

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Факультет комп'ютерних наук та технологій
Кафедра комп'ютерних систем та мереж

Пояснювальна записка

до дипломного проекту (роботи)

магістра

(ступінь вищої освіти)

на тему ВДОСКОНАЛЕНА СИСТЕМА ПЛАНУВАННЯ ЧАСУ

Виконав: студент 2 курсу, групи КНТ-512м
спеціальності _____

123 Комп'ютерна інженерія

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні системи та мережі

ІВАНОВ А. І.

(ПРИЗВИЩЕ та ініціали)

Керівник ТЯГУНОВА М.Ю

(ПРИЗВИЩЕ та ініціали)

Рецензент САМОЙЛИК С.С

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет Комп'ютерних наук і технологій
Кафедра «Комп'ютерні системи та мережі»
Ступінь вищої освіти магістерський
Спеціальність 123 Комп'ютерна інженерія
(код і найменування)
Освітня програма (спеціалізація) «Комп'ютерні системи та мережі»
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Зав. кафедри Кудерметов Р.К.

« » 2023 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

ІВАНОВА Андрія Ігоровича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Вдосконалена система планування часу

керівник проєкту (роботи) к. т. н., доцент, Тягунова Марія Юріївна

(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від «24» жовтня 2023 року № 400

2. Строк подання студентом проєкту (роботи) 10 грудня 2023 року

3. Вихідні дані до проєкту (роботи) додаток «Trello», додаток «Fibery», iOS-додаток «Календар», iOS-додаток «Нагадування»

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

1) Аналіз сучасних систем планування часу;

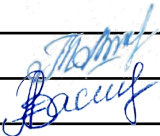
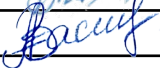
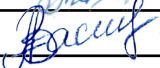
2) Проєкутвання системи;

3) Реалізація системи;

4) Результати роботи додатку;

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	ТЯГУНОВА М.Ю., доцент		
нормоконтроль	ПОЛЬСЬКА О.В., ст. викл.		

7. Дата видачі завдання «01» жовтня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

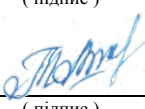
№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз сучасних систем планування часу	15.10.2023 р.	
2	Проекутвання системи	01.11.2023 р.	
3	Реалізація системи	25.11.2023 р.	
4	Оформлення отриманих результатів у ПЗ	01.12.2023 р.	

Студент


(підпис)

Андрій ІВАНОВ
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)


(підпис)

Марія ТЯГУНОВА
(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

ПЗ: 78 с., 36 рис., 1 табл., 30 джерел.

СИСТЕМА ПЛАНУВАННЯ ЧАСУ, ВДОСКОНАЛЕННЯ, ЗАВДАННЯ,
СПИСОК, ПОМІЧНИК, ДОДАТОК, iOS, SWIFT, GIT

Об'єкт дослідження – системи планування часу.

Предмет дослідження – шляхи вдосконалення систем планування часу.

Мета роботи – розробити систему, яка допоможе людині підвищити продуктивність своєї праці та збільшити час на відпочинок.

У першому розділі розглянуті існуючі системи планування часу, виявлено недоліки існуючих систем, виявлено вимоги для створення вдосконаленої системи планування часу.

У другому розділі проведено проектування системи, виявлено сутності, екрани та підсистеми, обрано архітектуру реалізації системи.

У третьому розділі наведено реалізацію спроектованої системи, яка розроблена за допомогою засобів розробки: середовища розробки «Xcode», мови програмування «Swift», системи контролю версій «Git», фреймворку «Foundation.framework», фреймворку «UIKit.framework», фреймворку «Combine.framework», фреймворку «XCTest», бібліотеки «Dependencies», бібліотеки «CombineCocoa», бібліотеки «SnapKit», «Realm» SDK.

У четвертому розділі наведено результати роботи системи.

В результаті виконання роботи досліджено існуючі системи планування часу та розроблено вдосконалену систему планування часу, що надає можливість оптимально планувати час та виконувати заплановані завдання.

ABSTRACT

Explanatory note to the master's work: 78 p., 36 figures, 1 tables, 30 sources.

TIME PLANNING SYSTEM, IMPROVEMENT, TASK, LIST, ASSISTANT, APPLICATION, iOS, SWIFT, GIT

Object research – time planning systems.

The subject of research is ways to improve the time planning systems.

The purpose of the work is to develop a system that will help a person increase productivity at work and allocate more time for relaxation.

The first section examines existing time planning systems, identifies shortcomings in existing systems, and establishes requirements for creating an improved time planning system.

In the second section, the system is designed, entities, screens, and subsystems are identified, and the implementation architecture of the system is selected.

In the third section, the implementation of the designed system is provided, which was developed using the following development tools: Xcode development environment, Swift programming language, Git version control system, the Foundation.framework, UIKit.framework, Combine.framework, XCTest.framework, «Dependencies» library, «CombineCocoa» library, «SnapKit» library, and «Realm SDK».

The fourth section presents the results of the system's operation.

As a result of the work, existing time planning systems were researched, and an improved time planning system has been developed, providing the ability to optimally plan time and execute planned tasks.

ЗМІСТ

Вступ.....	7
1 Аналіз сучасних систем планування часу.....	9
1.1 iOS-додаток «Календар».....	9
1.2 iOS-додаток «Нагадування».....	11
1.3 Dodatok Trello.....	12
1.4 Dodatok Fibery.....	14
1.6 Порівняння розглянутих систем.....	16
1.7 Аналіз отриманих результатів.....	17
2 Проектування системи.....	19
2.1 Проектування сутностей системи.....	19
2.2 Проектування екранів додатку.....	24
2.3 Проектування підсистем додатку.....	41
2.4 Проектування архітектури додатку.....	44
2.5 Висновки до розділу.....	46
3 Реалізація системи.....	46
3.1 Засоби розробки.....	47
3.2 Розробка додатку.....	53
3.3 Висновки до розділу.....	60
4 Результати роботи додатку.....	60
Висновки.....	74
Перелік джерел посилання.....	76

ВСТУП

В сучасному світі людина стикається з надмірною кількістю справ та обов'язків, які потрібно виконати, і вирішальним аспектом стає здатність грамотно керувати своїм часом. Під впливом зростаючої конкуренції ті, хто майстерно розподіляє свій час, стають більш успішними. Однак для досягнення ефективності використання часу не менш важливим є навичка знаходження часу на відпочинок, родину та інші аспекти життя. Таким чином, тема планування часу залишається актуальною для всіх, хто прагне знайти гармонію в своєму розпорядженні часом за допомогою системи, що ефективно сприяє цьому.

Перед усім, важливо розглянути ключові принципи, на яких базуються системи планування часу. Це включає в себе визначення мети та цілей, пріоритизацію завдань, розподіл часу з урахуванням термінів виконання та визначення проміжків для відпочинку та розваг. Ефективне планування часу вимагає не лише відданості виконанню завдань, але й гнучкості в реагуванні на непередбачувані обставини.

Однією з головних переваг використання систем планування є здатність створювати відчуття контролю над власним часом. Коли кожна дія має своє місце у плані, людина може більш ефективно використовувати свій потенціал та уникати стресу від невиконаних завдань. Планування також сприяє досягненню більш високого рівня продуктивності та визначенню пріоритетів в житті.

Незважаючи на безліч переваг, системи планування можуть стикатися з викликами. Один із найважливіших аспектів - це адаптація до змін у розкладі та пріоритетах. Під час непередбачених обставин, важливо мати гнучкість та вміння переглядати план, не втрачаючи ефективність. Також, важливо уникати надмірного завантаження, оскільки це може призвести до стресу та вигорання.

Системи планування часу стають необхідними в сучасному суспільстві, де конкуренція та темпи життя швидко зростають. Вони допомагають не лише досягати особистих та професійних цілей, а й створювати баланс між роботою,

відпочинком та сімейним життям. Застосування систем планування може вплинути на якість життя, забезпечуючи більшу свободу та контроль над власним часом.

Системи планування часу є важливим інструментом у сучасному світі. Вони не тільки допомагають управляти завданнями та досягати цілей, але й впливають на загальну якість життя, роблячи його більш збалансованим та задовільним. Важливо враховувати виклики, які можуть виникнути, та розвивати гнучкість у використанні систем планування, щоб досягти максимальної ефективності та задоволення від власного часу.

В сучасному світі існує велика кількість систем планування часу, які мають різноманітний функціонал, але врешті решт буває складно знайти систему, яка буде зручною і легкою у використанні. Кожна система має своє переваги та недоліки. Отже, обрана тема магістерської роботи є актуальною.

Мета роботи – розробити систему, яка допоможе людині підвищити продуктивність своєї праці та збільшити час на відпочинок.

Для досягнення поставленої мети необхідно виконати наступні завдання:

- провести аналіз найбільш популярних систем планування часу, щоб визначити їхні переваги та недоліки;
- враховуючи переваги та недоліки розглянутих систем, виявити вимоги для створення вдосконаленої системи планування часу;
- спроектувати структуру та алгоритм роботи системи згідно висунутих вимог;
- обрати засоби розробки системи;
- реалізувати систему планування часу;
- проаналізувати отримані результати.

1 АНАЛІЗ СУЧАСНИХ СИСТЕМ ПЛАНУВАННЯ ЧАСУ

На сьогоднішній день існує велика кількість різноманітних систем для планування часу та завдань. Кожна з них має свої особливості та недоліки. Всі вони відрізняються за областю використання. Одні більш підходять для планування власного розкладу, інші більш підходять для роботи і організації робочих завдань. Не зважаючи на велику кількість таких систем користувачу може бути складно підібрати систему, яка задовольнить усім його потребам [1, 2].

Для того щоб виявити певні тенденції, якою має бути сучасна система та виявити найбільш суттєві недоліки існуючих систем було вирішено взяти до аналізу одні із найбільш популярних систем планування часу та завдань. Одним із критеріїв при обранні систем було обрати такі системи, щоб одна від одною відрізнялися функціоналом та областю використання.

До аналізу було обрано наступні системи:

- iOS-додаток «Календар»;
- iOS-додаток «Нагадування»;
- Trello;
- Fibery.

1.1 iOS-додаток «Календар»

Додаток «Календар» (або англ. Calendar) [3] від Apple є вбудованим інструментом для організації та відстеження подій та зустрічей. Цей додаток допомагає користувачам керувати їхнім часом, планувати події та координувати розклад.

Розглянемо основні особливості цієї системи. Додаток «Календар» дозволяє користувачам створювати різні календарі для різних аспектів життя (робота,

особисте, важливі події тощо), що сприяє зручному управлінню розкладом. Користувачі у ньому можуть додавати події до календарів, встановлювати дати, час, тривалість, місце та інші деталі. Він надає можливість встановлювати нагадування про майбутні події, що допомагає не пропустити важливі дати та зустрічі. Додаток дозволяє надсилати запрошення на події та до календарів іншим користувачам, використовуючи можливості електронної пошти. Також користувачі можуть відображати події у різних режимах перегляду, таких як день, тиждень, місяць, дозволяючи швидко зорієнтуватися у своєму розкладі. «Календар» синхронізується з iCloud та іншими платформами, що дозволяє користувачам отримувати доступ до свого розкладу на різних пристроях [4].

Але у цьому додатку також можна виокремити і наступні недоліки. «Календар» в першу чергу призначений для керування подіями, і налаштування завдань обмежені. Він не надає багато опцій для організації та призначення завдань, порівняно зі спеціалізованими додатками для завдань [5]. Складно організовувати багато завдань. Для користувачів, які мають багато завдань або робочих завдань, управління ними в «Календарі» може бути незручним, оскільки відсутні зручні категорії або перегляди для завдань. Немає можливості створювати під завдання що може бути корисним для роботи с довгостроковим та складними задачами. Не така розширена автоматизація. «Календар» має базову систему нагадувань та повідомлень, але вона не така розширена та гнучка, як у деяких інших системах, що дозволяють створювати складні автоматизовані дії. Відсутність деяких спеціалізованих функцій. Додаток може виявитися обмеженим для специфічних завдань або професійних потреб. Наприклад, він може не надавати деякі аналітичні функції, які потрібні бізнес-користувачам. Обмежена можливість інтеграції з іншими системами. Інтеграція «Календаря» з іншими додатками та платформами обмежена порівняно з іншими календарними додатками, що може бути не так зручно для користувачів, які користуються багатьма іншими сервісами. Відсутність певних просунутих функцій збереження приватності. Додаток не надає деякі просунуті можливості контролю над

конфіденційністю та приватністю даних, які можуть бути важливі для деяких користувачів.

Додаток «Календар» є потужним та зручним інструментом для організації часу та подій. З його допомогою користувачі можуть планувати розклад, встановлювати нагадування, організовувати зустрічі та події. Він інтегрований з екосистемою Apple та синхронізується з іншими пристроями. Але додаток не підходить для планування та контролю більш складних завдань. Функціональності системи не вистачає для зручної та швидкої роботи з великою кількістю завдань.

1.2 iOS-додаток «Нагадування»

Додаток «Нагадування» (або англ. Reminders) [6] від Apple є універсальним інструментом для організації та відстеження завдань та нагадувань. Цей додаток допомагає користувачам легко створювати, управляти та відстежувати нагадування, щоб забезпечити вчасне виконання завдань.

Розглянемо основні особливості цієї системи [7]. Додаток «Нагадування» дозволяє користувачам швидко створювати нагадування з визначенням тексту, дати, часу та пріоритету. Користувачі у ньому можуть встановлювати нагадування на конкретні дати та часи, або вказувати повторення (щоденно, щотижня, щомісяця), що дозволяє гнучко планувати нагадування. Додаток дозволяє встановлювати пріоритети для завдань та додавати мітки для категоризації нагадувань. Також користувачі можуть організовувати нагадування у різні списки (наприклад, робочі завдання, особисті нагадування, покупки тощо), що допомагає зберігати порядок. Функція «Розумні списки» автоматично організовує і відображає майбутні нагадування за певними налаштуваннями (По тегам, даті, часу та іншим налаштуванням). «Нагадування» надає можливість створювати шаблони списків що дозволяє швидко створювати нові списки. Додаток автоматично синхронізується через iCloud між різними пристроями користувача, забезпечуючи

зручний доступ до завдань. Підтримується можливість додавати у списки інших користувачів що дозволяє редагувати список декільком користувачам. Додаток підтримує нагадування з використанням звукових оповіщень та сповіщень на екрані, що допомагає не пропустити важливі завдання.

Але у цьому додатку також можна виокремити і наступні недоліки. «Нагадування» в першу чергу призначений переважно для простих завдань і нагадувань. Він не має розширених можливостей, які зазвичай пропонують спеціалізовані додатки для управління завданнями та проектами. Він не надає зручного інструмента для організації завдань за проектами або категоріями. Це може бути незручним для користувачів, які мають багато різних завдань та проектів. Хоча додаток дозволяє призначати завданням дати, часи та пріоритети, він не надає багато інших параметрів налаштування, які можуть бути важливі для складних завдань. «Нагадування» не має високорівневої системи автоматизації, що робить його менш придатним для автоматизації рутинних завдань із складними логіками. Інтеграція «Нагадувань» з іншими додатками та сервісами обмежена порівняно з іншими системами управління завданнями, що може бути проблемою для користувачів, які розраховують на високий рівень інтеграції. «Нагадування» не надає деталізованих аналітичних інструментів для відстеження продуктивності або аналізу виконаних завдань.

Додаток «Нагадування» є добре підходящим для простих нагадувань і невеликих списків завдань, але може бути не достатнім для користувачів, які потребують розширеного функціоналу та організації складних завдань і проектів.

1.3 Додаток Trello

«Trello» [8] є популярним інструментом для управління проектами та організації завдань, створений для простоти та зручності в користуванні. Ця платформа надає користувачам можливість візуально відстежувати та

організовувати робочі завдання в реальному часі, сприяючи ефективній спільній роботі команди.

Розглянемо основні особливості цієї системи. Основною одиницею організації в «Trello» є карточки, які можна організовувати на дошках. Це дає користувачам можливість візуально подавати та категоризувати завдання. Інтерфейс «Trello» є інтуїтивним і легким у використанні, що робить його доступним для користувачів будь-якого рівня технічної підготовки. Додавання міток до карток дозволяють виділити та класифікувати завдання залежно від їх стану, пріоритету або інших аспектів. Кожна карточка дозволяє користувачам залишати коментарі, прикріплювати файли та спільно працювати над завданнями, що підвищує зручність спільної роботи. «Trello» надає можливість встановлювати терміни виконання завдань та отримувати нагадування, що сприяє забезпеченню дотримання графіку. Платформа підтримує інтеграцію з різними зовнішніми додатками та інструментами, такими як Slack, Google Drive тощо, полегшуючи обмін інформацією. Також система дозволяє встановлювати на дошки, розширення які надають більше можливостей для керування завданнями та проектами. «Trello» має ряд функцій автоматизації, які дозволяють спростити та автоматизувати керування завданнями та робочими процесами. «Trello Templates» - це набір готових шаблонів для створення дошок у «Trello», які допомагають швидко розпочати роботу над конкретними типами проектів чи завдань. Шаблони розроблені для різних галузей та типів завдань, і вони можуть бути використані як основа для нових дошок.

Але в «Trello» також можна виокремити наступні і наступні недоліки. Для того щоб розкрити весь потенціал «Trello» потрібно приділити багато часу на вивчення додаткового матеріалу. З ростом обсягу і кількості завдань на дошці «Trello» може стати складно виокремлювати та організовувати інформацію. Це особливо помітно на великих проектах. «Trello» - це веб-сервіс, тому ви повинні мати доступ до Інтернету для роботи з ним. Відсутність Інтернет-з'єднання може призвести до неможливості доступу до вашої інформації.

«Trello» є ефективним інструментом для управління проєктами та спільної роботи, завдяки своєму інтуїтивному інтерфейсу та зручним функціональним можливостям. Візуальна природа платформи сприяє прозорості, а спеціальні функції дозволяють налаштувати процеси організації та спільної роботи відповідно до потреб користувачів. «Trello» виконує роль потужного помічника у плануванні, відстеженні та реалізації завдань, що робить його цінним інструментом для різних сфер та проєктів.

1.4 Додаток Fibery

«Fibery» [9] представляє собою передовий інструмент для організації проєктів та спільної роботи, створений з урахуванням потреб користувачів у гнучкості, налаштованості та ефективності. Цей інструмент дозволяє командам та організаціям створювати власні структури даних та спеціалізовані робочі процеси, підлаштовуючись під унікальні вимоги кожного проєкту.

Розглянемо основні особливості цієї системи. Система «Fibery» надає користувачам можливість створювати власні типи сутностей та визначати для них властивості і зв'язки. Це дозволяє створювати належні схеми для будь-якого типу проєкту, незалежно від його специфіки.

Призначення завдань, визначення пріоритетів, керування часовими рамками — все це можна налаштувати відповідно до внутрішніх робочих процесів команди чи проєкту. Функції спільної роботи, обміну повідомленнями та коментарями дозволяють ефективно спілкуватися в межах команди, зменшуючи можливість непорозумінь. «Fibery» надає інтерактивні засоби візуалізації, такі як графіки Gantt, Kanban-дошки та діаграми, що допомагають відстежувати прогрес та аналізувати дані.

Платформа підтримує інтеграцію з іншими інструментами, такими як Slack,, Jira та інші, що полегшує обмін даними та забезпечує цілісність робочого процесу.

Вбудовані можливості автоматизації дозволяють зменшити участь людини на виконання рутинних завдань. Надається можливість створювати різні правила за якими можуть заповнюватися/змінюватися поля в сутностях.

Вбудований помічник на основі штучного інтелекту допомагає опановувати можливості системи та прискорює роботу. Переписка з помічником є у вигляді чата, що надає можливість вносити уточнення щодо відповіді помічника.

Але в цій системі також можна виокремити і наступні недоліки. «Fibery» надає багато можливостей для налаштування, що може бути перевагою, але також може стати недоліком для користувачів, які шукають простий та швидкий спосіб керувати завданнями.

Потрібен час для вивчення та налаштування системи під свої потреби. Початкові налаштування «Fibery» та освоєння інтерфейсу може виявитися важким завданням для новачків.

Для користувачів, які не мають досвіду у роботі з подібними системами, це може бути неперешкоджуваним бар'єром. «Fibery» - це веб-сервіс, тому для доступу до нього потрібне постійне підключення до Інтернету. Відсутність Інтернет-з'єднання може призвести до обмеженого доступу до даних. У порівнянні з іншими подібними інструментами, «Fibery» має обмежену кількість інтеграцій з іншими додатками та сервісами. Не завжди інтуїтивний інтерфейс. Деякі користувачі можуть вважати інтерфейс «Fibery» не таким інтуїтивно зрозумілим, як вони б хотіли, що може вимагати додаткового часу для навчання та пристосування.

