ВСТУП

Актуальність дослідження. Медицина є однією із найважливіших сфер в кожній країні світу, в тому числі і в Україні. Однією з можливостей підвищення рівня якості медицини є впровадження медичних інформаційних систем в медичні заклади. Хоча перші медичні системи були створені досить давно і в останній час вони активно розвиваються, їм виділено не досить багато уваги. Більшість систем переходять до стану технологічного старіння, доки нові ще не перейняли кращі практики функціонування.

Представлена робота присвячена розробці медичної інформаційної системи з основною функціональністю і подальше її вдосконалення методами машинного навчання.

Мета та завдання дослідження. Метою магістерської роботи є покращення процесу прогнозування спеціальності лікаря, що ґрунтується на наборі симптомів пацієнта за допомогою створення медичної інформаційної системи з базовою функціональністю на основі аналізу та вдосконалення алгоритму машинного навчання з вчителем.

Для досягнення поставленої мети у роботі необхідно виконати наступні завдання:

– виявити основні потреби медичних закладів і принципи їх роботи, які можна задовольнити програмними засобами;

– проаналізувати сучасні аналоги медичних інформаційних систем;

– розробити веб-додаток медичної інформаційної системи з можливостями, яка задовольнить потреби медичного закладу;

– виявити найбільш доречний алгоритм машинного навчання для прогнозування спеціалізації лікаря по симптомам пацієнта;

– провести вдосконалення алгоритму для поставленої задачі.

Об’єкт дослідження. Об’єктом дослідження є процес прогнозування спеціальності лікаря, що ґрунтується на наборі симптомів пацієнта.

Предмет дослідження. Предметом дослідження є розробка і вдосконалення медичної системи засобами машинного навчання для прогнозування лікаря ґрунтуючись на симптомах пацієнта.

Наукова новизна та практична значущість результатів роботи полягає в наступному:

– розроблено нову медичну систему з використанням нових програмних технологій;

– досліджено зв’язки медичних значень для підвищення ефективності навчання машинної моделі;

– вперше в Україні використано засоби машинного навчання для автоматизації запису на прийом в медичних закладах.

Програмною системою можуть користуватись як медичні працівники, так і потенціальні клієнти медичного закладу, які мають намір здійснити онлайн—запис на прийом. Розробка дозволить на основі обраних симптомів клієнтом, прогнозувати найбільш ймовірного спеціаліста, проаналізувати поточний розклад прийомів медичного закладу і створити відповідний прийом, без втручання медичного працівника.

РОЗДІЛ 1. ОСНОВИ МЕДИЧНИХ ІНФОРМАЦІЙНИХ СИСТЕМ В УКРАЇНІ ТА МАШИННІ МЕТОДОЛОГІЇ НАВЧАННЯ

1.1 Основні поняття про медичні інформаційні системи

Медична інформаційна система – інформаційна система призначена для автоматизації та спрощення робочих процесів лікувальної установи або закладу[1].

1.1.1 Класифікація медичних інформаційних систем

Класифікація медичних інформаційних систем(МІС) системи здійснюється за різноманітними ознаками.

МІС мають різні ступені автоматизації процесів, а також різні методи обробки інформації, саме тому їх розрізняють на автоматичні або автоматизовані.

Автоматизовані системи керуються за допомогою людей, тому обробка інформації та деяка частина системних операцій по її збору виконується людьми. При цьому існують медичні інформаційні системи, які виключають участь людини в будь яких процесах збору або обробки інформації. Такі системи називаються автоматичними [2].

МІС також поділяються за типом інформаційної бази, тому можна виділити системи які оперують даними, або знаннями. Варто зауважити, що системи, що оперують знаннями – це системи створені за участі спеціалістів, які безпосередньо беруть участь в розробці системи, вносять корективи і діляться експертною думкою. Саме тому результати функціонування таких систем дуже близькі до реальних медичних процесів.

Медичні інформаційні системи діляться за видом розв’язуваних задач, тому їх ділять на наступні групи:

– системи моніторингу;
– інформаційно-довідкові;
– вимірювальні системи;
– діагностичні системи, або системи прогнозу;

– системи для автоматизованого пошуку;
– інформаційно логічні;
– керуючі системи управління.

Інформаційно-довідкова система призначена для пошуку інформації, але окрім цього виконує згрупування необхідної інформації з метою формування спеціалізованого медичного документу.

Інформаційно-логічна система як правило призначена для трансформації інформації з метою отримання нової інформації, що відсутня в інформаційному масиві.

Системи управління створюються з спеціальною функціональністю направленою на прийняття керуючих рішень, групування та координацію робіт. Функції таких систем несуть в собі, як правило, лише організаційних характер і містять в собі мінімум медичного напрямку [3].

Найбільш популярними в сьогоденні стали інформаційно-пошукові системи (ІПС). Такі системи також поділяються в залежності від характеру інформації з якою належить працювати, а саме на документальні та фактографічні.

Судячи з назви, фактографічних інформаційно-пошукових систем можна зробити висновок що вони містять в собі набір фактичних даних. Якщо порівнювати з класичними прикладами, то така система являється аналогом технічних паспортів, каталогів, або довідників, що зроблені з паперу.

Для збереження фактичних даних комп’ютерних інформаційно-пошукових систем, як правило, використовуються реляційні бази даних (БД). Реляційні бази даних представляють з себе іменовані колонки, в яких зберігаються найменування різних характеристик об’єктів. Рядки містять в собі безпосередні дані, тобто значення іменованих колонок, для конкретних об’єктів. Представленням інформаційно-пошукових систем являється інформація зібрана в документ.

До прикладів описаної системи можна віднести картотеки історій хвороб, бібліографічні картотеки, або картотеки іншого характеру. Під час пошуку інформаційно-пошукова системі документального виду представляє

ідентифікатори необхідних документів, перелік заголовків, або умовні адреси, де можуть зберігатися необхідні документи.

Оцінку для знайденої інформації, яка розміщена в знайдених документах виконує працівник, тобто людина.

Для медичних інформаційних систем вид керуючих систем не дуже відрізняється від загального характеру, тобто виконує збір інформації про об’єкти управління, передачу інформації в спеціалізовані органи управління, функції створення керуючого рішення, обробку даних.

Також існує ієрархічний принцип класифікації медичних інформаційних систем, функція якого полягає в створенні багаторівневої структури охорони здоров’я, як окремої галузі.

Такий класифікатор у загальному випадку розділяється на чотири рівня:

– базовий рівень, також має назву клінічного;
– рівень лікувально-профілактичного закладу;
– територіальний рівень;
– державний рівень.

До базового рівня відносяться лікарі різного профілю. На лікувально-профілактичному рівні знаходяться диспансері, заклади швидкої допомоги, поліклініки, або стаціонар[3]. До територіального рівня вже належать спеціалізована або профільні медичні служби, або регіональні органи керування. Відповідно державний рівень містить в собі органи управління, або державні заклади.

У рамках всіх рівнів класифікації медичних інформаційних систем виконується за функціональним принципом, іншими словами в залежності від поставлених задач і цілей, що вирішуються системою.

Важливим аспектом медичних систем загалом, являється поділ закладів на рівні надання медичної допомоги.

Всього виділяють наступні рівні медичної допомоги:

– первинна медична допомога;
– вторинна медична допомога;

– третинна медична допомога.

Первинна медична допомога є основною і найбільш поширеною серед медичних закладів. Рівень надання такої допомоги є найбільш приближеним до населення. До первинної медичних допомоги можна віднести лікування найбільш поширених хвороб, отруєнь, травм, проведення консультації лікаря, діагностику, проведення різноманітних заходів проти епідемій та інше.

Первинна допомога як правило надається лікарями загальної практики, педіатрами, терапевтами, або сімейними лікарями.

Для закладів первинної медичної допомоги розміри та процедури виконання робіт встановлюється центральним органом виконавчої влади.

Коли створюються умови, за яких пацієнт має певні медичні показання, лікар, що працює в закладі первинної медичної допомоги, може видати направлення на вторинну, тобто планову спеціалізовану, або на третинну, тобто високоспеціалізовану медичну допомогу. Варто зауважити, що такі направлення не являються обов’язковою процедурою.

Спеціалізована медична допомога, вона ж вторинна, надається при умовах захворювання пацієнта, коли стандартних методів лікування первинної допомоги недостатньо і необхідно провести спеціальну діагностику та лікування. В закладах другого рівня надання медичної допомоги лікарі можуть надати лікування та консультацію, провести профілактику, або діагностику більш високого рівня в порівнянні з лікарями загальної практики.

Третинна медична допомога є найбільш спеціалізованою і надається тільки в випадках дуже тяжких або складних захворювань, при яких є необхідність проведення складних засобів лікування або діагностики. Заклади третинної медичної допомоги як правило являються стаціонарами, обладнанні високоспеціалізованим обладнанням і містить в собі фахівців вищого класу.

1.1.2 Інформаційне забезпечення медичних інформаційних систем

Основною характеристикою медичних інформаційних систем є наявність великого обсягу знань та даних.

Обробка знань та даних можна розділити на три базові етапи [4].

Елементи інформації, що знаходяться на першому етапі розміщуються у заздалегідь визначених структурах. Представниками таких структур є бази знань, або бази даних. Під час другого етапу бази знань та бази даних проходять процес упорядкування, при цьому змінюється порядок розташування інформації, взаємозв'язки між елементами інформації а також їх структура.

Під час третього етапу здійснюється використання, тобто експлуатація баз даних. Сюди можна віднести прийняття рішень, зміну інформації в базах, а також пошук необхідної інформації.

До інформаційного забезпечення медичних інформаційних систем відносять наступні речі:

– історія хвороб;
– епікриз;
– виписка з історії хвороб;
– стандартизована карта обстежень;
– діагностична оцінка;
– інформативна оцінка;
– критерії ефективності обстеження та лікування;
– перелік медичних термінів та понять.

У сьогоденні практики створення автономних медичних інформаційних систем переходить на другий план. В еру швидкого розвитку інтернету і масовості інформації управління системою в рамках окремого медичного підрозділу для вирішення окремих задач стає неефективним[4].

На зміну старим практикам наступає розвиток інтеграційних систем, які мають змогу взаємодіяти між собою, для швидкого обміну інформацією, а також використання окремих розвинених модулів в інших інформаційних системах.

Така взаємодія містить в собі багато різноманітних аспектів. В першу чергу сюди можна віднести стандарти даних загального відкритого характеру, які обробляються та зберігаються в медичних системах. Тут же варто відмітити забезпечення відкритих стандартів щодо механізмів та засобів взаємодії систем.

До другого основного аспекту відносять технічну стандартизацію медичних інформаційних систем. Зрозуміло, що технічні аспекти реалізації таких систем можуть відрізнятись один від одного, для них можуть використовуватись різні технології та підходи, але при цьому стає необхідність придержуватись максимально можливої стандартизації, коли діло стосується протоколів обміну інформацією, певних інтерфейсів, форматів даних і тому подібного.

Зважаючи на тенденції сьогодення стосовно медичних інформаційних систем варто мати на увазі, що необхідність і реальна можливість такої стандартизації являється дуже важливою частиною системи.

1.1.3 Медичні апаратно-комп’ютерні системи

Медична апаратно-комп’ютерна система (МАКС) є одним із різновидів медичних систем, які призначені для вирішення спеціалізованих задач. В сьогоденні перехід на використання комп’ютеризованої апаратури є напрямком в сфері медицини, який поширений в сучасних практиках. Використання комп’ютера в комбінації з управляючою або вимірювальною апаратурою в медичній сфері, дало можливість створювати нові ефективні засоби для реалізації автоматичного групування інформації про стан пацієнта, обробку представленої інформації в режимі реального часу, а також можливість управління станом хворого.

Описаний процес породив створення медичних апаратно-комп’ютерних систем. Такі системи змінили апаратні методи досліджень та інтенсивних терапій на новий якісний рівень.

МАКС надають можливість повної автоматизації, а також інформаційної підтримки для лікувального або діагностичного процесу. Варто зауважити, що це процеси виконуються при безпосередньому контакті з пацієнтом. МАКС мають альтернативну назву програмно-апаратних комплексів, або більш розгорнуту назву апаратно-комп’ютерні та мікропроцесорні медико-технологічні автоматизовані інформаційні системи.

МАКС можна віднести до медичних інформаційних систем, які призначені для закладів первинної медичної допомоги (базового рівня). Як було описано

раніше, на представленому рівні надання медичної допомоги відбувається безпосередній контакт з пацієнтами в режимі реального часу. Для таких систем характерна робота не лише обчислювальної техніки, а також спеціальне медичне обладнання, відеотехніка та засоби для онлайн-зв'язку.

Також до МАКС можна віднести наступні типи систем:

медичні системи, що призначені для моніторингу за станом хворих;

системи інформаційного аналізу даних стосовно ультразвукової діагностики, ЕКГ, томографії та ЕЕГ;

системи автоматичного аналізу даних вірусологічних та мікробіологічних досліджень, а також аналізу тканин та клітин людини.

Системи з переліченого списку направлені на підвищення якості лікувальної, профілактичної та діагностичної роботи. Особливо важливими такі системи постають при умовах з відсутністю необхідної кількості кваліфікованого персоналу, в умовах масового обслуговування та за недоліком часу.

Одними з основних задач МАКС є підвищення продуктивності роботи медичного персоналу та підвищення якості діагностичних і лікувальних процесів. Поставлені задачі вирішуються впровадженням комп'ютерних технологій в процеси лікування та діагностики. В той час якість цих процесів досягаються за допомогою повноти та швидкості роботи з спеціалізованою медичною інформацією.

За функціональними можливостями МАКС поділяються на:

– багатофункціональні;
– спеціалізовані;
– комплексні.

Спеціалізовані (одно функціональні) системи призначені для проведення досліджень окремих досліджень, які належать до певного виду. В випадку спеціалізованої системи буде проводитись лише електрокардіографічне дослідження.

Відповідно до назви, багатофункціональні системи забезпечують проведення досліджень не одного виду, як у спеціалізованих, а декількох видів. Тому такі

системи можуть забезпечити одночасно електрокардіографічне та електроенцефалографічне дослідження.

В ситуаціях, коли необхідне комплексне забезпечення стосовно медичних задач, використовуються комплексні системи. Для прикладу можна привести систему для моніторингу палати, де проводиться спостереження за пацієнтом, і в той же час проводити відслідковування важливих даних стосовно пацієнта і проводити контроль апаратів для штучної вентиляції легень.

МАКС також розділяють за призначенням, тому можна виділити наступні системи;

– системи що проводять функціональні та морфологічні дослідження;
– системи для моніторингу;
– системи призначені для керування лікувальним процесом;
– системи для проведення діагностики в лабораторних умовах;
– системи призначені для наукових медичних досліджень.

Системи, які спеціалізуються на проведенні функціональних та морфологічних досліджень також поділяються на певні групи. З них можна виділити наступні:

– дослідження системи кровообігу;
– радіонуклідні дослідження;
– магнітно-резонансна томографія;
– дослідження органів дихання;
– дослідження органів відчуття (зору, слуху та ін.);
– дослідження головного мозку та нервової системи;
– рентгенологічні дослідження (в тому числі комп'ютерна томографія);
– ультразвукова діагностика.

До цього було описано лише технічну, тобто апаратну частину призначену для МАКС, але не варто забувати про не менш важливу програмну частину, без якої апаратна немає ніякого сенсу.

Під програмним забезпеченням розуміють програми, алгоритми, математичні методи обробки даних і вони власне призначені для успішної роботи всієї медичної системи.

Для розробки програмного медичного забезпечення необхідні спеціалісти з відповідною технічно-медичною освітою. Також часто зустрічаються випадки, коли розробка супроводжується спеціалістами двох типів — технічні спеціалісти відповідають за програмну реалізацію, а медичні працівники виконують постановку задач, які необхідно реалізувати. Якщо річ заходить про розробку мікропроцесорних пристроїв, то тоді необхідні спеціалісти, які мають справу з мікроелектронікою.

Для створення найбільш досконалої системи, її необхідно оснастити інтегрованим програмним забезпеченням. За допомогою такого рішення, медичний працівник має змогу працювати з цілісною системою, яка повністю покриває процес дослідження, що включає в себе різноманітні етапи.

В описаному програмному забезпеченню відрізняють наступні функціональні модулі:

- підготовки до дослідження;
- безпосереднє проведення дослідження;
- аналіз та редагування записів;
- проведення обчислювального аналізу;
- опис висновку;
- робота з архівом.

1.2 Огляд та аналіз медичних систем в Україні

Для огляду та аналізу медичних систем України було відібрано наступні медичні системи:

– ЕМСІМЕД;
– Доктор Елекс;
– Medics.

Описаний вибір обґрунтовується наступними критеріями. Медична система ЕМСІМЕД була розроблена згідно з стандартами ISO та МОЗ України. В даній системі реалізований технічний захист інформації, а також містить великий перелік окремих модулів.

Доктор Елекс найпоширеніша на сьогоднішній день, медична система в Україні, яка була створена в 2005 років. Ще в 2018 році в цій системі був перелік близько 5 мільйонів пацієнтів, тому її можна вважати найбільшою за об’ємом даних в Україні [5]. В системі Доктор Елекс також реалізовані різноманітні модулі призначені для автоматизації медичних процесів.

Medics система молодша за віком, ніж попередні дві медичні системи, але розроблена за допомогою більш сучасним технічних засобів. Особливо варто відмітити, що це хмарна медична система [6].

Розробка медичної інформаційної системи «ЕМСІМЕД» забезпечила автоматизацію для основних медичних процесів, які необхідно забезпечити в лікувальних закладах сьогодення [7]. Представлена система позиціонує за свої переваги зручну роботу з документами, мінімізує помилку зі сторони людського фактору, підвищує ефективність профілактичних та лікувальних засобів, підвищує якість обслуговування пацієнтів, які були долучені до системи і користуються її можливостями. Також важливим аспектом являється зниження витрат на формування різноманітних звітів та діаграм.

ЕМСІМЕД має зручну архітектуру для користувача системи, а також володіє можливостями супроводу та розширення проекту, в умовах, коли медичному закладу необхідні додаткові реалізовані програмні засоби для своєї роботи. Система також володіє інтеграціями з іншими медичними системами. Так як система розроблена за принципом модульності, то присутня можливість зручного налаштування системи під окремий заклад і його потреби, при яких не потрібно змінювати програмний код, для реструктуризації бізнес процесів.

Описана система володіє одним із найбільших переліків модулів, які призначені для вирішення задач в медичних закладах:

– поліклініка;

– стаціонар;
– архів зображень PACS;
– управління партнерськими відносинами;
– статистика;
– управління персоналом;
– керування запасами;
– реєстратура;
– керування документами;
– управління організацією;
– лабораторія;
– медичні послуги;
– додаткові платні модулі.

Модуль поліклініка — модуль необхідний для автоматизації робочих процесів медичного працівника, а також для основної частини задача на кожен день, які виконуються в рамках лікувально-профілактичного закладу.

Модуль реєстратура – призначений для автоматизації основних задач, які проходять в реєстратурі. Основна виконувана робота модулю – спрощення роботи з вхідною інформацією, спрощення роботи з потенціальними пацієнтами, автоматизація регулярних операцій, які входять до обов’язків працівників реєстратури.

Управління документами — один із найважливіших інструментів, який призначений для вирішення проблем документообігу, а також задачі пов’язані за ним.

Керування персоналом — призначений для управління персоналом в лікувально-профілактичному закладі. Сюди відносяться картки працівників медичного закладу, розрахунок проведеного робочого часу працівника, генерація діаграм та звітів, що відносяться, що встановлені нормами державних актів.

