НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

«До захисту допущено»
Завідувач кафедри
_____ Наталія АУШЕВА
“___”_____2025 р.

Дипломна робота
на здобуття ступеня бакалавр

За освітньою програмою “Цифрові технології в енергетиці”

Спеціальності 122 “Комп’ютерні науки”

на тему: “Система контролю робочих сесій працівників за геоположенням, визначеним Telegram-ботом”

Виконав: студент 4 курсу, групи ТР-11
ГОЛОСНИЙ Данило Сергійович _____
(прізвище, ім’я, по батькові) (підпис)

Керівник ас. каф. ЦТЕ Микита ГОЛОВАКІН _____
(посада, науковий ступінь, вчене звання, ім’я, ПРІЗВИЩЕ) (підпис)

Рецензент ас. каф. ТАЕ, доктор філософії, Ольга ВЛАСЕНКО _____
(посада, науковий ступінь, вчене звання, ім’я, ПРІЗВИЩЕ) (підпис)

Н.контроль ст.викладач Ольга БЕСПАЛА _____
(посада, ім’я, ПРІЗВИЩЕ) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2025

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти – перший (бакалаврський)
спеціальність 122 “Комп’ютерні науки”
Освітньо-професійна програма “Цифрові технології в енергетиці”

ЗАТВЕРДЖУЮ
Завідувач кафедри ЦТЕ
_____ Наталія АУШЕВА
(підпис)
“___” _____2025 р.

ЗАВДАННЯ

на дипломну роботу студенту

ГОЛОСНОМУ Данилу Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи “Система контролю робочих сесій працівників за геоположенням, визначеним Telegram-ботом”

Науковий керівник ГОЛОВАКІН Микита Андрійович ас. каф. ЦТЕ
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “02” червня 2025 року № 1875-с

2. Термін подання студентом роботи 09.06.2025

3. Вихідні дані до роботи мова програмування Java, бібліотека TelegramBots, СУБД MySQL, середовище розробки IntelliJ IDEA.

4. Перелік питань, які потрібно розробити _____

1) провести аналіз серед наявних аналогів
2) визначити ключові функціональні вимоги до системи для контролю персоналу

3) аргументувати вибрані інструменти, алгоритми та технології для реалізації системи

4) побудувати архітектуру програмного забезпечення та бази даних

5) реалізувати Telegram-бот із підтримкою ролей працівника й адміністратора;

6) продемонструвати функціонування бота та перевірити його працездатність на тестових сценаріях

5. Орієнтований перелік ілюстративного матеріалу архітектура програмної системи, діаграма прецидентів, скриншоти бази даних, скриншоти взаємодії користувача з системою.

6. Дата видачі завдання 13.09.2024

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Вибір теми роботи	13.09.2024	виконано
2.	Аналіз методів та засобів розв’язання задачі	17-20.04.25	виконано
3.	Розробка архітектури та загальної структури системи	21-25.04.25	виконано
4.	Розробка окремих підсистем	25.04-02.05.25	виконано
5.	Програмна реалізація системи	03-07.05.25	виконано
6.	Оформлення пояснювальної записки	08-12.05.25	виконано
7.	Захист програмного забезпечення	12-16.05.25	виконано
8.	Передзахист	26-28.05.25	виконаном
9.	Захист	16.06-20.06.25	виконано

Студент _____ Данило ГОЛОСНИЙ
(підпис) (прізвище та ініціали)

Керівник _____ Микита ГОЛОВАКІН
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дипломна робота виконана на 51 сторінках, містить 21 ілюстрацій, 1 таблицю, 1 додаток, 23 джерела в переліку посилань.

Мета роботи – розробка Telegram-бота для контролю робочих сесій працівників із використанням геоположення.

Методи та засоби: об’єктно-орієнтоване програмування на мові Java, Telegram Bot API, бібліотека org.telegram.telegrambots, база даних MySQL, середовище розробки IntelliJ IDEA, архітектура клієнт-сервер, засоби побудови діалогових інтерфейсів з використанням reply- та inline-кнопок.

У дипломному проєкті проаналізовано існуючі засоби контролю робочого часу, спроєктовано Telegram-бот із системою ролей (працівник і адміністратор), розроблено функції для початку та завершення робочої сесії з передачею геопозиції, відображення графіка, перегляду профілю, статистики та обробки запитів на вихідні.

Результат – програмне рішення, що дозволяє підприємству ефективно вести облік часу роботи працівників, спрощує контроль за присутністю, а також покращує комунікацію між адміністрацією та персоналом.

Ключові слова: Telegram-бот, Java, облік працівників, база даних, геолокація, контроль часу, рольова модель, інтерфейс користувача, інтеграція з MySQL, автоматизація.

ABSTRACT

The diploma project consists of 51 pages, includes 21 illustrations, 1 tables, 1 appendices, and 23 references.

The aim of the project is to develop a Telegram bot for monitoring employees' work sessions using geolocation data.

Methods and tools: object-oriented programming in Java, Telegram Bot API, org.telegram.telegrambots library, MySQL database, IntelliJ IDEA development environment, client-server architecture, user interaction via reply and inline buttons.

The project analyzes existing work time tracking systems, designs a Telegram bot with a role-based system (employee and administrator), and implements functionality for starting and ending work sessions with geolocation capture, viewing schedules, user profiles, statistics, and processing leave requests.

The result is a software solution that allows enterprises to efficiently monitor working hours, simplifies attendance tracking, and improves communication between administration and staff.

Keywords: Telegram bot, Java, employee monitoring, database, geolocation, time tracking, role-based model, user interface, MySQL integration, automation.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ ТА ТЕРМІНІВ

API (Application Programming Interface) — інтерфейс прикладного програмування, набір засобів для взаємодії програмних компонентів.

СУБД — система управління базами даних.

MySQL — популярна система управління реляційними базами даних з відкритим вихідним кодом.

SQL — структурована мова запитів, що використовується для управління даними в реляційних базах.

JDBC (Java Database Connectivity) — інтерфейс для підключення Java-додатків до бази даних.

Telegram API — програмний інтерфейс для створення ботів і застосунків, що взаємодіють із месенджером Telegram.

Telegram Bot — програмний агент, що автоматизує взаємодію з користувачами в Telegram.

TelegramBots — бібліотека Java для роботи з Telegram API.

IntelliJ IDEA — інтегроване середовище розробки (IDE) для Java.

Maven — система автоматизації збірки Java-проєктів і управління залежностями.

SLF4J (Simple Logging Facade for Java) — фасад для логування в Java-додатках.

ВСТУП

У сучасних умовах розвитку цифрових технологій питання ефективного управління персоналом набуває все більшої актуальності. Зокрема, контроль за присутністю працівників на робочому місці, ведення обліку робочого часу та організація зручного графіка роботи є невід'ємною частиною організаційного процесу на підприємствах. У багатьох компаніях ці завдання досі вирішуються вручну або з використанням застарілих систем, що призводить до неефективного використання ресурсів, труднощів в адмініструванні та зниження продуктивності.

Проблема полягає у відсутності простих, зручних і адаптованих до реальних потреб підприємств програмних рішень, які б дозволяли в автоматичному режимі вести облік присутності, формувати графіки змін та забезпечувати прозорий облік робочого часу. Часто впровадження громіздких ERP-систем є недоцільним для малого чи середнього бізнесу, тому виникає потреба у легких і доступних засобах автоматизації.

Одним із таких можливих засобів є використання месенджера Telegram, який дозволяє створювати інтерактивних ботів для взаємодії з працівниками. Telegram-боти легко налаштовуються, доступні з будь-якого пристрою, мають інтуїтивно зрозумілий інтерфейс, а також не потребують встановлення додаткового ПЗ. Завдяки цьому бот може виконувати роль інструменту для реєстрації працівників, фіксації початку робочого дня, перегляду особистого графіку, подання запитів на вихідні, а також адміністрування процесу з боку керівника (директора).

У рамках даної дипломної роботи було розроблено Telegram-бот. Система забезпечує: реєстрацію працівників; фіксацію робочого дня за допомогою геолокації; перегляд профілю користувача та статистики відпрацьованих годин; доступ до календаря змін; подання запитів на вихідні дні; створення та редагування графіків з боку адміністрації.

1 ЗАДАЧА РОЗРОБКИ СИСТЕМИ КОНТРОЛЮ РОБОЧИХ СЕСІЙ ПРАЦІВНИКІВ ЗА ГЕОПОЛОЖЕННЯМ

В сучасних умовах підвищення ефективності роботи підприємств значна увага приділяється автоматизації управлінських процесів, зокрема контролю присутності працівників і плануванню їх робочого часу. Ручне ведення обліку часто є трудомістким і неточним, що призводить до помилок і зниження продуктивності. Автоматизовані системи, зокрема у вигляді Telegram-ботів, дозволяють спростити взаємодію між керівництвом та персоналом, покращити облік робочих годин і оперативно управляти робочим графіком.

1.1 Аналіз загальних вимог системи для контролю робочих сесій

Метою цієї роботи є створення автоматизованої інформаційної системи у вигляді Telegram-бота, яка дозволяє здійснювати ефективний контроль робочих сесій працівників з урахуванням їх географічного положення під час початку та завершення роботи. Така система покликана підвищити прозорість управління персоналом, зменшити ризики маніпуляцій з обліком робочого часу та покращити загальну продуктивність підприємства шляхом цифровізації процесів контролю.

Актуальність розробки системи зумовлена необхідністю підвищення прозорості обліку робочого часу, мінімізації впливу людського фактору та забезпечення зручності користування для робітників. Використання Telegram як основного інтерфейсу взаємодії обумовлено його широкою доступністю, мобільністю, простотою інтеграції та високим рівнем користувацької адаптації. Такий підхід дозволяє ефективно поєднати принципи зручності, безпеки та точності в рамках сучасної інформаційної системи контролю персоналу.

Основні завдання, реалізація яких сприятиме досягненню поставленої мети:

1. провести аналіз сучасних методів обліку робочого часу;

2. дослідити технічні та функціональні особливості аналогічних програмних рішень на ринку;

3. обґрунтувати вибір мови програмування, середовища розробки, системи управління базами даних та інших інструментів;

4. спроєктувати архітектуру Telegram-бота з урахуванням модульної побудови, ролей користувачів та обробки геолокаційних даних;

5. реалізувати ключові функції бота: реєстрацію користувачів, фіксацію робочих сесій з GPS-верифікацією, перегляд графіка, подання заявок на вихідні та адміністрування системи;

6. провести ручне тестування системи у середовищі Telegram;

7. визначити перспективи подальшого розвитку системи.

1.2 Основні функціональні вимоги до системи

Розроблювана система у вигляді Telegram-бота повинна підтримувати два основні типи користувачів — працівника та керівника (адміністратора), кожен із яких отримує доступ до функцій відповідно до своєї ролі та прав. Розробка механізму реєстрації та автентифікації користувачів із чітким розмежуванням ролей на адміністративну (керівник) та звичайну (працівник) є одним із ключових аспектів системи. Такий підхід дозволяє формувати ієрархічну модель доступу до функціональних можливостей системи, забезпечуючи інформаційну безпеку, централізоване адміністрування і розмежування повноважень.

Для працівників система передбачає можливість реєстрації та авторизації, а також реалізацію модуля фіксації початку та завершення робочої сесії з використанням координат геолокації мобільного пристрою. На основі даних про місцезнаходження відбувається верифікація факту фізичної присутності працівника в межах визначеної зони — наприклад, території підприємства чи

іншого затвердженого місця. Збереження координат дозволяє підтвердити дійсність виконання обов’язків, а також слугує додатковим інструментом забезпечення прозорості трудових процесів. Працівники можуть переглядати особистий профіль з інформацією про відпрацьований час, планувати робочі зміни через календар і подавати запити на вихідні дні.

Для керівника система надає інструменти для управління списком працівників, створення і редагування робочих графіків, обробки запитів на вихідні з можливістю їх підтвердження або відхилення, а також формування звітів для контролю робочого часу і дотримання трудової дисципліни. Важливою функціональною складовою є формування централізованої бази даних про робочі сесії кожного працівника з можливістю аналітичної обробки інформації. Система має підтримувати функції агрегування, фільтрації та вивантаження статистики, що створює передумови для об'єктивного оцінювання продуктивності, аналізу трудової дисципліни та прийняття обґрунтованих управлінських рішень.

Загалом система повинна забезпечувати високу доступність і простоту використання завдяки роботі через популярний месенджер Telegram, що не вимагає встановлення додаткового програмного забезпечення. Вона має мати інтуїтивно зрозумілий інтерфейс, адаптований як для мобільних пристроїв, так і для комп’ютерів. Важливим аспектом є захист персональних даних і конфіденційність інформації користувачів, а також надійність роботи з мінімальним ризиком втрати даних. Крім того, система повинна бути масштабованою, щоб задовольняти потреби підприємств різного розміру.

