# 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 навчальної інформаційної системи у вигляді окремих підсистем. Таким чином, тут реалізовані підсистеми дистанційного навчання, соціальної мережі на рівні «КПІ», а також автоматизації 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 планом з підтримкою мобільних платформ (AndroidOS 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 учбового плану як для студентів, так і для викладачів.
* Надавати завжди поточні дані, цим самим зменшити кількість помилок та неточностей.
* 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.

# 

# АНАЛІЗ ІСНУЮЧИХ СИСТЕМ РЕЙТИНГУ ДИСТАНЦІЙНИХ ЛАБОРАТОРНИХ СТЕНДІВ

## Аналіз прототипу системи оцінки дистанційних лабораторних стендів

Лабораторні стенди є істотним і основним компонентом всіх систем дистанційного навчання. Своєчасна оцінка лабораторних стендів займає центральне місце в забезпеченні ефективного навчання студентів. Крім того, для вдосконалення цих стендів необхідна достовірна інформація про лабораторний стенд. У багатьох випадках вибір методу навчання значною мірою залежить від даних наукових лабораторій.  
Лабораторії безумовно відіграють важливу роль у навчанні студента, у зв'язку з цим моніторинг та оцінка лабораторних потужностей потребують стандартизованого підходу і методології.

Оскільки на сьогоднішній день великої популярності набуває дистанційне навчання, то темою нашого диплому стала «Система рейтингу дистанційних лабораторних стендів». На нашу думку це гарна ідея для оцінки дистанційних лабораторій щоб зрозуміти для чого вони можуть бути використані, та оцінити користь цих лабораторій у дистанційному навчанні.

Тема оцінки лабораторних стендів є не новою, вона зародилась ще в 2013 році. На просторах глобальної мережі інтернет а саме на ресурсі Github нами було знайдено цікаві матеріали які були поштовхом для реалізації цієї ідеї.

Засновником цієї ідеї можна рахувати одного із викладачів нам невідомого Київського інституту/університету. Ідея була описана вдало, проте потребувала досить не простої реалізації та не була втілена в життя.

В цих матеріалах був частково описаний та реалізований додаток оцінки лабораторії, який давав змогу користувачу оцінити які ресурси, пристрої, дидактичні матеріали наявні в лабораторії, та надіслати сформований звіт власнику лабораторії. Багато корисної інформації про лабораторний стенд такий додаток не приносив би й інформативність була б мінімальною, в плані технічному звичайно можна було б зупинитись на цьому, але з сторони методичної ніхто не звертав увагу щодо оцінки лабораторії.

На нашу думку такого функціоналу для повного аналізу та оцінки було недостатньо, тому ми вирішили створити цілу систему, яка б складалась з WIN та WEB додатку , яка б мала значно більше можливостей та головне давала б користувачу можливість оцінювати лабораторні стенди різного типу , не залежно від напрямку / галузі застосування.

|  |  |
| --- | --- |
| **Переваги** | **Недоліки** |
| Можливість формувати звіт. | Формування звіту тільки на одній мові. |
| Наявність основних критеріїв оцінки лабораторного стенду. | Кількість критеріїв занадто вузька щоб дати об’єктивну оцінку. |
| Надсилання сформованого звіту. | Надсилання сформованого звіту по статично заданим параметрам. |

## Аналіз Google Form, як засобу для оцінки дистанційних лабораторних стендів

Ще одним аналогом оцінки можна рахувати Google Forms. Звичайно слово «аналог» це дуже грубо, тому що такий ресурс як Google Forms може використовуватись не тільки для оцінки лабораторних стендів, він є занадто гнучким у використанні й спрямований для реалізації різних цілей, але все ж таки ми можемо використовувати його для оцінки лабораторних стендів тому віднесемо його до аналогів. Звичайно винаходити велосипед ми не планували й аналог розробляти теж не було в думках, проте ми вирішили зробити систему яка б використовувалась саме для оцінки дистанційних лабораторних стендів й давала б змогу піднімати дистанційне навчання на новий рівень.

|  |  |
| --- | --- |
| **Переваги** | **Недоліки** |
| Легкість у використанні. | Потребує постійних змін у налаштуванні. |
| Динамічність щодо використання. | Немає орієнтованості на певну галузь. |
| Обмеженість доступу. | Використання тільки за допомогою мережі інтернет. |

## Постановка задачі

Основними відмінностями нашої системи та нереалізованого додатку є:

* можливість формувати оцінку стенду як в офф-лайн так в он-лайн режимах;
* синхронізація WIN та WEB додатку, що дозволяє отримати доступ до результатів в будь який час та в будь якому місці де є Інтернет покриття;
* значно збільшена кількість оціночних характеристик системи, та розбиття характеристик на блоки, що є зручним для розуміння з методичної точки зору;
* можливість користування та формування звіту на декількох мовах а саме: українська, російська та англійська;
* можливість збереження звіту у різних форматах та з різним оформленням , в залежності яке є більш зручним для користувача (WIN);
* можливість ділитись уже сформованими звітами за допомогою WEB додатку системи;
* надсилання сформованого звіту на електронну адресу;
* єдина реєстрація в системі через WEB або WIN додатки.

