РОЗДІЛ 1.
ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1. Загальна характеристика об’єкта дослідження

Моніторинг є системою, яка виконує постійне спостереження за явищами і процесами, що відбуваються в навколишньому середовищі і суспільстві, результати якого служать для керування прийняття наступного рішення на основі отриманих раніше даних.

Системи моніторингу можуть включати в себе [2]:

- системи, які виконують збір, спостереження, обробку, зберігання,
- передачу та аналіз даних про оточуюче середовище;
- системи для передбачення розвитку небезпечних ситуацій на основі зібраних раніше даних;

На сьогоднішній день більшість компаній світу використовують системи моніторингу робочого часу працівників, адже це найефективніший спосіб оцінки виконаної роботи співробітників. За допомогою системи моніторингу робочого часу працівників, можливо виявляти та вирішувати проблеми компанії ще до того, як ці проблеми стануть глобальними та приведуть до негативних наслідків.

Система моніторингу робочого часу робітників запобігає виникненню багатьох проблем. Це особливо важливо, якщо працівники отримують заробітну плату в залежності від кількості напрацьованих годин. Також, в залежності від функціоналу, система моніторингу робочого часу працівників також може запобігати конфлікти між колегами в командах, пов’язані із приналежністю виконаної роботи до конкретного працівника, адже системи моніторингу

робочого часу працівників зазвичай включають в себе не тільки функцію стеження за кількістю відпрацьованих годин.

Моніторинг – система, за допомогою якої здійснюється постійне спостереження та оцінка змін стану будь-якого технічного, соціального, природного та інших об’єктів. В нашому випадку моніторинг (також можна використовувати слово «відстеження» для опису даної проблеми) застосовується задля визначення конкретної кількості величини – часу. Система моніторингу робочого часу працівників призначена для відстеження часу, під час якого працівник виконував свою роботу [15].

Але відстежувати тільки час буде не ефективним вирішенням проблеми, якщо завдання працівників будуть складними та потребуватимуть більше одного дня на виконання. Тому, окрім відстеження часу роботи, в системи моніторингу робочого часу працівників зазвичай додають й інші функції. Наприклад, хорошою практикою є додання в систему моніторингу робочого часу працівників функцію знімку екрану, яка працює через певну кількість часу. Можливо також додати в систему моніторингу робочого часу працівників функцію, яка відстежує кількість натиснутих клавіш у хвилину, або кількість кліків мишкою та інше.

Задля ефективного спостереження за своїми працівниками, керівникам (або іншим відповідальним за діяльність своїх підлеглих людям) необхідно отримати ряд питань, що є дуже складним завданням. По-перше, потрібно дізнатися, коли саме працівники починають та закінчують роботу. Також потрібно точно знати скільки фактично годин триває робочий день персоналу. Важливо враховувати й те, скільки часу працівники витрачають на виконання поставлених їм задач, а скільки – в особистих цілях. Потрібно знати чим займаються працівники протягом робочого дня та наскільки ефективно працівники використовують робочий час. На превеликий жаль, у більшості випадків пошук відповідних рішень для моніторингу робочого часу працівників перетворюється у велику проблему. Саме тому і потрібні систему моніторингу робочого часу працівників, при правильній побудові, будуть здатні вирішити більшість вищевказаних проблем [17].

Наразі, тайм-менеджмент - це процес планування і усвідомленого контролю над часом, що витрачається на певні види діяльності, особливо для підвищення ефективності, результативності та продуктивності. Він включає в себе акт поєднання різних вимог до людини, пов'язаних з роботою, громадським життям, родиною, хобі, особистими інтересами і зобов'язаннями, з обмеженням часу.

Ефективне використання часу дає людині «вибір» витрачати або керувати діяльністю у зручний для нього час і в зручний для нього час. Управління часом може допомогти ряд навичок, інструментів і методів, використовуваних для управління часом при виконанні певних завдань, проектів і цілей з дотриманням встановленого терміну. Спочатку управління часом відносилося тільки до ділової або робочої діяльності, але з часом цей термін розширили і включили також і особисту діяльність. Система тайм-менеджменту - це продумана комбінація процесів, інструментів, технік і методів.

Тайм-менеджмент зазвичай необхідний в будь-якому управлінні проектом, так як він визначає час і обсяг завершення проекту. Також важливо розуміти, що як технічні, так і структурні відмінності в управлінні часом існують через різницю в культурних концепціях часу. Основні теми, що випливають з літератури по тайм-менеджменту, включають наступне [22]:

Створення середовища, що сприяє ефективності (з точки зору рентабельності, якості результатів і часу для виконання завдань або проекту),

Розстановка пріоритетів.

Пов'язаний з цим процес скорочення часу, що витрачається на непріоритетна завдання,

Реалізація цілей.

Тайм-менеджмент пов'язаний з наступними концепціями.

- Управління проектами: управління часом можна розглядати як управління підмножиною проекту і більш широко відомий як планування проекту та планування проекту. Тайм-менеджмент також був визначений як одна з основних функцій, визначених в управлінні проектами.

- Управління увагою ставиться до управління когнітивними ресурсами і, зокрема, часом, яке люди виділяють своїм розумом (і організовують уми своїх співробітників) для виконання деяких дій.
- Блокування часу - це стратегія управління часом, яка спеціально підтримує виділення відрізків часу для конкретних завдань, щоб сприяти більш глибокої зосередженості і продуктивності.

Організаційне управління часом - це наука виявлення, оцінки та скорочення втрат часу в організаціях. Він виявляє, повідомляє і фінансово оцінює стійке час, витрачений даремно час і ефективне час в організації, а також розробляє економічне обґрунтування для перетворення витраченого марно часу в продуктивний час за рахунок фінансування продуктів, послуг, проектів або ініціатив в якості позитивного повернення інвестицій.

У деяких літературних джерелах по тайм-менеджменту підкреслюються завдання, пов’язані зі створенням середовища, що сприяє «реальної» ефективності. Ці стратегії включають такі принципи, як [12]:

- «Організуватися» - сортування документів та завдань,
- «Захист свого часу» ізоляцією, ізоляцією і делегуванням повноважень,
- «Досягнення через управління метою і через фокусування на цілі» - мотиваційний акцент,
- «Відновлення від шкідливих звичок у часі» - відновлення від основних психологічних проблем, наприклад, прокрастинації.

Стратегії тайм-менеджменту часто асоціюються з рекомендацією ставити особисті цілі. У літературі підкреслюються такі теми, як:

- «Робота в пріоритетному порядку» - ставте цілі і розставляйте пріоритети,
- «Встановіть гравітаційні мети» - які автоматично привертають дії.

Ці цілі записуються і можуть бути розбиті на проект, план дій або простий список завдань. Для окремих завдань або цілей може бути встановлений рейтинг важливості, можуть бути встановлені терміни і призначені пріоритети. Результатом цього процесу є план зі списком завдань, розкладом або календарем

дій. Автори можуть рекомендувати щоденні, щотижневі, щомісячні або інші періоди планування, пов’язані з різним обсягом планування або аналізу. Це робиться різними способами, а сааме [16]:

1. Аналіз ABCD.

Техніка, яка давно використовується в управлінні бізнесом, - це категоризація великих даних по групам. Ці групи часто позначаються буквами A, B, C і D - звідси і назва. Дії оцінюються за такими загальними критеріями:

- A - Завдання, які сприймаються як термінові і важливі,
- B - Завдання, які важливі, але не термінові,
- C - Завдання, які не важливі, але термінові,
- D - Завдання, які не важливі і не термінові.

Потім кожна група впорядковується по пріоритету. Щоб ще більше уточнити пріоритети, деякі люди потім примусово ранжируют всі елементи «B» як «A» або «C». ABC-аналіз може включати більше трьох груп.

Багато компаній використовують програмне забезпечення для обліку робочого часу, щоб відслідковувати робочий час, оплачувані годинни. Наприклад, програмне забезпечення для управління юридичною практикою.

Варто відмітити, що багато програмних продуктів для управління часом підтримують декількох користувачів. Вони дозволяють людині давати завдання іншим користувачам і використовувати програмне забезпечення для спілкування. Додатки списку завдань можна розглядати як легке програмне забезпечення для управління персональною інформацією або управління проектами.

Сучасні програми зі списком завдань можуть мати вбудовану ієрархію завдань (завдання складаються з підзадач, які знову можуть містити підзадачі), можуть підтримувати декілька методів фільтрації і впорядкування списку завдань і можуть дозволяти пов’язувати довільно довгі замітки для кожна задача.

На відміну від концепції, що дозволяє людині використовувати кілька методів фільтрації, по крайней мере, один програмний продукт додатково містить

режим, в якому програмне забезпечення буде намагатися динамічно визначати кращі завдання в будь-який даний момент.

Таким чином, системи управління часом часто включають години або веб-додаток, що використовується для відстеження робочого часу співробітника. Системи тайм-менеджменту дають роботодавцям подання про свою робочу силу, дозволяючи їм бачити, планувати і управляти своїм часом співробітників. Це дозволяє роботодавцям управляти витратами на робочу силу і підвищувати продуктивність. Система тайм-менеджменту автоматизує процеси, позбавляючи від паперової роботи і виснажливих завдань.

1.2. Сучасний стан програмого забезпечення ефективного тайм-менеджменту

Програмне забезпечення для обліку робочого часу - це категорія комп'ютерного програмного забезпечення, яке дозволяє своїм співробітникам записувати час, витрачений на завдання або проекти. Програмне забезпечення використовується в багатьох галузях, в тому числі в тих, де працюють фрілансери і погодинна оплата. Він також використовується професіоналами, які виставляють своїм клієнтам погодинні рахунки. До них відносяться юристи, фрілансери і бухгалтерии [20].

Програмний інструмент для обліку робочого часу може використовуватися автономно або інтегруватися з іншими додатками, такими як програмне забезпечення для управління проектами, підтримка клієнтів та бухгалтерський облік. Програмне забезпечення для обліку робочого часу - це електронна версія традиційного паперового табеля робочого часу.

Крім програмного забезпечення для розкладу, програмне забезпечення для відстеження часу також включає програмне забезпечення для запису часу, яке використовує моніторинг активності користувача (UAM) для запису дій, виконуваних на комп'ютері, і часу, витраченого на кожну задачу.

Типи програм для обліку робочого часу [22]:

- Розклад.

Дозволяє користувачам вручну вводити час, витрачений на завдання.

- Облік часу / запис.

Автоматично записує дії, що виконуються на комп'ютері.

Програмне забезпечення для обліку робочого часу може бути [25]:

1. Автономним: використовується тільки для запису розкладів і створення звітів.

2. Інтегрованим в складі:

- Системи бухгалтерського обліку, наприклад, дані розкладу, що надходять безпосередньо в рахунку компанії.
- Білінгові системи, наприклад, для виставлення рахунків, особливо для підрядників, юристів.
- Системи управління проектами, наприклад, дані розкладу, використовувані програмним забезпеченням для управління проектами для візуалізації зусиль, що витрачаються на проекти або завдання.
- Системи нарахування заробітної плати, наприклад, для оплати співробітників на основі відпрацьованого часу.

Планування ресурсів, наприклад, двунаправленная інтеграція, дозволяє планувальникам розподіляти персонал за завданнями, які після завершення можуть бути підтверджені і перетворені в розкладу.

Програмне забезпечення для розкладу - це програмне забезпечення, що використовується для ведення розкладу. Він був популяризував, коли комп'ютери вперше були введені в офісне середовище з метою автоматизації важкої паперової роботи в великих організаціях. Програма розкладу дозволяє вводити час, витрачений на виконання різних завдань.

При використанні в компаніях співробітники вводять час, витрачений на виконання завдань, в електронні табелі обліку робочого часу. Ці розкладу потім можуть бути затверджені або відхилені керівниками або керівниками проектів.

З 2006 року програмне забезпечення для розкладу переміщається на мобільні платформи (смартфони, планшети, розумні годинник), Що дозволяє краще відстежувати співробітників, робота яких пов’язана з декількома офісами.

Програмне забезпечення для відстеження / запису часу автоматизує процес обліку робочого часу, записуючи дії, що виконуються на комп’ютері, і час, витрачений на кожне з них. Це програмне забезпечення покликане стати поліпшенням у порівнянні з програмним забезпеченням для розкладу. Його мета - дати загальну картину використання комп’ютера. Програмне забезпечення для автоматичного відстеження / запису часу записує і показує використання додатків, документів, ігор, веб-сайтів [28].

При використанні в компаніях це програмне забезпечення дозволяє контролювати продуктивність співробітників, записуючи завдання, які вони виконують на своїх комп’ютерах. Його можна використовувати для заповнення розкладів.

При використанні фрілансерами це програмне забезпечення допомагає створювати звіти для клієнтів (наприклад, табелі обліку робочого часу та рахунки-фактури) або підтверджувати виконану роботу.

Методи обліку часу.

Є кілька способів, якими компанії відстежують час співробітників за допомогою програмного забезпечення для обліку робочого часу.

1. Тривалий.

Співробітники вводять тривалість завдання, але не час її виконання.

2. Хронологічний.

Співробітники вводять час початку і закінчення завдання.

3. Автоматичний.

Система автоматично розраховує час, витрачений на завдання або цілі проекти, з використанням підключеного пристрою або персонального комп’ютера, а також введення даних користувачем за допомогою кнопок запуску і зупинки.

Користувачі можуть отримувати зареєстровані завдання і переглядати тривалість або час початку і закінчення.

4. На основі винятків.

Система автоматично записує стандартні часи роботи, за винятком затвердженого вільного часу або LOA.

5. Час приходу і відходу.

Співробітники вручну записують час прибуття та вибуття.

моніторинг

6. Система фіксує активне і неробочий час співробітників. Він також може записувати знімки екрану.

7. На основі розташування.

Система визначає робочий статус співробітників в залежності від їх місцезнаходження.

8. Планування ресурсів: за рахунок попереднього планування ресурсів розкладу співробітників можна легко перетворити в табелі обліку робочого часу.

Переваги програмного забезпечення для обліку робочого часу [2]:

- Відстеження часу може підвищити продуктивність, оскільки компанії можуть відстежувати час, витрачений на завдання, і краще розуміти, які дії змушують співробітників витрачати час даремно.

- Програмне забезпечення для обліку робочого часу підвищує підзвітність, документуючи час, необхідний для виконання поставлених завдань. Дані збираються в базі даних і можуть використовуватися для аналізу даних відділами кадрів.

Можливості, пропоновані програмним забезпеченням для обліку робочого часу, включають:

- Автоматичне створення рахунків для клієнтів або замовників професіонала в залежності від витраченого часу.

- Відстеження перевитрати коштів за проектами з фіксованою вартістю.

- Пакети управління персоналом, які відстежують відвідуваність, відсутність співробітників, проблеми з кадрами, заробітну плату, управління талантами і аналітику праці.

Різні методи відстеження часу використовуються працівниками на робочому місці, наприклад, використання програми відстеження часу, паперу або електронної таблиці, перфокарт, біометрії або POS.

Ручне відстеження часу або відстеження часу за допомогою паперу та електронних таблиць має 50% шансів на крадіжку часу. Також існує ймовірність того, що працівники можуть не вводити час, витрачений на електронні листи, зустрічі.

Згідно з опитуванням, проведеним Гарвардським оглядом бізнесу , 40% співробітників ніколи не відстежували час, витрачений на читання чи написання електронних листів (Рис.1.1).

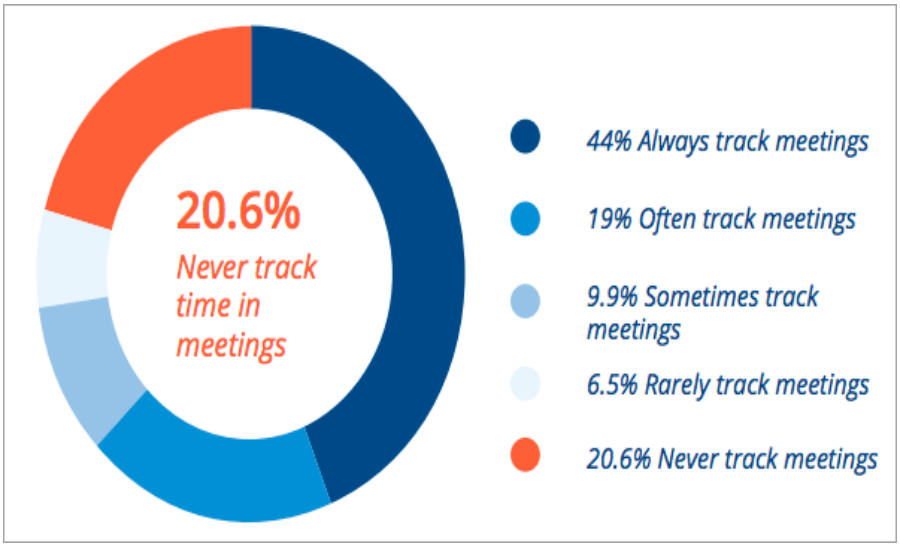


Рисунок 1.1 – Результати опитування респондетнів

На зображенні нижче показано процентне співвідношення точності з різними звичками заповнення робочого часу (Рис. 1.2).

