1. **Завдання та мета на розробку**

Завданням курсового проекту є реалізація інформаційної системи аптеки «Будьте здорові».

Метою роботи на курсовому проекті є поглиблення знань, отриманих при вивченні дисциплін «Об’єктно орієнтоване програмування», «База даних» та «Інструментальні засоби візуального програмування», удосконалення практичних навичок при створенні інформаційних систем.

Для реалізації програмного продукту обрано середовище швидкої розробки додатків Embarcadero RAD Studio 10.2, а саме в С++ Builder.

1. **Опис предметної області**

Аптека є кінцевим пунктом збуту ліків. Кожна аптека має свій асортимент товару для роздрібної торгівлі. Реалізовує ліки різного виду. До основних видів ліків належать:

* обезболюючі;
* антибіотики;
* судино-звужуючі;
* снодійні;
* антисептики;
* засоби особової гігієни;
* та інші ліки.

Рух ліків в аптеці відбувається наступним чином:

* Постачання ліків в аптеку – ліки для зберігання надходять на [склад](http://ua-referat.com/%D0%A1%D0%BA%D0%BB%D0%B0%D0%B4), для продажу на полицях у торговому залі.
* Реалізація товару – продаж покупцю, результатом якого є оформлення чеку.

Ліки зберігаються до їх продажу.

Облік ліків в аптеці передбачає наступні етапи:

* додавання інформації про [товар](http://ua-referat.com/%D0%A2%D0%BE%D0%B2%D0%B0%D1%80);
* видаток товару (продаж/списання);
* ведення обліку залишків товару.

При внесенні даних про прибуття товар фіксуються такі дані: дата приходу товару в аптеку: день, місяць, рік, назва товару, ціна, кількість.

Для ведення обліку в базі даних аптеки зберігається інформація про ліки, а саме: код товару, назва товару, виробник, ціна за одиницю, залишок, дата приходу на склад, кількість товару.

Завданням курсового проекту є розробка інформаційної системи для аптеки «Будьте здорові».

Програмний продукт повинен зберігати інформацію про ліки, які є в аптеці, з можливістю її модифікувати (добавити, вилучити, редагувати), та інформацію про продаж ліків.

Основні операції з ліками :

* додавання інформації про ліки (назва, кількість, яка є в наявності, категорія ліків, виробник, призначення, ціна);
* пошук ліків за назвою, за призначенням;
* сортування інформації за любим з критеріїв;
* постачання ліків в аптеку.

При розробці потрібно врахувати/передбачити можливість двох режимів доступу до даних:

− адміністратор – має всі права при роботі з даними, передивлятись всі таблиці, додавати;

− клієнт – може ознайомитись з інформацію про ліки, шукати ліки, які йому потрібні, сортувати ліки за ціною або алфавітом, назвою що є в базі, а також купувати.

Для представлення інформації про ліки, слід врахувати такі дані: їх назву, виробника, призначення, ціну, дату випуску та призначення. При фіксуванні препарату, який знаходиться на складі, необхідно також враховувати дату постачання та кількість.

Вся інформація про ліки повинна зберігатися в реляційній базі даних.

Представлення та обробку даних реалізувати за допомогою технологій об’єктно-орієнтованого програмування, а саме класи та методи.

1. **Порівняння з аналогами**

За приклад взято онлайн сайти аптек, адже автоматизовані інформаційні системи аптек, які розроблені як окреме програмне забезпечення немає у вільному доступі.

Для аналогів розглянемо сайти аптек «Бажаємо здоров’я» та «Доброго дня».

Розглянемо невелику аптеку на рисунку 1.

