

ЗМІСТ

ВСТУП	5
1. АНАЛІЗ ІСНУЮЧИХ СИСТЕМ “РОЗКЛАД” ТА	
ПОСТАНОВКА ЗАДАЧІ	7
1.1. Огляд функціоналу існуючої системи “Rozklad KPI” та	
“ KPI Weeks”	7
1.2. Встановлення переваг та недоліків існуючих систем.....	8
1.3. Постановка задачі	10
1.4. Склад та ієрархія задач та підзадач підсистеми	12
1.5. Схема вхідних та вихідних потоків для задач	13
1.6 Розробка вимог.....	14
1.6.1. Вимоги до функціоналу.....	15
1.6.2. Вимоги до інтерфейсу.....	19
1.6.3. Вимоги до архітектури	20
Висновки до розділу	21
2. РОЗРОБКА ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ.....	23
2.1. Структура локальної бази даних	23
Висновки до розділу	25
3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ.....	27
3.1. Вибір середовище та технології розробки	27
3.2. Архітектура додатку.....	38
3.3. Опис розробленого функціоналу	41

КАФЕДРА ТК				ІК11.02.0414.002 ПЗ			
Розроб.	Загорський П.М.			Розробка інформаційного та програмного забезпечення підсистеми Електронного Кампусу "Розклад" з підтримкою мобільних платформ. Розробка додатку для iOS платформ.	Літ.	Арк.	Акрушів
Керівник	Мелкумян К.Ю.						
Консульт.							
Н.контр.							
Зав.кафедри							
				НТУУ «КПІ» ФІОТ зр. ІК-11			

3.4. Особливості побудови мобільного додатку	43
Висновки до розділу	44
4. ВПРОВАДЖЕННЯ ТА ПРАКТИЧНЕ ЗАСТОСУВАННЯ ПІДСИСТЕМИ	46
Висновки до розділу	59
5. ОХОРОНА ПРАЦІ.....	60
5.1. Характеристика об'єкту та умови його експлуатації	61
5.1.1. Мікроклімат робочої зони.....	63
5.1.2. Освітлення	64
5.1.3. Виробничий шум	65
5.1.4. Виробничі випромінювання	67
5.1.5. Пожежна безпека	68
5.2. Інструкції з техніки безпеки	69
Висновки до розділу	71
ВИСНОВКИ ДО РОБОТИ	72
ПЕРЕЛІК ПОСИЛАНЬ	74

ВСТУП

Швидкий розвиток ІТ у світі зумовлює ріст популярності впровадження інформаційних систем у різні сфери діяльності людини. Навчання у ВНЗ не є винятком, а навіть навпаки, це чи не найголовніший і найцікавіший об'єкт для подібних впроваджень. В НТУУ «КПІ» вже ведеться розробка автоматизованої інформаційної системи «Електронний кампус».

«Електронний кампус» включає в себе багато різних аспектів інформаційної системи у вигляді окремих підсистем. Таким чином, тут реалізовані підсистеми дистанційного навчання, а також автоматизації організації навчального процесу.

Останній пункт є цільовою задачею в розробці, адже включає в себе трансформації всієї аналогової інформації по навчальному плану у інформаційний вигляд та її автоматизацію. Такі дії підвищать зручність роботи, зменшать кількість помилок, забезпечать доступ у будь-який час і відкриють нові можливості для ведення статистики даних в реальному часі.

Завданням даного дипломного проекту є розробка підсистеми «Розклад» для автоматизації та управління навчальним процесом з підтримкою мобільних платформ (Android та IOS). Впровадження та використання даної підсистеми дозволить:

- Отримати доступ до потрібної інформації з вашого телефона у будь який зручний для користувача час.
- Забезпечити зручну роботу з даними розкладу як для студентів, так і для викладачів.
- Надавати завжди коректні дані у реальному часі.
- Ефективно здійснювати оповіщення студентів та викладачів про зміни у навчальному розкладі у реальному часі.
- Інтегруватись у вже існуючу систему «Електронний кампус».

- Надати можливість додавання нового функціоналу з часом.

На жаль, навіть с такою кількістю переваг подібні підсистеми мають також певні недоліки, такі як: неможливість надання поточної інформації без інтернет зв'язку та прив'язаність до певних платформ. Якщо від першого недоліку позбавитись неможливо, то другий втратить свою актуальність у майбутньому.

Взявши до уваги те, що підсистема «Розклад» з подібним функціоналом немає реалізованих аналогів, то її розробка є актуальною.

					ІК1102.0414.002 ПЗ	Лист
						6
Змн.	Аркуш	№ докум.	Підпис	Дата		

1. АНАЛІЗ ІСНУЮЧИХ СИСТЕМ “РОЗКЛАД” ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Огляд функціоналу існуючої системи “Rozklad KPI” та “KPI Weeks”

Створення відмінного додатку, який буде мати більш розгорнутий функціонал ніж наші попередники, є пріоритетною задачею. Так у AppStore та Play-маркеті у вільному доступі є додатки «Rozklad Kpi» та «KPI Weeks», але знайшовши певні недоліки було вирішено створити схожий додаток але з своїм функціоналом та використовувати для цього новий тип архітектури.

Ці додатки можна вважати таким собі “першопроходцями” у даній темі додатком. Розробник вдосталь попрацював над створенням гарного додатку, але він має певний обмежений функціонал та нараховує ряд недоліків які не дозволяють інтегрувати його в систему дистанційного навчання «Електронного кампуса», тож було вирішено створити новий додаток, який буде синхронізуватись з базою даних «Електронного кампуса» та підтримувати будь яких студентів КПП. В цілому тема та ідея «KPI Weeks» вірна, але для повноти та унікальності системі не вистачає певного функціоналу який не так вже й просто реалізувати.

В даній дипломній роботі буде частково описана частина пов’язана з створення IOS додатку «Розклад», який дає змогу користувачу дивитися свій електронний профіль, перевіряти навчальний календар, змінювати деталі проведення заняття(для викладачів) та редагувати розклад для себе(студентам). Насправді, наше підсистема зведе до мінімуму спілкування викладача та студента щодо планування зустрічі, консультації, чи перенесення заняття, також студентам не потрібно буде іти до стенду у корпусі щоб просто записати свій розклад чи креслити щоденник для повного візуального планування свого тижня.

Ми вважаємо що саме наша підсистема буде мати достатній

					ІК1102.0414.002 ПЗ	Лист
						7
Змн.	Аркуш	№ докум.	Підпис	Дата		

функціонал та буде доступна будь якому користувачу(адже створено IOS/Android додаток для найбільш популярних мобільних платформ, та WEB додатку для усіх користувачів PC) та буде давати користувачу можливість для створення свого електронного розкладу незалежно від інших установ чи сервісів, на базі розкладу свого факультету.

Провівши аналіз було знайдено суттєві переваги та невеликі недоліки нашої підсистеми, які описано нижче :

Переваги:

- створення свого локального розкладу
- централізація інформації у системі «Електронний кампус»
- можливість формування різних планів на різні тижні та дні
- «кроссплатформенність»(додаток написаний для багатьох платформ)

Недоліки:

- для користуванням додатком необхідно залогінитися до «Електронного кампуса»
- немає підтримки windows mobile, отже користувачі цієї платформи будуть вимушені користуватися web додатком
- зредагований розклад студента буде збережений тільки на мобільному пристрої без синхронізації з базою даних «Електронного кампуса», отже при видаленні додатку чи втрати мобільного пристрою всі дані будуть втрачені також
- у офлайн режимі користувач може тільки дивитися “кешовану” інформацію(інформацію яка була показано при останньому онлайн режимі)

1.2. Встановлення переваг та недоліків існуючих систем

Однією з альтернатив системи для перегляду та планування розкладу беззаперечно можна вважати «KPI Weeks». Програма була створена як

					ІК1102.0414.002 ПЗ	Лист
						8
Змн.	Аркуш	№ докум.	Підпис	Дата		

перша система для перегляду свого розкладу. Дуже зручний інтерфейс та дизайн зробили цей додаток лідером серед користувачів. Одним з найбільших переваг використання цього додатку є легкість інтерфейсу. Взагалі, концепт цього додатку дещо відрізняється від додатку розробленого та створеного нами – у «KPI Weeks» користувач має можливість тільки переглядати свій розклад, не має можливості планувати щось та робити нотатки. Тож цей додаток можна вважати ідеальною мобільною версією WEB сервісу Rozklad.Kpi, від якого він на пряму залежить.

Також існує ще одна реалізація розкладу, а саме «KPI Weeks». Додаток аналогічний до «Розклад КПП», навіть джерелом даних виступає ресурс Rozklad.kpi.ua. Основною ж рисою є той факт, що додаток розроблений студентами для студентів.

Нашою метою ж є створення додатку з обмеженим але достатнім для планування та створення свого розкладу функціоналом. Варто відзначити переваги та недоліки(які відсутні у нашій системі) даного додатку.

Переваги:

- зручний інтерфейс
- додаток спрямований на виконання лише однієї функції, тож є дуже простим для користувача і не ускладнює життя
- додаток є мобільним, тож дуже доступний у будь який момент часу для студента або викладача
- дизайн робить додаток дуже приємним у користуванні

Недоліки:

- немає можливості редагування
- додаток має дуже обмежений функціонал, що є як недоліком так і перевагою. На нашу думку додаток потрібно розширити.

- додаток на пряму залежить від Rozklad.kpi.ua, тобто при несправності цього сервісу користування «KPI Weeks» буде також неможливе

1.3. Постановка задачі

Системи подібні нашому «Розкладу» є одними з основних компонентів будь якої системи навчання. Зручне планування занять як викладачем так і студентом є важливим фактором своєчасної підготовки до занять та створення свого розширеного розкладу на будь який час. Для планування розкладу (працівника деканату, викладача або студента) необхідна своєчасна інформація щодо розкладу занять.

На сьогоднішній день актуальним питанням стало планування свого навчання, економія будь якого вільного часу та оптимізація вже існуючих елементів розкладу, отже будь яке можливе спрощення цього завдання є дуже пріоритетною задачею. Таким чином ми визначили тему нашого дипломного проекту «Розробка інформаційного та програмного забезпечення підсистеми «Електронного кампусу» «Розклад» з підтримкою мобільних платформ». Проаналізувавши важливість даного питання ми вирішили що створення унікальної системи для планування свого розкладу буде дуже важливим та зробить планування та оцінку свого навчального часу більш зручним та незалежним, адже система буде працювати цілодобово незалежно від інших сервісів.

Створена підсистема “Розклад” за основу має перегляд свого навчального календаря та має розширений функціонал.

Переваги:

- можливість працювати у онлайн та офлайн режимі завдяки кешуванню. Для праці додатку в офлайн режимі потрібно зайти в онлайн режимі лише перший раз.

					ІК1102.0414.002 ПЗ	Лист
						10
Змн.	Аркуш	№ докум.	Підпис	Дата		

- створено також додатки для платформи Android/IOS та Web, отже додатком може користуватися людина майже з будь-якого девайсу.
- значно збільшена кількість функцій системи та додатку в цілому, весь додаток розбитий на певні контроллери та блоки, що є зручним для розуміння користувачем;
- можливість користуванням мобільним додатком на будь-якій мові
- збереження свого унікального розкладу як на сервері так і в уявній базі даних мобільного пристрою(так зване кешування)
- функція оповіщення інших користувачів про зміни у вашому розкладі(можливе лише для певного виду користувачів)
- синхронізація з електронним профілем КПП «Електронний кампус»

Переваги та недоліки прототипу наведені в табл.1.1

Таблиця 1.1 Переваги та недоліки прототипу підсистеми «Розклад»

Переваги	Недоліки
Створені Android, IOS та WEB додаток що додає нашій підсистемі унікальності	Підтримуються не всі відомі платформи
Наявність усіх мов при використанні мобільних додатків	Додаток дуже сильно залежить від системи “Електронний кампус” так як є його підсистемою
Значно розширена цільова аудиторія, тобто користуватись додатком зможе як студент, так і викладач, так і будь яка інша людина яка має в цьому інтерес.	

Закінчення таблиці 1.1

Система є багатофункціональною і виконує найбільш важливий функціонал	
Конфіденційність інформації та, забезпечення захисту персональних даних користувачів та їх розкладів.	

Завданням цього дипломного проекту є створення унікальної підсистеми на усіх відомих найбільш поширених платформах (IOS, Android, Web) та інтегрувати його у вже існуючу систему «Електронний кампус». Підсистема дає можливість формувати свій унікальний розклад, що надасть легкості та зробить набагато комфортним планування свого навчального розкладу. Перевагами додатку є його багатофункціональність, він задовольняє будь-якого користувача та відповідає потребам звичайних студентів. Підсистему можна використовувати з майже будь-якого мобільного пристрою, а також з WEB додатку, що робить цю підсистемою досягаємою при будь яких обставинах.

1.4. Склад та ієрархія задач та підзадач підсистеми

Метою розробки підсистеми є розширення функціоналу існуючої мобільної версії підсистеми, а також використання актуальних цілісних даних, що зберігаються в єдиному сховищі КПП.

До інформації підсистеми Розклад слід віднести:

- 1) загальнодоступну інформацію (публічна інформація, яка доступна всім)
- 2) інформацію, яка доступна для певної множини користувачів

3) власну інформацію користувача

Як наслідок, доступ до інформації здійснюється неавторизованим та авторизованим способом.

При авторизованому доступі до підсистеми виділяють наступні профілі користувачів:

- Викладач;
- Студент;
- Робітник деканату;

Слід виділити наступні режими роботи з підсистемою:

- онлайн
- оффлайн

Отже, по закінченню дипломного проектування має бути отримано підсистему(IOS), яка буде відповідати всім поставленим далі вимогам, зробить планування свого розкладу більш легким та доступним, а також буде конкурувати з існуючими аналогами за рахунок своїх переваг.

1.5. Схема вхідних та вихідних потоків для задач

Для задачі розробки мобільного додатку вхідним потоком дані (json масив переданий з сервера) які ми отримаємо через API.

Вихідним потоком є представлення на екрані мобільного додатку отриманої інформації. Зворотній зв'язок забезпечується за допомогою передачі запитів через інтерфейс додатку. Дані потоки показані на рис 1.1.

