ARTICLE
DOI 10.36074/grail-of-science.20.02.2026.116

РОЗРОБКА ПРОЕКТУ ІНТЕЛЕКТУАЛЬНОЇ ГІБРИДНОЇ СИСТЕМИ З АДАПТИВНОГО АСИСТУВАННЯ ОНЛАЙН-ЗУСТРІЧЕЙ НА БАЗІ NLP МОДЕЛЕЙ ОБРОБКИ ВЕЛИКИХ ДАНИХ

НАУКОВО-ДОСЛІДНА ГРУПА:

Лисий Денис Вікторович
магістрант,
Міжрегіональна академія управління персоналом, Україна

Рудніченко Микола Дмитрович
канд. техн. наук, доцент, доцент кафедри інформаційних технологій
Національний університет «Одеська політехніка», Україна

Шибаєва Наталя Олегівна
канд. техн. наук, доцент, доцент кафедри інформаційних технологій
Національний університет «Одеська політехніка», Україна

Петров Ігор Михайлович
доктор техн. наук, професор, професор кафедри судноводіння
Національний університет «Одеська морська академія», Україна

Шведов Денис Валерійович
аспірант кафедри інформаційних технологій
Національний університет «Одеська політехніка», Україна

Отрадська Тетяна Василівна
канд. техн. наук, доцент, доцент кафедри інформаційних систем
Національний університет «Одеська політехніка», Україна

Анотація. Розроблено архітектуру та реалізовано адаптивний асистент онлайн-зустрічей, здатний здійснювати автоматичне розпізнавання мовлення, семантичний аналіз діалогів та генерацію проактивних підказок у режимі реального часу. Система побудована на мікросервісній архітектурі з потоковою обробкою аудіоданих та забезпечує наскрізну латентність від мовлення до відображення транскрипту не більше двох секунд. Модуль розпізнавання мовлення реалізовано на базі Faster-Whisper з INT8-квантуванням через CTranslate2 та детекцією голосової активності Silero VAD, на чистому аудіо. NLP-конвеєр на основі fine-tuned DistilBERT виконує автоматичне витягування пунктів дій, питань та рішень з транскриптів зустрічей. Модуль

проактивних інтервенцій інтегрує евристичні правила з великими мовними моделями через Ollama/OpenAI API для генерації контекстуальних підказок учасникам під час обговорення. Проведено експериментальну валідацію на записах корпусу AMI та імітованих Zoom-сесіях. Порівняльний аналіз із комерційними аналогами Otter.ai та Fireflies.ai підтвердив унікальність рішення за критеріями підтримки проактивної фасилітації, локального розгортання та повноцінної роботи з українською мовою.
Ключові слова: *адаптивний асистент зустрічей; розпізнавання мовлення; обробка природної мови; витягування пунктів дій; проактивні інтервенції; обробка в реальному часі.*

Вступ

Масовий перехід до дистанційної та гібридної моделі роботи, який прискорився з 2020 року, перетворив онлайн-зустрічі на основний інструмент корпоративної комунікації. За даними досліджень, середній офісний працівник витрачає від 35 до 50 відсотків робочого часу на зустрічі, причому до 70 відсотків цих зустрічей оцінюються як непродуктивні [3] або такі, що не мають чітких результатів. Ключовими проблемами залишаються втрата прийнятих рішень, відсутність фіксації завдань, розмиття відповідальності та неефективне використання часу учасників.

Існуючі інструменти для підтримки онлайн-зустрічей, такі як Otter.ai та Fireflies.ai, зосереджені переважно на постфактум-документуванні: вони надають транскрипцію та короткі підсумки після завершення зустрічі. Проте цей підхід не вирішує фундаментальної проблеми — учасники дізнаються про неефективність зустрічі лише після її закінчення, коли можливість корекції ходу обговорення вже втрачена. Крім того, більшість комерційних рішень працюють виключно через хмарну інфраструктуру, що створює ризики для організацій з підвищеними вимогами до конфіденційності даних, зокрема у фінансовому, медичному та оборонному секторах. Таким чином, актуальною є задача створення системи, здатної не лише документувати, а й активно фасилітувати онлайн-зустрічі в режимі реального часу, виявляючи незакриті питання, відстежуючи прийняття рішень та пропонуючи проактивні підказки учасникам безпосередньо під час обговорення.

Аналіз досліджень та публікацій

Проблематика автоматичної обробки мовлення та аналізу діалогів активно досліджується в рамках кількох взаємопов'язаних напрямків: автоматичне розпізнавання мовлення, обробка природної мови, витягування інформації з діалогів та інтелектуальні асистенти для зустрічей.

У галузі розпізнавання мовлення ключовим досягненням останніх років стала модель Whisper, запропонована Radford [12]. Модель навчена на понад 680 тисячах годин мультимовних даних і демонструє робастність до шуму, акцентів та доменної варіативності без потреби у fine-tuning. Whisper підтримує понад 90 мов, включаючи українську, що робить її привабливою для мультимовних застосувань. Реалізація Faster-Whisper на базі CTranslate2, розроблена Guillaume Klein, досягає прискорення інференсу до 4 разів порівняно з оригінальною імплементацією за рахунок оптимізованих CUDA-ядер та ефективного управління пам'яттю GPU.