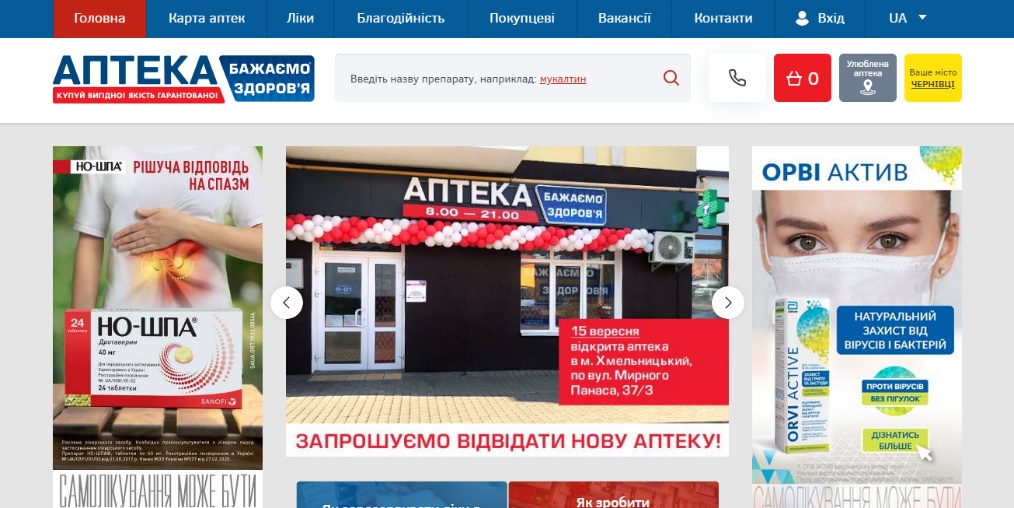
****

Рисунок 1 – головна сторінка сайту аптеки «Бажаємо здоров’я»

На головній сторінці сайті зображені рекламні рисунки такі як – інформація про відкриття нової аптеки та різних препаратів, є пошук за назвою ліків, меню з різними пунктами: Головна, карта аптеки, ліки, благодійність, покупцеві, вакансії, контакти, є особистий кабінет клієнта.

Розглянемо одну з можливостей функціоналу сайту, нехай пункт «Ліки» – це зображено на рисунку 2.

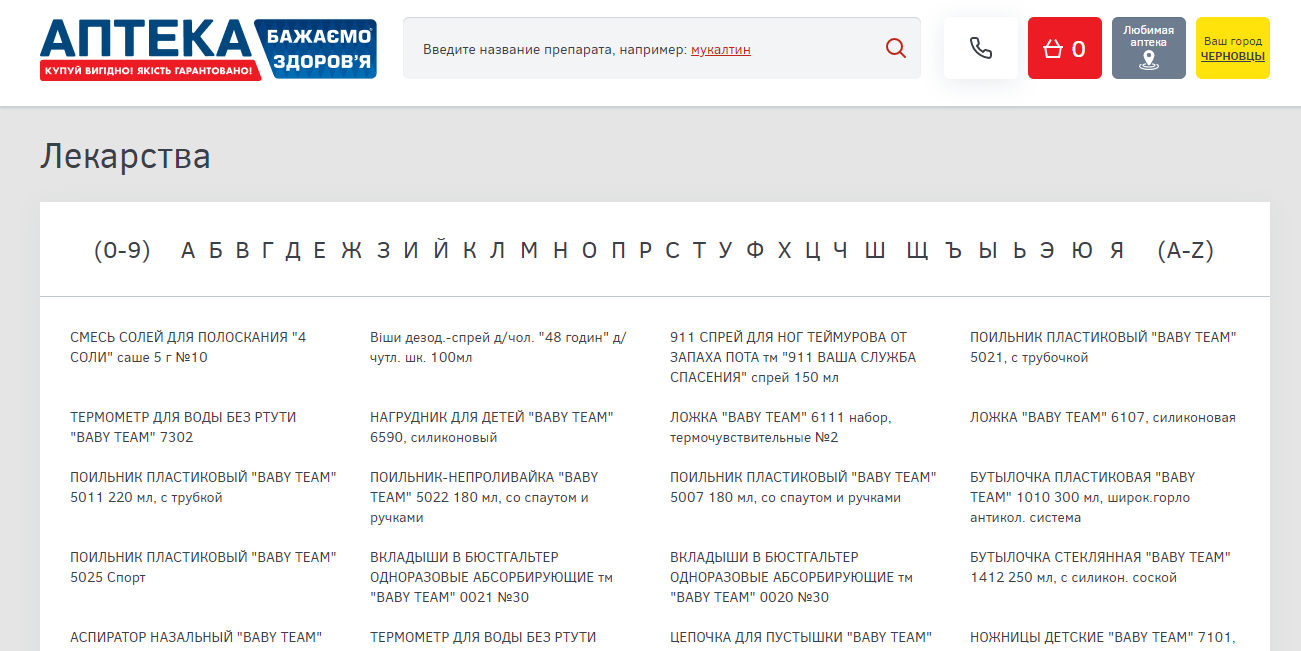


Рисунок 2 – пункт меню «Ліки».

На сторінці з ліками перед користувачем відкривається меню в якому можливий пошук медикаментів за алфавітом, на жаль, даний сайт не передбачає сортування за ціною або пошуку ліків за назвою виробника. На цій сторінці у покупця досі є можливість шукати препарат за його назвою.

Другий аналог – Аптека «Доброго дня» зображена на рисунку 3.