1.3 Потенційні користувачі системи

Розробка системи моніторингу робочих сесій працівників із використанням геолокації орієнтована на широке коло потенційних користувачів, для яких актуальним є підвищення ефективності обліку робочого часу, контроль присутності та оптимізація адміністративних процесів у межах управління

персоналом. Така система є особливо корисною для підприємств, які мають потребу у фіксації часу початку та завершення робочої зміни, а також у перевірці фактичного місця перебування працівника під час виконання трудових обов’язків.

До кола потенційних користувачів передусім належать виробничі підприємства, де облік трудового часу має бути максимально точним, а контроль дисципліни — жорстким. У таких умовах автоматизований підхід до реєстрації змін сприяє зменшенню впливу людського чинника та скороченню витрат часу на обробку табельної документації.

Значний інтерес до впровадження подібної системи також виявляють торговельні та сервісні компанії, в яких стабільність бізнес-процесів безпосередньо залежить від дотримання працівниками встановлених графіків. Застосування Telegram-бота дозволяє керівникам отримувати оперативну інформацію про присутність персоналу, контролювати ротацію змін і вчасно реагувати на порушення режиму.

Особливої актуальності система набуває у сферах, де працівники залучені до мобільної або польової діяльності. Наприклад, у будівельній галузі або логістиці, де персонал працює на змінних об’єктах або переміщується між локаціями, фіксація географічного положення працівника дозволяє забезпечити достовірність обліку та об’єктивність контролю. В умовах, коли стаціонарна система контролю відвідуваності є неможливою або економічно недоцільною, система контролю через Telegram-бот стає універсальним інструментом для дистанційного моніторингу робочої активності.

Установи з гнучким або змінним графіком, зокрема медичні заклади, охоронні фірми, навчальні центри, також отримують вигоду від впровадження такої системи. Вона дозволяє автоматизувати планування змін, подання заявок на відпустки або вихідні, а також організувати прозору взаємодію між працівником і адміністрацією.

Значну зацікавленість у використанні такої системи можуть виявити й малі та середні підприємства, які прагнуть цифровізувати облікові процеси без значних витрат на спеціалізоване програмне забезпечення чи складні IT-рішення. Telegram

як платформа є безкоштовною, інтуїтивно зрозумілою, не потребує встановлення окремих додатків, що значно спрощує процес впровадження.

Окрему категорію потенційних користувачів становлять компанії з віддаленим або гібридним режимом роботи, де виникає потреба у фіксації активності працівників без фізичного перебування в офісі. У таких умовах Telegram-бот із функцією геолокації дозволяє контролювати трудову дисципліну та забезпечити прозорість віддаленої співпраці. Це особливо актуально для компаній у сфері IT, маркетингу, дизайну, консалтингу та інших інтелектуальних послуг.

Таким чином, розроблена система є універсальним рішенням, яке може бути масштабоване відповідно до потреб конкретної організації. Її впровадження сприяє автоматизації управлінських функцій, зменшенню адміністративного навантаження, мінімізації ризиків маніпуляцій із трудовим часом і, як наслідок, підвищенню загальної ефективності роботи підприємства.

2 АНАЛІЗ ПРОБЛЕМИ АВТОМАТИЗАЦІЇ ОБЛІКУ ПЕРСОНАЛУ ТА ІСНУЮЧИХ РІШЕНЬ

Актуальність проблеми автоматизації контролю робочих сесій працівників зумовлена зростаючою потребою підприємств у підвищенні прозорості обліку трудової активності, особливо в умовах гнучких або віддалених форм зайнятості. У сучасному управлінні персоналом важливо не лише фіксувати початок і завершення робочого часу, але й перевіряти фізичне місце перебування працівника під час виконання трудових обов'язків. Такий підхід забезпечує об'єктивність даних, сприяє підвищенню трудової дисципліни та знижує ймовірність маніпуляцій із табелями обліку.

З огляду на це, особливої уваги заслуговують інструменти, що дозволяють поєднати реєстрацію робочих сесій із геолокаційною верифікацією. Аналіз предметної області дозволяє виокремити ключові виклики, пов'язані з реалізацією подібних функцій, а також проаналізувати наявні програмні продукти, що частково або повністю виконують аналогічні задачі. Визначення їх переваг і недоліків дає змогу обґрунтувати доцільність розробки нової, адаптованої системи, орієнтованої на специфіку контролю робочих сесій за географічним положенням працівника з використанням можливостей Telegram-платформи.

2.1 Методи обліку та реєстрації робочого часу персоналу

У сучасних умовах існує кілька основних методів обліку та реєстрації робочих сесій, які застосовуються залежно від розміру підприємства, галузі та технічних можливостей.

Традиційний паперовий облік залишається актуальним для підприємств із невеликим штатом, де ведення табелів робочого часу, журналів наказів, особових справ та інших документів здійснюється вручну. Такий підхід базується на

використанні уніфікованих форм, затверджених нормативно-правовими актами, і вимагає постійного контролю з боку кадрових служб або бухгалтерії. Основними перевагами паперового обліку є його простота та відсутність потреби в складному технічному забезпеченні, однак він має суттєві недоліки, зокрема низьку точність, ризик людських помилок і обмеженість у масштабуванні [1].

Із розвитком цифрових технологій все більшого поширення набувають електронні системи обліку, які дозволяють автоматизувати процеси збору, збереження й обробки інформації про присутність працівників. Такі системи часто є частиною корпоративних систем, що інтегрують кадровий облік із модулями нарахування заробітної плати, управління змінами та формування звітності. Вони забезпечують підвищення точності даних, зниження адміністративного навантаження на персонал, а також можливість оперативного аналізу інформації. Окремі рішення дозволяють відстежувати не лише початок і завершення робочого дня, а й активність працівника в межах зміни, включаючи час перерв, переміщення між підрозділами чи віддалену роботу [2].

Поширеними є також апаратні рішення на основі систем контролю доступу, які включають використання ідентифікаційних карток, магнітних браслетів, PIN-кодів або біометричних параметрів — відбитків пальців, геометрії обличчя, сканування сітківки ока тощо. Такі системи встановлюються на входах до приміщень або робочих зон і дозволяють фіксувати точний час входу та виходу, автоматично формуючи табелі обліку. У разі інтеграції з програмними рішеннями ці дані можуть одразу передаватися в централізовані облікові системи. Основною перевагою таких технологій є їхня висока точність і неможливість фальсифікації даних, що особливо важливо для підприємств із підвищеними вимогами до безпеки та дисципліни [3].

Останнім часом значного поширення набувають мобільні та хмарні рішення для обліку робочого часу, що базуються на використанні смартфонів або спеціалізованих мобільних додатків. Ці інструменти дозволяють працівникам здійснювати реєстрацію початку й завершення зміни через власний пристрій, незалежно від їх фізичного розташування. Особливо актуальною функцією є

верифікація геопозиції за допомогою GPS або мережевих координат, що дає змогу роботодавцю переконатися у фактичній присутності працівника на визначеному об’єкті. Такі рішення ефективні для мобільного персоналу, працівників виїзного обслуговування, логістичних компаній, будівельних бригад та інших категорій, які працюють поза межами офісу [4].

Крім цього, на ринку з’являються системи, які поєднують кілька методів обліку, створюючи гібридні моделі. Наприклад, поєднання системи контролю доступу з мобільним додатком дозволяє забезпечити контроль як на стаціонарному об’єкті, так і під час відряджень. В окремих випадках застосовуються рішення з використанням RFID-технологій, NFC-міток або навіть Wi-Fi-трекінгу для реєстрації присутності працівників у межах певної зони. Усе частіше використовуються аналітичні інструменти, які дозволяють виявляти закономірності в поведінці працівників, прогнозувати затримки або порушення графіків, що відкриває нові можливості для прийняття управлінських рішень.

Таким чином, технологічний спектр засобів обліку робочого часу постійно розширюється, дозволяючи підприємствам адаптувати відповідні рішення до своїх потреб, враховуючи масштаб, специфіку діяльності, рівень мобільності персоналу та необхідність контролю геоположення.

Актуальність використання гнучких, автоматизованих і мобільних рішень зростає в умовах динамічного ринку праці та дистанційних форм зайнятості, що обумовлює доцільність подальшого розвитку таких систем, зокрема на базі месенджерів як Telegram.

2.2 Огляд аналогічних систем

При розробці системи контролю робочих сесій працівників за геоположенням, важливо враховувати досвід та функціональні можливості існуючих рішень на ринку. Сьогодні широко представлені різноманітні HRM-платформи (Human Resource Management – системи управління людськими

ресурсами), які пропонують інструменти для обліку робочого часу, моніторингу присутності, управління графіками змін, а також забезпечують комунікацію між працівниками і керівництвом. Проте більшість з них базуються на веб- або мобільних застосунках без інтеграції з популярними месенджерами, що обмежує зручність використання і оперативність фіксації робочих сесій. Для обґрунтування доцільності створення нової системи варто проаналізувати можливості п’яти популярних існуючих сервісів: Time Doctor, Zoho People. Deputy, Hubstaff та TMetric.

Time Doctor — це новітня система обліку робочого часу, що орієнтована на потреби керівників та менеджерів задля ефективного моніторингу співробітників. Вона дозволяє автоматично відстежувати час, проведений співробітниками за комп’ютером, фіксувати скріншоти робочого екрану, аналізувати активність та продуктивність, а також контролювати дотримання встановленого графіку роботи. Система підтримує функції нагадувань про початок і завершення робочої сесії, що допомагає працівникам дотримуватися режиму. Детальна аналітика та звіти, які формує Time Doctor, дозволяють керівникам оцінювати ефективність роботи кожного працівника, виявляти зони для покращення та підвищувати дисципліну в колективі. Ці можливості роблять Time Doctor корисним інструментом для директорів, які прагнуть забезпечити прозорий і контрольований облік робочого часу [5][6].

Zoho People — це система для управління персоналом, яка надає набір інструментів для автоматизації та оптимізації процесів відстежування робочого часу, присутності працівників та управління відпустками. Вона дозволяє компаніям організувати реєстрацію робочого часу через вбудовані журнали відпрацьованих годин (timesheets), де співробітники можуть вручну вводити час початку і завершення робочого дня або ж використовувати автоматичне відслідковування часу. Однією з ключових особливостей Zoho People є функція автоматичного обчислення робочих годин, включаючи перерви на обід та інші неактивні періоди. Це дозволяє значно знизити кількість помилок при підрахунках робочого часу та забезпечує точність даних. Zoho People також підтримує функцію відстежування

локації за допомогою GPS, що дозволяє точно фіксувати місце перебування працівника на початку та в кінці робочого дня [7].

Deputy — це система для управління персоналом, яка спеціалізується на реєстрації робочого часу, управлінні графіками та автоматизації процесів відстежування присутності працівників. Вона дозволяє користувачам реєструвати час початку і завершення робочих змін через мобільний додаток, що є зручним для співробітників, які працюють на різних локаціях або у виїзних умовах. Одна з основних переваг — це інтеграція з GPS, що дозволяє автоматично перевіряти місцезнаходження співробітника, підтверджуючи, що він знаходиться на робочому місці під час початку і кінця зміни. Ця функція є особливо корисною для компаній, що мають виїзних працівників або для тих, де робота пов'язана з певними локаціями. Deputy також дозволяє керівникам створювати робочі графіки для співробітників і відстежувати зміни в реальному часі. Співробітники можуть отримувати сповіщення про зміни в графіку через мобільний додаток, а також мають можливість надавати запити на відпустку або вихідні, які можуть бути затверджені або відхилені безпосередньо в системі [8].

Hubstaff — це платформа для моніторингу працівників і обліку робочого часу, яка особливо корисна для керівників і власників бізнесу. Вона забезпечує автоматичне відстеження часу роботи співробітників, робить скріншоти екранів, веде облік використаних програм і вебсайтів, а також збирає дані про рівень активності (на основі рухів миші та клавіатури). Крім того, Hubstaff підтримує функцію GPS-відстеження, що є актуальним для польових співробітників або мобільного персоналу. Завдяки гнучкій системі звітності керівники отримують точну інформацію про витрачений час, виконані завдання та продуктивність команди. Платформа інтегрується з багатьма популярними інструментами управління проєктами, що робить її зручною в масштабних бізнес-процесах [9].

TMetric — це інструмент для обліку робочого часу, орієнтований як на окремих спеціалістів, так і на команди. Він дозволяє автоматично фіксувати час виконання завдань, створювати звіти по працівниках, вивантажувати дані про продуктивність та керувати ставками оплати. Для керівників TMetric пропонує

зручні аналітичні панелі, в яких відображається інформація про початок і кінець робочого дня співробітників, кількість відпрацьованих годин, перерви та простої. Крім того, у TMetric доступна функція геолокаційного контролю, що дозволяє моніторити місцезнаходження працівника під час робочої сесії, що є актуальним для мобільних бригад, логістичних або виїзних команд [10].

