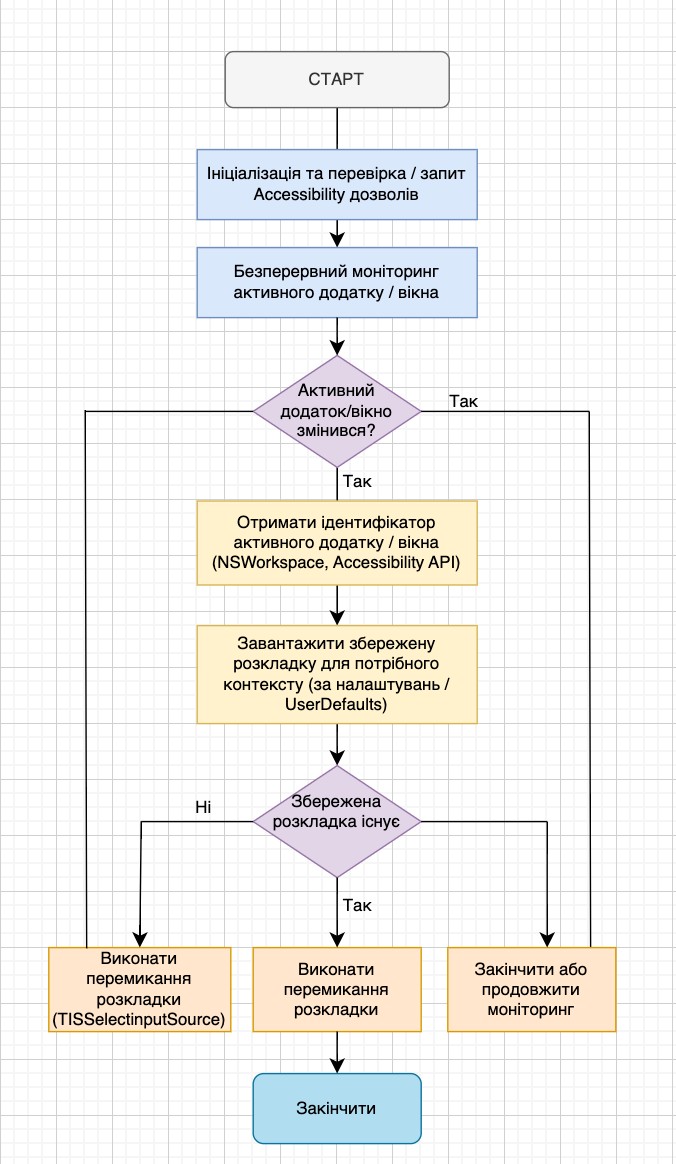
* Рішення про перемикання розкладки приймається на основі кількох умов:
  1. Активний додаток (або вікно) змінився.
  2. Для нового активного контексту існує збережене користувацьке налаштування розкладки.
  3. Бажана розкладка, визначена в налаштуваннях, відрізняється від поточної активної системної розкладки клавіатури.
* Лише при одночасному виконанні всіх цих умов алгоритм переходить до наступного кроку.

**6. Виконання перемикання розкладки:**

- Якщо рішення про перемикання позитивне, додаток програмно активує потрібну розкладку. Це виконується за допомогою функції TISSelectInputSource() з Carbon Framework. Ця функція приймає як аргумент об'єкт, що ідентифікує бажане джерело введення (Input Source ID), який раніше був отриманий з бази доступних розкладок. Після виклику цієї функції операційна система змінює активну розкладку клавіатури.

7. **Обробка винятків та помилок:**

- Протягом всього циклу роботи додаток має обробляти потенційні винятки, такі як відсутність необхідних системних дозволів, помилки під час взаємодії з системними API, або ситуації, коли бажана розкладка недоступна в системі. Користувачеві надаються відповідні сповіщення або інструкції.



арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

Рисунок 3.3 – Алгоритм автоматичного перемикання розкладки на macOS.

**3.1.3. Проектування інтерфейсу користувача macOS додатку**

Інтерфейс користувача macOS додатку розроблений з урахуванням ключових принципів Apple Human Interface Guidelines (HIG), забезпечуючи мінімалістичний дизайн, інтуїтивність та легкість використання. Оскільки додаток функціонує переважно у фоновому режимі, його взаємодія з користувачем є ненав'язливою та доступною лише за потребою. Основні елементи інтерфейсу включають іконку в меню-барі, контекстне меню та окреме вікно налаштувань.

Основний елемент інтерфейсу – Іконка в Menu Bar:

Іконка додатку розташована в системному меню-барі macOS (зазвичай, у верхньому правому куті екрану). Вона виконує кілька ключових ролей:

* **Індикатор стану:** Іконка може візуально відображати поточний стан роботи додатку (наприклад, активний/неактивний режим автоматичного перемикання).
* **Точка доступу:** Це основна точка взаємодії для швидкого доступу до функціоналу та налаштувань додатку.
* **Візуальна ідентифікація:** Забезпечує легку ідентифікацію додатку серед інших системних утиліт.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

**Контекстне меню (Menu Bar Menu):**

При натисканні на іконку додатку в меню-барі розгортається контекстне меню, яке надає користувачеві швидкий доступ до основних функцій та налаштувань. Типові опції, доступні в цьому меню:

* **"Увімкнути/Вимкнути автоматичне перемикання"**: Дозволяє миттєво активувати або деактивувати основну функцію додатку без необхідності заходити в повні налаштування.
* **"Налаштування..."**: Відкриває окреме вікно налаштувань, де користувач може детальніше конфігурувати поведінку додатку.
* **"Про додаток"**: Відображає інформацію про версію додатку, його розробника та ліцензійну інформацію.
* **"Вийти"**: Дозволяє коректно завершити роботу додатку. Кожна з цих опцій розроблена для забезпечення швидкого контролю та доступу до життєво важливих функцій.

**Вікно налаштувань (Preferences Window):**

Для більш детального керування поведінкою додатку передбачено окреме вікно налаштувань. Це вікно зазвичай має кілька секцій або вкладок для логічного групування параметрів:

* **"Загальні" (General):** Містить базові налаштування, такі як автоматичний запуск додатку при вході в систему.
* **"Правила" (Rules):** Це центральна секція, де користувач встановлює та управляє правилами відповідності "додаток-розкладка". Тут відображається список додатків, для яких встановлені індивідуальні налаштування розкладки. Користувач може додавати нові програми

до цього списку, вибираючи їх зі списку запущених додатків, або ж додаючи вручну за bundle ID. Для кожної доданої програми користувач може вибрати бажану розкладку зі спадного списку всіх доступних системних розкладок. Також тут можна видаляти існуючі правила.

* **"Дозволи" (Permissions):** Надає інформацію про необхідні системні дозволи (зокрема, Accessibility API) та посилання на відповідні секції системних налаштувань macOS, що спрощує процес надання дозволів.