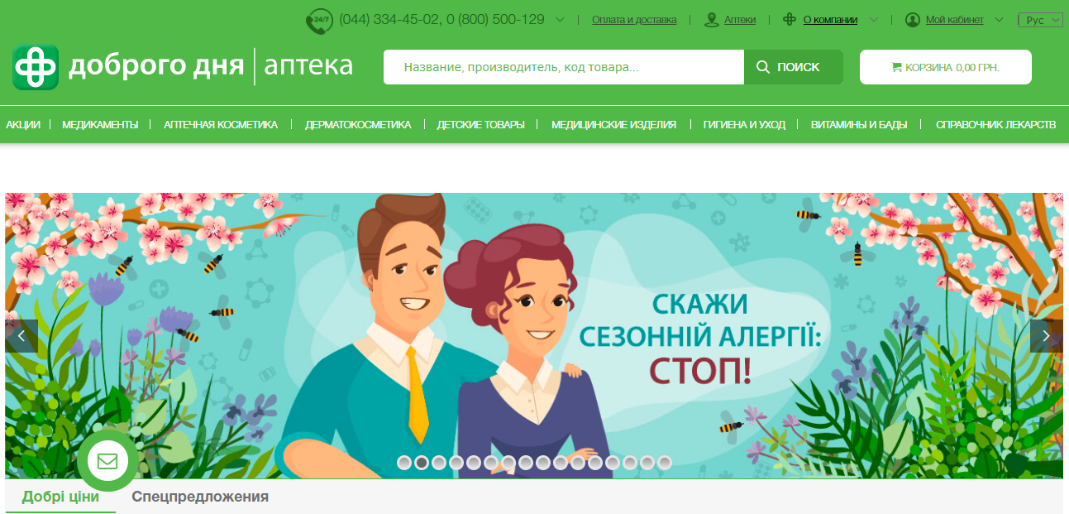


Рисунок 3 – головна сторінка сайту аптеки «Доброго дня».

Сайт має зрозумілий для користувача інтерфейс. Меню даного сайту дає можливість переглянути медикаменти за їх типами. На головній сторінці інформація щодо оплати. На сайті є інформація про оффлайн аптеки по всій Україні та контактна інформація. Є можливість створити акаунт користувача, а також реалізована корзина яка дає можливість користувачу купувати медикаменти, є пошук ліків за назвою.

Розглянемо функціонал сайту. Пункт меню «Медикаменти» – це зображено на рисунку 4.



Рисунок 4 - пункт меню «Медикаменти».

При відкритті медикаментів перед користувачем відкривається сторінка на якій зображено популярні ліки, що містять інструкцію, вказана ціна, протипоказання та ін. Можна шукати ліки за назвою, за брендом, виставити цінові рамки для препаратів.

Отже, виконавши дослідження можна дійти висновку, що необхідно спроектувати та реалізувати інтуїтивно-зрозумілий інтерфейс користувача.

|  |  |  |  |
| --- | --- | --- | --- |
| Можливості аптеки | Бажаємо здоров’я | Доброго дня | Власна розробка |
| Інтуїтивно зрозумілий інтерфейс | + | + | + |
| Пошук ліків за критеріями | | | |
| За назвою | + | + | + |
| За виробником | - | - | + |
| За типом | - | + | + |
| Сортування за критеріями | | | |
| Сортування за назвою | + | - | + |
| Сортування за виробником | - | - | + |
| Сортування за ціною | - | + | + |
| Особистий кабінет | | | |
| Клієнта | + | + | - |
| Адміністратора | + | + | + |

Вважаю, що розробка доцільна, адже вона буде реалізована не як сайт, а як окремий програмний продукт, аналогів якому немає.

1. **Життєвий цикл програмного продукту**

Життєвий цикл (ЖЦ) програмнго забезпечення – це сукупність стадій та етапів, які проходить інформаційна система в своєму розвитку від моменту прийняття рішення про початок удосконалення системи управління до моменту, коли інформаційна система припиняє своє існування (перестає функціонувати).

Модель життєвого циклу – структура, що складається із процесів, робіт та задач, які включають в себе розробку, експлуатацію і супровід програмного продукту; охоплює життя системи від визначення вимог до неї до припинення її використання.

Каскадна (або послідовна) модель. Передбачає строго послідовне в часі і однократне виконання всіх фаз проекту з детальним попереднім плануванням і визначеними вимогами.

Основною особливістю цієї моделі є розбиття всієї розробки на етапи. Перехід від одного етапу до іншого відбувається лише при умові повного завершення робіт на попередньому етапі. Кожен етап завершується випуском документації, достатньої для того, щоб обробка могла бути продовжена іншою командою розробників, зображено на рисунку 5.

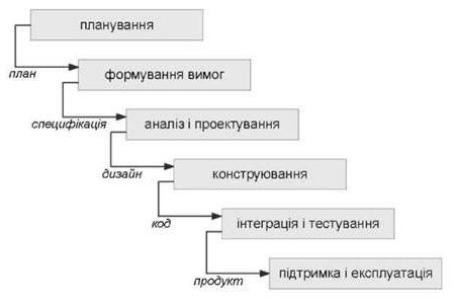


Рисунок 5 – Каскадна модель життєвого циклу.