Інтеграція великих мовних моделей (LLM) у системи аналізу зустрічей є відносно новим напрямком. Ollama, розроблений як платформа для локального запуску LLM, дозволяє використовувати моделі на кшталт Llama, Mistral та інших без передачі даних у хмару. OpenAI API надає доступ до моделей серії GPT [4] для складніших завдань генерації. Поєднання локальних та хмарних LLM у гібридній архітектурі дозволяє балансувати між конфіденційністю та якістю генерації.

Аналіз літератури та існуючих рішень дозволяє виділити кілька невирішених проблем. По-перше, відсутні системи, що поєднують транскрипцію в реальному часі з проактивною фасилітацією зустрічей. По-друге, існуючі комерційні платформи не підтримують локальне розгортання, що обмежує їх застосування в чутливих до конфіденційності середовищах. По-третє, якість підтримки української мови та сценаріїв code-switching залишається недостатньою для повноцінного використання у вітчизняному бізнес-контексті. Нарешті, інтеграція евристичних правил із нейромережевими моделями для підвищення повноти витягування структурованої інформації з діалогів потребує подальшого дослідження.

Мета роботи

Метою дослідження є розробка та експериментальна валідація адаптивного асистента онлайн-зустрічей, здатного здійснювати автоматичне розпізнавання мовлення, семантичний аналіз діалогів та генерацію проактивних підказок у режимі реального часу на основі мікросервісної архітектури з підтримкою локального розгортання.

Для досягнення поставленої мети необхідно вирішити такі завдання:

— розробити архітектуру мікросервісної системи, що забезпечує потокову обробку аудіоданих з низкою латентністю;

— реалізувати модуль захоплення та препроцесингу аудіо на основі Web Audio API з підтримкою AudioWorklet для обробки в окремому високопріоритетному потоці;

— інтегрувати модель Faster-Whisper з INT8-квантуванням для розпізнавання мовлення з підтримкою української мови та діаризації спікерів;

— розробити NLP-конвеєр на основі DistilBERT [13] для автоматичного витягування пунктів дій, питань та рішень із транскриптів зустрічей;

— реалізувати модуль проактивних інтервенцій з евристичними правилами та інтеграцією LLM для генерації контекстуальних підказок;

— провести експериментальну валідацію системи на реальних даних;

Об’єктом дослідження є процес автоматизованого аналізу та фасилітації онлайн-зустрічей в режимі реального часу. Предметом дослідження є методи та алгоритми розпізнавання мовлення, семантичного аналізу діалогів та генерації проактивних підказок на основі моделей глибокого навчання.

Розробка концепції та архітектури рішення

Концепція адаптивного асистента ґрунтується на real-time обробці аудіопотоків, мікросервісній архітектурі та подієво-орієнтованій інтеграції. Система призначена для автоматичного аналізу діалогів під час віртуальних зустрічей з метою виявлення рішень, завдань (пунктів дій) та надання проактивних підказок без суттєвих затримок. Ключовими вимогами є

модульність для незалежного масштабування та висока точність екстракції контексту. Backend побудований на базі мікросервісів, що взаємодіють через API Gateway та асинхронні черги повідомлень. Такий підхід забезпечує горизонтальне масштабування STT-сервісу [16] та NLP-модулів при зростанні навантаження.

Система приймає на вхід звуковий потік наради та користувацькі конфігурації, що специфікують параметри опрацювання, включно з цільовою мовою транскрибування, ступенем деталізації текстового виводу та категоріями сутностей для ідентифікації. Управлінський шар охоплює двосторонній протокол WebSocket [9] для синхронної передачі інформації, моделі обробки природної мови для виявлення ключових елементів, алгоритми визначення моментів втручання системи та принципи побудови користувацького досвіду.

Діаграма потоків даних нульового рівня, представлена на рис.1, деталізує процес обробки інформації від моменту захоплення аудіо до відображення результатів користувачу.

