ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

«УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

TimeSoft (застосунок)

Пояснювальна записка до проекту

за спеціальністю 122 – комп’ютерні науки

(освітня програма – Комп’ютерні науки)

|  |  |
| --- | --- |
| Виконали: | студенти 2 курсу  групи КН-19  Симоновський Євгеній  Белявцев Юрій  Вергун Дмитро  Колесніков Євгеній  Ластівка Тимофій  Тирон Владислав  Печкур Олександр |
|  |  |

м. Київ – 2021 рікВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

«УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

|  |  |
| --- | --- |
|  | ЗАТВЕРДЖУЮ:  завідувач кафедри комп’ютерних наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.С. Тимчук  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 р. |

ЗАВДАННЯ

на проект студентів 2 курсу

Симоновський Євгеній, Белявцев Юрій, Вергун Дмитро, Колесніков Євгеній, Ластівка Тимофій, Тирон Владислав, Печкур Олександр

|  |  |
| --- | --- |
| Тема проекту | Створення кросплатформенного застосунку з графічним інтерфейсом для аналізу витраченого часу за 5 тижнів. |
| Коротка постановка завдання | Графічне представлення та комфортне керування структурованою інформацією про витрачений час та, як наслідок - прогнозування витраченого часу у майбутньому. |
| Посилання на джерела інформації (не більше п’яти найменувань, які рекомендує керівник) | 1. https://www.riverbankcomputing.com/static/Docs/PyQt5/ 2. https://www.python.org/doc/ |
| Вимоги до роботи | Проект має передбачити проектування та розробку кросплатформного програмного забезпечення із GUI та застосуванням процедурної та об’єктно-орієнтованої парадигм програмування, методів та алгоритмів обчислень, структур даних і механізмів управління. |
| Термін представлення роботи до захисту | до 28 травня 2021 р. |

Дата видачі завдання 17 січня 2021 р.

|  |  |
| --- | --- |
| Студенти | Симоновський Євгеній  Белявцев Юрій  Вергун Дмитро  Колесніков Євгеній  Ластівка Тимофій  Тирон Владислав  Печкур Олександр |

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва етапів роботи | Термін виконання етапів роботи | Примітка про виконання |
| **Підготовчий етап** | | | |
| 1 | Вибір напряму проекту | 26.04.2021 | Виконано |
| 2 | Підбір та вивчення літературних та інших джерел з напряму проекту | 26.04.2021 | Виконано |
| 3 | Формування теми проекту | 26.04.2021 | Виконано |
| 4 | Розробка концепції проекту | 26.04.2021 | Виконано |
| 5 | Затвердження теми та завдання на проект | 26.04.2021 | Виконано |
| **Основний етап** | | | |
| 6 | Аналіз предметної області і постановка задачі | 26.04.2021 | Виконано |
| 7 | Проектування застосунку | 28.04.2021 | Виконано |
| 8 | Реалізація застосунку | 28.05.2021 | Виконано |
| 9 | Підготовка та подання керівнику пояснювальної записки до проекту | 28.05.2021 | Виконано |
| 10 | Доопрацювання застосунку та пояснювальної записки з урахуванням зауважень керівника | 28.05.2021 |  |
| 11 | Представлення керівнику доопрацьованого варіанту проекту | 28.05.2021 |  |
| **Завершальний етап** | | | |
| 12 | Підготовка до демонстрації роботи застосунку | 28.05.2021 |  |
| 13 | Підготовка доповіді та презентації | 28.05.2021 |  |
| 14 | Захист проекту | 16.06.2021 |  |

|  |  |
| --- | --- |
| Студенти | Симоновський Євгеній  Белявцев Юрій  Вергун Дмитро  Колесніков Євгеній  Ластівка Тимофій  Тирон Владислав  Печкур Олександр |

Симоновський Є.В., Белявцев Ю.В., Вергун Д.О., Колесніков Є.Д., Ластівка Т.Т., Тирон В.В., Печкур О.А. Створення кросплатформенного застосунку з графічним інтерфейсом для аналізу витраченого часу за 5 тижнів.

Пояснювальна записка до проекту за спеціальністю 122 – Комп’ютерні науки. – ВНЗ «Університет економіки та права «КРОК», Навчально-науковий інститут інформаційних та комунікаційних технологій, кафедра комп’ютерних наук, Київ, 2021.

У зв'язку з тим, що у сучасному світі складно самостійно контролювати витрати часу на ті чи інші завдання, що значно знижує показники особистої ефективності, виникла необхідність створити застосунок, який стане надійним помічником. Отже, результатом цього проекту став застосунок, який допомагає визначити прогалини у тайм-менеджменті, спрогнозувати витрати часу на ті чи інші типи людської діяльності протягом обраного періоду часу та спланувати максимально оптимальні часові витрати у майбутньому.

Ключові слова: застосунок, час, кросплатформений, користувач, телеграм-бот.

Табл. Х. Рис. Х. Бібліограф.: Х найм. (не менше 20 найменувань)

Symonovsky I.V., Beliavtsev Y.V., Vergun D.O., Kolesnykov Y.D., Lastivka T.T., Tyron V.V., Pechkur O.A. Creation of a cross-platform GUI application for a spent time analysis in 5 weeks.

Explanatory note to the project by specialty 122 – Computer science. – «KROK» University, Educational and Scientific Institute of information and communication technologies, Department of Computer Science, Kyiv, 2021.

Due to the fact that in the modern world it is hard to control the time spent on certain tasks on your own, which significantly reduces the indicators of personal efficiency, it became necessary to create an application that will become a reliable assistant. Thus, the result of this project is an application that helps to identify gaps in time management, to predict the time spent on certain types of human activities during a chosen period of time and to plan the most optimal time costs in the future.

Keywords: application, time, cross-platform, user, telegram-bot.

Tabl. X. Fig. X. Bibliography: X Items.

ЗМІСТ

[ВСТУП](#_c4vadyv1si5h) 6

[РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ](#_sd3am34oc58f) 8

1.1 [Проблеми, які вирішуються за допомогою тайм-менеджменту, його визначення, види та функції.](#_m9fseux8iur) 8

[Аналіз існуючих інформаційних технологій тайм-менеджменту.](#_9gnukbwkxpvh) 13

[Висновки по розділу](#_xljj3a6xtfcd) 16

[РОЗДІЛ 2. ПРОЕКТУВАННЯ ЗАСТОСУНКУ TIMESOFT](#_51b4q71btyys) 17

[Аналіз варіантів використання](#_akjnips74eov) 17

[Моделювання процесів і даних](#_8hqzwdvn2m0f) 20

[Діаграми діяльності.](#_33zkx25eyeg0) 20

[Опис ідентифікованих даних.](#_u9v3okyy5smc) 21

[Формат зберігання даних.](#_5tc1r1xoxkh) 23

[Проектування архітектури](#_ul2kci6tcgl2) 25

[Проектування графічного інтерфейсу користувача](#_hsxpt62t0y8q) 30

[Проектування структури](#_rlrzp1ckw29j) 35

[Висновки по розділу](#_rv316w76ulom) 36

[РОЗДІЛ 3. РЕАЛІЗАЦІЯ ЗАСТОСУНКУ TIMESOFT](#_7gbxoos3jurn) 37

[Особливості реалізації](#_2b3uz6jtbdfq) 37

[Конструювання](#_59x7335u1f6s) 38

[Реалізація графічного інтерфейсу користувача](#_ssuap641sxgm) 40

[Документування](#_n8v6srm3kt4f) 50

[Адміністрування](#_9m4vdglgrb41) 50

[Тестування](#_k3eeqcz1fmeh) 50

[Висновки по розділу](#_11xdhdei5aj0) 50

[ВИСНОВКИ](#_m5aqw6yv7zwv) 51

[СПИСОК ПОСИЛАНЬ](#_v8zax0ia9d5w) 52

[ДОДАТОК А](#_dxyghc9h9dho) 54

[ЕКРАНИ ЗАСТОСУНКУ TIMESOFT](#_k4ighsx0cru9) 55

[ДОДАТОК Б](#_tpydx8spdodz) 65

[ФРАГМЕНТИ ЛІСТИНГУ](#_cy621od7p4mo) 65

# ВСТУП

**Актуальність теми.**

Нагальна вимога сучасного світу – налагодження ефективного особистого тайм-менеджменту для оптимального виконання щоденних оперативних та майбутніх стратегічних завдань. Розвиток сучасних ігрових технологій, різноманітність інтернет-контенту, соціальні мережі створюють відчуття сильної та стійкої цікавості, але водночас ховають у собі небезпеку у вигляді розвитку фізіологічних та психологічних розладів, погіршення якості життя у цілому. Тому вибудовування особистого розпорядку дня та розподіл часу – надзвичайно важлива та актуальна тема для кожної людини, незалежно від статі, віку та роду занять.

Дослідженню тайм-менеджменту багато уваги приділяють різні сучасні публіцисти та науковці з багатьох країн світу. Найбільшого поширення отримали дослідження та роботи таких фахівців: Й. Кноблауха, Б. Санто, Л. Зайверта, К. Бішофа, Ст. Кові, П. Дойля, Дж. Коулі. Їх дослідження присвячені певним системам, які описують механізми ефективної роботи, алгоритми самоорганізації та спеціалізовані стратегії її досягнення.

Серед українських науковців можна виділити О. Колесова та А. Вацьківську, які висвітлюють методи організації та контролю за часовим ресурсом, а також С. Кулакову, яка досліджує специфіку впровадження європейської практики тайм-менеджменту в сучасних вітчизняних умовах.

Дрібні справи, накопичені у значній кількості, здатні красти час, який необхідно приділити виконанню дійсно важливих завдань, і навіть час на задоволення базових потреб, наприклад таких, як сон та прийом їжі. Для того, щоб оптимально розробити режим праці та відпочинку, дуже важливо помітити та визнати, що не весь час протягом дня (тижня, місяця) використовується раціонально.

Головний акцент у застосунку TimeSoft – візуалізація використання часу протягом певного періоду, що допомагає виявити прогалини у розпорядку та розподілі часу між різними видами діяльності.

Крім очевидного тактичного ефекту від раціонального розподілу часу, користувач застосунку отримує корисні звички такі, як регулярний сон, розвиток самодисципліни, і навіть покращити сімейні та особистісні стосунки, і як наслідок – підвищити якість життя в цілому.

**Мета і задачі проекту.**

Розробка системи фіксування витрат часу, контролю за ним протягом певного заданого періоду та візуалізації часових витрат за різними типи діяльності.

Для досягнення цієї мети необхідно розробити зручний та простий у користуванні застосунок, який підтримується найбільш поширеними операційними системами.

**Практична цінність проекту.**

У результаті реалізації проекту створено застосунок, який здатен фіксувати витрати часу користувачем у залежності від виду його діяльності та може допомогти у системному підвищенні особистої ефективності.

**Структура та обсяг пояснювальної записки.**

Структура та обсяг пояснювальної записки. Пояснювальна записка до проєкту складається із вступу, трьох розділів, висновків, списку посилань (Х найменувань) та додатків. Пояснювальна записка містить Х рисунків, Х таблиць. Загальний обсяг пояснювальної записки складає Х сторінку, основний зміст викладено на Х сторінках.

У вступі розглянуто проблеми розвитку інформаційних технологій в індустрії тайм-менеджменту, сформульовані мета і задачі проєкту, практичне значення отриманих результатів.

У першому розділі виконано огляд і аналіз сучасного стану інформаційних технологій в індустрії тайм-менеджменту на світовому та українському ринках. Виділено проблеми, які вирішує сучасний тайм-менеджмент, а також розглянуто його види та функції. Сформульовано постановку задачі даного проєкту.

У другому розділі розроблено модель управління поведінкою неігрового персонажа на базі методів штучного інтелекту, а саме нейронних мереж. Розроблено UML-проєкт системи управління поведінкою неігрового персонажа. У проєкті виділені сховище даних, логіка доступу до даних, логіка програми і логіка інтерфейсу.

У третьому розділі описана система створення і підтримки інтелектуальних неігрових персонажів відеоігор. Описано програмне забезпечення, до складу яких входять: пакет створення неігрових персонажів та пакет управління неігровими персонажами. Система має клієнт-серверну архітектуру. Програмна реалізація системи представлена у вигляді набору модулів, які має прозору структуру, містять уніфіковані структури даних, є розширюваними і повними.

У додатках приведено екрани розробленої системи і лістинги, які представлені фрагментарно.

# РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

## **Проблеми, які вирішуються за допомогою тайм-менеджменту, його визначення, види та функції.**

Стрімкий розвиток сучасного світу, тенденції глобалізації та цифрової трансформації економіки і суспільства ставлять перед людиною набагато більше завдань, ніж раніше. При цьому, вимоги до володіння такими якостями, як стресостійкість, здатність до багатозадачності значно зростають.

Проблема часу надзвичайно важлива і багатоаспектна, оскільки жодна сфера життєдіяльності людей не існує за межами часового чинника, поза розподілом в часі, планування і структурування актуальних цілей і намірів, мотивів і вчинків, завдань і операцій. Саме тому, на сучасному етапі, ефективне управління часом як ресурсом є головним завданням тайм-менеджменту [1, с. 190].

Час – це вичерпний, обмежений ресурс, тому надзвичайно важливо вміти використовувати його не тільки задля ефективності виконання робочих завдань, але і для здорового, якісного життя в цілому. Для того, щоб вдало розподіляти час, необхідно опанувати науку тайм-менеджменту.

Розглянемо найбільш поширені визначення тайм-менеджменту:

* Тайм-менеджмент – це технологія, що дозволяє невідновлюваний час життя використовувати відповідно до своїх особистих і бізнес-цілей і цінностей [2, с. 5].
* Тайм-менеджмент – це технологія грамотного розподілу часових ресурсів і збільшення ефективності їх використання, що представляє собою певний набір способів, орієнтованих на зменшення тимчасових витрат на виконання будь-яких справ без втрати якості [3, с. 49].
* Тайм-менеджмент - це управління собою і інтенсивне формування свого способу життя. Таким є визначення на розсуд Лотара Й.Зайверта, [4, с. 25].
* Тайм-менеджмент у найбільш загальному значенні Г.А. Архангельский представляє, як «технологію організації часу і підвищення ефективності його використання» [5, с. 15].
* Саме поняття «тайм-менеджмент» походить від англійського «time management» і означає технологію організації людиною свого часу і підвищення ефективності його використання. Таким чином, тайм-менеджмент розуміється як дія або сукупність дій, направлених на тренування усвідомленого контролю над кількістю часу, який витрачається на певні види діяльності і за допомогою якого можна істотно збільшувати свою продуктивність і результативність [6, с. 24].

Ключовою метою тайм-менеджменту вважається розстановка пріоритетів між завданнями, які поставлені перед людиною, і виконання в першу чергу більш вагомих з них. Зазвичай, скласти перелік справ не складно, але розставити пріоритети, тим більше з першого разу, – важче. Втім, вже давним-давно «винайдені» наступні дві методики, які базуються на відомому правилі Парето: 20% всіх справ дають 80% всіх результатів, і навпаки. Тобто, слід орієнтувати основні сили на ті справи, які знаходяться в рамках цих 20%, так як саме вони відносяться до розряду найбільш значущих [7, с. 26].

Першою методикою вважається «ABC-аналіз», згідно з якою: А-задачі – найбільш значущі, при тому, що на їх реалізацію буде потрібно не більше 10% часу. Вагомість А-задач складно переоцінити, тому що саме їх внесок в досягнення цілей оцінюється як 70%! В-завдання – це завдання середньої важливості, на них припадає приблизно 20% необхідного на виконання часу і 20% - по значущості, а ось С-завдання вимагають більше 70% часу, в той час як користь від них в сукупності не більше 10% .

Інша методика – це поєднання характеристик «важливість і терміновість». Переглянувши перелік справ, слід вивчити їх порівняльну важливість і терміновість. Згодом всі справи слід розподілити таким способом: А – важливі і термінові; В – важливі і нетермінові; С – неважливі і термінові; D – неважливі і нетермінові. Подібна послідовність мимоволі змусить задуматися. Спочатку звичайно розставляти пріоритети і розподіляти справи досить важко, проте пізніше це буде зробити набагато простіше, а витрачені зусилля монетизуються.

Автор книги «Тайм-менеджмент на комп'ютері», Б. Новак пропонує наступне визначення тайм-менеджменту: «це процес, спрямований на усвідомлене і відповідальне використання свого часу» [8, с. 11]. Багато людей скаржаться на те, що їм на добу, не вистачає годин, вони багато чого не встигають. Якраз в таких випадках необхідний тайм-менеджмент.

За Б. Новаком, основними завданнями тайм-менеджменту є:

* підвищення ефективності своєї діяльності;
* контроль справ (що і коли);
* аналіз щоденного розпорядку своїх справ та вжиття заходів щодо його оптимізації;
* вивчення та вдосконалення техніки використання свого часу;
* уміння позбавлятися від неефективних в плані використання часу видів діяльності;
* раціональне планування часу, завдяки чому найважливіші справи будуть виконуватися в найбільш доцільні для цього години і необхідні терміни;
* грамотна організація не тільки робочого, але і особистого часу, що дозволяє регулярно виділяти необхідний період для відпочинку.

Провести аналіз, куди йде час можливо шляхом ведення протягом двох тижнів щоденника, в нього має бути занесено все, що робиться упродовж усього часу, крім сну – з самого ранку і до пізнього вечора. Це допоможе виділити «нецільове» витрачання часу, а деякі справи «накласти» одну на одну, виконуючи їх одночасно. Наприклад, в містах, де є метро, ​​люди їдуть на роботу і вже в цей час починають виконувати якісь робочі завдання за допомогою ноутбука або інших гаджетів.

Також, крім ведення щоденника-хронометражу подій, потрібно записувати час, витрачений на кожен телефонний дзвінок, відвідування сторінок у соціальних мережах, так як за статистикою, саме телефонні розмови та пасивний скролінг інтернет-сторінок забирають найбільше часу.

Іноді виною браку часу стає не стільки відсутність організованості, скільки надлишок справ. Буває так, що людина під час планування свого часу на день або тиждень ставить перед собою занадто багато робочих завдань. У такому випадку, в першу чергу, необхідно уважно розставити пріоритети та проаналізувати, чи справді все, що заплановано, має бути виконаним за цей період часу? Якщо ні, то є сенс переглянути розпорядок та, як варіант, збільшити терміни виконання поставлених завдань або делегувати ці завдання комусь іншому.

Розглянемо види тайм-менеджменту відповідно до досліджень вітчизняного науковця Хитрої О.В. [9, с. 6]. на рис. 1.1.

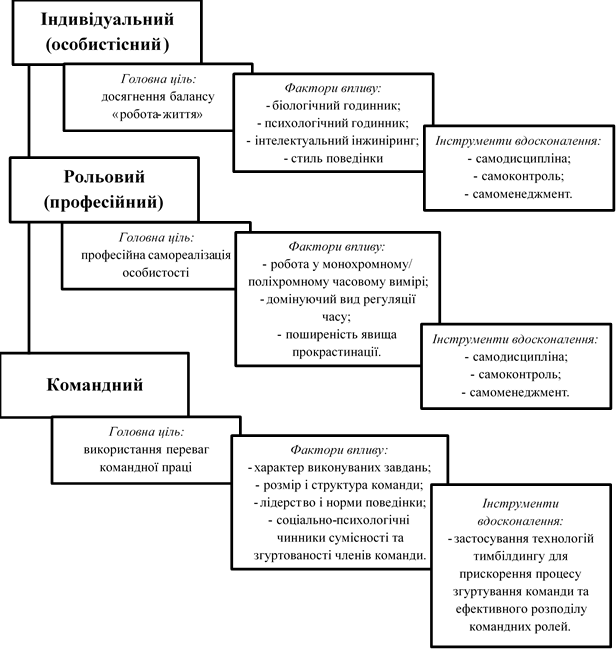


Рисунок 1.1 — Специфіка індивідуального, рольового та командного тайм-менеджменту

Отже, бачимо, що тайм-менеджмент є досить багатогранним явищем, яке має важливі функції, специфічні види та методики застосування.