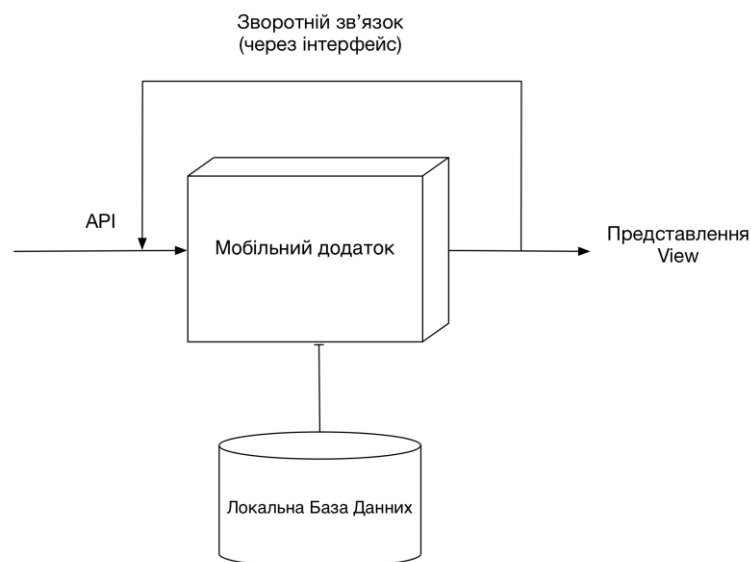


Рисунок 1.1. Схема вхідних та вихідних потоків задач

1.6 Розробка вимог

Сьогодення диктує певні вимоги для всіх додатків. Останні віяння в ІТ, показують, що все більше додатків переносять у WEB та на мобільні платформи, а інформацію все частіше зберігають у хмарних сховищах. Безперечно, у такого підходу є дуже багато плюсів: можливість працювати, майже з будь-якого пристрою, доступ до сервісу з будь-якої частини світу, такі додатки не потребують наявності комп'ютера та вимагають лише підключення до мережі Інтернет.

Але саме підключення до всесвітньої мережі деколи неможливе, хоча останнім часом інтернет стає все доступнішим і з'являється на кожному кроці, проте існують і такі місця, де підключення неможливе з певних причин. Ярким прикладом такого місця є літаки чи метро, а в сучасному ритмі життя, люди працюють майже весь час, і попрацювати в літаку або метро це нормальна справа. Але як бути, якщо потрібний додаток функціонує лише з підключення до інтернет-мережі? Тому багато сучасних програм мають можливість встановити спеціальний клієнт, який буде дозволяти працювати в режимі офлайн, а при підключенні синхронізувати напрацювання. Також однією з проблем є зручність

роботи. Зараз у світі смартфонів більше ніж людей, і багато веб-ресурсів переносяться у нішу мобільних додатків. Такий підхід зумовлений надзвичайною зручністю та гнучкістю.

Тому, було вирішено, що підсистема «Розклад» повинна бути реалізована для мобільних платформ та мати схожу технологію, що дозволить її використовувати без підключення до інтернет. Таким чином, якщо користувач має доступ до інтернету, то він може працювати або в онлайн-режимі, або в офлайн. В онлайн-режимі вся робота буде зберігатись у реальному часі на серверах, в офлайн-режимі всі напруцювання синхронізувати при першому підключенні.

1.6.1 Вимоги до функціоналу

Система для роботи з поточним навчальним розкладом повинна забезпечувати систему авторизації, що забезпечить певний рівень безпеки при використанні додатка. Уся персональна інформація та профілі користувачів уже містяться у системі «Електронного кампусу», тому з нашого боку достатньо обмежити доступ до цієї інформації шляхом введення авторизації.

Зважаючи на те, що підсистема створюється як для студентів, так і для викладачів, то внутрішня логіка додатку підтримує роботу з різними правами доступу до даних. Такий підхід дозволяє надати різний функціонал для користувачів, що в свою чергу забезпечує ще один рівень захисту даних.

Дана підсистема розрахована на надання можливості опрацювання даних в широкому діапазоні дій. Таким чином, функціонал додатку включає в себе не тільки стандартні можливості перегляду та видалення елементів розкладу, а й більш складні, такі як: редагування та додавання нових даних на рівні свого профілю, редагування та додавання на рівні груп.

					ІК1102.0414.002 ПЗ	Лист
						15
Змн.	Аркуш	№ докум.	Підпис	Дата		

Використовуючи додаток, користувачі матимуть змогу також вносити певні зміни у свій профіль, згідно з реалізованим функціоналом додатку. Ця функція не є цільовою та виконує суто допоміжну роль.

Важливим функціоналом додатку є оповіщення користувачів що пов'язані зі змінами даних розкладу, шляхом розсилання інформаційного повідомлення з коротким описом змін що відбулись у поточному розкладі користувача.

Більш детальні вимоги по функціоналу наведено в табл. 1.2

Таблиця 1.2 Вимоги до функціоналу мобільного додатку

Функція	Вимоги
Авторизація	Потрібно надати користувачу можливість авторизуватися, використовуючи вже існуючий профіль в системі «Електроний кампус». Авторизація – це процес валідації логіна і пароля. Якщо така пара є в БД системи, то користувачу надається доступ до внутрішніх даних.
Редагування профілю	Потрібно забезпечити функціонал, для редагування персональної інформації користувача. Змінити можна лише певні дані профілю, відповідно наданій користувачу інформації.

Продовження таблиці 1.2

Перегляд поточного навчального розкладу	Потрібно забезпечити можливість перегляду поточного навчального розкладу для всіх користувачів «Електронного кампусу». Реалізована функція розширеного перегляду з виведенням номеру корпусу, аудиторії, П.І.Б. викладача, годинами початку та закінчення, скороченою та повною назвою предмета.
Видалення елементів навчального розкладу	Потрібно реалізувати можливість видалення предметів розкладу. Видалення може відбуватись на двох рівнях: студента та викладача. Рівень студента дозволяє видаляти елементи розкладу лише для себе, ці зміни не відносяться більше ні для кого в системі. Рівень викладача видаляє елементи розкладу на рівні груп, тобто змінює розклад для всіх користувачів-студентів, що мають відношення до цього елементу.

Продовження таблиці 1.2

<p>Редагування існуючих елементів навчального розкладу</p>	<p>Потрібно надати змогу привносити зміни до уже існуючих елементів навчального плану. Редагування також може виконуватись на двох рівнях.</p> <p>Рівень студента дозволяє редагувати елементи розкладу виключно для себе.</p> <p>Рівень викладача вносить зміни на рівні груп, тобто змінює розклад для всіх користувачів-студентів, що мають відношення до цього елементу.</p>
<p>Створення нових елементів навчального розкладу</p>	<p>Потрібно забезпечити функцію створення нових елементів розкладу, таких як консультації, додаткові практики, тощо. Створення нових елементів також може виконуватись на двох рівнях.</p> <p>Рівень студента дозволяє створювати події лише для себе, залишаючи їх невидимими для всіх інших користувачів системи.</p> <p>Рівень викладача створює нові елементи на рівні груп, тобто вносить зміни для всіх користувачів-студентів відповідної групи.</p>

Закінчення таблиці 1.2

Забезпечення коректності вводу всіх даних	Потрібно перевіряти коректність всіх введених даних, та, при помилках під час введення даних, повідомляти про це користувача.
Функція сповіщення про зміни	Потрібно реалізувати функцію сповіщення про зміни усіх користувачів, які «зв'язані» з елементом що підлягає редагуванню.
Кешування інформації	Необхідно забезпечити можливість роботи в офлайн режимі шляхом кешування змін з синхронізацією в майбутньому.

1.6.2 Вимоги до інтерфейсу

З розвитком технологій вимоги до інтерфейсу невпинно зростають. Останні віяння мобільних технологій схиляють розробку додатків до впровадження здатності пристосовувати під різні розміри дисплеїв пристроїв. А також бути естетичним, лаконічним, гарним. Отже, додаток для роботи з поточним навчальним розкладом не є виключенням.

Сучасні вимоги до мобільних додатків диктують досить жорсткі правила до їх інтерфейсної частини. Потрібно розуміти що додаток створюється для користувача, і задоволення його потреб чи не найперший крок для будь-якого мобільного сервісу незалежно від платформи. Клієнт має справу тільки з інтерфейсом і не бере безпосередньої участі в роботі з внутрішніми модулями. Як результат, головний критерій для користувача – це дизайн.

Так як додаток розробляється для IOS платформ, то даний інтерфейс побудований за « iOS Human Interface Guidelines», що були запропоновані представниками Apple.

При розробці додатку була поставлена ціль «мінімум дій - максимум потрібної інформації». Для реалізації цієї мети був використаний набір інструментів наданий Apple для розробки інтерфейсів на IOS-платформу.

Як говорилося вище, додаток розрахований на підтримку різних пристроїв, як планшетів, так і смартфонів. Такий підхід накладає певні вимоги на проектування функціонального інтерфейсу. В результаті дизайн повинен бути «гумовим», що забезпечить коректне відображення графічних елементів на будь-якому з дисплеїв.

Підсистема «Розклад» створювалась з метою надання можливості перегляду поточного учбового плану, що, в свою чергу, накладає певні обмеження. Спосіб подання інформації залежить від значимості відповідних даних та виділяється акцентами. Так, назва предмету повинна займати головне положення в елементі поточного розкладу та виділятися більшим розміром шрифту. Дані про викладача та аудиторії розміщуються безпосередньо нижче назви, виводяться меншим шрифтом та курсивом.

Також велику роль відіграють кольори. Набір кольорів повинен відповідати вимогам «IOS Humane Interface Guidline», таким чином в додатку переважають нейтрально синій та білий кольори, з чорним текстом. Для акцентування уваги використовуються сірий, зелений та червоний. Так, для відображення неактивної інформації блоки з нею фарбуються в сірий.

1.6.3. Вимоги до архітектури

Для реалізації правильної і швидкої роботи в додатку «Розклад» було реалізовано патерн проектування MVC, а точніше його нащадка MVP. Реалізація цього шаблону зумовлює розбиття програми на блоки «model»,

«view» та «controller». Така структура накладає певні вимоги на елементи архітектури. Таким чином, «controller» виконує логіку та внутрішній функціонал додатку, використовуючи при цьому елемент даних «model», та відображає результат у «view». «Model» повинен виступати набором елементів додатку які служать джерелом даних. Користувач отримує інформацію на пряму з «view» та працює з додатком використовуючи «controller». Такий підхід використовується в дуже широкому спектрі розробки додатків, не тільки мобільних, а й десктопних. Використовуючи подібний підхід додаток набуває додаткової гнучкості, можливості масштабування та удосконалення.

Також, функціонал додатку «Розклад» накладає деякі вимоги на архітектуру, а саме створення внутрішнього кешу. В нашому випадку кешем буде слугувати локальна база даних. Такий підхід дозволить як пришвидшити роботу всього додатку, так і забезпечити офлайн режим. Внутрішня БД дублює відповідні таблиці серверної БД та зберігає там дані для відображення та роботи з додатком.

Висновки до розділу

В цьому розділі проведений аналіз предметної області та визначені місця задачі серед задач системи «Електронний Кампус». Також поставлені конкретні задачі, які необхідно виконати при розробці системи, для задач проведена їх декомпозиція, визначені вимоги та рекомендації для кожної підзадачі, яких потрібно дотримуватись при виконанні цих задач. Досліджені вхідні та вихідні потоки кожної підзадачі, за яким визначено послідовність їх виконання та розроблені алгоритми їх виконання.

Проаналізувавши та оцінивши всі існуючі аналоги системи для роботи з поточним навчальним розкладом КПП, нами було створено (було встановлено недоліки та сформульовано задачі, які слід вирішити) підсистему “Розклад”. Основною задачею створеною нами системи є

					ІК1102.0414.002 ПЗ	Лист
						21
Змн.	Аркуш	№ докум.	Підпис	Дата		

планування індивідуального розкладу користувача на базі вже існуючого розкладу в навчальному закладі, а саме в КПП. Основний удар зроблений на розробку унікальної архітектури та API, а також на створення мобільних додатків з дружнім користувачеві інтерфейсом.

При цьому інформація розкладу поділяється на:

- власну інформацію користувача
- інформацію, яка доступна для певної множини користувачів
- загальнодоступну інформацію (публічна інформація, яка доступна всім)

Після постановки задач наступним кроком є розробка інформаційного забезпечення системи.

					ІК1102.0414.002 ПЗ	Лист
						22
Змн.	Аркуш	№ докум.	Підпис	Дата		

2. РОЗРОБКА ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

2.1. Структура локальної бази даних

Основною задачею використання локальної бази даних є кешування необхідної користувачеві інформації, тобто збереження її локально на мобільному пристрої. Для цього нам не потрібно клонувати всю базу даних з сервера а тільки окрему її частину. Таким чином, створено локальну БД яка має 6 таблиць. Надалі конкретно про кожну таблицю.

Таблиця *Profile* створена для збереження основної інформації користувача, а саме :

- *course(Int64)*
- *id(Int64)*
- *faculty(String)*
- *group(String)*
- *name(String)*
- *surname(String)*

Та зв'язки з іншими таблицями :

- *eventsOwner*(тип :один до багатьох, таблиця: *Event*)
- *teachersOnStage*(тип :один до багатьох, таблиця: *Teacher*)
- *week*(тип :один до багатьох, таблиця: *Week*)

Таблиця *Teachers* створення для збереження інформації про викладача та має поля :

- *id(Int64)*
- *name(String)*
- *scienceDegree(String)*
- *secondName(String)*

Та зв'язки з іншими таблицями :

- *subjects*(тип :один до багатьох, таблиця: *Subject*)

Таблиця *Events* створення для збереження інформації про додану подію користувача та має поля :

- *id(Int64)*
- *name(String)*
- *type(String)*
- *place(String)*

Та звязки з іншими таблицями :

- *owner*(тип :один до одого, таблиця: *Profile*)

Таблиця *Week* створення для збереження інформації про розклад певого тиждня для користувача та має поля :

- *id(Int64)*
- *workDays(Schedule)*
- *modified(Bool)*

Та звязки з іншими таблицями :

- *createdEvents*(тип :один до одого, таблиця: *Events*)
- *schedule*(тип :багато до багатьох, таблиця: *Schedule*)

Таблиця *Schedule* створення для збереження інформації про розклад конкретного дня та складається з об'єктів таблиці *Subject* та має поля :

- *id(Int64)*
- *additionalEvents(Events)*
- *numberOfSchedules(Bool)*

Та звязки з іншими таблицями :

- *subjects*(тип :один до багатьох, таблиця: *Subject*)

Дана локальна база даних представлена на рис. 2.1.

