Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Центр післядипломної освіти

(повна назва)

Кафедра       Програмної інженерії  

(повна назва)

**АТЕСТАЦІЙНА РОБОТА**

**Пояснювальна записка**

рівень вищої освіти перший (бакалаврський)       

   Веб-система підтримки діяльності бюро перекладів 

(тема)

Виконав:

студент  4  курсу, групи     ПЗПІпз-18-1    

        Дубодєлов К. В.           

(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного

              забезпечення            

(код і повна назва спеціальності)

Освітня програма Програмна інженерія       

(повна назва освітньої програми)

Керівник    доц. Ворочек О.Г.        

(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри \_\_\_\_\_\_\_\_\_\_\_\_\_         З.В. Дудар

(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Центр післядипломної освіти

Кафедра Програмної інженерії

Рівень вищої освіти перший (бакалаврський)

Спеціальність 121 - Інженерія програмного забезпечення

(код і повна назва)

Освітня програма Програмна інженерія

(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри \_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис)

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20 20 р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові      Дубодєлову Кирилу Володимировичу

(посада, прізвище, ініціали)

1. Тема роботи: \_ Веб-система підтримки діяльності бюро перекладів\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

затверджена наказом по університету від 15 05 2020 р. № 58Стз\_

2. Термін подання студентом роботи до екзаменаційної комісії \_15\_ \_06\_\_\_ 2020 р.

3. Вихідні дані до роботи розробити веб-портал для взаємодії різних категорій учасників навчального процесу, середовище проектування та розробки JetBrains WebStorm 2019.2.2, платформа NodeJS, мова програмування JavaScript, СКБД MongoDB, ОС Windows 10\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Перелік питань, що потрібно опрацювати в роботі вступ, аналіз предметної області та постановка задачі, формування вимог до програмної системи, архітектура та проектування програмного забезпечення, опис прийнятих програмних рішень, опис роботи ПЗ, тестування ПЗ. Додатки: а) слайди презентації б) Software Requirements Specification в) приклади кодів програм, д) тези е) електронний архів\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва етапів роботи | Терміни  виконання етапів роботи | Примітка |
| 1 | Аналіз предметної галузі | 27.04.2020 | виконано |
| 2 | Формування вимог до системи | 04.05.2020 | виконано |
| 3 | Проектування системи | 11.05.2020 | виконано |
| 4 | Розробка програмного забезпечення | 18.05.2020 | виконано |
| 5 | Тестування системи | 20.05.2020 | виконано |
| 6 | Впровадження системи | 22.05.2020 | виконано |
| 7 | Підготовка пояснювальної записки | 25.05.2020 | виконано |
| 8 | Підготовка презентації та доповіді | 25.05.2020 | виконано |
| 9 | Перевірка на плагіат | 25.05.2020 | виконано |
| 10 | Нормоконтроль | 25.05.2020 | виконано |
| 11 | Рецензування | 29.05.2020 | виконано |
| 12 | Занесення диплома в електронний архів | 01.06.2020 | виконано |
| 13 | Попередній захист | 01.06.2020 | виконано |
| 14 | Допуск до захисту у зав. кафедри | 01.06.2020 | виконано |

Дата видачі завдання «21\_» \_квітня\_\_\_\_\_\_\_\_\_\_2020 р.

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис)

Kepiвник роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_доц. Ворочек О.Г.\_\_\_

(підпис) (посада, прізвища, ініціали)

**РЕФЕРАТ / ABSTRACT**

Пояснювальна записка до атестаційної роботи бакалавра: 85 с., 20 рис., 3 табл., 15 джерел.

БЮРО ПЕРЕКЛАДІВ, АВТОМАТИЗОВАНІ СИСТЕМИ УПРАВЛІННЯ БІЗНЕСОМ, G SUITE, NOSQL, GOOGLE APPS SCRIPT.

Об’єктом дослідження є система управління бюро перекладів.

Метою роботи є проектування та програмна реалізація веб дадатка, призначеного для повноцінного й повного управління основними робочими завданнями менеджера проектів в перекладацьких бюро, які використовують сервіси G Suite.

Методи розробки базуються на технологіях Javascript (Google Apps Script).

У результаті роботи спроектовано та розроблено програмний продукт по управлінню робочими завданнями менеджера проектів бюро перекладів середнього та маленького розміру.

# Зміст

[Зміст 5](#_Toc42973764)

[Вступ 7](#_Toc42973765)

[1 Аналіз предметної галузі 9](#_Toc42973766)

[1.1 Аналіз предметної галузі 9](#_Toc42973767)

[1.2 Аналіз наявних рішень 10](#_Toc42973768)

[1.3 Виявлення проблем та актуалізація рішень 18](#_Toc42973769)

[1.4 Постановка задачі 21](#_Toc42973770)

[2 Формування вимог для програмної системи 24](#_Toc42973771)

[2.1 Функціональні вимоги 24](#_Toc42973772)

[2.2 Нефункціональні вимоги 26](#_Toc42973773)

[3 Архітектура та проектування програмного забезпечення 28](#_Toc42973774)

[3.1 UML проектування ПЗ 28](#_Toc42973775)

[3.2 Вибір системи зберігання даних 32](#_Toc42973776)

[3.3 Проектування системи зберігання даних 35](#_Toc42973777)

[3.4 Приклади найцікавіших алгоритмів і методів 37](#_Toc42973778)

[3.5 Проектування UI системи 40](#_Toc42973779)

[4 Опис прийнятих програмних рішень 48](#_Toc42973780)

[4.1 Основний застосунок 48](#_Toc42973781)

[4.2 Модуль для роботи із сервісом Gmail 49](#_Toc42973782)

[4.3 Модуль для роботи із сервісом Google Sheets 53](#_Toc42973783)

[4.4 Модуль для роботи із сервісом Google Drive 55](#_Toc42973784)

[4.5 Модуль для роботи із сервісом Google Calendar 56](#_Toc42973785)

[5 Тестування розробленого програмного забезпечення 58](#_Toc42973786)

[5.1 Функціональне тестування навігації між картками 58](#_Toc42973787)

[5.2 Функціональне тестування контекстних карток 60](#_Toc42973788)

[5.3 Нефункціональне тестування продуктивності 61](#_Toc42973789)

[5.4 Тестування безпеки 62](#_Toc42973790)

[6 Впровадження програмного забезпечення 63](#_Toc42973791)

[6.1 Lingotek VendorMailer 63](#_Toc42973792)

[6.2 TranslateOn Mailer 65](#_Toc42973793)

[Висновки 67](#_Toc42973794)

[Перелік джерел посилань 68](#_Toc42973795)

[Додаток А Слайди презентації 70](#_Toc42973796)

[Додаток Б Приклад програмного коду 71](#_Toc42973797)

[Додаток В Електронні матеріали 76](#_Toc42973798)

# Вступ

Сьогодні майже всі галузі, особливо в міжнародному бізнесі, залежать від ефективності швидкості й ефективності комунікацій. Якість і досягнення цілей комунікацій забезпечуються завдяки професійним бюро перекладів та командам перекладачів.

Протягом останніх десятиліть у світі панує цифровізація, тенденціям якої підвладні також і перекладацькі компанії. Наразі все поширенішою стає тенденція відмови від штатних перекладачів та співпраці з лінгвістами-фрілансерами, які можуть знаходитись будь-де за межами офісу, міста та навіть країни розташування компанії-замовника. У великих бюро перекладів до 75% складають позаштатні перекладачі та редактори [1], у перекладацьких компаніях меншого розміру це значення може сягати 100%. З одного боку, збільшення кількості позаштатних спеціалістів дозволяє суттєво розширити асортимент пропонованих послуг, підвищити їхню якість, скорочує накладні витрати. Для лінгвістів подібний підхід означає можливість комфортної праці в себе вдома, розширення клієнтури та вільний графік.

Але збільшення кількості фрілансерів призводить до виникнення складніших завдань для менеджерів проектів у бюро перекладів. Проблема полягає в тому, що десятиріччя тому такий менеджер підтримував контакт лише з одним-двома десятками лінгвістів (зазвичай штатних). Сьогодні кількість спеціалістів, з якими доводиться працювати менеджеру проектів у бюро перекладів середнього розміру, вимірюється тисячами. Це потребує принципово нового підходу до організації комунікації між менеджером проектів та лінгвістами, а також до нових процедур розподілу проектів і збереження даних про виконавців, проекти та кінцевих замовників.

Для того щоб задовольнити ці потреби, були розроблені різноманітні програмні рішення. На жаль, частина із цих рішень має завищену вартість користування й обслуговування, тому недоступна для малих і середніх підприємств, або має обмежену функціональність, що не відповідає вимогам для ефективної комерційної діяльності у сфері перекладів.

Мета цієї роботи — спроектувати та розробити інформаційну систему, призначену для повноцінного й повного управління основними робочими завданнями менеджера проектів в перекладацьких бюро, які використовують сервіси G Suite.

Під час роботи розроблена інформаційна система, яка допомагає менеджеру проектів бюро перекладів ефективно виконувати основні робочі завдання: листування із замовниками та виконавцями, створення проектів та основних поточних робочих документів. Для реалізації системи було обрано мову програмування Google Apps Script.

# Аналіз предметної галузі

## Аналіз предметної галузі

Починаючи з 2000-х років сфера надання лінгвістичних послуг суттєво змінилася завдяки розвитку інтернет-комунікацій. В останні роки бюро перекладів змогли значно скоротити витрати на штатних перекладачів, користуючись послугами позаштатних фахівців у галузі перекладу, яким не потрібно оплачувати соціальне страхування та постійну зарплатню, а також не доводиться орендувати офіс для штатних спеціалістів. Через це більшість перекладацьких компаній надають перевагу співпраці з фрілансерами, скорочуючи штат перекладачів або повністю відмовляючись від нього. Також це дозволяє бюро перекладів розширити перелік послуг, що надаються, мовних пар та отримати нових замовників. Але водночас цей процес у рази збільшує обсяги даних, з якими щодня працюють менеджери бюро перекладів.

Подібна трансформація суттєво змінила як правила гри на ринку перекладацьких послуг, так і структуру самих бюро перекладів. Одночасно зі збільшенням кількості позаштатних фахівців, їхня роль у загальному процесі виконання проекту з перекладу починає зменшуватися. На перший план виходить працівник нового типу — менеджер проектів бюро перекладів. Це зовсім не той самий співробітник, який ще наприкінці ХХ сторіччя розподіляв завдання між кількома штатними перекладачами. Тепер менеджер проектів щодня працює з десятками або ж навіть сотнями спеціалістів. І основу його роботи становить спілкування за допомогою електронної пошти. «At its most basic, managing expectations is about proactive, clear, and explicit communication.»[2]. Від того, наскільки ефективним, швидким та прозорим буде процес комунікації менеджера проектів бюро перекладів із замовниками та виконавцями, залежить успіх та прибуток усієї компанії.

## Аналіз наявних рішень

Нові завдання потрібно виконувати за допомогою нового програмного забезпечення. Наразі можна умовно виділити три основні групи програмного забезпечення, з якими працюють менеджери проектів бюро перекладів.

### СAT Tools

СAT Tools (перекладацький комп’ютерний інструментарій) — текстові редактори, в які додано спеціальні можливості: робота з пам’яттю перекладу, глосаріями тощо. Прикладами такого перекладацького забезпечення є SDL Trados Studio, MemoQ, Wordfast. Подібні програми в першу чергу створені для полегшення праці перекладача завдяки забезпеченню збільшення швидкості роботи, зокрема, з типовими документами. Вихідний текст у таких програмах поділяється на сегменти, і під час роботи виконавець перекладає сегмент за сегментом. Переклад кожного сегменту зберігається у файлі спеціального типу (так звана «пам’ять перекладу»), і якщо у вихідному тексті зустрічається аналогічний (або подібний) сегмент, програма автоматично запропонує збережений варіант перекладу.

Але в подібних програмах закладені деякі функції й для менеджерів проекту (див. рис. 1.1). У першу чергу, це можливість створювати проектний пакет (по суті, модифікований архів, що містить усі робочі файли для перекладу: файли оригіналу, пам’ять перекладу, глосарії, статистичні дані проекту, допоміжні матеріали тощо). Створення такого пакету дозволяє вислати виконавцю одноразово всі файли, пов’язані з проектом, а також відстежувати прогрес роботи онлайн (за умови активованого режиму роботи з проектами через сервер).

Також такі програми дозволять менеджеру проектів отримати детальну статистику про вихідні файли та на основі цих даних розрахувати бюджет проекту як для замовника, так і для кожного виконавця.

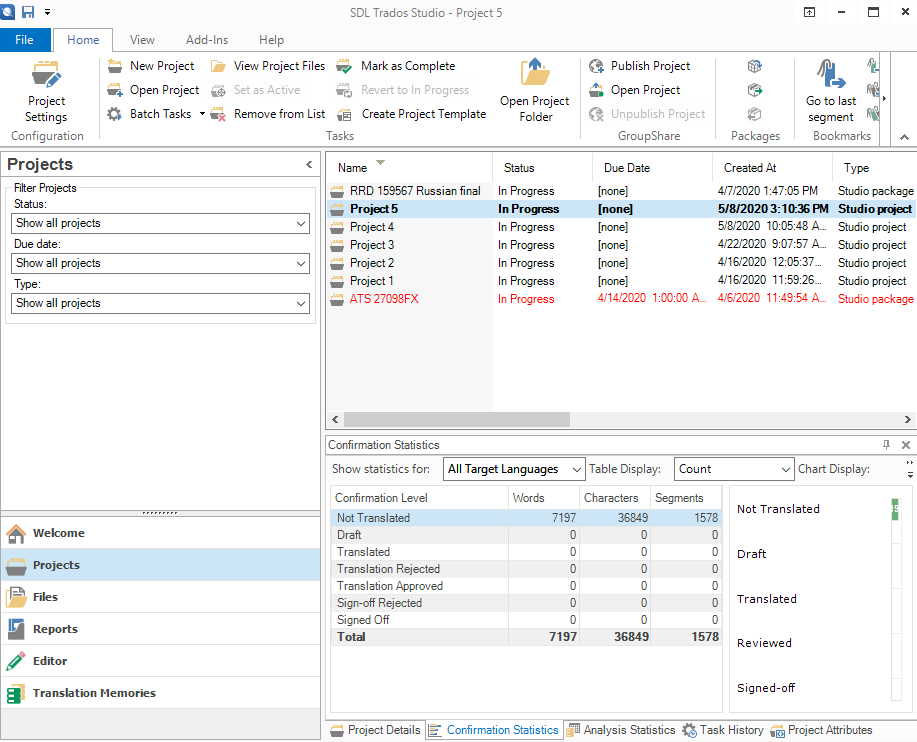


Рисунок 1.1 – Інтерфейс програми SDL Trados Studio

Основними недоліками такого програмного забезпечення є велика вартість (наприклад, підписка на SDL Trados 2019 для бюро перекладів становить 2495 євро на одного користувача на рік)[3] та, що важливіше, абсолютна відсутність інструментів для комунікації з перекладачами та редакторами. Це призводить до дублювання інформації та виникнення помилок (менеджер проектів, який стикається щодня з одним-двома десятками проектів, може просто загубити або переплутати файл, що матиме сумні наслідки для проекту).

### Системи управління бізнесом

Системи управління бізнесом — це пакет програм, що не призначений безпосередньо для перекладу. Їхня мета — ведення бази клієнтів та виконавців, у яких містяться тарифи, контактні дані, інформація про спеціалізацію, мовні пари, отримання замовлень, видача замовлень виконавцям, фінансовий облік і контроль, створення рахунків-фактур, моніторинг сплати тощо. Такі програми дозволяють менеджеру проектів бюро перекладів контролювати діяльність перекладачів і розподіляти проекти. Прикладами таких рішень є Plunet (розробник — Plunet GmbH, Німеччина), XTRF (розробник — XTRF Management Systems S.A., Польща), Netsuite (розробник — Oracle, США).