## **Аналіз існуючих інформаційних технологій тайм-менеджменту.**

Окремо доречним буде розглянути програми для комп'ютерів і смартфонів, які допомагають планувати час.

* Найпопулярнішим органайзером є Microsoft Outlook. Ця програма являє собою зручний планувальник завдань з функціями поштового клієнта. Схожими програмами є Time Assist Pro, Lotus Notes, а у склад пакету MS Office також входить програма-органайзер OneNote. У час високих технологій програми потроху витісняють паперові щоденники, хоча, для деяких людей ті, як і раніше залишаються звично зручним способом планування часу. Розглянемо деякі з існуючих програм, які дозволяють управляти часом.
* OneNote. Програма пропонує досить сильний інструментарій: текстові замітки легко сортуються по каталогах, а деревоподібна структура дозволяє утримувати записи в порядку. У замітці швидко і зручно можуть бути розміщені аудіозаписи і відео кліпи. При цьому, підтримується і рукописний ввід на OneNote на планшеті. Програми підтримують синхронізацію інформації між пристроями, у тому числі Windows Mobile. У Windows 8 також є і metro-додаток, що дозволяє звертатися до нотаток online.
* Мініплан. З одного боку, це – звичайний органайзер, в який можна додавати завдання і замітки, але це – досить потужний інструмент організації. Завдяки можливості оповіщення через email і телефон, не залишиться без уваги жодна справа, а робоча панель «тасквотч» дозволяє зберегти велику кількість заміток, додаючи справи прямо на годинник і навіть малювати.
* LeaderTask – програма, яка поєднує в собі функції щоденника, планувальника завдань, персонального інформаційного менеджера, календаря, адресної книги, заміток, нагадувань, синхронізації з комунікатором та має версії для мобільних пристроїв, які працюють на Android, iPhone, iPad, розподілу завдань усередині офісу, мережеву версію (для роботи в групі, призначення та контролю доручень), статистики використання робочого часу (час, витрачений на додатки і сайти). Програма дозволяє побачити завдання протягом місяця, тижня або дня, але при цьому завжди можливо повернутися до зроблених раніше нотаток, зробити зміни, встановити нагадування, або ж зв'язати з іншими завданнями календаря. Функціонал доповнюється можливістю індивідуального налаштування (виділення пріоритетів), здатністю візуалізувати дані аби мати уявлення або хід виконання проектів, синхронізувати свої дані між пристроями на різних платформах і з MS Outlook. Цей органайзер простий в освоєнні і користуванні. Сервіс також передбачає інструкцію, в тому числі offline- і online-допомогу.
* Remember the Milk - відомий і популярний онлайновий програм-менеджер завдань, який ефективно підтримує роботу з великою кількістю завдань, які можуть розподілятися за списками. Є можливість привласнювати завданням теги і встановлювати найрізноманітніші параметри. Є розгалужена система пошуку. Крім списку справ/планувальника, даний сервіс містить у собі низку сучасних технічних рішень, покликаних розвантажити пам'ять. Серед них: настройка E-mail і sms-повідомлень про майбутні справи, можливість синхронізації з календарем Google, управління нотатками Evernote, робота з додатком при відсутності Інтернету.
* Replicon – «web-based» органайзер, який дає можливість керувати часом як великим, так і маленьким організаціям. Працівники мають можливість використовувати Replicon для передачі інформації своїм колегам, а для керівників організацій є різноманітні модулі, що допомагають дізнатися нарахування по зарплаті, ціну проекту, рахунки клієнта, результативність праці. Аккаунт підтримує також багаторівневий доступ при дотриманні ієрархії дозволів.
* MinuteHound – «web-based» тайм-менеджер, що надає можливість побачити всі процедури в реальному часі. Цей органайзер є досить гнучким в налаштуванні, містить усі необхідні інструменти для управління навантаженнями і завданнями. Завдяки MinuteHound можна бачити, хто працює, а хто – ні, скільки часу було витрачено. За допомогою даного менеджера можна простежити за робочим процесом в режимі реального часу.
* Evernote - мультиплатформенний хмарний сервіс зберігання різноманітних нотаток, документів, зображень, медіафайлів. Відомості зберігаються на сервері проекту, і можуть бути синхронізовані в середовищі всіх пристроїв. До текстових нотаток можуть бути прикріплені файли (об'ємом до 25 мегабайт), а самі замітки можна розподілити по каталогам. Всі вкладені до записів зображення розпізнаються на сервері Evernote, і підсумки включаються в індекс для пошуку. У Evernote також передбачена можливість додавання аудіо нотаток або збереження місця знаходження. Крім того, дані завжди можна зашифрувати.
* Google Keep – електронний блокнот з можливістю створення звичайних записів і пронумерованих списків, найпростіша програма, яка підтримує всі мобільні платформи, а також має браузерну і офлайн версію. Google Keep – online-органайзер від Google дозволяє моментально створювати різні замітки в будь-якому місці, яке має доступ до Інтернету. Google Keep доступний з браузера або програми на Android. Користь даного сервісу у тому, що можна отримати доступ до всіх наявних записів з будь-якого пристрою, де виконано вхід у Google-аккаунт. Отже, можна почати працювати над документом зі смартфона ще в транспорті по дорозі на роботу, а продовжити вже в офісі з ПК.
* BillQuick Online – зразок органайзера, який пристосований як для окремої людини, так і для великих корпорацій. Він корисний і для персоналу, і для менеджерів проекту, так як існує безліч альтернатив налаштувань органайзера. У процесі обліку часу, органайзер фіксує дані так, що можливо використовувати вбудовані інструменти бізнес-аналітики, які пропонують варіанти для підвищення результативності. Крім online-сервісу, BillQuick має локальні версії.

Серед розглянутих програм є досить складні та багатофункціональні застосунки з потужним програмним забезпеченням, яке здатне налаштуватися для будь-яких груп користувачів, або для індивідуального користування.

## **Висновки по розділу**

Нашою метою є створення такого застосунку, яким би було зручно користуватися у будь-якому місці, у будь-який час; який би мав інтуїтивно зрозумілий інтерфейс та можливість експорту даних у Телеграм.

Для реалізації поставленого завдання необхідно:

1. Вивчити літературу з основ тайм-менеджменту.
2. Вивчити літературу, що охоплює правила та методи побудови застосунків з використанням баз даних.
3. Вивчити принципи побудови зручних графічних інтерфейсів на основі використання PyQt5.
4. Вивчити довідкові матеріали з Python ООП.
5. Вивчити методичну документацію для опанування принципів роботи та розміщення бази даних застосунку, на сервері.
6. Розробити базу даних з нормалізацією не нижче третього рівня для застосунку.
7. Розробити інтуїтивно зрозумілий дизайн застосунку.
8. Розробити застосунок, у якому користувач зможе фіксувати витрати часу на той чи інший вид діяльності.
9. Розробити Телеграм-бота.
10. Розробити види візуалізації введеної користувачем інформації.
11. Виконати тестування застосунку для виявлення та усунення недоліків.
12. Розробити адаптивний сайт для зручного користування як з мобільних пристроїв, так і з ПК для надання користувачам можливості завантажити застосунок.

# РОЗДІЛ 2. ПРОЕКТУВАННЯ ЗАСТОСУНКУ TIMESOFT

## **Аналіз варіантів використання**

* + 1. Функціональні та нефункціональні вимоги в таблиці 2.1.

Табл. 2.1. Функціональні вимоги

|  |  |  |
| --- | --- | --- |
| **Функціональні вимоги** | **Опис** | **TimeSoft** |
| Процесно-  орієнтовані | Процеси, які повинна виконувати система | Система повинна дозволяти клієнтам:  - зареєструватися в системі;  - зв'язати створений обліковий запис з телеграм-профілем користувача;  - внести інформацію;  - додавати власні категорії;  - змінити внесену інформацію;  - видалити внесену інформацію;  - переглядати історію внесеної інформації, як за весь час, так і на обрану дату (часовий діапазон);  - переглядати внесену інформацію по обраних категоріях;  - візуалізувати інформацію в різних формах (діаграма, таблиця, графік);  - отримати прогнозні дані. |
| Інформаційно-  орієнтовані | Інформація, яку система повинна містити | Система повинна зберігати наступні дані за весь час (за винятком видалених): логін, пароль, електронну пошту, телеграм-ідентифікатор, дані про активність, перелік категорій, витрачений час на активність, дати здійснення активностей. |

Табл. 2.2. Нефункціональні вимоги

|  |  |  |
| --- | --- | --- |
| **Нефункціональні вимоги** | **Опис** | **TimeSoft** |
| Обслуговування | Фізичне і технічне середовища, в яких система функціонує | Система повинна працювати на мобільних і стаціонарних пристроях, що підтримують програму Телеграм.  Система повинна бути сумісною і функціонувати в ОС: Windows, IOS, Linux, Android. |
| Продуктивність | Швидкість, потужність і надійність системи | Швидкість взаємодії між користувачем і системою не повинна перевищувати 2-х секунд. Система візуалізує дані на основі нових введених параметрів стану протягом 10 секунд після внесення змін. Система повинна бути доступна для використання 24 години на добу 365 днів на рік при наявності інтернету, крім аварійних ситуацій. Система підтримує 10 користувачів одночасно в будь-який час. |
| Безпека | Хто і за яких обставин дозволяє доступ до системи | Кожному користувачу доступні всі внесені ним дані (за винятком видалених) в будь-який час. Доступ до даних можна отримати тільки після авторизації в системі за допомогою логіна і паролю. |
| Культурні та політичні | Культурно-політичні чинники і правові вимоги, що впливають на систему | Культурно-політичний фактор полягає у впливі думки керуючих проектом викладачів на дозвіл запуску програми.  Правові вимоги та обмеження: вимоги до оформлення проекту та використання певних структурно-логічних схем. |

* + 1. Діаграма прецедентів.

Діаграма прецедентів – це тип поведінкової діаграми, який часто використовується для аналізу різних систем. Вона дозволяє візуалізувати різні типи ролей в системі і те, як ці ролі взаємодіють з системою. Діаграма прецедентів для TimeSoft зображена на рис. 2.1.

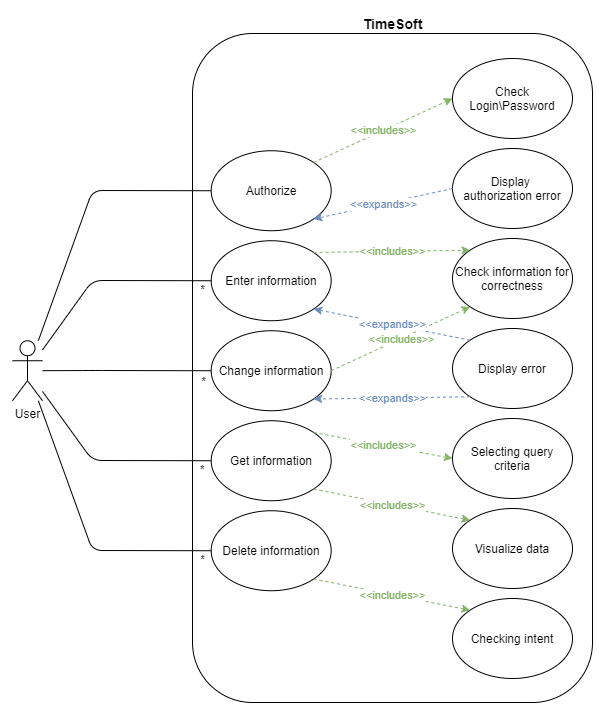


Рисунок 2.1 — Діаграма прецедентів застосунку TimeSoft

## **Моделювання процесів і даних**

* + 1. Діаграми діяльності.

При моделюванні поведінки системи виникає необхідність не тільки представити процес зміни її станів, але і деталізувати особливості алгоритмічної і логічної реалізації виконуваних системою операцій. Наочно це можна показати за допомогою діаграми діяльності. Діаграма діяльності для TimeSoft зображена на рис 2.2.

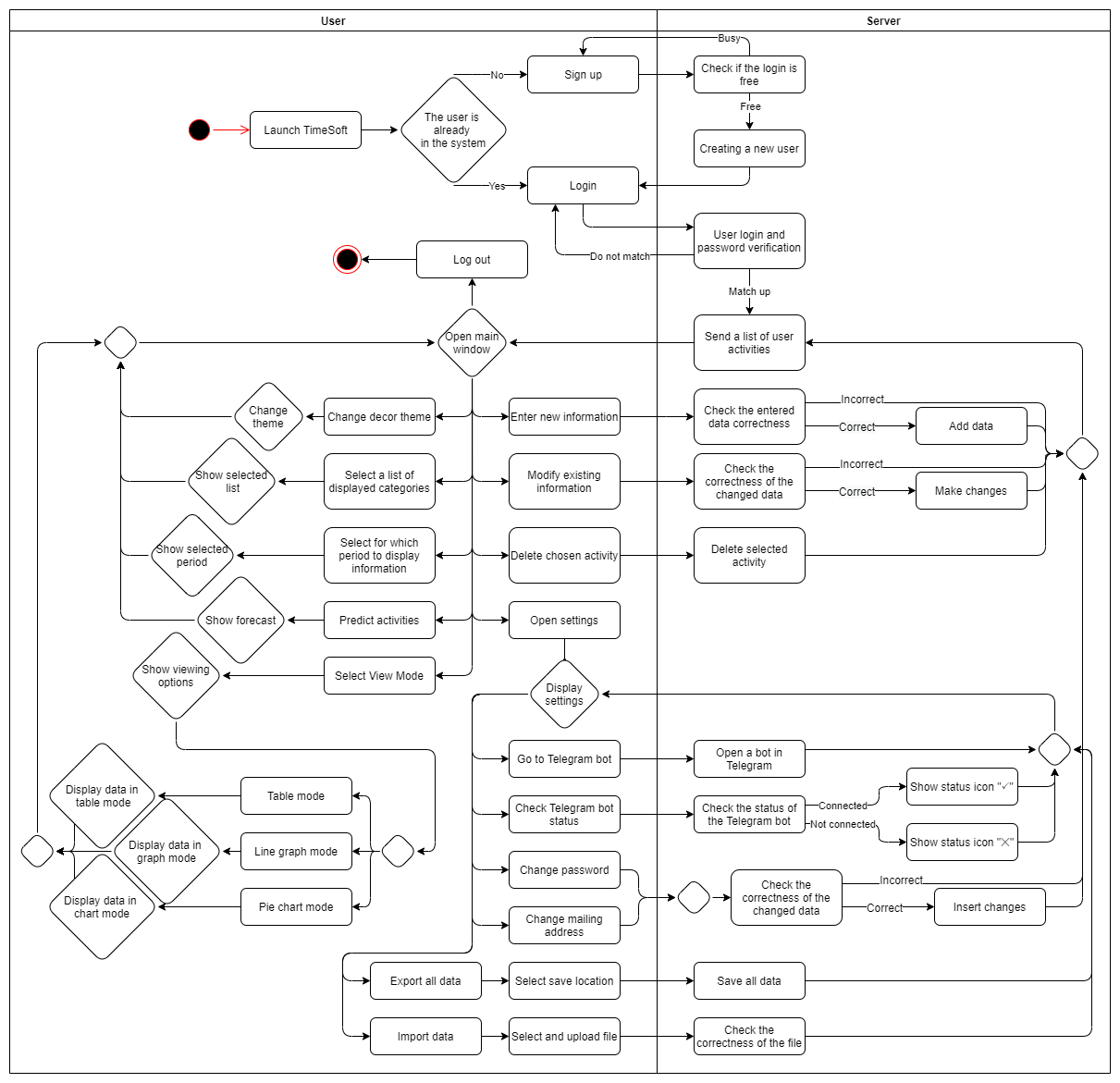


Рисунок 2.2 — Діаграма діяльності застосунку TimeSoft

* + 1. Опис ідентифікованих даних.

Ідентифікація – ототожнення, прирівнювання, уподібнення, розпізнавання іншої системи або об'єкта за наперед заданими критеріями. [10].

Перш за все, в додатку «Timfesoft» дані зберігаються в базі даних «PostgreSQL». Дані користувача – логін, електронна пошта і пароль – зберігаються в таблицях під назвами «USER\_NAME» та «USER\_PRIVATE» відповідно.

Також, є дані про активності користувача, які зберігаються в таблицях під назвами: «CATEGORY» – категорії користувача, «ACTIVITY» – дані, що містять лише назву активності та категорію, та «ACTIVITY\_LIST» – зведені дані про додані активності користувача.

Усі перераховані вище таблиці мають свій ідентифікаційний номер – так званий, унікальний ID-ідентифікатор (від англ. identity document, id) – унікальна ознака об'єкта, що дозволяє відрізняти його від інших об'єктів, тобто ідентифікувати. [11] Саме за цими ідентифікаторами і відбувається ідентифікація різноманітних даних, які вводить користувач.

Наприклад, розглянемо ідентифікацію логіну та паролю при авторизації користувача у додатку:

Перш за все, всі дані, які надходять від користувача, проходять перевірку на наявність/відсутність різноманітних символів, які можуть вивести з ладу роботу програми – наприклад, визвати аварійне незаплановане завершення програми. Теж саме стосується і довжини вхідних даних – максимальна кількість символів у рядку становить 60. Зазначені вище правила для перевірки даних стосуються наступних полів, що приймають дані від користувача:

* Вікно авторизації користувача: логін та пароль;
* Вікно реєстрації користувача: логін, електронна пошта та пароль;
* Вікно для додавання/редагування конкретної активності: назва активності, категорія, коментар до активності;
* Вікно налаштувань: нова пошта, поточний пароль, новий пароль та підтвердження нового паролю.

У вікні додавання/редагування активностей існує ще одне поле, яке має унікальну перевірку на вхідні дані – в це поле можна вводити тільки числа.

При проходженні всіх перевірок дані зберігаються у спеціальних змінних в тілі програми. Для нашого прикладу, дані – логін та пароль користувача – проходять наступну перевірку, а саме – звірку вхідних даних з уже існуючими в базі даних. Спочатку перевіряється наявність логіну у таблиці «USER\_NAME». Якщо такого логіну користувача не існує, виводиться відповідне повідомлення у графічному інтерфейсі користувача, інакше – логін пройде перевірку на його наявність, і ідентифікатор логіну з таблиці «USER\_NAME» – user\_n\_name, запам’ятається у відповідну змінну.

Тепер ми маємо перший id-ідентифікатор логін. Наступним кроком буде дізнатися приватний ідентифікатор для звірки паролю – user\_p\_id. Унікальні приватні ідентифікатори зберігаються у таблиці «USER» – маючи унікальний ідентифікатор user\_n\_name, ми повинні спочатку дізнатися унікальний ідентифікатор користувача – user\_id, що зберігається у останній зазначеній таблиці. Для цього, виконується sql-зпит, що містить наступну структуру:

SELECT user\_id WHERE user\_n\_name = «отриманий заздалегідь ідентифікатор user\_n\_name».

Для всіх зазначених вище кроків отримання різноманітних ідентифікаторів запити будуть відповідними.