Керування організацією — призначений для правильної роботи з структурами підприємств та їх підрозділами. Допомагає об’єднати заклади до єдиної структури та керувати ними.

Модуль стаціонару розрахований на автоматизацію процесів, що призначені для стаціонарних відділень.

Модуль лабораторія спрямований на підвищення рівня роботи спеціальних відділень для досліджень а також лабораторій. Також необхідний для швидкого обміну необхідною інформацією між цими підрозділами.

Модуль керування запасами призначений для ефективного планування інформації стосовно проведення необхідних закупок. Сюди можна віднести організацію обліку лікарських засобів, а також спеціального видаткового матеріалу для медичного персоналу.

Модуль статистики необхідний для створення спеціальних діаграм та звітів, що містять в собі необхідну згруповану інформацію лікувально-профілактичних закладів. Також присутня можливість завантаження файлів в електориний варіант і подальшу їх друк в разі потреби.

Архів зображень PACS розроблений для роботи з спеціальними медичними зображеннями отриманими в ході досліджень і подальшого їх збереження і використання в медичних процесах.

Наступним конкурентом постає медична система Доктор Елекс. Як уже було вказано раніше, ця система виділяється своєю базою пацієнтів, що є великою перевагою серед інших медичних систем.

Доктор Елекс містить в собі наступні модулі:

– електронна медична карта;
– редактор шаблонів документів;
– реєстратура;
– лікар;
– лабораторія;
– обробка відео і зображень;
– звіти;
– аудит документів;
– керівник;
– фінанси.

Електронна медична картка — це основний модуль представленої медичної системи. Вона призначена для збереження всієї необхідної інформації стосовно пацієнта — дані введені при реєстрації, результати проведених прийомів у лікаря, результати лабораторних досліджень, медичні знімки (рентген, МРТ).

Особливістю системи являється використання деревовидна структура шаблонізованих даних, які дають змогу формувати шаблонну інформацію без клавіатурного вводу. Для використання цієї реалізації медичному працівнику достатньо вибрати необхідні елементи дерева, для заповнення необхідних форм.

Також до унікальних особливостей, які не зустрічалися у попереднього аналогу можна віднести редактор шаблонів документів.

Редактор шаблонів документів — це спеціальний інструмент, за допомогою якого спеціаліст, без втручання в програмний код, має можливість налаштувати документ і привести його автоматичне заповнення даними лише засобами інтерфейсу системи.

Шаблони дозволяють:

- трансформувати зміст та зовнішній вигляд окремого документа, основуючись на потребах медичного працівника або клініки;
- використовувати заготовлені шаблони даних готового переліку лише натискаючи на низ курсором в системі;
- проводити формування документу в час проведення прийому, що допомагає оптимізувати процес і знижує ризик втратити важливі дані;
- створювати користувацькі норми для окремих лабораторних даних.

Всі інші модулі медичної системи Доктор Елекс мають теж саме призначення, що і в системі ЕМСІМЕД, тому їх подальший опис немає ніякого сенсу.

Наступною розглянутою системою є Medics. Як уже було зауважено, це хмарна медична інформаційна система, що краще виділяє її серед попередніх систем.

Представлена медична система презентує наступні можливості:

- електронна реєстратура;

- робочий кабінет лікаря;
- контроль за роботою на медзакладі;
- комунікація з клієнтом;
- реклама і маркетинг;
- формування бази користувачів в регіоні.

Серед переліченого списку варто виділити блок комунікації з клієнтом. На даний момент це перша система, яка пропонує модулі розраховані не лише на покращення та оптимізацію внутрішньої роботи закладу, а також модуль орієнтований на клієнтів клініки. Даний блок використовує соціальні месенджери та SMS-сповіщення для взаємодії з пацієнтами. В повідомленнях вказується інформація стосовно дати та часу проведення огляду, або обстеження, процесу вакцинації тощо. Також за допомогою блоку комунікацію є можливість дізнаватися актуальну інформацію стосовно наявності медичних препаратів, за програмою «Доступні ліки». Для окремих випадків є можливість надсилання приватних повідомлень клієнту, якщо в цьому є потреба.

Також цікавим нововведенням є проведення рекламних та маркетингових процесів, які спрямовані на підвищення репутації медичного закладу.

Окрім реалізованих модулів варто зауважити, що система являється хмарною, тобто працює за принципом сайту, що спрощує її впровадження та використання організаціями [8].

1.3 Моделі та алгоритми машинного навчання

1.3.1 Основні типи машинного навчання

Машинне навчання (Machine Learning) — це велика частинна, що відноситься до штучного інтелекту і займається вивченням методиками створення та побудови алгоритмів, які можуть навчатися [9].

Машинне навчання розділяється на два основні типи.

Перший спосіб полягає в навчанні за прецедентами, його також називають індуктивним навчанням. Даний тип оснований на пошуку закономірностей за окремими емпіричними даними.

До другого типу машинного навчання відносять дедуктивне. Дедуктивний тип навчання полягає в формуванні знань спеціалістів і трансформування їх до формату бази знань. Варто зауважити, що дедуктивне навчання скоріше відносять до області експертних систем. Тому навчання по прецедентах і є по своїй суті машинним навчанням.

Машинним навчанням є скупчення різним методів та математичних рішень, таких як методи оптимізації або математична статистика та інші класичні математичні дисципліни. Але цей процес не є суцільною математикою і має свої особливості, до яких відноситься певні проблеми ефективності при обчисленнях та процеси перенавчання.

Перш за все більшість методів навчання за прецедентами створювались для заміни стандартних статичних засобів, з метою підвищити ефективність виконуваних процесів. Коли методи стосуються вибірки даних та інтелекутальним аналізом, то це має назву Data Mining.

Розділи машинного навчання, які основним чином спрямовані на теоретичну частину виділяють в окремий напрям, який називають теорією обчислювального навчання (Computational Learning Theory, COLT).

До цього було машинне навчання було розглянуто лише с математичної точки зору, але насправді сюди відноситься і практична інженерна дисципліна.

Машинне навчання є певним процесом, де майже неможливо все теоретично розрахувати. Процес включає в себе відпрацювання методів та алгоритмів на практиці, спостереження за результатами та внесення певних змін до їх роботи.

Для створення ефективного алгоритму доводиться проводити організацію виборчого пошуку при вирішенні задач, для компенсації некоректних припущень, зроблених в теорії. Майже всі проведені дослідження при навчанні прецедентів основуються на проведенні експериментів на реальних та модельних даних, які мають змогу показати практичну цінність методу.

1.3.2 Постановка задачі навчання по прецеденту

Після визначення того, що в алгоритми не працюють без практики перейдемо до загальних принципів при постановці задач індуктивного навчання.

Для того щоб виконати постановку задачі, необхідно задати кінцеву множину прецедентів. Під множиною розуміється набір об’єктів або ситуацій. По кожному з таким об’єктів повинні бути зібрані деякі дані. Іншою назвою даних прецеденту є опис. Якщо був проведений збір всіх описів прецедентів, то це називається вибіркою для навчання. За зібраними даними стає необхідність виявити можливі взаємозв’язки, закономірності або залежності. Варто зауважити, що залежності повинні бути властиві не лише конкретній вибірці даних, але й повинні задовольняти всі можливі прецеденти, навіть ті, які ще не були застосовані.

Для таких ситуацій інколи використовується термін з назвою відновлення залежностей за емпіричними даними».

Серед всіх відомих способів опису прецедентів найбільш поширеним є опис за ознаками. Для його проведення вибирається деяка кількість показників, що вимірюються для всіх прецедентів. Показники можуть бути різного характеру. Найпростіші — це числові показники, які представляють собою числовий вектор. Також існують і більш складні варіанти, наприклад, зображення, тексти, відеоряди, часові ряди та інші.

В першу чергу при виконанні завдань навчання по прецедентах проводиться фіксація моделі відновлюваної залежності. Після цього додається показник якості. Його значення вказує значення якості, відносно того як модель описує вибірку даних. Для пошуку такого набору параметрів моделі, при яких показник якості має задовільне значення для конкретної вибірки, використовується алгоритм навчання (learning algorithm). Як правило, для процесу налаштування моделі за вибіркою використовуються чисельні методи оптимізації.

Необхідно відмітити, що поняття алгоритму в зарубіжній літературі зустрічається лише як обчислювальна процедура, яка робить налаштування моделі за певною вибіркою.

Виходом алгоритму навчання є функція, що має близьку за значенням невідому залежність. Варто зауважити, що класифікатором (classifier), як раз називають апроксимуючу функцію. Інколи вона має назву гіпотези або концепту. В задачах відновлення регресії - функцією регресії; іноді просто функцією.

У російськомовній літературі апроксимуюча функція називається алгоритмом, що за сенсом відрізняється від зарубіжного поняття. В даному випадку алгоритмом функція називається для того, щоб виділити її можливість припускати ефективну реалізацію.

1.3.3 Типологія задач навчання по прецедентах

Задачі по навчанню по прецедентах можна розділити на основні стандартні типи:

– навчання без вчителя;
– навчання з учителем;
– часткове навчання;
– навчання з підкріпленням;
– активне навчання;
– трансдуктивне навчання;
– метанавчання;
– динамічне навчання.

Найбільш розповсюджений спосіб машинного навчання — це навчання з учителем (supervised learning). Навчання з учителем проходить наступним чином. Окремо взятий прецедент представляє комбінацію з об’єкту як одного елементу з множини та відповіді для цього елементу. За мету навчання з учителем поставлено знайти залежності між об’єктами та їх відповідями і на основі цих результатів сформувати алгоритм, який уже зможе брати новий об’єкт з множини і знаходити для нього відповідь. Для визначення якості алгоритму береться середня помилка відповідей, які згенерував алгоритм, під час обробки всіх об’єктів вибірки.

Представлений вид навчання ділиться на наступні підтипи:

– задачі регресії;

– задачі прогнозування;

– задачі на класифікації;

– задачі ранжирування.

Особливістю задач класифікації (classification) є те, що відповіді для об’єкту в множині можуть мати лише обмежену кількість. Задачі такого характеру іменують мітками класів (class label). Безпосередньо в даному випадку класом є множина всіх об’єктів з даними значеннями мітки.

Унікальність задач по регресії (regression) є допустимі відповіді, що можуть бути числовим вектором, або дійсним числом.

Коли стає необхідність отримання відповідей множини об’єктів, з подальшим впорядкуванням їх значень відповідей використовуються задачі ранжирування (learning to rank). Представлений тип задач може містити в собі задачі класифікації, або задачі регресії. Така методика дуже часто використовується для інформаційного пошуку або для аналізу текстів.

Задачі прогнозування (forecasting) побудовані на часових рядах. Принцип основується на тому, що під час закінчення часового ряду необхідно виконати прогноз на майбутнє. Інколи для задач прогнозування використовуються методи класифікації, або регресії.

Наступним основним типом навчання є навчання без вчителя (unsupervised learning). Задачі без вчителя характерні тим, що при постановці задачі відповіді не надаються, тому приходиться встановлювати взаємозв’язки між об’єктами без вірних відповідей.

Задачі без учителя поділяються на такі підтипи:

– задача фільтрації викидів;

– задача кластеризації;

– задача скорочення розмірності;

– задача пошуку асоціативних правил;

– задача заповнення пропущених значень;

– задача побудови довірчої області.

Основною суттю задач кластеризації (clustering) є групування об’єктів в окремі кластери, які використовують дані схожості об’єктів. Якість такої методики визначається різними шляхами. Це може бути відношення внутрішньокластерних або відношення середніх міжкластерних відстаней.

Також існує комбіноване навчання, яке використовує поєднання навчання з учителем та без учителя. Така методика називається частковим навчанням (semi-supervised learning). Для такої методології характерна наявність пари об’єкту та відповіді, але відповіді заповнені лише для певної частини об’єктів. Яскравий приклад використання — автоматичне розбиття наборів текстів по рубрикам, зважаючи на те, що частина текстів уже має відношення до певних рубрик, а інша ні.

Також має місце навчання з підкріпленням (reinforcement learning). В даній ситуації об’єктами являються парні значення ситуації, та рішення, яке необхідно прийняти. В якості відповідей вказується оцінка правильності прийнятих рішень. Представлена методологія схожа із задачами прогнозування, так як тут також використовуються фактори часу. Прикладами практичних задач можна назвати створення інвестиційних рішень, автоматизацію управління технологічними процесами, або утворення процесу навчання роботів, які зможуть здатні до самонавчання.

1.4 Вибір інструментів для розробки та дослідження

Сьогодні використання додаткових інструментів розробки стало невід’ємною частиною створення програмного забезпечення. Це надає ряд переваг у вигляді швидкості та якості розробки. Щоб зрозуміти, які саме інструменти потрібні для розробки, в попередніх розділах був проведений огляд існуючих медичних інформаційних систем та алгоритмів машинного навчання. Роботу можна розділити на два основні етапи:

а) створення базового функціоналу медичної інформаційної системи;

б) провести вдосконалення медичної інформаційної системи за допомогою машинного навчання.

На основі отриманої інформації проведемо вибір інструментів, які допоможуть розробити та вдосконалити систему.

Перед виконанням завдання необхідно провести підготовчі роботи, які являються запорукою успішного виконання.

Важливим етапом розробки є встановлення необхідних інструментів на яких буде проводитися майбутня розробка програмного забезпечення.

Згідно з плану робіт стоїть необхідність створення медичної інформаційної системи з базовим функціоналом. Такі системи не являються малими за об’ємом і як правило для їх розробки використовуються «корпоративні» мови програмування. В даному випадку було вирішено використовувати мову програмування C# на платформі .NET Core для написання серверної частини системи. Для написання клієнтської частини було вирішено використовувати JavaScript-фреймворк Angular.

Зважаючи на інформацію описану раніше оберемо середовища розробки. Для зручного написання програмного коду на мові програмування C# зазвичай використовується Visual Studio.

Visual Studio - це інтегроване середовище розробки програмного забезпечення та ряд інших інструментальних засобів. Представлений продукт дозволяє розробляти як консольні програми, так і програми з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, WPF, а також веб-сайти, веб-застосунки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, що підтримуються Microsoft Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight.

Переваги використання представленого інтегрованого середовища розробки полягає в великому виборі різноманітного функціоналу, що зумовлює швидку та зручну розробку програмного забезпечення. Із основного можна виділити можливість зручного додавання фреймворків, які дуже просто інтегрувати в

систему розробки, зручне відображення програмної архітектури, широкий діапазон можливостей по дебагу програмного коду.

Також варто виділити можливість встановлення до Visual Studio додаткового плагіну ReSharper, що значно полегшує роботу програміста.

ReSharper (R #) - додаток (плагін), розроблене компанією JetBrains для підвищення продуктивності роботи в Microsoft Visual Studio.

Проводить статичний аналіз коду (пошук помилок в коді до компіляції) в масштабі всього рішення, передбачає додаткові кошти автозаповнення, навігації, пошуку, підсвічування синтаксису, форматування, оптимізації та генерації коду, надає 40 автоматизованих рефакторингів, спрощує юніт-тестування в середовищах MSTest і NUnit та ін. Підтримує мови програмування C # і VB.NET, а також надає можливість для роботи з ASP.NET, ASP.NET MVC, XML, XAML, сценаріями збірки NAnt і MSBuild.

Для фреймворку Angular було обрано редактор Visual Code. Це дуже легкий редактор програмного коду для кросплатформної розробки веб-додатків. Цей редактор підтримує ряд мов програмування, підсвітку синтаксису, рефакторинг, дебаг програмного коду та зручну навігацію по ньому.

Також при побудові запропонованої системи передбачається використання бази даних. Для використання баз даних запропоновано багато утиліт, одною з яких є SQL Server Management Studio.

Management Studio – це утиліта для Microsoft SQL Server 2005 і пізніших версій, яка призначена для конфігурації, управління і адміністрування всіх компонентів Microsoft SQL Server. Утиліта включає скриптовий редактор і графічну програму, яка працює з об'єктами і настройками сервера.

Для реалізацій потреб машинного навчання було вирішено використовувати безкоштовний кросплатформний та відкритий фреймворк машинного навчання ML.NET, що призначений для використання можливостей машинного навчання в додатках .NET.

Цей вибір був зроблений основуючись на побудові загального додатку медичної інформаційної системи на фреймоврку .NET. ML.NET дозволяє додавати

в додатки .NET можливості машинного навчання в автономному та підключеному режимах.

В основі ML.NET використовується модель машинного навчання. Ця модель визначає кроки, які необхідно виконати для отримання прогнозів на основі вхідних даних. За допомогою ML.NET є можливість отримати модель призначену для користувача, вказавши відповідний алгоритм, або імпортувавши навчені моделі.

Отриману модель можна додати в додаток і використовувати її для отримання прогнозів.

ML.NET працює в Windows, Linux і macOS з .NET Core або в Windows з .NET Framework, 64-розрядна версія підтримується на всіх платформах, 32-розрядна версія підтримується в Windows, за винятком деяких функцій.

За допомогою ML.NET є можливість отримати прогнози наступних типів:

– класифікація;
– регресія;
– виявлення аномалій;
– рекомендації;
– часові ряди;
– класифікація зображень.

РОЗДІЛ 2. РОЗРОБКА МЕДИЧНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Перед початком розробки медичної інформаційної системи необхідно визначити, які основні модулі будуть присутні. Після аналізу медичних систем популярних в Україні виділимо наступні елементи системи необхідні для зручної роботи:

- реєстратура;
- можливість керування персоналом;
- ведення обліку пацієнтів;
- можливість управління організацією;
- керування документами;
- автоматизація надання медичних послуг;
- введення звітів та аналітики;
- можливість роботи з архівами PACS.

Після того, як було отримано список необхідних елементів системи, розділимо їх на складові з точки зору побудови архітектури системи.

2.1 Проектування архітектурних складових медичної системи

Проектування будь-якої системи є одним із найважливіших етапів при створенні програмного продукту, адже від цього залежить подальший розвиток проекту, можливість його підтримки та подальша актуальність [10].

2.1.1 Проектування сутностей організація та підрозділ

Кожна клініка, яка буде використовувати медичну інформаційну систему являється організацією, яка має свої підрозділи. Це невід’ємна складова будь-якого закладу, який має велику кількість персоналу і багато внутрішніх процесів. Саме тому почнемо з опису таблиці організації.

Створену сутність організації іменуватимемо як Orgs. Таблиця Orgs міститиме у собі наступні поля:

– Id;
– GId;
– OwnerId;
– Name;
– OfficialName;
– Address;
– Phone;
– Email;
– TaxCode;
– ContactPerson;
– Note;
– IsActive;
– ParentId;
– LevelMedCare.

Поля Id та GId використовуються як ідентифікатори сутності організація. Поле Id при цьому використовує метод ітерації як унікальне значення, а поле GId використовує значення типу GUID і генерується випадковим чином.

Поле OwnerId в даному випадку просто дублюватиме поле Id і вводиться лише для зручності використання, тому що в подальшому представлене поле буде вводитися майже до всіх сутностей. Це потрібно для розуміння того, до якої організації що належить.

З найменування полів Name та OfficialName зрозуміло, що використовуватися вони будуть для найменування закладу, при цьому поле Name може містити в собі скорочену назву, тоді як OfficialName лише офіційну назву закладу. Поля Phone та Email міститимуть у собі значення офіційного телефону та електронної адреси відповідно.

Поле TaxCode створене для значення єдиного державного реєстру підприємств та організацій України (ЄДРПОУ).

Значення ContactPerson та Note являються текстовими полями. ContactPerson міститиме у собі інформацію про особу відповідальну за контакт з організацією.