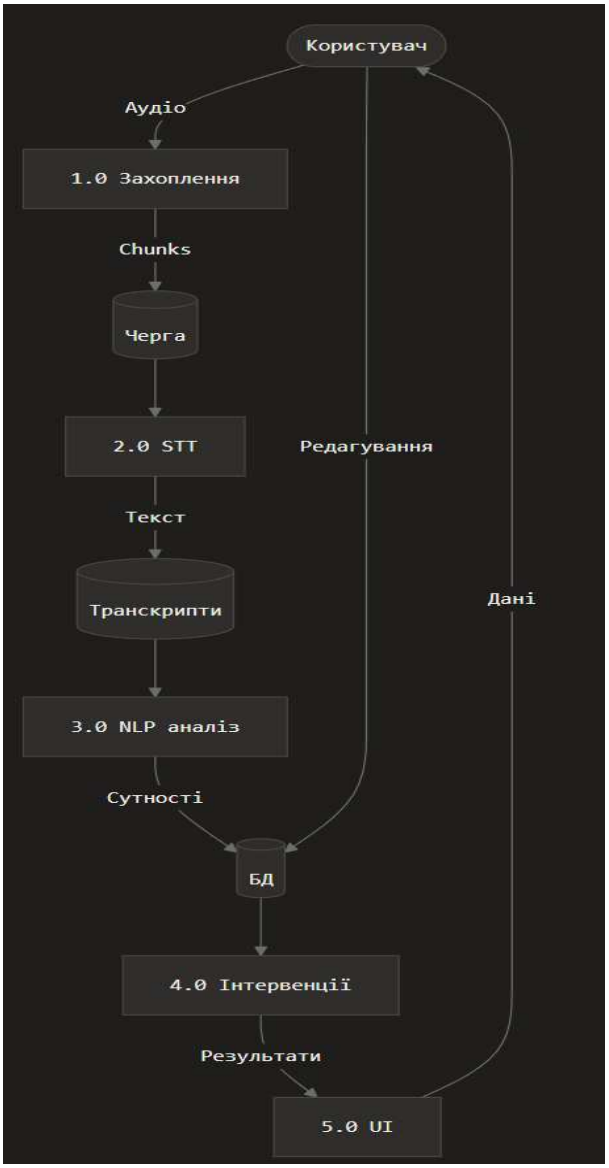

Рис. 1 Діаграма потоків даних

Розробка проєкту модулів системи

Детальне проєктування системи адаптивного асистента онлайн-зустрічей вимагає формалізації функціональних вимог, об'єктної моделі, поведінкових аспектів та фізичної архітектури розгортання. Для повноцінного опису проєкту використано нотацію UML, що дозволяє представити різні аспекти системи у стандартизованому вигляді та забезпечити єдине розуміння архітектури між усіма учасниками розробки.

Діаграма послідовності на рис.2 демонструє типовий сценарій проведення зустрічі з точки зору часової послідовності повідомлень між об'єктами системи. Взаємодія починається з ініціативи користувача розпочати зустріч через веб-інтерфейс. UI-компонент надсилає запит до WebSocket Server на встановлення постійного з'єднання, отримує підтвердження успішного підключення та активує модуль захоплення аудіо в браузері користувача.

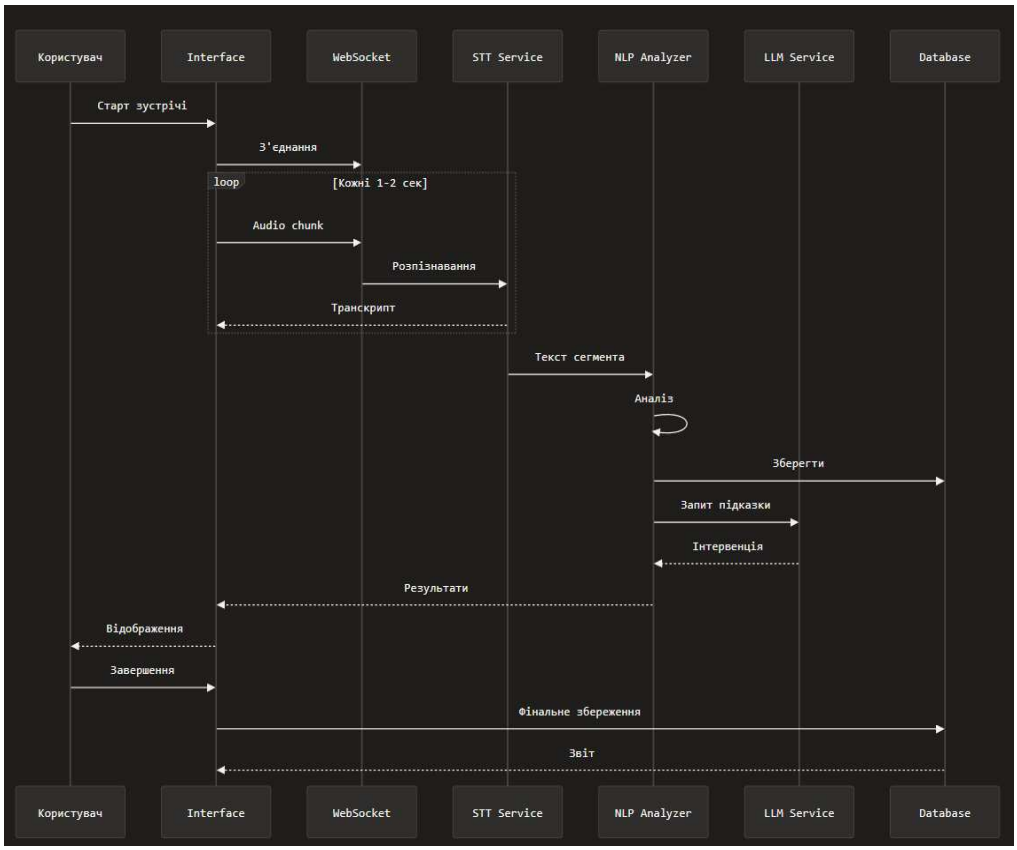

Рис. 2 Діаграма послідовності типового сценарію обміну повідомленнями між компонентами

Після активації Audio Processor починає циклічну передачу аудіофрагментів на сервер з інтервалом одна-два секунди. Кожен фрагмент передається через WebSocket з'єднання до Audio Processor на серверній стороні, який розміщує його в чергу для обробки. STT Service витягує фрагменти з черги та виконує розпізнавання мовлення з використанням моделі Whisper. Результат розпізнавання у вигляді partial transcript негайно надсилається назад до клієнта для відображення користувачеві, що забезпечує ефект роботи в реальному часі.

Діаграма компонентів на рис.3 представляє модульну організацію програмного забезпечення з виділенням логічних блоків та їхніх інтерфейсів взаємодії.