|  |  |
| --- | --- |
| **Переваги** | **Недоліки** |
| Наявність WIN та WEB додатку , що дає змогу використання у будь який час. | Підтримуються не всі відомі формати при збереженні звіту. |
| Наявність декількох мов при використанні та при формуванні звіту. | Наявність тільки 3-х мов при використанні додатку та збереженні звіту. |
| Значно розширена кількість критеріїв оцінки що дає змогу зробити аналіз системи більш об’єктивним. |  |
| Орієнтованість саме на оцінку лабораторних стендів. |  |
| Конфіденційність інформації та , забезпечення захисту персональних даних. |  |

Завданням цього дипломного проекту є створення WEB-клієнту, який має можливість оцінювати ДЛС, та формувати відповідний рейтинг, який може сказати про якість та корисність цієї ДЛС. Перевагою, WEB-клієнту, є те, що його можна використовувати не залежно від операційної системи чи розміру екрану, та у будь-якій точці Світу.

Отже, по закінченню дипломного проектування має бути отримано WEB-клієнт, який буде відповідати всім поставленим далі вимогам, та конкурувати з існуючими аналогами.

## ВИСНОВОК ПО РОЗДІЛУ

### Оцінивши всі можливі варіанти та проаналізувавши ситуацію, нами була створена «Система оцінки дистанційних лабораторних стендів».

Основним напрямком для оцінки є університетські лабораторні стенди, наша система є гарним додатком який дозволяє оцінити функціональність, повноту та практичність наявних в університеті лабораторій що в свою чергу забезпечує можливість обміну цими лабораторними стендами для навчання студентів, адже не буває в одному університеті всього ідеального, десь краща реалізація одних напрямків, десь інших а завдяки поєднанню кращих стендів можливе створення єдиного лабораторного порталу по різним предметам.

Цей додаток має принести користь для розвитку дистанційного навчання , адже саме завдяки ньому ми можем оцінити стенд й дати зрозуміти для яких категорій саме цей чи інший лабораторний стенд буде кращим у використанні.

На сьогоднішній день система дистанційного навчання набирає великої популярності, з кожним роком створюються нові й нові ресурси для дистанційного навчання, мало того ці ресурси стають все кращими для розуміння , легшими у використанні та без сумніву приносять велику користь самостійному розвитку людини.

Основною можливістю нашої системи є оцінка саме дистанційних лабораторних стендів. Власне оцінка відбувається шляхом виставлення параметрів та можливостей ДЛС.

# ВИМОГИ ДО МОБІЛЬНОГО ДОДАТКУ «РОЗКЛАД»

Сьогодення диктує певні вимоги для всіх додатків. Останні віяння в ІТ, показують, що все більше додатків переносять у WEB, 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 додатку з будь-якої частини світу, такі додатки не навантажують комп’ютер 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 підключені синхронізувати напрацювання. Також однією з проблема 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 напрацювання синхронізувати при першому підключенні.

## 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, оповіщення інших користувачів про зміни та інше.

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.

Більш детальні вимоги по функціоналу наведено в табл. 1.

*Таблиця 1. 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. |
| Перегляд поточного навчального розкладу | | Потрібно забезпечити можливість перегляду поточного навчального розкладу для всіх користувачів «Електронного кампусу». Реалізована функція розширеного перегляду з виведенням номеру корпуса, 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. |
| Редагування існуючих елементів навчального розкладу | | Потрібно надати 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 | | Потрібно забезпечити 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. |
| 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. |

## 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 AndroidOS платформ, то даний інтерфейс побудований за «Принципами дизайну для Android», що були запропоновані представниками Google.

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

Як говорилось 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 виділятись 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 «Принципів дизайну для Android», 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.

## 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.

## ВИСНОВОК ПО РОЗДІЛУ

В даному розділі були розглянуті основні вимоги для мобільних додатків, в тому числі і підсистеми «Розклад». Ці вимоги є обов’язковими, тому що наявність таких властивостей у системи, забезпечує її зручність та практичність.

Також, була вирішено задовольнити вимогу можливості працювати з системою при відсутності інтернету, що є надзвичайно важливо, адже інколи отримати доступ до мережі просто неможливо. Особливо, зважаючи на те, що використання цієї підсистеми-додатка планується в Україні, де покриття мережею інтернет, залишає бажати кращого. Ця вимога є комплексною для підсистеми і була вирішена шляхом реалізації такого підходу – за присутності зв’язку з сервером, усі данні завантажуються до додатку і одразу дублюються у нашу локальну БД. Таким чином створюється кеш, який оновлюється і дає можливість додатку користуватися інформацією навіть в оффлайн режимі.