Поле Note призначено для додаткової інформації про організацію. Це може бути короткий опис.

LevelMedCare – поле призначене для вказання рівня надання медичної допомоги.

ParentId – поле необхідне для розуміння ієрархії між організаційними структурами, якщо така існує. Загалом всі перелічені поля досить об’ємно зможуть передати інформацію про організацію.

Але окрім організації, постає необхідність проектування сутності підрозділу, як частину складової організації. Представлену сутність в подальшому будемо найменувати Departments і міститиме вона наступні поля:

- Id;
- GId;
- OwnerId;
- Code;
- Name;
- FullName;
- Note;
- DepartmentTypeId;
- DepartmentSubTypeId;
- Address.

Поля Id та GId в даному випадку також використовуються як ідентифікатори сутності. Поле Id використовує метод ітерації як унікальне значення, а поле GId використовує значення типу GUID і генерується випадковим чином.

Для сутності підрозділу поле OwnerId в даному випадку використовується для вказання того, до якої організації належить окремий підрозділ. В представленому полі буде зберігатися унікальний ідентифікатор організації.

Поле Code необхідно для вказання спеціального коду підрозділу, який є зарезервованим кодом.

Поля Name та OfficialName використовуватимуться для найменування підрозділів, при цьому поле Name може містити в собі скорочену назву, тоді як

OfficialName лише офіційну назву підрозділу. Note призначене для зберігання спеціальної інформації про конкретний підрозділ.

DepartmentTypeId та DepartmentSubTypeId числові ідентифікатори, які містять в собі значення довідника з типами підрозділів. DepartmentTypeId при цьому означає загальну класифікацію сутності – департамент, підрозділ, філія, в той час як DepartmentSubTypeId уже міститиме більш детальну інформацію про підрозділ. Наприклад підрозділ діагностичний, лікувальний або адміністративний. Довідник з списком типів департаментів буде описаний далі в роботі. І останнє поле це Address, виходячи з назви буде містити в собі текстове значення адреси підрозділу.

2.1.2 Проектування сутностей пацієнт та співробітник

Наступним за значенням в медичній системі є інформаційний облік пацієнтів та співробітників медичного закладу. Обидві сутності схожі між собою за змістом, тому було вирішено додати основну сутність з назвою Persons, яка міститиме загальну інформацію про людину, а також дві розширені сутності PatientCards та EmployeeCards, які міститимуть в собі інформацію про пацієнта та співробітника відповідно.

Основна сутність Persons повинна містити в собі основні атрибути, які змогли б описати людину, тому було вирішено додати наступні поля:

- Id;
- GId;
- OwnerId;
- DOB;
- Phone;
- Email;
- FullName;
- FirstName;
- SecondName;
- MiddleName;

– ShortName;

– GenderCode.

Поля Id та GId як і в інших сутностях використовуються в якості ідентифікаторів. Поле Id використовує ітерації як унікальне значення, а поле GId використовує значення типу GUID і генерується випадковим чином. OwnerId в використовується для вказання того, в якій організації була створена нова персона.

DOB – скорочення від Date of birth. Виходячи з назви зрозуміло, що поле призначене для збереження дати народження конкретної персони. Поля Phone та Email міститимуть у собі значення телефону та електронної адреси персони відповідно.

Наступні поля FullName, FirstName, SecondName, MiddleName, ShortName міститимуть в собі інформацію про прізвище, ім’я та по батькові. FullName при цьому призначене для зберігання повного прізвища ім’я і по батькові, а ShortName лише для повного прізвища і скороченого ім’я та по батькові.

GenderCode тестове поле, але міститиме в собі лише два значення - M та F, що скорочено з англійської Male та Female. Виходячи з цього розуміємо, що поле призначено для розуміння статі персони.

Після того як було описано сутність Persons з’явилася змога створити більш деталізовані сутності пацієнта та співробітника, які будуть унаслідуванні від базової.

Сутність PatientCards міститиме наступні поля:

– Id;

– GId;

– OwnerId;

– PersonId;

– RegNumber;

– RegDate.

Поля Id, GId, OwnerId виконують ті ж самі функції що і в інших сутностях.

PersonId містить в собі ідентифікатор, який посилається на сутність Persons. Таким чином є змога звязати всю необхідну інформацію про окрему персону з картою пацієнта.

Поле RegNumber містить в собі значення Id і призначено для створення унікального ідентифікатору пацієнта. З першого погляду здається що це немає ніякого сенсу, але насправді указане поле необхідно для інтеграції з іншими медичними системами. Ідентифікатор Id і справді являється унікальним, але він генерується кожного разу, при створенні нового об’єкту сутності. Основним правилом інтеграції являється стабільність ідентифікаторів, які не б змінювалися з часом.

Поле RegDate призначено для збереження дати створення карти пацієнта.

Після опису сутності карти пацієнта перейдемо до створення карти співробітника яка міститиме наступні поля:

– Id;
– GId;
– OwnerId;
– PersonId;
– RegNumber;
– RegDate;
– PositionTypeId;
– SpecializationId;
– DepartmentId.

Основні поля Id, GId, OwnerId, PersonId, RegNumber, RegDate нічим не відрізняються по своїй суті від однойменних полів описаних для сутності PatienCard.

Поле PositionTypeId містить в собі ідентифікатор на довідник позицій і необхідний для добавлення класифікації працівників. Наприклад працівник може бути лікарем, направляючим, або лаборантом.

Поле SpecializationId також містить в собі ідентифікатор який буде посилатися на довідник спеціалізації працівників медичного закладу.

Поле DepartmentId містить в собі ідентифікатор який посилається на раніше створену сутність підрозділу. Адже кожен працівник медичного закладу повинен числитись в певному підрозділі і дане значення дасть змогу утворити логічний зв'язок між сутностями.

2.1.3 Проектування сутностей прийом, діагностичний прийом та консультація

Теоретично було сплановано сутності організацію – що умовно представляє собою медичний заклад, а також сутності персон – пацієнтів та робітників медичного закладу. Тобто, було описано найважливіші сутності в інформаційній системі, тому настав час інформаційно описати взаємодію між пацієнтом і медичним працівником. Для цього створимо три сутності, які дуже схожі за своїм змістом, але дещо відрізняються за бізнес-процесами.

Прийом – базовий процес, під час якого відбувається взаємодія лікаря та пацієнта. Основна мета прийому – вирішити проблему пацієнта з медичної точки зору, провести бесіду, надати консультацію і тд.

Діагностичний прийом спрямований лише на постановку діагнозу пацієнта, при цьому постановка діагнозу безпосередньо пов’язана з спеціалізованим медичним обладнанням. Тобто представлений прийом використовується в тих випадках, коли неможлива постановка діагнозу пацієнта лише методами діалогу і візуальним контактом.

Консультація в даному випадку представлена як взаємодія лікар – лікар і використовується в випадку звернення медичних закладів з нижчим рівнем надання допомоги до медичних закладів з вищим рівнем надання медичної допомоги. Наприклад медичний заклад з первинним рівнем медичної допомоги проводить прийом пацієнта, але в нього виникають складнощі, які виходять за рамки лікування найпоширеніших хвороб і йому необхідна консультація лікаря спеціалізованої медичної допомоги.

Після розуміння того, які види прийому мають бути реалізовані в системі опишемо їх зміст. Створена сутність буде загальною для всіх видів прийомів, матиме назву Appointment і міститиме такі поля:

- Id;
- GId;
- OwnerId;
- CreateDate;
- AppointmentStateId;
- ScheduleTypeId;
- PatientCardId;
- EmployeeCardId;
- DepartmentId;
- Cost;
- StudyTypeId;
- ScheduleSlotId;
- AppointmentNumber.

Поля Id та GId як і в інших сутностях використовуються в якості ідентифікаторів. Поле Id використовує ітерації як унікальне значення, а поле GId використовує значення типу GUID і генерується випадковим чином. OwnerId в використовується для вказання того, в якій організації був створений новий прийом.

CreateDate поле яке містить в собі дату створення прийому. В цьому є необхідність, так як в подальшому буде створено графік прийомів для окремого лікаря або кабінету і нам необхідно буде розуміти коли буде відбуватися прийом.

AppointmentStateId – це поле яке посилатиметься на довідник зі статусами прийому. Прийом сутність непостійна, вона плине з часом і може змінювати свої статуси. Наприклад, до тих пір доки прийом ще не почався – він знаходиться в статусі «проект», але одразу після початку статус змінюватиметься на «в процесі». Відповідно після завершення прийому його статус змінюватиметься на «завершено».

ScheduleTypeId - також поле, яке може містити в собі ідентифікатори довідника з типом прийому. Таким чином використовуючи одну сутність, ми зможемо розуміти який тип прийому зараз відбувається – звичайний, діагностичний, або консультація.

PatientCardId та EmployeeCardId містять в собі ідентифікатори карти пацієнта та лікаря відповідно. Таким чином буде зрозуміло який спеціаліст проводить прийом і який пацієнт на цьому прийомі знаходиться.

DepartmentId існує для вказання в якому підрозділі відбувається прийом.

Cost – це поле для збереження вартості прийому. Не всі прийоми бувають безкоштовними, тому не виключаємо можливість того, що буде необхідно зберігати їх вартість.

StudyTypeId знову посилання на довідник, в якому буде зберігатися інформація про тип дослідження, якщо такий буде необхідний. Здебільшого дане поле спрямоване на діагностичні прийоми.

ScheduleSlotId містить в собі ідентифікатор посилання на спеціальну таблицю, де буде вестись облік часу прийому. Коли він був створений, коли його початок і закінчення.

AppointmentNumber – це спеціалізоване поле, яке буде містити в собі спеціальне нумерування прийому. Такий номер міститиме в собі спеціальний код для позначення типу прийому, спеціальної нумерації організації, області, рік створення, а також безпосередньо ідентифікатор самого прийому. Такий номер матиме наступний формат: П-26-2-19-1793.

2.1.4 Проектування сутності медичний документ

Наступна сутність яку необхідно описати – це медичний документ. Хоча здається що робота лікаря це лише взаємодія з людьми – насправді це не так. Найбільшу частину своєї роботи медичний працівник проводить заповнюючи необхідні документи. Тому можна сказати, що поточна частина медичної системи являється однією з найважливіших.

Найменуємо сутність медичних документів як MedDocs і вона матиме наступні поля:

– Id;
– GId;
– OwnerId;
– RegNumber;
– RegDate;
– AppointmentId;
– PatientId;
– PatientCardId;
– EmployeeCardId;
– DtmTemplateCode;
– DtmTemplateName.

Поля Id та GId як і в інших сутностях використовуються в якості ідентифікаторів. Поле Id використовує ітерації як унікальне значення, а поле GId використовує значення типу GUID і генерується випадковим чином. OwnerId в використовується для вказання того, в якій організації був створений новий медичний документ.

Поле RegNumber містить в собі значення унікальне числове занчення і призначено для створення унікального номеру документа. Цей номер буде використовуватись наприклад при інтеграції з іншими медичними системами.

Поле RegDate призначено для збереження дати створення медичного документу.

AppointmentId поле з унікальним ідентифікатором прийому. Поле необхідне для того, щоб зв’язати прийом з медичним документом, таким чином лікар зможе побачити візуальну належність окремого документу до прийому.

PatientId, PatientCardId, EmployeeCardId поля які раніше описувались в інших сутностях і відображають ідентифікатори персони пацієнта, карти пацієнта та співробітника медичного закладу. Насправді це не обов’язкові поля в представленій сутності, так як ми раніше зв’язали її з прийомом і всю необхідну

інформацію можемо дістати звідти. Але на практиці при створенні медичного документу нам далеко не завжди буде потрібна інформація з прийому, як правило це буде просто стандартна форма звіту, без будь-якої специфіки. Але при цьому нам обов’язково потрібно вказати на медичному документі який лікар проводив прийом та який пацієнт звернувся. З точки зору оптимізації внутрішніх процесів майбутнього програмного коду було вирішено додати представлені поля до сутності медичного документу.

DtmTemplateCode поле яке міститиме у собі спеціальний зарезервований код медичного шаблону. Сам медичний шаблон зберігатиметься в медичній системі у вигляді word-документу. За допомогою вказання даного коду буде зрозуміло, який шаблон документу буде необхідний в конкретній ситуації.

При цьому медичний шаблон буде створений таким чином, щоб завантажувати з системи всю необхідну інформацію і таким чином автоматизувати роботу медичного працівника і звільнити його робочий час для більш продуктивної роботи.

DtmTemplateName поле необхідне для збереження назви медичного документу. Саме ця назва буде відображуватись в системі медичним працівникам.

І так в даному параграфі були описані основні сутності медичної системи необхідні для автоматизації роботи медичного закладу. Далі будуть представлені окремі модулі розкладу працівників і прийомів, звітність, діаграми та інше, але це вже в основному програмна реалізація, яка візуально формує необхідну функціональність і окремі сутності для таких реалізацій не потрібні.

2.2 Програмна архітектура медичної системи

При проектуванні великих за об’ємом Веб-додатків прийнято будувати проект з розділенням на клієнтську та серверну частину. Як правило клієнтська частина відповідає за поведінку самого додатку, а також його структурних елементів, в той час коли серверна частина містить в собі бізнес-логіку проекту, займається основними обчисленнями, реалізовує функції інтеграції. Так як

медична інформаційна система являється досить великою за об’ємом і повинна мати здатність до розширення своїх можливостей було вирішено розділити розробку такої системи на клієнтську на серверну частини.

2.2.1 Побудова програмної архітектури клієнтської частини проекту

Так як клієнтська частина буде будуватися на JavaScript фреймворку Angular, то було вирішено використовувати широко відому і рекомендовану практику для побудови Веб-додатків - MVC. Загальний приклад структури MVC для Веб-додатку продемонстровано на рисунку 2.1. Але з врахуванням об’єму проекту було вирішено змінити структурну частину.

При новій архітектурі принцип роботи MVC буде збережено, але при цьому View частина буде розділена на декілька логічних частин, в той же час контролер також буде не один. Різні контролери будуть взаємодіяти з окремими представленнями.

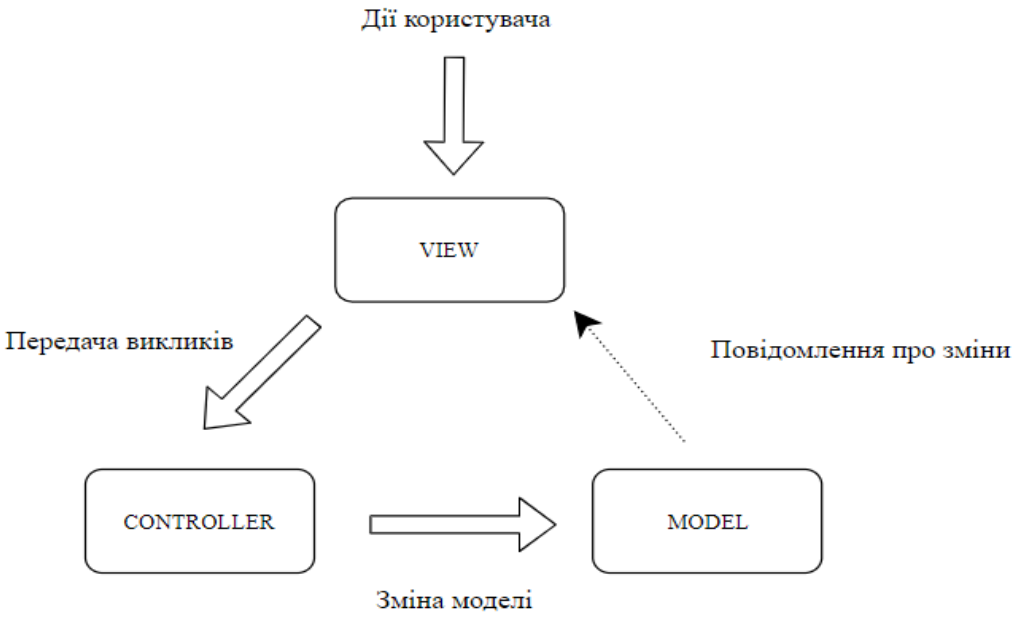

Рисунок 2.1 — Приклад класичної MVC архітектури

Всі представлення та контролери будуть логічно розміщені в одній теці, в той час коли всі моделі будуть винесені в окремий файл.

Представлення в архітектурі будуть роздроблені на чотири основних частини з наступними найменуваннями:

– List;
– View;
– Edit;
– Card.

Зображення такої архітектури можна побачити на рисунку 2.2.

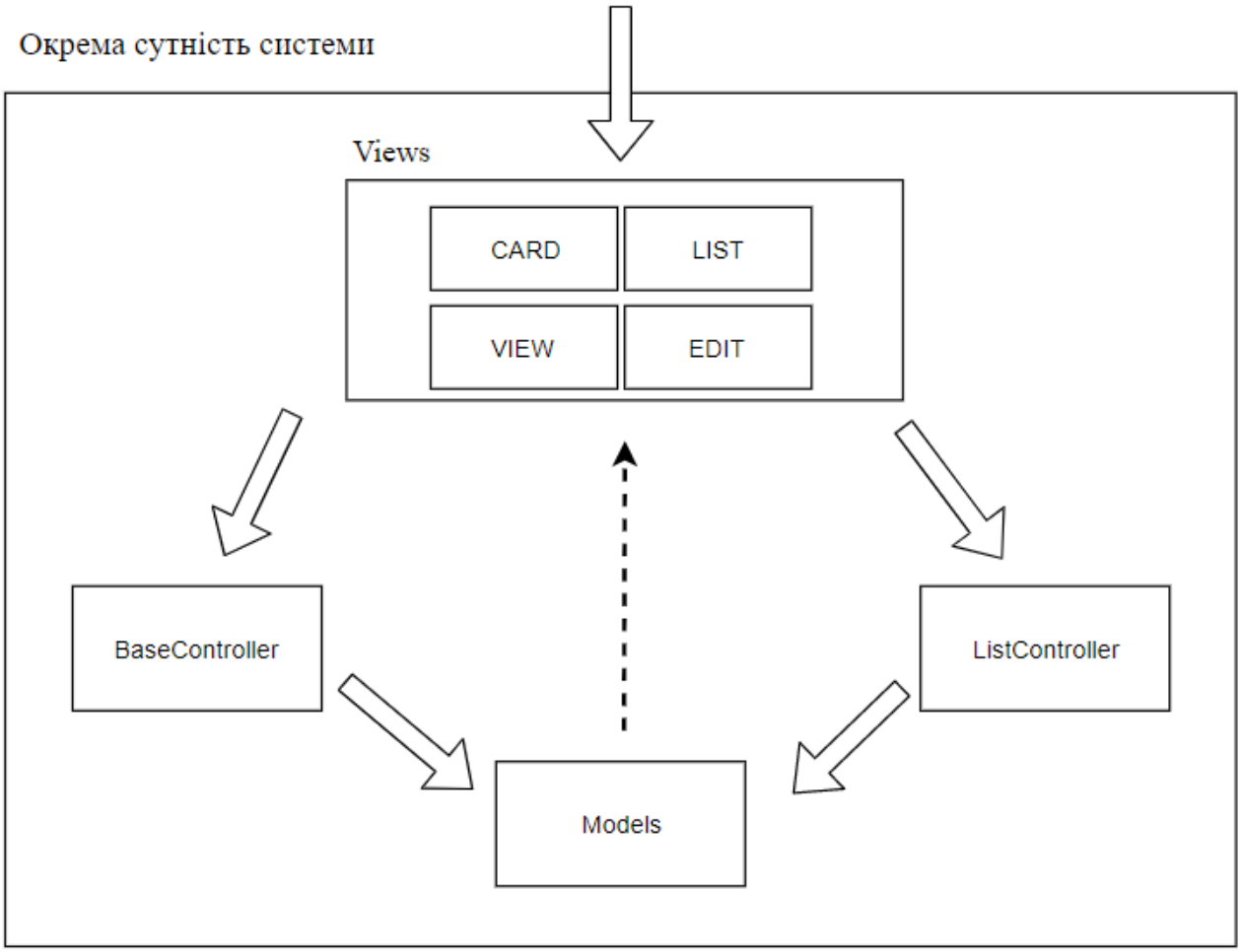

Рисунок 2.2 — Приклад кастомної MVC архітектури

List – представлення для відображення переліку сутності. Для користувача це буде виглядати у виді списку, результат якого можна отримати за допомогою пошуку. Також на даному представленні використовується параметри фільтрації пошуку.

