УДК 004.414;658.5.011;
UDC 004.414;658.5.011;
DOI: 10.33744/2308-6645-2023-1-55-024-030

ПРОЕКТУВАННЯ ДОДАТКУ ДЛЯ КЕРУВАННЯ ПРОЦЕСАМИ РОЗРОБКИ ПРОГРАМНИХ ПРОДУКТІВ

Безверхий О.І., доктор фізико-математичних наук, Національний транспортний університет, Київ, Україна, o_bezver@ukr.net, orcid.org/0000-0002-0834-6335
Сергієнко І.В., Національний транспортний університет, Київ, Україна, sergienko339@gmail.com, orcid.org/0009-0009-1722-2869
Шкабура О.Ю., Національний транспортний університет, Київ, Україна, ashkabura@gmail.com, orcid.org:0000-0002-3335-066X

DESIGN OF THE APPLICATION FOR MANAGEMENT OF SOFTWARE DEVELOPMENT PROCESSES

Bezverkhiy O.I., Doctor of physical and mathematical Science, National transport university, Kyiv, Ukraine, o_bezver@ukr.net, orcid.org/0000-0002-0834-6335
Sergienko I.V., National Transport University, Kyiv, Ukraine, sergienko339@gmail.com, orcid.org/0009-0009-1722-2869
Shkabura O.Yu., National Transport University, Kyiv, Ukraine, ashkabura@gmail.com, orcid.org:0000-0002-3335-066X

Вступ

На ринку програмних засобів для керування розробкою програмного забезпечення(ПЗ) існує велика кількість сервісів з схожим цільовим призначенням. Сучасні реалії процесу планування та розробки програмних продуктів вимагають від розробника взаємодії з великою їх кількістю, що суттєво впливає на продуктивність роботи та зменшує швидкість створення кінцевого продукту. Ефективність ІТ – спеціаліста визначається якістю виконаної ним роботи за визначений проміжок часу. На фактор ефективності суттєво впливає середовище роботи фахівця. В даному випадку розглядається можливість оптимізації робочого процесу програміста шляхом централізації усіх робочих аспектів у одному додатку, що дозволить ефективніше використовувати робочий час та будувати більш якісні і змістовні комунікації, пов'язані з робочим процесом.

1. Аналіз основних існуючих додатків

Під час планування процесу розробки було розглянуто наступні програмні додатки:

Slack – найпопулярніший в нас час додаток для синхронних та асинхронних комунікацій серед спеціалістів ІТ галузі. Програма має версії для всіх популярних операційних систем, веб-додаток а також додатки для мобільних операційних систем IOS та Android.

Jira – додаток для керування процесами розробки та тайм-менеджменту ІТ проектів, розроблений компанією Atlassian. Додаток дає можливість керувати процесами розробки ПЗ за двох методологій: Scrum та Kanban.

Trello – безкоштовна багато платформна система управління проектами, розроблена Trello Enterprise, дочірньою компанією Atlassian. Застосунок надає користувачам схожі до Jira можливості, проте фокус цього додатку належить стороні замовника, а не розробника (як у випадку з Jira).

Після аналізу трьох перелічених додатків було виділено такі особливості:

1. Інтерфейс додатків має доступну візуалізацію процесів, є сучасним та інтуїтивно зрозумілим.
2. Додатки містять реалізацію лише одного з основних процесів розробки ПЗ (планування, візуалізація прогресу або комунікації).
3. Декомпозиція процесів розробки потребує використання декількох з представлених додатків разом для досягнення оптимального робочого процесу.
4. Jira та Trello мають споріднений функціонал, що відноситься до планування завдань для розробки та візуалізацій прогресу виконаної роботи.
5. Slack має перевагу у реалізації концепцій комунікацій між користувачами, але йому бракує функціоналу двох попередніх додатків для забезпечення повноти робочого процесу.