«Fibery» є революційним інструментом для управління проектами та спільної роботи, який дозволяє командам створювати індивідуалізовані структури даних і оптимізовані робочі процеси.

Його гнучкість та можливості для налаштування роблять його цінним активом для дослідників, професіоналів та організацій, що прагнуть досягти максимальної продуктивності та співпраці, але він може не відповідати всім потребам і вимогам користувачів, особливо тим, хто шукає більш простий інструмент або має обмежений бюджет.

1.6 Порівняння розглянутих систем

Вивчивши способи порівняння різних систем та критерії вибору, були визначені наступні критерії [10,11] для подальшого порівняння розглянутих систем.

Легкість використання. Цей критерій оцінює, наскільки інтуїтивним та зрозумілим є інтерфейс системи для нових користувачів. Оцінка може визначатися на шкалі від 1 до 10, де 1 означає дуже складний інтерфейс, а 10 - дуже простий та легкий для освоєння.

Функціональність. Цей критерій визначає рівень функціональності та можливостей системи, такі як керування завданнями, календар, нагадування, тощо. Оцінка може відображати кількість функцій та їхню корисність, наприклад, від 1 до 10.

Інтеграція з іншими системами. Цей критерій визначає, наскільки добре система може інтегруватися з іншими додатками та сервісами, такими як електронна пошта, календарі, зберігання в хмарі тощо. Оцінка може визначатися на шкалі від 1 до 10, де 1 вказує на відсутність інтеграції, а 10 - на високий рівень інтеграції.

Автоматизація. Цей критерій визначає, наскільки система дозволяє автоматизувати завдання та створювати робочі потоки. Оцінка може відображати рівень автоматизації, де 1 вказує на відсутність автоматизації, а 10 - на високий рівень автоматизації.

Для отримання об'єктивної оцінки розглянутих систем було проведено опитування [12] за результатами котрого була створена порівняльна таблиця 1.1

Таблиця 1.1 - Оцінка систем для керування завданнями та часом

Критерії	Calendar	Reminders	Trello	Fibery	Sorted
Легкість в освоєнні та користуванні	9	10	7	5	8

Продовження таблиці 1.1

Критерії	Calendar	Reminders	Trello	Fibery	Sorted
Гнучкість налаштування задач	7	8	9	10	9
Зручність користування	9	9	8	7	10
Додаткові можливості системи	6	7	10	10	9
Можливість інтеграції з іншими системами	5	6	9	8	7
Можливості автоматизації	5	6	7	10	8

1.7 Аналіз отриманих результатів

Дослідивши існуючі системи планування часу, було виявлено деякі недоліки. Розглянемо їх нижче.

Щоб почати ефективно користуватися системою планування часу та дізнатися про всі можливості системи, доводиться ознайомлюватися з додатковими матеріалами (інструкціями, статтями, шукати інформацію на сторонніх ресурсах, мати навички програмування). Це вказує на те, що «поріг входження» є високим.

Ще суттєвим недоліком є те, що з часом загальна кількість створених задач стає дуже великою. Кожен раз, коли ми створюємо список, ми намагаємось внести в нього усі задачі, важливі і не дуже, довгострокові і ті, які займають декілька хвилин на виконання. Таким чином, список стає невиконаним дуже швидко. Доволі часто виникає також ситуації, що задачі, в яких вказано час виконання, потребують багато додаткового часу, якщо щось пішло не за графіком, і тоді потрібно робити перепланування, коли ці задачі можна бути виконати. Коли список задач стає великим, ускладняється і розстановка пріоритетів. Може бути

десяток задач з високим пріоритетом, в результаті чого важко вирішити з чого почати.

Також часом доводиться користуватися декількома системами планування часу. Наприклад, календар для планування подій, які виконуються за часом, а в іншій системі зберігається список покупок. Це створює проблеми синхронізації даних між різними системами та підтримання актуальності інформації.

Ще суттєвим недоліками є те, що плануючи список задач важко оцінити, скільки дійсно вільного часу у день для того, щоб виконати заплановані задачі, а також важко планувати довгострокові задачі, які потребують систематичного підходу і виконання певної кількості проміжкових задач.

Враховуючи всі перелічені недоліки, було виявлено наступні вимоги для створення вдосконаленої системи планування часу:

Створити низький поріг входження. Система має бути інтуїтивно зрозумілою, щоб її могли опанувати як діти, так і літні люди. Система не має бути перевантажена інформацією, яку бачить користувач, та опціями по налаштуванню системи;

В систему має бути вбудований помічник, який враховує потреби користувача та допомагає опанувати необхідні можливості системи у короткий термін часу. Взаємодія з помічником має бути реалізована в форматі чата. Помічник може пропонувати налаштування для системи та зміни у запланованому графіку;

Система не повинна пропонувати/відображати дві події в один і той самий час. Має відображатися лише одна подія;

Система має бути здатною аналізувати пріоритетність зі усіх задач, які збережені у системі та має бути здатною пропонувати, які задачі і коли можуть бути взяті до роботи; Система має бути адаптивною та адаптуватись під кожного користувача. Це означає що система має аналізувати дії користувача, його активність та підлаштовуватись автоматично під користувача;

Система має пропонувати часові рамки. Наприклад, сон, робота, вільний час. В кожному проміжку часу можуть бути заплановані задачі до виконання;

Система має бути здатною працювати з кіберфізичними системами заданими яких може бути запропоновані коригування у графіку задач;

Система повинна мати можливість інтеграції з іншими системами. Наприклад, з календарем для того, щоб відображати в календарі спланований графік.

2 ПРОЄКТУВАННЯ СИСТЕМИ

Під час написання дипломного проєкту було прийнято рішення реалізувати систему як iOS-додаток. iOS-додаток - це програмне забезпечення, спеціально розроблене для використання на пристроях, що працюють під управлінням операційної системи iOS, розробленої компанією Apple. iOS - це мобільна операційна система, яка використовується на iPhone.

На базі проведеного у першому розділі аналізу було визначено наступні вимоги, яким додаток має відповідати [13, 14]:

- додаток має працювати самостійно без виходу в інтернет;
- додаток повинен зберігати всі створені дані на клієнтській стороні;
- додаток має бути спроектован таким чином, щоб користувач сконцентрував свою увагу на теперішньому часі;
- додаток має бути спроектований з урахуванням можливості масштабування, додавання нових можливостей і функціональності.

2.1 Проєктування сутностей системи

В ході проєктування системи планування часу були визначені наступні сутності, якими оперує система [15].

2.1.1 Користувач

Під користувачем будемо розуміти сутність, яка відображає дані людини, яка зареєстрована в системі.

Користувач складається з наступних даних:

- ідентифікатор;
- ім'я;
- прізвище;
- електронна пошта;
- перелік списків;
- перелік періодів;
- перелік завдань;
- перелік ярликів.

Ідентифікатор - рядкове значення, унікальний згенерований ключ - значення за яким відбувається звернення до сутності. Обов'язковий тип даних. Ім'я - рядкове значення, відображає ім'я користувача. Обов'язковий тип даних. Фамілія - Строкове значення, відображає фамілію користувача. Обов'язковий тип даних. Електронна пошта - рядкове значення, адреса електронної пошти за якої зареєстрований користувач. Обов'язковий тип даних. Перелік списків - відображає інформацію які списки належать до користувача. Перелік періодів - відображає інформацію які періоди належать до користувача. Перелік завдань. - відображає інформацію які завдання належать до користувача. Перелік ярликів. - відображає інформацію які ярлики належать до користувача.

2.1.2 Завдання

Під завданням будемо розуміти сутність, яка відображає те, що запланував зробити користувач. Ця сутність є основною, якою оперує користувач у системі.

Завдання складається з наступних даних:

- ідентифікатор;
- назва;
- статус завдання;
- нотатки;

- дата початку;
- тривалість;
- ярлики;
- важливе;
- пріоритет;
- список;
- період.
- користувач.

Ідентифікатор - рядкове значення, унікальний згенерований ключ-значення за яким відбувається звернення до сутності. Обов'язковий тип даних. Назва - рядкове значення, відображає що має бути зроблено. Обов'язковий тип даних. Статус завдання - логічне значення, відображає статус виконання завдання. Якщо значення false як не виконане, якщо true як виконане. Нотатки - рядкове значення, додаткова інформація, яка стосується завдання. Необов'язковий тип даних. Дата початку - Кількість мілісекунд від 1960 року, відображає дату та час, коли починається виконання завдання. Необов'язковий тип даних. Тривалість - Кількість мілісекунд, відображає скільки потрібно годин та хвилин на виконання завдання. Необов'язковий тип даних. Ярлики - перелік рядкових значень. Ярлики допомагають у сортуванні та фільтрації завдань. Необов'язковий тип даних. Важливе. - логічне значення, надає можливість визначити завдання як важливе, що допомагає у сортуванні та фільтрації завдань. Необов'язковий тип даних. Пріоритет - значення типу пріоритет. Надає можливість пріоритизувати завдання, що допомагає у сортуванні та фільтрації завдань. Необов'язкове поле, завдання може не бути без обраного пріоритету. Список - значення типу список. Відображає, до якого списку належить завдання. Необов'язкове поле, завдання може не належати до списку. Період - значення типу період. Відображає до якого періоду належить завдання. Необов'язкове поле, завдання може не належати до періоду. Користувач - кожне завдання належить певному користувачу.

2.1.3 Список

Під списком будемо розуміти сутність, яка зберігає перелік завдань.

Список складається з наступних даних:

- ідентифікатор;
- назва;
- перелік завдань.

Ідентифікатор - рядкове значення, унікальний згенерований ключ-значення за яким відбувається звернення до сутності. Обов'язковий тип даних. Назва - рядкове значення, відображає назву списку. Обов'язковий тип даних.

2.1.4 Період

Під періодом будемо розуміти сутність, яка відображає часовий проміжок дня. Ця сутність допомагає розмежувати один тип завдань від інших. Наприклад, є період «робота» та «відпочинок». Завдання, які належать до роботи, виконуються в робочий період, а завдання, які належать до відпочинку, виконуються в період відпочинку.

Період складається з наступних даних:

- ідентифікатор;
- назва;
- дата початку;
- тривалість;
- перелік днів тижня;
- колір;
- перелік завдань.

Ідентифікатор - це рядкове значення, унікальний згенерований ключ-значення, за яким відбувається звернення до сутності. Він є обов'язковим типом даних. Назва - це також рядкове значення, відображає назву періоду і є обов'язковим типом даних. Дата початку налічує кількість мілісекунд від 1960 року, відображає дату та час, коли починається період. Також є обов'язковим типом даних. Тривалість вимірюється у кількості мілісекунд, є обов'язковим типом даних. Перелік днів тижня відображає, для яких саме днів тижня відображається період. Колір є рядкове значенням. Використовується для

візуального розділення одного періоду від іншого. Зберігає колір у HEX-форматі. Перелік завдань відображає, які завдання належать до даного періоду.

2.1.5 Ярлик

Під ярликом будемо розуміти сутність, яка надає користувачу додатковий інструмент сортування завдань. Користувачу надається можливість створювати будь яку кількість ярликів.

Ярлик складається з наступних даних:

- ідентифікатор;
- назва.

Ідентифікатор - рядкове значення, унікальний згенерований ключ-значення за яким відбувається звернення до сутності. Обов'язковий тип даних. Назва - рядкове значення, відображає назву ярлику. Обов'язковий тип даних.

2.1.6 Пріоритет

Під пріоритетом будемо розуміти, сутність яка відображає рівень пріоритетності завдання. Користувач може обрати лише пріоритет визначений системою і не може створювати власні пріоритети.

Пріоритет складається з наступних даних:

- рівень пріоритетності;
- назва.

Рівень пріоритетності - цілочисельне значення. Обов'язковий тип даних. Назва - рядкове значення. Відображає назву пріоритету.

Системою визначені наступні пріоритети:

- «Низький»;
- «Середній»;
- «Високий».

2.1.7 Зв'язок між сутностями

Для того щоб відобразити зв'язки між об'явленими сутностями, було створено схему залежностей між сутностями (див. рис. 2.1).

Користувач зберігає посилання на завдання, списки, ярлики, і періоди які були створені для користувача.

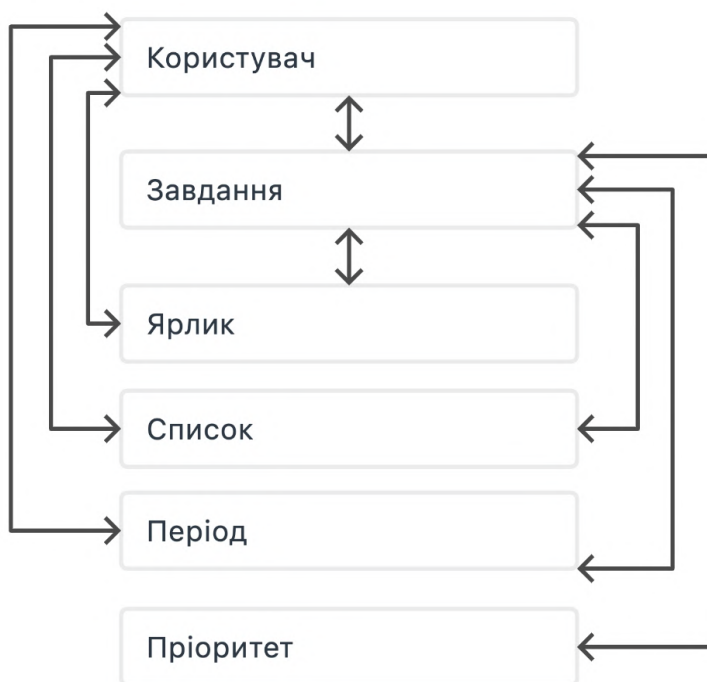


Рисунок 2.1 – Схема залежностей між сутностями

Завдання має посилання до якого списку, періоду, та користувача воно належить. Також кожен користувач має посилання на пріоритет та ярлики які використовуються для завдання.

Список має посилання на користувача до якого він належить, та посилання на завдання які відносяться до списку.

Період має посилання на користувача до якого він належить, та посилання на завдання які належать до періоду.

Ярлик має посилання на користувача до якого він належить, та посилання на завдання в яких ярлик використовується.

Пріоритет має посилання на завдання, в яких використовується пріоритет.

2.2 Проєктування екранів додатку

Під час проєктування екранів додатку були поставлені наступні задачі:

- дотримуватися сучасних стилів дизайну, таких як Flat Design;
- дизайн система обмежена палітрою кольорів;
- інтерфейс має бути інтуїтивно зрозумілий який дотримується вимог Apple Human Interface Guidelines [16];
- зберігати мінімалістичність в інтерфейсі екранів. Намагатися не додавати зайвих елементів;
- адаптивність під різні розміри екранів;
- використати читабельні шрифти.

Спочатку було визначено перелік екранів, які входять у додаток і є частиною інтерфейсу:

- екран «Планування»;
- екран «Нове завдання»;
- екран «Деталі завдання»;
- екран «Списки»;
- екран «Список»;
- екран «Ярлики»;
- екран «Періоди»;
- екран «Помічник».

Далі було визначено схему навігації між екранами додатку (див. рис. 2.2).

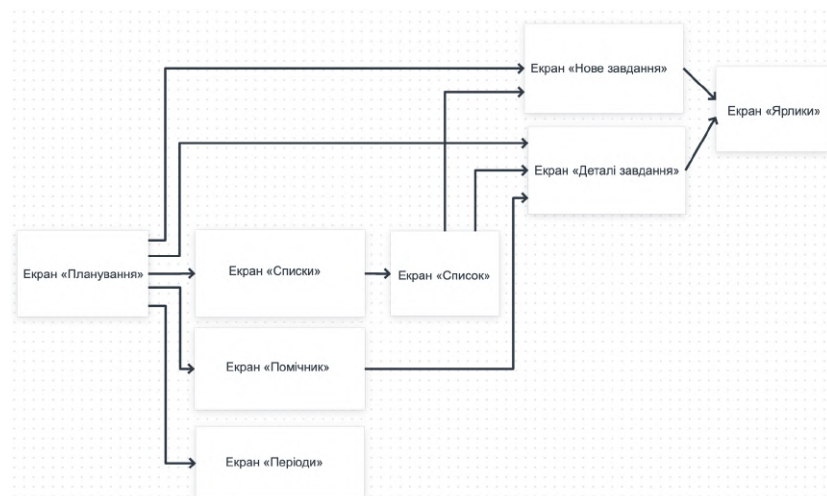


Рисунок 2.2 – Схема навігації між екранами

2.2.1 Екран «Планування»

Це перший екран (див. рис. 2.3), який бачить користувач коли заходить в систему. Екран планування є основним екраном, з яким користувач буде проводити більшість часу.

На екрані «Планування» користувачу надається можливість переглядати та планувати завдання лише для поточного дня. Можна як створювати нові завдання, так і обирати раніше створені та додавати їх у графік. Також у графік можна додавати періоди, видаляти додані та змінювати тривалість обраних періодів.

Екран складається з секцій: верхня секція, середня секція та нижня секція.

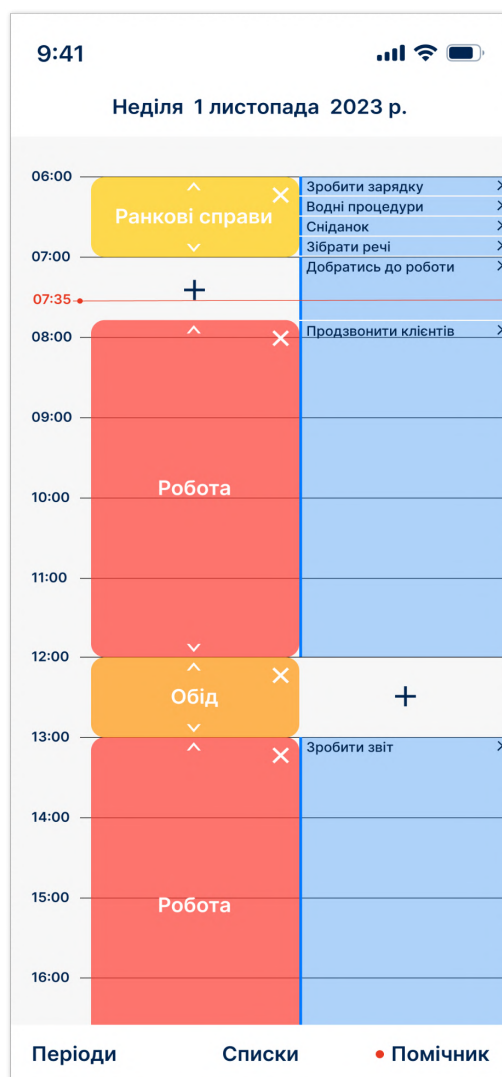


Рисунок 2.3 – Екран «Планування»

Верхня секція відображає такі дані, який сьогодні день тижня, день, місяць і рік.

Середня секція відображає графік поточного дня. Графік поточного дня відображає часову сітку погодинно, починаючи з «00:00» і закінчуючи «24:00». Окрім відображення часової сітки погодинно, окремим елементом відображається відображення поточного часу.

Часова сітка поділяється на дві частини: «Налаштування періодів» і «Планування завдань».

В «Налаштування періодів» всі вільні часові проміжки відображають елемент «Додати період» який додає період в обраний часовий проміжок. Якщо натиснути «Додати період», тоді з'явиться елемент вибору періоду з доданих періодів в систему. За замовчуванням період додається в часовий проміжок який дорівнює годині. Всі часові проміжки які займає період можливо змінювати потягнувши за верхню або нижню границю часового проміжка і тягнути в сторону збільшення або зменшення часового проміжку. Також надається можливість видаляти створений часовий проміжок. Якщо затиснути обраний часовий проміжок з'явиться можливість пересунути його на інший часовий проміжок який ще не зайнятий.

В «Плануванні завдань» в вільних часових рамках, які дорівнюють годину або менше, з'являється елемент «Спланувати завдання». При натисканні з'являється елемент, який надає можливість створити завдання або обрати з існуючих.

Якщо обрати «Створити», тоді відкриється екран «Нове завдання». Екран відкривається з вже налаштованими даними дати, часу та тривалості, які були обрані на екрані «Планування». По завершенню роботи з цим екраном, якщо створюється завдання воно додається в графік на екрані «Планування».

Якщо обрати «Вибрати», тоді відкриється екран «Списки». На цьому екрані надається можливість обрати список, тоді відкриється екран «Список» в якому можна обрати завдання. Якщо завдання вже додане на екрані «Планування», тоді воно не відображається серед завдань на екрані «Список».