Проте в процесі використання цього підходу виявився ряд його недоліків, спричинених насамперед тим, що реальний процес створення систем ніколи повністю не вкладався в таку жорстку схему. В процесі створення постійно виникала потреба в поверненні до попередніх етапів і уточнення або перегляд раніше прийнятих рішень. У результаті реальний процес створення ПЗ приймав вигляд, який зображено на рисунку 6.

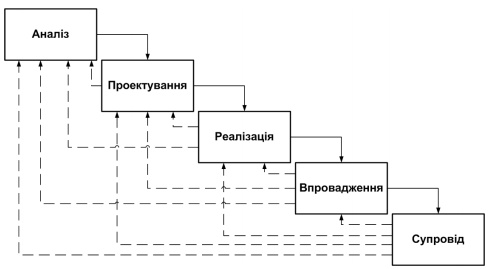


Рисунок 6 – Реальний процес розробки ПЗ за каскадною схемою

Каскадна модель добре себе зарекомендувала при розробці систем, для яких можна повністю сформулювати всі необхідні вимоги і критерії. Серед недоліків цієї моделі можна назвати: істотну затримку в отриманні кінцевих результатів, виявлення помилок, як правило на останньому етапі розробки, високий ступінь ризику. Послідовна модель характеризується жорсткою структурою, що ускладнює її застосування на практиці. Проте, більшість ІТ-розробників використовують саме каскадну модель.

Спіральна модель. На відміну від каскадної, передбачає ітераційний процес розробки інформаційної системи. Кожна ітерація є завершеним циклом розробки кінцевого продукту. На кожному витку (ітерації) спіралі створюється фрагмент або версія програмного продукту, уточнюється кінцева ціль і характеристики проекту, визначається його якість, плануються роботи для наступного витка. Особливу увагу приділяється аналізу ризиків, що впливають на організацію ЖЦ - аспектам взаємодії спеціалістів в проектній команді.

Використання спіральної моделі дозволяє здійснювати перехід на наступний етап виконання проекту, не дочекавшись повного завершення робіт на 10 попередньому – недороблену роботу можна буде завершити на наступній ітерації. Головна мета кожної ітерації – як найшвидше створити працездатний продукт, який можна показати замовнику. Спіральна модель зображена на рисунку 7.

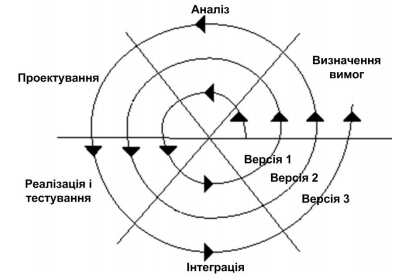


Рисунок 7 – Спіральна модель життєвого циклу

Серед переваг спіральної моделі слід відмітити спрощення внесення змін в проект при зміні вимог замовника, елементи системи інтегруються один в одного поступово. Основна проблема спіральної моделі - визначення моменту переходу на наступний етап.

Було обрану каскаду модель для програмного продукту «АІС аптеки Будьте здорові».

Принципова особливість каскадної моделі – це перехід на наступну стадію можливий для здійснення тільки після повного завершення роботи на поточній стадії, повернення на пройдені стадії не передбачається. Кожна стадія закінчується одержанням результатів, що є вихідними даними для наступної стадії, та випуском повного комплекту документації. На мою думку ця модель є найкраще продуманою та логічною, саме тому було обрано її.

Визначення вимог

Аналіз предметної області

Моделювання предметної області(принципи взаємодії між об’єктами предметної області)

ФОРМУВАННЯ ТЗ

Моделювання інтерфейсу системи

Розробка алгоритмів підзадач

Реалізація системи

Тестування системи на відповідність вимогам

Тестування системи на відповідність проекту

(верифікація)

Введення в експлуатацію

Супроводження

Каскадна модель передбачає строго послідовне в часі і однократне виконання всіх фаз проекту з детальним попереднім плануванням і визначеними вимогами, саме тому було обрано цю модель життєвого циклу.

Отже, після графічної реалізації можна приступати до практичної реалізації маючи точний план дій.

1. **Специфікація програмного продукту**

Призначення та мета розробки: програмне забезпечення, що розробляється, призначено для автоматизації процесів роботи з інформацією при продажу ліків, а саме: забезпечення виконання ряду функцій по веденню реєстрації продаж. Автоматизована система дозволить підвищити якість та прискорити обслуговування клієнтів, за рахунок швидкої обробки інформації, а також допоможе оперативно приймати управлінські рішення.