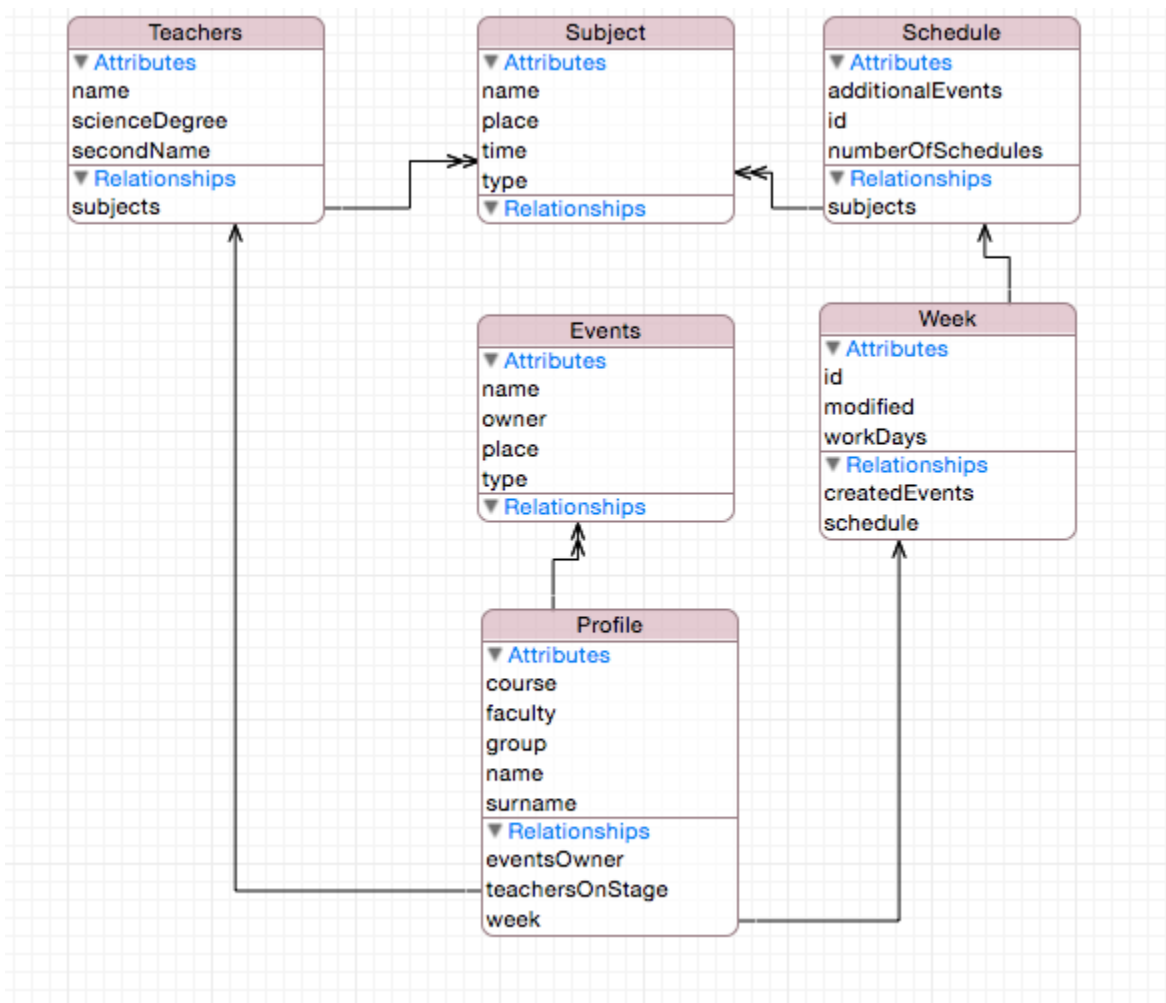


Рисунок. 2.1. Локальна база даних

Висновки до розділу

У даному розділі описана структура локальної бд яка використовується при вирішенні задачі “кешування”. Описані поля та таблиці зі зв’язками.

Кешування – це так зване збереження даних програми всередині самого мобільного пристрою незалежно від статусу чи доступності зовнішнього серверу. Провівши аналіз нашої підсистеми було вирішено робити кешування при згортанні додатку(так званому виході, для юзера, тому що у мобільному пристрої додаток перебуває у робочому стані у фоні ще деякий проміжок часу) та при переході від головних контролерів до допоміжних. Таким чином ми не будемо дуже сильно навантажувати

фоновий потік та зможемо зберігати дані користувача у всіх необхідних моментах та переходах.

Локальна БД має 6 таблиць та зв'язки між ними, повністю вся бд показана на рис. 2.1. Атрибути(поля) усіх таблиць та їх призначення детально описані в цьому розділі.

					ІК1102.0414.002 ПЗ	Лист
						26
Змн.	Аркуш	№ докум.	Підпис	Дата		

3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1. Вибір середовище та технології розробки

Враховуючи те що ІТ сфера дуже швидко розвивається та набирає обертів, мов програмування та середовищ розробки стає все більше і більше, однак при написанні мобільного додатку для платформи IOS не має можливості вибору, так як компанія Apple створила лише одне середовище та постійно його покращує, оновлює, підтримує.

Оскільки за весь період свого досвіду програмування я використовував цю середу розробки, додаток для мобільного пристрою було вибрано писати саме в цій середі. Для розробки IOS-додатку була обрана середа розробки Xcode, оскільки вона поєднує простоту та швидкість розробки, та ідеально підходить для даної задачі, має весь необхідний функціонал та дуже зручний інтерфейс білдер. Ці програмні продукти є вільним для користування, тому що компанія Apple піклується про своїх розробників і забезпечує їх усіма новинками та чудовою документацією що робить процес розробки більш зрозумілим при використанні певних прикладів від розробника середовища. Єдине за що потрібно платити – це за профіль у розробницькому товаристві Apple, участь у ньому дозволяє викладати свої мобільні додатки до appStore після детального перегляду та тестування зі сторони Apple. У нашій країні нажаль саме ця технологія розробки та напрямок є дуже рідким серед студентів, тож Apple не створило ніяких програм чи спеціальних пропозицій з цього приводу.

Важливим був вибір мови програмування, адже у Xcode можна писати на багатьох, основні з яких

- Swift
- Objective – c

Провевши аналіз було вирішено використовувати мову Swift, так як вона є більш новою і створено дуже багато нових функцій. Також створено дуже багато так званих third-party-libraries(бібліотеки які реалізують функціонал будь якого типу, створені досвідченими розробниками, їх можна дуже легко інтегрувати у свій проект та використовувати вже готові інтерфейси, функції тощо). Ще одною з причин було те, що за думкою експертів мова програмування “objective – c” вже “віджила своє”, так як вона була створена ще у минулому столітті. Навіть ведучі експерти Apple роблять удар на тому щоб розробники використовували вже мову Swift, “objective-c” все менше і менше підтримується навіть самою компанією Apple, так як нова документація створена в основному для Swift.

Також у самій мові програмування використовуються певні основні та допоміжні фреймворки, про які ми розповімо нижче.

Серед систем контролю версій, було обрано BitBucket. Адже він дозволяє безкоштовно розмішувати два закритих репозиторії.

У наступних пунктах детальніше описано про середовище розробки, про мову програмування та про вибрані технології.

Foundation

Foundation визначає базовий шар Objective-C класів(а при створенні мови Swift ці класи було інтегровано). На додаток до безліч корисних примітивних об'єктних класів, він вводить декілька парадигм, які визначають функціональність, не охоплених мовою Objective-C/Swift. Foundation розроблений з урахуванням цих цілей:

- Забезпечити невеликий набір основних класів допоміжними класами.
- Зробити розробки програмного забезпечення простіше, вводячи послідовні угоди для таких речей, як звільнення(вивільнення з пам'яті).
- Підтримка Unicode рядків, постійність об'єктів та їх розподіл.

- Забезпечити рівень незалежності OS, для підвищення мобільності.

Foundation включає в себе клас об'єктів кореневого класу, що представляють основні типи даних, такі як рядки і масиви байтів, класів колекцій для зберігання інших об'єктів, класів, що представляють системну інформацію, таку як дати, і класів, що представляють комунікаційні порти.

Foundation представляє кілька парадигм, щоб уникнути плутанини в загальних ситуаціях, і уявити рівень узгодженості між ієрархіями. Ця узгодженість робиться з деяким стандартним правилами, такими як, що для володіння об'єктом (тобто, хто несе відповідальність за розміщення об'єктів), та з абстрактними класами, як NSEnumerator. Ці нові парадигми скорочують кількість спеціальних та виключних випадків в якості API і дозволяють кодувати більш ефективно за рахунок повторного використання тих ж самих механізмів та технологій з різними видами об'єктів.

Ієрархія класів Foundation корениться в NSObject класу фреймворку Foundation. Згадки о Foundation складаються з декількох суміжних груп класів, а також з кількох окремих класів. Багато з груп утворюють те, що називають кластери-абстрактні класи, які працюють, як парасолька інтерфейсів для універсального набору приватних підкласів. NSString і NSMutableString, наприклад, діють в якості посередника для екземплярів різних підкласів, оптимізованих приватних для різних видів потреб зберігання. Залежно від способу ви використовуєте для створення рядка, ви отримаєте екземпляр відповідного класу.

UIKit

UIKit (UIKit.framework) забезпечує найважливішу інфраструктуру, необхідну для побудови та управління IOS додатків. Ця структура забезпечує вікно й вид архітектури, необхідної для управління для користувача інтерфейсом програми, відповідає за обробку подій,

					ІК1102.0414.002 ПЗ	Лист
						29
Змн.	Аркуш	№ докум.	Підпис	Дата		

необхідну для реагування на дії користувача, і додаток, який повинен виконувати основний цикл програми і взаємодіяти з системою.

На додаток до основних завдань, UIKit забезпечує підтримку таких функцій:

- Вид моделі контролера для інкапсуляції вмісту користувацького інтерфейсу
- Підтримка обробки сенсорів і руху на основі події
- Підтримка моделі документа, який включає інтеграцію iCloud
- Графічна підтримка та підтримка вікон, в тому числі підтримка зовнішніх дисплеїв
- Підтримка для управління фонових і переднього плану процесів програми
- Підтримка друку (“підказує” тобі варіанти наступних рядків коду)
- Підтримка для налаштування зовнішнього вигляду стандартних елементів управління UIKit
- Підтримка текстового та веб-контенту
- Підтримка для анімації змісту користувацького інтерфейсу
- Інтеграція з іншими додатками в системі через схеми URL і бібліотечних інтерфейсів
- Доступність підтримки користувачів з обмеженими можливостями
- Підтримка service Push Notification Apple
- Створення PDF
- Підтримка користувацького тексту, який взаємодіє з системною клавіатурою
- Підтримка спільного використання контенту за допомогою електронної пошти, Twitter, Facebook та інші

На додаток до фундаментального коду для створення вашого додатку, UIKit також включає підтримку для деяких конкретних пристроїв функцій, таких як наступні:

					ІК1102.0414.002 ПЗ	Лист
						30
Змн.	Аркуш	№ докум.	Підпис	Дата		

- Вбудованою камерою (де присутня)
- Фото бібліотеки користувача
- Ім'я пристрою й інформація про модель пристрою
- Інформація про стан батареї

XCODE

Xcode — інтегроване середовище розробки (IDE) виробництва Apple. Дозволяє створювати програмне забезпечення з використанням таких технологій як GCC, GDB, Java та ін. На сьогодні є єдиним засобом написання «універсальних»(Universal Binary) прикладних програм для Mac OS X та IOS.

Xcode включає в себе більшу частину документації розробника від Apple та Interface Builder - застосунок, який використовується для створення графічних інтерфейсів.

Пакет Xcode містить змінену версію вільного набору компіляторів GNU Compiler Collection і підтримує мови C, C++, Objective-C, Swift, Java, AppleScript, Python і Ruby з різними моделями програмування, включаючи (але не обмежуючись) Cocoa, Carbon і Java. Сторонніми розробниками реалізована підтримка GNU Pascal, Free Pascal, Ada, C #, Perl, Haskell. Пакет Xcode використовує GDB в якості back-end'a для свого відналагоджувача.

У серпні 2006 Apple оголосила про те, що DTrace, фреймворк динамічного трасування від Sun Microsystems, випущений як частина OpenSolaris, буде інтегрований в Xcode під назвою Xray. Пізніше Xray був перейменований в Instruments.

Xcode IDE знаходиться в центрі досвіду розвитку Apple. Тісна інтеграція з рамками Cocoa і Cocoa Touch робить з Xcode неймовірно продуктивне навколишнє середовище для будівництва дивовижних програми для Mac, iPhone, iPad і.

					ІК1102.0414.002 ПЗ	Лист
						31
Змн.	Аркуш	№ докум.	Підпис	Дата		

Саме тому що все так добре інтегровано робочі процеси ведуть себе природньо. В той час як ви складаєте новий інтерфейс, помічник редактора інтуїтивно представляє відповідний вихідний код в розділеній панелі вікна. Просто перетягніть мишкою для підключення елементів управління користувальницького інтерфейсу в кодї реалізації. Технології компілятора LLVM Apple, розбирають код, зберігаючи кожен символ, який ви бачите в LLDB відладчика відповідно до редактора і компілятора. У той час як ви займаєтесь кодуванням, той же двигун постійно на роботі, знаходячи помилки і пропонуючи “пофіксити” його для вашого коду.

Xcode навіть спілкується з сайтом розробника Apple, так що ви можете включити такі послуги, як Game Center або ощадної книжки у вашому додатку за допомогою одного кліка. У поєднанні з OS X Server, Xcode можна налаштувати віддалений бот постійно будувати, аналізувати, тестувати і навіть упаковувати ваш додаток. Коли ваш додаток буде готовий, Xcode буде пов'язувати і представляти ваш додаток в App Store.

Swift

Swift це нова мова програмування для IOS і OS X додатків, яка будується на кращому з C і Objective-C, без обмежень сумісності з C. Swift приймає безпечні моделі програмування і додає сучасні функції, щоб зробити програмування простіше, більш гнучким і веселим. Swift був створений з чистого аркуша, спираючись на зрілий і дуже коханий усіма Apple розробниками Cocoa та Cocoa Touch, це можливість переосмислити, як працює розробка програмного забезпечення.

Swift був роками у процесі становлення. Apple, заклали основу для Swift, просуваючи свій існуючий компілятор, відладчик, і фреймворкову інфраструктуру. Вони спростили управління пам'яттю з автоматичним підрахунком посилань (ARC). Їх стек, побудований на міцному фундаменті Foundation та Cocoa, був модернізований і стандартизований у всьому. Сам

					ІК1102.0414.002 ПЗ	Лист
						32
Змн.	Аркуш	№ докум.	Підпис	Дата		

Objective-C розвивалися, щоб підтримувати блоки, колекції літералей і модулів, що дозволяє фрейворкам впровадженню сучасних технологій без мовних зривів. Завдяки цим напрацюванням ми можемо тепер уявити нову мову для майбутнього розвитку програмного забезпечення Apple.

Swift буде знайомим розробникам які використовують Objective-C. Він приймає читаність названих параметрів Objective-C і силу динамічної моделі об'єкта Objective-C. Це забезпечує прямий доступ до існуючих фреймворків Cocoa і перемішує й поєднує взаємодію з Objective-C кодом. Swift вводить багато нових можливостей і об'єднує процесуальні та об'єктно-орієнтовані частини мови.