Завдання відображаються в елементі «Заплановане завдання», який відображає назву та час і тривалість завдання. Елемент «Заплановані завдання» можна затиснути і перемістити в інше місце по часовій сітці. Якщо натиснути на заплановане завдання, тоді відкриється екран «Деталі завдання». В «Плануванні завдань» можна видаляти заплановані завдання натиснувши хрестик в правому верхньому куту елемента «Заплановане завдання». Завдання не можуть перетинатися за часом. Діє правило одне завдання на один проміжок часу.

Нижня секція складається з кнопок: «Списки», «Періоди», «Помічник». Кнопка «Списки» відкриває екран «Списки». Кнопка «Періоди» відкриває екран «Періоди». Кнопка «Помічник» відкриває екран «Помічник». Якщо у підсистемі «Помічник» є пропонуємі поради, тоді кнопка «Помічник» відображає красний індикатор.

При першому запуску додатку, першим екраном відкривається екран «Помічник», який допомагає налаштувати періоди для додатку.

Екран «Планування» кожен новий день налаштовує періоди за шаблоном як періоди налаштовані на екрані «Періоди». Після налаштування періодів на екрані «Планування» періоди можна змінювати незалежно від налаштувань які визначені на екрані «Періоди».

Кожен новий день починається з вільним графіком планування завдань. Помічник пропонує користувачу зробити автоматичне перенесення завдань з попереднього дня які не були завершені. Для того щоб переглянути запропоноване оновлення, треба відкрити екран «Помічник».

2.2.2 Екран «Нове завдання»

Екран надає можливість користувачу створити нове завдання (див. рис. 2.4). Всі зміни на екрані зберігаються автоматично.

На екрані зверху розміщена навігаційна панель з елементами, розглянутими далі.

Кнопка назад надає можливість повернутись до попереднього екрану:

– заголовок екрану, має текст «Деталі завдання»;

– «Додати» підтверджує створення нового завдання. Приймає неактивний стан, якщо заповнені не всі необхідні поля для завдання.

9:41

< Нове завдання Додати

Назва

Нотатки

Дата - Час - Тривалість

Період Немає

Список Немає

Важливе ☐

Пріоритет Немає

Ярлики Немає

Рисунок 2.4 – Екран «Нове завдання»

Екран складається з наступних розділів:

– «Назва», яка відображає назву завдання. Якщо користувач натисне на розділ «Назва», тоді з'явиться клавіатура і у користувача буде можливість змінити назву завдання. Назва є обов'язковим полем для створення завдання.

– «Нотатки», які відображають додаткові деталі завдання. Якщо користувач натисне на розділ «Нотатки», тоді з'явиться клавіатура і у користувача буде можливість ввести текст. Цей розділ є не обов'язковим і може залишатися пустим.

– «Дата - Час - Тривалість», які відображають дату і час, коли запланована подія та тривалість, скільки потрібно часу на виконання завдання.

– «Період», цей розділ відображає до якого періоду належить завдання. Якщо обрана дата та час входить до одного з визначених системою періодів тоді буде відображена назва періоду. Якщо обрана дата та час не належить до жодного періоду або дата та час не обрані, тоді відображається текст «Немає». Цей розділ є інформативним і не надає користувачу можливості обирати на пряму інший період, період залежить суто від обраної дати та часу.

– «Список», цей розділ відображає, до якого списку належить завдання. Завдання може не належати до жодного списку. Якщо завдання не належить до списку тоді відображається текст «Немає».

– «Важливе», відображає логічне значення, чи помічене завдання як важливе чи ні.

– «Пріоритет», відображає пріоритетність завдання. Може бути 3 рівні пріоритетності, низький, середній та високий. Пріоритет може бути не обраний, в цьому випадку відображається текст «Немає». Для того щоб вибрати пріоритет має натиснути на розділ «Пріоритет», з'явиться елемент зі списком можливих варіантів, натиснувши на один із варіантів зберігається обраний пріоритет.

– «Ярлики» цей розділ відображає обрані ярлики до яких належить завдання. Якщо жодного ярлику не обрано відображається текст «Немає». Для того щоб обрати ярлики користувач має натисну на розділ.

2.2.3 Екран «Деталі завдання»

Екран надає можливість користувачу подивитись, всі деталі завдання та змінити їх (див. рис. 2.5). Всі зміни на екрані зберігаються автоматично.

На екрані зверху розміщена навігаційна панель з наступними елементами:

- кнопка назад;
- деталі завдання;
- чекбокс, який надає можливість змінити статус виконання завдання.

Пустий чекбокс відображає, що завдання невиконане. Чекбокс з галкою відображає завдання як виконане.

Екран складається з наступних розділів:

- «Назва»;
- «Нотатки»;
- «Дата - Час - Тривалість»;
- «Період»;
- «Список»;
- «Важливе»;
- «Пріоритет»;
- «Ярлики».

Властивості перелічених елементів такі самі як описано для екрана «Нове завдання».

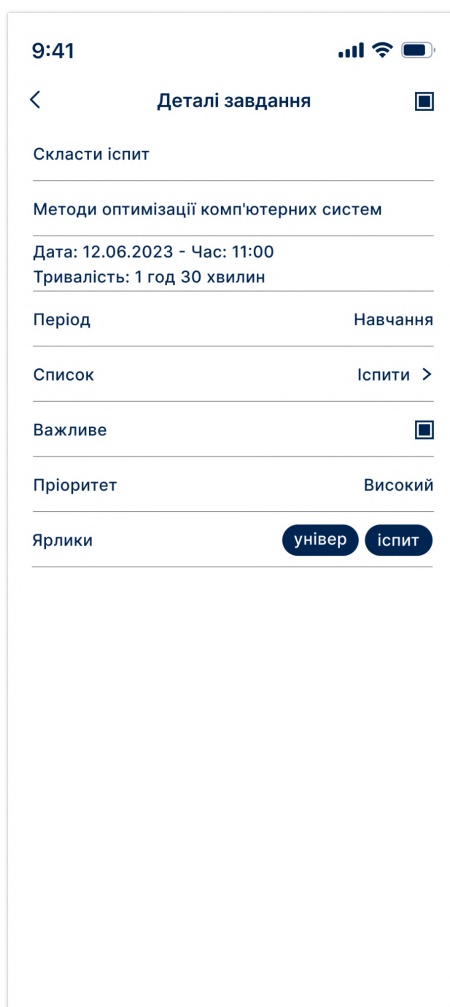


Рисунок 2.5 – Екран «Деталі завдання»

2.2.4 Екран «Списки»

Цей екран (див. рис. 2.6) надає можливість переглянути всі завдання які створені в системі. Завдання категоризуються по спискам.

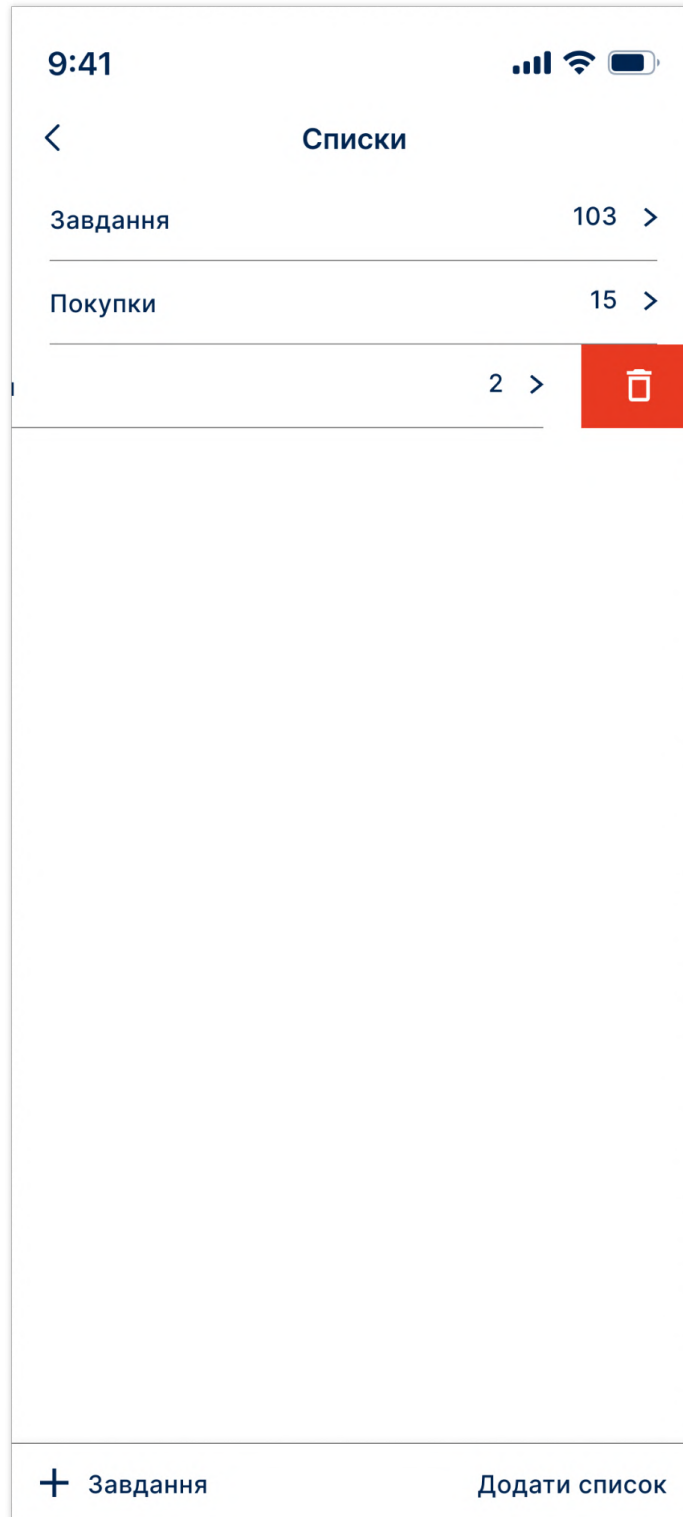


Рисунок 2.6 – Екран «Списки»

У верхній частині екрана розташована навігаційна панель яка складається з елементів:

- кнопка назад. Надає можливість повернутись до попереднього екрану;
- назва екрану з текстом «Списки».

Посередині екрана відображаються всі списки які є в системі. Першим списком відображається системний список «Завдання», який відображає всі завдання які були створені без вибраного списку. Далі відображаються списки створені користувачем. Кожен список відображає назву та значення скільки завдань до нього входить. Списки відображаються в порядку визначеним користувачем окрім списку «Завдання». Список «Завдання» завжди відображається першим. Для всіх інших списків порядок можна змінювати, для цього потрібно затиснути список і для якого потрібно змінити порядок і перетягнути його в інше місце по ієрархії, відпустивши список він автоматично зберігає своє положення. Якщо провести по списку свайпом ліворуч, відобразиться кнопка «Видалити», яка дозволяє видалити список. Якщо видалити список, тоді всі завдання з видаленого списку будуть відображені в системному списку «Завдання».

У нижній частині екрана екрана відображається панель з наступними елементами:

- «+ Завдання», викликає екран «Нове завдання для створення нового завдання»;
- «Додати список», надає можливість створити новий список. Натиснувши цю кнопку, в розділі де відображаються списки, в самому кінці після останнього списку додається елемент створення нового списку і відкривається клавіатура яка надає можливість ввести назву списку. Якщо ввести назву новго списку і натиснути на клавіатурі кнопку готово, тоді список буде збережено і додано до системи. Якщо назва лишиться пустою і натиснути кнопку готово, тоді завершиться процес створення списку і список не буде створено.

2.2.5 Екран «Список»

Цей екран надає можливість подивитись всі завдання, які належать до обраного списку (див. рис. 2.7).

Екран складається з секцій: навігаційна панель, завдання, створити завдання.

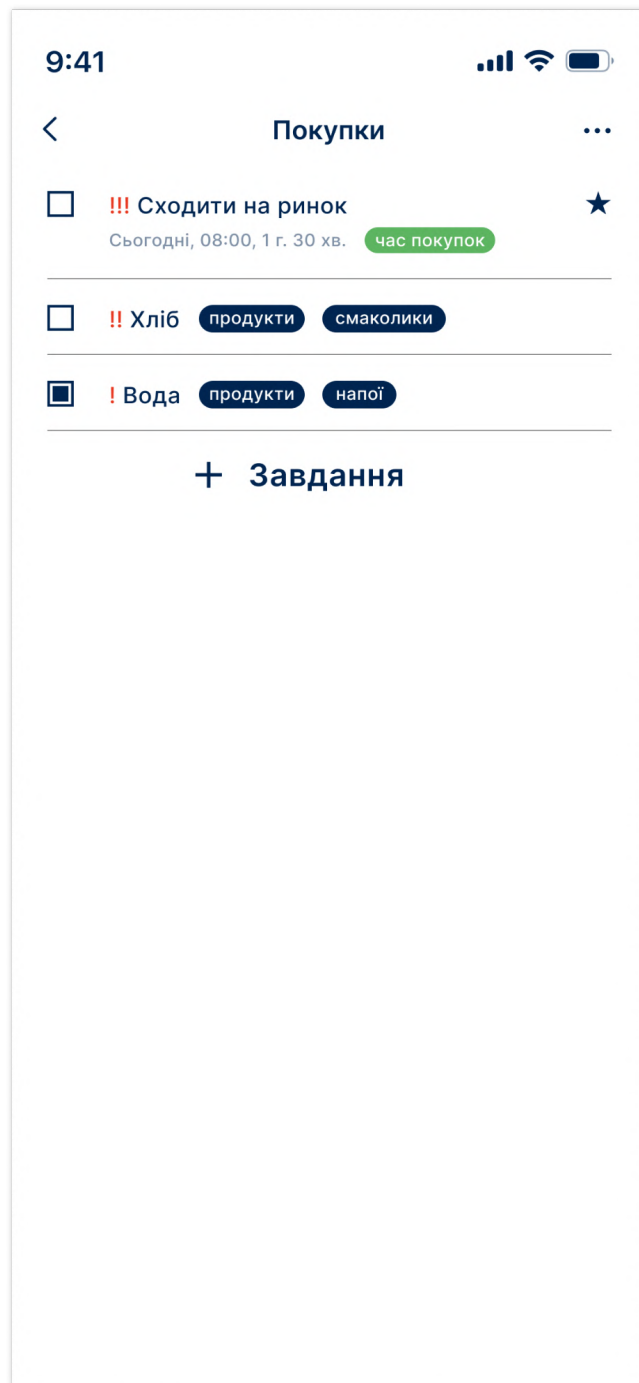


Рисунок 2.7 – Екран «Список»

Навігаційна панель складається з елементів: кнопка назад, назва списку, кнопка налаштувань.

Якщо натиснути на назву тоді з'явиться клавіатура і активується можливість коригувати назву списку. Якщо текст назви списку введено, то по натисканню кнопки «Done» на клавіатурі зміни зберігаються. Зберегти пусту назву не дозволено, кнопка done на клавіатурі стає неактивною.

Натиснувши на кнопку налаштувань з'являється меню з переліком налаштувань:

- «Сортувати». Сортувати можна за назвою, вручну (індивідуально вибраним порядком), датою, тривалістю, пріоритетом;

- обрати видимість виконаних завдань (відображати, не відображати);

- «Видалити список». Якщо вибрати пункт «Видалити список» перед видаленням з'явиться системне діалогове вікно з варіантами вибору «Відмінити» та «Видалити». Якщо видалити список, для всіх завдань які належать до цього списку буде витерте поле список і вони залишаться без визначеного списку.

Завдання відображає наступні дані:

- чекбокс який відображає чи виконане завдання. Натиснувши змінюється стан виконання завдання;

- назва завдання.

- якщо у завдання обрано пріоритет, відображається елемент, який позначає пріоритетність завдання. Три, два або один знаки оклику в залежності від рівня пріоритетності;

- якщо завдання помічене як важливе відображається зірочка;

- якщо обрані, відображаються дата час та тривалість;

- якщо обрані дата та час співпадають з одним із періодів, відображається період;

- якщо обрано ярлики, відображаються елементи, які відображають обрані ярлики;

- якщо натиснути на завдання, відкриється екран «Деталі завдання». Якщо натиснути на кнопку «Створити завдання», відкриється екран «Нове завдання».

2.2.6 Екран «Ярлики»

Цей екран (див. рис. 2.8) надає можливість обрати або прибрати ярлики які відносяться до завдання. Всі зміни на екрані зберігаються автоматично.

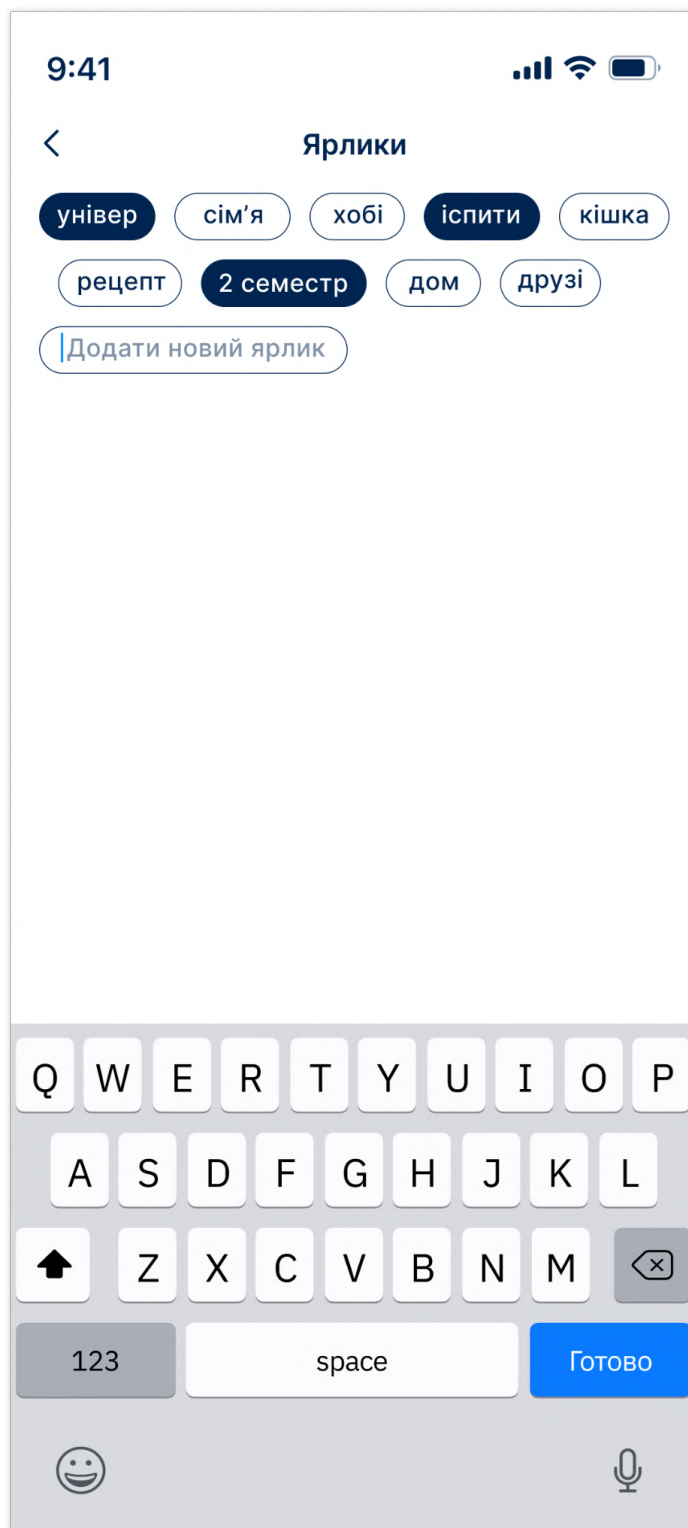


Рисунок 2.8 – Екран «Ярлики»

Екран складається з навігаційної панелі, секції вибору ярликів та секції створення нового ярлика.

Навігаційна панель відображає кнопку повернення до попереднього екрана та заголовок екрана з текстом «Ярлики».

Секція вибору ярликів відображає всі ярлики які були створені користувачем. Кожен ярлик має стан обраний або не обраний. Якщо натиснути на ярлик його стан змінюється.

Секція створення ярлику складається з елемента вводу який викликає клавіатуру. Якщо ввести назву і натиснути на клавіатурі кнопку «Готово ярлик буде створено». Якщо залишити поле вводу пустим і натиснути «Готово» клавіатура закриється. Якщо ввести назву вже існуючого ярлику, тоді новий ярлик не буде створено, а буде змінений стан ярлика на обраний, якщо попередньо він не був обраний, якщо попередньо був обраний тоді просто закривається клавіатура.

2.2.8 Екран «Періоди»

Цей екран (див. рис. 2.9) надає можливість переглянути всі періоди які визначені для системи. Також є можливість створювати нові періоди, вносити зміни в існуючі та видаляти. Всі зміни на екрані зберігаються автоматично.