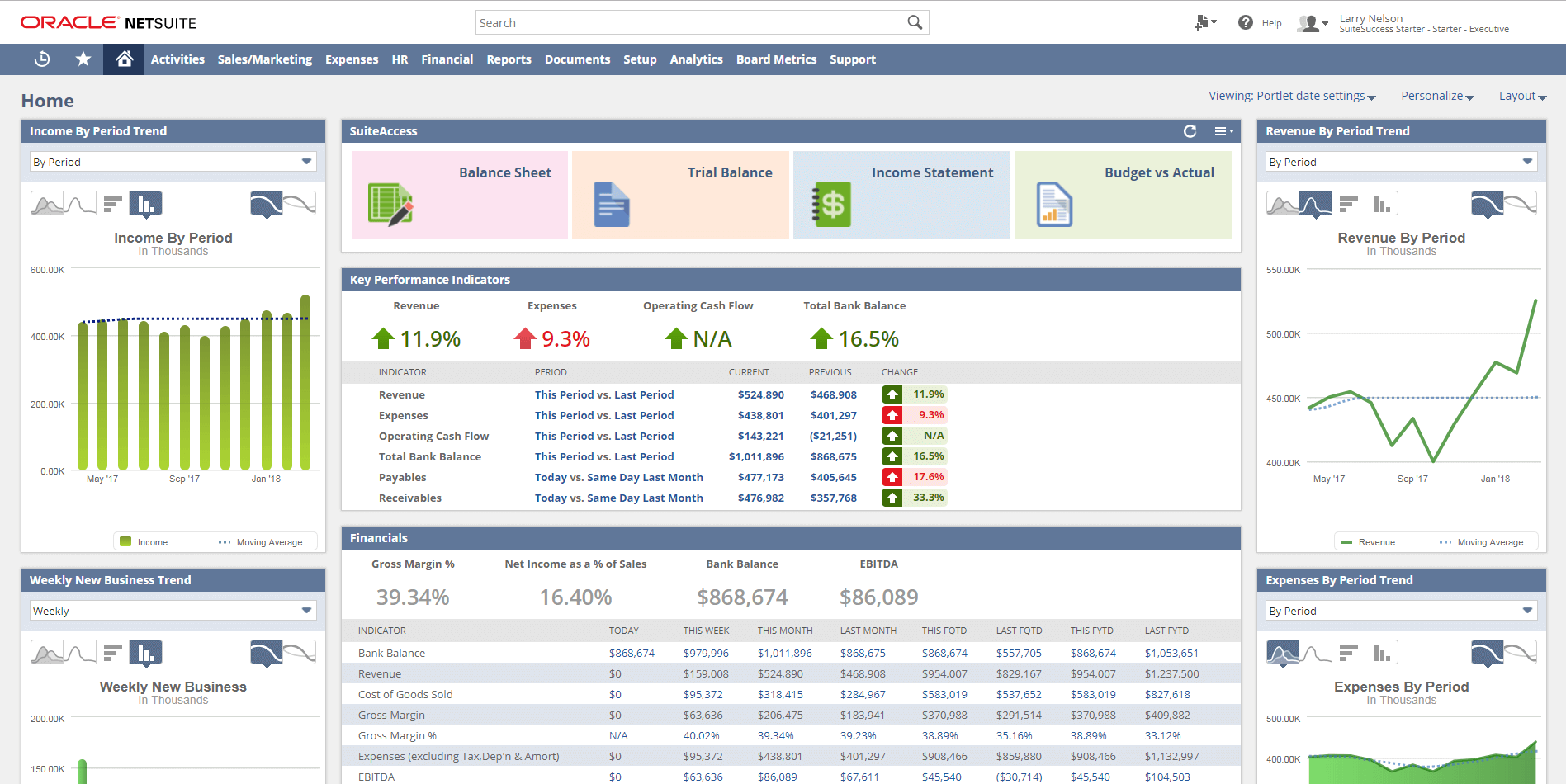


Рисунок 1.2 – Інтерфейс веб-сервісу Netsuite (Oracle)

Основними недоліками таких програм є відсутність інструментів для комунікації та синхронізації контактів з основними поштовими сервісами. У результаті в менеджера проектів завжди є не менше двох баз із даними виконавців: одна в поштовому сервісі та кілька — у подібних програмах. Синхронізація цих баз можлива лише за допомогою переносу інформації вручну, що також не гарантує відсутності помилок.

Також треба зазначити, що інтерфейс подібних програмних рішень переобтяжений (див. рис. 1.2), що призводить до ряду складностей у роботі менеджера проектів бюро перекладів.

### Гібридне програмне забезпечення

Гібридне програмне забезпечення — програми, що поєднують властивості перекладацького комп’ютерного інструментарію та систем управління бізнесом (із акцентом на функціональні можливості перекладацького комп’ютерного інструментарію). Найяскравішим прикладом такого типу програм є SmartCAT від ТОВ «Смарткат» і LTMS від Lingotek Inc (див. рис. 1.3).

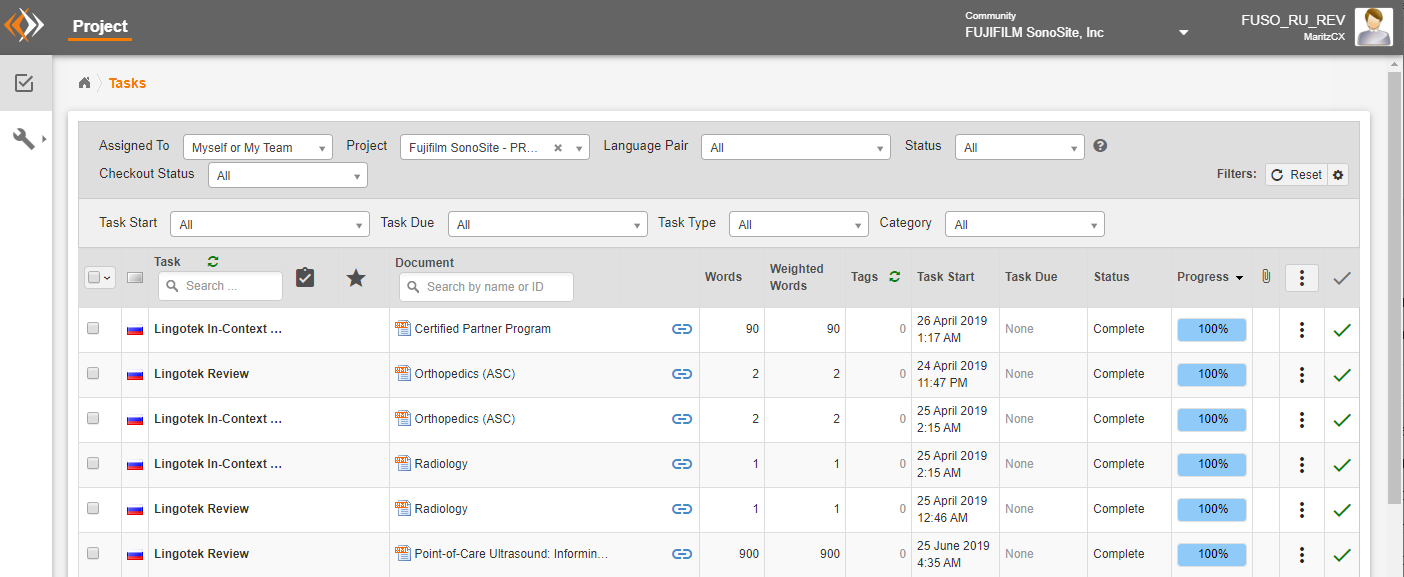


Рисунок 1.3– Інтерфейс веб-сервісу LTMS (Lingotek)

У першу чергу, це перекладацький комп’ютерний інструментарій для роботи онлайн, що поєднує базу перекладачів і функції біржі перекладів [4].

Такі рішення бувають дуже корисними у випадках термінових проектів, або коли штатна команда перекладачів завантажена роботою та потрібно залучити сторонні ресурси. Постійне використання таких платформ ускладнене, адже фінансова модель таких рішень передбачає відрахування до 20% від загального бюджету проекту на користь власників платформи [5].

Через відсутність універсального рішення великі перекладацькі компанії вимушені створювати власні програми для керування проектами. Прикладами таких рішень можуть бути Project Director компанії Transperfect (США) та Multitrans компанії RR Donneley (Велика Британія). Основними недоліками таких програм є те, що вони не є універсальними, структурно складні через переобтяжений інтерфейс, і для їхнього використання потрібне додаткове довгострокове навчання користувача перед початком роботи. Також ці програми використовуються лише тими перекладацькими компаніями, для потреб яких вони були розроблені, і вони не є поширеними програмними рішеннями.

Ці рішення мають дуже велику вартість щодо розробки та підтримки, тому дозволити собі їх можуть лише великі транснаціональні перекладацькі компанії (наразі нам відомо лише п’ять подібних компаній у всьому світі). Менеджери середніх та малих бюро перекладів вимушені користуватися кількома видами стороннього програмного забезпечення, кожне з яких лише частково відповідає потребам, що виникають під час щоденної роботи. Інформація про замовників і виконавців додається до кількох розрізнених баз даних, результатом чого є дублювання даних і можливі неточності та помилки, а також неефективно витрачений робочий час менеджерів проектів бюро перекладів. У випадку малих бюро перекладів та груп фрілансерів навіть часто трапляється ситуація, коли всю робочу інформацію про проекти просто вручну записують у документ Excel або Google Sheet.

Жоден із представлених на ринку видів програмного забезпечення за доступною ціною не має функції синхронізації контактів із розповсюдженими поштовими сервісами та календарями.

Лише два представники серед поширених видів програмного забезпечення для менеджерів проектів мають функцію створення листів електронної пошти — Plunet та XTRF. Оскільки в сучасних реаліях основним завданням менеджера проектів є саме листування за допомогою електронної пошти, розглянемо саме їх.

### Система Plunet

Plunet — це комплексна мережева система управління перекладацької компанією. Розроблена німецькою компанією Plunet GmbH (Німеччина) (див. рис. 2.4).

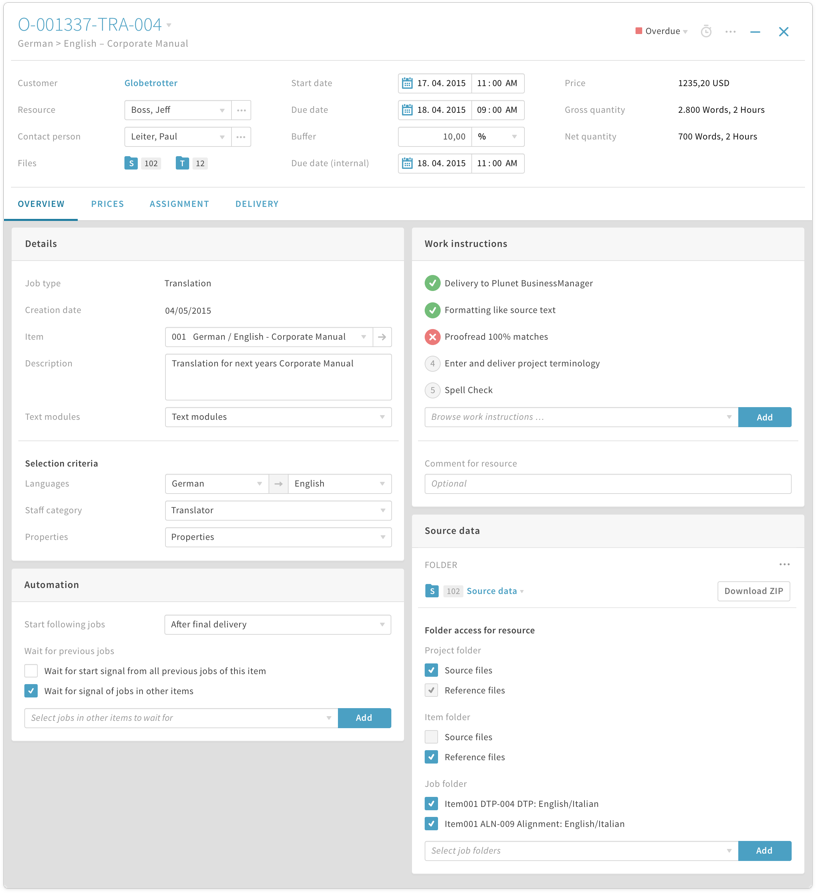


Рисунок 1.4– Інтерфейс створення проекту в сервісі Plunet

Основні переваги системи:

* можливість автоматизації всіх основних етапів бізнесу у сфері надання лінгвістичних послуг;
* можливість інтеграції (за допомогою платного плагіну) з основними CAT-програмами (SDL Trados Studio, MemoQ, Memsource);
* можливість (за допомогою платного плагіну) підключення сервісу розсилок електронною поштою.

Основні недоліки системи Plunet:

* висока вартість;
* складний інтерфейс;
* необхідність навчання робочого персоналу;
* відсутність можливості налаштування документів і звітів під потреби певної компанії;
* відсутність можливості синхронізації контактів і листів з основними сервісами електронної пошти.

Загалом Plunet є найпоширенішою системою управління бізнесом бюро перекладів у Європі [6]. Але через перелічені недоліки менеджери проектів вимушені вручну синхронізувати контакти зі своїми поштовими сервісами. Через високу вартість цю систему обирають бюро середнього та великого розміру.

### Система XTRF

Система XTRF була створена компанією XTRF Management Systems S.A. (Польща). Це комплексне рішення для управління проектами в бюро перекладів.

Основні переваги системи:

* вбудована в систему CAT-програма;
* можливість створення поштових розсилок для перекладачів;
* можливість налаштування системи під потреби компанії;
* потужна система генерації звітів;
* система створення основних документів за стандартами Євросоюзу.

Основні недоліки системи:

* дуже висока вартість (від 85 доларів США на місяць для одного користувача) [7];
* нестабільність в роботі серверів компанії;
* складний і дорогий процес налаштування системи під потреби певної компанії;
* обмежений функціонал вбудованої CAT-програми;
* складний інтерфейс (див. рис. 2.5).

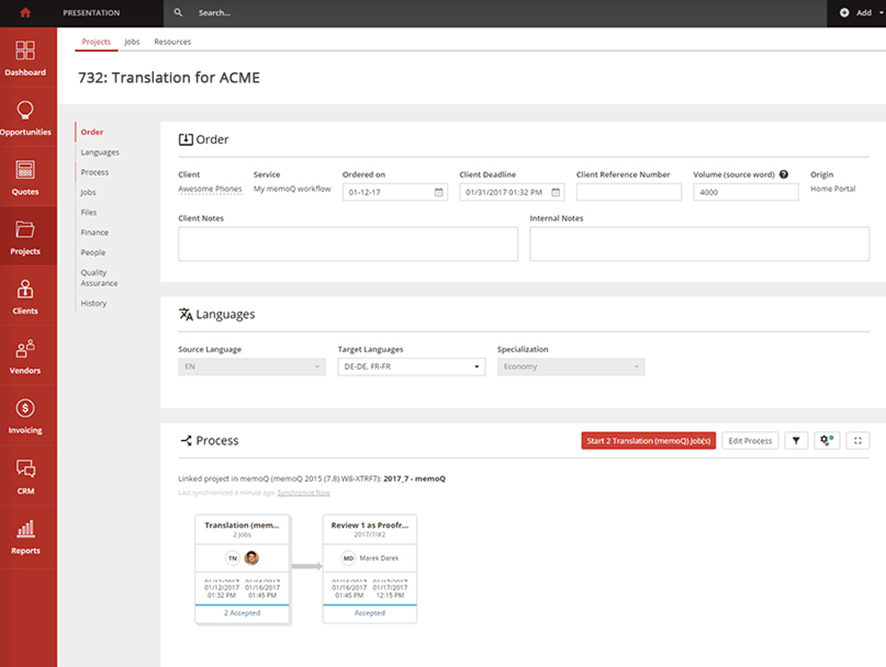


Рисунок 1.5– Інтерфейс створення проекту в системі XTRF

Також обидві згадані програми мають суттєвий недолік при використанні їх у роботі бюро перекладів невеликого та середнього розмірів. Основними замовниками подібних компаній є великі транснаціональні бюро перекладів, що зазвичай наполягають на виконання проектів у їхніх власних мережевих системах для роботи. При цьому і Plunet, і XTRF для початку роботи над проектом потребують вихідні файли, які компанія-замовник може просто не надати, обмежившись лише посиланням на проект у власній системі. У такому разі менеджер проекту в компанії-виконавиці змушений власноруч вносити всі дані про проект у власну систему, а перекладачам передавати посилання на роботу в системі компанії-замовника. Звісно, це часто призводить до помилок, та не є нормальним підходом до процесу організації роботи.

Після проведення аналізу наявного програмного забезпечення для роботи менеджерів проектів бюро перекладів можна зробити висновки, що не існує системи, яка б повною мірою відповідала сучасним потребам користувачів. Тому більшості менеджерів проектів у бюро перекладів доводиться одночасно працювати з кількома програмними рішеннями, кожне з яких лише частково відповідає потребам у роботі. Також жодна із представлених на ринку систем не інтегрована в сервіси електронного листування, що в сучасних реаліях перекладацького бізнесу призводить до помилок у процесі комунікації із замовниками та виконавцями, а це є критичним для нормального та ефективного робочого процесу бюро перекладів.

## Виявлення проблем та актуалізація рішень

Основне завдання менеджера проектів сучасного бюро перекладів — комунікація із замовником і виконавцями. Саме тому переважну більшість робочого часу менеджер проводить за листуванням за допомогою електронної пошти. А основною перевагою працівника стає швидкість реакції на лист електронної пошти, адже життєвий цикл проекту в бюро перекладів дуже короткий, він може складати лише декілька годин.