Зважаючи на переваги та недоліки програм аналогів було визначено наступні функціональні вимоги до розроблюваного продукту:

- Можливість авторизації та створення профілю

- Авторизація за допомогою сервісів Google
- Можливість створення та редагування робочих середовищ (організацій)
- Можливість динамічно налаштовувати користувацькі потреби робочого середовища
- Можливість створювати та налаштовувати чати та канали робочих середовищ
- Можливість редагувати кількість користувачів робочих середовищ, каналів та чатів
- Можливість створювати та редагувати повідомлення у чатах та каналах
- Можливість реагувати на повідомлення
- Динамічні згадки у повідомленнях
- Відповіді на повідомлення
- Можливість створювати та редагувати ролі у організації
- Можливість створювати та редагувати робочі дошки та завдання для них
- Аналітичні відображення даних організацій, чатів та робочих дошок
- Персональні налаштування користувача для кожної організації
- Інформування членів організації через e-mail повідомлення

Зважаючи на визначені функціональні вимоги було визначено структуру розроблюваного додатку.

2. Проектування та розробка додатку

Додаток включає у себе клієнт-серверну частину, а також синхронізовану базу та статичну базу даних (для зберігання різного типу даних) та файлове сховище для зберігання постійних файлів.

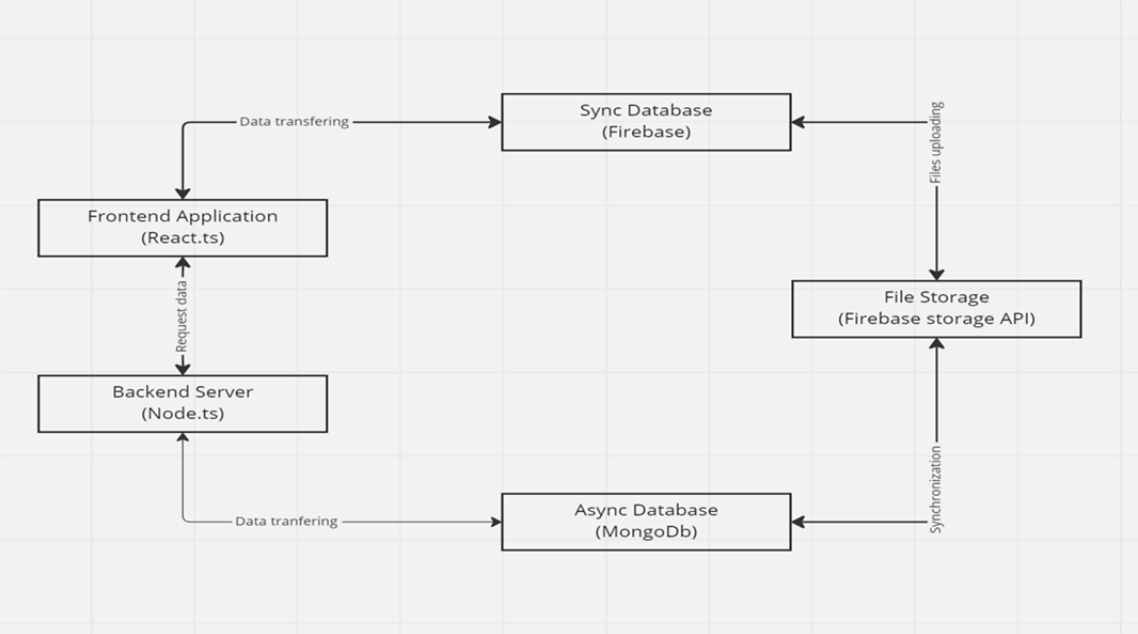

Рисунок 1 – Структура проекту
Figure 1 – Project structure

Для розробки Frontend складової додатку обрано мови програмування JavaScript та TypeScript, а також фреймворк React.js. Для Backend складової додатку було обрано мову Node.js . В якості синхронізованої системи зберігання інформації (бази даних) було обрано FireBase. Для реалізації зберігання інформації користувачів, зважаючи на концепцію проекту, використана no-sql база даних MongoBD.