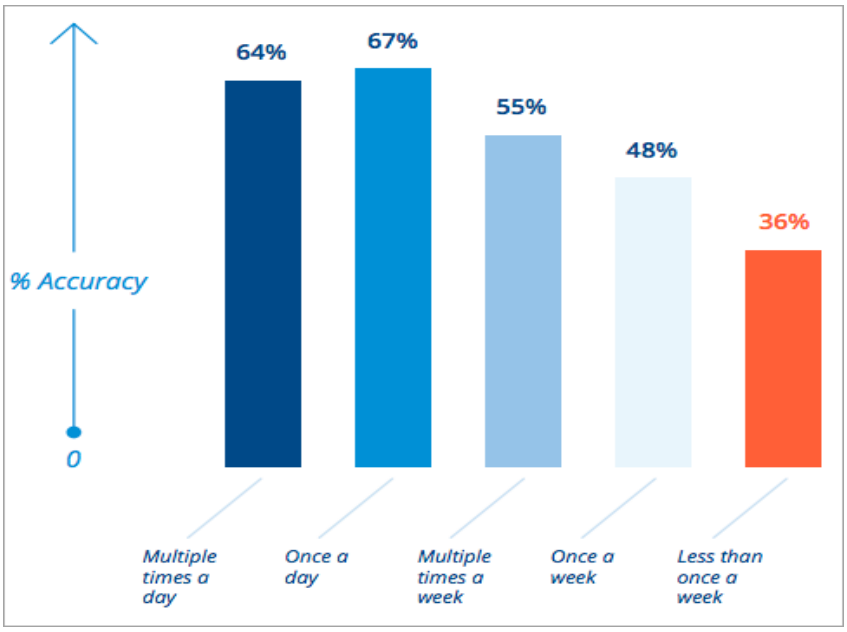


Рисунок 1.2. - Процентне співвідношення точності з різними звичками заповнення робочого часу

Щоб уникнути всіх цих неточностей, вам слід скористатися програмою табелів обліку робочого часу, яка буде відстежувати час, витрачений на виконання кількох завдань, підраховувати оплачувані години, допомогу з виставленням рахунків, відстеження BOM тощо. Ці програми можна використовувати на пристроях iOS та Android для заповніть табелі обліку робочого часу та відстежуйте час [18].

Отже, програмне забезпечення робочого часу співробітників повинно мати функції моніторингу в реальному часі, виставлення рахунків, виставлення рахунків, детальну звітність, простоту використання та підтримку декількох платформ. Використання програми розкладу робочого часу має кілька переваг, таких як управління роботою, спрощений процес оплати праці, виставлення рахунків клієнтам, підзвітність команди та ефективне використання часу працівника на проекти та завдання.

1.3. Постановка задачі дослідження

Метою є проектування СКЧ.

Ставимо перед собою наступні задачі:

1.проаналізувати теоретичні засади дослідження застосування систем керування продуктивістю та власним часом;

2. охарактеризувати сучасний стан програмного забезпечення ефективного тайм-менеджменту;

3. розглянути існуючі моделі керування часом;

4. дослідити алгоритм створення СКЧ;

5. спроектувати веб-застосунок для заданих умов та перевірити її ефективність;

6. розробити практичні пропозиції щодо застосування інструментів системного аналізу відносно впровадження клієнт-сервісних програм в процесі розробки СКЧ .

Висновки до розділу 1

Підсумовуючи перший розділ, можемо зробити такі висновки:

1. Визначено, що моніторинг є системою, яка виконує постійне спостереження за явищами і процесами, що відбуваються в навколишньому середовищі і суспільстві, результати якого служать для керування прийняття наступного рішення на основі отриманих раніше даних. Тайм-менеджмент - це процес планування і усвідомленого контролю над часом, що витрачається на певні види діяльності, особливо для підвищення ефективності, результативності та продуктивності. Він включає в себе акт поєднання різних вимог до людини, пов’язаних з роботою, громадським життям, родиною, хобі, особистими інтересами і зобов’язаннями, з обмеженням часу.

2. З’ясовано, що програмне забезпечення для обліку робочого часу - це категорія комп’ютерного програмного забезпечення, яке дозволяє своїм співробітникам записувати час, витрачений на завдання або проекти. Програмне забезпечення використовується в багатьох галузях, в тому числі в тих, де працюють фрілансери і погодинна оплата. Він також використовується професіоналами, які виставляють своїм клієнтам погодинні рахунки. До них відносяться юристи, фрілансери і бухгалтери.

3. Окреслено сучасний стан програмного забезпечення ефективності тайм-менеджменту та сформульовано задачі дослідження.

РОЗДІЛ 2.
МЕТОДИ ТА ЗАСОБИ ВИРІШЕННЯ ЗАДАЧІ

2.1. Огляд існуючих моделей керування продуктивністю та часом

Розглянемо існуючі моделі керування продуктивністю та часом.

1. Монітаск.

Monitask - це потужне автоматичне програмне забезпечення робочого часу для вашої команди. Не потрібно заповнювати та керувати табелями обліку робочого часу вручну - графіки робочого часу співробітника синхронізуються з веб-інформаційною панеллю в режимі реального часу. Крім того, вони на 100% автоматизовані.

Особливості:

Він простий у використанні.

Автоматичне відстеження часу.

Він включає функції відстеження завдань, моніторингу програм та детальних звітів.

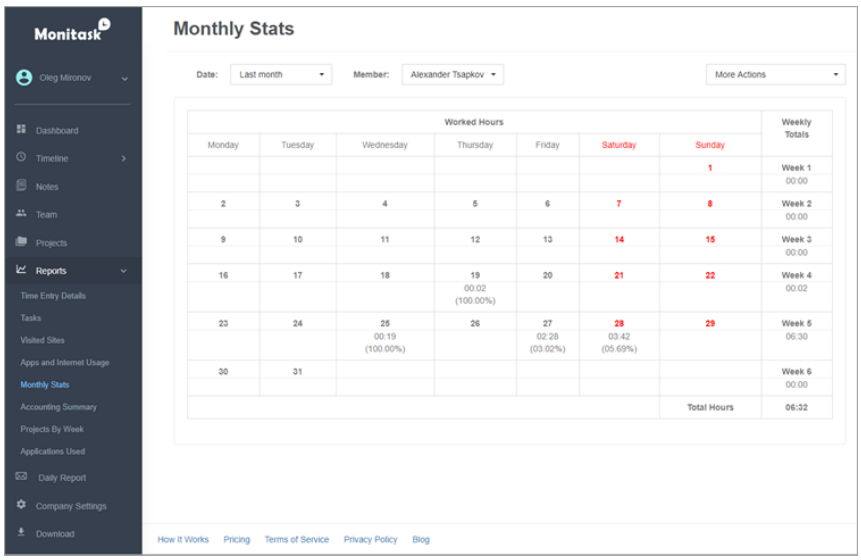


Рисунок 2.1 – Приклад програми Monitask

2) monday.com.

Monday.com надає програму управління часом для ефективнішого управління часом. Він надає різні функціональні можливості, такі як призначення власникам нових завдань, встановлення пріоритетів для кожного елемента, встановлення строків виконання тощо.

Це дасть вам уявлення про те, скільки часу витрачається на кожен проект та завдання. Його мобільний додаток доступний, отже ви можете відстежувати час на ходу. Його можна інтегрувати з улюбленими інструментами та централізувати вашу роботу в одному місці.

Особливості:

monday.com має можливості гнучких звітів. Гнучкі функції звітування дозволять вам аналізувати ваші дані за вашим бажанням. Ви можете розподілити час за проектами, клієнтами та завданнями.

Ви зможете поставити свою роботу на автопілоті.

Це простий і барвистий додаток, який надає точні часові рамки. Він підтримує як ручне, так і автоматичне відстеження часу.

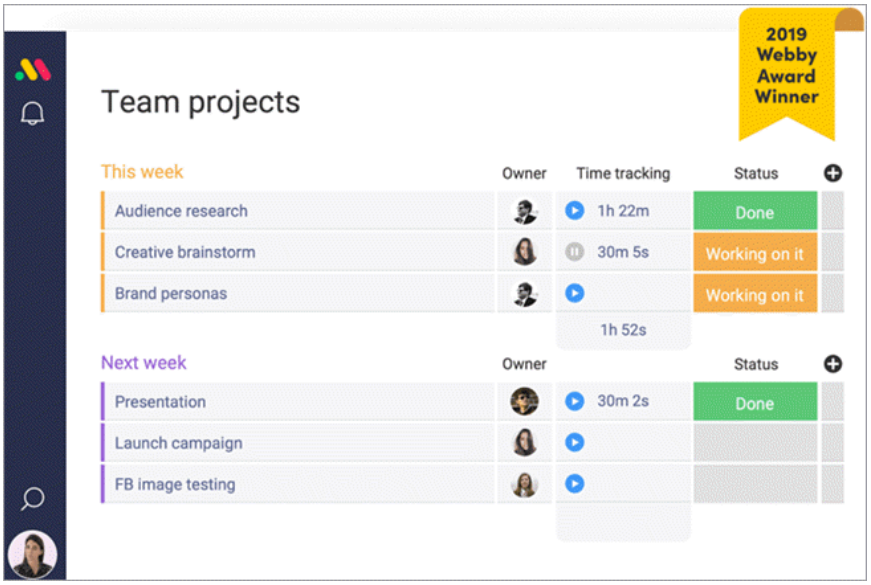


Рисунок 2.2 – Приклад програми monday.com

3) Paymo.

Платформа відстеження часу Paymo реєструє час за допомогою веб-таймера, віджета на робочому столі, Pay Plus та мобільного додатка. Він забезпечує можливість введення часу за допомогою клацання та опускання на табелі обліку робочого часу. Це детально відобразить вашу роботу.

Особливості:

Paymo надає багаті та зрозумілі картки введення часу.

Ви зможете налаштувати параметри робочого часу відповідно до ваших потреб.

Ви можете переглядати час команди за допомогою різних переглядів, таких як щоденний, щотижневий, щомісячний, перегляд порядку денного та активні таймери.

Paymo дозволить вам ділитися звітами про час зі своєю командою або клієнтами.

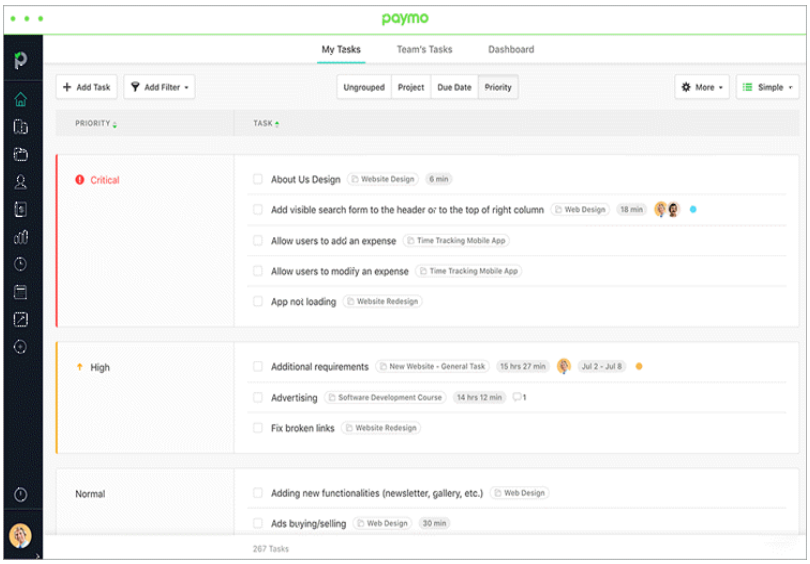


Рисунок 2.3 – Приклад програми Paymo

4) Бадді Панч.

Buddy Punch - це програмне забезпечення для відстеження часу співробітників із повністю настроюваним інтерфейсом. Його можна інтегрувати з популярними продуктами управління заробітною платою. Це простий веб-інструмент відстеження часу.

Це програмне забезпечення спростить ваш графік роботи в Інтернеті. Це дозволить вам генерувати щотижневі звіти вручну. Він має функції для встановлення нагадувань або сповіщень.

Особливості:

Buddy Punch можна легко інтегрувати з програмним забезпеченням для бухгалтерського обліку та оплати праці.

Ви можете увійти в систему за допомогою декількох режимів, таких як ім’я користувача та пароль, електронна адреса, розпізнавання обличчя тощо.

Він надає функції відстеження GPS, які можуть відстежувати та перевіряти кожну зміну кожного дня.

Він може відстежувати BOM, хворих або відпустки.

Це дозволить вам встановити правило і призначити його будь-якій кількості співробітників, які можуть допомогти вам з автоматичними перервами.

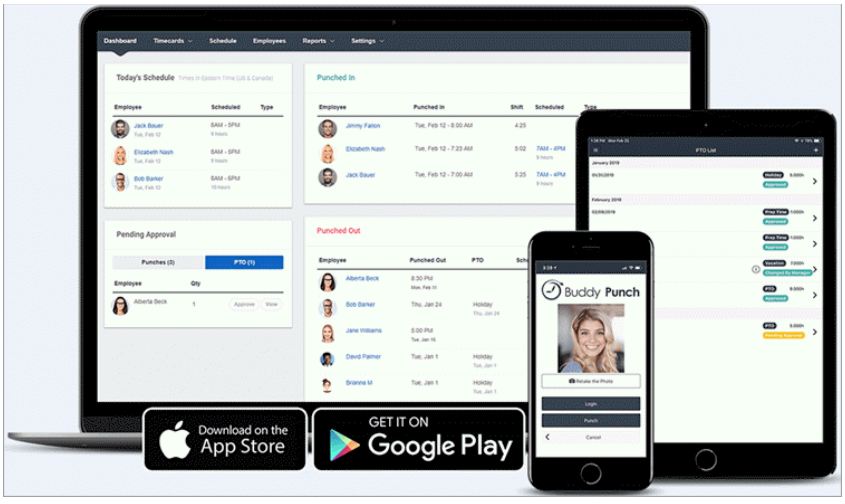


Рисунок 2.4 – Приклад програми Buddy Punch

5) Додаток розкладу TSheets.

TSheets - це програмне забезпечення робочого часу співробітників. Він працює на пристроях Android та iOS. Він має функцію відстеження часу на місці. Далі слід як ручне, так і автоматичне відстеження часу. Він включає функції планування роботи та введення на основі PIN-коду табелів обліку робочого часу.

Особливості:

- Дозволяє вводити час у таблицю робочого часу, вручну та спеціально.
- TSheets працює на будь-якому пристрої.
- 4-значний PIN-код для табелів обліку робочого часу.
- Він відстежує BOM.
- Автоматичні нагадування про вхід та вихід годинника.
- Він також надає попередження про надурочні роботи.

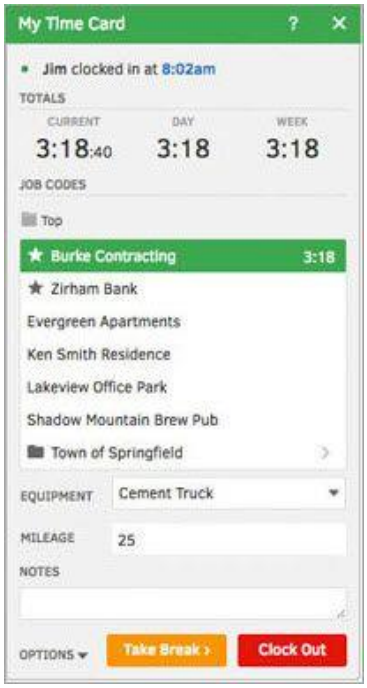


Рисунок 2.5 – Приклад програми TSheets

6) Clockify.

Clockify - це безкоштовний додаток розкладу робочого часу. Це онлайн-програма, яка дозволить працівникам заповнювати табелі обліку робочого часу. Ця програма працює в браузері. Він має функціональні можливості для розрахунку фонду оплати праці та оплачуваних годин.

Особливості:

Це допоможе впорядкувати процес збору робочого часу.

Підходить для щомісячних та погодинних працівників.

Дані робочого часу Clockify можуть бути використані для управління персоналом та заробітної плати, виставлення рахунків клієнтам, звітування про стан проекту та оцінки вартості управлінської діяльності.