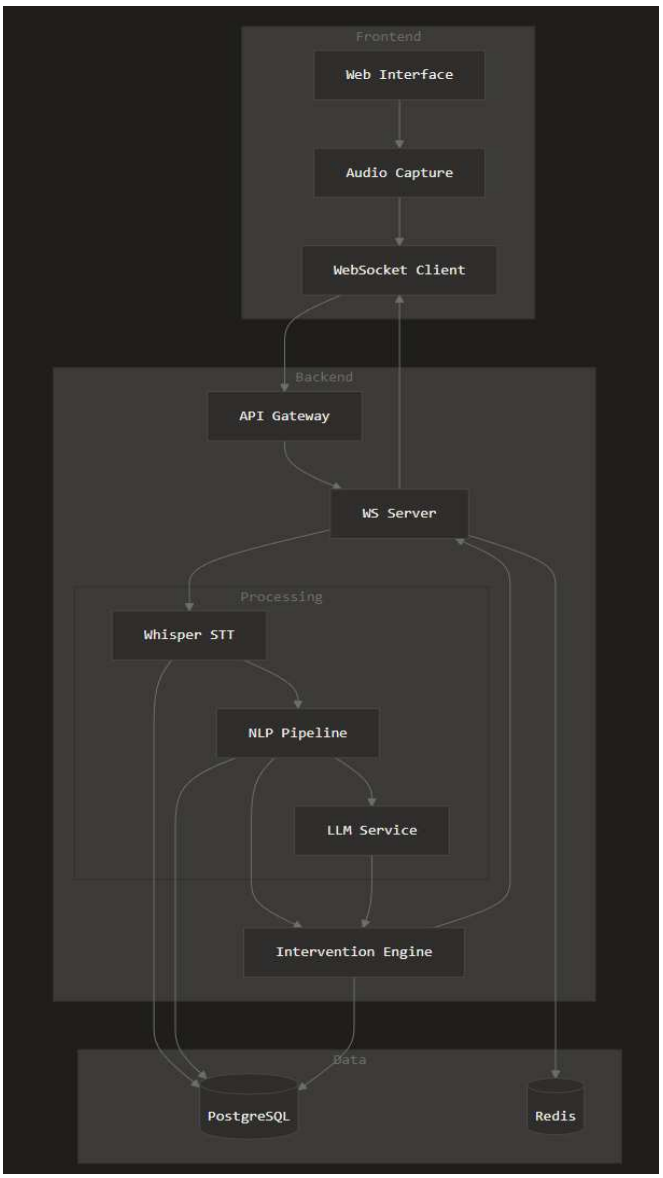

Рис. 3 Діаграма компонентів: модульна організація та інтерфейси

Блок NLP Pipeline реалізує конвеєрну обробку текстових даних з послідовним застосуванням різних алгоритмів аналізу. Компоненти конвеєра включають токенізатор для розбиття тексту на елементарні одиниці, екстрактор пунктів дій на базі моделі послідовного маркування, детектор питань через класифікацію речень, екстрактор рішень з аналізом модальності та контексту. Модульна архітектура конвеєра дозволяє додавати нові типи аналізу без модифікації існуючих компонентів.

LLM Service забезпечує інтерфейс до великих мовних моделей з підтримкою різних провайдерів. Компонент включає клієнта для локальної моделі через Ollama, альтернативного клієнта для OpenAI API у випадку відсутності локальних ресурсів, набір шаблонів промптів для типових завдань

генерації, парсер відповідей для витягування структурованої інформації з згенерованого тексту. Intervention Engine містить компоненти для детекції триггерів інтервенцій на основі аналізу контексту зустрічі, набір правил для визначення типу підказки відповідно до ситуації, механізм rate limiting для контролю частоти генерації повідомлень. Модуль забезпечує баланс між корисністю підказок та ризиком відволікання учасників зустрічі. LLM Service викликається за потреби для створення природномовного контенту інтервенцій. Результати всіх обробок повертаються через WebSocket Server назад до клієнтського інтерфейсу.

Реалізація ключових компонентів

Клієнтський модуль забезпечує захоплення аудіосигналу з мікрофона, попередню обробку та передачу на сервер через WebSocket з мінімальною латентністю. Pipeline побудовано на базі Web Audio API, що дозволяє працювати з аудіо безпосередньо у браузері. Захоплення здійснюється через navigator.mediaDevices.getUserMedia з параметрами sampleRate: 24000, channelCount: 1 (моно) та echoCancellation: true для усунення зворотного зв'язку від динаміків.

Обробку сигналу виконує AudioWorklet, який працює в окремому високопріоритетному потоці, гарантуючи відсутність переривань при навантаженні основного UI-потоку. PCMWorkletProcessor перетворює Float32 у 16-розрядний PCM (Int16Array), застосовує clamping для запобігання цифровому перевантаженню та використовує Transferable Objects для передачі буферів без копіювання. Аудіопотік розбивається на сегменти тривалістю 1–2 секунди і передається бінарними WebSocket-фреймами з кастомним заголовком (timestamp, прапорці діагностики), що економить близько 30% пропускної здатності порівняно з Base64. Механізм backpressure адаптивно пропускає кадри при перевантаженні мережі для уникнення накопичення затримки.

Серверний STT-модуль побудований на Faster-Whisper small з CTranslate2 inference pool. Модель завантажується на GPU при старті через FastAPI startup handler із singleton-патерном; резервний варіант — CPU з INT8-квантуванням. Audio buffer акумулює 0.5–1 с аудіо, а Silero VAD визначає паузи мовлення, формуючи сегменти за реальними межами висловлювань. Pipeline організовано як queue-based систему: WebSocket jobs надходять у bounded asyncio.Queue, worker pool передає chunks до Faster-Whisper, результати через callback спрямовуються до NLP-модуля або зберігаються в базі. Memory management включає torch.cuda.empty_cache, а backpressure контролюється через queue.maxsize та rate limiting.