View – базове представлення необхідне для відображення базової інформації про конкретну об’єкт сутності.

Представлення Edit використовується для редагування об’єкту сутності.

Card – це додаткове представлення, яке буде розміщено в правій стороні екрану відносно сутностей View та Edit. На цьому представленні буде розміщуватися додаткова інформація, яка напряму не пов’язана з об’єктом сутності.

Для описаних представлень планується використовувати два контролера. Перший буде обробляти поведінку представлень View, Edit та Card. Другий буде виконувати обробку представлення List.

Як бачимо з рисунку 2.2 принцип MVC архітектури зберігся. В представленому варіанті використовуються Model, View та Controller, але при цьому вони логічно відносяться до окремої сутності і пара контролерів взаємодіють з різними представленнями, а також всі моделі зібрані в одному місці для зручності використання.

2.2.2 Побудова програмної архітектури серверної частини проекту

Для серверної архітектури було вирішено використати класичну трьохрівневу архітектуру, яка має на увазі розділ додатку на три різних рівня. Приклад такої архітектури зображений на рисунку 2.3.

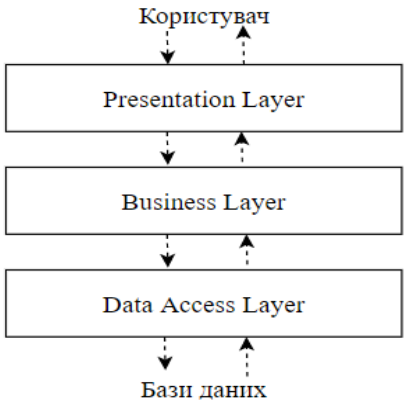

Рисунок 2.3 — Приклад класичної трьохрівневої архітектури

Але в даному випадку рівень представлення у нас виконує клієнтська частина веб-додатку. В якості взаємодії клієнтської і серверної частини використовується рівень контролера на сервері. Насправді це просто представлені посилання які використовуються всередині системи і за допомогою цих посилань взаємодіють дві різні частини проекту.

Рівень Business відповідає за основні обчислення в системі, тобто виконує основну бізнес-логіку проекту.

Рівень Data відповідає за вибірку даних та зв'язок з базами даних.

Приклад реалізованої архітектури представлений на рисунку 2.4.

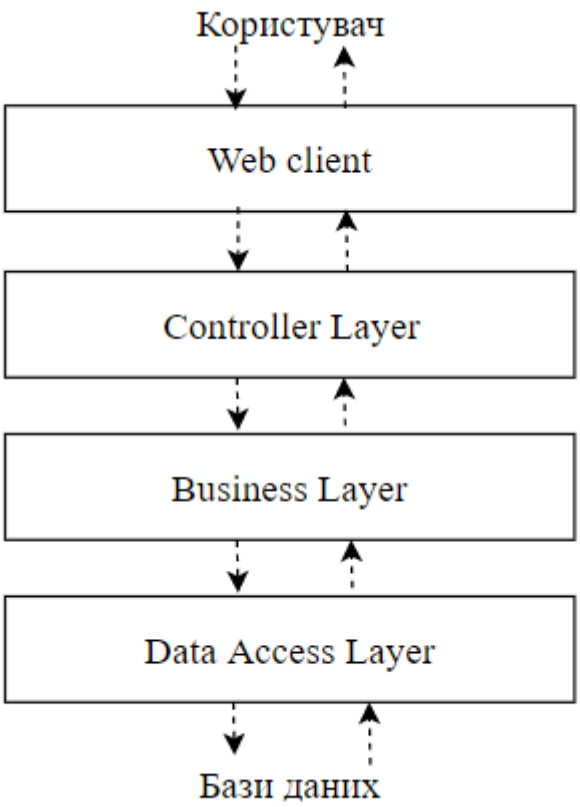

Рисунок 2.4 —Схема реалізованої архітектури

Використання описаної архітектури дозволить використовувати кращі архітектурні практики, але розділить при цьому загальний проект на дві інші. Таким чином буде змога вести розробку клієнтської та серверної частини

незалежно один від одного. Описаний підхід особливо зручний в ситуаціях, коли виникне необхідність підтримки та розвитку проекту, тому що до роботи можна залучити фахівців різного спектру і вести розробку паралельно.

2.3 Реалізація програмних модулів медичної системи

2.3.1 Розробка модулю карти пацієнта та співробітника

Після визначення з основою архітектури проекту перейдемо до розробки окремих модулів. Першим етапом розробки буде створення основних елементів сторінки медичної системи. Як правило це модулі навігації які розміщені на головній сторінці. За їх допомогою буде відбуватися перехід на необхідні частини сайту. Основна сторінка зображена на рисунку 2.5.

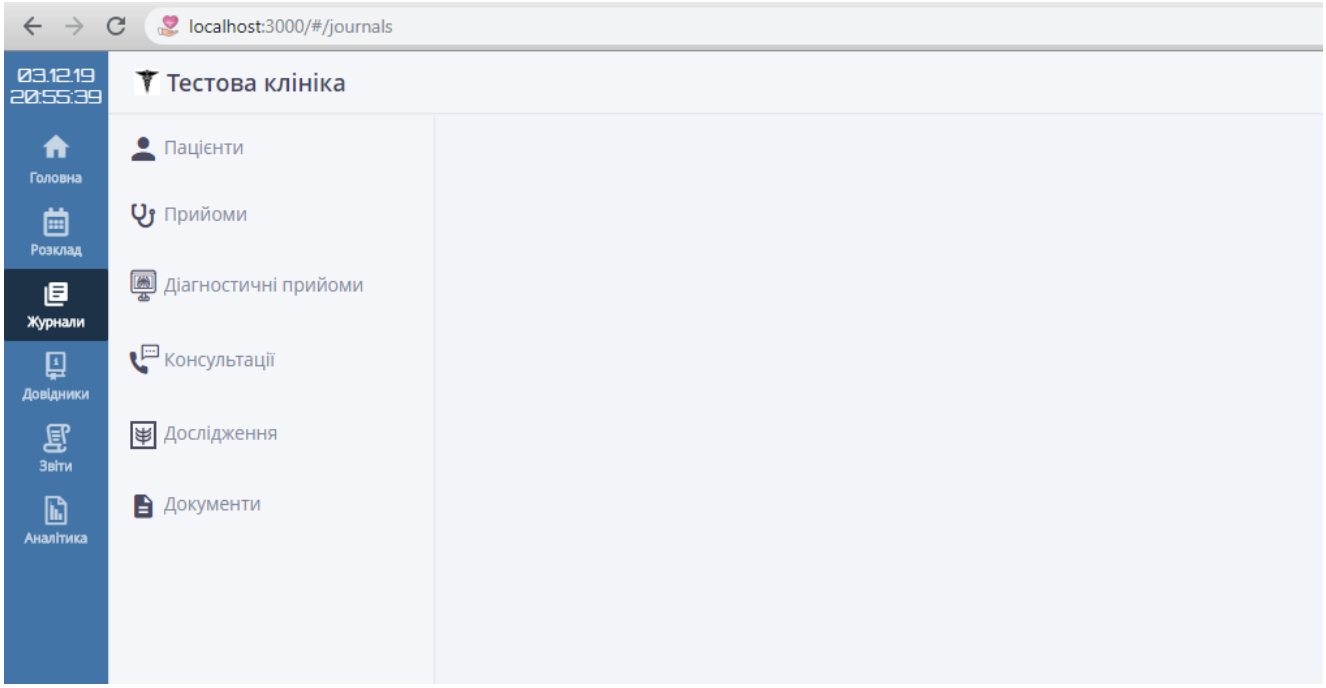

Рисунок 2.5 — Основна сторінка медичної системи

Було вирішено створити статичний елемент з іконками переходів, який буде розміщений в лівій стороні екрану. Елемент навігації матиме два рівні. Перший рівень завжди відображається і на ньому розміщуються основні каталоги системи.

Каталоги системи повинні мати зрозумілі назви та відповідні іконки. Це необхідно для того, щоб користувач системи інтуїтивно розумів, на який каталог йому потрібно натиснути, щоб знайти необхідний модуль системи.

Другий рівень містить в собі конкретні модулі системи по натисканню яких відбувається перехід на нову сторінку. Правила по роботі зі стилістикою відповідають нормам каталогів першого рівня.

Як бачимо з рисунку 2.5 в каталозі «Журнали» знаходяться одні з основних елементів системи, які були описані в розділі проектування сутностей.

Розглянемо реалізацію однієї з основних сутностей «Пацієнт». При натисканні в навігаційному елементі на іконку пацієнта відбувається перехід на сторінку список з існуючими пацієнтами. Сторінка зі списком пацієнтів зображена на рисунку .

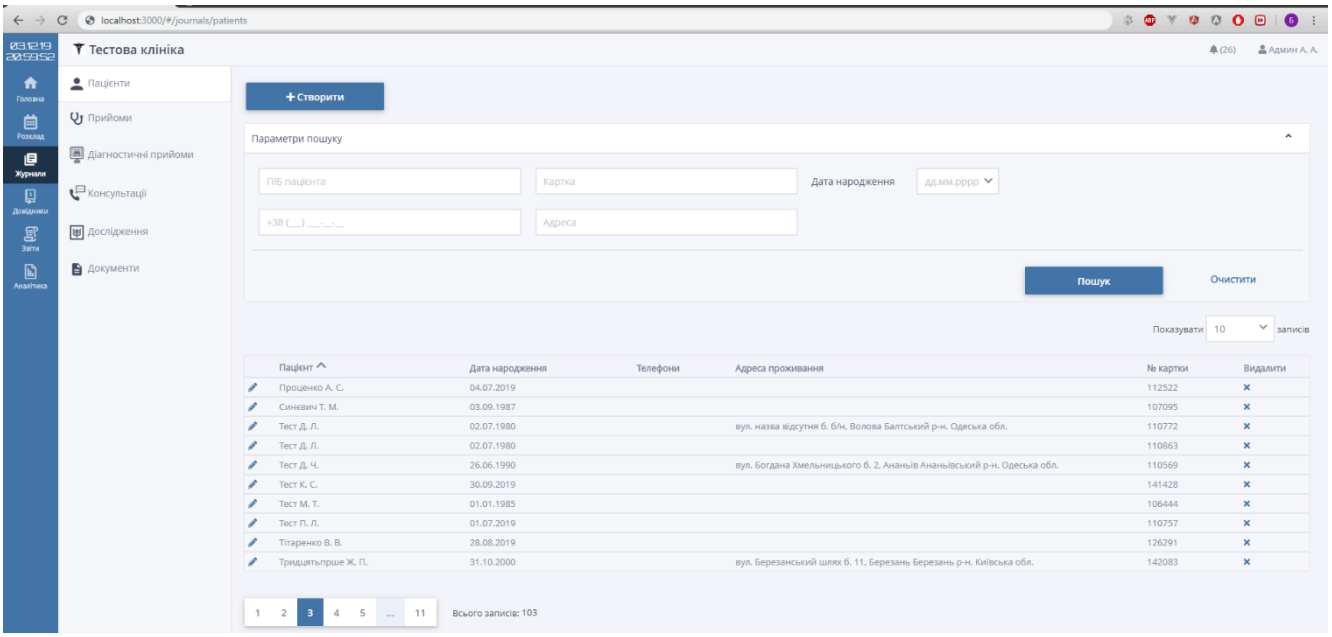

Рисунок 2.6 — Список картки пацієнтів

З точки зору розробки системи, представлення сторінка являється шаблонною сторінкою з назвою list і опрацьовується окремим контролером

Як і описувалось раніше, навіть при відкритті нової картки, ліва частина екрану, де знаходиться навігація, залишається статичною. Звісно при натисканні на

каталоги першого рівня — каталоги другого рівня зникають, залишаючи таким чином більше робочого простору для працівника. В такий спосіб медичний працівник дуже швидко може змінити свій робочий процес в залежності від ситуації.

Сторінку «Пацієнт» умовно можна поділити на декілька частин:

– кнопка «створити»;
– параметри пошуку;
– кнопки «пошук» та «очистити»;
– список пацієнтів;
– нумерація сторінок.

Кнопка «створити» призначена для створення картки нового пацієнта. При її натисканні відбувається перехід на сторінку створення нового об’єкту. Сторінку створення картки «пацієнта» зображено на рисунку 2.7.

Наступний блок називається параметри пошуку. В ньому знаходяться текстові поля для вводу необхідні для створення фільтрації при вибірці даних. В представленій картці знаходиться полів: ПІБ пацієнта, номер картки, дата народження, номер телефону, адреса.

Таким чином, у медичного працівника буде можливість відібрати конкретного пацієнта, якщо відомо хоча б один з представлених показників.

Кнопки пошуку та очистки пов’язані з параметрами пошуку. При заповненні необхідних полів і натисканні кнопки пошук відбувається відображення пацієнтів за заданими критеріями.

За замовчуванням відбувається пошук без параметрів фільтрації. При натисненні кнопки очищення всі параметри фільтрації зачищаються і відбувається новий запит до бази даних без заданих критеріїв.

В низу сторінки знаходиться нумерація сторінок списку. При їх натисненні відбувається новий запит до бази даних з метою відобразити новий список пацієнтів. При цьому можна виставити налаштування по сортуванню списку. Щоб виконати цю дію потрібно натиснути на ім’я колонки зверху списку. При повторному натисненні на кнопку відбувається сортування в зворотному напрямку.

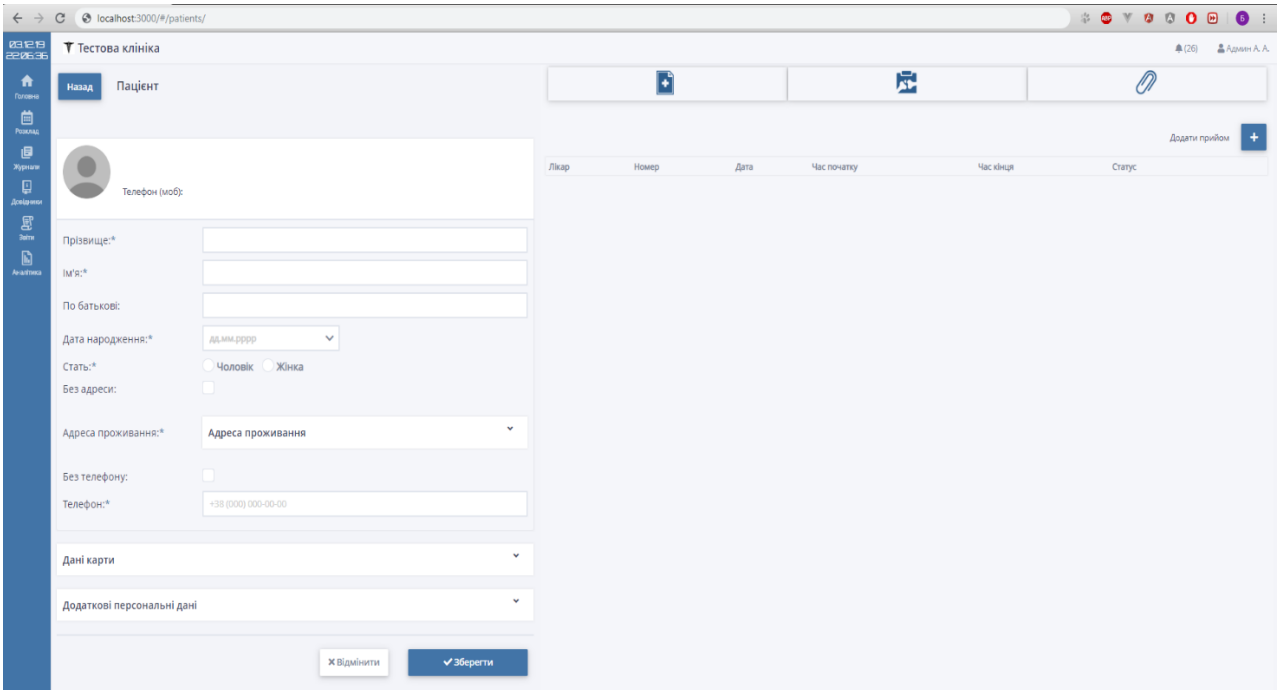

Рисунок 2.7 — Картка створення нового пацієнта

Наступний блок — безпосереднє відображення списку пацієнтів та короткої інформації про них, наприклад прізвище ім’я та по батькові, дата народження, номер телефону, адреса проживання, системний номер картки пацієнта та позначення можливості видалення картки.

Також в списку з початку знаходиться кнопка з іконкою олівця, при натисненні на яку є можливість перейти на сторінку перегляду картки пацієнта. Сторінку перегляду картки пацієнта зображено на рисунку 2.8.

При натисненні кнопки в кінці списку з іконкою хрестовини – відбувається процес видалення запису. Користувач при цьому залишається на сторінці із списком пацієнтів.

Сторінка створення нового пацієнта за контекстом буде подібна сторінці редагування картки. На представленій сторінці знаходяться поля прізвища, ім’я, по батькові, дата народження, стать, адреса, номер телефону та інші додаткові поля.

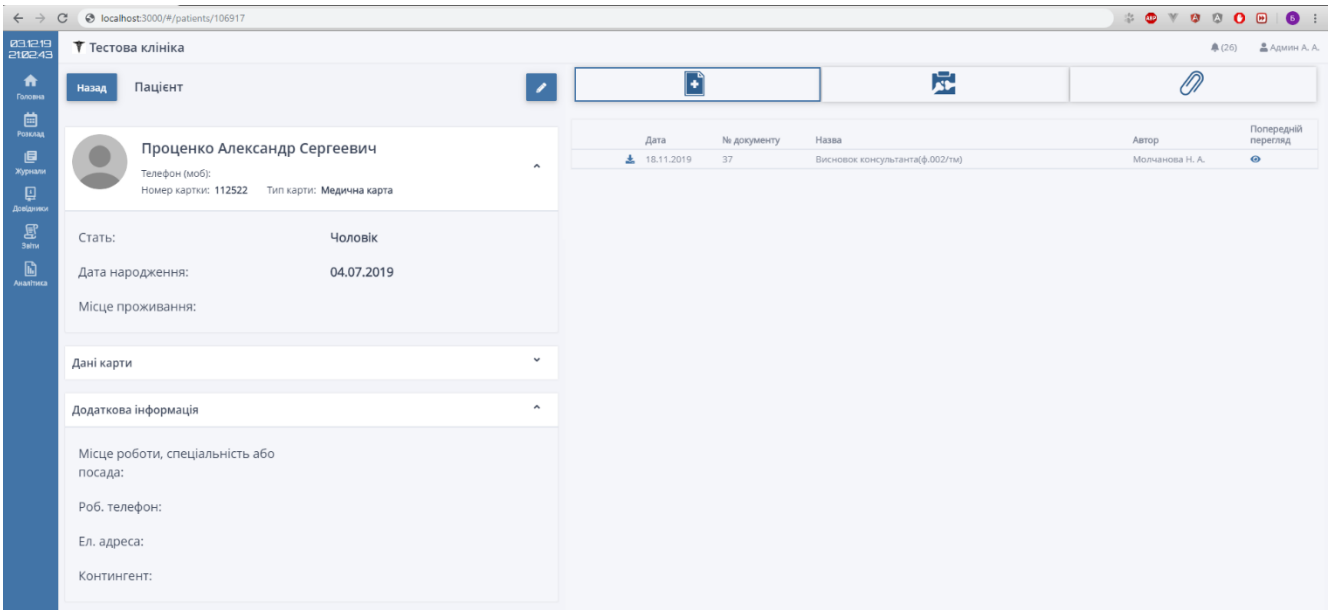

Рисунок 2.8 — Картка перегляду пацієнта

Візуально стає зрозуміло, що при відкритті сторінок перегляду, створення та редагування інформація змінюється лише в лівій частині екрану. Права частина екрану при цьому залишається незмінною і містить в собі три вкладки з різними іконками:

- медичні документи;
- дослідження;
- прикріплені файли.