Swift є дружнім до нових програмістів. Це перша промислової якості система мови програмування, яка є виразною і приємною, як у мові скриптів. Він підтримує так звані нові "дитячі майданчики", нова функція, яка дозволяє програмістам експериментувати з Swift і побачити результати негайно, без накладних витрат будівництва та запуск програми. Ця функція може використовуватися лише для експериментів тому що дуже сильно навантажує дебагер.

Swift поєднує найкраще в сучасному мисленні від Apple, інженерної культури. Компілятор оптимізований для роботи, і мова, оптимізована для розвитку, без шкоди для будь-кого. Вона призначена для масштабування від "Hello, World" до цілої операційної системи.

Для будь кого хто бажає вивчити цю нову та унікальну мову програмування у вільному доступі є книжка “Swift Programming Language Guide” від Apple, де дуже скурпульозно описані всі переваги та недоліки а також надані приклади коду та технологій/лексикону який використовується.

Cocoa

Cocoa — рідний прикладний програмний інтерфейс (API) об'єктно-орієнтований для операційної системи Mac OS X. Це один з п'яти основних

					ІК1102.0414.002 ПЗ	Лист
						33
Змн.	Аркуш	№ докум.	Підпис	Дата		

API, що доступні в Mac OS X, — Cocoa, Carbon, Toolbox (для роботи старих застосунків Mac OS 9), POSIX та Java. Такі мови, як Perl, Python та Ruby не вважаються основними, оскільки на них поки що пишеться не так багато серйозних застосунків.

Застосунки, що використовують Cocoa, зазвичай розробляються за допомогою середовища розробки Apple Xcode (в минулому називалася Project Builder) та Interface Builder з використанням мови Objective-C. Однак, середовище Cocoa також доступне і при розробці на інших мовах, таких як Ruby, Python та Perl за допомогою пов'язуючих бібліотек (RubyCocoa, PyObjC та CamelBones відповідно). Також можна писати Cocoa-програми на Objective-C в звичайному текстовому редакторі та вручну компілювати їх за допомогою GCC або make-сценаріїв для GNUstep.

З точки зору користувача, Cocoa-застосунки це застосунки, що написані з використанням програмного середовища Cocoa. Такі застосунки зазвичай мають характерний вигляд, оскільки це середовище багато в чому спрощує підтримку «людського інтерфейсу» Apple (Apple Human Interface Guidelines).

Модель-перегляд-поведінка (MVC)

Команди Smalltalk-програмістів з Xerox PARC врешті-решт виробили філософію, дозволившу спростити розробки та значно зменшити обсяг повторно використовованого коду. Відома як «парадигма модель-перегляд-поведінка» (MVC), ця концепція передбачає розділ застосунку на три набори взаємодіючих між собою класів. Класи моделі представляють дані, такі як документи, файли налаштувань або об'єкт в пам'яті. Перегляди, відображують ці дані (зазвичай візуально). Класи поведінки містять логіку, пов'язуючу моделі з відповідними переглядами, та забезпечують їх синхронізацію.

Архітектура Cocoa суворо дотримується принципів MVC. В OpenStep більшість класів були або переглядами високого рівня (класи AppKit), або відносно низькорівневими класами моделі (наприклад, NSString). Порівняно зі зхожими MVC-системами, в OpenStep бракувало сильної бази моделей. Наприклад, не існувало базового класу, який би представляв документ. Під час переходу до Cocoa, база моделей була неймовірно поширена, й стала включати декілька готових до використання класів, забезпечуючих функціональність, загальну для більшості застосунків користувача.

В Mac OS X 10.3 Apple представила сімейство класів MVC, забезпечуюче стандартну функціональність поведінки —NSController. Ці класи вважаються частиною системи Cocoa Bindings яка широко використовує такі протоколи як Key-Value Coding та Key-Value Observing. Термін **binding** (зв'язування) означає зв'язку двох об'єктів, часто перегляду та поведінки. Cocoa Bindings дозволяють розробнику зосередитися на описі зв'язків між об'єктами замість того, що детально описувати поведінку програми.

З виходом Mac OS X 10.4 Apple ще більш розширила основні класи, представив фреймворк Core Data, автоматизуючий відстеження змін в моделях та їх збереження (наприклад, в файл). Цей фреймворк значно полегшує роботу з даними в застосунку, надаючи автоматичну підтримку читання документів з файлу та збереження їх у файлі, а також архітектури скасування та повернення змін.

Забезпечуючи фреймворки для підтримки усіх трьох рівней MVC, Apple має на меті зменшити кількість «склеючого» коду, який примушені писати розробники, і звільнити таким чином їх час на написання унікальних для конкретного застосунка функцій.

Core Data

Core Data даних містить загальні та автоматизовані вирішення загальних завдань, пов'язаних з об'єктом життєвого циклу і управління об'єктами графа, в тому числі постійності. По фатку – це вбудована база даних у ваш мобільний пристрій. Його функції включають в себе:

- Відстеження змін і скасовування їх.
- Core Data забезпечує вбудовану в управління відміну і повтор базового редагування тексту.
- Обслуговування зв'язків.
- Core Data управляє змінами, у тому числі підтримкою узгодженості відносин між об'єктами.
- Core Data може зменшити накладні витрати пам'яті вашої програми по лінивим завантаженням об'єктів.
- Автоматична перевірка значень властивостей.
- Керовані об'єкти Core Data розширюють стандартні методи перевірки коду типу "ключ-значення", які гарантують те, що окремі значення лежать в межах допустимих діапазонів, так що комбінації значень є реальними та мають значення.
- Схема міграції.
- Робота зі зміною схеми для вашого застосування може бути важкою, з точки зору зусиль в області розвитку і затрат ресурсів під час виконання. Інструменти міграції схеми основних даних спрощують завдання зі змінами в схемі, а в деяких випадках дозволяють виконувати надзвичайно ефективний в специфічному випадку міграцію схеми.
- Повна, автоматична підтримка кодування типу "ключ-значення" і "ключ-значення" відстежування.
- На додаток до синтезу кодування типу "ключ-значення" і обстеження типу Єключ-значення" сумісні методи доступу для атрибутів, Core

Data синтезує відповідні методи доступу для зв'язків один до одного та один-багатьох.

- Угрупування, фільтрація та організації даних в пам'яті і в інтерфейсі.
- Автоматична підтримка для зберігання об'єктів в сховищах зовнішніх даних.
- Складні компіляції запиту.
- Замість того щоб писати SQL, ви можете створювати складні запити, пов'язуючи об'єкт NSPredicate з вибірки запиту. NSPredicate забезпечує підтримку основних функцій, пов'язаних підзапитів та інших передових SQL запитів. З Core Data, він також підтримує правильне кодування, локалізований пошук, сортування та регулярні вирази.
- Політика злиття.
- Core Data забезпечує вбудовану відстеження версій і оптимістичне блокування для підтримки автоматичного вирішення конфліктів.

BitBucket

Bitbucket («відро бітів») — веб-сервіс для хостингу проектів та їх спільної розробки, заснований на системі контролю версій Mercurial і Git. За призначенням і пропонованих функцій аналогічний GitHub (однак GitHub не надає безкоштовні «закриті» репозиторії, на відміну від Bitbucket), який підтримує Git і Subversion.

В даний час всім користувачам безоплатно надаються наступні можливості:

- Дисковий простір до 2 ГБ на репозиторій.
- Необмежена кількість публічних репозиторіїв.
- Необмежена кількість приватних репозиторіїв для команд до п'яти осіб.
- Доступ до репозиторіїв по протоколах HTTP і SSH.

- Можливість прив'язати обліковий запис на сервісі до власного домену.
- Вікі (окремо для кожного сховища, можна відключити).
- Система обліку помилок (окремо для кожного сховища, можна відключити).
- Інтеграція з Google Analytics, Twitter, Basecamp та іншими службами.
- RSS-стрічка історії змін.
- Управління приватністю окремо для кожного сховища.
- Для публічних репозиторіїв кількість користувачів не обмежена (BitBucket безкоштовний для проектів відкритого програмного забезпечення).
- До приватного (закритого) репозиторія може мати доступ до п'яти користувачів; більша кількість записів надається в рамках платного обслуговування (від \$10 до \$80 в місяць).

3.2. Архітектура додатку

Архітектура будь якого програмного забезпечення – це один з найважливіших модулів в розробці, так званий «наріжний камінь» додатку. Якісна архітектура забезпечить хорошу масштабуємість та гнучкість усього проекту, полегшить «читання коду» та розуміння логіки внутрішньої роботи. Погано ж продумана архітектура навпаки ускладнить майбутню розробку додатку, і чим більше проект буде удосконалюватись, тим складнішими і незграбними будуть нові доповнення, функціонал, тощо. В підсумку, в якийсь період розробки ми отримаємо програмний продукт з атрофованою гнучкістю та масштабуємістю, і тоді вигідніше буде розпочати розробку нового продукту, ніж підтримувати розробку старого. Саме тому архітектура найважливіший з модулів розробки, важливіше навіть ніж функціонал, так як якісна архітектура підтримує впровадження нових можливостей в роботу додатку.

					ІК1102.0414.002 ПЗ	Лист
						38
Змн.	Аркуш	№ докум.	Підпис	Дата		

Розробка мобільного додатку на платформі IOS зумовлює використання та реалізацію патерна проектування MVC, а саме його нащадка MVP.

MVP(Model-View-Presenter) - один з найпоширеніших архітектурних шаблонів проектування, похідний від MVC, принцип полягає у розділенні даних, візуального відображення та поведінки обробки подій у різні класи, а саме: Модель даних(Model), Представлення(View) та Пред'явник(Presenter). Подібна структура дозволяє змінювати інтерфейс користувача з мінімальними впливами на роботу з внутрішніми даними, та навпаки, вносити зміни в модель даних без модифікування інтерфейсу користувача.

Дана архітектура наведена на рис. 3.1.

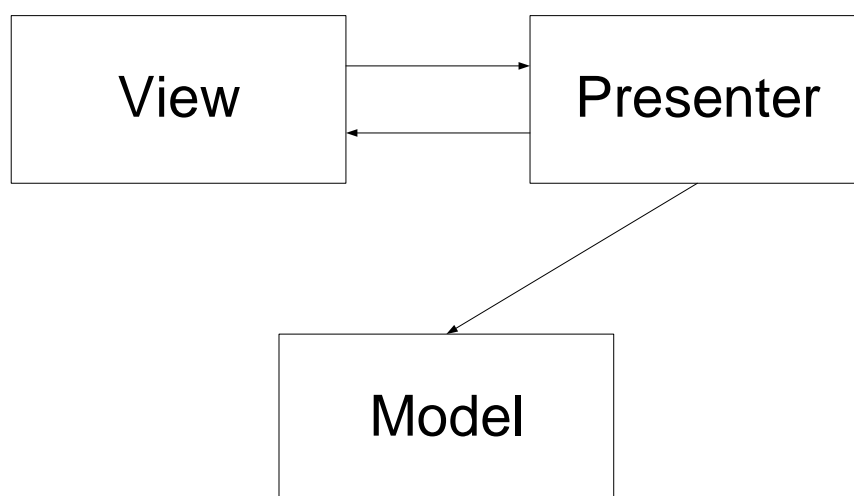


Рисунок 3.1. Архітектура додатку.

Мета шаблону – це забезпечення гнучкої архітектури програмного забезпечення, що в свою чергу полегшить майбутні зміни чи розширення додатку. Також надання можливості повторно використовувати окремі модулі та компоненти програми. Використовування цього патерну у

великих системах забезпечує певну впорядкованість їх структури і підвищує зрозумілість завдяки зменшенню ступеня складності.

Також, зважаючи на те, що основною вимогою до підсистеми «Розклад» була саме можливість працювати в режимі без доступу до мережі інтернет, то потрібно було побудувати таку архітектуру додатку, яка б могла виконувати таку вимогу. Проаналізувавши поставлене завдання, ми прийшли до висновку, що потрібно застосовувати локальну базу даних. Використовувати, СУБД для такого завдання немає потреби, адже можна обійтись використанням вже наданих платформою IOS інструментів. Таким чином локальна база даних буде створена за допомоги SQLite, що дозволить нам реалізувати наш задум просто та швидко, без підключення сторонніх технологій.

Для нормального функціонування підсистеми в цілому потрібна серверна СУБД, яка, в свою чергу, буде забезпечувати збереження всіх даних, які необхідні для роботи додатку.

Дана архітектура наведена на рис. 3.2.

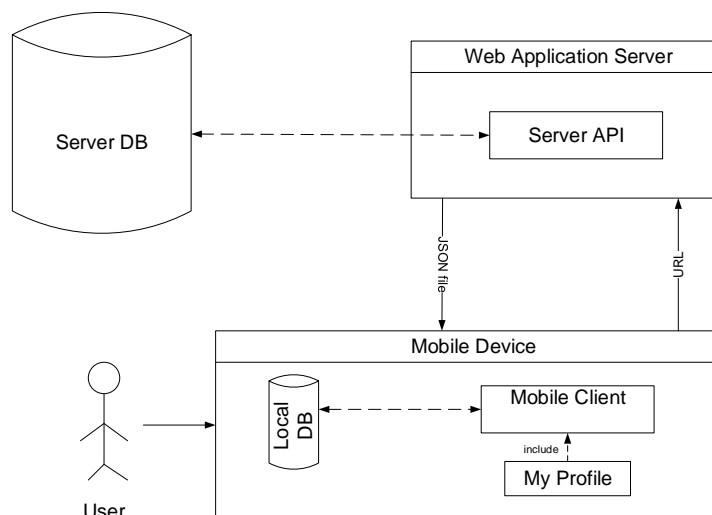


Рисунок 3.2. Архітектура підсистеми.

3.3. Опис розробленого функціоналу

Авторизація

Основним завданням авторизації є надання доступу до персональних даних користувача, а також захист інформації від несанкціонованого доступу. Авторизація проводиться в системі «Електронний кампус». При авторизації треба вказати такі дані як логін та пароль.

Процес авторизації є нічим іншим, як співставлення пари логін і пароль, з записами у базі даних додатку. Якщо, знайдена відповідна пара, то користувачу надається доступ до системи.

Усі логіни та паролі користувачів містяться на серверах «Електронного кампусу» в зашифрованому вигляді.

Редагування інформації профілю

Тут користувачу надається можливість змінювати інформацію профілю, та персональну інформацію. Наприклад, в цьому розділ можна змінити свій пароль, або зв'язати свій аккаунт у соціальних мережах з аккаунтом на цьому сайті.

Перегляд поточного навчального розкладу

Процес дозволяє користувачу переглядати дані поточного робочого плану. Кожний елемент розкладу, так званий «предмет» включає в себе дані про коротку та повну назву предмету, час початку та закінчення, номер корпусу та аудиторії, П.І.Б. викладача.

Дані подаються у вигляді розпорядку на кожен день, з можливістю переходу з однієї вкладки на іншу.