При розробці автори виходили з необхідності спеціалізації баз даних, завдяки чому їм вдалося відійти від принципу «один розмір під усе». За рахунок мінімізації семантики для роботи з транзакціями з'являється можливість вирішення цілого ряду проблем, пов'язаних з нестачею продуктивності, причому горизонтальне масштабування стає простішим.

Керуючись визначеними вимогами та загальною концепцією додатку було визначено інформаційні сутності:

- Organization – організація.
- User – користувач.
- Role – роль.
- Chat, Channel – чат та канал.

- Message – повідомлення.
- Work board – робоча дошка.
- Column – колонка завдань (складова робочої дошки).
- Task – завдання (складова колонки).

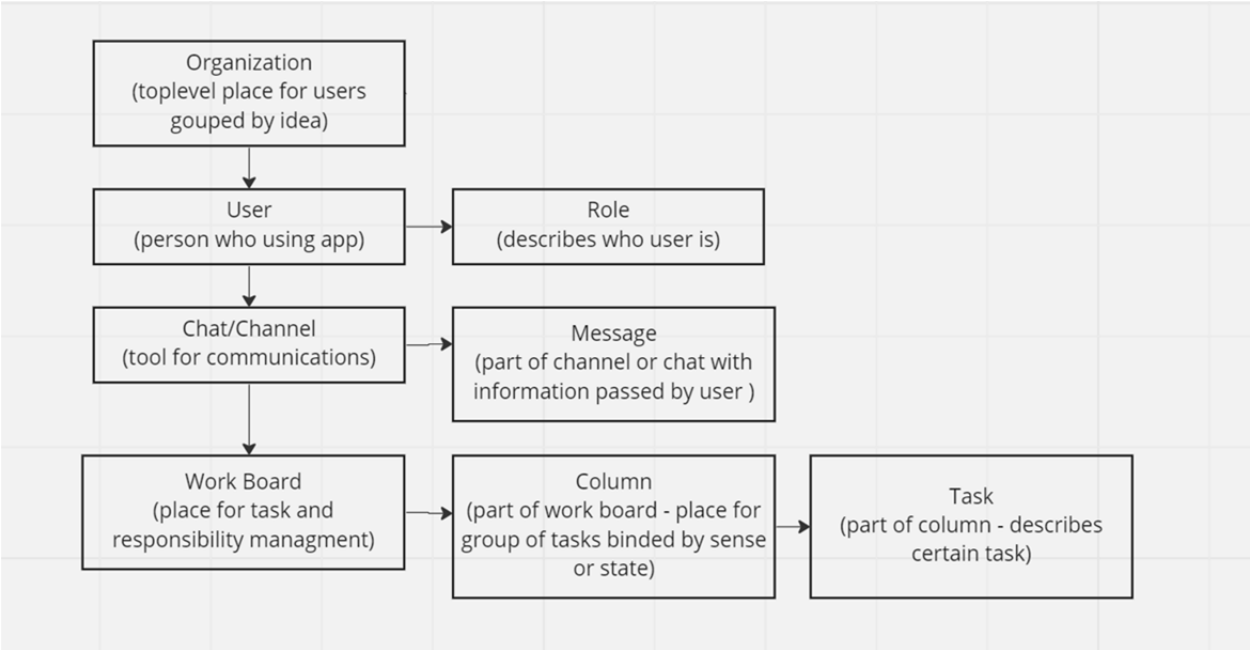

Рисунок 2 – Представлення інформаційної структури додатку
Figure 2 – Presentation of the information structure of the application

В процесі розробки проекту для зберігання статичної інформації було обрано об'єктно–орієнтовну базу даних MongoDB. Обрана база даних є статичною і буде використовуватися у проекті як середовище зберігання основної статичної інформації (такої, яка не потребує швидкісних оновлень у реальному часі). До такої інформації відносяться сутності: організація, чати та канали, ролі та користувачі. Найвищою інформаційною одиницею обраної бази даних є кластер. В рамках проектування та розробки бази даних проекту було використано такі типи даних як: String, Object, Date, ObjectId, Binary.