**Взаємодія з користувачем:**

* **Встановлення правил "додаток-розкладка":** Користувач взаємодіє з вікном налаштувань, де може додавати нові програми до списку. Це може бути реалізовано шляхом перетягування іконки програми у вікно налаштувань, або через кнопку "Додати" з подальшим вибором зі списку запущених додатків. Після додавання програми, поруч з її назвою з'являється спадний список (dropdown), з якого користувач обирає бажану розкладку.

**Надання дозволів**: Якщо дозволи Accessibility API не надані, додаток через UI-елементи (наприклад, повідомлення у вікні налаштувань або окремий діалог) направляє користувача до "Системних налаштувань" -> "Приватність і безпека" -> "Доступність", де користувач має вручну надати дозвіл для додатку.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

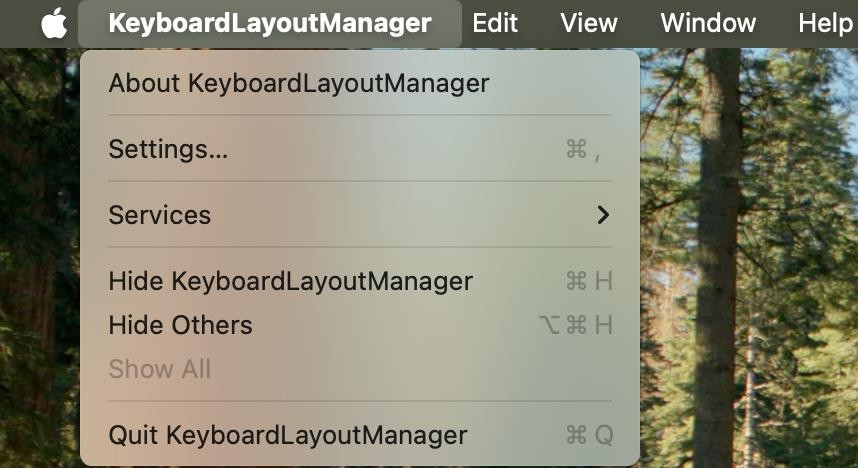


Рисунок 3.4 – Інтерфейс користувача macOS додатку (іконка в Menu Bar та контекстне меню).

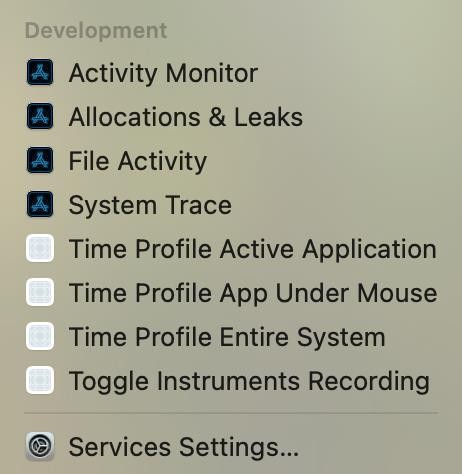


Рисунок 3.5 – Вікно налаштувань macOS додатку.



арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

Рисунок 3.6 – Встановлений додаток у Launchpad на macOS.

**3.2 Проектування та реалізація iOS додатку**

**3.2.1 Архітектура iOS додатку (Кастомна Клавіатура)**

Архітектура iOS додатку для автоматичного управління розкладками клавіатури суттєво відрізняється від десктопного рішення, що зумовлено особливостями платформи iOS та її політикою безпеки (sandbox). Даний додаток реалізовано як двокомпонентну систему, що складається з основного додатку-контейнера (Container App) та розширення клавіатури (Custom Keyboard Extension). Такий підхід є необхідним, оскільки дозволяє дотримуватися обмежень безпеки iOS, одночасно надаючи користувачеві можливість зручно керувати розкладками.

**Основний додаток-контейнер (Container App):**

* **Роль:** Основний додаток слугує контейнером для Custom Keyboard Extension. Його головне призначення – забезпечення інтерфейсу для управління налаштуваннями клавіатури, демонстрації функціоналу, надання інструкцій з активації розширення, а також, у випадку більш складної реалізації, управління мапінгами розкладок для різних мобільних додатків. Цей компонент також може містити всі необхідні ресурси, які потім можуть бути використані розширенням (наприклад, зображення клавіш, локалізаційні рядки тощо).
* **Обмеження:** Відповідно до архітектури iOS, основний додаток працює у власній "пісочниці" і не має прямого доступу до системного введення чи до інших додатків користувача. Він не може самостійно перехоплювати чи змінювати розкладку клавіатури під час роботи в інших програмах. Його функціональність обмежена взаємодією з користувачем всередині власного інтерфейсу та управлінням даними для розширення.

**Розширення Клавіатури (Custom Keyboard Extension):**

**Роль:** Це ядро мобільного рішення, яке є фактичною кастомною клавіатурою. Воно з'являється у будь-якому додатку, що підтримує введення тексту, коли користувач обирає його зі списку доступних клавіатур. Основна роль розширення полягає в:

* **Обробці натискань клавіш:** Реагує на дії користувача на клавіатурі.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

* **Відображенні візуальної розкладки:** Динамічно змінює символи, що відображаються на клавішах, відповідно до поточної або бажаної розкладки.
* **Логіці автоматичного перемикання розкладки:** Найважливіша функція, яка дозволяє розширенню визначати, в якому "хост-додатку" воно зараз активне (наприклад, через bundle ID хост-додатку), і на основі цього автоматично встановлювати візуальну розкладку, збережену для цього хост-додатку.
* **Введення тексту:** Надає оброблений текст назад до хост-додатку через системний API.

