Issue 12, 2022

https://doi.org/10.23939/

УДК 004.9

ІНФОРМАЦІЙНА СИСТЕМА ПЕРЕТВОРЕННЯ ЗВУКОВОГО УКРАЇНОМОВНОГО ТЕКСТУ НА ПИСЬМОВИЙ НА ОСНОВІ МЕТОДІВ NLP ТА МАШИННОГО НАВЧАННЯ

Юрій Тищук[1], Вікторія Висоцька[1, 2], Ольга Власенко[2, 3]

[1] Національний університет "Львівська політехніка", кафедра інформаційних систем та мереж, Україна, Львів, вул. С. Бандери, 12
[2] Університет Оснабрюка, Інститут комп'ютерних наук, Німеччина, Оснабрюк, вул. Фрідріха Янсена, 1
[3] Житомирський держаний університет імені Івана Франка, кафедра професійно-педагогічної, спеціальної освіти, андрагогіки та управління, Україна, Житомир, вул. Велика Бердичівська, 40
E-mail: yurii.tyshchuk.knm.2018@lpnu.ua, ORCID: 0000-0002-1083-3765
E-mail: Victoria.A.Vysotska@lpnu.ua, ORCID: 0000-0001-6417-3689
E-mail: olha.vlasenko@uni-osnabrueck.de, ORCID: 0000-0001-7258-2108

Розпізнавання мовлення передбачає різні моделі, методи та алгоритми аналізу та опрацювання записаного голосу користувача. Завдяки цьому люди можуть керувати різними системами, які підтримують один із видів розпізнавання мовлення. Система перетворення мовлення на текст є одним із видів розпізнавання мовлення, що використовує розмовні дані для подальшого їх опрацювання. Також передбачено декілька етапів для опрацювання аудіофайла, під час якого використовують електроакустичні засоби, алгоритми фільтрації в аудіофайлі для виокремлення релевантних звуків, електронні масиви даних для вибраної мови, а також математичні моделі, які складають із фонем найімовірніші слова. Завдяки перетворенню мовлення на текст істотно пришвидшується та полегшується робота, а також знижується рівень стресу в людей, професії яких тісно пов'язані із набиранням великих текстів на клавіатурі. Окрім цього, такі системи допомагають бізнесу, адже концепція віддаленої праці стає все популярнішою, а отже, компанії потребують інструментів для запису та систематизації нарад у вигляді письмового тексту. Об'єктом дослідження є процес перетворення україномовного тексту на письмовий на основі методів NLP та машинного навчання. Предмет дослідження – алгоритми опрацювання файлів для виокремлення релевантних звуків та розпізнавання фонем, а також математичні моделі для розпізнавання масиву фонем як конкретних слів. Метою виконання роботи є проєктування та розроблення інформаційної системи для перетворення звукового україномовного тексту на письмовий на основі Web-додатка Ukrainian Speech-to-text, який є технологією для точного та легкого аналізу україномовних аудіофайлів та подальшої їх транскрипції у текст. Застосунок підтримує завантаження файлів із файлової системи та запис, із використанням мікрофона, а також збереженням проаналізованих даних. Також у статті описано етапи проєктування та загальну типову архітектуру розробленої системи перетворення звукового україномовного тексту на письмовий. Як свідчать результати експериментальної апробації розробленої системи, кількість слів ніяк не впливає на точність алгоритму перетворення, а зменшення відсотка невелике і спричинене складністю слів та низькою якістю мікрофона, а отже, і записаного аудіофайла.

Ключові слова: мовлення на текст; розпізнавання мовлення; україномовні тексти; інформаційна система.

Вступ

З розвитком інформаційних технологій (ІТ), покращенням алгоритмів та збільшенням обсягів роботи за комп'ютером у суспільства з'являється потреба у швидкому та доступному методі голосового управління чи перетворення записаного голосу на письмовий текст [1–10]. Програми, які реалізують такі алгоритми, надзвичайно корисні для людей похилого віку та людей із обмеженими можливостями [11–18], а також для тих, кому доводиться друкувати велику кількість текстів або опрацьовувати нетиповий текст класичною українською мовою, наприклад, на діалекті [19–35]. Мета виконання роботи – проєктування та розроблення інформаційної системи (ІС) для забезпечення перетворення звукового україномовного тексту на письмовий. Для цього необхідно вирішити такі завдання:

· Дослідити предметну область (ПО), інструменти, призначення, алгоритми, готові рішення та визначити підхід до реалізації ІС розпізнавання звукового україномовного тексту.

· Виконати системний аналіз цілі, побудувати дерево цілей, матрицю попарних порівнянь, контекстну діаграму та її декомпозицію, а також ієрархію цілей.

· Розглянути та вибрати ІТ і програмні засоби (ПЗ) для реалізації ІС.

· Реалізувати інформаційну систему перетворення звукового україномовного тексту на письмовий, використавши вибрані технології.

Об'єкт дослідження – процес перетворення звукового україномовного тексту на письмовий. *Предметом дослідження* є методи та алгоритми, що використовують під час транскрипції аудіо в текст. *Наукова новизна дослідження* полягає у розробленні інформаційної технології перетворення звукового україномовного тексту на письмовий. *Практична значущість дослідження* полягає у розробленні алгоритму перетворення звукового україномовного тексту на письмовий та експериментальній апробації під час тестування розробленої ІС для визначення проблемних місць та аспектів функціонування таких систем.

Аналіз останніх досліджень та публікацій

Мова – це найпотужніший засіб спілкування, за допомогою якого люди висловлюють думки і почуття. Особливості мовлення розрізняються у кожній природній мові. Однак навіть у разі спілкування однією мовою темп і діалект у кожної людини різний. Це створює проблеми в розумінні та інтерпретації переданого повідомлення. Інколи довгі промови важко зрозуміти з-за різної вимови, темпу, тембру голосу, гучності тощо. Розпізнавання мови – міждисциплінарна галузь комп'ютерної лінгвістики, яка допомагає в розробленні ІТ/ІС розпізнавання мовлення та переведення у текстовий контент на основі методів speech-to-text. Ці методи відрізняються від просто розпізнавання голосу тим, що ПЗ навчається розуміти і розпізнавати вимовлені слова. ПЗ для розпізнавання голосу концентрується на визначенні голосових моделей індивідуальних людей. Для функціонування ІС перетворення мовлення на текст (ІСПМТ) необхідне поєднання спеціально навчених алгоритмів, комп'ютерних процесорів і апаратури захоплення звуку (мікрофонів). Навчені алгоритми розкладають безперервний складний акустичний сигнал на дискретні мовні одиниці (фонеми). Це найменша окрема звукова одиниця, на які можна розділити людське мовлення [36]. Це також мінімальні звукові одиниці, які носії мови можуть сприймати як різні, щоб створювати значущі відмінності між словами. Наприклад, носії англійської мови розуміють, що "though" і "go" – це два різних слова, оскільки їх перший приголосний звук відрізняється, хоча голосні звуки однакові. У мові може бути більше/менше фонем, ніж літер/графем. Наприклад, в англійській мові лише 26 літер, хоча деякі діалекти містять 44 різні фонеми. Ще більше ускладнює ситуацію те, що акустичні властивості певної фонеми залежать від мовця і контексту, в якому вона звучить. Наприклад, звук "l" у кінці слова "ball" акустично ближчий до голосного звуку "o", ніж до звука "l" на початку слова "loud" у

багатьох діалектах англійської мови. Алгоритми, що розділяють акустичні сигнали на фонеми, повинні враховувати контекст. Робочий процес ІСПМТ на основі AI складається із таких етапів [37]:

1. Звуки людини через мікрофон перетворюються із аналогових сигналів на цифрові.
2. ПЗ аналізує біти аудіофайла до сотих часток секунди, відшукуючи відомі фонеми.
3. Ідентифіковані фонеми порівнюють зі словниками БД поширених слів, фраз і речень.
4. ПЗ використовує складні математичні моделі для визначення найімовірніших слів, з яких і складає текст.

Сьогодні будь-яка людина зі смартфоном, підключеним до інтернету, має доступ до різних ІСПМТ. Потреба в ІСПМТ на основі AI speech-to-text різко зросла на корпоративному/споживчому IT-ринках [1–38]:

1. *Обслуговування клієнтів.* Багато підприємств покладаються на чат-боти або AI-застосунки в обслуговуванні клієнтів, наприклад, для зниження витрат і поліпшення якості обслуговування. Оскільки багато користувачів вважають голосовий чат кращим та комфортнішим, ефективне і точне функціонування ІСПМТ радикально поліпшує якість обслуговування клієнтів у інтернеті. Чат-боти AI із розширеними можливостями розпізнавання мови знижують навантаження на Call-центр. Вони працюють як перша лінія обслуговування, визначають наміри/потреби співрозмовника і перенаправляють на відповідну службу/ресурс.
2. *Пошук контенту.* Дедалі ширше використання мобільних пристроїв сприяє збільшенню попиту на ІСПМТ. Кількість потенційних користувачів різко зросла завдяки загальнодоступності ІСПМТ, доступ до яких безкоштовно надається на платформах IOS/Android.
3. *Електронна документація.* Існує безліч послуг і ПО, де ІСПМТ полегшують ведення документації. Наприклад, лікарям вона необхідна для швидшого й ефективнішого ведення медичних карт пацієнтів і записів діагнозів. Судові системи і державні установи можуть використовувати ІСПМТ для зниження витрат і підвищення ефективності ведення документації. Підприємства також можуть використовувати її під час важливих зустрічей і конференцій для ведення протоколів та інших спеціальних потреб. Пандемія Covid-19 виявила новий варіант використання ІСПМТ. Ураховуючи величезну кількість віддалених нарад і відео конференцій, функція перетворення аудіо на текст дає компаніям змогу підводити підсумки нарад, отримувати інформацію й аналітичні дані, записуючи розмови.
4. *Споживання контенту.* Глобальна доступність контенту є величезним стимулом для упровадження ІСПМТ. Оскільки потокове онлайн-мовлення витісняє традиційні форми розваг, попит на цифрові субтитри постійно зростає. Величезний ринок субтитрів у реальному часі, оскільки контент транслюється по всьому світу для глядачів з різним мовним рівнем. Існує величезний потенціал для використання ІСПМТ і в живих розвагах, таких як спортивні трансляції. Коментарі із миттєвими титрами можуть стати революційним рішенням, підвищивши доступність і загальну залученість користувачів.
5. *Дистанційна освіта*, зокрема для людей з особливими потребами.

Навіть найкращі ІСПМТ не можуть досягти 100 % точності. Нині це тільки 95 %, цього показника вперше досяг сервіс Google Cloud Speech в 2017 р. Численні фактори спричиняють виникнення цього 5 % рівня помилок найкращих в світі ІСПМТ.

· Акценти і діалекти – навіть звичайні люди часто не можуть зрозуміти, що хтось говорить на їх рідній мові, через місцеві діалекти та акценти. Програмування AI для виявлення усіх цих нюансів потребує часу і є дуже складним завданням.

· Контекст – омофони – це слова, які мають однакові або подібні звуки, але різні значення. Простий приклад – “гостиний” і “гостинний”. AI часто важко визначити ці омофони в реченні без навчання на цих словах у відповідних контекстах.

· Якість введення – фонові шуми істотно впливають на здатність AI точно перетворити мовлення на текст. Якщо користувач страждає від захворювання, наприклад, застуди або болю в

горлі, зміни, які вони вносять у мову, часто спричиняють у ІСПМТ похибку розпізнавання.

· Візуальні сигнали – для передавання повідомлення люди покладаються не тільки на голос, слова часто доповнюються мімікою і жестами, які підсилюють, а іноді й кардинально змінюють сенс сказаного. АІ нездатна розшифрувати ці сигнали, якщо тільки це не ІС опрацювання зображень і звуку, здатна аналізувати обидва набори даних у відеофайлах.

· Мови з низькими ресурсами – ІСПМТ для мов із меншим обсягом збережених даних.

· Змішання мов – у багатомовних спільнотах люди використовують слова з декількох мов у одній розмові. Це створює складності для ІСПМТ, оскільки необхідно опрацьовувати лексичні й граматичні моделі під час переходу між мовами і використовувати ширший загальний набір можливих моделей.

Для розпізнавання мовлення використовують переважно три алгоритми [39]: прихована марковська модель, динамічне викривлення часу, штучні нейронні мережі.

Прихована марковська модель (ПММ) – це статистична модель, в якій модельована ІС передбачається марковським процесом із прихованими станами. ПММ можна уявити як найпростішу динамічну баєсівську мережу. ПММ – це набір станів, пов'язаних переходами, що починається у певному початковому стані. На кожному кроці дискретного часу відбувається перехід у новий стан, а потім генерується один вихідний символ у цьому стані. Вибір переходу та вихідний символ випадкові, керуються розподілом імовірностей. ПММ можна уявити як "чорну скриньку", де послідовність вихідних символів, що генеруються із плином часу, є відкритою, але послідовність станів, що відвідуються з плином часу, прихована від очей. У разі застосування ПММ для розпізнавання мови стани інтерпретуються як акустичні моделі, що вказують, які звуки, швидше за все, будуть почуті під час відповідних сегментів мовлення; водночас переходи забезпечують тимчасові обмеження, вказуючи, як стани можуть йти один за одним у послідовності [39]. Алгоритми ПММ: Форварда (для розпізнавання ізольованих слів), Вітербі (для розпізнавання безперервного мовлення) та прямого-оборотного ходу (для навчання ПММ). Обмеження ПММ: постійне спостереження за кадрами; марковське припущення; відсутність формальних методів вибору топології моделі; необхідна велика кількість навчальних даних; припущення про умовну незалежність.

Алгоритм динамічної трансформації тимчасової шкали (АДТТШ). Найпростіший спосіб розпізнати ізольований зразок слова – порівняти його із кількома шаблонами слів, що зберігаються, та визначити, який із них підходить найкраще. Ця задача ускладнюється низкою факторів. По-перше, різні зразки слова матимуть дещо різну тривалість. Вирішення проблеми – нормалізувати шаблони та невідоме мовлення, щоб вони мали однакову тривалість. Однак інша проблема полягає у тому, що темп мови може бути непостійним (оптимальне вирівнювання між шаблоном та зразком мови може бути нелінійним). Алгоритм полягає у пошуку оптимального вирівнювання між двома заданими (залежать від часу) послідовностями за певних обмежень, хоч інтуїтивно зрозуміло, що послідовності деформуються нелінійно, щоб відповідати одна одній. Спочатку цей алгоритм використовували для порівняння різних мовних моделей. У таких ПО, як пошук даних та інформаційний пошук, алгоритм успішно застосовують для подолання тимчасових деформацій та різних швидкостей, пов'язаних із залежними від часу даними.

Штучну нейронну мережу (ШНМ) визначають як модель міркувань, що ґрунтується на людському мозку. Мозок складається із взаємопов'язаного набору нервових клітин, або базових одиниць опрацювання інформації (нейронів), і містить майже 10 мільярдів нейронів та 60 трильйонів зв'язків – синапсів між ними. Завдяки використанню декількох нейронів одночасно мозок виконує функції набагато швидше, ніж найшвидші сучасні комп'ютери. Кожен нейрон має дуже просту структуру, але велика кількість таких елементів є величезною обчислювальною потужністю. Нейрон сформований із тіла клітини, соми, кількох дендритів та одного аксона. ШНМ складається із низки дуже простих процесорів (нейронів) як аналогів біологічних нейронів у мозку. Нейрони з'єднані

зваженими зв'язками, що передають сигнали від одного нейрона до іншого. Вихідний сигнал передається через вихідний зв'язок нейрона. Цей зв'язок поділяється на кілька гілок, які передають той самий сигнал. Вихідні гілки закінчуються на вхідних з'єднаннях інших нейронів у ШНМ. Перцептрон – найпростіша форма ШНМ, що складається із одного нейрона з регульованими синаптичними вагами та жорстким обмежувачем. Існує два типи ШНМ:

1. ШНМ прямого поширення, в якій інформація рухається тільки в одному напрямку – вперед від вхідних вузлів, через приховані вузли (якщо вони є) та до вихідних вузлів. У мережі немає циклів чи петель [40].
2. Рекурентна ШНМ, у якій з'єднання між вузлами утворюють граф, орієнтований у часі. Це створює внутрішній стан мережі, що дає їй змогу проявляти динамічну поведінку в часі [41].