NLP-конвеєр перетворює транскрипти на структуровані результати через сегментацію, інференс та керування контекстом. Речення токенізуються та нормалізуються перед передачею до fine-tuned моделі DistilBERT, яка класифікує пункти дій (поріг довіри ≈ 0.8), запитання та рішення. Менеджер контексту підтримує ковзне вікно над останніми хвилинами розмови, відстежуючи відкриті питання та згадки учасників. Екстракція виконавців реалізована через NER у поєднанні з регулярними виразами, дедлайни визначаються часовими парсерами, пріоритетність — ранжуванням за ключовими словами. Детектор питань поєднує евристичні сигнали (питальні слова, пунктуація) з навченою

моделлю, а екстрактор рішень реагує на маркери «вирішено», «погоджено», «домовлено» з класифікацією за типами. Сегментатор тем застосовує TextTiling для виявлення меж переходу між темами обговорення.

LLM інтегрований як fallback через Ollama для локального інференсу та OpenAI API для складних випадків. Шаблонна промпт-інженерія з температурою 0.3, кешування та обмеження частоти забезпечують детерміновані результати. Загальна послідовність обробки: транскрипт → сегментація → інференс DistilBERT → евристики → оновлення контексту → структурований результат.

Розробка користувацького інтерфейсу

Користувацький інтерфейс виступає критичним компонентом системи адаптивного асистента онлайн-зустрічей, оскільки саме через нього учасники взаємодіють з результатами обробки аудіопотоку та отримують проактивні підказки в реальному часі. Проєктування інтерфейсу базувалося на принципах мінімалізму та когнітивної ефективності, де кожен елемент має чітке функціональне призначення без створення візуального шуму.

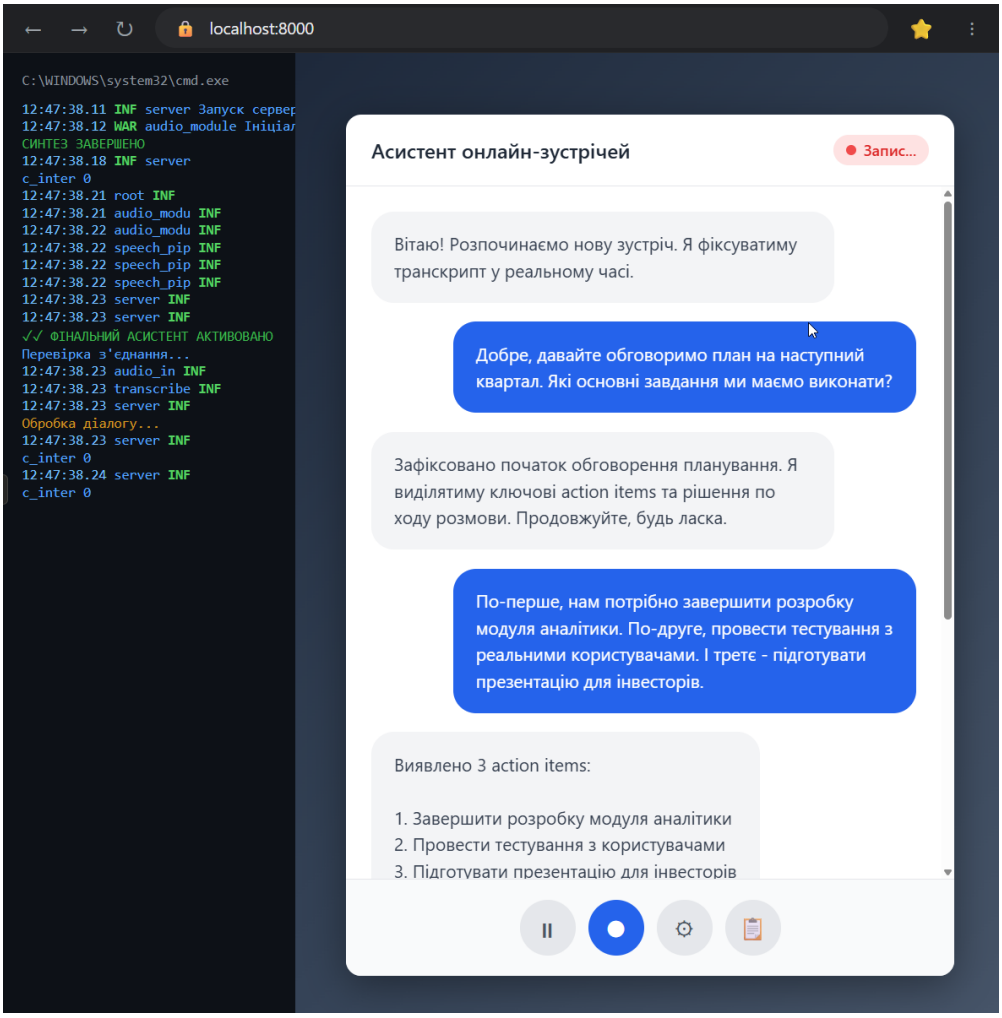

Рис. 4 Загальний вигляд інтерфейсу системи

Система реалізована як веб-додаток, що працює безпосередньо в браузері користувача без необхідності встановлення додаткового програмного забезпечення. На рис.4 представлено загальний вигляд інтерфейсу системи в

процесі активної зустрічі. Робочий простір розділено на дві основні області: ліва частина відведена під термінальне вікно з логами системи для моніторингу процесів обробки, права частина містить основну робочу область з панеллю діалогу та візуальним індикатором активності у вигляді кола.