Концепція MongoDB передбачає роботу з даними за допомогою методів схеми. Схема являє собою екземпляр класу Mongoose.Schema, який забезпечує екземпляр методами для роботи з даними. Завдяки сумісності MongoDB з середовищем мови JavaScript та Node.JS існує можливість реалізації функціоналу бази даних за допомогою інструментів мови програмування.

Клас схеми у MongoDB містить визначені методи для роботи з даними [14]. Під час розробки сайту проекту та інтеграції його інтерфейсу з базою даних було використано наступні методи: findById, findOne, find, save.

Перераховані методи реалізували функціональну частину додатку та інтеграцію з базою даних, що дозволило зробити сайт динамічним. Також значною перевагою MongoDB є швидкість реалізації запитів та структуроване зберігання інформації (об'єкти кожної схеми зберігаються у окремій директиві кластеру).

Статичні файли програми розміщуються на хостингу Firebase. Кроссплатформні рішення Firebase Messaging дозволяють відправляти повідомлення на пристрої користувачів програми. Повідомлення можуть бути відправлені на пристрої будь-якого типу, в тому числі на ПК – як на окремі, так і на групи або на всі пристрої, на яких встановлено додаток.

Серверна логіка додатку реалізована за допомогою бібліотеки Express та мови програмування TypeScript. Важливою складовою серверної частини додатку є роутер. Роутер визначає механізми взаємодії клієнта з сервером та дозволяє обробляти кожен з запитів, даючи на нього відповідь.