**Обмеження:** Розширення клавіатури працює у суворій "пісочниці" (sandbox). Це накладає значні обмеження:

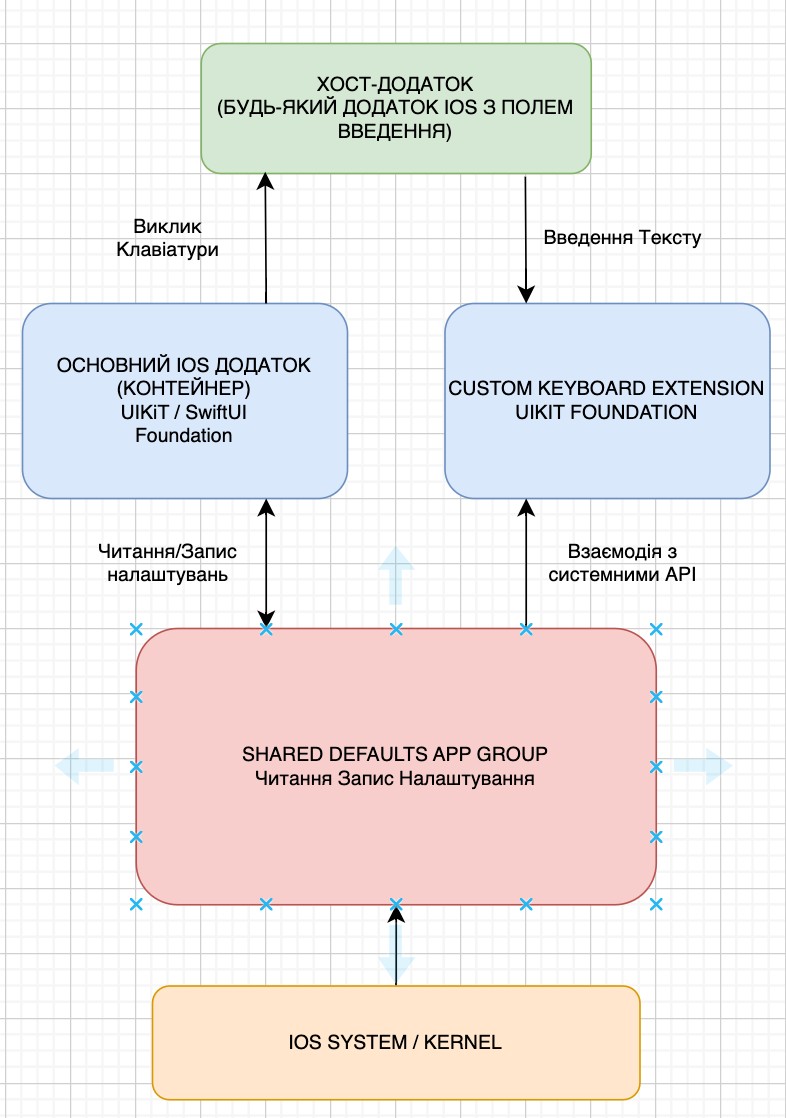
* **Доступ до пам'яті та CPU:** Розширення має обмежені ресурси пам'яті та процесорного часу, що вимагає оптимізованого коду.
* **Доступ до файлової системи:** Обмежений доступ до файлової системи за межами власного контейнера.
* **Доступ до мережі:** Може бути обмежений, що впливає на синхронізацію даних або завантаження додаткових ресурсів.
* **Безпосередній доступ до системи:** Не має повного доступу до системних функцій, як основний додаток, що ускладнює або унеможливлює деякі низькорівневі маніпуляції, які доступні на macOS.

**Взаємодія між Контейнером та Розширенням:**

Взаємодія між основним додатком-контейнером та розширенням клавіатури є критично важливою для обміну даними та налаштуваннями. Найбільш поширеним та рекомендованим механізмом для цього є використання Shared Defaults через App Groups.

* **Shared Defaults / App Groups:** Це дозволяє обом частинам програмного комплексу (контейнерному додатку та його розширенню) безпечно обмінюватися даними та зберігати спільні налаштування у спільному контейнері. Наприклад, якщо користувач встановлює правила "додаток-розкладка" в основному додатку, ці налаштування зберігаються у Shared Defaults, а потім звідти зчитуються Custom Keyboard Extension, коли воно активується в будьякому хост-додатку. Це забезпечує єдність конфігурації та функціоналу між двома окремими частинами програми.

Розробка ведеться з використанням **SwiftUI/UIKit** для побудови інтерфейсів та Foundation для базових функціональностей, що забезпечує нативну продуктивність та відповідність стандартам iOS.



арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

Рисунок 3.7 – Архітектура iOS додатку та Custom Keyboard Extension.

**3.2.2 Опис алгоритму роботи Custom Keyboard Extension**

Основним завданням Custom Keyboard Extension є надання інтелектуальної віртуальної клавіатури, яка адаптує свою візуальну розкладку до конкретного додатку, в якому вона використовується, забезпечуючи при цьому зручне та ефективне введення тексту. Алгоритм роботи розширення клавіатури оптимізовано для середовища iOS, враховуючи обмеження "пісочниці" (sandbox).

**Етапи алгоритму роботи Custom Keyboard Extension:**

**1. Активація Клавіатури:**

* Життєвий цикл розширення клавіатури починається, коли користувач вручну обирає її зі списку доступних клавіатур у системних налаштуваннях iOS (Налаштування > Загальні > Клавіатура > Клавіатури > Додати нову клавіатуру...).
* Після початкової активації, клавіатура запускається автоматично, коли користувач торкається поля введення тексту в будь-

якому додатку iOS і перемикається на неї (наприклад, через кнопку глобуса).

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

2. **Визначення Хост-Додатку та Завантаження Налаштувань:**

* **Визначення хост-додатку:** Через обмеження "пісочниці" Custom Keyboard Extension не може безпосередньо отримати bundle ID хостдодатку, в якому вона працює. Для реалізації функціоналу "розкладка за додатком" було використано механізм, що покладається на збереження останнього активного хост-додатку основним додаткомконтейнером, який має більший доступ до системних даних, а потім передачу цієї інформації до розширення через Shared Defaults. Таким чином, клавіатура "дізнається" про поточний хост-додаток опосередковано.

* **Завантаження налаштувань:** Після визначення хост-додатку (або отримання його ідентифікатора) клавіатура звертається до Shared Defaults (через App Group). Звідти зчитуються попередньо збережені користувацькі налаштування, які містять мапінг "хост-додаток - бажана розкладка".

**3. Візуалізація Розкладки:**

* Перемикання розкладки" на рівні Custom Keyboard Extension – це не зміна системної розкладки (як на macOS), а зміна візуального макету самої віртуальної клавіатури.
* Після отримання бажаної розкладки для поточного хост-додатку, клавіатура динамічно перемальовує свої клавіші. Наприклад, якщо для "Safari" встановлена українська розкладка, клавіатура відобразить українські літери; якщо для "Mail" – англійська, відобразяться англійські. Це забезпечує візуальне відповідність очікуванням користувача.

4. **Обробка Введення Користувача:**

* При натисканні на віртуальну клавішу, розширення клавіатури обробляє цю подію.
* Виходячи з поточної візуальної розкладки та натиснутої клавіші, формується відповідний текстовий символ або команда (наприклад, видалення).
* Сформований текст передається назад до активного поля введення хост-додатку за допомогою об'єкта textDocumentProxy та його методів insertText() для вставки символів або deleteBackward() для видалення.

**5. Взаємодія з Основною Програмою та Ручне Перемикання:**

* **Ручне перемикання:** Клавіатура зазвичай має кнопку глобуса, яка дозволяє користувачеві швидко перемикатися між встановленими системними клавіатурами, включаючи інші мовні розкладки або інші кастомні клавіатури. В межах даної кастомної клавіатури також може бути реалізована кнопка для швидкого циклічного перемикання між декількома основними розкладками, які користувач обрав як "улюблені". Це ручне

перемикання перевизначає автоматичну логіку для поточної сесії введення.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

**Взаємодія з основною програмою:** На клавіатурі може бути передбачена окрема кнопка (наприклад, іконка додатка), яка дозволяє користувачеві швидко перейти до основного додатку-контейнера для доступу до повних налаштувань.