Центральним елементом інтерфейсу виступає панель чату з заголовком "Асистент онлайн-зустрічей" та індикатором активного запису. Повідомлення візуально диференційовані за типом відправника: репліки користувача відображаються у синіх бульбашках справа, тоді як відповіді асистента мають світло-сірий фон зліва. Транскрипт формується в реальному часі з автоматичним виділенням витягнутих пунктів дій безпосередньо в тексті діалогу. На представленому скріншоті видно приклад автоматичної екстракції трьох завдань з обговорення та подальше призначення виконавців через діалоговий інтерфейс.

Механізм проактивних інтервенцій реалізовано через систему toast-нотифікацій у правому верхньому куті екрану. Колір лівої межі повідомлення відображає пріоритет: зелений для інформаційних підказок, жовтий для рекомендацій, червоний для важливих попереджень. Повідомлення автоматично зникають через п'ятнадцять секунд, а черга обмежена трьома одночасними нотифікаціями для запобігання перевантаженню екрану. Нижня частина панелі містить кнопки управління: пауза запису, активна кнопка запису, налаштування та експорт результатів. Панель налаштувань включає розділи для конфігурації аудіопараметрів, вибору мови розпізнавання, налаштування порогу впевненості та управління частотою інтервенцій. Експорт підтримує формати PDF, Markdown, JSON та спеціалізовані формати для Jira або Trello.

Дослідження на реальних даних

Валідація на реальних записах зустрічей — останній етап перед виходом у продуктивне середовище. Набір даних включає контрольовані студійні сесії та записи з Zoom, щоб система бачила і тишу, і шум, і одночасних мовців, і шуми клавіатур/кімнат, і артефакти стиснення або мережі.

AMI Meeting Corpus [7] надає п’ятнадцять відрізків (5–15 хвилин) із мозкових штурмів, презентацій та дискусій з перетинами, з чіткими мітками мовців і еталонними транскриптами для точності. Щоб захопити типові умовини користувачів, записано десять імітованих зустрічей Zoom із типовими гарнітурами, ехо-компенсацією, шумами стиснення та змінним рівнем звуку. Набір даних також варіює кількість учасників — від діалогів до груп до десяти людей — щоб оцінити деградацію при зростанні складності. Whisper (реалізація faster-whisper) показує стабільну криву погіршення за різних акустичних умов. Чисті шматки на NVIDIA RTX 3060 дають 3.8% WER, що відповідає опублікованим базовим результатам. Шум офісу збільшує WER до 5.2%, а одночасна мова — до 6.7% через невизначеність меж мовців. Латентність inference близько 900 мс на 3-секундний фрагмент. Варіація латентності мала (~40 мс), тому трансляція залишається консистентною. Детальні результати тестування системи розпізнавання мовлення в різних умовах представлено в таблиці 1. На рис.5 лінійний графік показує залежність WER та латентності від умов запису. Видно як погіршуються обидві метрики при ускладненні умов. Екстракція пунктів дій, питань і рішень оцінюється за precision, recall та F1. Action items мають precision

0.87 і recall 0.84 (F1=0.855), тобто більшість виявлених завдань є реальними, а більшість декларацій дій знаходяться, навіть якщо деякі непрямі втрачаються.

Таблиця 1

Результати розпізнавання мови на тестовому наборі

Умова	WER (%)	Латентність (мс)	RTF
Чисте аудіо	3.8	890	0.52
Легкий шум	5.2	920	0.54
Одночасні мовці	6.7	950	0.58

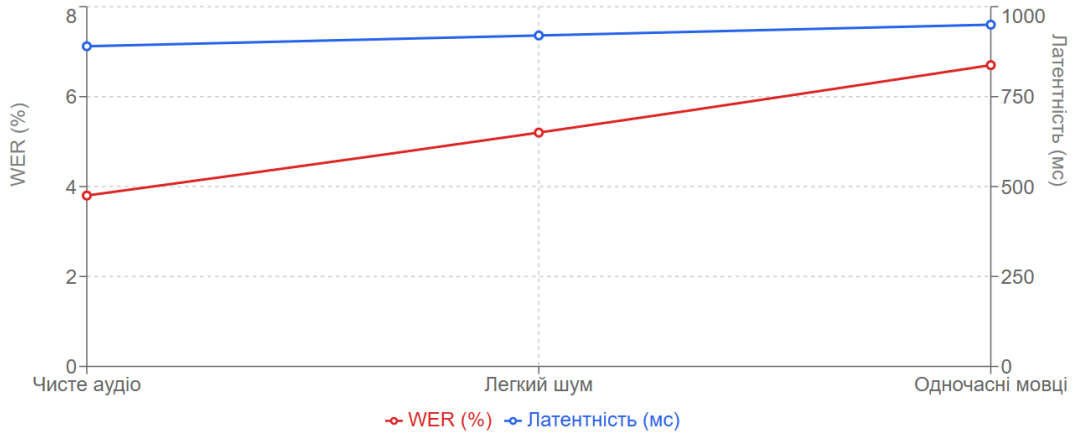

Рис. 5 Лінійний графік залежності WER та латентності від умов запису

Детектор питань: precision 0.92, recall 0.89 (F1=0.905) — запитальність залишається відносно очевидною, тоді як виявлення рішень найскладніше (precision 0.81, recall 0.78, F1=0.795), бо рішення часто формулюються як консенсусні узгодження.