Програмний продукт дозволить підвищити якість розрахунків та швидкому формуванню чеку, прайс-листів наявного товару та іншої звітності.

Застосування програмного забезпечення можливе в окремих аптеках.

Вхідні дані:

* Ліки (код, назва, рік випуску, термін придатності, виробник, призначення, ціна, кількість, дата постачання).

Вихідні дані:

* Формування загальної вартості чека.

ПЗ що розробляється повинно вирішувати наступні задачі: принципові, функціональні, сервісні.

До принципових задач відносять:

* забезпечення збереження та ефективний пошук даних пов’язаних з інформацією про ліки;
* зручний та інтуїтивний інтерфейс;
* збільшення швидкості обробки даних пов’язаних з формуванням звітності;
* мінімізацію можливості виникнення помилок у роботі з даними, які може зробити оператор;
* контроль даних, що додаються у базу даних.

До функціональних задач відносять наступні:

* збереження даних в довідниках;
* пошук даних за різними ключами;
* можливість розрахунків та збереження їх результатів;

До сервісних задач відносяться:

* організація головного меню;
* інформування користувача про причину відмови у виконанні тієї чи іншої операції;
* забезпечення маніпуляції з даними.

Вимоги надійності.

* програмний продукт повинен надійно працювати;
* Забезпечити різні рівні доступу, а саме користувач і адміністатор.
* розробник гарантує роботу програмного продукту без збоїв.

Програма повинна підтримувати спільний режим роботи з операційними системами Windows 7 та вищі.

1. **Структурна модель предметної області**

## Моделювання даних

Вся інформація про ліки зберігається в таблицях бази даних.

Таблиця Preparat призначення для зберігання інформації про ліки, її структура представлено на рисунку 6:

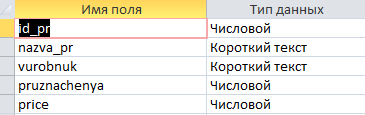


Рисунок 6 – Структура таблиці Preparat

Таблиця містить інформацію про ліки та має наступні поля:

* id\_pr – числове поле, ключ, включає в себе id препарату;
* nazva\_pr – короткий текст, призначено для збереження назви ліків;
* vurobnuk – короткий текст, інформацію про фірму виробника;
* pruznachenya – числове поле, інформація про тип категорії ліків;
* price – числове поле, ціна ліків.

Всі поля таблиці є обов’язковими для введення.

Таблиця Sklad призначення для зберігання інформації про наявні ліки в аптеці. її структура представлена на рисунку 7.

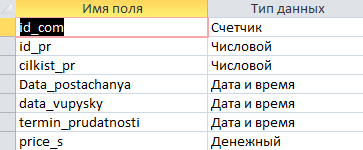


Рисунок 7 – Структура таблиці Sklad

Таблиця містить такі поля:

* id\_com – лічильник, id кожної комірки на складі;
* id\_pr – числове поле, id препарату, який знаходиться в комірці;
* cilkist\_pr – числове поле, кількість препарату в комірці;
* Data\_postachanya – дата/час, дата коли цю комірку заповнили препаратом;
* Data\_vupysky – дата/час, дата випуску ліків;
* termin\_prudatnosti – дата/час, дата до якої ліки можна приймати;

Таблиця Chek призначення для зберігання інформації про чек, її структура представлена на рисунку 8:



Рисунок 8 – Структура таблиці Chek

Таблиця Chek містить такі поля:

* id\_cheka – числове поле, id чека;
* Data\_CHEKA – дата/час, інформація про дату отримання чека.

Таблиця Operaciya призначення для зберігання інформаії про те, в якому чеку скільки препаратів було придбано, її структура представлена на рисунку 9.

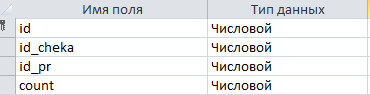


Рисунок 9 – Структура таблиці Operaciya.

* id – ключове поле;
* id\_Cheka – id чека;
* id\_pr – id препарата, який було придбано;
* count – кількість препарату, який було придбано.