В той час, коли представлення view, edit та create розміщуються на лівій частині екрану, праву частину завжди займає представлення card, яке відповідає за вміст додаткових вкладок з інформацією, яка прямим чином не належить до основною сутності, але може являтись її необов’язковою частиною.

Таким чином вкладка медичних документів містить в собі перелік створених і доданих медичних документів, які пов’язані з відкритою картою пацієнта.

Вкладка дослідження призначена для розміщення інформації стосовно проведених діагностичних досліджень. Загальним чином тут міститься посилання на файли формату DICOM, що розміщуються на спеціальному сервері файлів. За допомогою цієї вклади буде змога переглянути історію проведених досліджень, рентгенівських, МРТ, КТ знімків.

Вкладка прикріплені файли необхідний для додавання додаткових файлів різних форматів. Це можуть бути додаткові файли форматів pdf, word, jpg, jpeg, png та інші.

Як і описувалось раніше картка редагування схожа за своїм контекстом з карткою створення нового пацієнта і зображена на рисунку 2.9.

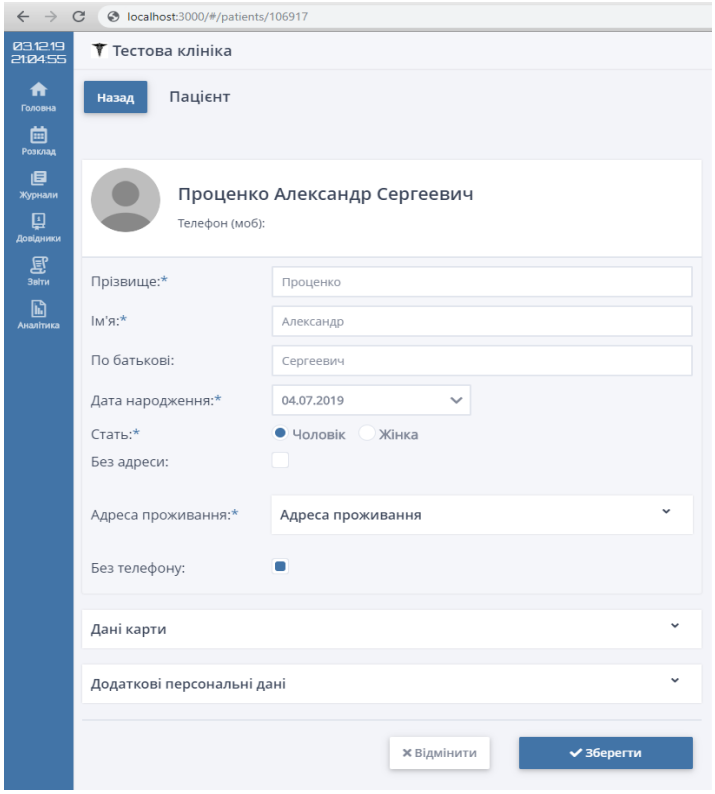

Рисунок 2.9 — Редагування карти пацієнта

На картці редагування медичний працівник має змогу редагувати інформацію стосовно пацієнта. Це важливо в ситуаціях коли інформація про пацієнта змінилася з часом, наприклад у пацієнта змінилось прізвище, або номер телефону. Після внесення змін необхідно натиснути кнопку зберегти, або кнопку відмінити, якщо в збереженні змін немає ніякої необхідності. При натисненні кнопки відмінити відбувається перехід на сторінку перегляду картки.

Картка співробітника наслідує майже всі поля, що й картка пацієнта, тому вони дуже схожі між собою. Відміною між ними можна вважати деякі специфічні

поля за типом спеціалізації та приналежності до підрозділу. В інших аспектах картки схожі за реалізацією.

2.3.2 Розробка модулів організації та підрозділ

Модулі підрозділів та організацій одні з найважливіших в системі, тому що без них немає ніякої можливості автоматизувати роботу медичних закладів. Але при цьому організації та підрозділи не дужа часто змінюються.

Дані про них вводяться як правило лише один раз і в подальшому використовуються як довідникова інформація для відображення, або для створення відповідних взаємозв’язків.

Саме тому ці модулі матимуть мінімум функціональності в медичній системі. Сторінка списку організацій матиме схожий зовнішній вигляд, тому що всі модулі створюються за однією клієнтською архітектурою.

В подальшому всі описані модулі будуть мати схоже відображення. Сторінка списку організацій зображено на рисунку 2.10.

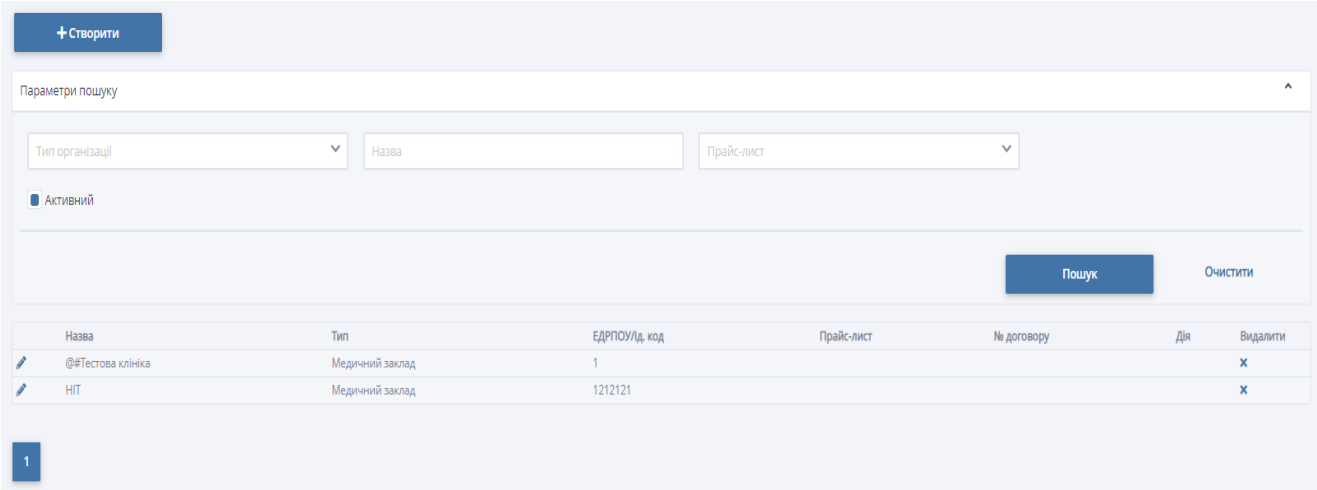

Рисунок 2.10 — Сторінка списку організацій

Як зображено на рисунку 2.10, сторінка відрізняється лише параметрами для сектору параметрів фільтрації, а також колонками відображуваної інформації в списку. Функціональність перегляду, видалення та переходу по сторінкам

залишається незмінним. При натисненні кнопки перегляду відбувається перехід на сторінку перегляду організації, яка зображена на рисунку 2.11.

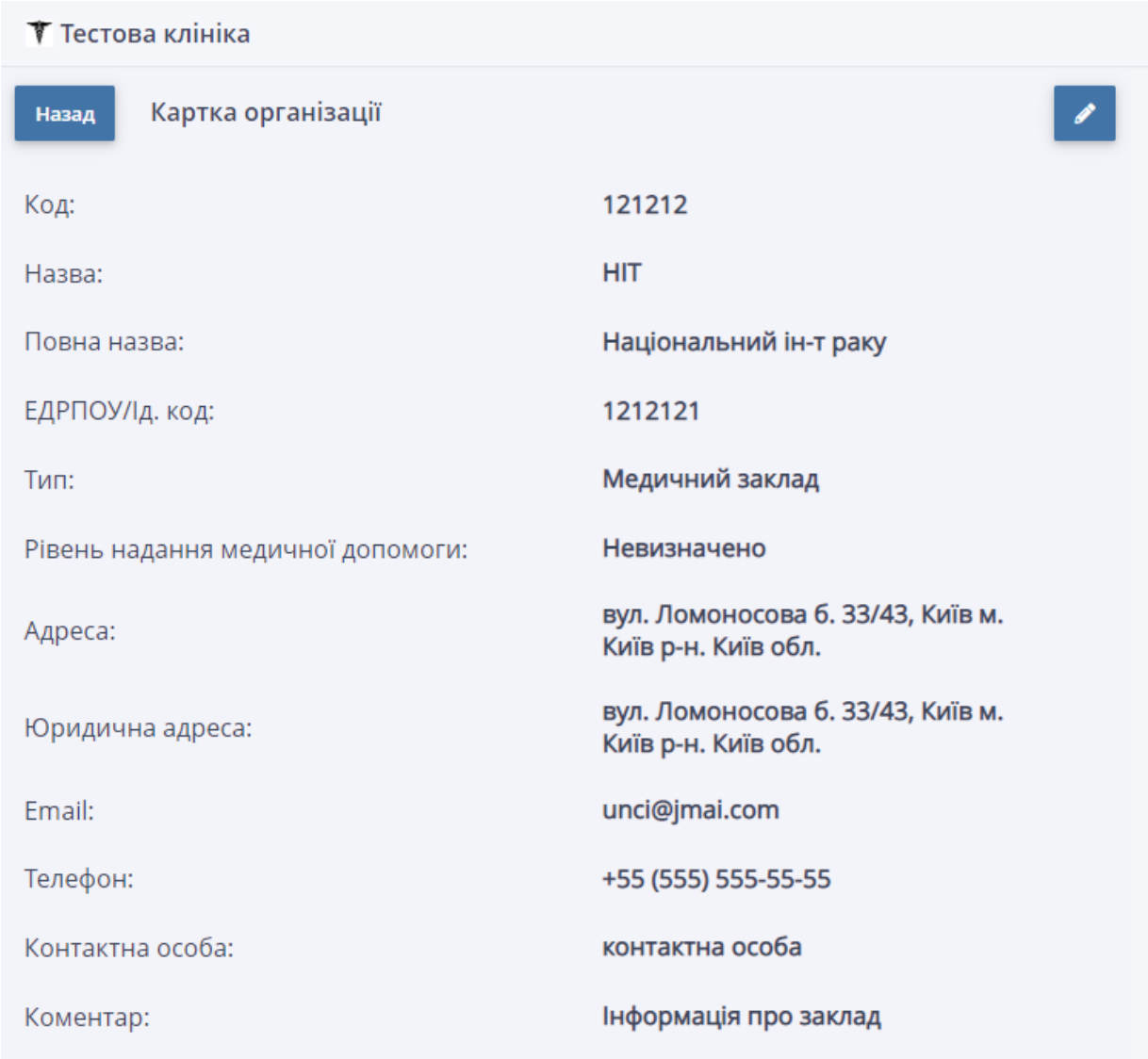

Рисунок 2.11 — Сторінка перегляду організації

На сторінці відображається необхідна інформація про організацію, яка була описана в розділі створення сутностей. При натисненні на кнопку редагування, яка знаходиться в правій верхній стороні екрану, то відбувається перехід на сторінку редагування організації, яка зображена на рисунку 2.12. В картці редагування організації окрім основних полів назви, повної назви та коду, також присутні унікальні поля, наприклад ЕДРПОУ, тип організації, а також можливість завантаження логотипу.

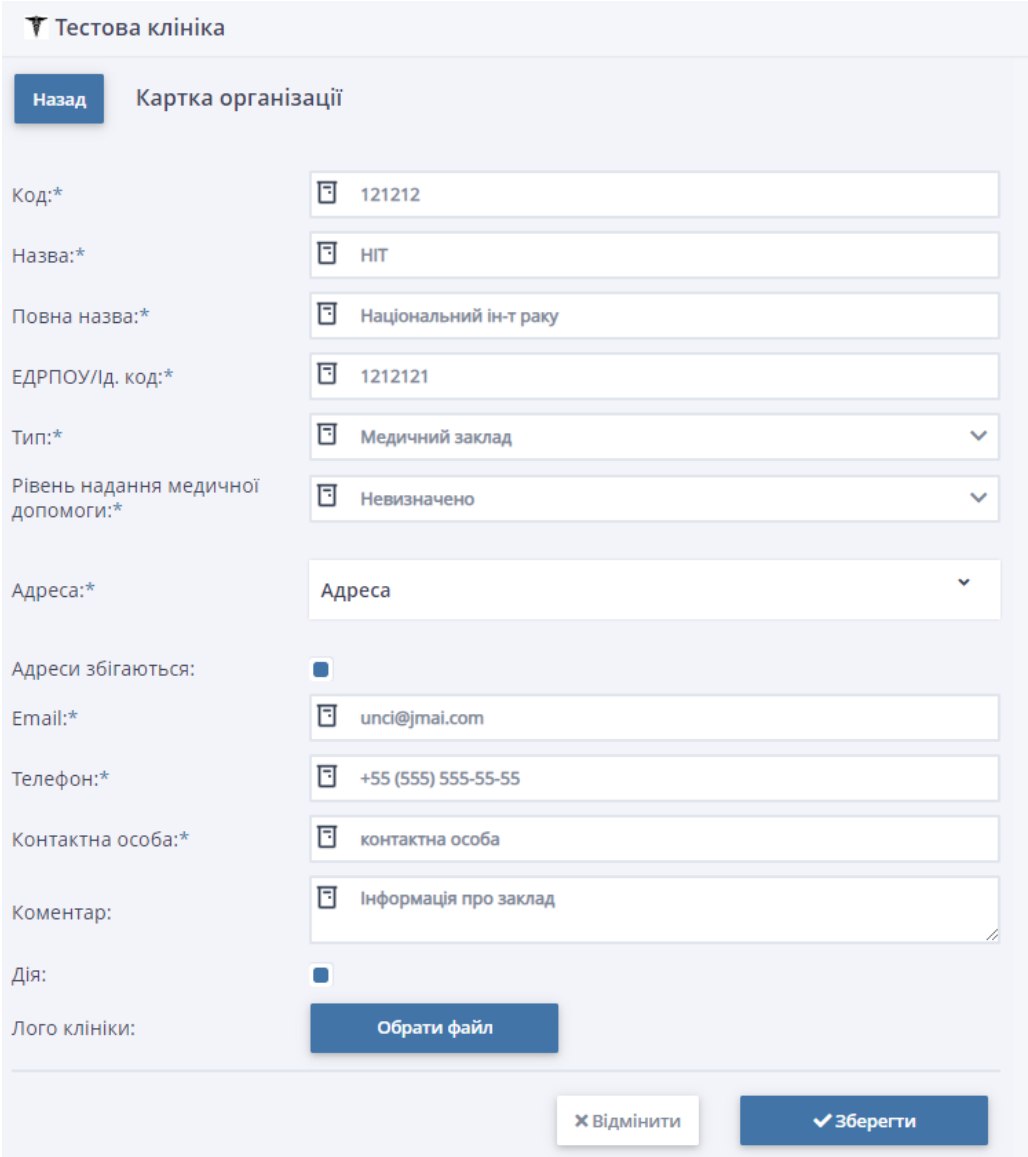

Рисунок 2.12 — Сторінка редагування організації

В цілому процес створення, перегляду, редагування картки підрозділу нічим не відрізняється від картки створення організації. Єдиною зміною можна вважати поля, які відображуються на сторінках і таблиці в базах даних куди буде збережена необхідна інформація.

2.3.3 Розробка модулів прийому та консультації

Після розробки організацій, підрозділів, пацієнтів та співробітників, а також всіх необхідних для цих сутностей довідники, необхідно перейти до процесу, який

пов’язує все раніше перелічене. Основною частиною медичного процесу являється прийом пацієнта у лікаря.

Як правило у закладах без медичних систем в цій ситуації у кожного пацієнта повинна буди паперова медична карта, купа папірців, а у медичного працівника безліч журналів і документів необхідних до заповнення при проведенні прийому. Для автоматизації цих дій було розроблено модуль прийому, який міститиме в собі всю необхідну інформацію про прийом, буде пов’язаний з пацієнтом, співробітником, організацією.

Так як прийомом завідує відповідальний лікар, то саме він буде користувачем системи в представленій ситуації. Перше вікно з яким буд взаємодіяти лікар це список доступних прийомів, який зображений на рисунку 2.13.

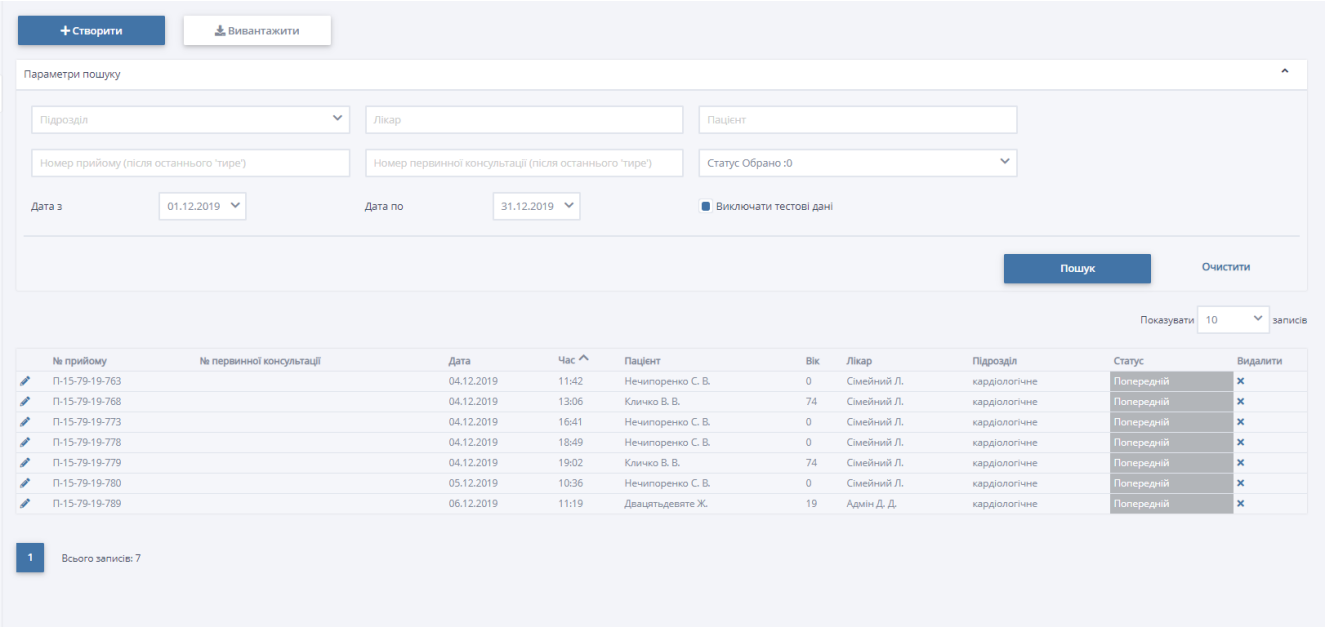

Рисунок 2.13 — Сторінка списку прийомів

На вказаній сторінці маємо стандартний набір елементів для клієнтської частини проекту – кнопки створення та вивантаження списку доступних прийомів, сектор з параметрами фільтрації, кнопки пошуку та очистки полів фільтрації, безпосередньо сам список прийомів та нумерація сторінок списку з переходами.

