**Лабораторна робота №2**

**Використання моделей проектування**

**програмного забезпечення «Memory.pro»**

Мета: використання навичок вибору моделі проектування для конкретного проекту.

**Завдання**

1. З’ясувати, до якого типу ПО відноситься ПЗ «Memory.pro». Обґрунтувати вибір та створити схеми проектування ПЗ на першому етапі розробки.
2. На основі завдання на розробку програмного забезпечення, зробити та обґрунтувати вибір стратегії розробки та моделі життєвого циклу проекту програмного забезпечення, зробити опис моделі, вказавши переваги та недоліки.
3. Скласти загальний календарний графік виконання етапів проекту ПЗ.
4. Скласти схему етапів розробки ПЗ.
5. Розробити та обґрунтувати вимоги до ПЗ, що розробляється.
6. Створити WBS структуру (схема декомпозиції робіт).
7. Створити статут проекту згідно теми роботи (у вигляді таблиці).

**Хід роботи**

1. **Визначення типу розроблюваного ПЗ. Схеми проектування ПЗ на першому етапі**

ПЗ «Memory.pro» можна віднести до прикладного програмного забезпечення – так як це програмне забезпечення буде використовуватись для розвитку та тренування пам’яті користувачів.

Програмне забезпечення «Memory.pro» відноситься до заказного програмного забезпечення. Після завершення розробки права власності на ПЗ переходять до замовника.

Схема проектування заказного програмного забезпечення:



1. **Вибір та обґрунтування стратегії та моделі життєвого циклу розроблюваного ПЗ**

Для розробки ПЗ «Memory.pro» було вирішено використовувати гнучкі методології розробки програмного забезпечення, а саме Kanban.

Основні причини вибору Kanban:

- Гнучкість, що дозволяє швидко адаптуватись до змін вимог ПЗ.

- Простота та наглядність

- Команда розробки ПЗ «Memory.pro» досить невелика, тому масивні методології розробки програмного забезпечення не підходять (наприклад Scrum включає досить багато ритуалів призначених на підтримку продуктивності команди, але коли команда складається лише з кількох осіб в цьому немає сенсу

- Завдяки методології Kanban можна наглядно бачити прогрес задачі (наприклад, спочатку вона може знаходитись у колонці проектування, потім вона переміщається у колонку Дизайн, Розробка, тощо), що дозволяє відразу помічати, якщо у певному етапі розробки впала продуктивність тощо.

- Можливість виконувати задачі у зручній послідовності та при необхідності змінювати її.

- Класичні методології проектування ПЗ зазвичай передбачаюсь максимально детальне проектування, в той же час гнучкі методології проектування (в тому числі Kanban) дозволяють на першому этапі обійтись більш обмеженим плануванням, та доповленням його у процесі розробки

- Класичні методології проектування зазвичай розділять розробку на послідовність самостійних етапів (наприклад проектування, після його завершення дизайн, потім розробка, потім тестування тощо), але такі методології більше підходять для великих компаній, з великою кількістю проектів. В команди ПЗ «Memory.pro» всього кілька осіб, тому недопустиме виконання в один момент часу лише одного етапу (адже при цьому частині команду доведеться очікувати завершення іншого)

Agile — клас методологій розробки програмного забезпечення, що базується на ітеративній розробці, в якій вимоги та розв'язки еволюціонують через співпрацю між багатофункціональними командами здатними до самоорганізації. Гнучка розробка — засіб для підвищення продуктивності розробників програмного забезпечення.

Більшість гнучких методологій націлені на мінімізацію ризиків, шляхом зведення розробки до серії коротких циклів, що мають назву ітерацій, які зазвичай тривають один-два тижні. Кожна ітерація сама по собі виглядає як програмний проект в мініатюрі, і включає всі завдання, необхідні для видачі мінімального приросту за функціональністю: планування, аналіз вимог, проектування, кодування, тестування і документування. Хоча окрема ітерація, як правило, недостатня для випуску нової версії продукту, мається на увазі те, що гнучкий програмний проект готовий до випуску наприкінці кожної ітерації. Після закінчення кожної ітерації, команда виконує переоцінку пріоритетів розробки.