В розділі був розглянутий основний функціонал додатку з детальним описом кожної з можливостей підсистеми. Більшість функцій розділені правами доступу, та можуть виконувати дії у двох режимах. В результаті отримуємо широкий спектр можливостей підсистеми з мінімалізацією дій, що виконує одну з основних вимог дизайну інтерфейсу. Детальніше функціонал описаний в табл. 1, з якої можна зробити висновок про задачі всієї підсистеми в цілому.

Розділ містить в собі опис реалізації вимог до архітектури додатку відповідно з патерном проектування MVC(MVP). Подібний підхід спрощує не тільки розробку, а й пришвидшує та оптимізує роботу внутрішньої логіки додатку «Розклад» в цілому. Як результат, підсистема-додаток буде гнучкою та масштабуємою.

Щодо дизайну, виконуються вимоги надані компанією Google, а саме «Принципи дизайну для Android». Реалізовано акцентування інформації шляхом розмірів шрифту, стилів та виділенням елементів певними кольорами. Також реалізований «гумовий» дизайн для забезпечення підтримки різних дисплеїв.

# РОЗРОБКА ДОДАТКУ

## 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.

Розробка мобільного додатку на платформи AndroidOS 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](https://uk.wikipedia.org/wiki/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 ступення 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 AndroidOS інструментів. Таким чином локальная 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.

## Середовище та технологія розробки

З кожним днем технологій для розробки додатків стає все більше, а ті які вже були, розвиваються все сильніше. Таким чином, навіть для однієї Android платформи можна знайти багато технологій-аналогів для розробки програмного продукту. Як результат, реалізація додатку можлива на таких мовах як Java, C#, JavaScript та навіть на C++.

Не зважаючи на великій вибір «рідною» мовою Android все ж таки є Java, тому для створення даного додатку було обрано саме її. Таке рішення забезпечить максимально швидку роботу додатку. Для розробки додатку використовується Android SDK, наданий компанією Google для розробки різноманітного програмного забезпечення для відповідної платформи. Так як додаток розрахований на збереження певних даних, то це зумовлює створення локальної бази даних. Так як БД не буде включати в себе велику кількість таблиць та зв’язків, то від СУБД вирішили відмовитись для спрощення роботи та оптимізації швидкості роботи програми. Була обрана система для керування базами даних SQLite, тому що Android SDK вже містить в собі бібліотеки для роботи з нею. Середовищем розробки стала Android Studio v1.1. Всі вищеописані продукти є безкоштовними, тому не потребують сторонніх витрат на купівлю ліцензій.

Серед систем для хостингу проектів був обраний BitBucket, основаного на системі контролю версій Git та Mercurial. Адже він дозволяє безкоштовно розміщувати закриті репозиторії.

Далі детальніше про кожну з технологій та програму.

**Android SDK**

Android SDK([software development kit](https://en.wikipedia.org/wiki/Software_development_kit)) включає в себе повний набір інструментів для розробки програмного забезпечення. Вони включають в себе відладчик, бібліотеки, емулятор, документації, зразки коду і підручники. Платформи для розробки що підтримуються включають в себе комп'ютери, що працюють на Linux, Mac OS X 10.5.8 або пізнішої версії, і Windows XP або пізнішої версії.

З 2015 року офіційним інтегрованим середовищем розробки (IDE) є [Android Studio](https://en.wikipedia.org/wiki/Android_Studio), розроблена Google з використання технологій від IntelliJ. Однак, розробники можуть використовувати інші IDE. Крім того, розробники можуть використовувати будь-який текстовий редактор для редагування XML і Java-файлів, а потім використовувати командний рядок щоб створити, побудувати і налагоджувати програми Android.

Додатки Android упаковані в .apk форматі і зберігаються в папці «/data/app» на Android OS (папка доступна тільки для суперкористувача з міркувань безпеки). АПК пакет містить файли .dex (байт-код скомпільований файли для подальшої роботи з віртуальною машинною Dalvik), файли ресурсів і т.д.

**SQLite**

SQLite — полегшена [реляційна система керування базами даних](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85). Втілена у вигляді [бібліотеки](https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%B1%D0%BB%D1%96%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC), де реалізовано багато зі стандарту [SQL](https://uk.wikipedia.org/wiki/SQL)-92. [Сирцевий код](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%80%D1%86%D0%B5%D0%B2%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4) SQLite поширюється як [суспільне надбання](https://uk.wikipedia.org/wiki/%D0%A1%D1%83%D1%81%D0%BF%D1%96%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BD%D0%B0%D0%B4%D0%B1%D0%B0%D0%BD%D0%BD%D1%8F), тобто може використовуватися без обмежень та безоплатно з будь-якою метою. Фінансову підтримку розробників SQLite здійснює спеціально створений консорціум, до якого входять такі компанії, як [Adobe](https://uk.wikipedia.org/wiki/Adobe), [Oracle](https://uk.wikipedia.org/wiki/Oracle_Corporation), [Mozilla](https://uk.wikipedia.org/wiki/Mozilla), [Nokia](https://uk.wikipedia.org/wiki/Nokia), [Bentley](https://uk.wikipedia.org/w/index.php?title=Bentley_Systems&action=edit&redlink=1) і [Bloomberg](https://uk.wikipedia.org/wiki/Bloomberg).