При натисканні кнопки перегляду прийому відбувається перехід на сторінку перегляду прийому, що зображена на рисунку 2.14.

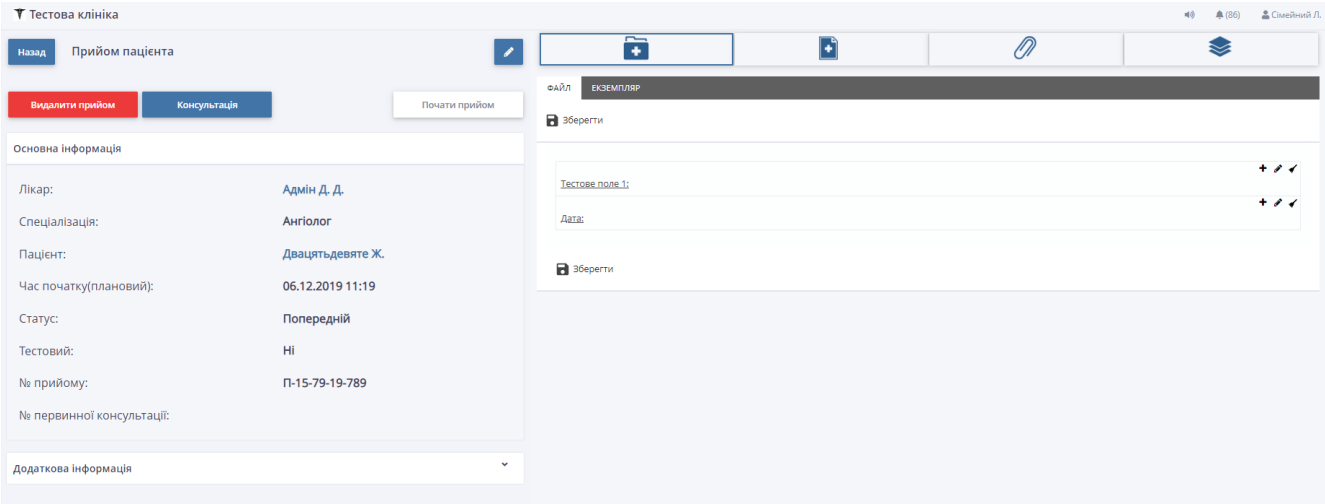

Рисунок 2.14 — Сторінка перегляду прийому

Як і було описано раніше прийом містить в собі інформацію про лікаря, його спеціалізацію, пацієнта що відвідує прийом, час початку прийому, статус прийому та комбінований номер прийому.

Сторінка прийомів має чотири додаткових вкладення — медичні дані, медичні документи, прикріплені файли та зв'язок з консультацією.

Медичні дані – це перша вкладка прийому і вона відрізняється від медичних документів своїм призначенням. За своєю основою медичні дані – це також медичний документ, але він закріплений до конкретного типу прийому і містить в собі базову інформацію яка як правило необхідна для проведення самого прийому або необхідних досліджень.

До цього списку можна віднести групу крові пацієнта, інформацію про проведення загальних аналізів сечі та крові, рівень кретину в крові, загальний вміст білку, рівень отриманого опромінення тощо.

Медичні дані як правило не формуються в окремий документ і зберігається в цифровому вигляді прийому. Формування подібної інформації немає ніякої необхідності, тому що вона, як правило, використовується лише під час проведення прийому.

Вкладка медичних документів містить в собі список створених медичних документів конкретного прийому. Процес створення медичного буде розглянутий в роботі пізніше.

Вкладка прикріплення файлів необхідна для додавання окремих документів в електронній формі, якщо в цьому стоїть потреба.

Остання вкладка прийому називається зв’язок з консультацією. Існує можливість, що прийом не буде створений як одинична сутність. Наприклад якщо прийом проводиться в організації первинної медичної допомоги і у закладу немає кваліфікації провести повноцінний процес, тому що в пацієнта унікальний випадок.

Таким чином організація первинної медичної допомоги може зв’язатись з організацією вторинної медичної допомоги та провести консультації з колегою-спеціалістом. Так як це окремий процес, його також потрібно фіксувати. Саме тому із звичайного прийому може створюватись ще одна сутність консультацій.

Тож саме вкладка зв’язок з консультацією міститиме в собі перелік консультацій проведених для вирішення питань конкретного прийому. При натисненні на консультацію із списку відбувається перехід на сутність консультацію, де продовжується робочий процес.

Якщо натиснути кнопку «почати прийом» статус прийому змінюється з «попередній» на «виконується». В цьому випадку поведінка системи змінюється і вже немає змоги редагувати ключову інформацію стосовно конкретного прийому, наприклад змінити дату лікаря, дату або час.

При натисненні на кнопку редагування і прийому відбувається перехід на сторінку редагування прийому, яка зображена на рисунку 2.15.

Варто відмітити, що навіть при переході на сторінку редагування, в правій частині екрану розміщені таби з інформацією, яка не відноситься до прийомів, але використовується в якості додаткової інформації. Тому при редагуванні прийому є можливість додавати, або редагувати додаткову інформації, наприклад медичні дані, додавати або змінювати медичні документи, додавати або видаляти прикріплені файли, або виконати перехід на пов’язану сутність, якщо в цьому є необхідність.

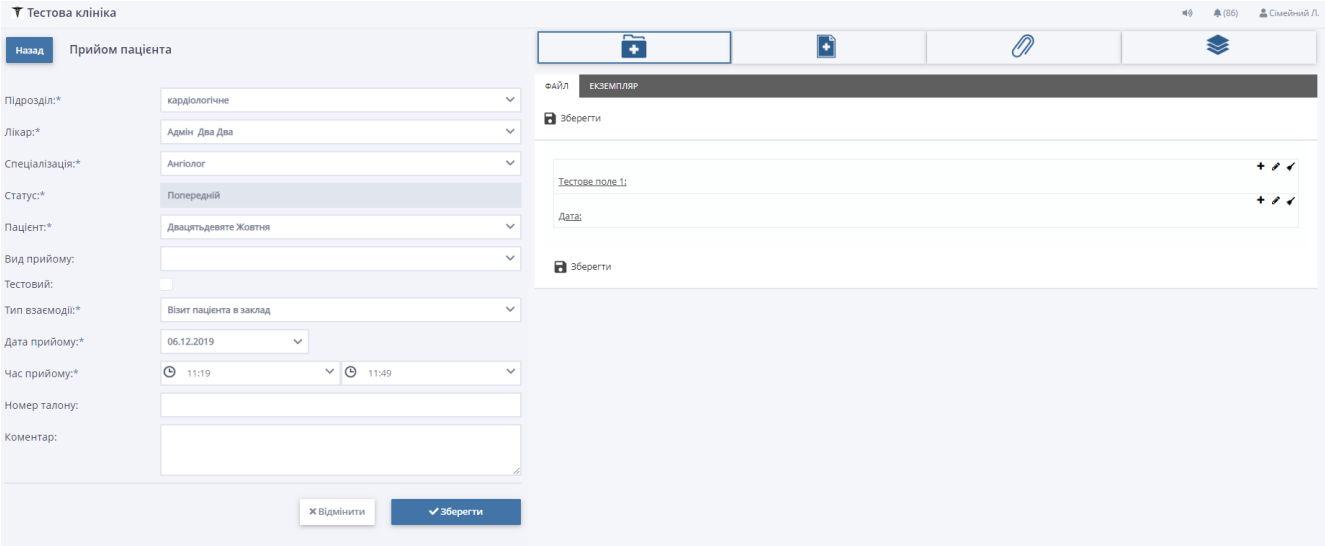

Рисунок 2.15 — Сторінка редагування прийому

Сторінка редагування прийому має базовий вигляд для використовуваної клієнтської архітектури. Основна особливість можливості редагування прийому — це зміна відповідального лікаря, у разі необхідності, зміна часу та дати прийому. Також окрім цього є можливість внесення змін до виконуючого підрозділу, пацієнта, який буде відвідувати прийом, а також типу прийому.

Після внесення всіх необхідних змін потрібно натиснути кнопку зберегти, або кнопки відхилити, якщо у проведених змінах зникла необхідність.

На цьому етапі перейдемо до сутності консультації. Консультація за своєю концепцією вважається прийомом, або його частиною в залежності від ситуації. Як було описано раніше консультація в поточному контексті означає зв’язок B2B.

Перша сторінка консультацій з якою буде взаємодіяти медичний працівник це сторінка зі списком консультацій. Сторінка зі списком має дуже схожий вміст зі сторінкою прийомів, адже вони по суті являються однією сутністю.

При натисненні кнопки перегляду консультації відбувається перехід на сторінку перегляду консультації, що зображена на рисунку 2.16.

Варто відмітити, що на сторінці перегляду консультації поруч з кнопкою редагування присутня кнопка для виконання копії поточної консультації. Це необхідно в ситуаціях, коли консультація проходить по одному і тому ж пацієнту і більшість інформації в консультації є однаковою.

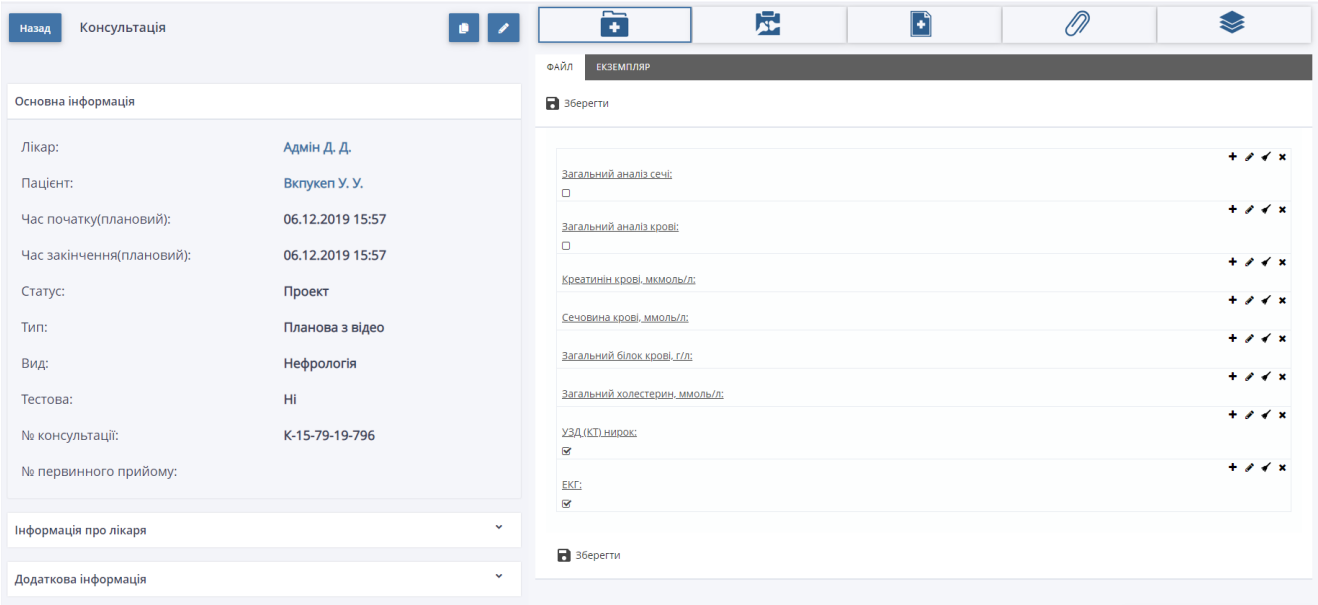

Рисунок 2.16 — Сторінка перегляду консультації

Сторінка перегляду консультацій містить в собі основну інформацію про відповідального лікаря, пацієнта, плановий час початку та закінчення консультації, статус, тип, вид, унікальний згенерований номер консультації та номер первинного прийому, якщо такий існує.

Вкладки консультації збігаються з вкладками прийомів, окрім двох відмінностей. В сутності консультації також є вкладка проведених досліджень. Це необхідно в разі передачі проведених досліджень, або зроблених рентгенівських знімків, що відносяться до пацієнта, адже під час консультації двох лікарів необхідно представити максимально всю необхідну інформацію, щоб вирішити проблему.

Другою відмінністю являється вкладення зв’язок з прийомом. Якщо в прийомі схожа вкладка була призначена для зберігання переліку пов’язаних консультацій, то тут навпаки — зберігається список пов’язаних прийомів з конкретною консультацією. Функціональність відповідає тій, що знаходиться в прийомі — при натисненні на прийом зі списку відбувається перехід на сутність цього прийому.

При натисненні кнопки редагування в консультації відбувається перехід на сторінку редагування, що зображена на рисунку 2.17.

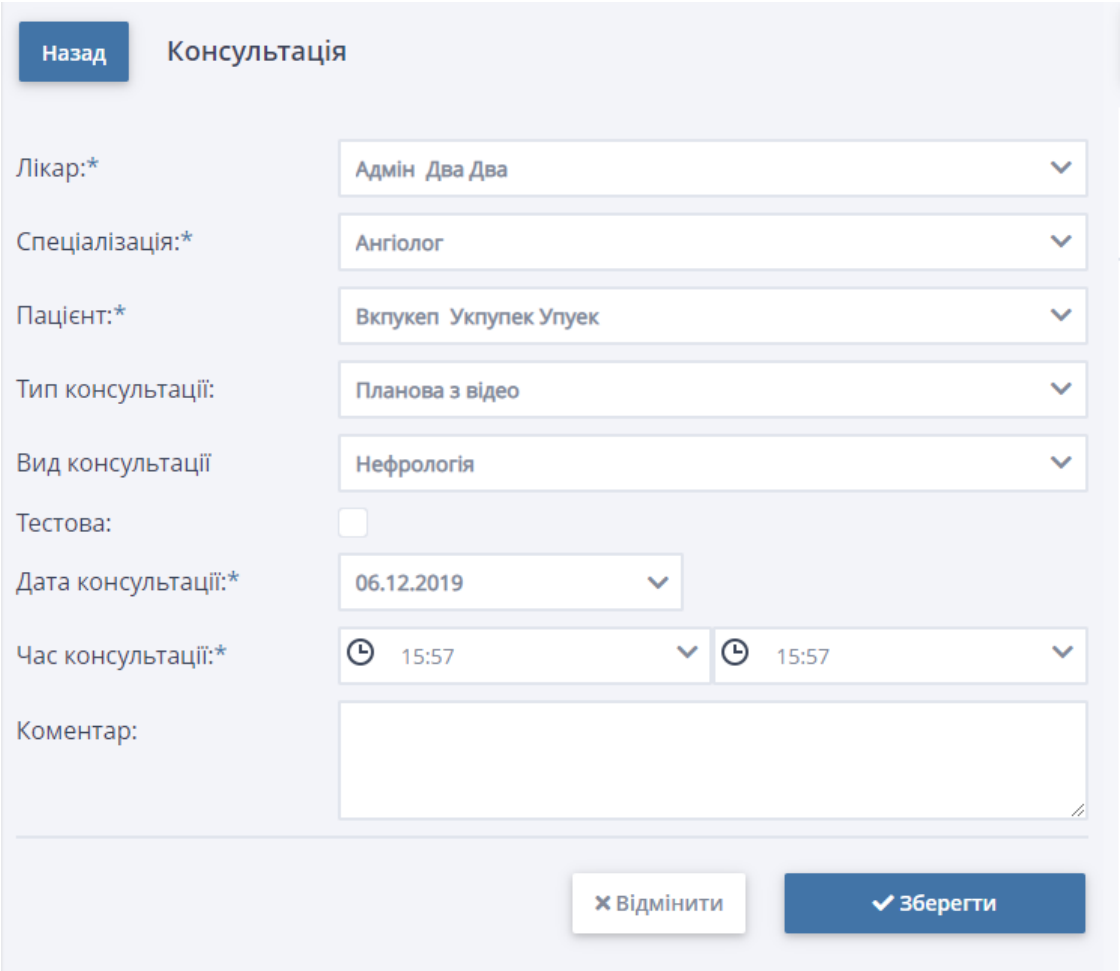

Рисунок 2.17 — Сторінка редагування консультації

Ситуація в даному випадку схожа з прийомом, тут є змога змінити відповідального лікаря, пацієнта, тип та вид консультації, дату та час проведення консультації. Також можна залишити коментар щодо проведеної консультації.

Варто зауважити, хоча існує можливість редагування, здавалось би, ключових елементів прийому та консультації, наприклад дату прийому, або пацієнта, зробити це можна лише в тому випадку, доки прийом або консультація знаходиться в відповідному статусі, наприклад «проект» або «попередній».

Зміна статусу відбувається при натисканні кнопки «консультація» що знаходиться в прийомі. В цей момент створюється сутній консультації, автоматично заповнюється вся можлива інформація. Далі відбувається стандартний процес консультації між двома медичними працівниками.

Після проведення всіх необхідних дій, один із співрозмовників завершує процес натисненням кнопки «завершити». В цей момент інформація фіксується в системі, статус змінюється на «завершено», редагувати таку сутність немає можливості, продовжується процес прийому.

2.3.4 Розробка медичних документів та звітності

В процесі виконання прийому або консультації стає необхідність створення безлічі документів, які відносяться до проведення всіх медичних процесів. Саме тому для автоматизації виконання прийомів та консультацій необхідно автоматичне створення шаблону документу, залишаючи на відповідальність лікаря заповнення лише частини необхідної інформації. Саме для цього була розроблена сутність медичних документів.

При класичному проведенні прийомів, часу на заповнення всіх документів не вистачає і це, як правило, призводить до двох ситуацій — медичний працівник заповнює всі необхідні форми, доки пацієнт просто очікує. В цій ситуації процес заповнення документів являється місцем затримки всього процесу прийому, що впливає також на інших пацієнтів і на весь процес в цілому.

Інший шлях розвитку подій — медичний працівник проводить роботу з документами після прийому, або ще гірше в після робочий час. Дана процедура рутинна, довга, забирає у працівника фізичні та ментальні сили. Варто зауважити, що при цьому інформацію, що необхідно записувати, як правило повторюється із разу в раз, так як більшість прийомів дуже схожі між собою.

Отож під час проведення прийому в медичній системі, якщо у медичного працівника виникає необхідність створення документу, йому необхідно перейти на вкладу медичні документи та натиснути на кнопку додати. Після цього відобразиться список шаблонних документів, які були створені для конкретної організації. Список медичних документів в прийомі зображений на рисунку 2.18.

Варто зауважити, навіть при тому, що документи спеціальних форм затверджені МОЗ України, вони можуть дещо відрізнятися за своїм змістом. Саме

тому для кожної клініки додаються свої окремі шаблони, які можуть змінюватись в рамках дозволеного під потреби організації.