Крім комерційних рішень, важливо також проаналізувати наукові підходи до проблеми контролю присутності та автоматизованого обліку робочого часу працівників. Нижче були розглянуті сучасні наукові дослідження.

Зокрема, у роботі Ramesh Kumar та Anjali Singh (2024) представлено «Smart Attendance System Using IoT and GPS for Remote Workforce Management», де описується інноваційна система для контролю віддалених працівників. Система об'єднує технології IoT і GPS, що дає змогу віддалено відстежувати місцезнаходження та час праці співробітників в реальному часі. Однією з ключових особливостей є використання геофенсінгу для автоматичного визначення моменту початку та завершення робочої зміни без потреби у ручному введенні даних. Такий підхід мінімізує людський фактор, знижує ризик шахрайства і забезпечує більш точний та надійний облік робочого часу [11].

У статті Sagar Mane та співавторів "Geolocation-Based Employee Attendance and Tracking System" розглядається сучасний підхід до автоматизації обліку присутності працівників, заснований на використанні технологій геолокації. Система, описана в дослідженні, поєднує GPS, геофенсінг і розпізнавання облич, що дозволяє точно визначати як місцезнаходження працівника, так і підтверджувати його особу в момент реєстрації приходу чи відходу з роботи [12].

Принцип роботи полягає в тому, що працівник, перебуваючи в межах визначеної геозони (віртуальної території навколо робочого місця), автоматично реєструється в системі. Для додаткової безпеки й точності також використовується технологія розпізнавання облич, яка підтверджує особу користувача під час входу або виходу [12].

2.3 Порівняння з аналогічними системами

У процесі аналізу існуючих рішень для обліку робочого часу, зокрема Time Doctor, Zoho People, Deputy, Hubstaff та TMetric. Вони пропонують широкий набір функцій, зокрема GPS-відстеження, ручний і автоматичний облік часу, календарі змін та інтеграцію з HR-системами.

Таблиця 2.1 – Порівняння існуючих аналогів та розробленої системи

Характеристика	**Time Doctor**	**Zoho People**	**Deputy**	**Hubstaff**	**TMetric**	**Власна система (Telegram-бот)**
Автоматичний облік робочого часу	так	так	так	так	так	так (з урахуванням геолокації)
GPS-відстеження	так	так	так	так	так	так
Запит на підтвердження геолокації в тракті активної зміни	ні	ні	ні	ні	ні	так
Ручне введення часу	так	так	так	так	так	ні (автоматично через месенджер)
Фіксація екрану / активностей	так	ні	ні	так	так	Ні
Вбудований календар графіку	Ні	так	так	ні	ні	так (інтерактивний через бот)
Подання заявок на вихідні	ні	так	так	ні	ні	Так
Комунікація з керівництвом	ні	так	так	ні	ні	так (через бот)
Інтеграція з месенджерами	ні	ні	ні	ні	ні	так (Telegram)
Цільова аудиторія	Бізнес	Бізнес	Бізнес	Бізнес	Бізнес/ фриланс	Малий бізнес, віддалені команди
Вартість	$ підписка	$ підписка	$ підписка	$ підписка	$ підписка	кастомізована під клієнта

Однак детальний аналіз (Таблиця 1) засвідчив, що більшість цих сервісів орієнтовані на корпоративний сегмент, мають складні інтерфейси та працюють за моделлю підписки, де повноцінний функціонал доступний лише в платних версіях.

Ключовими відмінностями розробленої системи стали:

- інтеграція з месенджером Telegram, що забезпечує швидкий і звичний доступ для користувача;

- автоматичний облік початку та завершення зміни з урахуванням геолокації без ручного введення часу;

- періодичні запити на підтвердження локації під час зміни для реального контролю присутності;

-більш вигідна фінансова модель у порівнянні з аналогами — відсутність плати за кожного користувача та можливість гнучкого розгортання без додаткових витрат.

Таким чином, система розроблена на базі Telegram-боту поєднує зручність використання, гнучкість контролю та функціональність, яких бракує у більшості комерційних аналогів. Зокрема періодичне підтвердження місцезнаходження забезпечують не лише формальний, а й фактичний контроль присутності працівників. Ці фактори стали вирішальними у виборі підходу до реалізації власного рішення.

3 ЗАСОБИ РЕАЛІЗАЦІЇ ПРОГРАМНОЇ СИСТЕМИ

Успішна розробка програмної системи значною мірою залежить від правильно обраного інструментарію. Для створення автоматизованої системи контролю працівників було проаналізовано низку сучасних мов програмування, середовищ розробки, засобів інтеграції з месенджерами та систем управління базами даних. Вибір конкретних технологій обумовлювався вимогами до функціональності, надійності, зручності супроводу, а також доступністю необхідних бібліотек для реалізації ключових можливостей системи — таких як облік робочого часу, обробка геолокації, робота з календарем і збереження даних. У цьому розділі докладно описано вибрані засоби реалізації та обґрунтовано їх використання в межах проєкту.

3.1 Вибір мови програмування та середовища розробки

Для реалізації системи контролю робочих сесій працівників за геоположенням, визначеним Telegram-ботом, було обрано мову програмування Java. Цей вибір обумовлений її універсальністю, об'єктно-орієнтованим підходом та кросплатформеністю, що дозволяє створювати програми будь-якого рівня складності — від мобільних додатків до високонавантажених серверних систем [13].

Java забезпечує сувору типізацію, що знижує ймовірність помилок на етапі компіляції, та автоматичне керування пам’яттю, що спрощує розробку складних систем . Крім того, мова має велику спільноту розробників та багатий набір бібліотек, що полегшує інтеграцію з різними сервісами, такими як Telegram API та MySQL [14].

Для розробки було обрано інтегроване середовище IntelliJ IDEA, яке є одним з найпопулярніших середовищ для Java-розробки. Це середовище надає розширені

можливості для автодоповнення коду, рефакторингу, налагодження та інтеграції з системами контролю версій . IntelliJ IDEA підтримує різні фреймворки та технології, що робить її універсальним інструментом для розробників [15].

Безкоштовна версія IntelliJ IDEA Community Edition є відкритою та достатньою для реалізації більшості функціональних вимог проєкту. Вона підтримує основні функції, необхідні для розробки на Java, включаючи підтримку Maven, Gradle, Git та інших інструментів.

Вибір Java та IntelliJ IDEA забезпечує надійну основу для розробки масштабованої та ефективної системи контролю працівників, що відповідає сучасним вимогам до програмного забезпечення.

3.2 Вибір месенджера та бібліотек для роботи з Telegram

Для реалізації системи контролю робочих сесій було обрано Telegram як основний канал комунікації між користувачами та системою. Цей вибір зумовлений широкою популярністю месенджера в Україні, його відкритим API та зручністю у використанні. Telegram надає розробникам потужні інструменти для створення чат-ботів, що дозволяє ефективно автоматизувати бізнес-процеси та забезпечити інтерактивну взаємодію з користувачами. Зокрема, Telegram API підтримує обробку повідомлень, команд, медіа-файлів, а також інтеграцію з іншими сервісами, що робить його привабливим вибором для реалізації подібних проєктів.

API (Application Programming Interface) — це набір правил і протоколів, що дозволяє різним програмам взаємодіяти між собою. Він виступає посередником, який визначає, як один застосунок може запитувати й отримувати дані або послуги від іншого, спрощуючи процес інтеграції та взаємодії між різними програмними компонентами [16].

Для розробки бота було обрано бібліотеку TelegramBots для мови Java. Ця бібліотека забезпечує комфортний інтерфейс для взаємодії з Telegram API,

підтримує опрацювання повідомлень, команд, і дозволяє без проблем реалізовувати потрібні функціональні можливості. Вибір Java як мови програмування та TelegramBots як бібліотеки обумовлений їхньою стабільністю, широкою підтримкою та активною спільнотою розробників [17].

3.3 Вибір системи управління базами даних MySQL

Система управління базами даних (СУБД) — це програмне забезпечення, яке дозволяє створювати, змінювати, зберігати та керувати базами даних. Вона забезпечує зручний інтерфейс для взаємодії з даними, контролює їхню цілісність, безпеку та забезпечує доступ одночасно для кількох користувачів [18].

Для розробки системи контролю робочих сесій було обрано систему управління базами даних MySQL, оскільки вона є однією з найпопулярніших і найнадійніших СУБД у світі. MySQL підтримує високу продуктивність, стабільність та масштабованість, що робить її оптимальним вибором для бізнес-застосунків середнього та великого масштабу. Крім того, MySQL пропонує зручний інтерфейс, гнучкість у роботі з різними типами даних та підтримує складні SQL-запити, що необхідно для ефективної обробки інформації про робочі сесії та локації працівників [19].

MySQL також відомий своєю зручністю в роботі завдяки інтуїтивно зрозумілому графічному інтерфейсу MySQL Workbench. Цей інструмент дозволяє розробникам легко створювати, моделювати та адмініструвати бази даних без необхідності писати складні SQL-запити вручну. MySQL Workbench забезпечує візуалізацію структури бази даних, автоматизоване створення схем, а також має інструменти для налагодження та оптимізації продуктивності. Такий підхід значно прискорює процес розробки і спрощує управління даними, що особливо важливо при реалізації системи обліку робочого часу і контролю персоналу [20].

3.4 Інші технології та інструменти, що використовуються

У межах розробки програмної системи було використано низку додаткових інструментів і бібліотек, що забезпечили ефективну та структуровану реалізацію проєкту.

Одним із ключових інструментів став Maven — система автоматизації збирання проєктів, яка дозволяє керувати залежностями, збіркою, тестуванням та іншими аспектами життєвого циклу застосунку. Завдяки Maven забезпечено централізоване управління бібліотеками, а також простоту масштабування й підтримки коду. Maven є стандартом для Java-проєктів і широко підтримується середовищем IntelliJ IDEA [21].

З метою реалізації підключення до бази даних MySQL було додано MySQL Connector/J (mysql-connector-java) — офіційний JDBC-драйвер, що дозволяє Java-застосунку здійснювати SQL-запити до бази даних. Він є невід’ємним компонентом інтеграції між Java-кодом і сервером баз даних.

Для логування в проєкті застосовано бібліотеку SLF4J (Simple Logging Facade for Java) разом із реалізацією slf4j-simple. Вона забезпечує просту, але гнучку систему ведення логів, що дає змогу відслідковувати роботу застосунку, зокрема дії користувачів і потенційні помилки під час виконання [22].

Усі ці компоненти додаються до проєкту через файл pom.xml, що дозволяє підтримувати чисту структуру застосунку та забезпечує повторюваність збірки на будь-якому пристрої або сервері.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ TELEGRAM-БОТА

Застосунок у вигляді Telegram-бота реалізовано як клієнт-серверну систему з поділом на окремі компоненти, що відповідають за взаємодію з користувачем, обробку логіки та зберігання даних. Бот надає користувачам зручний інтерфейс для фіксації початку та завершення роботи, надсилання геолокації, подання запитів на вихідні дні та взаємодії з календарем. Уся інформація зберігається в реляційній базі даних MySQL, яка забезпечує надійність і структурованість даних. Для реалізації системи використано об’єктно-орієнтований підхід, що дозволяє легко масштабувати та підтримувати проєкт.

4.1 Архітектура програмної системи

Архітектура програмного забезпечення — це структура та організація програмної системи, яка визначає її компоненти, їх взаємозв’язки та способи взаємодії. Вона впливає на продуктивність, масштабованість та безпеку програмного продукту, а також визначає можливість його подальшого розвитку та підтримки. Архітектура може бути монолітною, мікросервісною, клієнт-серверною або сервіс-орієнтованою, залежно від вимог до системи та її функціональності [23].

Архітектура Telegram-бота реалізована за багаторівневою моделлю, що дозволяє чітко розділити основні компоненти системи та оптимізувати взаємодію між ними. Такий підхід забезпечує високий рівень модульності, що спрощує подальше розширення функціональності, обслуговування та масштабування системи без втрати продуктивності чи стабільності. Загальна структура архітектури наведена на діаграмі (рисунок 4.1), яка ілюструє взаємозв’язки між окремими рівнями та їх функціональні обов’язки.

Клієнтський рівень реалізує інтерфейс взаємодії з користувачами через платформу Telegram. Основним завданням цього рівня є отримання та аналіз вхідних повідомлень від користувачів, розпізнавання їх типів (команди, текстові

повідомлення, натискання кнопок тощо), а також формування відповідних інтерфейсних елементів — таких як Reply-клавіатури та Inline-кнопки для реєстрації, управління робочим часом та іншими функціями бота. Рівень клієнта забезпечує своєчасну та коректну відправку повідомлень, тим самим гарантує безперебійний обмін інформацією між користувачем і сервером.