					ІК1102.0414.002 ПЗ	Лист
						41
Змн.	Аркуш	№ докум.	Підпис	Дата		

Видалення елементів навчального розкладу

Цей процес, надає користувачу можливість видаляти елементи розкладу. Слід зазначити що видалення може відбуватись на двох рівнях: студента та викладача.

Рівень студента дозволяє видаляти елементи розкладу лише для себе, ці зміни не відносяться більше ні для кого в системі.

Рівень викладача видаляє елементи розкладу на рівні груп, тобто змінює розклад для всіх користувачів-студентів, що мають відношення до цього елементу.

Редагування існуючих елементів навчального розкладу

Цей процес надає змогу привносити зміни до уже існуючих елементів навчального плану. Редагування також може виконуватись на двох рівнях.

Рівень студента дозволяє редагувати елементи розкладу виключно для себе.

Рівень викладача вносить зміни на рівні груп, тобто змінює розклад для всіх користувачів-студентів, що мають відношення до цього елементу.

Створення нових елементів навчального розкладу

Процес забезпечить функцію створення нових елементів розкладу, таких як консультації, додаткові практики, тощо. Створення нових елементів також може виконуватись на двох рівнях.

Рівень студента дозволяє створювати події лише для себе, залишаючи їх невидимими для всіх інших користувачів системи.

Рівень викладача створює нові елементи на рівні груп, тобто вносить зміни для всіх користувачів-студентів відповідної групи.

Функція сповіщення про зміни

					ІК1102.0414.002 ПЗ	Лист
						42
Змн.	Аркуш	№ докум.	Підпис	Дата		

Процес сповіщення про зміни усіх користувачів, які «зв'язані» з елементом що підлягає редагуванню. Реалізований шляхом push-сповіщень. Функція несе в собі автоматизований характер і спрацьовує при зміні даних на рівні викладача.

3.4. Особливості побудови мобільного додатку

Будь який додаток в наш час повинен бути мульти-локалізованим, тобто має підтримувати усі мови, якщо немає можливості підтримувати усі, то потрібно намагатися зробити додаток якнайбільш локалізованим. Також дуже актуальною є проблема вірного кешування, тобто щоб у фоні виконання програми зберігалися всі можливі зміни, для того щоб користувач не втратив свої не збережені дані чи щоб при вході у офлайн режимі він зміг отримати останню побачену ним “свіжу інформацію”. Ці дві вимоги є дуже важливими і в даному підпункті цього розділу ми розповімо вам про проблематику даної задачі, про методи вирішення та про спосіб вирішення який ми вибрали.

Існує багато методів локалізації IOS додатку, але найбільш поширених два. Перший – створення двох або більше інтерфейсів – а саме таку кількість інтерфейсів яка буде дорівнювати кількості мов які необхідно підтримувати. В цього метода є дуже багато недоліків і він раніше використовувався усіма розробниками так як не мав жодних аналогів і був по факту лише єдиним можливим варіантом вирішення даної задачі не враховуючи створення різних додатків на різних мовах. Цей варіант потребує дуже великих затрат часу та енергії на створення, а також він є не оптимізованим, так як існує дуже багато мов, і якщо ми бажаємо щоб наш додаток був унікальним є не раціональним створення такої великої кількості інтерфейсів.

Другий метод є найбільш оптимальним – а саме, це створення так званих локалізованих рядків які ми вкладаємо в поля в нашому інтерфейсі незалежно від типу елементу(текстове поле, текст, лейбл тощо). Задля

					ІК1102.0414.002 ПЗ	Лист
						43
Змн.	Аркуш	№ докум.	Підпис	Дата		

цього створюється один файл в якому всі потрібні рядки можна перевести для багатьох варіантів. Найбільш поширеною кількістю мов які підтримує додаток є три – мова регіону на який орієнтований додаток, англійська мова(як найбільш інтернаціональна) та ще одна будь яка мова як додаткова.

Провівши аналіз нами було вибрано другий метод за його мобільність та легкість у використанні.

Також дуже важливим є так зване кешування. Воно потрібне для того, щоб користувач не замислювався чи збереглись його дані, при переході між контролерами чи при згортанні до фону нашого додатку. Кешування може бути різного рівня, в залежності від задачі, тому що якщо зробити його усюди то програма може не встигати за діями користувача. Взагалі, кешування – це так зване збереження даних програми всередині самого мобільного пристрою незалежно від статусу чи доступності зовнішнього серверу. Провівши аналіз нашої підсистеми було вирішено робити кешування при згортанні додатку(так званому виході, для юзера, тому що у мобільному пристрої додаток перебуває у робочому стані у фоні ще деякий проміжок часу) та при переході від головних контролерів до допоміжних. Таким чином ми не будемо дуже сильно навантажувати фоновий потік та зможемо зберігати дані користувача у всіх необхідних моментах та переходах

Висновки до розділу

Перша частина розділу містить опис технологій та програм, що використовувались в розробці мобільного додатку «Розклад» для IOS-платформ.

Наступний розділ містить опис архітектури всього додатку, що включає в себе схему взаємозв'язку усіх внутрішніх елементів програми. Відповідна архітектура зображена на рис.3.1. Також розділ включає в себе

повну архітектуру підсистеми «клієнт-серверного» типу, що зображена на рис.3.2.

Розділ 3.3. присвячений детальному опису кожного з процесів та функцій мобільного додатку.

Останній розділ описує принципи та особливості побудови мобільного додатку. Таким чином піднялись питання мульти-локалізації та кешування, що зробить додаток більш зручним, гнучким та дасть змогу використовувати його без підключення до мережі інтернет.

					ІК1102.0414.002 ПЗ	Лист
						45
Змн.	Аркуш	№ докум.	Підпис	Дата		

4. ВПРОВАДЖЕННЯ ТА ПРАКТИЧНЕ ЗАСТОСУВАННЯ ПІДСИСТЕМИ

Темою мого дипломного проекту є створення IOS додатку, тож логічно буде показати практичне застосування саме цього додатку. Шляхом моделювання реальної ситуації, коли користувач запускає наш додаток вперше, та користується усіма доступними йому функціями. Нашим додатком будуть користуватися як студенти так і викладачі, але основною цільовою аудиторією є все ж перші, так що правильним буде показати приклад використання додатку звичайним студентом Київського Політехнічного Інституту.

Для початку, нам потрібно змодельовати ситуацію що наш додаток і створене API вже інтегровані у систему Київського Політехнічного Інституту “Електронний кампус”. Для користування нашим додатком, як вже було сказано раніше, потрібно мати свій унікальний акаунт і пароль від системи “Електронний кампус”. Дізнатися його можна в свого куратора, чи в самому конструкторському бюро. Тобто, по-перше, нам потрібно авторизуватися до системи маючи свої персональні дані. Для цього на першому екрані нашого додатку(рис. 4.1) є всі необхідні поля які користувач має заповнити.

Далі можливо два сценарія розвитку подій. Якщо користувач ввів не вірні логін чи пароль, система не дозволить йому користуватися додатком, та, звичайно, сповістить користувача про невірно введені дані за допомогою спеціального рор-уп вікна(рис 4.2). У випадку якщо користувач ввів усі дані вірно, йому буде запропоновано два варіанти для подальшої роботи з додатком. Перший варіант – зберегти логін і пароль для цього мобільного пристрою. Якщо ви користуєтесь додатком зі свого пристрою, вводити логін та пароль для авторизації при кожному запуску є дуже незручним та займає багато часу. Саме тому додаток запропонує зберегти їх, та входити у систему в наступний раз автоматично. Для людей які

користуються додатком не зі свого мобільного пристрою, коли додаток запропонує перший варіант, достатньо просто натиснути кнопку “ні”, у цьому випадку ніякі персональні дані не будуть збережені на мобільному пристрої та при наступному користуванні потрібно буде знову вводити логін та пароль. Ці дії показані на рис 4.3.