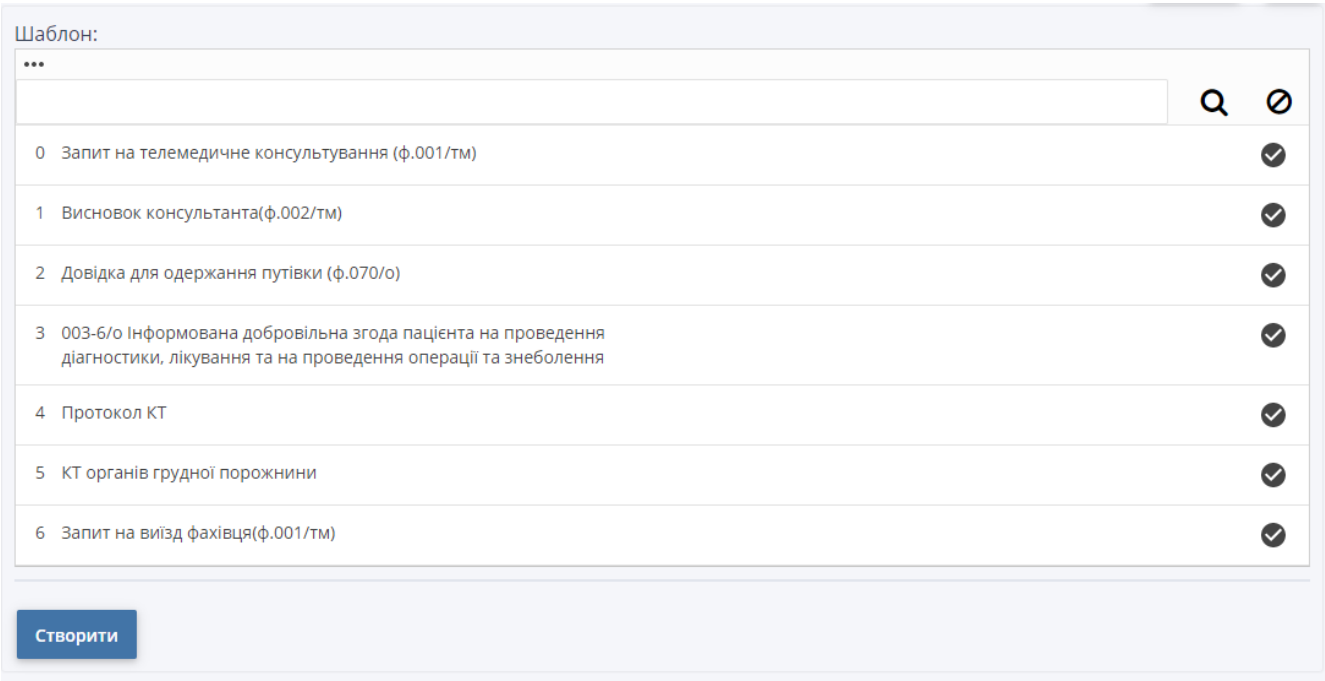

Рисунок 2.18 — Список шаблонів медичних документів

Після вибору медичного документу із списку і натисненні кнопки створити відбувається перехід на сторінку заповнення інформації документу і його перегляд.

Сторінка створення і попередній перегляд документу зображений на рисунку 2.19. З лівої сторони екрану знаходиться перелік шаблонізованих полів, які відносяться до відкритого документу. Дані поля мають різну поведінку і різне призначення. Наприклад поле може текстовим і містити в собі текст відповідно, а може бути булевим і виглядатиме як помітка з галкою. Також є поля додавання дат, секторів та чисел.

Після внесення інформації до полів з лівої частини екрану та натиснення кнопки зберегти, всі дані записуються до бази даних, створюється новий документ а на правій стороні екрану, оновлюється попередній перегляд, який відображає, як буде виглядати документ і яка інформація буде в ньому міститися.

Варто зауважити, що більшість полів в документ записуються автоматично, наприклад прізвище, ім’я, по батькові пацієнта, номер медичної карти пацієнта,

дата його народження, найменування закладу, де проводився прийом і контактна інформація про цей заклад, а також інформація про лікаря, що був відповідальний за проведення прийому.

Вся перелічена інформація міститься в різних сутностях системи і збирається в єдино при формуванні документу.

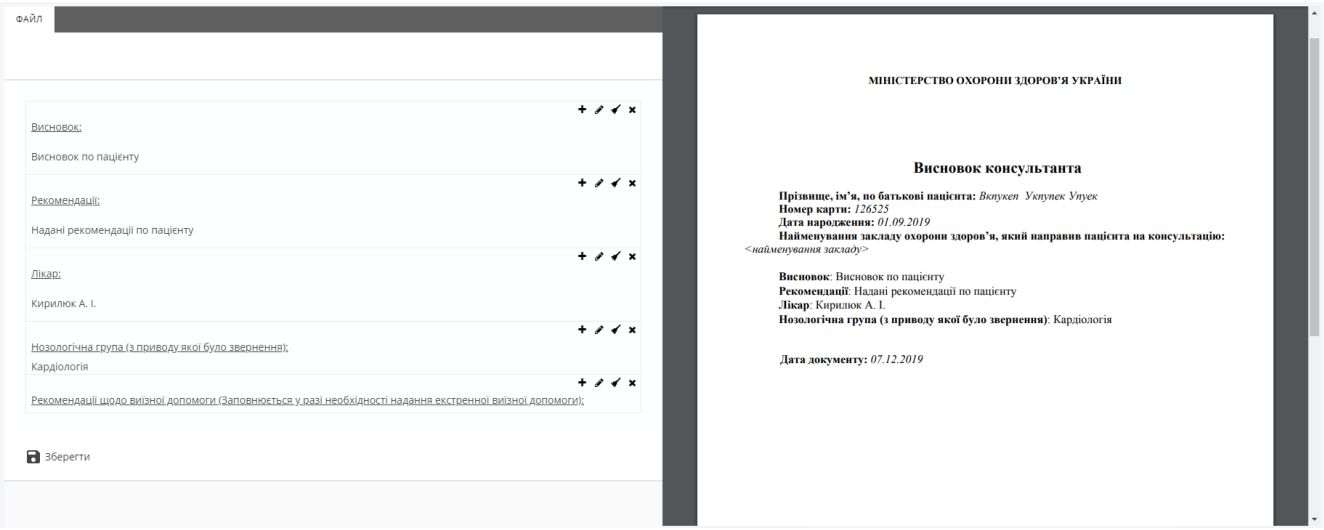

Рисунок 2.19 — Створення медичного документу

Після процесу збереження документу, він відобразиться в списку медичних документів, що належать до прийому, або іншої сутності системи, де він був створений, а також обов'язково в карті пацієнта, стосовно якого документ був створений.

Ще одним способом покращити роботу системи являється створення звітності в організації. Принцип дії полягає в зборі і правильному групуванні інформації. Звіти можуть відрізнятись в залежності від потреб клініки.

Перехід до звітів відбувається з лівої статичної частини екрану, при цьому відкриється сторінка зображена на рисунку 2.20. Основною особливістю звітів являється їх гнучкість в налаштуванні. Необхідно надати можливість корегувати вибірку даних по тому чи іншому звіту, тому було вирішено додати налаштування звітів.

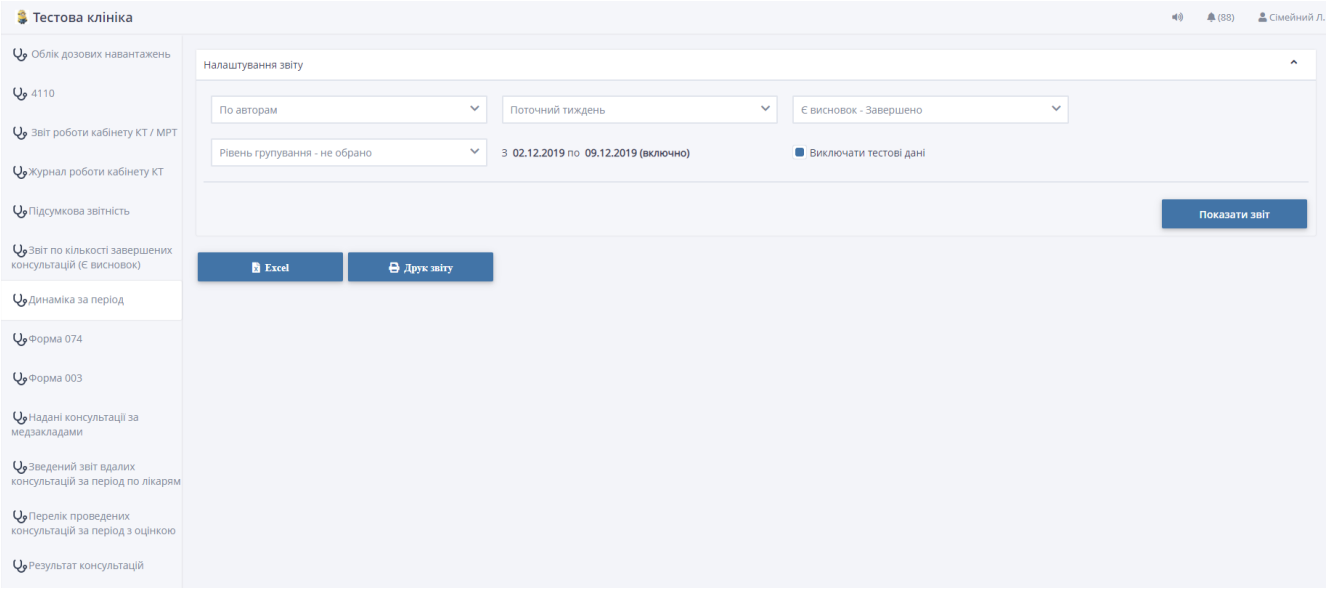

Рисунок 2.20 — перелік звітів та його наповнення

При виборі певних полів в параметрах фільтрації є змога впливати на поточних результат звіту.

При натисненні кнопки «показати звіт» знизу екрану з’явиться перелік відібраних даних. Це необхідно для того, щоб перевірити, чи все влаштовує користувача, перед тим як генерувати сам документ. Це не обов’язкова процедура для виконання, але може бути корисною, при умові що користувач підбирає необхідні налаштування и при цьому не буде завантажувати собі безліч файлів.

При натисненні кнопки Excel згенерується Excel-файл по проставленим параметрам фільтрації. Якщо немає потреби зберігати документ в електронному варіанті, його можна відразу надрукувати натиснувши при цьому кнопку «Друк звіту».

В залежності від обраного звіту із списку, набір полів для фільтрації може змінюватись відповідно від потреб конкретного звіту а також його змісту.

РОЗДІЛ 3. ВДОСКОНАЛЕННЯ МЕДИЧНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ ЗА ДОПОМОГОЮ МАШИННОГО НАВЧАННЯ

3.1 Підготовка концепції для вдосконалення медичної системи

Основною ідеєю вдосконалення системи є перенесення певних рішень з людини, на програмно навчену машину, яка матиме змогу приймати не гірші рішення. Після розглянутих аналогів медичних систем стає зрозуміло, що більшість рішень направлені на організацію внутрішніх медичних процесів та автоматизацію роботи персоналу. Але перед виконанням стандартних процедур закладу стоїть потреба запису на прийом, коли клієнт ще ніяким чином не пов’язаний з клінікою. В таких випадках запис на прийом може відбуватися двома шляхами:

– запис відбувається за допомогою працівника медичного закладу;
– запис відбувається силами самого клієнта.

Перший варіант зводить кількість проблем при запису клієнта до мінімуму, але постає проблема персоналу. Клініка повинна забезпечувати персонал, який буде відповідальний за запис клієнта.

Другий варіант більш привабливий для клініки і відносно простий для клієнта. Для клієнта не складно заповнити відповідну онлайн форму на сайті і дуже часто буває зручніше, ніж телефонний дзвінок і тим більше ніж власне звертання до клініки фізично. При всіх перевагах основною такого підходу є рішення самого клієнта. У більшості відомих систем з онлайн записом клієнт має змогу сам вибрати собі лікаря, до якого бажає піти на прийом, забронювати вільне місце в черзі. При цьому не факт, що клієнту потрібен саме вибраний ним лікар для вирішення його проблеми. Також безвідповідальні користувачі мають змогу забронювати більше вільних місць в черзі, таким чином порушуючи організовану роботу.

Для вирішення двох описаних проблем краще всього підходить спеціально навчена машина, яка мала б змогу приймати рішення не гірші за відповідального

працівника медичного закладу, автоматично виявляти складність причини відвідування клієнта медичного закладу, призначити йому відповідного лікаря, записувати клієнта на вільне місце в розкладі вибраного лікаря. Таким чином клієнту буде достатньо вказати перелік своїх симптомів і на їх основі будуть прийняті відповідні рішення, які задовольняють потреби медичного закладу та клієнта.

Представлений шаблон поведінки знизить затрати клініки на обробку запитів потенціальних пацієнтів, знижуючи при цьому перелік обов'язків з медичного персоналу. Також виключається можливість безглуздого бронювання вільних місць у розкладі медичного закладу, заважаючи там самим інших клієнтам, які дійсно потребують допомоги.

Для клієнта така концепція є більш привабливою, тому що зводить до мінімуму ймовірність неправильного вибору лікаря. Люди не зобов'язані знати всі тонкощі захворювань і їх симптомів, а спираючись на інформацію з вільних ресурсів, щодо їх симптомів, можуть приймати неправильні рішення.

Таким чином розроблене рішення дозволить мінімізувати часові та фінансові затрати для клініки, а також підвищить рівень обслуговування клієнтів.

3.2 Створення набору даних для моделі навчання

3.2.1 Збір та підготовка даних

Після підготовки концепції стає зрозуміло, що задачу можна локалізувати, до створення програмної машини, яка на основі обраних клієнтом симптомів прийматиме коректне рішення стосовно обраного для нього лікаря.

Для проведення машинного навчання необхідно провести ряд процедур [11][12]:

– збір даних;
– підготовка даних;
– побудова моделі;

– аналіз якості.

Існує близько 600 основних симптомів, які описують більшість розповсюджених хвороб і покривають весь спектр захворювань у людини. Для подальшої побудови моделі необхідно врахувати всі основні симптоми, які можуть виникнути у людини при захворюванні[13].

За мету конкретної задачі було поставлено прогнозування відповідного спеціаліста, тому відбираємо також списки спеціалізацій працівників медичних закладів.

Варто відмітити, що напряму лікарі ніяк не пов’язані з симптомами, вони лікують безпосередньо хвороби. Але при певному наборі симптомів, ймовірність наявності певних хвороб більша, ніж наявність інших. В окремих випадках комбінація певних симптомів, зводить ймовірність маже до нуля стосовно певних хвороб.

Тому при побудові моделі враховуємо, що взаємозв’язки між лікарем та симптомами непрямі, але машину необхідно навчити таким чином, щоб вона знаходила правильні зв’язки, інколи навіть для тих випадків, які ми не зможемо заздалегідь передбачити.

Для отримання необхідних даних проведено парсинг відповідних ресурсів [13], де подібна інформація є структурованою і після отримання необхідного набору даних перейдемо до наступного етапу — підготовка даних.

Першим самим головним етапом очистки даних є їх перегляд. Відповідно до кращих практик, було виявлено, що перед очисткою даних перш за все їх потрібно всі переглянути. Чистий набір даних дозволить моделі вивчити значимі ознаки і не дасть змогу перенавчитися по нерелевантному шумі. Тому після парсингу переглядаємо всі дані і оцінюємо їх значимість.

Далі перейдемо до основної процедури підготовки даних:

а) видаляємо всі некоректні символи, розділові знаки, спеціальні символи, які не відносяться до буквених;

б) проводимо токенізацію тексту розділивши його на окремі слова, наприклад певні скорочення;

в) видаляємо нерелевантні слова, які можуть вплинути на якість даних;

г) переводимо всі слова до одного нижнього регістру, для того, щоб слова по типу «Біль», «БІЛЬ», «біль» вважались одним і тим же словом, а не різними;

д) розглядаємо можливість суміщення слів, написаних з помилками, або маючих альтернативну назву;

е) розглядаємо можливість проведення лематизації, тобто приведення різних форм до одного слова.

Після проведення всіх описаних процедур відібрані дані можна вважати чистими і почати побудову моделі для машинного навчання[14]. Набір даних зображений на рисунку 3.1.

```
агрессивность аменорея амнезия анемия анорексия апатия артрит атрофия кожи зуд без сыпи
бели белое пятно на зубе бесплодие у мужчин беспокойство беспокойство, раздражительность
бледность кожных покровов болезненное мочеиспускание болезненность зубаболезненность при
боли в области живота боли в области повреждения боли в суставах боль в анальном отверст
боль в боку боль в веке боль в верхней части живота боль в верхней части спины боль в ви
боль в горле боль в груди боль в груди при вдохе боль в желудке боль в запястье боль в з
боль в зубах боль в икроножных мышцах боль в ключице боль в колене боль в конце мочеиспу
боль в костях боль в малом тазу во время менструаций боль в молочной железе боль в нижне
варикозные узлы и трофические язвы вздутие живота волдыри воспалённые гланды (нёбные мин
двоение перед глазами дезориентация и спутанность сознания депигментация кожи депрессия
загрудинная боль запор заторможенность затруднение носового дыхания затруднения при обще
изменение очертаний пигментного пятна изменение походки изменение речи изменение цвета м
изменение цвета ногтей изменение цвета пигментного пятна изменение цвета стула изменения
кашель кашель с кровью кожный зуд конвульсия красные глаза кровоточивость родинки кровь
мажущие кровянистые выделения между менструациями метеоризм метеочувствительность метрор
набухание и пульсация шейных вен набухшая грудь навязчивые мысли накручивание на палец п
налеты на небных миндалинах налеты творожистого характера на слизистых нарушение аппетит
набухание и пульсация шейных вен набухшая грудь навязчивые мысли накручивание на палец п
налеты на небных миндалинах налеты творожистого характера на слизистых нарушение аппетит
нарушение голоса нарушение зрения нарушение обоняния нарушение памяти нарушение роста на
насморк недержание мочи недоразвитие половых органов недостаточный размер молочных желез
обильное потоотделение обмороки образование язвы в области невуса ограничение подвижност
```

Рисунок 3.1 — Вигляд набору очищених даних для моделі навчання

Розуміється, що деякі симптоми будуть комбінацією декількох слів, тому при формуванні даних, для їх відмінності проміжки між словами одного симптому заповнюються пробілом, в той час, як проміжки між окремими симптомами заповнюються табами.

3.2.2 Формування структури даних для навчання

При побудові моделі для навчання розуміємо, що шуканим значенням при роботі алгоритму буде саме лікар, тому при структуруванні даних значення спеціалізацій буде знаходитись на першому місці.

За правилами використовуваного фреймворку в розроблюваному проекті, дані повинні мати форму [15] зображену в таблиці 3.1.

Таблиця 3.1 — Формат даних для навчання за методом двійкової класифікації

Label	Value
перше значення	0
друге значення	1
третє значення	1
четверте значення	0
п’яте значення	1

Колонка Label є шуканим значенням вибірки, aValue його значенням. В даному випадку продемонстрована одна з найпростіших наборів даних двійкової класифікації, де значення шуканого може бути лише 0 або 1. На документу Label та Value повинні розділятись табами і не містити після себе пустих символів по типу пробілу.

За описаним зразком опишемо перший набір даних для навчання використовуючи підготовлений до цього матеріал. В поточному випадку в якості шуканого значення є іменування спеціаліста, наприклад психіатр, терапевт, онколог і так далі. В якості класифікатора буде використано набір підготовлених симптомів.