Особливістю SQLite є те, що воно не використовує парадигму [клієнт-сервер](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80), тобто рушій SQLite не є окремим [процесом](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81_%28%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%29), з яким взаємодіє [застосунок](https://uk.wikipedia.org/wiki/%D0%97%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), а надає бібліотеку, з якою програма компілюється і рушій стає складовою частиною програми. Таким чином, як протокол обміну використовуються виклики функцій ([API](https://uk.wikipedia.org/wiki/API)) бібліотеки SQLite. Такий підхід зменшує накладні витрати, час відгуку і спрощує програму. SQLite зберігає всю базу даних (включаючи визначення, [таблиці](https://uk.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D1%8F_%28%D0%B1%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85%29), [індекси](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D0%B4%D0%B5%D0%BA%D1%81_%28%D0%B1%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85%29) і дані) в єдиному стандартному файлі на тому [комп'ютері](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80), на якому виконується застосунок. Простота реалізації досягається за рахунок того, що перед початком виконання транзакції весь файл, що зберігає базу даних, блокується; [ACID](https://uk.wikipedia.org/wiki/ACID)-функції досягаються зокрема за рахунок створення файлу-журналу.

Кілька [процесів](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81_%28%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%29) або [нитей](https://uk.wikipedia.org/wiki/%D0%9D%D0%B8%D1%82%D1%8C) можуть одночасно без жодних проблем читати дані з однієї бази. Запис в базу можна здійснити тільки в тому випадку, коли жодних інших запитів у цей час не обслуговується; інакше спроба запису закінчується невдачею, і в програму повертається код помилки. Іншим варіантом розвитку подій є автоматичне повторення спроб запису протягом заданого інтервалу часу.

У комплекті постачання йде також функціональна клієнтська частина у вигляді виконуваного файлу sqlite3, за допомогою якого демонструється реалізація функцій основної бібліотеки. Клієнтська частина працює з [командного рядка](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D0%B8%D0%B9_%D1%80%D1%8F%D0%B4%D0%BE%D0%BA), і дозволяє звертатися до файлу БД на основі типових функцій [ОС](https://uk.wikipedia.org/wiki/%D0%9E%D0%A1).

Завдяки архітектурі рушія можливо використовувати SQlite як на [вбудовуваних (embedded) системах](https://uk.wikipedia.org/wiki/%D0%92%D0%B1%D1%83%D0%B4%D0%BE%D0%B2%D0%B0%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), так і на виділених машинах з гігабайтними масивами даних.

Сама бібліотека **SQLite** написана мовою [C](https://uk.wikipedia.org/wiki/%D0%A1%D1%96_%28%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%29); існує велика кількість прив'язок до інших мов програмування, зокрема до [C++](https://uk.wikipedia.org/wiki/C%2B%2B), [Java](https://uk.wikipedia.org/wiki/Java), [Python](https://uk.wikipedia.org/wiki/Python), [Perl](https://uk.wikipedia.org/wiki/Perl), [PHP](https://uk.wikipedia.org/wiki/PHP), [Tcl](https://uk.wikipedia.org/wiki/Tcl) (засоби для роботи з Tcl включені в комплект постачання **SQLite**), [Ruby](https://uk.wikipedia.org/wiki/Ruby), [Haskell](https://uk.wikipedia.org/wiki/Haskell), [Scheme](https://uk.wikipedia.org/wiki/Scheme), [Smalltalk](https://uk.wikipedia.org/wiki/Smalltalk) і [Lua](https://uk.wikipedia.org/wiki/Lua), а також до багатьох інших. Повний список наявних засобів можна знайти на сторінці проєкту.

Особливості SQLite:

* [Транзакції](https://uk.wikipedia.org/wiki/%D0%A2%D1%80%D0%B0%D0%BD%D0%B7%D0%B0%D0%BA%D1%86%D1%96%D1%8F_%28%D0%B1%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85%29) [атомарні](https://uk.wikipedia.org/wiki/%D0%90%D1%82%D0%BE%D0%BC%D0%B0%D1%80%D0%BD%D0%B0_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D1%8F), послідовні, ізольовані, і міцні (ACID) навіть після збоїв системи і збоїв живлення;
* Встановлення без конфігурації — не потребує ані установки, ані адміністрування;
* Реалізує значну частину стандарту SQL92;
* База даних зберігається в одному крос-платформному файлі на диску;
* Підтримка терабайтних розмірів баз даних і гігабайтного розміру рядків і [BLOBів](https://uk.wikipedia.org/wiki/BLOB);
* Малий розмір коду: менше ніж 350KB повністю налаштований, і менш 200KB з опущеними додатковими функціями;
* Швидший за популярні рушії [клієнт-серверних](https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D1%96%D1%94%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) [баз даних](https://uk.wikipedia.org/wiki/%D0%A1%D0%9A%D0%91%D0%94) для найпоширеніших операцій;
* Простий, легкий у використанні [API](https://uk.wikipedia.org/wiki/API);
* Написана в [ANSI C](https://uk.wikipedia.org/wiki/ANSI_C), включена прив'язка до [TCL](https://uk.wikipedia.org/wiki/Tcl); доступні також прив'язки для десятків інших мов;
* Добре прокоментований [сирцевий код](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%80%D1%86%D0%B5%D0%B2%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4) зі 100% тестовий покриттям гілок;
* Доступний як єдиний файл сирцевого коду на ANSI C, який можна легко вставити в інший проект;
* Автономність: немає зовнішніх залежностей;
* Багатоплатформність: з коробки підтримується [Unix](https://uk.wikipedia.org/wiki/Unix) ([Linux](https://uk.wikipedia.org/wiki/Linux" \o "Linux) і [Mac OS X](https://uk.wikipedia.org/wiki/Mac_OS_X)), [OS/2](https://uk.wikipedia.org/wiki/OS/2), [Windows](https://uk.wikipedia.org/wiki/Windows) ([Win32](https://uk.wikipedia.org/wiki/Win32) і [WinCE](https://uk.wikipedia.org/wiki/WinCE)). Легко переноситься на інші системи;
* Поставляється з автономним клієнтом [інтерфейсу командного рядка](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D0%BE%D0%B3%D0%BE_%D1%80%D1%8F%D0%B4%D0%BA%D0%B0), який може бути використаний для управління базами даних SQLite.

**Android Studio 1.1**

Android Studio — [інтегроване середовище розробки](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BE%D0%B2%D0%B8%D1%89%D0%B5_%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8) для платформи [Android](https://uk.wikipedia.org/wiki/Android). Створена компанією Google за підтримки JetBrains на основі іншого інтегрованого середовища IntelliJ Idea. З 2015 року стала офіційним IDE для розробки під платформу Android.

Особливості Android Studio:

* Гнучка система збірки проекту на основі технології Gradle;
* Декілька варіантів збірки та генерації apk-файлів;
* Шаблони коду для допомоги в створенні загальних додатків;
* Багатий на функціонал редактор слоїв з підтримкою можливості drag-and-drop;
* Інструменти для моніторингу продуктивності, зручності, сумісності версій, тощо;
* Можливості застосування ProGuard і підписування додатку;
* Вбудований помічник для [Google Cloud Platform](http://developers.google.com/cloud/devtools/android_studio_templates/), який полегшує інтеграцію Google Cloud Messaging та App Engine.

Середовище розробки адаптоване для виконання типових завдань, що вирішуються в процесі розробки застосунків для платформи Android. У тому числі у середовище включені засоби для спрощення тестування програм на сумісність з різними версіями платформи та інструменти для проектування застосунків, що працюють на пристроях з екранами різної роздільності (планшети, смартфони, [ноутбуки](https://uk.wikipedia.org/wiki/%D0%9D%D0%BE%D1%83%D1%82%D0%B1%D1%83%D0%BA), годинники, окуляри тощо). Крім можливостей, присутніх в [IntelliJ IDEA](https://uk.wikipedia.org/wiki/IntelliJ_IDEA), в Android Studio реалізовано кілька додаткових функцій, таких як нова уніфікована підсистема складання, тестування і розгортання застосунків, заснована на складальному інструментарії Gradle і підтримуюча використання засобів [безперервної інтеграції](https://uk.wikipedia.org/wiki/%D0%91%D0%B5%D0%B7%D0%BF%D0%B5%D1%80%D0%B5%D1%80%D0%B2%D0%BD%D0%B0_%D1%96%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B0%D1%86%D1%96%D1%8F).

Для прискорення розробки застосунків представлена ​​колекція типових елементів інтерфейсу і візуальний редактор для їхнього компонування, що надає зручний попередній перегляд різних станів інтерфейсу застосунку (наприклад, можна подивитися як інтерфейс буде виглядати для різних версій Android і для різних розмірів екрану). Для створення нестандартних інтерфейсів присутній майстер створення власних елементів оформлення, що підтримує використання шаблонів. У середовище вбудовані функції завантаження типових прикладів коду з [GitHub](https://uk.wikipedia.org/wiki/GitHub).

До складу також включені пристосовані під особливості платформи Android розширені інструменти [рефакторингу](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D0%BD%D0%B3), перевірки сумісності з минулими випусками, виявлення проблем з продуктивністю, моніторингу споживання пам'яті та оцінки зручності використання. У редактор доданий режим швидкого внесення правок. Система підсвічування, статичного аналізу та виявлення помилок розширена підтримкою Android API. Інтегрована підтримка оптимізатора коду ProGuard. Вбудовані засоби генерації цифрових підписів. Надано інтерфейс для управління перекладами на інші мови.