Екран складається з двох секцій:

- навігаційна панель;
- список періодів.

Навігаційна панель складається з елементів: кнопка назад та заголовок екрану з текстом «Періоди».

Список періодів складається з елементів «Період» та «Створити період».

«Період» відображає назву періоду та обраний колір для періоду. Порядок періодів можна міняти. Для цього потрібно затиснути період і почати перетягувати його в інше місце. Відпустивши порядок періодів змінюється і зберігається автоматично. Якщо провести по періоду свайпом ліворуч, відобразиться кнопка «Видалити», яка дозволяє видалити період. Якщо видалити період, тоді всі відображення періоду видалиться звідусіль, де він був визначений для відображення.

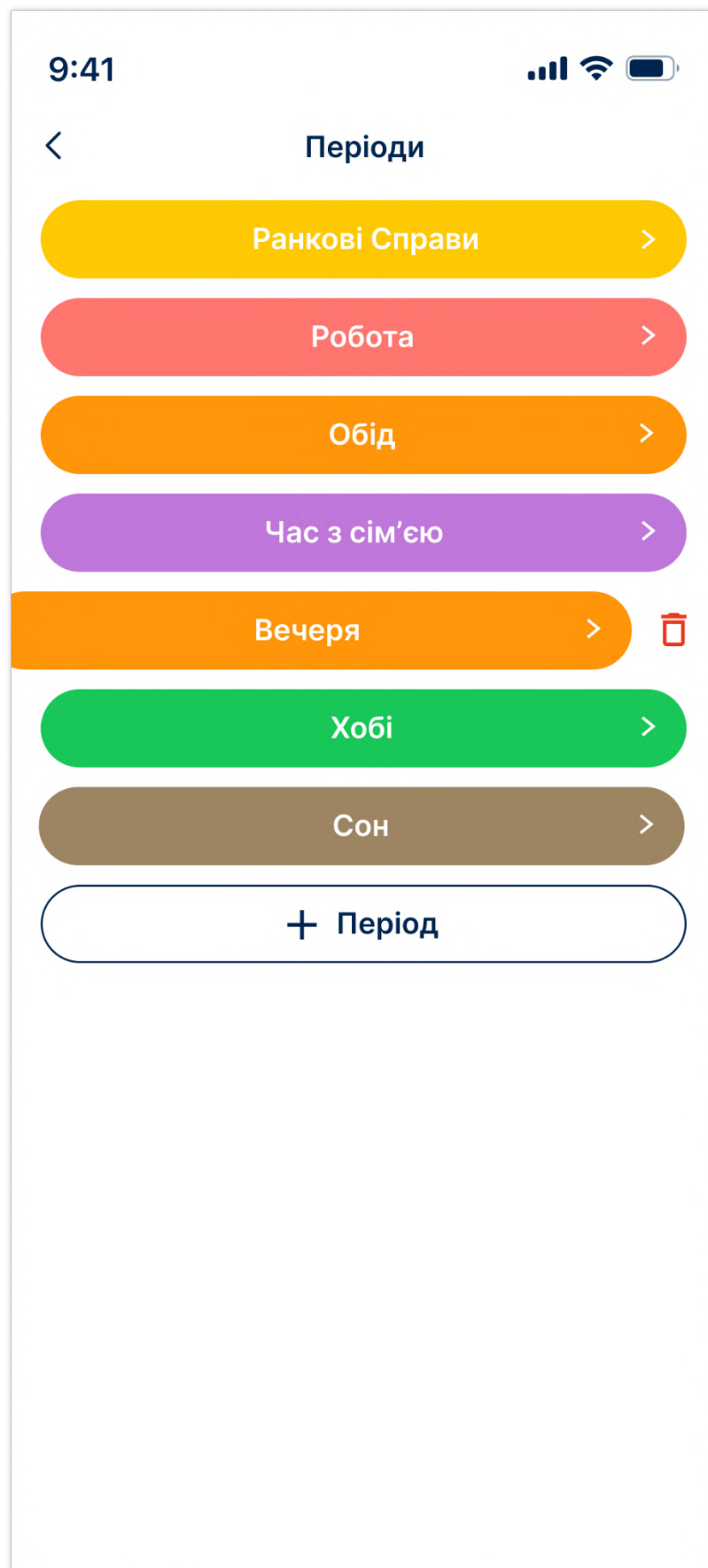


Рисунок 2.9 – Екран «Періоди»

«Створити період» надає можливість створити новий період. Натиснувши цю кнопку, вона змінюється в елемент створення періоду, який надає можливість ввести назву нового періоду і відкривається клавіатура, яка надає можливість ввести назву. Якщо ввести назву новго періоду і натиснути на клавіатурі кнопку «Готово», тоді період буде збережено і додано до системи. Якщо назва лишиться пустою і натиснути кнопку готово, тоді завершиться процес створення періоду і елемент створення поернеться до ісходного стану. Всі періоди створюються з базовим кольором який визначається дизайном додатку. Можно створювати періоди з однаковою назвою, на це обмежень немає.

Для того щоб почати редагування періоду потрібно натиснути на нього. Тоді обраний період розгорнеться на весь екран і перейди в стан редагування. Якщо натиснути на період ще раз він вийде із стану редагування.

Стан редагування складається з елементів: «Назва», «Колір», «Графік» (див. рис. 2.10).

«Назва» відображає назву списку. Для того щоб змінити назву потрібно натиснути по тексту назви, тоді з'явиться можливість змінювати назву. Назву не можна лишити без жодного введенного символу.

«Колір» відображає обраний колір для періоду і надає можливість змінити інший колір. Обирати можна лише серед кольорів визначених дизайном.

Елемент «Графік» відображає комплекс елементів які дозволяють налаштувати в який час і які дні тижня має відображатися період.

Елемент «Обрати день тижня» дозволяє обрати для якого дня тижня внести налаштування, в який час відображати період.

Елемент «Планування дня тижня» відображає часовий проміжок погодинно починаючи з «00:00» і закінчуючи «24:00». Всі вільні часові проміжки відображають елемент «Додати період» який додає період в обраний часовий проміжок. За замовчуванням період додається в часовий проміжок який дорівнює годині. Всі часові проміжки які займає період можливо змінювати потягнувши за верхню або нижню границю часового проміжка і тягнути в сторону збільшення або зменшення часового проміжку. Також надається можливість видаляти

створений часовий проміжок. Якщо затиснути обраний часовий проміжок з'явиться можливість пересунути його на інший часовий проміжок який ще не зайнятий. Серед часових проміжків періоду який редагується, також відображаються часові проміжки інших періодів які відображаються для плануємого дня тижня. Для часових проміжків з інших періодів також можна змінювати час та тривалість. Часові проміжки з інших періодів також можна видаляти але не можна додавати.

Елемент «Планування дня тижня» можна прокручувати щоб пересуватись по часовій сітці дня.

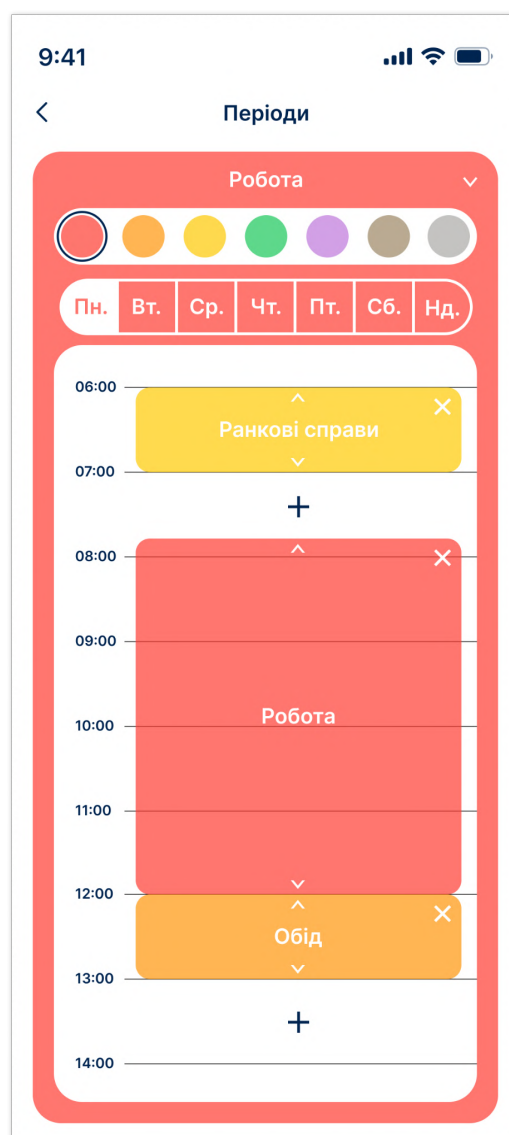


Рисунок 2.10 – Стан редагування періоду

2.2.9 Екран «Помічник»

Цей екран виконує функцію помічника який допомагає користувачу опанувати додаток.

Помічник має наступний функціонал:

- налаштувати періоди для додатку;
- перенести не виконані завдання з попереднього дня.

В наступних ітераціях додатку, функціонал помічника може бути розширеним.

Взаємодія з помічником реалізована в форматі чату. Користувачу надаються варіанти питань або варіанти відповідей які можна надати помічнику. Після того як був наданий запит до помічника він надає відповіді або надає уточнюючі питання.

2.3 Проєктування підсистем додатку

Після того як було спроектовано екрани додатку, було спроектовано підсистеми які виконують головну роль у керуванні даними та їх зміни в додатку.

Підсистеми - це частини або компоненти системи, які мають власні функціональні можливості і можуть працювати самостійно чи взаємодіяти з іншими підсистемами для досягнення спільної мети.

Було визначено наступні підсистеми [17] (див. рис. 2.11):

- підсистема керування базою даних;
- підсистема помічник;
- підсистема керування користувачем;
- підсистема керування завданнями;
- підсистема керування списками;
- підсистема керування періодами;
- підсистема керування ярликами.

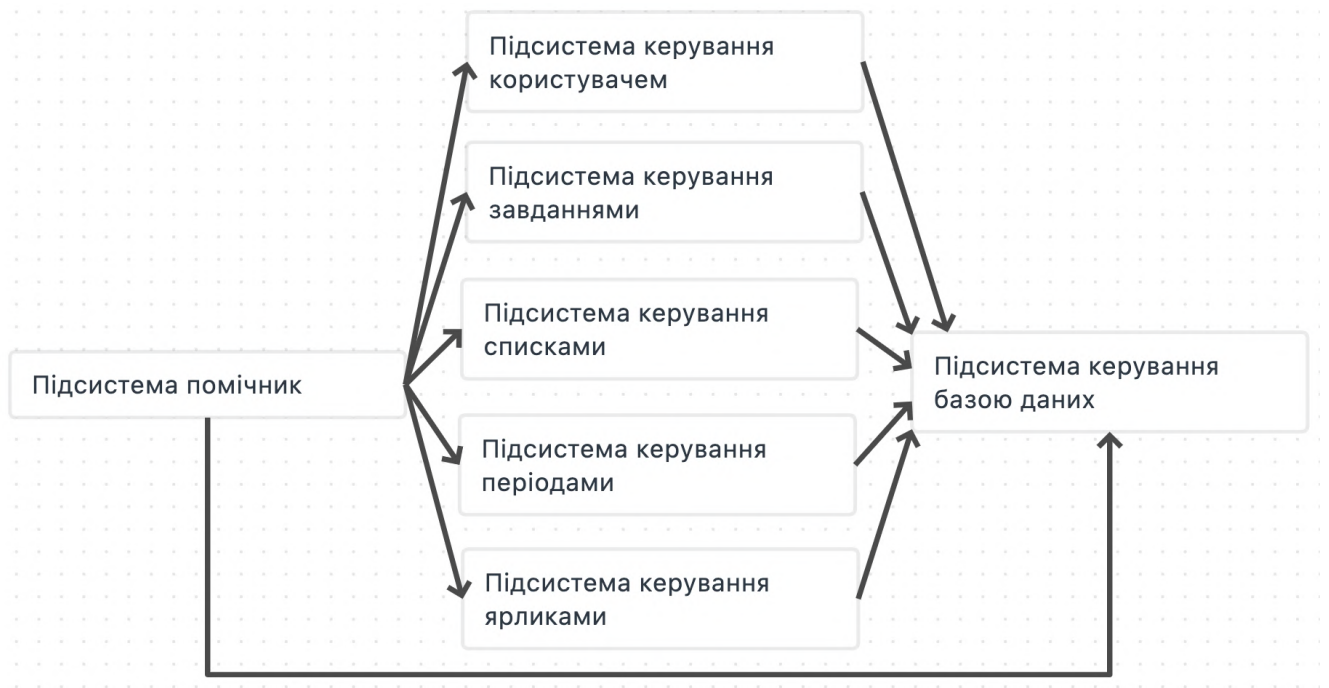


Рисунок 2.11 – Підсистеми додатку і зв'язок між ними

2.3.1 Підсистема керування базою даних

Відповідає за розгортання бази даних та її налаштування.

Надає інтерфейси:

- дістати сутність;
- створити сутність;
- видалити сутність;
- зберегти зміни сутності.

2.3.2 Підсистема помічник

Виконує роль головної підсистеми яка реалізує основну бізнес логіку додатку.

Виконує наступні завдання:

- первинне налаштування користувача;
- створення пріоритетів для системи;
- підбір та налаштування періодів враховуючи відповіді користувач;
- автоматичне переналаштування завдань на поточний день враховуючи відповіді користувача;

– автоматичне налаштування періодів на поточний день спираючись на періоди які визначені в системі.

Звертається до підсистем:

- підсистема керування базою даних;
- підсистема керування завданнями;
- підсистема керування періодами.
- підсистема керування списками;
- підсистема керування ярликами.

2.3.3 Підсистема керування завданнями

Відповідає за сутність «Завдання». Зберігає бізнес логіку пов'язану з тим як сутність «Завдання» має зберігатись у базі даних.

Надає інтерфейси:

- створити завдання;
- видалити завдання;
- оновити завдання.

Звертається до підсистеми керування базою даних.

2.3.4 Підсистема керування списками

Відповідає за сутність «Список». Зберігає бізнес логіку пов'язану з тим як сутність «Сутність» має зберігатись у базі даних.

Надає інтерфейси:

- створити список;
- видалити список;
- оновити список.

Звертається до підсистеми керування базою даних.

2.3.5 Підсистема керування періодами

Відповідає за сутність «Період». Зберігає бізнес логіку пов'язану з тим як сутність «Період» має зберігатись у базі даних.

Надає інтерфейси:

- створити період;
- видалити період;

- оновити період.

Звертається до підсистеми керування базою даних.

2.3.6 Підсистема керування ярликами

Відповідає за сутність «Ярлик». Зберігає бізнес логіку пов'язану з тим як сутність «Ярлик» має зберігатись у базі даних.

Надає інтерфейси:

- створити ярлик;
- видалити ярлик;
- оновити ярлик.

Звертається до підсистеми керування базою даних.

2.3.7 Підсистема керування користувачем

Відповідає за сутність «Користувач». Зберігає бізнес логіку пов'язану з тим як сутність «Користувач» має зберігатись у базі даних.

Надає інтерфейси:

- створити користувача;
- видалити користувача;
- оновити користувача.

Звертається до підсистеми керування базою даних.

2.4 Проектування архітектури додатку

В якості архітектури додатку був обран паттерн MVVM [14] (Model-View-ViewModel), доповнений паттерном Coordinators (див. рис. 2.12).

MVVM призначений для відокремлення проблем та покращення тестируваності та перевикористовуваності коду [18]. Він складається з трьох ключових компонентів:

- «Model» представляє дані та бізнес-логіку додатка. Вона інкапсулює структури даних, сервіси та взаємодії зовнішніми API чи базами даних. «Model»

не залежить від інших компонентів архітектури та є відокремленою від інтерфейсу користувача.

– «View» відповідає за відображення елементів інтерфейсу користувача та взаємодію з користувачем. У MVVM вид спостерігає за змінами в ViewModel і оновлює себе відповідно. Він не повинен містити жодної бізнес-логіки та повинен залишатися незалежним від «Model» та ViewModel.

– «ViewModel» виступає як міст між «Model» та «View». Він відкриває необхідні дані та команди, які «View» повинен відображати та обробляти користувацькі взаємодії. «ViewModel» отримує оновлення від «Model», трансформує дані, якщо потрібно, та відкриває їх «View». Він також містить логіку обробки дій користувача та взаємодіє з «Model» для виконання операцій, таких як отримання даних чи їх оновлення.

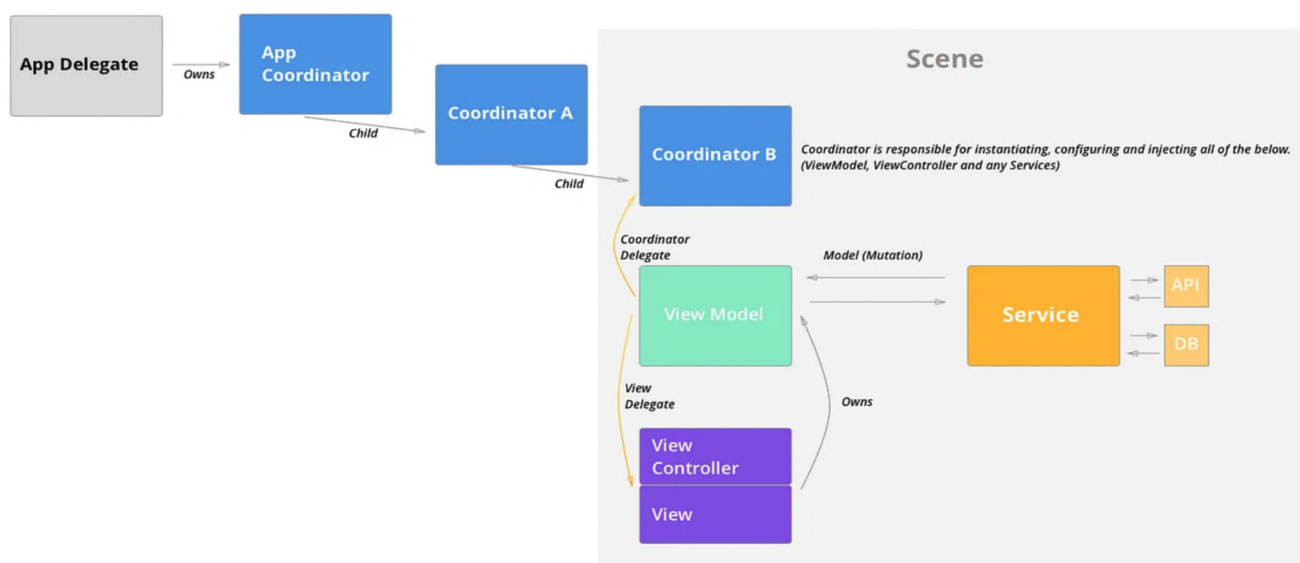


Рисунок 2.12 – Схема MVVM+Coordinators

Хоча MVVM забезпечує надійну структуру для управління потоком даних та відокремлення концернів, він не вирішує проблему навігації та контролю потоку в межах додатка. Тут на допомогу приходить паттерн Coordinators. Паттерн Coordinators допомагає у керуванні навігацією та потоком екранів в додатку, інкапсулюючи логіку навігації в окремі об'єкти координаторів.

Coordinator, це об'єкт, який бере на себе відповідальність за координацію потоку між різними екранами чи контролерами вигляду. Кожен екран чи контролер вигляду має свого власного координатора, який знає, як переходити на наступний екран і коли відхиляти чи відображати інші екрани. Координатори також можуть обробляти завдання, такі як впровадження залежностей та передача даних між екранами.

2.5 Висновки до розділу

Під час проєктування виявлені основні сутності, якими оперує система, встановлені зв'язки між цими сутностями, а також визначені набори даних, які вони зберігають. Далі були розроблені екрани додатку, для кожного з яких був наданий опис, а також розроблений дизайн. Кожен елемент дизайну екрана був детально описаний з вказівкою на його призначення. Крім того, для кожного екрана була розроблена бізнес-логіка, а також визначено, якими сутностями конкретний екран взаємодіє. Після проєктування екранів були спроектовані підсистеми додатку, і були надані описи зв'язків між цими підсистемами.

Завершальним етапом було проєктування архітектури додатку після завершення проєктування його компонентів.

3 РЕАЛІЗАЦІЯ СИСТЕМИ

Система реалізована як застосунок, на операційній системі iOS. Операційна система iOS є мобільною платформою, розробленою компанією Apple Inc. для їхніх мобільних пристроїв, зокрема iPhone, iPad і iPod Touch. Користувацький

інтерфейс iOS заснований на концепції прямої маніпуляції з використанням жестів Мультитач. Елементи інтерфейсу управління складаються з повзунків, перемикачів і кнопок. Він призначений для безпосереднього контакту користувача з екраном пристрою.