Agile акцентує увагу на безпосередньому спілкуванні «віч-на-віч». Більшість agile команд розташовані в одному офісі. Як мінімум вона включає і «замовників» (замовники, які визначають продукт, також це можуть бути менеджери продукту, бізнес аналітики або клієнти). Офіс може також включати тестувальників, дизайнерів інтерфейсу, технічних авторів і менеджерів.

Основною метрикою agile методів є робочий продукт. Віддаючи перевагу безпосередньому спілкуванню, agile-методи зменшують обсяг письмової документації в порівнянні з іншими методами.

Основні ідеї:

* Особистості та їхні взаємодії важливіші, ніж процеси та інструменти;
* Робоче програмне забезпечення важливіше, ніж повна документація;
* Співпраця із замовником важливіша, ніж контрактні зобов'язання;
* Реакція на зміни важливіша, ніж дотримання плану.

Принципи:

* Задоволення клієнта за рахунок ранньої та безперебійної поставки програмного забезпечення;
* Вітання змін вимог навіть наприкінці розробки (це може підвищити конкурентоспроможність отриманого продукту);
* Часта поставка робочого програмного забезпечення (кожен місяць або тиждень або ще частіше);
* Тісне, щоденне спілкування замовника з розробниками впродовж всього проекту;
* Проектом займаються мотивовані особистості, які забезпечені потрібними умовами роботи, підтримкою і довірою;
* Рекомендований метод передачі інформації — особиста розмова (віч-на-віч);
* Робоче програмне забезпечення — найкращий вимірювач прогресу;
* Спонсори, розробники та користувачі повинні мати можливість підтримувати постійний темп на невизначений термін;
* Постійна увага поліпшенню технічної досконалості та зручному дизайну;
* Простота — мистецтво не робити зайвої роботи;
* Найкращі технічні вимоги, дизайн та архітектура виходять у самоорганізованої команди;
* Постійна адаптація до мінливих обставин.

Найпопулярнішими agile методологіями є:

* Scrum
* XP
* Kanban
* Lean Software Development
* Тощо

Канбан — гнучка методологія розробки програмного забезпечення заснована на візуалізації процесу розробки на різних етапах (планування, дизайн, розробка тощо).

Основна ідея Канбан – відображення процесу розробки візуально у вигляди Канбан дошки. У найпростішому випадку вона може складатись з 3 колонок: що треба зробити, що виконується зараз, та що виконано.

Звичайно у реальних проектах Канбан дошки значно складніші. На них можуть відображатись усі етапи розробки певних задач (наприклад, при реалізації певної функціональності, вона поміщається в колонку дизайн, коли дизайнер завершив роботу він переносить її у колонку «розробка», розробник безпосередньо реалізує задачу та переносить її у колонку QA і тд.)

Також важливим моментом є встановлення лімітів на максимальну кількість задач в одній колонці.

Завдяки цьому можна наглядно бачити якщо в якійсь частині робочого процесу проблеми з продуктивністю (в тій колонці буде забагато задач).

Слово «канбан» походить з японської перекладається як «вивіска» чи «білборд». Він використовує обмеження роботи, що знаходиться в процесі виконання, як ключовий механізм для виявлення проблем в роботі системи та стимулює співпрацю для постійного її вдосконалення

Основні принципи:

* Почніть з того, що ви маєте зараз

Метод Канбан не описує конкретний набір ролей чи кроків процесу. Він стартує з ролями і процесами, що є у вас зараз, і стимулює постійні інкрементні та еволюційні зміни в системі.

* Погодьтесь домагатись інкрементних, еволюційних змін

Команда повинна погодитись, що постійні, інкрементальні зміни — це спосіб покращити систему, і зробити так, щоб ці покращення прижились. Глобальні зміни можуть виглядати більш ефективними, але мають більший ризик провалу, через опір та страх змін в організації. Канбан заохочує постійні невеликі зміни до поточної системи.

Ключові практик методу Канбан

* Візуалізуйте

Візуалізація процесів роботи допомагає в правильному розумінні змін, що плануються і допомагає впроваджувати їх згідно з планом.

Типовим способом візуалізувати процес роботи є використання дошки з колонками і картками. Колонки на дошці позначають різні кроки процесу роботи.

* Обмежуйте задачі в процесі виконання
* Керуйте потоком

Кожен перехід між станами в потоці моніториться, вимірюється і звітується. Активне управління потоком дозволяє оцінити позитивні та негативні ефекти змін у системі.