**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).

## База даних додатку

Фізична модель даних

ER діаграма

Опис сутностей

Сутність «Користувач», являє собою всю інформацію про користувача системи. А саме, email адресу, хеш паролю, номер телефону, ПІБ. А також у цій сутності є зв’язок з сутність ДЛС. Ця сутність, є доповненою від стандартної AspNetUser, а базові поля цієї сутності необхідні для коректної роботи системи авторизації ASP.NET Identity. Цю сутність було розширено, додаванням необхідних властивостей , таких як: ім’я, прізвище, по-батькові, номер телефону.

Сутність «ДЛС», являє собою всю інформацію яка описує ДЛС. По суті, це весь набір характеристик, які були описані в 2 розділі даного даного. А також, має в собі поле, дати останньої зміни, яке використовується для синхронізації.

## Опис процесів

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.

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

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.

## Особливості побудови локалізованого додатку

Одною з основних вимог, що були висунуті у другому розділі даного дипломного проекту, це створення багато мовного інтерфейсу з можливістю, легкого додавання мови.

## ВИСНОВОК ПО РОЗДІЛУ

# ПРАКТИЧНЕ ЗАСТОСУВАННЯ ДОДАТКУ

## ВИСНОВОК ПО РОЗДІЛУ

# ОХОРОНА ПРАЦІ

Система рейтингу дистанційних лабораторних стендів передбачає одне робоче місце інженера-програміста. Для роботи з модулем необхідне робоче місце обладнане ЕОМ. В даному розділі аналізується робоче місце інженера-програміста, розглядаються питання безпеки робочого місця, освітлення, пожежної безпеки, мікроклімату, шуму та електробезпеки.

## Характеристика робочого місця

Площа віртуального приміщення, в якому працює програміст, складає загальну площу 33 м2 , об’єм V = 99 м3 . Ширина приміщення D = 5,5 м, довжина L = 6 м, висота стелі H = 3м. У приміщенні знаходиться 4 робочих місць. Схематичний план приміщення наведено на рис. 4.1.

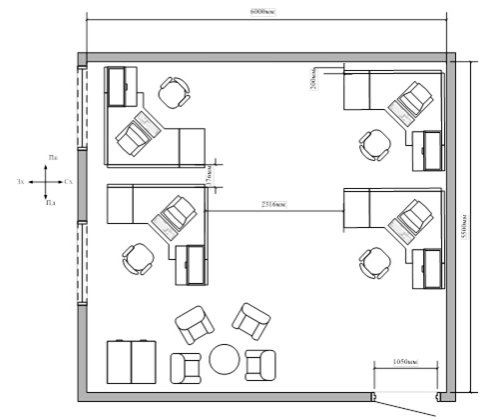


Рисунок.5.1. Схематичний план приміщення

Кожне робоче місце обладнане робочим столом кутової форми (1400х1400х700), робочим кріслом, персональним комп’ютером, що складається з системного блоку, монітору, клавіатури, миші.

Середня площа приміщення на одного працюючого складає: S`=S/N=8,5 м2 , середній об’єм на одного працюючого: V`=V/N=24,75 м3 , що відповідає нормативам НПАОП 0.00-1.28-10.

Кожне робоче місце має наступні характеристики: площа поверхні столу – 1,25 м2 ; висота, кут нахилу спинки та висота самого крісла регулюються; робочий стіл має простір для ніг заввишки 0,75 м, шириною 0,9 м; глибиною на рівні колін 0,45 м, на рівні витягнутої ноги 0,8 м; наявна підставка для ніг; відстань від екрану до ока працівника складає 0,7 м.

## Аналіз небезпечних виробничих факторів

### Характеристика виробничого шуму

Як вказано вище, в кабінеті знаходиться чотири робочих місць з ПК, кожне з яких устатковане монітором, вінчестером в системному блоці, вентиляторами системи охолоджування ПК та клавіатурою.. При роботі принтерів виникають шуми механічного і аеродинамічного походження, широкосмугові із аперіодичним підсилення, тому принтери винесено у окреме коридорне приміщення. Таким чином, єдиним джерелом шуму в досліджуваному приміщенні є жорсткий диск і складає 40 ДБА. [15]

За наявності декількох джерел шуму з однаковим рівнем інтенсивності Li загальний рівень шуму визначають за формулою: L = Li+10lgn. У нашому випадку таких джерел чотири, отже, загальний рівень шуму буде визначатися так: L=40 + 10lg 4 = 46 < 50 дБА, що відповідає ДСН 3.3.6.042-99.

### Характеристика випромінювання

На робочому місці мають бути встановлені сучасні монітори, характеристики яких відповідають нормам НПОПО 0.00-1.128-10: припустима інтенсивність потоку енергії 10Вт/м2 , напруженість електромагнітного поля по електричній складовій на відстані 50см від екрана – 10 В/м. [13]

## Мікроклімат робочої зони програміста

Робота інженера-програміста виконується сидячи і не потребує фізичного напруження, не пов’язана із ходінням, не супроводжуються фізичним напруженням, тому відноситься до категорії легких робіт Іб і мають дотримуватися наступні санітарні норми мікроклімату виробничих приміщень відповідно ДСН 3.3.6.042-99 ( див. табл.4.1.).