Після отримання user\_id потрібно буде дізнатися пароль користувача, відповідно до його логіну. Для цього виконаємо два останніх запити, які дадуть нам можливість дізнатися user\_p\_id – унікальний приватний ідентифікатор користувача (таб. «USER»), а потім за його допомогою дізнатися і пароль – user\_p\_password (таб. «USER\_PRIVATE»).

user\_p\_id:

SELECT user\_p\_id WHERE user\_id = «отриманий заздалегідь ідентифікатор user\_id з попереднього запиту».

user\_p\_password:

SELECT user\_p\_password WHERE user\_p\_id = «отриманий заздалегідь ідентифікатор user\_p\_id з попереднього запиту».

Тепер ми маємо всі три поля з бази даних, що дозволять нам провести перевірку на звірку введених користувачем даних. Ця перевірка, як і найперша у цьому розділі, проводиться у відповідних методах класів мови програмування Python.

* + 1. Формат зберігання даних.

Як вже було зазначено у пункті «Опис ідентифікованих даних», додаток «Timesoft» використовує базу даних «PostgreSQL» для зберігання та сортування даних, що надходять від користувача.

Перш за все, розглянемо які формати зберігання даних використовує база даних.

Наступні формати з англійської слід сприймати так:

* **UNIQUE** – приставка до типу даних, що означає «унікальне поле». Унікальний (лат.): надзвичайний у якомусь відношенні; рідкісний, винятковий, неповторний, неперевершений, небачений, нечуваний [12].
* **PRIMARY KEY** – первинний ключ. Первинний ключ — атрибут або набір атрибутів, який однозначно ідентифікує кортеж даного відношення. Первинний ключ обов'язково унікальний [13].
* **FOREIGN KEY** – вказує на стовпець таблиці, який буде представлений як зовнішній ключ [14].
* **REFERENCES "n" (x)** – вказує на ім'я пов'язаної таблиці, а x вказуэ на ім'я пов'язаного стовпця, на який буде вказувати зовнішній ключ [15].
* **ON DELETE CASCADE ON UPDATE CASCADE** – задаэ дію при видаленні і оновленні рядків з пов'язаної таблиці [16].
* **SERIAL** – лічильник, який збільшується на 1 кожен раз, коли додається новий запис [17].
* **NOT NULL** – це поле, обов'язково повинно бути заповнене [18].
* **NULL** – це поле, може перебувати порожнім [19].
* **INT** – ціле число, діапазон якого з -2147483648 до 2147483647 [20].
* **SMALLINT** – ціле число з -32768 до 32767 [20].
* **VARCHAR (n)** – текст з обмеженням максимальної кількості символів у n, де n – ціле число. Максимальна кількість символів (n) 65 535 [21].
* **DATE** – дата у форматі «YYYY-MM-DD» [22].

Таблиця «USER»:

Поле USER\_ID SERIAL NOT NULL UNIQUE,

Поле USER\_N\_ID VARCHAR (64) NOT NULL UNIQUE,

Поле USER\_P\_ID VARCHAR (64) NOT NULL UNIQUE,

Поле PRIMARY KEY (USER\_ID, USER\_N\_ID, USER\_P\_ID).

Таблиця «USER\_NAME»:

Поле USER\_N\_ID VARCHAR (64) NOT NULL UNIQUE

REFERENCES "USER" (USER\_N\_ID)

ON DELETE CASCADE ON UPDATE CASCADE PRIMARY KEY

Поле USER\_N\_name VARCHAR (64) NOT NULL UNIQUE

Поле USER\_N\_Telegram VARCHAR (64) NULL UNIQUE.

Таблиця «USER\_PRIVATE»:

Поле USER\_P\_ID VARCHAR (64) NOT NULL UNIQUE

REFERENCES "USER" (USER\_P\_ID)

ON DELETE CASCADE ON UPDATE CASCADE PRIMARY KEY,

Поле USER\_P\_email VARCHAR (64) NOT NULL UNIQUE,

Поле USER\_P\_password VARCHAR (64) NOT NULL.

Таблиця «SETTINGS»:

Поле USER\_ID INT NOT NULL REFERENCES "USER" (USER\_ID)

ON DELETE CASCADE ON UPDATE CASCADE PRIMARY KEY,

Поле SET\_theme VARCHAR (1024) NULL,

Поле SET\_preferences VARCHAR (4096) NULL.

Таблиця «CATEGORY»:

Поле CAT\_ID SERIAL NOT NULL UNIQUE,

Поле USER\_ID INT NOT NULL REFERENCES "USER" (USER\_ID)

ON DELETE CASCADE ON UPDATE CASCADE,

Поле CAT\_name VARCHAR (64) NOT NULL,

Поле PRIMARY KEY (USER\_ID, CAT\_name),

Поле UNIQUE (USER\_ID, CAT\_name).

Таблиця «ACTIVITY\_LIST»:

Поле ACTL\_ID SERIAL NOT NULL UNIQUE,

Поле USER\_ID INT NOT NULL,

Поле ACTL\_name VARCHAR(64) NOT NULL,

Поле CAT\_name VARCHAR(64) NOT NULL,

Поле FOREIGN KEY (USER\_ID, CAT\_name)

REFERENCES "CATEGORY" (USER\_ID, CAT\_name)

ON DELETE CASCADE ON UPDATE CASCADE,

Поле PRIMARY KEY (USER\_ID, ACTL\_name, CAT\_name),

Поле UNIQUE (USER\_ID, ACTL\_name, CAT\_name).

Таблиця «ACTIVITY»:

Поле ACT\_ID SERIAL NOT NULL UNIQUE PRIMARY KEY,

Поле USER\_ID INT NOT NULL,

Поле ACTL\_name VARCHAR(64) NOT NULL,

Поле ACT\_time SMALLINT NOT NULL,

Поле ACT\_date DATE NOT NULL,

Поле CAT\_name VARCHAR(64) NOT NULL,

Поле ACT\_comment VARCHAR(512) NULL,

Поле FOREIGN KEY (USER\_ID, ACTL\_name, CAT\_name)

REFERENCES "ACTIVITY\_LIST" (USER\_ID, ACTL\_name, CAT\_name)

ON DELETE CASCADE ON UPDATE CASCADE.

У тілі програми змінні, що зберігають дані, які надходять з бази даних, конвертуються у відповідні формати згідно з таблицею у кінці розділу.

## **Проектування архітектури**

* + 1. Архітектура бази даних.

Програмний застосунок TimeSoft зберігає дані в базі даних PostgreSQL, що працює на мові запитів SQL. Ми обрали PostgreSQL для розгортання на сервері Heroku. Програмний застосунок написаний мовою програмування Python 3. Для візуалізації інтерфейсу на GUI, використовується PyQt, який є реалізацією потужного фреймворка Qt для мови Python. Користувач може користуватися програмним застосунком через свій персональний комп'ютер, крім цього може підключити свій акаунт до телеграм боту TimeSoft, щоб дивитися інформацію свого акаунту віддалено зі смартфона або планшета.

Для успішного функціонування застосунку TimeSoft необхідна база даних з нормалізацією 3-го рівня (рис. 2.3).

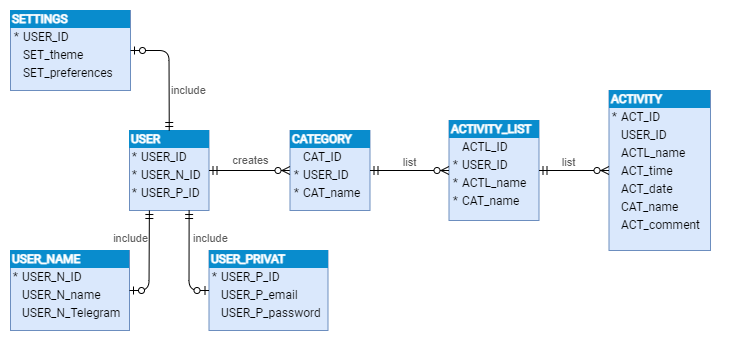


Рисунок 2.3 — ER-діаграма застосунку TimeSoft

База даних включає наступні параметри:

* інформація про користувача;
* інформація про обрані налаштування користувача;
* інформація про обрані категорії користувача;
* інформація про обрані активності користувача;
* інформація про зведені дані по двох таблицях з додаванням коментаря, дати активності, а також часу, витраченого на активність.

SQL запити на створення таблиць бази даних представлені на рис. 2.1-2.14

SQL запит на створення таблиці «Користувач» представлений на рис. 2.4.

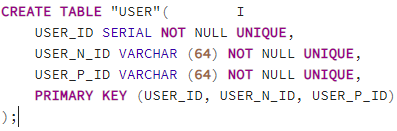


Рисунок 2.4 — SQL запит на створення таблиці «Користувач»

Приклад даних таблиці «Користувач» представлений на рис. 2.5.



Рисунок 2.5 — Приклад даних таблиці «Користувач»

SQL запит на створення таблиці представлений«Налаштування користувача» на рис. 2.6.

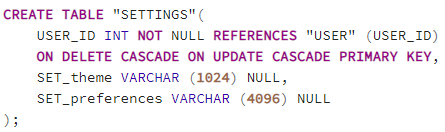


Рисунок 2.6 — SQL запит на створення таблиці «Налаштування користувача»

Приклад даних таблиці «Налаштування користувача» зображено на рис. 2.7.



Рисунок 2.7 — Приклад даних таблиці «Налаштування користувача»

SQL запит на створення таблиці «Логін користувача» зображений на рис. 2.8.

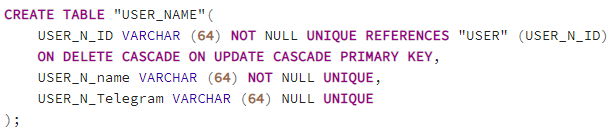


Рисунок 2.8 — SQL запит на створення таблиці «Логін користувача»

Приклад даних таблиці «Логін користувача» зображено на рис. 2.9.



Рисунок 2.9 — SQL Приклад даних таблиці «Логін користувача»

SQL запит на створення таблиці «Пароль користувача» зображено на

рис. 2.10.

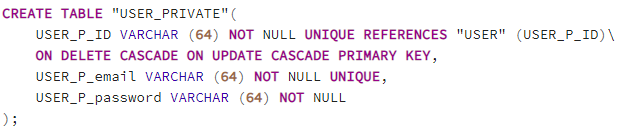


Рисунок 2.10 — SQL запит на створення таблиці «Пароль користувача»

Приклад даних таблиці «Пароль користувача» зображено на рис. 2.11.



Рисунок 2.11 — Приклад даних таблиці «Пароль користувача»

SQL запит на створення таблиці «Категорія користувача» зображено на

рис. 2.12.

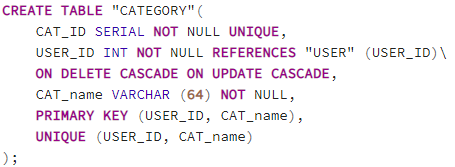


Рисунок 2.12 — SQL запит на створення таблиці «Категорія користувача»

Приклад даних таблиці «Категорія користувача» зображено на рис. 2.13.



Рисунок 2.13 — Приклад даних таблиці «Категорія користувача».

SQL запит на створення таблиці «Активність користувача» зображено на рис. 2.14.

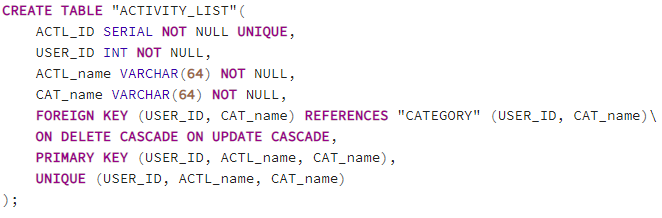


Рисунок 2.14 — SQL запит на створення таблиці «Активність користувача»

Приклад даних таблиці «Активність користувача» зображено на рис. 2.15.



Рисунок 2.15 — Приклад даних таблиці «Активність користувача»

SQL запит на створення таблиці «Зведена інформація користувача» зображено на рис. 2.16.

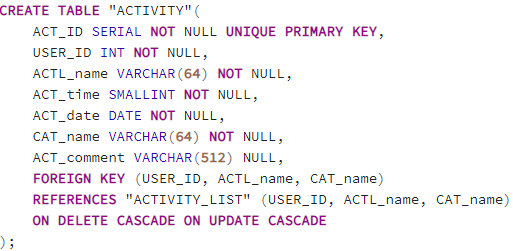


Рисунок 2.16 — SQL запит на створення таблиці «Зведена інформація користувача»

Приклад даних таблиці «Зведена інформація користувача» зображено на рис. 2.17.



Рисунок 2.17 — Приклад даних таблиці «Зведена інформація користувача»

* + 1. Вимоги до безпеки.

Важливо зазначити, хто і за яких обставин дозволяє доступ до системи. Отже, кожному користувачу доступні всі внесені ним дані (за винятком видалених) в будь-який час. Доступ до даних можна отримати тільки після авторизації в системі за допомогою логіна і паролю.

## **Проектування графічного інтерфейсу користувача**

* + 1. Діаграма навігації по екранах.

На діаграмі навігації ми зможемо побачити, який шлях робить користувач у програмі, щоб досягти своєї цілі.

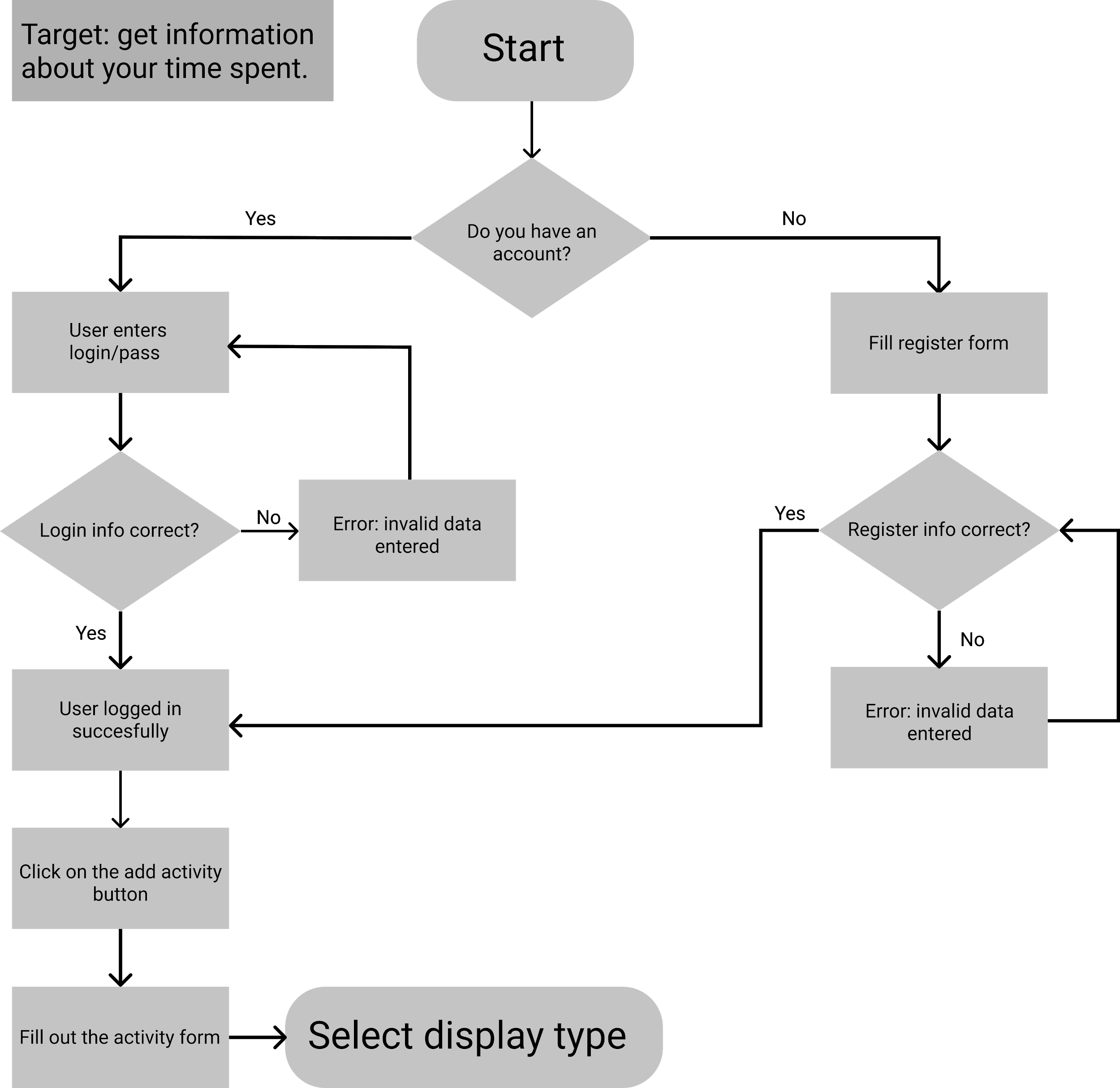


Рисунок 2.18 — Приклад даних таблиці «Зведена інформація користувача»

* + 1. Діаграма структури інтерфейсу.

Діаграма структури інтерфейсу застосунку показує причинно-наслідковий зв'язок між вибором користувача та результатом цього вибору. Іншими словами, ця діаграма ілюструє, що станеться після того, як користувач вибере ту чи іншу функцію у застосунку.

Діаграма структури інтерфейсу застосунку TimeSoft представлена на рис. 2.19.

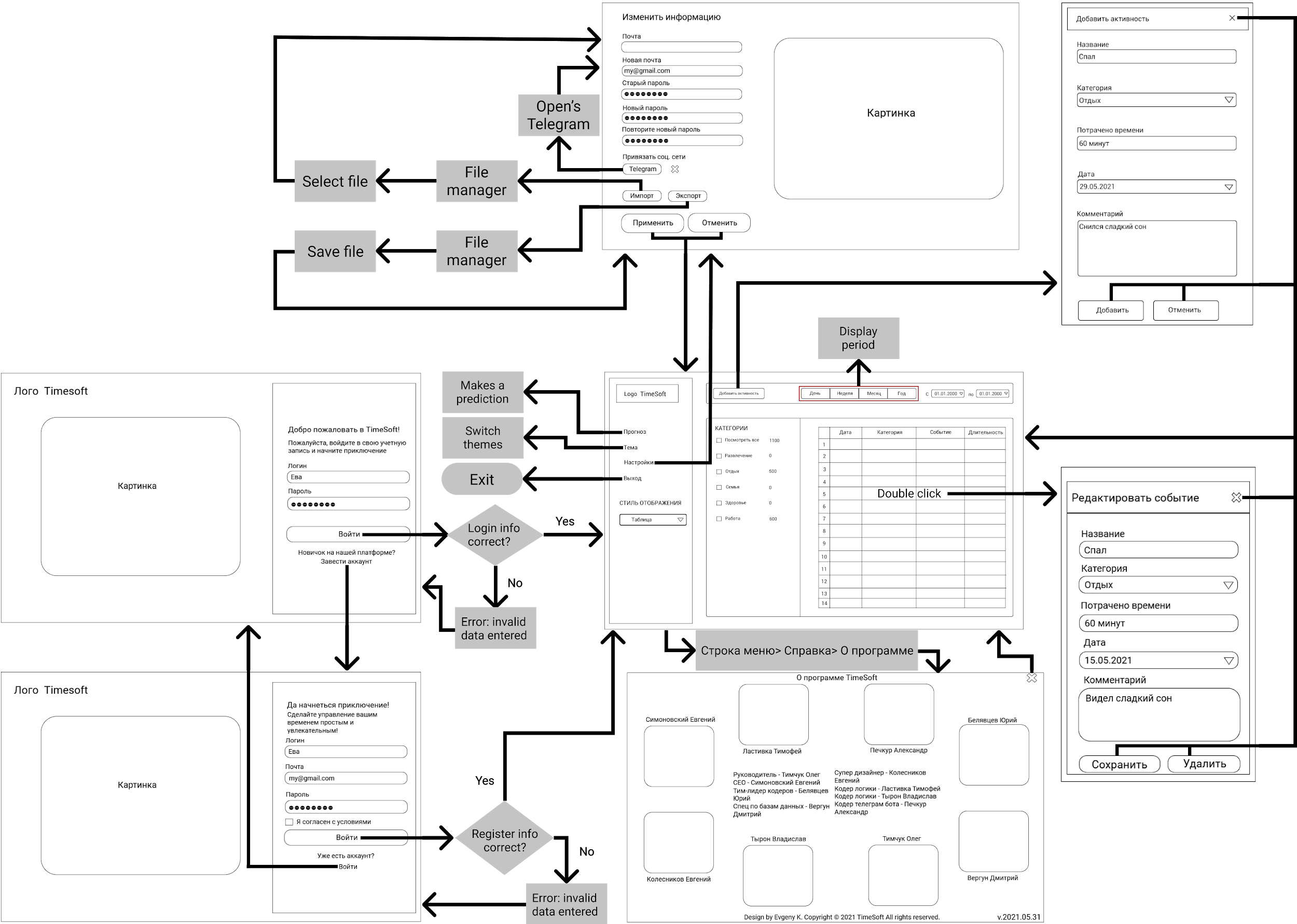


Рисунок 2.19 — Діаграма структури інтерфейсу

* + 1. Вайрфрейми.