* Зробіть політики явними

Поки механізм чи процес не стане явним, часто важко чи неможливо здійснювати обговорення щодо його вдосконалення. Без явного розуміння, як все працює, будь-які обговорення проблем стають емоційними та суб'єктивними. З явним розумінням можливо перейти до більш раціональних, емпіричних та об'єктивних обговорень проблем.

* Створіть цикли зворотнього зв'язку

Організації що не створили другий рівень зворотнього зв'язку — перегляд операцій, — зазвичай не бачать вдосконалення процесу поза локалізованим рівнем команди.

* Вдосконалюйте співпрацюючи, розвивайтесь експериментально (використовуючи моделі та науковий метод)

Метод Канбан пропагує малі поступові, постійні та еволюційні зміни які приживаються. Коли команди мають спільне розуміння теорій про роботу, процес, ризики, вони більш ймовірно будуть здатними виробити спільне розуміння проблем та запропонувати вдосконалення які будуть результатом консенсусу.

Метод Канбан радить використовувати науковий підхід до втілення змін.

1. **Загальний календарний графік виконання етапів проектування програмного забезпечення**

|  |  |  |  |
| --- | --- | --- | --- |
| Період | | Тривалість (днів) | Етап розробки ПЗ |
| Початок | Кінець |
| 10.09.19 | 25.09.19 | 15 | Визначення вимог |
| 26.09.19 | 10.10.19 | 15 | Специфікація вимог |
| 11.10.19 | 11.11.19 | 30 | Проектування та дизайн |
| 12.11.19 | 2.12.19 | 20 | Кодування |
| 3.12.19 | 15.12.19 | 12 | Тестування |
| 16.12.19 | 25.12.19 | 9 | Введення в експлуатацію |

1. **Розробка та обґрунтування вимог до розроблюваного ПЗ**

Необхідно розробити сервіс для тренування та розвитку пам’яті «Memory.pro».

Користувачі «Memory.pro» можуть вибрати тип тренування, тренуватись, отримати результати, та бачити статистику – свою, та загальну.

Таким чином система повинна забезпечувати можливості:

* Реєстрації
* Авторизації
* Виходу з системи

ПЗ «Memory.pro» буде реалізовано у вигляді веб сервісу, що буде складатись з двох частин: REST API та Frontend. Таким чином, для системи авторизації має застосовуватись стандарт авторизації OAuth2

Основні можливості що повинен забезпечувати сервісу:

* Тренування за вибраним типом тренування
* Отримання та збереження результатів
* Ведення статистики користувача
* Відображення загальної статистики

Усі дані для тренувань додаються у систему адміністратором. Тому система повинна давати можливість для додавання, редагування та видалення даних для тренувань.

Основним користувачем системи є User. Усі користувачі характеризуються наступними параметрами (параметри позначенні «\*» є обов’язковими):

* Id \* – унікальний ідентифікатор користувача, дається користувачу при реєстрації та не може бути зміненим
* Ім’я користувача \*
* Прізвище користувача \*
* Username \* – унікальний псевдонім користувача
* E-mail \*
* Дата реєстрації \*
* Роль \* - роль користувача (user, administrator etc)
* Дата народження
* Avatar – назва зображення користувача
* Країна
* Дата й час останньої активності
* Рейтинг користувача
* Кількість інформації що запам’ятав користувач

Кожен користувач має мати особистий кабінет, я кому зможе бачити свої дані, змінювати їх та бачити свою статистику. Також інші користувачі мають мати можливість бачити кабінети інших користувачів. Особиста інформації при цьому має бути прихованою.

Незареєстровані користувачі також повинні мати можливість користуватись сервісом та тренуватись у основних типах тренувань. Але при цьому статистика для таких користувачів не ведеться. Деякі типи тренувань можуть бути доступними лише для зареєстрованих користувачів.

Кожне тренування має складатись з наступних етапів:

1. Вибір типу тренування та його настроювання (кількість елементів, швидкість переходу до наступного елементу тощо).
2. Запам’ятовування
3. Завдання для відволікання (користувач має вирішити кілька простих математичних виразів, може бути не обов’язковим)
4. Згадування
5. Отримання результатів

Відповідно в базі даних крім даних для запам’ятовування повиті також зберігатись результати користувачів.