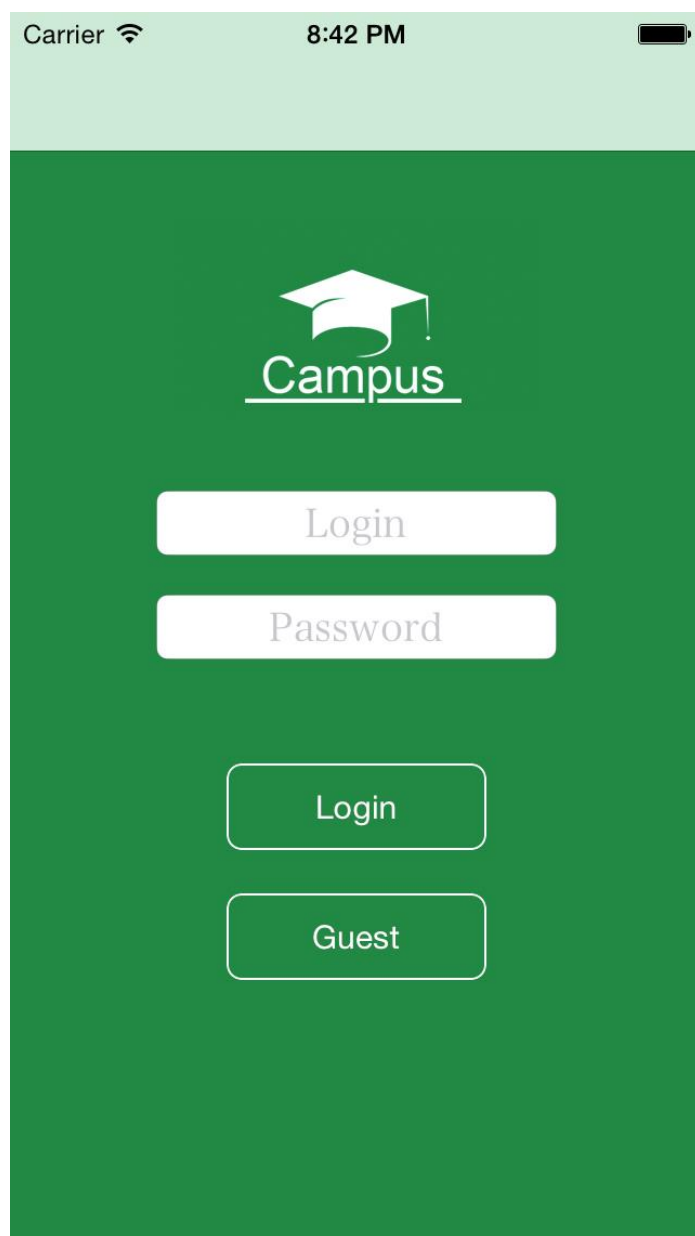


Рисунок 4.1. Контролер для авторизації користувача

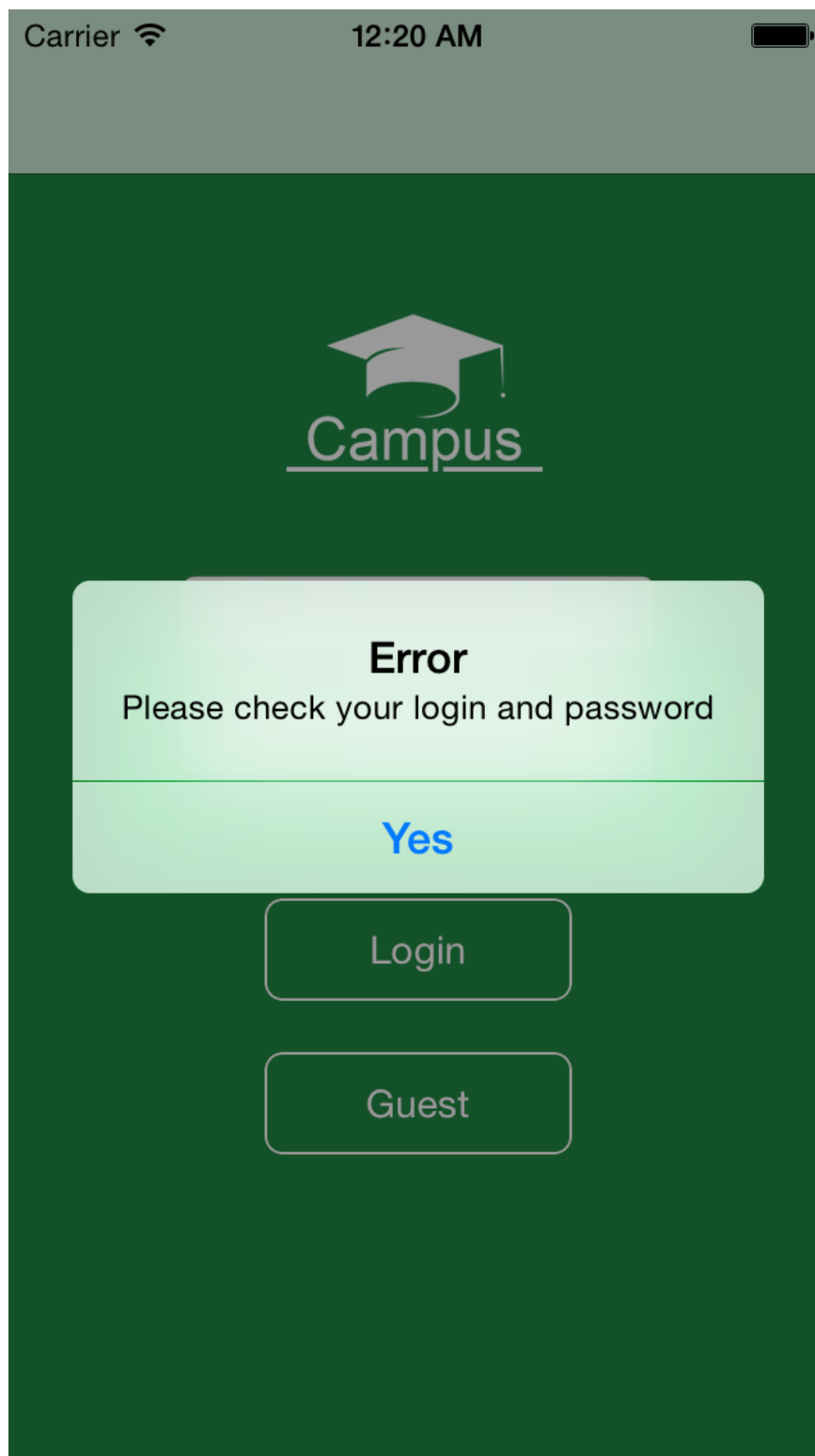


Рисунок 4.2. Pop-up вікно невірної авторизації

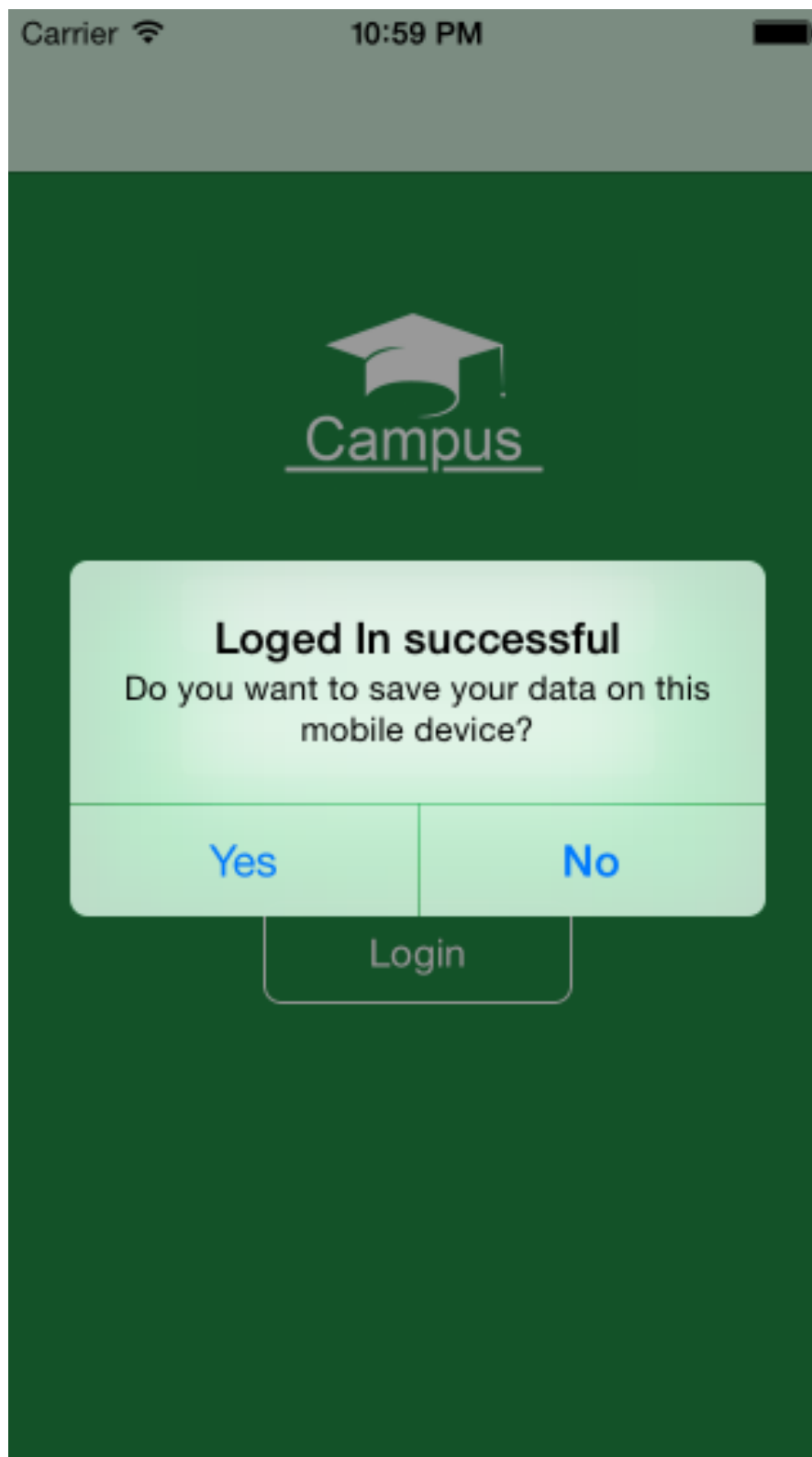


Рисунок 4.3. Pop-up вікно вірної авторизації

Після успішної авторизації користувач переходить на головне вікно нашого додатку – меню(рис 4.4). У меню є багато напрямків та функцій, про які ми хочемо вам розповісти більш детально. Також, в минулому

					ІК1102.0414.002 ПЗ	Лист
						49
Змн.	Аркуш	№ докум.	Підпис	Дата		

абзаці не було сказано про те, що якщо навіть користувач зберіг свої персональні дані на мобільному пристрої, нічого не заважає йому стерти їх за допомогою кнопки меню “Log out”. Після цього йому буде надана можливість ще раз ввести логін пароль та продовжити роботу з додатком. Цей функціонал створений для того щоб користувач мав можливість дати товаришу зайти у свій аккаунт з його мобільного пристрою, якщо є така необхідність.

Тепер розглянемо кожний пункт меню детальніше. Безперечно найбільш важливим пунктом меню є “Schedule” або “Розклад”. При натисканні користувача на відповідну кнопку меню з’являється новий контролер з таблицею розкладу студента(або викладача, на даному прикладі – студента)(рис 4.5).

У кожного розділу таблиці є свій заголовок с датою на яку показаний розклад У цьому контролері є декілька важливих функцій. Основна з них – це перегляд розкладу та планування. Про детальний опис ми розповімо пізніше. У верхній частині контроллера знаходиться дві кнопки, це “Add time” і “Choose date”. Сплануємо перший сценарій, якщо користувач захоче вибрати будь яку дату та перейти до неділі з цією датою. Нажавши кнопку “Choose date” користувач переходить на новий контролер на якому існує тільки два елемента – вибір дати та кнопка підтвердження(рис. 4.6) Після вибору дати обов’язково треба натиснути кнопку “вибір дати” для того щоб система отримала необхідні дані та зробила відповідні зміни. Після того, повернувшись на головний контролер, на якому вже відбулись необхідні зміни та відображається вибрана користувачем дата(рис. 4.7).

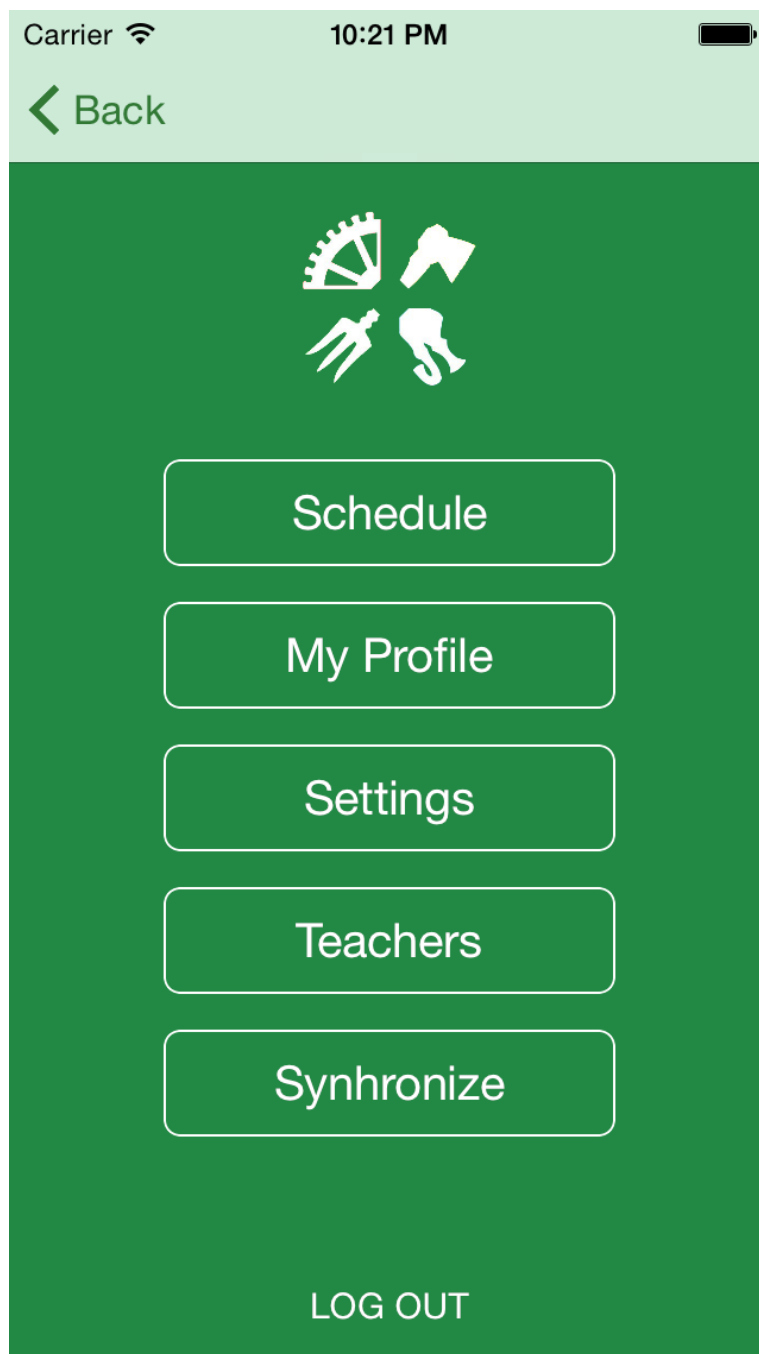


Рисунок 4.4. Головне меню

					ІК1102.0414.002 ПЗ	Лист
						51
Змн.	Аркуш	№ докум.	Підпис	Дата		

Carrier
5:05 PM

Back

Add time
Choose day

Понеділок 25.05

1 Основи охорони праці

8:30
10:05 Сидорченко Іванна Павлівна

Аудиторія :307-18 Лекція

2

Управління технічними системами - 2.Цифрові системи управління

10:25
11:55 Іваненко Олександр Андрійович

Аудиторія :418а-18 Лекція

3

Комп'ютерні навчаючі системи

12:20
13:40 Дмітрієва Антоніна Андріївна

Аудиторія :419-18 Лекція

Рисунок 4.5. Вікно “Розклад”

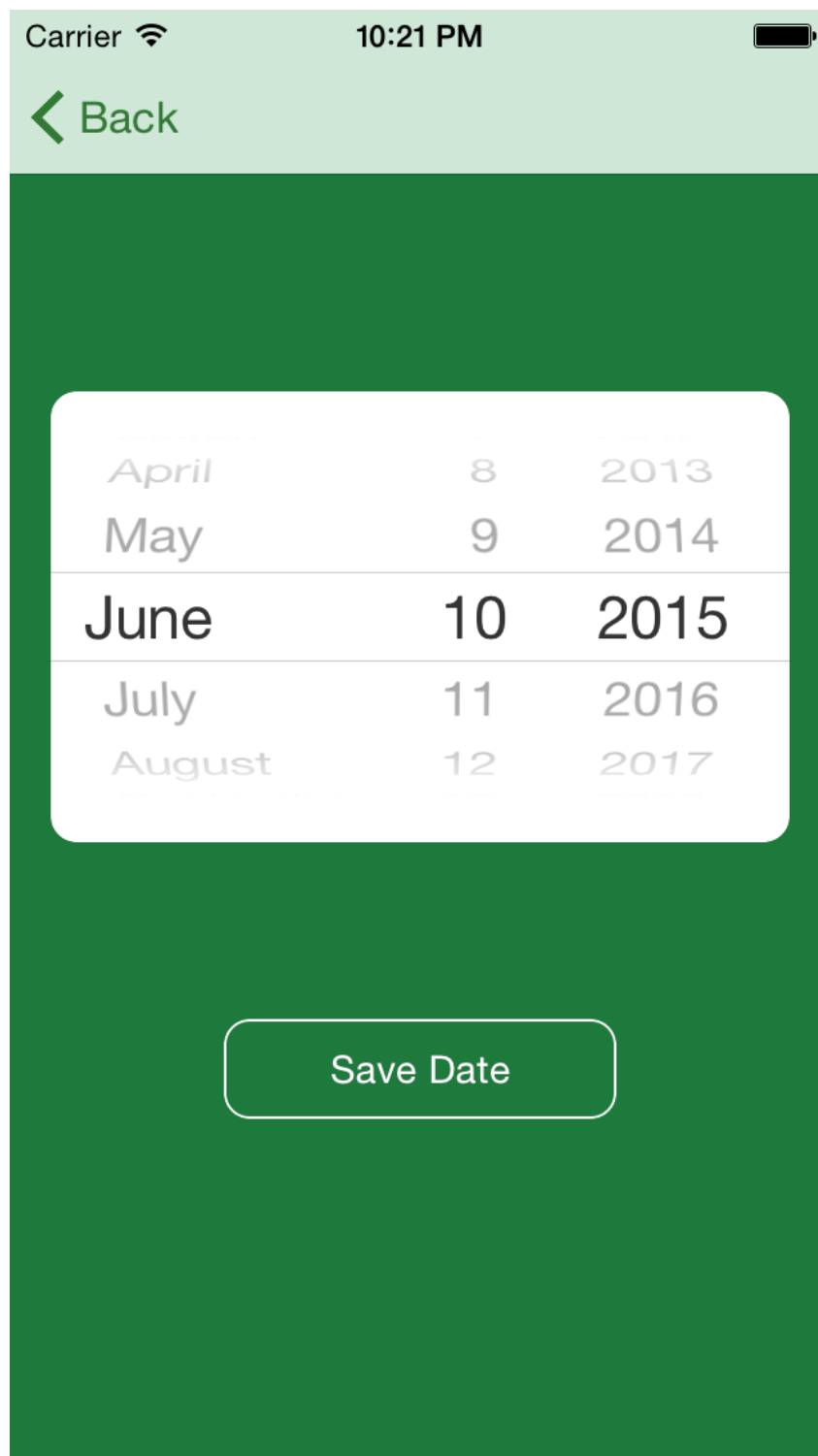
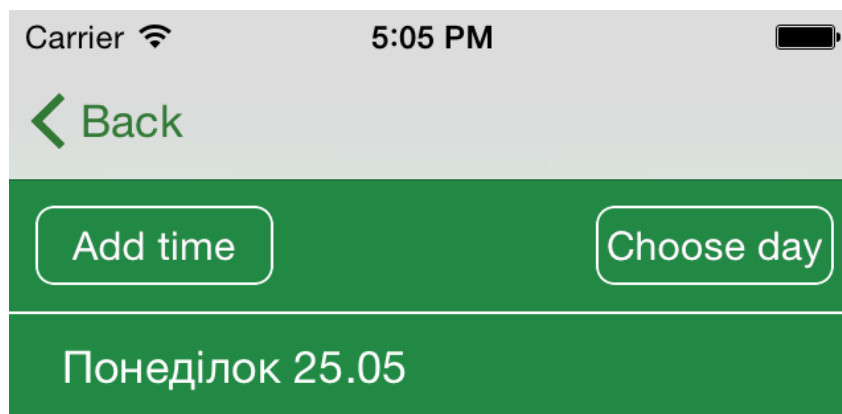


Рисунок 4.6. Вікно “Вибір дати”



① Основи охорони праці

8:30
10:05 Сидорченко Іванна Павлівна
Аудиторія :307-18 Лекція

② Управління технічними системами - 2.Цифрові системи управління

10:25
11:55 Іваненко Олександр Андрійович
Аудиторія :418а-18 Лекція

③ Комп'ютерні навчаючі системи

12:20
13:40 Дмітрієва Антоніна Андріївна
Аудиторія :419-18 Лекція

Рисунок 4.7. Вікно “Розклад” зі змінами

Отримавши необхідну дату користувач скоріше всього забажає додати яку-небудь подію, наприклад консультацію. Для цього, йому

потрібно натиснути кнопку верхнього меню контролера з розкладом “Add time”(рис 4.8).