Вайрфрейм – це макет, який дозволяє побачити, що і де буде розташовано у дизайні застосунку.

У вікні авторизації (рис. 2.20) ми можемо увійти у додаток. Якщо користувач немає аккаунту, то він зможе зареєструватися.

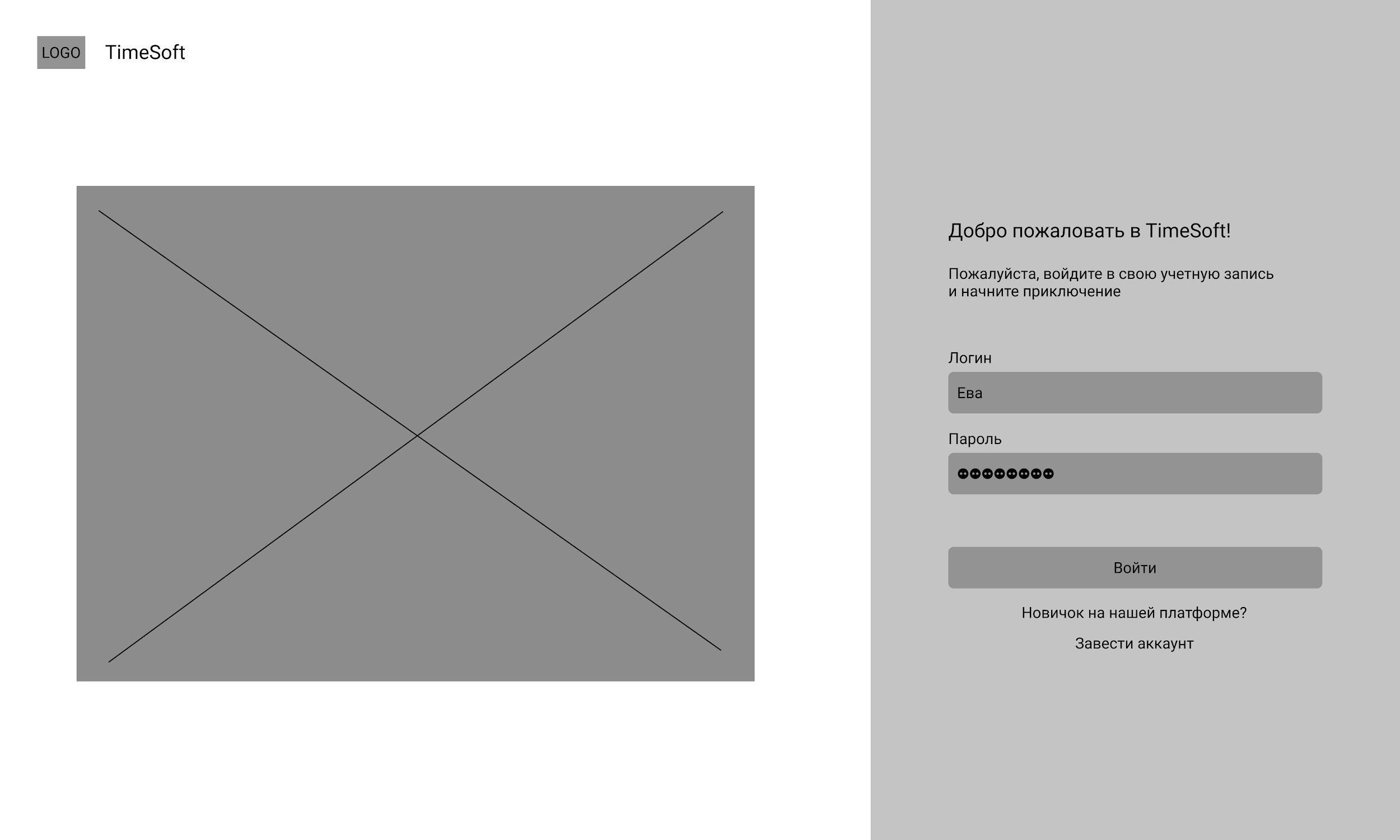


Рисунок 2.20 — Вайрфрейм вікна авторизації.

У вікні реєстрації (рис. 2.21) користувач вводить свої дані для того, щоб зареєструвати свій аккаунт. Якщо він вже має аккаунт, тоді можно перейти до попереднього вікна.

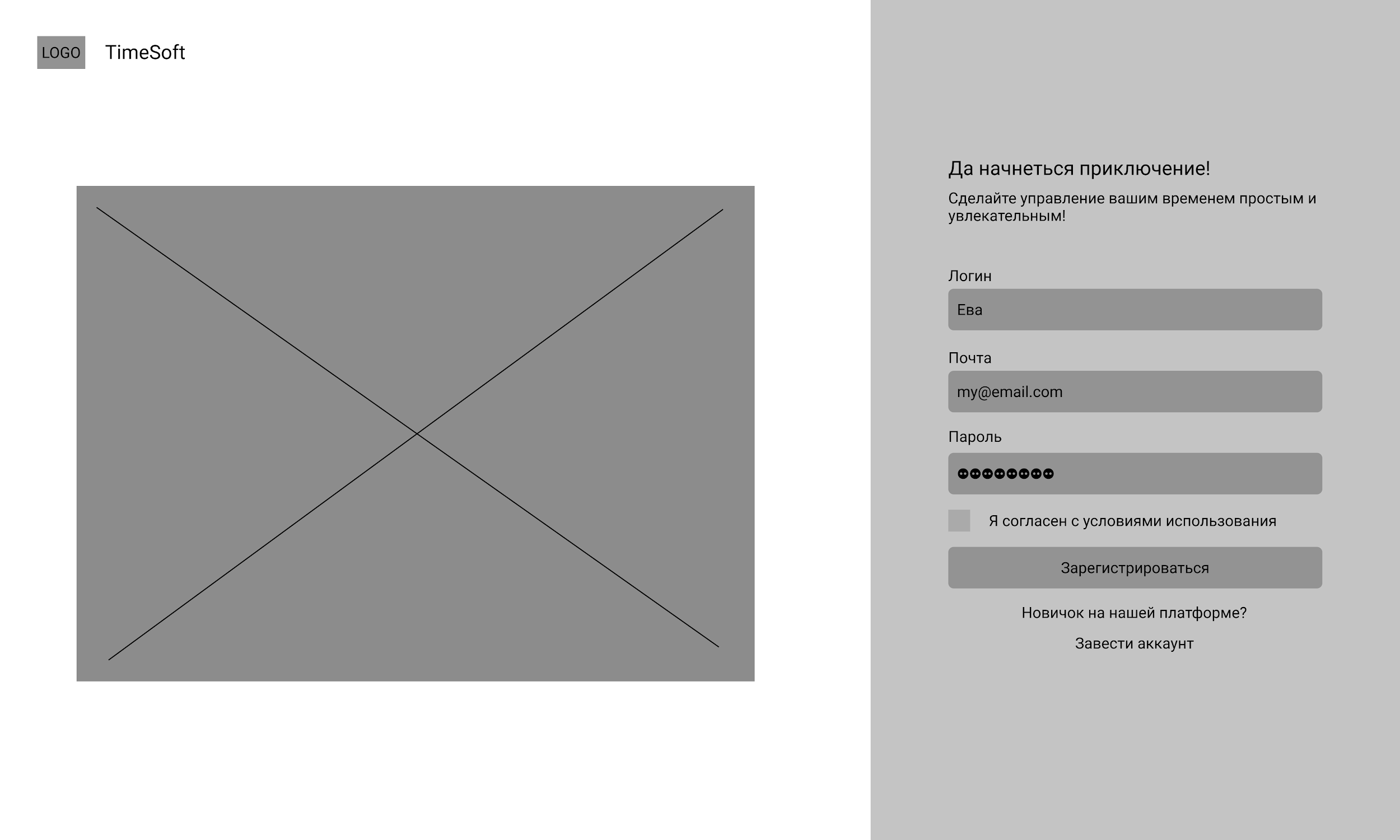


Рисунок 2.21 — Вайрфрейм вікна реєстрації

У головному вікні (рис. 2.22) користувач зможе використовувати всі функції, які він бачить.

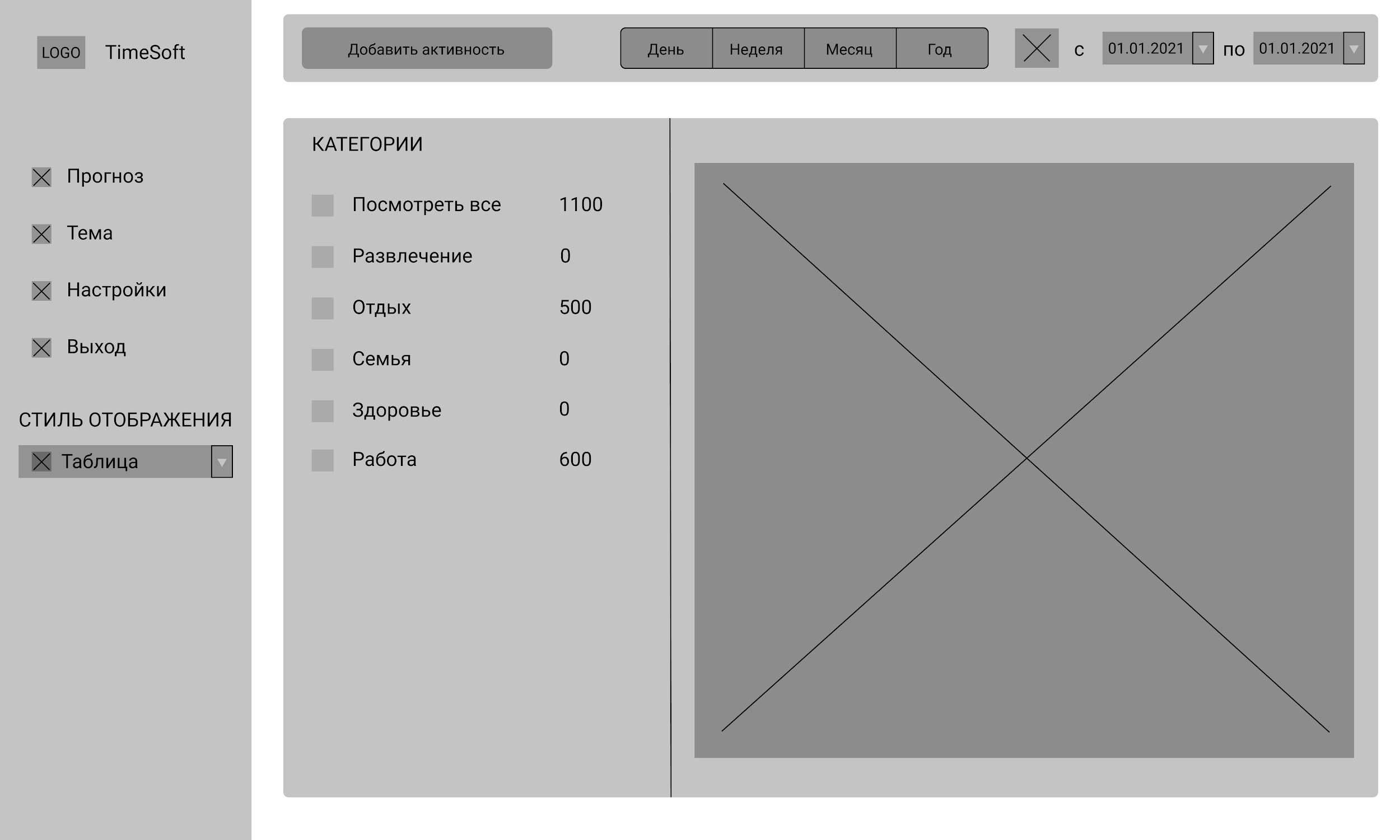


Рисунок 2.22 — Вайрфрейм головного вікна застосунку TimeSoft

У вікні налаштувань (рис. 2.23) користувач за потреби може змінити інформацію, що необхідна для авторизації, та експортувати/імпортувати свої активності.

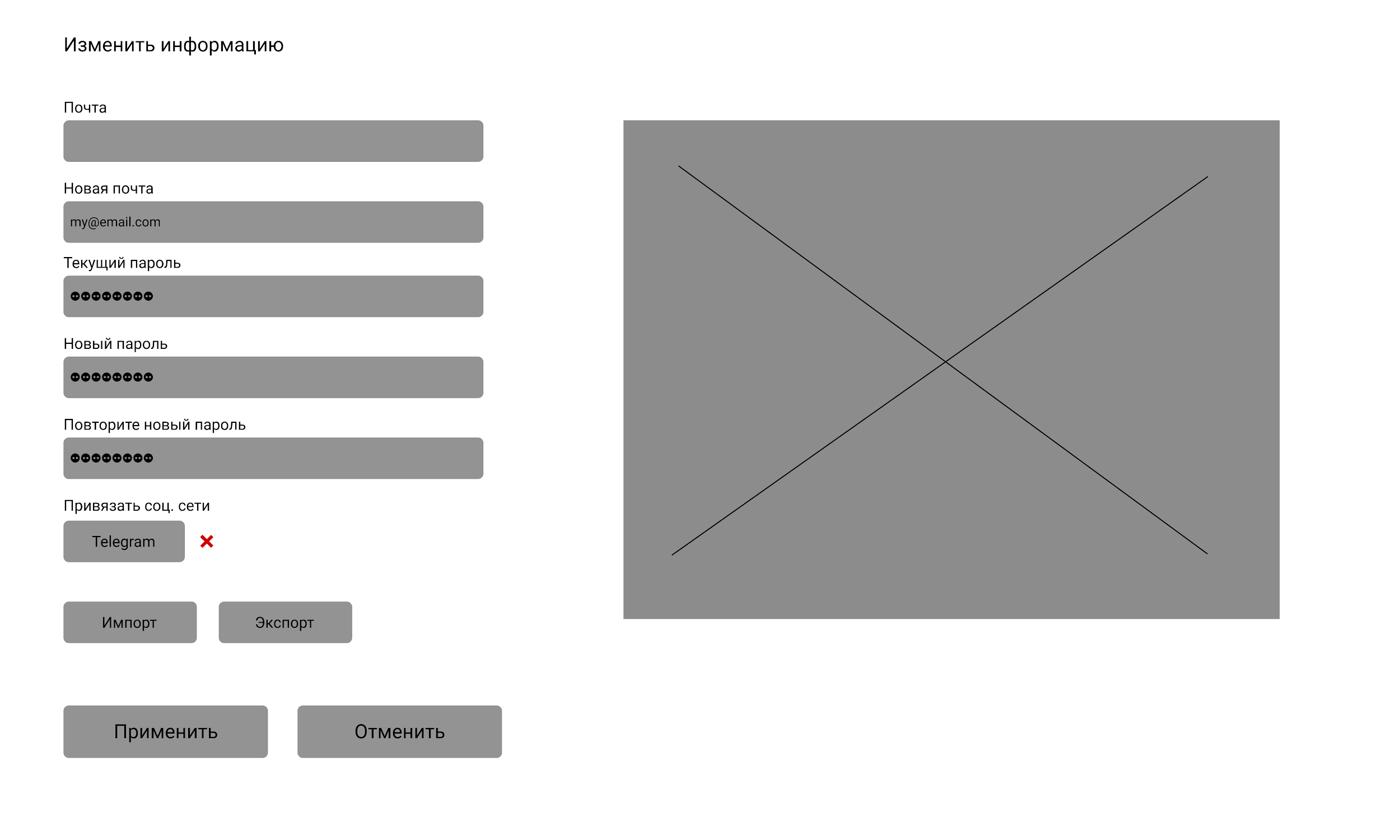


Рисунок 2.23 — Вайрфрейм вікна налаштувань

У вікні додавання активності (рис. 2.24) користувач зможе додавати свої події та категорії, а у вікні редагування активності (рис. 2.25) користувач може редагувати обрану подію.

|  |  |
| --- | --- |
| Рисунок 2.24 — Вайрфрейм вікна додавання активності | Рисунок 2.25 — Вайрфрейм вікна редагування активності |

У вікні про застосунок (рис. 2.26) користувач може переглянути інформацію про команду розробників та версію додатка.

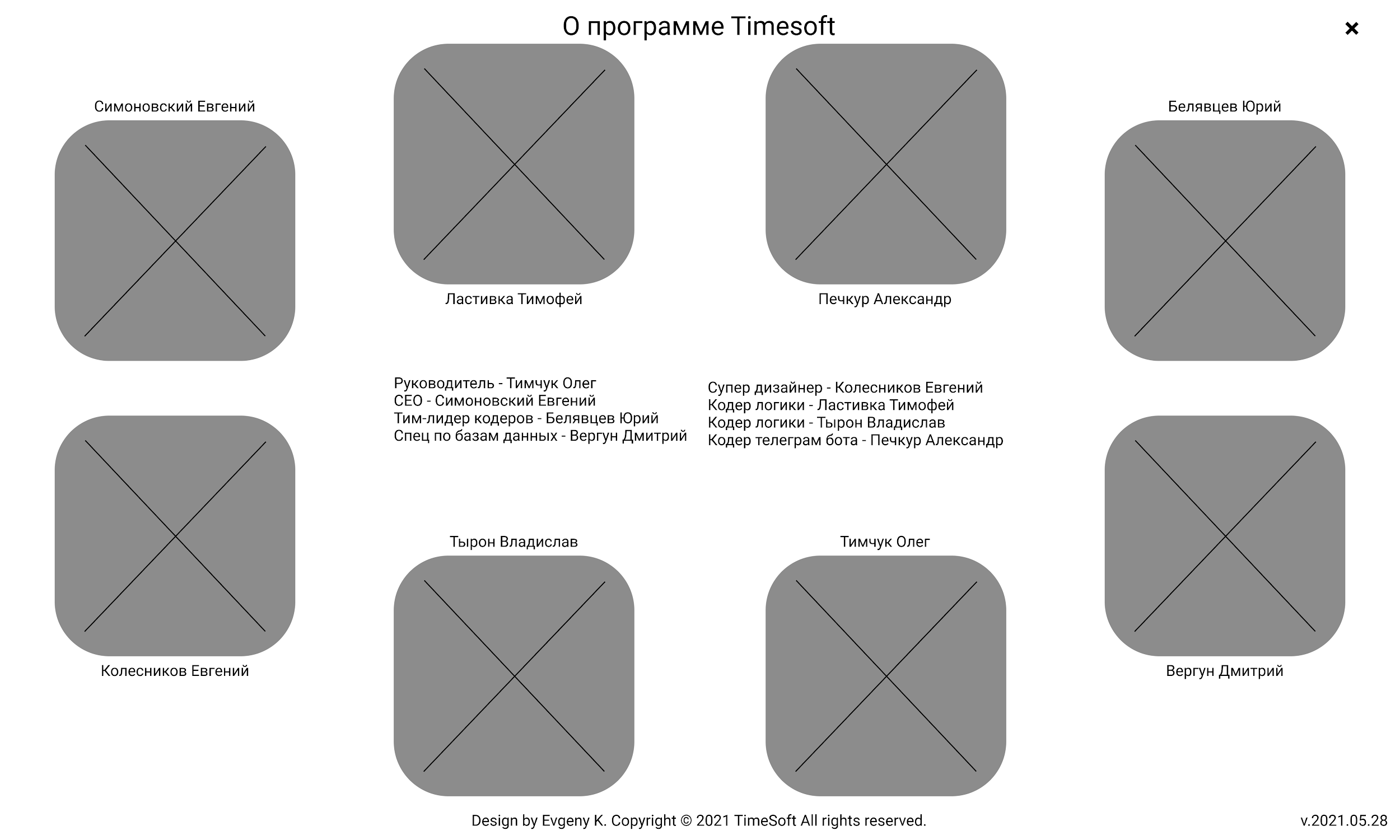


Рисунок 2.26 — Вайрфрейм вікна про застосунок

## **Проектування структури**

* + 1. Діаграма класів визначає типи класів системи і різного роду статичні зв'язки, які існують між ними рис. 2.27. На діаграмах класів зображуються також атрибути класів, операції класів та обмеження, які накладаються на зв'язки між класами. Основними елементами є класи і зв'язки між ними. Класи характеризуються за допомогою атрибутів і операцій.