Результати користувачів зберігаються у відповідних таблицях у базі даних (окремі таблиці для кожного типу тренувань).

Кожен результат має характеризуватись наступними атрибутами:

* Id користувача
* Id результату
* Дата початку тренування
* Час запам’ятовування
* Час згадування
* Оцінка
* Дані тренування (запам’ятовуванні дані, час на запам’ятовування кожного елементу, правильність відповіді користувача). Дані тренувань мають бути представлені у таблиці «Дані тренувань»

Деякі типи тренувань можуть містити додаткові атрибути.

Якщо тип тренування не передбачаю іншого, оцінювання тренування відбувається за наступною формулою:

Користувачі повинні мати можливість бачити статистику – свою так загальну, усіх користувачів. Також користувач має мати можливість бачити загальну кількість інформації, що він запам’ятав та загальну кількість інформації запам’ятовану усіма користувачами.

Користувач повинен мати можливість бачити рейтингову таблицю користувачів побудовану на основі рейтингу.

При настроюванні тренування користувач має мати можливість застосовувати «Шаблони показу» елементів даних. Шаблон показу – це текстовий рядок, що складається з символів X, що представляють елементи даних, та інших символів.

1. **Схема декомпозиції робіт (WBS структура проекту)**



1. **Статут проекту**

|  |  |
| --- | --- |
| 1. Назва проекту | Сервіс «Memory.pro» |
| 1. Стейкхолдери проекту | Замовник: Гашко Дмитро Андрійович |
| 1. Місія та цілі проекту | Розробка сервісу для тренування та розвитку пам’яті користувачів, що зможуть завдяки сервісу:   1. Тренувати пам’ять завдяки тренуванням з запам’ятовування різноманітних простих типів інформації (числа, карти, слова тощо) 2. Готуватись до тренувань з запам’ятовування 3. Аналіз свого прогресу 4. Покращення уяви 5. Загальний розвиток розумових здібностей |
| 1. Діло обставини та бізнес-задачі | Даний проект призначений тренування та розвитку пам’яті користувачів ПЗ «Memory.pro». Продуктом розробки буде web сервіс для тренування пам’яті, що буде складатись з двох частин – memory frontend та memory.pro api. |
| * 1. Оточення проекту | Проект буде розроблятись в компанії розробника. Замовник бере участь у розробці вимог, при приймальних роботах, тестуванні та при виявлені нових вимог на потреб. |
| 1. Фінансові показники проекту | 1. Junior Fullstack Developer – 1100$ за місяць роботи 2. Junior QA Engineer – 700$ за місяць роботи; 3. Designer – 700$ за місяць роботи 4. Product Manager – 500$ за місяць роботи   Витрати на розробку ПЗ за весь строк розробки: (1100$ + 700$ + 700$ + 500$) \* 4 = 12000$ за 4 місяці |
| 1. Технічні вимоги до проекту | Розроблюване ПЗ призначене для виконання будь-яких пристроях, на яких присутній сучасних браузер. Підтримувані браузери – усі браузери та версії, за останні 4 роки (Internet Explorer не підтримується).  Основними вимогами є:   * Реєстрація користувачів * Тренування * Формування та збереження результатів * Ведення статистики |
| 1. Ризики проекту та засоби боротьби з ними | * Найбільшими ризиками при розробці ПЗ при використані Agile методологій є ризик недостатнього проектування через що розширення ПЗ може стати досить складним. Для мінімізації цього ризику на проектування буде виділено досить велика частина часу. * При тимчасовій не змозі будь-якого члена команди виконувати свої зобов’язання виконання проекту може бути повністю призупинене (так як команда проекту досить мала). Тому необхідно планувати роботу так, щоб у учасників були задачі на випадок відсутності інших. А також треба врахувати можливість хвороб при плануванні періоду розробки. |

**Висновок:** виконавши лабораторну роботу, я визначив тип розроблюваного ПЗ, описав та обґрунтував стратегію та модель розробки ПЗ, що найкраще всього підходять для ПЗ «Memory.pro», на основі опису предметної області сформував вимоги до ПЗ, спланував за допомогою схем план роботи над проектом, а також створив календарний графік робіт, розробив WBS-структуру проекту (схему декомпозиції робіт) та статут проекту.