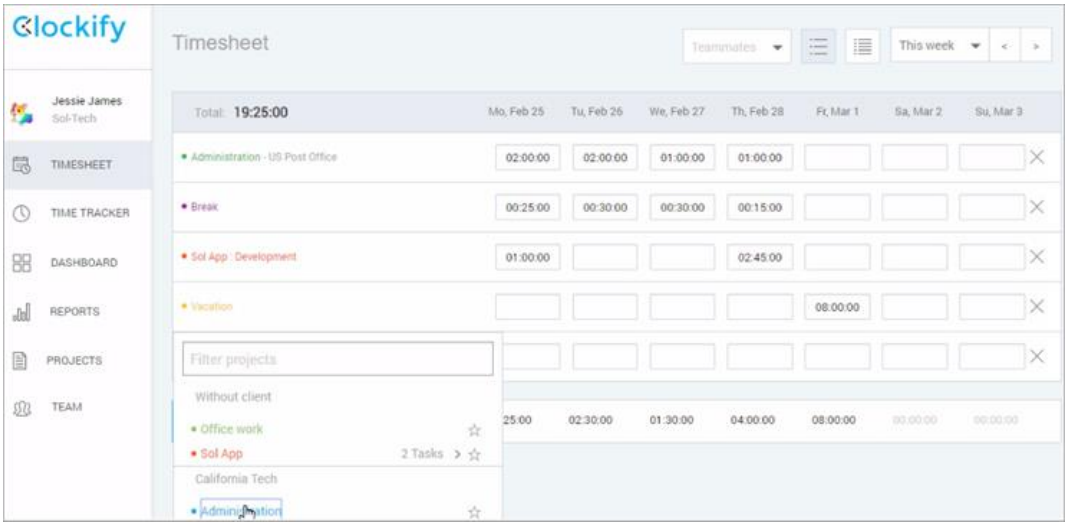


Рисунок 2.6 – Приклад програми Clockify

7) Домашня база.

Домашня база - це програма розкладу робочого часу з функціями планування, годинником часу, табелями робочого часу та командним зв’язком. Ці електронні таблиці робочого часу порівнюватимуть заплановані години. Табелі обліку робочого часу домашньої бази можна експортувати до популярних постачальників заробітної плати.

Особливості:

Ця програма обліку робочого часу відстежує пропущені зміни, пропущені тайм-аути та пропущені перерви.

Відстеження оплачених та неоплачених перерв.

Це допоможе вам дізнатися про вартість робочої сили в режимі реального часу.

Він виконує автоматичний розрахунок часу для загальної кількості годин, надурочних робіт та перерв.

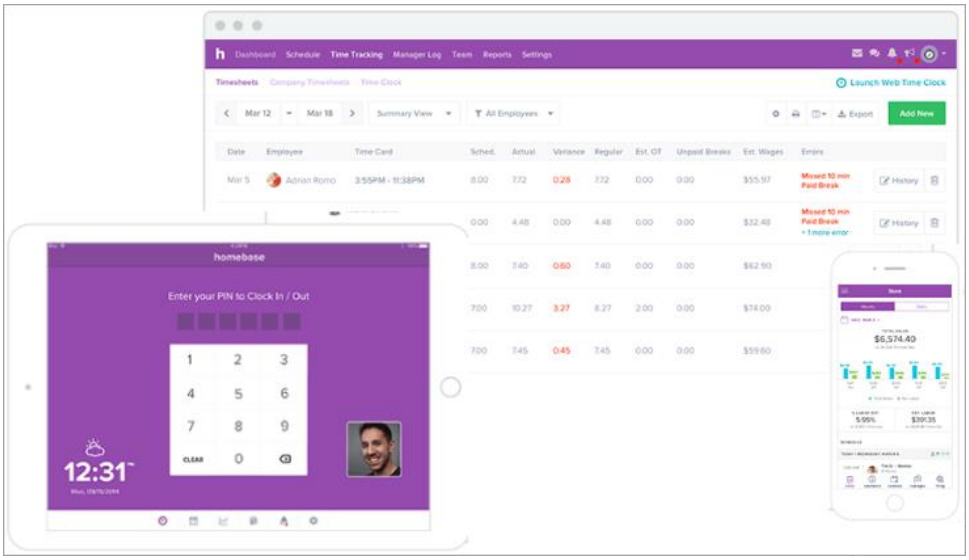


Рисунок 2.7 – Приклад програми Домашня база

8) ClickTime.

ClickTime - програма робочого часу співробітників. Графіки роботи ClickTime доступні на мобільних телефонах з можливістю перегляду та редагування. Мобільний додаток дозволить вам зафіксувати картину квитанцій.

Особливості:

- Відстеження часу за допомогою мобільних додатків.
- Захоплення часу для клієнтів, проектів та завдань.
- Відстеження витрат.
- Мобільний секундомір.

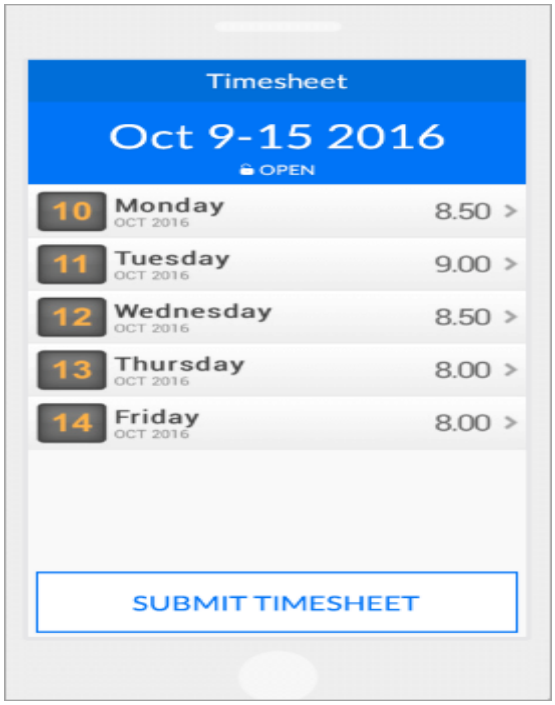


Рисунок 2.8 – Приклад програми ClickTime

9) ZoomShift.

ZoomShift - це онлайн-розклад роботи з функціями, такими як відстеження часу на телефоні, GPS-відстеження та нарахування заробітної плати. Його безкоштовний план підходить для малого бізнесу. Він надсилає автоматичні нагадування працівникам про годинник. Табелі робочого часу, експортовані із ZoomShift, можна безпосередньо надсилати постачальнику заробітної плати (Рис.2.9).

Особливості:

- Табелі обліку робочого часу доступні на основі дня, тижня та місяців.
- Його можна експортувати.
- Він покаже вам детальне порівняння запланованого та фактичного робочого часу (Рис. 2.10).

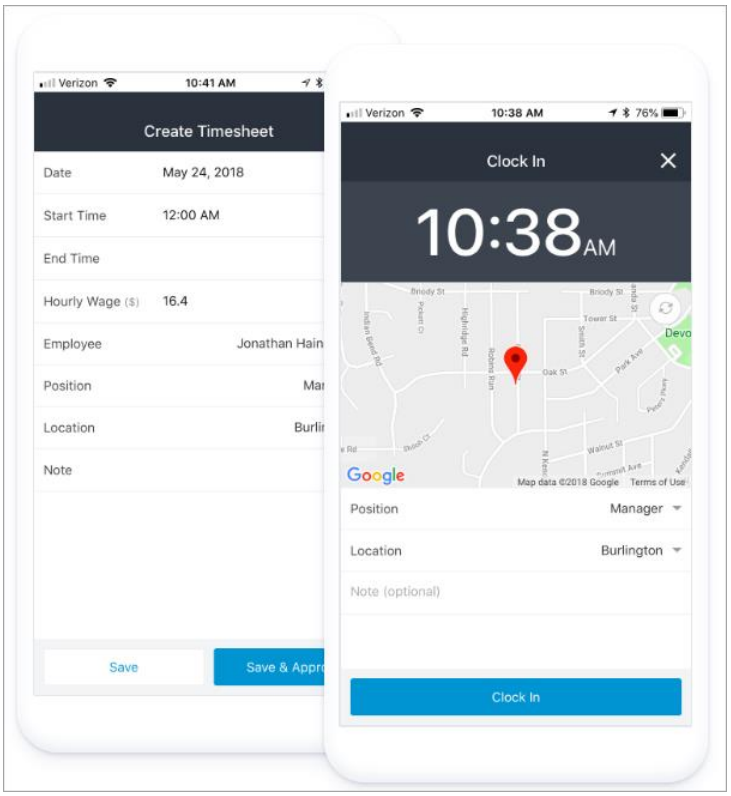


Рисунок 2.9 – Приклад програми ZoomShift

Name	Scheduled	Actual	Status
JE John Elias	9am - 4pm	---	Missed Shift
JE Jeff Erickson	9am - 2pm	9am - ?	Missed Clock Out
JM John Meyer	10am - 4pm	9:30am - 2pm	Early Clock In
SM Sarah Marks	11am - 2pm	? - 2pm	Missed Clock In
JH Josh Harper	---	9am - 2pm	Not Scheduled

Рисунок 2.10 - Порівняння запланованого та фактичного робочого часу

10) Табель робочого часу .io.

Табель роботи - це мобільна програма відстеження часу з такими функціями, як Мобільний трекер, звіти, управління проектами та настроювані рахунки-фактури.

Особливості:

Додаток розкладу часу допоможе вам виставляти рахунки за допомогою настроюваних рахунків-фактур.

Його можна експортувати у формати Excel та CSV.

Звіти та статистика (Рис. 2.11).

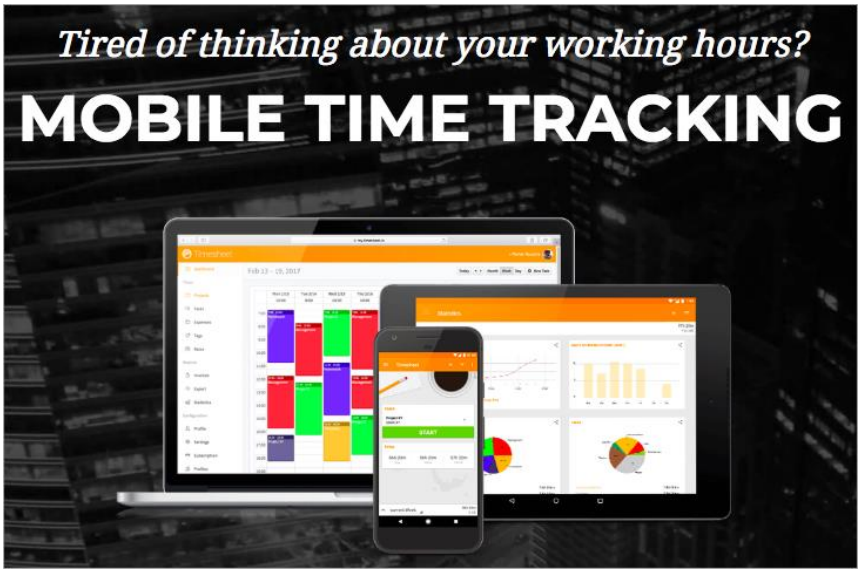


Рисунок 2.11 – Приклад програми Табель робочого часу

11) Запис часу.

Запис часу - це програма з графіком роботи з функціональними можливостями, такими як реєстрація та виїзд призначення завдань та щоденні примітки. Це дозволяє переглядати табелі робочого часу на день, тиждень або місяць. Він доступний для пристроїв Android. Це слідує за автоматичним типом відстеження.

Особливості:

Для звітів та резервного копіювання його можна інтегрувати з Google Drive, DropBox та OwnCloud.

Він має функціональні можливості для призначення завдань.

Він містить докладні примітки.

Це дозволяє експортувати звіти у формати Excel або HTML (Рис. 2.12).

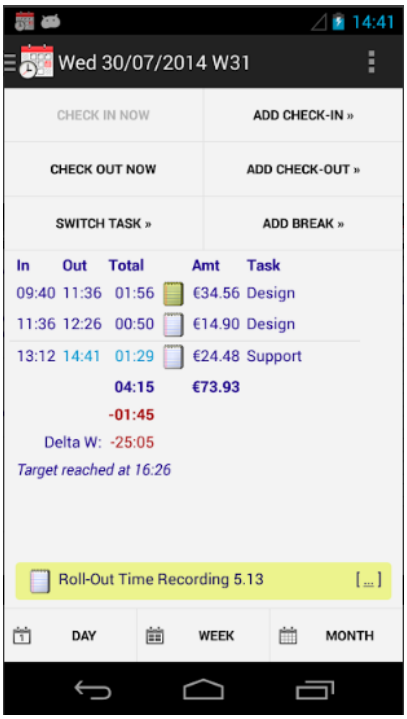


Рисунок 2.12 – приклад програми Запис часу

12) TimeCamp.

TimeCamp - це програмне забезпечення для відстеження часу з такими функціями, як моніторинг продуктивності, відстеження відвідуваності, управління проектами, управління командою та виставлення рахунків. Його мобільний додаток доступний для пристроїв iOS та Android.

Особливості:

TimeCamp пропонує функції для робочого дня та робочого дня тижня.

Тижневий графік роботи має такі функції, як графічний графік робочого часу та відстеження в режимі реального часу.

TimeCamp можна інтегрувати з улюбленим інструментом. Він також надає API для створення власних інтеграцій.

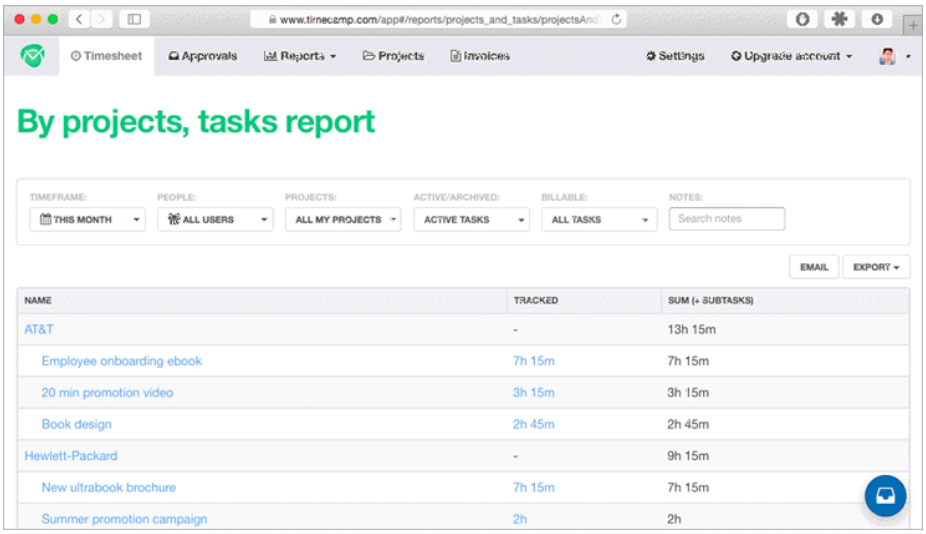


Рисунок 2.13 – Приклад програми TimeCamp

13) Hubstaff.

Hubstaff - це програмне забезпечення для відстеження часу, яке надає онлайн-табелі обліку робочого часу. Це програмне забезпечення робочого часу співробітників допоможе вам в адміністративній роботі з використанням функцій відстеження часу, виставлення рахунків та оплати праці. Він слідує як ручному, так і автоматичному типу відстеження.

Особливості:

Вручну, а також автоматичним введенням часу.

Додаток розкладу роботи доступний для Mac, Linux, Windows, iOS, Android та Chrome.

Створення завдань дозволяється шляхом розбиття проектів. Ця функція допоможе вам з більш точними табелями обліку робочого часу.

Hubstaff має функції для планування, моніторингу співробітників, GPS-відстеження та нарахування заробітної плати.

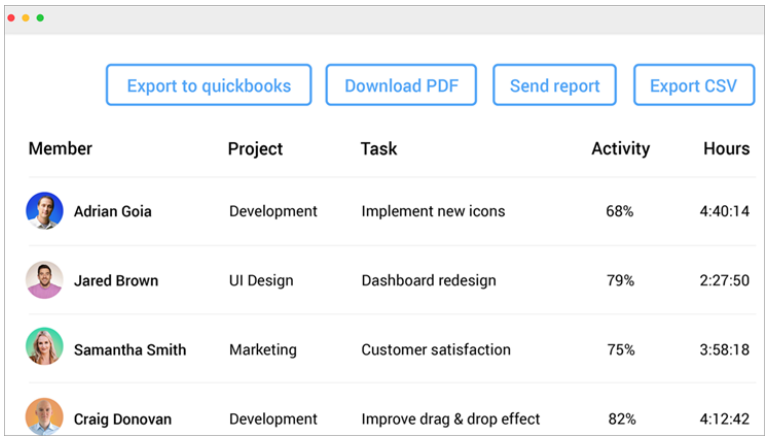


Рисунок 2.14 – Приклад програми Hubstaff

14) Перемикання.

Toggl надає програмне забезпечення обліку робочого часу в Інтернеті. Він слідує як ручному, так і автоматичному типу відстеження. Програмне забезпечення відстеження часу Toggl надасть вам розподіл часу для проектів, завдань та клієнтів. Його можна використовувати як настільний додаток, мобільний додаток або як розширення Chrome.

Особливості:

- Табелі робочого часу співробітників можна легко експортувати.
- Це забезпечить цінні перспективи для ваших щоденних проектів та завдань.
- Toggle забезпечить елегантні та проникливі звіти про час.