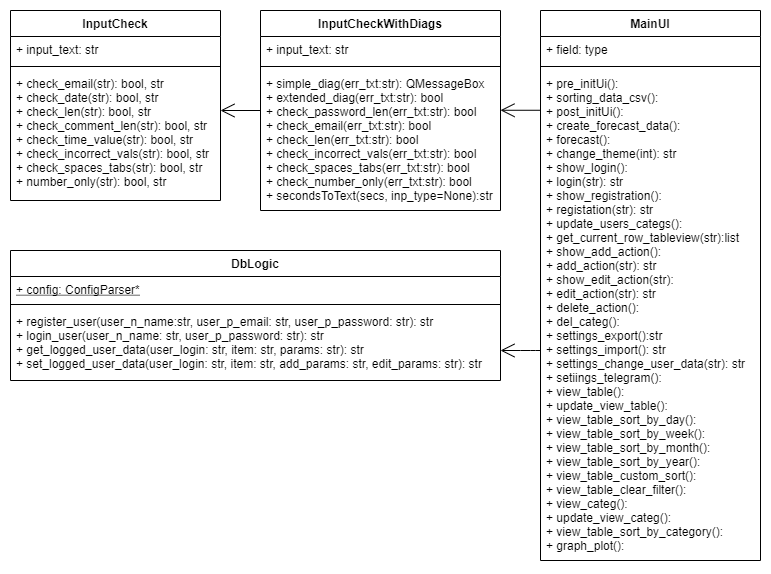


Рисунок 2.27 — Диаграмма классов

## **Висновки по розділу**

Результатом проектування застосунку «TimeSoft» є створення архітектури програмних та апаратних компонентів зі спроектованим графічним інтерфейсом користувача. У структурі застосунку було визначено класи, основні та другорядні сутності, а також послідовність подій, що зображені на відповідних UML-діаграмах. Також були визначені наступні компоненти: модуль для виконання SQL-запитів та інших операцій з базою даних, модуль для обробки часу і дати, бібліотека для інтерактивної візуалізації графіків, сховище даних, рівень доступу до даних, рівень програми та рівень інтерфейсу, а також формат зберігання даних.

# РОЗДІЛ 3. РЕАЛІЗАЦІЯ ЗАСТОСУНКУ TIMESOFT

## **Особливості реалізації**

Для розробки програми буде використана мова програмування «Python», для виконання операцій з базою даних – модуль «psycopg2», для обробки часу/дати – модуль «datetime», для обробки і аналізу даних – програмна бібліотека «pandas», а також бібліотека «pyqtgraph» для побудови та інтерактивної візуалізації графіків. Для створення бота, що працює з мессенджером «Телеграм», буде потрібна бібліотека зв'язку з сервером. Будемо використовувати для цього бібліотеку «telebot».

Мова Python – це інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією, що підтримує об’єктно-орієнтовану, імперативну та функціональну парадигми. Завдяки строгій динамічній типізації та виділенню блоків коду за допомогою відступів, Python дає можливість швидко написати код, що підкріплюється високим рівнем сприйняття тексту в порівняння з іншими мовами програмування. Для розробки програми застосунку була вибрана мова програмування Python, основними перевагами якої є наступні пункти:

* класи є одночасно об'єктами з усіма нижче наведеними можливостями;
* повна інтроспекція (можливість визначити тип і структуру об’єкту під час виконання програми);
* інтуїтивний та чистий синтаксис;
* ця мова зручна для розв'язання математичних проблем;
* стандартний дистрибутив, який має велику кількість корисних модулів;
* відкритий код, що дає можливість редагувати його іншим користувачам

Зважаючи на наведені вище пункти, була обрана мова програмування «Python». Це дозволило суттєво прискорити написання коду та зменшити кількість додаткових модулів та бібліотек. Документацію для всіх модулів та бібліотек можно знайти на офіційному сайті.

Для середовища обробки будемо використовувати Visual Studio Code.

## **Конструювання**

* + 1. Діаграма компонентів.

Діаграми компонентів дозволяють зобразити модель системи на фізичному рівні. Елементи даної діаграми – фізичні заміщувані модулі системи. Кожен компонент є повністю незалежним елементом системи.

Основне призначення діаграм компонентів – поділ системи на елементи, які мають стабільний інтерфейс і утворюють єдине ціле. Це дозволяє створити ядро системи, яка не буде змінюватися у відповідь на зміни, що відбуваються на рівні підсистем. Діаграма компонентів застосунку TimeSoft представлена на рис. 3.1.

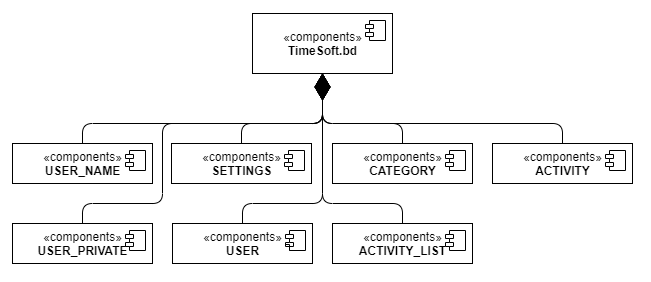


Рисунок 3.1. — Діаграма компонентів застосунку TimeSoft

* + 1. Діаграма пакетів.

Пакет (package) – це конструкція UML, призначена для упорядкування UML-моделей, а також для групування класів.

Пакети зв'язуються один з одним спеціальним відношенням – залежністю. Це спрямоване відношення, і йде воно від того пакету, який залежить, до пакету, який необхідний тому, першому, залежному. Це означає, що використовуваний пакет містить опис конструкцій, які залежний пакет імпортує, а не реалізує сам.

Досить важливо скласти діаграму пакетів під час процесу конструювання, тому що помилки при проектуванні структури пакетів застосунку часто призводять до значних незручностей, додаткової роботи і до втрати концептуальної цілісності додатки.

Отже, діаграма пакетів застосунку TimeSoft представлена на рис. 3.2.

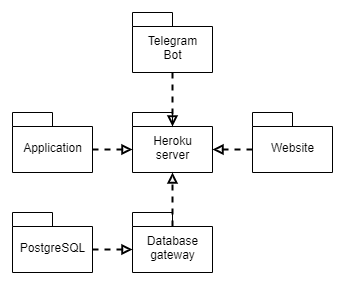


Рисунок 3.2 — Діаграма пакетів застосунку TimeSoft

* + 1. Діаграма розгортування.

Графічне представлення ІТ-інфраструктури може допомогти більш раціонально розподілити компоненти системи по вузлах мережі, від чого залежить в тому числі і продуктивність системи. Для цього побудуємо діаграму розгортування. Така діаграма також може допомогти вирішити безліч допоміжних завдань, пов'язаних, наприклад, із забезпеченням безпеки.

Діаграма розгортування показує топологію системи і розподіл компонентів системи по її вузлів, а також з'єднання – маршрути передачі інформації між апаратними вузлами. Діаграма розгортування застосунку TimeSoft представлена на рис. 3.3.

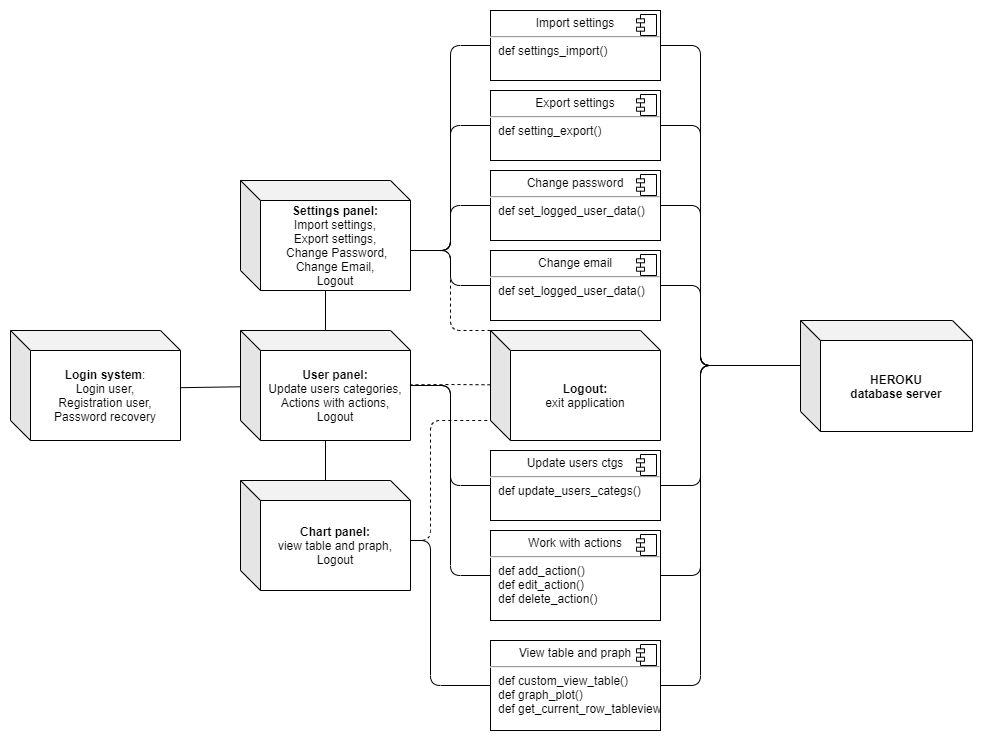


Рисунок 3.3 — Діаграма розгортування застосунку TimeSoft

## **Реалізація графічного інтерфейсу користувача**

Проектування представлення дозволяє в деталях показати, яка інформація і елементи управління повинні виводитися на формі застосунку.

* + 1. Структурна схема вікна авторизації (рис. 3.4) складається з декількох частин:
       1. Шапка вікна. Містить такі елементи:
* Іконка логотипу;
* Назва продукту.
  + - 1. Головна панель. Містить такі елементи:
* Привітання;
* Заголовок;
* Поле для введення логіну;
* Поле для введення паролю;
* Кнопка авторизуватися в застосунку;
* Кнопка-посилання для реєстрації нового аккаунта.
  + - 1. Ліва панель. Містить такий елемент:
* Картинка.



Рисунок 3.4 — Структурна схема вікна авторизації

* + 1. Структурна схема вікна реєстрації (рис. 3.5) складається з декількох частин:
       1. Шапка вікна. Містить такі елементи:
* Іконка логотипу;
* Назва продукту.
  + - 1. Головна панель. Містить такі елементи:
* Привітання;
* Заголовок;
* Поле для введення логіну;
* Поле для введення пошти;
* Поле для введення паролю;
* Прапорець я ознайомлений з правилами використання;
  + - 1. Ліва панель. Містить такий елемент:
* Картинка.

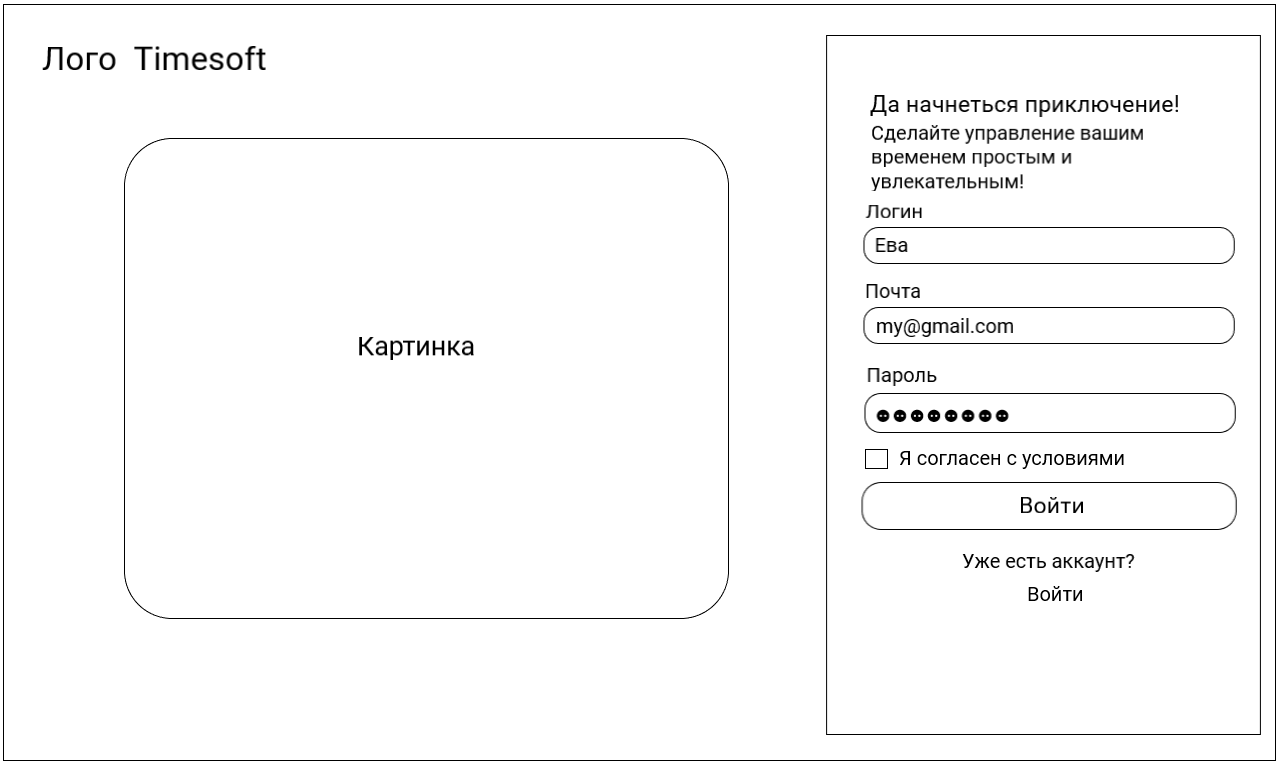


Рисунок 3.5 — Структурна схема вікна реєстрації

* + 1. Структурна схема головного вікна застосунку TimeSoft (рис. 3.6) складається з чотирьох частин:
       1. Шапка вікна містить такі елементи:
* Іконка логотипу;
* Назва продукту.
  + - 1. Панель меню містить такі елементи:
* Кнопка прогнозу;
* Кнопка зміни теми;
* Кнопка налаштування;
* Кнопка виходу з застосунку;
* Випадаючий список.
  + - 1. Панель навігації. Містить такі елементи:
* Кнопка додати діяльність;
* Кнопка день;
* Кнопка тиждень;
* Кнопка місяць;
* Кнопка рік;
* Календар початок вибраного періоду;
* Календар кінець вибраного періоду.
  + - 1. Головна панель. Містить такі елементи:
* Прапорці категорій;
* Табличний вигляд відображення інформації. В подальшому, в цьому блоці будуватимуться діаграми та графіки.

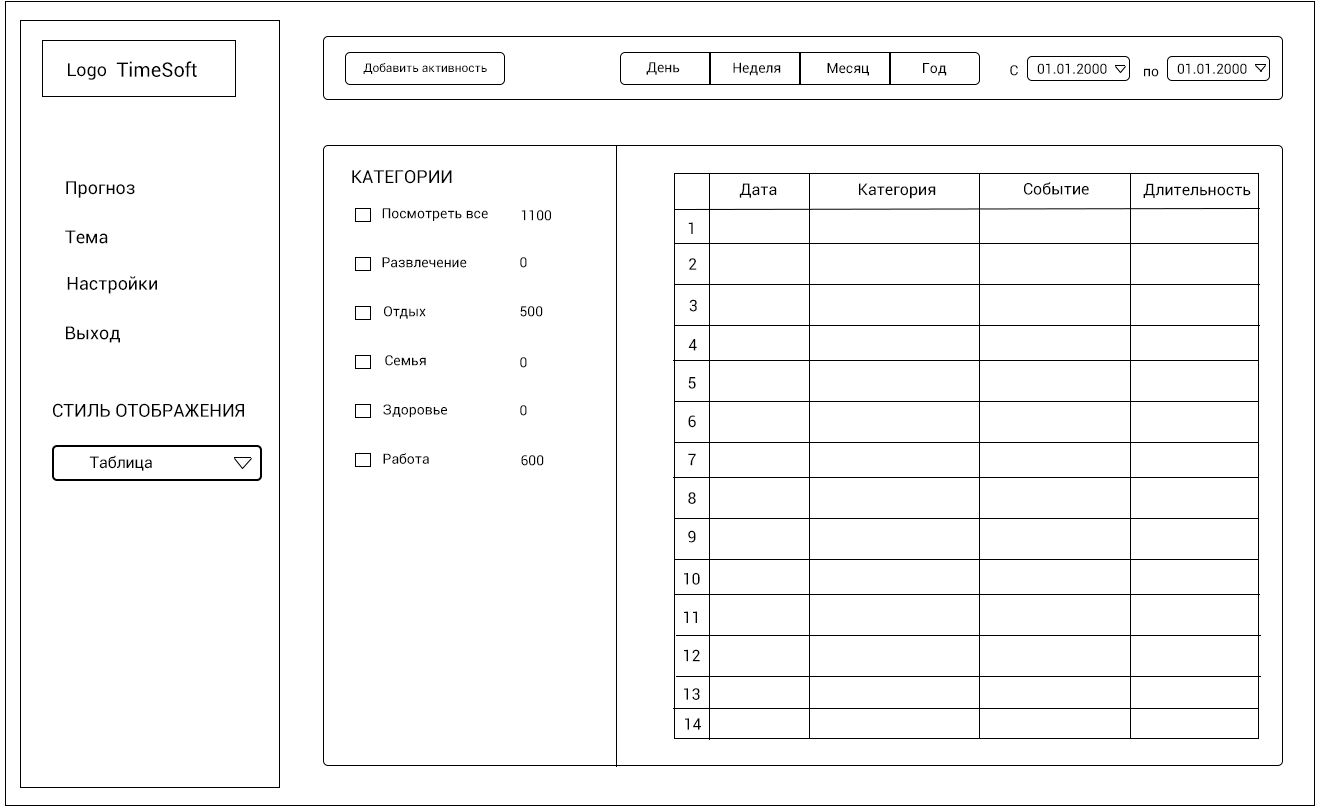


Рисунок 3.6 — Структурна схема головного вікна застосунку TimeSoft

* + 1. Структурна схема вікна налаштування (рис. 3.7) складається з декількох частин:
       1. Основна панель. Містить такі елементи:
* Назва розділу
* Поле для введення пошти;
* Поле для введення нової пошти;
* Поле для введення застарілого паролю;
* Поле для введення нового паролю;
* Поле для введення підтвердження нового паролю;
* Кнопка Telegram;
* Іконка індикатора прив’язки Telegram;
* Кнопка імпорту бази даних;
* Кнопка експорт бази даних;
* Кнопка застосувати;
* Кнопка відмінити;
  + - 1. Ліва панель. Містить такий елемент:
* Картинка.