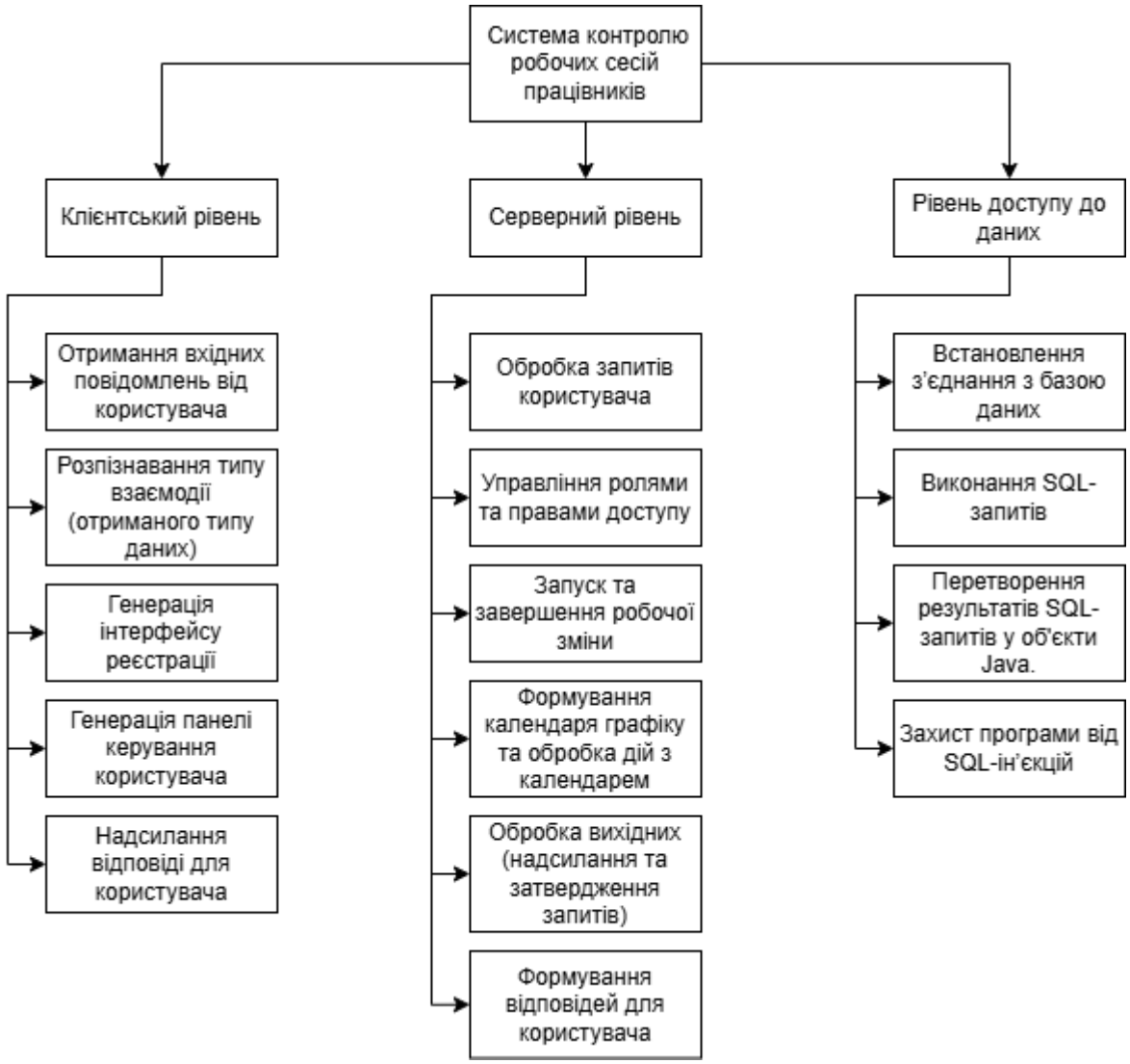

Рисунок 4.1 - Архітектура програмної системи Telegram-бота

Серверний рівень відповідає за основну бізнес-логіку системи. Тут відбувається обробка отриманих запитів із урахуванням ролей користувачів (працівник, адміністратор), а також виконання ключових функціональних операцій — реєстрації, фіксації початку та завершення робочих сесій із фіксацією геолокації,

генерації календаря робочих графіків, опрацювання запитів на вихідні дні. На цьому рівні реалізовані сервіси, які інкапсулюють окремі логічні блоки, що підвищує гнучкість, повторне використання коду та забезпечує розмежування відповідальностей між модулями системи.

Рівень доступу до даних відповідає за безпечне і ефективне збереження та вилучення інформації з бази даних. Забезпечується стабільне підключення до СУБД MySQL, виконання параметризованих SQL-запитів, що мінімізує ризик SQL-ін’єкцій, а також конвертація отриманих результатів у відповідні Java-об’єкти. Особливу увагу приділено організації зберігання даних про користувачів, їх ролі, робочі сесії, графіки та заявки на вихідні. Такий рівень забезпечує цілісність даних і гарантує їх коректну обробку на всіх етапах функціонування системи.

Таким чином, обрана архітектурна модель підтримує прозору взаємодію між користувачем і сервером, дозволяє ефективно розподіляти навантаження, підвищує безпеку та надійність роботи Telegram-бота в умовах реального використання на підприємстві.

4.2 Опис структури бази даних, зв’язки між таблицями

Для зберігання структурованої інформації про користувачів, їхню активність, робочі графіки та заявки на вихідні дні в системі Telegram-бота застосовується реляційна база даних MySQL. Вона реалізована із дотриманням нормалізації, що забезпечує уникнення дублювань, а також підтримку цілісності даних через встановлення логічних зв’язків між таблицями. Загальна структура бази даних представлена на рисунку 4.2, де відображено основні таблиці та їх міжтабличні зв’язки.

Головною таблицею є users, яка містить базову інформацію про зареєстрованих користувачів системи. Вона зберігає унікальний ідентифікатор користувача (id), що виконує роль первинного ключа і слугує зв’язком із іншими таблицями, зокрема work_sessions та day_off_requests. Крім ідентифікатора,

таблиця містить Telegram ID, ім’я, прізвище, псевдонім користувача, а також його роль у системі (наприклад, працівник або адміністратор). Додатково фіксуються дата створення облікового запису, статус активності користувача та час останньої активності, що важливо для моніторингу користувацької взаємодії.

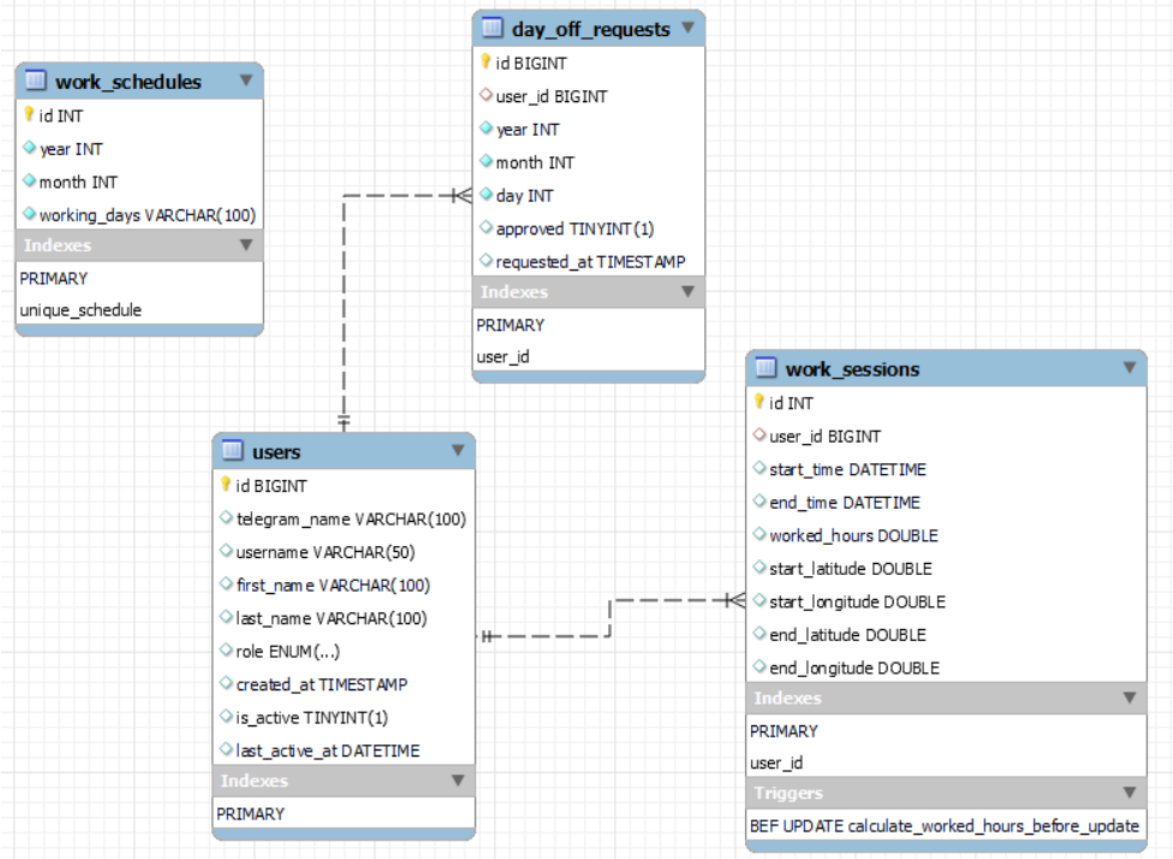

Рисунок 4.2 – Структура бази даних

Таблиця work_sessions відповідає за реєстрацію робочих сесій працівників. Вона містить поля для зберігання часу початку та завершення робочої зміни, координат геолокації, а також посилання на конкретного користувача через зовнішній ключ user_id.

Важливою функціональною складовою є тригер calculate_worked_hours_before_update, що автоматично виконує розрахунок тривалості робочої сесії у годинах при оновленні запису, що фіксує завершення

зміни. Це дозволяє мінімізувати можливі помилки ручного введення та забезпечує точність обліку відпрацьованого часу.

Таблиця work_schedules структурована для зберігання даних про робочі графіки користувачів. Вона містить поля year і month, які разом формують унікальний ключ для кожного місячного набору робочих днів. Це дає змогу гнучко налаштовувати графіки для різних періодів та користувачів, забезпечуючи відображення робочих і вихідних днів у календарі.

Для обробки заявок на вихідні дні використовується таблиця day_off_requests, що зберігає дату запиту, його статус (наприклад, «затверджено» або «відхилено»), а також дату і час створення запису. Ця таблиця також пов’язана з users через зовнішній ключ user_id, що гарантує можливість відстежувати, який користувач подав відповідний запит. Така організація даних забезпечує прозорість і контроль процесу погодження вихідних днів.

Загалом структура бази даних спроєктована з урахуванням вимог до надійності, безпеки та ефективності обробки інформації. Вона підтримує масштабованість системи та дозволяє легко додавати нові функціональні можливості без порушення цілісності існуючих даних.

4.3 Опис основних функцій

Розроблена система контролю робочих сесій працівників, побудована на базі Telegram-бота, реалізує комплекс ключових функціональних можливостей, спрямованих на автоматизацію управління персоналом та підвищення ефективності контролю робочого часу. Ці функції охоплюють повний життєвий цикл взаємодії користувача із системою — від реєстрації до обробки запитів на вихідні, забезпечуючи прозорість, надійність та зручність роботи.

Для наочності та кращого розуміння структури взаємодії між користувачами та системою, було розроблено діаграму прецедентів (рисунок 4.3), яка ілюструє основні ролі (актори) та прецеденти (функціональні сценарії) системи. На цій

діаграмі чітко відображено, які дії доступні працівнику, адміністратору, а також які автоматизовані функції виконує сама система.

Рисунок 4.3 – Діаграма прецедентів системи контролю робочих сесій працівників за геоположенням

Працівник має змогу пройти покрокову реєстрацію, де збираються особисті дані (прізвище, ім’я, посада) з використанням інтерактивних Inline-кнопок, що забезпечує зручність та швидкість процесу. Після успішної реєстрації користувач отримує доступ до функцій початку та завершення робочої зміни, які супроводжуються фіксацією часу та геолокації у базі даних.

Особливістю системи є автоматичне надсилання запитів на підтвердження геолокації протягом робочої зміни, що дозволяє реалізувати ефективний геофенсінг та контроль присутності в межах визначеної території.

Календар, який відображає робочі дні відповідно до графіка підприємства, забезпечує наочне маркування робочих та вихідних днів. Працівник може подавати запити на вихідні, які зберігаються в системі і надсилаються адміністратору для розгляду та підтвердження.