У операційній системі iOS застосунок - це програмне забезпечення, розроблене для виконання конкретних завдань чи надання певних послуг на мобільних пристроях Apple. Застосунки представляють собою ключовий елемент мобільного досвіду користувача і розширюють функціональність пристрою.

3.1 Засоби розробки

Ключовим етапом у створенні будь-якого програмного продукту є вибір інструментів розробки, які дозволять реалізувати весь необхідний функціонал та спростять програмування завдяки зручному інтерфейсу та використанню наявних технологій. Оптимальний вибір інструментів не лише полегшує взаємодію користувача з системою, але й робить процес розробки програмного забезпечення більш ефективним та швидким. Технології, використані під час розробки, включають:

- середовище розробки «Xcode»;
- мову програмування «Swift»;
- систему контролю версій «Git»;
- фреймворк «Foundation.framework»;
- фреймворк «UIKit.framework»;
- фреймворк «Combine.framework»;
- фреймворк «XCTest»;
- стороння бібліотека «Dependencies»;
- стороння бібліотека «CombineCocoa»;
- стороння бібліотека «SnapKit»;

– «Realm» SDK.

3.1.1 Середовище розробки «Xcode»

«Xcode» [19] (див. рис. 3.1) - це інтегроване середовище розробки (IDE) від Apple, призначене для створення програм для платформ iOS, iPadOS, macOS, watchOS та tvOS. Основною мовою програмування, яку підтримує «Xcode», є «Swift», а також «Objective-C».

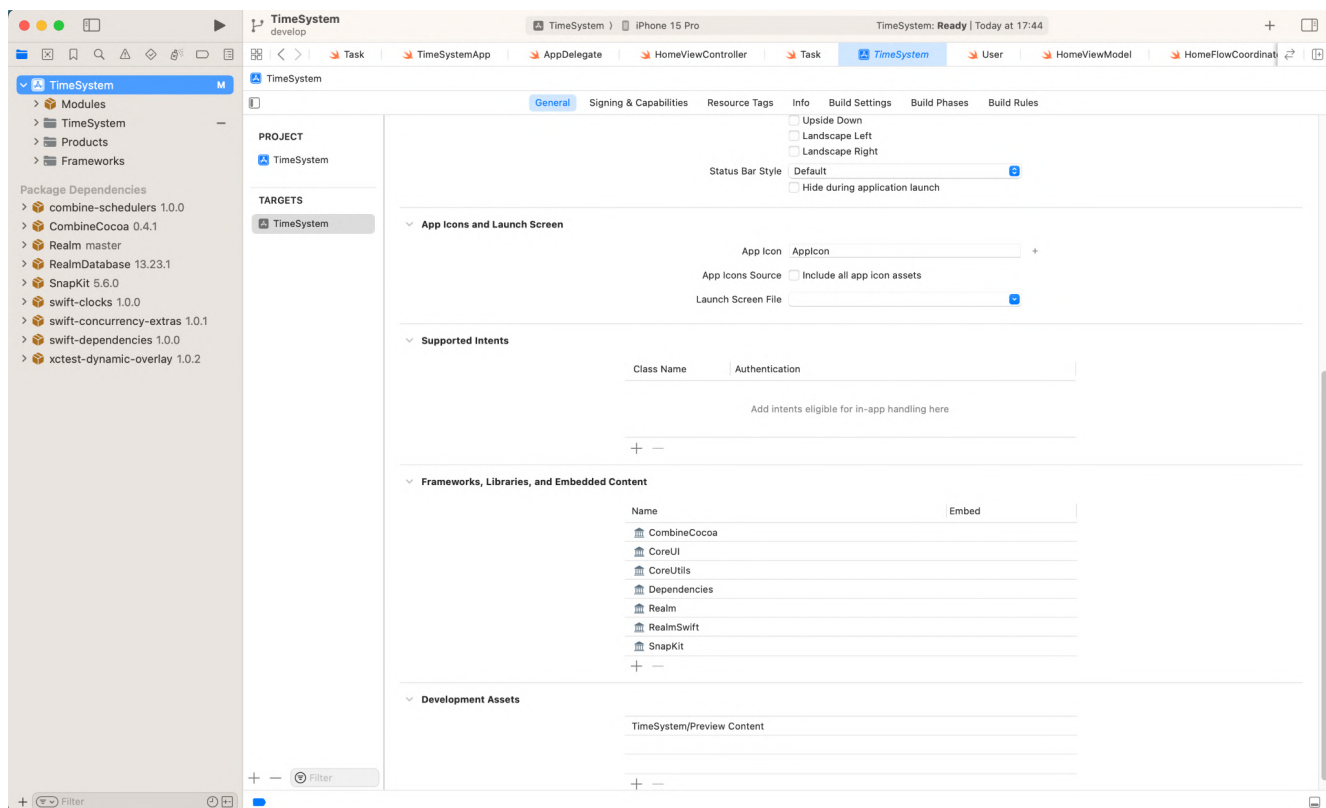


Рисунок 3.1 – Середовище розробки Xcode

«Xcode» надає потужний текстовий редактор з розширеними можливостями автодоповнення коду, висвітлення синтаксичних помилок та інструментами для швидкого переходу між файлами.

Є візуальний редактор інтерфейсу користувача, що дозволяє розробникам створювати графічний інтерфейс для своїх додатків безпосередньо в середовищі розробки.

«Xcode» має вбудовані інструменти для налагодження коду, включаючи точкові зупинки, перегляд значень змінних та відстеження викликів функцій.

«Xcode» надає інструменти для аналізу продуктивності коду та виявлення можливих проблем щодо швидкодії та оптимізації.

В «Xcode» вбудовано підтримку систем управління версіями, таких як Git, що полегшує спільну роботу над проєктами.

«Xcode» дозволяє розгортати готові додатки на реальних пристроях або симуляторі.

3.1.2 Мова програмування «Swift»

«Swift» це високорівнева мова програмування, розроблена компанією Apple для створення програм для їхніх платформ, таких як iOS, macOS, watchOS та tvOS. «Swift» була представлена в 2014 році як заміна мові «Objective-C», що дозволило розробникам створювати програми для продуктів Apple більш безпечними, ефективними та зручними для розробки [20].

«Swift» має вбудовані заходи безпеки, такі як інференція типів, що дозволяє виявляти та уникати багатьох типових помилок під час компіляції.

«Swift» оптимізована для високої швидкодії та ефективності в порівнянні з іншими мовами програмування.

Мова «Swift» включає сучасні концепції програмування та підтримує функціональне програмування, що робить її потужним інструментом для розробки.

«Swift» надає простий та елегантний синтаксис, що полегшує читання та написання коду.

3.1.3 Система контролю версій «Git»

Система контролю версій «Git» [21] це розподілена система управління версіями, яка призначена для відстеження змін у файловій системі під час розробки програмного забезпечення. «Git» був створений Лінусом Торвальдсом і став вкрай популярним інструментом серед розробників завдяки його простоті та потужній функціональності.

Кожен розробник, який працює з «Git», має повний доступ до історії проєкту та може вести роботу навіть в автономному режимі. Це дозволяє ефективно працювати з версіями коду навіть без підключення до центрального сервера.

«Git» надає потужні інструменти для створення та управління гілками, що дозволяє розробникам незалежно працювати над різними функціональностями або рішенням проблем, а потім об'єднувати зміни.

«Git» зберігає повну історію комітів, що дозволяє відстежувати кожну зміну в проєкті та робити відкат до певного стану коду.

«Git» підтримує віддалені репозиторії, що дозволяє розробникам спільно працювати над проєктом та обмінюватися змінами через мережу.

«Git» володіє високою швидкістю виконання операцій, навіть при роботі з великими проєктами та історіями.

3.1.4 Фреймворк «Foundation.framework»

«Foundation.framework» [22] це фреймворк в екосистемі розробки програмного забезпечення Apple, який надає основні класи та функції для роботи з базовими структурами даних, мережевими операціями, рядками, датами, та іншими ключовими функціональностями.

Фреймворк надає основні класи, такі як NSObject, NSArray, NSString, NSDate, які є основою для багатьох інших класів та об'єктів.

Foundation включає різноманітні класи для роботи з колекціями даних, такі як масиви (NSArray), словники (NSDictionary), та інші.

Фреймворк містить класи для обробки та маніпуляції рядками (NSString), а також роботи з текстовою інформацією.

Foundation можна легко виконувати мережеві запити, використовуючи класи, такі як NSURLConnection та NSURLSession.

Фреймворк дозволяє працювати з файловою системою, читати та записувати файли, використовуючи класи, такі як NSFileManager.

Foundation включає класи для роботи з датами та часом, включаючи NSDate та NSDateFormatter.

Фреймворк дозволяє розробникам використовувати механізми для локалізації своїх додатків та ресурсів.

3.1.5 Фреймворк «UIKit.framework»

«UIKit.framework» [23] це фреймворк в екосистемі розробки програмного забезпечення Apple, спеціально призначений для створення користувацьких інтерфейсів для додатків, які працюють на платформах iOS, iPadOS, та tvOS. «UIKit» надає набір класів та інструментів для розробки графічних інтерфейсів, включаючи елементи керування, анімацію, обробку подій та інше.

«UIKit» надає базові класи для створення та управління інтерфейсними елементами. «UIView» відповідає за відображення інтерфейсних об'єктів, а «UIViewController» за управління розташуванням та поведінкою екранів.

Фреймворк містить різноманітні елементи керування, такі як кнопки (UIButton), текстові поля (UITextField), перемикачі (UISwitch), та інші, які можна використовувати для створення взаємодії з користувачем.

«UIKit» надає високорівневі класи для створення таблиць (UITableView) та колекцій (UICollectionView), що дозволяють відображати списки даних та зображень у скролінгових областях.

«UIKit» має потужні інструменти для створення анімацій, включаючи блоки анімації, а також класи для малювання та обробки графіки.

Фреймворк надає механізми для обробки подій від користувача, таких як тапи, свайпи, та інші жести.

«UIKit» включає класи для роботи з зображеннями, аудіо та відео, а також можливості для роботи з камерою та галереєю пристрою.

3.1.6 Фреймворк «Combine.framework»

«Combine» [24] це інструмент для реактивного програмування. Основна його мета спростити та полегшити асинхронне програмування та роботу з потоками даних.

«Combine» дозволяє вам представляти потоки даних за допомогою об'єктів, відомих як видавці (publishers). Ці потоки можуть включати як сталий потік даних, так і змінювані значення.

Об'єкти, що підписуються на потоки даних, називаються підписниками (subscribers). Вони визначають логіку обробки отриманих значень та помилок в потоці даних.

«Combine» надає багато вбудованих операторів для обробки та модифікації потоків даних. Це дозволяє зручно виконувати операції, такі як фільтрація, картографування, об'єднання тощо.

«Combine» робить роботу з асинхронним кодом більш прозорою та менш вразливою до помилок завдяки використанню декларативного стилю програмування.

В «Combine» вбудована система обробки помилок, що спрощує роботу з винятками та асинхронними викликами.

3.1.7 Фреймворк «XCTest»

«XCTest» [25] це фреймворк для тестування програмного забезпечення в середовищі розробки Apple. Він призначений для автоматизації процесу написання та виконання тестових сценаріїв. «XCTest» надає розробникам потужні засоби для створення, запуску та аналізу тестів, що допомагає забезпечити надійність та стабільність програмного забезпечення.

«XCTest» надає класи, такі як «XCTestCase», які використовуються для визначення тестових випадків. Розробники можуть створювати методи в цих класах, які містять код для перевірки правильності функціоналу.

«XCTest» має вбудовані функції твердження для перевірки очікуваних результатів під час виконання тестів. Це включає перевірку рівності значень, наявності об'єктів, перевірку помилок та інші.

Розробники можуть використовувати методи «setUp» та «tearDown» для визначення коду, який виконується перед початком тесту та після його завершення відповідно.

«XCTest» дозволяє розробникам легко виконувати тести в асинхронному середовищі, використовуючи функції очікування (expectations) та зворотні виклики.

Розробники можуть об'єднувати тести в тестові набори для логічної організації тестових сценаріїв та запуску їх разом.

XCTest дозволяє розробникам виконувати тести інтерфейсу користувача та тести інтеграції для перевірки взаємодії різних компонентів додатка.

3.1.8 Бібліотеки

«Dependencies» - це бібліотека управління залежностями [26].

«SnapKit» - це бібліотека для розробки інтерфейсу користувача (UI) в додатках для iOS, яка пропонує простий та декларативний спосіб визначення автолейауту [27].

«CombineCocoa» - надає «Combine Publisher» для звичайних елементів керування UIKit [28].

3.1.9 «Realm» SDK

«Realm» це об'єктно-орієнтована база даних, спроектована для ефективного зберігання та обробки даних. Основна особливість Realm полягає в тому, що вона надає швидкий та простий спосіб роботи з локальними базами даних для розробників мобільних додатків [29].

«Realm» пропонує простий API для виконання основних операцій з базою даних, таких як додавання, видалення та зміна записів.

База даних «Realm» відома своєю високою швидкістю завдяки використанню технік, таких як нативна компіляція та індексація даних.

Для більш складних сценаріїв, «Realm» також пропонує сервіс «Realm Sync» для автоматичної синхронізації даних між пристроями та серверами.

3.2 Розробка додатку

Під час розробки були прийняті наступні рішення:

– весь UI розробляти програмно за допомогою бібліотеки «SnapKit»;

– керувати залежностями в додатку за допомогою бібліотеки «Dependencies».

3.2.1 Структура проєкту

Під час розробки була сформована наступна структура проєкту (див. рис. 3.2).

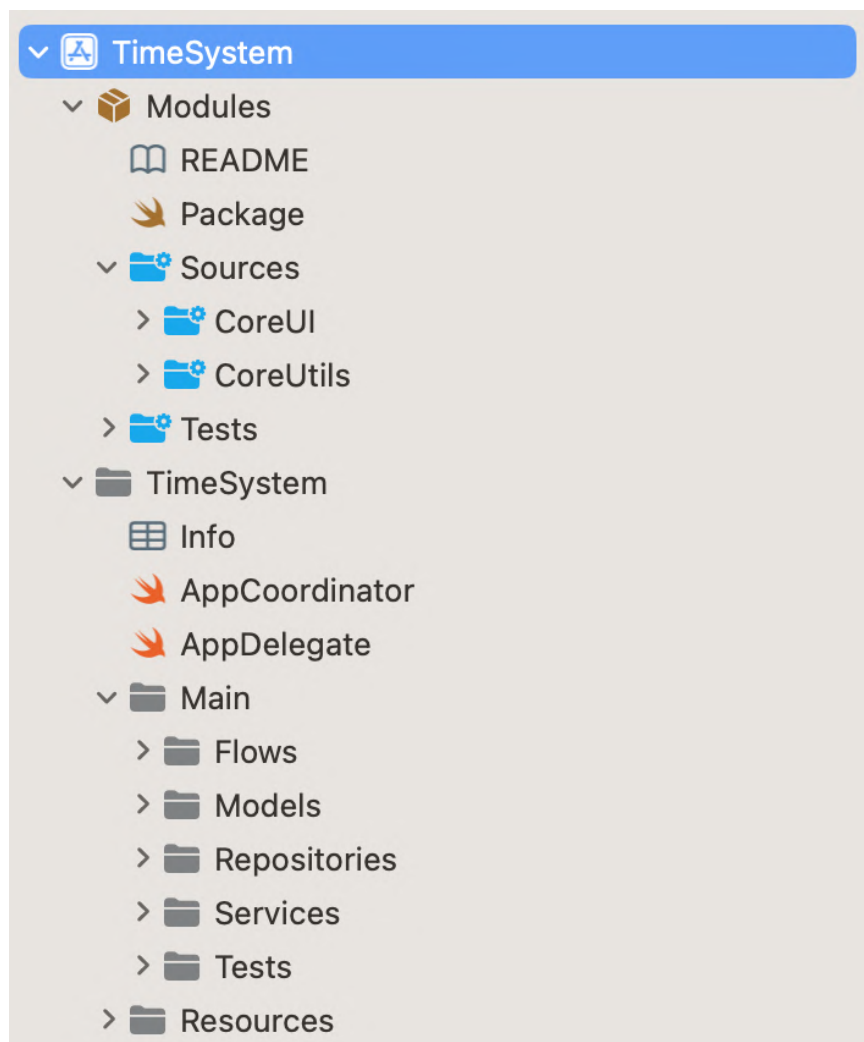


Рисунок 3.2 – Структура проєкту

Структура проєкта складається з двох основних частин: «Modules» та «TimeSystem».

«Modules» - це «Swift Package», який надає доступ до допоміжних пакетів, які використовуються в розробці додатку (див. рис. 3.3).

Swift Package - це пакетна система для Swift, яка дозволяє управляти залежностями та створювати переносимі пакети коду для використання в проєктах Swift.

```

4 import PackageDescription
5
6 let package = Package(
7     name: "Modules",
8     defaultLocalization: "en",
9     platforms: [.iOS(.v14)],
10    products: [
11        // Products define the executables and libraries a package produces, and make them visible to other packages.
12        .library(name: "CoreUI", targets: ["CoreUI"]),
13        .library(name: "CoreUtils", targets: ["CoreUtils"]),
14    ],
15    dependencies: [
16        // Dependencies declare other packages that this package depends on.
17        // .package(url: /* package url */, from: "1.0.0"),
18    ],
19
20    // MARK: - Targets
21    // Targets are the basic building blocks of a package. A target can define a module or a test suite.
22    // Targets can depend on other targets in this package, and on products in packages this package depends on.
23    targets: [
24        .target(
25            name: "CoreUI",
26            dependencies: [],
27        ),
28        .target(
29            name: "CoreUtils",
30            dependencies: [],
31        )
32    ]
33 )

```

Рисунок 3.3 – Реалізація пакету «Modules»

«Modules» надає доступ до двох таргетів: «CoreUI» та «CoreUtils».

«CoreUI» зберігає файли, які відповідають за реалізацію елементів визначених дизайн системою (див. рис. 3.4):

- «Buttons». Зберігає файли реалізації кнопок;
- «Views». Зберігає файли реалізації перевикористовуємії «UI» елементів;
- «Color». Файл, який надає доступ до всіх кольорів, які використовуються в додатку;
- «Font». Файл, який надає доступ до всіх шрифтів, які використовуються в додатку.

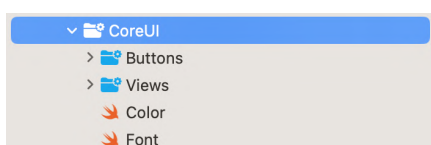


Рисунок 3.4 – Структура таргету «CoreUI»

«CoreUtils» зберігає файл «Utils», який зберігає реалізацію допоміжних функцій, які допомагають спростити реалізацію функціоналу додатку.