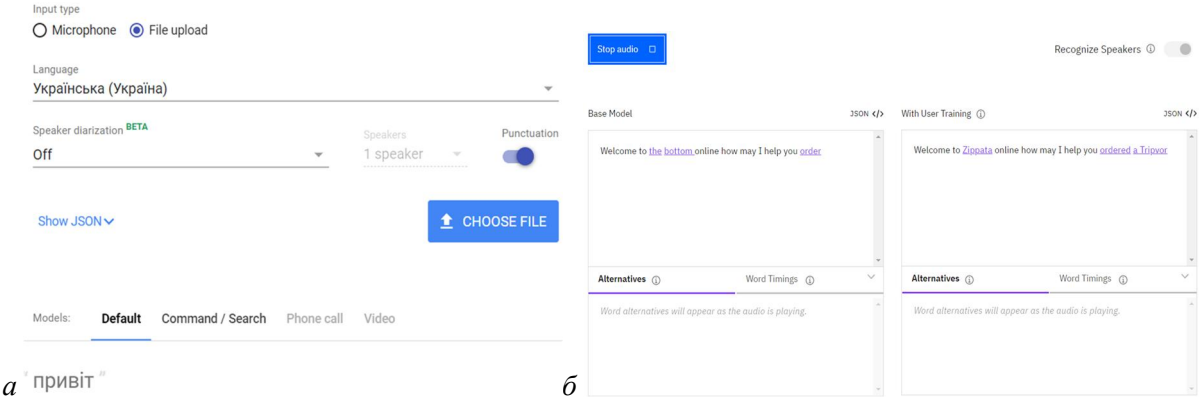
Серед переваг ШНМ – необхідність менш формального статистичного навчання, здатність неявно виявляти складні нелінійні зв'язки між залежними та незалежними змінними, виявляти всі можливі взаємодії між змінними-провісниками та доступність кількох алгоритмів навчання. Це нелінійні моделі, прості у використанні та розумінні порівняно зі статистичними методами. Вони є непараметричними моделями, тоді як більшість статистичних методів – параметричні моделі, які потребують вищого рівня знань у галузі статистики.

Сьогодні існує безліч ІСПМТ, деякі з яких доступні безкоштовно, багато інших – у вигляді SDK і API, призначених для корпоративних клієнтів.

Компанія Google [42], яка підтримує понад 120 мов, сьогодні є безперечним лідером у сфері розпізнавання мови. Голосовий пошук, транскрипція аудіо у текст та інші розширені послуги доступні в численних онлайн-сервісах Google, таких як Google Docs, Google Translate тощо. Перевагою цього рішення є велика кількість підтримуваних мов, а також пристосування до різних типів розмов – від телефонних до медичних. Окрім цього, ***Google Speech-to-text*** є доволі точним та швидким рішенням (рис. 1, *а*). До недоліків цього API належить висока вартість послуг, адже компанія надає лише перших 60 хв безкоштовно, а надалі бере плату за кожних 15 секунд запису. Також істотним обмеженням є те, що транскрипція файлів здійснюється лише за допомогою Google Cloud Bucket, який також є комерційним продуктом від Google Cloud.

IBM Watson [43] – один із провідних дослідників та розробників ІСПМТ. AI суперкомп'ютера Watson доволі добре відомий (рис. 1, б). Це орієнтований на підприємства сервіс із широким спектром застосунків, одним із яких є перетворення мовлення на текст. Нині він підтримує сім основних мов.

Компанія Dragon випустила один із перших ІСПМТ на споживчому ринку. Компанія, як і раніше, є надійним постачальником ІСПМТ, особливо серед медичних працівників. Сучасні версії ПЗ використовують передові функції AI. Здебільшого Dragon Anywhere, який є хмарною версією ***Dragon Naturally Speaking***, дає змогу перетворювати мовлення на текст як на настільному комп'ютері, так і на мобільному пристрої (рис. 2). Це найкраща ІСПМТ для мобільних пристроїв. Для користувачів, яким потрібен просунутіший словник, Dragon також пропонує програми Dragon Legal (для студентів-юристів) і Dragon Medical (для студентів-медиків).



а *б*

Рис. 1. Вигляд вікна демо для: а – Google Cloud Speech-To-Text; б – IBM Watson

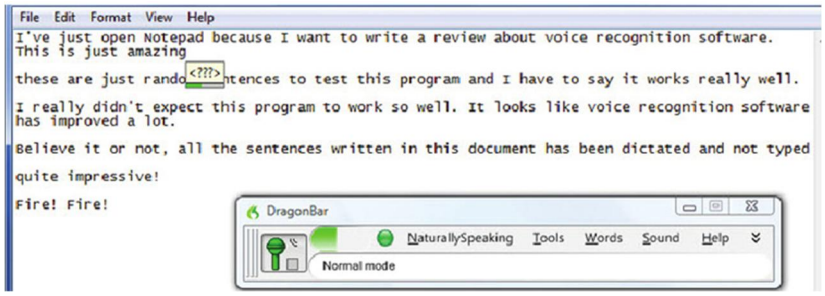


Рис. 2. Вигляд вікна Dragon NaturallySpeaking

Microsoft Dictate [44] – безкоштовна ІСПМТ, створена в Microsoft Garage як багатофункціональний застосунок, що використовує ту саму передову ІТ розпізнавання мови, яка використовується у віртуальному помічнику Microsoft Cortana. Dictate є доповненням Microsoft Office і добре працює з Word, PowerPoint і Outlook. Після встановлення можна отримати доступ до нього через вкладку “Диктовка”, яка відображається у правому верхньому куті панелі інструментів. Застосунок підтримує голосові команди для більшості стандартних операцій, таких як введення або редагування тексту, переміщення курсора на новий рядок і додавання знаків пунктуації (рис. 3, *а*).

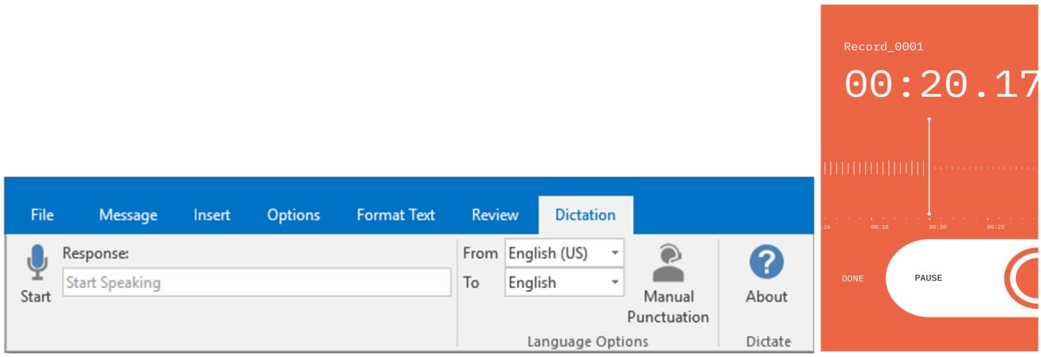


Рис. 3. Вигляд панелі для запису аудіо (а) та вікна запису аудіо (б)

Odrey [45] працює і як розшифрувальник голосових повідомлень, і як диктофон. Це означає, що можна записати голосове повідомлення за допомогою ІСПМТ, відтворити файли або просто завантажити файли в застосунок із пристрою. Застосунок транскрибує записану мову як з аудіо, так і з відеофайлів (рис. 3, *б*).

Для кращого розуміння та аналізу розглянутих рішень створено таблицю порівнянь (табл. 1). У ній розглянуто кожну з ІСПМТ за такими критеріямт: безкоштовна версія, підтримка завантаження

файлів, підтримка запису файлів, вид програми та підтримка української мови.

Таблиця 1

Таблиця порівнянь розглянутих рішень

Критерій	Google	IBM	Dragon	Microsoft	Odrey
Підтримка запису файлів	Так	Так	Так	Так	Так
Вид програми	SDK	SDK	Десктопна/мобільна	Інтегрована у MS Office	Мобільна
Підтримка української мови	Так	Ні	Ні	Так	Так
Безкоштовна версія	Частково	Частково (500 хв.)	Частково	Ні	Так
Підтримка завантаження файлів	Так	Так	Ні	Ні	Так

Більшість розглянутих ІСПМТ мають підтримку української мови, але всі вони реалізовані як десктопні застосунки, мобільні застосунки або SDK, тому для кращого користувацького досвіду вирішено створювати проєкт та демоверсію вебзастосунку, адже він не потребує скачування та встановлення додаткових програм. Також вирішено, що розроблювана ІСПМТ підтримуватиме два типи опрацювання файла: через запис файлів за допомогою мікрофона та через завантаження аудіофайла із файлової системи.

Формулювання мети та завдання

Головною метою є створення ІС перетворення звукового україномовного тексту на письмовий. Для цього необхідно досягти таких цілей: “Аналіз теми”, “Проєктування системи” та “Створення системи” (рис. 4, *а*). Для “Аналізу теми” необхідно виконати такі кроки:

· Проаналізувати конкретні рішення на ринку, виявити недоліки, щоб уникнути їх під час розроблення ІСПМТ.

· Аналіз алгоритмів для створення ІСПМТ для визначення найоптимальнішого способу виконання поставленого завдання.

· Аналіз наявних електронних масивів даних для пришвидшення навчання ІСПМТ, а також підвищення точності, з якою система переводить мовлення у текст. У наш час алгоритми та підходи до розв’язання таких задач швидко змінюються і необхідно постійно стежити за дослідженнями. Для “Проєктування системи” необхідно досягти трьох підцілей:

1. Побудова всіх необхідних діаграм. Потрібно побудувати діаграми класів типу IDEF0, в якій розглянуто всі процеси та ієрархію розроблюваної системи. Ця діаграма необхідна передусім для розуміння головної мети та процесів, які потрібно виконати.
2. Побудова внутрішньої архітектури модулів, класів та інфраструктури.
3. Вибір платформи для створюваної системи.

Критерієм для перших двох підцілей є “Якість”, а для третьої – “Практичність”. Остання ціль – “Створення системи”. На цьому етапі розробник матиме чітко сформульовану мету, а також діаграми та спроєктовану архітектуру. Для досягнення цієї цілі потрібно забезпечити такі підцілі:

· Реалізація міжмодульної взаємодії, яка має бути гнучкою та швидкою.

· Створення ІСПМТ на основі збирання та аналізу попередньої інформації.

· Тестування ІСПМТ та усіх функцій системи.

Критерії для останньої частини – “Швидкодія” і “Надійність”. Наступним етапом після побудови дерева цілей є визначення типу ІСПМТ. Для цього вибрано чотири основні типи ІС: інформаційно-аналітична, інформаційно-керуюча, система підтримки прийняття рішень та інформаційно-дорадча (рис. 4, *б*). Починати варто із матриці попарних порівнянь (табл. 2), а саме визначення оцінок переваг одних критеріїв системи над іншими. Створено матриці попарних

порівнянь для кожного з вибраних типів ІС відповідно до критерію якості (табл. 3).

Таблиця 2

Матриця попарних порівнянь критеріїв

№	Критерій	1	2	3	4	5	Оцінка	Вектор пріоритетів
1	Актуальність	1	1	2	1	2	1,32	0,24
2	Практичність	1	1	2	0,5	1	1	0,18
3	Якість	0,5	0,5	1	0,33	1	0,61	0,11
4	Швидкодія	1	2	3	1	3	1,78	0,33
5	Надійність	0,5	1	1	1	0,33	0,7	0,13

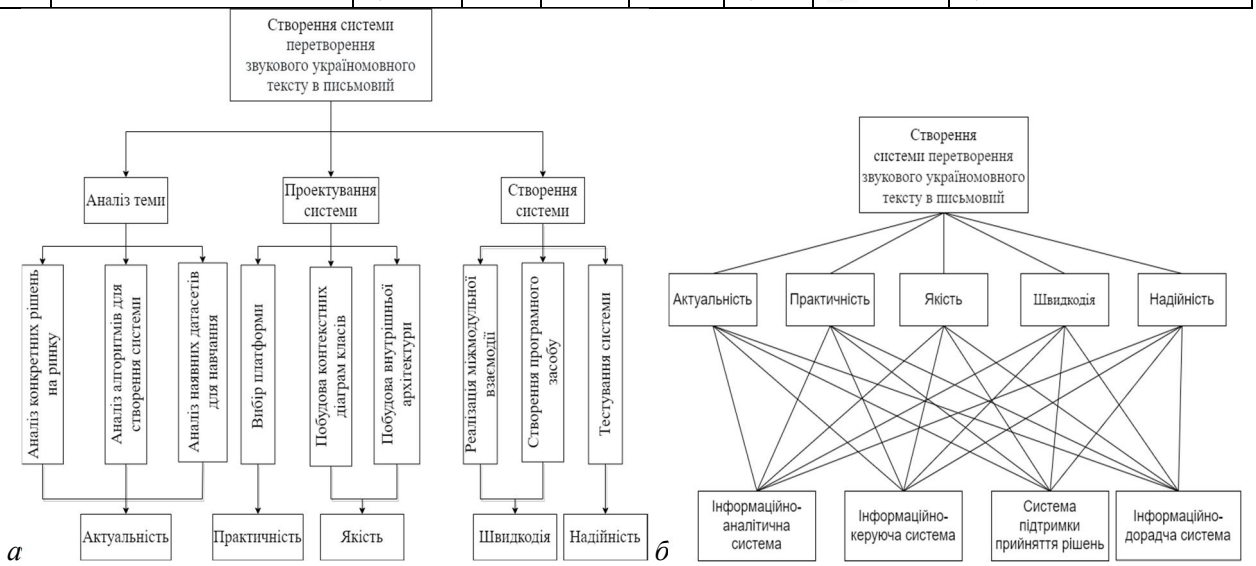


Рис. 4. Дерево цілей системи перетворення звукового україномовного тексту на письмовий (а) та ієрархічна структура задачі визначення типу ІС (б)

Таблиця 3

Матриця попарних порівнянь критеріїв

№	Тип інформаційної системи	1	2	3	4	Оцінка	Вектор пріоритетів
Матриця попарних порівнянь для критерію “Актуальність”							
1	Інформаційно-аналітична	1	4	2	1	1,68	0,38
2	Інформаційно-керуюча	0,25	1	0,5	0,5	0,5	0,11
3	Система підтримки прийняття рішень	0,5	2	1	0,5	0,84	0,19
4	Інформаційно-дорадча система	1	2	2	1	1,41	0,31
Матриця попарних порівнянь для критерію “Практичність”							
1	Інформаційно-аналітична	3	3	2	2	2,45	0,49
2	Інформаційно-керуюча	0,33	1	0,5	0,5	0,54	0,11
3	Система підтримки прийняття рішень	0,5	2	1	2	1,19	0,24
4	Інформаційно-дорадча система	0,5	2	0,5	1	0,84	0,17
Матриця попарних порівнянь для критерію “Якість”							
1	Інформаційно-аналітична	1	2	2	2	1,68	0,4
2	Інформаційно-керуюча	0,5	1	1	0,5	0,71	0,16
3	Система підтримки прийняття рішень	0,5	1	1	0,5	0,71	0,16
4	Інформаційно-дорадча система	0,5	2	2	1	1,19	0,28
Матриця попарних порівнянь для критерію “Швидкодія”							
1	Інформаційно-аналітична	1	3	3	3	2,28	0,5
2	Інформаційно-керуюча	0,33	1	1	2	0,88	0,19

3	Система підтримки прийняття рішень	0,33	1	1	0,5	0,62	0,14
4	Інформаційно-дорадча система	0,33	0,5	2	1	0,76	0,18
Матриця попарних порівнянь для критерію “Надійність”							
1	Інформаційно-аналітична	1	2	1	1	1,19	0,29
2	Інформаційно-керуюча	0,5	1	1	1	0,84	0,21
3	Система підтримки прийняття рішень	1	1	1	1	1	0,25
4	Інформаційно-дорадча система	1	1	1	1	1	0,25

Розраховано результуючі пріоритети для кожного із критеріїв (табл. 4).