Адміністратор у системі володіє розширеними правами доступу, що дозволяють йому здійснювати комплексне управління робочими процесами підприємства. Основним завданням адміністратора є координація діяльності працівників, а також забезпечення прозорості та контролю за їх робочими сесіями. Зокрема, адміністратор має можливість переглядати та редагувати список працівників, вносячи зміни щодо їхніх даних або ролей у системі, що дозволяє оперативно реагувати на кадрові зміни.

Крім того, адміністратор має доступ до моніторингу активних робочих сесій у реальному часі, що дає змогу відслідковувати статус працівників, час початку та завершення змін, а також їхнє географічне положення, підтверджене за допомогою геофенсінгу. Такий контроль дозволяє запобігати зловживанням, вчасно виявляти порушення та підвищує дисципліну на робочому місці.

Важливою функцією є також обробка запитів на вихідні та інші види відпусток. Адміністратор отримує повідомлення про подані заявки, має можливість їх схвалити або відхилити, а також коментувати рішення, що забезпечує прозорість та ефективність комунікації між працівниками і керівництвом. За потреби адміністратор може вносити корективи до графіків роботи, перерозподіляючи зміни між працівниками для оптимального покриття робочих змін і врахування їхніх побажань.

Додатково система надає адміністратору можливість формувати статистичні звіти та аналітику щодо відпрацьованого часу, кількості вихідних, продуктивності працівників, що є важливим інструментом для прийняття управлінських рішень та оптимізації ресурсів підприємства. Всі ці функції реалізовані таким чином, щоб полегшити роботу адміністратора, мінімізувати ризик помилок і підвищити загальну ефективність системи контролю.

Варто відзначити, що функціональні можливості реалізовані у вигляді окремих сервісів та обробників подій, що чітко розмежовують завдання кожного компонента, полегшуючи підтримку коду та його подальший розвиток.

4.4 Тестування системи

У процесі розробки системи контролю робочих сесій на базі Telegram-бота було проведено комплексне ручне (інтерактивне) тестування ключових функцій системи. Основна мета тестування полягала у верифікації коректності взаємодії користувачів із ботом, перевірці правильності збереження та обробки даних у базі, а також забезпеченні відповідності реалізованого функціоналу вимогам, визначеним ролями користувачів (працівник та адміністратор). Тестування відбувалось у реальному середовищі Telegram із використанням тестових акаунтів користувачів.

Процедура автоматичної реєстрації нового користувача була ретельно протестована із застосуванням двох типових ролей — адміністратора та працівника. Тестування проводилось у реальному середовищі, що дозволило перевірити послідовність покрокового введення персональних даних та коректність їх обробки системою. У ході тестів реєстрація проходила успішно, при цьому всі введені дані належним чином зберігалися в базі даних. Приклад структури таблиці користувачів із збереженими протестованими даними наведено на рисунку 4.4.

id	telegram_name	username	first_name	last_name	role	created_at	is_active	last_active_at
474261282		jo	Данило	Робітник	employee	2025-05-18 18:22:33	0	2025-05-18 19:20:58
6749457740	Danylo Holosnyi	NULL	Данило	Адмін	admin	2025-05-18 19:04:36	0	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 4.4 – Результат збережених даних користувача при тестуванні функції реєстрації

Механізм фіксації початку та завершення робочої зміни було протестовано із залученням користувача з роллю працівника, що дозволило перевірити повний цикл формування робочої сесії. У рамках тестування працівник ініціював початок зміни, передаючи свою геолокацію — для цього завчасно було вибрано валідну та достовірну точку, яка відповідала очікуваним параметрам і забезпечила успішний запуск робочої сесії. Система фіксувала точний час входу та координати місцезнаходження. Аналогічно, при завершенні зміни зберігалися час виходу та фінальна позиція користувача, після чого автоматично здійснювався розрахунок загальної тривалості сесії. Було підтверджено, що всі дані коректно записуються до таблиці work_sessions бази даних, а реалізований тригер забезпечує точний підрахунок відпрацьованих годин. Результати коректного збереження даних тестової робочої сесії зображено на рисунку 4.5.

id	user_id	start_time	end_time	worked_hours	start_latitude	start_longitude
41	474261282	2025-05-18 18:41:30	2025-05-18 18:43:06	0.026666666	52.383316	16.892866
42	474261282	2025-05-18 18:45:56	2025-05-18 18:45:59	0.000833333	52.383316	16.892866
43	474261282	2025-05-18 19:19:21	2025-05-18 19:20:58	0.026944444	52.383128	16.893192

Рисунок 4.5 - Результат збережених даних робочої сесії працівника при тестуванні функції початку та завершення праці з використанням зчитування місцеположення

Процес формування робочого графіка адміністративним користувачем був реалізований та протестований у рамках функціонального модуля Telegram-бота, що відповідає за керування плануванням робочого часу працівників. Відповідно до наданих прав доступу, користувач з роллю адміністратора має можливість ініціювати створення графіка роботи шляхом взаємодії з відповідним інтерфейсним блоком бота. У ході тестування дана можливість була перевірена шляхом вибору дат адміністратором, які мають статус робочих. Після завершення введення даних система автоматично зберігала графік у відповідну таблицю

work_schedules бази даних. Результат коректно збереженого тестового робочого графіку строком на два місяці зображено на рисунку 4.6.

id	year	month	working_days
5	2025	5	1,2,5,6,7,8,9,13,14,15,16,19,20,21,22,23,26,27,28,29,30
6	2025	6	2,3,4,5,6,9,10,11,12,13,16,17,18,19,20,23,24,25,26,27,30

Рисунок 4.6 - Результат збережених даних робочого графіку при тестуванні формування графіку працівників користувачем адміністратор

Механізм надсилання запитів на вихідні дні реалізовано як окремий підкомпонент функціональної частини Telegram-бота, орієнтований на взаємодію між працівником і адміністративною стороною системи. Працівник має змогу ініціювати запит на отримання вихідного дня через інтуїтивно зрозумілий інтерфейс бота, де обирає одну або декілька дат із запропонованого календаря. Після підтвердження вибору система автоматично формує запис у таблиці day_off_requests бази даних, який містить ідентифікатор користувача, вибрані дати, статус запиту та дату створення. При цьому реалізовано додаткові механізми перевірки цілісності даних: система здійснює перевірку на унікальність запиту — якщо на вибрану дату вже існує активний або оброблений запит від цього ж користувача, повторне додавання не виконується. Крім того, перед збереженням запиту здійснюється валідація обраної дати — забороняється подання запиту на дні, що вже минули відносно поточної дати. Такий підхід дозволяє забезпечити узгодженість даних у базі, виключити дублювання записів, а також мінімізувати кількість помилок, пов’язаних з людським фактором при поданні запитів.

Адміністратор отримує повідомлення з деталями кожного нового запиту, після чого має змогу ухвалити або відхилити його безпосередньо через Telegram-інтерфейс. Вибране рішення зберігається в базі даних шляхом оновлення відповідного запису у полі status, що гарантує прозорість усіх етапів обробки. У результаті проведеного тестування з використанням тестових облікових записів

було підтверджено, що процедура надсилання та обробки запитів відбувається стабільно: дані зберігаються коректно, статуси оновлюються відповідно до дій адміністратора, а система забезпечує необхідний рівень зворотного зв’язку для обох сторін. Результатами тестування даної функції є збережені дані про запити вихідних в базі даних зображені на рисунку 4.7.

id	user_id	year	month	day	approved	requested_at
15	474261282	2025	5	20	0	2025-05-18 19:09:05
16	474261282	2025	5	21	0	2025-05-18 19:09:05
17	474261282	2025	5	22	0	2025-05-18 19:09:05
18	474261282	2025	5	28	1	2025-05-25 22:19:47

Рисунок 4.7 - Результат збережених тестових даних запитів вихідних

Інтерактивне тестування Telegram-бота дало змогу переконатися в коректності реалізації всіх ключових сценаріїв взаємодії користувача із системою, стабільності роботи функціональних компонентів, а також відповідності фактичної поведінки очікуваним результатам. Система успішно пройшла перевірку в реальному середовищі з використанням тестових даних, що підтверджується відповідними записами в базі даних та скріншотами, які демонструють правильність виконання кожного етапу. Це свідчить про готовність рішення до практичного застосування.

5 ВЗАЄМОДІЯ КОРИСТУВАЧА З TELEGRAM-БОТОМ

У цьому розділі розглядаються ключові моменти роботи з Telegram-ботом, що забезпечують комфортне і інтуїтивне користування системою. Описано вимоги до обладнання та програмного забезпечення, процес встановлення бота, а також особливості інтерфейсу і типові сценарії взаємодії користувачів із системою.

5.1 Вимоги до програмного і апаратного забезпечення

Система взаємодіє з користувачами через месенджер Telegram, тому для повноцінної роботи Telegram-бота з боку користувача не вимагається спеціальне програмне чи апаратне забезпечення. Достатньо мати встановлений додаток Telegram на будь-якому сучасному пристрої. Telegram підтримується на Android і iOS смартфонах, а також доступний для використання на комп’ютерах під управлінням Windows, macOS та Linux. Крім того, існують браузерні версії Telegram, які працюють у більшості сучасних вебоглядачів, таких як Google Chrome, Mozilla Firefox, Microsoft Edge та Safari. Це робить систему доступною для найширшого кола користувачів — достатньо перейти за посиланням або відсканувати QR-код, щоб розпочати взаємодію з ботом.

З боку сервера, де розміщується Telegram-бот, існують певні технічні вимоги для його стабільного функціонування. Бот написаний мовою Java, використовує базу даних MySQL, а також взаємодіє з Telegram API через інтернет-з’єднання. Для його розгортання необхідний сервер з встановленою операційною системою Linux або Windows, підтримкою Java (версії 17 або новішої), а також доступом до MySQL Server. Базові апаратні характеристики включають щонайменше одне ядро процесора, від 512 МБ оперативної пам’яті (рекомендовано 1–2 ГБ для більшої стабільності), і близько 500 МБ вільного місця на диску. Сервер має бути постійно підключеним до мережі для забезпечення обміну даними з Telegram.

Розгорнути Telegram-бота можна як на локальному комп’ютері, так і в хмарі, наприклад, за допомогою таких платформ як Heroku, Render, DigitalOcean або AWS. Поширення бота здійснюється за прямим посиланням на його профіль у Telegram, а також може бути інтегроване на корпоративний сайт або представлено у вигляді QR-коду для швидкого доступу співробітників.

5.2 Інтерфейс користувача та сценарії взаємодії

Першим кроком у взаємодії користувача з ботом є запуск бота надіславши команду /start. Бот відповідає привітальним повідомленням і пропонує користувачу пройти реєстрацію. Процес реєстрації зображено на рисунку 5.1.

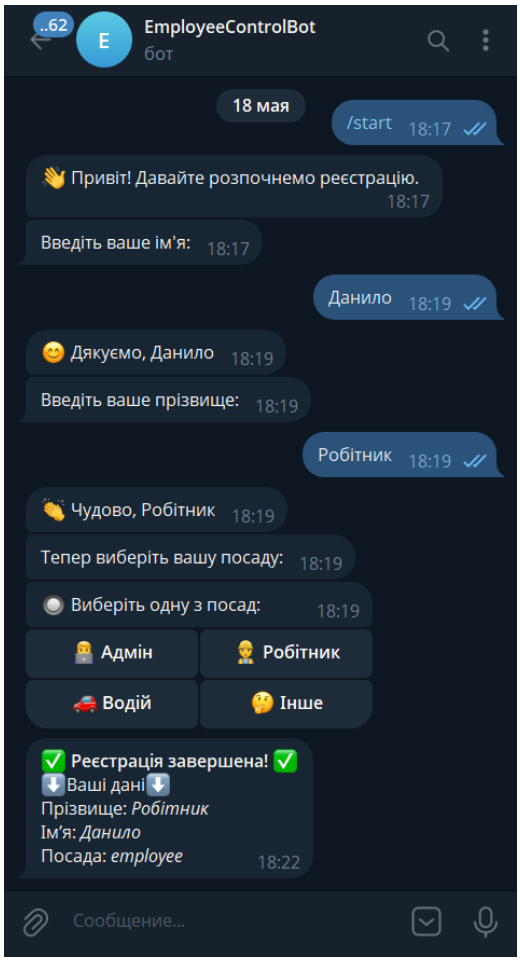

Рисунок 5.1 -Процес реєстрації користувача

Процес реєстрації відбувається покроково: бот послідовно запитує прізвище, ім'я та посаду користувача. Посаду можна вибрати через інтуїтивні inline-кнопки. Після кожного кроку бот підтверджує отримання даних, що забезпечує чіткість та прозорість процесу.

Після реєстрації бот відображає клавіатуру з основними командами, яка є завжди активною, що полегшує навігацію. Залежно від ролі (працівник чи адміністратор) бот відображає відповідне меню з командами, необхідними для роботи.

Сценарій взаємодії користувача «робітник» виглядає наступним чином. Робітник має клавіатуру на якій доступні наступні функції: почати роботу, звершити роботу, профіль, графік. Функціональну клавіатуру робітника зображено на малюнку 5.2.