Таблиця 5.1. Значення оптимальних, допустимих і фактичних параметрів

|  |  |  |  |
| --- | --- | --- | --- |
| Період | Температура повітря, tо C | Вологість повітря, φ, % | Швидкість руху повітря, V, м/с |
| Оптим. | Оптим. | Оптим. |
| Холодний | 22-24 | 60-40 | 0.1 |
| Теплий | 23-25 | 60-40 | 0.1 |

Для встановлення оптимального температурного режиму у холодний період року використовувати центральне водяне опалення [14].

У теплий період року - використовувати кондиціонер Panasonic CSE12NKD (CU-E12NKD) для площі 35 м2 з потужністю охолодження 3,5 кВт.

## Освітлення робочого місця

В приміщенні 2 вікна (1,4\*2,0 м) площею 2,8 м2 , які орієнтовані на західний бік.

Необхідний рівень освітлення на робочому місті визначають залежності від характеристик зорових робіт. Роботи за ПЕОМ і оформлення документації, відносяться до ІІІ розряду зорової роботи (високої точності) для цих робіт згідно ДБН В.2.5-28-2006 нормоване значення освітленості становить 400 лк.

Природного освітлення не достатньо для забезпечення нормованого значення освітленості 300-500 лк, тому необхідно використовувати штучне освітлення.

Розрахуємо штучне освітлення в приміщені. Його розрахунок проводиться за методом світлового потоку, з урахуванням потоків відображених від стелі і стін.

Згідно ДБН В.2.5-28-2006 «Природне і штучне освітлення» для даних робіт встановлена необхідна освітленість робочого місця 𝐸𝐸н = 400 лк.

Загальний світловий потік визначається за формулою:

де, 𝐸н− нормована освітленість;

𝑆 – площа застосування;

𝑘1=1,5 – коефіцієнт, що враховує старіння ламп і забруднення світильників;

𝑘2= 1,1 – коефіцієнт, що враховує нерівномірність освітлення приміщення;

𝑉 – коефіцієнт використання світлового потоку, який визначається відповідно до коефіцієнтів віддзеркалення стін, робочих поверхонь, стелі, геометрії приміщення і типів світильників V=0,7.

Розміри приміщення складають A=5.5 м, B=6 м, Н=3 м.

𝑆 = 𝐴 ∙ 𝐵 = 6 ∙ 5.5 = 33 м2

Оберемо з таблиці коефіцієнти використання світлового потоку:

− коефіцієнт віддзеркалення від побіленої стелі 𝑅стелі = 70% ;

− коефіцієнт віддзеркалення від стін білого кольору 𝑅стін = 50%;

− коефіцієнт віддзеркалення від темного лінолеуму 𝑅підлоги = 10%;

− індекс приміщення:

, де =0.75 м – висота робочої поверхні над підлогою.

Визначимо показник приміщення:

Тепер визначимо значення загального світлового потоку:

Для забезпечення загального штучного освітлення обираємо світильник з двома люмінесцентними лампами типу ЛД-80 .

Світловий потік однієї лампи ЛД-80 складає 𝐸л = 4250

Тепер визначимо кількість ламп, необхідну для освітлення приміщення:

Таким чином, щоб забезпечити світловий потік 𝐸заг = 26022 лм треба використовувати 8 ламп, тобто 4 світильники. Розміщуємо світильники у 2 ряди.

Електрична потужність однієї лампи ЛД-80 = 80Вт.

Потужність усієї освітлювальної системи:

𝑊заг = 80 ∗ 8 = 640 Вт.

За ДБН В.2.5.-28-2006 світловий потік має складати 300 – 500 лк, освітлення в приміщенні повністю відповідає вимогам нормативних документів.

План розміщення світильників зображено на рис. 5.2.

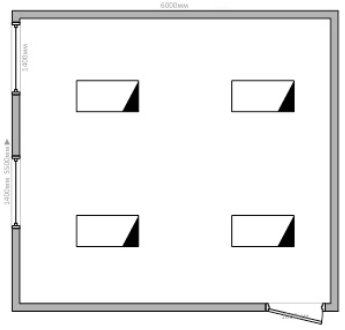


Рисунок.5.2. План освітлення приміщення

## Електробезпека

Приміщення кабінету згідно ПУЕ за небезпекою ураження електричним струмом можна віднести до І класу, тобто це приміщення без підвищеної небезпеки (сухе, без пилу, з нормальною температурою повітря, ізольованими підлогами і малим числом заземлених приладів).

На робочому місці програміста з всього устаткування металевим є лише корпус системного блоку комп'ютера, але тут використовуються системні блоки, що відповідають стандартові фірми IBM, у яких крім робочої ізоляції передбачений елемент для заземлення і провід з жилою, що заземлює, для приєднання до джерела живлення.