На рисунку 10 зображено логічну модель даних:

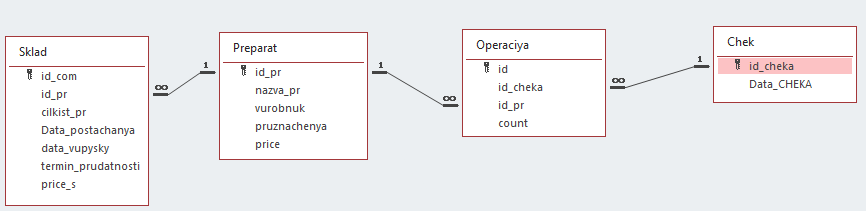


Рисунок 10 – Логічна модель даних

На схемі даних зображено таблиці та зв’язок між ними. Поле id\_pr в таблиці Sklad зв’язано багато до одного з ключовим полем «id\_pr» таблиці Preparat для того, аби в таблиці Sklad мати інформацію про ліки, скільки якого препарату лишилось на складі. Також ключове поле таблиці Preparat зв’язано 1 до багатьох з полем id\_pr в таблиці Operaciya, аби в таблиці Operaciya ми бачили який препарат в якому чеку замовили. Ключове поле таблиці Chek «id\_cheka» зв’язано 1 до багатьох з полем id\_cheka, для того, аби в таблиці Operaciya ми знали, в якому саме чеку було замовлено ліки.

1. **Діаграма використання**

**Функціональне моделювання**

Діаграма прецедентів – діаграма, на якій зображено відношення між користувачами та прецедентами в системі.

Актор – клієнт, який може переглядати дані, шукати, корегувати по кваліфікації, а також замовляти ліки, а адміністратор повинен спочатку авторизуватись, якщо він не пройде авторизацію, то він не зможе працювати з даними, до його можливостей роботи з даними входить перегляд та введення.

На рисунках 11 та 12 зображено діаграми прецедентів для користувача та адміністратора.

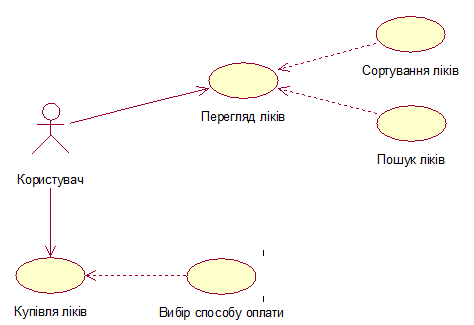


Рисунок 11 – Діаграма прецедентів користувача.

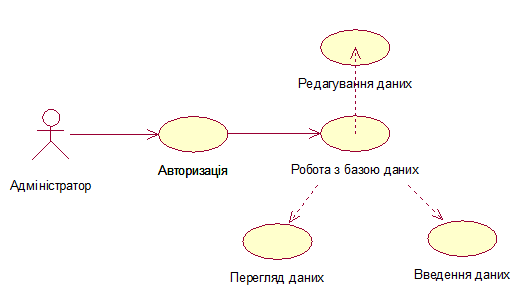


Рисунок 12 – Діаграма прецедентів адміністратора

Діаграма використання зображена на рисунку 13:

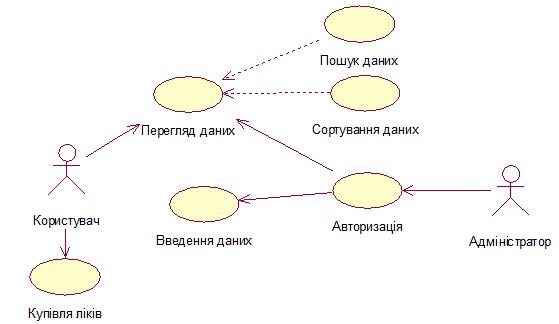


Рисунок 13 – Діаграма використання

1. **Діаграма взаємодій та станів**

**Діаграма взаємодії** — одна з моделей опису поведінки взаємодіючих груп об'єктів в UML. Як правило, кожна окрема діаграма взаємодії описує поведінку тільки в межах одного варіанта використання. На такій діаграмі прийнято відображати екземпляри об'єктів та повідомлення, якими ці об'єкти обмінюються один з одним в рамках даного варіанта використання.

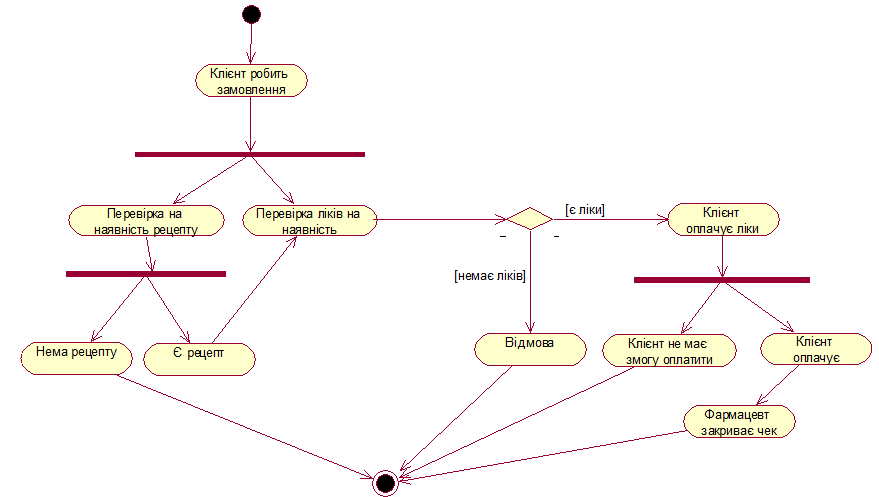
****

Рисунок 14 – Діаграма взаємодій АІС аптеки «Будьте здорові»