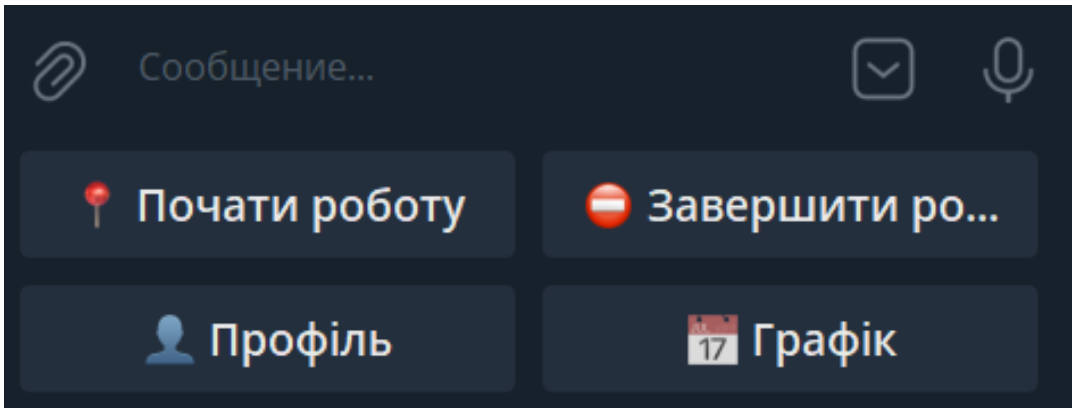

Рисунок 5.2 – Функціональна клавіатура робітника

Натискаючи кнопку «Почати роботу», працівник надсилає свою геолокацію, після чого бот фіксує час початку зміни, зберігає координати у базі даних та дає цьому робітнику статус «активний» (рисунок 5.3). Це дозволяє вести облік фактичного початку роботи.

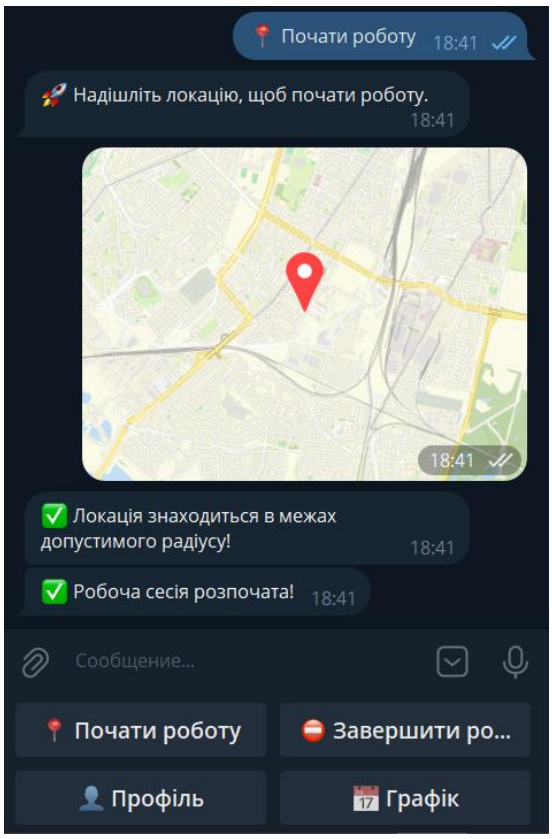

Рисунок 5.3 – Реєстрація початку роботи робітником

Кнопка «Звершити роботу» завершує активну зміну працівника (рисунок 5.4). Бот фіксує час закінчення роботи та оновлює статус працівника у системі як «неактивний».

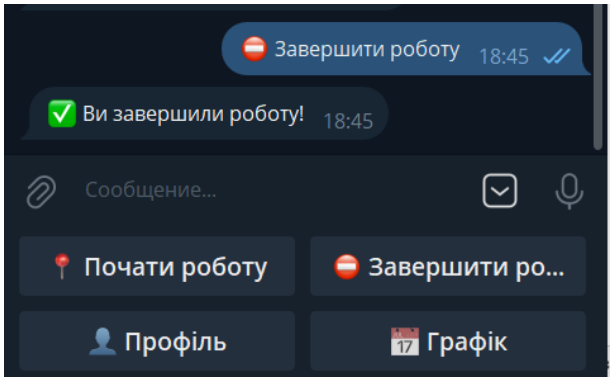

Рисунок 5.4 – Реєстрація завершення роботи робітником

Крім того в боті наявна функція, яка в продовж робочої сесії перевіряє чи робітник все ще на робочому місці, надсилаючи через деякий час користувачу повідомлення-нагадування з запитом на підтвердження своєї локації (рисунок 5.5). У користувача йде таймер впродовж якого він має підтвердити своє місцезнаходження на роботі. Якщо робітник цього не робить, то робоча сесія закінчується автоматично.

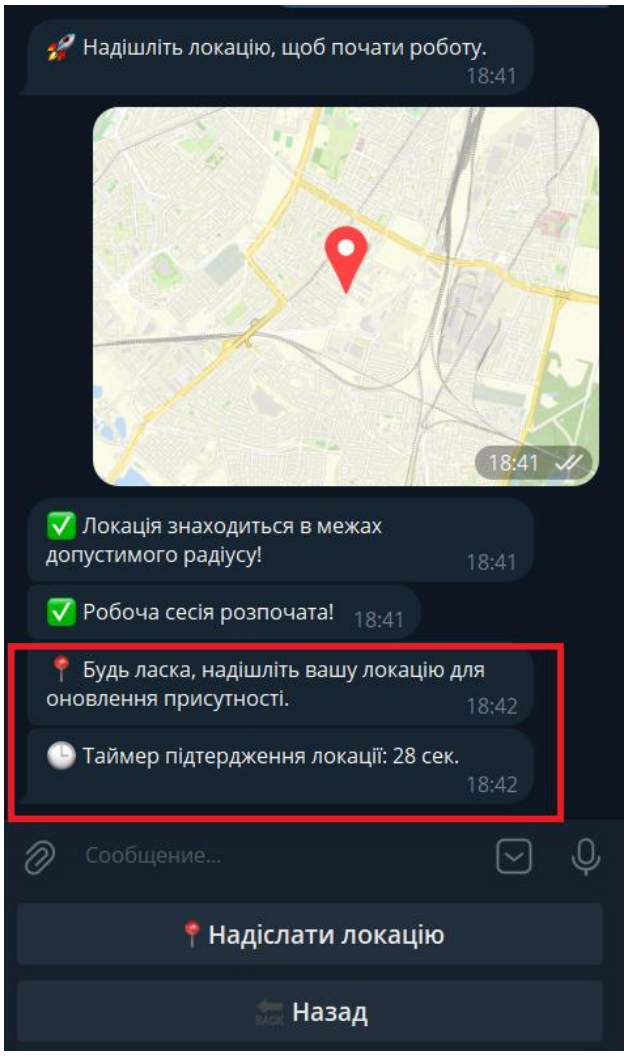

Рисунок 5.5 – Запит на підтвердження місцезнаходження впродовж робочої сесії

При натисканні «Профіль» бот надсилає працівнику інформацію про його обліковий запис: ім’я, прізвище, посаду, підсумок робочого часу та затверджені вихідні (рисунок 5.6).

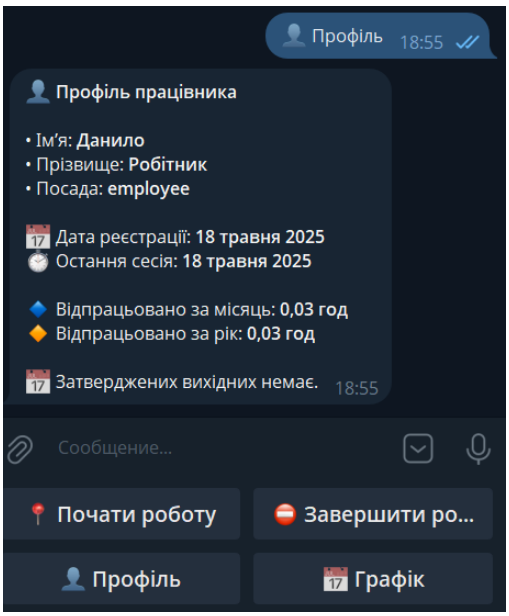

Рисунок 5.6 – Натискання на кнопку «Профіль»

Натиснувши на кнопку «Графік» працівнику буде виведено календар із відмітками робочих днів (у вигляді крапочок). Робочий календар зображено на рисунку 5.7.

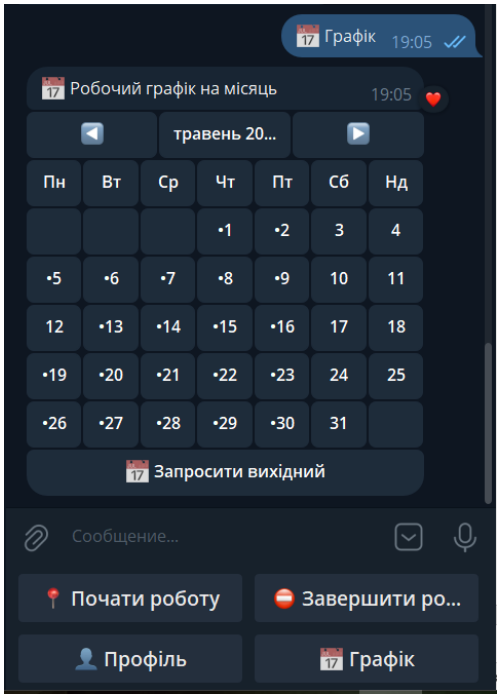

Рисунок 5.7 – Відображення робочого календаря для робітника

Під робочим календарем знаходиться кнопка «Запросити вихідні». Натиснувши на неї у користувача є можливість обрати собі вихідні (рисунок 5.8). Обрані дні будуть позначені хрестиком. Після вибору вихідних робітнику слід натиснути кнопку «Готово» і запит на його вихідні буде надіслано до адміністраторів. Крім того, в цьому меню відображаються вже раніше надіслані запити на вихідні та затверджені вільні дні.

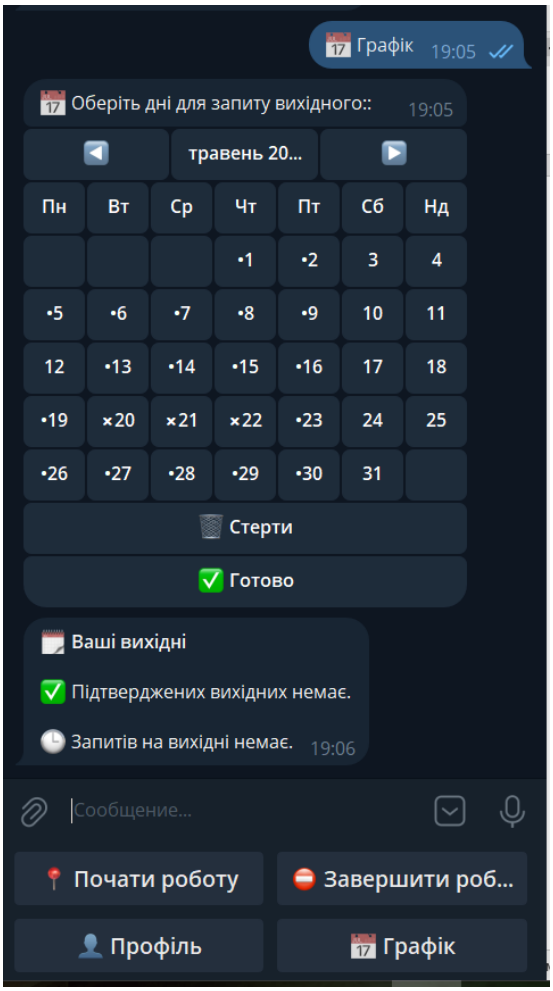

Рисунок 5.8 – Процес запиту вихідних робітником

Сценарій взаємодії користувача «адміністратор» виглядає наступним чином. Адміністратор має клавіатуру на якій доступні наступні функції: переглянути працівників, статистика, розклад, запити вихідних. Функціональну клавіатуру адміністратора зображено на малюнку 5.9.