Представлені на ринку системи управління бізнесом у сфері лінгвістичних послуг орієнтовані саме на деталі управління файлами проекту, а не на комунікаційну складову. Як наслідок, ускладнюється процес роботи менеджера.

Типова ситуація в робочому процесі — отримання листа від замовника з проханням вислати йому готові файли та безпосереднє пересилання цих файлів — стає складним завданням для менеджера. Тому що кількість постійних замовників навіть у маленькому бюро — декілька десятків, і в кожного менеджера одночасно знаходиться в роботі кілька проектів. Для того щоб просто вислати готовий переклад замовнику, менеджеру потрібно відкрити систему управління проектами та якимось чином співвіднести лист від замовника із проектом у системі. Особливо складним це завдання стає тоді, коли темою листа є «потрібен переклад» (а виходячи з практики, це приблизно дві третини листів від замовників).

Відсутність можливості синхронізації бази проектів із базою листів електронної пошти в поштовій скриньці створює масу проблем у діяльності менеджера, призводить до помилок у роботі та унеможливлює оперативне реагування на потреби клієнта.

Деякі програмні рішення (зокрема, Plunet та XTRF) намагаються вирішити цю проблему, максимально виключаючи листування електронною поштою з процесу роботи над проектом. За задумом, менеджер проектів повинен лише надіслати посилання на проект замовнику та виконавцям. Замовник сам завантажує файли для перекладу в систему, а виконавці виконують переклад або редагування тексту за допомогою вбудованих чи сторонніх (але підключених до системи) інструментів. На практиці менеджер стикається з тим, що й замовники, й виконавці дуже неохоче ідуть на таку співпрацю: інтерфейс подібних систем зазвичай переобтяжений зайвими функціями та потребує значного часу для освоєння. Складний інтерфейс також викликає затримки в роботі й у самого менеджера.

Також треба враховувати високу вартість користування наявними системами управління проектами, що дуже актуально для країн, що розвиваються.

Інший тип проблем пов’язаний із типізацією даних при роботі з програмним забезпеченням у бюро перекладів. Наприклад, один з основних параметрів проекту — об’єм роботи. Історично склалося, що для вимірювання цього параметра в усіх країнах світу використовують два різні підходи: розраховують об’єм роботи в кількості слів вихідного тексту або ж у годинах роботи виконавців. Також існує параметр «flat fee» (мінімальний об’єм роботи), що використовується для невеликих проектів та означає обсяг роботи, нижче якого всі проекти мають бути оплачені у фіксованому розмірі. При цьому кожен замовник і кожне бюро перекладів можуть одночасно використовувати кілька систем обчислення об’єму роботи, навіть у рамках одного проекту (наприклад, часто трапляється ситуація, коли замовник висилає проект із певною кількістю слів і додатково оплачує погодинну роботу для нетипових завдань).

При цьому кількість слів зазвичай розраховується згідно з формулою, пов’язаною з кількістю сегментів вихідного тексту та ступенем збігу цих сегментів між собою та з пам’яттю перекладу. Єдиної формули не існує, і кожне бюро перекладів використовує власну. Тому один і той самий проект може мати різну кількість слів для бюро-замовника та для бюро-виконавця.

Існують також специфічні для окремих регіонів системи обчислення об’єму роботи (наприклад, у країнах, які раніше входили до складу СРСР, часто використовують підрахунок «облікових аркушів» — це певна кількість символів вихідного тексту, для кожного бюро своя, іноді з урахуванням пробілів, іноді ні. А в країнах Азії поширена система підрахунку ієрогліфів у готовому перекладі).

Така ситуація дуже ускладнює роботу менеджера проектів бюро перекладів. Якщо програмне забезпечення, з яким він працює, передбачає використання строгої типізації об’єму робіт, менеджер повинен перераховувати цей параметр для кожного нетипового проекту, конвертуючи облікові аркуші в кількість слів або кількість символів — у години роботи.

Також треба згадати про проблему кадрів. Наразі нам невідомий жоден заклад вищої освіти, що готував би спеціалістів у галузі менеджменту проектів для бюро перекладів. При цьому такий менеджер одночасно повинен розумітися на питаннях лінгвістики та перекладу, фінансів та управління персоналом. Компанії вимушені брати на роботу студентів (найчастіше лінгвістів або представників інших гуманітарних спеціальностей) або ж узагалі людей без вищої освіти та навчати їх самостійно. А оскільки програмне забезпечення, з яким працюють менеджери проектів бюро перекладів, наразі досить складне, період такого навчання часто становить рік або навіть більше.

Відсутність реалізації комунікативної складової в спеціалізованому програмному забезпеченні, відсутність єдиної системи типізації даних про проекти та некваліфікований персонал — ось три проблеми, з якими наразі стикається будь-яке бюро перекладів. Саме ці проблеми в першу чергу було враховано при проектуванні та реалізації описуваної системи.

## Постановка задачі

Після аналізу предметної області та визначення основних проблем наявних систем можна зробити висновок, що в сучасного невеликого або середнього бюро перекладів є потреба в створенні системи управління проектами Translation Mailer, яка б відповідала таким вимогам:

* зрозумілий і звичний для непідготовленого користувача інтерфейс;
* швидка інтеграція системи в наявні бізнес-процеси певного бюро перекладів і невелика вартість обслуговування;
* синхронізація з одним із провідних сервісів електронного листування;
* можливість роботи з комп’ютера або мобільного пристрою.

Урахувавши всі вище перераховані параметри, можна створити стабільну та добре оптимізовану систему з управління проектами. Translation Mailer повинна мати такі функції:

* створення проекту перекладу, внесення основних даних;
* розподіл проектів на «зовнішні» (проекти, отримані бюро перекладів від замовника) та «внутрішні» (проекти, оформлені бюро перекладів для виконавця);
* автоматичне створення листів замовникам і виконавцям за шаблонами;
* автоматичне створення основних документів (акти виконаних робіт тощо);
* надання користувачеві доступу до перегляду та редагування основних баз даних (замовники, виконавці, проекти, шаблони тощо).

Для базової основи програмного продукту була обрана екосистема Google, а саме її частина, призначена для корпоративних клієнтів, — G Suite. Вибір зумовлений такими факторами:

* найбільший у світі та найпопулярніший поштовий сервіс Gmail;
* невелика вартість корпоративного обслуговування в рамках набору хмарних сервісів G Suite;
* набір безкоштовних потужних програмних продуктів для збереження та редагування інформації (Google Forms, Google Sheets);
* зрозумілий і звичний для користувача інтерфейс програмних продуктів;
* високий рівень технічної підтримки сервісів і продуктів;
* розвинена спільнота;
* можливість підключення сторонніх сервісів через розвинену систему API;
* стабільність роботи програмних продуктів і сервісів;
* наявність окремої мови програмування, призначеної для створення застосунків для роботи з програмними продуктами Google, та документації, що описує цю мову.

В якості мови програмування обрано Google Apps Script — діалект мови Javascript, спеціально створений компанією Google для тісної інтеграції програмних продуктів екосистеми G Suite між собою.

В якості бази даних створена NoSQL-подібна система на базі сервісу Google Sheets. У порівнянні з традиційними в подібних програмних продуктах реляційними базами даних такий підхід забезпечить простіший доступ до записів бази даних із боку користувача та значно меншу вартість обслуговування. Також при такому підході буде виключена необхідність конвертації основних типів даних між різними стандартами.

Система має включати такі частини:

* застосунок для обробки й аналізу листа в Gmail, представлення інформації про поточні проекти, автоматичного створення листів за шаблонами та автоматичної генерації відповідних документів;
* модуль внесення та збереження інформації на базі Google Sheets;
* модуль автоматичного створення подій на базі сервісу Google Calendar, що підключається;
* модуль автоматичного зберігання вхідних файлів на базі сервісу Google Drive, що підключається.

Для нормального функціонування системи потрібен комп’ютер або мобільний пристрій, на якому встановлено браузер Google Chrome, Mozilla Firefox, Internet Explorer, Microsoft Edge або Apple Safari. Також для роботи системи потрібне стабільне з’єднання з мережею Інтернет. Для користувача необхідно мати доступ до корпоративного облікового запису G Suite.

# Формування вимог для програмної системи

Вимоги до програмного забезпечення — це сукупність тверджень щодо атрибутів, властивостей або якостей програмної системи, що підлягає реалізації. Вони створюються в процесі розробки вимог до програмного забезпечення, у результаті аналізу вимог.

## Функціональні вимоги

Функціональні вимоги пояснюють, що саме має бути зроблено, ідентифікують завдання або дії, які повинні бути виконані. Функціональні вимоги визначають дії, які система повинна бути здатною виконати, описують вхідні та вихідні дані системи, її поведінку в певних сценаріях.

Для формування функціональних вимог даного програмного забезпечення для кожного із застосунків було розроблено користувацький опис вимог до розроблюваної системи, сформульованих як пропозиції користувача.

Функціональні вимоги до застосунку для роботи з листами електронної пошти:

* як користувач я хочу, щоб застосунок мав можливість представлення основної інформації про поточні проекти бюро перекладів;
* як користувач я хочу, щоб застосунок надавав можливість швидкого доступу до треду електронної пошти, пов’язаному з певним поточним проектом;
* як користувач я хочу, щоб застосунок мав можливість розподілу й аналізу вхідних листів електронної пошти за визначеними критеріями;
* як користувач я хочу, щоб дані про всі вхідні листи від замовників із новими проектами вносилися до окремої таблиці;
* як користувач я хочу, щоб при обробці застосунком листа від замовника з новим проектом автоматично запускався інтерфейс внесення інформації про новий проект;
* як користувач я хочу, щоб застосунок мав можливість створення чернеток та шаблонів листів для виконавців на основі внесеної інформації про новий проект;
* як користувач я хочу, щоб при обробці застосунком листа від виконавця застосунок надавав інформацію про внутрішній проект та швидкий доступ до треду електронної пошти, пов’язаному з зовнішнім проектом;
* як користувач я хочу, щоб застосунок давав можливість вносити повну необхідну інформацію про новий проект (строки, об’єм роботи, тип роботи тощо);
* як користувач я хочу, щоб застосунок давав можливість обрати виконавців проекту, виходячи із внесених даних про тип роботи;
* як користувач я хочу, щоб застосунок надавав інформацію про зайнятих і незайнятих на момент початку роботи над проектом виконавців;
* як користувач я хочу, щоб інформація про кожен новий проект та вся допоміжна інформація (тип роботи, замовник, виконавці, терміни виконання тощо) автоматично вносилися до окремої таблиці;
* як користувач я хочу, щоб застосунок надавав авторизованим користувачам, які мають на це право, можливість внесення та редагування інформації про проекти;
* як користувач я хочу, щоб застосунок мав можливість автоматичного створення основних документів для представлення замовнику (акти виконаних робіт тощо);
* як користувач я хочу, щоб застосунок мав можливість автоматичного створення основних документів для надання виконавцям (звіти про виконані проекти тощо);
* як користувач я хочу мати можливість простого редагування відповідних шаблонів листів електронної пошти та документів без зміни коду застосунку.

Функціональні вимоги до модулів, що підключаються:

* як користувач я хочу, щоб при створенні нового проекту всі документи, додані до листа електронної пошти від замовника, автоматично та без втручання з боку користувача зберігалися в окремому каталозі на базі сервісу Google Drive;
* як користувач я хочу, щоб при створенні нового проекту автоматично та без втручання з боку користувача створювалася відповідна подія (із терміном завершення) на базі сервісу Google Calendar.

Таким чином, були представлені всі функціональні вимоги, які поставлені перед системою.

## Нефункціональні вимоги

Нефункціональні вимоги визначають властивості, які система повинна мати, або обмеження, яких вона повинна дотримуватися, що не відносяться до поведінки системи.

Описувана система підтримки діяльності бюро перекладів повинна відповідати таким вимогам:

* система повинна мати можливість швидко інтегруватися в наявні бізнес-процеси різних бюро перекладів;
* система повинна мати можливість зміни інтерфейсу основного застосунку відповідно до потреб певного бюро перекладів без втручання в програмний код;
* інтерфейс основного застосунку повинен бути максимально простим та інтуїтивно зрозумілим для непідготовленого користувача;
* усі дані про проекти, які можуть бути внесені до бази даних у автоматичному режимі, повинні вноситися без втручання користувача;
* усі форми для внесення інформації повинні бути створені з мінімізацією можливості внесення користувачем некоректних даних;
* інтерфейс основного застосунку системи повинен бути створений з урахуванням рекомендацій та правил стайлгайду компанії Google;
* інсталяція застосунків повинна проводитися згідно з правилами та за допомогою сервісу G Suite Marketplace;
* доступ до інсталяції та роботи із застосунками повинні мати лише авторизовані користувачі в межах корпоративного облікового запису компанії в сервісі G Suite;
* авторизація користувачів повинна проводитися згідно з правилами авторизації компанії Google;
* при виникненні помилок у системі кінцевий користувач не повинен отримувати детальну інформацію про помилки, що сталися.

Таким чином, були представлені всі нефункціональні вимоги, які поставлені перед системою.

# Архітектура та проектування програмного забезпечення

## UML проектування ПЗ

Проектування програмного забезпечення починається з побудови UML діаграм, що описують основних акторів у системі та їхні можливості, різноманітні процеси й активності в системі. UML — це мова для візуалізації, специфікування, конструювання та документування артефактів програмних систем [8].

Загальний процес роботи над проектом представлений за допомогою діаграми прецедентів (див. рис. 3.1):

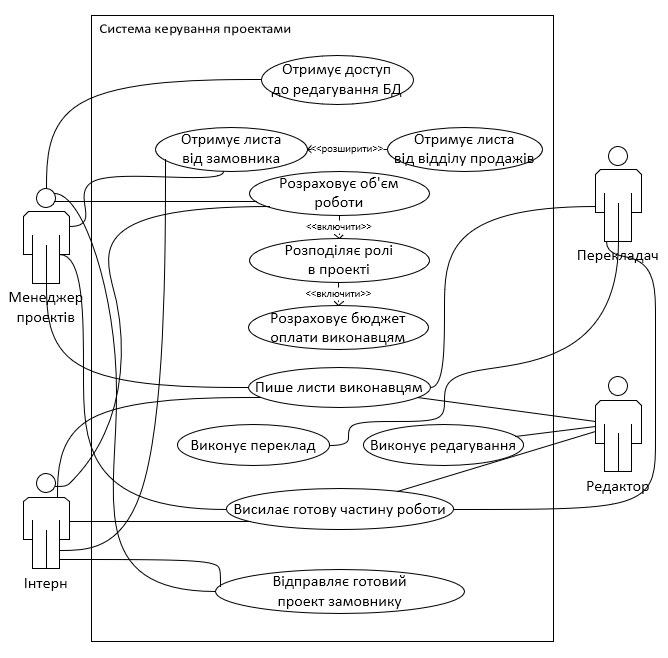


Рисунок 3.1 – Use Case діаграма роботи над проектом у бюро перекладів

Отримавши листа, виконавці розпочинають роботу над файлами проекту. А виконавши свою частину роботи, висилають готові файли менеджеру проектів. Менеджер, проаналізувавши, чи всі етапи проекту було належним чином виконано, висилає готові файли замовнику.

Саме для полегшення роботи менеджера проектів під час описаного процесу й була створена система Translation Mailer.

Однією з ключових особливостей системи Translation Mailer є мінімізація втручання в процес із боку користувача. Розглянемо, наприклад, принцип створення вікна (картки) основного застосунку в залежності від поведінки користувача в сервісі Gmail.

Спочатку з боку застосунку проходить перевірка, чи не відкрита основна сторінка сервісу Gmail зі списком листів електронної пошти? Якщо це так, застосунок отримує основну інформацію про поточні проекти з бази даних, і створює сторінку (картку) з відповідною інформацією. Або картку з інформацією про те, що активних проектів немає в наявності, якщо це так.

Якщо користувач відкрив лист електронної пошти, застосунок спочатку перевіряє наявність реєстрації унікального ідентифікатора треду в таблиці «зовнішніх» проектів. Якщо збіг знайдено, генерується картка з розширеною інформацією про відповідний проект.

Якщо збігу не було знайдено, унікальний ідентифікатор треду перевіряється згідно з таблицею «внутрішніх» проектів. Якщо збіг знайдено, генерується картка з розширеною інформацією про відповідний проект.

Якщо збігу треду не було знайдено в обох таблицях проектів, то, швидше за все, користувач відкрив лист із новим проектом від замовника (чи від представника відділу продажів). Тому застосунок генерує картку з формою реєстрації нового проекту.