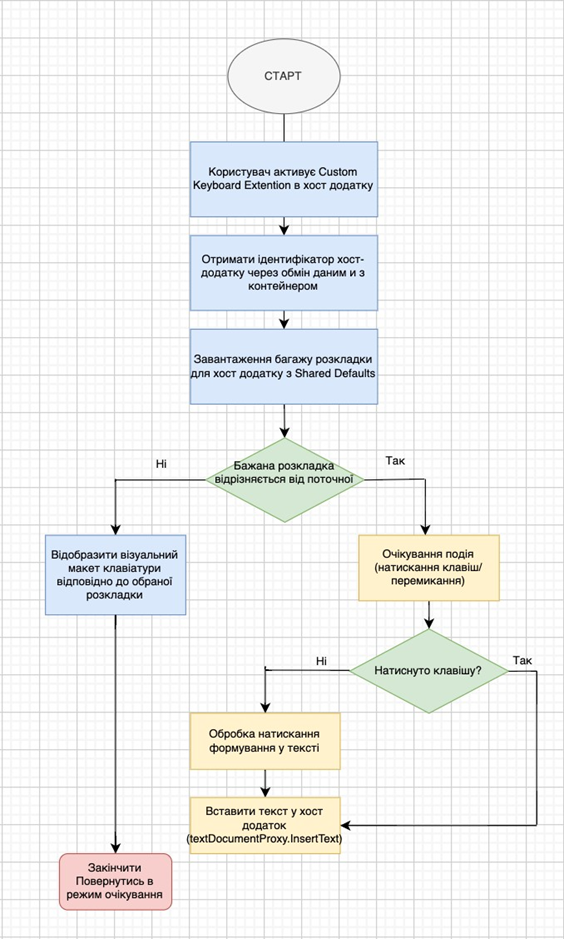


Рисунок 3.8 – Алгоритм роботи Custom Keyboard Extension для управління розкладками.

**3.2.3 Проектування інтерфейсу користувача iOS додатку (Кастомна Клавіатура)**

Інтерфейс користувача iOS додатку, що складається з контейнерної програми та розширення клавіатури, розроблений з урахуванням Apple Human Interface Guidelines (HIG) для мобільних пристроїв. Основний фокус зосереджено на забезпеченні зручності, мінімалізму та ефективності використання на сенсорних екранах, а також на інтуїтивно зрозумілій взаємодії, оскільки частина функціоналу працює у фоновому режимі.

**Інтерфейс Основного Додатку-Контейнера (Container App UI):**

Основний додаток-контейнер виступає в ролі "точки входу" для управління всіма основними налаштуваннями та конфігураціями Custom Keyboard Extension. Його інтерфейс є чистим та функціональним, забезпечуючи легкий доступ до ключових параметрів.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

**Вікно налаштувань:** Головний екран додатку слугує вікном налаштувань, яке поділене на кілька логічних секцій для зручності навігації:

1. "Активація Клавіатури": Ця секція надає покрокові інструкції щодо того, як користувач може активувати кастомну клавіатуру в системних налаштуваннях iOS. Це критично важливий крок, оскільки без активації розширення клавіатури його функціональність недоступна. Інструкції можуть містити текст, візуальні підказки або навіть посилання на відповідні системні налаштування.
2. "Правила Розкладок": Ця секція є центральною для налаштування логіки автоматичного перемикання. Тут користувач може встановлювати правила "додаток - розкладка". Інтерфейс включає:

* Список запущених або встановлених додатків, де користувач може вибрати програми, для яких він хоче задати специфічну розкладку.
* Для кожної обраної програми надається спадний список (або набір кнопок) для вибору бажаної розкладки (наприклад, українська, англійська, тощо) з числа доступних у системі.
* Функціонал для додавання нових правил, редагування існуючих та видалення.

3. "Довідка / Про додаток": Містить інформацію про версію додатку, посилання на підтримку або довідкові матеріали.

Інтерфейс Розширення Клавіатури (Custom Keyboard Extension UI):

Розширення клавіатури є фактичною віртуальною клавіатурою, що з'являється у будь-якому полі введення тексту в iOS. Її дизайн прагне максимально відповідати нативній клавіатурі iOS для забезпечення звичного та зручного досвіду користувача.

1. Візуальна розкладка: Основний простір клавіатури займає візуальне відображення поточної розкладки (літер, цифр, символів). Клавіші динамічно змінюють свій вигляд, відображаючи символи обраної розкладки відповідно до налаштувань.
2. Кнопки керування: На додаток до стандартних клавіш (літери, цифри, пробіл, Backspace, Shift, Enter/Return), клавіатура містить спеціальні функціональні кнопки:

* Кнопка глобуса: Стандартна кнопка iOS для швидкого перемикання між усіма встановленими системними клавіатурами, включаючи інші мовні розкладки або сторонні клавіатури.
* Кнопка перемикання розкладки (спеціальна): Якщо реалізовано, це власна кнопка на клавіатурі, яка дозволяє користувачеві вручну перемикатися між основними розкладками (наприклад, українська/англійська) без зміни клавіатури. Це зручно для швидкої зміни

розкладки в межах одного додатку, перевизначаючи автоматичну логіку.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

Кнопка переходу до контейнерного додатку (опціонально): Це може бути невелика іконка додатку, що дозволяє користувачеві швидко перейти до основного додатку-контейнера для доступу до детальних налаштувань.

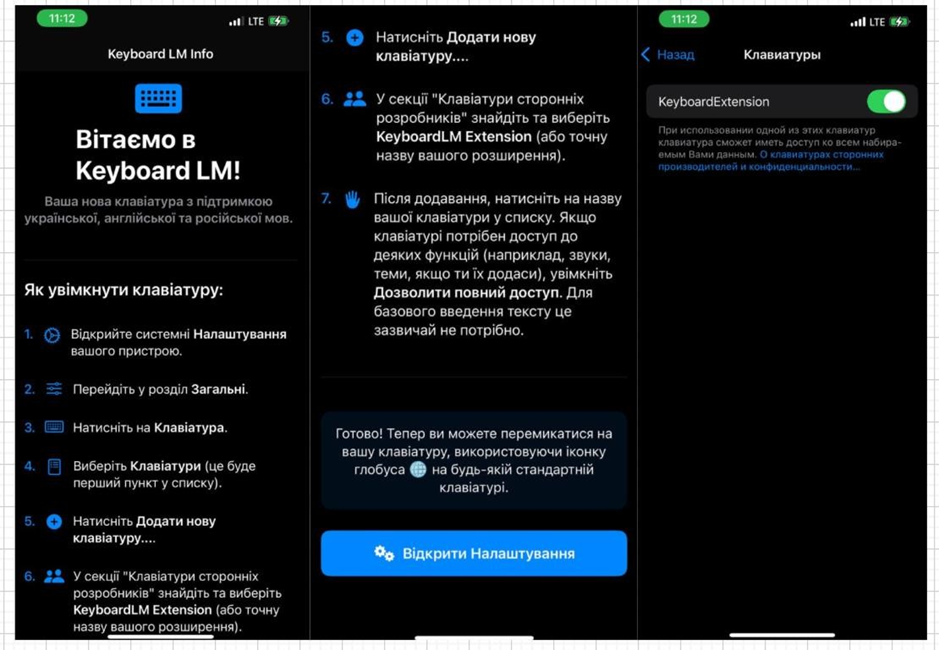


Рисунок 3.9 – Інтерфейс користувача iOS додатку (контейнерна програма).