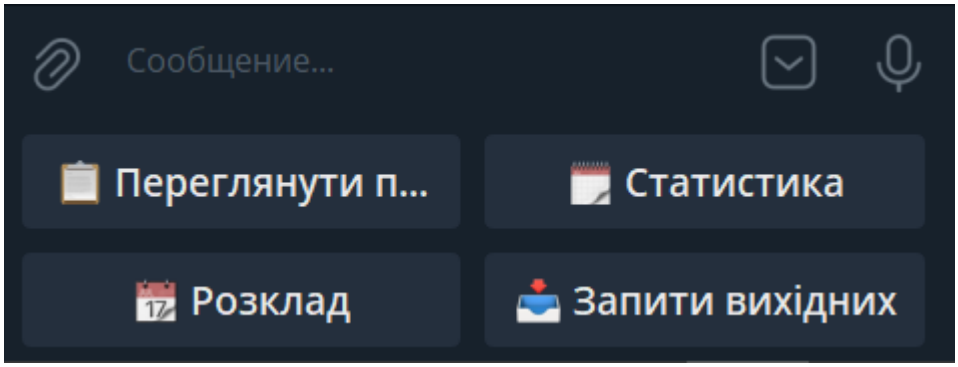

Рисунок 5.9 - Функціональна клавіатура адміністратора

Натиснувши кнопку «Переглянути працівників» адміністратор отримує зведену інформацію про роботу працівників: ім’я, прізвище, посаду загальний робочий час за місяць (рисунок 5.10). Це дозволяє оперативно оцінювати ефективність персоналу.

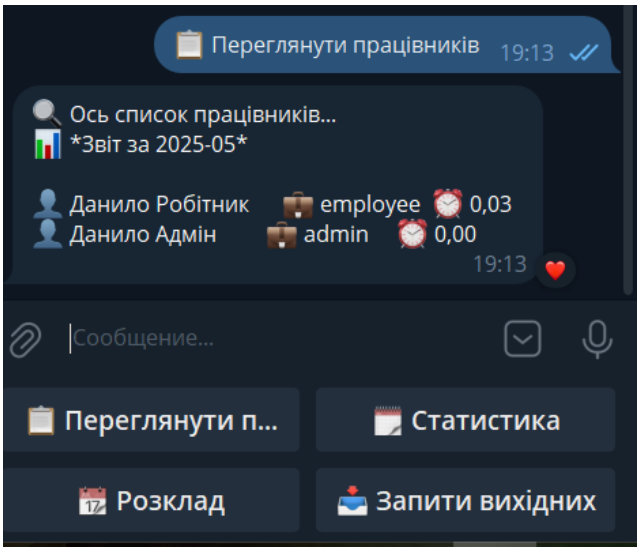

Рисунок 5.10 – Перегляд всіх працівників адміністратором

Кнопка «Статистика» виводить список активних працівників, тобто ті хто зараз в роботі. Цей список зображено на рисунку 5.11.

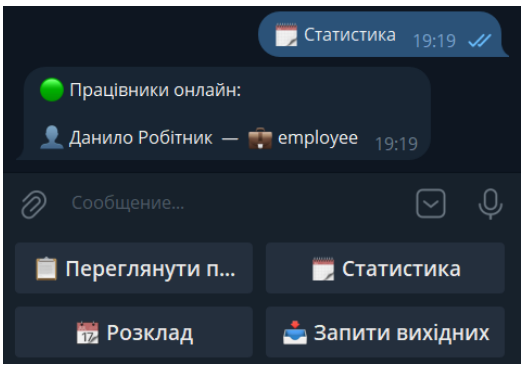

Рисунок 5.11 – Перегляд активних працівників адміністратором

При натисканні на «Графік» адміністратору виводиться робочий графік працівників у вигляді календаря з відмітками робочих днів (рисунок 5.12). Користувач має можливість змінювати робочий графік та зберігати зміни.

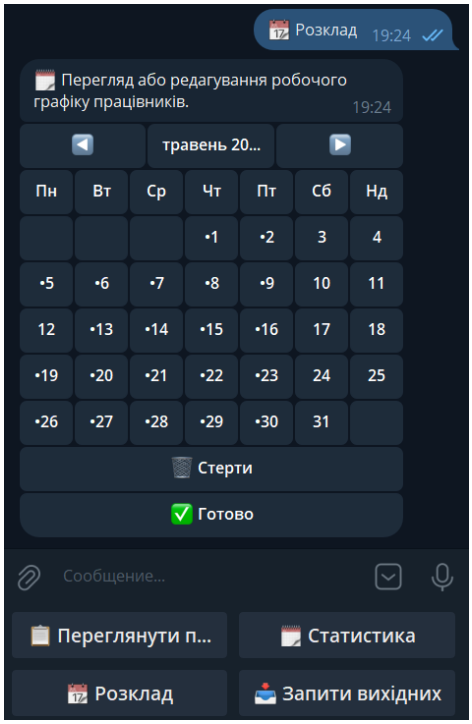

Рисунок 5.12 – Робочий календар який формує адміністратор

Кнопка «Запити вихідних» відкриває список запитів на вихідні від працівників. Адміністратор може перелік працівників та скільки вихідних вони запросили (рисунок 5.13).

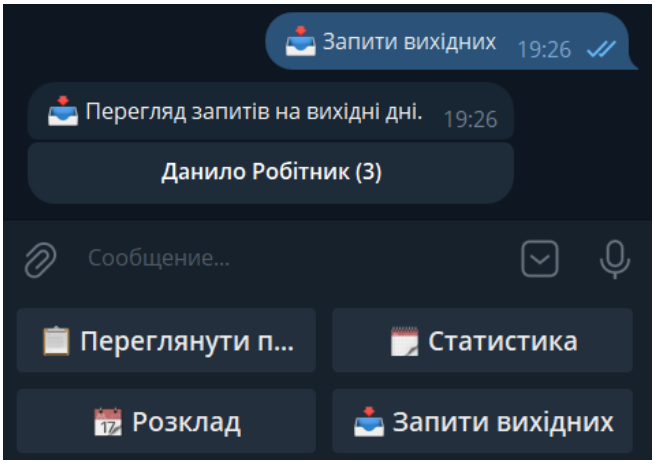

Рисунок 5.13 – Список робітників, які надіслали запит на вихідні

Натиснувши на робітника відкриється список запитів з датами, в які б цей працівник хотів би бути вільним. Адміністратор може затвердити вихідний, або проігнорувати (рисунок 5.14). Не підтверджені запити автоматично видаляться після того як настане дата на яку був надісланий запит.

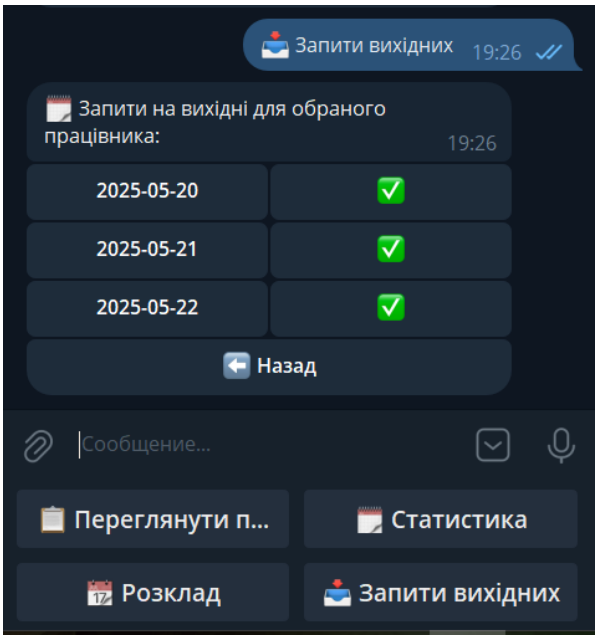

Рисунок 5.14 – Список запрошених вихідних обраного працівника

На додачу бот має функціонал, якщо робітник надсилає запит на вихідний, то цей запит прийде всім адміністраторам в чат боту. Це означає що вони будуть сповіщені і вірогідність пропустити або не побачити цей запит стає меншою.

Реалізований інтерфейс Telegram-бота був спроєктований з урахуванням простоти, зручності та логіки взаємодії користувача. Завдяки розділенню ролей «працівник» і «адміністратор» бот надає доступ лише до необхідних функцій, що мінімізує перевантаження інтерфейсу та полегшує навігацію.

Кожна дія супроводжується відповідним повідомленням або кнопками, що формує послідовний і логічний сценарій взаємодії. Візуальне оформлення з використанням inline- і reply-кнопок, повідомлень із підтвердженням та повідомлень-підказок дозволяє користувачам без додаткового навчання швидко освоїти систему.

Таким чином, інтерфейс Telegram-бота демонструє високу функціональність при збереженні простоти використання, що робить його ефективним інструментом для автоматизації контролю робочого процесу в умовах реального підприємства.

6 ПЕРСПЕКТИВИ РОЗВИТКУ СИСТЕМИ

Завершена реалізація Telegram-бота демонструє стабільну та функціональну основу для автоматизованого контролю робочих сесій персоналу. Проте, з огляду на зростаючі вимоги до систем управління персоналом, а також динаміку розвитку цифрових інструментів, постає необхідність подальшого вдосконалення проєкту. Цей розділ присвячений опису перспектив розвитку програмної системи, які охоплюють як розширення функціональних можливостей, так і підвищення зручності взаємодії користувачів із системою.

6.1 Інтеграція з веб-інтерфейсом для адміністратора

Інтеграція Telegram-бота з окремим веб-інтерфейсом для адміністратора є одним із ключових напрямків розвитку системи, що дозволить значно розширити можливості управління персоналом. Наявність веб-панелі надасть адміністративному користувачу доступ до більш широкого функціоналу, зокрема до аналітики, розширеного перегляду робочих сесій працівників, формування звітів, керування графіками та запитами. Веб-інтерфейс також спростить роботу з великими обсягами даних завдяки таблицям, фільтрам, пошуку та системам візуалізації інформації.

6.2 Автоматична генерація звітів

Впровадження механізму автоматичної генерації звітів є важливою складовою подальшої еволюції системи, що має на меті зменшення навантаження на адміністративний персонал та підвищення точності облікових операцій. Такий модуль дозволить формувати детальні звіти щодо відпрацьованого часу, кількості

вихідних днів, активності працівників, дотримання графіків та інших показників за задані часові періоди. Звіти можуть створюватися у вигляді PDF або Excel-документів і надсилатися автоматично на електронну пошту адміністратора або бути доступними для завантаження через веб-інтерфейс. В результаті, керівництво підприємства зможе оперативно отримувати достовірну інформацію для ухвалення управлінських рішень.

6.3 Обробка Live-локації працівника

Впровадження механізму обробки Live-локації працівника є одним із ключових напрямів удосконалення функціоналу Telegram-бота з позиції підвищення точності моніторингу робочих сесій. На відміну від одноразової передачі координат під час старту чи завершення зміни, трансляція поточної геолокації в реальному часі дозволяє забезпечити безперервний контроль за місцезнаходженням працівника протягом усього періоду активної сесії. Такий підхід є особливо актуальним для підприємств, де важливо контролювати пересування мобільного персоналу, зокрема водіїв, кур'єрів чи виїзних працівників.

6.4 Підтримка багатомовності

Розширення функціональності системи за рахунок впровадження підтримки багатомовності є важливим кроком на шляху до підвищення її універсальності та зручності використання в багатонаціональному середовищі. У межах цього удосконалення передбачається реалізація механізму вибору мови інтерфейсу Telegram-бота безпосередньо під час реєстрації або через налаштування профілю користувача. Завдяки використанню окремих мовних ресурсів (файлів локалізації) система зможе динамічно формувати повідомлення відповідною мовою, що значно поліпшить взаємодію з користувачем.

ВИСНОВКИ

1. В результаті аналізу існуючих підходів, методів та алгоритмів вирішення задачі було обґрунтовано використання Telegram-бота з геолокаційними технологіями для контролю робочих сесій працівників підприємства. Такий підхід дозволяє ефективно автоматизувати облік робочого часу та підвищити точність контролю.

2. На основі вивчення програмних засобів обґрунтовано застосування мови Java, середовища розробки IntelliJ IDEA, Telegram API та системи управління базами даних MySQL, що забезпечують надійність, масштабованість та можливість подальшої підтримки розробленої системи.

3. Розроблено програмну систему — «система контролю робочих сесій працівників за геоположенням, визначеним Telegram-ботом», що виконує функції реєстрації користувачів із розмежуванням ролей, ведення обліку початку та завершення робочих сесій із контролем геолокації, управління робочими графіками та обробки запитів на вихідні.

4. На основі проведеного тестування підтверджено коректність роботи розробленого програмного забезпечення, стабільність функціоналу та відповідність технічним вимогам, що свідчить про ефективність реалізації поставлених завдань.