Таблиця 4

Ієрархічний синтез критеріїв та типів систем

Критерій / Тип системи	Актуальність	Практичність	Якість	Швидкодія	Надійність	Результуючий пріоритет
Інформаційно-дорадча	0,31	0,17	0,28	0,18	0,25	0,23
Інформаційно-керуюча	0,11	0,11	0,16	0,19	0,21	0,15
Інформаційно-аналітична	0,38	0,49	0,4	0,5	0,29	**0,43**
Підтримки прийняття рішень	0,19	0,24	0,16	0.14	0,25	0,18

У результаті найбільшим результатом є 0,43, що відповідає інформаційно-аналітичний системі. Вибір цього типу системи допоможе зробити програму надійнішою, якіснішою та швидкіснішою.

Виклад основного матеріалу

IDEF0 – це методологія графічного описання систем і процесів діяльності організації як безлічі взаємозалежних функцій. Дає змогу досліджувати функції організації, не пов’язуючи їх із об’єктами, що забезпечують їх реалізацію [46–48]. Контекстна діаграма (рис. 5) складається із одного блока, що описує функцію верхнього рівня, її входи, виходи, управління та механізми, разом із формулюваннями мети моделі.

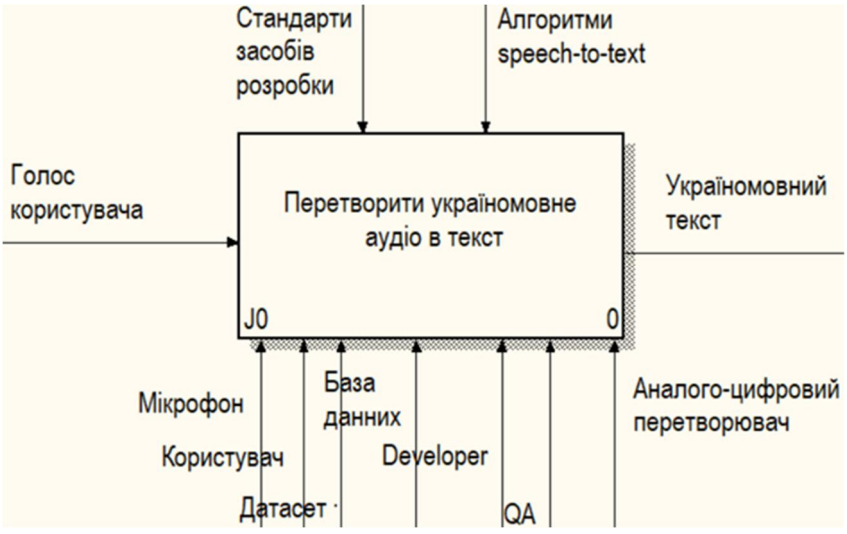


Рис. 5. Контекстна діаграма процесу “Перетворити україномовне аудіо на текст”

“Голос користувача” – це дані, які опрацьовує перетворення звуку на письмовий текст. Стрілками управління для цієї системи є “Стандарти засобів розробки” та “Алгоритми speech-to-text”. Ресурси, потрібні для виконання функції, це “Датасет”, “Мікрофон”, “Developer”, “QA”, “База даних” та “Аналого-цифровий перетворювач”. Після опрацювання вхідних даних, механізмів та за умови дотримання елементів управління на виході отримано “Україномовний текст”. Кожна **дочірня**

діаграма складається із дочірніх блоків і стрілок, які забезпечують необхідну на цьому рівні деталізацію батьківського блока. Отже, дочірня діаграма описує ту саму ПО, що і її батьківський блок. Для створення дочірньої діаграми (рис. 6) потрібно декомпонувати основний процес на підпроцеси. Створена діаграма верхнього рівня має три блоки: “Опрацювати аудіофайл”, “Відшукати фонеми”, “Перетворити набір фонем на текст”. Перший блок відповідатиме за запис аудіофайла та його опрацювання, щоб в результаті отримати “Аудіофайл з відповідними звуками”. “Відшукати фонеми” відповідатиме за розподіл аудіофайла на дуже малі частини та подальший пошук фонем для кожної із частин аудіофайла. Процес “Перетворити набір фонем на текст” необхідний для того, щоб із масиву знайдених фонем утворити повноцінний текст та відобразити для користувача.

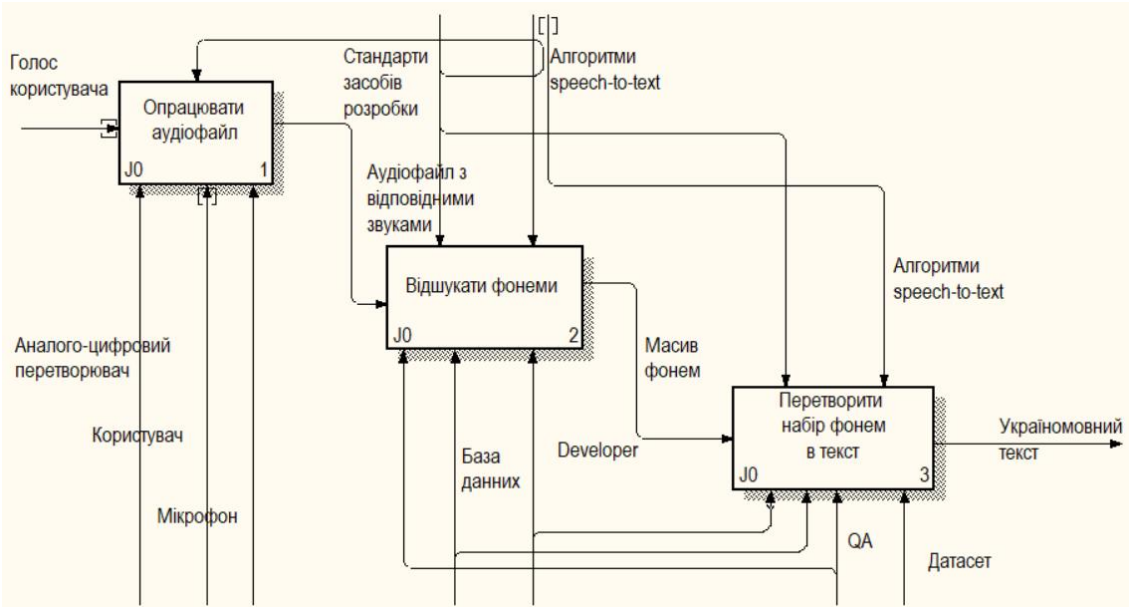


Рис. 6. Діаграма верхнього рівня процесу “Перетворити україномовне аудіо на текст”

Для детальнішого описання процесів верхнього рівня розроблюваної ІСПМТ здійснено їх декомпозицію. На рис. 7 зображено діаграму для першого підпроцесу: “Опрацювати аудіофайл”. Діграма має три блоки: “Почати запис аудіо” відповідає за початок запису голосу користувача, “Зберегти аудіофайл” – за закінчення запису та збереження аудіофайла, блок “Відфільтрувати звук” потрібен для того, щоб за допомогою аналого-цифрового перетворювача проаналізувати збережене аудіо та виділити лише релевантні звуки. Стрілкою управління для цієї діаграми є “Стандарти засобів розробки”, потрібні, щоб система працювала якісно, а дані користувача були у безпеці. Необхідні ресурси – “Developer”, “Користувач”, “QA”, “Мікрофон” та “Аналого-цифровий перетворювач”. Після того як аудіофайл записано та збережено, а його вміст відфільтровано, ІСПМТ отримає на виході “Аудіофайл з відповідними звуками”. Декомпозицію процесу “Відшукати фонеми” подано на рис. 8.

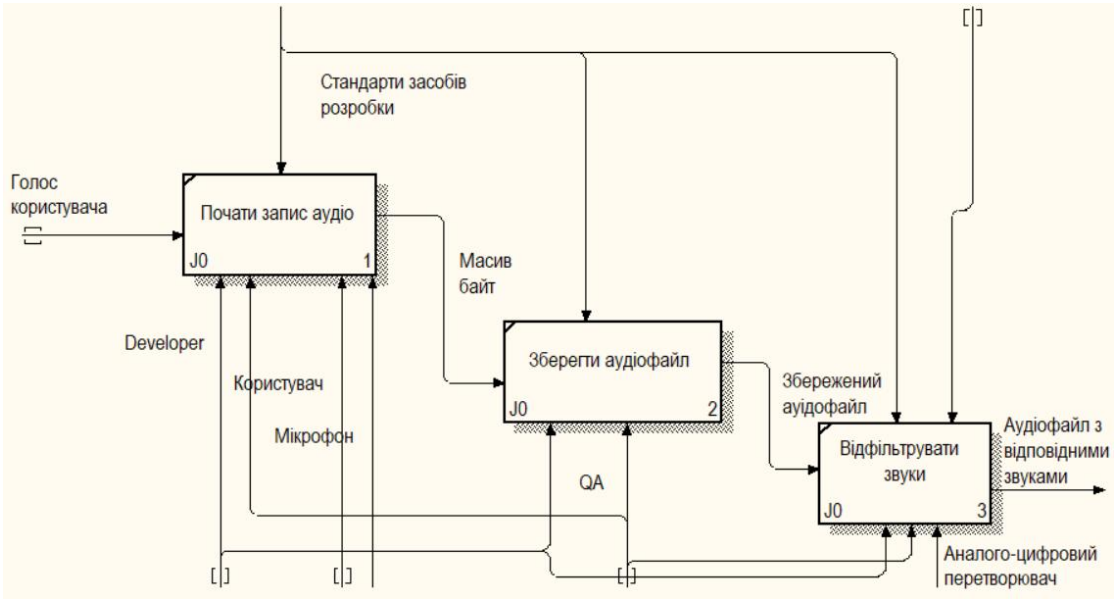


Рис. 7. Деталізована діаграма підпроцесу "Опрацювати аудіофайл"

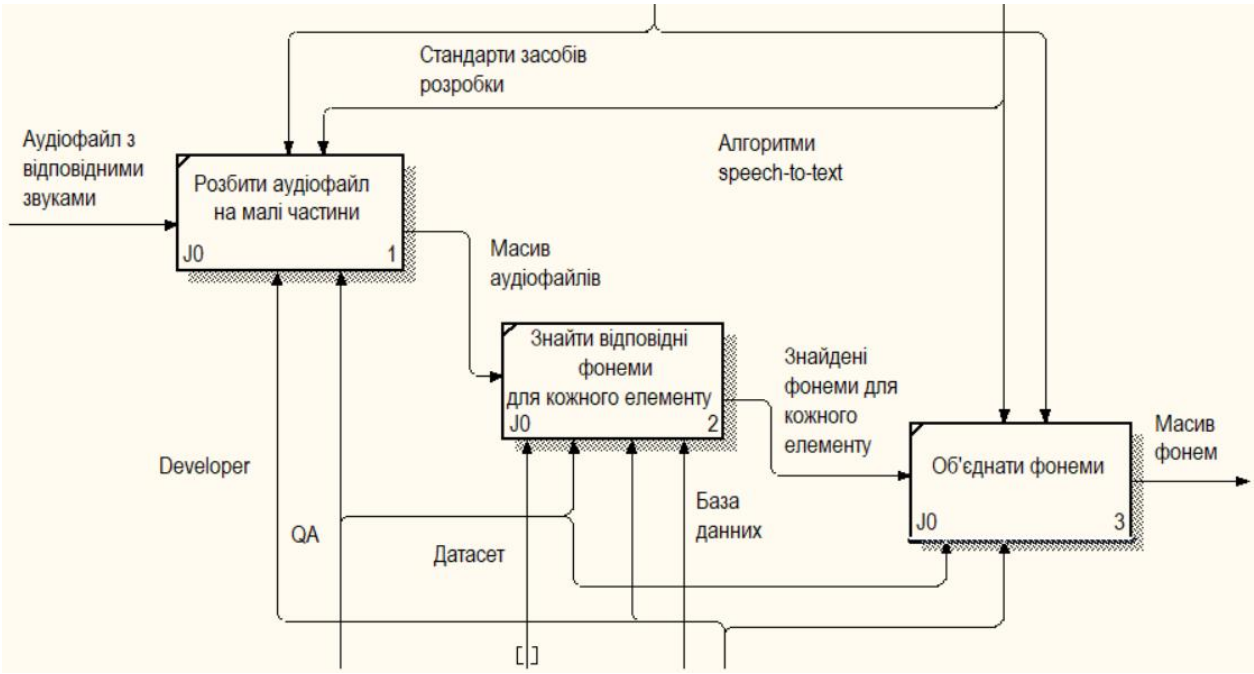


Рис. 8. Деталізована діаграма підпроцесу "Відшукати фонеми"

Діаграма складається із трьох процесів: "Розбити аудіофайл на малі частини", "Знайти відповідні фонеми для кожного елемента", "Об'єднати фонеми". "Відшукати фонеми" відповідатиме за розподіл аудіофайла на менші частини. Наступний процес "Знайти відповідні фонеми для кожного елемента" відповідатиме за знаходження відповідної фонеми для кожної з частин аудіофайла. Третій процес "Об'єднати фонеми" відповідатиме за збереження усіх знайдених фонем у належній послідовності. Стрілками контролю тут є "Стандарти засобів розробки", "Алгоритми speech-to-text", вони забезпечують гнучкість системи та правильність алгоритму пошуку фонем відповідно. Ресурсами є "Developer", який відповідатиме за розроблення модулів, "QA", "Датасет", який містить фонеми, та "База Даних", яка зберігає датасет. У результаті виконання цих трьох процесів ІСПМТ на виході отримає "Масив фонем".

Декомпозицію процесу “Перетворити набір фонем на текст” подано на рис. 9. Ця діаграма складається із двох процесів: “Прогнати через математичну модель” та “Відобразити текст”. Перший процес відповідатиме за перетворення масиву фонем на текст, приймає для цього “Масив фонем” як стрілку входу, а як ресурси “Датасет”, в якому є набір відомих речень, фраз чи слів, та “Developer”, що відповідатиме за реалізацію математичної моделі. “Відобразити текст” на вхід приймає “Перетворені на текст фонеми” та відповідає за відображення результату для користувача. Для цього процесу ресурсами є “Developer” та “QA”. Стрілками управління знову є “Стандарти засобів розробки” та “Алгоритми speech-to-text”. У результаті користувач отримає “Україномовний текст”. Ієрархію процесів для ІС перетворення україномовного аудіо на текст подано на рис. 10.