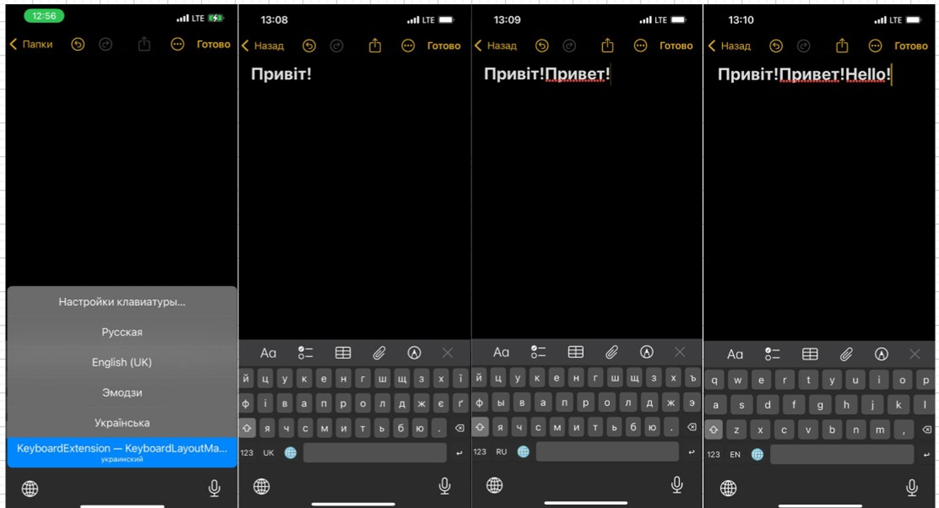


Рисунок 3.10 – Інтерфейс користувача Custom Keyboard Extension.

**3.3 Проектування та реалізація Веб-сайту**

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

**3.3.1 Архітектура Веб-сайту**

Веб-сайт у даному проекті розроблений як інформаційний портал, основним призначенням якого є надання детальної інформації про програмний комплекс (macOS та iOS додатки), його функціональні можливості, а також забезпечення платформи для зручної дистрибуції програмного забезпечення. Архітектура веб-сайту спроектована для забезпечення високої доступності інформації, стабільності роботи та легкості підтримки, оскільки його функціонал переважно є статичним.

**Опис компонентів:**

1. **Фронтенд (Client-Side):**

* **Роль:** Фронтенд веб-сайту відповідає за візуальне представлення всього контенту, взаємодію з користувачем через інтерфейс браузера та відображення інформації. Він забезпечує привабливий зовнішній вигляд, зручну навігацію та адаптивність до різних пристроїв (десктоп, мобільні).

Технології: Для розробки фронтенду використано стандартний веб-стек:

* **HTML (HyperText Markup Language):** Для структурування та розмітки всього контенту веб-сторінок.
* **CSS (Cascading Style Sheets):** Для стилізації візуальних елементів, забезпечення дизайну, кольорової гами, шрифтів та адаптивного відображення на різних розмірах екрану.
* **JavaScript:** Використовується для реалізації інтерактивних елементів інтерфейсу, таких як навігаційні меню, слайдери, анімації, а також для обробки подій на сторінці (наприклад, перевірка форм, якщо такі є, або завантаження додаткового контенту без перезавантаження сторінки).

2. **Бекенд (Server-Side) та База даних (Database):**

- Оскільки веб-сайт є переважно статичним інформаційним порталом і не передбачає обробки складних запитів користувачів, управління обліковими записами чи зберігання динамічних даних, окремий бекенд та база даних відсутні. Вся інформація надається у вигляді статичних HTML-сторінок, CSS-стилів та JavaScript-скриптів. Це значно спрощує розгортання, підвищує швидкість завантаження сторінок та знижує вимоги до серверних ресурсів.

3. **Веб-сервер (Web Server):**

* **Роль:** Веб-сервер відповідає за обслуговування HTTP-запитів від клієнтських браузерів та доставку статичних файлів (HTML, CSS, JavaScript, зображення, архіви додатків) до користувачів. Він є проміжною ланкою між файлами веб-сайту та кінцевим користувачем.
* **Технології:** Для хостингу такого типу веб-сайту можуть використовуватися стандартні веб-сервери, такі як Nginx або Apache, або спеціалізовані платформи для хостингу статичних сайтів (наприклад, GitHub Pages, Netlify, Vercel), які забезпечують високу швидкість та

надійність доставки контенту.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

Взаємодія компонентів:

Взаємодія компонентів веб-сайту є простою та лінійною. Користувач через свій веб-браузер надсилає HTTP-запит до веб-сервера. Веб-сервер отримує цей запит і, оскільки сайт статичний, просто знаходить запитуваний файл (наприклад, index.html, style.css, script.js або файл дистрибуції .dmg/.ipa) на файловій системі сервера і відправляє його назад до браузера користувача. JavaScript на фронтенді може обробляти події та змінювати вміст сторінки без додаткових запитів до сервера.

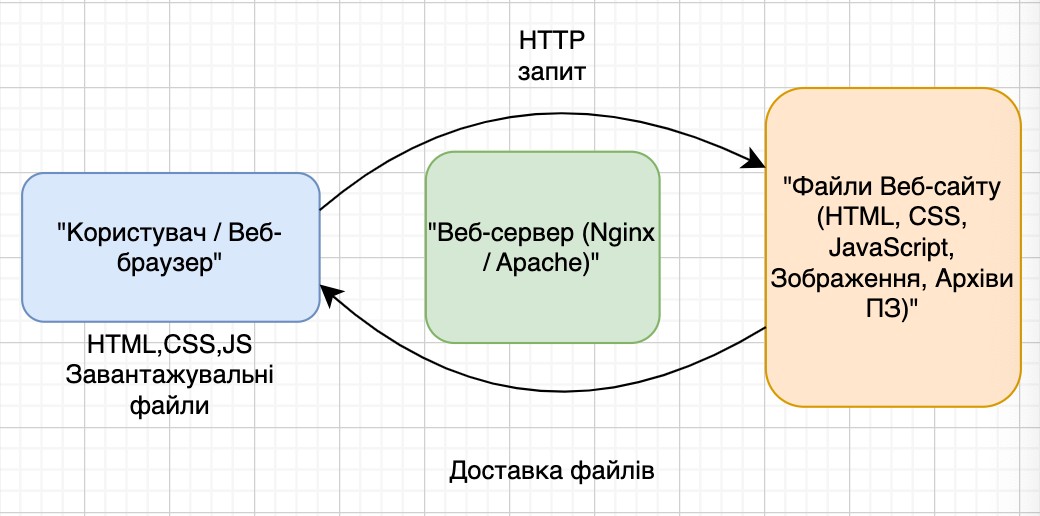


Рисунок 3.11 – Архітектура Веб-сайту.