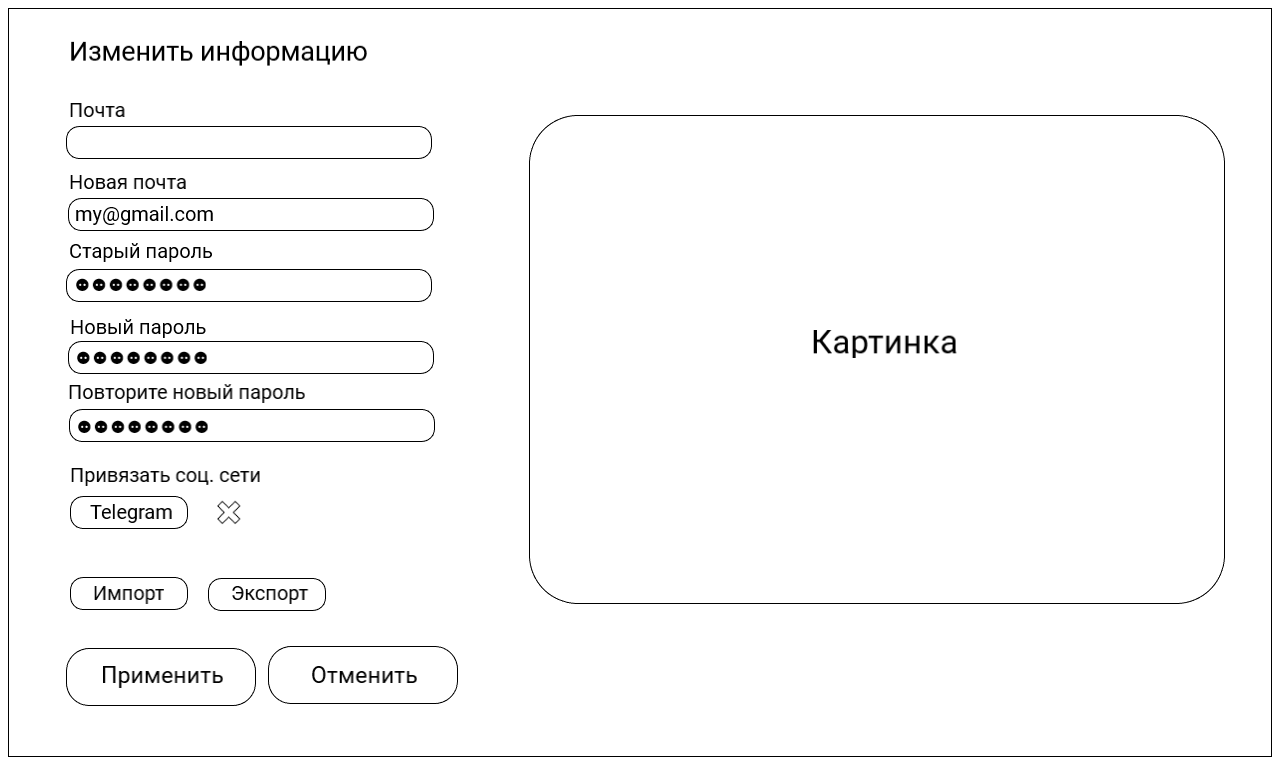


Рисунок 3.7 — Структурна схема вікна налаштування

* + 1. Структурна схема вікна додавання активності (рис. 3.8) складається з двох частин:
       1. Шапка вікна. Містить такі елементи:
* Назва вікна;
* Кнопка виходу.
  + - 1. Головна панель. Містить такі елементи:
* Поле для введення назви активності;
* Випадаючий список для вибору існуючої або введення нової категорії;
* Поле для введення часу, витраченого на активністі;
* Випадаючий список для вибору дати;
* Поле для введення коментаря;
* Кнопка додавання активності;
* Кнопка відміни створення нової активності.

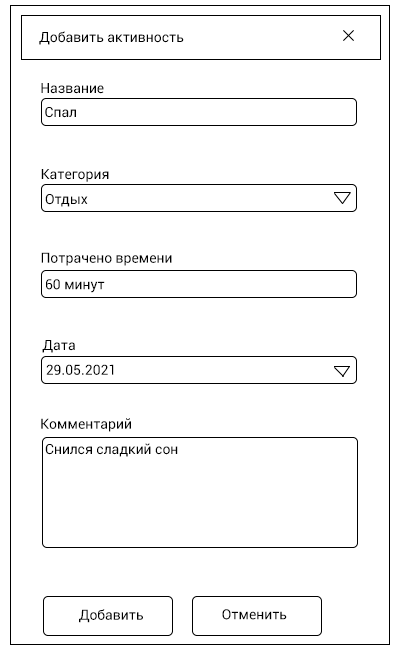


Рисунок 3.8 — Структурна схема вікна додавання активності

* + 1. Структурна схема вікна редагування активності (рис. 3.9) складається з двох частин:
       1. Шапка вікна. Містить такі елементи:
* Назва вікна;
* Кнопка виходу.
  + - 1. Головна панель. Містить такі елементи:
* Поле для введення назви активності;
* Випадаючий список для вибору існуючої або введення нової категорії;
* Поле для введення часу, витраченого на активністі;
* Випадаючий список для вибору дати;
* Поле для введення коментаря;
* Кнопка додавання активності;
* Кнопка відміни створення нової активності.

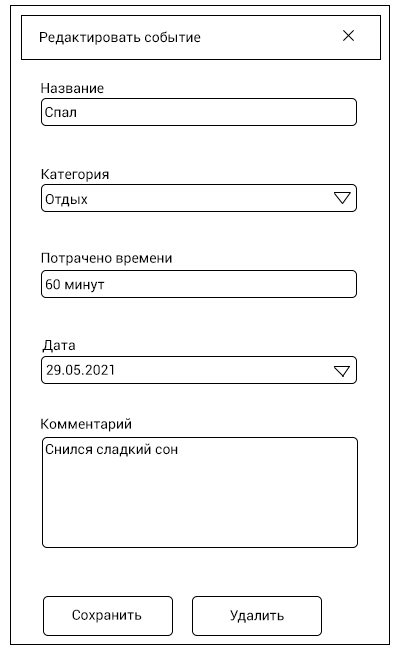


Рисунок 3.9 — Структурна схема вікна редагування активності

* + 1. Структурна схема вікна про програму (рис. 3.9) складається з однієї частини:
       1. Основна панель. Містить такі елементи:
* Назва вікна;
* Кнопка виходу;
* 8 картинок з фотографіями людей;
* 8 прізвищ та імен;
* 2 блоки з текстом про команду проекту;
* Авторські права;
* Версія продукту.
  + 1. Схема сигналів і слотів.

Припустимо, у нас є невелике вікно застосунку, що має тільки будь-який текст і кнопку. Якщо натиснути на кнопку – вона «видасть» так званий сигнал про те, що відбулася подія – в даному випадку, ця подія матиме властивість «Кнопка була натиснута». Причому, цей сигнал буде надсилатися при кожному натисканні на цю кнопку.

Розглянемо особливості слотів. Сам слот нагадує звичайну функцію (def), яка і буде реагувати при натисканні на кнопку, якщо сигнал кнопки підключити до даного слоту (функції).

Розглянемо приклад.

Є наступний код (рис. 3.10.), де ми маємо кнопку - qbtn; лейбл - self.lbl, який при запуску має текст «Lorem ipsum dolor»; і, власне, сам коннект сигналу до слоту - qbtn.clicked.connect(self.changeText).

При запуску програми, відкриється віконце з нашим текстом і кнопкою. При натисканні на кнопку, вона «видасть» той самий сигнал про те, що вона була натиснута. Рядок «qbtn.clicked.connect (changeText)» якраз таки пов'язує сигнал зі слотом – якщо подивитися на слот (функцію) textChanged, то можна помітити, що вона просто змінює текст у лейбла на «Yo, it works!».

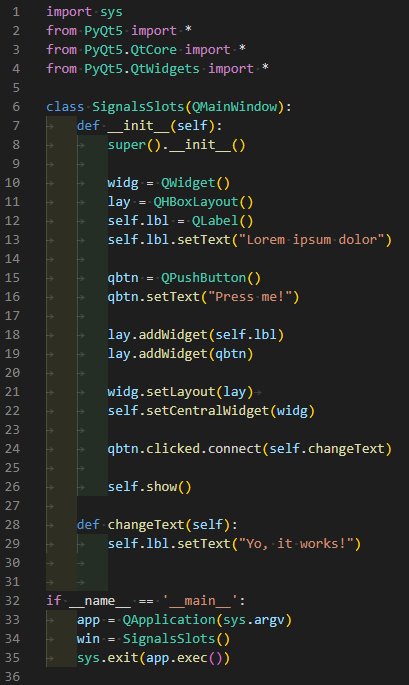
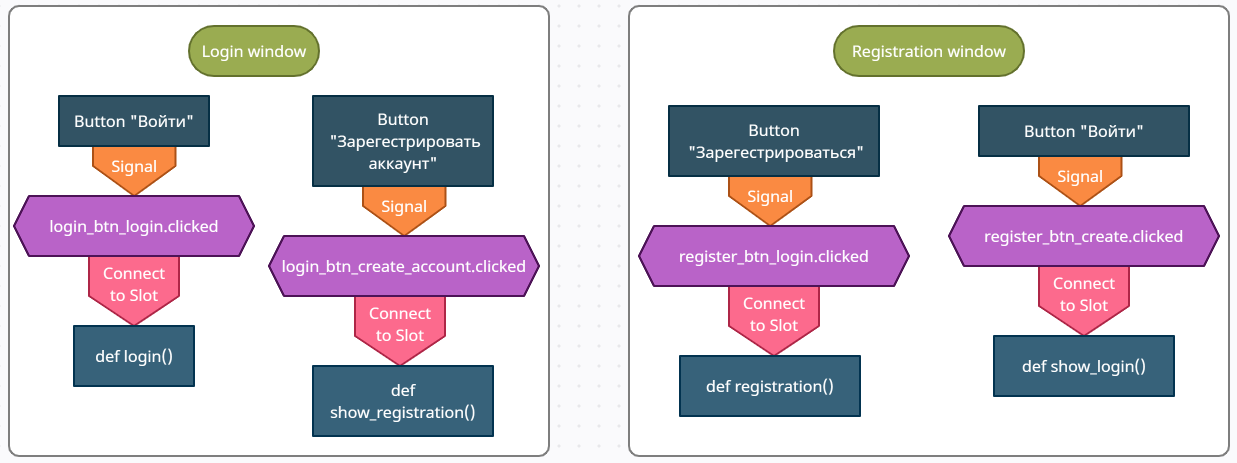


Рисунок 3.10 — Скріншот лістингу програми

До натискання кнопки бачимо віконечко, як на рис. 3.11, а після натискання кнопки бачимо віконечко, як на рис. 3.12:

|  |  |
| --- | --- |
| Рисунок 3.11 — Скріншот  до натискання кнопки «Press me» | Рисунок 3.12 — Скріншот  після натискання кнопки «Press me» |

У рисунках 3.13 - 3.18 будуть наведені схеми сигналів і слотів різних елементів управління з більшості вікон програми.  
  


|  |  |
| --- | --- |
| Рисунок 3.13 — Схема сигналів двох кнопок та слотів до них у вікні авторизування | Рисунок 3.14 — Схема сигналів двох кнопок та слотів до них у вікні реєстрація |

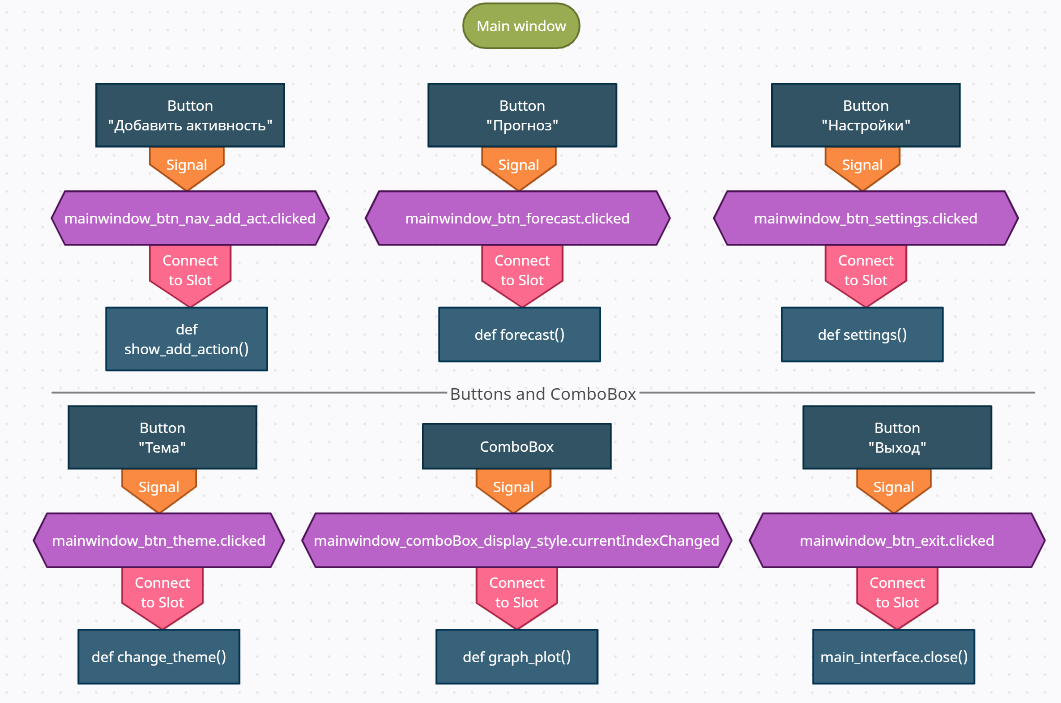


Рисунок 3.15 — Схема всіх елементів керування головного вікна

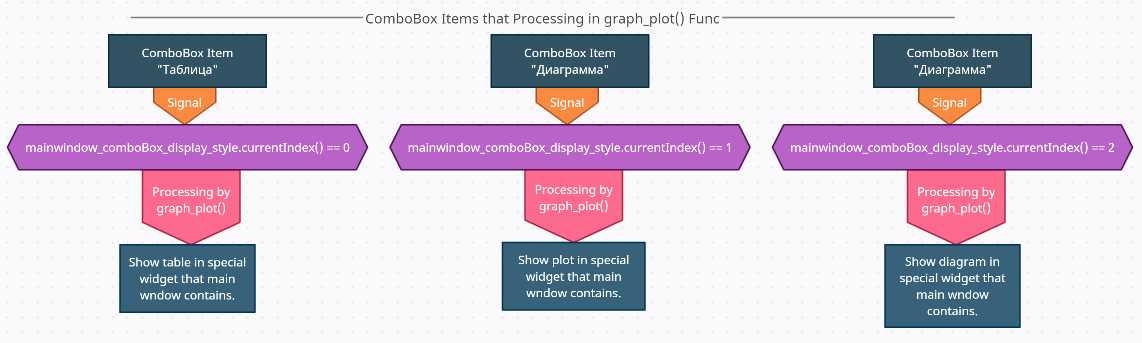
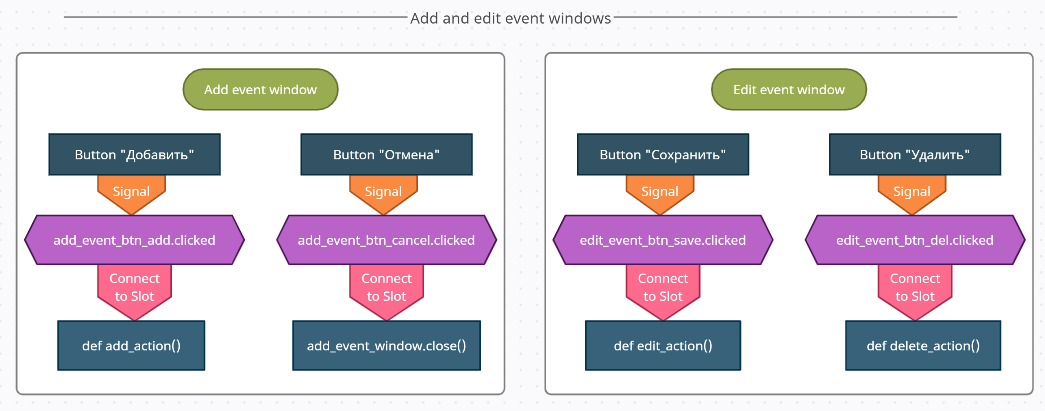


Рисунок 3.16 — Схема сигналів та слотів до елементу керування “ComboBox” головного вікна



|  |  |
| --- | --- |
| Рисунок 3.17 — Схема сигналів двох кнопок та слотів до них у вікні додавання активності | Рисунок 3.18 — Схема сигналів двох кнопок та слотів до них у вікні редагування активності |

Всі інші елементи управління, які були згадані в скріншотах вище, мають аналогічні властивості для сигналів і слотів.

## **Документування**

Перш за все, програмна документація — сукупність документів, що містять відомості, необхідні для розробки, виготовлення, супроводу та експлуатації програм[23]. Програмна документація є одним з видів технічної документації. (рис. 3.19.)