Головна мета – “Перетворити україномовне аудіо на текст”. Цей процес містить в собі дочірні процеси із їхніми підпроцесами:

1. Опрацювати аудіофайл: “Почати запис аудіо”, “Зберегти аудіофайл”, “Відфільтрувати звуки”.
2. Відшукати фонеми: “Розбити аудіофайл на малі частини”, “Знайти відповідні фонеми для кожного елемента”, “Об’єднати фонеми”.
3. Перетворити набір фонем на текст: “Прогнати через математичну модель”, “Відобразити текст”.

Орієнтуючись на аналіз аналогів, ми вирішили розробляти ІС як вебзастосунок, щоб користувач не був прив’язаний до конкретної платформи, а швидкість роботи ІС залежатиме лише від характеристик сервера. Згідно із даними сервісу StatCounter, найпопулярнішими браузерами з початку 2021 р. стали Google Chrome – 64,36 %, Safari – 19,13 %, Edge – 4,07 % [49].

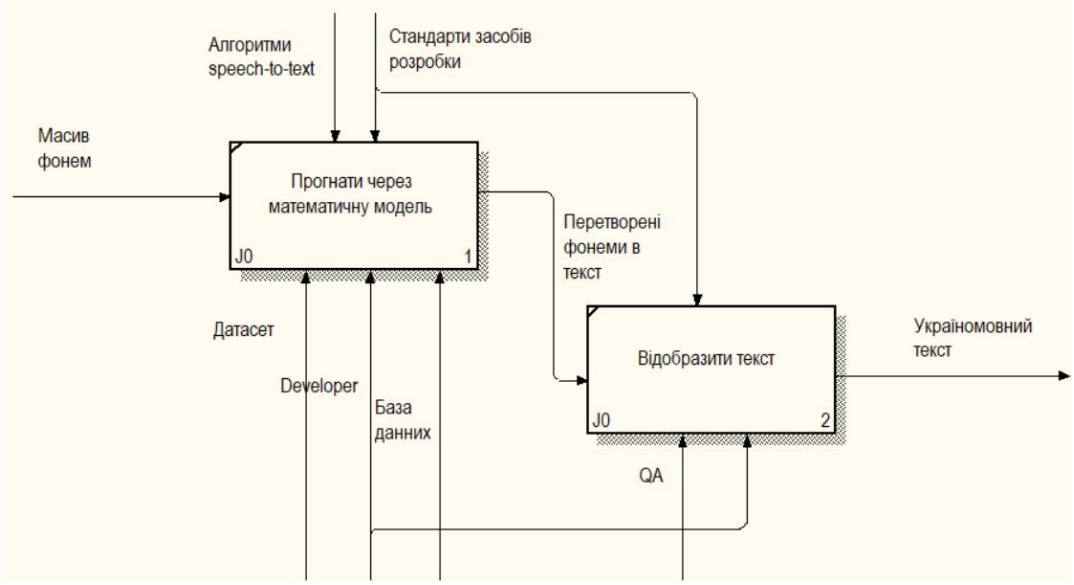


Рис. 9. Деталізована діаграма підпроцесу “Перетворити набір фонем на текст”

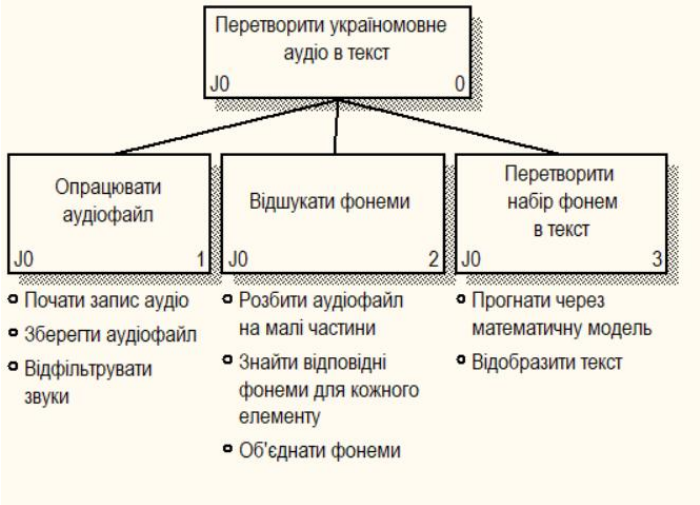


Рис. 10. Ієрархія процесів

Першу версію Google Chrome створено у 2008 р. на основі браузера із відкритим кодом Chromium із чистого аркуша, з урахуванням того, що нині більшість вебсайтів – не просто сторінки, а цілі вебзастосунки зі своєю інфраструктурою, командою розробників тощо. Перевагами такого підходу стали швидкість, безпека та стабільність. Не менш важливі характеристики – простий дизайн та зрозумілий інтерфейс. Для підтримки безпеки Chrome періодично завантажує оновлення із двох чорних списків і попереджує користувача, коли той намагається відвідати шкідливий сайт із цього переліку. Важливим параметром захищеності є також те, що для кожної вкладки створюється окремий процес, адже це дає змогу уникати ситуації, коли одна вкладка має можливість впливати на іншу. Крім того, створення окремого процесу для кожної вкладки позитивно позначається на стабільності браузера, адже навіть критичний збій у процесі однієї вкладки не зможе завадити роботі інших процесів. До недоліків варто зарахувати високе навантаження на оперативну пам’ять, для користувачів зі слабкими комп’ютерами це може стати критичною проблемою у разі роботи з великою кількістю вкладок.

Safari розроблений корпорацією Apple та входить до складу ОС MacOS та IOS. Цей браузер вперше випустили в 2003 р., лише для MacOS, і вже через чотири роки його імпортували в ОС Windows. Переваги Safari – простий дизайн та компактний інтерфейс. Для розробників браузер має функцію перегляду DOM вебсторінки, а також підтримує більшість сучасного функціоналу CSS і HTML, що полегшує розроблення інтерфейсу вебзастосунків. До недоліків можна зарахувати низьку швидкість запуску самого браузера, а також застарілість у питаннях безпеки.

Третій за популярністю браузер Edge – браузер компанії Microsoft, випущений у 2015 р. Працює на основі Chromium, що надає користувачам доступ до великої бібліотеки розширень. Спочатку Edge планували лише для ОС Windows 10, але потім імпортували й на інші платформи сім’ї Windows, MacOS та Linux. Його перевага – швидкість роботи, адже браузер використовує ту саму ІТ, що і Google Chrome, однак його не оминули проблеми з оперативною пам’яттю.

Окрім розглянутих браузерів, є численні менш популярні аналоги, також зі своїми перевагами та недоліками. Проте основним критерієм для розробника є функціонал, який підтримує та чи інша версія браузера. Ця відмінність пояснюється тим, що браузери використовують різні ІТ, які по-різному опрацьовують та відображають HTML теги та CSS стилі. Щоб охопити більшу кількість користувачів, система розроблятиметься як кросбраузерний вебзастосунок. Кросбраузерність – це властивість вебзастосунку однаково відображатися та функціонувати в усіх часто використовуваних браузерах. Далі варто проаналізувати, яку мову програмування найдоцільніше використовувати для серверної частини додатка. У квітні 2022 р. найпоширенішими варіантами були Java, Python, Node.JS [50].

Java – об'єктно-орієнтована мова програмування, розроблена у 1996 р. Ця мова ідеально підходить для написання як великих, так і малих вебзастосунків (табл. 5), завдяки розвиненій інфраструктурі мови. Також вона має технологію рендеру сторінок на боці сервера, це допомагає уникнути використання клієнтських фреймворків, таких як Angular, React, Vue. Ця ІТ підходить для застосунків, для яких важливий функціонал, а інтерфейс та дизайн другорядні.

Python – об'єктно-орієнтована мова програмування, але, на відміну від Java, із динамічною типізацією. До переваг цієї мови програмування можна зарахувати розвинену інфраструктуру (де можна знайти бібліотеки для будь-якої мети) та простий синтаксис. Недоліками Python є динамічна типізація, що ускладнює розроблення складного мультимодульного проєкту. Мова програмування є інтерпретованою, що призводить до сповільнення роботи ІС.

Node.JS [51] – мова програмування, в якій за основу взято JavaScript, що перетворило її на мову загального використання. Переваги Node.JS – асинхронна модель виконання запитів, а також неблокувальне введення/виведення. Недоліки: незрілість мови програмування, інфраструктура менш розвинена, ніж у Java чи Python, а також нездатність опрацьовувати складні завдання.

Важливо проаналізувати засоби, які дають змогу створювати вебзастосунки. Для Java здебільшого це Spring, для Python це Django, а у випадку Node.JS – це Express.

Spring [52] – фреймворк, який містить у собі набір різних інструментів (табл. 5), що полегшує роботу із автентифікацією, авторизацією, опрацюванням запитів, БД, мікросервісів тощо. Окрім великого вибору інструментів, Spring автоматизує велику кількість процесів та налаштувань, що прискорює написання коду та розгортання проєкту. Переваги – поширеність серед користувачів.

Django [53] – відкритий фреймворк для розроблення вебзастосунків на Python. Містить набір основних інструментів, таких як: ORM, автентифікація, сторінка адміністратора, форми тощо. До основних переваг можна зарахувати безпечність за замовчуванням, фреймворк бере на себе механізми запобігання атакам, наприклад SQL ін'єкціям та CSRF. На відміну від попереднього фреймворку, робота з ORM у Django застаріла, що ускладнює виконання складних транзакцій.

Express [54] – відкритий вебфреймворк для Node.JS. Реалізований як вільне та відкрите ПЗ із ліцензією MIT. Переваги: гнучкість, простота, розширюваність та продуктивність. Express.js разом із Node.js можна використовувати для створення API для односторінкових, багатосторінкових, гібридних мобільних і вебзастосунків.

Отже, проаналізувавши три різні вебфреймворки за такими аспектами, як різноманітність функціоналу, інфраструктура, документація, спільнота, продуктивність та якість, надаємо перевагу Java та Spring як найбільш зрілим та розвиненим ІТ.

Таблиця 5

Таблиця переваг та недоліків мови програмування Java

Переваги	Недоліки
ООП – концепція, яка використовується вже багато років та полегшує розроблення застосунків	Проблеми з продуктивністю порівняно з іншими низькорівневими мовами програмування
Мова програмування високого рівня, зі зрозумілим синтаксисом, що допомагає під час написання та перевірки коду	
Підтримує багато бібліотек, велика кількість з яких є уже зрілими технологіями, адже ця мова програмування є стандартом у корпоративному розробленні	Хоча на Java і можна розробляти GUI (прикладом може слугувати Intellij IDEA), але здебільшого це потребує великої кількості часу та дуже досвідченої команди
Крос-платформна мова програмування, а отже, не залежить від ОС	

Таблиця 6

Таблиця переваг та недоліків мови фреймворку Spring

Переваги	Недоліки
Фреймворк пропонує автоконфігурацію, завдяки чому зменшується кількість конфігураційних файлів та	Оскільки Spring налаштовує велику кількість залежностей за розробника, то і відстежити всі невикористовувані

складність налаштування проєкту	залежності доволі важко
Spring має вбудований сервер Tomcat, що зменшує час перезапуску застосунку	Для того, щоб якісно та швидко розробляти застосунки з використанням Spring, потрібно добре знати основи великої кількості різних частин фреймворку
Велика екосистема Spring дає змогу легко та швидко створювати вебзастосунки, взаємодіяти із БД та писати свої мікросервіси	

Окрім фреймворку Spring, для розроблення повноцінної серверної частини потрібно вибрати ІТ для зв'язку та взаємодією із БД, хмарним сховищем, а також тестування (табл. 6). Для зв'язку з БД Java пропонує невелику кількість фреймворків, проте більшість із них є зрілими ІТ. JDBC, Hibernate, Spring Data [55] – це основні технології, які допоможуть побудувати доменну модель та сервіси для операцій зі сховищем. Hibernate – це обгортка JDBC, а Spring Data – обгортка над Hibernate. Тому характеристики, які справді відіграють роль у виборі – це ефективність та простота написання коду. І в цьому виграє остання технологія – Spring Data, адже вона значно полегшує написання коду, а також має функцію автогенерації методів за допомогою правильних назв останніх у інтерфейсі. Процес налаштування фреймворку стає набагато легшим, адже здебільшого все, що потрібно, – це описати п'ять властивостей у спеціальному файлі "application.properties" (табл. 7).

Таблиця 7

Таблиця переваг та недоліків фреймворку Spring Data

Переваги	Недоліки
Spring Data як частина фреймворку Spring дає змогу доволі легко інтегруватись у вже написаний код із використанням цього фреймворку	Оскільки Spring Data бере на себе генерацію коду, потрібно досить добре знати основи та стежити за згенерованими запитами
Spring Data набагато легша у використанні, ніж Hibernate або JDBC, адже виконує велику кількість функцій за розробника	Доволі важке налаштування Entity класів, адже потрібно знати, як саме створювати зв'язки між класами, та вміти їх оптимізувати
Підтримує автогенерацію методів, лише маючи правильно названі методи в інтерфейсі	Дуже часто витоки пам'яті через надмірну кількість згенерованих запитів або через неоптимальність транзакцій
Містить доволі хорошу документацію	

Для роботи із хмарним сховищем вибрано Google Cloud Storage [56] – ІТ для зберігання та доступу до файлів у інфраструктурі Google Cloud Platform. Поєднує у собі продуктивність, безпеку, відносну простоту використання, масштабованість, а також там можна безкоштовно зберігати до 15 гігабайт файлів. Для інтеграції з Java служба пропонує бібліотеку із ідентичною назвою: Google Cloud Storage. Вся взаємодія відбувається через спеціальний інтерфейс "Storage", який поєднує у собі велику кількість методів для гнучкої роботи.

Для покриття тестами основними інструментами є Junit, AssertJ та Mockito. Вони полегшують написання як модульних, так і інтеграційних тестів. Модульні тести – це тести найменшої частини коду, яка ізольована від інших частин. Інтеграційні тести – це тести, за допомогою яких тестують не ізольовану частину функціоналу, а взаємодію між кількома модулями програми. Junit містить різноманітний функціонал, але переважно перевіряє та підтверджує, що результат справді відповідає певним очікуванням. Хоча Junit пропонує свій клас для перевірки результатів, але він є доволі негнучким та некомфортним для використання, тому розробники додають до проєкту AssertJ. Це бібліотека, яка значно розширює можливості перевірки тестів, а також уможливлює багатокрокові перевірки. Кожен клас містить певні залежності, які потрібно створювати та ініціалізувати для перевірки тестів. Такий підхід доволі незручний для модульного тестування, адже виникає необхідність перевіряти ізольований уривок коду. Тому для підміни об'єктів використовують бібліотеку Mockito, яка дає змогу налаштовувати та повертати фіктивні дані, не створюючи та не налаштовуючи об'єкт.

Для проєктованої ІС вибрано реляційну БД, адже всі моделі мають визначену структуру. Найкращий варіант для використання – PostgreSQL. Це БД з відкритим кодом та розширеним функціоналом, яка підтримує велику кількість нових функцій, наприклад, взаємодію із типом даних

json. На користь цього вибору свідчить і те, що більшість вебзастосунків використовує саме цей продукт як основну БД, а також вона має драйвер для мови програмування Java.

Визначившись із серверною мовою програмування, варто також вибрати один із фреймворків для клієнтської частини. Серед основних варіантів: Angular, React, Vue (табл. 8).