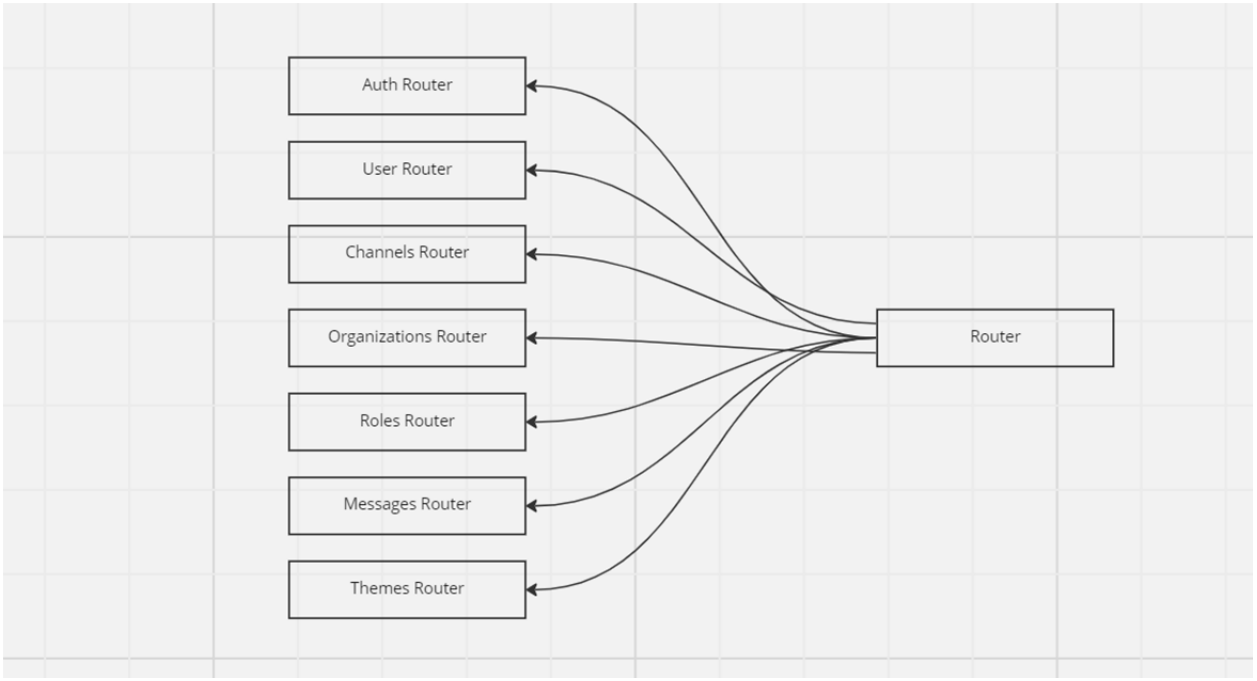

Рисунок 3 – Структура серверу додатку
Figure 3 – Application server structure

Таким чином, структура серверної логіки додатку складається з основного серверу та роутеру, що керує обміном інформації між клієнтом та сервером. Логічна структура модулів додатку визначає розподіл інформаційних ресурсів між вкладками та порядок зв'язків компонентів з серверною частиною.

Основною складовою клієнтського середовища додатку є візуальне представлення інтерфейсу. Спираючись на дані основних модулів застосунку була визначена структура інтерфейсу додатку у відповідності з призначенням кожного модуля.

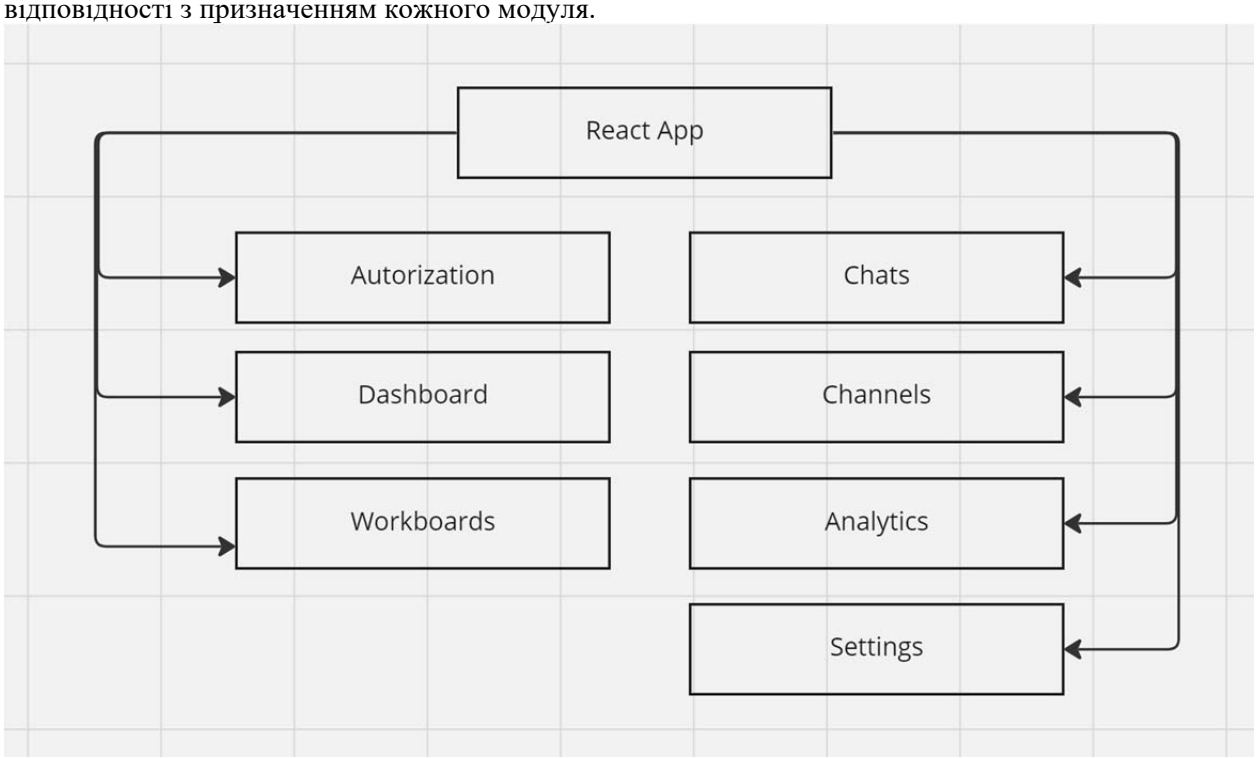

Рисунок 4 – Схема структурного розподілу модулів додатку
Figure 4 – Scheme of the structural distribution of application modules