«TimeSystem» відповідає за реалізацію основної частини додатку, яка включає в себе:

- «Info.plist». Відповідає за налаштування конфігурацій проєкту;
- «AppCoordinator.swift» (див. рис. 3.5) - файл, який зберігає реалізацію головного координатора додатку;
- Файл «AppDelegate.swift» (див. рис. 3.6) - файл, з якого починається робота додатку. Реалізує конфігурацію початкового екрана додатку;
- «Main». Зберігає основні файли реалізації системи (див. рис. 3.7):
- «Flows». Зберігає файли, які відповідальні за реалізацію екранів додатку (див. рис. 3.8):
 - «Helper». Зберігає файли, які відповідальні за реалізацію екрана «Помічник»;
 - «List». Зберігає файли, які відповідальні за реалізацію екрана «Список»;
 - «Lists». Зберігає файли, які відповідальні за реалізацію екрана «Нове завдання»;
 - «Planning». Зберігає файли, які відповідальні за реалізацію екрана «Планування»;
 - «Tags». Зберігає файли, які відповідальні за реалізацію екрана «Ярлики»;
 - «Task». Зберігає файли, які відповідальні за реалізацію екранів «Нове завдання» та «Деталі завдання»;
 - «Models». Зберігає файли, які відповідальні за реалізацію моделей сутностей системи (див. рис. 3.9):
 - «Task.swift». Відповідає за реалізацію моделі сутності «Завдання»;
 - «Priority.swift». Відповідає за реалізацію моделі сутності «Пріоритет»;
 - «User.swift». Відповідає за реалізацію моделі сутності «Користувач»;
 - «List.swift». Відповідає за реалізацію моделі сутності «Список»;
 - «Period.swift». Відповідає за реалізацію моделі сутності «Період»;
 - «Tag.swift». Відповідає за реалізацію моделі сутності «Ярлик»;

- «Repositories». Зберігає файли, які відповідають за реалізацію підсистем керування сутностями системи (див. рис. 3.10):
 - «ListsRepository.swift». Відповідає за реалізацію підсистеми керування сутностями «Список»;
 - «TagsRepository.swift». Відповідає за реалізацію підсистеми керування сутностями «Ярлик»;
 - «TasksRepository.swift». Відповідає за реалізацію підсистеми керування сутностями «Завдання»;
 - «UserRepository.swift». Відповідає за реалізацію підсистеми керування сутностями «Користувач»;
 - «PrioritiesRepository.swift». Відповідає за реалізацію підсистеми керування сутностями «Пріоритети»;
 - «PeriodsRepository.swift». Відповідає за реалізацію підсистеми керування сутностями «Період»;
- «Services». Зберігає файли які відповідають за реалізацію підсистем (див. рис. 3.11):
 - «DatabaseService.swift». Відповідає за реалізацію підсистеми «Керування базою даних». Має свій набір реалізації моделей сутностей системи які доступні для розуміння бази даних (Realm);
 - «HelperService.swift». Відповідає за реалізацію підсистеми «Помічник»;
 - «Tests». Зберігає файли які відповідають за реалізацію тестових сценаріїв, за допомогою який перевіряється коректна реалізація бізнес-логіки системи;
 - «DatabaseServiceTests.swift». Відповідає за реалізацію тестів для «DatabaseService.swift»;
 - «HelperServiceTests.swift». Відповідає за реалізацію тестів для «HelperService.swift»;
- «Resources». Зберігає файли які відповідають за надання доступу до таких ресурсів як «Картинки» «Кольори», «Шрифти», які використовуються в додатку.
- «Assets». Зберігає графічні матеріали додатку;
- «Color». Зберігає кольори які визначені дизайн системою.

```

1  import Foundation
2  import UIKit
3
4  @MainActor
5  public final class AppCoordinator {
6
7      private let window: UIWindow
8      private var navigation: UINavigationController = UINavigationController()
9
10     public init(with window: UIWindow) {
11         self.window = window
12         window.rootViewController = navigation
13         window.makeKeyAndVisible()
14     }
15
16     public func start() {
17         PlanningFlowCoordinator(with: navigation).start()
18         window.makeKeyAndVisible()
19     }
20 }
21
22
23 fileprivate final class NavigationController: UINavigationController {
24     override func viewDidLoad() {
25         super.viewDidLoad()
26         self.setNavigationBarHidden(true, animated: false)
27     }
28 }

```

Рисунок 3.5 – Реалізація файлу «AppCoordinator.swift»

```

1  import Foundation
2  import UIKit
3
4  @UIApplicationMain
5  class AppDelegate: UIResponder, UIApplicationDelegate {
6      var window: UIWindow?
7      var coordinator: AppCoordinator!
8
9      func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions:
10         [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
11         window = UIWindow(frame: UIScreen.main.bounds)
12         self.coordinator = AppCoordinator(with: window!)
13         coordinator.start()
14         return true
15     }
16 }

```

Рисунок 3.6 – Реалізація файлу «AppDelegate.swift»

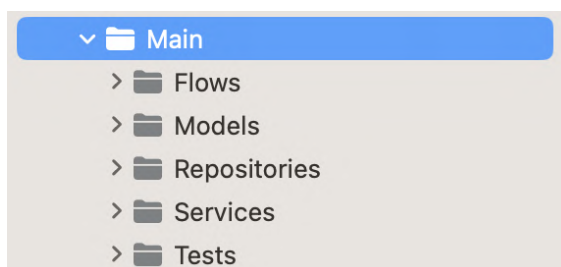


Рисунок 3.7 – Структура «Main»

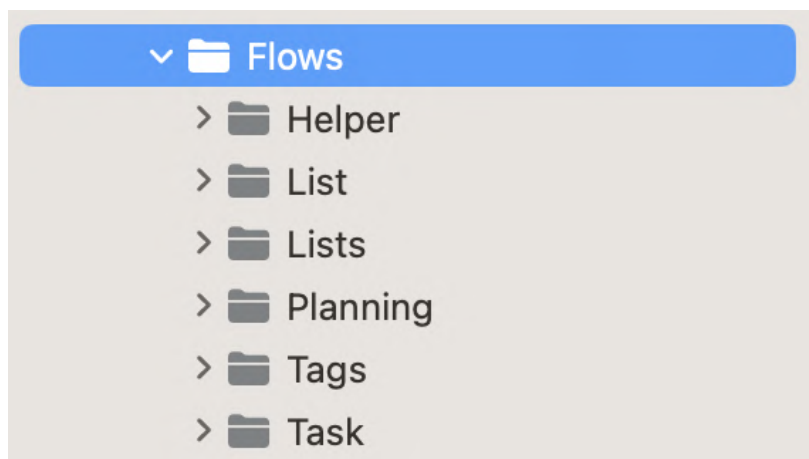


Рисунок 3.8 – Структура «Flows»

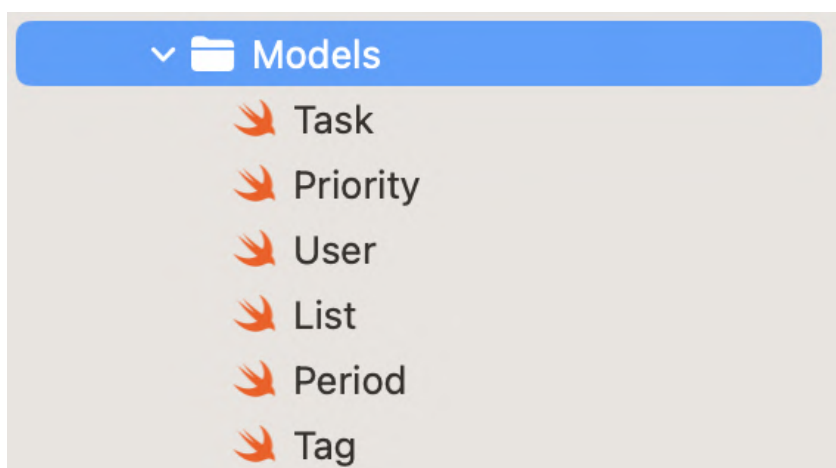


Рисунок 3.9 – Структура «Models»

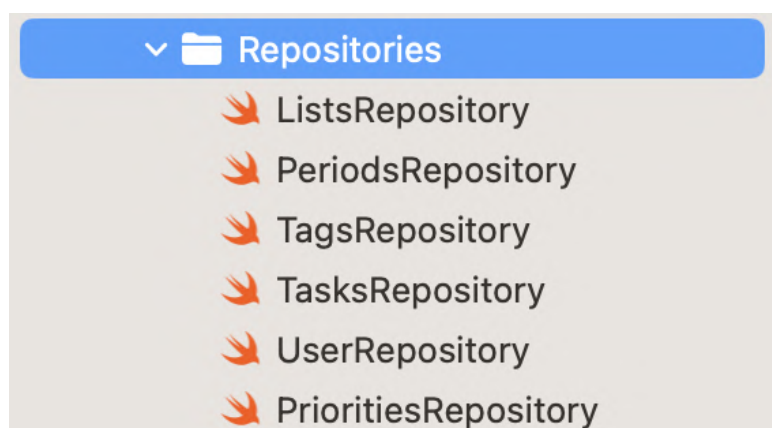


Рисунок 3.10 – Структура «Repositories»

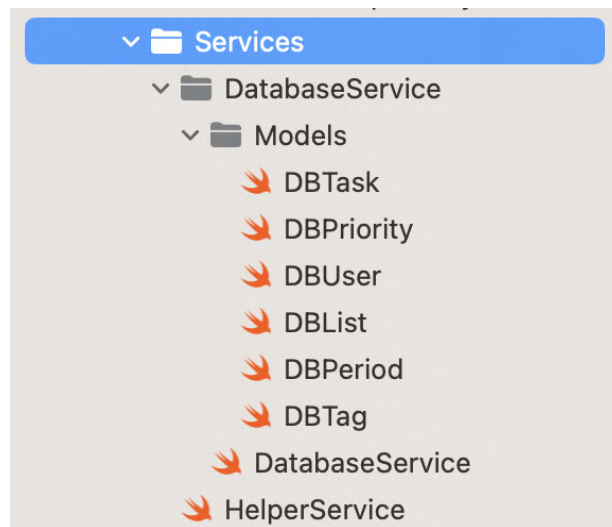


Рисунок 3.11 – Структура «Repositories»

3.3 Висновки до розділу

В цьому розділі було розглянуто обрані засоби розробки, такі як:

- середовище розробки «Xcode»;
- мова програмування Swift;
- фреймворк: «Foundation», «UIKit», «Combine», «XCTest»;
- бібліотеки: «Dependency», «SnapKit», «CombineCocoa»;
- база даних «Realm SDK»;

Було розглянуто структуру проєкту розробленого додатку.

4 РЕЗУЛЬТАТИ РОБОТИ ДОДАТКУ

Результатами реалізації системи є зроблені скріншоти, які відображають реалізований інтерфейс користувача.

Перший екран, який бачить користувач при вході в систему - це екран «Планування» (див. рис. 4.1). Відображає, який сьогодні день та поточний час. Також відображає визначені періоди для поточного дня та завдання. Надає інтерфейс налаштування періодів та завдань. Надає інтерфейс переходу на інші екрани.

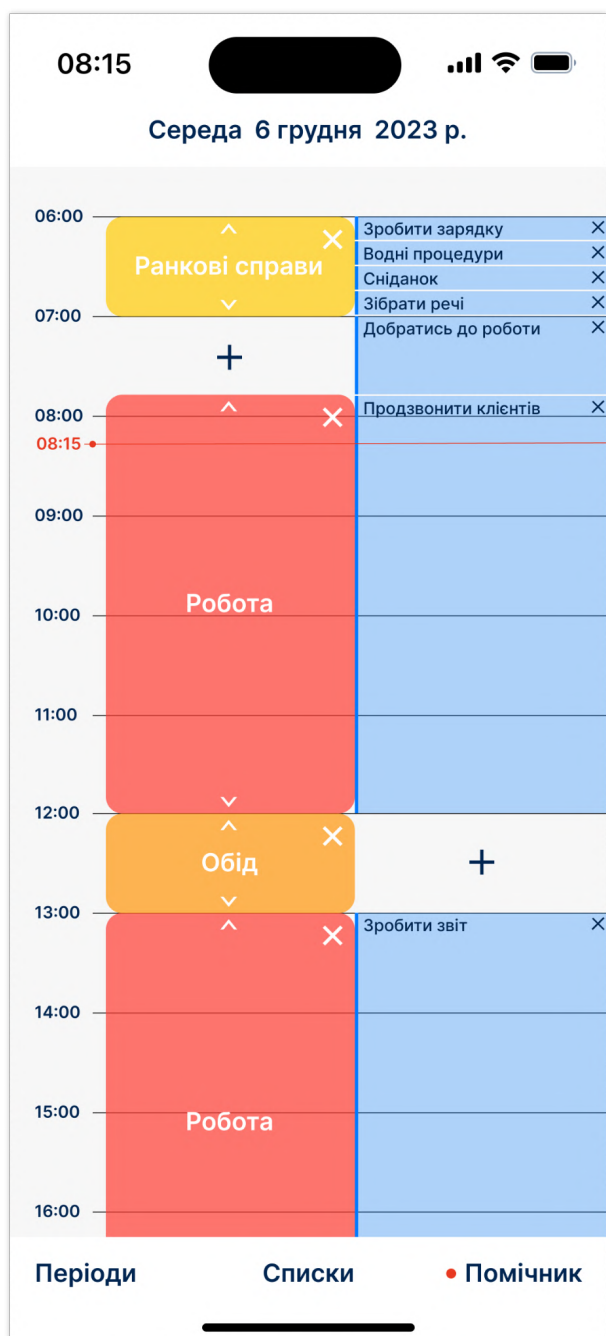


Рисунок 4.1 – Екран «Планування»

Якщо на екрані «Планування» натиснути кнопку «Періоди», відкриється екран «Періоди» (див. рис 4.2), який відображає всі періоди, які були створені в системі та надає можливість видаляти або додавати періоди.

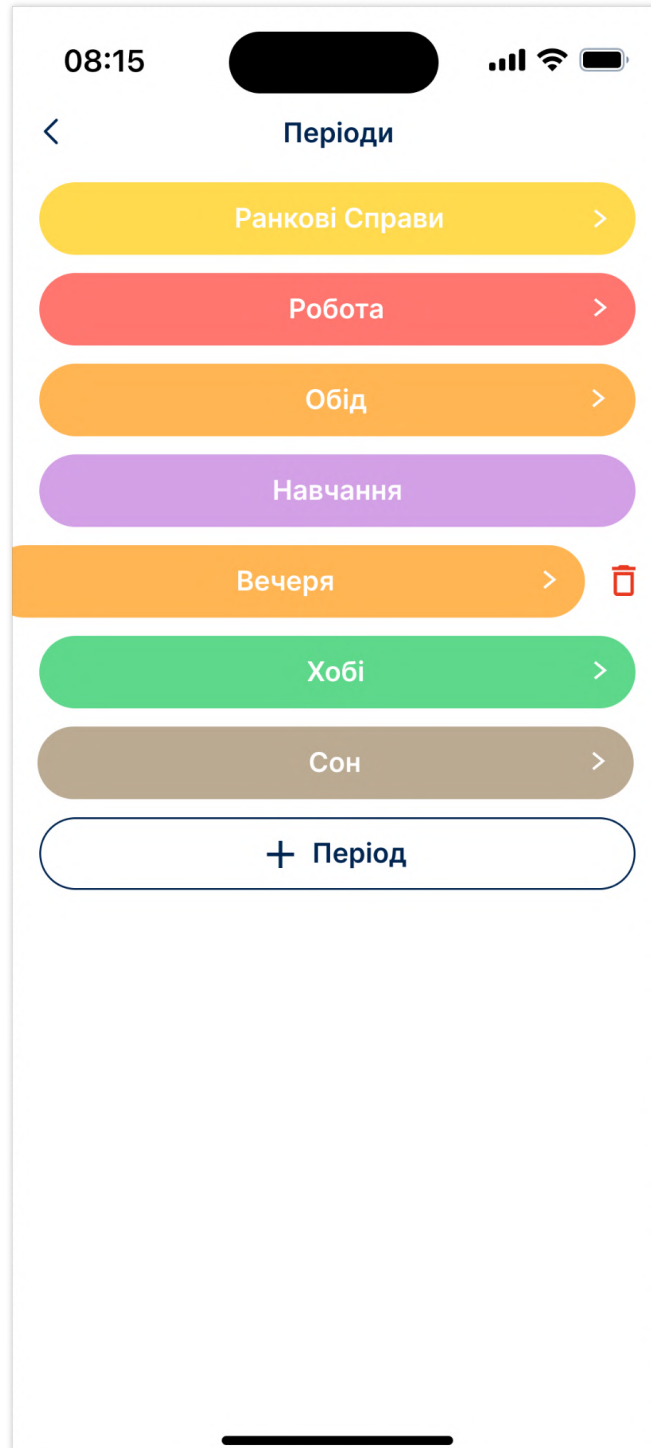


Рисунок 4.2 – Екран «Періоди»

Якщо на екрані «Періоди» натиснути кнопку «+ Період», тоді починається процес створення нового періоду (див. рис. 4.3).

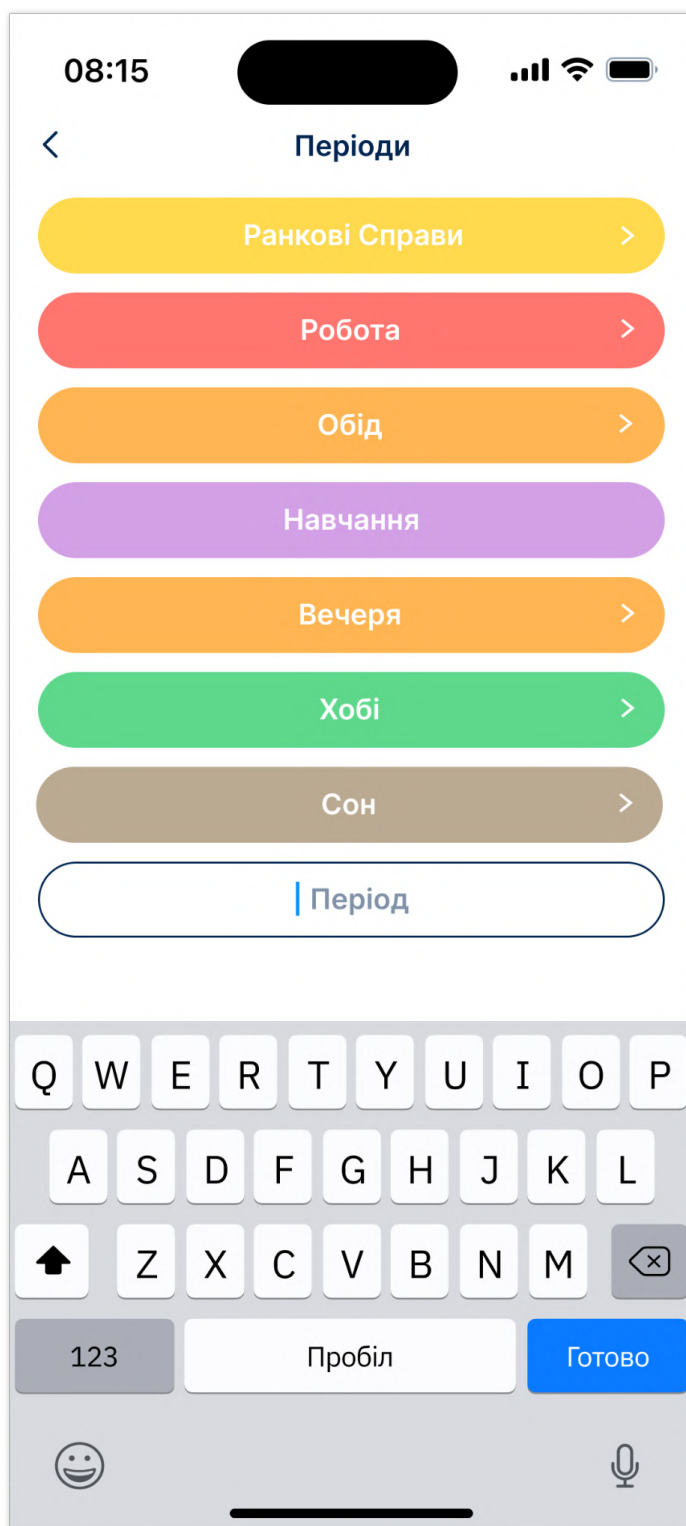


Рисунок 4.3 – Екран «Періоди», створення нового періоду

Якщо обрати один із періодів, обраний період трансформується в стан налаштування періоду (див. рис. 4.4).