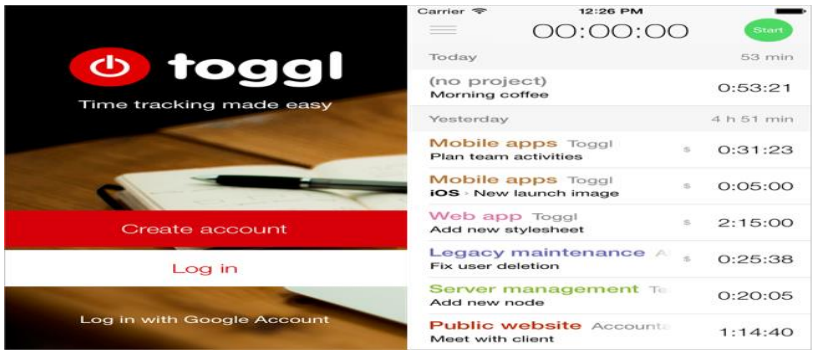


Рисунок 2.15 – Приклад програми Перемикання

15) Smarter Time.

Деякі хронофаги – наприклад, соціальні мережі, забирають у нас стільки цінного часу тільки через те, що ми не усвідомлюємо, скільки саме хвилин і годин витрачаємо на них. Результат – стрес та вигорання. Додаток допомагає відстежувати час, який ми інвестуємо в різні справи, щоб усвідомлено встановлювати баланс. Зокрема, така аналітика корисна для планування розкладу, тренування концентрації, встановлення цілей і навіть відстежування тривалості сну. Деталі: Android; безкоштовно, є платний контент.

16) Sectograph.

Ще один варіант візуалізації планів – цього разу у вигляді циферблата. Достатньо просто створити списки робочих та особистих справ – додаток отримує дані з Google-календаря та відображає їх на 12-секторній діаграмі. Це, як анонсують розробники, «загострює відчуття часу». Деталі: Android; безкоштовно, є платний контент.

17) One Big Thing.

Додаток простий, але може бути дуже корисним для тих, кому важко визначитися з пріоритетами: дозволяє обрати одне найголовніше завдання і нагадує про нього протягом дня. Це онлайн-варіант звичних стікерів, що допомагає використовувати в своїй роботі відомий принцип тайм-менеджменту «1-3-5»: 1 «велика» справа, 3 «середньої важливості» і 5 дрібних. Деталі: IOS; безкоштовно, є платний контент.

18) Планувальник від Agnessa Studio.

В цьому додатку кілька корисних інструментів – планувальник справ, цілей, органайзер, записник, нагадування, календар тощо. Зокрема, в ньому зручно керуватися популярним принципом тайм-менеджменту – «ділити слона на частини», тобто будь-яке завдання може бути роздроблене на підзавдання. Це дає розуміння кроків, які варто зробити для досягнення мети. Деталі: Android; безкоштовно, є платний контент.

19) Pomodoro Timer.

Додаток для тих, хто практикує техніку тайм-менеджменту «помідор». Треба створити перелік завдань, вибрати одне для виконання – і запускається 25-хвилинний таймер. Протягом цього часу необхідно займатися певною справою. Потім – 3-5-хвилинний відпочинок, згодом – новий цикл. Кожні 4 інтервали – велика перерва (до 30 хвилин). Головне не відволікатися на дрібниці протягом сфокусованої роботи і не ігнорувати відпочинок, щоб не втрачати продуктивність. Деталі: Android; безкоштовно.

20) Kanban Tool.

Інструмент базується на ідеях японської системи канбан. Інтерфейс складається з «дошок», на яких відображені завдання і проекти. Особливо зручний для командної роботи: завдання легко створювати та переміщувати, і зміни одразу бачать усі причетні до проекту. Деталі: Android, IOS; безкоштовно.

21) Pocket.

Цікавий контент, що випадково потрапляє до поля зору, відволікає увагу від важливих справ – наприклад, це статті, відео та інші матеріали. Цей додаток дозволяє зберігати сторінки в один клік, щоб подивитися їх потім, коли буде вільний час. Деталі: Android, IOS; безкоштовно, є платний контент.

22) Forest.

Цей додаток допомагаю візуалізувати свою продуктивність. Принцип роботи простий: коли ви починаєте виконувати завдання, яке вимагає зосередженості, то саджаєте віртуальне дерево на гаджеті. Якщо працюєте протягом часу, встановленого на таймері, без перерв на смартфон – дерево виросте, якщо відволікаєтеся – ні. Те, як швидко та густо росте ліс у додатку, дозволяє оцінити свою здатність фокусуватися. В налаштуваннях можна вказувати сайти та інші ресурси, які «крадуть» робочий час, щоб ефективніше контролювати себе. Деталі: Android, IOS; безкоштовно, є платний контент.

23) Stay Focused.

Рішення для тих, кого відволікають соціальні мережі, месенджери та інші ресурси. Додаток дозволяє обмежити доступ до певних сайтів – встановлювати

щоденний ліміт, обмеження в певні проміжки часу тощо. Допомагає розвивати навички самоконтролю. Деталі: Android; безкоштовно, є платний контент.

24) Tick Tick.

Зручний інструмент для створення списків справ. Дозволяє встановлювати дедлайни та налаштовувати нагадування, заповнювати та керувати своїм розкладом у календарі, розшерювати свої завдання для інших користувачів тощо. Деталі: Android, IOS; безкоштовно, є платний контент.

2.2. Підходи до проектування системи та алгоритм її створення

Першим кроком до вирішення поставлених завдань є визначення структури бази даних, в першу чергу слід визначити модель відносин бази даних.

Найпопулярнішою базою даних для сучасних додатків є MongoDB. MongoDB – це міжплатформна програма, з відкритим вихідним кодом, яка спеціально призначена для зберігання ієрархічних структур даних (документів), реалізована за допомогою підходу NoSQL та не потребує опису схеми таблиць.

MongoDB – це база даних документів із необхідною масштабованістю і гнучкістю, з наявністю необхідних для побудови системи моніторингу робочого часу працівників запитами та індексаціями. Модель документа MongoDB проста у вивченні та використанні, але все ж надає всі можливості, необхідні для задоволення найскладніших вимог у будь-якому масштабі.

Серед переваг, які надає база даних документів MongoDB, можна виділити такі:

- MongoDB зберігає дані у гнучких, схожих на JSON документах, тобто значення поля можуть змінюватися від документа до документа, а структура даних може змінюватися з часом.

- Модель документа відображає об’єкти у коді програми, що полегшує роботу з даними.

- Спеціальні запити, індексація та агрегація в режимі реального часу забезпечують потужні способи доступу та аналізу даних.

- MongoDB – це розподілена база даних по своїй суті, тому вбудована та зручна у використанні висока доступність, горизонтальне масштабування та географічний розподіл.

- MongoDB безкоштовний у використанні.

Сьогодні існує величезний попит на складні веб-додатки, які з кожним днем все більше витісняють з ринку настільні програми. Хоча останнім часом програми, що працюють під браузером, поступово починають поступати свою популярність мобільним додаткам, на сьогоднішній день існують два основних шаблона розробки веб-додатків – SPA (Single Page Application) і MPA (Multi Page Application).

MPA – це традиційний веб-додаток. При такому підході кожний раз, коли додаток запитує дані або відправляє їх на сервер, воно змушене отримувати нову сторінку в повному обсязі, а потім візуалізувати її в браузері. Для відносно простих додатків такий підхід цілком підходить і не викликає незручностей. Але коли мова йде про складний додаток з об’ємним для користувача інтерфейсом, з високим ступенем інтерактивності, то на сторінці з’являється безліч користувальницьких інструментів. Все це веде до того, що розмір трафіку помітно збільшується, і при багатосторінковому підході неминуче з’являються проблеми з продуктивністю.

JavaScript – мультипарадигмова (одночасно використовує багату сукупність ідей і понять, які визначають стиль написання комп’ютерних програм) інтерпретована мова програмування з динамічною типізацією (прийом, при якому змінна зв’язується з типом в момент надання значення, а не в момент оголошення змінної). Підтримує імперативний, функціональний і об’єктно-орієнтований стилі програмування. Є реалізацією специфікації ECMAScript.

При розробці метою було зробити мову, яка схожа на Java, але при цьому більш проста у використанні. Серед основних особливостей JavaScript можна виділити такі:

- Функції є об’єктами першого класу.

- Об’єкти з можливістю доступу до метаданих класів під час виконання програми.

- Наявність механізму, призначеного для обробки помилок часу виконання та інших можливих проблем (винятків), які можуть виникнути при виконанні програми.

- Функції є анонімними.

- Приведення типів здійснюється автоматично.

- Звільнення пам’яті від об’єктів, які не будуть використовуватися програмою в подальшому відбувається автоматично.

Система моніторингу робочого часу працівників передбачає собою веб-додаток, доступ до якого працівники можуть отримати через браузер. Оскільки SPA працює на основі JavaScript і завантажує інформацію за запитом з боку клієнта, то очевидним вибором мови програмування буде JavaScript.

Серед головних фреймворків на базі JavaScript можна виділити три основних – React.js, Angular.js, Vue.js.

Середовище розробки програмного забезпечення – система програмних засобів, яка використовується програмістами для розробки програмного забезпечення. Зазвичай середовище розробки включає в себе текстовий редактор, компілятор і/або інтерпретатор, засоби автоматизації збирання і відлагоджувач (debugger). Іноді також містить засоби для інтеграції з системами управління версіями і різноманітні інструменти для спрощення конструювання графічного інтерфейсу користувача. Багато сучасних середовищ розробки також включають браузер класів, інспектор об’єктів і діаграму ієрархії класів - для використання при об’єктно-орієнтованій розробці програмного забезпечення.

Хоча й існують середовища розробки, призначені для декількох мов - такі як Microsoft Visual Studio, зазвичай середовище розробки призначається для одного певного мови програмування - як наприклад, Visual Basic (Рис. 2.16).

Серед найпопулярніших середовищ розробки, які підтримують розробку на мові JavaScript з використанням фреймворку React, можна виділити два середовища розробки – VS Code та WebStorm.

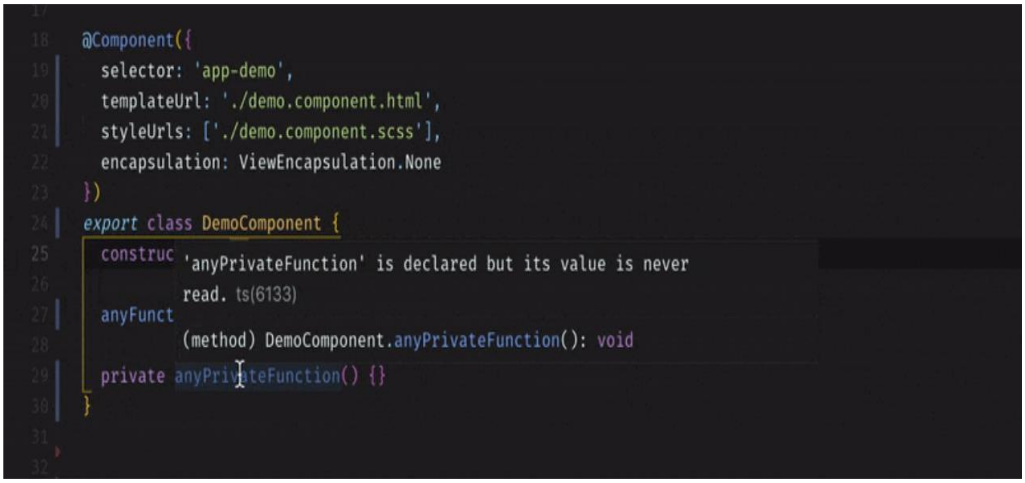


Рисунок 2.16- Вигляд редактору коду в Visual Studio Code

WebStorm забезпечує надійний, швидкий та гнучкий аналіз статичного коду. Цей аналіз виявляє помилки мови та часу виконання, пропонує виправлення та вдосконалення. Він також індексує весь ваш проект і може, наприклад, виявити всі невикористані методи, змінні та інше.

Ви також можете виявити невикористані методи в методах JavaScript, використовуючи VS Code та ESLint. Але якщо ви, наприклад, використовуєте проект TypeScript (наприклад, Angular), VS Code не виявляє невикористані публічні методи. WebStorm також має інтегрований тестовий старт.

Таким чином можливо запустити свої тести безпосередньо з IDE і навіть налагодити їх там (Рис. 2.17).

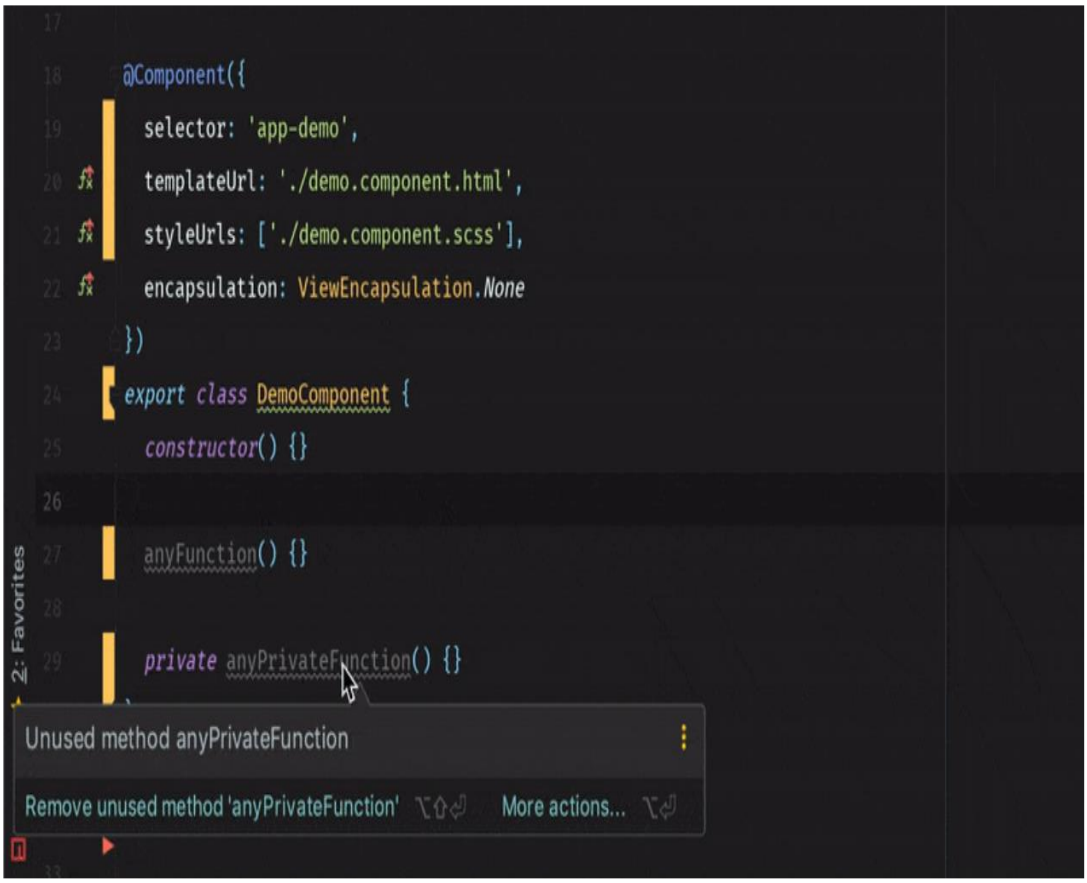


Рисунок 2.17 –Вигляд редактору коду в WebStorm

2.3. Тестування розробленої програми

Для проведення модульного тестування СКЧ нами було використано пакет MSTest.

У наступному прикладі (Рис. 2.18) показаний результат проходження системою тестувань.