Живлення до приміщення повинно відводитися від загального розподільного щитка. У приміщенні має використовуватися одна фаза трифазної мережі напругою 220В та частотою 50 Гц.

Використання нульового робочого провідника як нульового захисного провідника забороняється, а також не припустиме підключення цих провідників на щитку до одного контактного затискача

Для сприятливих умов роботи всі елементи електроприладів й устаткування мають бути виконані відповідно до умов техніки електробезпеки, мати необхідне ізоляційне покриття (подвійна ізоляція) і властивості, що виключає можливість поразки електричним струмом при підключенні й експлуатації встаткування. [16]

## Пожежна безпека

Дане приміщення згідно НАПБ Б.03.002-2007 відноситься до категорії В, тому що містить пил і волокна, меблі, комп’ютерну техніку, паперові документи та обладнано паркетом. Робочі зони приміщення згідно з законодавчою базою відносяться до класу П-ІІа.

Для автоматичного сповіщення про пожежу кімната повинна бути оснащена швидко реагуючими димовими оповісниками DOR-4046 (PolonAlfa). Після перевищення відповідного порогу автокорекції, посилає до працюючого спільно приймально-контрольного приладу сигнал сервісної тривоги, що знаходиться на КПП I поверху. Сповіщувач оснащений внутрішніми ізоляторами коротких замикань.

Для запобігання пожежі в кімнаті є чотири ручних вуглекислотних- брометілових вогнегасника ОУБ-3А. Згідно ППБУ-95 необхідно встановити 4 вогнегасника. Дане приміщення має площу 33 м2 , кількість вогнегасників відповідає нормі

Необхідним елементом є план евакуації. Він розташований в кімнаті біля дверей.

## Інструкція з техніки безпеки

До роботи на персональній електронно-обчислювальній машині (ПЕОМ) допускаються після вивчення даної інструкції особи, які пройшли попередній медичний огляд, встановлений курс навчання за даною професією, пройшли вступний та первинний (на робочому місці) інструктажі з питань охорони праці, пожежної безпеки, інструктаж і перевірку знань з електробезпеки і отримали ІІ кваліфікаційну групу.

Перед початком роботи на ПК користувач повинен:

а) пересвідчитися у цілості корпусів і блоків (обладнання) ПК;

б) перевірити наявність заземлення, справність і цілість кабелів живлення, місця їх підключення.

Не рекомендується вмикати ПК та починати роботу при виявлених несправностях. Під час роботи пересвідчившись у справності обладнання, не допускати у зону сторонніх осіб, увімкнути електроживлення ПК, розпочати роботу, дотримуючись умов інструкції з її експлуатації.

Не рекомендується:

а) замінювати змінні елементи або вузли та проводити перемонтаж при ввімкненому ПК;

б) з'єднувати і роз'єднувати вилки та розетки первинних мереж електроживлення, які знаходяться під напругою;

в) знімати кришки, які закривають доступ до струмопровідних частин мережі первинного електроживлення при ввімкненому обладнанні;

г) користуватися паяльником з незаземленим корпусом;

д) замінювати запобіжники під напругою;

е) залишати ПК у ввімкненому стані без нагляду. [17]

По закінченні робочого дня:

а) кнопкою "ВИМК" відключити електроживлення ПК згідно з інструкцією експлуатації, вийнявши вилку кабелю живлення з розетки;

б) впорядкувати робоче місце користувача ПК, прибравши використане обладнання та матеріали у відведені місця;

в) про виявлені недоліки у роботі ПК протягом робочого часу необхідно повідомити відповідним посадовим особам та спеціалістам.

Залишаючи приміщення після закінчення робочого дня, дотримуючись встановленого режиму огляду приміщення, необхідно:

а) зачинити вікна, кватирки;

б) перевірити приміщення на відсутність тліючих предметів;

в) відключити від електромережі всі електроприлади, електрообладнання та вимкнути освітлення;

г) зачинити вхідні двері приміщення на замок і ключ здати черговому охорони. [18]

# ВИСНОВК ПО РОЗДІЛУ

У ході розробки розділу вивчено приміщення, в якому розроблювалося програмне забезпечення. Описано та проаналізовано небезпечні й шкідливі виробничі фактори у виробничому приміщенні з радіоелектронною апаратурою.. У результаті проведення робіт встановлено, що об’єм і площа приміщення, які відводяться на одного працівника, відповідають нормативним значенням. Для цих факторів визначено нормативні показники у відповідності з діючою нормативно- технічною документацією та проведено порівняння з фактичними їх значеннями. З виявлених небезпечних та шкідливих факторів виділено найбільш несприятливі для яких було розроблено детальні заходи з охорони праці. Аналіз усіх розрахованих у даному розділі факторів показав результати, які дають всі підстави вважати, що розглянуте виробниче приміщення повністю відповідає всім нормативним документам і вимогам.

# ВИСНОВК

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 використання 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 Android. 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.