Таблиця 8

Таблиця порівнянь Angular, React та Vue

Критерії	Angular	React	Vue
Стабільність	Так	Так	Так
Розробники	Google	Facebook	Alibaba
Спільнота	Велика	Велика	Невелика
Мова програмування	TypeScript	JavaScript	JavaScript
Швидкість розроблення	Нормальна	Нормальна	Швидка
Компонентний фреймворк	Так	Так	Так
Хороша документація	Так	Так	Так
Повторне використання коду	Так	Ні	Часткове
Автономність	Не потребує підключення сторонніх технологій	Потребує підключення сторонніх технологій	Вимагає підключення сторонніх технологій

Отже, після аналізу різних параметрів клієнтським фреймворком вибрано Angular, адже для його використання не потрібно підключення сторонніх технологій, а хороша документація разом з повторним використання коду та TypeScript допоможе написати код швидко та зробити його легкопідтримуваним.

Оскільки розроблювана ІС – це вебзастосунок, потрібно вибрати технологію для подальшого розгортання проєкту на віддаленому сервері. Для цього підійдуть Docker та Kubernetes.

Docker – це програмна платформа, створена для автоматизації розгортання та управління проєктами. Дає змогу створити контейнер для застосунку зі всіма його залежностями, а відтак запускати на будь-якій Linux системі. До переваг Docker можна зарахувати: абстрагування застосунку від хоста, легкість у масштабуванні, ізольованість середовища та легкість використання шарів. Однак у цієї програмної платформи достатньо недоліків, наприклад, вона не підтримує зворотну сумісність. Це означає, що нові версії програми можуть не підтримувати конфігурацію, написану для старої версії. Також для застосунків, які матимуть велике навантаження, важливо дуже чітко та якісно написати конфігурацію, а із цим зазвичай не можуть впоратись новачки.

Kubernetes – використовують для автоматизації розгортання та управління контейнерними проєктами. Також підтримує Docker як середовище для запуску контейнерів. До управління проєктами належить: автоматизоване планування, можливість самовідновлення, автоматичне розгортання, балансування навантаження, автоматичне масштабування середовища. До його плюсів можна зарахувати: підвищення продуктивності коду, збільшення надійності розгорнутого застосунку, відносна дешевизна використання Kubernetes. Великим мінусом платформи є її складність, адже Kubernetes має велику кількість різних функцій.

Сучасні проєкти – результат роботи великої кількості розробників, які можуть перебувати в будь-якій частині світу. Тому, окрім локальних змін, розробник повинен мати можливість додавати ці зміни в кодову базу проєкту. Тому створюють спеціальне ПЗ [57] – системи контролю версій, які допомагають розробниками керувати змінами, переглядати їхніх авторів та ефективно взаємодіяти зі змінами. У VSC кожен розробник може створити власну гілку, де матиме змогу додавати, редагувати або видаляти код, не змінюючи основного коду проєкту чи гілки інших розробників. Такі системи дають змогу зробити процес розроблення зрозумілішим та прозорішим. Системи контролю версії поділяють на два типи:

1. Централізовані системи контролю версій, які є застосунком з клієнтсерверною архітектурою, коли репозиторій міститься на віддаленому сервері та лише в одному екземплярі. Найочевиднішим недоліком є саме центральний репозиторій: якщо він вийде з

ладу, то всі зміни та результати роботи будуть втрачені.

2. Розподілені системи контролю версій, які дають змогу зберігати копію репозиторію у кожного розробника, який працює над проєктом. Вони виділяють центральний репозиторій, в який надсилаються зміни із локальних репозиторіїв. Необхідна постійна синхронізація локального репозиторію із центральним. Такий тип систем вирішує проблему з одним центральним репозиторієм, який містить усю історію змін, а також збільшує автономність розробника.

Основні системи контролю версій, які доволі часто використовують у проєктах – Git, Mercurial та SVN. Git – це безкоштовна розподілена система контролю версій, яка спочатку була призначена для роботи над ядром ОС Linux (рис. 11, а). Вона сумісна із різними протоколами (HTTP, SSH, FTP), забезпечує криптографічну аутентифікацію, а також високу швидкість роботи. Одразу після встановлення Git пропонує дві утиліти для роботи з ним, а саме: Git Bash та Git UI. Після аналізування стає зрозумілим, що завдяки високій швидкості, гнучкості та інтеграції з великою кількістю програмних засобів GIT є найкращим рішенням.

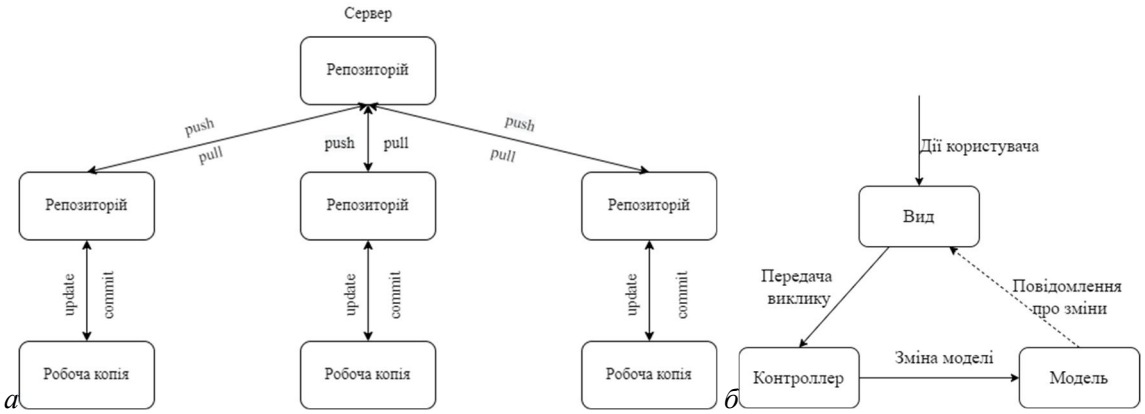


Рис. 11. Схема (а) роботи розподіленої системи контролю версій та (б) патерну MVC

Останнім важливим пунктом у проєктуванні ІСПМТ є вибір шаблону проєктування, адже завдяки цьому можна зробити програму гнучкою, надійною та легкою для тестування. Основним шаблоном, який використовують тепер у веброзробках, є MVC. Шаблон MVC (Model-View-Controller) [58] – це патерн проєктування вебзастосунків, який поєднує кілька шаблонів. У разі використання MVC модель даних програми розділена на три окремі компоненти, інтерфейс користувача і логіку взаємодії користувача з системою, завдяки чому модифікація одного з цих компонентів мінімально впливає на інші або не впливає зовсім. Концепція MVC поділяє дані, подання та опрацювання дій користувача на компоненти:

1. Модель – це об’єктна модель певної розроблюваної ПО, що містить, окрім самих даних, ще й методи роботи з цими даними. Особливістю моделі є те, що вона не містить інформації про способи візуалізації, тобто подання цих даних, а також безпосередньо не взаємодіє із користувачем.

2. Вид – відповідає за подання та відображення даних. Це може бути подання на рівні API, тобто експорт даних у JSON, XML або в іншому підтримуваному форматі, і за допомогою різних технологій рендеру сторінок.

3. Контролер – безпосередньо взаємодіє з користувачем, передаючи дані в ІС (рис. 11, б). Використовує модель для реалізації очікуваного функціоналу та передає його. На цьому рівні фільтрують та агрегують отримані дані, а також перевіряють права користувача.

Для вибору середовища розроблення спочатку потрібно проаналізувати вибрані ІТ. Для серверної частини – це Java, Spring, Spring Data, для клієнтської частини – Angular, HTML, CSS. Одним із найкращих середовищ розроблення для такого списку ІТ є Intellij IDEA Ultimate [59]. Це

середовище інтегрованого розроблення має безліч переваг, серед яких: інтуїтивно зрозумілий дизайн, велика кількість функціоналу, інтеграція зі Spring та велика бібліотека плагінів (рис. 12). Також підтримує можливість запуску проєктів, написаних на різних мовах програмування, а саме: Java, Angular, Kotlin, Node.js, React тощо. Для зручної роботи Intellij IDEA пропонує гнучкий спосіб пошуку файлів як за назвою, так і за написаним кодом або назвою методу. Окрім цього середовище має вбудований функціонал автогенерації коду (рис. 13), що полегшує розроблення для Java-розробників.

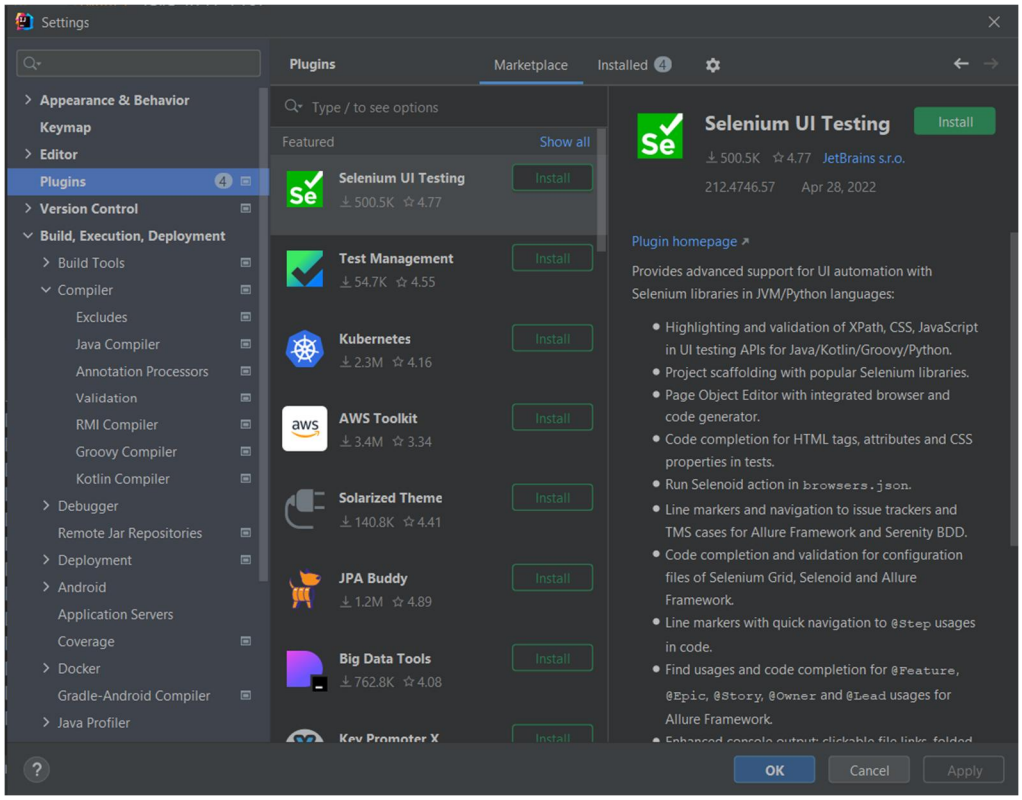


Рис. 12. Вигляд вікна плагінів

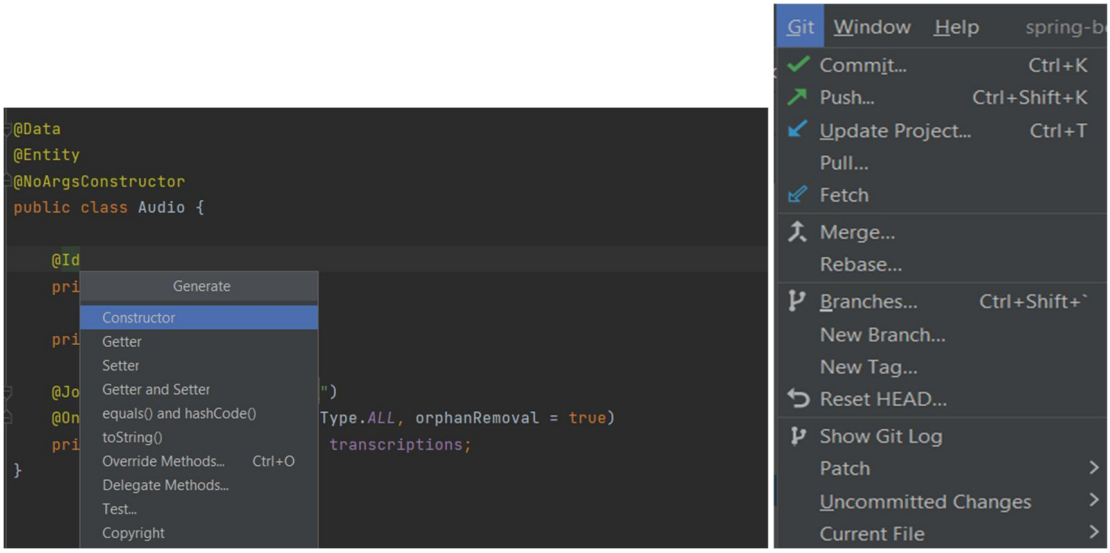


Рис. 13. Вигляд вікна генерації коду та вікна для взаємодії з GIT

Для роботи із БД вибрано Intellij IDEA Ultimate, яка забезпечує вбудований функціонал та підтримку підключення до різних БД, надсилання запитів та перегляду схем/даних. Є підтримка GIT

(рис. 13), а саме UI, на основі якого легко виконувати більшість робіт із системою контролю версій.

Під час проєктування ІСПМТ вибрано БД PostgreSQL, в якій зберігатимуться інформація про зареєстрованого користувача та завантажені аудіофайли (рис. 14) [60]. Відношення audio міститиме основну інформацію про записаний аудіофайл, а саме: Порядковий номер, Шлях до файла в Google Cloud Storage, Дату створення, Ідентифікатор користувача. Відношення transcription містить інформацію про всі можливі транскрипції аудіофайла: Порядковий номер, Транскрипцію, Ідентифікатор аудіофайла. Відношення user містить інформацію про юзера: Ідентифікатор користувача, Логін, Ім'я, Прізвище.

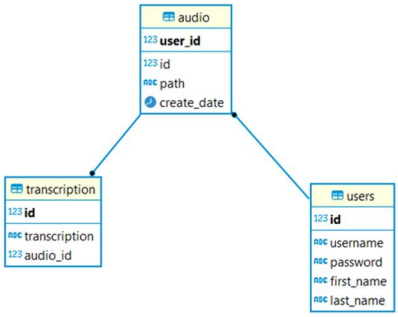


Рис. 14. Схема бази даних PostgreSQL

Розроблювана ІСПМТ орієнтована на будь-якого користувача. За допомогою цього функціоналу перетворення україномовного аудіозапису на текст доступне для всіх. Також користувач може зареєструватись у розроблюваній ІСПМТ, а надалі авторизуватись. Після цього йому стане доступною сторінка зі списком усіх його аудіофайлів, які трансльовано в письмовий текст, де він також зможе переглянути інформацію щодо кожного із цих файлів. Ця ІСПМТ має клієнт-серверну архітектуру. Реалізацію розділено на декілька різних модулів: для роботи з аудіофайлами; реєстрації та авторизації користувача; для роботи з БД; для роботи із хмарним сховищем. Модуль, що працює з аудіо файлами, призначений для опрацювання аудіофайла та подальшої трансформації у письмовий текст. Після опрацювання аудіофайла повертається масив з різними варіантами тексту. Також окремий модуль розроблено для системи реєстрації та авторизації користувача. Для процесу реєстрації користувача потрібно вказати пароль, логін та ім'я із прізвищем. Далі цей запис буде збережено в БД, а пароль зашифровано із використанням bcrypt алгоритму. Для цього використовують Spring Security:

```
@Configuration
@EnableWebSecurity
@RequiredArgsConstructor
@EnableGlobalMethodSecurity(prePostEnabled = true)
public class SecurityConfig extends WebSecurityConfigurerAdapter {
    @Value("${auth.uri}")
    private String authenticationPath;
    private final UserDetailsServiceImpl userDetailsService;
    private final JwtUnAuthorizedResponseAuthenticationEntryPoint entryPoint;
    private final JwtTokenAuthorizationOncePerRequestFilter requestFilter;
    @Bean
    public BCryptPasswordEncoder bCryptPasswordEncoder() {  return new BCryptPasswordEncoder();  }
    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws Exception {
        auth.userDetailsService(userDetailsService).passwordEncoder(bCryptPasswordEncoder());
    }
    @Bean
    @Override
    public AuthenticationManager authenticationManagerBean() throws Exception {
        return super.authenticationManagerBean();
    }
```

Після реєстрації користувач може здійснювати вхід за допомогою системи авторизації. У

функції логіна здійснюється перевірка, чи існує користувач із таким логіном та паролем. Для перевірки паролю неможливо просто отримати його з БД, адже він попередньо захешований. Тому для цього введений пароль спочатку хешується, а вже після цього порівнюється з тим, що міститься в БД. Якщо користувача не існує, то система повідомить про це, вивівши помилку на екран. В іншому випадку буде створено jwt токен авторизації, який поміщено у заголовок запиту. Надалі цей токен поміщатиметься у кожен запит, а система зможе перевірити його на валідність, щоб надати користувачеві доступ до деяких сторінок. Модуль взаємодії із БД відповідає за операції: підключення, збереження, оновлення, читання даних про записані аудіофайли або користувача. Для цього використано фреймворк Spring Data, який і виконує генерацію sql-запитів для подальшого їх надсилання до БД. Останнім модулем є модуль взаємодії із хмарним середовищем, щоб мати змогу ефективно зберігати та читати збережені аудіофайли. Використовується імпортована бібліотека Google Cloud Bucket.