Таблиця 2

Метрики витягання структурованої інформації

Тип	Precision	Recall	F1
Action items	0.87	0.84	0.855
Questions	0.92	0.89	0.905
Decisions	0.81	0.78	0.795

Ці результати перевищують rule-based підходи з регулярними виразами та ключовими словами, де precision action item не перевищував 0.63, а recall — 0.58 через уразливість до мовних варіацій та контексту.

На рис.6 групована стовпчикова діаграма для NLP метрик показує порівняння Precision, Recall та F1 для трьох типів сутностей. Найкраща точність у Questions (F1=0.905), найгірша у Decisions (F1=0.795). Аналіз помилок показує закономірності. Хибнопозитивні пунктів дій часто це загальні пропозиції чи гіпотетичні формулювання без наміру діяти («Можемо поглянути, як оптимізувати запити до бази»). Хибнонегативи виникають, коли завдання розподілене між кількома репліками або виражене непрямо («Час відгуку API непокоїть» → «Так, давайте вивчимо це»). Якщо явно не зазначено виконавця, модель може пропустити задачу.

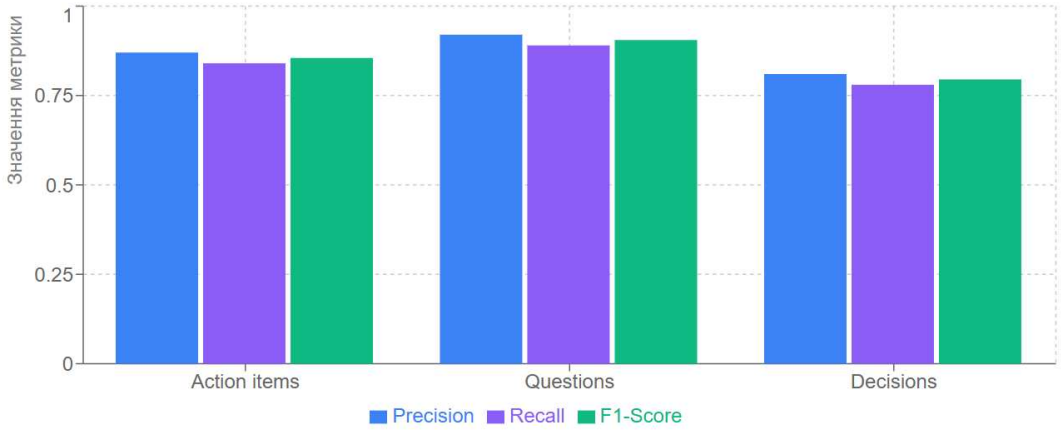

Рис. 6 Групована стовпчикова діаграма для NLP метрик

Питання помилково фіксуються, коли йдеться про риторичну фразу («Хіба не очевидно, що нам потрібні кращі тести?») або повідомлення про запит («Вона запитала, чи ми завершили»); хибнонегативи трапляються в невирішених реченнях («Дивлюся, чому затримка зросла») або в довгих конструкціях, де питальна інтонація замаскована.

Кодове перемикання мови, жаргон, абревіатури та технічні терміни збільшують складність. Whisper справляється з українсько-англійсько-російськими змішуваннями завдяки мультикомпонентному навчальному корпусу, але точність падає для рідкісних пар мов. NLP-моделі, навчені на загальному англомовному корпусі, губляться у технічних термінах; потрібне доменне донавчання [15].

Значення порогів налаштовуються відповідно до сценарію. Якщо пропущене завдання дорожче за зайві сповіщення, знижуємо поріг action-item з 0.7 до 0.6 — recall зростає на ≈5 п.п., а precision падає на ≈3 п.п. Якщо потрібно мінімізувати втручання, підвищуємо поріг до 0.8.

LLM-промти доповнені прикладами, інструкціями ігнорувати гіпотетичні пропозиції та проханням повернути оцінку впевненості; відповіді приводяться до JSON-схеми, температура залишається 0.3 для детермінізму. Для Voice Activity Detection крім початкової чутливості 0.5 використовуються значення 0.1 у шумних кімнатах (жертвується швидким виявленням) та 0.7 с паузи після мови; у швидких обговореннях паузу можна зменшити до 0.5 с, а в повільних — збільшити до 1 с.

Ці результати підтверджують, що система досягає готовності до розгортання, водночас виявляючи напрямки для покращень: настройка порогів, доменна адаптація та промт-інженерія. Особливо перспективними напрямками подальшого розвитку є інтеграція механізмів активного навчання для автоматичного покращення моделей на основі зворотного зв'язку від користувачів, а також розширення підтримки багатомовних сценаріїв зустрічей.

Порівняння з існуючими рішеннями

Для верифікації конкурентоспроможності розроблену систему порівняно з провідними комерційними аналогами Otter.ai [11] та Fireflies.ai [10]. Порівняння проводилося за критеріями функціональності, приватності та мовної підтримки таблиця 3. Аналіз показує, що хоча комерційні продукти мають переваги в

інтеграції з календарями та мобільними платформами, розроблена система займає унікальну нішу завдяки низці факторів.