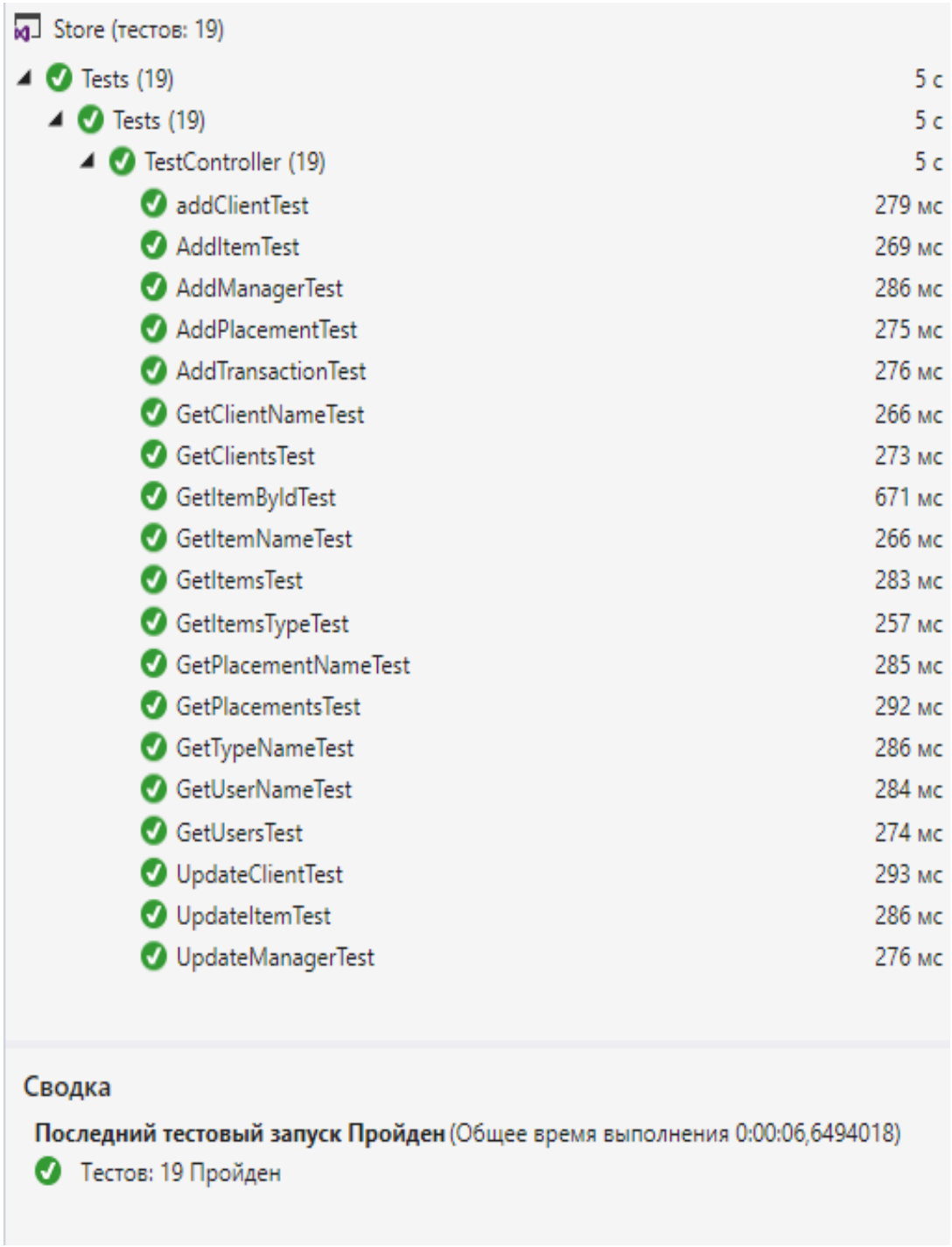


Рисунок 2.18 – Тестування розробленої СКЧ

В рамках тесту програми моніторингу тестувальники повинні переконатися, що в програмному забезпеченні виявлені помилки, перш ніж воно буде доставлено користувачам або замовникам.

Вони тестують не тільки ПЗ, але і впроваджені бізнес-процеси і систему в цілому.

Системний тест – дуже складний проект, що вимагає власних методів та інструментів.

На даному етапі, в програмних продуктах не було виявлено ніяких порушень, тестування проводилося, як перевірка даних, що повертаються зконтролерів, приклад виводу інформації показано на рисунку 2.19.

```
namespace Tests
{
    [TestClass]
    public class TestController
    {
        [TestInitialize]
        public void TestInit()
        {
            Controller ctr = Controller.TestInstance;
            ctr.clearBase();

            ctr.addManger("Иванов Василий",22, DateTime.Today.ToString());
            ctr.addManger("Каргаполов Юрий",31, DateTime.Today.ToString());
            ctr.addManger("Дягтерев Дмитрий",25, DateTime.Today.ToString());

            ctr.addClient("ЧП Прайм Тай", "051-123-213-33");
            ctr.addClient("ЗАТ Петровский кабинат", "051-723-612-54");
            ctr.addClient("ТОВ Интели сенс", "031-563-233-53");

            ctr.addPlacement("Главный склад",1);
            ctr.addPlacement("Сортировочный склад",2);
            ctr.addPlacement("База 1",1);
            ctr.addPlacement("База 2",1);

            ctr.addItem("Товар 1", "новый",2,3,1);
            ctr.addItem("Товар 2", "б/у",1,1,2);
            ctr.addItem("Товар 3", "новый",1,1,2);
            ctr.addItem("Товар 4", "новый",3,4,3);

            ctr.addItemType("Стол");
            ctr.addItemType("шкаф");
            ctr.addItemType("Стул");
        }
        [TestCleanup]
        public void TestCleanup()
        {
            Controller ctr = Controller.TestInstance;
            ctr.clearBase();
        }
        [TestMethod]
        public void GetItemByIdTest()
        {
            Controller ctr = Controller.TestInstance;
            var item = ctr.getItemById(1);
            Assert.IsTrue(item.id == 1);
        }
        [TestMethod]
        public void AddItemTest()
        {
            Controller ctr = Controller.TestInstance;
            ctr.addItem("testName", "testState", 1, 1, 1);
            var items = ctr.getItems();

            bool isAdded = false;

            foreach (var it in items)
                if (it.name == "testName")
                    isAdded = true;

            Assert.IsTrue(isAdded);
```

Рисунок 2.19 – Приклад частити системного тестування програми

Висновки до розділу 2

Підсумовуючи другий розділ, можемо зробити такі висновки:

1. Визначено існуючі рішення СКЧ та деталізовано найбільш ефективніші з них з точки зору пріоритезації під час оптимізаційних чинників.

2. Проаналізовано підходи до проектування СКЧ та визначено алгоритм її створення.

3. Проведено тестування автоматизованої СКЧ із використанням пакету MSTest.

РОЗДІЛ 3.
ПРАКТИЧНА РЕАЛІЗАЦІЯ ПРОГРАМИ ДЛЯ КЕРУВАННЯ ЧАСОМ

3.1. Розробка структурної схеми та архітектури прототипу

Для побудови схеми і архітектури СКЧ варто, перш за все, деталізувати завдання дослідження. Задачі мають включати наступні етапи:

1.Проектування програмного режиму функціонування СКЧ.

2. Створення структурної БД програми, яка моделюється.

3. Розробка інтерфейсу програми.

4. Моделювання ПЗ для моделі СКЧ.

5. Застосування режиму тестування всіх структурних компонентів розробленої програми задля визначення її ефективності.

Окреслемо сфери практичної реалізації розробленої системи.

Наступною стадією проектування маємо формалізувати постановку задач дослідження.

Оскільки підготовчі стадії реалізації вже були проведенини на стадії реалізації основної програми, можна зразу переходити до встановлення ПЗ.

Розробка СКЧ вимагає чіткого та структурованого підходу до її створення.

Функціональна схему системи складається з:

- веб-сервера,

- бази даних,

- веб-сторінки,

-серверу месенджера,

- месенджера.

В якості серверної системи управління базами даних була обрана MongoDB. Структура отриманої моделі даних представлена на рисунку 3.1.

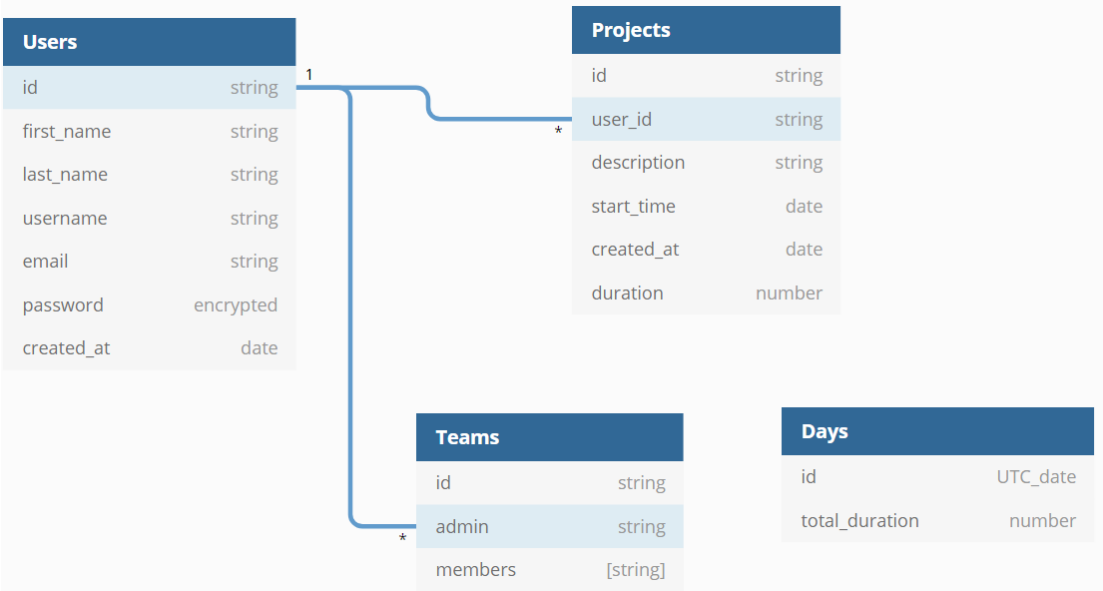


Рисунок 3.1 – Модель даних СКЧ

1. Користувач .

Дана сутність включає в себе наступні поля:

- username – коротке ім’я користувача;
- email – адреса електронної пошти користувача для реєстрації та логіну у веб-додаток;
- password – пароль користувача, який хешується при збереженні в базу даних;
- createdAt – дата створення користувача у форматі ISO 8601.

Перерахованих вище даних достатньо, щоб організувати процес реєстрації і аутентифікації користувача. Крім того, вищевказані дані знаходяться у вкладці «Profile» користувача та мають можливість бути редагованими.

2. Проект.

Дана сутність включає в себе наступні поля:

- userid – використовується для визначення відповідального за виконаний проект;
- description – залишений працівником, короткий опис виконуваного проекту;

- createdAt – дата створення проекту;
- startTime – час початку виконання проекту;
- duration – загальний час, який було витрачено на роботу над проектом.

За допомогою userid визначається приналежність виконаного проекту до працівника, який його виконував. За допомогою createdAt зберігається дата початку виконання проекту. startTime позначає дату та час початку роботи над проектами. duration визначає загальну кількість часу, який було витрачено на проект, та обраховується відніманням startTime від окремої змінної endTime.

В даному підрозділі представлені сценарії роботи програми у вигляді алгоритмів, які, будучи перенесені в код, є вирішенням в поставлених в п.1.3 задач.

Алгоритм підготовки додатку до роботи представлений на рисунках 3.2-3.4.

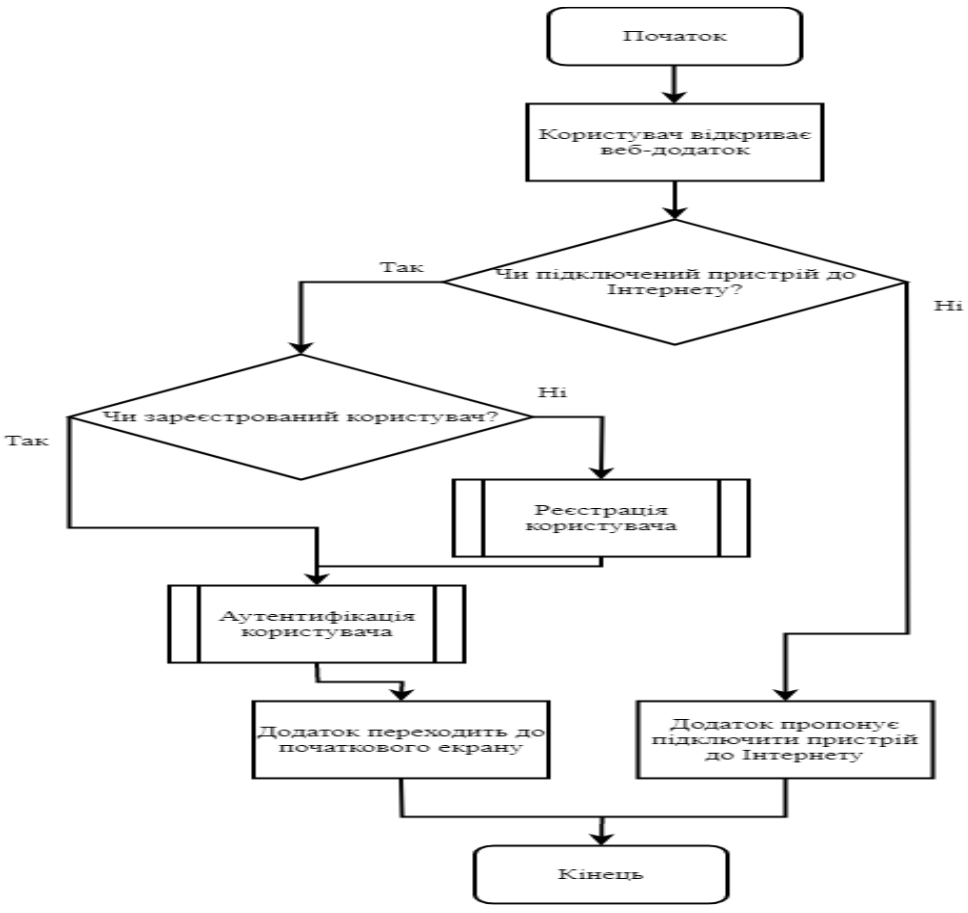


Рисунок 3.2 – Алгоритм підготовки програми до роботи (основний процес)