Нарешті, якщо користувач відкрив вікно написання нового листа електронної пошти, застосунок представить картку зі списком шаблонів тексту для швидкої вставки. Весь процес генерації описаний у відповідній діаграмі активності (див. рис. 3.2)

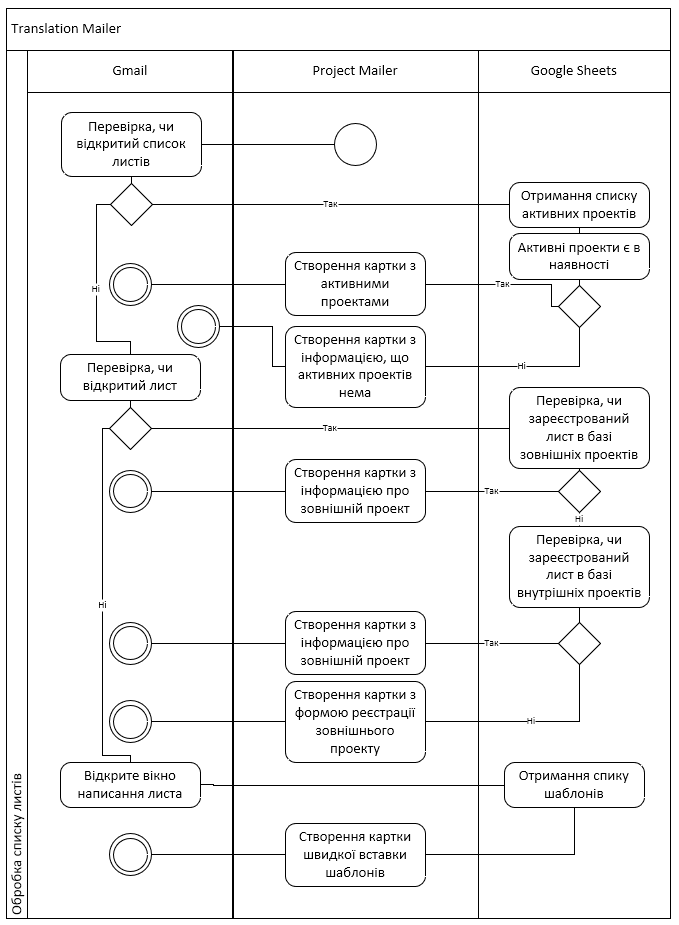
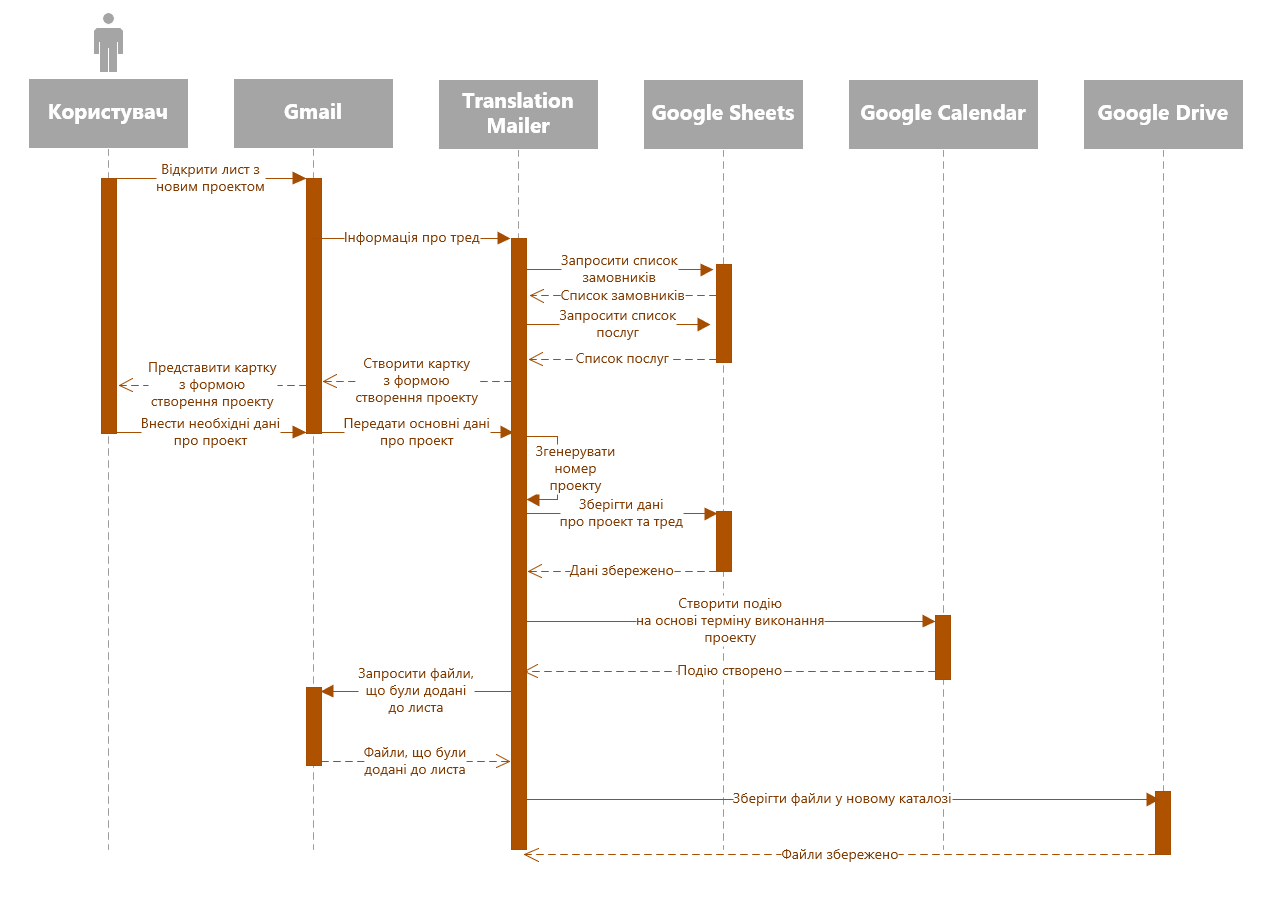


Рисунок 3.2 – Діаграма активності створення інтерфейсу основного застосунку в сервісі Gmail

Важливо зазначити, що для зміни картки з інформацією не потрібно жодних дій із боку користувача. Застосунок сам обирає, яку саме картку представити користувачеві в даний момент часу, виходячи з його поведінки при роботі із сервісом Gmail та аналізуючи інформацію про відкритий лист електронної пошти.

Подібний «принцип найменшого втручання» реалізовано й на інших ключових етапах взаємодії користувача з основним застосунком. Наприклад, процес реєстрації нового проекту представлено на відповідній діаграмі послідовності (див. рис. 3.3):

Рисунок 3.3 – Діаграма послідовності створення форми реєстрації проекту

Коли користувач відкриває лист електронної пошти із новим проектом, застосунок генерує картку з формою внесення даних. Користувач має внести лише ті дані, які неможливо отримати автоматично з листа електронної пошти чи з бази даних. При цьому застосунок намагається представити у формі дані за замовчуванням (наприклад, автоматично обираючи замовника на основі інформації про адресу електронної пошти відправника чи підставляючи в поле внесення бюджету проекту найбільшу суму у валюті, знайдену в тексті листа).

Усі дані про проект, які можна отримати в автоматичному режимі (наприклад, тема листа, унікальний ідентифікатор треду, дата реєстрації проекту, список доданих файлів) вносяться в базу даних без жодних дій із боку користувача (ці дані навіть не додаються в інтерфейс додатку, щоб не переобтяжувати його).

Усі дії, які можна виконати в автоматичному режимі (внесення даних у базу, генерація номеру проекту, збереження доданих до листа електронної пошти файлів у сервісі Google Drive, створення події з даними про проект та терміни виконання в сервісі Google Calendar) виконуються за допомогою лише однієї кнопки.

У подібний спосіб виконуються й інші основні етапи роботи користувача із застосунком.

Таким чином, можна стверджувати, що наведена схема роботи системи дозволяє суттєво спростити основні завдання менеджера проектів бюро перекладів при обробці вхідних листів та створенні нових проектів.

## Вибір системи зберігання даних

Традиційно для систем управління бізнесом бюро перекладів у якості систем зберігання даних використовуються різноманітні рішення на основі реляційних баз даних. В описуваній системі передбачається принципово інший підхід — створення NoSQL-подібної системи зберігання даних на базі сервісу Google Sheets.

На даний момент єдина система управління реляційними базами даних, яка підтримує зв’язок із застосунками, створеними в екосистемі Google за допомогою мови програмування Google Apps Script, — це система MySQL (за допомогою бібліотеки JDBC). Цілком логічно буде порівнювати обрану NoSQL-подібну систему зберігання даних саме з рішенням на базі MySQL.

У першу чергу, постає питання збереження точності інформації під час конвертації (якщо конвертація потрібна). Усі сервіси екосистеми Google, які задіяні в описуваній системі підтримки діяльності бюро перекладів, написані мовою програмування Javascript і мають відповідні набори примітивних типів даних. Мова програмування Google Apps Script є діалектом Javascript і оперує такими ж типами даних. Тобто, наприклад, при збереженні інформації із сервісу Gmail у таблиці Google Sheets за допомогою застосунку, написаному на Google Apps Script, не потрібно проводити конвертацію типів даних.

При використанні MySQL усі дані повинні бути конвертовані спочатку відповідно до стандарту мови програмування Java (саме цією мовою написана бібліотека JDBC, і тільки такі дані вона в змозі обробляти), а потім має бути проведена конвертація вже згідно зі стандартом MySQL. Якщо користувачеві потрібно буде представити наочно ці дані (а в екосистемі Google найпростіший спосіб представлення таблиць — саме сервіс Google Sheets), описаний процес конвертації повинен бути проведений у зворотному напрямку. При цьому на кожному з етапів конвертації може виникнути помилка, або ж буде втрачено частину інформації. Особливо це стосується числових даних, де точність має критичне значення.

Із тих же причин рішення на базі NoSQL-подібної системи зберігання даних працює значно швидше. Усі дані зберігаються на серверах компанії Google, що забезпечує найшвидший доступ до них із боку будь-якого сервісу екосистеми. Рішення на базі MySQL потребує стороннього сервера для реляційної бази даних і залучання в процес кожного запиту серверу JDBC.

Рішення на базі Google Sheets має значно нижчий «поріг входження» для користувача. Типова ситуація в роботі бюро перекладів — корегування даних уже запущених у роботу (або навіть виконаних) проектів. У разі використання Google Sheets користувач повинен буде просто відкрити відповідну таблицю та внести необхідні зміни (інтерфейс цього сервісу інтуїтивно зрозумілий будь-якому користувачеві, який мав справу з будь-яким пакетом офісних застосунків). Цю операцію користувач зможе провести навіть за допомогою мобільного пристрою, який підтримує роботу із сервісом Google Sheets, що іноді буває критичним. У разі використання рішення на основі реляційних баз даних потрібно буде створювати відповідний інтерфейс користувача або ж наймати в бюро перекладів окремого спеціаліста з навичками роботи із SQL-подібними базами даних.

Ураховуючи, що в системі Translate Mailer реалізована можливість зміни інтерфейсу за допомогою внесення змін у відповідні таблиці бази даних (без зміни програмного коду застосунку), використання NoSQL-подібної системи зберігання даних теж більш виправдано. Наведемо ще одну типову ситуацію в роботі невеликого бюро перекладів: від замовника надійшов проект з новим типом завдання. При цьому такий тип завдання не представлений у системі, але бюро перекладів готове його виконати. При використанні Translate Mailer та NoSQL-подібної системи зберігання даних менеджеру проектів бюро перекладів достатньо лише додати запис про новий тип завдання у відповідну таблицю Google Sheets — і після цього форма реєстрації проектів у основному застосунку Translate Mailer буде змінена. При використанні традиційних програмних рішень, що використовують реляційні бази даних, подібна операція або дуже ускладнена, або в більшості випадків просто неможлива.

Стабільність роботи теж має велике значення. Компанія Google гарантує корпоративним користувачам сервісу G Suite стабільний доступ до даних протягом 99,99 відсотків часу [9]. Стабільність системи на базі MySQL залежить від обраного серверу для зберігання бази даних, при цьому стабільність роботи бібліотеки JDBC не гарантована [10].

NoSQL-подібна система не потребує нормалізації. Тому вона значно простіше масштабується, а систему допомоги управління бюро перекладів можна швидше налаштувати під потреби конкретної компанії. Нормалізована реляційна база даних значно ускладнює цей процес.

Зрештою, важливим є питання вартості обслуговування системи. MySQL потребує оренди серверу, а іноді й залучення спеціаліста з відповідними навичками обслуговування реляційних баз даних [11]. Вартість роботи сервісу Google Sheets уже включена до абонентської плати корпоративних клієнтів G Suite.

Звісно, рішення на базі MySQL має й свої переваги (у першу чергу це можливість інтеграції даних зі сторонніми сервісами, а також зручність роботи з великими об’ємами даних). Але оскільки система передбачає роботу в бюро середнього або малого розміру, функціонал роботи з великими об’ємами записів буде незадіяний. Крім того, рішення на базі NoSQL-подібної системи не виключає інтеграцію з відповідною базою даних на базі MySQL у разі потреби.

## Проектування системи зберігання даних

Описувана система допомоги при управлінні бюро перекладів передбачає використання п’ятьох базових таблиць у NoSQL-подібній системі зберігання даних.

Оскільки, як вже було зазначено, один і той самий проект може мати різний об’єм роботи для бюро-замовника та для бюро-виконавця (у залежності від формули розрахунку об’єму роботи, що використовується в кожному бюро), різні строки виконання, різний список файлів тощо, при проектуванні таблиць бази даних сутність «проект» була умовно розділена на дві сутності — «зовнішній проект» (проект від замовника) та «внутрішній проект» (проект для виконавця). Кожен «внутрішній» проект пов’язаний із «зовнішнім» та є його логічним продовженням у життєвому циклі роботи над проектом у бюро перекладів. Кожному «зовнішньому» проекту можуть відповідати кілька «внутрішніх» (у разі виконання окремих етапів роботи різними виконавцями). Наприклад, одному «зовнішньому» проекту із типом завдання «переклад і редактура» (найрозповсюдженіший тип завдання) буде відповідати два внутрішніх проекти — один для перекладу, один для редактури (кожен зі своїм виконавцем). Кожному типу проектів відповідає своя таблиця.

Таблиця «зовнішніх» проектів включає номер проекту (генерується автоматично), унікальний ідентифікатор треду листа електронної пошти від замовника (використовується в якості первинного ключа), інформацію про замовника, про менеджера проектів, що зареєстрував цей проект, тип завдання, дату початку та кінця роботи, об’єм роботи, бюджет та статус проекту (завершений чи ні). При цьому поле, що відповідає за інформацію про об’єм роботи, має рядковий тип (як уже зазначалося, на практиці об’єм роботи зовнішнього проекту може вимірюватись різними не пов’язаними між собою величинами). Це виключає можливість операцій з розрахунком бюджету на основі об’єму, але на практиці бюджет «зовнішнього» проекту завжди визначає замовник, а бюро-виконавець може лише погодитися із цим бюджетом чи відмовитися від проекту.

Таблиця «внутрішніх» проектів містить інформацію про номер проекту (генерується на основі номеру «зовнішнього» проекту, типу завдання, дати початку роботи, унікального коду замовника та коду виконавця). Треба зазначити, що цей номер не використовується в якості ключа, а створений лише для теми листа електронної пошти з внутрішнім проектом. Це дає змогу менеджеру проектів бюро перекладів при отриманні листа від виконавця лише по одній темі, не відкриваючи листа, у більшості випадків зрозуміти, про який проект ідеться (звісно, це стосується лише невеликих бюро перекладів, зі штатом співробітників до ста осіб).

Також ця таблиця містить інформацію про ідентифікатор треду листа електронної пошти вихідного «зовнішнього» проекту (використовується в якості зовнішнього ключа), ідентифікатор треду внутрішнього проекту (первинний ключ), інформацію про виконавця, об’єм роботи (у цій таблиці це вже цілочисельний тип даних), дату початку та кінця роботи та бюджет.

Таблиця замовників включає інформацію про назву компанії, адресу, список співробітників, з якими підтримується контакт, типи завдань, розмір оплати для кожного з типів завдань.

Таблиця виконавців містить інформацію про ім’я та прізвище кожного виконавця, контактну адресу електронної пошти, список типів завдань, з якими працює виконавець, список робочих мовних пар і розмір ставки для оплати кожного з типів завдання.

Таблиця типів завдань містить інформацію про ці типи (для модифікації інтерфейсу форм внесення інформації про проект у основному застосунку) та код типу завдання (для генерації номера проекту).