**3.3.2 Проектування інтерфейсу користувача Веб-сайту**

Інтерфейс веб-сайту розроблений з акцентом на простоту, інтуїтивність, чистий дизайн та високу адаптивність (Responsive Design), щоб забезпечити оптимальний та приємний досвід користувача на будь-якому пристрої – від десктопних комп'ютерів до смартфонів. Дизайн веб-сайту повністю відповідає його призначенню як інформаційного порталу та платформи дистрибуції програмного забезпечення, надаючи користувачеві всю необхідну інформацію швидко та доступно.

**Основні розділи та сторінки:**

Веб-сайт структурований на кілька ключових розділів, кожен з яких виконує свою специфічну функцію:

* **Домашня сторінка (Landing Page):** Це перша сторінка, яку бачить користувач. Вона містить короткий, але змістовний вступ про програмний комплекс, його основну ідею та цінність. На домашній сторінці розміщені виразні кнопки заклику до дії (Call-to-Action), такі як "Завантажити для macOS" та "Завантажити для iOS", які дозволяють користувачеві швидко перейти до отримання додатків. Також можуть бути включені короткі переваги або ключові можливості продукту.
* **Сторінка "Продукт / Функціонал":** Цей розділ присвячений детальному опису функціональних можливостей як macOS, так і iOS додатків. Тут представлені докладні характеристики, пояснення принципу роботи, а також візуальний контент – скріншоти інтерфейсів додатків та, можливо, короткі демонстраційні відео, що ілюструють ключові особливості.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

* **Сторінка "Завантаження / Дистрибуція":** Спеціалізована сторінка для отримання програмного забезпечення. Для macOS користувач може завантажити файл дистрибуції у форматі .dmg безпосередньо з сайту. Для iOS додаток, враховуючи особливості дистрибуції мобільних додатків Apple, надаються прямі посилання на App Store. Це забезпечує безпечний та зручний спосіб отримання програмного забезпечення.
* **Сторінка "Довідка / FAQ / Підтримка":** Цей розділ містить відповіді на часто задавані питання (FAQ), докладні інструкції з встановлення та використання додатків, а також посилання на ресурси підтримки або форму зворотного зв'язку. Метою є надати користувачеві вичерпну допомогу та вирішити типові проблеми.
* Сторінка "Контакти": Містить інформацію для зв'язку з розробниками або службою підтримки, таку як електронна пошта, форми зворотного зв'язку або посилання на соціальні мережі.

**Елементи навігації:**

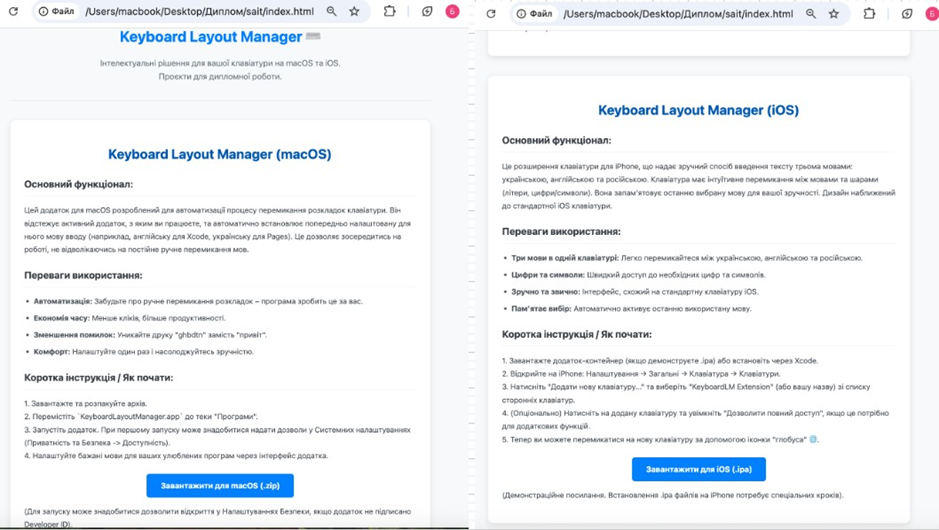
Навігація по веб-сайту спроектована таким чином, щоб бути максимально простою та інтуїтивною.

* **Головне меню:** Розташоване у верхній частині сторінки, воно надає швидкий доступ до всіх основних розділів сайту. На мобільних пристроях головне меню трансформується у бургер-меню для зручності використання на невеликих екранах.
* **Кнопки заклику до дії (Call-to-Action buttons):** Стратегічно розміщені на ключових сторінках, особливо на домашній та сторінці продукту, щоб заохотити користувача до завантаження додатків або отримання додаткової інформації ("Завантажити додаток", "Дізнатися більше", "Перейти до App Store").

**Дизайн та візуальні елементи:**

* **Кольорова схема та типографія:** Дизайн веб-сайту виконаний у мінімалістичному стилі з чистою та приємною кольоровою схемою, що забезпечує легку читабельність тексту та візуально не перевантажує користувача. Використано сучасні, легко читабельні шрифти, що відповідають загальній естетиці.

**Використання іконок, зображень та скріншотів:** Візуальний контент відіграє ключову роль у представленні продукту. Ефективне використання іконок допомагає швидко ідентифікувати функції, високоякісні зображення створюють приємне враження, а реальні скріншоти додатків демонструють їхній інтерфейс та функціонал у дії.



арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

Рисунок 3.11 – Інтерфейс користувача Веб-сайту (Домашня сторінка).

**Висновок за розділом 3**

У даному розділі було детально розглянуто процеси проектування та реалізації програмного комплексу, що включає десктопний додаток для macOS, мобільний додаток для iOS у форматі Custom Keyboard Extension, та супутній інформаційний веб-сайт. Кожен компонент було розроблено з урахуванням специфіки платформи та вимог до кінцевого функціоналу.

Для macOS додатку було розроблено архітектуру, засновану на тісній взаємодії з системними сервісами, зокрема Accessibility API та Carbon Framework (Text Input Services). Це дозволило реалізувати основний функціонал автоматичного перемикання розкладки клавіатури залежно від активного додатку, забезпечуючи глибоку інтеграцію в операційну систему. Розробка велася з використанням Swift та SwiftUI/AppKit, що гарантувало нативну продуктивність та відповідність стандартам macOS.