5. Отримані результати створюють основу для подальшого вдосконалення системи, серед яких перспективними є інтеграція з веб-інтерфейсом адміністратора, автоматична генерація звітів, удосконалення обробки живої геолокації та розширення ролей користувачів для підвищення адаптивності та зручності використання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кадровий облік на підприємстві 2024. Електронний облік кадрів – Блог. Вчасно. URL: https://vchasno.ua/kadrovyi-oblik-na-pidpryiemstvi/ (дата звернення: 18.05.2025).
2. Центр впровадження та автоматизації "softinform". SoftInform | Найкращий супровід для бізнесу. URL: https://www.softinform.com.ua/news/avtomatyzatsiia-obliku-robochoho-chasu-iak-erp-systemy-dopomahaiut-orhanizuvaty-protses/ (дата звернення: 18.05.2025).
3. Системи обліку робочого часу - налаштування, обслуговування - Львів. Електронні системи охорони - Контроль доступу - Відеоспостереження - Охоронна сигналізація - Домофони - Львів. URL: https://guard-lviv.com.ua/oblik-robochoho-chasu (дата звернення: 18.05.2025).
4. Переваги ведення обліку робочого часу в хмарі - USAP.ONLINE. USAP.ONLINE - Simple accounting platform. URL: https://usap.online/blog/perevahy-vedennia-obliku-robochoho-chasu-v-khmari/ (дата звернення: 26.05.2025).
5. Програмне забезпечення для моніторингу співробітників: вибір найкращої альтернативи табельного обліку для вашого бізнесу. Employee Monitoring Software CleverControl. URL: https://clevercontrol.com/uk/how-to-choose-a-time-doctor-alternative/ (дата звернення: 26.05.2025).
6. Time Doctor | Workforce analytics for distributed teams. Time Doctor | Workforce analytics for distributed teams. URL: https://www.timedoctor.com/ (дата звернення: 26.05.2025).
7. HR-Software-LÃsungen | cloudbasiertes personalverwaltungssystem | hr-system | zoho people. Zoho. URL: https://www.zoho.com/people/ (дата звернення: 18.05.2025).
8. Employee scheduling, HR & workforce management software – deputy. Deputy. URL: https://www.deputy.com/ (дата звернення: 18.05.2025).

9. Time tracking and productivity monitoring tool | hubstaff. Hubstaff. URL: https://hubstaff.com/ (дата звернення: 18.05.2025).
10. TMetric - time tracking software. TMetric: Free Time Tracking Software for High-Performing Teams. URL: https://tmetric.com/ (дата звернення: 26.05.2025).
11. Ramesh Kumar, Anjali Singh. "Smart Attendance System Using IoT and GPS for Remote Workforce Management." International Journal of Advanced Computer Science and Applications, vol. 11, no. 5, 2024.
12. Sagar Mane, Atharva Bharambe, Mihir Gawade, Nikhil Gandhi, Suyash Bhanwase. "Geolocation-Based Employee Attendance and Tracking System." International Research Journal of Modernization in Engineering Technology and Science, vol. 6, no. 11, November 2024. URL: https://www.irjmets.com/uploadedfiles/paper//issue_11_november_2024/63241/final/fin_irjmets1730650696.pdf .
13. Вибір першої мови програмування – огляд Java: Стаття з блогу IT-школи Hillel. Корисні матеріали: Статті та новини IT-індустрії | Комп'ютерна школа Hillel. URL: https://blog.ithillel.ua/articles/vybir-pershoi-movy-prohramuvannia-ohliad-java (дата звернення: 18.05.2025).
14. Що таке Java і де вона використовується: Стаття з блогу IT-школи GoIT. Корисні матеріали: Статті та новини IT-індустрії | Школа програмування GoIT. URL: https://goit.global/ua/articles/shcho-take-java-i-de-vona-vykorystovuietsia/ (дата звернення: 18.05.2025).
15. Java IntelliJ IDEA. Уроки для початківців. W3Schools українською. W3Schools українською. Безплатні уроки онлайн для початківців, школярів та студентів. URL: https://w3schoolsua.github.io/hyperskill/intellij-idea.html (дата звернення: 18.05.2025).
16. Що таке API (Application Programming Interface), як і де працює: Стаття з блогу IT-школи GoIT. Корисні матеріали: Статті та новини IT-індустрії | Школа програмування GoIT. URL: https://goit.global/ua/articles/application-programming-interface-api-shcho-tse-iak-i-de-pratsiuie/ (дата звернення: 18.05.2025).

17. Java Telegram Bot – бібліотека для створення Telegram-ботів: Обговорення на форумі DOU. DOU – Спільнота українських IT-спеціалістів. URL: https://dou.ua/forums/topic/38358/ (дата звернення: 18.05.2025).
18. Системи управління базами даних СУБД для малого та середнього бізнесу. Kyivstar Business Hub – корпоративний блог для бізнесу. URL: https://hub.kyivstar.ua/articles/sistemi-upravlinnya-bazami-danih-dlya-malogo-ta-serednogo-biznesu-yak-vibrati (дата звернення: 18.05.2025).
19. Система управління базами даних MySQL: Стаття з блогу Lemon School. Корисні матеріали: Статті про програмування та IT | Lemon School. URL: https://lemon.school/blog/systema-upravlinnya-bazamy-danyh-mysql (дата звернення: 18.05.2025).
20. MySQL Посібник Workbench: що таке, як встановити та використовувати. Guru99. URL: https://www.guru99.com/uk/introduction-to-mysql-workbench.html (дата звернення: 18.05.2025).
21. Підручник maven для java-проектів - стебло 2025. Fused Learning. URL: https://uk.fusedlearning.com/maven-tutorial-java-projects (дата звернення: 18.05.2025).
22. SLF4J – Simple Logging Facade for Java: Офіційний сайт бібліотеки. URL: https://www.slf4j.org/ (дата звернення: 18.05.2025).
23. Tovma O. Основні типи архітектури програмного забезпечення ≡ Блог ArtofBA. ArtofBA. URL: https://www.artofba.com/uk/post/main-types-of-software-architecture (дата звернення: 18.05.2025).

ДОДАТОК А

ТЕКСТ ПРОГРАМНОГО МОДУЛЯ

«РЕАЛІЗАЦІЯ СЕРВІСУ ДЛЯ ОБРОБКИ ГЕОЛОКАЦІЇ ТА НАДСИЛАННЯ НАГАДУВАНЬ ПРО ПІДТВЕРДЖЕННЯ МІСЦЕПОЛОЖЕННЯ»

УРК.НТУУ”КПІ ім. Ігоря Сікорського”_ІАТЕ_ЦТЕ_ТР-11

Аркушів 4

Київ – 2025

Програмні засоби:

– мова програмування – Java;

– бібліотека TelegramBots – взаємодія з Telegram API для надсилання повідомлень, обробки команд та роботи з геолокацією;

– обробка локації користувача – клас LocationService

– таймерна логіка з перевіркою активності – клас LocationReminderService

– взаємодія з базою даних – клас DBService, підключення до СУБД MySQL.

Клас LocationService виконує функцію перевірки географічного розташування працівника відносно заздалегідь заданої контрольної точки, яка умовно визначає межі робочої зони (наприклад, офіс). Основним методом класу є isLocationInRange, що реалізує обчислення відстані між координатами, отриманими від користувача, та координатами офісу із застосуванням формули великого кола. Якщо відстань не перевищує допустимого радіуса, вважається, що працівник перебуває у межах робочої території

```
package com.danylo.bot.service.location;

public class LocationService {

   private final double officeLatitude = 52.3831;  // Широта ()
   private final double officeLongitude = 16.8900; // Довгота ()
   private final double allowedRadius = 0.55; // Допустимий радіус (в кілометрах, 0.5 км = 500 м)

   //перевірка, чи локація в межах допустимого радіусу
   public boolean isLocationInRange(double userLatitude, double userLongitude) {
      final double R = 6371; // Радіус Землі в км

      // Обчислюємо відстань між точкою працівника і заданою точкою (офісом)
      double latDistance = Math.toRadians(userLatitude - officeLatitude);
      double lonDistance = Math.toRadians(userLongitude - officeLongitude);
      double a = Math.sin(latDistance / 2) * Math.sin(latDistance / 2)
            + Math.cos(Math.toRadians(officeLatitude)) * Math.cos(Math.toRadians(userLatitude))
            * Math.sin(lonDistance / 2) * Math.sin(lonDistance / 2);
      double c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
      double distance = R * c; // Відстань в км
      System.out.println("Відстань до офісу: " + distance);
      return distance <= allowedRadius;
   }
}
```

Клас LocationReminderService відповідає за реалізацію механізму періодичних нагадувань працівникам про необхідність підтвердження своєї присутності за допомогою надсилання геолокації. Основна функціональність цього класу полягає в запуску циклічних таймерів, які через задані інтервали часу надсилають користувачу повідомлення з проханням надіслати свою локацію. У випадку відсутності підтвердження протягом заданого часу, активується одноразовий таймер, який автоматично завершує зміну працівника.

```
package com.danylo.bot.service.location;

import com.danylo.bot.EmployeeBot;
import com.danylo.bot.chatMessage.TimerMessageService;
import com.danylo.bot.keyboards.ReplyKeyboards;
import com.danylo.bot.db.DBService;
import org.telegram.telegrambots.meta.api.methods.send.SendMessage;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;

import java.util.Map;
import java.util.concurrent.*;
import java.util.List;
import java.util.concurrent.CopyOnWriteArrayList;

public class LocationReminderService {

   private static final Map<Long, ScheduledFuture<?>> reminderTasks = new
ConcurrentHashMap<>();
   private static final Map<Long, ScheduledFuture<?>> timeoutTasks = new ConcurrentHashMap<>();
   private static final ScheduledExecutorService scheduler = Executors.newScheduledThreadPool(2);

   private static final List<Long> confirmedUsers = new CopyOnWriteArrayList<>();
   private static int timerTime = 30;

   public static boolean isReminderScheduled(long userId) {
      return reminderTasks.containsKey(userId);
   }

   public static boolean isTimeoutRunning(long userId) {
      return timeoutTasks.containsKey(userId);
   }

   public static void markUserConfirmed(long userId) {
      if (!confirmedUsers.contains(userId)) {
         confirmedUsers.add(userId);
      }
   }

   public static void unmarkUserConfirmed(long userId) {
```

```
        confirmedUsers.remove(userId);
    }

    public static boolean isUserConfirmed(long userId) {
        return confirmedUsers.contains(userId);
    }

    // Циклічний запуск таймера з нагадуванням про надсилання локації
    public static void startLocationReminders(EmployeeBot bot, long userId, long chatId) {
        stopLocationReminders(userId);

        System.out.println("Запуск таймера нагадувань");
        Runnable reminderTask = () -> {
            System.out.println("Надсилання нагадування про локацію");
            SendMessage message = new SendMessage();
            message.setChatId(chatId);
            message.setText("Будь ласка, надішліть вашу локацію для оновлення присутності.");
            message.setReplyMarkup(ReplyKeyboards.getLocationButton());
            try {
                bot.execute(message);
            } catch (TelegramApiException e) {
                e.printStackTrace();
            }

            // Запускаємо одноразовий таймер на 30 секунд для підтвердження
            markUserConfirmed(userId);
            startTimeoutTimer(bot, userId, chatId);
            TimerMessageService.startTimer(bot, chatId, timerTime, "підтвердження локації");
        };

        ScheduledFuture<?> scheduled = scheduler.scheduleAtFixedRate(reminderTask, 1, 1,
TimeUnit.MINUTES);
        reminderTasks.put(userId, scheduled);
    }

    public static void stopLocationReminders(long userId) {
        ScheduledFuture<?> task = reminderTasks.get(userId);
        if (task != null) {
            task.cancel(true);
            reminderTasks.remove(userId);
            unmarkUserConfirmed(userId);
        }
    }

    public static void cancelTimeout(long userId) {
        ScheduledFuture<?> timeout = timeoutTasks.get(userId);
        if (timeout != null) {
            timeout.cancel(true);
            timeoutTasks.remove(userId);
        }
    }
```

```
    // Перезапуск циклічного таймера після успішного надсилання локації
    public static void restartReminderTimer(EmployeeBot bot, long userId, long chatId) {
        System.out.println("Локація підтверджена. Перезапускаємо цикл нагадувань...");
        cancelTimeout(userId);
        unmarkUserConfirmed(userId);
        startLocationReminders(bot, userId, chatId);
    }

    // Одноразовий таймер на випадок, якщо користувач не надіслав локацію
    private static void startTimeoutTimer(EmployeeBot bot, long userId, long chatId) {
        cancelTimeout(userId);

        Runnable timeoutTask = () -> {
            System.out.println("Локація не підтверджена. Завершуємо зміну для користувача: " +
userId);
            SendMessage message = new SendMessage();
            message.setChatId(chatId);
            message.setText("Локація не підтверджена. Завершуємо зміну для користувача.");
            message.setReplyMarkup(ReplyKeyboards.getEmployeeKeyboard());

            boolean successEnd = DBService.logWorkEnd(userId);
            stopLocationReminders(userId);

            try {
                bot.execute(message);
            } catch (TelegramApiException e) {
                e.printStackTrace();
            }
        };

        ScheduledFuture<?> timeoutFuture = scheduler.schedule(timeoutTask, timerTime + 6,
TimeUnit.SECONDS);
        timeoutTasks.put(userId, timeoutFuture);
    }
}
```