Наведений список таблиць є базовим, без якого робота системи допомоги при управлінні бюро перекладів неможлива. Однак це не означає, що цей список таблиць є вичерпним. Використання ненормалізованої NoSQL-подібної системи зберігання даних дозволяє швидке залучення до загальної системи допоміжних таблиць (наприклад, таблиці інвойсів, валют, реквізитів), що дозволяє швидко масштабувати та налаштовувати систему у відповідності до потреб конкретної компанії.

## Приклади найцікавіших алгоритмів і методів

Програмний код основного застосунку системи Translate Mailer містить декілька цікавих методів.

Перший із них стосується роботи з таблицями NoSQL-подібної системи зберігання даних.

Будь-яка робота з таблицями Google Sheets реалізується в Google Apps Script за допомогою вбудованого метода getRange(). Цей метод приймає від одного до чотирьох чисельних параметрів і повертає фіксований діапазон клітинок таблиці. Наприклад, getRange(2, 3, 4, 5) поверне діапазон із чотирьох рядків і п’яти стовпців, починаючи з другого рядка та третього стовпця в таблиці.

Під час реалізації програмного коду основного застосунка виникла проблема: якщо використовувати метод getRange() у такий спосіб, то будь-яка зміна в структурі таблиць (як-то додавання стовпців, зміна їх положення та інше) призведе до некоректної роботи застосунка. При цьому можливість зміни структури таблиць критично важлива для налаштування системи Translate Mailer згідно з потребами кожного конкретного бюро перекладів, адже для кожного бюро така структура унікальна.

Було прийнято рішення реалізувати спосіб, за якого будь-яка операція з доступу до даних ву таблиці залежала б не від фіксованої адреси діапазону, а від імені таблиці й імен стовпців. Для цього, у першу чергу, було створено блок констант, які описують структури таблиць. Наприклад, ось так виглядає блок констант для таблиці внутрішніх проектів:

const IN\_PROJECTS = 'Internal Projects';

const IN\_PROJECTS\_HEADER\_ID = 'id';

const IN\_PROJECTS\_HEADER\_EXT\_PROJECT = 'External project';

const IN\_PROJECTS\_HEADER\_EXT\_PROJECT\_THREAD =

'External project thread';

const IN\_PROJECTS\_HEADER\_THREAD\_ID = 'Gmail thread';

const IN\_PROJECTS\_HEADER\_VENDOR = 'Vendor';

const IN\_PROJECTS\_HEADER\_TASK = 'Task';

const IN\_PROJECTS\_HEADER\_STARTDATE = 'Start date';

const IN\_PROJECTS\_HEADER\_DEADLINE = 'Deadline';

const IN\_PROJECTS\_HEADER\_SCOPE = 'Scope';

const IN\_PROJECTS\_HEADER\_AMOUNT = 'Amount';

При будь-якому зверненні до таблиці потрібно спочатку отримати її дані в форматі масиву масивів, а потім знаходити поточне положення необхідного стовпця за допомогою метода indexOf(). Ось так виглядає функція підготовки масиву для додавання даних до згаданої вище таблиці внутрішніх проектів:

let projectDataArray = [];

let inProjects = getSheetValues(IN\_PROJECTS);

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_EXT\_PROJECT)] = projectId;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_ID)] = inProjectId;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_THREAD\_ID)] = inProjectThreadId;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_EXT\_PROJECT\_THREAD)] = permlink;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_VENDOR)] = vendor;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_TASK)] = task;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_STARTDATE)] = startdate;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_DEADLINE)] = deadline;

Звісно, подібний підхід суттєво ускладнює процес розробки програмного коду застосунку. Але, з іншого боку, від відкриває можливість додавати стовпці в будь-яке місце ключових таблиць, що використовує застосунок, та змінювати їхнє положення в таблиці (без внесення змін в програмний код). Також є можливість змінювати при необхідності назву стовпця таблиці (змінив при цьому значення відповідної константи в програмному коді). Для зручності цієї операції всі константи винесені в окремий файл коду.

Інший цікавий метод стосується прискорення внесення даних про новий «зовнішній» проект користувачем. Форма реєстрації проекту містить спеціальне поле для внесення бюджету. Лист електронної пошти від замовника може містити цю інформацію. Для пошуку в тексті листа інформації про бюджет була створена спеціальна функція:

function getLargestAmount(message) {

let amount = 0;

let messageBody = message.getPlainBody();

let regex = /\$[\d,]+\.?\d{0,2}/g;

let match = regex.exec(messageBody);

while (match) {

amount = Math.max(amount,

parseFloat(match[0].substring(1).replace(/,/g,'')));

match = regex.exec(messageBody);

}

return amount ? '$' + amount.toFixed(2).toString() : null;

}

Вона приймає текст листа електронної пошти, шукає в ньому всі чисельні дані, які підходять під маску бюджету в обраній валюті, та повертає найбільше значення зі знайдених (або null, якщо подібних даних у тексті листа немає).

Звісно, текст листа електронної пошти може містити різні дані, які підходять під маску (наприклад, ставку за одне слово перекладу, або за одну годину роботи, оплату за певний етап проекту та інше). Але саме найбільше значення серед цих даних зазвичай і є бюджетом проекту.

Якщо подібні дані будуть знайдені, при створенні форми реєстрації проекту вони задаються як значення за замовчуванням у відповідному полі.

Ще один цікавий алгоритм пов’язаний із системою авторизованого доступу до деяких функцій основного застосунку системи.

Система передбачає два типи користувачів: «інтерн» і «менеджер проектів». Користувачу типу «інтерн» не доступний ряд функцій застосунку, а саме додавання інформації в базу даних про нового замовника чи нового виконавця, а також редагування інформації про наявний проект (такому користувачу просто не будуть надані відповідні кнопки в інтерфейсі застосунку).

Під час реалізації програмного коду виникла проблема. Система Translate Mailer доступна для інсталяції тільки авторизованим користувачам конкретного корпоративного облікового запису G Suite. Екосистема G Suite передбачає два типи авторизованих користувачів: звичайного користувача та адміністратора. Звісно, не можна було надавати права адміністратора всім користувачам системи Translate Mailer, яким потрібен доступ до повного функціоналу основного застосунку.

Звичайно, можна було б розподілити права користувачів у відповідності до прав доступу до конкретної таблиці бази даних. Але в такому випадку всі функції основного застосунку, що пов’язані з доступом до інформації з бази даних, не працювали б.

Проблема була вирішена таким чином: перед процедурою генерації картки інтерфейсу застосунок перевіряє права доступу конкретного користувача до певних клітинок таблиці бази даних (а саме можливість редагування заголовків стовпців). Якщо такої можливості немає, то користувач має права рівня «інтерн», і навпаки, якщо можливість є, — права рівня «менеджер проектів».

## Проектування UI системи

Оскільки робота описуваної системи передбачає тісну інтеграцію із сервісами екосистеми Google, цілком логічно підходити до питання створення дизайну, керуючись списком правил зі стилю інтерфейсу застосунків компанії Google. Згідно із цими правилами, кожен елемент користувацького інтерфейсу застосунку повинен органічно вписуватися в той сервіс екосистеми Google, з яким безпосередньо співпрацює застосунок [12].

Інтерфейс основного застосунку системи Translate Mailer побудовано з використанням технології Card-based interfaces. Застосунки G Suite, які побудовані за цією технологією, представляють інформацію й елементи керування на бічній панелі інтерфейсу хост-програми. Інтерфейс застосунку складається з основної ідентифікаційної панелі інструментів разом із однією або кількома картками.

Кожна картка — це певна «сторінка» інтерфейсу застосунку. Наприклад, головна картка основного застосунку системи Translate Mailer тісно інтегрована в три сервіси: Gmail, Google Drive та Google Calendar. Піктограма запуску основного застосунку системи Translate Mailer представлена в правій частині веб-сторінки сервісів (див. рис. 3.4):

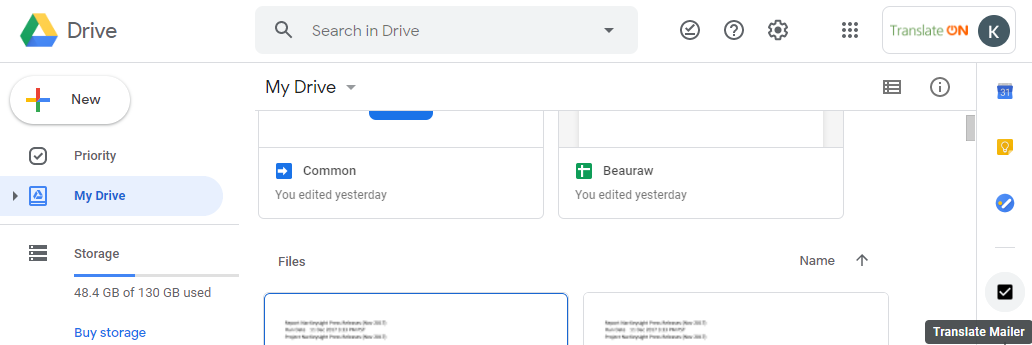


Рисунок 3.4 – Приклад розташування піктограми запуску застосунку в сервісі Google Drive

Після натискання піктограми в правій частині сторінки з’являється головна картка основного застосунку. Ця картка має однаковий вигляд у всіх трьох згаданих сервісах, незалежно від того, на пристрої якого типу був відкритий застосунок (комп’ютер чи мобільний пристрій), та яка версія сервісу представлена (повна, мобільна або ж застосунок одного із цих трьох сервісів у системі Android чи iOS).

На головній картці представлена стисла інформація про поточні «зовнішні» проекти, які зареєстровані в базі даних і мають статус «у роботі». Кожному користувачеві представлений список лише тих проектів, які були зареєстровані в системі саме цим користувачем.

Приклад розташування картки представлено на рисунку 3.5:

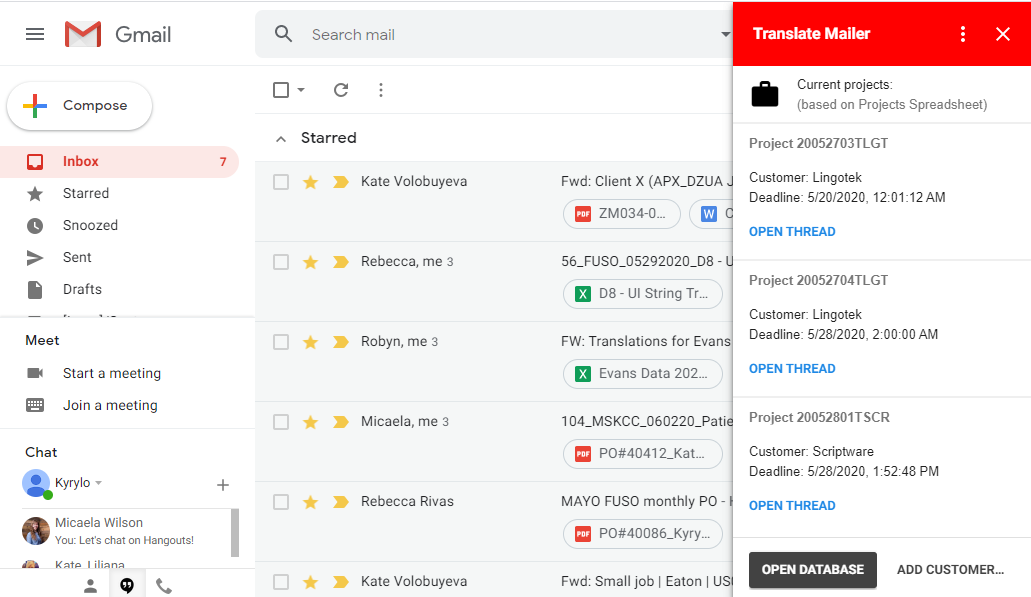


Рисунок 3.5 – Приклад інтеграції інтерфейсу застосунку в сервіс Gmail

У нижній частини головної картки основного застосунку представлено дві кнопки. Перша відкриває в новому вікні браузера документ Google Sheets із базою проектів, друга — форму Google Forms для додавання нового замовника або виконавця в базу даних. Якщо в користувача немає прав на подібні дії, цей блок кнопок просто не буде представлено в головній картці.

Під кожним блоком з інформацією про проект є кнопка з посиланням, яка при натисканні відкриває в окремому вікні браузера сторінку сервісу Gmail із відкритим тредом листів електронної пошти , який відноситься до цього проекту (див. рис. 3.6):

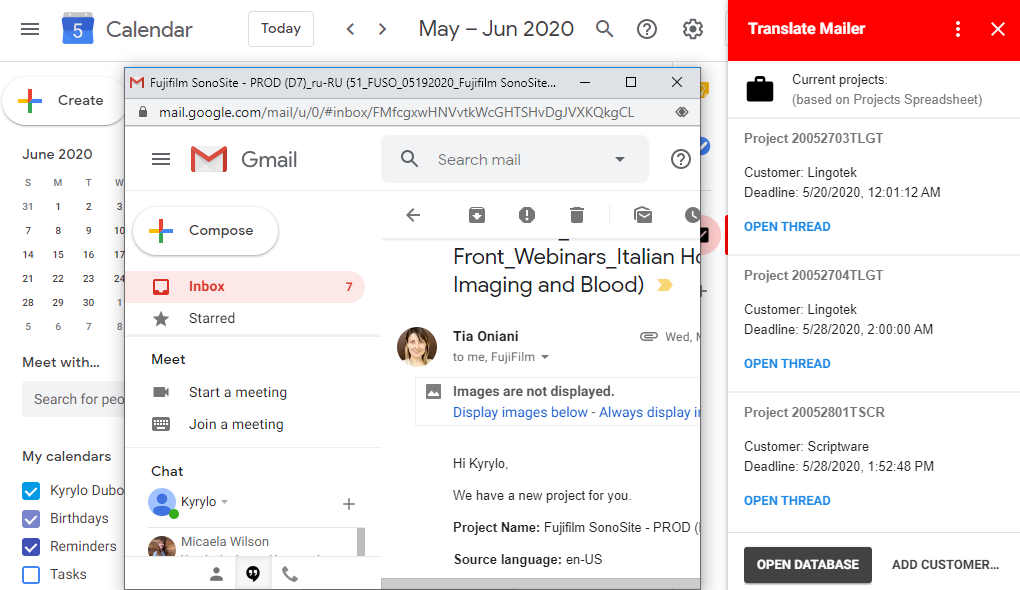


Рисунок 3.6 – Доступ до треду Gmail у сервісі Google Calendar

Розглянемо процес генерації інших карток інтерфейсу основного застосунку в сервісі Gmail.

Як уже було згадано, застосунок сам обирає, яку саме картку представити користувачеві в певний момент часу, аналізуючи його поведінку при роботі з електронною поштою. Наприклад, якщо користувач відкрив лист, тред якого вже був зареєстрований у базі даних у таблиці «зовнішніх» проектів, застосунок автоматично згенерує картку з розширеною інформацією про проект (див. рис. 3.7):

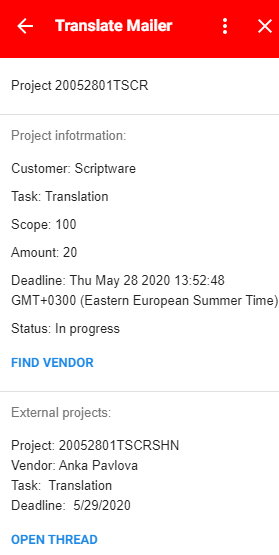


Рисунок 3.7 – Картка з розширеною інформацією про проект

Також у цій картці буде представлена стисла інформація про всі «внутрішні» проекти, які пов’язані із цим «зовнішнім» проектом (якщо такі проекти представлені в базі даних). Для зручності роботи на мобільних пристроях блок із цією інформацією має функцію згортання.

При натисканні кнопки пошуку виконавця застосунок згенерує відповідну форму (див. рис. 3.8):

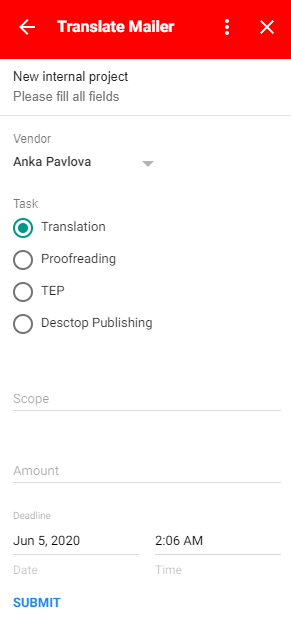


Рисунок 3.8– Форма пошуку виконавця

Усі поля представленої форми генеруються, виходячи з інформації, представленої у відповідній таблиці бази даних. Цей підхід у реалізації інтерфейсу форм дозволяє швидко налаштувати систему Translate Mailer згідно з потребами конкретного бюро перекладів (наприклад, якщо в цьому бюро інший список сервісів, що надаються, користувачеві потрібно лише змінити їхній перелік у відповідній таблиці Google Sheets, і форма зміниться).