Рисунок 4.8. кнопка “Add time”

Після натискання клавіші користувач потрапляє до нового контролера де користувачу пропонується задати всі необхідні дані у відповідних полях. Поля які повинні бути заповнені не обов’язково мають відповідну позначку(рис 4.9).

Після того як користувач заповнив усі необхідні поля, йому дається можливість вибрати день тижня на який буде поставлена ця подія. Після вибору тижня користувач натискає кнопку збереження та повертається на попереднє вікно і бачить зміни. Для прикладу якщо користувач – викладач, у попередньому меню буде також можливість сповістити групу для якої створено консультацію або подію про зміни. На цьому основний функціонал нашого додатку скінчився, але у нас також є додаткові можливості які полегшують життя користувачу. Наприклад – пункт меню “Teachers”. У цьому пункті користувач має можливість передивитися викладачів які викладають на даний момент у цього студента(рис 4.10). Також є пункт “Settings” у якому користувач має змогу вибрати мову меню, тому що важко створити додаток на одній мові яка буде задовольняти всіх загалом користувачів. Вибравши відповідну мову, усі пункти меню та інтерфейсу додатку будуть перекладені на цю мову. У майбутній версії користувач буде мати можливість міняти кольорову гамму додатку, що додасть більший вибір з приводу смакових уподобань наших користувачів.

Carrier
10:21 PM

Back

Name :

Place :

Type :

Teacher :

Pick date :

18.05
19.05
20.05
21.05
22.05

Save Changes

Рисунок 4.9. вікно “Add time”

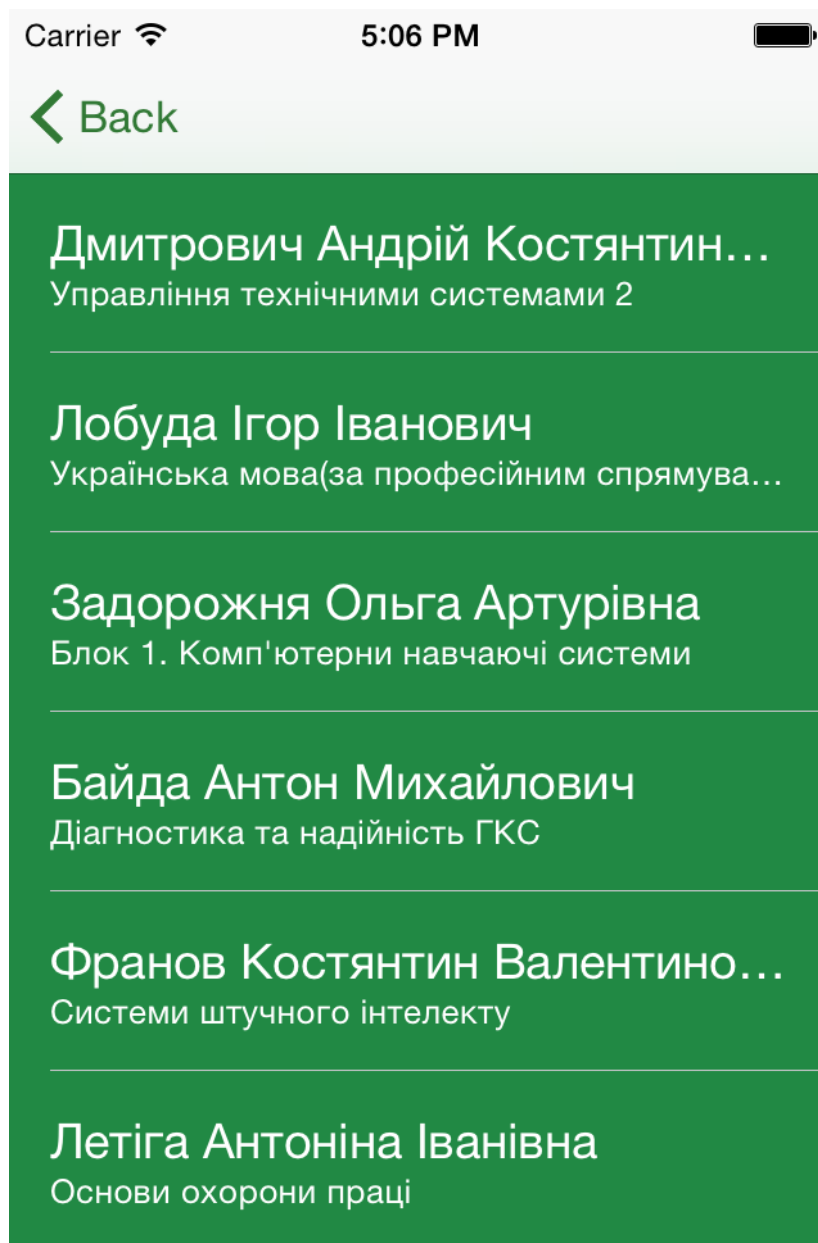




Рисунок 4.10. вікно “Teachers”

Наступним важливим пунктом є перегляд та можливе редагування свого профілю. Натиснувши кнопку “Profile” користувач переходить до вікна профілю(рис 4.11), де може натиснувши кнопку “Edit” перейти на


					ІК1102.0414.002 ПЗ	Лист
						57
Змн.	Аркуш	№ докум.	Підпис	Дата		

нове вікно та редагувати тільки дозволені поля. Закінчивши усі зміни потрібно натиснути кнопку “Save” та можна продовжувати роботу з додатком

Carrier  10:21 PM 

 Back





Name: Загорський Павло Михайлович
Position : Штатний Викладач
Academic Degree : Кандидат наук
Subdivision Name : Кафедра технічної кібернетики ФІОТ

Рисунок 4.11. вікно “Profile”

Останньою, але дуже важливою функцією, є можливість синхронізації з сервером. Детальніше, це означає що незалежно від того який розклад має користувач на своєму мобільному пристрої на даний момент, натиснувши кнопки синхронізації користувач залежно від того, де буде останнє оновлення, отримає або версію з сервера, або навпаки відправить усі свої дані на сервер, що гарантуватиме їх надійність та зробить їх захищеними від втрати через будь які можливі причини.

Висновки до розділу

В даному розділі дипломного проекту було розглянуто, на реальному прикладі, практичне використання створеного IOS додатку. Тут було розказано та показано всю послідовність дій, яка необхідна для використання додатку а також були надані різні сценарії можливих дій користувача. Таким чином даний розділ можна, також, використовувати як інструкцію та документацію до додатку для користувачів які завантажили його уперше та не мають певного досвіду у користуванні додатками такого типу.

Отже, даний додаток значно спрощує планування та перегляд розкладу та робить навчальний процес більш комфортним як для звичайних студентів так і для викладачів. Функціонал додатку не ускладнює життя користувачу своєю кількістю та водночас є оптимальним для системи даного типу.

5. ОХОРОНА ПРАЦІ

Дипломна робота на тему «Розробка інформаційного та програмного забезпечення підсистеми Електронного Кампусу "Розклад" з підтримкою мобільних платформ. Розробка додатку для iOS платформ».

Суб'єктом в даному розділі є інженер, який виконує проектування світлодіодних світильників з використанням персонального комп'ютера. Передбачається, що місцем його роботи буде комп'ютерна лабораторія на підприємстві, що розрахована на 3х осіб.

Робота на комп'ютері може мати негативний вплив фізичних чинників, призвести до серйозних проблем фізичного та психологічного стану.

Метою цього розділу є аналіз умов безпеки праці на обраному робочому місці, виявлення шкідливих і небезпечних факторів виробничого середовища і порівняння їх з діючими нормативами, а також розробка заходів, націлених на утворення умов праці, що відповідають вимогам усіх нормативно-правових актів з охорони праці.

Впровадження заходів з охорони праці дозволить гарантувати працівнику збереження його здоров'я та працездатності.

					ІК1102.0414.002 ПЗ	Лист
						60
Змн.	Аркуш	№ докум.	Підпис	Дата		

5.1. Характеристика об'єкту та умови його експлуатації

Робоче місце суб'єкта знаходиться в одній із комп'ютерних лабораторій підприємства, яка обладнана для роботи трьох інженерів. Лінійні розміри становлять 7м×5,5м, висота стелі 2,8м. У приміщенні, використовується змішане освітлення. Стіни пофарбовані в світло-жовтий колір а на підлозі лежить світлий паркет. Спрощений план приміщення приведено на рис. 5.1.

Як основні характеристики приміщення приймаються його геометричні розміри і кількість працюючих у ньому людей. Розміри аналізованого приміщення приведені в табл. 5.1.

Таблиця 5.1 Розміри приміщення

Найменування	Позначення	Значення, м
Довжина	A	7
Ширина	B	5,5
Висота	H	2,8

Таблиця 5.2 Площа та обсяг приміщення, на одного працюючого

Геометрична характеристика	Одиниця виміру	Нормативне значення	Фактичне значення
Площа, S	м ²	не менш 6.0	9,6
Обсяг, V	м ³	не менш 20	26,95

За даними, приведеним у табл. 5.2, можна зробити висновок, що геометричні розміри приміщення відповідають нормативним вимогам згідно ДНАОП 0.00-1.31-99 «Правила охорони праці під час експлуатації електронно-обчислювальних машин».

Основним робочим положенням є положення сидячи. Головними елементами робочого місця є письмовий стіл, крісло і комп'ютер.

З меблів в лабораторії знаходяться чотири столи для комп'ютерів, чотири крісла, дві шафи з документацією, та стіл для розташування іншої техніки.

З техніки тут розташовані чотири персональні комп'ютери, принтер та факс.

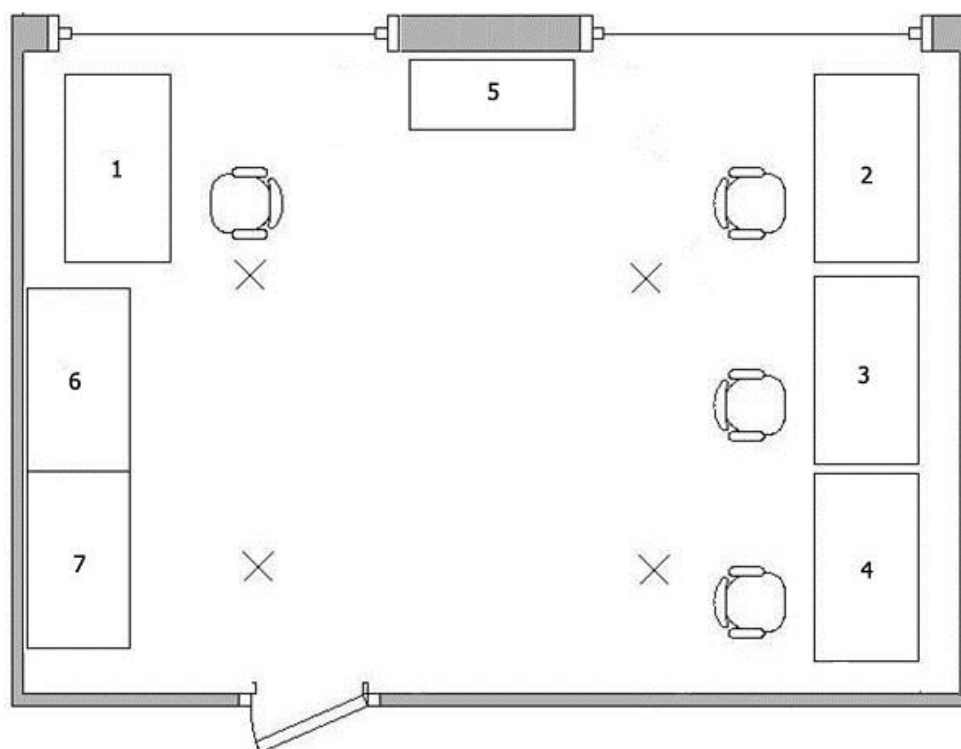


Рис. 5.1 Спрощений план приміщення

1,2,3,4 – робочі місця з комп'ютерами; 5 – стіл для принтера та факсу;
6,7 – шафи.

Розглянемо робоче місце користувача ПК з точки зору оцінки впливу шкідливих виробничих факторів відповідно до Гігієнічного нормативу ГН 3.3.5-8-6.6.1-2002 «Гігієнічна класифікація праці за показниками шкідливості та небезпечності факторів виробничого середовища, важкості та напруженості трудового процесу». Відповідно до цього документу, на працівника, який працює з комп'ютером діють такі шкідливі виробничі чинники:

1. Мікроклімат робочої зони;
2. Недостатність штучного освітлення;
3. Виробничий шум;
4. Виробничі випромінювання;
5. Пожежонебезпека.

5.1.1. Мікроклімат робочої зони

Відповідно до встановлених гігієнічно-санітарних вимог (ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень») роботодавець зобов'язаний забезпечити в приміщеннях для даного типу роботи (категорія Легка – 1а) оптимальні параметри виробничого середовища. Параметри мікроклімату згідно з нормами повинні бути наступними:

Таблиця 5.3 Норми мікроклімату для приміщень з ЕОМ

Пора року	Параметр мікроклімату	Оптимальне значення	Фактичне значення
Холодна	Температура повітря	22-24°C	23°C
	Відносна вологість повітря	60 - 40%	31%
	Швидкість руху повітря	0,1м/с	0,1м/с
Тепла	Температура повітря	23 - 25°C	24,5°C
	Відносна вологість повітря	60 - 40%	58%
	Швидкість руху повітря	0,1 м/с	0,1м/с

Причиною підвищеної температури робочої зони можуть бути освітлювальні пристрої, величина тепловиділення яких становить 35-60

Вт/м², а також комп'ютер, середня величина тепловиділення якого становить 310 Вт/м².

5.1.2. Освітлення

Причиною недостатності природного освітлення може бути неправильно спроектоване розміщення робочого місця відносно джерел природного освітлення або ж слабе світлопроникнення вікон через їх забрудненість.

Основним документом, який регламентує норми освітленості є ДБН.В.2.5-28-2006 «Природне і штучне освітлення». Освітлення у приміщеннях, де знаходиться робоче місце працівника, використовується змішане.

У якості природнього в даному приміщенні представлено одностороннє бокове освітлення через два вікна розміром 2,5м×1,5м . Напрямок розміщення вікон східний. Коефіцієнт природної освітленості ~ 1,7%.