Мобільний iOS додаток представлений як інтегрована система з контейнерного додатку та розширення клавіатури (Custom Keyboard Extension). Такий підхід забезпечив користувачам можливість автоматичного налаштування візуальної розкладки клавіатури безпосередньо в будь-якому додатку iOS, вирішуючи при цьому виклики, пов'язані з обмеженнями "пісочниці" мобільної платформи. Взаємодія між контейнером та розширенням реалізована за допомогою Shared Defaults та App Groups, а в якості ключових технологій використано Swift, UIKit та SwiftUI.

Веб-сайт, реалізований на основі HTML, CSS та JavaScript, слугує центральним інформаційним порталом та зручним засобом для дистрибуції програмного забезпечення. Його статична архітектура спрощує розгортання та підтримку, а адаптивний дизайн забезпечує оптимальне відображення на різноманітних пристроях.

Загалом, інтерфейс користувача всіх компонентів програмного комплексу розроблений з акцентом на простоту, інтуїтивність та відповідність Apple Human Interface Guidelines та загальноприйнятим веб-стандартам, що забезпечує високу зручність взаємодії на всіх платформах.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.03.ПТРПЗ

Таким чином, розроблений програмний комплекс є повноцінним багатоплатформним рішенням, яке успішно реалізує поставлені задачі з автоматизації керування розкладками клавіатури, забезпечуючи високу продуктивність, функціональність та зручність для кінцевого користувача.

**Висновок**

Змн

Арк.

№\_Документа№ документа

Підпис

Дата

Аркуш

КНУ.РБ.123.25.14.04.В

КНУ.ПК.123.22.01.Р

Розробив

Каменев

Перевірив

Кузнєцов

Н.контроль

Кузнєцов

Затвердив

Купін

Висновок

Літера

Аркушів

КІ-21

Кваліфікаційна робота присвячена розробці багатоплатформного програмного комплексу, спрямованого на автоматизацію процесу керування розкладками клавіатури в операційних системах macOS та iOS, а також на інформаційну підтримку користувачів через веб-сайт. Актуальність теми зумовлена необхідністю підвищення продуктивності та зручності користувачів, які працюють з декількома мовами та активно перемикаються між різними додатками, оскільки нативні засоби macOS та iOS не забезпечують контекстного автоматичного перемикання розкладок клавіатури.

Метою роботи було створення інтегрованого рішення, здатного автоматично адаптувати розкладку клавіатури до контексту активного додатку на різних платформах. Для досягнення цієї мети було проведено глибокий аналіз сучасних технологій, фреймворків та інструментів розробки, зокрема середовища Xcode, мови програмування Swift, фреймворків SwiftUI та UIKit, а також специфічних системних API Apple. Детально досліджено особливості використання Accessibility API та Carbon Framework (Text Input Services) для macOS, а також механізми Custom Keyboard Extension і App Groups для iOS. Вибір нативного підходу був обґрунтований необхідністю забезпечення глибокої системної інтеграції та оптимальної продуктивності.

У результаті виконаної роботи було розроблено та реалізовано:

* Десктопний додаток для macOS, який ефективно керує системною розкладкою клавіатури на основі активного вікна. Це досягнуто завдяки використанню механізмів Accessibility API для моніторингу активних додатків та Carbon Framework (Text Input Services) для програмного перемикання розкладок. Інтерфейс додатку, виконаний у стилі AppKit/SwiftUI, забезпечує зручне налаштування правил перемикання та моніторинг активності.
* Мобільний додаток для iOS, представлений як інтегрована система з контейнерної програми та розширення Custom Keyboard Extension. Це рішення надає користувачам віртуальну клавіатуру, що адаптує свою візуальну розкладку до хост-додатку, долаючи обмеження мобільної "пісочниці" за допомогою Shared Defaults для обміну налаштуваннями. Інтерфейс додатку та клавіатури відповідає Apple Human Interface Guidelines, гарантуючи інтуїтивне використання.

Розробка здійснена з використанням Swift, UIKit та SwiftUI.

* Веб-сайт, розроблений з використанням HTML, CSS та JavaScript, функціонує як інформаційний портал та платформа для дистрибуції

програмного забезпечення. Його статична архітектура забезпечує швидке завантаження та легкість доступу до інформації, а адаптивний дизайн – комфортне використання на будь-яких пристроях.

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.04.В

Загальний дизайн інтерфейсу користувача всіх компонентів програмного комплексу виконаний з акцентом на простоту, інтуїтивність та відповідність стандартам платформ, що забезпечує високу зручність взаємодії.

Таким чином, поставлену мету роботи було повністю досягнуто. Розроблений програмний комплекс є функціональним, надійним та зручним у використанні багатоплатформним рішенням, що значно покращує досвід користувачів, які працюють з кількома мовами, автоматизуючи рутинний процес перемикання розкладок клавіатури. Його нативні реалізації на macOS та iOS, а також супутній інформаційний вебсайт, створюють цілісну екосистему для автоматизованого управління розкладками клавіатури.

Перспективи подальшого розвитку проекту включають розширення функціоналу автоматичного визначення мови вводу, інтеграцію з хмарними сервісами для синхронізації налаштувань між пристроями користувача, та подальше вдосконалення інтерфейсу для ще більшої зручності.

**Список використаних джерел**

Змн

Арк.

№\_Документа№ документа

Підпис

Дата

Аркуш

КНУ.РБ.123.25.14.05.СВД

КНУ.ПК.123.22.01.Р

Розробив

Каменев

Перевірив

Кузнєцов

Н.контроль

Кузнєцов

Затвердив

Купін

Список використаних джерел

Літера

Аркушів

КІ-21

[1] Офіційна документація Apple Developer для macOS: Це основний ресурс для розробників під macOS, що містить API-довідники, гайди та приклади коду.

<https://developer.apple.com/documentation/macos>

[2] Принципи дизайну інтерфейсу Apple Human Interface Guidelines (macOS): Надає рекомендації щодо створення інтуїтивно зрозумілих, функціональних та естетично привабливих додатків, що відповідають екосистемі Apple.

<https://developer.apple.com/design/human-interface-guidelines/macos>

[3] Офіційна документація Apple Developer для iOS: Центральний ресурс для розробників мобільних додатків на iOS, що містить детальні APIдовідники та гайди.

<https://developer.apple.com/documentation/ios>

[4] Принципи дизайну інтерфейсу Apple Human Interface Guidelines (iOS): Надає основні правила та рекомендації для розробки додатків, які відповідають стандартам Apple щодо користувацького досвіду та візуального оформлення.

<https://developer.apple.com/design/human-interface-guidelines/ios>

[5] Розробка кастомних клавіатур в iOS (Custom Keyboards in iOS): Це пряме посилання на розділ документації Apple, який описує, як створювати власні розширення клавіатури, що є критично важливим для реалізації функціоналу управління розкладками на мобільній платформі.