Також у цій формі (та всіх інших формах внесення інформації в застосунку) реалізовано принцип мінімізації помилок при внесенні інформації. Усі поля форми, де інформація чутлива до типу даних, що вносяться, представлені у вигляді елементів, які унеможливлюють внесення некоректних записів. Поле вибору типу завдання — радіокнопка, поле вибору виконавця — розкривний список, поле терміну виконання проекту — елемент внесення дати та часу.

Аналогічно, якщо тред відкритого користувачем листа зареєстровано в списку «внутрішніх» проектів, буде згенерована подібна до представленої вище картка з повною інформацією про цей проект і стислою інформацією про пов’язаний «зовнішній» проект.

А якщо тред відкритого користувачем листа не зареєстровано в жодній таблиці проектів, це означає, що це лист із новим проектом від замовника. І застосунок автоматично представить форму внесення даних про новий проект (див. рис. 3.9):

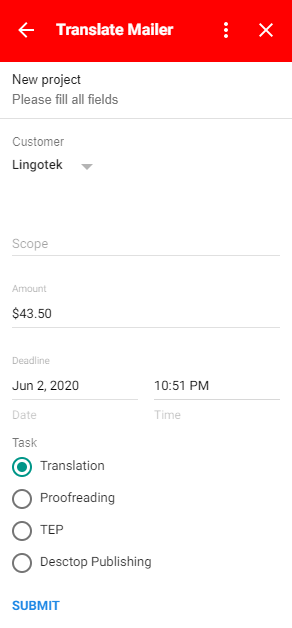


Рисунок 3.9 – Форма реєстрації нового проекту

При цьому застосунок сам підставить значення за замовчуванням для поля вибору замовника (проаналізувавши адресу електронної пошти відправника відкритого листа та порівнявши її з відповідною таблицею бази даних) та для поля внесення бюджету проекту (проаналізувавши текст листа). Це зроблено для полегшення роботи користувача та для зменшення кількості помилок при внесенні інформації про новий проект.

Якщо користувач запускає в сервісі Gmail меню створення листа електронної пошти, піктограма запуску основного застосунку системи Translate Mailer дублюється в нижній частині відповідного вікна (див. рис. 3.10):

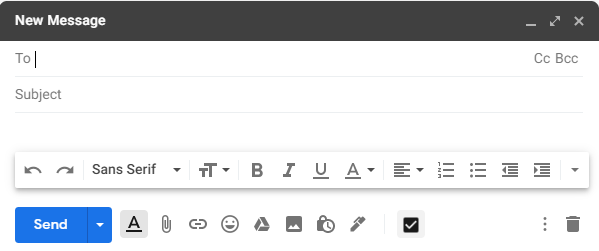


Рисунок 3.10 – Піктограма застосунку у вікні редагування листа електронної пошти

Дублювання піктограми проведено для того, щоб користувач мав змогу швидко отримати доступ до специфічної картки застосунку, яка містить шаблони для вставки в текст листа електронної пошти. Особливо це актуально при роботі з версією інтерфейсу Gmail для мобільних пристроїв. Із тієї ж причини це єдина картка застосунку, яка за замовчуванням запускається не автоматично, а лише після натискання піктограми.

На перший погляд, функціонал, представлений у цій картці, дублює відповідний сервіс вставлення шаблонів в сервісі Gmail. Але це не так: оскільки шаблони тексту, представлені в цій картці, базуються на записах у відповідній таблиці бази даних, ці шаблони будуть однакові для кожного користувача застосунку в рамках одного бюро перекладів (що дуже актуально при необхідності додавання єдиних для компанії інструкцій чи пояснень у листи електронної пошти з проектами). Інтерфейс такої картки представлений на рисунку 3.11:

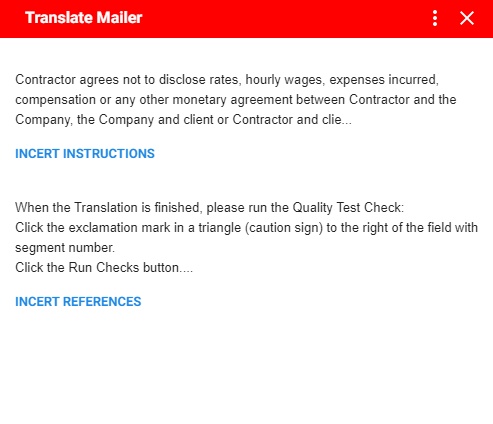


Рисунок 3.11 – Картка шаблонів тексту

Для зручності в роботі ця картка запускається в окремому вікні й автоматично закривається при вставці будь-якого шаблону зі списку.

Оскільки застосунки описуваної системи тісно інтегровані в дизайн сервісів екосистеми Google, можна стверджувати, що UX проектування було проведено з урахуванням користувацького досвіду користувача сервісів Gmail та Google Forms. Цей підхід до UX проектування дозволяє новому (але знайомому з продуктами Google) користувачу швидко опанувати керування застосунками описуваної системи [13].

UI проектування було здійснено з урахуванням усіх рекомендацій, наведених у списку правил зі стилю інтерфейсу застосунків компанії Google.

# Опис прийнятих програмних рішень

Система Translate Mailer — це ряд веб-серверних застосунків для роботи з різними сервісами екосистеми G Suite, об’єднаних у модульну структуру навколо основного застосунка, який працює відразу з кількома сервісами. Ряд модулів є обов’язковим для роботи системи, інші ж є опціональними: вони можуть бути включені в систему або ні, у залежності від потреб конкретного бюро перекладів.

## Основний застосунок

Основний застосунок системи Translate Mailer представлений як допоміжна панель у сервісах Gmail, Google Drive та Google Calendar. Оскільки технологія створення застосунків G Suite була представлена зовсім нещодавно, наразі це всі сервіси, в які можлива інтеграція таких застосунків.

Основна мета системи Translate Mailer — полегшити роботу менеджера бюро перекладів, у першу чергу надаючи йому актуальну інформацію про поточні проекти та можливість швидкої взаємодії з сервісом електронної пошти. Саме ці функції закладені в логіку роботи основного застосунку.

Розглянемо, наприклад, процес створення картки з поточними «зовнішніми» проектами. Спочатку застосунок отримує список усіх проектів із відповідної таблиці бази даних, потім на основі цього списку створює масив масивів із поточними проектами, і вже на основі цього масиву масивів будується інтерфейс картки:

for (let i = 0; i < nonCompletedProjects.length; i++) {

let cardSection = CardService.newCardSection();

let textButton = CardService.newTextButton()

.setText("Open thread")

.setOpenLink(CardService.newOpenLink()

.setUrl("https://mail.google.com/mail/u/0/#inbox/" +

nonCompletedProjects[i][gmailTreadIdColumnNumber])

.setOpenAs(CardService.OpenAs.OVERLAY)

.setOnClose(CardService.OnClose.RELOAD\_ADD\_ON));

let text = CardService.newTextParagraph()

.setText('Customer: ' +

nonCompletedProjects[i][customerColumnNumber] + "\n" +

'Deadline: ' +

nonCompletedProjects[i][deadlineColumnNumber].toLocaleString());

cardSection.setHeader('<b>Project ' +

nonCompletedProjects[i][idColumnNumber] + '</b>')

.addWidget(text)

.addWidget(textButton);

card.addSection(cardSection);

}

Для зручності в роботі користувача для кожного поточного проекту створюється своя секція з віджетом, який надає стислу текстову інформацію про проект, та кнопкою, яка при натисканні відкриває в новому вікні браузера Gmail тред, пов’язаний із проектом. Коли вікно буде закрито користувачем, картка автоматично оновлює інформацію, яку вона містить.

Логіка роботи основного застосунку не відрізняється в залежності від того, в інтерфейсі якого саме сервісу G Suite він був запущений.

## Модуль для роботи із сервісом Gmail

Ураховуючи список основних функціональних вимог до системи Translate Mailer, цілком логічним є те, що саме модуль для роботи із сервісом Gmail є найбільшим модулем з усіх. Логіка його роботи та побудова інтерфейсу карток залежить від поведінки користувача при користуванні сервісом Gmail. Нижче наведено приклад коду для вибору картки, яка буде представлена користувачеві, якщо той відкриває лист електронної пошти:

if (projectRow > 0) {

let projectInfoArray = projectsArray[projectRow];

let customer = projectInfoArray[getColumnNumber(projectsArray,

PROJECTS\_HEADER\_CUSTOMER)];

if (projectInfoArray[getColumnNumber(projectsArray,

PROJECTS\_HEADER\_STATUS)]) {

status = 'Completed';

}

let card = createProjectInfoCard (projectsArray,

projectInfoArray);

return [card.build()];

} else {

let date = getReceivedDate(message);

let prefills = [sender,

'',

getLargestAmount(message),

getReceivedDate(message)];

let card = createNewProjectCard (prefills);

return [card.build()];

}

Якщо тред був зареєстрований у базі проектів, буде представлена картка з інформацією про проект, якщо ні — картка з формою створення нового проекту.

Більшість функцій приховані від користувача та виконуються автоматично, без втручання з його боку. Наприклад, розглянемо процес обробки натискання кнопки під формою реєстрації «внутрішнього» проекту:

let body = createInProjectMailBody (vendor, task, startdate,

deadline, scope, amount);

GmailApp.createDraft(vendorEmail,

'Project ' + inProjectId,

body, {

htmlBody: body

attachments: blobsStatisticsPDF,

fileAttachments});

if (calendarApp) {

CalendarApp.createEvent(title, deadline, endTime);

}

let inProjectThreadId = GmailApp.getDrafts()[0].getMessageId();

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_EXT\_PROJECT)] = projectId;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_ID)] = inProjectId;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_THREAD\_ID)] = inProjectThreadId;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_EXT\_PROJECT\_THREAD)] = permlink;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_VENDOR)] = vendor;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_TASK)] = task;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_STARTDATE)] = startdate;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_DEADLINE)] = deadline;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_SCOPE)] = scope;

projectDataArray[inProjects[0]

.indexOf(IN\_PROJECTS\_HEADER\_AMOUNT)] = amount;

inProjectsSheet.appendRow(projectDataArray);

Послідовно виконуються такі дії:

* із таблиці бази даних отримується шаблон листа електронної пошти для виконавця;
* генерується номер проекту;
* створюється чернетка листа електронної пошти для обраного виконавця, яка містить усі дані про проект;
* створюється електронний документ у форматі PDF, який містить інформацію про бюджет проекту та деталі оплати, та додається до чернетки листа електронної пошти;
* отримуються файли з додатку до листа електронної пошти «зовнішнього» проекту та додаються до чернетки;
* створюється подія в сервісі Google Calendar (якщо відповідний модуль системи підключено);
* створюється масив даних і вноситься в базу даних.

Весь цей процес приховано від користувача, він лише натискає кнопку та отримує результат.

Якщо детально проаналізувати масив для внесення в базу даних, можна помітити, що він містить десять елементів. При цьому форма внесення даних про проект містить лише чотири поля, а інші дані для цього генеруються автоматично.

Коли користувач відкриває вікно написання листа електронної пошти, модуль автоматично генерує картку із шаблонами для вставки в текст. Нижче наведено код генерації цієї картки:

function onGmailCompose(e) {

let card = CardService.newCardBuilder()

let templatesArray = getSheetValues(TEMPLATES);

let section = CardService.newCardSection();

for (let i = 1; i < templatesArray.length; i++) {

let action = CardService.newAction()

.setFunctionName('incertTemplateInEmail')

.setParameters({'template' :

templatesArray[i][getColumnNumber(templatesArray,

TEMPLATES\_HEADER\_TEXT)]});

let button = CardService.newTextButton()

.setText('Incert ' +

templatesArray[i][getColumnNumber(templatesArray,

TEMPLATES\_HEADER\_NAME)])

.setOnClickAction(action);

let textWidget = CardService.newTextParagraph()

.setText('\n' + templatesArray[i][getColumnNumber(templatesArray,

TEMPLATES\_HEADER\_TEXT)]

.substr(0, 200) + '...');

section.addWidget(textWidget)

.addWidget(CardService.newButtonSet().addButton(button));

}

card.addSection(section);

return [card.build()];

}

Кількість секцій і віджетів у цій картці залежить від кількості шаблонів, які представлені у відповідній таблиці бази даних. Кнопка вставки має назву відповідного шаблону. Також кожна секція картки має текстовий блок, де представлено перші 200 символів тексту шаблону. Цей блок додано для зручності роботи користувача, адже буває важко згадати, який саме шаблон представлений, лише за його назвою.

Таким чином, модуль системи Translate Mailer, який відповідає за роботу із сервісом Gmail, допомагає менеджеру проектів бюро перекладів автоматизувати більшість типових щоденних завдань, що виникають при реєстрації проектів, внесенні даних і створенні відповідних листів електронної пошти та супровідних документів. Модуль є критичним для роботи системи Translate Mailer і не може бути відключеним.

## Модуль для роботи із сервісом Google Sheets

Елементи сервісу Google Sheets (Таблиці Google) поки що не можуть бути інтегрованими в систему карток застосунків G Suite безпосередньо. Але дані таблиць можуть бути використані іншими сервісами (зокрема, сервісом Gmail).

Створення цього модуля було зумовлено вибором типу системи зберігання даних. Модуль виконує ряд завдань, які притаманні серверній частині традиційних веб-серверних програмних рішень.

Будь-яка операція з даними в системі Translate Mailer починається з отримання всіх даних відповідної таблиці бази даних. Нижче наведено блок коду, який реалізує цю операцію:

function getSheetValues(sheetName) {

let sheet = SPREADSHEET.getSheetByName(sheetName);

let array = sheet.getDataRange().getValues();

return array;

}

Таким саме чином можна отримати дані одного стовпця з таблиці:

function getColumnValuesArray (sheetName, columnNname) {

let array = [];

let sheet = SPREADSHEET.getSheetByName(sheetName);

let values = sheet.getDataRange().getValues();

let columnNumber = values[0].indexOf(columnNname);

for (let i = 0; i < values.length; i++) {

array[i] = values[i][columnNumber];

}

return array;

}

Для реалізації можливості зміни структури таблиць бази даних без втручання в програмний код застосунків було створено ряд функцій, які дозволяють оцінювати структуру таблиць у реальному часі. Нижче наведено код функції отримання заголовків стовпців для подальшого аналізу:

function getHeaderFromSheet (sheetName) {

let sheet = SPREADSHEET.getSheetByName(sheetName);

let range = sheet.getRange(1,1,1,sheet.getLastColumn());

let values = range.getValues();

return values;

}

Також у цьому модулі реалізовано ряд специфічних функцій, які потрібні для автоматичної генерації даних без втручання користувача. Нижче наведено код функції генерації номеру «внутрішнього» проекту:

function getNewProjectNumber (customer, task, projects) {

let customers = getSheetValues(CUSTOMERS);

let tasks = getSheetValues(TASKS);

let customerCode = getVlookup (customers,

customers[0].indexOf(CUSTOMERS\_HEADER\_CUSTOMER),

customers[0].indexOf(CUSTOMERS\_HEADER\_CODE), customer);

let taskCode = getVlookup (tasks,

tasks[0].indexOf(TASKS\_HEADER\_TASKS),

tasks[0].indexOf(TASKS\_HEADER\_CODE), task);

let date = (new Date()).toISOString().slice(2,10).replace(/-/g,"")

for (let i = projects.length - 1; i > 0; i--) {

let idColumnNumber = projects[0].indexOf(PROJECTS\_HEADER\_ID);

if (projects[i][idColumnNumber].length > 8) {

let dateString = projects[i][idColumnNumber].substring(0,6);

if (dateString == date) {

return date + String("0" +

(parseInt(projects[i][idColumnNumber].substring(6,8)) +1))

.slice(-2) + taskCode + customerCode;

}

}

}

return date + '01'+ taskCode + customerCode;

}

Цей код використовується для створення теми листа з проектом і містить закодовану інформацію про дату реєстрації проекту, замовника, виконавця та тип завдання. Коли менеджер проектів бюро перекладів отримає лист у відповідь від виконавця, він зможе самостійно ідентифікувати проект лише за темою (це актуально в першу чергу для прискорення швидкості обробки вхідних листів при використанні мобільних пристроїв, зокрема смартфонів або смартгодинників, коли пристрій надає в короткому повідомленні лише тему листа та відомості про відправника).

Таким чином, модуль системи Translate Mailer, який відповідає за роботу із сервісом Google Sheets, обробляє всі необхідні запити до бази даних із боку інших модулів. Модуль є критичним для роботи системи Translate Mailer і не може бути відключеним.