**Діаграма станів** — діаграма, що визначає зміну станів об'єкту у часі, одна з діаграм моделювання поведінки в UML. Представляє об'єкт як автомат з теорії автоматів зі стандартизованими умовними позначеннями. Діаграма станів автоматичної інформаційної системи аптеки « Будьте здорові» зображена на рисунку 15:

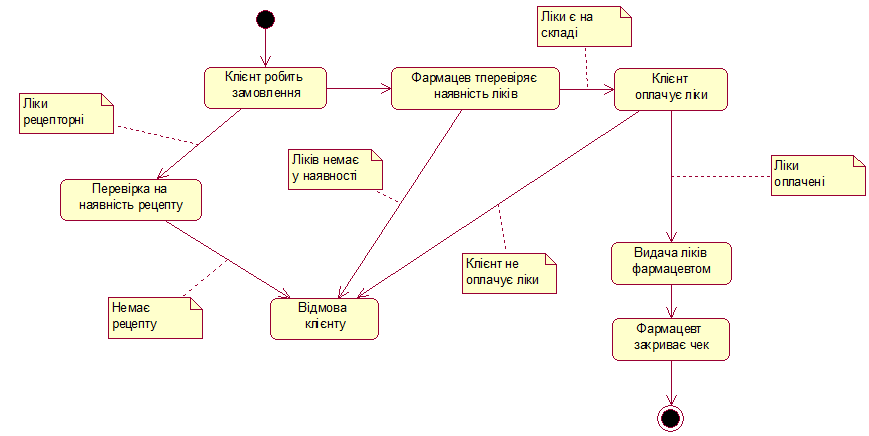
****

Рисунок 15 – Діаграма взаємодій АІС аптеки «Будьте здорові»

Діаграми станів застосовуються для того, щоб пояснити, яким чином працюють складні об'екти. Діаграма станів показує, як об'єкт переходить з одного стану в інший. На данній діаграмі зображена взаємодія існуючих елементів в аптеці, а саме фармацевта та клієнта, всі можливі розвитку подій та їх розв’язки.

1. **Діаграма дій**

Діаграма діяльності є не що інше, як спеціальний випадок діаграми станів, в якій всі або більшість станів є діями або станами піддіяльності. А всі або більшість переходів є нетригерними переходами, які спрацьовують після завершення дії або піддії в станах-джерелах. Кожна діаграма діяльності повинна мати єдиний початковий і єдиний кінцевий стан. При цьому кожна діяльність починається в початковому стані і закінчується в кінцевому стані.  
Діаграма дій АІС аптеки «Будьте здорові» зображено на рисунку 16:

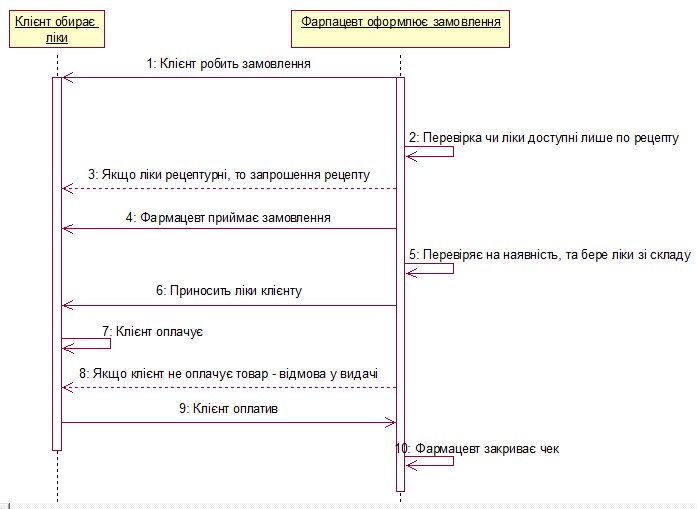
****

Рисунок 16 – діаграма діяльності АІС аптеки «Будьте здорові»

На діаграмі діяльності зображена послідовність взамодій клієнта з фармацевтом, варіанти розвитку подій які призведуть до кінцевої точки або ж закінчиться взаємодія завчасно, коли клієнт так і не отримає замовлення.

**10. Технічне завдання**

Технічне завдання на розробку програмного забезпечення для персоналу та адміністратора аптеки «Будьте здорові».