По-перше, локальність та приватність архітектури дозволяє розгортати рішення на власних серверах організації без передачі даних у хмару, що є критичним для фінансових та медичних установ [15]. По-друге, впроваджена система проактивної фасилітації надає інтелектуальні підказки безпосередньо під час зустрічі, що відрізняє її від сервісів, які обмежуються лише документуванням подій. По-третє, глибока локалізація забезпечує повноцінну підтримку української мови та розпізнавання код-міксу (code-switching), що є дефіцитним функціоналом на глобальному ринку. Таким чином, розроблене рішення є ефективною альтернативою комерційним сервісам для команд, що пріоритезують безпеку даних та спеціалізований функціонал фасилітації.

Таблиця 3

Порівняльний аналіз систем підтримки онлайн-зустрічей

Критерій	Розроблена система	Otter.ai	Fireflies.ai
Транскрипція в реальному часі	так	так	так
Виявлення задач	так	так	так
Виявлення запитань/рішень	так	обмежено	ні
Проактивні підказки	так	Ні	ні
Підтримка української мови	так	обмежено	ні
Локальне розгортання	так	Ні	ні
Відкритий код	так	Ні	ні
Мобільний додаток	ні	так	так

Список використаних джерел:

[1] Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., ... & Zhu, Z. (2016). Deep Speech 2: End-to-End Speech Recognition in English and Mandarin. *Proceedings of the 33rd International Conference on Machine Learning, 48*, 173–182.

[2] Baevski, A., Zhou, H., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *arXiv preprint*. https://arxiv.org/abs/2006.11477

[3] Bailenson, J. N. (2021). Nonverbal Overload: A Theoretical Argument for the Causes of Zoom Fatigue. *Technology, Mind, and Behavior, 2*(1). https://doi.org/10.1037/tmb0000030

[4] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems, 33*, 1877–1901.

[5] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of NAACL-HLT*, 4171–4186.

[6] Graves, A., Fernández, S., Gomez, F., & Schmidhuber, J. (2006). Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. *Proceedings of the 23rd International Conference on Machine Learning*, 369–376.

[7] Howard, J., & Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 328–339.
[8] Jurafsky, D., & Martin, J. H. (2023). *Speech and Language Processing* (3rd ed., draft). https://web.stanford.edu/~jurafsky/slp3/
[9] McCowan, I., Carletta, J., Kraaij, W., Ashby, S., Bourban, S., Flynn, M., ... & Wellner, P. (2005). The AMI Meeting Corpus. *Proceedings of the 5th International Conference on Methods and Techniques in Behavioral Research*, 137–140.
[10] Panayotov, V., Chen, G., Povey, D., & Khudanpur, S. (2015). Librispeech: An ASR Corpus Based on Public Domain Audio Books. *Proceedings of ICASSP*, 5206–5210.
[11] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems, 32*, 8024–8035.
[12] Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., & Sutskever, I. (2022). Robust Speech Recognition via Large-Scale Weak Supervision. *arXiv preprint*. https://arxiv.org/abs/2212.04356
[13] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint*. https://arxiv.org/abs/1910.01108
[14] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems, 30*, 5998–6008.
[15] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Rush, A. M. (2020). Transformers: State-of-the-Art Natural Language Processing. *Proceedings of EMNLP System Demonstrations*, 38–45.
[16] Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent Trends in Deep Learning Based Natural Language Processing. *IEEE Computational Intelligence Magazine, 13*(3), 55–75.

SYSTEM FOR ADAPTIVE ONLINE MEETING ASSISTANCE BASED ON NLP MODELS FOR BIG DATA PROCESSING

SCIENTIFIC RESEARCH GROUP:
Denys Lysyi
Master's student
Interregional Academy of Personnel Management, Ukraine
Mykola Rudnichenko
PhD in Technical Sciences, Associate Professor, Associate Professor of the Department of Information Technologies
National University «Odessa Polytechnic», Ukraine
Natalia O. Shybaieva
PhD in Technical Sciences, Associate Professor, Associate Professor of the Department of Information Technologies
National University «Odessa Polytechnic», Ukraine
Ihor M. Petrov
Doctor of Technical Sciences, Professor, Professor of the Department of Navigation
National University «Odessa Maritime Academy», Ukraine
Denys V. Shvedov
PhD student of the Department of Information Technologies
National University «Odessa Polytechnic», Ukraine

Tetiana V. Otradska
PhD in Technical Sciences, Associate Professor,
Associate Professor of the Department of Information Systems
National University «Odessa Polytechnic», Ukraine

Summary. *The paper presents the architecture and implementation of an adaptive online meeting assistant capable of automatic speech recognition, semantic dialogue analysis, and proactive hint generation in real time. The system is built on a microservice architecture with streaming audio processing and ensures end-to-end latency from speech to transcript display of no more than two seconds. The speech recognition module is implemented using Faster-Whisper with INT8 quantization via CTranslate2 and Silero VAD voice activity detection, on clean. The NLP pipeline based on fine-tuned DistilBERT performs automatic extraction of action items, questions, and decisions from meeting transcripts. The proactive intervention module integrates heuristic rules with large language models via Ollama/OpenAI API to generate contextual suggestions for participants during discussions. Experimental validation was conducted on AMI Meeting Corpus recordings and simulated Zoom sessions. A comparative analysis with commercial counterparts Otter.ai and Fireflies.ai confirmed the uniqueness of the solution in terms of proactive facilitation support, on-premise deployment capability, and full Ukrainian language support.*

Keywords: *adaptive meeting assistant; speech recognition; Whisper; natural language processing; DistilBERT; action item extraction; proactive interventions; microservice architecture; real-time processing; WebSocket.*