Інтерфейс авторизації дозволяє користувачеві увійти, або зареєструватися у додатку використовуючи email та пароль, або сервіси авторизації Google. Після авторизації користувач може обрати одну з існуючих, чи створити нову організацію. Цей процес визначає подальші дії користувача. Інтерфейс дашборду дозволяє користувачеві взаємодіяти з усіма компонентами додатку та представляє основну інформацію, серед якої бокове меню з компонентами, менеджер вікон та основний контент. Вкладки Chats та Channels є спорідненими за значенням та відображають список чатів та каналів, до яких поточний користувач має доступ. Кожний чат зі списку може бути відкритим для перегляду повідомлень. Повідомлення у чатах та каналах сортуються за датою та мають функціонал реакцій. Також у чаті користувач може надіслати нове повідомлення. Вкладка з налаштуваннями представляє інтерфейс взаємодії з інформацією користувача та організації. Інформацію можна змінити, оновити та видалити. Представлення аналітичної вкладки полягає у відображені інформації з активностей у чатах та каналах, а також графічного відображення ієрархії ролей у організації. Налаштування ролей є опціональним. Користувач має можливість змінювати ролі організації тільки якщо він є її власником. Також можливість зміни кількості учасників чатів і каналів організації належить лише її власнику. Це дозволяє запобігти негативним ситуаціям у ієрархічних рішеннях. Інтерфейс робочих дошок візуалізує процес роботи над завданнями, використовуючи принципи методології Scrum. Ця методологія є кращим рішенням для невеликих організацій. Механізм взаємодії з завданнями споріднена з програмою Jira, оскільки інтерфейс її взаємодії з робочим простором визнається найзручнішим. Користувач має змогу створити або видалити колонку для завдань на робочій дошці. Також є можливість створити, видалити та оновити завдання та його статус.

Основним напрямом модульного тестування є найважливіший функціонал клієнтської частини, а саме: модулі авторизації, чатів та налаштувань. У розроблюваному програмному додатку було протестовано клієнтську частину (Frontend). Для тестування було обрано бібліотеку тестів Jest та модуль бібліотеки React – react testing library. Ці засоби дозволять ефективно протестувати додаток та виявити будь-які відхилення від очікуваної функціональної поведінки.

Висновки

Проведено аналіз можливостей та функцій розроблюваної системи. Були розглянуті системи, аналогічні до розроблюваної, а саме: Slack, Atlassian Jira, Atlassian Trello. Встановлено основні переваги та недоліки цих систем. Зокрема, відмічено, що інтерфейс додатків має доступну візуалізацію процесів, є сучасним та інтуїтивно зрозумілим, додатки містять реалізацію лише одного з основних процесів розробки ПЗ (планування, візуалізація прогресу або комунікації), декомпозиція процесів розробки потребує використання декількох з представлених додатків разом для досягнення оптимального робочого процесу. В кінці розділу було визначено технології для розробки. А саме, було обрано бази даних Firebase та MongoDb, мови програмування JavaScript, TypeScript та середовище Node.js, фреймворки React та Express.js.

Визначено концепцію баз даних проекту, а саме: розподіллено дані на динамічні та статичні, створено та проілюстровано схематично структуру бази даних, визначено кількість полів та види інформації, що будуть зберігати ці поля. Реалізовано підключення до кластеру статичної бази даних MongoDb та підключення за клієнтським посиланням до динамічної бази даних Firebase, описано принцип роботи з запитами, описано колекцію методів взаємодії з інформацією та проініціалізовано схему користувача.