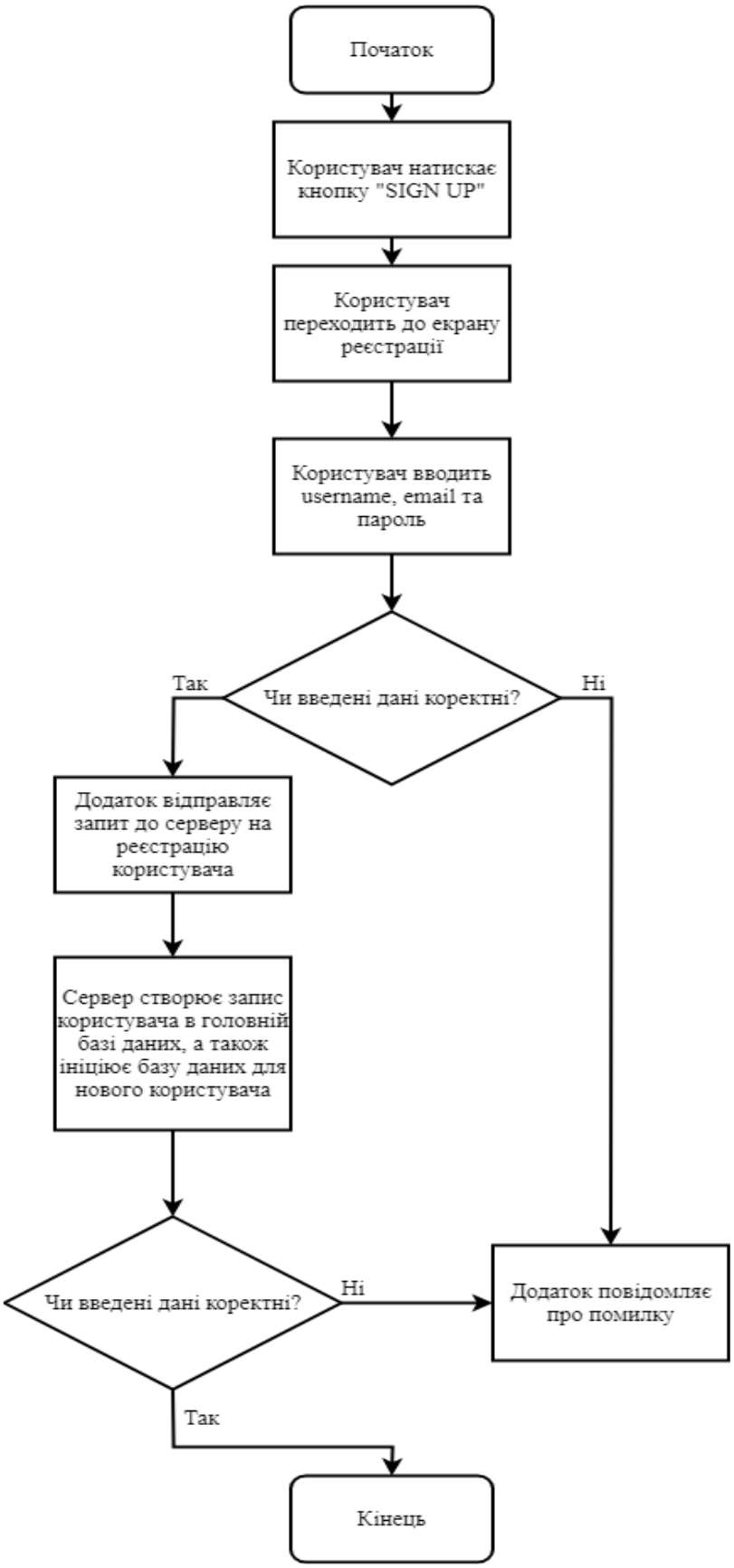


Рисунок 3.3 – Процес реєстрації

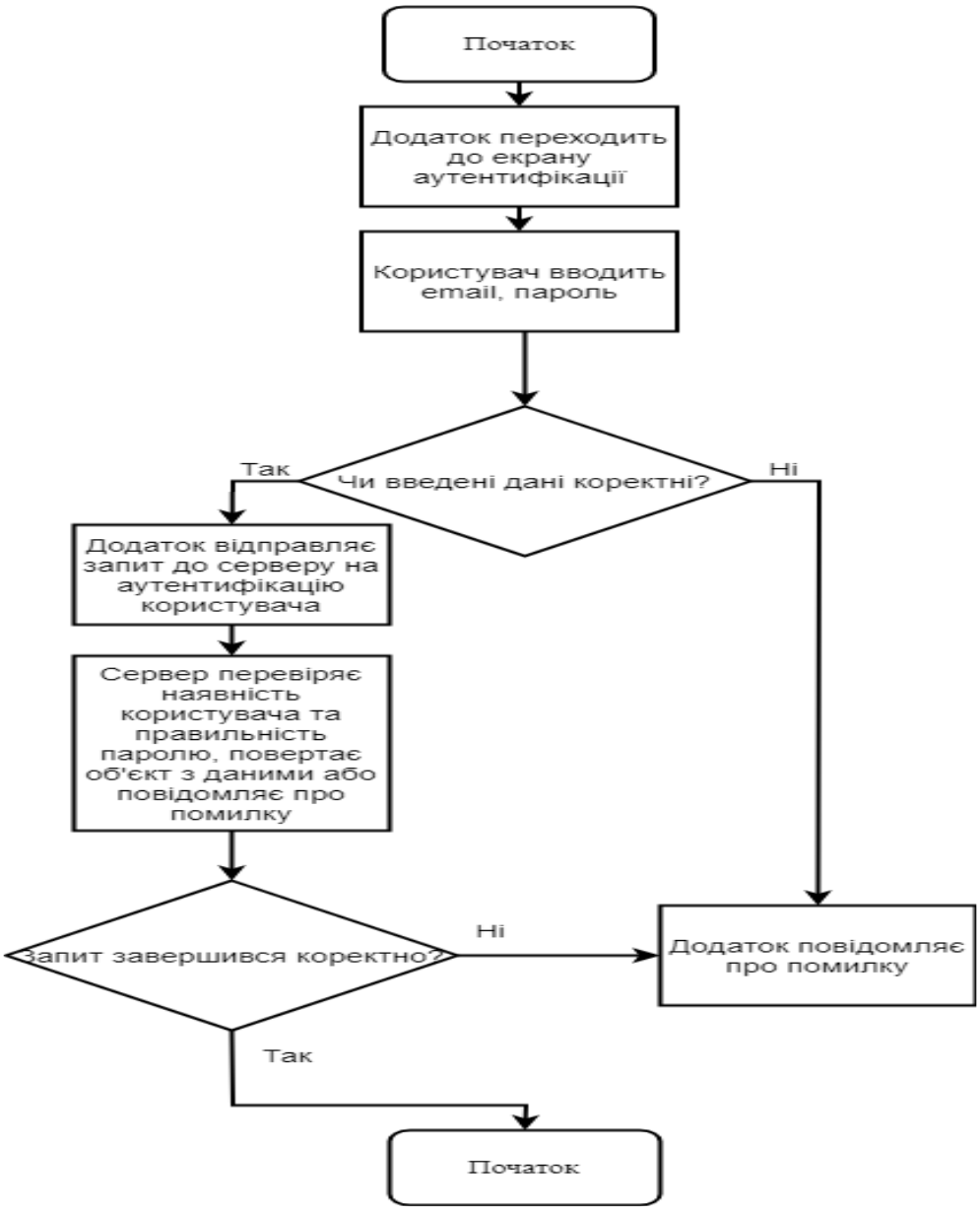


Рисунок 3.4 – Процес аутентифікації

3.2. Впровадження системи та перевірка її ефективності

Додаток складається з декількох загальних модулів, що забезпечують роботу інших, більш спеціалізованих:

– store.js (сховище) - сховище стану програми, забезпечує атомарність і відстеження змін стану;

– routes.js (маршрути) - модуль, який відповідає за переходи між різними екранами додатку;

– app.js - головний модуль, точка входу в додаток, відповідає за ініціалізацію всіх інших модулів і бібліотек при запуску.

Після завантаження програми ініціалізується app.js і store.js, після ініціалізації стану програми управління передається в код модуля routes.js.

Роутер на підставі стану програми може передавати управління одному з наступних модулів:

– Login.js - екран входу та реєстрації в додаток;

– Dashboard.js – екран статистичних даних користувача;

– Profile.js – екран, що відображає особисту інформацію користувача;

– Projects.js - екран списку виконаних (або у стані виконання) проектів;

– TimeTracker.js – екран, що відповідає за початок роботи додатку;

– Teams.js - екран команд користувача.

Кожен з цих модулів містить в собі опис користувальницького інтерфейсу, а також логіки роботи програми.

Для роботи з внутрішнім станом клієнтської частини мі використовуємо бібліотеку Redux. Ця бібліотека імплементує flux architecture. В основі такої структури є store, який зберігає дані та в залежності від actions змінює внутрішній стан. Елементи нашого додатку, що підписуються на store, отримують дані, які змінюються автоматично зі зміною внутрішнього стану. Щоб спростити роботу з внутрішнім станом, кореневий reducer розбили на кілька редюсерів.

В залежності від результату валідації, функція handleUserSignup повертає помилку зі статусом 400 або дістає дані з тіла запиту. Відбувається пошук користувача з переданою електронною поштою і, якщо такий користувач вже є в базі, повертаємо помилку: «User Already Exists». Якщо пройшли попередні перевірки, то створюємо об’єкт з отриманими даними, хешуємо пароль за допомогою бібліотеки Bcrypt, та зберігаємо користувача у базі даних. Генеруємо

authorization token з id отриманого користувача та відправляємо клієнту разом з об’єктом користувача.

```
const handleUserSignup = async (req, res) => {
const errors = validationResult(req);
if (!errors.isEmpty()) {
return res.status(400).json({
error: normalizeValidationResult(errors.array()),
});
}
const { username, email, password } = req.body;
try {
let user = await User.findOne({ email });
if (user) {
return res.status(400).json({ error: ‘User Already Exists’ });
}
user = new User({ username, email, password });
const salt = await bcrypt.genSalt();
user.password = await bcrypt.hash(password, salt);
await user.save();
const token = generateToken({ id: user.id });
res.status(200).json({
user,
token,
});
} catch (err) {
console.error(err.message);
res.status(500).json({ error: ‘Error in Saving’ });
}
};
```

Функція handleUserLogin спочатку валідує email та пароль. Якщо валідація не пройшла, повертає помилку зі статусом 400. Інакше, ці дані дістаються з тіла запиту та шукаємо користувача з такою електронною адресою. Якщо користувача не знайдено в базі даних, клієнту повертається помилка: «User Not Exists». Коли користувача знайдено, йде перевірка пароля: в Bcrypt бібліотеці викликаємо функцію compare, куди передаємо захешований пароль з бази даних та отриманий. Якщо паролі не співпадають, повертаємо помилку: «Incorrect Password». Так само, як і в попередній раз, генеруємо authorization token з id отриманого користувача та відправляємо клієнту разом з об’єктом користувача.

```
router.post(
  '/login',
  [
    check('email', 'Please enter a valid email').isEmail(),
    check('password', 'Please enter a valid password').isLength({ min: 6 }),
  ],
  handleUserLogin,
);
```

Рисунок 3.5 – /login endpoint

const handleUserLogin = async (req, res) => {
const errors = validationResult(req);
if (!errors.isEmpty()) {
return res.status(400).json({
error: normalizeValidationResult(errors.array()),
});
}
const { email, password } = req.body;
try {
const user = await User.findOne({ email });
if (!user) {
return res.status(400).json({
error: 'User Not Exist',
});
}
const isMatch = await bcrypt.compare(password, user.password);
if (!isMatch) {
return res.status(400).json({
error: 'Incorrect Password !',
});
}
const token = generateToken({ id: user.id });
res.status(200).json({
user,
token,
});
} catch (e) {
console.error(e);
res.status(500).json({
error: 'Server Error',
});
}

```
};
router.get('/me', auth, async (req, res) => {
  try {
    // request.user is getting fetched from Middleware after token authentication
    const user = await User.findById(req.id);
    res.json(user);
  } catch (e) {
    res.send({ message: 'Error in Fetching user' });
  }
});
```

Рисунок 3.6 – /me endpoint

projectRoute також обробляє три кінцеві точки (endpoint).

```
const express = require('express');
const {
  handleAddProject,
  handleGetProject,
  handleGetProjectsByUser,
} = require('./handlers');
const auth = require('../../middleware/auth');

const router = express.Router();

router.post('/', auth, handleAddProject);
router.get('/:projectId', auth, handleGetProject);
router.get('/', auth, handleGetProjectsByUser);

module.exports = router;
```

Рисунок 3.7 – Кінцеві точки, які оброблює projectRoute

Якщо прийшов POST запит, перевіривши функцією auth валідність авторизації, викликаємо функцію handleAddProject. Дістаємо description з тіла запиту та id користувача з самого запиту. Шукаємо проект для користувача з цим id. Якщо проект не знайдено – створюємо новий. Зберігаємо проект в базі даних та повертаємо клієнту.

```
const handleAddProject = async (req, res) => {
  const { description } = req.body;
  console.log(description)
  const { id: userId } = req;
```

```
try {
let project = await Project.findOne({ userId });
if (!project) {
project = new Project({ description, userId });
}
await project.save();
res.status(200).json({
project,
});
} catch (err) {
console.error(err.message);
res.status(500).json({ error: 'Error creating project' });
}
};
```

На рисунку 3.8 представлена форма входу в додаток. Дана форма містить два поля введення, які беруть відповідно адресу електронної пошти та пароль користувача. Кнопка «SIGN UP» дозволяє перейти до екрану реєстрації нового користувача. Кнопка «LOG IN» запускає процес аутентифікації після введення валідних значень у відповідні поля.

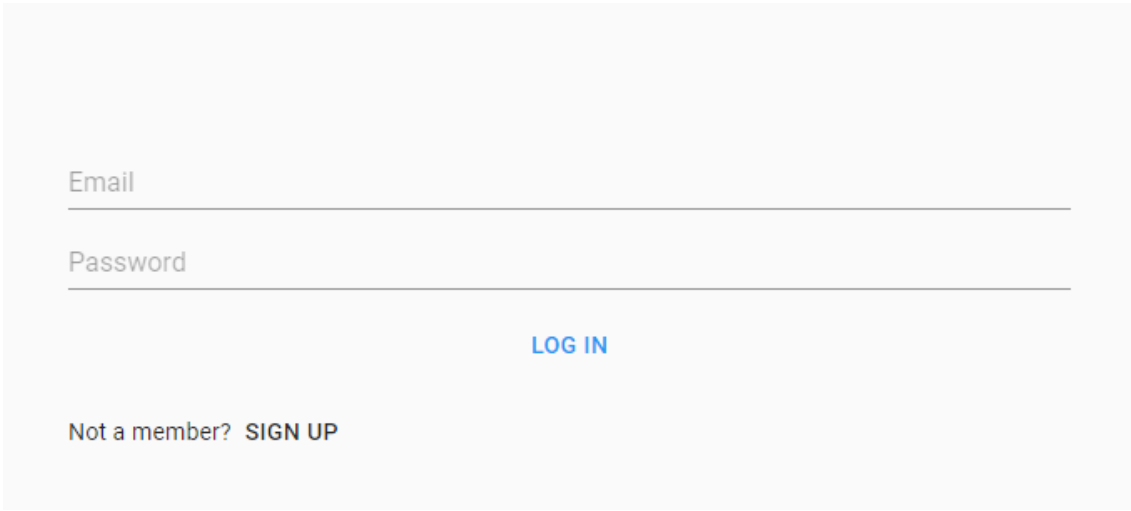


Рисунок 3.8 – Екран входу LOG IN

На малюнку 3.9 представлена форма реєстрації нового користувача. Дана форма містить три поля введення, відповідно ім'я (Username), адреса електронної пошти (Email) та пароль (Password) користувача. Кнопка «SIGN UP» запускає

процес реєстрації після введення валідних значень. Кнопка «LOG IN» дозволяє повернутися до екрану входу в додаток.

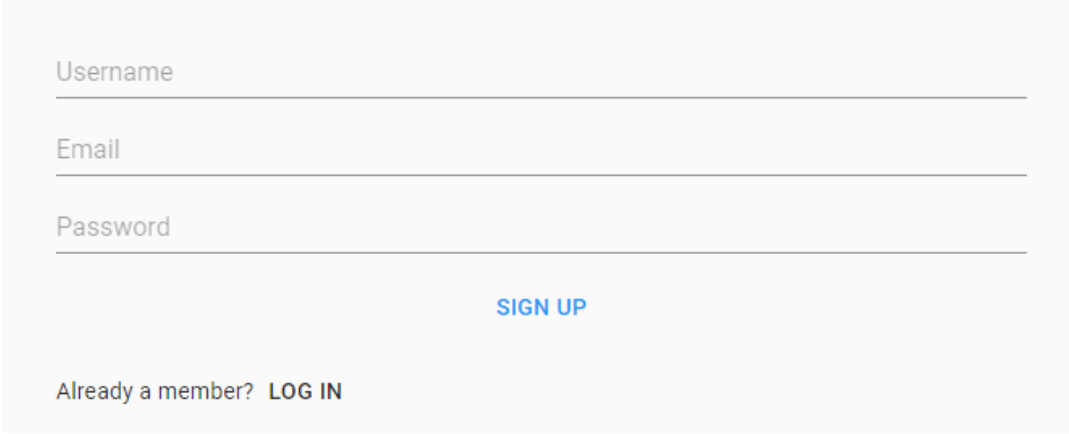


Рисунок 3.9 – Екран реєстрації SIGN UP

На рисунку 3.10 представлено екран завантаження програми. Даний екран не містить активних елементів і призначений для відображення прогресу завантаження даних з Redux Store.

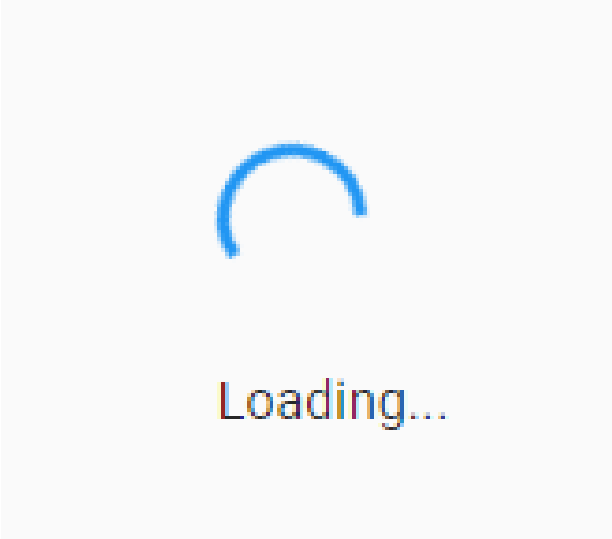


Рисунок 3.9 – Екран завантаження LOADING

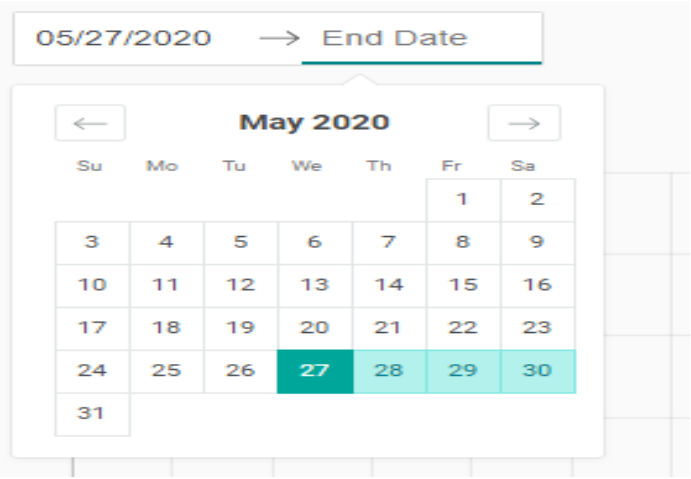


Рисунок 3.10 –Вибір періоду часу за допомогою кнопок Start Date і End Date

На рисунку 3.11 представлена панель меню. Дана форма містить посилання, що дозволяють перейти до інших категорій додатку, серед яких є «Profile», який у меню відображає поточні ім’я користувача та електронну адресу, «Time Tracker», «Dashboard», «Projects» і «Teams».