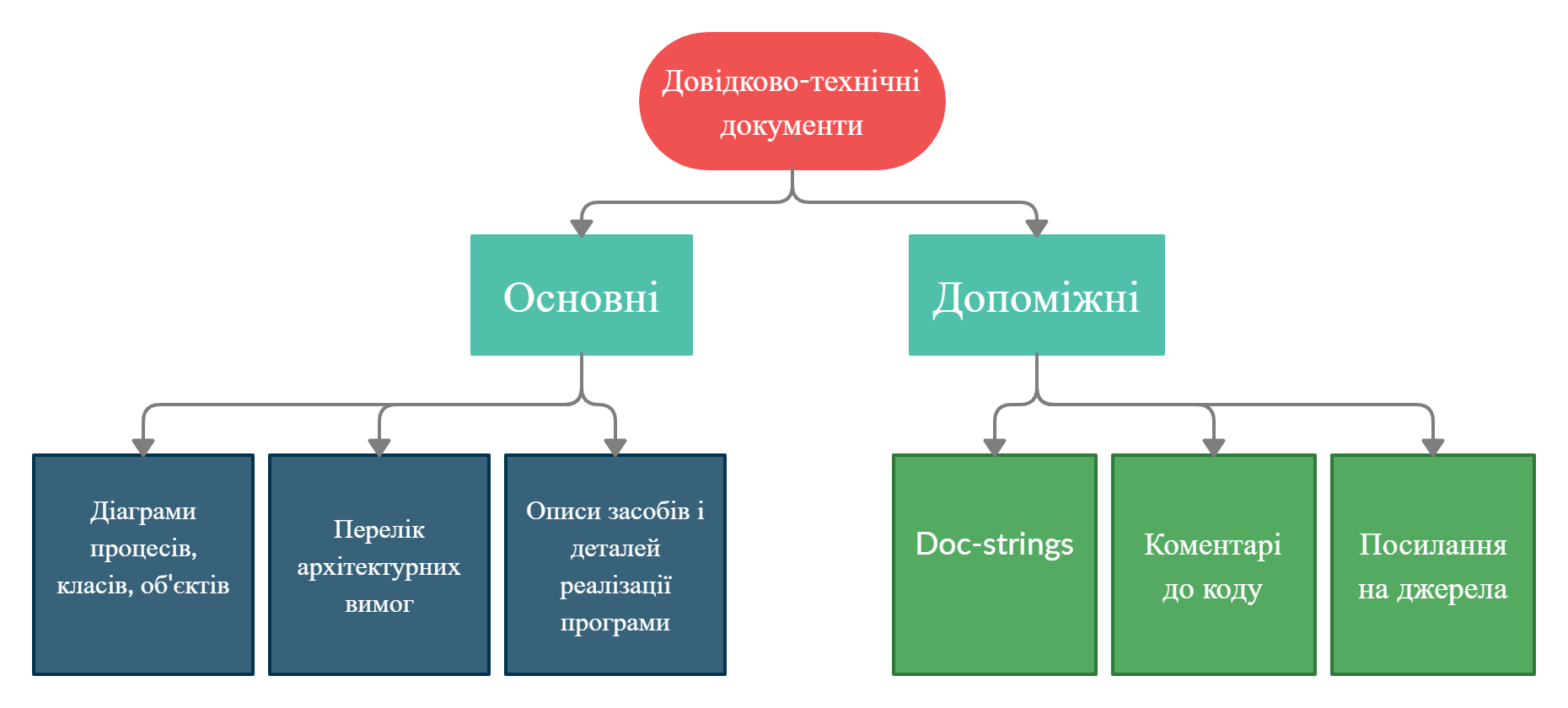


Рисунок 3.19 — Діаграма структури документування

## **Адміністрування**

Адміністрування - сукупність послуг з налаштування конфігурації та обслуговування програмних комплексів для забезпечення надійної безперебійної роботи. Також адміністрування включає управління доступом до програмного забезпечення та інші аспекти забезпечення безпеки, вирішення проблем, пов'язаних з сумісністю різних застосунків.

* + 1. Інструкції щодо інсталяції застосунку TimeSoft на Windows 10.

1. Встановіть Python.
   1. Перейдіть за посиланням на офіційний сайт мови Python <https://www.python.org/downloads/>
   2. Виберіть останню версію Python для вашої версії Windows та завантажте її.
   3. Інсталюйте завантажений файл та не забудьте поставити галочку в пункті PATH (рис. 3.20).

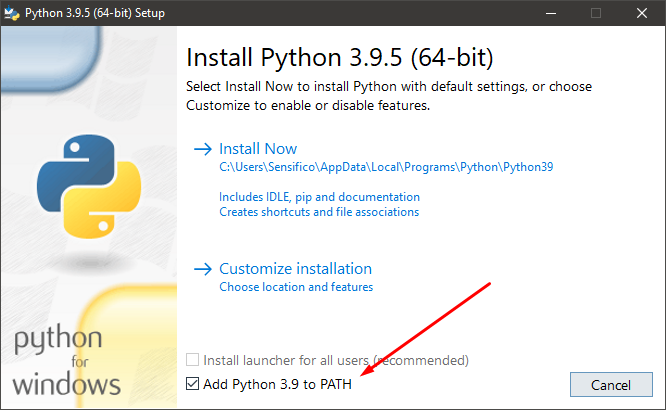


Рисунок 3.20 — Інсталювання Python

1. Встановіть необхідні пакети.
   1. Натисніть на клавіатурі одночасно клавіші WIN та R для запуску консолі рисунок 3.21.

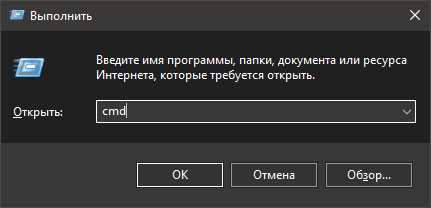


Рисунок 3.21 — Запуск консолі

* 1. Введіть в консоль команду (рис. 3.22):

*pip install psycopg2 PyQt5 PyQtChart pyqtgraph pandas matplotlib statsmodels*

та натисніть клавішу Enter для встановлення всіх необхідних пакетів.

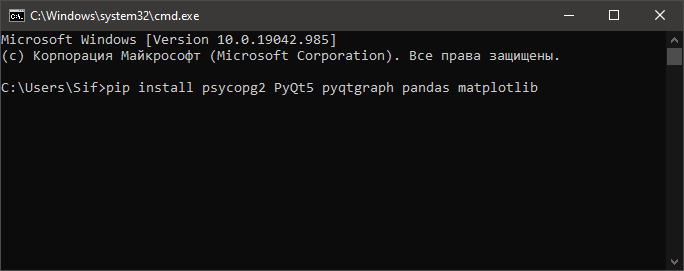


Рисунок 3.22 — Встановлення всіх необхідних пакетів

1. Запустіть застосунок.
   1. Перейдіть за посиланням <https://timesoft.pp.ua/> на сайт, та завантажте застосунок.
   2. Тепер можете сміливо запускати завантажений файл та користуватись цим чудовим застосунком.
      1. Якщо щось не працює, для зв'язку з розробниками застосунку TimeSoft, можно зв'язатися через електронну скриньку info@timesoft.pp.ua.
      2. Для використання застосунку потрібно:
2. Мінімум 50 МБ вільного місця;
3. Операційна система яка підтримує Python;
4. Python;
5. Пакети psycopg2, PyQt5, PyQtChart, pyqtgraph, pandas, matplotlib, statsmodels;
6. Доступ до інтернету.

## **Тестування**

Тестування є одним з найбільш усталених способів забезпечення якості розробки програмного забезпечення і входить в набір ефективних засобів сучасної системи забезпечення якості програмного продукту.

З технічної точки зору тестування полягає у виконанні програми на деякій множині вихідних даних і звірці одержуваних результатів із заздалегідь відомими (еталонними) з метою встановити відповідність різних властивостей і характеристик програми замовленим властивостями.

Ми провели тестування типів даних, які може ввести користувач через модуль unittest (рис. 3.23) та не виявили жодних помилок [24].

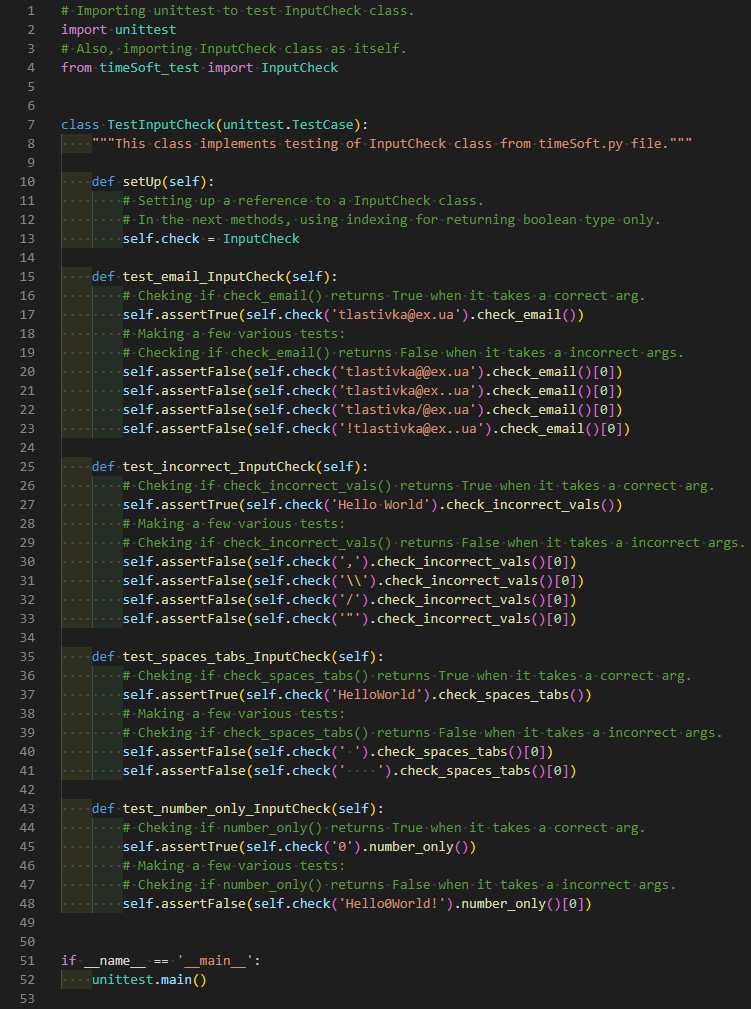


Рисунок 3.23 — Unit-тестування

## **Висновки по розділу**

У даному проекті була реалізована логіка програми, зв’язок з базою даних, яка знаходиться повністю в хмарі сервіс-платформи «Heroku», графічний інтерфейс користувача та телеграм бот. Программа дозволяє вносити/редагувати/видаляти та класифікувати зареєстрованому користувачу дані про свої активності, відображає їх у вигляді таблиці, графіку та діаграми, а також прогнозує скільки часу користувач витратить на певні активності в наступний проміжок часу (наприклад за наступний тиждень).

Застосунок представлений у вигляді «гнучкої» системи з набору залежних та незалежних модулів та бібліотек, що дає можливість вдосконалювати та розширювати функціонал програми у майбутньому.

# ВИСНОВКИ

В проекті «TimeSoft» були виконано і дотримано усіх вимог до програмного забезпечення, а саме: графічний інтерфейс користувача; часткове покриття unit-тестами; вихідний код програми, що відповідає стандарту оформлення коду; присутність відповідних коментарів або doc-strings до програмних елементів; розміщення проекту на GitHub; проект розгорнутий на локальному комп’ютері. Здійснено ретельний аналіз предметної області та чітко визначено задачу, реалізовано структуру та архітектуру з ідентифікуванням програмних та апаратних компонентів, створено схему розподілу компонентів, реалізовано графічний інтерфейс користувача та зберігання даних в хмарному сервісі.

Застосунок реалізовано мовою Python, представлено у вигляді «гнучкої» системи з набору залежних та незалежних модулів та бібліотек. Він дозволяє вносити, редагувати, видаляти та класифікувати зареєстрованому користувачу дані про свої активності, відображає їх у вигляді таблиці, графіку та діаграми, а також прогнозує, скільки часу користувач витратить на певні активності в наступний проміжок часу. Дана функціонуюча система «обгорнута» у графічний інтерфейс користувача, що має при собі свій унікальний, фірмовий стиль.

# СПИСОК ПОСИЛАНЬ

1. Болотова А.К. Психология организации времени: Учебное пособие / А.К. Болотова. М.: Аспект Пресс, 2012. 254 с.
2. Телегина Т.В. Тайм-менеджмент руководителей / Т.В.Телегина. М.: МФПУ «Синергия», 2013. 86 с.
3. Мыздрикова А.Ю. Тайм менеджмент как новое направление в современном управлении / А.Ю.Мыздрикова, О.В.Расторгуева, Е.А.Панявина // Современные наукоемкие технологии. 2014. № 7. С. 49-51.
4. Зайверт Л. Если спешишь – не торопись: новый тайм-менеджмент в ускорившемся мире: семь шагов к эффективности и независимости в использовании времени / Л.Й.Зайверт; перевод с немецкого Н.С.Сироткина. М.: Интерэксперт, 2011. 255 с.
5. Архангельский Г.А. Формула времени. Тайм-менеджмент на Outlook / Г.А. Архангельский. М.: Манн, Иванов и Фербер, 2014. 272 с.
6. Травин В.В. Менеджмент персонала предприятия / В.В.Травин, В.В.Дятлов. М.: Дело, 2012. 254 с.
7. Сидорова Н.А. Тайм-менеджмент / Н.А.Сидорова, Е.Б.Анисинкова. M.: Дашков и К, 2012. 220 с.
8. Новак Б.В. Тайм-менеджмент на компьютере. Как управлять своим временем эффективно. – СПб.: Питер, 2007. – 128 с.
9. Хитра О. В. Ефективний тайм-менеджмент як невід’ємний складник системи управління персоналом підприємства / О. В. Хитра // Науковий вісник Ужгородського національного університету. Серія : Міжнародні економічні відносини та світове господарство. - 2019. - Вип. 26(2). - С. 101-110. - Режим доступу: <http://nbuv.gov.ua/UJRN/Nvuumevcg_2019_26(2)__21>
10. “Введение в PostgreSQL с Python +Psycopg2” [Електронний ресурс] – <https://pythonru.com/biblioteki/vvedenie-v-postgresql-s-python-psycopg2>
11. “Ідентифікація” [Електронний ресурс] – [https://uk.wikipedia.org/wiki/Ідентифікація](https://uk.wikipedia.org/wiki/%D0%86%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D1%96%D0%BA%D0%B0%D1%86%D1%96%D1%8F)
12. “Create Unique Constraints” [Електронний ресурс] – <https://docs.microsoft.com/en-us/sql/relational-databases/tables/create-unique-constraints?view=sql-server-ver15>
13. “Create Primary Keys” [Електронний ресурс] – <https://docs.google.com/document/d/19Vvf8KtxNlUmrNH_jbhGGwfV0HncI9xOzKsTtBhejbY/edit#>
14. “SQL FOREIGN KEY Constraint” [Електронний ресурс] – <https://dev.mysql.com/doc/refman/5.6/en/create-table-foreign-keys.html>
15. “Create Foreign Key Relationships” [Електронний ресурс] – <https://docs.microsoft.com/en-us/sql/relational-databases/tables/create-foreign-key-relationships?view=sql-server-ver15>
16. “13.1.20.5 FOREIGN KEY Constraints” [Електронний ресурс] – <https://dev.mysql.com/doc/refman/8.0/en/create-table-foreign-keys.html>
17. “Глава 8. Типы данных” [Електронний ресурс] – <https://postgrespro.ru/docs/postgresql/9.6/datatype-numeric>
18. “SQL NOT NULL Constraint” [Електронний ресурс] – <https://www.w3schools.com/sql/sql_notnull.asp>
19. “NULL (SQL)” [Електронний ресурс] – <https://ru.wikipedia.org/wiki/NULL_(SQL)>
20. “int, bigint, smallint, and tinyint (Transact-SQL)” [Електронний ресурс] – <https://docs.microsoft.com/en-us/sql/t-sql/data-types/int-bigint-smallint-and-tinyint-transact-sql?view=sql-server-ver15>
21. “char and varchar (Transact-SQL)” [Електронний ресурс] – <https://docs.microsoft.com/en-us/sql/t-sql/data-types/char-and-varchar-transact-sql?view=sql-server-ver15>
22. “Date and Time Data Types and Functions (Transact-SQL)” [Електронний ресурс] – <https://docs.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql?view=sql-server-ver15>
23. “Програмна документація” [Електронний ресурс] – [https://uk.wikipedia.org/wiki/Програмна\_документація](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B0_%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D1%96%D1%8F)
24. “Тестирование в Python [unittest]. Часть 1. Введение” [Електронний ресурс] – <https://devpractice.ru/unit-testing-in-python-part-1/>

# ДОДАТОК A

## ЕКРАНИ ЗАСТОСУНКУ TIMESOFT

## 

Рисунок A.1 — Вікно авторизації (темна тема)

## 

Рисунок A.2 — Вікно реєстрації (темна тема)

## 

Рисунок A.3 — Головне вікно застосунку (темна тема)

## 

Рисунок A.4 — Вікно налаштувань (темна тема)

|  |  |
| --- | --- |
| Рисунок A.5 — Вікно додавання активності (темна тема) | Рисунок A.6 — Вікно редагування активності(темна тема) |

## 

Рисунок A.7 — Вікно про застосунок (темна тема)

## 

Рисунок A.8 — Вікно авторизації (світла тема)

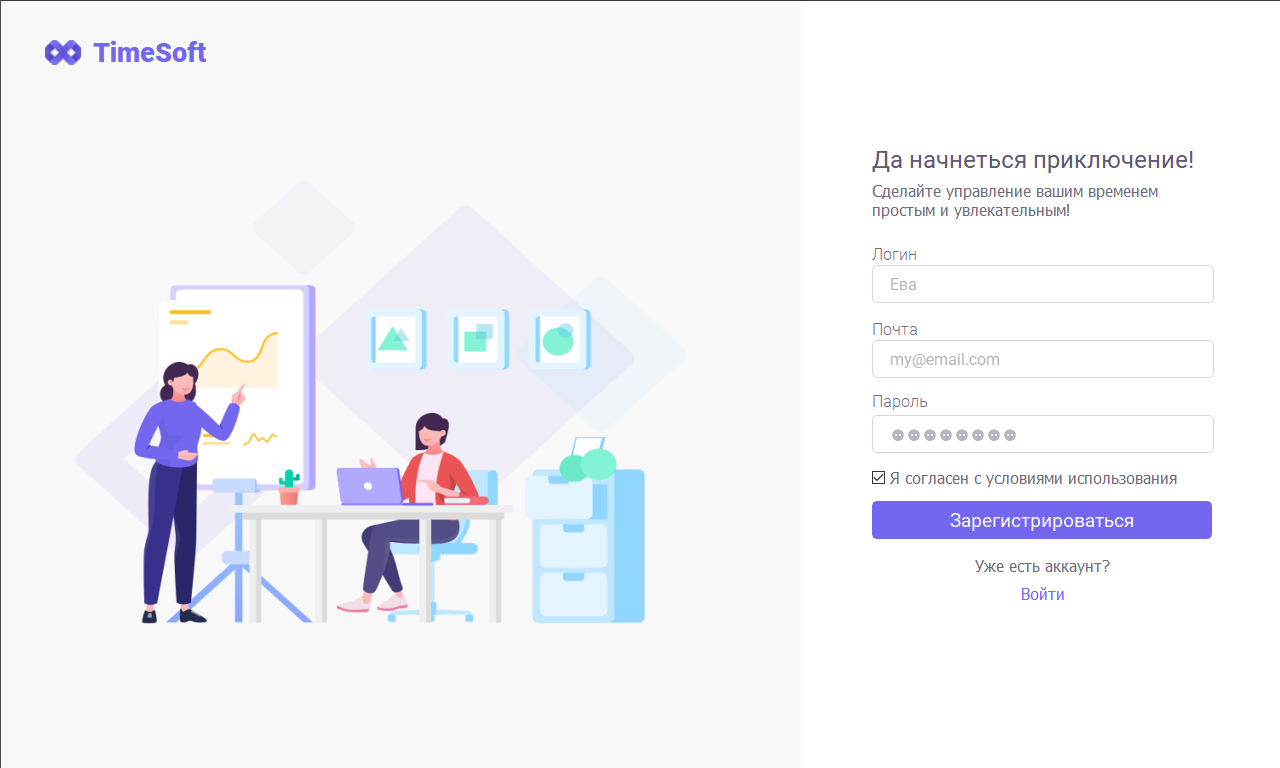


Рисунок A.9 — Вікно реєстрації (світла тема)