Для штучного освітлення в подібних приміщеннях необхідно використовувати джерела світла з досить великим ККД у світильниках, які розташовуються над робочими поверхнями у рівномірно – прямокутному порядку. Найкраще підходять в таких приміщеннях світлодіодні(LED) лампи, які мають один з найвищих показників світловіддачі. У нашому випадку використовуються чотири світильника зі звичайними люмінесцентними лампами.

Штучне освітлення повинно забезпечити на робочих подібних місцях освітленість 300 – 500 лк. На робочому місці,що розглядається, фактичне значення освітленості становить 200-250 лк. Це пов'язано з нерівномірністю розміщення світильників.

При правильно розрахованому і виконаному освітленні виробничих приміщень, очі працюючого на протязі тривалого часу зберігають

					ІК1102.0414.002 ПЗ	Лист
						64
Змн.	Аркуш	№ докум.	Підпис	Дата		

здатність добре розрізняти предмети, не стомлюючись. Такі умови сприяють зниженню виробничого травматизму і професійного захворювання очей. Рациональне освітлення має задовольняти ряду вимог і умов. Воно має бути:

- достатнім, щоб очі без напруги могли розрізняти предмети;
- постійним у часі, для цього напруга в мережі живлення не повинне коливатися більш ніж на 4%;
- рівномірно розподіленим по робочих поверхнях, щоб очам не приходилося попадати з дуже темного місця у світле і навпаки;
- таким, що не здійснює сліпучу дію на око людини як від самого джерела світла, так і від поверхонь, що віддзеркалюють його та знаходяться в полі зору працюючого. Зменшення сліпучої дії джерел досягається застосуванням світильників які розсіюють світло;
- не викликати різких тіней на робочих місцях. Цього можна уникнути при правильному розташуванні світильників.

Згідно ДБН В.2.5-28-2006 «Природне і штучне освітлення» для даних робіт встановлена необхідна освітленість робочого місця $E_n = 300$ лк.

5.1.3. Виробничий шум

На комп'ютеризованих робочих місцях основними джерелами шуму є вентилятори системного блоку, принтери. Сильний шум викликає труднощі з розпізнаванням кольірних сигналів, знижує швидкість сприйняття кольорів, гостроту зору, зорову адаптацію, порушує сприйняття візуальної інформації, зменшує на 5-12% продуктивність праці.

На даному робочому місці основними джерелами шуму є вентилятори системи охолодження системного блоку комп'ютера, а також принтери та факс. Також варто врахувати шум, що надходить ззовні, і який ліквідується використанням акустичних поглиначів звуку, а також вікон, що щільно закриваються.

Для покращення робочої обстановки необхідне технічне вдосконалення та періодичне обслуговування системних систем охолодження комп'ютерів. А принтери перемістити за межі лабораторії, або помістити в звукоізоляційну коробку.

Методи вимірювання шуму регламентовано ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку».

Розрахуємо рівень шуму в приміщенні.

Таблиця 5.4. Значення рівня шуму для типових джерел

Джерело шуму	Рівень шуму, дБА
Жорсткий диск	30
Вентилятор	45
Монітор	15
Клавіатура	8
Принтер	40
Факс	45

Максимальний час роботи принтера за один день – 1,5 години.

Робочий день $T = 8$ годин.

$$L_{\Sigma} = 10 \lg(10^3 + 10^{4,5} + 10^{1,7} + 10^{0,8} + 10^4 + 10^4) = 48,7 \text{ дБА}$$

При роботі на комп'ютері рівень шуму відповідно до постанови ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку» не повинен перевищувати 50 дБА, а фактичний 48,7 дБА. Отже, наше приміщення відповідає діючим санітарним нормам.

5.1.4. Виробничі випромінювання

Обладнання робочого місця за усіма вимогами і наявність сертифікованого персонального комп'ютера не дає повної гарантії електромагнітної безпеки користувача, навіть за умови використання сучасного рідкокристалічного монітора. Однак для підвищення захищеності користувача комп'ютера від прямого та опосередкованого впливу електромагнітних полів та випромінювань необхідно розглянути увесь інформаційно-технічний комплекс з точки зору надійності функціонування.

Вимірювання полів від персональних комп'ютерів показали, що сучасні сертифіковані відео монітори в основному відповідають вимогам чинних нормативних актів з електромагнітної безпеки. Однак при невірному взаємному розташуванні моніторів рівні полів на робочих місцях можуть перевищувати гранично допустимі.

Сумарне електромагнітне поле з боку сучасних рідкокристалічних моніторів значно менше за гранично допустиме. Проте у діапазоні 2-400 кГц мають місце досить великі рівні полів на частотах 150-200кГц.

Крім електромагнітних полів та випромінювань безпосередньо від монітора, на користувача додатково впливають так звані фонові поля – поля від сторонніх джерел, які знаходяться у приміщенні або поблизу від нього. Такими джерелами є мережі живлення і освітлення, побутові прилади (кондиціонер, обігрівач), мобільні телефони, бездротова мережа тощо. Досліди показали, що рівні напруженості полів на робочих місцях, розташованих, наприклад, поблизу працюючих кондиціонерів, збільшується на 15-20%. Значне зростання полів також спостерігається у просторах між масивними металевими предметами та комп'ютерами.

					ІК1102.0414.002 ПЗ	Лист
						67
Змн.	Аркуш	№ докум.	Підпис	Дата		

5.1.5. Пожежна безпека

Пожежну та вибухову безпеку регламентують Закон України «Про пожежну безпеку», а також ОНТП 24-86 «Визначення категорій приміщень і будівель по вибухопожежній і пожежній небезпеці»), які є обов'язковим для виконання всіма підприємствами незалежно від форми власності. Правила встановлюють загальні вимоги з пожежної безпеки.

Безпека людей має здійснюватися при виникненні пожежі в будь-якому місці виробничої будівлі, споруди або територій підприємства. При виникненні пожежі на людей можуть впливати небезпечні чинники: відкритий вогонь та іскри; підвищена температура повітря, предметів, обладнання; токсичні продукти горіння, дим; знижена концентрація кисню; обвалення і пошкодження будівель, споруд, установок, вибух.

Основними причинами пожежі та вибуху на підприємствах є наступні:

- несправність виробничого обладнання;
- несправність та перенавантаження електричного обладнання;
- необережне ставлення до вогню (паління, використання відкритого вогню в недозволених місцях, залишання без нагляду електрообладнання);
- порушення правил пожежної безпеки.

В приміщенні класу «В», що розглядається, повинно бути встановлена система пожежної сигналізації з димовими пожежними сповіщувачами (з розрахунку 2 шт. на кожні 20 м² площі приміщення) та переносні порошкові вогнегасники (для даного приміщення достатньо одного на лабораторію).

У сучасних комп'ютерах повинні бути передбачені всі захисні заходи щодо пожежі:

- монітори захищені від вибуху та займання;

					ІК1102.0414.002 ПЗ	Лист
						68
Змн.	Аркуш	№ докум.	Підпис	Дата		

- на процесорах стоять захисники, що виключає можливість займання процесора;
- заземлення.

Отже, пожежна та вибухова безпека забезпечується:

- використанням методів та пристроїв запобігання іскріння;
- своєчасним контролем за справним станом обладнання;
- систематичною очисткою вентиляційних каналів від пилу і перевіркою системи вентиляції;
- підтримкою чистоти та порядку всередині приміщення;
- відсутністю всередині приміщення легкозаймистих та вибухових речовин;
- застосуванням запобіжників;
- дотриманням протипожежних вимог до електрообладнання;
- захистом від блискавки будинку і устаткування відповідно;
- використанням пожежної сигналізації.

5.2. Інструкції з техніки безпеки

Дана інструкція діє для персоналу, що експлуатує комп'ютери та периферійне обладнання, а також побутові електроприлади (електрочайники, кавоварки тощо). Інструкція містить загальні вказівки щодо безпечного застосування електрообладнання в організації. Вимоги даної інструкції є обов'язковими, будь-які відхилення від неї є недопустимими. До самостійної експлуатації комп'ютерів та електроапаратури допускається лише спеціально навчений персонал.

Вимоги до безпеки:

- Перед початком роботи переконайтесь у:
 - справності електропроводки вимикачів

					ІК1102.0414.002 ПЗ	Лист
						69
Змн.	Аркуш	№ докум.	Підпис	Дата		

- справності штепсельних розеток, за допомогою яких обладнання підключається до мережі
 - наявності заземлення комп'ютера
 - справності комп'ютера та периферійних засобів.
- Задля запобігання пошкодження ізоляції проводів та виникнення короткого замикання забороняється: вішати будь-що на дроти, засовувати дроти та шнури за водопровідні труби, за батареї опалювальної системи, висмикувати штепсельну вилку з розетки за дріт (зусилля мають бути прикладені до корпусу вилки).
- Щоб запобігти враженню електричним струмом забороняється:
- часто вмикати та вимикати комп'ютер без необхідності;
 - торкатись деталей комп'ютера та периферійного обладнання вологими руками;
 - працювати з комп'ютером та периферійними пристроями, якщо вони мають порушення цілісності корпусу, порушення ізоляції дротів, несправну індикацію живлення, з ознаками електричної напруги на корпусі;
 - під напругою проводити ремонт комп'ютерів та периферійного обладнання. Ремонт електроапаратури проводять тільки спеціалісти-техніки з дотриманням необхідних технічних вимог;
 - очищувати від пилу та забруднення електрообладнання, коли воно знаходиться під напругою;
 - перевіряти справність електрообладнання в непристосованих для експлуатації приміщеннях зі струмопровідною підлогами, вологих, таких, що не дозволяють заземлити доступні металеві частини;

- при користуванні електроприладами торкатись одночасно трубопроводів, батарей опалення, металевих конструкцій, що з'єднані з землею.
- Після закінчення роботи необхідно знеструмити всі комп'ютери, периферійне обладнання та електроприлади.
- У випадку неперервного виробничого процесу дозволяється залишити ввімкненим тільки необхідне обладнання.

Висновки до розділу

У результаті проведеного аналізу умов безпеки праці на робочому місці працівника були виявлені шкідливі і небезпечні фактори, а також визначені та запропоновані варіанти вирішення його недоліків.

Так для покращення освітлення запропоновані нові та значно ефективніші світлодіодні лампи, яскравість яких складає 2520 лм, що нічим не гірше від люмінесцентних ламп, але строк служби яких при цьому довший в 5 – 10 разів. До того ж вони стійкі до механічних пошкоджень та низьковольтні, а значить – безпечніші.

Також було проведено розрахунок рівня шуму в приміщенні та було встановлено, що він задовольняє норми.

Окрім цього були розглянуті інструкції з охорони праці, питання пожежної безпеки та визначено необхідні умови для її забезпечення.

ВИСНОВКИ ДО РОБОТИ

Перший розділ присвячений існуючим аналогам систем для роботи з поточним навчальним розкладом, було проведено аналіз та визначено їх переваги та недоліки, а також проведено порівняльний аналіз системи яка була розроблена. Розділ завершується конкретно поставленою задачею дипломного проектування.

Далі, у другому розділі, на основі досліджених аналогів були розроблені вимоги до підсистеми «Розклад». Вимоги до функціоналу були конкретно представлені в табл. 2.1. Також були описані вимоги до архітектури та інтерфейсу.

Третій розділ включає в себе безпосередньо процес розробки підсистеми: опис та реалізацію архітектури підсистеми в цілому, архітектуру мобільного додатку, а також інформацію про технології та ПО що були використані при розробці. Також, описуються всі процеси, що відбуваються в системі.

В четвертому розділі, на прикладі було показано практичне використання додатку. Як результат, підсистема «Розклад» дуже корисна для студентів та викладачів, вона забезпечує швидку та зручну роботу з поточним навчальним планом. Даний розділ, цілком, може претендувати на вичерпну інструкцію по використанню мобільного додатку.

Розділ присвячений охороні праці описує основні шкідливі фактори та чинники, що негативно впливають на здоров'я людей, які працюють на ЕОМ з даною системою. Також було приведено інструктаж з техніки безпеки роботи в кімнаті з ЕОМ.

На основі дипломного проектування було розроблено систему для роботи з поточним навчальним розкладом, а саме мобільний додаток для платформи IOS. Основні переваги, якого: автоматизація роботи з поточним розкладом, зручний та швидкий перегляд інформації, редагування даних, можливість сповіщення інших користувачів, доступність системи з вашого

мобільного, здатність працювати з даними в режимі офлайн. Недоліком даного додатку є обмеження можливостей використання при відсутності доступу до мережі інтернет.

Отже, розроблена система може претендувати на систему для роботи з поточним навчальним розкладом, яка забезпечує легку та зручну роботу у будь який час.

Також, у майбутньому, для покращення підсистеми в цілому можна розширити функціонал, додати синхронізацію з календарем, дати користувачам можливість розшарювати файли за допомоги хмарних сховищ, таких як Dropbox, надати можливість створювати групові чати та інше. Розробка мобільного додатку під платформи Windows Phone є пріоритетною задачею. Але, це є досить складним, адже потребує багато часу на розробку та впровадження нового функціоналу.

					ІК1102.0414.002 ПЗ	Лист
						73
Змн.	Аркуш	№ докум.	Підпис	Дата		

ПЕРЕЛІК ПОСИЛАНЬ

1. «Комплекс методичних вказівок до виконання дипломних проектів»: підручник / [авт.кільк.: М.М. Поліщук, М.М.Ткач, В.П. Пасько, О.І. Чумаченко, О.І. Лісовиченко, О.А. Стенін], - Київ: Дорадо-Друк.2014.
2. Alex Berson. CLIENT/SERVER ARCHITECTURE, 2nd edition. McGraw-Hill,1996. – 569 с.
3. Jeff McWherter. Professional Mobile Application Development. Wrox, 2012 – 390с.
4. Mark L. Murphy. The Busy Coder's Guide to Android Development. CommonsWare, 2013. – 2092 с.
5. «The Swift Programming Language.»
6. Отрывок из книги: Apple Inc. «The Swift Programming Language». iBooks.
7. <https://itun.es/ua/jEUN0.1 SQLite>. [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/SQLite>
8. BitBucket. [Електронний ресурс]. – Режим доступу: <http://ru-wiki.org/wiki/Bitbucket>
9. Інструкція з охорони праці, техніки безпеки, пожежної безпеки при роботі з персональним комп'ютером. [Електронний ресурс]. – Режим доступу: <http://tr.su.court.gov.ua/sud1818/ohorona/>
- 10.Санітарні норми виробничого шуму, ультразвуку та інфразвуку ДСН 3.3.6.037-99 від 01.22.99. Міністерство охорони здоров'я України, 1999.
- 11.Закон України «Про пожежну безпеку». – К., 1993.
- 12.Жидецькій В.Ц., Джигирей В.С., Мельников О.В. Основи охорони праці. – Львів: Афіша, 2000-350 с.