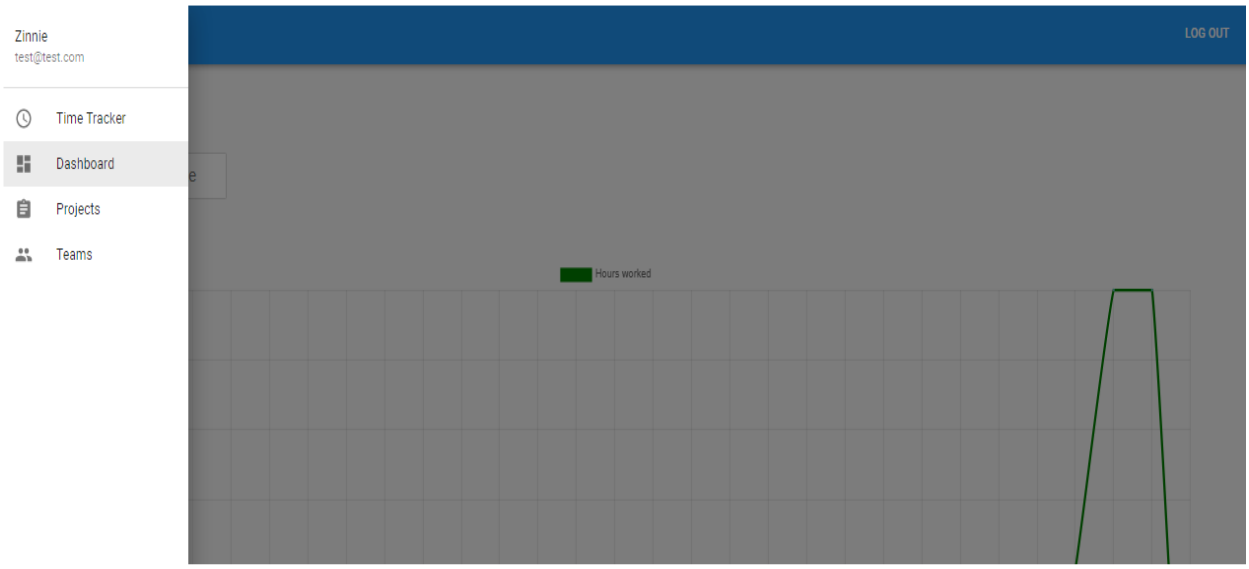


Рисунок 3.12 – Панель меню

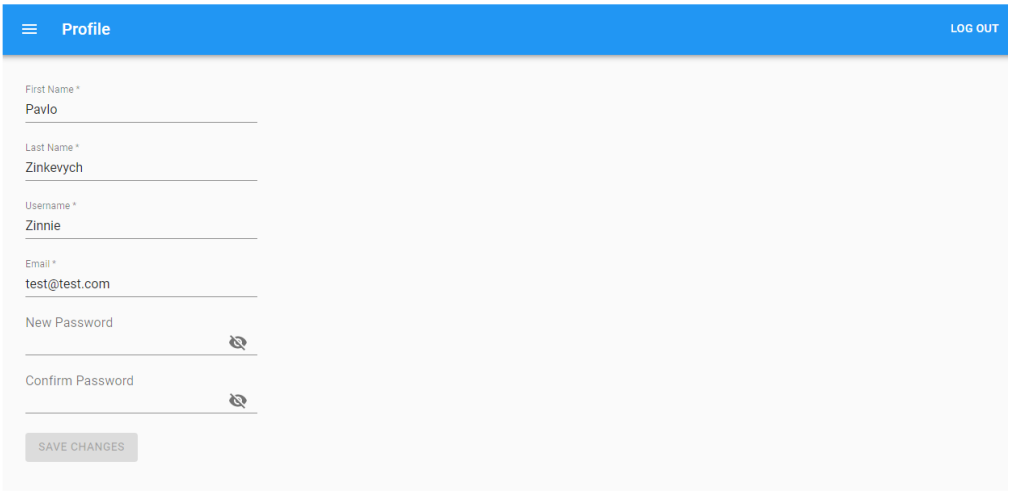


Рисунок 3.13 – Екран профілю користувача Profile

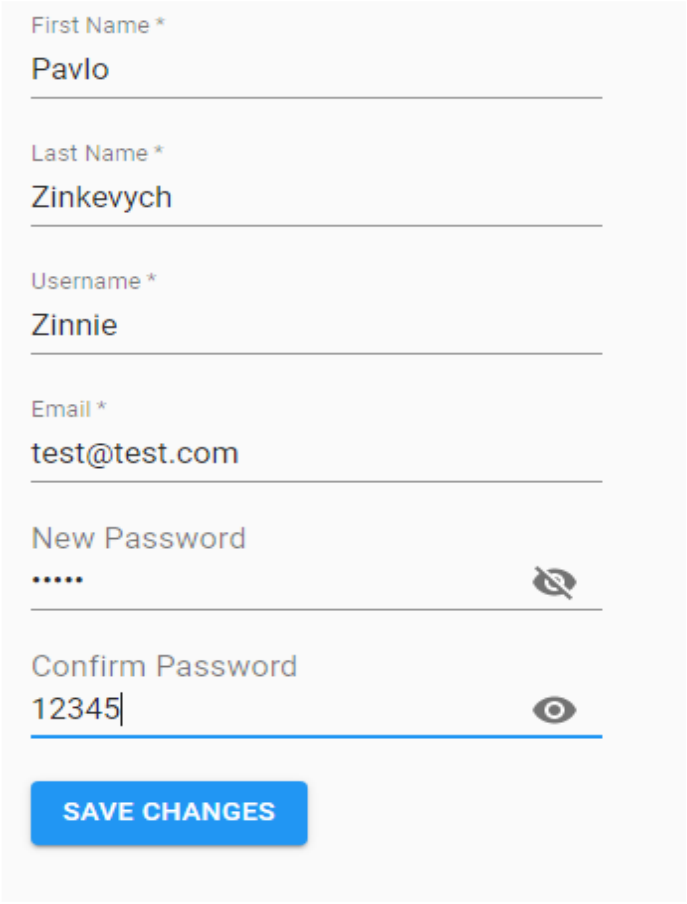


Рисунок 3.14 – Процес зміни паролю користувача

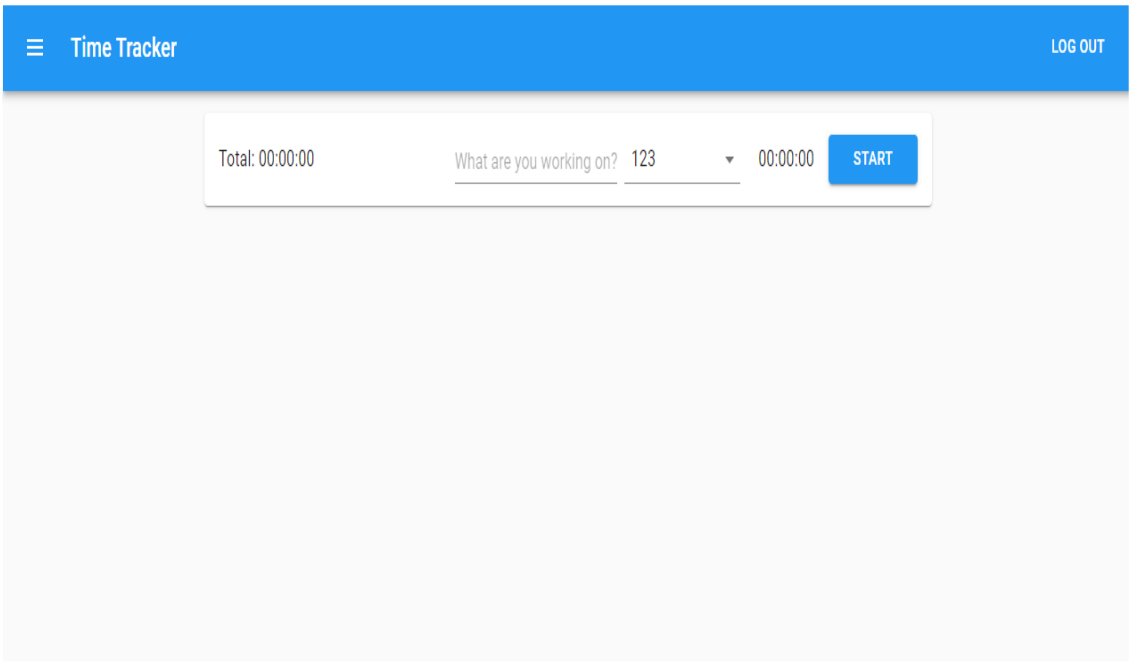


Рисунок 3.15 – Екран відліку часу Time Tracker

3.3. Пропозиції щодо успішного функціонування системи

Грамотна експлуатація СКЧ продовжує термін її корисної дії. Тому варто строго дотримуватися рекомендацій щодо її використання.

При експлуатації системи можуть виник акати дрібні несправності, з якими легко впоратися:

Іноді поганий зв’язок призводить до розриву сигналу між користувачем та адміністратором. Також причиною цього може стати відсутність підключення між компонентами обладнання.

З метою усунення неполадок необхідно протестувати режими каналів різних форматів, доречних для даної версії СКЧ.

Недотримання рекомендацій щодо коректного запуску системи в роботу, порушення послідовності включення обладнання або перебої з живленням можуть привести до тривалого зависання програми.

Таким чином, використання СКЧ потребує серйозного і відповідального ставлення. Якщо у користувача немає достатнього досвіду роботи з нею, то з

приводу усунення неполадок краще звернутися до професіоналів технічної підтримки.

Висновки до розділу 3

Підсумовуючи третій розділ, можемо зробити такі висновки:

1. Спроектовано схему та архітектуру прототипу СКЧ.

2. Розроблено модель функціонування системи та структуру БД, змодельовано інтерфейс програми, створено систему СКЧ.

3. Розроблено пропозиції щодо успішного функціонування системи.

ВИСНОВКИ

Підсумовуючи загальний зміст роботи, можемо зробити наступні висновки:

1. Визначено, що моніторинг – система, за допомогою якої здійснюється постійне спостереження та оцінка змін стану будь-якого технічного, соціального, природного та інших об’єктів. В нашому випадку моніторинг (також можна використовувати слово «відстеження» для опису даної проблеми) застосовується задля визначення конкретної кількості величини – часу. Система моніторингу робочого часу працівників призначена для відстеження часу, під час якого працівник виконував свою роботу.

2. Охарактеризовано, що тайм-менеджмент - це процес планування і усвідомленого контролю над часом, що витрачається на певні види діяльності, особливо для підвищення ефективності, результативності та продуктивності. Він включає в себе акт поєднання різних вимог до людини, пов’язаних з роботою, громадським життям, родиною, хобі, особистими інтересами і зобов’язаннями, з обмеженням часу. Ефективне використання часу дає людині «вибір» витрачати або керувати діяльністю у зручний для нього час і в зручний для нього час. Управління часом може допомогти ряд навичок, інструментів і методів, використовуваних для управління часом при виконанні певних завдань, проектів і цілей з дотриманням встановленого терміну. Спочатку управління часом відносилося тільки до ділової або робочої діяльності, але з часом цей термін розширили і включили також і особисту діяльність. Система тайм-менеджменту - це продумана комбінація процесів, інструментів, технік і методів. системи управління часом часто включають години або веб-додаток, що використовується для відстеження робочого часу співробітника. Системи тайм-менеджменту дають роботодавцям подання про свою робочу силу, дозволяючи їм бачити, планувати і управляти своїм часом співробітників. Це дозволяє роботодавцям управляти витратами на робочу силу і підвищувати продуктивність. Система тайм-менеджменту автоматизує процеси, позбавляючи від паперової роботи.

3. Проаналізовано, що програмне забезпечення для обліку робочого часу - це категорія комп'ютерного програмного забезпечення, яке дозволяє своїм співробітникам записувати час, витрачений на завдання або проекти. Програмне забезпечення використовується в багатьох галузях, в тому числі в тих, де працюють фрілансери і погодинна оплата. Він також використовується професіоналами, які виставляють своїм клієнтам погодинні рахунки. До них відносяться юристи, фрілансери і бухгалтери. Можливості, пропоновані програмним забезпеченням для обліку робочого часу, включають:

- Автоматичне створення рахунків для клієнтів або замовників професіонала в залежності від витраченого часу.
- Відстеження перевитрати коштів за проектами з фіксованою вартістю.
- Пакети управління персоналом, які відстежують відвідуваність, відсутність співробітників, проблеми з кадрами, заробітну плату, управління талантами і аналітику праці.

4. Дана система була побудована у вигляді мобільного додатку за допомогою програмного середовища розробки Visual Studio Code. Для написання програмного коду використовувався фреймворк React.js на основі мови JavaScript. Крім того, більшість компонентів для взаємодії з клієнтом було побудовано за допомогою популярної бібліотеки React –Material-UI. Для збереження стану додатку використовувалась бібліотека Redux. Для написання серверної частини використовувався Express фреймворк, який прослуховується на порту 4000 та передає декілька middleware функцій, одна з яких відповідає за взаємодію з клієнтом в не залежності від домену (задаючи заголовки відповідей Access-Control-Allow-Origin) та bodyParser, шо повертає відповіді у JSON-форматі. В якості аутентифікації запитів використовується middleware функція, яка перевіряє token кожного запиту та, у випадку успіху, передає керування функції з відповідного endpoint.

5. У хмарному сховищі MongoDB Atlas всі дані зберігаються на кластері. Кластер MongoDB створюється для будь-якого основного постачальника хмарних

обчислень за нашим вибором, За допомогою MongoDB Atlas. Використовуючи призначений для користувача інтерфейс на основі браузера Atlas, також можливо інтуїтивно налаштувати кластер і контролювати його продуктивність. Зв’язок з MongoDB, що зберігає дані як документи, відбувається за допомогою Mongoose – бібліотеки JavaScript. Всі документи мають JSON-формат.

6. Додаток не містить помилок і виконує поставлене перед ним завдання щодо створення СКЧ. Але даний веб-додаток не є ідеальним, оскільки його можна покращувати, додаючи більш широкий спектр функціоналу. Гнучкість даного веб-додатку дозволяє додавати функціонал в будь-який момент часу за необхідністю.

Тому, в майбутньому, має місце додати й такий функціонал:

- Можливо додати чат між членами команди, для того щоб користувачі не відволікались на інші додатки, а також могли спілкуватися безпосередньо в одному додатку;
- Якщо користувач не зайшов у свій обліковий запис і йому прийшло запрошення на вступ до команди, можна додати повідомлення, яке відправляється в один з популярних месенджерів (наприклад, Viber чи Telegram);
- Додати функцію відслідковування кількості натиснутих клавіш, щоб покращити оцінку ефективності використання часу;

Таким чином, даний веб-додаток є дуже перспективним і його можна розроблювати й надалі, якщо того будуть потребувати користувачі. Усі поставлені завдання і цілі були успішно виконані.

7. Вимоги до веб-додатку, що розроблявся:

- Моніторинг часу, проведеного працівником за роботою.
- Створення графіків, що показують кількість напрацьованих годин та ефективність працівника за вказаний проміжок часу.
- Створення команд, в склад яких входять підлеглі та лідер, що керує командою.

8. Вимоги до програмного забезпечення:

- Операційна система Windows, macOS, Linux.

- Доступ до мережі Інтернет.
- Будь-який браузер, який підтримує JavaScript (наприклад, Google Chrome, Opera, Firefox та інші).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Александреску А. В. Современное проектирование на C++. / А. В. Александреску – Вильямс, 2014. – 336 с.

2. Антонюк В. С. Методологія наукових досліджень: [Текст] : навч. посіб./ В.С. Антонюк, Л. Г. Полонський, В. І. Аверченков, Ю. А. Малахов. – К.: НТУУ «КПІ», 2015. – 286 с.

3. Баранова Т. Б. C++. Специальный справочник / Т. Б. Баранова– Питер, 2009. – 136 с.

4. Вишневский А. В. Microsoft SQL Server. Эффективная работа/ А. В. Вишневский. – СПб.: Питер, 2009, – 541 с.

5. Герберт Ш. C# 4.0. Полное руководство [Текст] / Ш. Герберт, - М.: Издательский дом «Вильямс», 2011. - 1056 с.

6. Гешвинде. PostgreSQL рук-во разраб и администратора/ Гешвинде – ДиаСофт, 2010. – 344 с.

7. Глушков В. М., Амосов Н. М., Артеменко И. А. Энциклопедия кибернетики. Том 2. Киев, - 1974. – С. 33-54.

8. Грофф Д. SQL: Полное руководство / Д. Грофф, П. Вайнберг. – К.: BHV, 2001. – 816 с.

9. Дейт Дж. Введение в системы баз данных / Дж. Дейт.– 8-е изд. – М. : Вильямс, 2005. – 1328 с.

10. Дунаев В. В. Базы данных. Язык SQL / В. В. Дунаев. – СПб. БХВ-Петербург, 2006. – 288 с.

11. Дюба И. П. СУБД. Сборник рецептов / И. П. Дюба – Символ-Плюс, 2003. – 1056 с.

12. Інтернет-ресурс. [Електронний ресурс]. - Режим доступу: https:// timetracker.yaware.com.ua/uk.

13. Інтернет-ресурс. [Електронний ресурс]. - Режим доступу: https:// timenotes.io/features-list.

14. Інтернет-ресурс. [Електронний ресурс]. - Режим доступу: https:// metanit.com/nosql/mongodb/1.1.php.

15. Інтернет-ресурс. [Електронний ресурс]. - Режим доступу: https:// agilie.com/en/blog/how-to-make-a-time-management-and-productivity-platform.

16. Інтернет-ресурс. [Електронний ресурс]. - Режим доступу: https:// www.devteam.space/blog/how-to-build-a-time-management-app.

17. Інтернет-ресурс. [Електронний ресурс]. - Режим доступу: https:// uk.myservername.com/10-best-free-employee-timesheet-apps-2021.

