МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

«УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМ

**ПРАКТИЧНА РОБОТА №3**

Тема: «**ПОБУДОВА ФУНКЦІОНАЛЬНОЇ СХЕМИ СИСТЕМИ ПЗ**»

Виконав:

Студент 3-го курсу

факультету інформаційних технологій

Заочної форми навчання

Групи ІПЗ

Шевчук Назарій Володимирович

Ужгород 2025

**Мета:** проведення функціональної декомпозиції розв'язуваної задачі; побудова функціональної схеми.

**Теоретичні відомості**

Проектування програмного забезпечення часто починається з функціональної декомпозиції розв'язуваної задачі. Функціональна декомпозиція задачі являє собою ієрархічне розбиття складної задачі на ряд простіше вирішуються невеликих завдань, які, в свою чергу, поділяються на підзадачі до тих пір, поки кожна необхідна деталь в ній не буде визначена досить ясно. Концепція ієрархічної декомпозиції настільки природна, що ми не завжди в змозі усвідомити, як часто нам доводиться використовувати її на практиці. Вона випливає з людської потреби мати справу з цілком визначеним числом дискретних джерел інформації, що піддається управлінню і проводити «відсікання» інформації до тих пір, поки число дискретних джерел не стане приблизно рівним семи.

Сувора ієрархічна декомпозиція підпорядковується правилам:

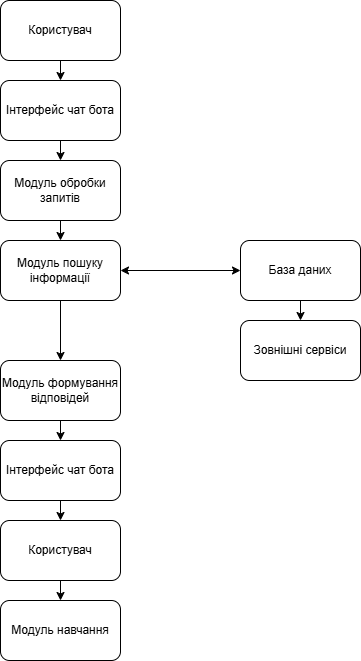
1. На кожному рівні ієрархії задача повинна мати закінчений вигляд на даному рівні деталізації;

2. На будь-якому рівні ієрархії кожне розбиття повністю охоплює окреме завдання (функцію), яка відповідає певному рівню деталізації.

**Практична частина**

Під час виконання лабораторної роботи необхідно провести функціональну декомпозицію розв'язуваної задачі, побудувати відповідну схему. Робота повинна бути оформлена у вигляді специфікації, що містить функціональну схему розв'язуваної задачі.

**Хід роботи:**



Інтерфейс чат-бота відповідає за прийом текстових запитів від користувача та відображення відповідей чат-бота, забезпечуючи взаємодію між користувачем та системою. Модуль обробки запитів аналізує текстовий запит користувача, визначає його намір та витягує ключові слова, необхідні для пошуку інформації. Модуль пошуку інформації шукає відповідну інформацію в базі даних або запитує інформацію з зовнішніх сервісів, використовуючи результати аналізу запиту. База даних зберігає інформацію, необхідну для відповідей чат-бота, та надає доступ до неї за запитом модуля пошуку інформації. Модуль формування відповідей формує відповідь на запит користувача на основі знайденої інформації та форматує її для відображення в інтерфейсі чат-бота. Модуль навчання збирає дані про запити користувачів та відповіді чат-бота, аналізує їх для покращення відповідей чат-бота та оновлює базу даних та алгоритми обробки запитів.

1. Методи проєктування програмного забезпечення:

1. Об'єктно-орієнтоване проєктування (ООП):
   1. Використання ООП дозволяє створити модульну та розширювану структуру бота.
   2. Наприклад, можна створити класи для обробки різних типів запитів, клас для роботи з базою даних, клас для взаємодії з API Telegram.

2. Висхідне і низхідне проєктування програмного забезпечення:

1. Низхідне проєктування:
   1. На початковому етапі можна використовувати низхідне проєктування для визначення основних функцій бота та його архітектури.
   2. Наприклад, спочатку визначити, що бот повинен вміти відповідати на запитання, шукати інформацію та виконувати певні завдання.
2. Висхідне проєктування:
   1. На етапі реалізації можна використовувати висхідне проєктування для розробки окремих компонентів бота, таких як модулі обробки запитів, пошуку інформації та формування відповідей.
   2. Ці компоненти потім об'єднуються в більші модулі.

3. Об'єктно-орієнтоване проєктування програмного забезпечення:

1. Для телеграм чат-бота ООП є дуже ефективним методом.
2. Можна створити класи:
   1. TelegramBot - для взаємодії з API Telegram.
   2. MessageHandler - для обробки повідомлень від користувачів.
   3. DatabaseManager - для роботи з базою даних.
   4. SearchEngine - для пошуку інформації.
3. Використання принципів ООП, таких як інкапсуляція, спадкування та поліморфізм, дозволяє створити гнучку та розширювану структуру бота.

4. Функціональна декомпозиція системи програмного забезпечення:

1. Функціональна декомпозиція дозволяє розбити бота на менші, більш керовані функціональні модулі.
2. Наприклад:
   1. Модуль обробки запитів: аналізує запити користувачів та визначає намір.
   2. Модуль пошуку інформації: шукає відповідну інформацію в базі даних або зовнішніх джерелах.
   3. Модуль формування відповідей: формує відповіді на запити користувачів.
   4. Модуль роботи з базою данних: зберігання та обробка інформації.
3. Такий підхід спрощує розробку, тестування та підтримку бота.

5. Надійність програмного забезпечення:

1. Надійність телеграм чат-бота є важливим аспектом.
2. Для забезпечення надійності необхідно:
   1. Ретельно тестувати бота на всіх етапах розробки.
   2. Використовувати методи обробки помилок.
   3. Забезпечити стабільну роботу бази даних.
   4. Контроль за роботою API telegram.
3. Використання журналювання (логінг) допомагає відслідковувати помилки та збої.

**Висновки**

На цій лабораторній роботі я здобув практичні навички у проведенні функціональної декомпозиції, що є важливим етапом у розробці будь-якого програмного забезпечення. У процесі виконання завдання була побудована детальна функціональна схема, яка чітко відображає структуру та взаємодію компонентів системи.

Ця схема дозволила мені краще зрозуміти логіку роботи програми, визначити ключові функціональні блоки та їх взаємозв'язки. Проведена функціональна декомпозиція виявилася надзвичайно корисним інструментом, оскільки вона дозволяє розбити складну задачу на більш прості підзадачі, що значно полегшує процес розробки та робить його більш систематизованим. Такий підхід не лише спрощує розуміння архітектури програми, але й підвищує її модульність та підтримуваність.