Розробити локальну базу даних для віддаленого доступу з усіх комп’ютерів мережи в аптеці. Створити 3 окремих додатки: для клієнта, адміністратора та персоналу, до персоналу входить фармацевти та комірники (кладовники).

Можливості адміністратора:

* Додавання інформації робітника після прийняття його на роботу;
* Редагування інформації персоналу;
* Видалення інформації робітника після звільнення;
* Відкритий доступ до всіх баз аптеки: препарат, склад, операцій та чек. Перевірка коректності даних в таблицях.

Можливості робітників:

**Фармацевт**

* Доступ до таблиці складу та чек;
* Можливість додавання записів в таблиці: чек та операція, під час опрацювання замовлення клієнта.

**Комірник**

* Доступ до таблиці складу;
* Додавання та редагування даних ліків в таблиці склад;
* Списання (видалення) ліків зі складу;

**Клієнт**

* Доступ до таблиці препарат;
* Можливість перегляду ліків які є в аптеці дистанційно, не чекаючи черги до фармацевта.

Для кожного користувача бази даних (Адмін, персонал, клієнт) створити окремі логіни для входу в базу та ролі. Пз для клієнта встановити на термінал в аптеці, для дистанційного перегляду ліків в разі необхідності.

В базі розробити 5 таблиць: Preparat, Sklad, Chek, Operaciya, Worker.

**Preparat**

* id\_pr – числове поле, ключ, включає в себе id препарату;
* nazva\_pr – короткий текст, призначено для збереження назви ліків;
* vurobnuk – короткий текст, інформацію про фірму виробника;
* pruznachenya – числове поле, інформація про тип категорії ліків;
* price – числове поле, ціна ліків.

Всі поля таблиці є обов’язковими для введення.

**Sklad**

* id\_com – лічильник, id кожної комірки на складі;
* id\_pr – числове поле, id препарату, який знаходиться в комірці;
* cilkist\_pr – числове поле, кількість препарату в комірці;
* Data\_postachanya – дата/час, дата коли цю комірку заповнили препаратом;
* Data\_vupysky – дата/час, дата випуску ліків;
* termin\_prudatnosti – дата/час, дата до якої ліки можна приймати;

**Chek**

* id\_cheka – числове поле, id чека;
* Data\_CHEKA – дата/час, інформація про дату отримання чека.

**Operaciya**

* id – ключове поле;
* id\_Cheka – id чека;
* id\_pr – id препарата, який було придбано;
* count – кількість препарату, який було придбано.

**Worker**

* id\_worker – ключове поле, унікальний код працівника;
* PIB – прізвище, ім’я, по батькові;
* Post – посада працівника;
* Login – логін для входу;
* Pass – пароль для входу;
* Phone\_number – номер телефону.

Розробити роль та дані для зв’язку з базою

Вхідні дані:

* Ліки (код, назва, рік випуску, термін придатності, виробник, призначення, ціна, кількість, дата постачання).

Вихідні дані:

* Формування загальної вартості чека.

Для реалізації програмного продукту обрано середовище швидкої розробки додатків Embarcadero RAD Studio 10.2, а саме в С++ Builder.

Borland C++ Builder є досить гнучким та зручним для створення програмних продуктів. Його об’єкти та розширені бібліотеки не тільки оформити отримання та динамічну обробку інформації, а й графічно і режимі реального часу відображати зміну вхідних параметрів, математичної моделі та системи і цілому.

Borland C++ Builder – це надзвичайно потужна система розробки прикладних програм для Windows. Вона впевнено займає передові позиції як серед професійних програмістів, так і серед людей, що ніколи професійно не займались програмуванням. Професіонали використовують C++ Builder для побудови складних аплікацій з розподіленими базами даних і для багатьох інших цілей. Спеціалісти саме різних спеціальностей створюють засобами Delphi невеликі прикладні програми для рішення своїх професійних задач.

Переваги C++ Builder полягає в мові C++, який лежить в основі. Це одна з найпотужніших сучасних алгоритмічних мов загального призначення. Наряду із своєю простотою C++ Builder дає широкі можливості по розробці складних і ефективних програм.

Інтегроване середовище розробки (ІСР) C++ Builder – це середовище, в якому є все необхідне для проектування, запуску і тестування програмних продуктів і де все націлено на полегшення процесу створення програм. ІСР інтегрує в себе редактор кодів, відгадчик, інструментальні панелі, редактор зображень, інструментарій баз даних та багато іншого.

C ++ Builder і Delphi стали одними з найпопулярніших на сьогоднішній день інструментів для створення як настільних, так і корпоративних інформаційних систем.

Термін розробки програмного забезпечення півтора місяці, з 08.01.2021 по 16.02.2021.