Розглянуто процес створення додатку проекту. Визначено основні функціональні вимоги та реалізовано логічні модулі frontend та backend структури додатку за допомогою визначених технологій розробки. Процес розробки та структуру застосунку було зображено схематично. Було проілюстровано компоненти інтерфейсу, а також описано способи зберігання інформації у клієнтському середовищі додатку.

Було обрано модульне тестування та протестовано клієнтську частину додатку технологіями Jest та React Testing Library.

ПЕРЕЛІК ПОСИЛАНЬ

1. Орлик С. Введение в программную инженерию и управление жизненным циклом ПЗ Программная инженерия. Программные требования [Електронний ресурс] / С. Орлик, Ю. Булуй – Режим доступу до ресурсу: http://www.sorlik.ru/swebok/3software_engineering_requirements.pdf.

2. Cockburn, A. (2015). Writing Effective Use Cases. Addison-Wesley. 270p

3. Standard Glossary of terms used in Software Testing. Version 1.2, ISTQB, 2006. [Електронний ресурс]. – Режим доступу: www.istqb.org/downloads/glossary

4. 1061-1998 IEEE Standard for Software Quality Metrics Methodology. – (Галузевий стандарт)

5. Alistair Cockburn. Methodology per project. Humans and Technology Technical Report, TR 99.04, Oct.1999 7691 Dell Rd, Salt Lake City, UT 84121 USA. [Електронний ресурс]. Режим доступу: http://alistair.cockburn.us/Methodology+per+project

6. Вигерс К. Разработка требований к программному обеспечению / Пер. с анг.,М. 2004.-576с.

7. Electron Js – Офіційна Документація [Електронний ресурс] – Режим доступу до ресурсу: https://www.electronjs.org/docs.

8. Simpson K. You don't know js: this & object prototypes / Kyle Simpson., 2017, 173р

9. MongoDB Офіційна документація [Електронний ресурс] – Режим доступу до ресурсу: https://docs.mongodb.com/.

REFERENCES

1. Orlyk S. Introduction to software engineering and software life cycle management. Software engineering. Program requirements [Electronic resource] / S. Orlyk, Yu. Buluy – Mode of access to the resource: http://www.sorlik.ru/swebok/3software_engineering_requirements.pdf.

2. Cockburn, A. (2015). Writing Effective Use Cases. Addison-Wesley. 270

3. Standard Glossary of terms used in Software Testing. Version 1.2, ISTQB, 2006. [Electronic resource]. – Access mode: www.istqb.org/downloads/glossary

4. 1061-1998 IEEE Standard for Software Quality Metrics Methodology. – (Industry Standard)

5. Alistair Cockburn. Methodology per project. Humans and Technology Technical Report, TR 99.04, Oct.1999 7691 Dell Rd, Salt Lake City, UT 84121 USA. [Electronic resource]. Access mode: http://alistair.cockburn.us/Methodology+per+project

6. Vygers K. Development of requirements for software / Trans. with Eng., M. 2004.-576p.

7. Electron Js – Official Documentation [Electronic resource] – Resource access mode: https://www.electronjs.org/docs.

8. Simpson K. You don't know js: this & object prototypes / Kyle Simpson., 2017, 173

9. MongoDB Official documentation [Electronic resource] – Resource access mode: https://docs.mongodb.com/.

РЕФЕРАТ

Безверхий О.І. Проектування додатку для керування процесами розробки програмних продуктів / О.І. Безверхий, І.В Сергієнко, О.Ю Шкабура. // Вісник Національного транспортного університету. Серія «Технічні науки». Науковий журнал. – К. : НТУ, 2023. – Вип. 1 (55).

У даній статті розглядається особливості процесу планування та розробки програмних продуктів, які вимагають від розробника взаємодії з великою їх кількістю, що суттєво впливає на продуктивність роботи та зменшує швидкість створення кінцевого продукту. В даному випадку розглядається можливість оптимізації робочого процесу програміста шляхом централізації усіх робочих аспектів у одному додатку, що дозволить ефективніше використовувати робочий час та будувати більш якісні і змістовні комунікації, пов’язані з робочим процесом.