## Модуль для роботи із сервісом Google Drive

Цей модуль є актуальним для бюро перекладів середнього розміру. У першу чергу він призначений для визначення прав доступу до файлів у спільних корпоративних каталогах сервісу Google Drive. Нижче наведено блок коду функції, що генерує специфічну картку при виборі користувачем файлу з такого каталогу:

function onDriveItemsSelected(e) {

let builder = CardService.newCardBuilder();

e['drive']['selectedItems'].forEach(

function(item){

let cardSection = CardService.newCardSection()

.setHeader(item['title']);

if (item['addonHasFileScopePermission']) {

cardSection.addWidget(

CardService.newTextParagraph().setText(

"This file takes up: " + getQuotaBytesUsed(item['id'])));

} else {

cardSection.addWidget(

CardService.newTextParagraph().setText(

"Needs permission to access this file's quota."));

let buttonAction = CardService.newAction()

.setFunctionName("onRequestFileScopeButtonClicked")

.setParameters({id: item.id});

let button = CardService.newTextButton()

.setText("Request permission")

.setOnClickAction(buttonAction);

cardSection.addWidget(button);

}

builder.addSection(cardSection);

});

return builder.build();

}

Ця функція буде корисна тоді, коли в користувача є тільки права перегляду каталогу (такі права встановлені за замовчуванням у сервісі Google Drive). За необхідності модуль згенерує повідомлення власнику файлу в загальному каталозі з проханням надати повний доступ.

Модуль є опціональним і може бути відключений.

## Модуль для роботи із сервісом Google Calendar

Цей модуль є актуальним для бюро перекладів, бізнес-модель яких передбачає роботу з великою кількістю проектів із коротким або дуже коротким життєвим циклом (від кількох годин до двох діб).

Модуль додає функції автоматичного створення подій у відповідному календарі в сервісі Google Calendar при запиті з боку інших модулів (у першу чергу з боку модуля для роботи із сервісом Gmail). Нижче наведено код функції створення події при реєстрації «внутрішнього» проекту:

function createEvent() {

if (response == ui.Button.YES) {

try {

for (var b = 0; b < vendorsMailList.length; b++) {

let row = vendorsMailList[b];

let sDate = new Date(row[18]);

function setTimeToDate(date, time) {

let t = new Date(time);

let hour = t.getHours();

let min = t.getMinutes();

let sec = t.getSeconds();

let dateMod = new Date(date.setHours(hour, min, sec, 0))

return dateMod;

}

let sTime = setTimeToDate(sDate, row[19]);

let event =

CalendarApp.getDefaultCalendar()

.createEvent(mailTheme, sTime, sTime, {description: body});

}

} catch (e) {}

}

Оскільки вбудовані методи системи розробки застосунків G Suite для роботи із сервісом Google Calendar ще тільки анонсовані, у цьому блоку коду використовується зовнішній API CalendarApp.

Модуль є опціональним і може бути відключений.

Таким чином, ми розглянули всі модулі, які представлено в системі Translate Mailer. Модульна структура системи дозволяє швидко налаштувати її для вирішення робочих завдань у рамках бізнес-логіки конкретного бюро перекладів. Також ця структура дозволить у майбутньому при розширенні списку вбудованих методів системи розробки застосунків G Suite включати в уже готові програмні продукти новий функціонал.

# Тестування розробленого програмного забезпечення

Процес тестування програмного забезпечення — обов’язковий етап при розробці будь-якого програмного продукту.

Для розробників корпоративних застосунків G Suite компанія Google створила окремий документ, де описано процес тестування [14]. Цей документ передбачає чіткий порядок і види тестування, які обов’язково треба задіяти до кожного застосунку та його функцій перед публікацією в сервісі G Suite Marketplace.

## Функціональне тестування навігації між картками

Корпоративні застосунки G Suite передбачають два види навігації між картками: явне (із додаванням до інтерфейсу карток відповідних елементів управління навігацією) та неявне (коли навігація базується на поведінці користувача). Згідно зі списком вимог, що було сформовано, застосунки системи Translate Mailer передбачають лише неявний вид навігації між картками.

За типом поведінки картки поділяються на неконтекстні (представлені на основних екранах сервісів Google, наприклад, у сервісі Gmail — на головному екрані зі списком листів електронної пошти) та контекстні (залежать від поведінки користувача, наприклад, картка, яка генерується після відкриття листа електронної пошти).

Для неявного виду навігації керівництво з тестування корпоративних застосунків G Suite передбачає обов’язкове послідовне тестування навігаційної поведінки кожної контекстної картки, представленої в застосунках, а також кожен перехід від контекстної до неконтекстної картки та навпаки.

Для функціонального тестування навігації між картками було створено ряд тест-кейсів, які наведено в таблиці 5.1 у спрощеному вигляді:

Таблиця 5.1 – Тест-кейси для функціонального тестування навігації між картками

|  |  |  |  |
| --- | --- | --- | --- |
| № | Описання кроків тесту | Очікуваний результат | Пройдено / провалено |
| 1 | Запуск основного застосунку в сервісі Gmail. | З’являється основна картка застосунку. | Пройдено. |
| 2 | Вибір листа електронної пошти, зареєстрованого в таблиці «зовнішніх» проектів. | З’являється картка з інформацією про «зовнішній» проект. | Пройдено. |
| 3 | Вибір листа електронної пошти, зареєстрованого в таблиці «внутрішніх» проектів. | З’являється картка з інформацією про «внутрішній» проект. | Пройдено. |
| 4 | Вибір листа електронної пошти, не зареєстрованого в таблицях проектів. | З’являється картка з формою реєстрації нового проекту. | Пройдено. |
| 5 | Перехід у меню створення листа електронної пошти, натискання піктограми застосунку. | З’являється картка з шаблонами для вставки в текст листа. | Пройдено. |
| 6 | Перехід на основну сторінку сервісу Gmail. | З’являється основна картка застосунку. | Пройдено. |
| 7 | Запуск основного застосунку в сервісі Google Calendar. | З’являється основна картка застосунку. | Пройдено. |
| 8 | Вибір події, пов’язаної з проектом | З’являється картка з інформацією про проект. | Пройдено. |
| 9 | Перехід на основну сторінку сервісу Google Calendar. | З’являється основна картка застосунку. | Пройдено. |
| 10 | Запуск основного застосунку в сервісі Google Drive. | З’являється основна картка застосунку. | Пройдено. |
| 11 | Вибір елементу зі списку файлів. | З’являється картка з інформацією про права доступу. | Пройдено. |

Таким чином була протестована навігаційна поведінка кожної з карток, представлених у системі.

## Функціональне тестування контекстних карток

Наступним кроком керівництво з тестування корпоративних застосунків G Suite передбачає функціональне тестування контекстних карток. Для проведення такого тестування було створено ряд тест-кейсів, які наведено в таблиці 5.2 у спрощеному вигляді:

Таблиця 5.2 – Тест-кейси для функціонального тестування контекстних карток

|  |  |  |  |
| --- | --- | --- | --- |
| № | Описання кроків тесту | Очікуваний результат | Пройдено / провалено |
| 1 | Заповнення форми реєстрації нового проекту у відповідній картці модуля роботи з сервісом Gmail. | У відповідній таблиці бази даних з’являється відповідний запис; перехід на основну картку застосунку. | Пройдено. |
| 2 | Заповнення форми пошуку виконавця у відповідній картці модуля роботи з сервісом Gmail. | У відповідній таблиці бази даних з’являється відповідний запис; створюється електронний документ із детальною інформацією про проект; створюється чернетка листа електронної пошти обраному виконавцю; створюється подія з проектом у сервісі Google Calendar (якщо цей модуль підключено) | Пройдено. |
| 3 | Натискання кнопки вставки шаблону в текст листа електронної пошти у відповідній картці модуля роботи з сервісом Gmail. | У текст листа електронної пошти вставляється відповідний шаблон. | Пройдено. |

Таким чином, було проведено функціональне тестування кожної контекстної картки, представленої в системі.

## Нефункціональне тестування продуктивності

Керівництво з тестування корпоративних застосунків G Suite не передбачає тестування продуктивності застосунку в цілому. Але допускає тестування продуктивності кожної окремої функції програмного коду.

На рисунку 5.1 представлено приклад результатів нефункціонального послідовного тестування функцій програмного коду:

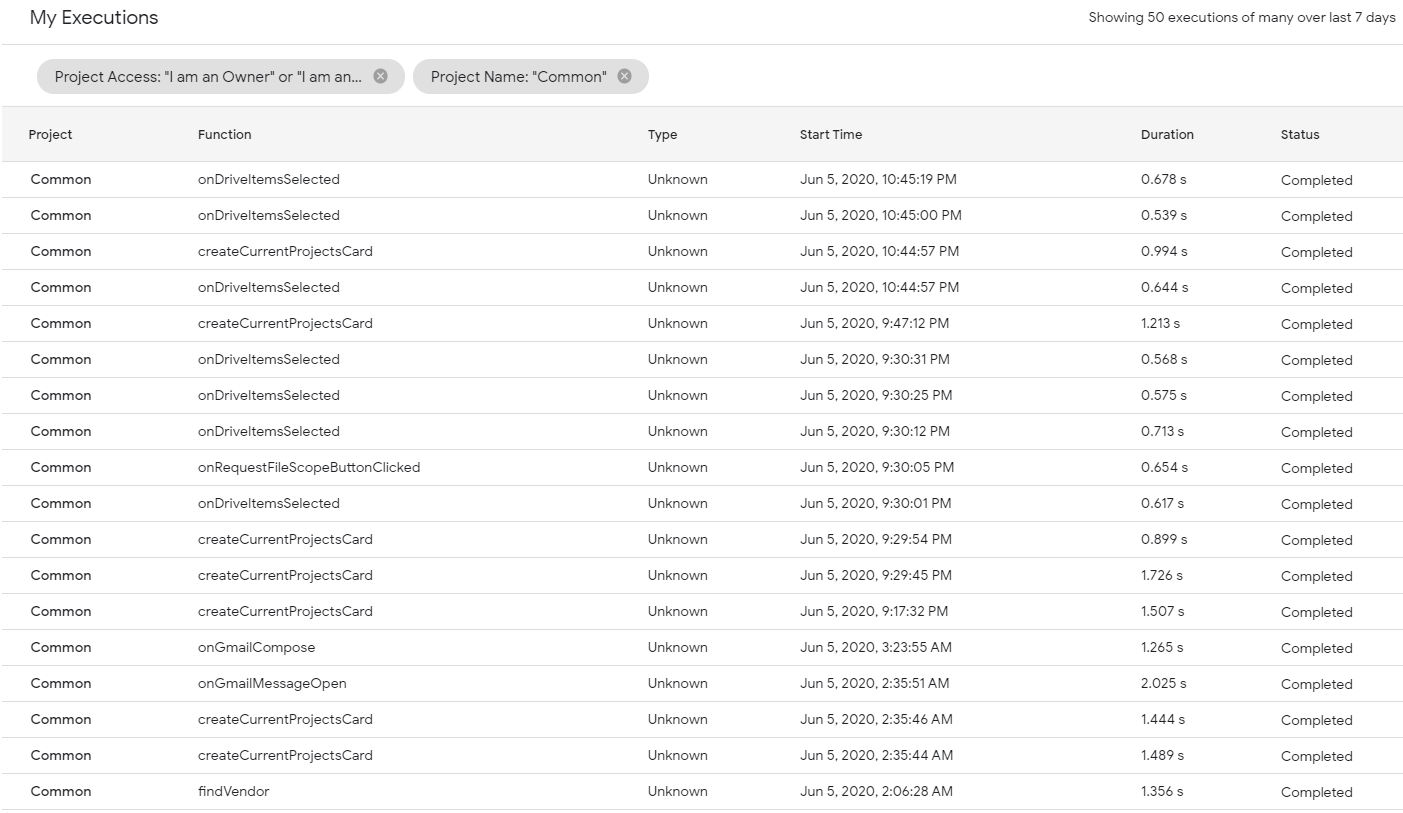


Рисунок 5.1 – Приклад результатів тестування продуктивності функцій

Максимально допустимим часом виконання функції вважається 30 секунд. Найкращім результатом вважається ситуація, коли всі функції виконуються швидше за 3 секунди.

За результатами тесту жодна із функцій програмного коду не виконувалася довше за 2 секунди 25 мілісекунд.

## Тестування безпеки

Останнім обов’язковим кроком є тестування безпеки, яке у випадку розробки корпоративних застосунків G Suite включає тестування авторизованого доступу до інсталяції та використання застосунків.

Із метою тестування безпеки було створено ряд тест-кейсів, які наведено в таблиці 5.3 у спрощеному вигляді:

Таблиця 5.3 – Тест-кейси для функціонального тестування контекстних карток

|  |  |  |  |
| --- | --- | --- | --- |
| № | Описання кроків тесту | Очікуваний результат | Пройдено / провалено |
| 1 | Авторизація користувача в системі авторизації G Suite; перехід на сторінку корпоративних застосунків G Suite Marketplace; інсталяція основного застосунку системи. | Основний застосунок системи інстальовано; в інтерфейсі відповідних сервісів Google з’явилася піктограма запуску застосунку. | Пройдено. |
| 2 | Спроба неавторизованого доступу до сторінки корпоративних застосунків G Suite Marketplace. | З’являється повідомлення «у доступі відмовлено». | Пройдено. |

Таким чином, були проведені всі необхідні етапи тестування, які описані в керівництві з тестування корпоративних застосунків G Suite.

# Впровадження програмного забезпечення

Оскільки не існує єдиних правил ведення бізнесу в галузі надання лінгвістичних послуг, кожне бюро перекладів має свою унікальну структуру бізнес-процесів. Як наслідок, для ефективної роботи кожного бюро потрібне своє програмне забезпечення, яке б повністю відповідало поставленим вимогам.

Система Translate Mailer від самого початку була спроектована з урахуванням таких реалій цієї сфери бізнесу, вона має ряд особливостей, що дозволяють швидко налаштувати систему у відповідності до потреб конкретної компанії.

Наразі вже створено два програмних рішення, в основу яких покладена система Translate Mailer.

## Lingotek VendorMailer

Компанію Lingotek було засновано у 2006 році в штаті Юта, США. Наразі компанія є одним із провідних розробників програмного забезпечення для перекладачів та бюро перекладів в країні. Протягом останніх п’яти років компанія входить до престижного рейтингу «Top 100 Companies in the Digital Content Industry» [15].

Також у компанії є відділ з надання лінгвістичних послуг. Він налічує понад два десятки співробітників (менеджери проектів та інтерни) та понад тисячу позаштатних перекладачів і редакторів.