Варто розуміти, що одна спеціалізація лікаря може покривати безліч симптомів, а також ці симптоми можуть пересікатись з іншими спеціалізаціями, тому спрощена структура буде виглядати як один до багатьох. Після підготовки файлу за описаною структурою отримаємо перший набір, готовий до навчання.

Фрагмент файлу з даними, що був підготовлений до навчання зображений на таблиці 3.2.

Таблиця 3.2 — Формат даних для навчання формату текст—текст

Label	Value
психотерапевт	агресивність
психотерапевт	аменорея
психотерапевт	анорексія
дерматолог	висип
дерматолог	свербіж
терапевт	біль в горлі

За описаною структурою описається більшість симптомів відповідно до спеціалістів, які ними займаються, але такий опис може мати в собі дві проблеми [16].

Як було описано раніше можливе пересічення спеціальності з окремими симптомами. Наприклад візьмемо останній запис з таблиці 3.2. Терапевт дійсно має відношення до симптому «біль в горлі». Цей симптом один із найрозповсюдженіших і дуже часто виникає при простуді або грипові. Звісно при таких хворобах потрібно відвідувати терапевта. Але виникають ситуації, коли цей симптом виникає при серйозних інфекційних захворюваннях, і краще звернутись до інфекціоніста. Таким чином виникає пересічення багатьох лікарів з одним симптомом і не завжди стає зрозуміло, яке співвідношення буде коректним в даній ситуації.

Другою проблемою є наявність комбінації симптомів. Як правило хвороби мають не один симптом, а декілька. Тому при ситуації, коли клієнт проводить запис на прийом онлайн, то він вибиратиме не один симптом з переліку, а набір симптомів.

Таким чином, машині необхідно аналізувати не одне конкретне значення симптому відповідно до спеціалізації лікаря, а набір симптомів. Згідно з отриманими даними, змінимо наповнення набору даних для навчання. Змінений набір даних зображений на таблиці 3.3.

Таблиця 3.3 — Змінений формат даних для проведення навчання

Label	Value
терапевт	кашель, біль в горлі, нежить, чихання, головна біль
гастроентеролог	дискомфорт шлунку, біль в животі, нудота
невролог	головна біль, нудота, запаморочення, короткочасна втрата пам'яті

Для вирішення описаних раніше проблем було вирішено деталізувати набір даних, з метою підвищити точність майбутнього алгоритму[17]. Як бачимо в таблиці 3.3 на кожну спеціалізацію може приходитись деякий набіл симптомів, що можуть відрізнятися за своєю кількістю. Також тут присутні пересічення однакових симптомів, наприклад для терапевта є симптом «головна біль», як і для невролога. В невролога також є схожий симптом «нудота», який присутній і для гастроентеролога.

Але варто відмітити, що навіть при пересіченні певних симптомів, набір всіх інших симптомів в основному описують окремі хвороби. В першому випадку, для терапевта перелічені основні симптоми звичайної простуди, а для невролога перелічені основні симптоми, які характерні при струсі мозку легкого, або середнього рівня. Хоча в обох випадках зустрічається симптом головної болі, ми даємо зрозуміти машині, що головна біль в поєднанні з кашлем – це скоріше простуда, а головна біль з запамороченням – швидше проблема пов’язана з травмою голови. Будуючи модель таким чином, ми збільшимо ймовірність правильного розрахунку в порівнянні з варіантом опису один спеціаліст – один симптом.

Після детального підбору набору симптомів стосовно одного спеціалісту, буде збільшена точність результатів стосовно відомих даних. Але основною особливістю машинного навчання є здатність робити правильні висновки стосовно тих даних, які ще не були відомі і потрапили до моделі вперше. В представленій ситуації може виникнути дві проблеми:

– набір даних для прогнозу надійде в зміненому порядку;
– надійде набір даних невідомий до цього.

Для вирішення двох описаних проблем, було проведено наступні дії:

– додано всі можливі комбінації симптомів в різних порядках;
– додано додаткову колонку з булевим значенням.

Приклад структури даних має вигляд, що зображений в таблиці 3.4.

Таблиця 3.4 — Кінцевий варіант набору даних для навчання

Label	Value	Truth
терапевт	кашель, біль в горлі, нежить, чихання, головна біль	1
терапевт	біль в горлі, чихання , кашель, головна біль, нежить	
гастроентеролог	дискомфорт шлунку, біль в животі, нудота	1
невролог	головна біль, нудота, запаморочення, короткочасна втрата пам'яті	1
психотерапевт	біль в суглобів, підвищення температури, біль в руці, слабкість	0

При додаванні однакових симптомів в різному порядку, буде виключена можливість неправильного результату в випадку, коли користувач системи ввів неповний набір симптомів, або в іншому від навченої моделі порядку. Наприклад вказання симптомів «головна біль» та «біль в горлі» в різних порядках, давали б різний результат, тому було вирішено створити всі можливі комбінації програмними засобами.

Додавання додаткового поля необхідно для введення критерію правдивості запису. Тобто окрім правильних записів де ми створюємо співвідношення спеціаліст — симптоми, ми також вводимо показник, чи є така комбінація вірною.

Останній запис таблиці 3.4 містить в собі співвідношення симптомів, які ніяким чином не пов’язані з спеціальністю лікаря. Таким чином, задається установка для подальшої моделі навчання, які комбінації не потрібно прогнозувати самостійно, в випаду, коли отриманий набір симптомів не зустрічався для моделі раніше. Таке рішення підвищить результативність правильних рішень.

3.3 Розробка шаблону програми для проведення навчання

Створення моделі машинного навчання не буде відноситись до основного проекту медичної інформаційної системи. Його не обов’язково додавати до основного проекту, на етапі розробки і необхідно це для полегшення змін в програмному коді та тестуванні результатів.

Створимо новий консольний проект на платформі .NET Core у середовищі програмування Visual Studio 2019.

Реалізуємо в створеному проекті два нових класи, які будуть описувати шаблони використовуваних даних. Перший матиме назву ModelInput і міститиме в собі властивості які будуть описувати колонки заготовлених даних:

– Label;
– Value;
– Truth.

Поле Label буде приймати в себе значення спеціалістів, також це поле буде призначено для виводу прогнозованих даних моделлю.

Поле Value міститиме у собі значення наборів симптомів, які відносяться до конкретного спеціаліста.

Поле Truth містить в собі числове значення 0 або 1 і означає чи є вказана комбінація спеціаліст — симптоми правильною. Призначенні 0, ми вказуємо, що

такого значення не існує, при вказанні 1 позначаємо для машини, що така вибірка є правдивою.

Структура ModelInput зображена на рисунку 3.2.

```
using Microsoft.ML.Data;

namespace CreatePredictModel.Model
{
    public class ModelInput
    {

        [ColumnName("Label"), LoadColumn(0)]
        public string Label { get; set; }


        [ColumnName("Value"), LoadColumn(1)]
        public string Theme { get; set; }


        [ColumnName("Truth"), LoadColumn(2)]
        public float Tuth { get; set; }

    }
}
```

Рисунок 3.2 — Модель класу для вхідних значень вибірки

Другою описаною моделлю буде ModelOutput і міститиме модель два значення для виводу результату:

- Prediction;
- Score.

Поле Prediction призначено для виводу текстового значення пропонованого результату прогнозування. Тобто це поле буде містити значення—назву спеціаліста.

Поле Score числове дробове значення яке необхідне для вказання точності результату дослідження. Простіше кажучи поле призначене для виводу відсотку успіху за думкою працюючого алгоритму.

Структура моделі ModelOutput зображена на рисунку 3.3.

```
using System;
using Microsoft.ML.Data;

namespace CreatePredictModel.Model
{
    public class ModelOutput
    {
        [ColumnName("PredictedLabel")]
        public string Prediction { get; set; }
        public float[] Score { get; set; }
    }
}
```

Рисунок 3.3 — Модель класу для вихідних значень вибірки

Після створення моделей для роботи з даними, в класі Program додамо шлях для завантаження даних до проекту з підготовленого файлу. Для цього опишемо наступний код.

Далі в основі проекту створимо ряд методів які необхідні для базової роботи проекту:

- LoadData;
- BuildAndTrainModel;
- Evaluate;
- UseModelWithSingleItem;
- UseModelWithBatchItem.

Метод LoadData необхідний для завантаження підготовлених даних до проекту та приведення їх до спеціальної форми. Створимо новий об’єкт контексту фреймворку ML .Net і передаємо його в якості аргументу до методу LoadData.

В методі LoadData з створеного контексту викликаємо властивість класу DataOperationsCatalog, яке призначене для завантаження і збереження даних і з нього викликаємо метод LoadFromTextFile, який приведений до моделі даних для вивантаження даних з файлів. Отримані дані приводимо до інтерфейсу IDataView і

виконуємо метод TrainTestSplit, який перевіряє правильність вказаних роздільників в файлі і якщо все гаразд, то виконує розділення файлів і приводить їх до відповідного вигляду. Метод LodaData зображений на рисунку 3.4.

```
public static TrainTestData LoadData(MLContext mlContext)
{

    IDataView dataView = mlContext.Data.LoadFromTextFile<SentimentData>(_dataPath, hasHeader: false);

    TrainTestData splitDataView = mlContext.Data.TrainTestSplit(dataView, testFraction: 0.2);


    return splitDataView;

}
```

Рисунок 3.4 — Метод завантаження файлів до проекту.

Після розділення і приведення даних до відповідного типу, виконується повернення готового об’єкту і далі виконується метод BuildAndTrainModel.

BuildAndTrainModel отримує підготовлені дані виконує їх трансформації і викликає метод для навчання моделі. В ході виконання роботи даний метод постійно змінювався, в залежності від виконуваного алгоритму прогнозування.

Тобто для різних випадків і алгоритмів в представленому методі просто змінювався виклик на інший алгоритм побудови моделі, таким чином спрощуючи роботу. Метод побудови та навчання моделі зображений на рисунку 3.5.

```
var trainer = mlContext.MulticlassClassification
              .Trainers.OneVersusAll
              (mlContext.BinaryClassification.Trainers.AveragedPerceptron
                (labelColumnName: "Label", numberOfIterations: 10, featureColumnName: "Features")
              ,labelColumnName: "Label")
              .Append(mlContext.Transforms.Conversion.MapKeyToValue("PredictedLabel", "PredictedLabel"));
var model = estimator.Fit(trainer);
return model;
```

Рисунок 3.5 — Метод побудови та навчання моделі

Для зміни алгоритму необхідно змінити його виклик з властивості Trainers на інший, передаючи при цьому інші параметри, наприклад кількість ітерацій, яку необхідно провести для навчання моделі, поле, яке необхідно спрогнозувати, та які поля будуть виступати в ролі класифікаторів.

Після опису моделі інші методи призначені для приведення даних до правильної форми і виведення результатів навчання до консолі, для оцінки розробником.

Після створення програми шаблону для створення моделі машинного навчання перейдемо до підбору алгоритму для поставленої задачі.

3.4 Підбір алгоритму для вирішення поставленої задачі

Описані набори даних в параграфі 3.2 насправді виникли в процесі постійного створення готових моделей і подальшого їх коригування відносно отриманих даних.

Так як використовується навчання з учителем, а в якості вхідних даних використовуються ознаки, то був використаний підхід опису ознаками [18].

Означимо через X множину спеціалістів, як прецедент предметної області, результатом виміру характеристик якої буде наступна формула 3.1.

$$f: X \rightarrow D_f. \tag{3.1}$$

де D_f — множина допустимих значень ознаки.

Для бінарної ознаки $D_f = \{0, 1\}$.

Першою спробою було використання найбільш поширеного методу бінарної класифікації. Але від такого способу довелось відмовитись, адже результатами прогнозу являється лише відповідь з двома показниками [18]. Якщо проводити побудову моделі на кожен окремий випадок пошуку лікаря по симптомам, то такий підхід міг би спрацювати. Але навчена модель повинна відповідати всьому масиву даних.

Тому наступним рішенням було використання методів багатокласової класифікації, тому що варіацій для симптомів більше двох. Як було означено раніше, відношення спеціаліст—симптом має зв’язок один до багатьох, що в машинному навчанні відповідає методології One-vs-All [19].

Найкращим варіантом в ситуації, що склалася з сформованим набором даних буде використання математичної моделі голосуючого персептрону [20].

Голосуючий персептрон це деяка математична модель «ванільного» персептрону, яка виконується при навчанні з учителем та має свою систему голосів для оцінки побудованої моделі.

Наприклад розглянемо набір даних із 10000 прикладів. Припустимо, що після перших 100 прикладів персептрон виявив хороший класифікатор. Тому він переходить до наступних 9900 прикладів, не вносячи при цьому ніяких оновлені. Він досягає останнього 10000 прикладу і робить помилку і руйнує весь вектор ваги, які добре відпрацював на 99.99% даних [20].

Для вирішення цієї проблеми було введено рішення додавати певні вектори ваги що «виживають» протягом довгого часу, щоб зберегти більше інформації в порівнянні з векторами які швидко скидаються.

Один із способів досягти цього — є шлях голосування. Якщо персептрон виживає протягом 30 прикладів, то він отримує голос 30. Якщо він вижив лише за два приклади, то він отримує лише 2 голоси [20]. Ґрунтуючись на отриманій інформації алгоритм голосуючого персептрону виглядає наступним чином формула 3.2.

$$\gamma = sign\left(\sum_{k=1}^{k} c^{(k)} sign\left(\omega^{(k)} * x + b^{(k)}\right)\right). \quad (3.2)$$

де x – вхідні вектори, w – вага, b- скалярне значення зміщення, с- лічильник успішних ітерацій.

Алгоритм голосуючого персептрону виконує навчання куди краще, від оригінального алгоритму і видає досить точні результати, але містить в собі дуже велику проблему — непрактичність.

Якщо обрати 500 записів під час персептроним навчанням, то одночасно необхідно зберігати 500 векторів ваги. Це потребує просто абсурдної кількості

пам’яті і робить представлений алгоритм в 500 раз повільнішим від оригінального алгоритму для даної ситуації.

Тому було вирішено дещо змінити алгоритм, але зберегти початкову ідею підтримки колекції вагових векторів і часу виживання. Проте під час тестування прогноз здійснюється відповідно до середнього вагового вектору, а не голосування, тому формула 3.3 виглядатиме наступним чином.

$$\gamma = sign\left(\sum_{k=1}^{k} c^{(k)}\left(\omega^{(k)} * x + b^{(k)}\right)\right). \tag{3.3}$$

За основними правилами алгебри змінимо вигляд поточної формули, для виведення часу прогнозу. Формула 3.4 має наступний вигляд.

$$\gamma = sign\left(\left(\sum_{k=1}^{k} c^{(k)}\omega^{(k)}\right)x + \sum_{k=1}^{k} c^{(k)}b^{(k)}\right). \tag{3.4}$$

Після того, як отримали формулу опишемо базовий алгоритм. Алгоритм зображений на рисунку 3.6.

$$
\begin{array}{l}
\omega \leftarrow (o, o, \dots o), \qquad b \leftarrow o \\
\upsilon \leftarrow (o, o, \dots o), \qquad \beta \leftarrow o \\
c \leftarrow 1 \\
for\ iter = \ 1 \dots MaxIter\ do \\
\quad for\ all(x, y) \in D\ do \\
\qquad if\ y(\omega * x + b) \ \leq o\ then \\
\qquad\quad \omega \leftarrow \omega + yx \\
\qquad\quad b \leftarrow b + y \\
\qquad\quad \upsilon \leftarrow \upsilon + ycx \\
\qquad\quad \beta \leftarrow \beta + yc \\
\qquad end\ if \\
\qquad c \leftarrow c + 1 \\
\quad end\ for \\
end\ for \\
return\ \omega - \dfrac{1}{c}\upsilon, b - \dfrac{1}{c}\beta
\end{array}
$$

Рисунок 3.6 — Загальний принцип роботи алгоритму

Після опису базового алгоритму, маємо змогу використати його на мові програмування C# і додати до раніше створеного проекту призначеного для тестування машинного навчання.

Запустимо проект і чекаємо доки завершиться процес навчання. Після цього пробуємо вводити слова симптомів пацієнтів і їх комбінації. Результати проведеного навчання зображені на рисунку 3.7.

```
======Результати методу OVA на основі персептрону=======

агресивність, головна біль, гуд в голові

5 наймовірніших результатів:
психіатр          83%
------------------------
психотерапевт     9%
терапевт          4%
хірург            3%
дерматолог        1%
```

Рисунок 3.7 — Результати побудованої моделі

Отримані результати в середньому становлять близько 85 відсотків точності, що задовольняє поставлену мету. Такий відсоток вибору спеціаліста значно вищий, ніж правильність вибору зроблена безпосередньо клієнтом. Також точність результатів навченої моделі можна збільшити за рахунок введення додаткових комбінацій симптомів і збільшення часу навчання.

ВИСНОВКИ

В ході написання магістерської роботи було вирішено завдання покращення процесу прогнозування спеціальності лікаря, що ґрунтується на наборі симптомів пацієнта. Для цього встановлено та вирішено наступні підзадачі.

По-перше було проведено дослідження основних потреб медичних закладів, які можна вирішити за допомогою технічно-програмних засобів з метою автоматизації роботи.

По-друге було проведено аналіз процесів та принципів роботи медичних закладів, для постановки і реалізації необхідних модулів в сучасній медичній інформаційній системі. Виявлено основні причини, які сповільнюють виконувані медичні процеси і вони, як правило пов'язані з необхідністю виконувати паперову роботу, під час, або після роботи з пацієнтом. Переведення більшості процесів до електронного варіанту дозволить автоматично генерувати необхідні документи та інформацію, знижуючи навантаження на медичного працівника, а також економити час.

По-третє проведено дослідження сучасних медичних інформаційних систем України, вивчено їх сильні та слабкі сторони, що дало змогу зрозуміти в якому напрямку необхідно проводити вдосконалення медичної системи. Найбільш популярні медичні інформаційні системи сьогодення є технологічно застарілими і як правило використовують десктопні додатки, які необхідно впроваджувати на кожну окрему робочу станцію за допомогою технічного спеціаліста. Також в таких системах використовуються інтерфейси старих практик, які не зручні для працівника медичної системи. Підтримка технологій на яких були розроблені такі системи перестають підтримуватись, що ускладнює подальший розвиток таких програмних продуктів.

По-четверте було розроблено веб-додаток медичної інформаційної системи, використовуючи інформацію щодо потреб медичних закладів, а також існуючих аналогів і їх технологічних можливостей.

Медична інформаційна система використовує нові програмні технології і одні з найкращих архітектурних підходів для веб-додатків зазначеного типу. При розробці системи було враховано сильні та слабкі сторони аналогів.

По-п’яте було проведено збір та спеціальна підготовка набору даних для проведення машинного навчання моделі. Було проведено декілька ітерацій підбору необхідної комбінації даних, які при навчанні моделі давали найкращий результат при прогнозуванні результату.

По-шосте був виявлений найбільш оптимальний алгоритм машинного навчання з учителем, для прогнозування лікаря-спеціаліста основуючись на інформації симптомів вказаної клієнтом системи, що записується на прийом. Для вирішення задачі був використаний підхід один до багатьох, в зв’язку з особливістю підготовлених даних для машинного навчання. Описаний підхід проявив найкращі результати для поставленої задачі.

По-сьоме було проведено аналіз використовуваного алгоритму машинного навчання та проведено оптимізацію по його роботі з метою покращити результати прогнозування даних.