Розроблювана ІСПМТ є вебзастосунком, тому, щоб її використовувати, необхідний засіб із доступом в інтернет та браузером. Наприклад, вимоги для ноутбука чи комп'ютера такі: Операційна система*:* Windows, MacOS, Linux; Процесор: Intel Pentium 4/Athlon 64; Пам'ять: 512 мб. Вхідними даними для ІСПМТ є аудіофайл, завантажений або записаний. Після його опрацювання у результаті роботи вебзастосунку користувач отримає вихідні дані – інформацію про аудіофайл та варіанти перетворення його на письмовий текст.

Створювана ІСПМТ є хорошим засобом для людей, які мають проблеми з друком великих обсягів тексту на клавіатурі або з прослуховуванням надісланого аудіофайла. ІС існує як вебзастосунок, що забезпечує її мобільність та легкість у використанні, під назвою "Ukrainian Speech-to-text". Клієнт розроблено на мові програмування TypeScript із використанням фреймворку Angular та середовища розроблення Intellij Idea Ultimate. Серверна частина використовує мову програмування Java і фреймворк Spring. Для написання серверної частини також використано середовище розробки Intellij Idea Ultimate. Програма існує як вебзастосунок, тому не прив'язана до ОС чи браузера. Користувачі матимуть змогу записати або завантажити аудіо та перевести його в текст, а авторизовані користувачі зможуть також переглядати всі записані аудіофайли. Основне завдання, яке вирішує розроблювана і ІСПМТ – перетворення україномовного аудіо на письмовий текст. Користувач може завантажити або записати, за допомогою мікрофона, аудіофайл, який система трансформує у письмовий україномовний текст та покаже його на відповідній сторінці. Додатковим завданням є збереження та перегляд списку завантажених аудіофайлів для користувачів, які зареєструвались у системі. Також при цьому доступний список можливих транслювань у письмовий текст. Особливістю ІС є те, що вона створена як вебзастосунок, що забезпечує легкий доступ до неї з будь-якого пристрою, підключеного до інтернету. Також, на відміну від багатьох рішень, цей застосунок містить два варіанти створення аудіофайла для його подальшого опрацювання, що полегшує користування. ІС доступна цілодобово, адже міститься на хмарному сервері, що також підвищує її надійність. Інтерфейс ІС є інтуїтивним, тому для її використання не потрібно особливих знань. Функціонал списку завантажених аудіофайлів забезпечує не лише перегляд, а й завантаження файлів.

Використовувати ІС можливо, підключившись до глобальної мережі інтернет, а також за наявності браузера нових версій. Для функціоналу запису аудіо користувач повинен надати дозвіл на використання мікрофона. Також для використання ІС рекомендовано встановити хороший мікрофон та чітко вимовляти слова. Те саме стосується файла, який користувач буде завантажувати, він повинен бути якісним та з чіткою вимовою. Застосунок протестовано у нових веббраузерах, а саме: Google Chrome 101.0.4951.54, Microsoft Edge 101.0.1210.39. В інших веббраузерах може відрізнятись інтерфейс сайта. Після того, як користувач відкриє у веббраузері основну сторінку, йому буде запропоновано два варіанти завантаження аудіофайла. Для першого варіанта (рис. 15) наявне одне поле для завантаження файла та кнопка "Upload", після натискання якої файл надсилається на сервер та аналізується. Для другого варіанта потрібно натиснути на кнопку "Record", що відкриє другий

елемент таблиці, в якому буде кнопка початку запису аудіо (рис. 16). Для початку запису аудіо необхідно натиснути кнопку “Start Audio Recording”, а щоб закінчити – кнопку “Stop Audio Recording” (рис. 16).

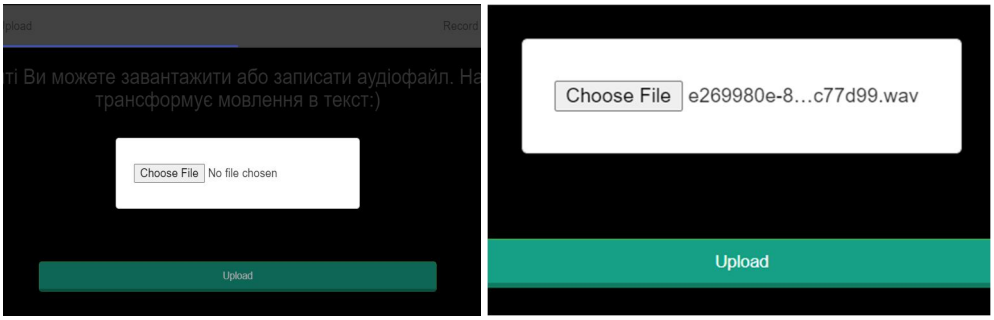


Рис. 15. Сторінка завантаження файла та вигляд сторінки після вибору файла

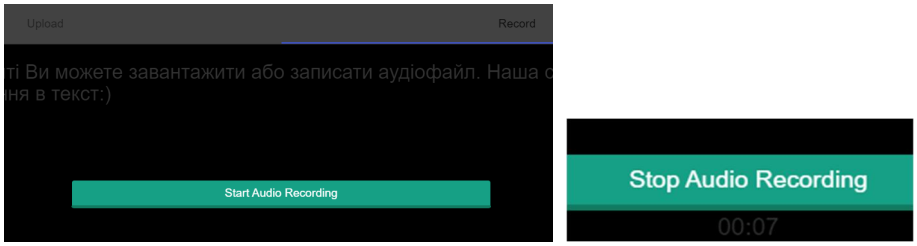


Рис. 16. Сторінка запису файла та вигляд сторінки після початку запису аудіофайла

ІСПМТ закінчить запис аудіофайла та створить два поля (рис. 17). За допомогою першого поля можна прослухати запис або завантажити його. В другому полі користувач повинен ввести назву записаного файла. Якщо залишити це поле пустим, то система сама згенерує назву на з хеш-коду аудіофайла. В обох випадках доступна кнопка “Upload audio”, після натискання якої ІС відправить дані на сервер, де відбуватиметься їх опрацювання. Далі користувача спрямовують на нову сторінку (рис. 17), де вказано назву файла та таблицю можливого текстового змісту аудіофайла. Користувачеві доступна сторінка із реєстрацією (рис. 18), що містить два поля, де потрібно ввести логін та пароль, який надалі використовуватиметься для входу в систему.

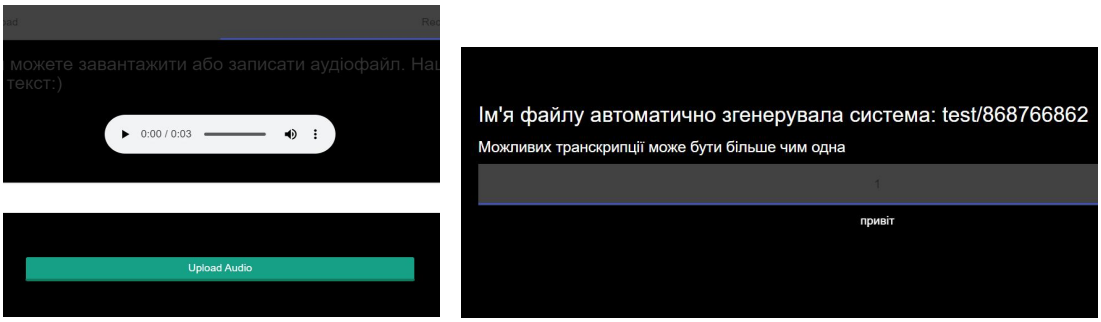


Рис. 17. Вигляд сторінки після закінчення запису аудіофайла та з інформацією про аудіофайл

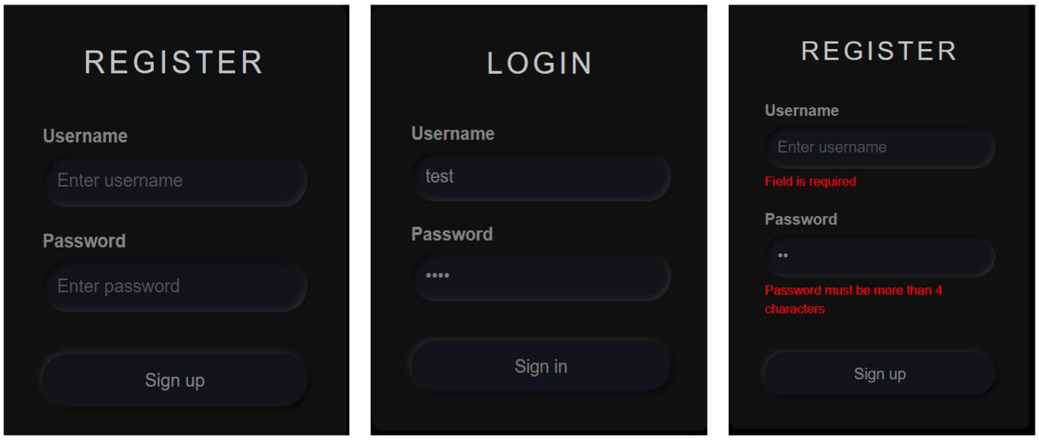


Рис. 18. Сторінка реєстрації, авторизації та приклад перевірки полів на сторінці реєстрації

Сторінка авторизації (рис. 18) містить два поля: логін, пароль, а також кнопку “Sign in”, після натискання якої користувача буде скеровано на головну сторінку. ІС при цьому згенерує jwt токен, який надає доступ до сторінки зі списком завантажених аудіофайлів. Також створено функціонал перевірки полів на правильність вмісту (рис. 18). Якщо користувач введе невалідні дані, то ІС проінформує його про це. ІС перевіряє поле “username” на наявність будь-якого тексту, а поле “password” на наявність будь-якого тексту, а також на розмір рядка, який повинен містити більше ніж чотири символи. Для того, щоб відкрити сторінку зі списком аудіо, програма додасть у хедер нову кнопку “Your list” (рис. 19), після натискання якої користувача буде скеровано на нову сторінку.

Сторінку зі списком усіх файлів, які завантажив користувач, умовно поділено на дві частини. Перша частина, розташована справа, містить кнопку “Your audio files” (рис. 19), після натискання якої ІС покаже список із назвами усіх файлів (рис. 19). Для перегляду інформації потрібно натиснути на будь-яку назву зі списку, після чого ІС продемонструє дані посередині сторінки (рис. 20): назву аудіофайла та можливі транскрипції.

Для виходу із вебзастосунку потрібно натиснути кнопку “Logout”, яка також міститься в хедері програми (рис. 20). Після цього користувача буде скеровано на нову сторінку (рис. 20), де є текстове поле з інформацією, а сам хедер також буде змінено і він міститиме нові кнопки (рис. 20): “Login”, “Register”, “Upload audio”. З метою глибшого аналізу та виявлення залежностей проаналізовано взаємозв’язок точності опрацювання та кількості слів, а також швидкості опрацювання файла залежно від кількості слів у ньому. Під час аналізу використано текст диктанту з української мови “Народження традиції”, який містить 185 слів та велику кількість складних слів із різними наголосами, що ускладнює перевірку. Для кожного з результатів поетапно протестовано аудіофайл із різною кількістю слів, а саме: 13, 36, 72, 96, 108, 140, 185 слів. У результаті першого аналізу (рис. 21, а) визначено середній відсоток точності 90 %. Кількість слів ніяк не впливає на точність алгоритму, а зменшення відсотка невелике і спричинене складністю слів та низькою якістю мікрофона, а отже, і записуваного файла.

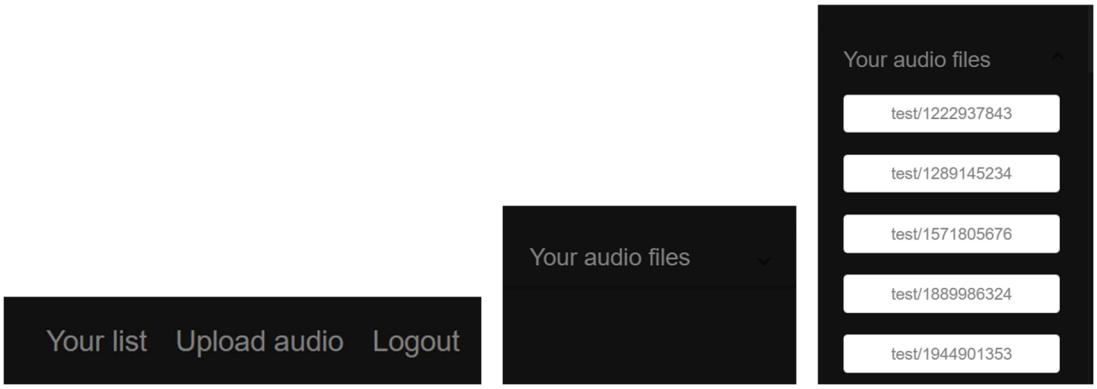


Рис. 19. Вигляд хедера після логіна, сторінки до натискання кнопки "Your audio files" та сторінки після натискання "Your audio files"