[https://developer.apple.com/documentation/uikit/keyboards\_and\_input/creating](https://developer.apple.com/documentation/uikit/keyboards_and_input/creating_a_custom_keyboard)

[\_a\_custom\_keyboard](https://developer.apple.com/documentation/uikit/keyboards_and_input/creating_a_custom_keyboard)

[6] Mozilla MDN Web Docs - Web Technologies (загальний огляд): Це комплексний ресурс, що надає вичерпну інформацію про всі основні веб-технології (HTML, CSS, JavaScript) та принципи їх роботи.

<https://developer.mozilla.org/en-US/docs/Web>

[7] HTML (Mozilla MDN):

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.05.СВД

<https://developer.mozilla.org/en-US/docs/Web/HTML>

[8] CSS (Mozilla MDN): <https://developer.mozilla.com/en-US/docs/Web/CSS>

[9] JavaScript (Mozilla MDN): <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

[10] AppKit - NSWorkspace (Apple Developer): Документація по NSWorkspace, що включає метод frontmostApplication для відстеження активних програм.

<https://developer.apple.com/documentation/appkit/nsworkspace>

[11] Accessibility (Apple Developer): Загальний огляд Accessibility API та його можливостей для взаємодії з елементами інтерфейсу.

<https://developer.apple.com/documentation/appkit/nsaccessibility>

[12] Creating a Custom Keyboard (Apple Developer Documentation): Це ключовий та найбільш релевантний ресурс з офіційної документації Apple, що надає вичерпну інформацію про архітектуру, розробку та життєвий цикл кастомних клавіатур на iOS.

[https://developer.apple.com/documentation/uikit/keyboards\_and\_input/creating](https://developer.apple.com/documentation/uikit/keyboards_and_input/creating_a_custom_keyboard)

[\_a\_custom\_keyboard](https://developer.apple.com/documentation/uikit/keyboards_and_input/creating_a_custom_keyboard)

[13] Input Method (Wikipedia): Загальний огляд концепції Input Method Editor (IME), що пояснює принципи роботи систем введення тексту, включаючи віртуальні клавіатури.

<https://en.wikipedia.org/wiki/Input_method>

[14] Офіційний сайт Karabiner-Elements: <https://karabiner-elements.app/>

[15] Офіційний сайт Ukelele:

<https://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&item_id=ukelele>

[16] Посилання (App Store): [https://apps.apple.com/us/app/gboard-thegoogle-keyboard/id1091700242](https://apps.apple.com/us/app/gboard-the-google-keyboard/id1091700242)

[17] Посилання (App Store): [https://apps.apple.com/us/app/swiftkeykeyboard/id917798363](https://apps.apple.com/us/app/swiftkey-keyboard/id917798363)

[18] Xcode (Apple Developer): Офіційна сторінка Xcode на сайті Apple Developer, що містить посилання на завантаження, документацію та останні новини. <https://developer.apple.com/xcode/>

[19] Офіційний сайт Swift: Головний ресурс для вивчення мови Swift, що містить документацію, посібники та останні новини про розвиток мови. <https://www.swift.org/>

[20] Visual Studio Code: Офіційний сайт редактора, де можна завантажити програму, знайти документацію та інформацію про розширення. <https://code.visualstudio.com/>

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.05.СВД

[21] AppKit (Apple Developer Documentation): Детальна довідка по класу

NSApplication та іншим компонентам фреймворку.

<https://developer.apple.com/documentation/appkit>

[22] Accessibility (Apple Developer Documentation): Загальний опис принципів та класів Accessibility API.

<https://developer.apple.com/documentation/appkit/nsaccessibility>

[23] NSWorkspace (Apple Developer Documentation): Документація класу NSWorkspace, що містить властивість frontmostApplication для визначення активної програми.

<https://developer.apple.com/documentation/appkit/nsworkspace>

[24] Foundation (Apple Developer Documentation): Комплексна довідка по класах та функціям фреймворку Foundation.

<https://developer.apple.com/documentation/foundation>

[25] SwiftUI (Apple Developer Documentation): Офіційна документація, що описує можливості та принципи роботи SwiftUI.

<https://developer.apple.com/documentation/swiftui>

[26] UIKit (Apple Developer Documentation): Комплексна довідка по фреймворку UIKit. <https://developer.apple.com/documentation/uikit>

[27] Foundation (Apple Developer Documentation): Детальна інформація про фреймворк Foundation.

<https://developer.apple.com/documentation/foundation>

[28] HTML (Mozilla MDN Web Docs): Вичерпна довідка по HTMLелементам та їх використанню. [https://developer.mozilla.org/enUS/docs/Web/HTML](https://developer.mozilla.org/en-US/docs/Web/HTML)

[29] CSS (Mozilla MDN Web Docs): Детальна інформація про властивості

CSS та їх застосування. [https://developer.mozilla.org/en-](https://developer.mozilla.org/en-US/docs/Web/CSS)

[US/docs/Web/CSS](https://developer.mozilla.org/en-US/docs/Web/CSS)

[30] JavaScript (Mozilla MDN Web Docs): Довідник та посібники з мови JavaScript. <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

[31] React: <https://react.dev/>

[32] Vue.js: <https://vuejs.org/>

[33] Angular: <https://angular.io/>

[34] Node.js: <https://nodejs.org/>

[35] Express.js: <https://expressjs.com/>

арк

№ документа

Підпис

Дата

арк

КНУ.РБ.123.25.14.05.СВД

[36] Swift Package Manager (Офіційний сайт): [https://swift.org/packagemanager/](https://swift.org/package-manager/)

[37] CocoaPods (Офіційний сайт): <https://cocoapods.org/>[38] Git (Офіційний сайт): <https://git-scm.com/>

[39] npm (Офіційний сайт): <https://www.npmjs.com/>

[40] Yarn (Офіційний сайт): <https://yarnpkg.com/>

[41] Git (Офіційний сайт): <https://git-scm.com/>

[42] Apple Inc. The Swift Programming Language. Cupertino : Apple Books, 2024. URL: https://docs.swift.org/swift-book/ (дата звернення: 01.06.2025).

[43] Apple Developer. Accessibility API. URL: https://developer.apple.com/documentation/accessibility/ (дата звернення: 03.06.2025).

[44] SwiftUI Documentation. Apple Developer Documentation. URL: <https://developer.apple.com/documentation/swiftui>

[45] Markdown Guide. Markdown.md. URL: https://www.markdownguide.org/ (дата звернення: 01.06.2025).

[46] Apple Inc. The Swift Programming Language. Cupertino : Apple Books, 2024. URL: https://docs.swift.org/swift-book/ (дата звернення: 01.06.2025).

[47] Apple Developer. Accessibility API. URL: https://developer.apple.com/documentation/accessibility/ (дата звернення: 03.06.2025).

[48] SwiftUI Documentation. Apple Developer Documentation. URL: https://developer.apple.com/documentation/swiftui (дата звернення: 02.06.2025).

[49] Markdown Guide. Markdown.md. URL: https://www.markdownguide.org/ (дата звернення: 01.06.2025).