Об’єкт дослідження – процес розробки додатку для керування процесами розробки програмних продуктів. Мета роботи: проектування розробки додатку для керування процесами розробки програмних продуктів, що буде представляти цілісну систему для розробки ПЗ, мінімізуючи складність комунікацій та сприйняття інформації та дозволить суттєво пришвидшити основні процеси роботи над розроблюваним програмним забезпеченням. Проаналізувавши найпопулярніші інструменти для розробки встановлено основні переваги та недоліки цих систем. Визначено концепцію баз даних проекту, а саме: розподіллено дані на динамічні та статичні, створено та проілюстровано схематично структуру бази даних, визначено кількість полів та види інформації, що будуть зберігати ці поля. Реалізовано підключення до кластеру статичної бази даних MongoDb та підключення за клієнтським посиланням до динамічної бази даних Firebase.

КЛЮЧОВІ СЛОВА: ПРОГРАМНИЙ ПРОДУКТ, МЕНЕДЖМЕНТ ОБОВ`ЯЗКІВ, SCRUM, АСИНХРОННА КОМУНІКАЦІЯ.

ABSTRACT

Bezverhiy O.I., Sergienko I.V, Shkabura O.Yu. Design of the application for management of software development processes. Visnyk National Transport University. Series «Technical sciences». Scientific journal. – Kyiv: National Transport University, 2023. – Issue 1 (55).

This article examines the features of the process of planning and developing software products that require the developer to interact with a large number of them, which significantly affects work productivity and reduces the speed of creating the final product. In this case, the possibility of optimizing the programmer's work process by centralizing all work aspects in one application is considered, which will allow more efficient use of work time and build more qualitative and meaningful communications related to the work process.

The object of research is the process of developing an application for managing software product development processes. The purpose of the work: designing the development of an application for managing the development processes of software products, which will represent a complete system for software development, minimizing the complexity of communications and information perception, and will significantly speed up the main processes of work on the developed software. Having analyzed the most popular tools for development, the main advantages and disadvantages of these systems have been established. The concept of project databases was determined, namely: data was divided into dynamic and static, the structure of the database was created and schematically illustrated, the number of fields and the types of information that these fields will store were determined. A connection to a static MongoDb database cluster and a client link connection to a dynamic Firebase database have been implemented.

KEYWORDS: SOFTWARE PRODUCT, RESPONSIBILITY MANAGEMENT, SCRUM, ASYNCHRONOUS COMMUNICATION.

АВТОРИ

Безверхий Олександр Ігорович, д.т.н., професор, Національний транспортний університет, Київ, Україна, e-mail: o_bezver@ukr.net

Сергієнко Ігор Вадимович, Національний транспортний університет, Київ, Україна, e-mail: sergienko339@gmail.com.

Шкабура Олександр Юрійович, Національний транспортний університет, Київ, Україна, e-mail: ashkabura@gmail.com

AUTHORS

Bezverkhiy Oleksandr Ihorovych, Doctor of Technical Sciences, professor, National Transport University, Kyiv, Ukraine, e-mail: o_bezver@ukr.net

Sergienko Igor.Vadymovych National Transport University, Kyiv, Ukraine, e-mail: sergienko339@gmail.com.

Shkabura Oleksandr Yuriiovych, National Transport University, Kyiv, Ukraine, e-mail: ashkabura@gmail.com.

РЕЦЕНЗЕНТИ:

Гавриленко Валерій Володимирович., д-ф.м-н., професор, Національний транспортний університет, Київ, Україна.

Івохін Євген Віктотрович.,д-ф.м-н., професор, Національний університет імені Тараса Шевченка, Київ, Україна.

REVIEWERS:

Gavrilenko Valeriy Volodymyrovych, Doctor of Physical and Mathematical sciences, professor, National Transport University, Kyiv, Ukraine.

Ivohin Evgen Viktorovych, Doctor of Physical and Mathematical sciences, professor, National university Taras Shevchenko, Kyiv, Ukraine.