18. Інтернет-ресурс. [Електронний ресурс]. - Режим доступу: https:// thepoint. rabota.ua/lovtsi-chasu-10-korysnyh-dodatkiv-dlya-efektyvnoho-taym-menedzhmentu.

19. Карпова И. П. Базы данных. Учебное пособие / И. П. Карпова – Питер, 2013. – 240 с.

20. Комашинский В. И. Смирнов Д. А. Внедрение в нейро-информационные технологии. / В. И. Комашинский, Д. А. Смирнов - СПб, 1999. – С. 33-48.

21. Кузнецов С. Д. Стандарты языка реляционных баз данных SQL: краткий обзор. // [Електронний ресурс] / С. Д. Кузнецов – Режим доступу: http://citforum. ru/ database/articles/art_2.shtml.

22. Липунцов Ю. П. Управление процессами. М: Компания АйТи, 2003. – С. 33-42.

23. Лотов А. В., Поспелова И. И. Многокритериальные задачи принятия решений: учеб.пособие. М.: МАКС Пресс, 2008. – С. 77-89.

24. Малюк А. А., Пазизин С.В., Погожин Н.С. Введение в защиту информации в автоматизированных системах. – М.: Горячая линия-Телеком, 2001. – 148 с.

25. Поспелов Г. С Искусственный интеллект - основа новой Системи автоматизації діяльності організації. // [Електронний ресурс] - Режим доступу: http://www.in-line.ru/solutions/business_appl.

26. Програма «Складський облік товарів» // [Електронний ресурс]. - Режим доступу: http://business-soft.net/uk/1c-inventory-control.html.

27. Реляционная структура данных // [Електронний ресурс]. - Режим доступу: http://citforum.ru/database/dbguide/3-1.shtml.

28. Симонович С. В. Информатика. Базовый курс: Учебник для вузов. Стандарт третьего поколения / С. В. Симонович – Питер, 2013. – 254 с.

29. Смирнова Г. Н. Проектирование экономических информационных систем: Учебник для студентов экономических вузов, обуч. по спец.: «Прикладнаяинформатика в экономике», «Прикладная информатика в менеджменте», «Прикладная информатика в юриспруденции». - М.: Финансы и статистика,2003. - 511 с.

30. Соколов В. Ю. Інформаційні системи і технології: Навчальний посібник –Київ ДУІКТ. - 2010. – С. 33-49.

31. Тарасов О. В., Лосєв М. Ю., Федько В. В. Використання мови SQL для роботи з сучасними системами керування базами даних. Харків: Вид. ХНЕУ, 2013. – 348 с.

32. Тарасов О. В., Федько В. В. Клієнт-серверні технології СУБД Oracle. Мова SQL Oracle. Навчальний посібник для самостійної підготовки студентів з навчальної дисципліни «Організація баз даних та знань». Харків: Вид. ХНЕУ ім. С. Кузнеця, 2015. – 384 с.

33. Тахагхогхи С., Вильянс Хью Е. Руководство по MySQL/ С. Тахагхогхи, Е. Вильянс Хью – Русская редакция, 2006. – 544 с.

34. Тестування ПЗ. // [Електронний ресурс]. - Режим доступу: http://mmsa. kpi.ua/files/didkovska-testing-part-ii.

35. Усков А.А. Методические указания по выполнению курсового проекта покурсу «Разработка и стандартизация программных средств иинформационных технологий». Смоленск: СФ АНО ВПО ЦС РФ «РУК»,2007. – С. 15-44.

36. Фаро С., Лерми П. Рефакторинг SQLite-приложений/ С. Фаро, П. Лерми – Симваол-Плюс, 2006. – 180 с.

37. Ясницький Л. Н. Введення в штучний інтелект. - 1-е. – Видавничий центр «Академяя», 2005. - С. 170-176.

38. API Server. // [Електронний ресурс] – Режим доступу: https://www. crossbase.de/DE/DE/loesungen/datenbereitstellung/api-server

39. ER: диаграммы сущность – связь. // [Електронний ресурс]. - Режим доступу: http://dl.sumdu.edu.ua /e-pub/db/278409/ER_Modeling.pdf.

40. JSP: JavaServerPages. // [Електронний ресурс]. - Режим доступу: http://www. javaportal.ru/java/articles/jsp.html.

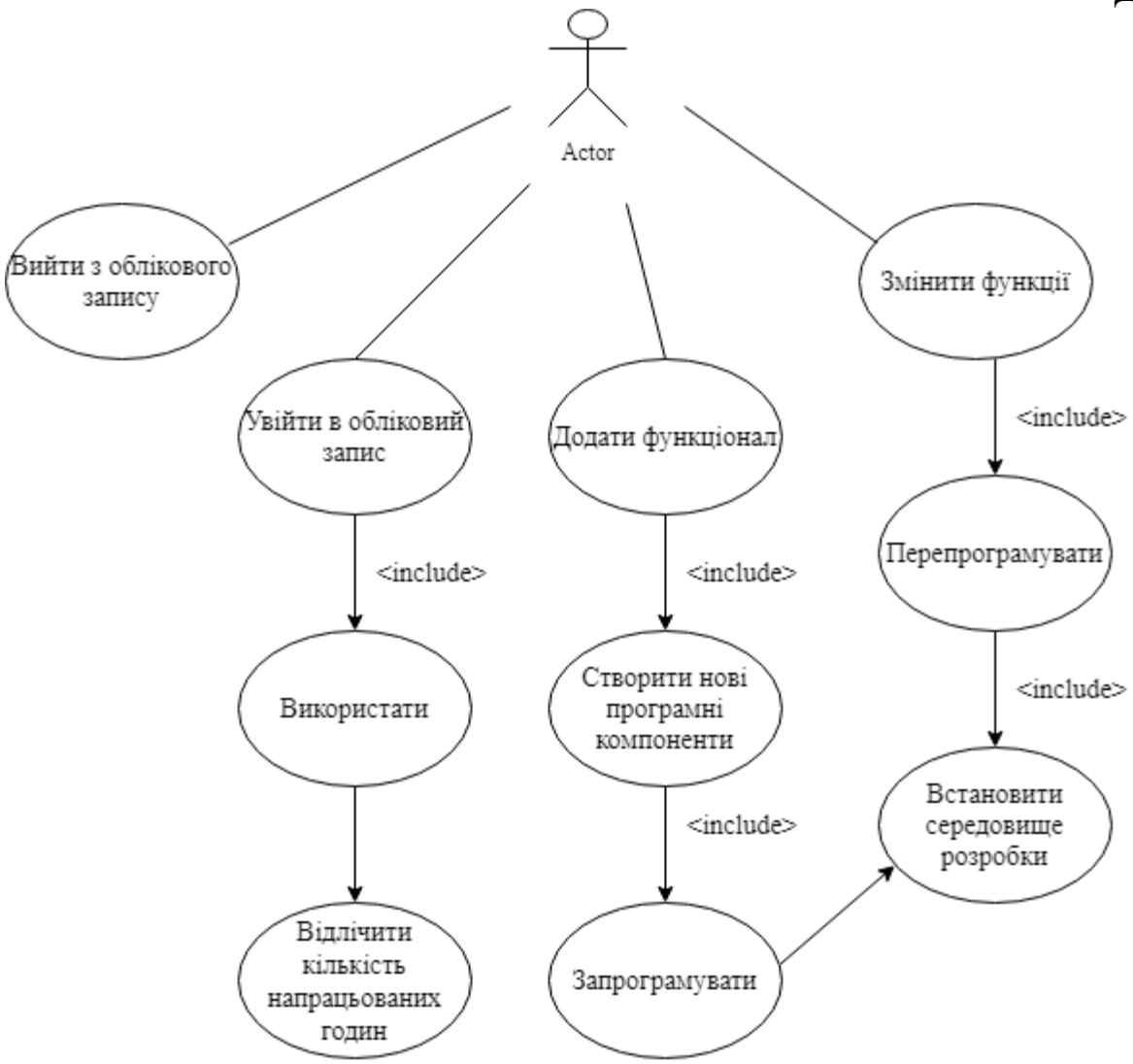
41. Moghaddam B. and Pentland A. «Probabilistic Visual Reconition for Object Recognition», Trans. IEEE Pattern Analysis and Machine Intelligence, July 1997. – P. 696–710.

42. Salamon J. A Dataset and Taxonomy for Urban Sound Research / J. Salamon, C. Jacoby, J. Bello. // 22nd ACM International Conference on Multimedia, Orlando USA. – 2014. - P. 17–44.

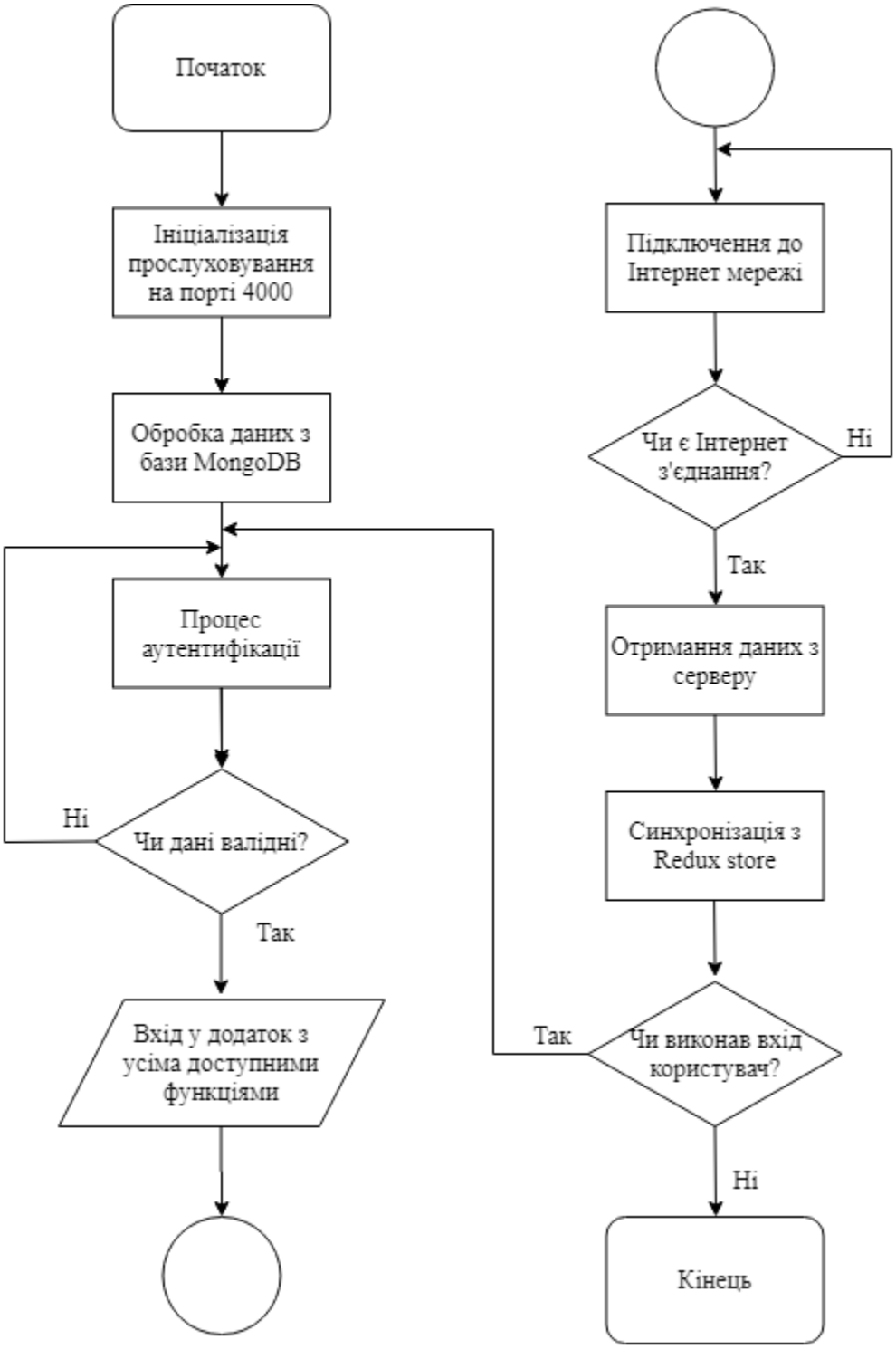
43. Richard C. Larson. Perspectives on Queues: Social Justice and the Psychology of Queueing. – INFORMS, 1987 – P. 895-905.

ДОДАТКИ

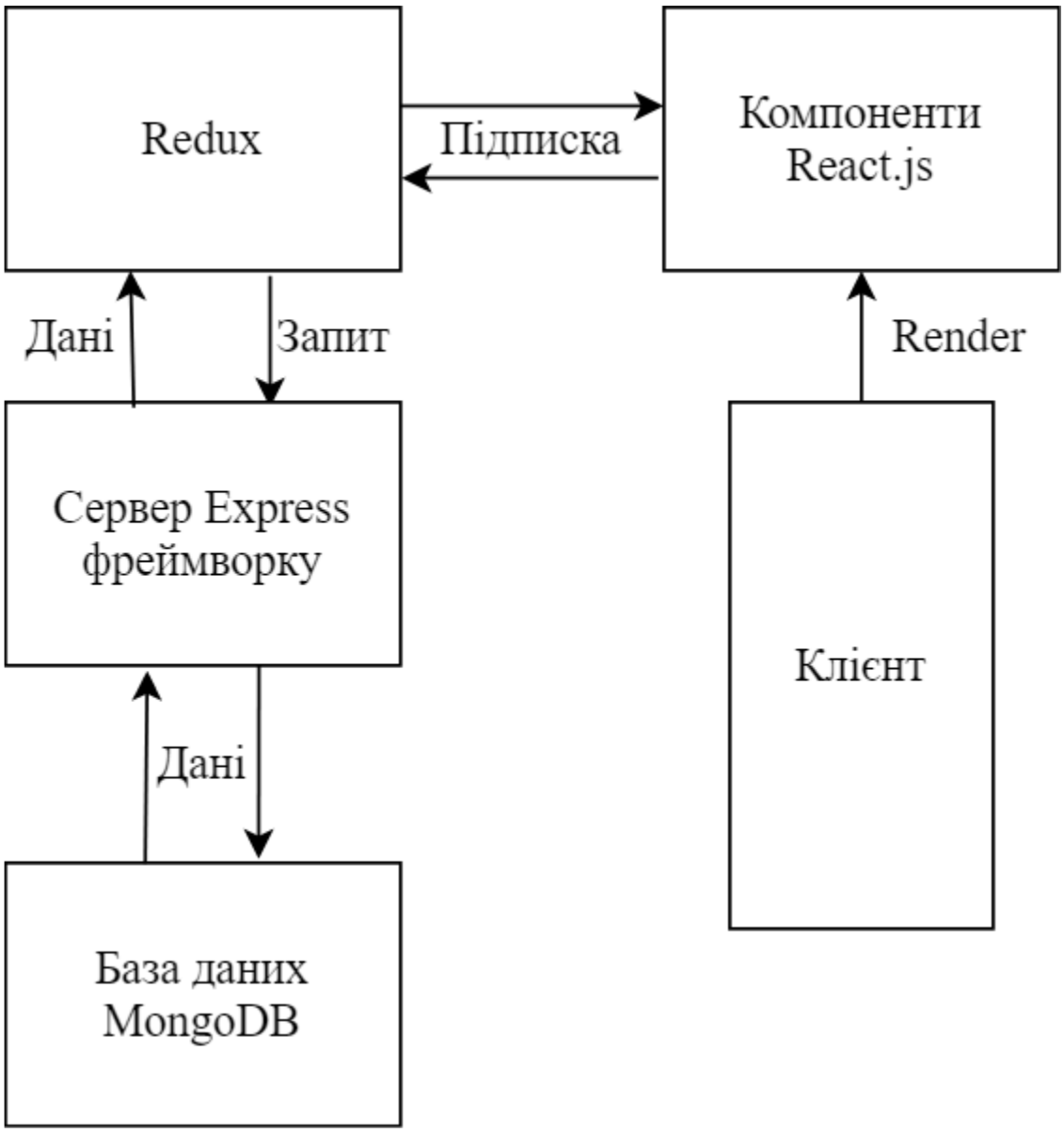
Додаток А


Actor
Вийти з облікового запису
Увійти в обліковий запис
Додати функціонал
Змінити функції
<include>
<include>
<include>
<include>
<include>
Використати
Створити нові програмні компоненти
Перепрограмувати
Встановити середовище розробки
Відлічити кількість напрацьованих годин
Запрограмувати

Додаток Б


Початок
Ініціалізація прослуховування на порті 4000
Обробка даних з бази MongoDB
Процес аутентифікації
Чи дані валідні?
Ні
Так
Вхід у додаток з усіма доступними функціями
Підключення до Інтернет мережі
Чи є Інтернет з'єднання?
Ні
Так
Отримання даних з серверу
Синхронізація з Redux store
Чи виконав вхід користувач?
Так
Ні
Кінець

Додаток В


Redux
Підписка
Компоненти
React.js
Дані
Запит
Render
Сервер Express
фреймворку
Клієнт
Дані
База даних
MongoDB