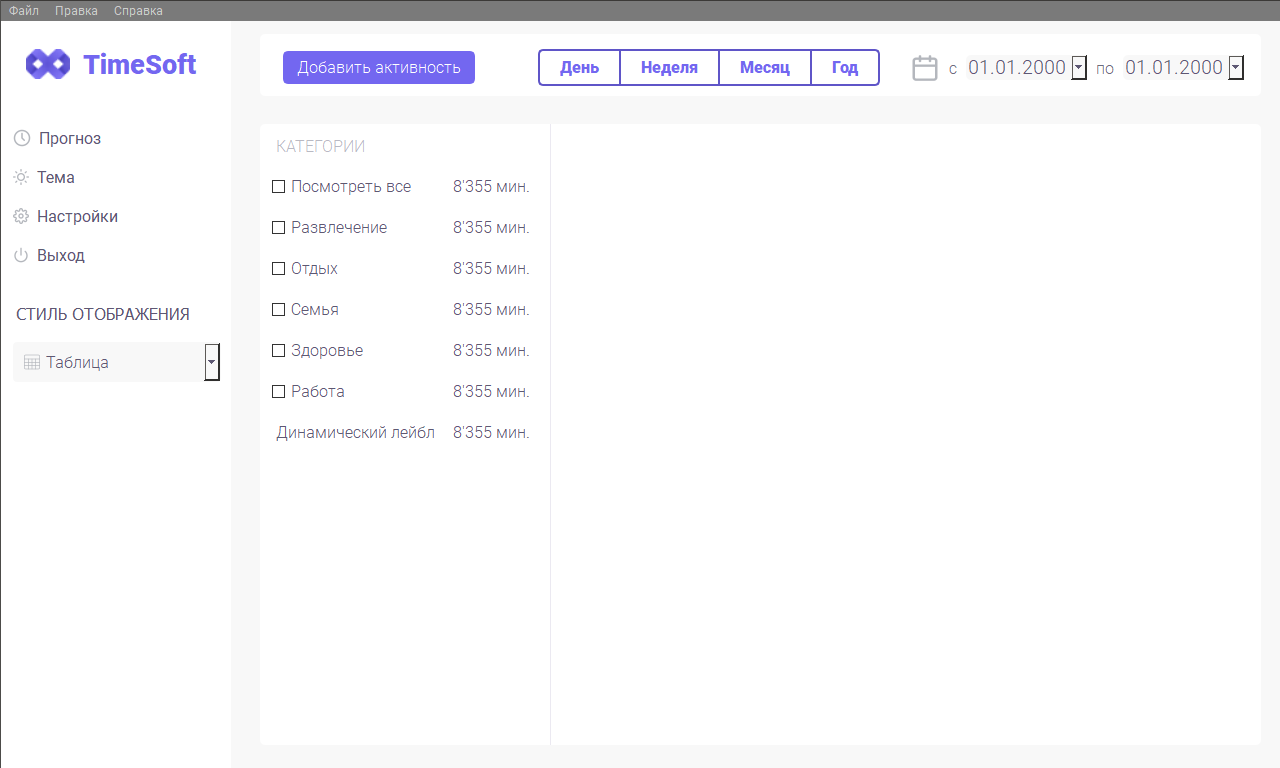


Рисунок A.10 — Головне вікно застосунку (світла тема)

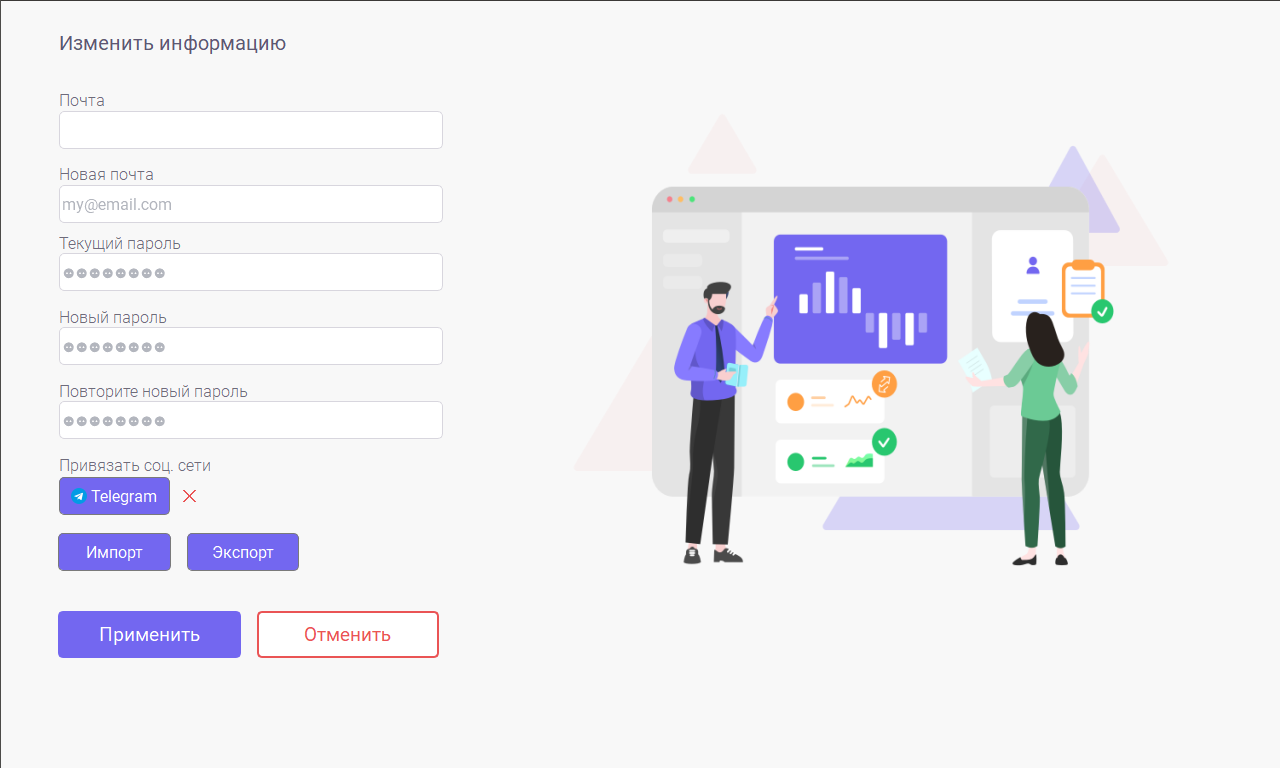


Рисунок A.11 — Вікно налаштувань (світла тема)

|  |  |
| --- | --- |
| Рисунок A.5 — Вікно додавання активності (світла тема) | Рисунок A.6 — Вікно редагування активності(світла тема) |



Рисунок A.7 — Вікно про застосунок (світла тема)

# ДОДАТОК Б

## ФРАГМЕНТИ ЛІСТИНГУ



Рисунок Б.1



Рисунок Б.2



Рисунок Б.3

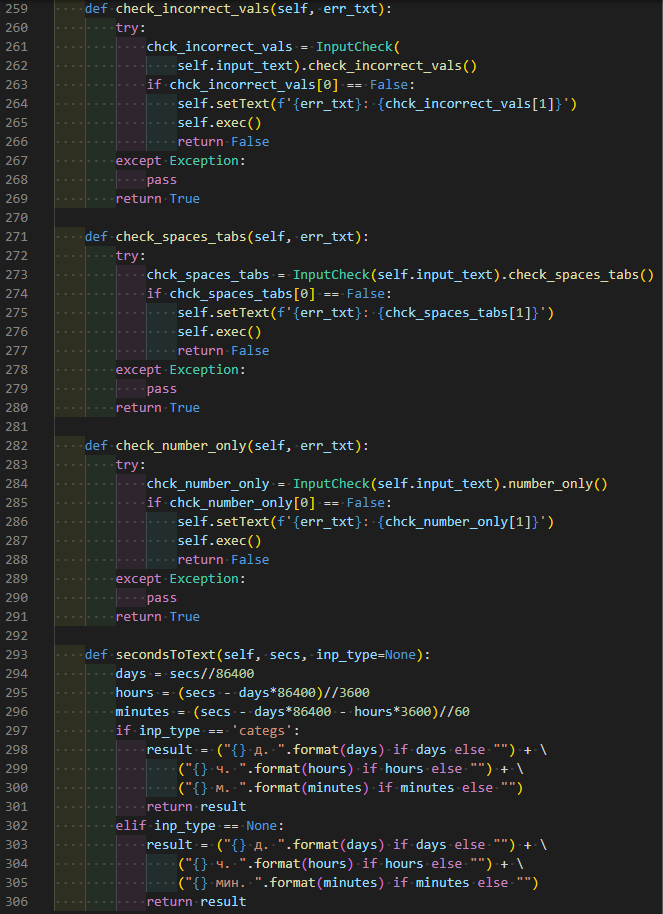


Рисунок Б.4

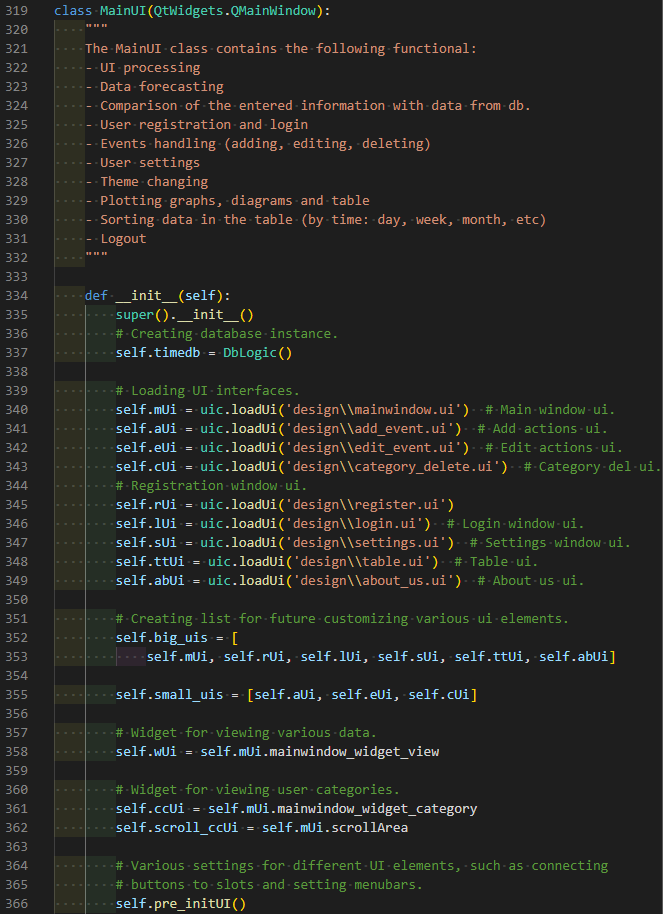


Рисунок Б.5

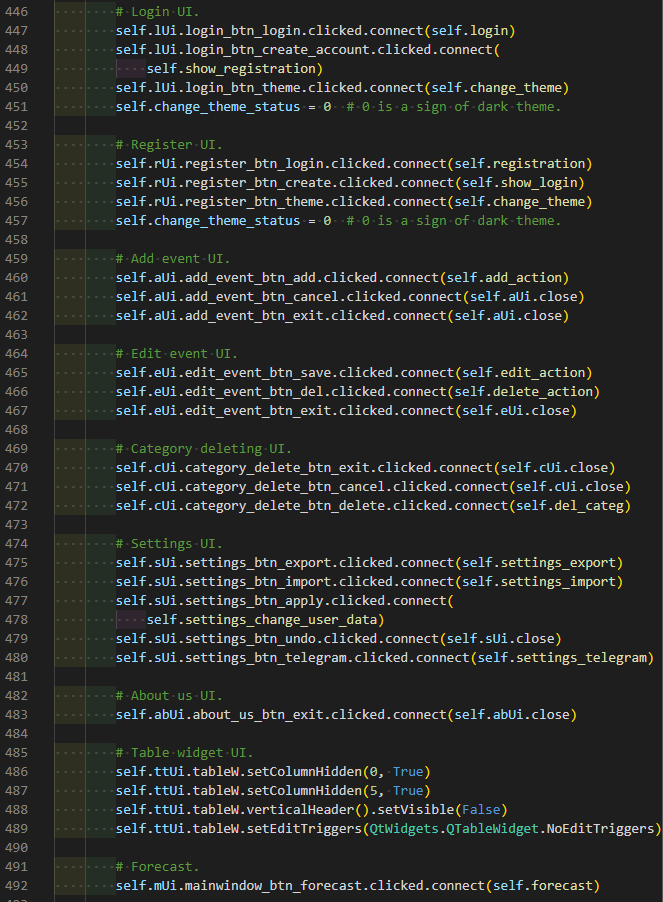


Рисунок Б.6

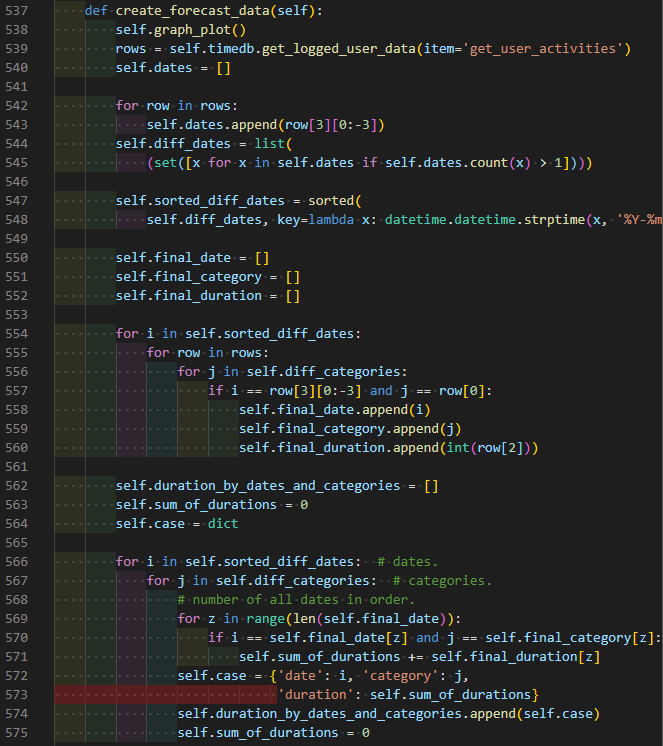


Рисунок Б.7

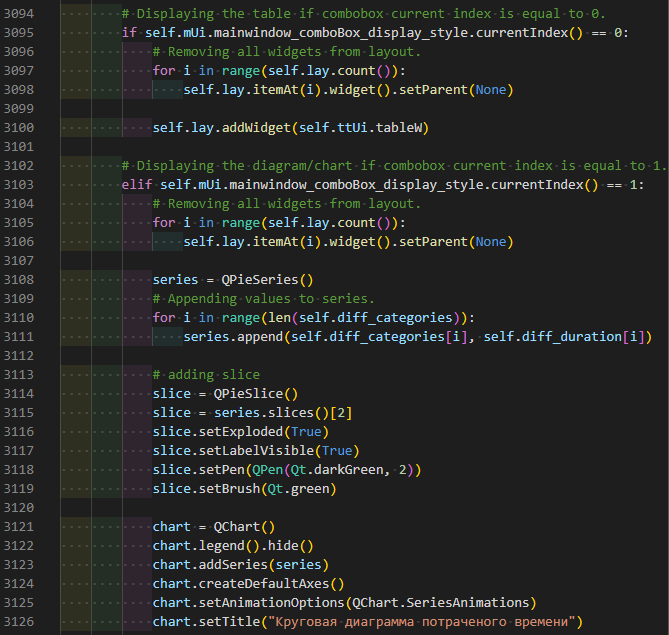


Рисунок Б.8



Рисунок Б.9

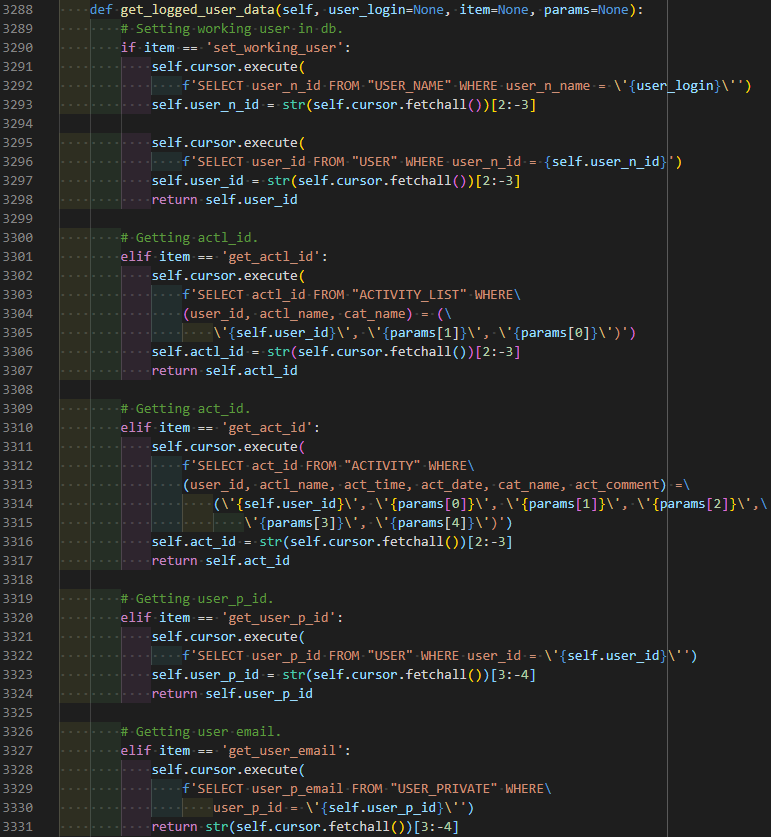


Рисунок Б.10

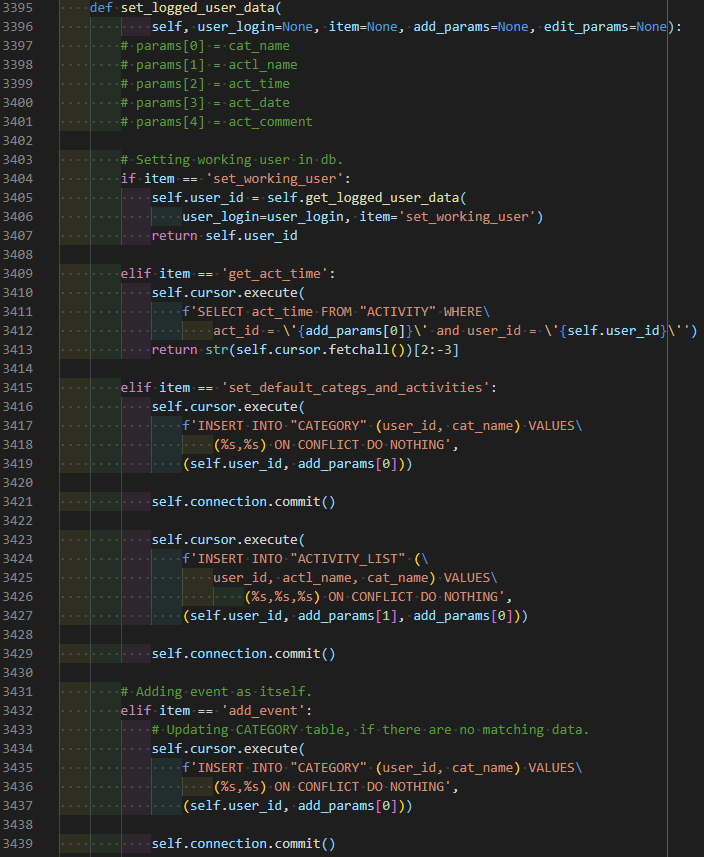


Рисунок Б.11