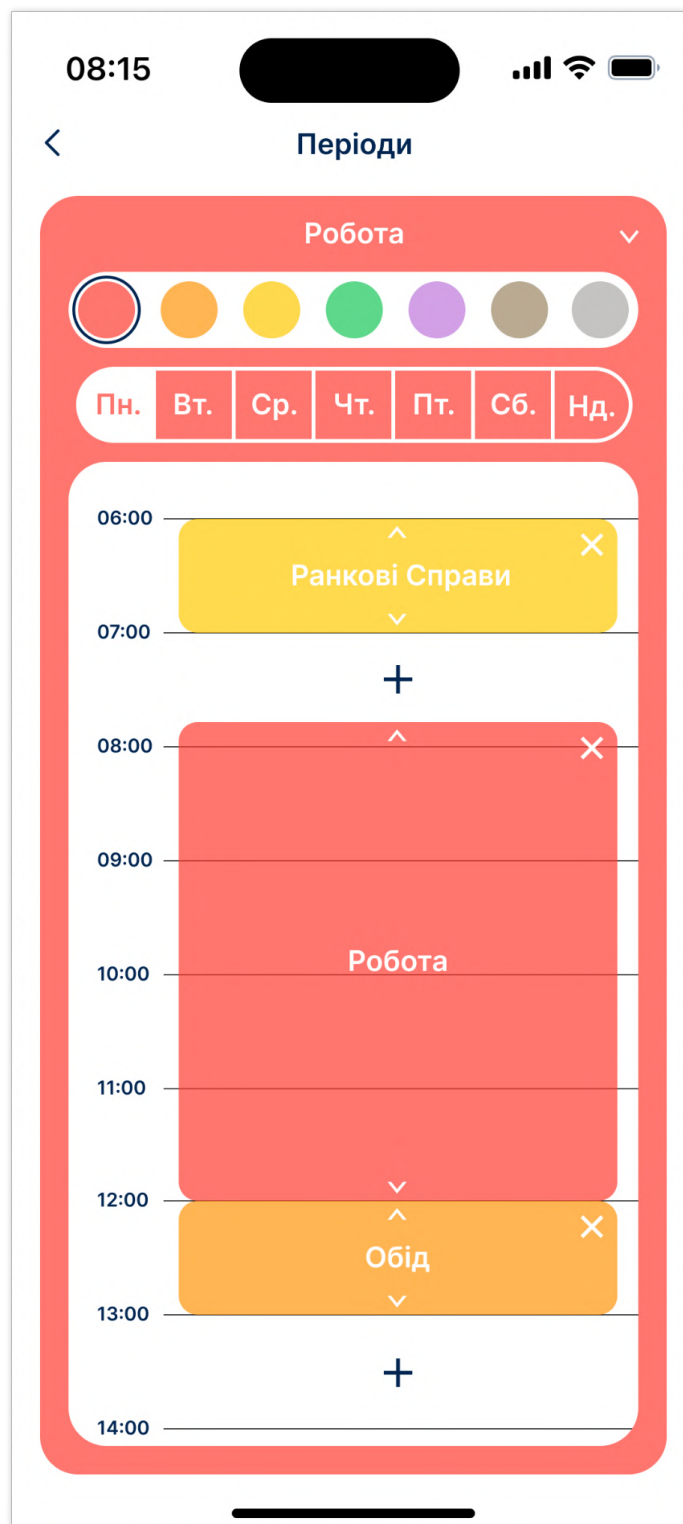


Рисунок 4.4 – Екран «Періоди», режим налаштування періоду

Якщо на екрані «Планування» натиснути кнопку «Списки», відкриється екран «Списки» (див. рис. 4.5), який відображає всі списки які були створені в системі. Надає можливість додавати або видаляти списки.

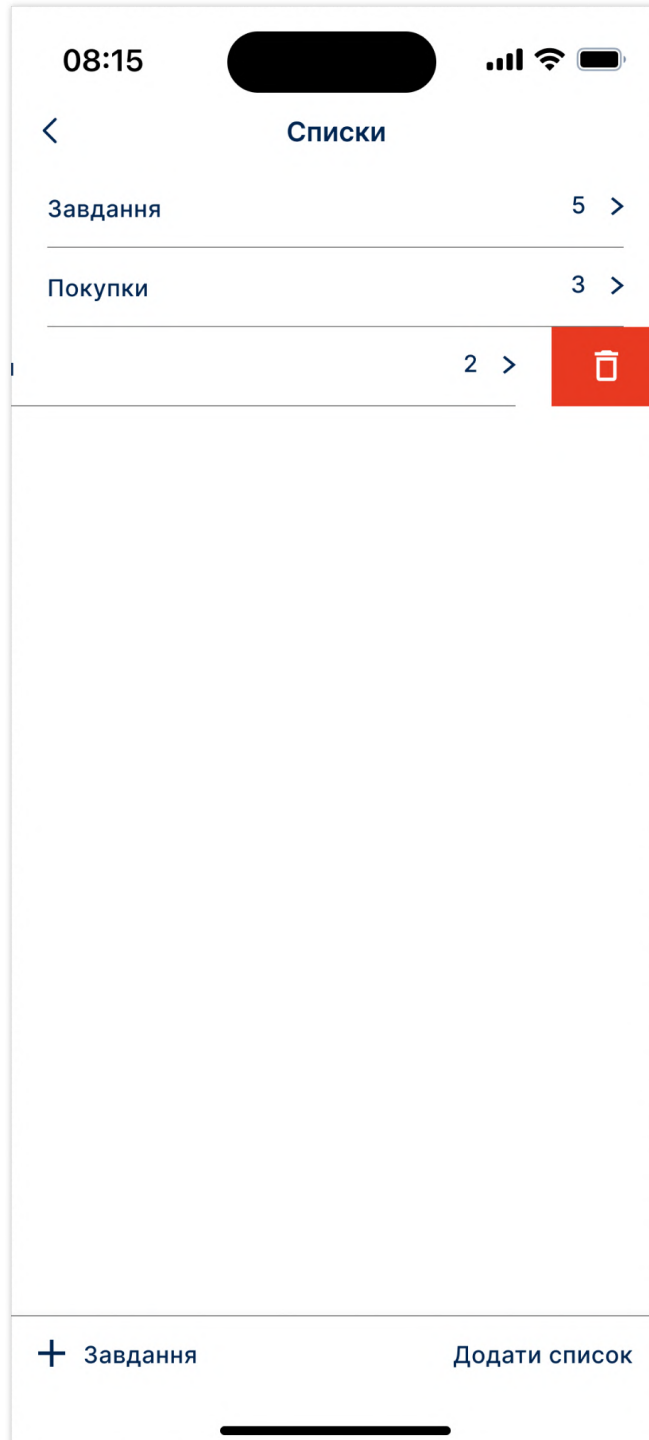


Рисунок 4.5 – Екран «Списки»

Якщо на екрані «Списки» натиснути кнопку «Додати список» то почнеться процес створення нового списку (див. рис. 4.6).

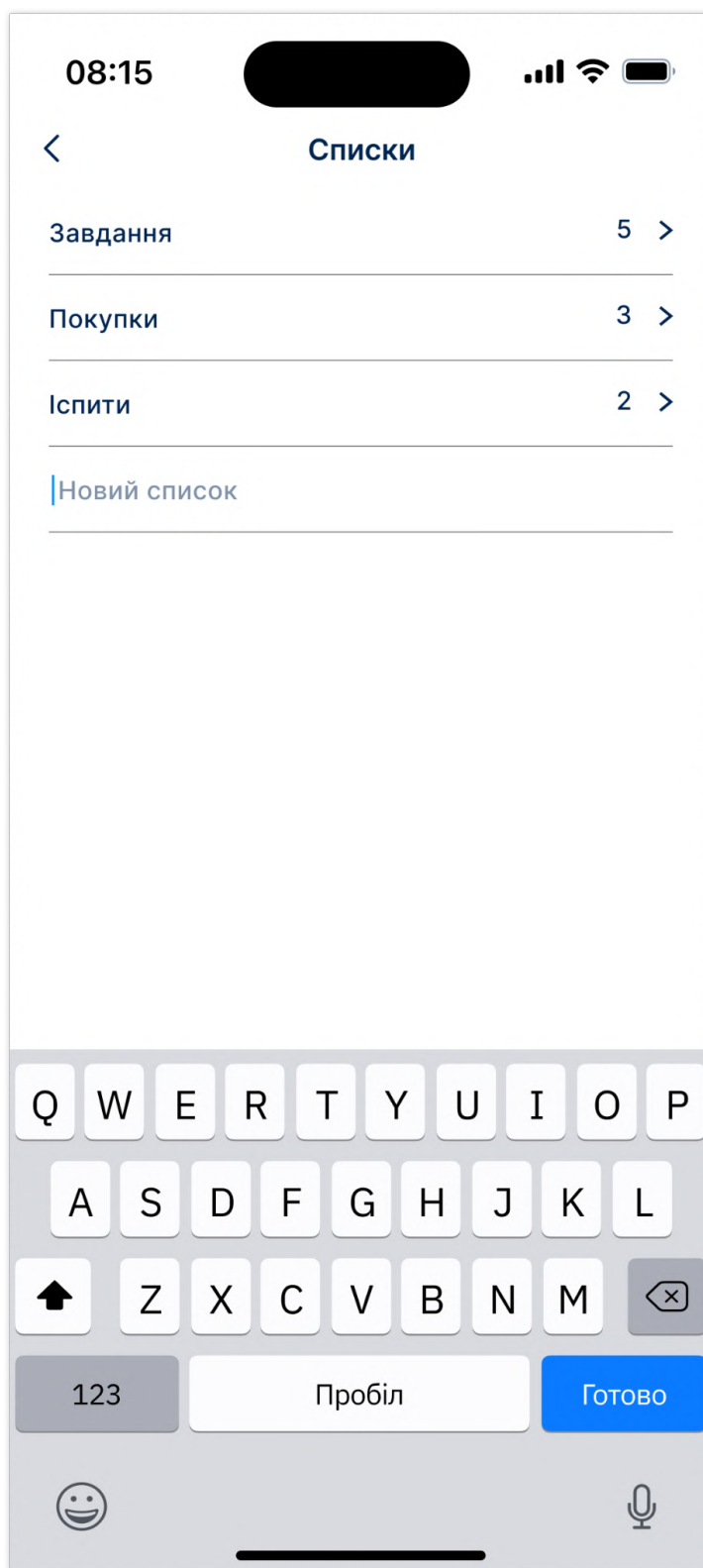


Рисунок 4.6 – Екран «Списки», створення нового списку

Якщо на екрані «Списки» обрати список із переліку, тоді відкриється екран «Список» (див. рис. 4.7), який відображає які завдання належать до обраного списку.

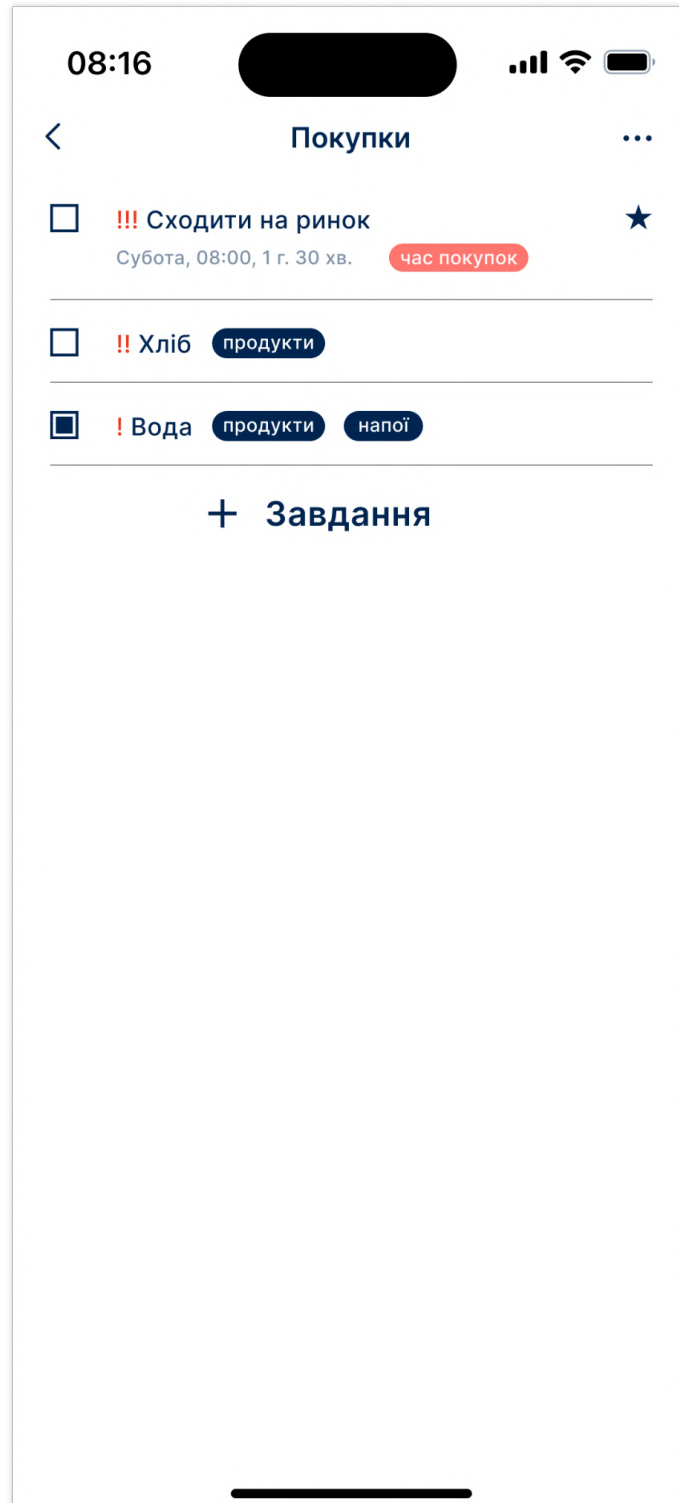


Рисунок 4.7 – Екран «Список»

Якщо натиснути на назву списку почнеться процес зміни назви списку (див. рис. 4.8).

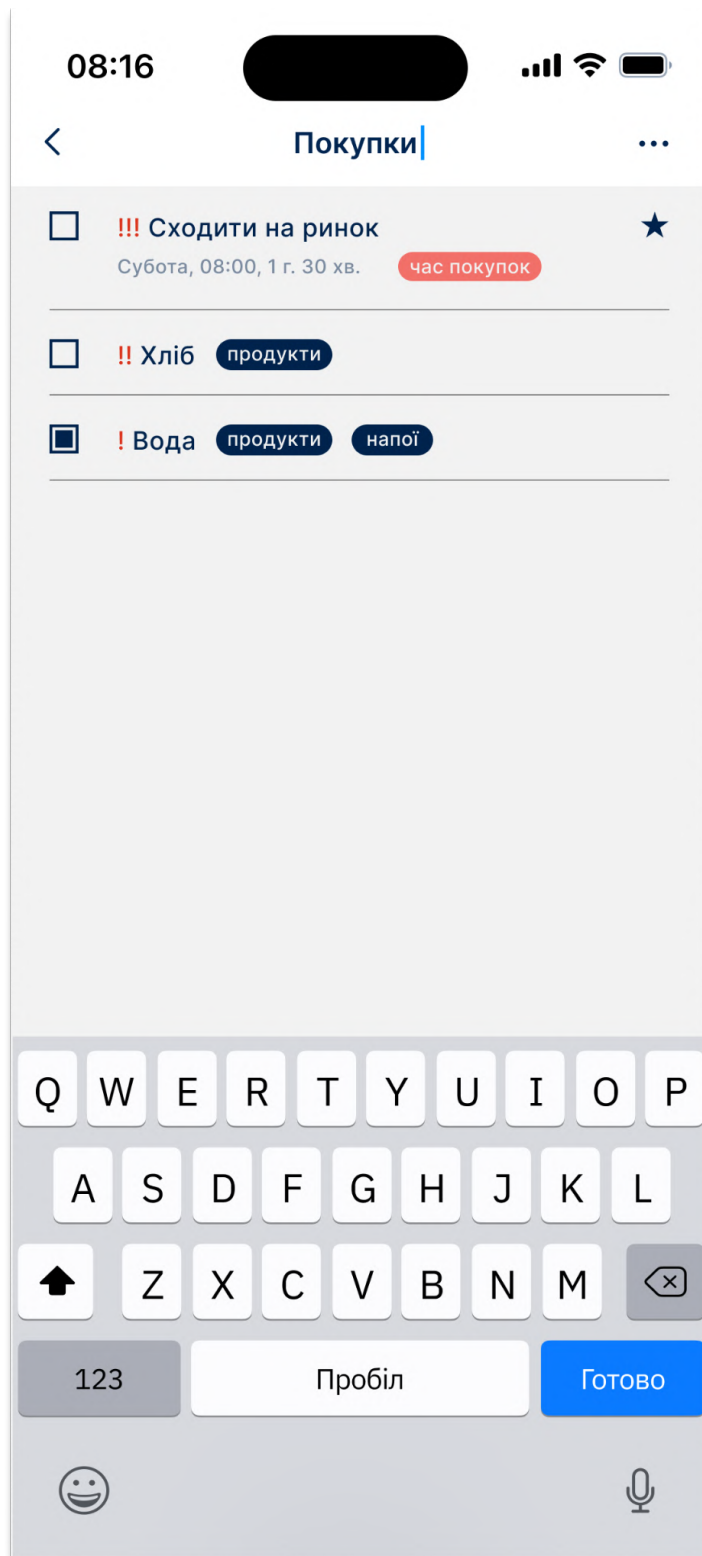


Рисунок 4.8 – Екран «Список», зміна назви

Якщо на екрані «Список» натиснути кнопку налаштувань відкриється елемент налаштувань списку (див. рис. 4.9).

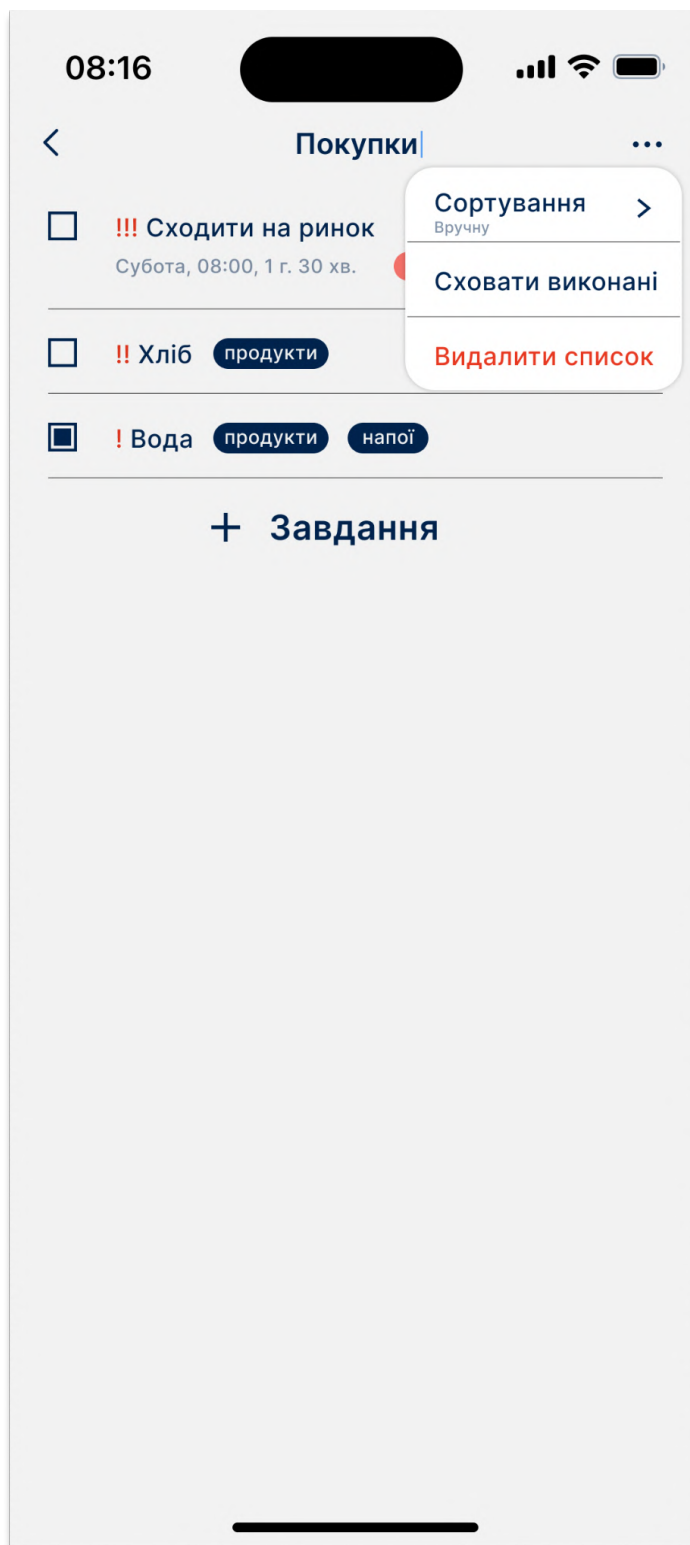


Рисунок 4.9 – Екран «Список», меню налаштувань

Якщо обрати розділ налаштувань «Сортування», тоді відобразяться можливі варіанти сортування за якими можна змінити сортування завдань в списку (див. рис. 4.10).