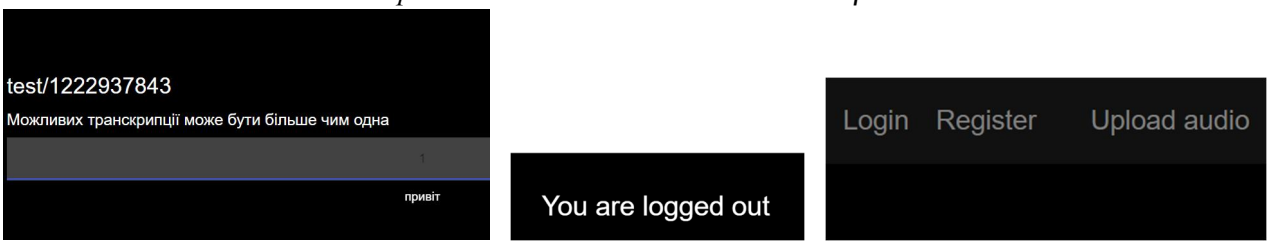
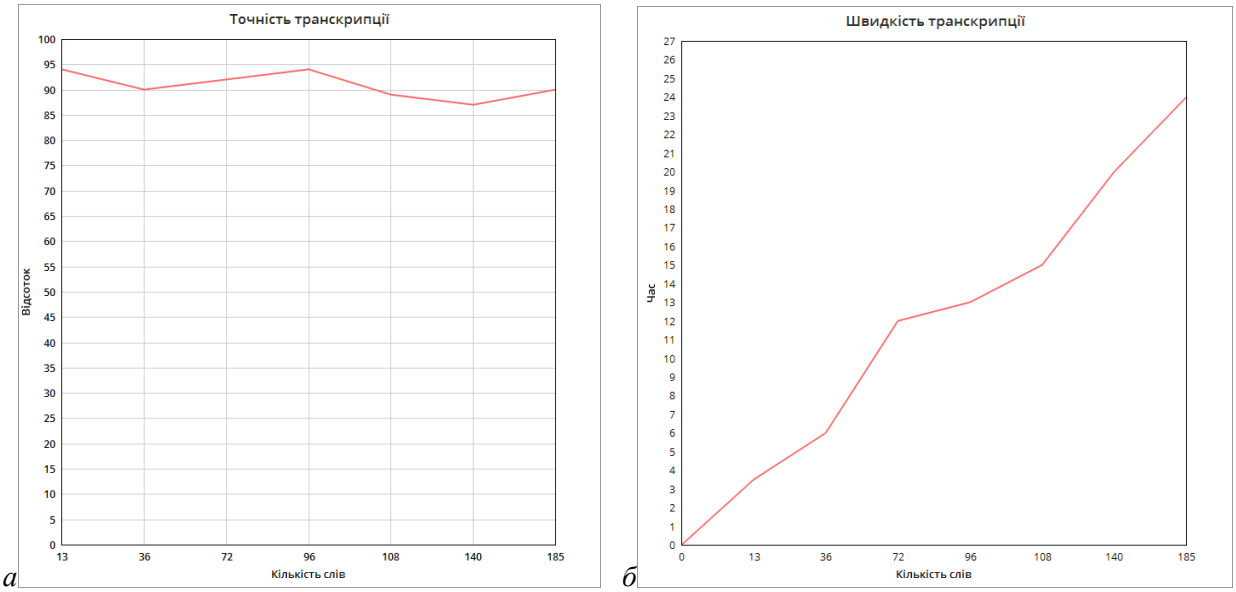


Рис. 20. Вигляд сторінки після натискання на одну з назв, після виходу та хедеру після виходу



а *б*

Рис. 21. Графік залежності (а) точності від кількості слів та (б) швидкості від кількості слів

За результатами другого аналізу залежності швидкості опрацювання аудіофайла від кількості слів у ньому (рис. 21, б) можна зробити висновок, що хоча ІСПМТ і повертає непогані результати, але її швидкість є перешкодою. Особливо це стосується великих файлів, адже їх опрацювання триває >20 секунд, а отже, в майбутньому для покращення результатів потрібно збільшувати характеристики хмарного сервера і самого алгоритму опрацювання.

Висновки

Основне завдання роботи – розроблення ІС для перетворення звукового україномовного тексту на письмовий. У результаті для створення вибрано інформаційно-аналітичний тип системи. Проаналізовано перетворення мовлення на текст, сфери використання, а також алгоритми. Окрім

цього, розглянуто готові рішення: Google Speech-to-text, IBM Watson, Dragon NaturallySpeaking, Microsoft Dictate та Odrey. На основі цього створено таблицю порівняння та вибрано вимоги до розроблюваної ІСПМТ. Здійснено проєктування ІСПМТ. Спочатку побудовано дерево цілей та на його основі розраховано матриці попарних порівнянь та вибрано тип ІС, а саме інформаційно-аналітичну. Після цього побудовано контекстну діаграму, дочірню діаграму, яка створена за допомогою декомпозиції та діаграми підпроцесів для кожного процесу в дочірній діаграмі. Проаналізовано найпопулярніші браузери, мови програмування, фреймворки, БД, системи контролю версій, системи розгортання та шаблони проєктування, після цього вибрано найдоцільніші для розроблення ІСПМТ. Вирішено, що система розроблятиметься як вебзастосунок із використанням Java, Spring, Spring Data, Google Cloud Storage, PostgreSQL, Junit, Mockito, AssertJ, Angular, GIT, Docker, Kubernetes та шаблон проєктування MVC. Реалізовано ІСПМТ, а також здійснено опис структури БД, функціонального призначення, логічної структури, технічних засобів, вхідних та вихідних даних. Також розглянуто особливості та функціональні обмеження ІСПМТ. Проаналізовано залежності точності від кількості слів та швидкості опрацювання від кількості слів. Результатом роботи є протестована та готова до використання інформаційна система перетворення україномовного звукового тексту на письмовий, яка надає такі основні функції: завантаження аудіофайла із системи; запис аудіофайла із використанням мікрофона; опрацювання та збереження аудіофайла; перегляд результатів транскрипції.

Список літератури

1. Драган, Я., Джичка, Н. (2010). Виявлення патології голосу на основі статистичної обробки голосних україномовних дикторів. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 686. С. 250–254.

2. Tymoshenko, K., Vysotska, V., Kovtun, O., Holoshchuk, R., Holoshchuk. S. (2021). Real-time Ukrainian text recognition and voicing. *CEUR Workshop Proceedings*, No. 2870, 357–387.

3. Tymoshenko, K. Vysotska, V. (2020). Algorithm of Text Recognizing in Ukrainian on the Video Mode. *Computational linguistics and intelligent systems* : proceedings of the 4nd International conference, 23–24 April 2020, Lviv, Ukraine, 81–89.

4. Dmytriv, A., Vysotska, V., Bublyk, M. (2021). The Speech Parts Identification for Ukrainian Words Based on VESUM and Horokh Using. *Computer Sciences and Information Technologies* (CSIT): proceedings of the IEEE 16th International Conference, 22–25 Sept. 2021, Lviv, Ukraine, 21–33. DOI: 10.1109/CSIT52700.2021.9648813.

5. Dmytriv, A., Holoshchuk, S., Chyrun, L., Holoshchuk, R. (2022). Comparative Analysis of Using Different Parts of Speech in the Ukrainian Texts Based on Stylistic Approach. *CEUR Workshop Proceedings*, Vol. 3171, 546–560.

6. Kubinska, S., Vysotska, V., Matseliukh, Y. (2021). User Mood Recognition and Further Dialog Support. *Computer Sciences and Information Technologies* (CSIT): proceedings of the IEEE 16th International Conference, 22–25 Sept. 2021, Lviv, Ukraine, 34–39. DOI: 10.1109/CSIT52700.2021.9648610.

7. Kubinska, S., Holoshchuk, R., Holoshchuk, S., Chyrun, L. (2022). Ukrainian Language Chatbot for Sentiment Analysis and User Interests Recognition based on Data Mining. *CEUR Workshop Proceedings*, Vol. 3171, 315–327.

8. Dyriv, A., Andrunyk, V., Burov, Y., Karpov, I., Chyrun, L. (2021). The user's psychological state identification based on Big Data analysis for person's electronic diary. *Computer science and information technologies*: proceedings of IEEE 16th International conference on computer science and information technologies. Lviv, Ukraine, 22–25 September, 2021, 101–112. DOI: 10.1109/CSIT52700.2021.9648810.

9. Berko, A., Matseliukh, Y., Ivaniv, Y., Chyrun, L., Schuchmann, V. (2021). The text classification based on Big Data analysis for keyword definition using stemming. *Computer science and information technologies*: proceedings of IEEE 16th International conference on computer science and information technologies. Lviv, Ukraine, 22–25 September, 2021, 184–188. 10.1109/CSIT52700.2021.9648764.

10. Aksonov, D., Gozhyj, A., Kalinina, I., Vysotska, V. (2021). Question-Answering Systems Development Based on Big Data Analysis. *Computer Sciences and Information Technologies* (CSIT): proceedings of the IEEE 16th International Conference, 22–25 Sept. 2021, Lviv, Ukraine, 113–118. DOI: 10.1109/CSIT52700.2021.9648631.

11. Лозицький, О. А. (2015). Прикладна програмна система опрацювання україномовних технічних текстів для людей з вадами зору. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 832. С. 315–331.

12. Лозицький, О. А., Кунанець, Н. Е. (2014). Система опрацювання технічних текстів українською мовою

з метою їх адаптації для людей з вадами зору. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 805. С. 316–324.

13. Лозицький, О. А., Пасічник, В. В. (2010). Комп'ютерні засоби освітніх процесів для людей з вадами зору. Аналітичний огляд. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 673. С. 325–339.

14. Кунанець, Н. Е., Лозицький, О. А., Пасічник, В. В. (2011). Організація освітніх та інформаційних процесів для людей з вадами зору із застосуванням спеціальних. *Інноваційні комп'ютерні технології у вищій школі* : матеріали 3-ї Науково-практичної конференції, 8–12 жовтня 2011 року, Львів. С. 156–159.

15. Лозицький, О. А., Пасічник, В. В. (2010). Стандарти, структура та технологія створення книг, що "розмовляють". *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 689. С. 281–294.

16. Кунанець, Н. Е., Лозицький, О. А., Пасічник, В. В. (2016). Інформаційні технології озвучування українською мовою математичних формул для осіб з вадами зору. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 843. С. 84–93.

17. Давидов, М. (2013). Синтез видимої артикуляції віртуального персонажа з аудіопотоку для системи сурдоперекладу. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 771. С. 94–100.

18. Крак Ю. В., Лозинська О. В., Пасічник В. В., Тернов А. С., Шкільнюк, Д. В. (2016). Математичні методи та прикладні інформаційні технології моделювання, перекладу та навчання для української жестової мови. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 854. С. 210–227.

19. Чабан, В. (2007). Два штрихи до українського правопису. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 593. С. 103–105.

20. Кунанець, Н. Е., Малиновський, О. Б. (2011). Інформаційно-мультимедійний продукт у бібліотеках. *Сучасні проблеми діяльності бібліотеки в умовах інформаційного суспільства*: матеріали третьої науково-практичної конференції, 29 вересня 2011 року, Львів. С. 225–229.

21. Dovbysh, A., Alieksieiev, V. (2018). Embedding speech recognition tools for custom software: Engines Overview. *Computational linguistics and intelligent systems* : proceedings of the 2nd International conference, 25–27 June 2018, Lviv, Ukraine. P. 114–121.

22. Lobur, M., Romaniuk, A., Romanyshyn, M. (2012). Defining an approach for deep sentiment analysis of reviews in Ukrainian. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 747. С. 124–130.

23. Romaniuk, A., Romanyshyn, M. (2013). Named-entity recognition for sentiment analysis of Ukrainian reviews. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 777. С. 83–86.

24. Kotsyba, N. (2013). Overview of the Ukrainian language resources within the multilingual European MULTEXT-East project. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 770. С. 122–129.

25. Palinska, O., Kaczala, O. (2013). Regional dialect of modern Lviv: language-contact processes. *Гуманітарні та соціальні науки*: матеріали IV Міжнародної конференції молодих вчених HSS-2013, 21–23 листопада 2013 року, Львів, Україна. С. 66–71.

26. Boiko, D. (2020). Using of Natural Language Processing in Chatbot. *Computational linguistics and intelligent systems* : proceedings of the 4nd International conference, 23–24 April 2020, Lviv, Ukraine. P. 410–415.

27. Басюк, Т. М., Василюк, А. С. (2019). Просування інтернет-ресурсів з використанням технологій голосового пошуку. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 5. С. 3–13. DOI: 10.23939/sisn2019.01.003.

28. Шевчук, Р. П. (2013). Ідентифікація та виконання голосових команд персональними мобільними помічниками із використанням продукційної моделі представлення знань. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 773. С. 143–150.

29. Васильцов, І. В., Карпінський, М. П., Кавка, С. Б. (2003). Структура системи аутентифікації суб'єктів за голосом. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 471. С. 144–148.

30. Hnatyuk, M. (2013). Prevailing tendencies of North Lemkian resettled dialects in Western Ukraine: phonetic aspect. *Гуманітарні та соціальні науки*: матеріали IV Міжнародної конференції молодих вчених HSS-2013, 21–23 листопада 2013 року, Львів, Україна. С. 78–79.

31. Галич, Ю. (2012). Порівняльний аналіз сучасних систем розпізнавання мови. 70–та студентська науково–технічна конференція: збірник тез доповідей, жовтень – листопад 2012 року, Нац. ун-т "Львівська політехніка". С. 198–199.

32. Nyzhnyk, O., Burov, Y., Zavushchak, I. (2020). Intelligent Climate Control System in Office Space. *Computational linguistics and intelligent systems*: proceedings of the 4nd International conference, 23–24 April 2020, Lviv, Ukraine. P. 349–351.

33. Рашкевич, Ю., Шиманьські, З., Фігура, Р. (2010). Динаміка зміни тривалостей структурних елементів дифтонгів польської мови у різних темпах вимови. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 672. С. 211–214.

34. Gadek, J. (2005). The database of emotional speech. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 534.

С. 165–172.

35. Дацишин, Х. (2018). Можливості прямої мови у відтворенні усного мовлення в друкованому медіатексті. *Вісник Нац. ун-ту "Львівська політехніка"*. Вип. 896. С. 145–149.

36. Warren, E. (2018). The 44 Phonemes in English. URL: https://www.dyslexia-reading-well.com/44-phonemes-in-english.html.

37. The Past, Present, and Future of Speech-to-Text and AI Transcription (2022). URL: https://imerit.net/blog/the-past-present-and-future-of-speech-to-text-and-ai-transcription-all-una/.

38. Innovative Uses of Speech Recognition Today. (2021). URL: https://summalinguae.com/language-technology/innovative-uses-of-speech-recognition/.

39. Tebelskis, J. (1995). Speech Recognition using Neural Networks. URL: https://isl.anthropomatik.kit.edu/pdf/Tebelskis1995.pdf.

40. Gupta, T. (2017). Deep Learning: Feedforward Neural Network. URL: https://towardsdatascience.com/deep-learning-feedforward-neural-network-26a6705dbdc7.

41. Recurrent Neural Networks (2022). URL: https://www.ibm.com/cloud/learn/recurrent-neural-networks.

42. Google Cloud Speech-to-text (2022). URL: https://cloud.google.com/speech-to-text.

43. IBM Cloud Watson Speech-to-text (2022). URL: https://www.ibm.com/cloud/watson-speech-to-text.

44. Microsoft Dictate (2022). URL: https://www.microsoft.com/en-us/garage/profiles/dictate/.

45. Odrey (2022). URL: https://odreyapp.com/.

46. Кустовська, О. В. (2005). Методологія системного підходу та наукових досліджень. Тернопіль: Економічна думка. 124 с.

47. Шершньова, З. Є. (2004). Стратегічне управління. Київ: КНЕУ. 221 с.

48. Швиданенко, Г., Ревуцька, Н. (2013). Формування бізнес-моделі підприємства. Київ: КНЕУ. 198 с.

49. StatCounter Global Stats (2022). Browser Market Share Worldwide Apr 2021 – Apr 2022. URL: https://gs.statcounter.com/browser-market-share.

50. Most used programming languages among developers worldwide, as of 2021 (2022). URL: https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/.

51. Shan, P. (2014). Node.js – reasons to use, pros and cons, best practices! URL: https://www.voidcanvas.com/describing-node-js/.

