Київський національний університет будівництва і архітектури Кафедра інформаційних технологій	Шифр спеціальності	Назва спеціальності, освітньої програми	Освітній рівень
	122	Комп'ютерні науки, Комп'ютерні науки	Магістр

«Затверджую»
Завідувачка кафедри
Тетяна ГОНЧАРЕНКО /_____/

Розробники силабуса
Світлана БІЛОЩИЦЬКА /_____/

Володимир ВАЦКЕЛЬ /_____/

СИЛАБУС

ВК "Інструментальні засоби розробки програмного забезпечення"

1) Статус освітньої компоненти: вибіркова	
2)Контактні дані викладачів:д.т.н, доцент Білощицька С.В., biloshchytska.sv@knuba.edu.ua , +380 (93) 825 35 93, https://www.knuba.edu.ua/elementor-161588/) асистент кафедри ІТ, Вацкель Володимир Юрійович, vatskel.vy@knuba.edu.ua, +380(67)3201013	
3) Пререквізити: «Дискретна математика», «Засоби керування якістю процесу розробки програмного забезпечення», «Об'єктно-орієнтоване програмування», «Internet-технології та мова програмування Java», «Системне програмування».	
4)) Коротка анотація дисципліни Дисципліна присвячена вивченню сучасних інструментів та середовищ для розробки програмного забезпечення. Здобувачі ознайомляться з різноманітними інструментами, які використовуються на всіх етапах життєвого циклу програмного забезпечення: від проектування, написання та налагодження коду до тестування, інтеграції, версійного контролю та деплойменту. Особлива увага приділяється інструментам автоматизації, засобам для управління командами розробників та організації робочих процесів. Курс спрямований на формування практичних навичок використання інтегрованих середовищ розробки (IDE), систем контролю версій (Git), засобів для автоматизації збірки і тестування, а також на підвищення ефективності розробки програмного забезпечення за допомогою сучасних інструментальних засобів.	
5) Структура курсу: лекції, лабораторні роботи, самостійні роботи, РГР, залік	
Загальна кількість кредитів ECTS	3,0
Сума годин:	90
Вид індивідуального завдання	РГР
Форма контролю	Залік

6) Зміст курсу:

1. Структура навчальної дисципліни

Змістовий модуль 1. Інструментальні засоби розробки програмного забезпечення

Лекція 1-2. *І*нструментальні засоби розробки програмних систем

1.1. Основні поняття визначення інструментальних засобів розробки

1.2. Історія розвитку інструментальних засобів

2.1. Переваги використання інструментальних засобів

2.2. Виклики та проблеми при виборі інструментальних засобів

Лекція 3. Інтегровані середовища розробки

3.1. Основні характеристики IDE

3.2. Переваги використання IDE для розробки на C #

3.3. Огляд популярних IDE для розробки на C #

3.4. Рекомендації щодо вибору IDE

Лекція 4. Налагодження програмного забезпечення

4.1. Налагодження у Visual Studio

4.2. Налагодження C # програм

4.3. Кращі практики налагодження

Змістовий модуль 2. Системи контролю версій та співпраця в команді

Лекція 5-6. Системи контролю версій

5.1. Основи систем контролю версій

5.2. Введення в Git

5.3. Введення в Subversion

6.1. Робота з репозиторіями

6.2. Розгалуження та злиття

6.3. Кращі практики роботи з системами контролю версій

Лекція 7. Система контролю версій: співпраця в команді

7.1. Central Workflow

7.2. Git-flow

7.3. Trunk-based Development

7.4. Feature Branch Workflow

7.5. Forking Workflow

Змістовий модуль 3. Автоматизація, DevOps і тестування

Лекція 8. Засоби автоматизації збірки проєктів

8.1. Основні концепції автоматизації збірки

8.2. Інструменти автоматизації збірки

8.3. Використання інструментів автоматизації збірки

Лекція 9. DevOps

9.1. Основні компоненти DevOps

9.2. Інструменти DevOps

9.3. Практичні аспекти впровадження DevOps

9.4. Виклики та майбутнє DevOps

Лекція 10. Інструменти для інтеграційного тестування

10.1.Інструменти для системного тестування

10.2. Впровадження автоматизованого тестування у процес неперервної інтеграції

10.3. Кращі практики автоматизованого тестування

2. Теми лабораторних занять

1. Розробка ідеї програмного продукту
2. Створення та налаштування проєкту в Visual Studio
3. Налагодження програмного забезпечення
4. Системи контролю версій
5. Системи контролю версій співпраця в команді
6. Засоби автоматизації збірки проєктів
7. Тестування програмного забезпечення
8. DevOps

3. Самостійна робота

1. Опрацювання лекційного матеріалу
2. Підготовка до практичних занять
3. Опрацювання тем, винесених на самостійну підготовку, в т.ч. конспектування за заданим планом
4. Робота з інтернет-ресурсами
5. Написання та оформлення РГР
6. Підготовка до заліку

4. Індивідуальні завдання

Розрахунково-графічна робота (РГР) спрямована на практичне застосування набутих знань з освітньої компоненти. На виконання РГР відводиться 12 годин самостійної роботи.

Приклади можливих тем для виконання РГР:

1. Проектування архітектури програмного забезпечення: побудова UML-діаграм для заданого проєкту.
2. Використання системи керування версіями Git для організації командної розробки.
3. Розробка ER-діаграми бази даних для заданої предметної області.
4. Реалізація CRUD-операцій у базі даних за допомогою SQL-запитів.
5. Проєктування архітектури програмного забезпечення із застосуванням патернів проєктування.
6. Розробка тест-кейсів для модульного тестування програмного забезпечення.
7. Автоматизація тестування програмного забезпечення з використанням Selenium або JUnit.
8. Налаштування CI/CD-процесу для програмного проєкту за допомогою Jenkins або GitHub Actions.
9. Використання інструментів статичного аналізу коду для оцінки якості програмного забезпечення.
10. Розробка прототипу графічного інтерфейсу користувача (GUI) з використанням Figma або іншого інструменту.
11. Побудова та аналіз Gantt-діаграми для управління IT-проєктом.
12. Моделювання процесів розробки програмного забезпечення за допомогою BPMN-нотації.
13. Інтеграція зовнішніх API у програмний проєкт (наприклад, використання REST API).

14. Розробка програмного модуля для аналізу великих даних із використанням бібліотек Python (Pandas, NumPy).
15. Побудова та аналіз діаграми класів для об’єктно-орієнтованого програмного забезпечення.
16. Застосування методології Scrum: створення backlog і спринтів для IT-проєкту.
17. Оптимізація продуктивності програмного забезпечення з використанням профайлерів (наприклад, VisualVM).
18. Розробка сценаріїв автоматизації розгортання програмного забезпечення за допомогою Docker.
19. Тестування продуктивності програмного забезпечення за допомогою JMeter.
20. Аналіз вимог до програмного забезпечення та побудова специфікацій у форматі SRS (Software Requirements Specification).

7) Посилання на сторінку електронного навчально-методичного комплексу дисципліни: https://org2.knuba.edu.ua/course/view.php?id=3690