Саме для полегшення щоденних завдань співробітників цього відділу була створена система Lingotek VendorMailer. Основні частини системи Translate Mailer було модифіковано згідно з потребами компанії та включено в систему Lingotek VendorMailer в якості модулів допомоги при роботі з електронною поштою та реєстрації проектів. На рисунку 6.1 представлено знімок екрана сторінки інсталяції основного застосунку системи в сервісі G Suite Marketplace:

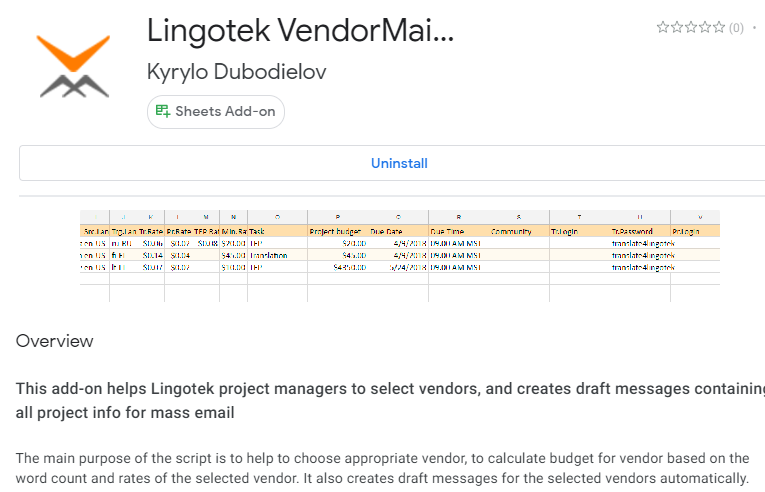


Рисунок 6.1 – Сторінка інсталяції основного застосунку системи Lingotek VendorMailer

Загальна статистика використання системи та статистика помилок у роботі наведені на рисунку 6.2:

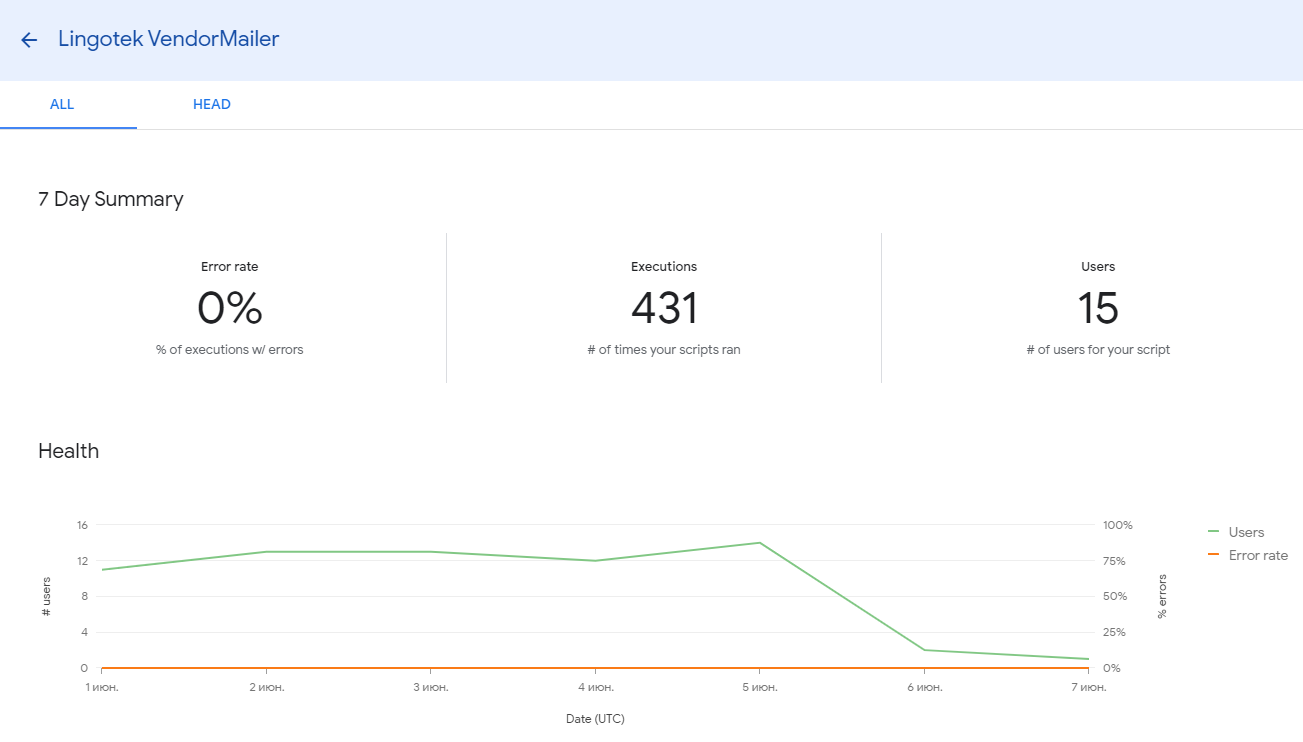


Рисунок 6.2 – Статистика використання системи Lingotek VendorMailer

Система отримала схвальні відгуки з боку користувачів і керівництва компанії.

Наразі триває процес створення технічного завдання для додання додаткових функцій.

## TranslateOn Mailer

Компанія TranslateOn — невелике бюро перекладів із Запоріжжя, Україна. Штат компанії налічує кілька менеджерів проектів і близько п’ятдесяти позаштатних перекладачів і редакторів.

Для забезпечення діяльності компанії на базі системи Translate Mailer була створена система TranslateOnMailer. Знімок екрану роботи програми представлено на рисунку 6.3:

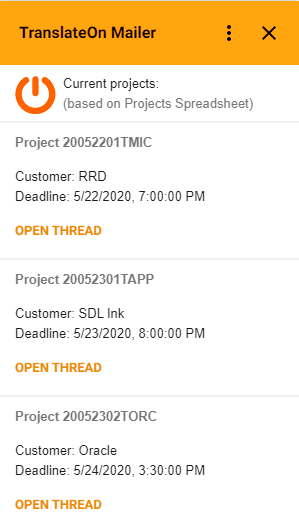


Рисунок 6.3 – Інтерфейс картки основного застосунку TranslateOn Mailer

Під час впровадження в систему було внесено зміни, що дозволили використання в її роботі вже наявної в компанії бази даних проектів. Це дозволило використання як наявних у компанії TranslateON програмних рішень, так і системи TranslateOn Mailer одночасно, без зміни структури бази даних.

Наразі триває процес навчання персоналу основних функцій системи, але вона вже отримала позитивні відгуки з боку менеджерів проектів цього бюро перекладів. У планах керівництва компанії повний перехід у роботі до використання системи TranslateOn Mailer в якості основної системи управління проектами.

Таким чином, систему Translate Mailer можна оцінювати як перспективне програмне рішення, продукти на базі якого можуть бути використані в бюро перекладів малого та середнього розміру в якості як основної, так і допоміжної системи в управлінні проектами.

# Висновки

У результаті роботи була створена веб-система підтримки діяльності бюро перекладів.

Була проаналізована предметна область надання лінгвістичних послуг. Були виявлені основні недоліки наявного програмного забезпечення для роботи менеджерів проектів бюро перекладів. На основі проаналізованих критеріїв і вимог було обґрунтовано та спроектовано нове програмне рішення.

Також було проведене UML проектування основних процесів системи, після чого розроблена архітектура системи.

Для зберігання даних системи було прийнято рішення використовувати No-SQL-подібну систему зберігання даних на базі сервісу Google Sheets.

Програмна частина системи була створена за допомогою мови програмування Google Apps Script.

Система включає модулі для роботи із сервісами Gmail, Google Sheets, Google Calendar та Google Drive.

На основі описуваної системи було створено системи управління проектами та листами електронної пошти для двох компаній із США й України, які працюють у сфері надання лінгвістичних послуг.

# Перелік джерел посилань

1. M. S. Zampaulo, 10 Simple Ways to Boost Your, Alexandria, VA, USA: American Translators Association, 2019.

2. A. Brantner, Some Fundamentals of Project Management, VA, USA: American Translators Association, 2016.

3. Products for Translation Teams // сторінка онлайн-магазину SDL. URL: https://www.sdltrados.com/store/ (дата звернення 12.05.2020)

4. The Leader in TMS Technology // корпоративна сторінка компанії Lingotek. URL: https://www.lingotek.com/translation-management-system/ (дата звернення 06.05.2020)

5. Автоматизируйте выплаты // корпоративна сторінка компанії Smartcat. URL: https://ru.smartcat.ai/payments/ (дата звернення 06.05.2020)

6. Your complete business and translation management system// корпоративна сторінка компанії Plunet GmbH. URL: https://www.plunet.com/en/translation-management-software/ (дата звернення 06.05.2020)

7. Pick your plan // сторінка онлайн-магазину XTRF. URL: https://www.xtrf.eu/buy/ (дата звернення 12.05.2020)

8. Фаулер М. UML. Основи, 3-є видання. Символ-плюс, 2015. – 237с.

9. Трансформація підходів до роботи в компаніях // корпоративна сторінка сервісу G Soite. URL: https://gsuite.google.com/intl/uk/enterprise/ (дата звернення 06.05.2020)

10. Herong Y. JDBC Tutorials - Herong's Tutorial Examples. Lulu Press, 2020. – 320с.

11. Почему не SQL? // Блог компании lsFusion. URL: https://habr.com/ru/company/lsfusion/blog/463095/ (дата звернення 13.05.2020)

12. UI style guide for editor add-ons // сторінка документації розробника. URL: https://developers.google.com/gsuite/add-ons/guides/editor-style (дата звернення 22.04.2020)

13. The Google UX Hiring Process// сторінка документації розробника. URL: http://googledesignmethod.libsyn.com/website/the-google-ux-hiring-process (дата звернення 25.04.2020)

14. Testing G Suite add-ons // сторінка документації розробника. URL: https://developers.google.com/gsuite/add-ons/how-tos/testing-gsuite-addons (дата звернення 02.06.2020)

15. The Top 100 Companies in the Digital Content Industry // сторінка видання Econtent. URL: http://www.econtentmag.com/Articles/Editorial/Feature/The-Top-100-Companies-in-the-Digital-Content-Industry-The-2016-2017-EContent-100-114156.htm (дата звернення 07.06.2020)

# Додаток А Слайди презентації

# Додаток Б Приклад програмного коду

Фрагмент коду модулю для роботи з сервісом Gmail:

function getCurrentMessage(e) {

let accessToken = e.messageMetadata.accessToken;

let messageId = e.messageMetadata.messageId;

GmailApp.setCurrentMessageAccessToken(accessToken);

return GmailApp.getMessageById(messageId);

}

function openGmailThread(threadId) {

return CardService.newActionResponseBuilder()

.setOpenLink(CardService.newOpenLink()

.setUrl('https://mail.google.com/mail/u/0/#inbox/' +

threadId))

.build();

}

function onGmailCompose(e) {

let card = CardService.newCardBuilder()

let templatesArray = getSheetValues(TEMPLATES);

let section = CardService.newCardSection();

for (let i = 1; i < templatesArray.length; i++) {

let action = CardService.newAction()

.setFunctionName('incertTemplateInEmail')

.setParameters({'template' :

templatesArray[i][getColumnNumber(templatesArray,

TEMPLATES\_HEADER\_TEXT)]});

let button = CardService.newTextButton()

.setText('Incert ' +

templatesArray[i][getColumnNumber(templatesArray,

TEMPLATES\_HEADER\_NAME)])

.setOnClickAction(action);

let textWidget = CardService.newTextParagraph()

.setText('\n' + templatesArray[i][getColumnNumber(templatesArray,

TEMPLATES\_HEADER\_TEXT)]

.substr(0, 200) + '...');

section.addWidget(textWidget)

.addWidget(CardService.newButtonSet().addButton(button));

}

card.addSection(section);

return [card.build()];

}

function incertTemplateInEmail (e) {

let text = e.parameters.template;

let response = CardService.newUpdateDraftActionResponseBuilder()

.setUpdateDraftBodyAction(CardService.newUpdateDraftBodyAction()

.addUpdateContent(text,

CardService.ContentType.MUTABLE\_HTML)

.setUpdateType(CardService

.UpdateDraftBodyType.IN\_PLACE\_INSERT))

.build();

return response;

}

function onGmailMessageOpen(e) {

let message = getCurrentMessage(e);

let subject = message.getSubject();

let thread = message.getThread();

let sender = thread.getMessages()[0]

.getFrom().replace(/^.+@([^>]+)>$/, "$1");

let firstMessage = thread.getMessages()[0];

let permlink = thread.getId();

let projectsArray = getSheetValues(PROJECTS);

let threadIdColumn = getColumnNumber(projectsArray,

PROJECTS\_HEADER\_THREAD\_ID);

let projectRow = getProjectRow(projectsArray, threadIdColumn,

permlink);

if (projectRow > 0) {

let projectInfoArray = projectsArray[projectRow];

let customer = projectInfoArray[getColumnNumber(projectsArray,

PROJECTS\_HEADER\_CUSTOMER)];

if (projectInfoArray[getColumnNumber(projectsArray,

PROJECTS\_HEADER\_STATUS)]) {

status = 'Completed';

}

let card = createProjectInfoCard (projectsArray,

projectInfoArray);

return [card.build()];

} else {

let date = getReceivedDate(message);

let prefills = [sender,

'',

getLargestAmount(message),

getReceivedDate(message)];

let card = createNewProjectCard (prefills);

return [card.build()];

}

}

function createProjectInfoCard (projectsArray, projectInfoArray) {

let card = CardService.newCardBuilder()

.setHeader(CardService.newCardHeader()

.setTitle('Project ' +

projectInfoArray[getColumnNumber(projectsArray,

PROJECTS\_HEADER\_ID)]));

let projectInfoSection = createProjectInfoSection (projectsArray,

projectInfoArray);

projectInfoSection.addWidget

card.addSection(projectInfoSection);

let internalProjects = getSheetValues(IN\_PROJECTS);

let externalProjectsArray = getExternalProjectsArray (

internalProjects, projectInfoArray[getColumnNumber(projectsArray,

PROJECTS\_HEADER\_THREAD\_ID)]);

if (externalProjectsArray.length > 0) {

let externalProjectsSection =

createExternalProjectsSection(internalProjects,

externalProjectsArray);

card.addSection(externalProjectsSection);

}

return card;

}

function createProjectInfoSection (projectsArray, projectInfoArray) {

let section = CardService.newCardSection()

.setHeader('Project infotrmation:');

for (let i = 1; i < PROJECT\_INFO\_WIDGETS.length; i++) {

let widget = CardService.newTextParagraph()

.setText(PROJECT\_INFO\_WIDGETS[i] + ': ' +

projectInfoArray[getColumnNumber(projectsArray,

PROJECT\_INFO\_WIDGETS[i])]);

section.addWidget(widget);

}

let projectId = projectInfoArray[getColumnNumber(projectsArray,

PROJECTS\_HEADER\_ID)];

let findVendorAction = CardService.newAction()

.setFunctionName('findVendor');

let findVendorButton = CardService.newTextButton()

.setText('Find vendor')

.setOnClickAction(findVendorAction);

section.addWidget(CardService.newButtonSet()

.addButton(findVendorButton));

return section;

}

function createExternalProjectsSection(internalProjects,

externalProjectsArray) {

let section = CardService.newCardSection()

.setHeader('External projects:')

.setCollapsible(true)

.setNumUncollapsibleWidgets(0);

for (let i = 0; i < externalProjectsArray.length; i++) {

let widget = CardService.newTextParagraph()

.setText('Project: ' +

externalProjectsArray[i][getColumnNumber(internalProjects,

IN\_PROJECTS\_HEADER\_ID)] + '\n' + 'Vendor: ' +

externalProjectsArray[i][getColumnNumber(internalProjects,

IN\_PROJECTS\_HEADER\_VENDOR)] + '\n' + 'Task: ' +

externalProjectsArray[i][getColumnNumber(internalProjects,

IN\_PROJECTS\_HEADER\_TASK)] + '\n' + 'Deadline: ' +

externalProjectsArray[i][getColumnNumber(internalProjects,

IN\_PROJECTS\_HEADER\_DEADLINE)].toLocaleDateString());

let textButton = CardService.newTextButton()

.setText("Open thread")

.setOpenLink(CardService.newOpenLink()

.setUrl("https://mail.google.com/mail/u/0/#inbox/" +

externalProjectsArray[i][getColumnNumber(internalProjects,

IN\_PROJECTS\_HEADER\_THREAD\_ID)])

.setOpenAs(CardService.OpenAs.OVERLAY)

.setOnClose(CardService.OnClose.NOTHING));

section.addWidget(widget)

.addWidget(textButton)

.addWidget(CardService.newTextParagraph().setText('\n'));

}

return section;

}

function findVendor (e) {

let card = CardService.newCardBuilder();

card.setHeader(CardService.newCardHeader()

.setTitle('New internal project')

.setSubtitle('Please fill all fields'));

let section = createInProjectFormSection(CardService

.newCardSection());

card.addSection(section)

return card.build();

}

function createInProjectFormSection (section) {

let vendorsArray = getColumnValuesArray(VENDORS,

VENDOR\_HEADER\_NAME);

if (vendorsArray) {

let vendorsWidget = CardService.newSelectionInput()

.setType(CardService.SelectionInputType.DROPDOWN)

.setFieldName(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[0])

.setTitle(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[0]);

for (let i = 1; i < vendorsArray.length; i++) {

vendorsWidget.addItem(vendorsArray[i], vendorsArray[i], false)

}

section.addWidget(vendorsWidget);

}

let tasksArray = getColumnValuesArray(TASKS, TASKS\_HEADER\_TASKS);

if (tasksArray) {

let taskTypeWidjet = CardService.newSelectionInput()

.setType(CardService.SelectionInputType.RADIO\_BUTTON)

.setFieldName(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[1])

.setTitle(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[1]);

for (let i = 1; i < tasksArray.length; i++) {

taskTypeWidjet.addItem(tasksArray[i], tasksArray[i], false)

}

section.addWidget(taskTypeWidjet);

}

section.addWidget(CardService.newTextInput()

.setFieldName(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[2])

.setTitle(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[2]))

.addWidget(CardService.newTextInput()

.setFieldName(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[3])

.setTitle(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[3]))

let datePickerWidget = CardService.newDateTimePicker()

.setFieldName(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[4])

.setTitle(NEW\_IN\_PROJECT\_CARD\_FIELDNAMES[4]);

datePickerWidget.setValueInMsSinceEpoch(new Date().getTime());

section.addWidget(datePickerWidget);

let submitNewInProjectForm = CardService.newAction()

.setFunctionName('submitNewInProjectForm');

let submitButton = CardService.newTextButton()

.setText('Submit')

.setOnClickAction(submitNewInProjectForm);

section.addWidget(CardService.newButtonSet()

.addButton(submitButton));

return section;

}

# Додаток В Електронні матеріали