52. Walls, C. (2014). Spring Boot in Action. New York: Manning Publications, 2014.

53. Nader, Y. (2022). What is Django? Advantages and Disadvantages. URL: https://hackr.io/blog/what-is-django-advantages-and-disadvantages-of-using-django.

54. Express.js Mobile App Development: Pros and Cons for Developers (2022). URL: https://apiko.com/blog/express-mobile-app-development/.

55. Pollack M., Gierke O., Risberg T. et al. (2012). Spring Data: Modern Data Access for Enterprise Java. Sebastopol, California: O'Reilly Media, 2012.

56. Google Cloud Storage (2022). URL: https://cloud.google.com/storage.

57. Chason, S., Straub, B. (2014). Pro Git. New York: Apress. 25 с.

58. MVC Pattern (2022). URL: https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm.

59. JetBrains Intellij Idea (2022). URL: https://www.jetbrains.com/idea/.

60. Пасічник, В. В., Резніченко, В. А. (2006). Організація баз даних та знань. Київ: BHV ПИТЕР. 384 с.

References

1. Dragan, Ya., Dzhychka, N. (2010). Detection of voice pathology on the basis of statistical processing of vocal Ukrainian announcers. *Bulletin of Lviv Polytechnic National University*, No. 686, 250–254.

2. Tymoshenko, K., Vysotska, V., Kovtun, O., Holoshchuk, R., Holoshchuk. S. (2021). Real-time Ukrainian text recognition and voicing. *CEUR Workshop Proceedings*, No. 2870, 357–387.

3. Tymoshenko, K. Vysotska, V. (2020). Algorithm of Text Recognizing in Ukrainian on the Video Mode. *Computational linguistics and intelligent systems*: proceedings of the 4nd International conference, 23–24 April 2020, Lviv, Ukraine, 81–89.

4. Dmytriv, A., Vysotska, V., Bublyk, M. (2021). The Speech Parts Identification for Ukrainian Words Based on VESUM and Horokh Using. *Computer Sciences and Information Technologies* (CSIT): proceedings of the IEEE 16th International Conference, 22–25 Sept., Lviv, Ukraine. 2021, 21–33. DOI: 10.1109/CSIT52700.2021.9648813.

5. Dmytriv, A., Holoshchuk, S., Chyrun, L., Holoshchuk, R. (2022). Comparative Analysis of Using Different Parts of Speech in the Ukrainian Texts Based on Stylistic Approach. *CEUR Workshop Proceedings*, Vol. 3171, 546–560.

6. Kubinska, S., Vysotska, V., Matseliukh, Y. (2021). User Mood Recognition and Further Dialog Support.

Computer Sciences and Information Technologies (CSIT): proceedings of the IEEE 16th International Conference, 22–25 Sept. 2021, Lviv, Ukraine, 34–39. DOI: 10.1109/CSIT52700.2021.9648610.

7. Kubinska, S., Holoshchuk, R., Holoshchuk, S., Chyrun, L. (2022). Ukrainian Language Chatbot for Sentiment Analysis and User Interests Recognition based on Data Mining. *CEUR Workshop Proceedings*, Vol. 3171, 315–327.

8. Dyriv, A., Andrunyk, V., Burov, Y., Karpov, I., Chyrun, L. (2021). The user's psychological state identification based on Big Data analysis for person's electronic diary. *Computer science and information technologies* : proceedings of IEEE 16th International conference on computer science and information technologies. Lviv, Ukraine, 22–25 September, 2021, 101–112. DOI: 10.1109/CSIT52700.2021.9648810.

9. Berko A., Matseliukh Y., Ivaniv Y., Chyrun L., Schuchmann V. (2021). The text classification based on Big Data analysis for keyword definition using stemming. *Computer science and information technologies*: proceedings of IEEE 16th International conference on computer science and information technologies. Lviv, Ukraine, 22–25 September, 2021, 184–188. DOI: 10.1109/CSIT52700.2021.9648764.

10. Aksonov, D., Gozhyj, A., Kalinina, I., Vysotska, V. (2021). Question-Answering Systems Development Based on Big Data Analysis. *Computer Sciences and Information Technologies* (CSIT): proceedings of the IEEE 16th International Conference, 22–25 Sept. 2021, Lviv, Ukraine, 113–118. DOI: 10.1109/CSIT52700.2021.9648631.

11. Lozytskyi, O. A. (2015). Applied software system for processing Ukrainian-language technical texts for people with visual impairments. *Bulletin of Lviv Polytechnic National University*, No. 832, 315–331.

12. Lozytskyi, O. A., Kunanets, N. E. (2014). A system for processing technical texts in the Ukrainian language with the aim of adapting them for people with visual impairments. *Bulletin of Lviv Polytechnic National University,* No. 805, 316–324.

13. Lozytskyi, O. A., Pasichnyk, V. V. (2010). Computer tools of educational processes for visually impaired people. Analytical review. *Bulletin of Lviv Polytechnic National University,* No. 673, 325–339.

14. Kunanets, N. E., Lozytskyi, O. A., Pasichnyk, V. V. (2011). Organization of educational and informational processes for people with visual impairments with the use of special. Innovative computer technologies in higher education: materials of the 3rd Scientific and Practical Conference, October 8–12, 2011, Lviv, 156–159.

15. Lozytskyi, O. A., Pasichnyk, V. V. (2010). Standards, structure and technology for creating "talking" books. *Bulletin of Lviv Polytechnic National University*, No. 689, 281–294.

16. Kunanets, N. E., Lozytskyi, O. A., Pasichnyk, V. V. (2016). Information technologies for voicing mathematical formulas in Ukrainian for people with visual impairments. *Bulletin of Lviv Polytechnic National University*, No. 843, 84–93.

17. Davydov, M. (2013). Synthesis of visible articulation of a virtual character from an audio stream for a sign language translation system. *Bulletin of Lviv Polytechnic National University*, No. 771, 94–100.

18. Krak Y. V., Lozinska O. V., Pasichnyk V. V., Ternov A. P., Shkilniuk, D. V. (2016). Mathematical methods and applied information technologies of modeling, translation and teaching for Ukrainian sign language. *Bulletin of Lviv Polytechnic National University*, No. 854, 210–227.

19. Chaban, V. (2007). Two touches to Ukrainian spelling. *Bulletin of Lviv Polytechnic National University*, No. 593, 103–105.

20. Kunanets, N. E., Malinovskyi, O. B. (2011). Information and multimedia product in libraries. Modern problems of library activity in the conditions of the information society: materials of the third scientific and practical conference, September 29, 2011, Lviv, 225–229.

21. Dovbysh, A., Alieksieiev, V. (2018). Embedding speech recognition tools for custom software: Engines Overview. *Computational linguistics and intelligent systems* : proceedings of the 2nd International conference, 25–27 June 2018, Lviv, Ukraine, 114–121.

22. Lobur, M., Romaniuk, A., Romanyshyn, M. (2012). Defining an approach for deep sentiment analysis of reviews in Ukrainian. *Bulletin of Lviv Polytechnic National University*, No. 747,124–130.

23. Romaniuk, A., Romanyshyn, M. (2013). Named-entity recognition for sentiment analysis of Ukrainian reviews. *Bulletin of Lviv Polytechnic National University*, No. 777, 83–86.

24. Kotsyba, N. (2013). Overview of the Ukrainian language resources within the multilingual European MULTEXT-East project. *Bulletin of Lviv Polytechnic National University*, No. 770, 122–129.

25. Palinska, O., Kaczala, O. (2013). Regional dialect of modern Lviv: language-contact processes. *Humanities and social sciences: materials of the IV International Conference of Young Scientists HSS-2013, November 21–23, 2013, Lviv, Ukraine*, 66–71.

26. Boiko, D. (2020). Using of Natural Language Processing in Chatbot. *Computational linguistics and intelligent systems*: proceedings of the 4nd International conference, 23–24 April 2020, Lviv, Ukraine, 410–415.

27. Basyuk, T. M., Vasylyuk, A. P. (2019). Promotion of Internet resources using voice search technologies. *Bulletin of Lviv Polytechnic National University*, No. 5, 3–13. DOI: 10.23939/sisn2019.01.003.

28. Shevchuk, R. P. (2013). Identification and execution of voice commands by personal mobile assistants using a production model of knowledge representation. *Bulletin of Lviv Polytechnic National University*, No. 773, 143–150.

29. Vasyltsov, I. V., Karpinsky, M. P., Kavka, P. B. (2003). The structure of the system of authentication of subjects by voice. *Bulletin of Lviv Polytechnic National University*, No. 471, 144–148.

30. Hnatyuk, M. (2013). Prevailing tendencies of North Lemkian resettled dialects in Western Ukraine: phonetic aspect. *Humanities and social sciences: materials of the IV International Conference of Young Scientists HSS-2013, November 21–23, 2013, Lviv, Ukraine*, 78–79.

31. Halych, Yu. (2012). Comparative analysis of modern speech recognition systems. 70th student scientific and technical conference: collection of theses of reports, October – November 2012, Lviv Polytechnic National University, 198–199.

32. Nyzhnyk, O., Burov, Y., Zavushchak, I. (2020). Intelligent Climate Control System in Office Space. *Computational linguistics and intelligent systems* : proceedings of the 4nd International conference, 23–24 April 2020, Lviv, Ukraine, 349–351.

33. Rashkevich, Yu., Szymanski, Z., Figura, R. (2010). Dynamics of changes in the durations of structural elements of Polish diphthongs at different pronunciation rates. *Bulletin of Lviv Polytechnic National University*. No. 672, 211–214.

34. Gadek, J. (2005). The database of emotional speech. *Bulletin of Lviv Polytechnic National University*. No. 534, 165–172.

35. Dacyshyn, H. (2018). Possibilities of direct speech in reproduction of oral speech in printed media text. *Bulletin of Lviv Polytechnic National University*, No. 896, 145–149.

36. Warren, E. (2018). The 44 Phonemes in English. URL: https://www.dyslexia-reading-well.com/44-phonemes-in-english.html.

37. The Past, Present, and Future of Speech-to-Text and AI Transcription (2022). URL: https://imerit.net/blog/the-past-present-and-future-of-speech-to-text-and-ai-transcription-all-una/.

38. Innovative Uses of Speech Recognition Today (2021). URL: https://summalinguae.com/language-technology/innovative-uses-of-speech-recognition/.

39. Tebelskis, J. (1995). Speech Recognition using Neural Networks. URL: https://isl.anthropomatik.kit.edu/pdf/Tebelskis1995.pdf.

40. Gupta, T. (2017). Deep Learning: Feedforward Neural Network. URL: https://towardsdatascience.com/deep-learning-feedforward-neural-network-26a6705dbdc7.

41. Recurrent Neural Networks (2022). URL: https://www.ibm.com/cloud/learn/recurrent-neural-networks.

42. Google Cloud Speech-to-text (2022). URL: https://cloud.google.com/speech-to-text.

43. IBM Cloud Watson Speech-to-text (2022). URL: https://www.ibm.com/cloud/watson-speech-to-text.

44. Microsoft Dictate. (2022). URL: https://www.microsoft.com/en-us/garage/profiles/dictate/.

45. Odrey (2022). URL: https://odreyapp.com/.

46. Kustovska, O. V. (2005). System approach methodology and scientific research. Ternopil: Economic thought.

47. Shershnyova, Z. E. (2004). Strategic management. Kyiv: KNEU. 221 p.

48. Shvydanenko, G., Revutska, N. (2013). Formation of the business model of the enterprise. Kyiv: KNEU.

49. StatCounter Global Stats (2022). Browser Market Share Worldwide Apr 2021 – Apr 2022. URL: https://gs.statcounter.com/browser-market-share.

50. Most used programming languages among developers worldwide, as of 2021 (2022). URL: https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/.

51. Shan, P. (2014). Node.js – reasons to use, pros and cons, best practices! URL: https://www.voidcanvas.com/describing-node-js/.

52. Walls, C. (2014). Spring Boot in Action. New York: Manning Publications, 2014.

53. Nader, Y. (2022). What is Django? Advantages and Disadvantages. URL: https://hackr.io/blog/what-is-django-advantages-and-disadvantages-of-using-django.

54. Express.js Mobile App Development: Pros and Cons for Developers (2022). URL: https://apiko.com/blog/express-mobile-app-development/.

55. Pollack M., Gierke O., Risberg T. et al. (2012). Spring Data: Modern Data Access for Enterprise Java. Sebastopol, California: O'Reilly Media, 2012.

56. Google Cloud Storage (2022). URL: https://cloud.google.com/storage.

57. Chason, S., Straub, B. (2014). Pro Git. New York: Apress. 25 p.

58. MVC Pattern (2022). URL: https://www.tutorialspoint.com/design_pattern/mvc_pattern.htm.

59. JetBrains Intellij Idea (2022). URL: https://www.jetbrains.com/idea/.

60. Pasichnyk, V. V., Reznichenko, V. A. (2006). Organization of databases and knowledge. Kyiv: BHV PITER.

INFORMATION SYSTEM FOR CONVERTING AUDIO IN UKRAINIAN LANGUAGE INTO ITS TEXTUAL REPRESENTATION USING NLP METHODS AND MACHINE LEARNING

Yurii Tyshchuk[1], Victoria Vysotska[1, 2], Olha Vlasenko[2, 3]

[1] Lviv Polytechnic National University, Information Systems and Networks Department, , 12, S. Bandera str., Lviv, Ukraine
[2] Osnabrück University, Institute of Computer Science, 1, Friedrich-Janssen-Str., Osnabrück, Germany
[3] Zhytomyr Ivan Franko State University, Professional and Pedagogical, Apecial Education, Andragogy and Management Department, 40, Velyka Berdychivska str., Zhytomyr, Ukraine
E-mail: yurii.tyshchuk.knm.2018@lpnu.ua, ORCID: 0000-0002-1083-3765
E-mail: Victoria.A.Vysotska@lpnu.ua, ORCID: 0000-0001-6417-3689
E-mail: olha.vlasenko@uni-osnabrueck.de, ORCID: 0000-0001-7258-2108

Speech recognition involves various models, methods and algorithms for analysing and processing the user's recorded voice. This allows people to control different systems that support one type of speech recognition. A speech-to-text conversion system is a type of speech recognition that uses spoken data for further processing. It also provides several stages for processing an audio file, which uses electroacoustic means, filtering algorithms in the audio file to isolate relevant sounds, electronic data arrays for the selected language, as well as mathematical models that make up the most likely words from phonemes. Thanks to the conversion of speech to text, people whose professions are closely related to typing a large amount of text on the keyboard, significantly speed up and facilitate the work process, as well as reduce the amount of stress. In addition, such systems help businesses, because the concept of remote work is becoming more and more popular, and therefore companies need tools to record and systematize meetings in the form of written text. The object of the research is the process of converting the Ukrainian-language text into a written one based on NLP and machine learning methods. The subject of the research is file processing algorithms for extracting relevant sounds and recognizing phonemes, as well as mathematical models for recognizing an array of phonemes as specific words. The purpose of the work is to design and develop an information system for converting audio Ukrainian-language text into written text based on the Ukrainian Speech-to-text Web application, which is a technology for accurate and easy analysis of Ukrainian-language audio files and their subsequent transcription into text. The application supports downloading files from the file system and recording using the microphone, as well as saving the analysed data. The article also describes the stages of design and the general typical architecture of the corresponding system for converting audio Ukrainian-language text into written text. According to the results of the experimental testing of the developed system, it was found that the number of words does not affect the accuracy of the conversion algorithm, and the decrease in percentage is not large and occurred due to the complexity of the words and the low quality of the microphone, and therefore the recorded file.

Key words: text-to-speech; speech recognition; Ukrainian-language; information system.