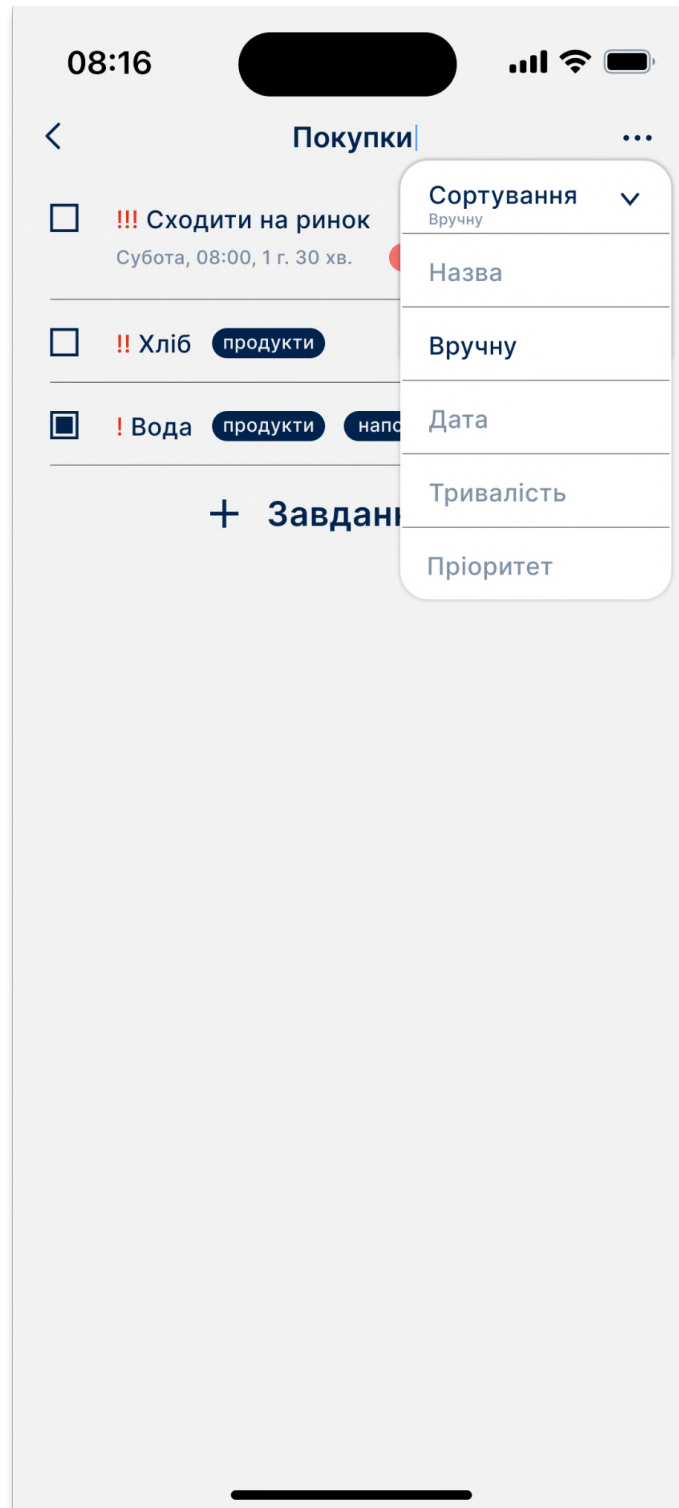
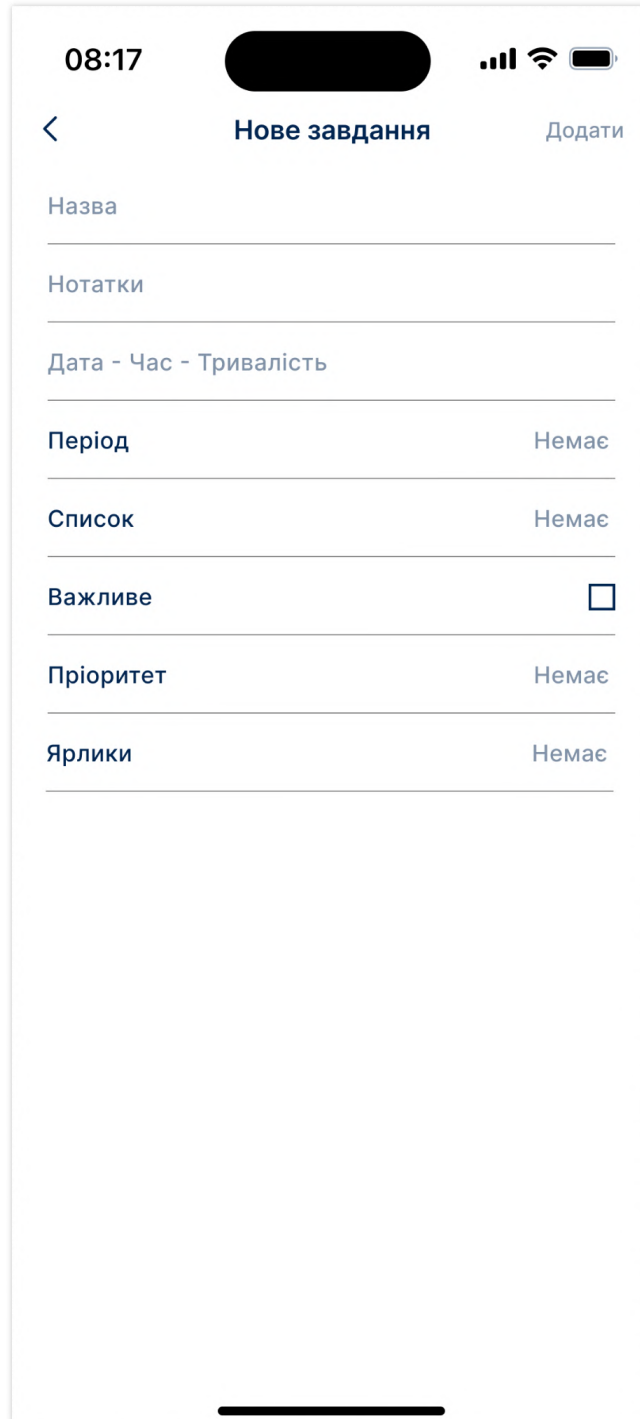


Рисунок 4.10 – Екран «Список», меню вибору типу сортування завдань

Якщо на екрані «Планування» або екрані «Списки» або екрані «Список» обрати створення завдання, тоді відобразиться екран «Нове завдання» (див. рис. 4.11), який надає можливість створити нове завдання в системі.



08:17

< Нове завдання Додати

Назва

Нотатки

Дата - Час - Тривалість

Період Немає

Список Немає

Важливе ☐

Пріоритет Немає

Ярлики Немає

Рисунок 4.11 – Екран «Нове завдання»

Екран «Деталі завдання» (див. рис. 4.12), надає можливість переглянути деталі завдання та змінити налаштування завдання.

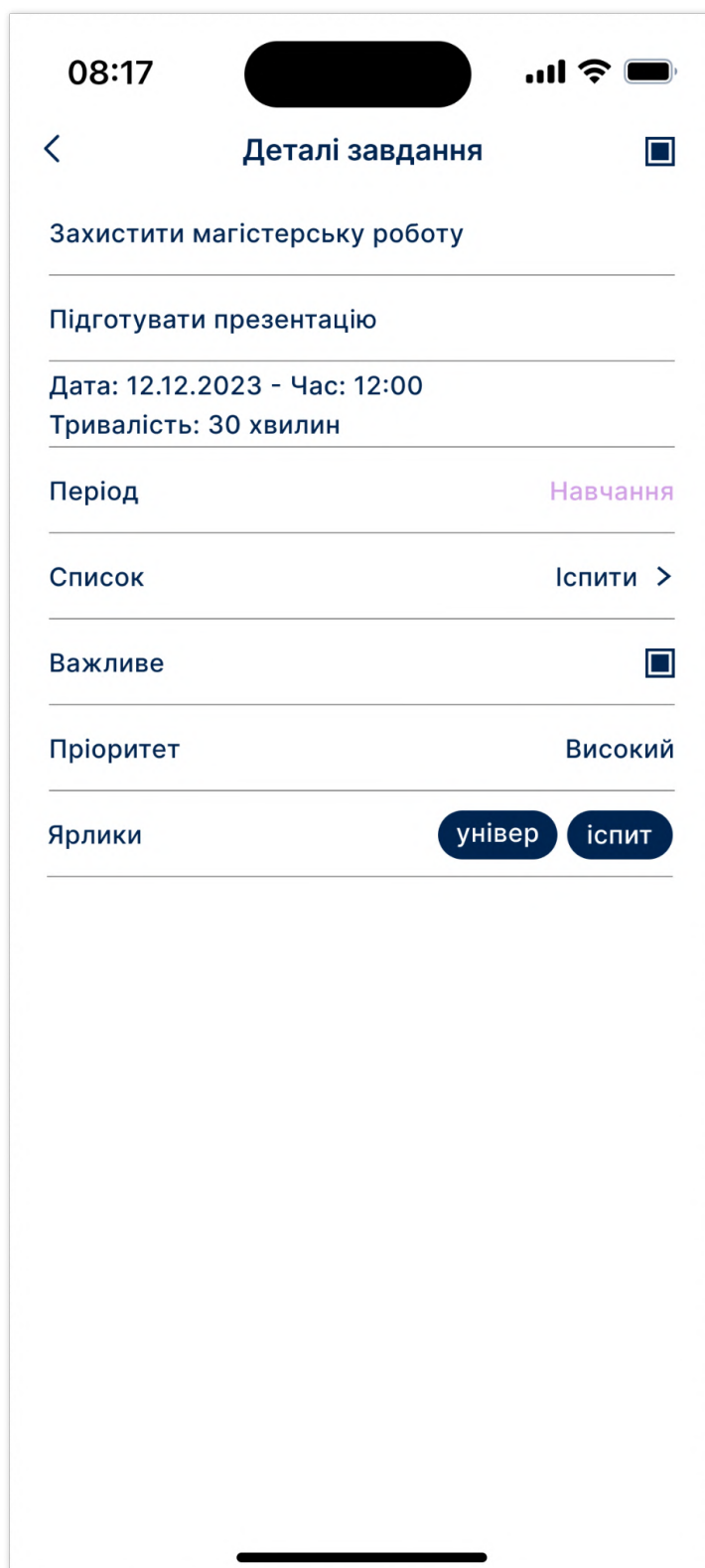


Рисунок 4.12 – Екран «Деталі завдання»

Екран «Ярлики» (див. рис. 4.13), надає можливість обрати ярлики які будуть належати до завдання яке редагується. Також надається можливість додати нові ярлики або видалити існуючі.

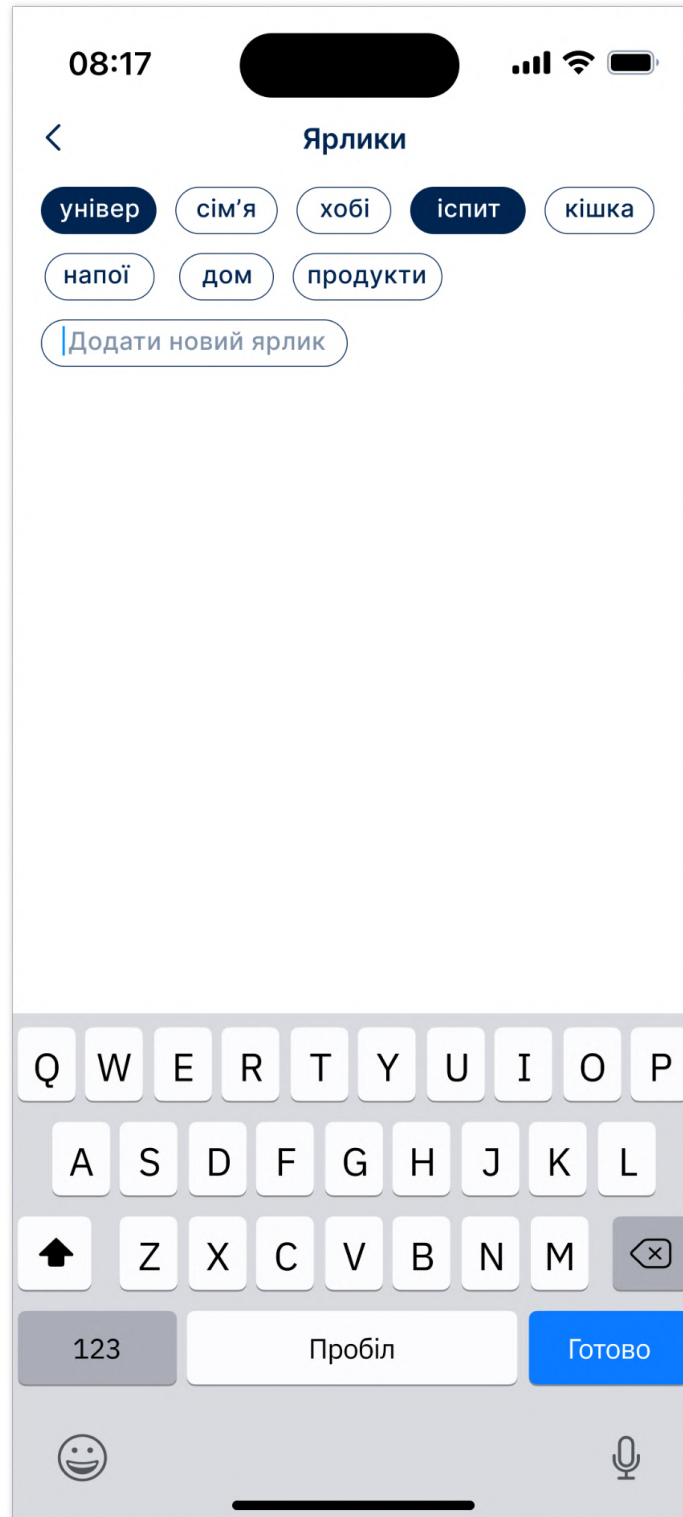


Рисунок 4.13 – Екран «Ярлики»

ВИСНОВКИ

У магістерській роботі було досліджено існуючі методи планування часу. В результаті проведеного дослідження, з'ясовано що розглянуті системи планування часу («Календар», «Нагадування», «Trello», «Fibery») пропонують багато можливостей для зручного та ефективного керування часом з багатьма функціональними можливостями.

Також було проведено опитування і за результатами опитування зроблена порівняльна таблиця. В результаті проведеного аналізу отриманих результатів з'ясовано, що сучасні системи планування часу мають певні недоліки.

Проаналізувавши недоліки існуючих систем планування часу, було запропоновано вимоги для створення вдосконаленої системи планування часу, які були використані при проектуванні власної системи.

Під час проектування було виявлено сутності («Користувач», «Завдання», «Список», «Період», «Ярлик», «Пріоритет») якими оперує система.

Далі було спроектовано екрани («Планування», «Періоди», «Списки», «Список», «Нове завдання», «Деталі завдання», «Ярлики»,) додатку якими оперує користувач.

Після цього було спроектовано підсистеми («Підсистема керування базою даних», «Підсистема помічник», «Підсистема керування завданнями», «Підсистема керування списками», «Підсистема керування періодами», «Підсистема керування ярликами», «Підсистема керування користувачем») додатку.

На завершальному етапі проектування було обрано архітектуру додатку (MVVM+Coordinators) яка була використана при розробці додатку.

Після того як було спроектовано систему, було розпочато етап реалізації системи.

Для реалізації системи були розглянуті і обрані засоби розробки:

– середовище розробки «Xcode»;

- мову програмування «Swift»;
- систему контролю версій «Git»;
- фреймворк «Foundation.framework»;
- фреймворк «UIKit.framework»;
- фреймворк «Combine.framework»;
- фреймворк «XCTest»;
- стороння бібліотека «Dependencies»;
- стороння бібліотека «CombineCocoa»;
- стороння бібліотека «SnapKit»;
- «Realm» SDK.

Система була розроблена як iOS-додаток.

Результатами виконаної роботи є скріншоти інтерфейсу розробленого додатку, на яких відображається інтерфейс спроектованих екранів.

Таким чином мета магістерського проєкту була досягнута.

Всі поставлені завдання було виконано:

- проведено аналіз найбільш популярних систем планування часу;
- визначено, які із використаних підходів, що дають переваги та зменшують недоліки подібних систем, будуть використані при розробці власної вдосконаленої системи планування часу;
- розроблено структуру та алгоритм роботи системи відповідно до висунутих вимог;
- реалізовано систему планування часу;
- проаналізовано отримані результати.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Куликовська Н. А. Візуалізація даних / Н. А. Куликовська, А. І. Іванов // Тиждень науки-2023. Факультет комп'ютерних наук і технологій : Тези доп. науково-техн. конф., Запоріжжя, 24–28 квіт. 2023 р. – Запоріжжя, 2023. – С. 32–33. URL: https://zp.edu.ua/uploads/dept_s&r/2023/conf/4.1/TN-2023-FKNT.pdf (дата звернення: 01.12.2023).
2. Time Management Statistics (New Research in 2022). URL: <https://www.timewatch.com/blog/time-management-statistics-in-2022/> (Last accessed: 01.12.2023).
3. App Store Preview. Calendar. URL: <https://apps.apple.com/us/app/calendar/id1108185179> (Last accessed: 01.12.2023)
4. Посібник користувача iCloud. Використання програми «Календар» на сайті iCloud.com. URL: <https://support.apple.com/uk-ua/guide/icloud/mmd67283e4/icloud> (дата звернення: 01.12.2023).
5. Apple Calendar. URL: <https://www.techradar.com/reviews/apple-calendar> (Last accessed: 01.12.2023).
6. App Store Preview. Reminders. URL: <https://apps.apple.com/us/app/reminders/id1108187841> (Last accessed: 01.12.2023).
7. Apple reminders review – best to-do app for those in the apple ecosystem? URL: <https://jessicabryson.com/apple-reminders-review-best-to-do-app-for-those-in-the-apple-ecosystem/> (Last accessed: 01.12.2023).
8. Trello. URL: <https://trello.com/> (Last accessed: 01.12.2023).
9. Fibery. URL: <https://fibery.io/> (Last accessed: 01.12.2023).
10. Top 10 Selection Criteria To Help You Compare ERP Systems. URL: <https://www.aptean.com/en-US/insights/blog/compare-erp-systems-selection-criteria> (Last accessed: 01.12.2023).

11. How do you compare different information systems? URL: <https://www.linkedin.com/advice/0/how-do-you-compare-different-information-systems> (Last accessed: 01.12.2023).
12. Оцінка систем для керування завданнями та часом. URL: <https://docs.google.com/forms/d/1TRmoHILjn0fWtAqwoGs3sb5p4M9HPVI3SLXfVmPBFYo> (Last accessed: 01.12.2023).
13. MEYR, Herbert; WAGNER, Michael; ROHDE, Jens. Structure of advanced planning systems. Supply chain management and advanced planning: concepts, models, software, and case studies, 2015, P. 99-106.
14. GOMAA, Hassan. Structuring criteria for real time system design. In: Proceedings of the 11th international conference on Software engineering. 1989. P. 290-301.
15. HUNDAL, M. S. A systematic method for developing function structures, solutions and concept variants. Mechanism and Machine theory, 1990, 25.3: P. 243-256.
16. Human Interface Guidelines. URL: <https://developer.apple.com/design/human-interface-guidelines> (Last accessed: 01.12.2023).
17. WARFIELD, John N. Developing subsystem matrices in structural modeling. IEEE Transactions on Systems, Man, and Cybernetics, 1974, 1:P. 74-80.
18. MVVM+Coordinators in Swift. URL: <https://medium.com/@hrzjanati/mvvm-coordinators-in-swift-e383a77c2802> (Last accessed: 01.12.2023).
19. Apple Developer. Xcode 15. URL: <https://developer.apple.com/xcode/> (Last accessed: 01.12.2023).
20. Swift programming language. URL: <https://www.swift.org/> (Last accessed: 01.12.2023).
21. Git. URL: <https://git-scm.com/> (Last accessed: 01.12.2023).
22. Apple Developer Documentation. Foundation. URL: <https://developer.apple.com/documentation/foundation> (Last accessed: 01.12.2023).
23. Apple Developer Documentation. UIKit. URL: <https://developer.apple.com/documentation/uikit> (Last accessed: 01.12.2023).

24. Apple Developer Documentation. Combine. URL: <https://developer.apple.com/documentation/combine> (Last accessed: 01.12.2023).
25. Apple Developer Documentation. XCTest. URL: <https://developer.apple.com/documentation/xctest> (Last accessed: 01.12.2023).
26. A dependency management library inspired by SwiftUI's "environment". URL: <https://github.com/pointfreeco/swift-dependencies> (Last accessed: 01.12.2023).
27. GitHub - SnapKit/SnapKit: A Swift Autolayout DSL for iOS & OS X. URL: <https://github.com/SnapKit/SnapKit> (Last accessed: 01.12.2023).
28. GitHub - CombineCommunity/CombineCocoa: Combine publisher bridges for UIKit. URL: <https://github.com/CombineCommunity/CombineCocoa> (Last accessed: 01.12.2023).
29. Realm Swift SDK. URL: <https://www.mongodb.com/docs/realm/sdk/swift/> (Last accessed: 01.12.2023).
30. Іванов А. І. Дослідження систем планування часу : Технологія-2023 : матеріали XXVI міжнар. наук.-практ. конф., 26 травня 2023 р., м. Київ / [уклад. Є. І. Зубцов]. – Київ : Східноукр. нац. ун-т ім. В. Даля, 2023. С. 225. URL: <https://drive.google.com/file/d/1iMR9CTxDEC7tLY6qTrJjKtMbE2iCXeOy/view?usp=sharing> (дата звернення: 01.12.2023).